Drowsiness Detector

COMP 195 Section 1

https://github.com/comp195/spring-2021-final-project-drowsiness-detector/wiki

Jacob Angulo
j_angulo1@u.pacific.edu

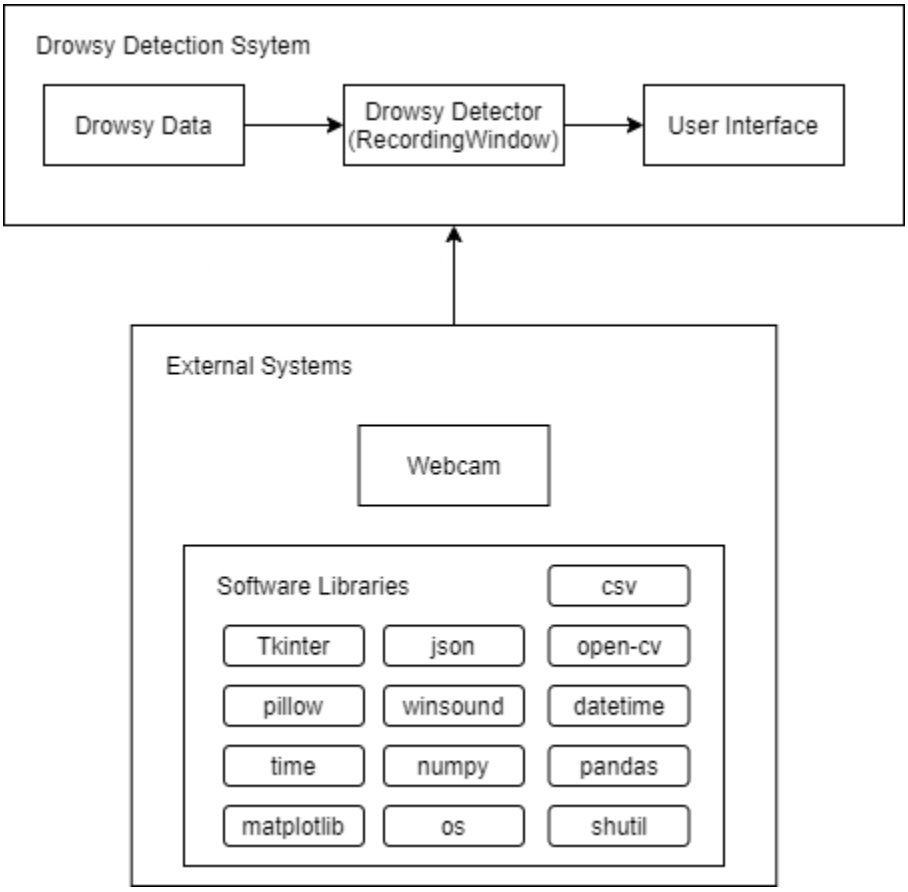
Sonali Patil
s_patil1@u.pacific.edu

Richard Shin
r_shin2@u.pacific.edu

Last Revision: 4/30/2021

Table of Contents

Systems Architecture



Software Modules

Json:

- An encoder and decoder that allows creation of JSON formatted data.

OpenCV:

- An open source software library for computer vision and machine learning.

Pillow:

- An open source software library responsible for image processing.

NumPy:

- An open source scientific computing library for working with arrays.

Pandas:

- An open source software library for data manipulation and analysis.

Matplotlib:

- An open source plotting library used to create data visualizations models.

Winsound:

- A Python module that provides access to basic sound-playing machinery provided by Windows platforms.

Time:

- A Python module that provides time-related functions.

OS:

- A Python module that provides operating system dependent functionality.

Shutil:

- A Python module that offers high-level file operations.

CSV:

- A Python module that implements read and write operations on tabular data in CSV format

User Interface

Tkinter:

- A standard Python GUI package that provides a fast and simplistic way to create GUI applications.

External Systems

Webcam for Computer:

- Computer built-in cameras or USB pluggable cameras

Hardware, Software, and System Requirements

Hardware Requirements:

RAM:

- 512 MB utilization

Disk Space:

- 256 MB utilization (minimum)
- Size will vary depending on amount of exported data

CPU Speed:

- 2 GHz+ Dual Core Processor

Camera Requirements:

- 1080p or 720p Webcam
- Must be Windows 10 compatible

Software Requirements:

Matplotlib:

- This Python library produces figures and data-visualizations in a variety of interactive formats that can be invoked by a Python program.
- Version 3.4.1

Pandas:

- This Python library provides data structures that are designed to make working with structured data simple and efficient.
- Version 1.2.4

Numpy:

- This Python library provides an N-dimensional array object and mathematical functions to use for various calculations.
- Version 1.20.2

Opencv-python:

- This Python library provides the ability to analyze image data from video files and webcam footage.
- Version 4.5.1.48

Pillow:

- This Python library provides image processing capabilities to given programs.
- Version 8.2.0

System Requirements:

Operating System:

- Windows 10 Operating System
    - Version: 20H2 or newer

External Interfaces

Webcam

Webcams compatible with the application include either built-in cameras within a computer or an external webcam that can be plugged into a computer. Camera resolution should be at least 720p for proper eye detection and action monitoring.

Application and Webcam Interaction

The webcam will interact with the application during the recording phase, where the user's eyes will be monitored for when they remain closed for a prolonged period of time. The data collected will then be used to display statistics on the user's drowsiness and sleep.

