|  |
| --- |
| Name: PMB Jayakody |
| Student Reference Number: 10899554 |



|  |  |  |
| --- | --- | --- |
| Module Code: PUSL2020 | Module Name: Software Development Tools and Practices | |
| Coursework Title: PUSL2020 Software Development Tools and Practices Coursework Report | | |
| Deadline Date:08/04/2024 | | Member of staff responsible for coursework:  Mr. Rasika Ranaweera and Mrs. Pavithra Subhashini |
| Programme: Bsc(Hons) Software Engineering – Plymouth University | | |
| Please note that University Academic Regulations are available under Rules and Regulations on the University website [www.plymouth.ac.uk/studenthandbook](http://www.plymouth.ac.uk/studenthandbook). | | |
| Group work: please list all names of all participants formally associated with this work and state whether the work was undertaken alone or as part of a team. Please note you may be required to identify individual responsibility for component parts.   |  |  |  | | --- | --- | --- | | **Student Name** | **NSBM ID** | **Plymouth ID** | | PMB Jayakody | 27353 | 10899554 | | JADD Lakdineepa | 26432 | 10899597 | | DNL Premathilaka | 27509 | 10899668 | | YGA Amarasinghe | 24735 | 10899158 | | RAVL Perera | 26852 | 10899656 | | WMRRB Wijerathne | 24802 | 10899730 |   ***We confirm that we have read and understood the Plymouth University regulations relating to Assessment Offences and that we are aware of the possible penalties for any breach of these regulations. We confirm that this is the independent work of the group.***  Signed on behalf of the group: PMB Jayakody | | |
| Individual assignment: ***I confirm that I have read and understood the Plymouth University regulations relating to Assessment Offences and that I am aware of the possible penalties for any breach of these regulations. I confirm that this is my own independent work.***  Signed : | | |
| Use of translation software: failure to declare that translation software or a similar writing aid has been used will be treated as an assessment offence.  I \*have used/not used translation software.  If used, please state name of software………………………………………………………………… | | |
| **Overall mark \_\_\_\_\_% Assessors Initials \_\_\_\_\_\_ Date\_\_\_\_\_\_\_\_\_** | | |

\*Please delete as appropriateSci/ps/d:/students/cwkfrontcover/2013/14

**Table of Contents**

1. Introduction / Overview……………………………………………………..4
2. Technologies…………………………………………………………………5-7
3. Diagrams……………………………………………………………………..8-9
4. Evidence of development…………………………………………………….10-17
5. Description of Test Cases……………………………………………………18-22
6. Structure and role of any mock objects………………………………………23
7. How to run the unit tests and the integration tests……………………………24-25
8. References……………………………………………………………………..26
9. Workload matrix………………………………………………………………26

**Introduction / Overview**

NSBM Green University is embarking on the development of a comprehensive web application. This initiative aims to provide a user-friendly platform for both incoming students seeking suitable housing options and landlords/rental agencies wishing to advertise their properties. Under the guidance of the new warden, this project seeks to enhance the overall experience of students transitioning to university life by facilitating easy access to accommodation near the campus.

The web application features functionalities tailored to four distinct user groups: landlords, the warden, students, and a Web Master (administrator). Landlords have the ability to register, upload property details, and manage rental requests from students. The warden, serving as the overseer of the accommodation process, is empowered to review and approve advertisements, ensuring they meet the university's standards. Students, the primary beneficiaries of this platform, have access to a comprehensive list of available accommodations, complete with detailed descriptions and the option to reserve their preferred choice. Additionally, the Web Master plays a crucial role in administering user accounts and disseminating relevant information to improve awareness about housing options.

**Technologies**

The NSBM Green University accommodation web application is developed using a variety of technologies to ensure a seamless user experience and efficient functionality.

**ASP.NET Framework:** ASP.NET provides the framework for building web applications. It includes features for routing requests, managing sessions, and handling authentication and authorization.

**SQL Server Database:** The SQL Server database stores the application's data. It is accessed through Entity Framework or other data access technologies in ASP.NET.

