Red Hat 3scale API Management 2.3

Product

Walkthrough of of different product features such as API integration, Analytics, Developer portal, and API environments.

Last Updated: 2019-05-08

Red Hat 3scale API Management 2.3 Product

Walkthrough of of different product features such as API integration, Analytics, Developer portal, and API environments.

Legal Notice

Copyright © 2019 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, JBoss, OpenShift, Fedora, the Infinity logo, and RHCE are trademarks of Red Hat, Inc., registered in the United States and other countries.

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

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

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

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

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

The OpenStack ® 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

This guide documents product features for Red Hat 3scale API Management 2.3.

Table of Contents

CHAPTER 1. API ANALYTICS 3
1.1. PREREQUISITES 3
1.2. DETERMINE THE METRICS AND METHODS YOU WANT TO TRACK 3
1.3. CREATE YOUR METRICS AND METHODS 3
1.4. SET UP REPORTING 4
1.5. CHECK THAT TRAFFIC IS REPORTED CORRECTLY 4
1.6. TROUBLESHOOTING 6
1.7. CONTROLLING WHO SEES THE ANALYTICS 7
1.8. ASYNCHRONOUS AND BATCH TRAFFIC REPORTING 8
1.9. ACCESSING ANALYTICS DATA BY API AND EMAIL REPORTS 8

CHAPTER 2. API VERSIONING 9
2.1. GOAL 9
2.2. PREREQUISITES 9
2.3. URL VERSIONING 9
2.4. ENDPOINT VERSIONING 12
2.5. CUSTOM HEADER VERSIONING 12

CHAPTER 3. GETTING STARTED 13
3.1. PREREQUISITES 13
3.2. CONNECTING ECHO API TO 3SCALE 13
3.2.1. Step 1: Define your API and create your first API key 14
3.2.1.1. 1. Define your API: Add metrics and methods 14
3.2.1.2. 2. Configure any limits you wish to impose on API usage 15
3.2.1.3. 3. Create a new developer account and API credentials 16
3.2.2. Step 2: Integrate via API gateway in the staging environment 18
3.2.3. Step 3: Capture traffic for specific methods 19
3.3. CONGRATULATIONS! 19
3.4. WHAT'S NEXT? 20
3.5. CLOSING THE LOOP 20
3.6. HELP 20

CHAPTER 4. ZERO TO HERO DEVELOPER PORTAL 21
4.1. GOAL 21
4.2. PREREQUISITES 21
4.3. STEP 1: PLAN YOUR PORTAL CONCEPT 21
4.4. STEP 2: SET UP YOUR CMS EDITING ENVIRONMENT 21
4.5. STEP 3: DEFINE PAGE LAYOUT TEMPLATES 22
4.6. STEP 4: CREATE YOUR PAGE HIERARCHY 23
4.7. STEP 5: EDIT YOUR PAGE HEADERS 24
4.8. STEP 6: POPULATE IMAGES AND OTHER ASSETS INTO THE CMS 24
4.9. STEP 7: FULLY CUSTOMIZE YOUR BRANDING WITH CSS 25
4.10. STEP 8: GO LIVE 25

CHAPTER 1. API ANALYTICS

This guide is designed to help you tune your API analytics to track the items you need to know about and to see top applications and trends over time.

Knowing how your API is used is a crucial step for managing traffic, provisioning for peaks, and identifying top users so you can help them achieve greater success with your API.

1.1. PREREQUISITES

Complete the basics of connecting your API to 3scale before using this guide.

The guide assumes that you are using one of the existing 3scale code plugins to perform an integration. You can follow a similar flow with other integration methods. Check the APIcast Overview chapter of the documentation to learn more about the available integration options.

1.2. DETERMINE THE METRICS AND METHODS YOU WANT TO TRACK

3scale acts as an infinitely scalable data repository for your API statistics, and you can track almost any metric for your API. For example:

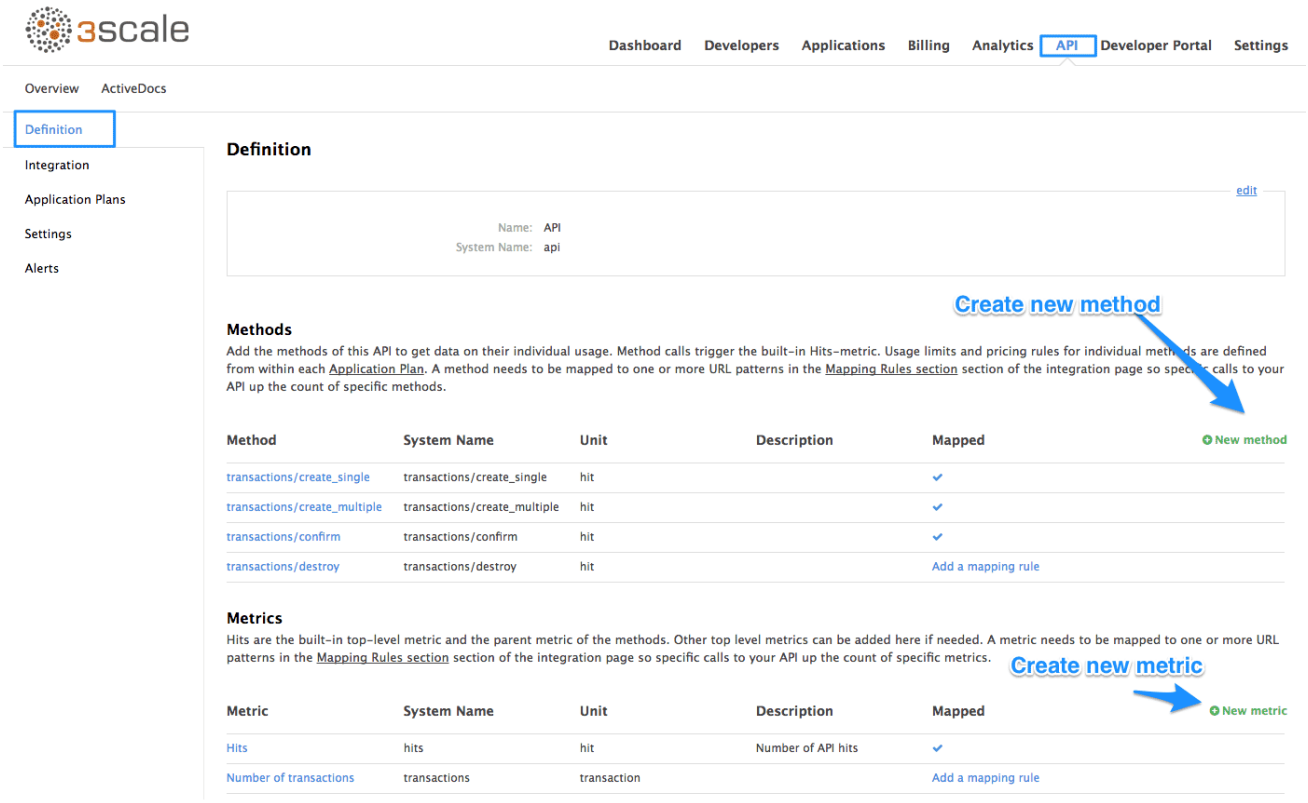
- *Hits/transactions*: calls to the API. Hits are included by default as metrics on all APIs. Hits can be overall calls to the API or broken out into individual methods on the API.
- *Data transfer*: quantity of MB/GB of data uploaded and downloaded via the API
- *CPU hours*: compute time (or some other internal resource) associated with calls to the API
- *Results returned*: count of the number of records or data objects being returned
- *Disk storage*: total disk storage in use by an account

You can track more metrics that are relevant to your API. 3scale can track an arbitrary number of metrics and methods, as long as it is a countable quantity that can be incremented over time.

1.3. CREATE YOUR METRICS AND METHODS

After you chose your metrics, register them in the 3scale Admin Portal. Navigate to the the **Dashboard** > **API** section and select **Definition** for the API you want to manage.

Figure 1.1. Create new method



Add metrics and methods to the service. Provide them with a friendly name and a system name, which is used later in your plugin configuration. For more details about creating methods and metrics, see defining your API on 3scale.

1.4. SET UP REPORTING

Once you have configured the 3scale system with the names of the metrics to track, it is time to tweak your plugin setup to report the right metrics. The precise manner of doing this depends on the plugin or integration method in use. By default, the plugins report the **hits** (API transactions) metric only.

Generally speaking, you need to follow these steps.

1. The application should pass the appropriate metric/method names to the plugin as determined by the incoming API call. The metric/method value and the increment required is an argument of authorize and/or report methods the plugin exposes.

2. You can also report the traffic using the 3scale Service Management API. You can find information about different endpoints in the 3scale APIs ActiveDocs section. 3scale ActiveDocs are available in your Admin Portal under the **Documentation → 3scale API Docs**section.

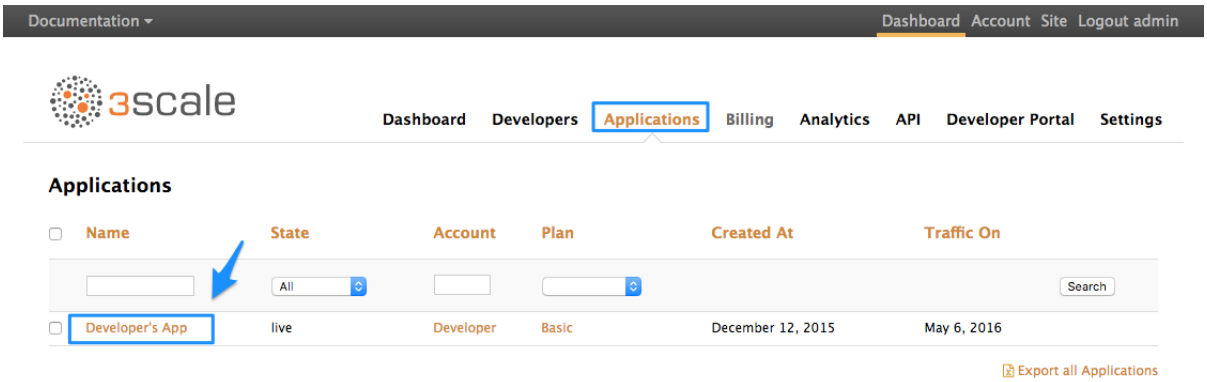
When you report traffic for a specific API method, use the method **system name** in the metric argument. This automatically increments the counter both for the method reported and the hits metric.

1.5. CHECK THAT TRAFFIC IS REPORTED CORRECTLY

Once the API and 3scale connection is established, you can send traffic to the API and watch it register on the API Analytics dashboard. You need an existing developer account and an application with API credentials to be able to perform the steps in this section. Follow these steps to create a developer account and get an application with API credentials.

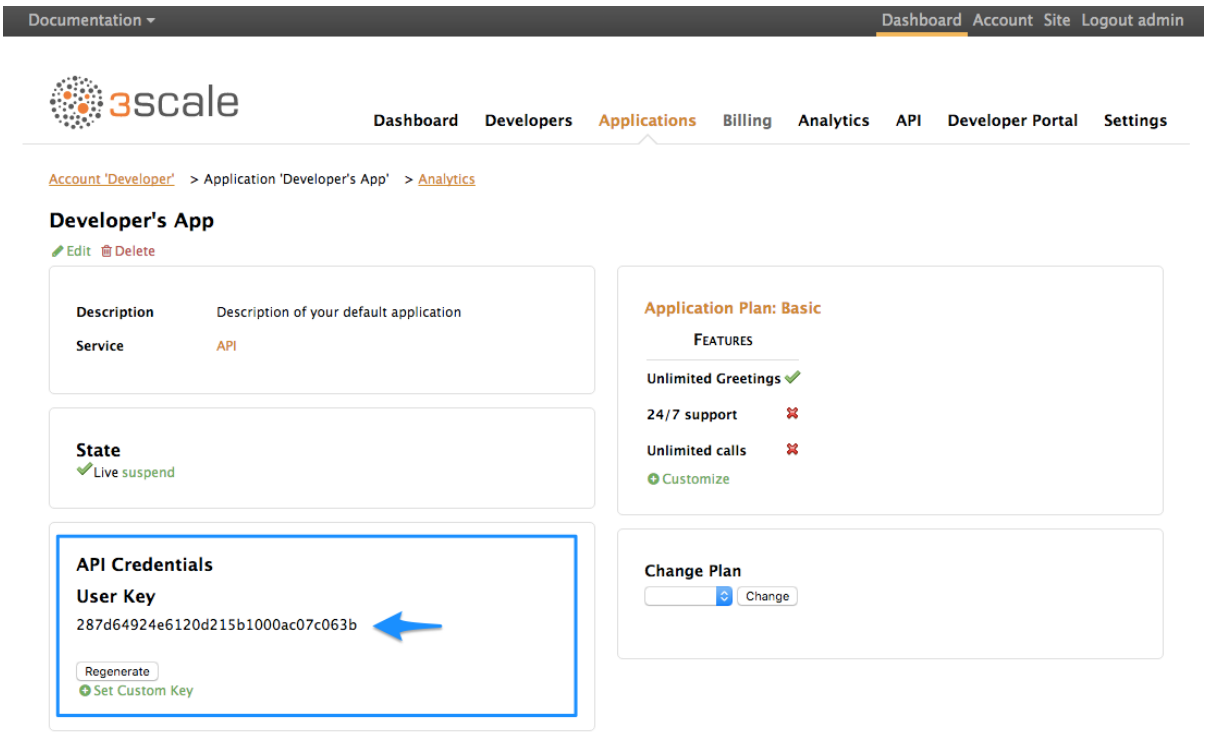
1. Open the Getting Started guide.
2. Navigate to **Dashboard → Applications** in the API Analytics dashboard to see the list of existing applications.
3. Select an application by clicking on its name.

