University of Plymouth

School of Engineering, Computing, and Mathematics

COMP3000
Computing Project
2022/2023

FaceCCTV

Harry Stephen Parker

10651820

BSc (Hons) Computer Science (Cyber Security)

Acknowledgements

There are many people I was fortunate enough to have around me that I would like to thank for their guidance and support throughout this project; firstly, my supervisor David Walker for guiding the project and ensuring I was on the right path while focusing on the right aspects and answering any questions I had about the project. Additionally, I want to thank my friends for always being supportive and my family especially: Gill, Stephen, Ollie, and Alex.

Abstract

Research shows that the crime rate percentage in the UK has been on the rise, seeing a 10% increase in crimes recorded in the year ending September 2022, 6.6 million respectively, compared to the last year which wasn't affected by the pandemic which was the year ending March 2020, which had 6.1 million crimes recorded. This is only fair as in the year ending in September 2021, police recorded that crime fell to 5.8 million due to pandemic restrictions. In the year 2022, only approximately 5% which equates to 350,000 crimes were brought to court and the criminal was charged. You wonder why? This is partly due to the poor quality of CCTV, which makes finding the right person more difficult. This report, the process of planning, designing and developing a cross-platform application called "FaceCCTV" that security analysts can use to upload their CCTV images and receive a high-quality snapshot of all the faces detected in the image. This application's complete software development life cycle will be documented thoroughly, starting with an introduction which will provide a brief outline of this document. From there, the project's background will be explained, mostly justification of why the application should be developed. Some market research will be carried out to find what potential gaps in the market the software can fill. Then the relevant legal, social, ethical and professional issues will be discussed, to ensure that the application is taken very seriously, with a key goal of data security as it uses sensitive data. After this, the functional and non-requirements of the project will be expressed gathering the end goal for users. After the planning section of the project had been completed, the design process followed with the primary goal of producing prototypes of the interface and diagrams of user interaction. By having a precise understanding of the elements mentioned above, the development approach began with deciding on a tech stack and breaking the project into sprints, alongside any other necessary project management. Then the actual development process of the software and final testing to ensure that the project's objectives and aims have been reached. An evaluation and summary will follow to conclude the report, discussing the project's accomplishments, setbacks, and overall management.

Table of Contents

Acknowledgements **1**
Abstract **1**
Table of Contents **2**
1. Introduction **5**
1.1. Aims and Objectives 6
1.1.1. Main Objectives 6
1.1.2. Side Objectives 7
1.1.3. Security Objectives 7
1.2. Minimum Viable Product 7
2. Background **8**
2.1. Project Background 8
2.1.1. What is Face Detection in Artificial Intelligence? 9
2.2. Existing Applications 10
2.2.1. Functionality 10
2.2.2. User Findings 10
2.3. Limitations of Current Solutions 11
3. Legal, Social, Ethical and Professional Issues **12**
3.1. Legal 12
3.2. Social 12
3.3. Ethical 12
3.4. Professional 13
4. Design **14**
4.1. User Requirements 14
4.1.1. User Persona 14
4.1.2. User Stories 14
4.1.3. Functional Requirements and Product Backlog 15
4.1.4. Non-Functional Requirements 16
4.2. Diagrams 16
4.2.1. Architecture Diagram 17
4.2.3. Sitemap 19
4.2.4. API Class Diagram 20
4.3. User Interface 21
4.3.1. Wireframes 21
4.3.2. Mockup Designs 22
4.3.3 Final Designs 23
5. AI Architecture **24**

6. AI Development Pipeline **26**
6.1. Stage 1 - Design and Research 26
6.2. Stage 2 - Data Collection and Pipeline 26
6.3. Stage 3 - Deep Learning CNN 27
6.4. Stage 4 - Building the Neural Network 27
6.5. Stage 5 - Training, Testing and Plotting Results 27
6.6. Stage 6 - Deploying AI 29
7. Development Approach **30**
7.1. AI Model 30
7.1.1. Face Detection 30
7.1.2. Image Enhancement 30
7.2. API 30
7.2.1. FastAPI 31
7.3. Client-Side Development 31
7.3.1. Web App 31
7.3.2. Desktop App 31
7.4. Security 32
7.4.1 HTTPS and SSL Certificates 32
7.5. Development Tools 32
7.6. Version Control System 33
7.7. Hosting and Production 33
8. Project Management **34**
8.1. Risk Assessment 34
8.2. Gantt Chart 37
8.3. Kanban Boards 38
9. Development **39**
9.1. Initial Challenges 39
9.2. Sprints 39
10. Testing and Quality Assurance **44**
10.1. Usability Testing 44
11. Project Closure **45**
11.1. Aims and Objectives Review 45
11.2. Project Evaluation 45
11.2.1. Tech Stack Evaluation 46
11.2.2. Project Management Evaluation 46
11.2.3. Personal Reflection 47
11.2.4. Areas for Future Improvability 48
11.3. Conclusions 49

References **50**
Appendices **53**
Appendix 1: Kanban Boards 53
Appendix 2: Software Engineering 57
API Return Types 57
Appendix 3: Production Build Test Results 58
Appendix 4: API Documentation 60
Appendix 5: Biweekly Supervisor Meetings 61
Appendix 6: User Guide 62
AI Model 62
Web App 63
API 63
Web App local use 63
Desktop App 64
Walkthrough on the use of the app 64

Word Count: 10196

GitHub Repository (Source Code): https://github.com/Parker06/FaceCCTV

Dataset links for AI Model:

https://drive.google.com/drive/folders/1Y11KhmhUfg3q6JRAv4idBBt-OuFEvQBv?usp=sharing (Custom Collected)

https://www.tensorflow.org/datasets/catalog/wider_face (WIDER Face dataset)

http://vis-www.cs.umass.edu/fddb/ (FDDB Dataset)

LInk for AI Models:
https://drive.google.com/drive/folders/1uvfchtz6BeO0tNeOHRmjQnztngTclMSs?usp=sharing

Website: https://facecctv.co.uk

Demo Video: https://youtu.be/bSYXfr-A_B4

1. Introduction

In this report, the full software development life cycle process of a Face Detection Artificial Intelligence Tool called “FaceCCTV" will be explained. The software itself is a cross-platform application, which includes a web, an a desktop app. The project aims to provide users with an tool in which they can upload images from their CCTV footage and allow the program to detect faces in the pictures and enhance them to give them a hand in identifying criminals or ‘bad guys’.

Firstly, the report will provide a thorough background for the project, alongside in-depth analysis and research to identify areas that the software can improve from existing solutions. The background research would serve as a foundation upon which the aims and objectives of the project will be based.

Once the project's background has been discussed, the legal, social, ethical, and professional issues will be explained and analysed. This is crucial as the application will be handling sensitive data so this section is important. Subsequently, the design process will follow, with the gathering of user requirements, and the production of wireframes and mockups to ensure the interface is consistent across different platforms and accessible to users. Diagrams would also be used to visualise how the user would interact with the interface.

Alongside the design process after gathering the user requirements then we can discuss the architecture of the Artificial Intelligence Model to really understand how it will work and then we will move on to the AI development pipeline which elaborates on the development process that the AI will undergo.

As the previous sections have been discussed, it would then be possible to decide on a suitable tech stack, as well as a reasonable development approach with the appropriate tools and technologies. Having decided on a tech stack and approach, it would then be possible to better estimate the schedule of the project and plan it. The project management will therefore be discussed, along with the sprints as part of the agile methodology.

The development, testing, and quality assurance of the software will then be analysed and discussed, with the use of diagrams to show the architecture of the software.

Finally, a summative and conclusive overview along with an evaluation will be provided.

1.1. Aims and Objectives

The aims and objectives of FaceCCTV were primarily identified through the market research that was carried out and results are outlined in section 2. The primary aim of the application is to provide its users with free, server-hosted, and cross-platform software, with a responsive web app and desktop app that allows individuals to upload their CCTV footage to get in return an enhanced copy with all of the faces labelled with bounding boxes. This would all be available to the user in a unified and accessible application with the necessity for a modern, simple user interface, alongside some emphasis on HCI and user customizability.

1.1.1. Main Objectives

The main objectives of this project are to…

- Research the market and feedback obtained from FaceCCTV's competitors to help identify user requirements.
- Research the legal, social, ethical and professional requirements.
- Develop a server-side hosted API to communicate with the front end and the AI model.
- Create an AI that can detect the faces of people.
- Once detected, the AI enhances the image so the user can see clearly.
- These functions can be individually used.
- Develop a production-ready, free, and server-side hosted, unified web, and desktop application for client-side functionality.
- Produce a report that thoroughly examines the software development life cycle of the project, the research carried out before and while developing it, and the challenges faced throughout its duration along with lessons learned from creating a substantial and complex piece of software alone.

1.1.2. Side Objectives

These are objectives which should only be tackled once the main objectives are completed if there is time in the development process left.

- The AI is optimally developed so there aren't any performance issues.
- The AI should be able to handle any image, in size and resolution.
- The Image could be colourized with AI after the enhancement process.

1.1.3. Security Objectives

As this project is handling very sensitive data, the app nor the developer is not permitted to allow users to store CCTV on my website for later use in case of a data breach. Also if the developer had the intent to access the data themselves this could conflict with the rules of the Human Rights Act 1998, 2018 GDPR Regulations, etc…

As users can upload images to the website via a mobile network connection or through WiFi, the images will be encrypted in transit to ensure that unauthorised parties can't intercept them whilst uploading/downloading.

1.2. Minimum Viable Product

A minimum viable product is released to a small number of initial users that essentially test that system which has basic functionality, as this project is working with Artificial Intelligence it can not be an MVP without the core aspect so once the AI has been developed to a basic functional level where it will work for most images then a set of usability tests will be carried out. For the product to qualify as a minimum viable product, the following features need to be present and working:

- Navigation - the user should be able to easily navigate around the app.
- Image Upload and Download - the user should be able to upload an image and download the enhanced image.
- Face Detection - the AI should be able to detect faces.
- Image Enhancement - Once detected all of the faces are, the AI should be able to enhance the image’s quality.
- It should be a web app at the least when pushed into production.

2. Background

2.1. Project Background

Overall, evidence shows CCTV has reduced crime. Research shows that the crime rate percentage in the UK has been on the rise, seeing a 10% increase in crimes recorded in the year ending September 2022, 6.6 million respectively, compared to the last year which wasn't affected by the pandemic which was the year ending March 2020, which had 6.1 million crimes recorded. This is only fair as in the year ending in September 2021, police recorded that crime fell to 5.8 million due to pandemic restrictions. In the year 2022, only approximately 5% which equates to 350,000 crimes were brought to court and the criminal was charged.

In the College of Policing located in the UK, they found that CCTV was associated with a statistically significant decrease in crime. Overall, across a range of settings, crime was found to have decreased by 13% in places with CCTV compared to those without. Three of the 76 studies reported a statistically significant increase in crime. They go on to analyse the effectiveness of CCTV on different crime types. The meta-analysis showed that drug-related crimes decreased by 20% (six studies) and vehicle and property crime decreased by 14% (23 studies and 22 studies respectively) in places that had CCTV compared to those that did not. No overall statistically significant effect was observed for violent crime (29 studies) or disorder (six studies). Review one reported that CCTV did not overall lead to crime displacement – only six of the 50 studies that included adjacent treatment and control areas reported evidence of displacement.

Considering the aforementioned figures, we can identify the need for CCTV cameras in homes, shops and even on the city streets. However, "The CCTV footage from security cameras appears to be grainy and of low-quality because of the file resolution and compression, how it was recorded, and the cropping that usually occurs on such video files, among others." (Slaughton, 2022). This results in the CCTV footage used in court and shared online is often not good quality which can be difficult to identify individual people. This is a worry considering that modern-day smartphones and tablets have better quality cameras, not on zoom levels but also the technology behind taking photos and recording videos which do not reduce file resolution. Many businesses don't have high-end CCTV setups due to the high costs that even a sharp standard 4K camera can cost, anywhere from £350 up to the thousands depending on the number of cameras and maintenance involved. CCTV could just be bought as it has a deterrent effect in some places as having CCTV in place can make criminals uncomfortable about

performing illegal action as they may be aware. These types of cameras are usually low end only for deterring people and can often produce grainy footage which is quite difficult to use if a crime does happen.

This is an afterthought for most businesses because they don't see a return on investment in CCTV cameras compared to how often they get physically robbed or broken into. If businesses do want to investigate their footage but it is low quality then they may result in hiring an outside professional who can enhance the quality of the footage so objects appear sharper. Yet again, this requires costs.

Combining the rising rate of crimes and the exorbitant cost of CCTV cameras, it is inevitable that software and Artificial Intelligence can become the new commonplace due to the ease of access and often low price for licensing. As such, FaceCCTV aims to fill gaps in the market that other apps do not or simply do not include the whole package.

2.1.1. What is Face Detection in Artificial Intelligence?

Face detection or commonly known as facial recognition technology is a set of algorithms that work together to identify people in a video image or images themselves. Artificial Intelligence is usually integrated into facial recognition technology to use pre-trained images in a dataset to help the algorithms look for what they want, this technique is known as machine learning. The term "Artificial Intelligence" is a general subset within computer science using smart machines that can execute operations that require human intelligence, such as self-driving cars.

2.2. Existing Applications

With most software development, there is research required to investigate the current market to see what existing applications excel in and what they lack. With this data, the idea of developing a solution is to fill in the gaps where the market does not reach, normally features or user experience. Below are different existing solutions that are similar to what FaceCCTV offers to determine what functionalities they provide.

2.2.1. Functionality

The first thing that users find out about an application is what they are limited and not limited to doing within the application. This table provides a simple overview of what existing solutions have to offer.

	Application		
	Remini	PicTrev	Ribbet.ai
Functionality			
Photo Enhancer	✓	✖	✖
Face Attributes	✖	✖	✓
Face Detection	✖	✓	✓
Multi Image Upload	✖	✓	✖

Table 1: Functionality of existing solutions

2.2.2. User Findings

Despite having listed the functionality and features that current solutions provide, as a developer, it is oftentimes difficult to objectively judge and determine what the user wants. Consequently, an incredibly useful way to obtain valuable market research data is by reading the reviews that users have left on existing applications. By doing so, one can get a better understanding of what users looking for similar apps value. To compete with existing solutions, the new solution must offer an incentive for users to switch over. User reviews are perfect for identifying areas to improve upon. After analysing reviews

on all 3 applications, the key problem is that not a single application had all of the features listed in the above table, this means users would have to move across multiple tools to perform what FaceCCTV offers in one place. Most of the tools also require user login via an account from a third-party service such as Google or Microsoft which is unnecessary for a simple task, such as improving the quality of an image.

2.3. Limitations of Current Solutions

While studying the competition is invaluable, it is noteworthy to consider that FaceCCTV is based on an existing application that the developer aims to rebuild from scratch while adding a host of new features and requested functionality. The existing software is called Remini. By combining the feedback from Remini, with the user reviews of existing solutions, while also developing additional features with an improved and more performant tech stack, FaceCCTV can provide excellent value to its users.

3. Legal, Social, Ethical and Professional Issues

When developing software, it is imperative to consider the legal, social, ethical, and professional issues that it entails. Considering this project involves sensitive photos of the general public, it is important to research the relevant laws and ensure that, as an AI (Artificial Intelligence) Face Detection Tool, it complies with all rules and regulations.

3.1. Legal

As a result of the sensitive nature of anyone accessing footage of people without their consent, it is crucial to protect not only user data once stored (Data Protection Act, 2018) but also during transit. The relevant requirements of the Data Protection Act, in the case of FaceCCTV, include protecting user data against unauthorised access.

Furthermore, the regulation requires that any data collected be necessary and that such collection be transparent to the user. Moving on, users must also have the ability to delete their data following the “right to be forgotten”, as well as request it or download it as part of the “right to data portability” (GDPR, 2016). Also, the developer took into consideration that a tool like this might be used in law enforcement, especially in court where digital evidence is becoming more accepted so this has to perform at a standard where it would be accepted for modifying digital evidence.

3.2. Social

When using a face recognition tool such as FaceCCTV, numerous social issues come with it. The social impact of face recognition and finding out people can introduce insecurity and fear as anyone can use this app. With any app or tool, if it’s accessible to the public then it can be used for malicious behaviour, such as burglars finding vulnerable people to make a profit from.

3.3. Ethical

The majority of the ethical issues that concern this project are related to usability testing which will be carried out to prevent accessibility and usability issues when using the app. If the project gets to the point where the side objective of adding image colourisation is within the timeframe given then the developer has to take into account

that the image colourisation process will more than likely be performed by computers and AI. When using AI, there is a limitation to how much data it has been exposed to when it has been trained, as a result, unintentional stereotypes may appear and colourise sections incorrectly, most noticeably people of colour so this has to be taken seriously to avoid any misjudgement.

3.4. Professional

In addition to the statement of usability testing, a great deal of focus will be given towards maintaining the anonymity and confidentiality of participants, while respecting their rights by thoroughly explaining each session's activities and making sure they fully consent to participate.

Furthermore, as far as anonymity and confidentiality are concerned, the same approach will be taken when dealing with potential suggestions, questions, and bug reports from users, where they will be kept anonymous.

Additionally, another factor to consider would be the usage of copyrighted work during development, and the proper attribution of third-party services and/or libraries. To address this, all the libraries and external resources used throughout the software or otherwise will be credited in the "README" file of the version control repository.

4. Design

One of the key phases of the software development life cycle is the “design” phase, as it begins the process of turning a concept or idea into a tangible product. The process starts with the identification of user requirements. Properly understanding user requirements is an absolute necessity, and is critical to the success of any piece of software (Maguire and Bevan, 2002). User personas are also created, which are then used to derive the user stories. Once the functional and non-functional requirements have been identified, diagrams are used to visualize the components of the software and how they interact with one another. Finally, wireframes and mockups are created based on the requirements.

