SlEEGp: Sleep Tracking Using Electroencephalography

CM10251
Group 11
16 April 2019

Group Member	**Username**	**Degree Course**
Mathew Allington	mma82	MComp Computer Science (yr. 1)
Tom Draper	td544	MComp Computer Science (yr. 1)
Aethan Foot	ajf75	BSc Computer Science (yr. 1)
Alexander Ito-Low	ail24	BSc Computer Science (yr. 1)
Christophe Millischer	cm2307	BSc Computer Science with Business (yr. 1)
Soren Mortensen	snm48	BSc Computer Science (yr. 1)
Samuel Sogbesan	ss3222	BSc Computer Science (yr. 1)
Ravit Songthammakul	rs2347	BSc Computer Science (yr. 1)

SlEEGp: Sleep Tracking Using Electroencephalography

Mathew Allington, Tom Draper, Aethan Foot,
Alexander Ito-Low, Christophe Millischer, Soren Mortensen,
Samuel Sogbesan, Ravit Songthammakul

Abstract

The main aim of our project is to create an interface in which the user can actively evaluate, measure and improve their sleep habits. With the use of an EEG headset, we took raw brain-wave data and algorithmically determined the sleep state of the user throughout the course of their sleep. We then extracted an overall percentage of time that the user was actually awake over the course of their sleep. Using this data we motivated the user to improve the quality and length of their sleep with the use of a level system. Our final interface consisted of four main screens. The first screen allows the user to capture EEG data with the use of a capture screen which facilitates the capture and storage of the data streamed from the headset. The view screen permits the user to playback said data in the form of a graph, allowing for media player style controls (play, buffer, speed etc). The analyse screen presents classified data to the user, which the user could use to manage, view and update sleep records, as well as manage their goals. I an effort to allow the user to track their progress, The fourth screen was simply a statistics screen which displayed to the user their total experience points and level.

Supervised by:

Hyde, Jo
cssjkh@bath.ac.uk

Contents

1 Introduction

1.1 Overview of Domain

Personal informatics is a term used to refer to devices and software that help people gather information about themselves, so they can reflect upon it and gain motivation to make changes to their lifestyle and habits to improve their overall wellbeing. Personal informatics is used for effectively motivating people to gain self-knowledge, change behaviours and build habits (Rapp and Cena, 2016).

The area of personal informatics has started to expand in popularity in recent years mainly due to the increased availability and usability of affordable hardware. Consumer products such as the FitBit and Apple Watch allow users to collect data on a wide variety of metrics including heart rate, blood pressure, motion and many others. Products such as the Neuroon, a wearable electroencephalograph (EEG) eye mask, and Zeo Sleep Manager Pro, an EEG headband, allow the user to collect information on brainwaves for the purpose of sleep tracking (Liang and Chapa Martell, 2018).

Another factor that has contributed to the growth of personal informatics is the ubiquity of smartphones, meaning that users have an ever-present device that allows them to collect and collate data from their personal informatics hardware. Many personal informatics apps also add an element of socially driven competition and gamification, driving users' motivation to continue to use them and push their friends to also begin using this technology.

1.2 Challenges

Although personal informatics systems for wellbeing has been on the rise, it inherently presents flaws that need to be taken into account. In a survey conducted by Rapp and Cena (2014) where participants were regular personal informatics users, it was revealed that the most significant shortcomings of personal informatics systems supplied by commercial companies was a lack of understanding for the end user's requirements and an absence of assistance and alerts for users who didn't meet their goals. Apart from users who are familiar with personal informatics systems it is also important to consider the challenges faced by the common user; a user who is new to using a personal informatics device. It was discovered by Rapp and Cena (2014) that the main challenge for personal informatics systems was the lack of motivation faced by the end user to continue to use the system. These challenges need to be taken into account because these hinder the end user from improving themselves which is contradictory to the goal of personal informatics systems.

1.2.1 Health Risks

One crucial problem in the realm of health is sleep deprivation. Sleep deprivation is defined by the British Medical Association as "a lack of sufficient sleep resulting from disruption to the natural sleep cycle" (2018). This is important to highlight because as as opposed to fatigue, sleep deprivation isn't subjective. In accordance to Alhola and Polo-Kantola (2007), it was estimated that the main effect of sleep deprivation was the reduction in cognitive performance. This includes: impaired attention; longer delays in making decisions; poor quality of decisions and a reduction in long memory. This is especially important to monitor for individuals who have high risk jobs. In 2010, 158 passengers' lives were lost when an Air India Express plane overshot a runway by 600 metres. A leaked government report was stated to have found that the accident unfolded due to the pilot's severe sleep deprivation (British Broadcasting Corporation, 2010). Even in circumstances where the individual isn't responsible for people's lives, a reduction in cognitive performance is still observed. Hence, the validity of this problem is justified.

Sleep deprivation has the potential to be a dangerous to anyone and even fatal, if it is not identified and managed. Over a third of adults get less than seven hours of sleep during

a typical 24-hour period. Lack of sleep affects a person's abilities and health in many ways and can limit their ability to perform in their line of work. This can be seriously, especially for professions that the general public's lives rely on greatly such as doctors and nurses. Each year, 100,000 deaths occur in US hospitals due to medical errors and sleep deprivation in the medical staff has been shown to contribute to this statistic (American Sleep Association, 2019). Driving whilst feeling drowsy has been shown to be similar to driving whilst under the influence of alcohol and can be attributed to a portion of car collisions each year (National Sleep Foundation, 2019). There are few visible indications that an individual is getting less sleep than recommended and it's difficult to know if you are deprived of sleep which often makes it more damaging as it may be challenging to identify and then attempt to curb. Similarly, there's no legal limit for sleep deprivation for situations such as driving a vehicle. Sleep deprivation is difficult to accurately diagnose without the aid of external equipment.

1.2.2 Solution

Our software system aims to reduce fatigue in individuals due to a lack of sleep. EEGs have been shown to detect sleeping problems based on brainwave patterns (Mayfield Certified Health, 2016). Monitoring and recording the individual user's brainwave data during their sleep allows the system to be tailored specifically for that individual. If the system were to be used most nights, the headset would be able to measure and record a large quantity of reliable data for the system to process making any results the system provides more likely to be accurate and therefore helpful for the user. If any recorded brainwave data were to be displayed in a graph format, this may help the user to better understand their recorded sleep data and allow them to spot trends and make more appropriate decisions as to how to change their actions.

2 Agile Software Process Planning and Management

2.1 Risk Management

2.1.1 Sprint Setbacks

Setbacks had the potential to occur at any time during the project. Using a Scrum software development methodology meant that the group had regular sprint deadlines up until the final project deadline. A delay in task completion or other setback would have meant we were unable to finish a sprint at the intended time, resulting in future deadlines also being pushed back. This was likely to happen at some point in the project due to the potential of group members falling ill, new tasks needing to be prioritised over older ones, and the effort required for tasks having been underestimated due to not being fully understood. Only two opportunities to sign off a sprint were available each week, which forced the group to be organised, as each failed sprint would have resulted in a significant delay.

When a setback occurred in the first sprint, the group agreed to extend the sprint, given the still substantial amount of time left to complete the project. Had this been near the end of the project, or had the group been required to complete more than three sprints during the project, this level of flexibility may not have been an option.

The first sprint was initially intended to be a week long, but after two failed sprint reviews, it took three weeks to complete. After a group discussion, this was put down to a lack of organisation and communication between members, and it was noted that the reasons for each sprint failure could have been easily avoided. For example, the second failed review was due to a mismatch between the requirements and implementation.

From that point forward, the group aimed to improve communication by increasing the number of meetings per week and ensuring members contacted the group if they were unable to attend. An easily accessible meeting schedule was also set up for the group. Tasks would be assigned to pairs rather than individuals going forward, to eliminate single points of failure and reduce the likelihood of a high priority task remaining incomplete before the next sprint deadline.

- Identified issues during 1st of March (meeting minutes 10).
- Discussed issued and refactoring of requirements during meeting 8th of March (meeting minutes 11).

2.1.2 First Sprint - Preparation

At the beginning of a project there is a large amount of group preparation that must be done. As this coursework is the first piece of work we have completed on the course that involves an agile approach to software development, a slow start to the project is likely. The group would need to get organised and any software that the group would need in future to complete the project would need to be installed on all or select individual's machines.

After completing the first sprint, our group may have underestimated the amount of time this takes up and this is likely to have contributed to sprint 1 setbacks. Before we began the started, working with scrum software development was a new concept and many had no experience with sprint structure. and later realised we had.

After the group passed sprint one, this was removed from the risks list as it was no longer going to be a potential issue.

2.1.3 Infection Risk (Mumps)

During our project, there was an outbreak of mumps on campus. This meant that there was a real possibility one of our members could catch the disease, albeit fairly low. If this were to happen, the member would be unable to work for a significant period of the project,

leaving more work for the other members of the group. If it were to happen, or if a member was generally absent for a extended period of time, we agreed to split that members work between the remaining members.

- Discussed during tutorial meeting 22nd of March (meeting minutes 18).

2.1.4 Unexpected Absences

Similarly to the mumps outbreak, a group member may become ill or have unforeseen circumstances that would result in them not being able to attend the scrum meetings.

2.1.5 Time management (Other Deadlines)

As this project runs across an eleven week period, there will be multiple large courseworks set and completed in other modules during this timespan. If members of the team do not time manage appropriately, some members may lower their priority for this project and shift focus to other courseworks, especially during the days before a deadline. To combat this, members may have create a time management strategy for themselves, or contribute more on this project during periods that they are more available.

- Discussed during meeting 21st of March (meeting minutes 17)

2.1.6 Group Member Leaving Group

This project takes place during the first year of our degree therefore students may make a decision to leave the course. This occurred with our group towards the middle of the project, meaning we had one fewer member to work on the sprint tasks from week to week. After discussion agreed to split the departing member's future tasks between the remaining seven of us. As we had only lost one member out of eight and the member had made frequency contributions before his application to leave, we felt this would be manageable for the few remaining weeks we had left working on the project. Having one fewer member than we were previously used to was something the group had to keep in mind moving forward with the project. It would also increase the possibility of failing the current sprint which would cause further issues, as discussed in 'Sprint Setbacks'.

We feel as we have solved this issues by managing the work allocation if it occurred and as we moved further to the final deadline a, as a result this was later removed from the list of risks.

- Discussed during tutorial meeting 22nd of March (meeting minutes 18).

2.2 Evidence of Agile Approach

2.2.1 Sprint Structure

Our schedule was segmented into separate sprints. To initiate a sprint we first start by holding a sprint planning meeting. In this meeting we discuss which epics (features) we would like to take from the project backlog to develop. We would then break this down into approachable sub-tasks which would then be placed onto the project backlog. The group would then be given chance to express any concerns or comments on a particular task so that they could be resolved.
In order to estimate work load, we would use a mechanism called "Scrum Poker". A process in which each group member would use an Application on their phone to estimate a value for that task, each member would chose a value on their phone and the whole group would reveal their value at the same time. This value signifies the workload of that particular task. As a result, we had a metric which we could actively measure to evaluate work-rate of the team. After this, group members are able to assign themselves to tasks by associating

themselves with a particular requirement, then place it on the sprint tasks list.

2.2.2 Pair Programming

We found that it was often effective to pair people together on one programming task. This useful because pairs would be able to better solve problems and overcome setbacks more effectively. Where a member had a strength which the other member did not share, the member would be able to compensate for it. For example, if member A excelled more in Maths and member B excelled more in the programming, then member A would be able to concentrate on classification algorithms and member B would know the best way to implement it. So in this analogy, member A would act as a "driver", then member B would act as a "navigator" essentially implementing what Member A told it to. Both members would be actively engaging in the task, which means potential mistakes could be more easily spotted. The pair members would then actively swap the roles through out, decreasing stagnation on the task. We realised that this relied on at least one team member having a satisfactory level of programming experience.

2.2.3 Stand-up Meetings

Any problems could be addressed through the use of regular standup meetings. In these meetings group members could discuss any setbacks they are experiencing, then our group was able to resolve any problems in that meeting. This would sometimes means adding an extra (more experienced) group member to that particular task or either discussing alternative solutions or methods. Of course this would sometimes mean modifying certain requirements to better fit the program overall.

2.2.4 Test-Driven Development

Our whole sprint was motivated by achieving a set of outcomes which would be tested at the end of the sprint. Each test would determine an outcome which the developer would have to achieve. This decreased the chance of developing defects in the program, because each task would be ensured to produce the exact result required. Testing would take place at the end of the sprint and would be signed off by the scrum master (our tutor). They would be presented a requirements backlog which contains all of the features that have been completed in that sprint. In a separate document, a lists of tests corresponding with each requirement is defined. Each test ensures the completion of that requirement by defining an expected outcome which the program should produce.

2.3 Throughput Estimation (Burn-down Charts)

As already discussed, the metric which we used to estimate workload was task estimation. From this we were able to produce burn down graphs which would we could correlate and interpolate an estimated time of completion. We could use it to actively determine whether we were completing tasks at a fast enough rate.

2.4 Management of Requirement Development

There were many processes which enable us to actively develop the requirements as we went along. One mechanism would be the use of HLA diagrams. These diagram helped us construct the fundamental processes which would drive the core functions of the program.

The construction of these processes then produced a set of requirements, which could be further update while these diagrams were being further developed. Requirements were also derived from the discussions made in the group meetings, members would discover key dependencies derived from initial requirements which would becomes requirements in their own right. As well, as this requirements would be created when programming in order to meet the intended result of other functions. Another source of requirements was the data gathered from our surveys, eg. the amount of time a user would like view data over.

3 Software requirements specification

3.1 Problem domain and purpose of our system

As mentioned earlier sleep deprivation is a prevalent problem that society faces. When constructing the application we wanted it to be accessible to the general public as it was identified earlier that sleep deprivation has the potential to affect everyone. It was identified that the main functionality of our system would be to aid users to track and compare their sleep through the Neuromind headset.

3.2 Stakeholders involved and the elicitation process:

In order to elicit requirements, a questionnaire was constructed. Majority of the stakeholders involved were flight attendants from United Airlines. In order to get a more diverse range of views to accommodate for diversity, an interview was conducted on a university student with a part time job. Responses from participants validated our specific problem by justification both in the questionnaire and in the interviews. Through getting views from more stakeholders this enabled our product to be more inclusive towards the end user. Through being more open to responses from a broader range of user, general ethical principals; especially artifact 1.4 from the ACM code of ethics (being fair and not discriminating) has been addressed (Computing Machinery, 2018).

3.3 UseCase Diagram:

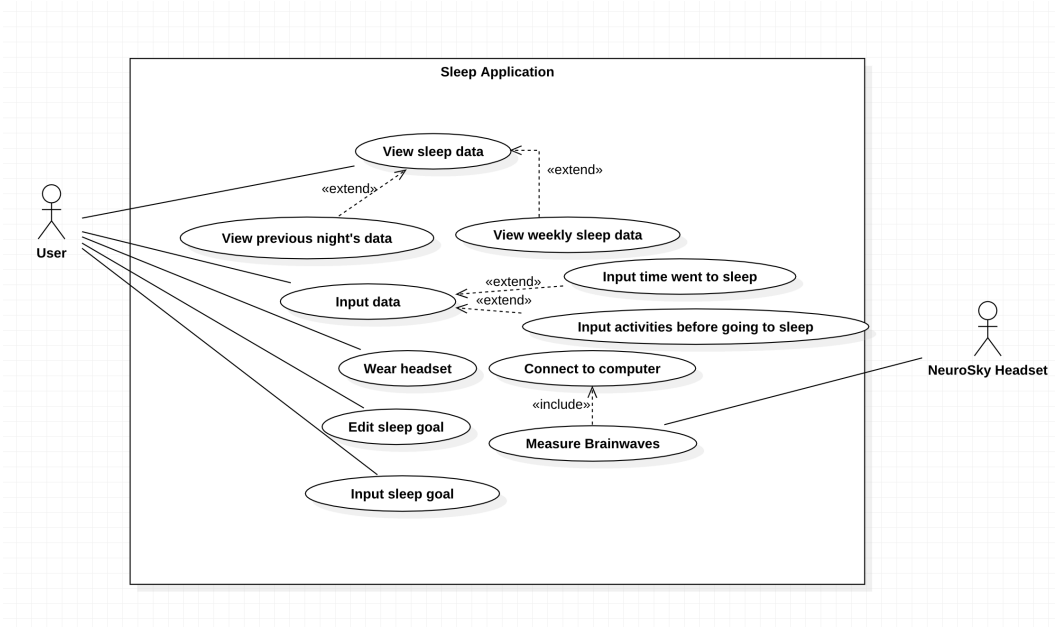


Figure 1: Use Case Diagram

Semantics: The headset is the external hardware which the user has to wear in order to get a recording. Once the user is wearing the headset, it can connect to the computer and then measure the user's brainwaves. The user is the general audience who will interact with our system. The user can view sleep data they have inputted into the system using the headset. They can view either their previous night's data or last weeks data. They can also set personal sleep goals which will allow them to then manage their goals that will be

saved in the system. They can also input the time they have previously went to sleep on nights they did not wear the headset.

3.4 Scenarios:

3.4.1 Scenario 1:

Scenario 1: Actor: User of the system Goal: Measuring sleep data Pre conditions: 1. The user has neurosky headset 2. The user has thinkgear socket connector installed on their laptop 3. The user has our application installed.

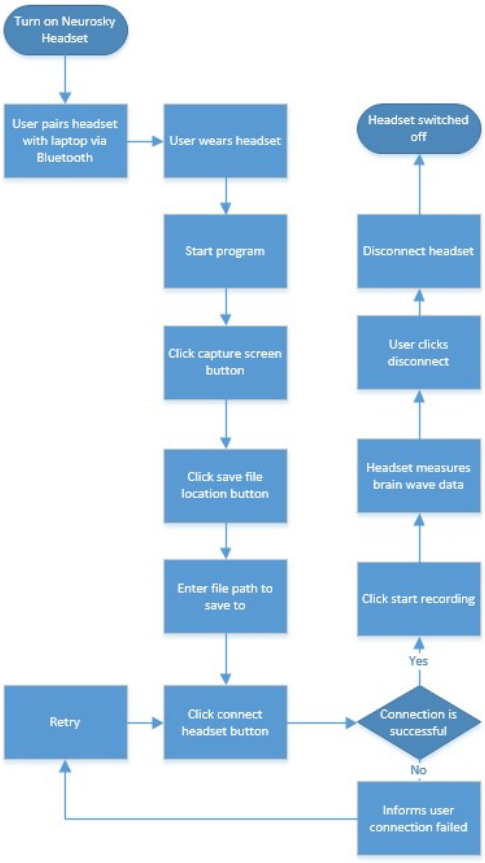


Figure 2: Scenario 1

3.4.2 Scenario 2:

Scenario 2: Actor: The user of the system Goal: View line graph of brain activity throughout the time the headset was worn. Preconditions: The user has the application installed. Scenario 1 has been completed at least once. The user has already opened the application.

