

Florida International University
Knight Foundation School of Computing and Information Sciences

Software Engineering Focus

Final Deliverable

Project Title: Pet Guardian

Team Members: Jose Hermida, Matthew Larrosa, Alex Narvaez, Valerie Hernandez, Diego Jones

Product Owner(s): Jose Hermida

Instructor: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) 2016 Florida International University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

The PetGuardian application is a comprehensive digital platform designed to assist pet owners in managing their pets' health and well-being. This application enables users to efficiently track critical health information, such as vaccination schedules, bloodwork results, and medication routines, through an intuitive and user-friendly interface. Additionally, the app includes a location-based veterinarian finder, allowing users to input their zip code and receive a curated list of nearby veterinary services. To further support pet owners, the application features an educational article section that provides best practices and tips for maintaining optimal pet health.

Built with a modern tech stack that includes ReactJS, NodeJS, SQLite, HTML, CSS, and JavaScript, the application is both responsive and scalable. The frontend provides a seamless user experience, while the backend ensures reliable data management and processing. Rigorous testing, including device compatibility checks, API validation with Postman, and regression testing after each sprint, ensured a robust and stable platform.

This innovative solution addresses the common challenges pet owners face in organizing health data, accessing reliable veterinary care, and obtaining accurate information about pet wellness. By integrating these features into a single platform, the application empowers pet owners to take a proactive approach to their pets' health, enhancing convenience, accessibility, and peace of mind.

Table of Contents

INTRODUCTION.....	5
CURRENT SYSTEM.....	5
PURPOSE OF NEW SYSTEM.....	5
 USER STORIES	
IMPLEMENTED USER STORIES.....	6
PENDING USER STORIES.....	6
 PROJECT PLAN	
HARDWARE AND SOFTWARE RESOURCES.....	7
SPRINTS PLAN.....	8
<i>Sprint 1</i>	8
<i>Sprint 2</i>	9
<i>Sprint 3</i>	10
<i>Sprint 4</i>	11
<i>Sprint 5</i>	12
<i>Sprint 6</i>	13,14
<i>Sprint 7</i>	14,15
 SYSTEM DESIGN	
ARCHITECTURAL PATTERNS.....	15
SYSTEM AND SUBSYSTEM DECOMPOSITION.....	16,17
DEPLOYMENT DIAGRAM.....	17
DESIGN PATTERNS.....	18
 SYSTEM VALIDATION.....	18,19
 GLOSSARY.....	19
 APPENDIX.....	20
APPENDIX A - UML DIAGRAMS.....	20
STATIC UML DIAGRAMS.....	20
DYNAMIC UML DIAGRAMS.....	20
APPENDIX B - USER INTERFACE DESIGN.....	21
APPENDIX C - SPRINT REVIEW REPORTS.....	21,22,23,24,25,26
APPENDIX D - INSTALLATION/MAINTENANCE DOCUMENT, SHORTCOMINGS/WISHLIST DOCUMENT AND OTHER DOCUMENTS.....	26,27
 REFERENCES.....	27

INTRODUCTION

Current System

Pet owners nowadays confront considerable obstacles in maintaining their dogs' health and well-being since there is no comprehensive, all-in-one solution. Existing technologies are frequently fragmented, forcing users to switch between applications or rely on manual techniques like paper-based health records, generic reminders, or dispersed web searches for veterinary treatment. These constraints make it difficult to keep track of vital health data such as immunization schedules, bloodwork results, and prescription regimens properly. Furthermore, locating reputable veterinarian services and obtaining accurate, trustworthy information on pet health best practices is time-consuming and frequently difficult. This disconnected approach can result in gaps in pet care, compromising both the health of the dogs and the peace of mind of their owners.

Purpose of New System

The PetGuardian app aims to transform pet health management by eliminating the inadequacies of current options. This complete digital platform offers pet owners an easy-to-use interface for tracking and organizing critical health information including vaccination schedules, bloodwork results, and medication regimes. PetGuardian also contains novel features including a location-based doctor locator that uses zip code inputs to generate a tailored list of nearby veterinary providers. To further assist pet owners, the site includes an educational article section including professional insights and best practices for maintaining optimal pet health.

