Learning Outcome 1 Supporting Document

Stakeholders:

A stakeholder in a system can be described as a person or entity that has a vested interest in the creation, development, and release of the system. Stakeholders play an essential role in the development of a system, as they represent the interests of different groups of people.

Understanding each group's perspectives and concerns, ensures that the system released meets the needs and expectations of all the parties. In the case of HARty, we have identified the following main stakeholders:

- End Users - The end users of HARty are individuals interested in tracking and monitoring their physical health. They have a personal interest in the system, as they can use it to set and achieve their personal fitness goals in a simple and intuitive manner. The end users consist of consumers looking for an effective and convenient tool to help them stay healthy and active.
- Developers - The developer of a system is in charge of building and testing the system. Their purpose is to design and develop the system in accordance to the end users' needs, while ensuring that the quality and performance of the final product meets a high standard. In the case of HARty, this would be me and my team who helped me develop the ML algorithm.
- Google Firebase - Firebase provides the cloud database and authentication services for our system (HARty). The perspective of Firebase is that of a service shareholder, who provide a secure and reliable for the storage and management of user data.
- Organisation manufacturing Respeck - The organisation manufacturing Respeck is a stakeholder as they provide the Inertial Measurement Unit (IMU) sensor used within our system. Their perspective is that of a service shareholder, that is delivering a high-quality and accurate sensor that can be used to track an end user's physical activity and step count.

Therefore, while working on my project HARty I took into consideration the stakeholders mentioned above. However, if HARty was released to the general public (outside the scope of my project), it may attract a wider range of stakeholders who have a vested interest in the system. These stakeholders could include:

- Potential investors or sponsors who are interested in financing and supporting the project.
- Health and wellness organisations that promote physical activity.
- Government organisations if they choose to use the system in their own health and wellness initiatives.
- Research institutions that choose to use the system for academic or scientific research.

These additional stakeholders would go on to play a significant role in the development and release of this system. This means that it is important to understand their needs and expectations while developing and testing the system.

Requirements for Stakeholders:

The stakeholders mentioned previously have specific requirements that must be met when the system (HARty) is implemented. The requirements for each stakeholder are outlined below.

End Users:

- Simple and intuitive UI (User Interface) - The end users need to have a UI that is user friendly and straightforward to use. This will help ensure that the end users have a positive user experience with the system.
- Accurate step and activity tracking - The system should have the ability to accurately track the end user's step count and predict their current activity in real-time. This will allow them see their progress and and help them reach their fitness goal.

- Convenient connection with Respeck sensor - The system should have the ability to seamlessly connect and disconnect with the end user's Respeck sensor. This will help ensure that the end users have a positive user experience.
- Personalise goals - The end user should be able to effortlessly adjust their daily step counter target in the system. In addition to this, they should also be able to turn on and off the notification from the Android system reminding them to stay active. They should also be able to modify the time between each notification. This will allow the end user to stay motivated and help them stay on track to achieving their personal fitness goals.
- Data privacy and security - The system should store a user's personal data in a safe and secure manner. This will help ensure that the end users have a positive user experience with the system, knowing that their data is protected.
- Create and manage a user profile - The system should allow an end user to create and manage an account from within the Android application. The user should be able to view their past activity statistics, make changes to their account, and if needed be able to delete their account and personal data.
- Convenience in switching HAR algorithms - The system should allow an end user to seamlessly switch between the two human activity recognition (HAR) algorithms. This will allow the user to personalise the system, and only receive data that is relevant to them.

Developers:

- Met technical requirements - The system should meet all the technical requirements, decided in the early stages of the implementation.
- Positive user experience - The system should meet the end users' requirements, and ensure that it provides them with a positive and friendly user experience.
- Correct integration within the system - The system should seamlessly integrate with the Respeck sensor, and the Firebase Realtime Database and Authentication services.
- Data privacy and security - The system should ensure that all user data is protected, and the relevant encryption, data privacy and security standards are met.
- Quality assurance - The system should be well tested, and all known errors should have been fixed. In addition to this the system should have a high performance and quality standard.
- Scalable system architecture - The system architecture of HARty should be scalable to support any future upgrades, such as bug fixes or the addition of new features.
- Well documented code - The codebase of the system should be well structured and documented, for effective debugging and troubleshooting during further development or testing. The code should also be in adherence to the best programming practices used in the industry currently.

