



**L** OVELY  
**P** ROFESSIONAL  
**U** NIVERSITY

---

*Transforming Education Transforming India*

## **INT222: ADVANCED WEB DEVELOPMENT**

**WETRAVEL**

**NAME:** Shivam Kashyap

**REG NO :** 12000809

**SECTION:** KM062B64

**SUBMITTED TO:** Dr. Allam Mohan

## Introduction

This We Travel Express online application was designed with the Indian culture's joint family arrangement in mind.

India has a vast cultural heritage. The culture of India refers to a group of small unique civilizations. Indian culture includes clothing, festivals, dialects, religions, music, dance, architecture, cuisine, and art.

You will need a website to plan your trip location to rest, meditate, and spend time with your loved ones and friends in order to observe this.

India is without a doubt a symbol of world togetherness.

This Ladakh alpine community is rapidly ascending, with the towering Himalayas offering a breath-taking backdrop and plenty of possibilities for adrenaline junkies to get their fill.

The Anaimalai or Anamala Hills, also known as the Elephant Mountains, are a group of mountains that form the southern portion of the Western Ghats. Kangchenjunga Mountain, LUSHAI Hills, and SHIVALIK Hills can be seen on the website's mountain page.

No trip to India would be complete without a stop to this beach paradise.

This former Portuguese colony has something for everyone: temples and cathedrals for the cultural vulture, bars and clubs for the party animal, and restaurants that provide almost every cuisine under the sun on the website's beaches page.

India offers a wide range of adventures and unique locales. It is a colorful world with a wide range of cultural attractions, including majestic structures, historic temples, and tombs.

The ancient cultural heritage of the country is inextricably linked to its current technological existence. Varanasi, one of India's most important religious centers, has to be seen to be believed.

The place is bustling with devotees, sadhus, tourists, and locals, and it is brimming with sights that are uniquely Indian.

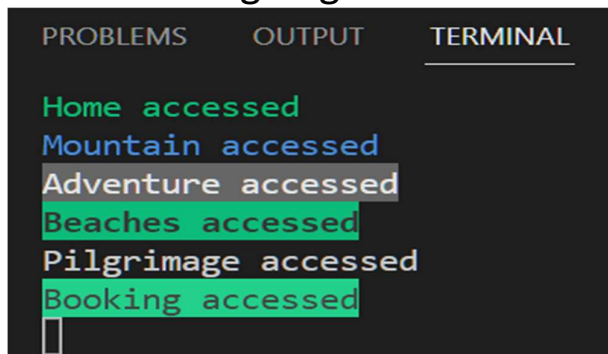
The temples and ghats (steps leading to the Ganga) are a sight to behold, with thousands of devotees performing religious rituals and Hindu priests performing rites. You can see them on the website's pilgrimage page.

## TECHNOLOGIES USED:

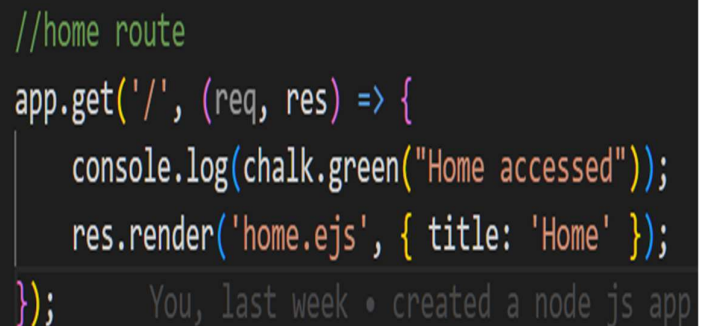
1. HTML
2. CSS
3. Express
4. Java Script
5. Bootstrap

