JAYPEE UNIVERSITY OF INFORMATION TECHNOLOGY

CLOUD COMPUTING: CONCEPTS, TECHNOLOGY & ARCHITECTURE(20B1WCI532)

Team Members-	Aakash Thakur (231030064)
	Prakhar Soumya (231030061)
Batch-	A13
Branch-	B- Tech CSE (Core)
Project Report -	TO DO LIST USING FIREBASE

Introduction :

Cloud computing has become an essential part of modern software development, enabling applications to store and process data online rather than on local devices. This project, **Cloud To-Do List**, is a simple yet powerful web application that uses **Firebase Realtime Database** to store user tasks in the cloud.

The application allows users to:

- Add tasks
- Mark tasks as completed
- Delete tasks
- Auto-sync data across devices in real-time

Objective of the Project :

The main goals of this project are:

1. To understand how cloud databases work.
2. To explore Firebase as a backend-as-a-service (BaaS) platform.
3. To implement real-time syncing of data between client and cloud.

4. To build a functional web application using HTML, CSS, and JavaScript.

5. To perform basic CRUD (Create, Read, Update, Delete) operations on a cloud database.

Technologies Used

Frontend:

- HTML5 – Structure of the webpage
- CSS3 – Styling the interface
- JavaScript (ES6) – Logic and Firebase operations

Cloud Platform:

- Firebase (Google Cloud)
    - Firebase Project
    - Realtime Database
    - Web App SDK (Modular version v9)

Development Tools:

- Code Editor (VS Code preferred)

- Local server (VSCode Live Server or Python HTTP server)
- Firebase Console

.

System Architecture :

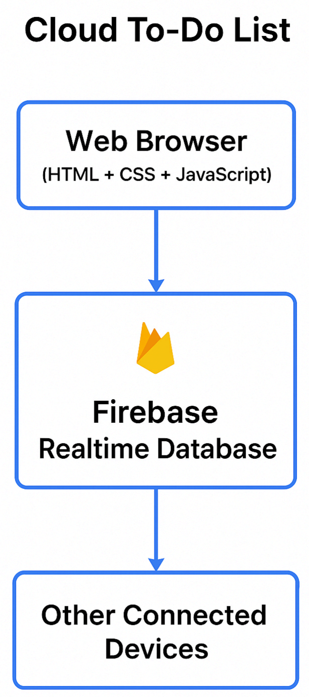
Client-Side (Browser)

- User interacts with the UI.
- JavaScript sends/receives data from Firebase using API calls.