4.1. User Requirements

The user requirements of the product are based on three foundations:

- The sizable amount of feedback for FaceCCTV’s predecessor (Remini AI)
- The common shortcomings and complaints regarding similar applications.
- A user persona that shows the needs of police officers who require a tool to help their job.

4.1.1. User Persona

Paul works in the CCTV operating and security department of his local police force. He knows how a computer works and can operate around CCTV cameras, however, he’s not familiar with advanced skills such as photo editing or complex software. He spots a person in an image but can’t quite make out who it is. He requires a solution where he can upload the image to detect the same face and enhance it so it becomes clearer.

4.1.2. User Stories

As a police officer, he wants a solution that…

- Allows him to upload an image from the CCTV footage.
- Detects the faces in the image.
- Enhances them so he can identify them easier.
- Allows him to export the enhanced image for later use.

As a security specialist, he wants a solution that…

- Allows him to upload an image from his CCTV footage.
- Detects the faces in the image.
- Enhances them so he can identify them easier.
- Allows him to export the enhanced image for later evidence.

4.1.3. Functional Requirements and Product Backlog

Having identified the user requirements. It is then possible to create a product backlog, which lists the functional requirements (derived from the user stories) broken up into several sprints. Each sprint is 2 weeks long, these dates should be considered with 5-hour work days, 35-hour weeks, around 5 hours per day including weekends.

Sprint	Start	End	Tasks
Sprint 0	30/09/2022	13/10/2022	• UI Design Concepts • High-Level Planning • User Stories & Requirements
Sprint 1	14/10/2022	27/10/2022	• Setting Up The Environment • Gathering Test Data
Sprint 2	28/10/2022	10/11/2022	• Preparing the data • Choosing a model • AI Training
Sprint 3	11/11/2022	24/11/2022	• AI Evaluation • Parameter Tuning • Making Predictions
Sprint 4	25/11/2022	08/12/2022	• AI Face Detection Test • AI Image Enhancement • AI Websocket hosting • AI Testing
Sprint 5	09/12/2022	22/12/2022	• Web App / Desktop App Setup

			• Basic Implementation
Sprint 6	23/12/2022	05/01/2023	• Web App UML Class Diagram • Navigation & Common UI Elements • Image Upload / Download
Sprint 7	06/01/2023	19/01/2023	• Settings • Notifications • Encryption for Upload / Download
Sprint 8	20/01/2023	02/02/2023	• Mobile App • Core Components • Navigation
Sprint 9	03/02/2023	16/03/2023	• Implement AI • Image Upload / Download • Encryption for Upload / Download
Sprint 10	17/02/2023	02/03/2023	• Testing once Deployed

Table 2: Product Backlog

4.1.4. Non-Functional Requirements

Since the software is self-hosted, most of the non-functional requirements of the AI, such as speed, availability, capacity etc.., are entirely dependent on the device hosting it. But speed and usability are the most important factors on the client side.

4.2. Diagrams

Following the identification of the user requirements, the solution needs to be planned in detail. Using diagrams, it is possible to visualise exactly how the different parts of the system interact with each other, and what the structure of the codebase looks like.

4.2.1. Architecture Diagram

FaceCCTV would have a client-server architecture, as clients would not interact with one another, and would interact with a FastAPI RESTful API. Its architecture would be classed as a 3-Tier Architecture, with the front-end app hosted on a server but accessed anywhere, alongside the API and AI being stored on a separate server. The API would act as middleware between the front end and the AI model, handling image requests. It was decided that this would be the ideal architecture as the software aims to provide multi-user functionality, with users being able to upload images for the AI to give them a result.

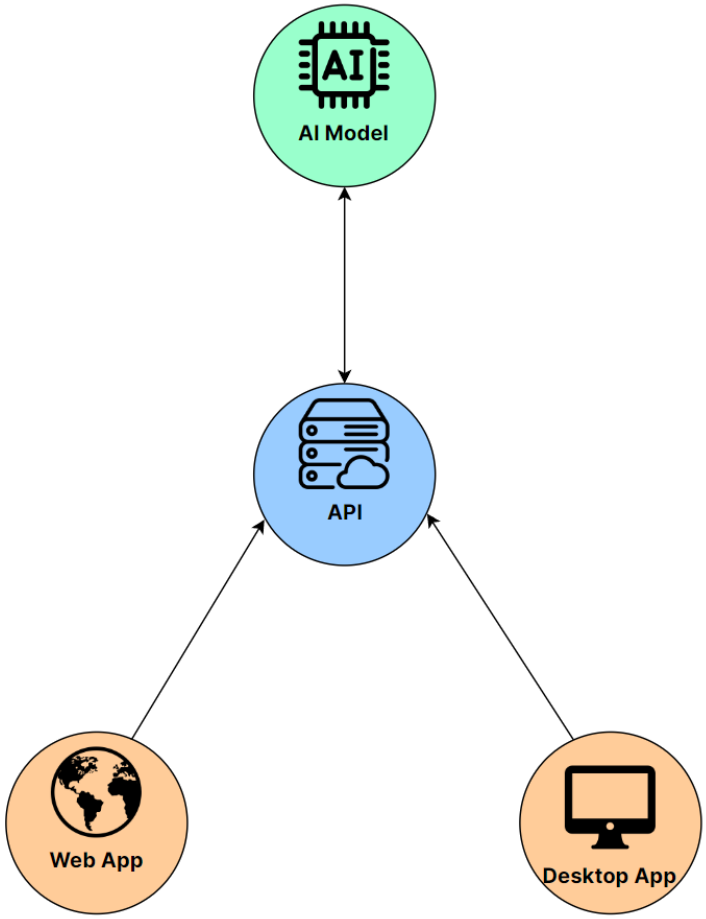


Figure 1: Architecture Diagram

The image above shows the architecture of the application. Users would be able to use any of the platforms to interact with FaceCCTV's API, which retrieves and sends images from the AI model.

4.2.2 Activity Diagram

An activity diagram has been provided below to visualise how the user would interact with the AI.

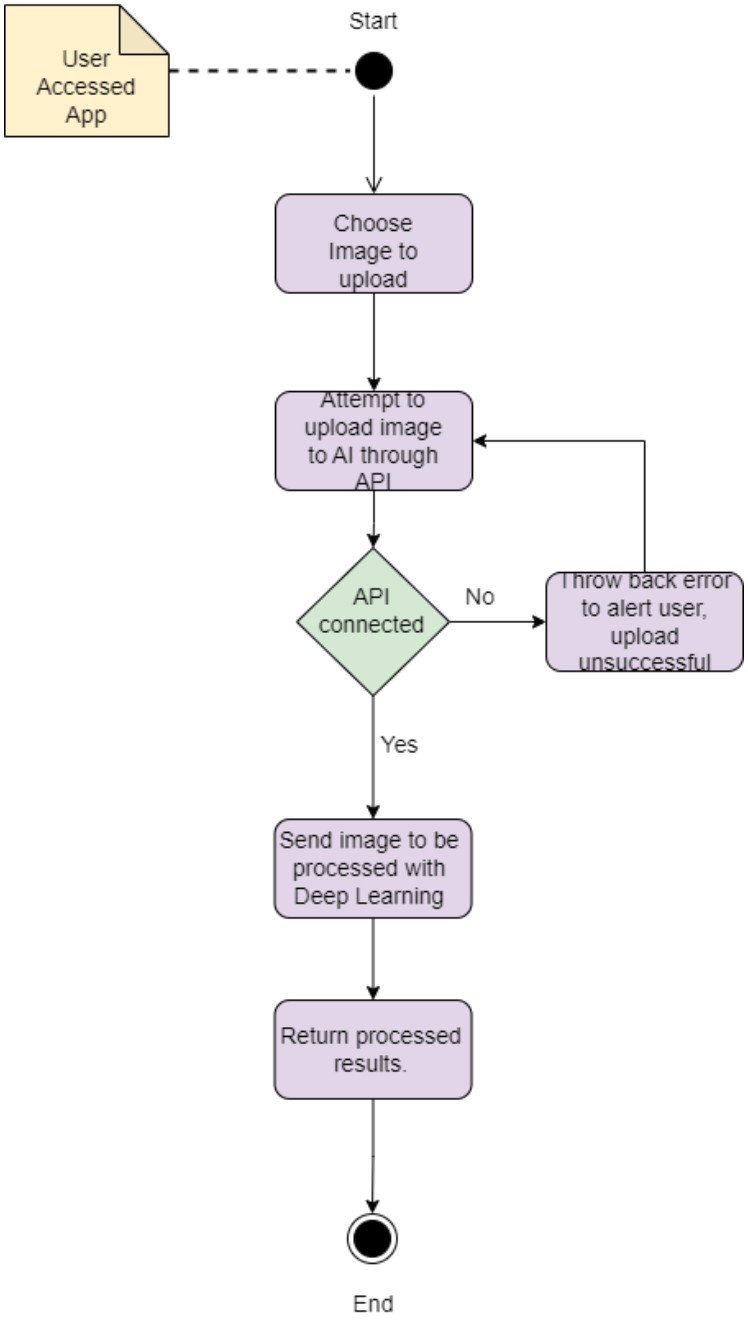


Figure 2: Activity Diagram

4.2.3. Sitemap

A sitemap has been provided below to visualise how the user would interact with the application to navigate to the different parts of the app

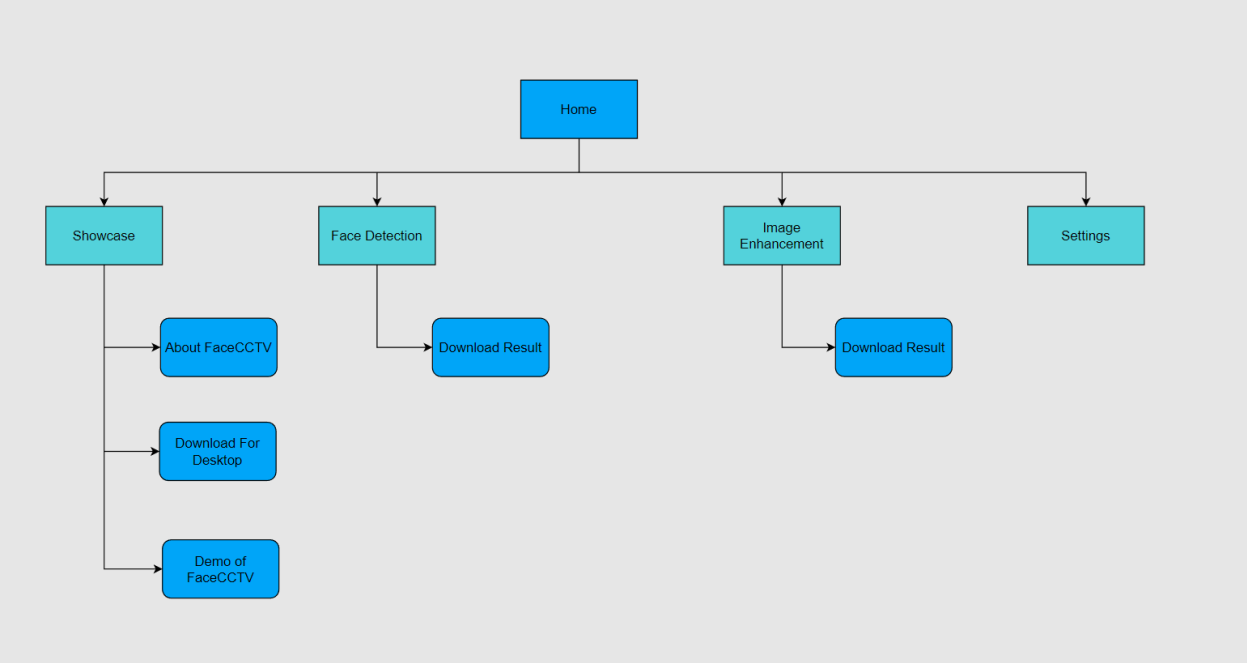


Figure 3: Sitemap

4.2.4. API Class Diagram

For the AI Model to be used externally outside of calling it in its Python file. A Python framework, FastAPI, was used to generate an API. As Python was used rather than, C# for a typical REST API, the class diagram had to be adapted to a more functional programming style.

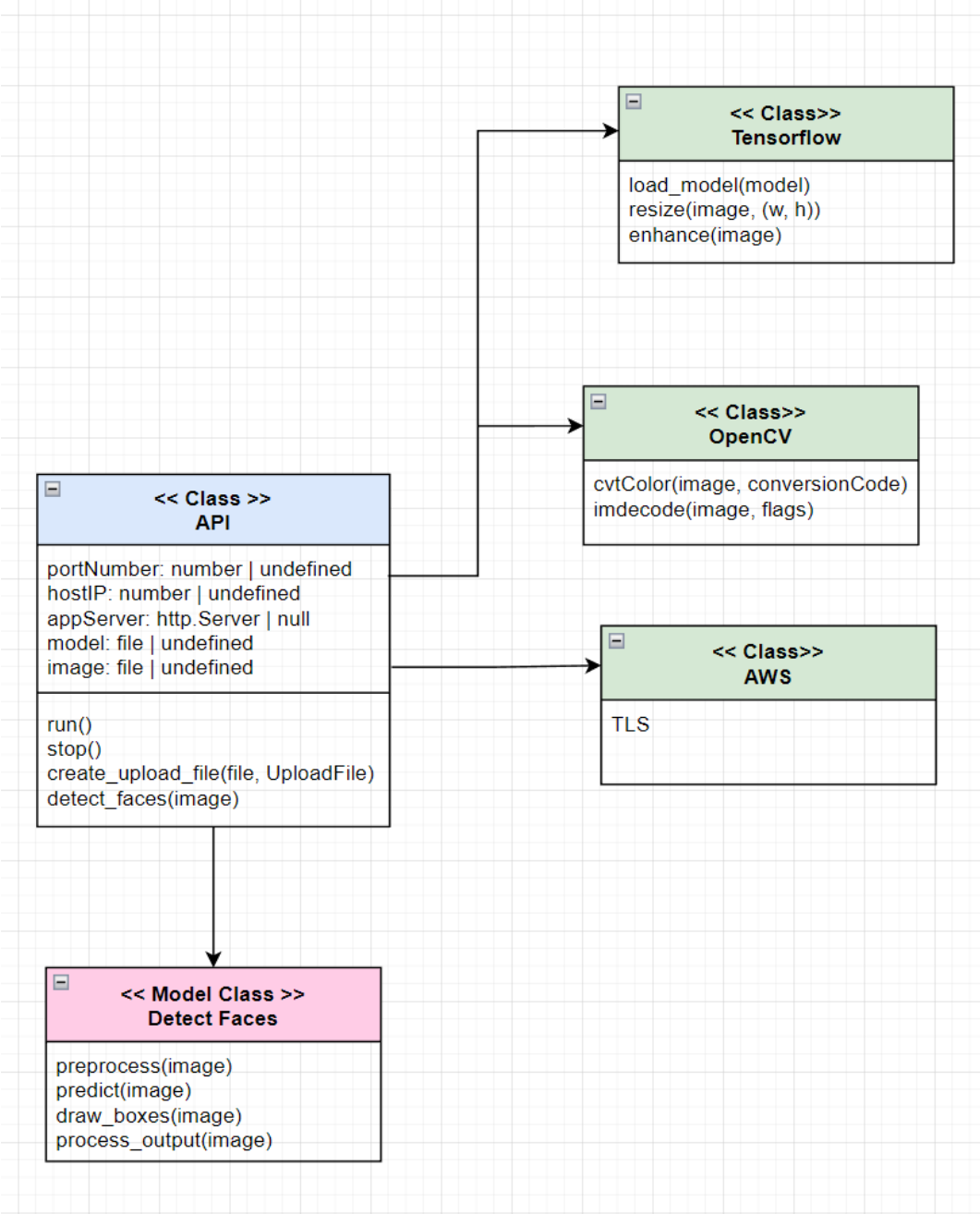


Figure 4: API Class Diagram

4.3. User Interface

Having identified the functionalities that the software would provide, it becomes relevant to design a user interface that lets users access the features listed in section 4.1.3.

4.3.1. Wireframes

Before designing detailed mockups of the user interface, wireframes were created. The first one below is the “Image Enhancement” page, and the second is the “Settings” page

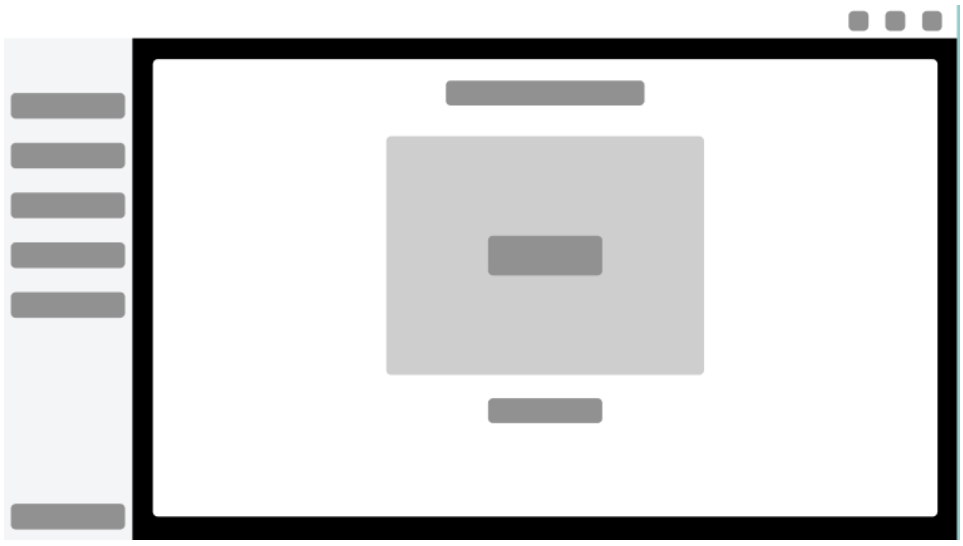


Figure 5: Image Enhancement Page Wireframe

Figure 6: Settings Page Wireframe

4.3.2. Mockup Designs

Using the wireframes as a foundation, detailed mockups were designed using Figma. The eventual user interface of the application was based on a pixel-perfect recreation of the mockups. Below, the “Image Enhancement” page and the “Notifications” page are shown.

