|  |
| --- |
| **Plan de Proyecto CAPSTONE “PawFinder”** |

**Proyecto de Desarrollo de Aplicación para la Búsqueda de Mascotas Perdidas**

Curso: CAPSTONE\_301V

Profesor: Patricio Soto – Natalia Fuentes

Integrantes: Alonso Arellano

Javier Villalobos

Jorge Villalobos

Carrera: Ingeniería en Informática

Fecha: 14 de agosto de 2024

**Índice**

[**1.** **Resumen Ejecutivo** 3](#_Toc184518962)

[**2.** **Objetivos del Proyecto** 4](#_Toc184518963)

[**3.** **Alcance del Proyecto** 5](#_Toc184518964)

[**4.** **Plan de Gestión del Cronograma** 7](#_Toc184518965)

[**5.** **Plan de Gestión de Recursos** 10](#_Toc184518966)

[**6.** **Plan de Gestión del Presupuesto** 12](#_Toc184518967)

[**7.** **Plan de Gestión de Riesgo** 14](#_Toc184518968)

[**8.** **Plan de Pruebas** 18](#_Toc184518969)

[**9.** **Cierre del Proyecto** 22](#_Toc184518970)

[**10.** **Lecciones Aprendidas** 24](#_Toc184518971)

# **Resumen Ejecutivo**

* 1. **Descripción del Proyecto:**

PawFinder es una aplicación web y móvil diseñada para centralizar la información de mascotas perdidas, encontradas y en adopción, facilitando su búsqueda y conexión entre dueños y rescatistas. Inspirada en plataformas como Sosafe, pero enfocada exclusivamente en animales, PawFinder también ofrece funcionalidades para donaciones y publicidad, garantizando su gratuidad para los usuarios finales.

* 1. **Objetivos del Proyecto:**

1. Ayudar a las familias a reencontrarse con sus mascotas perdidas.
2. Generar un impacto positivo en la comunidad animalista.
3. Crear un espacio centralizado que unifique la información actualmente dispersa en redes sociales y otras plataformas.
4. Ofrecer opciones de adopción y donaciones para apoyar causas relacionadas con el bienestar animal.
   1. **Importancia del Proyecto:**

La dispersión de información en múltiples plataformas dificulta las búsquedas efectivas de mascotas perdidas. PawFinder busca resolver este problema al centralizar datos, optimizar la comunicación y promover la colaboración entre los usuarios, ayudando a reducir el tiempo de reencuentro o adopción.

* 1. **Principales Elementos del Plan:**

1. **Plazos:** Se espera completar el desarrollo y realizar pruebas dentro de un período de 3 meses, seguido de un lanzamiento público.
2. **Costos:** Los costos del proyecto se limitan principalmente a infraestructura tecnológica (servidores, base de datos, servicios de almacenamiento) y desarrollo de software.
3. **Entregables Clave:**
   * Aplicación móvil, con las secciones de login, registro de mascotas perdidas y encontradas, adopciones, donaciones y publicidad.
   * Integración de Firebase Storage para el manejo de imágenes y MongoDB para la gestión de datos.
   * Plan de pruebas finalizado para asegurar la calidad del sistema antes del lanzamiento.

# **Objetivos del Proyecto**

* 1. **Objetivo General:**

Diseñar, desarrollar e implementar una aplicación web y móvil llamada PawFinder que centralice la información de mascotas perdidas, encontradas y en adopción, permitiendo conectar a las familias con sus mascotas y fomentar la colaboración en la comunidad animalista.

* 1. **Objetivos Específicos:**

1. **Análisis y planificación del sistema:**
   * Realizar un análisis de los requisitos funcionales y no funcionales para definir claramente las funcionalidades necesarias, como registro de usuarios, gestión de publicaciones de mascotas perdidas y encontradas, y la sección de adopciones.
   * Estudiar las necesidades del mercado para identificar las principales características que diferencien a PawFinder de otras soluciones existentes.
2. **Diseño de la aplicación:**
   * Crear prototipos de alta fidelidad que definan la estructura visual y de navegación del sistema.
   * Diseñar una arquitectura técnica adecuada que integre tecnologías como Firebase Storage para imágenes y MongoDB para la base de datos.
3. **Desarrollo del sistema:**
   * Implementar el backend con FastAPI en Python, conectando la base de datos MongoDB y asegurando la funcionalidad para manejar datos de usuarios y mascotas.
   * Desarrollar la aplicación móvil utilizando Ionic y Angular, incluyendo las secciones de login, registro de mascotas perdidas/encontradas, adopciones, donaciones y publicidad.
   * Implementar la integración de pasarelas de pago para gestionar donaciones.
4. **Pruebas y validación:**
   * Diseñar y ejecutar un plan de pruebas para evaluar la funcionalidad, usabilidad y seguridad del sistema.
   * Realizar pruebas end-to-end con Cypress y verificar la API mediante Swagger UI.
5. **Preparación para el lanzamiento:**
   * Documentar todas las funcionalidades y realizar manuales de usuario.
   * Desplegar la aplicación en un entorno accesible para los usuarios, asegurando la estabilidad del sistema.
   1. **Criterios de Éxito:**
6. **Cumplimiento de funcionalidades:**

La aplicación debe permitir el registro y gestión de usuarios, mascotas perdidas/encontradas, y publicaciones de adopción de manera efectiva.

