A picture containing text

Description automatically generated

The research, development, and analysis of an app to analyse images of coins and return the total amount of money within the image

Dissertation

DT211C

BSc in Computer Science (Infrastructure)

**Jason Moran**

**C18394953**

**Brendan Tierney**

School of Computer Science

Technological University, Dublin

**06/12/2021**

Abstract

I’m currently working in a hotel. I find that one of the things that slows me down and causes problems is receiving a handful of coins from a customer, totalling to €17.98, and the customer looking to pay for something that costs €15.37. These kinds of situations cause 2 slowdowns as firstly you have to count up how much money the customer gave you and then have to count up the change.

During my research, I am yet to find an application, mobile or PC based that will simplify/ speed up this task. I have also heard from a number of co-workers that this can be not only annoying as it causes a pointless increase in work but can also cause stress as this extra time will mean other customers will be waiting.

As well, there is no current way to fix this problem except by investing in a €1000 machine that you drop the coins into, which are often very bulky and very expensive, meaning no business will invest in these for staff for an issue that can be solved with a little bit of counting. As well often at the end of the day, managers will have to spend about 45 minutes to an hour, sometimes even longer, counting up all of the notes and coins in all of the tils to see how much money was earned that day, a problem that could be solved by about 5 pictures being analysed almost instantly

Declaration

I hereby declare that the work described in this dissertation is, except where otherwise stated, entirely my own work and has not been submitted as an exercise for a degree at this or any other university.

Signed:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Jason Moran

Date: 09/12/2021

Acknowledgements

I would like to thank a number of people for all of the support and inspiration on this project. To start with, I would like to thank my supervisor, Brendan Tierney. They helped me to a massive extent and were both very accommodating and helpful whenever I had any issues or problems I encountered. They were a brilliant guide in the correct things to do and things to avoid, and he motivated me to keep going with the project, even at times I thought I might not get it done. I would also like to thank Damian Gordan, the FYP co-ordinator. He gave me loads on help during the early stages of this project and truly inspired me to take on this idea when I was unsure about it. The last lecturer I would like to thank is Jonathan McCarthy. While he had no direct impact on my project, I was able to spend many hours getting advice from him coming towards the end of the project, and he truly did help reduce a lot of stress with the advice that was given. I would also like to thank my family, and particularly my younger brother, Shane Moran. If it wasn’t for long late-night conversations with him to clear my head, I wouldn’t have brainstormed ideas that helped solve the many problems I encountered with the project. I would also like to give a special thanks to a few of my friends from college. These include Gabriel Hynes, Dylan O’Connor, Aaron Murphy, Luke Dowdall, Tony Morris, Robbie Hatfield and Niall McNamara. Without the assistance from any of these, the struggle on this project would’ve been twice as hard and solutions to problems I found would’ve taken twice as long to come up with. I would also like to thank all of the staff of the Marine Hotel. These include Aoife Colgan, Arkadiusz Siwonia, Shane Moran (again), Ciara Lawless, Patricia O’Neill, Michael Siwonia, Ciara Keogh, Connie Keogh, Krunoslav Cicak, Ruzel De Gracia, Eve Meehan, Jordan McClean, Rachel Gosker and Thomas Collins. These are all the individuals who helped me with the testing and evaluation of the app and gave input on what features they would like to see included. Without these, the app would be a far ways away from what it is today.

Table of Contents

[1. Introduction 7](#_Toc100768269)

[1.1. Project Background 7](#_Toc100768270)

[1.2. Project Description 7](#_Toc100768271)

[1.3. Project Aims and Objectives 8](#_Toc100768272)

[1.4. Project Scope 9](#_Toc100768273)

[1.5. Thesis Roadmap 9](#_Toc100768274)

[*Design* 10](#_Toc100768275)

[*Development* 10](#_Toc100768276)

[*Testing and Evaluation* 10](#_Toc100768277)

[*Redevelopment* 10](#_Toc100768278)

[*Conclusions and Future Work* 10](#_Toc100768279)

[2. Literature Review 12](#_Toc100768280)

[2.1. Introduction 12](#_Toc100768281)

[2.2. Alternative Existing Solutions 12](#_Toc100768282)

[2.3. Technologies Researched 13](#_Toc100768283)

[2.4. Other Relevant Research 15](#_Toc100768284)

[2.5. Existing Final Year Projects 16](#_Toc100768285)

[2.6. Conclusions 17](#_Toc100768286)

[3. Prototype Design 19](#_Toc100768287)

[3.1 Introduction 19](#_Toc100768288)

[3.2. Software Methodology 19](#_Toc100768289)

[3.3. Overview of System 21](#_Toc100768290)

[3.4. Front-End 22](#_Toc100768291)

[Final Design 23](#_Toc100768292)

[Use Case Diagrams 24](#_Toc100768293)

[3.5. Middle-Tier 25](#_Toc100768294)

[3.6. Conclusions 26](#_Toc100768295)

[4. System Development 27](#_Toc100768296)

[4.1. Introduction 27](#_Toc100768297)

[4.2. System Development 27](#_Toc100768298)

[Initial Prototype 27](#_Toc100768299)

[Front End Development 28](#_Toc100768300)

[Back End Development 34](#_Toc100768301)

[Analysis Development 35](#_Toc100768302)

[Analyzer Functionality 35](#_Toc100768303)

[4.3. Issues and Changes 40](#_Toc100768304)

[4.4. Conclusions 41](#_Toc100768305)

[5. Testing and Evaluation 42](#_Toc100768306)

[5.1. Introduction 42](#_Toc100768307)

[5.2. Plan for Testing 42](#_Toc100768308)

[Test Plan 43](#_Toc100768309)

[5.3. Plan for Evaluation 44](#_Toc100768310)

[5.4. Conclusions 45](#_Toc100768311)

[6. Conclusions and Future Work 46](#_Toc100768312)

[6.1. Introduction 46](#_Toc100768313)

[6.2. Future Work 46](#_Toc100768314)

[6.3. Conclusions 46](#_Toc100768315)

[Bibliography 47](#_Toc100768316)

# 1. Introduction

## Project Background

I’m currently working in a hotel. I find that one of the things that slows me down and causes problems is receiving a handful of coins from a customer, totalling to €17.98, and the customer looking to pay for something that costs €15.37. These kinds of situations cause 2 slowdowns as firstly you have to count up how much money the customer gave you and then have to count up the change.

During my research, I am yet to find an application, mobile or PC based that will simplify/ speed up this task. I have also heard from a number of co-workers that this can be not only annoying as it causes a pointless increase in work but can also cause stress as this extra time will mean other customers will be waiting.

As well, there is no current way to fix this problem except by investing in a €1000 machine that you drop the coins into, which are often very bulky and very expensive, meaning no business will invest in these for staff for an issue that can be solved with a little bit of counting. As well often at the end of the day, managers will have to spend about 45 minutes to an hour, sometimes even longer, counting up all of the notes and coins in all of the tils to see how much money was earned that day, a problem that could be solved by about 5 pictures being analysed almost instantly.

As can be seen in the research papers 1. (Bhurke and Sirdeshmukh, 2021) and 2. (Sawant and More, 2016), there applications and scripts that exist to detect what currency is within an image, but there is little to anything about analysing images to calculate the total value of money within the image.

## Project Description

This application, which has yet to be named, is going to be a mobile app mainly for making life slightly easier. It will allow the counting and sorting of coins to be streamlined and made both way faster and easier. The idea is that the user takes a picture of some coins, it then runs the image through a coin detection and image analyses algorithm to find the location and value of each coin within the image.

