PetConnect Developer Guide

Contents

1 Introduction **2**

2 Project Structure **2**
2.1 home.py . . . 2
2.1.1 Login and Authentication . . . 2
2.1.2 Fetching and Displaying Recent Adoption Requests . . . 2
2.1.3 Marking Requests as "Seen" . . . 2
2.1.4 Interactive Menu and Refresh Option . . . 2
2.2 adopters.py . . . 2
2.2.1 File Management . . . 2
2.2.2 Form Submission and Data Entry . . . 2
2.2.3 Editable Data Table . . . 2
2.2.4 Validation and Sanitization . . . 3
2.3 dogs.py . . . 3
2.3.1 Adding New Dog Data . . . 3
2.3.2 Data Editor for Dog Records . . . 3
2.3.3 Interactive Menu and Page Layout . . . 3
2.4 Foster Homes Page . . . 3
2.4.1 File Management for Foster Homes . . . 3
2.4.2 Data Entry for New Foster Homes . . . 3
2.4.3 Interactive Menu and Navigation . . . 3
2.5 The Applications Page . . . 3
2.5.1 Viewing and Filtering Applications . . . 3
2.5.2 Scoring System for Applicants . . . 3
2.5.3 Displayed Results . . . 3

3 Connections and Backend Interactions **4**
3.1 Backend and Database Connection . . . 4
3.2 Caching and Performance Optimization . . . 4
3.3 Session Management and Access Control . . . 4

4 Data Management **4**
4.1 Adopter Data . . . 4
4.2 Dog Data . . . 4
4.3 Foster Homes . . . 4
4.4 Shopping List Items Data . . . 4

5 Dependencies **4**
5.1 Cross-Page Data Dependencies . . . 4
5.2 Data Validation and Formatting . . . 5
5.3 File Management Dependencies . . . 5

6 Security Best Practices **5**

1 Introduction

PetConnect is a web-based platform designed to facilitate the connection between adopters and dogs through a streamlined adoption process. The platform is built using Python and Streamlit for the frontend, with backend components handling database interactions, user management, and predictive models for matching dogs to adopters.

As development progresses, our team is committed to delivering a robust solution that supports PetConnect's mission by enhancing its ability to facilitate faster and more effective adoptions, ultimately making a substantial impact on the community and the lives of pets and adopters.

This guide provides a comprehensive overview of the system's architecture, components, and key functions, focusing on data processing and predictive model integration.

2 Project Structure

2.1 home.py

The `home.py` page in the PetConnect platform is designed for managing user access and tracking updates to recent adoption form submissions. The page offers a secure login feature, allowing only authorized users to access new adoption requests. Key functions include:

2.1.1 Login and Authentication

The page features a secure login form where users enter a username and password. The password is hashed using SHA-256 encryption, and the credentials are validated against a stored user database to ensure only authorized users can access the system.

2.1.2 Fetching and Displaying Recent Adoption Requests

Once logged in, users can view adoption requests submitted within the last two days. These records are fetched from Google Sheets, filtered by submission date, and displayed in a user-friendly format. Each record includes details such as the applicant's name, the dog they are interested in, additional information, and their contact number.

2.1.3 Marking Requests as "Seen"

After reviewing a submission, users can mark the request as "seen" by clicking a button next to each record. This updates the "Seen" column in Google Sheets, ensuring that requests are tracked effectively.

2.1.4 Interactive Menu and Refresh Option

Users can refresh the data displayed on the page by clicking a "Refresh" button, which clears cached data and reloads the latest submissions.

2.2 adopters.py

The `adopters.py` page allows users to view, edit, and add new adopter information to the system. Key functions include:

2.2.1 File Management

- `save_file(adopter_id, file)`: Saves uploaded files to the local `Data/Adopters/` directory.
- `delete_file(file_name)`: Deletes a specified file from the local storage.
- `load_file(file_name)`: Retrieves a file for download.

2.2.2 Form Submission and Data Entry

The New Adopter Form collects various fields such as name, address, phone number, and adoption date. The form data is then processed and added to the Google Sheet.

2.2.3 Editable Data Table

Adopter data fetched from Google Sheets is displayed in an editable table. Users can edit the data directly in the UI and save changes back to the Google Sheet.

2.2.4 Validation and Sanitization

Before updating the Google Sheet or saving new entries, NaN and infinite values are sanitized to avoid issues with Google Sheets or data storage. Adopter ID uniqueness is validated to prevent conflicts.

