Product Setup and Management

**Security**

* An application MUST have a dedicated set of keys for the QA environment.
* An application MUST have a dedicated set of keys for the PROD environment.
* Application Keys cannot be shared between the QA and PROD environments.

**Product Setup in Apigee Edge**

1. Products in Apigee Edge will be set up using the Apigee Edge UI.
2. Products in Apigee Edge will be set up and managed by a product owner only.
3. Every product will be associated to one and only one environment on Apigee Edge:
   1. Environment: QA
   2. Environment: PROD
4. Apigee Edge products without an associated environment or with multiple environments will not be displayed to app developers in the developer portal.
5. Student Services will have dedicated products on Apigee Edge categorized as “STUDENT\_SERVICE\_PRODUCT\_TYPE” and identified through association with a custom attribute. In this case, the custom attribute is a name/value pair of:
   1. Name: PRODUCT\_TYPE
   2. Value: STUDENT\_SERVICE
6. Student Services products can be associated with both PROD and QA environments or, alternatively, associated with the QA environment only. The latter case is intended to be used for a cases where an API product is under development and not ready for production deployment.

For example: As a student services product owner, I want to create PRODUCT A on Apigee Edge. In this case, the following syntax will be applicable: In the QA environment, the product name is “PRODUCT\_A\_QA” and in the PROD environment, the product name is “PRODUCT\_A\_PROD”.

1. Products unrelated to Student Services will be categorized as “OTHER\_PRODUCT\_TYPE” and also identified through association with a custom attribute. In this case, the custom attribute is a name/value pair of:
   1. Name: PRODUCT\_TYPE
   2. Value: OTHER
2. Products without the custom attribute “PRODUCT\_TYPE” existing or defined will be classified as “OTHER\_PRODUCT\_TYPE” by default.
3. Products created on Apigee Edge will utilize the following naming convention: [PRODUCT\_NAME]\_[ENV\_NAME] where
   1. PRODUCT\_NAME: Name of the Product
   2. ENV\_NAME: Name of the Apigee Edge environment to which the product is associated
4. App domains and app subdomains will be named in Apigee Edge UI using the following convention:
   1. Name: APP\_DOMAIN\_NAME
   2. Value: Predefined list of values [Example: Student]
   3. Name: APP\_SUB\_DOMAIN\_NAME
   4. Value: Predefined list of values [Example: Classes]
   5. Name: PORTAL\_VISIBILITY
   6. Value: Predefined list of values [Not Logged In, Logged In, Not Visible]
5. Products created on Apigee Edge will utilize APP\_DOMAIN\_NAME and APP\_SUB\_DOMAIN\_NAME for category mapping and portal navigation.
6. Products without the custom attribute “PORTAL\_VISIBILITY” existing or defined will be treated as “Not Visible” by default in the developer portal.

**Product Setup in the Developer Portal**

1. Products for app developers are created in the developer portal
2. Every product on Apigee Edge intended to be surfaced to app developers will have an entry in the developer portal API catalog
3. A user with the role of product owner will create the API catalog entry in the developer portal
4. An API catalog entry will have the characteristics shown in Table 2.1
5. The API RESOURCES field will fetch, and list all published API DOCUMENT ENTITY names. This field value will have one or more than one data mapping and will be configured manually by the product owner.
6. Empty API RESOURCES will have “inactive” status and will not display into the portal.
7. Product status cannot be changed manually. Any mandatory field not fetched as expected will default to “inactive” status.
8. Duplicate product attribute information be created in the developer portal.

