The Bridge API

How-To Manual for using The Bridge API

Updated April 22, 2016

Table of Contents

Getting Started

CollegiateLink API Docs

Throughout this guide and on The Bridge API itself, we may refer to the CollegiateLink API and its own set of documentation. As of April 22, 2016, you can find those docs at

http://support.collegiatelink.net/hc/en-us/articles/204033994-Documentation-for-Web-Services-API-Version-2-0-

To find information about a specific endpoint (e.g. Events), find and click on the '/api/events' link. This page will show you the various parameters developers can use, and what values they accept, when requesting data from that endpoint. Except for 'page' and 'pageSize', these are what we call Filters in our system. Towards the bottom of the page, you can see an example of the data a developer might get in response from the endpoint.

Logging In
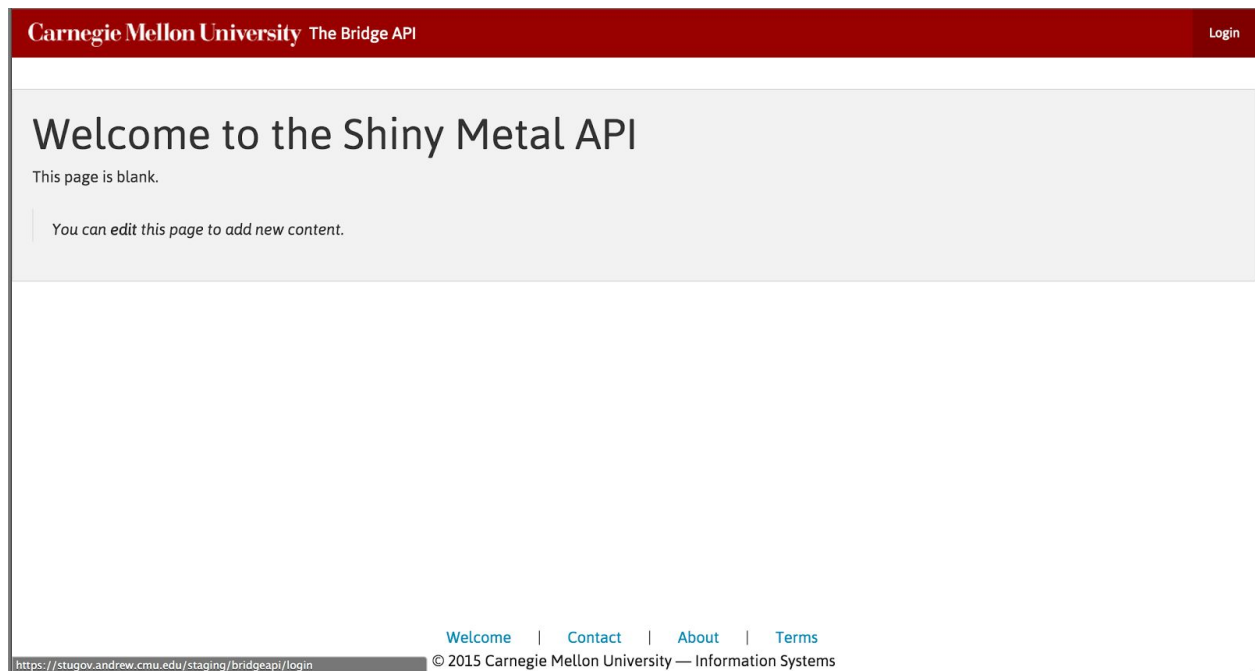
Click login to get started. It will redirect you to Shibboleth login

Upon successful login, you'll see your homepage

But, if you try and log in as an inactive (banned) user, the application will reject your login

Viewing Pending Keys or Keys Requiring Approval

After requesters submit their application, the application will then enter the administrator review process. All keys at this stage in the process will be listed under the table titled "Keys Awaiting Administrator Review". Before these keys can be sent to other administrators for approval and released, they will first have to be reviewed by an administrator. Appropriate action must be taken based upon the responses to the application questions.