The main purpose of this app is to speed up monotonous tasks involving coins, whether that’s managers counting the total earned in one day or its regular workers counting their pocket of tips during their lunch break, so the focus needs to be on making the app and algorithm work as fast as possible. Including little checks such as a brightness scanner to tweak certain parameters within the coin detection algorithm, or working with ratios to make it so the distance the image is taken at not being an issue are all thing that not only need to be included but also a major part of the application, as an application that takes just as long to work out the total as counting it yourself is pretty pointless.

It is also important for this project to contact the people who work in these positions (such as myself and my co-workers) and ask them what specific features they would like to see. This makes sure that the app is catered towards and designed for the people it was meant to be used by and who inspired its creation. These are the people who work in these situations so they are the ones who would know the most what would be beneficial and helpful and what would be a waste of development time

The approach to building this app will focus the majority of the time on developing the algorithm as getting it as fast and efficient as possible has to be the main focus. Additional features and then the app itself should not take too long to design and develop as the simpler and easier to navigate the ui, the better it will be for those fast paced stressful moments where you need the coins counted quickly.

Figure 1 below shows a sample output for what the application will show, with the addition of a pop with a couple optionsA group of coins

Description automatically generated with medium confidence.

## Project Aims and Objectives

The overall aim of the project is a simple one: to provide an application that can make the counting and evaluating of large amounts of coins as quick and simple as possible.

The goal for this project is to provide a modern, easy to use and accessible android phone application that will help users to make certain laborious tasks as quick and easy as possible. The development of this app will be focused around to main things, speed and ease of use. The faster and easier to use, the better.

To achieve these aims some milestones were set. These milestones involved setting flexible dates to complete certain parts of the project. By setting specific time frames to have certain parts of the application developed it was insured that the final application would be completed to the best attempt by the end.

The purpose of this project is to make it more affordable (free) to complete simple repetitive tasks. There are already machines that exist that do this, but these machines unfortunately are owned mainly by banks, and cost minimum €5000 which a lot of small businesses or general workers just cannot spare. It frees up people’s time, ultimately saving both money and stress for businesses and individuals

The app isn’t about making a tool for big banks to use, nor is it a banking app itself. It is designed to help reduce the time wasted on tasks and the stress caused by tasks that in this day and age should be almost automated anyway. It is supposed to make the lives of an individual or small businesses easier

## Project Scope

The app is not about designing the fastest coin recognition app nor is it about finding every single detail within the app. The focus is on making a fast, simple, easy to use application that almost anyone can gain some benefit from. It doesn’t have to be some ground-breaking algorithm for image analyses or circle detection that returns a result almost instantly, it just needs to be fast enough to make the users life easier. The app doesn’t need to 1001 different features including a link to your actual bank account, the simpler the app, the more user friendly, the better.

## Thesis Roadmap

This section provides a brief summary of each of the chapters covered in this report.

Research

This chapter explores background research related to image processing, and more specifically image processing for the purpose of coin detection. It will explore everything from the technologies most commonly used to more alternative technologies that may be useful. It will also take a look at similar applications to this one, and other final year project and areas of research that cover the same or similar topics.

### Design

This chapter delves into the methodology chosen for this project and how these choices came to be. Following this, detailed use-cases and personas related to the desired system will be presented. Finally, the designed technical architecture and software testing plans will be discussed.

### Development

The development chapter breaks down the entire development process of the system regarding the technical architecture outlined in the design chapter. Some of the challenges that may be encountered in the development process will be explored also.

### Testing and Evaluation

This chapter describes how all the testing and evaluation of the system will be executed. Each phase of testing will be described in detail, followed by an in-depth account of all user feedback received during user evaluation trials. Finally, the system will be evaluated to see if it adhered to Nielsen’s Heuristics for User Interface Design.

### Redevelopment

This chapter outlines some of the development steps that can be taken as a result of the feedback gained from the user evaluation. The changes possible and the importance of these changes will be examined.

### Conclusions and Future Work

This chapter will reflect on the entirety of the project and will discuss the conclusions drawn so far, personal reflections made, and the future work planned for the project.

# 2. Literature Review

## 2.1. Introduction

In this chapter some of the key areas of research that are important in this project will be presented. These topics include exploring some of the current applications out there used for coin detection and analysis, different methods of image processing that can be used for coin detection, and a number of different technologies that could be used within the project

## 2.2. Alternative Existing Solutions

There are existing apps on the market to help a user to analyse coins, but so far, I am yet to find one with the same function as my own. As rare coin collection is an entire hobby and career for some people, most applications and systems I have been able to find are focussed on the analyses of a singular coin and finding it in a database rather than the function of my app

The following are mobile apps that analyse images for coin detection:

Coinoscope: Identify coin by image

Coinoscope is an app/service allowing the user to snap a picture of a coin with their phone camera and the app will show them a list of similar coins. The user can then click on the coin to see additional information in the web browser of their phone. The user can also save search results and build a photo collection of coins.

Graphical user interface, website

Description automatically generated A screenshot of a cell phone

Description automatically generated with medium confidence A picture containing timeline

Description automatically generated

Figure 1 – Coinoscope App screenshots

Coinoscope is a very intuitive and easy to use app. It offers the user to find the value of almost any coin if sold online on sites such as ebay. While it’s of my personal opinion that the UI is a little blocky and could use a small bit of work, it definitely is only minor and does not affect the usability or the cleanliness of the app. If coin collection was something I enjoyed I would definitely use this app.

CoinDetect: Euro coin detector

CoinDetect is an app that is very similar to coinoscope. The user just needs to snap a picture of a euro coin with their phone camera and the app will tell you the country of origin, the year of issue and other information about the coin.

A screenshot of a cell phone

Description automatically generated with low confidence Graphical user interface, website

Description automatically generated A screenshot of a cell phone

Description automatically generated with medium confidence

Figure 2 – CoinDetect App Screenshots

The app has a very similar design to coinoscope, with similar layout and colour scheme. While this is true, the coin detection seems to work a little better and is a little more accurate. It also doesn’t allow for selling and buying coins online, rather it allows you to store a collection of all of the euro coins you have found. Still, it only allows the user to scan one coin at a time.

## 2.3. Technologies Researched

Mobile Technology

**Android** is a Linux-based mobile OS developed by Google. Android OS is the current leading mobile OS used with 88 percent of all smartphones sold in Q2 of 2018 having the Android Operating System. Android’s opensource code allows developers and device manufactures to customize and distribute the platform across the market. Java language is the go-to choose for native android app development, although android devices can run a range of programming languages, python being one in particular. Android developers have the opportunity to distribute their programs to any device running android, and this ranges from phones and tablets to watches and cars.

**iOS** is a mobile Operating System developed by Apple. iOS is limited to being used on only Apple-made devices, unlike Android which can be used on a variety of devices. iOS is the second most popular mobile OS after android. As apple have control over both iPhone hardware and software, they have the ability to create an effective and clever user experience. This also allows apple to offer regular software updates to all its devices without having to require testing and approval from different manufacturers. Apple does a much better job than android at keeping most of their users’ devices on the most up to date iOS version. iOS applications are developed using XCode on the Swift platform.

In **conclusion**, the Android operating system is the best choice for the development of my app as there is past experience in creating apps for this platform. Java will be the language of choice as it is used in the android studio development environment. Android also provides a large range of libraries and APIs that will be used in the development of this application.

Databases

For this application local storage is used but there is no need for a full database.