Google Firebase:

- System reliability - The system should maintain a constant and reliable connection to the Internet while in use. This will ensure seamless data transfer between the Android application and the cloud Firebase services.
- Data privacy and security - The system should ensure that the user data is protected and encrypted before being uploaded and stored in the cloud Firebase Realtime database.
- Secure Firebase usage - The system should ensure that secure and efficient Firebase methods from the Firebase toolkit are used within the HARty codebase. This includes setting up the Firebase database with secure read and write rules, to prevent unauthorised access.

Respeck:

- Reliable data transfer and sensor connection - The system should accept the data transmitted from the Respeck at a rate of 25Hz (50 datapoints every 2 seconds). In addition to this, the system should ensure that it is able to connect, disconnect and reconnect with the Respeck sensor in a simple manner.

- QR code scanning - The end user should be able to use the system to scan the QR code containing the MAC address of the sensor through the Android application.

Functional & Non-Functional Requirements:

Functional Requirements of HARty:-

Functional requirements are specific measurable attributes that describe the requirements a system must contain to meet the requirements and expectations of all stakeholders. These requirements contain information on what features should be implemented, and how the system should behave and operate.

Based on this understanding, I have been able to identify the following functional and qualitative requirements as listed below:

- Simple user interface - A simple and intuitive UI divided between three pages (Dashboard, Profile, and Settings) should be designed. The Android application should make use of consistent UI styles and features throughout the application, to provide a improved user experience.
- User profile management - The Android application should allow a user to create and manage their profile. This also includes the ability to view past activity statistics, reset their password, update their account information, and deleting their account if necessary.
- Step counting - The Android application should allow a user to monitor their step count in real time, and view their daily step count. This requirement also includes the ability to modify a user's daily step target and the creation of the step counter algorithm. The step counter algorithm calculates the user's step count by comparing acceleration data received from the Respeck sensor against a set threshold over a period of 2 seconds.
- Activity tracking - The system should be able to predict an activity being performed by the user in real time. Activities predicted should differ based on the HAR algorithm selected by the user. The system should also include the ability for the user to view their daily activity breakdown, which contains the amount of time they spent doing each activity.
- 2 HAR algorithms - The system should come with 2 different HAR TensorflowLite models. The two different HAR classifiers are the Essential Features classifier and the All Features classifier. The Essential Features CNN-classification model is capable of predicting the following activities (Sitting/Standing, Lying Down, Walking, and Running). Each of these groups includes variations of the activity performed in different ways. The only limitation of the Essential Features classifier is its inability to handle activities that involve stairs or too much general movement. The All Features CNN-classification model is capable of predicting the following activities (Sitting, Sitting bent forward, Sitting bent backward, Desk Work, Standing, Lying down left, Lying down right, Lying down on stomach, Lying down on back, Climbing Stairs, Descending Stairs, Running, Walking, and General Movement). The All Features classifier is able to predict the way each activity was performed in greater detail, making it more capable and powerful than the Essential Features classifier.
- Algorithm selection - The Android application should allow the user to select a HAR algorithm, depending on their needs. This requirement also includes the ability to switch between the 2 HAR algorithms seamlessly.
- Connection with Respeck sensor - The Android application should be able to connect, disconnect, reconnect with the Respeck sensor and receive data from it in real-time at a rate of 25Hz. This requirement also includes the ability for the Android application to maintain a stable connection with the Respeck, while being resource friendly.
- Reminder to Move notifications - The Android application should be able to provide notifications to a user when they have been performing the same activity for a pre-set inactivity time. This requirement also includes the ability for the user to modify the inactivity time, and to turn this feature on and off.
- User data storage and retrieval - The Android application should be able to store and retrieve user data (user's step count and activity statistics for a certain date) to and from the Firebase Realtime database. This requirement also includes the ability to display the data in a simple and easy to read manner.

