# Mobile App Prototype

# Preliminary Architectural Design (UML)

## Component + Use Case UML Diagram

App modules/components:

* UI Layer: User login, home, list creation, and list viewing screens
* Business Logic Layer: Handles saving, editing, deleting lists
* Data Layer: Local storage or SQLite DB
* Sync Layer (Optional): For cloud backup/sync if required later

How users interact with the app:

* Create List
* Add/Edit/Delete Item
* Save List
* View Lists
* Mark Item as Purchased

## Architecture Overview

|  |  |
| --- | --- |
| Component | Description |
| UI Layer | User login, home, list creation, and list viewing screens |
| Business Logic Layer | Handles saving, editing, deleting lists |
| Data Layer | Local storage or SQLite DB |
| Sync Layer (Optional) | For cloud backup/sync if required later |

## Use Case Diagram (Main Use Cases)

A diagram of a user

AI-generated content may be incorrect.

* Create List
* Add/Edit/Delete Item
* Save List
* View Lists
* Mark Item as Purchased

# Paper Prototype (Key Screen Sketches)

We can draw or describe these 6 screens:

* Welcome Screen – App name, option to "Start"
* Home Screen – View saved lists, "Add New List"
* New List Screen – Input for title, add item field, save button
* List Detail Screen – View items, mark as purchased, delete item
* Edit Item Screen – Change item name or quantity
* Settings Screen – Optional: dark mode, sync, clear all

# Python Script (Print Flow & Screen Info)

class ShoppingListPrototype:

    def \_\_init\_\_(self):

        self.pages = [

            "Welcome Screen",

            "Home Screen",

            "New List Screen",

            "List Detail Screen",

            "Edit Item Screen",

            "Settings Screen"

        ]

        self.flow = {

            "Welcome Screen": ["Home Screen"],

            "Home Screen": ["New List Screen", "List Detail Screen", "Settings Screen"],

            "New List Screen": ["List Detail Screen", "Home Screen"],

            "List Detail Screen": ["Edit Item Screen", "Home Screen"],

            "Edit Item Screen": ["List Detail Screen"],

            "Settings Screen": ["Home Screen"]

        }

    def display\_summary(self):

        print("Shopping List App Prototype Summary\n")

        print(f"Total Pages: {len(self.pages)}")

        print("Pages:")

        for page in self.pages:

            print(f"  - {page}")

        print("\nPage Navigation Flow:")

        for page, next\_pages in self.flow.items():

            print(f"  {page} ➝ {', '.join(next\_pages)}")

if \_\_name\_\_ == "\_\_main\_\_":

    prototype = ShoppingListPrototype()

    prototype.display\_summary()

# Execution

A screenshot of a computer

AI-generated content may be incorrect.

# References

Pressman, R. S., & Maxim, B. R. (2020). Software Engineering: A Practitioner’s Approach. McGraw-Hill Education.

Sommerville, I. (2016). Software Engineering (10th ed.). Pearson.