Cronograma Simplificado - Triqueta Digital

Período: 22 de Septiembre - 22 de Noviembre de 2025 (61 días)
Equipo: 5 personas
Entrega MVP: 2 de Noviembre de 2025
Entrega Final: 22 de Noviembre de 2025

Semana 0: Planificación y Diseño Inicial

22 - 28 de Septiembre de 2025

Tareas:

- Definir alcance completo del proyecto y requisitos funcionales
- Diseñar arquitectura SOA modular del sistema
- Crear modelo de datos PostgreSQL con todas las entidades
- Setup de repositorio Git con branching strategy
- Configurar Docker Compose para desarrollo local
- Escribir SRS completo (Software Requirements Specification)
- Crear diagrama de arquitectura y flujo de datos
- Definir convenciones de código (PEP 8, ESLint, Git commits)

Semana 1: Setup e Infraestructura

29 Sep - 5 Oct de 2025

Tareas:

- Crear proyecto FastAPI con estructura modular completa
- Configurar SQLAlchemy 2.0 (async) y Pydantic schemas
- Crear primera migración Alembic (tabla usuarios)
- Setup de Redis para caché y rate limiting
- Configurar logging, manejo de errores y middleware CORS
- Crear proyecto React + Vite con TypeScript
- Configurar Tanstack Router y Axios con interceptors
- Integrar TailwindCSS y Shadcn UI
- Crear componentes base (Button, Input, Card, Layout)
- Levantar PostgreSQL y Redis en Docker

Sprint 1: Autenticación y Usuarios

6 - 12 Oct de 2025

Tareas:

- Implementar modelos: User, UserProfile, RefreshToken
- Crear servicio de autenticación completo (auth_service.py)
- Implementar hash de passwords con bcrypt
- Generar y validar JWT con algoritmo RS256
- Crear endpoints: POST /auth/register, POST /auth/login
- Crear endpoints: POST /auth/refresh, POST /auth/logout

- Implementar rate limiting en endpoints de autenticación
- Crear endpoints de perfil: GET /users/me, PATCH /users/me
- Implementar contexto de autenticación en React (AuthContext)
- Crear páginas: /register, /login, /profile
- Configurar interceptor de Axios para JWT automático
- Implementar refresh token automático en frontend
- Crear protected routes con Tanstack Router
- Escribir 20+ tests unitarios de autenticación
- Escribir tests E2E de registro y login

Sprint 2: Módulo de Actividades - Parte 1

13 - 19 Oct de 2025

Tareas:

- Crear modelo Activity con todos los campos
- Implementar servicio activity_service.py
- Implementar búsqueda full-text en PostgreSQL
- Crear endpoints CRUD: POST, PUT, DELETE /actividades (admin only)
- Crear endpoint: GET /actividades (con filtros y paginación)
- Crear endpoint: GET /actividades/{id}
- Crear endpoint: GET /actividades/search
- Crear servicio API activities.ts en frontend
- Implementar hooks de React Query (useActivities, useActivity)
- Crear componente ActivityCard
- Crear página /actividades con listado y paginación
- Crear componente SearchBar
- Crear página /actividades/$id con detalle completo
- Escribir 25+ tests unitarios

Sprint 3: Módulo de Actividades - Parte 2 + Admin

20 - 26 Oct de 2025

Tareas:

- Implementar servicio de importación (activity_import_service.py)
- Crear parser de CSV con validación completa
- Crear parser de JSON con validación completa
- Implementar detección de duplicados (título + fecha + ubicación)
- Crear endpoint: POST /admin/actividades/import
- Implementar estados de actividades: activa, pendiente_validacion, rechazada
- Crear endpoints: POST /admin/actividades/{id}/aprobar y /rechazar
- Crear página /admin/actividades con tabla de gestión
- Implementar ActivityForm completo (crear/editar)
- Crear hooks: useCreateActivity, useUpdateActivity, useDeleteActivity
- Implementar componente ActivityFilters (Sheet con todos los filtros)
- Implementar upload de archivos CSV/JSON
- Crear feedback visual de importación (resumen de errores)
- Agregar enlace “Administración” en Navbar (solo visible para admins)
- Escribir 15+ tests de importación y validación

- Tests E2E de CRUD completo desde admin panel

Sprint 4: Favoritos + Recomendaciones IA

27 Oct - 2 Nov de 2025 / MVP COMPLETO

Tareas:

