UOW MALAYSIA KDU PENANG UNIVERSITY COLLEGE

BACHELOR OF COMPUTER SCIENCE (HONS)

PROJECT TITLE – [SMART CABINET SYSTEM]

GO YIK YEK (0188039)

JANUARY 2021

UOW MALAYSIA KDU PENANG UNIVERSITY COLLEGE

BACHELOR OF COMPUTER SCIENCE (HONS)

PROJECT TITLE – [Smart Cabinet System]

Student name: Go Yik Yek
Student ID: 0188039
Supervisor Name: Dr. Khoo Hee Kooi

A dissertation submitted in partial fulfilment of the regulations governing the award of the degree of Bachelor of Computer Science (HONS) at University of Lincoln and UOW Malaysia KDU Penang University College

Application Project
JANUARY 2021

DECLARATIONS

I declare the following:

(1) that the material contained in this dissertation is the end result of my own work and that due acknowledgement has been given in the bibliography and references to **ALL** sources be they printed, electronic or personal.

(2) the Word Count of this Dissertation is .....13,531…(Excluding Codes).....

(3) that unless this dissertation has been confirmed as confidential, I agree to an entire electronic copy or sections of the dissertation to being placed on the eLearning Portal (Canvas), if deemed appropriate, to allow future students the opportunity to see examples of past dissertations. I understand that if displayed on eLearning Portal it would be made available for no longer than five years and that students would be able to print off copies or download.

(4) I agree to my dissertation being submitted to a plagiarism detection service, where it will be stored in a database and compared against work submitted from this or any other School or from other institutions using the service.

In the event of the service detecting a high degree of similarity between content within the service this will be reported back to my supervisor and second marker, who may decide to undertake further investigation that may ultimately lead to disciplinary actions, should instances of plagiarism be detected.

SIGNED:

[student signature]
Go Yik Yek
Date: ____15 April 2021_____

ACKNOWLEDGEMENT

First of all, thank you to UOW KDUPG for giving me this opportunity to work on my final year project. This is a once-in-a-lifetime opportunity that has allowed me to gain not just knowledge but to build a strong mind with a liberal attitude. Propitiously, UOW KDUPG has provided me space, resources, and precious chances that lead me on building this project.

Secondly, special gratitude to Ms. Nursakirah Binti Ab Rahaman Muton for guiding us throughout the individual project. Ms. Nursakirah is our course lecturer for IP1 and IP2 and Ms. Nursakirah has provided a lot of guidance on writing the IP1 and IP2 reports. In addition, thank you Mr. Danny Chen and Ms. Rosmah Ismail who both are my markers for my IP1 project proposal helped me to improve and perfect my proposal and my report writing skills.

Thank you, Dr. Khoo Hee Kooi, who was my supervisor for guiding me throughout the entire individual project. Dr. Khoo always patiently brings me valuable lessons in this project especially Python programming. Dr. Khoo has shared essential guidelines on improving my proposed system. I am gladful that Dr. Khoo was my supervisor who was spending time having a companion oversee each stage of development on my product. Exclusive gratitude to Dr. Joshua Thomas for having afford to witness my project as a second market in IP2.

Finally, I am thankful for everyone that has helped me, as well as those amigos, all passionate lecturers, and proactive campus staff, every moment of delightful life lesson that went through from the whole amazing study on campus. Which brings a huge contentment experience for me to move on. Besides that, special thanks to my family and friends for encouraging me all the time and keep boosting my vibe to do well on this individual project. Without each of them, the project development is impossible to carry out.

ABSTRACT

This is a Node.js merging with Python (OpenCV) project which targets to develop a smart cabinet system which will be based on IoT logic systems. The purpose of developing a smart cabinet system is to save on costs related to trips such as fuel and vehicle maintenance as making traditional lockers have connectivity on allowing pedestrians, travelers and businesses more options in picking a place to protect their equipment or product in public. The first method that will be used in this OpenCV QR code. The facial recognition algorithm that would be implemented into this project will be the LBPH. The user interface will be using express framework from Node.js, this feature allowing customer to unlock their allocated box and place their physical parcel or items according to their expected duration as storage, the another feature will be allowing the customer select target destination as there will be theoretically the placed parcel that placed by customer will be taking to the target destination by delivering staff and lastly letting the recipient self-service pick up by unlock with QR code. And customers will receive the QR code via email by default after payment succeeds. The CNN network for liveness detection will be implemented into this project adds a layer of security for facial recognition, this is to allow when customers have additional methods to unlock the case and also prevent photo spoofing by non legitimate users. There will be results of the testing of the API, overall features, and the unit testing of the system. In conclusion, the system process should be able to illustrate the entire system flow to launch and demonstrate.

TABLE OF CONTENT

Contents

CHAPTER 1
INTRODUCTION

1.1 Introduction

Smart cabinet locker is a substantive container technology impetus for automation public storing and superintending delivered parcels or packages. As e-commerce and online purchasing platforms are becoming new trends to ignite the explosive demand for advanced parcel lockers. The commercial facilities and multi-tenant apartments, multi-domain industries are highly relying on such technology to schedule the transport for parcels migrating to multiple recipients. Seamlessly, most of the logistics and e-commerce companies around the globe are starting to appraise toward investing in smart IoT lockers. For case in point, in April 2019, DHL Express had made a collaboration with SwipBox and PostNord to expand the smart locker network in Denmark. The companies seek through the potential of usage to and planned on expanding their service coverage across Denmark and continue receiving enormous revenues due to the high activity of the market demands of parcel lockers (Chris Dawson. 2019).

Nowadays, parcel shipment and high-frequency purchasing are becoming a norm for consumers having the right deal with business merchants that sell products online without constraint by relying on just physical merchants. As the usage of smart locker in Malaysia, still has not generally got attention by public or not been fully utilizing and even if there were existing digital locker may not have greater connectivity and flexible functionality provided. While during the internet age of rapid growth of e-commerce activity, the local small-and-medium scale business would face a new challenge of competing due to such small-medium scale businesses not being able to have much greater capital invest on the instant delivery compared to colossal shopping sites such as Lazada, Bangood (Vernon Chua, 2020). Hence consumers more likely to want to purchase stuff via online delivery as such enormous online sites may dominate the market which the local business may have an impact.

Individual customers usually purchase little amounts of products, seamlessly they will expect fast delivery especially when people have situations of the SOP social distancing for the pandemic in global. So there will not be any better option in the current situation as even the order that is made online may still need to wait at least for days or more than a week for a parcel to arrive (Angelin Yeoh, 2018). And since there are limited resources such as planning the delivery and not having the space provided to place the supplies for the delivery from local retailers. Another case study was local retailers may still have the goods that consumers need in fact, but they might miss out on the chance of having to deal with the potential customer due to reasons such as unable to do the delivery of the product directly.

As smart lockers here provide the solution, which is named Smart Cabinet Malaysia is available here on supporting features as temporary storage of shipments. And providing the solutions of the alternative to the way of storing parcels and letting consumers pick up their goods quickly or for storing their belongings in public. The goal here is also to make the business market more balanced and fair as even the competition for having a new option on delivering goods for suppliers and retailers by using the smart cabinet system.

1.2 Background of the project

Since the day having courier service in mankind, the importance of having a secure place to store, drop off or pick up for delivery has dramatically grown after the demands of online purchasing and traveling activities becoming omnipresent in every country. There are options such as deliveries to centralize the distribution point which was chosen by the courier service provider normally to randomly deliver service costs relatively low but causes many problems like packages missing or spoiled. Although in Malaysia there are places like the airport and intermodal transit-oriented stations that have provided such smart lockers, conventionally the only purpose of such kind digital lockers is just letting people store stuff and it. Most public lockers have a lack of selection such as providing choices likable delivering customers' packages to another location (LIN Han, 2017). Here will bring on the smart cabinet system solution which has a link with the backend and manageable including allocate staff to gain a new task to deliver the parcel from position A cabinet to destination cabinet B.

This report will introduce a low-cost and lightweight solution suitable for deployment with a smart cabinet system that integrates face recognition using the LBPH algorithm. Although face recognition was just an option for users to decide whether he/she wants to set facial recognition for temporary belongings or products storing purpose, the users will able to unlock automatically once they have chosen it, the facial recognition goes without inconvenient authentication process by using an automatic recognition procedure that uses face recognition to verify users and grant access to the cabinet box; for delivering service user will just unlock their boxes with QR code scanning since both pick-up and drop off destination that is dynamically chosen by the user. The ideal goal here is to build an integrated simple smart cabinet interface that allows cabinets to be frequently changed usage from multiple users without sacrificing the overall security.

Recently, researchers have concluded that smart lockers systems are capable of keeping several data points to create an audit trail (Tuesta. 2020). Staff can be held accountable for loss and/or upkeep by using an automatically smart locker. Paying sufficient attention to symmetrical processes and effective management inventory is mandatory with smart lockers, which can result in reducing waste and effective productivity and goods.

1.3 Proposed research methodology

Throughout my research on this project, it is important to study and learn about the growth truth of smart cabinet locker systems. In such logistical matters, society may eventually increase the chance which arises with the problem of the last mile distribution of goods movement from departure to recipients' doorsteps or specified pickup address. (Kushal Nahata, 2019) The primary problems of the last mile delivery are regard to the limitation of overhead expenses, insufficient security in public infrastructure, imperfect of service, no attestation of the safety of goods, inconvenient allocation of goods, lack of digitization, etc. Intending to unravel the fundamentals problem of the last mile delivery, instead of overage spending to improve or upgrade technical equipment the domiciliary experts are bringing out the suggestion on changing the operation fashion by composing with the integrated information system. Most of the public logistics facilities are considered outdated, with an extreme deficiency incapable of informatization, hence traditional systems will not be able to fulfill the provisional development of e-commerce. The essential point here will be solving the existing problem with a new solution, developing a smart cabinet system, and eventually enhancing consumer safety, accommodation, privacy regulation, on the other hand decreasing the cost of the last mile distribution ultimately.

The second research about this project is applying the CRUD, REST API call back function using Node.js. To calibrate the reasonable concept and to convert it into usable production, having research and finding proper references are important before being able to construct a blueprint according to the business logic. At provision, systematic analysis and all accountable factors will be required to list down, such as all the directional flow of use cases on how the target user can accommodate the system in reality. Including the step on before and after the customer had interacted with the system, the further step will be letting the user either scan to unlock the locker with their QR code or facial recognition. The project itself is not as complicated as expected, just that it genuinely has to fragment the problem into smaller pieces carried out and afterward having the fusion of tested features when these phases were applicable in good determination.

The methodology of the SDLC (System Development Life Cycle) that will be used for developing this project will be The methodology of the SDLC-(System Development Life-Cycle) that chooses on developing this project will be the Waterfall Module. The Waterfall module is more suited for developing an individual project since each task must be accomplished in a prefix timeframe after every stage must appear for review and affirmation by the lecturer. In assumption, once the testing has been done, there will be some feedback from my supervisor about the product updates. Thus, within the dateline will keep updating the product until the beta prototype matches all promising requirements. The completed system will summarize the fundamentals for developing the final product. A prototype is purpose-built with basic client features as the QR code scanning with OpenCV, LBPH algorithm, and Keras CNN liveness detection must be implemented into the prototype. Besides, this SDLC model will provide a more integrated end product due to the POC is based on the customer or tester's needs.

1.4 Proposed work

Smart cabinet system is the main role in this crucial project, it behaves as a self-service device Smart cabinet system isn't traditional lockers, robust. The program is designed to decrease wait times, improve customer satisfaction, and reduce high expenses by integrating smart locker-specific software, more undertaking as an automated smart locker deployment than one might comprehend (Transparencymarketresearch.com. 2021). According to research, most of the domestic and building in developed countries are highly relying on automated smart locker systems to receive delivery items. When the residentials owner is outdoors during daytimes, it helps save time for both customers and delivery service providers.

The work proposed is the development of a proof of concept for a smart cabinet system, here will divide into three parts of the subsystem, starting from the backend API, secondly the staff portal, and the client ware single page application (SPA) that can run on general operating systems such as Microsoft Windows. And three of the subsystems will be used to illustrate the whole program. Just that only the client ware will be included with Python code for image processing purposes for reading the QR code and also supporting the feature of facial recognition. Only the client ware and the staff portal will have the user interface which is structured with the MVC model. The purpose of MVC is for separating the route, functions, and HTML content making code more maintainable which also becomes tidier as even full-stack developers adopting the standard for the MVC model.

Talking about the data management, the backend API only serves for the client ware and otherwise for simple admin functionality. The staff portal is an independent just centrical focus on the feature for delivering staff to acquire their task and it works just like a site with login and logout. The reason why the staff portal is independent was, the site has to be accessible just only for those staff who prepare to work. The staff portal site server must be handling each of those sessions; unlike the client ware is usually stateless in terms when there are customer come to interact with the interface of client ware, the HTTP action only is called out as well using another kind of session handling knowns as a token.

In the program, to verify the customer and staff are the legitimate users for the unlocking purpose for the cabinet locker, the client ware will require OpenCV for handling image data from a digital camera. The reason is that to use OpenCV can be used in any platform which means this is compatible with any kind of operating system as well as programming languages that are widely adopted in the industry. OpenCV contains implementations of up to 2500 algorithms (Prithvi Dev. 2020). It is an open-source library for commercial purposes along with educational purposes. Within the project, OpenCV will be having the essential roles in decoding the message from the QR code and also having tasks such as facial detection. The system will be using HAAR cascade for normalization of facial images to get a preprocessed image while extracting image information. With the intention that accompanies the LBPH classifier during the recognition.

1.5 Definition of key concepts

1.5.1 LBPH

The Local Binary Patterns Histograms (LBPH) concept was introduced as early as 1994 by Ojala et al (Ojala et al., 1996). LBPH algorithm was commonly used for facial recognition. This algorithm is based on the local binary operator [19], The LBPH operator is a powerful feature for texture classification utilized for local binary features by taking account of the Local Binary patterns [21] which [1assists to shrink the local unique features vector from the proportion of face image. Give an example: To construct the LBP texture descriptor from a 3x3 grid of matrix as the output 2D array must be the same width and height for even greater scale in condition, the first step will convert the image into grayscale. Secondly, the LBP sampling is thresholding the binary ratio of neighborhoods' pixel intensities within the center pixel. It calculated by comparing with the center point and the case here will be computed whether that is greater-equal than the center which surrounds by eight pixels and becomes 8 binaries of LBP code. In the assumption of the size of an image in 256 pixels, it will be 8x8x256=16.384 positions in the final histogram. The final combined histogram will represent the characteristics of the source image. At the time of measuring the matching label or identity value from the test image, it requires comparing 2 images to find the closest histogram. Here will be relied on a distance measure by either Chi-square or Euclidean distance algorithm, then return a distance value is also known as a confidence value. The lower the confidence means that both histograms compared from the original image with the testing image will have the closer characteristic.

1.5.2 CNN

CNN was introduced by Yann Le Chun in 1994. As before understanding the convolution neural network, the basic concept comes from the multilayer perceptrons logically works like human brain synapses with connected nodes. Similar terms of the artificial neural network work by passing the value to another node, the nodes are only activated when reached a threshold value. The value that satisfies the threshold will further pass to the next layer to process. CNN network will consist of many layers. The number of layers will be depending on the purpose that each layer will persist on learning to detect different features of an image by recursively adjusting the weight values example shown in Figure g.13. In general, CNN consisting the convolutional layer, pooling layer, Relu layer, and fully connected (Keiron Teilo O'Shea. 2015) layer.

1.5.3 DBMS

A database management system (DBMS) is a program that allows organizations to concentrate data, effectively manage data, and provide reliable data access for application programs. DBMS behaves as an interface among software between physical drives. When an application program invokes a data file, whether an entity or relevant data in target row or just selected column depending on query statement and statements. DBMS will according to the query order to have 3 types of activities such as search, update, or delete. The SELECT command is for finding those relevant data from the database and return the result to the application (Meiryani. 2019). Instead of using traditional data structure solutions, a programmer does not need to take additional tasks

for considering the size and format of each data element. DBMS is used in the program so that it able to let the operating system archive data in drive accordingly. DBMS can lighten afford for the programmer chiefly making sure the end-user understands the overall design structure of datasets, by dividing the conceptual entities structure logically and physically. (Tilahun Mihret. 2019)

1.5.4 REST API

The REST architecture was lately introduced in the year 2000, Thomas Fielding, a computer scientist who proposed the principles that are based on support for the World Wide Web. In the abstract, by referring to the REST principles (Roy Thomas, 2000), REST interfaces are solely relying on Uniform Resource Identifiers (URI) for resource inquiry and interaction, and usually sent through the Hypertext Transfer Protocol (HTTP) for message transfer (Zell Liew 2018). A REST service URI only provides the link location and name of the resource, which serves as a unique resource identifier. An API is an application programming interface. It is a set of regulatory structures that allow two different programs to communicate with each other. Which comply in a decodable manner such as the format of XML or JSON(Javascript Object Notation). Generally, developers will create their API on the server and allow the client to interact with it. API as an accessing bridge between the server and clients for the client have the right on getting the resources on web services. It's also a way for servers or systems to share and control the privilege of resources and information while maintaining security, management, and authentication based on the type of users.

1.5.5 QR

A QR code is an abbreviation for quick response code, which is a two-dimension machine-readable optical label that contains alpha-numeric information, usually use for retaining manufacturing details for items inventory management. QR code is an evolved version of barcodes that was first developed in 1994 by Denso Wave Incorporated, Japan(Hwa Chang 2014). In beginning, QR came into general use as an identification tracking label for all kinds of items or products, commercial advertisements, and other publicity. Nowadays, the QR code is ubiquitous, not only used to provide links to the homepage or sharing specific content such as image content or videos. As the usage of information systems by the organization, QR code also allows consumers to access payment to merchants instantly. Surprisingly, QR is also used as a tool for recording attendance for education and jobs, even making checking the ticket digitally in public transportation like trains and flights. There are benefits of QR code not just providing enough length in adding encryption keys but also having 3 different levels of error correction which make use of the algorithm to recover the partial lost information due to physical friction of object contact.

1.6 Aim of objective

There will be five goals in this project. The first goal of this project is to create the backend for the client ware using Nodejs as an API endpoint having connection and manage the authentication via a central database using MySQL database. The second goal is to design the user interface of the client ware using HTML CSS, although the previous proposal has mentioned using Tkinter and due to the flexible technical limitation, here to clarify that UI framework been changed to HTML as easy on creating rigid view also to complete the functionality in time. The third goal is to use OpenCV to apply the QR code reading feature and also the feature for facial recognition using the LBPH algorithm. The fourth goal is making the staff portal so that staff can log in to gather new tasks for transferring service and so on. The backend can generate the location details and the new QR code for letting the smart cabinet client ware be able to scan and unlock. And the last goal is to apply liveness detection using CNN for facial recognition to enhance the security of preventing fake photo attacks.

This report will explain the methods used to develop the system and the challenges when developing the system. There will also be main topics related to this project along with the ground of existing works that cover the project. Next, the report will also be discussing the elements that comprise the project. Finally, this report will contain a summary of evaluating the project process, conclusions, and recommendations.

CHAPTER 2
ANALYSIS/LITERATURE REVIEW

2.1 LITERATURE REVIEW

2.1.1 Node.js for application development.

JavaScript introduced its concepts in AJAX in the late 1990s providing real-time responsiveness behavior to the web pages (Sebastian Peyrott 2017). And generally become a reason why JavaScript was always believed to be a client-side scripting language and that irrelevant server-side programming. In the past, JavaScript is only a client-side scripting programming language, however, after being rebuilt by Ryan Dahl for several years he published Nodejs and making JavaScript more capable of executing on the server-side (Paul Kril, henceforth the JavaScript server-side era had just begun. Nodejs is based on a V8 compiler developed and open source by Google which uses JavaScript source code to native machine code at runtime. Again it is significant to notice that Node.js is distinct from JavaScript, since JavaScript is certainly the main core of Nodejs, it is just built on top of JavaScript just that both are using the same language.

In Node.js developers are not required to manually establish threading while creating scalable servers. Node.js uses an event-driven model which uses the triggering of callback functions upon carrying out a task or generation of error. (Joyent, 2016) Leading the server having capability utilized the computing power of concurrent

processing in rapidly responding to multiple clients simultaneously without hassle in blocking the other process during the function keep getting called. The V8 engine will take responsibility for translating JavaScript into native machine language, instead of working overtime to interpret it as bytecode, giving Node.js its speed. The next benefits of using Node.js would be ideal for simple HTML or CRUD applications in which developers don't need to separate API call sections scattered on different pages of code and more manageable by Model View Control structure, and all data comes directly from the server. Making Node.js application feasible controls on scalability, it will not be a big deal even if more traffic flows to the application had run parallelly via V8 within Node.js.

Here will also introduce the Node.js package manager, it is a marketplace for open-source JavaScript tools, which plays an important role in the life cycle of this technology. There are more than 836,000 libraries available in the npm registry, and estimating 10,000 new npm published by the community every week (Seldo. 2018), shows the value of the Node.js ecosystem is intensively rich. Those packages mostly have provided tools in supporting specific tasks without need developers taking too much time reconstructing general-purpose programs technically, for example, Passport.js specifically handles the task of a session from the client. There will be tons of package support in different tasks such as specifically package for handling filesystem, image processing, socket, and so forth. Node.js able to implement the call which is very convenient for just requiring direct installation using npm command and calling out the feature within the code easily and usage is according to the need of the project.

In terms of saving the bandwidth and focus on the performance between the server and client, Node.js is an optimal choice for "single-page application" sites, where all the rendering process only requires happens on the client' side, and the backend only provides each response with the format of JSON via defined API (Reynolds. 2019). Node.js is a perfect fit for embedded devices as well as servers, the industry expertise has already adopted Node.js for development for Internet Of Things technologies. Microsoft has also adopted Node.js for IoT development in its developer resources and published its new platform Microsoft Windows IoT core (Paul Krill, 2021). Industry developers also mentioned they adopted Node.js also because it is not just introducing new thinking of how to build perfect software, but also a great way to express the existing things in a novel way.

2.1.2 Facial recognition

Authentication is an important process in computer security. Human face recognition is a significant ramification study of biometric authentication and has been applied widely in many applications, including surveillance systems, virtual and augmented reality, and physical facilities access control systems and network-based security. Commonly facial recognition consists of three steps, the first two steps will be Face Detection and Feature Extraction phases, these two processes could run simultaneously, by learning interest criteria and extracting the physical characteristics of the human face normally will be using a face detector called “Haar Cascade classifier” (Abdelmgeid, 2019). The purpose of the feature extraction process is to remove the interference of the background and

improve the accuracy of image classification, before the training of the classifier it also requires cropping out the face region in the image, and using those normalized images then preparing to feed into the classifier algorithm for training. The third step of facial recognition is about applying Machine learning models to classify the faces in the images amongst the recognized people. Matching these features with the testing images can identify the person or deny access for those people who are unable to recognize them by the classifier. There are several challenging and varying parameters in face detection and identification like illumination, different poses, change expressions, low-quality input images, etc.

Classifiers in facial recognition can be split into appearance-based or model-based algorithms. Appearance-based representation is based on collecting various statistics of the pixels' values straight forward on the face image. It works regardless of 3-dimensional models which immediately produce the 2D arrays called templates derived from intensities and edge detectors of the image as it needs a step of computing output by histograms. Model-based face recognition methods help to formulate a model of the human face that attains facial variations, such as matching extracts the distance and relative position features of facial interest point (Cui Xu, 2017). It can effectively countermeasure problems of invariant sizes, object orientation, and intensity of light. The other superiorities of these solutions are the scalability of the interpretation of face images and quick response in matching. The general extraction methods are distinguished into typical methods that concise on points, edges, lines, and curves (Abdelmgeid, 2019). Otherwise, the feature template-based approach is based on the method to compare the input image with a set of templates. Which using statistical tools like Support Vector Machines (SVM), Principal Component Analysis (PCA), Linear Discriminant Analysis (LDA), depend on structural matching methods that consider geometrical primitive features.

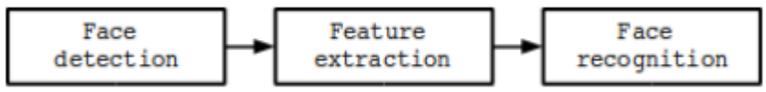


Figure 2.1 three important steps for facial recognition

2.1.3 Smart locker system

Smart locker also broadly known as an automated parcel locker, IoT intelligent box, delivery box, and universal public shareable box, has been urged as a sustainable alternative solution for domestic delivery to eliminate not at home problems and last-mile challenges (João Lopes, 2020). Sometimes, a Smart locker can be represented as an unoccupied, automated box, externally installed mounting on a wall, at the customers' residence. The smart locker can be installed in any popularly stable place such as apartment entrance, shopping mall entrance, public railway, or bus station, community spaces, and much more. The smart ockers are normally able to unlock by an electronic key or an access code (Locker & Lock®, 2021); the recipient can be notified of the delivery through a mobile gadget or email. Nowadays, The most advanced smart locker is even able to store edibles without apprehension of getting spoiled due to its

temperature control components. Moreover, by strengthening the efficiency with sophisticated data management tools and the pathfinding algorithm on “parcel locker”, the logistic solution provides better vehicle routing, reducing fuel consumption for logistics operators, and reduces drivers’ pressure of amiss addresses, and increases accuracy on schedule deliveries (Hương Trần, 2020).

According to the situation of modern logistic style in Europe, smart lockers can satisfy customers with the service because of its benefits such as making shipping costs more reasonable for consumers, leveraging the convenience and proactiveness in delivery and return services (Faugere, 2018). Concerning the environmental impact, the use of smart lockers on daily delivery can reduce the emissions of greenhouse gases (Jon Glasco, 2019) (Cathy Macharis. 2017). The greatest advantage and innovative side of a smart locker system are that it integrates customers' action into part of the system process, the intention of parcel sending or receiving. Parcel collecting is a labor-intensive and exorbitant part of warehouse management and distribution (Zhe Ding, 2013). Therefore, the system operation will require extraordinary attention during the arrangement and precise plan for dispatch operation.

2.2 Existing solution

2.2.1 Traditional public locker

When using the traditional locker, organizations will be wasting expenses every year, and putting company, customer, and employees at risk of losing assets. By distinguishing the difference between traditional versus intelligent locker systems, it helps comprehend how vital it is by accepting the change of technologies. In traditional locker management, mechanical keys are handled by employees, and records are kept to determine who has what keys. In general, the rule to handle the usage of traditional locker will include a spreadsheet or paper record, which is highly undependable and unsafe from accidentally lost rather than adopting a digital tracking system (DeBourgh, 2019). Another way around security issues may include even using a physical key or RFID, there is a high chance that the keys can get duplicated by a high tech thief, it leads to another factor that the locker owner has to be careful on handling the key. Especially when there are cases that some forgetful locker user losing even the backup of the mechanical key then there will be another cost to replace the mechanical lock or require finding management to duplicate a new key. Therefore using digital keys such as biometrics and QR codes and assigned only resource users does not require the user to physically protecting the key all over time (DeBourgh, 2019). As management for the locker will be less complicated in revoking the key after reaching the due and at the same time still being able to track all of the users who used the locker.

.

2.2.2 DHL Packstation

Deutsche Post DHL Packstation is the successful locker network in Germany, are the antecedent and most powerful parcel locker network in Europe (Paul Chapman, 2017). The Packstation service was first introduced in German in 2003 and until now established of 6.500 Packstations and more than 24.000 postal outlets or DHL parcel shops, serving millions of online shoppers (Frank Appel, 2019) The majority market for Packstation is the increasing number of individual especially dedicated pupils and engrossed professionals, who purchase products online but are not present at home often during the daytime to accept their deliveries. It benefits those who are time-poor to deposit parcels during day shift operating hours. To ensure security enough to tighten, all Packstation customers have to provide their phone numbers during registration.

The specification of Packstaion has the maximum size for parcels is 60 × 35 × 35 cm. As soon as the incoming parcel has arrived at the Packstation, the system will notify recipients by SMS and email (reichelt.com, n.d). Recipients then have seven business days to collect the package. The customer will require authentication using Gold Card(Smart Card with a magnetic strip) and mTAN(mobile transaction authentication number) for retrieving the parcel (technical-tips.com. 2021). The Packstation also accepts payment via Maestro debit cards for collect-on-delivery mail. The density of the coverage of Packstation has almost existed two German households, which around 1 km will have at least one Packstation now. (Last-Mile Prophets, 2020)

Besides that, the skyrocketing of online trade and online sales are indicating a potential opportunity for organizations investing in the automated smart locker system market. The rise of contactless secure locker solutions is dramatically elevated during the coronavirus pandemic, especially in the U.S. and Germany (CEP-Research, 2020). Similar to the solution from India technologies, the Smartbox's solution has garnered market recognition for its self-service digital locker systems that purpose secure transfer of goods and parcels during COVID-19. Companies around the globe are increasing efforts to fund developing comprehensive technological strategies to sustain economies continuously in pandemic (CEP-Research, 2020). Especially in times of the middle crisis, Packstation is a very advantageous invention in eliminating the risk of infection, such as reduce the direct contact between parcel carrier and recipient.

On the other way, the latest technologies solutions are carried out by a start-up company named Postybell42 the same company that makes the Packstations as a subsidiary of DHL has developed an IoT-enabled mailbox focused on the last mile delivery problem for domestic and apartment (Macnab. 2017). The mailbox had all attached sensors within the mailbox that transmit the occupancy state in real-time, it also attached proximity sensors and humidity sensors that detect when props are placed and also monitor the humidity inside the mailbox (Matthias Heutger. 2015). Whenever delivery happens it will alert and signal to the recipient’s phone. For example, DHL Paketkasten or Parcelbox bring on reminding customers to check their mailbox or keep track of goods no matter where recipients are (dpd.com, 2020). This is the solution to comprise the demand of e-commerce which domestic owners can request to install a personal parcel locker at their properties’ entrance, the price for normal packages is

around 99 euro and for those who just want to rent the monthly fee will start from 1.99 euros (Bonn, 2014). The system was established in Germany DHL had confirmed the future development of temperature-controlled smart lockers (Matthias Heutger. 2015). The plan for replacing traditional mailboxes and ensuring the most parcels, groceries, and other environment vulnerable products such as electronics or medicals can reach the hands of consumers safely (postandparcel.info. 2016).

2.2.3 China Cainiao logistic

In Shenzhen, 1030 vendors had signed the program of Tmall service station, which was renamed as Cainiao service station set up in 2013 after Alibaba reinforced the investment in logistics by their subsidiary Cainiao logistics affiliate for the following years (James Wang. 2017). Remarkably, Cainiao is based on committed deliberation by Alibaba to enhance final delivery service coverage. Despite a large volume of purchasers occurring on Tmall.com and Taobao.com which are mostly lying on low-and-middle-income residents in suburban districts and urban villages (James Wang. 2017). Despite the deployment of collection-and-delivery points by logistics operators in town is not as dense as in metropolitan areas. Cainiao partnered with Ant Financial, Meiyijia the populated franchise supermarket in China, on providing capital to improve end-to-end delivery scatter all over each cities corner (Apex-insight.com. 2018).

Cainiao has also established a pilot program in Shanghai that allows culinary orders to be sent over to parcel storage facilities. The Cainaio have invested in a new branch office in the city's downtown Jing'an district, it provides service including meals and groceries ordered from a stall or restaurants can be placed in proffered Cainiao smart locker (Fbicgroup.com. 2018). The operation concentrates to eliminate the inconvenience of customers receiving food and goods deliveries when overcrowding happened. The program aims to reduce contiguously and increase hygiene on speculation beneficial for both customers and delivery persons (Campbell. 2020). These lockers are substantially accustomed and serve for storing parcels purchased from e-commerce sites. Customers can have Cainiao Box installed outside their resident door to accept deliveries to mitigate stolen packages when the recipient is outdoors. Also, the box will have facial recognition features to scan users' faces or personal QR codes (Chinaplus.cri.cn. 2019), logistical may need to get permission from the customer prior so that transfer handlers are to place their orders in these lockers. Cainiao had over expanded the initiative to other cities and beyond the epidemic period, as seeking Shanghai's debut of logistics was a golden trend. By the way, when the coronavirus pandemic hit, the new regulation has restricted couriers from prohibiting entering accommodation complexes and facilities buildings, consequently causing packages stuffed around apartments’ entrances (Rita Liao. 2020). To resolve the issue, Cainiao had accompanied the state government on making strategies that comply with intelligent logistics to cover not just high density of population but also comprehensively span towards the building of relevant facilities.

Currently, Cainiao Network has collected volumetric data which exceeded 70 percent of China's sum of couriers record, it involves the connection of thousands of domestic and foreign logistics and warehouse operators and 1.7 million delivery men (Yaqing, 2017). According to the data provided by marketing analysis consultancy AskCI, it shows that about 206,000 sets of smart delivery lockers were operating in China as of 2017, and that will increase to 750,000 by 2020 (Chinaplus.cri.cn. 2019). Cainiao utilizes innovative big data and AI algorithms on optimizing delivery routes for guiding deliveries on probabilistically traffic conditions and accountable regard on more accurate order details (Post & Parcel. 2017).

CHAPTER 3
SYNTHESIS/REQUIREMENTS SPECIFICATION

3.1 REQUIREMENTS

3.1.1 FUNCTIONAL REQUIREMENT

I. Users unlock their allocated box with a QR code
The first requirement for the system is providing the QR code scanning which requires the OpenCV to read from the camera feed. The client ware will be able to check the contained message within the code before passing it to the server to match the hash value

II. Users can unlock their allocated box with their faces. (optional and only for local storage purpose)
The second requirement for the system is to provide facial recognition features and utilized OpenCV for image recognition. On identifying existing faces while having the data acquisition for LBPH training. When customers have confirmed paying for the service, that facial training will be processed behind the scene and afterward, the customer will be able to unlock their box using facial recognition.

III. Users can make payments after filling up in form.
The third feature is allowing customers to make the payment after they provided important contact details, especially email addresses. The form will have validation on the text field for making sure the customer has typed in the correct format, as the payment process will be going through with Stripe API.

IV. Users can cancel the form.
Customers can cancel the form as they wish to change plans in case.

V. Staff able to manually get new tasks if there were new transfer demands. Or cancel their task in the case.
Staff can access their portal account to acquire new tasks when new transferring tasks are created from customer requests..

VI. Staff can unlock the box from the cabinet set with a QR code.
Staff can gather the location detail along with the QR code to unlock the cabinet box whenever the task acquired is made.

VII. The cabinet client ware can communicate with the back end.
The smart cabinet client ware will manage the session and preserve the credential to make connections with the server. The keep of session token synchronizing the data when customers interact with the system interface.

3.1.2 NON-FUNCTIONAL REQUIREMENT

i. SECURITY

Increase the physical security of the lock or make the system hack-proof. The client ware is highly dependent on the connection of the back end, therefore there are concerns of the privacy base on the network design for the production when the system gets integral on hardware. The comprehensive system may have a chance of facing the adversity thread of connection being manipulated or intercepted. Besides locker is meant for allowing anyone to stay close and use, it will have a case in not knowing calamity factor causing user loss their item while using the locker due to leaked QR information and so on reasons. If the locker was unlocked, the system will not have the propulsion ability to identify the actual identity of the person who made the box unlocked.

ii. RELIABILITY

The facial detection on the project has limitations based on the light condition of the camera place on the system. If the image acquisition process has generated datasets of face mixed with noise image due to the environmental factor. Or user does not stay in the correct position in front of the camera during the scanning, it will affect the recognition accuracy which possibly rejects the correct user.

iii. USABILITY

A smart cabinet system is focusing on managing the allocation of resources of space and providing an integrated solution of logistic terms. Which does not promote additional features such as selecting the size of the locker or features such as able to control the temperature inside the locker box. Although such features are feasible to build in reality it requires an expensive cost to composite as well understand the extra domain of electronics mechanism.

3.2 ANALYSIS MODELS

3.2.1 USE CASE DIAGRAM

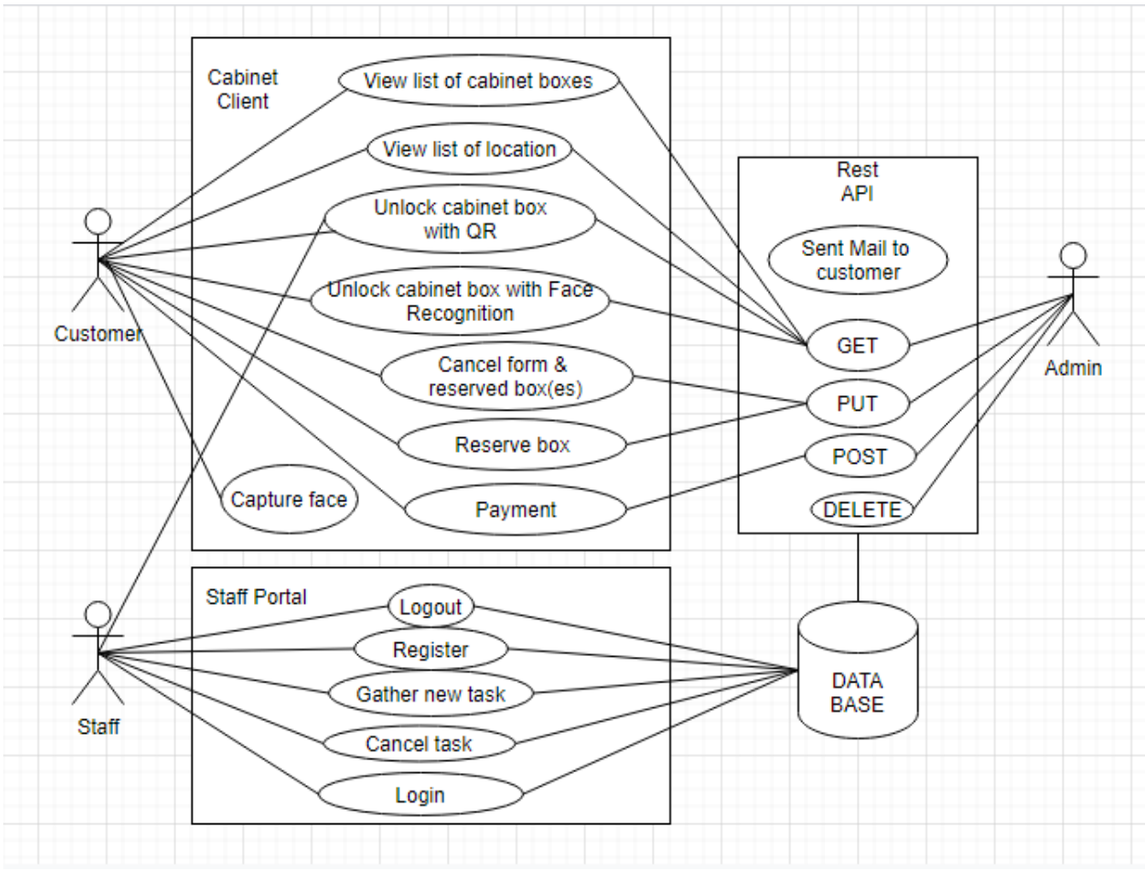


Figure 3.2 Use case diagram for smart cabinet system

According to the use case diagram on Figure 3.2, there are eight use cases for customers, five use cases for staff, and the cabinet client ware will be able to make requests to the Restful backend API by using the preset account credential to exchange tokens for HTTP requests. First will explain the use case for the customer, the use case of “view list of cabinet boxes” is for providing the list of the available box so that the customer can make a selection on the box id. The use case of “view list of location” is for providing a list of available location details for parcel transferring service so that customers can pick the destination to deliver their item. The use case of ”Reserve box” is an action that when a customer has selected any existing box id and clicked for proceeds, the client ware will request the server to issue the usage on the box id to change the state become a reserved state. The use case of “Payment” is a form for offering customers to make payment after having filled in the required information such as storing duration and email. The use case of “Cancel form & reserved boxes” is another action when a customer decides to cancel the page, the client ware will request the server to change the reserved box id becoming state empty so that the next customer will again wait for getting chosen when using the cabinet. The use case (Unlock cabinet box with face recognition) and (Unlock cabinet box with QR) is allowing the customer to use the QR or facial recognition to unlock their booked box. The use case about ”Capture face” is for customers who select to use facial recognition for unlocking will necessarily gather on the customers' facial detail in preparation for letting the recognizer be able to identify new customer faces.

Next will be all about the use case for staff, as the staff will be able to register and create a new account for their own to become a new parcel delivering staff. After registering, customers may be able to login into their account and prepare to acquire the new task. When there are new tasks, and staff has clicked to acquire anyone, staff will eventually perceive 2 QR codes which are about to unlock the source and destination locations' box.

The staff portal is built separately from the requirement of API, as the client ware and admin will rely on the backend API calls to manipulate and read data from the database. Whereas, the rest of the API only serves for admin and the smart cabinet client ware. As the rest server will manage to listen to the valid token and HTTP method before making the action on retrieving, updating, or inserting data to the database. The rest of the API also has its use case of sending email messages to customers for the details of QR code and also delivering notification messages.

3.2.2 ERD

In the Figure 3.3 ERD diagrams, all the relationships between the entities can be straightforwardly identified by the link that connected them. There will always be a primary key that exclusively identifies the records and foreign keys which are related to another entity. The figure 5.28 at the appendix is the ERD for this Smart Cabinet Malaysia system. There are a total of 6 tables on the database. In primary, table 'staff' are responsible to store the information for staff which also contain the bcrypt hash for giving the staff access to the web portal. Secondly will be table 'cabinet_set' which contains the information of the cabinet set which includes the client key with bcrypt encryption for client ware authentication. Thirdly are table 'boxes' used to store the detail which particular on the specific one locker about the reserving or allocation detail and so on, in storing the temporary QR hash key to allow customer or staff to unlock the particular box.

Fourth is the table 'box_servicing' which any new rows of records will get generated after the successful payment made by the customer, it stores details of the customer information, servicing type, the time of starting and ending of the service. Table 'box_servicing; will contains of two foregin key from boxes id which local storage service only require usage on one box and transfer location will consist relation of two different boxes id. Fifth table 'transfer_allocation' is when there is latest transfer demand in table box_servicing, the staff will be in charge of taking those tasks manually. As the table will contain the status and timestamp that staff had handled, cancelled or completed the taken task. Lastly is the independent table for admin for access in having the CRUD task for the table that previously mentioned. The purpose of each column will be explained in more detail in the following data dictionary section.

