Informe Técnico de Avance – Fase 2 (Enriquecido)

Proyecto: SynapQuest – Plataforma de aprendizaje adaptativo para TDAH

Asignatura: Capstone (PTY4614)

Equipo: Franco Constanzo, Fernanda Marin, Javiera Bustamante

Fecha: 10/26/2025

# Índice

## Abstract (English)

SynapQuest is an adaptive learning platform designed to support students with ADHD by delivering short, motivating study activities that adjust to the learner’s attention level and progress. The project focuses on three pillars: (1) continuous attention and performance assessment through lightweight interaction metrics and brief questionnaires; (2) an adaptive recommendation engine to select the best content modality (videos, concise summaries, interactive exercises); and (3) gamification elements (XP, coins, levels, streaks) to sustain engagement. During Phase 2 we worked under Scrum with time‑boxed sprints instead of a Gantt chart, producing a functional MVP: a Node/Express + MongoDB backend with validated schemas (users, subjects, materials, activities, attempts, XP events), a React/TypeScript UI for student flows (subjects and activities) and an admin console (dashboard, ranking, materials management). Evidence includes mockups, API endpoints, and integrated views. Early usability checks confirm the value of short tasks, immediate feedback and visual progress tracking for ADHD needs. Upcoming work prioritizes a basic attention assessment, refinement of the recommendation logic, and a structured evaluation with user testing to measure retention, satisfaction and learning gains.

## Resumen (Español)

SynapQuest es una plataforma de aprendizaje adaptativo orientada a estudiantes con TDAH, que ofrece actividades cortas y motivadoras ajustadas al nivel de atención y progreso del usuario. El proyecto se apoya en tres pilares: (1) evaluación continua de la atención y del desempeño mediante métricas ligeras de interacción y cuestionarios breves; (2) un motor de recomendación para elegir la modalidad de contenido más adecuada (videos, resúmenes concisos, ejercicios interactivos); y (3) gamificación (XP, monedas, niveles, rachas) para mantener la motivación. En la Fase 2 trabajamos con Scrum y sprints en lugar de carta Gantt, entregando un MVP funcional: backend Node/Express + MongoDB con esquemas validados (usuarios, materias, materiales, actividades, intentos, eventos de XP), interfaz React/TypeScript para flujos de estudiante y un panel de administración (dashboard, ranking, gestión de materiales). Las evidencias incluyen mockups, endpoints de API y vistas integradas. Las primeras pruebas de usabilidad confirman el valor de tareas breves, retroalimentación inmediata y seguimiento visual del progreso. El trabajo próximo prioriza una evaluación atencional básica, el refinamiento del motor de recomendación y una evaluación con usuarios para medir retención, satisfacción y aprendizaje.

## Evidencias de avance (Phase 2)

- Esquema de datos en MongoDB con validación $jsonSchema (users, subjects, materials, activities, activity\_attempts, xp\_events).  
- Mockups navegables: login/registro, selección de tipo de TDAH, materias/actividades, y panel administrativo.  
- MVP backend (Node/Express + MongoDB) con endpoints iniciales y autenticación básica.  
- UI React/TypeScript para materias, actividades y consola admin (dashboard, ranking, gestión de materiales).  
- Integración de gamificación: XP, monedas y niveles visibles en interfaz.  
- Pruebas de humo y revisiones por sprint con ajustes derivados de retroalimentación.

## Metodología aplicada

Aplicamos Scrum con sprints de una a dos semanas, ceremonias de dailies, reviews y retrospectivas. No utilizamos carta Gantt; la planificación se gestionó en backlog priorizado, con tableros Kanban, criterios de aceptación por historia y definición de terminado (DoD) mínima (lint, type‑check, prueba de humo y documentación breve). Para mitigar dependencias entre frontend y backend, adoptamos versionado simple de endpoints y checklist de breaking changes en PRs.

## Ajustes realizados y justificación

- Separación de rutas públicas y de administración para mejorar seguridad y claridad de consumo en frontend.  
- Refactor del modelo de XP (eventos con origen: activity, streak, admin\_adjust, purchase) para trazabilidad y métricas.  
- Normalización de banners/materiales y manejo de archivos grandes (validación de tamaño/MIME) tras errores 422/500.  
- Priorización de una evaluación atencional básica antes de un motor ML completo, para obtener evidencia temprana.  
- Documentación ligera (ADR breves) y acuerdos de contrato API para reducir retrabajo.