- Integration with Firebase - The Android application should be connected to the relevant Firebase project account, with the correct rules and permissions.
- View device settings - The Android application should allow the user to view the current network and bluetooth status of the Android device in real time.
- Daily statistics reset - The system should automatically reset and store the user's daily statistics (step count and activity breakdown) to the Firebase Realtime at the start of each day.
- Android compatibility - The Android application should be able to run on any and all devices which run an Android OS of 11 or above.
- Display onboarding sequence - The Android application should display a simple sequence of pages containing information about HARty, whenever prompted by the user.
- Respeck QR code - The Android application should be able to scan the QR code containing the MAC address of the Respeck successfully.
- Gather required permissions - The Android application should be able to gather the required permissions to operate from the user.

Non-Functional Requirements of Harty:-

Non-functional requirements are non-measurable attributes that describe the behaviour, performance, and the operations of the system. These requirements describe how a system should work and behave and therefore they are very subjective and cannot be quantified . This plays a major impact on how the user perceives the final system produced.

Non-functional (NF) requirements can be further broken down into further categories as follows:

- Qualitative requirements - These requirements describe the desired qualities in the system, from the perspective of the end user. This includes examples such as accessibility, and ease of use.
- Safety requirements - These requirements describe how the system should keep itself, users, and user data in a safe manner. This includes examples such as error handling, dealing with system crashes and data protection.
- Correctness requirements - These requirements describe the correct behaviour a system should follow. This includes examples such as ensuring all the functions are executed as required, data validation, and error handling.
- Liveness requirements - These requirements describe the states of the system which allow it to make progress to it's required goal(s). This includes examples such as performance, and responsiveness of the application / system.
- Security requirements - These requirements describe the required security needs that should be in our system. This includes examples such as data encryption, and system backup / recovery.

Based on this understanding, I have been able to identify the following requirements. These requirements are interrelated and are are difficult to categorise separately, this is why they are listed together below:

- Usability - This requirement refers to the ease at which the users can navigate and perform actions within the Android application. This also includes the ability for users to customise and personalise the Android application according to their preferences.
- Reliability - This requirement refers to the dependability of the system as a whole. This includes the ability for HARty to maintain stability, and the consistency of activities predicted and steps counted. In addition to this, this requirement also includes the ability for HARty to maintain a stable connection with the Respeck sensor and the relevant Firebase services.
- Compatibility - This requirement refers to the compatibility of the Android application with different versions of the Android OS, and the ability for the application to run on a wide range of devices. Since we have decided the minimum Android OS version to be 11, this requirement would involve testing with Android OS versions 12 and 13.
- Maintainability - This requirement refers to the ability of the system to deal with the release of updates such as bug fixes and the addition of new features. In the case of HARty, this

requirement is not a major priority as the project is not public and it is not actively being worked on.

- Performance - This requirement refers to the speed, responsiveness, and efficiency of the Android application. The following factors such as the response times of the app, processing time for the machine learning algorithms, and the amount of CPU, RAM, storage and battery life used, are taken into consideration.
- Security - This requirement refers to the security of the system as a whole. The main focusses are the security of the user data and the authentication system used in the Android application. This requirement also involves ensuring that the system is protected against unauthorised access and data breaches, by implementing secure login methods and data storage practices.
- Scalability - This requirement refers to the ability of a system to handle an increase in load and demand. In the case of HARty, the requirement of scalability is important to ensure that the system is able to accommodate an increasing number of users and data, and computational power.
- Availability - This requirement refers to the ability of the system to function and provide it's services to users at all times. In the case of HARty, the requirement of availability is important to ensure hat the Android application is up and running, and all components of the system are functioning properly.
- Accuracy - This requirement refers to the accuracy of the machine learning algorithms in classifying a user's activities and the accuracy of the step counting algorithm in detecting and counting the steps taken by the user.
- Customisability - This requirement refers to the ability of users to personalise the Android application according to their preferences. This includes customising the daily step counter target and the HAR algorithm used to predict the current activity.
- Data Storage and Management - This requirement refers to the ability of the Android application to store and retrieve user data efficiently and reliably. This includes using efficient and reliable data storage and retrieval techniques from he Firebase toolkit, and ensuring that the user data is persisted even if the Android application crashes.
- User Experience - This requirement refers to overall experience of the user while they are using the system. This requirement includes the design and appearance of the Android application, and sending simple, non-intrusive and non-distracting Android notifications.