Welcome, Benjamin

Keys Awaiting Approval or Confirmation

All keys that have been made public to approvers have been approved and confirmed.

Keys Awaiting Administrator Review

Key	Requester	Date Submitted
Ben's Key	bklam	8:20 PM, April 25 2015

Users

Introduction

This is the users dashboard. All of the users in the system will be listed in this panel. It is separated into two tables, with users who have requested/have an API key on the left and administrators and staff on the right.

Viewing A Specific User

A specific user's information can be viewed simply by clicking on the user's name

All Users

Requesters

Andrew	Role	Applications	Currently has a key with API access?	Active?	
mdf	Requester	0	No	Yes	↻

Staff

Andrew	Role	Applications	Currently has a key with API access?	Active?	
jkcorrea	Staff	0	No	No	↻
aditisar	Administrator (Approver)	1	Yes	Yes	↻
bklam	Administrator (Approver)	0	No	Yes	↻

In this example, clicking on aditisar will take us to the information page for aditisar. Included information for a user is the name, andrewID, role, active status, and any API keys that may belong to that given user. This doubles as a link to find out more information on an API key

Name: Aditi Sarkar

Andrew Id: aditisar

Role: Administrator (Approver)

Active: Yes

1 User Key

• test email

[Edit user] | [Back to users list]

https://stugov.andrew.cmu.edu/staging/bridgeapi/home

Editing Users

Users can be edited by clicking the paper and pencil icon directly to the right of each name

Clicking on the icon will lead to the editing page for users

A user has a specific role in the system. This can be changed by clicking the drop down menu and selecting another role, pictured below:

User Keys

Introduction

This is the dashboard for administrators to take a look at all of the User Keys that have been issued in the system. It will list the name of each requested user key, who it was requested by, when the initial application was submitted, when an application is ready for approval, when the application has been confirmed by all necessary staff members, the expiration date for the key, and whether or not a key is active and able to be used by the requester. A user key is an API key in the system that can be given to requesters that they can use to pull only specific data from The Bridge.

Viewing Pending Keys/Keys Requiring Approval

This has already been covered in the introduction with the home page, but here it is listed again for your convenience. This home page simply lists all of the applications for keys that have been submitted and the current status of the applications, whether the applications have been reviewed yet or if they've been approved yet.

Welcome, Benjamin

Keys Awaiting Approval or Confirmation

All keys that have been made public to approvers have been approved and confirmed.

Keys Awaiting Administrator Review

Key	Requester	Date Submitted
Ben's Key	bklam	8:20 PM, April 25 2015

Home | Welcome | Contact | About | Terms
© 2015 Carnegie Mellon University — Information Systems

Reviewing Keys & Sharing To Admins for Approval

Clicking on the names of keys from the approval/review dashboard will lead to a details page for a specific API key application. This example followed the link title "Ben's Key". Here, the answers to the application questions can be reviewed by an administrator and appropriate steps can be taken. An admin can choose to share the key application with other approvers for approval but only after editing the allowed permissions for a specific key based on the answers to the application questions and setting an expiration date. An admin can also choose to remand an application, sending the application back to the requester for modifications. The requester will also be notified by email that the application has been remanded so he/she will know to make modifications.

Releasing a key for approval will also notify staff members that they should take a look and approve the key

