MTech(IS)

Intelligent Process Automation

Project Report

**INTELLIGENT TEMPERATURE SCREENING SYSTEM**

**Team Members**

CAO LIANG – A0012884E

GENG LIANGYU – A0195278M

HAN DONGCHOU FRANCIS – A0195414A

ONG BOON PING – A0195172B

TAN CHIN GEE – A0195296M

1.0 EXECUTIVE SUMMARY

Our intelligent screening system is aimed at relieving the financial and manpower pressure of small-medium enterprises (SMEs) during this period of novel coronavirus (covid19). In these trying times, SMEs might struggle to finance the cost of deploying a thermal scanner. Moreover, the manual process of temperature screening is a laborious task involving a lot of administrative work arising from recording employees’ temperature.

We have built a low-cost and flexible system that automates temperature screening by utilizing robotic process automation, face recognition, and Google Cloud artificial intelligence services. The system can be hosted on devices that can run a web-browser (e.g., laptop and smartphone) and featured a graphical user interface that is simple to use. The system architecture comprises: (1) UiPath to coordinate the functionalities of the system for each process, (2) face recognition using Dlib library with Python, and (3) number recognition using Google Cloud Vision API.

While the system is running, it will automatically identify employees using face recognition and check against records to determine if the employee should be under isolation at home. Non-isolating employees will take their temperature using a non-contact thermometer and present the reading to a camera. The system captures and records the temperature.

If the temperature is 38 degree Celsius or higher, then the system will send emails to alert the Human Resource manager and the employee’s manager. Employees who are supposed to be under isolation will be instructed to stay put while the system alerts his superiors. Employees can proceed to their office if they are not under isolation order and has a temperature lower than 38 degree Celsius.

2.0 PROBLEM DESCRIPTION

As of 24th April 2020, the number of covid19 cases has exceeded 12,000 in Singapore. Covid19 is a pandemic that has swiftly evolved from a localized public health emergency in China to a global crisis in a matter of a few months. The threat of this pandemic has forced governments around the world to take unprecedented drastic measures to control the spread of the virus. The daily taking of one’s temperature has become one of the defining measures implemented by the authority during the last few months.

**2.1 Validate the Business Case - Intelligent Temperature Screening System**

A temperature of 38 degree Celsius or higher (i.e., having a fever) is one of the clinical symptoms associated with being infected with covid19. Therefore, the action of temperature-taking is now a standard practice among businesses locally. The process of taking a person’s temperature typically required one or more staffs to man the temperature screening station. The number of staff deployed to perform this task will depend on the size of the human traffic and the temperature sensor that is employed. A temperature screening station equipped with an infrared-red temperature sensor hooked to a monitor can be manned with fewer staffs than one using a hand-held forehead thermometer.

Placing staff to man temperature screening station either displaces staff from performing their normal duty or add additional work onto staff. Manned temperature screening station also exposes the staff to the risk of being infected with covid19 through their frequent contacts with other people. This is a situation that is both an ineffective use of manpower and risky to the health of staff. In short, the pain-points of the current temperature screening process are low efficiency and high infection risk.

A better solution can be achieved through enhancing the process of temperature screening using the technology of Intelligent Process Automation (IPA) and Robotic Process Automation (RPA). We are confident that there is a valid business case for an intelligent system that minimizes the manning of staff while maintaining high efficiency of temperature screening.

3.0 BUSINESS PROCESS MAPPING

The team analyzed the temperature screening process employed by our respective organizations and discussed what the ideal situation could be like after employing IPA and RPA. The “as is” diagram is a typical flow of the processes of temperature screening (Fig 1).

The temperature screening process typically comprises the following three groups of people:

1. Employee
2. Receptionist
3. Employee’s Manager & HR Manager

The receptionist of the organization begins the process by updating the list of employees who are serving Stay Home Notice (SHN). This list is the blacklist record. The receptionist is tasked with identifying blacklisted employees and temperature screening at the entrance to the building. The receptionist is also tasked with updating the company record of new employee.

