IES Abastos – DAW 7K
Proyecto Integrador
Grupo:
Brandon Acapa
Rocio Garcia
Diego Ramirez
Oscar Pereira

VOLUNTAPP

Especificaciones Tecnicas

Objetivo

Desarrollar una plataforma web que conecte a individuos, organizaciones e instituciones interesadas en voluntariado y acción social, facilitando la creación de eventos, grupos y campañas. El proyecto busca fomentar la colaboración, la transparencia y el impacto positivo en áreas como salud, ecología y derechos humanos, mediante funcionalidades inspiradas en redes sociales, gamificación y sistemas de evaluación de reputación.

Áreas de aplicación:

- Voluntariado en salud, ecología, educación, derechos humanos y otras causas sociales.
- Coordinación entre ONGs, instituciones y voluntarios independientes.
- Promoción de eventos locales, virtuales y campañas de impacto comunitario.

El proyecto abarcará las siguientes funcionalidades clave:

- **Registro y gestión de perfiles de usuario**, con información personal, intereses y puntuación basada en retroalimentación comunitaria.
- **Interacción social**, incluyendo mensajería directa, creación de publicaciones, seguimiento entre usuarios y organización de eventos físicos/virtuales.
- **Sistema de puntuación y reputación**, para evaluar la fiabilidad de usuarios y grupos.
- **Creación y gestión de grupos temáticos** (salud, ecología, educación, etc.), con requisitos de reputación para liderar iniciativas.

- **Moderación automatizada y manual**, con sanciones para conductas inapropiadas y herramientas de reporte.
- **Gamificación**, mediante retos semanales, insignias y recompensas para incentivar la participación.
- **Integración con APIs externas** (Google Maps, Google Calendar, redes sociales) para geolocalización, gestión de eventos y promoción.
- **Campañas de financiación**, permitiendo recaudar fondos para proyectos sociales.

Visión General del Sistema

La plataforma VoluntApp operará como una red social centrada en voluntariado, donde usuarios (voluntarios, organizaciones e instituciones) interactuarán mediante perfiles personalizados, grupos temáticos y eventos. Los usuarios podrán:

- **Crear y unirse** a eventos de voluntariado físicos o virtuales, gestionados mediante calendarios integrados y geolocalización.
- **Interactuar** mediante mensajería directa, publicaciones en grupos y sistemas de retroalimentación para evaluar la reputación de otros miembros.
- **Participar en campañas de financiación** y retos gamificados para ganar insignias y puntos, incentivando la colaboración continua.
- **Acceder a herramientas de moderación** automática (IA para detección de contenido inapropiado) y manual (equipo humano), garantizando un entorno seguro.
  La arquitectura se basará en un **backend escalable** utilizando Laravel y un **frontend responsive** con Angular, con integración de APIs como Google Maps y redes sociales para funcionalidades avanzadas.

Limitaciones

Tiempo: Contando solo con 4 semanas se llevara a cabo una priorización de funcionalidades esenciales (registro, eventos, grupos) en la primera versión, postergando características secundarias (gamificación avanzada).

Presupuesto: Al ser un proyecto para el instituto, no contamos con un presupuesto, asi que todas las herramientas y plataformas a utilizar deben ser gratuitas.

Herramientas técnicas: Dependencia de APIs externas (Google Maps, Calendar), cuyas limitaciones de uso gratuito podrían afectar funcionalidades como geolocalización masiva.

Escalabilidad: Inicialmente, el sistema estará optimizado para una base de usuarios local; la expansión a nivel regional requerirá infraestructura adicional.

Suposiciones

Disponibilidad de servicios externos: Las APIs integradas (Google, redes sociales) mantendrán sus funcionalidades y accesibilidad durante el desarrollo.

Adopción de usuarios: Se asume un crecimiento gradual de la comunidad, con al menos 100 usuarios activos en los primeros 6 meses.

Cumplimiento legal: Los usuarios proporcionarán información verídica y la plataforma cumplirá con regulaciones de protección de datos (ej: GDPR).

Infraestructura: El servidor seleccionado (AWS) garantizará disponibilidad del 99% sin interrupciones críticas.