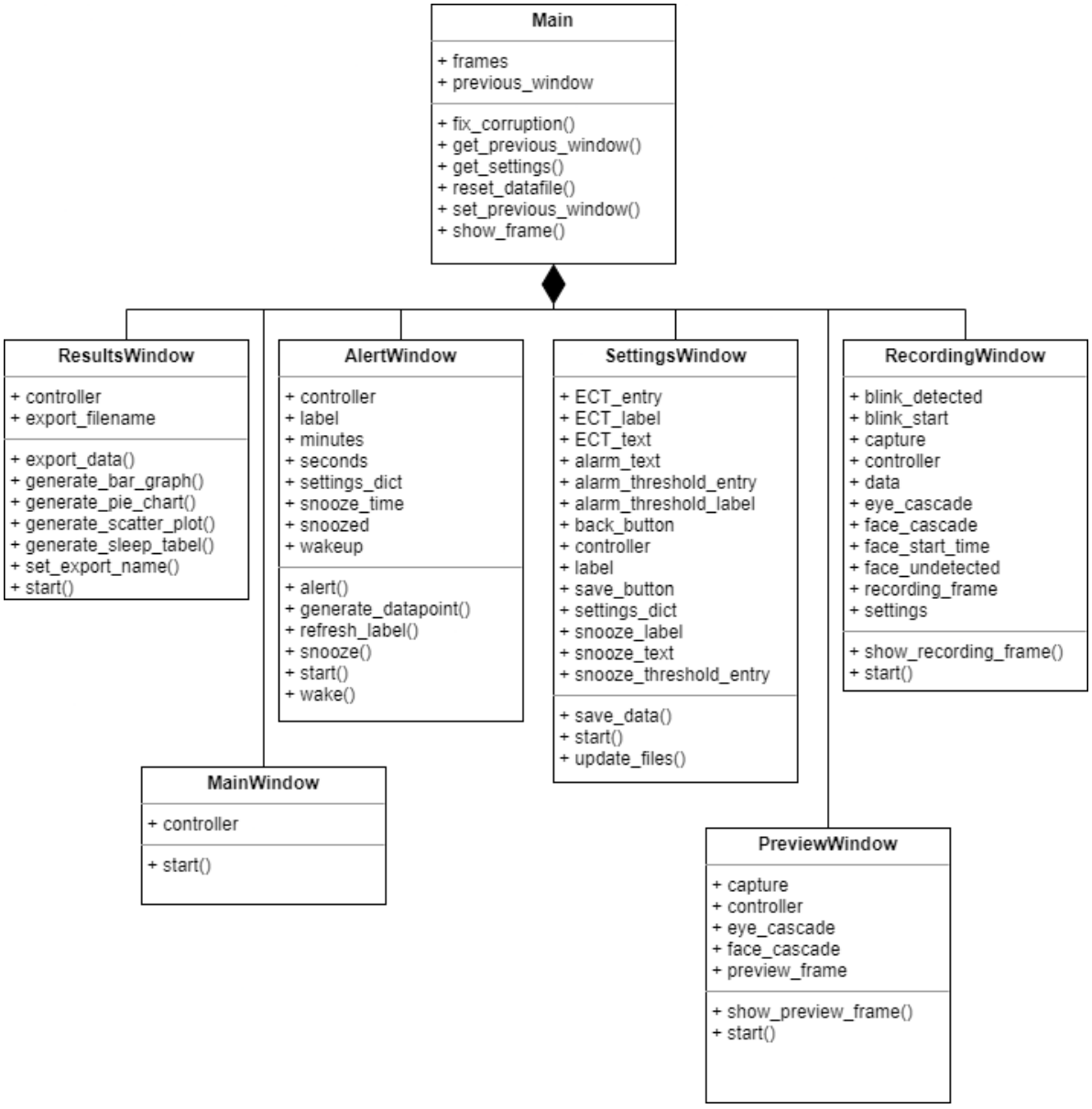
Other Functionality:

- The program will monitor user face activity for attentiveness.
    - A lack of facial detection for a user-specified period of time will cause an alert to appear.
- The program will record the duration of time that an user's eyes remain closed.
- The program will create visualization models with eye detection data collected through the webcam external interface.

Challenges that could occur with the application and webcam interaction include eye detection capabilities not being specific enough, accuracy of ratio measurements with the eye not being precise, and time monitoring being off when detecting eye movement. As it stands, eye and face detection is best effort and through the use of an "eye closure threshold", mistakes made by the eye and face detection algorithms are not translated into data points.

