A Comparison between The Effectiveness of Face Detection System with Machine Learning (ML) Algorithms using Software Automation

1.0 Introduction

1.1 Background

QB is an established company within Southeast Asia that focuses on deploying and developing queue management systems for clients. With the continuous development of the system, the test scripts for software testers have grown longer over the period and have become a tedious and time-consuming task. It begs the question of whether employing software testers to execute the test scripts would be an efficient solution to the problem. Although the overarching problem would be to develop automation of test scripts to improve efficiency, the group has decided to scope down one of the test scripts which involves Artificial Intelligence (AI). Within the scope of AI, the group has decided to determine the reliability of facial detection within the queue management system. With the overarching aim of developing the test script for Queue Bee, the group has to first determine the accuracy of the facial detection system.

1.2 Motivation for Solution Development

The group currently acts as a consulting company providing automation and AI services. The current research required the group to determine the effectiveness and accuracy of the QB facial detection system. By establishing a benchmark, it enables the comparison of the effectiveness and accuracy of the client’s algorithms in comparison with the facial detection algorithms developed by the model. The overarching goal would be to build a foundation for facial recognition by detecting AI-generated images. Moreover, with the client facial detection currently operating as a mature product within Southeast Asia, the vision and mission would be to facilitate incremental change to the current system. Therefore, the current research only explores the comparison between the algorithms as an advisor to the client.

1.3 Significant contribution

1.3.1 Developing software testing via automation (Selenium).

The traditional route of executing software testing would be conducted manually. Despite that, human error continuously occurs, whether it is fatigue or accidental or repetitive menial tasks. Therefore, Selenium was introduced to improve the efficiency and productivity of QB's software testing process. In engaging with the client, it was found that there were more than 3000 test scripts to be executed manually. It was highlighted that QB is unable to execute every test script, resulting in an inconsistent system to fully evaluate the quality assurance of QB's system interface. By introducing Selenium, the group significantly contributed to developing software testing via automation, reducing cost and errors, while increasing efficiency, productivity and reliability.

1.3.2 Identifying the optimized algorithm for facial detection.

The research facilitates identifying various image detection algorithms within the machine learning industry. With careful consideration, Convolutional Neural Networks (CNN) was selected as the optimized algorithms for image detection from a literature review perspective. To strengthen the research, the algorithm was tested against the benchmark (QB's system), determining the overall performance.

1.3.3 Stress testing the algorithms on various classes in the Teachable machine, observing gaps between algorithm's performances.

The current research evaluates the gaps in QB's facial detection system, it categorizes the test set into several categories. Enabling the researcher to identify the algorithm's weak point. It contributes to allowing the group to advise the gap in the QB's system.

2.0 Literature Review

2.1 Face detection technology

Face detection is a crucial first step in many computer vision applications, including face detection, surveillance systems, human-computer interaction, and image or video analysis. It involves identifying the presence and location of human faces within an image or video frame. As of now, there are two main approaches when it comes to face detection, which are the traditional approach and the machine learning-based approach.

2.1.1 Traditional Approaches

Traditional face detection methods relied heavily on handcrafted features and classifiers. Techniques such as Viola-Jones Haar cascade classifiers (Viola & Jones, 2004) were widely used due to their simplicity and computational efficiency. However, these methods often struggled with variations in lighting conditions, pose, and occlusions, leading to suboptimal performance in real-world scenarios.

2.1.2 Machine Learning-Based Approaches

- Support Vector Machines (SVM):

SVMs have been employed for face detection by learning discriminative features from labeled data. SVM-based approaches often combine multiple classifiers trained on various features like Histogram of Oriented Gradients or Local Binary Patterns (Dalal & Triggs, 2005). While SVMs have shown reasonable performance, they may lack the capability to learn complex patterns in high-dimensional feature spaces.

- Convolutional Neural Networks (CNN):

CNNs have revolutionized face detection due to their ability to automatically learn hierarchical features from raw pixel data. Models like the Convolutional Single Shot Multibox Detector (Liu et. al, 2016) and the Region-based Fully Convolutional Networks (Dai et. al, 2016) have demonstrated remarkable accuracy in detecting faces across different scales and orientations.

Moreover, architectures like the Faster R-CNN (Ren, 2015) have improved efficiency by integrating region proposal networks with CNNs.

- Deep Learning Architectures:

Beyond CNNs, deeper architectures such as Residual Networks (He et. al, 2016) and Inception (Szegedy et. al, 2016) have been utilized for face detection tasks. These architectures leverage skip connections, inception modules, or neural architecture searches to enhance feature representation and classification accuracy.

3.0 Problem Statement

- Lack of an automated software testing system for face detection

The main problem we are addressing is to understand the effectiveness and reliability of QB's face detection system. As mentioned in the above literature review, face detection in general is still yet to be a mature technology. Although most industries have implemented face detection systems for daily usage, system errors still occur from time to time. For example, as pointed out by Zhou et. al, a low-quality image or a blurry face image would affect the accuracy of the face detection system. Thus, it is important to perform software testing on the face detection system before deploying it for public usage.

Upon discussion with the company, it was found that the information technology department had a total of over 3000 test scripts currently implemented. The current process would be to manually execute the test scripts one at a time upon every patch implemented in the system, which leads to inefficiency and the desire from the company to automate the entire process.

3.1 Research aim and objectives

This section outlines the aim and objectives of this research by adopting the SMART methodology, which stands for "Specific, Measurable, Achievable, Relevant, and Time-based".

Research aim	To assess the effectiveness and reliability of face detection system
Research objective 1	To deploy an automated software testing system to test the face detection system
Research objective 2	To implement a Machine Learning program in detecting human faces and non-human faces
Research objective 3	To compare the result of facial detection between the actual face detection system and the Machine Learning model

3.2 Research scope / Problem scope

3.2.1 Research Questions

1. Which automation tools would be used to automate the software testing process for QB’s facial detection system?
2. Which facial detection algorithms are applied to develop the model?
3. What is the efficiency difference between automation and manual testing?

3.2.2 Scope Limitations

- QB does not provide access to back-end software.

Based on the conversation between the client and the group, the client only provides access to a web portal demo webpage. The portal only provided the necessary access for the web page to detect human faces. Therefore, no further research would be conducted on QB’s facial detection algorithm.

- Only one test script was developed within the current research

With an estimate of 3000 test scripts, the research was scoped down to one test script. The selection for this particular test script was due to the overlapping nature of AI decision-making and the Capstone project. Thus, the remaining extension of the test scripts will be conducted on the capstone project.

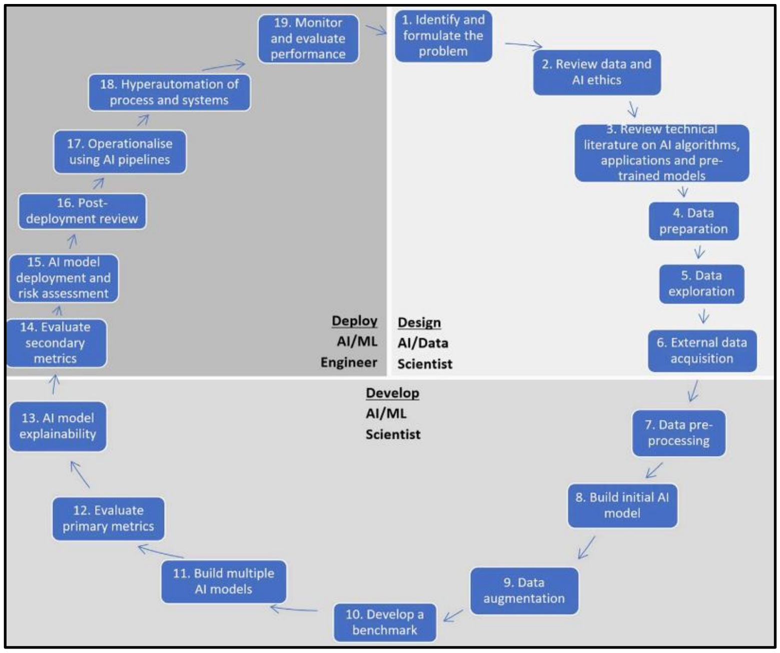
- Deep Fake Image

With the current scope of research, the project excluded deep fake images due to overcomplexity requirements for implementing the algorithm. Therefore, the current scope of excluded deep fake images as the goal of the research would be to determine the effectiveness of facial detection only between the two algorithms.

4.0 Elaboration on the AI Solution

4.1 Solution Methodology Lifecycle

This research adopts one of the most comprehensive AI solution lifecycles for the design, development, and deployment of AI solutions, which is known as the “CDAC AI Life Cycle” (De Silva & Alahakoon, 2022). The lifecycle contains 19 constituent stages, however, this research paper will only focus on the three main phases in general, which are design, develop, and deploy.



1. Identify and formulate the problem
2. Review data and AI ethics
3. Review technical literature on AI algorithms, applications and pre-trained models
4. Data preparation
5. Data exploration
6. External data acquisition
Design
AI/Data
Scientist
7. Data pre-processing
8. Build initial AI model
9. Data augmentation
10. Develop a benchmark
11. Build multiple AI models
12. Evaluate primary metrics
13. AI model explainability
Develop
AI/ML
Scientist
14. Evaluate secondary metrics
15. AI model deployment and risk assessment
16. Post-deployment review
17. Operationalise using AI pipelines
18. Hyperautomation of process and systems
19. Monitor and evaluate performance
Deploy
AI/ML
Engineer

Figure X: The CDAC Life cycle

4.2 Solution Design

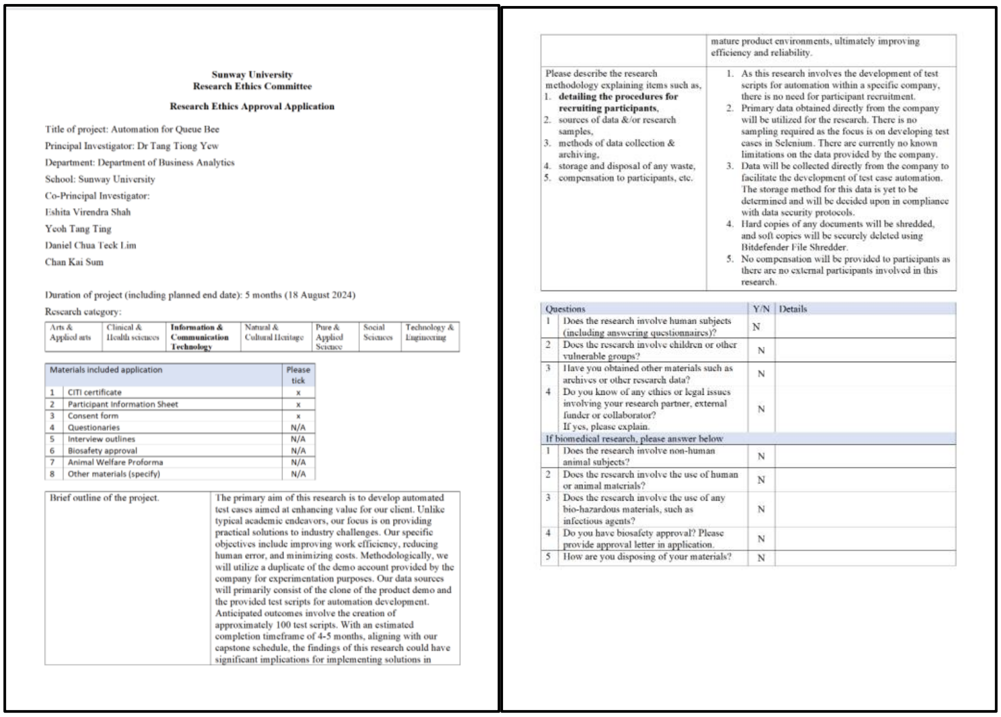
<u>*4.2.1 Identify problem*</u>

As mentioned in 3.0 Problem Statement, the main problem is to identify the effectiveness and reliability of QB’s face detection system.

4.2.2 Review AI ethics

Various research has highlighted the necessity of integrating ethical principles throughout the AI lifecycle to ensure ethical design, implementation, and deployment of AI systems (Zhou et al., 2020). The ethical challenges in AI development have been addressed, emphasizing the need to resolve the social dilemmas that arise from the widespread deployment of AI (Strümke et al., 2021).

To ensure this research adheres to the ethical principles of Artificial Intelligence, an ethical form entitled "SBS Quality Assurance Ethic Form" was submitted to the Research Ethics Committee at Sunway University. The ethical form outlines the research project and research methodology to be carried out during the research. All content written adhered to the guidelines of responsible AI to ensure the research is carried out ethically. Below are the screenshots of the project's ethics form.



Sunway University
Research Ethics Committee

Research Ethics Approval Application

Title of project: Automation for Queue Bee

Principal Investigator: Dr Tang Tiong Yew

Department: Department of Business Analytics

School: Sunway University

Co-Principal Investigator:

Eshita Virendra Shah

Yeoh Tang Ting

Daniel Chua Teck Lim

Chan Kai Sum

Duration of project (including planned end date): 5 months (18 August 2024)

Research category:

Arts & Applied arts	Clinical & Health sciences	**Information & Communication Technology**	Natural & Cultural Heritage	Pure & Applied Science	Social Sciences	Technology & Engineering

Materials included application		Please tick
1	CITI certificate	x
2	Participant Information Sheet	x
3	Consent form	x
4	Questionaries	N/A
5	Interview outlines	N/A
6	Biosafety approval	N/A
7	Animal Welfare Proforma	N/A
8	Other materials (specify)	N/A

Brief outline of the project.	The primary aim of this research is to develop automated test cases aimed at enhancing value for our client. Unlike typical academic endeavors, our focus is on providing practical solutions to industry challenges. Our specific objectives include improving work efficiency, reducing human error, and minimizing costs. Methodologically, we will utilize a duplicate of the demo account provided by the company for experimentation purposes. Our data sources will primarily consist of the clone of the product demo and the provided test scripts for automation development. Anticipated outcomes involve the creation of approximately 100 test scripts. With an estimated completion timeframe of 4-5 months, aligning with our capstone schedule, the findings of this research could have significant implications for implementing solutions in mature product environments, ultimately improving efficiency and reliability.
Please describe the research methodology explaining items such as, 1. **detailing the procedures for recruiting participants**, 2. sources of data &/or research samples, 3. methods of data collection & archiving, 4. storage and disposal of any waste, 5. compensation to participants, etc.	1. As this research involves the development of test scripts for automation within a specific company, there is no need for participant recruitment. 2. Primary data obtained directly from the company will be utilized for the research. There is no sampling required as the focus is on developing test cases in Selenium. There are currently no known limitations on the data provided by the company. 3. Data will be collected directly from the company to facilitate the development of test case automation. The storage method for this data is yet to be determined and will be decided upon in compliance with data security protocols. 4. Hard copies of any documents will be shredded, and soft copies will be securely deleted using Bitdefender File Shredder. 5. No compensation will be provided to participants as there are no external participants involved in this research.