The receptionist checks against the blacklist when an employee shows up at the entrance to the building. If the employee is not on the blacklist, she will take and record the temperature of the employee. The receptionist will raise an alert to the employee’s manager and HR manager if the employee is on the blacklist or has a temperature of 38 degree Celsius or higher. Blacklisted employee will be told to remain put and feverish employee is instructed to return home (Fig 1). The employee can enter the building if he/she is not on the blacklist or have a temperature that is lower than 38 degree Celsius.

Figure 1. “as is” diagram of the temperature screening process

A screenshot of a cell phone

Description automatically generated

We re-designed the temperature screening process with incorporated RPA and IPA that will assist the receptionist to take employee’s temperature and image of the employee’s face. The receptionist only needs to press the start button on a graphical user interface to set up the camera and start the image capture and temperature-taking software running in the background.

Before an employee enters the building, he/she uses a forehead thermometer to take temperature and stands in front of the camera. The feeds from the camera will be fed into the image capture and temperature-taking software inside a computer. The image capture software will identify the employee by name through face recognition, and the temperature-taking software will read the numerical digits on the thermometer.

The system will check for the conditions: (1) employee is on the blacklist, and/or (2) employee has a temperature of 38 degree Celsius or higher. It will raise an alert to the employee’s manager if the condition(s) is/are fulfilled (Fig 2). A message “You are supposed to be in isolation. Please stay here and someone will look for you.” will be displayed to blacklisted employee. Otherwise non-blacklisted and non-feverish employees can enter the building.

Figure 2. “to be” diagram of the temperature screening process

A screenshot of a cell phone

Description automatically generated

**3.1 Product Plan – Intelligent Temperature Screening System**

The core of our AI solution relies on RPA software bots and Google Cloud AI (Fig 3). Specifically, our solution uses both UiPath and Google Vision API to automate half of the repetitive processes and reduces the bulk of work on the receptionist. The system structure comprises of a UiPath RPA application that operates in tandem with Python scripts that perform face recognition of employees and rule-based retrieval of information from employee records. The system also passes requests to Google Cloud Vision API to read the numerical digits of the reading from the thermometer.

Figure 3. Overview of the System Structure

**3.2 Market Research of Local Temperature Screening Systems**

We next conducted a quick market research for products that are similar to our AI solution. Thermal scanners developed since the 2003 SARS epidemic have played a vital role in mass screening of temperature among travelers and citizens. Such scanners help staff to identify feverish individuals without the use of thermometers. Sensor in the scanner feeds thermal images into a computer and displays feverish foreheads as red spots on silhouetted images on a computer screen. Each scanner costs close to $90,000.

These thermal scanners serve an urgent need during the 2003 SARS epidemic with the minimal of incorporated functionalities. The recent boom in commercialized usage of artificial intelligent in smart products has not gone unnoticed by makers of thermal scanners or government agencies.

Thermal scanner manufacturers, for example, D-Link and Megvil have introduced commercial products that can recognize faces and sent alert autonomously. Government agency Integrated Health Information Systems (IHiS) has likewise piloted their own AI-powered temperature screening system, iThermo. The iThermo system comprises a smartphone connected to thermal and 3D laser cameras. The system measures forehead temperature through recognizing human facial features from thermal images.

The iThermo system is available for subscription at $1,000 per month. The costs of both D-Link’s and Megvil’s thermal scanners are not freely available on their respective websites. However, it is reasonable to expect the cost to be at least a few $10,000.

We determined that there are existing products on the markets with comparable functions to our AI solution. However, our solution has the distinct market advantage of being at least a tiny fraction of the costs of these products. We can keep the costs low because we developed the system using common code library and used existing hardware such as a laptop and its webcam. Therefore, we don’t incur expenses on custom-build software and hardware that translate into cost-saving for our end-users.

