

# Mini E-Commerce Practice Project

- **GitHub-first workflow (mandatory)**
- Clear **software engineering deliverables** (requirements → UML → DB → implementation)
- Simple, enforceable **checklist + grading** (optional but helpful)

By the end of this practice, you will be able to:

- Design database tables correctly
- Define relationships between entities
- Normalize tables up to **3NF**
- Map **UML Class Diagrams** → **Database Tables** → **Code**
- Implement a **local database** inside a mobile app
- Use **GitHub** properly (repo, commits, branches, README)

---

## 1) GitHub Setup (Must be done first)

Before any design work:

1. Create a **new GitHub repository** named:

**MiniECommerce-DB-Practice-GroupX** (replace X)

2. Add a clear **README.md** that includes:
  - Project title
  - Team members
  - Tools used (Android/Room or SQLite, etc.)
  - Short description of what you will build
3. Create these folders in the repo:
  - /docs (requirements + UML + ERD + screenshots)
  - /app (source code)
4. Minimum GitHub requirements:
  - **Perform meaningful commits per group**
  - Commits must be descriptive (e.g., “block of code”, “Add Product entity + initial ERD”)
  - Use **branches** (at least one branch + a merge back to main)

## 2) Software Engineering Technique & Database Design

Students must apply these SE techniques:

### A) Requirements & Analysis (Software Engineering)

#### 1. Entity Identification

- UML Diagrams (Analysis Level):
  - Use Case Diagram
  - Activity Diagram
- UML Diagrams (Design Level):
  - Class Diagram
  - Sequence Diagram
  - Deployment Diagram

### B) Entity Relationships (ER) Diagram

Each group must create an ER Diagram and data schema showing:

- Entities + attributes
- PK and FK
- Cardinality (1-to-Many)

### C) Database Design

#### 1. Normalization (Up to 3NF)

- Remove redundancy
- Ensure data integrity

#### 2. ER Modeling

- Define:
  - Primary Keys (PK)
  - Foreign Keys (FK)
  - Cardinality (1-to-Many)

### 3) Identify Database Entities (Mini E-Commerce)

From the system requirements, identify **at least** these entities:

Entity	Description
Product	Stores product information
Cart	Represents the shopping cart
CartItem	Links products to the cart

You may add more tables if needed (recommended), such as:

- **User** (if you want multi-user carts)
- **Category** (products grouped)
- **Order** and **OrderItem** (if you extend beyond cart)

### 4) Define Relationships

You must implement and explain these relationships:

- **Cart ↔ CartItem**
  - One cart has many cart items
  - **(1-to-Many)**
- **Product ↔ CartItem**
  - One product can appear in many cart items
  - **(1-to-Many)**

## 5) Database Implementation

Implement a local database or Cloud database using one option:

### Android

- SQLite

### iOS

- Core Data
- SQLite

### Cloud database

- You can use cloud database.

## 6) Mapping UML → Database → Code

You must map all the discussed above together.

## 7) Minimum App Features (Simple UI)

Your app must demonstrate database operations:

1. **Insert Products** (hardcoded seed or simple form)
2. **List Products**
3. **Add to Cart**
4. **View Cart Items**
5. **Update Quantity**
6. **Remove Item**

Provide screenshots in:

- /docs/screenshots/ (in GitHub)

## 8) Required Submission Structure (GitHub Repo)

Your final repository must contain:

- /docs
  - Requirements (short)
  - UML Class Diagram (image/PDF)
  - ER Diagram (image/PDF)
  - Mapping explanation
  - Screenshots
- /app
  - Source code
- README.md
  - How to run the app + database choice + features list