



Facial Recognition Project

Arnas Steponavicius Aaron Moran Thomas Kenny

April 22, 2020

Project Repository: <https://github.com/Moran98/facial-recognition>

Contents

1	Introduction	2
2	System Requirements	2
3	Technology Used	2
4	Architecture of the solution	3
5	Design Methodology	3
6	Features	5
7	Limitations and Bugs	6
8	Testing Plans	6
9	Recommendation for Future Development	7
10	Conclusions	7
11	Acknowledgements	8

1 Introduction

When we formed our team there was many ideas floating around of what to design and make. We tried to think of an idea that was new and exciting for the three of us. We wanted to make design and make something that was challenging and would improve our ability as software students. The idea we settled on was Facial Recognition.

With extensive research into this field we looked at many languages to use. We thought it would be a good idea to learn a new language while working on this project. This language was Python. Python is one of the top languages used worldwide at the moment so we thought this would be beneficial for us to learn this language. We chose this language for various reasons, one of these reasons being that there is various packages python offers. We found online that there is a library called **face_recognition** that suited our needs for this project.

We had various meetings in the beginning to find the pros and cons of Facial Recognition. We had to map out the project and treat this like it was a job. This included making a plan of what we wanted to have done to present in our weekly meetings, breaking up the project into steps so we wouldn't get overwhelmed or held up on one task and discussing what each member of the team would focus on. Each task will be explained in further detail.

GitHub was vital with our project. It was important that we constantly used GitHub to display the progress we were making. We could handle issues we were having, test different ideas and we could all be on the same page as we work. We mainly tracked our developments and any bugs which had to be fixed by using the Issues section on GitHub, Once the issue was resolved we would mark it as complete. This made each member of the group aware of a bug and then we could all attempt to resolve the issue.

2 System Requirements

- Python 3.3+
- C/C++ compiler installed
- CMake <https://cmake.org/download/>
- Camera

3 Technology Used

Language:

- Python

Framework:

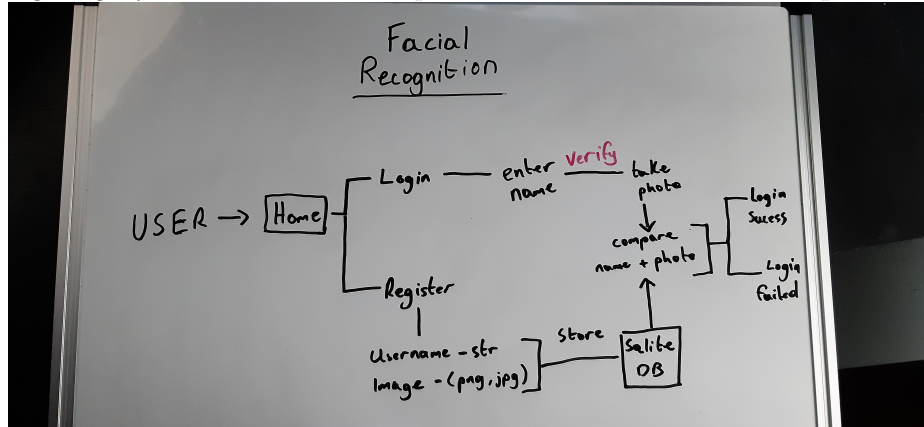
- Django

Dependencies:

- pip install face_recognition
- pip install opencv-python
- pip install numpy
- pip install pillow

4 Architecture of the solution

We designed this project to be user friendly and simplistic. From the following images you can see the UI is simple which makes for a better user experience.



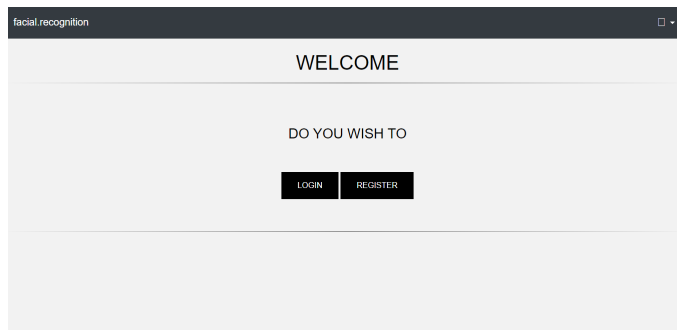
The thought we had in mind for this project was for companies to use our Register and Login software and enhance it to suit their needs. This software can be easily tailored from colour to design with customer request. The UI design is located on one CSS file. This makes for updating style simple and quick to achieve. The python and HTML files are formatted for ease of navigation. The order of the files are simple and understandable. This makes for an easier approach to edit specific files. We made sure to have good package management so any new adaptations to the project were easy to navigate to this reduced any clutter files which were not of any use to the project.