Remanding A Key

Remanding a Key will notify the requester that his/her application has been sent back for further modification with an email similar to below.
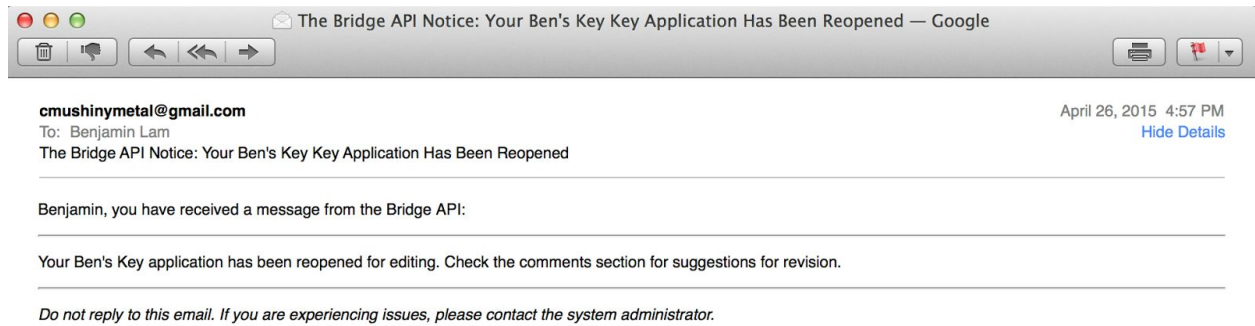
It will send the application status for a requester back to the initial state, pictured below:
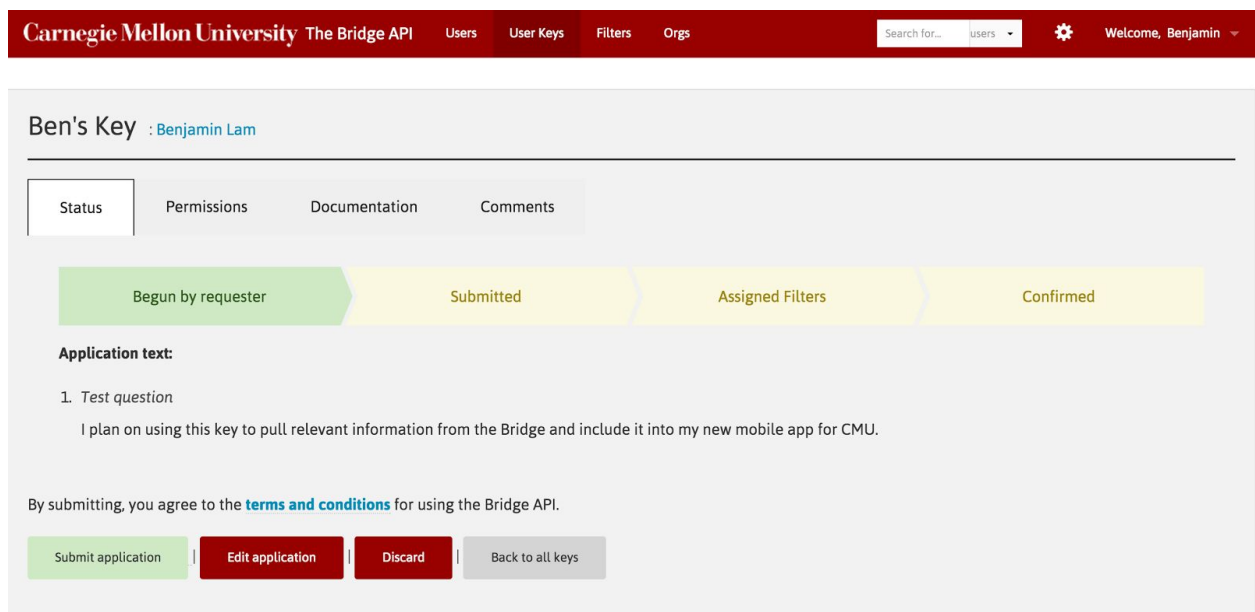
Assigning Permissions to New User Key Applications