# Desarrollo de ingeniería

## 1. Visión general y alcance del MVP

El MVP cubre: (a) autenticación básica y perfil; (b) exploración de materias y actividades; (c) consola administrativa para dashboard, ranking y gestión de materiales/actividades; y (d) trazabilidad de XP/monedas. El foco es validar la hipótesis TDAH con actividades breves, retroalimentación inmediata y seguimiento de progreso.

## 2. Arquitectura y stack

- Frontend: React + TypeScript (vistas de estudiante y panel admin).  
- Backend: Node.js (Express) con API REST.  
- Base de datos: MongoDB con validación $jsonSchema.  
- Control de versiones: GitHub; gestión: tablero Kanban y sprints.  
- Entorno: desarrollo local; integración front–back por endpoints REST.

## 3. Modelo de datos (MongoDB)

Colecciones principales:  
• users: perfil, rol (student/admin), nivel, xp, coins, streaks y progreso por materia.  
• subjects: nombre, descripción, slug y banner.  
• materials: PDF/Video/PPT/Link asociados a materias; orientación TDAH opcional.  
• activities: tipos (multiple\_choice, true\_false, video\_quiz, ppt\_review, embedded\_quiz), xpReward y config.  
• activity\_attempts: score, conteos, duración y xpAwarded por usuario/actividad.  
• xp\_events: ledger de XP/coins (source: activity, streak, admin\_adjust, purchase).

## 4. Backend (API REST)

Principios:  
• Rutas públicas vs. administrativas separadas.  
• Autenticación con sesiones/tokens y recuperación de contraseña.  
• Validación de payloads y manejo de errores (422/500) con mensajes claros.  
• Subida de archivos con límites de tamaño/MIME y normalización de URLs.  
  
Endpoints ejemplo (no exhaustivo):  
GET /subjects (catálogo público), GET /admin/subjects (gestión admin), POST /admin/materials (alta de materiales), GET /admin/students (listado y métricas), POST /auth/forgot-password, POST /auth/reset-password.

## 5. Frontend (vistas clave)

• Estudiante: login/registro, selección de tipo TDAH, materias, actividades con feedback inmediato y barra de progreso.  
• Admin: dashboard con evolución de XP semanal y actividad reciente, ranking de estudiantes, gestión de materias y materiales.

## 6. Gamificación y métricas

Sistema de niveles, XP y monedas con registro de eventos; rachas (streaks) y progreso por materia. El dashboard presenta XP semanal y actividad reciente para seguimiento operativo.

## 7. Metodología aplicada (Ágil, sin Gantt)

Trabajamos con metodología ágil (Scrum) mediante sprints de 1–2 semanas, dailies, reviews y retrospectivas. No utilizamos carta Gantt; la planificación se gestionó con backlog priorizado y tablero Kanban. Se adoptó Definition of Done mínima (lint, type‑check, prueba de humo y notas de cambios). Para mitigar dependencias entre frontend y backend, usamos versionado simple de endpoints y checklist de breaking changes en PRs.

## 8. Evidencias de avance

- Esquema de datos en MongoDB con validación $jsonSchema (users, subjects, materials, activities, activity\_attempts, xp\_events).  
- Mockups navegables de flujos clave: login/registro, selección de TDAH, materias/actividades y panel administrativo.  
- MVP backend (Node/Express + MongoDB) con endpoints iniciales y autenticación básica.  
- UI React/TypeScript para materias, actividades y consola admin (dashboard, ranking, gestión de materiales).  
- Integración de gamificación (XP, monedas, niveles) visible en la interfaz.  
- Pruebas de humo y revisiones por sprint con ajustes derivados de retroalimentación.

## 9. Ajustes realizados y justificación

- Separación de rutas públicas y de administración para mejorar seguridad/claridad en frontend.  
- Refactor del modelo de XP (orígenes: activity, streak, admin\_adjust, purchase) para trazabilidad y métricas.  
- Normalización de banners/materiales y manejo de archivos grandes (validación tamaño/MIME) tras errores 422/500.  
- Priorización de evaluación atencional básica antes de un motor ML completo, para obtener evidencia temprana.  
- Documentación ligera (ADRs breves) y acuerdos de contrato API para reducir retrabajo.