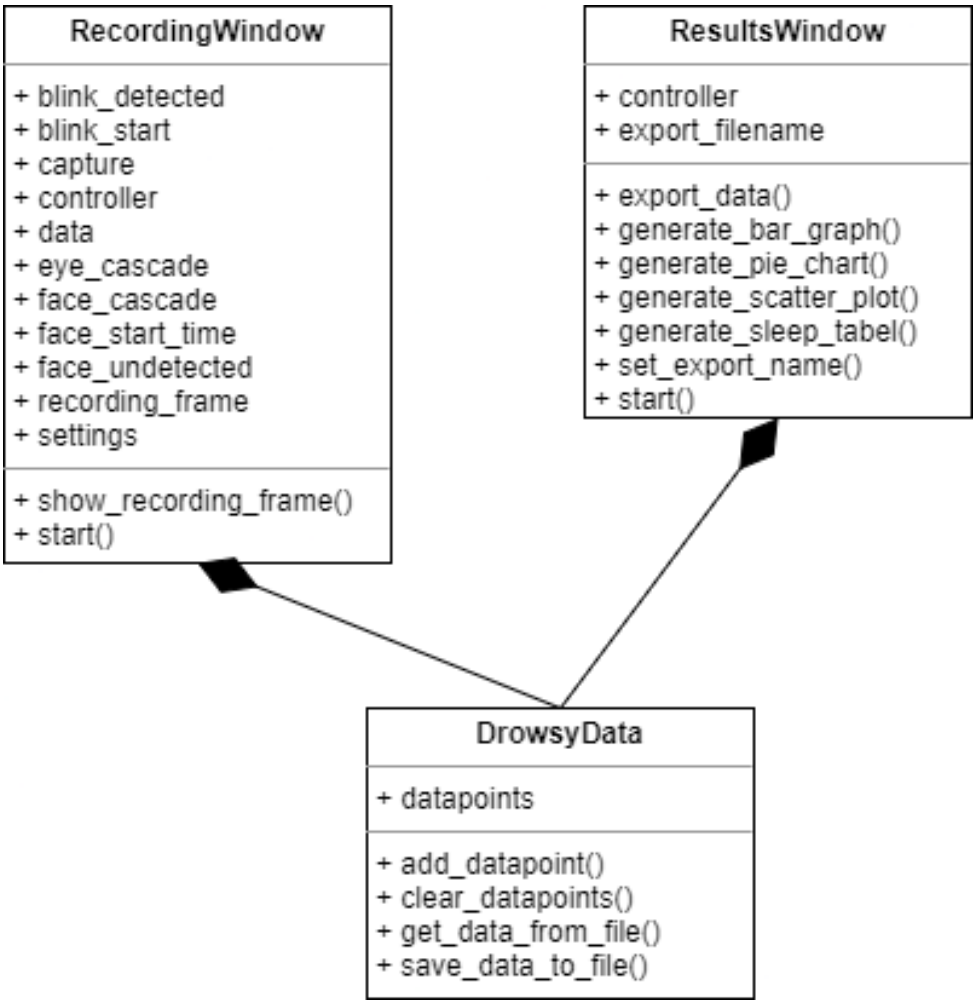
Software Design

Class Diagram A



This class diagram shows the classes that make up the user interface and defines a composite relationship between a specific window (i.e. MainWindow) and the general Main class definition. According to this specification, the individual window classes are instantiated by the Main class, and thus the Main class owns the lifetime of these instances.

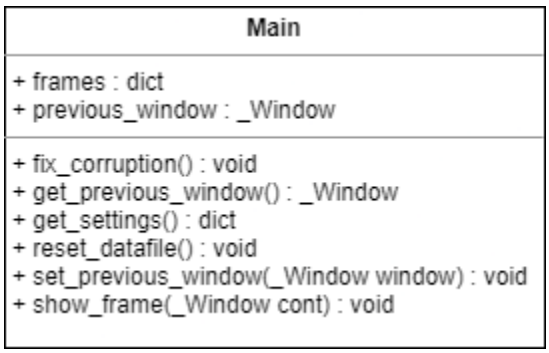
Class Diagram B



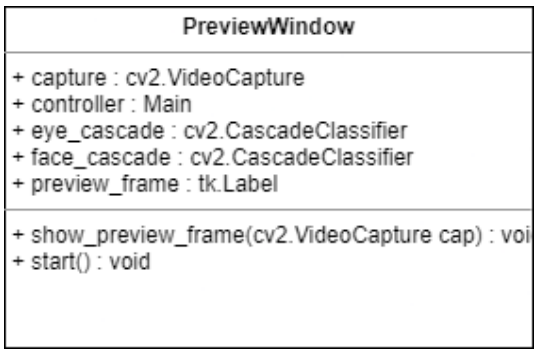
This class diagram shows how the DrowsyData class is related to the Recording and Results windows. DrowsyData objects are composed by the windows, therefore these objects are only active when the windows need the internal data structure (datapoints) and file I/O methods.