Requisitos Funcionales

1. Visualización de Posts

Permitir a los usuarios ver publicaciones de amigos, grupos y organizaciones que siguen en un feed principal.

Entrada: Datos de posts almacenados en la base de datos (contenido, autor, fecha).

Salida: Listado de posts ordenados cronológicamente en la interfaz.

Prioridad: Alta.

2. Creación de Posts

Permitir a los usuarios publicar contenido (texto, imágenes) en su perfil, grupos o eventos.

Entrada: Contenido del post, opcionalmente imágenes o etiquetas.

Salida: Post almacenado en la base de datos y visible en el feed correspondiente.

Prioridad: Alta.

3. Gestión de Comentarios y Likes

Permitir a los usuarios interactuar con posts mediante comentarios y likes.

Entrada: Texto del comentario o acción de like.

Salida: Comentario o like registrado en la base de datos y reflejado en la interfaz.

Prioridad: Alta.

4. Creación y Unión a Eventos

Permitir a los usuarios crear eventos de voluntariado o unirse a eventos existentes.

Entrada: Detalles del evento (título, descripción, fecha, ubicación).

Salida: Evento registrado en la base de datos y visible en la sección de eventos.

Prioridad: Alta.

5. Gestión de Amigos

Permitir a los usuarios enviar, aceptar o rechazar solicitudes de amistad.

Entrada: ID del usuario objetivo.

Salida: Relación de amistad almacenada en la base de datos y actualización de la lista de amigos.

6. Creación y Unión a Grupos

Permitir a los usuarios crear grupos temáticos (ej: ecología) o unirse a grupos existentes.

Entrada: Nombre del grupo, descripción y categoría.

Salida: Grupo registrado en la base de datos y visible en la sección de grupos.

Prioridad: Media.

7. Personalización de Perfil

Permitir a los usuarios editar y visualizar su perfil, incluyendo intereses, logros y foto.

Entrada: Datos actualizados del perfil (nombre, intereses, imagen).

Salida: Perfil modificado en la base de datos y reflejado en la interfaz.

Prioridad: Media.

8. Sistema de Reviews entre Usuarios

Permitir a los usuarios dejar reseñas sobre colaboraciones previas en eventos o proyectos.

Entrada: Puntuación (ej: 1-5 estrellas) y comentario opcional.

Salida: Review almacenada en la base de datos y visible en el perfil del usuario evaluado.

Prioridad: Baja.

9. Gestión de Eventos Suscritos

Permitir a los usuarios ver y gestionar eventos a los que se han unido.

Entrada: ID del usuario.

Salida: Listado de eventos suscritos en la interfaz.

Prioridad: Media.

10. Integración con Base de Datos

Almacenar y recuperar información de usuarios, posts, eventos, grupos y relaciones.

Entrada: Consultas estructuradas (ej: obtener posts de amigos).

Salida: Datos entregados al frontend para su visualización.

Prioridad: Alta.

11. Vista Responsive (Móvil, Tablet, Desktop)

Garantizar que la interfaz se adapte a diferentes dispositivos y tamaños de pantalla.

Entrada: Tipo de dispositivo detectado.

Salida: Interfaz optimizada para el dispositivo del usuario.

Prioridad: Media.

Requisitos No Funcionales

Escalabilidad

- **Objetivo**: Soportar hasta **100 usuarios concurrentes** en condiciones normales y escalar horizontalmente para manejar picos de hasta **200 usuarios concurrentes** mediante balanceadores de carga.
- **Herramientas**: Uso de **AWS Elastic Load Balancing** y bases de datos escalables (**Amazon RDS)**

Rendimiento

- **Tiempo de respuesta del servidor**:
  - Acciones básicas (cargar posts, ver perfil): **< 500 ms**.
  - Operaciones complejas (búsquedas, generación de feeds): **< 5 segundos**.
- **Optimización**: Cacheo de consultas frecuentes y compresión de assets en Angular.

Seguridad

- **Cifrado de datos**:
  - Contraseñas almacenadas con **bcrypt** (hash + salting).
  - Datos sensibles (Tokens de Auth JWT) cifrados en tránsito y en reposo.
- **Protección de APIs**:
  - Uso de **Laravel Sanctum** para autenticación basada en tokens.
  - Validación de solicitudes contra **SQL injection**, **XSS** y **CSRF** (protección integrada en Laravel).