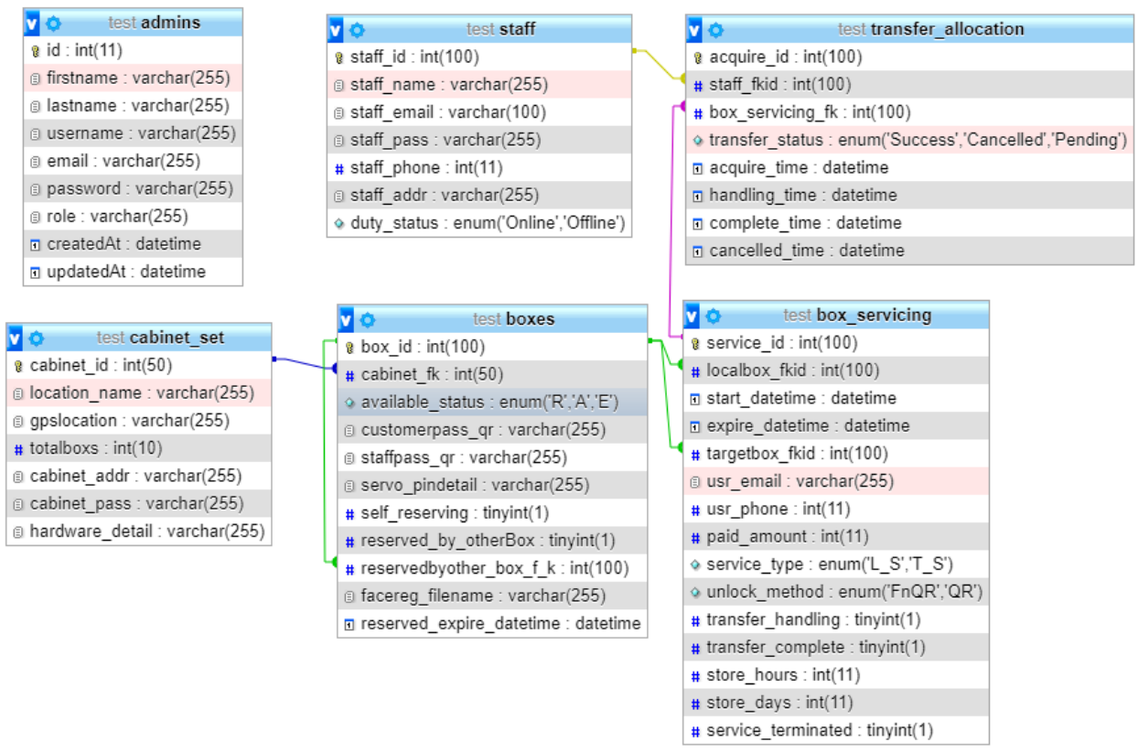


Figure 3.3 ERD diagram for smart cabinet system

3.2.3 Data Dictionary

Entity Name	Attributes	Description	Data Type	Null
boxes	box_id	Box ID.	int(100)	no
	cabinet_fk	The main host is the current box linked to the cabinet set. Foreign key ID.	int(50)	yes
	available_status	The reservation state, Reserved, Allocated, Empty	enum('R', 'A', 'E')	yes
	customerpass_qr	Random alphanumeric string	varchar(255)	yes
	staffpass_qr	Random alphanumeric string	varchar(255)	yes
	servo_pindetail	The pin detail in string	varchar(255)	yes
	self_reserving	Indicator whether the current state is in reserve by itself.	tinyint(1)	no
	reserved_by_other Box	Indicator whether the current state is in reserve by another box.	tinyint(1)	no
	reservedbyother_b	The foreign key of another	int(100)	yes

	ox_f_k	box id that reserved this box.		
	facereg_filename	The pattern of filename that the cabinet set had acquired the images of faces from customers. Use to check if the filename here is retained, if there are no matches on the client ware, the client will delete all images except the pattern string which the box is still in allocated states.	varchar(255)	yes
	Reserved_expire_datetime	The time which reserved was expired.	datetime	yes

Entity Name	Attributes	Description	Data Type	Null
box_servicing	service_id	New service as customer allocated box(s). Service ID.	int(100)	no
	localbox_fkid	Allocated box foreign key from boxes ID.	int(100)	no
	start_datetime	Service start time	datetime	no
	expire_datetime	Service expire time	datetime	no
	targetbox_fkid	Allocated box foreign key from boxes ID (destination to drop off)	int(100)	yes
	usr_email	Email of customer	varchar(255)	no
	usr_phone	Phone number of customer	int(11)	no
	paid_amount	Amount that paid (RM)	int(11)	no
	service_type	Servicing type, either choose to be used as local temporary storage or transfer to another location and storing.	enum('L_S', 'T_S')	no
	unlock_method	Method used to unlock the boxes, normally using QR, it can be Facial Recognition and QR.	enum('FnQR', 'QR')	no
	transfer_handling	The boolean state value indicates	tinyint(1)	no

		whether transfer has started to be handled by staff.		
	transfer_complete	The boolean state value indicates the transferring process has success.	tinyint(1)	no
	store_hours	The amount of hours to be stored.	int(11)	no
	store_days	The amount of day(s) to be stored.	int(11)	no
	service_terminated	State of the service ending for local ore target box.	tinyint(1)	no

Entity Name	Attributes	Description	Data Type	Null
cabinet_set	cabinet_id	Cabinet set ID	int(50)	no
	location_name	The location name of the cabinet set.	varchar(255)	yes
	gpslocation	The longitude latitude string.	varchar(255)	yes
	totalboxs	The amount of boxes that the cabinets include.	int(10)	no
	cabinet_addr	The location address of the cabinet set.	varchar(255)	yes
	cabinet_pass	The cabinet client ware password.(with bcrypt encryption)	varchar(255)	yes
	hardware_detail	String to remark about running with any electronic board.	varchar(255)	yes

Entity Name	Attributes	Description	Data Type	Null
staff	staff_id	Staff ID	int(100)	no
	staff_name	Staff full name	varchar(255)	yes
	staff_email	Staff email address	varchar(100)	yes
	staff_pass	Staff password	varchar(255)	yes
	staff_phone	Staff phone number	int(11)	yes
	staff_addr	Staff address	varchar(255)	yes

	duty_status	Staff online status (reserve for further development)	enum('Online', 'Offline')	yes

Entity Name	Attributes	Description	Data Type	Null
Transfer_ allocation	acquire_id	The transferring id that was generated, during staff gathered a new task. .Primary key	int(100)	no
	staff_fkid	The staff id who gathered the task. Foreign key from staff ID.	int(100)	no
	box_servicing_fk	The servicing detail id. Foreign key box servicing ID.	int(100)	no
	transfer_status	The status to show whether that the delivering is going to process, been successful, or cancelled.	enum('Success', 'Cancelled', 'Pending')	no
	acquire_time	The moment the staff gather this task.	datetime	no
	handling_time	The moment that staff started to pick up the parcel.	datetime	yes
	complete_time	The moment that staff delivered the parcel to the destination cabinet.	datetime	yes
	cancelled_time	The moment that staff have cancelled the current task.	datetime	yes

Entity Name	Attributes	Description	Data Type	Null
Admins	id	Admin ID.primary key	int(100)	no
	firstname	Admin firstname	varchar(255)	yes
	lastname	Admin lastname	varchar(255)	yes
	username	Admin username	varchar(255)	yes
	email	Admin email	varchar(255)	yes
	password	Admin password	varchar(255)	yes
	role	Admin role type	varchar(255)	yes

	createAt	Admin created date	datetime	no
	updateAt	Admin update date	datetime	no

3.2.4 SQL Procedure

Procedure 01 clear_expired_reserved_boxes	

	BEGIN
	DECLARE finished INTEGER DEFAULT 0;
	DECLARE BoxId INT;
	-- declare boxid cursor for finding any row when reserving reaching due
	DECLARE BoxesCursor
	CURSOR FOR
	SELECT `box_id` FROM `boxes` WHERE `available_status`='R' AND `reserved_expire_datetime` < NOW();
	-- declare NOT FOUND handler
	DECLARE CONTINUE HANDLER
	FOR NOT FOUND SET finished = 1;
	
	OPEN BoxesCursor;
	
	updateRboxDue: LOOP
	FETCH BoxesCursor INTO BoxId;
	IF finished = 1 THEN
	LEAVE updateRboxDue;
	END IF;
	
	UPDATE `boxes` SET `available_status`='E',`self_reserving`='0',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL, `reserved_expire_datetime`=NULL WHERE `box_id`=BoxId;
	END LOOP updateRboxDue;
	CLOSE BoxesCursor;
	commit;
	
	END

Explanation	Resetting those boxes id which is reserved state which without becoming “Allocated” state before the reserving expire time.

Procedure 02 update_end_local_store_serv	

	BEGIN
	
	DECLARE finished INTEGER DEFAULT 0;
	DECLARE BoxId INT;
	DECLARE ServiceId INT;
	
	-- declare boxid cursor for finding any row when local storage service reaching due
	DECLARE L_S_EndCursor
	CURSOR FOR
	SELECT `localbox_fkid`,`service_id` FROM `box_servicing` WHERE `expire_datetime` < NOW() AND `service_type`='T_S' AND `transfer_complete`='1' AND `service_terminated`='0';
	
	-- declare NOT FOUND handler
	DECLARE CONTINUE HANDLER
	FOR NOT FOUND SET finished = 1;
	
	OPEN L_S_EndCursor;
	
	updateLocalStoreBoxDue:LOOP
	FETCH L_S_EndCursor INTO BoxId, ServiceId;
	IF finished = 1 THEN
	LEAVE updateLocalStoreBoxDue;
	END IF;
	
	UPDATE `box_servicing` SET `service_terminated`='1' WHERE `service_id`=ServiceId;
	UPDATE `boxes` SET `customerpass_qr`=NULL, `staffpass_qr`=NULL,`available_status`='E',`self_reserving`='0',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL, `reserved_expire_datetime`=NULL WHERE `box_id`=BoxId;
	END LOOP updateLocalStoreBoxDue;
	CLOSE L_S_EndCursor;
	commit;
	
	END

Explanation	Set those local storage records to service terminated if the expiration was reached, after ward resetting the box id status into Empty state.

Procedure 03 update_end_transfer_store_serv	

	BEGIN
	
	DECLARE finished INTEGER DEFAULT 0;
	DECLARE BoxId INT;
	DECLARE ServiceId INT;
	
	-- declare boxid cursor for finding any row when transfer service reaching due
	DECLARE L_S_EndCursor
	CURSOR FOR
	SELECT `localbox_fkid`,`service_id` FROM `box_servicing` WHERE `expire_datetime` < NOW() AND `service_type`='T_S' AND `transfer_complete`='1' AND `service_terminated`='0';
	
	-- declare NOT FOUND handler
	DECLARE CONTINUE HANDLER
	FOR NOT FOUND SET finished = 1;
	
	OPEN L_S_EndCursor;
	
	updateLocalStoreBoxDue:LOOP
	FETCH L_S_EndCursor INTO BoxId, ServiceId;
	IF finished = 1 THEN
	LEAVE updateLocalStoreBoxDue;
	END IF;
	
	UPDATE `box_servicing` SET `service_terminated`='1' WHERE `service_id`=ServiceId;
	UPDATE `boxes` SET `customerpass_qr`=NULL, `staffpass_qr`=NULL,`available_status`='E',`self_reserving`='0',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL, `reserved_expire_datetime`=NULL WHERE `box_id`=BoxId;
	END LOOP updateLocalStoreBoxDue;
	CLOSE L_S_EndCursor;
	commit;
	
	END

Explanation	Set all transfer service (T_S) types of service in box servicing table which transfer_complete state equal to 1 and been reached due to change the service terminated state to 1. Again resetting those box id.

3.3 DESIGN SPECIFICATIONS

At first, the current statement will be explaining the interface design of the Smart Cabinet Malaysia system client ware. So far the overall design must be as simple as possible so that the user would not spend too much time learning how to use the system. So far, the design of smart cabinet clients is simple and consistent so that when a user uses the system, customers do not take time to relearn how to use the system. After some studies on the interface design based on several websites of commercial SPA sites, an upshot has been created. The template of the content view is rendered by EJS express engine, the layout of the interface would be divided into four different sections.

Based on Appendix Figure 1, the first part starts from the main navigation panel, four clickable links allow the user to navigate to different features of the system. Next will be Appendix Figure 2.1, the second part of the interface is for letting the customer select the local box(id) from the cabinet set from the machine user currently interacting with, those numbers inside the square in color green within the grid in the page are currently available to choose and reserve. After selecting and clicking the proceed button that will be redirected to a page of form based on Appendix Figure 2.2, it contains a text field requiring customers to provide their email and phone details, also there are date picker and an hours drop-down list which use to select the duration that customer wishes to store their item in the locker. The price label is placed beside the drop-down list, whenever the drop-down list and date picker shows in Appendix Figure 2.3 was changed the price inside the label will change seamlessly together with the estimated due label purpose to show the time that the estimated expiration date for the selected box.

Continued still about Appendix Figure 2.2, below the estimated due label it will have two checkboxes, one is selected by default which is for unlocking method using QR and the other check box is for letting customers decide whether they want to use facial recognition to unlock afterward. If a customer favorable to use the facial recognition and clicked the checkbox, it will pop a dialog message which requires the user's consent before the capturing shows on Appendix Figure 2.4 here user still can decide to start capturing or cancel. In case customers choose to start capture, a few seconds later there will be an OpenCV frame showing up and start taking the image of the user’s face. Hereafter the face scanned, the form page will load and the facial recognition checkbox will get selected shows in Appendix Figure 2.6. Lastly, if the required information is provided to the form next will proceed the payment using Stripe API shows in Appendix Figure 3.6.

The third part would select as transfer storage referring to Appendix Figure 3.1, which was the interface when a customer will firstly choose to reserve a local box and then proceed to select the target location referring to Appendix Figure 3.2 which is another location that has covered by the service of the cabinet means another machine or another set of the smart cabinet. After selecting the target location and clicking proceed, the customer will see another grid of boxes in numbers referring to Appendix Figure 3.3, those are the boxes and their unique id for the target cabinet which is about letting the

customer choose any of the target box ids that will be going to transfer to. And lastly will be the form for setting the storing duration and making payment. The form design and procedure are similar to the local storage just the difference was transfer storage form has an additional label to show the name of the destination referring to Appendix Figure 3.4, as well as an additional label for showing target box id which is currently reserved. Thereby, another difference is the form for transfer storage doesn't have the checkbox for facial recognition forasmuch only able to unlock with QR code.

About the unlocking part, it will refer to Appendix Figure 4.1 users will see a list of the grid of green squares with numbers after clicking the unlock button from the home page on Appendix Figure 1. Customers can select their allocated box id then proceed to unlock using either QR code or facial recognition shows in Appendix Figure 4.2, in this page customer will see a label of box id on the above two images the label is the id that was previously selected by customers. The first image is for unlocking using a QR code, another is for facial recognition. Once the customer clicks the icon image with QR code with a hand, a few seconds later the OpenCV window will appear and try to read any existing code in front of the camera referring to Appendix Figure 4.3. In another case, once a customer clicked to choose to unlock using facial recognition to unlock, a few seconds later an OpenCV window eventually showed up based on Appendix Figure 4.4 within the camera feed it comes with a message on the above to ask for the customer to blink their eyes slowly. After minutes if the OpenCV does not detect a relevant face or QR code, the page referring to Appendix 5.1 will pop a message showing "no face or QR are detected". The other condition when a user has success scanned with the correct face or QR code, the page will redirect to show the box id is unlocked referring to Appendix Figure 5. Otherwise, when the QR or face is scanned but it was incorrect, it will show the box id unlocked failed in page referring to Appendix Figure 5.2.

The final part of client ware would be the F.A.Q page in Appendix E_b Figure 2.10 consists of the detail and tutorial about using the smart cabinet and information about the pricing and so on for whom the system owner who developed it. About the unlocked page it will only show after payment success or failure or during the unlocking using QR code and face.

Next will be a discussion about the design of the staff portal, the staff code itself is independent and running on port 5000. Based on Appendix 7.1 it shows the first page has two forms and one of the forms at the left is for letting the staff log in, and the form at the right is used for letting new users register to become a new staff. Nothing special for the design but direct and candid, as after user login will be able to see the view either on Appendix 7.2 and 7.3. Here is the home page when the staff view, if they have done the task or currently not taking any task yet, there will be no additional details for the box id and the location address showing. If the staff had gathered a new task, the page would show like Appendix 7.3 to let the staff know the new location that needs to pick up the parcel by unlocking the relevant cabinet box id by using the provided QR code and then scan with the remote Smart Cabinet Malaysia client ware which installed on machine according to the "A" and "B" address and box id that given. The left QR code is the source

location’s cabinet box and the right will be the target destination. The box id will be shown below the key emoji on the page. And there are no extra controls on the page which staff are only a few buttons Able to gather new tasks, cancel new tasks, and log out. Based upon Appendix Figure 7.4, in condition if the staff is not able to gather a new task, they will eventually see an alert dialog and the HTML label under the text of “Total available task:” will be changed to ‘0’ due to no new customer being requesting for the transfer service or it met a condition which is plausible that all task been all taken by other staff during the situation.

.

3.4 PRODUCT CODE

F.O.C = Function or Code

3.4.1 Backend API (F.O.C 1.1 - 1.33)

Backend Index.js (F.O.C 1.1)

The index.js in the backend contains the server setting such as the express engine, bodyParser for JSON request handling, activating of cross origin resource sharing, importing passport session handling package, default token encryption key, port for server to listen, the path accessible for the public. The index.js also consist of importing the endpoint handling code or client.js for the process of client request and admin.js for admin request from routes folder. On line 11-18, the section of code is used for checking the JSON format, if the format is not expected form it will return a value 400 of bad request unless the format is correct it will allow request forward to routes.

F.O.C 1.1	**Index.js (Code)**

1 - 5	import express from 'express';
	//package for web application feature
	import passport from 'passport';//required middleware provided by Node.js for authentication
	import bodyParser from 'body-parser';//Parse incoming request bodies in a middleware
	import models from './models'
	//require the sequelize model
	import cors from 'cors';
	//allows restricted resources on a web page to
	//be requested from another domain outside the domain

8 - 9	const app = express();//initialize express call app.use(require("body-parser").json());//handle the JSON request

11 - 18	app.use(function(err, req, res, next) {
	// ⚙ function to catch errors from body-parser
	if (err instanceof SyntaxError && err.status === 400 && "body" in err)
	{ res.status(400).send({ code: 400, message: "bad request" });

	//return bad request message if the body is invalid
	} else next();
	//here code is for checking any request that not in required format });

20 - 21	let port = process.env.PORT || 4000; //initialize default port let secret_ = process.env.SECRET || "SECERT"; //initialize secret key

24 - 35	// set the view engine to ejs
	app.set('view engine', 'ejs');
	
	// make express look in the public directory for assets (css/js/img)
	app.use(express.static(__dirname + '/public'));
	
	//cross origin resource sharing
	app.use(cors());
	
	// force: true will drop the table if it already exits // models.sequelize.sync({ force: true }).then(() => { models.sequelize.sync().then(() => { console.log('Drop and Resync with {force: true}'); });

37 - 42	// passport middleware app.use(passport.initialize()); // passport config require('./config/passport')(passport);//passport auth setting for admin require('./config/clientpassport')(passport);//passport auth setting for client

44 - 49	//default route app.get('/', (req, res) => res.send('Hello my World')); //initialize the request handling end point require('./routes/admin.js')(app);
	require('./routes/client.js')(app);

52 - 57	//create a server var server = app.listen(port, function() { var host = server.address().address;//listening uri var port = server.address().port;//listening port console.log('App listening at http://%s:%s', host, port); });

Backend Config/clientpassport.js (F.O.C 1.2)

The clientpassport.js take responsible on decoding the JWT key from client request. By checking the client id after JWT had decrypted with the secret key. If the database consists of CABINET_SET id that decode from JWT on line 10 - 20, the return value will become true.

F.O.C 1.2 config/clientpassport.js	
1 - 2	import { Strategy as JwtStrategy, ExtractJwt } from'passport-jwt' //require passport library to extract JWT token import models from '../models'// require model
4 - 8	const Client = models.cabinet_set; // require the model of client

	let secret_ = process.env.SECRET || "SECERT";//default key
	const opts = {};//create new option object
	opts.jwtFromRequest=ExtractJwt.fromAuthHeaderAsBearerToken();
	//read bearer
	opts.secretOrKey = secret_;
	//encryption key to decode the message within the key

10 - 20	// create jwt strategy
	module.exports = passport => { //export the passport call
	passport.use("strategyClient",
	//verify these credentials with the database
	new JwtStrategy(opts, (jwt_payload, done) => {
	//decode the message from JWT
	Client.findAll({ where: { cabinet_id: jwt_payload.cabinet_id } })
	//find the extracted ID from JWT in database
	.then(client => { if (client.length) { //if the client exist
	return done(null, client); // return true
	console.log("client valid "+client) }
	return done(null, false); //else return false
	}) .catch(err => console.log(err)); //catch error
	})
	);
	};

Backend routes/client.js (F.O.C 1.3 - 1.33)

F.O.C 1.3	**routes/client.js(code)**
1.	import passport from 'passport'; // authentication middleware for Node.js
2 - 8	import { login, findAllClient, niHao, unlockforcustomerwithface, unlockforcustomer, unlockbeforetransfer, unlockaftertransfer,testMail, box_itself_reserve, box_reserve_another_box, listofLocations,list_ALL_localCabinetboxess,list_of_faceFile_in_allocatedBox, list_of_boxess, list_of_targetCabinetboxess, list_of_localCabinetboxess, confirm_payment, cancel_reserved_box } from '../controllers/client'; //import all function from controllers/client.js
10	module.exports = (app) => { //export all the callback function here to index.js
13	app.post('/api/client/login', login); //End point to login not require JWT at first

15 - 125
#Contained duplicated
	app.get('/api/client/nihao' //setting for api endpoint
	, passport.authenticate("strategyClient", //validation the JWT with passport
	{ session: false }), niHao //the name of the function going to call);

syntaxes	// app.get= request data but not modifying,
	//app.put= update data,
	//app.post = create data

Because there is repeating code when using the database execute call within client.js , here will just explain all at once together. And the rest of the function only will focus on the important statement after the query and also the purpose of the query command.

F.O.C 1.4	**Common syntaxes in backend client.js**

Database command execute	dbConnection.execute(queryX
	,[placeholderX])
	.then(([rows]) => {
	console.log(" row affected> "+rows.affectedRows);
	})

Catch query error	.catch(err => {
	if (err) throw err;
	if (err){
	console.log('error Handling value set complete value to servicingID> '+ servicingID);
	}
	});

Defining variable for holding the requesting body value with Ternary operator with default null string	var xxx_ = (req.body.xxx !== undefined) ? req.body.xxx : "null";
Defining variable for holding the requesting body value	var abc = req.body.abC;
Response body JSON format	res.status(200).json({ box_itself_reserve:'Yes', msg:'Reserved', boxid: boxID });
Get the requester id	var currentCabinetID = getTokenID(req.headers.authorization);

F.O.C 1.5	**client.js (Code)**
25 - 33	let transporter = nodemailer.createTransport({ host: 'smtp.gmail.com', port: 465,

	secure: true, // true for 465, false for other ports
	auth: {
	user: backendemail,
	pass: backendmailpass,
	},
	});
	//nodemailer email authentications setting

904 - 912	export {
	login,
	findAllClient,
	niHao,
	unlockforcustomer,
	unlockforcustomerwithface,
	unlockbeforetransfer,
	unlockaftertransfer,
	box_itself_reserve,
	box_reserve_another_box,
	cancel_reserved_box,
	list_ALL_localCabinetboxess,
	list_of_localCabinetboxess,
	list_of_targetCabinetboxess,
	listofLocations,
	list_of_faceFile_in_allocatedBox,
	confirm_payment
	}
	//Export all functions make it callable

F.O.C 1.6 const login = (req, res) => {	

36 - 42	const { errors, isValid } = validateLoginClientForm(req.body);
	
	// check validation
	if(!isValid) {
	return res.status(400).json(errors);
	}
	//Calling form validation function validateLoginClientForm() if return not met requirement it will return 400 Bad Request.

44 - 52	const { location_name, password } = req.body;
	console.log(location_name);
	console.log("Client_ "+Client_);
	Client_.findAll({
	where: {
	location_name :req.body.location_name

	}
	})
	.then(client => {
	//Execute sequelize query to find the cabinet client id using location name
	//client => is the result return from the callback

55 - 58	if (!client.length) {
	errors.location_name = 'client not found!';
	return res.status(404).json(errors);
	}
	// if there is at least a row of result from the query.

60	let originalPassword = client[0].dataValues.cabinet_pass //fetch result pass to originalPassword variable

63 - 66	bcrypt
	.compare(password, originalPassword)
	.then(isMatch => {
	if (isMatch) {
	//using bcrypt encryption function to compare the result hash with the requested input.

69 - 80	const { cabinet_id, location_name } = client[0].dataValues;
	const payload = { cabinet_id, location_name }; //jwt payload
	// console.log(payload)
	
	jwt.sign(payload, secret_, {
	expiresIn: 86400 //24 hours
	}, (err, token) => {
	res.json({
	success: true,
	token: 'Bearer ' + token,
	address: client[0].dataValues.cabinet_addr
	});
	// Returning signature of JWT token if the password matched, the expire time is around 24 hours. Using jwt.sign to create the token which including information of the ‘payload’ and encerypt with the ‘secret_’ so that the server will use to verify this key which only the server is able to decode.
	//response back to client using JSON format res.json.

F.O.C 1.7 const confirm_payment = (req, res) => {	
100 - 160	//skipped contain repeated syntaxes
123 - 129	if (serviceType == "L_S"){ //checkService Type Local_Storing or Transfer_Storing

	placehoders = [localBid,currentSessionID];
	query = "SELECT `box_id` FROM `boxes` WHERE `box_id`=? AND `cabinet_fk` = ? AND `self_reserving` = 1";
	}else if (serviceType == "T_S"){
	placehoders = [targetBid,localBid,currentSessionID];
	query = "SELECT B.`box_id` as targetboxid, A.`box_id` as localbox FROM `boxes` as B INNER JOIN `boxes` AS A ON A.`box_id` = B.`reservedbyother_box_f_k` WHERE B.`box_id` =? AND B.`reservedbyother_box_f_k` = ? AND A.`cabinet_fk`=? AND A.`self_reserving`=1";
	}
	// Changing the query for retrieving the cabinet set info ,and box information. As since Local servicing only involves one box information, as Transfer servicing involves two different boxes hosted by different cabinet sets thus need to be joined. The local box should be in state of self reserved, and if condition T_S service the target box must be reserved by the local box with check of B.`reservedbyother_box_f_k = x.

131 - 136	dbConnection.execute(query
	,placehoders)
	.then(([rows]) => {
	console.log("Running query> "+query)
	if (rows.length > 0){
	console.log("Found ");
	// Execute the prepared query with a searching row that matches the placeholder. If the row is greater than 0 means found. Means found box(es) been reserved previously

139 - 140	insertNewService(localBid,targetBid,customerEmail,customerPhone, payAmount,serviceType,faceScannQr,storingDay,storingHours, res); //Call the insert new box servicing to insert the information.
142 -144	if (serviceType == "L_S"){ updateBoxAvailableStatus1(localBid,currentSessionID, face_recFilename);} //If service type is L_S call updateBoxAvailableStatus1()

144 -147	else if (serviceType == "T_S"){
	updateBoxAvailableStatus1(localBid,currentSessionID,"");
	updateBoxAvailableStatus2(targetBid,localBid,currentSessionID);}
	// Else if serviceType equal Transfer_Status call updateBoxAvailableStatus1() and updateBoxAvailableStatus2()

F.O.C 1.8	**function insertNewService(vlocalbox_fkid,vtargetbox_fkid, vusr_email, vusr_phone, vpaid_amount, vservice_type,**

	vunlock_method,vstore_days, vstore_hours, res){
	//skipped contain repeated syntaxes

178 - 183	if (vtargetbox_fkid ==""){
	placehoders =[vlocalbox_fkid, numday, numhrs,vusr_email,
	vusr_phone, vpaid_amount, vservice_type,
	vunlock_method,vstore_hours, vstore_days];
	query = "INSERT INTO `box_servicing`(`localbox_fkid`, `start_datetime`, `expire_datetime`,`targetbox_fkid`, `usr_email`, `usr_phone`, `paid_amount`, `service_type`, `unlock_method`, `store_hours`, `store_days`) VALUES (?,now(),NOW()+INTERVAL ? DAY +INTERVAL ? HOUR ,NULL,?,?,?,?,?,?,?)";
	}
	// query to insert new row to box_servicing table (for local storing)

183 - 188	}else{
	placehoders =[vlocalbox_fkid, numday, numhrs,vvtboxid,
	vusr_email, vusr_phone, vpaid_amount,
	vservice_type,
	vunlock_method, vstore_hours, vstore_days];
	query = "INSERT INTO `box_servicing`(`localbox_fkid`, `start_datetime`, `expire_datetime`,`targetbox_fkid`, `usr_email`, `usr_phone`, `paid_amount`, `service_type`, `unlock_method`, `store_hours`, `store_days`) VALUES (?,now(),NOW()+INTERVAL ? DAY +INTERVAL ? HOUR ,?,?,?,?,?,?,?,?)";
	}
	// query to insert new row to box_servicing table(for transfer storing)

194 - 197	if (rows.insertId != 0){
	console.log("New box_servicing row created >ID: "+rows.insertId);
	queryLatestInsertedServicingRow(rows.insertId,vservice_type, vunlock_method, res);
	}
	//Check if the information is successfully inserted, if it does call queryLatestInsertedServicingRow() function.

F.O.C 1.9	**function updateBoxAvailableStatus1(localBid,currentSessionID, face_recFilename){**
211 - 213	dbConnection.execute("UPDATE `boxes` SET `reserved_expire_datetime`=NULL, `available_status`='A',`customerpass_qr`=?, `facereg_filename`=? WHERE `box_id`=? AND `cabinet_fk`=? AND `available_status`='R'

	AND `self_reserving`= 1" ,[randomHash,face_recFilename, localBid, currentSessionID]) //for localbox to update customerQr to new random hash and changing reserving status(R)reserved status becoming (A)allocated

F.O.C 1.10	**function updateBoxAvailableStatus2 (targetBid,localBid,currentSessionID){**

226 - 228	dbConnection.execute("UPDATE `boxes` SET `reserved_expire_datetime`=NULL, `available_status`='A',`customerpass_qr`=? WHERE `box_id`=? AND `reservedbyother_box_f_k`=? AND `available_status`='R' AND `reserved_by_otherBox`=1 AND `cabinet_fk`!=?"
	,[randomHash,targetBid,localBid,currentSessionID])
	.then(([rows]) => {
	// Query for update the reserved target box to allocated status

F.O.C 1.11	**async function mailing(boxid1,startTime,expireTime, locationAddr1, qrbox1, boxid2,locationAddr2, qrbox2, svType, serviceID,customerEmail) {**

242-244	if (typeof customerEmail !=='undefined'){
	targetReceiver = customerEmail;
	}
	//IF the customerEmail provided is undefined set targetReceiver to default value.

245 - 254	var qR1 ={
	sid: serviceID,
	cqr: qrbox1
	};// QR object 1 contains service ID and QR code string
	var qR2 ={
	sid: serviceID,
	cqr: qrbox2 //customer QR
	};// QR object 2 contains service ID and QR code string
	let img = await QRCode.toDataURL(JSON.stringify(qR1));
	let img2 = await QRCode.toDataURL(JSON.stringify(qR2));
	//(Convert Json into Qr image)

258 - 268	if (svType =="L_S"){
	htmlstruc =”HTML1”;// HTML content shortcutted
	}else if(svType =="T_S"){
	htmlstruc = “HTML2”;// HTML content shortcutted
	}
	// IF service types are either L_S or T_S prepare the HTML information, both provided information is different.

270 - 277	let info = await transporter.sendMail({
	attachDataUrls: true,//to accept base64 content in messsage
	from: '"□" <smartcabinetmalaysia.com>', // sender address
	to: targetReceiver, // list of receivers
	subject: "Hello ✓ From Smart Cabinet Malaysia", // Subject line
	text: "Smart Cabinet QR access code please do not share to unknown parties", // plain text body
	html: htmlstruc // html body
	});
	//Using nodemailer to send the information to the 'targetReceiver' who made a new request to use the smart cabinet.

F.O.C 1.12	**Function queryLatestInsertedServicingRow (servicingID,serviceType,vunlock_method, res)**

286 - 287	var query; //init variable to save query
	console.log("Running queryLatestInsertedServicingRow"); await new Promise(resolve => setTimeout(resolve, 5000));
	//wait for the time for the update of box id approximate 5 second

289 - 296	if (serviceType =="L_S"){
	query = "SELECT A.`localbox_fkid`, A.`usr_email`, A.`paid_amount`,,A.`start_datetime`, A.`expire_datetime`, C.`customerpass_qr`, C.`facereg_filename`, C.`servo_pindetail` ,F.`cabinet_addr` FROM `box_servicing` AS A INNER JOIN `boxes` AS C ON C.`box_id` = A.`localbox_fkid` INNER JOIN `cabinet_set` AS F ON F.`cabinet_id` = C.`cabinet_fk` WHERE A.`service_id`=?";
	if (vunlock_method="FnQR"){
	requireTrain ="yes";
	}
	}else if(serviceType =="T_S"){
	query = "SELECT A.`localbox_fkid`,A.`usr_email`, A.`paid_amount`, A.`targetbox_fkid` ,A.`start_datetime`, A.`expire_datetime`, C.`customerpass_qr`, C.`facereg_filename`, C.`servo_pindetail`, D.`customerpass_qr` AS `customerpass_qr2` ,F.`cabinet_addr`, G.`cabinet_addr` AS `cabinet_addr2` FROM `box_servicing` AS A INNER JOIN `boxes` AS C ON C.`box_id` = A.`localbox_fkid` INNER JOIN `boxes` AS D ON D.`box_id` = A.`targetbox_fkid` INNER JOIN `cabinet_set` AS F ON F.`cabinet_id` = C.`cabinet_fk` INNER JOIN `cabinet_set` AS G ON G.`cabinet_id` = D.`cabinet_fk` WHERE A.`service_id`=?";
	}
	//Query for the latest inserted box_servicing row and the linked box(es) and the cabinet information
	//T_S involve two boxes and L_S only involve one while querying its cabinet set information it require on the join

288	var requireTrain ="no"; // the clientware will check this return value after, in the response whether the face recognition require to train
310 - 312	mailing() // call mailing function

313 - 319	res.status(200).json({
	validation:'yes',
	gpiopin: rows[0].servo_pindetail,
	boxid : rows[0].localbox_fkid,
	trainFace : requireTrain,
	frecogfilename : rows[0].facereg_filename
	});
	// response value after payment success return back to the client whom sent request previously

F.O.C 1.13	**const unlockforcustomer = (req, res) => {**
331 - 360	//skipped contain repeated syntaxes

339 343	var querycustomerQR = "select A.`service_id`, A.`expire_datetime`, B.`servo_pindetail`, B.`box_id` from box_servicing A inner join boxes B on A.`localbox_fkid` = B.`box_id` or A.`targetbox_fkid` = B.`box_id` WHERE A.`service_id` = ? AND A.`expire_datetime` >= NOW() AND B.`box_id` =? AND B.`customerpass_qr`=? limit 1";
	dbConnection.execute(querycustomerQR
	,[serviceId, boxId, customerQr])
	.then(([rows]) => {
	if (rows !== 'undefined' && customerQr !==""){
	
	// if the query result consist a row and the customerQr value is not empty
	Finding the record matching serviceId, boxId, customerQr.

348 - 353	res.status(200).json({
	validation:'yes',
	expiration: rows[0].expire_datetime,
	gpiopin: rows[0].servo_pindetail,
	boxid : rows[0].box_id
	});
	// response the client with JSON

F.O.C 1.14	**const unlockforcustomerwithface = (req, res) => {**
	//skipped contain repeated syntaxes

372 - 374	var querycustomerFace = "select A.`service_id`, A.`expire_datetime`, B.`servo_pindetail`, B.`box_id`,B.`facereg_filename` from box_servicing A inner join boxes B on A.`localbox_fkid` = B.`box_id` or A.`targetbox_fkid` = B.`box_id` WHERE A.`expire_datetime` >= NOW() AND B.`box_id` =? AND B.`facereg_filename`LIKE ? ORDER BY A.`service_id` DESC limit 1"; dbConnection.execute(querycustomerFace ,[boxId ,labelstrrmovelinebreak]) //Check the localbox id consisting of the label that was provided on request. Also whether the box_servicing id is still not expire

F.O.C 1.15 const unlockbeforetransfer = (req, res) => {	
397 - 436	//skipped contain repeated syntaxes

408 - 410	var querybeforetransferQR = "select A.`service_id`, A.`expire_datetime`, B.`servo_pindetail`, B.`box_id`, T.`transfer_status`, T.`acquire_id` from box_servicing A inner join boxes B on A.`localbox_fkid` = B.`box_id` inner join transfer_allocation T on A.`service_id` = T.`box_servicing_fk` WHERE T.`acquire_id` = ? AND A.`expire_datetime` >= NOW() AND B.`box_id` =? AND B.`staffpass_qr`=? AND T.`complete_time` IS NULL AND A.`transfer_complete` ='0' limit 1";
	dbConnection.execute(querybeforetransferQR
	,[tid, bid, bqr])
	//Search for whether the local box is having the match of staff QR pass.
	//Involve relationship with tables on box_servicing, transfer_allocation, boxes

F.O.C 1.16 const unlockaftertransfer = (req, res) => {	
439 - 502	//skipped contain repeated syntaxes
450 - 452	var queryaftertransferQR = "select A.`service_id`,TIMESTAMPDIFF(HOUR,now(),A.`expire_datetime`) as difference_in_hours, A.`localbox_fkid` , A.`expire_datetime`, A.`usr_email`, B1.`box_id` as bid1, B2.`servo_pindetail`, B2.`box_id` as bid2, T.`transfer_status`, T.`acquire_id`, C1.`location_name` as locationA, C1.`cabinet_addr` as boxAaddr, C2.`location_name` as locationB, C2.`cabinet_addr` as boxBaddr from box_servicing A inner join boxes B1 on A.`localbox_fkid` = B1.`box_id` inner join boxes B2 on A.`targetbox_fkid` = B2.`box_id` inner join transfer_allocation T on A.`service_id` = T.`box_servicing_fk` inner join cabinet_set C1 on B1.`cabinet_fk` = C1.`cabinet_id` inner join cabinet_set C2 on B2.`cabinet_fk` = C2.`cabinet_id` WHERE T.`acquire_id` = ? AND B2.`box_id` =? AND B2.`staffpass_qr`=? AND T.`handling_time` IS NOT NULL AND T.`complete_time` IS NULL AND A.`transfer_complete` ='0' limit 1"; dbConnection.execute(queryaftertransferQR

	,[tid, bid, aqr]) //Search for whether the local box (source destination) has been scanned with the first QR code for staff delivery which is assumed the parcel has been picked up by staff by handling the task, also check if the expire time of the servicing id is less than an hour.
454 - 455	var t_status = (typeof rows[0].transfer_status !== 'undefined') ? rows[0].transfer_status : "no"; //Define value if query object transfer_status is undefined set to default value of “no”
456	if (rows.length > 0 && aqr !=="" && t_status =="Pending"){ // check if a row exists and the aqr = after QR (target box staff qr string) is not empty and also transfer status in transfer_allocation for the id is still Pending state.
459	var lateYesNo = "no"; // value for determining the transfer is late or not.

462 - 466	if(rows[0].difference_in_hours > '1'){
	console.log("Not late :"+rows[0].difference_in_hours);
	//Call to update the box_servicing Transfer status to Success
	setTransferCompleteOnBoxServ(rows[0].service_id, lateYesNo);
	}
	//if still greater than an hour means not late
	// call the setTransferCompleteOnBoxServ function

466 - 471	}else{
	lateYesNo = "Yes";
	console.log("Been late :"+late);
	//Call to update the box_servicing
	// set transfer status to Success
	setTransferCompleteOnBoxServ(rows[0].service_id, lateYesNo);
	}
	//The transferring success time and the expire time of the box_servicing had lesser than an hour
	//Call setTransferCompleteOnBoxServ function.

473	setTransferCompleteOn_TAllocation(tid); //skipped contain repeated syntaxes
477 - 480	NotifyEmailforParcelReached(rows[0].serviceID, rows[0].usr_email, rows[0].expire_datetime,rows[0].locationA, rows[0].bid1, rows[0].boxAaddr,rows[0].locationB, rows[0].bid2, rows[0].boxBaddr,lateYesNo).catch(console.error); //Calling the NotifyEmailforParcelReached( ) function
482 - 483	setEmptytoLocalBox(rows[0].localbox_fkid);

	settargetboxtoSelfReserve(rows[0].bid2);
	//calling funtiosetEmptytoLocalBoxn
	//calling settargetboxtoSelfReserve() function

F.O.C 1.17 const box_itself_reserve = (req, res) => {	
	//skipped contain repeated syntaxes
515 - 516	dbConnection.execute("UPDATE `boxes` SET `available_status`='R',`self_reserving`='1', `reserved_expire_datetime`=now() +INTERVAL 8 MINUTE WHERE `box_id`=? AND `cabinet_fk`= ? AND `available_status`='E'" ,[boxID, currentSessionID]) // set the local box to self reserving and available status to R with 8 minutes of expiry.

F.O.C 1.18 const box_reserve_another_box = async (req, res) => {	
541 - 575	//skipped contain repeated syntaxes
547	queryReserving = "UPDATE `boxes` SET `available_status`='R',`reserved_by_otherBox`='1',`reservedbyother_box_f_k`= ?,`reserved_expire_datetime`=now() +INTERVAL 8 MINUTE WHERE `box_id`=? AND `cabinet_fk`<> ? AND `available_status`='E'"; //Update query to set the target box from Empty state to Reserved state, with an expiration time of 8 minutes. (Target box reserved by local box)

549 - 553	if (Number(localbid) && Number(targetbid)){ // check if provide info is number
	dbConnection.execute(queryReserving
	,[localbid, targetbid, currentSessionID])
	.then(([rows]) => {
	if (rows.affectedRows == 1){
	//do statement if query changed row

F.O.C 1.19 const list_of_localCabinetboxess = (req, res) => {	
578 - 594	//skipped contain repeated syntaxes
580	var querylistofBoxess = "SELECT `box_id` FROM `boxes` WHERE `cabinet_fk` =? AND `available_status` ='E'"; //List all empty boxes which are in Empty state for the client.

F.O.C 1.20	**const list_of_faceFile_in_allocatedBox = (req, res) => {**
598 - 618	//skipped contain repeated syntaxes
600	var querylistofBoxess = "SELECT `facereg_filename` FROM `boxes` WHERE `cabinet_fk` =? AND `available_status` ='A' AND `facereg_filename`!=""; // Searching all face recognition filenames from all boxes that are hosted by the cabinet set that send the request.

F.O.C 1.21	**const list_ALL_localCabinetboxess = (req, res) => {**
621 - 637	//skipped contain repeated syntaxes
623	var querylistofBoxess = "SELECT `box_id` FROM `boxes` WHERE `cabinet_fk` =?"; //Search all boxes that hosted by the client that made request

F.O.C 1.22	**const list_of_targetCabinetboxess = (req, res) => {**
640 -664	//skipped contain repeated syntaxes
645 - 647	var querylistofBoxess = "SELECT `box_id` FROM `boxes` WHERE `cabinet_fk` =? AND `available_status` ='E' AND `cabinet_fk` !=?"; dbConnection.execute(querylistofBoxess ,[cabinetId, currentCabinetID]) //Find list of Empty state boxes from target location id

F.O.C 1.23	**const listofLocations =(req, res) => {**
667 - 686	//skipped contain repeated syntaxes
670 - 672	var querylistofLocation = "SELECT `cabinet_id`, `location_name`,`cabinet_addr` FROM `cabinet_set` WHERE `cabinet_id` <>?"; dbConnection.execute(querylistofLocation ,[currentCabinetID]) // Query to search the other cabinet set information except the client itself that requested it.