Class Specifications

Main Class

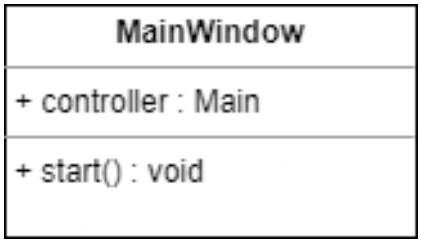


SettingsWindow Class

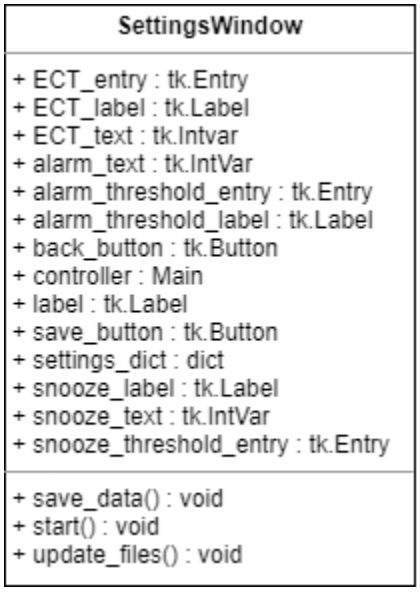


Note: _Window indicates the type could be of any window class

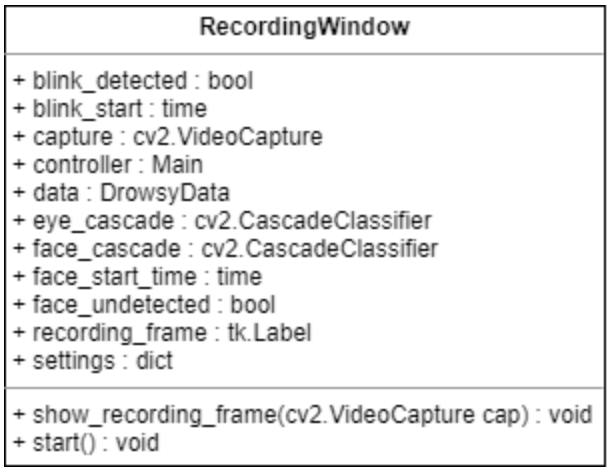
MainWindow Class



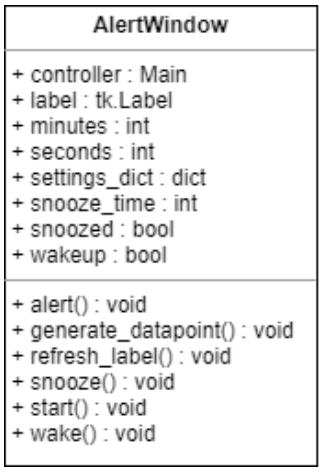
SettingsWindow Class



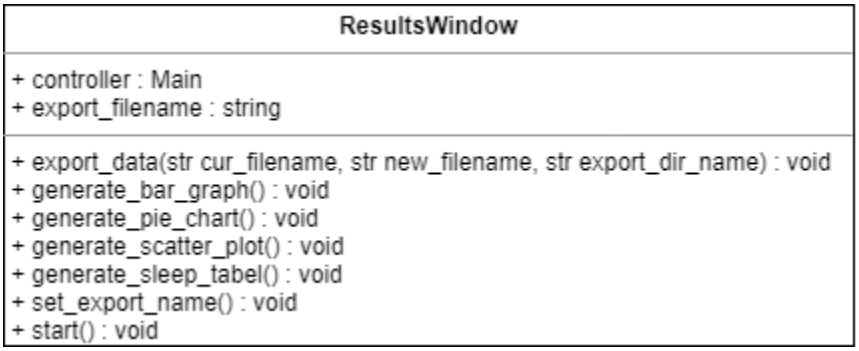
RecordingWindow Class



AlertWindow Class

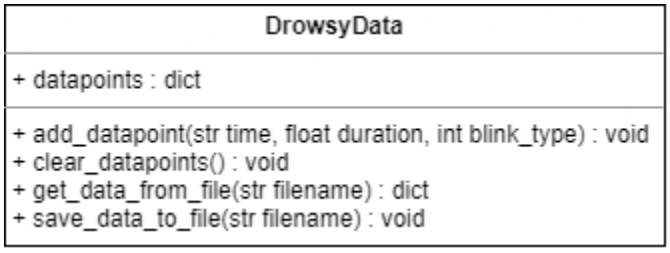


ResultsWindow Class



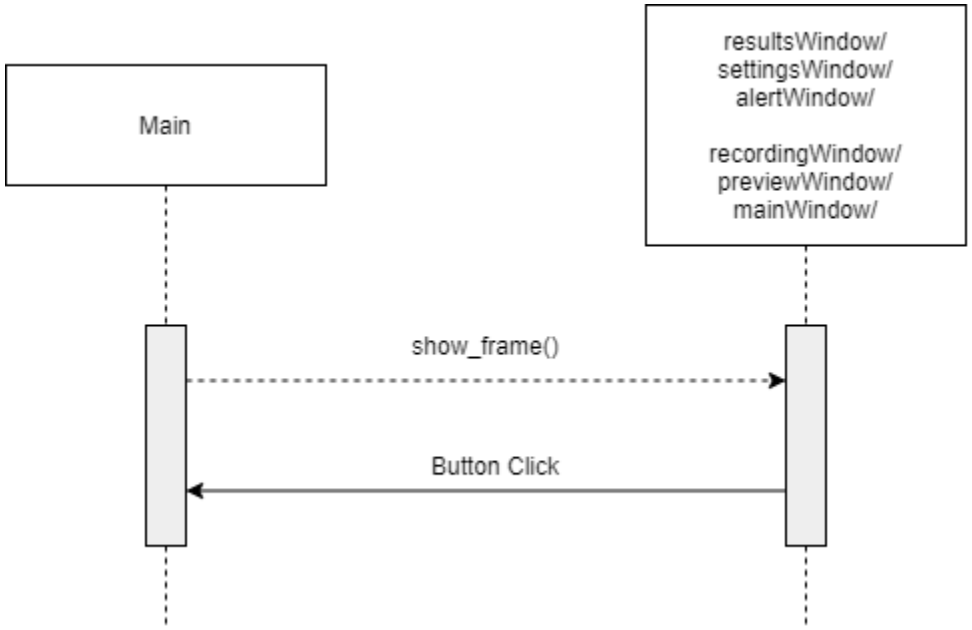
Note: str is an abbreviation for string

DrowsyData Class



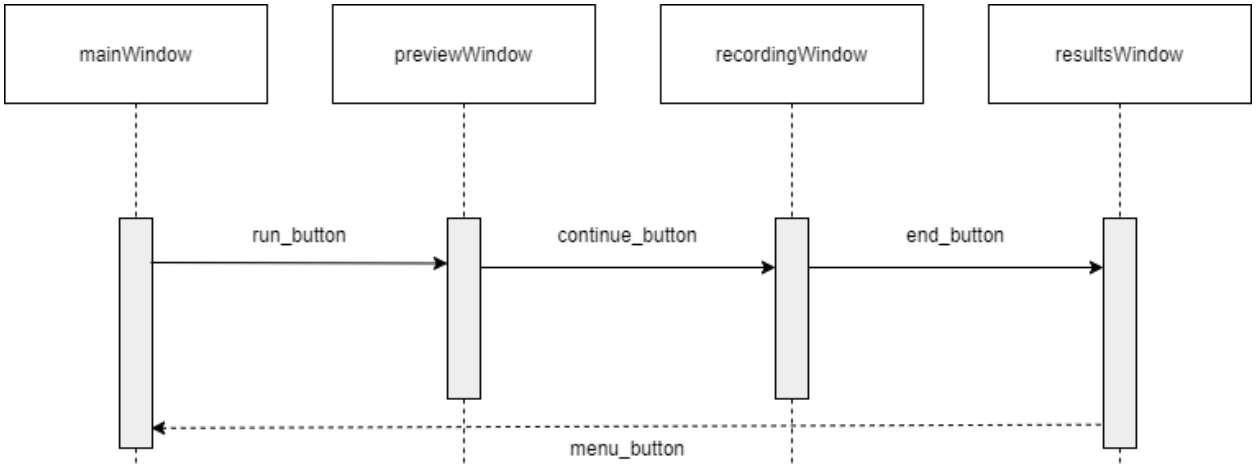
Interaction Diagrams

Main and Window Classes Interaction



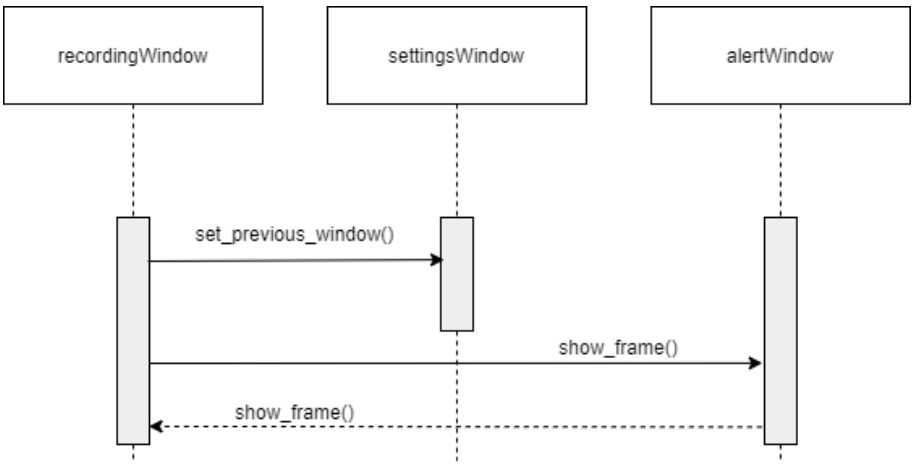
