Team 10

Project Documentaton: Travel Bucket List

Author: Hack Heros

Github Link: https://github.com/rhtknt01/TravelBucketList.git

Website Link: https://travel-bucket-list-static-website.s3.ap-south-1.amazonaws.com/index.html

Problem Statement: Many travelers like to maintain a list of dream destinations and experiences they wish to explore in their lifetime—commonly called a "travel bucket list." However, they often lack a simple and visually inspiring platform to view and organize their travel goals in one place.

The goal is to create a static, visually appealing web page that showcases a curated "Travel Bucket List." This site should present a selection of travel destinations.
The website will serve as inspiration and motivation for users to dream about their future travel adventures.

Application Architecture:

The Travel Bucket List application allows users to maintain a list of places they plan to visit or have already visited. Users can add destinations, mark them as visited, and filter them based on their status.

Key Components:

- Frontend: Built with HTML, CSS, and JavaScript using Vite as the build tool. It handles displaying the travel bucket list, adding destinations, marking them as visited, and filtering the list.
- State Management: Utilizes LocalStorage to persist data (bucket list, visited places) even after page reloads.

- Emoji Representation: Each city or place in the list is associated with an emoji, stored in a JavaScript object (cityEmojiMap), to represent the city's unique characteristics.

Features:

- Add a Destination: Users can enter the name of a city, optionally add notes, and save it to their bucket list.
- Mark as Visited: Users can toggle the visited status of a destination.
- Filters: Users can filter the list based on visited or not visited places.
- Delete: Users can remove destinations from their bucket list.
- Clear All / Clear Visited: Options to clear all places or only the visited ones.

Future Scope:

Display of destination images, titles, and short descriptions.

Technologies Used:

• **HTML:** Structure and layout of the app.
• **CSS:** Styling the app for a clean and intuitive design.
• **JavaScript:** Handling the dynamic functionality, such as adding Destinations.

1.Files Structure:

- index.html: Main structure of the application, which includes the form for adding new places, displaying the list, and the filtering controls.
- style.css: Defines the styles and UI elements.
- main.js: Contains the logic for handling form submissions, managing the state (bucket list), and rendering the list dynamically based on user actions.

2. Development Process

Setup:

1. Install Vite globally (if not already installed):
   npm install -g vite

2. Initialize a new project:
   npm init @vitejs/app travel-bucketlist
   cd travel-bucketlist
   npm install

Creating the App:

1. Create index.html for the structure of the application.
2. Add style.css for styling the application.
3. Write the JavaScript logic in main.js for handling actions like adding items, saving data to LocalStorage, and rendering the list.
4. Implement the city-emoji mapping in JavaScript for displaying relevant emojis.

Development Tools:

- Vite: For bundling and serving the app.
- LocalStorage: For persisting the travel bucket list data.
- JavaScript: For dynamic behavior, such as adding items, deleting, toggling the visited status, and saving data to LocalStorage.

Development Workflow:

1. Writing the Code:
   - Develop the HTML, CSS, and JavaScript.
   - Implement core functionality like adding destinations, toggling visited status, and removing items.
   - Use event delegation for handling user interactions efficiently.
   - Ensure compatibility across various browsers.

2. Testing:
   - Manually test each feature (add a place, mark visited, clear visited, delete).
   - Test compatibility with modern browsers (Chrome, Firefox, Safari, Edge).
   - Verify LocalStorage persistence after page reloads.

3. Refining the Application:
- Refactor the code for readability and optimization.
- Fix cross-browser issues.
- Ensure accessibility features (e.g., ARIA roles).

3. Deployment Steps

1. Build the Application:

Run the following command to build the app:
npm run build
This will generate the final static files inside the dist folder.

2. Deploy the Application:

Option 1: Vercel
- Go to Vercel and log in or sign up.
- Create a new project and import the repository.
- Vercel automatically detects the Vite project and deploys it.

Option 2: Netlify
- Go to Netlify and log in or sign up.
- Click on "New Site from Git" and choose your Git provider.
- Select your repository, and Netlify automatically detects Vite to build and deploy the app.

Option 3: GitHub Pages
- Use the vite-plugin-gh-pages to deploy the app to GitHub Pages.

3. Post-Deployment:

- After deploying, share the public URL.
- Verify the functionality by testing adding items, marking visited, and ensuring proper emoji rendering.

4. Source Code Repository

GitHub Repository Setup:

1. Create a New Repository on GitHub:
   - Go to GitHub and log in.
   - Click on the "New" repository button.
   - Provide a name and description for the repository.

2. Push Your Code:
   - Initialize a Git repository in your project folder:
     git init
     git add .
     git commit -m "Initial commit"
     git remote add origin https://github.com/your-username/travel-bucketlist.git
     git push -u origin master

Collaboration:
- Other developers can clone the repository using:
  git clone https://github.com/your-username/travel-bucketlist.git
- Developers can work on features or bug fixes in their own branches and submit pull requests.

Example Repository Structure:

```
/travel-bucketlist
    ├── index.html
    ├── style.css
    ├── main.js
    ├── package.json
    ├── README.md
    ├── dist/ (Build folder after running `npm run build`)
```

1. **HTML Structure (index.html):**

```
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="icon" href="/airplane.png">
<link rel="stylesheet" href="./src/style.css" />
<meta charset="UTF-8" />
<meta
     name="viewport"
     content="width=device-width,initial-scale=1,maximum-
scale=1,user-scalable=no"
   />
<title>Travel Bucket List</title>
</head>
<body>
<main id="app" role="main" aria-label="Travel bucket list
application">
<header>
<h1>Travel Bucket List ✈️</h1>
</header>

<div id="counter" aria-live="polite" aria-
atomic="true">Loading...</div>

<form
       id="add-form"
       aria-label="Add a new travel destination to your bucket list"
>
<input
```

```
        type="text"
        id="new-place"
        placeholder="Add a new place to visit"
        aria-required="true"
        autocomplete="off"
        aria-describedby="placeHelp"
        maxlength="50"
        required
      />
<textarea
        id="new-notes"
        placeholder="Add notes (optional)"
        aria-label="Notes for the visit destination"
        maxlength="250"
        rows="3"
></textarea>
<button type="submit" aria-label="Add place to bucket list">
        Add Place
</button>
</form>

<nav id="filters" role="tablist" aria-label="Filter travel destinations">
<button
        type="button"
        class="active"
        data-filter="all"
        role="tab"
        aria-selected="true"
        tabindex="0"
>
```

```
        All
</button>
<button
        type="button"
        data-filter="notvisited"
        role="tab"
        aria-selected="false"
        tabindex="-1"
>
        Not Visited
</button>
<button
        type="button"
        data-filter="visited"
        role="tab"
        aria-selected="false"
        tabindex="-1"
>
        Visited
</button>
</nav>

<ul
      id="bucket-list"
      aria-live="polite"
      aria-relevant="additions removals"
      aria-label="Travel bucket list"
></ul>

<section id="clear-buttons">
```

```
<button id="clear-visited" aria-label="Clear all visited places">
        Clear Visited
</button>
<button id="clear-all" aria-label="Clear entire bucket list">
        Clear All
</button>
</section>
</main>

<script type="module" src="./src/main.js"></script>
</body>
</html>
```