Requirements based on level:

The requirements for HARty can be broken down further and tested at various levels, as shown below. It is important to note that the following list of requirements is not exhaustive and only represents a limited number of them.

System Level Requirements:

- User Interface - This requirement ensures that the UI elements meet the required design specifications, and are working as expected.
- Performance - This requirement ensures that the system's overall performance, speed and stability is high under different scenarios and loads.
- Compatibility - This requirement ensures that the HARty Android application is compatible with devices running Android OS 11 or above.
- Functionality - This requirement ensures that the main functions of the system work correctly together, when the whole system is under constant use.

Unit Level Requirements:

- Activity Classification - This requirement ensures a high performance and accuracy of the Essential Features and All Features HAR algorithms when classifying everyday activities.
- Step Counter - This requirement ensures that a high accuracy of the step counting algorithm in detecting steps and updating the step count. This includes dealing with edge cases of low acceleration values or sudden changes in acceleration data.

- User Authentication - This requirement ensures that the main functionality of the Firebase Authentication System is working correctly in managing the user authentication into the HARty Android application. This includes email validation, account creation and password encryption.
- Data Storage and Retrieval - This requirement ensures that the main functionality of the Firebase Realtime Database system is working correctly in storing and retrieving user data.

Integration Level Requirements:

- Sensor Integration - This requirement ensures that the integration of the Respeck sensor with the HARty Android application is working correctly, and the data is being received and processed in an accurate manner.
- HAR Algorithm Integration - This requirement ensures that the integration of the Essential Features and All Features HAR algorithms with the HARty Android application is working correctly, and ensures that the predictions are being processed and displayed accurately in the UI.
- Firebase Integration - This requirement ensures that the integration of the cloud Firebase service with the HARty Android application is working correctly, and ensures all the services are working accurately and in real-time.
- UI and Functionality Integration - This requirement ensures that the integration of the UI (Android application) and the key functionalities of the systems are working together correctly. This could include ensuring key functionalities such as activity classification, step counting, profile management, and settings management with the system as a whole.
- Notification System Integration - This requirement ensures that the reminder to move notification system and the Android Notification Service work together correctly, and that the reminder notifications are triggered at the required times.

As stated previously, the requirements listed above are not exhaustive and are only a limited number of examples. To effectively test these System, Unit and Integration Level requirements, a combination of the testing strategies described below can be employed. The System, Unit and Integration testing strategies are particularly suited for this purpose, as they are specifically designed to test and validate these requirements.

Testing Strategies:

During the Software Testing course, we learned about various approaches to software testing that guarantee the proper functioning of the system and ensure that the user experience is up to standard. Based on my understanding, I have selected the following testing approaches as relevant strategies for testing the HARty system:

- Unit Testing - This testing strategy focuses on testing the individual components of the codebase, ensuring that they function as expected independent from the rest of the system. In the context of HARty, unit tests can be used to verify that all the individual classes and functions are working correctly.
- Integration Testing - This testing strategy focuses on testing if the different components of the system interact and work together as expected. This can help us detect issues that are usually not known when testing components individually. In the case of HARty, integration testing can be used to verify that different components of HARty (such as the Respeck sensor and HAR algorithms) are able to work together to achieve a goal.
- System Testing - This testing strategy focuses on testing the system as a whole, and ensuring that everything works as expected. In the context of HARty, we can use system testing to ensure that all the features of HARty work together in real-time to produce the correct output and states.
- Manual Testing - This testing strategy focuses on manually testing different components of the system, without the use of test scripts. In the context of HARty, there are various components within the system (such as the step counting algorithm), which can only be correctly tested by a real human. A test script or other software will not be able to replicate a human taking steps, and ensuring the correct step count is displayed.

