Twatter 2.0

Documentation

Introduction

The idea of this project is to create a Twitter like system where you can tell the world what is wrong. For example did is something wrong with your workplace, but you can't talk it in the open? Did you buy an Apple only notice it is rotten and missing something? So in conclusion, you can pour your heart less anonymously (under a nickname) out on Twatter.

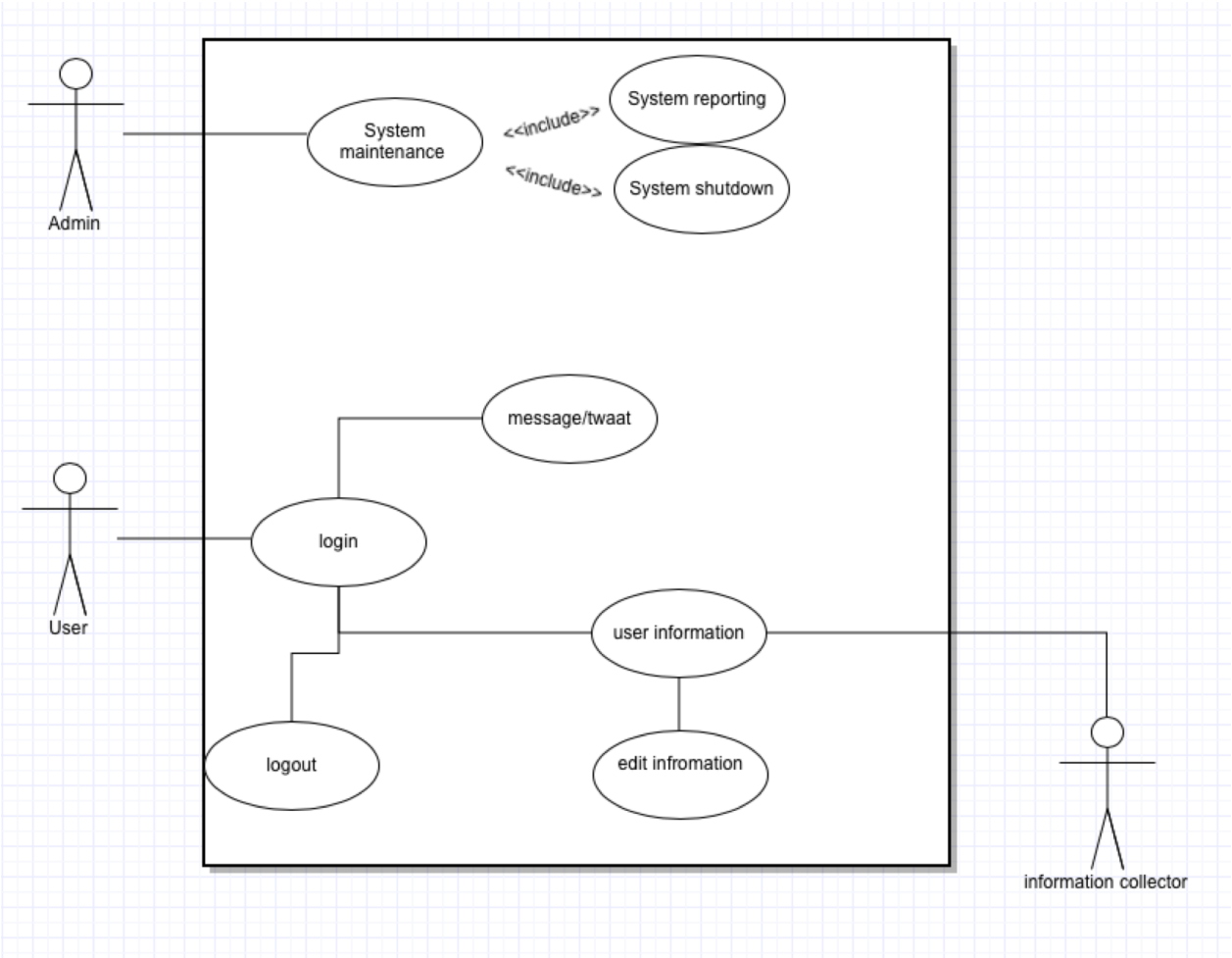
The goal of the system is to help people to relieve some stress and pour their hearts out to the world. It's quite a simple idea, but it is still not implemented to the world of web.