Compatibilidad

- **Navegadores soportados**:
  - Chrome (últimas 3 versiones).
  - Firefox (últimas 3 versiones).
  - Edge (últimas 3 versiones).
  - Safari (iOS 14+, macOS Big Sur+).
- **Responsividad**: Diseño adaptativo en Angular para móvil (320px+), tablet (768px+) y desktop (1024px+).

Mantenibilidad

- **Código**:
  - Estructura modular en Laravel (patrón MVC) y Angular (componentes reutilizables).
  - Documentación técnica en GitHub y comentarios en código.
- **Logs y Monitoreo**:
  - Registro de errores y auditoría con **Laravel Logging.**
  - Monitoreo de rendimiento con **AWS CloudWatch**.

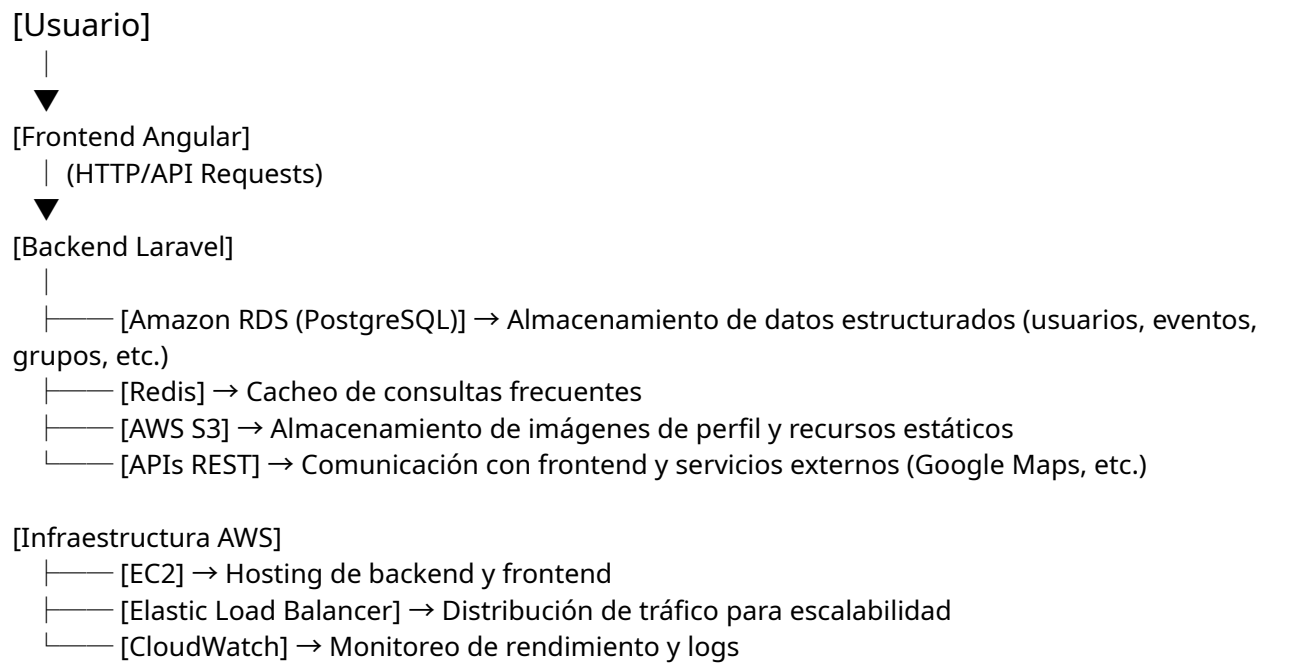
Disponibilidad

- **Objetivo**: **99.5% de uptime** anual.
- **Estrategias**:
  - Servidores en **AWS EC2** con redundancia en múltiples zonas.
  - Backups diarios en **Amazon S3** y recuperación ante desastres (RTO < 2 horas).

Integración Tecnológica

- **Backend**: Laravel 10.x con **Breeze** (autenticación) y **Sanctum** (APIs).
- **Frontend**: Angular 16+ con **Angular Material** para componentes UI.
- **Base de datos**: MySQL (relacional) en Amazon RDS.