Figure 1.2. Applications



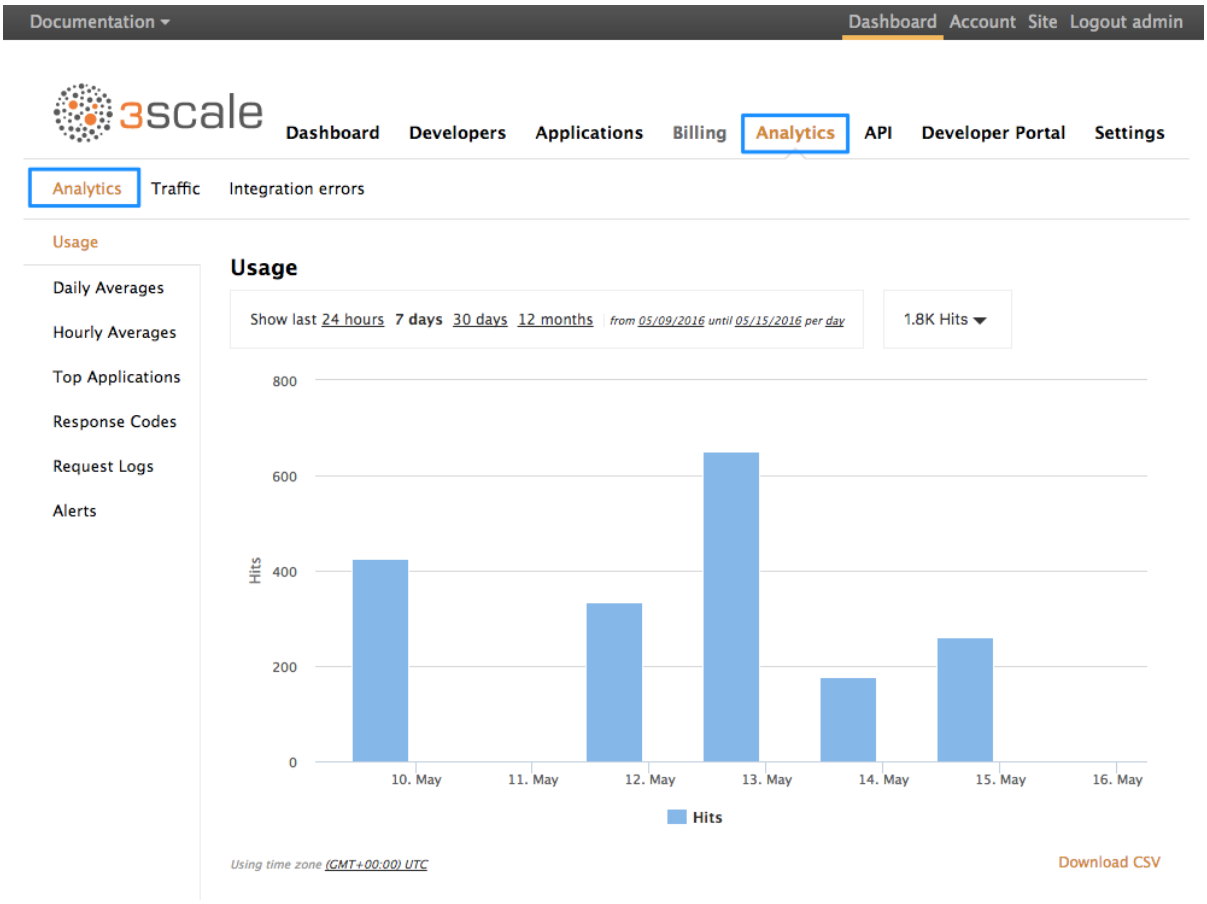
4. Find the API credentials for the selected application. The credentials depend on the selected authentication type and can be a user key (API key), an application ID and application keys, or a client ID and client secret. For more information about the available authentication modes, see the authentication patterns article.

Figure 1.3. API credentials



5. Use these keys to make calls to your API in the normal way (for example, from the command line using cURL or from the browser for API endpoints using the GET method). The precise calls to make depend on the structure of the methods on your API. Traffic from these calls appear in the Analytics section for your API.

Figure 1.4. Analytics usage



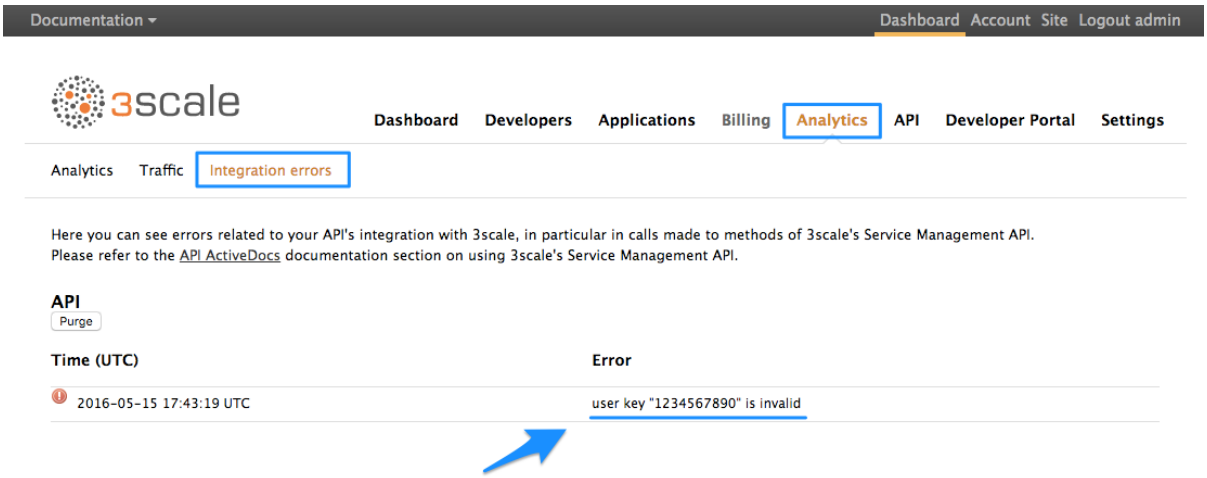
1.6. TROUBLESHOOTING

If traffic does not display on the usage charts in the Analytics section, perform the following checks.

- **Are authorize/report calls responding correctly?**
  All plugins call the 3scale Service Management API, which has predetermined response codes. Authorize calls for valid keys should return responses with HTTP code **200**. Report calls should respond with code **202**.

- **Are there errors in the integration error console?**
  The log of integration errors detected by 3scale can be found in **Analytics → Integration errors**.

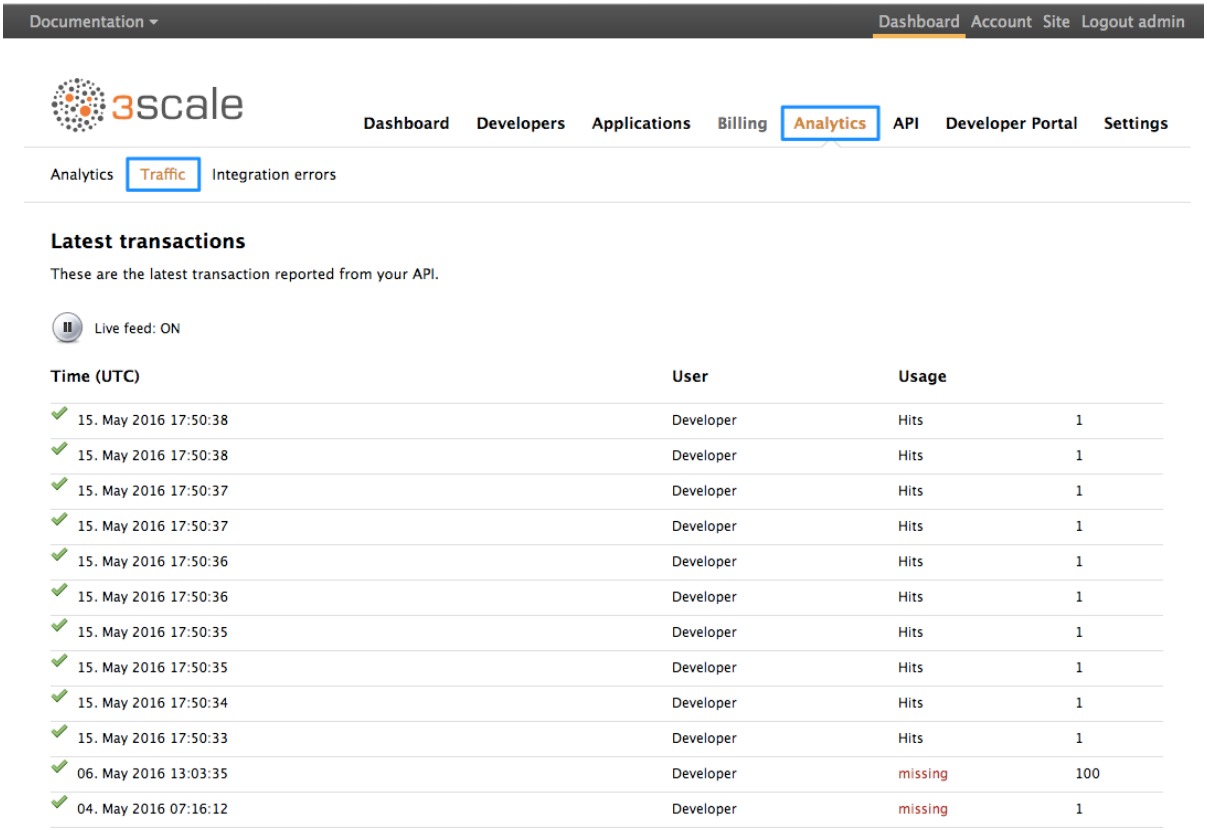
Figure 1.5. Integration errors



- **Are the correct metric and method names being used?**
  The most common reason for failure is that the method and metric names passed in report calls do not correspond to those created in the API settings of your Admin Portal. Check that you are using the correct **system names** for each metric/method.

  You can also check which metrics are being reported to 3scale in the **Analytics → Traffic** section.

Figure 1.6. Analytics traffic



1.7. CONTROLLING WHO SEES THE ANALYTICS

By default, the usage statistics are visible to the API provider through the Admin Portal and to the developers who created applications through the Developer Portal. (Each developer can see only the usage statistics for their own applications.) You have the ability to hide the analytics views from the

Developer Portal. See the Developer Portal section to learn more about customizing the Developer Portal.

1.8. ASYNCHRONOUS AND BATCH TRAFFIC REPORTING

The API usage is usually reported to 3scale after each call; however, the following approaches can be also used.

- **Asynchronous traffic reports**
  If you use plugin integration, you can make report calls asynchronously and avoid additional latency in your API. Deployment options provided by 3scale (hosted, self-managed) use this approach by default. See APIcast for more details.

- **Batch reports**
  You can also report in batches rather than call-by-call by bundling groups of calls together and sending the reports for them on a minute-by-minute basis or by some other criteria.

1.9. ACCESSING ANALYTICS DATA BY API AND EMAIL REPORTS

Besides the usage graphs in the Analytics section, there are other methods of getting your API's analytics data.

