# Reading guide

For this project I decided to make a website where users can buy and sell products(Games).

The documentation I decided to make are:

* Project plan
  + Contains the functions and user stories for this project
* Design document
  + Contains important information about the design and functioning of the rest API
* Research document
  + Contains research questions and answers that are relevant for this project
* Security report
  + A report on how this application protects or don’t protect against the OWASP top 10 security risks
* Test plan
  + Contains the planning for the testing that needs to be done for this project
* Test cases
  + A list of manual tests that need to be done
* Test report
  + A report on what test cases passed and failed
* UX feedback report
  + A list of feedback gotten for potential users

This application consists of a rest API and a react front end.

For the API, to ensure code quality I made 121 unit tests covering 90.5% of the code and a CI/CD pipeline using SonarQube and docker.

It uses http and websockets to communicate with the front end, the authorization and authentication are made using JWT(json web tokens).

There are also custom exceptions to better display what went wrong. The DTO(data transfer object) pattern is applied and implemented using for example DTO Convertors(classes that convert DTOs to entities and vice versa), all the sold principles are also used.

The database used is MySQL to connect to it ORM(object related mapping) is used, the object related mapping also implements polymorphism, one to one, one to many and many to many joins. The database itself has indexes for the fields that are most often used and searched by.

The react front end is a lot simpler then the API it has nothing fancy, just simple JavaScript with some react elements like reactDOM, useState, UseEffect, useParams, useNaigate, Route, Link… it has some regex for validating input, axios is used to connect to the API.

No Bootstrap is used, all the html and css is written manually which is one of the reasons for the simplistic look of the application. I didn’t put much work into the css. I would say it’s the most boring thing I had to do for this project so I did the least possible.

For the testing there are unit, integration and (maybe) end to end tests.