## DEPENDENCIES USED:

1. Express
  - a. It is a minimal and flexible Node.js web application framework that provides a robust set of features for web and mobile applications.
2. Nodemon
  - a. It is a program that assists in the development of Node.js-based applications by automatically restarting the node application when directory file changes are detected.
3. Chalk
  - a. It is used to decorate the output on the console. With the help of this, we can see which task is going in at the console level.



```
PROBLEMS  OUTPUT  TERMINAL
Home accessed
Mountain accessed
Adventure accessed
Beaches accessed
Pilgrimage accessed
Booking accessed
█
```



```
//home route
app.get('/', (req, res) => {
  console.log(chalk.green("Home accessed"));
  res.render('home.ejs', { title: 'Home' });
});
```

#### 4. express-validator

- a. It helps to validate the data in multiple manners so that no ambiguous data is present in the form.

```
53 //posting data from booking page and getting in json format
54 //for validating we write in [] below
55 app.post('/booking', uncodparser, [
56   check('no_travelar', "Cant be empty")
57     .exists()
58     .isLength({ min: 1 }),
59   check('no_senior_citizen', "Cant be empty")
60     .exists()
61     .isLength({ min: 1 }),
62   check('trip_days', "Cant be empty")
63     .exists()
64     .isLength({ min: 1 }),
```

#### 5. Path

- a. It is a middleware that helps to locate the static file that is used in the application while developing it.

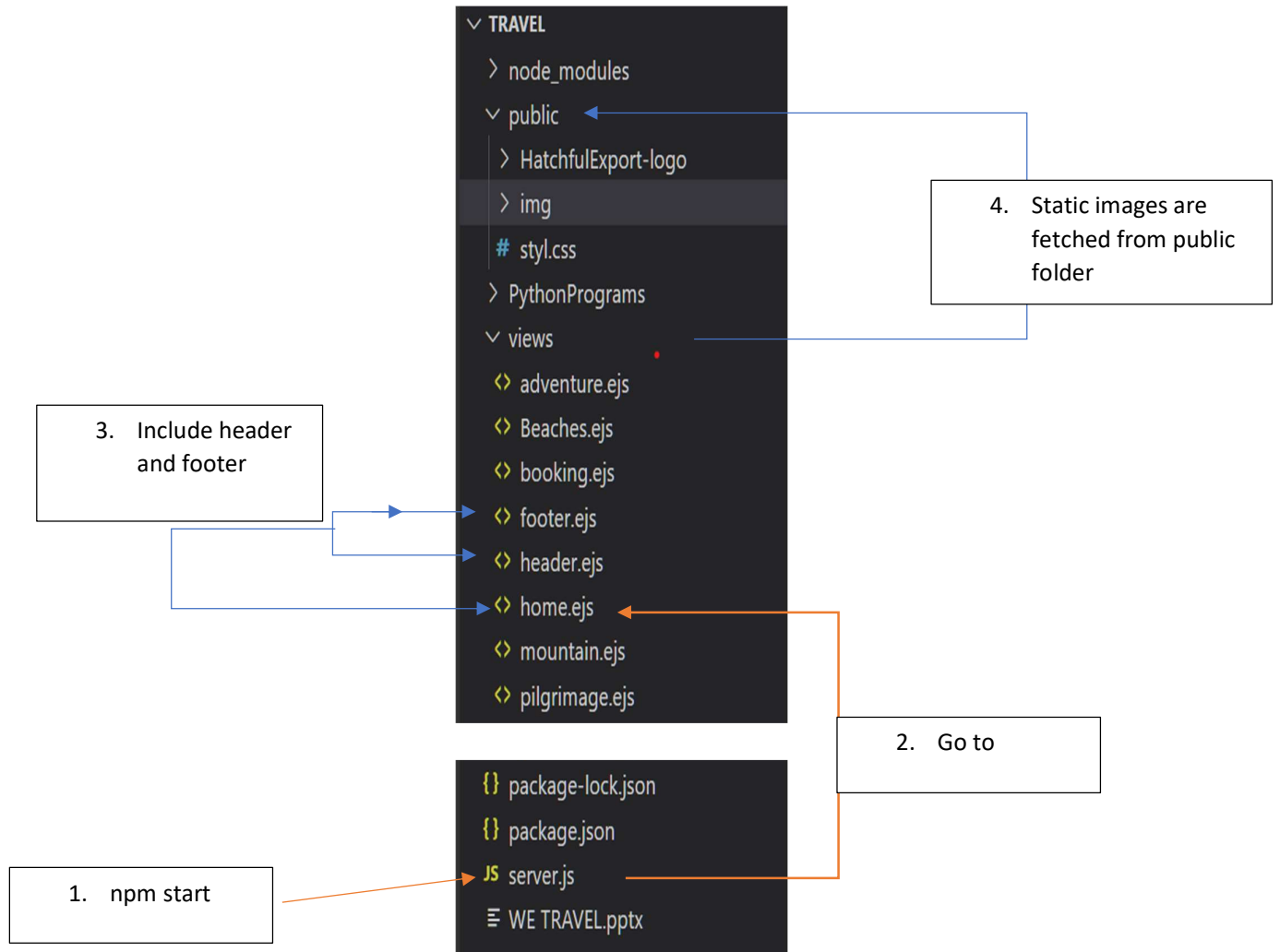
```
// using this middleware to serve static files
app.use(express.static('public'));
```

```
✓ TRAVEL
  > node_modules
  ✓ public
    > HatchfulExport-logo
    > img
    # styl.css
```

