Ensuring my application works by testing it.

**Test plan**

Sam Philipsen | 13-1-2022

# Versioning table

|  |  |  |
| --- | --- | --- |
| Date | Version | Changes |
| 05-11-2021 | V0.1 | Created document. Added UAT’s and test strategy. |
| 13-1-2022 | V0.2 | Merged UAT’s and test strategies from design document and planning document into this document. |

# Test strategy

What tests will be performed:

The backend will have unit tests testing the important classes in the backend, like the controller classes and the manager classes. The game classes will also have unit tests for the individual methods; however they will also have tests testing the entire class by going through the routine of playing the game the class presents.

The aim is to have a >80% unit test coverage.

For the frontend the natural flow that an average user should go through will be tested. For example the logging in process, or playing the cointoss game. It is difficult to test the games because the result is random.

Why?:

Unit tests are a good way to test every optimal outcome of a method, and eventually have tested the entire class. By unit testing every method that is unique you are covering most of the scenarios you expect a user to go through. This ensures your classes work like you want them to and catches any errors you do not want. The reason the unit test coverage is not 100% is because testing literally every method is over doing it. This means you are testing methods that should have no chance of failure, like a simple getter or setter. Testing those would be unnecessary and a waste of time.

Integration testing is not necessary because the unit tests already cover enough of the method’s classes. For the game classes however, it’s important to know if the class flow follows what the design intended. Fully testing the flow of a game class means it works in the order that it is intended to work.

How?:

For the backend, the JUnit version 4.13.1 tools are used to perform the unit tests.

For the frontend, Cypress is used.

## User acceptance tests (UAT’s)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID | Name | Pre-condition | Test data | Expected result |
| UAT-1 | Playing a game of coin toss | User is in the game and has decided to play | User’s name: Peter  Password: 123  ID: 0  Points: 100 | User choses a side of the coin and waits until the game starts. |
| UAT-2 | Betting on myself to win a game | User has bet 50 points on himself to win. | User’s name: Peter  Password: 123  ID: 0  Points: 100 | User either receives double his bet back, or loses his entire bet based on if he won or lost. |
| UAT-3 | Winning a game of coin toss | User has won the game. | User’s name: Peter  Password: 123  ID: 0  Points: 100 | User receives double their bet back, if they put a bet in, and is then thrown out of the game. |
| UAT-4 | Losing a game | User has lost the game | User’s name: Peter  Password: 123  ID: 0  Points: 100 | User loses their bet and gets thrown out of the game. |
| UAT-5 | Joining a game | User is logged in and on the game-list screen | User’s name: Peter  Password: 123  ID: 0  Points: 100 | User has clicked on a game that they want to play, and the website is now changed for them to be in the game. |
| UAT-6 | Looking for an available game to play | User is on the game-list screen |  | User or guest can look through the list and see what kind of games are offered. |
| UAT-7 | Hosting a game for people to join | User is in the host settings menu screen | User’s name: Peter  Password: 123  ID: 0  Points: 100 | The user is presented with options for the session they want to start. These options are: type of game, maximum number of players and the minimum bet required. After choosing the preferred settings, the user can start the game. |
| UAT-8 | Registering an account | Guest is on the register page |  | The guest has to fill in the options of e-mail, password and name. The system will check if the user does not exist already. If not, an account is created. |
| UAT-9 | Editing account information | User is on the account information page | User’s name: Peter  Password: 123  ID: 0  Points: 100 | The user sees the editable fields (name, password and email) and can change them. Upon clicking ‘save’ these changes are saved. |