1. **Usabilidad:**  
   El sistema debe ser intuitivo y fácil de usar tanto en la versión web como en móvil, asegurando una experiencia positiva para los usuarios.
2. **Impacto en la comunidad:**

Alcanzar al menos 100 usuarios activos durante el primer mes de lanzamiento, con un número significativo de publicaciones relacionadas con mascotas perdidas/encontradas.

1. **Estabilidad del sistema:**

Realizar pruebas exitosas que aseguren tiempos de respuesta inferiores a 2 segundos en el uso promedio de la aplicación.

1. **Gestión de donaciones:**

Implementar y verificar una pasarela de pagos funcional que permita transacciones seguras y efectivas.

# **Alcance del Proyecto**

* 1. **Descripción del Alcance:**
     1. **Incluido en el proyecto:**
  + Desarrollo de una aplicación móvil funcional llamada PawFinder.
  + Implementación de las funcionalidades principales:
    - Registro de usuarios y login seguro.
    - Publicación y gestión de mascotas perdidas y encontradas.
    - Foro de adopciones con filtros y búsquedas personalizadas.
    - Integración de un módulo de donaciones utilizando una pasarela de pago sencilla como WebPay.
    - Publicidad dentro de la aplicación para generar ingresos y mantener el servicio gratuito.
  + Integración de Firebase Storage para gestionar el almacenamiento de imágenes.
  + Implementación de una base de datos centralizada con MongoDB para el manejo eficiente de datos.
  + Pruebas funcionales y de calidad del sistema utilizando herramientas como Cypress y Swagger UI.
    1. **Excluido del proyecto:**
  + Análisis de datos avanzado (considerado como una posible mejora futura).
  + Desarrollo de funcionalidades adicionales fuera del alcance definido, como integración con redes sociales o notificaciones push avanzadas.
  + Garantía de soporte post-implementación a largo plazo, ya que el enfoque está en el desarrollo inicial.
    1. **Entregables Principales:**

1. **Aplicación móvil funcional:** Incluyendo las secciones de login, mascotas perdidas y encontradas, adopciones, donaciones y publicidad.
2. **Backend operativo:** Implementado con FastAPI y conectado a MongoDB para manejar datos de usuarios, publicaciones y transacciones.
3. **Sistema de gestión de imágenes:** Integración con Firebase Storage para el almacenamiento externo de imágenes.
4. **Documentación técnica:**
   * Manuales de usuario y administrador.
   * Especificaciones de la API.
   * Registro de pruebas realizadas y resultados obtenidos.
5. **Prototipos visuales:** Diseños UI/UX que guían el desarrollo y aseguran la facilidad de uso.
6. **Pasarela de pago operativa:** Sistema integrado para gestionar donaciones.
   * 1. **Criterios de Aceptación del Cliente o Partes Interesadas:**
7. **Funcionalidad completa:** Todas las funcionalidades definidas deben estar operativas y probadas.
8. **Usabilidad:** La aplicación debe ser intuitiva, con un diseño amigable para los usuarios, accesible tanto desde dispositivos móviles como desde la web.
9. **Estabilidad:** El sistema debe manejar un volumen inicial estimado de al menos 100 usuarios concurrentes sin interrupciones.
10. **Seguridad:** Cumplir con los estándares básicos de seguridad, incluyendo autenticación y autorización adecuadas.
11. **Eficiencia:** El tiempo de carga de las secciones principales no debe superar los 2 segundos.
12. **Documentación:** Entregar toda la documentación requerida para el uso, mantenimiento y desarrollo futuro del sistema.

# **Plan de Gestión del Cronograma**

El proyecto PawFinder se desarrollará en 7 fases principales con actividades específicas y plazos definidos. El cronograma detalla los hitos clave y los plazos estimados para garantizar el cumplimiento de los objetivos del proyecto:

* 1. **Inicio del Proyecto: (14 de agosto - 18 de agosto)**
* Aprobación del proyecto y asignación de recursos.
* Creación del acta de constitución y planificación del presupuesto.
  1. **Planificación: (21 de agosto - 1 de septiembre)**
* Definición de requisitos, especificación técnica y diseño de la arquitectura.
* Desarrollo del cronograma y análisis de riesgos.
  1. **Desarrollo de la Aplicación: (4 de septiembre - 27 de octubre)**
* Desarrollo del backend, frontend e integración de funcionalidades clave.
* Configuración de servidores y base de datos.
  1. **Pruebas de Integración y Calidad: (30 de octubre - 10 de noviembre)**
* Pruebas del backend, frontend, y validación de funcionalidades.
  1. **Implementación y Despliegue: (13 de noviembre - 17 de noviembre)**
* Configuración del entorno de producción, migración de datos y lanzamiento.
  1. **Marketing y Promoción: (20 de noviembre - 24 de noviembre)**
* Estrategia de marketing digital y lanzamiento de material publicitario.
  1. **Mantenimiento y Soporte: (27 de noviembre - 22 de diciembre)**
* Resolución de errores y actualización de funcionalidades.
  1. **Actividades y Tareas:**

