

CM10313 Tascii Web App, Tasciirs (Group 12)

Group member	Username	Degree Course
John Steward	js3511	MComp(hons) Computer Science + 1 year placement (year 1)
Sam Briggs	spb55	MComp(hons) Computer Science (year 1)
Tanapat Supatchapichai	ts2183	MComp(hons) Computer Science (year 1) + 1 year Placement
Daniel Ledger	dal34	BSc(hons) Computer Science (year 1)
Aman Aman	aa3264	BSc(hons) Computer Science (year 1)
Jayanth Vasanth Kumar	jvk24	MComp(hons) Computer Science and Artificial Intelligence + 1 year Placement (year 1)
Sharon Silver	ss3617	BSc(hons) Computer Science + 1 year Placement (year 1)
Henry Allen	hca32	BSc(hons) Computer Science (year 1)
Carolina O'Neill Marques	conm20	MComp(hons) Computer Science and Artificial Intelligence + 1 year placement (year 1)

Tascii

Authors: Sam Briggs, Daniel Ledger, John Steward, Carolina O'Neill Marques, Henry Allen, Tanapat Supatchapichai, Jayanth Vasanth Kumar, Sharon Silver, Aman Aman

Abstract

Our Personal Informatics system will be focused on how different types of music can affect a user's productivity. We will be able to see what sort of music each user is listening to through their Spotify, and the user will set productivity goals, for example, a to do list for the week, with tasks they hope to accomplish and a level of difficulty for each task so we can gauge how productive they have been. The difficulty of the tasks they complete will also mean they get more productivity points for their account, which they will be able to share with other people. This is an arbitrary score system that will contribute towards milestones that the user will be able to set for themselves, in order to get rewards like badges/trophies. If we were to further evolve our system, we would add 'workspaces' where users can share what music they have been listening to and stream that music with other users/co-workers. This would potentially allow users to share what music has particularly helped them be more productive, so they can help their co-workers be more productive as well.

Table of Contents

Abstract	1
Table of Contents	2
1. Introduction	4
2. Agile Software Process Planning & Management	6
3. Software Requirements Specification	8
3.1 Domain of the informatics system	8
3.2 Devising Initial requirements	8
3.3 Use Case diagrams	8
3.4 Requirements Table	9
4. Software Design	11
4.1 Data flow diagram	12
4.2 UML Class diagrams	14
4.3 Justification of chosen design	15
4.3.2 Create Account screen	16
Figure 2	16
4.3.4 Dashboard screen	17
4.3.5 Calendar screen	19
4.3.6 Tasklist screen	19
4.3.7 Spotify recommendation screen	20
Figure 6	20
4.3.8 Achievements/Badge library	21
5. Software Testing	23
6. Reflection & Conclusion	25
References	27
Appendices	28
Appendix A - Final GCF	28
Appendix B - Agile meeting minutes	29
Mt. 1	29
Mt. 2	30
Mt. 3	32
Mt. 4	33
Mt. 5	34
Mt. 6	35
Mt. 7	36
Mt. 8	37
Mt. 9	38

Mt. 10	39
Mt. 11	40
Mt. 12	41
Mt. 13	42
Mt. 14	43
Mt. 15	44
Mt. 16	45
Mt. 17	46
Mt. 18	47
Mt. 19	48
Mt. 20	49
Mt. 21	50
Mt. 22	51
Mt. 23	52
Appendix C - Questionnaire	53
Appendix D - UI design mockups	57
Desktop proposed designs	57
Mobile proposed designs	58
Appendix E - UML Diagrams	60

1. Introduction

Author: John Steward

A huge problem that students struggle with nowadays, especially in current times during the pandemic, is productivity. Since most students are having to work from home and learn online, they are finding it harder and harder to keep up with the workload. We are constructing a Personal Informatics system that will focus on the correlation between different types of music that people listen to while working, and productivity.

Our main stakeholders will be University students of similar ages to us, since they will get the most use out of a system like this. The main objective of our product will be to help those students to be more productive, and therefore improve their studies.

When first creating an account for our product, the user will be given a series of questions to better suit the app to them, such as if they want to set up their own goals or have the app automatically set points milestones for them. They will also be asked to create a to do list for each week, to gauge how productive they are, and will set a difficulty for each task so, if they complete a task with a high difficulty, they will get more points than if they completed a low difficulty task. They may then be able to share the points they received for the week and overall on a leaderboard so they can see their productivity level against friends or co-workers.

According to one article, many users found that they felt much more compelled to do better and get more points when they were comparing their results to the results of people that they met through the system they used, rather than comparing to close family and friends [1]. This is especially helpful in today's environment considering that most students have only met each other online, making it easier for them to compete against each other. Users found, however, that, with persuasive systems, they only cared about getting more points, not about what the points actually represented and only cared about reaching milestones, to the point that, with something like a FitBit, they would walk around their house at the end of the day to reach the next step milestone [1]. This could still be good, considering they are still getting the benefits of their exercise, but with our product, it could be detrimental as people may just be doing their work with the sole intention of reaching the next point milestone. For example, a student may be getting pre-recorded lectures done just to get more points, and not even thinking about the actual content of the lectures. This is unlikely, however, since studies are much different to something like exercise, and students know they need to keep the content they go through in their mind.

Studies have shown that listening to music while you work can significantly reduce stress[2], leading to increased productivity, since it is much easier to focus on what you are doing when you are not under a lot of stress.

It's not just whether you are or are not listening to music that affects your productivity. The main factors relating to music that affect productivity are the type/genre of music that you listen to, the duration of your listening session(the longer you are listening to music, the less stressed you will feel and therefore be more productive) and, most importantly, whether or not you are already familiar with the music that you are listening to while you are working [2].

If you are unfamiliar with the music that you are listening to, it is much more likely that you will be distracted by the music, therefore leading to reduced productivity as you will be more focused on the music than your work. If you are familiar with the music, it will be much easier to focus on your work, and it may even reduce stress further depending on your memories associated with a certain song or genre of

music. Therefore, we are considering making 'workspaces' within our product where you can share songs that have helped you to be more productive, and allow multiple people to stream the same music together while they work. This will allow users to pick which music they can stream alongside co-workers or fellow students that will help them to study and get work done easier.

However, a meta-analysis from 2011 stated that background music "disturbs the reading process and has a small detrimental effect on memory"[3], so in our case, it could potentially be detrimental to be listening to music while doing something like looking through lecture notes, however it will still be useful to do while doing coursework.

One study showed that people listening to background music with lyrics showed a significantly lower attention test score than those who were listening to background music without lyrics[4]. This means that students will be much more likely to be able to focus on doing their work if they are listening to music that does not have lyrics than if they listen to music with lyrics. Some studies also show that listening to background music alleviates restlessness and distraction in psychiatric patients and enables their activities to proceed more smoothly[4]. This further supports the theory that listening to music can help students with concentrating on their work, since reduced restlessness obviously means that it will be easier to focus on one thing, instead of getting distracted by other things.

Another recent study tracking software engineers has also shown that listening to music while working boosted their mood, in turn leading to better quality of work and productivity[5]. This means that listening to music doesn't necessarily directly improve productivity, but can improve the worker's mindset and mood which in turn improves their productivity, so it is getting indirectly improved by the music they listen to. This therefore means that whether productivity is improved still depends on the type of music you listen to. If you don't enjoy the music, or it lowers your mood, you may see a drop in productivity.

A study published in the Applied Cognitive Psychology Journal explored the differing effects of background music on introverts and extroverts. 10 of each were given a memory test that required immediate and delayed recall, as well as a reading comprehension test. It showed that, for both groups, music was detrimental for immediate recall, but introverts performed worse in the delayed recall and in the reading comprehension[6]. This shows that music can have a different effect on everyone depending on how their mind works, so different types of music can have a different effect on one person compared to another, and some may perform better when having no music at all. Our app will be tracking the different types of music people are listening to and how productive they are to go alongside that, so they will be able to get a grasp on what music positively and what music to avoid if it is detrimental to their productivity.

2. Agile Software Process Planning & Management

Author: Tanapat Supatchapichai

An agile software development process can be described as an iterative development, where requirements and solutions evolve through collaboration of the team [7]. This type of process focuses on teamwork and communication [8], and is used in the development of this project. The rest of this section will document how this project was carried out as an Agile process through how sprints were planned and tracked, and how Scrum meetings were used to evolve ideas and features for the software system.

An important aspect of an Agile process is to be able to be adaptable to change and progress incrementally [9]. The process of this project was made to be adaptable through planning and tracking sprints. Sprints are a short period of time where a team works to complete a predetermined amount of tasks [10]. First off, during the duration of the project, sprints are kept tracked through Trello, an online collaboration tool to organise tasks into lists. There are five lists that are used on Trello for this project: product backlog, sprint backlog, doing, testing, and done. Product backlogs are a list of requirements and features that will be implemented, and sprint backlogs are a list of requirements and features that will be developed for the current sprint. During each sprint, members of the team are able to pick out tasks that are on the sprint backlog to develop and test out so that the set of requirements set out for the sprint is completed. This allows flexibility for each team member, allowing them to work on tasks that they are more comfortable working on.

Scrum meetings are a meeting held by the team during a project's development, where features are discussed and sprints are planned [11]. The project's sprints are planned on a Scrum meeting, which is when the team meets up to discuss what features will be developed for the sprint. Before sprints started, the team met twice a week: Monday and Friday to discuss features and requirements that we wanted to see in the project. There, pre-sprint tasks were discussed and completed, including researching articles, conducting user requirements surveys, creating use cases, and setting up GIT, Docker, and IntelliJ to allow the team, which is only able to work together online, to work on the project. After the sprints started, the team decided to meet three times a week: Monday, Wednesday, and Friday. There are two types of Scrum meetings that occur weekly: check-in meetings, where the team gets together to discuss the progress of the current sprint and help out with any potential problems, and sprint meetings, where the team gets together to plan out what features will be developed in the upcoming sprint. Specifically, Mondays were typically used for sprint meetings, if a new sprint is about to start, while Wednesdays and Fridays are used for check-in meetings. With that being said, on all of these meetings, features are always discussed and improved. Through these meetings, the team is able to adapt to any problems that come up while incrementally progressing the project with each sprint.

Through these Scrum meetings, designs, ideas, and features are discussed and evolved to become better suited for the project. First of, designs of the project and use case were evolved through the discussion on them in meetings. After discussing the initial ideas and features that the team wants to include in the project, a member came up with a use case diagram for the project. During the next meeting, the team discussed the use case and decided to improve it by further detailing the login function for the application, which now includes what happens when a user creates a new user account. As for designs on the project, multiple members had come up with UI mockups and the application logo for the project. These UI mockups and logos were improved through an exchange of opinions between the members of the team on various things such as: color, shape, and the translation of the UI mockup onto an actual device. During the discussion on the UI mockups in one Scrum meeting, an idea was made to implement a system in place to allow users to change the theme of the application and lead to the creation of the system inside the

application. Another example of features being discussed and evolved from Scrum meetings is when a member suggested a feature to give users badges as a progression system for the application. From then, the idea was expanded with badge designs being created and critiqued, and badges being implemented into the application as a result.