- Crear modelo Favorite con relación usuario-actividad
- Crear endpoints: POST /favoritos, GET /favoritos, DELETE /favoritos/{id}
- Actualizar popularidad_favoritos al guardar/eliminar
- Implementar servicio de recomendaciones (recommendation_service.py)
- Crear algoritmo híbrido (popularidad + etiquetas + localidad + disponibilidad)
- Crear endpoint: GET /recomendaciones
- Generar explicaciones personalizadas de recomendaciones
- Implementar caché de recomendaciones en Redis
- Crear job para recalcular popularidad_normalizada diariamente
- Crear servicio favorites.ts en frontend
- Implementar hooks: useFavorites, useAddFavorite, useRemoveFavorite
- Crear componente FavoriteButton
- Integrar FavoriteButton en ActivityCard
- Crear página /favoritos con listado de favoritos del usuario
- Crear servicio recommendations.ts en frontend
- Crear hook useRecommendations
- Crear página /recomendaciones con explicaciones
- Escribir 22+ tests de favoritos y recomendaciones
- Tests E2E del flujo completo de favoritos
- **Deploy del MVP a ambiente staging**

Sprint 5: Dashboard de Administración

3 - 9 Nov de 2025

Tareas:

- Crear endpoint: GET /admin/dashboard con todas las métricas
- Implementar queries optimizadas para métricas agregadas
- Calcular métricas: usuarios totales, activos, nuevos
- Calcular métricas: actividades por localidad y por tipo
- Implementar caché de 5 minutos para dashboard
- Crear endpoints de gestión de usuarios (admin)
- Crear endpoint: GET /admin/usuarios (listado con filtros)
- Crear endpoints: PATCH /admin/usuarios/{id}/rol y /estado
- Calcular top 10 actividades más populares
- Calcular top 10 etiquetas más usadas
- Crear página /admin/dashboard en frontend
- Integrar librería de gráficos (Recharts)
- Implementar gráfico de torta: actividades por localidad
- Implementar gráfico de barras: actividades por tipo
- Crear cards con métricas clave
- Crear página /admin/usuarios con tabla y acciones
- Implementar diseño responsive del dashboard

- Escribir 10+ tests de dashboard y gestión de usuarios
- Tests E2E del panel de administración completo

Sprint 6: ETL + Importación Automatizada

10 - 16 Nov de 2025

Tareas:

- Crear modelo ETLExecution para logs de procesos
- Crear endpoints: GET /admin/etl/status, POST /admin/etl/run
- Crear endpoint: GET /admin/etl/executions/{id}/logs
- Implementar SSE o polling para logs en tiempo real
- Crear script ETL completo en directorio /etl/src/
- Implementar extractor de API IDRD
- Implementar extractor de Portales Distritales (CSV/JSON)
- Implementar transformer para normalización de datos
- Implementar loader para inserción en PostgreSQL
- Agregar detección de duplicados en ETL
- Implementar logging completo del proceso ETL
- Crear Dockerfile para script ETL
- Configurar Docker Compose para ejecutar ETL
- Crear página /admin/etl en frontend
- Visualizar estado de última ejecución de ETL
- Implementar botón para ejecutar ETL manual
- Mostrar logs en tiempo real durante ejecución
- Crear página /admin/validacion para actividades pendientes
- Implementar acciones: aprobar, editar+aprobar, rechazar
- Escribir 12+ tests del pipeline ETL completo
- Tests E2E de ejecución y validación de ETL

Sprint 7: Testing Final y Deployment

17 - 22 Nov de 2025 / ENTREGA FINAL

Tareas:

- Code review completo de todo el backend
- Code review completo de todo el frontend
- Refactoring y optimizaciones de código
- Validar todos los índices de base de datos
- Implementar rate limiting en todos los endpoints públicos
- Performance testing completo de API
- Optimizar queries lentas identificadas
- Implementar caché en endpoints frecuentes
- Validar cobertura de tests >80% backend, >70% frontend
- Optimizar bundle size del frontend (code splitting)
- Implementar lazy loading de componentes pesados
- Implementar lazy loading de imágenes
- Testing de accesibilidad completo (WCAG 2.1 AA)
- Testing en múltiples navegadores (Chrome, Firefox, Safari, Edge)
- Testing en múltiples dispositivos (desktop, tablet, mobile)

- Validar Lighthouse Performance >90 (desktop), >80 (mobile)
- Validar Lighthouse Accessibility >95
- Suite completa de tests E2E con Playwright
- Load testing con 200 usuarios concurrentes
- Security scan (SAST con Bandit/Semgrep)
- Dependency security check
- Configurar CI/CD completo en GitHub Actions
- Setup de servidor de producción
- Configurar monitoreo y alertas
- Configurar backups automáticos de base de datos
- **Deploy final a producción**
- Actualizar toda la documentación (README, API, DEPLOYMENT)
- Crear video demo del proyecto (5-10 min)
- Preparar presentación final del proyecto
- Realizar retrospectiva del equipo