- **Analytics API**
  You can use the 3scale Analytics API. It is a flexible way to extract all the analytics data for your API in either XML or JSON format.

- **Daily and weekly traffic reports (SaaS only)**
  These reports provide the aggregated data about your traffic, including information about new subscribers to your API and top applications. To enable these reports in the **Account > Notifications** section of your Admin Portal, find the **weekly aggregate reports** and **daily aggregate reports** check boxes. If enabled, these reports are emailed to the admin user of your 3scale account.

- **CSV export (SaaS only)**
  There is a **download CSV** link on each analytics view page, and you can download the usage statistics in .csv format.

  Download CSV image::guides-api-analytics-download-csv.png[width=100px]

CHAPTER 2. API VERSIONING

The 3scale API Management Platform allows API versioning. You have three ways to version your API correctly when you manage your API with 3scale. The following methods are examples of how you could version your API within the 3scale Gateway, which provides extra features due to 3scale's architecture.

2.1. GOAL

This guide is designed to give you enough information to implement an API versioning system within 3scale.

Suppose you have an API for finding songs. Users can search for their favorite songs by different keywords: artist, songwriter, song title, album title, and so on. Assume you had an initial version (v1) of the API and now you have developed a new, improved version (v2).

The following sections describe the three most typical ways of implementing an APR versioning system using 3scale:

- URL versioning
- Endpoint versioning
- Custom header versioning

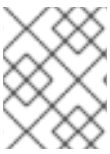
2.2. PREREQUISITES

Complete the basics of connecting your API to 3scale before using this quick start guide.

2.3. URL VERSIONING

If you have different endpoints for searching songs (by artist, by song title, and so on), with URL versioning you would include the API version as part of the URI, for example:

1. api.songs.com/v1/songwriter
2. api.songs.com/v2/songwriter
3. api.songs.com/v1/song
4. api.songs.com/v2/song
5. and so on



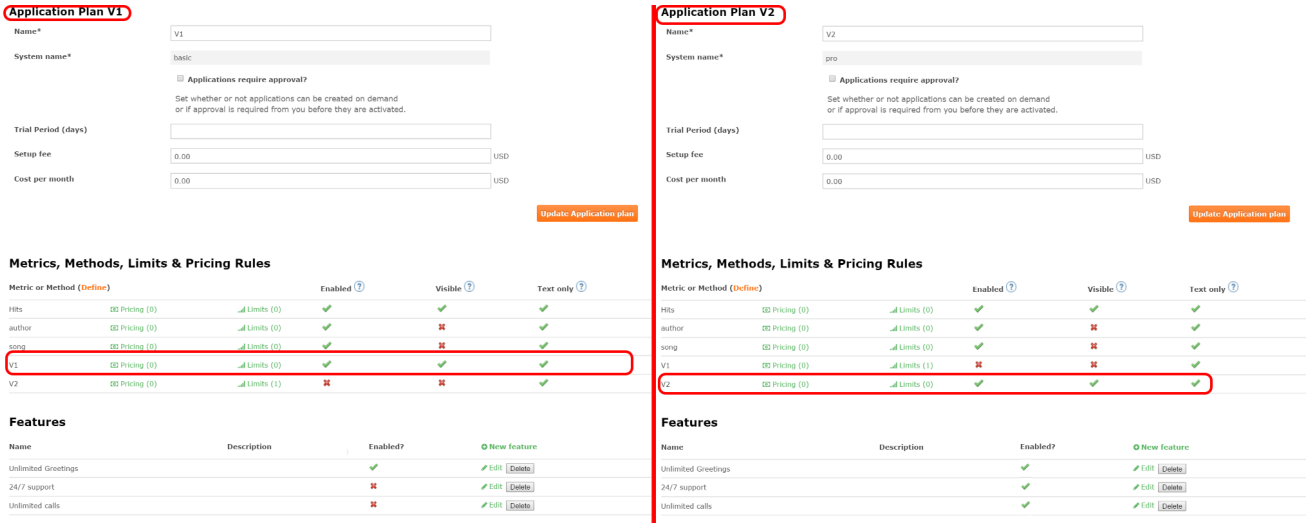
NOTE

When you use this method, you should have planned since v1 that you were going to version your API.

The 3scale Gateway would then extract the endpoint and the version from the URI. This approach allows you to set up application plans for any version/endpoint combination. You can then associate metrics with those plans and endpoints, and you can chart the usage for each endpoint on each version.

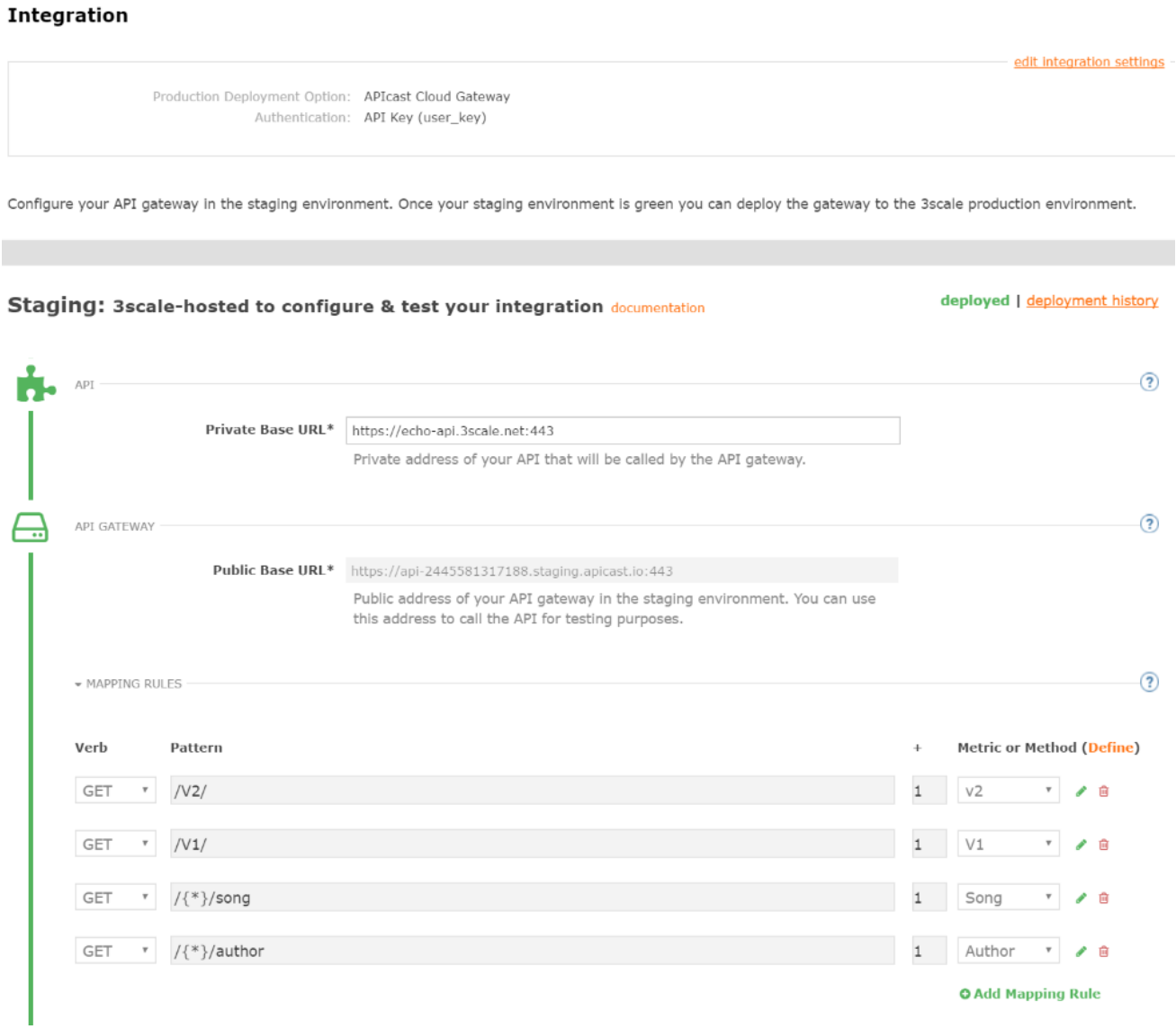
The following screen capture shows 3scale's flexibility.

Figure 2.1. Versioning Plan Feature



The only thing left to do is go to **Dashboard → Integration** on your 3scale Admin Portal and map your URIs to your metrics, as shown in the following diagram.

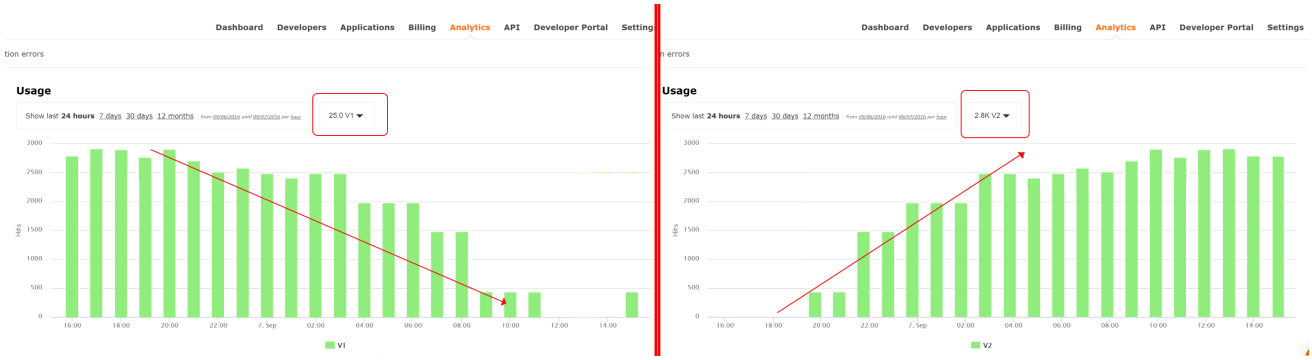
Figure 2.2. Mapping URIs to metrics



You now have two different versions of your API, each with different features enabled. You also have full control and visibility on their usage.

If you want to communicate to all of your users that they should move to the API v2, you can send an internal note asking them to do so. You can monitor who makes the move and see how the activity on v1 decreases while the activity on v2 increases. By adding the metric in your authorization calls to 3scale, you can see how much overall traffic is hitting v1 vs. v2 endpoints and get an idea of when it is safe to deprecate v1.

Figure 2.3. Versioning



If some users continue to use v1, you can filter out only those users to send another internal note about switching to v2.

3scale provides a three-step method for sending deprecation notices.

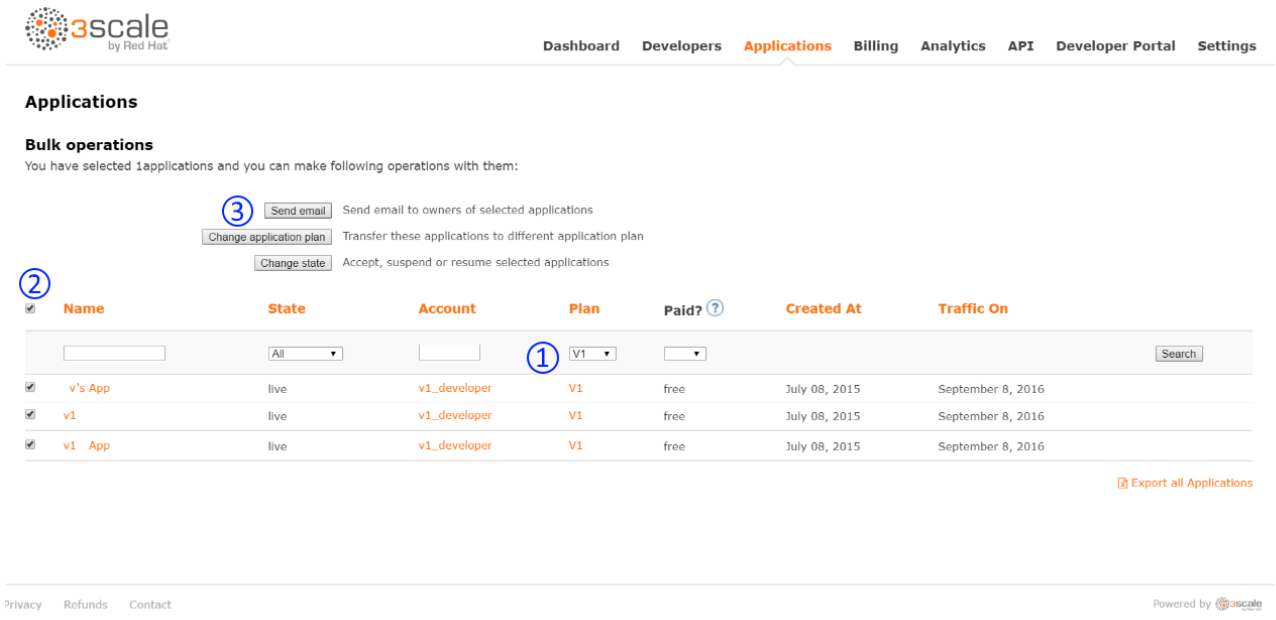
1. Navigate to the **Applications** tab and filter the list by the application plan that you want to send the deprecation note and click **Search**.

