PERSONAL DESK ASSISTANT

Tamanna Kar

Computing A Level Project – Tamanna Kar 6170

Centre No. 14285

Contents:

1 Project Proposal

Description

I would like to use my existing knowledge of robotics from my recent participation in the school robotics club (where we designed and programmed a robot using a given python library) in order to create a “robot” which can link to your phone/pc and using gesture recognition through machine learning in order to read out the time or just read out your calendar notifications when needed.

Why this Project?

I wanted to do this project because I understand the struggle of long-sighted people since both my parents wear glasses and always forget them and then ask me to read the time for them at home, for example. Developed even further, this project would be helpful for any visually impaired people or otherwise who want a simple way to know about their notifications without being next to their phone so that they don’t miss anything important and stay productive when working.

Comparing to existing systems

Currently, major systems have an AI helper like Apple’s Siri or Amazon’s Alexa however they don’t use gesture control and they are not specialised for productivity. So I think my idea will be useful since it is specific and personalised.

Requirements

Robotics side will include making use of a programmable microcontroller/ circuit board, such as Arduino, which can take on many functionalities, using camera/ light/ motion sensor for input and speaker for output.

If instead I use a Raspberry Pi Pico, the programming side will be done using Thonny IDE with MicroPython which facilitates for coding and testing/ debugging. Also using python API for retrieving phone/pc calendar notifications, using python libraries for processing, sorting this data etc. and using python API for text-to-speech in order to send output to the "robot".

2 Analysis Stage

2a Problem Definition

I am aiming to produce a IOT device for anyone who wants to improve their productivity and not be distracted by their devices. All day, every day people are surrounded by technology with their mobile phones and smart devices/ wearable technology (eg. smart watches) and there are constant distractions from every direction.

One option to solve this is to make your phone silent/ disable notification to smart devices. However, this is only appropriate solution in the short-term because this will make you want to physically check your phone more, taking focus away from your work. It could also make you more stressed in case you miss vital information from someone or an activity on your calendar.

This is a major problem that all people face, children who need to focus on schoolwork, working professionals who have a job to do, college students who have to balance studying and socializing, parents who have to manage home/family and working time etc. etc.

All these people have different working/studying habits and different time management, but they all need to be productive in order to accomplish their goals, therefore they need a product which will allow them to manage their time in their own way by helping them keep focused but get important reminders when needed.

Currently many people use many different systems to do this such as timer apps. These apps make use of time management techniques such as the pomodoro timing technique or others to encourage people to organise their time in a set way and keep track of how long they work. However, major downfalls in this is the use of active phone screen for timer, so people have more attention on the phone screen rather than working, and lack of notification management, so notification from other people and apps can still interrupt focus. Also enforced working time, which can be difficult when someone has different sized tasks, different concentration levels for specific tasks or they have other activities included (not always at their desk).

These problems are similar to other productivity apps for example for notification managers or note taking apps. Overwhelming complexity of features and/or the use of phone screen are the problems causing lack of productivity, especially in current circumstances with COVID-19 causing people to change from active office/school environment to working from home with distractions. Around 20% of businesses reported "reduced communication", "negative impact on working culture", as well as "reduced productivity" over time from new "work from home" changes – www.ons.gov.uk, from the Business Insights and Conditions Survey in the UK, May 2022.

A different reason why I feel my product is useful is that it can greatly aid the working habits for people visually impaired or even those with long sightedness (reading glasses) since my product will not only have no screen for input – instead, gesture recognition – there will also be auditory output (speaker with text-to-speech conversion). This will make it much easier for visually impaired people, or otherwise, to interact and manage their time in order to be more productive in the long-term.

2b Justification of Computational Methods

I think the problem with distractions and productivity in general is solved best using computational methods because:

- It is more efficient to make use of technology
  - » I will be making use of computer vision and machine learning in order to implement gesture recognition to trigger certain outputs
  - » This relies on computational power/ speed/ accuracy, all of which are higher than if a user had to directly give input to an app or write physically in a journal etc.
- It is more adaptable to different people's working styles/ time management
  - » I plan to draw data from online sources (and/or python libraries) such as for date and time, and since I will be building this as a personal project I will also be feeding data from my personal calendar in order to test its use/ reliability/ effectiveness
  - » In reality this would mean a user would be able to have their own personal data linked to this device and it would be personalised to their needs and requirements
  - » The design of the product inherently means that a user will use it in their own personal way, interacting and engaging with it in their own time as they see fit which automatically makes it adaptable to their working style/ routine, unlike written calendars or management apps which are tailored to certain styles or have excess features which are confining and not able to be personalised
- It is more accessible for people to use everyday
  - » I plan to design it as a stand-alone IOT device, to be used as a "desk robot" but could be placed anywhere similar like the bedside table or a windowsill
  - » The overall design making use of hand gestures and audio output lends itself to being child/elderly/disability friendly as well as accessible to all people at any time of day in any working environment (typically home) as opposed to a written diary or fridge calendar which is not engaging or helpful
- It is more reliable since it is automated and takes effect in real time
  - » The standout benefit of a computational method solution is the automation because this means the user is not constantly focusing on the phone screen or listening out for their smart watch to buzz
  - » Use of automation through computational methods also makes it much more reliable since it will act as an "always-on" system, and which will be able to provide information in real time – calendar notifications/ time-sensitive reminders – and without the user having to rely on their own memory to keep checking
- It is more engaging and so more likely to be used repeatedly
  - » Since my product will not just be another software implementation of the same old ideas and it will be a physical project with real time response I think it will be much more engaging than mobile apps or websites
  - » Once again this lends itself to child/elderly/disability friendly widening the target user group to more people who will find the product useful
  - » The product also makes of speech output, which will appeal to both the busy worker who moves around and away from their desk, but also the easily distracted child, making it more engaging and usable for all situations and types of people
- It is more versatile since it can draw from existing data such as online calendars
  - » Other systems require users to tediously input data manually so that the app, for example, can set up a timer/ reminder, whereas my product will use IOT set up to

connect online and retrieve prewritten details for events and reminders, for example, from google calendar using python APIs
    - Since it uses the users own data, it will be highly personalized and fully customisable just by setting whichever details you wish in your own calendar which is much more efficient than manually inputting data into a new system like mobile app

2c Implementation of Computational Methods

- Problem recognition
    - The overall problem is a combination of implementing hand tracking solutions to machine learning model for gesture recognition and IOT device set up to draw data from online services and making use of text-to-speech conversion to output this
    - This will be done using many different python APIs, built-in libraries, hardware specific implementation and a newly trained machine model for gesture recognition through computer vision and object detection processes
- Decomposition
    - The problem can be broken down into different steps and sub-problems helping to simplify and spread out the workload and programming efforts required in an efficient way
    - For example, the process of hand tracking can be broken down into steps: camera (hardware) input → image analysis → hand tracking API implementation → data retrieval → data feed into ML model
- Abstraction
    - There will be a lot of abstraction used in this project since real life image data is being analysed but only the hand in particular is retrieved and only the positioning data will be used in the ML model – this is an example of multi-layered abstraction ensuring that only the most important data is kept, and the model is not overwhelmed/ distorted by irrelevant information such as the image background
    - Similarly, the data retrieved from online sources will need to be cleaned and sorted before being fed into text-to-speech for output using different hardware implementation
- Reusable components
    - Image analysis could be implemented as a function which is called with every image frame before being fed into the ML model
    - Hardware interaction for input and output can also be implemented as reusable subroutines for each time they are used
    - Much of the online data retrieval will be prewritten series of steps which will be recalled each time depending on the data required denoted by the gesture, for example a peace sign could trigger it to output time whereas rock and roll sign might trigger it to output the date

2d Justification of Use of Python

- Has a lot of online documentation which is helpful during development process
- Many different libraries and APIs have been produced for new/extended/improved functionality and ease-of-development
- I have already learnt this language, and this is a good opportunity to develop those skills
- Comes with MicroPython/ CircuitPython variants for simpler hardware interfacing with Raspberry Pi Pico

3 Project Investigation

3a Initial Questionnaire for Stakeholders

Part A: to gain insight into experience with gesture vs voice control products/ experiences

1. Do you own/ use any type of voice-controlled "desk robot" and which? eg. Amazon Alexa, Siri (on mobile), Google Nest etc.
2. If so, how often do you use it and mainly for what uses? If not, does it appeal to you and/or would you want to use it for any uses in particular?
3. What would you consider its best features in terms of design and functionality?
4. What would you consider its major downfalls in terms of design and functionality?
5. Can you describe your experience with them and how they have helped you in your personal life in terms of ease of use and productivity?
6. Have you come across/ used any gesture-controlled devices and which? eg. VR headsets, certain earphones, gesture perceptive smart/wearable devices/controllers/appliances etc. etc.
7. If so, how often do you use it and mainly for what uses? If not, does it appeal to you and/or would you want to use it for any uses in particular?
8. What would you consider its best features in terms of design and functionality?
9. What would you consider its major downfalls in terms of design and functionality?
10. Can you describe your experience with them and how they have helped you in your personal life in terms of ease of use and productivity?
11. What is your opinion on comparing gesture and voice control in the home for working/studying?

Part B: to gain insight into working habits and expectations more specifically

1. What do you currently use for time management, organisation, keeping reminders etc. and can you be specific?
2. What would you say are the benefits and drawbacks of your current method(s)?
3. Any feature(s) you would consider a requirement in a productivity centred device/technology?
4. Any feature(s) you would appreciate having in a productivity centred device/technology?
5. Any feature(s) you would deem unsuitable for a productivity centred device/technology?
6. Anything else you would like to add?

Explanation:

Q A1-2 is to gauge the user's experience level with voice-controlled products
Q A3-5 is to understand the user's opinion and experiences with the products they have used
Q A6-7 is to gauge the user's experience level with gesture-controlled products
Q A8-10 is to understand the user's opinion and experiences with the products they have used
Q A11 is to gain better understanding of peoples' perception of gesture vs. voice-controlled devices

Q B1-2 is to find out about their current working style and methods for productivity/ organisation
Q B3-5 is to find out about their viewpoint on productivity/ organisation technology
Q B6 is to allow for any further comments/ ideas etc. etc.

Responses

I have chosen two stakeholders who I feel would benefit most from this type of product as they have busy schedules, majority of their work is at a desk and a product which does not rely on a screen and instead acts as an always on “desk gadget” is most effective to their routine.

Feedback from Person A:

Person A was clear about not having much involvement which voice-controlled/ gesture-controlled devices/ systems but showing clear enthusiasm for the concept. They considered productivity and in particular calendar management to be the most effective use-case of smart devices and they also highlighted a key benefit of gesture-control over voice-control being side-by-side functionality when in a working environment (ie. can’t say voice commands aloud).

Person A also talked about calendars being the main source of organisation/ productivity tool that they use and that they are deterred from using other tools when they require extensive setup and/ or are complicated to use. Person A also mentioned how effective a proactive alert can be compared to an easily miss-able and unengaging notification alert. They also highlighted a major factor in effectiveness which is other types of calendar information such as all-day or single reminders being forgotten when focusing on events.

Feedback from Person B:

Person B had much more experience with smart systems such as Samsung Bixby and google voice assistant on Chromecast/ Google tv. They highlighted a major downfall in voice-controlled systems being the lack of accuracy and extensive setup required especially as language/ accent barrier causes issues with ease-of-use.

Person B indicated a more on the go use-case of gesture control being when you don’t want to touch the keyboard/ touch screen due to being in the middle of a physical task. However, they also pointed out a downfall in gesture-controlled systems being unwanted input since any movement can be mistaken for intentional input. Person B agreed with Person A, re-iterating the importance of TTS voice output similar to a real personal assistant as this provides much more useful information compared to just a notification tone.

Person B highlighted a particular concern being privacy and since the main plan of the project is to be on a stand-alone IOT device I think this will ensure that it will not be dependent on any other accounts/ logins/ devices etc.

Summary

I feel important points to note are:

1) Simple user-interaction and little-to-no setup should be key features to hook users
2) Gesture-controlled commands are more flexible and allow for multitasking
3) Voice output (text-to-speech) are more proactive therefore more engaging + effective
4) Gestures will be specific hand positions as opposed to easily misinterpreted movements
5) Both calendar events and reminders will be considered – more widely accessible

These will be the overall features I will be using to design the product and the base considerations that will go into writing the user requirements.

3b Market Research and Analysis of “Smart” Devices

IOT device: Fitbit Smart Watch

I own a Fitbit Smartwatch so I can analyse its use and functionality from personal experience. The Fitbit Smartwatch is highly customizable wearable technology, making it highly adaptable, and it provides health and fitness tracking and information for the user. It requires a lot of data such as time, date, location, and personal information of the user in order to function.

It makes use of a touchscreen, side buttons and motion gestures such as twisting your arm up to read the time. It also uses an always-on sensor to trigger turning on the screen which is much more energy saving. It has a simple software interface which shows real time data and analysis however much of its hardware/ sensors are abstracted away or hidden within the overall design, making it more user friendly in general.

What can I apply?

I could also make use of the sensor on/off trigger since an IOT standalone device requires some well thought out energy saving features.

Also I will take into consideration the benefit of simple gestures for individual tasks to make the product as intuitive as possible and to make it simpler to understand and use.

Gesture-controlled System: Flutter

Flutter is a gesture-controlled program (no motion) to control media such as music or films using input through a webcam. It is supported on Windows Media Centre and other media sources such as YouTube. It has specific set of features eg. start, stop, skip etc. etc. Most importantly it is optimised for an SPI interface camera which stands for Serial Peripheral Interface. It can run at high speeds within short distances, and an SPI's data rate can go up to 60 Mbps.

What can I apply?

I can also include an SPI camera module for my Raspberry Pi Pico which I think will greatly improve its performance and results since it is meant for real time machine learning + image processing.

"Stationary Robot" device: Amazon Alexa

Amazon Alexa is the software integrated with the much older Amazon Echo hardware, both of which are very high-level technologies. Amazon Alexa is able to synchronise with many different types of devices and draw relevant data from them and, for example, function as a media player, a smart home control hub, make/take phone calls, search the web, work like an alarm clock, record audio etc. etc. It has a very wide range of functionality and compatibility making it very versatile and helpful in many situations however it is not specialized for any particular task unlike my own product.

What can I apply?

It is very convenient as a small "static robot" which I will also be incorporating in my design. Similarly, Amazon Alexa is also making use of LED lights to signal when it is on/off/syncing etc. which I think could also be a helpful addition to my product since this would it much more accessible to the user to interface and understand the hardware of the product.

3c Market Research and Analysis of Gesture Recognition Systems

Many modern systems have integrated machine learning into their products from auto-completion in search bars to full customer service chat-bots. All these systems follow the same method of producing results:

1) Pre-defined framework with which to store, manipulate and analyse data for specific insight, this is often designed using neural networks and therefore has high resource requirements
2) Training environment in which either:
    a. The input and output data is given, and a backend algorithm is formulated to reach this result (supervised)
    b. The input data is given, and a backend algorithm is formulated to reach it's own conclusions, which may be adjusted after human analysis/ response (unsupervised)
3) Testing environment in which the input data is given, and output data is compared to expected output data to determine the system's reliability and accuracy

I do not have access to high-level CPU; however I also do not want to rely on cloud services to make use of ML capable hardware. This is because my project will be reliant on I/O via peripherals and this has inherent delay and when web-scraping/ connecting to Google API, therefore I do not want had a third latency layer through cloud computing.

Due to limited resources, I will develop my own "machine learning" system which incorporates these plans using Google MediaPipe library which is a base ML system which picks up the landmarks of particular human features. I will use their Hand library to achieve 21 hand landmarks which I can then analyse. It is very well documented and kept up to date, so I feel this is a reliable dependency.

