DROWSINESS DETECTOR

PRD v2

//TODO: TEAM NAME

Danielle Robinson	daniellerobinson@ucsb.edu	*Lead*
Victoria Sneddon	vsneddon@ucsb.edu	*Scribe*
Andrew Polk	polk@ucsb.edu	*Developer*
David Sun	davidsun@ucsb.edu	*Developer*
Brandon Tran	brandonrtran@ucsb.edu	*Developer*

INTRODUCTION

Problem
Aerospace provides technical guidance for all aspects of space systems. Current satellite launches require a team of trained professionals to be alert at all times of the day. However, full 24 hour attentiveness cannot be reasonably expected by human beings. Thus, in order to help Aerospace solve complex space-related science and engineering problems, we strive to solve the issue of drowsiness using facial recognition technology.

Background
Aerospace's technical guidance has workers in situations where they need to be alert and respond quickly to changing circumstances. These can be high stakes situations where being drowsy or not paying enough attention could result in something going wrong. We want to provide an automated way for detecting and notifying if employees are drowsy or otherwise distracted. Our solution to tiredness detection would not only create a safer, more productive work environment, but it would also allow for more successful launch of satellites.

Science
Many drowsiness detectors exist in automobile software, especially in cars. Because of that, we can imitate their solutions with our own implementations. The OpenCV framework, a real-time open source computer vision library, allows us to use facial landmarks to determine if the person is drowsy. We can use it to calculate things, such as the duration for which an eye is closed, and then if it crosses a threshold, respond with an audible alarm. One automobile company has a drowsiness detection solution that utilizes an infrared camera above the steering wheel, detecting more complex signs of tiredness such as frequent blinking, deviations in steering, and distracted head movements. Although not all of these are implementable, we plan on using similar ideas in our own solutions.

Assumptions
For our project we are going to be using a web camera to do facial analysis on the Aerospace employee. This requires not only an active employee, but also for them to have the expected facial features (2 eyes, nose, mouth) to do analysis on.

Innovation
Our project is innovative in the way that we detect whether or not a face displays signs of drowsiness. We not only use facial recognition, we examine each part of the face to determine if it displays signs of sleepiness. Face examination includes looking for different signs of drowsiness like yawns and

changes in blink speed and frequency. The facial recognition portion also utilizes OpenCV.

Project Specifics
We plan on using Python for our coding language along with a C++ library called dlib for image processing. The hardware we will use is a webcam. The framework we will use is OpenCV. Our version control is both Git and GitHub. For project management, we will have Trello and Google Drive to communicate and stay updated on each other's progress. Finally, we will use Slack in order to communicate informally between group members.

Team Goals
Our plan is to split our project into 4 sprints. During the first sprint, we plan on creating a foundation for the design/planning of the project. We will work on the proof of concept for the tools we're going to use as well as determine how we will analyze whether someone is drowsy or not. Furthermore, we will determine how we want to process images, install the framework and libraries for backend processing, and create an established workflow.

For the second sprint, we will begin by working on the basic framework of the project as well as the image processing for data inputs. We will also have continuous integration with GitHub and finally, we will continue to test different ways of determining drowsiness.

Our third sprint will consist of us integrating individual components of the code as well as finalizing the best way to process images and determine drowsiness.

For our final fourth sprint, we will work on a presentation that displays what we've accomplished so far. We will create a demo of our current project and create a plan for what we will be adding next quarter.

Stretch Goals
Our team would like to create individual user profiles. At the start of the program, the user will be prompted to either log in or create a new account. This information will then be stored in a database. With the creation of each new user profile, we will perform a calibration step at the beginning of their first session. This will analyze their blinking rate, frequency, size of open eyes and size of their yawn. This will then allow for the threshold values to be specific to a user for detecting drowsiness.