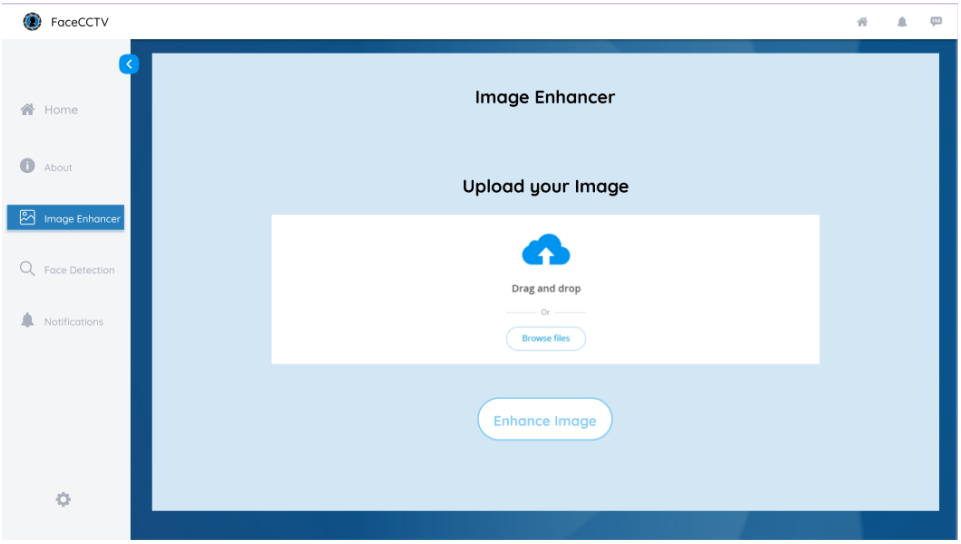


Figure 7: Image Enhancer Page Prototype

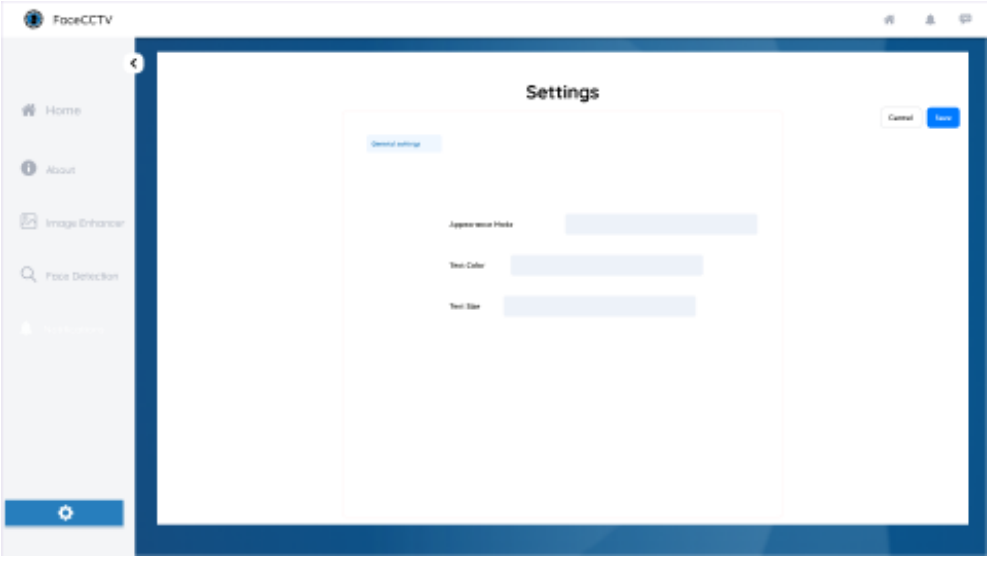


Figure 8: Settings Page Prototype

4.3.3 Final Designs

When it came to developing the final designs, an entirely new layout was built to maximise screen space for the necessary components. So the sidebar was converted to a top navigation bar to save vertical space and then the background went from blue and white to an interactive slider so the user can interact with it to see the process.

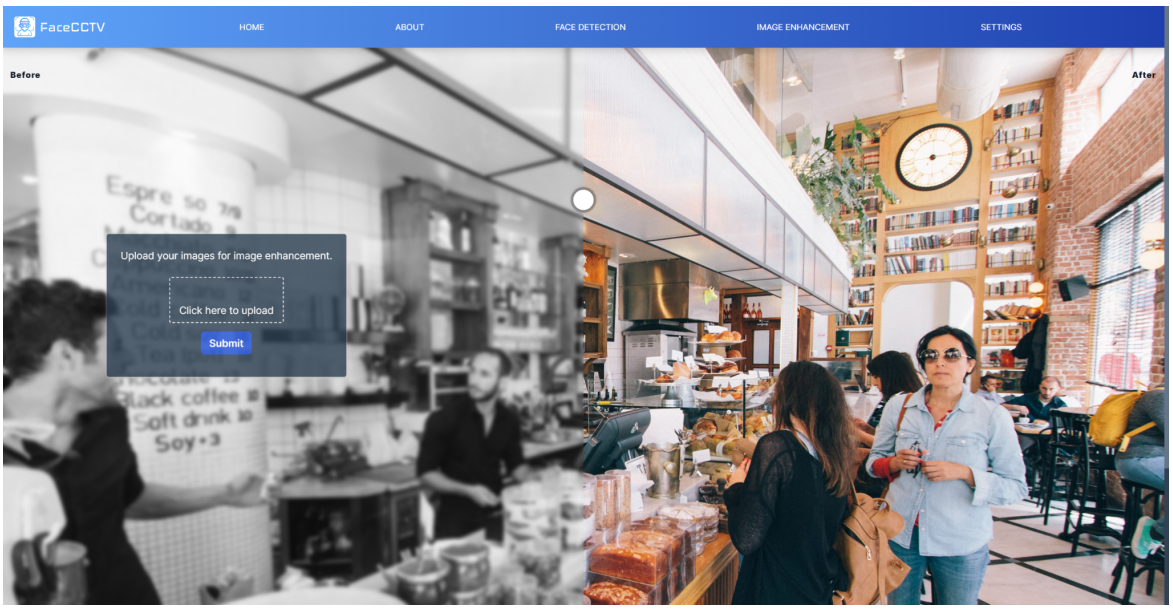


Figure 9: Final Image Enhancement Designs

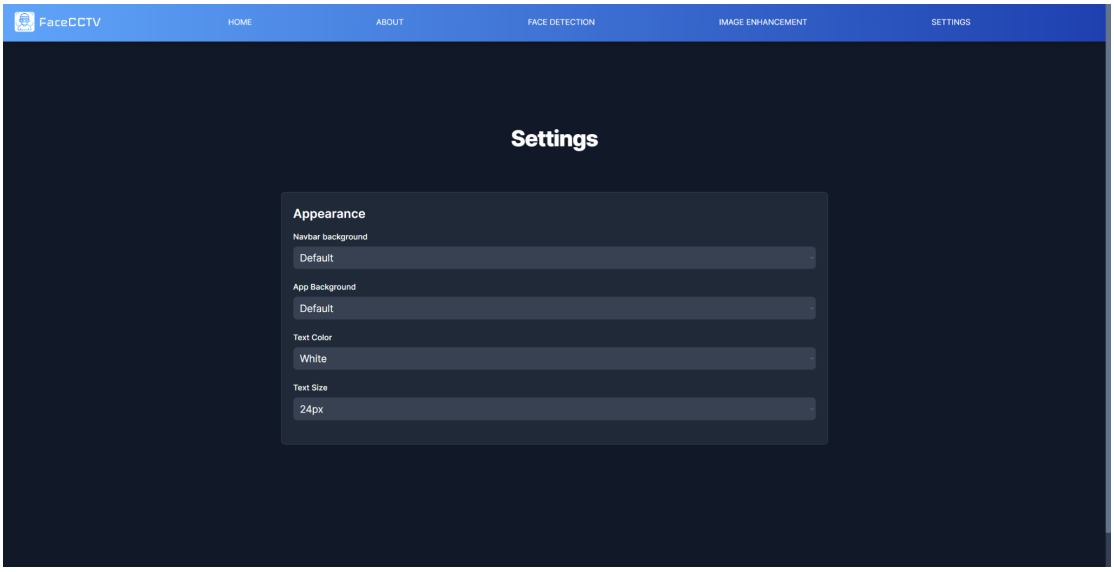


Figure 10: Final Settings Designs

5. AI Architecture

At a high level, this AI Model uses a type of machine learning algorithm called a deep learning Convolutional Neural Network (CNN). The idea behind using a deep learning model is that it has multiple layers of learning, similar to the human brain, allowing it to learn and become more intelligent from a large amount of data. Each layer learns a concept from said data that subsequent layers build on.

Usually in a deep learning CNN, particularly in any type of object detection, there are two models. One is a classifier model, where the model classifies what the object is within the image and each object usually has its class to define the object type e.g. Face. This is going to return a boolean answer, yes or no. Then the other model is a regression model which is a type of statistical model where it tries to estimate the relationship between two or more dependent variables.

FaceCCTV will use AI to detect if there's a face or more in the image using a classification model and using the regression model to estimate where the bounding box will go around the face using two opposite coordinates and the height and width.

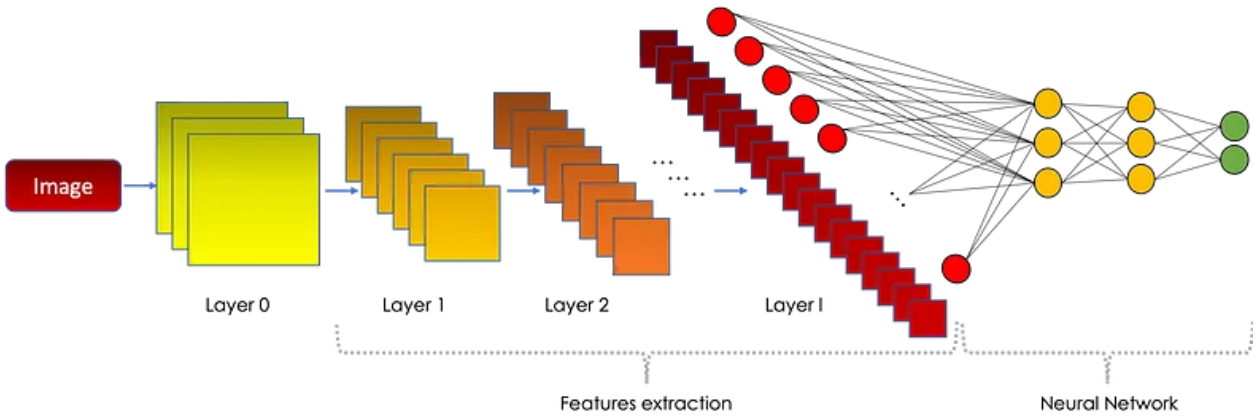


Figure 11: Deep Learning CNN Architecture

With a neural network, it is common to practise that certain types of loss functions should be created when designing the model. The classification model uses a loss type called binary-cross-entropy, also known as log loss. Which determines if the probability predicted for the output is closer to 0 or 1.

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i \cdot log(p(y_i)) + (1 - y_i) \cdot log(1 - p(y_i))$$

Figure 12: Binary-cross-entropy loss function

When it comes to the regression model, the developer is using a different type of loss called localisation loss which is estimating the loss of the predicted coordinates and size of the bounding box that should fit around the object.

$$L_{loc}(x, l, g, f) = \sum_{i \in POS} \sum_{m \in \{cx,cy,w,h\}} (x_{ij}^{p} smooth_{L1}(f_m(l_i) - \hat{g}_j^m))$$

Figure 13: Localisation loss function

Then TensorFlow's keras functional API will be imported to run the VGG16 model which has multiple layers to train on.

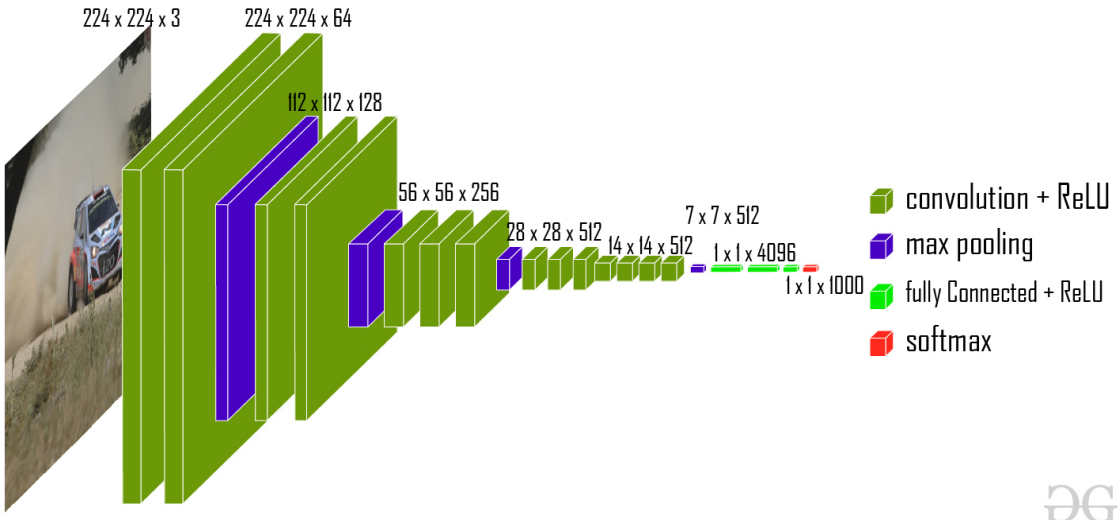


Figure 14: VGG16 Model layer diagram

6. AI Development Pipeline

6.1. Stage 1 - Design and Research

The project began with researching the fundamentals of machine learning and deep neural networks through numerous online papers and articles. Diving into the world of using Artificial Intelligence for Face Detection, the developer had a clear path to which the AI model was going to be developed. Before this project, the developer had limited subject knowledge, the model should be easy to debug and find errors whilst being important to understand how different components of the final model functioned in the real world. This stage also included high-level planning regarding the visuals that would represent the user interface of the front end that allowed users to interact with the AI.

6.2. Stage 2 - Data Collection and Pipeline

After the design stage, data collection was a priority for deep learning to take full advantage of. There are many datasets on the web used for popular services. The developer decided to use some existing datasets. One of them was the WIDER Face dataset which has been around since 2016 and has been one of the most popular benchmark datasets for face detection. The other was the FDDB (Face Detection Dataset Benchmark) dataset.

The WIDER Face dataset is around 3.5GB in size. It has 32,203 images and 393,703 labelled faces. These images have a high degree of variability, in scale, pose and occlusion across all 3 splits. The dataset is organised based on 61 “classes”, based on which scenario the picture was taken from (examples are: sports games or parades). The training, validation and testing splits were set from randomly picking images from each class into a 40-10-50 split into the respective set.

The FDDB (Face Detection Dataset and Benchmark) dataset has 2845 images and 5000 labelled faces. Similar to the WIDER face dataset, the dataset has images that have variability, in scale, pose and occlusion. However this dataset used 2 classes, no face and face to provide AI models simple classification.

Another dataset was used which was collected by the developer themselves, which was collected by using a Python script to extract each frame from a video exported from youtube of people in a variety of places with different lighting and distances which

equated to roughly 20,000 frames which were then processed in Adobe Photoshop to apply a CCTV Photo Effect to make the stock footage appear from a CCTV camera using Photoshop Automation Tool. The faces were labelled using a third-party tool called ‘labelme’ to draw bounding boxes around each face and put the bounding box details including 2 coordinates of opposite corners and the width and the height in a JSON file. The dataset had to be split by the developer themselves similar to the FDDB dataset which saw a 70-20-10 split in the training, testing and validation splits.

6.3. Stage 3 - Deep Learning CNN

As previously mentioned in greater detail in Section 5, The developer decided to use a Deep Learning CNN which is a type of deep learning algorithm. It is a type of algorithm that uses supervised loss within their training. CNNs are built to “automatically learn and synthesise a problem-specific feature extractor from a training set, without making any assumptions or using any hand-made design concerning the features to extract or the areas of the face pattern to analyse” - (Shu Zhan, Qin-Qin Tao, Xiao-Hong Li, 2016).

6.4. Stage 4 - Building the Neural Network

It was built using TensorFlow ‘’functional’ API using the library’s keras module where the developer used the pre-built VGG16 model as a base model to build our classification and regression models on top which then was tested before they created the losses to each model to determine if it had a good accuracy score.

