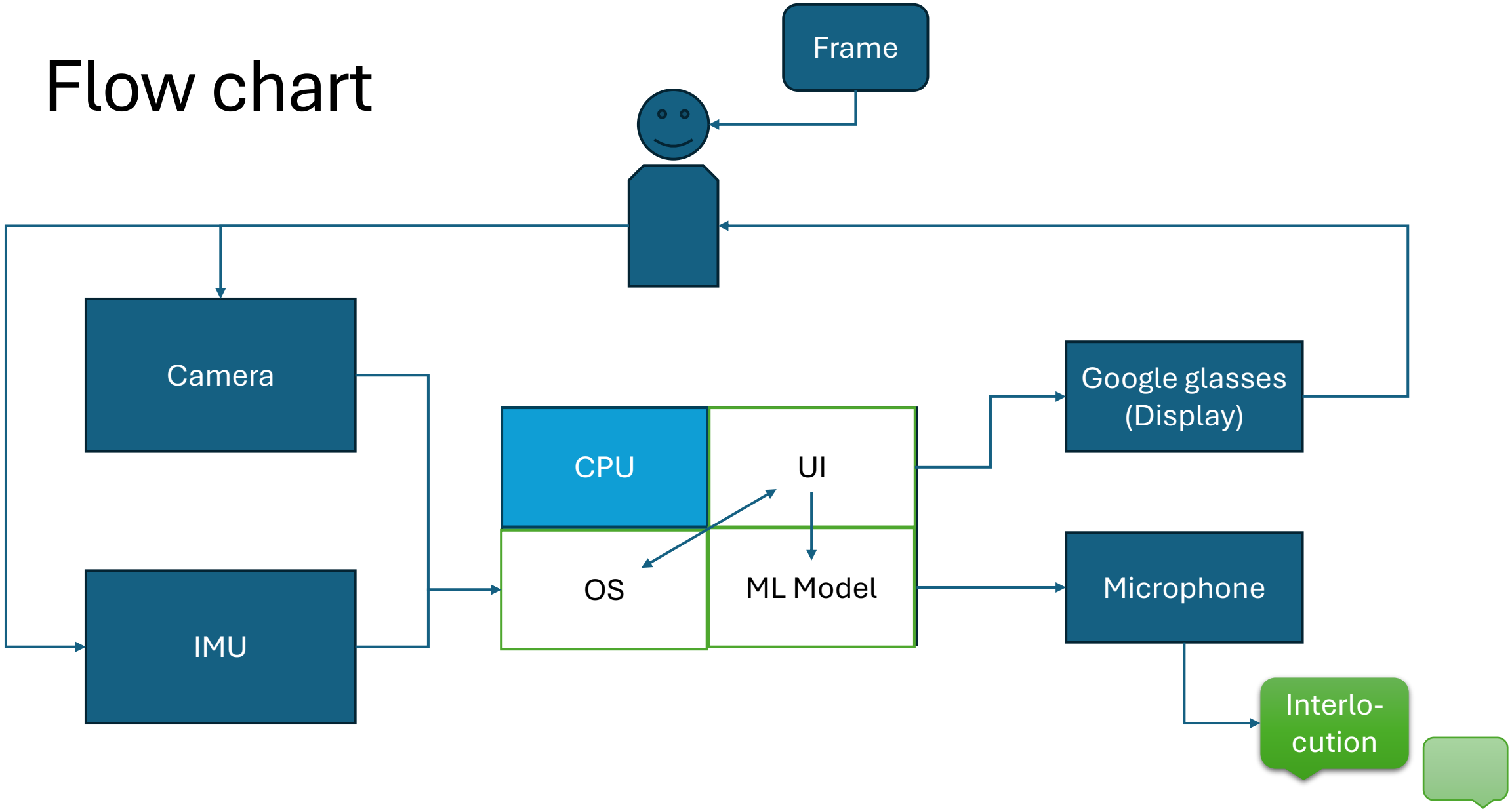
Eye-tracking glasses

DAPP Group 3

Key components

- Camera & Eye-tracking
- UI design
- CPU
- Text-to-speech generator
- IMU sensor
- Batteries
- Visual display and microphone
- Frame

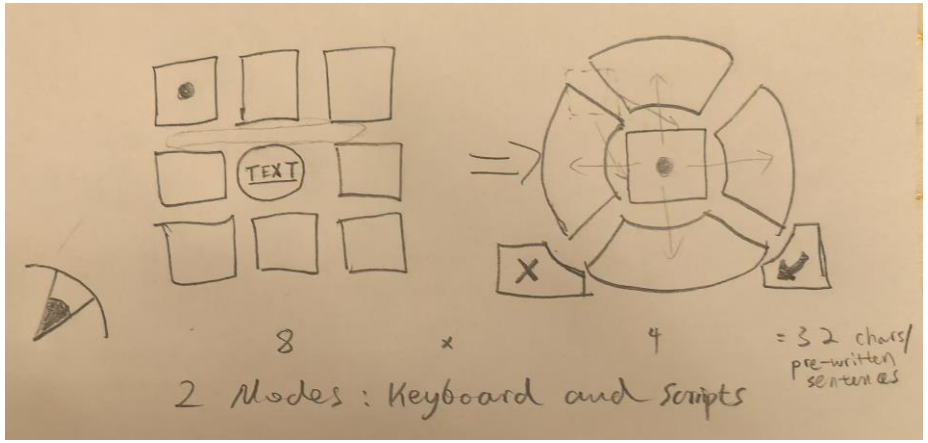
Flow chart


Frame
Camera
IMU
CPU
UI
OS
ML Model
Google glasses (Display)
Microphone
Interlo-cution

