PROJECT REPORTFORMAT

**1. INTRODUCTION :**

**1.1 Project Overview**

* A brief summary of what your project is about (1–2 paragraphs).
* Mention the domain, user group, and what the project aims to solve.

**1.2 Purpose**

* Define the goal of the project.
* Why was it initiated? What needs or gaps does it fulfill?

**2. IDEATION PHASE :**

**2.1 Problem Statement**

* Clearly define the problem you're addressing.
* Use data or anecdotes if relevant to support its importance.

**2.2 Empathy Map Canvas**

* Describe the **user persona**: What do they *say, do, think, feel*?
* Helps in understanding the user’s needs and pain points.

**2.3 Brainstorming**

* List ideas generated.
* Show how they were evaluated or shortlisted.

**3. REQUIREMENT ANALYSIS :**

**3.1 Customer Journey Map**

* Map the steps a user takes to interact with the solution.
* Highlight moments of friction and opportunities.

**3.2 Solution Requirement**

* List functional and non-functional requirements.
* What must the system do? What quality attributes should it have?

**3.3 Data Flow Diagram**

* Graphically represent how data moves within your system.
* Include Level 0 and Level 1 diagrams if possible.

**3.4 Technology Stack**

* Specify technologies used: frontend, backend, database, APIs, etc.

**4. PROJECT DESIGN :**

**4.1 Problem Solution Fit**

* Show how your solution addresses the defined problem.
* Validate it with research or feedback.

**4.2 Proposed Solution**

* Describe the final solution in detail.
* Include user interface, key features, and workflows.

**4.3 Solution Architecture**

* Present the system architecture.
* Use diagrams (e.g., layered, MVC, client-server).

**5. PROJECT PLANNING & SCHEDULING :**

**5.1 Project Planning**

* Timelines, deliverables, resource allocation.
* Use Gantt charts or tables if applicable.

**6. FUNCTIONAL AND PERFORMANCE TESTING :**

**6.1 Performance Testing**

* Describe how the application was tested.
* Include response time, load testing, error handling, etc.

**7. RESULTS :**

**7.1 Output Screenshots**

* Add actual UI or CLI screenshots.
* Describe what each screenshot demonstrates.

**8. ADVANTAGES & DISADVANTAGES :**

* Bullet-point pros and cons of your solution.
* Be honest—acknowledging drawbacks is a strength.

**9. CONCLUSION :**

* Summarize what was achieved.
* Reaffirm the impact of the solution.

**10. FUTURE SCOPE :**

* Suggest features for future enhancement.
* Mention scalability or broader applications.

**11. APPENDIX :**

**Source Code (if any)**

* Attach or link to relevant source code files.
* Can also include additional diagrams or tables

**Data Set Link : https://www.kaggle.com/datasets/muhammad0subhan/fruit-and-vegetable-disease-healthy-vs-rotten**

**Project Demo Link** : [**https://drive.google.com/file/d/1twJATBtbTxx\_kZld16PzJV6y-SG3Vx9C/view?usp=drivesdk**](https://drive.google.com/file/d/1twJATBtbTxx_kZld16PzJV6y-SG3Vx9C/view?usp=drivesdk)

Git Hub Link : **https://github.com/MURTHI-KEERTHIKA/Smart-Sorting-Transfer-Learning-for-Identifying-Rotten-Fruits-and-Vegetables**