Testing Policy

Golf Course Mapper

Team Recursive Recursion

September 21, 2018

Contents

1 Introduction

1.1 Purpose

This *Testing Policy Document* is intended to be a guide on the testing procedure, tools and practices that are followed when working on the *Golf Course Mapper* project. The remainder of this section describes the different repositories used within the project and the remainder of this document then continues to describe the testing policies applied to each individual repository, excluding the documentation repository.

1.2 Repositories Structure

There are four (4) repositories that are used for the project: `web-app`, `mobile-app`, `mapper-api` and `documentation`. A short description of the contents of each repository is listed below.

- `web-app`

  Contains web application subsystem used for mapping and managing golf courses. The web application is built using *Angular 6*.

- `mobile-app`

  Contains the *Android* app used for viewing golf courses. The mobile app is built using native Java code.

- `mapper-api`

  Contains the *Entity Framework Core* API used to manage and provide access to the database. The database used is *PostGIS*, an extention of *PostgreSQL*.

- `documentation`

  Contains all documentation relating to the project, including this file. Documents are written in LaTeX. The repository also contains all additional images and published PDF versions of the documents.

2 Testing Process

The following describes the process that is followed when testing code. The process described is the same for all repositories.

2.1 Targets

The following tests are performed to ensure that the system is operating correctly and satisfies the functional requirements, as outlined in the *Software Requirements Specification* (SRS).

- *Unit Tests* are performed to ensure that individual functions and components of each system component performs correctly.
- *Integration Tests* are performed to ensure that the different subsystems operate correctly. This is further described in Section 5.
- *Acceptance Tests* are performed along with the client to ensure that the system meets the client's requirements and expectations.

The following tests are performed to ensure that the system complies with the non-functional system requirements, as outlined in the SRS.

- *Stress Testing* is performed on the API to ensure that it meets the performance requirements.
- *Usability Testing* is performed to receive feedback on the UI design of the web and mobile applications. This also ensures that the UI is as user-friendly as possible.

2.2 Procedure

Unit tests must be written and passed before code can be merged to the `master-dev` branch. This ensures that code on the dev branch is at least functionally correct. For details on how unit tests are performed for the mobile app and Mapper API, see the sections 4 and 5, respectively.

Integration tests must be passed before code can be merged to the `master` branch. This ensures that code on the master branch is always in a usable state. For details on how integration tests are performed, see Section 5.

Acceptance tests are performed monthly with the clients to verify that the system complies with their requirements.

3 Web App Testing

This section describes the testing policies adopted in the `web-app` repository. Being a subsystem that mainly acts as a frontend, only usability tests are performed. The history of the tests are shown in Section 3.2.

3.1 Usability Testing

In order to satisfy the usability requirement of the system, it is necessary to perform a usability test to ensure that the system is user-friendly enough to non-technical users. A formal usability test is performed, asking testers to perform specific tasks on the system.

Testers are then asked to provide critical feedback on the system, which is then analyzed. The results are then used to identify key issues in the design of the user interface.

3.2 History

Figure 1 illustrates the usability test performed on 20 September, 2018. Six testers from various backgrounds were asked to perform a serious of tasks on the web application, without any prior introduction to the user interface. The testers were then given a form to fill out, asking them to provide any feedback.

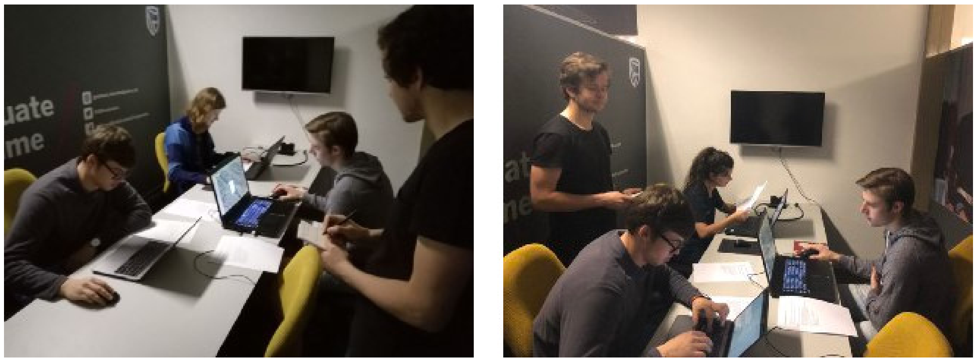

Figure 1: Usability test in progress

4 Mobile App Testing

This section describes the testing policies adopted in the `web-app` repository. The tools used for testing are described in Section 4.1. The layout of the test cases and history of the tests are shown respectively in Sections 4.2 and 4.3.

4.1 Tools

What tool, justify choice and give configuration details (how to use)...

4.2 Test Cases

Where (folder) are the tests located...

4.3 History

Screenshots...

5 Mapper API Testing

This section describes the testing policies adopted in the `web-app` repository. The tools used for testing are described in Section 5.1. The layout of the test cases and history of the tests are shown respectively in Sections 5.2 and 5.3.

5.1 Tools

What tool, justify choice and give configuration details (how to use)...

5.2 Test Cases

Where (folder) are the tests located...

5.3 History

Screenshots...