Afeka – Academic College of Engineering in Tel-Aviv
Computer Science

December 22, 2024
Gal Alfandary 206396731, Lina Flat 322690181

Instructor: Dr. Sharon Yalov-Handzel.

11402: Detailed requirements -Lost & Pawnd

Introduction

The "Lost & Pawnd" project is a smart system designed to help people report and locate lost and found pets. It provides an easy-to-use and reliable platform that connects people who have lost their pets with those who have found stray animals.

By using advanced technologies, including AI-based image recognition and GPS location services, the system aims to quickly and accurately match lost pets with found reports. It focuses on data security, accessibility, and user-friendly design to serve a wide audience while complying with modern privacy regulations.

Stakeholders

- **Pet Owners:** Pet owners use the app to report lost pets and search for found ones by uploading photos, descriptions, and locations. They need an easy-to-use interface, accurate matching, and timely notifications.
- **Pet Finders:** Pet finders report animals they find by uploading photos and location details. They need a simple process and recognition for their efforts in reuniting pets with their owners.
- **App Development Team**: Developers build and maintain the app, requiring clear goals, detailed requirements, and efficient tools to deliver a functional product.

Functional Requirements

1. **User Registration and Authentication**:
   - Users can create accounts and log in securely using email or social media accounts.
2. **Lost and Found Pet Reporting**:
   - Users can post details of lost or found pets, including photos, descriptions, and location details.
   - Posts may include fields for pet breed, color, size, and distinguishing features.

3. **File Attachments:**
   - Users can attach pet photos from their device or take new pictures directly using their phone's camera for upload.

4. **Image Recognition for Pet Matching**:
   - Use AI-based image recognition (YOLO algorithm) to match uploaded images of found pets with lost pet reports.
   - Display visual similarity scores for potential matches using an intuitive and user-friendly interface

5. **Geolocation Features**:
   - Allow users to set geolocation for reports and filter results by proximity.
   - Provide interactive maps to display the locations of lost or found pets. (google maps API).

6. **Notifications and Alerts**:
   - Notify users of potential matches via email or push notifications.
   - Provide alerts for new posts in the user's specified area.

7. **Search by Filters**:
   - Allow users to filter posts by location, date, breed, and other criteria.

8. **Mobile Accessibility**:
   -Develop a responsive mobile app for Android and iOS using react native.

9. **Data Security and Privacy**:
   - Implement secure data storage and ensure compliance with privacy laws like GDPR.

Non-Functional Requirements

1. **Performance:**
   - Ensure fast response times (< 2 seconds for most queries).
   - Support simultaneous usage by at least 10,000 users.

2. **Scalability:**
   - Design the system to handle growth in user base and data volumes without degradation in performance.

3. **Reliability and Availability:**
   - Guarantee 99.9% uptime with redundancy and failover mechanisms.

4. **Usability:**
   - Provide an intuitive interface that is easy to use for all age groups.
   - Include accessibility features for visually impaired users.

5. **Security:**
   - Ensure secure user authentication and authorization using firebase.

- Encrypt sensitive data both in transit and at rest.

6. **Maintainability:**
   - Modular codebase to support easy updates and bug fixes.
   - Documentation for all APIs and components.

7. **Compliance:**
   - Ensure compliance with GDPR and CCPA privacy regulations.

8. **Portability:**
   - The app must run seamlessly on Android and IOS.

9. **Monitoring and Logging:**
   - Implement real-time monitoring and error logging to detect and resolve issues quickly.

Use Cases

1. **User Registration:**
   - User opens the app and selects the "Sign Up" option.
   - User enters email, password, and personal details or connects via social media.
   - System verifies email or social media credentials and creates the account.

2. **Lost Pet Report:**
   - User logs in and selects "Report Lost Pet."
   - User uploads photos, enters pet details (breed, color, size), and sets the location.
   - System stores the data and updates the database.

3. **Found Pet Report:**
   - User logs in and selects "Report Found Pet."
   - User uploads photos and enters pet details.
   - System matches the image using AI and suggests possible matches.

4. **Search for Pets:**
   - User selects "Search for Pets" and applies filters for location, breed, and date.
   - System returns matching results and highlights visual similarities.

5. **Notifications:**
   - System sends notifications to users about matches or new posts in their area.
   - User receives alerts and views the details.

6. **Map View:**
   - User opens the map view to locate lost and found pets near their area.
   - System displays markers on the map with details when clicked.

7. **Profile Management:**
   - User updates profile details, including contact information and notification preferences.
   - System saves the updates and ensures data consistency.

Technological Requirements

1. **Programming Languages and Frameworks:**
   - Frontend: React Native for mobile.
   - Backend: Node.js. o AI Model: YOLO image processing.

2. **Database:**
   - Use a relational database (e.g., PostgreSQL) for structured data.
   - Integrate a NoSQL database (e.g., MongoDB) for handling large volumes of unstructured data like images.
3. **Cloud Infrastructure:**
   - Deploy on a cloud in firebase for scalability.
   - Use services like AWS S3 for image storage and AWS Lambda for serverless functions.

4. **AI and Machine Learning:**
   - Train a model on datasets like Oxford-IIIT Pet Dataset or Kaggle's Dogs vs. Cats Dataset.
   - Use pre-trained models such as YOLOv5 for real-time image processing.

5. **Geolocation Services:**
   - Use Google Maps API or Mapbox for geolocation and interactive maps.

6. **Security:**
   - Implement HTTPS for secure communication.
   - Use firebase for user authentication.
   - Encrypt sensitive data in transit and at rest.

7. **APIs:**
   - Use REST or GraphQL for seamless communication between frontend and backend.

8. **Testing and Quality Assurance:**
   - Use tools like Jest for frontend testing and Pytest for backend.
   - Perform stress testing with tools like Apache JMeter.

Architectural Requirements

1. **Architecture:**
   - Use a microservices architecture for scalability and ease of deployment.
   - Employ a layered architecture for clear separation of concerns.

2. **Scalability:**
   - Design the system to handle increased user traffic and data volumes.
   - Use load balancers and auto-scaling groups to manage traffic spikes.

3. **Integration:**
   - Partner with external platforms like Facebook and Instagram for social media sharing.

4. **Reliability:**
   - Ensure 99.9% uptime with redundancy and failover mechanisms.
   - Use a Content Delivery Network (CDN) for faster content delivery.

5. **Monitoring and Logging:**
   - Use monitoring tools like New Relic or Datadog to track system performance.
   - Implement centralized logging for all the systems with tools like ELK Stack (Elasticsearch, Logstash, Kibana).