After reviewing the answers to the application, the next step for the administrator is to assign permissions allowed for the given requester. The administrator can assign these access rights by clicking on the "Edit access rights" button pictured below
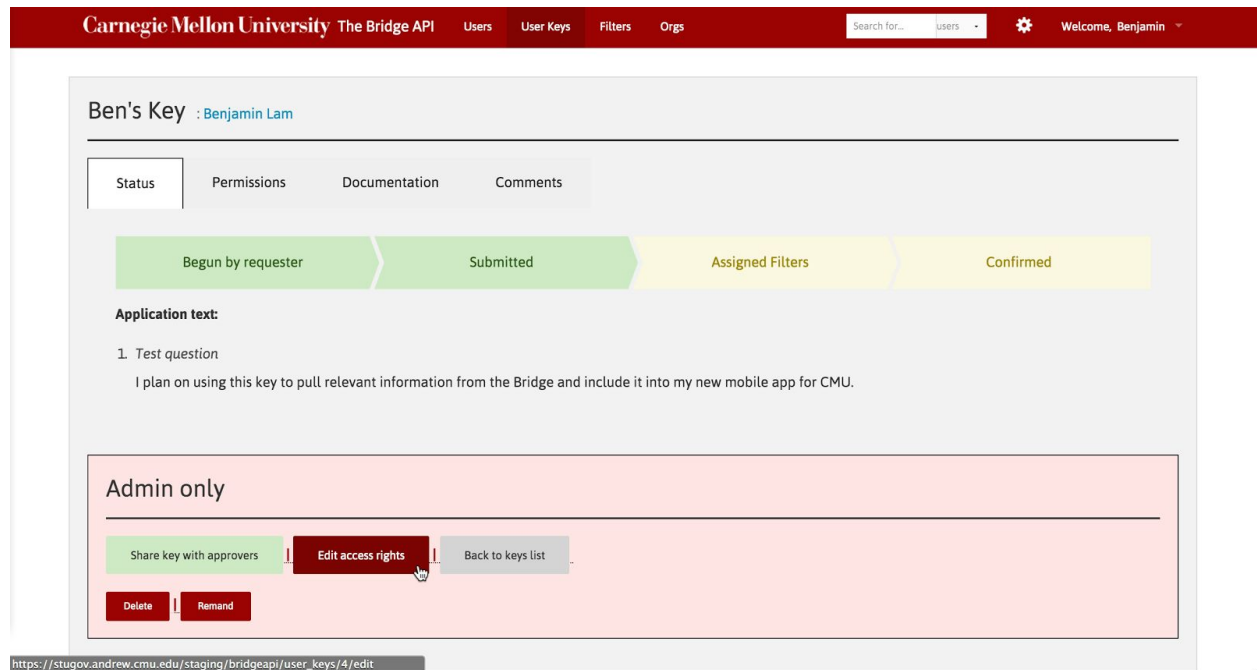
The permissions panel will then pop up, pictured below.
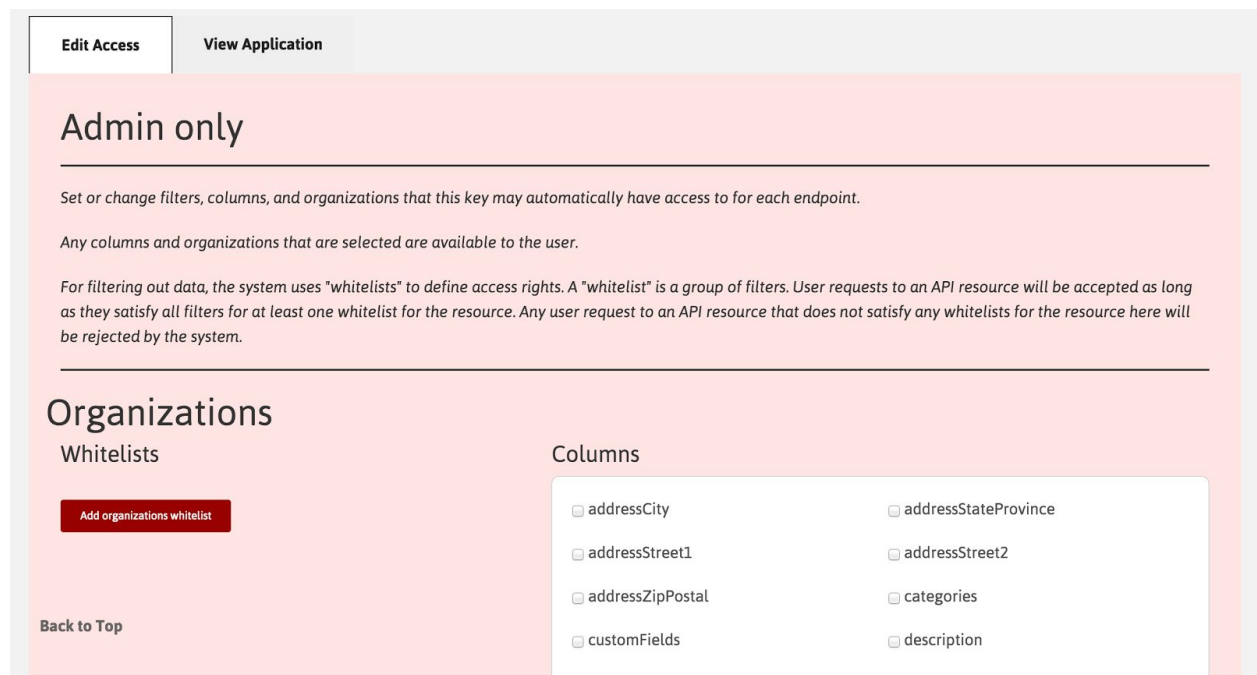
Whitelists and Columns

