Raspberry PI Security System

Project Engineering

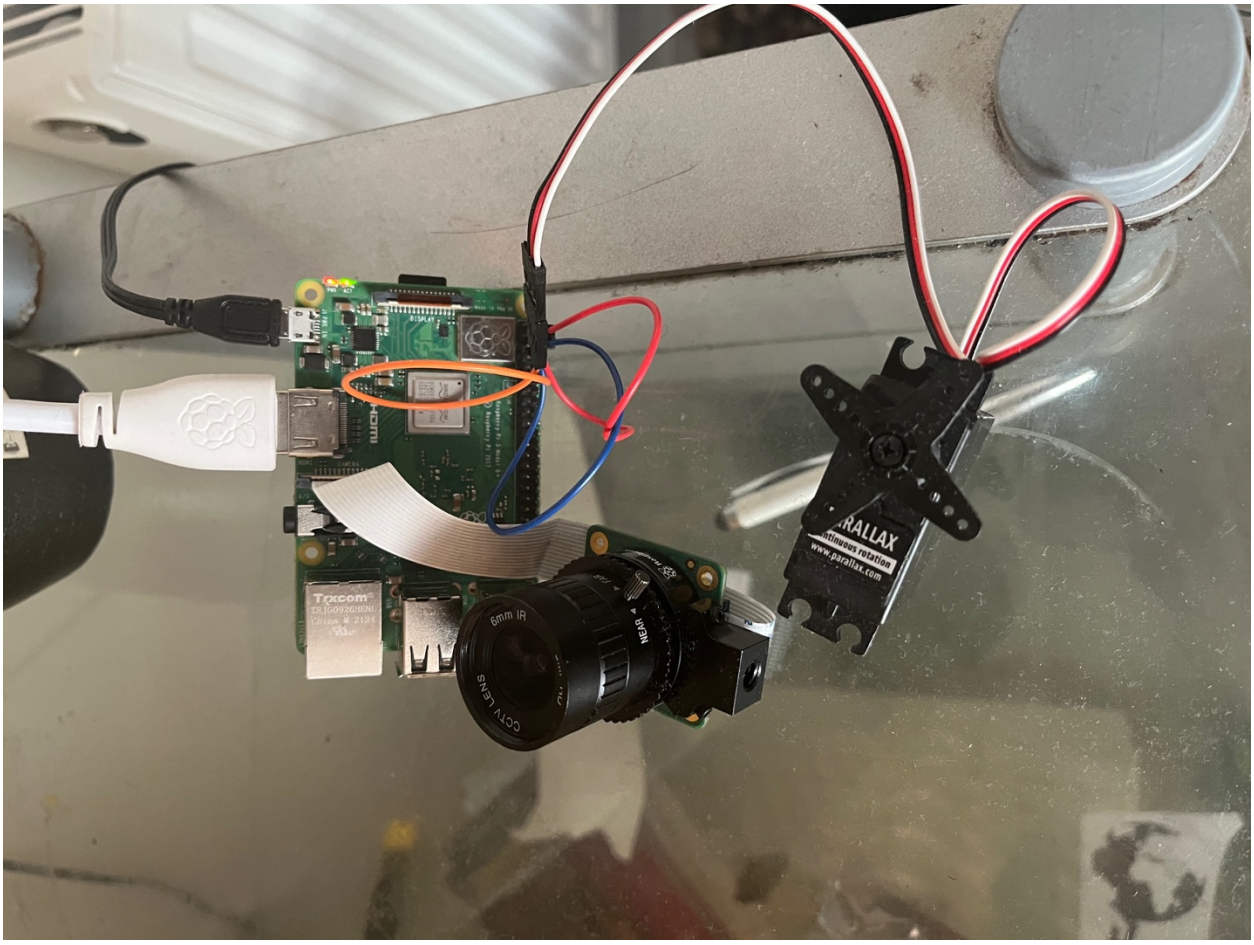
Year 4

Junior Ajala

Bachelor of Engineering (Honours) in Software and Electronic Engineering

Galway-Mayo Institute of Technology

2021/2022


Txcom
6mm IR
CCTV LENS
PARALLAX
continuous rotation
www.parallax.com

Declaration

This project is presented in partial fulfilment of the requirements for the degree of Bachelor of Engineering (Honours) in Software and Electronic Engineering at Galway-Mayo Institute of Technology.

This project is my own work, except where otherwise accredited. Where the work of others has been used or incorporated during this project, this is acknowledged and referenced.

_____*Junior Ajala*_____

Acknowledgements

These are useful links that's helped me throughout my project:

https://github.com/ageitgey/face_recognition - helped me with the main face recognition and I used sample code

https://www.youtube.com/watch?v=o-x1PE0LVKM – helped in the startup and format and used a bit of hos code in my project

Face Detection using Haar Cascades — OpenCV-Python Tutorials beta documentation (opencv24-python-tutorials.readthedocs.io) – gave me a better understanding for facial recognition

https://getbootstrap.com/docs/5.1/getting-started/introduction/ - used this css code in project(helped with nav bar)

https://www.youtube.com/watch?v=hek9vsW8oKs – helped with Sqlalchemy and also with some flask and used abit of her code for database