This interaction highlights how the resultsWindow, settingsWindow, recordingWindow, alertWindow, previewWindow, and mainWindow work with the Main class. Main provides the original controller variable and the other window classes use show_frame() to change which window is raised to the front.

mainWindow, previewWindow, recordingWindow, and settingsWindow Interaction



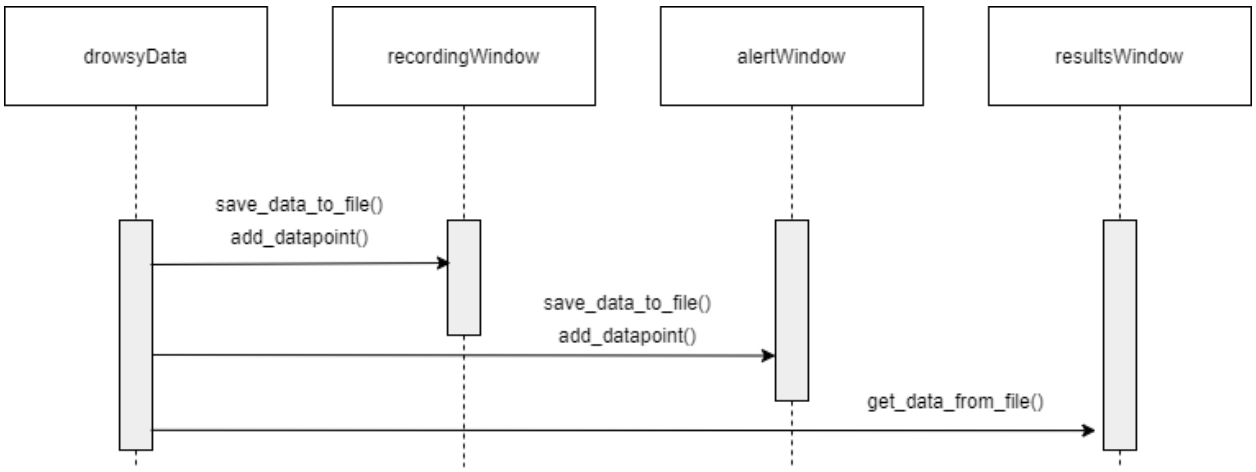
This diagram highlights how mainWindow, previewWindow, recordingWindow, and resultsWindow interact with each other. After the Run button is clicked, mainWindow travels to previewWindow. When the Continue button is clicked, previewWindow travels to recordingWindow. From there, when the Back button is clicked on the resultsWindow, it travels back to the mainWindow class.