PetGuardian offers a scalable, dependable, and user-friendly experience by integrating a current tech stack, which includes ReactJS, NodeJS, SQLite, HTML, CSS, and JavaScript. The technology not only streamlines pet health management, but it also allows pet owners to take an active role in their dogs' well-being, improving convenience, accessibility, and overall quality of care.

User Stories

The following section provides the detailed user stories that were implemented in this iteration of the PetGuardian project. These user stories served as the basis for the implementation of the project's features. This section also shows the user stories that are to be considered for future development.

Implemented User Stories

1. As a pet owner, I want to be able to add and manage pet records, keeping track of things like vaccines, bloodwork results, and medications.
 - Feature: CRUD operations for pet health records
2. As a pet owner, I want to be able to search for veterinarians near a specific location by entering a zip-code, making it easy to find veterinary care.
 - Feature: Location-based veterinarian finder using the Google Maps APIe routine.
3. As a user, I want a secure login and account management system, so that my data is private and protected.
 - Feature: User login subsystem/database

Pending User Stories (pending push)

1. As a pet owner, I want to access articles and tips on pet health and wellness, so that I can improve my pet's care routine.
 - Feature: Educational subsystem with curated content like articles

Project Plan

The development of the PetGuardian application was meticulously planned and executed using agile development techniques. The project was divided into two-week sprints, each structured to ensure consistent progress and timely delivery of features. Daily standup meetings played a pivotal role in maintaining team alignment, where we discussed the progress made, identified challenges, and planned next steps. These meetings also fostered collaboration, enabling the team to work together to resolve any issues that arose during development.

During each sprint, we conducted backlog meetings to evaluate the tasks in the backlog and prioritize items for the upcoming sprint. This process ensured that the most critical features were addressed first and allowed us to adapt to changing requirements or feedback effectively. Regular communication between team members occurred daily through various messaging apps, beyond formal meetings, to monitor progress and address any blockers promptly, ensuring the project stayed on schedule. All team members were engaged and provided insight to progress the PetGuardian application.

Hardware and Software Resources

The project relied on a combination of software and hardware components. On the software side, the application was built using ReactJS for the frontend, providing a dynamic and user-friendly interface, and NodeJS for the backend, ensuring efficient server-side processing. SQLite was chosen for the database due to its lightweight yet reliable capabilities, storing and managing critical health data like vaccination schedules and bloodwork records. HTML and CSS were utilized for structuring and styling the user interface, while JavaScript served as the primary programming language for both frontend and backend logic. The hardware requirements were minimal, relying on standard development machines and mobile devices for testing.

By adhering to this agile framework and leveraging effective communication and collaboration, we successfully developed a comprehensive pet wellness application that met most project goals and delivery deadlines.

Sprints Plan

Sprint 1

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. Understand user requirements for Pet Guardian
- User Story – 2. Design front screens for Pet Guardian Login and Sign up.
- User Story – 3. As a new user, I want to be able to register an account using my email and a password, so that I can securely access the app.
- User Story – 4. As a registered user, I want to be able to log in to my account, so that I can access my pet's information.

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. Understand user requirements for Pet Guardian [1 Story Point]
 - User Story – 3. As a new user, I want to be able to register an account using my email and a password, so that I can securely access the app.
 - User Story – 4. As a registered user, I want to be able to log in to my account, so that I can access my pet's information.
- Jose Manuel Hermida-Larios
 - User Story – 1. Understand user requirements for Pet Guardian [1 Story Point]
 - User Story – 2. Design front screens for Pet Guardian Login and Sign up [4 Story Points]
- Valerie Hernandez
 - User Story – 1. Understand user requirements for Pet Guardian [1 Story Point]
 - User Story – 2. Design front screens for Pet Guardian Login and Sign up [4 Story Points]
- Diego Jones
 - User Story – 1. Understand user requirements for Pet Guardian [1 Story Point]
 - User Story – 2. Design front screens for Pet Guardian Login and Sign up [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. Understand user requirements for Pet Guardian [1 Story Point]
 - User Story – 3. As a new user, I want to be able to register an account using my email and a password, so that I can securely access the app.
 - User Story – 4. As a registered user, I want to be able to log in to my account, so that I can access my pet's information.

