
Gemnasium docs Documentation

Release 1.0.0-rc1

Gemnasium

February 01, 2017

1	Welcome	3
2	Release Notes	5
2.1	1.0.0-rc1 - 2017-01-16	5
2.2	1.0.0-beta4 - 2016-12-15	5
2.3	1.0.0-beta3 - 2016-11-29	5
2.4	1.0.0-beta2 - 2016-11-18	6
2.5	1.0.0-beta1 - 2016-10-21	6
3	Getting Started	7
3.1	What is Gemnasium Enterprise?	7
3.2	First steps	7
4	Prerequisites	9
4.1	System Requirements	9
5	Firewall configuration	11
5.1	Incoming traffic	11
5.2	Outgoing traffic	11
6	Run Gemnasium Enterprise	13
6.1	Docker Hub Account	13
6.2	Preparing volumes	13
6.3	License key	13
6.4	Configuring SSL	13
6.5	Running without SSL	14
6.6	SELinux	15
6.7	Volumes	15
6.8	Logging	16
6.9	Obtaining a shell	16
7	SMTP setup	17
7.1	Configuration	17
8	Environment Variables	19
8.1	Variables persistence	19
8.2	Variables List	19
9	Upgrade Gemnasium Enterprise	21

9.1	Using latest version	21
9.2	Using tagged version	21
10	Troubleshooting	23
11	Data Backup	25
11.1	Snapshots	25
11.2	Using docker	25
12	GitHub	27
12.1	GitHub.com	27
12.2	GitHub Enterprise	28
13	GitLab	29
13.1	GitLab.com	29
13.2	GitLab CE/EE	30
14	Bitbucket Cloud	31
14.1	Adding OAuth consumers on Bitbucket.org	31
14.2	Configure Gemnasium Enterprise to use Bitbucket.org	31
14.3	Advanced configuration	32
15	Slack	33

Contents:

Welcome

Welcome to the Gemnasium Enterprise 1.0.0 documentation, where you can find information and guides to help you with Gemnasium Enterprise and start exploring its features.

Use the left navigation bar to browse the documentation, the Search bar in the top-left to look for something specific, or the links below to access some highlights.

Note: Gemnasium Enterprise Edition will be referred as “GEE” in this documentation

Release Notes

2.1 1.0.0-rc1 - 2017-01-16

This is the last pre-release before 1.0.0, if no bug is found.

- [FEATURE] Add Bitbucket.org (Bitbucket Cloud) Support
- [FEATURE] Add project logs with realtime update
- [FEATURE] Improve project notification channels configuration
- [FEATURE] Allow to edit existing project notification channel
- [FEATURE] Improve user notifications configuration
- [BUG] Fix various UI bugs
- [BUG] Some PHP packages were not fully synced

2.2 1.0.0-beta4 - 2016-12-15

- [FEATURE] “New package release” notifications via Slack and email
- [BUG] Fix file upload form when adding unsupported file
- [BUG] Fix left menu bar behavior on small devices layout
- [BUG] Fix oauth signup error handling

2.3 1.0.0-beta3 - 2016-11-29

- [FEATURE] GitLab Support
- [FEATURE] New notifications in the UI

Known issues:

- [BUG][GITLAB] Symlinks on dependency files are not followed
- [BUG][GITLAB] Dependency files greater than 2MB are ignored
- [BUG] Can’t sign-in using an oauth account if the same email is already used

2.4 1.0.0-beta2 - 2016-11-18

- [FEATURE] Display commits in project page
- [FEATURE] Internal logging (live feeds will be available in beta3)
- [BUG] Fix a security issue when adding a project to a team. The tokens of the team owner were used instead of the user's.
- [BUG] Fix display issues in Firefox
- [BUG] Fix UI Cache issues
- [BUG] Offline projects color was not updated when pushing new dependency files
- [BUG] Sync was failing when commit already existed
- [BUG] Fix a bug preventing to upload new files in Offline projects

Known issues:

- [FEATURE] Gitlab support is delayed to beta3
- [BUG] Can't sign-in using an oauth account if the same email is already used

2.5 1.0.0-beta1 - 2016-10-21

- First private beta
- GitHub.com and GitHub Enterprise support
- Slack integration for notifications

Getting Started

3.1 What is Gemnasium Enterprise?

Gemnasium Enterprise Edition (GEE) is the self-hosted version of Gemnasium, designed to run on your own servers, inside your network. A private license key is required to use the software, otherwise no data will be synced with gemnasium.com (making your instance unusable).