Table 2.1:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Field Name | Required | Editable | Data Source | Description |
| 1 | Product Name | Yes | No | A single select list box - will list of products on Apigee Edge associated to one environment | Prepopulated drop down listing all available products from Edge through a management API call. |
| 2 | Product Type | Yes | No | Apigee Edge: Product | Value based on the product name selected in row #1 - populated using the value of the attribute “PRODUCT\_TYPE” |
| 3 | Environment | Yes | No | Apigee Edge: Product | Value based on the product name and product type selected in row #1 and row #2 - populated using the value of the attribute “ENVIRONMENT” |
| 4 | Description | Yes | No | Apigee Edge: Product | Value based on the product name and product type selection in row #1, row #2, and row #3 - populated using the value of the attribute “DESCRIPTION” |
| 5 | Product Domain Name | Yes | No | Apigee Edge: Product | Value based on the product name and product type selection in row #1, row #2, and row #3 - populated using the value of the attribute “APP\_DOMAIN\_NAME” |
| 6 | Product Subdomain | Yes | No | Apigee Edge: Product | Value based on the product name and product type selection in row #1, row #2, and row #3 - populated using the value of the attribute “APP\_SUB\_DOMAIN\_NAME” |
| 7 | Portal Visibility | Yes | No | Apigee Edge: Product | Value based on the product name and product type selection in row #1, row #2, and row #3 - populated using the value of the attribute “PORTAL\_VISIBILITY” |
| 8 | Image | No | Yes | Developer Portal | A default configuration image which a product owner will able to replace via image upload – images must meet the following criteria:  1. One file only.  2. 2 MB limit.  3. Allowed types: jpg & gif  4. Dimensions TBD |
| 9 | API RESOURCES [e.g., Swagger Docs] | Yes | Yes | Developer Portal | Entity reference value from API DOCUMENT ENTITY |

**Configuring Swagger Docs on Developer Portal**

1. API Resource documents for Application Consumers will be created on the developer portal.
2. Every API resource documents that should be exposed to App Consumer will have a API RESOURCES associated to API Catalog Entry on Developer Portal.
3. Every API catalog entry may have multiple API RESOURCES.
4. A User with the role of a Product owner will create the API RESOURCES Entry in the developer portal.
5. An API RESOURCES Entry will have the characteristics as shown in Table 2.2

Table: 2.2:

|  |  |  |
| --- | --- | --- |
| Field Name | Required | Description |
| Name | Yes |  |
| OPEN API SPECIFICATION FILE | Yes | Specification source type:  - File  - URL  If the File option is selected, it should meet the following validation criteria.  - One file only.  - 2 MB limit.  - Allowed types: yml, yaml, json  If the URL option is selected, it should meet the following validation criteria:  Enter the full file URL, such as http://example.com/doc.yml - allowed extensions: yml, yaml, json |
| Description | No | Textarea field limited to 256 characters |
| Publishing Status | No | If this field is not selected, the swagger document(s) will be considered not yet published |

Role Setup and Management in Developer Portal

**Super Admin**

* Manage content pages.
* Assign role to Portal Admin.
* Super admin will login through the Drupal login screen.

**Portal Admin**

* Manage content pages.
* Create/edit/remove app domain.
* Add/remove app domain owner.
* Add/edit/delete teams.
* Add/edit/remove team owner.
* Assign privileges to team members to add/edit/delete apps.

**App Domain Owner**

* Add/remove subdomain owner.
* Manage domain content, update document and product information within the domain.

**App Sub Domain Owner**

* Add/remove sub domain owner
* Manage subdomain content, update document and product information within the subdomain.

**Team Owner**

* Add team member.
* Remove team member.
* Add team app and individual app.

**Team Member**

* Add team app and individual app

**Individual Member**

* On a user’s first shibboleth portal authentication, an individual member account will be created and assigned to the authenticated user.

With the exception of super admin all accounts will be created via shibboleth authentication.

Authorization will be managed through developer portal roles. There are no attributes available through shibboleth to manage authorization.

Shibboleth will always provide authentication. Authorization will be based on roles assigned to portal users within the developer portal.

Once successfully authenticated via shibboleth, a developer portal account will be created with the following user information:

* First name

* Last Name
* Email Address
* Unique ID

App Creation Workflow

**App Creation Workflow for OTHER\_PRODUCT\_TYPE Products:**

“My Apps” will be visible to app developers under “Apps” in the developer portal navigation.