Whitelists are the administrator's method of controlling which records will show up when requesters hit the Bridge API with an API key. The system will only allow requesters to hit

endpoints with specific parameters and specific values for the parameters. This is determined by which API key is used, so that the system will allow for different users to have access only to pertinent or allowed information. <u>The allowed filters and filter values in the system are set in the filters dashboard</u> (covered later).
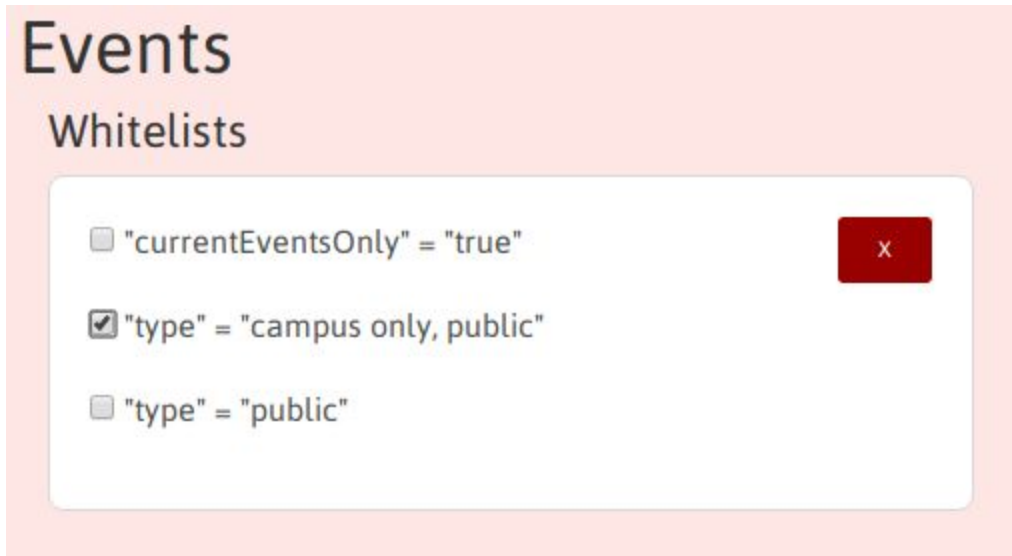
The system uses whitelists and groups of whitelists to allow for flexibility in permissions of a user key. **Whitelists are represented in the permissions page as white boxes containing filters.** Each white box is a separate whitelist, and each checked checkbox tells the system that that filter/value combo must be present in a user's request to fulfill that whitelist. A very basic example of using a whitelist is to only allow a user to request data that is filtered by a certain value:

In the above example, the user will **only** be able to retrieve data for organizations that are not hidden (i.e. their requests to the API will only work when they have "excludeHiddenOrganizations=true" in their request).

And in the next example, the user will **only** be able to retrieve data on events that are either 'public' or 'campus only' (i.e. their requests will have to include either 'type=public' or 'type=campus only'):

Finally, if we wanted the user to **either** make requests for events that are public, OR make requests for events that are current, OR make requests for events that are both public and current, we would use a group of whitelists.