3. Software Requirements Specification

Authors: Henry Allen (Writeup) & Everyone (Requirements Table)

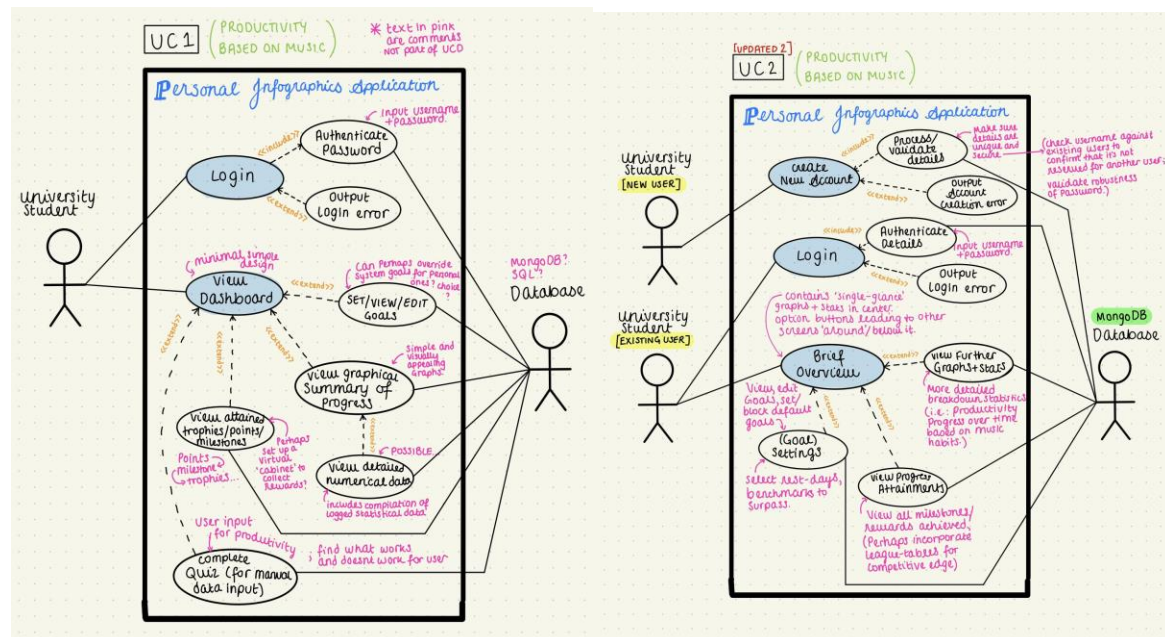
3.1 Domain of the informatics system

As established in the introduction to this report, our goal was to devise a system to allow target users (University Students) to measure their productivity against the kind of music they listen to. We determined that users should be able to track this information in some form, and the system should highlight potential relationships between them.

3.2 Devising Initial requirements

Part of our surveying process included questions about what sort of ways data should be tracked, and how this should be shown to a target user (See Appendix C). This included graph types, types of lists, and potential use of different metrics such as milestones and badges. Our surveys allowed us to create a loosely defined set of initial features, represented by the use case diagrams provided below. In continuation, we then also created several UI mock-ups, loosely defining the organisation of the system through the different 'screens' users would see. This proved very useful, allowing us to group client-side requirements under which screen they would be used for (See requirements 1.0 – 1.5); this made the requirement list much clearer for the front-end developers and also made delegation of these tasks more efficient. Different front-end developers in the group could work on separate screens to greatly speed up the development process. Requirements for the back-end followed a similar process, and were devised with the front-end in mind. These mostly consist of various endpoints for the client-server API, as well as managing the database (schema, etc.).

3.3 Use Case diagrams



3.4 Requirements Table

Functional Requirements		
1.0	Requirement Name: User Dashboard	Author: Jayanth
	<u>Description:</u> The system is to display a user interface to allow the user to track their progress and set tracking goals for the future	Priority: High(3)
1.1	Goals	Author: Jayanth
	<u>Description:</u> The system allows the user to set goals that they wished to be tracked and thus improve upon.	Priority: Medium(2)
1.2	Quiz for manual input and user needs	
	<u>Description:</u> An initial follow through page for the user to answer questions for personalization.	Priority: Low(1)
1.3	Accolades Tab	Author: Jayanth
	<u>Description:</u> An achievements page to give users motivation in their journey.	Priority: Medium(2)
1.4	Progress Summary	Author: John
	<u>Description:</u> A detailed progress summary for the user to analyze their improvement.	Priority: Medium(2)
1.5	Trends based on user activity	Author: Tanapat
	<u>Description:</u> Trends on the tasks the user does along with their music taste to give a better idea of their productivity.	Priority: Medium(2)
2.0	User Login	Author: Daniel,Sam
	<u>Description:</u> Over all system requirements for building a user login service on the client and server side.	Priority: High(3)
2.0.0	Database Schema	Author: Sam
	<u>Description:</u> The initial database design for the system.	Priority: High(3)
2.0.1	Endpoints	Author: Sam

	<u>Description:</u> A URL to which requests can be made for various server side utilities	Priority: High(3)
2.0.1.1	Create Task Endpoint	Author: Sam
	<u>Description:</u> An endpoint to create a task in the database	Priority: High(3)
2.0.1.2	Edit Task Endpoint	Author: Tanapat
	<u>Description:</u> An endpoint to edit task details in the database.	Priority: High(3)
2.0.1.3	Delete Task Endpoint	Author: Aman
	<u>Description:</u> An endpoint to delete tasks in the database.	Priority: High(3)
2.0.3	Constant Rest output	Author: Daniel
	<u>Description:</u> Converts any java object into a consistent JSON string.	Low
2.0.4	Client-Basic Rest Reader	Author: Daniel
	<u>Description:</u> Converts the JSON body into a object for use in React	Priority: High(3)
2.0.5	Client Auth Service	Author: Daniel
	<u>Description:</u> OAuth service to check client details.	Priority: Medium(2)
2.1	Password Verification	Author: John
	<u>Description:</u> An authentication process is initialised after the user inputs their required login details.	Priority: Medium(2)
2.1.0	Client Login Page Mock up	Author: Henry, Sharon
	<u>Description:</u> Client page design which the Front-End Development is based on	Priority: Medium(2)
2.1.1	Client Login Page	Author: John
	<u>Description:</u> The Login page of the web app through which the user may verify their credentials.	Priority: High(3)
2.2	Login Verification	Author: John
	<u>Description:</u> Check the login details with database.	Priority: Medium(2)
2.3	Create a New Account Page	Author: John, Dan
	<u>Description:</u> Create a new user account, taking in username, email and password	Priority: Medium(2)
3.0	Task Overview Page	Author: Henry

	Description: Gives an overview of the tasks in a pie chart form	Priority: Medium (2)
3.0.1	Task list	Author: Henry, Carolina
	Description: The list of tasks that need to be done	Priority: Medium (2)
3.1	Calendar	Author: Sharon
	Description: Gives the due date of tasks in a calendar grid	Priority: Low (1)
Non - Functional Requirements		
1.0	Security	Author: Sam, Aman, Tanapat
	Description: The app provides branching level of security with the OAuth service as well verification on the main page. The security is prioritised to keep the user data safe and is kept in mind through each stage development.	Priority: High(3)
2.0	Responsiveness	Author: Carolina, Jayanth, Henry
	Description: Each of the components on the front end are designed to provide a fast response on user clicks to give a smooth experience to the user. On the backend the services are in tune with the front end to allow for quick data transfer as well.	Priority: High(3)
3.0	User-Friendly UI	Author: Daniel, Sharon, John
	Description: A significant portion of the Development time was put into the UI design to give a comfortable user experience. The UI has been developed to be simple and effective	Priority: Medium(2)

4. Software Design

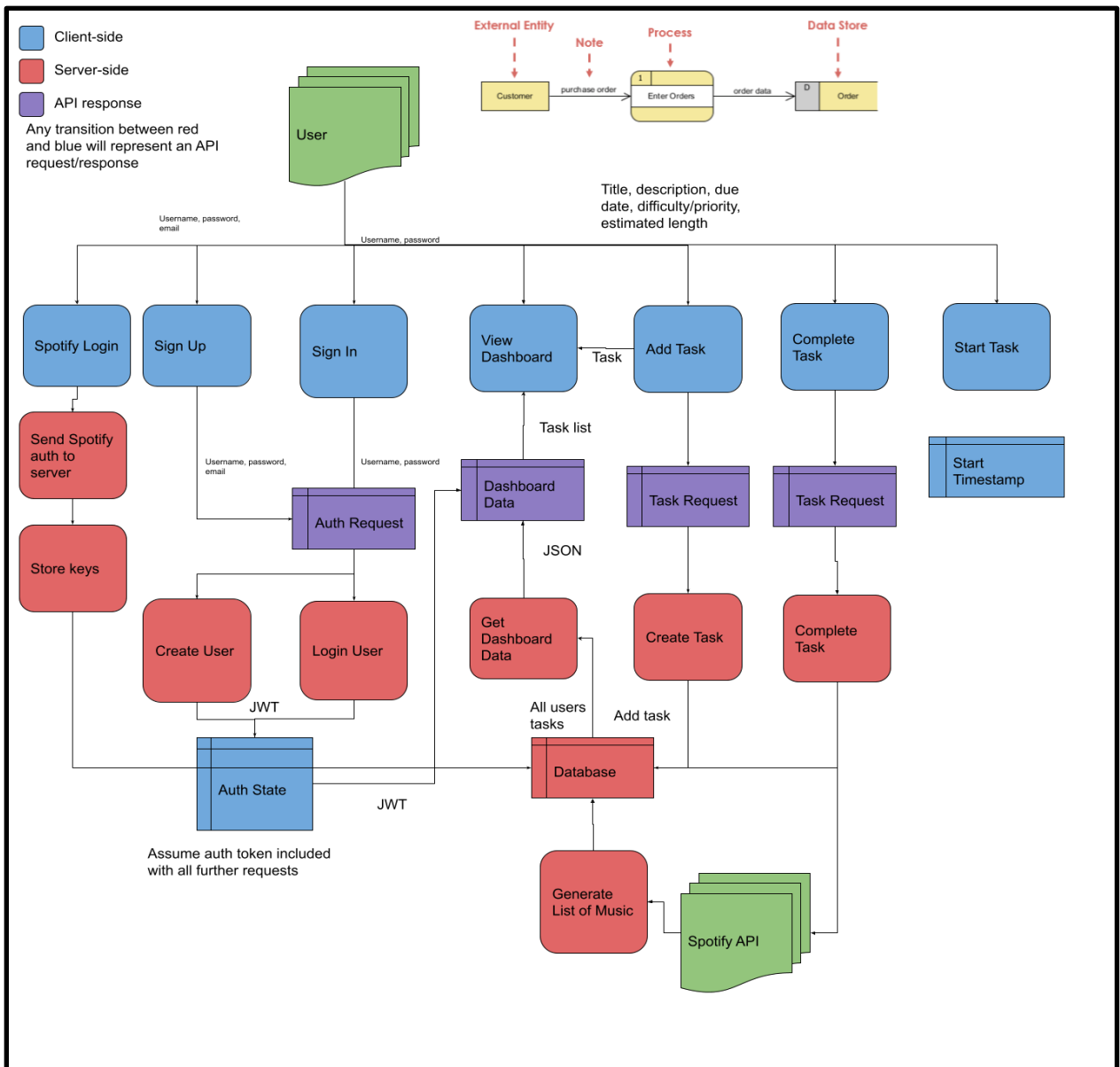
Authors: Jayanth Vasanth Kumar & Daniel A. Ledger

To ensure a smooth transition between the propositions of raw ideas from group members to the method of development of the software, a robust design phase must be completed. This 'bridging phase', so to speak, allows us to collectively visualise what our application can *actually* do through a variety of diagrams. From the flow of data throughout the application between the clients and the server, to the database schema design in which the data is stored, including the class structure of components and processes following OO principles.

