
Biostar Central Documentation

Release 2.0

Istvan Albert

Apr 29, 2020

Contents

1	Features	3
2	Support	5
3	Quick Start	7
3.1	Install	7
3.2	Manage	10
3.3	Deploy	12
3.4	Customize	13
3.5	About	15
4	Indices and tables	17

BioStar is a [Python](#) and [Django](#) based Q&A software licensed under the *MIT Open Source License*. It is a simple, generic, flexible and extensible Q&A framework.

The site has been developed by **scientists and for scientists**. It aims to address requirements and needs that scientific communities have.

Biostar is the software that runs several science oriented Q&A sites:

- Biostars Bioinformatics Q&A at: <https://www.biostars.org>
- Galaxy User support site: <https://biostar.usegalaxy.org>
- Metabolomics Q&A: <http://www.metastars.org>
- Neurostars: <http://www.neurostars.org>

CHAPTER 1

Features

The site has been developed by scientists for scientists. It aims to address specific needs that scientific communities have.

- Standard Q&A: post questions, answers, comments, user moderation, voting, badges, threaded discussions
- Email integration: import previous posts from mailing lists, reply to posts via email
- RSS Planet: feed aggregation from different sources
- External authentication: authenticate users with a different web service

CHAPTER 2

Support

The software is open source and free to use under the most permissible license.

The developers of the software are also available to provide commercial level support for deploying Biostar sites for entire organizations. Contact: admin@biostars.org

Requirements: *Python 2.7*

Official documentation is located at <http://docs.biostars.org>

CHAPTER 3

Quick Start

From the biostar source directory:

```
# Install the requirements.
pip install --upgrade -r conf/requirements/base.txt

# Load the environment variables.
source conf/defaults.env

# Initialize database, import test data, and run the site.
./biostar.sh init import index run
```

Visit <http://localhost:8080> to see the site loaded with default settings. Enjoy.

For more information see the documentation below:

3.1 Install

The sourcecode can be obtained via:

```
git clone https://github.com/ialbert/biostar-central.git
```

3.1.1 Getting started

Get the source and switch to the source directory. The recommended installation is via virtualenv and pip:

```
# Install the requirements.
pip install --upgrade -r conf/requirements/base.txt

# Load the environment variables.
source conf/defaults.env
```

(continues on next page)

(continued from previous page)

```
# Initialize, import test data and run the site.
./biostar.sh init import run
```

Visit `http://localhost:8080` to see the site loaded with default settings.

The default admin is `foo@bar.com` password `foobar`. The default email handler will print to the console. You can reset the password for any user then copy paste the password reset url into the browser.

Run the manager on its own to see all the commands at your disposal:

```
./biostar.sh
```

To enable searching you must the content with:

```
./biostar.sh index
```

3.1.2 Blog Aggregation

Biostar has the ability to aggregate blog feeds and allow searching and linking to them. List the RSS feeds in a file then:

```
# Initialize with new feed urls (see example)
python manage.py planet --add biostar/apps/planet/example-feeds.txt

# Download all feeds (usually performed daily)
python manage.py planet --download

# Add one new blog entry for each feed the downloaded file (if there is any)
python manage.py planet --update 1
```

3.1.3 Social authentication

The social logins settings will need to be initialized with the proper authentication parameters. Typically this involves creating an application at the provider and obtaining the credentials.

See the `conf/defaults.env` for the proper variable naming.

Adding Facebook authentication:

- Create Authentication App: <http://developers.facebook.com/setup/>
- More information: Facebook Developer Resources: <http://developers.facebook.com/docs/authentication/>

Adding Google authentication:

- Google Developer Console: <https://cloud.google.com/console/project>
- Create new project and copy data from credentials
- Callback must be `http://domain/accounts/google/login/callback/`

Twitter:

- Add your application at Twitter Apps Interface: <http://twitter.com/apps/>

3.1.4 External authentication

Other domains can provide authentication for Biostar by setting a cookie with a certain value. For this to work Biostar will have to be set to run as a subdomain of the hosting site.