Figure 3: Scenario 2

3.5 Non-Functional Requirements

1: Software Development Process		
ID	**Requirement**	**Details**
1.1	The system must adhere to the Agile/Scrum methodology.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
1.2	Software development should include at least 3 sprints.	*Priority:* Medium *Dependencies:* 1.1, 1.3 *Source:* Coursework specification
1.3	Each sprint should last between 1 and 3 weeks.	*Priority:* Medium *Dependencies:* 1.1, 1.2 *Source:* Coursework specification
1.4	Must review the functional requirements at every scrum meeting.	*Priority:* High *Dependencies:* 1.3 *Source:* Coursework specification
1.5	Must incorporate risk management into the software process.	*Priority:* High *Dependencies:* 1.1 *Source:* Coursework specification
2: Expanding Initial Requirements		
ID	**Requirement**	**Details**
2.1	The system must expand upon the initial non-functional requirements.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
2.2	The system must expand upon the initial functional requirements and add additional functionality to the system.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
2.3	Additional requirements should be established by conducting interviews on potential users, reading articles on personal informatics and examining existing personal informatics systems.	*Priority:* High *Dependencies:* 2.1, 2.2 *Source:* Coursework specification
3: Background Research		
ID	**Requirement**	**Details**

3.1	Must read and cite 4 articles in the field of personal informatics.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
3.2	Should read and cite at least eight articles.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification
3.2	Should read and cite at least eight articles.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification
3.2	Should read and cite at least eight articles.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification
3.3	Citations of core article must be peer-reviewed articles.	*Priority:* High *Dependencies:* 3.1 *Source:* Coursework specification
3.4	Additional citations may be made to web articles of unknown quality.	*Priority:* Low *Dependencies:* None *Source:* Coursework specification
3.5	Must classify measurements of stages of sleep (REM cycles).	*Priority:* High *Dependencies:* None *Source:* Tuck (2018), Feinberg et al. (1988), Carlson (2012, p.291)

4: Ethical Issues

ID	Requirement	Details
4.1	Must account for ethical issues in the specification, design, development and testing of the system.	*Priority:* High *Dependencies:* None *Source:* Coursework specification

5: Testing

ID	Requirement	Details
5.1	Must adopt a test-driven development approach with production of test plans.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
5.2	Must provide evidence of testing.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
5.3	Must state hypothesis which clearly link claims to observable behaviours of your system in operation.	*Priority:* High *Dependencies:* None *Source:* Coursework specification
5.4	Must conduct a valid analysis of empirical data generated from an experiment (should include Student's t-test).	*Priority:* High *Dependencies:* None *Source:* Coursework specification

3.6 Functional Requirements

6: Viewing and Collecting Sleep Data		
ID	**Requirement**	**Details**
6.1	The system must be able to interface with the Neurosky headset.	*Priority:* High *Dependencies:* 6.1.1, 6.1.2, 6.1.3 *Source:* Meeting minutes #12
6.1.1	The system must be able to connect to the Neurosky headset.	*Priority:* High *Dependencies:* None *Source:* Meeting minutes #9
6.1.2	The system must be able to disconnect from the Neurosky headset.	*Priority:* High *Dependencies:* 6.1.1 *Source:* Meeting minutes #9
6.1.3	The system must be able to read in raw brainwave data from the Neurosky headset.	*Priority:* High *Dependencies:* 6.1.1 *Source:* Meeting minutes #7
6.2	The system must store raw brainwave data.	*Priority:* High *Dependencies:* 6.2.1, 6.2.2 *Source:* Coursework specification, meeting minutes #7
6.2.1	The system must facilitate the saving of converted brainwave data to a binary file (.ec).	*Priority:* High *Dependencies:* 6.1.3 *Source:* Meeting minutes #12
6.2.2	The system must facilitate the conversion of brainwave data from binary (.ec) to CSV format.	*Priority:* High *Dependencies:* 6.2.1 *Source:* Meeting minutes #12
6.3	The system should be able to extract information about sleep cycles from recorded raw data.	*Priority:* High *Dependencies:* 6.3.1, 6.3.2 *Source:* Meeting minutes #13
6.3.1	The system should extract the attention and meditation data points from the raw brainwave data.	*Priority:* High *Dependencies:* 6.2 *Source:* Meeting minutes #16
6.3.2	The system should calculate a moving average of the attention and meditation values for the whole data set.	*Priority:* High *Dependencies:* 6.3.1 *Source:* Meeting minutes #16
6.3.3	The system must extract the percentage of time during a recording when the user was asleep vs. awake.	*Priority:* High *Dependencies:* 6.3.2 *Source:* Meeting minutes #16
6.3.4	The system must calculate the time slept	*Priority:* High *Dependencies:* 6.3.2 *Source:* Sleep application questionnaire, meeting minutes #16
6.4	The system should be able to present sleep cycle data to the user.	*Priority:* High *Dependencies:* 6.4.1,6.4.2 *Source:* Coursework specification, meeting minutes #7, sleep app questionnaire

6.4.1	The system should be able to present the percentage slept and the time slept in the form of a data table.	*Priority:* High *Dependencies:* 6.3 *Source:* Meeting minutes #17
6.4.2	The system should be able to present processed data in the form of a line graph.	*Priority:* High *Dependencies:* 6.3 *Source:* Sleep app questionnaire
6.5	The system must permit the manual entry of the average time slept for each day over a week.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification, sleep app questionnaire, British Medical Association (2018)
6.6	The system should apply a "known" sorting algorithm which allows users to sort tracking data	*Priority:* Medium *Dependencies:* 6.2 *Source:* Coursework specification
6.7	The system should implement a command-line interface allowing the user to perform all functions of the system.	*Priority:* High *Dependencies:* 6.7.1, 6.7.2, 6.7.3, 6.7.4, 6.7.6 *Source:* Meeting minutes #13
6.7.1	The command-line interface should allow the user to record data from the headset and choose where to store it.	*Priority:* High *Dependencies:* 6.1.3, 6.2.1 *Source:* Meeting minutes #13
6.7.2	The command-line interface should allow the user to specify an amount of time to record for, in minutes or in seconds.	*Priority:* High *Dependencies:* 6.7.1 *Source:* Meeting minutes #13
6.7.3	The command-line interface should allow the user to convert a binary (.ec) file to CSV format, and choose where to save the converted file.	*Priority:* High *Dependencies:* 6.2.2 *Source:* Meeting minutes #13
6.7.4	The command-line interface should allow the user to extract and calculate a moving average of the attention and meditation data.	*Priority:* High *Dependencies:* 6.3.2 *Source:* Meeting minutes #16
6.7.5	The command line interface must allow the user to process sleep cycle data from a raw recorded data file.	*Priority:* High *Dependencies:* 6.3.3 *Source:* Meeting minutes #16
6.7.6	The command line interface must display processed sleep cycle data in the form of a table.	*Priority:* High *Dependencies:* 6.6 *Source:* Meeting minutes #13
6.8	The system should accomodate the use of a Graphical User Interface	*Priority:* High *Dependencies:* 6.8.1, 6.8.2, **??**, **??**, 6.8.3, 6.8.5, 6.8.6, 6.8.7, **??** *Source:* Sleep application questionnaire and meeting minutes #23
6.8.1	The Graphical User Interface should allow the user to connect and disconnect their neurosky headset.	*Priority:* High *Dependencies:* 6.1.1,6.1.2 *Source:* Meeting minutes #23

6.8.2	The Graphical User Interface should allow the user start and stop recording brainwave data from the headset and choose where to save it.	*Priority:* High *Dependencies:* 6.1.3, 6.2.1 *Source:* Meeting minutes #23
6.8.3	The Graphical User Interface should allow the user to convert a binary (.ec) file to CSV format, and choose where to save the converted file.	*Priority:* High *Dependencies:* 6.2.2 *Source:* Meeting minutes #23
6.8.4	The Graphical User Interface should allow the user playback their brainwave data from a selected CSV file	*Priority:* High *Dependencies:* 6.8.2 *Source:* Meeting minutes #23
6.8.5	The Graphical User Interface must allow the user to process raw brainwave data stored in a user file.	*Priority:* High *Dependencies:* None *Source:* Meeting minutes #23
6.8.6	The Graphical User Interface must display the percentage slept and the time slept in the form of a table.	*Priority:* High *Dependencies:* 6.3 *Source:* Meeting minutes #23
6.8.7	The Graphical User Interface must allow the user to view the selected processed brainwave data in the form of a line graph.	*Priority:* High *Dependencies:* 6.4.2 *Source:* Sleep app questionnaire and Meeting Minutes #23

7: Identifying Trends in Data

ID	Requirement	Details
7.1	The system must allow the user to compare sleep data over a week.	*Priority:* High *Dependencies:* 6.2, 6.5 *Source:* British Medical Association (2018), sleep app questionnaire

8: Goals and Achievements

ID	Requirement	Details
8.1	Allow users to manage targets (goals) for time going to bed.	*Priority:* Medium *Dependencies:* 6.2, 6.5 *Source:* Coursework specification, sleep app questionnaire
8.2	Should permit user to set a daily or weekly goal for time going to bed.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification, meeting minutes #7
8.3	Should permit user to update or change goal.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification
8.4	Should motivate user to achieve their goals via a point system.	*Priority:* Medium *Dependencies:* 8.1, 8.2 *Source:* Coursework specification, sleep app questionnaire

9: Manage Data Periodically

ID	Requirement	Details
9.1	Users should be given the option to delete their sleep cycle data	*Priority:* Low *Dependencies:* 6.2, 6.3 *Source:* Meeting minutes #13, sleep app questionnaire

3.7 How conflicts between requirements were managed:

Throughout the course of the project, requirements were refined when group meetings were conducted and observations by the group were made. From the very beginning our group was sure to acknowledge and discuss proposals made by other colleagues in the group ensuring that article 1.5 (respect the work required to produce new ideas, inventions, creative works, and computing artifacts) was met. In particular, our team noticed that in order for the system to successfully calculate sleep cycle data, it had to be extracted from the data calculated by the headset.

Another major requirement that was refined was the client interface. This initially did not have any sub-requirements, but later sub-requirements had to be added in order for the stakeholder to be able to view processed data. Without continuous professional communications between group members and stakeholders whilst maintaining high standards of professional competence, conduct and ethical practice (article 2.4 from the ACM code of conduct), these optimisations would not have been possible.

Furthermore, towards the end of our project the client requirement remained, but another functional requirement that we wanted to integrate into our software was the graphical user interface (GUI). It had been expressed in the questionnaire that a graph to view sleep cycle data would be desirable for the end user and adhering to article 3.1 (ensure that the public good is the central concern during all professional computing work) this requirement had to be taken into account when documenting requirements for our system. Just having a graphical user interface independent of the other functionalities would segment our system, so as a group we devised to incorporate the GUI into our framework to ensure that it would be easier for the user to interact with the system whilst maintaining a cohesive whole.

4 Design

4.1 HLA design

In order to gain a better understanding of our system, we chose to produce three HLA diagrams which form a general idea of the separation of responsibilities in our system. The fundamental entity within our system is the EEG headset. The purpose of this device is to accumulate data about each sleep sessions which then in turns get processed and manipulated in the system. Due to our systems main focus being the motivation of its users towards attaining a more optimal sleep pattern, we have dedicated various subsystems to serving this purpose. The implementation of said goal system enables the user to set manageable goals.

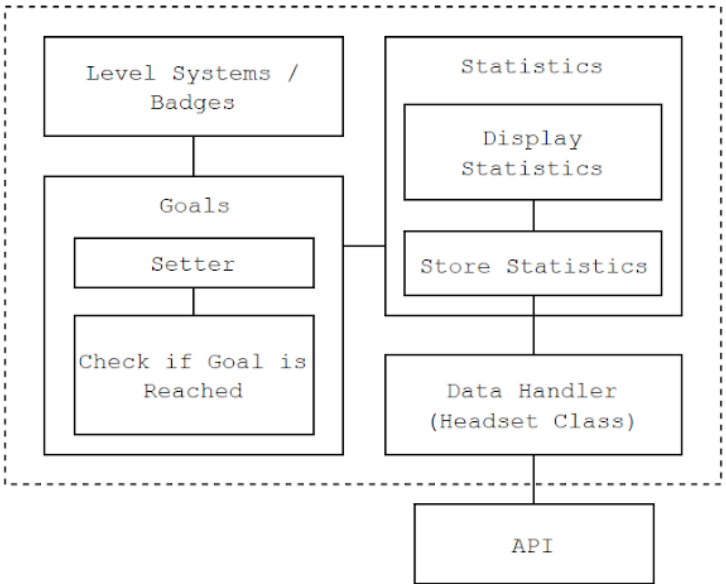


Figure 4: HLA first draft

In this first draft, the purpose of the ”Statistics” subsystem is to provide a group of functionalities that correspond to manipulating, processing and presenting the data received from the data handler, which in turn is connected to the external API. This subsystem would likewise store the data stretching a months time, which in turn will be displayed over various graphs. After a month has elapsed, the stored data would get removed from the system for the sake of user privacy, and also to free space so that the system can store new data for dates in the future. The data would be gathered through the use of a data handler which handles raw data and processes them to be utilised in our system. With the purpose of motivating the user to progress towards their goal of improving their sleep quality, we have included a level system which awards the user experience points as they progress towards their, levelling once a certain experience point have been reached. We implemented this subsystem so that the user can feel a sense of progress towards their objective. This level system would be connected to the goal system as the level would correspond to how frequently the goals have been attained.

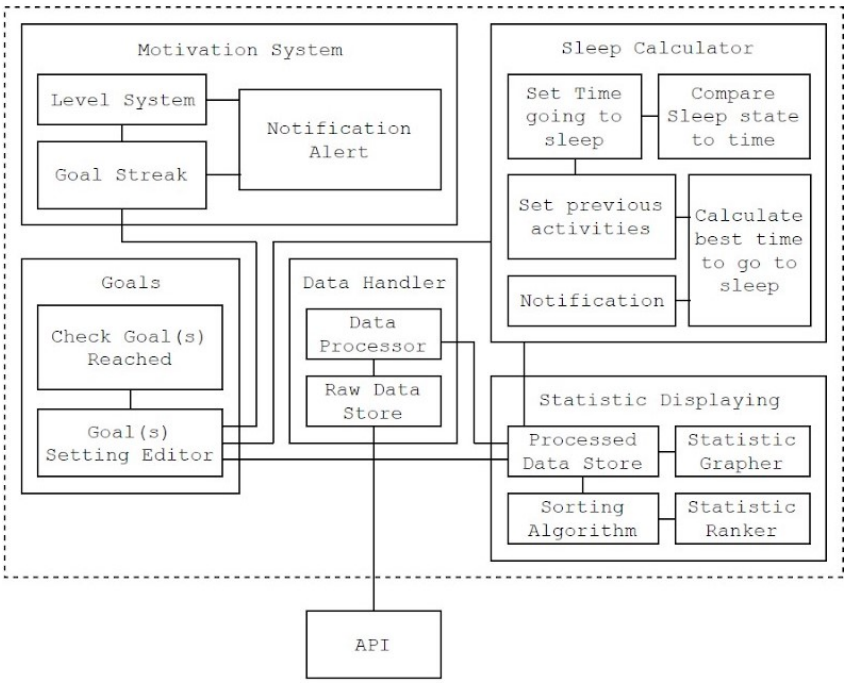


Figure 5: HLA second draft

Primarily, we expanded the Data Handler subsystem to contain 2 separate sub components: the “Data Processor”, dedicated to the serialization of raw data from the EEG headset and the “Raw Data Store”, which acts as a persistent store for the data once it has been processed. The sleep calculator system handles the responsibility of calculating the optimal times for the user to go to sleep taking into account previous activities the user has done. This subsystem is connected directly to the statistic displaying subsystem so that the calculated data can be rendered to the user. The notification system is a subcomponent of our motivation system and the sleep calculator. This function notifies the user through whether their sleep goals have been reached, what level they are at and the recommendations for the best times to go to sleep. This would happen through the GUI which will be implemented in the third HLA. The notification function would only notify the user once the GUI application has been opened.

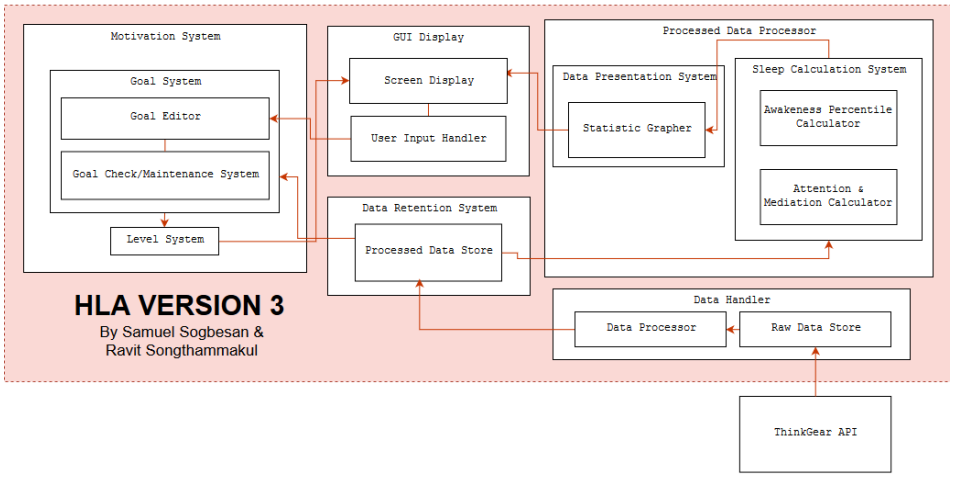


Figure 6: HLA final draft

Our final HLA design contains several refactors and adjustments to the systems within it based on reflections from our group meetings and discussions. The result is that we have decided to remove the function which gives recommendations of the most suitable time going to bed. As this function would require a very detailed algorithm which requires us to do some further extensive research in order to supply a reliable result in return. Another functionality that we have decided to remove from our system is the notification system. The reason being that the data about each sleep sessions are already recorded and been represented in terms of a line graph already, meaning that a notification system would be quite unnecessary as the graph is already self explanatory whether the user has reached their goal or not.

The “Processed Data Store” sub-component of the “Data Retention System” is used to store processed raw data within the system sen t from the ThinkGear API component which then gets fed into other subsystems like the “Sleep Calculation System” and the “Goal System”.

To further expand our system, we have decided to include our GUI display system. This is mainly used to display data packaged by the “Data Presentation System” onto graphs. The “GUI Display” system consists of two sub-components being the “Screen Display” which displays the graph about each sleep sessions using processed data gathered from the sleep calculation system, and lastly the “User Input Handler”. This sub-component is crucial as it enables the user to manually edit their goals within the system, and likewise, this would be displayed on the screen to notify the user that sleep goals have been modified.

4.2 UML model

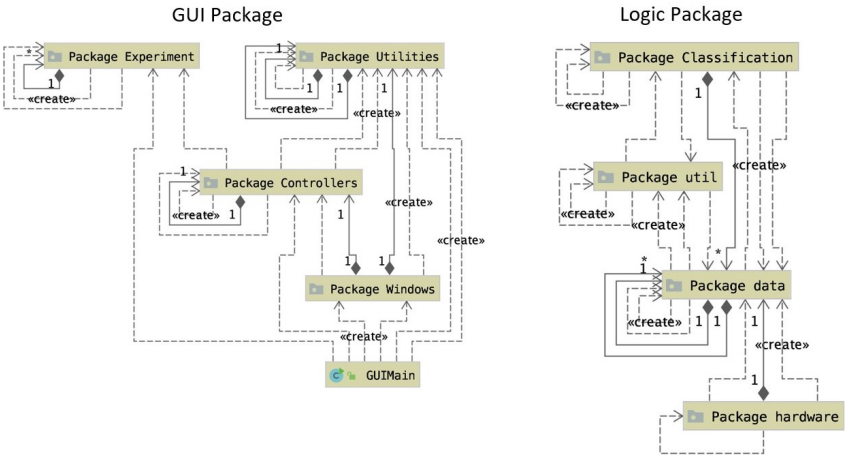


Figure 7: The diagram above provides a general overview of the relationships between the sub-packages within our system

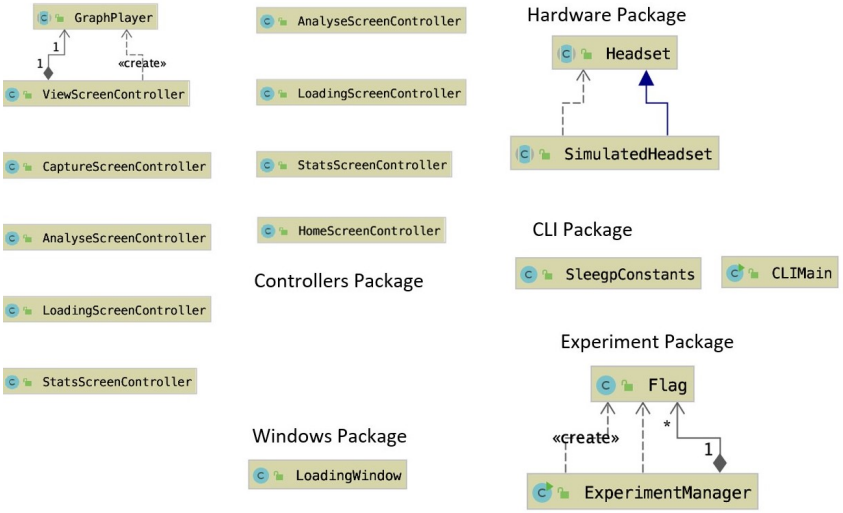


Figure 8: sub-packages from the diagram above.