**Local storage:**

The default storage for native android applications is SQLite. Users are able to access old scans and re-scan them or save them, so it is important these images are saved**.**

APIs

There is a variety of Application Programming Interfaces (APIs) to make use of when developing a native android application. Currently there are no APIs planned to be used in the development of this app, although some API’s will be researched and potentially implemented throughout the development of the project.

Version Control

I used Git to help with version control and backup of the project. This was important as it meant if there were any issues or problems during development, which there were many, it was possible to backtrack, find the issue and fix it.

## 2.4. Other Relevant Research

Nielsen’s Heuristics

Nielsen’s Heuristics are 10 general principles for interaction design(24). These heuristics are extremely important for any app, especially a heavily user-oriented app like this. These heuristics were also kept in mind while comparing existing coin detection apps currently available.

These heuristics will be compared to for the duration of the development of the app.

Accessibility

Accessibility is of utmost importance in the development an app where the primary focus is speed. It is important to know what things to do, but also what things to avoid when designing apps such as this. Some factors that must be considered in the development of any app include:

Ease of navigation – how easily a new user can find their way around the app.

Simplicity – The ability for a new user to figure out the functions of the system.

Smoothness – How fast the app moves to new pages and loads certain items.

These three things should be the primary focus of this app as it should make the user’s life easier and shouldn’t require a whole manual just to scan a single picture. A new user should be able to open the application and navigate it almost as quickly and efficiently as an experienced user. The application also needs to be accurate, as there’s no point being the fastest if you’re wrong

Smart Design

Giving users a clean interface will make them more likely to use it, so the less junk on the screen the better.

Data Privacy

As the app will not contain much significant personal data, data privacy is not of the utmost importance. Still, it should at least be considered and kept in mind to make sure nothing important is left vulnerable and easy to steal.

## 2.5. Existing Final Year Projects

A couple of Final Year Projects from previous years were looked at in the research phase of the project. There was an attempt to focus on some projects with some relevance to the themes that will be tackled in the creation of the coin recognition system.

Bird Species Image Classification Project - Ciaran McHale

The goal of this project is to give bird watchers and bird enthusiasts the ability to identify a particular bird they have seen in the wild or a photo they have of a bird. The area of computer science this is in is image processing, image classification, machine learning and computer vision. This project is a phone application that facilitates birdwatchers. The user is able to take out their phone and take a picture of the bird they have seen or a previously captured photo, and it returns what type of species it is. It usually returns the correct bird species after the image is captured.

This app allows the user to capture or upload an image of a bird and it will return what species it belongs to. The point of this app is to allow users who enjoy birdwatching the ability to confirm what kind of bird they have caught or seen.

TetScan: Using Pixel Scanning and AI to play Tetris – Igor Bolek

Training The goal of this project was to develop an application capable of playing a Tetris game on its own by using Image Processing techniques to scan the game pixels on screen and reconstructing the game state from observation. This game state is then used as input for an AI algorithm to predict the best move and emulate key presses required to perform that move. Part of this project was also the implementation of machine learning component that uses deep reinforced learning algorithm to train a DQN agent to play Tetris. This demonstrated the ability of AI to control the game without injecting any code or exchanging any data other than pixel observation.

## 2.6. Conclusions

With the necessary gained knowledge of image processing gained throughout the research stage of the project, the development stage can begin. I have learned a lot about the way in which item detection can occur and understand the scope that the app must cover

From looking at similar projects that used well known methods for item detection, to looking at new and quite unknown technologies and methods to generate a desired result. I have looked through documentation about a lot of the different technologies and methods available and I feel I now have a good grasp as to what is required

The technologies best suited for the project were decided after researching many different options.

Requirements Table

|  |  |  |
| --- | --- | --- |
| **Name** | **Description** | **Priority** |
| Take picture | Enables a user to take an image from within the app | HIGH |
| Analyse Image | Allow the system to analyse the image and run it through the coin detection algorithm | HIGH |
| Return result | Return a result for the user to see | HIGH |
| Store image | Save a processed image with the result attached to it | MEDIUM |
| Delete Images | Delete old images if result was incorrect/taking too much storage | MEDIUM |
| Re-test old images | Run an image through the algorithm again if the result was incorrect | MEDIUM |

# 3. Prototype Design

## 3.1 Introduction

Following on from the previous chapter, where some of the key background research was presented, these themes will be continued in this chapter, where the design of the system will be presented. The first section will look at the software methodologies employed in this project which describes which methodology was chosen and why. After that, some example use-cases and user personas will be presented. The next section outlines the technical architecture of the system and will discuss in depth how the system architecture is planned to work. This will cover both front-end and back-end aspects of the system. The final section will discuss the plan for testing and evaluation of the system.

## 3.2. Software Methodology

Waterfall Methodology

The Waterfall Methodology is a linear and sequential approach to the development of software. Waterfall Methodology follows an 8-step approach in a development cycle.

Diagram

Description automatically generated

Figure 3 (27)

The benefits of the waterfall model are that it will be easy to use and manage; it will give the developer and users a clear idea of the scope, which would be beneficial for the time constraint for this project.

A big disadvantage of using waterfall for this project would be that there will be changing requirements during the development of the system as end user feedback is recorded. It is unlikely that we can be sure of all the functionality of the system, especially when the main function of the system is speed and ease of use. By the nature of the Waterfall model, once a stage is completed then it should not be revisited. There is no way to be this sure of the requirements at such an early stage.

The app will have evolving needs as research and feedback will take place for the duration of the project – Waterfall Methodology will not be suitable for the project.

Agile Methodology

Agile Methodology follows an incremental approach. This approach will allow the developer to be more flexible with the development of the project. In agile methodologies the project is broken into small chunks which are completed in “sprints”. At the end of these sprints project priorities are revaluated and the design and development plan for the project is changed accordingly.

Shape, arrow

Description automatically generated

Figure 4 (27)

Agile methodology is well suited for the project as changes during the implementation stages are expected to occur; testing can occur through all development stages meaning bugs are caught during the development instead of at the end; development is more user-focused, and feedback can be gained from the contacted organisations after certain sprints to ensure the project is up to standard.

With Agile Methodology it may be easy to stray off the initial project plan, so it will be important to check back to the original plan and keep it in mind.

An Agile Methodology is more suitable for this project, a combination of feature driven development and test-driven development. This will make it so the development focusing on one feature at a time, testing as I go to make sure everything works. For the development of this project there will be an emphasis on short development cycles.

## 3.3. Overview of System

A feature driven development (FDD) approach will be used where a feature will be planned, implemented, and tested as part of a sprint. Once that feature is implemented, development of the next feature will begin.

This approach is necessary for this project as features are the main driving point.

Design and code will be delivered in stages. A feature will be through designed and researched before implementation begins. It would not suit this project to complete all the design up front followed by all coding.

The current general approach for the project will be as follows:

1. Design and implement basic version of the image processing algorithm to show that it can work.
2. Design and implement the application with just a basic ui
3. Design a feature (for example taking the picture)
4. Implement the feature
5. Test the feature
6. Repeat steps 3-5 implementing all desired features – implement by priority of the features.

The application is primarily a 1-Tier application, although with the algorithm running where the user can’t see it and the ability to store and access old images, it could be considered 2-Tier under some circumstances. For the ease of documentation, it will be explained as a 2-Tier system

## 3.4. Front-End

The presentation tier is the front-end layer in the system and consists of the user interface. This layer will allow the user to interact with the application through the mobile phone and display content to them. Here the user will use their phone to take pictures and look at results

Paper Prototypes were used for the first iteration of the screen layouts and to provide allow fidelity prototype. These screen mock-ups were useful as they could be shown to potential users and give them a feel for the desired end product.