Sprint 2

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. As a pet owner, I want to create a profile for my pet with basic details (name, age, breed, and photo) [4 Story Points]
- User Story – 2. As a pet owner, I want to easily navigate the home screen with clear buttons or icons [4 Story Points]
- User Story – 3. As a pet owner, I want to add a vet appointment to my calendar, so that I can stay organized and track upcoming health events for my pet. [4 Story Points]

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. As a pet owner, I want to create a profile for my pet with basic details (name, age, breed, and photo) [4 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 2. As a pet owner, I want to easily navigate the home screen with clear buttons or icons [4 Story Points]
- Valerie Hernandez
 - User Story – 3. As a pet owner, I want to add a vet appointment to my calendar, so that I can stay organized and track upcoming health events for my pet. [4 Story Points]
- Diego Jones
 - User Story – 3. As a pet owner, I want to add a vet appointment to my calendar, so that I can stay organized and track upcoming health events for my pet. [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. As a pet owner, I want to create a profile for my pet with basic details (name, age, breed, and photo) [4 Story Points]

Sprint 3

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. As a user once an appointment is created I would like to be able to view the appointment, having it as a reminder, and also if needed I would like to be able to edit appointment time, change the date, or cancel if need be. [4 Story Points]
- User Story – 2. As a user I would like to access any prescriptions that may be correlated to my pet. [4 Story Points]

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. As a user once an appointment is created I would like to be able to view the appointment, having it as a reminder, and also if needed I would like to be able to edit appointment time, change the date, or cancel if need be. [4 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 2. As a user I would like to access any prescriptions that may be correlated to my pet. [4 Story Points]
- Valerie Hernandez
 - User Story – 2. As a user I would like to access any prescriptions that may be correlated to my pet. [4 Story Points]
- Diego Jones
 - User Story – 2. As a user I would like to access any prescriptions that may be correlated to my pet. [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. As a user once an appointment is created I would like to be able to view the appointment, having it as a reminder, and also if needed I would like to be able to edit appointment time, change the date, or cancel if need be. [4 Story Points]

Sprint 4

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. A login page is provided with fields for email and password. The page includes a "Login" button and a "Forgot Password" link. [4 Story Points]
- User Story – 2. User credentials (email and password) are validated against a SQL database. Upon successful validation, the user is redirected to their dashboard. If the credentials are incorrect, an error message ("Invalid email or password") is displayed. [4 Story Points]
- User Story – 3. Passwords are securely hashed and stored in the SQL database. Input validation prevents SQL injection or other vulnerabilities. [4 Story Points]
- User Story – 4. A "Forgot Password" link redirects users to a password reset page. [1 Story Points]

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. A login page is provided with fields for email and password. The page includes a "Login" button and a "Forgot Password" link. [4 Story Points]
 - User Story – 4. A "Forgot Password" link redirects users to a password reset page. [1 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 2. User credentials (email and password) are validated against a SQL database. Upon successful validation, the user is redirected to their dashboard. If the credentials are incorrect, an error message ("Invalid email or password") is displayed. [4 Story Points]
- Valerie Hernandez
 - User Story – 2. User credentials (email and password) are validated against a SQL database. Upon successful validation, the user is redirected to their dashboard. If the credentials are incorrect, an error message ("Invalid email or password") is displayed. [4 Story Points]
 - User Story – 3. Passwords are securely hashed and stored in the SQL database. Input validation prevents SQL injection or other vulnerabilities. [4 Story Points]
- Diego Jones
 - User Story – 3. Passwords are securely hashed and stored in the SQL database. Input validation prevents SQL injection or other vulnerabilities. [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. A login page is provided with fields for email and password. The page includes a "Login" button and a "Forgot Password" link. [4 Story Points]
 - User Story – 4. A "Forgot Password" link redirects users to a password reset page. [1 Story Points]

Sprint 5

The product owner chose the following user stories to be done during the sprint.

- User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- User Story – 2. As a new or returning user, I want a login/sign-up page that is simple and secure, so I can access my account or create a new one with ease. [4 Story Points]
- User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
 - User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 2. As a new or returning user, I want a login/sign-up page that is simple and secure, so I can access my account or create a new one with ease. [4 Story Points]
- Valerie Hernandez
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- Diego Jones
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
 - User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]

Sprint 6

The product owner chose the following user stories to be done during the sprint. These stories carried over from the following sprint because of the amount of time they required.

