# Southampton Solent University

# Coursework Assessment Brief

# Assessment Details

|  |  |
| --- | --- |
| Module Title: | Object Oriented Design and Development |
| Module Code: | QH0543 |
| Module Leader: | Hardik Bakari |
| Level: | 5 |
| Assessment Title: | Individual Assignment – PointsOfInterest (Part Two) |
| Assessment Number: | AE2 |
| Assessment Type: | Software Development |
| Restrictions on Time/Word Count: | Around 1000-2000 words (guidance only) plus diagrams |
| Consequence of not meeting time/word count limit: | There is no penalty for submitting below the word count limit, but students should be aware that there is a risk they may not maximise their potential mark.  There is no penalty for submitting above the word limit |
| Individual/Group: | Individual |
| Assessment Weighting: | 60% |
| Issue Date: | 18 November 2023 |
| Hand In Date: | Friday 9th February 2024, 16:00 |
| Planned Feedback Date: | Within 20 working days of submission |
| Mode of Submission: | on-line |
| Number of copies to be submitted: | 1 |

Hit Tastic

# INTRODUCTION

Hit Tastic application is built on a technical architecture that provides a scalable and efficient platform for web applications. This project comprehends Java, JSP (Java Server Pages), JDBC (Java Database Connectivity), and SQLite.

1. Java: As the foundation, Java offers platform independence, which means Hit Tastic can run on any device that supports a Java Virtual Machine (JVM). This makes the application highly portable and versatile.
2. JSP: Used for building the dynamic content of the website, JSP allows for the seamless creation of HTML pages mixed with Java code. This enables a more interactive user experience without compromising the application's speed or security.
3. JDBC: used to connect the application with the SQLite database, enabling robust data management capabilities. JDBC provides a flexible framework for executing SQL queries, updating data, and retrieving results.
4. SQLite: A lightweight, file-based database, SQLite offers a simple and efficient way to store, query, and manage data without the need for a complex database server.

The application's architecture also incorporates servlets, entities, and DAO (Data Access Object) patterns:

* Servlets: Act as a bridge between the web requests from users and the server's responses. They play a crucial role in handling the logic of user authentication, directing users to the correct pages, and processing search queries.

A screenshot of a computer program

Description automatically generated

* Entities: These are Java classes that represent the application's data models, such as users, comments, and points of interest. They encapsulate the application's core data attributes and behaviors, making data management more organized and object-oriented.

A group of white boxes with red dots

Description automatically generated

* DAO: This pattern abstracts and encapsulates all access to the data source. DAO manages the connection with the data source to obtain and store data. This approach promotes a loose coupling between the application's business logic and the data access mechanisms, enhancing modularity and making the code more maintainable and testable.

A computer screen shot of a computer code

Description automatically generated with medium confidence

# PRESENTATION:

## DATABASE TABLES:

For the realization of this project, the database was created and populated in SQLite:  
  
A screenshot of a computer

Description automatically generated  
A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A computer screen shot of a black screen

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

## ENTITIES:

Entities are created with constructors, setter, and getter methods.

Point of interest and User entities are mainly being reused from the group project but integrated and adapted to this application.

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

A computer screen shot of a computer screen

Description automatically generated

## DATABASE CONNECTION:

To connect the database to the project, the JDBC SQLite JAR folder has been added as a dependency.

A screenshot of a computer program

Description automatically generated

A screenshot of a computer

Description automatically generated

The dependency has been added as a dependency in the pom.xml file.

A screen shot of a computer code

Description automatically generated

A class has been created for the connection:

For the implementation and development of this application, the database was created in a different folder from the project.

However, the database was moved to the same folder as the project to zip it. The URL path should then be:

“String JDBC\_URL = “jdbc:sqlite:HitTastic.db”;”

A yellow and black text

Description automatically generated

## EXECUTION:

The program initiates with the index page which Welcome the User and presents a menu for the user to be able to navigate the search option as a guest or to login.  
  
A screenshot of a computer program

Description automatically generated

A screenshot of a website

Description automatically generated

Each of the option transfers the user to the requested JSP page:

A computer code with many colorful text

Description automatically generated with medium confidence

If the user chooses to sign up, the JSP page is rendered with a form action, when a user interacts with the buttons, the form is submitted and passes the request to the Servlet which then handles the methods through the DAO.  
A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

The servlet retrieves the form data, opens a new connection to connect with the database, and takes the attribute to create a new User.  
If the user is registered successfully, it sends the user to the registration success that confirms successful sign-up and prompts to log in, otherwise displays an error. The servlet then closes the connection to avoid leaks.