F.O.C 1.24	**function getTokenID (passtoken) {**

692 - 701	if (passtoken) {
	var authorization = passtoken.split(' ')[1],decoded;
	try {
	decoded = jwt.verify(authorization, secret_);
	//verify request token with secret key

	} catch (e) {
	console.log("error getCurrentTokenID "+decoded);
	}
	console.log("decoded.cabinet_id : "+decoded.cabinet_id);
	clientId = decoded.cabinet_id;
	return clientId;
	//return the id

F.O.C 1.25	**function updateTrasnferHandlingSetTime(transferID){**
706 - 718	//skipped contain repeated syntaxes
708 - 709	dbConnection.execute("UPDATE `transfer_allocation` SET `handling_time`=now() WHERE `acquire_id`=?" ,[transferID]) //for before transfer the first scan for the first box by the staff using Location A QR code

F.O.C 1.26	**async function NotifyEmailforParcelReached(serviceID, customerEmail, expdate, locationA, boxAid,boxAaddr, locationB, boxBid, boxBaddr, lateYesNo){**
722 - 765	//skipped contain repeated syntaxes

724 - 760	if (lateYesNo =="Yes"){
	var dt = new Date();
	dt.setHours(dt.getHours() + 2);
	
	htmlstruc = "<h3>Hi, your □ parcel been success reached to ✨boxid: "+boxBid+"✨</h3>"+
	"<h4>Currently your parcel had placed at □</h4>"+
	"<h4>Destination address: "+boxBaddr+" □</h4>"+
	"<h4>Destination name: "+locationB+" □</h4>"+
	"<h4>Destination boxID: "+boxBid+" □</h4>"+
	"<h4>Box Expiration: "+dt+" □□</h4>"+
	"<h5>Sorry that the parcel been late reaching the expire time been added 2 hours.</h5>"+
	"<h5>Transfer from previous location: "+locationA+"</h5>"+
	"<h5>previous address: "+boxAaddr+"</h5>"+
	"<h5>Previous Cabinet boxid: "+boxAid+"</5>";
	//haven't add time yet
	}else if(lateYesNo =="no"){
	htmlstruc = "<h3>Hi, your □ parcel been success reached to ✨boxid: "+boxBid+"✨</h3>"+
	"<h4>Currently your parcel had placed at □</h4>"+

	"<h4>Destination address: "+boxBaddr+" □</h4>"+
	"<h4>Destination name: "+locationB+" □</h4>"+
	"<h4>Destination boxID: "+boxBid+" □</h4>"+
	"<h4>Box Expiration: "+expdate+" □□</h4>"+
	"<h5>Remember take away the belongings before expire reach.</h5>"+
	"<h5>Transfer from previous location: "+locationA+"</h5>"+
	"<h5>previous address: "+boxAaddr+"</h5>"+
	"<h5>Previous Cabinet boxid: "+boxAid+"</5>";
	}
	// send mail with defined transport object
	let info = await transporter.sendMail({
	attachDataUrls: true,//to accept base64 content in messsage
	from: '"□Smart Cabinet Malaysia" <smartcabinetmalaysia.com>', // sender address
	to: customerEmail, // list of receivers
	subject: "Hello ✓ From Smart Cabinet Malaysia", // Subject line
	text: "Your parcel has delivered to target destination > "+locationB, // plain text body
	html: htmlstruc // html body
	});
	//(for after transfer)
	//method sent mail to notify user parcel was sent to target location

F.O.C 1.27	**function setTransferCompleteOnBoxServ(servicingID, vlate){**
767 - 789	//skipped contain repeated syntaxes

770 - 780	var setTransferComplete ="";
	if (vlate =="no"){
	console.log("value late> "+vlate);
	setTransferComplete = "UPDATE `box_servicing` SET `transfer_complete`='1' WHERE `service_id`=?";
	}else{
	console.log("value late> "+vlate);
	setTransferComplete = "UPDATE `box_servicing` SET `transfer_complete`='1',`expire_datetime`=DATE_ADD(now(),interval 2 hour) WHERE `service_id`=?";
	}
	dbConnection.execute(setTransferComplete
	,[servicingID])
	//UPDATE the box_servicing ID for the transfer_complete boolean to 1 also if the expiration time is less than an hour start from the moment the staff placed the parcel it will add 2 more hours to it.

F.O.C 1.27	**function setTransferCompleteOn_TAllocation(transferAquireID){**
792 - 804	//skipped contain repeated syntaxes
794 - 796	dbConnection.execute("UPDATE `transfer_allocation` SET `complete_time`=now(), `transfer_status`='Success' WHERE `acquire_id`=?" ,[transferAquireID]) .then(([rows]) => { // Set the transfer allocation (acquire id) to completed status and the complet time is the time that request

F.O.C 1.28	**function setEmptytoLocalBox(ttheBoxID){**
807 - 819	//skipped contain repeated syntaxes
824 - 825	dbConnection.execute("update `boxes` set `available_status`='E',`customerpass_qr`='',`staffpass_qr`='',`self_reserving`=0, `reserved_by_otherBox`=0,`reservedbyother_box_f_k`=NULL WHERE `box_id`=? AND `available_status`='A'" ,[ttheBoxID]) .then(([rows]) => { // set the local box from Allocated state to Empty state which previously picked up , and now the parcel been reached the target box.

F.O.C 1.29	**function settargetboxtoSelfReserve(boxID){**
822 - 834	//skipped contain repeated syntaxes
824 - 825	dbConnection.execute("UPDATE `boxes` SET `staffpass_qr`='',`self_reserving`='1',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL WHERE `box_id`=?" ,[boxID]) //Set the target box to self reserving state

F.O.C 1.30	**const niHao = (req, res) => {**
837 - 840	const niHao = (req, res) => { var currentSessionID = getTokenID(req.headers.authorization); res.status(200).json("ni hao ma:" + currentSessionID); }; // Ping the server and response the client if server functioning still

F.O.C 1.31	**const cancel_reserved_box = (req, res) => {**
842 - 961	//skipped contain repeated syntaxes

856 - 860	if (boxID2 !==""){
	cancel_reserve_both(res, boxID1, boxID2, currentCabinetID);
	}else{
	methodSetLocalboxToEmpty(res, boxID1, boxID2, currentCabinetID);
	}
	// if the boxID2 value is not empty call cancel_reserve_both function
	//else call methodSetLocalboxToEmpty function

F.O.C 1.32	**function methodSetLocalboxToEmpty(res, boxId,boxId2, currentCabinetID){**
864 - 882	//skipped contain repeated syntaxes
856 - 868	var querysetBacktoEmptyForLocalBox = "UPDATE `boxes` SET `available_status`='E',`self_reserving`='0',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL WHERE `box_id`=? AND `cabinet_fk`= ? AND `available_status`='R'"; dbConnection.execute(querysetBacktoEmptyForLocalBox ,[boxId, currentCabinetID]) // method set local box to empty (cancel reserving)

F.O.C 1.33	**function cancel_reserve_both (res, localboxID, targetboxID, currentCabinetID) {**
885 - 902	//skipped contain repeated syntaxes
888 - 890	var setBacktoEmptyForTargetBox = "UPDATE `boxes` SET `available_status`='E',`self_reserving`='0',`reserved_by_otherBox`='0',`reservedbyother_box_f_k`=NULL WHERE `box_id`=? AND `reservedbyother_box_f_k`= ? AND `available_status`='R'"; dbConnection.execute(setBacktoEmptyForTargetBox ,[targetboxID, localboxID]) // cancel_reserve_both (trigger while after going back to the Main page)

3.4.2 Remote client ware code (F.O.C 2.1 – 3.13)

Client ware app.js (F.O.C 2.1)

App.js is the main file that the clientware will first run.

F.O.C 2.1	**app.js (Code)**
1 - 8	var express = require('express'); //is a framework middleware to respond to HTTP Requests to the localhost page.

	var path = require('path'); //handling the directory to read files within the project folder.
	
	var logger = require('morgan');
	//HTTP request logger middleware for Node. js
	
	var index = require('./routes/index'); //require to import the route for request
	
	var app = express(); //initialize express call object
	
	var isJSON = require('is-json'); // use to verify JSON type
	
	let functionCallApi = require("./routes/callApi"); //require callApi functions
	var bodyParser = require('body-parser');
	
	//takes responsibility to handle the type of request sent from the browser.

10 - 14	setInterval(myFunctionLoop, 1000 * 60 * 60* 23);
	function myFunctionLoop(){ functionCallApi.callLogin(); }
	
	//Every 23 hours run this //to actively refresh the token making sure client able to connect to the server

17 - 20	app.set('views', path.join(__dirname, 'views'));
	app.set('view engine', 'ejs');
	app.engine('html', require('ejs').renderFile);
	//require the express set up the HTML views to prepare load by the ejs engine Embedded JavaScript templating to generate HTML render response.

23 - 24	app.use(bodyParser.urlencoded({ extended: false }));
	app.use(bodyParser.json());
	//bodyParser is for parsing {urlencoded} bodies and only looks at requests where the Content-Type header matches the JSON type option

27	app.use('/', index); //root route

30 - 41	app.use(function(req, res, next) {
	var err = new Error('Not Found');
	err.status = 404; next(err); }); // error handler
	
	app.use(function(err, req, res, next) {
	res.status(err.status || 500);
	res.render('error', {status:err.status, message:err.message});
	}); // render the error page
	//error handler used to respond back if there were no existing endpoints or mismatch for the request requirement

Client ware ejs files

There are a total of 13 ejs files within the views folder which provide the client ware to render pages. Most of the pages are consistently designed therefore here only will pick something special to explain. Start from Appendix E_b Figure 2.1 to Appendix E_b Figure 2.9. Those files are index.ejs, local_store.ejs, localstore_user_form.ejs, select_boxunlock.ejs, transf_store.ejs, transferstore_selecttargetbox.ejs, transferstore_user_form.ejs, transfstore_selectplace.ejs, unlock.ejs, unlockingstatus.ejs. The file naming is meant for what it intends to serve which the views are all shown at Appendix D.

As the ejs page will work somehow just like PHP able to echo the variable to the HTML page content. Regularly the page content such as front end css , header and Jquery JavaScript are able to be reused by just using syntax <%=value%>. Based on Appendix E_b Figure 2.3 and 2.4 it shows that ejs engines are able to manage to inject the content to ordinary pages conveniently. For the user interface in order to let customer able to select the list of destination show on Appendix D Figure 3.2,some of the ejs are applied with JQuery code shows on Appendix E_b 2.1 make the page able to listen to the response from server and change or add the HTML append on the dropdownlist. But In the case of the current project, the request will be sent to localhost itself at first and afterward the client itself will forward again the response to the server since all of the pages are not necessarily handled by session and no require login by user.

From Appendix E_b 2.2 line 82 to 90, the action="/selected_box_to_unlock" is the client having post to self endpoint. All existing actions such as including the calling the QR code scanning are called from the page form referring to Appendix E_b 2.7. There is another way to prove that ejs work so alike PHP, as the ejs page is able to according to logic to render what it should. Example if the value is provided from the response inside the <% %> syntax is able to include the if else or loop so that certain information will keep showing or hiding until the statement meets and it would change view on page example referring to Appendix E_b Figure 2.8.

Client ware route/apiCall.js (F.O.C 2.2 – 2.23)

Here are all functions that use to listen to the server response with JSON encode from apiCall.js file. Most of them are similar with the same setting such as HTTP Header which require the bearer token; the difference of them is just HTTP methods (PUT, GET, POST), and the request body is based on the purpose to send requisite data. There are a total 15 callable functions which will get invoked by the index.js. The reason to call this client as client ware is because whatever that user did to control, is basically require the token provided by the server which only the client ware system knows about the credential information. Since most of the REST API calling the code patterns are almost identical here will only explain those syntaxes which are unfamiliar around.

F.O.C 2.2	**Repeated Syntaxes in apiCall.js (Code)**
Defining end point	var url = host+"/api/client/endpointX";
Data prepare to sent to server	var data = "";

JSON object example	let oginObj={
	"Req1": valuex,
	"Req2": value Y
	};

Request option 1	requestify.get(url, {
	method: 'GET',//'PUT','POST'
	body: data,
	dataType: 'json',
	headers :{
	Authorization:"Bearer " + tokenn
	}

Request option 2	var options = {
	'method': 'GET',//'PUT','POST'
	'url': url,
	'headers': {
	'Content-Type': 'application/json',
	Authorization:"Bearer " + tokencache // token
	},
	body: JSON.stringify(oginObj),
	timeout: 10000
	};

Read server response 1	var responsee = response.getBody(); copyresponse = JSON.stringify(responsee);
Read server response 2	copyresponse = JSON.parse(response.body);

Error handler syntax default	function(error) {
	//statement
	});
	if (error) throw new Error(error);

F.O.C 2.3 apiCall.js (Code)	

1 - 3	const request = require('request');
	var http = require('http');
	var requestify = require('requestify');
	// there are request and requestify libraries both are used for making requests to

	the server. Http is for handling response
4 - 5	var fs = require('fs');var ncp = require("ncp"); // fs is having control over filesystem, require ncp is for having file copy to another directory.
6	const del = require('del'); // del is for deleting files in pattern for it's name will be used on deleting the expired face recognition dataset.
7	var nrc = require('node-run-cmd'); //Used for directly calling python to run without needing to listen to the console response.
8	var isJSON = require('is-json'); //The is-json is for checking if the specific data belongs to JSON format.
9	const c_stripePublicKey = 'pk_test' //The the Stripe API public key, third party API used to simulate payment using a credit card.
10	require('dotenv').config(); //Values here require dotenv for the node js able to have control over the system variable inside the .env file.
11	const clientName = process.env.CLIENTUSRM || 'home'; // 'CLIENTUSRM' value from file and set to the 'clientName' constant if the file does not have the value it just sets default value as home.

F.O.C 2.4	**extractToken(bodyretrunedAfterlogin)**

22 - 30	var xcopyObj = "";
	var obj1 = bodyretrunedAfterlogin;
	//console.log("Type of Obj1 "+typeof(obj1));
	xcopyObj = obj1.token;
	//console.log("xcopyObj value: "+xcopyObj);
	
	var currenttoken = "";
	if (xcopyObj.startsWith("Bearer ")){
	currenttoken = xcopyObj.substring(7, xcopyObj.length);
	/* taking the token string save it into the environment variable
	{ "success": true,
	"token": "Bearer only want the string here",
	"address": " place" };
	Extract the token*/

33 - 35	const clientusrm = process.env.CLIENTUSRM;
	const clientpass = process.env.CLIENTPASS;
	tokencache = currenttoken;
	//Prepare the string to update the .env value
	//tokencache is the global variable on retaining new token value copy the extracted new token from currenttoken variable

36 - 39	updateToken = "CLIENTUSRM="+clientusrm+"\r\n"+
	"CLIENTPASS="+clientpass+"\r\n"+
	"CLIENTOKEN="+currenttoken;
	fs.writeFile('.env', updateToken, function (err) {
	if (err) throw err;
	});
	//Combining 3 string values and write the string into the .env file

42	callBackend(); //Calling callBackend() function.

F.O.C 2.5	**callBackend()**

50 - 80	function callBackend(){
	//skipped contain repeated syntaxes
	}

69 - 72	if (copyresponse.includes("ni hao ma:")){
	console.log("Yes reponsed");
	return "responsed";
	}else{console.log("No reponsed"); }
	// If the server had a response which included “ni hao ma” in JSON string means that the client had successful contact with the server.

74	tokencache = process.env.CLIENTOKEN++; //update the global value with the latest CLIENTOKENT read from .env

F.O.C 2.6	**callLogin()**

83 - 112	function callLogin(){
	//skipped contain repeated syntaxes
	}

103	extractToken(copyresponse); //calling extractToken() function

F.O.C 2.7	**callRetrieveListOfLocations(respass)**
116 - 154	function callRetrieveListOfLocations(respass){

	//skipped contain repeated syntaxes }

136 - 138	if (copyresponse.includes("cabinet_id")){
	console.log("Retrieved list of locations ");
	}else{console.log("No reponsed for calling/api/client/list_of_location");}
	// if server response including keyword “cabinet_id” console.log

139	var trimmed = findsubstr(copyresponse); // trimme the server responded value and return to trimmed using findsubstr()
145	respass.send(trimmed); //response value to back to the page that requested

F.O.C 2.8	**callRetrieveListOfBoxsLocal(respass)**

157 - 195	function callRetrieveListOfBoxsLocal(respass){
	//skipped contain repeated syntaxes
	}

177 - 179	if (copyresponse.includes("box_id")){
	console.log("Retrieved list of empty boxes(current cabinet) ");
	}else{console.log("No expected reponsed for calling
	/api/client/list_of_localCabinetboxes");}
	// if server response including keyword “box_id” console.log

186	respass.send(trimmed); //response value to back to the page that requested

F.O.C 2.9	**callRetrieveListOfBoxsTarget(respass, cabid)**

198 - 237	function callRetrieveListOfBoxsTarget(respass, cabid){
	//skipped contain repeated syntaxes
	}

219 - 228	if (copyresponse.includes("box_id")){
	console.log("Retrieved list of empty boxes(current cabinet) ");
	}else{console.log("No expected reponsed for calling
	/api/client/list_of_localCabinetboxes");}
	var trimmed = findsubstr(copyresponse);
	
	respass.send(trimmed);
	// if server return include keyword “box_id” console.log else console.log
	//response value to back to the page that requested

F.O.C 2.10	**callRetrieveListALLBoxsLocal(respass)**

240 - 277	function callRetrieveListALLBoxsLocal(respass){
	//skipped contain repeated syntaxes
	}

261 - 268	if (copyresponse.includes("box_id")){
	console.log("Retrieved list of empty boxes(current cabinet) ");
	trimmed = findsubstr(copyresponse);
	}else{console.log("No expected reponsed for calling /selectboxtounlock_list");}
	response.body;
	console.log("trimmed output: "+trimmed)
	console.log("trimmed"+isJSON(trimmed));
	respass.send(trimmed);
	// if server return include keyword “box_id” console.log else console.log
	//response value to back to the page that requested

F.O.C 2.11	**callReserveLocalbox(respass, boxid, serviceType)**

280 - 338	function callReserveLocalbox(respass, boxid, serviceType){
	//skipped contain repeated syntaxes
	}

305 - 314	if (serviceType == "L_S"){} //if service type is equal local storage

306 - 313	if (copyresponse.includes("Reserved")){
	console.log("Reserved selected box id");//keys.stripePublishableKey
	respass.render('localstore_user_form', {page:'User details',
	menuId:'l_s4b',rlocalboxid: boxid,
	stripePublishableKey: c_stripePublicKey});
	}else{
	respass.render('local_store', {page:'Local Storage',
	menuId:'l_s4'});
	//if the server responded value including keyword “Reserved”
	Response render localstore_user_form.ejs
	Else render local_store.ejs

314	}else if(serviceType == "T_S"){ // else if service type is transfer service
315 - 322	if (copyresponse.includes("Reserved")){ console.log("reserved local box for transfer purpose");

	respass.render('transfstore_selectplace',{page:'Select 1 available destination to transfer',menuId:'contact', rlocalboxid: boxid});
	}else{
	respass.render('transf_store', {page:'Select 1 available local box',menuId:'contact'});
	//if the server responded value including keyword “Reserved”
	Response transfstore_selectplace.ejs
	Else render transf_store.ejs

325 - 327	}else{
	respass.render('index', {page:clientName, menuId:clientName});
	}
	//Else serviceType value is not either ‘T_S’ or ‘L_S’ response to render index.ejs

F.O.C 2.12 callReserveTargetbox(respass, rlocalbox, rtargetbox, targetlocationName, locationID)	

341 - 390	function callReserveTargetbox(respass, rlocalbox, rtargetbox, targetlocationName, locationID){
	//skipped contain repeated syntaxes
	}
	//rlocalbox = reserved local box , rtargetbox = target box going to reserve

365	if (copyresponse.includes("Reserved")){ //if the server responded value including keyword “Reserved”
367	respass.render('transferstore_user_form' //response transferstore_user_form.ejs page

371 - 372	}else{
	respass.render('transferstore_selecttargetbox',
	//response transferstore_selecttargetbox.ejs page

F.O.C 2.13 callUnlockForCustomer(respass, svcID, qrString, boxid)	

393 - 430	function callUnlockForCustomer(respass, svcID, qrString, boxid){
	//skipped contain repeated syntaxes
	}

418 - 419	if (copyresponse_STR.includes("boxid") &&
	copyresponse_STR.includes("yes")
	&& copyresponse.boxid ==boxid){
	// if server response including keyword “boxid” and “yes” and the responded object boxid value is equal to the boxid that passed in this function

421	respass.render('unlockingstatus', selectedbid:copyresponse.boxid, failed:"success" //response to render unlockingstatus.ejs and show the word success

424 - 426	}else{
	console.log("No expected reponsed for calling /selectboxtounlock_list");
	respass.render('unlockingstatus',selectedbid:boxid, failed:"false"
	//response to render unlockingstatus.ejs and show the word failed

F.O.C 2.14 callUnlockForCustomerWithFace(respass, labelstring, boxid)	

433 - 469	function callUnlockForCustomerWithFace(respass, labelstring, boxid){
	//skipped contain repeated syntaxes
	}

458 - 459	if (copyresponse_STR.includes("boxid") &&
	copyresponse_STR.includes("yes")
	&& copyresponse.boxid ==boxid)
	// if server response including keyword “boxid” and “yes” and the responded object boxid value is equal to the boxid that passed in this function

461	respass.render('unlockingstatus' //response to render unlockingstatus.ejs and show the word success

463 - 465	}else{
	console.log("No expected reponsed for calling /selectboxtounlock_list");
	respass.render('unlockingstatus'
	//response to render unlockingstatus.ejs and show the word failed

F.O.C 2.15 callUnlockForBeforeTransfer(respass, tranfID, beforeqrString, boxid)	

472 - 514	function callUnlockForBeforeTransfer(respass, tranfID, beforeqrString, boxid){
	//skipped contain repeated syntaxes
	}

499 - 500	if (copyresponse_STR.includes("boxid") &&
	copyresponse_STR.includes("yes")
	&& copyresponse.boxid ==boxid){
	// if server response including keyword “boxid” and “yes” and the response object boxid value is equal to the boxid that passed in this function

502	respass.render('unlockingstatus'

	/response to render unlockingstatus.ejs and show the word success

504 - 506	}else{
	console.log("No expected reponsed for calling /selectboxtounlock_list");
	respass.render('unlockingstatus'
	//response to render unlockingstatus.ejs and show the word failed

F.O.C 2.16	**callUnlockForAfterTranfser(respass, trasnfID, afterqrString, boxid)**

518 - 554	function callUnlockForAfterTranfser(respass, trasnfID, afterqrString, boxid){
	//skipped contain repeated syntaxes
	}
	if (copyresponse_STR.includes("boxid") &&
	copyresponse_STR.includes("yes")
	&& copyresponse.boxid ==boxid){
	console.log("Sent unlocked message
	>boxid:"+copyresponse.boxid);
	respass.render('unlockingstatus', {page:'Unlocking
	status', menuId:'contact',
	selectedbid:copyresponse.boxid,
	failed:"success" });
	}else{
	console.log("No expected reponsed for calling
	/selectboxtounlock_list");
	respass.render('unlockingstatus', {page:'Unlocking
	status', menuId:'contact',
	selectedbid:boxid, failed:"false" });
	}
	// if server response including keyword “boxid” and “yes” and the responded object boxid value is equal to the boxid that passed in this function
	IF TRUE response to render unlockingstatus.ejs and show the word success
	Else //response to render unlockingstatus.ejs and show the word failed

F.O.C 2.17	**callInsertBox_servAfterPayment(respass,c_email,c_phone ,c_days,c_hrs,c_price,c_f_opt,c_lbid,c_tbid,c_regfname)**

557 - 630	function callInsertBox_servAfterPayment(respass,c_email,c_phone,c_days,c_hrs,c_price,c_f_opt,c_lbid,c_tbid,c_regfname){
	//skipped contain repeated syntaxes
	}

604 - 612	if (copyresponse.trainFace =="yes"){ moveFaceFilefromTemptoRealDataset();

	respass.render('unlockingstatus', {page:'Unlocking status',
	menuId:'contact',selectedbid:copyresponse.boxid,failed:"success",
	showPaymentStatus:"Yes" });
	}
	// if server response including keyword “Yes”
	Call the function moveFaceFilefromTemptoRealDataset
	Redirect to unlockingstatus.ejs page

612 - 614	else{
	console.log("Not going to train the face file");
	respass.render('unlockingstatus'
	//Else response to render unlockingstatus.ejs and show the word failed

F.O.C 2.18 callforCancelReservedbox(c_lbid,c_tbid)	

633 - 666	function callforCancelReservedbox(c_lbid,c_tbid){
	//skipped contain repeated syntaxes
	}

F.O.C 2.19 moveFaceFilefromTemptoRealDataset()	
670 - 671	const srcDir = 'python/temp_dataset'; const destDir = 'python/dataset'; //declare the variable for source directory and destination directory
673	if (fs.existsSync(srcDir) && fs.existsSync(destDir)) { // if both paths are existing

675 - 677	try {
	ncp(srcDir , destDir ,
	function (err) {
	//copy all files from source folder to target folder

678 - 679	if (err) {
	return console.error(err);
	// if error return error

680 - 682	}else{
	console.log('Folders copied recursively');
	del.sync([srcDir+'/**']);
	//else delete all files from source file

684	retriveAvailableFaceFilenameFromAllocatedBox(); //call func()

F.O.C 2.20	**retriveAvailableFaceFilenameFromAllocatedBox()**

696 - 763	function retriveAvailableFaceFilenameFromAllocatedBox(){
	//skipped contain repeated syntaxes
	}

712 - 713	//the directory that dataset of faces to save var pathdataset_face = "python/dataset/";

717 - 719	if (copyresponse_STR.includes("facereg_filename")){
	var newArray = [];
	// array to save list of filenames for prepare to delete
	var arraytoMatch =[];
	//array to save filenames to excluded from delete

722 - 730	//example of filename User.1.1.jpg to User.1.2.jpg … to User.1.22.jpg
	//for loop to generate the filename that want to preserve
	for (var i = 0; i < responsee.length; i++) {
	for (key in responsee[i]) {
	console.log('Key: '+ key + ' Value: ' + responsee[i][key]);
	for(let step = 1; step < 22; step++){
	arraytoMatch.push(responsee[i][key]+[step]+".jpg");
	}
	}
	}

734 - 736	// acquire all files name in the 'dataset' folder var allfilesarray = fs.readdirSync(pathdataset_face, {withFileTypes: true}) .filter(item => !item.isDirectory()) .map(item => item.name)
741	//then only preserve the files name that require to delete var myfilteredArray = allfilesarray.filter((item) => !arraytoMatch.includes(item));

744 - 747	for (key in myfilteredArray) {
	//console.log('Key: '+ key + ' Value: ' + myfilteredArray[key]);
	newArray.push(pathdataset_face+myfilteredArray[key]);
	}//add directory string to the filename and prepare to delete

749 - 751	newArray.push("!"+pathdataset_face);
	//save the path to excluding the file itself
	console.log(newArray.toString()+"\n");
	irrelevant_files_delete(newArray);
	//call function to delete irrelevant files

F.O.C 2.21	irrelevant_files_delete(array)
771	del.sync(array); //delete all file which match the name that contain in the array
773	trainFaceRecog(); // call trainFaceRecog() function

F.O.C 2.22	trainFaceRecog()
783	nrc.run('python python/trainer.py'); //Running the child process trainer.py

F.O.C 2.23	cleaningTempFaceFile()

798 - 808	var pathdataset_face = "python/temp_dataset";
	if (fs.existsSync(pathdataset_face)) {
	//console.log("File Exist");
	try {
	//fs.unlinkSync(pathdataset_face);
	del.sync([pathdataset_face+'/**']);
	console.log("Files in tempdata is deleted.");
	} catch (error) {
	console.log(error);
	}
	}
	// del.sync to delete all files in the tempdata folder

Client ware route/index.js (F.O.C 2.24 – 2.49)

F.O.C 2.24	route/index.js (Code)
1	var express = require('express'); // require express framework
2	var router = express.Router(); //define the routing path to access the functions here
4	let functionCallApi = require("./callApi"); // require to import the functions exported from callApi.js
5	const stripe = require('stripe')('sk_test…') //initialize the stripe payment api with private key
6	const c_stripePublicKey = 'pk_test...'

	//initialize stripe public key
7	require('dotenv').config(); //Initialize to read environment variables
8	const clientName = process.env.CLIENTUSRM || 'home'; //read CLIENTUSRM from .env file
9	functionCallApi.callLogin(); //call function callLogin

F.O.C 2.25	**router.get('/'**

13 - 15	router.get('/', function(req, res, next) {
	res.render('index', {page:clientName, menuId:clientName});
	});
	// render index.ejs page

F.O.C 2.26	**router.get('/callbackend'**

18 - 21	router.get('/callbackend', function(req, res, next) {
	let response = functionCallApi.callBackend();
	res.send(response);
	});
	// Calling the function callBackend from callApi.js

F.O.C 2.27	**router.get('/local_store'**

24 - 26	router.get('/local_store', function(req, res, next) {
	res.render('local_store', {page:'Local Storage', menuId:'l_s4'});
	});
	// render local_store.ejs page

F.O.C 2.28	**router.post('/local_store_selected'**
29	var box_id = req.body.boxid; // init variable
31	if (box_id !="undefined"){ // check if variable is empty

32 - 33	var svcType = "L_S";
	functionCallApi.callReserveLocalbox(res, box_id, svcType);
	
	//Init variable for service type

	//call function callReserveLocalbox

34 - 36	res.render('local_store', {page:'Local Storage', menuId:'l_s4'});
	
	//if the box_id value is not defined render page local_store.ejs

F.O.C 2.29	**router.get('/localstore_user_form'**

40 - 43	router.get('/localstore_user_form', function(req, res, next) {
	res.render('localstore_user_form', {page:'User details',
	menuId:'l_s4b',rlocalboxid: '0',stripePublishableKey:'x'});
	});
	// render localstore_user_form.ejs page

F.O.C 2.30	**router.get('/transfstore_select_localbox'**

45 - 47	router.get('/transfstore_select_localbox', function(req, res, next) {
	res.render('transf_store', {page:'Select 1 available local box', menuId:'contact'});
	});
	render transfer_store.ejs page

F.O.C 2.31	**router.post('/transfer_store_selected_localbox'**
50	var box_id = req.body.boxid; // init variable get from post body
52	if (req.body.boxid !== undefined){ // if the request object is defined

53 - 54	var svcType = "T_S";
	functionCallApi.callReserveLocalbox(res, box_id, svcType);
	//var svcType is for service type as transfer storage T_S
	//call the function callReserveLocalbox

55 - 57	}else{
	res.render('transf_store', {page:'Select 1 available local box',
	menuId:'contact'});
	}
	//if request object undefined redirect to page transf_store.ejs

F.O.C 2.32	**router.get('/transfstore_select_localbox_list'**

60 - 62	router.get('/transfstore_select_localbox_list', function(req, res, next) {
	let results = functionCallApi.callRetrieveListOfBoxsLocal(res);
	});// Call function callRetrieveListOfBoxsLocal

F.O.C 2.33	router.get('/transfstore_selectplace'

64 - 68	router.get('/transfstore_selectplace', function(req, res, next) {
	res.render('transfstore_selectplace',{page:'Select 1 available destination to transfer', menuId:'contact', rlocalboxid:"Not supposed to be here (not reserving a local box yet)"});
	});
	// render transferstore_selectplace.ejs page

F.O.C 2.34	router.post('/tranfer_store_selected_location'

71 - 76	var passedresvdLBox = (req.body.reserved_localboxid !== undefined) ?req.body.reserved_localboxid : "x";
	//req.body.reserved_localboxid;
	var passedlocationID = (req.body.select_location !== undefined) ?req.body.select_location : "x";//req.body.select_location;
	var passedlocationname = (req.body.location_name !== undefined) ? req.body.location_name : "x";//req.body.location_name;
	// init variables get from post body if any empty replace with “x”

78	if (Object.keys(req.body).length !== 0 && Number(passedresvdLBox) && Number(passedlocationID)) { // if passed object exist and both value are convertible tonumber type
81 - 83	res.render('transferstore_selecttargetbox', {page:'Select 1 Target box ID to reserve',menuId:'contact', location_id: passedlocationID,location_name:passedlocationname, rlocalboxid:passedresvdLBox }); //response transferstore_selecttargetbox.ejs to page

84 - 86	}else{
	res.render('index', {page:clientName, menuId:clientName});
	}
	//Else the request body is empty response to show index.js

F.O.C 2.35	router.get('/transfstore_selectplace_list'

89 - 91	router.get('/transfstore_selectplace_list', function(req, res, next) {
	let results = functionCallApi.callRetrieveListOfLocations(res);
	});
	// Calling function callRetrieveLostOfLocation

F.O.C 2.36	router.get('/transfstore_select_targetbox'

93 - 97	router.get('/transfstore_select_targetbox', function(req, res, next) {
	res.render('transferstore_selecttargetbox', {page:'Select 1 Target box ID to allocate', menuId:'contact', rlocalboxid:'x',
	blocation_id:'x'});
	});
	// render transferstore_selecttargetbox..ejs page

F.O.C 2.37	router.get('/transfstore_select_targetbox_list/:cabid'

99-102	router.get('/transfstore_select_targetbox_list/:cabid', function(req, res, next) {
	console.log("req.params.cabid"+req.params.cabid)
	let results = functionCallApi.callRetrieveListOfBoxsTarget(res, req.params.cabid);
	});
	// Calling function callRetrieveListOfBoxsTarget

F.O.C 2.38	router.post('/transfer_store_selected_targetbox'

106 - 109	var passedresvdLBox = req.body.reserved_localboxid;
	var passedlocationID = req.body.location_id;
	var passedlocationname = req.body.location_name;
	var passedresvdTBox = req.body.boxid;
	
	// init variables get from the post body.

110 - 111	functionCallApi.callReserveTargetbox(res, passedresvdLBox, passedresvdTBox,passedlocationname, passedlocationID); //Call callReserveTargetbox function

F.O.C 2.39	router.get('/transferstore_user_form'

116 - 119	router.get('/transferstore_user_form', function(req, res, next) {
	res.render('transferstore_user_form', {page:'User details', menuId:'',location_name:'x',rlocalboxid:'x',targetbid:'x',stripePublishableKey:'x'});
	});
	// render transferstore_user_form.ejs page

F.O.C 2.40	router.get('/selectboxtounlock'
122 - 124	router.get('/selectboxtounlock', function(req, res, next) { res.render('select_boxunlock', {page:'Select Box to Unlock', menuId:'6'});

	}); // render select_boxunlock.ejs page

F.O.C 2.41	**router.get('/selectboxtounlock_list'**

126 - 128	router.get('/selectboxtounlock_list', function(req, res, next) {
	functionCallApi.callRetrieveListALLBoxsLocal(res);
	});
	// Calling callRetrieveListAllBoxsLocal function from callApi.js

F.O.C 2.42	**router.post('/selected_box_to_unlock'**

130 - 138	router.post('/selected_box_to_unlock', function(req, res, next) {
	var box_id = (req.body.boxid !== undefined) ? req.body.boxid : "";
	if (box_id !="undefined" && Number(box_id)){
	res.render('unlock', {page:'Choose an Unlocking Method', menuId:'',
	selectedbid:box_id});
	}else{
	res.render('select_boxunlock', {page:'Select Box to Unlock', menuId:'6'});
	}
	});
	// if box_id value is not undefined and changeable to number type
	Response render unlock.ejs page
	Else response render select_boxunlock.ejs

F.O.C 2.43	**router.get('/unlock'**

140 -143	router.get('/unlock', function(req, res, next) {
	var box_id = (req.params.boxid !== undefined) ? req.params.boxid : "";
	res.render('unlock', {page:'Choose an Unlocking Method',
	menuId:'',selectedbid:box_id});
	});
	// render unlock.ejs page

F.O.C 2.44	**router.post('/unlockingwithQR'**
147	var boxID = (req.body.boxid !== undefined) ? req.body.boxid : "x"; // init variables get from post body if any empty replace with “x”
150	if (req.body.boxid !== undefined && Number(boxID)){ // if boxid value is not undefined and changeable to number type
151	var pythonReturn = 'undefined'; // define a new variable to prepare reading from python output

152 - 153	var spawn = require('child_process').spawn,
	py = spawn('python', ['python/qr.py']);
	// invoke the child process which is qr.py

156 - 169	let st = new Date();
	var refreshIntervalId = setInterval(function () {
	let time = new Date() - st;
	// if time is over 40 secs, and the script has not been killed...
	if (time > 40000 && py.killed == false) {
	py.stdin.pause();
	py.kill();//kill the python process
	py.stdin.end();
	console.log('child killed: '+py.killed);
	clearInterval(refreshIntervalId);
	killmeplease(res);
	}
	}, 1000);
	
	// set the timer to count the time if over 40 second halt the qr.py child process

171 - 176	function killmeplease(res){
	console.log("loop killing");
	clearInterval(refreshIntervalId);
	res.render('unlock', {page:'Choose an Unlocking Method',
	menuId:'', popAlert:"True", selectedbid:boxID});
	}
	//internal function to kill refreshIntervalId loop

178 - 179	py.stdout.on('data', function(data){
	pythonReturn = data.toString();
	
	// reading the python console output behind the scene

182 - 208	if (pythonReturn != 'undefined'){
	//skipped
	}else{
	res.render('index', {page:clientName, menuId:clientName});
	}
	//if the python process had output do things else redirect to index.ejs page

185 - 188	if(typeof detectedQr.sid !== 'undefined' && typeof detectedQr.cqr !== 'undefined'){
	var s_sid = detectedQr.sid; var s_cqr = detectedQr.cqr;
	functionCallApi.callUnlockForCustomer(res, s_sid, s_cqr, boxID);
	}
	// if detected QR code JSON object contain sid: and cqr:
	Call function callUnlockForCustomer

190 - 194	if(typeof detectedQr.tid !== 'undefined' && typeof detectedQr.bqr !== 'undefined'){
	var s_tid = detectedQr.tid; var s_bqr = detectedQr.bqr;
	functionCallApi.callUnlockForBeforeTransfer(res, s_tid, s_bqr, boxID);
	console.log("running callUnlockForBeforeTransfer");
	}
	// if detected QR code JSON object contain tid: and bqr:
	Call function callUnlockForBeforeTransfer

196 - 199	if(typeof detectedQr.tid !== 'undefined' && typeof detectedQr.aqr !== 'undefined'){
	var s_tid = detectedQr.tid; var s_aqr = detectedQr.aqr;
	functionCallApi.callUnlockForAfterTranfser(res, s_tid, s_aqr, boxID);
	}
	// if detected QR code JSON object contain tid: and aqr:
	Call function callUnlockForAfterTranfser

201	clearInterval(refreshIntervalId); //clear the timer loop that counting the running time of child process

F.O.C 2.45 router.post('/unlockingwithFace'	
212	var boxID = (req.body.boxid !== undefined) ? req.body.boxid : "x"; // init variable get from post body if any empty replace with “x”

215 - 265	if (req.body.boxid !== undefined && Number(boxID)){
	//skipped
	}else{
	res.render('index', {page:clientName, menuId:clientName});
	}
	//if request body contain boxid and is number do things
	Else redirect to index.ejs page.

216	var pythonReturn = 'undefined'; // variable to read python output

217 - 223	var exec = require('child_process').execFile,
	py2 = py2 = exec('cmd',['/C', 'bat.bat'],{shell: false,
	detached: true,
	windowsHide: true});
	py2.on('uncaughtException', function (err) {
	console.log(err);
	});
	//because of some technical reason that I was not able to directly spraw the python code which implemented tensorflow, therefore created a bat file to call

	the python file using require('child_process').execFile.

225 - 239	let st = new Date();//timer
	var refreshIntervalId = setInterval(function () {
	let time = new Date() - st;
	// if time is over 120 secs, and the script has not been killed...
	if (time > 120000 && py2.killed == false) {
	py2.stdin.pause();
	py2.kill();//kill the running python
	py2.stdin.end();
	console.log('child killed: '+py2.killed);
	clearInterval(refreshIntervalId);
	killmeplease(res);
	}
	}, 1000);
	
	// initialize timer and loop to cout if over 120 second, if the spawned process does not respond to any output then will kill the child process.

241 - 246	function killmeplease(res){
	console.log("loop killing");
	clearInterval(refreshIntervalId);
	res.render('unlock', {page:'Choose an Unlocking Method', menuId:'',popAlert:"True", selectedbid:boxID});
	}
	//Function to kill the loop and redirect to unlock.ejs page

248 - 249	py2.stdout.on('data', function(data){
	pythonReturn = data.toString();
	//listening output from the child process on Python

252	if (pythonReturn !== 'undefined'){ // if the python returned value
254	clearInterval(refreshIntervalId); // call to kill the timer loop

255 - 257	py2.stdin.pause();
	py2.kill();//kill the python
	py2.stdin.end();
	// kill the Python child process

258	functionCallApi.callUnlockForCustomerWithFace(res, pythonReturn, boxID); //Call function callUnlockForCustomerWithFace

F.O.C 2.46 router.post('/start_facial_capture'	
269	var boxID = (req.body.boxid !== undefined) ? req.body.boxid : "x"; // init variable get from post body if any empty replace with “x”

272 - 316	if (req.body.boxid !== undefined && Number(boxID)){
	//skipped
	}else{
	res.render('index', {page:clientName, menuId:clientName});
	}
	// If request body is not empty and is number
	Else redirect to index.ejs page

273	var pythonReturn = 'undefined'; // declare new variable to read python output

274 - 275	var spawn = require('child_process').spawn,
	py3 = spawn('python', ['python/record_face.py']);
	//spawn the child process which named as record_face.py

277 - 290	let st = new Date();//timer
	var refreshIntervalId = setInterval(function () {
	let time = new Date() - st;
	// if time is over 120 secs, and the script has not been killed...
	if (time > 120000 && py3.killed == false) {
	py3.stdin.pause();
	py3.kill();//kill the python
	py3.stdin.end();
	console.log('child killed: '+py3.killed);
	clearInterval(refreshIntervalId);
	killmeplease(res);
	}
	}, 1000);
	//Set a timer loop if the python does not respond over 120 second, kill the python child process and call the killmeplease function to end this loop.

