Course : (WDDM120) -Web Programming Application	Assignment 3-4	Contribution:25%(each)- 50% of course
Instructor : Kadeem Best	Date Given: March 7th 2020	Date Due April 12th 2020

Notes for the Student: This Project corresponds to Assignment 3 and 4 and is a continuation of the work you have already completed in Assignments 1 and 2.

Background: You will need to have access to an IDE or text editor and have a thorough understanding of Node.js, Express,Git ,(GitHub or Bitbucket) ,Heroku, MongoDB and Mongoose.

Assignment Regulations

- This project must be done individually.
- **An in-person demonstration of this project is required. Failure to demonstrate your project would result in 0 marks.**
- You must create a private Git repository and add me as a collaborator.
- **Because this project is due on the last day of class, no late submissions will be accepted.**
- Please review Humber's Policies on Academic Integrity, specifically:

  "Academic honesty is central to the learning environment enjoyed by all members of The Humber College Institute of Technology and Advanced Learning (hereafter referred to as"Humber `` or"the College") and an expectation of all Humber applicants, students, faculty and staff. A breach of academic honesty is considered to be an offense against the academic integrity of the learning environment. This policy is intended to ensure adherence to Humber's standard of honesty and scholarly integrity in all activities relating to teaching, supervision, research, administrative and consultative work of faculty and staff and to protect the academic integrity of the institution.." ***Thus, ensure that your code or any part of it is not duplicated by another student(s).***

Technical Requirements

- All back-end functionality MUST be done using Node JS and Express.
- Your views MUST be created with Express-Handlebars
- You can use a CSS Framework such as Bootstrap, Bulma or Materialize CSS to make your website responsive and aesthetically pleasing. Or, you can also continue to use Vanilla CSS to build your website. The option is yours.
- You can use any animation library (JavaScript or CSS) that you want.This is not mandatory, it is up to you.
- You are **NOT** allowed to use any Front-End JavaScript Frameworks, specifically, **No React, No Vue, No Angular.**
- You must use MongoDB as your database engine

Detailed App Specification

This assignment is a continuation of Assignment 1 and 2, thus all the requirements for this assignment is to be made “on top” of your Assignment 1 and 2.

Assignment 3 (25%)

Application Architecture

1. Your application **MUST** be structured **FULLY** according to the MVC Design pattern. Thus, your views, models and controllers must be separated as per the design pattern requirements.

2. **All** sensitive credential information **must** be stored in **environment variables**. Examples include: your sendgrid access token, MongoDB connection string, etc. Implement environment variables locally using the dotenv 3rd party package.

User Registration Module

You are required to implement database functionality for your registration page that was previously implemented in Assignment 1 and 2 . Thus, when a user fills out the registration form and then hits the submit button, provided that all the validation criteria were not violated, your website must then create a user account in your database.

Once the user account is created, your web application must then redirect the user to a dashboard page.

Regarding your database functionality, the following rules must be followed :

1. Setup and configure a MongoDB cloud service using MongDB Atlas https://www.mongodb.com/cloud/atlas.
2. Connect your web application to your mongoDB database using an ODM called **Mongoose.**
3. Name your database and collections appropriately.
4. Ensure that the email field in your registration form **is unique**, thus your application must prohibit different users from having the same email in the database.
5. Passwords **must not** be stored in plain text in the database, thus your application must store passwords in an encrypted format. You can use a 3rd party package called **bcryptjs** to do the aforementioned.

Authentication Module

You are required to implement a fully functional authentication module with the following features:

- Your application must allow an Administrator and regular users, i.e, customers who want to book room, to **log-in via the login form created in Assignment 1 and 2.**
- Upon a successful authentication(entering an email and password pair that exists in the database) **a session must be created to maintain the user state until they have logged out of the application.**
- **To implement sessions** in an Express app you can use https://github.com/expressjs/session
- Upon an unsuccessful authentication, the application **must** display an appropriate message (Example: **Sorry, you entered the wrong email and/or password**)
- Also after successfully authenticating,the application must determine if the person logging in is an Administrator or a regular user and will be redirected to their respective dashboard.
- A regular user will be directed to a user dashboard and an Administrator will be directed to an Administrator dashboard.
- Both dashboards, must show the user's name(first name and lastname) and a logout link
- The logout link must destroy the session created when the user initially authenticated.
- Specific routes can only be accessed when users are logged-in, thus those routes must be protected.

Administrator Module

You are required to implement an Administrator module that allow an Administrator to do the following :

1. **Create room:** The Administrator must be able to add rooms to the database. See the following data that must be added when a room is created:
    a. Title
    b. Price per night,
    c. Room description or details,
    d. Room location
    e. Set room as a Featured room (or not).
    f. Upload a photo of the room(to keep the project simple, only upload one image per room).

2. Ensure that all created rooms that were entered into the database are populated on the front-end of the web application, specifically on the room listing page that was created in Assignment 1 and 2. Additionally, only rooms that were set as “Featured Room” should be populated in the “Featured Room” section of the home page. **Please note, a visitor to the web application does not need to be logged in to view the rooms that were created by the Administrator.**

3. Ensure that a user can only upload an image, i.e jpgs,gifs,pngs, for the room photo.

4. View a list of all created rooms.

5. Edit and change room details for a selected room. Example, title, description, price, etc

Please note, an Administrator must be logged-in to the web application to be able to do the aforementioned (create rooms, view all created rooms, edit and change room details). To facilitate the logging in of the Administrator, create a regular user in the Front-End and then long into your MongoDB Atlas account and manually change a field in the document to make allow the application to able to make the distinction between regular user vs Administrator - **I will further expound on this in class.**

Search Module

Visitors to the web application should be able to search for rooms via a location. This can be easily implemented by having a drop down list of your room locations and a search button.

