Project Brief | Virtual Pet Manager (VPM)

# Project Overview

The Virtual Pet Manager (VPM) is a software application designed to provide a virtual pet ownership experience for users who may not be able to care for real pets due to allergies, space constraints, busy lifestyles, or other limitations. By offering a customizable and interactive experience, VPM allows users to engage in various pet care activities, such as feeding, playing, grooming, and walking their virtual pets. The project aligns with the theme of "Small Audience, Real Impact," addressing the specific needs of niche user groups who seek the companionship and responsibility of pet ownership without the associated physical or time constraints.

# Target Audience

The VPM is designed for the following small audience groups:

* **Individuals with Allergies or Space Constraints:** People who cannot have real pets due to physical or health limitations.
* **Busy Professionals or Students:** Individuals with demanding schedules who need a flexible way to engage with pets.
* **Fans of Virtual Pet Games:** Enthusiasts of virtual pet simulations looking for innovative and engaging features.
* **Prospective Pet Owners:** Individuals considering adopting a real pet and wanting to understand the responsibilities involved.
* **Educational Institutions and Parents:** Schools or parents looking for educational tools to teach children about responsibility and pet care.

# Requirements

## Functional Requirements

1. **Graphical User Interface (GUI):**
   1. The project must include a graphical user interface (GUI) for the main functionalities of the application.
   2. The GUI must be implemented using **JavaFX**.
2. **Authentication System:**
   1. The project must feature an authentication system that includes:
      1. **Sign-Up/Sign-In** functionality.
      2. **Password recovery** options.
   2. The authentication system must include both the GUI components and the underlying models to manage user accounts securely.
3. **Data Persistence System:**
   1. The project must have a persistency system to **store, retrieve, and update** user data.
   2. This system should be integrated with both the GUI and the models to ensure data consistency and reliability.
4. **Application Windows:**
   1. The project must include one or more application windows where the core functionalities of the application are performed.
   2. These windows must be integrated with the GUI and models to facilitate user interaction with the application.

## Software Requirements

1. **Java:**
   1. The project must be developed using **Java**. This is non-negotiable, as the focus of the course is on learning software development skills in Java, not web design or development.
2. **JavaFX:**
   1. For the GUI, **JavaFX** must be used to create and manage the interface windows and user interactions.

## Specifications

1. **Small Audience, Real Impact:**
   1. The project must address a specific issue faced by a small or niche audience, rather than a broad or large-scale problem.
   2. The audience should require further qualification beyond just "users" and could be defined by geographical, cultural, linguistic, educational, social, or other specific backgrounds.
2. **Focus on Object-Oriented (OO) Software Development:**
   1. The project must demonstrate the principles of object-oriented software development. This includes proper use of classes, inheritance, encapsulation, and polymorphism.
3. **Practical, Realistic Scope:**
   1. While the project should be engaging and challenging enough to maintain interest over the semester, it should remain realistic and manageable within the timeframe.
   2. The project should not aim to be a groundbreaking new software idea but rather focus on solving a specific, real-world problem in a simple and effective way.
4. **Non-Web-Based Application:**
   1. Although many software ideas could be implemented as websites or web apps, this project must be a desktop application developed in Java, with no reliance on web technologies.

# Functionality

## Core Functionality

Below is a list of core functionalities in accordance with the guidelines given:

1. **Authentication:**
   1. **Sign up/sign in:** User’s may make an account or sign in to an existing one.
2. **Pet Creation:** 
   1. **Selection:** Users may select a virtual pet from a discrete list of options (e.g. rock, cat, dog)
   2. **Customization:** Users may personalize their pet (e.g. name, colour).
3. **Basic Interactions:** 
   1. **Feed:** Users select appropriate food for their pet(s). This improves the pet(s) mood and health
   2. **Wash:** Users may wash their pet(s). This improves the pet(s) mood and health.
4. **Mood gauge:**
   1. Treatment of the pet(s) will alter the pet(s) mood.
5. **Pet Management:**
   1. **Add Pet:** Users may add new pets to their collection
   2. **Remove Pet:** Users may remove a pet, and its care history will be updated accordingly
   3. **Multiple Pets:** A consolidated view of all pets, allowing users to manage their virtual pet family.