Experiment manager within our Experiment package is a temporary interface used for measuring time taken for the user to enter certain states contained within the GUI. As for our Hardware package, the SimulatedHeadset class implements Headset class. For the Controllers Package, the ViewScreenController class playbacks the recordings to the GraphPlayer class.

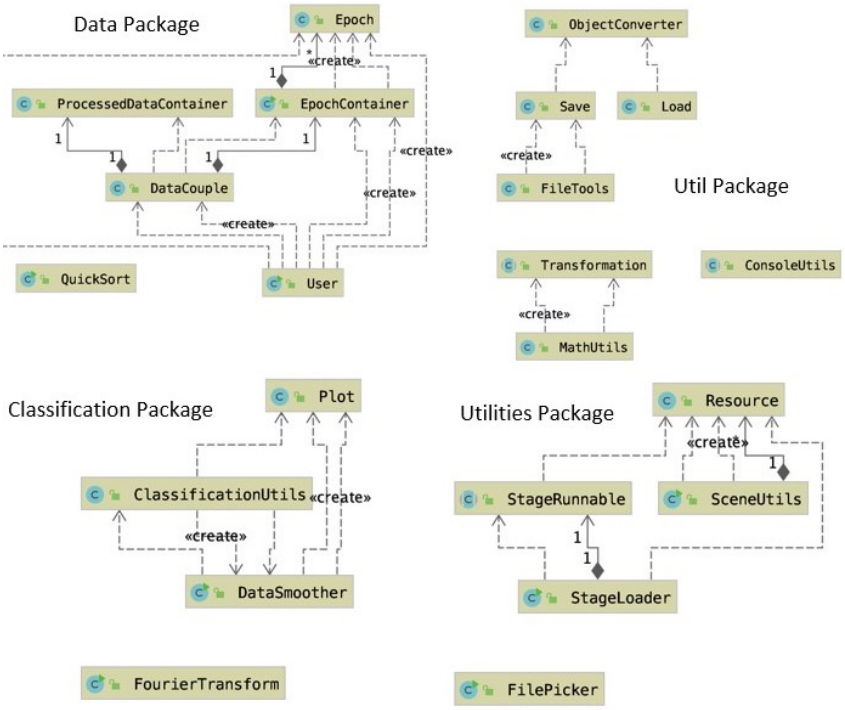


Figure 9: More sub-packages from the first segment of the UML diagram

For our Utilities Package, our StageLoader class is responsible for loading graphics from the Resource class which then in turns get displayed in the window. Implemented by the StageLoader class is the StageRunnable which contains a set of instructions associated with setting up that window (e.g. window size). The resource class is responsible for loading a graphics node from FXML file which contains the formatting of the window. In order to save the user data, the top level container object is serialised in bytes to load the top level container which later can then be deserialised. This is facilitated in the Save and Load class within the Util Package. The DataSmoother class decreases the noise within the data, eliminating the sporadic data which can then be utilised in the Plot class which stores the coordinates.

More details about the UML model are shown in the appendix.

4.3 Finite State Machine Model Semantics

A detailed finite state machine model are shown in the appendix.

State	Description
Waiting	The system is idle until it detects connection from user
Connecting	The system starts connection with detected headset
Calibrating	The system calibrates and syncs with the headset
Storing	The system exports container data to external file (such as a .csv or .ec)
Recording	The system receives and records information from the headset
Displaying	The system gives back live feed of results to the user
Drawing	The system is changing the current GUI on display, for example transitioning to a new screen or updating GUI elements

Stimulus	Description
Application starts	The user launches the main application
Detects connection from headset	The application detects a headset connection (by means of the ThinkGear API)
Connection with headset initialised	The application communicates with target device to form the basis of further communication (handshake)
User presses button	The user interacts with the graphical user interface, namely uses a button on the interface.
Automatic	System changes state without stimulus
Recording time elapsed	The system has been recording (in the recording state) for a period of time that satisfies the time the user specified
Time elapsed	Enough time has passed since the previous state was entered that the application can move to the next state

4.4 Justification of chosen designs for the final system

The final design of the HLA and UML reflects the neat compartmentalisation of our overall system, and formed the basis of our implementation. The decision to split the HLA into 5 subsystems proved to be a strength of our system as it enabled us to create distinct units of code that individually deliver core functionality to the system in our implementation. The separation of these responsibilities into various systems allowed the components to be developed and debugged apart from one another, making the development process more concurrent as we were often able to work on the implementation of a particular feature without being wholly dependent on the completion of another. By the end of development, the relationships between the subsystems would be instrumental in modelling how they would interface in implementation.

Our final HLA and UML design provide a clear mapping of the intended flow of data around the system using the relationships between components. The flow of data was deliberately engineered to protect data by restricting which components can access it, and in turn the nature of that access (read versus write). These functionalities above was implemented in line with requirement 1.5, 8 and 6.8 as a means of risk management. To further correlate our designs with our requirements, a Goal and Level system was implemented. This is with the sole purpose of motivating the user to progress towards their goal of improving their sleep quality. As a core component of our system, the responsibility of displaying sleep data through GUI was given to the "GUI Display" system.

The subsystems in our HLA clearly relate the requirements we solicited to our sprint tasks, streamlining the delegation of said tasks as they could be more easily grouped. The flexibility of our HLA design made room for the changing demand of the requirements. Subsystems could be remoulded and updated in order to better reflect the vision of the end product. This is evident through the transition from our second to third HLA design, which saw radical changes to the wider subsystems.

The purpose of our Finite State Machine diagram was to describe the general transition of states our system would follow in its final iteration. Through the repeated appearance of states such as "Drawing" and "Waiting", the primary functions of our system could be widely simplified and easier understood through the highly abstracted perspective the Finite State Machine provides. Furthermore, the stimuli used to describe what makes the system change state provide guidance on what triggers a transition of state rather than being over specific and prescribing how it should be done in implementation, leaving room for the team to decide. It could be inferred that Finite State Machine diagram had influence

on the final system design due to the extent that it mirrors it.

5 Testing

5.1 Testing Plans

In order to aid the composition of our system, test tables for every sprint were devised. The tests in the test table fully verified our system, as the tests across all of our sprints' test tables mapped to requirements in our Software Requirements Specification. Table 3 and Table 4 contain the test tables for sprint 1. The other sprints' test tables are in the appendix.

5.2 Evidence of Testing

5.3 Experiment

The purpose of our experiment was to compare two different user interfaces in terms of the time taken to access specific information in order to determine which graphical user interface suited our final system.

Hypothesis Finding:

1. a specific date, given the amount of time slept, and
2. an amount of time slept, given a date

are faster when using a graphical user interface to display the data in the form of a bar chart than in the form of a table.

Independent Variable The method used to display data; i.e. either a bar chart, with date on the x-axis and time slept on the y-axis, or a table, with a row for each date and the time slept for that date in a column of that row.

Dependent Variable The total time taken to both find a specific date based on an amount of time slept, and an amount of time slept given a date, one after another.

Procedure Participants were first asked to familiarise themselves with our system; they were given a minute to do so and then they were asked to locate a specific data point that corresponded to a date and another that corresponded to the time slept. A timer built-in to our program was used to measure the participants' response times and to reduce confounding factors. Half of the participants started with the interface using a bar chart and the other half of the participants started with the interface that utilised the table.

Data Table 5 contains data collected from the experiment. A total of 10 participants were involved in the test, each participant being chosen at random.

Results Results were confirmed by the use of a T-test calculator provided by GraphPad (n.d.)

An independent-sample t-test was conducted to compare time taken to find data in a graphical user interface using a bar chart and a graphical user interface using a table.

There was a significant difference in the time taken to access the time taken to find data For the GUI with a bar chart (M=112251.00, SD=89529.21) and GUI with a table (M=49920.10ms, SD=18920.9ms) conditions; t=2.1540, p = 0.0450

Hence by the outcome of the experiment the hypothesis has been refuted.

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Connection and Event handling of NeuroSky Headset Data							
Verify that application connects to the headset	6.1.1	ThinkGear Connector application is running on host computer	Create an instance of `Headset`, implementing any required methods, and check the return value of `capture()`	Host and port of headset connection socket	Return value of `capture()` is `true`, "Connected" printed to console	Same as expected	Pass
Verify that the system reads in raw brainwave data from the Neurosky headset	6.1.3	ThinkGear Connector application is running on host computer, application is connected to headset	Create an instance of Headset, implementing update() so that it writes received data into an EpochContainer and prints the data	Host and port of headset connection socket, number of seconds to record data for	Raw data output to console indicating brainwave data in a series of epochs	Raw data output to console indicating brainwave data in a series of epochs	Pass
Storage of Data							
Verify that the system can store raw brainwave data	6.2		Run tests for 6.2.1 and 6.2.2		Tests for 6.2.1 and 6.2.2 pass	Tests for 6.2.1 and 6.2.2 passed	Pass
Verify that the system can save data to a binary .ec file	6.2.1	Valid data has already been read from the headset into an EpochContainer, test for 6.7.1 passes	Save EpochContainer data to file using saveToFile() method	EpochContainer instance, path to the location in which to save .ec file in format /path/to/file.ec	File written at path /path/to/file.ec containing EC data from the EpochContainer instance	File written at path /path/to/file.ec containing EC data from the EpochContainer instance	Pass

Table 3: Sprint 1 Test Table (Part 1)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Storage of Data (Continued)							
Verify that the system can convert an .ec file to .csv format	6.2.2	Valid binary .ec data has already been written to a file, test for 6.7.2 passes	Load binary .ec file using EpochContainer .loadContainerFromFile(). Write the output of genCSV() method to file using FileTools.write()	Path to binary .ec file in format /path/to/file.ec, path to location in which to save .csv file in format /path/to/file.csv	File written at path /path/to/file.csv containing CSV data converted from the .ec file	File written at path /path/to/file.csv containing CSV data converted from the .ec file	Pass
Storage of Data (Continued)							
Verify that the system's command-line interface allows the user to record data from the headset and choose where to store it	6.7.1	None	Run the program and verify that the option for recording data is present, and when that option is selected, the user is prompted for a file path	None	The interface presents the expected options and allows the user to enter a file path	The interface presents the expected options and allows the user to enter a file path	Pass
Verify that the system's command-line interface allows the user to convert an existing binary .ec file to .csv format, and choose where to save the .csv file	6.7.2	None	Run the program and verify that the option for converting an .ec file is present, and that it prompts the user for both a path to the .ec file and a path to the .csv file	None	The interface presents the expected options and allows the user to enter both file paths	The interface presents the expected options and allows the user to enter both file paths	Pass

Table 4: Sprint 1 Test Table (Part 2)

Starting	Setup 1	Setup 2
Setup 1	84.394 s	61.523 s
Setup 2	31.106 s	135.565 s
Setup 2	58.984 s	65.701 s
Setup 2	51.237 s	340.293 s
Setup 2	39.361 s	67.588 s
Setup 1	53.357 s	171.851 s
Setup 2	76.320 s	36.466 s
Setup 1	28.513 s	82.397 s
Setup 1	41.905 s	62.362 s
Setup 1	34.024 s	98.764 s

Table 5: Experiment Data

Quality examined throughout the project: A specific quality of our system that was examined throughout the whole of the development process of our system was the ease of accessing data through an interface. We justify that the approaches taken as a group to ensure this quality has been met so that the end user finds it easy to find and view data. We began to examine quality in our test table for our second sprint; initially our focus was on enabling the viewing of data but feedback from our scrum master led us to consider the ease of viewing data in our system. This was further justified by the responses from the interview and the questionnaire that viewing data in graph was important as users wanted a visualisation of their data. A graphical user interface was devised to ensure that it was clearer to view brainwave data and processed data as in our system. Specific quantities of processed brainwave data would be easier to view on the multiple tabs offered by our system.

Furthermore, when designing our system, we modelled an experiment around this quality enabling stakeholders to interact with our system and as opposed to qualitative feedback about the ease of use of viewing data, quantitative feedback was also generated through the use of measuring the time taken for the user to find a specific piece of data. The experiment performed is illustrated above. From the conclusion of the above experiment we considered the results of the experiment and decided to include a table includes fields about the user sleep; namely the date that they slept, the percentage slept and the amount of sleep received. This ensured that a large amount of processed data could be viewed with ease by the end user interacting with our system.

Regarding the processes taken to address and examine the ease of viewing data it can be concluded that the means of doing so has been justified by our actions.

6 Reflection and Conclusion

6.1 Critique

What Worked Well The requirements we wrote were adaptable and open to change, and they didn't hold us down. Since the requirements were flexible, our UML designs were flexible and easy to build.

When verification work was performed correctly, it streamlined the testing process; when we had test tables prepared, performing testing of the system was made much easier. This helped us to cover all our bases and make sure nothing slipped through.

The experiment indicated that group members' initial expectations were not necessarily accurate. Through our comparison of graph- vs. table-based information presentation, we found that a table was significantly better for finding individual values; this did not match with most group members' initial intuitions. As a result, the group feels that performing the experiment was a valuable use of time.

Improvements We had duplication in various places between requirements, and we had to revise many requirements to remove this duplication. In some places, we used language that was too specific to our system, and we needed to revise requirements in order to add clarification about what that language referred to. Towards the end of the project, different group members made updates to different versions of the requirements specification document, and this required us to spend a lot of time reconciling these parallel changes.

Our work on high- and low-level designs is difficult to critique, because it was largely retroactive. This was partly due to our implementation of an Agile software process, and partly due to a disproportionate focus on implementation. Due to this retroactive nature, the designs were possibly less impactful on the implementation of the system, and needed updates as the implementation changed at various points.

Our test tables could have been improved by including more tests that were specific to individual components of the system, as well as integration tests for large portions of the system. Our tests were mainly scoped in the middle of the range, such that they pertained to individual requirements, many of which made use of more than one component of the system.

The results of the experiment may also suggest that the graph used in our graphical user interface (GUI) could do with some improvement to make it more usable.

6.2 Reflection on Process

Most group members feel that their semester 1 projects were very unstructured in comparison to this semester's project, and felt that evidence of Agility was severely lacking last semester. These group members felt that, after Sprint 1, this group was much more organised, despite the ongoing process among all group members of learning how to work in an Agile environment. Group members felt that our Sprint 1 organisation was mainly lacking due to poor communication, which was generally agreed to have improved throughout this semester.

Many group members felt that the frequent deadlines we implemented during this semester were helpful, as they prevented work from "slipping through the cracks".

The group used scrum poker to estimate the sizes of tasks. Several group members felt that these estimations were not utilised fully. This was partly due to the length of our first sprint, which meant that it would have been difficult to accurately estimate team velocity, and therefore make determinations of how much work to put into each sprint. This problem was worsened by the fact that the group misinterpreted the coursework specification in such a way that all of the functional requirements were treated as required.

Many group members felt that the group was more responsive to setbacks than we would have been in semester 1, due to the Agile approach allowing us to avoid "bottlenecks" when, for example, design tasks needed to be completed before beginning implementation tasks.

Lessons Learned

- The use of tools like `git` and GitHub helps greatly with a wide range of things, such as reducing conflicts between group members' work, streamlining the process of sending code to other group members, and clearly defining what work different group members are responsible for. GitHub allowed us to perform and enforce peer code review, which helped ensure high quality of code and helped keep group members up to date on the code being written by others.
- The use of Trello was helpful due to the ability to assign specific group members to individual tasks, and set deadlines to enforce the assignment of tasks if necessary. Trello was also helpful in organising and separating sprints. Trello has several drawbacks, however, such as the difficulty in breaking up tasks into sub-tasks. For example, small subtasks were best organised as checklists on cards corresponding to larger tasks, and this made it difficult to assign subtasks to individual different group members.
- The importance of having multiple structured meetings per week was emphasised by this semester's coursework. Having structured meetings with a well-defined agenda and goals help greatly with saving time and reducing friction when it comes to defining what tasks need to be performed.
- Having a high-quality platform for communication (e.g. Slack) is very important for group organisation.
- The use of Google Drive to store files such as images of diagrams and videos was helpful for the group. Some group members have expressed a wish for a better solution for this use case, such as software more in line with GitHub.
- Using a high-quality build system for our code (Gradle) made life much easier for the group, in particular due to the fact that different group members' systems did not necessarily have the same Java version, tools, and dependencies installed.
- Using LaTeX for the requirements made it much easier to set up cross-references and automatic numbering, and increased the ease of collaboration and merging of changes from multiple group members.
- JavaFX, and particularly SceneBuilder, made the process of developing a GUI much more streamlined.

7 References

Alhola, P. and Polo-Kantola, P., 2007. Sleep deprivation: impact on cognitive performance. *Neuropsychiatric Disease and Treatment* [Online], 3(5), pp.553–567. Available from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC2656292/.

American Sleep Association, 2019. *Sleep and sleep disorder statistics* [Online]. 28 March. 28 March. Available from: https://www.sleepassociation.org/about-sleep/sleep-statistics/.

British Broadcasting Corporation, 2010. *Air india plane crash: 'sleepy' pilot blamed* [Online]. British Broadcasting Corporation. Available from: https://www.bbc.co.uk/news/world-11772562.

British Medical Association, 2018. *Fatigue and sleep deprivation—the impact of different working patterns on doctors* [Online]. Available from: http://bmaopac.hosted.exlibrisgroup.com/exlibris/aleph/a23_1/apache_media/E6Q21YYTL9GF3G9R5B1HDNPA8E51PT.pdf.

Carlson, N.R., 2012. *Physiology of behavior*. 11th ed. Pearson.

Computing Machinery, A. for, 2018. *Acm code of ethics and professional conduct* [Online]. Available from: https://www.acm.org/code-of-ethics [Accessed 15 April 2019].

Feinberg, I., Baker, T., Leder, R. and March, J.D., 1988. Response of delta (0-3 hz) eeg and eye movement density to a night with 100 minutes of sleep. *Sleep* [Online], 11(5), pp.473–87. Available from: https://www.ncbi.nlm.nih.gov/pubmed/3227227.

GraphPad, n.d. *Graphpad* [Online]. Available from: https://www.graphpad.com/quickcalcs/ttest1.cfm [Accessed 14 April 2019].

Liang, Z. and Chapa Martell, M.A., 2018. Validity of consumer activity wristbands and wearable eeg for measuring overall sleep parameters and sleep structure in free-living conditions. *Journal of Healthcare Informatics Research* [Online], 2(1-2) (), April, pp.152–178. Available from: https://doi.org/10.1007/s41666-018-0013-1.

Mayfield Certified Health, 2016. *Electroencephalogram (eeg)* [Online]. Mayfield Certified Health. Available from: https://mayfieldclinic.com/pe-eeg.htm [Accessed 27 March 2019].

