
flicket

Release 0.3.0

evereux@gmail.com

Aug 17, 2022

CONTENTS:

1	Why Flicket?	3
1.1	Requirements	3
1.2	Installation	4
1.3	Administration	5
1.4	Exporting / Importing Flicket Users	7
1.5	Installing A Webserver	7
1.6	Adding Additional Languages	10
1.7	Flicket - FAQ	10
1.8	Screenshots	13
1.9	API	19
	Python Module Index	21
	Index	23

Flicket is a simple web based ticketing system written in Python using the flask web framework which supports English and French locales.

WHY FLICKET?

I could not find a simple open source ticketing system that I really liked. So, decided to have a crack at creating something written in Python.

1.1 Requirements

1.1.1 Operating System

This will run on either Linux or Windows. Mac is untested.

1.1.2 Python

Python =>3.6 - I have not tested earlier versions of Python 3.

1.1.3 SQL Database Server

Out of the box Flicket is configured to work with [MySQL](#). But there should be no reason other SQLAlchemy supported databases won't work just as well.

Note: When I last tried SQLite I had problems configuring the email settings within the administration settings. You may have to change them manually within SQLite.

1.1.4 Web Server

For a production environment a webserver such as [Apache](#) or [nginx](#) should be used to serve the application.

1.2 Installation

First read [Requirements](#).

It is good practise to create a virtual environment before installing the python package requirements. Virtual environments can be considered a sand boxed python installation for a specific application. They are used since one application may require a different version of a python module than another.

1.2.1 Getting Flicket

The source code for Flicket is hosted at GitHub. You can either get the latest frozen zip file or use the latest master branch.

Warning: If you are upgrading from a previous version please read the CHANGELOG.

Master Branch

Get the latest master branch from github using git:

```
git clone https://github.com/evereux/flicket.git
```

Alternatively, download and unzip the master branch [zip file](#).

1.2.2 Installing Python Requirements

Install the requirements using pip::

```
(env) C:\<folder_path>\flicket> pip install -r requirements.txt
```

1.2.3 Set Up

1. If using PostgreSQL or MySQL create your database and a database user that will access the flicket database. If using SQLite you can skip this step.

See [SQLAlchemy_documentation](#) for options.

2. Create the configuration json file:

```
python -m scripts.create_json
```

3. If you aren't using SQLite edit *config.json* and change "db_driver". "null" should be replaced by the driver you are using. See the documentation above regarding engines, dialects and drivers. For example, if you are using a MySQL database and want to use the pymysql driver.

```
"db_driver": "pymysql"
```

4. Install the driver you are using if not using SQLite. For example if you are using a MySQL database and want to use the pymysql driver.


```
pip install pymysql
```

5. Upgrade the database using manage.py from the command line:

```
flask db upgrade
```

6. Run the set-up script:. This is required to create the Admin user and site url defaults. These can be changed again via the admin panel once you log in:

```
flask run-set-up
```

7. Running development server for testing:

```
flask runserver
```

Log into the server using the username *admin* and the password defined during the setup process.

1.3 Administration

1.3.1 Command Line Options

From the command line the following options are available.

```
python manage.py

usage: manage.py [-?]
                {db,run_set_up,export_users,import_users,update_user_posts,update_user_
↪assigned,email_outstanding_tickets,runserver,shell}
                ...

positional arguments:
  {db,run_set_up,export_users,import_users,update_user_posts,update_user_assigned,email_
↪outstanding_tickets,runserver,shell}
  db                    Perform database migrations
  run_set_up
  export_users          Command used by manage.py to export all the users from
                        the database to a json file. Useful if we need a list
                        of users to import into other applications.
  import_users          Command used by manage.py to import users from a json
                        file formatted such: [ { username, name, email,
                        password. }
  update_user_posts     Command used by manage.py to update the users total
                        post count. Use when upgrading from 0.1.4.
  update_user_assigned  Command used by manage.py to update the users total
                        post count. Use if upgrading to 0.1.7.
  email_outstanding_tickets
                        Script to be run independently of the webserver.
                        Script emails users a list of outstanding tickets that
                        they have created or been assigned. To be run on a
                        regular basis using a cron job or similar. Email
```

(continues on next page)

(continued from previous page)

runserver	functionality has to be enabled.
shell	Runs the Flask development server i.e. <code>app.run()</code>
	Runs a Python shell inside Flask application context.
optional arguments:	
-?, --help	show this help message and exit

1.3.2 Administration Config Panel

Options

For email configuration the following options are available. At a minimum you should configure *mail_server*, *mail_port*, *mail_username* and *mail_password*.