Create, update, and delete (CRUD) activities are carried out via stored procedures in the web application designed using ASP.NET with a SQL Server database. Stored procedures are SQL queries that are precompiled and stored in the database.

1.

A computer screen with a white screen

Description automatically generated

**Create Procedure:** The create procedure is responsible for adding a new accommodation listing to the database. It would typically accept parameters such as the property details and use an INSERT statement to insert a new record into the accommodations table.

A computer screen with a white screen

Description automatically generated 2.

**Update Procedure:** The update procedure is used to modify an existing accommodation listing. It would accept parameters and updated details, and use an UPDATE statement to update the corresponding record in the database.

3.

A screenshot of a computer

Description automatically generated

**Delete Procedure:** The delete procedure is used to remove an accommodation listing from the database. It would accept parameters and use a DELETE statement to delete the corresponding record.

**Google Maps API:** The Google Maps API is a key component of the NSBM Green University accommodation system web application, providing interactive maps and location-based services.

**Map Display:** The Google Maps API is used to display an interactive map on the web application's interface. This map shows the locations of available accommodations around the NSBM Green University campus.

**Marker Placement:** The API allows for the placement of markers on the map to indicate the locations of boarding houses. Each marker represents a specific accommodation listing, and users can click on these markers to view more details about each property.

**Geocoding:** The Google Maps API's geocoding functionality is used to convert addresses (e.g., boarding house locations) into geographic coordinates (latitude and longitude). This allows for accurate placement of markers on the map.

**Map Controls:** The API provides various controls, such as zoom and pan controls, to allow users to navigate the map easily and explore different areas around the university campus.

**Integration with Database:** The Google Maps API is integrated with the web application's database to dynamically update the map with new accommodation listings and their corresponding markers.

**Diagrams**

ER diagram

A diagram of a company

Description automatically generated

**Assumptions:**

We assume that the Student and Warden have a National ID as the primary key.

One student can place only one reservation at once.

As well one property has more reservations. Because there are more rooms in a boarding house. Therefore, it can have more reservations.

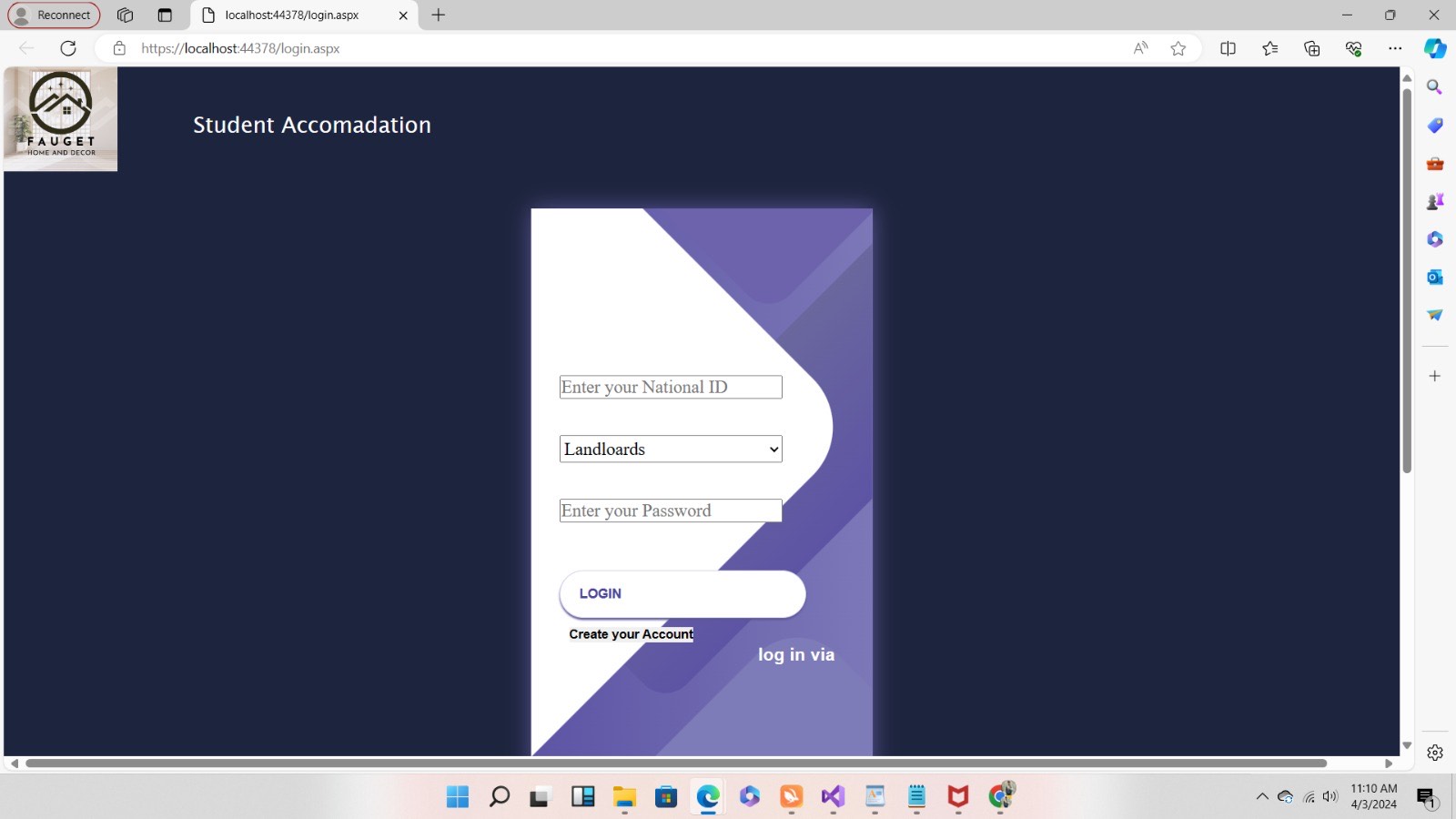
We assume that one warden has more properties.

