



# Red Hat Quay 3.11

## Red Hat Quay API Guide

Red Hat Quay API Guide





## Legal Notice

Copyright © 2024 Red Hat, Inc.

The text of and illustrations in this document are licensed by Red Hat under a Creative Commons Attribution–Share Alike 3.0 Unported license ("CC-BY-SA"). An explanation of CC-BY-SA is available at

<http://creativecommons.org/licenses/by-sa/3.0/>

. In accordance with CC-BY-SA, if you distribute this document or an adaptation of it, you must provide the URL for the original version.

Red Hat, as the licensor of this document, waives the right to enforce, and agrees not to assert, Section 4d of CC-BY-SA to the fullest extent permitted by applicable law.

Red Hat, Red Hat Enterprise Linux, the Shadowman logo, the Red Hat logo, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

Linux<sup>®</sup> is the registered trademark of Linus Torvalds in the United States and other countries.

Java<sup>®</sup> is a registered trademark of Oracle and/or its affiliates.

XFS<sup>®</sup> is a trademark of Silicon Graphics International Corp. or its subsidiaries in the United States and/or other countries.

MySQL<sup>®</sup> is a registered trademark of MySQL AB in the United States, the European Union and other countries.

Node.js<sup>®</sup> is an official trademark of Joyent. Red Hat is not formally related to or endorsed by the official Joyent Node.js open source or commercial project.

The OpenStack<sup>®</sup> Word Mark and OpenStack logo are either registered trademarks/service marks or trademarks/service marks of the OpenStack Foundation, in the United States and other countries and are used with the OpenStack Foundation's permission. We are not affiliated with, endorsed or sponsored by the OpenStack Foundation, or the OpenStack community.

All other trademarks are the property of their respective owners.

## Abstract

Use the Red Hat Quay API

# Table of Contents

<b>PREFACE</b> .....	<b>18</b>
<b>CHAPTER 1. USING THE RED HAT QUAY API</b> .....	<b>19</b>
1.1. ACCESSING THE QUAY API FROM QUAY.IO	19
1.2. CREATING AN OAUTH ACCESS TOKEN	19
1.3. ACCESSING YOUR QUAY API FROM A WEB BROWSER	21
1.4. ACCESSING THE RED HAT QUAY API FROM THE COMMAND LINE	21
1.4.1. Get superuser information	21
1.4.2. Creating a superuser using the API	22
1.4.3. List usage logs	23
1.4.3.1. Example for pagination	23
1.4.4. Directory synchronization	26
1.4.5. Create a repository build via API	26
1.4.6. Create an org robot	27
1.4.7. Trigger a build	27
1.4.8. Create a private repository	27
1.4.9. Create a mirrored repository	27
<b>CHAPTER 2. RED HAT QUAY APPLICATION PROGRAMMING INTERFACE (API)</b> .....	<b>29</b>
2.1. AUTHORIZATION	29
Scopes	29
2.2. APPSPECIFICTOKENS	29
2.2.1. createAppToken	29
POST /api/v1/user/apptoken	29
Request body schema (application/json)	30
Responses	30
2.2.2. listAppTokens	30
GET /api/v1/user/apptoken	30
Query parameters	30
Responses	30
2.2.3. getAppToken	31
GET /api/v1/user/apptoken/{token_uuid}	31
Path parameters	31
Responses	31
2.2.4. revokeAppToken	31
DELETE /api/v1/user/apptoken/{token_uuid}	31
Path parameters	31
Responses	32
2.3. BUILD	32
2.3.1. getRepoBuildStatus	32
GET /api/v1/repository/{repository}/build/{build_uuid}/status	32
Path parameters	32
Responses	32
2.3.2. getRepoBuildLogs	33
GET /api/v1/repository/{repository}/build/{build_uuid}/logs	33
Path parameters	33
Responses	33
2.3.3. getRepoBuild	33
GET /api/v1/repository/{repository}/build/{build_uuid}	33
Path parameters	33
Responses	33

2.3.4. cancelRepoBuild	34
DELETE /api/v1/repository/{repository}/build/{build_uuid}	34
Path parameters	34
Responses	34
2.3.5. requestRepoBuild	34
POST /api/v1/repository/{repository}/build/	35
Path parameters	35
Request body schema (application/json)	35
Responses	35
2.3.6. getRepoBuilds	36
GET /api/v1/repository/{repository}/build/	36
Path parameters	36
Query parameters	36
Responses	36
2.4. DISCOVERY	36
2.4.1. discovery	37
GET /api/v1/discovery	37
Query parameters	37
Responses	37
2.5. ERROR	37
2.5.1. getErrorDescription	37
GET /api/v1/error/{error_type}	37
Path parameters	37
Responses	37
2.6. GLOBALMESSAGES	38
2.6.1. createGlobalMessage	38
POST /api/v1/messages	38
Request body schema (application/json)	38
Responses	38
2.6.2. getGlobalMessages	39
GET /api/v1/messages	39
Responses	39
2.6.3. deleteGlobalMessage	39
DELETE /api/v1/message/{uuid}	39
Path parameters	39
Responses	39
2.7. LOGS	40
2.7.1. getAggregateUserLogs	40
GET /api/v1/user/aggregatelogs	40
Query parameters	40
Responses	40
2.7.2. exportUserLogs	40
POST /api/v1/user/exportlogs	40
Query parameters	40
Request body schema (application/json)	41
Responses	41
2.7.3. listUserLogs	41
GET /api/v1/user/logs	41
Query parameters	41
Responses	42
2.7.4. getAggregateOrgLogs	42
GET /api/v1/organization/{orgname}/aggregatelogs	42
Path parameters	42

---

Query parameters	42
Responses	43
2.7.5. exportOrgLogs	43
POST /api/v1/organization/{orgname}/exportlogs	43
Path parameters	43
Query parameters	43
Request body schema (application/json)	44
Responses	44
2.7.6. listOrgLogs	44
GET /api/v1/organization/{orgname}/logs	44
Path parameters	44
Query parameters	44
Responses	45
2.7.7. getAggregateRepoLogs	45
GET /api/v1/repository/{repository}/aggregatelogs	45
Path parameters	45
Query parameters	45
Responses	45
2.7.8. exportRepoLogs	46
POST /api/v1/repository/{repository}/exportlogs	46
Path parameters	46
Query parameters	46
Request body schema (application/json)	46
Responses	47
2.7.9. listRepoLogs	47
GET /api/v1/repository/{repository}/logs	47
Path parameters	47
Query parameters	47
Responses	47
2.8. MANIFEST	48
2.8.1. getManifestLabel	48
GET /api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}	48
Path parameters	48
Responses	48
2.8.2. deleteManifestLabel	49
DELETE /api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}	49
Path parameters	49
Responses	49
2.8.3. addManifestLabel	49
POST /api/v1/repository/{repository}/manifest/{manifestref}/labels	49
Path parameters	49
Request body schema (application/json)	50
Responses	50
2.8.4. listManifestLabels	50
GET /api/v1/repository/{repository}/manifest/{manifestref}/labels	50
Path parameters	50
Query parameters	50
Responses	51
2.8.5. getRepoManifest	51
GET /api/v1/repository/{repository}/manifest/{manifestref}	51
Path parameters	51
Responses	51
2.9. MIRROR	52

---

2.9.1. syncCancel	52
POST /api/v1/repository/{repository}/mirror/sync-cancel	52
Path parameters	52
Responses	52
2.9.2. syncNow	52
POST /api/v1/repository/{repository}/mirror/sync-now	52
Path parameters	52
Responses	53
2.9.3. getRepoMirrorConfig	53
GET /api/v1/repository/{repository}/mirror	53
Path parameters	53
Responses	53
2.9.4. changeRepoMirrorConfig	53
PUT /api/v1/repository/{repository}/mirror	53
Path parameters	54
Request body schema (application/json)	54
Responses	54
2.9.5. createRepoMirrorConfig	55
POST /api/v1/repository/{repository}/mirror	55
Path parameters	55
Request body schema (application/json)	55
Responses	56
2.10. NAMESPACEQUOTA	56
2.10.1. listUserQuota	56
GET /api/v1/user/quota	56
Responses	56
2.10.2. getOrganizationQuotaLimit	57
GET /api/v1/organization/{orgname}/quota/{quota_id}/limit/{limit_id}	57
Path parameters	57
Responses	57
2.10.3. changeOrganizationQuotaLimit	57
PUT /api/v1/organization/{orgname}/quota/{quota_id}/limit/{limit_id}	57
Path parameters	57
Request body schema (application/json)	58
Responses	58
2.10.4. deleteOrganizationQuotaLimit	58
DELETE /api/v1/organization/{orgname}/quota/{quota_id}/limit/{limit_id}	58
Path parameters	58
Responses	59
2.10.5. createOrganizationQuotaLimit	59
POST /api/v1/organization/{orgname}/quota/{quota_id}/limit	59
Path parameters	59
Request body schema (application/json)	59
Responses	60
2.10.6. listOrganizationQuotaLimit	60
GET /api/v1/organization/{orgname}/quota/{quota_id}/limit	60
Path parameters	60
Responses	60
2.10.7. getUserQuotaLimit	61
GET /api/v1/user/quota/{quota_id}/limit/{limit_id}	61
Path parameters	61
Responses	61
2.10.8. listUserQuotaLimit	61



---

GET /api/v1/user/quota/{quota_id}/limit	61
Path parameters	61
Responses	62
2.10.9. getOrganizationQuota	62
GET /api/v1/organization/{orgname}/quota/{quota_id}	62
Path parameters	62
Responses	62
2.10.10. changeOrganizationQuota	63
PUT /api/v1/organization/{orgname}/quota/{quota_id}	63
Path parameters	63
Request body schema (application/json)	63
Responses	63
2.10.11. deleteOrganizationQuota	63
DELETE /api/v1/organization/{orgname}/quota/{quota_id}	63
Path parameters	64
Responses	64
2.10.12. createOrganizationQuota	64
POST /api/v1/organization/{orgname}/quota	64
Path parameters	64
Request body schema (application/json)	64
Responses	64
2.10.13. listOrganizationQuota	65
GET /api/v1/organization/{orgname}/quota	65
Path parameters	65
Responses	65
2.10.14. getUserQuota	65
GET /api/v1/user/quota/{quota_id}	65
Path parameters	65
Responses	66
2.11. ORGANIZATION	66
2.11.1. createOrganization	66
POST /api/v1/organization/	66
Request body schema (application/json)	66
Responses	66
2.11.2. validateProxyCacheConfig	67
POST /api/v1/organization/{orgname}/validateproxycache	67
Path parameters	67
Request body schema (application/json)	67
Responses	67
2.11.3. getOrganizationCollaborators	68
GET /api/v1/organization/{orgname}/collaborators	68
Path parameters	68
Responses	68
2.11.4. getOrganizationApplication	68
GET /api/v1/organization/{orgname}/applications/{client_id}	68
Path parameters	68
Responses	68
2.11.5. updateOrganizationApplication	69
PUT /api/v1/organization/{orgname}/applications/{client_id}	69
Path parameters	69
Request body schema (application/json)	69
Responses	70
2.11.6. deleteOrganizationApplication	70

---

DELETE /api/v1/organization/{orgname}/applications/{client_id}	70
Path parameters	70
Responses	70
2.11.7. createOrganizationApplication	71
POST /api/v1/organization/{orgname}/applications	71
Path parameters	71
Request body schema (application/json)	71
Responses	71
2.11.8. getOrganizationApplications	72
GET /api/v1/organization/{orgname}/applications	72
Path parameters	72
Responses	72
2.11.9. getProxyCacheConfig	72
GET /api/v1/organization/{orgname}/proxycache	72
Path parameters	72
Responses	73
2.11.10. deleteProxyCacheConfig	73
DELETE /api/v1/organization/{orgname}/proxycache	73
Path parameters	73
Responses	73
2.11.11. createProxyCacheConfig	74
POST /api/v1/organization/{orgname}/proxycache	74
Path parameters	74
Request body schema (application/json)	74
Responses	74
2.11.12. getOrganizationMember	74
GET /api/v1/organization/{orgname}/members/{membername}	74
Path parameters	75
Responses	75
2.11.13. removeOrganizationMember	75
DELETE /api/v1/organization/{orgname}/members/{membername}	75
Path parameters	75
Responses	75
2.11.14. getOrganizationMembers	76
GET /api/v1/organization/{orgname}/members	76
Path parameters	76
Responses	76
2.11.15. getOrganization	76
GET /api/v1/organization/{orgname}	76
Path parameters	76
Responses	77
2.11.16. changeOrganizationDetails	77
PUT /api/v1/organization/{orgname}	77
Path parameters	77
Request body schema (application/json)	77
Responses	78
2.11.17. deleteAdmindedOrganization	78
DELETE /api/v1/organization/{orgname}	78
Path parameters	78
Responses	78
2.11.18. getApplicationInformation	79
GET /api/v1/app/{client_id}	79
Path parameters	79