2.3 dogs.py

The `dogs.py` page is a user-friendly interface for managing dog-related data. Key functions include:

2.3.1 Adding New Dog Data

Users can input detailed information about a new dog, such as ID, name, breed, and various health and adoption statuses. The `add_dog_to_google_sheet()` function appends the new dog's data as a row in the Google Sheet, ensuring that all dog records are up-to-date.

2.3.2 Data Editor for Dog Records

Users can view and edit the dog data stored in the Google Sheet through an editable DataFrame rendered by `st.experimental_data_editor()`. Users can modify the records directly, and a save button is provided to update these changes in the Google Sheet.

2.3.3 Interactive Menu and Page Layout

The app provides an interactive navigation menu using `option_menu`, with options such as "Add Dog" and "View All Data". The page layout is configured with `st.set_page_config()` for a wide display.

2.4 Foster Homes Page

The `foster_home.py` page is a comprehensive interface for managing foster home data within the PetConnect application. Key functions include:

2.4.1 File Management for Foster Homes

Files uploaded by users are saved with a naming convention based on the foster home's ID. The `save_file()` function handles file uploads, while the `delete_file()` function allows users to delete files related to a foster home.

2.4.2 Data Entry for New Foster Homes

The page features a form where users can enter new foster home information, including house size, contact information, and maximum capacity.

2.4.3 Interactive Menu and Navigation

The page provides an interactive menu with options like "Add Foster Home", "Find Foster Home", and "Edit Documents". This allows users to seamlessly switch between viewing all foster homes, adding new records, or managing related documents.

2.5 The Applications Page

The `applications.py` page provides an interface for managing adoption applications. Key functions include:

2.5.1 Viewing and Filtering Applications

Users can view all adoption requests or filter them by specific criteria such as dog name, applicant name, and date.

2.5.2 Scoring System for Applicants

The `score_adopter()` function calculates a score for each applicant based on predefined criteria. The scoring system adjusts points based on factors such as dog compatibility with children and other pets, applicant's previous adoption history, presence of a garden, and other factors.

2.5.3 Displayed Results

The calculated scores are displayed in a table showing each applicant's name, the timestamp of their application, and their score.

3 Connections and Backend Interactions

The PetConnect platform's core functionality relies on the interaction between its backend and various data sources, primarily Google Sheets and local files.

3.1 Backend and Database Connection

The backend interacts with Google Sheets, which serves as the primary database for adopters, dogs, and foster homes. This interaction is managed via the `gspread` Python package.

3.2 Caching and Performance Optimization

To optimize performance and limit API requests, the platform uses Streamlit's caching mechanism (`@st.cache_data()`), which stores data fetched from Google Sheets.

3.3 Session Management and Access Control

Session management in PetConnect is handled through `st.session_state['logged_in']`, which ensures that only authenticated users can access or modify sensitive data.

4 Data Management

The platform manages several datasets, each crucial to its operations:

4.1 Adopter Data

- **File:** adopters sheet
- **Fields:** Name, contact information, preferences for dog breed, size, and age.
- **Functionality:** Used to match potential adopters with available dogs.

4.2 Dog Data

- **File:** dogs sheet
- **Fields:** Dog ID, breed, age, health status, foster home assignment.
- **Functionality:** Tracks available dogs and their adoption status.

4.3 Foster Homes

- **File:** fosterhomes sheet
- **Fields:** Foster home ID, location, capacity, dogs currently housed.
- **Functionality:** Tracks foster homes, available spaces, and current dogs.

4.4 Shopping List Items Data

- **File:** Items sheet
- **Fields:** Name, image, Size, compatibility, ID.
- **Functionality:** Used to create a shopping list uniquely for each dog based on their characteristics.

5 Dependencies

The platform has several important dependencies between pages, functions, and operations, ensuring proper data synchronization and interaction between shared components.

5.1 Cross-Page Data Dependencies

For example, the Dogs Page and Applications Page both depend on up-to-date dog data. When a dog's information is updated on the Dogs Page (such as its availability), it directly affects the Applications Page.

5.2 Data Validation and Formatting

Date fields must follow a standard format (YYYY-MM-DD) to prevent issues in functions that rely on date calculations.

5.3 File Management Dependencies

The Dogs Page manages uploaded files, such as medical records and images, which are then referenced by other sections of the platform.

6 Security Best Practices

- **Encryption:** Sensitive data (e.g., API keys) is encrypted during transmission and storage.
- **Access Control:** Only authorized users with the correct permissions can access and modify sensitive data.