292 - 298	function killmeplease(res){
	console.log("loop killing");
	clearInterval(refreshIntervalId);
	res.render('localstore_user_form', {page:'User details',
	menuId:'',rlocalboxid:
	boxID,stripePublishableKey:c_stripePublicKey});
	}
	// Function use to halt the timer loop

300 - 301	py3.stdout.on('data', function(data){

	pythonReturn = data.toString(); //Reading the python child process output
306	if (pythonReturn !== 'undefined'){ // if the child process responded
308	clearInterval(refreshIntervalId); //kill the loop
311 - 314	res.render('localstore_user_form', {page:'User details', menuId:'l_s4b',rlocalboxid: boxID ,stripePublishableKey: c_stripePublicKey,facerecFileName: labelstrrmovelinebreak}); // response localstore_user_form.ejs to page

F.O.C 2.47	**router.get('/unlocked'**

321 - 323	router.get('/unlocked', function(req, res, next) {
	res.render('unlockingstatus', {page:'Unlocking status', menuId:'contact'});
	});
	
	// response render unlockingstatus.ejs

F.O.C 2.48	**router.post('/paymentprocess'**
326 - 334	var c_email = (req.body.customerEmail !== undefined) ? req.body.customerEmail : "x"; var c_phone = (req.body.customerPhone !== undefined) ? req.body.customerPhone : "x"; var c_days = (req.body.numb_days !== undefined) ? req.body.numb_days : "0"; var c_hrs = (req.body.numb_hours !== undefined) ? req.body.numb_hours : "1"; var c_price = (req.body.hidden_price !== undefined) ? req.body.hidden_price : "1"; var c_f_opt = (req.body.selectFaceRecog !== undefined) ? req.body.selectFaceRecog : "x"; var c_lbid = (req.body.hidden_box1id !== undefined) ? req.body.hidden_box1id : "x"; var c_tbid = (req.body.hidden_box2id!== undefined) ? req.body.hidden_box2id : ""; var c_regfname = (req.body.face_recFilename !== undefined) ? req.body.face_recFilename : ""; // init variables get from post body if any empty replace with “x”

335 - 345	var amount = 200;
	  if (Number(c_price)*100 >=amount){
	  amount = Number(c_price)*100;
	 }
	//Because Stripe Api require amount at least RM 2 and above
	//here add the checking for confirm the amount not lesser than it

346 - 349	stripe.customers.create({
	  email: req.body.stripeEmail,
	  source: req.body.stripeToken
	  })
	//initialize Stripe object

350 - 354	.then(customer => stripe.charges.create({
	  amount,
	  description: 'Payment to Smart Cabinet System',
	  currency: 'myr',
	  customer: customer.id
	  }))
	//Invoking Stripe Api to charge payment

356 - 359	.then(charge => functionCallApi.callInsertBox_servAfterPayment(res,c_email, c_phone,c_days,c_hrs,c_price,c_f_opt,c_lbid,c_tbid,c_regfname)); //calling callInsertBox_servAfterPayment with 10 arguments

F.O.C 2.49	**router.post('/cancel_reserved_box**
367 - 370	var boxID1 = (req.body.hidden_box1id4cancel !== undefined) ? req.body.hidden_box1id4cancel : ""; var boxID2 = (req.body.hidden_box2id4cancel !== undefined) ? req.body.hidden_box2id4cancel : ""; // init variables get from post body if any empty replace with “void”
372	functionCallApi.callforCancelReservedbox(boxID1,boxID2); // calling function callforCancelReservedbox with 2 arguments
373	functionCallApi.cleaningTempFaceFile(); //calling cleaningTempFaceFile to clean files in tempdata folder
374	res.render('index', {page:clientName, menuId:clientName}); //redirect to index.ejs page

.env file(client)

1	CLIENTUSRM=xxx //client username
2	CLIENTPASS=test123 //client password
3	CLIENTOKEN= //the client authentication token use to make request to server

Bat.bat file (client)

1	@echo off //set console message hidden
2	D: //drive D
3	python python/detector_liveness.py //calling the python file
4	exit() //exist cmd

3.4.2.2 Client ware Python (F.O.C 3.1 – 3.13)

qr.py – for QR code reading (F.O.C 3.1)

F.O.C 3.1	**qr.py (Code)**
1	import cv2 #import open cv

4 - 9	font = cv2.FONT_HERSHEY_SIMPLEX
	text = "Press Q to Exit"
	text2 = "QR code scanner"
	# get boundary of this text
	textsize = cv2.getTextSize(text, font, 1, 2)[0]

11	cap = cv2.VideoCapture(0) #start camera capturing
13	detector = cv2.QRCodeDetector() #use opencv to detect the QR code and decode the message

14 - 26	while True:
	  _, img = cap.read()
	  # detect and decode
	  data, bbox, _ = detector.detectAndDecode(img)
	  # check if there is a QRCode in the image
	  if bbox is not None:
	  # display the image with lines
	  for i in range(len(bbox)):

	# draw all lines
	cv2.line(img, tuple(bbox[i][0]), tuple(bbox[(i+1) % len(bbox)][0]), color=(255, 0, 0), thickness=2)
	if data:
	print(data)#"[+] QR Code detected, data:",
	break

29 - 35	textX = int((img.shape[1] - textsize[0]) / 2)
	textY = int((img.shape[0] + textsize[1]) / 2 + 190)
	textY2 = int((img.shape[0] + textsize[1]) / 2 - 190)
	# add text centered on image
	cv2.putText(img, text, (textX, textY), font, 1, (255, 255, 255), 2)
	cv2.putText(img, text2, (textX, textY2), font, 1, (255, 255, 255), 2)

36 - 37	cv2.namedWindow("img", cv2.WND_PROP_FULLSCREEN) cv2.setWindowProperty("img",cv2.WND_PROP_FULLSCREEN,cv2.WINDOW_FULLSCREEN) cv2.imshow("img", img) #OpenCv windows setting pop to middle with full screen
40 - 41	if cv2.waitKey(1) == ord("q"): #wait q key break
42 - 43	cap.release()#stop capturing cv2.destroyAllWindows()#close all windows

detector_liveness.py – Facial recognition (F.O.C 3.2 – 3.6)

F.O.C 3.2	**Detector_liveness.py(code)**

1 - 33	import os#import operating system
	import cv2#import opencv
	#import face_recognition
	import numpy as np
	from collections import defaultdict#an unordered collection of data
	# values that are used to store data values like a map.
	from imutils.video import VideoStream#used to capture frame
	from eye_net import * #import all functions from eye_net.py
	import sqlite3#require sqlite
	from pathlib import Path#require to read file path
	
	#font setting
	font = cv2.FONT_HERSHEY_SIMPLEX
	text = "Press Q to Exit"

	text2 = "Face Recognition"
	text3 ="please blinks slowly"
	text4 ="Allow only one person's face"
	# get boundary of this text
	textsize = cv2.getTextSize(text, font, 1, 2)[0]
	#font setting end
	p = Path()
	
	absolutepath = str(p.absolute())#"D:/Sem7/client/real_client/node-website-master/python/"
	absolutepathx = absolutepath.replace('\\', '/')+'/python'#require file path in /python folder
	conn = sqlite3.connect(absolutepathx+'/sqlite_database.db')#require sqlite db
	c = conn.cursor()#define sqlite cursor
	
	fname = absolutepathx+"/lbph_recognizer/trainingData.yml"#require trained LBPH file
	if not os.path.isfile(fname):#if file does not found
	print("Please train the data first> absolutepath"+absolutepathx)
	exit(0)#exit if trainingData.xml not found
	recognizer = cv2.face.LBPHFaceRecognizer_create()#Initialize LBPH Recognizer
	recognizer.read(fname)#read all label name

F.O.C 3.3	**def init():**
36 - 39	face_cascPath = absolutepathx+'/haar_features/haarcascade_frontalface_alt.xml' open_eye_cascPath = absolutepathx+'/haar_features/haarcascade_eye_tree_eyeglasses.xml' left_eye_cascPath = absolutepathx+'/haar_features/haarcascade_lefteye_2splits.xml' right_eye_cascPath =absolutepathx+'/haar_features/haarcascade_righteye_2splits.xml' #define path of the trained haar cascade feature files
41 - 44	face_detector = cv2.CascadeClassifier(face_cascPath) open_eyes_detector = cv2.CascadeClassifier(open_eye_cascPath) left_eye_detector = cv2.CascadeClassifier(left_eye_cascPath) right_eye_detector = cv2.CascadeClassifier(right_eye_cascPath) #initialize haar cascades classifier for utilize
47 - 49	video_capture = VideoStream(src=0).start()#start capturing model = load_model()#load the model from eye_net.py

51	return (model,face_detector, open_eyes_detector, left_eye_detector,right_eye_detector, video_capture) #returning all values

F.O.C 3.4 def isBlinking(history, max_Frames):	
55	for i in range(max_Frames): #for loop in range of max_Frames
56	pattern_ = '1' + '0'*(i+1) + '1' #define the pattern of the string
57 - 58	if pattern_ in history: return True #if within the history consist the 101/110 / 0110 pattern return true
59	return false #return false by default

F.O.C 3.5 def detect_and_display(model, video_capture, face_detector, open_eyes_detector, left_eye_detector, right_eye_detector, eyes_detected):	

62 - 63	frame = video_capture.read()
	#fetch existing image to value frame
	gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
	#convert to grayscale

65 - 72	# Detect faces
	faces = face_detector.detectMultiScale(
	gray,
	scaleFactor=1.2,
	minNeighbors=5,
	minSize=(50, 50),
	flags=cv2.CASCADE_SCALE_IMAGE
	)

74 - 77	# for each detected face
	for (x,y,w,h) in faces:
	# draw rectangle to face
	cv2.rectangle(frame,(x,y),(x+w,y+h),(0,255,0),3)

79	#using LBPH recognitzer to predict image ID and return confidence ids,conf = recognizer.predict(gray[y:y+h,x:x+w])

81 - 83	#fetch the labels (for lbph facial recognition)
	c.execute("select name from users where id = (?);", (ids,))
	
	result = c.fetchall() #fetch all available labels in sqlite db

	name = result[0][0] #save exist name labels to variable name

85 - 87	#fetch the labels (for lbph facial recognition)
	eyes = []#define eyes array
	face = frame[y:y+h,x:x+w]#define position of face with color
	gray_face = gray[y:y+h,x:x+w]#define position of face without color

90 - 96	# Eyes detection
	    # check if eyes are open (with glasses taking into account)
	    open_eyes_glasses = open_eyes_detector.detectMultiScale(
	        gray_face,
	        scaleFactor=1.1,
	        minNeighbors=5,
	        minSize=(30, 30),
	        flags = cv2.CASCADE_SCALE_IMAGE
	    )

98 - 101	# if open_eyes_glasses cascade detected eyes then they are open
	    if len(open_eyes_glasses) == 2:
	        eyes_detected[name]+='1'
	        for (ex,ey,ew,eh) in open_eyes_glasses:
	            cv2.rectangle(face,(ex,ey),(ex+ew,ey+eh),(0,255,0),2)
	             #draw rectangle to eye

104 - 111	# otherwise try detecting eyes using left and right_eye_detector
	    # which can detect open and closed eyes
	    else:
	        # separate the face into left and right sides
	       left_face = frame[y:y+h, x+int(w/2):x+w]
	        left_face_gray = gray[y:y+h, x+int(w/2):x+w]
	
	        right_face = frame[y:y+h, x:x+int(w/2)]
	        right_face_gray = gray[y:y+h, x:x+int(w/2)]

114 - 120	# Detect the left eye
	        left_eye = left_eye_detector.detectMultiScale(
	            left_face_gray,
	            scaleFactor=1.1,#image size is reduced for each image
	            minNeighbors=5, #number of neighbourhood to retain
	            minSize=(30, 30),#reject if smaller than the size
	            flags = cv2.CASCADE_SCALE_IMAGE
	        )

123 - 129	# Detect the right eye
	right_eye = right_eye_detector.detectMultiScale(
	right_face_gray,
	scaleFactor=1.1,
	minNeighbors=5,
	minSize=(30, 30),
	flags = cv2.CASCADE_SCALE_IMAGE
	)

131	eye_status = '1' # assume the eyes are open

135 - 149	# For each eye check whether the eye is closed.
	# If one is closed we conclude the eyes are closed
	for (ex,ey,ew,eh) in right_eye:
	color = (0,255,0) #green color
	pred = predict(right_face[ey:ey+eh,ex:ex+ew],model)
	if pred == 'closed':#eye_net predict closing eye for right eye
	eye_status='0'
	color = (0,0,255) #red color
	cv2.rectangle(right_face,(ex,ey),(ex+ew,ey+eh),color,2)
	#draw rectangle on right eye
	for (ex,ey,ew,eh) in left_eye:
	color = (0,255,0)#green color
	pred = predict(left_face[ey:ey+eh,ex:ex+ew],model)
	if pred == 'closed':#eye_net predict closing eye for left eye
	eye_status='0'
	color = (0,0,255)#red color
	cv2.rectangle(left_face,(ex,ey),(ex+ew,ey+eh),color,2)
	#draw rectangle on left eye
	eyes_detected[name] += eye_status

151 - 152	count_faces=str(len(faces))#cout number of face(s) cv2.putText(frame, count_faces, (x, y+10), cv2.FONT_HERSHEY_SIMPLEX,0.75, (0, 255, 0), 2) #draw text to the show frame

155 - 162	# Check, if the eye has blinked
	# If yes, echo the User.ID(label)
	if (isBlinking(eyes_detected[name],3) and conf < 50 and count_faces == '1'):
	print('User.'+str(ids)+'.')#print the label id
	cv2.rectangle(frame, (x, y), (x+w, y+h), (0, 255, 0), 2)
	# Display name and draw rectangle to face in color green
	y = y - 15 if y - 15 > 15 else y + 15
	cv2.putText(frame, name, (x, y),

	cv2.FONT_HERSHEY_SIMPLEX,0.75, (0, 255, 0), 2) exit(0)

F.O.C 3.6	**if __name__ == "__main__":**
168	(model, face_detector, open_eyes_detector,left_eye_detector,right_eye_detector, video_capture) = init() #run init function to retrieve back values
171	eyes_detected = defaultdict(str)#set variable to dictionary type
172 - 173	while True: frame = detect_and_display(model, video_capture, face_detector, open_eyes_detector,left_eye_detector,right_eye_detector, eyes_detected) # call the detect_and_display with passing 7 parameters.
174 - 180	textX = int((frame.shape[1] - textsize[0]) / 2) textY = int((frame.shape[0] + textsize[1]) / 2 + 190) textY2 = int((frame.shape[0] + textsize[1]) / 2 - 190) textY3 = int((frame.shape[0] + textsize[1]) / 2 - 169) textX3 = int((frame.shape[1] - textsize[0]) / 2 - 25) textX4 = int((frame.shape[1] - textsize[0]) / 2 - 100) textY4 = int((frame.shape[0] - textsize[1]) / 2 + 150) #text position settings

183 - 186	# add text centered on image
	cv2.putText(frame, text, (textX, textY), font, 1, (255, 255, 255), 2)
	cv2.putText(frame, text2, (textX, textY2), font, 1, (255, 255, 255), 2)
	cv2.putText(frame, text3, (textX3, textY3), font,1, (255, 255, 255),2)
	cv2.putText(frame, text4, (textX4, textY4), font, 1, (255, 255, 255),2)

187 - 189	cv2.namedWindow("Face Liveness Detector", cv2.WND_PROP_FULLSCREEN) cv2.setWindowProperty("Face Liveness Detector",cv2.WND_PROP_FULLSCREEN,cv2.WINDOW_FULLSCREEN) cv2.imshow("Face Liveness Detector", frame) #opencv window setting to fullscreen position in middle of screen
190 - 193	if cv2.waitKey(1) & 0xFF == ord('q'): #wait for q key to exit break cv2.destroyAllWindows()#halt opencv video_capture.stop()#stop frame capture

eye_net.py (F.O.C 3.7 - 3.8)

The following section of code in F.O.C 3.7 line 63 - 72 is for loading a pre-trained weight and model configuration for convolutional neural networks which is based on LeNet5 architecture. The CNN is trained by 4800 eye open and closing images in size of 24x24. The loaded model will compile the model with the 'binary_crossentropy' loss function and 'ADAM' cost optimization algorithm. By referring to Appendix D figure 9, the model will consist of 2 convolution layers, 2 pooling layers, 1 flattening layer and 3 dense layers for output the value on classifying the eye image as belonging to closing or opening (F.O.C 3.8 LINE 108 - 112 model.predict(img)).

F.O.C 3.7	**def load_model():**

63 - 72	json_file = open(absolutepathx +'/eye_model/model.json', 'r')#find model.json
	loaded_model_json = json_file.read()#copy to this variable
	json_file.close()#close the file after read
	#Parses a JSON model configuration string and returns a model instance
	loaded_model = model_from_json(loaded_model_json)
	# load weights into new model
	loaded_model.load_weights(absolutepathx +"/eye_model/model.h5")# load the trained weight files.
	#complie the model with loss function and optimizer setting
	loaded_model.compile(loss='binary_crossentropy', optimizer='adam', metrics=['accuracy'])
	return loaded_model #RETURN the model that complied

F.O.C 3.8	**def predict(img, model):**

108 -112	img = Image.fromarray(img, 'RGB').convert('L')#convert to grayscale
	img = imresize(img, (IMG_SIZE,IMG_SIZE)).astype('float32')#resize to 24x24
	img /= 255 #only allow 0 - 255 value
	img = img.reshape(1,IMG_SIZE,IMG_SIZE,1)#configure the input shape array
	prediction = model.predict(img)#call model predict to predict the class of image

113 -119	if prediction < 0.1:
	prediction = 'closed'#eye is probably closed
	elif prediction > 0.9:
	prediction = 'open'#eye is probably opening
	else:
	prediction = 'idk'#unknown state
	return prediction

record_face.py - for face cropping procedure (F.O.C 3.9)

F.O.C 3.9	record_face.py (Code)

1 - 43	import cv2#import opencv
	import numpy as np#import numpy
	import sqlite3#import sqlite 3
	import os#import operating system for file manage
	import screeninfo#import to gather screen detail
	from pathlib import Path#import parthlib to get absoolute path
	p = Path()
	
	screen_id = 0
	screen = screeninfo.get_monitors()[screen_id]#gather first screen info
	#gather abosolute path of the /python directory
	absolutepath = str(p.absolute()).replace('\\', '/')+'/python'
	conn = sqlite3.connect(absolutepath +'/sqlite_database.db')
	#connect sqlite db
	
	#temporary file when user cancel form the facial data that recently captured will able delete instantly
	if not os.path.exists(absolutepath +'/temp_dataset'):
	os.makedirs(absolutepath +'/temp_dataset')
	#create directory if not exist
	
	#the real file to train, images will move from temp to here
	if not os.path.exists(absolutepath +'/dataset'):
	os.makedirs(absolutepath +'/dataset')
	#create directory if not exist
	
	c = conn.cursor()#initialize db cursor
	
	face_cascade = cv2.CascadeClassifier(absolutepath +'/haar_features/haarcascade_frontalface_default.xml')#location of the haar cascade file that place
	
	cap = cv2.VideoCapture(0)#start camera capture
	width = cap.get(cv2.CAP_PROP_FRAME_WIDTH)
	#gather the width of frame
	height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
	#gather the height of frame
	#default name as''
	uname = ''known''
	#execute sqlite command
	c.execute('INSERT INTO users (name) VALUES (?)', (uname,))
	#read the last row ID

	uid = c.lastrowid
	
	sampleNum = 0#use to count amount of captured image
	
	# org font setting for position start from bottom-left corner
	org = (50, 50)
	
	# Line thickness of 2 px
	thickness = 2

44 - 54	while True: #condition in true
	ret, img = cap.read()#capture frame
	y = int((width-height)//2)#find actually middle point of frame
	gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)#convert frame to greyscale
	faces = face_cascade.detectMultiScale(gray, 1.3, 5)#detect the exist face on frame
	for (x,y,w,h) in faces:
	sampleNum = sampleNum+1#increment count
	
	#write the detected face image to tempdata folder
	cv2.imwrite(absolutepath +"/temp_dataset/User."+str(uid)+"."+str(sampleNum)+".jpg",gray[y:y+h,x:x+w])
	cv2.rectangle(img, (x,y), (x+w, y+h), (255,0,0), 2)
	cv2.putText(img, 'Scanning', org, cv2.FONT_HERSHEY_SIMPLEX, 1, (255, 0, 0) , thickness, cv2.LINE_AA)
	#cv2.waitKey(100)

56 - 65	window_name = 'img' cv2.namedWindow(window_name, cv2.WND_PROP_FULLSCREEN) cv2.moveWindow(window_name, screen.x - 1, screen.y - 1) cv2.setWindowProperty(window_name, cv2.WND_PROP_FULLSCREEN,cv2.WINDOW_FULLSCREEN) # add window setting to full screen,show middle of screen cv2.imshow(window_name,img)#show frame cv2.waitKey(1);#automatically close if sampleNum > 20:#if more than 20 images print('User.'+str(uid)+'.') break #halt
66 - 71	cap.release()#release camera

	conn.commit()#anchor sqlite to stop using it conn.close()#quit connection with sqlite cv2.destroyAllWindows()#cv close windows

trainer.py - LBPH trainer (F.O.C 3.11 – 3.12)

F.O.C 3.11	**Trainer.py/(Code)**
1 - 12	import os#import os import cv2#import opencv import numpy as np #import numpy from PIL import Image#import pillow from pathlib import Path#require read path p = Path() absolutepath = str(p.absolute()).replace('\\', '/')+'/python' recognizer = cv2.face.LBPHFaceRecognizer_create()#define opencv LBPH recognizer path = absolutepath +'/dataset' if not os.path.exists(absolutepath +'/lbph_recognizer'):#check if the file exist os.makedirs(absolutepath +'/lbph_recognizer')

30 - 34	Ids, faces = getImagesWithID(path)
	#call to extract image data from dataset folder
	recognizer.train(faces,Ids)
	#LBPH Training process
	recognizer.save(absolutepath + '/recognizer/trainingData.yml')
	#save trained file as yml
	print("data trained")
	cv2.destroyAllWindows()#exit cv

F.O.C 3.12	**def getImagesWithID(path):**
16	imagePaths = [os.path.join(path,f) for f in os.listdir(path)] #fetch all image from the path given
17 - 18	faces = []#init variable IDs = []#init variable
19	for imagePath in imagePaths: #read all existing image

20 -25	faceImg = Image.open(imagePath).convert('L')
	#converting image to greyscale
	faceNp = np.array(faceImg,'uint8')
	#encode face image to unit8 array

	ID = int(os.path.split(imagePath)[-1].split('.')[1]) #split the name pattern from User.1.12 take the middle value(id) faces.append(faceNp)#take the encoded value push into the faces array IDs.append(ID)#take the extracted id from image files name push to IDs array
	return np.array(IDs), faces #return IDs array values and faces

F.O.C 3.13 create_label_database.py /(Code) **//Create sqlite3 db for storing new faces' label**	
1 - 18	import sqlite3 conn = sqlite3.connect('sqlite_database.db') c = conn.cursor() sql = """ DROP TABLE IF EXISTS users; CREATE TABLE users (id integer unique primary key autoincrement, name text); """ c.executescript(sql) #execute sqlite3 script conn.commit() #handle to summarize action for db conn.close()

3.4.3 Staff portal Index.js (F.O.C 4.1 - 4.20)

F.O.C 4.1 Index.js (Code)	
1	const express = require('express');//require making http response
2	const path = require('path');//require to gather the path information
3	const cookieSession = require('cookie-session');//require to manage the session key
4	const bcrypt = require('bcryptjs'); //require to decrypt the password in db
5	const dbConnection = require('./database');//read the database config

6	const { body, validationResult } = require('express-validator'); //require to handle request from session
7	const cryptoRandomString = require('crypto-random-string'); //require to generate true random strings
9 - 10	var transferStatus ="awaiting to next"; var transferId = "Proceed to NEW task~"; //values for ejs view file to propagate
13	app.use(express.static(__dirname+'/public')); //require to share public resource file for front end
14	app.use(express.urlencoded({extended:false})); //decode the pattern of the url

17 - 18	app.set('views', path.join(__dirname,'views'));
	app.set('view engine','ejs');
	// Set the view engine to appen ejs file from view folder

21 - 25	app.use(cookieSession({
	name: 'session',
	keys: ['key1', 'key2'],
	maxAge: 3600 * 1000 // set SESION duration for 1 hour
	}));
	// apply session cookie for middleware

43 - 53	app.get('/public/qrcode.js', function(req, res) {
	res.sendFile(path.join(__dirname, 'public', 'qrcode.js'));
	});
	app.get('/public/jquery.min.js', function(req, res) {
	res.sendFile(path.join(__dirname, 'public', 'jquery.min.js'));
	});
	app.get('/public/homeqr.js', function(req, res) {
	res.sendFile(path.join(__dirname, 'public', 'homeqr.js'));
	});
	// send JavaScript file to front end page

488	app.listen(5000, () => console.log("Server is Running...port 5000")); //Listening to the port 5000

F.O.C 4.2 const ifNotLoggedin = (req, res, next)	

28 - 33	if(!req.session.isLoggedIn){
	return res.render('login-register');
	}

	next(); //check if the page session is invalid is true redirect to login-register.ejs

F.O.C 4.3 const ifLoggedin = (req,res,next) => {	

35 - 40	if(req.session.isLoggedIn){
	return res.redirect('/home');
	}
	next();
	//check if the page session is still valid is true redirect to home.ejs

F.O.C 4.4 app.get('/', ifNotLoggedin, (req,res,next) => {	

58 - 59	dbConnection.execute("SELECT A.`localbox_fkid` , A.`targetbox_fkid` , B.`transfer_status`, B.`acquire_id`, C.`staffpass_qr`, D.`staffpass_qr` AS `staffpass_qr2`, E.`staff_name` ,F.`cabinet_addr`, G.`cabinet_addr` AS `cabinet_addr2` FROM `box_servicing` AS A INNER JOIN `transfer_allocation`AS B ON B.`box_servicing_fk` = A.`service_id` INNER JOIN `boxes` AS C ON C.`box_id` = A.`localbox_fkid` INNER JOIN `boxes` AS D ON D.`box_id` = A.`targetbox_fkid` INNER JOIN `staff` AS E ON B.`staff_fkid` = E.`staff_id` INNER JOIN `cabinet_set` AS F ON F.`cabinet_id` = C.`cabinet_fk` INNER JOIN `cabinet_set` AS G ON G.`cabinet_id` = D.`cabinet_fk` WHERE B.`staff_fkid` = ? ORDER BY B.`acquire_time` DESC LIMIT 1",[req.session.userID])
	.then(([rows]) => {
	//Select the current transfer allocation id and all information for staff to follow the details to do the transferring job. It requires information such as the cabinet address, boxes id and also the QR code string.

64	if (rows.length > 0){ //if row exist
65 - 72	transferID = (typeof rows[0].acquire_id !== "undefined") ? rows[0].acquire_id : "Gather New Task~"; t_status = (rows[0].transfer_status !=="") ? rows[0].transfer_status : "None"; qrPass1 = (rows[0].transfer_status =="Pending") ? rows[0].staffpass_qr : "None"; qrPass2 = (rows[0].transfer_status =="Pending") ? rows[0].staffpass_qr2 : "None"; boxID1 = (rows[0].localbox_fkid !=="") ? rows[0].localbox_fkid : "None"; boxID2 = (rows[0].targetbox_fkid !== "") ? rows[0].targetbox_fkid : "None"; boxAddr1 = (rows[0].cabinet_addr !=="") ? rows[0].cabinet_addr : "None";

	boxAddr2 = (rows[0].cabinet_addr2 !== "") ? rows[0].cabinet_addr2 : "None"; //Define variable to copy the object value from result after query with default value if there are empty result for those value

74 - 82	var jsonQRpass1 = JSON.stringify({
	tid: transferID,
	bqr: qrPass1
	}) ;
	
	var jsonQRpass2 = JSON.stringify({
	tid: transferID,
	aqr: qrPass2
	}) ;
	//construct 2 JSON object , for letting the front end page able to generate the QR code

84 & 86	res.render('home',{ //render home.ejs page

F.O.C 4.5 app.post('/register', ifLoggedin,	

116 - 121	body('user_email','Invalid email address!').isEmail().custom((value) => {
	return dbConnection.execute('SELECT `staff_email` FROM `staff` WHERE `staff_email`=?', [value])
	.then(([rows]) => {
	if(rows.length > 0){
	return Promise.reject('This E-mail already in use!');
	}
	//Check user_email request object is in email format
	If it is email format the query will search for whether the staff table has an existing record associated with it.
	If the row was found in record append error message.

125 - 126	body('user_name','Username is Empty!').trim().not().isEmpty(), body('user_pass','The password must be of minimum length 6 characters').trim().isLength({ min: 6 }), //Check the request username is not empty, check the password is at least longer than 6 characters.
130	const validation_result = validationResult(req); //check the request body is valid value
131	const {user_name, user_pass, user_email} = req.body; //pass the request object to defined variable

133	if(validation_result.isEmpty()){ IF validation_result has no error
135	bcrypt.hash(user_pass, 12).then((hash_pass) => {
137 - 139	dbConnection.execute("INSERT INTO `staff`(`staff_name`,`staff_email`,`staff_pass`) VALUES(?,?,?)",[user_name,user_email, hash_pass]) .then(result => { res.send(`your account has been created successfully, Now you can <a href="/">Login</a>`); //Insert new staff rows to the stuff table, if success return message.

140 - 143	}).catch(err => {
	// THROW INSERTING USER ERROR'S
	if (err) throw err;
	});
	//catch the error

150 - 159	else{
	// COLLECT ALL THE VALIDATION ERRORS
	let allErrors = validation_result.errors.map((error) => {
	return error.msg;
	});
	// REDERING login-register PAGE WITH VALIDATION ERRORS
	res.render('login-register',{
	register_error:allErrors,
	old_data:req.body
	});
	//if form validation not meet requirement send error message to page

F.O.C 4.6 app.post('/', ifLoggedin, [	

165 - 172	body('user_email').custom((value) => {
	return dbConnection.execute('SELECT `staff_email` FROM `staff` WHERE
	`staff_email`=?', [value])
	.then(([rows]) => {
	if(rows.length == 1){
	return true;
	}
	return Promise.reject('Invalid Email Address!');
	});
	//Search for user_email in staff table, if found return true else append error

174	body('user_pass','Password is empty!').trim().not().isEmpty(),

	//check password is not empty

176 - 177	const validation_result = validationResult(req);
	//form validation
	const {user_pass, user_email} = req.body;
	//copy request objects

178	if(validation_result.isEmpty()){ //if form without error
183	bcrypt.compare(user_pass, rows[0].staff_pass).then(compare_result => { //bcrypt function to compare query result pass with requested pass.

184 - 189	if(compare_result === true){
	req.session.isLoggedIn = true;
	req.session.userID = rows[0].staff_id;
	req.session.userName = rows[0].staff_name;
	res.redirect('/');
	}
	// if the result is true create session with 3 values , redirect to root page

190 - 193	else{
	res.render('login-register',{
	login_errors:['Invalid Password!']
	});
	// redirect to login-register.ejs page and show error message

F.O.C 4.7 app.get('/logout',(req,res)=>{	

218 - 219	req.session = null;
	res.redirect('/');
	//destroy session

F.O.C 4.8 app.post('/canceltask', ifNotLoggedin, function(req, res){	
226 - 227	dbConnection.execute("SELECT * FROM `transfer_allocation` WHERE `staff_fkid` =? AND `transfer_status` = 'Pending' AND `handling_time` IS NULL ORDER BY `acquire_id` DESC LIMIT 1" ,[req.session.userID]) //search for the latest transfer allocation record for the staff which transfer_status is still ‘Pending’ state and also handling_time column is empty.

228 - 230	.then(([rows]) => {
	//console.log(rows[0]);
	if (rows.length > 0){
	// if row exist

231 - 232	bs_id = rows[0].box_servicing_fk;
	transferID_ = rows[0].acquire_id;
	//copy result to variable

236 - 238	setTransferHandlingToZero(bs_id); // call setTransferHandlingToZero function
	cancelTheCurrentAquiredTask(transferID_);
	// call cancelTheCurrentAquiredTask function
	res.redirect('/'); // redirect to root page

F.O.C 4.9 app.get('/gathertask',ifNotLoggedin, function(req, res){	

250 - 254	dbConnection.execute("SELECT * FROM `transfer_allocation` WHERE `staff_fkid`= ? order by `acquire_id` DESC LIMIT 1"
	,[req.session.userID])//,[]
	.then(([rows]) => {
	//search the latest transfer allocation id still in 'Pending' state
	if (rows.length > 0){

256 - 261	if(rows[0].transfer_status=='Pending'){
	console.log('Not finishing current task');
	console.log('Current task still need to be done/cancel~');
	res.redirect('/');
	}else{
	console.log('no latest Pending task, seeking and acquiring new task');
	checkNewTransferQuest(req.session.userID);
	res.redirect('/');
	}
	//if the status is still 'Pending' do nothing return to root page else call checkNewTransferQuest() function

F.O.C 4.10 app.get('/count_total_newtask', ifNotLoggedin, function(req, res){	

281 - 283	dbConnection.execute("SELECT COUNT(`service_id`) AS countts FROM `box_servicing` WHERE `transfer_handling`=0 AND `transfer_complete`=0 AND `service_type`='T_S'"
	,[req.session.userID])
	.then(([rows]) => {
	//select and count total rows of new T_S from box_servicing table

285 - 288	if (rows.length > 0){
	tscount = rows[0].countts;
	res.status(200).send((tscount).toString());
	}
	//if record found response to page with the total value

288 - 291	}else{
	console.log('No result for T_S count');
	res.status(200).send('0');
	}
	//sent the value zero to page

F.O.C 4.11 function addRowTotransfrAllocation(dataStaffID, box_servicingID)	

300 -302	dbConnection.execute("INSERT INTO `transfer_allocation` (`staff_fkid`, `box_servicing_fk`, `transfer_status`, `acquire_time`) VALUES (?, ?, 'Pending', current_timestamp())"
	,[dataStaffID, box_servicingID])
	.then(([rows]) => {
	//Insert new row to transfer_allocation as allowing staff to handling the new task.

F.O.C 4.12 function update_staffQR(boxID){	
318	var randomHash = cryptoRandomString({length: 20, type: 'alphanumeric'}); //create random hash as 20 character long in alphanumeric

320 - 322	dbConnection.execute("UPDATE `boxes` SET `staffpass_qr`=? WHERE `box_id`= ?"
	,[randomHash, boxID])
	.then(([rows]) => {
	//Update the staffpass_qr to the box_id with the new hash

F.O.C 4.13 function retrieveCurretTaskWithQR(StaffID, t_acquireID){	
340 - 342	var queryCurrentTask = "SELECT A.`localbox_fkid` , A.`targetbox_fkid` , B.`transfer_status`, B.`acquire_id`, C.`staffpass_qr`, D.`staffpass_qr` AS `staffpass_qr2` , E.`cabinet_addr`, F.`cabinet_addr` AS `cabinet_addr2` FROM `box_servicing` AS A INNER JOIN `transfer_allocation` AS B ON B.`box_servicing_fk` = A.`service_id` INNER JOIN `boxes` AS C ON C.`box_id` = A.`localbox_fkid` INNER JOIN `boxes` AS D ON D.`box_id` = A.`targetbox_fkid` INNER JOIN `cabinet_set` AS E ON E.`cabinet_id` = C.`cabinet_fk` INNER JOIN `cabinet_set` AS F ON F.`cabinet_id` = D.`cabinet_fk` WHERE B.`staff_fkid` = ? AND B.`acquire_id` = ? ORDER BY B.`acquire_time` DESC LIMIT 1"; dbConnection.execute(queryCurrentTask ,[StaffID, t_acquireID]) //check current transfer details (after acquired task) RETRIEVE QR PASS
345 - 356	if (rows.length > 0){ transferStatus = rows[0].transfer_status;

	transferID_ = rows[0].acquire_id;
	console.log("retrieveCurretTask runned once");
	console.log(rows[0]);
	//console.log(transferID_);
	return rows[0];
	res.render('home',{
	name:req.session.userName,
	transfer_status:transferStatus,
	transfer_id : transferID_
	});
	// if row exist copy result to variable and response to show value on home.ejs page

F.O.C 4.14	**function setTransfer_handling(servicingID){**

369 - 371	dbConnection.execute("UPDATE `box_servicing` SET `transfer_handling`=1 WHERE `service_id` = ?"
	,[servicingID])
	.then(([rows]) => {
	//Query for update the box_servicing table' id for
	transfer_handling (set status boolean to 1)

F.O.C 4.15	**function checkCurrentTransfer(staffID){**

385 - 388	return dbConnection.execute("SELECT * FROM `transfer_allocation` WHERE `staff_fkid`= ? order by `acquire_id` DESC LIMIT 1"
	,[staffID])
	.then(([rows]) => {
	//query to check current transfer status on latest id for staff from transfer_allocation table

389 - 393	if (rows.length > 0){
	console.log('Current transfer(`box_servicing`) >'+ rows[0].acquire_id);
	retrieveCurretTaskWithQR(staffID, rows[0].acquire_id)
	return rows[0];
	}
	//if row found call retrieveCurretTaskWithQR() function.

F.O.C 4.16	**function checkNewTransferQuest(staffID){**
406 - 411	dbConnection.execute("SELECT * FROM `box_servicing` WHERE `service_type` = 'T_S' AND `transfer_handling` = 0 AND `transfer_complete` = 0 LIMIT 1" // discard check on condition of `expire_datetime` >

	CURRENT_TIMESTAMP()
	//let the service able to continue even expired
	)//,[staffID]
	.then(([rows]) => {
	if (rows.length > 0){
	//Find out the latest Transfer need from box_servicing table

414 - 418	addRowTotransfrAllocation(staffID, rows[0].service_id);
	update_staffQR(rows[0].localbox_fkid); // update qr string for local box
	update_staffQR(rows[0].targetbox_fkid); // update qr string for target box
	setTransfer_handling(rows[0].service_id);//set transfer handling status
	checkCurrentTransfer(staffID);//retrieve info to show detail on page
	// call functions if Transfer service found in box_servicing table

F.O.C 4.17 function retrieveBoxQRPassforStaff(boxID){	

436 - 438	dbConnection.execute("SELECT `staffpass_qr` FROM `boxes` WHERE `box_id`=?"
	,[boxID])
	.then(([rows]) => {
	//retrieve the staffpass_qr information

F.O.C 4.18 function setTransferHandlingToZero(boxServicingID){	

455 - 457	dbConnection.execute("UPDATE `box_servicing` SET `transfer_handling`=0 WHERE `service_id`=? AND `transfer_handling` =1"
	,[boxServicingID])
	.then(([rows]) => {
	//function to set transfer handling back to 0 from table box_servicing
	//(means transfer not handling by anyone currently)

F.O.C 4.19 function cancelTheCurrentAquiredTask(transferID_){	

471 - 473	dbConnection.execute("UPDATE `transfer_allocation` SET `transfer_status`='Cancelled',`cancelled_time`=CURRENT_TIMESTAMP WHERE `acquire_id`= ?"
	,[transferID_])
	.then(([rows]) => {
	//query to cancel the current task
	(allow cancel only before transfer_handling was made means if still not scanned with first' box Qr code)

F.O.C 4.20 app.use('/', (req,res) => {	
484	res.status(404).send(' 404 Page Not Found! '); //response 404 if none of the requests is available.

3.5 TEST CASES AND RESULTS

There are a total of 19 test cases and results. Each of these test cases is to ensure that the three major parts of the smart cabinet system, which was shown in the Figure 3.2 use case diagram are workable. Black box testing is to determine the appropriate behavior of the system, whereas white box testing is to test any unexpected behavior lead by unacceptable input cause by the internal structure of the code. All of the test plans will be summaries in chapter 4.

3.5.1 Form in clientware (client)

The test case is for testing the form validation for the client ware for preventing any illegal format of data from being sent to the server. The execution step is to test the text field on the email and phone field and click the submit button. As the test result shows in Appendix C 1, if a customer does not fill in the proper format of email or phone, there will be a pop-up alert message to notify the customer to change the info in the text field. It works after the submit button is clicked within the form page.

3.5.2 Unlocking with QR (client)

The QR code reading testing is for either customer and staff able to use their received code to unlock those cabinet boxes. Which the provided QR code contains JSON format {sid:xx, cqr} or {tid:xx, bqr:xx} or {tid:xx, aqr:xx}, if the QR code does not consist of any pair of those JSON codes after 40 seconds it will pop message "No QR code or Face detected, times up." Condition Else if the format is correct but the provided password or id was incorrect it will show unlock fail unless the customer or staff had provided the correct pass and it will show "box id unlocked successfully".

3.5.3 Facial recognition (client)

Facial recognition is to test the accuracy of when customers can unlock their box using facial recognition. If there were invalid person's face showed on the LBPH recognition, the Python could not yield the valid label value back to node.js. Therefore, the recognition will run up to 2 minutes, and when the time reaches and still not having the legit face presented the system will show a message of (No QR code or Face detected, times up.) The testing method is normally using the face photo that gets from the internet on testing the recognition feature which it had successfully rejected unknown faces. The test also included using legit face but is on the phone screen, since the recognition has applied the liveness detection, even the image provided is legitimate but it does not meet the requirement of blinking consequently it will not provide the label value from the recognition. As only when the real person shows up in front of the camera and has the action of eye blink, the system grants access to the box id to unlock when the label output is correct.

3.5.4 Deliver staff portal

There are two test cases done for the staff portal which is the form for login and register. The execution step is just testing the text field on the given form on the staff portal shown in appendix D Figure 7.1. For the staff registering form as long as the email contains a .com style format, the name and password can be accepted in the majority of ASCII code except for those escaping quotes or emoji. When the email format is correct, the name and password do not contain non-general characters it will show a message of (your account has been created successfully, Now you can Login) else it will show (Invalid email address!) and so on. The next test case is about the staff login, the form does not accept non-Unicode characters and when the email and password are correct then it will be redirected to the home page otherwise it will show a message of “invalid email address or password”.

3.5.5 API testing (backend)

There are a total of 14 test cases for the API endpoint testing. All of the requests are required to provide a valid JWT token. And most of the endpoint will accepting the specific objects in JSON format, if server endpoint is exist but there are mismatch JSON object, the endpoint will return message looks like { "validation": "no", "msg": "record not found" } or example { "tid": "tid is required"} . Most of the endpoint is accepting either information about box id, QR string, and target cabinet id, except those endpoints in test case on test 01 nihao, Api test 08 list_of_location, test 09 list_all_localCabinetboxes, test 10 list_of_localCabinetboxes, test 12 list_of_facefile_in_allocatedBox those endpoints do not require in accepting other values with just need of the true token. Only the payment endpoint has more value to check as if any of the JSON objects provided is incorrect or insufficient it will show {objectx: is required}.

CHAPTER 4
PRODUCT EVALUATION

4.1 Product Evaluation

In general, the product had fulfilled the basic requirements established in the experiment despite having certain imperfections. This section will analyze the requirements by their eligibility and weakness.

4.1.1 Form in clientware (client)

The form in the client ware only consists of two text fields for customers to provide their contact. Since the phone number is only strictly preset as accepting exact 10 digits, and the email field is constantly checking by JQuery the action when the user typing on it. If the two fields did not meet the pattern the submit button will not able to get pressed, thus it will force the customer to type in the correct format before submitting the form.

4.1.2 Unlocking with QR (client)

The unlocking with QR is the main feature for the program, the testing result is consistent which requires the customer to receive the QR image from email, and staff users are compulsory to acquire tasks in the staff portal. The QR reader is using the OpenCV “cv2.QRCodeDetector()”. It will decode the message contained in any image that is presented in front of the camera during scanning, the decoded message again returns to Node.js for checking if it contains the JSON object it will continue sending the message to the backend for checking. Whenever a prerequisite JSON object is found, the decoded code will compare with the server record and when the QR hash code is matched, the server will return a JSON object as the client ware will respond to the server response to unlock the box id. The code format for customers will contain “cqr” customer QR and “sid” service id; the code format for staff users will contain “bqr” before transfer QR or “aqr” after transfer QR and a “tid” transfer id. The QR contains a random 20 character long alphanumeric character which around 20^36 possibility of combinations, thus making Bruteforce with generating image and do the scanning attack become impractical.

4.1.3 Facial recognition (client)

The facial recognition tests are using OpenCV LBPH recognizer, the accuracy will affect as training files are depending on the quality of the facial image during the facial detection. The current system has only applied Haar Cascade face detection method which is a prevailing and easy way in finding faces just that Haar Cascade will have disadvantage that might get a higher chance in collecting false positive faces during the image acquisition. The recognition process will require running the CNN for eye detection and the LBPH for recognizing valid customers. If the user does not take enrollment before on the cabinet client system, the LBPH will not return any label value back to Node.js. Since there are a lot of failure examples in the past about the news of facial recognition getting bypassed by image attack, the CNN network here takes the important roles in identifying whether a person's face is alive and has capable action on blinking his/her eye during the recognition. The test case is successfully rejecting faces that do not have the eye blinking movement. If the LBPH recognition had validated and CNN liveness net was detected the user making the blink, the Python will return a label value to Node.js client ware and going to pass to the server to check if the label value exists on record. If the label value exists the server will return JSON to the client and the client will show the message of box id unlocked successfully.