To get a valid license key, please contact our support.

3.2 First steps

Gemnasium Enterprise is designed as a collaborative tool. Each project must be added inside a team context, that's why the first thing to do is to create your team. The name must be unique in the Gemnasium Enterprise instance.

3.2.1 Add your colleagues

Go to the “Team” section of the main menu, and press the “+ Add member” button to invite co-workers. The role of the invited user will grant him/her access to your team:

- “Admin” users will be able to manage project and users
- “Basic” users will be able to access projects only

3.2.2 Add projects

The first shortcut in the main menu “+ Add project” will allow you to add new projects to a team. Once the team is selected, a source must be chosen. By default, only “offline” projects are available, because your instance doesn't know anything about external sources.

Offline projects will allow to upload dependencies files (Gemfile, Gemfile.lock, package.json,...) directly to a project. This kind of project is called “offline”, because Gemnasium can't synchronize the files with an external source.

To add projects from github.com, or GitHub Enterprise, an external source must be created. Please refer to the corresponding pages in the “INSTALLATION AND CONFIGURATION” section of this documentation. More source types will be available in the next versions of Gemnasium Enterprise.

Prerequisites

Note: You must have a valid subscription to setup Gemnasium Enterprise. If you don't have a subscription yet, please contact our support.

4.1 System Requirements

4.1.1 Hardware

- Physical or virtual system, or an instance running on a public or private IaaS.
- 1 vCPU
- Minimum of 8GB RAM
- Minimum 15GB hard disk space

4.1.2 Software

- OS: RedHat (RHEL, Atomic Host, Centos & Fedora flavors), Debian Stable. Any OS running docker should work but is not officially supported.
- Docker $\geq$ 1.9.1

Firewall configuration

Gemnasium Enterprise is designed to run behind your firewalls, inside your network. It should be completely isolated from the outside, especially from incoming connections.

5.1 Incoming traffic

Gemnasium only exposes two ports:

Port	Usage	Protocol
80	clear connection	http
443	secure connection	https

It's your responsibility to configure your network and firewall to restrict access to these ports. By default, Gemnasium exposes port 80 only to redirect to port 443. Custom ports might be used, refer to [License key](#) if needed.

5.2 Outgoing traffic

While Gemnasium Enterprise should be completely isolated from the outside for incoming connections, some outgoing traffic is necessary for normal operation. The following ports must be open on your firewall for outgoing traffic:

Address	Port	Protocol	Usage
sync.gemnasium.com	443	tcp	Sync with Gemnasium main DB
index.docker.io	443	tcp	Pull updates of gemnasium/enterprise image

5.2.1 What data is sent to sync.gemnasium.com?

Your content is private, and will remain private. But synchronizing all packages, versions, changelogs, advisories, etc. with Gemnasium's main DB would require a huge amount of storage and a lot of bandwidth.

To avoid this situation, your instance of Gemnasium Enterprise periodically sends a list of the packages (dependencies) used in your projects to <https://sync.gemnasium.com>. In return, Gemnasium syncer provides all the metadata corresponding to this request, including the security advisories.

Note: Private dependencies are completely ignored by the syncer.

Advisories can be created at any time on Gemnasium.com; that's why your instance synchronizes hourly. If one of your projects is using a dependency affected by a security issue, you will be instantly notified by your instance.

Note: Gemnasium is using data (advisories) from security companies (partnerships only). Your data will never be submitted directly to these companies, but Gemnasium may share anonymized statistical data.

Run Gemnasium Enterprise

Gemnasium Enterprise is shipped as a single, all-included [docker image](#).

6.1 Docker Hub Account

In order to be able to download the Gemnasium Enterprise docker image from the [Docker Hub](#), you must be authenticated on docker hub, and provide the username used to pull the image to Gemnasium.

To have access to the Gemnasium Enterprise image, send us the Docker Hub user you want; we will share the image with that user.

6.2 Preparing volumes

Persistent volumes are needed to store Gemnasium Enterprise data. The easiest way to get started, is to create local volumes on your server, but it can be any kind of volume supported by the docker engine.