National Sleep Foundation, 2019. *Drowsy driving vs. drunk driving: how similar are they?* [Online]. National Sleep Foundation, 27 March. 27 March. Available from: https://www.sleepfoundation.org/articles/drowsy-driving-vs-drunk-driving-how-similar-are-they.

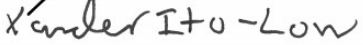
Rapp, A. and Cena, F., 2014. Self-monitoring and technology: challenges and open issues in personal informatics. In: Stephanidis, C. and Antona, M. eds. *Universal access in human-computer interaction. design for all and accessibility practice*. Springer International Publishing, pp.613–622.

Rapp, A. and Cena, F., 2016. Personal informatics for everyday life: how users without prior self-tracking experience engage with personal data. *International Journal of Human-Computer Studies* [Online], 94, pp.1–17. Available from: https://doi.org/https://doi.org/10.1016/j.ijhcs.2016.05.006.

Tuck, 2018. *Stages of sleep and sleep cycles* [Online]. Available from: https://www.tuck.com/stages/ [Accessed 27 March 2019].

A Group Contributions

Group contributions are as follows:

Name	%	Signature
Mathew Allington	16	
Tom Draper	13.8	
Aethan Foot	15	
Alexander Ito-Low	15.4	
Christophe Millischer	2	
Soren Mortensen	16	
Samuel Sogbesan	10.3	
Ravit Songthammakul (Gen)	11.5	

Notes Christophe Millischer's contribution is very low due to the fact that he changed course part of the way through the semester, and naturally stopped contributing to the project at that point.

B Sprint Backlogs

B.1 Sprint 1

6: Viewing and Collecting Sleep Data		
ID	**Requirement**	**Details**
6.1.1	The system must be able to connect to the Neurosky headset.	*Priority:* High *Dependencies:* None *Source:* Meeting minutes #9
6.1.3	The system must be able to read in raw brainwave data from the Neurosky headset.	*Priority:* High *Dependencies:* 6.1.1 *Source:* Meeting minutes #7
6.2	The system must store raw brainwave data.	*Priority:* High *Dependencies:* 6.2.1, 6.2.2 *Source:* Coursework specification, meeting minutes #7
6.2.1	The system must facilitate the saving of converted brainwave data to a binary file (.ec).	*Priority:* High *Dependencies:* 6.1.3 *Source:* Meeting minutes #12
6.2.2	The system must facilitate the conversion of brainwave data from binary (.ec) to CSV format.	*Priority:* High *Dependencies:* 6.2.1 *Source:* Meeting minutes #12
6.7.1	The command-line interface should allow the user to record data from the headset and choose where to store it.	*Priority:* High *Dependencies:* 6.1.3, 6.2.1 *Source:* Meeting minutes #13
6.7.3	The command-line interface should allow the user to convert a binary (.ec) file to CSV format, and choose where to save the converted file.	*Priority:* High *Dependencies:* 6.2.2 *Source:* Meeting minutes #13

B.2 Sprint 2

6: Viewing and Collecting Sleep Data		
ID	**Requirement**	**Details**
6.3.1	The system should extract the attention and meditation data points from the raw brainwave data.	*Priority:* High *Dependencies:* 6.2 *Source:* Meeting minutes #16
6.3.2	The system should calculate a moving average of the attention and meditation values for the whole data set.	*Priority:* High *Dependencies:* 6.3.1 *Source:* Meeting minutes #16
6.7.1	The command-line interface should allow the user to record data from the headset and choose where to store it.	*Priority:* High *Dependencies:* 6.1.3, 6.2.1 *Source:* Meeting minutes #13
6.7.2	The command-line interface should allow the user to specify an amount of time to record for, in minutes or in seconds.	*Priority:* High *Dependencies:* 6.7.1 *Source:* Meeting minutes #13

6.7.4	The command-line interface should allow the user to extract and calculate a moving average of the attention and meditation data.	*Priority:* High *Dependencies:* 6.3.2 *Source:* Meeting minutes #16

B.3 Sprint 3

6: Viewing and Collecting Sleep Data		
ID	**Requirement**	**Details**
6.3.3	The system must extract the percentage of time during a recording when the user was asleep vs. awake.	*Priority:* High *Dependencies:* 6.3.2 *Source:* Meeting minutes #16
6.4.1	The system should be able to present the percentage slept and the time slept in the form of a data table.	*Priority:* High *Dependencies:* 6.3 *Source:* Meeting minutes #17
6.6	The system should apply a "known" sorting algorithm which allows users to sort tracking data	*Priority:* Medium *Dependencies:* 6.2 *Source:* Coursework specification
6.7.2	The command-line interface should allow the user to specify an amount of time to record for, in minutes or in seconds.	*Priority:* High *Dependencies:* 6.7.1 *Source:* Meeting minutes #13
6.7.5	The command line interface must allow the user to process sleep cycle data from a raw recorded data file.	*Priority:* High *Dependencies:* 6.3.3 *Source:* Meeting minutes #16
6.7.6	The command line interface must display processed sleep cycle data in the form of a table.	*Priority:* High *Dependencies:* 6.6 *Source:* Meeting minutes #13

B.4 Sprint 4

6: Viewing and Collecting Sleep Data		
ID	**Requirement**	**Details**
6.5	The system must permit the manual entry of the average time slept for each day over a week.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification, sleep app questionnaire, British Medical Association (2018)
6.8	The system should accomodate the use of a Graphical User Interface	*Priority:* High *Dependencies:* 6.8.1, 6.8.2, **??**, **??**, 6.8.3, 6.8.5, 6.8.6, 6.8.7, **??** *Source:* Sleep application questionnaire and meeting minutes #23
6.8.1	The Graphical User Interface should allow the user to connect and disconnect their neurosky headset.	*Priority:* High *Dependencies:* 6.1.1,6.1.2 *Source:* Meeting minutes #23
6.8.2	The Graphical User Interface should allow the user start and stop recording brainwave data from the headset and choose where to save it.	*Priority:* High *Dependencies:* 6.1.3, 6.2.1 *Source:* Meeting minutes #23

6.8.3	The Graphical User Interface should allow the user to convert a binary (.ec) file to CSV format, and choose where to save the converted file.	*Priority:* High *Dependencies:* 6.2.2 *Source:* Meeting minutes #23
6.8.4	The Graphical User Interface should allow the user playback their brainwave data from a selected CSV file	*Priority:* High *Dependencies:* 6.8.2 *Source:* Meeting minutes #23
6.8.6	The Graphical User Interface must display the percentage slept and the time slept in the form of a table.	*Priority:* High *Dependencies:* 6.3 *Source:* Meeting minutes #23
6.8.7	The Graphical User Interface must allow the user to view the selected processed brainwave data in the form of a line graph.	*Priority:* High *Dependencies:* 6.4.2 *Source:* Sleep app questionnaire and Meeting Minutes #23

7: Identifying Trends in Data

ID	Requirement	Details
7.1	The system must allow the user to compare sleep data over a week.	*Priority:* High *Dependencies:* 6.2, 6.5 *Source:* British Medical Association (2018), sleep app questionnaire

8: Goals and Achievements

ID	Requirement	Details
8.1	Allow users to manage targets (goals) for time going to bed.	*Priority:* Medium *Dependencies:* 6.2, 6.5 *Source:* Coursework specification, sleep app questionnaire
8.2	Should permit user to set a daily or weekly goal for time going to bed.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification, meeting minutes #7
8.3	Should permit user to update or change goal.	*Priority:* Medium *Dependencies:* None *Source:* Coursework specification
8.4	Should motivate user to achieve their goals via a point system.	*Priority:* Medium *Dependencies:* 8.1, 8.2 *Source:* Coursework specification, sleep app questionnaire

9: Manage Data Periodically

ID	Requirement	Details
9.1	Users should be given the option to delete their sleep cycle data	*Priority:* Low *Dependencies:* 6.2, 6.3 *Source:* Meeting minutes #13, sleep app questionnaire

C Additional Test Tables

The following pages contain test tables for sprints 2–4.

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Process of Data							
Verify that the program extracts attention and meditation data from raw brain wave data	6.3.1	Valid binary .ec data has already been written to a file	Follow the steps of the test for 6.7.3	Path to binary .ec file in format /path/to/file.ec	The attention and meditation raw data points are output to the console	The attention and meditation raw data points are output to the console	Pass
Verify that a moving average of the attention and meditation is calculated	6.3.2	Valid binary .ec data has already been written to a file	Follow the steps of the test for 6.7.3	Path to binary .ec file in format /path/to/file.ec	The attention and meditation moving averages are output to the console	The attention and meditation moving averages are output to the console	Pass
Command-line Interface							
Verify that the system's command-line interface allows the user to record data from the headset and choose where to store it	6.7.1	The headset is connected	Run the program with the arguments `record -o <output-file>`	Path to the location where the outputted .ec file should be stored in format /path/to/file.ec	The program outputs various log messages as the recording happens, and outputs the recorded data to the file at the location specified	The program outputs various log messages as the recording happens, and outputs the recorded data to the file at the location specified	Pass
Verify that the system's command-line interface allows the user to convert an existing binary .ec file to .csv format, and choose where to save the .csv file	6.7.2	Valid binary .ec data has already been written to a file	Run the program with the arguments "convert <input-file> -o <output-file>"	Path to the location of the input .ec file (in the format /path/to/file.ec) and the location where the outputted .csv file should be stored (in the format /path/to/file.csv)	Valid .csv data is output to the file at the location specified	Valid .csv data is output to the file at the location specified	Pass

Table 10: Sprint 2 Test Table (Part 1)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Command-line Interface (continued)							
Verify that the system's command-line interface allows the user to extract and calculate a moving average of the attention and meditation data	6.7.3	Valid binary .ec data has already been written to a file	Run the program with the arguments "process <raw-data-file>"	Path to binary .ec file in format /path/to/file.ec	The attention and meditation moving averages are output to the console	The attention and meditation moving averages are output to the console	Pass

Table 11: Sprint 2 Test Table (Part 2)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Process of Data							
Verify that the command-line interface extracts the percentage of time during a recording when the user was asleep vs awake	6.3.3	See pre-conditions for 6.7.6.	Run tests for 6.7.6.	See test for 6.7.6.	Test for 6.7.6 passes.	Same as actual result	Pass
Presenting Data in Table							
Verify that a table is created	6.4.1	See pre-conditions for 6.7.6.	Run tests for 6.7.6	See test for 6.7.6.	Test for 6.7.6 passes.	Same as actual result	Pass
Sorting Data							
Verify that the system sorts sleep cycle data	6.6	See pre-conditions for 6.7.6.	Run tests for 6.7.6	See test for 6.7.6.	Test for 6.7.6 passes.	Same as actual result	Pass

Table 12: Sprint 3 Test Table (Part 1)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Command-line Interface							
Verify that the command-line interface allows the user to specify an amount of time to record for, in minutes or in seconds	6.7.2	Headset is connected.tead	Run the program with arguments `record -s <seconds>`, and `record -m <minutes>`.	None	The program records for a maximum of <seconds> seconds when the -s flag is used, and <minutes> minutes when the -m flag is used	Same as actual result	Pass
Verify that the command-line interface allows the user to process sleep cycle data from a raw recorded data file.	6.7.5	Valid binary .ec data has already been written to a file.	Run the program with arguments `process -o /path/to/output.sd /path/to/raw/data.ec`.	A raw recorded data file.	A binary .sd file is produced at the location specified.	Same as actual result	Pass
Verify that a command-line interface allows the user to view sorted sleep cycle data in the form of a table.	6.7.6	Valid processed data has been produced and written to a .sd file.	Run the program with arguments ‘display /path/to/processed/data.sd‘.	A processed sleep data file.	Processed data (percentages of time asleep vs awake) are output in a table, sorted in descending order by percentage.	Same as actual result	Pass

Table 13: Sprint 3 Test Table (Part 2)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Capture Screen							
Verify that the capture screen disconnects from the headset	6.8.1	A headset has been successfully connected to the API	User presses button to initiate disconnect	The state of the Neurosky API	The connected headset is successfully disconnected	Same as expected	Pass
Verify that the program records data from the headset	6.8.2	A headset has been successfully connected to the Neurosky API	User presses presses "Connect Headset" button	State of recording buttons (Enabled /Disabled)	The program will detect the connected headset and enable recording functionality	Same as expected	Pass
Analyse Screen							
Verify that percentage slept and time slept from the last week are presented in a table with the correct timestamp	6.8.6	A valid data container has been loaded	On the home screen press the Analyse button, then go to the Last Week tab. If no data is currently displayed press the process button in the top right of the screen and data recorded will be processed and added.	A binary .usr file with complete data couple(s).	That row(s) of a timestamp and corresponding percentage slept and time slept are displayed in a table.	Same as expected	Pass
Verify that the selected processed data is displayed in line chart	6.8.7	A valid data container has been loaded	On the home screen press the Analyse button, then go to the Graph tab and select a timestamp from the combobox.	A binary .usr file with a complete data couple.	That a line graph displays the processed data of the data couple selected.	Same as expected	Pass

Table 14: Sprint 4 Test Table (Part 1)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
View Screen							
Verify that the user selects a file and it outputs real time updating	6.8.4	User has files in directory	Navigate to the view screen then select a valid .ec file from the file menu	A .ec file passed in by the user	File is accepted and data is loaded onto playback graph	Same as expected	Pass
Verify that data within a file can be played back	6.8.4	User has loaded a valid Epoch Container onto the program	User uses playback button	A .ec file passed in by the user	Playback begins and the graph is updated accordingly	Same as expected	Pass
Manual Entering of Data							
Verify that the analyse screen updates the daily sleep goal via text box input	6.5	A valid data container exists	Navigate the the analyse screen -> Goal (Tab) -> Current Goal (text box), enter valid daily hours goal (1-24 hours)	Sleep goal data	The goal is updated and the graph is updated accordingly	Same as expected	Pass
Verify that the analyse screen updates the daily sleep goal via text box input	6.5	A valid data container exists	Navigate to the analyse screen -> Goal (Tab) -> Current Goal (text box), enter invalid daily hours goal (1-24 hours)	Sleep goal data	The input is rejected and the user is prompted to input new data	Same as expected	Pass
Exporting of Data							
Verify that the view screen is able to export files to a .csv file	6.8.3	A valid data container exists	Navigate to the view screen, then load a valid .ec file from the file menu, or load a container using “Load from user”. Then choose “Export CSV” from the file menu and save it to a file.	A valid Epoch-Container.	A valid .csv file is exported.	Same as expected	Pass

Table 15: Sprint 4 Test Table (Part 2)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
Playback of Data							
Verify that the raw brain wave data can be illustrated on a graph and played back	6.8.4	A valid data container has been loaded	Follow steps for 6.4.1 and 6.4.2	A binary .usr file with complete data couple(s).	The view will display playback of raw data and the analyse screen will provide an overview of processed sleep time probability graph	Same as expected	Pass
Data Comparison							
Verify that the system displays a table with the users sleep data, to compare over a week	7.1	A valid data container exists	Navigate to Analyse Screen -> last week (tab)	A binary .usr file	Table with the columns: Date, Percentage Slept, Time Slept. Each row containing relevant data from each sleep.	Same as expected	Pass
Goals							
Verify the user has access to their goal data	8.1	Valid goal data exists	On the home screen press the Analyse button, the default screen will display the goal.	A binary .usr file	That the user can see their goal on the relevant screen	Same as expected	Pass
Verify that users are able to set their goal data	8.2	-	On the home screen press the Analyse button, the default screen will have a text field to set the goal.	A binary .usr file	That the user can set their goal on the relevant screen	Same as expected	Pass
Verify that users are able to update their goal data	8.3	Valid goal data exists	On the home screen press the Analyse button, the default screen will have a text field to set the goal.	A binary .usr file	That the user can edit their goal on the relevant screen	Same as expected	Pass

Table 16: Sprint 4 Test Table (Part 3)

Test Case	Requirement	Pre-conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
			Motivation System				
Verify that points are added for each epoch container in that week	8.4	-	On the home screen press the Stats button.	A binary .usr file	That the points are added and displayed on the stats screen	Same as expected	Pass
			Data Management				
Verify that a selected data couple can be deleted by the user	9.1	-	On the home screen press the Analyse button, then go to the Last Week tab, then select a row you want to delete and then press the delete button.	A binary .usr file	That the selected data couple is deleted and removed from the binary .usr file	Same as expected	Pass

Table 17: Sprint 4 Test Table (Part 4)

C.1 Meeting Minutes

C.2 Meeting 1

Date & Time	08/02/19 from 11:15–12:05
Location	8W 2.1
Meeting type	Friday meeting
Attendees	Soren, Gen, Christophe, Aethan, Mat, Xander
Absent	Sam, Tom
Agenda	Introduction
Discussion	Communication methods were discussed as well as establishing a meeting date.
Conclusions	GitHub, Trello, Slack and a Google Drive folder were set up.

C.2.1 Action Items

Action Item	Person Responsible	Deadline
Read coursework specification	Whole group	12/02/19
Track down missing group members	Whole group	12/02/19
Think of an idea for the project	Whole group	12/02/19

C.3 Meeting 2

Date & Time	12/02/19 from 13:15–14:15
Location	8W 1.28
Meeting type	Regular meeting
Attendees	Soren, Gen, Christophe, Aethan, Mat, Xander, Tom, Sam
Absent	None
Agenda	Sharing of ideas
Discussion	We set up a backlog of functional and non-functional requirements on Trello and discussed possible ideas for the personal informatics project.
Conclusions	EEG headset to measure a base variable sleep was agreed upon as our idea. A Skype meeting was planned to cover the technology and the project scope.

C.3.1 Action Items

Action Item	Person Responsible	Deadline
Research EEG headset	Whole group	15/02/19

C.4 Meeting 3

Date & Time	15/02/19 from 16:30–17:05
Location	Skype
Meeting type	Skype meeting
Attendees	Soren, Mat, Xander, Tom
Absent	Sam, Christophe, Gen (apologies), Aethan (apologies)
Agenda	Scope of the project and the tools to aid us with the execution of implementing our system.
Discussion	Trello vs GitHub project boards and the scope of implementation were discussed.
Conclusions	We are going to use Trello as means for monitoring the agile process and we plan to borrow Dr. Simon Jones's EEG headset.

C.4.1 Action Items

Action Item	Person Responsible	Deadline
Research measuring sleep and EEG	Whole group	15/02/19
Install the Trello add on: Scrum by Vince	Whole group	15/02/19
Watch the videos provided by Matt	Whole group	15/02/19

C.5 Meeting 4

Date & Time	15/02/19 from 11:15–12:05
Location	8W 2.1
Meeting type	Friday meeting
Attendees	Tom, Xander, Christophe, Aethan, Gen, Sam, Soren
Absent	Mat (apologies)
Agenda	`git` skills
Discussion	`git` tutorial led by Soren.
Conclusions	We need to talk about our first sprint.