2. CSS Styling (style.css):

The style.css file contains styles to make the app visually.

```
@import
url('https://fonts.googleapis.com/css2?family=Poppins:wght@400;600;700&displa
y=swap');

 /* Reset and base */
 *, *::before, *::after {
  box-sizing: border-box;
 }

 body {
  margin: 0;
  background: linear-gradient(135deg, #74ebd5 0%, #ACB6E5 100%);
  font-family: 'Poppins', sans-serif;
  display: flex;
  justify-content: center;
  align-items: flex-start;
```

```
  min-height: 100vh;
  padding: 1rem 0;
  color: #2c3e50;
  user-select: none;
}

#app {
  background: #fff;
  width: 100%;
  max-width: 350px;
  min-height: 600px;
  box-shadow: 0 12px 20px rgba(0,0,0,0.16);
  border-radius: 16px;
  display: flex;
  flex-direction: column;
  padding: 24px 28px;
}

header {
  text-align: center;
  margin-bottom: 6px;
  user-select: text;
}

header h1 {
  font-weight: 700;
  font-size: 1.9rem;
  margin: 0;
  color: #223343;
  text-shadow: 0 1px 2px rgba(0,0,0,0.1);
}

#counter {
  font-weight: 600;
  font-size: 1rem;
```

```
  margin-bottom: 18px;
  color: #6c5ce7;
  text-align: center;
  letter-spacing: 0.05em;
  user-select: text;
}

form {
  display: flex;
  flex-direction: column;
  gap: 12px;
}

form input[type="text"] {
  border: 2px solid #6c5ce7;
  border-radius: 12px;
  padding: 12px 16px;
  font-size: 1rem;
  font-weight: 500;
  transition: border-color 0.3s ease;
  outline-offset: 2px;
  outline-color: transparent;
}

form input[type="text"]:focus {
  border-color: #341f97;
  outline-color: #b0a7f7;
}

form textarea {
  border: 2px solid #6c5ce7;
  border-radius: 12px;
  padding: 12px 16px;
  resize: vertical;
  min-height: 60px;
```

```
  font-size: 0.95rem;
  font-weight: 400;
  font-family: 'Poppins', sans-serif;
  transition: border-color 0.3s ease;
  outline-offset: 2px;
  outline-color: transparent;
}

form textarea:focus {
  border-color: #341f97;
  outline-color: #b0a7f7;
}

form button {
  background: #6c5ce7;
  color: white;
  font-weight: 700;
  padding: 14px 0;
  text-transform: uppercase;
  font-size: 1.05rem;
  border-radius: 14px;
  border: none;
  cursor: pointer;
  transition: background-color 0.3s ease;
  box-shadow: 0 6px 15px rgba(108,92,231,0.45);
}

form button:hover,
form button:focus {
  background: #341f97;
  box-shadow: 0 8px 22px rgba(52,31,151,0.65);
  outline: none;
}

#filters {
```

```
  margin: 20px 0 14px;
  display: flex;
  justify-content: center;
  gap: 20px;
}

#filters button {
  background: transparent;
  border: none;
  cursor: pointer;
  font-weight: 700;
  font-size: 1rem;
  padding: 6px 10px 2px;
  border-bottom: 3px solid transparent;
  color: #6c5ce7;
  transition: all 0.25s ease;
  user-select: none;
}

#filters button:hover,
#filters button.active {
  color: #341f97;
  border-color: #341f97;
  outline: none;
}

#filters button:focus-visible {
  outline: 2px solid #341f97;
  outline-offset: 4px;
}

ul#bucket-list {
  list-style: none;
  margin: 0;
  padding: 0;
```

```
  overflow-y: auto;
  flex-grow: 1;
  max-height: 430px;
  scroll-behavior: smooth;
}

ul#bucket-list li {
  background: #f8f9fc;
  border-radius: 14px;
  padding: 14px 18px;
  margin-bottom: 12px;
  box-shadow: 0 3px 9px rgba(108, 92, 231, 0.1);
  display: flex;
  flex-direction: column;
  transition: background-color 0.3s ease;
}
ul#bucket-list li.visited {
  background: #e4e7eb;
  color: #7b8a99;
  text-decoration: line-through;
}

.item-header {
  display: flex;
  align-items: center;
  justify-content: space-between;
  gap: 10px;
}

.item-name {
  font-weight: 600;
  font-size: 1.05rem;
  word-break: break-word;
  flex-grow: 1;
  user-select: text;
```

```
}

.item-notes {
  margin-top: 6px;
  font-size: 0.9rem;
  color: #74828f;
  white-space: pre-wrap;
  line-height: 1.3;
  user-select: text;
}

button.action-btn {
  background: transparent;
  border: none;
  cursor: pointer;
  font-size: 1.3rem;
  color: #6c5ce7;
  padding: 6px 6px 3px;
  border-radius: 8px;
  transition: color 0.2s ease, background-color 0.2s ease;
  flex-shrink: 0;
}
button.action-btn:hover,
button.action-btn:focus {
  color: #341f97;
  background: #e0dbf9;
  outline: none;
}

button.action-btn.visit {
  color: #00b894;
}
button.action-btn.visit.visited {
  color: #636e72;
}
```

```
#clear-buttons {
  margin-top: 8px;
  display: flex;
  justify-content: space-between;
  gap: 16px;
}

#clear-buttons button {
  flex: 1;
  background: #d63031;
  color: white;
  font-weight: 700;
  border: none;
  border-radius: 14px;
  padding: 12px 0;
  font-size: 0.95rem;
  cursor: pointer;
  box-shadow: 0 5px 15px rgba(214,48,49,0.5);
  transition: background-color 0.3s ease;
}
#clear-buttons button:hover,
#clear-buttons button:focus {
  background: #b71c1c;
  box-shadow: 0 7px 20px rgba(183,28,28,0.7);
  outline: none;
}

/* Scrollbar styling */
ul#bucket-list::-webkit-scrollbar {
  width: 8px;
}
ul#bucket-list::-webkit-scrollbar-thumb {
  background-color: #6c5ce7;
  border-radius: 6px;
```

```
}

/* Mobile tweaks */
@media (max-width: 360px) {
  #app {
    padding: 20px 22px;
    min-height: 580px;
  }

  form button,
  #clear-buttons button {
    font-size: 0.9rem;
    padding: 10px 0;
    border-radius: 12px;
  }

  ul#bucket-list li {
    padding: 12px 14px;
    margin-bottom: 10px;
  }

  .item-name {
    font-size: 1rem;
  }

  .item-notes {
    font-size: 0.85rem;
  }
}
```