4.1 Data flow diagram

The first design diagram to showcase is the data flow diagram (DFD). Data flow diagrams are predominantly used during software development for the ability to view the inputs and outputs between each 'entity' in the model. It is also worthwhile to note that DFDs do not usually contain no conditions, control flows or any form of iteration [12].

We decided to choose this diagram to represent the intermediary level of design for our application, allowing us to view the transfer of data between the different blocks of abstracted processes in both the server and client side. For instance, the credentials being passed through from the user (via the client side) as they create an account to the login handlers on the server side, and the authentication state with the appropriate server response. The diagram is shown below:



Compared to a typical 'high-level' data flow diagram, our style includes the API response from the server, returning data for each respective given request from the client side. This would prove to be largely beneficial to all members of the group (given the split of the groups to work on server and client side processes and components), which would allow for everyone to view each and every endpoint between the server and client side, as well as the data being passed through, therefore allowing everyone to work on their aspect of the project seamlessly between each sprint (and Git merge to the main 'master' branch).

4.2 UML Class diagrams

The Object Oriented Programming (OOP) paradigm is an incredibly important style of programming which was employed in our project in order to maintain and preserve collections of data through abstracted objects. This data is encapsulated within the objects (essentially a blueprint, of sorts) along with in-built and predefined methods to manipulate it accordingly.







We have incorporated classes and an OOP structure throughout a majority of the flow of the code, especially in the Java and server-side part of the software. This will improve reusability - as objects bind, often unique, data within separate instances which can then in turn be individually referenced or manipulated. The use of classes here eliminates any code repetition to attach data and methods to what are essentially 'variables'. Furthermore, classes ensure both security and ease of maintenance, since the data for each object is bound within the created instance of that object (does not lead to data interference between instances). This also means that since methods are bound to the classes (from which instances are created from), further readability is ensued as methods can be called to perform processes on the created instances themselves.

The next set of pages contain a list of class diagrams under each respective category within which they operate.

[\(See Appendix E for UML Diagrams\)](#)

The above set of UML class diagrams form the bulk of the designed structure of the server-side Java written to manage the majority of the server endpoints, database processing as well as responding to any requests sent from the client side (i.e. storing inputted data for newly created tasks, input validation for login credentials etc).

Here is a brief explanation of the icons and symbols used in the diagrams:

Icon/Symbol	Meaning
	Reference to a class
	Reference to a field or attribute in a class (a gray diamond shape to the bottom left indicates a static field)
	Reference to a method or function in a class (a gray diamond shape to the bottom left indicates a static method)
	Reference to a Spring annotation (Spring uses 'dependency injection' to configure and bind the application, declaring the components is enough for Spring to piece everything together at runtime)
 or 	Reference to access modification - <i>private</i> and <i>public</i> respectively. Used on either methods/functions or attributes/fields in a class, and defines the level of access given to instances of classes to methods and attributes

We have focused on an Object Oriented model involving *high cohesion* and *low coupling* [13], meaning that we rely on classes being defined mostly for a unique purpose, while maintaining distance between other classes (reducing interdependence) such that there would be a reduction, or better yet, an elimination of object and instance dependencies. Following from this, (and the famous saying: 'Composition over Inheritance' [14] stemming from the hugely popular book: '*Design Patterns*' (1994)), we implemented composition more so than inheritance. With Inheritance, a given class will 'inherit', so to speak, the methods and attributes (those which are accessible from the parent class), giving the child class the ability to access, use and override them as required. While this may have its own advantages, an extension of a child from a parent via inheritance will likely result with a tightly coupled system; changes in a parent class with children could risk breaking the system and relationship. Conversely, composition with association and aggregation does not increase coupling, since there is no distinct defined relationship between any classes. Modular creation of classes to complete different processes and tasks, passing objects as parameters, would lead to a system which is loosely coupled and highly cohesive.

4.3 Justification of chosen design

Below, we will discuss the justification of each page design individually, explaining how we believe they have met both the functional and non-functional requirements of our specification, each will contain the reference to the requirement that it satisfies as well as why it satisfies it.

4.3.1 Login screen

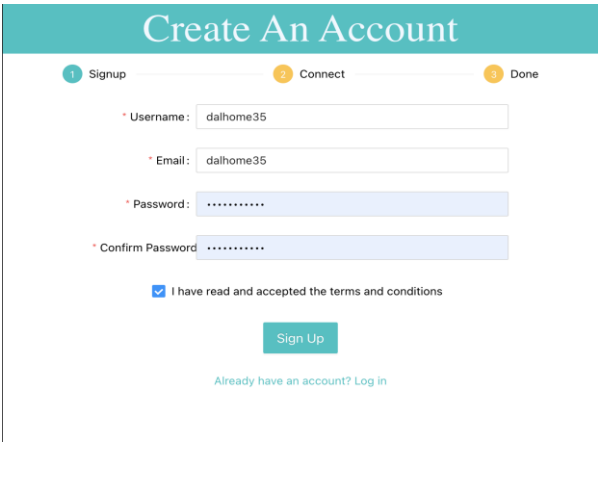
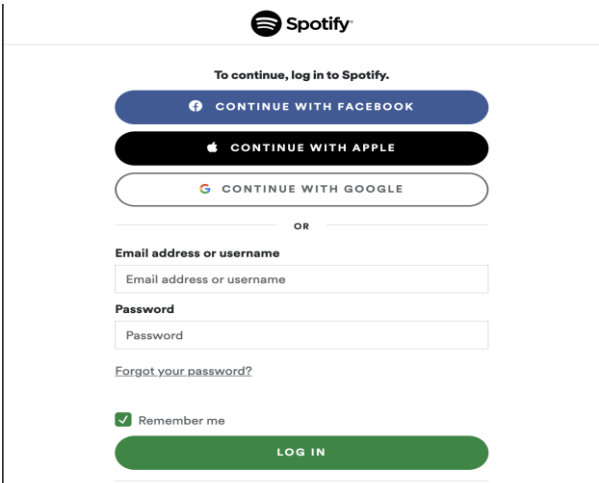
Figure 1

 <p>Log In</p> <p>* Username: dalhome3</p> <p>* Password:</p> <p>Log In</p> <p>Don't have an account? Create one</p>	 <p>Log In</p> <p>Username or Password Incorrect</p> <p>* Username: test</p> <p>* Password:</p> <p>Log In</p> <p>Don't have an account? Create one</p>
---	--

Component	Description	Requirement satisfied
Login credential entry	This component allows for entry of the login credentials (username and password), following the structure of the proposed design mockups (<i>Refer to appendix C for the mockups</i>)	2.1.0 & 2.1.1
Login credential verification	Functional aspect, allowing the user entered credentials to be validated on the server side (compared against the database), and displaying appropriate messages to inform the user (when incorrect or when fields are left empty)	2.2

4.3.2 Create Account screen

Figure 2

 <p>Create An Account</p> <p>1 Signup 2 Connect 3 Done</p> <p>* Username: dalhome35</p> <p>* Email: dalhome35</p> <p>* Password:</p> <p>* Confirm Password:</p> <p><input checked="" type="checkbox"/> I have read and accepted the terms and conditions</p> <p>Sign Up</p> <p>Already have an account? Log in</p>	 <p>Spotify</p> <p>To continue, log in to Spotify.</p> <p>CONTINUE WITH FACEBOOK</p> <p>CONTINUE WITH APPLE</p> <p>CONTINUE WITH GOOGLE</p> <p>OR</p> <p>Email address or username</p> <p>Password</p> <p>Forgot your password?</p> <p><input checked="" type="checkbox"/> Remember me</p> <p>LOG IN</p>
---	--

Component	Description	Requirement satisfied
Create account, credential entry	This component allows the user to create a new account, taking in the username, email and password twice. Details are verified, and sent to the server (<i>Refer to appendix D for the mockups</i>)	2.3
Spotify account pairing	Additional step in the create account process involves pairing the user's spotify account. This is done by redirecting the user to the spotify web 'portal' to enter the user's details.	Sets up foundation for 1.5