Questions		Y/N	Details
1	Does the research involve human subjects (including answering questionnaires)?	N	
2	Does the research involve children or other vulnerable groups?	N	
3	Have you obtained other materials such as archives or other research data?	N	
4	Do you know of any ethics or legal issues involving your research partner, external funder or collaborator? If yes, please explain.	N	
If biomedical research, please answer below			
1	Does the research involve non-human animal subjects?	N	
2	Does the research involve the use of human or animal materials?	N	
3	Does the research involve the use of any bio-hazardous materials, such as infectious agents?	N	
4	Do you have biosafety approval? Please provide approval letter in application.	N	
5	How are you disposing of your materials?	N	

Figure X: Ethic Application Form

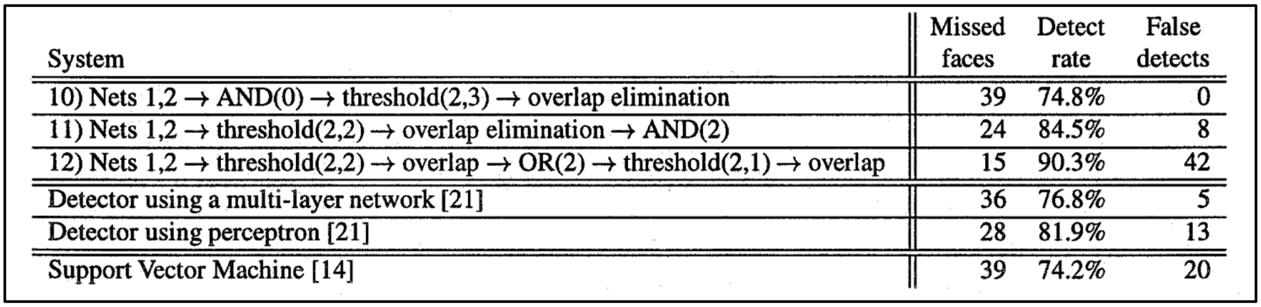
<u>4.2.3 Review technical literature on AI</u>

This section outlines the past studies that had been carried out under similar settings, which in this case is AI face detection. Four studies are being highlighted regarding the usage of different AI models for face detection. This section provides a technical summary based on the four selected papers.

<u>Study 1: Neural Network-based Face Detection by Rowley et. al in 1998</u>

- Summary

This research focused on developing a neural network-based system for detecting upright frontal faces in images. The system utilizes retinal connected neural networks to examine small windows of an image and determine whether each window contains a face. The system can detect between 77.9 percent and 90.3 percent of faces in a set of 130 test images with an acceptable number of false detections.



System	Missed faces	Detect rate	False detects
10) Nets 1,2 → AND(0) → threshold(2,3) → overlap elimination	39	74.8%	0
11) Nets 1,2 → threshold(2,2) → overlap elimination → AND(2)	24	84.5%	8
12) Nets 1,2 → threshold(2,2) → overlap → OR(2) → threshold(2,1) → overlap	15	90.3%	42
Detector using a multi-layer network [21]	36	76.8%	5
Detector using perceptron [21]	28	81.9%	13
Support Vector Machine [14]	39	74.2%	20

Figure X: Comparing results between various Machine Learning algorithms

- Key findings:

Main findings	**Explanations**
Multiple Network Arbitration	The system employs multiple networks to enhance performance over a single network, resulting in improved accuracy in face detection.
Positive Example Alignment	A straightforward procedure for aligning positive face

	examples for training is presented, contributing to effective training of the neural network.
Bootstrap algorithm for negative examples	The use of a bootstrap algorithm for collecting negative examples eliminates the need for manually selecting nonface training examples, thereby streamlining the training process.
Heuristics for improved accuracy	Simple heuristics, such as leveraging the fact that faces rarely overlap in images, are employed to further enhance detection accuracy.

- Limitation:

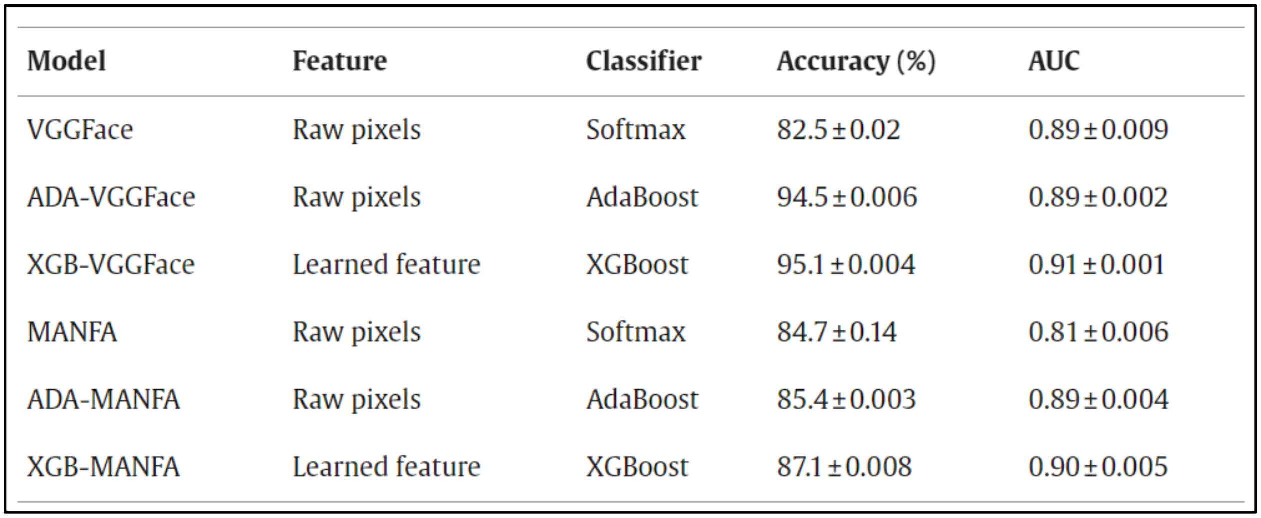
Key point	**Explanation**
Lack of experimenting with multiple facial angles	While the current system is limited to detecting upright faces looking at the camera, future work could involve training separate versions for different head orientations and exploring applications beyond face detection.

Study 2: Face image manipulation detection based on a convolutional neural network by Dang et. al in 2019

- Summary

This research addresses the critical issue of facial image manipulation, the paper introduces a deep learning-based framework for accurately detecting manipulated face images. The framework includes a customized convolutional neural network (MANFA) for feature extraction, a hybrid framework (HF-MANFA) utilizing AdaBoost and XGBoost to handle imbalanced datasets, and a large manually collected and validated dataset for training and testing.

- Results



Model	Feature	Classifier	Accuracy (%)	AUC
VGGFace	Raw pixels	Softmax	82.5 ± 0.02	0.89 ± 0.009
ADA-VGGFace	Raw pixels	AdaBoost	94.5 ± 0.006	0.89 ± 0.002
XGB-VGGFace	Learned feature	XGBoost	95.1 ± 0.004	0.91 ± 0.001
MANFA	Raw pixels	Softmax	84.7 ± 0.14	0.81 ± 0.006
ADA-MANFA	Raw pixels	AdaBoost	85.4 ± 0.003	0.89 ± 0.004
XGB-MANFA	Learned feature	XGBoost	87.1 ± 0.008	0.90 ± 0.005

Figure X: Performance of different models on image detection

The proposed MANFA model, along with its hybrid variations, demonstrates superior performance compared to existing expert and intelligent systems. MANFA achieves an accuracy of 84.7% and an AUC of 0.81, while the hybrid models (ADA-MANFA and XGB-MANFA) outperform MANFA, with XGB-MANFA achieving an accuracy of 87.1% and an AUC of 0.9. Additionally, transfer learning using a pre-trained VGGFace model further improves performance, with XGB-VGGFace achieving the highest accuracy of 95.1% and an AUC of 0.91 on a balanced dataset. The results highlight the effectiveness of batch normalization and dropout layers in reducing overfitting and enhancing robustness.

- Key findings

Main findings	**Explanations**
Effectiveness of MANFA model	The MANFA model, combined with AdaBoost and XGBoost classifiers, outperforms existing systems in detecting manipulated face images, achieving higher accuracy and AUC.
Robustness of hybrid framework	Hybrid frameworks, integrating deep learning with boosting algorithms, provide robust solutions for addressing imbalanced datasets and achieving superior performance in manipulated image detection.

Enhanced accuracy with transfer learning	Transfer learning using pre-trained models like VGGFace significantly improves detection accuracy, with XGB-VGGFace achieving state-of-the-art performance.
Role of batch normalization and dropout	Batch normalization and dropout layers play a crucial role in reducing overfitting and enhancing the robustness of the detection model.

- Limitations

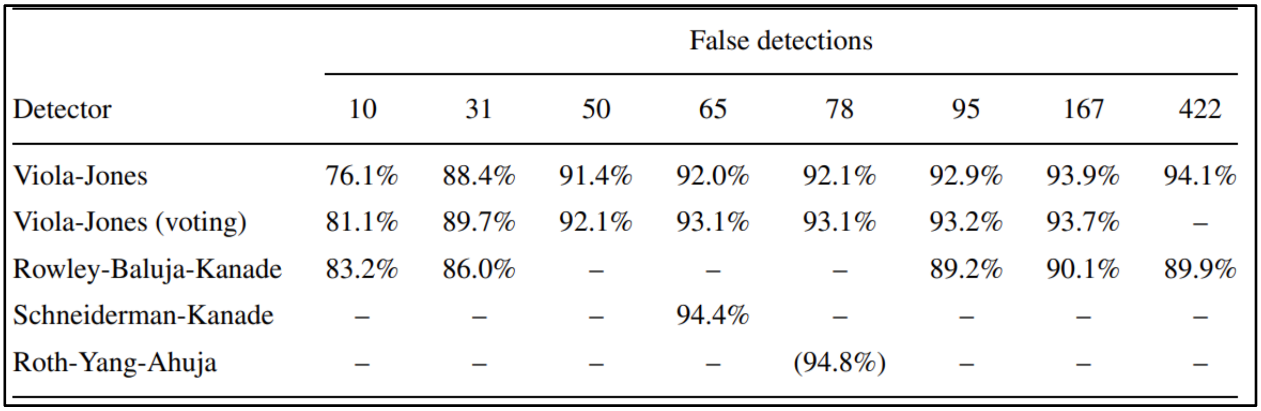
Main findings	**Explanations**
Lack of preprocessing	The absence of pre-processing techniques such as image whitening transformation or augmentation could potentially limit the model's performance.
Constraints in detecting computer-generated images	While proficient in detecting manually altered face images, the model may struggle with identifying computer-generated images like those produced by Generative Adversarial Networks (GANs).
Lack of localization techniques	The model detects manipulation but does not localize manipulated regions within the image, indicating the necessity for further research on localization techniques for improved detection precision.

<u>Study 3: Robust real-time face detection by Viola & Jones in 2004</u>

- Summary

This paper introduces a real-time face detection framework designed for rapid processing without compromising the detection accuracy. It presents three key contributions: the introduction of the "Integral Image" representation for quick feature computation, the development of an efficient classifier using AdaBoost, and the implementation of a cascade method to discard background regions swiftly. Experiments demonstrate the system's performance comparable to previous approaches while operating at 15 frames per second on a conventional desktop.

- Result



	False detections							
Detector	10	31	50	65	78	95	167	422
Viola-Jones	76.1%	88.4%	91.4%	92.0%	92.1%	92.9%	93.9%	94.1%
Viola-Jones (voting)	81.1%	89.7%	92.1%	93.1%	93.1%	93.2%	93.7%	–
Rowley-Baluja-Kanade	83.2%	86.0%	–	–	–	89.2%	90.1%	89.9%
Schneiderman-Kanade	–	–	–	94.4%	–	–	–	–
Roth-Yang-Ahuja	–	–	–	–	(94.8%)	–	–	–

Figure X: Detection rates across different systems

The paper compares the detection rates of its framework with other published systems, showcasing its competitive performance. Specifically, the framework achieves a detection rate of 77.8% with 5 false positives on the MIT test set, outperforming previous systems in terms of both speed and accuracy.

- Key findings

Main findings	**Explanations**
Efficiency of integral image	The Integral Image representation enables rapid feature computation, reducing the need for multi-scale image processing typically required for face detection.
Effectiveness of AdaBoost classifier	The classifier built using AdaBoost effectively selects critical visual features, resulting in a computationally efficient yet accurate detection system.
Advantages of Cascade method	The cascade of classifiers efficiently discards background regions, allowing more computation to be allocated to

	promising face-like regions, thus improving overall detection accuracy.

- Limitations

Main findings	**Explanations**
Constraint on face rotation	The detector is trained on frontal, upright faces with only rough alignment, resulting in limited tolerance to face rotation. Faces tilted beyond approximately ±15 degrees in-plane and ±45 degrees out-of-plane may cause the detector to become unreliable.
Failure under harsh backlighting	Harsh backlighting conditions, where faces appear dark against a relatively light background, sometimes lead to detection failures.
Over-reliant on eyes as key detection	The result shows that while occlusion of the mouth may not significantly affect detection, occlusion of the eyes typically leads to detection failure.

Study 4: Survey of face detection under low-quality images by Zhou et. al in 2018

- Summary

This paper examines the performance of existing face detection algorithms under low-quality conditions, which are common in practical applications like surveillance systems. It reviews state-of-the-art detectors, evaluates their performance on the FDDB benchmark dataset, and investigates their degradation in detecting faces under various levels of blur, noise, and contrast.

- Result

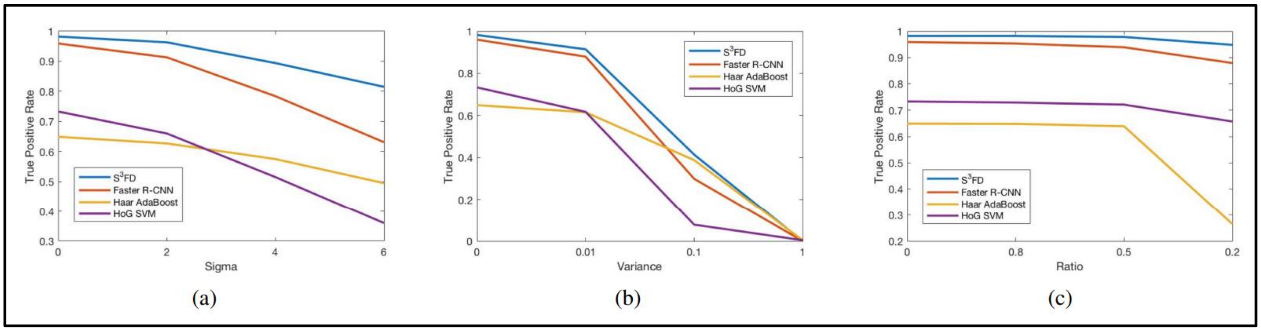


Figure X: Comparing the performance of different models across blur, noise, brightness and contrast.

The study evaluates four face detection models: Viola-Jones (Haar Cascade), HoG-SVM, Faster R-CNN, and S3FD, on images distorted with blur, noise, and altered brightness and contrast. The results reveal that both traditional and deep learning-based detectors struggle with low-quality images. While scale-variant models like S3FD show some resilience to blur due to their multi-layer feature extraction, all models experience significant performance degradation under high levels of noise and contrast alteration.

- Key findings

Main findings	**Explanations**
Sensitivity to low-quality images	Both hand-crafted and deep learning features exhibit sensitivity to low-quality images, impacting the detection performance across all tested conditions.
Impact of blurriness on detection efficiency	Blur affects the efficiency of face detection algorithms, with multi-layer feature extraction aiding scale-variant models like S3FD in maintaining detection performance for smaller faces.
Challenges faced by the occurrence of noise	High levels of additive noise severely impair detection efficiency across all models, indicating the inability of pre-trained networks to effectively utilize noise-free features.

