Carter Williams

9/25/22

Dr. Zhonghang

**CS 396 Phase 1**

**Project Description**

Phase 1 of the CS 396 term project includes building a website that provides services of forum, documentation, searching, linking, and others. This project was implemented with the Python web framework Django along with SQLite as a database. Users can sign up for the forum with a username and password and then can post under discussion topics. They can create a title, provide textual content, and upload files to these posts. Other users can then comment under these posts. Only admins of the website can create discussion topics through the ./admin/ web interface. Admins can also adjust/delete posts, topics, users, and anything else stored in the database. Admins can be created by another admin through the ./admin web interface or by passing the “createsuperuser” argument to manage.py.

The front end of this project was implemented using Django templates. Templates allow for webpages to be created using HTML, CSS, and JavaScript, but it also adds some functionality with Python code inside HTML files. This will be explained further in the Technical Details section.

Anybody can use the codebase for their own purpose by following the ReadMe inside the project’s files.

**Functions Implemented**

Many functions have been implemented in order to make the forum website perform various actions. Most of these functions are inside the ./fintech/views.py file, but some of Django’s models also have overridden methods. These functions will be split into sections based on which file they are implement

./fintech/views.py

1. def login\_(request)

The function login\_ handles the url ./login/ and this is where users get authentication. The template login.html provides the UI for users to interact with this function. If the user is already authenticated, the user is redirected to the home page. If the user is not authenticated, the inputted username and password are retrieved through request.POST.get, and Django’s authenticate method (django.contrib.auth) authenticates the user. If it is a valid login, the user is logged in and redirected to the homepage, but if they are not, a message is displayed that the username and password are not found.

1. def register(request)

The function register handles the url ./register/ and this is where users go to add a new user in the database. The template register.html provides the UI for the users to interact with this function. If the user is already authenticated, the user is redirected to the home page. If they are not authenticated, the fields username, password, and password confirm are retrieved through request.POST.get. The password and password confirm are checked for equality, and if they match, a new user is created through the User.objects.create\_user function (django.contrib.auth.models). The user is then redirected to the login page in order to login to their new user.

1. def logout\_user(request)

The function logout\_user handles the url ./logout/ and simply logs the user out of their account. This is accomplished through the logout function (django.contrib.auth). The user is then redirected to the login page.

1. def index(request)

The function index handles the base url and displays some initial information to the user. Boards, topics, someposts, and the total number of posts are gathered from the database. All of this is then passed to index.html through the context variable. Index then displays the board name, the total number of posts on the website, the topics, and two posts from each topic. Users can navigate to new topics or posts by clicking their titles.

1. def view\_topic(request, topic\_id)

View\_topic is the function that handles the url ./topic/<int:topic\_id>/, and it displays all relevant information to the topic. The topic and posts under the topic are retrieved from the database and passed to topic.html to be displayed to the user. The posts can then be clicked on by the user in order to view the individual post

1. def view\_post(request, topic\_id, post\_id)
2. def create\_post(request, topic\_id)
3. def edit\_post(request, topic\_id, post\_id)
4. def delete\_post(request, topic\_id, post\_id)
5. def post\_comment(request, topic\_id, post\_id)
6. def display\_images(request, topic\_id, post\_id, image\_name)
7. def authenticate\_for\_update(request, user)
8. def return\_post\_files(user, post)
9. def handle\_uploaded\_file(f, topic\_id, post\_id, user\_id)
10. def get\_notifications()

./fintech/models.py