4.0 SOLUTION

The process of temperature screening using our system is displayed in Figure 4. There are three sub-functions: (1) face recognition, (2) thermometer temperature reading, and (3) sending emails.

When an employee is presented to the camera, the UiPath application triggers the python script “read\_temp\_google2.py”, which updates the image\_login.csv file based on face recognition results, and updates the alert.csv file based on the blacklist.csv file and employee’s temperature reading results to generate the alert csv file that records the change in employee’s temperature. The UiPath application takes the changes from the alert csv file and sends an email to management if the employee is either having a fever or is blacklisted.

Figure 4: System Archtecture

A screenshot of a cell phone

Description generated with very high confidence

**4.1 System Design/Model – Components of the System**

1. User Interface: The web user interface comprises four functions that are (a) to start the system, (b) to stop the system, (c) to update new employee’s details, and (d) to upload new employee’s photo.

2. Folder of the Photos of Employee’s Faces: The system uses these photos to perform face recognition.

3. Python Application read\_temp\_google2.py: This python script performs both face recognition and the temperature reading tasks.

* The face recognition is powered by the Dlib based face\_recognition Python library. The code takes a real-time video stream and uses Dlib to convert images into HoG array to find the face area. It applies a 128-bit array encoding on each face. The cosine distance between the video-stream faces’ encoded array and the known encoded employee faces will determine the similarity of faces.
* The recognized face will be recorded to image\_login csv file with a timestamp. Image will also be recorded in the computer with the date-stamp and employee’s name saved as the jpg filename. The image\_login csv file is initialized with all known employees who are given a value of 99999 that represents non-detection at the onset. However, a date-stamp will replace the initial 99999 value once the employee’s face is recognized.
* The script checks against the blacklist csv file to determine whether the employee was blacklisted and forbidden from entering the company. A detected blacklisted employee is recorded in the alert csv file.
* The temperature reading sequence will be invoked after face recognition is completed. The application captures the real time images of the thermometer reading with OpenCV. Then Google Cloud Vision API will be invoked and the recognized digits and characters are extracted from the API results. The application determines the temperature digits according to regular expression, and add “.” if it is missing. The final temperature will be returned to the application main function flow, and if the temperature is not located successfully, the main function flow will try to continue to detect the temperature unless a blacklisted employee is detected.

4. UiPath RPA: This application contains two parts: Main Flow Sequences and Email Module.

* The main flow sequences automate the starting of the camera and temperature-taking of the employees through console commands during start-up, which are entered by UiPath RPA application. After start-up, the face recognition and temperature reading functions will be constantly running. Hence, the UiPath RPA is observing the alert csv file in an infinite loop. The main sequences will invoke Email module to send email if the employee triggered alert = 1 (i.e., having a temperature of 38 degree Celsius or higher) or alert = 2 (i.e., blacklisted employee).
* The Email Module reads the email configuration from external excel file which enables the email module to be flexible to changes. It will send out email message for two types of scenarios: (a) temperature of 38 degree Celsius or higher and (b) blacklisted employee. Each scenario has their own template in the email configuration which controls the email content and format. Also, the email receivers (Department Manager and HR Manager) are configurable from the email configuration file. The emails are sent through SMTP server, and the SMTP server configurations (server IP, port, account name, password etc.) are also included in the email configuration file. So even though we are currently utilizing Gmail server, it is easy to change to other SMTP servers to send out email by changing the email configuration file. The email module also includes the codes for testing the sending of emails based on different conditions. Therefore, it helps in debugging and deployment.

**4.2 Outputs of the System**

System Alert Output

1. Name: A string object. The system outputs the name of the identified employees that is stored in the records.

2. Alerts: A numerical value that is either 1 or 2. The system generates an alert of 1 when it detects an employee with temperature of 38 degree Celsius or higher, or generates an alert of 2 when it detects an employee on the blacklist. An employee on the blacklist is forbidden from entering the company. A value of 99999 indicates that an employee is not presented to the camera.

3. Temperature: A float value rounded to one decimal place. The system outputs the temperature reading that is recognized by Google Cloud Vision API.