In the example below, the first whitelist we create satisfies the first part of that above statement: allow the user to make requests for events that are public. The second whitelist allows the user to make requests for events that are current. And the third whitelist allows the user to make requests for events that are both current and public.

When the user makes his/her request, their request must match at least one of the whitelists present.

Events

Whitelists

☐ "currentEventsOnly" = "true"
☐ "name" = ""
☐ "organizationId" = ""
☑ "type" = "public"

[X]

☑ "currentEventsOnly" = "true"
☐ "name" = ""
☐ "organizationId" = ""
☐ "type" = "public"

[X]

☑ "currentEventsOnly" = "true"
☐ "name" = ""
☐ "organizationId" = ""
☑ "type" = "public"

[X]

[Add events whitelist]

Columns

☐ addressCity
☐ addressStreet1
☐ addressZipPostal
☐ allowSelfReportAttendance
☐ approvedByUserId
☐ approvedOn
☐ customFields
☐ endDateTime
☑ eventName
☐ externalLocationId
☐ locationId
☑ organizationId
☐ otherLocation
☐ rsvpOption
☑ status
☐ submittedByUserId
☐ submittedOn
☐ typeId

☐ addressStateProvince
☐ addressStreet2
☐ allowOnTranscript
☐ approvedByUserCampusEmail
☐ approvedByUsername
☐ categories
☑ description
☐ eventId
☑ eventUrl
☑ flyerUrl
☐ locationName
☐ organizationName
☐ rsvpMaxSpotsAllowed
☑ startDateTime
☐ submittedByUserCampusEmail
☐ submittedByUsername
☐ thumbnailUrl
☑ typeName

Lastly, in this example, the columns allowed in the response are only the ones that are checked in the panel. This user will only be able to get records with the eventName, status, organizationId, description, eventUrl, flyerUrl, startDateTime, and typeName columns.

Special Access For Organizations

This panel is used to give information privileges to a user if they are a member of a particular organization or club. For instance, if a user is in AB Tech, a user should be able to see all private events associated with AB Tech, even if the user is only allowed to see public events.

Special Access for Checked Organizations

☐ 1000plus Day of Service	☐ 5th Year Scholar Program	☐ 90/30 Fund	☐ AB Coffeehouse
☐ AB Comedy	☐ AB Concerts	☐ AB Films	☐ AB Lectures
☐ AB Political Speakers	☐ AB Publications	☐ AB Skibo	☐ AB Special Events
☐ AB Tech	☐ AB Underground	☐ AC Mellon Men's Soccer Team	☐ ACM@CMU
☐ Active Minds at Carnegie Mellon	☐ Activities Board	☐ Activities Fair	☐ Actuarial Club
☐ Advancing College Essays	☐ Agape Christian Ministry	☐ AISH	☐ All University Orchestra
☐ ALLIES	☐ Allies Grad	☐ Alpha Chi Omega	☐ Alpha Epsilon Delta
☐ Alpha Epsilon Pi	☐ Alpha Kappa Alpha Sorority Inc.	☐ Alpha Kappa Delta Phi	☐ Alpha Kappa Psi Professional

Comments

Administrators are able to add comments to a user key application through the comments panel, either for other staff members or directly to the applicant.

Carnegie Mellon University The Bridge API   Users   User Keys   Filters   Orgs   Search for... users ▾   ⚙   Welcome, Benjamin ▾

Ben's Key   : Benjamin Lam

Status	Permissions	Documentation	Comments

Messages from Administrator
None

Staff Discussion (Private)
None

Add a comment

Message

https://stugov.andrew.cmu.edu/staging/bridgeapi/user_keys/4#commentsPanel

In the example below, a message is being sent to the requester.

| Status | Permissions | Documentation | **Comments** |

Messages from Administrator
None

Staff Discussion (Private)
None

Add a comment

Message

Wow, this is an impressive application! Let's try and get this key out to you as fast as possible|

Check here to post this comment as a note to the requester.   ☑

Add comment

Comments can also be left between staff members if the check box is left unchecked. These are **private** messages that only staff and administrators can see. An example is pictured below for the user key application.