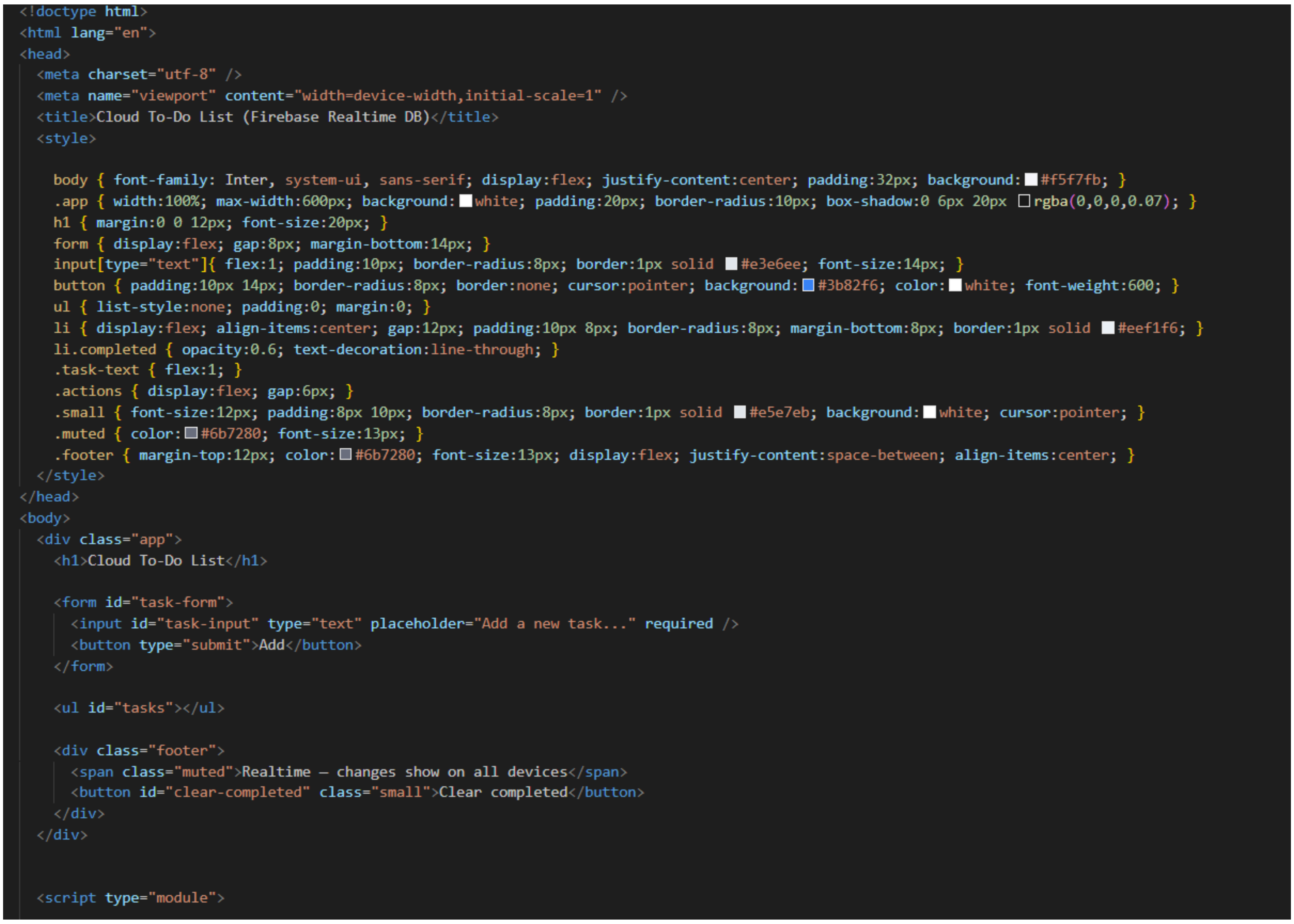
Cloud-Side (Firebase)

- Stores tasks inside a root node called tasks.
- Sends updates back to the UI instantly.
- Handles data synchronization.

