ECE1779 Assignment 1

# Description

This application is written in python and html using the flask library and it performs as an online text detector. It automatically draws rectangular boxes around the text on the photos which users uploaded. Users can explore all the photos they uploaded and compare the photos before and after text detection in pairs.

# Prerequisite

An Python 3.7 interpreter is required.

The following python packages are required in order to start the server.

* Click
* PyMySQL
* Flask
* Flask-SQLAlchemy
* Jinja2
* MarkupSafe
* SQLAlchemy
* Werkzeug
* aiofiles
* aiohttp
* async-timeout
* attrs
* chardet
* idna
* imutiles
* itsdangerous
* multidict
* opencv-python
* pip
* setuptools
* yarl

# How to use the application

## Initialize the web application

To initialize the web application, run the start.sh at Desktop. The web application should run at port 5000. So, make sure port 5000 is available.

## Using the application

If users are not log in, the nav bar will have log in button as well as sign up button. The user can use log in button to log in. Also, user can use sign up button to sign up.

While user has logged in, nav bar has home, log out, and upload button. Log out button will lot out account. Home button is directed to home page at which user can view and upload the images. At upload page, user can upload ‘png’, ‘jpg’, ‘jpeg’ and ‘fig’ type images. After uploading, web app will detect and use green rectangle mark up the text in the images. User can view their uploaded images at view page. By click ‘view’ button under the images, users are able to view the original image and processed image at same time.

# API

Tests can be done by sending API requests. You are allowed to create new users and upload images using an existing account.

The register request requires the following interface.

URL: <http://f91.ca:5000/api/register>

local URL: <http://0.0.0.0:5000/api/register>

method = POST

POST parameter: name = username, type = string

POST parameter: name = password, type = string

The upload request requires the following interface.

URL: <http://f91.ca:5000/api/upload>

local URL: <http://0.0.0.0:5000/api/upload>

enctype = multipart/form-data

method = POST

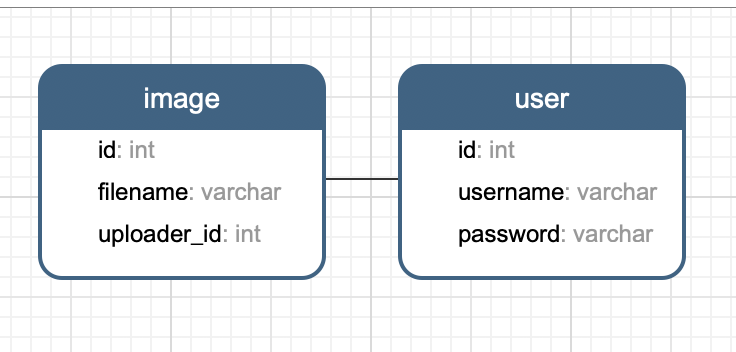
POST parameter: name = username, type = string