2. Click the multiselector to select all of the users for that particular version. New options display and allow you to perform bulk operations, such as **Send email**, **Change Application Plan**, and **Change State**.

3. Click **Send email** and follow the steps to send a deprecation notice to those customers who are still under the obsolete version.

The following image provides a visual reference.

Figure 2.4. Sending deprecation note



For each authrep call that is made to an endpoint, you authenticate only once but report twice: once for the endpoint and once for the API version. There is no double-billing because the call can be authenticated only one time. For each call you make to any endpoint of a specific API version, you aggregate the hits on a convenient metric named after the version number (v1, v2, and so on), which you can use to compare full version traffic with each other.

2.4. ENDPOINT VERSIONING

You have the endpoint change for each version (api.cons.com/author_v1) with endpoint versioning. The gateway extracts the endpoint and the version from the endpoint itself. This method , as well as the previous method, allows the API provider to map external URLs to internal ones.

The endpoint versioning method can only be performed with the on-premise deployment method as it requires a URL rewrite using the LUA scripts that are provided as part of the on-premise configuration.

EXTERNAL		INTERNAL
api.songs.com/songwriter_v1	could be rewritten to	internal.songs.com/search_by_songwriter
api.songs.com/songwriter_v2	could be rewritten to	internal.songs.com/songwriter

Almost everything (mapping, application plans features, and so on.) works exactly the same as in the previous method.

2.5. CUSTOM HEADER VERSIONING

With custom header versioning, you use a header (that is, "x-api-version") instead of the URI to specify the version.

The gateway then extracts the endpoint from the path and the version from the header. Just as before, you can analyze and visualize any combination of path/version that you want. This approach has several inconveniences, regardless of the API management system you use. See API versioning methods, a brief reference for more information. Here are a few pointers on how 3scale works.

- Just like the previous method, custom header versioning can only be applied to on-premise hosted APIs because it requires some parsing/processing of the request headers to correctly route the authrep calls. This type of custom processing can only be done using Lua scripting.
- With this method, the fine-grained feature separation of the previous methods is much harder to achieve.
- One of the biggest advantages of using this methodology, and the main reason some API providers choose it, is because the URL and endpoints of your customers will never change. When a developer wants to switch from one API version to another, they only have to change the header. Everything else works the same.

CHAPTER 3. GETTING STARTED

By the end of this guide, your API traffic will be protected by API keys, tracked, and monitored by 3scale with basic rate limits and controls in place. A fictional "Echo API" serves as an example, which you can substitute with your own API.

Getting your API up and running with 3scale is straightforward and easy to accomplish by following the steps here. You'll get traffic flowing and monitored as well as be able to issue rate-limited developer keys.

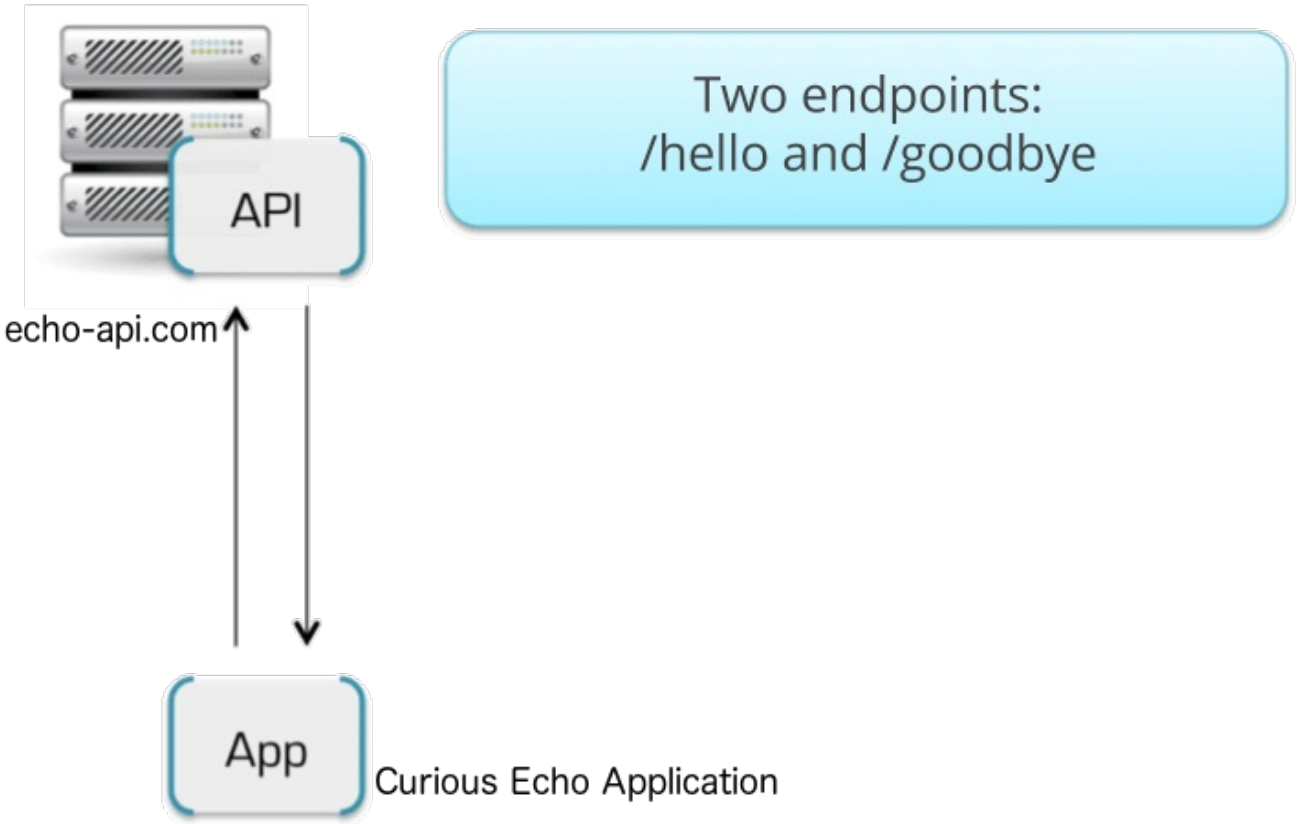
For more background on architecture and a general overview, head to the Technical Overview page.

Remember that if you have a production API, you should do this in a staging/non-production environment initially to avoid disruption for existing API users.

3.1. PREREQUISITES

To run this example you can use a simple test API called "Echo API" hosted at https://echo-api.3scale.net.

You'd need to have a simple application, for example "Curious echo," which will call the API. This may be as simple as a command line call, a mobile app, or any code that can call a remote server.



3.2. CONNECTING ECHO API TO 3SCALE

In order to connect Echo API to 3scale, you need to follow three simple steps:

1. Access your 3scale Admin Portal and set up your first plans and metrics and your first API keys.
2. Integrate your API with 3scale using the API gateway in the staging environment (for development only).

3. Map your API endpoints to 3scale methods and metrics.

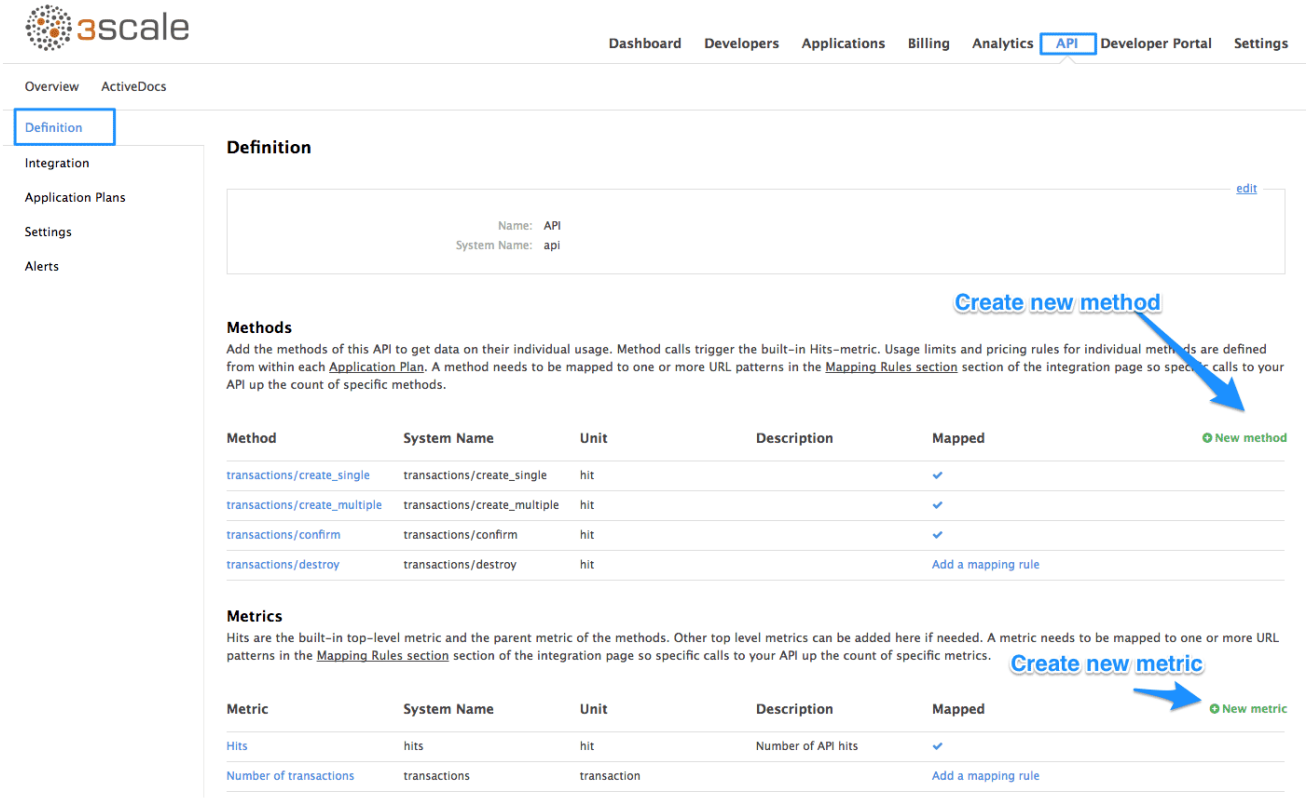
3.2.1. Step 1: Define your API and create your first API key

Your 3scale Admin Portal (http://YOURDOMAIN-admin.3scale.net) provides access to a number of configuration features. For now, focus on getting the minimum setup required to deploy your API:

1. Define your API: Add the metrics and methods.
2. Configure any limits you may wish to impose on API usage.
3. Head to the **Developers** area to create a new developer account and API credentials.