For each user we will then calculate statistics such as how we are detecting they are tired (yawns, blink patterns, eyes being closed for prolonged periods of time). These statistics will be stored in our database and then uploaded to a website where a user can log into their account and view their

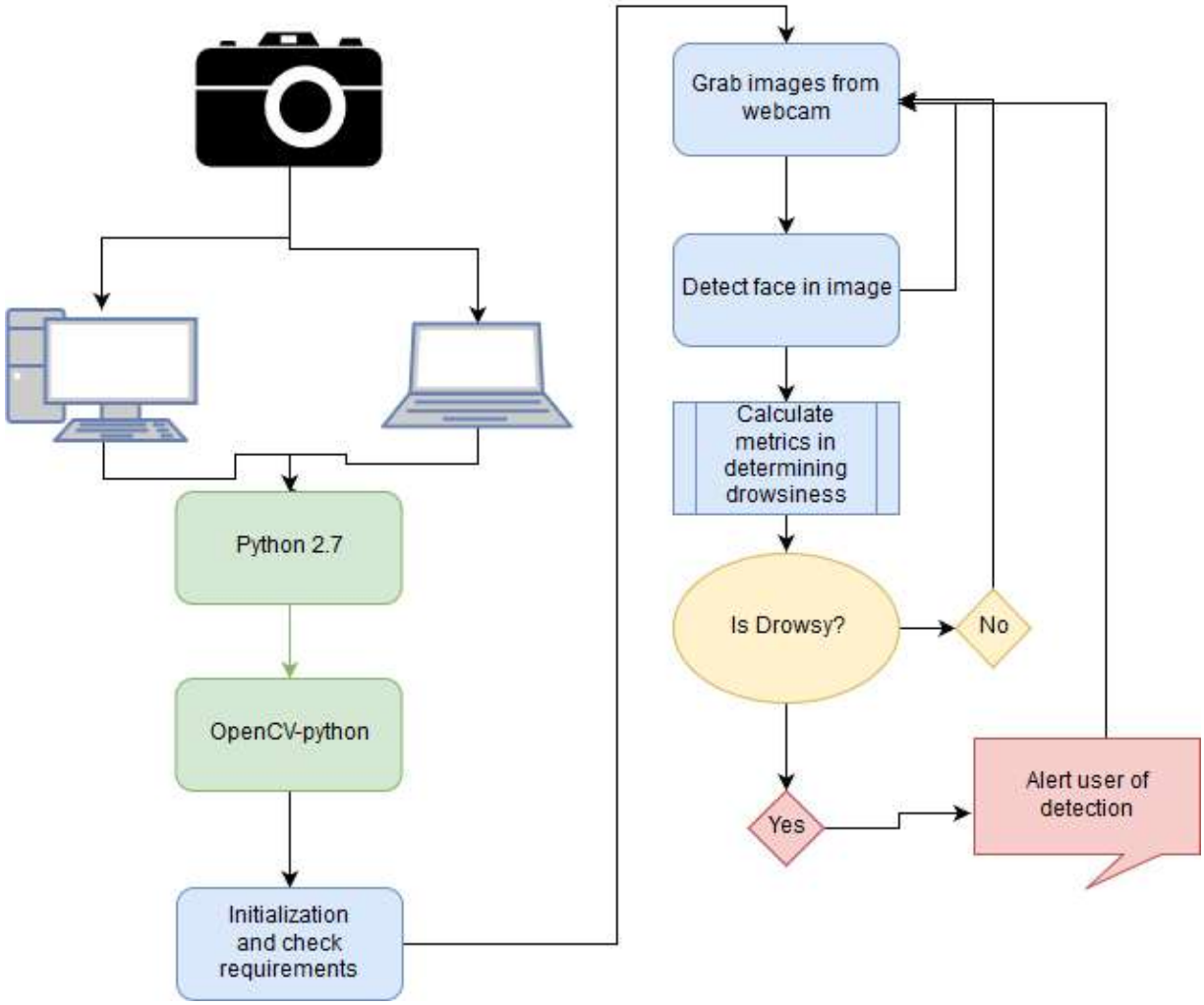
statistics.