Responses	79
2.12. PERMISSION	79
2.12.1. getUserTransitivePermission	79
GET /api/v1/repository/{repository}/permissions/user/{username}/transitive	79
Path parameters	79
Responses	79
2.12.2. getUserPermissions	80
GET /api/v1/repository/{repository}/permissions/user/{username}	80
Path parameters	80
Responses	80
2.12.3. changeUserPermissions	80
PUT /api/v1/repository/{repository}/permissions/user/{username}	81
Path parameters	81
Request body schema (application/json)	81
Responses	81
2.12.4. deleteUserPermissions	81
DELETE /api/v1/repository/{repository}/permissions/user/{username}	81
Path parameters	81
Responses	82
2.12.5. getTeamPermissions	82
GET /api/v1/repository/{repository}/permissions/team/{teamname}	82
Path parameters	82
Responses	82
2.12.6. changeTeamPermissions	83
PUT /api/v1/repository/{repository}/permissions/team/{teamname}	83
Path parameters	83
Request body schema (application/json)	83
Responses	83
2.12.7. deleteTeamPermissions	83
DELETE /api/v1/repository/{repository}/permissions/team/{teamname}	84
Path parameters	84
Responses	84
2.12.8. listRepoTeamPermissions	84
GET /api/v1/repository/{repository}/permissions/team/	84
Path parameters	84
Responses	84
2.12.9. listRepoUserPermissions	85
GET /api/v1/repository/{repository}/permissions/user/	85
Path parameters	85
Responses	85
2.13. POLICY	85
2.13.1. createOrganizationAutoPrunePolicy	85
POST /api/v1/organization/{orgname}/autoprunepolicy/	85
Path parameters	86
Request body schema (application/json)	86
Responses	86
2.13.2. listOrganizationAutoPrunePolicies	86
GET /api/v1/organization/{orgname}/autoprunepolicy/	86
Path parameters	86
Responses	86
2.13.3. getOrganizationAutoPrunePolicy	87
GET /api/v1/organization/{orgname}/autoprunepolicy/{policy_uuid}	87
Path parameters	87

Responses	87
2.13.4. deleteOrganizationAutoPrunePolicy	87
DELETE /api/v1/organization/{orgname}/autoprunepolicy/{policy_uuid}	88
Path parameters	88
Responses	88
2.13.5. updateOrganizationAutoPrunePolicy	88
PUT /api/v1/organization/{orgname}/autoprunepolicy/{policy_uuid}	88
Path parameters	88
Request body schema (application/json)	88
Responses	89
2.13.6. createRepositoryAutoPrunePolicy	89
POST /api/v1/repository/{repository}/autoprunepolicy/	89
Path parameters	89
Request body schema (application/json)	89
Responses	90
2.13.7. listRepositoryAutoPrunePolicies	90
GET /api/v1/repository/{repository}/autoprunepolicy/	90
Path parameters	90
Responses	90
2.13.8. getRepositoryAutoPrunePolicy	90
GET /api/v1/repository/{repository}/autoprunepolicy/{policy_uuid}	90
Path parameters	91
Responses	91
2.13.9. deleteRepositoryAutoPrunePolicy	91
DELETE /api/v1/repository/{repository}/autoprunepolicy/{policy_uuid}	91
Path parameters	91
Responses	91
2.13.10. updateRepositoryAutoPrunePolicy	92
PUT /api/v1/repository/{repository}/autoprunepolicy/{policy_uuid}	92
Path parameters	92
Request body schema (application/json)	92
Responses	92
2.13.11. createUserAutoPrunePolicy	93
POST /api/v1/user/autoprunepolicy/	93
Request body schema (application/json)	93
Responses	93
2.13.12. listUserAutoPrunePolicies	93
GET /api/v1/user/autoprunepolicy/	93
Responses	93
2.13.13. getUserAutoPrunePolicy	94
GET /api/v1/user/autoprunepolicy/{policy_uuid}	94
Path parameters	94
Responses	94
2.13.14. deleteUserAutoPrunePolicy	94
DELETE /api/v1/user/autoprunepolicy/{policy_uuid}	94
Path parameters	95
Responses	95
2.13.15. updateUserAutoPrunePolicy	95
PUT /api/v1/user/autoprunepolicy/{policy_uuid}	95
Path parameters	95
Request body schema (application/json)	95
Responses	95
2.14. PROTOTYPE	96

---

2.14.1. updateOrganizationPrototypePermission	96
PUT /api/v1/organization/{orgname}/prototypes/{prototypeid}	96
Path parameters	96
Request body schema (application/json)	96
Responses	96
2.14.2. deleteOrganizationPrototypePermission	97
DELETE /api/v1/organization/{orgname}/prototypes/{prototypeid}	97
Path parameters	97
Responses	97
2.14.3. createOrganizationPrototypePermission	97
POST /api/v1/organization/{orgname}/prototypes	97
Path parameters	98
Request body schema (application/json)	98
Responses	98
2.14.4. getOrganizationPrototypePermissions	98
GET /api/v1/organization/{orgname}/prototypes	98
Path parameters	98
Responses	99
2.15. REPOSITORY	99
2.15.1. createRepo	99
POST /api/v1/repository	99
Request body schema (application/json)	99
Responses	99
2.15.2. listRepos	100
GET /api/v1/repository	100
Query parameters	100
Responses	100
2.15.3. changeRepoVisibility	101
POST /api/v1/repository/{repository}/changevisibility	101
Path parameters	101
Request body schema (application/json)	101
Responses	101
2.15.4. changeRepoState	102
PUT /api/v1/repository/{repository}/changestate	102
Path parameters	102
Request body schema (application/json)	102
Responses	102
2.15.5. getRepo	102
GET /api/v1/repository/{repository}	102
Path parameters	103
Query parameters	103
Responses	103
2.15.6. updateRepo	103
PUT /api/v1/repository/{repository}	103
Path parameters	103
Request body schema (application/json)	103
Responses	104
2.15.7. deleteRepository	104
DELETE /api/v1/repository/{repository}	104
Path parameters	104
Responses	104
2.16. REPOSITORYNOTIFICATION	105
2.16.1. testRepoNotification	105

---

POST /api/v1/repository/{repository}/notification/{uuid}/test	105
Path parameters	105
Responses	105
2.16.2. getRepoNotification	105
GET /api/v1/repository/{repository}/notification/{uuid}	105
Path parameters	105
Responses	106
2.16.3. deleteRepoNotification	106
DELETE /api/v1/repository/{repository}/notification/{uuid}	106
Path parameters	106
Responses	106
2.16.4. resetRepositoryNotificationFailures	107
POST /api/v1/repository/{repository}/notification/{uuid}	107
Path parameters	107
Responses	107
2.16.5. createRepoNotification	107
POST /api/v1/repository/{repository}/notification/	107
Path parameters	107
Request body schema (application/json)	108
Responses	108
2.16.6. listRepoNotifications	108
GET /api/v1/repository/{repository}/notification/	108
Path parameters	108
Responses	109
2.17. REPOTOKEN	109
2.17.1. getTokens	109
GET /api/v1/repository/{repository}/tokens/{code}	109
Path parameters	109
Responses	109
2.17.2. changeToken	110
PUT /api/v1/repository/{repository}/tokens/{code}	110
Path parameters	110
Request body schema (application/json)	110
Responses	110
2.17.3. deleteToken	110
DELETE /api/v1/repository/{repository}/tokens/{code}	110
Path parameters	110
Responses	111
2.17.4. createToken	111
POST /api/v1/repository/{repository}/tokens/	111
Path parameters	111
Request body schema (application/json)	111
Responses	111
2.17.5. listRepoTokens	112
GET /api/v1/repository/{repository}/tokens/	112
Path parameters	112
Responses	112
2.18. ROBOT	112
2.18.1. getUserRobots	112
GET /api/v1/user/robots	113
Query parameters	113
Responses	113
2.18.2. getOrgRobotPermissions	113

---

GET /api/v1/organization/{orgname}/robots/{robot_shortname}/permissions	113
Path parameters	113
Responses	113
2.18.3. regenerateOrgRobotToken	114
POST /api/v1/organization/{orgname}/robots/{robot_shortname}/regenerate	114
Path parameters	114
Responses	114
2.18.4. getUserRobotPermissions	114
GET /api/v1/user/robots/{robot_shortname}/permissions	115
Path parameters	115
Responses	115
2.18.5. regenerateUserRobotToken	115
POST /api/v1/user/robots/{robot_shortname}/regenerate	115
Path parameters	115
Responses	115
2.18.6. getOrgRobot	116
GET /api/v1/organization/{orgname}/robots/{robot_shortname}	116
Path parameters	116
Responses	116
2.18.7. createOrgRobot	116
PUT /api/v1/organization/{orgname}/robots/{robot_shortname}	116
Path parameters	116
Request body schema (application/json)	117
Responses	117
2.18.8. deleteOrgRobot	117
DELETE /api/v1/organization/{orgname}/robots/{robot_shortname}	117
Path parameters	117
Responses	118
2.18.9. getOrgRobots	118
GET /api/v1/organization/{orgname}/robots	118
Path parameters	118
Query parameters	118
Responses	119
2.18.10. getUserRobot	119
GET /api/v1/user/robots/{robot_shortname}	119
Path parameters	119
Responses	119
2.18.11. createUserRobot	120
PUT /api/v1/user/robots/{robot_shortname}	120
Path parameters	120
Request body schema (application/json)	120
Responses	120
2.18.12. deleteUserRobot	120
DELETE /api/v1/user/robots/{robot_shortname}	120
Path parameters	120
Responses	121
2.19. SEARCH	121
2.19.1. conductRepoSearch	121
GET /api/v1/find/repositories	121
Query parameters	121
Responses	121
2.19.2. conductSearch	122
GET /api/v1/find/all	122

---

Query parameters	122
Responses	122
2.19.3. getMatchingEntities	122
GET /api/v1/entities/{prefix}	122
Path parameters	123
Query parameters	123
Responses	123
2.20. SECSCAN	123
2.20.1. getRepoManifestSecurity	123
GET /api/v1/repository/{repository}/manifest/{manifestref}/security	123
Path parameters	123
Query parameters	124
Responses	124
2.21. SUPERUSER	124
2.21.1. createInstallUser	124
POST /api/v1/superuser/users/	124
Request body schema (application/json)	124
Responses	125
2.21.2. listAllUsers	125
GET /api/v1/superuser/users/	125
Query parameters	125
Responses	125
2.21.3. listAllLogs	126
GET /api/v1/superuser/logs	126
Query parameters	126
Responses	126
2.21.4. createServiceKey	126
POST /api/v1/superuser/keys	126
Request body schema (application/json)	126
Responses	127
2.21.5. listServiceKeys	127
GET /api/v1/superuser/keys	127
Responses	127
2.21.6. changeUserQuotaSuperUser	128
PUT /api/v1/superuser/organization/{namespace}/quota/{quota_id}	128
Path parameters	128
Request body schema (application/json)	128
Responses	128
2.21.7. deleteUserQuotaSuperUser	128
DELETE /api/v1/superuser/organization/{namespace}/quota/{quota_id}	128
Path parameters	128
Responses	129
2.21.8. createUserQuotaSuperUser	129
POST /api/v1/superuser/organization/{namespace}/quota	129
Path parameters	129
Request body schema (application/json)	129
Responses	129
2.21.9. listUserQuotaSuperUser	130
GET /api/v1/superuser/organization/{namespace}/quota	130
Path parameters	130
Responses	130
2.21.10. changeOrganizationQuotaSuperUser	130
PUT /api/v1/superuser/users/{namespace}/quota/{quota_id}	130



Path parameters	130
Request body schema (application/json)	131
Responses	131
2.21.11. deleteOrganizationQuotaSuperUser	131
DELETE /api/v1/superuser/users/{namespace}/quota/{quota_id}	131
Path parameters	131
Responses	131
2.21.12. createOrganizationQuotaSuperUser	132
POST /api/v1/superuser/users/{namespace}/quota	132
Path parameters	132
Request body schema (application/json)	132
Responses	132
2.21.13. listOrganizationQuotaSuperUser	133
GET /api/v1/superuser/users/{namespace}/quota	133
Path parameters	133
Responses	133
2.21.14. changeOrganization	133
PUT /api/v1/superuser/organizations/{name}	133
Path parameters	133
Request body schema (application/json)	133
Responses	134
2.21.15. deleteOrganization	134
DELETE /api/v1/superuser/organizations/{name}	134
Path parameters	134
Responses	134
2.21.16. approveServiceKey	135
POST /api/v1/superuser/approvedkeys/{kid}	135
Path parameters	135
Request body schema (application/json)	135
Responses	135
2.21.17. deleteServiceKey	135
DELETE /api/v1/superuser/keys/{kid}	135
Path parameters	136
Responses	136
2.21.18. updateServiceKey	136
PUT /api/v1/superuser/keys/{kid}	136
Path parameters	136
Request body schema (application/json)	136
Responses	136
2.21.19. getServiceKey	137
GET /api/v1/superuser/keys/{kid}	137
Path parameters	137
Responses	137
2.21.20. getRepoBuildStatusSuperUser	137
GET /api/v1/superuser/{build_uuid}/status	137
Path parameters	137
Responses	138
2.21.21. getRepoBuildSuperUser	138
GET /api/v1/superuser/{build_uuid}/build	138
Path parameters	138
Responses	138
2.21.22. getRepoBuildLogsSuperUser	139
GET /api/v1/superuser/{build_uuid}/logs	139