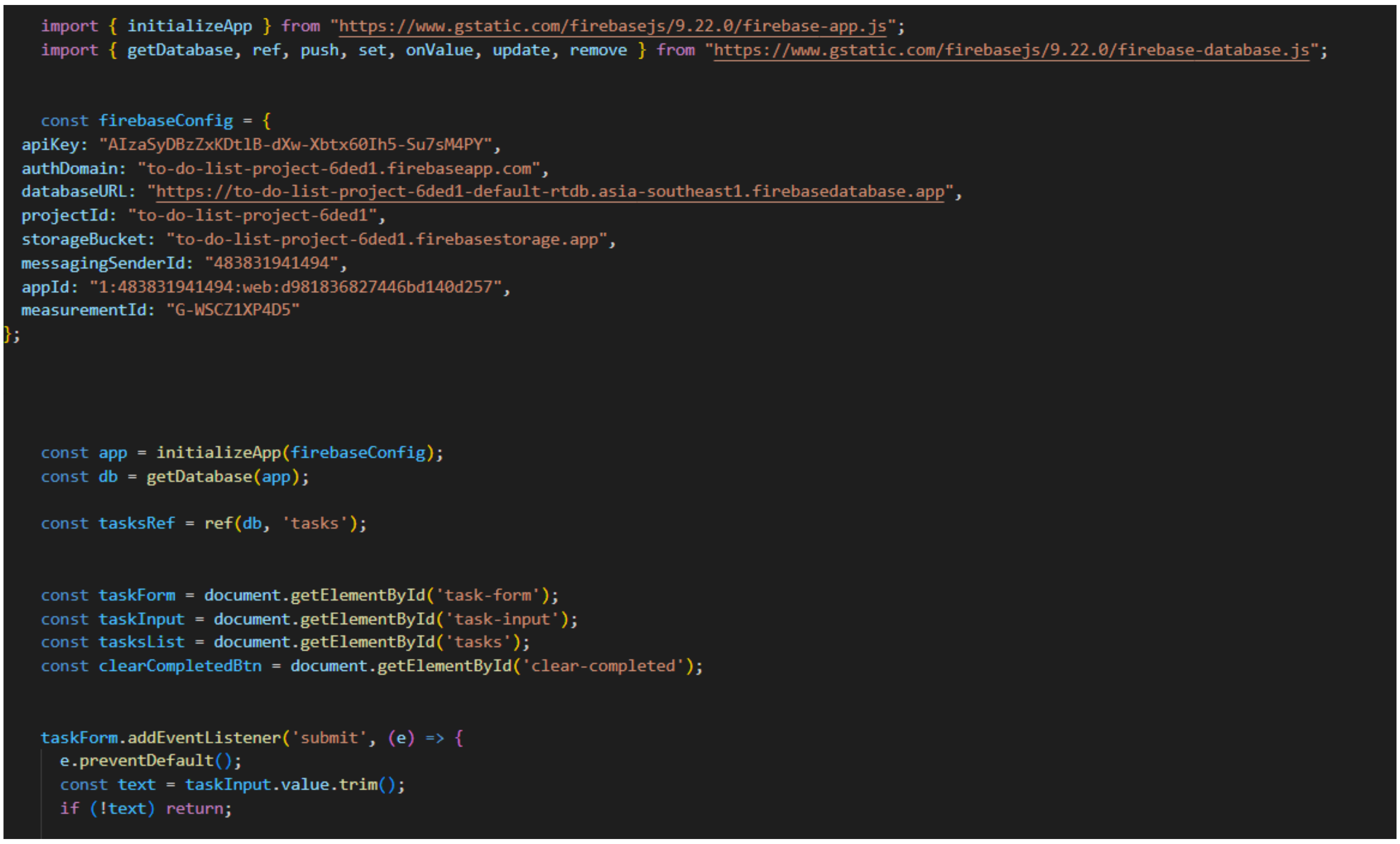
FLOW CHART :