Another stretch goal would be to create a mobile android drowsiness detection app. One of challenges here would be with the phone's processing power because we have to take up to 30 frames per second to analyze blink frequency. We would have to see what we could do with constrained resources.

Finally, we could create an embedded solution with an ARM + GPU provided by Aerospace.

SYSTEM ARCHITECTURE

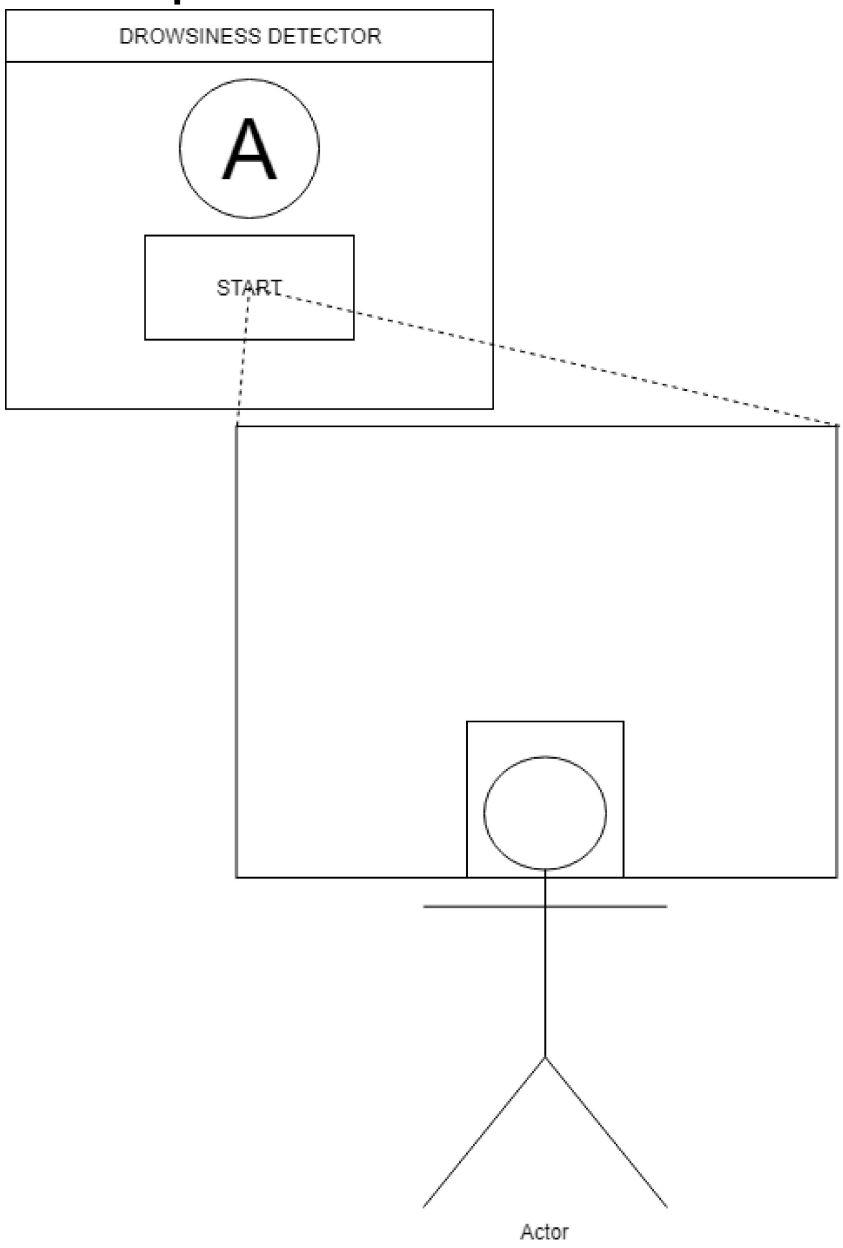
High Level Diagram



User Interaction and Design

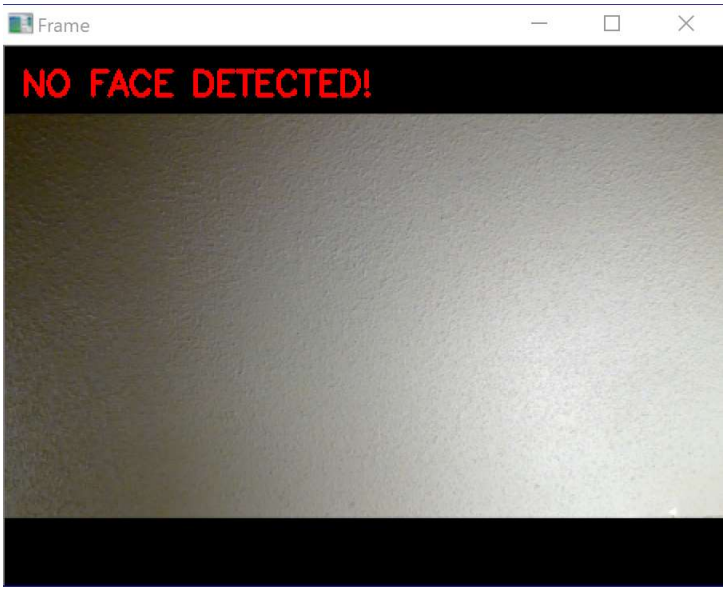
A user will run the executable program which will then check to make sure correct version of Python and OpenCV are installed. There will be a button to press start. Once start is pressed, the program will start all functionality and detect whether the user is drowsy. The button can then be pressed to stop the webcam functionality and then pressed again to resume.

