Week 1: Initial Setup and Core Features Development

Objective: Set up the project and build the foundational features for the Admin and Customer modules.

Day 1-2: Project Setup

- Create a GitHub repository and set up the project environment.
- Define the project structure and install necessary Flutter dependencies (e.g., provider for state management, sqflite or firebase for database).
- Create a shared database schema:
    - products table: id, name, price, description, quantity, photo.
    - orders table: id, customer_id, product_id, quantity, status, transaction_proof.
- Test database connection.

Day 3-4: Admin Authentication

- Implement an admin login page with form validation.
- Hardcode admin credentials for simplicity or set up authentication logic.

Day 5-7: Product Management (Admin)

- Design and implement the Add Product page with form fields for product details.
- Allow adding photos (optional) using the device camera or gallery.
- Implement functionality to remove products from the database.
- Test adding and removing products functionality.

Week 2: Customer Features and Cart Functionality

Objective: Develop customer-facing features, including product browsing, cart management, and order placement.

Day 8-10: Product Browsing (Customer)

- Create a customer home page to display the list of products from the database.
    - Use a grid or list view to show product name, price, and photo.
    - Include a search bar to filter products by name or category (optional).
- Implement product detail view when a product is clicked.

Day 11-12: Cart Management

- Implement "Add to Cart" functionality.
    - Allow customers to specify product quantities.
    - Store cart items in a local database or state.
- Create a Cart page to display selected items with the total price.
    - Allow customers to update item quantities or remove items from the cart.

Day 13-14: Order Placement

- Implement "Place Order" functionality.
    - Validate cart items and quantities before placing an order.
    - Store order details in the orders table.
- Add an optional login prompt for customers placing orders.

Week 3: Advanced Features and Final Testing

Objective: Complete advanced features, integrate transaction proof functionality, and ensure the app is ready for deployment.

Day 15-16: Transaction Proof Upload

- Add functionality to upload transaction proof (photo or TID) after placing an order.
    - Save the proof in the database linked to the order.
- Display a confirmation message upon successful submission.

Day 17-18: Order Management (Admin)

- Implement an Admin page to view all orders.
    - Display customer details (if logged in), product details, quantity, and status.
- Add functionality to confirm or reject orders.
- Update the database with the status of each order.

Day 19-20: UI/UX Enhancements

- Enhance the UI for all pages to ensure a consistent design.
- Add loading indicators and error handling for database operations.

- Make the app responsive for different device sizes.

Day 21: Final Testing and Deployment

- Perform end-to-end testing for both Admin and Customer workflows.
- Fix bugs and optimize app performance.
- Prepare the app for submission:
    - Add a README with usage instructions.
    - Push the final code to GitHub.