A picture containing diagram

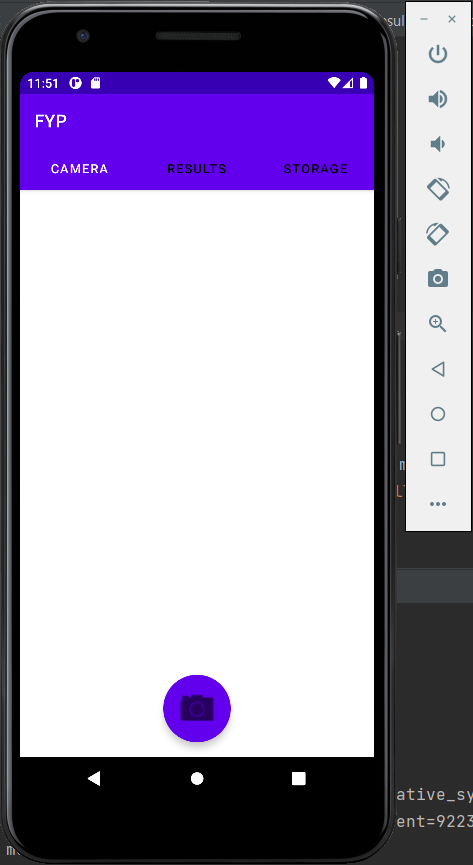
Description automatically generated

Figure 5

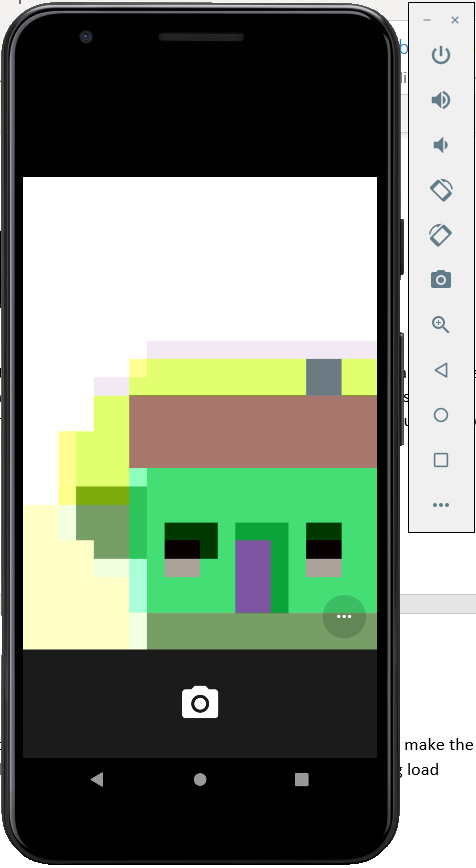
Beyond this there was very little need for a more detailed prototype as developing it within android studio would take the exact same length of time as a medium fidelity prototype but would result in something functional and usable. For that reasoning below is the final design of the app

## Final Design

I decided to use tabs in the application. This is because I felt the tabs gave me the cleanest look and worked most similarly to what I had originally imagined when designing the app. I decided to give the app 3 tabs as before in the low fidelity diagram, however I moved the tabs to the top of the screen as I felt this design was a little more user friendly and a little more inline with the industry standard for similar applications. I screenshot of what the app looked like when I first got it running can be seen below



As you can see there are three tabs up the top for the three different sections of the app. There is a camera button that redirects the user to the dedicated camera app that comes pre-installed on the phone. I chose this over implementing an actual camera feature within the app as I found this would only add on more of a load that the system needs to process, and when the goal is to make the app return a result as fast as possible, it’s much better to give the app a lighter processing load. By pressing the camera button down the bottom it brings you to this camera.



From here, the user can take a picture of the coins, where the phone and dedicated camera app will then take this picture and feed it back into the application. From here, the application saves the picture to the device, and then takes the image and starts to work with it.

## 

## Use Case Diagrams

Use case diagrams are used to identify system functionality and communicate system behaviour. The use case diagrams in the below figures show the progression of the system functionality.

Diagram

Description automatically generated

Figure 6

Diagram

Description automatically generated

Figure 7

## 3.5. Middle-Tier

The middle tier or application layer contains the functional logic which will drive the application’s core capabilities.

The middle layer will provide the underlying functionality for users to switch between screens; analyse images; view saved images; and much more.

For example, Java objects will be implemented to provide access to the camera. These objects will communicate and interact with the camera allowing for an image to be shown to the user. These objects will also assist in loading up old images.

## 3.6. Conclusions

In this chapter we looked at the design of the system, first exploring the methodology that will be used in the development process, next a broad overview of the technical architecture was outlined, then into detail on the design. Finally testing was discussed regarding the different testing methods that will be followed and the software test plan.

Based on the key themes discussed in this chapter, the next chapter will cover the development process and will be revisiting many of the same issues covered here. The development chapter will discuss how these designs were implemented including any challenges or changes encountered along the way.

# 4. System Development

## 4.1. Introduction

This chapter continues with the issues explored in the previous chapter and will outline the development process undertaken in this project. This chapter will present the key development processes and the challenges encountered during the creation of this system.

## 4.2. System Development

The first step for the development of the application was to set up version control. It was decided that GIT was the best option for this project as it is easily integrated with Android Studio. Bitbucket would be used to manage the version control using GIT as it is a reliable web-based hosting service. After that project was created and linked to GitHub and thus the development could begin. The link to the github repo is https://github.com/Jamoran3579/FYP

For the initial prototype it was decided that the goal was to have a functioning version of the algorithm developed in python that could take in an image and find the coins within the image and evaluate them to a certain extent

## Initial Prototype

The algorithm was developed as a Jupyter Notebook using the opencv library.

Graphical user interface, text, application, email

Description automatically generated

Python code was used to develop the algorithm and it was run through Jupyter.

In order to test the code, I had to find an image online which could be ran through the algorithm

A group of coins

Description automatically generated with medium confidence

As can be seen by the above image, not only was the algorithm able to detect exactly where each coin was located, but it also managed to detect the value of each coin perfectly. This result was then output directly to the user as can be seen below

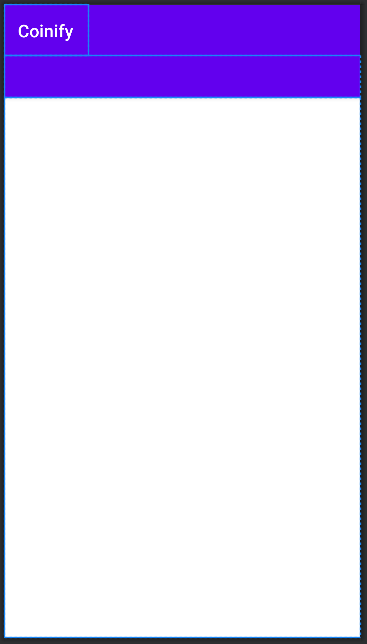
Graphical user interface, text

Description automatically generated with medium confidence

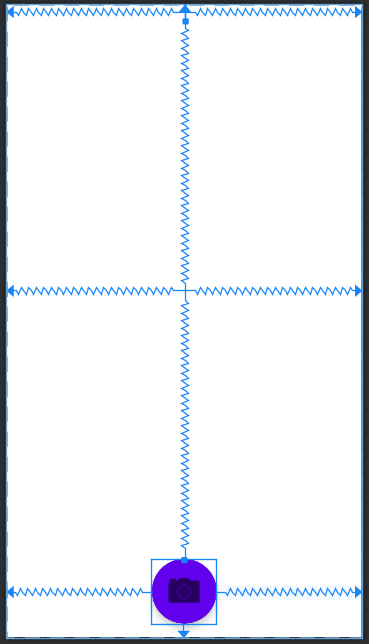
It does have a minor issue where some of the parameters within the Circle detection algorithm have to be altered depending on the light level or size of the image, but these things can be added when moved to the mobile app through the use of a phone’s light sensor or through analysing the size of the image

