Ismail Kefokeris

Assignment Description

I am tasked with developing a website named PlacesToStay, the site should allow users to lookup information on places they may want to stay whilst away, such as hotels, bed and breakfasts, and hostels. The website should make use of Node and Express as the back end, and MySQL for the database.

COM518 - AE1

Contents

[**Task 1** 2](#_Toc101702470)

[**Task 2** 2](#_Toc101702471)

[**Task 3** 2](#_Toc101702472)

[**Task 4** 2](#_Toc101702473)

[**Task 5** 2](#_Toc101702474)

[**Task 6** 2](#_Toc101702475)

[**Task 7** 2](#_Toc101702476)

[**Task 8** 2](#_Toc101702477)

[**Task 9** 2](#_Toc101702478)

[**Task 10** 3](#_Toc101702479)

[**Task 11** 3](#_Toc101702480)

[**Task 12** 3](#_Toc101702481)

# **Task 1**

**Complete: Yes**

For task 1 I needed to implement an Express route to find all accommodations in a given location. To create this route for accommodations I must first create a new file to store all routes that will traverse through the route “/acc/”



This can then be connected to “app.js” after being exported.



A screenshot of a computer

Description automatically generated with medium confidence

The route I have gone for is “/acc/search/:location” (location being the name of the location). This will be an Express GET route as its retrieving data from the server.



I needed to connect my GET route with the Accommodation Controller (accController) which handles the logic and communicating with the DAO (database).

Text

Description automatically generated

In the accommodation controller we have an async function defined called “findAccByLocation” which is used in our route, this function will run when the route is called. The function begins with a try statement which is there handle any errors that may occur with the database. Inside the try block we start by querying our database using the controllers DAO which is defined on creation and passed through the location.

Text

Description automatically generated

The DAO is asked to call a function “findAccByLocation” which will query the database.

A screenshot of a computer

Description automatically generated with medium confidence

Above are the insides of the function and as can be seen it returns a new Promise, this is so we won’t have to worry about call-backs and allows us to write sequential code with async/await. My function for this promise takes in the two parameters resolve, and reject before finally querying the database looking for all accommodation options in a given location, if there were an error the function reject will run rejecting the promise, otherwise it will check whether the query has returned any results and depending on the answer to that it would either send back nothing to the controller or the information it has.

Text

Description automatically generated

Now moving back to the controller. If nothing was sent back from the DAO, the controller would pick this up and return both a user-friendly message back to the user and a HTTP status code of 404. Otherwise, the API will return the results.

# **Task 2**

**Complete: Yes**

For task 2, I must find all accommodation of a specific type in a specific location, as this task is still focused on the searching of accommodation I will stay within the same routes file used for task 1 “accRoutes”, “/acc”





Similarly to task 1 I created a get route in accRouter, I defined the route “search/:type/:location” (type and location would be changed out for a type and location) and connected it again to the accommodation controller which in this case is calling the function “findAccByLocationType”.

Text

Description automatically generated

This function goes through pretty much the same steps as task 1 “findAccByLocation” function, the difference is this async function uses the dao to call “findAccByLocationType” with the variables location and type which will be used when querying the database.

Text

Description automatically generated

# **Task 3**

**Complete: Yes**

Task 3 requires me to instead of retrieving data from the database I must input data into the database. I need to create a simple system to book a place at any given accommodation. The API must be given the accommodation ID, Number of people, Date, and username. The username was not part of the question but when looking through the database there was a column for username, and it seemed best to add it now then later.



To start of I decided to place it in the “accRoutes” routes because it relates to accommodations and seemed appropriate.



# **Task 4**

**Complete: Yes**

# **Task 5**

**Complete: Yes**

# **Task 6**

**Complete: Yes**

# **Task 7**

**Complete: Yes**

# **Task 8**

**Complete: Yes**

# **Task 9**

**Complete: Yes**

# **Task 10**

**Complete: No**

# **Task 11**

**Complete: No**

# **Task 12**

**Complete: No**