Weather Service Backend Design Document

# Introduction

The Weather Service backend provides weather data for multiple cities, allowing users to retrieve current weather conditions and detailed forecasts. This document outlines the system architecture and design patterns to be followed in the service's API.

# System Architecture

The backend service will follow a layered architecture, consisting of the following components:

* API Layer: Exposes RESTful endpoints to interact with the weather data.
* Data Access Layer: Manages integration with the external weather API providers.

# Technologies

The backend service will be built using the following technologies:

* Programming Language: JavaScript and Node.js
* Frameworks: Express.
* Database: **N.A.**
* External APIs: OpenWeatherMap or WeatherAPI API

# Design Patterns

The backend service will follow the following design patterns:

* Model-View-Controller (MVC): Separates the application into three components - Models (data), Views (API endpoints), and Controllers (business logic).
* Pagination Pattern: Implements pagination for retrieving weather data in chunks to improve performance.
* Filtering Pattern: Enables filtering of weather data by city name, city code, date, and moment to provide customized results.

# API Endpoints

The backend service will expose the following API endpoints:

* **/weather**: Retrieves weather data of multiple cities, supporting filtering by city name or city code, and pagination.
* **/forecast**: Provides a detailed forecast for the next X days, allowing filtering by city, date, and moment.
* **/current**: Retrieves the current weather conditions of a particular city.

Each endpoint will specify the required parameters and request/response formats.

# Authentication and Authorization

To secure the API, implement authentication and authorization mechanisms. This can include token-based authentication (e.g., JWT) and role-based access control to restrict access to certain endpoints.