6.5. Stage 5 - Training, Testing and Plotting Results

Training the model was done using two methods declared in a Python class object which was then extracted to plot the accuracy scores for each epoch over time. This process took the longest, most iterations took around a day or two due to the gigantic size of the datasets. After training in the testing phase, the results can be plotted on graphs using subplots then the AI model evaluates if it is an acceptable score to use later on. From the results gathered and numerous runs, the developer was getting accuracy scores of anything above 85% from early tests up to 95% nearer to development based on the configuration with the data used. Figures 15 to 17 shows the scores and loss functions while training with the WIDER Face dataset.

```
Output exceeds the size limit. Open the full output data in a text editor
Found 12880 images belonging to 61 classes.
Found 3226 images belonging to 61 classes.
Learning rate:  0.0001
Epoch 1/50
402/402 [==============================] - 2048s 5s/step - loss: 3.4029 - accuracy: 0.2127 - val_loss: 3.2255 - val_accuracy: 0.2494 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 2/50
402/402 [==============================] - 1893s 5s/step - loss: 2.5315 - accuracy: 0.3720 - val_loss: 3.0849 - val_accuracy: 0.2794 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 3/50
402/402 [==============================] - 1901s 5s/step - loss: 2.1335 - accuracy: 0.4685 - val_loss: 3.0414 - val_accuracy: 0.2872 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 4/50
402/402 [==============================] - 1902s 5s/step - loss: 1.8579 - accuracy: 0.5332 - val_loss: 3.0910 - val_accuracy: 0.2881 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 5/50
402/402 [==============================] - 1887s 5s/step - loss: 1.6312 - accuracy: 0.5957 - val_loss: 3.1313 - val_accuracy: 0.2922 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 6/50
402/402 [==============================] - 1961s 5s/step - loss: 1.4342 - accuracy: 0.6406 - val_loss: 3.0853 - val_accuracy: 0.3038 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 7/50
402/402 [==============================] - 1958s 5s/step - loss: 1.2736 - accuracy: 0.6829 - val_loss: 3.1275 - val_accuracy: 0.3069 - lr: 1.0000e-04
Learning rate:  0.0001
Epoch 8/50
...
402/402 [==============================] - 1778s 4s/step - loss: 0.0962 - accuracy: 0.9818 - val_loss: 3.8485 - val_accuracy: 0.3066 - lr: 5.0000e-05
Learning rate:  5e-05
Epoch 50/50
402/402 [==============================] - 1778s 4s/step - loss: 0.0941 - accuracy: 0.9821 - val_loss: 3.8852 - val_accuracy: 0.3103 - lr: 5.0000e-05
```

Figure 15: Training epoch progress and scores

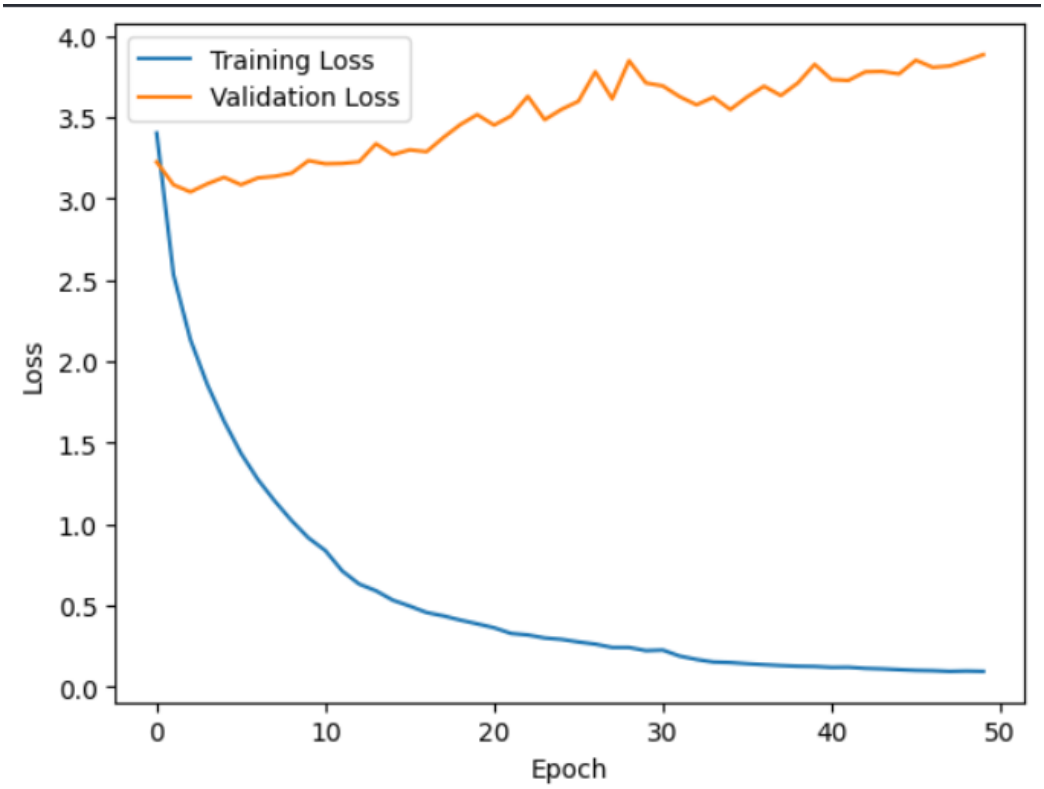


Figure 16: Epoch loss scores

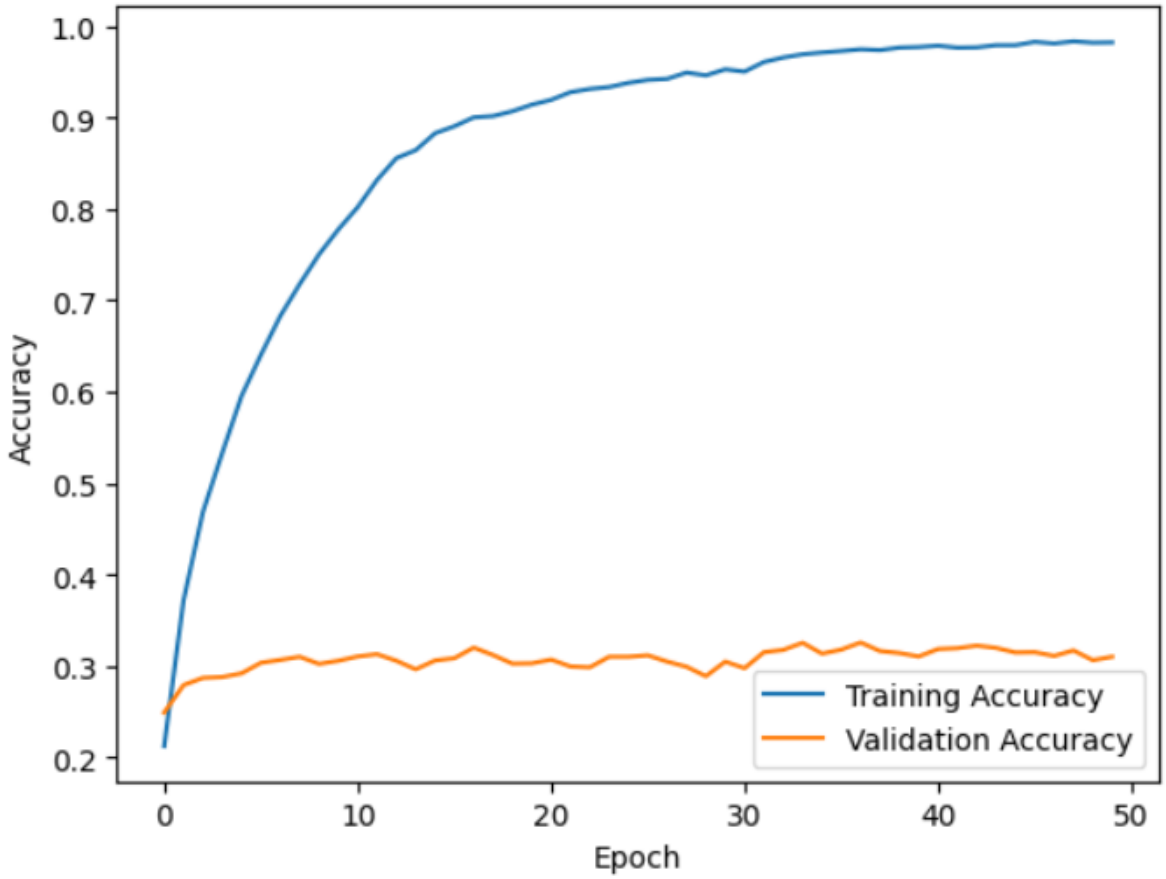


Figure 17: Epoch Accuracy Scores

6.6. Stage 6 - Deploying AI

After the AI model was trained, its configuration was saved as an singular H5 file which is when data is saved in the hierarchical data format which is a multidimensional array to compensate for images having multiple dimensions then later on the developer can load the model and then the program can predict the outcomes from different sets of test data.

7. Development Approach

Once the functional and non-functional requirements were clear, and mockups of the user interfaces were designed, choosing a tech stack became a much simpler task. “When choosing a technology stack for app development, you first need to rely on the requirements of your project” (Puzhevich, 2020). After carefully considering the user requirements, the tech stack described below was chosen.

7.1. AI Model

7.1.1. Face Detection

The main concept for FaceCCTV is to detect faces automatically through Artificial Intelligence, more specifically Deep Learning. The ideal programming language was Python. The reason is that it has a huge community relating to developing machine learning projects due to the ever-expanding range of libraries that allow people to execute tasks easier than in other languages. More specifically, it was coded in jupyter notebook so that the developer could test parts out in a non-linear fashion rather than having to run the whole program which made debugging errors easier to find in the long run.

7.1.2. Image Enhancement

After FaceCCTV detects faces in the photos, the developer used several python libraries such as OpenCV and Pillow to implement modules to perform image enhancement techniques such as unblurring and sharpening the images. Alongside using a pre existing CAFFE model, to colourise the image. This was coded in the API script rather than in its own python script to save confusion and have easier access if the API endpoints did not work..

7.2. API

The API will serve as a web host for the AI, meaning the app can communicate with it. So Python’s FastAPI library was selected to create a RESTful API to allow the app to receive and send images to the AI. We are hosting the API through an Amazon web service EC2 instance, to be configured with HTTPS and SSL so the images will be

uploaded and downloaded with encryption whilst being server-side, allowing access publicly.

7.2.1. FastAPI

The justification for using a newer API Framework such as FastAPI rather than a very established framework such as Django has its pros and cons. As the developer is dealing with an AI Model retrieving and sending images, FastAPI is a lightwork framework compared to Django so it offers very high performance. FastAPI also offers flexibility - the code layout has the freedom and doesn't have to be developed in a certain layout which means the learning curve is less steep. The only downside to FastAPI is being a newer framework which means the documentation support is not as thorough as Django's so it means a lot more interaction when it comes to finding the optimal solution.

7.3. Client-Side Development

7.3.1. Web App

The web app will be developed in HTML, CSS and JavaScript with Tailwind CSS. There are no other frameworks or templates as the user interface will be easier to maintain if all the platforms use compatible frameworks. Tailwind CSS is a custom CSS framework which offers great support for multiple platforms and frameworks so it felt like the safest option to go with while adding a modern feel to the user experience. The web app will be hosted on a server rather than locally through a domain provider and the hosting is done through Google Firebase. This will help non-technical users undertake the usability test study due to the requirements of installing multiple frameworks and dependencies.

7.3.2. Desktop App

Through Electron, the desktop app can have the same codebase as the web application. This would simplify debugging and testing, while also providing users with software that feels native to the system. Additionally, it offers cross-platform support, automatic updates, and increased development speed and productivity. Its success has resulted in popular applications such as Discord, Slack, Atom, Trello, and others using it.

7.4. Security

FaceCCTV is dealing with sensitive images that include CCTV footage of people who may not know they are being used in this tool, without some sort of security mechanism could lead to information misuse. So several security practices have been put in place to protect user data.

7.4.1 HTTPS and SSL Certificates

Using SSL Digital Signature Certificates will ensure that an internet connection between the web address and the client's device is always secure and safeguards any sensitive data that is sent between the web app and the client. This prevents criminals from hijacking and modifying information while transitioning from the device to the app or vice versa. HTTPS nowadays is the bare minimum of web security, HTTPS is the primary protocol used to send data between a web browser and a website. HTTPS is encrypted in order to increase security of data transfer.

```
ssl on;
ssl_certificate /etc/nginx/ssl/server.crt;
ssl_certificate_key /etc/nginx/ssl/server.key;
```

7.5. Development Tools

The complete list of the tools used to design, implement and produce the project.

- Visual Studio Code was chosen as the code editor, as it has great support to allow developers to add 'extensions' to allow them to code in any language.
- Figma was used as a design prototype tool where everything from the initial wireframes up to the latest mockup designs before they were transferred into code.
- Adobe Color was used to test different colour schemes to check if they were pleasing to the user and also if they were reasonable for people who are colour-blind.
- GitHub Desktop will be utilised to interact with the VCS (Version Control System).

- Google Chrome, Firefox, and Microsoft Edge will be used to test the web application on different browsers.
- Photoshop will be used to design many of the assets that will be used in the final project.

7.6. Version Control System

Through the use of a VCS, it is possible to keep track of changes in the codebase, while always having a previous point to revert to should a new but broken implementation be too complicated to reverse manually. Considering the complexity and codebase size of FaceCCTV, the features mentioned above are invaluable. Furthermore, the GitHub repository acts as an off-site backup should any of the devices used during development stop working. Regular commits ensure no work is ever lost, and that the project can be accessed on any device that supports Git.

7.7. Hosting and Production

Through the use of Firebase hosting and AWS respectively, the front end of the web app and the API are hosted on the web so the usability testing can be undertaken easier. The electron app is published and can be downloaded through the web app and installed through an exe to be run on the user’s computer.

8. Project Management

Competent project management can be the difference between a project succeeding or failing (Gido and Clements, 2014). As such, the project's schedule and sprints needed to be planned only after it was clear what the user requirements were, what the design of the application would look like, and the development approach that would be taken to achieve the objectives of the project. By having the aforementioned information before any detailed planning, it was possible to take into account the developer's experience with the technology, the complexity of the software and codebase, as well as the potential obstacles that could present themselves throughout the project.

8.1. Risk Assessment

The development of a minimum viable product should not involve any identified risks. However, the risk level will increase as the project becomes more complex. At this point, the developer is very familiar with front-end development so there's a lot of variety that they could use. On the other hand, when it comes to the AI functionality of the project some techniques will be used that they are less knowledgeable about.

In preparation for this project, research has been thoroughly carried out, such as what frameworks and languages to use as a backup plan in the scenario where it becomes difficult to execute the required aspects with the initially intended tech stack.

Risk	Likelihood	Impact	Score	Level	Solution
AI not performing optimally	4	4	16	Very Critical	The AI should be given enough test data to train so that the error rate is reduced to a reasonable percentage.
Poor Time Management	3	3	9	Critical	To make use of the Gantt Chart created to track the sprints and time management relative to the progress made.

Risk	Likelihood	Impact	Score	Level	Solution
Performance Quality	3	3	9	Critical	Making sure that the code is written efficiently enough to ensure the tasks of the software are executed without performance issues.
Tech Stack Unfamiliarity	3	3	9	Critical	Research the different technologies, read the various documentation and if not then have additional plans to use other familiar technologies.
Understanding the complexity of the project	3	2	6	Medium	Make sure that the goal(s) of the project are clearly understood without going too far out of scope and sticking to the requirements based on the user stories.
Data Loss	1	4	4	Medium	The code and the project's assets should be regularly committed to a version control environment.

Table 3: Risk Assessment

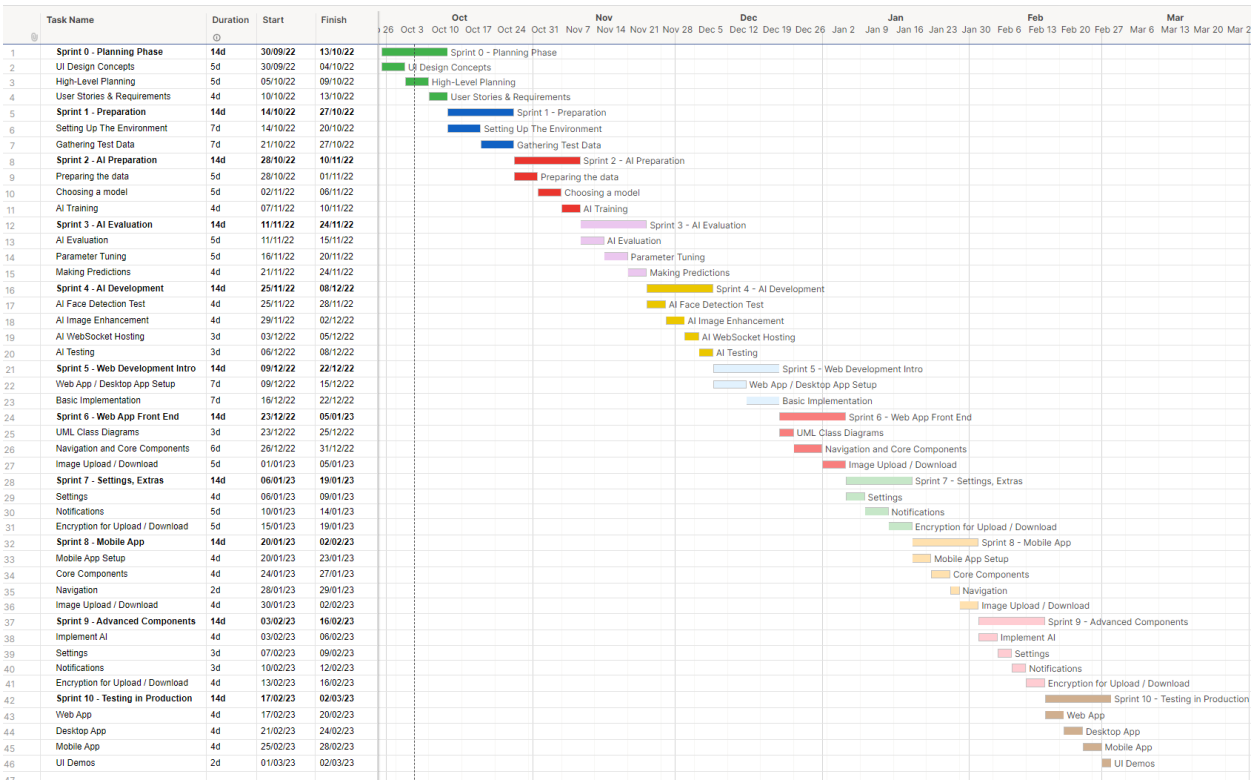
The table directly below is a risk matrix which presents how the developer calculated a risk score for each risk to the project.

