Software Projects Milestone 7

Testing Plan

Group Number: 24

Concept Name: Smart Shelf Inventory Management System

This milestone will detail the plans for testing the software as you develop it. This will be based on the testing lecture, which uses material from Ian Sommerville's Software Engineering book.

For this milestone, you need to present your plans for:

1. Unit testing, where individual program units or object classes are tested. Unit testing should focus on testing the functionality of objects or methods. Every unit should be tested using a variety of test data.
2. Component testing, where several individual units are integrated to create composite components. Component testing should focus on testing component interfaces.
3. System testing, where some or all of the components in a system are integrated and the system is tested as a whole. System testing should focus on testing component interactions.

As testing is implicit in every coding and integration task, it would make sense to detail the tests that will be applied to each functional component in the software architecture.

Our project has 4 main functional components – The User interface, the Web server, the MySQL database and the Arduino. Various aspects of each of these systems will be tested individually (usually with mock data) and then in combination. Until eventually the system is ready to be tested as a whole.

Unit Testing:

User Interface

- Confirmation of design layout with users.
- Ensure Proposed site map contains space for all MVP features
- Ensure Pages display as intended (on multiple browsers and live on the server).
- Display and scaling across a range of devices.
- Ensure UI elements can scale with data (using mock data)
- Ensure Pages can accept Dynamic content and have been converted to templates properly (testing with mock data)
- Ensure the UI templates contain space/options for all the MVP functionalities

Web Server

- Ensure it runs and page routing works as desired (requests don't die etc).
- Ensure Database is set up and accessible.
- Functions for processing data from Arduino created independently and tested with mock data.
- Functions for processing database data created independently and tested with mock data (range of scenarios).

MySQL database

- Test that database ERD fields are correct - can every MVP operation be performed using the data stored in the proposed database
- Create the database and test that ERD schema is implemented properly
- Enter mock data - ensure autofill data is performed where required
- Ensure relationships between tables are correct and all data required by operations accessible via queries.
- Test the queries with a variety of mock data.

Arduino

- Test the output of the load cell to the Arduino.
- Test the range and sensitivity of the load cell.
- Test generating weight data
- Test Arduino internet connectivity (and static IP?)
- Test formatting of Arduino output (Arduino SQL)

Component Testing

Arduino -> WebServer -> Database

- Test the Arduino can connect and send data to the web server.
- Test the data is processed correctly by functions
- Test the data is input correctly to the database.
- Test the Arduino data is accurate using items of known weight
- Test Arduino refresh frequency
- Test that the Arduino can be set to refresh frequently (in case of need for a live demo)
- Lots of testing and generating of weight data for use on other (inactive) shelves.

Database -> Webserver -> UI

- Test that data from the database can be requested by the middleware and displayed in the UI in the desired way (e.g. bars showing the current shelf occupancy scaling properly)
- Test these behaviors with odd data (-ve numbers, very large numbers etc)
- Ensure the UI templates all work with the database data they are meant to display
- Ensure The functions processing data from the database are called properly and send the correctly formatted data to the UI (using data with known expected outputs)

UI -> Webserver -> Database -> Webserver - > UI

- Ensure requests from users generate the right requests to the database which are then passed by the correct functions and returned as correct responses
- Ensure User is able to input to relevant fields in the database (e.g. Item names)
- Ensure when user selects different display options these changes are entered into the relevant fields in the database

System Testing:

- Test setting up a variety of items on the database
- Test performing every MVP functionality several times
- Test Accurate display of the most recent shelf weight data from the shelf.
- Ensure the system can handle frequent updates from the Arduino (as may be preferable for a live demo, but not in normal operation)