Although MediaPipe returns 3D co-ordinates, I will only use x and y (2D) co-ordinates in order to save storage space and simplify the computation required as I feel it will be unnecessary to calculate depth from the camera because:

a) Different peoples' hands may demonstrate the gesture in different ways, and I do not want gesture identification to become dependent on that factor
b) Since the different gestures can be identified naturally from silhouettes (2D) images so I know this is a reasonable compromise

My initial aim with this system is to provide input frames (still images from video feed) and use the landmark co-ordinates to distinguish spread of the hand shape using relative distances. This will allow for variation such as hand-size, distance/ angle from camera etc. making it more adaptable.

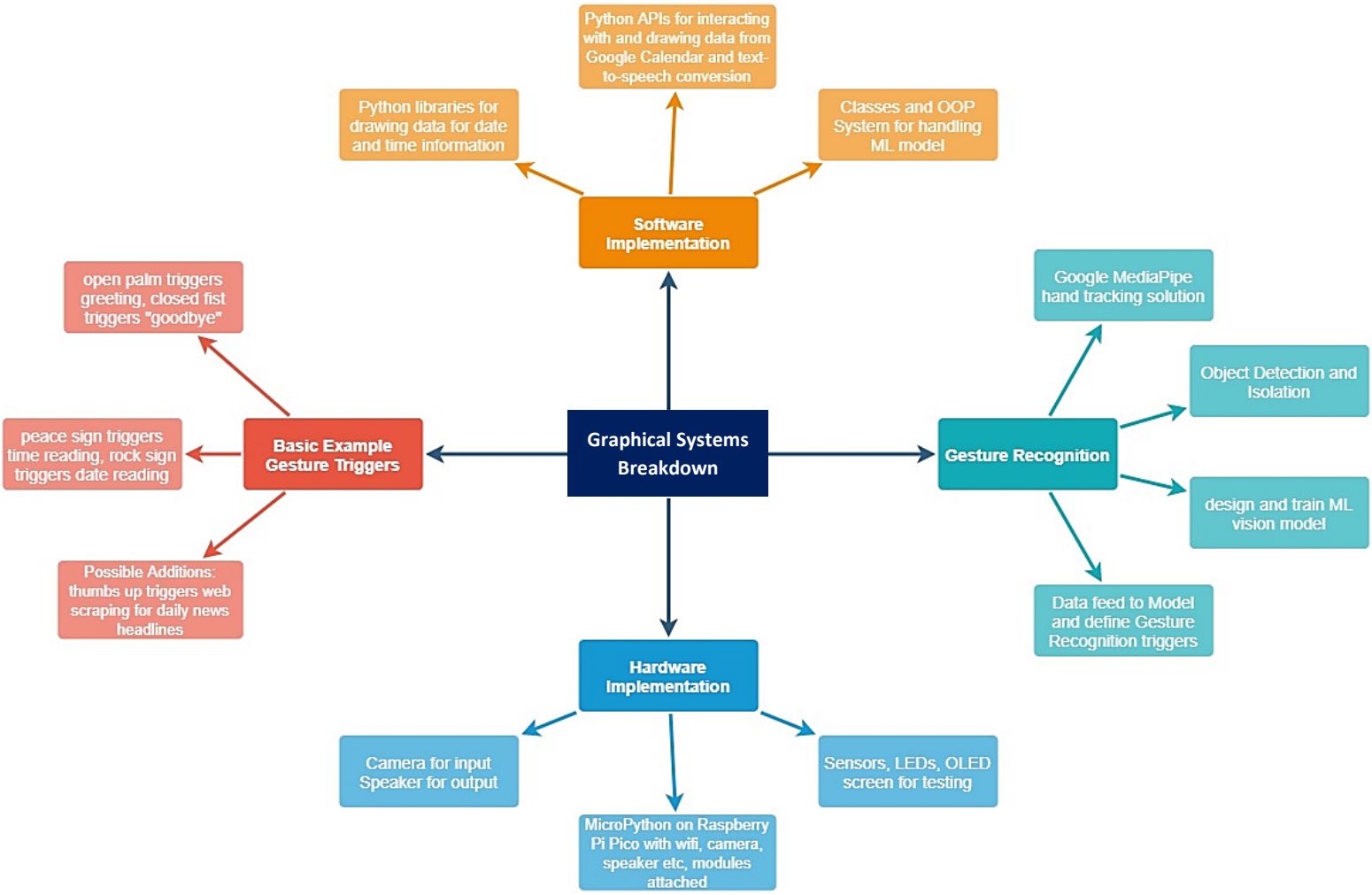
Also, I will make sure to incorporate error value as this is a main feature in machine learning systems I have come across in my research. This will allow offset in the recognition so that there can be some limited variation in how different people perform the different gestures. The gestures I will be using are socially accepted and widely known so that they are not difficult to learn (ie. not sign language specific): open palm, closed fist, rock star, peace sign, Spock greeting (star trek) and thumbs up.

3d Tabular Components Breakdown

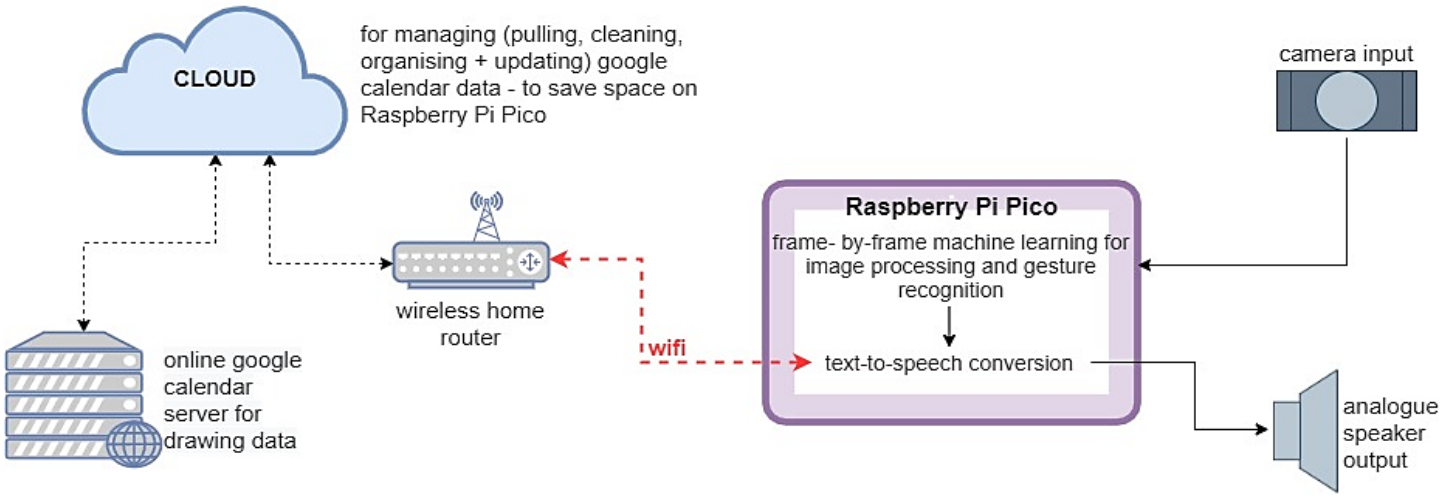
Inputs	Using Raspberry Pi Pico with camera module for image input Possibly using relay switch or other sensor of on/off functionality (see Possible Features below)
Python APIs and libraries	API for python hand tracking – Google's Mediapipe solutions are a well-trained and trusted implementation of machine vision and hand identification however it has no specific gesture recognition API for retrieving data through google calendar, the most widely accessible and used calendar system currently for all ages and uses In-built python libraries for retrieving data such as date/time information before manipulating data so it is suitable for output Python API for text-to-speech (analogue) converter for the output to the user
Image processing + ML model	I will be using hand-tracking from google API however I will be setting up and training my own machine model to use this data feed for custom real-time gesture recognition
IOT implementation	Using Wi-Fi module on Raspberry Pi Pico to connect to my home router to access the internet to retrieve data with google calendar API and for any addition features (see Possible Features below)
Outputs	Speaker for analogue speech output, including greetings, reading out the time, the date, any pre-set gesture outputs as well as reminders and event notifications from the calendar Will also make use of a small OLED screen and LED lights for testing purposes during the hardware set up and tests
Limitations	Restricted camera angle – may need to invest in wider camera angle to ease user experience and increase likelihood of successful image processing/gesture recognition Stand-alone IOT device will be heavily reliant on battery source and other energy source implementation would requires extensive hardware changes/ additions Raspberry Pi Pico is a simple microcontroller with hardware facilities but machine learning model might cause overheating by processing weight on CPU from high/constant resource requirements so I may have to make use of cloud-delivery for the ML vision model Also if the user changes and/or adds new reminders/events in their calendar then the program needs to recognise this and be able to add this to the metaphorical queue of outputs and not be missed out

Possible Features	If "always-on" system is used then this will be constantly processing frame passively until a gesture is recognized If "always-on" system is not used then either relay button could be used to act as an on/off switch, or a motion/light/heat sensor to automate on/off switching – so it is the only "always-on" component Since stand-alone IOT devices need to be energy efficient as well as capable, making use of solar panel energy input could be beneficial To broaden features of gesture recognition, I could include web scraping for news to provide user with daily headlines since important part of productivity is awareness with current events

3e Graphical Systems Breakdown



3f Initial Data Flow Diagram

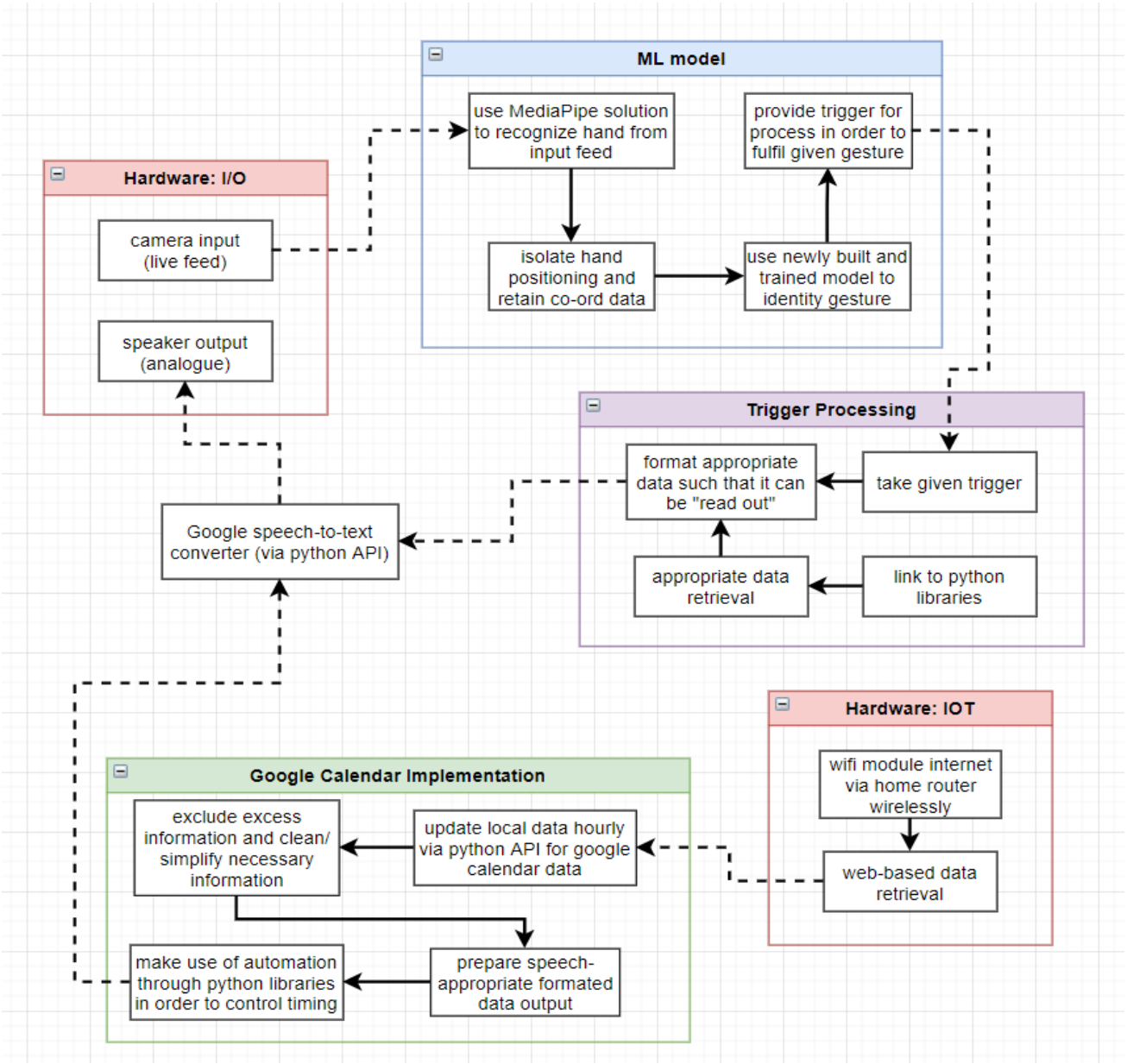


3g End User Requirements/ Success Criteria

No.	Criteria
	Google API side
1	Google calendar API implementation is able to retrieve all of the correct and up-to-date data
2	Data is updated periodically per unit time as decided eg. per hour
3	Database is used to store/ manage/ order the event data to ease further processing
4	Information is retrieved at the accurate output time and text-to-speech functionality works
	Gesture Recognition side
5	cv2 and mediapipe library function together to process each frame for landmark positions
6	Gesture matrices are accurately built from test data and stored in accessible location
7	Error calculation method can reliably determine, and output, gesture received
	Overall System
8	Raspberry Pi Pico is able to function and both camera and speaker module run well
9	Text-to-speech and audio output resource is shared effectively between the two sides

4 Design Stage

4a Structural "Activity" Diagram – to be interpreted as graphical data + algorithms flowchart



4b Decomposition of Files + Data Structures + Algorithms + APIs:

Google Calendar Implementation	
GC access and data retrieval API	– algorithm to execute this will include for regular "updates" (ie. overwriting saved data with newly retrieved data on set interval such as every hour) – will require consideration of current time so as not to load past data/ notifications which are not relevant
Data cleaning	– implemented as function for every piece of data retrieval

	– will follow set rules to reconfigure given data into more "user friendly phrases" which sounds more natural in text-to-speech
Data storage and recall	– since this data is inherently in a given order with set times for output, complex data structure of queue can be used alongside python time module to control output at relevant times – this will allow for automatic time zone synchronization since python time module draws time data from the local machine OS – python “queue” can be better implemented using OOP design which will minimize confusion in the layout and access of the “queue” implementation – the data can be stored using python pickle module as a separate binary file which is more secure than a direct database
ML Model	
MediaPipe solutions framework (Google)	– live, input feed would provide full view of user/user’s hand which will be filtered to simplify data by locating and isolating hand position within the image using OpenCV to interface with real-time input – benefits of MediaPipe solutions is that it has world-wide wealth of practice data and can work effectively in any environment, background, colour/ contrast, or lighting/ filter – so very dependable – the only saved data is the co-ordinates of the hand contour points which are recognized by the MediaPipe solutions and can be used to draw onto the live image (beneficial for testing)
Personalized ML model	– implemented as secondary layer of the hand gesture recognition system in order to recognize the specific hand-gesture used in real-time – will require a lot of testing and training data which I will provide for myself, will make use of tinyML concepts since the Pico is a micro-device – also responsible for triggering the appropriate response to carry out the intended request of the user
Trigger Processing	
Python library links	– retrieval of appropriate data through stored data from .db files such as if the user makes an open palm this will be recognised as a call for greeting and one from a set group of greeting phrases will be used – alternatively, retrieval of appropriate data may come from specific use of python libraries such as if the user makes a rock music sign this will be recognised as a call for date reading and is found via the "date" module
Reformatting data	– because all data output to user is through analogue speakers it must be in such a format that is more natural when "speaking" as opposed to written to be read by the user as most devices work currently

	– reformatting will require specialized algorithm to work with a given expected input format and always return a set output format; regularity is most practical to code and most reliable method of implementing an automated algorithm
Output	
Text-to-Speech – Google API	– every output of the full program will be required to pass through this so I could implement this as a standalone algorithm which is available to access from any part of the program – all data at this point will be strings therefore individual reformatting is required to ensure a smooth transition from digital to analogue data

4c Restatement of User Req. as Design + Testing Objectives

No.	Criteria	How it will be tested
	Google API side	
1	Google calendar API implementation is able to retrieve all of the correct and up-to-date data	View web interface and script output side-by-side
2	Data is updated periodically per unit time as decided eg. per hour	Demonstrate mechanism within code and if possible show periodic database entries
3	Database is used to store/ manage/ order the event data to ease further processing	Explained through written code and screenshots how the db is implemented
4	Information is retrieved at the accurate output time and text-to-speech functionality works	Video of pop-up output at specific expected time to show evidence
	Gesture Recognition side	
5	cv2 and mediapipe library function together to process each frame for landmark positions	Screenshots to demonstrate MP output on live camera feed
6	Gesture matrices are accurately built from test data and stored in accessible location	Screenshot filing and database system used
7	Error calculation method can reliably determine, and output, gesture received	Screenshots of input images, then expected and actual response comparison
	Overall System	
8	GUI is sufficient to demonstrate the functionality (not a direct objective)	Screenshot of working desktop UI running full main program
9	Text-to-speech and audio output resource is shared effectively between the two sides	Main program explained via code and then possibly further video evidence

Due to sourcing issues with Raspberry Pi Pico and greater importance on efficient and reliable backend software, I have decided to follow through with a Proof-of-Concept Method and leave out the Raspberry Pi Pico. Instead I have decided to demonstrate functionality and user access with desktop UI, I will explain my decision further later on in the development process (*5c*).