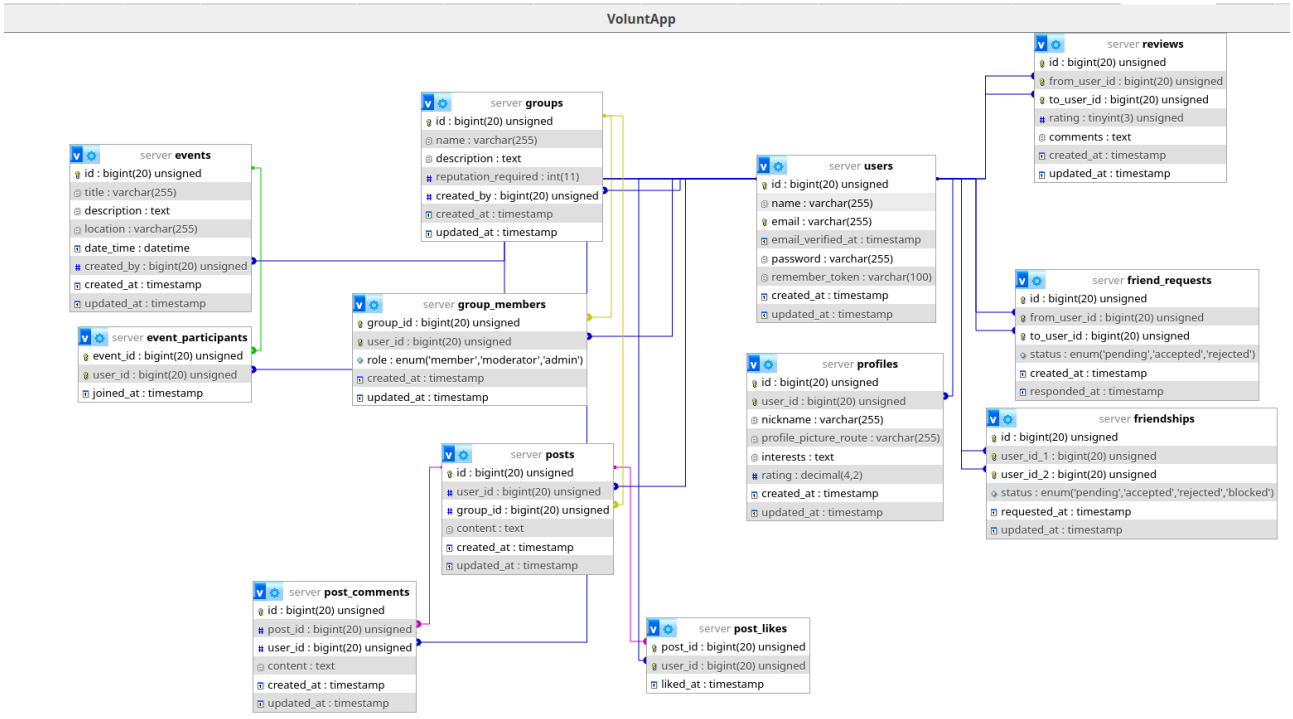
Diagrama de Arquitectura General



Flujo de Datos entre Módulos

1. **Usuario interactúa con Angular**: Realiza acciones como ver posts, unirse a eventos o enviar solicitudes de amistad.
2. **Angular envía solicitudes HTTP**: Ejemplo: GET /api/posts para obtener publicaciones.
3. **Laravel procesa la solicitud**: Autenticación mediante **Sanctum** (tokens JWT). Validación de datos y lógica de negocio (ej: crear un evento).
4. **Acceso a la base de datos**: Consultas a MySQL para CRUD (Crear, Leer, Actualizar, Eliminar).
5. **Respuesta al Frontend**: Datos en formato JSON (ej: lista de eventos).
6. **Almacenamiento en Laravel**: Subida de imágenes de perfil mediante APIs de Laravel.

Diagrama de Base de Datos (ERD)



Componentes Principales

- **Frontend (Angular)**:

Tecnologías: Angular 16+, Angular Material, RxJS.

Funcionalidad: Interfaz responsive para gestionar eventos, grupos, posts y perfiles.