UI Mockup

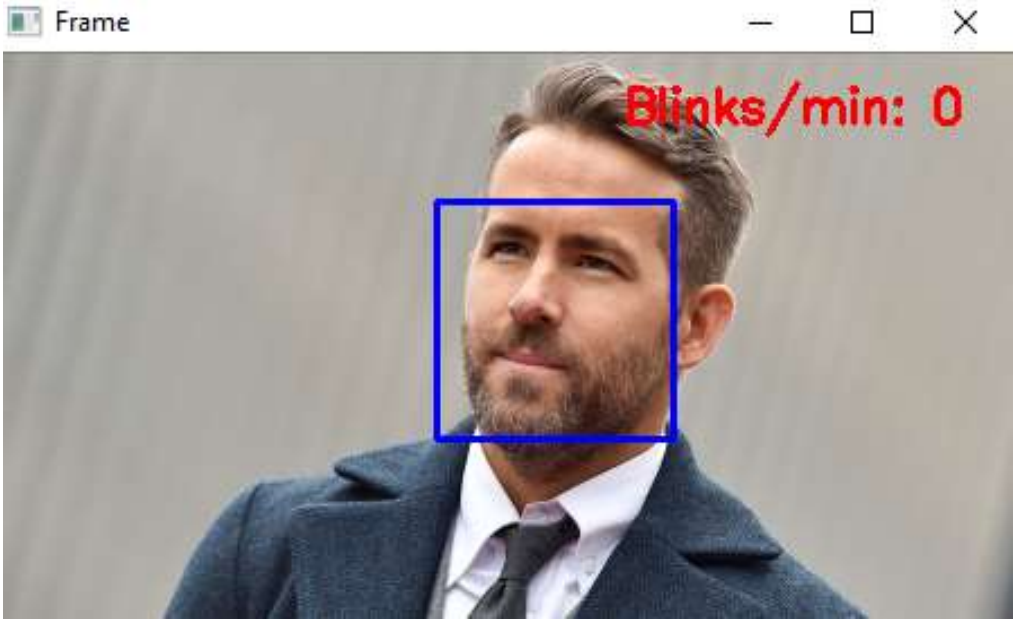


Current UI

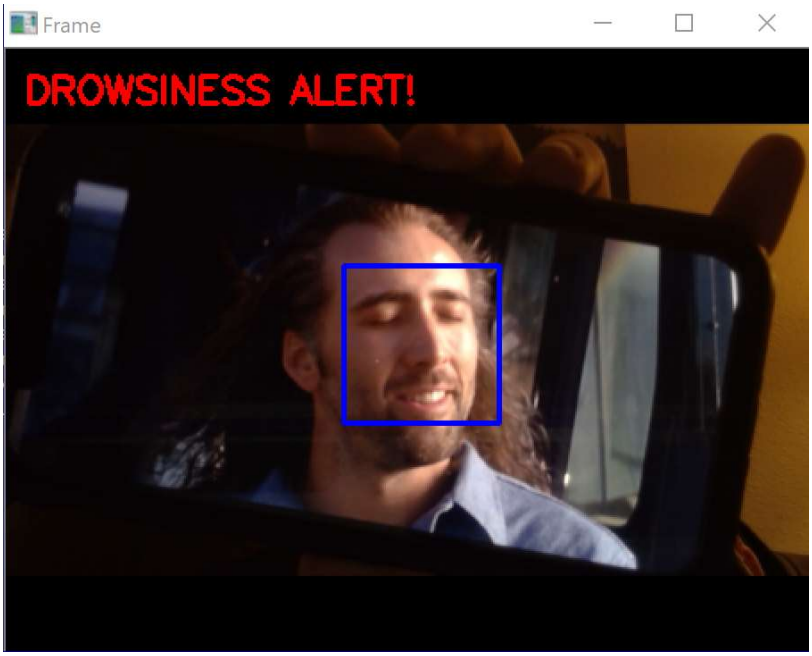
This is the current button which starts/stops the webcam once you start the program.



If the drowsiness detector does not recognize