Path parameters	139
Responses	139
2.21.23. getRegistrySize	139
GET /api/v1/superuser/registrysize/	139
Path parameters	139
Responses	140
2.21.24. postRegistrySize	140
POST /api/v1/superuser/registrysize/	140
Path parameters	140
Request body schema (application/json)	140
Responses	140
2.22. TAG	141
2.22.1. restoreTag	141
POST /api/v1/repository/{repository}/tag/{tag}/restore	141
Path parameters	141
Request body schema (application/json)	141
Responses	141
2.22.2. changeTag	142
PUT /api/v1/repository/{repository}/tag/{tag}	142
Path parameters	142
Request body schema (application/json)	142
Responses	142
2.22.3. deleteFullTag	143
DELETE /api/v1/repository/{repository}/tag/{tag}	143
Path parameters	143
Responses	143
2.22.4. listRepoTags	143
GET /api/v1/repository/{repository}/tag/	143
Path parameters	143
Query parameters	144
Responses	144
2.23. TEAM	144
2.23.1. getOrganizationTeamPermissions	144
GET /api/v1/organization/{orgname}/team/{teamname}/permissions	145
Path parameters	145
Responses	145
2.23.2. updateOrganizationTeamMember	145
PUT /api/v1/organization/{orgname}/team/{teamname}/members/{membername}	145
Path parameters	145
Responses	145
2.23.3. deleteOrganizationTeamMember	146
DELETE /api/v1/organization/{orgname}/team/{teamname}/members/{membername}	146
Path parameters	146
Responses	146
2.23.4. getOrganizationTeamMembers	147
GET /api/v1/organization/{orgname}/team/{teamname}/members	147
Path parameters	147
Query parameters	147
Responses	147
2.23.5. inviteTeamMemberEmail	147
PUT /api/v1/organization/{orgname}/team/{teamname}/invite/{email}	148
Path parameters	148
Responses	148

---

2.23.6. deleteTeamMemberEmailInvite	148
DELETE /api/v1/organization/{orgname}/team/{teamname}/invite/{email}	148
Path parameters	148
Responses	149
2.23.7. updateOrganizationTeam	149
PUT /api/v1/organization/{orgname}/team/{teamname}	149
Path parameters	149
Request body schema (application/json)	149
Responses	149
2.23.8. deleteOrganizationTeam	150
DELETE /api/v1/organization/{orgname}/team/{teamname}	150
Path parameters	150
Responses	150
2.24. TRIGGER	150
2.24.1. activateBuildTrigger	150
POST /api/v1/repository/{repository}/trigger/{trigger_uuid}/activate	151
Path parameters	151
Request body schema (application/json)	151
Responses	151
2.24.2. listTriggerRecentBuilds	151
GET /api/v1/repository/{repository}/trigger/{trigger_uuid}/builds	151
Path parameters	151
Query parameters	152
Responses	152
2.24.3. manuallyStartBuildTrigger	152
POST /api/v1/repository/{repository}/trigger/{trigger_uuid}/start	152
Path parameters	152
Request body schema (application/json)	152
Responses	153
2.24.4. getBuildTrigger	153
GET /api/v1/repository/{repository}/trigger/{trigger_uuid}	153
Path parameters	153
Responses	153
2.24.5. updateBuildTrigger	154
PUT /api/v1/repository/{repository}/trigger/{trigger_uuid}	154
Path parameters	154
Request body schema (application/json)	154
Responses	154
2.24.6. deleteBuildTrigger	155
DELETE /api/v1/repository/{repository}/trigger/{trigger_uuid}	155
Path parameters	155
Responses	155
2.24.7. listBuildTriggers	155
GET /api/v1/repository/{repository}/trigger/	155
Path parameters	155
Responses	156
2.25. USER	156
2.25.1. createStar	156
POST /api/v1/user/starred	156
Request body schema (application/json)	156
Responses	156
2.25.2. listStarredRepos	157
GET /api/v1/user/starred	157

---

Query parameters	157
Responses	157
2.25.3. getLoggedInUser	157
GET /api/v1/user/	157
Responses	157
2.25.4. deleteStar	158
DELETE /api/v1/user/starred/{repository}	158
Path parameters	158
Responses	158
2.25.5. getUserInformation	158
GET /api/v1/users/{username}	158
Path parameters	159
Responses	159
2.26. DEFINITIONS	159
2.26.1. ApiError	159
2.26.2. UserView	159
2.26.3. ViewMirrorConfig	160
2.26.4. ApiErrorDescription	161
<b>CHAPTER 3. API CONFIGURATION EXAMPLES .....</b>	<b>162</b>
3.1. EXTERNAL_REGISTRY_CONFIG OBJECT REFERENCE	162
3.2. RULE_RULE OBJECT REFERENCE	162



## PREFACE

The Red Hat Quay application programming interface (API) is an OAuth 2 RESTful API that consists of a set of endpoints for adding, displaying, changing and deleting features for Red Hat Quay.

Red Hat Quay abides by the [Semantic Versioning \(SemVer\) specifications](#). The following conditions are met with each major, minor, and patch release:

- Major versions of Red Hat Quay might include incompatible API changes. For example, the API of Red Hat Quay 2.0 differs from Red Hat Quay 3.0.
- Minor versions of Red Hat Quay, for example, 3.y, adds functionality in a backwards compatible manner.
- Patch versions of Red Hat Quay, for example, 3.y.z, introduces backwards compatible bug fixes.

Currently, Red Hat Quay uses the **api/v1** endpoint for 3.y.z releases.

This guide describes the **api/v1** endpoints and the browser-based examples for accessing those endpoints.

# CHAPTER 1. USING THE RED HAT QUAY API

Red Hat Quay provides a full [OAuth 2](#), RESTful API that:

- Is available from endpoints of each Red Hat Quay instance from the URL <https://<yourquayhost>/api/v1>
- Lets you connect to endpoints, via a browser, to get, delete, post, and put Red Hat Quay settings by enabling the Swagger UI
- Can be accessed by applications that make API calls and use OAuth tokens
- Sends and receives data as JSON

The following text describes how to access the Red Hat Quay API and use it to view and modify setting in your Red Hat Quay cluster. The next section lists and describes API endpoints.

## 1.1. ACCESSING THE QUAY API FROM QUAY.IO

If you don't have your own Red Hat Quay cluster running yet, you can explore the Red Hat Quay API available from Quay.io from your web browser:

```
https://docs.quay.io/api/swagger/
```

The API Explorer that appears shows Quay.io API endpoints. You will not see superuser API endpoints or endpoints for Red Hat Quay features that are not enabled on Quay.io (such as Repository Mirroring).

From API Explorer, you can get, and sometimes change, information on:

- Billing, subscriptions, and plans
- Repository builds and build triggers
- Error messages and global messages
- Repository images, manifests, permissions, notifications, vulnerabilities, and image signing
- Usage logs
- Organizations, members and OAuth applications
- User and robot accounts
- and more...

Select to open an endpoint to view the Model Schema for each part of the endpoint. Open an endpoint, enter any required parameters (such as a repository name or image), then select the **Try it out!** button to query or change settings associated with a Quay.io endpoint.

## 1.2. CREATING AN OAUTH ACCESS TOKEN

OAuth access tokens are credentials that allow you to access protected resources in a secure manner. With Red Hat Quay, you must create an OAuth access token before you can access the API endpoints of your organization.

Use the following procedure to create an OAuth access token.

### Prerequisites

- You have logged in to Red Hat Quay as an administrator.

### Procedure

1. On the main page, select an Organization.
2. In the navigation pane, select **Applications**.
3. Click **Create New Application** and provide a new application name, then press **Enter**.
4. On the **OAuth Applications** page, select the name of your application.
5. Optional. Enter the following information:
  - a. **Application Name**
  - b. **Homepage URL**
  - c. **Description**
  - d. **Avatar E-mail**
  - e. **Redirect/Callback URL prefix**
6. In the navigation pane, select **Generate Token**.
7. Check the boxes for the following options:
  - a. **Administer Organization**
  - b. **Administer Repositories**
  - c. **Create Repositories**
  - d. **View all visible repositories**
  - e. **Read/Write to any accessible repositories**
  - f. **Super User Access**
  - g. **Administer User**
  - h. **Read User Information**
8. Click **Generate Access Token**. You are redirected to a new page.
9. Review the permissions that you are allowing, then click **Authorize Application**. Confirm your decision by clicking **Authorize Application**.
10. You are redirected to the **Access Token** page. Copy and save the access token.





## IMPORTANT

This is the only opportunity to copy and save the access token. It cannot be reobtained after leaving this page.

### 1.3. ACCESSING YOUR QUAY API FROM A WEB BROWSER

By enabling Swagger, you can access the API for your own Red Hat Quay instance through a web browser. This URL exposes the Red Hat Quay API explorer via the Swagger UI and this URL:

```
https://<yourquayhost>/api/v1/discovery.
```

That way of accessing the API does not include superuser endpoints that are available on Red Hat Quay installations. Here is an example of accessing a Red Hat Quay API interface running on the local system by running the `swagger-ui` container image:

```
# export SERVER_HOSTNAME=<yourhostname>
# sudo podman run -p 8888:8080 -e API_URL=https://$SERVER_HOSTNAME:8443/api/v1/discovery
docker.io/swaggerapi/swagger-ui
```

With the `swagger-ui` container running, open your web browser to localhost port 8888 to view API endpoints via the `swagger-ui` container.

To avoid errors in the log such as "API calls must be invoked with an X-Requested-With header if called from a browser," add the following line to the `config.yaml` on all nodes in the cluster and restart Red Hat Quay:

```
BROWSER_API_CALLS_XHR_ONLY: false
```

### 1.4. ACCESSING THE RED HAT QUAY API FROM THE COMMAND LINE

You can use the `curl` command to GET, PUT, POST, or DELETE settings via the API for your Red Hat Quay cluster. Replace `<token>` with the OAuth access token you created earlier to get or change settings in the following examples.

#### 1.4.1. Get superuser information

```
$ curl -X GET -H "Authorization: Bearer <token_here>" \
  "https://<yourquayhost>/api/v1/superuser/users/"
```

For example:

```
$ curl -X GET -H "Authorization: Bearer mFCdgS7SAIoMcnTsHCGx23vcNsTgziAa4CmmHIsg"
http://quay-server:8080/api/v1/superuser/users/ | jq
{
  "users": [
    {
      "kind": "user",
      "name": "quayadmin",
      "username": "quayadmin",
      "email": "quayadmin@example.com",
      "verified": true,
```

```

"avatar": {
  "name": "quayadmin",
  "hash": "357a20e8c56e69d6f9734d23ef9517e8",
  "color": "#5254a3",
  "kind": "user"
},
"super_user": true,
"enabled": true
}
]
}

```

### 1.4.2. Creating a superuser using the API

- Configure a superuser name, as described in the Deploy Quay book:
  - Use the configuration editor UI or
  - Edit the **config.yaml** file directly, with the option of using the configuration API to validate (and download) the updated configuration bundle
- Create the user account for the superuser name:
  - Obtain an authorization token as detailed above, and use **curl** to create the user:

```

$ curl -H "Content-Type: application/json" -H "Authorization: Bearer
Fava2kV9C92p1eXnMawBZx9vTqVnksvwNm0ckFKZ" -X POST --data '{
  "username": "quaysuper",
  "email": "quaysuper@example.com"
}' http://quay-server:8080/api/v1/superuser/users/ | jq

```

- The returned content includes a generated password for the new user account:

```

{
  "username": "quaysuper",
  "email": "quaysuper@example.com",
  "password": "EH67NB3Y6PTBED8H0HC6UVHGGGA3ODSE",
  "encrypted_password":
"fn37AZAUQH0PTsU+vIO9IS0QxPW9A/boXL4ovZjIFtUPrBz9i4j9UDOqMjuxQ/0HTfy38go
KEpG8zYXVeQh3IOFzuOjSvKic2Vq7xdtQsU="
}

```

Now, when you request the list of users, it will show **quaysuper** as a superuser:

```

$ curl -X GET -H "Authorization: Bearer mFCdgS7SAIoMcnTsHCGx23vcNsTgziAa4CmmHlsg"
http://quay-server:8080/api/v1/superuser/users/ | jq

```

```

{
  "users": [
    {
      "kind": "user",
      "name": "quayadmin",
      "username": "quayadmin",
      "email": "quayadmin@example.com",
      "verified": true,

```

```

    "avatar": {
      "name": "quayadmin",
      "hash": "357a20e8c56e69d6f9734d23ef9517e8",
      "color": "#5254a3",
      "kind": "user"
    },
    "super_user": true,
    "enabled": true
  },
  {
    "kind": "user",
    "name": "quaysuper",
    "username": "quaysuper",
    "email": "quaysuper@example.com",
    "verified": true,
    "avatar": {
      "name": "quaysuper",
      "hash": "c0e0f155afcef68e58a42243b153df08",
      "color": "#969696",
      "kind": "user"
    },
    "super_user": true,
    "enabled": true
  }
]
}

```

### 1.4.3. List usage logs

An internal API, `/api/v1/superuser/logs`, is available to list the usage logs for the current system. The results are paginated, so in the following example, more than 20 repos were created to show how to use multiple invocations to access the entire result set.