Robustness in contrast and brightness	Deep networks and traditional methods show better robustness in detecting faces under altered brightness and contrast due to the normalization process during testing.

- Limitation

Since this research is published as a form of survey in other papers, there are no experiments available to analyze their limitations. Instead, this research highlights the limitation of current models in detecting faces under low quality images, urging future researchers to dive deeper into solving these real-life problems.

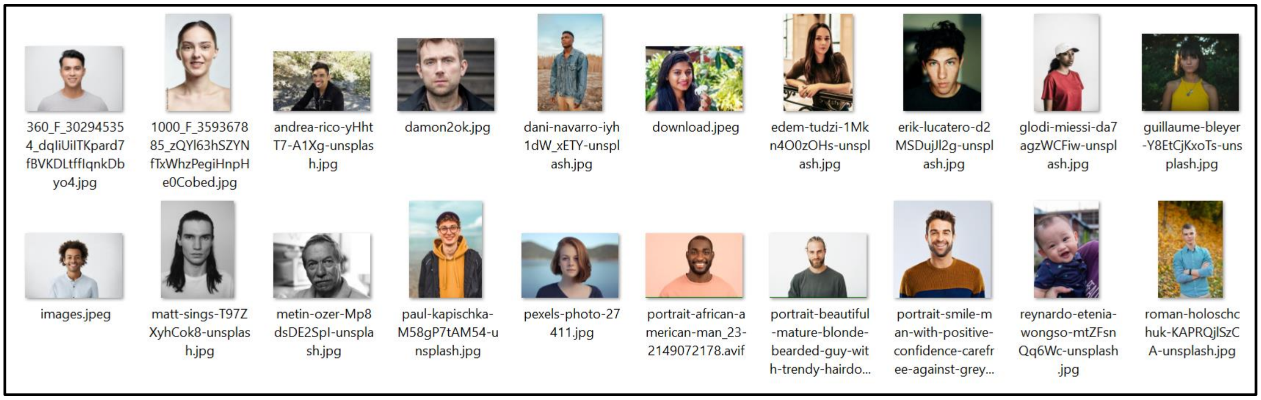
4.2.4 Data preparation

This research focuses on three main experiments, each containing two datasets.

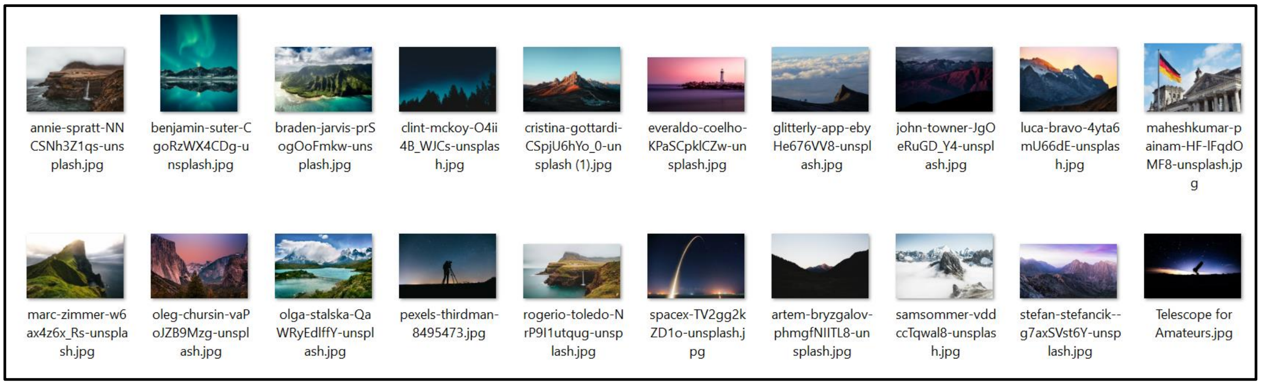
Experiment	**Human faces**	**Animal images**	**Landscape images**	**Blurry human faces**	**Testing images**
Test A	20				10 * Human faces
Test B		20			10 * Animal images
Test C			20		10 * Landscape images
Test D				20	10* Blurry human faces

All images are retrieved from royalty-free websites to avoid any copyright issues, such as Unsplash, Pixabay, and Pexels. Meanwhile, each dataset will contain 20 images.

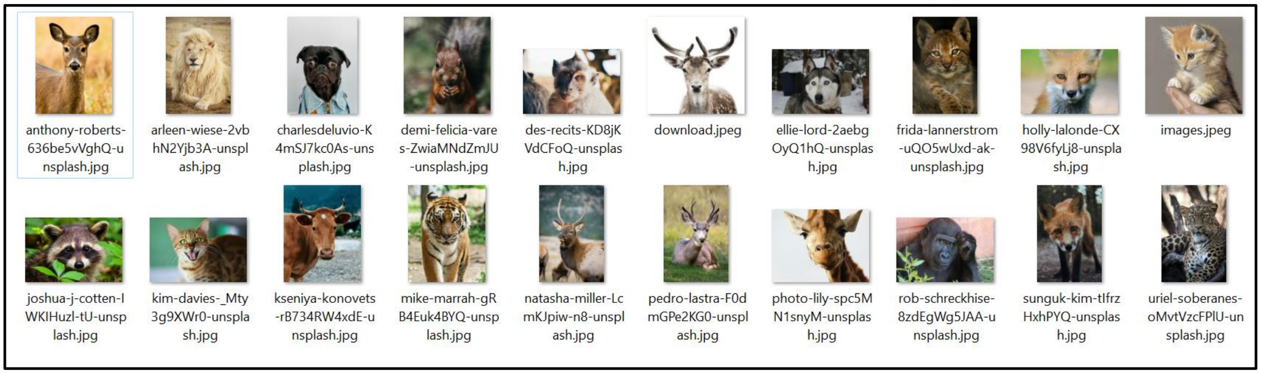
- Human faces (Training Set 20 pieces)


360_F_302945354_dqliUilTKpard7fBVKDLtfflqnkDbyo4.jpg
1000_F_359367885_zQYl63hSZYNfTxWhzPegiHnpHe0Cobed.jpg
andrea-rico-yHhtT7-A1Xg-unsplash.jpg
damon2ok.jpg
dani-navarro-iyh1dW_xETY-unsplash.jpg
download.jpeg
edem-tudzi-1Mkn4O0zOHs-unsplash.jpg
erik-lucatero-d2MSDujJl2g-unsplash.jpg
glodi-miessi-da7agzWCFiw-unsplash.jpg
guillaume-bleyer-Y8EtCjKxoTs-unsplash.jpg
images.jpeg
matt-sings-T97ZXyhCok8-unsplash.jpg
metin-ozer-Mp8dsDE2Spl-unsplash.jpg
paul-kapischka-M58gP7tAM54-unsplash.jpg
pexels-photo-27411.jpg
portrait-african-american-man_23-2149072178.avif
portrait-beautiful-mature-blonde-bearded-guy-with-trendy-hairdo...
portrait-smile-man-with-positive-confidence-carefree-against-grey...
reynardo-etenia-wongso-mtZFsnQq6Wc-unsplash.jpg
roman-holoschchuk-KAPRQjlSzCA-unsplash.jpg

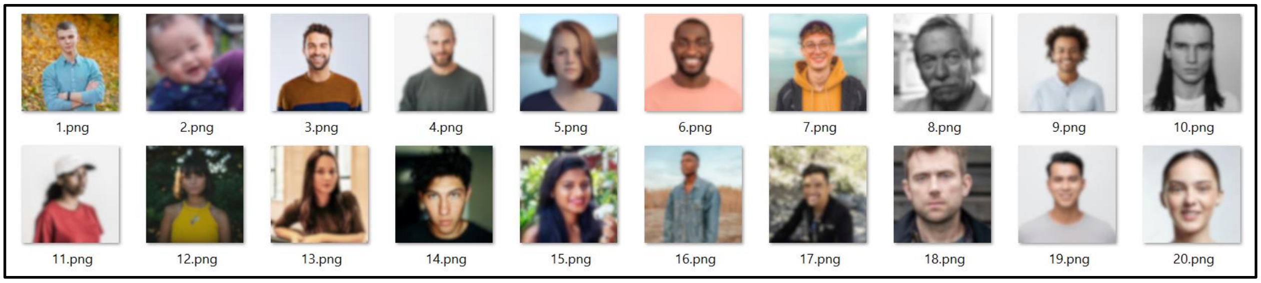
- Landscape / scenery images (Training Set 20 pieces)


annie-spratt-NNCSNh3Z1qs-unsplash.jpg
benjamin-suter-CgoRzWX4CDg-unsplash.jpg
braden-jarvis-prSogOoFmkw-unsplash.jpg
clint-mckoy-O4ii4B_WJCs-unsplash.jpg
cristina-gottardi-CSpjU6hYo_0-unsplash (1).jpg
everaldo-coelho-KPaSCpklCZw-unsplash.jpg
glitterly-app-ebyHe676VV8-unsplash.jpg
john-towner-JgOeRuGD_Y4-unsplash.jpg
luca-bravo-4yta6mU66dE-unsplash.jpg
maheshkumar-painam-HF-lFqdOMF8-unsplash.jpg
marc-zimmer-w6ax4z6x_Rs-unsplash.jpg
oleg-chursin-vaPoJZB9Mzg-unsplash.jpg
olga-stalska-QaWRyEdlffY-unsplash.jpg
pexels-thirdman-8495473.jpg
rogerio-toledo-NrP9I1utqug-unsplash.jpg
spacex-TV2gg2kZD1o-unsplash.jpg
artem-bryzgalov-phmgfNIITL8-unsplash.jpg
samsommer-vddccTqwal8-unsplash.jpg
stefan-stefancik--g7axSVst6Y-unsplash.jpg
Telescope for Amateurs.jpg

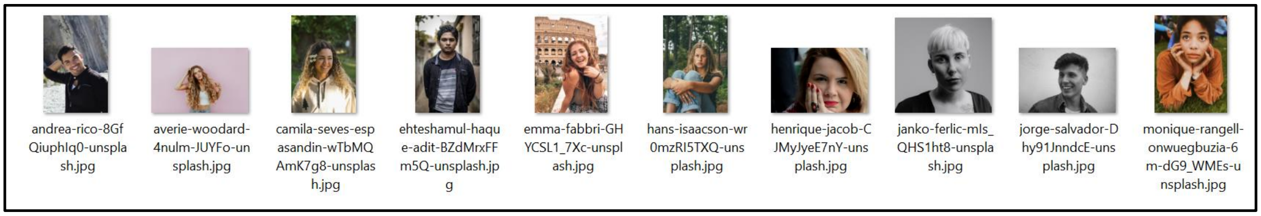
- Animal images (Training Set 20 pieces)


anthony-roberts-636be5vVghQ-unsplash.jpg
arleen-wiese-2vbhN2Yjb3A-unsplash.jpg
charlesdeluvio-K4mSJ7kc0As-unsplash.jpg
demi-felicia-vares-ZwiaMNdZmJU-unsplash.jpg
des-recits-KD8jKVdCFoQ-unsplash.jpg
download.jpeg
ellie-lord-2aebgOyQ1hQ-unsplash.jpg
frida-lannerstrom-uQO5wUxd-ak-unsplash.jpg
holly-lalonde-CX98V6fyLj8-unsplash.jpg
images.jpeg
joshua-j-cotten-IWKIHuzl-tU-unsplash.jpg
kim-davies-_Mty3g9XWr0-unsplash.jpg
kseniya-konovets-rB734RW4xdE-unsplash.jpg
mike-marrah-gRB4Euk4BYQ-unsplash.jpg
natasha-miller-LcmKJpiw-n8-unsplash.jpg
pedro-lastra-F0dmGPe2KG0-unsplash.jpg
photo-lily-spc5MN1snyM-unsplash.jpg
rob-schreckhise-8zdEgWg5JAA-unsplash.jpg
sunguk-kim-tlfrzHxhPYQ-unsplash.jpg
uriel-soberanes-oMvtVzcFPlU-unsplash.jpg

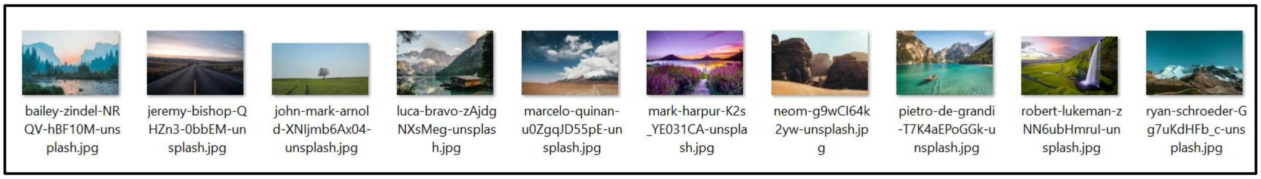
- Blurry human images (Training Set 20 pieces)



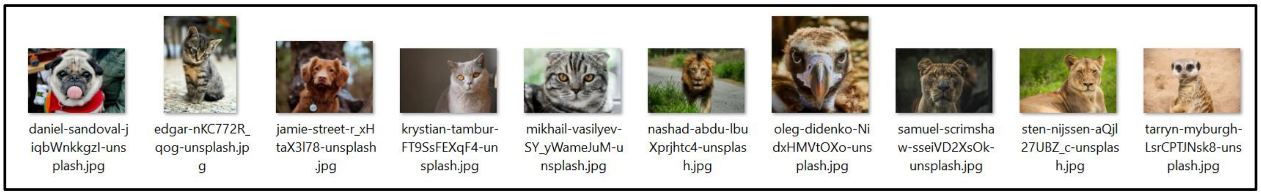
- Human faces (Testing images 10 pieces)


andrea-rico-8GfQiuphIq0-unsplash.jpg
averie-woodard-4nulm-JUYFo-unsplash.jpg
camila-seves-espasandin-wTbMQAmK7g8-unsplash.jpg
ehteshamul-haque-adit-BZdMrxFFm5Q-unsplash.jpg
emma-fabbri-GHYCSL1_7Xc-unsplash.jpg
hans-isaacson-wr0mzRI5TXQ-unsplash.jpg
henrique-jacob-CJMyJyeE7nY-unsplash.jpg
janko-ferlic-mls_QHS1ht8-unsplash.jpg
jorge-salvador-Dhy91JnndcE-unsplash.jpg
monique-rangell-onwuegbuzia-6m-dG9_WMEs-unsplash.jpg

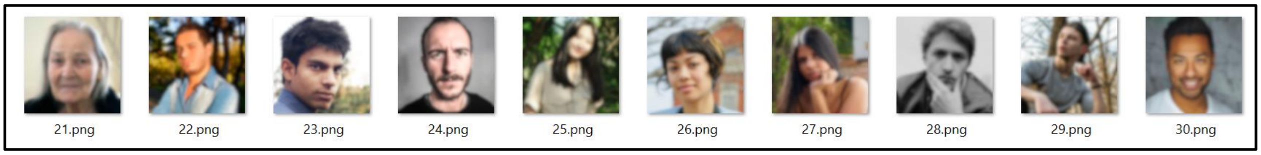
- Landscape images (Testing images 10 pieces)


bailey-zindel-NRQV-hBF10M-unsplash.jpg
jeremy-bishop-QHZn3-0bbEM-unsplash.jpg
john-mark-arnold-XNIjmb6Ax04-unsplash.jpg
luca-bravo-zAjdgNXsMeg-unsplash.jpg
marcelo-quinan-u0ZgqJD55pE-unsplash.jpg
mark-harpur-K2s_YE031CA-unsplash.jpg
neom-g9wCI64k2yw-unsplash.jpg
pietro-de-grandi-T7K4aEPoGGk-unsplash.jpg
robert-lukeman-zNN6ubHmruI-unsplash.jpg
ryan-schroeder-Gg7uKdHFb_c-unsplash.jpg