POST parameter: name = password, type = string

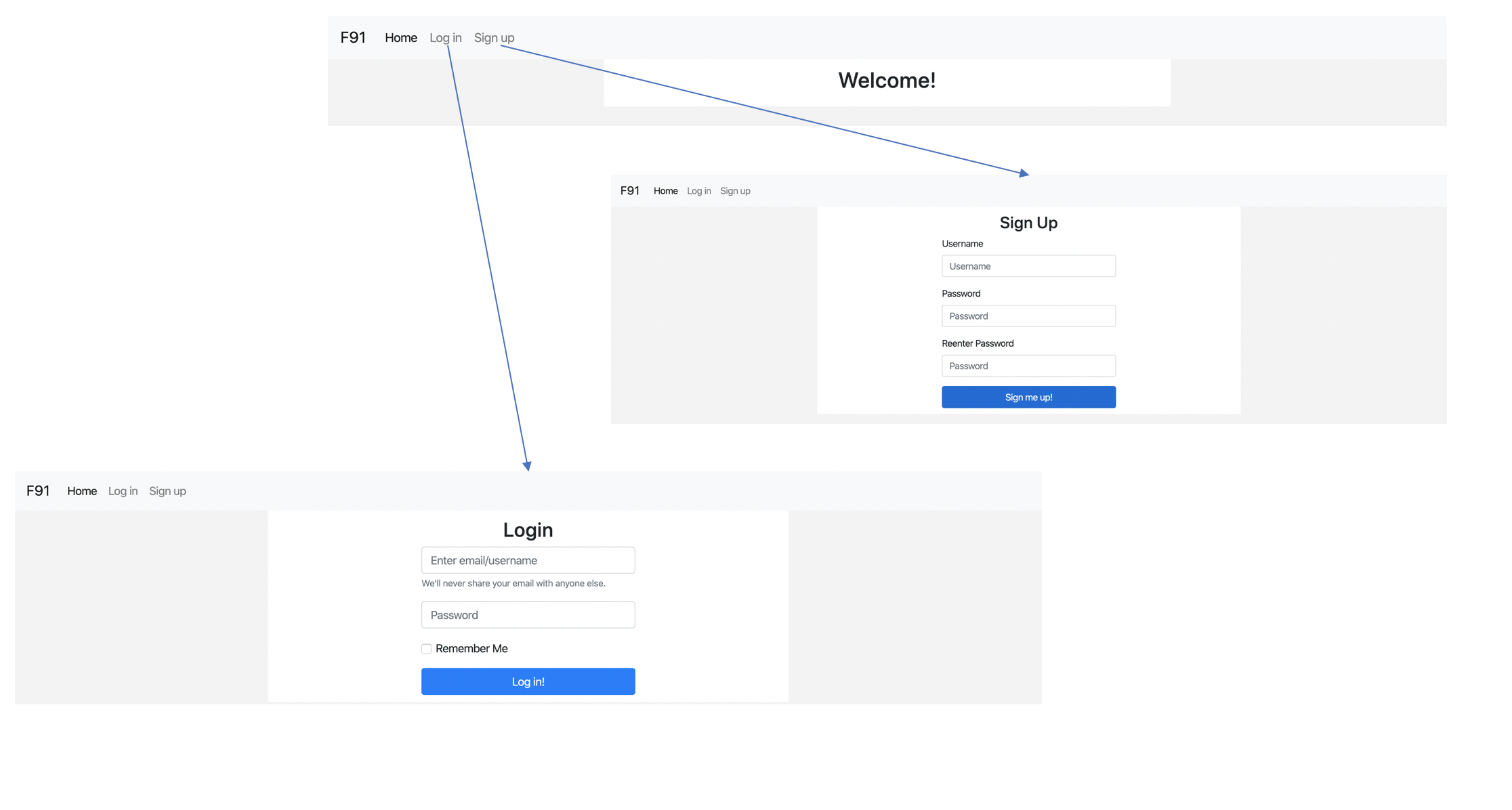
POST parameter: name = file, type = file