https://www.youtube.com/watch?v=Z1RJmh_OqeA – helped me with flask

https://www.youtube.com/watch?v=MVgr302PNwY- helped with setting up camera

Table of Contents

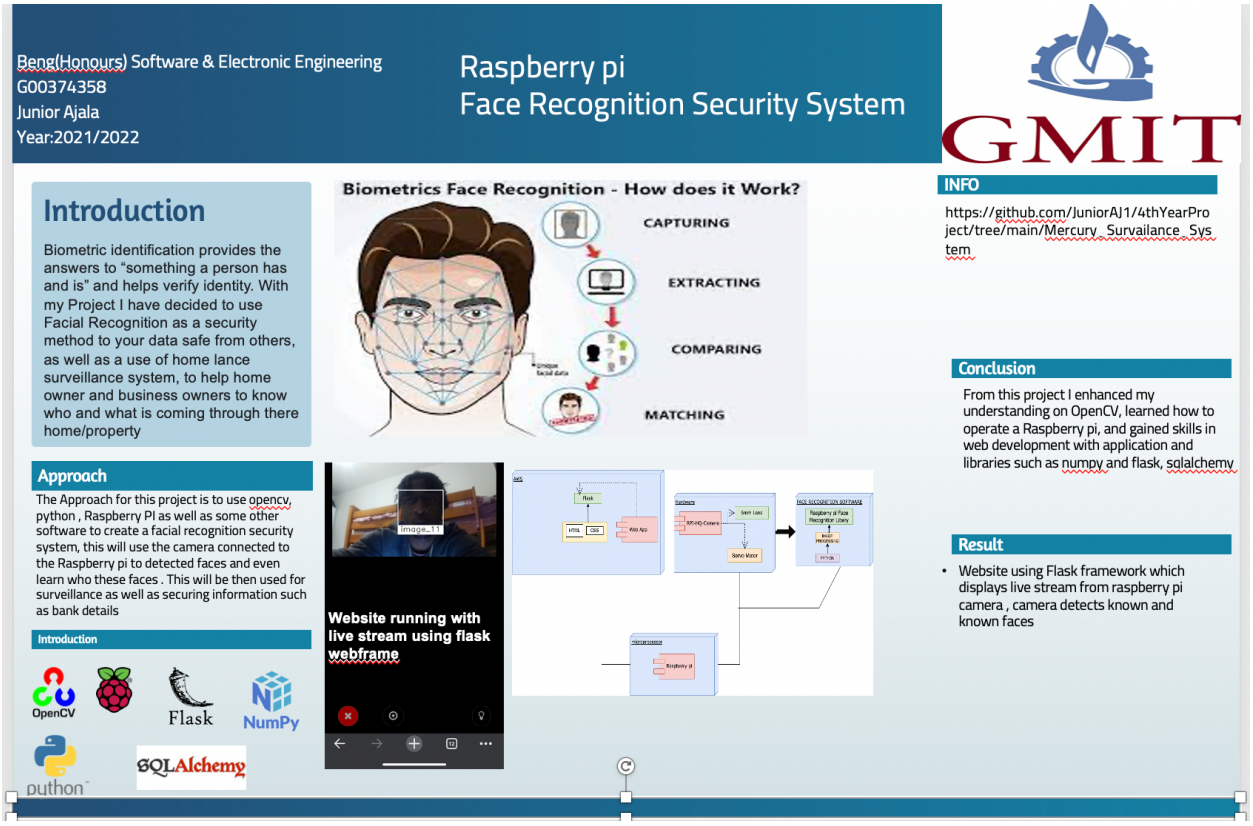
1 Summary

During my third-year work placement in Xperi I was assigned to the image and video processing team here I designed and wrote programs that implement automated test cases to check the quality of image and audio processing products. Analysized, organized and processed large data sets, created scripts that automate the processing of very large sets of imaging and audio data. Prepared detailed reports by writing code to filter and process complex test results data, Carried out comparative evaluations, automated and manual exploratory testing of imaging/audio products in consumer devices and in software applications.

From the Work I done in Xperi I gained interest in Face recognition software and decided to base my 4th year project on it.

In this Project I wrote code to use facial recognition in home surveillance system, so from outside the home or business you can get a live feed of who is going and coming, if a unknown face is in the premises the user will get an alert from there phone or an email

2 Poster


Beng(Honours) Software & Electronic Engineering
G00374358
Junior Ajala
Year:2021/2022
Raspberry pi
Face Recognition Security System
GMIT
Introduction
Biometric identification provides the answers to "something a person has and is" and helps verify identity. With my Project I have decided to use Facial Recognition as a security method to your data safe from others, as well as a use of home lance surveillance system, to help home owner and business owners to know who and what is coming through there home/property
Biometrics Face Recognition - How does it Work?
CAPTURING
EXTRACTING
COMPARING
MATCHING
INFO
https://github.com/JuniorAJ1/4thYearProject/tree/main/Mercury_Survailance_System
Approach
The Approach for this project is to use opencv, python , Raspberry PI as well as some other software to create a facial recognition security system, this will use the camera connected to the Raspberry pi to detected faces and even learn who these faces . This will be then used for surveillance as well as securing information such as bank details
Introduction
OpenCV
Flask
NumPy
python
SQLAlchemy
Website running with live stream using flask webframe
Conclusion
From this project I enhanced my understanding on OpenCV, learned how to operate a Raspberry pi, and gained skills in web development with application and libraries such as numpy and flask, sqlalchemy
Result
Website using Flask framework which displays live stream from raspberry pi camera , camera detects known and known faces

3 Introduction

Over the Years biometrics security has become very important in security purposes. Protecting personal information is getting increasingly difficult as technology pervades every part of modern life and our society becomes increasingly computerized. Passwords and keys were long thought to be enough for data security, but they now appear to be more ineffective in the face of sophisticated hacker attacks. Passwords are, in fact, the weakest link in a company's security system. This is due to the fact that they may be shared, and even those with high entropy can be cracked using a variety of techniques. Recent stories of network security breaches and identity thefts reinforce the need for a solid authentication technique in today's world. As a result, biometric security has become a priority, as it is the only reliable way to verify an individual's identification. Biometric features are physical and behavioural characteristics that are intrinsic and unique to each individual, and include fingerprints, face, iris, stride, voice, and so on. As a result, biometric security systems can validate an individual's identity with utmost accuracy and dependability because biometric qualities are ingrained in their DNA. For this Project my Goal is to use the Biometric feature Face Recognition, to be used to aid a surveillance system. With the code I have Written home owners or Business owners, will be able to catch a live video of the house/business and with the facial recognition will help detect who was around and alert owner of any unfamiliar faces.