5 Design Methodology

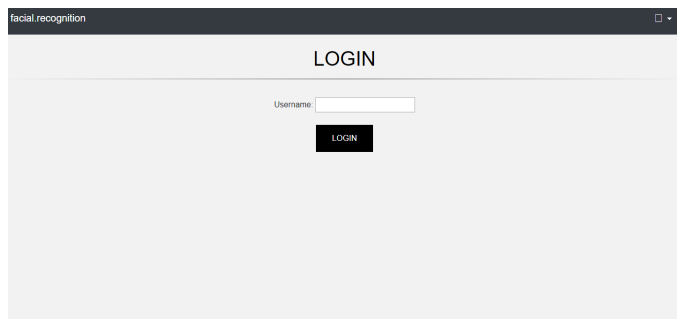
When we were deciding on how our User Interface should be implemented we drew out a few designs using wire frame. After creating multiple styles and

designs, we all agreed on the most simplistic design that does not overwhelm the user with information and was very easy to use. As you can see the design is very simple and clear. We thought about adding more to the pages like a template website but decided on "a less is more" approach to the design. Each page implements a layout.html file that stores all the style sheet's that have been used and bootstrapping for each of the web pages. Django has an organized folder system to make the project cleaner and allows easy linking between style sheet's and HTML files. The design is user friendly and self explanatory for the user in terms of instructions and usability.

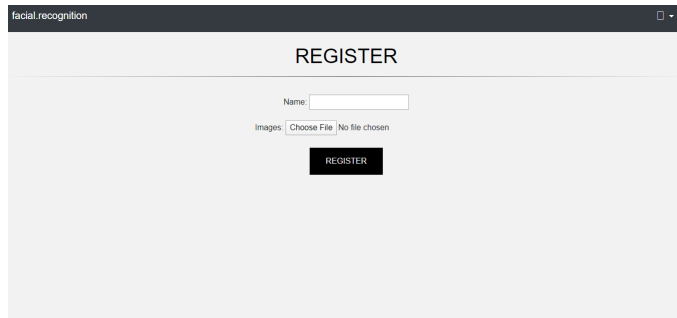
Home Screen:



Login Screen:



Register Screen:



The screenshot shows a web browser window with the title 'facial_recognition'. The main heading is 'REGISTER'. Below the heading, there is a 'Name' input field. Underneath the name field, there is an 'Images' label followed by a 'Choose File' button and the text '(No file chosen)'. At the bottom of the form, there is a black 'REGISTER' button.

6 Features

The main aim of the features implemented in this project is to provide security by using facial recognition to be able to access the inner system. Whether this software will be used for a personal use or within a company environment.

- When the user runs the program they will be greeted with the option of Login or Register. The home page is very simple, user friendly and self explanatory.
- The second feature is the register page which needs the user needs to submit their name and a clear frontal picture of themselves. These details are then saved to the database. The user must enter in details into the required input boxes or they cannot proceed in creating an account. Both options are required so no user is signing up without a registration photo.
- The main feature is the login. The user needs to enter a valid name that has been registered otherwise an error will pop up. If the name is valid the user is then brought to the verification page, where they need to take a photo of themselves. If the photo matches the image the users registered image they are let through into the site.
- Administrator access is a feature Django provides the developers with and is a very useful feature that can be used to create, update or delete a user profile within the database. This feature made it simple for each one of us to test our project and to visually see if the correct details were being stored for the user in the database.

7 Limitations and Bugs

As with any programming project we encountered a plethora of bugs and limitations whilst working on this project. The most frustrating one of all was in the initial stages of the project. We had chosen to work with the **Flask** framework and had a problem with taking an image on the web page using JavaScript, encoding the image and sending it to our database via Ajax. This problem that we had set us back for a while until we ultimately decided to swap framework to **Django**.

