During our sprint planning we took the features and broke them down into tasks.

We then created some gherkin specs for each task as a collaborative exercise.

One person in the team tried to ensure that we stuck to a consistent language and vernacular while describing the tasks as gherkin specs.

This exercise took around 2-3 hours in total to complete the specs for every feature for the prototype.

The process was not too painful because the features were already well defined with the PO having a clear vision of how the prototype would work.

## Implementation

We took a TDD approach.

Having the gherkins made this really easy as the tests were already defined with:

The **context** of the test

The **action** that would be carried out

The **result** of that action

We took a feature and started by copying the gherkin specs from the first task into specflow.

We then wrote the code to implement the test logic, while stubbing out the implementation code.

We had some difficulty deciding on the level we should be aiming our tests at (unit, integration, front-end, etc) we decided to make the decision on an individual gherkin spec bases and tagged the tests appropriately so that they could be identified.

## Outcomes

During development we found that we hardly had to seek further clarification for the work due to the unambiguous way that it had already been defined.

QA were also able to test the functionality without having to seek clarity from dev or the PO on exactly how something should work.

QA also had a better understanding of what was already being covered by unit test as each test was built directly from an implementation of the gherkin spec.

The main feedback from QA was around usability and design with no issues reported around the feature logic.

On completing the work we found that we had successfully implemented the vision of the PO without any need to make changes due to misunderstanding of feature requirements. We have yet to find any significant bugs within the prototype that we have built.

## Summary

Defining the gherkin specs upfront that were agreed by the PO, devs, and QA allowed us to take a TDD approach without the added burden on the individual developer to design the test scenarios themselves. This ensured a high level of code coverage.

Even without the TDD benefit of the gherkin specs, the clarity of the task alone was worth the upfront effort due to the greatly reduced need to seek clarity during the development process.