Use case diagram

A diagram of a room

Description automatically generated

**Evidence of development**

****

The login page allows users to access the system by entering their national ID and password. Additionally, users are required to select their user type (e.g., landlord, student, warden) to determine their access rights within the system. The page provides a secure authentication process to ensure that only authorized users can log in and access the system's features based on their user type.

**A computer screen with a white background

Description automatically generated**

Developers**:** PMB Jayakody

DNL Premathilake

**A screenshot of a computer

Description automatically generated**

The landlord's signup page allows individuals to create a new account to list their properties on the accommodation-finding website. Users are required to provide their national ID, full name, email, phone number, address, landlord pass number, and password. This information is used to authenticate the landlord's identity and manage their property listings on the platform.

Developers: YGA Amarasinghe

DNL Premathilaka

1. **A screenshot of a computer

   Description automatically generated**

The landlord page provides a platform for landlords to manage their property listings. Landlords can enter their national ID, select the type of accommodation (e.g., house, apartment), provide a phone number, specify the house address, set the student accommodation fee, define the duration of the rental agreement, and add a description of the property. The page includes buttons for inserting new listings, updating existing listings, deleting listings, and retrieving listings for viewing or editing. This interface allows landlords to easily manage their property information and make updates as needed.

Developers: YGA Amarasinghe

DNL Premathilaka

1. A screenshot of a computer

   Description automatically generated

The warden's signup page allows individuals to create a new account to manage property listings on the accommodation-finding website. Users are required to provide their national ID, full name, email, phone number, address, and password. This information is used to authenticate the warden's identity and manage property listings on the platform.

**A computer screen with a white screen

Description automatically generated**

Developers: JADD Lakdineepa

WMRRB Wijerathna

A screenshot of a computer

Description automatically generated

The NSBM Green University accommodation website offers a convenient platform for students to find suitable boarding houses near the campus. With an easy sign-up process, students can explore a variety of accommodations, view details and photos, and contact landlords directly through the website. The interactive map feature helps students locate available boarding houses and make informed decisions about their accommodation. Sign up today to find your ideal place to stay during your studies at NSBM Green University.