- Animal images (Testing images 10 pieces)


daniel-sandoval-jiqbWnkkgzI-unsplash.jpg
edgar-nKC772R_qog-unsplash.jpg
jamie-street-r_xHtaX3l78-unsplash.jpg
krystian-tambur-FT9SsFEXqF4-unsplash.jpg
mikhail-vasilyev-SY_yWameJuM-unsplash.jpg
nashad-abdu-lbuXprjhtc4-unsplash.jpg
oleg-didenko-NidxHMVtOXo-unsplash.jpg
samuel-scrimshaw-sseiVD2XsOk-unsplash.jpg
sten-nijssen-aQjl27UBZ_c-unsplash.jpg
tarryn-myburgh-LsrCPTJNsk8-unsplash.jpg

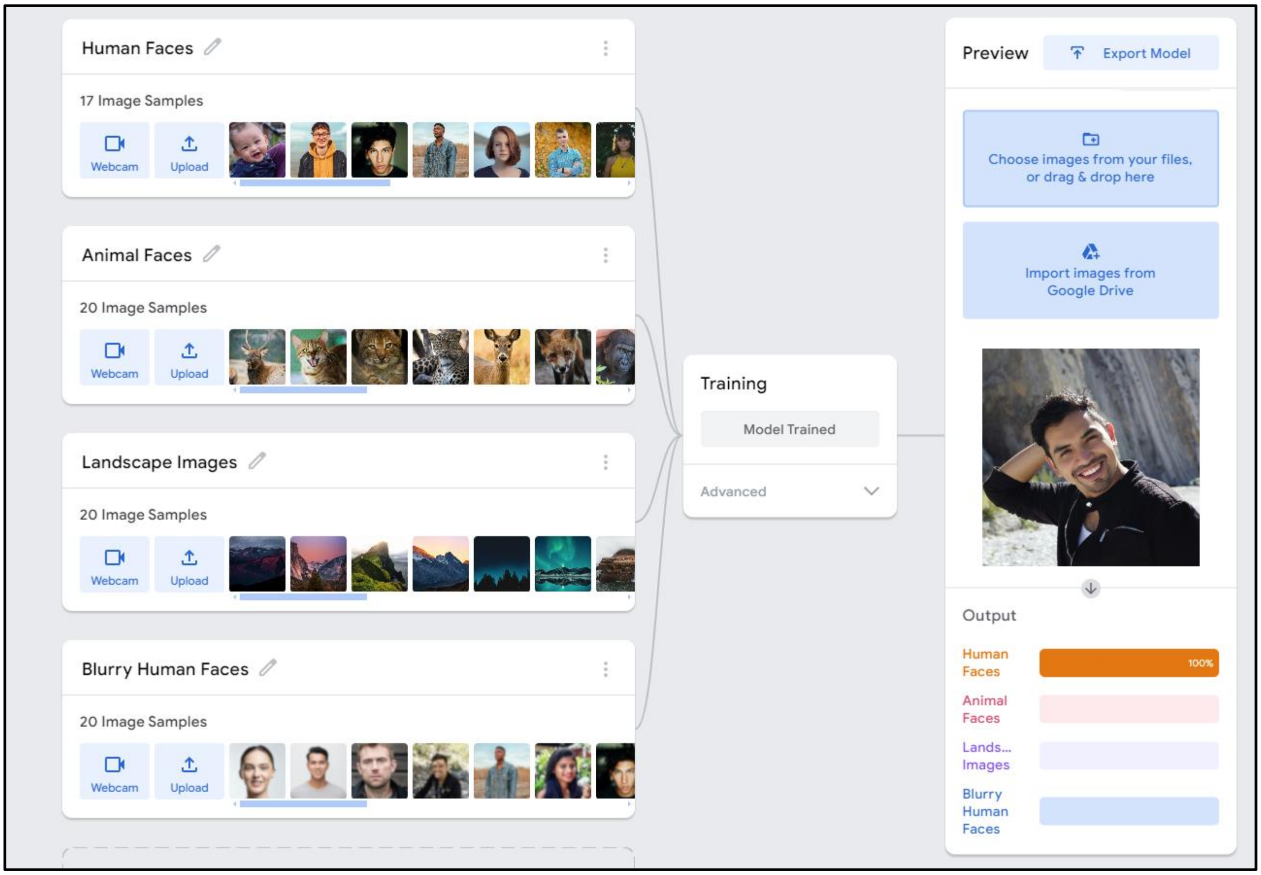
- Blurry human images (Testing images 10 pieces)



4.3 Solution Implementation (Develop & Deploy)

Since this research focuses on comparing the result between QB and Techaeble Machine Learning algorithms, the prime metric would be detecting the image accuracy. There are two main phases to be executed.

4.3.1 Phase 1: Image Detection on Google's Teachable Machine



Google's Teachable Machine is a valuable tool in the realm of artificial intelligence education. It has been recognized for its simplicity and ease of use, requiring no coding skills to build machine-learning models (Gupta & Homchan, 2021). It has been used to train models for image classification, aiding in tasks such as object detection and recognition (Mathew & Mahesh, 2021). Teachable Machine uses Convolutional Neural Network (CNN) as the main Machine Learning algorithm to train and recognize patterns (Malahina et al., 2022). The tool's versatility has been demonstrated in various applications, from real-time attendance systems to object detection in unconventional settings (Feng & McDonald, 2023). Thus, it is the perfect choice as the main comparative tool for QB's face detection system.

There are 4 main classes being set in the pre-processing phase:

- Human faces
- Animal faces
- Landscape images
- Blurry human faces

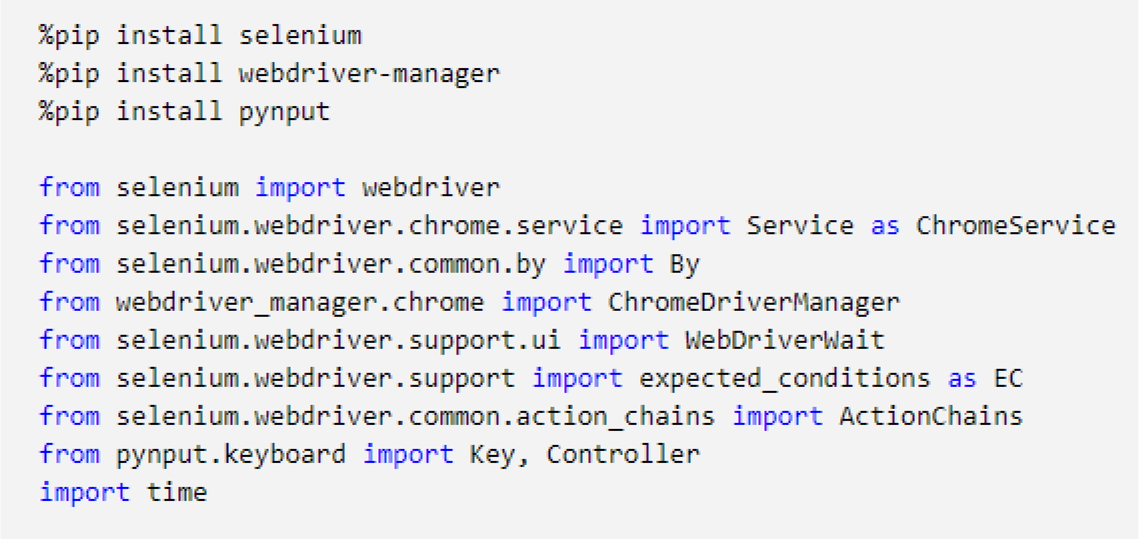
Each of these classes has 20 images for training the machine-learning model. Once the model is trained, the system will prompt the user to input a testing image to determine the testing result. All tests utilize the same training images, the difference occurs in the testing images among Test A, Test B, Test C, and Test D.

4.3.2 Phase 2: Face detection on QB's system

This phase focuses on developing the automation for testing QB's face detection accuracy. The finding and result are not included and only the programming aspect is included to allow the reader to comprehend the code. The result would be discussed within the conclusion section.

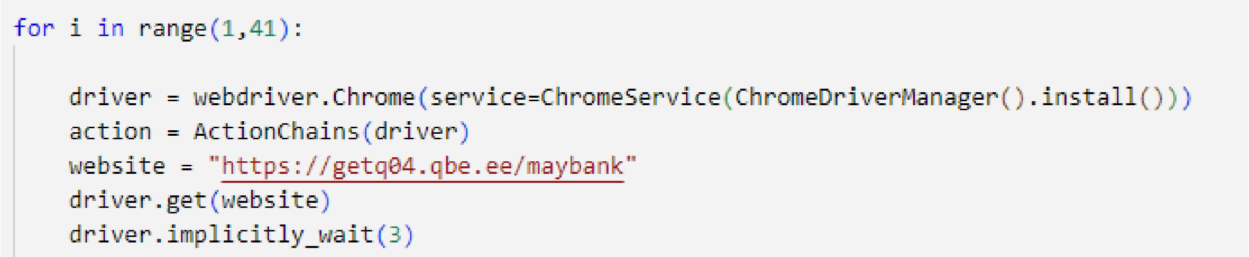
Explanation

The pip install package for the implementation of Selenium can be seen below. The package imported several functions to enable the operation of automation.



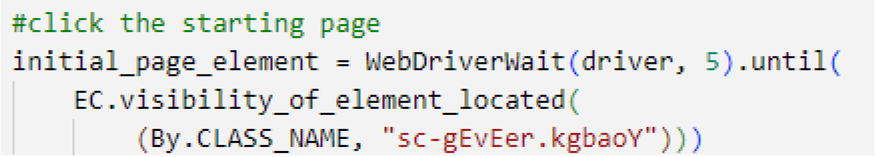
```
%pip install selenium
%pip install webdriver-manager
%pip install pynput

from selenium import webdriver
from selenium.webdriver.chrome.service import Service as ChromeService
from selenium.webdriver.common.by import By
from webdriver_manager.chrome import ChromeDriverManager
from selenium.webdriver.support.ui import WebDriverWait
from selenium.webdriver.support import expected_conditions as EC
from selenium.webdriver.common.action_chains import ActionChains
from pynput.keyboard import Key, Controller
import time
```

The code was written in a for loop as there are 40 images to be tested. Hence, the automation process would repeat 40 cycles. The driver variable downloads the Chrome browser, which is compatible with Selenium, creating a path to the ChromeDriver as an argument to execute. The action variable enables the automation of low-level interaction such as mouse movement, clicking, and others. The driver.get() function directs the web driver Chrome to initiate the website. An implicit wait of 3 seconds was set, providing a buffer for a page to refresh if required.



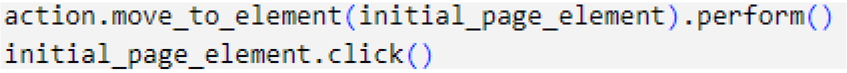
```
for i in range(1,41):

    driver = webdriver.Chrome(service=ChromeService(ChromeDriverManager().install()))
    action = ActionChains(driver)
    website = "https://getq04.qbe.ee/maybank"
    driver.get(website)
    driver.implicitly_wait(3)
```

The code would allow the driver to wait for 5 seconds until the specific element within the website is visible. Within this case, it would be the element identified by the class which is `sc-gEvEer.kgbaoY`.



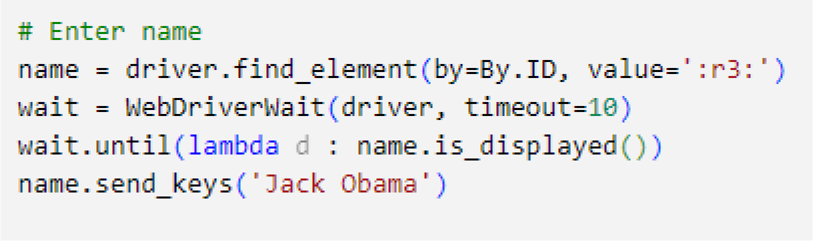
```
#click the starting page
initial_page_element = WebDriverWait(driver, 5).until(
    EC.visibility_of_element_located(
        (By.CLASS_NAME, "sc-gEvEer.kgbaoY")))
```

The mouse will then move to the element and an action of click is performed.



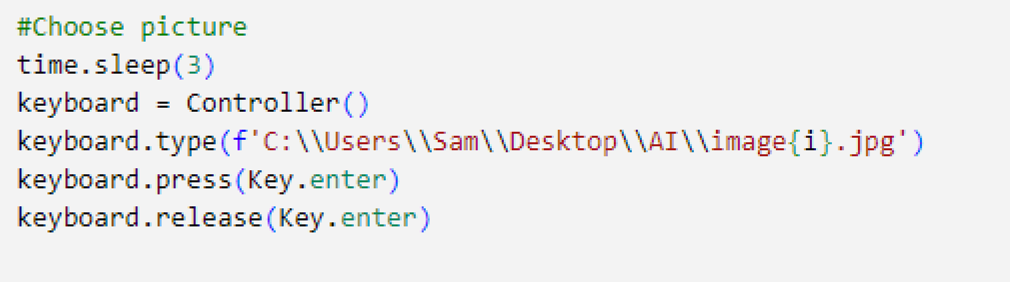
```
action.move_to_element(initial_page_element).perform()
initial_page_element.click()
```

The section of the code enables typing text into the input field. The element is identified by using an ID, providing a timeout period of 10 seconds until the ID is found and the text is typed.



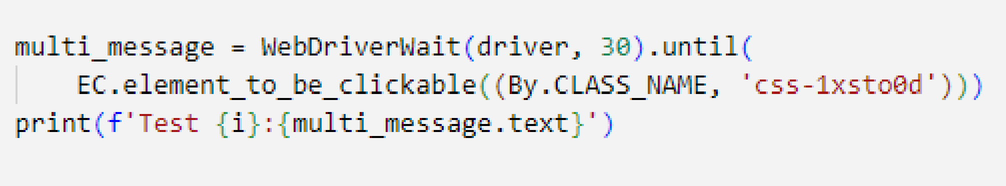
```
# Enter name
name = driver.find_element(by=By.ID, value=':r3:')
wait = WebDriverWait(driver, timeout=10)
wait.until(lambda d : name.is_displayed())
name.send_keys('Jack Obama')
```

Uploading the image requires the use of time.sleep() function, as the import of image is obtained from the computer instead. Therefore, a manual instruction to wait for 3 seconds is set. The keyboard function was implemented instead as the action operates outside of Chrome WebDriver. The path extension to the image was included and {i} dynamically changes as it has a loop of 40 times thus it will dynamically change from 1 to 40. The enter key is press and release.



```
#Choose picture
time.sleep(3)
keyboard = Controller()
keyboard.type(f'C:\\Users\\Sam\\Desktop\\AI\\image{i}.jpg')
keyboard.press(Key.enter)
keyboard.release(Key.enter)
```

The system will then analyze the picture to detect the presence of the human face. An allowance of 30 seconds is provided for the Queue Bee system to analyze. Once the analysis is complete, the code will print the result of the facial detection output.



```
multi_message = WebDriverWait(driver, 30).until(
    EC.element_to_be_clickable((By.CLASS_NAME, 'css-1xsto0d')))
print(f'Test {i}:{multi_message.text}')
```

4.4 Solution Output

This section outlines the output from both Teachable Machine and QB's facial detection system.