- User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- User Story – 2. As a new or returning user, I want a login/sign-up page that is simple and secure, so I can access my account or create a new one with ease. [4 Story Points]
- User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
 - User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 2. As a new or returning user, I want a login/sign-up page that is simple and secure, so I can access my account or create a new one with ease. [4 Story Points]
- Valerie Hernandez
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- Diego Jones
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]
- Alex Narvaez-Echeverria
 - User Story – 1. As a user, I want an account dashboard that provides an overview of my account activity, so I can easily access and manage my information. [4 Story Points]

- User Story – 3. As a developer, I want a structured and optimized database configuration to store user data and support application features efficiently. [4 Story Points]

Sprint 7

The product owner chose the following user stories to be done during the sprint. These stories carried over from the following sprint because of the amount of time they required.

- User Story – 1. As a healthcare provider, I want an API to securely store and retrieve bloodwork data, so that I can easily access patient results.[4 Story Points]
- User Story – 2. Medication Routine API: As a patient, I want an API to track my medication routine, so I can get reminders and monitor my adherence [4 Story Points]
- User Story – 3. Vaccines API: As a doctor, I want an API to track patient vaccinations, so I can ensure they are up-to-date on required immunizations. [4 Story Points]
- User Story – 4. Presentation Board: As a presenter, I want a digital board to create and organize my presentation slides, so that I can easily share my content during meetings.

The team members indicated their willingness to work on the following user stories.

- Matthew Larrosa
 - User Story – 1. As a healthcare provider, I want an API to securely store and retrieve bloodwork data, so that I can easily access patient results.[4 Story Points]
 - User Story – 2. Medication Routine API: As a patient, I want an API to track my medication routine, so I can get reminders and monitor my adherence [4 Story Points]
 - User Story – 3. Vaccines API: As a doctor, I want an API to track patient vaccinations, so I can ensure they are up-to-date on required immunizations. [4 Story Points]
- Jose Manuel Hermida-Larios
 - User Story – 4. Presentation Board: As a presenter, I want a digital board to create and organize my presentation slides, so that I can easily share my content during meetings.
- Valerie Hernandez
 - User Story – 4. Presentation Board: As a presenter, I want a digital board to create and organize my presentation slides, so that I can easily share my content during meetings.
- Diego Jones

- User Story – 4. Presentation Board: As a presenter, I want a digital board to create and organize my presentation slides, so that I can easily share my content during meetings.
- Alex Narvaez-Echeverria
 - User Story – 1. As a healthcare provider, I want an API to securely store and retrieve bloodwork data, so that I can easily access patient results.[4 Story Points]
 - User Story – 2. Medication Routine API: As a patient, I want an API to track my medication routine, so I can get reminders and monitor my adherence [4 Story Points]
 - User Story – 3. Vaccines API: As a doctor, I want an API to track patient vaccinations, so I can ensure they are up-to-date on required immunizations. [4 Story Points]

SYSTEM DESIGN

The System Design section describes the key design decisions and approaches that influenced the creation of the PetGuardian application. This section gives an overview of the architectural patterns utilized, shows the system's structure using a package diagram, discusses the primary subsystems, and describes the design patterns used to create a scalable, efficient, and maintainable platform.

Architectural Patterns

The PetGuardian program is designed with a client-server architecture. This design was selected to divide the frontend and backend activities, allowing for scalability and flexibility during development and maintenance. The Model-View-Controller (MVC) design pattern is used at both ends to provide modularity and separation of concerns:

Model: Responsible for managing the main data that is kept in a SQLite database. This includes pet health information, user profiles, and educational content.

View: ReactJS, HTML, and CSS are used to implement the user interface, which guarantees a user experience that is both intuitive and responsive with the user.

Controller: Node.js is used to process user requests and perform business logic, which enables the frontend and backend to communicate with one another in a smooth manner.

System and Subsystem Decomposition

The system is decomposed into several key subsystems to streamline functionality and development:

1. Health Data Management Subsystem:

- Manages vital pet health information, such as vaccination schedules, bloodwork results, and medication regimens.
- Provides CRUD operations for health records, allowing easy data entry and retrieval.

