

A picture containing icon

Description automatically generated

**CCT College Assessment Cover Page**

|  |  |
| --- | --- |
| **Programme Title:** | BSc (Hons) in Computing in Information Technology |
| **Student Full Name:** | *Abhinav Borgohain, Mary Karanja, Jenny Priscila Matarrita Zuniga, Kah Chung Wong* |
| **Student Number:** | 2023287, 2023331, 2023104, 2023005 |
| **Module Title(s)**: | *Cross Platform Development* |
| **Assignment Title:** | Weather App |
| **Lecturer(s)**: | David González ([dgonzalez@cct.ie](mailto:dgonzalez@cct.ie)) |
| **Date of Submission:** | 27/05/2025 |
| **Submission Deadline Date:** | 27/05/2025 |

**Declaration**

**By submitting this assessment, I confirm that I have read the CCT policy on Academic Misconduct and understand the implications of submitting work that is not my own or does not appropriately reference material taken from a third party or other source. I declare it to be my own work and that all material from third parties has been appropriately referenced. I further confirm that this work has not previously been submitted for assessment by myself or someone else in CCT College Dublin or any other higher education institution.**

## **Weather App Project – Team Technical Report**

### **Team Members**

* **Jenny Priscila Matarrita Zuniga (2023104)** – Lead Programmer
* **Kah Chung Wong** **(2023005)**– UI/UX Design
* **Abhinav Borgohain (2023287)** – Testing
* **Mary Karanja (2023331)** – Documentation

### **Design Decisions**

The app was structured using a **tab-based layout** powered by expo-router, dividing functionality across four key screens:

* **Home** (current location weather)
* **Forecast** (5-day forecast)
* **Explore** (search by city – currently Dublin-only)
* **Settings** (toggle units and preferences)

**Design Goals:**

* Keep the user interface **clean and responsive**
* Use **background images** for visual appeal
* Modular components like ThemedText, ThemedView, and HapticTab for a consistent look and feel
* Support **both Android and iOS**, with in-app browser support for links

### **API Integration: Challenges & Solutions**

**API Used:** [Open-Meteo API](https://open-meteo.com/)

* No authentication required, which simplified integration
* Data fetched via standard fetch() calls with dynamic query parameters

**Challenges:**

* Handling device permissions via expo-location
* API errors if the user denied location access
* Explore page only supports **“Dublin”**, limiting dynamic search potential

**Solutions:**

* Fallback error messages like *“Permission denied”* and *“Only Dublin supported”*
* Wrapping fetch calls in try/catch blocks
* Showing appropriate loading indicators and error messages

### **State Management Approach**

The project uses **React’s built-in hooks**:

* useState() for local component state (weather data, loading, errors)
* useEffect() to trigger side effects like API calls after mount

**No global state library** (like Redux or Zustand) was needed due to the app’s small scope and single-purpose screens.

### **🔍 Reflections on the Development Process**

# Roles and Responsibilities

* **Project Manager/Documentation Specialist: Mary**
* - **\*\*Responsibilities\*\***: Manage project goals, timelines, worked with team members to ensure accurate documentation. Create and maintain comprehensive documentation, including project structure, API documentation, and testing strategies.
* **Frontend Developer (UI/UX): Kah Chung**
* - **\*\*Responsibilities\*\***: Develop the frontend of the application, including the user interface, navigation, and data fetching. Implement the weather data fetching and display logic. Test the frontend components and ensure responsiveness.
* **Backend Developer: Jenny**
* - **\*\*Responsibilities\*\***: Develop the backend of the application, including the API endpoints and data fetching. Implement the weather data fetching and display logic. Test the backend API endpoints and ensure responsiveness.
* **QA Tester: Abhinav**- **\*\*Responsibilities\*\***: Test the application thoroughly, including frontend and backend components. Ensure the application meets all requirements and performs as expected. Provide feedback to the development team for improvements incase of bug issues

**Lessons Learned:**

* Permissions and device APIs need special attention and fallback logic
* Component reusability helps reduce UI duplication
* Testing weather APIs can be tricky due to live data changes

### **🚀 Potential Future Improvements**

* Expand **Explore** to support multiple cities via a geocoding API (e.g., OpenCage or Mapbox)
* 🌡️ Allow switching between **Celsius and Fahrenheit** (hook into settings toggle)
* Add graphical visualizations (e.g., temperature trends)
* Offline caching using AsyncStorage or a lightweight database
* Add unit tests and integration tests with Jest or Detox
* Wrap into a PWA for web support (already partially functional on web via Expo)