## Front End Development

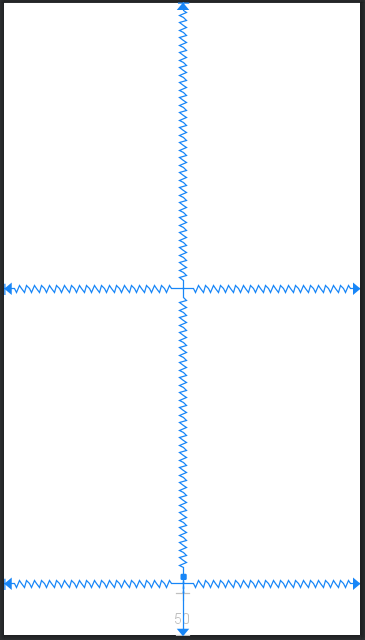
The next stage was to develop the front-end side of the application. I had already decided how I wanted the app to look/function, so I had hoped this wouldn’t take too long. I started by creating a blank tabular project in android studio, which comes with 2 tabs already created. As I wasn’t very knowledgeable with android studio, I had to do some research into how these tabs worked, so I could develop separate pages for each tab, and so that I could add a third tab to the page. This involved creating and editing a file called SectionsPagerAdapter.java, which is a file used for getting different fragments and titles depending on which tab you have open. I decided to have 3 total tabs, Camera (where pictures would be taken and confirmed), Results (where the user can view the results of the analysis) and Storage (where the user can view past results). Each tab was designed as an xml file, which could be called on by the PagerAdapter whenever a tab was changed. Below is a screenshot of each of the pages and its contents.



This is the main activity for the application. I wanted it to be as simple to understand and view as possible. The clearer and easier for new users to understand the application, the better. In the top left corner of the app is the name of the app, Coinify. Below this is a small purple bar which will hold all of the tab titles. Then below this is a large white box, which holds each xml file as it is loaded in.



The first xml file I will cover is the camera\_tab\_layout.xml. This file contains the contents of the camera tab. There are three main components to this tab. Firstly we have a TextView up the very top, which will show the user the results of the image once analysed. I will delve deeper into why this is necessary a little later. Secondly, we have an ImageView, which can be seen at the very centre of the screen. This is used to show the user what image was taken, to make sure they are happy with the image. Finally, we have the FloatingActionButton at the bottom of the screen. This was placed to float just slightly above the very bottom so as to make the look of the page slightly more appealing to the user. The image of a camera is a built-in icon to android studio, and I felt it better described what the button did rather than using text. Lastly both the button and image were sized up slightly in order to make it easier for the user to see.



Next, we have the results\_tab\_layout.xml. This page only has 2 main features. The first of these is an ImageView, to show the users the resultant image after the analysis so they can see if any coins have been missed by the algorithm. Secondly, we have a TextView down the bottom. This is to display the results to the user after analysis. For the same reason as the tab above, this was placed to float slightly above the bottom of the page so as to make it feel slightly cleaner and more user friendly. With the goal of this app focusing primarily on the user experience and usability of the app I felt that adding little things to make the app easier to use and read and to make the app feel cleaner was important, as by increasing the quality in the look of the app users tended to be happier to wait slightly longer to get a result.



The final xml file I will cover is the storage\_tab\_layout.xml. This is the tab where all of the old images that have been scanned will be displayed. As you can see above, because of scheduling errors and unforeseen problems, development of this feature was halted until a later date. The idea will be for it to house multiple little fragments, each with an image title, the total result, and a small thumbnail image that the user can view. The page will more than likely be scrollable, allowing the user to scroll through all past results.



Above is a screenshot of what the application looks like once it has returned a result for the user. As you can see there is an image displayed to the user, with the total amount found within the image displayed at the top. Above that, all of the available tabs are visible, with the currently selected one in white.

## Back End Development

The next thig that needed to be developed was the backend of the app. While there isn’t a full database required, the app still needs to be able to store photos that are taken and access them. To do this, a FileProvider was implemented within the app. I then have a function called createImageFile() which will create a temporary file that can then be used by the FileProvider to save to the phone.



Above you can see a screenshot of the code. The name of the file is built from the date and time in which the picture is taken. We then find the storage directory on the phone in which this app is allowed to store files, and set that to the storageDir. File.createTempFile() is then called in order to create a temporary file using the parameters we have provided. Once this file is created, we take it’s path and save it, allowing us to save content to this location, and then return the temporary file to where it was called so it can be worked with.

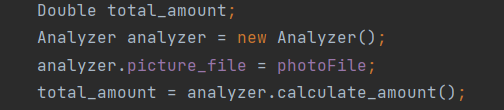
From this point, we use the FileProvider to get a URI for the file we are working with. We then provide this URI to the Intent that is requesting a picture be taken by the camera. By doing this, there is no need to return the image to the application from the camera. Instead, when the camera takes an image that the user accepts, it is saved directly to the location we selected by providing the URI. Then, when the user is returned to the app, they have access to the picture as the app has the path to the image saved as a plaintext string. The app can then check to see if the photoFile exists,



and if so, it can move on to analysing the image. From here the camera tab creates an implementation of the analyzer Object. This is a java class that I have created, specifically designed to analyze an image and return a result.

## Analysis Development

I decided to develop the analyzer in an entirely separate class so as to separate out the work that was being done by the app. To start with, I wanted the analyzer to be able to be an instantiable object that can be used by whatever class wants to use it, so I instantiated an instance of the analyzer object within the camera class. From here, I set the image within the analyzer to be the one at the path that I have saved as a String from earlier. I then run the calculate\_amount() function which will return the total amount of money found in the image I created earlier.



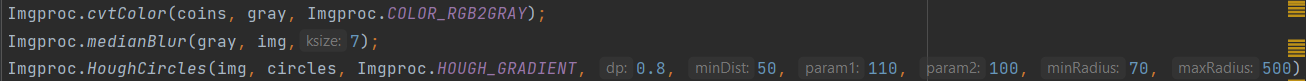
This is how the analyzer is interacted with from an outside point of view. Next I will delve into the functionality of the analyzer

## Analyzer Functionality

There are 2 functions within the anayzer class. The first of which is detect\_coins(). It takes an image “File picture\_file” which is pre-defined by the user and returns a MAT containing all coins found within the image.