Camera and eye-tracking

- Motion, direction and blink
- Sampling from one eye will probably be enough
- Trace the continuous-time motion and need to record the event up to a few seconds
- Apply filters and remove glares
- Figure out where and how to place the camera

UI design



- Interface design and programming
- Two modes, one for typing and one for pre-selected scripts
- Two panels, one has 8 directions and the other has 4, which further specifies the choice
- Need to be interactive, learn from the existing body of software and website design knowledge
- This should be a small and complete OS program stored in the CPU like your Android/Apple/Harmony powered mobile

Text-to-speech generator

- Generate sound files that can be directly played by the mic
- Ideally, this could be a small-to-medium sized ML model that is being trained on the patient's data (one of our teammate's)
- The model's parameters are stored in the CPU, may need to write a small script to prepare the model's output into proper sound file
- Alternatively, if that's shown to be too difficult, we use the existing platform that has a ready-made solution

IMU

- Pick up on the user’s head movements and convert into commands that supplement eye motions
- Need to decide on how we should define and encode different commands, how they correspond to the movements intuitively
- Will need to write a small script to process the data
- How to place? Where to purchase, and what type?

CPU

- Raspberries Pi or others
- Low-level programming, coordinating the camera & IMU (hardware) and UI & Text-to-speech (software)
- Manage data stream and perform computations
- Will probably need several programming languages for the system-level design
- Need to consider our needs and specifications realistically, don’t need excess hardware power

Batteries

- What should we use for our final product?
- What should we use for our demonstration?
- Load-line and consideration over the device’s power consumption
- Charging speed, duration of use and battery life
- Heating and cooling, cannot exceed 41 degrees in exterior

Display and microphone

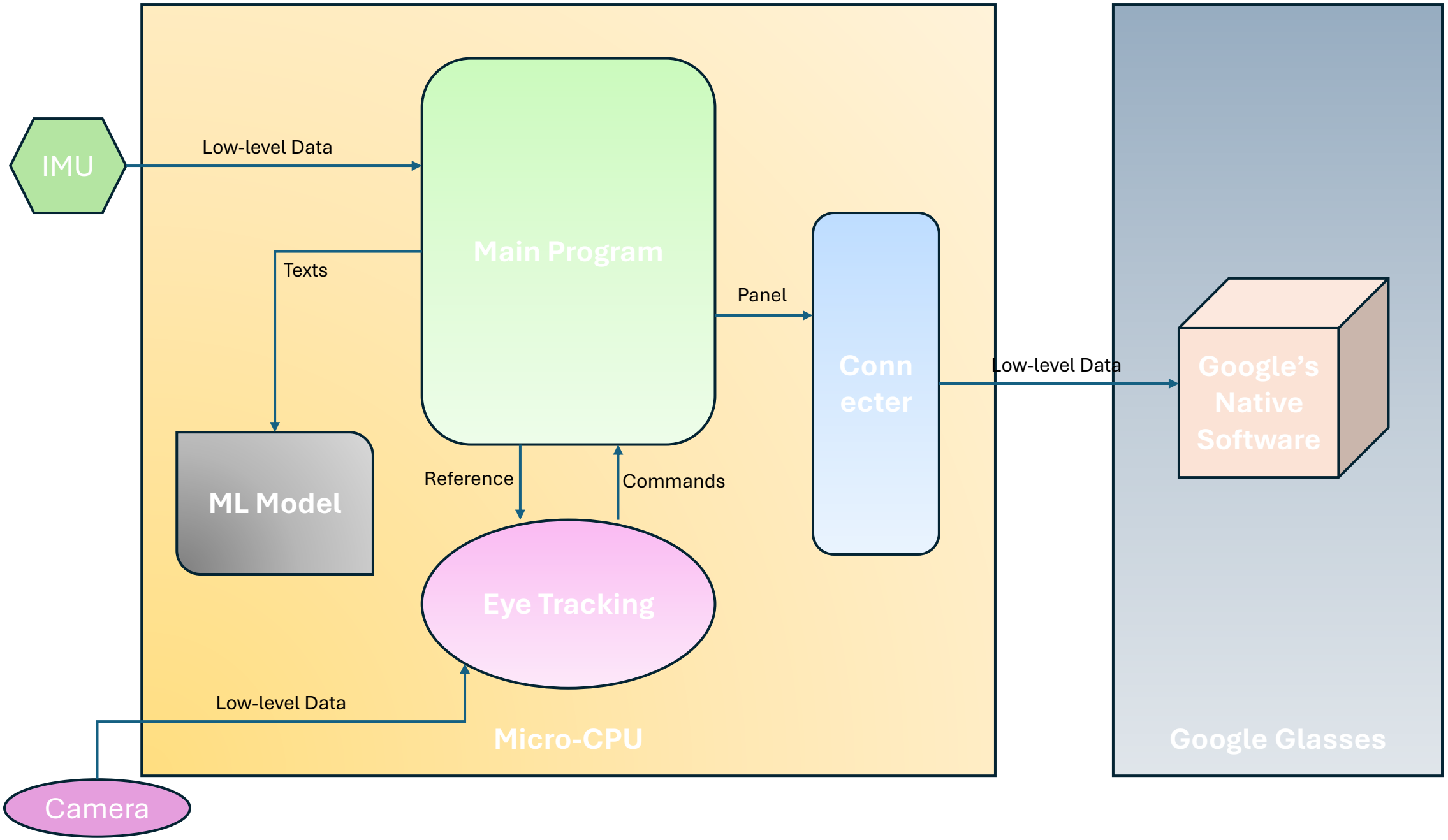
- No. 1 choice is to directly plug into the Google glasses to do the job
- Need to figure out the hardware