4.3.4 Dashboard screen

Figure 3

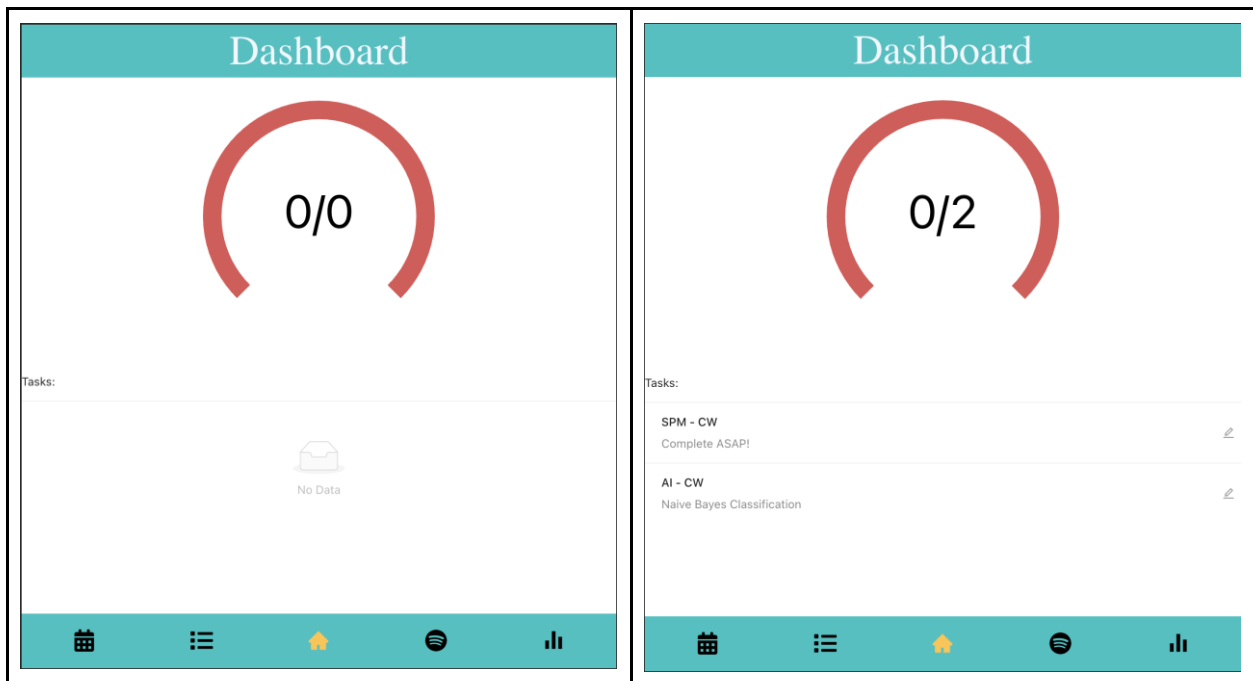


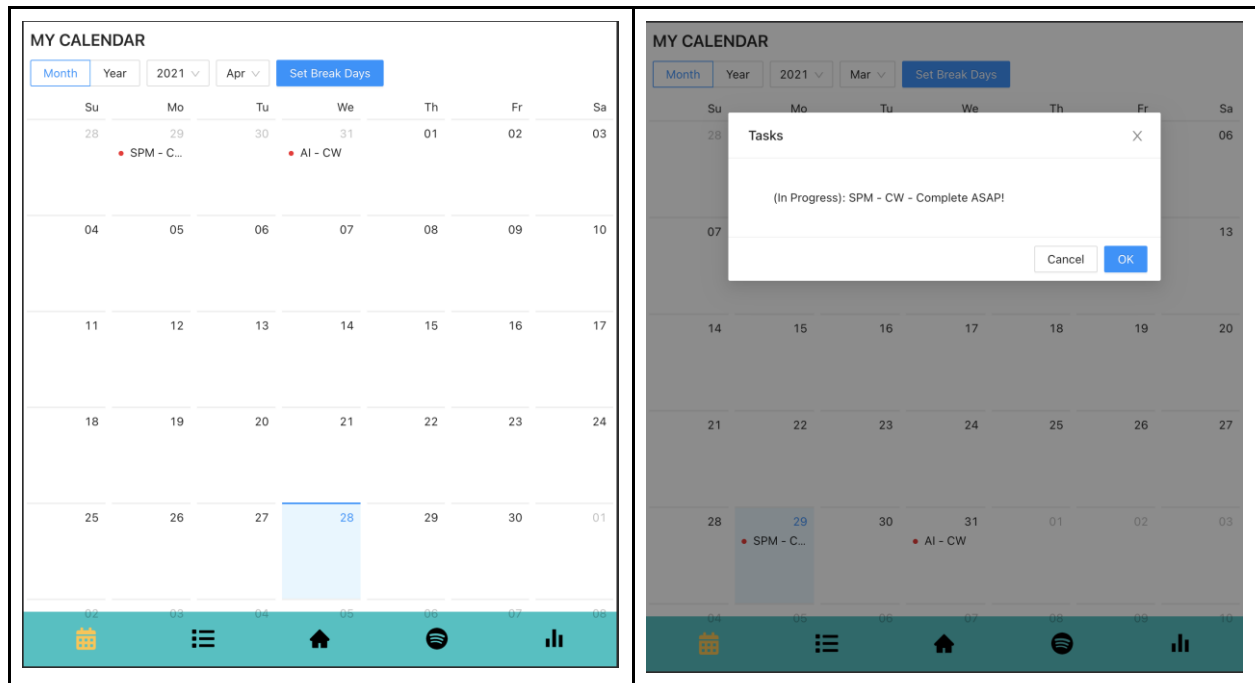
Figure 1 above illustrates our proposed design in its final stages of production for the dashboard screen.

Component	Description	Requirement satisfied
Circular progress bar	This component is central to the dashboard, being one of the first things the user sees upon logging in. The progress bar essentially fills up as more tasks are completed (along with the total tasks completed in the middle).	1.0

Current taskset to complete	This component is visible below the progress bar and displays a list of the currently set tasks to be completed	1.0
-----------------------------	---	------------

4.3.5 Calendar screen

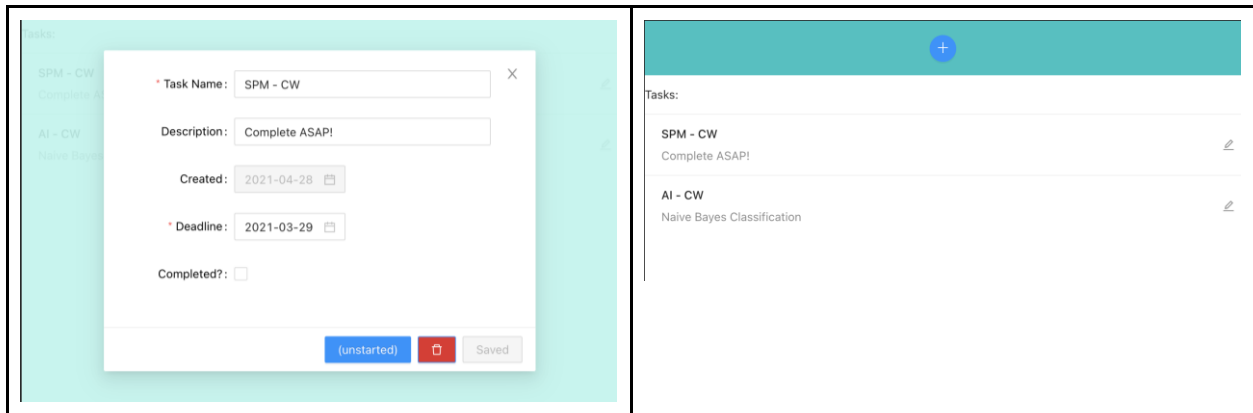
Figure 4



Component	Description	Requirement satisfied
Calendar grid	This component consists of a whole calendar grid configurable to be able to change the scope of the time period (such as month or year layout). Includes the tasks visible on the day, together with a color coding system relative to how near it is to the current date	3.1
Zoom-in feature	An addition to the calendar grid implemented is the ability to view, in greater size, the tasks on a particular day, by clicking on a day.	3.1

4.3.6 Tasklist screen

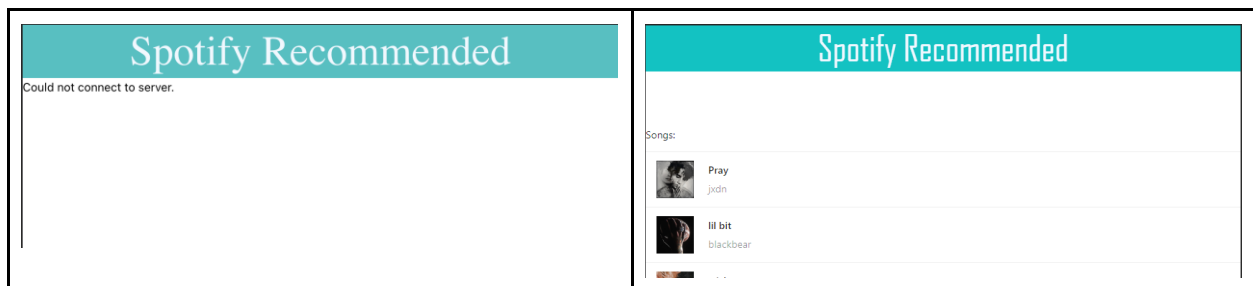
Figure 5



Component	Description	Requirement satisfied
Tasklist entry	Component allows the ability to add a new task, given the title, description and due date via a popup 'modal' entry window. A small scale calendar pops up to ease date entry to simply a single click	3.0.1
Tasklist	Component also displays the currently created tasks in a list format. The tasks can also be edited further along the line, if the user wishes, updating this change in the calendar too.	3.0.1

4.3.7 Spotify recommendation screen

Figure 6

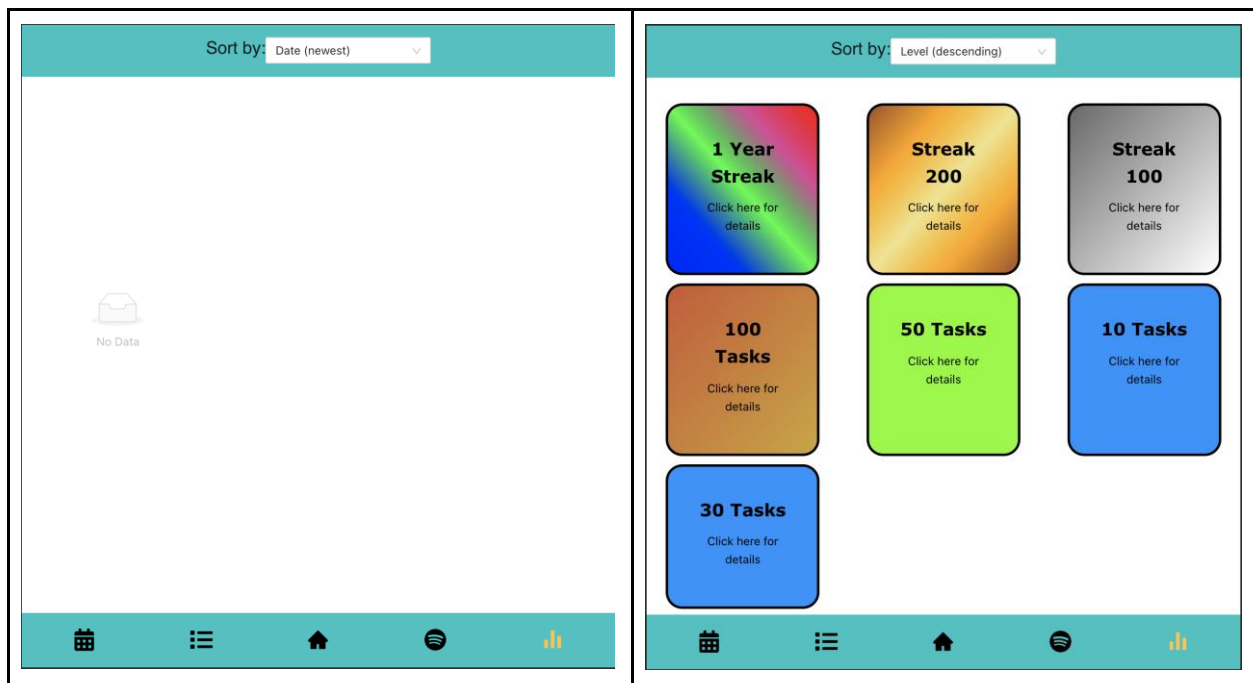


Component	Description	Requirement satisfied
Spotify song recommendations	Component displays a dynamic list of song recommendations, with some influence from their workflow. The song recommendation system is also a	1.5

	way to encourage the user to try out new and popular songs. Ideally generate a user-tailored set of songs which optimises their workflow.	
Spotify website link	Each song recommendation contains a thumbnail, title and song artist, which can additionally be clicked to lead the user directly to the spotify page for that particular song.	1.5

4.3.8 Achievements/Badge library

Figure 7



Component	Description	Requirement satisfied
Accolades tab	Component displays a library of achievements/badges earned. Badges can be opened up further to reveal the description and the date earned. The color of the badge is reflective of the priority level.	1.3
Accolades tab	Addition of sorting, which allows the list of badges to be sorted based on the date and priority level.	[Addition]

Apart from the functional requirements, an important non-functional requirement which we focused largely

on was the user interface (UI) and user experience (UX) [**Requirement 3.0**]. Developing software with polished themes and a simple, yet practical design making it more likely that the user will continue to use the app, without any changes or additions from a functional standpoint. Users of software naturally tend to stick with the application if it performs their needs and requirements within a concise and simple package; too complex of a design may risk losing users as it may ultimately lengthen their workflow unnecessarily. Our design focused on a bottom navigation bar layout, allowing the user to choose the screen to go to by simply clicking the icon associated to the screen on the bar, with the icon color highlight indicating the currently chosen screen. In order to ensure uniformity and consistency between all screens and pages, design components (such as drop down menus, pop up modals, entry fields etc) are all imported from the React UI framework, Ant Design [15], as well as screens following a common color scheme and palette matching well with the rest of the design components.