| Fase/Actividad | Duración Estimada | Dependencias Previas |
| --- | --- | --- |
| Inicio del Proyecto | 1 semana | N/A |
| Definición de Objetivos | 1 día | N/A |
| Aprobación del Proyecto | 1 día | Definición de Objetivos |
| Asignación de Recursos | 1 día | Aprobación del Proyecto |
| Acta de Constitución del Proyecto | 2 días | Asignación de Recursos |
| Planificación del Presupuesto | 2 días | Asignación de Recursos |
| Planificación | 2 semanas | Inicio del Proyecto |
| Requisitos del Proyecto | 3 días | Acta de Constitución del Proyecto |
| Especificación Técnica | 3 días | Requisitos del Proyecto |
| Diseño de Arquitectura de Software | 4 días | Especificación Técnica |
| Cronograma Detallado | 2 días | Diseño de Arquitectura de Software |
| Análisis de Riesgos | 2 días | Cronograma Detallado |
| Plan de Gestión de Calidad | 2 días | Cronograma Detallado |
| Desarrollo de la Aplicación | 8 semanas | Planificación |
| Desarrollo del Backend | 3 semanas | Diseño de Arquitectura de Software |
| Configuración de Servidor y BD | 1 semana | Desarrollo del Backend |
| Desarrollo del Frontend | 3 semanas | Configuración de Servidor y BD |
| Integración de Funcionalidades | 2 semanas | Desarrollo del Frontend |
| Pruebas de Integración y Calidad | 2 semanas | Desarrollo de la Aplicación |
| Pruebas del Backend | 5 días | Integración de Funcionalidades |
| Pruebas del Frontend | 5 días | Pruebas del Backend |
| Implementación y Despliegue | 1 semana | Pruebas de Integración y Calidad |
| Configuración de Producción | 3 días | Pruebas del Frontend |
| Migración de Datos | 2 días | Configuración de Producción |
| Lanzamiento | 2 días | Migración de Datos |
| Marketing y Promoción | 2 semanas | Implementación y Despliegue |
| Estrategia de Marketing | 1 semana | N/A |
| Material Publicitario | 1 semana | N/A |
| Mantenimiento y Soporte | 4 semanas | Implementación y Despliegue |
| Monitoreo y Resolución de Errores | 2 semanas | N/A |
| Actualizaciones | 2 semanas | Monitoreo y Resolución de Errores |

* 1. **Cronograma**

| Fase / Actividad | Duración Estimada | Fecha de Inicio | Fecha de Fin | Entregables / Hitos |
| --- | --- | --- | --- | --- |
| 1. Inicio del Proyecto | **1 semana** | **14 de agosto** | **18 de agosto** | **Aprobación del proyecto** |
| Definición de Objetivos | 1 día | 14 de agosto | 14 de agosto | Objetivos definidos |
| Aprobación del Proyecto | 1 día | 15 de agosto | 15 de agosto | Documento aprobado |
| Asignación de Recursos | 1 día | 16 de agosto | 16 de agosto | Recursos asignados |
| Acta de Constitución del Proyecto | 2 días | 17 de agosto | 18 de agosto | Acta de constitución |
| Planificación del Presupuesto | 2 días | 17 de agosto | 18 de agosto | Presupuesto inicial |
| 2. Planificación | **2 semanas** | **21 de agosto** | **1 de septiembre** | **Plan del proyecto** |
| Requisitos del Proyecto | 3 días | 21 de agosto | 23 de agosto | Documentación de requisitos |
| Especificación Técnica | 3 días | 24 de agosto | 28 de agosto | Especificaciones técnicas |
| Diseño de Arquitectura de Software | 4 días | 29 de agosto | 1 de septiembre | Diagrama de arquitectura |
| Cronograma Detallado | 2 días | 30 de agosto | 31 de agosto | Cronograma aprobado |
| Análisis de Riesgos | 2 días | 1 de septiembre | 1 de septiembre | Informe de riesgos |
| Plan de Gestión de Calidad | 2 días | 1 de septiembre | 1 de septiembre | Plan de calidad |
| 3. Desarrollo de la Aplicación | **8 semanas** | **4 de septiembre** | **27 de octubre** | **Sistema funcional** |
| Desarrollo del Backend | 3 semanas | 4 de septiembre | 22 de septiembre | API funcional |
| Configuración de Servidor y Base de Datos | 1 semana | 4 de septiembre | 8 de septiembre | Base de datos en producción |
| Implementación de API y Servicios | 2 semanas | 11 de septiembre | 22 de septiembre | API desplegada |
| Desarrollo del Frontend | 3 semanas | 25 de septiembre | 13 de octubre | Interfaces de usuario |
| Diseño de Interfaces (UI/UX) | 1 semana | 25 de septiembre | 29 de septiembre | Prototipos UI/UX |
| Implementación de Interfaces | 2 semanas | 2 de octubre | 13 de octubre | Interfaces funcionales |
| Integración de Funcionalidades Clave | 2 semanas | 16 de octubre | 27 de octubre | Plataforma integrada |
| Módulo de Donaciones | 1 semana | 16 de octubre | 20 de octubre | Donaciones integradas |
| Análisis de Datos y Reportes | 1 semana | 23 de octubre | 27 de octubre | Análisis de datos |
| 4. Pruebas de Integración y Calidad | **2 semanas** | **30 de octubre** | **10 de noviembre** | **Informe de pruebas** |
| Pruebas del Backend | 5 días | 30 de octubre | 3 de noviembre | Resultados de pruebas backend |
| Pruebas del Frontend | 5 días | 6 de noviembre | 10 de noviembre | Resultados de pruebas frontend |
| Pruebas de Seguridad | 3 días | 6 de noviembre | 8 de noviembre | Informe de seguridad |
| Validación de Funcionalidades | 2 días | 9 de noviembre | 10 de noviembre | Aprobación de funcionalidades |
| 5. Implementación y Despliegue | **1 semana** | **13 de noviembre** | **17 de noviembre** | **Aplicación en producción** |
| Configuración del Entorno de Producción | 3 días | 13 de noviembre | 15 de noviembre | Servidor configurado |
| Migración de Datos y Pruebas en Producción | 2 días | 16 de noviembre | 17 de noviembre | Datos migrados y validados |
| Lanzamiento de la Aplicación | 2 días | 17 de noviembre | 17 de noviembre | Lanzamiento oficial |
| 6. Marketing y Promoción | 2 semanas | 20 de noviembre | 24 de noviembre | Estrategia de marketing ejecutada |
| Estrategia de Marketing Digital | 1 semana | 20 de noviembre | 24 de noviembre | Estrategia aprobada |
| Material Publicitario | 1 semana | 20 de noviembre | 24 de noviembre | Publicidad creada y lanzada |
| 7. Mantenimiento y Soporte | **4 semanas** | **27 de noviembre** | **22 de diciembre** | **Soporte continuo** |
| Monitoreo y Resolución de Errores | 2 semanas | 27 de noviembre | 8 de diciembre | Errores resueltos |
| Actualizaciones y Nuevas Funcionalidades | 2 semanas | 11 de diciembre | 22 de diciembre | Funcionalidades adicionales |
| Soporte al Usuario | Continuo | 27 de noviembre | Continuo | Asistencia al usuario |