A screenshot of a computer code

Description automatically generated

For demonstration purposes, to illustrate how the system recognise a simple user from an admin the two highlighted user and admin will be used throughout this report.

A screenshot of a computer

Description automatically generated

In the login the user is prompted with a form to input the email and password:

A computer screen shot of a program

Description automatically generated

When the form is submitted, the request is sent to loginServlet which checks if the user credentials are in the database and checks that both fields are filled with details. If the email or password is left empty by the user, or If credentials do not match an existing user in the DB, then the app prompts the user to fill both fields and is redirected to the login JSP page. If the credentials are correct and match one on the database, the system checks for the user role and renders different pages with different features.

A screenshot of a computer screen

Description automatically generated

If the user is an admin, it will be presented with the admin page:

A screenshot of a computer

Description automatically generated

A yellow text on a white background

Description automatically generated

A screenshot of a computer

Description automatically generated

The view all users button, redirect the request to the viewallusers Servlet

A screen shot of a computer code

Description automatically generated

Which retrieves all the user lists thanks to the get All Users method in the User DAO class:

A white screen with black text

Description automatically generated

and returns it to the View All Users Admin page, where the admin has the feature to modify the credentials of a user or delete the user.

A computer screen shot of a yellow key

Description automatically generated

A screenshot of a computer

Description automatically generated

If the admin decides to modify a user's credentials the form redirects the request to the Modify user servlet:

A computer screen shot of a code

Description automatically generated

A screenshot of a computer screen

Description automatically generated

Which handle the request and update the values in the data base.

A computer screen shot of a yellow circle

Description automatically generated

A yellow marker on a white background

Description automatically generated





If the admin wants to delete a user click on the delete button, and the admin is redirected to the delete user jsp page, which does a double check and confirms if the admin wants to delete the user, if it was a mistake, the admin can just exit from the delete page, or he can continue and delete the user.

The request will be sent to the delete User Servlet which checks for the ID of the user and performs a query to SQLite to delete the user from the database.

A computer screen shot of text

Description automatically generated

A screenshot of a computer screen

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

If the admin decides to delete the user, the user is deleted from the database successfully:

A screenshot of a computer

Description automatically generated

The view all comments button for the admin page, pass the request to the view all comments servlet, execute the query in the DAO class and renders a list of all the comments to the JSP page:

A computer code with yellow text

Description automatically generated

A yellow lines on a white background

Description automatically generated

A screenshot of a computer

Description automatically generated

All the comments can be approved by the admin or deleted. To achieve this, the Comment DAO executes the delete Comment method, where the query is to delete the user with that specific ID.

A screenshot of a computer

Description automatically generated

The view all points of interest for an admin user, the method get all points of interest is called by the servlet for the Point of interest DAO that queries all the points of interest from the point of interest table in the database and return the list to the JSP page

A screenshot of a computer

Description automatically generated

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

The View All Points of interest JSP page, allows the admin to add a new point of interest through a modal container. This form has fields for the name, location, and type of the point of interest. When submitted, the form sends a POST request to the Add New Point Of Interest Servlet.

A screenshot of a computer code

Description automatically generated

The add a new point of interest servlet, utilize the add new point of interest method from the point of interest DAO and after executing the query and add a new point of interest to the point of interest table in the database, return the list of all points of interest again.

A screenshot of a computer

Description automatically generated

A computer screen shot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

If to login is a normal user, the system recognises it and displays the user page

A screenshot of a login page

Description automatically generated

A screenshot of a computer

Description automatically generated

The normal user has only the option to search for a Point of interest or to visualize all the points of interest.

To enable the user to Search for a point of interest either by location or by type/ category, the PointOfInterest.jsp page, prompts the user to choose with a friendly drop menu from which he can choose type:  
  
A computer code with yellow squares

Description automatically generated with medium confidence

Based on the user's choice, to search for type, another drop menu is displayed prompting the user to select which type of point of interest he wants to display:

A screenshot of a computer code

Description automatically generated

A screenshot of a computer

Description automatically generated

To search for the location, all the alphabet letters are displayed:

A screen shot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

Both requests are transferred to the search POI Servlet, which manages both posts, get methods (), and process request()

A computer code with many lines

Description automatically generated with medium confidence

A screenshot of a computer code

Description automatically generated

It retrieves the methods from the Point Of Interest DAO class which executes two different methods based on the user input and returns the data to display.  
  
A screenshot of a computer code

Description automatically generated

A screen shot of a computer

Description automatically generated

Other features for the user to add a like or a comment to a relative point of interest are displayed below:

A screenshot of a computer

Description automatically generated

A screenshot of a computer code

Description automatically generated

A screenshot of a computer program

Description automatically generated  
  
  
A screenshot of a computer

Description automatically generated