4d Test Strategy Planning

Testing During Development

Criteria have been segmented (see previous page) with general testing requirements in order to validate the success of that particular task within the full program while carrying out iterative development – ie. combining segmented modules/ subroutines to build up a prototype on which testing and improvements can be made – and validating at every intermediate stage.

Large proportion of input data in this program will be graphical vector data (images), so it is difficult to demonstrate trace tables for these processes. This means a higher testing workload during development and more subjective/ complex testing requirements, however, resulting in majority of changes only being small edits due to the nature of the program as a whole. So a more in-depth testing plan is in place for post-development (see below) to ensure the quality of the final product.

Testing Post-Development

1) **Performance Testing:**

   A) Test that Google Calendar implementation works by setting up series of fake tasks/ reminders at set times and testing:
   a. the accuracy and delay in the data being output via speaker
   b. the clarity after cleaning and reformatting into strings
   c. the effectiveness of the output as response to the user's expectations

   B) Test the ML model system and surrounding algorithms by inputting real-time gesture data via camera module and testing:
   d. the reliability of the camera input and the intermediary stages of data cleaning and preparation for recognition
   e. the accuracy in recognition of the gesture, and in turn triggering the correct output
   f. the effectiveness of the output as response to the user's input

2) **Acceptance Testing:**
   Final Questionnaire for all users:

   1. Do you feel that this product has benefitted you beyond the other time management tools/ methods you use? And how/ why?
   2. Do you feel this product has benefitted you in conjunction with other time management tools/ methods you use? And how/ why?
   3. Can you specify some positive features of the product? Expand on your experiences:
   4. Can you specify some negative features of the product? Expand on your experiences:
   5. In what situation would you consider this a useful/ helpful product to have? (eg. a type of person/ environment/ job etc.)
   6. What do you feel is something that is missing from this product and/ or something that shouldn't be a part of this product?

   Justification:
   Q1-2: to better understand the user's impression of the product their overall experience
   Q3-4: to gain deeper insight into the user's experience with using the product
   Q5: to collect information on how to take the product forward as in real life projects
   Q6: to get general feedback on the product's range of features and uses for the specific user

5 Development Stage

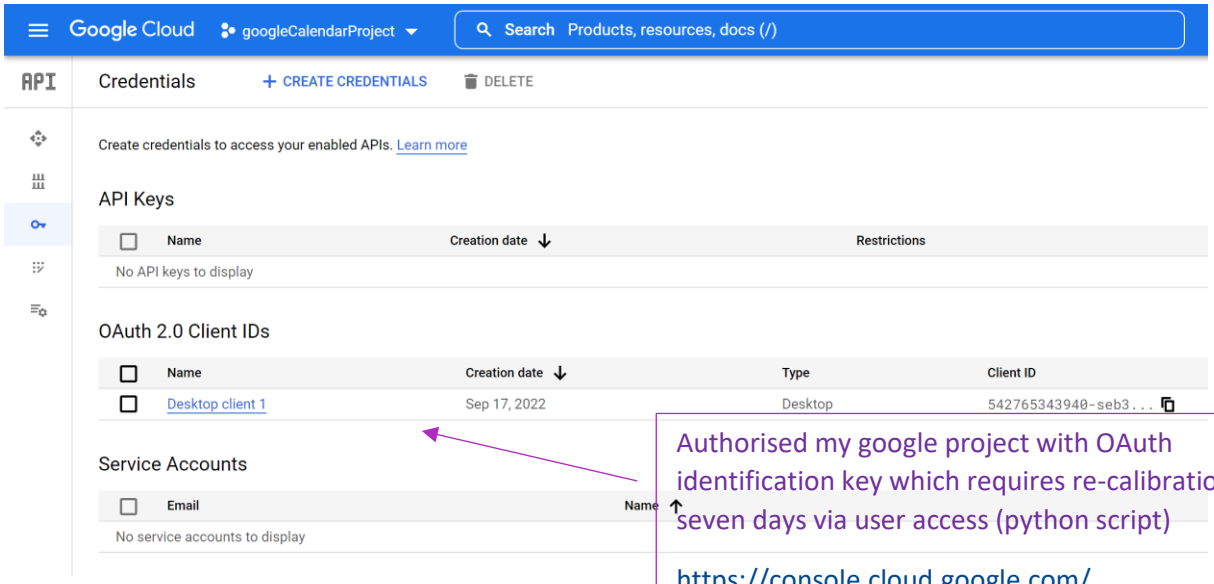
Any success criteria references in this section are links to the final testing where they are assessed.

In order to show different sides of development I will describe the stages for google calendar API integration and I will analyse the (somewhat) final product for gesture recognition system.

5a Development stages for google calendar API

Initial Stage 0

Initialised a google cloud project via google development platform with calendar API access https://developers.google.com/



Authorised my google project with OAuth identification key which requires re-calibration every seven days via user access (python script)

https://console.cloud.google.com/

gCalSetUp.py

```python
from google_auth_oauthlib.flow import InstalledAppFlow
import pickle

scopes = ["https://www.googleapis.com/auth/calendar"]

flow = InstalledAppFlow.from_client_secrets_file("client_secret.json", scopes=scopes)
credentials = flow.run_console()

pickle.dump(credentials, open("token.pkl", "wb"))
print(dir(flow))

```

→ takes “client ID” access key from above from json file and runs google authentication library

→ flow.run_console() is able to return webpage link in order to “re-calibrate” authentication for the given user from which a key is copied back and stored in a binary (pkl) file called “token”

gcalAPI.py

```python
### CAL API SERVICE INITIATION
import pickle
from apiclient.discovery import build

class gAPI():

    def __init__(self):
        #gcal api access
        self.cred = pickle.load(open("token.pkl", "rb"))
        self.service = build("calendar", "v3", credentials=self.cred)

    ### RETRIEVE CALENDAR DATA
    def returnData(self):
        calendarsInfo = self.service.calendarList().list().execute()
        calendarIds = [calendarsInfo["items"][i]["id"] for i in range(len(calendarsInfo["items"]))]
```

→ starting lines of API integration via OOP – you can see the token is used as credentials to build a “service” object to interact with the google calendar API

Review:

At this point I have a working connection to the google calendar API which I will now be able to write functions for collection and cleaning of calendar data.

Development Stage 1

To familiarise myself with API system and python library syntax I experimented with how to make and retrieve events and how event data is organised/ stored/ accessed.
Now saved in reference.py

```python
### GET EVENTS

calendarEvents = service.events().list(calendarId=calendarIds[0]).execute()
print(calendarEvents["items"][0])


### MAKE AN EVENT

testEvent = {
    "summary": "Random Test Event",
    "location": "10 Downing Street, London",
    "description": "test event for a level project",
    "start": {
        "dateTime": "2022-09-30T16:35:00",
        "timeZone": "Europe/London"
    },
    "end": {
        "dateTime": "2022-09-30T17:00:00",
        "timeZone": "Europe/London"
    },
    # "recurrence": []
    "attendees": [],
    "reminders": {
        "useDefault": False,
        "overrides": [
            {
                "method": "email",
                "minutes": 15
            }
        ]

    }
}
service.events().insert(calendarId=calendarIds[0], body=testEvent).execute()

```

Review:
From this experiment I was able to understand how data is handled within the google calendar system, how to reference different event data and how to use the google calendar pre-written API.

Using this I will be able to implement object-oriented programming in order to make functions which are able to interact with the google calendar API and effectively perform web-scraping to handle the event data from my personal google calendar. I will build these functions onto the existing gAPI class structure written in stage 0.

I plan to write two functions:

1) “returnData” which will return raw data via python’s google calendar library as shown above and this will include data on all event types (normal, all-day and reminder type)
2) “cleanData” which will parse through this data and return only the useful information in a simplified, concise tabular format (2d array)

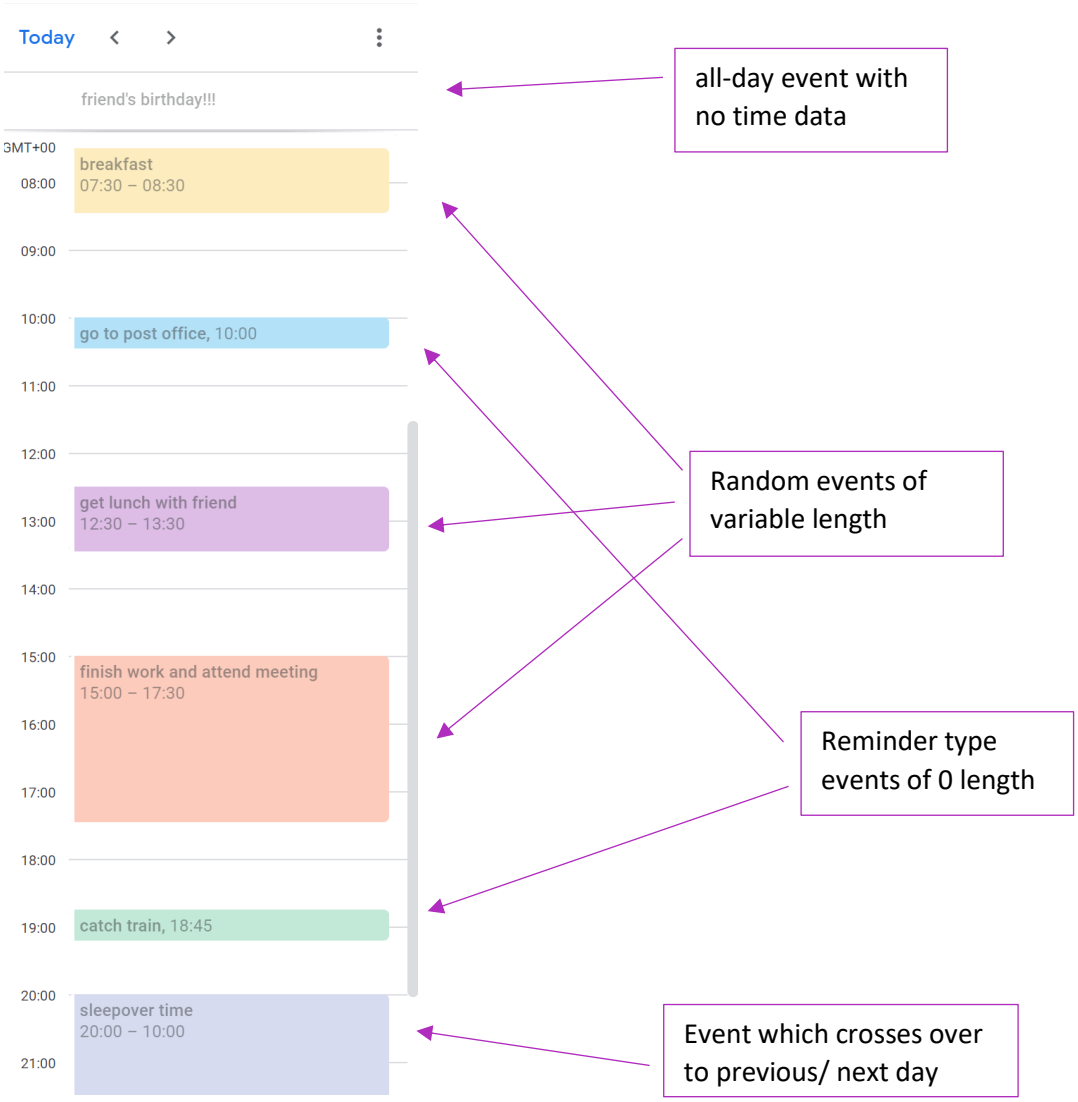
Development Stage 2

```python
### define class for google calendar API integration.
class gAPI():

    ### access credentials and setup service object via constructor.
    def __init__(self):
        self.cred = pickle.load(open("token.pkl", "rb"))
        self.service = build("calendar", "v3", credentials=self.cred)

    ### method for returning raw data from google calendars.
    def returnData(self):
        calendarsInfo = self.service.calendarList().list().execute()
        calendarIds = [calendarsInfo["items"][i]["id"] for i in range(len(calendarsInfo["items"]))]

        today = datetime.date.today()
        tomorrow = today + datetime.timedelta(days=1)
        timeNow = datetime.datetime.now().strftime("%H:%M:%S")

        if str(datetime.datetime.utcnow().astimezone().tzinfo) == "GMT Summer Time":
            offset = "+01:00"
        else:
            offset = "+00:00"

        dailyEvents = self.service.events().list(
            calendarId=calendarIds[0],
            singleEvents=True,
            timeMax=f"{str(tomorrow)}T00:00:00{offset}",
            timeMin=f"{str(today)}T{timeNow}{offset}",
            timeZone="Europe/London",
            orderBy="startTime"
            ).execute()

        dailyEventsInfo = [(dailyEvents["items"][i]["summary"],
                            dailyEvents["items"][i]["start"],
                            dailyEvents["items"][i]["end"],
                            dailyEvents["items"][i]["reminders"]) for i in range(len(dailyEvents["items"]))]

        return dailyEventsInfo

```

Testing for this module:

Problem	Change	Explanation/example
Initially, I wanted to read eventData from all the daily events, but it was difficult to design query to only pick up events which start in range	Tested many variations of midnight (00:00:00) and working with datetime module to input the current date (today and tomorrow)	The service object is able to return data stored in google calendars from within a time range where timeMax means latest start time and timeMin means earliest start time
I then decided to only draw event data for upcoming events (start time in future) to reduce processing complexities in later functions	Changed the timeMin to fit the requirement but some trial and error was required to understand the correct format	I made use of python datetime library to provide current date and time details such that only events after the current time are taken from my calendar
In Britain we follow two different time zones at different times of the year, so this affects the current time required for the query as this is independent of the clock time	Needed to automatically handle change in clocks (GMT vs BST) by assigning offset - variable to denote change in current time	Offset variable was required to correct time range in the service query so utilised datetime module again to get current data

