Men’s Shed Web Application Mid-Point Review

SOC09109 2022-3 TR2 001 - Group Project

Table of Contents

Project Description

Dunfermline Men's Shed objective.

The committee of the Dunfermline Men's Shed (MS) aspires to create a comprehensive and engaging resource for their community The MS has access to some additional facilities and is keen to attract additional members and promote diverse activities. They already have selection of interest groups away from the typical MS Activities of woodwork or metalwork. As part of their aspirations, the intended application should provide a platform upon which they can build as the need arises.

The project

Following the initial meeting with MS on the 2nd of February 2023 MS representatives highlighted their vision for the Dunfermline association, their long-term objectives, and aspirations. The MS team want to build an exemplar of a modern adaptive association embracing the wider community and diverse interests. i.e., Computer club, guitar club and workshop facilities. But also, a lending library for the many tools, books, games, videos, and publications that have been donated by the people of the community at large.

The proposed application is a tool for both the members to use and the management of the organisation to track the donated equipment and manage that equipment, including the facility at large and the membership.

The feature of the application requested includes the following:

- The app should provide borrowing/reserving features. This shows when something is in use and when it will be available again. This requires the following database and app features:
- Data administrator portal
- Add/update/retire resources.
- User registration Resource calendar
- Resource status (e.g., Physical Library items still to be audited with pictures, ISBN digits). Mobile Power Tools tested and audited.
- The software should be adaptable.
- The Shed workshop should be bookable for 15 minutes. The Shed and activity determine the maximum time. Deliverables
- Design and implementation of the system back-end (including the database)
- Design and implementation of the app, which should run at least on Android and desktop devices.
- Source code
- Documentation as appropriate

Agreement then was confined to the following list.

1. Core application with:
    a. User interface
    b. Database
    c. Administrators access to maintain Data.
    d. Administrators User interface.

The MS has provided the following items to assist with the development of the application:

1. Management personal information for the Administrator's access
2. A spreadsheet with tools and images to display as being available for use.

label	implementation
done	SQLite backend implemented
done	restrict users from accessing dangerous tools that have or could have a severe impact on health and safety
done	date of the last time an admin checked that an item
Needs done	the booker has: read the health and safety requirements of the item, has the proper training required to use the item requested, or the PPE is needed for the item.
Risks	Had to change from SQL to SQLite because it works better with python
Risks	The image is stored in the database at the SQL level will just be linked to the picture.

Table 1 Structure & Possible Risks of developing the backend.

Mid-Point Review Structure, Functionality & Challenges of Implementing the Backend.

The MS application has been structured following the design plan mentioned previously in the initial PIR report. As you can see from appendix 4, I have implemented this design of the packaged file structure using four main directories.

The app directory holds the main file run.py that initialises the web application. The app directory also contains the main directory called MS, which stores the files for the web applications. The MS directory is located within the app directory, allowing for communication between the two directories creating a file-packaged structure for better readability and robustness. The main database for the men's shed web application is also located in the app directory. This is used to store and retrieve the men's shed data used for the web application. The database was created using the CreationDataBase.sql file to create the tables along with the attributes associated with each table. This will allow the given men's shed data to be uploaded and stored to the created SQLite database for use in the men's shed web application.

The MS directory that is stored within the app directory is used to store the main files for the men's shed web application along with the templates directory which is used to hold the html files for the web app and the static directory which is used to hold the CSS and image files. The __init__.py file has been used to initialise the flask application; this file will also be used to store the web apps secret key and any other software that needs to be initialised with the web app. The forms.py will be used to store any flask forms needed for the web app. Forms such as users sign up and login forms can be stored here. The modules.py is used to store any code that may affect the database during the web application development. The routes.py page is the main page of the web application; this page make use of html files to display the pages to the users. This allows us to implement features to each html page. The routes page also holds the function that establishes a connection to the database for use of storing and retrieving data.

Some of the challenges I have faced while developing the backend of the men's shed web application have been correctly setting up the packaged file structure. Each file and directory must be correctly imported from one another to allow communication between each file. If a file or directory does not establish a connection, then this can cause the web app not to work correctly or not work at all. It has been a crucial part of the development to understand when features require to communicate with other files and directories and to know which ones to import from. It is also important for the group to clearly comment on what the functions do and what files or directories they import from so other members of the group can understand the packaged file structure.