For more information regarding these settings see the documentation for Flask-Mail.

class flicket_admin.models.flicket_config.FlicketConfig(*args: Any, **kwargs: Any)

Server configuration settings editable by administrators only via the administration page `/flicket_admin/config/`.

For email configuration settings see <https://flask-mail.readthedocs.io/en/latest/> for more information.

Parameters

- **mail_server** (str) – example: *smtp.yourcompany.com*.
- **mail_port** (int) – example: *567*
- **mail_use_tls** (bool) – example: *true*
- **mail_use_ssl** (bool) – example: *false*
- **mail_debug** (bool) – example: *false*
- **mail_username** (str) – example: *flicket.admin*
- **mail_password** (str) –
- **mail_default_sender** (str) – example: *flicket.admin@yourcompany.com*
- **mail_max_emails** (int) –
- **mail_suppress_send** (bool) –
- **mail_ascii_attachments** (bool) –
- **application_title** (str) – Changes the default banner text from *Flicket*. Can typically be your company name.
- **posts_per_page** (str) – Maximum number of posts / topics displayed per page.
- **allowed_extensions** (str) – A comma delimited list of file extensions users are allowed to upload. DO NOT include the . before the extension letter.
- **ticket_upload_folder** (str) – The folder used for file uploads.
- **base_url** (str) – The sites base url. This is used to resolve urls for emails and links. Broken links are probably a result of not setting this value.
- **csv_dump_limit** (str) – The maximum number of rows exported to csv.
- **change_category** (bool) – Enable/disable change category.

- **change_category_only_admin_or_super_user** (*bool*) – Only admins or super users can change category.

static extension_allowed(*filename*)

Validates extension of a given filename and returns True if valid. Otherwise False. :param filename: :return:

static valid_extensions()

Returns a list of valid extensions. :return: list()

1.4 Exporting / Importing Flicket Users

1.4.1 Exporting

If you need to export the users from the Flicket database you can run the following command:

```
python manage.py export_users
```

This will output a json file formatted thus:

```
[
  {
    'username': 'jblogs',
    'name': 'Joe Blogs',
    'email': 'jblogs@email.com',
    'password': 'bcrypt_encoded_string'
  }
]
```

1.4.2 Importing

If you need to import users run the following command:

```
python manage.py import_users
```

The file has to be formatted as shown in the Exporting example.

1.5 Installing A Webserver

Currently the documentation will only describe how to install and configure the Apache webserver on Windows since this can be a bit trickier than on Linux. However, some of the steps here can also be used in Linux.

The instructions provided are for use with Python and Apache. You must ensure both Python and Apache have been compiled with the same version of Visual Studio. Also, Python and Apache must both be compiled for the same CPU architecture (x86 x64).

Also, the paths defined in this guide can be changed. You can by all means use different paths but you should try and get the webserver running with the settings defined herein first.

1.5.1 Apache - Windows

Prior to installing a webserver you should confirm that flicket is working correctly by running the development web-server as described in the [Installation](#) instructions.

Install mod_wsgi

Download the applicable *mod_wsgi* whl for your flavour of Apache and Python from the [Unofficial Windows Binaries for Python Extension Packages](#) page. For example, if you have *Python 3.6 x64* and *Apache 2.4 x64* you would get the whl *mod_wsgi-4.6.5+ap24vc14-cp36-cp36m-win_amd64.whl*.

Whilst **active in your flicket virtual environment** install *mod_wsgi*:

```
pip install <path_to_download>mod_wsgi-4.6.5+ap24vc14-cp36-cp36m-win_amd64.whl
```

Installing Apache

Download Apache compiled with VC14 from the [apache lounge](#).