Frame

- Mechanical design (hard hardware)
- CAD and 3D printing
- Integrate all components abovementioned
- Our test version just need to put everything together for testing
- Our later version should consider aesthetics, efficiency, safety, etc. (We could only do this once we have decided the model/type of other components)
- If we need to use the Google glasses, consider how to embed it into our frame

Software dev 2025/1/21

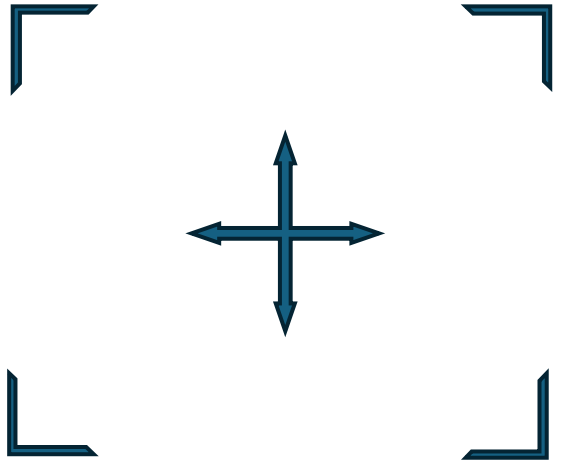
- **Main program** including the display board and menu (Run in the Raspberries Pi)
- **Eye tracking** algorithm (Video data stream send from the camera to Pi and processed there)
- Connecter, helping the Raspberries Pi talk with Google Glasses
  - Display directly, has sound and animated (may not be possible)
  - Send image files constantly and opened with the glasses whilst play the audio using Raspberries Pi
- Text to sound model, a customised, small-sized ML model stored locally in Pi that will generate the MP3 given the text input


IMU
Low-level Data
Main Program
Texts
Panel
Conn
ecter
Low-level Data
Google's
Native
Software
ML Model
Reference
Commands
Eye Tracking
Low-level Data
Micro-CPU
Google Glasses
Camera

Main program

- OOP and procedural
- Menu options: Font sizes, volumes, calibration, mode (typing or saved scripts), create scripts, enter, back, quit, home, …
- State variables: Current panel, selected sub-panel, typed string, saved scripts, …
- Keyboard arrangement: Finalise with the design team
- Commands and responses: e.g. when X command is sent in, type in ‘Y’ or select ‘MODE ONE’, etc.

Eye tracking



- Image processing, filtering to expose the pupil
- Track the angle/coordinate of the pupil
- Calibration
  - We could interpolate the eye gaze coordination
  - The display could be arbitrarily distanced and oriented or scaled
- Interpolating eye gaze coordinate and match onto a pre-defined meshed grid
- Have several built-in panel arrangement data stored
- If the program know which panel the user is navigating (told by the main), it retrieve the command based on the coordinate information in the grid and send this command back to the main program.

Connecter

- At this stage, I have no idea how it’s like, only that it will be a separate program specifically for the Google glasses apart from the main/eye-tracking that is run on the Pi
- Need to fiddle with the Google glasses and see

- Probably need to deal with transmitting and receiving different data files
- Also get the correct credentials so the devices can stay connected
- Involve more lower-level dirty works

Text to sound

- Once the ‘enter’ button is hit in the main, whatever in the ‘typed string’ is relayed to this model
- The model then output a synthesised MP3 sound file that mimics the person’s voice
- It might be wiser to outsource this part to some existing service provider on the internet that can help us train and customise the model (may cost money)
- Internet search: methods (high-level view) on how to build such a model and what services are out there