a face in the webcam, then it will alert the user that no face is detected.



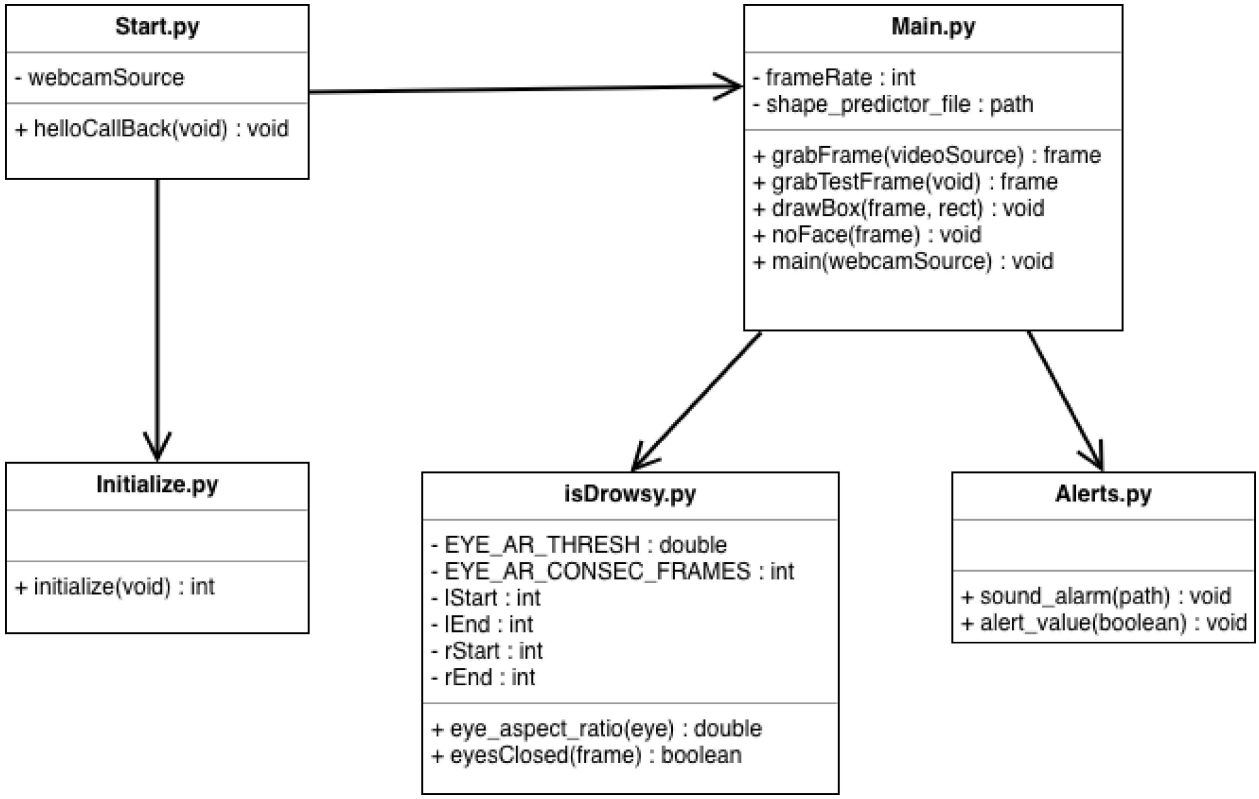
If the drowsiness detector recognizes a face in the webcam, then it will draw a box around the face of the user.