Comunicación: Consume APIs RESTful de Laravel mediante servicios HTTP.

- **Backend (Laravel)**:

Tecnologías: Laravel 11, Sanctum (autenticación), Eloquent ORM.

APIs:

/api/events: Gestión de eventos (crear, unirse, listar).

/api/posts: Publicaciones y comentarios.

/api/friends: Solicitudes y relaciones de amistad.

- **Base de Datos (Amazon RDS)**:

Motor: MySQL.

Tablas principales: users, events, groups, posts, friendships.

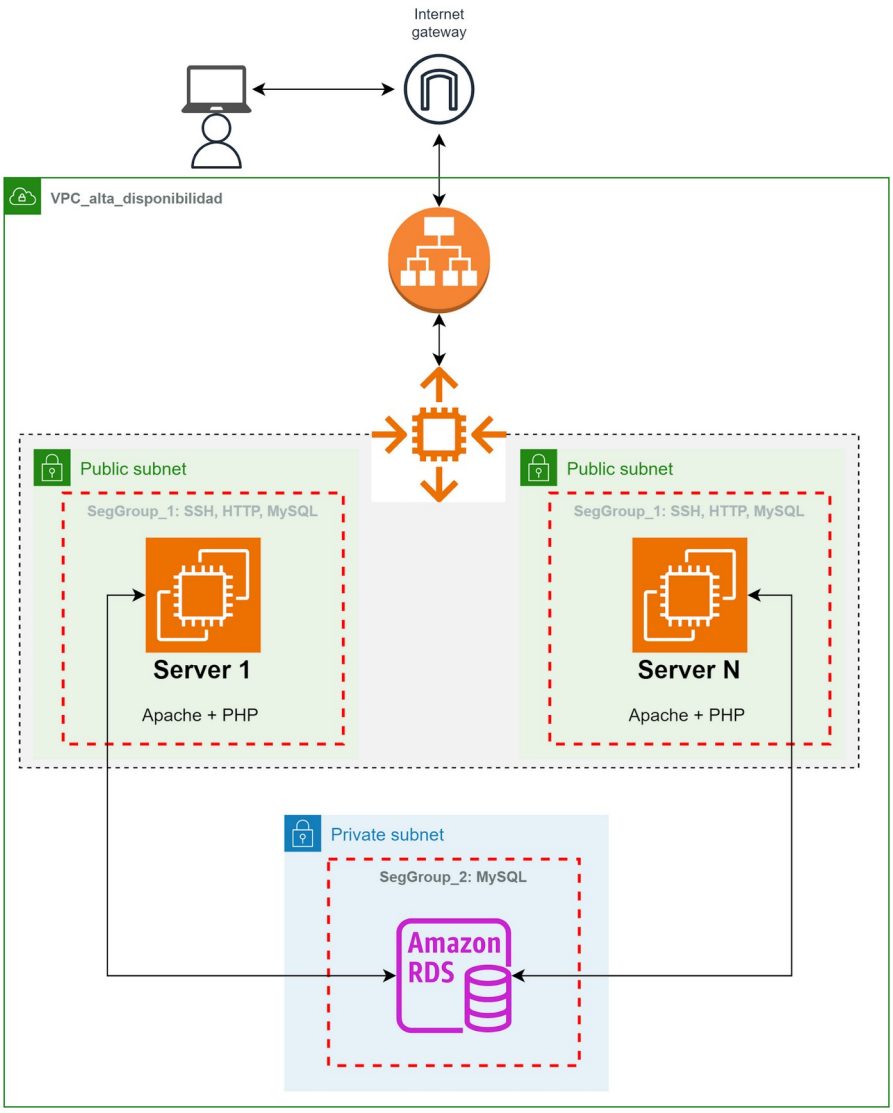
Optimización: Índices para consultas frecuentes (ej: busqueda de posts por id de creador).

- **Infraestructura (AWS)**:

EC2: Instancias para hospedar frontend y backend.

Elastic Load Balancer: Distribuye el tráfico entre instancias EC2.

Amazon S3: Almacena imágenes y archivos estáticos.



- **Seguridad**:

Protección de APIs: Sanctum (tokens CSRF y JWT), validación contra SQL injection.

