**Step 1: Project Setup**

**1. Create a GitHub Repository**

* **Name the Repository:** Choose a clear and professional name, like ecommerce-microservices.
* **Initialize the Repository:** Add a README.md file where you'll outline the project's goals, architecture, and setup instructions.
* **Branch Strategy:** Implement a branching strategy (e.g., main for production-ready code, develop for ongoing development, and feature branches for individual microservices).

**2. Set Up the Basic Structure**

* **Multi-Module Maven/Gradle Project:**
  + Create a root project directory.
  + Under this directory, set up sub-modules for each microservice (e.g., user-service, product-service, order-service, etc.).
  + Ensure each module has its own pom.xml (for Maven) or build.gradle (for Gradle), and add common dependencies to the parent project.

**3. Initialize Spring Boot Applications**

* **Basic Spring Boot Setup:**
  + For each microservice module, initialize a Spring Boot application.
  + Add necessary dependencies for each service. For example, the user-service might include Spring Web, Spring Data JPA, Spring Security, etc.
* **Configuring Application Properties:**
  + For now, use basic application properties (application.properties or application.yml), but plan to externalize configurations later using Spring Cloud Config or Kubernetes ConfigMaps/Secrets.

**4. Version Control and Initial Commit**

* **Git Commit:** Once the basic structure is set up, make an initial commit with a clear message like “Initial project setup with multi-module structure.”

**Next Steps:**

Once the basic project setup is complete, we can move on to developing the User Service. This will involve creating models, repositories, controllers, and setting up security.