See also:

Please refer to Docker Volumes for more information: <https://docs.docker.com/engine/tutorials/dockervolumes/>

To create local volumes, on you server:

```
docker volume create --name gemnasium-data
docker volume create --name gemnasium-logs
```

6.3 License key

As you will see, in all the `docker run` commands, there is a `-e LICENSE_KEY=YOUR_OWN_LICENSE_KEY \`. You need to replace `YOUR_OWN_LICENSE_KEY` with the license we gave you. If you don't have one, please get in touch and we'll get that sorted.

6.4 Configuring SSL

A valid certificate must be provided to run Gemnasium Enterprise with the integrated SSL server (nginx). If you don't need Gemnasium Enterprise to serve content on https directly, go directly to the section: [Running without SSL](#).

The certificate files **must** be named after the server name.

Example: for *gemnasium.example.com*, the certificate files **must** be named:

- *gemnasium.example.com.cert.pem* for the certificate
- *gemnasium.example.com.key.pem* for its private key

Gemnasium will look for 2 files with the *.cert.pem* and *.key.pem* suffix.

If the certificate has an intermediate chain, it must concatenated after the server certificate:

```
cat server.cert.pem ca-chain.cert.pem > gemnasium.example.com.cert.pem
```

The 2 files **must** be available in */etc/gemnasium/ssl*, inside the container.

```
docker run --detach \
  --name gemnasium \
  --restart always \
  -v /host/path/to/ssl:/etc/gemnasium/ssl \
  -p 80:80 -p 443:443 \
  -e LICENSE_KEY=YOUR_OWN_LICENSE_KEY \
  -v gemnasium-data:/var/opt/gemnasium/ \
  -v gemnasium-logs:/var/log/ \
  -v /var/run/docker.sock:/var/run/docker.sock \
  gemnasium/enterprise:latest
```

Note: Gemnasium needs the docker socket to be mounted only if the Reports feature is being used. If not, the line *-v /var/run/docker.sock:/var/run/docker.sock* can be safely removed.

This will pull and start Gemnasium Enterprise. Your instance will be available at <https://gemnasium.example.com> after a few seconds.

If you need to use a different port for https than 443, use the *EXTERNAL_URL* env var to specify the full URL of your Gemnasium Enterprise server, including the port used:

```
docker run --detach \
  --name gemnasium \
  --restart always \
  -v /host/path/to/ssl:/etc/gemnasium/ssl \
  -p 80:80 -p 8443:443 \
  -e LICENSE_KEY=YOUR_OWN_LICENSE_KEY \
  -e EXTERNAL_URL=https://gemnasium.example.com:8443/ \
  -v gemnasium-data:/var/opt/gemnasium/ \
  -v gemnasium-logs:/var/log/ \
  -v /var/run/docker.sock:/var/run/docker.sock \
  gemnasium/enterprise:latest
```

and start browsing <https://gemnasium.example.com:8443/>

6.5 Running without SSL

Warning: We strongly discourage running Gemnasium Enterprise without any SSL termination. This section is present if you already have SSL terminations, like secured reverse-proxies, ssl appliances, etc.

Run the image:

```
docker run --detach \
  --name gemnasium \
  --restart always \
  -e REDIRECT_HTTP_TO_HTTPS=false \
  -p 80:80 \
  -e LICENSE_KEY=YOUR_OWN_LICENSE_KEY \
  -v gemnasium-data:/var/opt/gemnasium/ \
  -v gemnasium-logs:/var/log/ \
  -v /var/run/docker.sock:/var/run/docker.sock \
  gemnasium/enterprise:latest
```

Note: The environment variable REDIRECT_HTTP_TO_HTTPS is *true* by default, and must be false in this case.

The service is available after a few seconds on the port 80 of your server.