A computer screen with a white screen

Description automatically generated

Developers: JADD Lakdineepa

WMRRB Wijerathna

A screenshot of a map

Description automatically generated

The map UI in the web application presents the locations of boarding houses around NSBM Green University. It provides an interactive map where students can view markers representing the different boarding house locations. Users can zoom in and out of the map to see more details and click on markers to view more information about each boarding house, such as photos, facilities, and rental prices. The map UI enhances the user experience by visually displaying the available accommodations and helping students easily explore and select the most suitable option for their needs.

Developers: RAVL Perera

PMB Jayakody

1. A screenshot of a computer

   Description automatically generated

The map UI in the NSBM Green University accommodation website provides a user-friendly interface for students to view the locations of boarding houses near the campus. The map displays markers representing each boarding house, and upon clicking on a marker, students can view details such as photos, facilities, rental prices, and the landlord's contact phone number.

The map UI also includes options to accept or delete a boarding house listing. Landlords can accept reservation requests directly through the map UI, and if a property is no longer available, they can easily delete the listing. This functionality enhances the user experience by providing a comprehensive view of available accommodations and enabling direct communication with landlords.

A computer screen with a white screen

Description automatically generated

Developers: RAVL Perera

PMB Jayakody

**Description of Test cases**

There are test cases for the accommodation-finding website we designed:

1. Landlord Registration:

Test that landlords can successfully register for an account.

Verify that all required fields (such as name, email, and password) are validated.

Ensure that duplicate registrations are prevented.

1. Property Listing:

Test that landlords can add a new property listing.

Verify that all required fields (such as property details, images, rental information) are included.

Ensure that landlords can update and delete their property listings.

1. Student Browsing:

Test that students can browse the list of available properties.

Verify that students can view property details, including images and rental information.

Ensure that properties are displayed correctly on the map interface

1. Reservation Request:

Test that students can send a reservation request for a property.

Verify that landlords receive notification of the reservation request.

Ensure that landlords can accept or reject reservation requests.

1. Warden Approval:

Test that the warden can view all property listings.

Verify that the warden can select an individual property to view details.

Ensure that the warden can approve or reject property listings.

1. User Account Management:

Test that users (landlords, students, warden) can successfully log in to their accounts.

Verify that users can update their account information (such as email, and password).

Ensure that users can delete their accounts if needed.

1. Admin Functionality:

Test that the admin can create accounts for landlords, students, and the warden.

Verify that the admin can post articles on the website for awareness.

Ensure that the admin can manage user accounts and access permissions.

1. System Performance:

Test the website's performance under different loads (e.g., number of users, simultaneous property listings).

Verify that the website responds quickly and efficiently.

1. Compatibility Testing:

Test the website's compatibility with different web browsers.

Verify that the website is responsive and displays correctly on different devices (e.g., desktop, tablet, smartphone).

1. Security Testing:

Test the website's security measures (e.g., secure login, data encryption).

Verify that user data is protected and not accessible to unauthorized users.

These test cases cover a range of functionalities to ensure that the accommodation-finding website works correctly and meets the needs of its users.

Below are some of the test cases designed.

1. Test Case ID: AC\_001

Test Description: Verify that landlords can successfully register for an account and that all required fields (such as name, email, and password) are validated. Ensure that duplicate registrations are prevented.

Prerequisite: The website registration page is accessible and functional.

Test Procedure:

1. Navigate to the landlord registration page.
2. Enter valid data for all required fields (name, email, password).
3. Click the "Register" button.
4. Verify that the registration is successful and a confirmation message is displayed.
5. Attempt to register again with the same email address used in step 2.

Input Data:

Name: John Doe

Email: johndoe@example.com

Password: Test123!

Expected Result:

The registration process should be successful.

A confirmation message should be displayed indicating successful registration.

Duplicate registrations with the same email should be prevented.

Actual Result:

The registration process is successful.

A confirmation message is displayed.

Duplicate registration with the same email is prevented.