Selecting “My Apps” will show a “Create App” button and list all existing apps associated with the individual app developer.

Clicking “Create App” navigates to an app creation form. The following data entry fields will be displayed in this form:

|  |  |  |
| --- | --- | --- |
| Field Name | Required | Description |
| App Name | Yes | Apps created on Apigee Edge will adopt the following naming convention:  Individual Member:  [UNIQUENAME\_PRODUCT\_TYPE\_ENVIRONMENT]  Team Member:  [TEAMNAME\_UNIQUENAME\_PRODUCT\_TYPE\_ENVIRONMENT] |
| Product Type | Yes | “Product Type” is a radio button with two distinct options:  1. Student Services Product Type  2. Other Product Type  “Other Product Type” must be chosen to create an “OTHER\_PRODUCT\_TYPE” app |
| Environment | Yes | “Environment” is a drop down select with two distinct options:  1. QA  2. PROD |
| Description | No | “Description” is a textarea limited to 256 characters |
| Select Product | Yes | The “Select Product” button, when clicked, will launch a modal (i.e., lightbox) window listing all available products based on previous inputs provided through the “Product Type” and “Environment” fields - Single or multiple products may be chosen - A given product may not be chosen more than once for a single app |

**App Creation Workflow for STUDENT\_PRODUCT\_TYPE Products:**

App developers will able to see the “My Apps” under “Apps” in the Developer Portal navigation.

Selecting “My Apps” will show a “Create App” button and list all existing apps associated with the individual app developer.

Clicking “Create App” navigates to an app creation form. The following data entry fields will be displayed in this form:

|  |  |  |
| --- | --- | --- |
| Field Name | Required | Description |
| App Name | Yes | Apps created on Apigee Edge will adopt the following naming convention:  Individual Member:  [UNIQUENAME\_PRODUCT\_TYPE\_ENVIRONMENT]  Team Member:  [TEAMNAME\_UNIQUENAME\_PRODUCT\_TYPE\_ENVIRONMENT] |
| Product Type | Yes | “Product Type” is a radio button with two distinct options:  1. Student Services Product Type  2. Other Product Type  “Other Product Type” must be chosen to create an “OTHER\_PRODUCT\_TYPE” app |
| Environment | Yes | “Environment” is a drop down select with two distinct options:  1. QA  2. PROD |
| Description | No | “Description” is a textarea limited to 256 characters |
| Select Product | Yes | The “Select Product” button, when clicked, will launch a modal (i.e., lightbox) window listing all available products based on previous inputs provided through the “Product Type” and “Environment” fields - Single or multiple products may be chosen - A given product may not be chosen more than once for a single app - Upon form submission, if the value of the “Environment” field is “PROD”, two apps (QA and PROD) will be created in Apigee Edge and the developer portal - If the app for QA already exists, the app will be created only for PROD and selected products will added into the existing QA app - Upon form submission, if the value of the “Environment” field is “QA”, only one app (QA) will be created in Apigee Edge and the Developer Portal. |

This app creation flow will be applicable for the following portal roles: individual developer, team member, and team owner.

A user acting in the team member role must navigate through “My Teams > Teams Apps” rather than “My Apps > Apps”.

Users acting in the team member role who select “My Team” will be presented with a list of teammates for each team for which the user is a member.

Users acting in the team member role will also be presented with a list of team-related apps for each given team and may be allowed to add, update, and/or delete those team apps, depending on configurable permissions managed by the portal administrator.

All apps can be either an “Other Product Type”, or a “Student Services Product Type”.

App Edit/Delete Workflow

An app developer will reach the Edit App page by drilling down through the My Apps page and selecting a specific app to edit.

The Edit App form contains the following fields:

|  |  |
| --- | --- |
| Field Name | Editable |
| App Name | Yes |
| Product Type | No |
| Environment | No |
| Description | Yes |
| Select Products | Yes |