Cookie settings

The cookie value needs to contain the email:hash as value. For example if the EXTERNAL_AUTH django settings are:

```
# Cookie name, cookie secret key pair
EXTERNAL_AUTH = [
    ("foo.bar.com", "ABC"),
]
```

If an unauthenticated user sends a cookie named foo.bar.com with the value:

```
foo@bar.com:d46d8c07777e3adf739cfc0c432759b0
```

then Biostar will automatically log in the user. It will automatically create an account for the user if the email does not already exist.

Setting the EXTERNAL_LOGIN_URL and EXTERNAL_LOGOUT_URL settings will also perform the redirects to the external site login and logout urls:

```
EXTERNAL_LOGIN_URL = "http://some.site.com/login"
EXTERNAL_LOGOUT_URL = "http://some.site.com/logout"
```

Generating the value is simple like so:

```
email = "foo@bar.com"
digest = hmac.new(key, email).hexdigest()
value = "%s:%s" % (email, digest)
```

Prefill post

Set the title, tag_val, content and category fields of a get request to pre-populate a question:

```
http://localhost:8080/p/new/post/?title=Need+help+with+bwa&tag_val=bwa+samtools&
→content=What+does+it+do?&category=SNP-Calling
```

3.1.5 Migrating from Biostar 1.X

Due to the complete rework there is no database schema migration.

Instead users of Biostar 1 site are expected to export their data with a script provided in Biostar 1 then import it with a management command provided with Biostar 2.

The migration will take the following steps:

1. Set the BIOSTAR_MIGRATE_DIR environment variable to point to a work directory that will hold the temporary data, for example `export BIOSTAR_MIGRATE_DIR=~/.tmp/biostar_export`
2. Load the environment variables for the Biostar 1 site then run `python -m main.bin.export -u -p -v`. This will dump the contents of the site into the directory that BIOSTAR_MIGRATE_DIR points to.

3. Load the environment variables for you Biostar 2 site then run the `./biostar.sh import_biostar1` command.

Some caveats, depending how you set the variables you may need to be located in the root of your site. This applies for the default settings that both sites come with, as the root is determined relative to the directory that the command is run in.

3.2 Manage

There are a number of data management commands that come with Biostar.

3.2.1 The `biostar.sh` manager

The **biostar.sh** shell command automatizes a number of commonly used tasks. Run it with no parameters to get help on a typical usage:

```
Usage:

$ biostar.sh <command>

Multiple commands may be used on the same line:

$ biostar.sh init import run

Commands:

init      - initializes the database
run       - runs the development server
index     - initializes the search index
test      - runs all tests
env       - shows all customizable environment variables

import    - imports the data fixture JSON_DATA_FIXTURE=import/default-fixture.json.
→gz
dump      - dumps data as JSON_DATA_FIXTURE=import/default-fixture.json.gz
delete    - removes the sqlite database DATABASE_NAME=biostar.db

pg_drop   - drops postgres DATABASE_NAME=biostar.db
pg_create - creates postgres DATABASE_NAME=biostar.db
pg_import f.gz - imports the gzipped filename into postgres DATABASE_NAME=biostar.db

Use environment variables to customize settings. See the docs.

DJANGO_SETTINGS_MODULE=biostar.settings.base
```

3.2.2 Subcommands

In addition there are a number of data management commands that are implemented for the each app. Run:

```
python manage.py help
```

And look for the output for the app `[server]`, these commands will look like:

```
[server]
    biostar_pg_dump
    delete_database
    import_biostar1
    import_mbox
    initialize_site
    prune_data
    usermod
    sqlfix
    sitemap
    user_crawl
    test_email
    test_task
    patch
```

You can run each of these subcommands with the `-h` flag to get more information on them.

3.2.3 Command line tagging

There is a command line tool to perform content tagging based on a regular expression. The invocation is:

```
workon biostar
source live/deploy.env
python manage.py patch --tag "regexp:tag1,tag2,tag3"
```

Where the regular expression `regexp` will be searched against the content and when found matching tags `tag1`, `tag2`, `tag3` will be applied. Example:

```
python manage.py patch --tag "gff:gff, interval"
```

To detect what posts would be tagged but not actually perform the tagging pass the `--dry` command. In that case only the post titles will be listed:

```
python manage.py patch --tag "gff:gff, interval" --dry
```

This command will navigate through all questions in the database.