4.4.1 Test A: Facial Detection with Human Faces

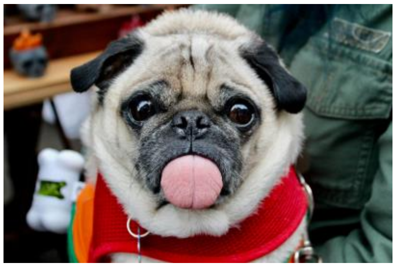
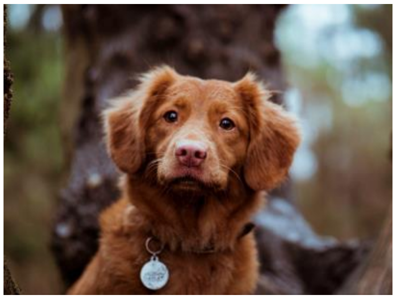
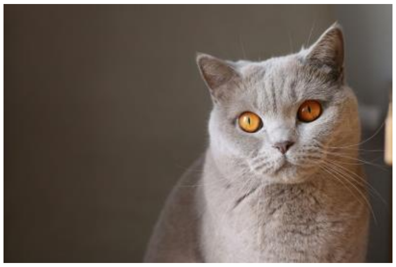
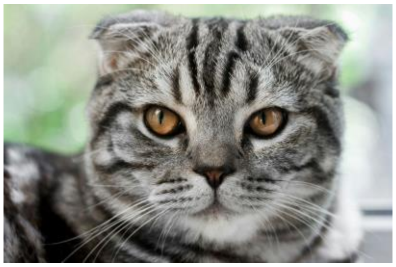
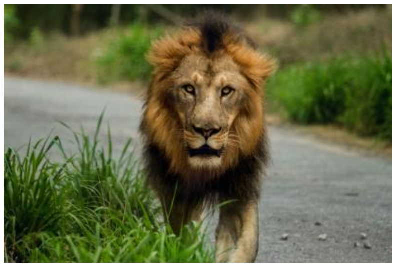
Testing images	QB	Teachable Machine
	Pass	Pass
	Pass	Pass
	Pass	Pass

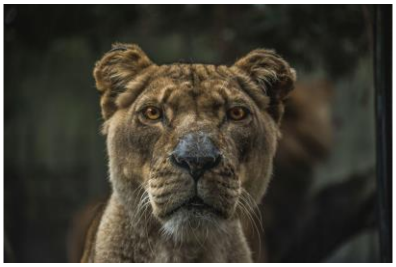
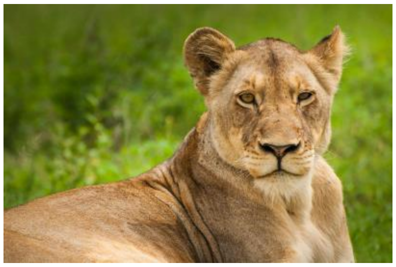
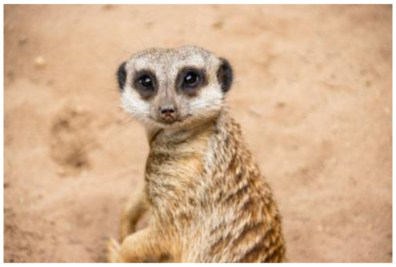
	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass

	Pass	Pass
	Pass	Pass

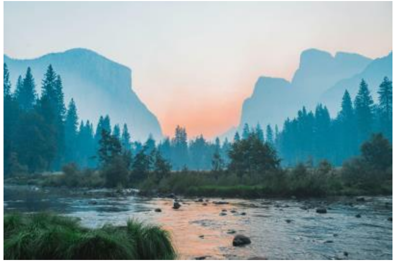
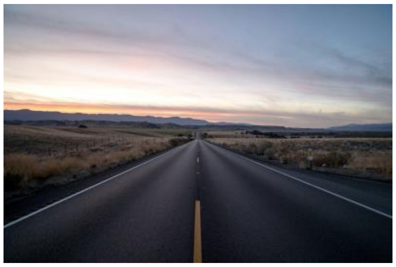
4.4.2 Test B: Facial Detection with Animal Faces

Testing images	**QB**	**Teachable Machine**
	Pass	Pass
	Pass	Pass

	Fail	Pass
	Fail	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass

	Fail	Pass
	Fail	Pass
	Pass	Pass

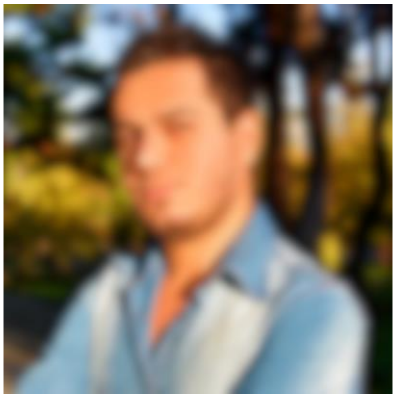
4.4.3 Test C: Facial Detection with Landscape Images

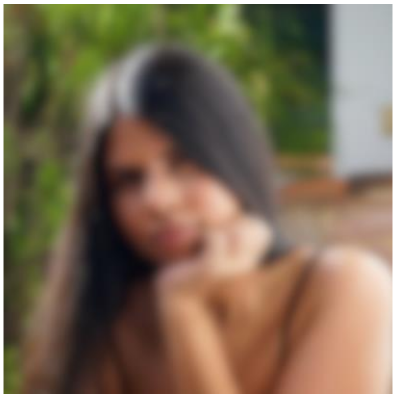
Testing images	**QB**	**Teachable Machine**
	Pass	Pass
	Pass	Pass

	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass

	Pass	Pass
	Pass	Pass

4.4.4 Test D: Facial Detection with Blurry Human Faces

Testing images	QB	Teachable Machine
	Pass	Pass
	Pass	Pass
	Pass	Pass

	Pass	Pass
	Pass	Pass
	Pass	Pass
	Pass	Pass

	Pass	Pass
	Pass	Pass
	Pass	Pass

4.4.5 Discussion

This section outlines the overall result retrieved from the four tests mentioned above.

Test Set	**QB (Pass)**	**Teachable Machine (Pass)**
A (Human Faces)	100%	100%
B (Animal Faces)	60%	100%

C (Landscape Images)	100%	100%
D (Blurry Faces)	100%	100%
Average Output	**90%**	**100%**

Based on the above results, it can be deduced that the Teachable Machine has a passing rate of 100% for all test sets. Meanwhile, for QB's face detection system, only Test Set B has a passing rate of 60%, while the rest had a passing rate of 100%.

As a result, QB's machine-learning algorithm has a lower accuracy in identifying the differences between a human face and an animal face.

5.0 Conclusion

The research was conducted to determine the effectiveness and reliability of the facial detection system between both algorithms to satisfy the research aim. The underlying objectives are:

1. **To deploy an automated software testing system to test the face detection system**

With the development of automation utilizing tools such as Selenium on Python, the software testing system was successfully built. Upon testing the code, it took around 24 minutes to upload the image from the test set for QB's algorithm. On the other hand, the manual test on a teachable machine took an average of 45 minutes to execute the same script. Therefore, Selenium was 195% quicker when compared to manually executing the script.

2. **To implement a Machine Learning program in detecting human faces and non-human faces**

Based on phase 1, the model utilizes the CNN algorithm to detect human faces. Upon training the model, the model obtained a 100% accuracy for facial detection between human and non-human faces.

3. **To compare the result of facial detection between the actual face detection system and the Machine Learning model**

Based on the result, it could be seen that the CNN algorithm had 10% better performance when compared to QB's algorithm. The four image failures were from the animal section, highlighting QB's algorithm does have weak points towards animal images, especially dog and lion images.

5.1 Limitation

5.1.1 Lack of transparency

With the research conducted, the project does not have access to QB's facial detection algorithm. The project does not have insight regarding the algorithms thus unable to determine the comparison of algorithms specifically. Unable to identify the algorithm used by QB, it raises a secondary concern of whether the gap of performance could derive from the training set, hyper-parameter changes, and many more.

5.1.2 Small-scale testing

The research was conducted on a small scale whereas only 80 images were utilized for training whereas 40 images were reserved for test sets. With QB's system appearing as a mature product, it begs to question whether the current model could perform similarly to a larger sample size (Althnian et al, 2021; Navarro et al, 2021).

5.1.3 Lack of image diversity

The current research only has a total of 80 images for training. Although the test was successful, scaling the research into a larger sample size required a larger set of images for training. With QB's current venture within the Southeast Asia region, the current model may present certain biases therefore the extension of diverse data would be necessary to develop a robust model (Navarro et al, 2021; Viola & Jones, 2004; Rowley, Baluja and Kanade, 1998).

5.2 Future Work

5.2.1 Testing the model on AI-generated images

The current research does not cover the scope of AI-generated images within the research. Therefore, future research could include generated images within the training and test set. Future work could evaluate the performance of detection and recognizing of AI-generated images.

Depending on the evaluation of the model, the research may extend into a different research scope to detect deep fakes.

5.2.2 Testing the model on a larger sample size

As mentioned in the limitation part, QB's currently venture into several regions within Southeast Asia. The product is currently operating on a larger scale therefore the model will be required to undergo stress testing to observe the performance of the model. It furthermore allows the researcher to evaluate the algorithm activity when the algorithm is under heavy load. Moreover, the observation of similarity in performance is evaluated to determine the robustness, effectivity and efficiency of the CNN model.

5.2.3 Expands the training images based on region QB's venturing

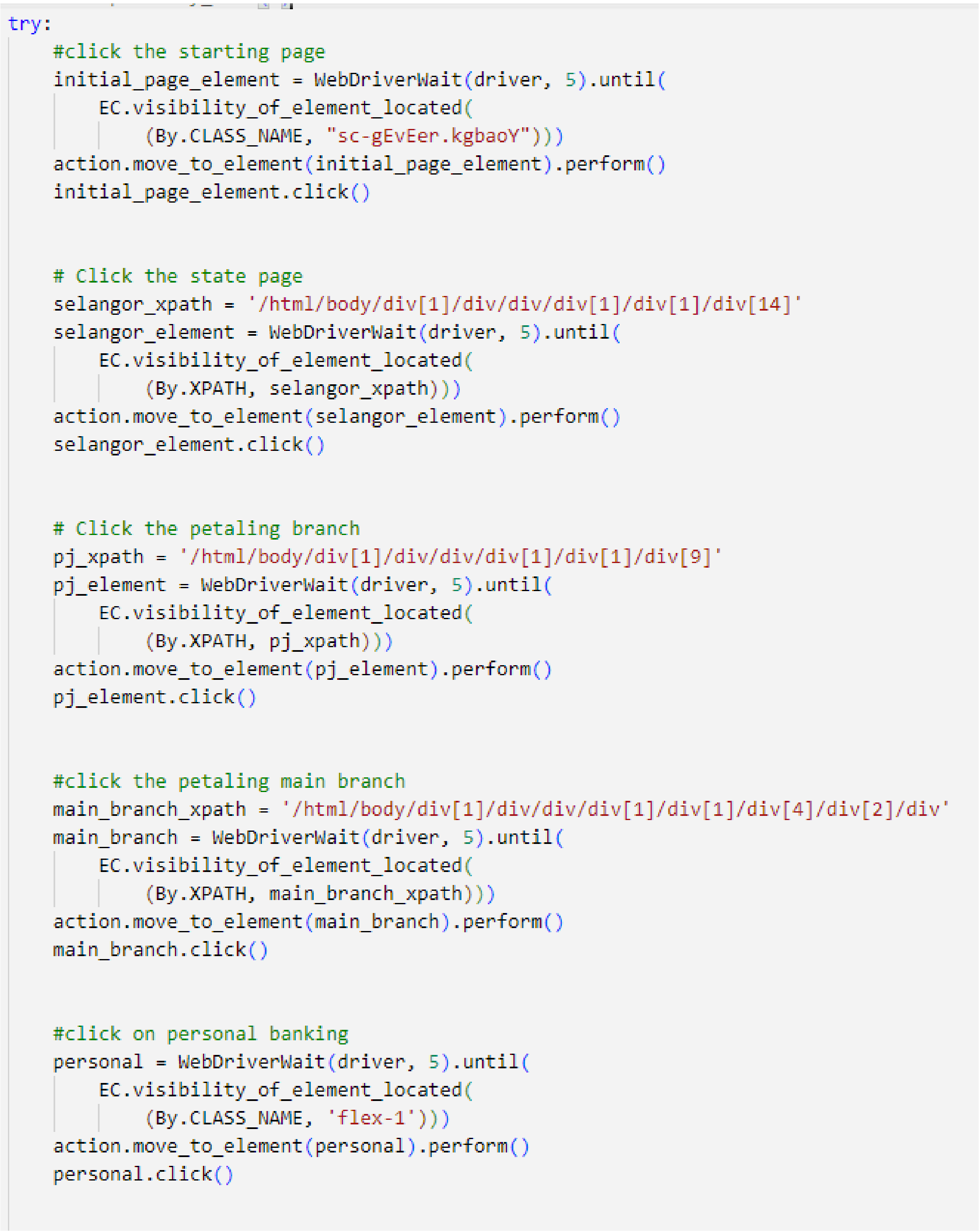
In relevance to the above paragraph, the inclusion of image diversity from several regions could be tested to evaluate the model performance. Further research could be conducted on the inclusion and exclusion of diversified racial facial images vs non-facial images, allowing the researcher to determine if diversity affects the performance of the model.