Status: Pass

Severity: Medium

Defect ID: None

Executed by: YGA Amarasinghe

Type: Functional

Pos / Neg: Positive

Written By: PMB Jayakody

Comments: The test case passed successfully, and the duplicate registration was prevented as expected.

1. Test Case ID: AC\_002

Test Description: Verify that landlords can add a new property listing, including all required fields such as property details, images, and rental information. Ensure that landlords can update and delete their property listings.

Prerequisite: The landlord is logged in to their account and the property listing page is accessible.

Test Procedure:

1. Log in to the landlord account.
2. Navigate to the property listing page.
3. Click on the "Add New Property" button.
4. Enter all required details for the new property listing (property details, images, rental information).
5. Click the "Save" button.
6. Verify that the new property listing is added successfully.
7. Update the property listing with new information.
8. Verify that the property listing is updated successfully.
9. Delete the property listing.
10. Verify that the property listing is deleted successfully.

Input Data:

Property details: Address, description, facilities.

Images: Upload at least one image of the property.

Rental information: Rental price, availability.

Expected Result:

The new property listing should be added successfully.

The updated property listing should reflect the new information.

The property listing should be deleted successfully.

Actual Result:

A new property listing is added successfully.

The property listing is updated with new information.

The property listing is deleted successfully.

Status: Pass

Severity: Medium

Defect ID: None

Executed by: YGA Amarasinghe

Type: Functional

Pos / Neg: Positive

Written By: DNL Premathilaka

Comments: The test case passed successfully, and landlords can add, update, and delete their property listings as expected.

1. Test Case ID: AC\_003

Test Description: Verify that students can browse the list of available properties, view property details including images and rental information, and ensure that properties are displayed correctly on the map interface.

Prerequisite: The student is logged in to their account and the property browsing page is accessible.

Test Procedure:

1. Log in to the student account.
2. Navigate to the property browsing page.
3. Browse the list of available properties.
4. Click on a property to view details.
5. Verify that property details are displayed correctly, including images and rental information.
6. Check the map interface to ensure that properties are displayed correctly based on their locations.

Input Data: None

Expected Result:

Students should be able to browse the list of available properties.

Property details, including images and rental information, should be displayed correctly.

Properties should be displayed correctly on the map interface.

Actual Result:

Students can browse the list of available properties.

Property details are displayed correctly, including images and rental information.

Properties are displayed correctly on the map interface.

Status: Pass

Severity: Low

Defect ID: None

Executed by: YGA Amarasinghe

Type: Functional

Pos / Neg: Positive

Written By: RAVL Perera

Comments: The test case passed successfully, and students can browse properties and view details as expected.

**Structure and role of any mock objects**

In the context of software development and testing, mock objects are simulated objects that mimic the behavior of real objects in a controlled way. They are often used in unit testing to isolate the code being tested and to simulate the behavior of external dependencies. In developing the accommodation-finding web application, we could use mock objects for the following purposes:

Database Mock:

**Structure:** A mock database object that mimics the behavior of a real database, allowing us to test database interactions without actually querying a database.

**Role:** Used to simulate database operations such as saving and retrieving data, ensuring that our code interacts correctly with the database layer.

Email Service Mock:

**Structure:** A mock email service object that simulates sending emails without actually sending them.

**Role:** Used to test email notification functionality, ensuring that the website can send notifications to landlords and students without actually sending real emails.

Map Service Mock:

**Structure:** A mock map service object that simulates map functionality without actually interacting with a mapping API.

**Role:** Used to test map integration on the website, ensuring that properties are displayed correctly on the map interface.

Authentication Service Mock:

**Structure:** A mock authentication service object that simulates user authentication without actually verifying user credentials.

**Role:** Used to test user authentication and authorization functionality, ensuring that users can log in and access their accounts.

Notification Service Mock:

**Structure:** A mock notification service object that simulates sending notifications to users without actually sending them.

**Role:** Used to test notification functionality, ensuring that users receive notifications about reservation requests and property updates.