3.2.1.1. 1. Define your API: Add metrics and methods

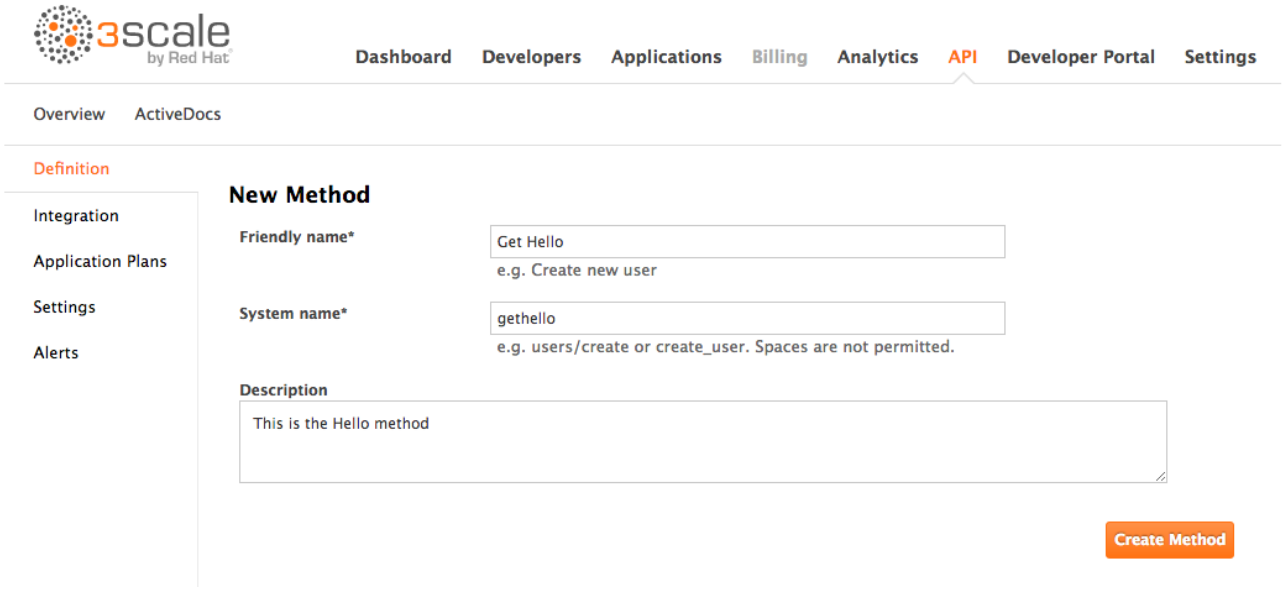
Here you can add as many methods and metrics as you need. By default, they'll be available in all plans of your service.



For more details about how to add methods and metrics, you can check out our documentation page about /docs/access-control/api-definition-methods-metrics[defining your API on 3scale].

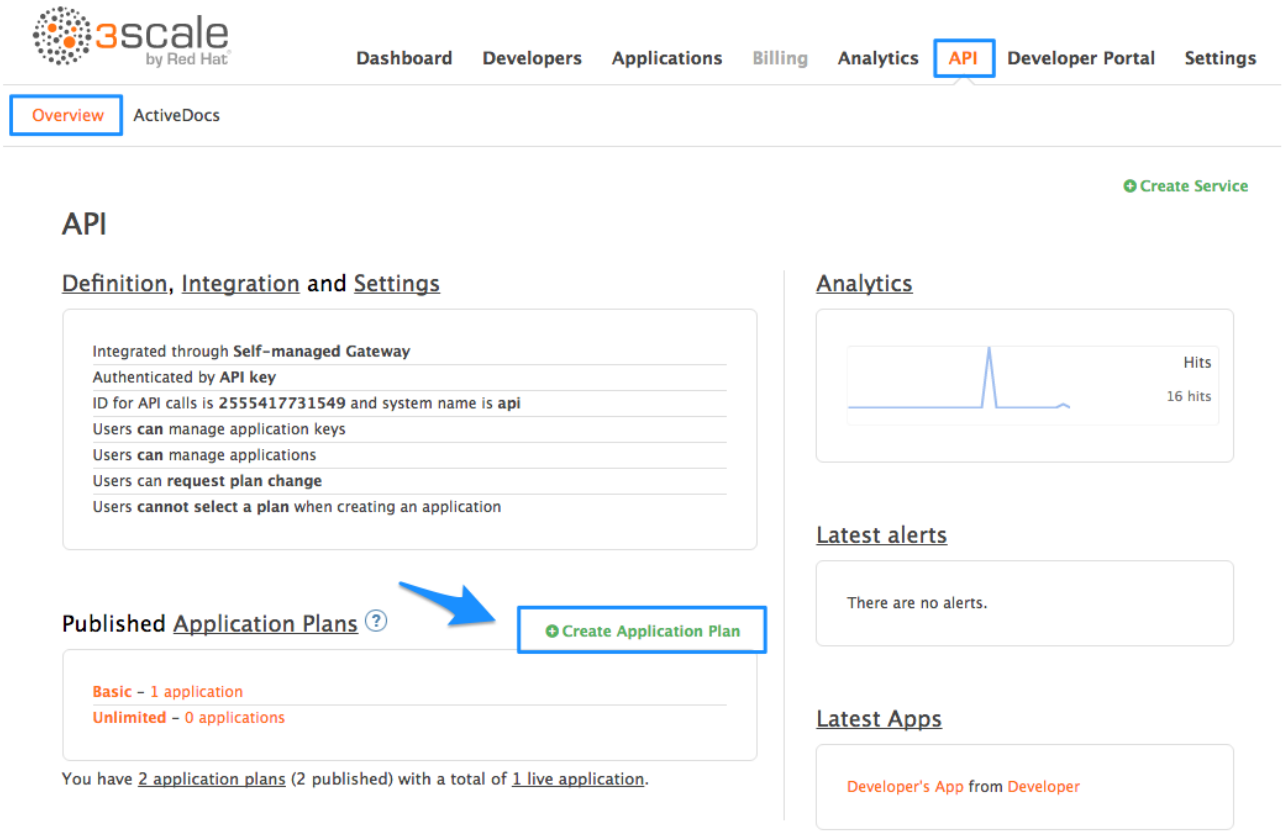
For this simple test, add just two methods under "hits" with system names:

- **gethello**
- **getgoodbye**

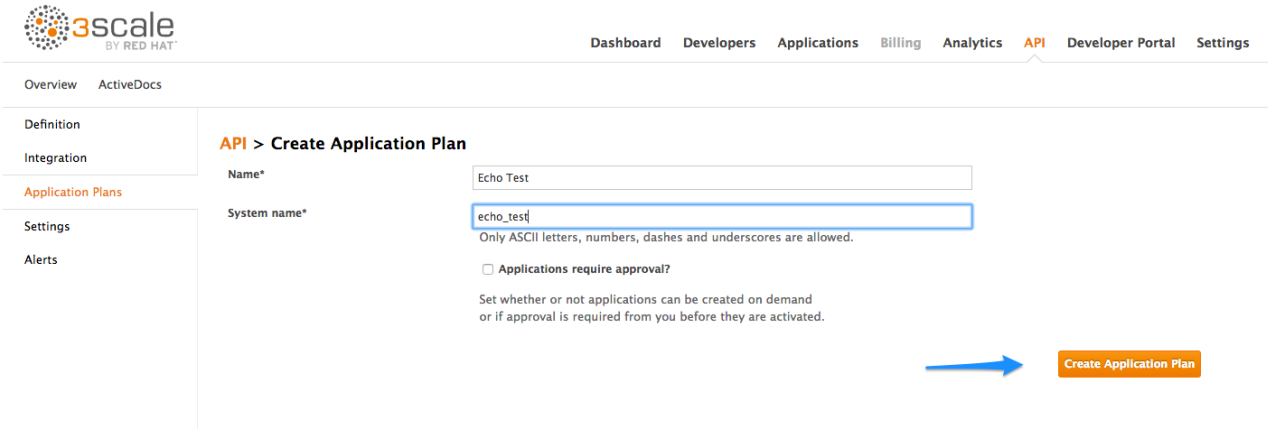


3.2.1.2. 2. Configure any limits you wish to impose on API usage

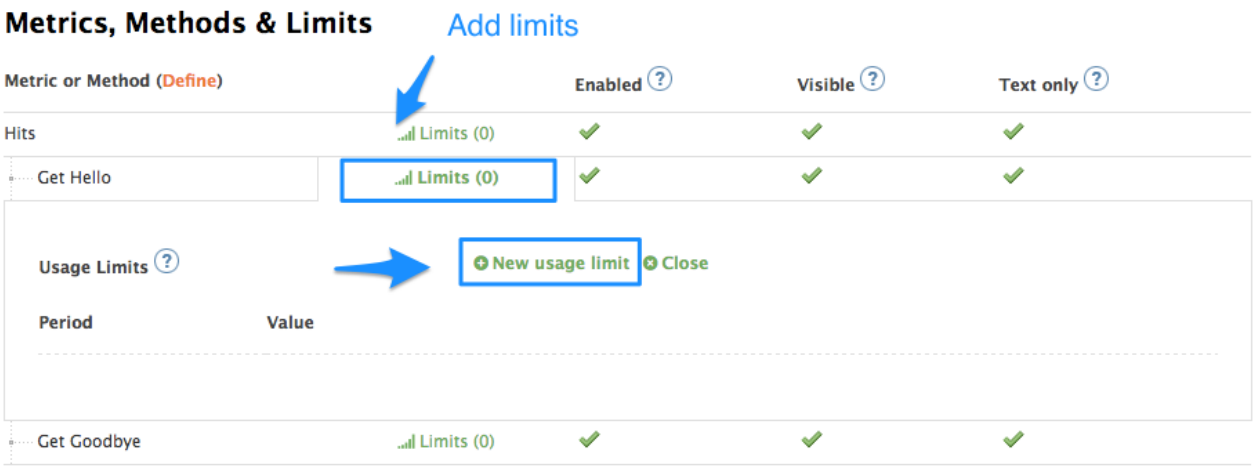
In addition to creating the metrics/methods, you can also add limits to any of the API usage metrics under each plan. Let's create a new application plan for this example. In order to do this, navigate to the API tab and click on **Create Application Plan**.



In the form that opens, specify the desired name – for example "HelloEchoTest" – and the system name. Then click on **Create Application Plan** button.



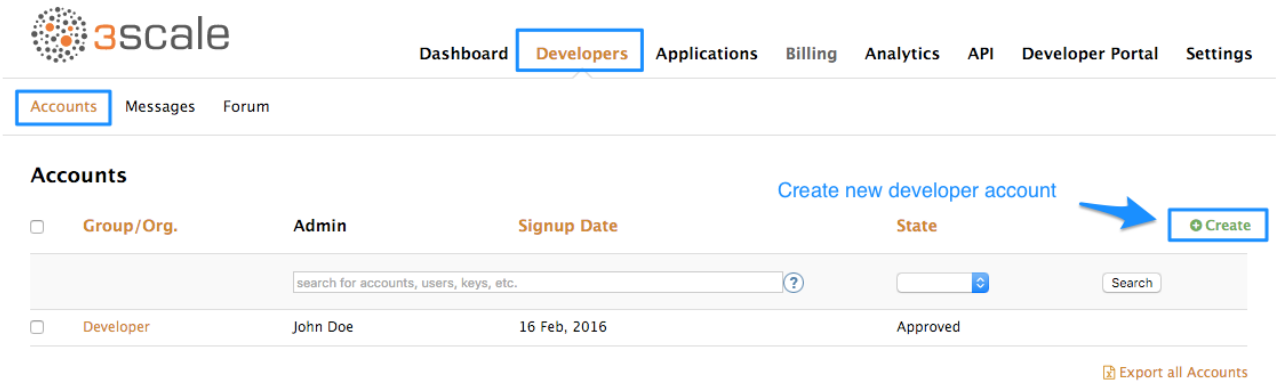
After the previous step, you should see the list of application plans. Click on the "HelloEchoTest" plan to create limits for the metrics and methods. You should be able to see all the metrics and methods that you defined in the previous step. Click on the "Limits" icon under any metric or method. Adding a limit to the Hits metric applies the rule across all the methods under Hits; adding limits to a method only applies to that method. You can create different plans with different limits later on.



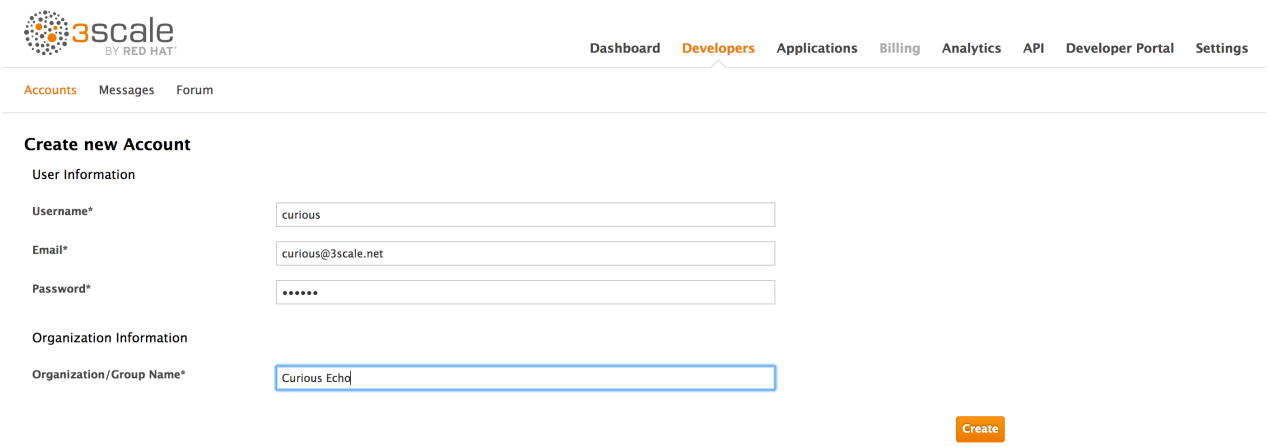
Limits restrict the number of API calls an application on this plan can do per minute/hour/day/etc.