6.6 SELinux

Gemnasium Enterprise can't be run directly on SELinux servers, because:

1. The volumes will be readonly by default
2. The docker socket will be restricted to the host

Use this command instead:

```
docker run --detach \
  --name gemnasium \
  --restart always \
  -v /host/path/to/ssl:/etc/gemnasium/ssl \
  -p 80:80 -p 443:443 \
  -e LICENSE_KEY=YOUR_OWN_LICENSE_KEY \
  -v gemnasium-data:/var/opt/gemnasium/:Z \
  -v gemnasium-logs:/var/log/:Z \
  -v /var/run/docker.sock:/var/run/docker.sock:Z \
  gemnasium/enterprise:latest
```

This will label the content inside the container with the exact MCS label that the container will run with, basically it runs `chcon -Rt svirt_sandbox_file_t -l s0:c1,c2 /var/db` where `s0:c1,c2` differs for each container.

See also:

More info: <http://www.projectatomic.io/blog/2015/06/using-volumes-with-docker-can-cause-problems-with-selinux/>

Please refer to this project to install the proper SELinux module to fix the second point.

6.7 Volumes

Gemnasium is storing data in two folders, which should be mounted as volumes

Local location	Location in container	Usage
gemnasium-data (volume)	/var/opt/gemnasium	Gemnasium data
gemnasium-logs (volume)	/var/log	Gemnasium logs

Gemnasium data is composed mostly of the PostgreSQL database files, but also nsq data, etc. These files must be backed up, refer to the [Data Backup](#) section.

The `/var/log` contains the OS logs, and everything dedicated to gemnasium in `/var/log/gemnasium`.

Finally, as explained in the [License key](#) section, your certificate and key must be available in the `/etc/gemnasium/ssl` folder.

6.8 Logging

By default, all logs will be sent to the standard output of the container (`stdout`), along with files in `/var/log`. This makes it easier to troubleshoot if needed.

6.8.1 Graylog

Gemnasium Enterprise can be configured to log to a distant [Graylog](#) server. To enable this feature, use the following environment variables:

Env variables	Usage
<code>GRAYLOG_SERVICE_HOST</code>	Graylog input hostname/ip
<code>GRAYLOG_SERVICE_PORT</code>	Graylog input port

Example:

```
docker run --detach \
  --name gemnasium \
  --restart always \
  -v /host/path/to/ssl:/etc/gemnasium/ssl \
  -p 80:80 -p 443:443 \
  -v gemnasium-data:/var/opt/gemnasium/ \
  -v gemnasium-logs:/var/log/ \
  -v /var/run/docker.sock:/var/run/docker.sock \
  -e GRAYLOG_SERVICE_HOST=logs.example.log
  -e GRAYLOG_SERVICE_PORT=1515
  gemnasium/enterprise:latest
```

Both variables must be set to activate the GELF output.

6.9 Obtaining a shell

The docker image doesn't have a SSH server, because docker provides everything needed to get a shell console inside the container:

```
docker exec -it gemnasium bash
```

will create a new bash session, with the root user.

Warning: With great power comes great responsibility: as root, you can damage files inside the container, including your persisted data.

SMTP setup

In order to send notifications (team invitations, digests, etc.), Gemnasium Enterprise needs a SMTP server. There is no such server included in the docker image, so an external SMTP server should be used. Note that SMTP connectivity is not fully required, just recommended.

7.1 Configuration

SMTP can be configured by passing environment variable to the docker container:

Env variables	Usage
SMTP_SERVICE_HOST	Server host address
SMTP_SERVICE_PORT	Server port
SMTP_USER_NAME	Username
SMTP_PASSWORD	Password
SMTP_INSECURE	Skip SSL verification

The SMTP password can be plain auth, or a secret (CRAM-MD5 auth). These variables are all optional.

SSL and TLS are supported, and will be automatically used if the port (SMTP_SERVICE_PORT) is 465 or 587.

Environment Variables

Gemnasium Enterprise is entirely configured using environment variables.

8.1 Variables persistence

If you don't want to specify by hand all environment variables in `docker run`, you can create a file including all the variables to be set: Compose expects each line in an env file to be in `VAR=VAL` format. Lines beginning with `#` (i.e. comments) are ignored, as are blank lines.

To use the environment file, add the `--env-file=[file name]` to your `docker run` command line.

With `docker-compose`, the name should be named `.env` in the same directory as your `docker-compose.yml` file, and will be loaded automatically. Remember to specify the env variables expected in the file, otherwise `docker-compose` will simply ignore them.