#### 6. body-parser

- a. It parses the data that is present on the body part of the application while posting the data due to which it is easy to store the data into a JSON file.

## FLOW OF OPERATION DONE BY WEB APPLICATION:



## WORKING ON WEB APPLICATION:

### 1. Starting the server with the help of “npm start”

```
PS E:\B.TECH\Semester5th\advanced_web_development\travel> npm start

> travel@1.0.0 start
> nodemon server.js

[nodemon] 2.0.20
[nodemon] to restart at any time, enter `rs`
[nodemon] watching path(s): *.*
[nodemon] watching extensions: js,mjs,json
[nodemon] starting `node server.js`
Server is running on port 2022
```

### 2. This will read the server.js

### 3. From the requested URL it will find the app.get method having the home (localhost:2022) URL.

```
//home route
app.get('/', (req, res) => {
  console.log(chalk.green("Home accessed"));
  res.render('home.ejs', { title: 'Home' });
});
```

### 4. The file home. ejs will be fetched from the views folder. After this, there will be “Home accessed” output will appear in the console.

```
Server is running on port 2022
Home accessed
```

### 5. From this page you can access any part of India concerning MOUNTAINS. ADVENTURE, BEACHES, PILGRIMAGE, and also book your visit to any place.

6. On hovering over the date section it will change to the current time with the help of the onmouseover function and replacewith method in js.

```
<button class="but2" onmouseover="mouseovera()"; style="color: white">
  <h3 id="SECOND">
    <script>
      const myDate = new Date();
      document.write(myDate.getDate() + "/" + (myDate.getMonth() + 1)
        + "/" + myDate.getFullYear());
    </script>
  </h3>
</button>
<script>
  function mouseovera() {
    document.getElementById("SECOND").replaceWith(myDate.getHours()
      + ":" + myDate.getMinutes());
  }
</script>
```

7. When you click on the Mountain button, it will fetch the href link.

```
<ul class="ul">
  <li id="li"><a href="/">Home</a></li>
  <li id="li"><a href="/mountain">Mountain</a></li>
  <li id="li"><a href="/adventure">Adventure</a></li>
  <li id="li"><a href="/Beaches">Beaches</a></li>
  <li id="li"><a href="/pilgrimage">Pilgrimage</a></li>
</ul>
```

- a. From here it will go to server.js and trigger the `"/mountain"` method.
- b. This will fetch the mountain .ejs from the views folder



8. Now coming to the “Booking” part of our application.
- a. Here I used HTML form to take input from the user

Details

No. Of Traveler

No. Of Senior Citizen

Trip Days

Email

Phone

Area Code

Phone Number

Destination

--Choose Destination--

Are you opt for travel insurance ?

