🌤️ Weather Dashboard - Project Instructions

COVER PAGE

Application Name: Weather Dashboard

Group Number: Group 14

Platform: React Native using Expo CLI

Group Members

Role	Developer	Responsibilities
Developer 1	**Anupa Ragoonanan**	API Integration & Progress Management
Developer 2	**Harry Joseph**	Interactive Components & Gestures
Developer 3	**Raj Patel**	Modals, Notifications & Animations
Developer 4	**Kerlan Augustine**	Navigation & Project Documentation

Weather Dashboard Development Team 14

CPAN 213 - Mobile App Development Project

Phase 1 Submission - November 23, 2025

Phase 1

A. Purpose of the Application - Why is your application needed?

In today's mobile-first world, reliable weather information is essential for daily planning. While many weather apps exist, they often lack user-friendly interfaces, offline functionality, and personalized features.

Our **Weather Dashboard** addresses these gaps by providing:

- **Enhanced User Experience:** Dynamic animations and gesture navigation for engaging interactions
- **Offline Functionality:** Cached weather data accessible without internet connectivity
- **Cross-Platform Design:** React Native implementation for native performance
- **Multi-Location Support:** Essential for students and professionals monitoring multiple locations

B. What functionalities do you expect to implement?

Core Features:

- **Real-Time Weather:** Current conditions using OpenWeatherMap API (temperature, humidity, wind speed)
- **7-Day Forecast:** Extended weather predictions with daily highs/lows
- **Multi-Location Management:** Save and switch between multiple weather locations
- **Offline Capability:** AsyncStorage for cached data access without internet

Interactive Elements:

- **Touch Gestures:** Pull-to-refresh and swipe navigation
- **Animations:** Smooth transitions using React Native's Animated API
- **Modal Interfaces:** Location selection and detailed weather overlays
- **Push Notifications:** Weather alerts and daily updates

Technical Features:

- **API Integration:** Secure OpenWeatherMap calls with error handling

- **Navigation:** React Navigation between home, details, and settings screens
- **Data Persistence:** Local storage for user preferences and locations

C. What is the general structure of your application?

Our application follows React Native best practices with a modular architecture organized into screens (HomeScreen, DetailsScreen, SettingsScreen), reusable components (WeatherCard, LocationModal, NavigationBar), services for API integration and storage, and utility functions for consistent theming.

Architecture Flow: App.js initializes navigation, HomeScreen displays current weather, DetailsScreen shows extended forecasts, SettingsScreen manages locations, Services handle API calls and storage, Components ensure UI consistency.

D. How do you plan to distribute the functionalities among the group members?

Anupa Ragoonanan - API Integration & Data Management

- OpenWeatherMap API integration, error handling, and AsyncStorage implementation
- HomeScreen development and progress tracking

Harry Joseph - Interactive Components & User Experience

- DetailsScreen with extended forecast views and touch gestures
- Animations, pull-to-refresh functionality, and performance optimization

Raj Patel - Modals, Notifications & Visual Polish

- Modal components, push notification system, and SettingsScreen
- UI/UX consistency and visual design

Kerlan Augustine - Navigation & Project Integration

- React Navigation setup, architecture design, and project management
- Documentation, testing coordination, and quality assurance

Collaboration: Weekly meetings, shared components, Git workflow, and continuous integration.