Messages from Administrator

Administrator   8:24 PM, April 25 2015   [x]
Wow, this is an impressive application! Let's try and get this key out to you as fast as possible

Staff Discussion (Private)
None

Add a comment

Message

I think there are still a few people who haven't approved this fine young man's key yet. Could we try and hurry this process please?|

Check here to post this comment as a note to the requester.   ☐

Add comment

Back to Top

Expiration Dates

Expiration dates must be set before a key can be released to other staff members to approve. This is located at the bottom of the edit access rights panel.

Finish Editing Access Rights

Clicking on "Update application" will save all access rights changes

Release For Approval

Once permissions have been set, it can be released to be approved by the rest of the staff. The user key page breadcrumbs should now have progressed to "assigned filters" and a green notification box should appear at the top of the page.

Confirming Keys

The home page will now show that there are keys waiting for confirmation now that filters have been assigned. Confirming a key can be done from the admin only box on a user key page.

Confirmed Keys

Once all of the approvers have approved a user key, the actual API key value will then be generated and will be visible to the requester.

test email : Aditi Sarkar

| Status | Permissions | Documentation | Comments |

| Begun by requester | Submitted | Assigned Filters | Confirmed |

API Key Value	Expiration Date
f4a78c5eee714568510ba98f0128291b09e613d29acda509c9cad767d700ec5e	May 22, 2015

Application text:

1. *Test question*

balkjfld;s

Documentation

Documentation for a specific API key will be available for both the requester and admin to see. The documentation outlines specifics of how to use the API, along with examples of specially-tailored sample API calls that can be made using the user's andrewID and API key.

| Status | Permissions | Documentation | Comments |

Authentication

All requests to the API require an API key and the corresponding andrewID of the individual who initially requested the API key. The API key is generated pending the approval of the Student Activities Administration.

Your andrewID is: aditisar
Your API key is: 116cbbd0dcb9119fcdbe61f921072088d2697c01deaac828ab06489ab4ff94d5

Making A Request to the Shiny Metal API

All requests to the Shiny Metal API need to be `POST` requests with two header fields: `andrew_id` and `api_key` . Requests follow the same syntax as the CollegiateLink API. Only specific endpoints and combinations of parameters are allowed for specific keys, however, as determined by the Student Activities Office. You can check the set of allowed permissions and parameters for your API key by clicking on the above "Permissions" tab.

The base URL for making API calls is: `https://stugov.andrew.cmu.edu/staging/bridgeapi/api/v1/`

Pagination

All API calls can also have two pagination parameters, `page` and `pageSize` . `page` is a parameter that determines which page of the overall response will be returned. `pageSize` will determine how many records will be returned in one response. In order to get all relevant records, an iterator will need to be constructed in client code to repeatedly request specific pages.

Back to Top

There are examples of test API calls that can be made:

Testing API Credentials

For authentication purposes, there is an endpoint named "demo_endpoint" that is useful for simply testing whether a set of credentials is correct

curl	Ruby	Python	PHP

Sample API Call

```
curl -X POST \
  -H 'andrew_id: aditisar' \
  -H 'api_key: 116cbbd0dcb9119fcdbe61f921072088d2697c01deaac828ab06489ab4ff94d5' \
  'http://localhost:3000/api/v1/demo_endpoint'
```

Expected Response:

```
{"message": "Success! You've made an API request with valid credentials!"}
```

Along with tabs that show requests in different languages

Testing API Credentials

For authentication purposes, there is an endpoint named "demo_endpoint" that is useful for simply testing whether a set of credentials is correct

curl	Ruby	Python	PHP

Sample API Call

```
require 'base64'
require 'openssl'
require 'net/http'
require 'JSON'

uri = URI.parse('https://stugov.andrew.cmu.edu/staging/bridgeapi/api/v1/demo_endpoint')
https = Net::HTTP.new(uri.host, uri.port)
https.use_ssl = true
https.verify_mode = OpenSSL::SSL::VERIFY_NONE
req = Net::HTTP::Post.new(uri)
# set the headers
req['andrew_id'] = 'aditisar'
req['api_key'] = '116cbbd0dcb9119fcdbe61f921072088d2697c01deaac828ab06489ab4ff94d5'
res = https.request(req)
JSON.parse(res.body)
```