**Image Augmentation and Detection:**

There are 3 main functions from the Opencv library that are required to be run on the image in order to detect the coins.



cvtColour() – This function is used to take the original input image, and convert it to a grayscale image. This is necessary as the function for circle detection will only work on grayscale images. Provided to it are the original image, the object in which you wish to store the new image, and the colour conversion being used. For this application, while the documentation says images will be loaded in BGR, I found the best results are achieved using an RGB-GRAY conversion. After some research, I discovered this was due to the fact that the image is saved to the phone, then loaded back into the app so it is loaded in as a normal RGB image.

medianBlur() – This function is designed to add a level of blur over the image and to artificially lower the quality of the image. The idea behind this is it removes some of the much finer details from the image, which makes looking for general shapes a lot easier as you are less likely to pick up on minor details such as chips or scrapes. Provided to this function is the grayscale image, the object it is being saved to, and an integer size of the blur tool. After some analysis and research, I found the app responded best to a blur of size 7 in most light levels, although the potential for this to be variable in future iterations is definitely there.

HoughCircles() – This function is what is used for the actual circle detection. This function has a lot of variables provided to it. For starters, the grayed and blurred image is provided, and then an object to store the results. Next is the algorithm that will be used for circle detection. I found that for most light levels HOUGH\_GRADIENT worked the best. Next is a float value which represents the scale at which the detection will work. As the app is designed for newer phones with higher quality cameras, producing images with more pixels, I went with a scale of 0.8 so as to ensure to get even the smallest of circles. This could cause slight issues on older phones, however the app cannot even be installed on phones that are too out of date, so this issue should not occur too often and is easily fixable in a future bug patch. The next value provided is an integer to represent the minimum number of pixels that circles are required to have between their centres before it is considered a new circle. I found this was important, as for coins such as €1 and €2, they have a perfect circle in their centre as part of the design.



It was also great at preventing shadows of the coin be detected as these circles would have almost identical centres, most only being off by 10-20 pixels. I went with a value of 50 here so as to allow stacked coins to still be detectable, but to prevent this issue.

The next 2 values were Parameter1 and Parameter2. These were 2 of the more important values and are sort of the bread and butter of how the algorithm works. This algorithm uses an implementation of the canny edge detector in order to find the circles, so parameters for the canny edge detector have to be passed here. The first value is the higher threshold of the canny edge detector, with the lower threshold being half of this value. This value works best sitting somewhere around 100, with higher values increasing the ease in which a circle is detected.

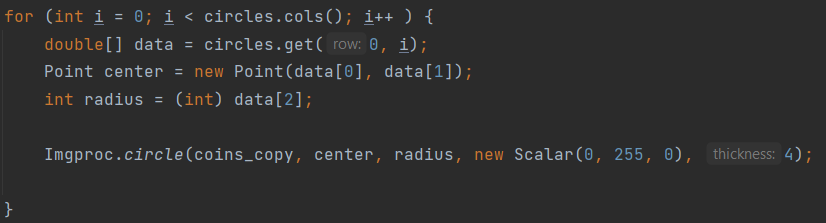
Parameter2 represents the accumulator threshold for the circle centres at the detection stage. This can almost be seen as a “perfectness” measure, and the higher this number, the more perfect the circle has to be for it to be counted. If the value is too low, false circles will be returned. If it’s too high, some coins will be missed. For most light levels, 100 seemed to be the optimum value, although a good value here can range from 75 – 125 depending on the clarity of the picture. Unfortunately, these 2 parameters require a lot of changing and variation, so a high priority in a future patch would be to include a way to allow the user to change these values, or to use some automatic detection of things such as gama, distance, etc. to change these values.

The final 2 values passed to the function are the minimum and maximum radius of the circles. It was important to leave a large distance between the values as I wanted to allow the user to have as much range in the distance they take the picture from and found a range of 70-500 pixels worked perfectly for the radius.

With these 3 functions, I got an output Mat containing all of the detected circles, with their centres and radiuses. This could then be looped through, giving me access to each individual circle

**Drawing of circles:**

From this point I created a copy of the original input image, and looped through the list of circles, drawing them on the image using the circle() function.

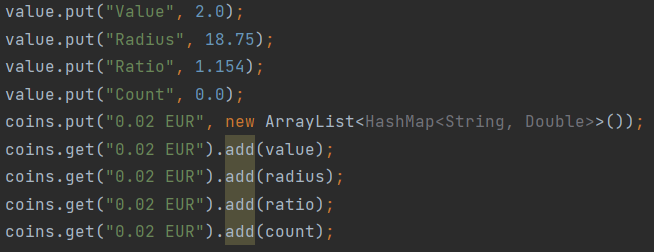


Provided to this function is the image I wish to draw on, the centre and radius of the current circle, the colour I wish to draw with (green), and the thickness of the line in pixels. This looped through all of the circles, each time extracting the data from the Mat() circles for the next entry. As most of the filtering was done at the detection stage, there is no need to sort the circles by size or to have any worry about the size of the circles.

The end of the function just involves writing this image to the file from earlier, replacing the original image, and returning the Mat circles so that this data can be used elsewhere.

**Coin Data Storage:**

Next I have the function calculate\_amount(), which is the only other function in the analyzer class. This function is designed to take a list (Mat) of circles and calculate the value of all the coins in the image, returning a double containing the total amount found. To start with, I had to create a way to store the data for each coin, such as its radius and its ratio in size when compared to other coins. I decided to use the 1c as my base coin and compare all other coins to it as it is easy to sort a list and get the smallest item. An implementation of this storage would have been a lot cleaner in Python, and while I considered it and just passing the image to a webserver running the python code, I felt that a standalone running app would be better for the functionality I want. In order to create this storage, I had to do some research into HashMaps and ArrayLists in java. Eventually with this understanding, I created a HashMap “Dictionary”, where the first value is a String containing the name of the coin eg. “0.01 EUR”, and the second value being an ArrayList. This ArrayList then contained a second HashMap, with the first value being a string denoting what the current entry denotes, such as “ratio” or “count”, and the second value being a double which represents that value. While I’m aware this is a very crude and awkward implementation, this was unfortunately the easiest way to do it from within Java.



Each coin had a set of instructions to run like the one above, which will make sure each individual value that is required is saved, and then store it in the appropriate location within the HashMap which was named coins.

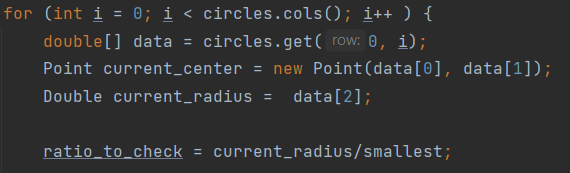
From here a new Mat is created called circles, and the detect\_coins() function from above is called to return the Mat circles, which we can now work with

**Circle evaluation:**

There are multiple steps involved in the evaluation of each of the circles. To start with, I had to find a value to compare each coin’s radius against. The easiest way to do this, was to assume that the smallest circle found within the image was a 1 cent coin. This does require the user to have a 1 cent coin within the image anytime they want to use the application, however I found any other method of value detection caused its own insane number of problems. Later I will delve deeper into this problem, but as for now, I’m explaining the functionality of the app. In order to assign this value, I created an array with all of the radius values.

This array was then sorted by size, making the first value the smallest. This value was then saved in a variable called smallest and was used for later comparison. The reason I decided to go with ratios of this radius when compared to that of a larger coin, is because if a user takes a picture from far away or close up, the size of the radius will change as there will be more or less pixels within the image. However, the ratio of radiuses will be constant, giving the user more range to play with when taking the picture.

From this point I had to loop through the circles Mat containing all of the circles and their data.