6. References

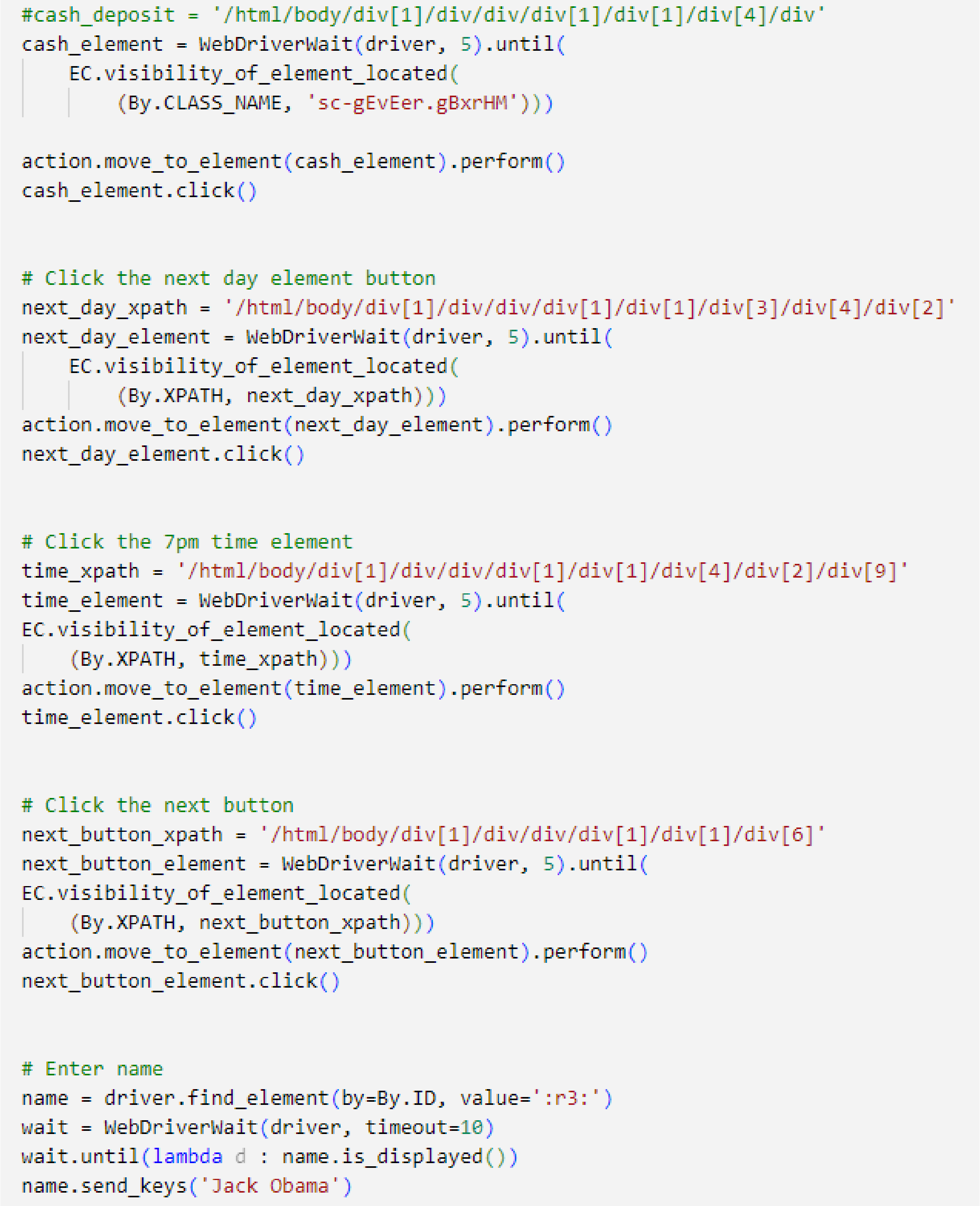
1. Althnian, A., AlSaeed, D., Al-Baity, H. H., Samha, A. K., Dris, A. B., Alzakari, N., … & Kurdi, H. (2021). Impact of dataset size on classification performance: an empirical evaluation in the medical domain. Applied Sciences, 11(2), 796. https://doi.org/10.3390/app11020796
2. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 2016, pp. 2818-2826, doi: 10.1109/CVPR.2016.308.
3. Dai, Jifeng & Li, Yi & He, Kaiming & Sun, Jian. (2016). R-fcn: Object detection via region-based fully convolutional networks.
4. Feng, K. J. K. and McDonald, D. W. (2023). Addressing ux practitioners' challenges in designing ml applications: an interactive machine learning approach. Proceedings of the 28th International Conference on Intelligent User Interfaces. https://doi.org/10.1145/3581641.3584064
5. Gupta, Y. and Homchan, S. (2021). Short communication: insect detection using a machine learning model. Nusantara Bioscience, 13(1). https://doi.org/10.13057/nusbiosci/n130110
6. H. A. Rowley, S. Baluja and T. Kanade, "Neural network-based face detection," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 20, no. 1, pp. 23-38, Jan. 1998, doi: 10.1109/34.655647.
7. He, Kaiming & Zhang, Xiangyu & Ren, Shaoqing & Sun, Jian. (2016). Deep Residual Learning for Image Recognition. 770-778. 10.1109/CVPR.2016.90.
8. Liu, Wei & Anguelov, Dragomir & Erhan, Dumitru & Szegedy, Christian & Reed, Scott & Fu, Cheng-Yang & Berg, Alexander. (2016). SSD: Single Shot MultiBox Detector. 9905. 21-37. 10.1007/978-3-319-46448-0_2.
9. Malahina, E. A. U., Hadjon, R. P., & Bisilisin, F. Y. (2022). Teachable machine: real-time attendance of students based on open source system. The IJICS (International Journal of Informatics and Computer Science), 6(3), 140. https://doi.org/10.30865/ijics.v6i3.4928

10. Mathew, M. and Mahesh, T. Y. (2021). Object detection based on teachable machine. Journal of VLSI Design and Signal Processing, 7(2). https://doi.org/10.46610/jovdsp.2021.v07i02.003
11. Navarro, C. L. A., Damen, J. A., Takada, T., Nijman, S. W. J., Dhiman, P., Ma, J., … & Hooft, L. (2021). Risk of bias in studies on prediction models developed using supervised machine learning techniques: systematic review. BMJ, n2281. https://doi.org/10.1136/bmj.n2281
12. N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05), San Diego, CA, USA, 2005, pp. 886-893 vol. 1, doi: 10.1109/CVPR.2005.177.
13. Ren, Shaoqing & He, Kaiming & Girshick, Ross & Sun, Jian. (2015). Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. IEEE Transactions on Pattern Analysis and Machine Intelligence. 39. 10.1109/TPAMI.2016.2577031.
14. Strümke, I., Slavkovik, M., & Madai, V. I. (2021). The social dilemma in artificial intelligence development and why we have to solve it. AI and Ethics, 2(4), 655-665. https://doi.org/10.1007/s43681-021-00120-w
15. Viola, P., & Jones, M. J. (2004). Robust Real-Time Face Detection. International Journal of Computer Vision, 57(2), 137–154. doi:10.1023/b:visi.0000013087.49260.fb
16. Zhou, J., Chen, F., Berry, A., Reed, M. D., Zhang, S., & Savage, S. (2020). A survey on ethical principles of ai and implementations. 2020 IEEE Symposium Series on Computational Intelligence (SSCI). https://doi.org/10.1109/ssci47803.2020.9308437
17. Zhou, Y., Liu, D., & Huang, T. (2018). Survey of Face Detection on Low-Quality Images. 2018 13th IEEE International Conference on Automatic Face & Gesture Recognition (FG 2018). doi:10.1109/fg.2018.00121

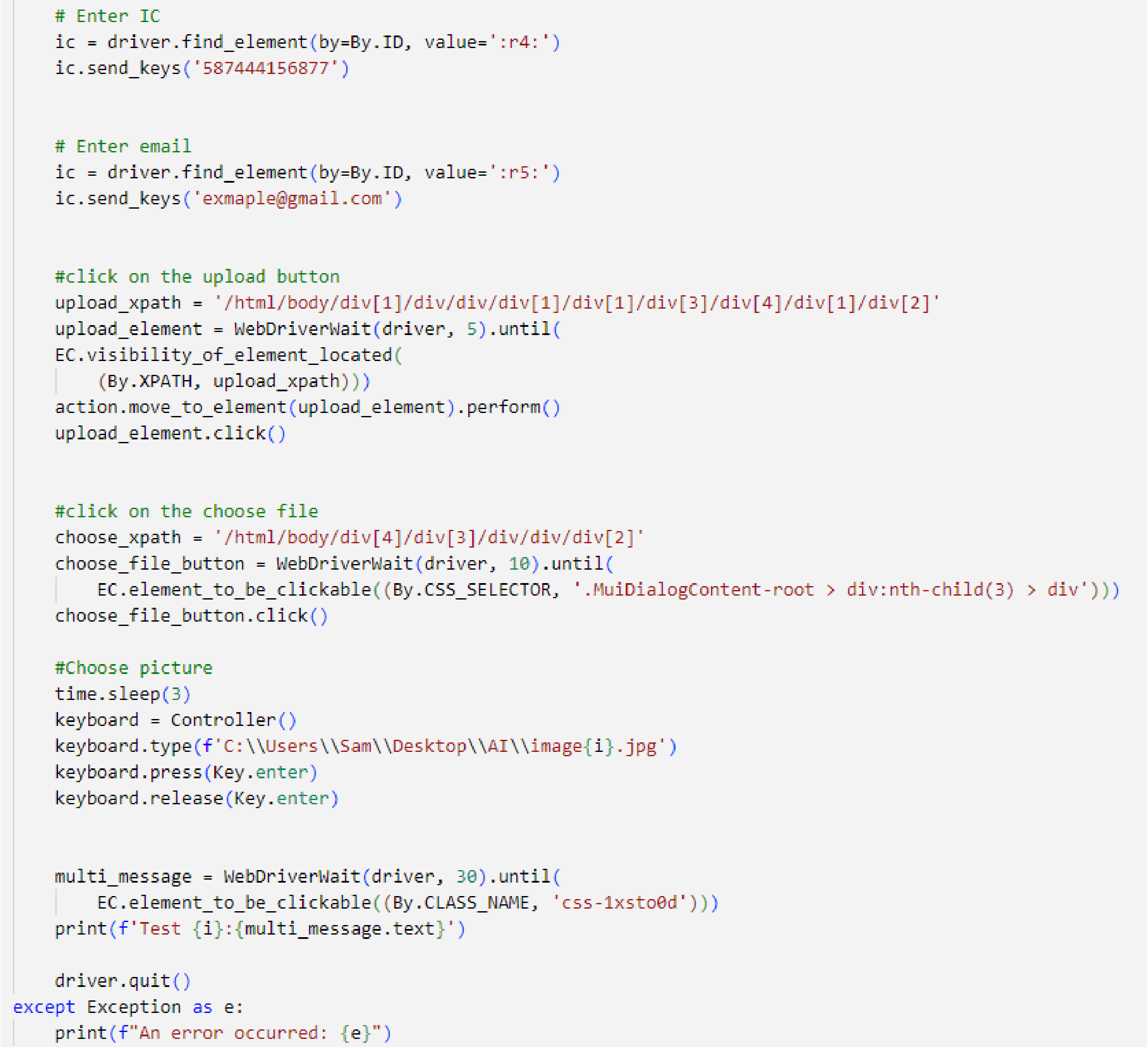
7. Appendix



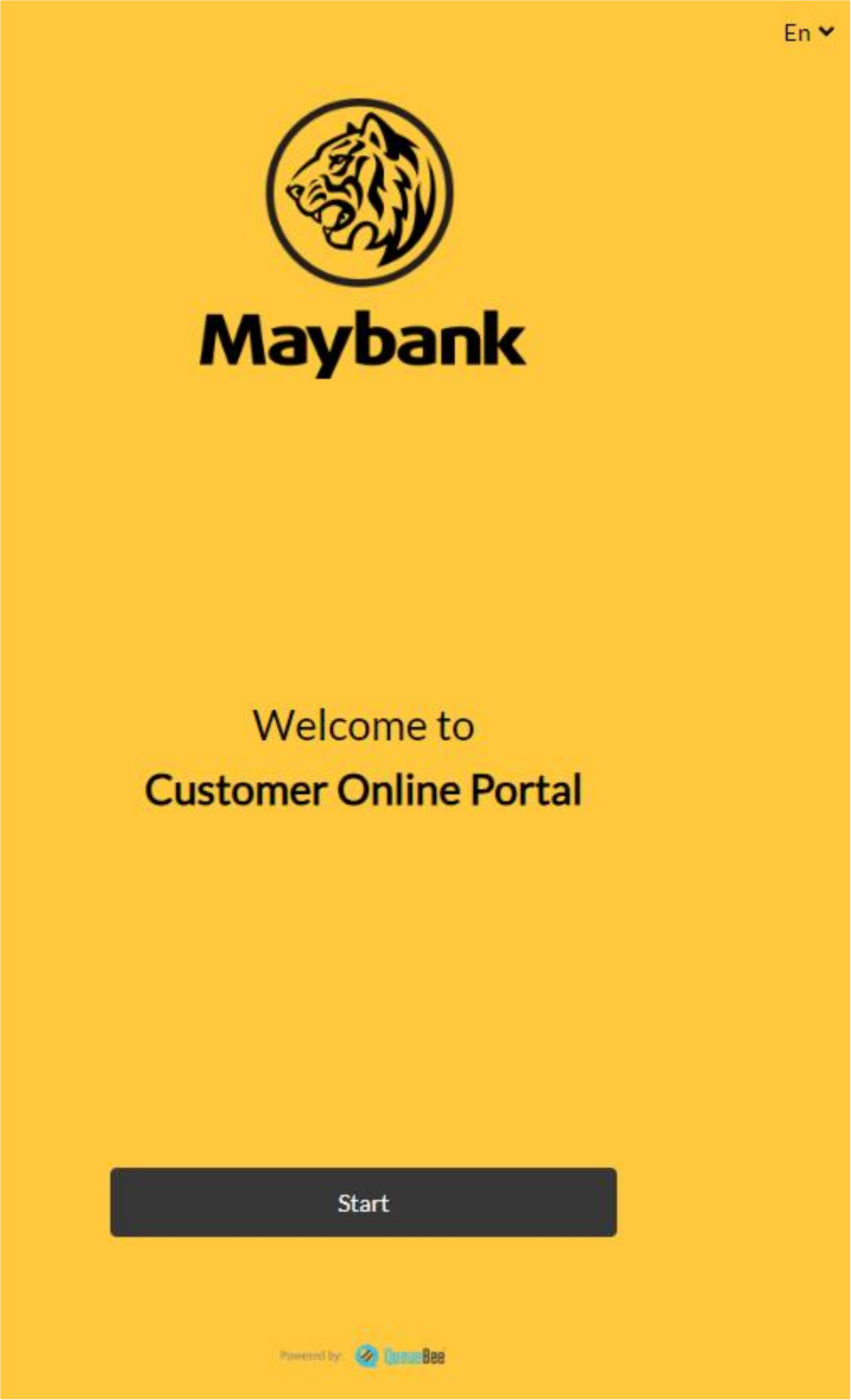
```
try:
    #click the starting page
    initial_page_element = WebDriverWait(driver, 5).until(
        EC.visibility_of_element_located(
            (By.CLASS_NAME, "sc-gEvEer.kgbaoY")))
    action.move_to_element(initial_page_element).perform()
    initial_page_element.click()


    # Click the state page
    selangor_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[14]'
    selangor_element = WebDriverWait(driver, 5).until(
        EC.visibility_of_element_located(
            (By.XPATH, selangor_xpath)))
    action.move_to_element(selangor_element).perform()
    selangor_element.click()


    # Click the petaling branch
    pj_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[9]'
    pj_element = WebDriverWait(driver, 5).until(
        EC.visibility_of_element_located(
            (By.XPATH, pj_xpath)))
    action.move_to_element(pj_element).perform()
    pj_element.click()


    #click the petaling main branch
    main_branch_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[4]/div[2]/div'
    main_branch = WebDriverWait(driver, 5).until(
        EC.visibility_of_element_located(
            (By.XPATH, main_branch_xpath)))
    action.move_to_element(main_branch).perform()
    main_branch.click()


    #click on personal banking
    personal = WebDriverWait(driver, 5).until(
        EC.visibility_of_element_located(
            (By.CLASS_NAME, 'flex-1')))
    action.move_to_element(personal).perform()
    personal.click()
```



```python
#cash_deposit = '/html/body/div[1]/div/div/div[1]/div[1]/div[4]/div'
cash_element = WebDriverWait(driver, 5).until(
    EC.visibility_of_element_located(
        (By.CLASS_NAME, 'sc-gEvEer.gBxrHM')))

action.move_to_element(cash_element).perform()
cash_element.click()


# Click the next day element button
next_day_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[3]/div[4]/div[2]'
next_day_element = WebDriverWait(driver, 5).until(
    EC.visibility_of_element_located(
        (By.XPATH, next_day_xpath)))
action.move_to_element(next_day_element).perform()
next_day_element.click()


# Click the 7pm time element
time_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[4]/div[2]/div[9]'
time_element = WebDriverWait(driver, 5).until(
EC.visibility_of_element_located(
    (By.XPATH, time_xpath)))
action.move_to_element(time_element).perform()
time_element.click()


# Click the next button
next_button_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[6]'
next_button_element = WebDriverWait(driver, 5).until(
EC.visibility_of_element_located(
    (By.XPATH, next_button_xpath)))
action.move_to_element(next_button_element).perform()
next_button_element.click()


# Enter name
name = driver.find_element(by=By.ID, value=':r3:')
wait = WebDriverWait(driver, timeout=10)
wait.until(lambda d : name.is_displayed())
name.send_keys('Jack Obama')
```

