Database Applications
ISYS1101/1102 | Semester 2 2024
Assignment 4: Build a Web Database Application

Assessment Type	Individual Assessment
Due Date	23:59 Sunday 27 October 2024
Demonstrations	During the Week starting on Monday 28 October 2024
Silence Period	Starts at 5:00PM Friday 25 October 2024
Weight	35%
Submission	Online, via Canvas. Submission instructions are provided on Canvas.

1 Overview

1.1 Assessment Criteria

This assessment will determine your ability to:

1. write MQL statements required for CRUD (create, read, update and delete) operations on a MongoDB database;
2. by embedding above MQL statements as appropriate, write a complete web application using HTML, JavaScript, Node.js, Express.js, MongoDB and any other required tools;
3. Demonstrate the functionality of your web database application.

1.2 Learning Outcomes

This assessment will assess on how you attained the following course learning outcomes:

CLO 1: apply advanced data analysis and modelling concepts, physical design, integrity, security and transaction management.
CLO 4: build an efficient database application with an emphasis on storage management, indexing and query optimisation;
CLO 6: develop a simple web-based interface for a database.

2 Assessment Details

2.1 Preparation Work

Mandatory:

You are required to be able to write code in HTML, JavaScript, Node.js, and Express.js scripting languages to build a fully-fledged web database application. More importantly, you should be able to use mongodb library within a Node.js program. In order to acquire this pre-requisite knowledge, you must complete Week 11 lab session.

We build the web application based on the queries you submitted for Assignment 2. If your queries did not work, it is a good idea to get them corrected on Mongo Compass before embedding them in this assignment work.

Optional:

The other important element in the MERN stack is React.js. We did not cover React in the preceding lab sessions. So, the use of React is optional. However, if you have previously used it in other courses or have a good working knowledge of React, you are allowed to use it to build the front-end of the application.

2.2 Assignment Task Description

Task 1: Build a simple web database application

The application you are required to build is based on sample_airbnb database you have been using in the past few labs as well as the Assignment 2.

Your web application will present AirBnB clients with an interface where they can filter listings based on their priorities and then will allow them to choose one listing from the presented list and add a new booking for their requested dates.

The application will have a minimum of three pages:

Homepage: Page 1

This page will have two parts: (1) the top section will consist of a simple form with three form input fields: Location, the type of the property and the number of bedrooms; (2) the bottom section will initially list some random property listings.

The location is a mandatory input. Type of property and number of bedrooms are dropdown lists. However, these two inputs are optional, i.e. the clients can choose to leave them unselected and submit the form.

After the form is submitted, the bottom part of the webpage will get refreshed with property listings that matches with the filtering criteria the client has submitted. For example, if they have chosen Barcelona as the location and left other two inputs empty, it will display all properties in the Barcelona market (address.market). If a client had filled all three fields (say, 3-bedroom apartments in Barcelona) then your application will display a further narrowed-down result set.

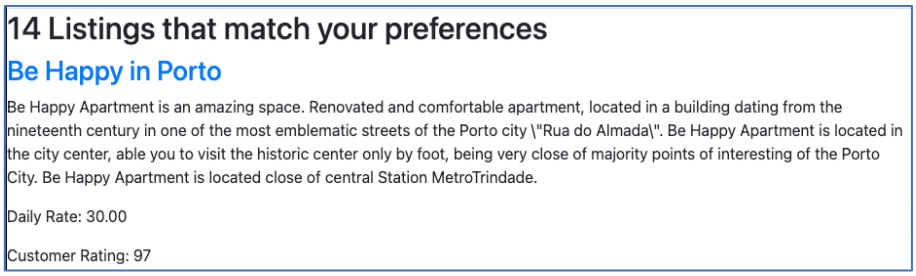
Each property listing on this page should comprise of the name of the property, summary, daily price, and review score rating (review_scores.review_scores_rating).

Each property listing's name is displayed as an active hyperlink, allowing the client to choose the property and proceed to the next stage (booking stage) of the application.

This hyperlink should carry the listing_id as a hyperlink query parameter (or URL parameter, e.g.: https://localhost:3000/bookings.html?listing_id=10083468) and will allow the bookings page to manage the bookings for the chosen property.

