**Software Test Plan (STP)**

**Students:**

**Name:** Or Avichzer Elmalih **| ID:** 318659455

**Name:** Noa Danon **| ID:** 207229303

**Name:** Mark Tsyrlin **| ID:** 324273226

**1. Introduction:**

This Software Test Plan outlines the testing strategy for the PAWsitive Life mobile application. The goal of this testing effort is to verify that the app meets its functional and non-functional requirements, including the ability to create dog profiles, deliver AI-generated personalized articles, and provide location-based services for dog parks. The tests aim to ensure correct functioning of user interactions, backend integration, and third-party API services.

**2. Test Items:**

* Dog Profile Management Module
* Personalized Article Recommendation Engine
* Dog Park Locator Feature
* Reminder System for Dog Care
* User Registration and Authentication

**3. Features to be Tested:**

* User registration, login, and authentication
* Adding, editing, and deleting dog profiles
* Request and display of personalized articles based on dog attributes
* Google Maps integration to display nearby dog parks
* Setting and receiving reminders for vaccines and treatments
* End-to-end data flow from frontend to Firebase and third-party APIs

**4. Features Not to be Tested:**

* Admin panel for content management (future version)
* Offline mode functionality
* Full stress testing under extreme load
* Advanced content moderation of AI-generated articles

**5. Testing Strategy:**

* We will employ the following types of testing:
* Unit Testing: For internal components such as form validation, utility classes, and data formatting.
* Integration Testing: Between app components and Firebase, Google Maps API, and OpenAI API.
* System Testing: To verify complete functionality from user input to final display.
* Acceptance Testing: Based on user scenarios to ensure the app delivers expected value.

Test cases will be developed from the requirements and detailed design document. Manual and semi-automated methods will be used as appropriate.

**6. Test Environment:**

* **Frontend:** Android application developed in Kotlin
* **Backend:** Firebase Realtime Database and Authentication
* **Third-party Services:** Google Maps API, OpenAI API
* **Tools:**
  + Android Studio with Firebase Assistant
  + GitHub for version control
  + Gradle for builds
  + Logcat for runtime monitoring
  + Optional: Firebase Crashlytics for crash tracking

Testing will be conducted on physical Android devices and emulators running the latest Android OS versions.