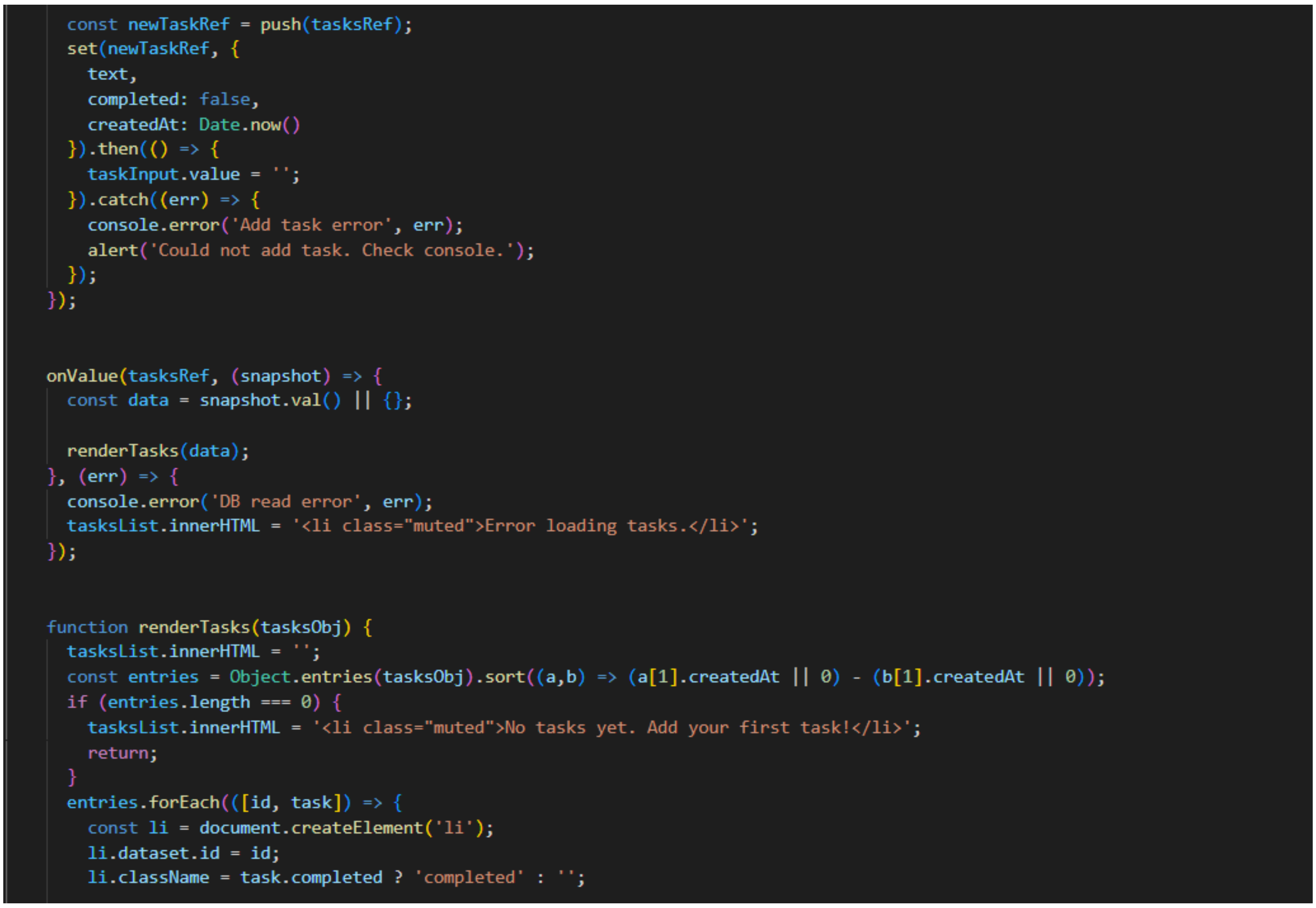
CODE :