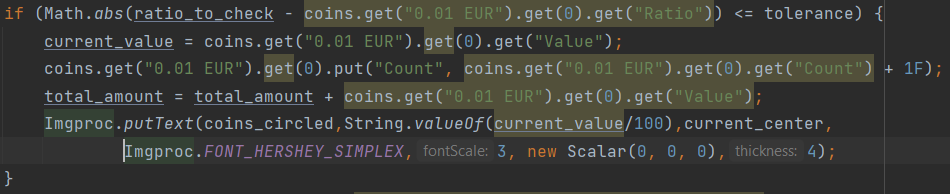
I pulled out both the circle centre and circle radius so I could figure out where this coin sits. From here, I set a variable called ratio to check equal to the current radius divided by the smallest one we found earlier. This gives me the ratio of the radius of the circle I am currently working on, which I can use to compare to the official ratios found at – “<https://ec.europa.eu/info/business-economy-euro/euro-area/euro-coins-and-notes/euro-coins/common-sides-euro-coins_en>”. I also included a small amount of tolerance should the coin be damaged, the detection algorithm not work perfectly, etc.

In order to find out what the value of the current coin was, I had to use a long set of if{}if else{} statements as unfortunately within Java, the ability to use a switch statement on a double is not available.

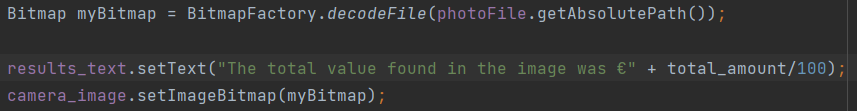


This if statement does a number of things. Firstly, it will get the “Ratio” value for the current coin we are looking at from the coins HashMap, in this case, it is the 1 cent coin or “0.01 EUR” entry. Once we have this ratio, we then subtract this from the ratio for the coin we currently want to check. If the circle we are working on is the value of the coin we are currently testing, the value should come out around 0, with the potential for some small variation plus or minus 0.03. Then the absolute of this value is calculated, telling us just how far away from the 1 cent ratio it is. From this point, we check if this is within our tolerance range of 0.03. If so, we perform a number of actions on this coin. If not, we move onto the next coin.

Once a coin has been found which’s ratio is within the tolerance of the one we are checking, I then have to take a few actions



Firstly, I save the value of the current coin I am working with, into a variable called current\_value. I then increase the count of the coin I am currently working with by one. From here, I increase the total amount found by the value of the current coin. From here, I then write on the image the value of the current coin, using the centre of the circle provided earlier. This is written in black, the same size as the circle. Next I write this image to the same file path as before, and return the total amount that was found within the image to where calculate\_amount() was called, in this case, back in the Camera class.



From here, a Bitmap of the image is gotten. The total amount is then put onto the screen where the results\_text box sits, and the bitmap is set as the camera image on the camera Tab.

## 4.3. Issues and Changes

I faced a number of challenges during the development of this project. These ranged from lack of knowledge within a certain field prior to development, all the way to issues with not finding solutions in documentation to problems that occurred. One thing I would change from the get-go is the order in which the different components were developed. As I studied image processing in Semester 1 of 4th Year, my focus during this Semester was the coin recognition algorithm. Realistically, the app should have been designed first, and if the project was to be re-done, that is the route I would take.

I also had a number of issues with android studio, and more specifically, the camera module. The way the app was designed was almost from the top down, meaning somethings were set in stone, such as the use of tabs and fragments, before even looking at the camera module and finding it has difficulty with this. In future, some research into all components and their compatibility would be paramount.

I also faced an issue with android studio when I was importing the opencv module. Unfortunately however, this issue could not have been solved with prior research. A lot of the issue here was caused by the previous configuration of android studio, however the main problem was caused by the documentation for the installation of modules just not being up to date with the current version. This problem took a lot of work to fix, and in future, working on the project from a clean install of all applications and modules would definitely be the way to go.

Another issue I faced was just finding time in general to work on the project, as having a lot of assignments plus working what was essentially a full-time work week made it very difficult to manage time in a way that would work. Setting out a timetable and schedule right from the get-go would definitely have helped to manage the time a little better and allocate more time to working on the project early on.

I also faced a number of issues health wise. These ranged from injuries to illnesses, and it definitely made working on the project more difficult. Unfortunately, these kinds of problems are unavoidable and in future iterations of the project or if I was to re-do it, I would imagine this issue would appear far less often

Another problem I encountered was the accuracy of the detection algorithm itself. This was a major problem and was one I discovered very late into development. The built-in function in the opencv library called hough\_circles was the main function that gave me problems. These accuracies of this algorithm just weren’t up to the standard of an application such as this one, so if I were to re-do the project, my main focus would be on developing my own detection algorithm or finding a different algorithm to work with. This change in it of itself would have a major affect on the usability of the project and would create a much better user experience.

Another problem that was encountered was the use of tabs. These made it so that most things within the application had trouble working. Anything from moving data from one tab to another, to returning results from the camera application. Everything I did while using the tabs required some work around as there were always problems and very little documentation

Also if I was to re-do the project, I would probably focus on more of a machine learning/AI method of image recognition and coin detection. While I did enjoy working on the project the way I did and it did return accurate results, I feel AI has reached a level of accuracy and efficiency now that it is not too difficult to implement, not too load baring on the device, and still gives accurate results in a relatively fast timeframe.

Another change I would make if I was to re-do the application is I would host the analysis tool on a webserver, making the client a lot lighter. This way the user can just take a picture, which is then sent off to a webserver, and a result is returned to the user. This would also allow me to work with languages and tools I was a lot more comfortable with for the analysis, therefore creating a tool that is not only more accurate but also has more functionality.

I also encountered a problem where I accidentally deleted the entire application at one point. Luckily, I had the app backed up in numerous locations, however some config files within android studio were also removed, which were not included in the backup. This meant that I had to perform a fresh install and copy and paste the contents of each file from the backup in order to fully recover the app. If I was to repeat the project, it probably goes without saying that I would try to avoid deleting the entire project, however the inclusion of version control and backups through the use of git definitely made it possible to recover.

## 4.4. Conclusions

In this I continued with the issues explored in the previous chapter and outlined the development process undertaken in this project. This chapter presented the key development processes and the challenges encountered during the creation of this system. While the problems were numerous and common, most of the issues encountered I found some way to work around them or just remove them entirely. I have also explored a small bit in this section about the research that went into creating this system, and some very fine detail about specific sections that I felt were important to explore in greater detail.

# 5. Testing and Evaluation

## 5.1. Introduction

In this chapter I will outline my plan for testing the project, as well as design a short test plan for a few tests that a feature has to pass in order to be considered working and to move on to a new feature. I will also talk about how I plan to evaluate the project as a whole and how I plan to analyse the project at the end

## 5.2. Plan for Testing

The Project was continuously tested throughout its entire life cycle. The app was run on both emulators and real android phones throughout the development stages to ensure that it is working as intended and to provide hands on user trials.

Backing up and committing the project consistently using GIT version control ensured that any changes can be rolled back when there were major errors. Having multiple project backups helped in the inevitable “accidently deleting an important part of the project”.

Another plan to evaluate the project was by asking test groups to use the app. These groups consisted of a variety of possible end users, including people from the hospitality industry such as my co-workers or people in the retail industry/ service industry.

Unit testing was used as there were so many individual parts to the project that make up the whole app. Unit testing is where tests will be created which interact directly with the application. These tests will validate each unit of the software to ensure it is performing correctly. Unit testing is essential for the proposed agile approach as after each section all parts of the program created will be tested.

Blackbox testing will also be used as it provides a different angle when testing the project. The tester will not know about the internal structure/ design/ implementation of the item being tested. The black box testing method will attempt to find errors in areas such as: Interface errors, performance errors, incorrect functions, initialization errors and more.