#### 1.4.3.1. Example for pagination

##### First invocation

```

$ curl -X GET -k -H "Authorization: Bearer qz9NZ2Np1f55CSZ3RVOvxjeUdkzYuCp0pKggABCD"
https://example-registry-quay-quay-enterprise.apps.example.com/api/v1/superuser/logs | jq

```

##### Initial output

```

{
  "start_time": "Sun, 12 Dec 2021 11:41:55 -0000",
  "end_time": "Tue, 14 Dec 2021 11:41:55 -0000",
  "logs": [
    {
      "kind": "create_repo",
      "metadata": {
        "repo": "t21",
        "namespace": "namespace1"
      },
      "ip": "10.131.0.13",
      "datetime": "Mon, 13 Dec 2021 11:41:16 -0000",

```

```
"performer": {
  "kind": "user",
  "name": "user1",
  "is_robot": false,
  "avatar": {
    "name": "user1",
    "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
    "color": "#ad494a",
    "kind": "user"
  }
},
"namespace": {
  "kind": "org",
  "name": "namespace1",
  "avatar": {
    "name": "namespace1",
    "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
    "color": "#e377c2",
    "kind": "org"
  }
}
},
{
  "kind": "create_repo",
  "metadata": {
    "repo": "t20",
    "namespace": "namespace1"
  },
  "ip": "10.131.0.13",
  "datetime": "Mon, 13 Dec 2021 11:41:05 -0000",
  "performer": {
    "kind": "user",
    "name": "user1",
    "is_robot": false,
    "avatar": {
      "name": "user1",
      "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
      "color": "#ad494a",
      "kind": "user"
    }
  },
  "namespace": {
    "kind": "org",
    "name": "namespace1",
    "avatar": {
      "name": "namespace1",
      "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
      "color": "#e377c2",
      "kind": "org"
    }
  }
},
...
{
  "kind": "create_repo",
```

```

"metadata": {
  "repo": "t2",
  "namespace": "namespace1"
},
"ip": "10.131.0.13",
"datetime": "Mon, 13 Dec 2021 11:25:17 -0000",
"performer": {
  "kind": "user",
  "name": "user1",
  "is_robot": false,
  "avatar": {
    "name": "user1",
    "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
    "color": "#ad494a",
    "kind": "user"
  }
},
"namespace": {
  "kind": "org",
  "name": "namespace1",
  "avatar": {
    "name": "namespace1",
    "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
    "color": "#e377c2",
    "kind": "org"
  }
}
}
],
"next_page":
"gAAAAABhtzGDsH38x7pjWhD8MJq1_2FAgqUw2X9S2LoCLNPH65QJqB4XAU2qAxYb6QqtlcWj9eI6
DUiMN_q3e3I0agCvB2VPQ8rY75WeaiUzM3rQIMc4i6EIR78t8oUxVfNp1RMPiRQYYZyXP9h6E8LZZhq
TMs0S-SedaQJ3kVFtkxZqJwHVjgt23Ts2DonVoYwtKgl3bCC5"
}

```

## Second invocation using next\_page

```

$ curl -X GET -k -H "Authorization: Bearer qz9NZ2Np1f55CSZ3RVOvxjeUdkzYuCp0pKggABCD"
https://example-registry-quay-quay-enterprise.apps.example.com/api/v1/superuser/logs?
next_page=gAAAAABhtzGDsH38x7pjWhD8MJq1_2FAgqUw2X9S2LoCLNPH65QJqB4XAU2qAxYb6Q
qtlcWj9eI6DUiMN_q3e3I0agCvB2VPQ8rY75WeaiUzM3rQIMc4i6EIR78t8oUxVfNp1RMPiRQYYZyXP9h
6E8LZZhqTMs0S-SedaQJ3kVFtkxZqJwHVjgt23Ts2DonVoYwtKgl3bCC5 | jq

```

## Output from second invocation

```

{
  "start_time": "Sun, 12 Dec 2021 11:42:46 -0000",
  "end_time": "Tue, 14 Dec 2021 11:42:46 -0000",
  "logs": [
    {
      "kind": "create_repo",
      "metadata": {
        "repo": "t1",
        "namespace": "namespace1"
      },
    },
  ],
}

```

```

    "ip": "10.131.0.13",
    "datetime": "Mon, 13 Dec 2021 11:25:07 -0000",
    "performer": {
      "kind": "user",
      "name": "user1",
      "is_robot": false,
      "avatar": {
        "name": "user1",
        "hash": "5d40b245471708144de9760f2f18113d75aa2488ec82e12435b9de34a6565f73",
        "color": "#ad494a",
        "kind": "user"
      }
    },
    "namespace": {
      "kind": "org",
      "name": "namespace1",
      "avatar": {
        "name": "namespace1",
        "hash": "6cf18b5c19217bfc6df0e7d788746ff7e8201a68cba333fca0437e42379b984f",
        "color": "#e377c2",
        "kind": "org"
      }
    }
  },
  ...
]
}

```

#### 1.4.4. Directory synchronization

To enable directory synchronization for the team **newteam** in organization **testadminorg**, where the corresponding group name in LDAP is **ldapgroup**:

```

$ curl -X POST -H "Authorization: Bearer 9rJYBR3v3pXcj5XqlA2XX6Thkww4gld4TCYLLWDF" \
  -H "Content-type: application/json" \
  -d '{"group_dn": "cn=ldapgroup,ou=Users"}' \
  http://quay1-server:8080/api/v1/organization/testadminorg/team/newteam/syncing

```

To disable synchronization for the same team:

```

$ curl -X DELETE -H "Authorization: Bearer 9rJYBR3v3pXcj5XqlA2XX6Thkww4gld4TCYLLWDF" \
  http://quay1-server:8080/api/v1/organization/testadminorg/team/newteam/syncing

```

#### 1.4.5. Create a repository build via API

In order to build a repository from the specified input and tag the build with custom tags, users can use requestRepoBuild endpoint. It takes the following data:

```

{
  "docker_tags": [
    "string"
  ],
  "pull_robot": "string",

```

```
"subdirectory": "string",
"archive_url": "string"
}
```

The **archive\_url** parameter should point to a **tar** or **zip** archive that includes the Dockerfile and other required files for the build. The **file\_id** parameter was apart of our older build system. It cannot be used anymore. If Dockerfile is in a sub-directory it needs to be specified as well.

The archive should be publicly accessible. OAuth app should have "Administer Organization" scope because only organization admins have access to the robots' account tokens. Otherwise, someone could get robot permissions by simply granting a build access to a robot (without having access themselves), and use it to grab the image contents. In case of errors, check the json block returned and ensure the archive location, pull robot, and other parameters are being passed correctly. Click "Download logs" on the top-right of the individual build's page to check the logs for more verbose messaging.

#### 1.4.6. Create an org robot

```
$ curl -X PUT https://quay.io/api/v1/organization/{orgname}/robots/{robot shortname} \
-H 'Authorization: Bearer <token>'
```

#### 1.4.7. Trigger a build

```
$ curl -X POST https://quay.io/api/v1/repository/YOURORNAME/YOURREPONAME/build/ \
-H 'Authorization: Bearer <token>'
```

Python with requests

```
import requests
r = requests.post('https://quay.io/api/v1/repository/example/example/image', headers={'content-type':
'application/json', 'Authorization': 'Bearer <redacted>'}, data={<request-body-contents>})
print(r.text)
```

#### 1.4.8. Create a private repository

```
$ curl -X POST https://quay.io/api/v1/repository \
-H 'Authorization: Bearer {token}' \
-H 'Content-Type: application/json' \
-d '{"namespace": "yournamespace", "repository": "yourreponame",
"description": "descriptionofyourrepo", "visibility": "private"}' | jq
```

#### 1.4.9. Create a mirrored repository

##### Minimal configuration

```
curl -X POST
-H "Authorization: Bearer ${bearer_token}"
-H "Content-Type: application/json"
--data '{"external_reference": "quay.io/minio/mc", "external_registry_username": "", "sync_interval":
600, "sync_start_date": "2021-08-06T11:11:39Z", "root_rule": {"rule_kind": "tag_glob_csv",
"rule_value": [ "latest" ]}, "robot_username": "orga+robot"}'
https://${quay_registry}/api/v1/repository/${orga}/${repo}/mirror | jq
```

## Extended configuration

```
$ curl -X POST
-H "Authorization: Bearer ${bearer_token}"
-H "Content-Type: application/json"
--data '{"is_enabled": true, "external_reference": "quay.io/minio/mc", "external_registry_username":
"username", "external_registry_password": "password", "external_registry_config":
{"unsigned_images": true, "verify_tls": false, "proxy": {"http_proxy": "http://proxy.tld", "https_proxy":
"https://proxy.tld", "no_proxy": "domain"}}, "sync_interval": 600, "sync_start_date": "2021-08-
06T11:11:39Z", "root_rule": {"rule_kind": "tag_glob_csv", "rule_value": [ "*" ]}, "robot_username":
"orga+robot"}' https://${quay_registry}/api/v1/repository/${orga}/${repo}/mirror | jq
```



## CHAPTER 2. RED HAT QUAY APPLICATION PROGRAMMING INTERFACE (API)

This API allows you to perform many of the operations required to work with Red Hat Quay repositories, users, and organizations.

### 2.1. AUTHORIZATION

oauth2\_implicit

#### Scopes

The following scopes are used to control access to the API endpoints:

Scope	Description
<b>repo:read</b>	This application will be able to view and pull all repositories visible to the granting user or robot account
<b>repo:write</b>	This application will be able to view, push and pull to all repositories to which the granting user or robot account has write access
<b>repo:admin</b>	This application will have administrator access to all repositories to which the granting user or robot account has access
<b>repo:create</b>	This application will be able to create repositories in to any namespaces that the granting user or robot account is allowed to create repositories
<b>user:read</b>	This application will be able to read user information such as username and email address.
<b>org:admin</b>	This application will be able to administer your organizations including creating robots, creating teams, adjusting team membership, and changing billing settings. You should have absolute trust in the requesting application before granting this permission.
<b>super:user</b>	This application will be able to administer your installation including managing users, managing organizations and other features found in the superuser panel. You should have absolute trust in the requesting application before granting this permission.
<b>user:admin</b>	This application will be able to administer your account including creating robots and granting them permissions to your repositories. You should have absolute trust in the requesting application before granting this permission.

### 2.2. APPSPECIFICTOKENS

Manages app specific tokens for the current user.

#### 2.2.1. createAppToken

Create a new app specific token for user.

**POST /api/v1/user/apptoken**

**Authorizations:** oauth2\_implicit (**user:admin**)

### Request body schema (application/json)

Description of a new token.

Name	Description	Schema
<b>title</b> <i>required</i>	Friendly name to help identify the token	string

### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.2.2. listAppTokens

Lists the app specific tokens for the user.

### GET /api/v1/user/apptoken

**Authorizations:** oauth2\_implicit (**user:admin**)

#### Query parameters

Type	Name	Description	Schema
query	<b>expiring</b> <i>optional</i>	If true, only returns those tokens expiring soon	boolean

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>

HTTP Code	Description	Schema
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.2.3. getAppToken

Returns a specific app token for the user.

**GET** /api/v1/user/apptoken/{token\_uuid}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>token_uuid</b> <i>required</i>	The uuid of the app specific token	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.2.4. revokeAppToken

Revokes a specific app token for the user.

**DELETE** /api/v1/user/apptoken/{token\_uuid}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>token_uuid</b> <i>required</i>	The uuid of the app specific token	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.3. BUILD**

Create, list, cancel and get status/logs of repository builds.

**2.3.1. getRepoBuildStatus**

Return the status for the builds specified by the build uuids.

**GET** `/api/v1/repository/{repository}/build/{build_uuid}/status`

**Authorizations:** `oauth2_implicit (repo:read)`

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.3.2. getRepoBuildLogs

Return the build logs for the build specified by the build uuid.

**GET** /api/v1/repository/{repository}/build/{build\_uuid}/logs

**Authorizations:** oauth2\_implicit (repo:read)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.3.3. getRepoBuild

Returns information about a build.

**GET** /api/v1/repository/{repository}/build/{build\_uuid}

**Authorizations:** oauth2\_implicit (repo:read)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.3.4. cancelRepoBuild

Cancels a repository build.

**DELETE** /api/v1/repository/{repository}/build/{build\_uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.3.5. requestRepoBuild

Request that a repository be built and pushed from the specified input.

**POST /api/v1/repository/{repository}/build/****Authorizations:** oauth2\_implicit (**repo:write**)**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Request body schema (application/json)**

Description of a new repository build.

Name	Description	Schema
<b>file_id</b> <i>optional</i>	The file id that was generated when the build spec was uploaded	string
<b>archive_url</b> <i>optional</i>	The URL of the .tar.gz to build. Must start with "http" or "https".	string
<b>subdirectory</b> <i>optional</i>	Subdirectory in which the Dockerfile can be found. You can only specify this or dockerfile_path	string
<b>dockerfile_path</b> <i>optional</i>	Path to a dockerfile. You can only specify this or subdirectory.	string
<b>context</b> <i>optional</i>	Pass in the context for the dockerfile. This is optional.	string
<b>pull_robot</b> <i>optional</i>	Username of a Quay robot account to use as pull credentials	string
<b>tags</b> <i>optional</i>	The tags to which the built images will be pushed. If none specified, "latest" is used.	array of string <b>non-empty unique</b>

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.3.6. getRepoBuilds

Get the list of repository builds.

**GET** /api/v1/repository/{repository}/build/

**Authorizations:** oauth2\_implicit (repo:read)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Query parameters

Type	Name	Description	Schema
query	<b>since</b> <i>optional</i>	Returns all builds since the given unix timecode	integer
query	<b>limit</b> <i>optional</i>	The maximum number of builds to return	integer

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.4. DISCOVERY

API discovery information.



## 2.4.1. discovery

List all of the API endpoints available in the swagger API format.

**GET /api/v1/discovery**

**Authorizations:**

### Query parameters

Type	Name	Description	Schema
query	<b>internal</b> <i>optional</i>	Whether to include internal APIs.	boolean

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.5. ERROR

Error details API.

### 2.5.1. getErrorDescription

Get a detailed description of the error.

**GET /api/v1/error/{error\_type}**

**Authorizations:**

### Path parameters

Type	Name	Description	Schema
path	<b>error_type</b> <i>required</i>	The error code identifying the type of error.	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	<a href="#">ApiErrorDescription</a>
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.6. GLOBALMESSAGES

Messages API.

### 2.6.1. createGlobalMessage

Create a message.

**POST** `/api/v1/messages`

**Authorizations:** `oauth2_implicit (super:user)`

**Request body schema (application/json)**

Create a new message

Name	Description	Schema
<code>message</code> <i>required</i>	A single message	object

### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.6.2. getGlobalMessages

Return a super users messages.

**GET /api/v1/messages**

**Authorizations:**

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.6.3. deleteGlobalMessage

Delete a message.