# **Plan de Gestión de Recursos**

* 1. **Recursos Humanos: Roles y Responsabilidades**

El equipo del proyecto PawFinder está compuesto por los siguientes roles, cada uno con responsabilidades específicas:

1. **Javier Villalobos (Líder del Proyecto):**
   * Coordinación general del proyecto.
   * Supervisión de las fases de desarrollo y aseguramiento de calidad.
   * Gestión de reuniones con las partes interesadas y resolución de problemas.
2. **Jorge Villalobos (Desarrollador Backend):**
   * Implementación y mantenimiento de la API utilizando Python con FastAPI.
   * Configuración y gestión de la base de datos MongoDB.
   * Desarrollo del sistema de almacenamiento de imágenes con Firebase Storage.
3. **Alonso Arellano (Desarrollador Frontend):**
   * Desarrollo de la interfaz de usuario utilizando Ionic y Angular.
   * Implementación de las funcionalidades clave de la aplicación móvil y web.
   * Diseño de las interfaces de usuario (UI/UX).
4. **Colaboradores Externos (si aplica):**
   * Consultores o asesores para la integración de pasarelas de pago (WebPay).
   * Especialistas en marketing digital para la fase de promoción y lanzamiento.
   1. **Recursos Materiales**

El proyecto requiere de las siguientes herramientas, equipos y materiales:

1. **Herramientas de Desarrollo:**
   * Python (FastAPI) para el backend.
   * MongoDB y MongoDB Compass para la gestión de datos.
   * Firebase Storage para el almacenamiento de imágenes.
   * Ionic y Angular para el desarrollo del frontend.
2. **Infraestructura Tecnológica:**
   * Servidores cloud para el despliegue del backend y base de datos.
   * Equipo de desarrollo: computadoras con capacidad para ejecutar entornos de desarrollo y simuladores.
3. **Herramientas de Gestión y Comunicación:**
   * Jira o Trello para el seguimiento de tareas.
   * GitHub para el control de versiones.
   * Slack o Google Meet para reuniones de equipo.
4. **Otros Materiales:**
   * Recursos visuales para el diseño de interfaces (íconos, imágenes, paletas de colores).
   * Servicios de pasarelas de pago (WebPay o similar).
   1. **Asignación de Recursos**

La asignación de tareas a los integrantes del equipo es la siguiente:

|  |  |
| --- | --- |
| Tarea | Responsable(s) |
| Coordinación del proyecto | Javier Villalobos |
| Desarrollo de la API | Jorge Villalobos |
| Configuración de la base de datos | Jorge Villalobos |
| Desarrollo del frontend (UI/UX) | Alonso Arellano |
| Implementación del módulo de donaciones | Jorge Villalobos y Alonso Arellano |
| Integración de Firebase Storage | Jorge Villalobos |
| Pruebas de funcionalidad y calidad | Todo el equipo |
| Estrategia de marketing digital | Javier Villalobos y colaboradores externos |
| Preparación del entorno de producción | Jorge Villalobos |
| Documentación del proyecto | Todo el equipo |

# **Plan de Gestión del Presupuesto**

* 1. **Estimación de Costos**

El presupuesto del proyecto PawFinder se ha estimado en pesos chilenos (CLP) basado en las necesidades identificadas en las fases y tareas del proyecto:

1. **Personal (Trabajo del Equipo):**
   * Tiempo de desarrollo del backend (FastAPI y MongoDB): **$1.200.000 CLP**.
   * Tiempo de desarrollo del frontend (Ionic y Angular): **$1.200.000 CLP**.
   * Diseño de interfaces (UI/UX): **$800.000 CLP**.
   * Gestión del proyecto: **$600.000 CLP**.
   * **Total estimado para personal:** **$3.800.000 CLP**.
2. **Tecnología y Herramientas:**
   * Licencias de software y herramientas (Firebase. MongoDB Atlas): **$250.000 CLP**.
   * Servicios de hosting en la nube: **$200.000 CLP**.
   * Herramientas de diseño gráfico y recursos visuales: **$100.000 CLP**.
   * **Total estimado para tecnología:** **$550.000 CLP**.
3. **Infraestructura:**
   * Servidores y configuración del entorno de producción: **$400.000 CLP**.
   * Pasarela de pago (WebPay): **$100.000 CLP**.
   * **Total estimado para infraestructura:** **$500.000 CLP**.
4. **Marketing y Promoción:**
   * Estrategias de marketing digital: **$300.000 CLP**.
   * Material publicitario: **$200.000 CLP**.
   * **Total estimado para marketing:** **$500.000 CLP**.
5. **Contingencias:**
   * Reserva para imprevistos o ajustes en el desarrollo: **$400.000 CLP**.

**Total Estimado del Proyecto:** **$5.750.000 CLP**.

* 1. **Presupuesto Aprobado**

El total de recursos financieros asignados para el proyecto PawFinder es de **$5.750.000 CLP**. Este presupuesto cubre todas las necesidades previstas, incluyendo personal, tecnología, infraestructura, marketing y contingencias.

* + 1. **Control de Costos**

Se implementarán las siguientes medidas para gestionar y controlar los costos durante el desarrollo del proyecto:

1. **Revisión Periódica:**
   * Se realizarán revisiones quincenales de los gastos utilizando un informe detallado que compare costos reales frente al presupuesto asignado.
2. **Autorización de Cambios:**
   * Cualquier ajuste o incremento en el presupuesto deberá ser aprobado por el líder del proyecto, Javier Villalobos, previo análisis de su impacto en las demás categorías.
3. **Uso de la Reserva de Contingencia:**
   * La reserva de **$400.000 CLP** solo se utilizará para cubrir costos imprevistos relacionados directamente con el desarrollo o tecnología.
4. **Herramientas de Seguimiento:**
   * Se utilizará una hoja de cálculo dinámica para registrar y controlar todos los gastos en tiempo real. Esta hoja será revisada semanalmente por el equipo.
5. **Documentación de Cambios:**
   * Todos los cambios en el presupuesto deberán ser documentados y aprobados, incluyendo la justificación y su impacto en el cronograma y las metas del proyecto.

# **Plan de Gestión de Riesgo**

* 1. **Tipos de Riesgos Identificados**
     1. **Riesgos del Proyecto**

Ponen en peligro la ejecución del proyecto PawFinder, exigiendo recursos adicionales para su finalización:

* **Presupuesto:** Incremento en los costos, por ejemplo, si se requiere contratar servicios externos para la publicidad de la app o ampliar la infraestructura en Firebase.
* **Planificación:** Retrasos en las fases de desarrollo debido a imprevistos técnicos.
* **Personal:** Necesidad de más desarrolladores o de personal con mayor experiencia en áreas específicas como minería de datos.
* **Recursos:** Falta de herramientas adecuadas, como servicios cloud adicionales para almacenamiento.
* **Requisitos:** Expansión de las funcionalidades inicialmente previstas, como agregar nuevas integraciones o características demandadas por los usuarios.
  + 1. **Riesgos Técnicos**

Impactan la calidad final del aplicativo:

* **Requisitos:** Aparición de nuevas necesidades técnicas, como compatibilidad con otros sistemas de identificación de mascotas.
* **Diseño:** Modificaciones en la arquitectura de la app para soportar más tráfico o nuevas funcionalidades.
* **Implementación:** Retrasos por dificultades técnicas al integrar Firebase con MongoDB o al implementar machine learning.
* **Interfaz:** Problemas de usabilidad en la interacción del usuario con el registro de mascotas.
* **Verificación:** Necesidad de realizar pruebas adicionales para garantizar estabilidad y seguridad en múltiples plataformas.
* **Mantenimiento:** Recursos adicionales necesarios para actualizaciones regulares post-lanzamiento.
* **Incertidumbre técnica:** Falta de experiencia con nuevas tecnologías aplicadas, como minería de datos en el backend.
* **Tecnologías desconocidas:** Aprendizaje de herramientas adicionales si los requerimientos del proyecto cambian, como migrar a servicios más avanzados de pago.
  + 1. **Riesgos del Negocio**

**Ponen en peligro la continuidad o viabilidad del proyecto:**

* **Utilidad:** Baja demanda del aplicativo en el mercado.
* **Estrategia:** Desalineación con los objetivos estratégicos de los integrantes del proyecto.
* **Dificultad de venta:** Dificultades para atraer usuarios o aliados clave, como veterinarias o tiendas de mascotas.
* **Apoyo:** Pérdida de apoyo de los gestores superiores o falta de interés de posibles patrocinadores.
* **Presupuesto:** Limitación de fondos para el desarrollo o la promoción de la aplicación.
  1. **Gestión de Riesgos**
     1. **Estrategia Proactiva**

Se aplicará un enfoque proactivo para minimizar riesgos antes de que se conviertan en problemas:

* Identificar los riesgos potenciales en cada fase del desarrollo.
* Priorizar los riesgos en función de su impacto y probabilidad de ocurrencia.
* Elaborar planes de contingencia para abordar riesgos específicos (por ejemplo, alternativas de almacenamiento si Firebase tiene limitaciones).
* Evaluar la viabilidad del proyecto periódicamente para detectar desviaciones tempranas.
  + 1. **Implementación de la Gestión de Riesgos**
* **Valoración:** Estimar el impacto de cada riesgo.
* **Identificación:** Mapear riesgos en categorías técnicas, del proyecto y del negocio.
* **Análisis:** Analizar causas y posibles efectos.
* **Priorización:** Utilizar una matriz de riesgos con un rango de impacto (de Muy Bajo a Muy Alto).
* **Control:** Establecer medidas para reducir riesgos significativos.
* **Planificación:** Desarrollar un plan detallado de mitigación.
* **Resolución:** Implementar soluciones cuando los riesgos se materialicen.
* **Monitorización:** Revisar periódicamente los riesgos identificados y actualizarlos.
  1. **Matriz de Riesgos**

La matriz de riesgos de PawFinder será utilizada para clasificar y priorizar los riesgos según:

* **Probabilidad de ocurrencia:** Desde Muy Baja (1) hasta Muy Alta (5).
* **Impacto:**
  + **Muy Bajo (10):** Poco relevante.
  + **Bajo (20):** Afecta marginalmente al proyecto.
  + **Moderado (30):** Impacto significativo.
  + **Muy Alto (50):** Puede llevar al fracaso del proyecto.

Ejemplo de Matriz:

|  |  |  |  |
| --- | --- | --- | --- |
| Riesgo | Probabilidad | Impacto | Nivel de Riesgo |
| Retraso en Firebase | 3 | 30 | Moderado |
| Baja demanda inicial | 2 | 50 | Muy Alto |
| Problemas de usabilidad | 4 | 20 | Bajo |

| Actividad | Posibles Riesgos | Probabilidad | Impacto | Valor | Mitigación |
| --- | --- | --- | --- | --- | --- |
| Acta de constitución del proyecto | N/A | N/A | N/A | N/A | N/A |
| Revisión de Documentos | Inconsistencias en los requerimientos | Baja (2) | Alto (40) | 80 | Revisar los requerimientos con los integrantes del equipo y partes interesadas para asegurar que sean claros, corregir y validar nuevamente. |
| Calendario del plan | Retraso en las fechas comprometidas | Alta (4) | Moderado (30) | 120 | Usar metodologías ágiles como Scrum para iteraciones cortas y actualizaciones frecuentes. Monitorear avances con herramientas como Jira o Trello. |
| Estrategia de pruebas y casos de prueba | Pruebas insuficientes o incorrectas | Moderada (3) | Muy Alto (50) | 150 | Crear un plan de pruebas exhaustivo. Automatizar pruebas unitarias, de integración y de aceptación. Validar continuamente con usuarios finales. |
| Ambiente de desarrollo, certificación y producción | Ambiente inadecuado para el despliegue | Moderada (3) | Muy Alto (50) | 150 | Configurar entornos en la nube como Firebase y asegurar su correcto funcionamiento antes del despliegue. Realizar pruebas de carga y estrés. |
| Modelo Relacional (MongoDB) | Mal diseño de la base de datos | Moderada (3) | Alto (40) | 120 | Validar el modelo junto con el equipo y expertos en bases de datos. Implementar revisiones periódicas del esquema de datos con pruebas de integridad. |
| Desarrollo del software | Fallas en las herramientas o interrupción por equipos obsoletos | Baja (2) | Moderado (30) | 60 | Utilizar control de versiones con GitHub para garantizar respaldo. Realizar mantenimiento preventivo de equipos y software utilizado. |
| Interfaz Usuario | Problemas de usabilidad y navegación | Alta (4) | Muy Alto (50) | 200 | Realizar pruebas de usabilidad con usuarios reales. Ajustar diseño iterativamente según feedback recolectado. |
| Manual de Usuario | Manual incompleto o confuso | Baja (2) | Moderado (30) | 60 | Revisar el manual con usuarios finales y asegurarse de incluir ejemplos prácticos, capturas de pantalla y pasos claros para cada funcionalidad. |
| Correcciones y observaciones (UAT) | Una corrección provoca errores en otros módulos | Moderada (3) | Moderado (30) | 90 | Aplicar buenas prácticas de control de versiones y realizar pruebas de regresión antes de desplegar cambios al sistema. |
| Documentación y paso a producción | Falta de documentación o pérdida de versión actualizada | Moderada (3) | Moderado (30) | 90 | Mantener documentación actualizada en repositorios como Notion o GitHub. Realizar respaldos diarios de versiones clave del proyecto. |
| Sistema de donaciones | Problemas de integración con pasarela de pago | Alta (4) | Muy Alto (50) | 200 | Realizar pruebas exhaustivas con la pasarela de pago elegida. Tener una pasarela de pago alternativa configurada como contingencia. |
| Publicidad y monetización | Baja efectividad en la publicidad | Moderada (3) | Alto (40) | 120 | Optimizar estrategias de marketing digital mediante el uso de herramientas como Google Ads o análisis de datos del usuario. |
| Análisis de datos y ML | Errores en la interpretación de patrones | Moderada (3) | Muy Alto (50) | 150 | Validar los resultados del modelo de machine learning con un equipo especializado y usuarios reales antes de implementar decisiones. |
| Registro de usuarios | Problemas con la seguridad de datos personales | Alta (4) | Muy Alto (50) | 200 | Implementar cifrado SSL/TLS y políticas de privacidad estrictas. Realizar auditorías de seguridad periódicas. |

# **Plan de Pruebas**

El plan de pruebas para PawFinder asegura que cada funcionalidad de la plataforma cumpla con los requisitos establecidos y funcione correctamente bajo diversas condiciones.