☒ Yes ☐ No

Register

- b. On clicking the register button the data filled will be “posted” to the server.js “/booking”

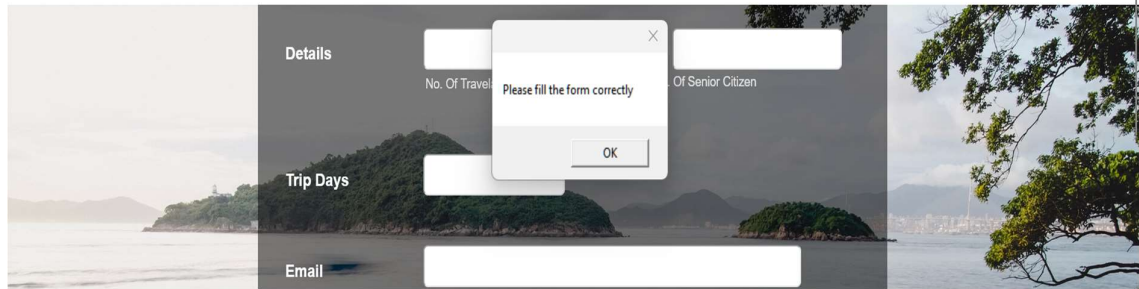
```
201 <div class="main">
202
203   <form method="POST" action="/booking">
204     <div id="name">
205       <h2 class="name">
206         Details </h2>
```

- c. It will validate the data, if the data is ok then it will redirect to the home page and reflect successful at the place of the title

```
//posting data from booking page and getting in json format
//for validating we write in [] below
app.post('/booking', uncodemparser, [
  check('no_travelar', "Cant be empty")
    .exists(),
  check('no_senior_citizen')
    .exists(),
  check('trip_days', "Cant be empty")
    .exists(),
  check('email', "Email is not valid")
    .isEmail()
    .normalizeEmail(),
  check('phone', "Phone number is not valid")
    .exists()
],
(req, res) => {
```

9. If there is any error the “/booking” will redirect itself with the “try again” title name and show the error.

```
/* creating a var to store error message and
   coming back to the registration page*/
const errors = validationResult(req)
if (!errors.isEmpty()) {
  console.log(chalk.bgRed(JSON.stringify(errors)));
  var alert = require('alert');
  alert(`Please fill the form correctly`);
  res.render('booking.ejs', { title: 'TRY AGAIN' });
}
```



The screenshot shows a travel form titled 'Details' with fields for 'No. Of Travel', 'Of Senior Citizen', 'Trip Days', and 'Email'. A modal dialog box is displayed in the center with the message 'Please fill the form correctly' and an 'OK' button. The background of the form features a scenic view of a beach and mountains.

10. The successful output will create a text file with the help of “fs.writeFileSync and JSON.stringify();”

```

76     else {
77         //storing the data that came from the body form
78         const data = (req.body);
79         // setting phone number as the file name
80         const filename = data.email;
81         // converting the data to string format and storing it in file
82         fs.writeFileSync(filename + ".txt", JSON.stringify(data));
83         console.log(chalk.bgYellowBright("Data stored"));
84         res.render('home.ejs', { title: 'Sucessfull' });
85     }
86 }

```

11. The file will be created from the “email” input that the user provided.

```

{} package-lock.json
{} package.json
JS server.js
shivam@gmail.com.txt U

```

12. In this manner you came and enjoyed the get-together of Indian culture and planned your holidays.

**OUTCOME:**

With the help of this web application project, I was able to apply what I learnt in class about JavaScript foundations to NodeJS.

During this period, I learned new concepts like as get, post, routing, server, and nodemon.

This assignment taught me to delve further into documents and explore my problem in order to solve it and find solutions.