My Project will consist a raspberry pi and Picamera, on the Pi's OS I wrote python code with the help of OpenCV and the face_recognition libraries which helped with the Facial Recognition and a Flask framework which allowed me to host a webpage using python and HTML

4 Getting Started

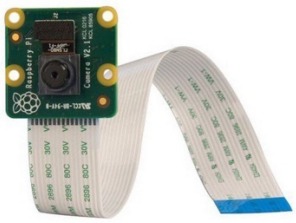
4.1 Raspberry pi

For my project I decided to run and write my code on a raspberry pi 3, “Raspberry Pi is the name of a series of single-board computers that runs Linux, but it also provides a set of GPIO pins, allowing you to control electronic components”.Raspberry Pi is used learn programming skills, build hardware projects, do home automation, implement Kubernetes clusters and Edge computing, and even use them in industrial applications, the reason I came to using the Pi was because I wanted to show I could work with an embedded system and that I could work with a Linux operating system as I want to further my career as an embedded software engineer

When I got the Raspberry pi in December, there were a few steps I had to take to get it up and running, the first was to buy a micro SD card and Install the Raspberry Pi OS using Imager. The imager allows for Anything that’s stored on the SD card will be overwritten during formatting. If your SD card currently has any files on it. I then powered the Pi and loaded it up with an External screen with a HDMI and I connected my mouse and keyboard to it. My first task was to be able to write code from the raspberry pi on my laptop, for this I needed to connect to it remotely. I connected the Pi to the internet and downloaded VNC viewer on my laptop and pi. With this I was able to display the operating system on my laptop and I was ready to start coding

4.2 Pi Camera

With this project, I needed a camera that works well with the raspberry pi, luckily there was a camera very compatible with the pi made by the same company. I watched a couple videos on YouTube to set it up and learned code to operate it, when first giving a camera I was given the Raspberry pi camera Module v2



With this camera, everything worked fine but it was a bit pixilated when taking videos, as the camera was made quite some time ago I wasn't to surprised, so I was later giving a Raspberry pi HQ camera and a 6mm lens which could be adjusted for lighting

When I got the camera I just messed around with code, I first learned how to start a video and take pictures I later created a script which ran a video and when the user presses the space key it took a screenshot from the video, this was later used to set up the user into the database for face recognition

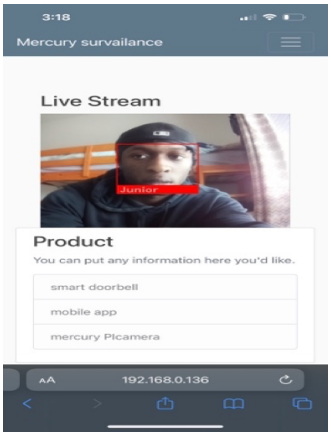
4.3 Thonny Python

On the raspberry pi operating system, thonny python was already installed as it was Pi's Default Python IDE, Thonny Python is an integrated development environment for Python that is designed for beginners. There was also another code editor called Greaney, most of my code was planned to be in Python so I used Thonny python, and wrote the html and css in Greanny.

With thonny Python I was also able to send Signal to the GPIO pins on the Pi which helped control some hardware such as the servo motor and camera

4.4 OpenCV and Numpy

OpenCV is a great tool for image processing and performing computer vision tasks. It is an open-source library that can be used to perform tasks like face detection, objection tracking, landmark detection, and much more. The library is equipped with hundreds of useful functions and algorithms, which are all freely available to us. Some of these functions are really common and are used in almost every computer vision task. In my Project I use opencv to sense and an accurate reading of the users face. OpenCv also allows and help stream the video, and capture snippets from it , for example when an intruder is spotted or an unknown face is spotted I can use opencv to take a snippet zoom in, enhance the resolution and store it, In my code a mixture of opencv and numpy are used when writing a box around the captured face



So in the photo above, as seen a the camera has sensed my face and there is now using numpy which is a Python library used for working with arrays. It also has functions for working in domain of linear algebra, Fourier transform, and matrices, I use numpy to store essential array needed for the facial recognition when the face is then detected thanks the face_recognition library, opencv is then used to rescale the image and face location and then displays a box around the users face printing the name from the said array