* 1. **Objetivo**

El objetivo de este plan es validar el correcto funcionamiento de la aplicación PawFinder, describiendo los tipos de pruebas realizadas, herramientas empleadas y métricas obtenidas, garantizando la calidad del software en términos de funcionalidad, integración y experiencia de usuario.

* 1. **Pruebas Realizadas**
     1. **Pruebas End-to-End (E2E)**

**Propósito:** Validar correcto flujo de la aplicación, en los diferentes formularios y registros.

**Herramienta Utilizada:** Cypress

**Casos de Prueba:**

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Funcionalidad | Descripción | Resultado Esperado |
| P001 | Registro de Usuario | Validar que el usuario es capaz de registrarse con datos validos | “Usuario Registrado Exitosamente” |
| P002 | Registro de Usuario | Probar datos inválidos para el formulario de registro | “Error de formato erróneo” |
| P003 | Login | Comprobar Login exitoso con credenciales correctas | “Token Generado” |
| P004 | Login | Validar que el Login falle por credenciales incorrectas | “Credenciales incorrectas” |
| P005 | Publicación de mascota | Probar publicación de una mascota perdida con todos los campos completos. | “Publicación creada exitosamente” |
| P006 | Publicación de mascota | Probar publicación sin campo obligatorio | “Campo (ej) es obligatorio” |
|  |  |  |  |

**Métricas:**

* Números de casos probados: 6
* Casos exitosos: 6
* Casos Fallidos: 0

(Todas las métricas corresponden a la etapa final del proyecto)

* + 1. **Pruebas de Integración**

Propósito: Verificar la interacción entre el frontend y la API, asegurando que las respuestas sean correctas

Herramienta Utilizada: Swagger UI

Casos de Prueba:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Funcionalidad | Descripción | Resultado Esperado |
| P007 | Confirmación POST/GET/PUT/DELETE  USUARIOS | Verificar que el api maneja correctamente a los usuarios CRUD | Respuestas 200 en los endpoints |
| P008 | Confirmación POST/GET/PUT/DELETE  Reportes Mascotas | Verificar que el api maneja correctamente los reportes generados por usuarios y admins | Respuestas 200 en los endpoints |
| P009 | Verificación de manejo de errores | Verificar que el manejo de excepciones dentro de la API funciona correctamente | El sistema no se cae al recibir un parámetro erróneo |
| P010 | Validación de Correcta autenticación del usuario. | Validar que el Login y sign up funcione correctamente y el usuario pueda acceder al sistema | Generación de token por parte del sistema, respuesta 200 |

**Métricas:**

* Números de endpoints probados: 19
* Casos exitosos: 19
* Casos Fallidos: 0

(Todas las métricas corresponden a la etapa final del proyecto)

* + 1. **Pruebas Manuales**

Propósito: Comprobar la experiencia de usuario y detección de errores no identificables durante pruebas automatizadas

Herramienta Utilizada: Simulación manual, navegadores y MongoDB Compass

Casos de Prueba:

|  |  |  |  |
| --- | --- | --- | --- |
| ID | Funcionalidad | Descripción | Resultado Esperado |
| P011 | Diseño y usabilidad en dispositivos móviles | Verificar que el aplicativo se adapta correctamente a las dimensiones verticales de un dispositivo móvil | Toda l UI accesible por el usuario |
| P012 | Funcionalidad de botones y formularios | Probar Botones y formularios no testeados durante las pruebas automatizadas | Funcionalidades correctas |
| P013 | Flujo completo | Flujo completo de la aplicación (desde la creación del usuario, inicio de sesión, creación de reporte y eliminación del mismo) | Ningún error durante el flujo principal de la aplicación |
| P014 | Verificacion datos almacenados MongoDB | Los datos deben guardarse correctamente en la base de datos | Datos estructurados correctamente |

**Métricas:**

* Total de Casos Manuales: 4
* Casos exitosos: 4
* Casos Fallidos: 0

(Todas las métricas corresponden a la etapa final del proyecto)

* + 1. **Resultados Pruebas:**

Resumen general de pruebas:

|  |  |  |  |
| --- | --- | --- | --- |
| Tipo de prueba | Casos Totales | Casos Exitosos | Tasa de Éxito |
| End-to-End (E2E) | 6 | 6 | 100% |
| Pruebas de Integración | 19 | 19 | 100% |
| Pruebas Manuales | 4 | 4 | 100% |
| Total Global | 29 | 29 | 100% |

**Métricas detalladas:**

Porcentaje de éxito Global = (Casos Exitosos/ Casos Totales) \* 100  
 ( 29 / 29 )\*100 = 100

**100% de Éxito Global**

Este es el resultado esperado, debido a la naturaleza del proyecto, solo un 100% de éxito es aceptable, considerando que los cálculos corresponden a la etapa final del proyecto.