C.5.1 Action Items

Action Item	Person Responsible	Deadline
None		

C.6 Meeting 5

Date & Time	19/02/19 from 11:15–13:05
Location	8W 1.28
Meeting type	Tuesday meeting
Attendees	Tom, Xander, Aethan, Gen, Soren, Sam, Mat
Absent	Christophe (apologies)
Agenda	First sprint
Discussion	Created first HLA draft by taking the requirements functional requirements and refactoring them to get ready to start our first sprint. We then ensured that we were on the right JDK as there were clashes with running code sent by other users. Soren taught us about branches and we looked at the code Mat wrote to ensure that data from the headset was being transmitted to the console. It was at this stage that sprint tasks were formulated.
Conclusions	*None given*

C.6.1 Action Items

Action Item	Person Responsible	Deadline
Learn JavaFX	Whole group	*None given*

C.7 Meeting 6

Date & Time	21/02/19 from 14:15–15:05
Location	8W 1.28
Meeting type	Tuesday meeting
Attendees	Tom, Xander, Aethan, Gen, Soren, Christophe, Sam, Mat
Absent	None
Agenda	Reflection
Discussion	We reflected upon the previous tasks to be verified before continuing with the first sprint.
Conclusions	*None given*

C.7.1 Action Items

Action Item	Person Responsible	Deadline
Think about requirements	Xander, Gen	*None given*
Continue planning the code	Soren, Mat	*None given*

C.8 Meeting 7

Date & Time	22/02/19 from 09:15–10:05
Location	1W 3.56
Meeting type	Friday early meeting
Attendees	Xander, Aethan, Gen, Soren, Mat, Sam
Absent	Tom, Christophe
Agenda	Prioritising requirements
Discussion	We used the whiteboard to prioritize backlog requirements using high, medium and low to classify them and then discussed which requirements to focus on. We broke requirements down in order to make them more specific as they were too broad before. See Figure 10 for additional notes.
Conclusions	*None given*

C.8.1 Action Items

Action Item	Person Responsible	Deadline
None		

C.9 Meeting 8

Date & Time	26/02/19 from 12:15–13:05
Location	8W 1.28
Meeting type	Tuesday meeting
Attendees	Xander, Aethan, Gen, Soren, Sam, Tom, Christophe, Mat
Absent	None
Agenda	Checking the progress of the first sprint
Discussion	As a group we sat down and went through the work of our other colleagues and made sure that everything was on track to getting completed for Friday.
Conclusions	*None given*

C.9.1 Action Items

Action Item	Person Responsible	Deadline
Finish requirement draft	Xander, Gen	01/03/19
Finish HLA draft	Aethan	01/03/19
Finish test draft	Mat, Sam	01/03/19
Understand more about `git`	Whole team	01/03/19
Write a program that validates tests	Mat, Tom	01/03/19

C.10 Meeting 9

Date & Time	01/03/19 from 10:15–11:05
Location	8W 2.10
Meeting type	Friday meeting
Attendees	Xander, Gen, Soren, Sam, Tom, Mat, Aethan
Absent	Christophe
Agenda	Sprint 1 refactoring
Discussion	We realized that our sprint required us to backlog the requirements and the tests that we were going to verify and validate in the upcoming sprint.
Conclusions	It was advised by the group that a requirement that ensured the connectivity of the headset to the application be added in as well as a disconnect feature.

C.10.1 Action Items

Action Item	Person Responsible	Deadline
Backlog requirements	Xander	01/03/19
Backlog tests	Mat, Sam	*None given*

Addendum In the sprint review meeting on the afternoon of the same day, it appeared that test 5 failed. Therefore, our team organised a meeting for Monday to regroup.

C.11 Meeting 10

Date & Time	01/03/19 from 10:15–11:05
Location	8W 2.10
Meeting type	Monday meeting
Attendees	Xander, Gen, Soren, Sam, Tom, Mat
Absent	Christophe, Aethan
Agenda	Sprint 1 refactoring
Discussion	We discussed as a group what needed to be altered in the test for sprint 1 by talking about the failed test. Then allocated sprint tasks to team members.
Conclusions	Members have been allocated work; the plan was to review work in the CSED tutorial.

C.11.1 Action Items

Action Item	Person Responsible	Deadline
Draw a finite state machine draft	Sam, Christophe	08/03/19
Create a use case diagram and a scenario	Xander, Tom	08/03/19
Fix tests for Jo	Whole group	08/03/19
Research JUnit	Mat, Soren	08/03/19

C.12 Meeting 11

Date & Time	08/03/19 from 10:15–11:05
Location	8W 1.28
Meeting type	Friday meeting
Attendees	Aethan, Christophe, Gen, Mat, Sam, Soren, Tom, Xander
Absent	None
Agenda	Writing the introduction
Discussion	As a group, we brought research to the meeting and started to draft an introduction.
Conclusions	At the signing off on the first sprint it appeared that the requirements backlog didn't match up with the software tests thus a skype meeting needed to be arranged.

C.12.1 Action Items

Action Item	Person Responsible	Deadline
None		

C.13 Meeting 12

Date & Time	26/02/19 from 12:15–13:05
Location	8W 1.28
Meeting type	Skype meeting
Attendees	Xander, Sam, Mat
Absent	
Agenda	Going through requirements
Discussion	We looked through the requirements table and modified the requirements based on the scope of our first sprint. We then planned to validate our work the next time the whole group was present.
Conclusions	In the requirements it was decided to map functional requirements relating to the headset interface as one requirement with sub-requirements, to group common attributes together. It was also suggested that requirements with sleep cycle data be left until the next sprint as the data our application processed was brainwave data. Specific requirements for files facilitated by application had to be specified. Finally, the requirement about storing data was refactored so that it included capability for the user to store their data in a file.

C.13.1 Action Items

Action Item	Person Responsible	Deadline
Sign off questionnaire linking to requirements	Whole group	*None given*
Discuss a method of sorting data in meeting	Whole group	*None given*
Refactor requirements backlog	Xander	15/03/19
Refactor test backlog	Mat, Sam	15/03/19

C.14 Meeting 13

Date & Time	12/03/19 from 11:30–12:20
Location	8W 1.28
Meeting type	Monday meeting
Attendees	Soren, Xander, Christophe, Aethan
Absent	Sam, Gen, Tom (apologies), Mat (apologies)
Agenda	Validating requirements
Discussion	During the meeting as a group we discussed the requirements in relation to our first increment of the software to ensure that the requirements mapped to the functionality that we wanted our system to provide.
Conclusions	We concluded that: • Our system must extract related data so our application domain is restricted to sleep. • Our system should implement a command line interface as part of the increment. • The raw data should be deleted if the user doesn't intend to keep hold of it.

C.14.1 Action Items

Action Item	Person Responsible	Deadline
Write a README for versioning of the system	Soren	*None given*
Put the requirements being tested into the backlog	Xander	15/03/19
Map the test table to the requirements backlog	Mat, Sam	15/03/19

C.15 Meeting 14

Date & Time	15/03/19 from 10:15–11:05
Location	8W 2.10
Meeting type	Friday meeting
Attendees	Tom, Aethan, Soren, Xander
Absent	Gen, Sam, Mat, Christophe
Agenda	*None given*
Discussion	Jo advised us to meet up to discuss further actions to be taken in ensuring that our progress caught up as we were starting to fall behind in terms of our sprint progress. At the sprint test we managed to sign off our first sprint.
Conclusions	We concluded that we need to meet ASAP after the first sprint.
Action Items	Action items have moved to Trello.

C.16 Meeting 15

Date & Time	18/03/19 from 11:15–12:05
Location	CB 4.7
Meeting type	Monday meeting
Attendees	Aethan, Christophe, Gen, Mat, Sam, Soren, Tom, Xander
Absent	None
Agenda	Action plan for future sprints
Discussion	As a group we reviewed what needed to be changed in order to catch up. It was decided to have 3 meetings in the week: one on Monday, one on Tuesday and one on Thursday. The Monday session would be to assign sprint tasks to people and estimate point values, the Tuesday meeting would be to work through sprint tasks, the Thursday meeting would be to plan the next sprint and the already scheduled Friday meeting would be to review sprint progress. We then started to allocate these times in our timetables.
Conclusions	Organise, Estimate point values, Allocation, Review, Next task. Have people validate their work more thoroughly by cross training and peer review.
Action Items	Action items have moved to Trello.

C.17 Meeting 16

Date & Time	19/03/19 from 11:15–13:05
Location	CB 4.6
Meeting type	Tuesday meeting
Attendees	Aethan, Christophe, Gen, Mat, Sam, Soren, Tom, Xander
Absent	None
Agenda	Organizing priorities in the backlog, selecting sprint backlog and reducing items into tasks.
Discussion	We first refactored the requirements backlog so that we could clearly see priorities of tasks. We then discussed about choosing suitable requirements for the sprint backlog, then broke them down into tasks, assigned them and estimated them through the use of Trello.
Conclusions	It was decided to extract specifically attention and meditation from the data points collected by the headset and calculate a moving average of the extracted values. It was also noted that the command line interface had to implement these features as well, hence requirements 6.3.1 and 6.3.2 were created as well as requirement 6.7.4.
Action Items	Action items have moved to Trello.

C.18 Meeting 17

Date & Time	21/03/19 from 11:15–12:05
Location	CB 3.10 and CB 4.7
Meeting type	Thursday meeting
Attendees	Soren, Mat, Gen, Sam, Xander, Aethan,Tom
Absent	Christophe
Agenda	Planning next sprint
Discussion	We allocated sprint tasks from the requirements backlog, split them down into tasks, estimated them. When planning, we also considered possible risks of other projects from seperate subjects and acknowledged the risk by planning to monitor work more closely and having more frequently.
Conclusions	It was decided that to give the user of our system more options when viewing data requirement 6.4 Had to be expanded into subsections 6.4.1 allows the user to view data in the form of a table and 6.4.2 allows the user to view the data in the form of a graph as opposed to just being able to view The data in the form of a graph as it allows for flexibility.
Action Items	Action items have moved to Trello.

C.19 Meeting 18

Date & Time	22/03/19 from 10:15–11:05
Location	8W 2.10
Meeting type	Friday meeting
Attendees	Xander,Gen,Tom, Matt, Sam, Aethan
Absent	Christophe, Soren(apologies)
Agenda	Acknowledgment of group progress
Discussion	We found out that Christophe had been absent for some meeting without reason and planned work allocation around this and also the potential risk of the mumps outbreak. We also worked on our tasks respectively. We then completed our second sprint in the evening.
Conclusions	Locate Christophe and continue to regularly meet up
Action Items	Action items have moved to Trello.

C.20 Meeting 19

Date & Time	25/03/19 from 13:15–14:15
Location	CB 4.7
Meeting type	Monday meeting
Attendees	Xander, Tom, Mat, Soren, Aethan
Absent	Gen, Sam, Christophe
Agenda	Allocating sprint tasks to people
Discussion	We allocated sprint 3 tasks to people and estimated their story points. We also started to discuss the experiment that we would perform. It was suggested that the experiment compared two Graphical user interfaces. Christophe may be leaving the course so we had to allocate tasks that could accommodate for the risk
Conclusions	Meet up tomorrow
Action Items	Action items have moved to Trello.

C.21 Meeting 20

Date & Time	26/03/19 from 11:15–13:15
Location	CB 4.7
Meeting type	Work Meeting
Attendees	Xander, Tom, Soren, Gen
Absent	Christophe (suspended), Mat (apologies), Aethan
Agenda	Continue to work on assigned tasks
Discussion	We discussed design features with the client interface that needed to be addressed including a client interface that waits in the main method until the client exits the app and allows the client to exit the application.
Conclusions	Meet up on Thursday
Action Items	Action items have moved to Trello.

C.22 Meeting 21

Date & Time	28/03/19 from 11:15–13:15
Location	1W 2.103
Meeting type	Thursday meeting
Attendees	Xander, Tom, Soren, Mat, Aethan
Absent	Christophe (suspended), Gen (apologies), Sam (apologies)
Agenda	Planning sprint 4
Discussion	We prioritized our sprint backlog for sprint 4. Then worked on our tasks assigned to us in sprint 3.
Conclusions	Continue to work on assigned tasks.
Action Items	Action items have moved to Trello.

C.23 Meeting 22

Date & Time	29/03/19 from 10:15–11:05
Location	8W 2.1
Meeting type	Friday meeting
Attendees	Xander, Tom, Soren, Mat, Aethan
Absent	Christophe (suspended), Gen, Sam
Agenda	Discussion about group contribution.
Discussion	In our sprint we discussed and reflected group contributions using the Trello board. Our group realized that the Trello board wasn't too accurate in determining the amount of contributions for each member so we decided to use the Trello board as a rough guideline. In consideration to the meeting minutes, a recommendation was to use meeting minutes as a factor to determine the group contributions.
Conclusions	Think about tasks to come over the weekend.
Action Items	Action items have moved to Trello.

C.24 Meeting 23

Date & Time	01/04/19 from 13:15–14:05
Location	CB 4.7
Meeting type	Monday meeting
Attendees	Xander, Tom, Aethan, Soren, Mat
Absent	Christophe (suspended), Gen, Sam (apologies)
Agenda	Sprint 4 allocation
Discussion	In this meeting we discussed the implementation of a graphical user interface as we wanted to improve user experience. This entailed splitting the Graphical User interface into 3 segments: A capture screen where the user could record sleep cycles,a view screen where the user could play back their night's sleep, an analyse screen where the user could analyse their sleep for each night in further detail and a goal screen where the user could set weekly goals for their sleep.
Conclusions	Continue to work throughout the week
Action Items	Action items have moved to Trello.

C.25 Meeting 24

Date & Time	02/04/19 from 13:15–14:05
Location	CB 4.13
Meeting type	Tuesday meeting
Attendees	Tom, Aethan, Soren, Gen, Sam
Absent	Christophe (suspended), Xander, Mat(apologies)
Agenda	Work meeting
Discussion	We worked on our respective tasks, no new conclusions were made
Conclusions	None
Action Items	Action items have moved to Trello.

C.26 Meeting 25

Date & Time	04/04/19 from 11:15–13:05
Location	CB 4.9
Meeting type	Thursday meeting
Attendees	Tom, Aethan, Soren, Sam, Xander, Mat
Absent	Christophe (suspended), Gen
Agenda	Work meeting
Discussion	We worked on our respective tasks. No conclusions were raised.
Conclusions	None
Action Items	Action items have moved to Trello.

C.27 Meeting 26

Date & Time	05/04/19 from 10:15–11:05
Location	8W 2.1
Meeting type	Friday meeting
Attendees	Everyone
Absent	Christophe (suspended)
Agenda	Work meeting
Discussion	We discussed postponing the sprint to get all the requirements done, we acknowledged the disparity in group contributions and talked about making the group contributions more even. We then analysed our current sprint tasks.
Conclusions	Continue to work during the week.
Action Items	Action items have moved to Trello.

C.28 Meeting 27

Date & Time	08/04/19 13:15–14:05
Location	1 W3.56
Meeting type	Monday meeting
Attendees	Soren, Matt, Aethan, Xander, Gen, Sam
Absent	Christophe (suspended), Tom
Agenda	Planning final sprint meeting
Discussion	We set deadlines of the tasks we needed to get done in order to have everything ready for Thursday's tutorial.
Conclusions	Continue to work on tasks during the week.
Action Items	Action items have moved to Trello.

C.29 Meeting 28

Date & Time	09/04/19 11:15–13:05
Location	8W 1.28
Meeting type	Tuesday meeting
Attendees	Everyone
Absent	Christophe (suspended)
Agenda	Work on respective tasks
Discussion	As a team, we continued to work on our assigned tasks.
Conclusions	Continue to work on tasks assigned on trello
Action Items	Action items have moved to Trello.

C.30 Meeting 29

Date & Time	09/04/19 11:15–13:05
Location	8W 1.28
Meeting type	Tuesday meeting
Attendees	Everyone
Absent	Christophe (suspended)
Agenda	Work on respective tasks
Discussion	As a team, we continued to work on our assigned tasks.
Conclusions	Continue to work on tasks assigned on trello
Action Items	Action items have moved to Trello.

C.31 Meeting 30

Date & Time	12/04/19 10:15–11:05
Location	8W 2.1
Meeting type	Friday final meeting
Attendees	Everyone
Absent	Christophe (suspended)
Agenda	Conclude on final sprint tasks and finish documentation for Tuesday
Discussion	As a team we concluded our final sprint. We then reflected on our process to reach our final system
Conclusions	Finish documentation for Tuesday
Action Items	Action items have moved to Trello.

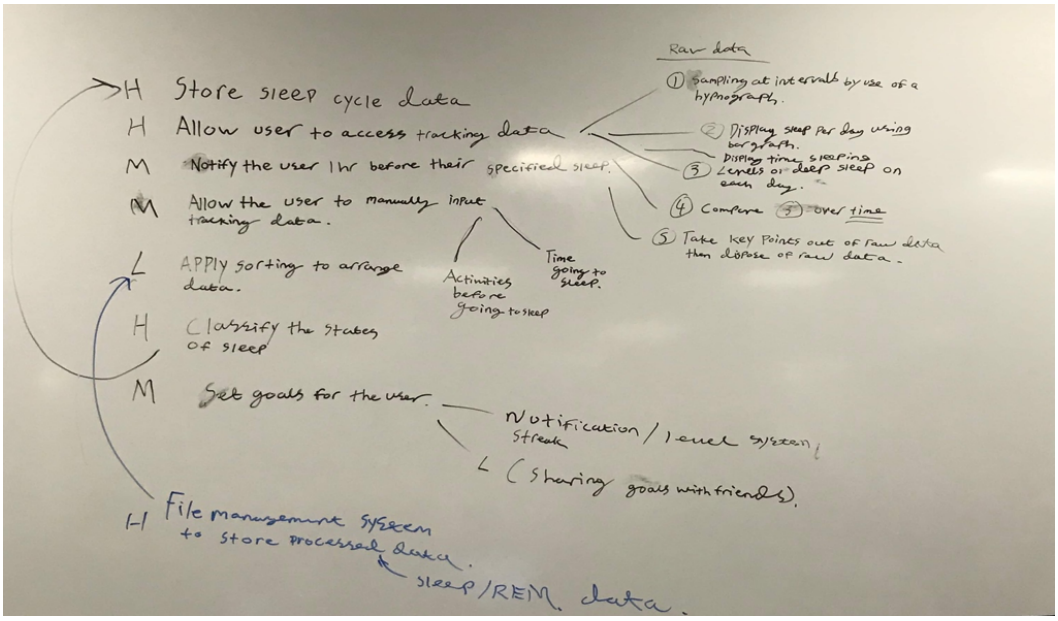


Figure 10: Meeting 7 Additional Notes

C.32 Questionnaire

A questionnaire to elicit requirements was conducted and the responses are shown below.