3. **JavaScript Functionality (main.js):**

The main.js file contains the JavaScript that handles the core functionality of the **Travel Bucket List**. It includes functions to add Locations.

```
const cityEmojiMap = {
  delhi: "🕌",
  mumbai: "🌆",
  newyork: "🗽",
  losangeles: "🎬",
  london: "🎡",
  paris: "🗼",
  tokyo: "🗾",
  sydney: "🌉",
  dubai: "🏙️",
  america: "US",
  india: "IN",
  japan: "JP",
  france: "FR",
  uk: "GB",
  australia: "AU",
  canada: "CA",
  beijing: "🏯",
  shanghai: "🏙️",
  singapore: "🦁",
  seoul: "🏙️",
  bangkok: "🏯",
  kualalumpur: "🏙️",
  rome: "🏛️",
  berlin: "🏰",
  madrid: "🏰",
  barcelona: "🏰",
  vienna: "🎻",
  amsterdam: "🚲",
```

```
  athens: " 🏛 ",
  moscow: " 🏰 ",
  chicago: " 🌆 ",
  sanfrancisco: " 🌉 ",
  toronto: " 🌆 ",
  vancouver: " ⛰ ",
  mexicocity: " 🌆 ",
  lasvegas: " 🎰 ",
  buenosaires: " 🌆 ",
  riodejaneiro: " 🏖 ",
  saopaulo: " 🌆 ",
  lima: " 🌆 ",
  bogota: " 🌆 ",
  cairo: " 🏜 ",
  capetown: " 🏖 ",
  johannesburg: " 🌆 ",
  lagos: " 🌆 ",
  nairobi: " 🌆 ",
  melbourne: " 🎡 ",
  auckland: " 🌆 ",
  jakarta: " 🌆 ",
  capeTown: " 🏖 ",
  hoChiMinhCity: " 🌆 ",
  dubai: " 🌆 "
};


// Function to fetch emoji based on place name
function getEmojiForPlace(name) {
  const lowerName = name.toLowerCase();
```

```
  // Try exact match first
  if (cityEmojiMap[lowerName]) {
    return cityEmojiMap[lowerName];
  }

  // Fallback: Try matching part of the name
  for (const key in cityEmojiMap) {
    if (lowerName.includes(key)) {
      return cityEmojiMap[key];
    }
  }

  return " 🌍 "; // Default fallback
}

// DOM Elements
const form = document.getElementById('add-form');
const inputPlace = document.getElementById('new-place');
const inputNotes = document.getElementById('new-notes');
const list = document.getElementById('bucket-list');
const counter = document.getElementById('counter');
const filters = document.getElementById('filters');
const clearVisitedBtn = document.getElementById('clear-visited');
const clearAllBtn = document.getElementById('clear-all');

// State
let bucketList = JSON.parse(localStorage.getItem('bucketList')) || [];
let currentFilter = 'all';

// Helpers
function saveList() {
  localStorage.setItem('bucketList', JSON.stringify(bucketList));
}
```

```
function updateCounter() {
  const total = bucketList.length;
  const visited = bucketList.filter(i => i.visited).length;
  counter.textContent = total
    ? `Visited ${visited} / ${total} places`
    : 'Your bucket list is empty.';
}

function createButton(className, title, html, tabIndex, onClick) {
  const btn = document.createElement('button');
  btn.className = className;
  btn.title = title;
  btn.innerHTML = html;
  btn.tabIndex = tabIndex;
  btn.addEventListener('click', onClick);
  return btn;
}

// Render Logic
function renderList() {
  list.innerHTML = '';
  const filteredList = bucketList.filter(item => {
    if (currentFilter === 'visited') return item.visited;
    if (currentFilter === 'notvisited') return !item.visited;
    return true;
  });

  if (!filteredList.length) {
    const emptyMsg = document.createElement('li');
    Object.assign(emptyMsg.style, {
      fontStyle: 'italic',
      color: '#74828f',
      textAlign: 'center',
      userSelect: 'text'
    });
```

```
  emptyMsg.textContent = 'No places to show.';
  list.appendChild(emptyMsg);
  updateCounter();
  return;
}

const fragment = document.createDocumentFragment();

filteredList.forEach(item => {
  const li = document.createElement('li');
  li.className = item.visited ? 'visited' : '';
  li.tabIndex = 0;

  const header = document.createElement('div');
  header.className = 'item-header';

  const nameSpan = document.createElement('span');
  nameSpan.className = 'item-name';
  // Add the emoji in front of the place name
  nameSpan.textContent = `${getEmojiForPlace(item.name)} ${item.name}`;

  const visitBtn = createButton(
    'action-btn visit' + (item.visited ? ' visited' : ''),
    item.visited ? 'Mark as not visited' : 'Mark as visited',
    item.visited ? '✔' : '⭕',
    0,
    e => {
      e.stopPropagation();
      item.visited = !item.visited;
      saveList();
      renderList();
    }
  );
  visitBtn.setAttribute('aria-pressed', item.visited);
```

```
    const deleteBtn = createButton(
      'action-btn delete',
      'Remove from list',
      '🗑️',
      0,
      e => {
        e.stopPropagation();
        bucketList = bucketList.filter(p => p !== item);
        saveList();
        renderList();
      }
    );

    header.append(nameSpan, visitBtn, deleteBtn);
    li.appendChild(header);

    if (item.notes?.trim()) {
      const notes = document.createElement('div');
      notes.className = 'item-notes';
      notes.textContent = item.notes.trim();
      li.appendChild(notes);
    }

    fragment.appendChild(li);
  });

  list.appendChild(fragment);
  updateCounter();
}

// Event Listeners
form.addEventListener('submit', e => {
  e.preventDefault();
```

```
  const place = inputPlace.value.trim();
  const notes = inputNotes.value.trim();
  if (!place) return;

  const exists = bucketList.some(p => p.name.toLowerCase() ===
place.toLowerCase());
  if (exists) {
    alert('This place is already on your bucket list!');
    inputPlace.focus();
    return;
  }

  bucketList.push({ name: place, notes, visited: false });
  saveList();
  renderList();

  inputPlace.value = '';
  inputNotes.value = '';
  inputPlace.focus();
});

filters.querySelectorAll('button').forEach(button => {
  button.addEventListener('click', e => {
    e.preventDefault();

    filters.querySelectorAll('button').forEach(btn => {
      btn.classList.remove('active');
      btn.setAttribute('aria-selected', 'false');
      btn.tabIndex = -1;
    });

    button.classList.add('active');
    button.setAttribute('aria-selected', 'true');
    button.tabIndex = 0;
    currentFilter = button.dataset.filter;
```

```
    renderList();
  });
});

clearVisitedBtn.addEventListener('click', () => {
  if (!bucketList.some(p => p.visited)) {
    alert('No visited places to clear!');
    return;
  }

  if (confirm('Are you sure you want to clear all visited places?')) {
    bucketList = bucketList.filter(p => !p.visited);
    saveList();
    renderList();
  }
});

clearAllBtn.addEventListener('click', () => {
  if (!bucketList.length) {
    alert('Your bucket list is already empty!');
    return;
  }

  if (confirm('Are you sure you want to clear your entire bucket list?')) {
    bucketList = [];
    saveList();
    renderList();
  }
});

// Initial call
renderList();
```

Output of Project:

Travel Bucket List

Visited 1 / 2 places

chennai

Gateway to south indias culture

ADD PLACE

All Not Visited Visited

~~Mumbai~~

~~City of dreams, vibrant nightlife.~~

Clear Visited Clear All

Visited 1 / 3 places

Add a new place to visit

Add notes (optional)

ADD PLACE

All Not Visited Visited

🕌 Delhi

Vibrant culture, rich history, landmarks

~~🌆 Mumbai~~

~~City of dreams, vibrant nightlife.~~

🌎 chennai

Gateway to south indias culture

Clear Visited Clear All

Travel Bucket List

Visited 1 / 3 places

Add a new place to visit

Add notes (optional)