## Optional Functionality

1. **Advanced Customization:**
   1. Additional options for pet customization, such as selecting specific breeds, patterns, or accessories for pets.
2. **Additional Interactions:**
   1. **Playing:** Users can play simple mini-games or choose toys that suit their pet’s preferences. This keeps the pet active and boosts its mood.
   2. **Grooming:** Additional grooming activities like brushing, clipping, or other care routines.
   3. **Walking (For Certain Pets):** Users can simulate walking their pet, which contributes to the pet’s physical health and happiness.
   4. **Toy Customization:** Users can customize toys for their pets, impacting the pet's mood and activity.
3. **Detailed Mood Feedback:**
   1. **Feedback System:** The pet provides more complex textual or visual feedback based on its mood (e.g., “I’m feeling great!” or “I’m feeling neglected. Please feed me!”).
   2. **Consequences of Mood:** Prolonged neglect can lead to a decrease in the pet’s overall well-being, possibly requiring more intensive care to return it to a positive state.
      1. Pets may run away if their negative mood is maintained.
4. **Rewards and Gamification:**
   1. **Pet Store Points (PSP):** A system where users earn points for maintaining a positive mood in their pets, which can be used to buy pet customizations, tools, or additional pets.
   2. **Mini-games**: Added minigames to certain interactions
5. **Care Management Enhancements:**
   1. **Routine Tracking:** Detailed tracking of the user’s care routine, including how often they feed, play with, groom, and walk their pet.
   2. **Reminders:** Users receive notifications or reminders for essential activities like feeding or walking their pet. For example, “Your pet is hungry! Don’t forget to feed it.”
   3. **Care History:** A log of the user’s interactions and care routines, allowing users to review and adjust their routine if necessary.
6. **Community Features:**
   1. **Pet Sharing:** Users can share their virtual pets with others within the application’s community.
   2. **Leaderboards:** Implementation of leaderboards to track users’ achievements and progress.
   3. **Community Challenges:** Optional challenges where users can participate in group activities, fostering a sense of community.
7. **Education Mode:**
   1. **Educational Content:** Provides users with facts and information about real pet ownership.
   2. **Quizzes:** Interactive quizzes related to pet care, rewarding users with pet store points (PSP) for correct answers.

# Work Breakdown

## Team Member 1: GUI Development (Part 1)

* **Responsibilities:**
  + Design and implement the overall layout using JavaFX.
  + Create the main application window and mood gauge interface.
  + Implement the basic structure for navigation and interaction elements (buttons, menus).
  + Collaborate with Team Member 4 for integrating core functionalities.
* **Estimated Workload:** 20%

## Team Member 2: Authentication System and Data Integration

* **Responsibilities:**
  + Develop the authentication system, including sign-up, sign-in, and password recovery.
  + Integrate the authentication system with the GUI.
  + Work with Team Member 3 on connecting the data persistence layer.
  + Implement security features such as encryption and account recovery.
* **Estimated Workload:** 20%

## Team Member 3: Data Persistence System

* **Responsibilities:**
  + Design and implement the data persistence system.
  + Ensure data is stored, retrieved, and updated efficiently.
  + Integrate the data persistence layer with the authentication system and core functionalities.
  + Collaborate with Team Members 2 and 4 to ensure data consistency.
* **Estimated Workload:** 20%

## Team Member 4: Core Functionality Implementation

* **Responsibilities:**
  + Implement core functionalities: pet creation, customization, and basic interactions (feed, wash).
  + Integrate core functionalities with the mood gauge and data persistence system.
  + Collaborate with Team Member 1 for GUI integration and Team Member 3 for data handling.
* **Estimated Workload:** 20%

## Team Member 5: Testing, Documentation, and Feature Enhancements

* **Responsibilities:**
  + Develop a comprehensive testing strategy, including unit and integration tests.
  + Coordinate testing efforts with other team members to ensure feature stability.
  + Document the codebase, create a user manual, and write developer guides.
  + Implement any optional or additional features based on time availability.
* **Estimated Workload:** 20%

# Rough Timeline (8 Weeks)

**Week 1: Planning and Initial Setup**

* **Tasks:**
  + Finalize project requirements and responsibilities.
  + Set up the project environment and repository.
  + Team Member 1 begins initial GUI layout design.
  + Team Members 2 and 3 start planning the authentication system and data persistence.

**Week 2: Authentication and GUI**

* **Tasks:**
  + Team Member 2 begins authentication system development.
  + Team Member 1 continues GUI development, focusing on structure and navigation.
  + Team Member 3 begins data persistence design.

**Week 3: Core Functionality and Data Integration**

* **Tasks:**
  + Team Member 4 starts implementing pet creation and customization.
  + Team Member 2 integrates authentication with the GUI.
  + Team Member 3 starts implementing data persistence and begins integration with core functionalities.

**Week 4: Core Functionality Implementation**

* **Tasks:**
  + Team Member 4 continues with core functionality (feeding, washing).
  + Team Member 3 ensures core functionalities are correctly integrated with the data system.
  + Team Member 1 refines the GUI based on new core functionalities.

**Week 5: Advanced Integration and Testing**

* **Tasks:**
  + Team Member 4 implements additional core functionalities (mood gauge interaction).
  + Team Member 5 starts testing early features and continues writing documentation.
  + Team Member 3 continues data integration and ensures consistency.

**Week 6: Final Feature Development and Testing**

* **Tasks:**
  + All team members collaborate on finalizing features and ensuring smooth integration.
  + Team Member 5 intensifies testing efforts, focusing on identifying and fixing bugs.
  + Team Member 1 works on final GUI polish and user experience enhancements.

**Week 7: Final Adjustments and Documentation**

* **Tasks:**
  + Implement any final features or enhancements.
  + Team Member 5 finalizes documentation and ensures all project components are well-documented.
  + All team members work on final bug fixing and optimization.

**Week 8: Final Testing, Review, and Submission**

* **Tasks:**
  + Perform final testing and debugging.
  + Review and finalize documentation and user manuals.
  + Prepare the project for submission, including packaging and final checks.