		IMPACT						
LIKELIHOOD	Risk Matrix	1	2	3	4	5		KEY
	1	1	2	3	4	5		Low
	2	2	4	6	8	10		Medium
	3	3	6	9	12	15		Critical
	4	4	8	12	16	20		Very Critical
	5	5	10	15	20	25		

Table 4: Risk Matrix

8.2. Gantt Chart

A detailed Gantt Chart was put together in Sprint 0, to essentially plan out the schedule of the project. The chart, along with the product backlog shown in section 4.3.1, was used to help to develop the project in a structured manner whilst allowing agile changes to happen if something didn't happen accordingly.



	Task Name	Duration	Start	Finish
1	**Sprint 0 - Planning Phase**	**14d**	**30/09/22**	**13/10/22**
2	UI Design Concepts	5d	30/09/22	04/10/22
3	High-Level Planning	5d	05/10/22	09/10/22
4	User Stories & Requirements	4d	10/10/22	13/10/22
5	**Sprint 1 - Preparation**	**14d**	**14/10/22**	**27/10/22**
6	Setting Up The Environment	7d	14/10/22	20/10/22
7	Gathering Test Data	7d	21/10/22	27/10/22
8	**Sprint 2 - AI Preparation**	**14d**	**28/10/22**	**10/11/22**
9	Preparing the data	5d	28/10/22	01/11/22
10	Choosing a model	5d	02/11/22	06/11/22
11	AI Training	4d	07/11/22	10/11/22
12	**Sprint 3 - AI Evaluation**	**14d**	**11/11/22**	**24/11/22**
13	AI Evaluation	5d	11/11/22	15/11/22
14	Parameter Tuning	5d	16/11/22	20/11/22
15	Making Predictions	4d	21/11/22	24/11/22
16	**Sprint 4 - AI Development**	**14d**	**25/11/22**	**08/12/22**
17	AI Face Detection Test	4d	25/11/22	28/11/22
18	AI Image Enhancement	4d	29/11/22	02/12/22
19	AI WebSocket Hosting	3d	03/12/22	05/12/22
20	AI Testing	3d	06/12/22	08/12/22
21	**Sprint 5 - Web Development Intro**	**14d**	**09/12/22**	**22/12/22**
22	Web App / Desktop App Setup	7d	09/12/22	15/12/22
23	Basic Implementation	7d	16/12/22	22/12/22
24	**Sprint 6 - Web App Front End**	**14d**	**23/12/22**	**05/01/23**
25	UML Class Diagrams	3d	23/12/22	25/12/22
26	Navigation and Core Components	6d	26/12/22	31/12/22
27	Image Upload / Download	5d	01/01/23	05/01/23
28	**Sprint 7 - Settings, Extras**	**14d**	**06/01/23**	**19/01/23**
29	Settings	4d	06/01/23	09/01/23
30	Notifications	5d	10/01/23	14/01/23
31	Encryption for Upload / Download	5d	15/01/23	19/01/23
32	**Sprint 8 - Mobile App**	**14d**	**20/01/23**	**02/02/23**
33	Mobile App Setup	4d	20/01/23	23/01/23
34	Core Components	4d	24/01/23	27/01/23
35	Navigation	2d	28/01/23	29/01/23
36	Image Upload / Download	4d	30/01/23	02/02/23
37	**Sprint 9 - Advanced Components**	**14d**	**03/02/23**	**16/02/23**
38	Implement AI	4d	03/02/23	06/02/23
39	Settings	3d	07/02/23	09/02/23
40	Notifications	3d	10/02/23	12/02/23
41	Encryption for Upload / Download	4d	13/02/23	16/02/23
42	**Sprint 10 - Testing in Production**	**14d**	**17/02/23**	**02/03/23**
43	Web App	4d	17/02/23	20/02/23
44	Desktop App	4d	21/02/23	24/02/23
45	Mobile App	4d	25/02/23	28/02/23
46	UI Demos	2d	01/03/23	02/03/23

Figure 18: Gantt Chart

Bigger Version: [link here]

8.3. Kanban Boards

Throughout the development phase of the project, a software management concept known as a “Kanban Board" was used. ClickUp provides access to said concept which can be used to separate tasks from the product backlog into 3 columns, usually called “To Do”, “In Progress” and “Done”. While it seems like a simple concept, it massively increases productivity and keeps the development on track to the sprint plans that were set.

Additional categories were made to determine “bugs”, “issues”, “improvements”, “requirements” and “stuck”. The “complete” column differs from the “Done” category as this is when the developer can review the task in the “Done” column and the task would get marked complete if they were happy with the standard it was performed to. This was used to keep track of smaller tasks within the general sprint plans. Also, additional tags were used to keep track of which tasks are for what part of the project. This list contains “Desktop”, “Web app”, “Mobile”, “AI API”, “AI Model”, “Planning”, “Documentation”, and “Testing”. These tags can be filtered to view certain Kanban boards. Individual Kanban boards can be seen in Appendix 1.

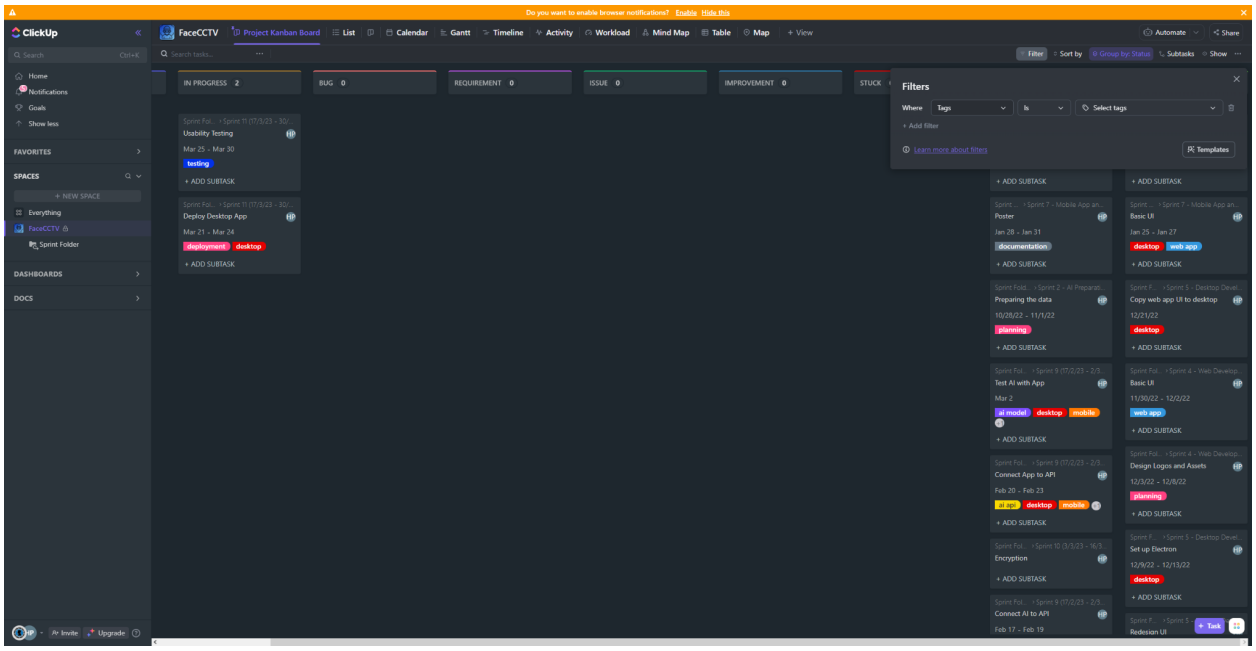


Figure 19: Kanban Board Example

9. Development

This project involved building the system with the chosen tech stack based on user requirements and mockups, while UML diagrams served as a reference so that the developer knew exactly what the software consisted of; this ensured that good software engineering practices were followed.

9.1. Initial Challenges

There were two major obstacles required a solution before development could begin as they posed potentially severe limitations on the project:

1. The AI wasn't detecting faces in photos where there was a clear face (whether it was a small image itself or the faces were small on a large image, for example, a crowd).
2. The WIDER Face Dataset annotations originally were in a different file format than the JSON format in which the developer's self-collected images were created.

To resolve the aforementioned issues, the following ideas were thought of and eventually implemented:

1. This was fixed by using more training data that related to the issue so the AI could learn through multiple training sessions in various situations. On top of this, the developer made sure that each image when in use in the API was preprocessed properly to a standard that the AI was used to its full potential, the use of a function from the OpenCV Python Library was used to resize and store each image in the correct format that the AI could use without losing quality.
2. In the jupyter notebook, a universal function was created that could work with any dataset so that the annotations could be interpreted properly whether they were JSON files or text files.

9.2. Sprints

The schedule of the project consisted of 15 sprints in total, all of which will be evaluated and discussed below. Each sprint's duration was 2 weeks, with 7-hour work days and 35-hour weeks. Please note that the tables below contain summaries of each sprint's outcome and that Appendix 2 goes into further technical detail regarding the work that

was carried out, the challenges that were faced, as well as the solutions used to overcome them

Sprint 0	**Start: 30/09/2022**	**End: 13/10/2022**
Tasks	UI Design Concepts High-Level Planning User Stories and Requirements	
Outcomes	Sprint 0 marked the initiation of the project, so the tasks mostly involved setting things up and getting ready to start development. During this sprint, a lot of research was done on the potential technologies that could be used. The project was also carefully planned out. Both of these activities proved to save a lot of time and trouble later on, as compatibility between platforms was planned from the start and did not become an issue at any point. The tasks in this sprint were completed earlier than expected, and no challenges were faced.	

Sprint 1	**Start: 14/10/2022**	**End: 27/10/2022**
Tasks	Setting Up Development Environment Gathering AI Test Data Setting up Github Repository and workflow directories	
Outcomes	This sprint entailed setting up the structure of the working environment and GitHub repository so it can be efficiently used while developing later. Also gathering the data for the AI's training, testing and validation splits were collected without issues.	

Sprint 2	**Start: 28/10/2022**	**End: 10/11/2022**
Tasks	Preparing the data Choosing an AI Model AI Training	
Outcomes	During this sprint, the deep learning CNN was chosen and starting to take shape after the data was prepared into a format which would be ready to be trained with.	

Sprint 3	Start: 11/11/2022	End: 24/11/2022
Tasks	AI Face Detection AI Data Loading AI Model AI Predictions	
Outcomes	The main bulk of the AI development was completed here without issues.	

Sprint 4	Start: 25/11/2022	End: 08/12/2022
Tasks	AI Face Detection Testing AI Image Enhancement AI Testing	
Outcomes	This sprint was moved to sprint 9 or 10 depending on how sprints 5-8 went. This was due to the supervisor wanting a visual representation of what the progress was looking like and the developer is experienced in developing websites from previous projects so it would be beneficial from a far point of view to make more progress quickly this way rather than struggle with the AI which the developer has less experience with.	

Sprint 5	Start: 09/12/2022	End: 22/12/2022
Tasks	Web App / Desktop App Setup Basic Implementation	
Outcomes	This sprint went as planned as the earlier prototype designing helped along with the developer's experience building software.	

Sprint 6	Start: 06/01/2023	End: 19/01/2023
Tasks	Navigation and Core Components Image Upload / Download	
Outcomes	This sprint was delayed by 2 weeks due to the developer having COVID-19. However, this sprint went as planned as the earlier prototype designing helped along with the developer's experience building software.	

Sprint 7	Start: 20/01/2023	End: 02/02/2023
Tasks	Settings Notifications Encryption for uploading and downloading	
Outcomes	This sprint went as planned as the earlier prototype designing helped along with the developer's experience building software.	

Sprint 8	Start: 03/02/2023	End: 16/02/2023
Tasks	Mobile app Setup Core Components Navigation Image Upload / Download	
Outcomes	The developer decided to discard the idea of the mobile app. This was decided after they discovered some issues with testing the AI so the new sprints 8-9 were gathering more sufficient data as the AI wouldn't register a face in some photos when there was a clear face.	

Sprint 9	Start: 17/02/2023	End: 02/03/2023
Tasks	Implement AI in mobile app Settings Encryption	
Outcomes	Same with the previous sprint, the mobile app development was not marked as important to make way for the necessary components to have a suitable product ready for usability testing.	

Sprint 10	Start: 03/03/2023	End: 16/03/2023
Tasks	API Development Hosting the web app Hosting the API	
Outcomes	This sprint went as planned as the developer used FastAPI which was all written in one Python file which could then be hosted and deployed on a VM server using AWS, which did cause some initial issues. Compared to outsourcing the web and desktop apps which were completed without any difficulties due to the developer's experience.	

Sprint 11	Start: 17/03/2023	End: 30/03/2023
Tasks	Web App testing Desktop testing UI Demos	
Outcomes	This sprint was moved to sprint 9 or 10 depending on how sprints 5-8 went. This was due to the supervisor wanting a visual representation of what the progress was looking like and the developer is experienced in developing websites from previous projects so it would be beneficial from a far point of view to make more progress quickly this way rather than struggle with the AI which the developer has less experience with.	

Sprint 12	Start: 31/03/2023	End: 12/04/2023
Tasks	Usability Design Feedback Final AI Model Training	
Outcomes	This sprint went as planned by the developer, as they received feedback well and made improvements on the areas that were lacking. issues. The developer trained the AI Model on more training datasets. This was a long process due to the vast amount of data and the size of the model.	

Sprint 13	Start: 13/04/2023	End: 26/04/2023
Tasks	Final touches	
Outcomes	This sprint went well as planned because the task was to tidy up the web app to make sure it was accessible for all users.	

Sprint 14	Start: 27/04/2023	End: 10/05/2023
Tasks	Final Usability Testing Viva	
Outcomes	This sprint was the final sprint of the software development life cycle where the developer was making sure that everything was working successfully and deployed correctly to show to other developers and clients.	

Tables 5-19: Sprint Reviews

10. Testing and Quality Assurance

A collection of quality assurance and testing methods were used to ensure the project and its codebase was at a commercial level, and ready to be pushed to production and distributed to be utilised by real users. Due to the developer's previous experience with deploying applications to the web, as was the case with FaceCCTV's predecessor, judging the performance and quality of the application was easier. Despite this, usability tests allowed for a more objective evaluation.

10.1. Usability Testing

Usability tests were carried out at the end of the development phase, which was the first to involve any user interface development. Participants were asked to complete a variable number of high-level tasks related to the features that were added to that sprint. Any problems identified during the session were added to the kanban board on Clickup as 'improvements' or 'issues' and were either resolved as soon as possible or at the end of the subsequent sprint depending on the priority of the issue. The tests were carried out in the user's spare time where they had to complete a Google form rating the user interface and experience as well as certain aspects, rating out of 5. It also included a long feedback section where the user can give feedback that can not be given with numbers.

To comply with the codes of ethical conduct, participants were asked for consent before starting the session, and any notes taken did not include any information that could identify the participants. A total of 10 participants took part in the test. During each test, each participant used the web app. The API and web app were hosted for them, and they were given a URL they could access without having to install and set up Node.js and pip for themselves.

11. Project Closure

“Project closure is a time to take stock, assess the successes and failures, and take learnings that you can transfer to future projects” (Rowland, 2019). Indeed, evaluating the performance and state of a project is invaluable when it comes to actually learning and improving one’s abilities.

11.1. Aims and Objectives Review

The overall goal of FaceCCTV was to provide its users with a tool that could allow them to upload images and receive an output where the image was enhanced in quality and the faces were detected by AI. While this goal was achieved, it is important to revisit the objectives that were identified at the start of this report.

- Objective 1: Research the market and feedback obtained from FaceCCTV’s predecessor to help identify user requirements.
    - As seen in sections 2 and 4, this objective was accomplished, and key missing features were found in existing solutions. Furthermore, FaceCCTV's predecessor was used to gain insight into what users want in a financial portfolio application.
- Objective 2: Research the legal, social, ethical and professional requirements.
    - This objective was met in section 3 by researching the relevant laws and regulations surrounding handling CCTV footage. The social, ethical, and professional implications of the project were also explored and discussed.
- Objective 3: Develop a production-ready, free, and hosted unified web, and desktop tool, for anyone to use anywhere.
    - Section 9 shows that this objective was met, and a fully working application was developed for every intended platform.

11.2. Project Evaluation

As part of the project evaluation stage, the development approach, including the technologies and tools used to develop the project, will be evaluated. Additionally, the project management approach and its general direction will be discussed to identify its performance. Finally, a personal reflection will be provided to explore some personal opinions regarding the project, as well as any meaningful learnings throughout the months.

11.2.1. Tech Stack Evaluation

A great deal of research and thought was put into the development approach chosen for this project, including the tech stack. Intending to create a unified interface, having a tech stack with highly compatible technologies was an absolute necessity. This eventually proved to provide a significant boost in productivity and efficiency, as many of the functions written for one platform could simply be used in the application of another platform without having to rewrite the code. This not only sped up the development process but also debugging and testing. The choice to avoid templates and frameworks for the web app and, by extension, the desktop app, also proved to be the ideal approach, as a custom user interface could be created from scratch that would be uniform across all platforms.

11.2.2. Project Management Evaluation

Despite every sprint being completed in time without delays, it is worth pointing out that the project's schedule and workload were incredibly demanding. A conscious decision was made to plan the project only after having decided precisely what development approach and which technologies would be used, and what the requirements were. This decision was, perhaps, one of the only reasons the project was completed on time.