The project will be released on an VPS running in Germany with quite low specs. It will be accessible under http://valit.us/ and the local part of the project will be done on a Macbook and OSX environment. The project is not created to support or to be run on Windows computers, but accessing http://valit.us/ is destined to work everywhere, mobile or desktop - no matter.

Backend running on Python and framework Flask and SQL database will run in PostgreSQL. All dependencies will be added in requirements.txt and can be installed through $p*ip install -r requirements.txt*.

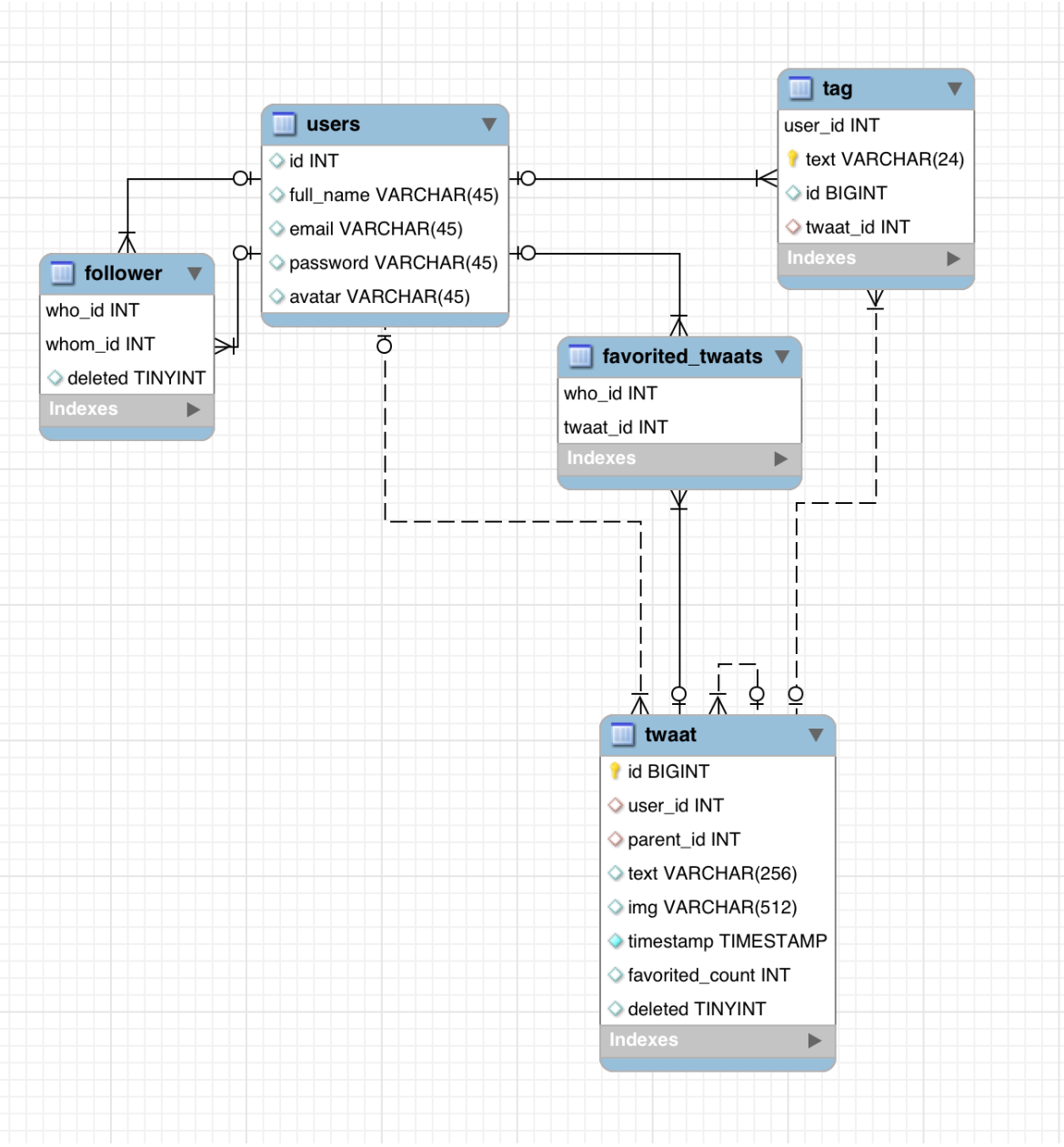
Browser is not required to have support for javascript, as this project aims to almost 0% javascript. Some evenlisteners/button click handlers are made in JS.