5. Software Testing

Authors: Aman Aman & Daniel Ledger

5.1 Test Cases Table

	Test Scenario	Test Case	Requirement	Pre-Conditions	Test Steps	Test Data	Expected Result	Actual Result	Pass/Fail
1	Check Endpoints	Verify that the /get endpoint works properly	3.2.2	Set up Unit tests using Spring Testing	1. Create a random task 2. Get one random task 3. Check validity of get function: One task as well as many tasks	Random user data created in database			Pass
		Verify that the /delete endpoint works properly	3.2.3	Set up Unit tests using Spring Testing	4. Check if the random task is editable 5. Check if the task can be deleted. (Repeat tests to check invalidity as well.)	Random user data created in database	Remove task from the Database	Sets the remove task to null	Fail* (Now Fixed)
		Verify that the /create endpoint works properly	3.2.1	Set up Unit tests using Spring Testing		Random user data created in database			Pass
2	Auth Test	Login Valid	2.0	1.Create Data Store 2. Get URL 3. Setup a random string generator.	1. Create a random user and add it to the data store. 2. Test if the user is present. 3. Test if the user logged in/ signed up.	Generate a random user.	Access token and refresh token > 0	Access token and refresh token > 0	Pass
		Login Not Exist	2.0			Random username and password.	"No user Found"	"No user Found"	Pass
		SignUp Valid	2.0			Random username, password and email.	Returns a user present in the database	Returns a user present in the database	Pass

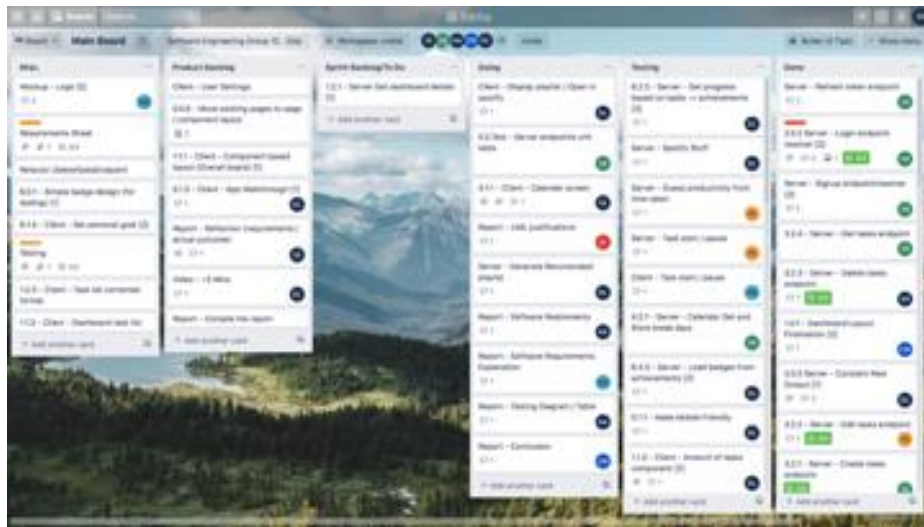
		SignUp Already Exists	2.0			Generate a random user.	"Username or email already exists."	"Username or email already exists."	Pass
3	JWT Test	Token generation	NF 3	1. Create a jwt and set the required data. 2. Create a random generator to use in testing	1. Generate a string. 2. Generate a token from the generated string. 3. Check if token validation returns true. 4. Repeat three times.	A token is generated in the class.	TRUE	TRUE	Pass
4	Data Tests	Achievements	1.4	1. Set Up a data store. 2. Find a user. If null, create a new one. 3. Instantiate a random object.	1. Get current time for creation time. 2. Create a new Achievement object. 3. Perform update and insert the new achievement into the achievements array. 4. Return okay with a new task. 5. Repeat test again.	Random Achievement object	"Okay" + New task	"Okay" + New task	Pass
		Milestones	1.4	1. Set Up a data store. 2. Find a user. If null, create a new one. 3. Instantiate a random object.	1. Get current time for creation time. 2. Create new Milestones objects. 3. Perform updates and insert the new milestone into the milestone array. 4. Return okay with a new task. 5. Repeat test again.	Random Milestone Object	"Okay" + New task	"Okay" + New task	Pass

6. Reflection & Conclusion

Author: Sharon Silver

Before starting the project, we researched to find out what the user wants, we surveyed to find out what metrics our target audience would be interested in monitoring. Our findings lead us down the path of monitoring how the genre of music you listen to can affect your productivity. We then collected the user's most desired features to create a use case diagram. Here we thought about the different types of users using our app and what each would expect to get out of this. We then used this use case diagram to form the basis of our product backlog to help us design future sprints.

Aside from the planning of what we were going to do, we also set up a good system on how we would do these things efficiently. We set up a docker environment so that everyone would be able to run the code. We set up a shared GitHub repository so that we could all work on the project at the same time and review each other's code. We hosted 2-3 meetings every week to update the team on our progress and any blockers we might be facing. We also had a Trello board so that we could monitor everyone's progress and the progress of everyone in general. The Trello board also meant that our meetings were very efficient and it made it very easy to choose what tasks we wanted to do and move things from the product backlog into sprints.



Before each sprint, we held a meeting to discuss which tasks from the product backlog would be included in our sprint. With each sprint, we aimed to have a usable piece of software that the user would be able to see, interact with and give us feedback on. For example, when planning the first sprint, we chose to have a fully working login and signup page which would direct you to an empty dashboard. This particular sprint was to meet the user requirement to log in and sign up for a new account. This meant that we were able to get feedback from the stakeholder on the usability of our login and sign up page. We also set times for how long each sprint would be ahead of time depending on how long we would estimate each task to take. We did stick to the sprint times but some tasks did end up being carried over to the following sprint, whilst some tasks were completed earlier than expected.

At the start of each sprint, everyone in our teams would claim the task which they wanted to do. Our Trello board made this very easy. We would work on the task or tasks we chose for the duration of that sprint. Due to the component layout of our project and using API's to communicate with our database, it was easy to separate tasks into self-contained pieces for everyone to work on individually. At the end of each sprint, someone had the task to combine these components into pages. We would update the team on the progress of our claimed tasks in the group meetings and on the Trello board. Testing happened during each sprint of the features we were making as we were making them, allowing us to catch things that may break much earlier than waiting until the end. The continuous communication we had during each sprint allowed us to make faster progress than we otherwise would have as we were able to receive help, ask for amendments with other people's code (e.g. a server-side API that hadn't yet been written) in a very timely manner. We didn't just use the meetings to communicate, we also frequently sent messages in our discord chat.

After each sprint, we made sure that we received feedback from our stakeholders. We selected a few people in our target demographic (students) and asked them about what they think about the feature(s) we have completed. This allowed us to ensure that we were on the right track to creating a product that the stakeholders would find beneficial. It also allowed us to evolve our user requirements to be more in line with what the user wants. We also discuss the tasks we have completed and the ones we didn't and move these tasks onto the next sprint before selecting new tasks from the product backlog to add to the next sprint. We also discuss features we think we should add to the product backlog based on both what the stakeholder has stated and things that we noticed during development ourselves. We had to keep a balance between what was realistic for us to finish and what features the user wanted. Keeping our initial user requirements simple often allowed us to have the capacity to take on more.

In terms of the evolution of the requirements, our frequent conversations with the stakeholders allowed this to happen. For example, our app was able to show trends between the Spotify genres that the user listened to and their productivity (monitored through to-do lists), but it wasn't until going through this feature with a stakeholder that we felt this requirement could be improved (within the time we had to complete the project). This is when we were encouraged to add a feature that created Spotify playlists based on the genre which inspires the most productivity for them so that they could listen to it without having to create their own playlists based on the advice we were given.

My team was quite lucky not to face many challenges. Despite the project taking place completely online, due to our constant communication, this never ended up being a challenge. One main initial challenge we faced was that not many people have worked with react before starting this project, but this ended up not being an issue as more experienced members of the team were very supportive and were willing to help as soon as possible so that we were never stuck on anything for too long.

If we had the opportunity to do this project again, we would probably add more features, be more adventurous and complete more sprints. As everyone is now familiar with the technologies we would be able to hit the ground running compared to the learning curve that most team members had to go through to familiarise ourselves with the new technologies.

References

- [1] Fritz et al. (2014), Persuasive technology in the real world: a study of long-term use of activity sensing devices for fitness
- [2] Teresa Lesiuk (2005), The effect of music listening on work performance
- [3] Zaria Gorvett (2020), Does music help us work better? [online] BBC Worklife, Available from: <https://www.bbc.com/worklife/article/20200317-does-music-help-us-work-it-depends>
- [4] Yi-Nuo Shih et al. (2009), Background music: Effects on attention performance
- [5] Kemmis, S., 2019. *The Science of Music and Productivity*. [online] Zapier.com. Available from: <https://zapier.com/blog/music-and-productivity/> [Accessed 28 Apr. 2021].
- [6] Furnham, A. and Bradley, A., 1997. Music While You Work: The Differential Distraction of Background Music on the Cognitive Test Performance of Introverts and Extraverts. [online] Available from: https://www.researchgate.net/publication/230286985_Music_while_You_Work_The_Differential_Distraction_of_Background_Music_on_the_Cognitive_Test_Performance_of_Introverts_and_Extraverts [Accessed 28 Apr. 2021].

----- Agile Software Process Planning and Management -----

- [7] Cprime. 2021. *What is AGILE? - What is SCRUM? - Agile FAQ's* | Cprime. [online] Available at: <https://www.cprime.com/resources/what-is-agile-what-is-scrum/> [Accessed 28 April 2021].
- [8] Agile Alliance |. 2021. *What is Agile Software Development?*. [online] Available at: <https://www.agilealliance.org/agile101/> [Accessed 28 April 2021].
- [9] PRESSMAN, and Bruce R. Maxim. *ISE EBook Online Access for Software Engineering: a Practitioner's Approach*, McGraw-Hill US Higher Ed ISE, 2019. ProQuest Ebook Central, <https://ebookcentral.proquest.com/lib/bath/detail.action?docID=5989441>.
- [10] Rehkopf, M., 2021. *Sprints* | Atlassian. [online] Atlassian. Available at: <https://www.atlassian.com/agile/scrum/sprints> [Accessed 28 April 2021].
- [11] Anon, n.d. *What is a Scrum Meeting? | Definition, Overview, and Examples*. [online] Productplan.com. Available from: <https://www.productplan.com/glossary/scrum-meeting/> [Accessed 28 Apr. 2021].