ADD PLACE

All Not Visited Visited

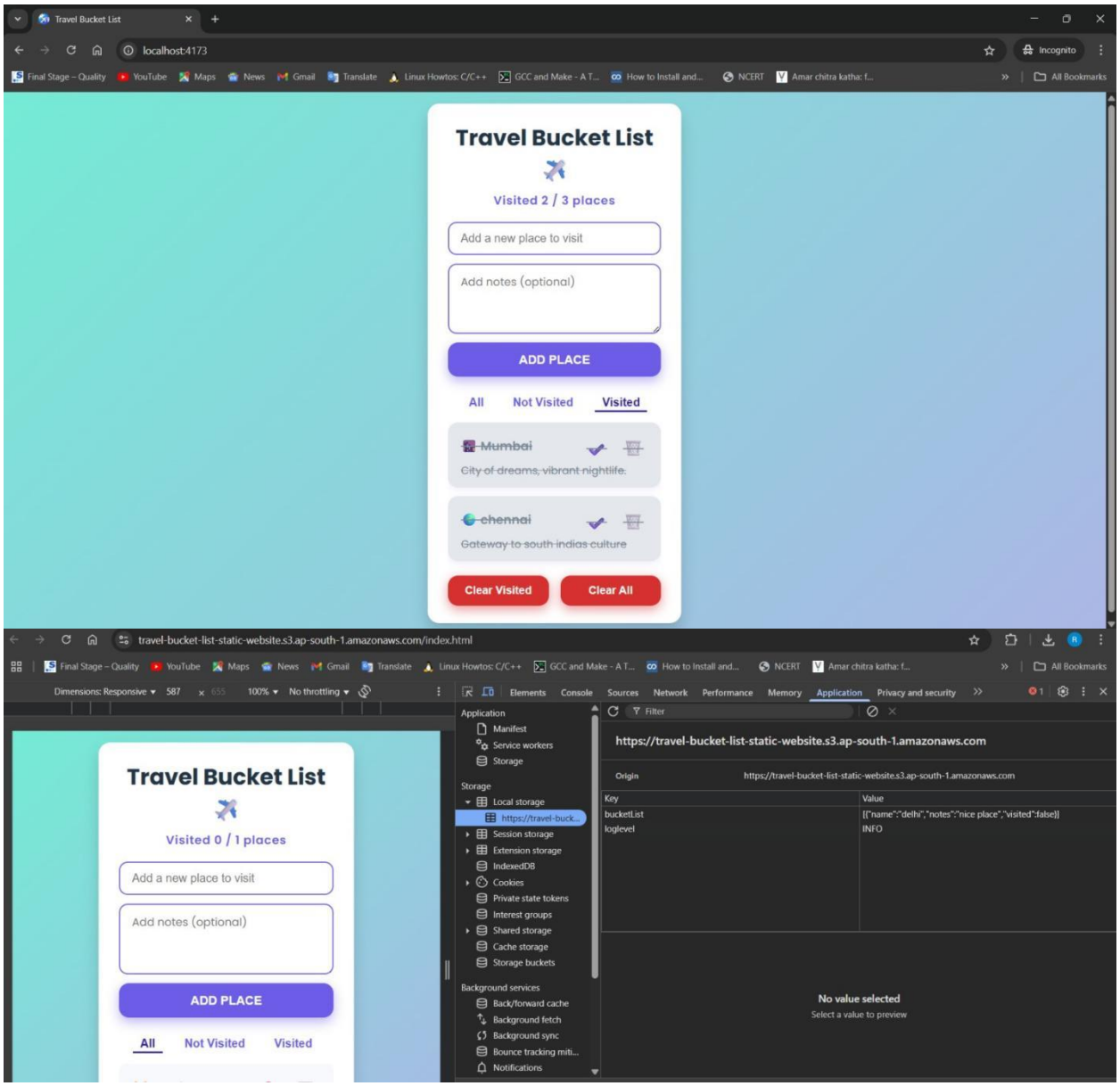
Delhi

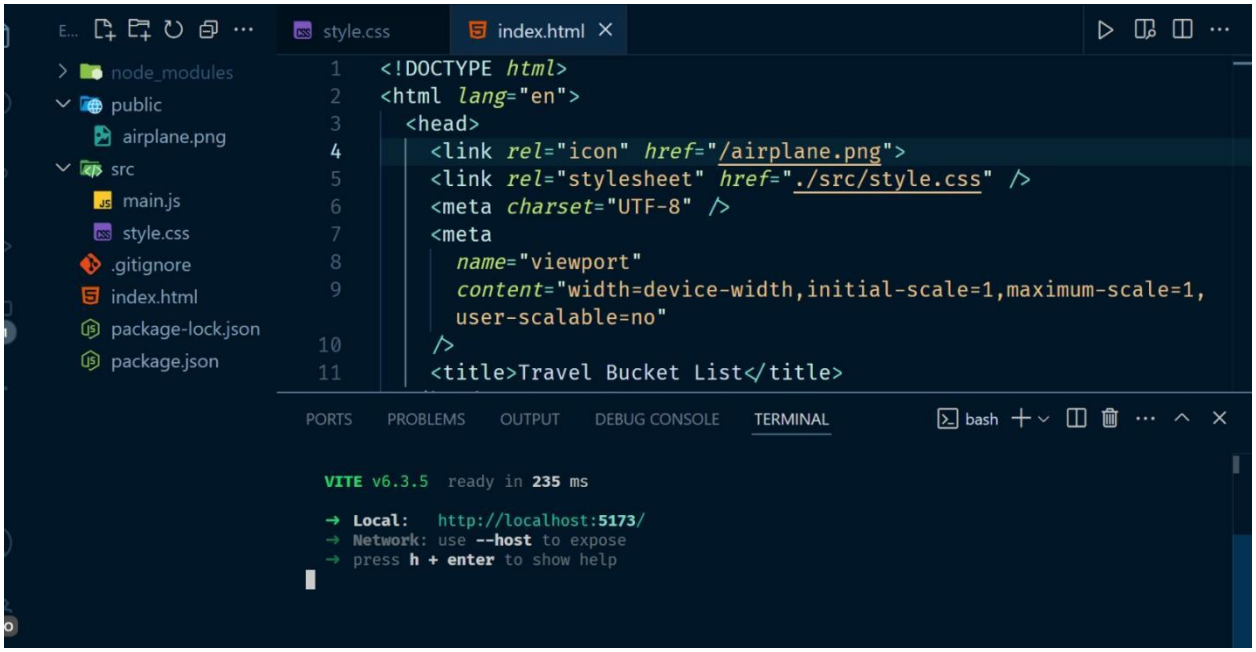
Vibrant culture, rich history, landmarks

