THE UNIVERSITY OF WESTERN AUSTRALIA

CITS3200 Project Team 18

UWA Academic Skills Drop-in - Client queue flow and usage reporting software

Alex Mai, Jordan Hartley, Frinze Lapuz, Jake Yendell, Alex Hoffman, Liam Hovell

Table of contents

1. What is this project?

1.1 Team

2. User

2.1 User Documentation

3. Administrator

3.1 Administrator Documentation

4. Developer

4.1 Technical Documentation for Developer

4.1.1 Client-Side Application

4.1.2 Server-Side Application

4.2 Requirements

4.2.1 Acronyms, Abbreviations and Definitions

- UWA: The University of Western Australia

4.2.2 Aim and Scope

4.2.3 Requirements

Stage 1 Functional Requirements

The following are the core functional requirements for the first stage application

USERS

Stage 2 Functional Requirements

"Nice to have"

Some of the "nice to haves" of this project will be covered in this requirements documents. Howevever, "nice to haves" usually will come along as the users of the system see fit. This will be documented in the Issue Tracking Management sytem of the code repository.

Non-functional Requirements

Identifier	Name	Description
NFR1	Security	Only authenticated and authorised users should be able to perform actions such as adding equipment, updating equipment location and information, or searching for specific equipment.
NFR2	Performance	The loading time should not hinder the user experience and productivity of the user in the website. The **page/actions** should have a loading time < 5 seconds on most computing environments on standard internet connections**
NFR3	Maintainable and extensible	The website should be relatively easy to update and extended to accomodate for new contexts.
NFR4	Recoverable	In the event of the web server or database server crashing, all stored data should be fully recoverable.
NFR5	Intuitive user interface	The website should have an intuitive / easy-to-use user interface, so that users will be able to easily use the website and update the equipment database
NFR6	Compatibility	The application should be compatible with recent versions of the major browsers (Safari, Chrome, Firefox and Edge) on laptop and desktop computers
NFR7	Deployability	The application should be compatible with deployment in the SHL VPS

4.2.4 Proposed Solution

The proposed solution is to build a custom web application that will encompass and satisfy the requirements (by completing the suggested "ideal solution" as per the Aim and Scope).

Some research for existing design solution has been done for this project see Appendix: Existing Design Solution. The beauty of custom web application for the team is that it upskills the current software engineers as aligned in the purpose of this unit, and the high possibility of extending application depending on the requirements without being constrained with the bulk of codebase

of other unmaintained opensource projects. Comparatively to enterprise systems, most enterprise systems will charge per users that use the system, this easily becomes expensive because the amount of users that will use the system should be able to accomodate the number of users that are interested in looking for the assets.

Core Technologies

The custom web application will aim to satisfy all the requirements in here along with the "Nice to haves" as they come up. The application will be built using the ...

The authentication system will be outsourced to the UWA Pheme Authentication API to allow any users in with a UWA Pheme account to login.

DOCKER

Docker is a deployment technology that allows virtualization in a server to allow the packaging of software into containers for deployment. To satisfy NFR7 - Deployability, the web platform will use docker to allow the deployment through the SHL VPS Server.

Furthermore, Docker will be used for orchestration of different services in development to increase speed of development, and reduce inconsistency between developers devices (NFR3 - Maintainable and extensible).

CODE QUALITY

The code quality will be ensured by peer reviews between the developers in the team.

CODE STORAGE AND DEVELOPMENT CONTROL

Git source control will be used, using the remote UWA System Health Lab organisational GitHub (NFR3 - Maintenace and Extensibility).

Prototype

See the Prototype mentioned in Figma Interface Prototype

Execution Team

The development of the web platform will be performed by

4.2.5 Development and Methodology

As per the staged requirement, majority of the development will take place on the stage 1 whereas stage 2 are feature-based requirements for the syste,.

4.2.6 Appendix

Existing Design Solutions

Some of the design solutions that have been considered with great detail and justification are here.

4.3 Coding Patterns

4.3.1 Casing

This codebase will be using camel casing.

4.3.2 Linters / Formatters

This will automatically format your code if you install ESLint in VS Code or type `yarn lint` in the specific folders.

Make sure you have installed the devDependencies so additional linters can be used.

4.3.3 Github Issues and Pull Requests

Most changes in the codebase can be matched to a github issue that contains description of the work that needs to be done. Each of the pull request are matched to this github issue with the branch name that has a standard `c{Issue Number}-{branch name}`. The issue number allows referencing especially when resolving reason for change.

4.3.4 Development with Docker

The development is done with Docker to orchestrate multiple services as defined in the `docker-compose.yml` file:

- Documentation at localhost:8001

4.4 Frontend/Client-Side App

4.4.1 Frontend

4.5 Backend/Server-side App

4.5.1 Backend

4.6 Continous Integration Pipeline

These are just scripts that run whenever you do pull requests and successful merges. There are a couple of scripts that are currently configured see `.github/workflows`:

4.6.1 Automated Documentation Deployment `docs.yml`

This automatically deploys this documentation whenever `main` is updated with new changes.