Blacklist Records

1. Name: A string object. The name of the blacklisted employee that is stored in the records.

2. Blacklisted: A Boolean value. A value of 1 indicates that an employee is blacklisted, and a value of 0 indicates otherwise.

Emails

1. Alert Email for Employees with Temperature of 38 Degree Celsius or Higher

* Email will be sent to the department manager and HR manager to inform them that the system has detected an employee with a temperature of 38 degree Celsius or higher.
* The email content and format are configurable using the email configuration excel file.

2. Alert Email for Blacklisted Employees

* Email will be sent to the department manager and HR manager to inform them that the system has detected a blacklisted employee.
* The email content and format are configurable using the email configuration excel file.

5.0 CONCLUSION

We have achieved our objectives of building an intelligent screening system for SMEs. The system is a low-cost and labor-saving solution that can replace the existing manual process of temperature taking among many SMEs. In the course of building this system, we have further enhanced our knowledge of how to turn our knowledge into engineering solutions. The resultant product is timely as we hope to play our parts in helping the community combats covid19.

APPENDIX A. USER MANUAL

Prerequisites to run ipa-web

1. Need java8/JRE8 to be installed.
2. Make sure the java/JRE has been added in path env variable (by default, it is there).
3. Make sure UiPath is installed so that user is able to trigger the "start process" properly.

Files needed:

1. config.properties, config file for the UiPath and processes which will be closed during stop processes.

1.1 uiRobot\_path, this need to be updated to user local installation path, like below:

C:/Users/AppData/Local/UiPath/app-19.12.0-beta0061/UiRobot.exe

1.2 temperature\_uiPath\_file, this need to be updated to sequence.xaml in current project, like below:

C:/Users/Temperature\_Checks\_for\_SMEs/executables/executables/Sequence.xaml

1.3 process\_list, while user click "stop process", processes configured here will be closed, like WindowsCamera, UiRobot etc.

The process name might be different due to different machine, so there is a need to configure this manually, otherwise users will have to stop the process manually.

2. employee.csv, includes the employee details input from WebUI.

3. known\_people, when the users upload employees’ photos, it will be added into this folder.

How to start ipa-web

1. Go to the ipa-web jar file folder and update the config.properties to reflect the correct file path. For the use of each prop, please refer to above config.properties.

2. Go the command window and cd to the folder which contains the ipa-web-0.1.0.jar. E.g. cd C:\code\_repo\Temperature\_Checks\_for\_SMEs\executables\

3. Execute the command below, and there will be output generated.

java -jar ipa-web-0.1.0.jar

4. Web server started successfully if you see the following output.

"Root WebApplicationContext: initialization completed in 2832 ms"

5. Open any browser and access URL "http://localhost:8090/"

How the UI looks like

After opening the browser, you should see the UI as depicted in Figure 5.

Fig 5. Main Page of User interface

![A picture containing clock

Description automatically generated]()

The “Start Process” and “Stop Process” is used to start and stop the system. Next, the “Add Employee Details” is a function that allows the user to update the details of new employees into record (Fig 6) while the “Upload Employee Photos” loads the photo of new employees (Fig 7).

Python Environment

1. Anaconda needs to be installed with an env named “ml1P13”.

2. requirements.txt includes all the dependencies for this project.

3. All the files in the executable folder are needed during runtime.

Fig 6. “Add Employee Details” Page

![A screenshot of a cell phone

Description automatically generated]()

Fig 7. “Upload Employee Photos” Page

![A screenshot of a social media post

Description automatically generated]()

**Tan Chin Gee**

APPENDIX B. INDIVIDUAL PROJECT REPORT

Personal contribution to group project

My contributions to the project were:

(1) initial coding of face\_recognition.py,

(2) writing, compiling, editing, formatting, and proof-reading the report,

(3) crafting and narrating the presentation video, and

(4) coding of aruco\_detect.py.