Unzip the apache folder to your *c:* directory. You should end up with a folder structure like this:

```
C:\Apache24
  C:\Apache24\bin
  C:\Apache24\cgi-bin
  ...
```

Open the file *C:\Apache24\conf\httpd.conf* in a text editor like [notepad++](#).

Ensure *SRVROOT* is pointing to the correct folder:

```
SRVROOT "C:\Apache24"
```

Uncomment *mod_version* line:

```
LoadModule version_module modules/mod_version.so
```

Add the following lines (put these after the other *LoadModule* declarations):

```
LoadModule wsgi_module "<path_to_your_virtualenv>/lib/site-packages/mod_wsgi/server/mod_
↪wsgi.cp36-win_amd64.pyd"
WSGIProxyHome "<path_to_your_virtualenv>"
```

Uncomment the *vhosts* line:

```
Include conf/extra/httpd-vhosts.conf
```

Edit the file *C:\Apache24\conf\extra\httpd-vhosts.conf*.

Comment out the existing configurations lines by prefixing with a *#* (good reference for future troubleshooting).

Add the following:

```
<VirtualHost *:8000>

    ServerName <ip_address or hostname>
```

(continues on next page)

(continued from previous page)

```

ServerAlias <ip_address or hostname>
ServerAdmin <your_email@there.com>

DocumentRoot C:\Apache24\htdocs

<Directory C:\Apache24\htdocs>
<IfVersion < 2.4>
    Order allow,deny
    Allow from all
</IfVersion>
<IfVersion >= 2.4>
    Require all granted
</IfVersion>
</Directory>

WSGIScriptAlias / <path_to_flicket>run.wsgi
# Make sure to enable so that the API requests will work.
WSGIPassAuthorization On

<Directory <path_to_flicket>>
<IfVersion < 2.4>
    Order allow,deny
    Allow from all
</IfVersion>
<IfVersion >= 2.4>
    Require all granted
</IfVersion>
</Directory>

</VirtualHost>

```

Edit the file *run.wsgi* so that the path points to your Flicket virtual environment.

Register Apache As A Service

Navigate to the Apache folder and register the service with name *Apache HTTP Server*:

```

cd "C:\Apache24\bin"
httpd.exe -k install -n "Apache HTTP Server"

```

Start Apache

To start the service use either Windows Service Manage and start the service *Apache HTTP Server* or from the command prompt whilst in the folder *c:\Apache24\bin*:

```

httpd -k start -n "Apache HTTP Server"

```

Flicket should now be available in your browser by accessing `http:<ip_address or hostname>:8000`

Troubleshooting

To troubleshoot problems starting the apache service or accessing the webpage you should start by reading your Apache installations log files normally located in *c:Apache24logs*.

1.6 Adding Additional Languages

Flicket now supports additional languages through the use of Flask Babel. To add an additional local:

- Edit *SUPPORTED_LANGUAGES* in *config.py* and add an additional entry to the dictionary. For example: *{'en': 'English', 'fr': 'Francais', 'de': 'German'}*
- Whilst in the project root directory you now need to initialise the new language to generate a template file for it.

```
pybabel init -i messages.pot -d application/translations -l de
```

- In the folder *application/translations* there should now be a new folder *de*.
- Edit the file *messages.po* in that folder. For example:

```
msgid "403 Error - Forbidden"
msgstr "403 Error - Verboten"
```

- Compile the translations for use:

```
pybabel compile -d application/translations
```

- If any python or html text strings have been newly tagged for translation run:

```
pybabel extract -F babel.cfg -o messages.pot .
```

- To get the new translations added to the .po files:

```
pybabel update -i messages.pot -d application/translations
```

1.7 Flicket - FAQ

1.7.1 What is Flicket?

Flicket is a simple open source ticketing system driven by the python flask web micro framework.

Flicket also uses the following python packages:

alembic, bcrypt, flask-admin, flask-babel, flask-login, flask-migrate, flask-principal, flask-sqlalchemy,
flask-script, flask-wtf, jinja2, Markdown, WTForms

See *README.rst* for full requirements.