5 Project Architecture

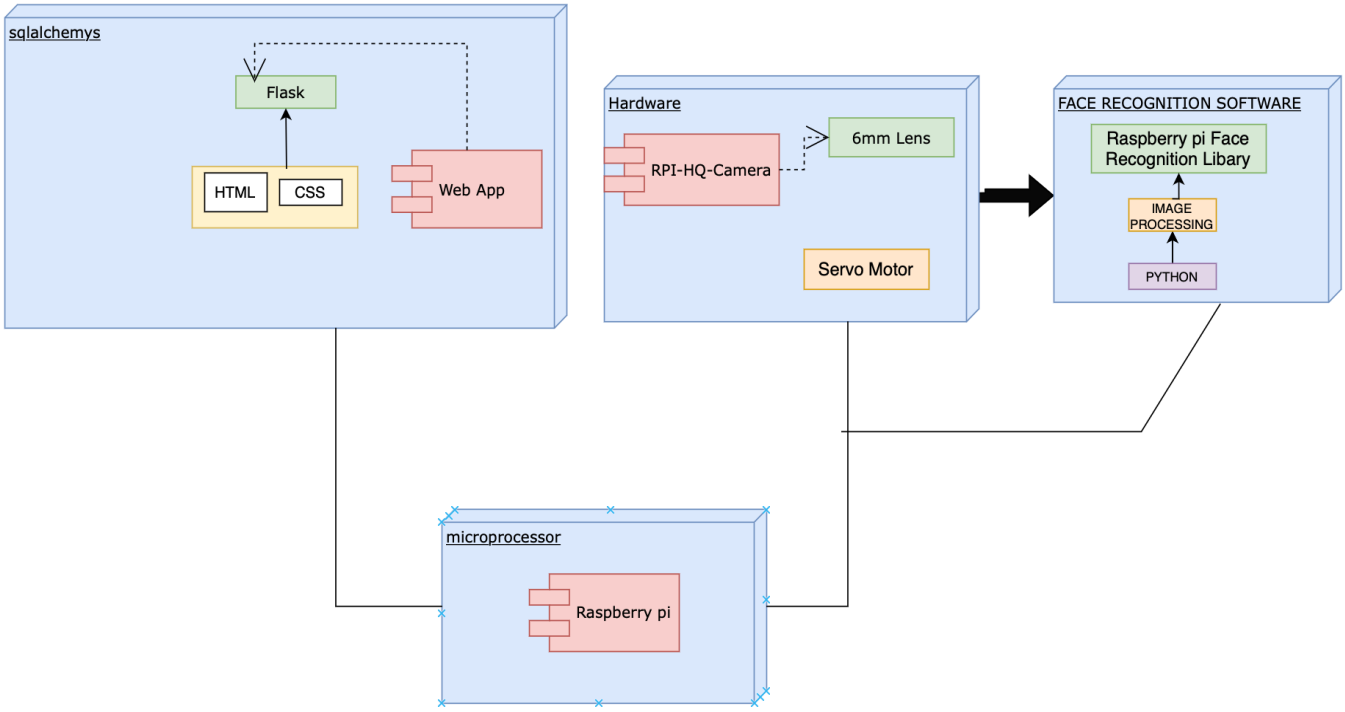
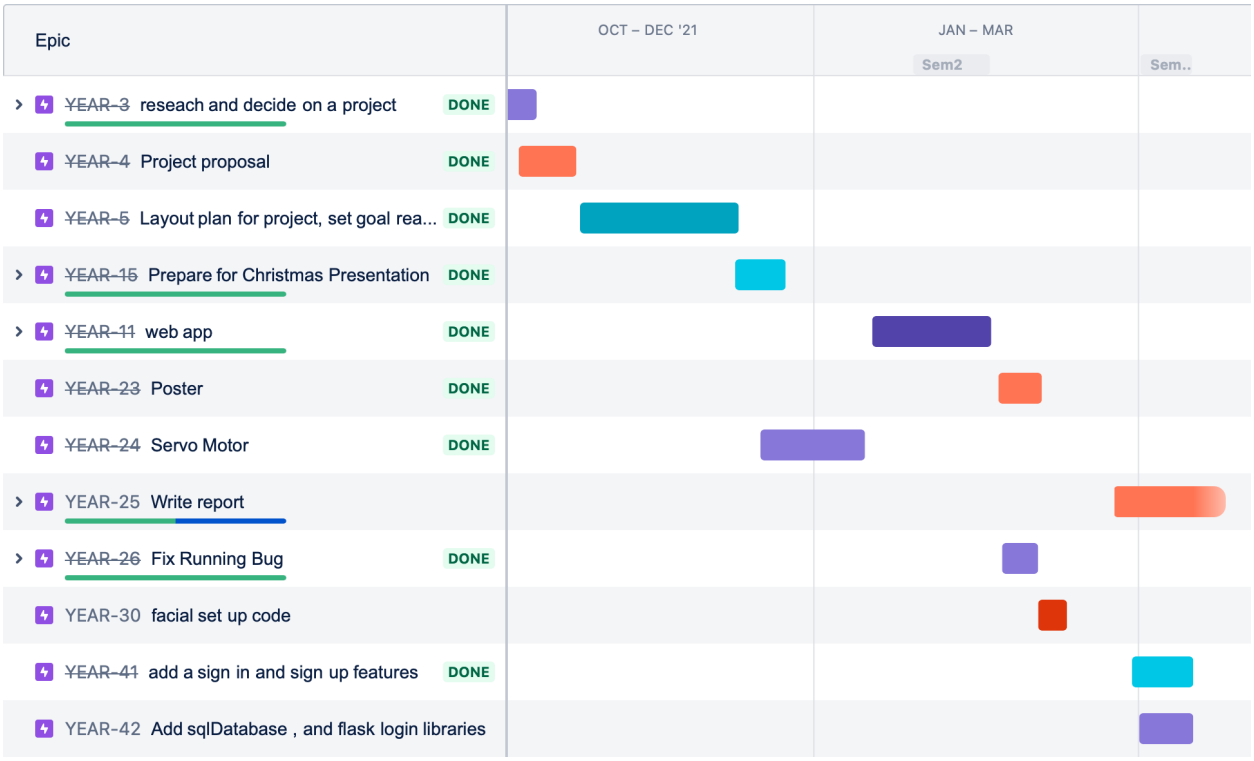


Figure 5-1 Architecture Diagram

6 Project Plan



For my Project plan and Project Management I used a website called Jira, from my recent work placement I got familiar with Jira, and learned more about its functions , after coming back from placement I learned a lot of my class mates also used Jira at work, I came to terms that Jira mustt be a very popular website that a lot of companies use for Project management. This grew my motivation on using Jira for my final project as it show Employers that I'm still very familiar with the software

With Jira I created Sprints and task which had due date on when it should be done, so for example I set a task from myself on Designing the webpage since I had other projects and assignment I would estimate the duration to about 2 weeks.

7 Main Aspects

In this section I will discuss the main aspects and functionality of my project and its code, another brief Summary of my Project is A Surveillance webpage that uses Flask frame work to run. My Project uses a raspberry pi to execute the code and has an external Pi camera attached to it. In this webpage users login or create an account which will lead a live Steam view of the contents of the PiCamera, with help of OpenCV and the Facial recognition, the camera is now able to detected a person's face, if the users is not in the database(profiles folder), the camera will detect the user as unknown, if the camera does the detect a face from the profiles folder, the name of the person will then be printed below.

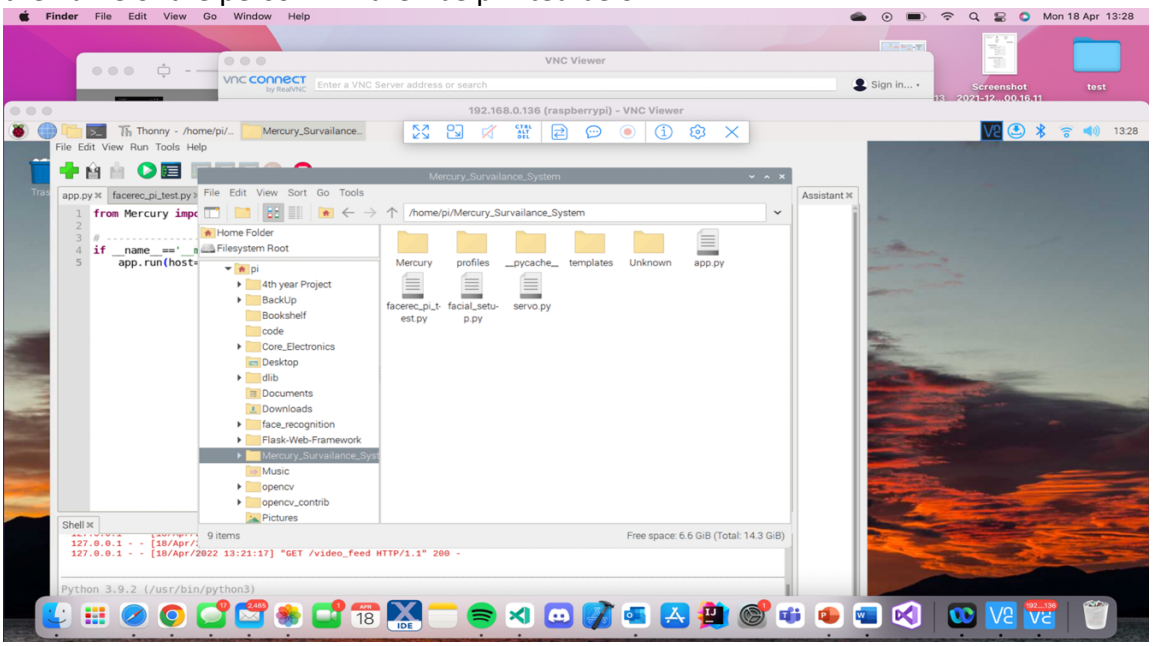


Figure 7-1 Main Project Folder

The above photo is the main folder which stores my code and photos of faces to be recognized by the camera. The reason for the folder being called "Mercury surveillance system" is because that's what I've decide to call this company. The main folder where the code for the flask

website is in the mercuy folder.