Use cases



Use case is quite simple: There is a user who can login/logout and post a message or read them. User can follow other users, which then gives their TWAATs on his/hers feed. User can ‘love’ single TWAATs which are then displayed on his/hers loved page, user can also unlove from the page. User can edit information from settings and remove followed from their profiles.

SQL Schema



Current status of the SQL schema. No changes will be made here. Relations can be seen with arrows

Tables and attributes

Most important table is users with the attribute id. All other attributes are not required by the DB. User password and email are used to login. Avatar to display user image as relative path.

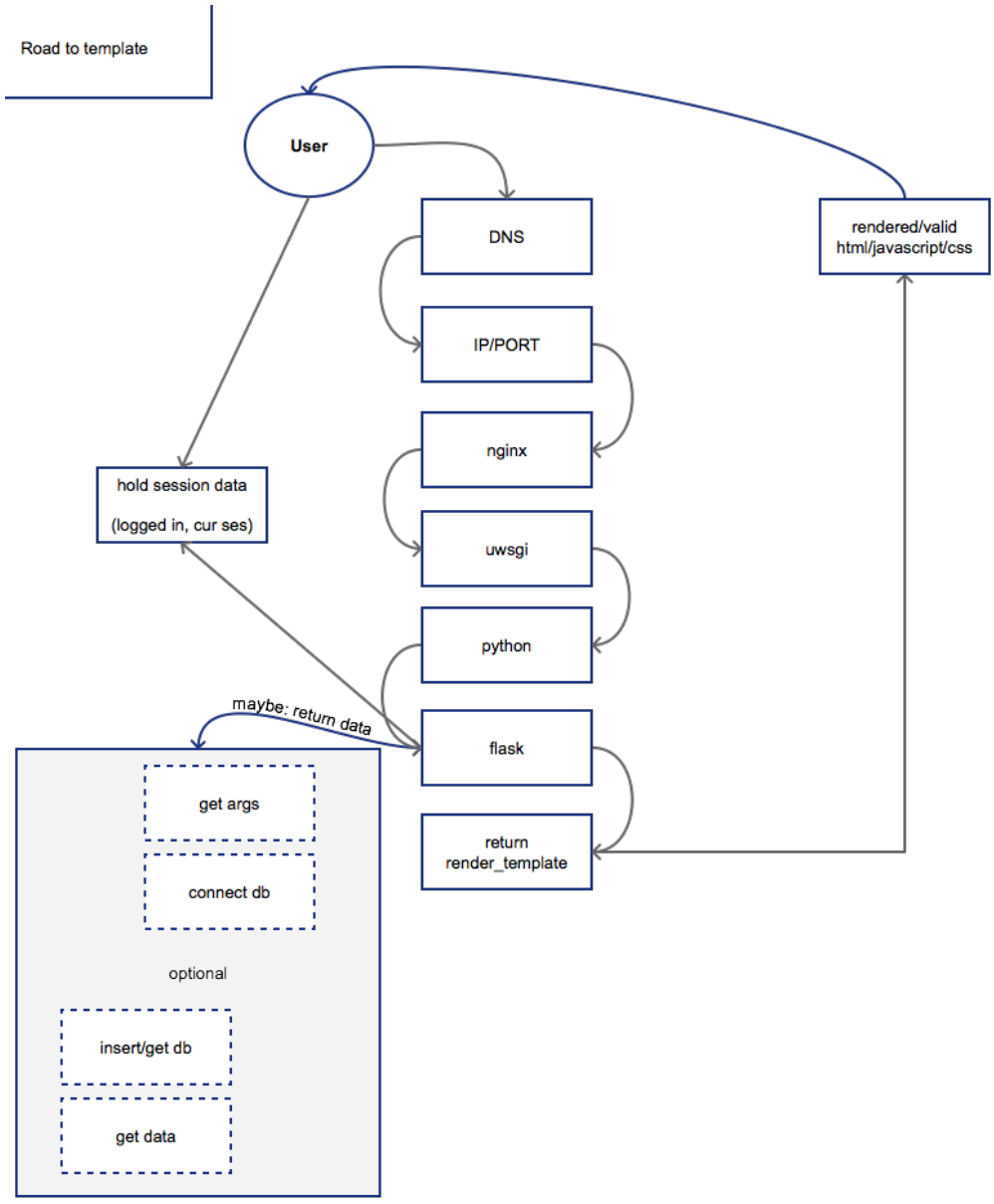
Follower consists of two users, who is following and whom is followed.