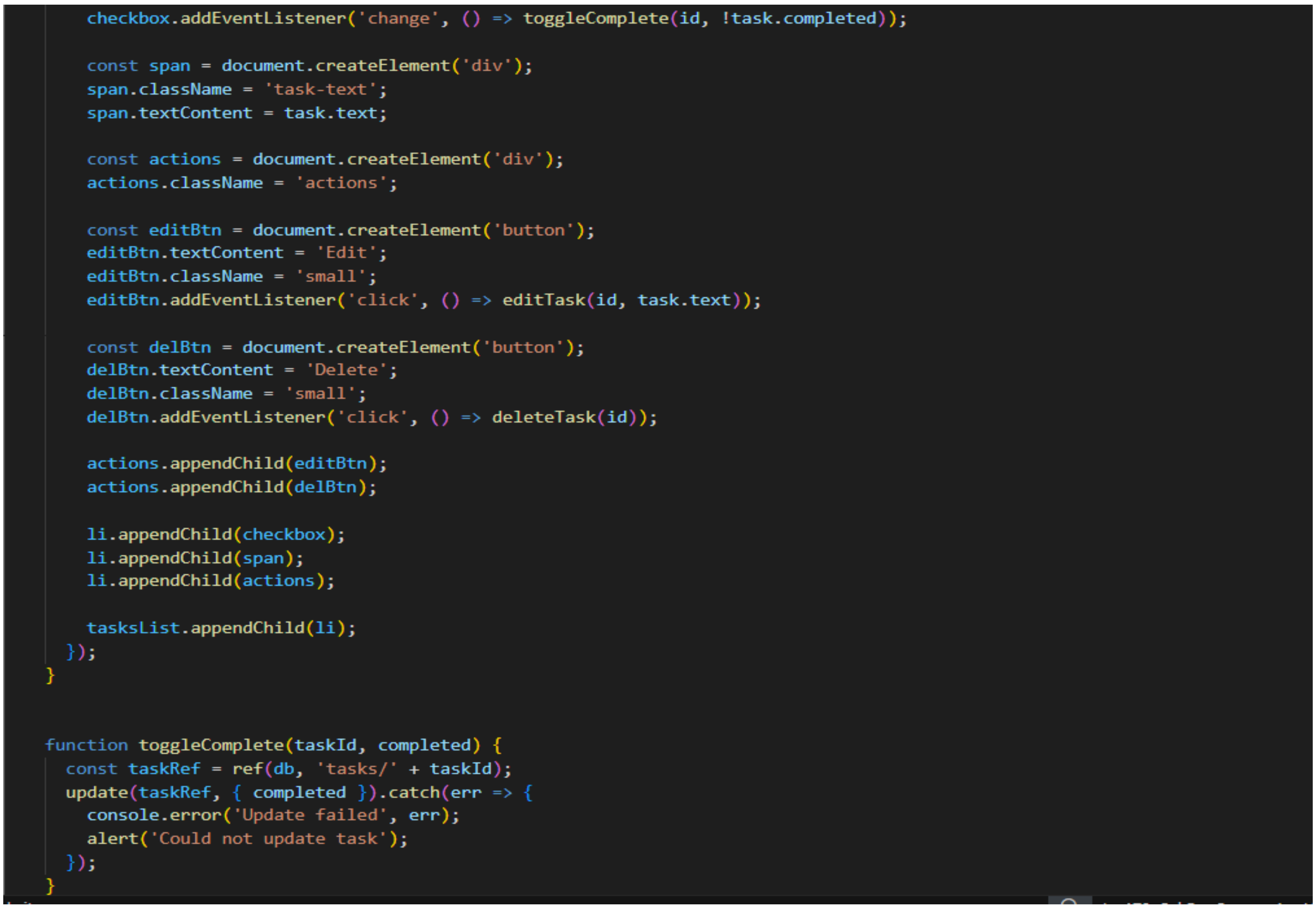
```html
<!doctype html>
<html lang="en">
<head>
  <meta charset="utf-8" />
  <meta name="viewport" content="width=device-width,initial-scale=1" />
  <title>Cloud To-Do List (Firebase Realtime DB)</title>
  <style>

    body { font-family: Inter, system-ui, sans-serif; display:flex; justify-content:center; padding:32px; background:#f5f7fb; }
    .app { width:100%; max-width:600px; background:white; padding:20px; border-radius:10px; box-shadow:0 6px 20px rgba(0,0,0,0.07); }
    h1 { margin:0 0 12px; font-size:20px; }
    form { display:flex; gap:8px; margin-bottom:14px; }
    input[type="text"]{ flex:1; padding:10px; border-radius:8px; border:1px solid #e3e6ee; font-size:14px; }
    button { padding:10px 14px; border-radius:8px; border:none; cursor:pointer; background:#3b82f6; color:white; font-weight:600; }
    ul { list-style:none; padding:0; margin:0; }
    li { display:flex; align-items:center; gap:12px; padding:10px 8px; border-radius:8px; margin-bottom:8px; border:1px solid #eef1f6; }
    li.completed { opacity:0.6; text-decoration:line-through; }
    .task-text { flex:1; }
    .actions { display:flex; gap:6px; }
    .small { font-size:12px; padding:8px 10px; border-radius:8px; border:1px solid #e5e7eb; background:white; cursor:pointer; }
    .muted { color:#6b7280; font-size:13px; }
    .footer { margin-top:12px; color:#6b7280; font-size:13px; display:flex; justify-content:space-between; align-items:center; }
  </style>
</head>
<body>
  <div class="app">
    <h1>Cloud To-Do List</h1>

    <form id="task-form">
      <input id="task-input" type="text" placeholder="Add a new task..." required />
      <button type="submit">Add</button>
    </form>

    <ul id="tasks"></ul>

    <div class="footer">
      <span class="muted">Realtime — changes show on all devices</span>
      <button id="clear-completed" class="small">Clear completed</button>
    </div>
  </div>


  <script type="module">
```