----- Software Design -----

- [12] Wikipedia (2021), Data Flow diagrams, [Online] Available from: https://en.wikipedia.org/wiki/Data-flow_diagram (DFD introduction and descriptions) [Accessed 25/04/21]
- [13] Medium (2020), Coupling and Cohesion, [Online] Available from: <https://medium.com/clarityhub/low-coupling-high-cohesion-3610e35ac4a6> (Description and examples of coupling and cohesion) [Accessed 26/04/21]
- [14] Wikipedia (2021), Composition over Inheritance, [Online] Available from: https://en.wikipedia.org/wiki/Composition_over_inheritance (Composition vs. Inheritance explanation, examples in some common programming languages) [Accessed 26/04/21]
- [15] Ant design documentation [Online] Available from: <https://ant.design> (Ant Design home page and information from)

Appendices

Appendix A - Final GCF

Name	Contributions
John	10
Sam	10
Tanapat	8
Daniel	10
Aman	8
Jayanth	8
Henry	10
Carolina	8
Sharon	8

Appendix B - Agile Meeting Minutes

Mt. 1

date: 12/02/2021

time: 9:15am

attendance:

Carolina O'Neill Marques

John Steward

Sam Briggs

Tanapat Supatchapichai

Daniel Ledger

Aman Aman

Jayanth Vasanth Kumar

Sharon Silver

Henry Allen

report filled by: Carolina O'Neill Marques

issues discussed:

introduction of team members (name, CompSci experience,...)

what OOP to use ? → Java

what subject ? → sleep → main subject

→ with → productivity

→ coffee consumption

→ mood

assignment of the main articles to read

next meeting

actions: read the articles to gather information

article 1 <https://dl.acm.org/doi/10.1145/2556288.2557383>: john and sam

article 2 <https://dl.acm.org/doi/10.1145/3264924> : aman and jay

article 3 <https://dl.acm.org/doi/10.1145/2851581.2892401> : carolina, sharon and tanapat

article 4 <https://www.tandfonline.com/doi/full/10.1080/0144929X.2018.1436592> : daniel and henry

planned date: Wednesday

planned time: 14:00

planned platform: Discord

Mt. 2

date: 17/02/2021

time: 14:00

attendance:

Carolina O'Neill Marques
John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen

report filled by: Carolina O'Neill Marques

issues discussed:

- 1st article (John and Sam): User goal setting
 - negative obsession → dependency of our PI
 - incentive (badges, trophies, rewards, activity points) is important → otherwise users see no point on doing things
 - representation of data is important → might influence users without actually being totally true
 - manual user input = burden
 - users set their own milestones/goals
 - compare yourself to other users similar to yourself = motivation (even more so if they meet their peers) (ex: club leaderboards, compare to the average)
 - give the users many options for goal presentation and setting
 - think about difficulty to insert goals and data → difficult = demotivation
- 2nd article (Henry and Daniel): representation of data to users
 - visualisation
 - neat and pretty → likable
 - not raw data but intuitive data (graphs, abstract...)
 - multiple data is the way to go
 - correlation between data interest people
 - data history important → see a pattern
 - give users options → personalisation
 - simple/complex visualization
 - more/less data
 - ⇒ should we do one of them or both? maybe one
 - people need to understand/identify with their data
- 3rd article (Carolina, Sharon and Tanapat): finding interesting correlations in multifaceted PI systems
 - users find correlations interesting
 - see changes

- surprising → not obvious
- positive
- shouldn't filter correlation only by users goals as users like to be surprised
- give advice to achieve goals?
- turn negativity into positive or question form

- 4th article (Aman and Jay):
 - positive feedback → users feel better and search deeper
 - negative feedback → users feel worse
 - users believe system more than themselves
 - ask people what they want to better
 - sleep
 - mood
 - stress

conclusion for our PI :

- milestones
- user profiling
- users should see their progress → graph with timeline with colour modification →

animation ?

establishment of a Trello and a GitHub space

next meeting

actions: think about questions to put on survey to gather requirements

planned date: Friday

planned time: 9:15 am

planned platform: MS Teams

Mt. 3

date: 19/02/2021

time: 9:15

attendance:

Carolina O'Neill Marques
John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen

report filled by: Carolina O'Neill Marques

issues discussed:

questions for survey

Whether or not people wanted special things tracked or general points?

Do you prefer (list of graphs)?

listing different things we thought about tracking

Did you know that ... affects ...?

What are you currently struggling with?

Which motivates you more, lead/group statistics or your own personal goals?

Do you prefer positive or negative ...?

How important would you rate personalisation ?

Add a feature where we give advice to better yourself

next meeting

actions:

do use cases (all)

do survey (Carolina)

think about what the actual requirements are (all)

What kind of form would our PI take? (all)

presentation of different architectures (Dan and Sam)

planned date: Wednesday

planned time: 14:00

planned platform: Discord

Mt. 4

date: 24/02/2021

time: 15:45

attendance:

Carolina O'Neill Marques
John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen

report filled by: Carolina O'Neill Marques

issues discussed:

survey results analysis:

bar charts

positive reinforcement

productivity and music → manually fill in & spotify API

rewards for specific goals and you own

motivates → specific goals and leaderboards

personalisation is really important

get advice for better your bad habits

get rewards every time their achieve a goal

reward system → all of them : Milestone, badges, points, trophies, streak

do a to do list → to calculate percentage of productivity

→ allow to put the difficulty of a task → weights more

→ percentage of tasks done → streak of productive day

→ schedule rest days (like weekend) → pause streak to support mental health

use cases

→ in a separate docs

discussion about possible architectures and approaches

→ summary in a separate docs

→ do mock-ups

→ client-server based

next meeting

actions:

ask the extend of OOP to the tutor

do use cases based on features with decided

planned date: Friday

planned time: 9:15

planned platform: MS Teams

Mt. 5

date: 26/02/2021

time: 9:15

attendance:

Carolina O'Neill Marques
John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen

report filled by: Carolina O'Neill Marques

issues discussed:

explain our idea to Olivia
review on some use cases (in separate doc)
suggest songs based on mutual interest between users: ex. User A likes music X and user B like music X so if user A likes music Y suggest music Y to user B
decided on all implementations we will use (also in separate doc)

next meeting

actions:

continue to do use cases based on features with decided
start reading some articles about our correlation

planned date: Monday

planned time: 14:00

planned platform: Discord

Mt. 6

date: 01/03/2021

time: 14:05

duration: 2hr 30m

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen
Carolina O'Neill Marques

report filed by: Henry Allen

Actions:

- Setup git; refer to presentation if you need to go over installing again or if you haven't set up an SSH key. Make sure you can successfully pull.
- Setup docker; check first run with localhost:8080 & 3000. Refer to presentation again for details.
- Setup IntelliJ; Again check for a first run from intellij, same as above.
- Everyone with the exception of Jay done these - Jay ran into some errors, hopefully can get fixed before Wednesday :)

For next time:

- See Trello, individual tasks assigned last time are listed there. Use cases for most.

Planned date: Wednesday

Planned time: 14:00

Planned platform: Discord.

Mt. 7

date: 03/03/2021

time: 13:10

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen
Carolina O'Neill Marques

report filed by: Henry Allen

Actions:

- Looked over the start of the report from John.
- Defined requirements - see requirement table & trello.
- Began adding to product backlog

For next time:

- Dataflow diagrams (Friday)
- Final edits to use cases.
- Add more to product backlog if necessary.

Planned date: Friday

Planned time: 9:15

Planned platform: MS Teams

Mt. 8

date: 05/03/2021

time: 9:15

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen
Carolina O'Neill Marques

report filed by: Carolina O'Neill Marques

Actions:

- Data flow diagram
 - In this url <https://docs.google.com/drawings/d/1v8G-9nodtmvA07TETGxoMGYy773OtJ45Dx5ISIUdF1Y/edit>

For next time:

- UI Mockups (Sharon and Henry)
- Suggestions for users (Carolina)
- Finish Intro (John)

Planned date: Monday

Planned time: 14:00

Planned platform: Discord

Mt. 9

date: 08/03/2021

time: 14:00

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Sharon Silver
Henry Allen
Carolina O'Neill Marques

report filed by: Carolina O'Neill Marques

Actions:

- Discussion about UI Mockups that are in drive
- Discussion about schema outline located here
https://docs.google.com/document/d/1425obwJJHylqJ9-6xBOJw0bOOEzzQ_iOs2UeF76INsc/edit
- Intro finished by John and read
- App pages :
 - Login page
 - Workspace page
 - Progress page
 - Dashboard
- Mobile app
- Assigned tasks of this sprint to each member in Trello
- Created a new Git branch called development

For next time:

- Trello tasks (all)

Planned date: Wednesday

Planned time: 14:00

Planned platform: Discord

Mt. 10

date: 10/03/2021

time: 14:00

duration: 15 min

attendance:

John Steward

Sam Briggs

Tanapat Supatchapichai

Daniel Ledger

Aman Aman

Jayanth Vasanth Kumar

Henry Allen

Carolina O'Neill Marques

(Sharon Silver is sick)

report filed by: Carolina O'Neill Marques

Actions:

- Discussed where we are on our tasks, discussed our individual progress on the tasks
- Viewed the Dashboard Mockups

For next time:

- Continue our tasks

Planned date: Friday

Planned time: 9:15

Planned platform: MS Teams

Mt. 11

date: 12/03/2021

time: 9:15

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver

report filed by: Carolina O'Neill Marques

Actions:

- Discussed were we are on our tasks with Olivia
- Doing a test was suggested

For next time:

- Finish our tasks

Planned date: Monday

Planned time: 14:00

Planned platform: Discord

Mt. 12

date: 15/03/2021

Time: 14:00

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver

report filed by: Carolina O'Neill Marques

Actions:

- Merging the branches into main
- Went through Sharon's mockups
- Started new sprint with new backlog displayed on trello

For next time:

- Start our tasks

Planned date: Wednesday

Planned time: 14:00

Planned platform: Discord

Mt. 13

date: 17/03/2021

Time: 14:00

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver

report filed by: Carolina O'Neill Marques

Actions:

- Catching up on the progress on our tasks

For next time:

- Continue our tasks

Planned date: Friday

Planned time: 9:15

Planned platform: MS Teams

Mt. 14

date: 19/03/2021

time: 9:15

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques

report filed by: Carolina O'Neill Marques

Actions:

- Discussed where we are on our tasks with Olivia
- Need to do testing
- Created sprint contribution form → to complete

For next time:

- Continue our tasks

Planned date: Monday

Planned time: 14:00

Planned platform: Discord

Mt. 15

date: 22/03/2021

time: 14:00

duration:

attendance:

John Steward
Sam Briggs
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver

report filed by: Carolina O'Neill Marques

Actions:

- Merged certain branches

For next time:

- Finish tasks and start on new ones

Planned date: Wednesday

Planned time: 14:00

Planned platform: Discord

Mt. 16

date: 26/03/2021

time: 9:15

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver

report filed by: Carolina O'Neill Marques

Actions:

- Filled Olivia in our progress
- Merged certain branches

For next time:

- Finish tasks and start on new ones

Planned date: Monday

Planned time: 13:00

Planned platform: Discord

Mt. 17

date: 29/03/2021

time: 13:00

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver
Sam Briggs

report filed by: Carolina O'Neill Marques

Actions:

- Merged certain branches

For next time:

- Continue on tasks

Planned date: Wednesday

Planned time: 12:30

Planned platform: Discord

Mt. 18

date: 07/04/2021

time: 13:00

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Henry Allen
Carolina O'Neill Marques
Sharon Silver
Sam Briggs

report filed by: Carolina O'Neill Marques

Actions:

- Merged certain branches

For next time:

- Continue on tasks

Planned date: Monday

Planned time: 12:30

Planned platform: Discord

Mt. 19

date: 12/04/2021

time: 13:00

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Carolina O'Neill Marques
Sharon Silver
Sam Briggs

Henry was travelling back to uni

report filed by: Carolina O'Neill Marques

Actions:

- Merged certain branches

For next time:

- Continue on tasks

Planned date: Wednesday

Planned time: 13:00

Planned platform: Discord

Mt. 20

date: 14/04/2021

time: 12:30

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Carolina O'Neill Marques
Sharon Silver
Sam Briggs
Henry Allen

report filed by: Carolina O'Neill Marques

Actions:

- Merged certain branches

For next time:

- Continue on tasks

Planned date: Friday

Planned time: 9:15

Planned platform: MS Teams

Mt. 21

date: 16/04/2021

time: 9:15

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Carolina O'Neill Marques
(Sharon Silver)
Sam Briggs
Henry Allen

report filed by: Carolina O'Neill Marques

Actions:

- Filled Olivia in our progress

For next time:

- Continue on tasks

Planned date: Tuesday

Planned time: 13:00

Planned platform: Discord

Mt. 22

date: 20/04/2021

time: 13:00

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Carolina O'Neill Marques
Sharon Silver
Sam Briggs
Henry Allen

report filed by: Carolina O'Neill Marques

Actions:

- Started final sprint: final layouts/design and report

For next time:

- Start on tasks

Planned date: Friday

Planned time: 9:15

Planned platform: MS Teams

Mt. 23

date: 23/04/2021

time: 9:15

duration:

attendance:

John Steward
Tanapat Supatchapichai
Daniel Ledger
Aman Aman
Jayanth Vasanth Kumar
Carolina O'Neill Marques
(Sharon Silver)
(Sam Briggs)
Henry Allen

report filed by: Carolina O'Neill Marques

Actions:

- Filled Olivia in our progress

For next time:

- Finish tasks and report
- Submit coursework

Planned date: Monday

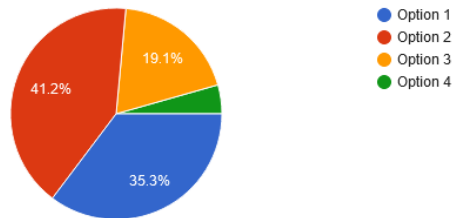
Planned time: 13:00

Planned platform: Discord

Appendix C - Questionnaire

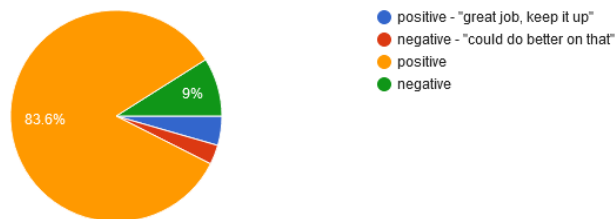
Which graph style do you prefer?

68 responses



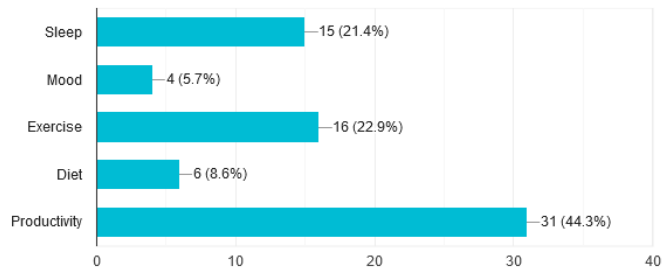
Do you prefer positive or negative reinforcement?

67 responses



What aspect are you currently struggling with the most?

70 responses



Did you know that your diet affects your

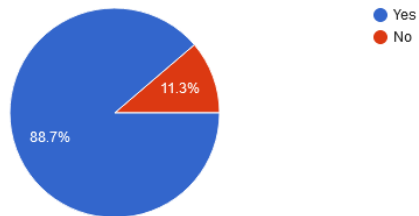


Did you know that exercise affects your



Did you know that the weather affects your mood?

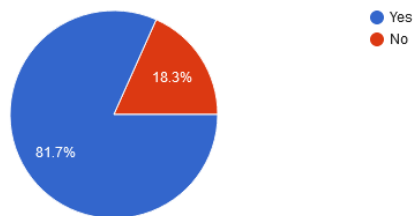
71 responses



Did you know that music affects your productivity?

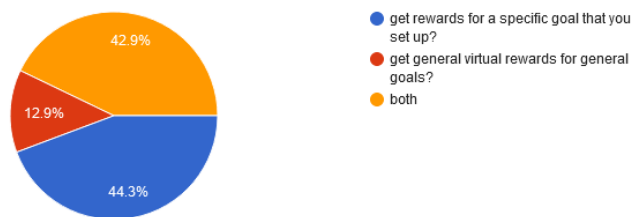


71 responses



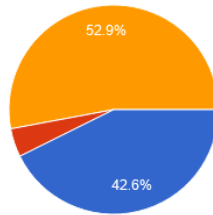
When you're using a Personal Information app, do you prefer to...

70 responses



What motivates you more...

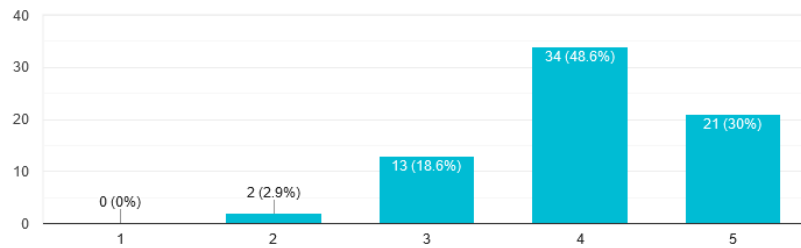
68 responses



- comparing your data to other users with leaderboards and group statistics?
- just focusing on yourself and your personal goals ?
- just focusing on yourself and your goals ?

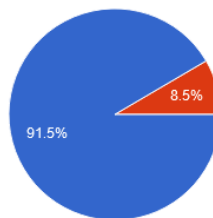
How important is personalisation in an app for you?

70 responses



Would you like to get advice to better your bad habits?

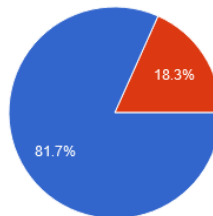
71 responses



- Yes
- No

Would you like to get rewards every time you achieve a goal?

71 responses

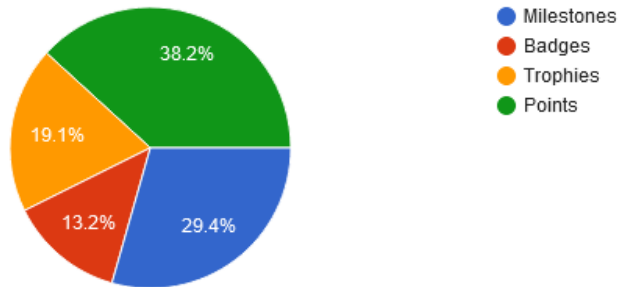


- Yes
- No

If you would, what type of reward you prefer?



68 responses



Appendix D - UI design mockups

Desktop proposed designs

#LOGIN

APP TITLE/FONT LOGO (ONCE DECIDED)

username

password

Login

Don't have an account? [Create one](#)

#ACCOUNT CREATION 1

Current step (1/2)

CREATE AN ACCOUNT

● ●

email address

confirm email

username

password

confirm password

Next

100% opacity for selected field.

~80% opacity for unselected fields.

Erroneous fields

~40% opacity when there are incomplete/erroneous fields up to 100% when all fields are valid.

#ACCOUNT CREATION 2

Current step (2/2)

CREATE AN ACCOUNT

● ●

Link Spotify Account

Finish

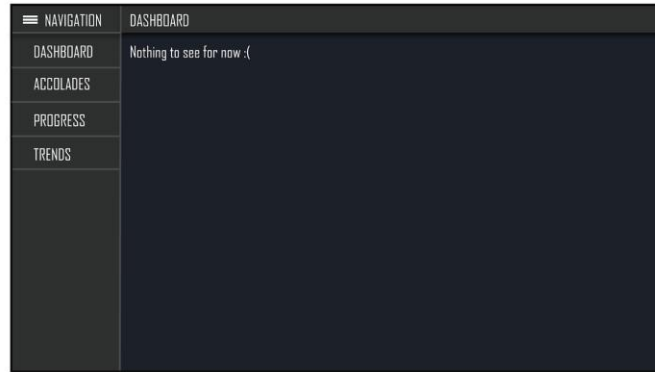
Not sure on the implementation here, assuming going with a pop-out window / browser redirect to link. Upon completion:

✓ Spotify Account Linked

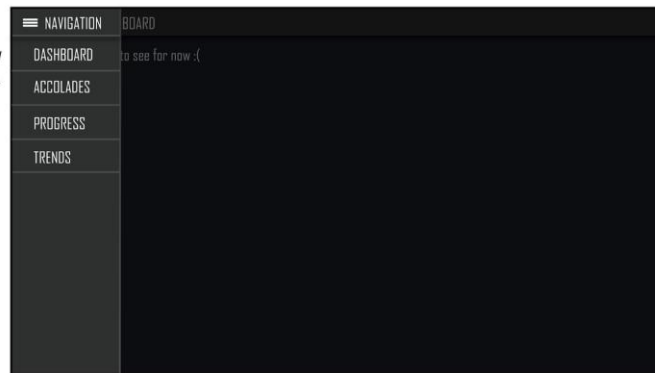
Redirects to #LOGIN

#NAVIGATION

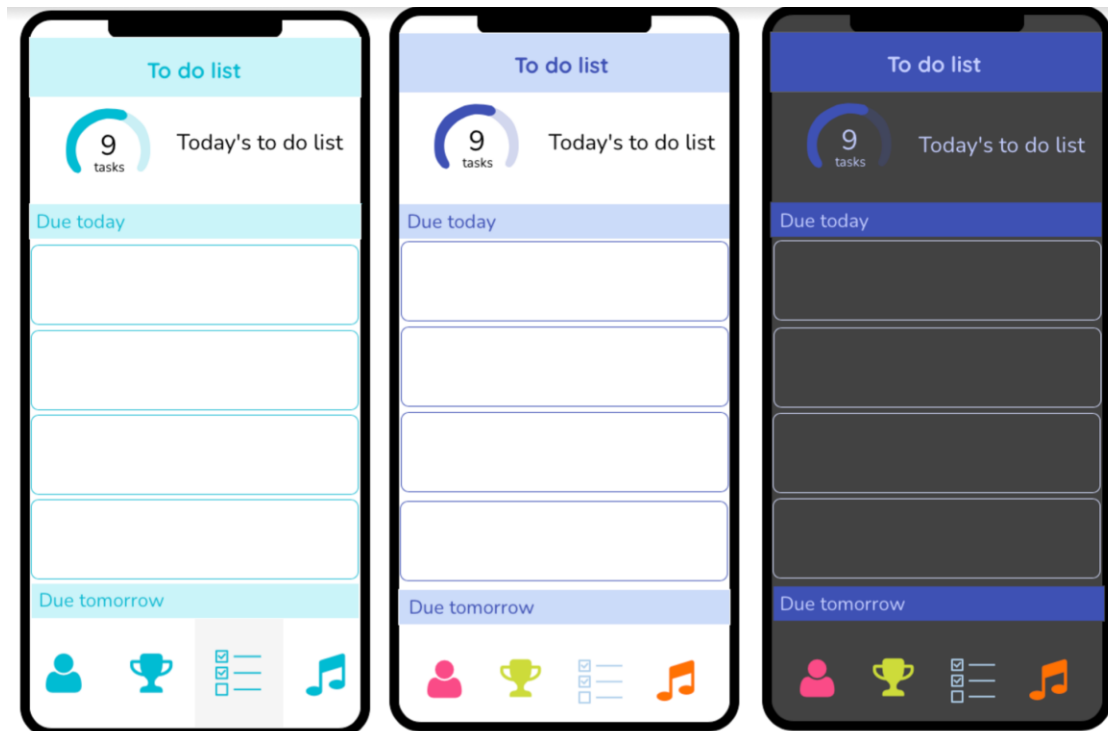
Proposal 1: Sliding navigation bar; the main window gets shifted to the right and its size is reduced.