Another challenge that has been noted while developing the file-package structure is creating and establishing a secure connection to the SQLite database that is held within the app directory. At the beginning of development, we tried to create and initialise the database using a python file held within the MS directory; after some development, I found that the SQLite database could be created from the terminal using the SQL file that has been created for the database. Once this was established, I had to find a way to ensure that a database connection could be established within the packaged file structure. Using a python function held within the routes.py file that was able to secure a connection with the database, we are now ready to store and retrieve data.

Overall, the development and implementation of the men's shed web application has been a challenging yet rewarding task, with the development of the web app steadily progressing to a well-designed, functional and robust web application that the men's shed team will be able to make excellent use of.

User Interface

In the development of our application, we wanted to consider the user base of our product. The Men's Shed organisation is built on allowing individuals to access a wide range of resources, no matter their experience or ability. Therefore, we believed that it was highly important that our website reflected this accessibility. When it came to the layout and design of our page, we decided on some key ideas to prioritise:

1) The pages contain information which is concise and helpful
2) The layout is readable and easy to navigate
3) The style of each page looked professional

These were the key points in designing our interface. Our page should be informative but only contain necessary information. It should be clear and easy to navigate for users of all different abilities and experiences. And finally, considering that our application was to be delivered to a real-world client for professional use, we should aim to develop an interface which looked professional and well-made. We aimed to use colour schemes, fonts and a layout which would reflect this.

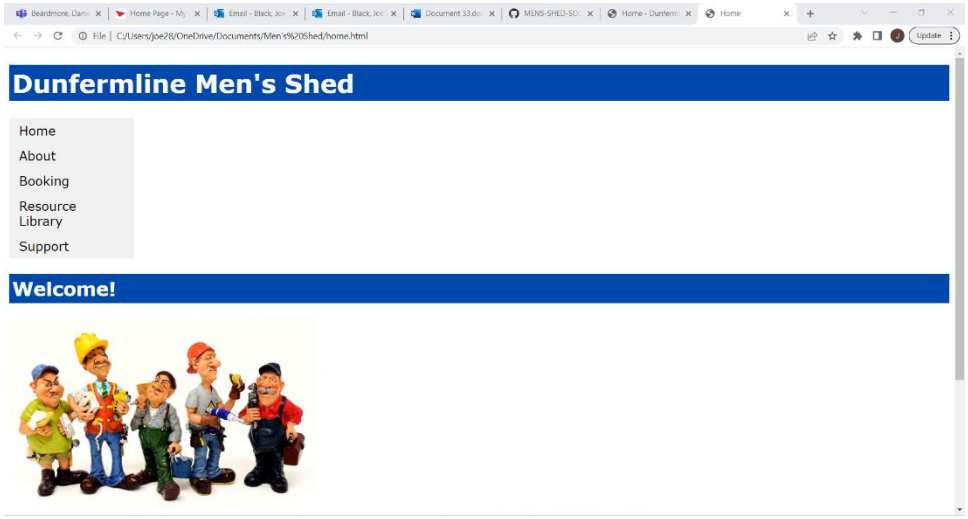
We ensured that our interface contained all the necessary information without overcrowding the pages with useless information. Our users may suffer from poor eyesight or dyslexia, so it was important to limit the amount of text where possible and consider other ways in which we could communicate the features to maintain high accessibility, such as images.

As for navigation, we decided that we should keep the number of pages to a minimum where possible. To keep the navigation straightforward and clear, six main pages were decided on;

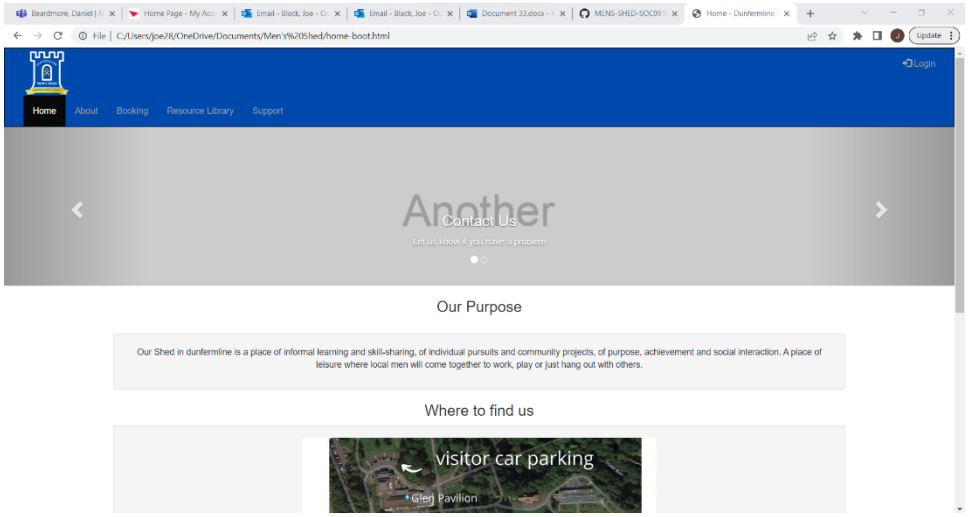
1) Home
2) About
3) Resource Library
4) Booking
5) Support
6) Login

