

| **1. Resumen avance Proyecto APT** |
| --- |

| Resumen de avance proyecto APT | Durante este periodo se ha avanzado significativamente en el desarrollo del proyecto PsicoLink, una plataforma orientada a la gestión de atención psicológica en línea. Se han generado y desarrollado las vistas principales que compondrán la aplicación final, priorizando la implementación de una funcionalidad clave: el módulo de agendamiento de citas, el cual ya incorpora validaciones y reglas de negocio en el servidor, garantizando la integridad de los datos y evitando solapamientos entre sesiones.  Además, se estableció la conexión completa entre el frontend (Ionic/Angular) y el backend (Django REST Framework), logrando la autenticación de usuarios mediante JWT, el registro y login funcional, y la gestión de citas con validaciones según el rol del profesional (psicólogo, terapeuta, psiquiatra, etc.). El sistema actualmente cuenta con una versión preliminar en producción, lo que permite visualizar y probar las funcionalidades principales del flujo de usuario. |
| --- | --- |
| Objetivos | Respecto a los objetivos, no ha sido necesario modificarlos, ya que el desarrollo se mantiene alineado con los propósitos iniciales de ofrecer una solución que conecte pacientes con profesionales de salud mental mediante una experiencia segura, moderna y eficiente. |
| Metodología | Se ajustó la metodología incorporando un enfoque más iterativo con Sprints cortos y revisiones funcionales al final de cada uno, lo que ha mejorado la capacidad de integración continua y la validación temprana de los entregables. |
| Evidencias de avance | Las evidencias que se presentan en este informe permiten demostrar el avance real y tangible del proyecto, validando la correcta aplicación de la metodología y el cumplimiento progresivo de los objetivos específicos.  **Evidencias presentadas:**   1. **Historias de Usuario y Product Backlog**    * Se detalla el conjunto de funcionalidades priorizadas y sus criterios de aceptación.    * Permite evidenciar la trazabilidad entre requerimientos y entregables. 2. **Sprint Backlog**    * Registro de tareas realizadas durante cada iteración.    * Muestra la planificación, desarrollo e incremento alcanzado por Sprint. 3. **Incremento de Producto**    * Se incluye la aplicación funcional con los módulos de registro, login y agendamiento operativo.    * Validaciones de negocio implementadas en el servidor (evita solapamiento de citas y controla duraciones según rol). 4. **Modelo de Datos y Diagrama de Componentes**    * Refleja la estructura lógica del sistema, incluyendo relaciones entre entidades como usuarios, profesionales y citas. 5. **Código Fuente (Frontend y Backend)**    * Implementado con Angular/Ionic y Django REST Framework, integrando JWT y PostgreSQL. 6. **Pruebas Automatizadas (en desarrollo)**    * Se están implementando pruebas unitarias y de integración para garantizar la fiabilidad del sistema y prevenir regresiones.   **Justificación:** Estas evidencias demuestran la consolidación del producto en etapas clave del ciclo de desarrollo. Además, reflejan el uso de buenas prácticas técnicas y metodológicas, aplicando pruebas, control de versiones, estándares de calidad y despliegue funcional en entorno real. |

| **2. Monitoreo del Plan de Trabajo** |
| --- |

| Plan de Trabajo | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Competencia o unidades de competencias | Actividades | Recursos | Duración de la actividad | Responsable[[1]](#footnote-0) | Observaciones | Estado de avance | Ajustes |
| Gestionar proyectos informáticos, ofreciendo alternativas para la toma de decisiones de acuerdo a los requerimientos de la organización. | Acta de Constitución de Proyecto | -Plantilla de Acta de Constitución  -Reunión de kick-off | Sprint 1 | Lucas Cisternas, Fernando Cavada, Joaquín Madariaga | Dificultades: Posible falta de alineación inicial sobre el alcance. Facilitadores: La plantilla estandarizada guía la discusión y asegura que se cubran todos los puntos críticos. | Completado |  |
| Gestionar proyectos informáticos, ofreciendo alternativas para la toma de decisiones de acuerdo a los requerimientos de la organización. | Roadmap | -Herramientas (Jira, Confluence, Miro)  -Visión del Producto | Sprint 1 | Lucas Cisternas, Fernando Cavada, Joaquín Madariaga | Dificultades: definir los tiempos de cada tarea para cumplir en el tiempo y al mismo tiempo dando un margen de error | Completado | Las tareas y orden de estas fueron modificadas |