It was decided that a mix of both unit testing and black box testing would be an adequate way to test the project. Manual testing will take place too wherein the developer will play the end user and use the app features to ensure correct behaviour.

Evaluation of this project was done by the developer and potential user.

## Test Plan

|  |  |  |  |
| --- | --- | --- | --- |
| **Test No.** | **Test Description** | **Expected Outcome** | **Pass?** |
| 1 | Does the phone app load when the icon is selected? | The app will load and bring the user to the camera screen. | Yes |
| 2 | Does the camera open correctly? | Once permission is given by the phone user, does the camera display onscreen. | Yes |
| 3 | Can a user take a picture? | A picture is taken, and it is shown to the user. | Yes |
| 4 | Can the user analyse the image? | User presses analyse and the image is put through the algorithm, returning the result. | Yes |
| 5 | Can a user re-test an image? | The image is fed back into the algorithm using different parameters, resulting in a new result. | No |
| 6 | Can the user save an image? | Image is saved to device storage. | Yes |
| 7 | Can the user view old images? | User presses to view storage and all past images and results are shown. | Not yet implemented |
| 8 | Can the user delete images? | User presses delete and the image and corresponding data is removed from device storage. | No |

## 5.3. Plan for Evaluation

Evaluation of this system is equally as important as testing. The reason for this is that user experience is one of the main complexities of the application. On the surface we want the application to be as easy to use as possible, even though underneath this there is a lot of complex code and decision being derived. As previously mentioned, the usability of the system being intuitive and instinctive was of high priority through the entire development process. Having the system be evaluated by potential users would hopefully ensure that the system usability is of high quality and of an industry level standard.

This section will go into detail as to how the system was evaluated. Evaluating the system was done by surveying potential users (including co-workers) and by evaluating against the predefined metrics of Nielsen’s Heuristics for User Interface Design.

1. Visibility of system status

The system is very good at informing the user as to what is currently going on, however minor improvements could be made, such as when analysing the image, a small notification could be given to the user to inform them of this

1. Match between system and real world

Any data that is read or interacted with by the user is all done using plain English or numbers rather than referring to parts using the programming terminology or technical language

1. User Control and Freedom

Any parts of the application that a user could enter by mistake have a clearly accessible exit option

1. Consistency and Standards

All parts of the application follow a strict set of rules or templates. One word does not correspond to multiple features, the same way one feature does not have multiple words that refer to it.

1. Error prevention

It’s very hard for an error to occur within the app, and should an error occur, the app handles the problem cleanly and gracefully

1. Recognition rather than recall

The user will very rarely have to remember what a certain button does or the particular functionality of a certain feature, rather everything is labelled well and recognisable that most of the time the user can make out what its function is

1. Flexibility and efficiency of use

As the system is such a barebones application and is designed around speed and efficiency, there is very little that can be done by an experienced user in regard to shortcuts, as every input is functional

1. Aesthetic and minimalist design

The application tends to keep any and all irrelevant information away from the user and gives them a nice clean environment to work with.

1. Help user recognise, diagnose, and recover from errors

As the system is very good at avoiding errors by itself, there is very rarely a time when the application requires a user to recover from an error

1. Help and Documentation

Unfortunately there was no way to include documentation as to the functionality of the application, however, even in the black box testing, user’s never required any sort of documentation and were very easily able to figure out the applications functionality

## 5.4. Conclusions

This chapter reviewed the testing and evaluation of the system. The testing consisted of performing Unit Testing, Integration Testing and Security Testing both while developing the system and on completion. The evaluation included evaluation from a variety of potential users and evaluating the system against Nielsen’s heuristics.

# 6. Conclusions and Future Work

## 6.1. Introduction

In this section I will mainly be focusing on any future work I plan to do with the project, and any future development or ideas I have to expand the project.

## 6.2. Conclusions

There are a few points I would like to make as the conclusion to this project. Firstly, throughout this project I have developed a much greater respect than I already had for people who have to work with the front end of a system. It causes challenges at almost all twists and turns, and it is definitely something I would have to research a lot more before doing any more work with it. I have also learned that when working with libraries such as OpenCV, the language that it was originally created for (in this case Python) is by far the best language to work with it in. This is because the documentation will be up to date and be most detailed for this implementation of the library. I have also come to the conclusion that while fixing problems using websites such as StackOverflow is definitely the most common way people in the industry/college solve problems, it is by far the least reliable. Provided whatever language you are using has good, clear, easy to understand and up to date documentation, it is by far the first thing I will look at in the future. The last conclusion I have come to is to be more careful in the allocation of time during the development of a project. A lot of the issues I encountered, ended up causing delays in certain features, and these delays were not accounted for in the time allocation. This then caused problems as other features that had been intended to be included were then not given enough time to be worked on. In any future projects that I work with, I will be careful to allocate time in a more manageable way so as to allow time for fixing problems or doing research.

## 6.3. Future Work

There are a number of things I would like to add and change about the app going forward. For starters, I would like to use a different coin detection algorithm develop my own algorithm. This would allow for a higher level of accuracy while using the app and would massively increase its usability. If developing my own algorithm, I would like to do some research into developing a genetic algorithm, so as to make it more accurate and to get more and more accurate in the future. I would also in the future like to move the analysis side of the application over to a webserver that can be accessed by the application, therefore reducing the size of the application and making it more accessible for older devices. I would also like to get both the results and storage tab working, with the ability to retest old images, delete them, and generally just complete everything I set out to do from the beginning of development. These features were designed to be part of the application from the get-go so they will be the number one priority in regard to work being done in the future. The last piece I would work on in the future is to expand development over to IOS devices as well. This would involve some research into the react native language and would require a decent bit of learning. This would be important for the project as it would open up the app to the other half of the mobile market, however something I did notice is that when it comes to people in the service industry, there is a large bias towards using IOS phones, and found from my own experience that it is about 70% of users in this industry use apple devices. This change would open up the application to so many other people, and I feel it is essential to be done in the future should the app be successful

# Bibliography

1. • Mobile OS market share 2018 | Statista [Internet]. [cited 2018 Nov 26]. Available from: https://www.statista.com/statistics/266136/global-market-share-held-by-smartphone-operating-systems/

2. iOS: What you need to know about Apple’s mobile OS | finder.com [Internet]. [cited 2018 Nov 27]. Available from: https://www.finder.com/ios-operating-system

3. Agile &amp; Waterfall Methodologies – A Side-By-Side Comparison | IT Staffing &amp; Technical Consulting [Internet]. [cited 2018 Nov 28]. Available from: http://www.base36.com/2012/12/agile-waterfall-methodologies-a-side-by-side-comparison/

4. Scrum or Waterfall - Which Methodology Best Suits Your Project Requirement [Internet]. [cited 2018 Nov 28]. Available from: https://www.mindinventory.com/blog/scrum-vs-waterfall-methodology/

5. Black Box Testing - Software Testing Fundamentals [Internet]. [cited 2018 Dec 2]. Available from: http://softwaretestingfundamentals.com/black-box-testing/

6. Coinoscope: Identify coin by image [Internet]. [cited 2021 Dec 15]. Available from: https://play.google.com/store/apps/details?id=com.miccron.coinoscope&hl=en\_IE&gl=US

7. CoinDetect: Euro coin detector [Internet]. [cited 2022 Jan 01]. Available from: https://play.google.com/store/apps/details?id=com.miccron.coindetect&hl=en\_IE&gl=US