When we began our project with Flask we faced issues such as installations for different python packages working on specific operating systems smoothly and then not working as expected on other operating systems. We also faced problems with collaboration on the project as a new push to the repository would affect another contributors ability to develop the program as installations were not installing correctly. Thanks to Django we were able to just install the Django package and then work on developments without any issues.

Within a few days of making the switch we managed to get our images uploading to a database and then setting up the routes for the other links in our web application.

Another bug we had encountered was an Index out range error whenever an image that did not have a recognisable face on it was loaded in, but it was tackled with simple try catch statement.

For further insight into issues we faced you can view them in our GitHub repository Issues section where we often assigned issues our project was facing and solved them as a group together. This helped us to learn problem solving as a team as well as individuals.

8 Testing Plans

We went through extensive testing whilst making this project. Every commit that was uploaded to our repository was tested beforehand before the commit. Testing played a vital role in the production of our project. We had to make sure each small part was tested so the user experience would be up to standard. We found problems along the way whilst making of the project.

One of these problems we encountered was the upload of an image.

Through White Box Testing we discovered and fixed the errors we had. We also use a Black Box Testing approach to see what the user experience was like on a user that wasn't apart of the production group. This gave us insight into how a user without experience of using the website was able to navigate throughout the project without any issues and what we need to improve on.

We tested the the features of our project a countless amount of times and noticed a few bugs in the process of this, whenever we found a new bug in the system we worked on a fix and almost always faced another issue somewhere else but we worked as a group to overcome these issues. Testing was such a

vital part of our project since our project is based around a user login and register system it would be a major flaw to allow an user into the incorrect account. This way we made sure there would be no faults or bugs remaining when registering and logging in once the project was completed.

9 Recommendation for Future Development

For future development we will take on board everything we learned whilst doing this project. The team enjoyed the challenge of working with and learning a new language. This is something we can carry over to future development.

We now know even with extensive testing you can still hit many roadblocks along the way. Our experience of this was the framework. At the start we decided on using flask as our framework but we encountered multiple problems with our database and images. With this in mind we made the choice to switch to **Django**. This decision was made for the better of the project and once changed the project came together quickly. These types of problems will be beneficial for us in the future. Our learning of GitHub will be useful in the future. Through various lectures we learned how to use Issues on GitHub this made dealing with problems more manageable.

For future development the approach we would definitely take and begin with would be to fully research and decide on our design decisions, frameworks, implementations, time-management and methodologies. This way we can have it fully scaled out what we are going to achieve week by week. By implementing more structure we may not face any issues like changing frameworks or being indecisive on a design decision. This way we can get our project completed in a timely manner before our deadlines and then have extra time to clean up the code and increase our testing of the project.

10 Conclusions

To conclude,

The making of this project proved to be no easy task. Through making this project each of us learnt and improved various skills, whether it be improved communication skills, working as a team, time management, design and coding ability.

This project will benefit us in our future careers as Software Developers as we moved out of our comfort zones and learned a new programming language and designed and completed a project within a given time frame. After finishing the project and looking back we feel we could have done better in some areas. For example we could have researched more into certain areas such as adding specific user detail's to the database, selecting a framework which systems suited us most and also keeping consistent with developments within the project, This however gave us challenges to overcome.

We now know how to deal with these types of problems if they were to occur in the future when we are working professionally as Software Developers. Ultimately, we are pleased with the project we have produced and enjoyed the independent learning process of a project assigned with free roam of ideas to implement.

11 Acknowledgements

<https://www.fullstackpython.com/table-of-contents.html>

<https://www.fullstackpython.com/django.html>

<https://www.fullstackpython.com/flask.html>

<https://www.sqlalchemy.org/>

<https://www.pythonpool.com/face-detection-using-opencv-and-python/>

face_recognition library

https://github.com/ageitgey/face_recognition/issues/175

Django Cheat Sheet:

<https://github.com/lucrae/django-cheat-sheet/>

Django File Uploading:

<https://docs.djangoproject.com/en/3.0/topics/http/file-uploads/>

<https://medium.com/@muehler.v/node-js-opencv-for-face-recognition-37fa7cb860e8>

Directory Traversal in Python

<https://www.pythoncentral.io/how-to-traverse-a-directory-tree-in-python-guide-to-os-walk/>

Sentdex

<https://www.youtube.com/channel/UCfzlCWGWYyIQ0aLC5w48gBQ>