- Functional Testing - This testing strategy focuses on testing if the system is able to meet all the requirements decided upon prior to the development of the system, and if all the functionalities work as expected. In the case of HARty, functional testing can be used to verify that we built our system as expected, and that functions such as activity prediction predict a activity with an accuracy rate of 96-100%.
- User Acceptance Testing (UAT) - Also known as *Usability Testing* or *Acceptance Testing,* this testing strategy focuses on involving users to test the HARty system to determine whether the system produced meets their requirements and expectations. This testing strategy is used to verify the user experience, ease of use, and the overall user experience. This form of testing is known as Black Box Testing (BBT). BBT is a form of testing which is performed without any knowledge of the internal workings of the system, while keeping a key focus on the input and output of the system. In the case of HARty, we can use UAT to verify if the system meets an end user's particular requirements.
- Performance Testing - This testing strategy focuses on evaluating the performance of the system under different use cases and loads. This helps us identify any performance bottlenecks in the system, and also helps us understand the scalability and reliability of the system. In the context of HARty, we can use performance testing to determine and enhance the CPU, RAM and battery usage.
- Security Testing - This testing strategy focuses on testing the security of the system as a whole and any user data stored within it. This testing strategy can help us identify any vulnerable points within the system, and ensures that the system and user data maintain the concept of confidentiality, availability and integrity. In the case of HARty, we can use security testing to detect and prevent unauthorised access of the system and end user data.
- Regression Testing - This testing strategy focuses on testing the system after every change to the codebase. This is done to ensure that all of the changes made, have not effected the functionality of previously working code. This testing strategy helps ensure is system is stable and reliable over time. In the case of HARty, this can help the developers ensure that their new changes are not breaking anything else in the codebase.

Limitations and Evaluation of Test Strategy:

I believe that testing HARty effectively requires taking a blended approach to the testing strategies mentioned above using the Test Driven Design methodology. The most effective approach would be to create and utilise test scripts to automate all the tests that can be automated, while the remainder of the tests are conducted manually by a human. I think this because there are certain components of HARty which require human intervention and verification during testing and validating, which cannot be accomplished through an automated test script. This would allow us to comprehensively test HARty, and ensure that all of it's features are working correctly, thereby reducing the likelihood of errors and bugs in HARty.

Using the automated testing strategies discussed above, we are able to test the HARty system thoroughly. However, as previously mentioned, there are certain features and functionality of HARty that cannot be tested through an automated test script. For these areas, a manual testing strategy is required. Therefore, the most effective approach would to integrate automated and manual testing.

Given that a substantial portion of the application relies on data obtained from the Respeck sensor (Bluetooth IMU), the developers have to concentrate on manually testing various aspects of our application. These include the step counting algorithm, the HAR algorithms, and the daily breakdown algorithm. One of the main reasons for this is that the emulator within Android Studio does not have the capability to support Bluetooth, and therefore Bluetooth functionality can only be tested on actual devices. Additionally, since these parts of the application are dependent on information gathered using a body sensor as well as human input to determine the pass/fail status of a test, it is not possible to automate the process.

In addition to this, the development of machine learning algorithms also requires testing to determine the accuracy of the ML models. This testing cannot do done inside the test suite for the Android application, but instead requires the use of cross verification techniques (such as LOOCV) and testing the ML models against previously unseen Respeck data to confirm it's accuracy and range.

Therefore, in this section we have seen how to apply various testing strategies in the context of HARty. We have discussed the strategies available and the limitations we face with each. The subsequent LO documents explore these concepts in more detail and provide evidence of using the testing strategies mentioned above in HARty.