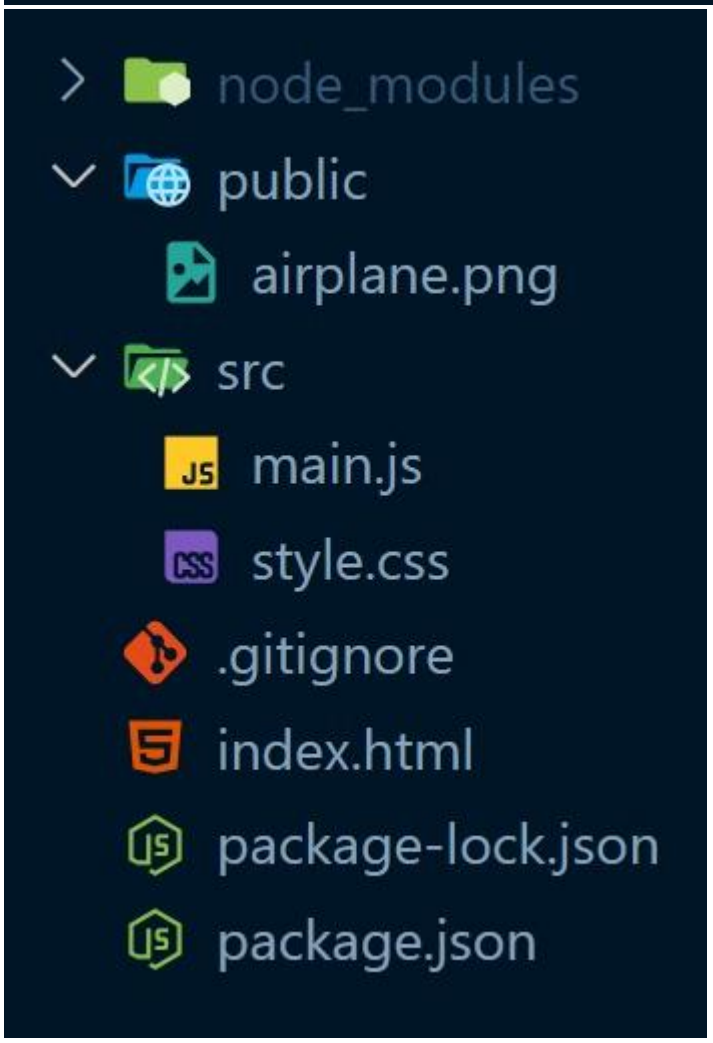
chennai

Gateway to south indias culture

Clear Visited Clear All


Travel Bucket List
localhost:4173
Visited 2 / 3 places
Add a new place to visit
Add notes (optional)
ADD PLACE
All
Not Visited
Visited
Mumbai
City of dreams, vibrant nightlife.
chennai
Gateway to south indias culture
Clear Visited
Clear All
travel-bucket-list-static-website.s3.ap-south-1.amazonaws.com/index.html
Travel Bucket List
Visited 0 / 1 places
Add a new place to visit
Add notes (optional)
ADD PLACE
All
Not Visited
Visited
Application
Manifest
Service workers
Storage
Storage
Local storage
Session storage
Extension storage
IndexedDB
Cookies
Private state tokens
Interest groups
Shared storage
Cache storage
Storage buckets
Background services
Back/forward cache
Background fetch
Background sync
Bounce tracking miti...
Notifications
https://travel-bucket-list-static-website.s3.ap-south-1.amazonaws.com
Origin
https://travel-bucket-list-static-website.s3.ap-south-1.amazonaws.com
Key
Value
bucketList
[{"name":"delhi","notes":"nice place","visited":false}]
loglevel
INFO
No value selected
Select a value to preview

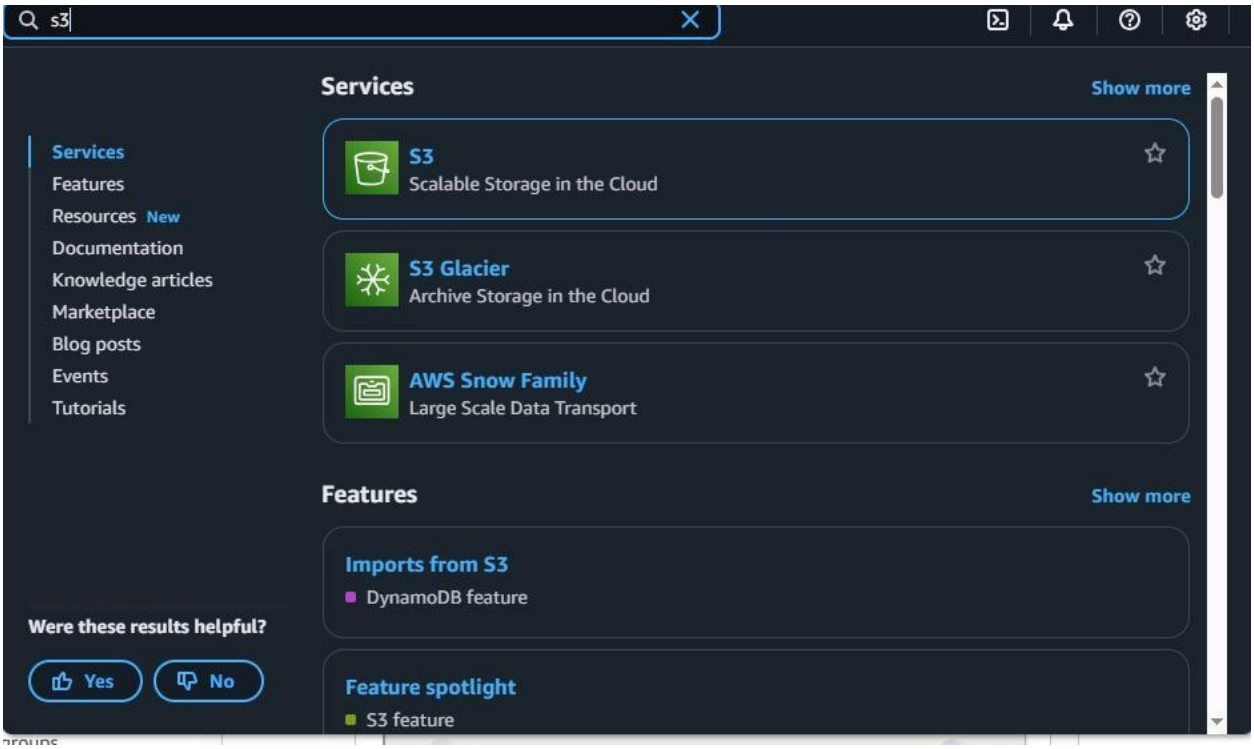

node_modules
public
airplane.png
src
main.js
style.css
.gitignore
index.html
package-lock.json
package.json
style.css
index.html
<!DOCTYPE html>
<html lang="en">
<head>
<link rel="icon" href="/airplane.png">
<link rel="stylesheet" href="./src/style.css" />
<meta charset="UTF-8" />
<meta
name="viewport"
content="width=device-width,initial-scale=1,maximum-scale=1,
user-scalable=no"
/>
<title>Travel Bucket List</title>
PORTS
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
bash
VITE v6.3.5 ready in 235 ms
Local: http://localhost:5173/
Network: use --host to expose
press h + enter to show help


node_modules
public
airplane.png
src
main.js
style.css
.gitignore
index.html
package-lock.json
package.json

Conclusion:

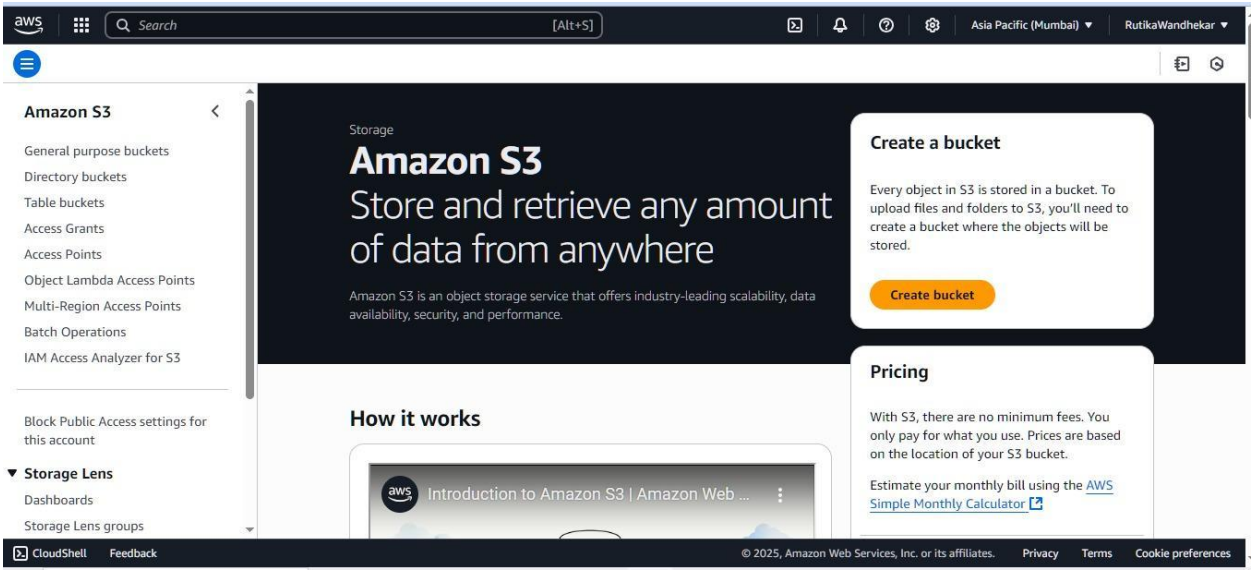
The TravelBucketList project demonstrates a clean and functional web application built with fundamental front-end technologies. It effectively enables users to maintain a list of dream travel destinations, supporting interactive features like adding, marking as visited, and removing places. The project showcases practical DOM manipulation and styling techniques, making it a strong example of beginner to intermediate-level web development. With a user-friendly interface and straightforward functionality, this application not only enhances user engagement but also encourages further exploration into dynamic JavaScript applications.

Step 1: Open S3 in AWS Console
Go to: https://s3.console.aws.amazon.com/s3
Search for S3 in the AWS Console and click on it.



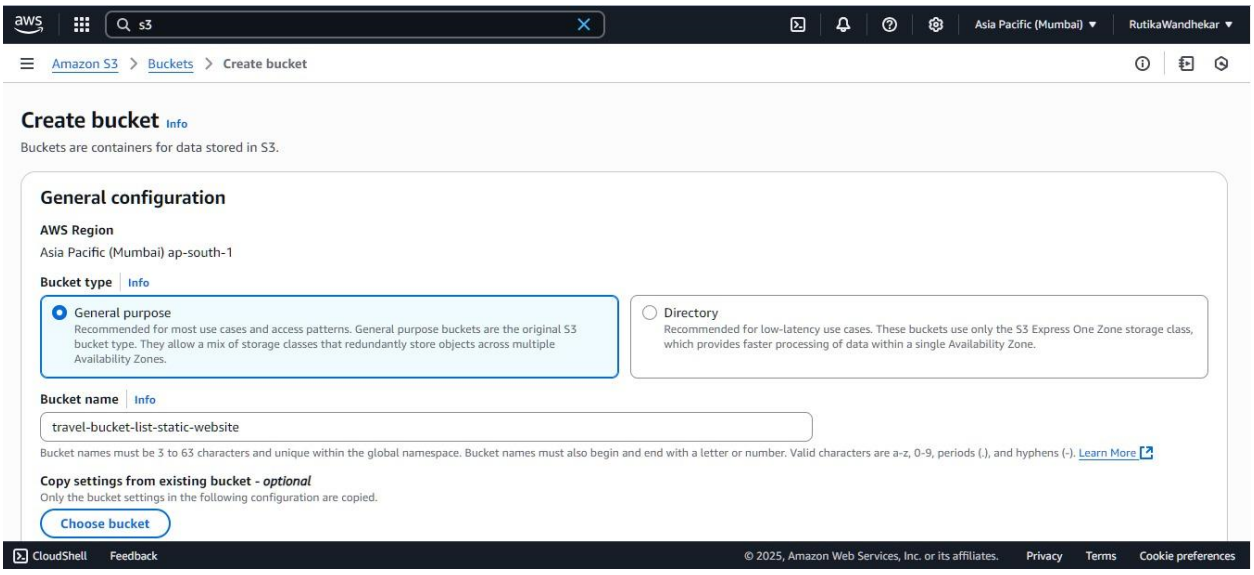
Step 2. Create a Bucket

- Click Create bucket
- Enter a unique bucket name (e.g., travel-bucket-list-site)
- Choose a region (e.g., ap-south-1)
- Uncheck: “Block all public access” under permissions
- Confirm by checking the warning box
- Click Create bucket



Step 3. Upload Website Files

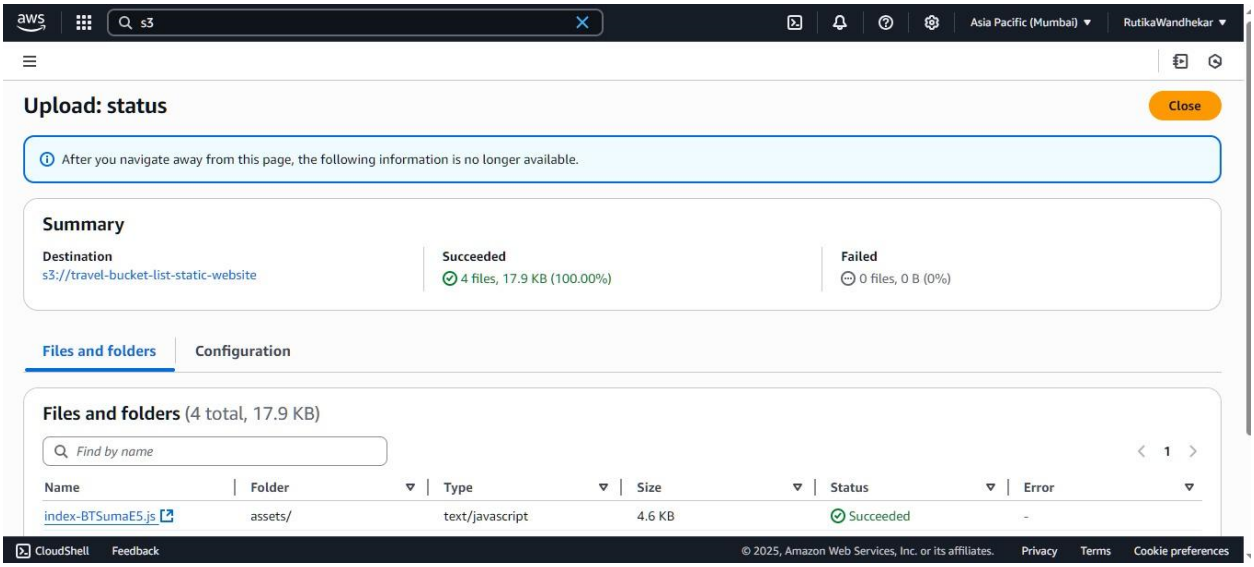
- Open your newly created bucket
- Go to the Objects tab
- Click Upload
- Add files like index.html, styles.css, script.js, images, etc.
- Click Upload