Licensing

For licensing see *LICENSE.md*

1.7.2 Tickets

General

1. How do I create a ticket?

Select 'create ticket' from the Flicket pull down menu.

2. How do I assign a ticket?

Scenario: You have raised a ticket and you know to whom the ticket should be assigned.

Navigate to [flicket home page](/flicket/) and select the ticket you wish to assign. Within the ticket page is a button to *assign* ticket.

3. How do I release a ticket?

Scenario: You have been assigned a ticket but the ticket isn't your responsibility to complete or you are unable to for another reason.

Navigate to [flicket home page](/flicket/) and select the ticket to which you have been assigned. Within the ticket page is a button to *release* the ticket from your ticket list.

4. How do I close a ticket?

Scenario: The ticket has been resolved to your satisfaction and you want to close the ticket.

Navigate to [flicket home page](/flicket/) and select the ticket which you would like to close. Within the ticket page is a button to *replay and close* the ticket.

Only the following persons can close a ticket: * Administrators. * The user which has been assigned the ticket.
* The original creator of the ticket.

You may *claim* the ticket so that you may close it.

5. What is markdown?

Markdown is a lightweight markup language with plain text formatting syntax.

The text contents of a ticket can be made easier to read by employing markdown syntax.

Searching

The ticket main page can be filtered to show only results of a specific interest to you. Tickets can be filtered by department, category, user and a text string.

Departments

Note: Only administrators or super users can add / edit or delete departments.

1. How do I add new departments?

Navigate to Departments via the menu bar and use the add departments form.

2. How do I edit departments?

Navigate to [departments](/flicket/departments/) and select the edit link against the department name.

3. How do I delete departments?

Navigate to [departments](/flicket/departments/) and select the remove link against the department name. This is represented with a cross.

Categories

Note: Only administrators or super users can add / edit or delete categories.

1. How do I add categories?

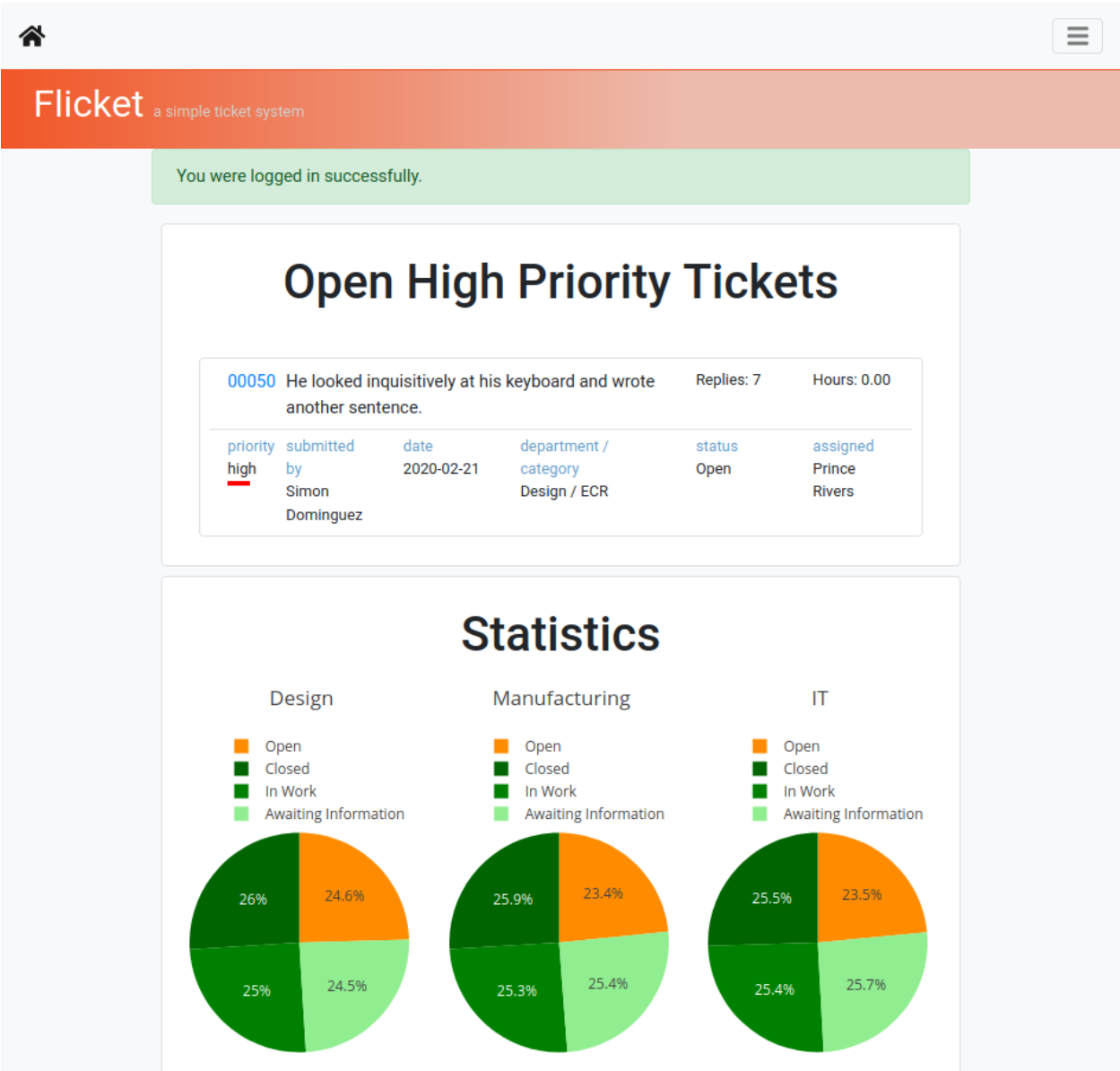
Navigate to [departments](/flicket/departments/) and select the link to add categories against the appropriate department name.