Tasa Promedio de Ejecución:  
 - End-to-End = 2-3 min

* Integración= 8-9 min
* Manuales = 15 min

**Cobertura de las Pruebas:**

* End-To-End (E2E): Cotiene principalmente el área de autenticación de usuarios
* Integración: Abarca todos los endpoints de la aplicación (19 en total)
* Manual: Se comprendieron casos relacionados a la experiencia de usuario y revisión de los datos cargados en la base de datos por la API de Python.
  + 1. **Revisiones pendientes:**
* Optimización Almacenamiento: Optimizar el manejo de documentos e imágenes almacenados en FIREBASE
* Mejoras en el CSS del aplicativo para una mayor claridad
  + 1. **Tecnologías Utilizadas**

|  |  |  |
| --- | --- | --- |
| Tipología | Descripción | Usos |
| Cypress | Framework de pruebas end-to-end (E2E) que permite simular la interacción del usuario con la aplicación. | Validar los flujos completos, como el registro, inicio de sesión, reporte de mascotas perdidas y donaciones. |
| Simular acciones del usuario en tiempo real y verificar la respuesta de la plataforma. |
| Swagger UI | Herramienta para probar y documentar la API de PawFinder. | Probar los endpoints de la API relacionados con el registro de usuarios, gestión de reportes y obtención de datos. |
| Verificar que los datos enviados y recibidos sean correctos y que los códigos de respuesta cumplan con las especificaciones. |
| Navegadores Web | Entornos para realizar pruebas de compatibilidad y rendimiento. | Verificar la compatibilidad de PawFinder en navegadores como Google Chrome, Mozilla Firefox y Microsoft Edge. |
| Realizar pruebas de rendimiento, tiempos de carga y visualización adecuada en diferentes resoluciones. |
| MongoDB Compass | Herramienta gráfica para la gestión y verificación de datos en la base de datos | Verificar que los datos registrados en la plataforma (usuarios, reportes, adopciones) sean almacenados correctamente |
| Comprobar la integridad de los datos después de realizar operaciones CRUD (Crear, Leer, Actualizar, Eliminar). |

# **Cierre del Proyecto**

* 1. **Criterios de Cierre**

El proyecto PawFinder se considerará terminado cuando se cumplan las siguientes condiciones:

1. **Funcionalidad Completa:**

Todas las funcionalidades principales (registro de usuarios, publicación de mascotas perdidas/encontradas, adopciones, donaciones y publicidad) deben estar desarrolladas, probadas y operativas.

1. **Cumplimiento de Requisitos:**

El sistema debe cumplir con los requisitos funcionales y no funcionales definidos en el ERS (Especificación de Requisitos del Software).

1. **Pruebas Satisfactorias:**

Todas las pruebas (end-to-end, integración y manuales) deben alcanzar una tasa de éxito del 100%, garantizando estabilidad, usabilidad y seguridad.

1. **Entrega Documental Completa:**
   * Manuales de usuario y administrador.
   * Especificaciones técnicas de la API.
   * Informe final de pruebas y métricas.
2. **Despliegue en Producción:**

La aplicación debe estar desplegada y accesible desde plataformas móviles y web, con el entorno de producción configurado y probado.

* 1. **Entregables Finales**

1. **Aplicación Funcional:**
   * Aplicación móvil lista para su uso, con todas las funcionalidades operativas.
   * Integración de Firebase Storage para manejo de imágenes y MongoDB como base de datos.
2. **Documentación Técnica:**
   * Manual de usuario detallado.
   * Manual de administrador y guía de mantenimiento.
   * Especificaciones de la API para futuras integraciones.
3. **Resultados de Pruebas:**
   * Informes detallados de pruebas end-to-end, integración y manuales.
   * Registros de métricas y tasas de éxito.
4. **Infraestructura Desplegada:**
   * Servidor de producción configurado.
   * Base de datos operativa con respaldo inicial.
5. **Estrategia de Marketing y Promoción:**
   * Material publicitario finalizado y estrategias implementadas para promover la aplicación.

# **Lecciones Aprendidas**

El proyecto PawFinder representa una solución tecnológica innovadora y efectiva para abordar la problemática de las mascotas perdidas, integrando funcionalidades clave que centralizan la información y facilitan la conexión entre dueños de mascotas y la comunidad. A lo largo de su desarrollo, se han implementado estándares de calidad rigurosos, asegurando un sistema escalable, seguro y fácil de usar, que se adapta a las necesidades de los usuarios finales.

La integración de herramientas avanzadas, como Firebase para autenticación y almacenamiento, MongoDB para la gestión de datos y pasarelas de pago seguras como WebPay y PayPal, garantiza un alto nivel de confiabilidad y desempeño. Además, el módulo de análisis de datos permite extraer patrones valiosos para la toma de decisiones, potenciando las oportunidades de adopción y la ubicación estratégica de servicios relacionados.

En términos de impacto social, PawFinder no solo contribuye a la reunificación de familias con sus mascotas, sino que también promueve la adopción responsable y la colaboración comunitaria. Con un modelo sostenible que combina donaciones y publicidad, se asegura la continuidad y expansión de la plataforma sin costo directo para los usuarios.

El proyecto establece un precedente importante en el ámbito de las aplicaciones comunitarias, con un enfoque centrado en el usuario, accesibilidad y responsabilidad social. Se espera que PawFinder no solo cumpla con sus objetivos iniciales, sino que evolucione con el tiempo, incorporando nuevas funcionalidades basadas en las necesidades emergentes y la retroalimentación de sus usuarios.