3.2.4 Example commands

Frequently used commands:

```
# Set the password for a user identified by their userid
python manage.py usermod -u 2 -p abcde

# Set the password for a user identified by their email
python manage.py usermod -e foo@bar -p abcde

# Rebuild the entire search index
python manage.py rebuild_index

# Reindex only what has changed in the last hour
python manage.py update_index --age 1

# Import 100 posts from a mbox file into biostar
```

(continues on next page)

(continued from previous page)

```
python manage.py import_mbox -f filename -l 100

# Create a postgres database dump
python manage.py biostar_pg_dump
```

3.3 Deploy

3.3.1 Getting started

Thanks to the modular structure of its design Biostar is able to integrate with a wide variety of backends and provides a number of configuration scripts and helper methods to different deployment options.

The choices made when deploying Biostar depend on the expected levels of traffic and number of posts that the site needs to manage. The examples that we provide are the two extremes, some deployments may use a combination of settings from both.

Example files can be found in the `live/deploy` folder.

The basic rule is to create a settings file based on the default settings. This means that the customized settings file will start with:

```
from biostar.settings.base import *
```

Then subsequently override the various settings for the current deployment. For example:

```
from biostar.settings.base import *
SITE_DOMAIN = "mysite.com"
SERVER_EMAIL = "myemail@mysite.com"
```

etc.

Technically a django deployment needs only a settings file, but in practice we use an environment file to populate a shell environment and a settings file that pulls some of these variables out of the environment.

We recommend that you start with the files in `live/deploy` folder and copy them another python package folder. The `simple.env` file shows the minimally necessary variables that need to be set.

```
source live/deploy/simple.env ./biostar.sh test
```

The `deploy.env` must specify the correct django settings module.

To run periodic scripts make sure that they load up the environment variables before executing the script.

3.3.2 Low traffic deployment

Suited to websites that distribute information to smaller organizations. It can be achieved with just python based solutions. Install the dependencies with:

```
pip install -r conf/requirements/deploy.txt
```

Copy the `conf/deploy/deploy.env` and `conf/deploy/deploy.py` files to a different location. Customize these as needed. To run the site invoke the waitress server that was installed above:

```
source live/deploy/simple.env
waitress-serve --port 8080 live.deploy.simple_wsgi:application
```


Create a crontab entry that updates the index every 30 minutes:

```
source live/deploy/simple.env
biostar.sh update_index
```

You are done.

3.3.3 High traffic deployment

While not required to be turned on the site supports compressing and precompiling the site assets. To make use of this functionality you will need to have `lessc` to be installed and you will need to set the `USE_COMPRESSOR=True` in your settings file.

To deploy the site with `postgresql` and `elasticsearch` install the requirements:

```
pip install --upgrade -r conf/requirements/all.txt
```

Start with the `conf/defaults.env` and files under `conf/deploy/*` and customize them. We typically copy these into the `live` folder. Remember to add an `__init__.py` file in this folder if you want to import your settings from it.

For high performance installation we recommend deploying the production servers with the following stack:

- Front end webserver with `nginx`
- Biostar WSGI running via `gunicorn`
- `Postgresql` as the database
- `Redis` as the job queue
- `Celery` for running the asynchronous jobs
- `Supervisord` keeping everything running
- `Elasticsearch` as the search engine

The `conf/server` folder has configuration files for `nginx`, `gunicorn` and `supervisord`. The `conf/fabs` folder has Fabric files to automate a large number of site deployment operations.

3.4 Customize

3.4.1 Getting started

To customize Biostar you will need to overwrite the settings.

The settings are located in two files, an environment file and a python module.

Technically speaking only the python module is absolutely required. We chose to put certain variables into an environment so that it is easier to share the same values across multiple settings as well as to have them available at the command line.

The capitalized variables defined in the `biostar/settings/base.py` file directly affect the operation of the site. Each group of variables is fairly well documented to describe what it is used for. Typically only a small subset will ever need to be changed.

A typical run sources a shell program and loads variables. The python module then looks for certain variables in the environment and uses those to set its parameters.

3.4.2 Custom modules

To get started create a new empty python file. Say `custom.py`. This file will govern the entire operation of your site and is the so called **django settings module**.