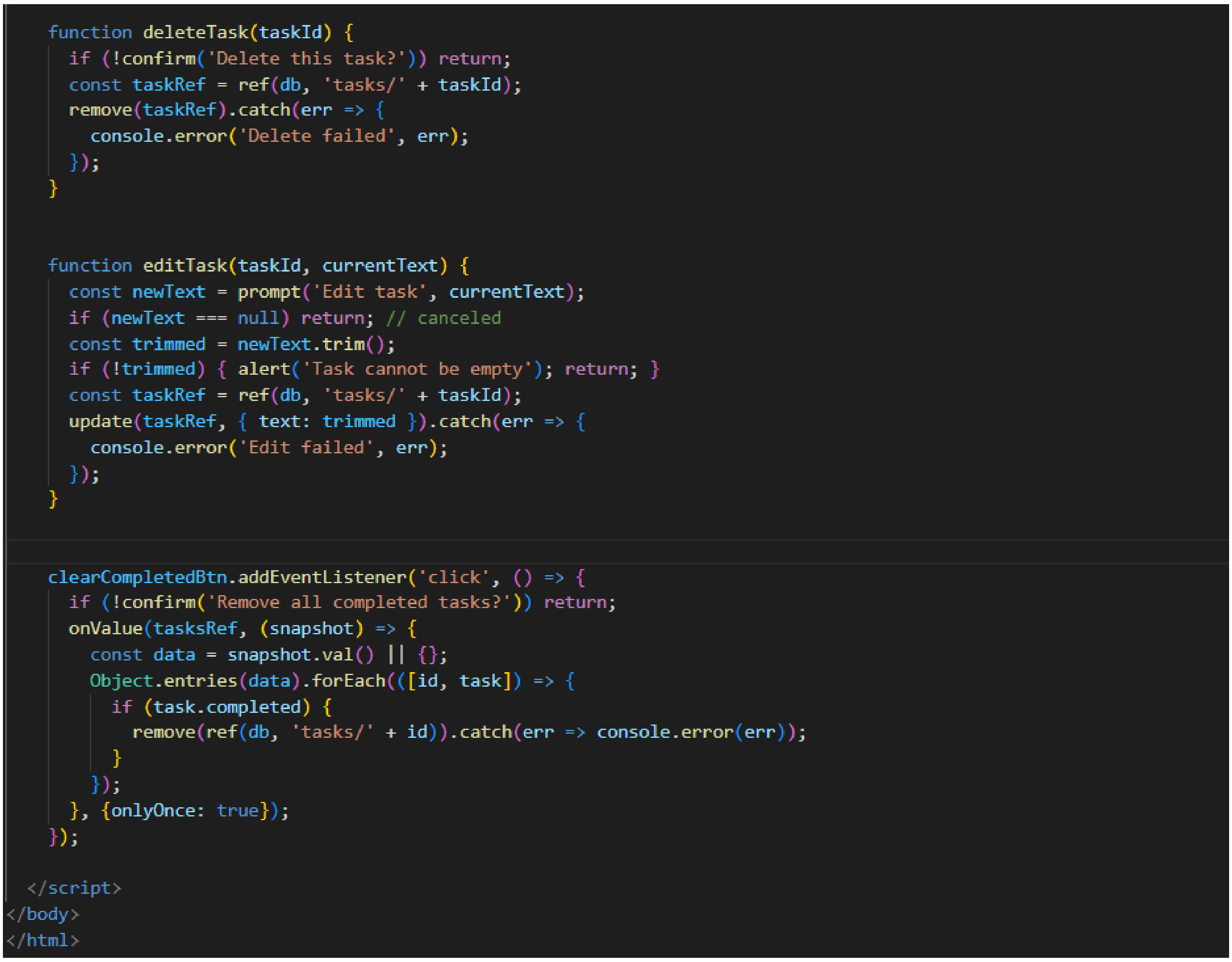
```javascript
  import { initializeApp } from "https://www.gstatic.com/firebasejs/9.22.0/firebase-app.js";
  import { getDatabase, ref, push, set, onValue, update, remove } from "https://www.gstatic.com/firebasejs/9.22.0/firebase-database.js";


  const firebaseConfig = {
 apiKey: "AIzaSyDBzZxKDtlB-dXw-Xbtx60Ih5-Su7sM4PY",
 authDomain: "to-do-list-project-6ded1.firebaseapp.com",
 databaseURL: "https://to-do-list-project-6ded1-default-rtdb.asia-southeast1.firebasedatabase.app",
 projectId: "to-do-list-project-6ded1",
 storageBucket: "to-do-list-project-6ded1.firebasestorage.app",
 messagingSenderId: "483831941494",
 appId: "1:483831941494:web:d981836827446bd140d257",
 measurementId: "G-WSCZ1XP4D5"
};




  const app = initializeApp(firebaseConfig);
  const db = getDatabase(app);

  const tasksRef = ref(db, 'tasks');


  const taskForm = document.getElementById('task-form');
  const taskInput = document.getElementById('task-input');
  const tasksList = document.getElementById('tasks');
  const clearCompletedBtn = document.getElementById('clear-completed');


  taskForm.addEventListener('submit', (e) => {
    e.preventDefault();
    const text = taskInput.value.trim();
    if (!text) return;
```



```javascript
    const newTaskRef = push(tasksRef);
    set(newTaskRef, {
      text,
      completed: false,
      createdAt: Date.now()
    }).then(() => {
      taskInput.value = '';
    }).catch((err) => {
      console.error('Add task error', err);
      alert('Could not add task. Check console.');
    });
  });


  onValue(tasksRef, (snapshot) => {
    const data = snapshot.val() || {};

    renderTasks(data);
  }, (err) => {
    console.error('DB read error', err);
    tasksList.innerHTML = '<li class="muted">Error loading tasks.</li>';
  });


  function renderTasks(tasksObj) {
    tasksList.innerHTML = '';
    const entries = Object.entries(tasksObj).sort((a,b) => (a[1].createdAt || 0) - (b[1].createdAt || 0));
    if (entries.length === 0) {
      tasksList.innerHTML = '<li class="muted">No tasks yet. Add your first task!</li>';
      return;
    }
    entries.forEach(([id, task]) => {
      const li = document.createElement('li');
      li.dataset.id = id;
      li.className = task.completed ? 'completed' : '';
```