By using mock objects, we can isolate the code being tested from external dependencies, making our tests more reliable and easier to maintain. Mock objects allow us to simulate complex interactions in a controlled environment, helping us to identify and fix bugs more effectively.

**How to run the unit tests and the integration tests**

Here is the breakdown of how to run unit tests and integration tests, along with functional test plans and a critical analysis of the test strategy:

Running Unit Tests:

**Tools:** Use a unit testing framework such as JUnit (for Java), NUnit (for . NET)

**Procedure:**

Write unit tests for individual components or classes of the software.

Execute the unit tests using the testing framework.

Analyze the results to ensure that each unit of code behaves as expected in isolation.

Running Integration Tests:

**Tools:** Use a testing framework that supports integration testing, such as Selenium (for web applications), Postman (for APIs), or Mockito (for Java).

**Procedure:**

Write integration tests that verify the interaction between different components or modules of the software.

Execute the integration tests using the testing framework.

Analyze the results to ensure that the integrated components work together correctly.

Functional Test Plans:

**User Registration:**

Test that users can register for an account successfully.

Verify that registration form validation works correctly.

**Property Listing:**

Test that landlords can add new property listings.

Verify that property details are saved correctly in the database.

**Reservation Request:**

Test that students can send reservation requests for properties.

Verify that landlords receive notification of reservation requests.

**Map Integration:**

Test that properties are displayed correctly on the map interface.

Verify that users can interact with the map to view property details.

Critical Analysis of Test Strategy:

**Unit vs. Integration Testing:** We have used a combination of unit and integration testing to ensure that both individual components and integrated modules work correctly. This approach allows us to catch bugs early in the development process and ensure that the software functions as expected in a real-world environment.

**Functional Test Plans:** Our functional test plans cover key functionalities of the software, such as user registration, property listing, reservation requests, and map integration. By testing these functionalities, we can ensure that the software meets the requirements of the stakeholders and provides a seamless user experience.

**Automation:** Where possible, we have automated our tests using testing frameworks and tools. Automation helps us to run tests more efficiently and frequently, reducing the risk of human error and ensuring consistent test results.

**Continuous Testing:** We have integrated testing into our development process, running tests continuously as new code is added or modified. This approach helps us to identify and fix bugs early, reducing the cost and effort of fixing bugs later in the development cycle.

Overall, our test strategy focuses on ensuring the reliability, functionality, and usability of the software, while also considering efficiency and scalability in our testing approach.

**References**

Wadepickett (no date) *ASP.NET documentation*, *Microsoft Learn*. Available at: https://docs.microsoft.com/en-us/aspnet (Accessed: 07 April 2024).

(No date) *ASP.NET MVC. the tutorial*. Available at: https://www.tutorialspoint.com/asp.net\_mvc/asp.net\_mvc\_tutorial.pdf (Accessed: 07 April 2024).

CodeProject (no date) *ASP.NET*, *CodeProject*. Available at: https://www.codeproject.com/KB/aspnet/ (Accessed: 07 April 2024).

*What is ASP.NET core?: .NET* (no date) *Microsoft*. Available at: https://dotnet.microsoft.com/learn/aspnet/what-is-aspnet-core (Accessed: 07 April 2024).

*Newest ‘asp.net’ questions* (no date) *Stack Overflow*. Available at: https://stackoverflow.com/questions/tagged/asp.net (Accessed: 07 April 2024).

**Workload Matrix**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Plymouth Id** | **NSBM Id** | **Work contribution** |
| PMB Jayakody | 10899554 | 27353 | Login UI, Google API, Testing |
| JADD Lakdineepa | 10899597 | 26432 | Sign up UI related to the warden and student, Testing |
| DNL Premathilaka | 10899668 | 27509 | Login UI, Sign up UI related to the landlord, Testing |
| YGA Amarasinghe | 10899158 | 24735 | Sign up UI related to student and Landlord, Testing |
| RAVL Perera | 10899656 | 26852 | Sign up UI related to the warden, Google map API, Testing |
| WMRRB Wijerathne | 10899730 | 24802 | Sign up UI related to student and warden, Testing |