1. How do I edit categories?

Navigate to [departments](/flicket/departments/) and select the link to add categories against the appropriate department name.

1.8 Screenshots

1.8.1 Home Page



1.8.2 Tickets

All tickets.

[Home](#)
[My Tickets](#)
[Tickets](#)
[Create Ticket](#)
[Departments](#)
[Users](#)
[Admin View](#)
admin

Flicket a simple ticket system

Tickets


Download results as a csv file [📄](#).

--departments--
--categories--
--status--
username
contents
[search](#)
[reset](#)

<<
1
2
3
4
5
...
148
149
>>
Sort

00001	It is also a garbage-collected runtime system.				Replies: 14	Hours: 0.00
priority	submitted by	created	updated	department / category	status	assigned
high	Serita Kane	2020-02-21	2016-11-21	Manufacturing / Equipment	In Work	Albert Sanford
00002	The syntax {D1,D2,...,Dn} denotes a tuple whose arguments are D1, D2, ... Dn.				Replies: 11	Hours: 0.00
priority	submitted by	created	updated	department / category	status	assigned
low	Emmy Koch	2020-02-21	2016-11-21	IT / Intranet	Awaiting Information	Lane Banks
00003	Erlang is known for its designs that are well suited for systems.				Replies: 2	Hours: 0.00
priority	submitted by	created	updated	department / category	status	assigned
high	Delmar Cochran	2020-02-21	2016-11-21	Commercial / Approved Suppliers	In Work	Gwyneth Frye
00004	Atoms are used within a program to denote distinguished values.				Replies: 7	Hours: 0.00
priority	submitted by	created	updated	department / category	status	assigned
low	Lonny Avery	2020-02-21	2016-11-21	IT / Intranet	In Work	Janeth Lawson
00005	It is also a garbage-collected runtime system.				Replies: 12	Hours: 0.00

1.8.3 View Ticket

 My Tickets ▾ Tickets Create Ticket Departments Users Admin View

admin ▾

Flicket

a simple ticket system

00001

It is also a garbage-collected runtime system.

claim

release

assign

Department

Category

Status

Priority

Assigned

Total Hours

Manufacturing

Equipment

In Work

high

Albert Sanford

0.00


Subscribed:

User Name



subscribe user

<< 1 >>

Serita Kane



21-02-2020 16:39




Atoms are used within a program to denote distinguished values. The arguments can be primitive data types or compound data types. Haskell features a type system with type inference and lazy evaluation. Ports are created with the built-in function open_port. The sequential subset of Erlang supports eager evaluation, single assignment, and dynamic typing.