recordingWindow, settingsWindow, and alertWindow Interaction



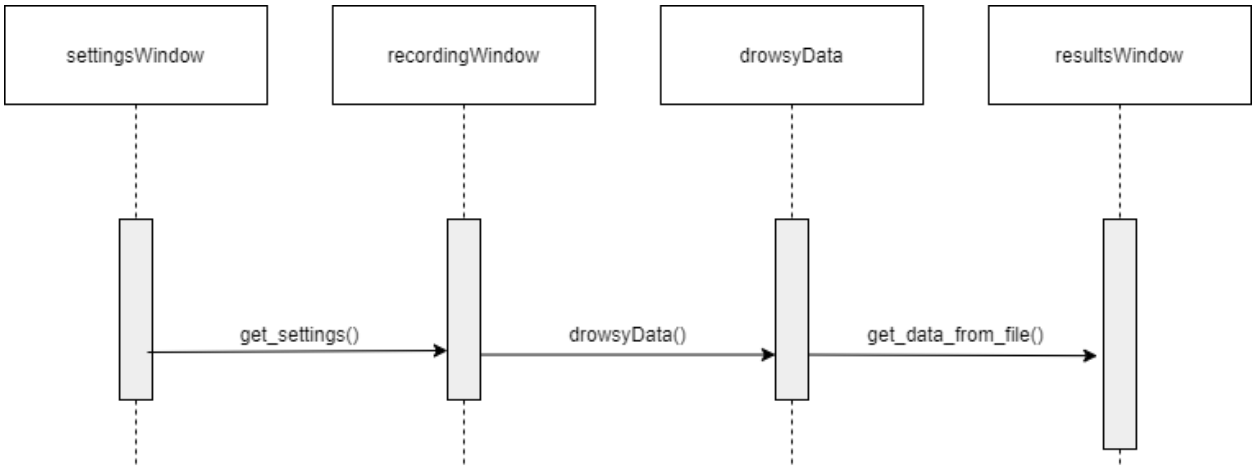
This diagram shows how recordingWindow interacts with the settingsWindow and alertWindow classes.

recordingWindow, alertWindow, resultsWindow, and drowsyData Interaction



This diagram shows how recordingWindow, alertWindow, and resultsWindow interact with the drowsyData class. When the blink threshold is passed on the recordingWindow, it calls upon add_datapoint() to add the datapoint to the csv file. This datapoint contains a datetime value and a duration value for how long the blink lasted. When the alert threshold is passed on the recordingWindow, save_data_to_file() is called upon to save all of the data points to the csv before recordingWindow travels to alertWindow. When generating the graphs based on the collected data, resultsWindows uses get_data_from_file() to get all the data points collected from the drowsyData class.

settingsWindow, recordingWindow, resultsWindow, and drowsyData Interaction



This diagram shows the interaction between settingsWindow, recordingWindow, resultsWindow, and drowsyData. To get the threshold values for blink and alert detection set in the settingsWindow, recordingWindow calls upon get_settings() to obtain the indicated values. To get the functionalities of drowsyData to add data points and save them to the data file, recordingWindow calls drowsyData(). Finally, when generating the graphs based on the collected data, resultsWindows uses get_data_from_file() to get all the data points collected from the drowsyData class.

Design Considerations

The various window classes were designed to be generalized by the Main class. Therefore, any quality of life features that get added to all windows can simply be added to the Main class as either a method or a member variable. The GUI frames that the user will interact with are represented by these classes and thus they are the starting point of any user.

The settings window (leveraging the settingsWindow class) will save the settings into a settings file located in the same folder as the program. This file will hold data such as the time threshold for blink detection, time threshold for an alert to be triggered, and the snooze interval. When the drowsiness detection system is instantiated, various calls will be made to retrieve and implement the values from the settings file.

Classes such as recordingWindow, alertWindow, and resultsWindow leverage drowsyData methods and therefore require the class for full implementation. The recordingWindow class's data point functionality is provided by drowsyData, while alertWindow also requires the same functionality to add ending data points. The alertWindow class also has the power to take the user to the recordingWindow phase and save those previous data points. The final class, resultsWindow, gathers data and creates its graphs using drowsyData which retrieves its data from the existing csv file.

Another file that will be leveraged by this program is the historical data file which is a CSV file. This file will hold date-time, duration of eye closure, and type of blink event. Data obtained during runtime will be saved to this data file and can be recalled whenever necessary. The drowsyData class is responsible for collecting these data points, saving the data points to the file, and eventually getting the data points when they are needed to generate the graphs in resultsWindow.

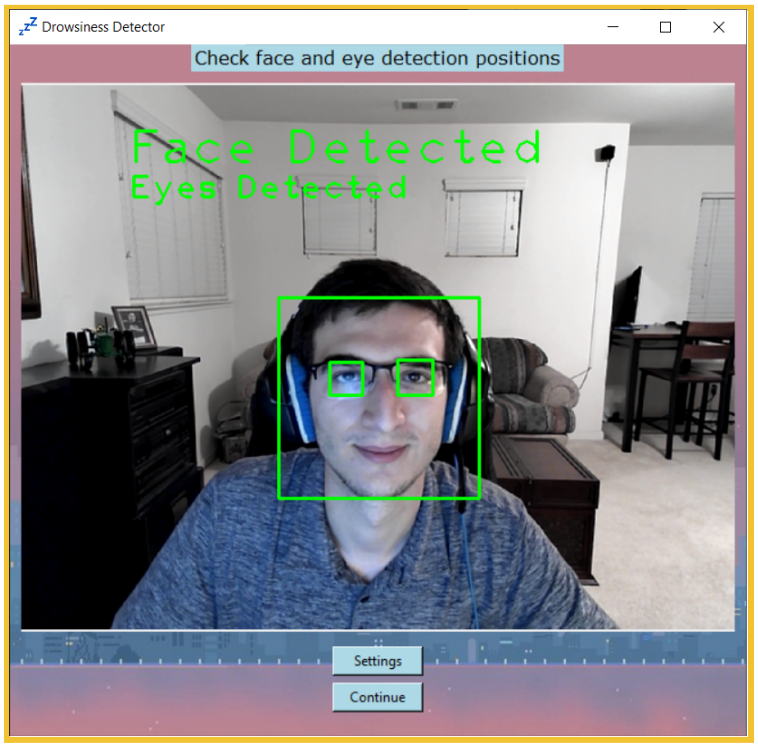
There are three event types: A blink that only passes the eye closure threshold (ECT), a blink that passes the ECT and alarm threshold, and when the 'Wake Up' button is pressed. These events are 1, 2, and 3 respectively in the CSV.