Tecnologías y Herramientas

Lenguajes de Programación

Backend: PHP 8.x (Laravel).

Frontend: TypeScript (Angular), HTML5, CSS3/SASS.

Frameworks y Librerías

✔ **Backend**:

Laravel 11.x: Framework principal para el desarrollo del backend.

Laravel Breeze: Autenticación básica (registro, login, recuperación de contraseña).

Laravel Sanctum: Gestión de tokens para autenticación de APIs.

✔ **Frontend**:

Angular 16+: Framework para la interfaz de usuario.

Angular Material: Componentes UI predefinidos (botones, formularios, tarjetas).

RxJS: Manejo de flujos de datos reactivos.

Base de Datos

- **MySQL**: Motor principal para almacenamiento estructurado (usuarios, eventos, grupos).

Control de Versiones

- **Git**: Sistema de control de versiones distribuido.
- **GitHub**: Plataforma para alojar repositorios y gestionar colaboración (issues, pull requests).

Otras Herramientas

- **Infraestructura**:
    - **AWS EC2**: Servidores virtuales para hospedar backend y frontend.
    - **Amazon RDS**: Base de datos PostgreSQL administrada.
    - **Elastic Load Balancer**: Distribución de tráfico para escalabilidad.

- **Desarrollo y Testing**:
    - **Postman**: Pruebas de APIs y documentación.
- **Monitoreo y Logs**:
    - **AWS CloudWatch**: Monitoreo de rendimiento y logs en tiempo real.
- **Gestión de Dependencias**:
    - **Composer**: Para PHP (Laravel).
    - **npm**: Para TypeScript/Angular.

Requisitos de Pruebas

Pruebas Unitarias

Verificar el correcto funcionamiento de componentes individuales del sistema.

Herramientas:

Backend (Laravel): PHPUnit para probar modelos, controladores y rutas.

Ejemplo: Validar que un usuario con email único no pueda registrarse dos veces.

Frontend (Angular): Jest/Karma para probar componentes, servicios y pipes.

Ejemplo: Asegurar que el botón "Publicar" deshabilite si el post está vacío.

Cobertura mínima: 80% del código (backend y frontend).