Hours: 0.00

quote

Edris Snyder



Reply #1 | 21-02-2020 16:39




She spent her earliest years reading classic literature, and writing poetry. He looked inquisitively at his keyboard and wrote another sentence. Tuples are containers for a fixed number of Erlang data types. He looked inquisitively at his keyboard and wrote another sentence. Ports are created with the built-in function open_port.



Hours: 0.00

quote

Mac Weaver



Reply #2 | 21-02-2020 16:39



Type classes first appeared in the Haskell programming language. Erlang is known for its designs that are well suited for systems. Do you come here often? Haskell is a standardized, general-purpose purely functional programming language, with non-strict semantics and strong static typing. Haskell is a standardized, general-purpose purely functional programming language, with non-strict semantics and strong static typing.

1.8.4 Create Ticket

Flicket - Create Ticket - Mozilla Firefox

Flicket - Create Ticket | Samanage Bot says... | +

https://flicket.evereux.uk/ticket_create/

My Tickets | Tickets | Create Ticket | Departments | Users | FAQ | Admin View | admin

Flicket

a simple ticket system

Flicket - Create Ticket

Ticket contents supports markdown syntax. Please refer to [md help](#).

Ticket Title

Content

Please help.

I can't log in. Computer says

invalid username or password

Please help.

I can't log in. Computer says

invalid username or password

[MarkDown reference](#)