**DELETE /api/v1/message/{uuid}**

**Authorizations:** oauth2\_implicit (**super:user**)

### Path parameters

Type	Name	Description	Schema
path	<b>uuid</b> <i>required</i>		string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.7. LOGS

Access usage logs for organizations or repositories.

### 2.7.1. getAggregateUserLogs

Returns the aggregated logs for the current user.

**GET /api/v1/user/aggregatelogs**

**Authorizations:** oauth2\_implicit (**user:admin**)

#### Query parameters

Type	Name	Description	Schema
query	<b>performer</b> <i>optional</i>	Username for which to filter logs.	string
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.2. exportUserLogs

Returns the aggregated logs for the current user.

**POST /api/v1/user/exportlogs**

**Authorizations:** oauth2\_implicit (**user:admin**)

#### Query parameters

Type	Name	Description	Schema
------	------	-------------	--------

Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

### Request body schema (application/json)

Configuration for an export logs operation

Name	Description	Schema
<b>callback_url</b> <i>optional</i>	The callback URL to invoke with a link to the exported logs	string
<b>callback_email</b> <i>optional</i>	The e-mail address at which to e-mail a link to the exported logs	string

### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.3. listUserLogs

List the logs for the current user.

**GET /api/v1/user/logs**

**Authorizations:** oauth2\_implicit (**user:admin**)

#### Query parameters

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string

Type	Name	Description	Schema
query	<b>performer</b> <i>optional</i>	Username for which to filter logs.	string
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.4. getAggregateOrgLogs

Gets the aggregated logs for the specified organization.

**GET** /api/v1/organization/{orgname}/aggregatelogs

**Authorizations:** oauth2\_implicit (**org:admin**)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Query parameters

Type	Name	Description	Schema
query	<b>performer</b> <i>optional</i>	Username for which to filter logs.	string

Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.5. exportOrgLogs

Exports the logs for the specified organization.

**POST /api/v1/organization/{orgname}/exportlogs**

**Authorizations:** oauth2\_implicit (**org:admin**)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Query parameters

Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

**Request body schema (application/json)**

Configuration for an export logs operation

Name	Description	Schema
<b>callback_url</b> <i>optional</i>	The callback URL to invoke with a link to the exported logs	string
<b>callback_email</b> <i>optional</i>	The e-mail address at which to e-mail a link to the exported logs	string

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.7.6. listOrgLogs**

List the logs for the specified organization.

**GET /api/v1/organization/{orgname}/logs****Authorizations:** oauth2\_implicit (**org:admin**)**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Query parameters**

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string
query	<b>performer</b> <i>optional</i>	Username for which to filter logs.	string



Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.7. getAggregateRepoLogs

Returns the aggregated logs for the specified repository.

**GET** /api/v1/repository/{repository}/aggregatelogs

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Query parameters

Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.7.8. exportRepoLogs

Queues an export of the logs for the specified repository.

**POST /api/v1/repository/{repository}/exportlogs**

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Query parameters

Type	Name	Description	Schema
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

#### Request body schema (application/json)

Configuration for an export logs operation

Name	Description	Schema
<b>callback_url</b> <i>optional</i>	The callback URL to invoke with a link to the exported logs	string
<b>callback_email</b> <i>optional</i>	The e-mail address at which to e-mail a link to the exported logs	string

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.7.9. listRepoLogs**

List the logs for the specified repository.

**GET /api/v1/repository/{repository}/logs**

**Authorizations:** oauth2\_implicit (repo:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Query parameters**

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string
query	<b>endtime</b> <i>optional</i>	Latest time for logs. Format: "%m/%d/%Y" in UTC.	string
query	<b>starttime</b> <i>optional</i>	Earliest time for logs. Format: "%m/%d/%Y" in UTC.	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.8. MANIFEST

Manage the manifests of a repository.

### 2.8.1. getManifestLabel

Retrieves the label with the specific ID under the manifest.

**GET** `/api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}`

**Authorizations:** `oauth2_implicit (repo:read)`

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string
path	<b>labelid</b> <i>required</i>	The ID of the label	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.8.2. deleteManifestLabel

Deletes an existing label from a manifest.

**DELETE** /api/v1/repository/{repository}/manifest/{manifestref}/labels/{labelid}

**Authorizations:** oauth2\_implicit (repo:write)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string
path	<b>labelid</b> <i>required</i>	The ID of the label	string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.8.3. addManifestLabel

Adds a new label into the tag manifest.

**POST** /api/v1/repository/{repository}/manifest/{manifestref}/labels

**Authorizations:** oauth2\_implicit (repo:write)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string

**Request body schema (application/json)**

Adds a label to a manifest

Name	Description	Schema
<b>key</b> <i>required</i>	The key for the label	string
<b>value</b> <i>required</i>	The value for the label	string
<b>media_type</b> <i>required</i>	The media type for this label	

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.8.4. listManifestLabels**

GET /api/v1/repository/{repository}/manifest/{manifestref}/labels

Authorizations: oauth2\_implicit (repo:read)

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string

**Query parameters**

Type	Name	Description	Schema
------	------	-------------	--------

Type	Name	Description	Schema
query	<b>filter</b> <i>optional</i>	If specified, only labels matching the given prefix will be returned	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.8.5. getRepoManifest

GET /api/v1/repository/{repository}/manifest/{manifestref}

Authorizations: oauth2\_implicit (repo:read)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

## 2.9. MIRROR

### 2.9.1. syncCancel

Update the sync\_status for a given Repository's mirroring configuration.

**POST** /api/v1/repository/{repository}/mirror/sync-cancel

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.9.2. syncNow

Update the sync\_status for a given Repository's mirroring configuration.

**POST** /api/v1/repository/{repository}/mirror/sync-now

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string



**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.9.3. getRepoMirrorConfig**

Return the Mirror configuration for a given Repository.

**GET** `/api/v1/repository/{repository}/mirror`

**Authorizations:** `oauth2_implicit (repo:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	<a href="#">ViewMirrorConfig</a>
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.9.4. changeRepoMirrorConfig**

Allow users to modifying the repository's mirroring configuration.

**PUT** `/api/v1/repository/{repository}/mirror`

**Authorizations:** `oauth2_implicit (repo:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Request body schema (application/json)**

Update the repository mirroring configuration.

Name	Description	Schema
<b>is_enabled</b> <i>optional</i>	Used to enable or disable synchronizations.	boolean
<b>external_reference</b> <i>optional</i>	Location of the external repository.	string
<b>external_registry_username</b> <i>optional</i>	Username used to authenticate with external registry.	
<b>external_registry_password</b> <i>optional</i>	Password used to authenticate with external registry.	
<b>sync_start_date</b> <i>optional</i>	Determines the next time this repository is ready for synchronization.	string
<b>sync_interval</b> <i>optional</i>	Number of seconds after next_start_date to begin synchronizing.	integer
<b>robot_username</b> <i>optional</i>	Username of robot which will be used for image pushes.	string
<b>root_rule</b> <i>optional</i>	A list of glob-patterns used to determine which tags should be synchronized.	object
<b>external_registry_config</b> <i>optional</i>		object

**Responses**

HTTP Code	Description	Schema
201	Successful invocation	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.9.5. createRepoMirrorConfig

Create a RepoMirrorConfig for a given Repository.

**POST** /api/v1/repository/{repository}/mirror

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Create the repository mirroring configuration.

Name	Description	Schema
<b>is_enabled</b> <i>optional</i>	Used to enable or disable synchronizations.	boolean
<b>external_reference</b> <i>required</i>	Location of the external repository.	string
<b>external_registry_username</b> <i>optional</i>	Username used to authenticate with external registry.	
<b>external_registry_password</b> <i>optional</i>	Password used to authenticate with external registry.	
<b>sync_start_date</b> <i>required</i>	Determines the next time this repository is ready for synchronization.	string

Name	Description	Schema
<b>sync_interval</b> <i>required</i>	Number of seconds after next_start_date to begin synchronizing.	integer
<b>robot_username</b> <i>required</i>	Username of robot which will be used for image pushes.	string
<b>root_rule</b> <i>required</i>	A list of glob-patterns used to determine which tags should be synchronized.	object
<b>external_registry_config</b> <i>optional</i>		object

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.10. NAMESPACEQUOTA

### 2.10.1. listUserQuota

GET /api/v1/user/quota

Authorizations: oauth2\_implicit (user:admin)

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>

HTTP Code	Description	Schema
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.2. getOrganizationQuotaLimit

GET /api/v1/organization/{orgname}/quota/{quota\_id}/limit/{limit\_id}

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>limit_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.3. changeOrganizationQuotaLimit

PUT /api/v1/organization/{orgname}/quota/{quota\_id}/limit/{limit\_id}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>limit_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

### Request body schema (application/json)

Description of changing organization quota limit

Name	Description	Schema
<b>type</b> <i>optional</i>	Type of quota limit: "Warning" or "Reject"	string
<b>threshold_percent</b> <i>optional</i>	Quota threshold, in percent of quota	integer

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

#### 2.10.4. deleteOrganizationQuotaLimit

DELETE /api/v1/organization/{orgname}/quota/{quota\_id}/limit/{limit\_id}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>limit_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.5. createOrganizationQuotaLimit

POST /api/v1/organization/{orgname}/quota/{quota\_id}/limit

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

#### Request body schema (application/json)

Description of a new organization quota limit

Name	Description	Schema
<b>type</b> <i>required</i>	Type of quota limit: "Warning" or "Reject"	string
<b>threshold_percent</b> <i>required</i>	Quota threshold, in percent of quota	integer

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.6. listOrganizationQuotaLimit

GET /api/v1/organization/{orgname}/quota/{quota\_id}/limit

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>



HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.7. getUserQuotaLimit

GET /api/v1/user/quota/{quota\_id}/limit/{limit\_id}

Authorizations: oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>limit_id</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.8. listUserQuotaLimit

GET /api/v1/user/quota/{quota\_id}/limit

Authorizations: oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
------	------	-------------	--------

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.9. getOrganizationQuota

GET /api/v1/organization/{orgname}/quota/{quota\_id}

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.10.10. changeOrganizationQuota

PUT /api/v1/organization/{orgname}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

#### Request body schema (application/json)

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>optional</i>	Number of bytes the organization is allowed	integer

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.11. deleteOrganizationQuota

DELETE /api/v1/organization/{orgname}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

**Path parameters**

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.10.12. createOrganizationQuota**

Create a new organization quota.

**POST /api/v1/organization/{orgname}/quota**

**Authorizations:** oauth2\_implicit (super:user)

**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>		string

**Request body schema (application/json)**

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>required</i>	Number of bytes the organization is allowed	integer

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.13. listOrganizationQuota

GET /api/v1/organization/{orgname}/quota

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.10.14. getUserQuota

GET /api/v1/user/quota/{quota\_id}

Authorizations: oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.11. ORGANIZATION

Manage organizations, members and OAuth applications.

### 2.11.1. createOrganization

Create a new organization.

**POST /api/v1/organization/**

**Authorizations:** oauth2\_implicit (**user:admin**)

**Request body schema (application/json)**

Description of a new organization.

Name	Description	Schema
<b>name</b> <i>required</i>	Organization username	string
<b>email</b> <i>optional</i>	Organization contact email	string
<b>recaptcha_response</b> <i>optional</i>	The (may be disabled) recaptcha response code for verification	string

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.2. validateProxyCacheConfig

POST /api/v1/organization/{orgname}/validateproxycache

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>		string

#### Request body schema (application/json)

Proxy cache configuration for an organization

Name	Description	Schema
<b>upstream_registry</b> <i>required</i>	Name of the upstream registry that is to be cached	string

#### Responses

HTTP Code	Description	Schema
202	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.3. getOrganizationCollaborators

List outside collaborators of the specified organization.

**GET** /api/v1/organization/{orgname}/collaborators

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.4. getOrganizationApplication

Retrieves the application with the specified client\_id under the specified organization.

**GET** /api/v1/organization/{orgname}/applications/{client\_id}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>client_id</b> <i>required</i>	The OAuth client ID	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses



HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.5. updateOrganizationApplication

Updates an application under this organization.

**PUT /api/v1/organization/{orgname}/applications/{client\_id}**

**Authorizations:** oauth2\_implicit (**org:admin**)

#### Path parameters

Type	Name	Description	Schema
path	<b>client_id</b> <i>required</i>	The OAuth client ID	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Request body schema (application/json)

Description of an updated application.

Name	Description	Schema
<b>name</b> <i>required</i>	The name of the application	string
<b>redirect_uri</b> <i>required</i>	The URI for the application's OAuth redirect	string
<b>application_uri</b> <i>required</i>	The URI for the application's homepage	string
<b>description</b> <i>optional</i>	The human-readable description for the application	string

Name	Description	Schema
<b>avatar_email</b> <i>optional</i>	The e-mail address of the avatar to use for the application	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.6. deleteOrganizationApplication

Deletes the application under this organization.

**DELETE** /api/v1/organization/{orgname}/applications/{client\_id}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>client_id</b> <i>required</i>	The OAuth client ID	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.11.7. createOrganizationApplication

Creates a new application under this organization.

**POST** /api/v1/organization/{orgname}/applications

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Request body schema (application/json)

Description of a new organization application.

Name	Description	Schema
<b>name</b> <i>required</i>	The name of the application	string
<b>redirect_uri</b> <i>optional</i>	The URI for the application's OAuth redirect	string
<b>application_uri</b> <i>optional</i>	The URI for the application's homepage	string
<b>description</b> <i>optional</i>	The human-readable description for the application	string
<b>avatar_email</b> <i>optional</i>	The e-mail address of the avatar to use for the application	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.8. getOrganizationApplications

List the applications for the specified organization.