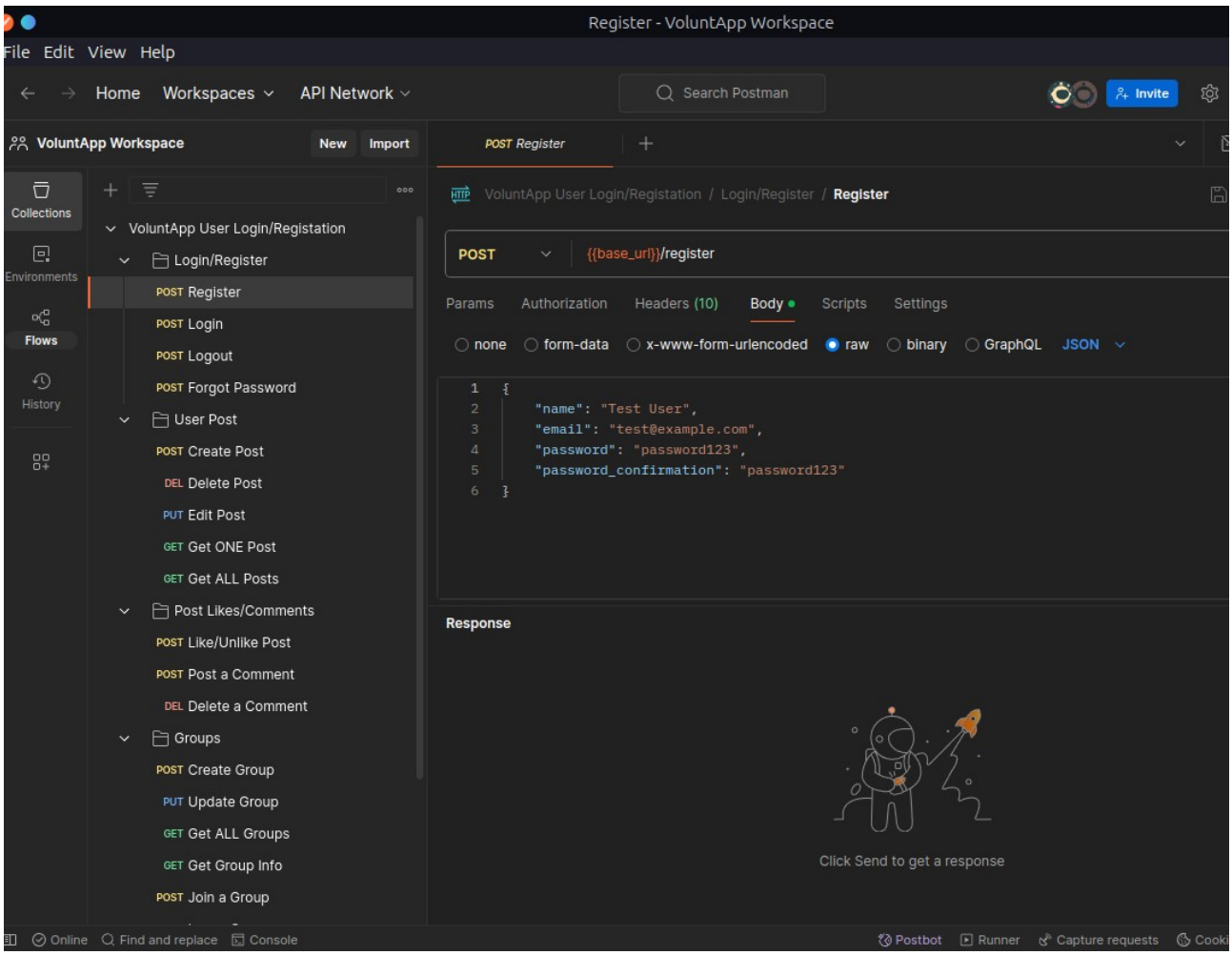
Pruebas de Integración

Objetivo: Validar la interacción entre módulos y servicios.

Herramientas:

Postman: Pruebas de APIs REST (endpoints de eventos, posts, amigos).

Ejemplo: Crear un evento y verificar que aparezca en el feed de usuarios suscritos.


Register - VoluntApp Workspace
File Edit View Help
Home Workspaces API Network
Search Postman
Invite
VoluntApp Workspace
New Import
Collections
Environments
Flows
History
VoluntApp User Login/Registation
Login/Register
POST Register
POST Login
POST Logout
POST Forgot Password
User Post
POST Create Post
DEL Delete Post
PUT Edit Post
GET Get ONE Post
GET Get ALL Posts
Post Likes/Comments
POST Like/Unlike Post
POST Post a Comment
DEL Delete a Comment
Groups
POST Create Group
PUT Update Group
GET Get ALL Groups
GET Get Group Info
POST Join a Group
POST Register
VoluntApp User Login/Registation / Login/Register / Register
POST
{{base_url}}/register
Params Authorization Headers (10) Body Scripts Settings
none form-data x-www-form-urlencoded raw binary GraphQL JSON
{
"name": "Test User",
"email": "test@example.com",
"password": "password123",
"password_confirmation": "password123"
}
Response
Click Send to get a response
Online Find and replace Console
Postbot Runner Capture requests

Escenarios clave:

Integración entre autenticación (Sanctum) y acceso a recursos protegidos.

Pruebas de Usuario

Objetivo: Validar la experiencia del usuario final y la usabilidad.

Tipos:

Pruebas de Usabilidad:

- **Participantes**:

  10-15 usuarios reales (voluntarios y organizaciones).

- **Tareas**:

  Crear un evento, unirse a un grupo y dejar un comentario.

  Navegar entre secciones en móvil y desktop.

Métricas: Tiempo de finalización, errores detectados, satisfacción (escala 1-5).

Pruebas de Aceptación:

Criterios: Cumplimiento de requisitos funcionales (ej: sistema de reviews entre usuarios).

Cronograma

El proyecto se organizará en **4 sprints de 1 semana (5 días laborales)** cada uno, priorizando funcionalidades críticas y entregando incrementos funcionales en cada iteración.

Sprint 1: Diseño y Prototipado

Definir arquitectura, wireframes y preparar entorno técnico.