Testing involves running the program with variety of fake events to judge accuracy of output.
Eg. google calendar test events via google web interface:



Time of test: 15:30

```
[("friend's birthday!!!", {'date': '2023-02-13'}, {'date': '2023-02-14'}, {'useDefault': False}),
 ('finish work and attend meeting', {'dateTime': '2023-02-13T15:00:00Z', 'timeZone': 'Europe/
London'}, {'dateTime': '2023-02-13T17:30:00Z', 'timeZone': 'Europe/London'}, {'useDefault':
True}), ('catch train', {'dateTime': '2023-02-13T18:45:00Z', 'timeZone': 'Europe/London'}, {'
dateTime': '2023-02-13T18:45:00Z', 'timeZone': 'Europe/London'}, {'useDefault': True}), ('
sleepover time', {'dateTime': '2023-02-13T20:00:00Z', 'timeZone': 'Europe/London'}, {'
dateTime': '2023-02-14T10:00:00Z', 'timeZone': 'Europe/London'}, {'useDefault': True})]
```

As shown above, returnData function is able to accurately return the information on the events/ reminders provided in the google calendar system by the user (in green) . However one limitation is if the event currently running this will also be picked up (in red) even though it started in the past, this will have to be dealt with in the cleanData function where I will have to make a final check.

```
54      ### parsing through raw data to return ready to process data.
55      def cleanData(self, dailyEventsInfo):
56          timeNow = datetime.datetime.now().strftime("%H:%M:%S")
57          eventData = []
58          for eventInfo in dailyEventsInfo:
59              data = [eventInfo[0]]
60              valid = False
61              if len(eventInfo[1]) == 1:
62                  eventTime = ["None", 0, 0, 0, 1]
63                  valid = True
64              else:
65                  start, end = eventInfo[1]["dateTime"], eventInfo[2]["dateTime"]
66                  sT, eT = start.index("T"), end.index("T")
67                  startTime = [str(i) for i in start[sT+1:sT+6].split(":")]
68                  endTime = [str(i) for i in end[eT+1:eT+6].split(":")]
69
70                  if ":".join(startTime) > timeNow:
71                      valid = True
72                      hrs = int(endTime[0])-int(startTime[0])
73                      mins = int(endTime[1])-int(startTime[1])
74
75                      if mins<0:
76                          mins += 60
77                          hrs -= 1
78                      if hrs<0:
79                          hrs += 24
80
81                      if not eventInfo[3]["useDefault"]:
82                          rem = eventInfo[3]["overrides"][0]["minutes"]
83                      else:
84                          rem = 30
85
86
87                      eventTime = [":".join(startTime), hrs, mins, rem, 0]
88
89              if valid:
90                  data.extend(eventTime)
91                  eventData.insert(0, tuple(data))
92
93          return eventData
94
```

Testing for this module:

Problem	Change	Explanation/example
Variation in event types requires more complex processing	Assigned empty list to “eventData” variable	“all-day” and “normal” events can both be processed in the same function
Variation in formatting for start/ end times, duration, reminder (time before) will be too complex to manage later	Line 64-83 is purely for formatting purposes in order to make data more directly useable	Since the API system uses 24hr notation, more checks are needed to ensure duration and reminder information are accurately represented

```
[('sleepover time', '20:00', 14, 0, 30, 0), ('catch train', '18:45', 0, 0, 30, 0), ("friend's
    birthday!!!", 'None', 0, 0, 0, 1)]
```

Shown above is the given test data which has now been cleaned (+ ordered) to the following format: (“name”, “start time”, duration hrs, duration mins, reminder time in mins, all-day or not)

Review:

Due to how the functions work as part of OOP design it will be more effective to use a database with binary storage (more efficient) to handle this data over time since my project will need to store and reference this data at specific/unplanned times of the day. Therefore, now that data has been organised in tabular format, I will insert the data such that each row represents an event.

THIS IS PROOF OF SUCCESS CRITERIA 1

Development Stage 3

db_init.py

```
### DATABASE INTEGRATION
### --> initialisation of dbs

import sqlite3 as sql

con = sql.connect("cal_data.db")
c = con.cursor()

c.execute("""CREATE TABLE events (
            name TEXT,
            startTime TEXT,
            hrs INTEGER,
            mins INTEGER,
            reminder INTEGER,
            all_day INTEGER
    )""")

con.commit()
con.close()

con = sql.connect("output_files.db")
c = con.cursor()

```

Using SQLite python library I was able to write SQL syntax from within python in order to initialise a flat file database (single table) to store cleaned and organised data.

I used the pre-written methods within the class structure as re-usable code. This is efficient because it will be the same process every time so I can rely on my functions since they have been tested already.

```
    ### collecting eventData from above methods in order to store in "events" database.
    def eventsUpdate(self):
        eventData = self.cleanData(self.returnData())

        conn = sql.connect("cal_data.db")
        cursor = conn.cursor()

        cursor.execute("DELETE FROM events")
        cursor.executemany("INSERT INTO events VALUES (?, ?, ?, ?, ?, ?)", eventData)

        conn.commit()
        conn.close()

    ### pulling eventData from the "events" dababase.
    ### --> ready to process and output.
    def dataRetrieval(self):
        conn = sql.connect("cal_data.db")
        cursor = conn.cursor()

        cursor.execute("""SELECT * FROM events
            WHERE all_day IS TRUE""")
        all_day = cursor.fetchall()

        cursor.execute("""SELECT * FROM events
            WHERE all_day IS FALSE""")
        eventData = cursor.fetchall()

        conn.commit()
        conn.close()

        return all_day, eventData

```

Review:
This was very simple to write, just some application of sqlite3 library and query syntax. Not much testing involved as I was relying on pre-written functions above which I had pretested and defined the database accordingly to match data types etc. etc.
Now the main task is to work on how to make use of these functions in an effective way which allows me to build on the output side of the project as a whole.

Development Stage 4

I plan use google text-to-speech library to form audio files which can be output via any type of speaker; however this has more demanding storage requirements, and I don't want to rely on naming and storing many audio files since they demand more space. Instead I will add to my initialisation program to define a new database only for the output files and I can store my short audio clips as binary file.

```
### DATABASE INTEGRATION
### --> initialisation of dbs

import sqlite3 as sql

con = sql.connect("cal_data.db")
c = con.cursor()

c.execute("""CREATE TABLE events (
            name TEXT,
            startTime TEXT,
            hrs INTEGER,
            mins INTEGER,
            reminder INTEGER,
            all_day INTEGER
    )""")

con.commit()
con.close()

con = sql.connect("output_files.db")
c = con.cursor()

c.execute("""CREATE TABLE files (
            fileName TEXT,
            audioBinary BLOB,
            length REAL,
            reminder TEXT
    )""")


con.commit()
con.close()

```

Review:
Made use of my pre-existing knowledge on sqlite3 database syntax and the module structure of my program. Now all output files (short audio clips of the event message containing google calendars API data) will be read out by Google's open-source library "google text-to-speech" and will be saved in the database as binary file using "BLOB" datatype. It essentially signifies unknown/unspecified datatype, which is a unique feature of python's sqlite3 library, allowing me store audio files as binary data and read/ write directly from within python (which has binary data conversion built in). I can now use this as my goal in the next stage.

Development Stage 5

In this stage the goal is to fulfil the “output_files” database created previously, which contains the file name, audio data in binary format, length, and time to output the given message.

```
130     ### uses above methods to interact with database and prepare data for output.
131     ### --> audio stored as binary data in "output_files" database.
132     def dataProcessing(self):
133
134         all_day, eventData = self.dataRetrieval()
135
136         inputs = []
137         while eventData:
138             output1, output2 = "", ""
139
140             ev = eventData.pop()
141             dur = str(ev[2]) + " hrs and " + str(ev[3]) + " mins"
142             output1 += f"you have {ev[0]} event in {ev[4]} mins at {ev[1]} for {dur},"
143             output2 = f"you have {ev[0]} event starting now at {ev[1]}"
144
145             if all_day:
146                 alldayOut = "\nyour all day events include: "
147                 for a_d in all_day:
148                     alldayOut += f"{a_d[0]} event, "
149                 output1 += alldayOut[:-2]
150
151             file1 = f"gtts_{ev[0].replace(' ', '')}_1.mp3"
152             gTTS(output1, lang="en").save(file1)
153             file2 = f"gtts_{ev[0].replace(' ', '')}_2.mp3"
154             gTTS(output2, lang="en").save(file2)
155
156             with open(file1, "rb") as f:
157                 binary1 = f.read()
158                 length1 = MP3(f).info.length
159                 rem = ((datetime.datetime.strptime(ev[1], "%H:%M")
160                     -datetime.timedelta(minutes=ev[4])).strftime("%H:%M")).replace(":", ".")
161             os.remove(file1)
162
163             with open(file2, "rb") as f:
164                 binary2 = f.read()
165                 length2 = MP3(f).info.length
166                 start = ev[1].replace(":", ".")
167             os.remove(file2)
168
169             db_input = [[file1, binary1, length1, rem],
170                         [file2, binary2, length2, start]]
171
172
173             inputs.extend(db_input)
174
175         conn = sql.connect("output_files.db")
176         cursor = conn.cursor()
177
178         cursor.execute("DELETE FROM files")
179         cursor.executemany("INSERT INTO files VALUES (?, ?, ?, ?)", inputs)
180
181         conn.commit()
182         conn.close()
183
```

Testing for this module:

Problem	Change	Explanation/example
Initially I wrote separate methods for processing and inputting the data into the “output_files” database but this was inefficient and slow	Merged the two methods in order to simplify the code and avoid carrying arrays of data across as arguments	This was still a reliable decision as the code required to complete these jobs isn’t going to be needed anywhere else and there is a lot of data being handled in this algorithm

I needed to plan when to output each message in terms of order without having all the audio files readily loaded	Taking advantage of database structure of adding new records in queue like order (LILO ie. input at the “end” and output from the “top”) to organise and keep track of timing of the events	This simplified the algorithm since the events where already drawn in order from the google calendar API and stored in the same order in “events” database
Came across issue of “deciding” what time to output each event message or if multiple different messages were required	It was most intuitive use of the data to output twice, once at the time of the reminder (reminder time) and once at the time of the event (actual start time)	I found this to be successful strategy as this would benefit both people who like to be prepared in advance and those who need a reminder to get going at the right time since I relate to this issue a lot

Here I have used https://sqliteviewer.app/ to view the contents of my SQLite database files, which shows organised and easily accessible data storage and management.

cal_data.db :

	name	startTime	hrs	mins	reminder	all_day
	Search column...	Search column...	Search column...	Search column...	Search column...	Search column...
1	sleepover time	20:00	14	0	30	0
2	catch train	18:45	0	0	30	0
3	friend's birthday!!!	None	0	0	0	1

output_files.db :

	name	audio	length	reminder
	Search column...	Search column...	Search column...	Search column...
1	gtts_catchtrain_1.m...	51 KB	13.056	18.15
2	gtts_catchtrain_2.m...	18.56 KB	4.752	18.45
3	gtts_sleepovertime_...	50.44 KB	12.912	19.30
4	gtts_sleepovertime_...	18.19 KB	4.656	20.00

It is in the reverse order for cal_data so that it can be entered into output_files in the correct order (somewhat following stack system from cal_data to queue system in output_files)

Review:
Since my method is to store two audio messages for each event, this means there will be double the amount of audio data than expected. Therefore in order to manage the “output_files” database I will design the output algorithm in such a way that it minimises on storage demand and maximises on efficiency and is able to be run stand-alone and handle errors such as an empty database etc. etc.

THIS IS PROOF OF SUCCESS CRITERIA 3

Development Stage 6

```
184
185     def dataOutput(self):
186         conn = sql.connect("output_files.db")
187         cursor = conn.cursor()
188 
189         cursor.execute("""SELECT * FROM files
190             """)
191         fileData = cursor.fetchall()
192         ### [file, binary, length, rem]
193 
194         conn.commit()
195         conn.close()
196 
197         for file in fileData:
198             with open(file[0], "wb") as f:
199                 f.write(file[1])
200 
201             match, valid = False, False
202 
203             while not match:
204                 timeNow = datetime.datetime.now().strftime("%H.%M")
205 
206                 if float(file[3]) >= float(timeNow):
207                     valid = True
208                 else:
209                     match = True
210 
211                 if float(file[3]) <= float(timeNow) and valid:
212                     os.system(f"start {file[0]}")
213                     time.sleep(file[2]+5)
214                     match = True
215 
216             os.remove(file[0])
217 
```

Testing for this module:

Problem	Change	Explanation/example
Although some events are in the future their reminder may be in the past which is impossible to output on time	Had to introduce an extra layer of confirmation to check whether an output message is valid before matching to the current time	If I were trying to compare a timestamp for an output message in the past this would cause an endless loop
The audio for each message would cut off suddenly and Microsoft Music Player (audio app) would close abruptly	Initially spiralled down a research hole to find out how to control external apps before simply attempting to elongate the waiting time till the python script cut off the audio app	I released this was because the mp3 file had finished running so the app closed and python script took over faster than the audio could finish naturally making it sound unnatural

Review:

At the end of this stage I now have a full working and tested prototype of OOP approach to collecting, cleaning, storing, and handling google calendar data via API authorised web access. The next steps include testing for user feedback and further modifications for improvements.

The following video evidence utilises the same example input shown at the start of development phase. Video evidence of Google Calendar Implementation: https://youtu.be/JI1NyUE9ZFc

THIS IS PROOF OF SUCCESS CRITERIA 4

5b Demonstration of Algorithms + Approach used for Gesture Recognition

hand_class.py

```python
### MEDIAPIPE HAND MODEL CLASS

### import libraries.
import os
import cv2
import mediapipe as mp
import time
import uuid
import numpy as np

### define class for mediapipe solution.
class HandModel():

    ### integrate mediapipe solution via constuctor.
    ### --> ie. accesible by all instance-reliant methods.
    def __init__(self):
        self.solutions = mp.solutions
        self.model = self.solutions.hands.Hands(static_image_mode=False, max_num_hands=1)
        self.draw = self.solutions.drawing_utils

    ### method for collecting images for training data
    def collectImages(self, IMAGES_PATH, labels, no_imgs):
        for label in labels:
            cap = cv2.VideoCapture(0)
            print(f"images for {label}")
            time.sleep(5)

            for i in range(no_imgs):
                valid = False

                while not valid:
                    ret, image = cap.read()
                    cv2.imshow(f"{label} {i}", image)

                    if cv2.waitKey(1) == ord("c"):
                        valid = self.validateImage(image)

                    if cv2.waitKey(1) == ord("q"):
                        break

                imgName = os.path.join(IMAGES_PATH, label+"_"+f"{str(uuid.uuid1())}.jpg")
                cv2.imwrite(imgName, image)

                time.sleep(2.5)

                if cv2.waitKey(1) == ord("q"):
                    break

            cap.release()
            cv2.destroyAllWindows()

```

HandModel class used to implement OOP structure

constructor used to access all MediaPipe solutions from every method

"IMAGES_PATH" is pre-set variable which leads to folder containing training images