Expected Response:

Back to Top

```
{"message": "Success! You've made an API request with valid credentials!"}
```

localhost:3000/user_keys/8#rubyTestRequest

The rest of the documentation panel:

More Sample API Calls

For example, assuming you have access to the `Events` endpoint and can see all `public` events

| curl | Ruby | Python | PHP |

Sample API Call:

```
curl -X POST \
  -H 'andrew_id: aditisar' \
  -H 'api_key: 116cbbd0dcb9119fcdbe61f921072088d2697c01deaac828ab06489ab4ff94d5' \
  'http://localhost:3000/api/v1/events?type=public'
```

Sample Response:

```
{
    "pageNumber": 1,
    "pageSize": 2,
    "totalItems": 1223,
    "totalPages": 612,
    "items": [
        {
            "eventName": "Super Fun Picnic",
            "organizationName": "Cool Organization",
            "endDateTime": "1329048200000",
            "externalLocationId": null,
            "locationId": null,
            "locationName": "",
            "addressStreet1": "",
```

Back to Top

Filters Dashboard

Introduction

Filters are the constraints that can be assigned to a specific user key. Filters are for a specific resource on the Collegiate Link API and can also have a specific value associated with them, such as "public" for "type" on the Events resource.

Viewing A Specific Filter

Clicking on a filter name will lead to a page with more information about that specific filter. Included information is the filter name, along with links to user keys that use the filter.

Carnegie Mellon University The Bridge API   Users   User Keys   Filters   Orgs   Search for... users ⚙ Welcome, Benjamin ▾

Filter Details

Resource: events

Filter name: "type" = "public"

3 User Keys

- test email
- test
- Ben's Key

Back to filters list

Delete filter

Creating/Editing A Filter

Just like users and user keys, new filters can be created by selecting a resource, a filter name, and a specific value for the filter. The filter names match with filter names for the CollegiateLink API, and the filter value can be set manually. Using the example below, if you a filter to only allow requests for a specific organizationId of 1085, you could set the filter name to 'organizationId' and filter value to '1085'.

Additionally, multiple filter values can be set by using commas to separate them. Using the same example, if you wanted to let requests in for organizationId's 1085 through 1090 you would enter a filter value of '1085,1086,1087,1088,1089,1090'.

Finally, if you wanted to let any value in, you can use the wildcard character '*' as the filter value.

Organizations Dashboard

Introduction

Organizations are the organizations a user can belong to in the system. The dashboard is pictured below.

Viewing A Specific Organization

Just like users, user keys, and filters, clicking the organization name is a link that leads to a page with more detailed information regarding a specific organization.

Requester Perspective

Requester Dashboard

Requesting a New Key

From a requester perspective, If you'd like to start a new application for an API key, clicking on New User Key from the dashboard will lead to a new application page, where you can fill in the appropriate responses to the questions.

After submitting this, the requester will see a finalization page to submit their application

Once you have reviewed your application, you can either go back and edit your application or submit the application to be viewed by administrators and receive approval.

The page will then look like this once you have submitted your request

The requester will then wait for the administrator to assign the correct access rights and finalize the application process. The requester will receive a confirmation email like below after submitting their application.

And the dashboard will now look like this, since there is now a pending request:

Welcome, Benjamin

Apply for a new key

Your Existing Keys

None

Your Incomplete Applications

None

Your Pending Key Requests

Key	Date Started
Ben's Key	8:17 PM, April 25 2015

Your Expired Keys

Back to Top None

From this point on, the requester will receive email notifications once any progress has been made on the application, whether it has been accepted/approved or it has been remanded.