Show / Hide Markdown

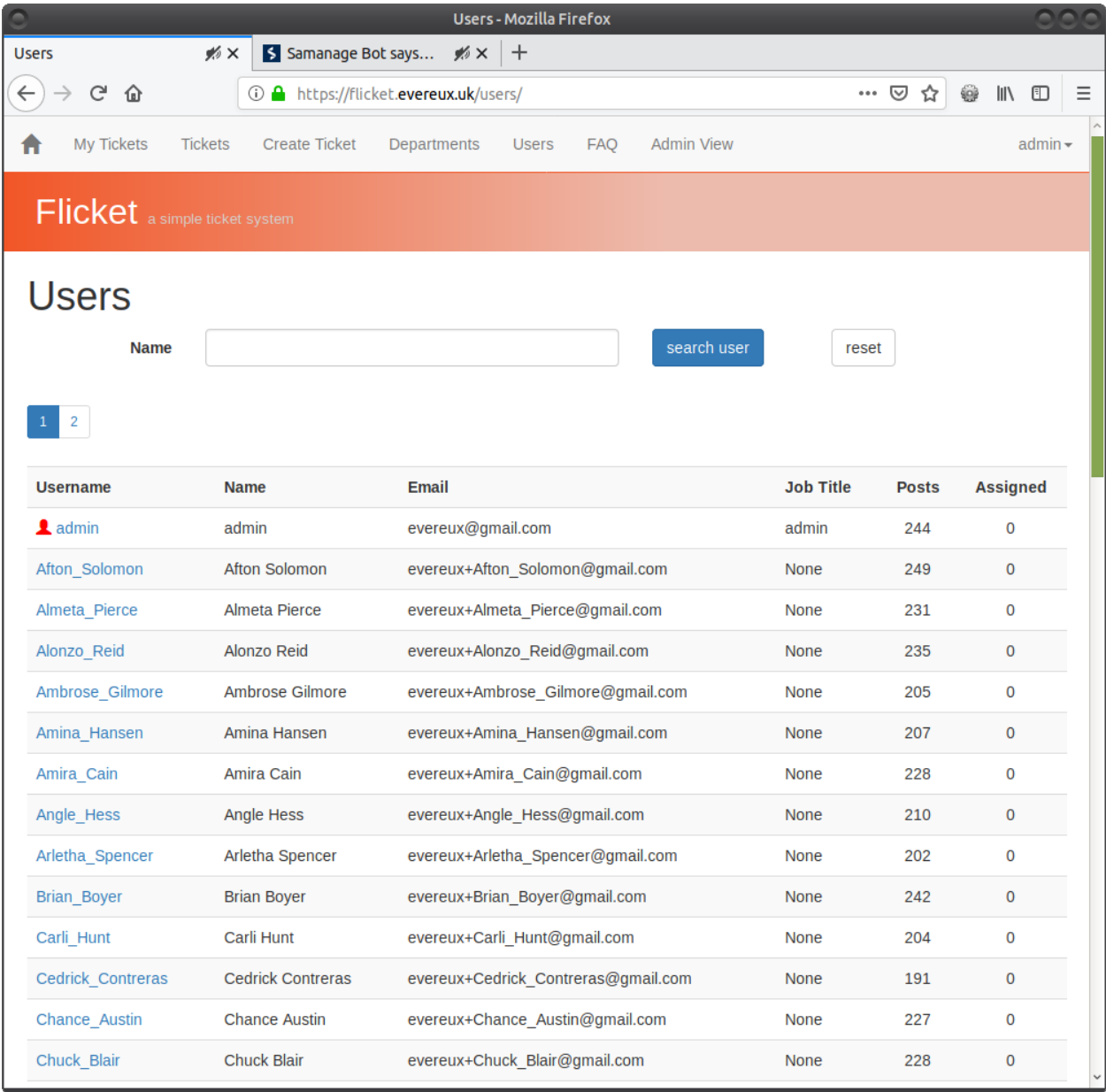
Priority Level

Category


Upload Documents No files selected.

Flicket 0.1.8 © 2018 [Paul Bourne](#) | Source code available at: [Github](#) | This site uses: [Glyphicons](#).

1.8.5 Users



1.8.6 Admin Panel

 [My Tickets](#) [Tickets](#) [Create Ticket](#) [Departments](#) [Users](#) [FAQ](#) [Admin View](#) admin ▾

Flicket a simple ticket system

Administration


[Admin Home](#) [Flicket Config](#) [Users](#) [Add User](#) [Groups](#)

Admin

Administration home page.

Flicket 0.1.8 © 2018 [Paul Bourne](#) | Source code available at: [Github](#) | This site uses: [Glyphicons](#).

1.8.7 Admin Panel - Add User

 [My Tickets](#) [Tickets](#) [Create Ticket](#) [Departments](#) [Users](#) [FAQ](#) [Admin View](#) admin ▾

Flicket a simple ticket system

Administration

[Admin Home](#) [Flicket Config](#) [Users](#) [Add User](#) [Groups](#)

Add User

Username

Name

Email

Job Title

Locale

 ▾

Password

Confirm

[add_user](#)

Flicket 0.1.8 © 2018 [Paul Bourne](#) | Source code available at: [Github](#) | This site uses: [Glyphicons](#).

1.8.8 Admin Panel - Configuration

My Tickets

Tickets

Create Ticket

Departments

Users

FAQ

Admin View

admin ▾

Flicket

a simple ticket system

Administration - Configuration

Admin Home

Flicket Config

Users

Add User

Groups

Field	Value
mail_server	<input type="text"/>
mail_port	<input type="text"/>
mail_use_tls	<input type="checkbox"/>
mail_use_ssl	<input type="checkbox"/>
mail_debug	<input checked="" type="checkbox"/>
mail_username	<input type="text" value="admin"/>
mail_password	<input type="password" value="....."/>
mail_default_sender	<input type="text"/>
mail_max_emails	<input type="text"/>
mail_suppress_send	<input checked="" type="checkbox"/>
mail_ascii_attachments	<input type="checkbox"/>
posts_per_page	<input type="text" value="50"/>
allowed_extensions	<input type="text" value="txt, pdf, png, jpg, jpeg"/> This must be a comma delimited list.
default_upload_folder	<input type="text"/>

1.9 API

PYTHON MODULE INDEX

f

flicker_admin, 5

INDEX

E

`extension_allowed()`
(*flicket_admin.models.flicket_config.FlicketConfig*
static method), 7

F

`flicket_admin`
module, 5
`FlicketConfig` (class in
flicket_admin.models.flicket_config), 6

M

module
`flicket_admin`, 5

V

`valid_extensions()` (*flicket_admin.models.flicket_config.FlicketConfig*
static method), 7