What I have learnt

Through this project, I gained a deeper hand-on understanding of creating a working prototype product that incorporated what I had learnt in the Intelligent Software Agent course. It is not just the technical skills and know-hows that I have gained through the project. More importantly, I summarized that the other lesson that I learnt was the impact of AI in addressing pressing societal concerns. I will elaborate further on what I meant by my insights on the impact of AI, after discussing the technical skills that I had learnt.

The Intelligent Temperature Screening System comprises the following modules: (1) UiPath, (2) face recognition using Dlib pre-trained neural network, and (3) Google Cloud Vision API for number recognition. The UiPath software bot ties together both the python codes that implement face recognition and reading of temperature into a functional system. This showcased to me the role of UiPath as a coordinating system that is able to pull other functionalities as required to achieve an intended goal. A simple RPA bot can become an IPA bot just by incorporating AI functionalities, for example, through the Google Cloud Vision API.

Working with the codes of face\_recognition.py had taught me the finer points of how to write python codes to implement a face recognition system. More importantly, I now understand the basic concept of transfer learning because I had used the pre-trained weights that the author had trained the model on, instead of performing my own training on the model. Because the author had done a good job in training the model, the script’s performance on face recognition was quite high without needing further training on my part.

Although it was a shame that the aruco\_detect.py wasn’t incorporated in the final project, I had gained valuable lessons in how to track and detect a moving object in video feed through the use of ArUco markers. I was initially stumped by the difficulty of detecting and tracking the display of the thermometer by trying to detect the edges of its display screen. After much further readings, I learned about the ArUco marker detection i.e., aruco module in OpenCV that finally helped me to capture the thermometer display in a video feed (Fig 8).

Fig 8. Screenshot of Thermometer Display using ArUco Marker Detection

A picture containing green, sitting, microwave, oven

Description automatically generated

On the social impact of AI, we have crafted this project into an AI solution that serves the immediate need of the community that is beyond just to fulfil a requirement of the course. This fully demonstrated that our knowledge and engineering skills can be brought to bear on important issues that our society faces.

How I can apply the knowledge and skills in other situations

I worked in a charity and my work involves conducting research on child abuse. In my field of work, I encounter challenges incorporating the latest research into recommendations that social workers can then use to help their clients e.g., children and youths. It is difficult to keep up with the large volume of new research that is published monthly.

By using RPA and IPA, I can lessen my load by programming the software agent to automatically download articles that fit certain keywords e.g., abuse, bully ...etc. Then the bot can make a call to Google Cloud for services to summarize the key findings of these articles. The summarized output will improve my productivity by reducing the time needed to read these articles and that will help me to complete my recommendations to the social workers quicker.

**Cao Liang**

Personal contribution to group project

My contributions to the project were: (1) Email module for sending emails, (2) Implement python function for thermometer temperature reading using Google Cloud Vision API, and (3) Write parts of reports on system architecture diagram, and relevant parts on Email module and temperature reading function, and (4) Perform integration with other team members on Email module and temperature reading function.

What I have learnt

I learned the UiPath module development from Email module. The email module was developed with a separate xaml file and the main UiPath application invoked the email module with arguments. This approach exhibited many advantages, and the most important was that it allowed to split the large UiPath RPA application to be smaller parts, and different people could work on different parts concurrently, so it speeded up the project development considerably. It also allowed the individual module to be tested thoroughly and could be easily for debugging and improvement. I developed the test xaml file for testing email module, and found it was useful for integration and debugging.

I also learned the good practice to store the configuration outside the xaml source file. This allowed the integration, testing and deployment much easier, since all the configuration could be edited without any change in the UiPath application.

The Google Could Vision API was utilized for thermometer temperature reading. I learned to setup the google cloud API account, and configure the Vision API service, and integration with python script. I also learned the abundant and powerful solutions that were available from the Vision API which were helpful for my future wok.