This would allow users to clearly determine where to find each function or piece of information on the application without overcomplicating things. With this in mind, we began to implement these pages aiming towards a working prototype.

When corresponding with the back-end development team, the suggestion was made that we change our method of implementation. Before, we were using raw HTML code; pages made from scratch, but we had been recommended to use Bootstrap and Jinja2 templates as an alternative. This method of HTML implementation is much more commonly used when developing Python Flask web applications, as it is based on pre-made templates which can be edited to suit your application. We made the change to Bootstrap templates after this meeting, and this is the method of implementation we have used since. This eliminated many of the formatting/style issues we were facing with our page and makes for an overall clearer and more professional-looking web app.



The Home page layout when designed without Bootstrap templates



The web page design following the change to Bootstrap

On top of creating a more visually appealing layout, this also makes linking each page with the Python Flask application much easier, meaning that the implementation process became much more straightforward for both the front-end (user interface and web page implementation) and back-end (database and Python Flask application implementation) teams.

Security and Testing Mid-Point Review

This section outlines the initial requirements, current progress and outstanding objectives for the security and testing of the project.

1. Initial Requirements

With regards to security and testing the following objectives were planned out.

- Adhere to data and design best practices during development.
- Database information to be stored securely using encryption.
- Passwords stored in a hash format using suitable algorithm.
- Strict input validation.
- Automatic HTTP redirection to HTTPS.
- Password policy enforcement for user accounts.
- Testing framework documentation and deployed application testing.

2. Current Status

The application structure is set, and the team is following best practices to the best of their abilities during the development.

The password hashing algorithm to be used in the application has been agreed with the team to be Bcrypt. Currently, the user login functionality is not implemented; however, the basic Bcrypt initialisation has been placed.

Input validation has not been implemented to a great degree yet, as the focus has been on producing the skeleton framework of the application so far.

There is no live deployment of the application so far. All interactions with the app are currently local.

User account functionality is not active at this current moment.

A testing framework has been created to cover functionality testing, user experience and security testing. This is almost ready to go when a deployed version of the app is completed.

3. Outstanding Objectives

The following are requirements still to be completed for the security and testing section.

- Ensure that information stored on the database is encrypted.
- Full implementation of Bcrypt into the user authentication process.
- Implementation of strict input validation.
    - Automatic HTTP redirection to HTTPS on live connections.

- Password policy enforcement for user accounts.

Risks

Some of the possible risks when developing the backend of the Men's Shed Application is the risk of having incomplete admin features due to time constraints; this may lead to crucial functions not working correctly. This could lead to the Men's Shed company failing to reach goals such as having the ability to share and manage resources among multiple Men's Shed locations.

Another risk that may occur when developing the backend of the Men's Shed web application could be the decided file structure that I plan to implement. Errors may occur when importing each file within the directory to communicate with one another, which would lead to slower production and more bugs when implementing new functions.

Finally, one last possible risk that may occur during the development of the Men's Shed web app could be the possibility of modifications or an introduction of a bug into the SQL database that will be used for the web app. This could lead to one or more functions not working correctly if they are unprepared for such changes to occur.

Appendix 1, Follow-Up Register, an excerpt of a live excel document to record and mitigate Risks, Changes and Issues, contains a more detailed review of the potential risks.