Web Architecture

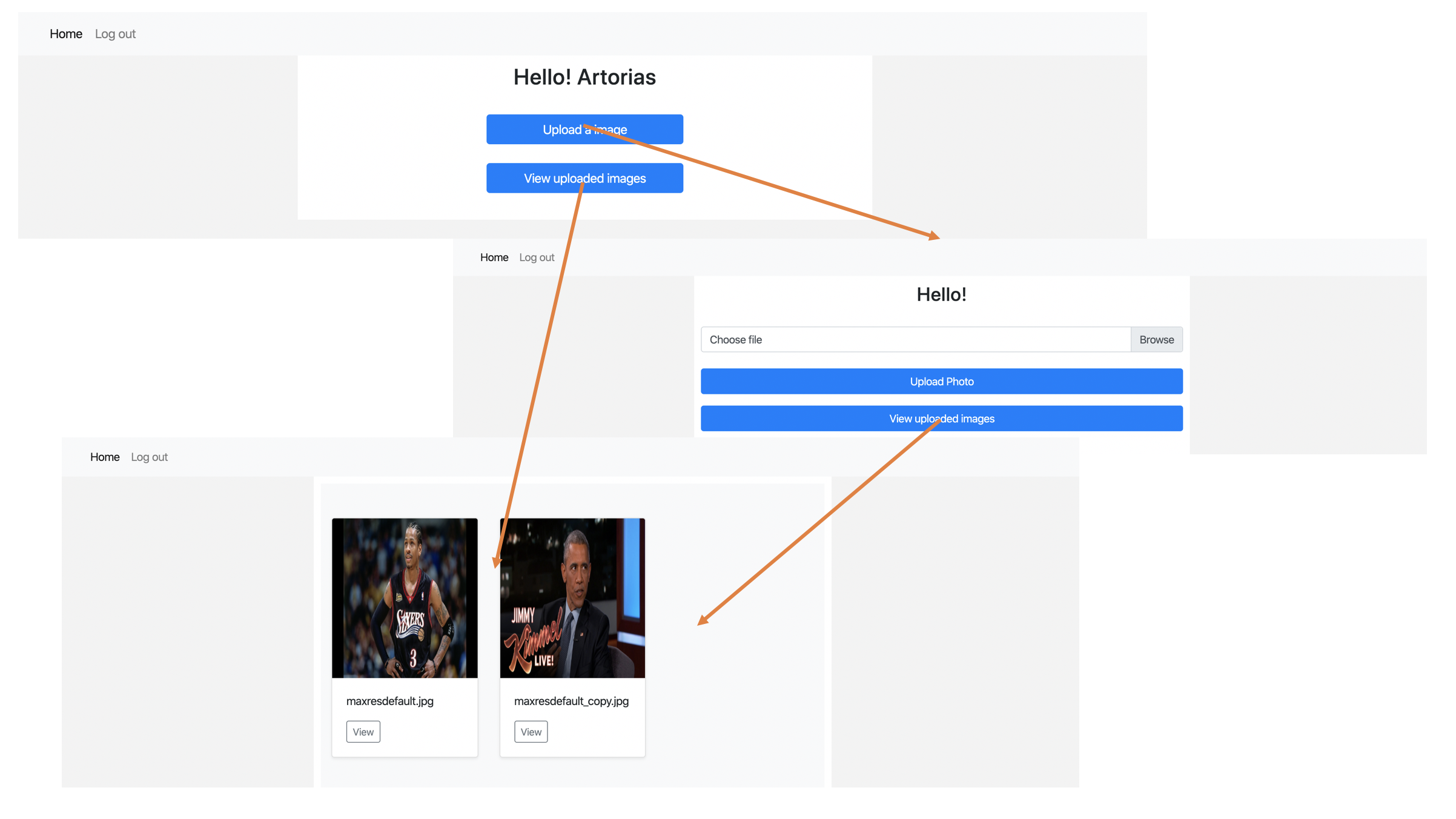
The database consists of two tables, image and user, which are connected to each other by sharing the user id. Hence, each user is only allowed to access the image he or she uploaded. Moreover, since each image is given a unique image id, different images with the same filename will be treated as different images. The detailed database structure is shown below. The uploader\_id in table ‘image’ is refer to id in table ‘user’.