How to host: https://www.w3schools.com/nodejs/nodejs_get_started.asp

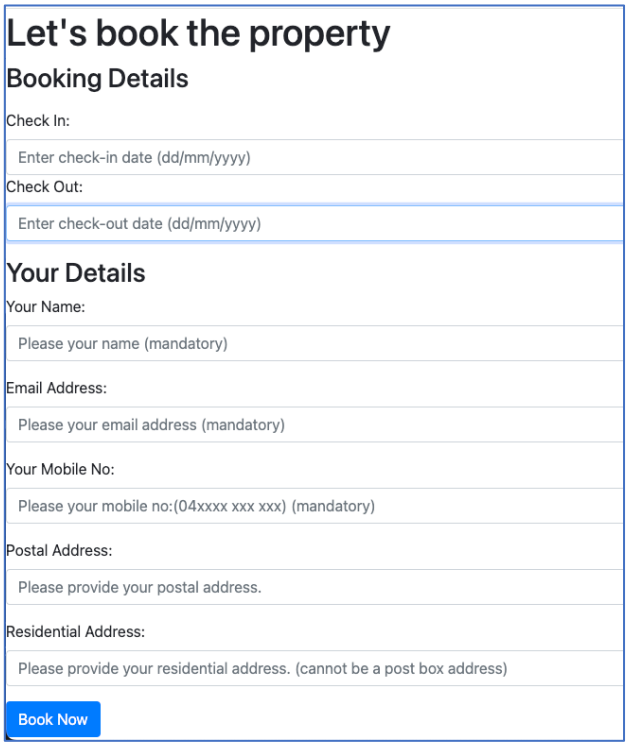
A typical listing will appear on this page as shown below. As you see fit, you may use css style sheets, Bootstrap or other css frameworks, to format the output.



Bookings page: Page 2

This page will also made up of a form which allows the clients to input booking start date, end date, client name, email address, daytime phone number, mobile number, postal address and home address.

To keep your web form simple, it is NOT a requirement in this assignment to enter the other information such as the deposit paid at the booking, the balance due, the due date for the balance payment, and number of guests, and guest details.



<- Use a WHILE LOOP to ensure

Booking Confirmation page: [age 3

After the booking information is submitted and new booking data is stored on the database, a simple booking confirmation will appear. This page will have a simple hyperlink to return to the homepage.

System requirements

Hosting

This is just an application development exercise, so, it is not required to host it in a proper web hosting platform. You can use Visual Studio Code as your interactive development environment and host it locally (say host it on port 3000 on localhost and accessed locally on your browser with homepage URL: localhost:3000/index.html) For more information on hosting it locally, refer to Week 11 lab sheet.

Database

You should use cloud-based MongoDB Atlas as your database backend.

Database Schema

The sample database "sample_airbnb" has one document collection called "listingsAndReviews" which contain basic information on property listings. However, it does not have provision for storing booking details. In order to store them, you will need to extend the data model used in this database. You have two choices: **embedding** booking information in the current listingsAndReviews collection, or use the **referencing** approach by having additional document collections. In assignment 2, you must already have explored the merits in both of these approaches. We do not have a preference between these two approaches. As long as your model can accommodate these data and be able to retrieve bookings for a given listings, you can choose either of these models.

Technology Stack

It is an assignment requirement to use Node.js with Express.js as your development platform. You are free to add React.js for frontend development. However, you do not lose marks for not using React.

---

3 Submission

1. DO NOT change the connection string, the markers should be able to run the application on your database. However, do not use the RMIT single-sign-on password or any secure passwords on Mongo Atlas.
2. DO NOT delete or change the .json files created at the project initialization stage.
3. Zip everything in your code base, preserving the internal directory structure within the code base.
4. Create a plain-text file named README containing any specific instructions to the markers (such as the port number to use in the localhost) that they need for testing your server, list of libraries that are required to install prior to starting your application, etc.
5. Submit the zip file and the README file on Canvas.

3.3 Assessment Declaration

When you submit work electronically, you agree to the RMIT assessment declaration.

3.4 Late Submissions & Extensions

A penalty of 10% per day is applied to late submissions up to 5 business days, after which you will receive zero marks.

Short extensions may be granted by the course coordinator up to 1 business day *before* the due date in accordance with RMIT Assessment Adjustment process. However, extensions are not guaranteed and require suitable documentation. The course coordinator may refer requests to Special Considerations.

Special Consideration *may result in an equivalent assessment*, which may take the form of a timed assessment assessing the same knowledge and skills of the assignment and are generally granted on an individual basis. For more information refer to the RMIT Special Consideration process.

4 Marking Guidelines

4.1 Task 1

- 18 marks for a fully-functional homepage that displays the filtered list of properties based on user input.

4.2 Task 2

- 17 marks for a fully-functional bookings page.

The detailed breakdown is provided on the marking Rubric available on Canvas.

5 Academic Integrity and Plagiarism (Standard Warning)

Academic integrity is about the honest presentation of your academic work. It means acknowledging the work of others while developing your own insights, knowledge and ideas. You should take extreme care that you have:

- Acknowledged words, data, diagrams, models, frameworks and/or ideas of others you have quoted (i.e., directly copied), summarised, paraphrased, discussed or mentioned in your assessment through the appropriate referencing methods
- Provided a reference list of the publication details so your reader can locate the source if necessary. This includes material taken from Internet sites. If you do not acknowledge the sources of your material, you may be accused of plagiarism because you have passed off the work and ideas of another person without appropriate referencing, as if they were your own.

RMIT University treats plagiarism as a very serious offence constituting misconduct. Plagiarism covers a variety of inappropriate behaviours, including:

- Failure to properly document a source
- Copyright material from the internet or databases
- Collusion between students

For further information on our policies and procedures, please refer to the RMIT Academic Integrity Website.

The penalty for plagiarised assignments includes zero marks for that assignment, or failure for this course. Please keep in mind that RMIT University uses plagiarism detection software.