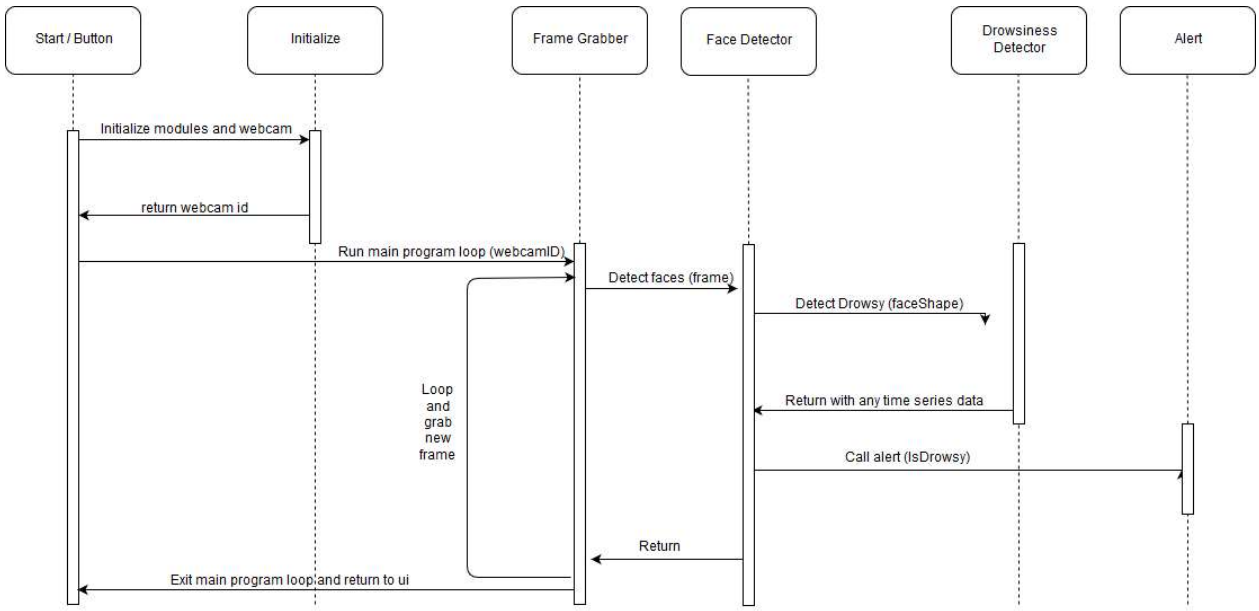
If the drowsiness detector detects a face and the eyes of the user have been closed for more than a few seconds, the user is alerted with a message they are drowsy and an alarm will sound for a few seconds.

