COLLEGE CODE : 9509

COLLEGE NAME:HOLYCROSS ENGINEERING COLLEGE

DEPARTMENT:CSE

STUDENT NM-ID:A21ED64081D8052AD7D4921642CB1DB7

Roll No:950923104015

Date:15.09.2025

Completed the project named as Phase 2

TECHNOLOGY PROJECT NAME:IBM-FE-Live Weather Dashboard

Submitted by,

Name:Isool Rabiya N

Mobile No:6369428920

1. Tech Stack Selection

Frontend:

- Framework/Library: React.js (for component-based UI)
- Styling: Tailwind CSS (lightweight, responsive styling)
- Visualization: Chart.js / Recharts (for weather trend graphs)

Backend:

- Server: Node.js + Express (API integration & routing)
- Weather API: OpenWeatherMap / WeatherAPI (real-time weather

data) **Database:** MongoDB (for storing user preferences, recent searches,

caching data)

- Deployment & Others:
- Hosting: Vercel / Netlify (frontend), Heroku / AWS (backend)
- Version Control: Git + GitHub
- Authentication (optional): Firebase / JWT

2. UI Structure / API Schema Design

UI Structure:

Header/Navbar → App name, search bar, settings

Main Dashboard

Current weather card (temperature, location, icon)

Weather details (humidity, pressure, wind speed, sunrise/sunset)

Forecast section (next 5 days)

Graph (temperature trend, rain probability)

Sidebar/Settings → Units (C/F), theme switcher, saved

locations API Schema:

{

```
 “location”: “Chennai”,
 “coordinates”: { “lat”: 13.0827, “lon”: 80.2707 },
 “current”: {
  “temperature”: 30,
  “humidity”: 78,
  “pressure”: 1012,
  “wind_speed”: 4.5,
  “weather”: “Cloudy”,
  “icon”: “04d”
 },
 “forecast”: [
  { “date”: “2025-09-15”, “temp”: 29, “weather”: “Rainy” },
  { “date”: “2025-09-16”, “temp”: 31, “weather”: “Sunny” }
 ]
}
```


3. Data Handling Approach

- Fetch weather data via REST API calls.
- Cache frequently used locations in MongoDB to reduce API calls.
- Transform API data into normalized format before rendering.
- Error Handling: Show fallback message if API fails (e.g., “Unable to load weather data”).

- State Management: React Context API / Redux for global state (selected city, theme, units).

4. Component / Module Diagram

Components:

App → Root container

Navbar → Search + settings

WeatherCard → Current weather

WeatherDetails → Humidity, pressure, wind

ForecastList → 5-day forecast

WeatherGraph → Trend visualization

SettingsPanel → Theme, units

App

├— Navbar

├— WeatherCard

├— WeatherDetails

├── ForecastList

├── WeatherGraph

└── SettingsPanel

5. Basic Flow Diagram

User → Search City → API Call (Weather API)

↓

Backend (Node.js) → Cache/DB Check → Fetch if not available

↓

Process Data → Send Response

↓

Frontend (React) → Update State → Render Dashboard