

# Week 1 – Frontend Design & UI Planning

## Objective:

Design and finalize the complete frontend structure before backend work.

## Activities:

- Design user-side screens using HTML, CSS, and Bootstrap concepts (Home, Product List, Product Detail, Cart, Checkout, Orders, Wallet)
- Design admin-side screens (Product Management, Order Management, Coupon Management, Sales & Reports)
- Define navigation flow and screen hierarchy
- Finalize Figma designs as UI reference

## Deliverables:

- Complete frontend screen list
- Figma designs and UI flow diagrams

# Week 2 – Backend Architecture & API Design

These modules are done **first** because **everything else depends on them**.

## 1. Authentication & Authorization Module

### Why first:

- Required to identify users and admins
- Enables role-based access for all other modules

## 2. User Management Module

### Why next:

- Orders, wallet, reviews, and addresses all depend on users

- Address management is required before checkout

### **3. Category & Brand Management Module**

**Why early:**

- Products cannot exist without categories and brands
- Required for product classification and reporting

### **4. Product Catalog Management Module**

**Why now:**

- Core business entity
- Required before cart, inventory, or orders

### **5. Inventory & Stock Management Module**

**Why after products:**

- Inventory is meaningless without products
- Required to validate cart and checkout

## **Week 3 – Business Logic & Workflow Enforcement**

These modules sit **on top of the foundation** and enforce system behavior.

### **6. Cart Management Module**

**Why here:**

- Depends on users, products, and inventory
- Acts as a staging area before order creation

### **7. Checkout & Order Creation Module**

**Why next:**

- Converts cart into an immutable order

- Locks pricing, address snapshot, and payment intent

## **8. Payment & Wallet Management Module**

**Why after order creation:**

- Payments must reference an existing order
- Wallet deductions and refunds depend on order state

## **9. Order Lifecycle Management Module**

**Why now:**

- Enforces valid order state transitions
- Depends on payment outcomes

## **10. Administrative Order Management Module**

**Why after lifecycle rules:**

- Admin actions must respect lifecycle constraints
- Prevents illegal manual changes

These modules enhance functionality but depend on core workflows.

## **11. Coupon Management Module**

**Why here:**

- Applied during checkout
- Depends on cart and order structure

## **12. Review & Ratings Module**

**Why later:**

- Depends on completed orders
- Tied to verified purchases

## **13. Notification Module**

**Why later:**

- Triggered by events (order created, payment success)
- No independent business logic

## **14. File & Media Management Module**

**Why here:**

- Supports product images and user profiles
- Independent but not blocking core workflows

## **15. Sales & Reporting Module**

**Why late:**

- Depends on orders, payments, products, and brands
- Read-only analytical module

## **16. System Settings & Audit Module**

**Why last:**

- Observes actions from all other modules
- Requires stable workflows to log

# **Week 4 – Corrections, Testing & Deployment**

**No new modules — only:**

- **Validation**
- **Corrections**
- **Deployment**