

Project Documentation Format (Including Data Structure Implementation)

1. Acknowledgment

A short paragraph thanking the guide, institution, and team members for their guidance and support throughout the project.

2. Introduction to the Project / Problem Definition

Clearly state the problem your project solves, why it is useful, and who the target users are. Mention technologies used (HTML, CSS, JavaScript, Python, MySQL, etc.).

3. Project Proposal & Plan (with Timeline)

Provide a summary of your project idea, objectives, and goals. Include a simple timeline or Gantt chart.

Example Table:

Phase	Task	Duration
Phase 1	Research & Requirement Gathering	Week 1
Phase 2	Database Design	Week 2
Phase 3	Front-end Design	Week 3
Phase 4	Back-end Integration	Week 4
Phase 5	Testing & Documentation	Week 5

4. Description of Project

Describe how your project works, list all modules/features (e.g., Login, Registration, Dashboard, Reports), and explain how front-end and back-end interact.

5. Implementation of Data Structure Concept

Explain which **data structure** concept has been implemented in your project (e.g., Stack, Queue, Linked List, Tree, Graph, Hashing, etc.).

Describe:

- The purpose of using it in your project
- How it is applied in your code (with logic or flow)
- Benefits of using that structure over simple lists or arrays

You may also include a short Python code snippet or pseudocode demonstrating its implementation.

6. ER Diagram

Draw a simple Entity-Relationship Diagram showing tables and relationships.

Example: 'User' table linked to 'Orders' table using user_id.

7. Data Dictionary

Create a table describing each database table.

Example:

Table Name: users

Column Name | Data Type | Description

-----|-----|-----

user_id | INT (PK) | Unique ID for user

username | VARCHAR(50) | User's login name

email | VARCHAR(100) | User's email address

8. UML Diagrams (Basic Level)

Include basic UML diagrams:

1. Use Case Diagram – shows system functions and user roles.
2. Activity Diagram – shows workflow or process steps.

9. Screenshots / Layouts / Models

Add labeled screenshots of all major pages or modules (e.g., Figure 1: Login Page).

10. Observations and Learning

Describe what you learned technically (HTML, Python-DB connection, debugging, applying data structures) and personally (time management, teamwork, etc.).

11. Challenges and Constraints

Mention difficulties faced during the project, such as debugging issues, time limitations, or integration problems.

12. Review of Plan vs Outcome

Compare what was planned and what was actually achieved.

Example Table:

Planned Activity | Actual Outcome

-----|-----

Implement login & registration | Completed

Add analytics dashboard | Partially completed

Integrate data structure logic | Completed

13. Conclusion and Future Scope

Summarize your project's outcome and suggest improvements or future enhancements (e.g., optimizing data structure, adding mobile version, cloud hosting).

Optional (Advanced Additions)

If applicable, include:

- Class Diagram (object-oriented design)
- Deployment Diagram (if deployed on cloud)
- API Documentation (for Flask/Django projects)