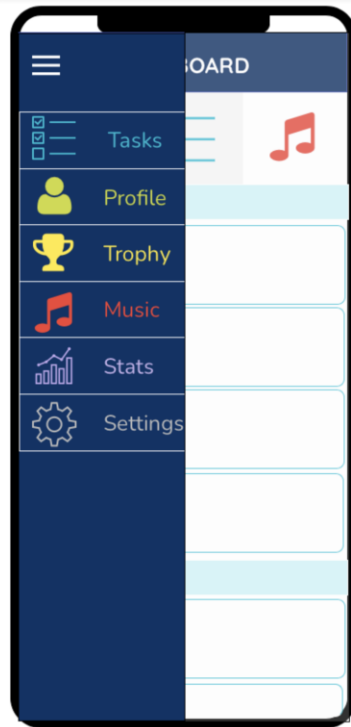
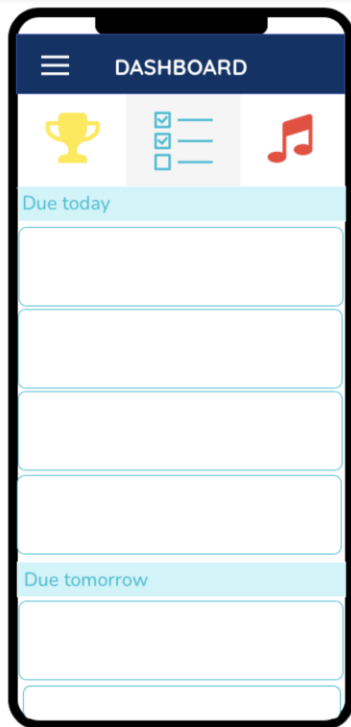


Proposal 2: Pop out navigation bar; the main window is covered with a ~60% opacity overlay. Visual clarity may be worse, however formatting may be easier to deal with. (This may be redundant if formatting for any-size windows is done well)



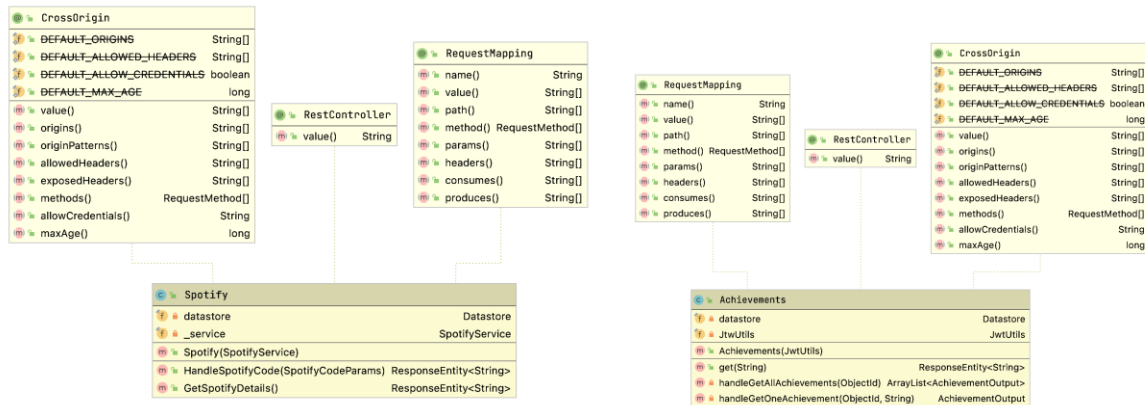
Mobile proposed designs



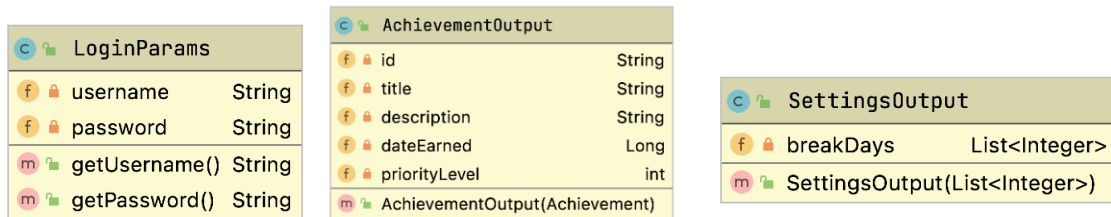


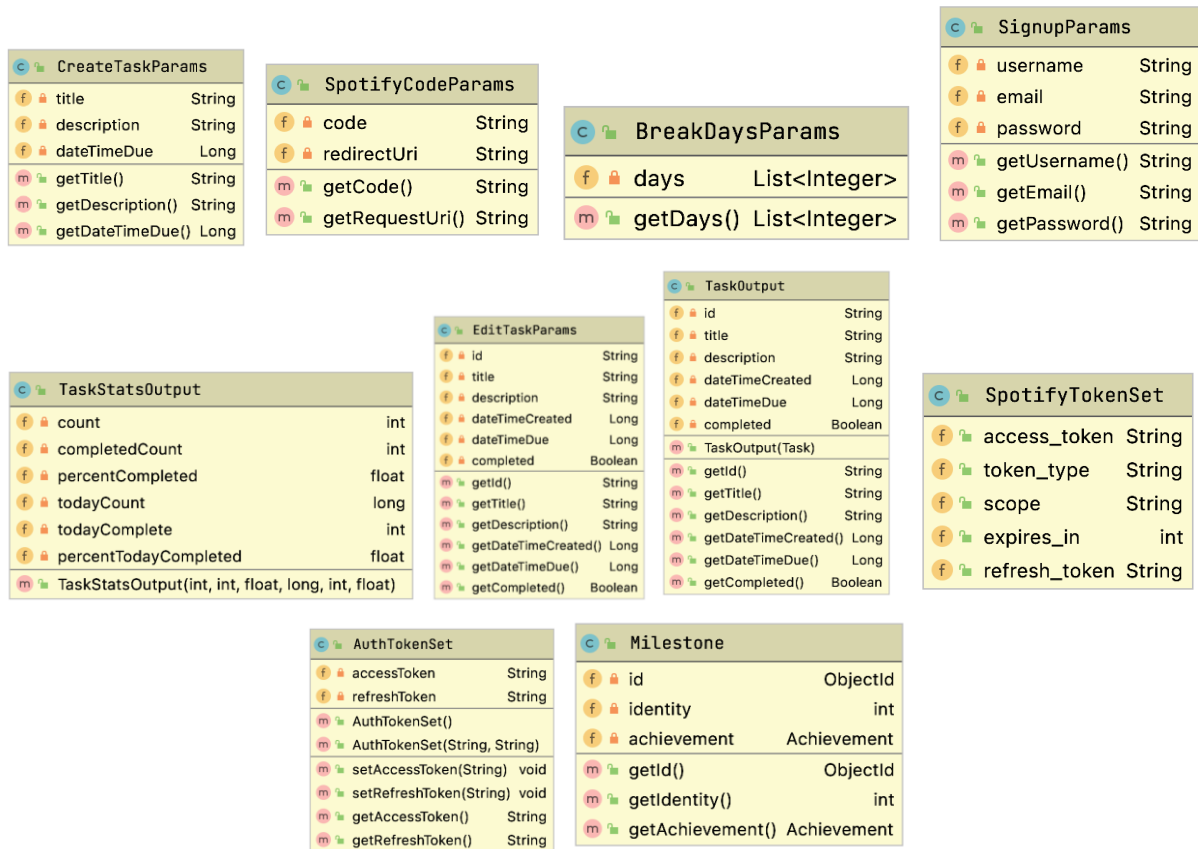
Appendix E - UML Diagrams

Endpoints classes

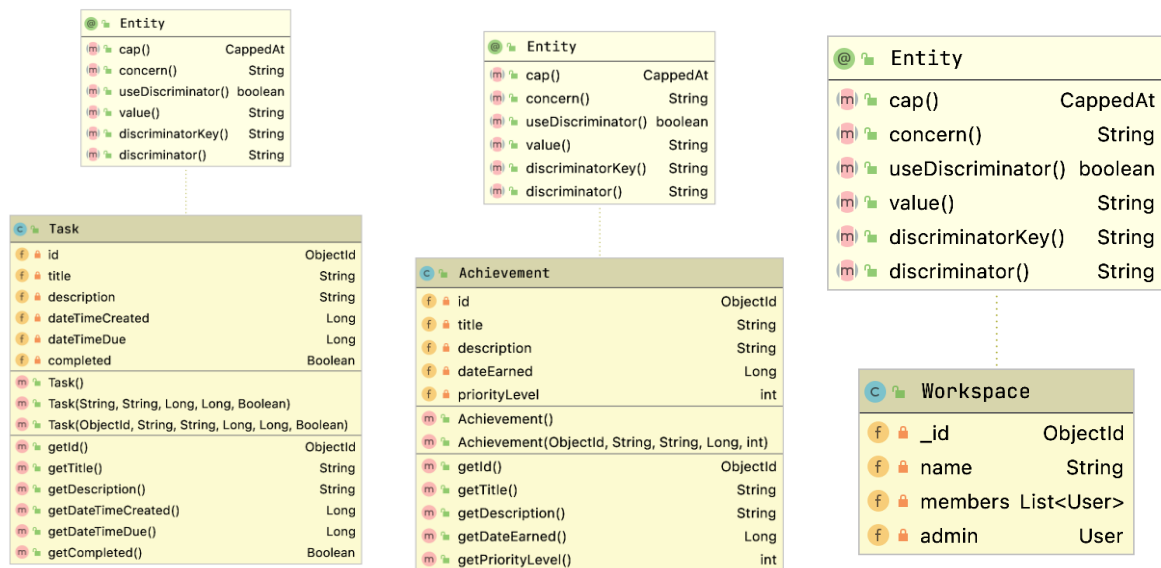


JSON Mapping classes

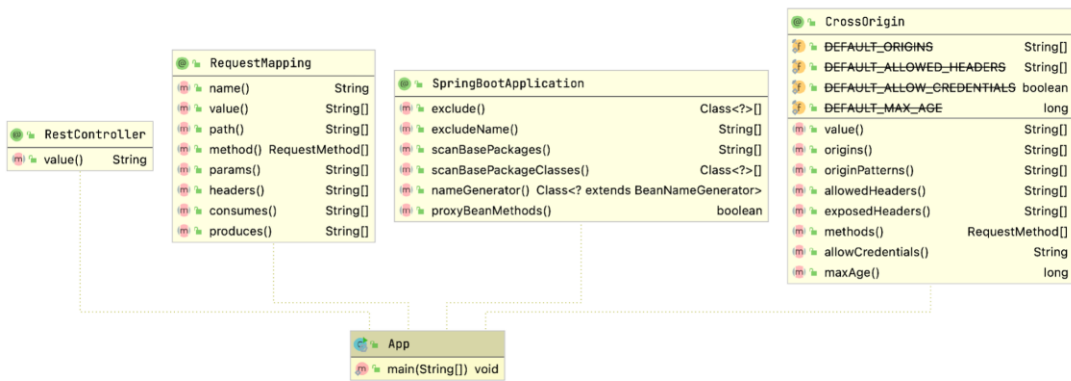




Schema classes



Main functional classes



Utility classes