4.1.4 Deliver staff portal

The staff portal has a register and login form. For system demonstration purposes the registration is allowing a kind of user to make a register as long the email format is having .com or .net pattern, and name and password in Unicode standard then the user can create a new account. The staff login only allows users that have registered accounts to access the portal page. If the user provided a non-Unicode character and submitted the form, the response will be like “Illegal mix of collations” else if the password/email is incorrect it shows ”email/password is incorrect” during login.

4.1.5 API testing (backend)

The API testing is for ensuring the client ware can receive a usual response from the server for each different request of the endpoint. The testing is going through by using the Postman rest debugging tool, fortunately, during the debugging times, it has discovered insufficient JSON object checking bugs for some endpoints. After realizing such bugs, on those endpoints code has added extra validation to ensure the given type and object is in the correct format. The previous version will crash due to not being able to handle the non JSON format, after adding code shows on code A) Backend API in index.js line 11- 18. The server turns out to be able to return 400 bad request codes for all bad JSON formats that are unable to keep and process. Moreover, the majority of endpoints code now contained the checking for data format along with imperative type. The latest endpoints had been updated to reject unwanted format and respond to the suitable data when the passing request counterpart is stipulated.

4.2 EVALUATION – EVALUATION OF THE PROJECT PROCESS

Although the project does not require a lot of complex equations for calculating, but had a lot of logical workflows that needed to be determined. Starting from the prototyping stage of the project, there was a minor dilemma by choosing the UI framework as before, experiencing the TKinter framework, and finding out the limitations and complexity on building more pages and harder manageable in separating content and the API calling code. There will need to be a concern on the performance of threading for adding the camera frame inside TKinter view, there another reason is the design of Tkinter view only provided lesser options with just buttons and some widgets. If the page contains dynamic content such as each time the number of buttons on a page will require change there will be more technical problems to solve. After listening to the supervisor's wise suggestions, the project had decided on some change of UI from TKinter into CSS HTML using Node.js.

During the time of researching the MVC structure for Node.js, to catch up on the progress, and since the staff portal is one of the requirements, in beginning taking some more time to understand the practical usage of Node.js. By following some tutorials provided by developers, it makes more clarity on knowing the NPM command to install the required package before building the first Node.js program. After testing with developer code, there will be more pictures and plans to construct the basic connection to the MYSQL database with Node.js using the ‘mysql2’ package. Almost week 2 of development, the staff portal was constructed able to log in and register to the home page. Although there will be some differences from the staff portal in MVC compare with the backend structure, the callback function and some of the patterns will be the same. Just that the backend code will not providing any views content and it requires separating the endpoints into routes files which purpose in authenticating the client calls. The staff portal is eventually able to retrieve information and the QR code by using the ‘qrcode.js’ frontend library in converting the JSON content into a QR image.

Hereby until week 3rd, the staff portal part was 90% percent completed, it's time to shift the development and concentrate on the backend API which is used for clients in

handling all the requests. When talking about authentication, since the client ware does not behave like the browser connecting a server using a conventional session value. Which will require the client to make a JWT token from the server when the server received the correct password and username. By using the username and password will not directly call the endpoint, which when only the server has issued the JWT token back to the client, the client only has the permission in calling the endpoints. In the current stage, there will be more API testing, it will require retrieving some data thus it needs to be adding some more dummy values like one of the cabinets will contain numbers of the box id. During week 4, there are some problems in using the Sequelize ORM as it works just like the shortcut way in querying using object style. As the ORM doesn't behave like a conventional way of querying in the fashion of joining tables. The Sequelize ORM will try to retrieve data that have relation to more than 2 tables, and even if when it does not require certainly columns fetching back to result. The Sequelize will still pull all columns out to results which is unnecessary. Therefore, I had changed the plan not to using Sequelize which tests from developer examples, and used back the raw query.

During week 5th the backend API has accomplished more than 60%, and in this stage will be going to build the client ware interface. By following the sketch design shown in appendix D Figure 8, start developing from the main page, box selecting page, form, and unlocking page using express ejs layout template. Nothing special, with ejs all views files which equally to .ejs files will store inside the views folder. The express engine will render views or responses whenever the user clicks a link and triggers the localhost (router.get('/endpoint), the syntax res.render('index') will return index.ejs page for example. And the syntax router.post('/xx') is for accepting forms that contain values. As in the functional requirement, using QR code scanning is one of the main features, because of the lack of experience of using Node.js for machine learning purposes, here all image processing modules will be needed in using Python language. With the intention of letting Python code able to get executed by Node.js client ware which making the command prompt console not noticeable by a customer, here unprohibited will require another package called ‘node-run-cmd’, it is a package which allows the Node.js to execute any command with the child process. The qr.py is for reading QR code using OpenCV. By using the syntax of “cv2.QRCodeDetector” and “detector.detectAndDecode(img)”, now the system has the feature of reading QR code.

Until week 6th, the client interface is considered done, and the other important step is to make the client ware able to call the backend API. The ‘requestify’ package is the first package that tests on the call to backend from client ware. The client ware username password is stored in the .env file when the client ware code is being interpreted by the system. The ‘dotenv’ package will read the credential inside .env file, the value will be passed to the request setting and waiting for the server response and JWT token. And that token will continue reuse until before reaching an expiration of 23 hours, the client ware has used Interval syntax to renew the token for calling the functions inside the callApi.js file. The callApi.js contained all request settings for sending data to the server and listening to the response. Meanwhile making all requests in callApi.js for the client, there are some technical issues which are unknown for Requestify unable to

make POST request but only able to make GET request consequently found another package ‘request’ which can make POST request and solved the problem.

Until week 7th the client ware can view the list of locations, list of boxes, reserving box id and even submitting forms to the server. By listening to the proposition from Dr.Khoo project supervisor, the form had eventually added a third-party payment API STRIPE in simulation the process of having actual payment so that the system would seem close to the production standard. From the required use case, as after the customer goes through the payment, the customer may get a QR code used to unlock their box. The backend had applied the ‘nodemailer’ package as when payment was made, the ‘nodemailer’ would send a message to the customer email address which contained QR code and allocated cabinet box info. The progress still right on track, forthwith stage is adding the facial recognition using LBPH. The LBPH recognizer will be required to add the index of the label when a new person making new enrollment thus will store the index value using SQLite for making sure each index is unique. The SQLite also uses when a new dataset being collected the filename will have a distinctive index number for each person. Still here will continue using the ‘node-run-cmd’ to call out the Python code for facial recognition and facial enrolling procedure. Only the local storing purpose contains such features for unlocking the box using facial recognition.

Until week 8th the backend is finally accomplished, just left the liveness detection for the face recognition. Before adding the liveness eye blink detection, there are problems with compatible issues for TensorFlow which are not supporting on the laptop that using Intel Pentium CPU. The problems occur during installed TensorFlow from the official source and test to run the basic Keras it just keeps showing the AVX not supporting. Luckily found another source to manually install the old package for TensorFlow able to work with the older chipset. The liveness net that implements is gathered from Van Eetveldt's article which is not self-made. The liveness CNN that utilizes is based on LeNet-5 architecture, which only consists of two convolution layers, two average pooling layers, one flattening layer, and three dense layers. And the network has been trained by the Closed Eyes In The Wild (CEW) dataset, which contains 4800 images with 24*24 sizes of open and close eyes. The trained eye detection model has approximately 94% accuracy in identifying whether the eye is closed or open. The recognition is just loading the pre-trained model.h5 weight from eye_net.py from the coding section def load_model() and combining the LBPH recognizer code for returning label value only if the CNN predictor has detected action of an eye blink.

Lastly will be reaching the module testing of each of the required functions. According to the SDLC cycle, the testing is the most critical part, although it feels cumbersome and tedious but at least able to disclose those weaknesses on the system. Thanks to the guideline that keeps calling attention to about taking seriously each input, the previous version of the system honestly contained not few errors which were currently fixed by adding back that validation particularly types checking and catching for client ware, backend API, and staff portal.

CHAPTER 5 Conclusion

The final product of this project demonstrated the full procedure of a smart cabinet system, which will be required to run the backend API code (on the server), staff portal (on the server), and client ware on the remote computer. The purpose is to represent the overall cycle of the customer to book their locker and select the type of service from the system. Furthermore, the approach used in this project does not require expensive equipment; a low-budget camera with 720p resolution could be used to capture the image that will be processed by this system without affecting the reading of QR code and having face recognition process.

With the combination of the main requirements of allocating cabinet boxes, unlock box id with QR with OpenCV, unlock box id with CNN liveness detection and LBPH facial recognition, and database workflow the product is very casual for carrying in demonstration the application for smart cabinets. In conceptual contrast, customers could use the system to store their items and decide to deliver items for business or other purposes, by linking the role of the delivering staff to handle the task of sending parcels to the customer selected target cabinet.

5.1 Future Recommendation

There are a few improvements that could be made to overcome the weakness of the system. The first weakness that can be improved is the use of Dlib Histogram of Oriented Gradient (HOG) facial detection to reduce the false positive face being on preventing non-face image get collected. The design imperfections of client ware will get good-looking without require using a browser to present the interface which is independent UI by implementing the Electron.js

Due to the insufficient time and limit, the current client ware of the cabinet system has not integrated any control for the hardware GPIO for physically commanding on relay or servo. As there is research on Socket.io, which makes the physical locker box sending the occupancy state to the backend with sensors for being able to track the cabinet box in real-time.

References

Angelin Yeoh. 2018.Not home? Get your parcels delivered to a locker or cafe near you . Available at: <https://www.thestar.com.my/tech/tech-news/2018/10/04/check-out-these-local-parcel-delivery-services> [Accessed 12 April 2021].

Abdelmgeid. 2019. An Accurate System for Face Detection and Recognition. [online] Available at: <https://www.researchgate.net/publication/334770252_An_Accurate_System_for_Face_Detection_and_Recognition> [Accessed 11 April 2021].

Apex-insight.com. 2018. China Parcels Market Market Insight Report 2018. [online] Available at: <https://apex-insight.com/wp-content/uploads/2019/10/Apex-Insight-Sample-Report.pdf> [Accessed 11 April 2021].

Bonn. 2014. DHL Paketkasten now available nationwide. [online] Available at: <https://www.dhl.com/en/press/releases/releases_2014/group/dhl_paketkasten.html#.YHMbXegzbIV> [Accessed 11 April 2021].

Cui Xu. 2017. (PDF) Facial Expression Recognition Based on TensorFlow Platform. [online] Available at: <https://www.researchgate.net/publication/319487998_Facial_Expression_Recognition_Based_on_TensorFlow_Platform> [Accessed 11 April 2021].

Chris Dawson. 2019. Danish parcel lockers pilot with PostNord and SwipBox. [online] Available at: <https://tamebay.com/2019/04/postnord-swipbox-parcel-box-pilot.html> [Accessed 12 April 2021].

Cathy Macharis. 2017. Ecological and economic impact of automated parcel lockers vs home delivery. [online] Available at: <https://www.researchgate.net/publication/317847539_Ecological_and_economic_impact_of_automated_parcel_lockers_vs_home_delivery> [Accessed 11 April 2021].

Campbell. 2020. Chinese Company Cainiao Could Revolutionize Global Logistics. [online] Available at: <https://time.com/5914173/cainiao-logisitics-alibaba-china-trade/> [Accessed 11 April 2021].

Chinaplus.cri.cn. 2019. Cainiao smart lockers now all support facial recognition. [online] Available at: <http://chinaplus.cri.cn/news/business/12/20190314/261854.html> [Accessed 11 April 2021].

CEP-Research. 2020. DHL Parcel opens 5,000th Packstation locker in Germany. [online] Available at: <https://www.cep-research.com/news/dhl-parcel-opens-5000th-packstation-locker-in-germany> [Accessed 11 April 2021].

DeBourgh. 2019. The Difference Between Smart Lockers and Traditional Ones and Why it Matters to Your Security - DeBourgh All-American Lockers. [online] Available at: <https://www.debourgh.com/blog/the-difference-between-smart-lockers-and-traditional-ones-and-why-it-matters-to-your-security/> [Accessed 11 April 2021].

dpd.com. 2020. ParcelLock Paketkasten: Tipps & Tricks | Support » DPD. [online] Available at: <https://www.dpd.com/de/en/support/tipps-und-tricks/parcellock-paketkasten/> [Accessed 11 April 2021].

Fbicgroup.com. 2018. China Retail & E-commerce Quarterly. [online] Available at: <https://www.fbicgroup.com/sites/default/files/CREQ_06.pdf> [Accessed 11 April 2021].

Frank Appel. 2019. DHL Annual Reports 2019. [online] Available at: <https://www.annualreports.co.uk/HostedData/AnnualReports/PDF/OTC_DPSGY_2019.pdf> [Accessed 11 April 2021].

Faugere. 2018. Smart Locker Based Access Hub Network Capacity Deployment in Hyperconnected Parcel Logistics. [online] Available at: <https://www.researchgate.net/publication/326059448_Smart_Locker_Based_Access_Hub_Network_Capacity_Deployment_in_Hyperconnected_Parcel_Logistics> [Accessed 12 April 2021].

Hương Trần. 2020. (PDF) SMART LOCKER -A SUSTAINABLE URBAN LAST-MILE DELIVERY SOLUTION: BENEFITS AND CHALLENGES IN IMPLEMENTING IN VIETNAM. [online] Available at: <https://www.researchgate.net/publication/343127138_SMART_LOCKER_-A_SUSTAINABLE_URBAN_LAST-MILE_DELIVERY_SOLUTION_BENEFITS_AND_CHALLENGES_IN_IMPLEMENTING_IN_VIETNAM> [Accessed 11 April 2021].

Hwa Chang. 2014. An introduction to using QR codes in scholarly journals. [online] Available at: <https://www.researchgate.net/publication/271098121_An_introduction_to_using_QR_codes_in_scholarly_journals> [Accessed 11 April 2021].

Joyent, 2016, Node.js, "Home page of Node.js," [Online].Available:https://nodejs.org/en/. [Accessed 01 May 2021].

João Lopes. 2020. Bloq.it : What is a smart locker?. [online] Available at: <https://bloq.it/what-is-smart-locker/> [Accessed 11 April 2021].

Jon Glasco. 2021. Last Mile Delivery Solutions in Smart Cities and Communities. [online] Available at: <https://hub.beesmart.city/en/solutions/smart-mobility/last-mile-delivery-solutions-in-smart-cities> [Accessed 11 April 2021].

James Wang. 2017. Understanding the diversity of final delivery solutions for online retailing: A case of Shenzhen, China. [online] Available at: <https://www.researchgate.net/publication/317423988_Understanding_the_diversity_of_final_delivery_solutions_for_online_retailing_A_case_of_Shenzhen_China> [Accessed 11 April 2021].

Kushal Nahata, 2019. Smart Lockers: Making last-mile delivery convenient for customers. [online] Kushal Nahata. Available at: <https://www.financialexpress.com/money/smart-lockers-making-last-mile-delivery-convenient-for-customers/1804716/> [Accessed 11 April 2021].

Keiron Teilo O'Shea. 2015. An Introduction to Convolutional Neural Networks. [online] Available at: <https://www.researchgate.net/publication/285164623_An_Introduction_to_Convolutional_Neural_Networks> [Accessed 11 April 2021].

Locker & Lock®. 2021. Smart Locker Cabinet - Locker & Lock®. [online] Available at: <https://lockernlock.com/solution/smart-locker-cabinet/> [Accessed 11 April 2021].

LIN Han. 2017. Research on Optimization of Intelligent Express Locker: In the Case of the Intelligent Express Locker in S University. [online] Available at: <https://core.ac.uk/download/pdf/236302652.pdf> [Accessed 11 April 2021].

Last Mile Prophets, 2020. Deutsche Post DHL's Packstation network - is 12,000 enough?. [video] Available at:https://www.youtube.com/watch?v=Yoxjy_5Qp9U&ab_channel=LastMileProphets: [Accessed 11 April 2021]

Meiryani. 2019. Database Management System. [online] Available at: <http://www.ijstr.org/final-print/june2019/Database-Management-System.pdf> [Accessed 11 April 2021].

Macnab. 2017. URBAN FREIGHT DISTRIBUTION Summary of Urban Logistics Trends and EU Programmes in Place. [online] Available at: <https://www.europeanfreightleaders.eu/wp-content/uploads/2018/02/Summary-of-Urban-Logistics-Trends-and-EU-Programmes-in-Place-May-2017-FL.pdf> [Accessed 12 April 2021].

Matthias Heutger. 2015. INTERNET OF THINGS IN LOGISTICS. [online] Available at: <https://www.dhl.com/content/dam/Local_Images/g0/New_aboutus/innovation/DHLTrendReport_Internet_of_things.pdf> [Accessed 11 April 2021].

Prithvi Dev. 2020. Important Libraries of OpenCV. [online] Available at: <https://medium.com/javarevisited/important-libraries-of-opencv-56b14705bf0e> [Accessed 11 April 2021].

Paul Chapman. 2017. UK Consumer and Small Business Parcels Services (C2X): Market Insight Report 2017. [online] Available at: <https://uk.linkedin.com/in/paulchapman27> [Accessed 11 April 2021].

Paul Krill. 2021. Windows goes Node for Microsoft's Internet of things. [online] Available at: <https://www.infoworld.com/article/2924297/windows-10-goes-node-js-in-microsofts-internet-of-things-plan.html> [Accessed 12 April 2021].

Post & Parcel. 2017. Alibaba's Cainiao teaming up with automotive manufacturers for "smarter, greener delivery fleets" | Post & Parcel. [online] Available at: <https://postandparcel.info/80099/news/parcel/alibabas-cainiao-teaming-up-with-automotive-manufacturers-for-smarter-greener-delivery-fleets/> [Accessed 11 April 2021].

postandparcel.info. 2016. MILLION-SHIPMENT MILESTONE FOR DHL SMARTSENSOR. [online] Available at: <https://postandparcel.info/75187/news/million-shipment-milestone-for-dhl-smartsensor/> [Accessed 11 April 2021].

reichelt.com. n.d. Never at home when your parcel arrives?. [online] Available at: <https://www.reichelt.com/Infocenter0/12/index.html?ACTION=12&PAGE=53> [Accessed 12 April 2021].

Roy Thomas. 2000. Architectural Styles and the Design of Network-based Software Architectures. [online] Available at: <https://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm> [Accessed 11 April 2021].

Reynolds. 2019. Why is Node.js for SPAs? - Quora. [online] Available at: <https://www.quora.com/Why-is-Node-js-for-SPAs> [Accessed 11 April 2021].

Rita Liao. 2020. TechCrunch is now a part of Verizon Media. [online] Available at: <https://techcrunch.com/2020/11/11/china-3-billion-parcels-post-covid-singles-day/> [Accessed 11 April 2021].

Seldo. 2018. npm Blog Archive: This year in JavaScript: 2018 in review and npm’s predictions for 2019. [online] Available at: <https://blog.npmjs.org/post/180868064080/this-year-in-javascript-2018-in-review-and-npms> [Accessed 11 April 2021].

Sebastian Peyrott. 2017. A Brief History of JavaScript. [online] Available at: <https://auth0.com/blog/a-brief-history-of-javascript/> [Accessed 11 April 2021].

technical-tips.com. 2021. DHL Goldcard for pack station lost - you should do. [online] Available at: <https://technical-tips.com/blog/internet/map-for-dhlpackstation-lost--21507> [Accessed 11 April 2021].

Tuesta. 2020. Analysis of Characteristics and Efficiency of Smart Locker System. [online] Available at: <http://www.tj.kyushu-u.ac.jp/evergreen/contents/EG2020-7_1_content/pdf/Pages_111-117.pdf> [Accessed 11 April 2021].

Transparencymarketresearch.com. 2021. Automated Smart Locker System Market. [online] Available at: <https://www.transparencymarketresearch.com/automated-smart-locker-system-market.html> [Accessed 11 April 2021].

Tilahun Mihret. 2019. Introduction to Database Systems. [online] Available at: <https://www.researchgate.net/publication/336588372_Lecture_I_-_Introduction_to_Database_Systems> [Accessed 11 April 2021].

Vernon Chua. 2020. How are small brick-and-mortar retailers in Malaysia coping with the e-commerce revolution | e27. [online] Available at: <https://e27.co/how-are-small-brick-and-mortar-retailers-in-malaysia-coping-with-the-ecommerce-revolution-20200305/> [Accessed 12 April 2021].

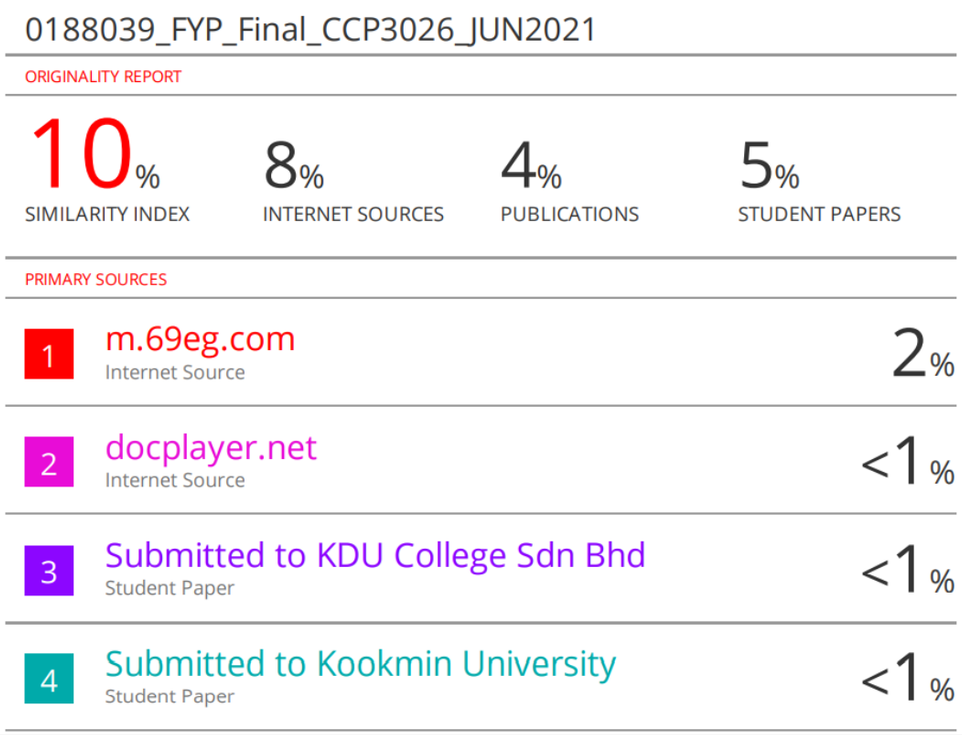
XueMei. 2017. A real-time face recognition system based on the improved LBPH algorithm. [online] Available at: <https://ieeexplore.ieee.org/document/8124508> [Accessed 11 April 2021].

Yaqing. 2017. Big Fight for Big Data-- Beijing Review. [online] Available at: <http://www.bjreview.com/Business/201706/t20170612_800098004.html> [Accessed 11 April 2021].

Zhe Ding. 2013. Evaluating Different Last Mile Logistics Solutions. [online] Available at: <https://www.diva-portal.org/smash/get/diva2:763544/FULLTEXT01.pdf> [Accessed 12 April 2021].

Zell Liew. 2018. Understanding And Using REST APIs — Smashing Magazine. [online] Available at: <https://www.smashingmagazine.com/2018/01/understanding-using-rest-api/> [Accessed 11 April 2021].

TURNITIN



0188039_FYP_Final_CCP3026_JUN2021

ORIGINALITY REPORT

10%	8%	4%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	m.69eg.com Internet Source	2%
2	docplayer.net Internet Source	<1%
3	Submitted to KDU College Sdn Bhd Student Paper	<1%
4	Submitted to Kookmin University Student Paper	<1%

APPENDICES

APPENDIX A – PROJECT PROPOSAL

A. Project title

IoT based smart cabinet system (intelligent lockers)

B. Introduction

Transporting and logistic services are becoming popular and advancing after being involved in internet business service however, the usage of smart cabinets that are provided in public facilities are so limited and rare in Malaysia. Even that it has served to provide service in some places like in Penang Sentral, the cabinet serve in a local area which the cabinet is not bringing extra service like having a backend that can allocating staff to migrate the stock of personal belongings to another place which when the user needs the service of delivering stuff in the right time. Since conventional cabinet storage doesn't consist of the power of sharing resources information over the internet, by introducing the concept of smart IoT cabinet system the aim is to make independent cabinet storage have capability sharing information on a unanimous platform such as all hardware can communicate using the cloud.

Such a cabinet also implies the face recognition algorithm to improve the functionality where users might have in case of unable to recognize the password to unlock the cabinet as the time access their equipment from the locker. Facial recognition is the most trending technology that has been shaping the way that people live in society for security purposes and convenience as well as supporting the inspection of the legitimate user during accessing public services.

The usage of smart cabinets has been introduced in foreign countries to serve businesses to have to deliver service for selling their product without requiring customers to step into the shop in a smart cabinet service covered state. It also helps most of the shopkeepers sustain their business by relying on the system that can allocate staff to pick up the package to the final destination.

C. Background of the project

Since the day having courier service in mankind, the importance of having a secure place to store, drop off or pick up for delivery has dramatically grown after the demands of online purchasing and traveling activities becoming omnipresent in every country. Although that in Malaysia some of the places like air-port and most of the intermodal transit-oriented stations have provided such smart lockers, because of conventionally the only purpose just letting people to storing stuff and it doesn’t have much choice for like deciding the package that put in the box can deliver to another location. If the system has a link with the cloud and can schedule and allocate staff to deliver the materials from the cabinet it will be bringing potential wealth to society.

This paper presents a low-cost and lightweight solution suitable for deployment with a smart cabinet system that integrates face recognition. Although face recognition was optional for users to decide whether he/she needed to set facial recognition for storing purpose, as users can unlock automatically without inconvenient authentication process by using an automatic recognition procedure that uses facial images to recognize users and grant (or deny) him/her access to the cabinet; for delivering service it can just unlock with QR code scanning or passwords since each pick-up and drop off destination that is dynamically chosen. The ideal goal here is to build an integrated simple interface which allows cabinets to be frequently changed for multiple users without sacrificing the overall security.

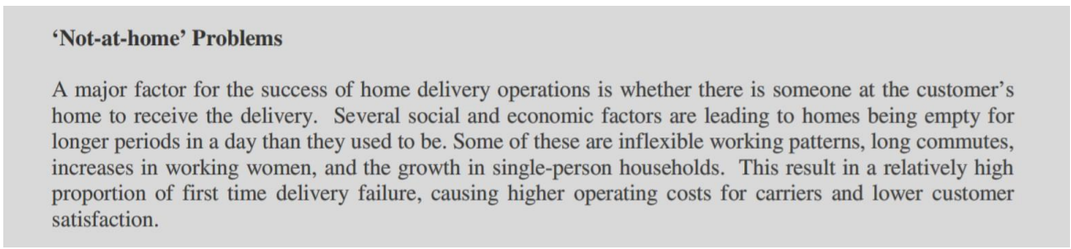


Figure c

Figure c shows a screenshot of the statement about the issue to receive delivery for domestic in the United State. Minyoung Park .Issues in emerging home delivery operations.2004.

D. Problem statement

This project is for providing a novel solution with a new design with giving a rapid in-state delivering serving and instant storage accessing control system for benefit travelers, freelancers, handicaps, householders, online merchants, desk warriors and so on able to store or deliver stuff to places in a covered area. The smart IoT cabinet which is connected and able to report its reserving state is highly convenient that anyone can select service as transfer or just for storing stuff immediately. Because the cost of personal delivery will be double up on approaching and returning, such IoT cabinet service also helps to reduce delivering cost and also decrease on the contagious rate which better enhances hygiene during the intact process of delivering. It will also benefit a work that does not have a fixed address to receive packages by just sending QR code passcode to the courier handler to unlock and secure the success delivered packages in any of the boxes of smart IoT cabinet that user had currently subscribed on their proximate state that covered. Second, such cabinets that are connected and manageable by a central system are potential in sharing the information on tracking the available space and letting businesses have more choices in delivering their stock to customers instantly in a state.

Travelers can enjoy continuous trips after landing at the destination without wasting half of the day voluntarily transferring their luggage to the hotel at first, as they just can reserve the cabinet space then confirm the location that they wish to send. Next is about the reason for proposing the idea was people who are working in the office or any, they may not have time to pick up the packages to address that they wish to receive. Since mailboxes have no function to secure the delivered stuff for the user, seamlessly consumers have to worry about the delivered package might get stolen, damaged or missing hence such smart IoT cabinet systems are the solution that provides users a space that can receive any object they tend to receive. It also makes life easier when consumers are unable to catch up the drop off time and rather than going to the courier servicing center to pick up after missed, during the time before drop off consumers may have the choice to send the code to courier staff and ask for drop off the stock inside the booked cabinet which is near the delivering location by unlocking the cabinet with the QR code.

In public facilities, there are varieties of pedestrians regardless of the intentions of traveling or doing important business. Sometimes they might need to carry a heavy bag with a laptop, helmet or some other equipment, because of the safety of the belongings instead of avoiding getting stolen by putting stuff on vehicles after parking. If such cabinet systems are available, it not just helps people that want to feel comfortable while walking inside the mall or for finding a place to have a meal and enjoy without hassle. They can just decide to rent a case and put in the stuff for a while and feel free to experience the environment.

E. Research Objective

1. Write a Project Initiation Document (PID).
2. Write a proposal on Decision maker analysis IoT cabinet system.
3. Literature review on choice suggestion techniques.
4. List down functional requirements and non-functional requirements in the commentary.

5. Create design diagrams like use caseidiagram, EntityiRelationshipidiagram, and class diagram.
6. Implementation of Decision-maker analysis overall IoT cabinet system.
7. Using a white box and black box for testing.
8. Evaluation on product and project flow.
9. Conclusion and miscellaneous adjustment on products.
10. Write documentation in the final report.

F. Proposed research methodology

Node.js:
The backend of the system will be using node.js as the main core, it works differently from PHP as it will serve asynchronously, which means the program itself does not need extra third source framework or require on customizing the cron job for certain routines. Since each of the hardware components is integrated when the system core is running it has to be actively keeping track of the online status of each of the client base to synchronize the control. The purpose of having node.js as a backend was it will directly access the data and manage the resources within the database. The third reason node.js was chosen was that it can handle multiple simultaneous connections efficiently. Moreover, Node.js has a long history since the internet age began and most of the npm packages are open source shared among developers, it began from front end base technology becoming a great advanced server language today and still under maintenance under Linux Foundation's Collaborative-Projects program. (Netguru.com, 2017)

Express.js:_
A web based applicationIframework-design for Node.js, released as free released and open-source software_under the-MIT License. It-is an integrated toolkit-used for building web applications and-web APIs. It provides intensive features in HTML and HTTP request and response by giving developers seldom defining those URI routing for accessing the resources. (Robbie Jaeger, 2017)

Figure f NODE.JS running socket.io service.

Figure f is a screenshot of the example output from socket.io on node.js under process of listening to the connection and receiving responses from the client.

Jsonwebtoken:
JWT is an openistandard (RFC 7519) that definesia compact and substantive way for information between parties securely transmitting theiJSON object. In such information format can be generally decode by any interface which supports the protocol and is easily validated and trusted because it is protected by digitally'signed. (jwt.io, no date)

Socket.io:_
A mature JavaScriptilibrary for-realtime web applications. Itienables real-time, support fully bi-directional communication between webiclients and servers. (Tutorialspoint.com. 2020)

Python:
In this project, Python will be required to serve as the client based interface to be able to interact with the system seamlessly is to identify users using facial recognition and so on. First of all, Python is a programming language that is very easy to understand and grasp.

It uses simple syntax and can handle big data. The integration of mathematics formulas that apply for solutions is shorter in length codes compared to most other programming languages. The major reason that chose python to serve was that it has a major source of coding in MachineLearning and especially on Raspberry Pi Projects, it plays as a protagonist role in the wide terms of AI. (Tutorialspoint.com, 2020)

Tkinter:
Popular UI framework for python, it can easily design any number of widgets to the main window and apply features after the event Trigger on the widgets. Such as buttons, dropdownlist, combobox, checkbox, progress bar, labels and so on. (Aditya Sharma. 2019)

OpenCV:
A library supported in a variety of the programming language main for processing the real-time computer vision, supporting in the task for filtering image, handling 2d or 3d types of video and image to helps computer able to extract the features from visual and go through any of the processes by image processing algorithm will able detection any viewable objects and distinguish from instances of the predefined classes. (Catherine Huang, 2018)

Dlib:
Aimodern C++ toolkit containing compact machine learning algorithms, used in wideiindustry and academia in a large range of domains includingirobotics, embedded system, mobile application, and large High Performance Computing (HPC) environments. (Dlib.net. 2019)

Tensorflow:
An open-source ML framework for building and computing data flow graphs, allowing for the free creation and training of artificial neural networks regardless levels of complexity. (Guru99.com, 2020)

Keras:
High-level neuralinetwork application programming1interface, or API, written in Python and capable of running on top of TensorFlow. It supports making models, defining layers, or setting up multiple input-output models, handling activation functions and so on for creating customized neural networks in a project. (Github. 2019)

Scikit-learn:
Library that focuses on the general machine learning for the Python programming language. (Jason Brownlee, 2014)

MySQL:
An open-source’relational database management system (RDBMS) for managing the data schema in the ACID concept. (Tutorialspoint.com, 2020)

G i. Integration and analysis of Facial Recognition Algorithms

To integrate facial recognition on the system here will provide some basic analysis for each of the different facial recognition algorithms such as Eigenface, Fisherface, LBPH in the following statement. Here will start from explaining the beginning step of the process that requires facial recognition for the preparatory procedure, face detection will start_on capturing images to further exclude the image background fromithe processing and subjecting exclusively on the region of interest. For the detection step, OpenCV’s implementation of the Viola-Jones algorithm known as Haar Cascade will be used in the project (Paul Viola. 2002).

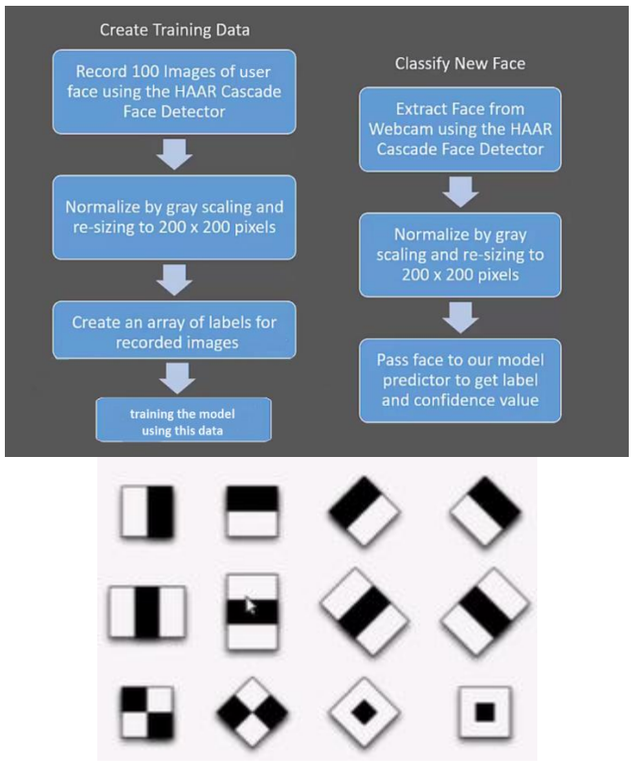


Figure g.1

Figure g.1 shows the normalization process of face recognition with various Haar Cascade patterns.

Haar-likeifeatures and cascade classifiers (Paul Viola. 2002) in Figure g.1 are usually considered as adjacent orthogonal regions at a specific location in a detection-on the division of the image size, afterisums up the pixel intensities in each of the regions and-additional calculate the difference between these-sums. Further finding the instinctive identical parts with haar-wavelets, these-features have been defined as haar-features in the primary phase in face detection. In order to extract the face by classifying and saving good alignments it also discards those pictures that have bad alignments so that it can proceed to the next process of normalization.

G ii. Local Binary Pattern Histogram

LBPH was introduced after 1990 as the main algorithm that is used in most of the project including current fyp will choose as a face recognizer for recognizing the faces for legitimate users. It works by stored in the form of a matrix of pixels. LBP is an operator used to summarize the local special structure of an image. It_is defined_as an ordered set of binary comparisons of pixel intensities between the pixel in the center and adjacent by eight pixels. Fig g.2 is-the threshold function-that is used to check if the-value of it's neighbor is larger than or equivalent to the central value it is set to 1 either or 0. After concate theseithresholded values and get the 8-bit binary number which will further decode into a decimal. This decimal number is known as;the pixel-LBP value and its range is from 0- 255.

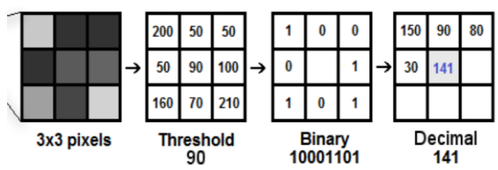


Figure g.2

Figure g.2 shows the thresholding process with '3x3' sampling points pixel after transformations to grayscale. From Kelvin Salton doiPrado.2017. Face Recognition: Understanding LBPH-Algorithm.

$$s(x) = \begin{cases} 1 & \text{if} \quad x \geq 0 \\ 0 & \text{if} \quad x < 0 \end{cases}$$

Figure g.3 logic of threshold function for LBP

After the LBP value histogram has been generated from the region, it works by counting the number of LBP values which have the common in the region. Then it will go through with processing histograms in each of the regions on the image and combine all the histograms and form a single_histogram and this is called an-extracted feature or vector_of the image. By comparing with two histograms and returning the target image which has the closest histogram, the closest match will be calculated by the Euclidean distance formula on Figure g.2 and depending on the setting given by the confidence measurement once the closest match has been found it will output the ID which has high confidence percentage of match.

$$D = \sqrt{\sum_{i=1}^{n} (hist1_i - hist2_i)^2}$$

Figure g.4 formula of Euclidean distance

- Advantages of LBPH

1. Simplicity of computational
2. High discriminative power
3. Invariance to grayscale changes and good performance

- Disadvantages of LBPH

1. Not invariant to rotations
2. Limited on captured certain structural information. Only pixels with consisting difference are used, the information of magnitude in lower peaks gets ignored.
3. The size of the features may increase exponentially depending on the number of the vicinity which leads to expansion on the computational complexity as well as in terms of the higher usage of memory storage space and computation times.

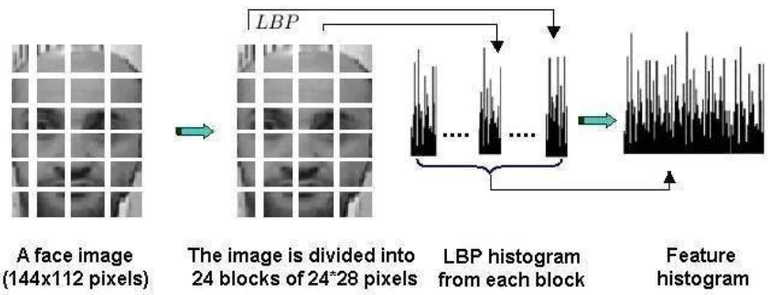


Figure g.5

The Figure g.5 shows the process within LBPH divided (144 x 112 pixels) face image into 24 blocks to calculate each of the density gradients and turn into a full histogram of the subject. Burhanuddin Ahmed. 2018. Image Recognition With SVM And Local Binary Pattern.

G iii. About classifiers use in facial recognition

In the previous section, LBPH models are the chosen method to apply for facial recognition, there are also classifiers commonly used to strengthen the accuracy rate during calculating the features such as K-NN, SVM, and CNN here will give some brief explanation. For the other classifiers like Eigenface-LDA and Fisherface-PCA will be excluded in the last chapter of appendix.

K Nearest Neighbor

KNN_is an algorithm that is an elementary,_supervised_machine_learning algorithm which is dedicated to both regression and classification problems(Shichao Zhang, 2017). Normally it will be using either one from the three types of distance function and they are Euclidean, Manhattan, Minkowski. The classification is carried out by comparing feature vectors of different nearest K points. The main side effect of kNN is about the complexity of searching the nearest neighbors for each sample. KNN works by searching the distances between-instances in the data, selecting the specified number giving hypothesis to determine if the K value is the closest to the related group of points, then it will overseeding on the votes for subjected by the most frequent label (for classification) or averages the labels (for regression).

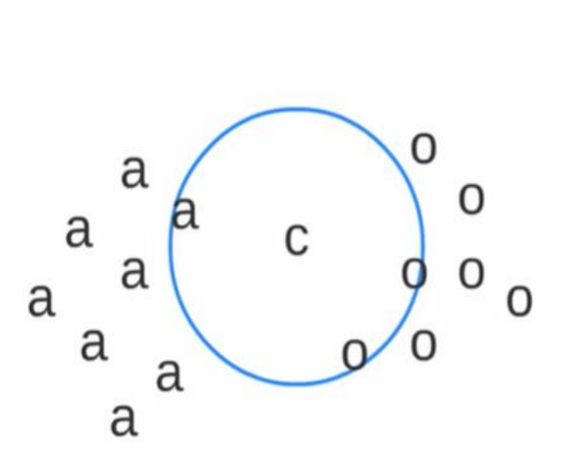


Figure g.11

An example in Figure g.11 is for illustrating the kNN finding position distance from the letter '*c*' to determine closest class by comparing the closest frequency between group '*a*' and group '*o*'.

Support Vector Machine

SVM is a-non-probabilisticilinear classifier, it can support non-linear problems with 'kernel trick'. It is considered as a model that can learn from the past input data and makes a future prediction as to the output. SVM is a supervised learning method that gives aspect at the data and sorts it into one of the two categories. In Figure g.12 it has two dotted lines aside from the red line, those scatter points(star & circle shaped) nearest to the dotted line from both of the classes are known as support_vectors. The distance-in between the dotted lines is called support vectors or the margin. SVM aims to exaggerate the range of the margin. The optimal hyperplane forowhich would be found once the margin is in maximum range after multiple attempts on varieties of angle on leveling (Savan Patel, 2017).

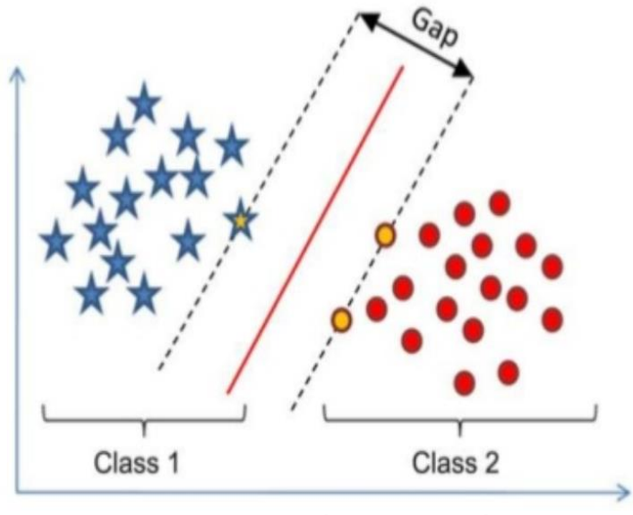


Figure g.12 Example of SVM Classification

Convolution Neural Network

As before understanding the convolution neural network, the basic concept comes from the multilayer perceptrons logically works like human brain synapses with connected nodes, it works by passing the value to another node when they are only activated when reached a threshold value. CNN was introduced by Yann Le Chun in 1994, CNN will consist of tens or hundreds of layers depending on the purpose that each layer will persist on learning to detect different features of an image by recursively adjusting the weight values example shown in Figure g.13. The most important layers in CNN are convolutional_layer, pooling layer, Relu_layer,_and fullyiconnected (Keiron Teilo O'Shea. 2015) layer and each of these layers has their role, here will explain.