Deliverables Map

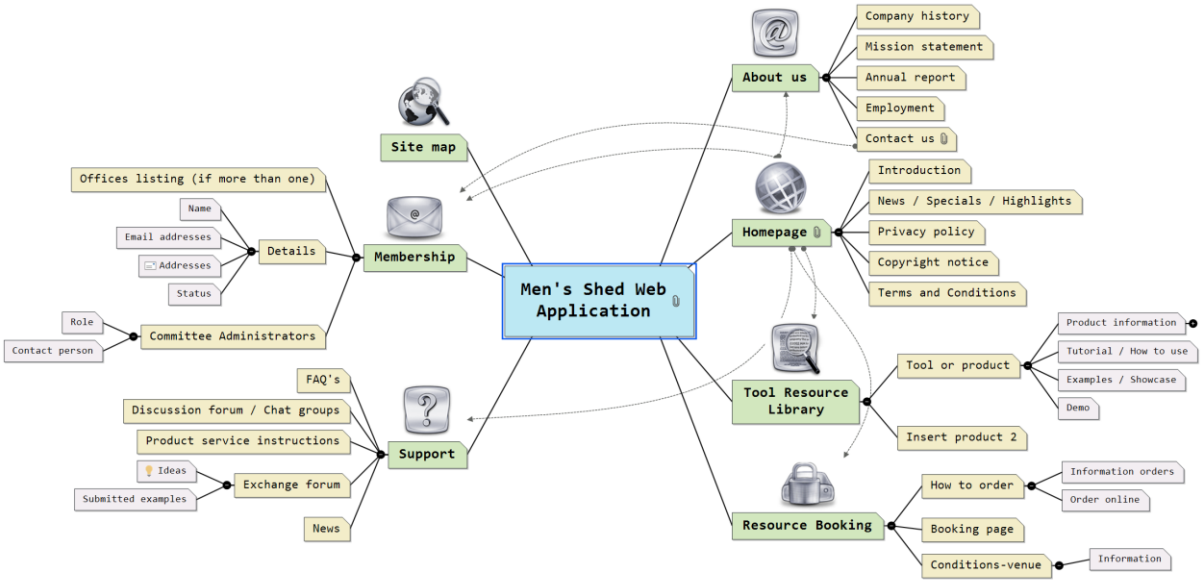


Figure 1 Deliverables Map

Deliverable's timeline

Application progress aims; along with the development of the server-side administration (Backend team) structures and database implementation, the User Interface (UI) team have structured the UI development along the timeline detailed in the table below.

Week	Objectives
Week 8	Adjust features implemented by client request.
Week 9-10	Testing of front-end UI, identifying any potential errors and correcting code
Week 11	Final testing, prepare interface for final submission.
Week 12	Submit code

Table 2 Deliverable timeline

Stakeholder List

The stakeholders for this project include the following personnel.

Role	Name	Organization
Project manager	John Johnston	Napier University PT
Main Client Contact	Ron Skirving	Men’s Shed Committee
Sponsor	Iain Donald	Napier University Tutor
Project Team	Jonathan Cloke	Napier University PT
Project Team	Rory Mackintosh	Napier University PT
Project Team	Joe Black	Napier University PT
Project Team	Daniel Beardmore	Napier University PT
Project Team	Duncan Hastie	Napier University PT
Men’s Shed	Committee members	Men’s Shed Committee

Table 3 Stakeholder List

Appendix 1 Follow-Up Register

Type	Cause	Effect	Impact	Likelihood	Importance	Response	Response	Custodian
R	Coding	Inefficient application and failures. Incorrect programming, being under pressure	70	50	2	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	Joining coding elements between team members	Inefficient application and failures. Incorrect programming, being under pressure	70	50	2	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	SQL Scripting errors	Inefficient application and failures. Incorrect programming, being under pressure	70	50	2	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	Broken Object Level Authorization	Integrity of the site being compromised	60	30	1	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	Broken User Authentication	Unauthorised access to the site	50	30	1	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	Broken Function Level Authorization	Application failure	40	30	1	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
R	Security Misconfiguration	Application security compromise	40	30	1	Avoid	Testing and Adjust	JJ, RM, JB, JC, DB, DH
C	Template HTML schema	Continuity simplicity and reducing application size	70	80	1	Accept	Change methods	JB, DB, RM, DH.
C	Database Technology SQLite	For the common ground and robustness of the database	70	80	1	Reduce	Alternative to SQL Alchemy	RM, DH, JJ

Appendix 2 Project Mid-term Peer Review

Reviewer: **Team:**

Reviewee: **Team:**

Date of review:

Project description

Reviewer's comments and recommendations

Response and actions taken

Deliverables map

Reviewer's comments and recommendations

Response and actions taken

Follow-up register

Reviewer's comments and recommendations

Response and actions taken

Quality of document (clarity, presentation, etc.)

Reviewer's comments and recommendations

Response and actions taken

Appendix 3 Client Project Initiation Report Feedback

Appendix 4 Packaged File Structure Diagram