2. Veterinarian Finder Subsystem:

- Implements a location-based search feature that allows users to find veterinarian services by entering their zip code.
- Integrates third-party geolocation APIs to deliver accurate and selected results.

3. Educational Content Subsystem:

- Provides a collection of information and resources on the best practices in pet care.
- Includes an admin module for dynamically updating and managing content.

4. User Authentication and Authorization Subsystem:

- Protects user accounts with role-based access control and encryption techniques.
- Protects data privacy and restricts unauthorized access to sensitive information.

5. Testing and Validation Subsystem:

- Include frameworks and tools for device compatibility tests, API validation, and regression testing that ensures system uniformity and stability across several platforms.

Deployment Diagram

The deployment diagram shows how the system's parts are physically spread out:

1. Client Devices

- Computers, tablets, and smartphones in which frontend application runs in a browser.
- Frontend hosted on a CDN for fast delivery.

2. Application Server

- Runs backend with NodeJS.
- Handles API requests, business logic, and interfaces with the database.

3. Database Server

- Hosts SQLite database for storing pet health records, user data, and the educational content.
- Secure and optimized

4. External API Services

- Integrates with the google maps API for geolocation and veterinarian service data.

The deployment ensures a robust and responsive system.

Design Patterns

The following design techniques were used to make the system work better and be easier to maintain:

1. Observer Pattern
 - Used for real-time updates to the user interface when health data is modified.
2. Singleton Pattern
 - Single database connection instance which is reused across the system for peak resource usage.
3. Builder Pattern
 - Facilitates the construction of complex objects, such as dynamic user dashboards or pet health summaries, with varying configurations.

SYSTEM VALIDATION

PetGuardian underwent extensive testing to ensure its reliability, functionality, and overall user experience. The validation process included comprehensive testing on desktops, tablets, and smartphones to confirm responsiveness and compatibility across a wide range of devices and screen sizes. This was essential to ensure that users could seamlessly interact with PetGuardian, regardless of the platform they chose.

To validate backend functionality and ensure smooth communication between the frontend and backend, Postman was used to rigorously test all API routes. Each endpoint was evaluated for correct responses, robust error handling, and data integrity under various scenarios. This ensured that PetGuardian could accurately process user inputs and provide reliable outputs.

After each sprint, regression testing was performed to verify that new features—such as pet health data tracking, location-based vet recommendations, and access to curated health articles—integrated smoothly without impacting existing functionality. These iterative tests ensured that the system remained stable and cohesive throughout the development lifecycle.

The team's daily communication and collaboration further strengthened the validation process, enabling quick identification and resolution of issues. By combining device testing, API

validation, and iterative regression testing, PetGuardian was thoroughly validated to meet all functional requirements, delivering a dependable and user-friendly platform for pet owners.

GLOSSARY

Navbar:

React component that organizes all the anchor tags for relevant sections in the pet guardian application such as Pet profile, Medical Section and Zip Code Vet Finder.

Google Maps API:

Interface for location-based mapping services developed to help users locate veterinarians nearby a set inputted zipcode anywhere in the world.

Bloodwork API:

Bloodwork API accepts information such as a pet's name, date when bloodwork occurred, and notes that can provide more context on the bloodwork.

Medication API:

Medication API accepts information such as a pet's name, medication name, start date and end date for prescription, and notes that can provide more context on the medication routine.

Vaccine API:

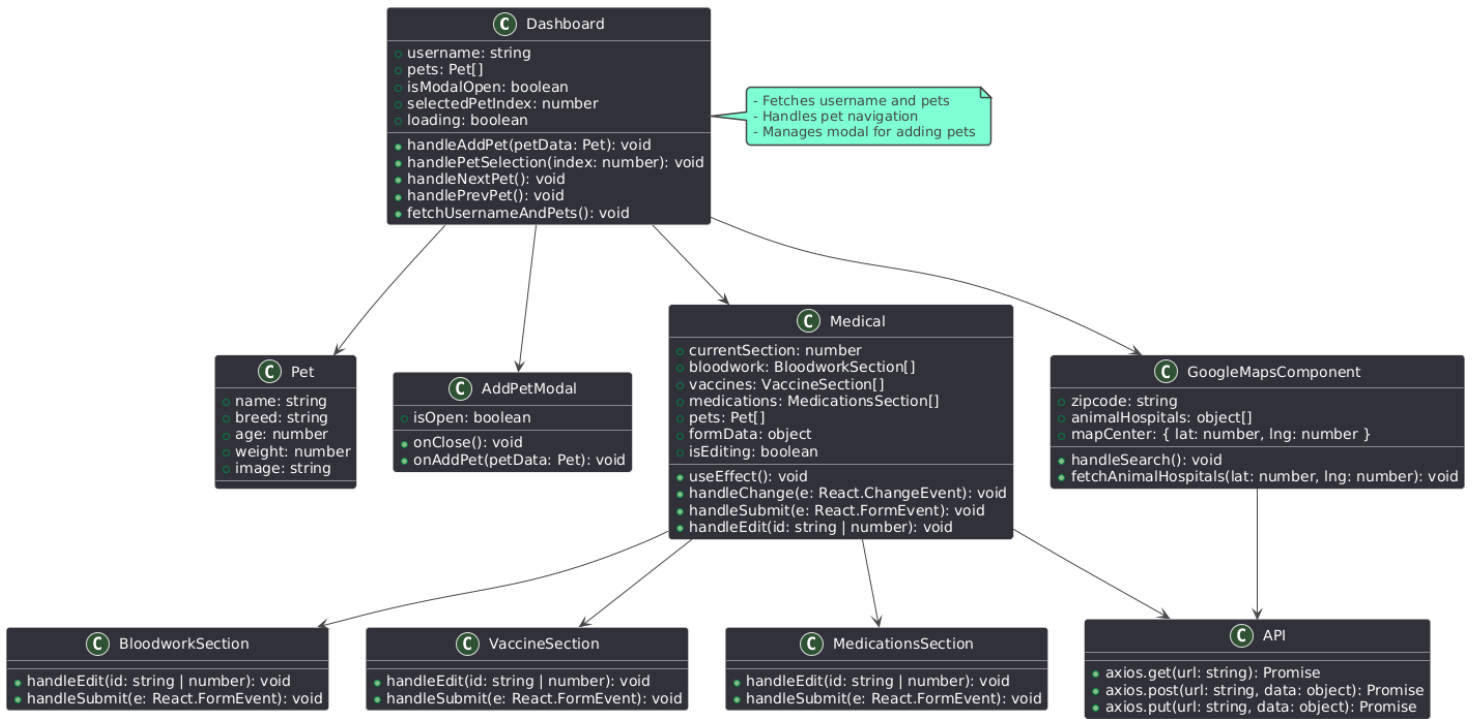
Interface for accessing vaccination related data, retrieve vaccine types, schedules, and manufacturer details and also to set reminders to track vaccine due dates.

Vet Finder:

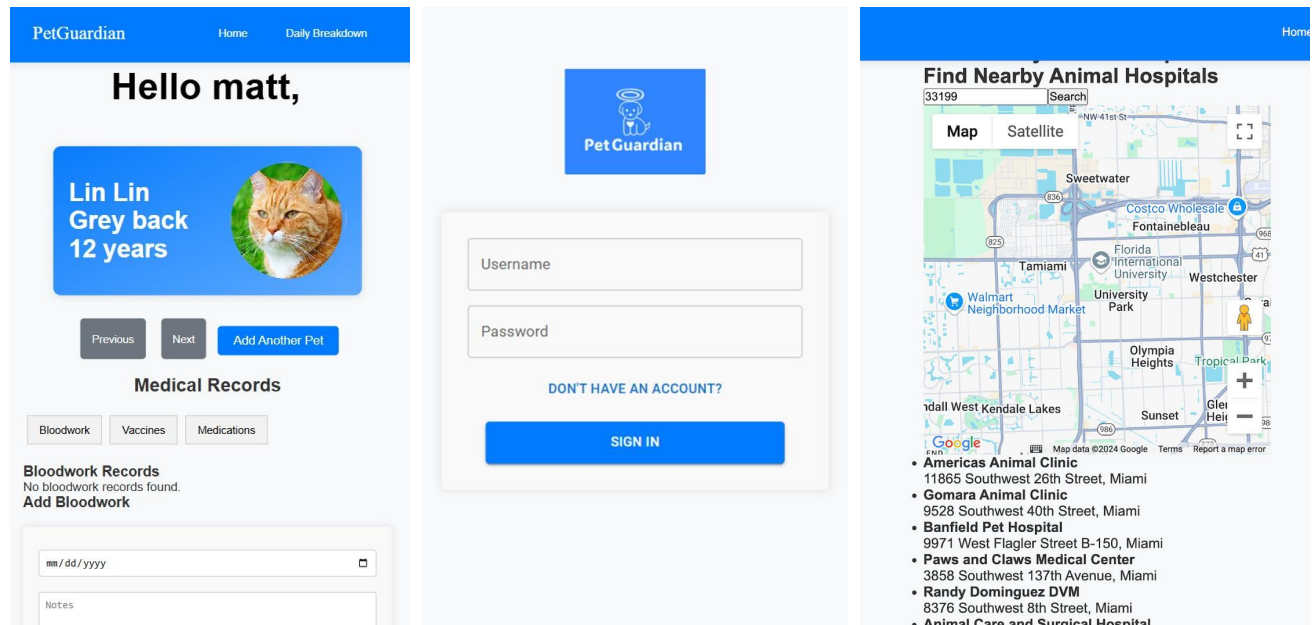
React component that utilizes Google Maps API to find Vet Office within a provided zip code. Populates a google map with markers on location as well as generates a list of those locations.

APPENDIX

Appendix A - UML Diagrams



Appendix B - User Interface Design



Appendix C - Sprint Review Reports

Sprint 1

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 10:00

End time: 10:35

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

User Story 1: Account Setup

As a new user, I want to be able to register an account using my email and a password, so that I can securely access the app.

Acceptance Criteria:

A user is able to successfully create an account on our platform using their email and setting up a secure password, after creation the profile can be viewed, edited, and signed out of.

User Story 2: Account Login

As a registered user, I want to be able to log in to my account, so that I can access my pet's information

Acceptance Criteria:

A user is able to, when signed out, login to their existing account using the email and password given during account creation, their profile should be unchanged upon sign in and just like after creation your profile should be able to be viewed, edited, and signed out of.

Sprint 2

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 10:45 PM

End time: 11:15 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

User Story 1: Pet profile setup

User Story 2: Home screen navigation

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

User Story 3: Add appointment

How this should be reflected on the user story definition in Mingle: As a pet owner, I want to add a vet appointment to my calendar, so that I can stay organized and track upcoming health events for my pet.

Acceptance Criteria:

- The user can add an appointment through a calendar view or form.
- The appointment details (vet name, date, time) are easy to input and edit.
- The user can see upcoming appointments displayed on the dashboard or calendar.

Sprint 3

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 10:20 PM

End time: 10:55 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

User Story1: Appointment Editing

- As a user once an appointment is created I would like to be able to view the appointment, having it as a reminder, and also if needed I would like to be able to edit appointment time, change the date, or cancel if need be.

User story 2: Prescription (Rx) Overview

- As a user I would like to access any prescriptions that may be correlated to my Pet.

User story 3: Prescription (Rx) Overview

- As a user I would like to access any prescriptions that may be correlated to my Pet.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

- None

Sprint 4

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 11:00 PM

End time: 11:25 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners: All.

User Story 1 : Login Page

- A login page is provided with fields for email and password.

User story 2: Forgot Password

- A "Forgot Password" link redirects users to a password reset page

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Spring Planning meeting.

User Story 3: SQL Integration

User Story 4: Security

Sprint 5

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 10:00 PM

End time: 10:30 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story 1: Pet Profile

- A section where users can input and store information about their pets, including name, breed, age, and a picture.

User Story 2: Medical Section

- A section dedicated to tracking a pet's medication and vaccine history.

User Story 3: Vet Locator

- A map feature where users can input their zip code to find nearby veterinary clinics.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

User Story 4: Pet Status Points

- A system where users can earn points for various pet-related activities, such as taking their pet for a walk. This feature has been moved to a future sprint.

