PetForU App — Enhanced Proposal

Overview

PetForU is a Java-based console application designed to help users find their ideal pet based on: – MBTI personality type – Lifestyle preferences (energy level, space, allergies, time, budget) – Pet species and breed preferences

The application uses a rule-based scoring system to suggest a match from a curated pet database. This tool aims to simulate a smart recommendation system for first-time or uncertain pet adopters.

Architecture (MVC Pattern)

Layer	Components
Model	`Pet`, `UserProfile`, `PetType`, `MBTI`, `PetMatcher`, `Matcher` (interface)
View	`ConsoleView` for all console interactions
Controller	`PetMatchController` to coordinate user input, matching logic, and output
Data	`PetDatabase` with sample pets stored in memory

Key Features

1. Personality & Lifestyle Questionnaire

- Users enter their MBTI type (e.g., INFP, ESTJ)
- Lifestyle factors: energy level, allergies, available time, living space, and budget

2. Rule-Based Matching Engine

- Assigns a compatibility score to each pet using:
  - MBTI compatibility logic
  - Lifestyle fit (energy, budget, space)
  - Allergy safety

- Breed or species match

3. Modular Matching Logic

- `Matcher` interface allows for easily adding new match algorithms
- `PetMatcher` implements current scoring strategy

4. Clean Console Output

- Displays the top-matched pet with:
  - Name, species, breed
  - Compatibility notes
  - Reason for the match

Suggested Enhancements

These could be included as stretch goals or future improvements: – Save/load user profiles to/from files – Add a GUI using JavaFX – Import pet data from external JSON or XML files – Display top 3 matches instead of just one – Let users rate their match for future improvement

---

Team Responsibilities

Member Name	Responsibilities
A	User input and profile builder: implement MBTI + lifestyle questionnaire, validate responses, construct `UserProfile`
B	Pet data and model: design and populate `PetDatabase`, define `Pet` and `PetType`, build pet attributes
C	Matching logic: create the `Matcher` interface and implement the `PetMatcher` class with scoring system
D	Console output and integration: implement `ConsoleView`, integrate controller flow, handle result display, perform system testing

Each team member is also responsible for: – Writing unit tests for their module – Commenting code and generating JavaDocs – Participating in the final demo and contributing to documentation

---

Project Timeline

Week 1

- Finalize the app's core features and system design using the MVC pattern.
- Design key data model classes: `Pet`, `UserProfile`, `PetType`, `PetDatabase`.
- Draft the MBTI and lifestyle questionnaire structure and flow.
- Set up the GitHub repository and establish the project folder structure.
- Define the rule-based matching strategy and scoring rubric.
- Create a shared task board to organize development responsibilities.

Week 2

- Implement user input handling and construct the `UserProfile` based on questionnaire responses.
- Populate the `PetDatabase` with diverse sample pet entries.
- Develop the `Matcher` interface and begin implementing the scoring logic in `PetMatcher`.
- Build the initial version of `ConsoleView` to simulate user interaction with placeholder outputs.

Week 3

- Finalize and refine the matching algorithm and scoring system.
- Integrate all components using the `PetMatchController` to connect input, matching, and output.
- Enhance the console output formatting for clear and user-friendly display of match results.
- Begin writing unit tests for each module (model, matcher, input/output).

Week 4

- Conduct full application integration testing and fix any bugs.
- Complete all unit tests and verify accuracy of matches.
- Write and finalize documentation including README, JavaDocs, and user instructions.
- Rehearse and deliver the final live demo presentation.
- Submit the complete codebase, documentation, and retrospective.

How to Run the Application

Prerequisites

- Java JDK 11 or later

- Any IDE (e.g., IntelliJ IDEA, VS Code) or command line terminal

Compile and Run

```bash javac model/*.java view/*.java controller/*.java data/*.java PetForUApp.java java PetForUApp
```