**GET** /api/v1/organization/{orgname}/applications

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.9. getProxyCacheConfig

Retrieves the proxy cache configuration of the organization.

**GET** /api/v1/organization/{orgname}/proxycache

**Authorizations:**

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.10. deleteProxyCacheConfig

Delete proxy cache configuration for the organization.

**DELETE** /api/v1/organization/{orgname}/proxycache

Authorizations:

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.11.11. createProxyCacheConfig

Creates proxy cache configuration for the organization.

**POST /api/v1/organization/{orgname}/proxycache**

**Authorizations:**

**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Request body schema (application/json)**

Proxy cache configuration for an organization

Name	Description	Schema
<b>upstream_registry</b> <i>required</i>	Name of the upstream registry that is to be cached	string

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.12. getOrganizationMember

Retrieves the details of a member of the organization.

**GET /api/v1/organization/{orgname}/members/{membername}**

**Authorizations:** `oauth2_implicit (org:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>membername</b> <i>required</i>	The username of the organization member	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.11.13. removeOrganizationMember**

Removes a member from an organization, revoking all its repository privileges and removing it from all teams in the organization.

**DELETE** /api/v1/organization/{orgname}/members/{membername}

**Authorizations:** oauth2\_implicit (org:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>membername</b> <i>required</i>	The username of the organization member	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.14. getOrganizationMembers

List the human members of the specified organization.

**GET /api/v1/organization/{orgname}/members**

**Authorizations:** oauth2\_implicit (**org:admin**)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.15. getOrganization

Get the details for the specified organization.

**GET /api/v1/organization/{orgname}**

**Authorizations:**

#### Path parameters



Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.16. changeOrganizationDetails

Change the details for the specified organization.

**PUT /api/v1/organization/{orgname}**

**Authorizations:** oauth2\_implicit (**org:admin**)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Request body schema (application/json)

Description of updates for an existing organization

Name	Description	Schema
<b>email</b> <i>optional</i>	Organization contact email	string
<b>invoice_email</b> <i>optional</i>	Whether the organization desires to receive emails for invoices	boolean
<b>invoice_email_address</b> <i>optional</i>	The email address at which to receive invoices	

Name	Description	Schema
<b>tag_expiration_s</b> <i>optional</i>	The number of seconds for tag expiration	integer

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.11.17. deleteAdminedOrganization

Deletes the specified organization.

**DELETE** /api/v1/organization/{orgname}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.11.18. getApplicationInformation

Get information on the specified application.

GET /api/v1/app/{client\_id}

Authorizations:

### Path parameters

Type	Name	Description	Schema
path	<b>client_id</b> <i>required</i>	The OAuth client ID	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.12. PERMISSION

Manage repository permissions.

### 2.12.1. getUserTransitivePermission

Get the fetch the permission for the specified user.

GET /api/v1/repository/{repository}/permissions/user/{username}/transitive

Authorizations: oauth2\_implicit (repo:admin)

### Path parameters

Type	Name	Description	Schema
path	<b>username</b> <i>required</i>	The username of the user to which the permissions apply	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.2. getUserPermissions

Get the permission for the specified user.

**GET** /api/v1/repository/{repository}/permissions/user/{username}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>username</b> <i>required</i>	The username of the user to which the permission applies	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.3. changeUserPermissions

Update the permissions for an existing repository.

PUT /api/v1/repository/{repository}/permissions/user/{username}

Authorizations: oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>username</b> <i>required</i>	The username of the user to which the permission applies	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Description of a user permission.

Name	Description	Schema
<b>role</b> <i>required</i>	Role to use for the user	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.4. deleteUserPermissions

Delete the permission for the user.

DELETE /api/v1/repository/{repository}/permissions/user/{username}

Authorizations: oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>username</b> <i>required</i>	The username of the user to which the permission applies	string

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.5. getTeamPermissions

Fetch the permission for the specified team.

**GET** /api/v1/repository/{repository}/permissions/team/{teamname}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>teamname</b> <i>required</i>	The name of the team to which the permission applies	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>

HTTP Code	Description	Schema
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.6. changeTeamPermissions

Update the existing team permission.

PUT /api/v1/repository/{repository}/permissions/team/{teamname}

Authorizations: oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>teamname</b> <i>required</i>	The name of the team to which the permission applies	string

#### Request body schema (application/json)

Description of a team permission.

Name	Description	Schema
<b>role</b> <i>required</i>	Role to use for the team	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.7. deleteTeamPermissions

Delete the permission for the specified team.

**DELETE** /api/v1/repository/{repository}/permissions/team/{teamname}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>teamname</b> <i>required</i>	The name of the team to which the permission applies	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.8. listRepoTeamPermissions

List all team permission.

**GET** /api/v1/repository/{repository}/permissions/team/

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	



HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.12.9. listRepoUserPermissions

List all user permissions.

**GET /api/v1/repository/{repository}/permissions/user/**  
**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.13. POLICY

### 2.13.1. createOrganizationAutoPrunePolicy

Creates an auto-prune policy for the organization

**POST /api/v1/organization/{orgname}/autoprunepolicy/**  
**Authorizations:** oauth2\_implicit (org:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Request body schema (application/json)**

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>required</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>required</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.2. listOrganizationAutoPrunePolicies**

Lists the auto-prune policies for the organization

**GET** /api/v1/organization/{orgname}/autoprunepolicy/

**Authorizations:** oauth2\_implicit (org:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.3. getOrganizationAutoPrunePolicy

Fetches the auto-prune policy for the organization

**GET** /api/v1/organization/{orgname}/autoprunepolicy/{policy\_uuid}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.4. deleteOrganizationAutoPrunePolicy

Deletes the auto-prune policy for the organization

**DELETE /api/v1/organization/{orgname}/autoprunepolicy/{policy\_uuid}****Authorizations:** oauth2\_implicit (org:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.5. updateOrganizationAutoPrunePolicy**

Updates the auto-prune policy for the organization

**PUT /api/v1/organization/{orgname}/autoprunepolicy/{policy\_uuid}****Authorizations:** oauth2\_implicit (org:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string

**Request body schema (application/json)**

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>required</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>required</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

## Responses

HTTP Code	Description	Schema
204	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.6. createRepositoryAutoPrunePolicy

Creates an auto-prune policy for the repository

**POST** /api/v1/repository/{repository}/autoprunepolicy/

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>optional</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>optional</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.7. listRepositoryAutoPrunePolicies**

Lists the auto-prune policies for the repository

**GET** `/api/v1/repository/{repository}/autoprunepolicy/`

**Authorizations:** `oauth2_implicit (repo:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.8. getRepositoryAutoPrunePolicy**

Fetches the auto-prune policy for the repository

**GET** `/api/v1/repository/{repository}/autoprunepolicy/{policy_uuid}`

**Authorizations:** `oauth2_implicit (repo:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.9. deleteRepositoryAutoPrunePolicy**

Deletes the auto-prune policy for the repository

**DELETE** /api/v1/repository/{repository}/autoprunepolicy/{policy\_uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.10. updateRepositoryAutoPrunePolicy

Updates the auto-prune policy for the repository

PUT /api/v1/repository/{repository}/autoprunepolicy/{policy\_uuid}

Authorizations: oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>optional</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>optional</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>



HTTP Code	Description	Schema
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.11. createUserAutoPrunePolicy

Creates the auto-prune policy for the currently logged in user

**POST** /api/v1/user/autoprunepolicy/

**Authorizations:** oauth2\_implicit (**user:admin**)

**Request body schema (application/json)**

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>required</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>required</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.12. listUserAutoPrunePolicies

Lists the auto-prune policies for the currently logged in user

**GET** /api/v1/user/autoprunepolicy/

**Authorizations:** oauth2\_implicit (**user:admin**)

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.13. getUserAutoPrunePolicy

Fetches the auto-prune policy for the currently logged in user

**GET** /api/v1/user/autoprunepolicy/{policy\_uuid}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.13.14. deleteUserAutoPrunePolicy

Deletes the auto-prune policy for the currently logged in user

**DELETE** /api/v1/user/autoprunepolicy/{policy\_uuid}

**Authorizations:** oauth2\_implicit (user:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.13.15. updateUserAutoPrunePolicy**

Updates the auto-prune policy for the currently logged in user

**PUT /api/v1/user/autoprunepolicy/{policy\_uuid}**

**Authorizations:** oauth2\_implicit (user:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>policy_uuid</b> <i>required</i>	The unique ID of the policy	string

**Request body schema (application/json)**

The policy configuration that is to be applied to the user namespace

Name	Description	Schema
<b>method</b> <i>required</i>	The method to use for pruning tags (number_of_tags, creation_date)	string
<b>value</b> <i>required</i>	The value to use for the pruning method (number of tags e.g. 10, time delta e.g. 7d (7 days))	

**Responses**

HTTP Code	Description	Schema
204	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.14. PROTOTYPE

Manage default permissions added to repositories.

### 2.14.1. updateOrganizationPrototypePermission

Update the role of an existing permission prototype.

**PUT** /api/v1/organization/{orgname}/prototypes/{prototypeid}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>prototypeid</b> <i>required</i>	The ID of the prototype	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Request body schema (application/json)

Description of a the new prototype role

Name	Description	Schema
<b>role</b> <i>optional</i>	Role that should be applied to the permission	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.14.2. deleteOrganizationPrototypePermission

Delete an existing permission prototype.

**DELETE** /api/v1/organization/{orgname}/prototypes/{prototypeid}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>prototypeid</b> <i>required</i>	The ID of the prototype	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.14.3. createOrganizationPrototypePermission

Create a new permission prototype.

**POST** /api/v1/organization/{orgname}/prototypes

**Authorizations:** oauth2\_implicit (org:admin)

**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Request body schema (application/json)**

Description of a new prototype

Name	Description	Schema
<b>role</b> <i>required</i>	Role that should be applied to the delegate	string
<b>activating_user</b> <i>optional</i>	Repository creating user to whom the rule should apply	object
<b>delegate</b> <i>required</i>	Information about the user or team to which the rule grants access	object

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.14.4. getOrganizationPrototypePermissions**

List the existing prototypes for this organization.

**GET /api/v1/organization/{orgname}/prototypes****Authorizations:** oauth2\_implicit (org:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.15. REPOSITORY**

List, create and manage repositories.

**2.15.1. createRepo**

Create a new repository.

**POST /api/v1/repository**

**Authorizations:** `oauth2_implicit` (`repo:create`)

**Request body schema (application/json)**

Description of a new repository

Name	Description	Schema
<b>repository</b> <i>required</i>	Repository name	string
<b>visibility</b> <i>required</i>	Visibility which the repository will start with	string
<b>namespace</b> <i>optional</i>	Namespace in which the repository should be created. If omitted, the username of the caller is used	string
<b>description</b> <i>required</i>	Markdown encoded description for the repository	string
<b>repo_kind</b> <i>optional</i>	The kind of repository	

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.15.2. listRepos

Fetch the list of repositories visible to the current user under a variety of situations.

**GET /api/v1/repository**

**Authorizations:** `oauth2_implicit (repo:read)`

#### Query parameters

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string
query	<b>repo_kind</b> <i>optional</i>	The kind of repositories to return	string
query	<b>popularity</b> <i>optional</i>	Whether to include the repository's popularity metric.	boolean
query	<b>last_modified</b> <i>optional</i>	Whether to include when the repository was last modified.	boolean
query	<b>public</b> <i>optional</i>	Adds any repositories visible to the user by virtue of being public	boolean
query	<b>starred</b> <i>optional</i>	Filters the repositories returned to those starred by the user	boolean
query	<b>namespace</b> <i>optional</i>	Filters the repositories returned to this namespace	string

#### Responses



HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.15.3. changeRepoVisibility

Change the visibility of a repository.

**POST /api/v1/repository/{repository}/changevisibility**

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Change the visibility for the repository.

Name	Description	Schema
<b>visibility</b> <i>required</i>	Visibility which the repository will start with	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.15.4. changeRepoState

Change the state of a repository.

**PUT** /api/v1/repository/{repository}/changestate

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Change the state of the repository.

Name	Description	Schema
<b>state</b> <i>required</i>	Determines whether pushes are allowed.	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.15.5. getRepo

Fetch the specified repository.

**GET** /api/v1/repository/{repository}

**Authorizations:** oauth2\_implicit (repo:read)

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Query parameters**

Type	Name	Description	Schema
query	<b>includeTags</b> <i>optional</i>	Whether to include repository tags	boolean
query	<b>includeStats</b> <i>optional</i>	Whether to include action statistics	boolean

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.15.6. updateRepo**

Update the description in the specified repository.

**PUT /api/v1/repository/{repository}**

**Authorizations:** `oauth2_implicit (repo:write)`

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Request body schema (application/json)**

Fields which can be updated in a repository.

Name	Description	Schema
<b>description</b> <i>required</i>	Markdown encoded description for the repository	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.15.7. deleteRepository

Delete a repository.

**DELETE** /api/v1/repository/{repository}

**Authorizations:** oauth2\_implicit (repo:admin)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.16. REPOSITORYNOTIFICATION

List, create and manage repository events/notifications.

### 2.16.1. testRepoNotification

Queues a test notification for this repository.

**POST** /api/v1/repository/{repository}/notification/{uuid}/test

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>uuid</b> <i>required</i>	The UUID of the notification	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.16.2. getRepoNotification

Get information for the specified notification.

**GET** /api/v1/repository/{repository}/notification/{uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

Type	Name	Description	Schema
path	<b>uuid</b> <i>required</i>	The UUID of the notification	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.16.3. deleteRepoNotification

Deletes the specified notification.