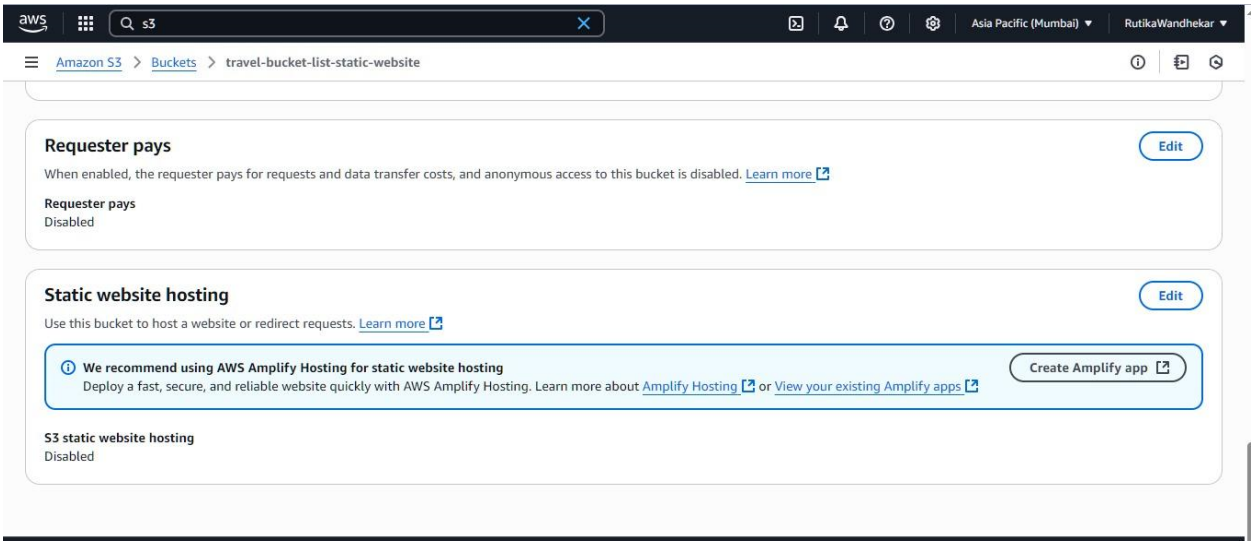
Create bucket

Step 4. Go to bucket

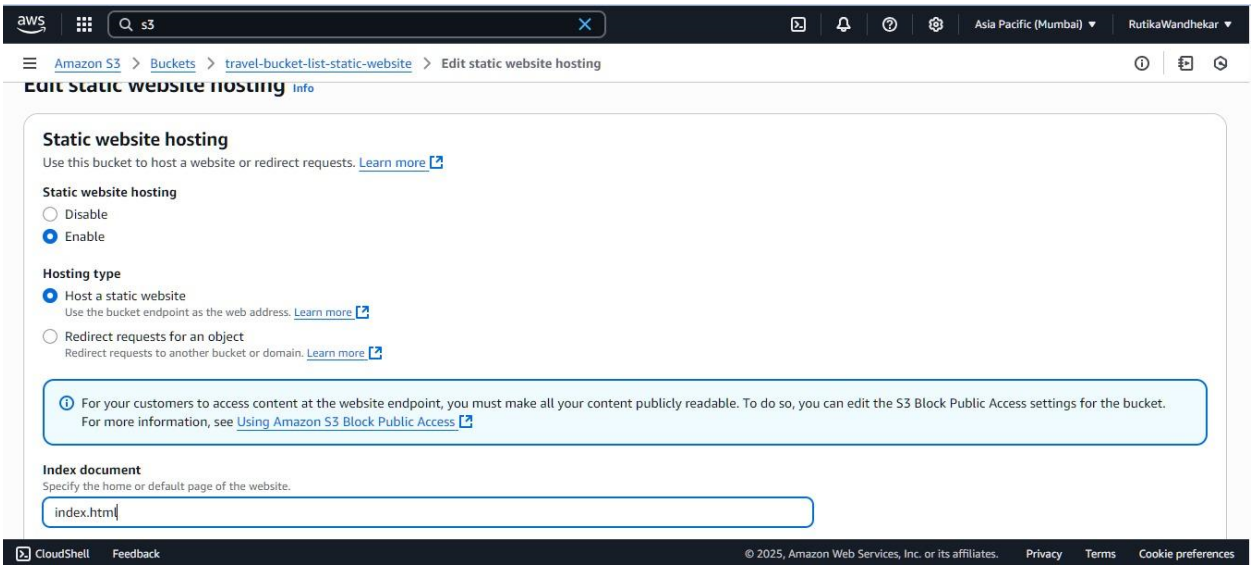
- Click on upload and upload the files.