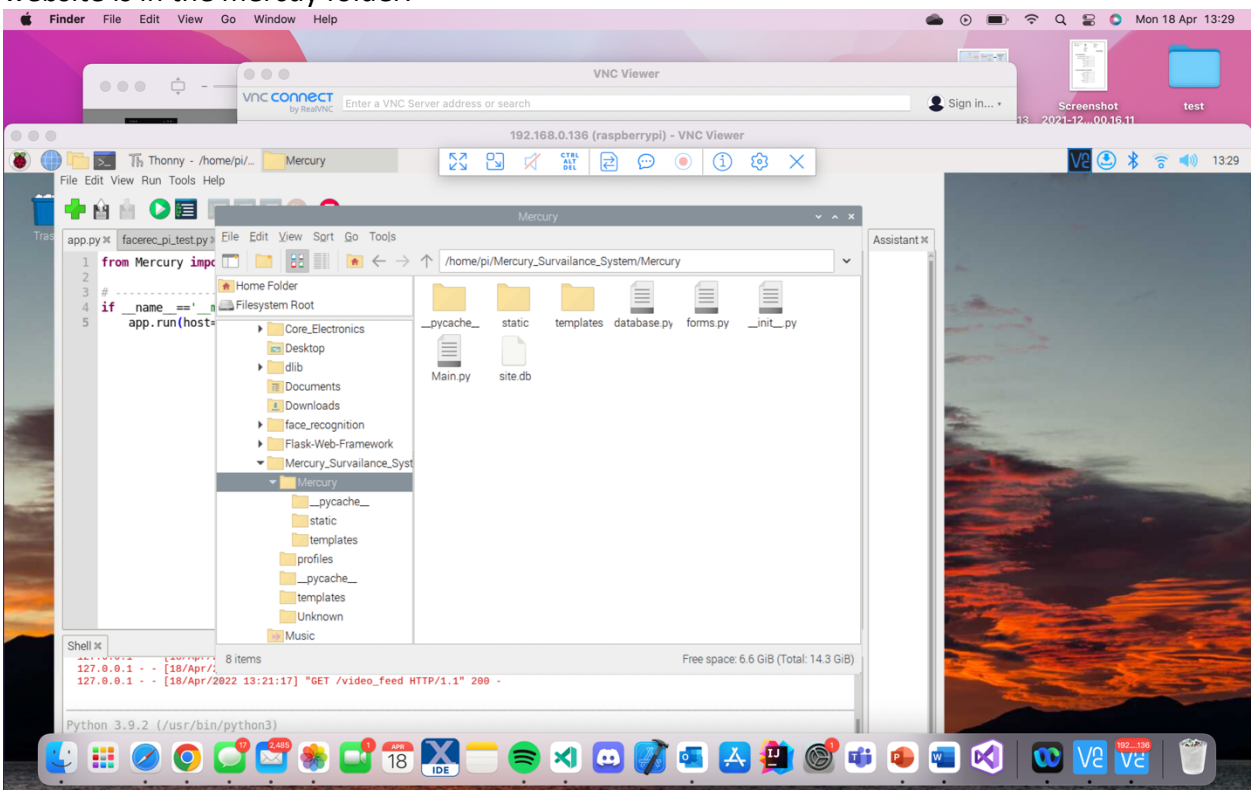


Figure 7-2 Main Project Folder

Here I packaged my python code and divided it into different python files so it could be easier for others to read. My “site” database is also here where I store registered users’ information. I store my Css code in the static folder which corresponds with my HTML code which I store in the template folder. The __init__.py file is an instillation file which helps import all the libraries used