## 10. Calidad y pruebas

Pruebas de humo por flujos críticos (autenticación, carga de archivos, creación de actividades). Semillas de datos para demos, checklists de QA por historia y control de breaking changes en PRs.

## 11. Próximas iteraciones

• Módulo básico de evaluación atencional (cuestionarios + telemetría ligera).  
• Recomendación por reglas/heurísticas y posterior refinamiento.  
• Pruebas con usuarios para métricas de retención/satisfacción y aprendizaje.  
• Mayor cobertura de tests y documentación ligera (ADRs).

# Conclusiones

A lo largo de la Fase 2, Scrum nos ayudó a mantener el enfoque en el valor incremental y la retroalimentación temprana. La lección principal fue la importancia de los contratos de API explícitos para evitar la repetición del trabajo durante la integración. Las tareas de aprendizaje cortas y bien delimitadas con retroalimentación inmediata benefician claramente las necesidades de los usuarios con TDAH; este principio guió tanto la microcopia de la interfaz de usuario como el diseño de la interacción. El panel de administración resultó útil para validar los flujos de datos (XP, intentos de actividad, rachas) y para mostrar métricas operativas para la toma de decisiones. A continuación, reforzaré las comprobaciones automatizadas (pruebas de pelusa, tipo, humo), finalizaré un módulo de evaluación de atención mínima y alinearé la línea base de recomendaciones con resultados de retención medibles. Finalmente, seguiremos evitando un diagrama de Gantt rígido en favor de ciclos de inspección y adaptación, que han sido más efectivos para la incertidumbre en esta etapa.

## Conclusions (English – individual)

Throughout Phase 2, Scrum helped us maintain focus on incremental value and early feedback. The main lesson was the importance of explicit API contracts to prevent rework during integration. Short, well‑bounded learning tasks with immediate feedback clearly benefit ADHD needs; this principle guided both UI microcopy and interaction design. The admin dashboard proved useful to validate data flows (XP, activity attempts, streaks) and to surface operational metrics for decision‑making. Next, I will reinforce automated checks (lint, type, smoke tests), finalize a minimum attention‑assessment module and align the recommendation baseline with measurable retention outcomes. Finally, we will keep avoiding a rigid Gantt in favor of inspect‑and‑adapt cycles, which has been more effective for uncertainty at this stage.

## Reflection (English – individual)

I adjusted scope and sequencing based on blockers and teacher feedback: we prioritized a basic attention assessment and schema stability over early, complex ML. This decision reduced risk and improved our ability to demonstrate value in each review. I also learned to communicate technical changes through small, traceable PRs that include acceptance criteria and a clear impact note for the frontend. Evidence gathered so far (working schemas, integrated screens, dashboard metrics) indicates we are on the right track. For Phase 3, I will focus on writing lightweight ADRs, improving test coverage for critical flows (auth, file upload, activity attempts), and refining our Kanban WIP limits to keep cycle times short. Using sprints instead of a Gantt chart has increased adaptability and supported continuous learning, which is essential for building an ADHD‑aware learning platform.

# Reflexiones individuales

## Integrante 1

Reflection (English): I adapted scope and sequencing based on blockers and feedback. Prioritizing a basic attention assessment over early complex ML reduced risk and improved our ability to show value in reviews. I will focus on strengthening automated checks (lint, type, smoke tests), documenting decisions with short ADRs, and keeping WIP limits tight to maintain flow.

## Integrante 2

Reflection (English): I adapted scope and sequencing based on blockers and feedback. Prioritizing a basic attention assessment over early complex ML reduced risk and improved our ability to show value in reviews. I will focus on strengthening automated checks (lint, type, smoke tests), documenting decisions with short ADRs, and keeping WIP limits tight to maintain flow.

## Integrante 3

Reflection (English): I adapted scope and sequencing based on blockers and feedback. Prioritizing a basic attention assessment over early complex ML reduced risk and improved our ability to show value in reviews. I will focus on strengthening automated checks (lint, type, smoke tests), documenting decisions with short ADRs, and keeping WIP limits tight to maintain flow.