- The convolutional layer will normally compute the output ofineurons that are connected to local regions in the input, it will apply with various types of filters to serve for the feature extraction. Each of the filters has their bias value, when a certain input pattern has matched to the filter it will trigger the activation on and there will be only the activated features that can be continued on carried-forward into the next layer.

- Pooling_layer is used for-simplifying the total output to perform nonlinear downsampling, combating any of the overfitting occurrences and shrinking the scales of parameters making the network more compact as when it needs toilearn in the future.

- Fully connected layers are intentionally used to output a vector of K dimensions where.K is the number_of classes that the network manages to predict. Such vector value will include the probabilities about available classes of'any image being classified. E.g assume there are 3 classes able to classify in theifully\connected layer and the outcome of the results could be a closer percentage on at least one of the classes or could be none.

- Relu layer, Rectified Linear Unit is used as a non-linear activation function to avoid the vanishing gradient problem occurring in sigmoids. Easy to say it deals with mapping any negative values and converting them into zero and maintaining accuracy with pure positive values so it allows for the faster and more effective training process.

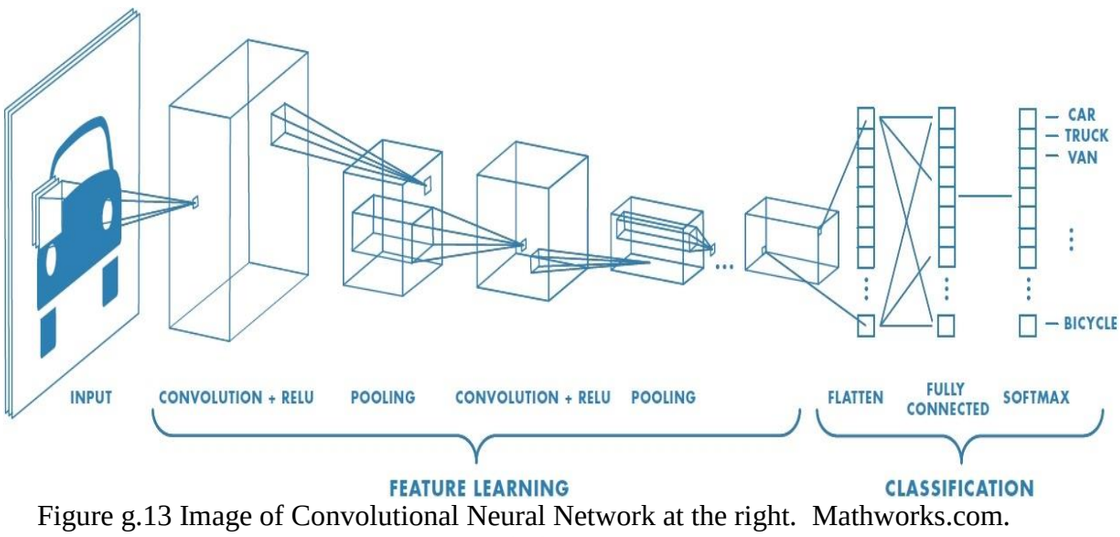


Figure g.13 Image of Convolutional Neural Network at the right. Mathworks.com. 2020.

H. Proposed Work

Throughout my research on this project, I have found out that I need to do some research and learn about the specifications on suitable face recognition algorithms that come with liveness detection techniques such as eye-blink based heuristic approach (Jordan Van Eetveldt, 2019) which can be applied to the smart cabinet system. And also the backend part requires studying the Node.js Rest API to link connections to the cabinet clients.

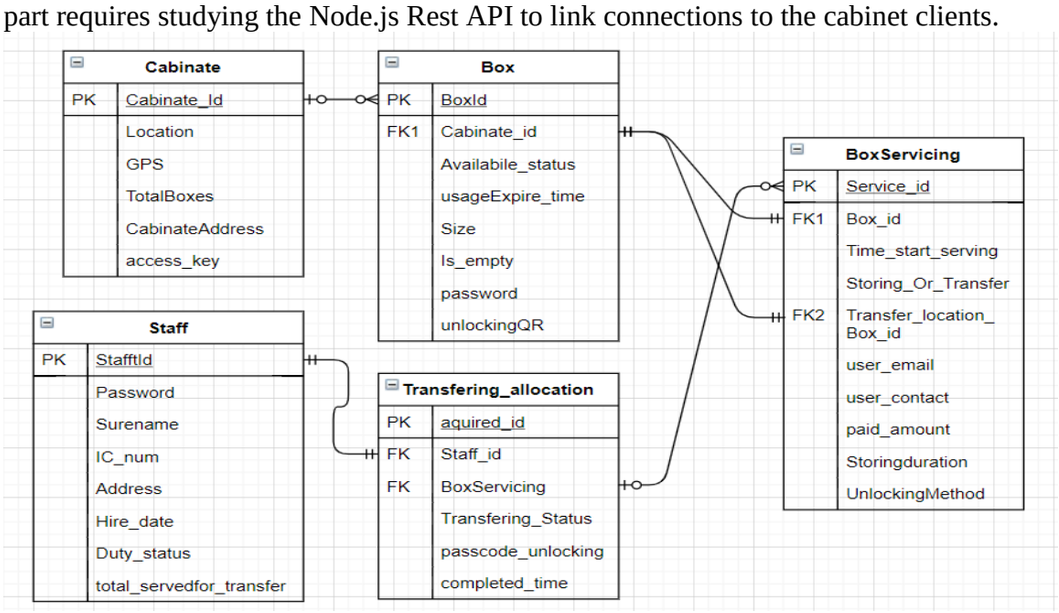


Figure g.1

Figure g.1 shows the prototype ERD diagram for smart IoT cabinet system backend.

Here will give the explanation for the system workflow regard on ERD diagram above Figure g.1 while example users interacting with the smart IoT cabinet system which will develop by python language and Tkinter. At first, users will choose the service on the smart cabinet, the user will decide to pick the vacant box that is not under reserve state. Second steps users will choose a service to transfer purpose or use it as storage.

In scenario 1, assume that a user has selected the service as transferring, when users have chosen the available storage, users will need to provide information like contact and email for being able to receive transfer status and also the QR code for unlocking the cabinet. Next, users will have to pick any of the destinations that they want to transfer to another smart cabinet which is covered in the area.

In scenario 2, if a user has selected the service like storage, users will get informed about whether he/she wants to use their face to unlock the box in the future. If a user decides to use their face to unlock, the system will show a scanning process for telling that user needs to proceed with the facial capture, the scanning process will be implemented with the LBPH algorithm as introduced in the last previous chapter (in G i.). Else if users decide not to unlock with face, they can set their password accordingly. The facial data will keep in the client system until the expiration time of serving has reached and will get deleted automatically.

Machine learning in authentication is a vital step to protect not just only for information but also protecting physical properties. In this context, face biometrics is potentially beneficial for storing purposes as it will be part of the functionality on the smart cabinet system for certain users during the storing period will be more frequent to interact with the system. Face biometrics is natural, intuitive, convenient, and less human-invasive. Unfortunately, there were several research studies that ordinary face recognition algorithms are vulnerable to presentation attacks even using low-tech equipment which is low cost, such as fooling the system with copies of the legitimate users' image (Yu Tian, 2020). Before implementing the facial recognition algorithm to the system here, it has to take into consideration the liveness detection accuracy to add a layer of checking and enhance the security of the system by choosing the optimal liveness detection solution which supports the low-cost hardware like Raspberry Pi.

There are solutions to detect the liveness of the user, the purpose is to detect any action that is to attempt spoofing and reject those fake images once the faces are captured. Here will provide a screenshot of the shot brief for each of the face liveness detection methods that are generally used shown in Figure e.11.

Figure e.11

Figure e.11 shows a list of example solutions for enhancing facial recognition reliability from presentation attack. Saptarshi Chakraborty. An overview of face liveness detection. 2014. For implementing facial liveness detection, a pre trained keras convolutional neural network (CNN introduced in previous chapter) model will be the additional features used to differentiate on fake face attack during the scanning procedure. While the user in scanning the system will detect the eye heuristic of any blinking, the model that was chosen is called LeNet5 introduced by LeCun in 1998 it was a mature multilayer neural network designed for recognizing handwritten and machine-printed character (LeCun, 1998), the model that chosen had pre trained with 4800 images of open closed eye dataset provided by Closed Eyes In The Wild (CEW) through the neural network which consists of 7 layers including three convolutional layers and two pooling layers capable in filtering vector of 128 features. If a user tries to fool the system by using legit user photos, it will refuse the recognition when the neural network detects it was an attack.

After the (selected cabinet services) case mentioned above happened, users will need to provide information like contact and email for being able to receive transfer status(this for only transfer) and also for getting the QR code for unlocking the cabinet. Once the type of service is selected, the system will calculate the amount that the user has to pay, and the user will proceed to checkout. After payment was done by users, they may be able to receive an invoice on their email as well as the QR code for unlocking the cabinet. Those records and data will be saved on the MySQL database which requests and responses of the client base are handled by the Node.js rest API. All requests will be sent through the HTTP method GET & POST, where the client request to the backend will verify by the session JWT token. The token value will store inside the raspberry pi storage, as before gathering the token the first time to connect to the node.js backend requires the match with the password, it is stored in entity name 'access_key' from ERD at Figure g.1 the table named '*Cabinate'*.

Node JSON Web tokens(JWT) are the package responsible for having communication with the python client JWT tokens similar to how the ATM system works, after plugging the access token to an authentication system and getting access to restricted data that belongs to the client. Also, it will require express.js to create a simple staff and admin page for showing the transferring detail and the online status of the client base. Inside the ERD shown in Figure g.1 the table named '*Box'* has an entity named 'Is_empty' is used for checking space occupied status for boxes by using proximity sensors and the value will keep updating for every minute. Regarding the record for how each of those

boxes serves from the cabinet, the table name '*BoxServicing'* is for recording the details of current users at that period that was under usage for the box.

```
>>> import jwt

>>> encoded_jwt = jwt.encode({'some': 'payload'}, 'secret', algorithm='HS256')
>>> encoded_jwt
'eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9.eyJzb211IjoicGF5bG9hZCJ9.4twFt5NiznN84AWoo1d7KO1T_yoc0Z6X

>>> jwt.decode(encoded_jwt, 'secret', algorithms=['HS256'])
{'some': 'payload'}
```

Figure g.2 import JWT library from python.

Figure g.2 is about the example code for python to encode and decode the message via JWT format. From pyjwt.readthedocs.io, by Jos é Padilla 2015. Welcome to PyJWT.

In hypothesis that when the transfer service was made, in order to let the staff know about the detail, there will be a web login page available just for only staff to access so that the staff can login and acquire the next mission to receive the message of a new location to transfer stuff with the smart cabinet system. The detail will be showing the cabinet GPS location of the transfer service ordered box ID, target GPS location of the destination cabinet box and the image of QR code for unlocking. The only control for the staff is press to acquire new missions or cancel the current mission. Those staff accessing information are stored in the table named '*Staff*' ERD at Figure g.1.

About the ERD at Figure g.1 the entity named 'Available_status' in table name '*Box*' are used when the original location of the boxes state will remain the "engage" state until the staff have successfully transferred the stuff to the remote location that user had preset eventually the box at the origin point, soon will be change the available state into "vacant" if the deliver was succeed so that next user can use the resource. For knowing the staff had placed the allocating stuff into the cabinet, inside the cabinet will have a proximity sensor, if the staff had transferred to the target location to the right box, the proximity sensor value will update status to the server(entity 'Is_empty' in Figure g1 ERD table name Box). Inside the ER diagram in Figure g.1 for the table named '*Transfer Allocation'* are used to record the status whether the staff has completed the current task for delivering and within the table it has an attribute named Transfer Status to record the status whether is a success or failure during the process of delivering. If the staff has uncertain reasons they can decide to cancel the current mission, and it can allocate other available staff to acquire the mission to continue transferring again. Once the staff canceled the mission the QR code for the previous staff to unlock the box will become unusable, as the next staff had gathered the mission for the box ID by right only being able to access the box during duty.

The methodology of the SDLC-(System Development Life-Cycle) that chooses on developing this project will be the=Waterfall Module. The Waterfall module is more suited for developing this project since each of the tasks must be completed in a certain stage and every phase must be reviewed and approved by the lecturer.

In conclusion, the completion_of this project will be shown in using a Raspberry pi with connected hardware and installed code for demonstrating the concept of the smart IoT cabinet. This project is appraisal=to be accomplished in December oflthe year-2020.

I. Aim of theiproject

1. This application aims to overcome the complicated in finding local transfer service and for giving potential to businesses to deliver goods anytime instantly, bringing users a trustworthy storage service by connecting each of the cabinets to a system.

2. It also gives users to choose the service as transfer or storing at the same time supporting tracking on available space and keeps safe of goods.

3. The application also integrated facial recognition algorithms and QR code used for unlocking the cabinet during before and after moving stuff to or from the cabinet.

4. Backend systems will allocate staff to deliver packages after transfer orders were made by a user.

J.i Skills

1. Python

Description:
Python is the main programming language used for development on the smart cabinet and since that it plays a huge role in facial recognition. In addition, it needs the support of many Machine Learning libraries and rich resources available for experimenting various supervised, unsupervised learning as well as it can just require minimum lines of syntax to run a CNN convolutional neural network in an easy way.

Resources:
The resources are from Intelligence System course materials although the course did give much detail on developing image recognition instead it still has a similar concept of machine learning. Most of all, the resources can be retrieved from the internet.

2. Intelligent System

Description: The fundamentals of AI and its concept and its algorithm implementation.
Resources: The resources can be retrieved online.

3. Software Engineering

Description: Applying a suitable methodology and knowing its benefits and flaws.
Resources: The resource can be retrieved from software engineering course materials.

4. Database System

Description: Using the queries to retrieve data and insert data from the database.

5. Node.js

Description: Developing the requesting server for the client base to save the data and communicate with the database.
Resources: The resources can be retrieved from online.

6. HCI

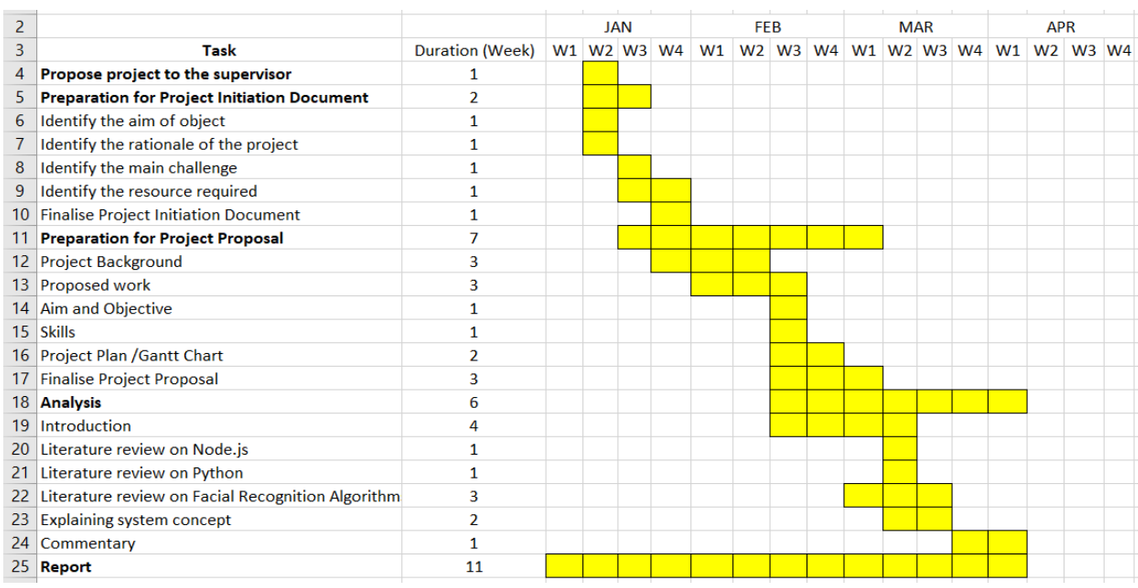
Description: Developing the user interface based on the Human-Computer Interaction theory.
Resources: Referring to interface design it needs theory by understanding human perception.

J. ii Skills table

Skill	Description
Web Application	Diploma Web Application subject - Ms. Shyamala
Artificial Intelligence	CAI3024 - Intelligent Systems - Mr. Ang Sau Loong
Software Engineering	CSE3033 - Software Engineering - Ms. Shyamala
Database System	Diploma database subject - Ms. Shakira
OOP	Object-oriented System Analysis and Design - Ms. Tan Phit Huan
Node.js	Tutorial series learn from Udemy and Youtube.
Python	CAI3024 - Intelligent Systems - Mr. Ang Sau Loong
User Interface Design	Diploma UI subject - Ms. Justtina John

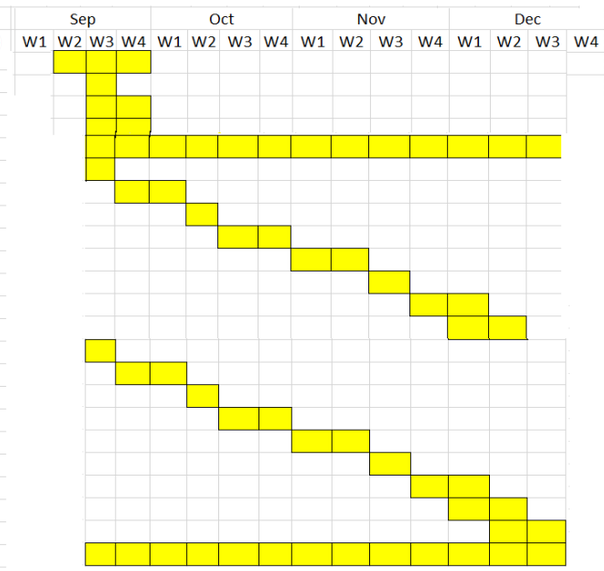
K. GanttCharts

- FYP1



- FYP2

	Task	Duration (Week)
25	**Design**	3
26	Use Case Diagram	1
27	ER Diagram	2
28	Class diagram	2
29	**Implementation**	11
30	Module 1 Building client UI with Tkinter	1
31	Module 2 Implement facial recognition	2
32	Module 3 Building backend functionality	1
33	Module 4 Link client side to the backend	2
34	Module 5 Add allocation staff to transfer feature	2
35	Module 6 Add QR scanning feature	1
36	Module 7 Add liveness detection	2
37	**Testing**	11
38	Testing on module 1	1
39	Testing on module 2	2
40	Testing on module 3	1
41	Testing on module 4	2
42	Testing on module 5	2
43	Testing on module 6	1
44	Testing on module 7	2
45	**Evaluation product and project process**	1
46	**Caonslusion and future work**	1
47	**Report**	11



L. References

Aditya Sharma. 2019. Introduction to GUI With Tkinter in Python. [online] Available at: https://www.datacamp.com/community/tutorials/gui-tkinter-python [Accessed 8 Mar. 2020].

Arxiv.org. 2014. AN OVERVIEW OF FACE LIVENESS DETECTION. [online] Available at: https://arxiv.org/ftp/arxiv/papers/1405/1405.2227.pdf [Accessed 8 Mar. 2020].

Burhanuddin Ahmed. 2018. Image Recognition With SVM And Local Binary Pattern. [online] Available at: <https://medium.com/@burhanahmeed/image-recognition-with-svm-and-local-binary-pattern-289cc19ba7fe> [Accessed 11 March 2020].

baseapp.com. (no date.). A Comprehensive Guide to Facial Recognition Algorithms - Part 1. [online] Available at: https://www.baseapp.com/computer-vision/a-comprehensive-guide-to-facial-recognition-algorithms/ [Accessed 8 Mar. 2020].

builtin.com. 2020. A Step by Step Explanation of Principal Component Analysis. [online] Available at: https://builtin.com/data-science/step-step-explanation-principal-component-analysis [Accessed 8 Mar. 2020].

Catherine Huang. 2018. An introduction to OpenCV [online] Available at: <https://medium.com/@cath.h858/an-introduction-to-opencv-ccc25fd6f310 > [Accessed 10 March 2020].

Dong-Chen He. 1990. Texture Unit, Texture Spectrum, And Texture Analysis. [online] Available at: <https://www.academia.edu/26128123/Texture_Unit_Texture_Spectrum_And_Texture_Analysis> [Accessed 10 March 2020].

Dlib.net. 2019. Dlib C++ Library. [online] Available at: <http://dlib.net/> [Accessed 6 March 2020].
Eprints.utar.edu.my. 2018. Anti-theft Security System Using Face Recognition. [online] Available at: http://eprints.utar.edu.my/3058/1/fyp_CT_2018_CGY_-_1305524.pdf [Accessed 8 Mar. 2020].

Guru99.com. 2020. What is TensorFlow? Introduction, Architecture & Example Available at: <https://www.guru99.com/what-is-tensorflow.html > [Accessed 6 March 2020].

GitHub. 2019. Keras-Team/Keras. [online] Available at: <https://github.com/keras-team/keras> [Accessed 6 March 2020].

Josh Starmer StatQuest: PCA main ideas in only 5 minutes!. (2017). [video] Available at <https://www.youtube.com/watch?v=HMOI_lkzW08: StatQuest with Josh Starmer.> [Accessed 9 March 2020].

Jordan Van Eetveldt, Medium. 2019. Real-Time Face Liveness Detection With Python, Keras And Opencv. [online] Available at: <https://towardsdatascience.com/real-time-face-liveness-detection-with-python-keras-and-opencv-c35dc70dafd3> [Accessed 10 March 2020].

Jason Brownlee. 2014. > A Gentle Introduction to Scikit-Learn: A Python Machine Learning Library. [online] Available at: <https://machinelearningmastery.com/a-gentle-introduction-to-scikit-learn-a-python-machine-learning-library/> [Accessed 10 March 2020].

jwt.io. (no date). > Introduction to JSON Web Tokens. [online] Available at: <https://jwt.io/introduction/> [Accessed 10 March 2020].

Keiron Teilo O'Shea. 2015. An Introduction To Convolutional Neural Networks. [online] Available at: <https://www.researchgate.net/publication/285164623_An_Introduction_to_Convolutional_Neural_Networks> [Accessed 6 March 2020].

K. Sunil Manohar Reddy. 2017. Comparison of Various Face Recognition Algorithms. [online] Available at: http://www.ijarset.com/upload/2017/february/21_IJARSET_sunilreddy.pdf [Accessed 8 Mar. 2020].

Kelvin Salton do Prado. (2017). Face Recognition: Understanding LBPH Algorithm. [online] Available at: https://towardsdatascience.com/face-recognition-how-lbph-works-90ec258c3d6b [Accessed 8 Mar. 2020].

Matthew Stewart. 2019. Simple Introduction To Convolutional Neural Networks. [online] Available at: <https://towardsdatascience.com/simple-introduction-to-convolutional-neural-networks-cdf8d3077bac> [Accessed 11 March 2020].

Medium. 2016. Machine Learning is Fun! Part 4: Modern Face Recognition with Deep Learning. [online] Available at: https://medium.com/@ageitgey/machine-learning-is-fun-part-4-modern-face-recognition-with-deep-learning-c3cffc121d78 [Accessed 8 Mar. 2020].

Netguru.com. 2017. When, How And Why Use Node.js as Your Backend. [online] Available at: https://www.netguru.com/blog/use-node-js-backend [Accessed 8 Mar. 2020].

Paul Viola. 2002. The Viola/Jones Face Detector. [online] Available at: <https://www.cs.ubc.ca/~lowe/425/slides/13-ViolaJones.pdf> [Accessed 6 March 2020].

Pritchard, J., Stephens, M. and Donnelly, P., 2000. Inference Of Population Structure Using Multilocus Genotype Data. [online] Genetics. Available at: <https://www.genetics.org/content/155/2/945> [Accessed 10 March 2020].

Peter N. Belhumeur. 1997. Eigenfaces Vs. Fisherfaces: Recognition Using Class Specific Linear Projection. [online] Available at: <https://cseweb.ucsd.edu/classes/wi14/cse152-a/fisherface-pami97.pdf> [Accessed 17 March 2020].

Robbie Jaeger. 2017. Introduction to Node's Express JS [online] Available at: https://medium.com/@jaeger.rob/introduction-to-nodes-express-js-db5617047150 [Accessed 8 Mar. 2020].

Superdatascience.com. 2017. Face recognition using OpenCV and Python: A beginner's guide. [online] Available at: https://www.superdatascience.com/blogs/opencv-face-recognition [Accessed 8 Mar. 2020].

Southwestsolutions.com. 2017. Smart Cabinet Locker Package Mail Delivery System Keyless Locking Access Control. [online] Available at: https://www.southwestsolutions.com/smart-cabinet-locker-package-mail-delivery-system-keyless-locking-access-control [Accessed 8 Mar. 2020].

Shichao Zhang. 2017. Learning k for kNN Classification. [online] Available at: <https://www.researchgate.net/publication/312507655_Learning_k_for_kNN_Classification> [Accessed 11 March 2020].

Savan Patel. 2017. Chapter 2 : SVM (Support Vector Machine) — Theory. [online] Available at: <https://medium.com/machine-learning-101/chapter-2-svm-support-vector-machine-theory-f0812effc72> [Accessed 11 March 2020].

Sewell, M., 2007. Principal Component Analysis (PCA). [online] Martin Sewell. Available at: <http://www.stats.org.uk/pca/> [Accessed 11 March 2020].

Scikit-learn.org. 2019. Faces Recognition Example Using Eigenfaces And Svms — Scikit-Learn 0.22.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/auto_examples/applications/plot_face_recognition.html> [Accessed 9 March 2020].

slideshare.net 2017. Fisherfaces Face Recognition Algorithm. [online] Slideshare.net. Available at: https://www.slideshare.net/VishnuKN1/fisherfaces-face-recognition-algorithm [Accessed 9 Mar. 2020].

towardsdatascience.com. 2018. Support Vector Machine — Introduction to Machine Learning Algorithms. [online] Available at: https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47 [Accessed 8 Mar. 2020].

Tutorialspoint.com. 2020. Socket.IO - Overview - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/socket.io/socket.io_overview.htm> [Accessed 6 March 2020].

Tutorialspoint.com. 2020. Python Tutorial - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/python/index.htm> [Accessed 6 March 2020].

Tutorialspoint.com. 2020. MySQL - Introduction - Tutorialspoint. [online] Available at: <https://www.tutorialspoint.com/mysql/mysql-introduction.htm> [Accessed 6 March 2020].

Udara Bibile. 2017. Introduction to Socket.IO. [online] Available at: https://medium.com/@chathuranga94/introduction-to-socket-io-600025322cd2 [Accessed 8 Mar. 2020].

V. Deshmukh, S. 2017. Face Detection and Face Recognition Using Raspberry Pi. [ebook] International Journal of Advanced Research in Computer and Communication Engineering, p.2. Available at: https://ijarcce.com/upload/2017/april-17/IJARCCE%2014.pdf [Accessed 8 Mar. 2020].

www.researchgate.net. 2012. Face recognition using ortho-diffusion bases. [online] Available at: https://www.researchgate.net/publication/233545388_Face_recognition_using_ortho-diffusion_bases [Accessed 8 Mar. 2020].

Yann LeCun. 1994. Convolutional Networks For Images, Speech, And Time-Series .... [online] Available at: <http://yann.lecun.com/exdb/publis/pdf/lecun-bengio-95a.pdf> [Accessed 11 March 2020].

Yann LeCun. 1998. Gradient-Based Learning Applied To Document. [online] Available at: <http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf> [Accessed 24 March 2020].

Yu Tian. 2020. Face Anti-Spoofing By Learning Polarization Cues In A Real-World Scenario. [online] Available at: <https://arxiv.org/pdf/2003.08024v2.pdf> [Accessed 16 March 2020].

M. Resources table

Hardware	Software
Laptop	Python
Raspberry Pi	Node.js
Mechanical Servos	Visual studio code
BreadBoard & jumpers	Anaconda Spyder
Door sensors	
Camera	
Proximity Sensor	

N. Structure of Reports:

0. Appendix (Miscellaneous Analysis of Eigenface and Fisherface)

Fisherface

Fisherface is one of the most prevalent algorithms on face recognition and extensively believed it was superior to techniques compared to others algorithms which are more accurate and more reliable than eigenface. Such a method is based on Principle Component Analysis then applied with Latent Dirichlet allocation (LDA) (J. K. Pritchard, M. Stephensa, 2000) Bayes theorem linear transformation techniques, its focus on looks for the dimension, by maximizing the difference between the means of the classes normalized by variance to obtain features of image characteristic. Fisherface kinda looks alike to Eigenface but is more complex than Eigenface in terms of searching the`projection of face space. The technique works by calculation of ratio of in between-class scatters to within-class scatter therefore it-will take greater processing time, and during featuring it will require larger storage of the face for recognition. Besides, dueto the requirement for finer classification, the dimensioniof projection in face space is not as compact compared to Eigenface._

The Fisherface is an enhancement of better classification of various classes of images, especially remains effective even though facial images have the good strength of resistance on large disturbance by invariant illumination level and still able to detect different facial expressions. According to the research report “Comparison of Various Face Recognition Algorithms by K. Sunil Manohar Reddy”, the Fisherface has tested the result by using face image which about 50% person wearing spectacle to compare the error rate with EigenFace recognition and the result shows that FisherFace can reduce error rate from 7.3 percent to 0.6 percent even succeed in determining if a person is wearing glasses.

$$S_B = \sum_{i=1}^{c} N_i(\mu_i - \mu)(\mu_i - \mu)^T$$ Scatter between classes

$$S_W = \sum_{i=1}^{c}\sum_{j=1}^{n}(x_j - \mu_i)(x_j - \mu_i)^T$$ Scatter within classes

$$W_{opt} = \underset{W}{\operatorname{argmax}} \frac{|W^T S_B W|}{|W^T S_W W|}$$ The solution is the set W which maximize this ratio

Figure o.7 The formulas used in FisherFace

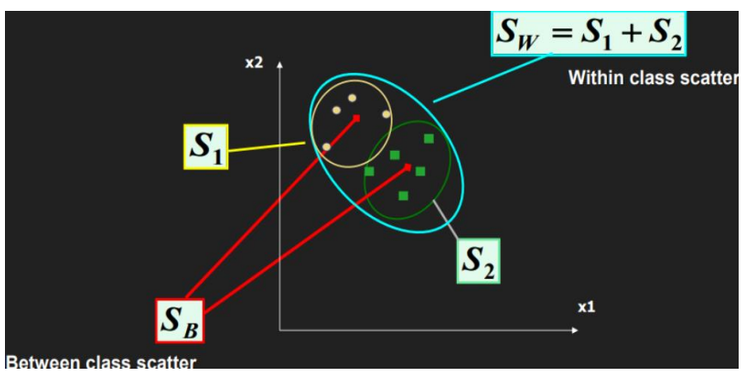


Figure o.8 PCA plot

Figure o.8 Scatter plot of two classes for linear discriminant analysis. Josh Starmer. 2017 StatQuest: PCA main ideas in only 5 minutes.

FisherFace requires data that is assumed to be distributed uniformly in each class, the objective is to exaggerate the ratio of the allying class scatter to within-class scatter illustrated with Figure g.8. After having the average of all faces of all persons in the dataset, it will successively compute the covariance matrix with the comer data by applying the equations shown in Figure o.7.

- Advantage of FisherFace

The single major advantage of using the FisherFace technique is that it is more adaptive to the invariant intensity of light. Further, although it is similar to Eigenfaces, it gives an enhancement of classification accuracy of differentiimage classes of images to even support the detection of facial expression.

- Disadvantage of FisherFace

The disadvantages of Fisherface was it is considered more complicated than the_Eigenface with using methods by finding the projectiontof face space. Computation'requires a-ratio of between-classes scatters-to within-class-scatters, intentionally it requires an enormous amount of processing time. Besides that, due to the cost for better classification, it may require a higher=dimension of projection-in_Face space is not as lightweight as Eigenface, resulting in occupying larger storage because of the normalized faces image effect to require an extensive amount of_time during recognition (K. Sunil Manohar Reddy. 2017).

EigenFaces

EigenFaces algorithm first introduced by Matthew Turka and Alex Paul Pentland in 1991, usually based upon a mathematical function called Principal Components Analysis(PCA) linear transformation techniques invented in 1901 by Karl Pearson. (Sewell, M., 2007). EigenFaces recognizer has used PCA dimension reduction by using the characteristics of a cluster of points compared to different groups of clusters. By projecting each image into face space, it trains itself via extracting principal components, simultaneously it in addition retains those records about the special characteristic which belongs to a person. It also relies on illumination as a component feature. In consequence, lights and shadows may be picked up by EigenFaces and get classified as them representing part of a face. Fig o.9 is about how Eigenface finds the dimensions of data with the highest variance by converting images into vector form. Whenever introduced a new image to the eigenface algorithm, it will cycle the same process as sequences:

1. Extract the principal-components from the induced image and generate the key vector which is represented as a face.

2. After acquiring the vectors with variations, by comparing those features with the list of components it will literally get stored during training.

3. Check if the image is sufficiently close to the face space, and select the ones with the best match.

4. Return the subject's label correlated with that best match component and classify it as known otherwise be classified as unknown.

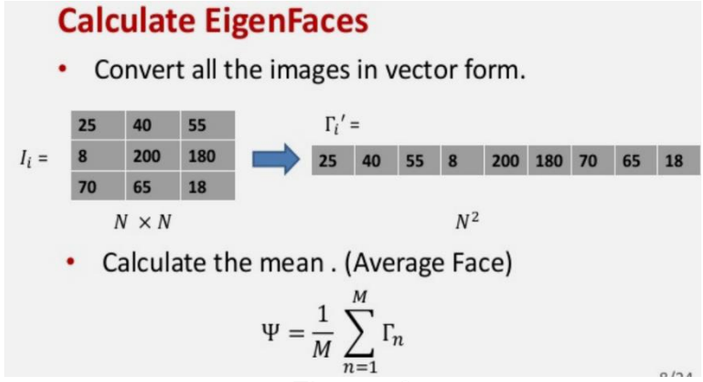


Figure o.9

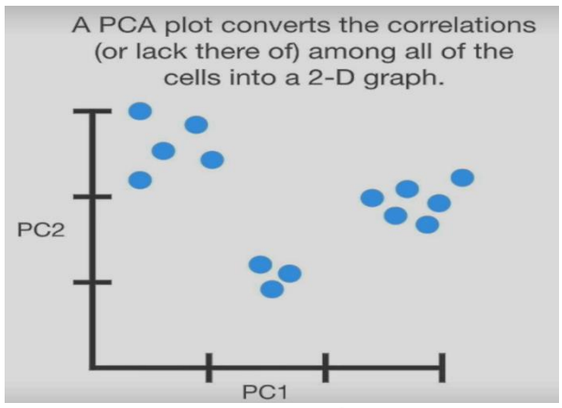


Figure o.10

Figure o.10 Eigenvectors and eigenvalues are the linear algebraoconcepts that are used to compute from the covariance matrix by picking the dimension of the greatest variance to determine the principal components-of the data. The figure above shows those clusters of points are different classes that contain groups of locations of points.

- Advantage of EigenFace algorithm

Eigenface provides a casual and inexpensive cost on face recognition in that: Regards on the training process of eigenface is automatic and elementary to program. Eigenface_adequately reduces statistical`complexity in the representation of the facial image. Reduce the chance of recalculation by depending on the image database which has been preprocessed, face recognition can be achieved in real-time. (Peter N. Belhumeur. 1997)

- Disadvantages of EigenFace algorithm

Eigenfaces are too sensitive to lighting, require scaleiand face angle, and highly depending on a light intensity condition stable environment. Eigenface has a weakness on distinguishing expression changes. It depends mainly on illumination encoding and which does not prescript much useful information regarding the actual face. Despite that, such methods are `appearance-based' in nature. Thus it requires a lot of learning which is a very time-consuming process while making it difficult to keep updating the face database. Lastly, Eigenface recognition is efficient only when collected the number of face classes is larger than the dimensions of the face space.

Marking Scheme

CCP3012-3026 Individual Project Marking Scheme (For Investigative Project / Application Project) Project Proposal Review (100 Marks, Weighted @ 20%)								
Student ID:	***Student Name: 1st Marker / 2nd Marker:***							
LEARNING OUTCOME	**MARKING CRITERIA**	**SCALE**					**Task Marks (Max. 100)**	**Task Weighted Marks**
		Fail (0-49)	**3rd Class (50-59)**	**2nd Lower Class (60-69)**	**2nd Upper Class (70-79)**	**1st Class (80-100)**		
Part A: Problem Statement								
CLO1: Construct a problem statement working from unstructured ideas	**1. Background to Project & Proposed Work**	Discussion was very poorly written and inadequate.	Discussion showed adequate understanding of issues and technology involved.	Discussion showed good understanding of issues and technology involved.	Discussion showed very good understanding of issues and technology involved.	Discussion showed excellent understanding of issues and technology involved.		/30
	2. Aims and Objective of the Project	The aims and objective of the project were very poorly described.	The aims and objective of the project were adequately described.	The aims and objective of the project were fairly well described.	The aims and objective of the project were clear and succinctly described.	The aims and objective of the project were very clear and succinctly described.		/30
							Subtotal (60%)	/60
Part B: Project Planning								
CLO2: Plan the conduct of an investigative project	**3. Technical Skills and the Hardware/ Software Resources**	Skills and resources required were mostly not identified.	Skills and resources required were fairly adequate, clear and realistic.	Skills and resources required were good, clear and realistic.	Skills and resources required were very good, clear and realistic.	Skills and resources required were excellent, clear and realistic.		/10
	4. Sources of Information/ Bibliography	Depth and breadth of sources provided was very poor.	Depth and breadth of sources provided was adequate.	Depth and breadth of sources provided was good.	Depth and breadth of sources provided was very good.	Depth and breadth of sources provided was excellent.		/10
	5. Proposed Structure of the Report	Structure not cross-referenced with the objectives.	Structure adequately cross-referenced with the objectives.	Structure well cross-referenced with the objectives.	Structure very well cross-referenced with the objectives.	Structure thoroughly well cross-referenced with the objectives.		/10
	6. The Project Plan	Not realistic and in line with real-world expectations.	Somewhat realistic and in line with real-world expectations.	Fairly realistic and in line with real-world expectations.	Mostly realistic and in line with real-world expectations.	Realistic and in line with real-world expectations.		/10
							Subtotal (40%)	/40
							Total (100%)	/100

Product Mark

Product marking criteria & deliverables

FITNESS FOR PURPOSE (40%)	**Mark Range (to total 40%)**
Meeting of Requirements as identified during project. Quality of functionality. HCI (if relevant). Other criteria & deliverables (specify) :-	To be agreed
Default criteria and mark ranges	
Meeting of Requirements as identified during project.	0 – 15
Quality of functionality.	0 – 15
HCI.	0 – 10
BUILD QUALITY (60%)	**Mark Range (to total 60%)**
Requirements specification and analysis Design specification Code quality Test plans and results Other criteria & deliverables (specify) :-	To be agreed
Default criteria and mark ranges	
Requirements specification and analysis	0 – 15
Design specification	0 – 15
Code quality	0 – 15
Test plans and results	0 – 15

Report Mark

Application Project

Semester 1 (Weighted @ 20%)	
Project Proposal	100%
Semester 2 (Weighted @ 80%)	
Report	40%
Project Outcome	50%
Viva	10%
Total	100%

Report

Abstract & Introduction	5%
Analysis	30%
Synthesis	30%
Evaluation & Conclusions	30%
Presentation	5%
Total	100

Product

Fitness for Purpose	40%
Build Quality	60%
Total	100%

Viva

Presentation / Demonstration	50%
Discussion	50%
Total	100

CCP3012_0188039

ORIGINALITY REPORT

3%	0%	1%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to University of Wales Institute, Cardiff Student Paper	<1%
2	Submitted to King's College Student Paper	<1%
3	Yaxing Wang, Lin Qi, Xin Guo, Lei Gao. "Face recognition based on histogram of the 2D-FrFT magnitude and phase", 2014 International	<1%

UOW MALAYSIA KDU

PART OF THE UNIVERSITY OF WOLLONGONG AUSTRALIA GLOBAL NETWORK

CCP3014-3024 Individual Project
Project Proposal Review
Marking Scheme (100 Marks, Weighted @ 20%)

Student Name: Go Yik Yek	**Student ID: 0188039**
Supervisor: Mr. Danny Chen	**2nd Marker: Ms. Rosmah Ismail**
Date of Project Proposal Review: 31 March 2020	

Review Outcomes:

The topic is appropriate to the student's programme — Yes / No

The project contains sufficient practical work using computing skills relevant to the programme. — Yes / No

The proposal (select one)…

- is accepted without changes
- needs the changes listed overleaf.
- cannot be made satisfactory and a new topic is required.

Project Type (select one):

Application Project

Investigative Project

Ethics Form (select one):

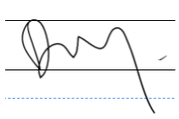
- Form B has been reviewed.
- Form B is required but has not been provided.
- Form A is required and has already been submitted to the Project supervisor
- Form A is required and has not yet been submitted.
- Other (explain)

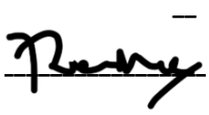
Risk Assessment (select one):

- Reviewed
- Required but not provided
- Not Required

Signatures:

Student  **Go Yik Yek**

Supervisor  **Mr. Danny Chen**

Second Marker  **Ms. Rosmah Ismail**

Changes To Proposed Aims/Objectives
No changes to the proposed aims/objectives
Changes To Project Plan
No changes
Resource Issues
No issues
Other Comments
Need to re-check in-text citations, to follow the right formatting. Need to check on figures attached. Make sure the figures are labelled and has a title.

UOW Malaysia KDU Penang University College
Research Ethics Registration and Approval Form

Ethical category:
Red ☐
Blue ☐
Green ☑

Section One: Registration ***[To be completed by student]***

Title of research project/dissertation	Smart IoT Cabinet with facial recognition
Researcher's name	Go Yik Yek
Programme of study	UCSE2
Academic Year	2020
Module code	CCP3012
Supervisor's name	Mr. Danny Chen
Second marker's name	Ms. Rosmah Ismail
Start Date of Project	10.05.2020

Brief outline of research topic:

To implement face recognition to cabinet system using raspberry Pi. The raspberry Pi will be programmed by python as a client ant it is connecting to servos, and sensors on detecting the available of the case and sent the online status to the Node.js back end. The smart IoT Cabinet will serve two main functions on either just for temporary storing in a period or else choose the service as to transfer the stuff in the case to another location in conceptual. The authentication for temporary storing will using facial recognition algorithm, PIN number or QR code, and the transfer purpose only allow to use QR code to unlock. Node.js is the backend to handle the request and store the data for client base.

Short description of proposed research methods including identification of participants:

1. LBHP Verification or authentication of a facial image: it basically compares the input facial image with the facial image which is related to the user.

2.Improve the system security for facial recognition by adding a pre-trained model with Keras Neural network in detecting eyes for the liveness detection.

