## 3. User Stories

# 3.1 - Sprint 2 Goals

For sprint 2, our goal is to have a skeleton of the final application up and running, with all the essential features working. Allowing us to present a working application to the client gives us time to incorporate their feedback before commencing sprint 3 and finalising the application. As a team, we have identified three overarching goals which will be complete during sprint 2, being:

1. Collect and store sample sound level recordings
2. Build out the back end of the application
3. Develop a dashboard that displays the sound levels in a meaningful way

# 3.2 - User Stories

# To help understand what needs to be completed for each of the goals identified for sprint 1, we have developed a number of user stories.

# The navy needs to store sound recordings on board the ships to help monitor the soldier's wellbeing

# The health officer needs to have access to the sound levels soldiers are exposed to onboard navy ships, helping them make better decisions to manage fatigue levels

# For the health officers to interpret the sounds readings, we need to display the findings in a dashboard

# Having sound data readily available on different locations onboard the ship will help schedule the soldiers to maximise their productivity levels

# The health officer will require the sound levels to be analysed on a person-person basis, a group basis, and the entire crew in order to understand how the level of noise influences fatigue

# Thales will need to integrate the sound monitoring software into their suite of offers before deploying it on the navy ships

# The majority of the user stories will be identified for sprint 3 as we flesh out functionality. We are focused on developing a frame in which we can add user features further down the track.

# 3.3 - Sprint 2 Tasks

We can further break down the three goals for sprint 2 into smaller actionable sub-tasks.

1. Collect and store sample sound level recordings
   1. Use a phone to collecting sample sound readings
   2. Collect samples in a variety of different environments (quiet, normal, loud, very loud)
   3. Collect samples from an environment where the noise levels spike. Helpful when testing whether we can pick up abnormal noise increases.
   4. Convert sound recordings into a format we can store and conduct our analysis from
2. Build out the back end of the application
   1. Carefully plan out the schema for the database
   2. We will be using flask and SQL lite for the backend
   3. Build the database with sample data collected from the above point
   4. Write functions that will be required to access and store data in the database
   5. The database will be running on a local server for sprint 2
3. Develop a dashboard that displays the sound levels in a meaningful way
   1. The dashboard will be built using Typescript and React
   2. Come up with a variety of visualisation in PowerPoint to give the client an indication of what they will be looking at.
   3. Check to ensure the visualisations satisfy the client's requirements
   4. Transform the visualisations into a dashboard
   5. Visualise the sample sound recordings in the dashboard

We have chosen to work on all major components of the application simultaneously, so as we develop the application and iteratively tweak the functionality and performance, all parts will still work together. For example, suppose the sound collection methodology must change for an unforeseen reason. In that case, we can quickly adapt the backend to handle the data, and the dashboard can be adjusted to use the data.