The original brief for my project was self-defined. I proposed using deep learning for estimating the story point value. I decided that Training and evaluation of the model will be the main focus of the project. A secondary goal will be to host the ML model in a service that triggers on ticket creation, adding the estimate to the ticket.

My original brief did not talk about which model architecture should be used and said that instead this should be decided while working on the project. I decided quite early on to use BERT, this is because I saw some other usage of BERT for text classification, and expected to find an example of ticket estimation with it. I was unable to find any which is one of the reasons that I decided to use BERT for my project. I also considered using GPT or Llama for the base of my model but I found them too resource intensive to realistically be able to do a full investigation, particularly for the grid search.

During the project spent a lot of time writing the code and training my model, which meant I did not have time to implement a hosting solution for the model. Therefore, I fulfilled the main goal but did not fulfil the second goal. However I did discuss the second goal through the “Practical Application” section of the project. This enabled me to explore how a model could be hosted as a service in AWS or a similar service. Unfortunately, I was unable to investigate the exact implementation of this. Implementing this fully would have also allowed me to confirm the 3rd objective of the project as explained in the introduction, which is to be able to classify tickets in less than 10 seconds. One of the things which could have been better is that I should have moved on from getting a good accuracy sooner and started looking at implementing a hosting solution.

Another thing that could have gone better is that I should have been more methodical with my initial investigation. When trying different network sizes and depths, I wasted time redoing experiments because I wasn’t documenting them in a methodical and sensible way. Instead, I was committing my code at specific points with the results in the commit message. However, this did not work very well, as I often forgot to save the model sometimes and due to the file format (Jupyter notebook) the code differential was not very clear. Once I realised that this was a problem, I endeavoured to find a better solution. This came in two forms. I started saving the entire output of every run, with information about the architecture and hyper parameters in the file name. I also made a list of everything I needed to try and documented the best accuracy and F1 score in a text file. Once I made these changes to the way that I was investigating, I made considerable progress. This taught me the importance of planning and documentation.

In terms of the accuracy of my project, although the accuracy was 70% which is quite high it only classifies into 3 classes. My original brief was to classify the tickets into particular story points, which could either have been to classify into 10 catagories or could be treated as a regression problem and output a single value for story points or a single value for a time estimatiojn. Unfortunately I decided that there was not enough data to be able to train a model to do this. This is because I would only