By having a clear understanding of the workload, it was possible to better plan the flow of the project and to have a realistic estimate of how long each task would take. Additionally, the use of Kanban Boards made it possible to decompose the problem into smaller parts that could be tackled without being overwhelming. The flexibility and iterative nature of the Agile methodology also made it easy to rectify any issues with the software, and to have a working application at the end of every sprint; this also made it simpler to explain how much progress was made during each standup meeting with the supervisor of the project as shown in Appendix 6. Looking at the commit chart provided by GitHub in Figure 20, it is possible to see that the project was worked on regularly, with sprints involving the majority of client-side functionality having the most commits (Late January to early February) with the AI model coming in a close second (November through to December):

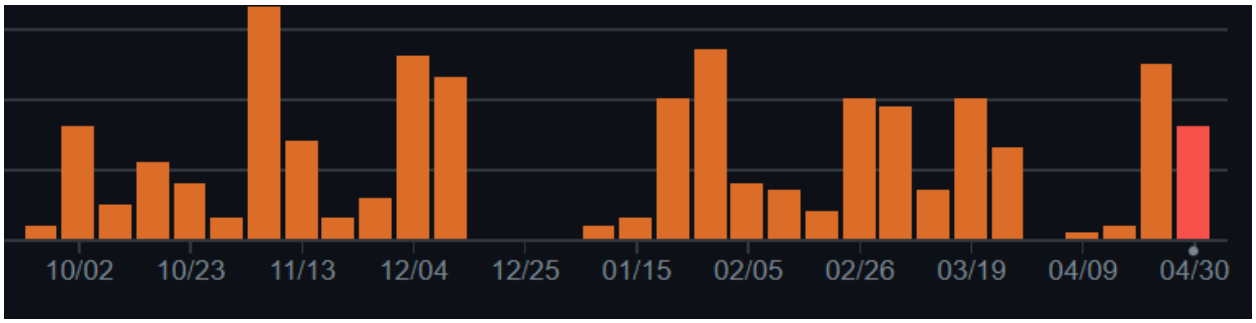


Figure 20 - Commit Frequency

11.2.3. Personal Reflection

I can say without a doubt that this project was one of my best learning experiences during my time studying computing. Before starting my university education I had no programming experience. Taking a foundation year degree in computer science helped me get up to speed. I found I enjoyed studying and it reaffirmed my aspiration to develop my IT skills. My aspiration was to develop software to protect people in the cyber world as well as the physical world by using the skills and tools I had learnt.

During my time at university, I have become more experienced and enjoyed developing software, especially front end. However, every project had always been rather disorganised and not planned, whereas this one was fully planned about 2 weeks before the start of my final year, and was regularly worked on with a schedule I planned early on in the development at all times. When deciding which technologies to use, I was fully aware of the fact that I could select ones I was already highly familiar with, and know with full certainty that I could avoid having to put the effort in to learn new things. In the end, I decided to continue with technologies that I was very comfortable with because other technologies that provide the same result would add more complexity due to the learning curves that they have.

At the same time, I wanted my final project to be something I could learn a lot from, and something I could be proud of. I didn't have a lot of experience developing Artificial Intelligence besides the second-year module which is why I was very interested in challenging myself to develop an AI Model from the ground up rather than using an existing industry-level AI Model. Elsewhere, FastAPI, ElectronJS and, to an extent, Python were all technologies I had no previous experience with; they were all selected

not only because they perfectly fit the requirements of the project but also due to their popularity in the industry at the moment.

Skills learned from this project will be invaluable when looking for a job, and I cannot wait to show up and talk to future colleagues about FaceCCTV.

11.2.4. Areas for Future Improvability

There are several features in the final state of the software that could be improved upon or added if given additional time. Namely, the AI Model could of been improved more to increase the efficiency of the Face Detection AI. As of right now, it is functional for specific orientations of pictures and the angle/size of the faces which is not convenient for a user who may have an image that have faces that are not suitable for the AI. Upon detection, the image is enhanced in quality. However the API can not colourise the images which may be helpful in establishing colours of key objects in images e.g. clothes or locations. Implementing this through a library of training another deep learning model would allow for this to happen.

11.3. Conclusions

Overall, the objectives that were set out at the initiation of the project were met, and the time spent on researching and choosing an appropriate tech stack paid off; the in-depth analysis of the current market proved to be worthwhile, as the finished product offered features that current solutions do not have in one place.

Before starting development, the relevant legal, social, ethical, and professional issues were also thoroughly researched to ensure that the application complied with all the relevant laws and regulations while being developed responsibly and ethically wherever and whenever possible.

Highly detailed plans, along with a clear project management approach made development progress smoothly. One of the key contributors in this sense was also the approach taken when it came to solving problems; instead of using trial and error to solve any major issues, such as the ones pointed out at the start of section 9, solutions were planned out and designed before beginning any sort of actual implementation. Thanks to the methodical approach taken, by the end of the project, a fully working hosted, and cross-platform application that individuals can use to manage their finances was developed. Furthermore, the rigorous testing and iterative development process resulted in a more refined final product that had been tested for usability every major step of the way as part of a more user-centric development process.

As previously mentioned, if given additional time, certain features would have been worthwhile to improve upon or add.

References

R. L. A, A. K. S, K. B. E, A. N. D and K. K. V, "A Survey on Object Detection Methods in Deep Learning," 2021 Second International Conference on Electronics and Sustainable Communication Systems (ICESC), Coimbatore, India, 2021, pp. 1619-1626, doi: 10.1109/ICESC51422.2021.9532809. [Accessed: 15 April 2023].

brainhub.eu. (n.d.). *5 Reasons Why You Should Build Electron Desktop App*. [online] Available at: https://brainhub.eu/library/electron-desktop-app/ [Accessed 28 Apr. 2023].

Calipsa (2020). *CCTV statistics in the UK: your questions answered*. [online] www.calipsa.io. Available at: https://www.calipsa.io/blog/cctv-statistics-in-the-uk-your-questions-answered. [Accessed: 17 March 2023].

College of Policing. (n.d.). *Closed-circuit television (CCTV)*. [online] Available at: https://www.college.police.uk/research/crime-reduction-toolkit/cctv#:~:text=The%20meta-analysis%20in%20Review. [Accessed: 24 March 2023].

crimerate.co.uk. (n.d.). *UK Crime and Safety Statistics | CrimeRate*. [online] Available at: https://crimerate.co.uk/#:~:text=The%20crime%20rate%20in%20the. [Accessed: 17 March 2023].

D. Garg, P. Goel, S. Pandya, A. Ganatra and K. Kotecha, "A Deep Learning Approach for Face Detection using YOLO," *2018 IEEE Punecon*, Pune, India, 2018, pp. 1-4, doi: 10.1109/PUNECON.2018.8745376. [Accessed: 15 April 2023].

GDPR (2018). *General Data Protection Regulation (GDPR)*. [online] General Data Protection Regulation (GDPR). Available at: https://gdpr-info.eu/. [Accessed: 14 March 2023].

GOV.UK (1995). *Disability Discrimination Act 1995*. [online] Legislation.gov.uk. Available at: https://www.legislation.gov.uk/ukpga/1995/50/contents. [Accessed: 14 March 2023].

Gov.uk (2018). *Data Protection Act 2018*. [online] Legislation.gov.uk. Available at: https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted. [Accessed: 14 March 2023].

Kornienko, D.V., Mishina, S.V., Shcherbatykh, S.V. and Melnikov, M.O. (2021). Principles of securing RESTful API web services developed with python frameworks. *Journal of Physics: Conference Series*, [online] 2094, p.032016. doi:https://doi.org/10.1088/1742-6596/2094/3/032016. [Accessed: 27 March 2023].

Lazar, J. (2001). *User-centered Web Development*. [online] *Google Books*. Jones & Bartlett Learning. Available at: https://books.google.co.uk/books?hl=en&lr=&id=wrIkWmtt-soC&oi=fnd&pg=PR3&dq=web+development&ots=TOy7Eplnvh&sig=-JBqRKuHM56zujI4PiMpsouYncE#v=onepage&q=web%20development&f=false [Accessed 27 Apr. 2023].

Porter, G. (2009). CCTV images as evidence. *Australian Journal of Forensic Sciences*, 41(1), pp.11–25. doi:https://doi.org/10.1080/00450610802537960. [Accessed: 21 April 2023].

Science ABC. (2020). *Bad Footage Quality: Why Are Security Cameras So Low Quality?* [online] Available at: https://www.scienceabc.com/eyeopeners/why-is-the-quality-of-cctv-footage-still-so-low.html. [Accessed: 14 March 2023].

Theophano Mitsa (2019). *How Do You Know You Have Enough Training Data?* [online] Medium. Available at: https://towardsdatascience.com/how-do-you-know-you-have-enough-training-data-ad9b1fd679ee. [Accessed: 21 March 2023].

Torgeir Dingsøyr, Sridhar Nerur, VenuGopal Balijepally, Nils Brede Moe, A decade of agile methodologies: Towards explaining agile software development, Journal of Systems and Software, Volume 85, Issue 6, 2012, Pages 1213-1221, ISSN 0164-1212. Available at: https://www.sciencedirect.com/science/article/pii/S0164121212000532. [Accessed: 24 March 2023].

Tsiodoulos, D. (n.d.). *Comparison of hamburger and bottom bar menu on mobile devices for three level navigation Comparison of hamburger and bottom bar menu on mobile devices for three level navigation*. [online] Available at: http://www.diva-portal.org/smash/get/diva2:922114/FULLTEXT01.pdf. [Accessed: 21 March 2023].

Y. Wang and X. Duan, "Research on Face Recognition Algorithm Based on Deep Learning," 2021 IEEE 21st International Conference on Communication Technology (ICCT), Tianjin, China, 2021, pp. 1139-1142, doi: 10.1109/ICCT52962.2021.9657956. [Accessed: 21 April 2023].

W. Yang and Z. Jiachun, "Real-time face detection based on YOLO," 2018 1st IEEE International Conference on Knowledge Innovation and Invention (ICKII), Jeju, Korea (South), 2018, pp. 221-224, doi: 10.1109/ICKII.2018.8569109. [Accessed: 5 April 2023].

Shu Zhan, Qin-Qin Tao, Xiao-Hong Li, Face detection using representation learning, Neurocomputing, Volume 187, 2016, Pages 19-26, ISSN 0925-2312. Available at: https://www.sciencedirect.com/science/article/pii/S0925231215018573. [Accessed: 14 March 2023].

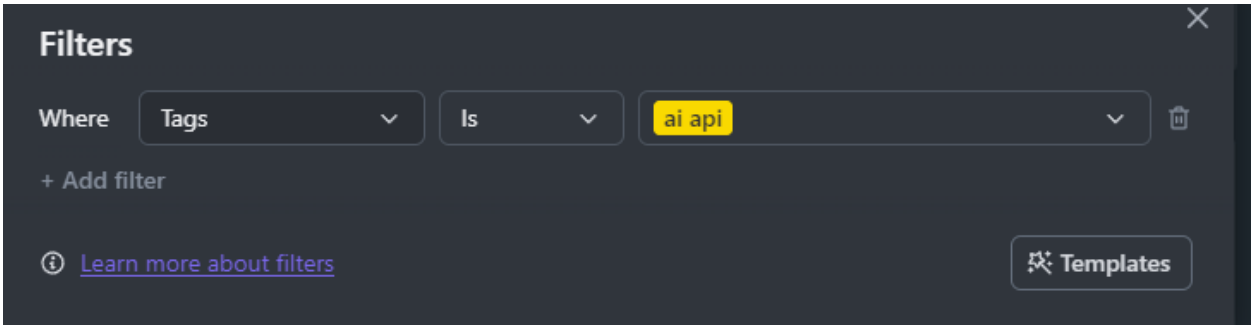
Appendices

Appendix 1: Kanban Boards

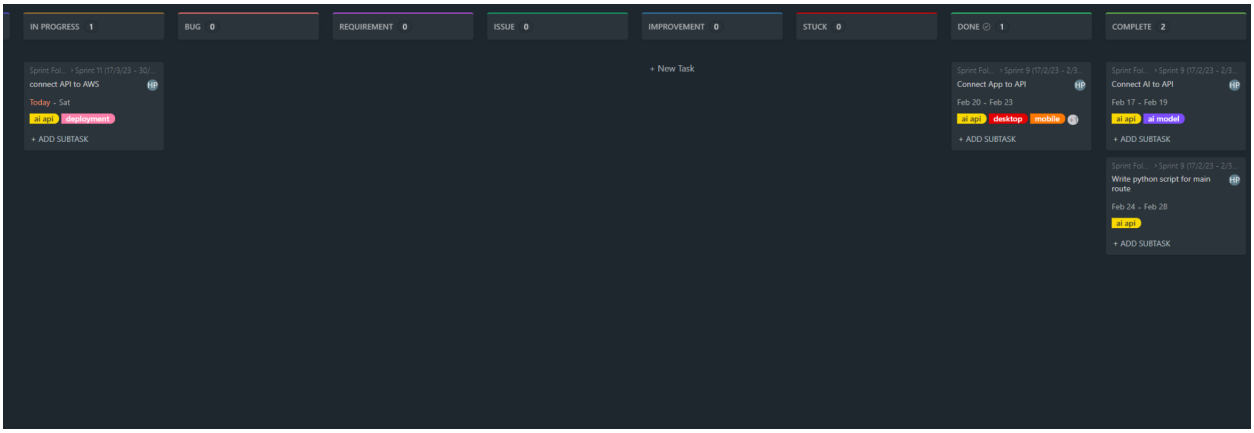
A total of 8 tags on one Kanban Board were created using the board feature on ClickUp. At the start of each sprint, the relevant tasks would be moved to the “In Progress” column being tagged with the respective relative part e.g. API or AI Model. Once a task was completed, it was moved to the “Done” column before it was reviewed and if the developer was happy then the task was moved to the “Complete” column. The screenshots below were taken at the end of the project, so all of the tasks appear in the rightmost column.

Additional categories were made to determine “bugs”, “issues”, “improvements” and “requirements” to keep track of smaller tasks within the general sprint plans. The tasks in the Kanban board were filtered by tags to view those tasks related to a specific part of the project.