3.2.1.3. 3. Create a new developer account and API credentials

Select the **Developers > Accounts** menu item and click on the **create** button.



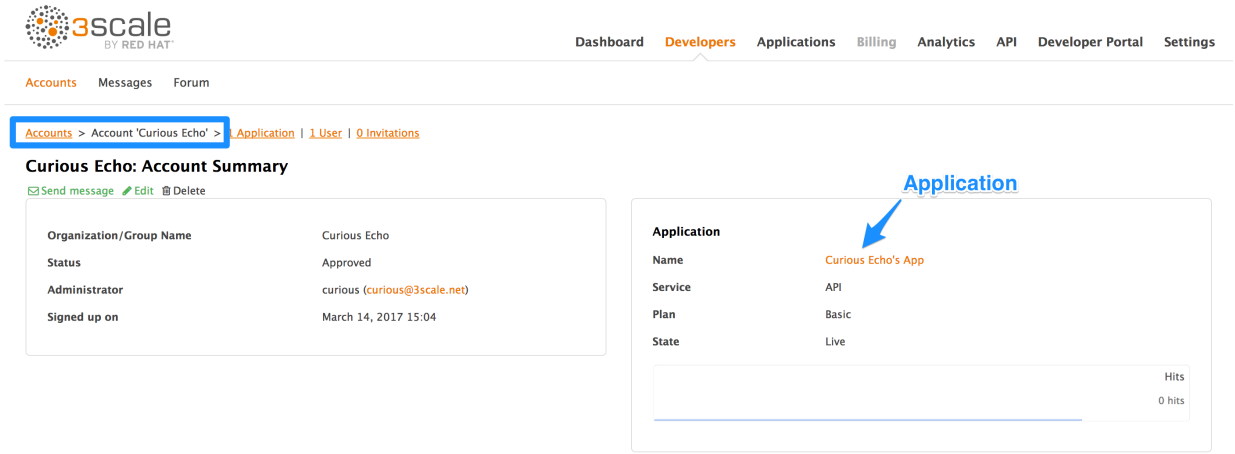
Fill in some information for the new developer who will access the API.



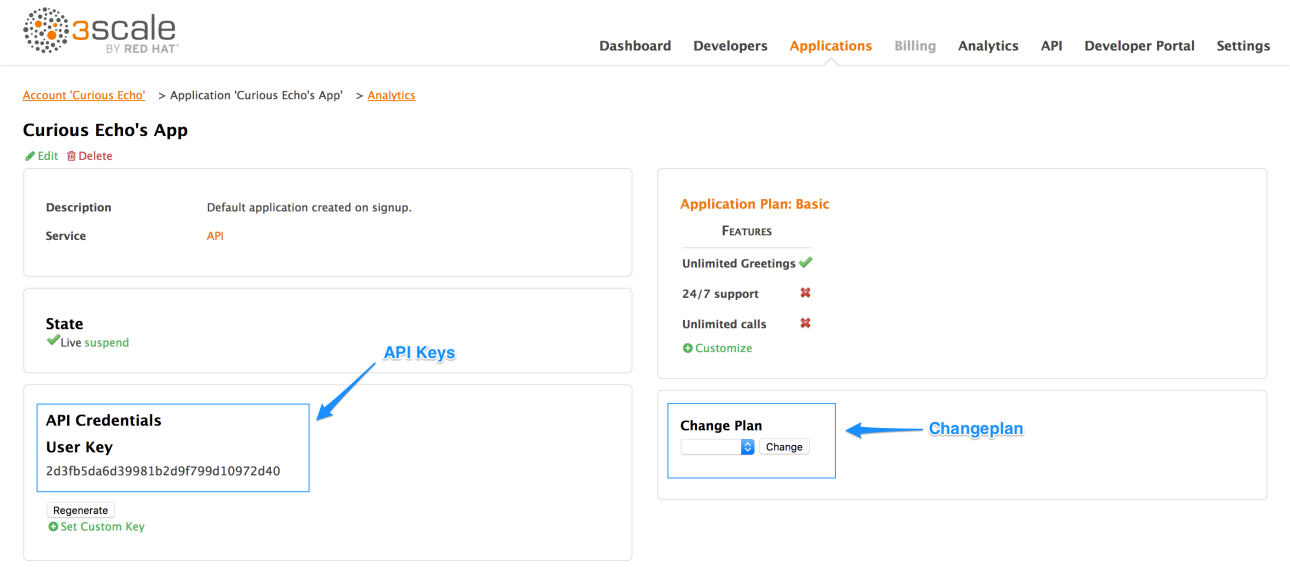
Once you click **create**, select the new account from the list to go to the home page.

The account area lists all the companies and developers signed up to use the API. New companies can be added from the dashboard, from the API, or by self-service signup on the developer portal.

When you create a new developer account, you will also be creating a new application for that account.



Applications will each have a unique key to access the API. To find that key, click on the application name and check the API credentials section.



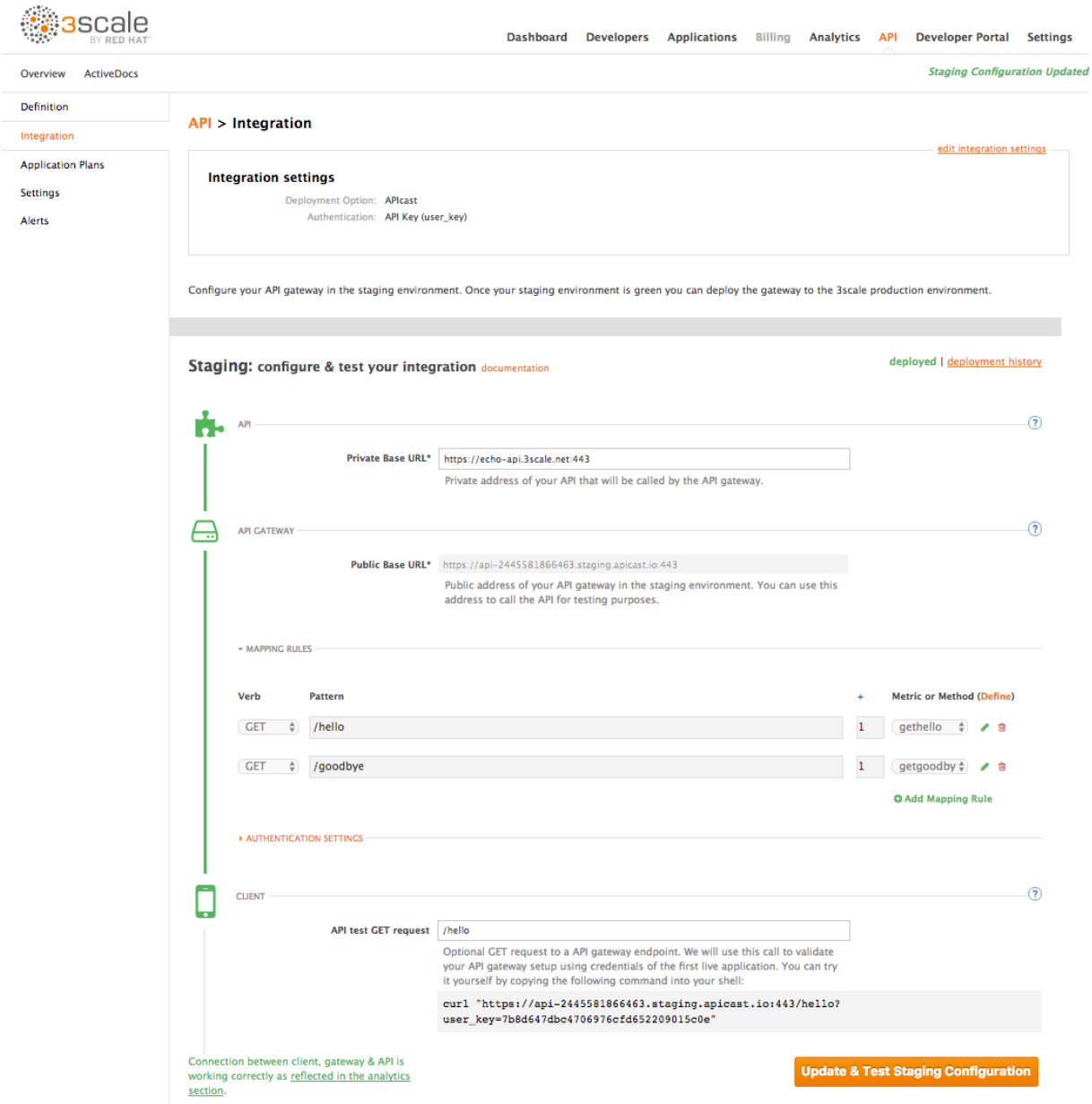
These are the keys the "Curious Echo" app will use to call the Echo API. Lastly, on the right-hand side of the application details page (see screenshot above), select the **change plan** dropdown and select the plan you created and named earlier ("Echo Test" in the example) and confirm the change. This applies the new plan to this application.

You have now configured the management system for your first application.

For the next steps, make sure you're using the APIcast hosted deployment option. This option has a **staging** area where you can easily try out your configuration.

3.2.2. Step 2: Integrate via API gateway in the staging environment

Once you sign into your 3scale account, go to **API > Integration**.



Set the address of your API backend in the staging environment. This is the address of the server where your API is running. Now you can input a valid resource path for your API, which will be used to validate the API gateway in the staging environment. After that, hit **update & test staging configuration**. If everything goes well, you will see a green vertical line in the staging area and the full test call made to verify connection. It will look like this:

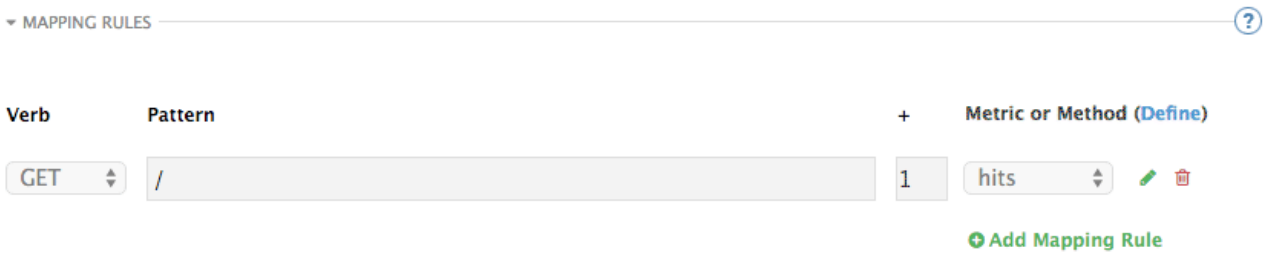
```
curl "https://api-xxx.staging.apicast.io/hello?user_key=USER_KEY"
```

USER_KEY is the key of one of the sample applications that were created when you first logged into your 3scale account. (If you missed that step, just create a developer account and an application within that account.)

Try it without app credentials, then with incorrect credentials. Then once authenticated, try to send API calls within and over any rate limits that you've defined.

3.2.3. Step 3: Capture traffic for specific methods

By default you start with a very simple mapping rule.



This rule says that any **GET** request that starts with **"/"** will increment the metric **hits** by 1. You will most likely remove this rule since it's too generic. You can learn more about how to manage Mapping rules on this documentation page .

The mapping rules define which metrics (and methods) you want to report depending on the requests to your API. For instance, below you can see the rules for the Echo API.

You're matching the API endpoints with the methods, which you defined earlier in application plans.

- **/hello**
- **/goodbye**

Now you can repeat traffic testing for the mapped methods and check their traffic in the **Analytics** section of your Admin Portal.

3.3. CONGRATULATIONS!

Your API is now connected to 3scale. You can now apply API management features to manage and track your API traffic!