3. Connect the python client to the Node.js backend system.

UOW Malaysia KDU Penang University College
Research Ethics Registration and Approval Form

Ethical category:
Red ☐
Blue ☐
Green ☒

Ethical considerations in the research project	YES	NO
1. Does your research involve an external organisation or partner?	☐	☒
2. Does your research involve human participants?	☐	☒
3. If yes to Q.2, will you inform the participants about the research?	☐	☒
4. Will you obtain their consent using the standard consent form?	☐	☒
5. Is any deception involved?	☐	☒
6. Do any participants constitute a 'vulnerable group'?	☐	☒
7. Will the research involve the following information?		
Commercially sensitive	☐	☒
Personally sensitive	☐	☒
Politically sensitive	☐	☒
Legally sensitive	☐	☒
8. Is the research likely to have any significant environmental impacts?	☐	☒
9. Are there likely to be any risks for the participants in your research?	☐	☒
10. Are there likely to be any risks for you in conducting the research?	☐	☒
11. If yes [to 5, 6, 7, 8, 9 or 10 above] have you identified steps to address the issues and mitigate any risks to participants, yourself or the environment?	☐	☒

Statement to explain how any issues identified above will be addressed and what steps will be taken to mitigate such risks or adverse impacts

All information collected will be kept in the evidences folder and will not disclose to any thirdparty.

Ethical category of research project *[Please tick as appropriate]*

Red	☐	vulnerable participants; human tissue; sensitive data; risks to participants & researchers etc.
Blue	☐	human participants requiring informed consent; commercially sensitive information etc.
Green	☒	no participants involved; secondary data only; no sensitive data

I have read the University College Ethics Policy and Procedures and confirm that the answers I have given above are correct. Where issues arise under items 5, 6, 7, 8, 9 or 10 [above] I have described in writing how I intend to approach these issues in the research.

Researcher's signature: _____  **_____**
Date: _____17.04.2020_____

Section 1 Ethics Registration to be submitted to Principal Supervisor or Module Tutor and allocated to a reviewer as follows:
Green risk - may be approved by Supervisor
Blue risk - to be submitted for approval by one independent reviewer (second marker)
Red risk - to be submitted for approval by two independent members of Research Committee

UOW Malaysia KDU Penang University College
Research Ethics Registration and Approval Form

Ethical category:
Red ☐
Blue ☐
Green ☑

Section Two: Approval

Supervisor/Module Tutor's name confirming ethical risk status	**Danny Chen Sien Yau**

Ethical approval *[Please tick as appropriate]*

Green - Ethical approval is given without conditions (supervisor may approve).	☒
Blue – (to be approved by one independent reviewer) Ethical approval is given with the following conditions: * Information to be provided to all participants. * Participant consent to be obtained in accordance with the University College procedures. * Data to be stored and destroyed securely in accordance with University College guidelines. * Adherence to Data Protection Act. * Anonymity to be offered to participants. * Commercial confidentiality to be provided to organisations(s). * Software vulnerabilities, exploits, hazardous software etc. not to be published without prior 'responsible disclosure' to the affected supplier. * Other (please state):	 ☐ ☐ ☐ ☐ ☐ ☐ ☐ ☐
Red - Project is referred to Research Committee for approval by two independent reviewers.	☐

Name & role of reviewer 1: Danny Chen Sien Yau (Supervisor)

Signature: _____ _____

Date: 17/04/2020

Name & role of reviewer 2: Rosmah binti Ismail

Signature: _____ _____

Date: 10-4-2020

Outcome of Review:

Overall the proposed idea is worth to be executed. Student only need to check on the report format, figures and citations.

APPENDIX B – USE CASE TABLES

Use Case	View list of cabinet boxes
Summary	This use case is to view list of cabinet boxes
Actor	Customer
Trigger	Customers clicked the Unlock, Local Storage or Transfer Storage button on the main page.
Primary Scenario	Pages are being shown on screen.
Alternative Scenario	Non Applicable
Exceptional Scenario	Non Applicable
Pre-Condition	There is data in the database. There is any empty state box(es).
Post-Condition	Select any boxes available on the page and choose to proceed or else cancel. If proceed, for local storage service it will redirect to form. For transfer storage service it will redirect to select target location page or the form page.
Assumption	1. The client ware is connected to the server and the database consists of data. 2. The client ware received JSON response

Use Case	View list of locations
Summary	This use case is to show customers the available destination for the other set of cabinet boxes.
Actor	Customer
Trigger	Customer reserved local box id.
Primary Scenario	1. Customer clicked the transfer storage button. 2. Customers select the local boxid and proceed to reserve.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	1. The server had updated the selected local box id from Empty state

	to Reserved state.
Post-Condition	1. Proceed to select target box id page.
Assumption	1. The client ware is connected to the server and the database consists of data. 2. The client ware received JSON response

Use Case	Unlock Cabinet Box with QR
Summary	This use case is to let customer to unlock cabinet box with QR code
Actor	Customer
Trigger	Customer selected a local box id and clicked the QR scan icon button.
Primary Scenario	1. Customer had clicked the unlock button on the main page. 2. Customer had selected a box id. 3. Customer clicked the QR icon icon button.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	1. Customer had a scanned face while interacting with the form. 2. Customer had paid for the service. 3. Customers received QR code in email after paying.
Post-Condition	If the QR scanning detected relevant data sent to server to check
Assumption	Not Applicable

Use Case	Unlock cabinet box with facial recognition
Summary	This use case is to let the customer unlock the cabinet box with face recognition.
Actor	Customer
Trigger	Customer had selected a local box id and clicked the facial recognition icon button.
Primary Scenario	1. Customer had clicked the unlock button on the main page. 2. Customer had selected a box id. 3. Customer clicked the face recognition icon button.
Alternative Scenario	Not Applicable

Exceptional Scenario	Not Applicable
Pre-Condition	1. Customer had a scanned face while interacting with the form. 2. Customer had paid for the service.
Post-Condition	If the recognition has output of the value passed to the server to check.
Assumption	The LBPH recognition file is trained after the customer paid the service.

Use Case	Cancel form or reserved box(es)
Summary	This use case is to let customer to cancel form or reserved box(es)
Actor	Customer
Trigger	Use clicked the cancel button
Primary Scenario	Navigated to the non main pages which use to reserve box id.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	Going through the pages after clicking the ‘storage’ or ‘transfer storage’ button of selecting box page, selecting destination page and the form page. After clicking the cancel button right on the right upper corner on the page.
Post-Condition	Not Applicable
Assumption	1. Customers are able to click the cancel button which showed on the page.

Use Case	Reserve box
Summary	This use case is to let the customer select and reserve the available box before the form page shows.
Actor	Customer
Trigger	Select either transfer storage or local storage button. Then if there are available box shows on the page the customer will choose any one and proceed.
Primary Scenario	Main page is shown.

Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	Navigated to page local_store.ejs, transferstore_selecttargetbox.ejs, or transf_store.ejs page.
Post-Condition	Proceed to from or get redirect to select target location page
Assumption	The backend provided a response to the page and JavaScript had updated the content on the page.

Use Case	Payment
Summary	This use case is to allow customers to make payments.
Actor	Customer
Trigger	Customer clicked the submit button.
Primary Scenario	1. Customer reserved local box id for local storage service. 2. Customers filled up their forms.
Alternative Scenario	1. Customer reserved local box id and target box id for transfer storage service. 2. Customers filled up their forms.
Exceptional Scenario	Not Applicable
Pre-Condition	The form has filled with email and phone number information in the right format.
Post-Condition	1. The Stripe API shows up and lets customers make payment. 2. IF the payment succeeds the client ware will forward the request to the server to make a new servicing record.
Assumption	1. The Stripe API is able to contact with the client ware. 2. The database is functioning normally.

Use Case	Capture face
Summary	This use case is to let the customer have facial scanning to gather their face data.
Actor	Customer

Trigger	Clicked the confirm button in the alert dialog box after checking the Face Recognition check box.
Primary Scenario	1. Customer selected local storage service. 2. Customer reserved box id after selecting it. 3. The form is displayed to the customer.
Alternative Scenario	Non Applicable
Exceptional Scenario	Non Applicable
Pre-Condition	1. A box id is re by the customer. 2. The customer had made a fece record during the time interacting with the form and afterward decided to pay for the service. 3. The sqlite file, and the folder to store face dataset exist in the system.
Post-Condition	Proceed to payment with Stripe or cancel the form.
Assumption	The OpenCV python is running properly after clicking.

Use Case	Staff login
Summary	This use case is to allow staff to
Actor	Staff
Trigger	User fill in login form and pressed login button.
Primary Scenario	Navigate to the staff portal page.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	1. The system database has the staff account.
Post-Condition	1. If the username and email are correct, create a session redirect to the home page.
Assumption	The staff portal is online. And the database is working fine.

Use Case	Staff Logout
Summary	This use case is to allow staff to logout.
Actor	Staff
Trigger	Staff clicked the logout button.
Primary Scenario	Staff' device browsers still consiste the session key which is not reaching expiration.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	Staff had logged in to their account
Post-Condition	Destroy the session and show the login_register page.
Assumption	The staff portal is online. And the database is working fine.

Use Case	Staff register
Summary	This use case is to allow staff to register
Actor	Staff
Trigger	Fill in and submit the register form.
Primary Scenario	Navigate to the staff portal.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	Not Applicable
Post-Condition	Insert new staff to table.
Assumption	The staff portal is online. And the database is working fine.

Use Case	Staff gather new task
Summary	This use case is to allow staff to gather a new task.
Actor	Staff

Trigger	Clicking the gather new task button.
Primary Scenario	Staff had logged in to the portal.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	The box_servicing table should have at least one new “T_S” transfer service which was recently requested by a customer.
Post-Condition	Insert new record to transfer allocation table.
Assumption	The staff portal is online. And the database is working fine.

Use Case	Staff cancel task
Summary	This use case is to allow staff to cancel their task.
Actor	Staff
Trigger	Staff had clicked the cancel task button.
Primary Scenario	Staff had acquired a task which in Pending state. And the QR are not being used.
Alternative Scenario	Not Applicable
Exceptional Scenario	Not Applicable
Pre-Condition	The database consists of the current task in aquire_id for transfer_allocation table.
Post-Condition	Set the current record to cancel state.
Assumption	The staff portal is online. And the database is working fine.

APPENDIX C – TEST CASE

1) Cabinet client customer detail form

Test Scenario ID	Testing the customer detail form
Test Case Description	Test the form validation for client ware
Prerequisite	1. Let the client ware to run 2. Accessing the customer form page
Tests execution steps 1. Placing detail to the form that showed	

No.	Inputs	Expected Output	Actual Output	Status
1.	Email:%%%%%%%%% phone:1	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
2.	Email:http://localhost:3000/transfer_store_selected_targetbox phone:12	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
3.	Email:asdasd****** phone:123	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
4.	Email:□□□□ phone:1234	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
5.	Email:中文翻译@gmail.com phone:12345	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
6.	Email:%#$%g phone:123456	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
7.	Email:_+KJHG.{} phone:1234567	Invalid mail / Invalid phone	Invalid mail / Invalid phone	Pass
8.	Email:432sdfsd@z.net phone:12345678	Invalid phone	Invalid phone	Pass
9.	Email:facebook@yahoo.com phone:123456789	Invalid phone	Invalid phone	Pass
10.	Email:sdfsdf_+/*******	N/A	N/A	Pass

	@gmail.com phone:1264567896			
11.	Email:sdfsdf_+/*****□ □**@gmail.com phone:1234567890	Invalid mail	Invalid mail	Pass

2) Unlocking with QR

Test Scenario ID	QR scanning test			
Test Case Description	Testing the QR code scaning feature			
Prerequisite	.The Nodejs able to call up the qr.py code			
Tests execution steps 1. Using any qr code to test the scanning				
No.	Inputs	Expected Output	Actual Output	Status
1.	Using QR not contain JSON object	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
2.	Using QR contain JSON object but not the right value { "localboxID": "localboxID must be number", "targetboxID": "targetboxID must be number" }	No respond	No respond	Pass
3.	Using QR contain JSON object and the right value {"sid":59,"cqr":"LFWZNjZG0FQKVFtFj6x3"}	Unlocking Successfully	Unlocking Successfully	Pass
4.	Using QR contain JSON object and the false value {"sid":59,"cqr":"LFWZNjZG0FQKVFtFj6x3"}	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
5.	Using QR contain	No QR code or Face	No QR code or Face	Pass

	invalid JSON object{"sid":&@$,"cqr":"@##@*@"}	detected, times up.	detected, times up.	
6.	Using QR contain invalid JSON {"sid":&@$,"cqr":"@##@*@, "tid":"11", "bqr":"12","tid":"qqq", "aqr":"was"}	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
7.	QR contain incorrect tid:,bqr but correct sid:, cqr:, to correct local box id {"tid":13,"bqr":"h8OWoRv2QT6afnwgrIv7","sid":"60","cqr":"AUefhN7xUXXpTWMDLPSP"}	Unlocking Successfully	Unlocking Successfully	Pass
8.	QR contain correct , sid,cqr, and incorrect tid, aqr to correct local box id {"sid":"60","cqr":"AUefhN7xUXXpTWMDLPSP""tid":13,"aqr":"JjerN50LVAVJKGoGieIp"}	Unlocking Successfully /Unlocking failed	Unlocking Successfully /Unlocking failed	Pass
9.	QR contain correct tid: and aqr to correct target box id {"tid":13,"aqr":"JjerN50LVAVJKGoGieIp"}	Unlocking Successfully	Unlocking Successfully	Pass

3) test case for deliver staff portal

Test Scenario ID	Staff Register
Test Case Description	Testing the staff registering form
Prerequisite	The staff code is running and connected to the database.

Tests execution steps				
No.	Inputs	Expected Output	Actual Output	Status
1.	name:SELECT * FROM staff email:SELECT * FROM staff@gmail.com pass:SELECT * FROM staff	Invalid email address!	Invalid email address!	Pass
2.	name:SELECT * FROM staff email:testerstaff@gmail.com pass:xyz%27+OR+1%3D1+--	your account has been created successfully, Now you can Login	your account has been created successfully, Now you can Login	Pass
3.	name:x' or 1=1 -- email:testgg□□□□@gmail.com pass:x' or 1=1 --	Invalid email address!	Invalid email address! Illegal mix of collations (latin1_swedish_ci,IMPLICIT) and (utf8mb4_unicode_ci,COERCIBLE) for operation '='	Pass
4.	name:"SELECT * FROM staff " email:testgg@gmail.com pass:"SELECT * FROM staff "	your account has been created successfully, Now you can Login	your account has been created successfully, Now you can Login	Pass
5.	name:!@#$#%^%&*^(&)*() email:test3@com pass:!@#$#%^%&*^(&)*()	Invalid email address!	Invalid email address!	Pass
6.	name:!@#$#%^%&*^(&)*() email:test3@1.com pass:!@#$#%^%&*^(&)*()	your account has been created successfully, Now you can Login	your account has been created successfully, Now you can Login	Pass
7	name:`12sdsda sdf dsfsdfsdfsdfsdfsdf34	Invalid email address!	your account has been created successfully,	Fail

	5□, □□♂, □, □, □, □, □, ♥, □ email:popopo////****%%%%%fl@bsb.com pass:□, □□♂, □, □, □, □, □, ♥, □		Now you can Login	
8	name:□, □□♂, □, □, □, email:□@q l.com pass:□, □□♂, □, □, □,	Invalid email address!	Invalid email address!	Pass
9	Using the mail already registered name:test email:test@gmail.com pass:123123	This E-mail already in use!	This E-mail already in use!	Pass
10.	name:UNION SELECT 1,@@version;- - email:123F@a.net pass:123123	your account has been created successfully, Now you can Login	your account has been created successfully, Now you can Login	Pass
11.	name:Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) email:Chrome/89.0.4389.90 Safari/537.3 adssaddsasda@777.com pass:Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/89.0.4389.90 Safari/537.3	your account has been created successfully, Now you can Login	your account has been created successfully, Now you can Login	Pass

Test Scenario ID	**Staff Login**
Test Case Description	Testing the staff login to the system
Prerequisite	The staff code is running and connected to the database.

Tests execution steps

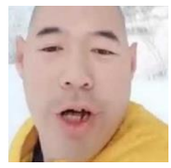
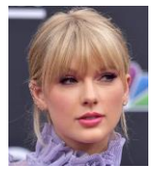
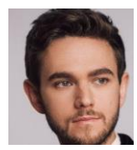
No.	Inputs	Expected Output	Actual Output	Status
1.	Email: pass:	Invalid Password!	Invalid Password!	Pass
2.	Email:test@gmail.com pass:x' AND 1=(SELECT COUNT(*) FROM staff); --	Invalid Password!	Invalid Password!	Pass
3.	Email:x'; DROP TABLE users; -- @nnmail.com pass:x'; DROP TABLE users; --	Invalid Email Address!	Invalid Email Address!	Pass
4.	Email:test@gmail.com pass:Invalid Email Address!	Invalid Password!	Invalid Password!	Pass
5.	Email:□□□□□□□□□□□□@gmail.com pass:□□□□□□□□□□□□□□□□	Invalid Password!	Illegal mix of collations (latin1_swedish_ci,IMPLICIT) and (utf8mb4_unicode_ci,COERCIBLE) for operation '='	Pass
6.	Email:test@gmail.com pass:test123 Legit account	Logged in	Logged in	Pass

4) Facial recognition

Test Scenario ID	Face recognition for clientware
Test Case Description	Testing the facial recognition feature
Prerequisite	It is required to run the facial recognition code.

Tests execution steps

1. Providing image of face or personnel face.

No.	Inputs	Expected Output	Actual Output	Status
1.	Using irrelevant face photo 	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
2.	Using irrelevant face photo 	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
3.	Using irrelevant face photo 	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
4.	Using irrelevant face photo 	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
5.	Using irrelevant face photo 	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
6.	Using relevant photo with capture by	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass

	phone 			
7.	Using relevant real face without eye blink	No QR code or Face detected, times up.	No QR code or Face detected, times up.	Pass
8.	Using relevant face with eye blink	Unlocked	Unlocked	Pass

5) API testing 01 - 14

Test Scenario ID	Api test 01 nihao
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.
Tests execution steps 1. Using Postman rest api testing tool	

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	unauthorised	unauthorised	Pass
2.	Using valid token without JSON object	ni hao ma: cabinet id	ni hao ma: 1	Pass
3.	Using valid token with random JSON object { "serviceId222": "3", "boxId123":"3", "customerQr123123123123":"D234DSF3DSFasd35gdfj" }	ni hao ma: cabinet id	ni hao ma: 1	Pass
4.	Using valid token with valid object but provide with bad value { "serviceId": "asd@", "boxId":"@3", "customerQr":"@@" }	ni hao ma: cabinet id	ni hao ma: 1	Pass

Test Scenario ID	Api test 02 unlockforcustomer
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorize	Unauthorize	Pass
2.	Using valid token but without JSON value	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
3.	Using valid token with random JSON value { "serviceId222": "3", "boxId123":"3", "customerQr123123123123":"D234DSF3DSFasd35gdfj" }	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
4.	Using valid token with valid object but provide with bad value { "serviceId": "asd@", "boxId":"@3", "customerQr":"@@" }	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
5.	Using valid token with valid object and provide with correct code	{ "validation": "yes", "expiration": "2021-03-27T15:42:13.000Z", "gpiopin": "pin17", "boxid": 4 }	{ "validation": "yes", "expiration": "2021-03-27T15:42:13.000Z", "gpiopin": "pin17", "boxid": 4 }	Pass
6.	Using valid token with valid object and provide with correct code but the service is expired	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass

Test Scenario ID	Api test 03 unlockforcustomerusingface
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
2.	Using valid token but provide no JSON value	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
3.	Using valid token with valid object but provide with bad value { "boxId": "@@", "labelString": "@@@@@@@d" }	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
4	Using valid token with valid object and provide with correct value { "boxId": "5", "labelString": "User.1." }	{ "validation": "yes", "expiration": "202?-0?-30THH:MM:SS.000Z", "gpiopin": "pinX", "boxid": X }	{ "validation": "yes", "expiration": "2021-04-30T04:42:21.000Z", "gpiopin": "pin4", "boxid": 5 }	Pass
5	Using valid token with valid object and provide with but incorrect value { "boxId": '5', "labelString": 'User.1.\r\n' }	{ "validation": "no", "msg": "record not found" }	{ "code": 400, "message": "bad request" }	Acceptable
6	Using valid token with valid object and provide with correct value but the service is expired { "boxId": "5", "labelString": "User.1." }	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass

Test Scenario ID	Api test 04 unlockbeforetransfer
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "tid": "tid is required", "bqr": "bqr is required", "bid": "bid is required" }	{ "tid": "tid is required", "bqr": "bqr is required", "bid": "bid is required" }	Pass
3.	Using valid token but provided wrong JSON { "boxId": "5", "labelString": "User.1." }	{ "tid": "tid is required", "bqr": "bqr is required", "bid": "bid is required" }	{ "tid": "tid is required", "bqr": "bqr is required", "bid": "bid is required" }	Pass
4.	Using valid token and JSON but provided bad value	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
5.	Valid token and correct tid and bid except bqr { "tid": "12", "bqr": "@@aaaas", "bid": "3" }	"msg": "record not found" }	"msg": "record not found" }	Pass
6.	Using valid token with valid object and provide with correct value	{ "validation": "yes", "expiration": "YYYY-MM-DDTHH:MM:SS.000Z", "gpiopin": "pinX", "boxid": X }	{ "validation": "yes", "expiration": "2021-03-23T12:33:31.000Z", "gpiopin": "pin16", "boxid": 3 }	Pass

Test Scenario ID	Api test 05 unlockaftertransfer
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "tid": "tid is required", "aqr": "aqr is required", "bid": "bid is required" }	{ "tid": "tid is required", "aqr": "aqr is required", "bid": "bid is required" }	Pass
3.	Using valid token but provided wrong JSON { "boxId": "5", "labelString": "User.1." }	{ "tid": "tid is required", "aqr": "aqr is required", "bid": "bid is required" }	{ "tid": "tid is required", "aqr": "aqr is required", "bid": "bid is required" }	Pass
4.	Using valid token and correct JSON but wild bad value { "tid": "\\[][[;;", "aqr": "`12`1****", "bid": "232<>?ZZ□" }	{ "validation": "no", "msg": "record not found" }	{ "validation": "no", "msg": "record not found" }	Pass
5.	Using valid token and correct JSON with correct value { "tid": 12, "aqr": "JMLoFtIdrBlPyPBAd3QH", "bid": 30 }	{ "validation": "yes", "expiration": "YYYY-MM-DDTHH:MM:SS.000Z", "gpiopin": "pinX", "boxid": X }	{ "validation": "yes", "expiration": "2021-03-23T12:33:31.000Z", "gpiopin": "pin22", "boxid": 30 }	Pass

Test Scenario ID	Api test 06 box_itself_reserve
Test Case Description	Testing the backend works stably on returning suitable data.

Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "box_id": "box_id is required" }	{ "box_id": "box_id is required" }	Pass
3.	Using valid token but provided wrong JSON { "boxId": "5", "labelString": "User.1." }	{ "box_id": "box_id is required" }	{ "box_id": "box_id is required" }	Pass
4.	Using valid token and correct JSON but wild bad value { "box_id": "232<>?ZZ□" }	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass
5.	Using valid token and correct JSON with correct value but is wrong format { "box_id": 28" }	{ "code": 400, "message": "bad request" }	{ "code": 400, "message": "bad request" }	Pass
6.	Using valid token and correct JSON with correct value in right format	{ "box_itself_reserve": "Yes", "msg": "Reserved", "boxid": "X" }	{ "box_itself_reserve": "Yes", "msg": "Reserved", "boxid": "28" }	Pass
7.	Using valid token and correct JSON with correct value in right format which the box id not belongs to the cabinet sets	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass

Test Scenario ID	Api test 07 box_reserve_another_box
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool				
No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass
3.	Using valid token but provided wrong JSON { "boxId": "5", "labelString": "User.1." }	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass
4.	Using valid token and correct JSON with correct value but is wrong format { "box_id": 28" }	{ "code": 400, "message": "bad request" }	{ "code": 400, "message": "bad request" }	Pass
5.	Using valid token and correct JSON with correct value in right format which the local and target box id is belongs other cabinet sets {"localbid":"20","targetbid": "21"}	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass
6.	Using valid token and correct JSON with correct value in right format but the target box id is belongs to other cabinet sets.(been reserved one local cabinet box id) {"localbid":"9","targetbid":" 21"}	{ "box_reserve_another" : "Yes", "msg": "Reserved", "boxid": "X" }	{ "box_reserve_another": "Yes", "msg": "Reserved", "boxid": "21" }	Pass
7.	Using valid token and correct JSON with correct value in right format but the target box id is belongs to other cabinet sets.(not reserve any local cabinet box id) {"localbid":"9","targetbid":" 21"}	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass

8.	Using valid token and correct JSON with random value {"localbid":"!@#!@#","targetbid":"~~d!@□"}	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass

Test Scenario ID	Api test 08 list_of_location
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "rows": [{ "cabinet_id": X, "location_name": "XXXX XXl", "cabinet_addr": " XXXXX" }] }	{ "rows": [{ "cabinet_id": 2, "location_name": "Penang Sunway Hotel", "cabinet_addr": " 33, Lorong Baru, George Town, 10400 George Town, Pulau Pinang" }] }	Pass
3.	Using valid token but provide random JSON value {"localbid":"!@#!@#","targetbid":"~~d!@□"}	{ "rows": [{ "cabinet_id": X, "location_name": "XXXX XXl", "cabinet_addr": " XXXXX" }] }	{ "rows": [{ "cabinet_id": 2, "location_name": "Penang Sunway Hotel", "cabinet_addr": " 33, Lorong Baru, George Town, 10400 George Town, Pulau Pinang" }] }	Pass

Test Scenario ID	Api test 09 list_all_localCabinetboxes
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	
2.	Using valid token but provide no JSON value	{ "rows": [{ "box_id": x }, { "box_id": x }] }	{ "rows": [{ "box_id": 1 }, { "box_id": 2 }, { "box_id": 3 }, { "box_id": 4 }, { "box_id": 5 }, { "box_id": 6 }, { "box_id": 7 }, { "box_id": 8 }, { "box_id": 9 }, { "box_id": 10 }] }	Pass
3.	Using valid token but provide random JSON value {"localbid":"!@#!@#" ,"targetbid":"~~d!@ □"}	{ "rows": [{ "box_id": x }, { "box_id": x }] }	{ "rows": [{ "box_id": 1 }, { "box_id": 2 }, { "box_id": 3 }, { "box_id": 4 }, { "box_id": 5 }, { "box_id": 6 }, { "box_id": 7 }, { "box_id": 8 }, { "box_id": 9 }, { "box_id": 10 }] }	Pass

Test Scenario ID	Api test 10 list_of_localCabinetboxes
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but provide no JSON value	{ "rows": [{ "box_id": x}] }	{ "rows": [{ "box_id": 1 }, { "box_id":	Pass

			3 }, { "box_id": 7 }, { "box_id": 8 }, { "box_id": 9 }] }	
3.	Using valid token but provide random JSON value {"localbid":"!@#!@#","targetbid":"~~d!@□"}	{ "rows": [{ "box_id": x}] }	{ "rows": [{ "box_id": 1 }, { "box_id": 3 }, { "box_id": 7 }, { "box_id": 8 }, { "box_id": 9 }] }	Pass

Test Scenario ID	Api test 11 list_of_targetCabinetboxes/:cabinetid
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Valid token with random value in url http://localhost:4000/api/client/list_of_targetCabinetboxes/fgfghfgh~!@@SSD	{ "msg": "Failed" }	{ "msg": "Failed" }	Pass
3.	Valid token with valid value in url http://localhost:4000/api/client/list_of_targetCabinetboxes/2	{ "rows": [{ "box_id": X }] }	{ "rows": [{ "box_id": 12 }, { "box_id": 15 }, { "box_id": 16 }, { "box_id": 17 }, { "box_id": 19 }, { "box_id": 20 }] }	Pass
4.	Valid token with valid value in url but the id is from the current cabinet id http://localhost:4000/api/client/list_of_targetCabinetboxes/1	{ "rows": [] }	{ "rows": [] }	Pass

Test Scenario ID	Api test 12 list_of_facefile_in_allocatedBox
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database. 2. When the specific set of cabinet boxes contain value in ‘faceregfilename’ column.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token	[{ "facereg_filename": "User.1." }, { "facereg_filename": "User.45." }] / OR []	[{ "facereg_filename": "User.1." }, { "facereg_filename": "User.45." }]	Pass
3.	Using valid token but provide random JSON value {"localbid":"!@#!@#","targetbid":"~~d!@□"}	[{ "facereg_filename": "User.1." }, { "facereg_filename": "User.45." }] / OR []	[{ "facereg_filename": "User.1." }, { "facereg_filename": "User.45." }]	Pass

Test Scenario ID	Api test 13 confirm_payment
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	valid token { "customerEmail": "meiwe@netmail.ccc",	Show invalid msg	{ "customerPhone": "Phone number must be 10 digit long", "storingDay":	Pass

	"customerPhone":"234", "storingDay":"x", "storingHours":"x", "payAmount":"xx", "localBid":"xx", "targetBid":"xx", "serviceType":"x", "faceScannQr":"xx", "face_recFilename": "x" }		"storingDay is required in digit", "storingHours": "storingHours is required in digits", "payAmount": "payAmount is required in digits", "localBid": "localBid is required", "targetBid": "targetBid is required to be number", "serviceType": "serviceType is required either T_S or L_S", "faceScannQr": "faceScannQr is required either FnQR or QR" }	
3.	valid token { "customerEmail": "meiwe@netmail.ccc", "customerPhone":"23%%%%%%%%4", "storingDay":"-1", "storingHours":"-1", "payAmount":"-9", "localBid":"-1", "targetBid":"-1", "serviceType":"-1", "faceScannQr":"-1", "face_recFilename": "-1" }	Show invalid msg	{ "customerPhone": "Phone number must be 10 digit long", "storingDay": "storingDay is required in digit", "storingHours": "storingHours is required in digits", "payAmount": "payAmount is required in digits", "serviceType": "serviceType is required either T_S or L_S", "faceScannQr": "faceScannQr is required either FnQR or QR" }	Pass
4.	valid token { "customerEmail": "meiwe@n####c", "customerPhone":"2123123123123123123", "storingDay":"3333.2", "storingHours":"3333.33", "payAmount":"23432.34", "localBid":"222", "targetBid":"111", "serviceType":"L_S", "faceScannQr":"FnQR", "face_recFilename": "-1" }	Show invalid msg	{ "customerEmail": "customerEmail is required(in right format)", "customerPhone": "Phone number must be 10 digit long" }	Acceptable
5.	valid token { "customerEmail":	Show invalid msg	{ "customerEmail": "customerEmail is	Pass

	"□e@n####c", "customerPhone":"□", "storingDay":"□", "storingHours":"□", "payAmount":"□", "localBid":"□", "targetBid":"□", "serviceType":"T_S", "faceScannQr":"QR", "face_recFilename": "□" }		required(in right format)", "customerPhone": "Phone number must be 10 digit long", "storingDay": "storingDay is required in digit", "storingHours": "storingHours is required in digits", "payAmount": "payAmount is required in digits", "localBid": "localBid is required", "targetBid": "targetBid is required to be number" }	
6.	valid token { "customerEmail": "□e@n بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ@gmail.com", "customerPhone":"□ بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "storingDay":"□ بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "storingHours":"□ بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "payAmount":"□ بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "localBid":"□ بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "targetBid":" بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ□", "serviceType":" بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "faceScannQr":"Q بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ", "face_recFilename": " بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ" }	Show invalid msg	{ "customerEmail": "customerEmail is required(in right format)", "customerPhone": "Phone number must be 10 digit long", "storingDay": "storingDay is required in digit", "storingHours": "storingHours is required in digits", "payAmount": "payAmount is required in digits", "localBid": "localBid is required", "targetBid": "targetBid is required to be number", "serviceType": "serviceType is required either T_S or L_S", "faceScannQr": "faceScannQr is required either FnQR or QR" }	Pass
7.	valid token { "customerEmail": "mddddd*&@gmail.coml.ccc", "customerPhone":"234", "storingDay":"234", "storingHours":"123", "payAmount":"34", "localBid":"3",	Show invalid msg	{ "customerPhone": "Phone number must be 10 digit long", "serviceType": "serviceType is required either T_S or L_S", "faceScannQr": "faceScannQr is required either FnQR or QR" }	

	"targetBid":"3", "serviceType":"T_s", "faceScannQr":"qr", "face_recFilename": "x" }			
8.	valid token { "customerEmail": "mddddd*&@gmail.coml.ccc", "customerPhone":"0234123123", "storingDay":"234", "storingHours":"123", "payAmount":"34", "localBid":"3", "targetBid":"3", "serviceType":"T_S", "faceScannQr":"FnQR", "face_recFilename": "x" }		{}	Pass

Test Scenario ID	Api test 14 cancel_reserved_box
Test Case Description	Testing the backend works stably on returning suitable data.
Prerequisite	1. Server running the backend code and connected with the database.

Tests execution steps

1. Using Postman rest api testing tool

No.	Inputs	Expected Output	Actual Output	Status
1.	Using invalid token	Unauthorized	Unauthorized	Pass
2.	Using valid token but without value	{ "localboxID": "localboxID is required /targetboxID is optional" }	{ "localboxID": "localboxID is required /targetboxID is optional" }	Pass
3.	Using valid token but provide random JSON value {"localbid":"!@#!@#", "targetbid":"~~d!@□" }	{ "localboxID": "localboxID is required /targetboxID is optional" }	{ "localboxID": "localboxID is required /targetboxID is optional" }	Pass

4.	Using valid token and the id of reserved local box id and another local box id which is not reserving {"localboxID":"9","localboxID":"1"}	{}	{}	Pass
5.	Using valid token and the id of reserved local box id {"localboxID":"9"}	{}	{}	Pass
6.	Using valid token and Correct JSON with random value {"localboxID":"}vfvvf"}	{ "localboxID": "localboxID must be number" }	{ "localboxID": "localboxID must be number" }	Pass
7.	Using valid token and {"localboxID":"91","targetboxID":"29s"}	{ "targetboxID": "targetboxID must be number" }	{ "targetboxID": "targetboxID must be number" }	Pass
8.	Using valid token and {"localboxID":"-91","targetboxID":"-29"}	{ "localboxID": "localboxID must be number", "targetboxID": "targetboxID must be number" }	{ "localboxID": "localboxID must be number", "targetboxID": "targetboxID must be number" }	Pass
9.	Using valid token and Valid reserved localbox id and target box id {"localboxID":"9","targetboxID":"29"}	{}	{}	Pass
10.	Using valid token and Valid reserved localbox id and non reserved target box id {"localboxID":"9","targetboxID":"29"}	{}	{}	Pass
11.	{"sid":"&@$","cqr":"@##@*@", "tid":"11", "bqr":"12","tid":"qqq", "aqr":"was"}	Unlocking Failed	Unlocking Failed	Pass

APPENDIX D – USER INTERFACE DESIGN

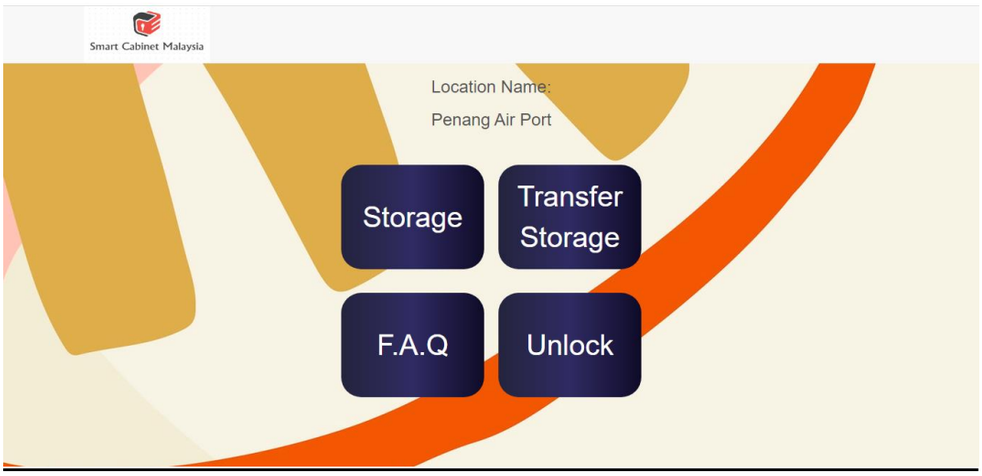


Figure 1 Main page of the user interface

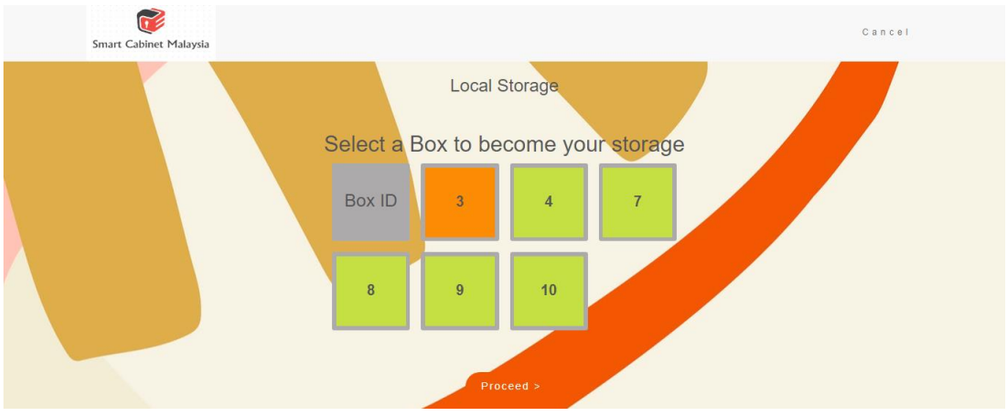


Figure 2.1 Select as local storing.

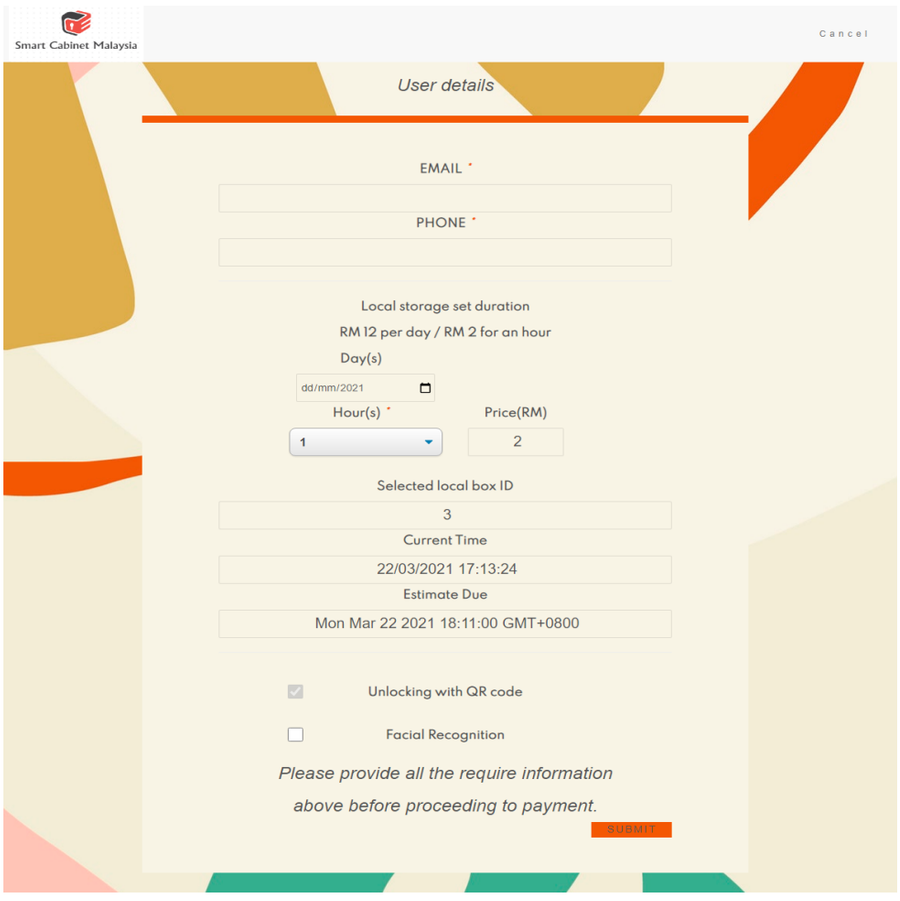


Figure 2.2 Form for local storing.

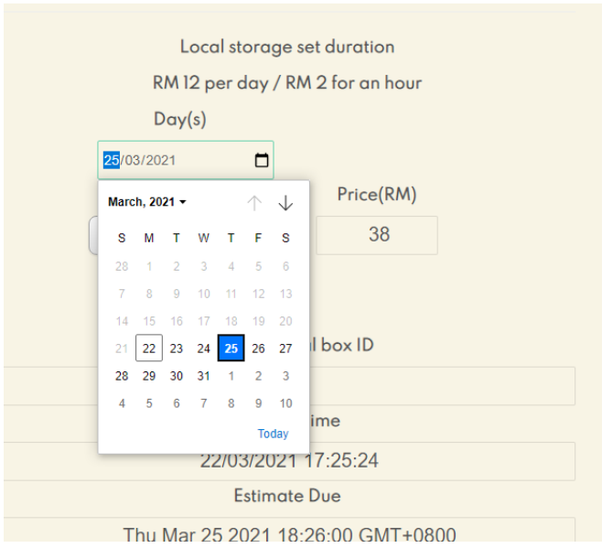


Figure 2.3 Form for local storing date selection.

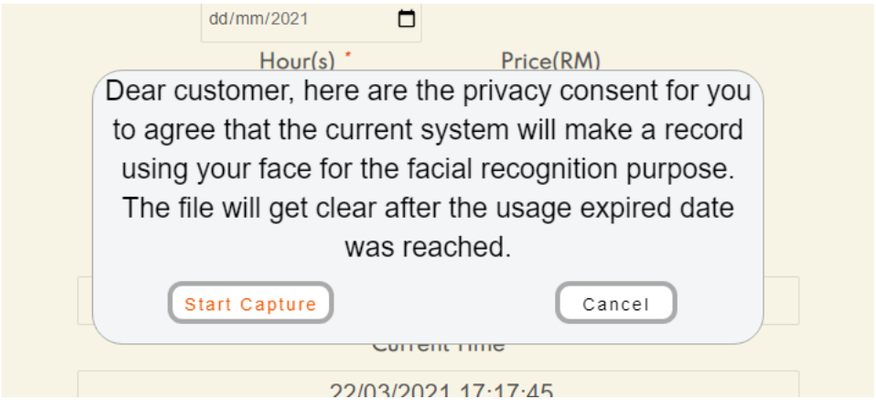


Figure 2.4 Form for local storing asking for facial capturing.

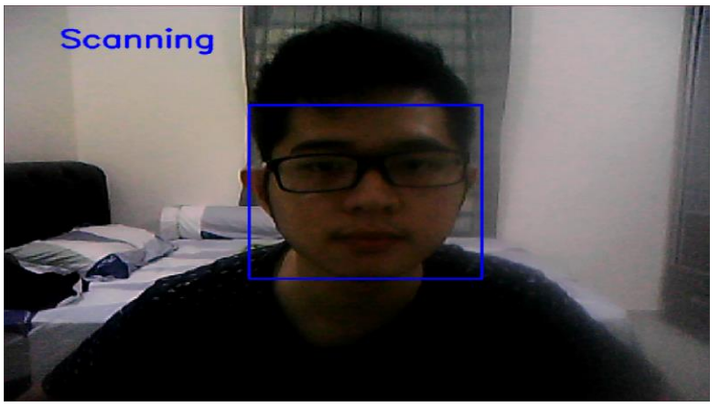


Figure 2.5 Facial scanning

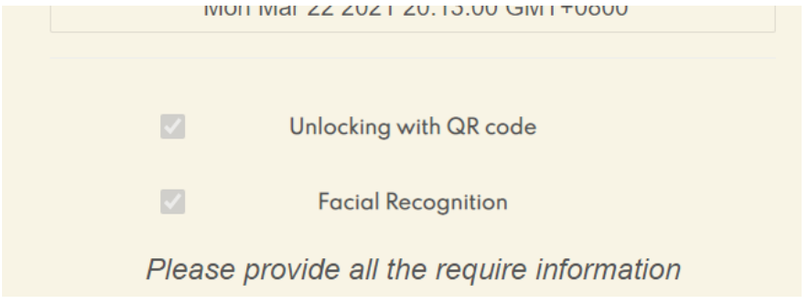


Figure 2.6 Checkbox was selected for after scanned face.