"labels" is pre-set list of gesture names

every image file is saved with a unique name therefore well-organised

```
51
52      ### method for validating image landmarks are visible.
53      def validateImage(self, image):
54          lms = self.model.process(image).multi_hand_landmarks
55          if lms:
56              return True
57
58          return False
59
60      ### method for providing normalised average gesture matrix from training data.
61      def setGestureMatrices(self, IMAGES_PATH, MATRIX_PATH, labels):
62          for label in labels:
63              for (root, dirs, files) in os.walk(IMAGES_PATH):
64                  if root == f"{IMAGES_PATH}\\{label}":
65                      distMatrix = np.zeros((10, 10))
66                      n=0
67                      for file in files:
68                          n += 1
69
70                          imgPath = os.path.join(root, file)
71                          image = cv2.imread(imgPath)
72
73                          lm_lst = self.landmarks(image)
74                          if lm_lst != []:
75                              imageMatrix = np.array(self.distMatrix(lm_lst, False))
76
77                              distMatrix = np.add(distMatrix, imageMatrix)
78                              distMatrix = self.normaliseMatrix(np.divide(distMatrix, n))
79
80                              matPath = os.path.join(MATRIX_PATH, f"distMatrix_{label}.npy")
81                              np.save(matPath, distMatrix)
82
83      ### method for returning hand landmarks via mediapipe solution.
84      def landmarks(self, frame, returnLms=False):
85          lms = self.model.process(frame).multi_hand_landmarks
86          h, w, c = frame.shape
87          lm_lst = []
88
89          if lms:
90              for lm in lms:
91                  for i, l in enumerate(lm.landmark):
92                      lm_lst.append((int(l.x*w), int(l.y*h)))
93
94          if returnLms:
95              return lms, lm_lst
96          return lm_lst
97
98      ### method for annotating image with landmarks.
99      ### --> testing purposes
100     def annotate(self, frame):
101         lms, lm_lst = self.landmarks(frame, True)
102
103         if lms:
104             for lm in lms:
105                 self.draw.draw_landmarks(frame, lm, self.solutions.hands.HAND_CONNECTIONS,
106                     connection_drawing_spec=self.draw.DrawingSpec((0, 255, 0), 2))
107
108             for i in [4, 8, 12, 16, 20]:
109                 cv2.circle(frame, (lm_lst[i][0], lm_lst[i][1]), 10, (255, 0, 0), cv2.FILLED)
110
111         return frame
112
```

training images are used to cumulatively build a dataset for a particular gesture and then find average which represents a specific signature within each matrix saved for reference later in binary files

these methods were built to improve efficiency, organisation, and usability of the program since they are re-used many times – reduces debugging time

```
112
113     ### method for returning distance between any two landmarks.
114     def distance(self, lm_lst, a, b):
115         import math
116
117         x_diff = lm_lst[a][0] - lm_lst[b][0]
118         y_diff = lm_lst[a][1] - lm_lst[b][1]
119
120         dist = math.sqrt(x_diff**2 + y_diff**2)
121
122         return int(dist)
123
124     ### method for providing distance matrix from landmarks.
125     def distMatrix(self, lm_lst, normalise=True):
126         distMatrix = []
127         keyPoints = [0, 4, 5, 8, 9, 12, 13, 16, 17, 20]
128
129         for n in keyPoints:
130             dists = []
131             for m in keyPoints:
132                 dists.append(self.distance(lm_lst, n, m))
133
134             distMatrix.append(dists)
135
136         if normalise:
137             return self.normaliseMatrix(distMatrix)
138         else:
139             return distMatrix
140
141     ### method for normalising given distance matrix such that distances become ratios.
142     def normaliseMatrix(self, matrix):
143         matrix = np.array(matrix)
144         minval, maxval = matrix.min(), matrix.max()
145
146         normalised = np.around((((matrix-minval)/(maxval-minval))*100), 2)
147
148         return normalised
149
150     ### method for returning sum of errors between any two matrices.
151     def calcError(self, distMatrix, gestMatrix):
152         errorMatrix = np.abs(np.subtract(np.array(distMatrix), np.array(gestMatrix)))
153         error = np.sum(errorMatrix)
154
155         return error
156
157     ### method for pulling gesture matrix data from binary file to numpy array.
158     def fetchGestureMatrix(self, option):
159         MATRIX_PATH = "gestureMatrices"
160
161         matPath = os.path.join(MATRIX_PATH, f"distMatrix_{option}.npy")
162         distMatrix = np.load(matPath)
163
164         return distMatrix
165
```

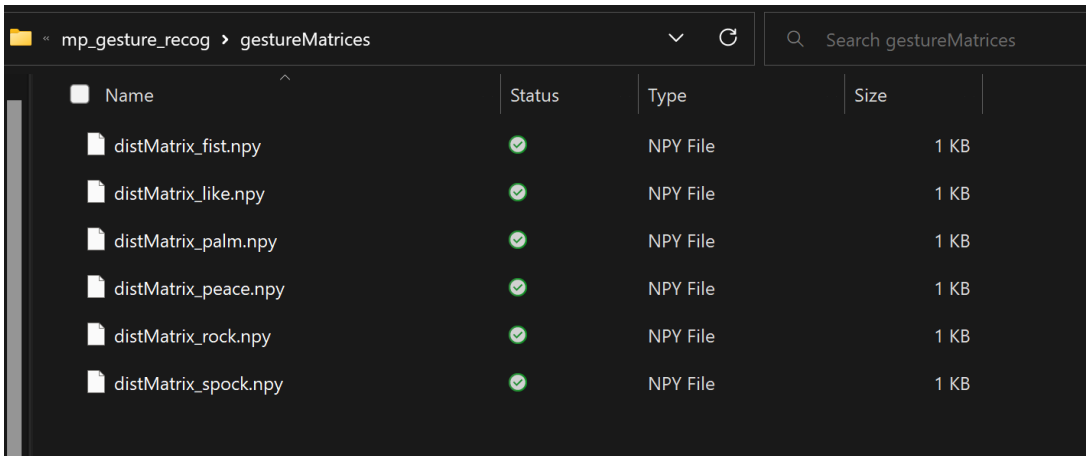
these methods work in tandem to provide matrix of distances between pre-determined “keyPoints” which reduce the dataset to only include meaningful data.

purpose of normaliseMatrix method is to overcome challenge of variable distance from camera/ size of hand problem which is unpredictable factor of the user

after a pre-set average gesture matrix is fetched then an “error” value can be found as the sum of differences between

Evidence of Gesture Matrices and filing approach

I did not use a database as I decided secure binary files to be the most effective method as shown:



Why:

1) Dense, repeated numerical data is most efficiently stored in binary files which can then be accessed as needed and read when called to via numpy function call
2) It is more secure and requires specific commands to be opened/ read from/ written to therefore less risk of accidental editing and/ or deletion

Example file shows normalised landmark distances between each of the chosen key hand landmarks:

Dtype = float64
Shape = (10, 10)

	1	2	3	4	5	6	7	8	9	10
1	0.0	77.55	100.0	73.48	95.2	59.87	92.96	68.61	94.23	78.76
2	77.55	0.0	32.09	9.69	40.46	29.11	60.84	53.08	83.98	77.66
3	100.0	32.09	0.0	33.44	32.1	48.88	62.0	69.83	90.44	92.18
4	73.48	9.69	33.44	0.0	43.68	25.99	62.51	51.77	85.12	76.9
5	95.2	40.46	32.1	43.68	0.0	43.38	29.53	49.72	59.23	67.42
6	59.87	29.11	48.88	25.99	43.38	0.0	49.65	29.32	66.41	54.18
7	92.96	60.84	62.0	62.51	29.53	49.65	0.0	37.0	29.66	44.64
8	68.61	53.08	69.83	51.77	49.72	29.32	37.0	0.0	40.3	25.4
9	94.23	83.98	90.44	85.12	59.23	66.41	29.66	40.3	0.0	31.98
10	78.76	77.66	92.18	76.9	67.42	54.18	44.64	25.4	31.98	0.0

Diagonal of "0"s can be seen which represents minimum distance between the equivalent hand landmarks – in reality this is actually 0 distance.

Additionally, the number 100 can be seen showing maximum distance between two different hand landmarks – in reality this is a small number measured as distance in pixels between image co-ordinates, but it is 100% of the normalised distance so the biggest.

Downfall of this method is repeated data since the file represents a two-way table of values, however, this layout ensures it is easy to refer to each value when calculating error and it is the same for each pre-set gesture matrix.

THIS IS PROOF OF SUCCESS CRITERIA 6 AND 7

```python
    ### method for returning closest match within a custom threshold for any given gesture.
    def deterGesture(self, frame, source):
        lm_lst = self.landmarks(frame)

        options = [
        # label,    error,  threshold,  output
        ["palm",    0,      500,        "hello"],
        ["fist",    0,      2000,       "goodbye"],
        ["rock",    0,      600,        "forex"],
        ["peace",   0,      600,        "weather"],
        ["like",    0,      800,        "headline"],
        ["spock",   0,      400,        "spock"]
        ]

        if lm_lst:
            distMatrix = self.distMatrix(lm_lst)

            for option in options:
                gestMatrix = self.fetchGestureMatrix(option[0], source)
                option[1] = self.calcError(distMatrix, gestMatrix)

            bestMatchOrder = sorted(options, key=lambda op:op[1])

            if bestMatchOrder[0][1] < bestMatchOrder[0][2]:
                return bestMatchOrder[0][0]

            return "unknown"
        return "none"
```

using pre-defined 2d array of “options” to link necessary data, this method brings together the previous methods in order to provide a “bestMatchOrder” of the “options” to the current unknown gesture

setUp.py

```python
### IMAGE COLLECTION FOR TRAINING DATA AND AVERAGE GESTURE MATRICES
### DEFINED FOR REFERENCE

### import libraries.
import os
import numpy as np
import cv2
### import class module from file.
from hand_class import HandModel
mp_model = HandModel()


### image collection.
IMAGES_PATH = "collectedImages"
labels = ["palm", "fist", "rock", "peace", "spock", "like"]
no_imgs = 7

# mp_model.collectImages(IMAGES_PATH, labels, no_imgs)


### gesture matrices defined.
MATRIX_PATH = "gestureMatrices"

# mp_model.setGestureMatrices(IMAGES_PATH, MATRIX_PATH, labels)
```

simplified program draws from “HandModel” class to:

1) collect and organise training images using “collectImages” method
2) iterate over this data to build and normalise pre-defined gestures using “setGestureMatrices” method

main.py

```python
### RECOGNITION OF PRESET GESTURES

### import libraries.
import cv2
### import class module from file.
from hand_class import HandModel
mp_model = HandModel()

### live feed.
cap = cv2.VideoCapture(0)
while True:
    ret, frame = cap.read()

    ### image manipulation.
    frame = cv2.flip(frame, 1)
    h, w, c = frame.shape

    ### recognition from preset gestures.
    gesture = mp_model.deterGesture(frame)

    # colour format = BGR
    cv2.putText(img=frame, text=gesture, org=(50, 50), fontScale=1,
        fontFace=cv2.FONT_HERSHEY_SIMPLEX, color=(255, 0, 0), thickness=2)

    ### feed output and base case.
    cv2.imshow("live feed", frame)
    if cv2.waitKey(1) == ord("q"):
        break

### end live feed.
cap.release()
cv2.destroyAllWindows()
```

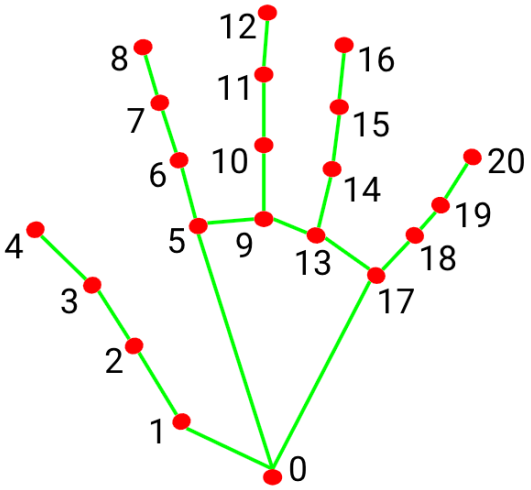
THIS IS PROOF OF SUCCESS CRITERIA 5

main running loop which is used to continuously update and determine which gesture is being performed by the user using “deterGesture” method

written description of recognised gesture is given on the live feed

base case is defined to ensure the camera can be released and loop is finite

Hand Detection System Explained



0. WRIST
1. THUMB_CMC
2. THUMB_MCP
3. THUMB_IP
4. THUMB_TIP
5. INDEX_FINGER_MCP
6. INDEX_FINGER_PIP
7. INDEX_FINGER_DIP
8. INDEX_FINGER_TIP
9. MIDDLE_FINGER_MCP
10. MIDDLE_FINGER_PIP
11. MIDDLE_FINGER_DIP
12. MIDDLE_FINGER_TIP
13. RING_FINGER_MCP
14. RING_FINGER_PIP
15. RING_FINGER_DIP
16. RING_FINGER_TIP
17. PINKY_MCP
18. PINKY_PIP
19. PINKY_DIP
20. PINKY_TIP

I have designed my algorithm to be dependent on spread of the hand shape, not on distance or hand size, angle, depth from the camera, or left/right hand detection - making it much more robust when testing.

Breakdown of Testing Process

Training: involved working on improving prototype for variation of pre-determined gestures in different environments.

To ensure my training data was not affected by external/ irrelevant factors, I introduced as much variation as possible while keeping strictly within the acceptable range for the recognition algorithm.

In general, I have written a trace table to show the variation and resulting capability of my program:

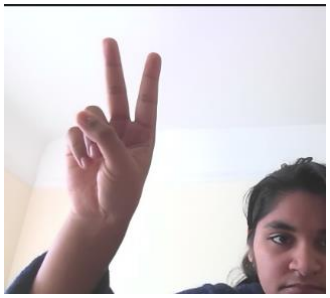
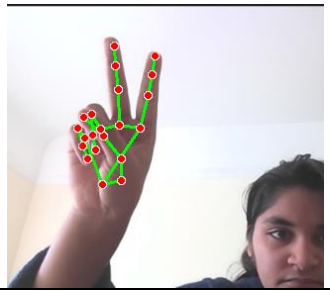
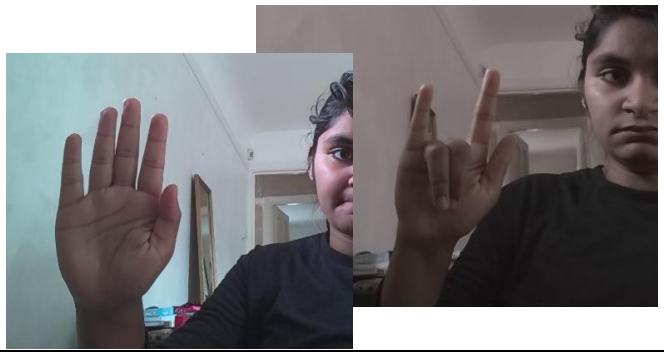
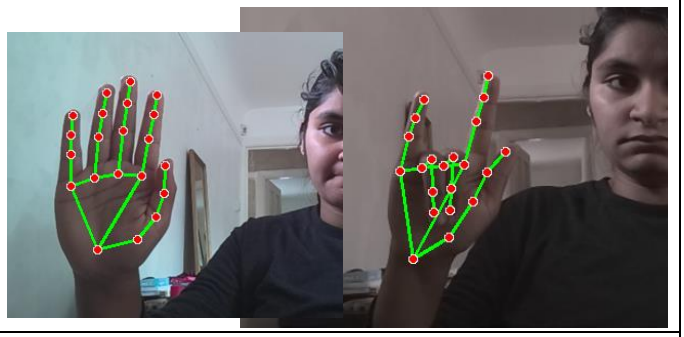
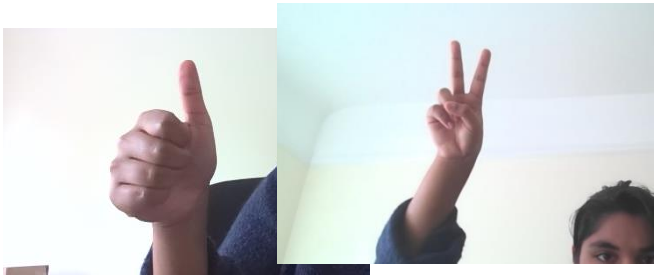
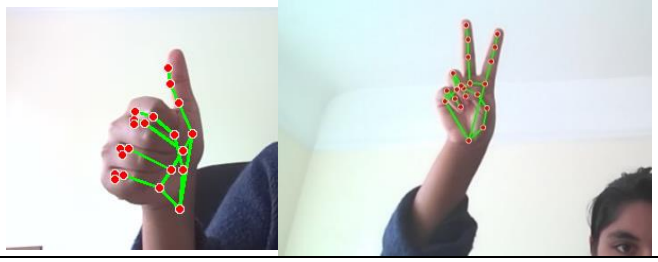
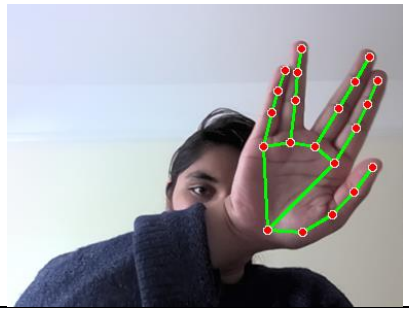
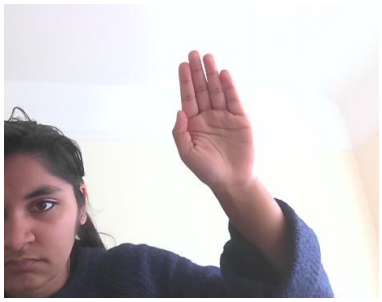
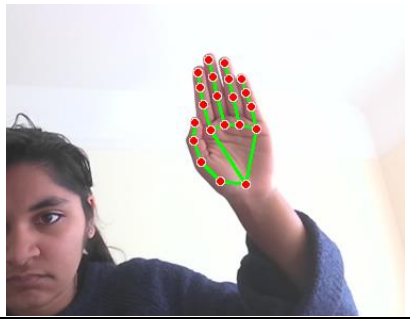
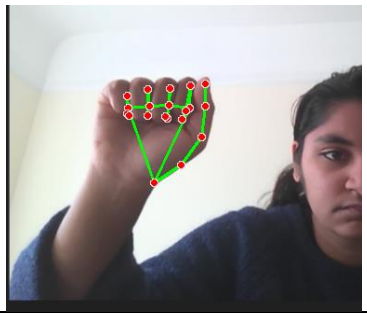
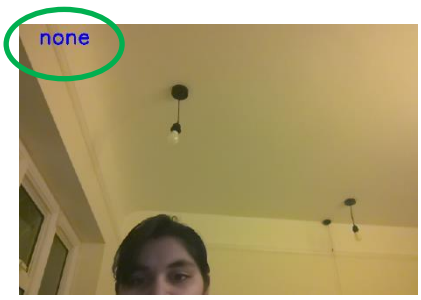
Type of variation	**Successful outcome?**
Clear and bright (normal) 	✓ 
Dull lighting/ colour-tinted 	✓ 
Far/ close up 	✓ 
angle/ tilt of hand 	✓ 

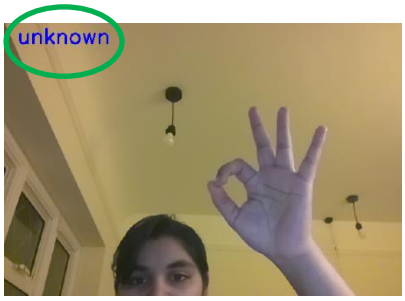
Image clarity (ie. introducing movement) 	✓ 
Right hand 	✓ 
Left hand 	✓ 

This is effectively an all-round representation of my training data

Testing: as shown in real time, can be seen in the following video evidence:
https://youtu.be/RZT1R_ElcfM

In particular:





First image shows algorithms response to lack of any hand landmarks detected at all whereas second image shows response to an error in recognition or insufficient similarity to pre-written gestures → responsive error handling

5c Short Mid-development Questionnaire

My original plans were hindered by my lack of resources and need to focus on meeting the success criteria as restated during test planning. Therefore I decided to consult my stakeholders for advice on following Proof-of-Concept Method via desktop UI.

1) What general features are needed to demonstrate proof-of-concept?
2) What extra/ supporting UI features would be beneficial to include?
3) What additional design considerations should I keep in mind when creating the UI?

I used these questions to understand, as potential users, what they would want to see in, effectively a prototype, of the project and how proof-of-concept method would play out.

Responses

Feedback in terms of proof-of-concept:

Person A was very directed towards simplicity, making sure the original I/O systems were demonstrated separately but ran concurrently and efficiently as well. I plan to use threading in order to achieve this, and resource locking (similar to record locking in databases) in order to manage the shared I/O resource (audio output).

Person B mentioned using performance modelling alongside a Raspberry Pi Pico sim to strengthen my proof-of-concept argument. This can be implemented as an additional testing phase and further expanded on for future development in evaluation section.

Feedback in terms of UI development:

Person A said that a limitation of the original design was no user directions/ guide (although this could have been implemented physically when designing the final end product) so I should use this opportunity to provide the user with a basic guide on inputting data (ie. hand gestures). They also mentioned real-time user feedback as a way to making the system more user-friendly.

Person B was eager to see calendar details in some way since this would make a productive and relevant use of space on the screen and add baseline functionality (even though this is not the final product).

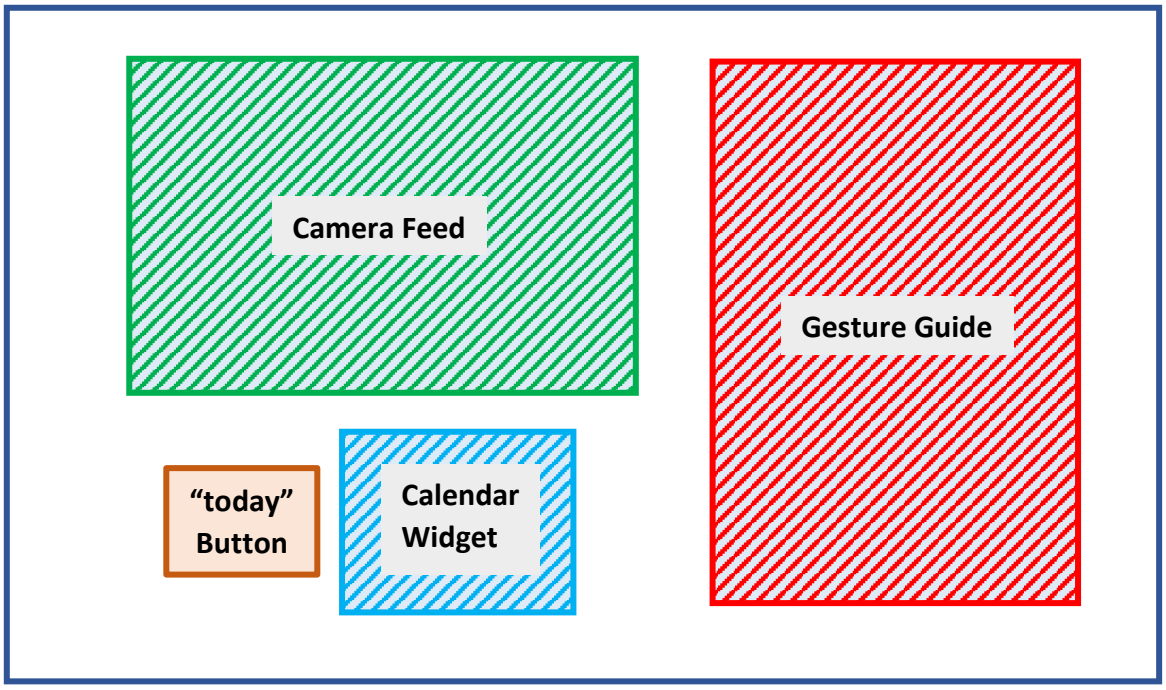
In summary, I will be using tkinter to build a simple graphical interface, this is another well documented and maintained library which contains many specialised sub-libraries which I can use to achieve specific outputs/ GUI elements.

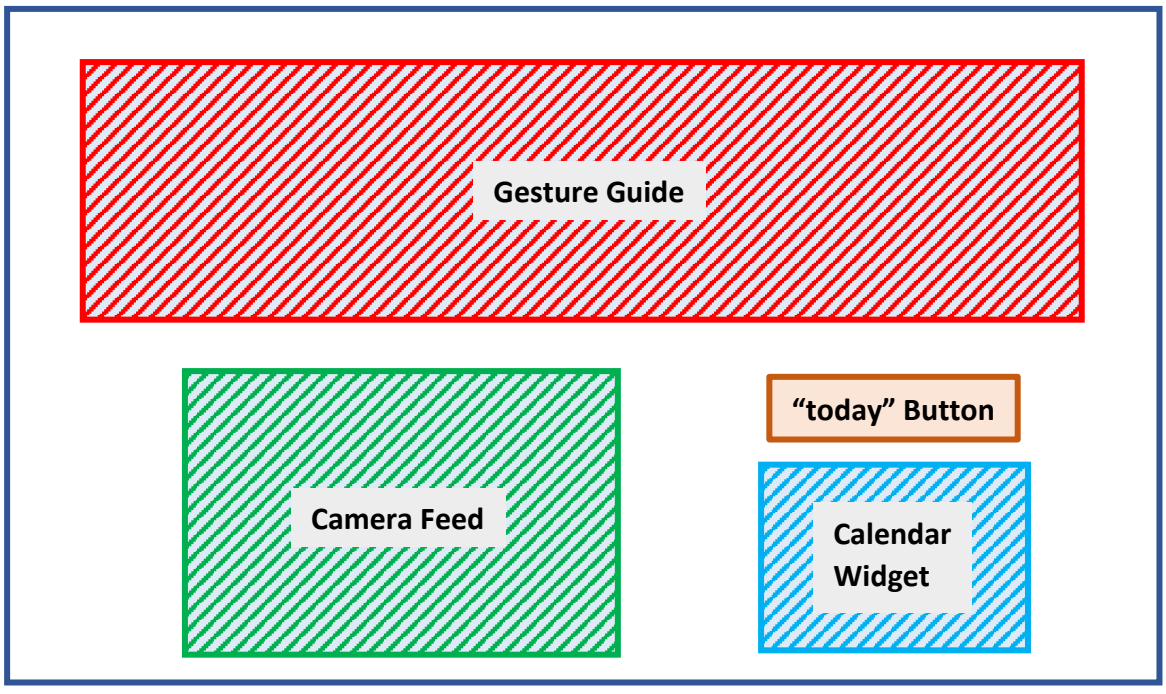
Summary

Therefore my overall GUI will incorporate:

1) A camera feed to make it obvious the gesture recognition system is running
    a. This will pause when a gesture is recognised, and the speaker is outputting audio
    b. In attempt to make it more user-friendly as suggested I can also output on the screen if unknown or unclear hand gesture is being provided to prompt the user to adjust their particular input
2) A gesture guide to show the accepted hand gestures and their output
    a. This should help the user better understand the functionality of the product and being able to make full use of the system
3) A calendar widget which is simply a pre-designed widget within tkinter
    a. I will also program a button to “jump back to today” to make it more intuitive

5d Planning GUI Layout





Final layout:

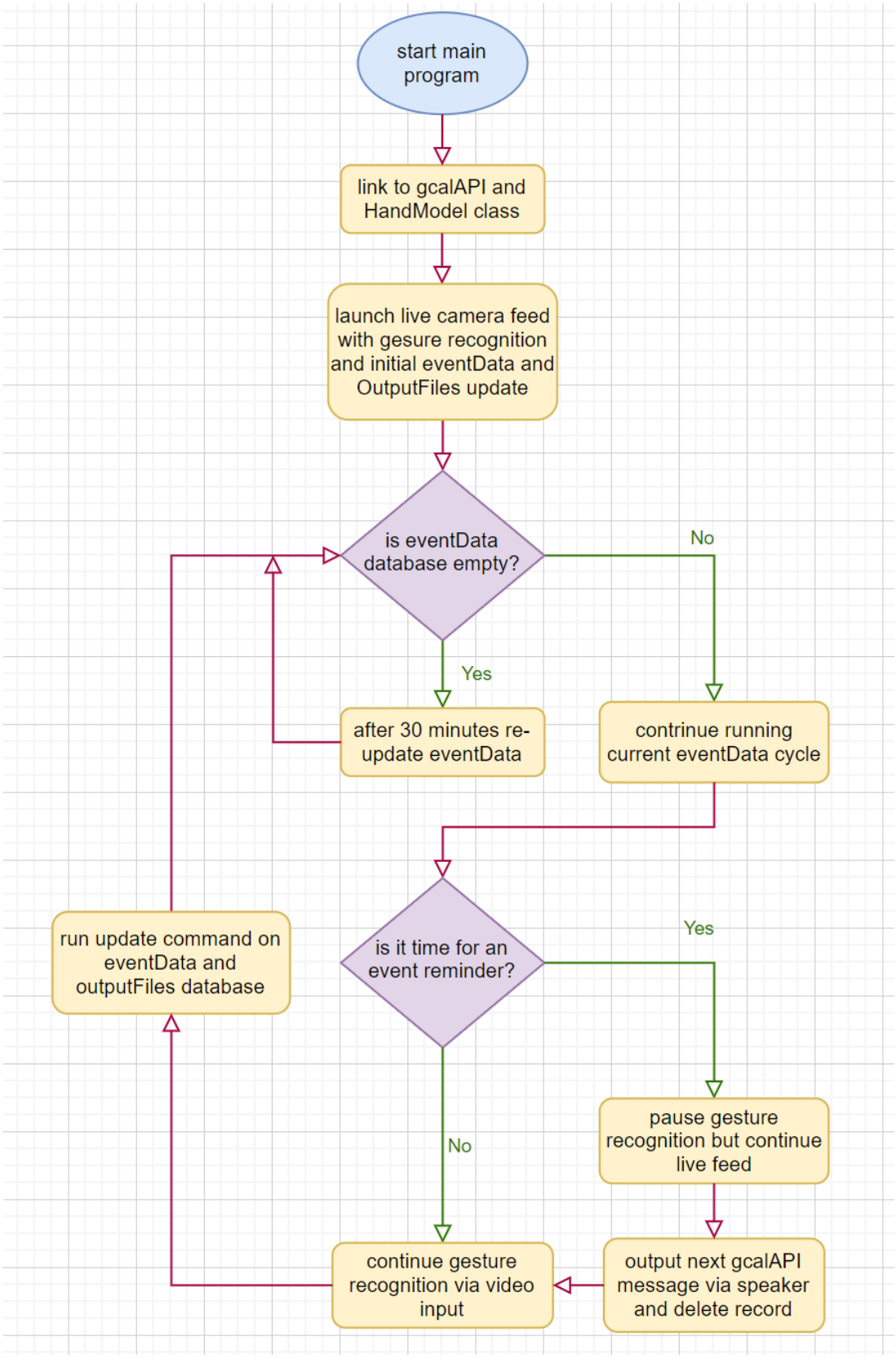
1) I will put camera feed in the top left as this is where people look first and will bring the most focus to this element (as shown in layout 1)
2) I will use horizontal gesture guide and vertical calendar widget + button as these seem most balanced on the page and can easily form a segmentation of the page which seems more organised to the eye (as shown in layout 2)

5e Development of Main Program

Initial Stage 0:

1) Combine main programs of gesture recognition and google calendar API systems
2) Make use of threading in order to run both constantly and use I/O locking to manage shared resource via threading events (inherently global but private variables)
3) Set up web-scraping/ python library calls to return information for gesture triggered outputs

My initial plan:



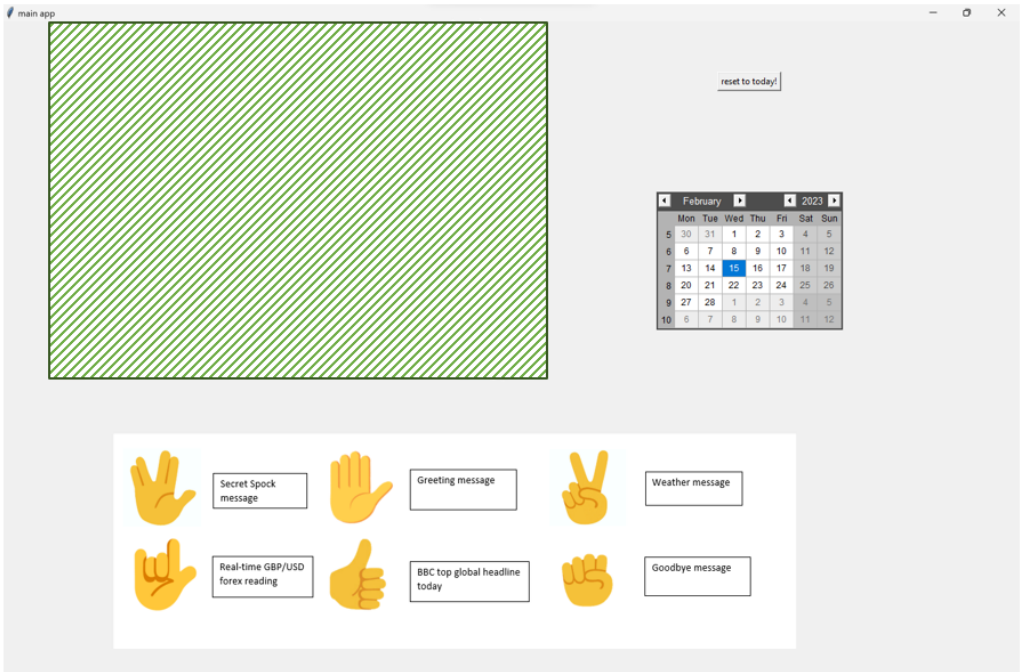
Development Stage 1

```python
### MAIN PROGRAM

### import libraries.
from tkinter import *
from tkcalendar import *
import datetime
from PIL import ImageTk, Image
import numpy as np
import threading

import cv2
from mp_gesture_recog.hand_class import HandModel

from gcal_and_db.gcalAPI import gAPI
import os, sys
import sqlite3 as sql
import time

### setup global instances of pre-written classes.
global mp_model
mp_model = HandModel(source=1)
global obj
obj = gAPI(source=1)

### setup simple interface ising tkinter library.
rawDate = datetime.date.today()
date = [int(i) for i in str(datetime.date.today()).split("-")]

root = Tk()
root.title("main app")
root.geometry(f"{root.winfo_screenwidth()}x{root.winfo_screenheight()}")

def resetDate():
    cal.selection_set(rawDate)

feedL = Label(root)
feedL.grid(row=0, column=0, columnspan=2, rowspan=2, padx=75)

guideI = ImageTk.PhotoImage(Image.open("gesture_guide.png"))
guideL = Label(root, image=guideI)
guideL.grid(row=2, column=0, columnspan=3, padx=60, pady=70)

todayB = Button(root, text=f"reset to today!", command=resetDate)
todayB.grid(row=0, column=2)

cal = Calendar(root, selectmode="day", year=date[0], month=date[1], day=date[2])
cal.grid(row=1, column=2, padx=85)

```

Starting off with all the necessary imports required and initial tkinter setup for a simple GUI.

“gesture_guide.png” is a word document screenshot of the gestures and pre-assigned outputs.

The exact design, layout and sizes will be changed this is just an initial set up.



Development Stage 2

I then implemented the threading system and moved all of these “main” parts of the main program into its own section:

Local instances of the prewritten classes are initiated here to reduce arguments required when calling functions/ defining threads

```
### main program which contains classes, GUI and threading setup.
if __name__ == "__main__":

    ### setup global instances of pre-written classes.
    global mp_model
    mp_model = HandModel(source=1)
    global obj
    obj = gAPI(source=1)

    ### setup simple interface ising tkinter library.
    rawDate = datetime.date.today()
    date = [int(i) for i in str(datetime.date.today()).split("-")]

    root = Tk()
    root.title("main app")
    root.geometry(f"{root.winfo_screenwidth()}x{root.winfo_screenheight()}")

    def resetDate():
        cal.selection_set(rawDate)

    feedL = Label(root)
    feedL.grid(row=0, column=0, columnspan=2, rowspan=2, padx=75)

    guideI = ImageTk.PhotoImage(Image.open("gesture_guide.png"))
    guideL = Label(root, image=guideI)
    guideL.grid(row=2, column=0, columnspan=3, padx=60, pady=70)

    todayB = Button(root, text=f"reset to today!", command=resetDate)
    todayB.grid(row=0, column=2)

    cal = Calendar(root, selectmode="day", year=date[0], month=date[1], day=date[2])
    cal.grid(row=1, column=2, padx=85)

    ### setup threading for both sides and audio event for resource locking.
    audio = threading.Event() #(set=lock, clear=unlock)

    feed = threading.Thread(target=start_feed)
    feed.start()

    gcal = threading.Thread(target=start_gcal)
    gcal.start()

    ### make sure when closed via command or button all necessary duties are finished
    root.protocol("WM_DELETE_WINDOW", close)
    root.mainloop()
```

This is used to set a default calendar date and be able to reset it when the user chooses to (via button)

“audio” is set up as a threading event which is a subclass in the threading library

Development Stage 3

When implementing the two systems as functions, I copied the pre-written programs and added conditions to check if audio event was set (ie. locked) or not before outputting audio.

```
### function for updating calendar event output files.
def update():
    obj.eventsUpdate()
    obj.dataProcessing()

    conn = sql.connect(os.path.join("gcal_and_db", "output_files.db"))
    cursor = conn.cursor()

    cursor.execute("""SELECT * FROM files
        """)
    fileData = cursor.fetchall()
    ### [file, binary, length, rem]
    conn.commit()
    conn.close()

    return fileData
```

```
### function which acts as a thread for google calendar API manipulation.
def start_gcal():
    global fileData
    fileData = update()

    for file in fileData:
        with open(file[0], "wb") as f:
            f.write(file[1])

        match, valid = False, False

        while not match:
            timeNow = datetime.datetime.now().strftime("%H.%M")

            if float(file[3]) >= float(timeNow):
                valid = True
            else:
                match = True

            if float(file[3]) <= float(timeNow) and valid:

                while audio.is_set():
                    time.sleep(3)

                audio_out(file[0], file[2])
                match = True

        os.remove(file[0])
```

Since every upcoming event creates both a reminder and "now" message, a final check is needed to confirm if it is in the future – for example an event in ten minutes (valid) could have a reminder 30 minutes before hand (invalid) so all messages must be checked and deletes once dealt with in some way

Review:

By this point, I have successfully built a GUI interface capable of pulling and organising event data from google calendars for text-to-speech output. Further development will focus on setting up feed with live gesture recognition as well as implementing shared audio resource between them.

Development Stage 4

```
### function for locking and locking speaker I/O resource and running mp3 files.
def audio_out(file, t):
    audio.set()
    os.system(f"start {file}")
    time.sleep(t+1)
    audio.clear()
```

shared I/O resource between the two threads is handled by common function

```
### function which acts as a thread for image capture + analysis.
def start_feed():
    global cap
    cap = cv2.VideoCapture(0)

    while True:
        ret, frame = cap.read()

        ### image manipulation.
        frame = cv2.flip(frame, 1)

        while audio.is_set():
            time.sleep(3)

        ### recognition from preset gestures.
        gesture = mp_model.deterGesture(frame, source=1)
        feedI = ImageTk.PhotoImage(Image.fromarray(cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)))
        feedL.configure(image=feedI)
        feedL.update()

        ### colour format = BGR
        if gesture == "unknown":
            cv2.putText(img=frame, text="unknown", org=(50, 50), fontScale=1,
                fontFace=cv2.FONT_HERSHEY_SIMPLEX, color=(255, 0, 0), thickness=2)

        elif gesture != "none":
            audio_file, audio_len, end = mp_model.gestureOutput(gesture)
            audio_out(audio_file, audio_len)
            os.remove(audio_file)

            if end:
                close()

        feedI = ImageTk.PhotoImage(Image.fromarray(cv2.cvtColor(frame,cv2.COLOR_BGR2RGB)))
        feedL.configure(image=feedI)
        feedL.update()
```

Testing for this module:

Problem	Change	Explanation/example
Pausing camera feed during audio output in order to stop gesture recognition	Making use of time.sleep() in the audio_out() function in to stop any camera frame update	Using time.sleep() will cause the thread to hang for the duration of the mp3 file so that the camera cannot be updated
Jittery camera feed output compromising effective gesture recognition	Although the camera feed is updated twice in the same loop it was much more efficient to not use a separate subroutine because of the delay	Before the if statement is required to show the gesture before triggering output, after the if statement it is required to update message
Unknown vs none for user feedback	Had to separate out the if statement in order to distinguish whether a hand was present in the frame or not	Only wanted to output “unknown” on the camera feed if an unrecognisable gesture is given but not when no gesture is given at all

Development Stage 5

I then wrote the “close()” function to make sure both methods of closing the tkinter application resulted in a “clean” exit (ie. all the unused files deleted, and camera connection closed).

```
### function which is called by exit button or gesture command to close the
### application and remove any files which were waiting to be output.
def close():
    for f in fileData:
        try:
            os.remove(f[0])
        except:
            pass

    cap.release()
    cv2.destroyAllWindows()

    root.destroy()
```

In order to provide useful audio outputs for each gesture I then when back into hand_class.py and wrote a new method in order to fulfil this function call from above:

```
audio_file, audio_len, end = mp_model.gestureOutput(gesture)
```

```
194
195     def gestureOutput(self, gesture):
196         end = (gesture == "goodbye") * True
197
198         ### imports
199         import bbc_feeds
200         import requests
201         from gtts import gTTS
202         from mutagen.mp3 import MP3
203
204
205         if gesture == "hello":
206             output = "welcome to your personal assistant"
207
208         elif gesture == "goodbye":
209             output = "thank you for using the personal assistant"
210
211         elif gesture == "forex":
212             rate_data = requests.request("GET",
213             'https://api.exchangerate.host/convert?from=GBP&to=USD').json()
214
215             output = "exchange rate of 1 GBP to USD is {:.2f}".format(rate_data["info"]["rate"])
216
217         elif gesture == "weather":
218             weather_data = requests.request("GET",
219             """https://api.tomorrow.io/v4/weather/forecast?location=london&timesteps=1d&
220             units=metric&apikey=eIj2AEBht9V8WkKdoKZ45PIAou8Y1TCr""").json()
221
222             cloud = weather_data["timelines"]["daily"][1]["values"]["cloudCoverAvg"]/100
223             precip = weather_data["timelines"]["daily"][1]["values"]["precipitationProbabilityAvg"]
224             temp = weather_data["timelines"]["daily"][1]["values"]["temperatureAvg"]
225
226             output = """average london temperature tomorrow is {:.0f} degrees with {:.0%} cloud
227                 cover and {:.0%} chance of rain""".format(temp, cloud, precip)
228
229         elif gesture == "headline":
230             output = bbc_feeds.news().world()[0]["title"]
231
232         else:
233             output = "live long a prosper"
234
235
236         file = f"{self.source}/{gesture}_message.mp3"
237         gTTS(output, lang="en").save(file)
238         length = MP3(file).info.length
239
240         return file, length, end
241
242
```

New method written in previous class for MediaPipe integration

For those outputs which required web-scraping/ library dependencies, most efficient method was to isolate delay using if statement

Although this is a repeated routine (also used in google cal. side) which could have benefitted from modular design, they required different return criteria, so I felt it more appropriate to keep the two sides separate

6 Evaluation

6a Post Development Final Testing

The following testing will be used to inform and justify evaluation, as planned prior to development:

1) Performance testing → to test the separate and ultimately combined main program on performance factors such as speed, accuracy, clarity, efficiency etc.
2) Acceptance testing → to test the features and usability of the overall product from the potential user perspective (ie. stakeholder feedback)

Performance Testing

The following evidence demonstrates several calendar events and hand gestures being handled and the resulting output of the main program.

https://youtu.be/TUVfeoefOTU

Summary

Key points	**Expected outcome**	**Actual outcome**	**Notes**
Google calendar reminder TTS output	On time, clear and simple to understand	✓	This was test 1 – exactly on time as shown by on-screen clock
Google calendar event TTS output	On time, clear and simple to understand	✓	This was test 2 – exactly on time as shown by on-screen clock
Google calendar future event TTS output	Clear instructive information, only necessary/ useful information provided	✓	This was test 3 - provided only useful information was how much time left and time and duration of the event – in fact even more on time than windows notifications (see last 30s)
Gesture recog. pre-set output (palm)	Immediate, clear response	✓	Slight delay due to being first mp3 file therefore having to wait for audio player to launch
Gesture recog. non-pre-set output	Minorly delayed, clear and sensible response	✓	Example of at least one headline, weather, exchange rate and Spock message shown in the video
Gesture recog. pre-set output (fist)	Immediate, clear response resulting in clean termination of the program	✓	Final gesture before termination, the red highlighting shown is syntactical and does not show any error
GUI proof-of-concept usability	UI is clear and simple to use	✓	See image 1 below
Additional features	They are non-invasive and actually useful	✓	See image 2 below