The UiPath integration with email module was not smooth, and it was caused by the limitation of UiPath. UiPath was not good for handling complex logic, and hard to debug for the different execution path due to different scenarios. So, I learned that the complex logic should be avoided to be handled by UiPath, and it was better to be implemented in a separate module, or an external process such as python script.

How I can apply the knowledge and skills in other situations

I will plan to automate certain tasks in my work with the techniques learned from the course and the project. One of them was the search for malware hash values and malicious domain names. I worked as threat and forensic consultant, and dealt with malware daily. There were many malware samples that I should perform certain routinely tasks on, including search its file hash values, and its domain names. These tasks were tedious, but was compulsory to perform. Therefore, I plan to write python script with TagUI to complete the task, and it will save my time and get the results much faster.

**Ong Boon Ping**

Personal contribution to group project

A) Construct the UiPath sequence

B) Create face recognition program based on ageitey’s face\_recognition library.

C) Integration of the face recognition program, UiPath with the digit recognition program.

What I have learnt

I have learnt that face recognition is one of the important areas in IPA. It helps to recognize the human face and able to help the system in identify the correct human face just based on the face image given.

I also learn that face\_recognition program can also be triggered by UiPath and UiPath is able to help in sending email warning. This automation is certainly helpful. This is also an example where automation can be done even though the face recognition script is mainly done in Python.

I also learnt that the Google Vision can help to recognize the digit with great accuracy. With proper integration, the temperature reading and human face can be recognized in great accuracy.  
   
How I can apply the knowledge and skills in other situations

I can apply this project in my work in the future. UiPath is good in RPA automation and can be applied in my industry.

In electronic testing, data can be processed in python AI scripts easily. However, the action cannot be taken automatically due to different operating system used across testing equipment and servers. Hence, UiPath can come into place and automation can be done intelligently.

**Francis Han**

Personal contribution to group project

My contributions to the project were: (1) Leading the team and formalizing the project requirements, (2) Doing the first level of integration of the different components of the system (3) Performing SIT testing (4) Do the first flow of the script for the video.

What I have learnt

Getting the different components to work together took a lot of time from all of us. Especially with the Covid19 situation, we had to integrate, test and review repeatedly and remotely. Fortunately, we already know one another well. Well enough to know who to look for to fix which problem and to take up which tasks.

I find that the UiPath is a very suitable tool for automating workflows as well as doing non-complex activities like sending emails. In fact, it’s functioning like a virtual human orchestrating all the different components and functions. But not the complex stuffs – this is why we have the Python scripts working alongside UiPath.

I had the chance to test the various text recognition API from the cloud vendors. Only Google Cloud could cut it. While pre-built API functions have the advantage of simplifying deployments, the robustness depends a lot on the actual use cases and whether they are what the prebuilt API functions are trained for.

How I can apply the knowledge and skills in other situations

This is the first time I have worked on an RPA project integrated with Face Recognition and Text Recognition. Though the project is intended to help SMEs to do employee screening, the same framework can be used for many other purposes.

At my workplace, I sometimes discuss automation solutions with customers. This experience would come in useful in some of the conversations.

**GENG LIANGYU**

Personal contribution to group project

(1) Build the Web GUI as a centralized place to control the system

(2) Troubleshooting the python code and UiPath issues.

(3) Integrate all components together and did the performance tuning.

(4) Test and package the project.

What I have learnt

I have learnt how to use UiPath to automate the whole flow with abundant build-in features in it. We are able to avoid lots of complexities while integrating different systems.

I also learn that how to use face recognition program, and how it is being implemented in detail.

As we are also using google cloud APIs, this helps me sharpen my skills as well. With the extensive APIs which we can rely on, it helps to build AI systems in a more efficient way.

How I can apply the knowledge and skills in other situations

With what I have learnt in UiPath, I can use those to integrate different systems which don’t need high performances, instead of starting a coding project from scratch. It is much more efficient and can help reduce the cost as well.

For the google cloud APIs and face recognition, I can apply these knowledges in my future works when we start to build the AI systems.