**DELETE** /api/v1/repository/{repository}/notification/{uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>uuid</b> <i>required</i>	The UUID of the notification	string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.16.4. resetRepositoryNotificationFailures

Resets repository notification to 0 failures.

**POST** /api/v1/repository/{repository}/notification/{uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>uuid</b> <i>required</i>	The UUID of the notification	string

#### Responses

HTTP Code	Description	Schema
204	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.16.5. createRepoNotification

**POST** /api/v1/repository/{repository}/notification/

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Request body schema (application/json)**

Information for creating a notification on a repository

Name	Description	Schema
<b>event</b> <i>required</i>	The event on which the notification will respond	string
<b>method</b> <i>required</i>	The method of notification (such as email or web callback)	string
<b>config</b> <i>required</i>	JSON config information for the specific method of notification	object
<b>eventConfig</b> <i>required</i>	JSON config information for the specific event of notification	object
<b>title</b> <i>optional</i>	The human-readable title of the notification	string

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.16.6. listRepoNotifications**

List the notifications for the specified repository.

**GET /api/v1/repository/{repository}/notification/****Authorizations:** oauth2\_implicit (repo:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string



**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.17. REPOTOKEN**

Manage repository access tokens (DEPRECATED).

**2.17.1. getTokens**

Fetch the specified repository token information.

**GET** `/api/v1/repository/{repository}/tokens/{code}`

**Authorizations:** `oauth2_implicit` (`repo:admin`)

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>code</b> <i>required</i>	The token code	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.17.2. changeToken

Update the permissions for the specified repository token.

**PUT** /api/v1/repository/{repository}/tokens/{code}

**Authorizations:** oauth2\_implicit (repo:admin)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>code</b> <i>required</i>	The token code	string

### Request body schema (application/json)

Description of a token permission

Name	Description	Schema
<b>role</b> <i>optional</i>	Role to use for the token	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.17.3. deleteToken

Delete the repository token.

**DELETE** /api/v1/repository/{repository}/tokens/{code}

**Authorizations:** oauth2\_implicit (repo:admin)

### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>code</b> <i>required</i>	The token code	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.17.4. createToken

Create a new repository token.

**POST** /api/v1/repository/{repository}/tokens/  
**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Description of a new token.

Name	Description	Schema
<b>friendlyName</b> <i>required</i>	Friendly name to help identify the token	string

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.17.5. listRepoTokens

List the tokens for the specified repository.

**GET** `/api/v1/repository/{repository}/tokens/`  
**Authorizations:** `oauth2_implicit (repo:admin)`

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.18. ROBOT

Manage user and organization robot accounts.

### 2.18.1. getUserRobots

List the available robots for the user.

**GET /api/v1/user/robots****Authorizations:** oauth2\_implicit (**user:admin**)**Query parameters**

Type	Name	Description	Schema
query	<b>limit</b> <i>optional</i>	If specified, the number of robots to return.	integer
query	<b>token</b> <i>optional</i>	If false, the robot's token is not returned.	boolean
query	<b>permissions</b> <i>optional</i>	Whether to include repositories and teams in which the robots have permission.	boolean

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.18.2. getOrgRobotPermissions**

Returns the list of repository permissions for the org's robot.

**GET /api/v1/organization/{orgname}/robots/{robot\_shortname}/permissions****Authorizations:** oauth2\_implicit (**user:admin**)**Path parameters**

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.3. regenerateOrgRobotToken

Regenerates the token for an organization robot.

**POST** /api/v1/organization/{orgname}/robots/{robot\_shortname}/regenerate

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.4. getUserRobotPermissions

Returns the list of repository permissions for the user's robot.

**GET /api/v1/user/robots/{robot\_shortname}/permissions****Authorizations:** oauth2\_implicit (user:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.18.5. regenerateUserRobotToken**

Regenerates the token for a user's robot.

**POST /api/v1/user/robots/{robot\_shortname}/regenerate****Authorizations:** oauth2\_implicit (user:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string

**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.6. getOrgRobot

Returns the organization's robot with the specified name.

**GET** /api/v1/organization/{orgname}/robots/{robot\_shortcode}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>robot_shortcode</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.7. createOrgRobot

Create a new robot in the organization.

**PUT** /api/v1/organization/{orgname}/robots/{robot\_shortcode}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters



Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

### Request body schema (application/json)

Optional data for creating a robot

Name	Description	Schema
<b>description</b> <i>optional</i>	Optional text description for the robot	string
<b>unstructured_metadata</b> <i>optional</i>	Optional unstructured metadata for the robot	object

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

#### 2.18.8. deleteOrgRobot

Delete an existing organization robot.

**DELETE** /api/v1/organization/{orgname}/robots/{robot\_shortname}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
------	------	-------------	--------

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.9. getOrgRobots

List the organization's robots.

**GET** /api/v1/organization/{orgname}/robots

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Query parameters

Type	Name	Description	Schema
query	<b>limit</b> <i>optional</i>	If specified, the number of robots to return.	integer
query	<b>token</b> <i>optional</i>	If false, the robot's token is not returned.	boolean

Type	Name	Description	Schema
query	<b>permissions</b> <i>optional</i>	Whether to include repositories and teams in which the robots have permission.	boolean

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.10. getUserRobot

Returns the user's robot with the specified name.

**GET** /api/v1/user/robots/{robot\_shortcode}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>robot_shortcode</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.11. createUserRobot

Create a new user robot with the specified name.

**PUT** /api/v1/user/robots/{robot\_shortname}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string

#### Request body schema (application/json)

Optional data for creating a robot

Name	Description	Schema
<b>description</b> <i>optional</i>	Optional text description for the robot	string
<b>unstructured_metadata</b> <i>optional</i>	Optional unstructured metadata for the robot	object

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.18.12. deleteUserRobot

Delete an existing robot.

**DELETE** /api/v1/user/robots/{robot\_shortname}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>robot_shortname</b> <i>required</i>	The short name for the robot, without any user or organization prefix	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.19. SEARCH

Conduct searches against all registry context.

### 2.19.1. conductRepoSearch

Get a list of apps and repositories that match the specified query.

**GET /api/v1/find/repositories**

**Authorizations:**

#### Query parameters

Type	Name	Description	Schema
query	<b>includeUsage</b> <i>optional</i>	Whether to include usage metadata	boolean
query	<b>page</b> <i>optional</i>	The page.	integer
query	<b>query</b> <i>optional</i>	The search query.	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.19.2. conductSearch

Get a list of entities and resources that match the specified query.

**GET /api/v1/find/all**

**Authorizations:** oauth2\_implicit (repo:read)

#### Query parameters

Type	Name	Description	Schema
query	<b>query</b> <i>optional</i>	The search query.	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.19.3. getMatchingEntities

Get a list of entities that match the specified prefix.

**GET /api/v1/entities/{prefix}**

**Authorizations:**

**Path parameters**

Type	Name	Description	Schema
path	<b>prefix</b> <i>required</i>		string

**Query parameters**

Type	Name	Description	Schema
query	<b>includeOrgs</b> <i>optional</i>	Whether to include orgs names.	boolean
query	<b>includeTeams</b> <i>optional</i>	Whether to include team names.	boolean
query	<b>namespace</b> <i>optional</i>	Namespace to use when querying for org entities.	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.20. SECSCAN**

List and manage repository vulnerabilities and other security information.

**2.20.1. getRepoManifestSecurity**

GET /api/v1/repository/{repository}/manifest/{manifestref}/security

Authorizations: oauth2\_implicit (repo:read)

**Path parameters**

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

Type	Name	Description	Schema
path	<b>manifestref</b> <i>required</i>	The digest of the manifest	string

### Query parameters

Type	Name	Description	Schema
query	<b>vulnerabilities</b> <i>optional</i>	Include vulnerabilities informations	boolean

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.21. SUPERUSER

Superuser API.

### 2.21.1. createInstallUser

Creates a new user.

**POST** /api/v1/superuser/users/

**Authorizations:** oauth2\_implicit (**super:user**)

**Request body schema (application/json)**

Data for creating a user

Name	Description	Schema
<b>username</b> <i>required</i>	The username of the user being created	string
<b>email</b> <i>optional</i>	The email address of the user being created	string



**Responses**

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.21.2. listAllUsers**

Returns a list of all users in the system.

**GET** `/api/v1/superuser/users/`

**Authorizations:** `oauth2_implicit (super:user)`

**Query parameters**

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string
query	<b>limit</b> <i>optional</i>	Limit to the number of results to return per page. Max 100.	integer
query	<b>disabled</b> <i>optional</i>	If false, only enabled users will be returned.	boolean

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.3. listAllLogs

List the usage logs for the current system.

**GET /api/v1/superuser/logs**

**Authorizations:** oauth2\_implicit (**super:user**)

#### Query parameters

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string
query	<b>page</b> <i>optional</i>	The page number for the logs	integer
query	<b>endtime</b> <i>optional</i>	Latest time to which to get logs (%m/%d/%Y %Z)	string
query	<b>starttime</b> <i>optional</i>	Earliest time from which to get logs (%m/%d/%Y %Z)	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.4. createServiceKey

**POST /api/v1/superuser/keys**

**Authorizations:** oauth2\_implicit (**super:user**)

**Request body schema (application/json)**

Description of creation of a service key

Name	Description	Schema
<b>service</b> <i>required</i>	The service authenticating with this key	string

Name	Description	Schema
<b>name</b> <i>optional</i>	The friendly name of a service key	string
<b>metadata</b> <i>optional</i>	The key/value pairs of this key's metadata	object
<b>notes</b> <i>optional</i>	If specified, the extra notes for the key	string
<b>expiration</b> <i>required</i>	The expiration date as a unix timestamp	

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.5. listServiceKeys

GET /api/v1/superuser/keys

Authorizations: oauth2\_implicit (super:user)

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.21.6. changeUserQuotaSuperUser

PUT /api/v1/superuser/organization/{namespace}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string
path	<b>quota_id</b> <i>required</i>		string

### Request body schema (application/json)

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>optional</i>	Number of bytes the organization is allowed	integer

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.21.7. deleteUserQuotaSuperUser

DELETE /api/v1/superuser/organization/{namespace}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string

### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.8. createUserQuotaSuperUser

POST /api/v1/superuser/organization/{namespace}/quota

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

#### Request body schema (application/json)

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>required</i>	Number of bytes the organization is allowed	integer

### Responses

HTTP Code	Description	Schema
201	Successful creation	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.9. listUserQuotaSuperUser

GET /api/v1/superuser/organization/{namespace}/quota

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.10. changeOrganizationQuotaSuperUser

PUT /api/v1/superuser/users/{namespace}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

Type	Name	Description	Schema
path	<b>quota_id</b> <i>required</i>		string

### Request body schema (application/json)

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>optional</i>	Number of bytes the organization is allowed	integer

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.11. deleteOrganizationQuotaSuperUser

DELETE /api/v1/superuser/users/{namespace}/quota/{quota\_id}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string
path	<b>quota_id</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.12. createOrganizationQuotaSuperUser

POST /api/v1/superuser/users/{namespace}/quota  
 Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

#### Request body schema (application/json)

Description of a new organization quota

Name	Description	Schema
<b>limit_bytes</b> <i>optional</i>	Number of bytes the organization is allowed	integer

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>



### 2.21.13. listOrganizationQuotaSuperUser

GET /api/v1/superuser/users/{namespace}/quota

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.14. changeOrganization

Updates information about the specified user.

PUT /api/v1/superuser/organizations/{name}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>name</b> <i>required</i>	The name of the organization being managed	string

#### Request body schema (application/json)

Description of updates for an existing organization

Name	Description	Schema
<b>email</b> <i>optional</i>	Organization contact email	string

Name	Description	Schema
<b>invoice_email</b> <i>optional</i>	Whether the organization desires to receive emails for invoices	boolean
<b>invoice_email_address</b> <i>optional</i>	The email address at which to receive invoices	
<b>tag_expiration_s</b> <i>optional</i>	The number of seconds for tag expiration	integer

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.15. deleteOrganization

Deletes the specified organization.

**DELETE** /api/v1/superuser/organizations/{name}

**Authorizations:** oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>name</b> <i>required</i>	The name of the organization being managed	string

## Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.16. approveServiceKey

POST /api/v1/superuser/approvedkeys/{kid}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>kid</b> <i>required</i>	The unique identifier for a service key	string

#### Request body schema (application/json)

Information for approving service keys

Name	Description	Schema
<b>notes</b> <i>optional</i>	Optional approval notes	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.17. deleteServiceKey

DELETE /api/v1/superuser/keys/{kid}

Authorizations: oauth2\_implicit (super:user)

**Path parameters**

Type	Name	Description	Schema
path	<b>kid</b> <i>required</i>	The unique identifier for a service key	string

**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.21.18. updateServiceKey**

PUT /api/v1/superuser/keys/{kid}

Authorizations: oauth2\_implicit (super:user)

**Path parameters**

Type	Name	Description	Schema
path	<b>kid</b> <i>required</i>	The unique identifier for a service key	string

**Request body schema (application/json)**

Description of updates for a service key

Name	Description	Schema
<b>name</b> <i>optional</i>	The friendly name of a service key	string
<b>metadata</b> <i>optional</i>	The key/value pairs of this key's metadata	object
<b>expiration</b> <i>optional</i>	The expiration date as a unix timestamp	

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.19. getServiceKey

GET /api/v1/superuser/keys/{kid}

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>kid</b> <i>required</i>	The unique identifier for a service key	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.20. getRepoBuildStatusSuperUser

Return the status for the builds specified by the build uuids.

GET /api/v1/superuser/{build\_uuid}/status

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.21. getRepoBuildSuperUser

Returns information about a build.