Image 1

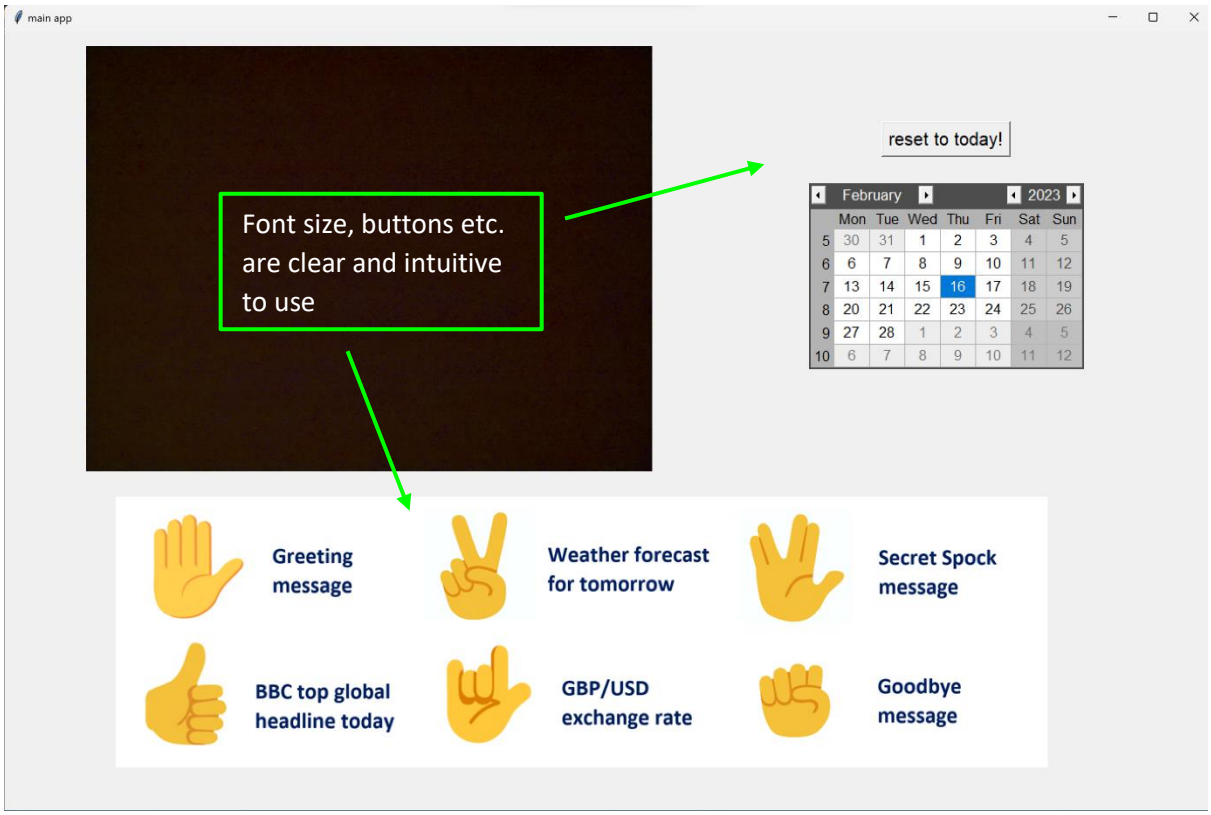
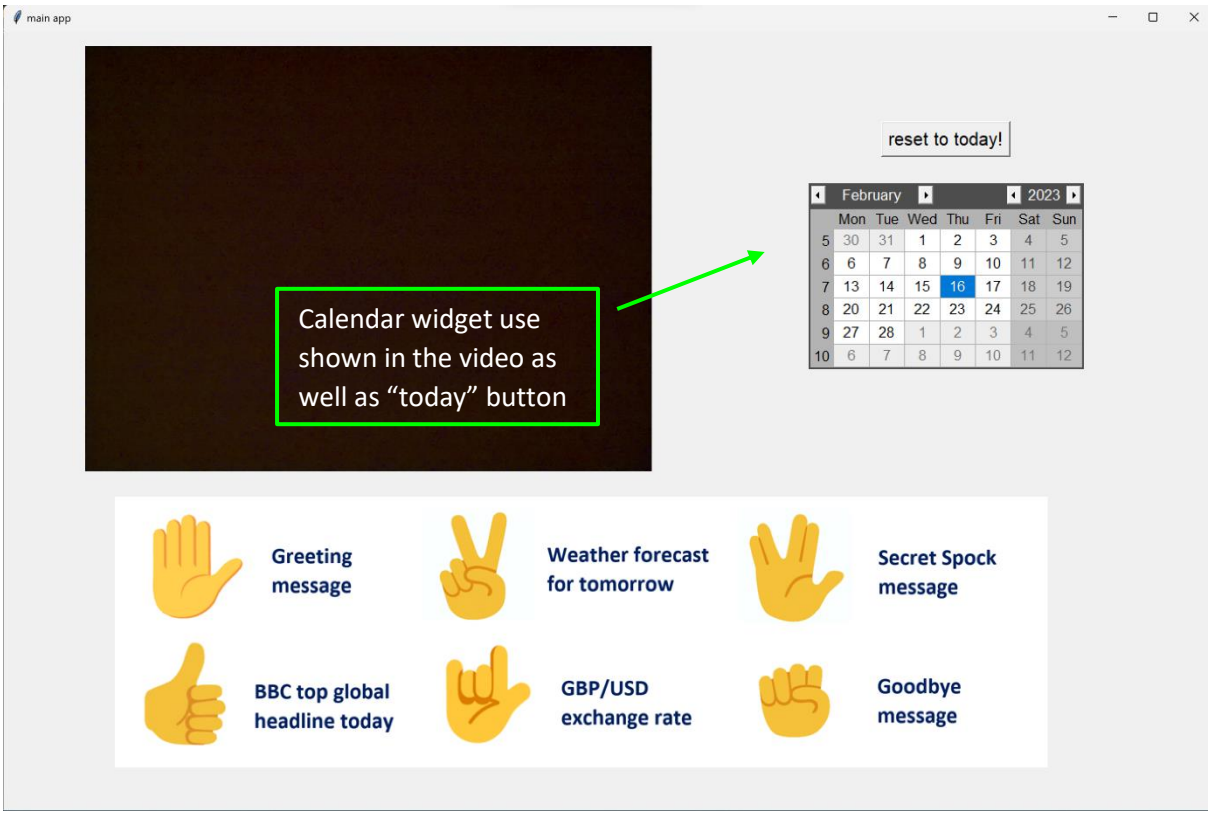


Image 2



Acceptance testing

The following questions were designed in the initial testing plan before development took place:

1. Do you feel that this product has benefitted you beyond the other time management tools/ methods you use? And how/ why?
2. Do you feel this product has benefitted you in conjunction with other time management tools/ methods you use? And how/ why?
3. Can you specify some positive features of the product? Expand on your experiences:
4. Can you specify some negative features of the product? Expand on your experiences:
5. In what situation would you consider this a useful/ helpful product to have? (eg. a type of person/ environment/ job etc.)
6. What do you feel is something that is missing from this product and/ or something that shouldn't be a part of this product?

I feel that although the outcome of my project has changed (a GUI prototype has been developed instead of a standalone IOT device) this set of questions can still be used to gain important insight into the final product in terms of software features and functionality from the stakeholder (ie. potential user) perspective.

Person A:

They were pleased to find that despite the prototype making use of desktop GUI, none of the functionality was dependent on it and they could make use of the product without looking/ when forgetting to wear reading glasses etc. which means they were able to work more efficiently.

However, they also highlighted that "more proactive use of colour would suit partially blind/ visually impaired people" better in terms of UI design which would be required if, for example, the project was implemented as a smart watch/ on-the-go device.

Person B:

They were clear about the value of the product as it is a "hands free extension of the google calendar interface" meaning it was more assessable and adaptable to different circumstances. They found it to be much more effective than manually looking up/ setting up lookup tools and having to keep track of time independently.

They also highlighted major benefit of how it has reduced their time wasted scrolling social media/ newsfeeds etc. for latest information. However, they also mentioned an improvement being customising/ deactivating the outputs of the particular gestures (ie. personalisation). They considered personalisation to be the main feature to be lacking in the product because this is most important factor in people's personal working style/ routine etc.

Summary

Both stakeholders were able to see the hands-free/ screen-independent value in the product and both found it to be of value in everyday life including outside of working hours when you just need an update, hear about upcoming events/ notifications, or want to hear the latest news, for example. Both stakeholders were also satisfied with the proof-of-concept turn taken during development and the subsequent use of desktop UI despite a key element of the design being no graphical interface.

Therefore I feel that the project has been successful in terms of usability from the potential user perspective, considering the adaptations which had to be made.

6b Final Assessment of Success Criteria

Evidence of each success criteria referenced during Design Stage is condensed here. **SEE KEY BELOW**

Success Criteria	Reference/ Explanation	Fulfilled?
1 Google calendar API implementation is able to retrieve all of the correct and up-to-date data	Shown during development, test input via web interface and resulting output in console window	
2 Data is updated periodically per unit time as decided eg. per hour	At the current stage, event data is updated once the current events have been dealt with (ie. at the start of each day), this needs to be worked on to ensure updates every 30 mins – 1 hr	
3 Database is used to store/ manage/ order the event data to ease further processing	This has been explained through written code and screenshots on how the db is implemented during mid-development testing stages	
4 Information is retrieved at the accurate output time and text-to-speech functionality works	Video evidence was provided to show specific output with on screen clock to demonstrate accuracy	
5 cv2 and mediapipe library function together to process each frame for landmark positions	Video evidence was used in development to demonstrate mediapipe output on live camera feed	
6 Gesture matrices are accurately built from test data and stored in accessible location	During development, use of binary files were explained and justified for storing and accessing gesture matrices in order to be referred to within gesture recognition	
7 Error calculation method can reliably determine, and output, gesture received	Code was explained and testing breakdown included many screenshots to show variation and capability of the API, as further shown in video evidence	
8 GUI is sufficient to demonstrate the functionality (not a direct objective)	As this is subjective feature of the program it can always be improved with more user feedback, also GUI was in proof-of-concept methodology therefore I consider this a successfully fulfilled compromise	
9 Text-to-speech and audio output resource is shared effectively between the two sides	Main program planning and development explained how the use of I/O locking and final video demonstrates well managed audio resource in full program	

System Requirement Assessment Results Key

Majority of criteria were successfully fulfilled showing overall success with the project development goals.

Some future development goals can be drawn from the remaining results.

No completely unfulfilled criteria showing no neglected success points and all-rounded approach to development and testing.

6c Overall Project Review

Limitations

- Not on Raspberry Pi
  - » Due to changes in the success criteria to adapt to hardware difficulties (as explained previously) a major limitation is the GUI interface as alternative to Raspberry Pi Pico
  - » Main program was intended to run on Raspberry Pi Pico, therefore in terms of future development, the main project will require I/O facilities (camera and speaker) as well as Wi-Fi module to connect to the internet to retrieve google calendar and web scraping data
  - » After developing a fully working system, it is clear this project has a high storage demand therefore Raspberry Pi system would also have to manage this either with:
    - On-board physical storage of reasonable capacity however this will require high speed as image analysis per camera frame is a very fast process
    - Online (cloud) storage which is somewhat unlimited storage in this sense because a single user using this product will not overrun, for example, free google cloud storage limits however this introduce a new issue of security and privacy for the user which would have to be dealt with
- Software dependencies
  - » The program requires several python libraries in order to run such as for interacting with google calendars and managing that data with SQLite, performing frame-by-frame mediapipe gesture recognition, real time audio output, web scraping for gesture outputs etc. etc.
  - » have managed to set up all of these dependencies without requiring any specific user details/ data apart from google calendar API which is inherently covered by google themselves and the user's original google account contract
- "eventData" update issue
  - » Event data does not update periodically as stated in the requirements, and this is a hinderance to the user as the program will not be able to sync with any updates made on their Google Calendar during the day

    - » Currently, “eventData” is recalibrated when the current run of synced events finishes therefore at minimum automatically by the end of the day, however it can be manually recalibrated whenever the app is closed and restarted (not minimised)
    - » I have spoken more about how I would work on this in future development goals, see below
- Initial/ different user Log In
    - » Currently in order to access google calendar API I have used the project system provided by google to run an authentication before drawing data which requires reconnecting every 7 days
    - » Under the assumption there would be a single end user using a standalone IOT device (Raspberry Pi Pico) I did not plan or design any direct log in system which in hindsight would have been the appropriate approach
    - » In the main code, the “gCalSetUp.py” can be used to log in via any google account available on the device to authenticate the API however this is a manual process, and an improvement here would be to add it to the main program in some way
- Error Handling
    - » In an effort to complete the success criteria and produce a full working program I have neglected to include the error handling required to deal with the uncommon issues even though they may seem trivial such as:
        - unstable/ insufficient/ inconsistent internet connection leading to inability to draw google calendar data and/or web scrape for gesture outputs
        - weak/ interrupted connection to google calendar API resulting in empty event data
        - camera/ speaker module fault/ error (in terms of both software and hardware) meaning frames cannot be processed or audio cannot be output
        - etc. etc.
    - » These features make up the basis of the whole program and will require more attention in future development to ensure reliable system, see below

Accessibility

- Independent of the GUI
    - » Main program is able to function independently of the GUI interface ie. minimising the window will not hinder any functionality
    - » This allows for more freedom in how the user uses the product and more adaptability for different working styles/ layouts
- The GUI interface does not require any input/ output from the user
    - » Although the proof-of-concept method introduced a digital screen into the solution which was the main interface I wanted to avoid in the initial proposal, I made sure to fulfil my overall aim of helping visually impaired people
    - » Therefore my project still fulfils the initial goal I had to produce a productivity tool which helps visually impaired/ partially blind people where the interface does not depend on typed input/ written output
- Adaptability in future Application
    - » due to the current version of the program being a fully functioning desktop app with minimal GUI, the applications of this program are wide depending on the path chosen, for example, could be embedded into smart (possibly wearable) devices, could be built into desktop systems, could be placed anywhere around the house

Maintenance

- Many software dependencies
  - » To help deal with libraries/ APIs needed, I have written a requirements.txt file to manage which ones have to be installed
  - » These are needed in order for the main program to run smoothly and for all functionality to be accessible on a new device
- No backup/ error message
  - » Although minor error handling is used in the main program, initially raw data from gCal API is manipulated and stored for audio output
  - » Any error/ corruption/ incomplete information could possibly cause problems further in the in the program or confusion for the user, so this needs to be dealt with
- Many data requirements
  - » Another problem which could cause unwanted damage to the usability of the program is deletion/ corruption of the core files necessary for the main functionality to run
  - » This includes the google calendar token, json project key, calendar databases, gesture matrices etc. etc. - see below for how I would deal with this

6d Future Development Goals

Goal	**Actions**	**Aim**
1 Implement Raspberry Pi Pico as Main System	- plan out my time better, make use of project management tools to achieve this - prepare more raspberry pi knowledge in advance to be able to fulfil initial criteria - research more into existing solutions to build more cohesive/ adaptable solution	Limitation Improvement
2 Improve gCal system to update event data periodically	either - introduce a new variable to control “time since last update” or reverse and force update periodically or - adapt the original function to only receive event data from the coming hour leading to minimum of hourly update	Limitation improvement
3 Produce automated script for installing dependencies	- automated script will have to have administrative authority so the program would need user permissions so that it can check and install dependencies before running - this will save time for the user and increase ease-of-use	Limitation improvement
4 Introduce log in system	- this will allow new users to log in with new google account, and existing users to recalibrate the calendar API or change accounts - if Raspberry Pi Pico is used as the main interface, then an initial GUI setup may be required to fulfil this criterion	Limitation improvement

5 Ensure error handling throughout	- implement error handling to automatically deal with initial or sudden loss of connection to API, google calendars, physical modules etc. with appropriate error message and restart the app	Limitation improvement
6 Setup error message and output for missing/ incomplete calendar data	- this will remove risk of crashing/ buffering/ lagging when data transmission from gCal API is unable to communicate clearly - this will improve user accessibility as someone who does not have specialised knowledge and is not able to understand/ fix the problem will not be left with a faulty program	Maintenance improvement
7 Setup backup system for important files	- important files include the gesture matrices (binary) files and the calendar databases (both cal_data and output_files) either - an automated script for rerunning the code which reproduces these files or - an online available copy of the files could be provided for users who have accidently deleted/ corrupted it	Maintenance improvement

These would be the second iteration success criteria if I, and/or other developers, were to pursue further development for this project in the future.

7 Full Project Link

Here is link to the full project as a read-only private GitHub Repository hosted on https://gitfront.io: https://gitfront.io/r/TamannaKar/tZVjAX9H7TcS/A-Level-Computing-Project/ (reload to view fully)

If the project is cloned/downloaded for testing, dependencies can be installed using command line: **pip install -r /path/to/requirements.txt**

8 Bibliography

During Analysis:
https://medium.com/@mikevardy/the-problem-with-the-pomodoro-technique-b51a07f4a62e
https://visionaware.org/everyday-living/helpful-products/using-apps/seeing-ai-app/

During Investigation and Design:
https://ecosmarthomepros.com/pros-and-cons-of-smart-home-technology/
https://www.youtube.com/watch?v=tLJlOh1UlHw
https://medium.com/mlearning-ai/real-time-hand-gesture-recognition-2bff427a6a89

During Development:
https://stackoverflow.com/ https://www.programiz.com/ https://www.geeksforgeeks.org/
https://www.guru99.com/ https://www.youtube.com/ https://www.w3schools.com/
https://www.tutorialspoint.com/index.htm https://console.cloud.google.com/
https://developers.google.com/calendar/api/guides/overview https://sqliteviewer.app/
https://github.com/csmailis/NPYViewer https://pyimagesearch.com/
https://python.readthedocs.io/en/stable/library/tkinter.html https://scribles.net/
https://www.analyticsvidhya.com/blog/2021/09/a-beginners-guide-to-image-processing-with-opencv-and-python/ https://google.github.io/mediapipe/ https://realpython.com/