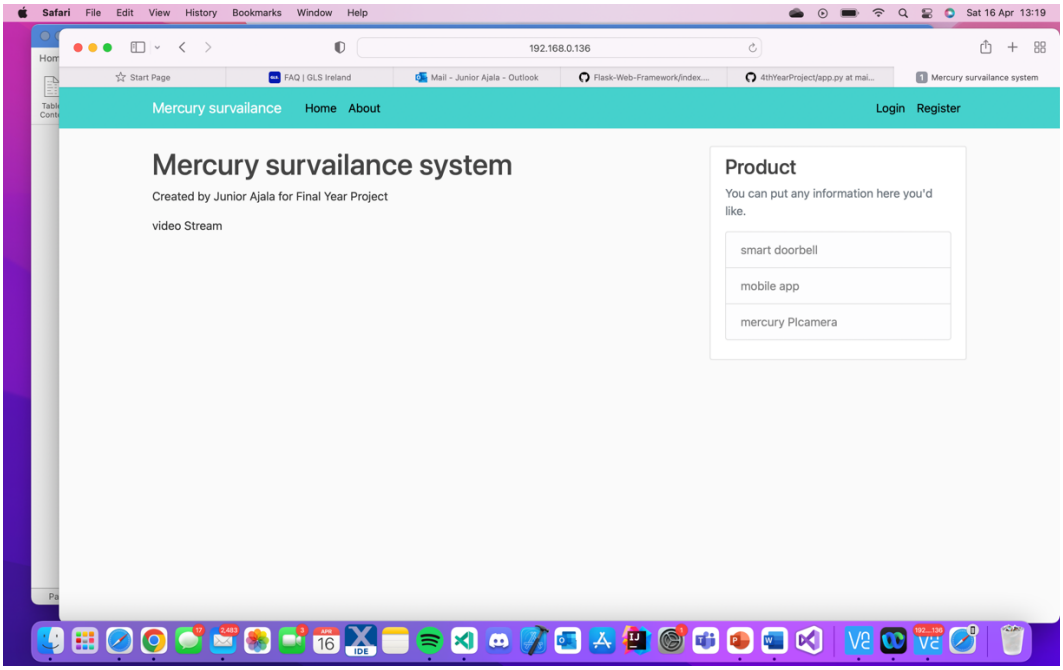
7.1 Flask

Flask is a small and lightweight Python web framework that provides useful tools and features that make creating web applications in Python easier. It gives developers flexibility and is a more accessible framework for new developers since you can build a web application quickly using only a single Python file, unlike react and a few other frameworks it does not require particular tools or libraries. It has no database abstraction layer, form validation. Since I was my project mainly was in Python I chose flask, there was another Python framework called Django, after further research I seen more people use flask with there raspberry pi for issues such as bugs and some library issues, so I stuck with flask just in case If had issues I could look at the raspberry pi forum page or YouTube for help. In my project the flask code is written in the main.py. here I wrote code that helps with routing, through pages, I also wrote code that links to the HTML in the template to help format the webpage and make it look user friendly

7.2 Webpage

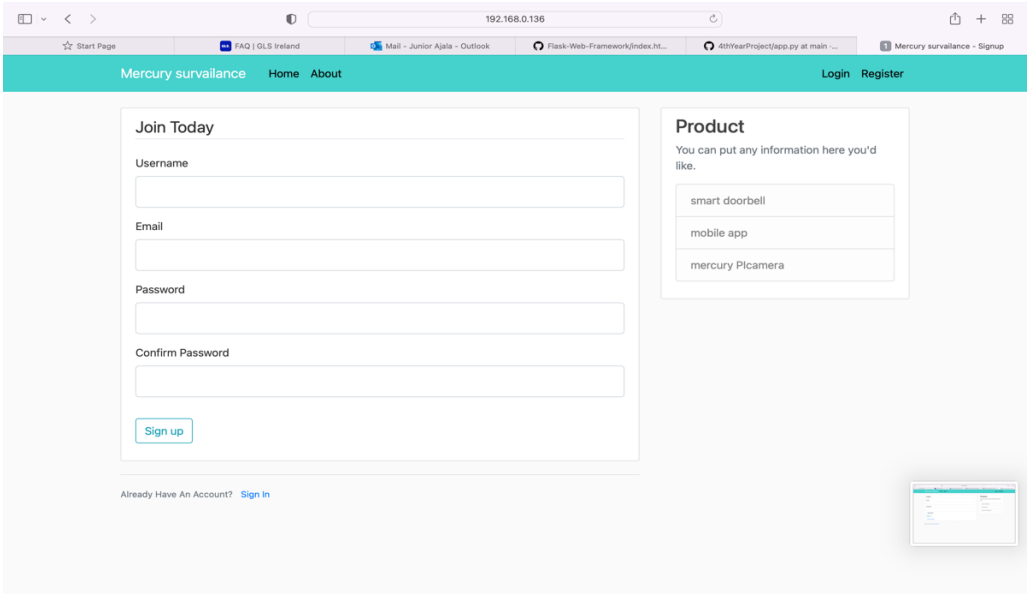
Using flask and Html I created a number of pages with the Webpage, which are routed together on the main.py folder using the flask libraries

Home page:



Here the users can see a brief view of the company name and the nav bar

Regisitration/Sign Up Page:

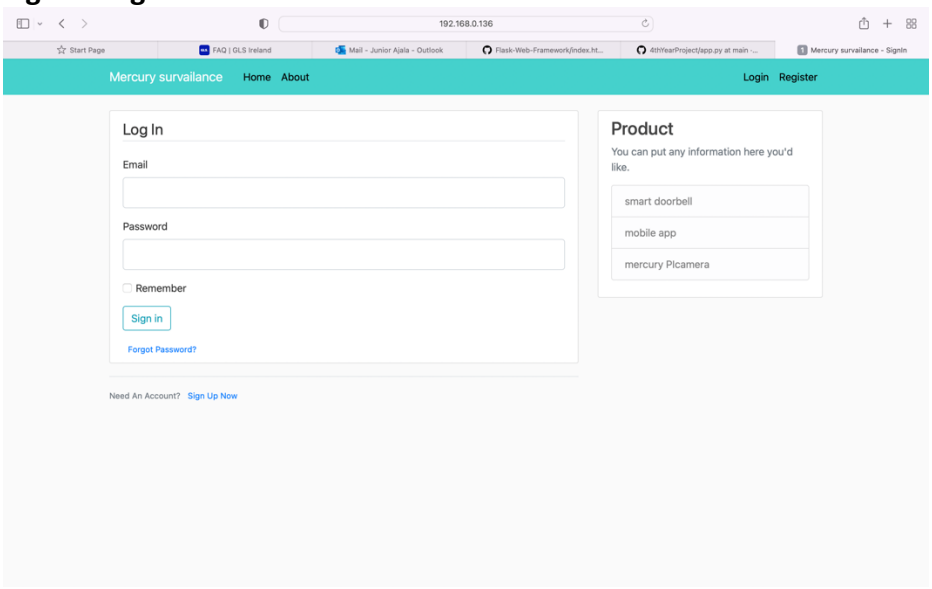


Here in the Signup page users can enter a username ,email and a password when the user presses the “signup” button. The webpage then checks from some validations. In my forms.py I wrote code that checks for validators , A validator simply takes an input, verifies it fulfils some criterion, for example I wrote code so that the maximum Characters in there user name is 20

```
class RegistrationForm(FlaskForm):
    username = StringField('Username',validators=[DataRequired(), Length(min = 2, max = 20)])
    email = StringField('Email', validators=[DataRequired(),Email()])
    password = PasswordField('Password', validators=[DataRequired()])
    confirm_password = PasswordField('Confirm Password', validators=[DataRequired(), EqualTo('password')])
    submit = SubmitField('Sign up')
```

There also code to check if the “Password” and the “Confirm Password” match. If the validators catch an error there be a warning on the screen, for example if the email isn’t identified as an email, users will get an warning message saying, “Please check email”
After all validators have passed with no errors, users Username, email and password are then put into the SQLalchemy database and user is moved into the sign in page

Sign in Page:



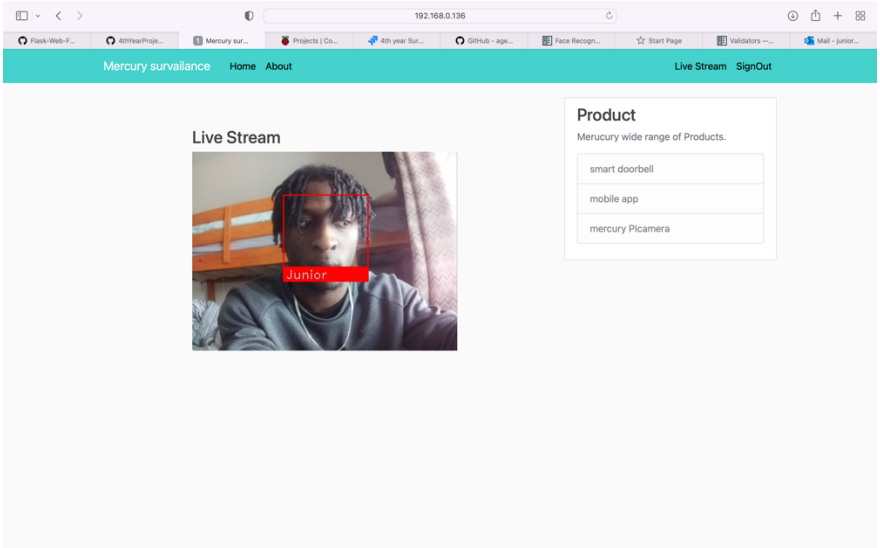
In the Log In page users enter the same username and password they entered into the registration page and the webpage will then check and see if the database. if these are in and if they match with the right table in the database it will redirect the user to the live feed page

```
@app.route('/SignIn', methods=['GET','POST'])
def Signin():
    if current_user.is_authenticated:
        return redirect(url_for('Live_stream'))
    form = SigninForm()
    if form.validate_on_submit():
        user = User.query.filter_by(email=form.email.data).first() #checks database for the email entered
        if user and bcrypt.check_password_hash(user.password,form.password.data):
            login_user(user,remember=form.remember.data)
            return redirect(url_for('Live_stream'))
        else:
            flash('Login Unsuccessful. ', 'danger')
    return render_template('SignInpage.html', title= 'SignIn',form=form)
```

The code above shows how the webpage first checks to see if the username and password are in the database. Users then have an option for the webpage to remember then and it checks for that's. the redirect(url_for('Live_stream')) is used to direct the user to the live stream page. This will only happen if the users details are in the database and if the validators pass with no errors. As with the Registration the signup Validators are also in the form.py folder

```
class SigninForm(FlaskForm):
    email = StringField('Email', validators=[DataRequired(),Email()])
    password = PasswordField('Password', validators=[DataRequired()])
    remember = BooleanField('Remember')
    submit = SubmitField('Sign in')
```

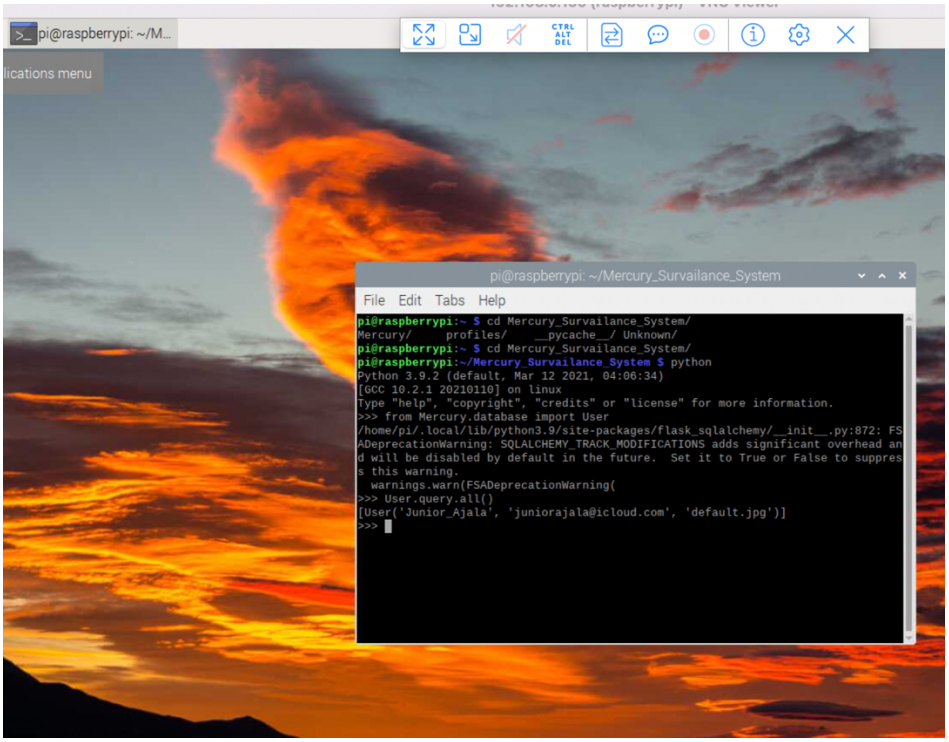
Live Feed page:



When the user logins in through the signup page successfully there are now directed to the live stream page here users get live footage from the PIcamera, and the facial recognition is now at work

7.3 Database

For the mercury Surveillance system website, as we've seen in order for users to view the live footage they must register their account then sign in. how does the webpage remember the users sign up info and where is it stored? This is thanks to SQLalchemys database, "SQLAlchemy is the Python SQL toolkit and Object Relational Mapper that gives application developers the full power and flexibility of SQL". Since I was using a SQL lite the database was stored in my project directory as "site.db". so with this was able to access and see the data in the site.db through the command prompt, where I can see users username and emails.