Sprint 6

Sprint Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 10:30 PM

End time: 11:00 PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

User Story 1: Bloodwork API

- As a healthcare provider, I want an API to securely store and retrieve bloodwork data, so that I can easily access patient results.

User Story 2: Medication Routine API:

- As a patient, I want an API to track my medication routine, so I can get reminders and monitor my adherence.

User Story 3: Vaccines API

- As a doctor, I want an API to track patient vaccinations, so I can ensure they are up-to-date on required immunizations.

User Story 4: Presentation Board

- As a presenter, I want a digital board to create and organize my presentation slides, so that I can easily share my content during meetings.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

None

Sprint 7

Sprint 7 Review Meeting Minutes

Attendees: Jose Manuel Hermida-Larios, Valerie Hernandez, Diego Jones, Matthew Larrosa, Alex Narvaez-Echeverria

Start time: 11:00PM

End time: 11:30PM

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- Finishing final feature to be present in the final deliverable.
- Work on Final deliverable sections.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting.

None

Appendix D - Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Installation

1. Prerequisites:
 - a. Node.js (version 14 or higher)
 - b. SQLite3
2. Clone the repository in VS Code:
 - a. `git clone https://github.com/yourusername/pet-guardian.git`
 - b. `cd pet-guardian`
3. Install backend dependencies and start up backend:
 - a. Create a new terminal
 - b. type `cd backend`
 - c. type `npm install`
 - d. type `node server.js`
4. Install front dependencies and start up frontend:
 - a. Create a terminal
 - b. You should be in the pet-guardian directory
 - c. type `npm install`
 - d. type `npm run start`

Shortcomings

- Absence of checks for duplicate entries within the database, for example a user will add a new pet. After the pet creation, the user can try to add another pet with the same exact information and the database doesn't check if the pet exists already.
- Local Host, our application is a full stack application but one shortcoming is that it's hosted in local host as opposed to a web server. Our api keys are local to our machines,

but if we were trying to test the application and didn't have the API keys for example. The Vet finder feature would not work properly.

Wish List

- Acquire a domain and a web server for our application to be hosted on
- Implement checks for existing entries in the database
- Implement a feature where users can upload document from the vet to be saved on web app

REFERENCES

- Google Maps API Walkthrough:
 - <https://developers.google.com/maps/get-started>
- Google Maps Platform:
 - https://mapsplatform.google.com/?utm_source=google&utm_medium=cpc&utm_campaign=google_maps_brand_us_2&gad_source=1&gclid=Cj0KCQiAvP-6BhDyARIsAJ3uv7Yv2owqonBDPA8-5c4IpXWOFzkNk0Awxb08comAh31yvBLAV7rBdSAaAq7pEALw_wcB&gclsrc=aw.ds
- Google Cloud Platform:
 - https://cloud.google.com/gcp?utm_source=google&utm_medium=cpc&utm_campaign=na-US-all-en-dr-bkws-all-all-trial-e-dr-1707554&utm_content=text-ad-non-e-any-DEV_m-CRE_665735450627-ADGP_Hybrid+%7C+BKWS+-+EXA+%7C+Txt-Core-Google+Cloud-KWID_43700081237254435-kwd-26415313501&utm_term=KW_google%20cloud%20platform-ST_google+cloud+platform&gad_source=1&gclid=Cj0KCQiAvP-6BhDyARIsAJ3uv7bZldeerWRt00A48NnFbOkRTzSl_-tcKSh33E0SMod5otM4BFU26SwaAvYJEALw_wcB&gclsrc=aw.ds
- Platform for diagramming (Mermaid):
 - https://www.mermaidchart.com/landing?utm_source=google_ads&utm_medium=primary_search&utm_campaign=PMax1-US&gad_source=1&gbraid=0AAAAAqtlhyww9fGtDkK5rUjKm0-jlul_B&gclid=Cj0KCQiAvP-6BhDyARIsAJ3uv7Y1jACBahlksBZ8GlvZiFEFr0WwMdVml3zBwu2fg3iVL-7TpXEnOEaAnBEEALw_wcB
- VsCode:
 - <https://code.visualstudio.com/>
- SQLite:
 - <https://www.sqlite.org/>
- Node.Js:
 - <https://nodejs.org/en>