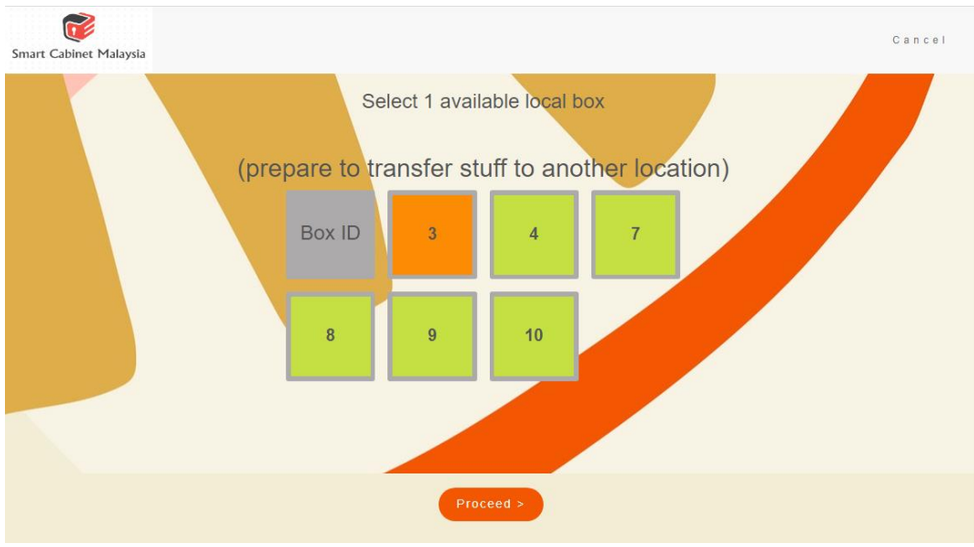


Figure 3.1 Select as transfer storing(select local box).

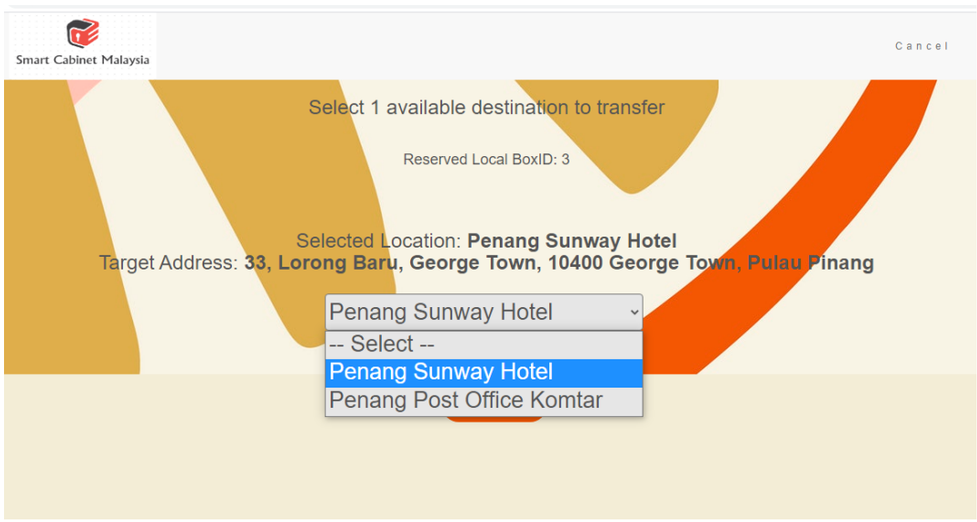


Figure 3.2 Select a target destination.

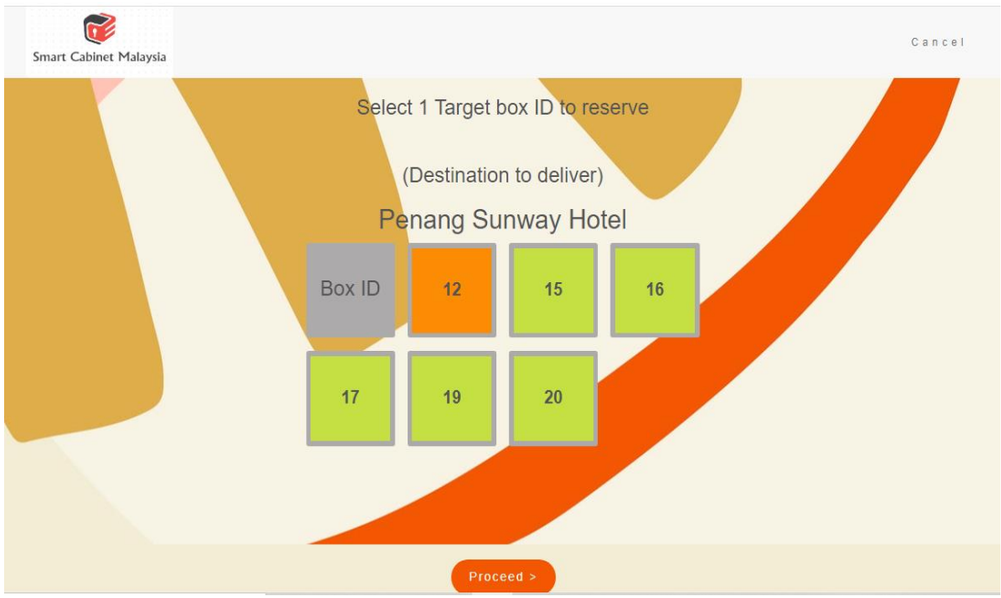


Figure 3.3 Select box id for reserve on target cabinet.

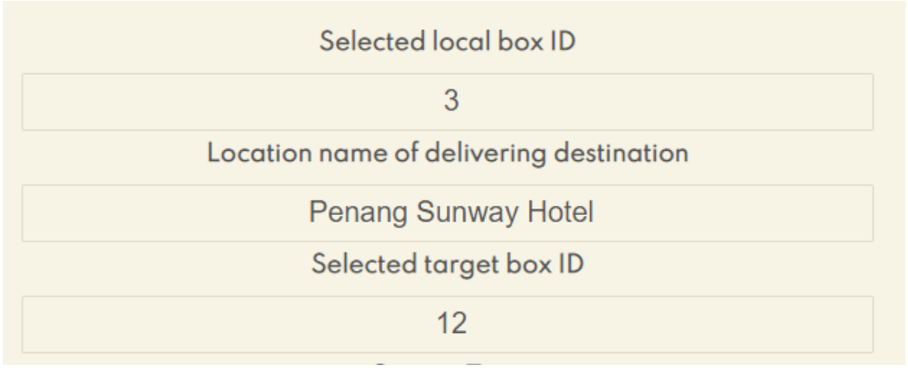


Figure 3.4 Section of form for transfer storing.

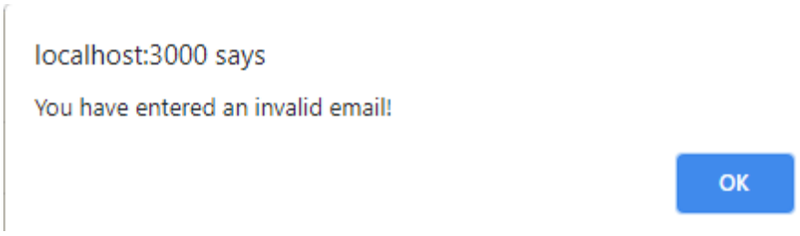


Figure 3.5 Alert dialog.

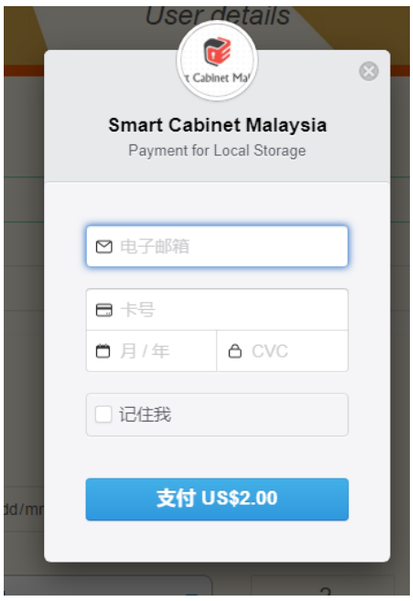


Figure 3.6 Stripe Payment API

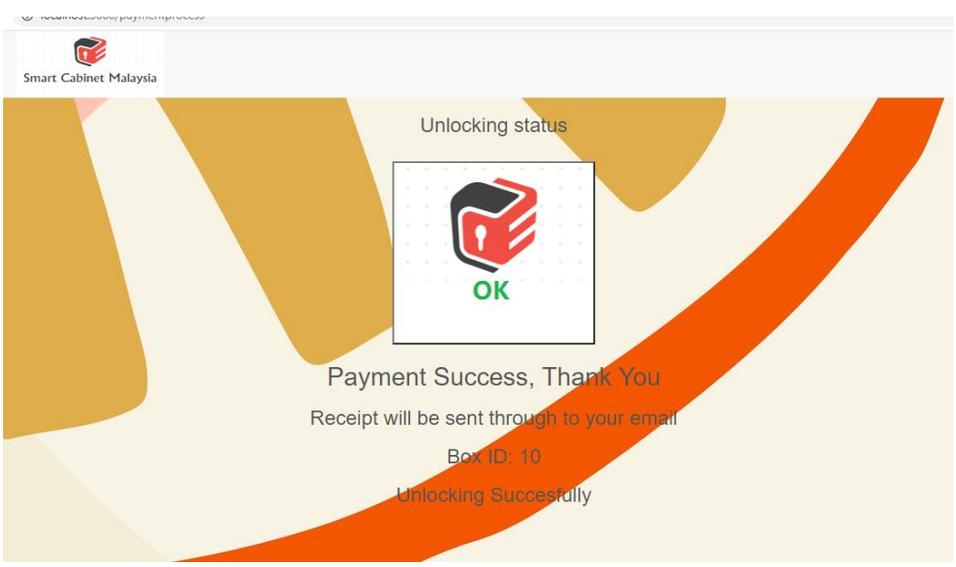


Figure 3.7 Page for after payment succeeds.

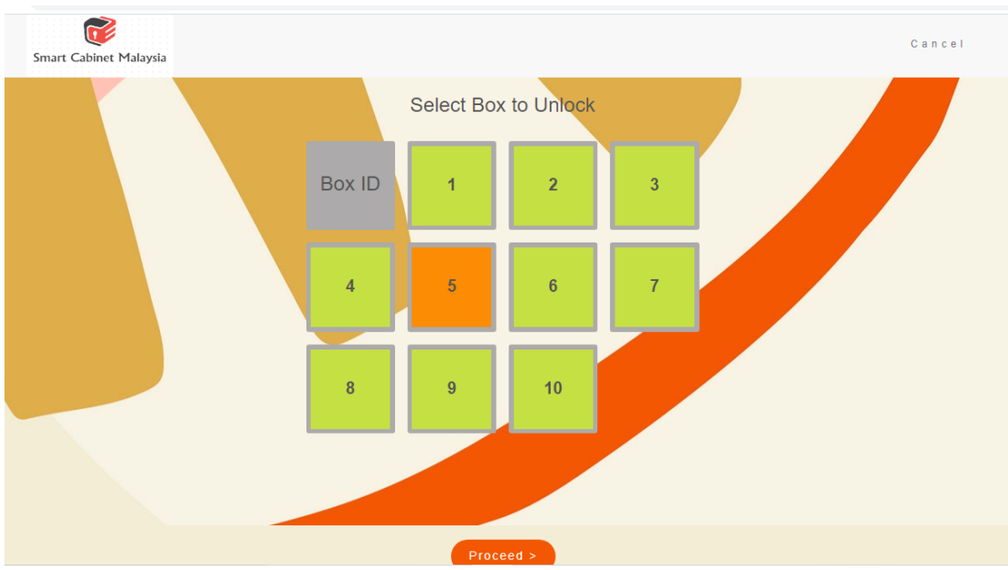


Figure 4.1 Select a box id from the local cabinet set to unlock.

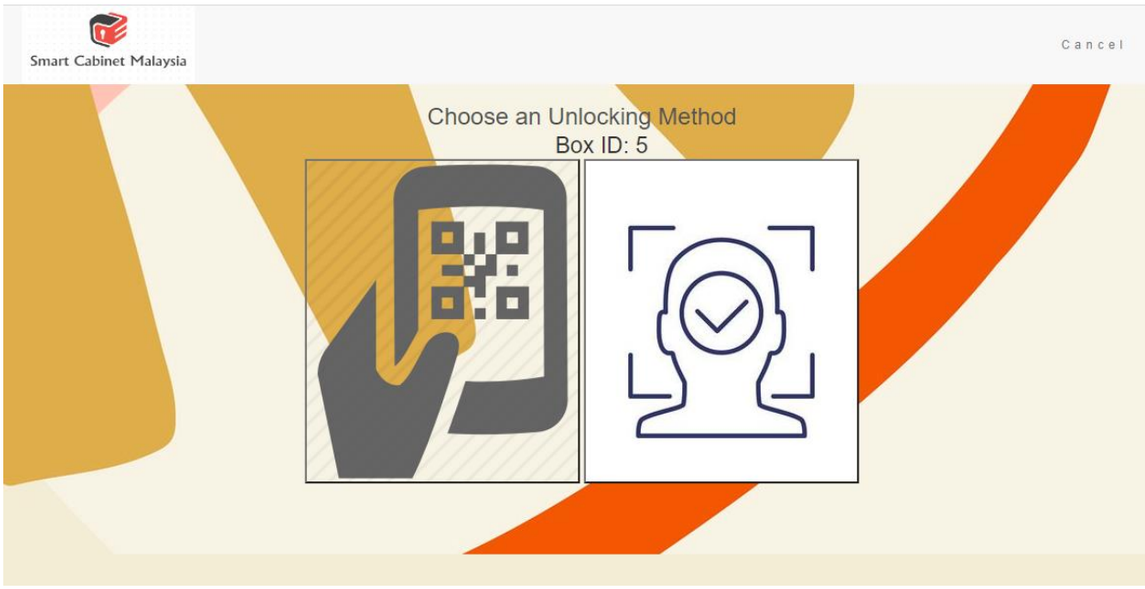


Figure 4.2 Select an unlocking method, either QR or face recognition.

Figure 4.3 Qr code scanner.

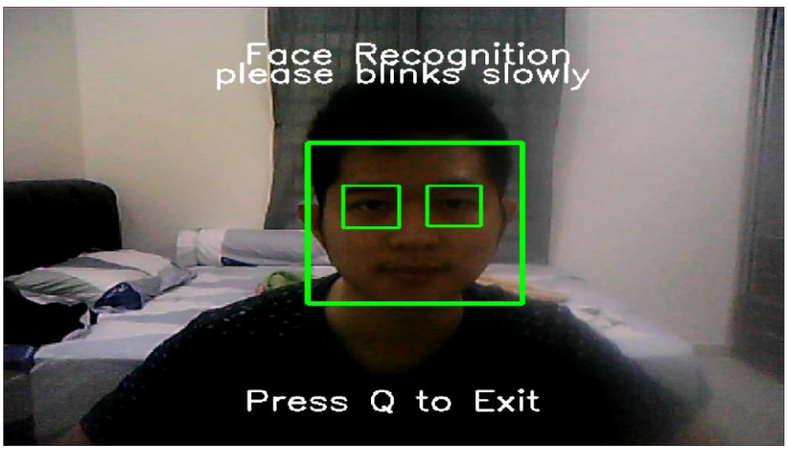


Figure 4.4 Facial recognition scanner.

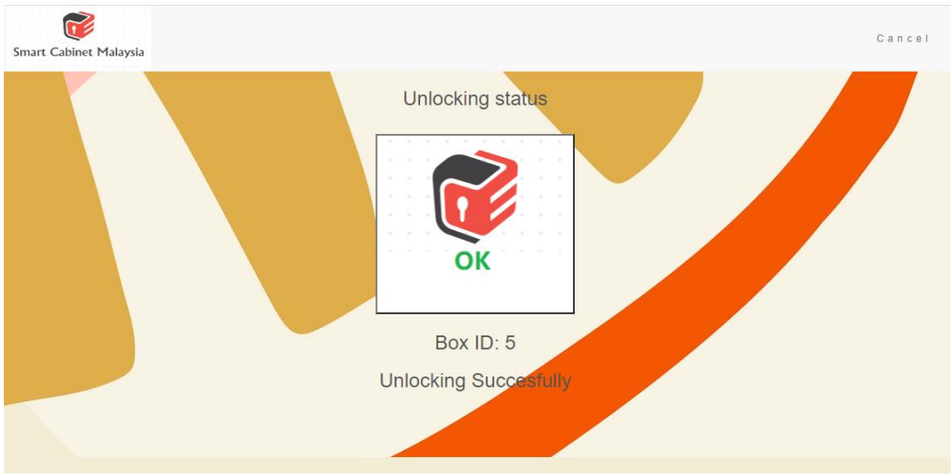


Figure 5 Unlocking success.

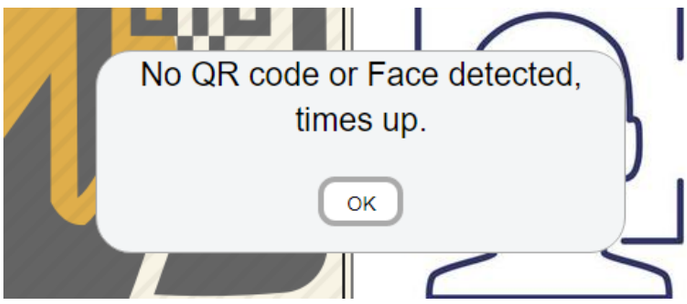


Figure 5.1 Scanning times up.

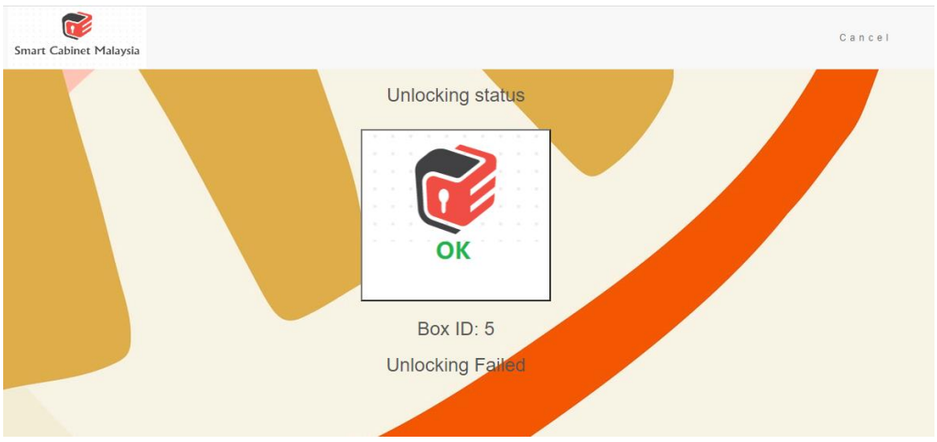


Figure 5.2 Unlocking Failed

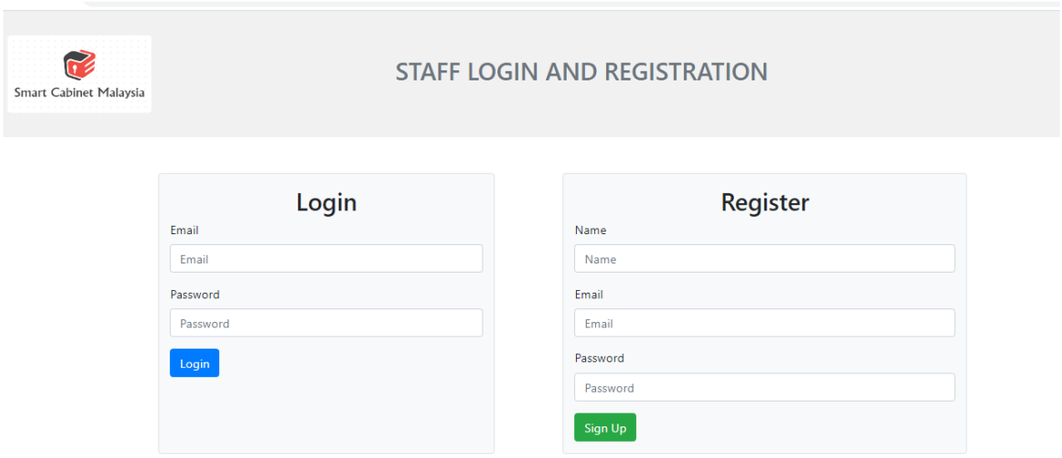


Figure 7.1 Staff portal for login and register.

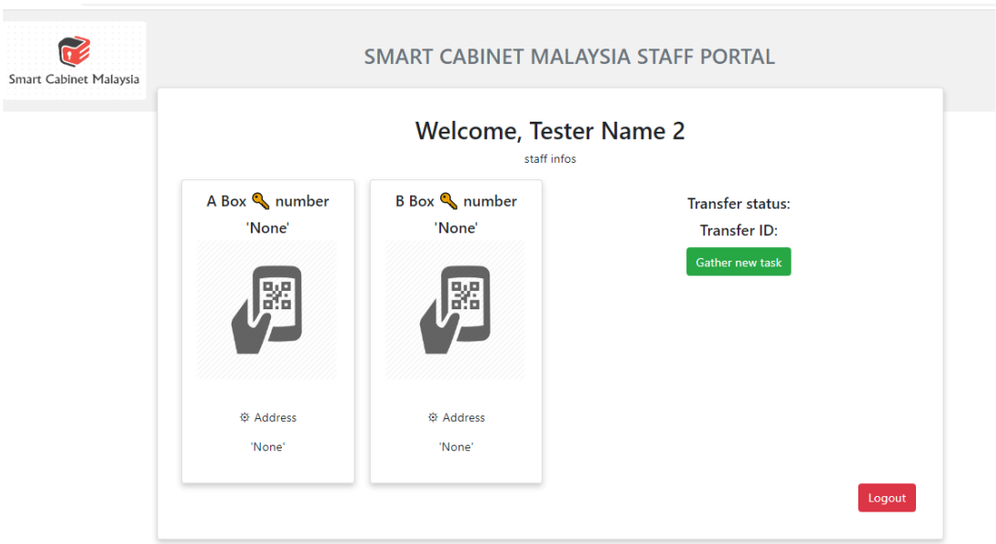


Figure 7.2 New registered staff portal after login.

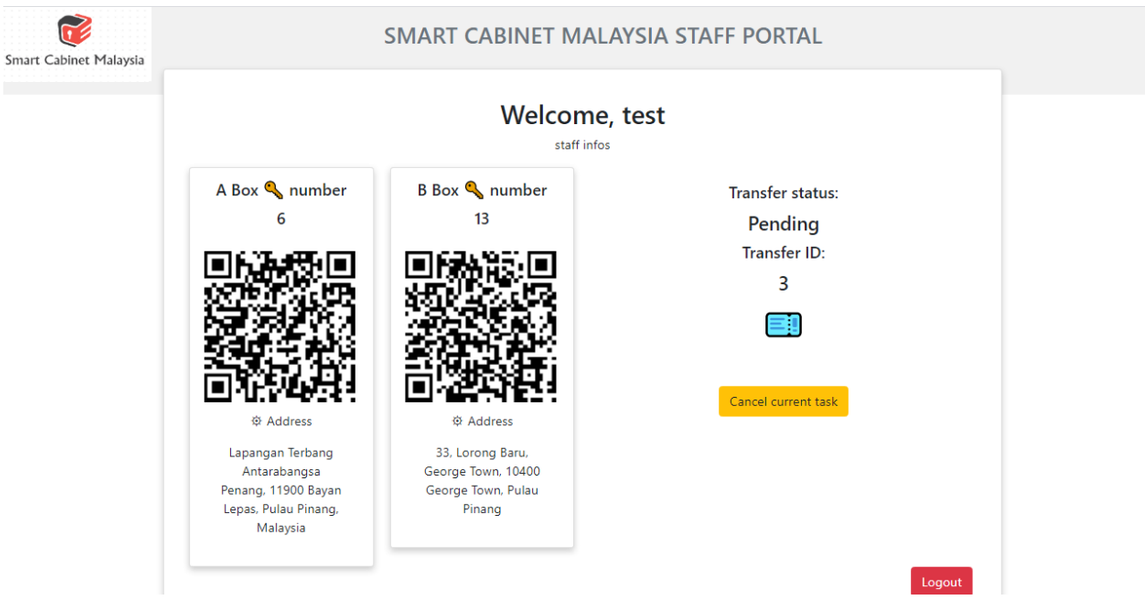


Figure 7.3 Staff portal after login and gathered tasks.

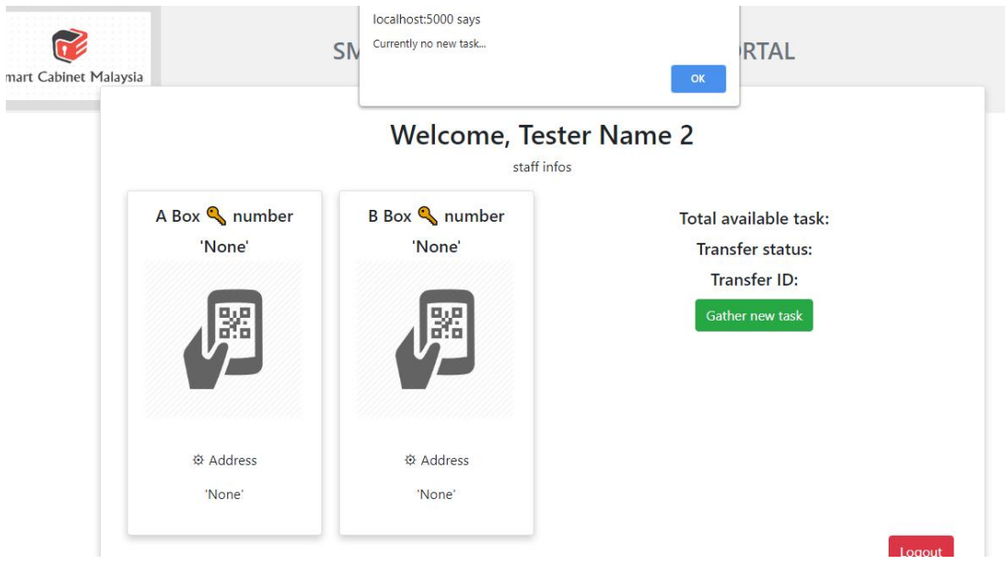


Figure 7.4 Staff portal if there were no new transfer task found.

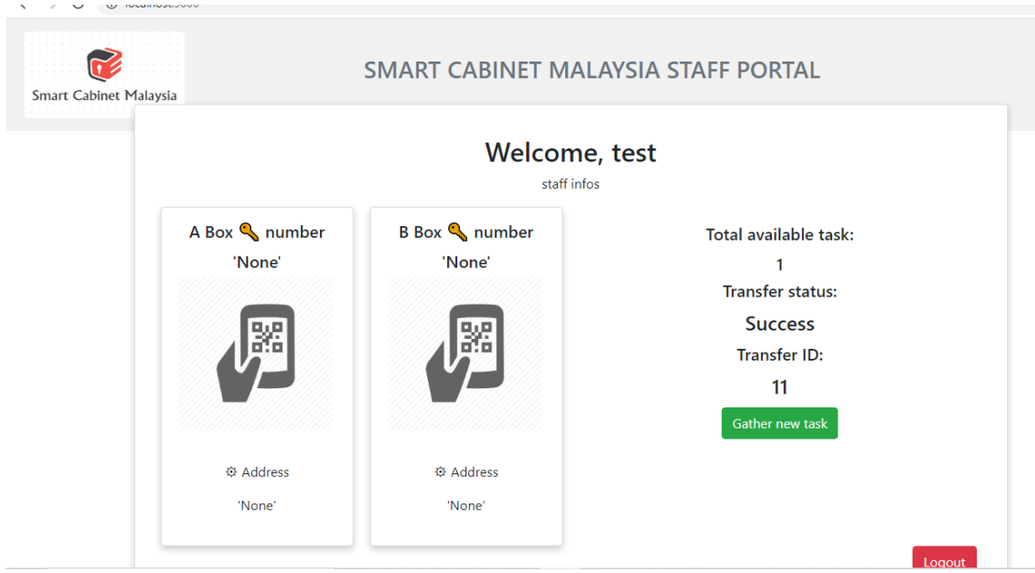


Figure 7.4 Staff portal after staff have placed parcel to 'B Location'(Scanned QR).

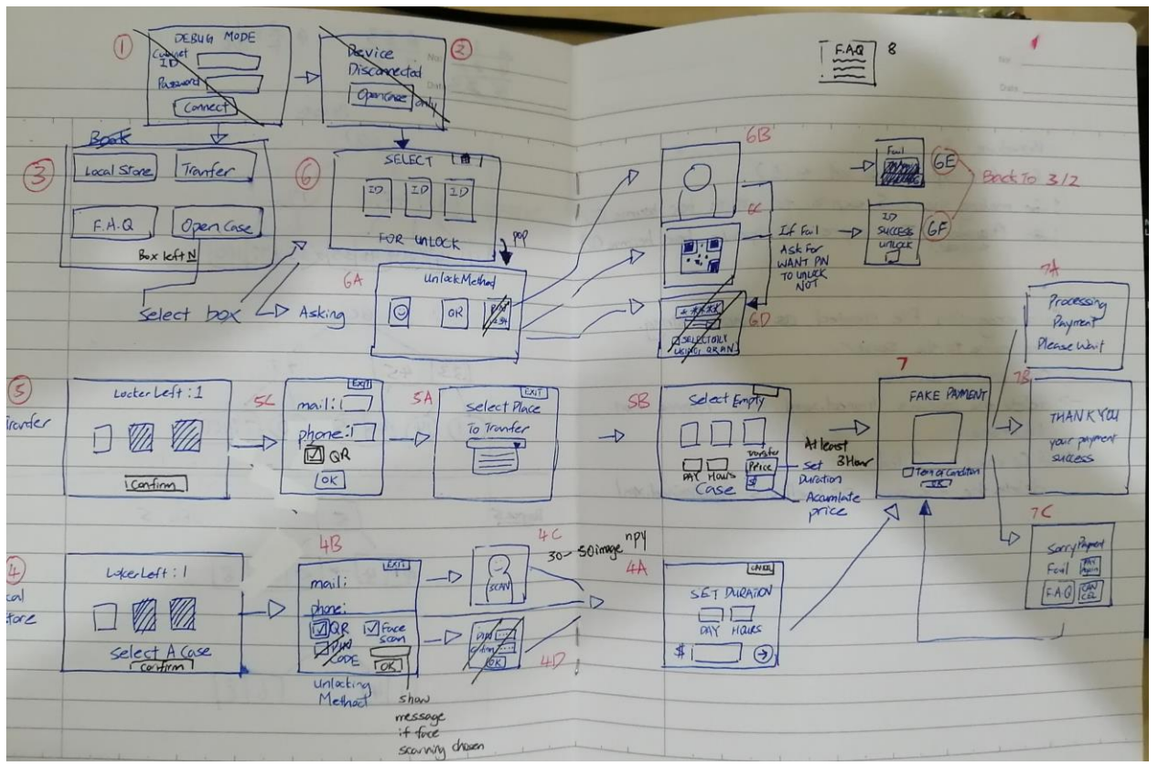
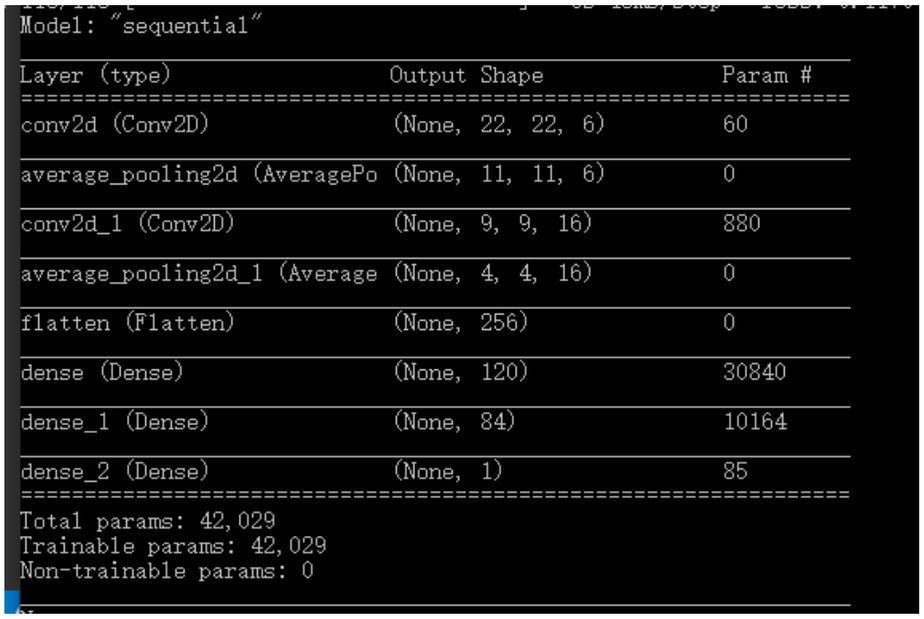


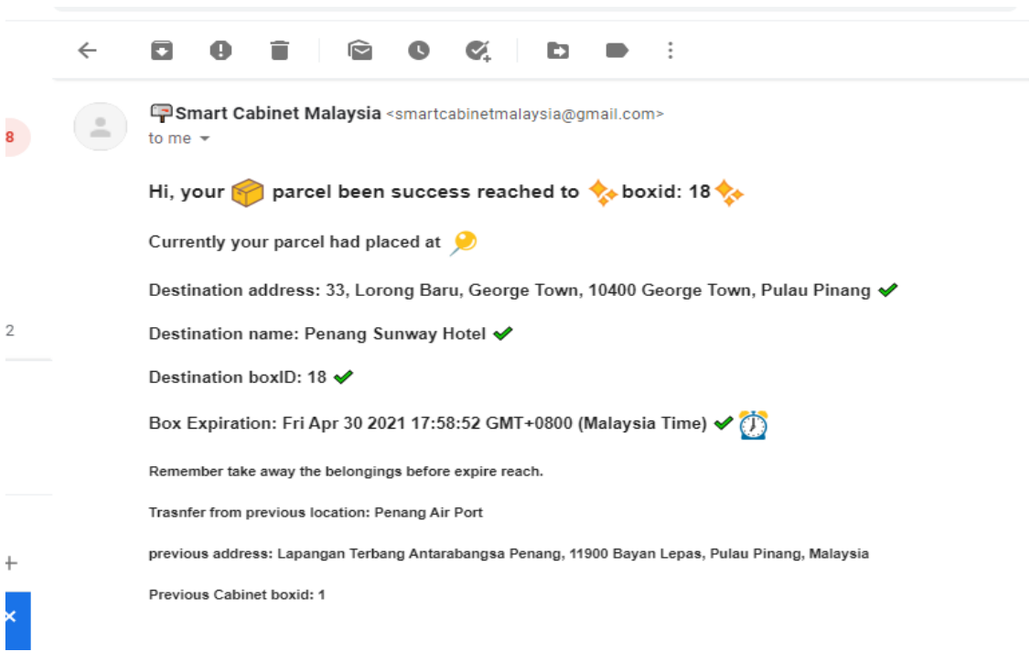
Figure 8 Design sketch of clientware interface



```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 22, 22, 6)         60
_________________________________________________________________
average_pooling2d (AveragePo (None, 11, 11, 6)         0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 9, 9, 16)          880
_________________________________________________________________
average_pooling2d_1 (Average (None, 4, 4, 16)          0
_________________________________________________________________
flatten (Flatten)            (None, 256)               0
_________________________________________________________________
dense (Dense)                (None, 120)               30840
_________________________________________________________________
dense_1 (Dense)              (None, 84)                10164
_________________________________________________________________
dense_2 (Dense)              (None, 1)                 85
=================================================================
Total params: 42,029
Trainable params: 42,029
Non-trainable params: 0
_________________________________________________________________
```

Figure 9 CNN for predicting open or close eyes.

APPENDIX E



Smart Cabinet Malaysia <smartcabinetmalaysia@gmail.com>
to me

Hi, your parcel been success reached to boxid: 18

Currently your parcel had placed at

Destination address: 33, Lorong Baru, George Town, 10400 George Town, Pulau Pinang

Destination name: Penang Sunway Hotel

Destination boxID: 18

Box Expiration: Fri Apr 30 2021 17:58:52 GMT+0800 (Malaysia Time)

Remember take away the belongings before expire reach.

Trasnfer from previous location: Penang Air Port

previous address: Lapangan Terbang Antarabangsa Penang, 11900 Bayan Lepas, Pulau Pinang, Malaysia

Previous Cabinet boxid: 1

Appendix E_a Email message after the delivery was successfully context will sent from backend to the customer email address.

Appendix E_b client ware EJS Files

```
function populateSelect() {
    // THE JSON ARRAY.

  fetch('http://localhost:3000/selectboxtounlock_list')
  .then(res => res.json())
  .then((out) => {
      console.log('Output: ', out);
      var birds = out;
      var ele = document.getElementById('cellInject');
      for (var i = 0; i < birds.length; i++) {
      // POPULATE SELECT ELEMENT WITH JSON.
      ele.innerHTML = ele.innerHTML +
      '<div class="grid-cell" ><input onClick="myRadiobtn()" type="radio" form="myform" name="boxid" value="' +
        birds[i]['box_id'] + '" id="' + 'r'+birds[i]['box_id'] +
        '" /><label class="radio" for="' + 'r'+birds[i]['box_id'] + '"><p class="innerText">' +
         birds[i]['box_id'] + '</p></label></div>';
      }
  }).catch(err => console.error(err));

}
populateSelect();
```

Figure 2.1 Render list of boxes using javascript to add HTML content.

```
<div class="container-fluid bg-3 text-center">
  <h3><%= page %></h3><br>
  <form action="/selected_box_to_unlock" name="myform" method="post" id="myform"></form>
  <div class="game-container">
    <div class="grid-container radios" id="cellInject">
      <div class="grid-cell">
          <span><h3 id="box_idtext">Box ID</h3></span>
      </div>
    </div>
  </div>
  <button type="submit" form="myform"  class="btn2" id="mySubmit"> Proceed > </Button>

</div>
```

Appendix E_b Figure 2.2

```
<% include partials/script %>
```

Appendix E_b Figure 2.3

```
<% include partials/head %>
</head>
<body>
<% include partials/menu %>
```

Appendix E_b Figure 2.4

```
var phoneno = /^\d{10}$/;
var mailformat = /^[a-zA-Z0-9.!#$%&'*+/=?^_`{|}~-]+@[a-zA-Z0-9-]+(?:\.[a-zA-Z0-9-]+)*$/;
function ValidateEmail(inputText, customerPhoneNumber){

    if(inputText.value.match(mailformat) && customerPhoneNumber.value.match(phoneno)){
        /*alert("Valid email address!");
        document.myform.customerEmail.focus();*/
        document.getElementById("form_submit").disabled = false;
        return true;
    } else{
        if (!customerPhoneNumber.value.match(phoneno) || !inputText.value.match(mailformat))
            if (!inputText.value.match(mailformat)){
                alert("You have entered an invalid email!");
                document.myform.customerEmail.focus();
                document.getElementById("form_submit").disabled = true;
            }
            if (!customerPhoneNumber.value.match(phoneno)){
                document.myform.customerPhone.focus();
                alert("You have entered an invalid phone !");
                document.getElementById("form_submit").disabled = true;
            }
        }

        return false;
    }
}
```

Appendix E_b Figure 2.5 Form validation

```
function populateSelect() {
    // THE JSON ARRAY.
    fetch('http://localhost:3000/transfstore_selectplace_list')
    .then(res => res.json())
    .then((out) => {
        console.log('Output: ', out);
        var birds = out;
        var ele = document.getElementById('sel');
        for (var i = 0; i < birds.length; i++) {
        // POPULATE SELECT ELEMENT WITH JSON.
        ele.innerHTML = ele.innerHTML +
        '<option id="location_id" value="' + birds[i]['cabinet_id'] + '" data-addr="'
        + birds[i]['cabinet_addr']+'">'  + birds[i]['location_name'] + '</option>';
    }
    }).catch(err => console.error(err));
}
```

Appendix E_b Figure 2.6

```
<div class="container-fluid bg-3 text-center">    <% var boxidvalue = (typeof selectedbid !='undef
  <form action="/unlockingwithQR" name="myform" method="post" id="myform"></form>
    <input type="hidden" id="location_id" form="myform" name="boxid" value="<%=boxidvalue%>">
  <form action="/unlockingwithFace" name="myform2" method="post" id="myform2"></form>
    <input type="hidden" id="location_id2" form="myform2" name="boxid" value="<%=boxidvalue%>">
  <h3><%= page %></h3><br>
  <button type="submit" class="btn1" form="myform" > </Button>
    <% //onclick="document.location='/unlockingwithQR/ =boxidvalue '" %>
    <div id="confirm">
      <div class="message">No QR code or Face detected, times up.</div><br>
      <button class="yes">OK</button>
    </div>
```

Appendix E_b Figure 2.7 Hidden value in HTML form

```ejs
<div class="container-fluid bg-3 text-center">
  <h3><%= page %></h3><br>
  <button type="submit" onclick="window.location.href='/'" class="btn1"> </Button>
    <%var paymstatus =  (typeof showPaymentStatus !== 'undefined') ? showPaymentStatus : "X"%>
    <% if (paymstatus == "Yes"){%>
    <h2>Payment Success, Thank You</h2>
    <h3>Receipt will be sent through to your email</h3>
    <%}%>
    <% if (paymstatus == "No"){%>
      <h2>Sorry your payment has failed</h2>
      <h3>Due to times up reasons or error.</h3>
    <%}%>

    <%var box_IDx =  (typeof selectedbid !== 'undefined') ? selectedbid : "X"%>
    <h3>Box ID: <%=box_IDx %></h3>
    <%var varfail = (typeof failed !== 'undefined') ? failed : "fail";%>
    <% if (varfail == "success"){%>
    <h3>Unlocking Succesfully</h3>
    <%}else{%>
      <h3>Unlocking Failed</h3>
    <%}%>
</div>
```

Appendix E_b Figure 2.8 EJS syntax

```ejs
<script
src="https://checkout.stripe.com/checkout.js" class="stripe-button"
data-key="<%=stripePublishableKey%>"
data-amount="100"
data-name="Smart Cabinet Malaysia"
data-description="Payment for Local Storage"
data-image="scmlogo.png"
data-locale="auto"
data-panel-label='Checkout'>
</script>
```

Appendix E_b Figure 2.9 Strpe settings

Appendix E_b Figure 2.10 FAQ page contain user guide for smart cabinet system.