Step 5. Go to properties
- static website hosting
- Edit

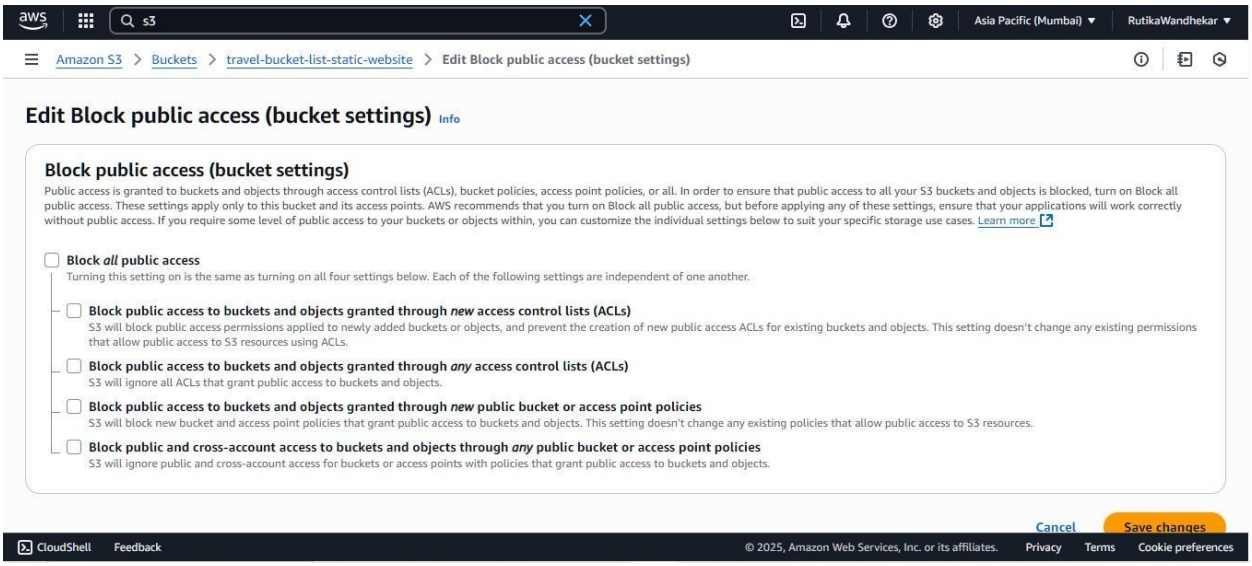


Step 6. Enable it
- Add index document: index.html
- Save



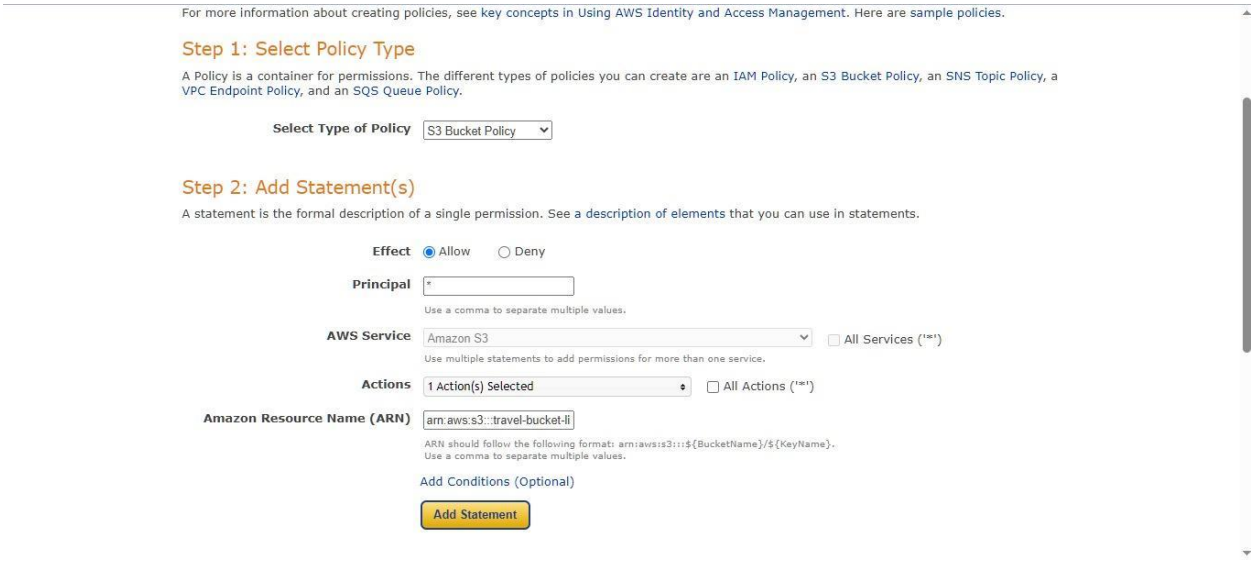
Step 7. Go to Permission

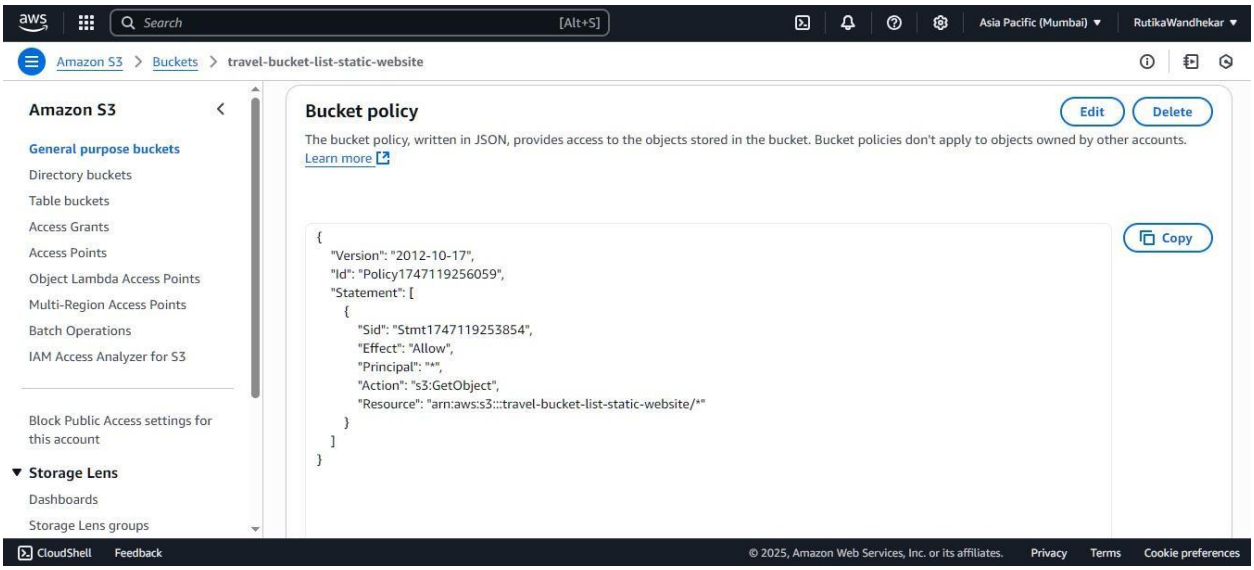
- block public access
- edit
- uncheck it
- save



Step 8. Go to Bucket policy

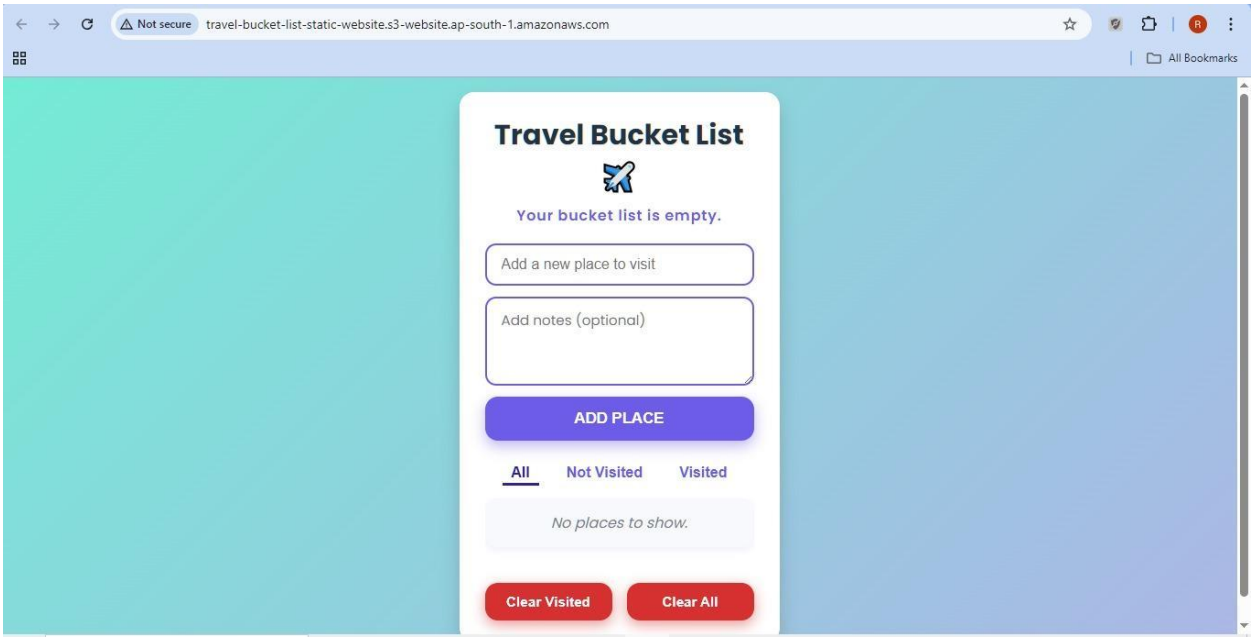
- edit
- policy generator
- select type of policy -S3
- effect - allow
- principal
- action - getobject
- arn-
- add statement
- generate policy
- copy and paste
- save





Step 9. Test the url

- Paste it on the browser - website should be live!



<u>Yogita Patil (TL)</u>

Developers

- Rasakatla Hari Priya
- Rohit Kunte
- Shaikh Affan Shaikh Usman
- Sakshi Wagh
- Shanmugapriyan AG
- Mudavath Siddu

Cloud Engineers

- Manish Kumar
- Rutika Wandhekar
- Neha Bagul