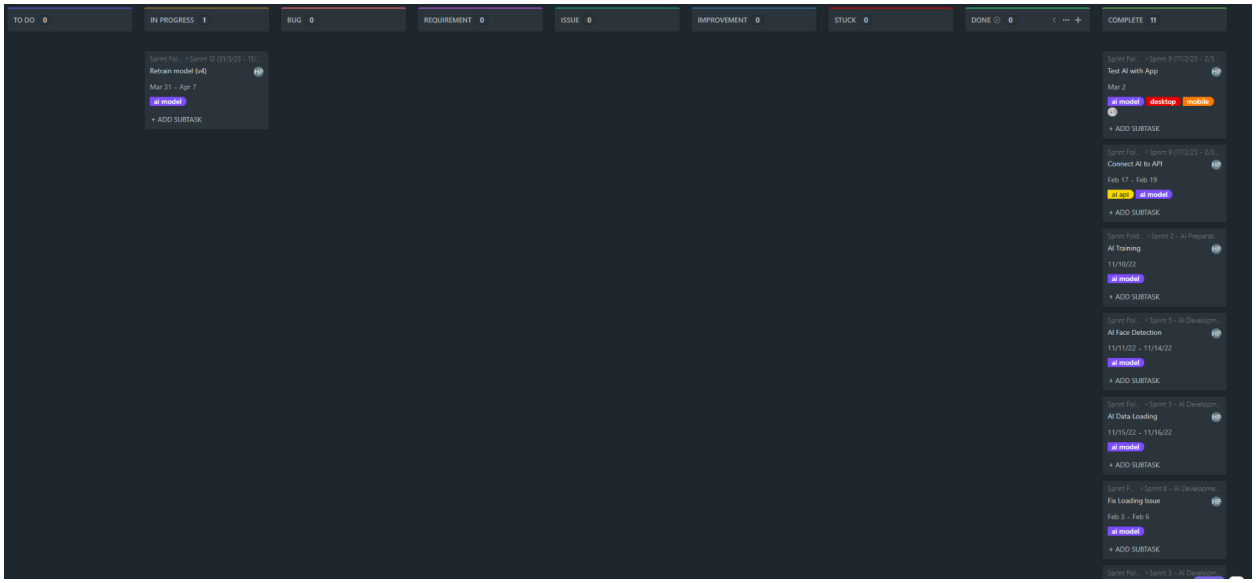
Kanban Board Filter Example



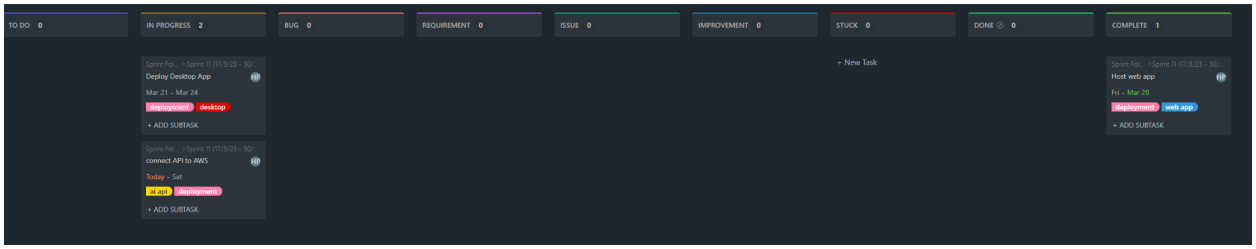
API



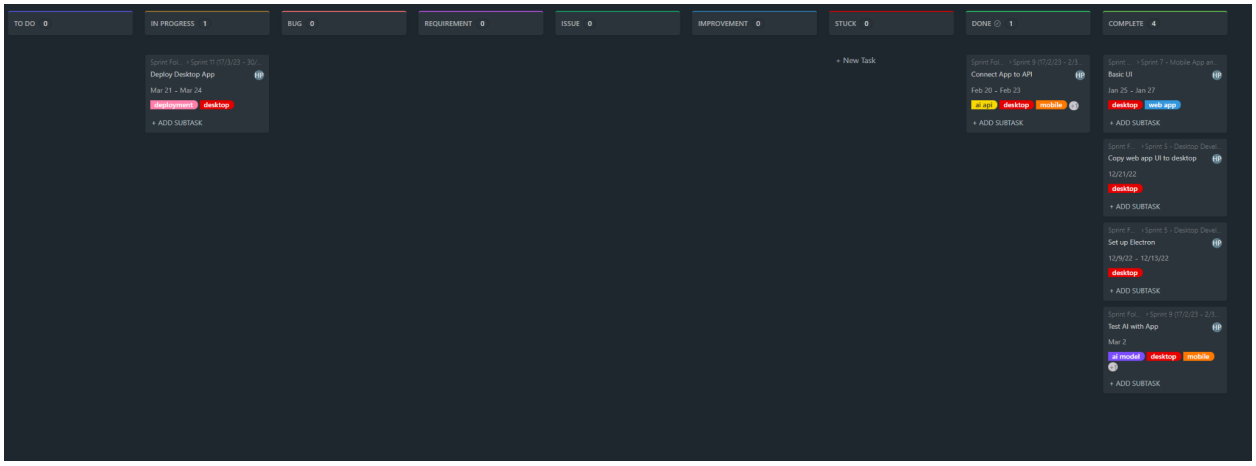
AI Model



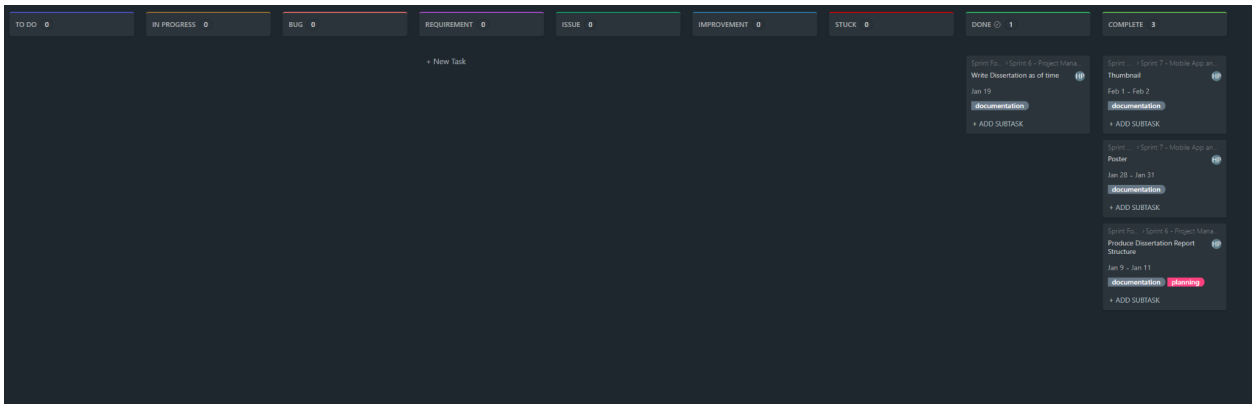
Deployment



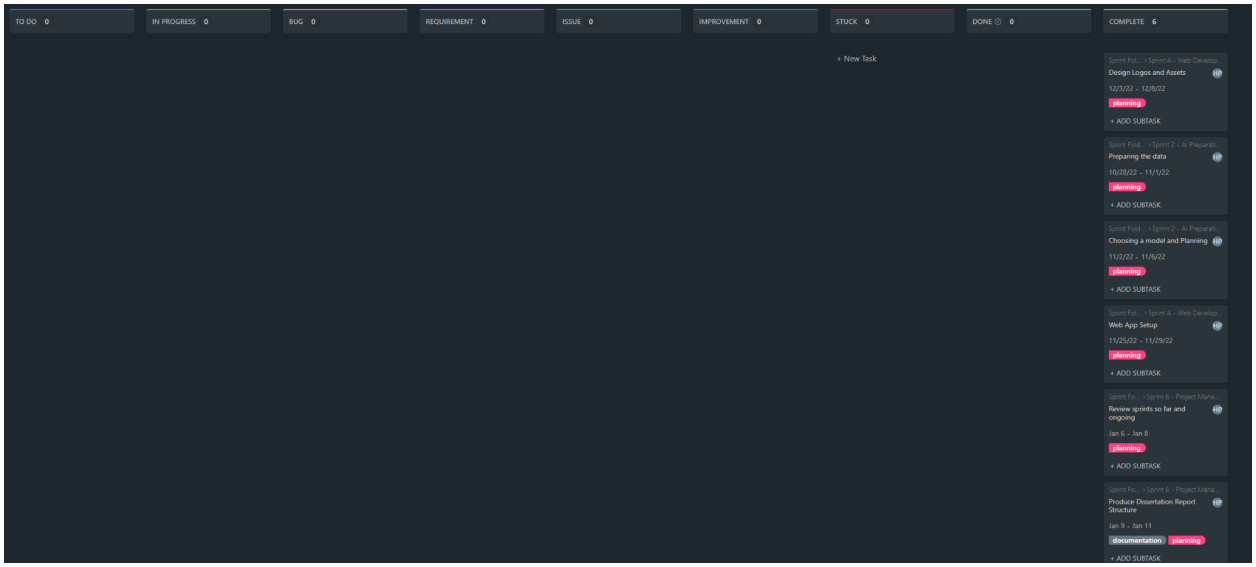
Desktop



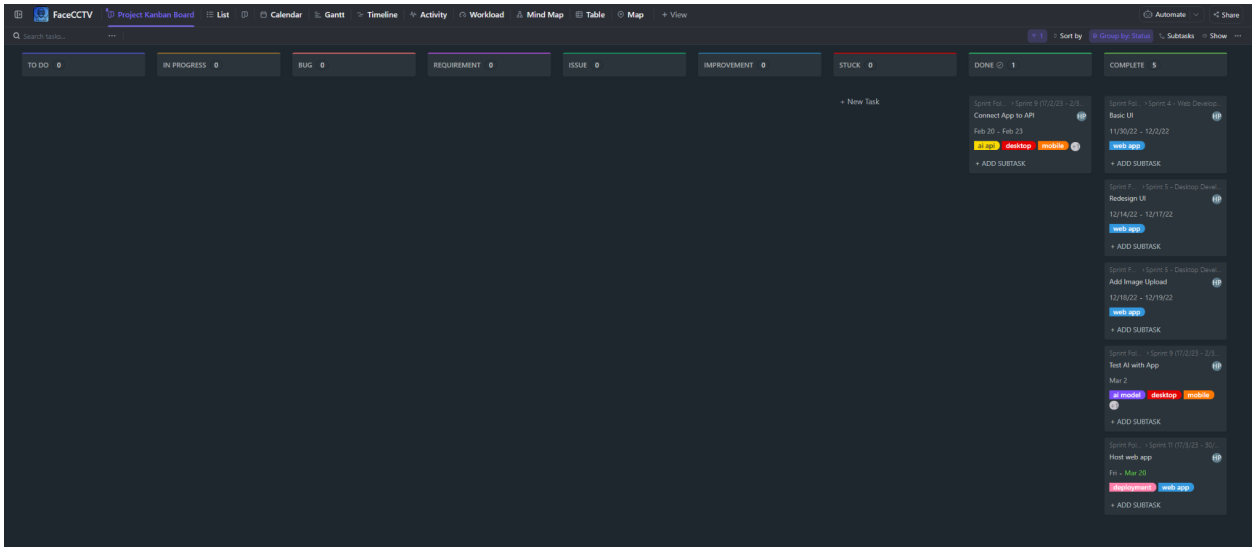
Documentation



Planning



Web App



Appendix 2: Software Engineering

Considering FaceCCTV's codebase has more than 80,000+ lines of code, only the most noteworthy parts will be analyzed below. These include solutions to major challenges faced during development and innovative approaches towards handling performance and other issues.

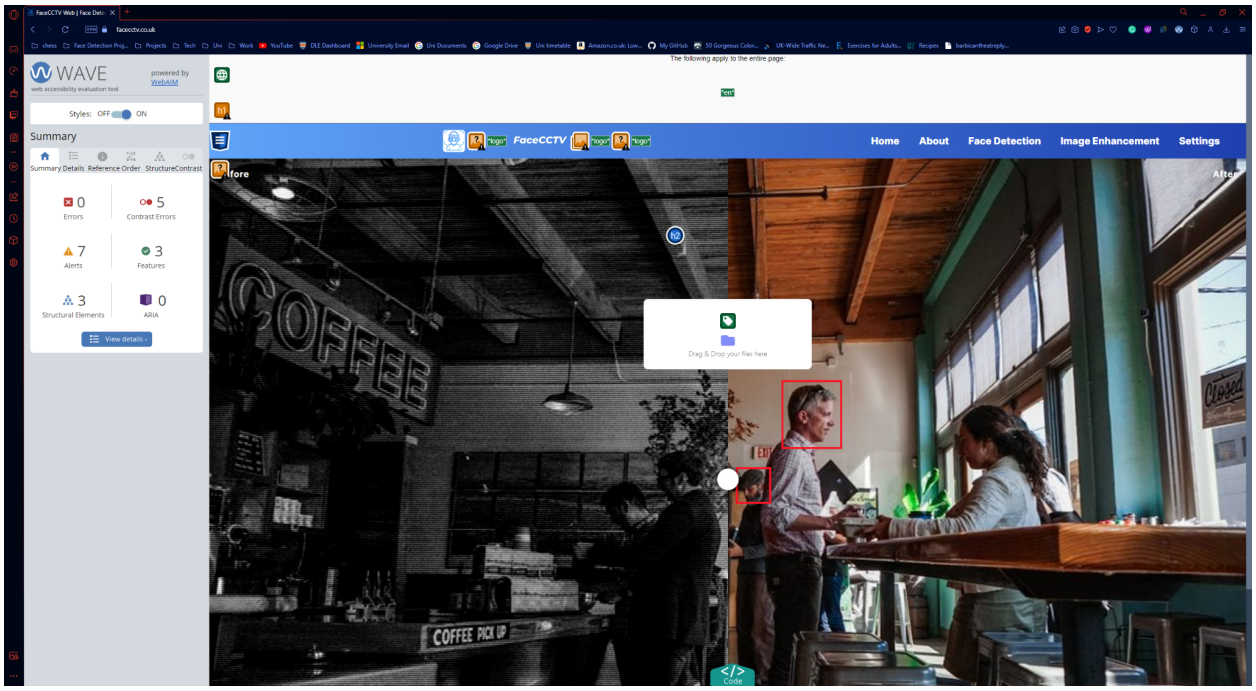
API Return Types

As we are using images in the API. We have to send them from the html form and back to the front end in a particular byte so that it can be viewed correctly by the user and the API as both communicate differently. So as the image is uploaded it has to be converted into a numpy array which then can be read by the AI and then converted back to a image byte stream for the API to return it as a Streaming Response which allows the API pass the image bytes back to the web app and be viewed as a image.

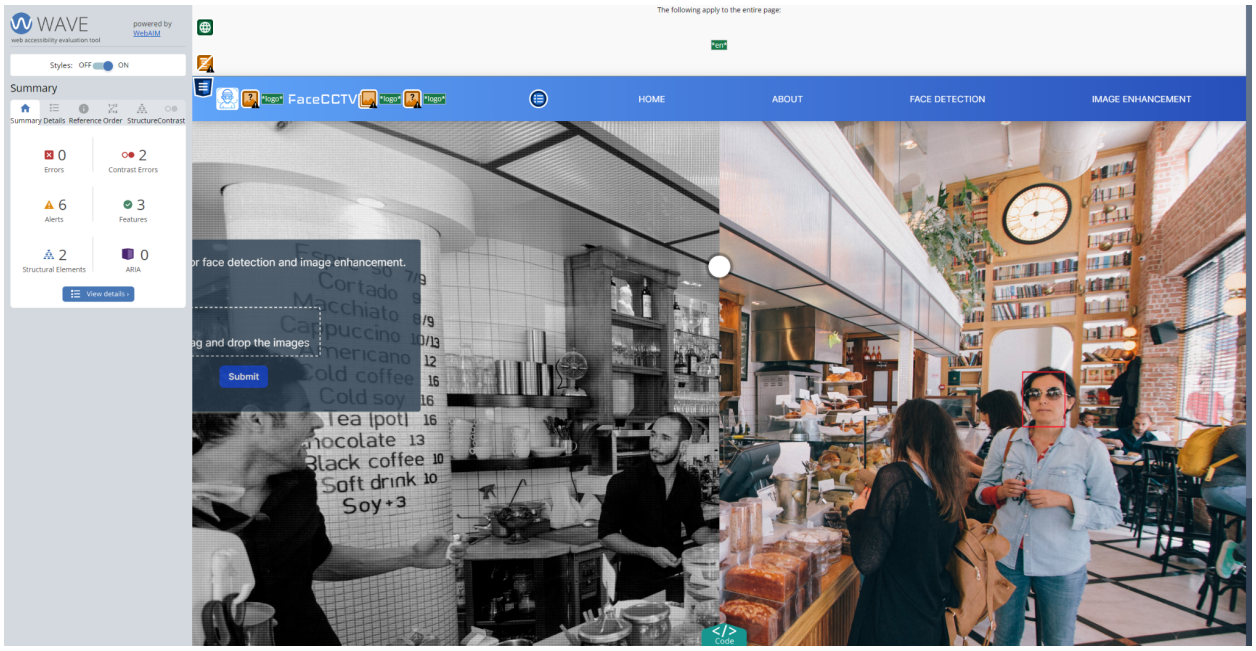
```
fine an API endpoint to handle image uploads
.post("/task/full-image-examination")
c def DetectFacesAndImproveQualityImage(file: UploadFile = File(...)):
contents = await file.read()

# Convert the byte stream into a numpy array
nparr = np.frombuffer(contents, np.uint8)

# Read the numpy array as an image using OpenCV
img = cv2.imdecode(nparr, cv2.IMREAD_COLOR)

# Convert the image to grayscale
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)

# Detect faces
faces = face_model.detectMultiScale(gray, scaleFactor=1.1, minNeighbors=5)

# Draw bounding boxes on the image
for (x, y, w, h) in faces:
    cv2.rectangle(img, (x, y), (x+w, y+h), (0, 255, 0), 2)

# Apply a bilateral filter to reduce noise while preserving edges
bilateral_img = cv2.bilateralFilter(img, 9, 75, 75)

# Convert the enhanced image back to bytes
enhanced_img_bytes = cv2.imencode(".jpg", bilateral_img)[1].tobytes()

# Return the enhanced image as a StreamingResponse
return StreamingResponse(io.BytesIO(enhanced_img_bytes), media_type="image/jpeg", headers={'Content-Disposition': 'attachment; filename=result.jpg
```

Appendix 3: Production Build Test Results

A WAI Evaluation Tool was used to check whether or not the application complied with the Web Accessibility Initiative guidelines.

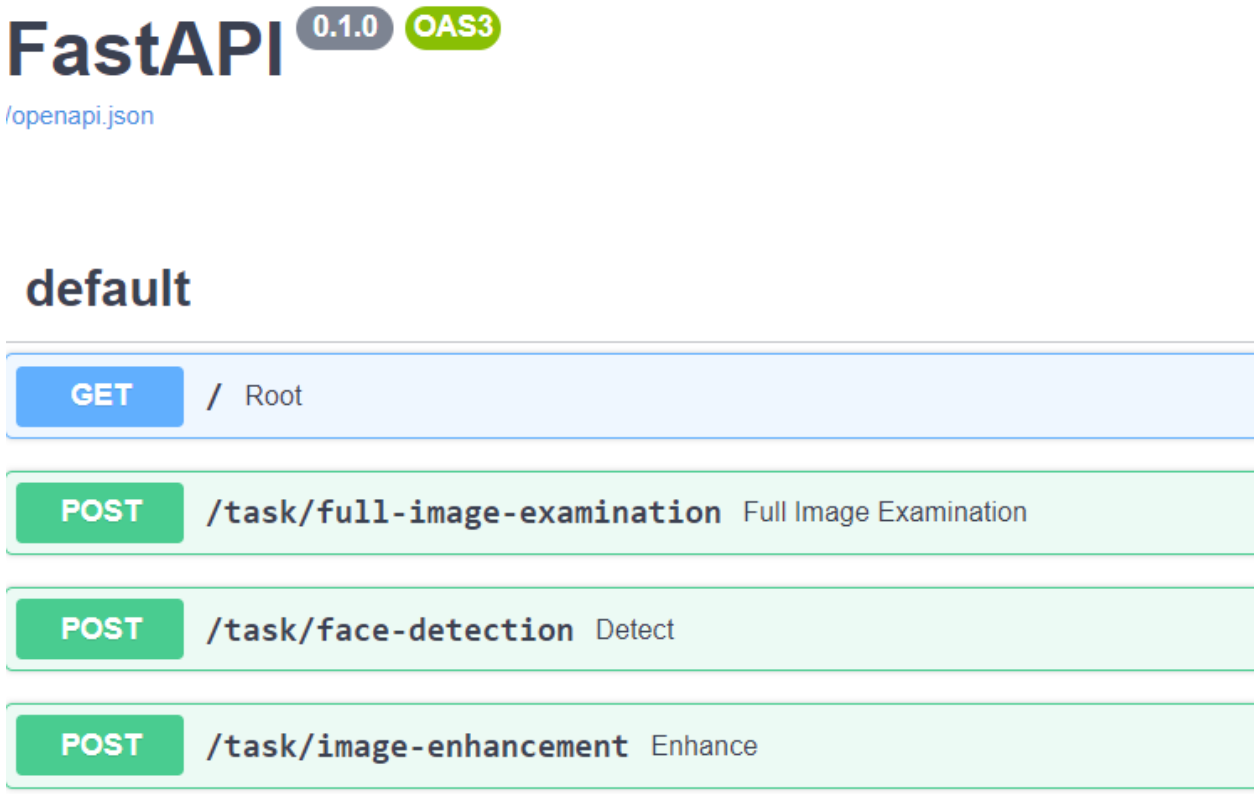


When first tested, the application had 5 errors, all 5 were contrast errors. However, there were alerts which are mostly related to “suspicious alt text” in case the photos won’t load. The other alert was the slider button wasn’t able to be triggered by keyboard inputs, which could impact those who do not use a mouse.