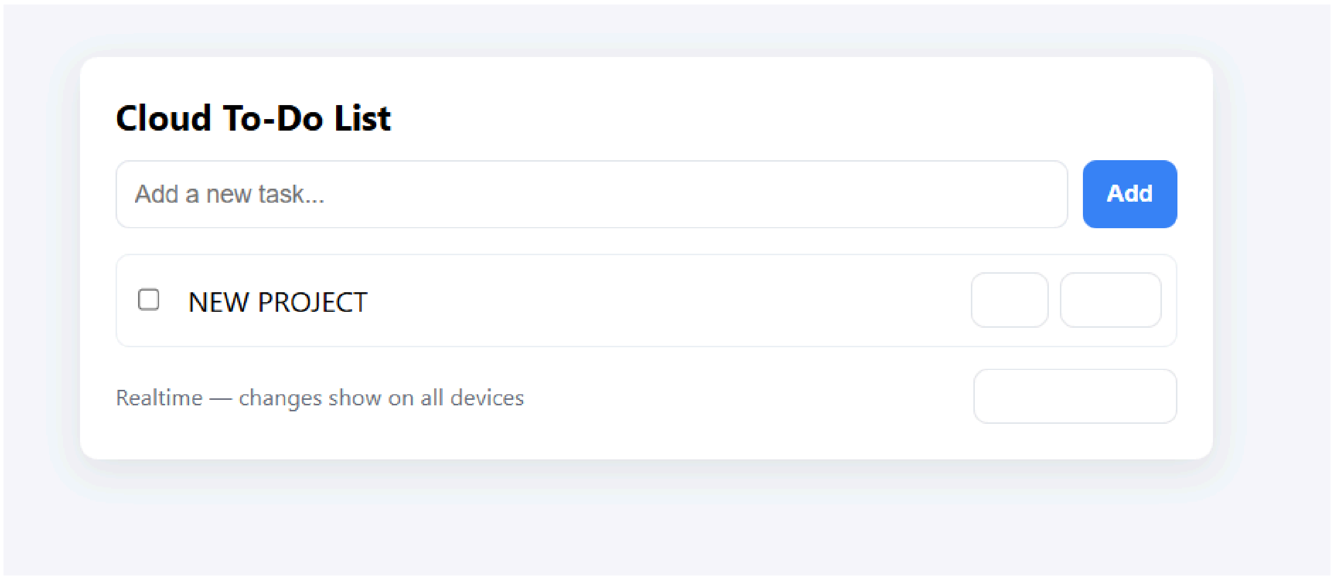


```
      checkbox.addEventListener('change', () => toggleComplete(id, !task.completed));

      const span = document.createElement('div');
      span.className = 'task-text';
      span.textContent = task.text;

      const actions = document.createElement('div');
      actions.className = 'actions';

      const editBtn = document.createElement('button');
      editBtn.textContent = 'Edit';
      editBtn.className = 'small';
      editBtn.addEventListener('click', () => editTask(id, task.text));

      const delBtn = document.createElement('button');
      delBtn.textContent = 'Delete';
      delBtn.className = 'small';
      delBtn.addEventListener('click', () => deleteTask(id));

      actions.appendChild(editBtn);
      actions.appendChild(delBtn);

      li.appendChild(checkbox);
      li.appendChild(span);
      li.appendChild(actions);

      tasksList.appendChild(li);
    });
  }


  function toggleComplete(taskId, completed) {
    const taskRef = ref(db, 'tasks/' + taskId);
    update(taskRef, { completed }).catch(err => {
      console.error('Update failed', err);
      alert('Could not update task');
    });
  }
```



```
    function deleteTask(taskId) {
      if (!confirm('Delete this task?')) return;
      const taskRef = ref(db, 'tasks/' + taskId);
      remove(taskRef).catch(err => {
        console.error('Delete failed', err);
      });
    }


    function editTask(taskId, currentText) {
      const newText = prompt('Edit task', currentText);
      if (newText === null) return; // canceled
      const trimmed = newText.trim();
      if (!trimmed) { alert('Task cannot be empty'); return; }
      const taskRef = ref(db, 'tasks/' + taskId);
      update(taskRef, { text: trimmed }).catch(err => {
        console.error('Edit failed', err);
      });
    }


    clearCompletedBtn.addEventListener('click', () => {
      if (!confirm('Remove all completed tasks?')) return;
      onValue(tasksRef, (snapshot) => {
        const data = snapshot.val() || {};
        Object.entries(data).forEach(([id, task]) => {
          if (task.completed) {
            remove(ref(db, 'tasks/' + id)).catch(err => console.error(err));
          }
        });
      }, {onlyOnce: true});
    });

  </script>
</body>
</html>
```

OUTPUT :


Cloud To-Do List
Add a new task...
Add
NEW PROJECT
Realtime — changes show on all devices