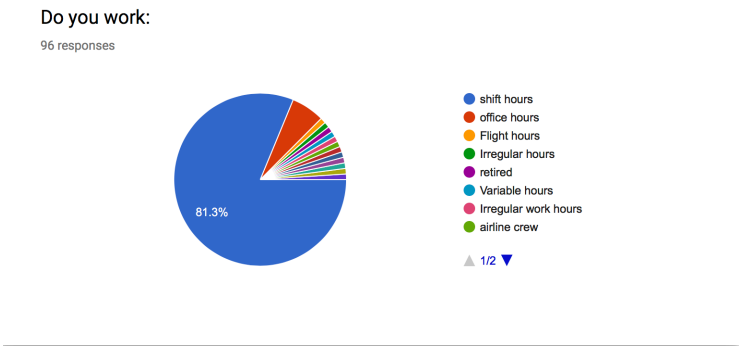


Figure 11: Question1

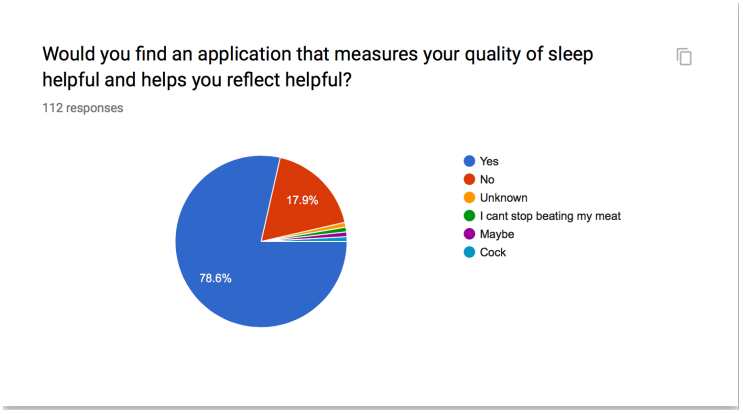


Figure 12: Question2

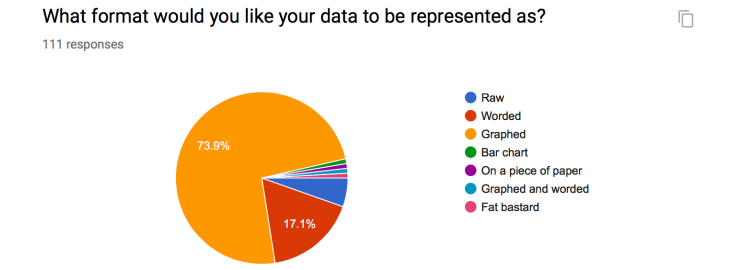


Figure 13: Question3

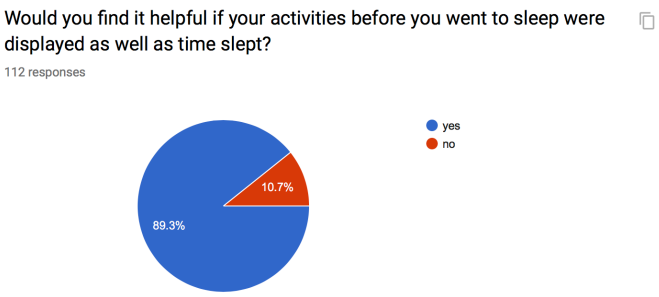


Figure 14: Question4

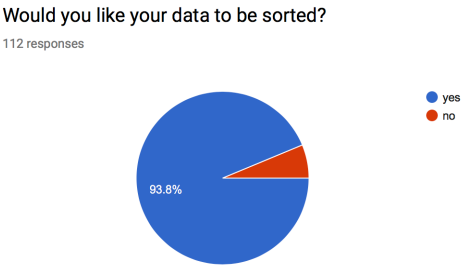


Figure 15: Question5

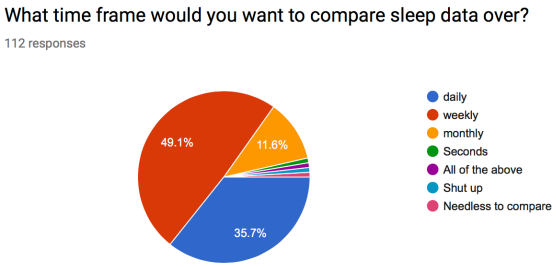


Figure 16: Question6

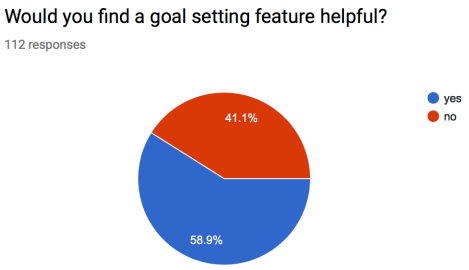


Figure 17: Question7

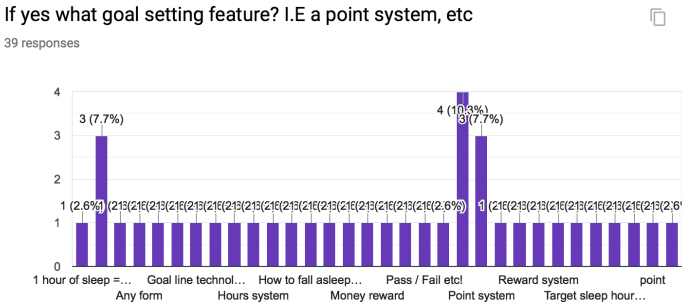


Figure 18: Question8

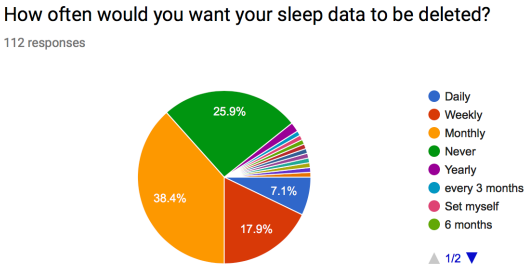


Figure 19: Question9

C.33 Interview

Interview

Samuel Sogbesan & Ravit Songthammakul

Interviewee: Marcelina Stanowska (Accounting & Finance Student, University of Bath)
Date & Time: 19:30, 20/03/2019

1. *How would you best describe your working hours? (Ask for clarification in pattern, hours worked and perceived intensity)*
   "I work around 5 hours a day 3 times a week at Second Bridge in town. The hours seem light but due to the type of work it is, it definitely takes its toll on me. I'm not a great sleeper too, so it takes a while for me to settle."
2. *How do you currently maintain and manage your sleep? How successful is your current solution? Where does it fall short?*
   "Honestly, I currently don't have a steadfast solution to "manage" my sleep like that. Normally after work or uni I'm knackered, but cause of other commitments might not get to bed before 12 am most of the time."
3. *Can you detail how you use your phone and laptop? How interested would you be in a software solution as an alternative to your current one?*
   "My phone is always on me. I'm quite an active social media user, and find it an important tool from day to day. Not to mention I make use of it for music etc. My laptop is what i use for my work, I carry it to all my lectures. At the same time, I use it alot to stream my favourite shows too. Other than that, I don't have much use for it. I'd actually like to use it more often if I could, I know there are a world of more useful things I could be using it for."
4. *How comfortable are you sleeping with or around peripherals?*
   "As uncomfortable as it sounds, I often sleep with my Apple Airpods in my ears and put on a podcast as it helps me to get to sleep easier. So I don't think peripherals are a problem for me."
5. *How well do you respond to competition? Do you think your sleep pattern would benefit from a solution with a gamified element?*
   "Competition is in my nature! I've always enjoyed being involved in competition. Although I'm not sure how that could be applied to my sleep, I'd be very inviting to anything that can incentivise me to sleep earlier!"
6. *In our solution we intend to locally store your sleep data to analyse and later present back to you in the form of graphs and other transformative media. In relation to your privacy, what are your views on us retaining your data? Would you value a feature to control to what extent this is?*
   "Privacy is not a giant priority for me - that being said, having autonomy over how much data is saved, since you said it is saved on my device, it would be nice to control how much is saved, since it might end up being alot of data and I don't have a particularly high memory phone."

Figure 20: Interview Questions

7. *We intend to add a component to the software in order to motivate you to follow the optimal sleep pattern for you as closely as possible. For a goal setting feature, what are the must have components you would expect to see? Are there any components that you feel are necessary for the success of a component such as this?*
"In my personal opinion, the most critical feature to be added is the alerting system which notifies the user how well their sleep quality have been. Another useful feature that could get implement can be an option for the user to be able to alter their goals whether they prefer to sleep long hours with less quality or shorter hours with a better quality of sleep."

Figure 21: Interview Questions

C.34 UML Diagrams

Below are our UML class models for each class

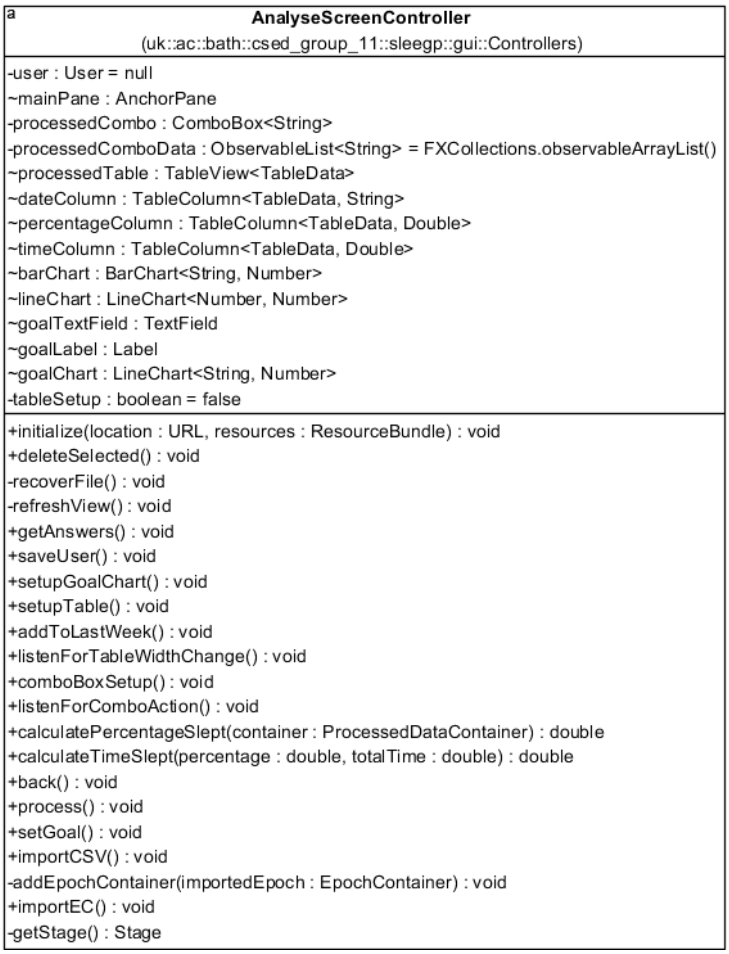


Figure 22:

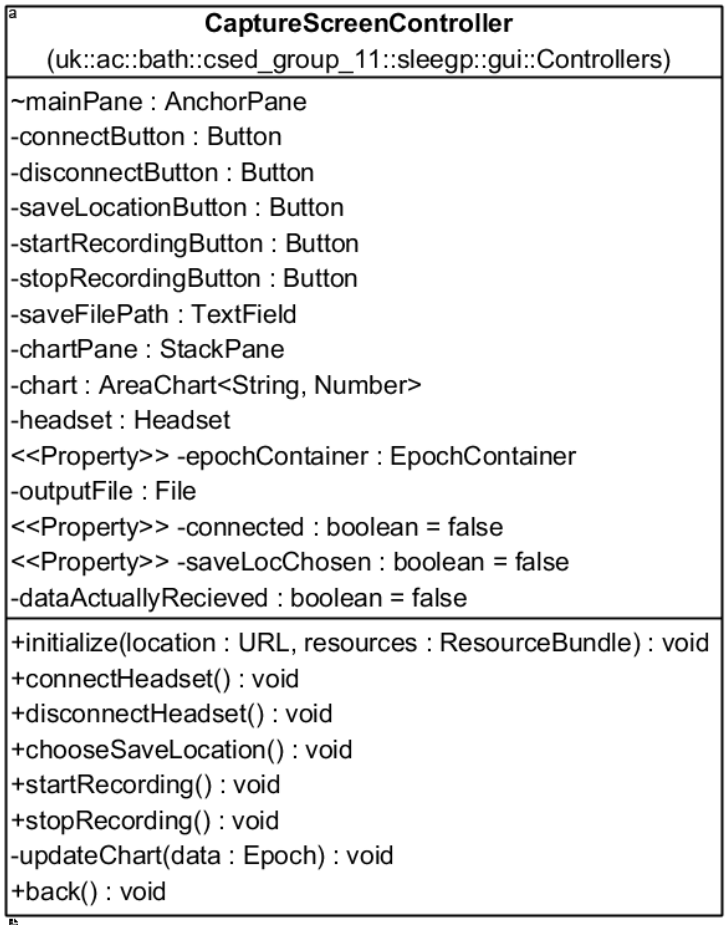


Figure 23:

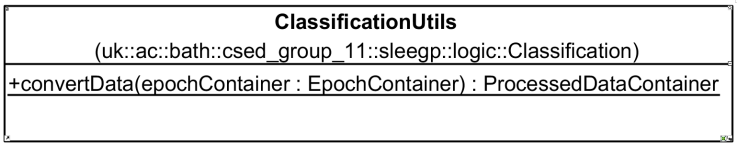


Figure 24:

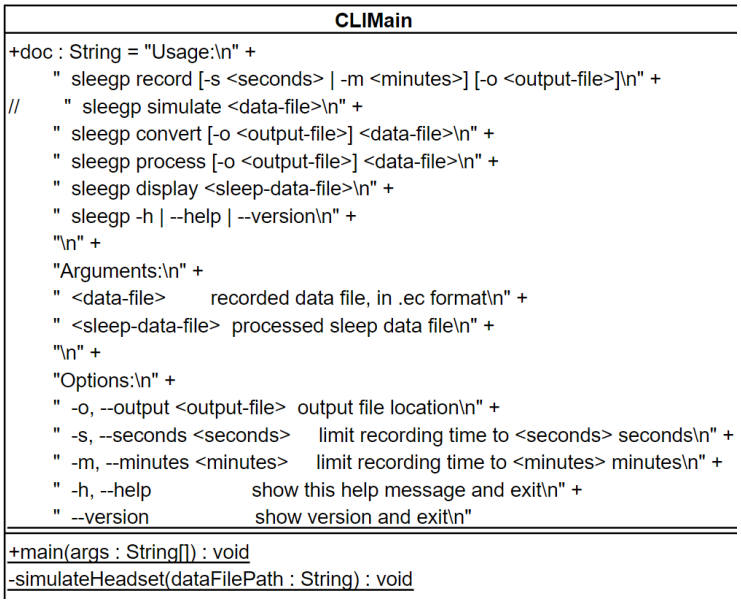


Figure 25:

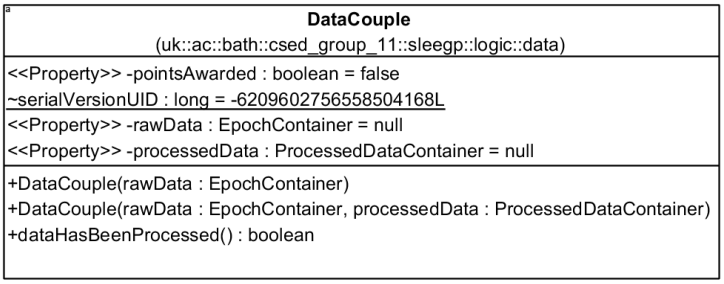


Figure 26:

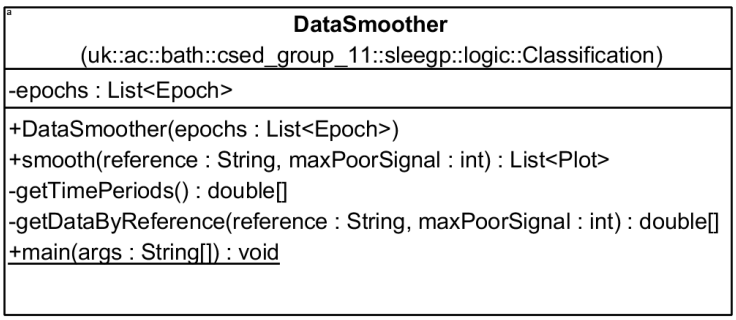


Figure 27:


Epoch
(uk::ac::bath::csed_group_11::sleegp::logic::data)
~serialVersionUID : long = -8619870403385249300L
<<Property>> -timeElapsed : long
<<Property>> -timeStamp : String
<<Property>> -attention : int
<<Property>> -meditation : int
<<Property>> -delta : int
<<Property>> -theta : int
<<Property>> -lowAlpha : int
<<Property>> -highAlpha : int
<<Property>> -lowBeta : int
<<Property>> -highBeta : int
<<Property>> -lowGamma : int
<<Property>> -highGamma : int
<<Property>> -poorSignalLevel : int
+Epoch(JSON : String, timeElapsedMillis : long)
+Epoch(timeElapsed : int)
+Epoch(csvRow : String)
-addData(obj : JSONObject) : void
+setIntField(reference : String, data : int) : void
+toString() : String
+getCSV() : String
+getCSVHeader() : String
+timeElapsed() : long
+getByReference(reference : String) : int
+compareTo(o : Epoch) : int

Figure 28:

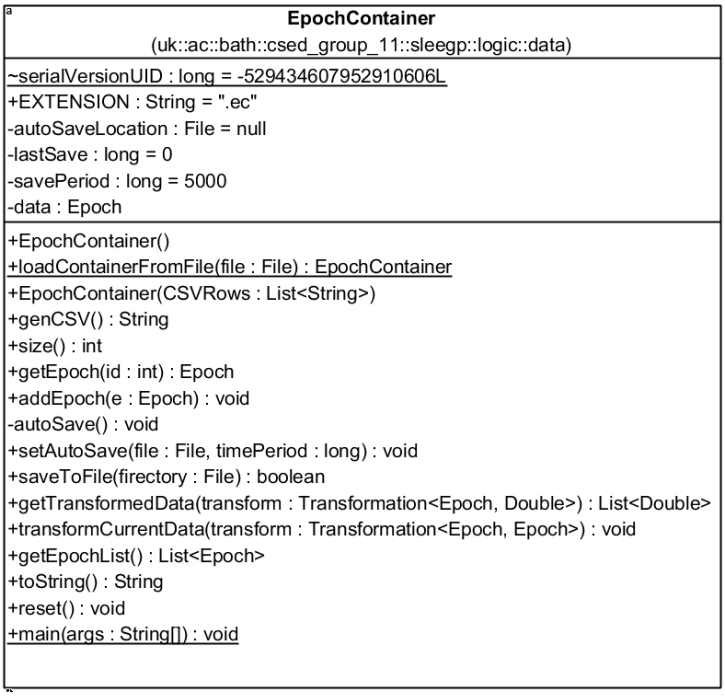


Figure 29:

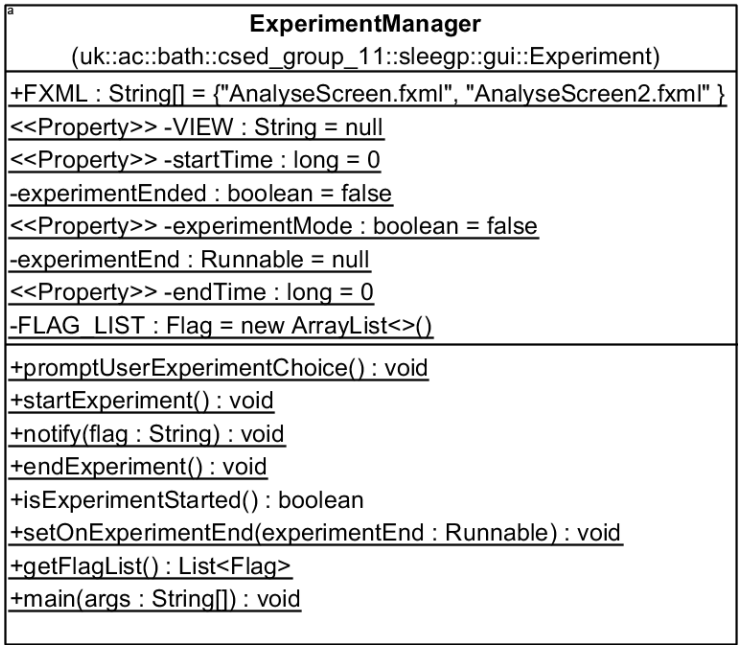


Figure 30:

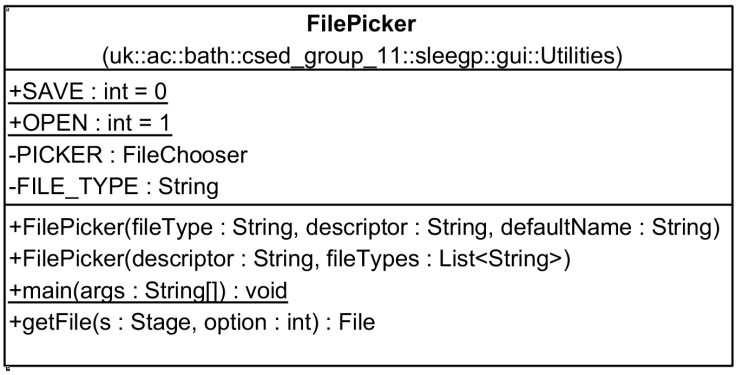


Figure 31:

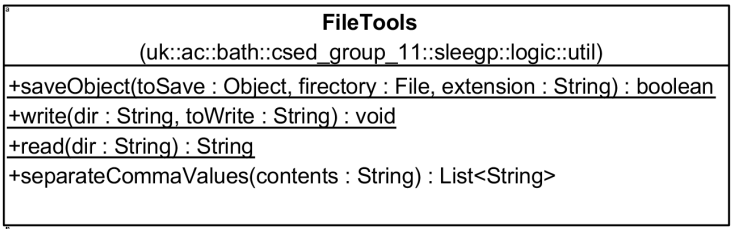


Figure 32:

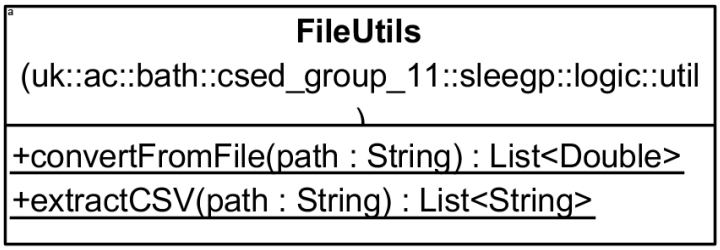


Figure 33:

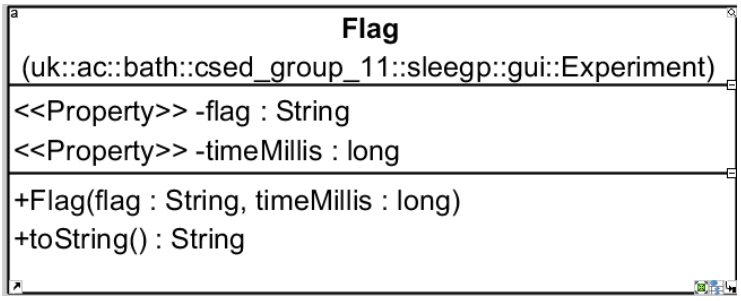


Figure 34:

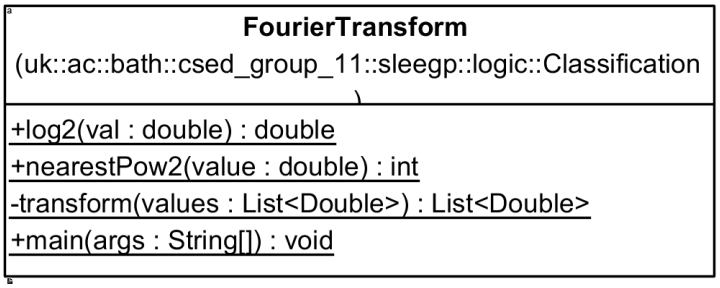


Figure 35:

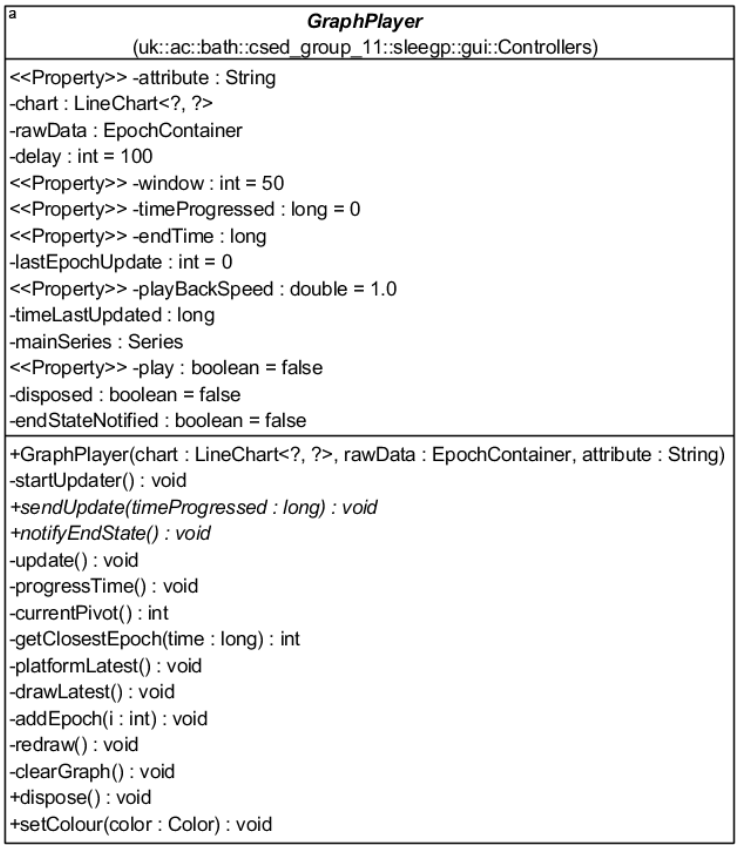


Figure 36:

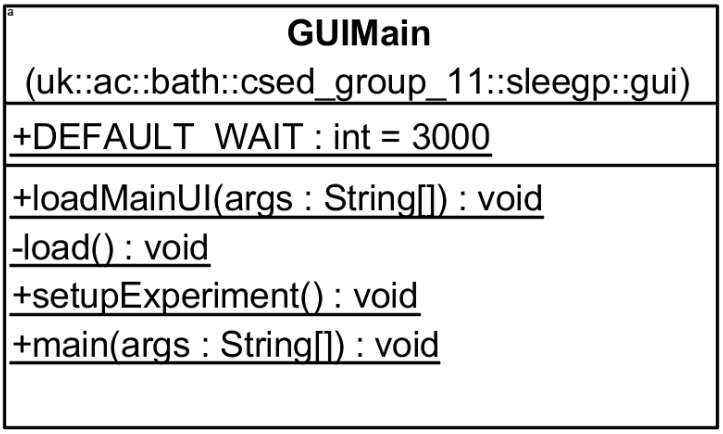

GUIMain
(uk::ac::bath::csed_group_11::sleegp::gui)
+DEFAULT_WAIT : int = 3000
+loadMainUI(args : String[]) : void
-load() : void
+setupExperiment() : void
+main(args : String[]) : void

Figure 37:

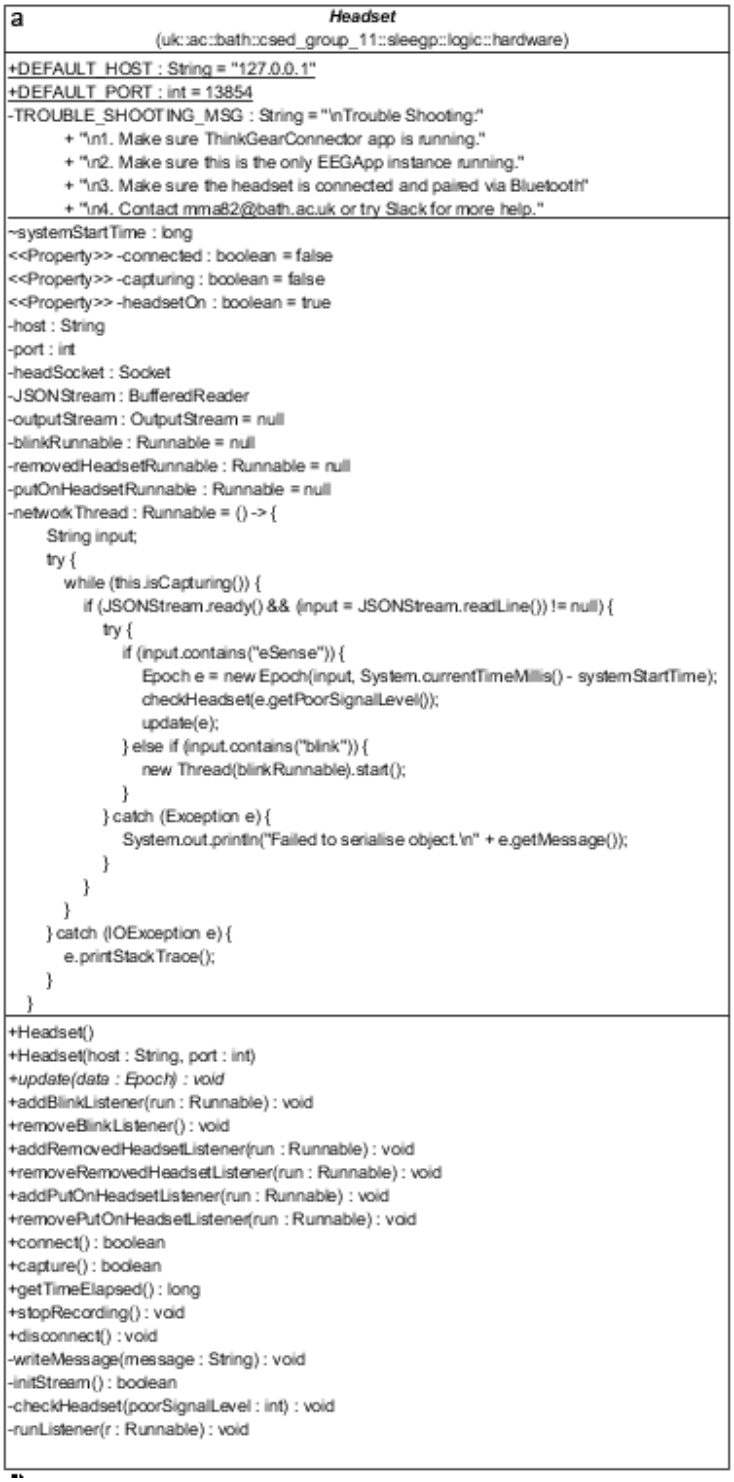

a
Headset
(uk::ac::bath::csed_group_11::sleegp::logic::hardware)
+DEFAULT_HOST : String = "127.0.0.1"
+DEFAULT_PORT : int = 13854
-TROUBLE_SHOOTING_MSG : String = "\nTrouble Shooting:"
+ "\n1. Make sure ThinkGearConnector app is running."
+ "\n2. Make sure this is the only EEGApp instance running."
+ "\n3. Make sure the headset is connected and paired via Bluetooth"
+ "\n4. Contact mma82@bath.ac.uk or try Slack for more help."
~systemStartTime : long
<<Property>> -connected : boolean = false
<<Property>> -capturing : boolean = false
<<Property>> -headsetOn : boolean = true
-host : String
-port : int
-headSocket : Socket
-JSONStream : BufferedReader
-outputStream : OutputStream = null
-blinkRunnable : Runnable = null
-removedHeadsetRunnable : Runnable = null
-putOnHeadsetRunnable : Runnable = null
-networkThread : Runnable = () -> {
String input;
try {
while (this.isCapturing()) {
if (JSONStream.ready() && (input = JSONStream.readLine()) != null) {
try {
if (input.contains("eSense")) {
Epoch e = new Epoch(input, System.currentTimeMillis() - systemStartTime);
checkHeadset(e.getPoorSignalLevel());
update(e);
} else if (input.contains("blink")) {
new Thread(blinkRunnable).start();
}
} catch (Exception e) {
System.out.println("Failed to serialise object.\n" + e.getMessage());
}
}
}
} catch (IOException e) {
e.printStackTrace();
}
}
+Headset()
+Headset(host : String, port : int)
+update(data : Epoch) : void
+addBlinkListener(run : Runnable) : void
+removeBlinkListener() : void
+addRemovedHeadsetListener(run : Runnable) : void
+removeRemovedHeadsetListener(run : Runnable) : void
+addPutOnHeadsetListener(run : Runnable) : void
+removePutOnHeadsetListener(run : Runnable) : void
+connect() : boolean
+capture() : boolean
+getTimeElapsed() : long
+stopRecording() : void
+disconnect() : void
-writeMessage(message : String) : void
-initStream() : boolean
-checkHeadset(poorSignalLevel : int) : void
-runListener(r : Runnable) : void

Figure 38:

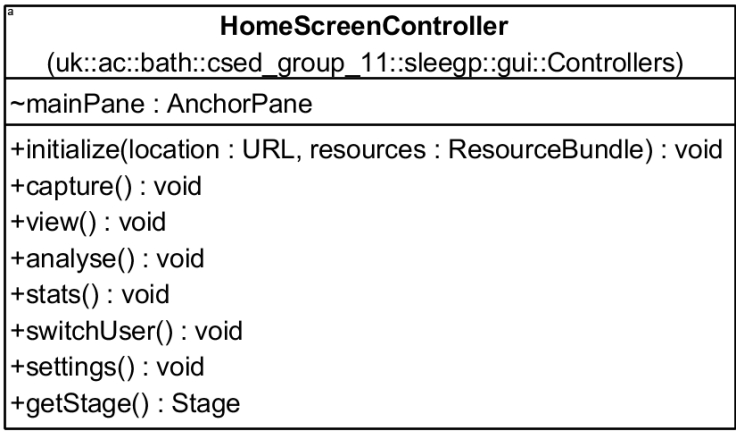


Figure 39:

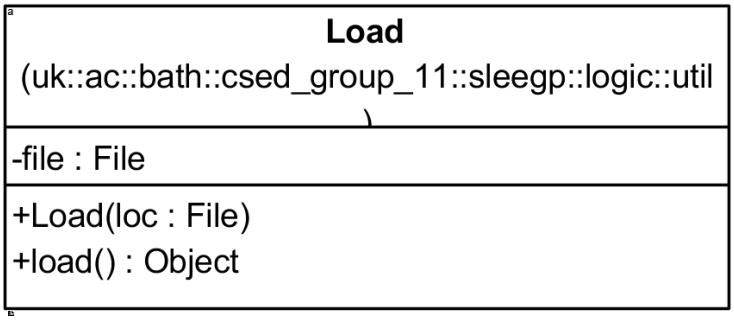


Figure 40:

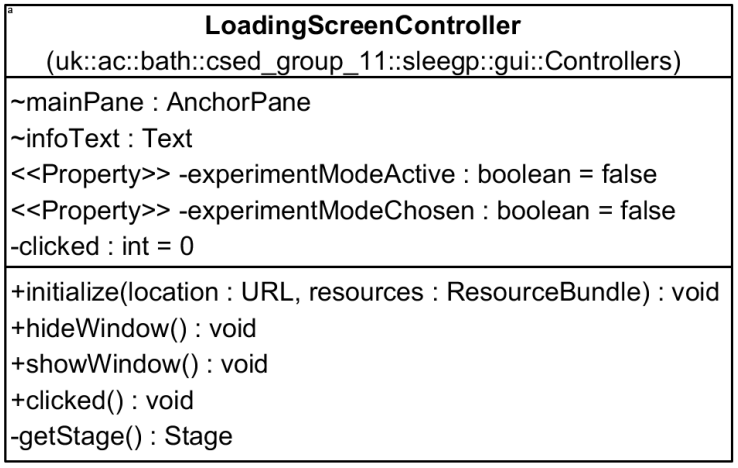


Figure 41:

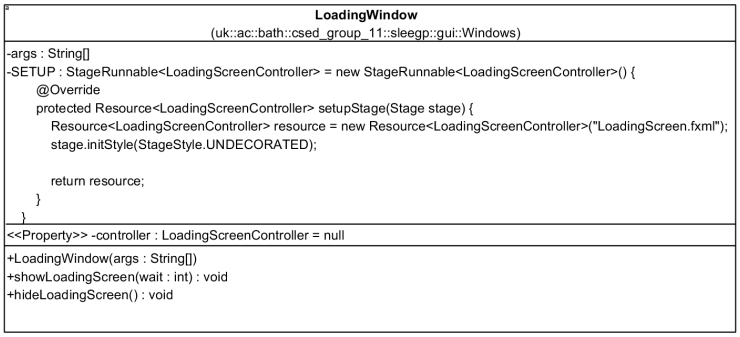


Figure 42:

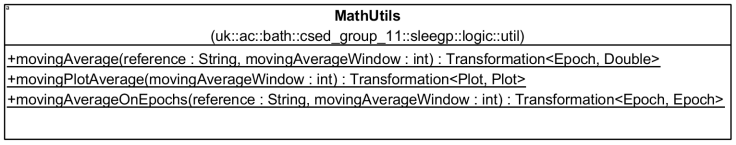


Figure 43:

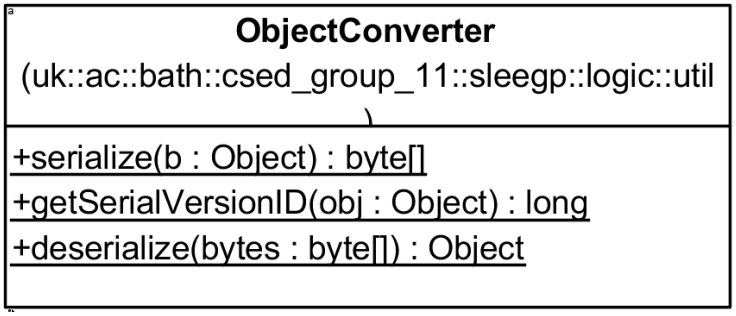


Figure 44:

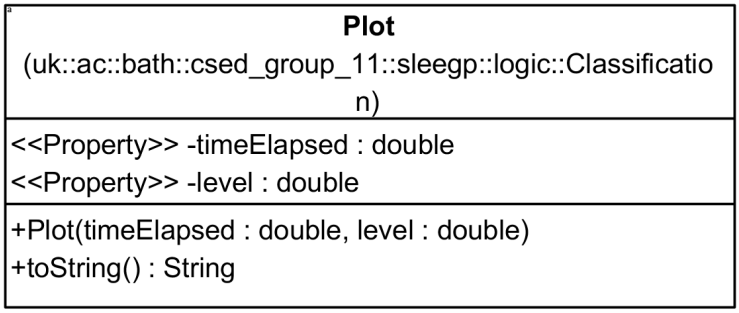


Figure 45:

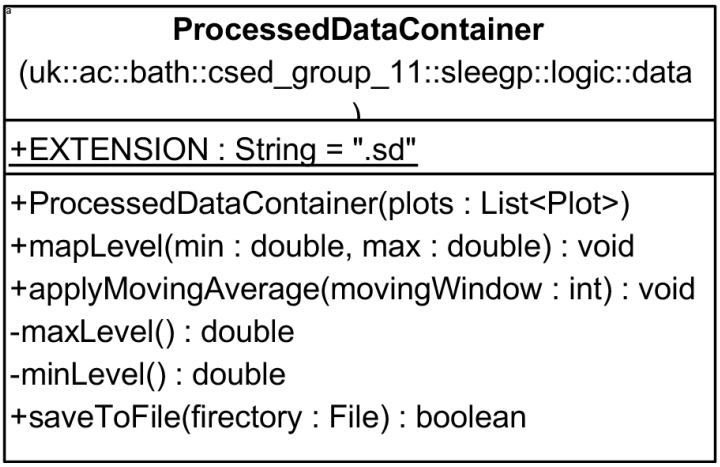


Figure 46:

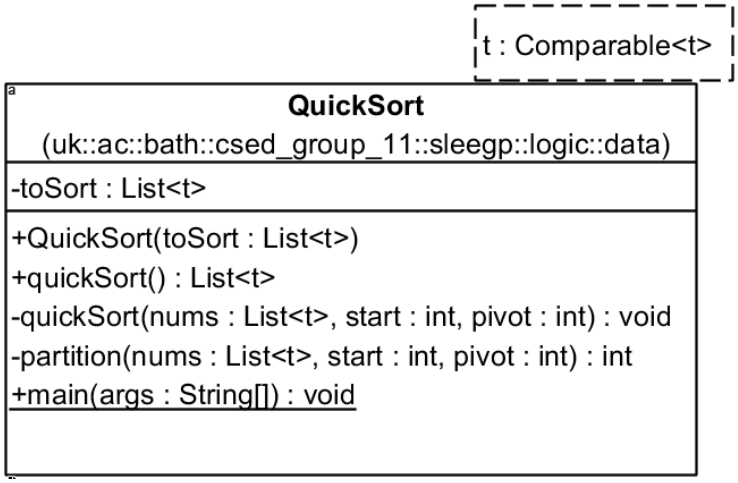


Figure 47:

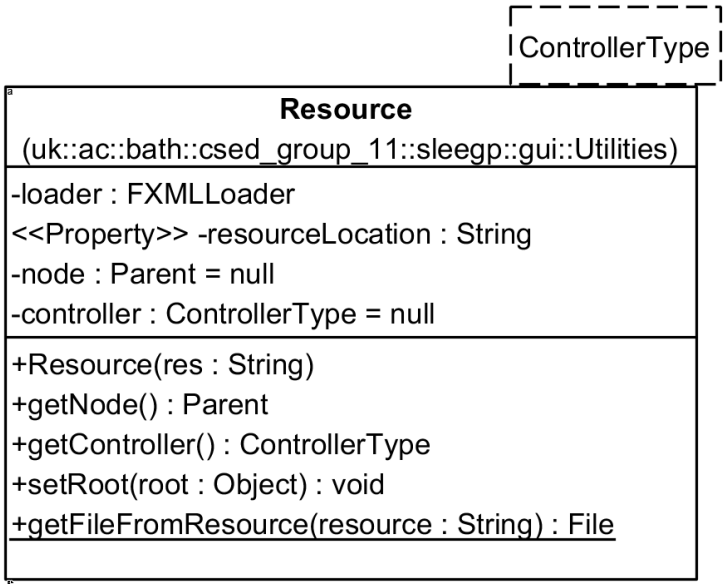


Figure 48:

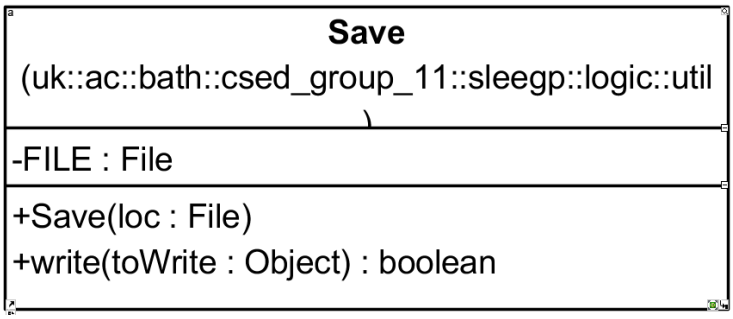


Figure 49:

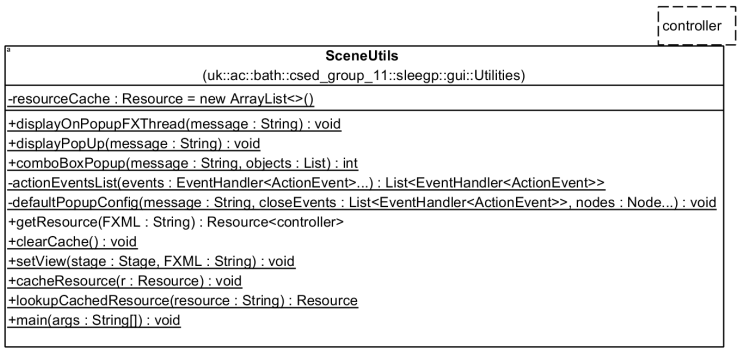


Figure 50:

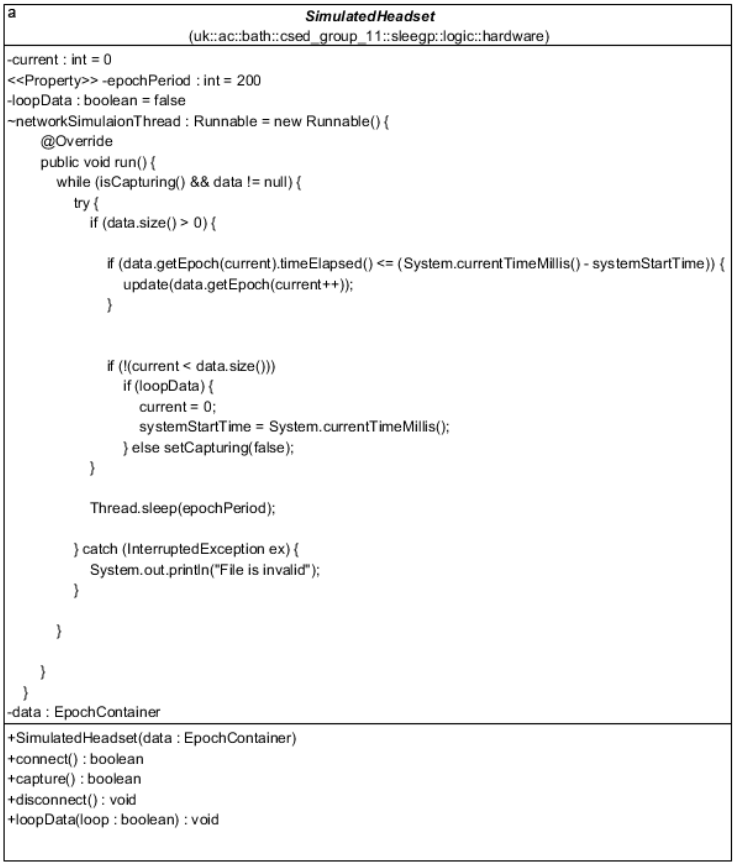


Figure 51:

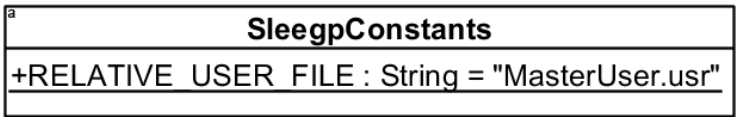


Figure 52:

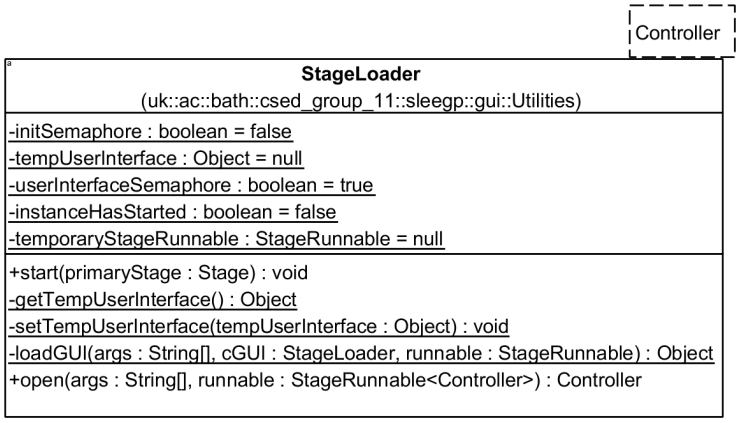


Figure 53:

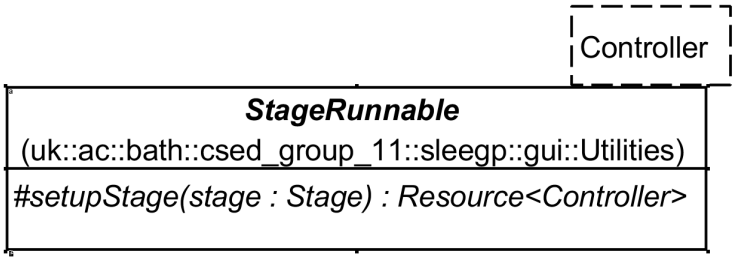


Figure 54:

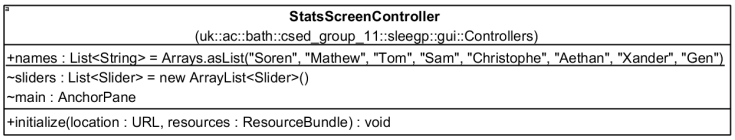


Figure 55:

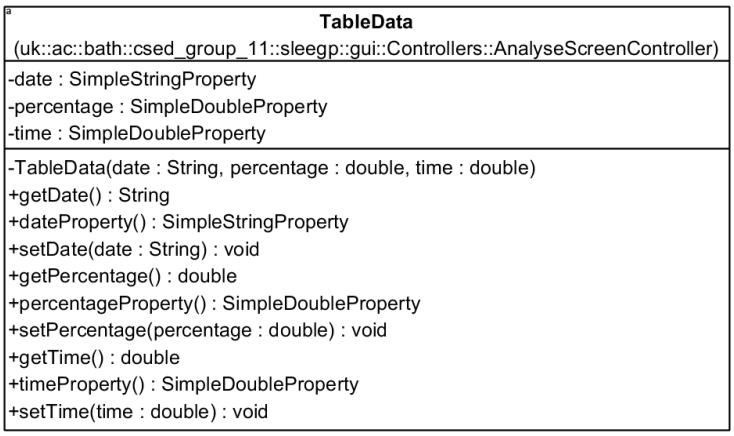


Figure 56:

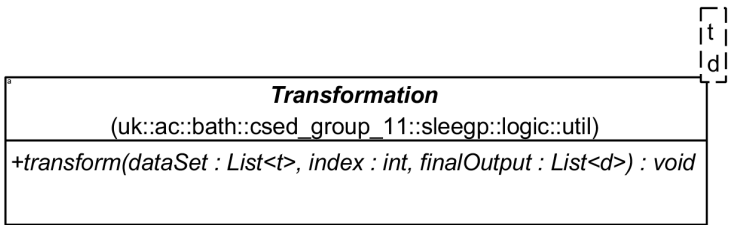


Figure 57:

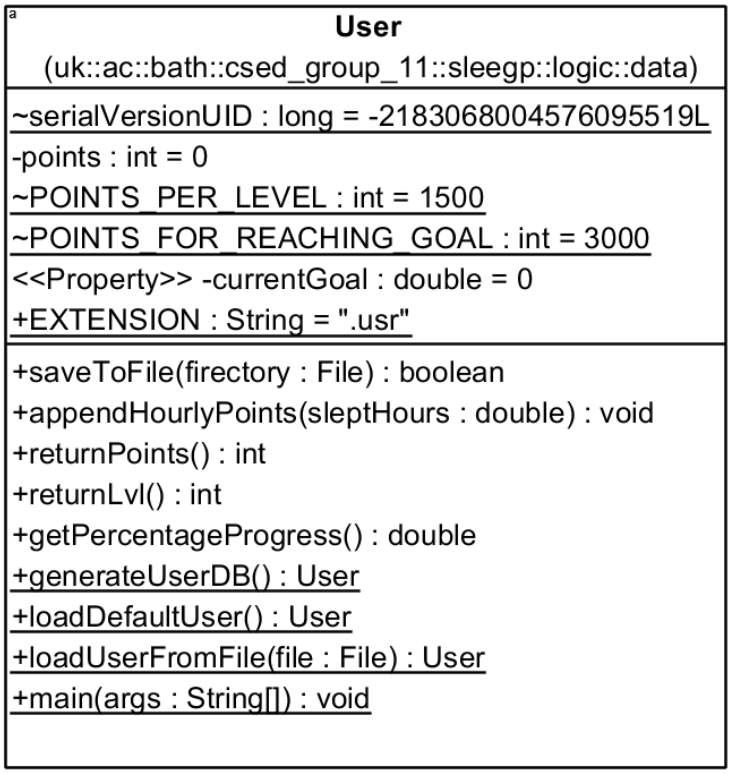

a
User
(uk::ac::bath::csed_group_11::sleegp::logic::data)
~serialVersionUID : long = -2183068004576095519L
-points : int = 0
~POINTS_PER_LEVEL : int = 1500
~POINTS_FOR_REACHING_GOAL : int = 3000
<<Property>> -currentGoal : double = 0
+EXTENSION : String = ".usr"
+saveToFile(firectory : File) : boolean
+appendHourlyPoints(sleptHours : double) : void
+returnPoints() : int
+returnLvl() : int
+getPercentageProgress() : double
+generateUserDB() : User
+loadDefaultUser() : User
+loadUserFromFile(file : File) : User
+main(args : String[]) : void

Figure 58:

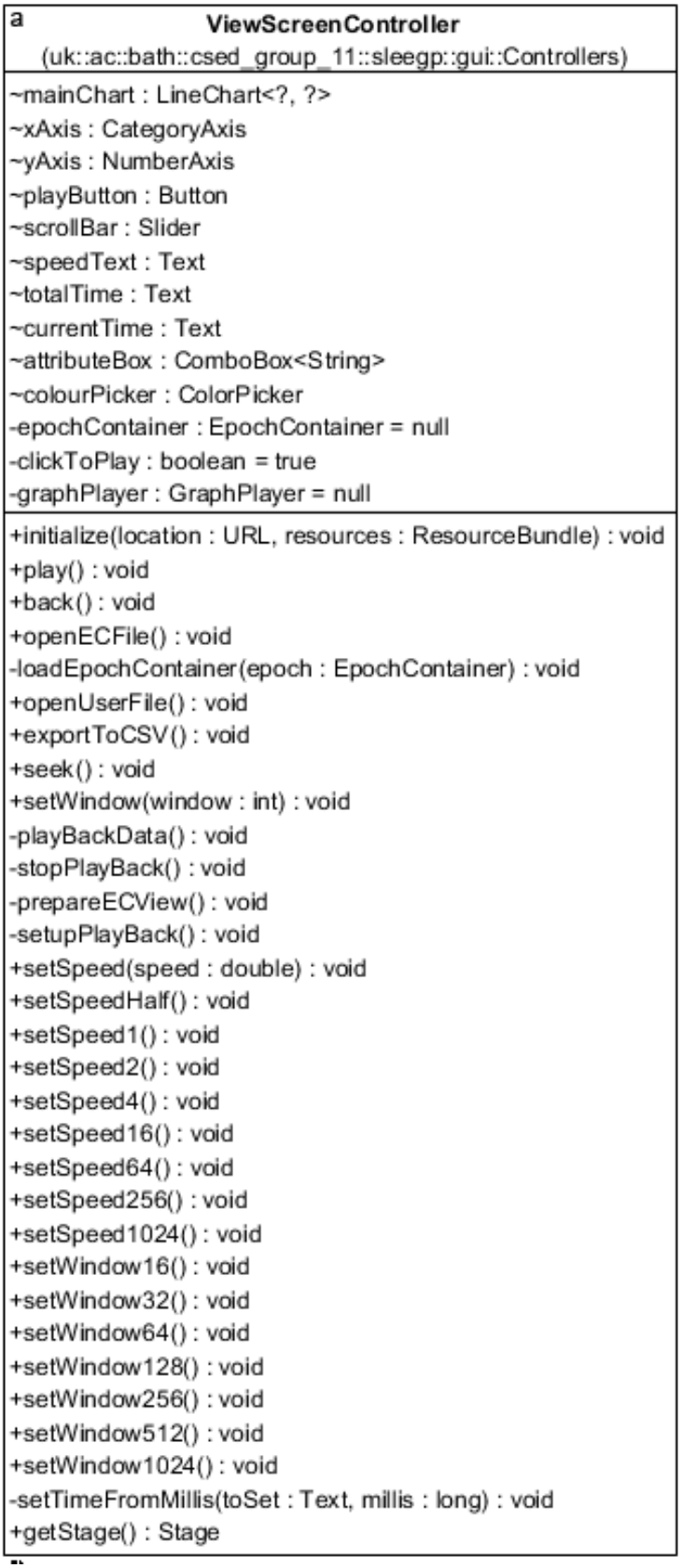

a
ViewScreenController
(uk::ac::bath::csed_group_11::sleegp::gui::Controllers)
~mainChart : LineChart<?, ?>
~xAxis : CategoryAxis
~yAxis : NumberAxis
~playButton : Button
~scrollBar : Slider
~speedText : Text
~totalTime : Text
~currentTime : Text
~attributeBox : ComboBox<String>
~colourPicker : ColorPicker
-epochContainer : EpochContainer = null
-clickToPlay : boolean = true
-graphPlayer : GraphPlayer = null
+initialize(location : URL, resources : ResourceBundle) : void
+play() : void
+back() : void
+openECFile() : void
-loadEpochContainer(epoch : EpochContainer) : void
+openUserFile() : void
+exportToCSV() : void
+seek() : void
+setWindow(window : int) : void
-playBackData() : void
-stopPlayBack() : void
-prepareECView() : void
-setupPlayBack() : void
+setSpeed(speed : double) : void
+setSpeedHalf() : void
+setSpeed1() : void
+setSpeed2() : void
+setSpeed4() : void
+setSpeed16() : void
+setSpeed64() : void
+setSpeed256() : void
+setSpeed1024() : void
+setWindow16() : void
+setWindow32() : void
+setWindow64() : void
+setWindow128() : void
+setWindow256() : void
+setWindow512() : void
+setWindow1024() : void
-setTimeFromMillis(toSet : Text, millis : long) : void
+getStage() : Stage

Figure 59:

C.35 Finite State Machine

This is our Finite State Machine Diagram

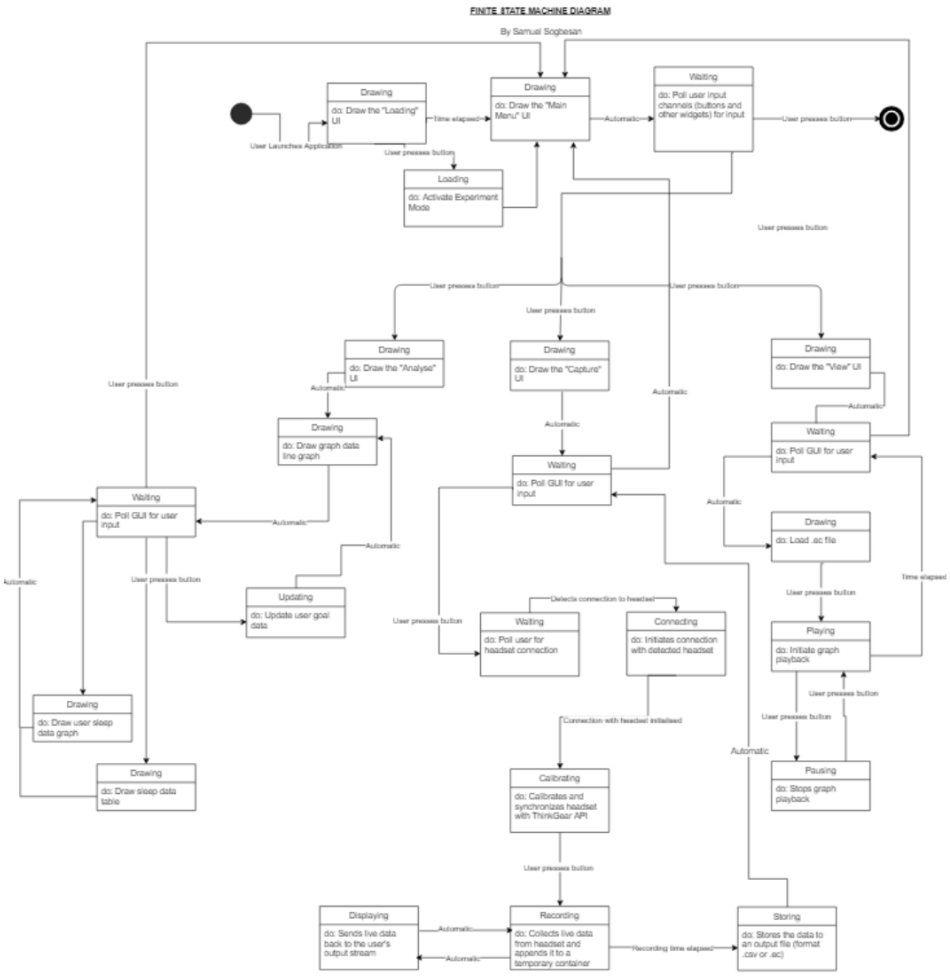


Figure 60: Finite State Machine