**GET** /api/v1/superuser/{build\_uuid}/build  
**Authorizations:** oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.22. getRepoBuildLogsSuperUser

Return the build logs for the build specified by the build uuid.

GET /api/v1/superuser/{build\_uuid}/logs

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>build_uuid</b> <i>required</i>	The UUID of the build	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.23. getRegistrySize

GET /api/v1/superuser/registrysize/

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	<b>namespace</b> <i>required</i>		string

Description of a image registry size

Name	Description	Schema
<b>size_bytes*</b> <i>optional</i>	Number of bytes the organization is allowed	integer
<b>last_ran</b>		integer

Name	Description	Schema
queued		boolean
running		boolean

## Responses

HTTP Code	Description	Schema
200	CREATED	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.21.24. postRegistrySize

POST /api/v1/superuser/registrysize/

Authorizations: oauth2\_implicit (super:user)

#### Path parameters

Type	Name	Description	Schema
path	namespace <i>required</i>		string

#### Request body schema (application/json)

Description of a image registry size

Name	Description	Schema
last_ran		integer
queued		boolean
running		boolean

## Responses



HTTP Code	Description	Schema
201	CREATED	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.22. TAG

Manage the tags of a repository.

### 2.22.1. restoreTag

Restores a repository tag back to a previous image in the repository.

**POST** `/api/v1/repository/{repository}/tag/{tag}/restore`

**Authorizations:** `oauth2_implicit (repo:write)`

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>tag</b> <i>required</i>	The name of the tag	string

#### Request body schema (application/json)

Restores a tag to a specific image

Name	Description	Schema
<b>manifest_digest</b> <i>required</i>	If specified, the manifest digest that should be used	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.22.2. changeTag

Change which image a tag points to or create a new tag.

**PUT /api/v1/repository/{repository}/tag/{tag}**

**Authorizations:** `oauth2_implicit (repo:write)`

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>tag</b> <i>required</i>	The name of the tag	string

#### Request body schema (application/json)

Makes changes to a specific tag

Name	Description	Schema
<b>manifest_digest</b> <i>optional</i>	(If specified) The manifest digest to which the tag should point	
<b>expiration</b> <i>optional</i>	(If specified) The expiration for the image	

#### Responses

HTTP Code	Description	Schema
201	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>

HTTP Code	Description	Schema
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.22.3. deleteFullTag

Delete the specified repository tag.

**DELETE** /api/v1/repository/{repository}/tag/{tag}

**Authorizations:** oauth2\_implicit (repo:write)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string
path	<b>tag</b> <i>required</i>	The name of the tag	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.22.4. listRepoTags

**GET** /api/v1/repository/{repository}/tag/

**Authorizations:** oauth2\_implicit (repo:read)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

### Query parameters

Type	Name	Description	Schema
query	<b>onlyActiveTags</b> <i>optional</i>	Filter to only active tags.	boolean
query	<b>page</b> <i>optional</i>	Page index for the results. Default 1.	integer
query	<b>limit</b> <i>optional</i>	Limit to the number of results to return per page. Max 100.	integer
query	<b>filter_tag_name</b> <i>optional</i>	Syntax: <op><name> Filters the tag names based on the operation.<op> can be 'like' or 'eq'.	string
query	<b>specificTag</b> <i>optional</i>	Filters the tags to the specific tag.	string

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.23. TEAM

Create, list and manage an organization's teams.

### 2.23.1. getOrganizationTeamPermissions

Returns the list of repository permissions for the org's team.

**GET /api/v1/organization/{orgname}/team/{teamname}/permissions****Authorizations:****Path parameters**

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.23.2. updateOrganizationTeamMember**

Adds or invites a member to an existing team.

**PUT /api/v1/organization/{orgname}/team/{teamname}/members/{membername}****Authorizations:** oauth2\_implicit (**org:admin**)**Path parameters**

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>membername</b> <i>required</i>	The username of the team member	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.23.3. deleteOrganizationTeamMember

Delete a member of a team.

If the user is merely invited to join the team, then the invite is removed instead.

**DELETE** /api/v1/organization/{orgname}/team/{teamname}/members/{membername}  
**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>membername</b> <i>required</i>	The username of the team member	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>

HTTP Code	Description	Schema
404	Not found	<a href="#">ApiError</a>

### 2.23.4. getOrganizationTeamMembers

Retrieve the list of members for the specified team.

GET /api/v1/organization/{orgname}/team/{teamname}/members

Authorizations: oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Query parameters

Type	Name	Description	Schema
query	<b>includePending</b> <i>optional</i>	Whether to include pending members	boolean

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.23.5. inviteTeamMemberEmail

Invites an email address to an existing team.

**PUT /api/v1/organization/{orgname}/team/{teamname}/invite/{email}****Authorizations:** oauth2\_implicit (org:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>email</b> <i>required</i>		string
path	<b>teamname</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.23.6. deleteTeamMemberEmailInvite**

Delete an invite of an email address to join a team.

**DELETE /api/v1/organization/{orgname}/team/{teamname}/invite/{email}****Authorizations:** oauth2\_implicit (org:admin)**Path parameters**

Type	Name	Description	Schema
path	<b>email</b> <i>required</i>		string
path	<b>teamname</b> <i>required</i>		string
path	<b>orgname</b> <i>required</i>		string



**Responses**

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.23.7. updateOrganizationTeam**

Update the org-wide permission for the specified team.

**PUT** `/api/v1/organization/{orgname}/team/{teamname}`

**Authorizations:** `oauth2_implicit (org:admin)`

**Path parameters**

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

**Request body schema (application/json)**

Description of a team

Name	Description	Schema
<b>role</b> <i>required</i>	Org wide permissions that should apply to the team	string
<b>description</b> <i>optional</i>	Markdown description for the team	string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	

HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.23.8. deleteOrganizationTeam

Delete the specified team.

**DELETE** /api/v1/organization/{orgname}/team/{teamname}

**Authorizations:** oauth2\_implicit (org:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>teamname</b> <i>required</i>	The name of the team	string
path	<b>orgname</b> <i>required</i>	The name of the organization	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.24. TRIGGER

Create, list and manage build triggers.

### 2.24.1. activateBuildTrigger

Activate the specified build trigger.

**POST** /api/v1/repository/{repository}/trigger/{trigger\_uuid}/activate

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Name	Description	Schema
<b>config</b> <i>required</i>	Arbitrary json.	object
<b>pull_robot</b> <i>optional</i>	The name of the robot that will be used to pull images.	string

#### Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.2. listTriggerRecentBuilds

List the builds started by the specified trigger.

**GET** /api/v1/repository/{repository}/trigger/{trigger\_uuid}/builds

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

### Query parameters

Type	Name	Description	Schema
query	<b>limit</b> <i>optional</i>	The maximum number of builds to return	integer

### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.3. manuallyStartBuildTrigger

Manually start a build from the specified trigger.

**POST** /api/v1/repository/{repository}/trigger/{trigger\_uuid}/start

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

**Request body schema (application/json)**

Optional run parameters for activating the build trigger

Name	Description	Schema
<b>branch_name</b> <i>optional</i>	(SCM only) If specified, the name of the branch to build.	string
<b>commit_sha</b> <i>optional</i>	(Custom Only) If specified, the ref/SHA1 used to checkout a git repository.	string
<b>refs</b> <i>optional</i>	(SCM Only) If specified, the ref to build.	

## Responses

HTTP Code	Description	Schema
201	Successful creation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.4. getBuildTrigger

Get information for the specified build trigger.

**GET** /api/v1/repository/{repository}/trigger/{trigger\_uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.5. updateBuildTrigger

Updates the specified build trigger.

**PUT** /api/v1/repository/{repository}/trigger/{trigger\_uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Request body schema (application/json)

Options for updating a build trigger

Name	Description	Schema
<b>enabled</b> <i>required</i>	Whether the build trigger is enabled	boolean

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>

HTTP Code	Description	Schema
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.6. deleteBuildTrigger

Delete the specified build trigger.

**DELETE** /api/v1/repository/{repository}/trigger/{trigger\_uuid}

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>trigger_uuid</b> <i>required</i>	The UUID of the build trigger	string
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.24.7. listBuildTriggers

List the triggers for the specified repository.

**GET** /api/v1/repository/{repository}/trigger/

**Authorizations:** oauth2\_implicit (repo:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

## Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

## 2.25. USER

Manage the current user.

### 2.25.1. createStar

Star a repository.

**POST /api/v1/user/starred**

**Authorizations:** oauth2\_implicit (repo:read)

**Request body schema (application/json)**

Name	Description	Schema
<b>namespace</b> <i>required</i>	Namespace in which the repository belongs	string
<b>repository</b> <i>required</i>	Repository name	string

## Responses

HTTP Code	Description	Schema
201	Successful creation	



HTTP Code	Description	Schema
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.25.2. listStarredRepos

List all starred repositories.

**GET /api/v1/user/starred**

**Authorizations:** oauth2\_implicit (**user:admin**)

#### Query parameters

Type	Name	Description	Schema
query	<b>next_page</b> <i>optional</i>	The page token for the next page	string

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.25.3. getLoggedInUser

Get user information for the authenticated user.

**GET /api/v1/user/**

**Authorizations:** oauth2\_implicit (**user:read**)

#### Responses

HTTP Code	Description	Schema
200	Successful invocation	<a href="#">UserView</a>
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.25.4. deleteStar

Removes a star from a repository.

**DELETE** /api/v1/user/starred/{repository}

**Authorizations:** oauth2\_implicit (user:admin)

#### Path parameters

Type	Name	Description	Schema
path	<b>repository</b> <i>required</i>	The full path of the repository. e.g. namespace/name	string

#### Responses

HTTP Code	Description	Schema
204	Deleted	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

### 2.25.5. getUserInformation

Get user information for the specified user.

**GET** /api/v1/users/{username}

**Authorizations:**

**Path parameters**

Type	Name	Description	Schema
path	<b>username</b> <i>required</i>		string

**Responses**

HTTP Code	Description	Schema
200	Successful invocation	
400	Bad Request	<a href="#">ApiError</a>
401	Session required	<a href="#">ApiError</a>
403	Unauthorized access	<a href="#">ApiError</a>
404	Not found	<a href="#">ApiError</a>

**2.26. DEFINITIONS****2.26.1. ApiError**

Name	Description	Schema
<b>status</b> <i>optional</i>	Status code of the response.	integer
<b>type</b> <i>optional</i>	Reference to the type of the error.	string
<b>detail</b> <i>optional</i>	Details about the specific instance of the error.	string
<b>title</b> <i>optional</i>	Unique error code to identify the type of error.	string
<b>error_message</b> <i>optional</i>	Deprecated; alias for detail	string
<b>error_type</b> <i>optional</i>	Deprecated; alias for detail	string

**2.26.2. UserView**

Name	Description	Schema
<b>verified</b> <i>optional</i>	Whether the user's email address has been verified	boolean
<b>anonymous</b> <i>optional</i>	true if this user data represents a guest user	boolean
<b>email</b> <i>optional</i>	The user's email address	string
<b>avatar</b> <i>optional</i>	Avatar data representing the user's icon	object
<b>organizations</b> <i>optional</i>	Information about the organizations in which the user is a member	array of object
<b>logins</b> <i>optional</i>	The list of external login providers against which the user has authenticated	array of object
<b>can_create_repo</b> <i>optional</i>	Whether the user has permission to create repositories	boolean
<b>preferred_namespace</b> <i>optional</i>	If true, the user's namespace is the preferred namespace to display	boolean

### 2.26.3. ViewMirrorConfig

Name	Description	Schema
<b>is_enabled</b> <i>optional</i>	Used to enable or disable synchronizations.	boolean
<b>external_reference</b> <i>optional</i>	Location of the external repository.	string
<b>external_registry_username</b> <i>optional</i>	Username used to authenticate with external registry.	
<b>external_registry_password</b> <i>optional</i>	Password used to authenticate with external registry.	
<b>sync_start_date</b> <i>optional</i>	Determines the next time this repository is ready for synchronization.	string

Name	Description	Schema
<b>sync_interval</b> <i>optional</i>	Number of seconds after next_start_date to begin synchronizing.	integer
<b>robot_username</b> <i>optional</i>	Username of robot which will be used for image pushes.	string
<b>root_rule</b> <i>optional</i>	A list of glob-patterns used to determine which tags should be synchronized.	object
<b>external_registry_config</b> <i>optional</i>		object

#### 2.26.4. ApiErrorDescription

Name	Description	Schema
<b>type</b> <i>optional</i>	A reference to the error type resource	string
<b>title</b> <i>optional</i>	The title of the error. Can be used to uniquely identify the kind of error.	string
<b>description</b> <i>optional</i>	A more detailed description of the error that may include help for fixing the issue.	string

## CHAPTER 3. API CONFIGURATION EXAMPLES

### 3.1. EXTERNAL\_REGISTRY\_CONFIG OBJECT REFERENCE

```
{
  "is_enabled": True,
  "external_reference": "quay.io/redhat/quay",
  "sync_interval": 5000,
  "sync_start_date": datetime(2020, 0o1, 0o2, 6, 30, 0),
  "external_registry_username": "fakeUsername",
  "external_registry_password": "fakePassword",
  "external_registry_config": {
    "verify_tls": True,
    "unsigned_images": False,
    "proxy": {
      "http_proxy": "http://insecure.proxy.corp",
      "https_proxy": "https://secure.proxy.corp",
      "no_proxy": "mylocalhost",
    },
  },
}
```

### 3.2. RULE\_RULE OBJECT REFERENCE

```
{
  "root_rule": {"rule_kind": "tag_glob_csv", "rule_value": ["latest", "foo", "bar"]},
}
```