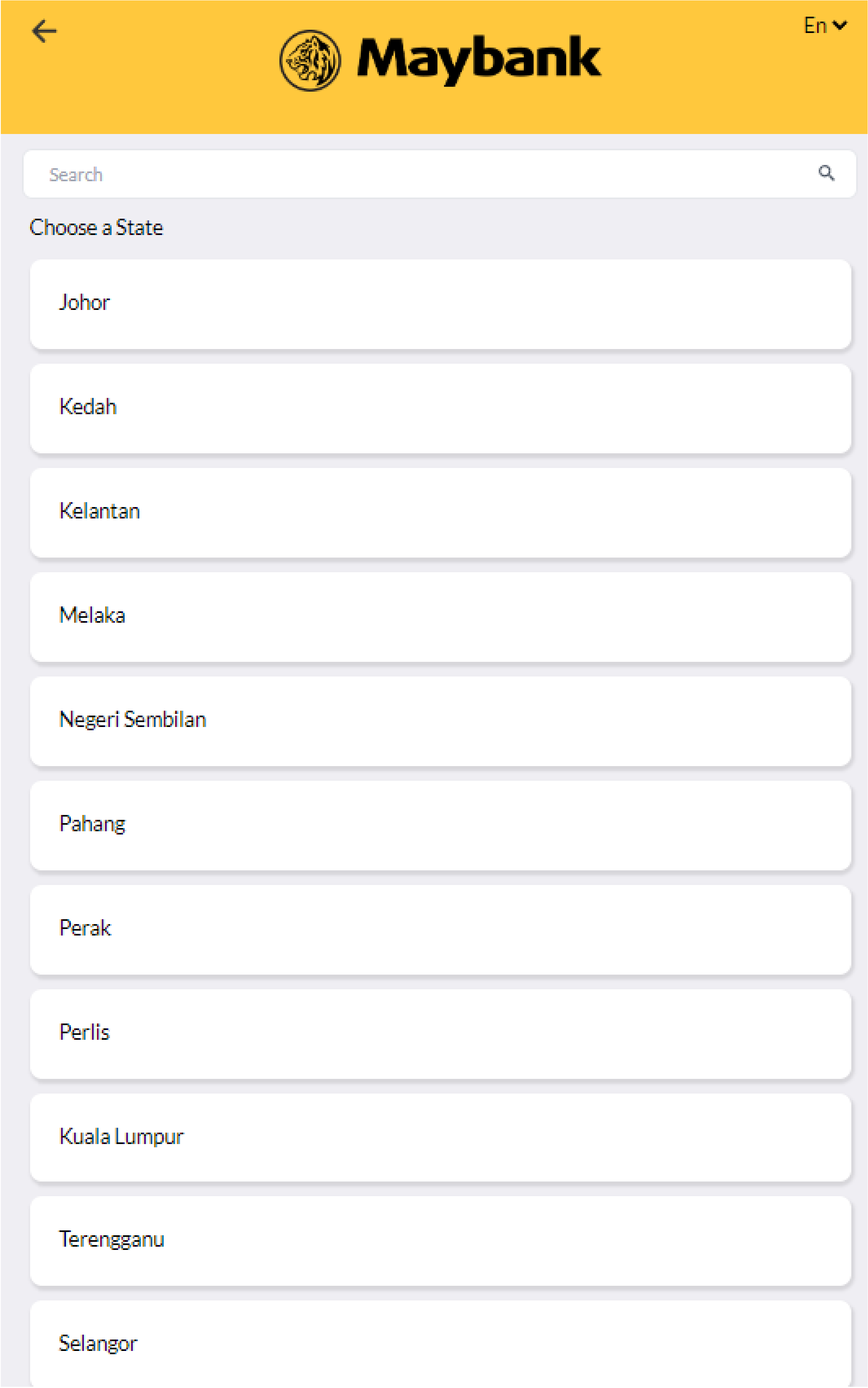


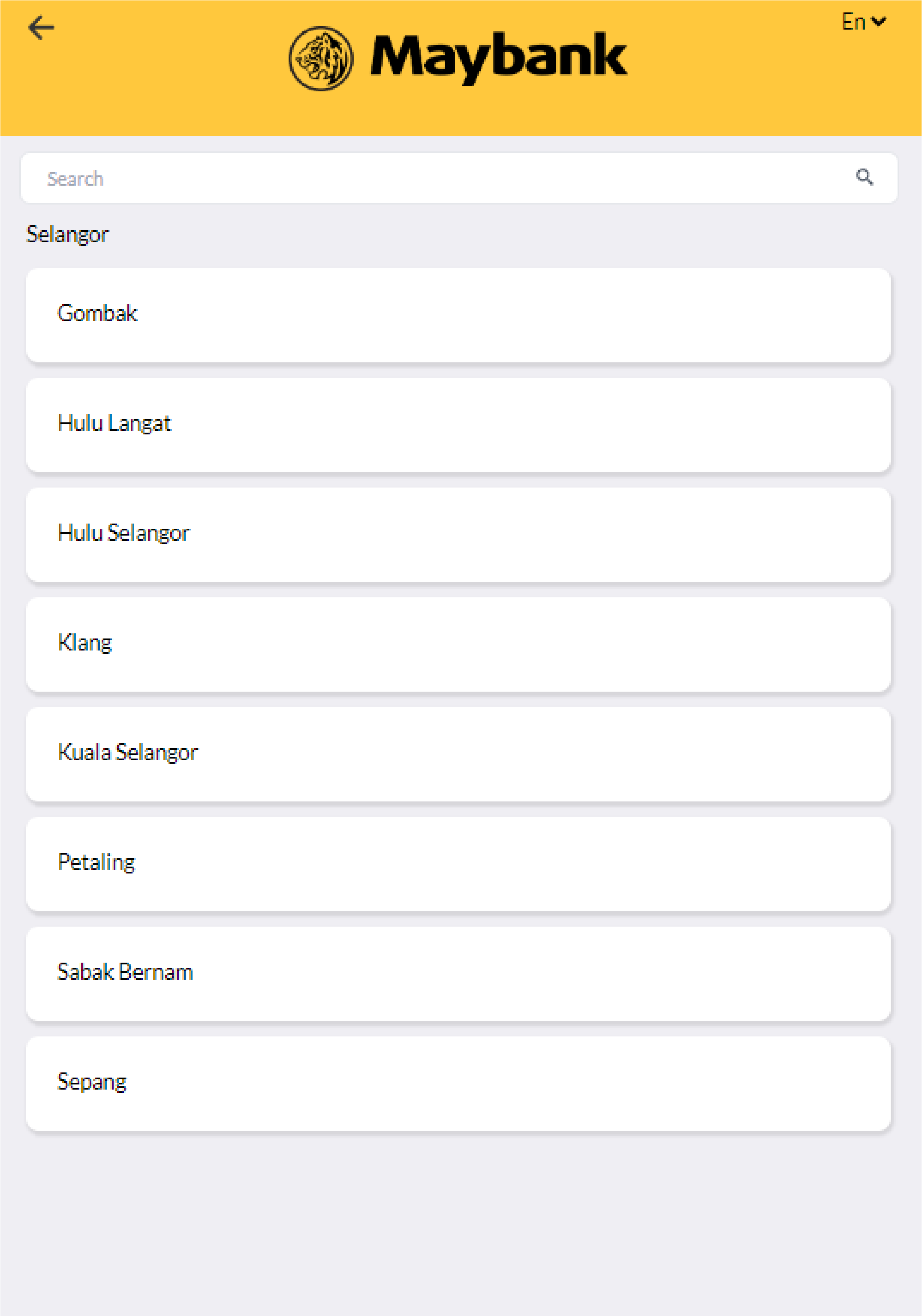
```
    # Enter IC
    ic = driver.find_element(by=By.ID, value=':r4:')
    ic.send_keys('587444156877')


    # Enter email
    ic = driver.find_element(by=By.ID, value=':r5:')
    ic.send_keys('exmaple@gmail.com')


    #click on the upload button
    upload_xpath = '/html/body/div[1]/div/div/div[1]/div[1]/div[3]/div[4]/div[1]/div[2]'
    upload_element = WebDriverWait(driver, 5).until(
    EC.visibility_of_element_located(
        (By.XPATH, upload_xpath)))
    action.move_to_element(upload_element).perform()
    upload_element.click()


    #click on the choose file
    choose_xpath = '/html/body/div[4]/div[3]/div/div/div[2]'
    choose_file_button = WebDriverWait(driver, 10).until(
        EC.element_to_be_clickable((By.CSS_SELECTOR, '.MuiDialogContent-root > div:nth-child(3) > div')))
    choose_file_button.click()

    #Choose picture
    time.sleep(3)
    keyboard = Controller()
    keyboard.type(f'C:\\Users\\Sam\\Desktop\\AI\\image{i}.jpg')
    keyboard.press(Key.enter)
    keyboard.release(Key.enter)


    multi_message = WebDriverWait(driver, 30).until(
        EC.element_to_be_clickable((By.CLASS_NAME, 'css-1xsto0d')))
    print(f'Test {i}:{multi_message.text}')

    driver.quit()
except Exception as e:
    print(f"An error occurred: {e}")
```

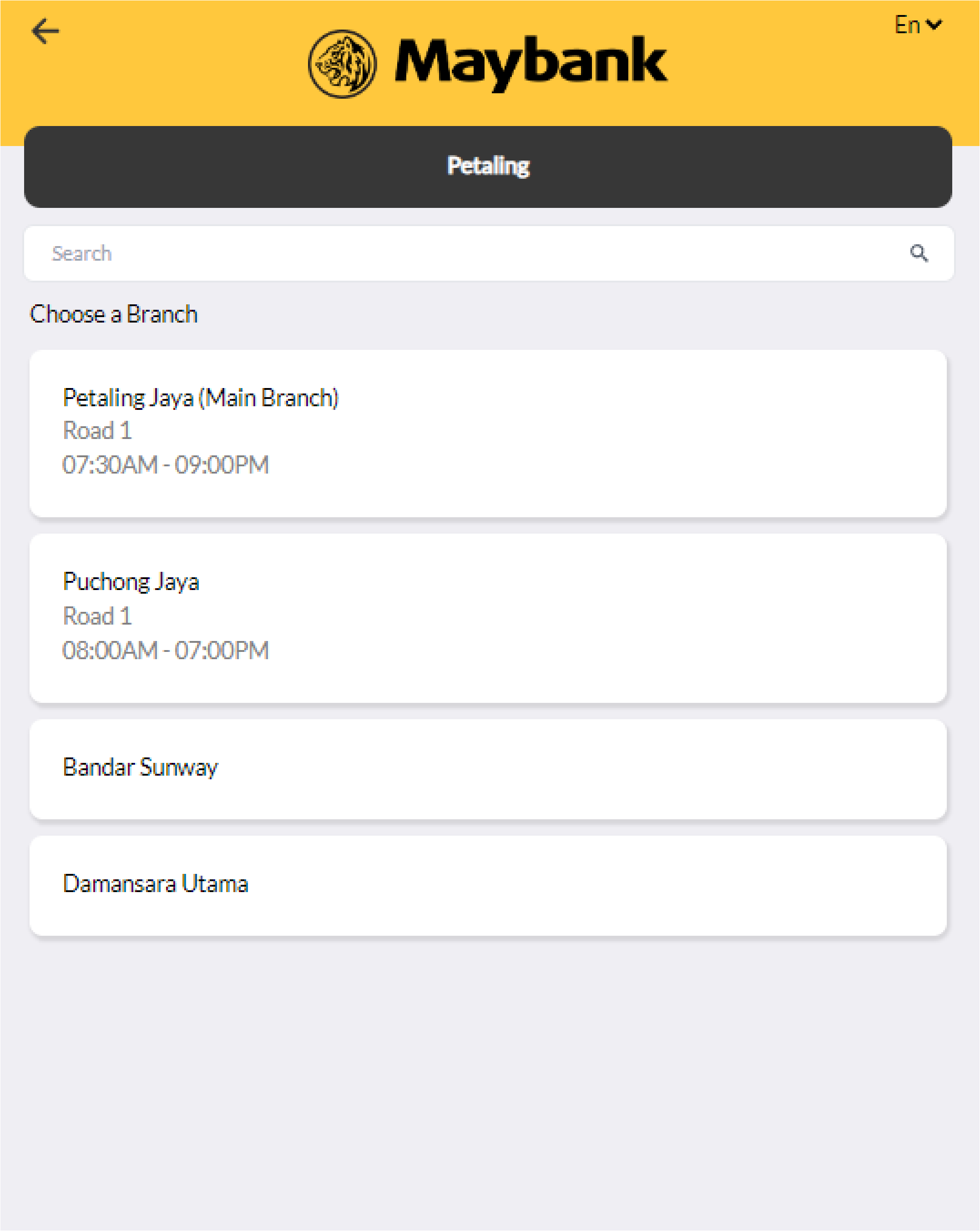


Welcome to

Customer Online Portal


Maybank
En
Search
Choose a State
Johor
Kedah
Kelantan
Melaka
Negeri Sembilan
Pahang
Perak
Perlis
Kuala Lumpur
Terengganu
Selangor


Maybank
En
Search
Selangor
Gombak
Hulu Langat
Hulu Selangor
Klang
Kuala Selangor
Petaling
Sabak Bernam
Sepang



