IMPLEMENTATION DOCUMENT – SE Deliverable

Project Title: Pet Authentication and Welfare System(PAWS)

Team ID : PES1UG22CS265_PES1UG22CS277

Team Members:

1. K R Rakshith (SRN : PES1UG22CS265)
2. Karthik H Kademani (SRN : PES1UG22CS277)

Date: 03/10/2024

1. Work Breakdown Structure (WBS)

ID	Task	Description	Duration (Days)	Dependencies
1	Requirements Gathering	Gather detailed requirements and final approval	5	None
2	ER Diagram and Relational Schema Design	Design the database entities and relationships	4	Task 1
3	Frontend Wireframing	Create wireframes for the user interface	3	Task 1
4	Backend API Design	Design and plan the REST API endpoints	5	Task 2
5	Frontend Development	Implement frontend components for the web app	10	Task 3
6	Backend Development	Build and implement the REST API	12	Task 4
7	Database Setup	Create and configure the database schema in MySQL	6	Task 2
8	API Testing	Test all backend API endpoints using Postman	4	Task 6
9	Frontend-Backend Integration	Connect the frontend with the backend API	7	Task 5, Task 6
10	User Acceptance Testing	End-to-end testing with real users	5	Task 9
11	Final Deployment	Deploy the system to AWS or Heroku	3	Task 10

2. Effort Estimation and Gantt Chart

Effort Estimation (Person Months)

Based on the above WBS, we estimate the following effort for each task in **person-months**.

Task	**Effort (Person Months)**
Requirements Gathering	0.25
ER Diagram and Schema Design	0.20
Frontend Wireframing	0.15
Backend API Design	0.30
Frontend Development	0.50
Backend Development	0.60
Database Setup	0.20
API Testing	0.10
Frontend-Backend Integration	0.30
User Acceptance Testing	0.15
Final Deployment	0.10
Total Effort	**2.85**

Gantt Chart

A Gantt chart can be created using tools like **GanttProject** or **Excel**. The chart should reflect the timeline for each task, its duration, and dependencies, based on the WBS. Each task will be represented on the timeline, showing when it begins and ends.

---

3. Coding Details

Technology Stack:

2. **Backend:** Streamlit(Python)
3. **Frontend:** Streamlit(Python)
4. **Database:** MySQL or PostgreSQL
5. **Version Control:** Git (via GitHub)

Code Modules:

1. **User Authentication Module:**
    - Handles user login, registration, and role-based access control (pet owners, vets, admins).
    - Uses JWT (JSON Web Tokens) for secure session management.
2. **Pet Management Module:**
    - Handles pet registration, editing, and viewing.
    - Links pets to owners via Owner_ID.
3. **Medical Records and Vaccination Tracking Module:**
    - Allows veterinarians to add, update, and view medical records.
    - Tracks vaccination schedules.

Folder Structure:

- **Frontend/Client:**
    - Components
    - Pages
    - Styles
    - Utils
- **Backend/Server:**
    - Routes
    - Models
    - Controllers
    - Middleware
- **Database:**
    - Schema.sql
    - Seed Data

Coding Standards:

- **Naming Conventions:** CamelCase for variables and functions, PascalCase for classes.
- **Commenting:** Each function should be well-commented to explain its purpose.
- **Error Handling:** Use try-catch blocks in the backend to manage errors and provide meaningful error messages to users.
- **Version Control:** Use Git for version control. All commits must be descriptive, and pull requests should be reviewed by a team member before merging.