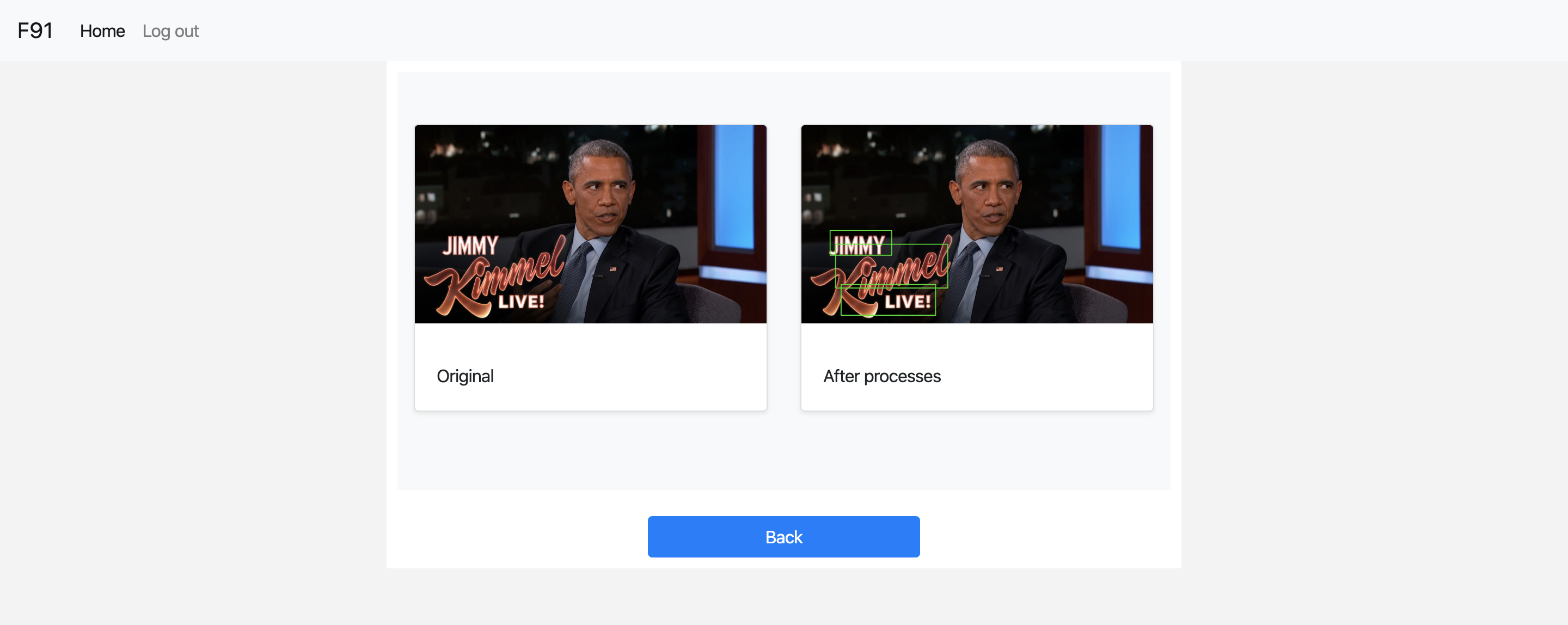
The application structure is shown as follows.



There are two buttons on the main page. The ‘Login’ button will direct the user to the login page while the ‘Sign up’ button will lead the user to the sign-up page. After logging in or creating an account, the user will be directed to the user page.



When clicking the ‘view’ button, users can see the difference between the original photos that they upload and the photos after text detection.



# Application Structure

## Code Structure

The body of the application is packed in the “app” directory. When the application starts, “\_\_init\_\_.py” will be executed and basic parameters and settings in order to run the program will be initialized. The core of the application is the “views.py” file. This file is written to handle all the requests received from the clients, such as showing users the photos they uploaded. The structure of the database is defined in “model.py”.

A static directory is created to store all the css files as well as all the photos uploaded by users. When a new account is registered, a folder with the same name of such user is created as well. All the images the user uploaded will be stored in his or her folder so that we can easily tell the uploader of an image.

“text\_detection.py” contains the function to draw rectangles around texts in the photo. When an image is uploaded, another image with rectangles around texts will also be created and saved to the user’s directory so that he or she can see the difference.

All the html templates are stored in the “template” folder under the “app” directory. This allows us to separate the html files from the python files. Moreover, one html template can be used in several different situations.

## Cookies

For security purposes, cookies are used so that we can know the status of the user. When the application starts, a *session* object is created to store the login information of the user. All the data in the *session* object is encrypted using a secret key, which is generated randomly in the initialization phase of the application. Once the user logged in, the username will be stored in *session* so for further verification. Users are required to reenter their usernames and passwords every 24 hours for the purpose of safety. They can also manually log out to drop their information in *session*. The application will automatically redirect the user to the login page if it cannot find the user’s corresponding information in *session*.

## File restriction

Users are only allowed to upload images with the extensions mentioned above. A file is verified twice before being posted on the user’s page. First, when the server receives the file, it will check the file’s extension. If the extension is not allowed, the client request will not be accepted. For instance, the server will not let a user upload a file with a pdf extension. Second, the server will check the readability of the file during image processing. If the file is unreadable, it will be rejected by the server. For example, the user can manually change the extension of a file from others to jpg, making it look an image file. However, such file cannot be read by the text detection function and hence these kinds of files are blocked from uploading.

The application only accepts files with sizes less than 10MB. Otherwise the load of the server will be too heavy.