8.2 Variables List

Env variables	Usage	Required	De- fault
Main			
<code>LICENSE_KEY</code>	Your GEE private license key	required	
Logging			
<code>GRAYLOG_SERVICE_HOST</code>	Graylog input hostname/ip	optional	
<code>GRAYLOG_SERVICE_PORT</code>	Graylog input port	if <code>GRAYLOG_SERVICE_HOST</code> is set	
SMTP			
<code>SMTP_SERVICE_HOST</code>	SMTP server host	optional	
<code>SMTP_SERVICE_PORT</code>	SMTP server port	optional	25
<code>SMTP_USER_NAME</code>	SMTP user	optional	
<code>SMTP_PASSWORD</code>	SMTP password (plain auth) or secret (CRAM-MD5 auth)	optional	
<code>SMTP_INSECURE</code>	SMTP skip SSL verification	optional	false

Upgrade Gemnasium Enterprise

While the data about packages, versions, etc. are automatically updated, and stored in your local database, Gemnasium Enterprise won't upgrade itself automatically.

9.1 Using latest version

Every night, a new build of Gemnasium Enterprise source code will generate a `gemnasium/enterprise:latest` image.

To upgrade Gemnasium Enterprise to a new version:

1- Stop the container:

```
docker stop gemnasium
```

2- Remove the container:

```
docker rm gemnasium
```

3- Pull the new image:

```
docker pull gemnasium/enterprise:latest
```

4- Run the image again (see *Preparing volumes*). Gemnasium Enterprise will update your data automatically, if necessary.

9.2 Using tagged version

Living on the edge is probably not a good idea on the long term. Gemnasium will provide new tagged version for each release.

The available tags are listed here: <https://hub.docker.com/r/gemnasium/enterprise/tags/>

Troubleshooting

Read container logs:

```
docker logs -f gemnasium
```

Enter running container:

```
docker exec -it gemnasium /bin/bash
```

Data Backup

Gemnasium Enterprise will store its data in a persistent volume (cf. *Volumes*).

It is YOUR responsibility to backup this volume. Gemnasium Enterprise has no capacities of backing up data.

11.1 Snapshots

Disk snapshots are probably the best option to backup your data. All the data stored in the persistent volume will be saved at once, and can be restored at anytime.

If a restore is needed, remember to stop the container before restoring anything:

```
docker stop gemnasium
```

and remove it completely:

```
docker rm gemnasium
```

Once the data is restored, run again the image: *Preparing volumes*

11.2 Using docker

As explained in the “[Manage data in containers](#)” [docker](#) page, docker may be used to create a tarball:

```
docker run --rm --volumes-from gemnasium -v $(pwd):/backup ubuntu tar cvf /backup/backup.tar  
/var/opt/gemnasium
```

It is highly recommended to stop the container before doing this operation.

11.2.1 Restoring data with docker

With your backup in the local directly (pwd), untar your archive to restore the data in the `gemnasium-data` volume:

```
$ docker run --rm -v $(pwd):/backup -v gemnasium-data:/var/opt/gemnasium/ gemnasium/enterprise bash -c 'tar xvf /backup/backup.tar -C /var/opt/gemnasium'
```

Once the data is restored, run again the image: *Preparing volumes*

12.1 GitHub.com

Before being able to add projects from github.com, Gemnasium Enterprise needs to be configured to be able to access it.

There're parts to it. First, creating an OAuth application on GitHub and then configuring Gemnasium Enterprise to use that application.

12.1.1 Adding OAuth applications on GitHub

The first step we need to do create two new OAuth applications. You read that right; Gemnasium Enterprise needs two different OAuth applications. One to enable “Login with GitHub” and one to synchronize projects.

To do that:

- go on <https://github.com/settings/developers> (or alternatively, add the application to an organization: [https://github.com/organizations/\[your_org\]/settings/applications](https://github.com/organizations/[your_org]/settings/applications))
- click on the “Register a new application”
- Set the name to “Gemnasium Enterprise Login”, Homepage URL to your Gemnasium Enterprise base URL (example: “<https://gemnasium.example.com/>”) and finally the callback URL to be `{homepage URL}/auth/auth/github.com/login/callback` where you replace the home with your Gemnasium Enterprise host (example: <https://gemnasium.example.com/auth/auth/github.com/login/callback>)
- Click on the “Register application” button.

