Order Service with Pharmacy Integration - Implementation Report

GitHub Repo Link: https://github.com/joyal7701/PharmaOrderIntegration

Design Decisions:

Project Structure:

The project is organized into several directories for clarity and separation of concerns:

src: Contains the source code of the application.

pharmacies: Holds pharmacy-related classes, each representing a specific pharmacy and its operations.

routes: Defines the routes and their corresponding handlers for the application.

app.ts: Initializes the Express application, sets up middleware, and defines the main application logic.

server.ts: Starts the Express server.

test: Holds unit tests for the pharmacies, and general routes.

package.json: Lists project dependencies and configuration.

tsconfig.json: TypeScript configuration file.

jest.config.js: Jest configuration file.

README.md: Markdown file for GitHub

Scalability and Extensibility:

- Pharmacy Abstract Classes: A common abstract class Pharmacy is created to define the structure that each pharmacy should follow. This ensures consistency and allows for easy addition of new pharmacies in the future.
- Dynamic Route Generation: The routes are dynamically generated based on the available pharmacies. This design ensures that adding a new pharmacy only requires creating a new class and adding it to the `pharmacies` object in the `pharmacy.ts` file.

Unit Tests:

- Unit tests are written using the Supertest library with Jest to test the API endpoints for each pharmacy. The tests cover various scenarios, including creating orders, retrieving orders, and handling invalid payloads.
- Separate test files are created for each pharmacy and general routes to keep the tests organized.

Assumptions:

- In-memory data storage approach is used for this assessment to store orders for each pharmacy.
- The provided sample payloads are used as a basis for designing the order creation endpoints for each pharmacy.

Instructions for Running the Application and Tests:

- Install Dependencies:
  npm install
- Run the Application:
  npm start
  The server will be running at `http://localhost:3000`.
- Run Tests:
  npm test
  This command will execute all the unit tests using the Jest - Supertest framework.