Place the drop down list to the top of the room listing page. When a user visits that page, they should be presented with all rooms stored in the database. However, when a user selects a specific location category from the drop down list and clicks the search button, the application must then present a list of all the rooms associated with the selected location. category. Please note, a user does not have to be logged in to search for rooms.

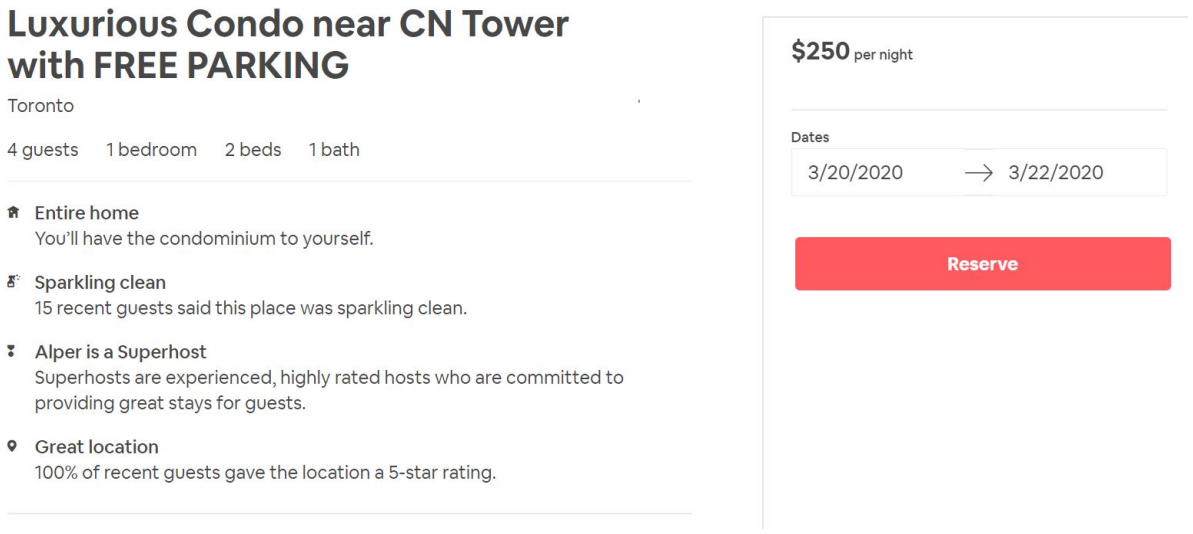
Assignment 4 (25%)

Room Description Page

Only logged in users should be able to “book” rooms by adding selected rooms to their room booking cart.

From the room listing page, when a user clicks on a particular room, they should be navigated to the **Room Description page** of the clicked room and from that page they can add the room to their room booking cart.

Sample of a Room Description Page Example :



The Room Description page of any room should list the following:

A. Room Image
B. Room title
C. Room description
D. Room price(per night)
E. Dates-Number of Nights (start date - end date)
F. Reserve Button

When the user clicks the “**Reserve**” button, the given room will be added to the user's room booking Cart.

Please note that you must prevent two different users from booking a room during the same time period. For example, if person X books room A from March 7th to March 11th. Person Y cannot book a room during any of those days.

Room Booking Cart Module

The user's dashboard should have a link to the logged-in user room shopping cart . This page must display all the rooms that were added to the room booking cart, how many nights were booked for each added room and the total amount for each booked room. This is calculated by **the cost of room per night X number of nights** . Also, the page should have a "Place Booking order" button.

When the **"Place Booking order**" button is pressed, your web application should "clean" your room booking cart, send an email to the logged in user's email, indicating all the rooms that were booked, the number nights booked for each room as well as their order total.

Assignment 3 Rubric

Criteria	**Not Implemented 0**	**Partially Implemented 1**	**Fully Implemented 2**
User Registration ● MongoDB cloud service is setup Database and collection have appropriate names ● User's data is inserted into the database when the user fills out then form and hits the submit button. ● User is redirected to a dashboard page when form is submitted ● Email Is unique ● Password is stored in encrypted format (12)			
Authentication			

• **A Session is created to maintain the user state until the user logs out of the application.** • Upon an unsuccessful authentication, the application displays an appropriate error message • The application directs an Administrator to their dashboard and a regular user to his or her dashboard. • Both dashboards, show the user's name(first name and lastname) and a logout link • The logout link destroys the session. • Routes that can only be accessed when users are logged-in, must be protected.			

Criteria	**Not Implemented 0**	**Partially Implemented 1**	**Fully Implemented 2**
Administrator Module • An Administrator account is manually created in the database. • An Administrator can create rooms with all the necessary data • An Administrator can upload a photo for a room • Only a Images can be uploaded as a photo. • All rooms created are populated on the Front-End of the website, specifically the room listing Page • Rooms that are set as featured			

rooms are rendered in the appropriate section on the home page. ● The Administrator can view a list of all created rooms. ● Administrator can edit room details **Search Module** ● Search form has a drop down list of location categories and a search button. ● Visitors to the web application can search for rooms via a location category.			

Total : 44 MARKS

Assignment 4 Rubric

Criteria	**Not Implemented 0**	**Partially Implemented 3**	**Fully Implemented 5**
Booking Module ● Only logged in users can add rooms to their room booking cart. ● User's must be able to view all their rooms in their room booking cart and the room booking cart must display all rooms added, the number of nights booked for each room and the total amount to be paid ● When user clicks "Place Booking Order", the application must "clear" the room booking cart and send email with the entire order information			

Booking Room ● Multiple people cannot book the same room during the same time period	**0**	**3**	**6**

Look and Feel ● Overall site looks polished on all devices.	**0**	**5**	**10**

Total : 31 MARKS

THE END