| Ofrecer propuestas de solución informática analizando de forma integral los procesos de acuerdo a los requerimientos de la organización. | Levantamiento inicial de requerimientos y definición del Product Backlog | -Herramientas (Jira, Trello, Excel)  -Técnicas de entrevista | Sprint 1 | Lucas Cisternas, Fernando Cavada, Joaquín Madariaga | Dificultades: Requerimientos ambiguos o contradictorios. | Completado |  |
| Desarrollar una solución de software utilizando técnicas que permitan sistematizar el proceso de desarrollo y mantenimiento, asegurando el logro de los objetivos. | Preparacion e Implementacion ambiente de Desarrollo | -IDEs (Android Studio)  -Repositorio de código (GitHub) | Sprint 1 | Fernando Cavada, Joaquín Madariaga | Dificultades: Conflictos de configuración entre entornos o dependencias. | Completado |  |
| "Construir modelos de datos y el modelo arquitectónico de una solución sistémica que soporte los procesos de negocio de la organización, de acuerdo a un diseño definido, escalable en el tiempo y cumpliendo los estándares de la industria." | Construcción BD inicial + Mockups | -DBMS PostgreSQL, Supabase  -Diagramas: modelo de datos  -Herramienta: Figma | Sprint 2 | Fernando Cavada | Riesgo: posibles cambios en requerimientos que obligan a reestructurar tablas. Se recomienda validación temprana en Sprint Review.  Facilitadores: Figma te permite diseñar mockups interactivos. | Completado |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Funcionalidad mínima: login de usuario | -Proyecto de la aplicación creado (templates de inicio)  -Modelo de datos de la base de datos de usuarios.  -Base de datos construida | Sprint 2 | Joaquín Madariaga | Dificultades: Riesgo de bloqueo de cuentas por malas prácticas y es complejo implementar medidas de seguridad robustas (ej. encriptación y prevención de ataques)  Facilitadores: La implementación de un ambiente de pruebas sólido y la metodología Scrum para gestionar la complejidad. | Completado |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | API Django básica | -Base de datos PostgreSQL, Supabase.  -Frontend desarrollado (con al menos templates de inicio)  -Herramientas de pruebas: Jasmine y karma. | Sprint 2 | Joaquín Madariaga | Dificultades: Diseñar endpoints consistentes y seguros, especialmente al manejar datos sensibles.  Facilitadores: La documentación y Django REST Framework ya provee herramientas que aceleran la construcción de APIs confiables. | Completado |  |
| Programar consultas o rutinas para manipular información de una base de datos de acuerdo a los requerimientos de la organización. | Pruebas unitarias de login | -Infraestructura basica completa (frontend básico + base de datos para usuarios)  -Pytest | Sprint 2 -> Sprint 4 | Lucas Cisternas | Riesgo: tiempo adicional si surgen errores de normalización. | Retrasado | Se retrasaron hacia el cuarto sprint para priorizar la integración de una funcionalidad clave del sistema |
| Programar consultas o rutinas para manipular información de una base de datos de acuerdo a los requerimientos de la organización. | Ampliación BD (profesionales) | -Base de datos construida  -Modelo de datos | Sprint 3 | Joaquín Madariaga | Puede requerir ajustar la entidad anteriormente creada para usuarios generales. | Completado |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Funcionalidad: búsqueda y filtrado de profesionales | -Frontend en Angular/Ionic.  -Consultas optimizadas para la Base de datos PostgreSQL. | Sprint 3 -> Sprint 4 | Fernando Cavada | Dificultades: Riesgo de tener consultas mal optimizadas que empeoren gravemente la experiencia de usuario  Facilitadores: Una buena definición de los filtros y el uso de Django ORM para facilitar la implementación de estos. | En curso / Retrasado | El filtrado de profesionales se movió para el cuarto sprint. |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Validación de usuarios (Roles) | -• Django REST Framework (SimpleJWT, permissions)  • Base de datos PostgreSQL (tablas de Users, Groups, Permissions) | Sprint 3 -> Sprint 4 | Joaquín Madariaga | Dificultades: Garantizar que la jerarquía de permisos se aplique consistentemente en todos los endpoints y vistas. Riesgo de fugas de datos si un rol accede a información restringida de otro. Facilitadores: Los grupos y permisos integrados de Django ofrecen una base sólida para la gestión de accesos. | En curso / retrasado | Se retrasaron hacia el cuarto sprint para priorizar la integración de una funcionalidad clave del sistema |