You will need some the “Client ID” and “Client Secret” you see on the confirmation page for the next section. You can keep that tab open and open a new tab to <https://github.com/settings/developers> to create the second application.

To create the second application required:

- go on <https://github.com/settings/developers>
- click on the “Register a new application”
- Set the name to “Gemnasium Enterprise Sync”, Homepage URL to your Gemnasium Enterprise base URL (example: “<https://gemnasium.example.com/>”) and finally the callback URL to be `{homepage URL}/auth/auth/github.com/sync/callback` where you replace the home with your Gemnasium Enterprise host (example: <https://gemnasium.example.com/auth/auth/github.com/sync/callback>)
- Click on the “Register application” button.

12.1.2 Configure Gemnasium Enterprise to use GitHub

A convenient script is provided to configure your instance:

```
docker exec -it gemnasium configure
```

Your Gemnasium Enterprise users are now able to login using their GitHub account. A new source named “GitHub” is also available on the “Add Project” screen.

12.2 GitHub Enterprise

GitHub Enterprise is no different than GitHub, the steps are the same as above, just replace *github.com* with your GitHub Enterprise instance URL.

When using the configuration script, make sure to select “GitHub Enterprise” instead of “GitHub.com”. The script will ask for your instance URL, and to name it.

Several GitHub Enterprise instances can be configured, just name them accordingly.

12.2.1 Requirements

Gemnasium Enterprise has been tested against GitHub Enterprise $\geq 2.7.X$. If your version is older than 2.7 series, please contact our support.

13.1 GitLab.com

Before being able to add projects from your GitLab Enterprise (GHE) instance, Gemnasium Enterprise needs to be configured to be able to access it.

There're two parts to it. First, creating an OAuth application on GitLab Enterprise and then configuring Gemnasium Enterprise to use that application.

13.1.1 Adding an OAuth application on GitLab Enterprise

The first step we need to do create a new OAuth application.

To do that:

- go on <https://ghe.example.com/settings/developers> where *ghe.example.com* is the hostname of your GHE instance
- click on the “Register a new application”
- Set the name to “Gemnasium Enterprise Login”, Homepage URL to your Gemnasium Enterprise base URL (example: “<https://gemnasium.example.com/>”) and finally the callback URL to be {homepage URL}/auth/auth/{GitLab Enterprise host}/login/callback where you replace the home and the GHE host (example: <https://gemnasium.example.com/auth/auth/gitlab-enterprise.example.com/login/callback>)
- Click on the “Register application” button.

You will need some the client ID & secret you see on the confirmation page for the next section. You can keep that tab open and open a new tab to <https://gitlab.com/settings/developers> to create the second application.

To create the second application required:

- go on <https://ghe.example.com/settings/developers> where *ghe.example.com* is the hostname of your GHE instance
- click on the “Register a new application”
- Set the name to “Gemnasium Enterprise Sync”, Homepage URL to your Gemnasium Enterprise base URL (example: “<https://gemnasium.example.com/>”) and finally the callback URL to be {homepage URL}/auth/auth/{GitLab Enterprise host}/sync/callback where you replace the home and the GHE host (example: <https://gemnasium.example.com/auth/auth/gitlab-enterprise.example.com/sync/callback>)
- Click on the “Register application” button.

13.1.2 Configure Gemnasium Enterprise to use GitLab Enterprise

A convenient script is provided to add everything at once:

```
docker exec -it gemnasium configure
```

Select “GitLab Enterprise”, and then fill the corresponding fields with the values from the apps created above.

Your Gemnasium Enterprise users are now able to login using their GitLab Enterprise account. A new source with the name entered in the configure script is also available on the “Add Project” screen.

13.2 GitLab CE/EE

GitLab.com is actually running the same code base as GitLab CE/EE, so the steps are the same as above. Just replace GitLab.com with your private GitLab instance URL.

When using the configuration script, make sure to select “GitLab CE/EE” instead of “GitLab.com”. The script will ask for your instance URL, and to name it.

Several GitLab instances can be configured, just name them accordingly.

13.2.1 Requirements

Gemnasium Enterprise is compatible with GitLab ≥ 8.14 . There is a bug in nginx affecting lower versions of GitLab.