Tag is a “hashtag” ex. #hello. It is used to create user groups and follow certain tags. Currently this feature is not active.

Twaat is another main table, with id as most important attribute. user_id is a ref to users table and text/img/timestamp are secondary attrs.

Favorited_twaats is similar to follower, has twaat_id and user_id.

Road to a view



Roadmap

Note: Only public page is index

index
- -> login
  - ->frontpage
- -> register
  - -> index

login
- -> frontpage
  - ->profile
  - ->loved
  - ->settings

frontpage
- -> post_twaat

profile
- -> if not own: follow/unfollow
- -> per twaat: love

loved
- -> per twaat: delete love

settings
- -> change settings /submit

Installing

To install this project you need to have python and pip and postgres.

1. $ pip install -r requirements.txt
2. $ psql -d <yourdbname> < sql/create_sql.sql
3. setup config.py and move it to settings_local.py
4. $ ./dev

Other & thoughts

Bugs et cetera

There are some known bugs, as the server environment for some reason does not exactly correspond the development environment which causes problems. Ex. in relative paths and python imports. Everything is working in development, so if you are looking at http://valit.us and something is not working **please run it in localhost to verify that it is not working.**

Most of the time in the project was used to fiddle with VPS and getting the pipeline to work. After the version 2.0 refactor most of the bugs were solved and the pipelining works very well too.

Still not sure why import .. from foo is required in localhost and import foo is okay in release.

Thoughts

Even though the course material was heavily based on PHP (wtf?) I still managed to learn something useful for me, ex. fielding around with python, setting up a conventional GIT pipeline to deploy wanted branch, setting up UWSGI + NGINX + PYTHON + FLASK etc.

As of upcoming features, there are none as the course killed my motivation for the project :)
But otherwise, if the motivation pops up ever, adding things are easy as the project is mainly done with the 'modular' way of doing things.

One thing is to implement the tag feature where users can follow and read posts submitted to 'tags'.

Final words
It was a nice course and project. I choose this kind of project because I have never really used any social medias and had no idea how to use one or how they look from inside. So after creating the layout and functionality, on the last week I signup on Twitter, only to note my dreams and fantasies were not so far away from the truth.

VPS

Keeping VPS up to date

Note: most of the commands require **super user rights (sudo)**
SSH to your VPS.
Creating a git pipeline to push to :

```
$ cd /var
$ mkdir repo && cd repo
$ mkdir twatter.git && cd twatter.git
$ git init -- bare  # this creates just the version control and no ex-
tra files
```

Creating a hook

For more information about hooks, see git guide: Customizing-Git-Git-Hooks
We are using post receive:

```
$ cd hooks
$ touch post-receive
$ nano post receive
#!/bin/sh
git --work-tree=/var/www/valitus --git-dir=/var/repo/twatter.git check-
out -f
```

Note that my 'public dir' is 'valitus' and my repository is 'twatter'.

Exit VPS now.

Configuring repository

```
$ cd ~/respos/twatter
$ git remote add live ssh://mxo@mxo.ninja/var/repos/twatter.git
$ git add .
$ git commit -m "lets go live"
$ git push origin live
```

And now, we can see our changes live :)

Remember to check that the /var/ folders are owned by you! I had minor problem as my /var/repos/ was owned by 'root' and I had no access pushing it:

```
remote: fatal: Unable to create temporary file '/var/repos/twatter.git/
objects/pack/tmp_pack_XXXXXX': Permission denied
```

This was fixed by `chown mxo /var/repos/ -R` where -R is recursive (subfolders etc)