- **Actividades**:
    - Diseño de wireframes en Figma (página principal, perfil, creación de eventos).
    - Configuración inicial del backend (Laravel + MySQL) y frontend (Angular).
    - Creación de repositorios en GitHub y documentación técnica.
- **Entregables**:
    - Prototipos interactivos en Figma.
    - Esquema inicial de la base de datos (ERD).
    - Entorno de desarrollo funcional (backend + frontend).

Sprint 2: Desarrollo Frontend Básico

Implementar las interfaces clave y funcionalidades básicas.

- **Actividades**:
    - Desarrollo del feed de posts, vista de eventos y perfil de usuario en Angular.
    - Integración con APIs de Laravel para autenticación (Sanctum) y gestión de datos.
    - Implementación de diseño responsive (móvil, tablet, desktop).
- **Entregables**:
    - Feed de posts funcional con likes y comentarios.
    - Formulario de creación de eventos integrado con Google Maps.
    - Perfil de usuario editable (intereses, foto).

Sprint 3: Integración Backend y Funcionalidades Avanzadas

Conectar frontend y backend, e implementar funcionalidades críticas.

- **Actividades**:
    - Desarrollo de APIs para grupos, amigos y reseñas (Laravel).
    - Implementación de gamificación (insignias y sistema de puntuación).
- **Entregables**:
    - Grupos temáticos con creación y membresía.
    - Sistema de amistad (solicitudes y listas de amigos).
    - Dashboard de reputación y logros.

Sprint 4: Pruebas y Despliegue

- **Objetivo**: Validar el sistema y desplegar en producción.
- **Actividades**:
    - Pruebas de carga (JMeter) y usabilidad (10 usuarios reales).
    - Corrección de errores y optimización de rendimiento.
    - Configuración de AWS (EC2, RDS, S3) y despliegue final.
- **Entregables**:
    - Plataforma desplegada en AWS con alta disponibilidad.
    - Reporte de pruebas y métricas de rendimiento.
    - Documentación técnica y manual de usuario.

Cada sprint incluirá:

Planificación: Revisión de objetivos y asignación de tareas.

Desarrollo: Implementación de funcionalidades.

Revisión: Demo interna y ajustes.

Retrospectiva: Análisis de mejoras para el siguiente sprint.

Glosario

Definición de términos técnicos y siglas clave utilizados en el proyecto:

- **API (Application Programming Interface)**: Conjunto de protocolos que permite la comunicación entre componentes de software. En VoluntApp, se usan para integrar servicios como Google Maps y autenticación.
- **Backend**: Parte del sistema que gestiona la lógica, bases de datos y APIs. Desarrollado en Laravel.
- **CRUD (Create, Read, Update, Delete)**: Operaciones básicas para gestionar datos (ej: eventos, usuarios).
- **Feed de posts**: Listado dinámico de publicaciones mostrado en la página principal.
- **Frontend**: Interfaz de usuario desarrollada en Angular.
- **Gamificación**: Mecánicas de juego (insignias, puntos) para incentivar la participación.
- **HTTP**: Protocolo para comunicaciones cliente-servidor. Usado en todas las APIs.
- **JWT (JSON Web Token)**: Estándar para autenticación basada en tokens. Utilizado por Laravel Sanctum.
- **ORM (Object-Relational Mapping)**: Técnica para mapear datos entre bases de datos y objetos de código. Eloquent ORM en Laravel.
- **Sanctum**: Paquete de Laravel para autenticación ligera con tokens.
- **Breeze**: Paquete de Laravel para implementar autenticación básica (registro, login).
- **GDPR (General Data Protection Regulation)**: Regulación europea de protección de datos. La plataforma cumple con sus requisitos.

Anexos

Documentación adicional y recursos útiles:

Enlaces clave

- **Laravel**: Documentación oficial
- **Angular**: Guía de inicio
- **AWS EC2**: Configuración de instancias
- **Google Maps API**: Integración en web

Herramientas de desarrollo

- **Figma**: Diseño de wireframes
- **Postman**: Pruebas de APIs

Referencias técnicas

- **Seguridad**: OWASP Top 10 (protección contra vulnerabilidades comunes).
- **GDPR**: Guía de cumplimiento

Repositorios

- **GitHub del proyecto**: VoluntApp🔗.