3.4. WHAT'S NEXT?

Now that you've tested your integration with 3scale in a staging environment, you can select a production deployment option. You can either continue with the NGINX gateway solution or try the plugin integration. For NGINX gateway integration, check out the following documentation:

- APIcast overview
- Advanced APIcast configuration

If you prefer to integrate with 3scale though one of the available code plugins, you can find more information about how to set them up and what programming languages are supported on the following documentation pages:

- Plugin Setup
- Code Libraries

3.5. CLOSING THE LOOP

In the example, new API credentials were generated from the Admin Portal to keep things simple. Once you've set up a developer portal, new developers can use the it to automatically create accounts and receive their credentials.

3.6. HELP

If you have trouble setting up your API, head over to the troubleshooting tutorial.

CHAPTER 4. ZERO TO HERO DEVELOPER PORTAL

Most best practice reviews of API deployments agree that a well structured developer portal and great documentation are key elements to assure adoption. Your developer portal is not only a source of documentation. It is also your main hub to manage interactions with developers and for developers to access to their API keys in a secure way.

4.1. GOAL

By the end of this tutorial, you'll have a developer portal up and running to promote your API, allow developers sign up for accounts, and access their API keys.

4.2. PREREQUISITES

There are a few other areas for you to set up that are interdependent with the portal. You can take care of them before or after:

- Set up your application plans – so you have access rights configured for the API keys that you'll issue.
- Configure your developer signup flows –, which can be anywhere on the spectrum from self-service, to approval required, to invite only (with signups disabled).
- Setu p a custom domain for your portal (optional) – – This can have a lead time of 1-2 weeks and is normally done at the same time you create a custom outbound email address.

4.3. STEP 1: PLAN YOUR PORTAL CONCEPT

Before you even open the 3scale CMS, it's a good idea to plan out the concept for your portal. The better organized you are, the more efficiently you'll be able to complete these steps.

The most important elements to plan out are your:

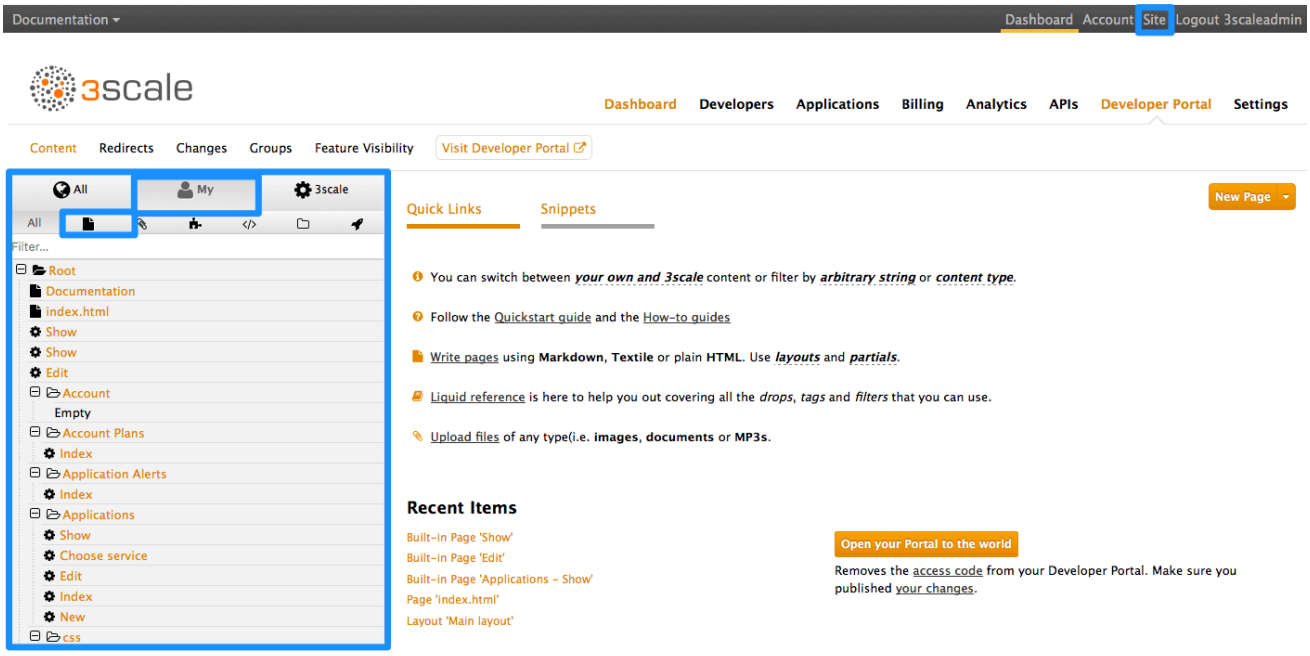
- Site map –– a skeleton of what the portal structure will be
- Top menu bar –– the navigation that will be repeated on every page
- Side menu bars –– for access to individual pages within each section
- Page layout guidelines –– to give your portal a consistent look and feel

4.4. STEP 2: SET UP YOUR CMS EDITING ENVIRONMENT

The best setup of your editing environment has proven to be:

- A tab showing *yourcompany-admin.3scale.net/p/admin/cms* logged in with your admin credentials, which gives access to CMS for the portal
- Second tab pointing to *yourcompany.3scale.net*, the public view of your portal (if you access this though the Site link, you don't have to worry about the access code)

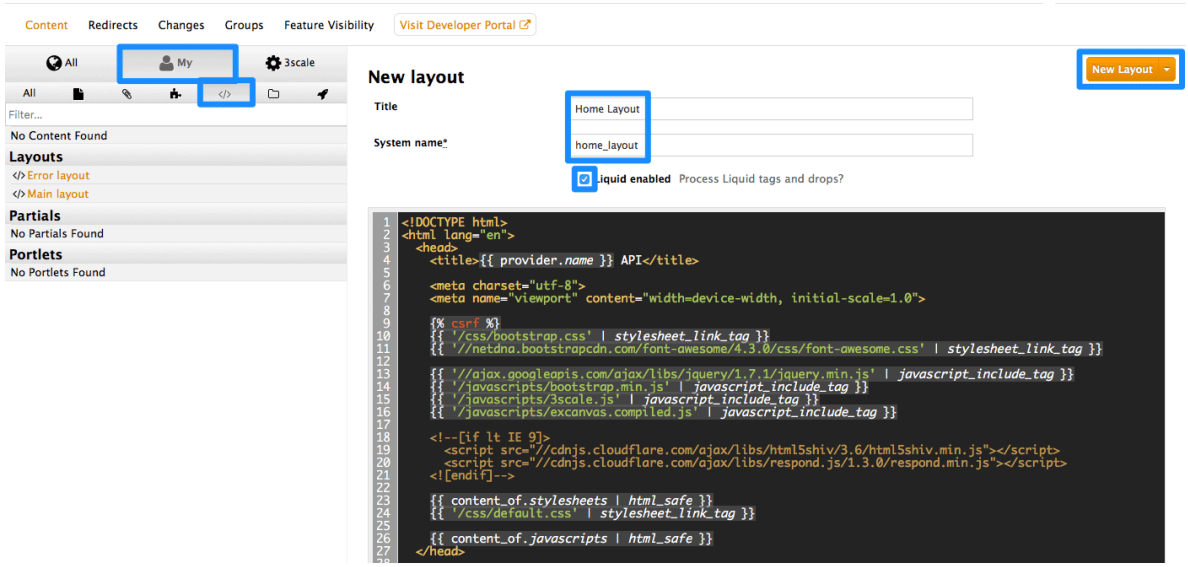
In the Admin Panel, you can see all your developer portal's elements in the left sidebar:



4.5. STEP 3: DEFINE PAGE LAYOUT TEMPLATES

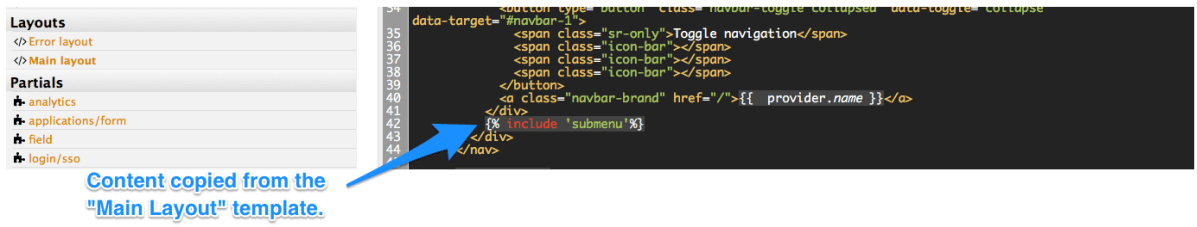
The general idea is to define a separate layout for each of the different page styles in your portal. There is one standard layout called "main layout" when you start. It's best not to make any changes to this until you are quite expert at using the CMS because this is used by all the system-generated pages. Usually you'll want to have a unique style for the home page of your portal.

1. The main layout template will be a starting point for your customizations. Create a new layout, and copy/paste the content of main layout into it.



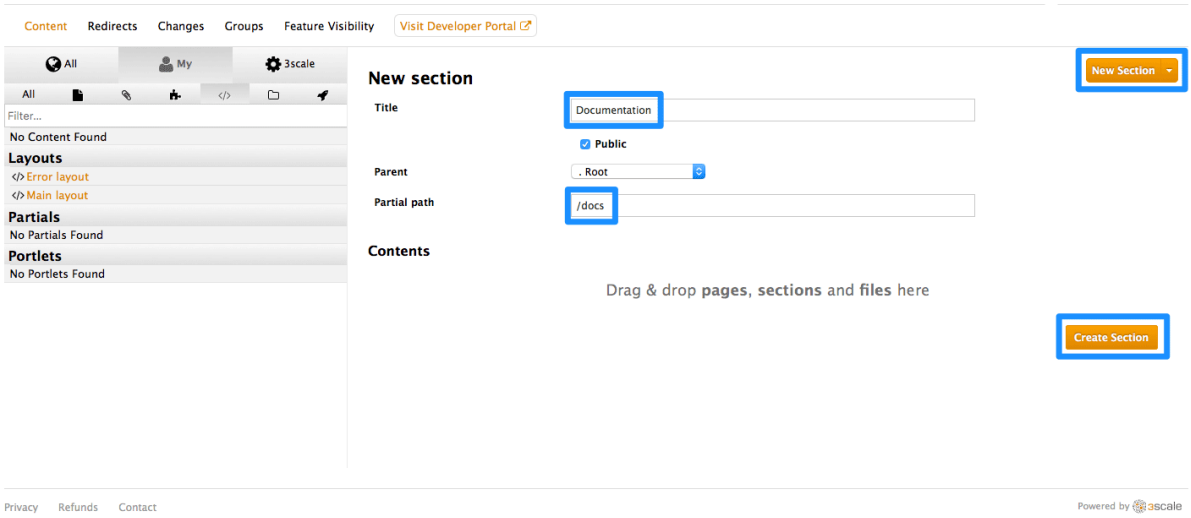
2. Remove the sidebar menu by deleting this line from your "home layout":