Petaling

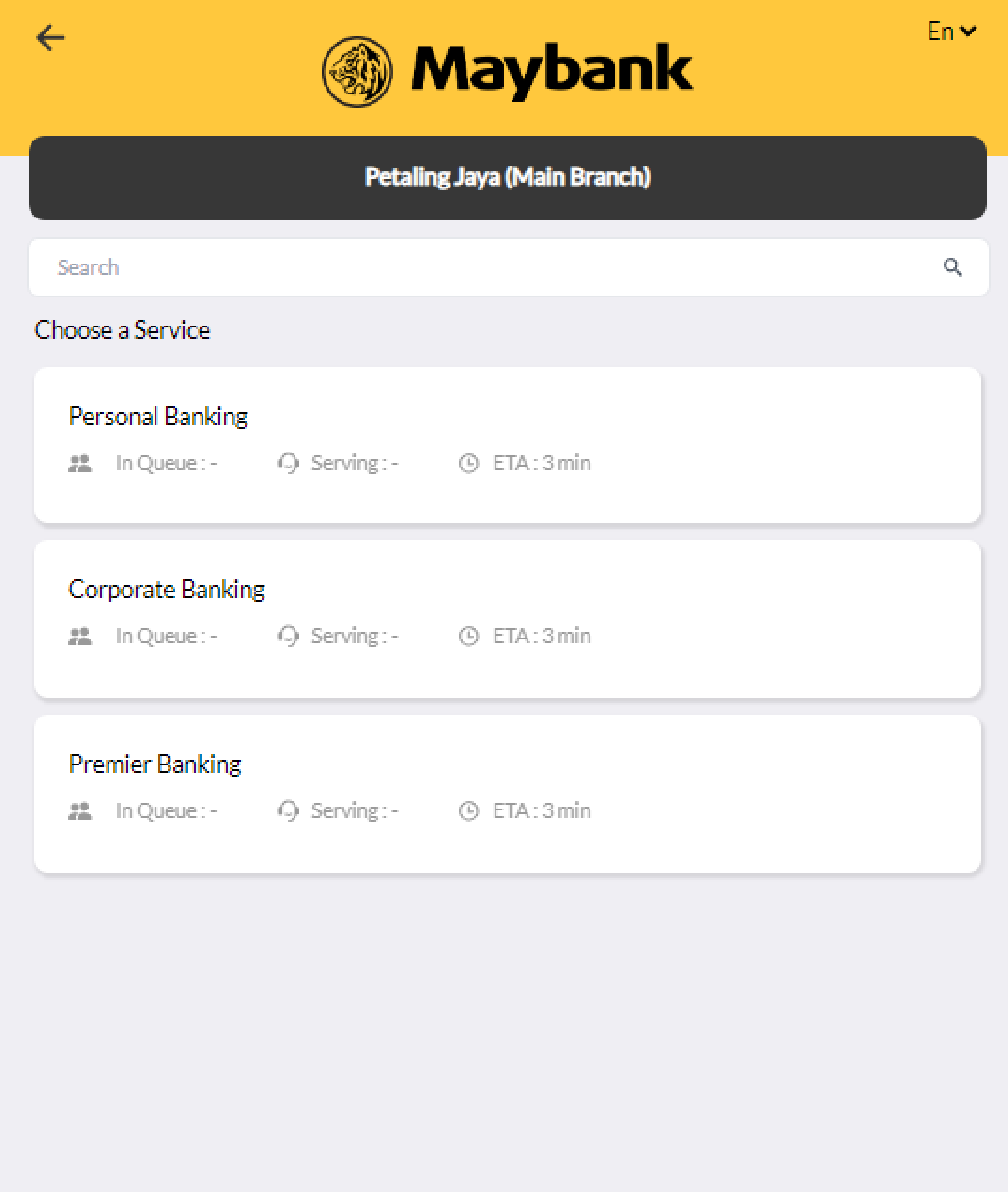
Choose a Branch

Petaling Jaya (Main Branch)
Road 1
07:30AM - 09:00PM

Puchong Jaya
Road 1
08:00AM - 07:00PM

Bandar Sunway

Damansara Utama



Petaling Jaya (Main Branch)

Choose a Service

Personal Banking

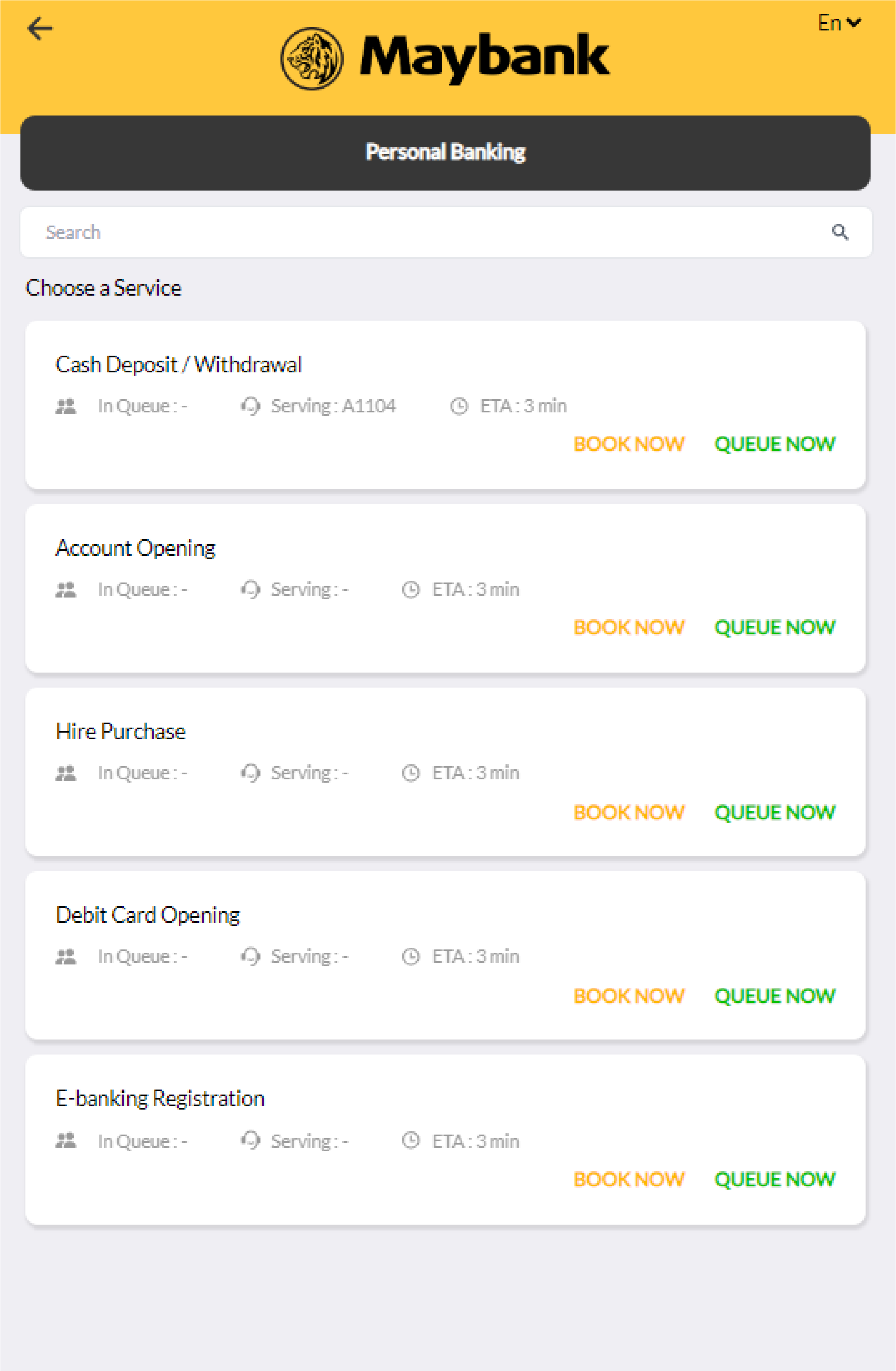
In Queue : - Serving : - ETA : 3 min

Corporate Banking

In Queue : - Serving : - ETA : 3 min

Premier Banking

In Queue : - Serving : - ETA : 3 min



Choose a Service

Cash Deposit / Withdrawal

In Queue : - Serving : A1104 ETA : 3 min

BOOK NOW QUEUE NOW

Account Opening

In Queue : - Serving : - ETA : 3 min

BOOK NOW QUEUE NOW

Hire Purchase

In Queue : - Serving : - ETA : 3 min

BOOK NOW QUEUE NOW

Debit Card Opening

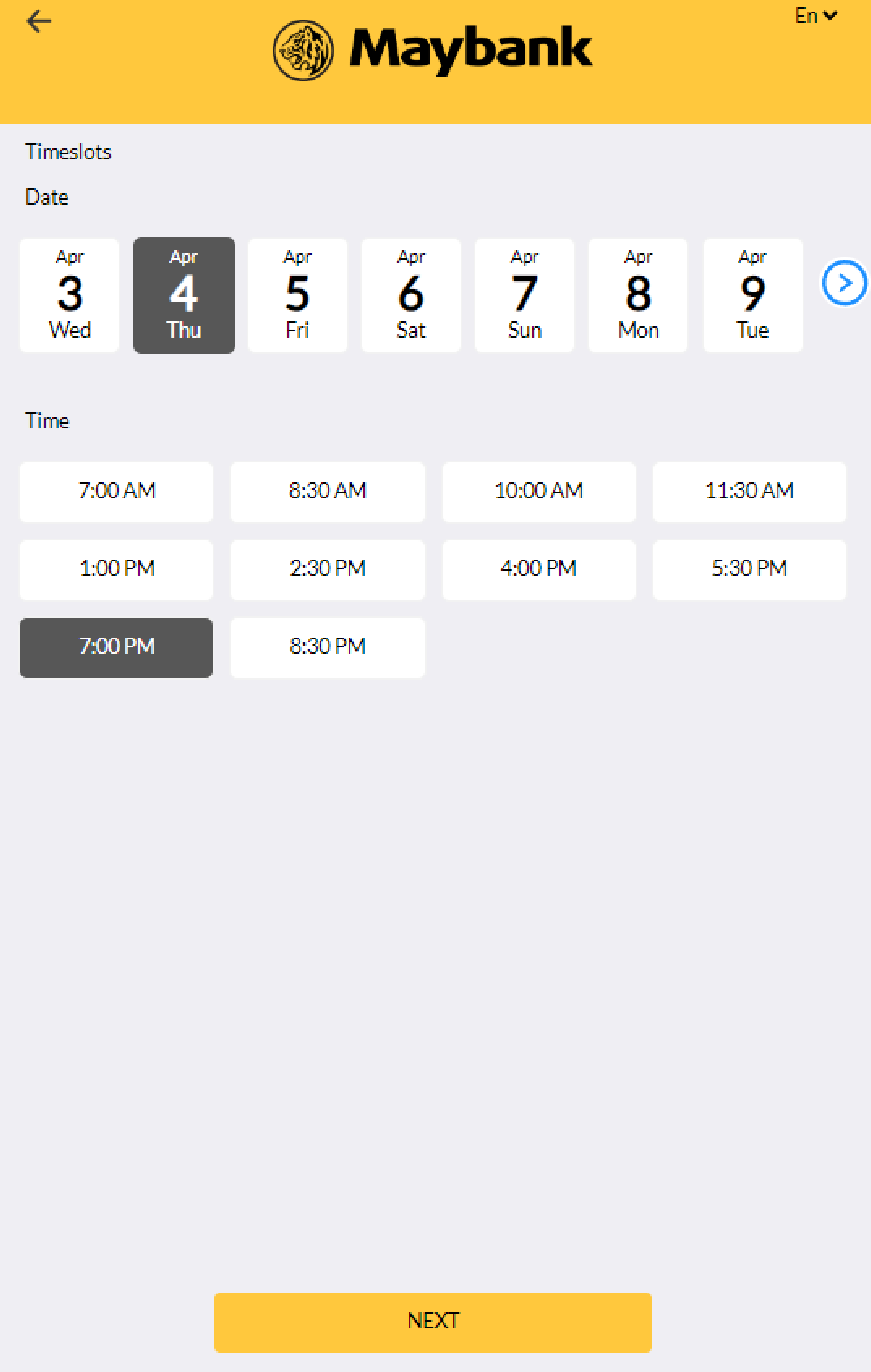
In Queue : - Serving : - ETA : 3 min

BOOK NOW QUEUE NOW

E-banking Registration

In Queue : - Serving : - ETA : 3 min

BOOK NOW QUEUE NOW



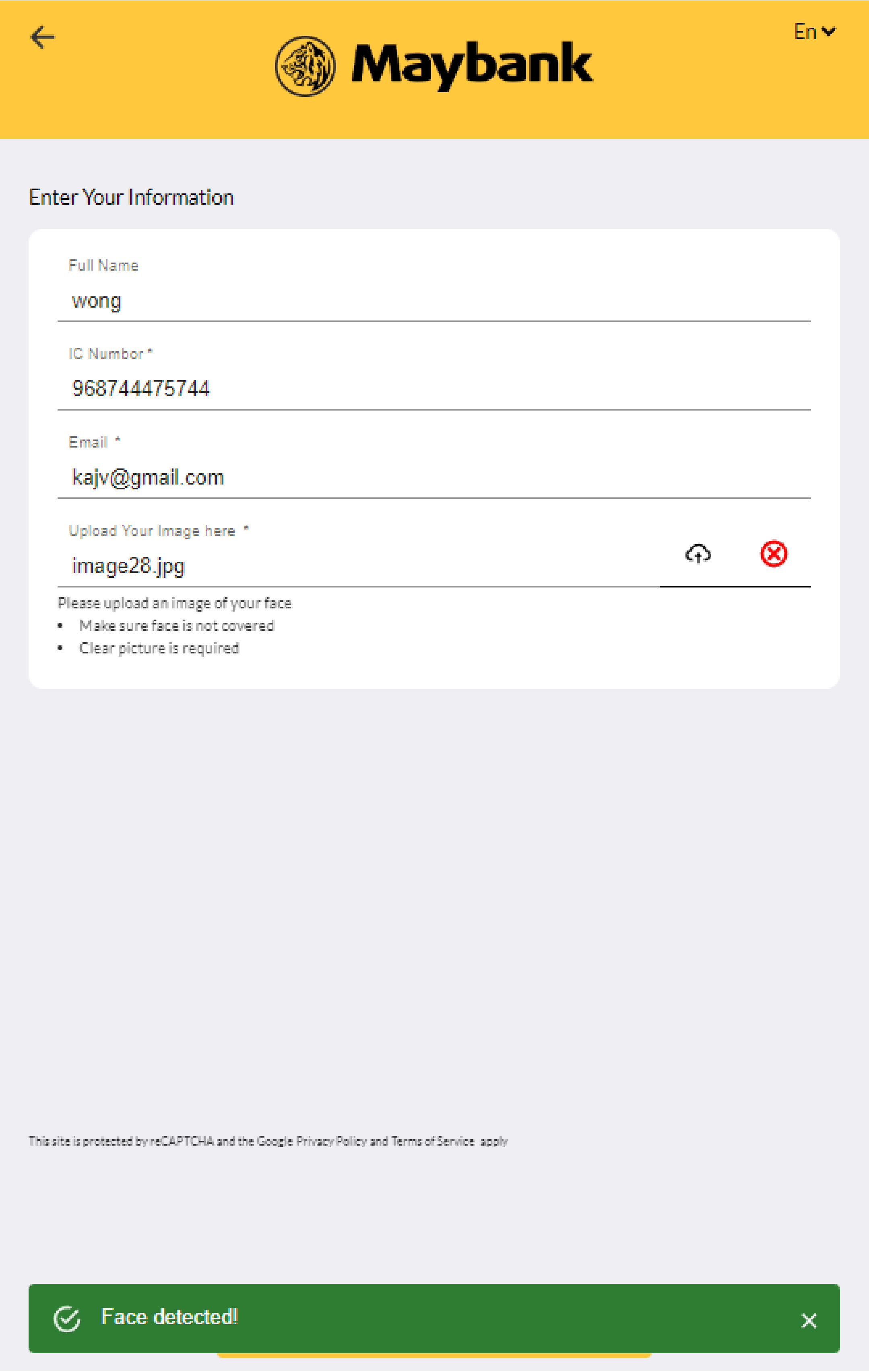
Timeslots

Date

Apr	[✓ Apr]	Apr	Apr	Apr	Apr	Apr
3	[✓ 4]	5	6	7	8	9
Wed	[✓ Thu]	Fri	Sat	Sun	Mon	Tue

Time

7:00 AM	8:30 AM	10:00 AM	11:30 AM
1:00 PM	2:30 PM	4:00 PM	5:30 PM
[✓ 7:00 PM]	8:30 PM		



Enter Your Information

Full Name
wong

IC Number *
968744475744

Email *
kajv@gmail.com

Upload Your Image here *
image28.jpg

Please upload an image of your face

- Make sure face is not covered
- Clear picture is required

This site is protected by reCAPTCHA and the Google Privacy Policy and Terms of Service apply

Face detected!