SYSTEM MODELS

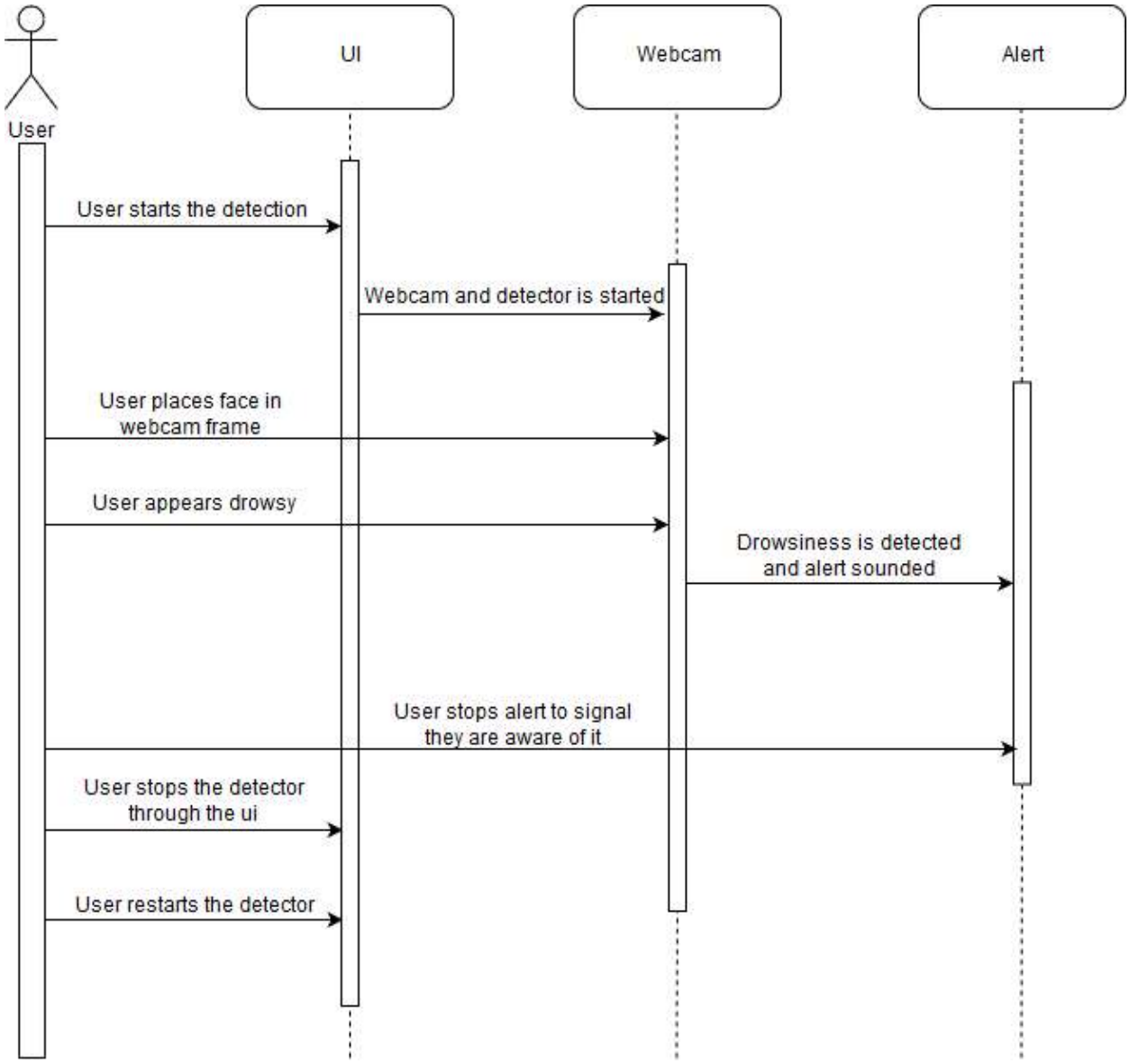
UML Diagram



Module Sequence Diagram



User Sequence Diagram



REQUIREMENTS (USER STORIES)

1. As a user, I can try to open and run the executable program without a webcam.

   Acceptance Test: The program will then deliver an error because it tests for a camera upon running.