Bitbucket Cloud

Before being able to add projects from bitbucket.org (A.K.A. Bitbucket Cloud), Gemnasium Enterprise needs to be configured to be able to access it.

First add an OAuth consumer on bitbucket.org, then configure Gemnasium Enterprise to use that OAuth consumer.

14.1 Adding OAuth consumers on Bitbucket.org

The first step is to go to open the “Add OAuth consumer” form:

- Go to the Bitbucket Settings page.
- If needed, use the dropdown list and select the Bitbucket account of your company.
- Click on the “OAuth” item in the sidebar.
- Click on the “Add consumer” button.

Or simply visit: [https://bitbucket.org/account/user/\[your_account\]/oauth-consumers/new](https://bitbucket.org/account/user/[your_account]/oauth-consumers/new)

Then fill the form:

- Set the name to “Gemnasium Enterprise”.
- Set the callback URL to be `{homepage URL}/auth/auth/bitbucket.org` where you replace the home with your Gemnasium Enterprise host (example: `https://gemnasium.example.com/auth/auth/bitbucket.org`).
- Set the URL to your Gemnasium Enterprise base URL (example: “`https://gemnasium.example.com/`”).
- Grant permissions to: read account, read repositories, read and write webhooks.
- Save the new OAuth consumer.

You’ll then be redirected to the OAuth Consumers page and the consumer named “Gemnasium Enterprise” will be visible in the list. Click on its name to expand the item and retrieve the credentials: the consumer key and its secret.

14.2 Configure Gemnasium Enterprise to use Bitbucket.org

A convenient script is provided to configure your instance:

```
docker exec -it gemnasium configure
```

Your Gemnasium Enterprise users are now able to login using their Bitbucket.org account. A new source named “Bitbucket.org” is also available on the “Add Project” screen.

14.3 Advanced configuration

The default configuration described above enables both Bitbucket.org Sign In and project synchronization with Bitbucket.org. The advanced configuration makes it possible to restrict the integration to one of these two features.

To enable Bitbucket.org Sign Up, add an OAuth consumer with the permissions “account” and “email”, then run these commands to configure Gemnasium Enterprise accordingly:

```
docker exec -it gemnasium auth provider create bitbucket.org bitbucket https://bitbucket.org/site/oauth2/authorize
docker exec -it gemnasium auth clients create bitbucket.org login OAUTH_CONSUMER_KEY OAUTH_CONSUMER_SECRET
```

To enable Bitbucket.org Synchronization, add an OAuth consumer with the permissions “repository” and “webhook”, then run these commands to configure Gemnasium Enterprise accordingly:

```
docker exec -it gemnasium auth provider create bitbucket.org bitbucket https://bitbucket.org/site/oauth2/authorize
docker exec -it gemnasium auth clients create bitbucket.org sync OAUTH_CONSUMER_KEY OAUTH_CONSUMER_SECRET
docker exec -it gemnasium repo-syncer sources create bitbucket.org "Bitbucket Cloud" bitbucket https://bitbucket.org
```

OAUTH_CONSUMER_KEY and OAUTH_CONSUMER_SECRET must be replaced with the key and the secret of the OAuth consumer created on Bitbucket.org. See [Adding OAuth consumers on Bitbucket.org](#) above.

Slack

Because it's an enterprise version installed on your server, you need to do a few things in order to add Slack support.

The first step is to create a new Slack application, on Slack itself. Here are the steps:

- Go on https://api.slack.com/apps?new_app=1
- Set the App Name to be “Gemnasium Enterprise”, and then click on the “Create App” button.
- You will need the Client ID and Client Secret in a moment. You can keep them around or go back to see them when you need them.
- (optional) By default, there is no Gemnasium icon for the messages sent on Slack. If you want it, which we suggest, you can upload it on the “Basic Information” screen of the app you just created, in the “Display Information” section. You can use this icon.
- Go in the “OAuth & Permissions” section (in the left menu) and set the redirect URL to <https://gemnasium.example.com/auth/auth/slack.com/webhook/callback>

Now, we need to configure Gemnasium Enterprise to use it:

```
docker exec -it gemnasium configure
```

Select “Slack”, and then fill the corresponding fields with the values from the apps created above.

Now you only need to configure Slack notifications for a project in the web UI. You can do that in the “Settings” of each project.