**Real-Time Cab Booking & Tracking System – Capstone Project**

**Project Goal**

Build a **real-time cab booking and tracking platform** with **Spring Boot (REST + Reactive APIs)**, **Apache Kafka**, **Docker**, and a **CI/CD pipeline**. The system must leverage **AI tools**:

* **ChatGPT** → for explanations, detailed documentation, architecture brainstorming, troubleshooting, and generating user-facing chatbot responses.
* **GitHub Copilot (and similar tools)** → for rapid code generation, boilerplate setup, and automated unit/integration test creation.

**Business Need**

The cab company struggles with:

* **Delayed booking confirmations** for customers.
* **Slow driver notifications** when rides are assigned.
* **Unreliable real-time tracking** under heavy traffic load.
* **Manual surge pricing**, leading to unfair fares.
* **Slow deployment cycles** with frequent downtime.

They want a **modern, scalable, fault-tolerant system** that uses **event-driven architecture, reactive APIs, and AI-assisted development** to solve these challenges.

**Core Challenges**

1. **Concurrency & Real-Time Responsiveness**
   * Handle thousands of ride requests concurrently.
   * Send booking confirmations to customers within **2 seconds**.
   * Ensure drivers receive assignments instantly.
2. **REST vs Reactive APIs**
   * Use **normal REST APIs** for low-concurrency operations (signup, feedback, payments).
   * Use **Reactive APIs** for high-throughput, real-time scenarios (driver location streaming, ride status updates).
3. **Event Streaming with Kafka**
   * Booking confirmations → published asynchronously to avoid blocking.
   * Driver location updates → streamed via Kafka topics.
   * Notification service → consumes ride status events and pushes updates to customers.
4. **Deployment & CI/CD**
   * Containerize all services with **Docker**.
   * Automate build, test, and deployment with **Jenkins/GitHub Actions**.
   * Ensure **zero downtime deployments**.
5. **AI Tool Integration**
   * **ChatGPT**
     + Provide detailed explanations of architectural trade-offs (e.g., “Why Kafka vs RabbitMQ?”).
     + Help debug errors with context-based suggestions.
     + Generate **customer chatbot responses** (“Where is my driver?”, “Cancel my ride”, “How much will this cost?”).
   * **GitHub Copilot**
     + Generate **Spring Boot controller/service/repository boilerplate code**.
     + Suggest **unit and integration tests** for REST + Reactive APIs.
     + Auto-complete repetitive CRUD methods.
   * **AI Code Review Tools**
     + Analyze pull requests for performance and security issues.
   * **AI Testing Tools**
     + Generate synthetic data for load testing.
     + Automate test case creation for Kafka consumers/producers.

**Modules to Develop**

1. **Customer Service (Normal REST APIs)**
   * User registration & login
   * Booking cab requests
   * Payment processing
   * Feedback submission
2. **Driver Service (Reactive APIs + Kafka)**
   * Driver login & availability status
   * Receive bookings via Kafka consumer
   * Update ride status via reactive APIs
   * Stream live GPS location updates
3. **Ride Management Service (Reactive APIs + Kafka)**
   * Assign nearest available driver
   * Manage ride lifecycle (Booked → Ongoing → Completed)
   * Handle ride cancellations
4. **Notification Service (Reactive APIs + Kafka)**
   * Send real-time ride updates (SMS/Push)
   * Deliver trip summaries & invoices
5. **Analytics & AI Service**
   * Surge pricing engine (AI/ML-based demand prediction)
   * Driver performance analysis
   * AI-powered customer support (via ChatGPT API)

**Expected Deliverables**

* **Spring Boot Services**: REST + Reactive APIs for customers, drivers, rides, and notifications.
* **Kafka Topics**: For booking confirmations, driver availability, ride status, and GPS updates.
* **Dockerized Microservices**: Each service runs independently.
* **CI/CD Pipeline**: Automated build, test, deploy (GitHub Actions/Jenkins).
* **AI-Integrated Features**:
  + ChatGPT for explanations + customer chatbot.
  + Copilot for code generation + automated tests.
  + AI tools for code review and load testing.

**Real-World Constraints**

* Must scale to **1000+ requests per second**.
* Booking confirmation time **≤ 2 seconds**.
* Fault-tolerant Kafka processing with retry mechanisms.
* Must support **zero downtime CI/CD deployments**.
* AI tools should **assist but not replace developer validation**.

This makes it a **true real-time capstone project**:

* Covers **Spring Boot REST & Reactive APIs**.
* Uses **Kafka for real-time events**.
* Deploys with **Docker + CI/CD**.
* Embeds **AI tools (ChatGPT + Copilot)** in both **development** and **production support** stages.

Do you want me to now **break this into a milestone-based execution roadmap (week-wise)** where I’ll also show *exactly where ChatGPT and Copilot will be applied* in each stage?