> {% include 'submenu'%}

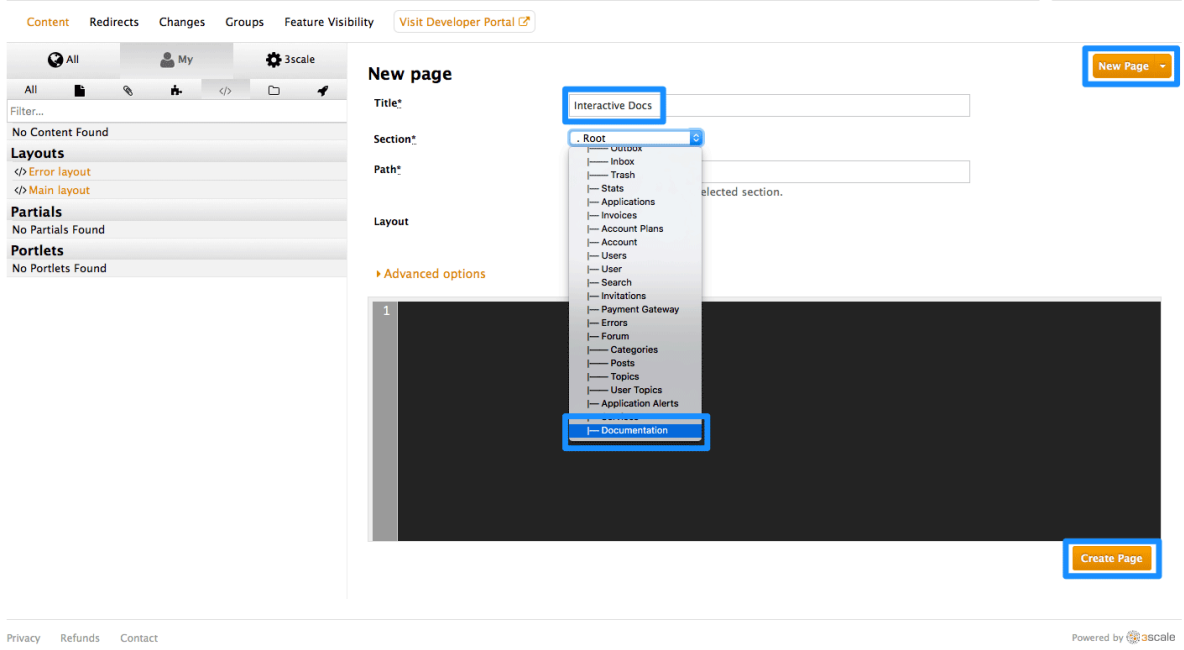


4.6. STEP 4: CREATE YOUR PAGE HIERARCHY

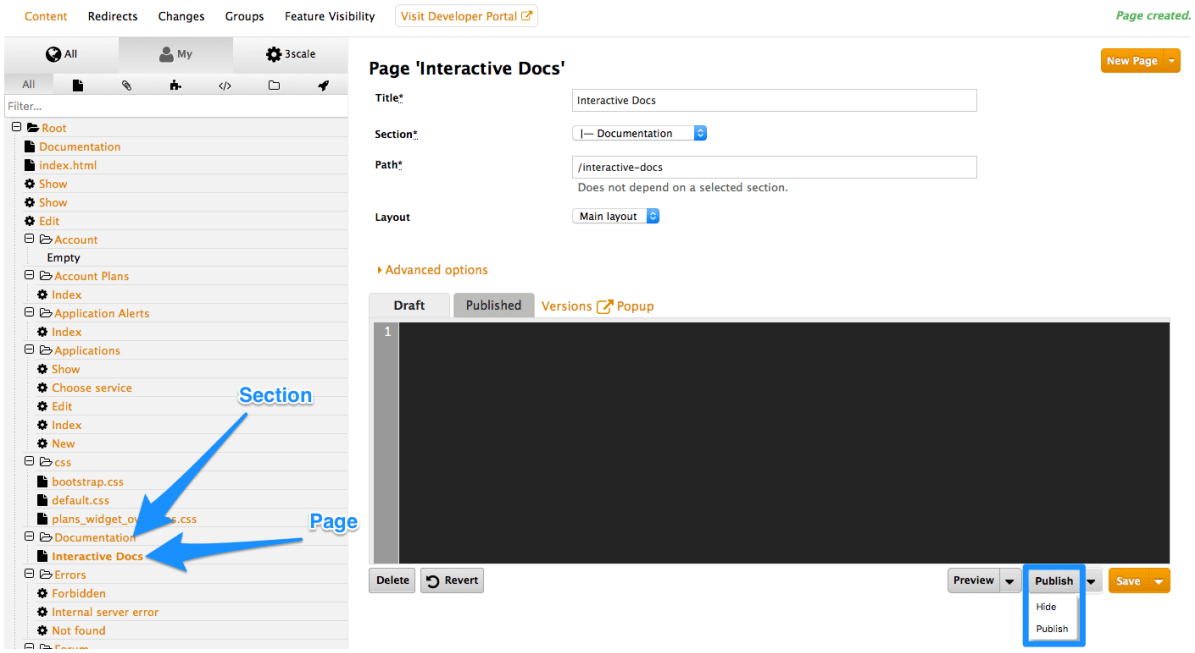
1. Begin at the root level in the site map, and add a new section for each of your top menu items (add sections by expanding the “new” button on the righthand side). Assign a title, parent section, and the path.



2. Similar to adding a section, add a page. Choose the desired section in order to structure your URL paths consistently. Next, select the layout that the page will be using. After completing the page content, hit “create page”. If you are writing a lot of content, you may prefer to use a markup language like Textile or Markdown, which you can select in the advanced page settings.



3. View the draft preview and refine the page content until you are happy and ready to publish it.

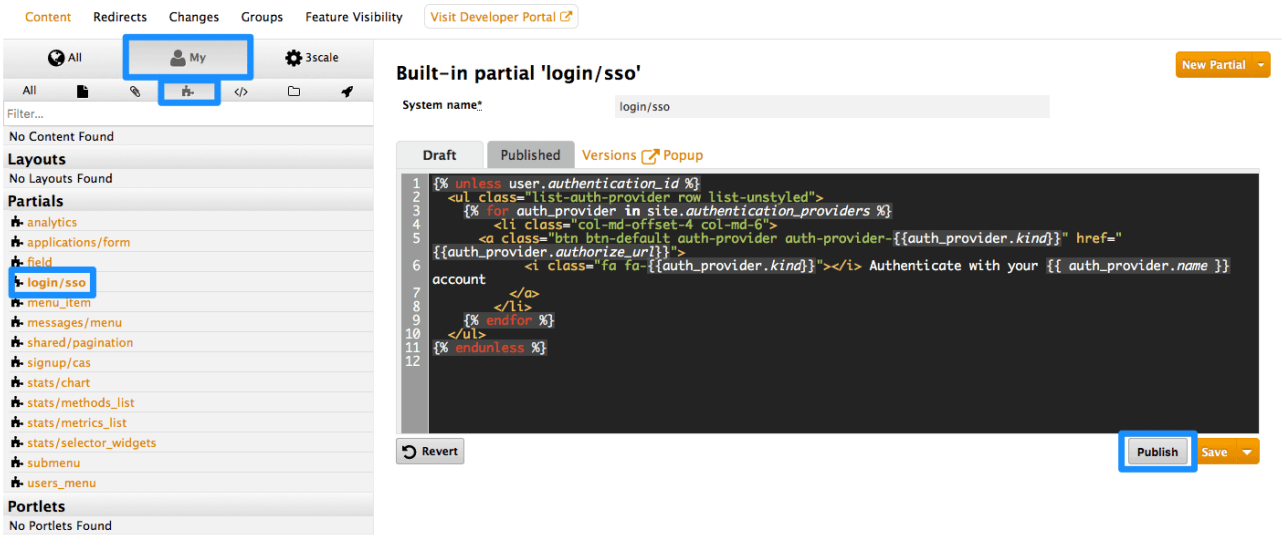


4. Repeat for all pages in the section.

5. Then repeat for all sections in the site.

4.7. STEP 5: EDIT YOUR PAGE HEADERS

All repetitive page elements such as headers and footers are defined in the portal CMS section called “partials”. If you have only one layout, or very few of them, you can omit this step and include the header and footer inside the layouts code. However, remember to customize these elements in the layout. For example, the default “menu” partial should be edited to reflect your site map.



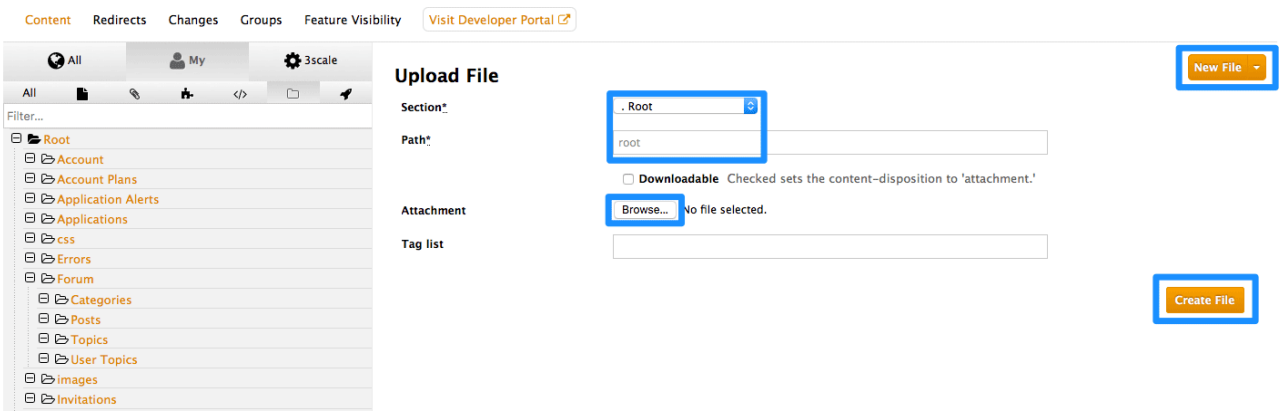
4.8. STEP 6: POPULATE IMAGES AND OTHER ASSETS INTO THE CMS

For images or other files, first load the files into the content library, then insert a link into your text content.

1. Select New File in order to choose your file, and identify where you will save it on your site.

2. Copy the URL to the image

3. Now you can add your HTML or <a> tag, and paste in the URL for your image.



4.9. STEP 7: FULLY CUSTOMIZE YOUR BRANDING WITH CSS

There is a default stylesheet called default.css, which is quite large and complex. Rather than extend this, it's better to create your own stylesheet with your own customizations to overwrite the defaults.

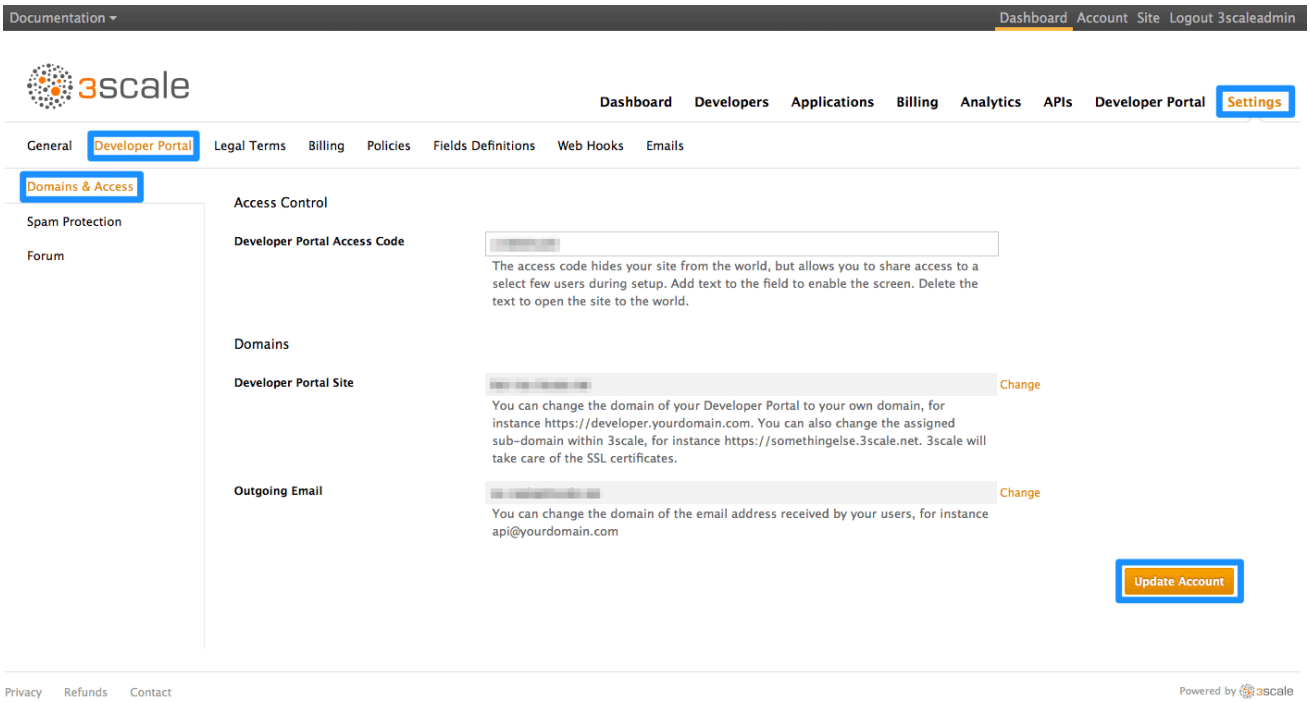
You can create a new stylesheet the same way you create a page – just remember to choose an appropriate MIME content type in the advanced page settings. Then add the link to your custom CSS in your layout templates after the link to default.css, for example:

```
<link rel="stylesheet" href="/stylesheets/custom.css" />
```

4.10. STEP 8: GO LIVE

The final task is to view the entire portal site and check all the workflows. You can publish each page or all of the pages in the Changes section. Once you're happy with everything, make a final check that all pages have been published.

Now you're ready to remove any access code for the portal:



Congratulations! Your developer portal is now live and ready to help build your developer community.