Moving to now where the web app only has 2 contrast errors.

Appendix 4: API Documentation



Method	Route	Input	Response	Description
GET	/	-	{ message: “Welcome message” }	Home Page.
GET	/status	-	{ status:”online” }	Used to check if the API is working.
GET	/api	-	Showcase about page	Webpage about the tool.
GET	/docs	-	200 Ok	Opens the Swagger FastAPI Documentation.

POST	/task/full-image-examination	Image	200 Ok or 500	Returns the image with detected faces and enhanced.
POST	/task/face-detection	Image	200 Ok or 500	Returns the image with detected faces.
POST	/task/image-enhancement	Image	200 Ok or 500	Returns the image with enhanced quality.

Appendix 5: Biweekly Supervisor Meetings

The supervisor of the project held biweekly stand-up meetings to check its progress. To prepare for the meetings, PowerPoint presentations were created. They can be seen by following the link below:

https://github.com/Parker06/FaceCCTV/tree/main/documentation/MeetingMinutes

Appendix 6: User Guide

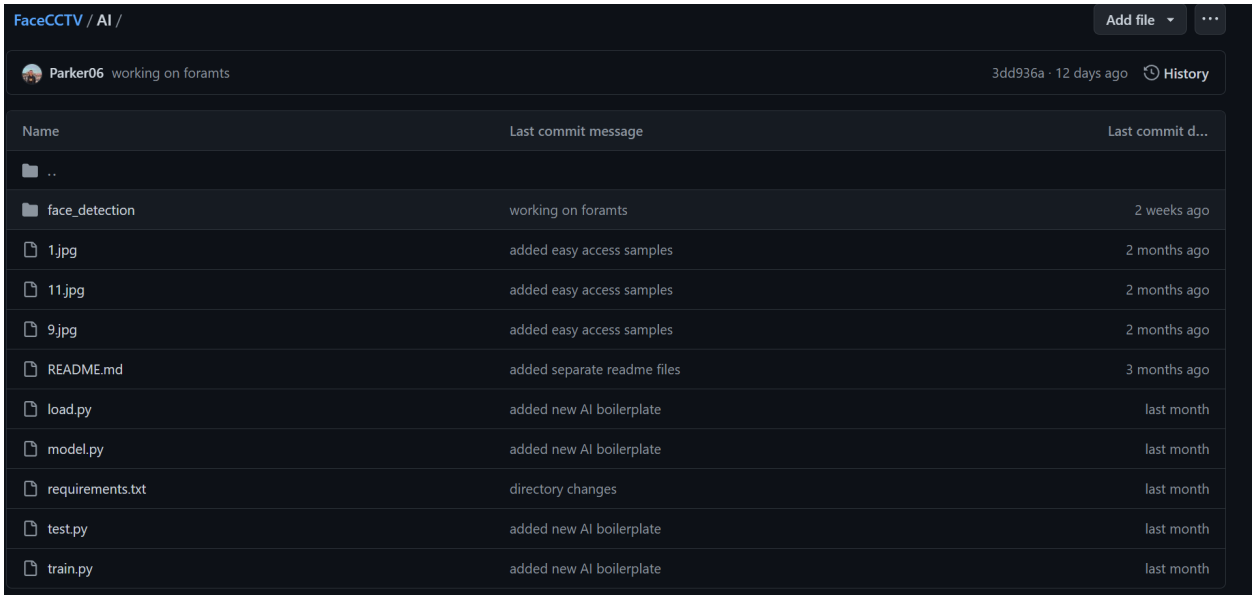
Considering the application is meant for everyday users of all technical backgrounds, a user guide has been created to make it clearer for readers who are not experienced with technology themselves what the application can do, and how a user would utilise it. The rest of this document is purely screenshots and explanations on how to interact with the application, including the setup process. Please note that setup instructions have been provided on the repository as well for regular users who would not read this report:

https://github.com/Parker06/FaceCCTV/blob/main/WALKTHROUGH.md

AI Model

Users can download the jupyter notebook that holds the python scripts that build the AI model alongside other necessary files under the AI section of the GitHub repository:

https://github.com/Parker06/FaceCCTV/tree/main/AI



Once downloaded, the contents of the folder can be extracted. Users must use a CLI such as Git Bash, Terminal or Powershell, navigate into the AI folder and run the `pip **install** -r requirements.txt` command to install the python modules that FaceCCTV's AI Model uses.

Web App

The web app is hosted on a domain so it requires no user interaction.

The domain is https://facecctv.co.uk/

API

The API folder has the most up to date version of the h5 model file and hosted on a server which the website is connected to. However you can run it locally for experimental use or if the server is offline. First you need to download the `API.zip` file from the latest release: https://github.com/Parker06/FaceCCTV/releases/tag/V.1.0.0.

Once downloaded, extract the content, and place the FaceCCTV folder wherever you want. Using a CLI such as Git Bash, Terminal or Powershell, cd into the FaceCCTV/API/ directory and run the `pip **install** -r requirements.txt` command to install the python modules that FaceCCTV's API uses. Once done, run the `uvicorn main:app –reload` command to start the fastapi API local host server. The API will be hosted on http://127.0.0.1:8000 which you can modify in the HTML files. You can type http://127.0.0.1:8000/docs to view the FastAPI Swagger documentation.

By default, port `8000` is used.

Web App local use

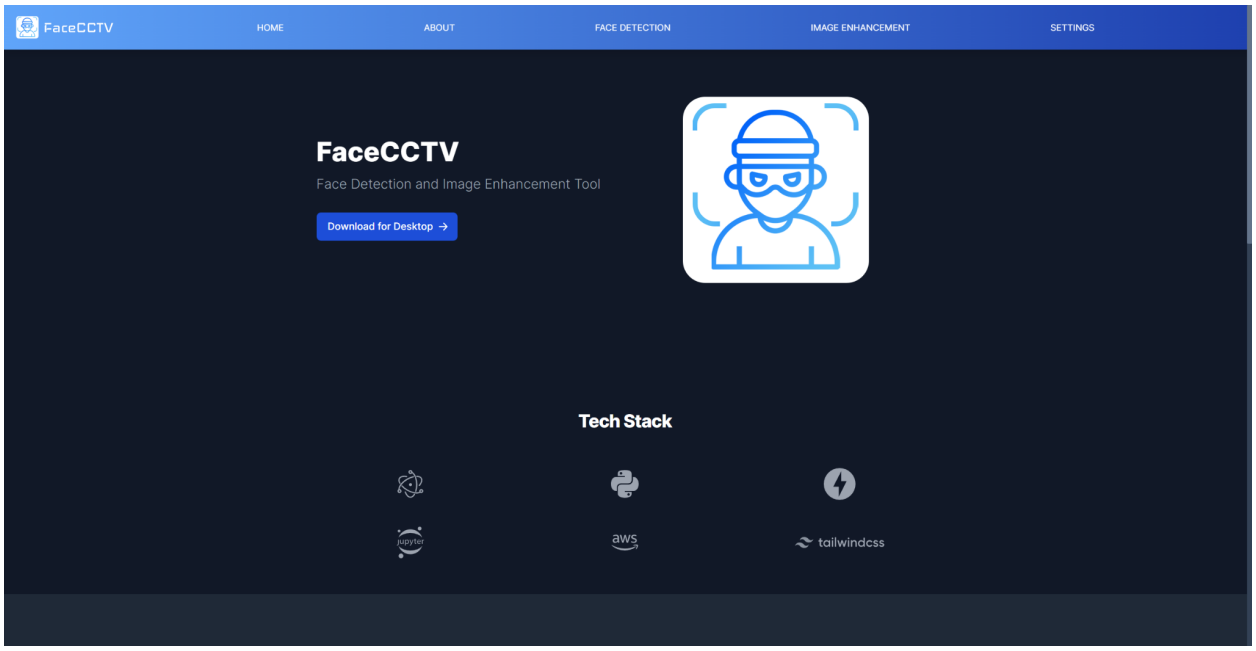
After following the instructions above, you should be able to use the web app immediately by simply going the web address and use the app. (or http://127.0.0.1:3000 / http://localhost:3000 if using the host device itself to access it).

Desktop App

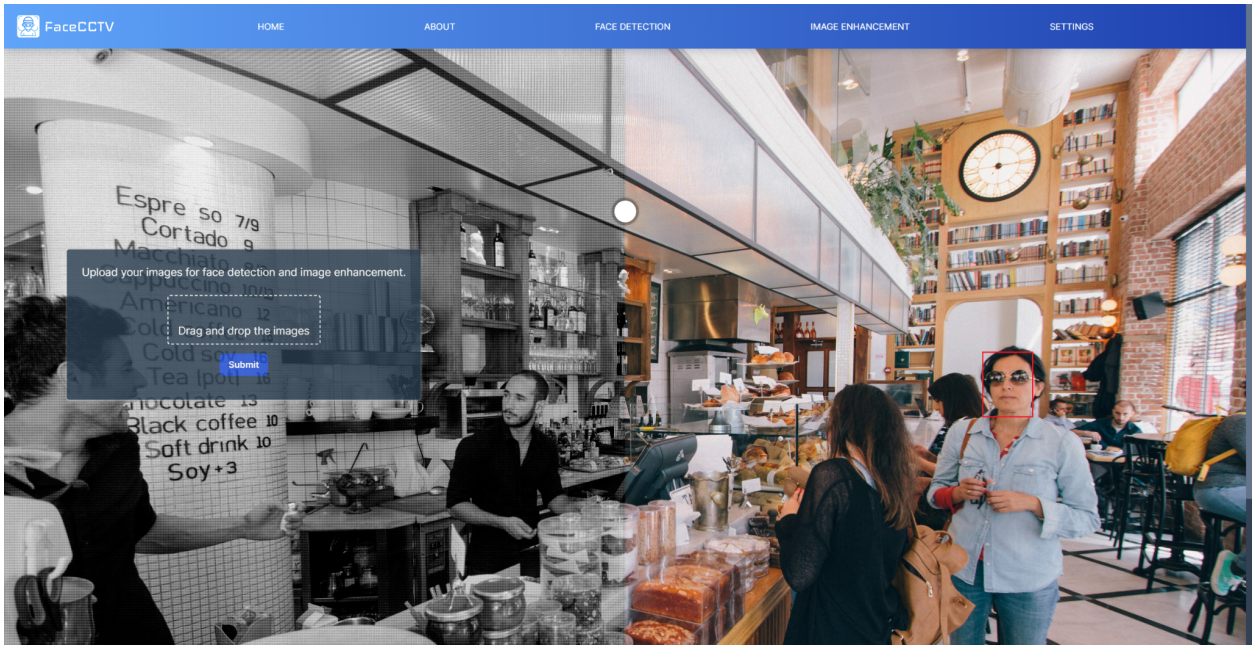
Download the latest version of the desktop app from the Releases section https://github.com/Parker06/FaceCCTV/releases/tag/V.1.0.0, and install it. Once installed, open the app, and then the API should be connected automatically based on the third party server hosting the API or you are running the API yourself.

Walkthrough on the use of the app

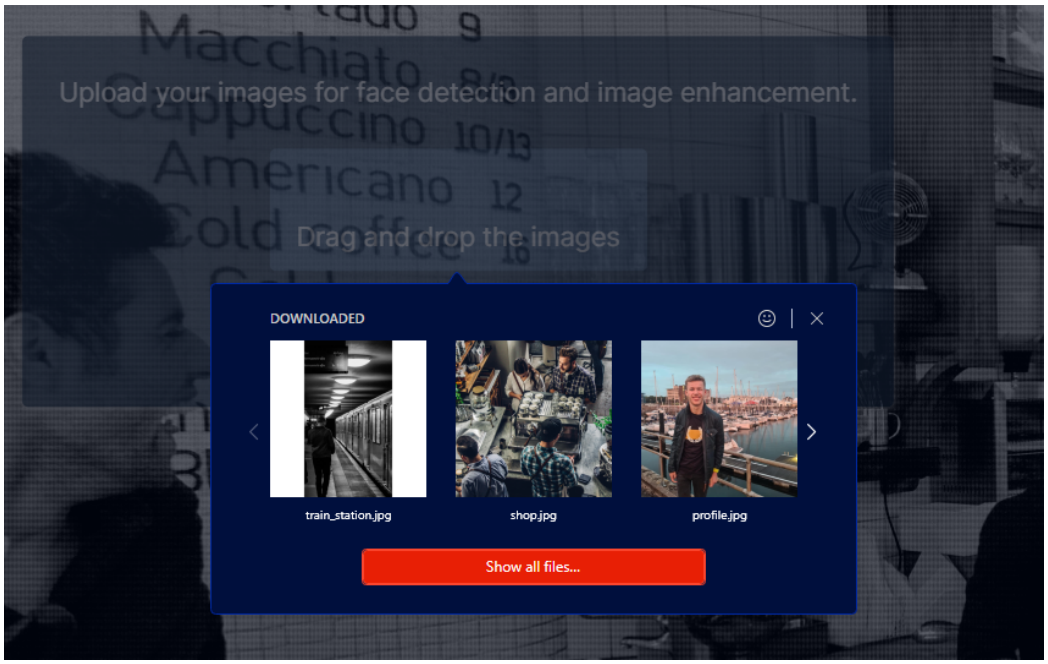
User can access the web app by typing in the search engine's search bar "https://facecctv.co.uk". From there they can navigate to the About Page to download the desktop app. Or navigate to the releases section of the GitHub repository.



After entering the URL, users will see the Home Page where they can navigate to other pages or get straight on to what the app is about. No login is required.

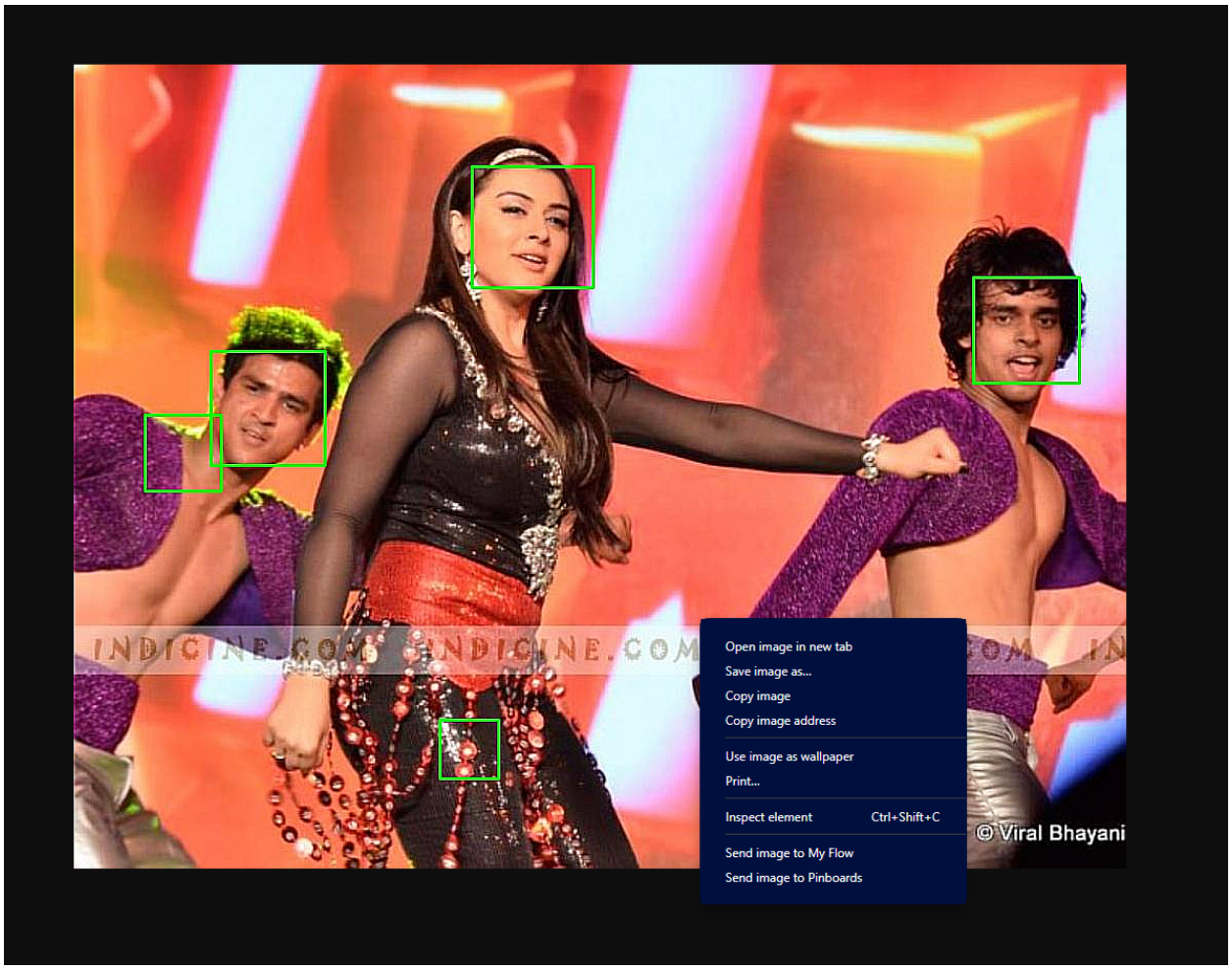


They can drag and drop your image in the upload button or simply click to upload an image. The app only accepts .jpeg, .jpg and .png.



Once they have uploaded an image they are greeted with a loading animation

It may take a few seconds to run the AI through the API. Afterwards, it will redirect you to the results page where you can download the result as a .jpeg or .png.



Notes:

- The Face Detection and Image Enhancement Pages follow the same procedure.
- The user settings only change on your device so if it's a new device then you will have to set it up again.