User Interface Design

User Main Window

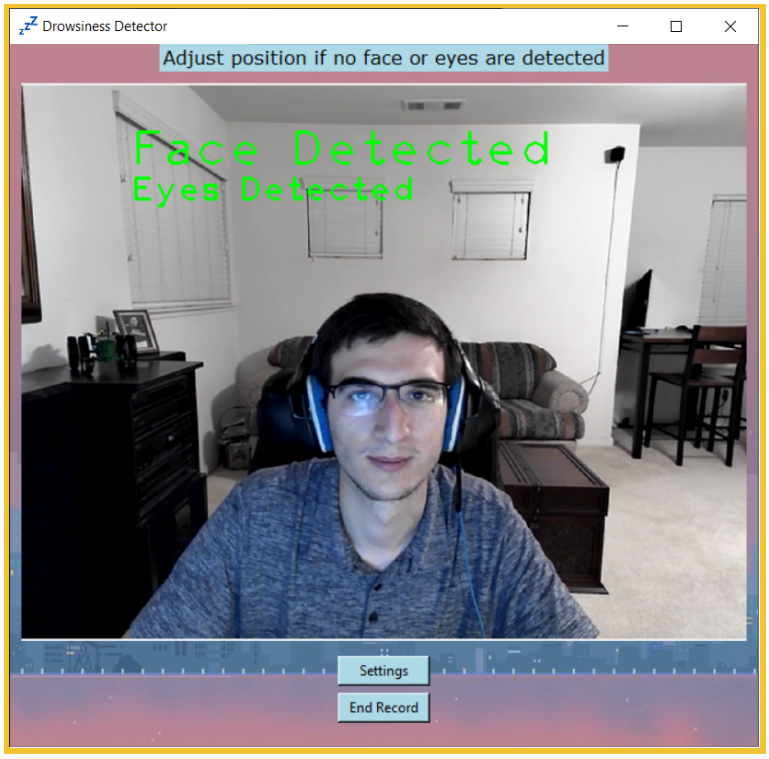
When the user first opens the application, they are shown the Main Window. This is the starting page of the application where they are welcomed to the Drowsiness Detector application. From here, users can choose to click the 'Run' or 'End Program' button. By clicking the 'Run' button the application will begin running and take the user to the Preview Page. By clicking the 'End Program' button the application will terminate after a given number of seconds. To also terminate the application, the user can simply click on the 'X' icon located at the top right corner and end the application abruptly.

Preview Window



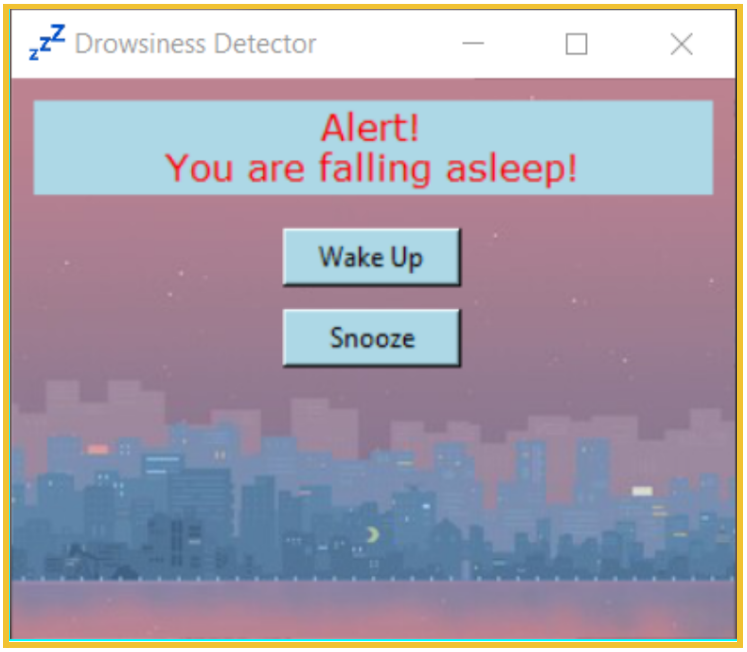
After clicking the 'Run' button on the Main page, the user is taken to the Preview Window. On this window, the user is able to preview what they look like with their camera and whether the application is able to properly detect their eyes and face. If the user's eyes and face are in view, a green notice message will state "Face Detected" or "Eyes Detected" depending on detection. A box around the detection location will also appear notifying the user that their face or eyes are in view. From here, the user can either click 'Settings' to configure their blink detection thresholds or click 'Continue' to go to the recording page.