For the database the main code for it is written in the database.py file. Here I created models for the database table. In each table the user name , email and photo are stored

```
class User(db.Model,UserMixin): #usermixin adds all the required methods needed for the login_manger
    id = db.Column(db.Integer, primary_key=True)
    username = db.Column(db.String(20),unique=True,nullable=False)
    email = db.Column(db.String(100),unique=True,nullable=False)
    image_file = db.Column(db.String(21),nullable=False, default='default.jpg')
    password = db.Column(db.String(60),nullable=False)

    def __repr__(self): # how are object is printed
        return f"User('{self.username}', '{self.email}', '{self.image_file}')"
```

The “unique” attribute allows for only username and email of that type to be in the database, so no one can use the same email twice or the same user name twice. The code also helps store the users password but from the command prompt it is not shown this was to keep the users password private and safe. Another way I secured the users password is by hashing it in the db. To the hash the password I used bcrypt. “The bcrypt hashing function allows us to build a password security platform that scales with computation power and always hashes every password”. So once the user has created the password and the password has passed all validators it is then hashed by bcrypt, this is done in the main.py file

```
        hash_p = bcrypt.generate_password_hash(form.password.data).decode('utf-8') # this hashes the password so it can be secured from any db hacks
        user = User(username=form.username.data, email=form.email.data,password=hash_p) #add users info and passwords to database
        db.session.add(user)
        db.session.commit()
```

In this code we first hash the password using the bcrypt , we then put the ‘user’ in the database format, we then add and commit this info into our SQL database

7.4 Facial Recognition

In my project I wrote two files that use facial recognition, one is in the main part of the project which is facial recognition in the mercury webpage and the other is just a normal file that run the live footage on the pi, this is in the facialrec_pi_test.py and for the webpage the facial recognition code is written in the main.py file which is in the mercury file. The facial detection works thanks to facial recognition library I got on
https://gist.github.com/ageitgey/1ac8dbe8572f3f533df6269dab35df65

This page helped be download and install all the Dlibs and libraries for the detection
There was also example code which helped a lot with my code. Since I didn't understand the library at first these example code where a big help in the facial recognition aspect of my code

```
# Face recognition code
camera = cv2.VideoCapture(10000)
# Load a sample picture and learn how to recognize it.
junior_image = face_recognition.load_image_file("/home/pi/Mercury_Survailance_System/profiles/Juniorr.jpg")
junior_face_encoding = face_recognition.face_encodings(junior_image)[0]
```

The code starts by using by using OpenCV to start a video with cv2.VideoCapture
We then scan through the picture and the with the face recognition library the code now does machine learning and learns how to recognize the face on the picture

```
31  # Create arrays of known face encodings and their names
32  known_face_encodings = [
33      junior_face_encoding,
34      ironMan_face_encoding
35  ]
36  known_face_names = [
37      "Junior",
38      "IronMan"
39  ]
```

We then add these face encoding to an array which is later used in identification. Due to some errors I wasn't able to use all photos in the profiles folder so, I added them manually.

```
67              for face_encoding in face_encodings:
68                  # See if the face is a match for the known face(s)
69                  matches = face_recognition.compare_faces(known_face_encodings, face_encoding)
70                  name = "Unknown"
71                  # Or instead, use the known face with the smallest distance to the new face
72                  face_distances = face_recognition.face_distance(known_face_encodings, face_encoding
73                  best_match_index = np.argmin(face_distances)
74                  if matches[best_match_index]:
75                      name = known_face_names[best_match_index]
76
77                  face_names.append(name)
```

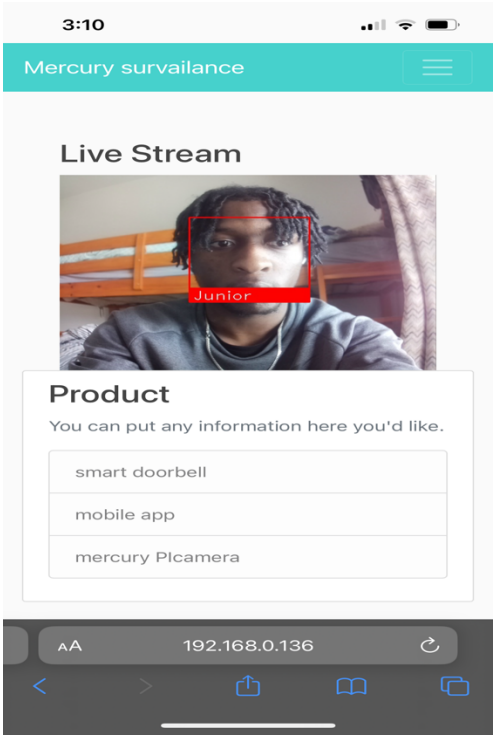
This code if for when the camera detects a face, here it compares detected face with faces in the known_face_names array if it's a match we declare the name variable as which face was detected and this name variable is then used to print the person's name underneath

8 Ethics

The in today's time , the quality of CCTV is vast and becoming better, I think thanks to these now high resolution footage we can secure and protect peoples home and business with things such as facial recognition. With facial recognition people have that extra layer of security not just in homes but in personal data. I hope to further my understanding. In using biometric security to help secure and protect peoples data

9 Conclusion

In conclusion used a Raspberry pi and I created a webpage that was run using a flask framework, in the webpage users and register their accounts which will be saved in the SQLalchemy Database, in this database a user's username email and password are stored and using Bcrypt the password are hashed which will help prevent hacks. With these saved details users can then sign in which will lead them to a live stream page that will allow users a full view of footage that's coming from the pi camera, with the code written the Pi camera is able to detect for faces and able to recognise faces in the profiles folder. This project really helped in my skills in coding and project management. Using Jira helped wider my understanding in working In a big team with other software engineers. Was a very great experience creating a project that I was very interested in. for now this was just a prototype of my project but in the future I would like to expand this project and make surveillance company where users feel safe and secured at home



10 Reference

[x] IEEE Signal Processing Society. “Signal Processing for 5G”, *YouTube*, 2019. [Online]. Available: https://www.youtube.com/watch?v=uca6X4Ykcmg&list=PLcZOnmyqlalacL9YqkhyufLQGIW_C78Os&index=6. Accessed: Feb 2, 2021.

[x] M. Lynch. “Discrete Fourier Transform (DFT)”, Lecture, Digital Signal Processing, Galway-Mayo Institute of Technology, Galway, 2020.

[x] OpenCV. “Face Detection Using Haar Cascades”, *OpenCV-Python Tutorials*. [Online]. Available: https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html#face-detection. Accessed: Feb 2, 2021.

https://www.npmjs.com/package/bcrypt - Bcrpyt

https://www.sqlalchemy.org/ - SQLAlchemy

https://getbootstrap.com/docs/5.1/getting-started/introduction/

https://gist.github.com/ageitgey/1ac8dbe8572f3f533df6269dab35df65