Place your django settings module into a folder that python will recognize as a package directory (has a `__init__.py` in it). For example we use the `live` directory in the biostar source. The file will then be located in `live/custom.py`. Into this new django settings module place the following:

```
from biostar.settings.base import *
```

This line will ensure that all the default variable are loaded from the base module. We can now selectively override one or more of these base variables.

By default the emails are printed to the console. To use a typical SMTP based email service add the following to your custom settings module so that the entire file looks like:

```
from biostar.settings.base import *
EMAIL_BACKEND = 'django.core.mail.backends.smtp.EmailBackend'
```

We are done with django settings module. Now an environment file needs to be created.

Copy the `conf/defaults.env` file to a new file. The name typically matches the settings file that you want to activate. Let's call it `live/custom.env`:

```
cp conf/defaults.env live/custom.env
```

Modify the `custom.env` file so that in it the `DJANGO_SETTINGS_MODULE` variable points to your custom django settings module `live/custom.py`. This needs to follow the Python convention for import, this means to use a dot instead of a slash like so `live.custom` find and override the line to look like this:

```
export DJANGO_SETTINGS_MODULE=live.custom
```

In this environment file you may also override other variables. Notably find the `EMAIL_HOST`, `EMAIL_PORT`, `EMAIL_USER`, `EMAIL_PASSWORD` variables and fill in the values that are specific to your internet provider. If you get an error sending emails then this information is not set properly.

To use the new environment to start the site by sourcing this script instead of the default one. You will need to run this once per terminal:

```
source live/custom.env
```

To test that the email was set up correctly:

```
python manage.py test_email
```

This command will send a test email to the email address listed in `ADMIN_EMAIL` in the environment file.

A typical site initialization would be:

```
source live/custom.env
./biostar.sh delete init import run
```

There are no limitations how many settings one may have. To check which environment is loaded run the `biostar.sh` manager on its own. The last printout will display the current django settings module.

3.4.3 Advanced example

To change the upper navigation bar look inside `biostar.settings.base` find and copy the following categories into your custom django settings module `custom.py` file then modify as wish:

```
# The start categories. These tags have special meaning internally.
START_CATEGORIES = [
    "Latest", "Tutorials", "Tools", "Jobs", "Forum", "Unanswered",
]

# These should be the most frequent (or special) tags on the site.
NAVBAR_TAGS = [
    "Assembly", "RNA-Seq", "ChIP-Seq", "SNP",
]

# The last categories. Right now are empty.
END_CATEGORIES = [

]

# These are the tags that will show up in the tag recommendation dropdown.
POST_TAG_LIST = NAVBAR_TAGS + ["software error"]

# This will form the navbar
CATEGORIES = START_CATEGORIES + NAVBAR_TAGS + END_CATEGORIES
```

See the `conf/defaults.env` and for all the parameters that can to be customized.

The `SITE_STYLE_CSS` and `SITE_LOGO` settings permit loading up custom styles. See the `/static/themes` folder for examples.

3.5 About

3.5.1 Team

For the most up to date information on contributors see: <https://github.com/ialbert/biostar-central/graphs/contributors>

Core developers

- Istvan Albert

Contributors

- Pindi Albert
- Alexandr Levchuck
- Eric Normandeau
- Egon Willingham

3.5.2 Supporting organizations

Biostar was initiated to serve the needs of the bioinformatics community.

The code has been released as an open source software thanks to the support from the following institutions :

- *The Pennsylvania State University* and the *Huck Institutes for the Life Sciences* at Penn State.
- *National Institutes of Health (NIH)*, specifically grant **NIH 5R25HG006243-02**, *Analyzing Next Generation Sequencing Data* Principal investigator: **Dr. Titus Brown** (Michigan State University)

Citing Biostar

- Parnell LD, Lindenbaum P, Shameer K, Dall'Olio GM, Swan DC, et al. 2011 BioStar: An Online Question & Answer Resource for the Bioinformatics Community. PLoS Comput Biol 7(10): e1002216. doi:10.1371/journal.pcbi.1002216

3.5.3 Support

We can also provide custom support for organizations or institutions. Depending on the situation and arrangements the costs would typically involve hosting fees and hourly system administration type charges. For more information contact **admin@biostars.org**

CHAPTER 4

Indices and tables

- `genindex`
- `modindex`
- `search`