Recording Window



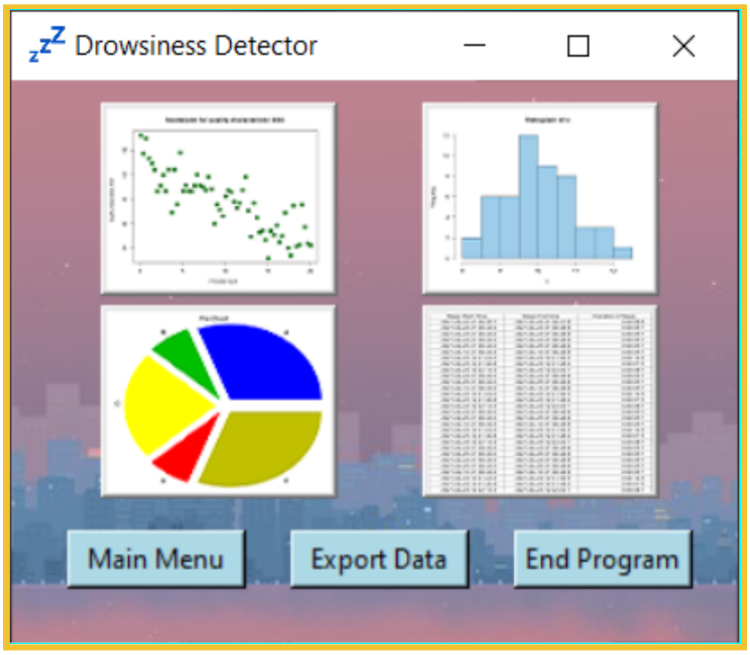
Following the Preview Window, the user is taken to the Recording Window where, by default, the application will begin recording eye motion. The user's eyes and face should always be detected unless they blink or fall asleep. If the user's blink lasts longer than the blink threshold, a datapoint is added to the data csv file, which will later be used to display the graphs in the Results Window. If the user's eyes remain closed for longer than the alert threshold, they will be taken to the Alert Window as they are thought to have fallen asleep. To access the Settings page to change the default timers, the user can click the 'Settings' button on the window. If the user wishes to end recording they can click 'End Record' to be taken to the Results Window.

Alert Window



When the user has their eyes closed, or the application cannot detect their eyes or face for a prolonged period of time, the Alert Window appears. The Alert Window will set off a sound alarm notifying the user to wake up. The user then has the option to click the 'Wake Up' button or 'Snooze' button. The 'Wake Up' button will take the user back to the recording session and resume the previous recording. The 'Snooze' button will stop the sound alarm for a set amount of time and then begin again after that allocated amount of time is over.

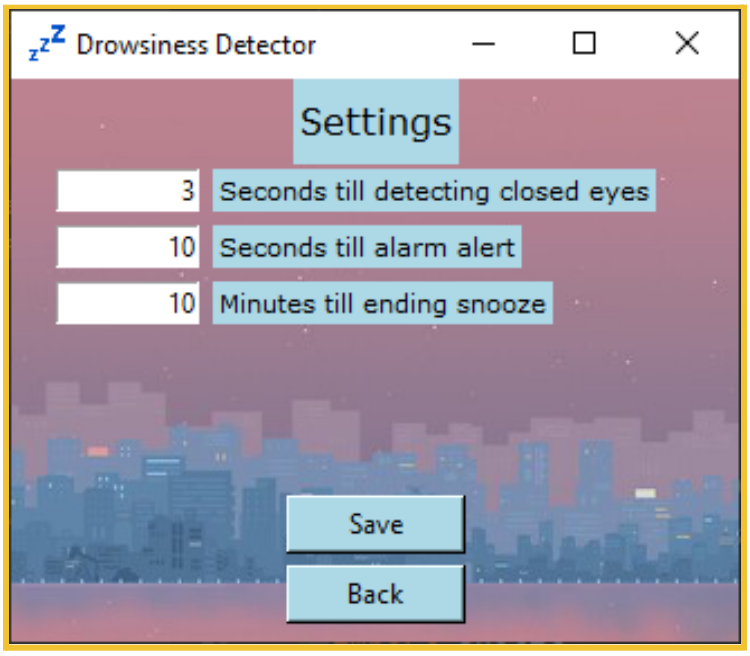
Results Window



When the user clicks the 'End Recording' button on the Recording Window, they are taken to the Results Window to see the collected data from that session. There are four types of data visualisations available to the user which can be opened by clicking their thumbnails. These graphs include a scatter plot for time of blink and duration, a histogram showing how many blinks occurred during each hour of recording, a pie chart showing the blink duration frequency, and a sleep table showing the times and durations the user fell asleep for.

The 'Main Menu' button will send the user back to the Main Menu to start another recording session. The 'Export Data' button copies the current data file into an exports folder using the current timestamp as the filename. Finally, the 'End Program' button will close all windows and terminate the program.

Settings Page



Clicking the 'Settings' button will send the user to the Settings Window. On the Settings Window, the user can set the amount of time (seconds) required for a blink to generate a datapoint, the amount of time (seconds) required for a blink or undetected-face to generate an alert, and the duration (minutes) of the 'snooze' button. The user can click the 'Save' button which will write the data to the settings file and return them to their prior page, or click the 'Back' button which simply discards the new settings and returns the user to the prior page.

Glossary of Terms

GUI (Graphical user interface): Interactive visual components utilized to create user interfaces in computer software.

RAM (Random access memory): A form of short-term computer memory that can be read or written in.

CPU (Central processing unit): A processor that performs logic, arithmetic, and controlling operations for computer programs.

MB (Megabyte): A unit of memory

GHz (Gigahertz): A unit of frequency

CSV (Comma Separated Value): A text file in which the data is delimited (seperated) by commas.

JSON (JavaScript Object Notation): An open standard file format, and data interchange format, that uses human-readable text to store and transmit data objects consisting of attribute–value pairs and array data types.

References

About OpenCV. (2020, November 04). Retrieved February 08, 2021, from https://opencv.org/about/

Introducing JSON. (n.d.). Retrieved April 30, 2021, from https://www.json.org/json-en.html

Johnston, L. (2020, June 30). What should i know before buying a webcam? Retrieved February 08, 2021, from https://www.lifewire.com/before-you-buy-a-webcam-2640480#:~:text=The%20lower%20the%20resolution%2C%20the,more%20common%20and%20more%20affordable

The Front End for ML and Data Science Models. (n.d.). Retrieved February 08, 2021, from https://plotly.com/

Tkinter - Python Interface to tcl/tk. (n.d.). Retrieved February 08, 2021, from https://docs.python.org/3/library/tkinter.html