   Git Commit *Trello Card*

2. As a user I try to open and run the executable program without Python 2.7 and/or OpenCV installed.

   Acceptance Test: Program will deliver an error saying I must download Python and/or OpenCV in order to run the program.

   Git Commit *Trello Card*

3. As a user I run the program to detect my face.

   A*cceptance Test:* I will know if my face is detected because there is a box drawn around it.

   Git Commit *Trello Card*

4. As a user I can start running the program without my face in the frame.

   Acceptance Test: The program will continue to try and detect a face and alert the user no face is detected.

   Git Commit *Trello Card*

5. As a drowsiness detector, I determine if the eyes are closed through analysis of the 6 eye landmark points by calculating the eye aspect ratio.

   Acceptance Test: If the eye is closed for a predefined number of seconds, an alert will be triggered.

   Git Commit *Trello Card*

6. As a user, my eyes are closed which is an indicator of drowsiness.

Acceptance Test: The user will close their eyes and the user is alerted that they are sleeping.

Git Commit *Trello Card*

7. As a user, I am alerted that I am drowsy and click the alert to let it know that I have been notified and the program will continue to run in the background.

   Acceptance Test: A button pops up when the user is drowsy and will continue alerting the user until the resume button has been pressed.

   Trello Card

8. As a user I can interact with the program's Python GUI window to start it. To do this, I press a start button which will then run the start.py module.

   Acceptance Test: Python GUI launches with button displayed when program is called.

   Git Commit *Trello Card*

9. As the drowsiness detector, I will detect only one face.

   Acceptance Test: If there are multiple faces detected then alert the user that the program only functions with one face.

   Git Commit *Trello Card*

10. As a drowsiness detector, a users yawn will be detected and then counted.

    Acceptance Test: Print the number of yawns detected in a sequence.

    Git Commit *Trello Card*

11. As a user, my yawns are detected as an indicator of drowsiness.

    Acceptance Test: The user will yawn multiple times in a sequence and the user is alerted that they are drowsy.

    Git Commit *Trello Card*

12. As a user if I blink, the program will update a running count of

blinks on its graphical interface.

Acceptance Test: Show a blinking person and watch the counter increase.

Git Commit *Trello Card*

13. As a user, my frequency of blinking is detected as an indicator of drowsiness.

Acceptance Test: The user will blink slowly and frequently and the user is alerted that they are drowsy.

Git Commit *Trello Card*

14. As the drowsiness detector, I will perform a calibration on the user's face to adjust drowsiness analysis to be specific for the user.

Acceptance Test: The threshold values for eye aspect ratio, mouth aspect ratio and blink frequency will change.

Trello Card

15. As a user I can create a username/account or log in to a previous account upon starting the program.

Acceptance Test: The user can enter a username and it will be stored in memory.

Trello Card

16. As a the drowsiness detector, I can calculate statistics about a user's drowsiness.

Acceptance Test: While the program is running for a specific user, information about how often they are drowsy and how it is being recognized are computed.

Trello Card

17. As a user, I can check my drowsiness statistics on a website by logging in to my account.

Acceptance Test: The website allows a user to log in and check their drowsiness statistics.

Trello Card

18. As the drowsiness detector I can send all user statistics to a website after a session ends.

    Acceptance Test: The statistics calculated are correctly sent and displayed on a website after the program is closed.

    Trello Card

19. As a database, I can store all user accounts and associated statistics.

    Acceptance Test: When a new user is created or a previous user has new activity, all new information will be sent and stored in the database with any previous information.

    Trello Card

20. As a user, the program can detect my face in extreme lighting conditions.

    Acceptance Test: User's face is recognized in a dark or bright room.

    Trello Card

APPENDICES

OpenCV
Windows/Mac OS
Webcam/Camera
Python (2.7)