| Desarrollar e implementar servicios web que permitan la integración y consulta de información, asegurando eficiencia y escalabilidad. | API Django para búsqueda | Framework Django  - Django Rest Framework (DRF)  - Documentación de la API REST | Sprint 3 | Fernando Cavada | Dificultades: Posible complejidad en la definición de filtros avanzados y optimización de consultas.  Facilitadores: Uso de Django Rest Framework acelera la creación de endpoints y manejo de búsquedas. | Completado |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Pruebas unitarias + integración básica | • Framework pytest + pytest-django  • Biblioteca unittest de Django  • Factory Boy para datos de prueba | Sprint 3 |  | Dificultades: Puede consumir tiempo de desarrollo inicial significativo. Facilitadores: A largo plazo, las pruebas automatizadas ahorran tiempo, previenen regresiones y facilitan los despliegues confiables. Herramientas como Pytest y la TestSuite de Django son de gran ayuda. | En curso | Se combinaron con las pruebas de validación de usuarios y las pruebas de login. |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Ampliación BD (citas y mensajes) | • PostgreSQL DBMS  • Django ORM (models, migrations) | Sprint 4 | Joaquín Madariaga | Dificultades: Mantener la integridad referencial y consistencia de los datos al modelar relaciones complejas (ej: estado de una cita, historial de mensajes). Facilitadores: El ORM de Django simplifica la creación de estos modelos y sus relaciones, automatizando constraints a nivel de base de datos. | En curso |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación | Funcionalidades: Agendamiento de citas + chat basico | -Backend en Django  -Base de datos (tablas de agenda)  -Calendario en frontend (Angular/Ionic),  -Librerías de gestión de tiempo/fechas. | Sprint 4 | Joaquín Madariaga | Dificultades: se deben evitar conflictos de horarios y asegurar disponibilidad en tiempo real, el chat puede consumir más recursos del equipo del esperado.  Facilitadores: existen librerías de calendario y validación de fechas que pueden integrarse fácilmente. Ademas Supabase ofrece soporte nativo implementar un chat en tiempo real, con buena documentación y ejemplos de integración. | En curso / Retrasado | La funcionalidad completa de agendamiento se retrasó unos días por priorizar el funcionamiento del login, para solventar esto, se priorizará esta funcionalidad antes que las pruebas del login. |
| Asegurar la calidad del software mediante la validación del código, pruebas automatizadas y verificación de la integración entre componentes. | Pruebas unitarias e integración | - Framework de pruebas (PyTest / unittest)  - Cobertura de código  - GitHub  - Repositorio en GitHub/GitLab | Sprint 4 | Lucas Cisternas | Dificultades: Posible falta de cobertura en escenarios críticos y errores en la integración entre módulos.  Facilitadores: Automatización de pruebas y pipeline CI/CD permiten detectar fallas rápidamente. | Retrasado / en curso | Se ajustaron las pruebas a solo la funcionalidad de agendamiento, retrasando las pruebas al sistema de chat para el quinto sprint |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Historial de consultas | -PostgreSQL (tablas para registrar consultas).  -Django ORM.  -frontend para visualización. | Sprint 4 | Lucas Cisternas | Dificultades: Riesgo de vulnerabilidades en el almacenamiento de datos sensibles.  Facilitadores: El control de accesos en Django permite limitar quién puede consultar o modificar la información. | En curso |  |
| Implementar soluciones sistémicas integrales para automatizar u optimizar procesos de negocio de acuerdo a las necesidades de la organización. | Exploración inicial de Videollamada | -Integración con Zoom  -Frontend para las vistas | Sprint 4 | Joaquin Madariaga,  Fernando Cavada. | Riesgo: Dependencia de un servicio externo, lo que implica estar sujetos a cambios en políticas, precios o limitaciones de la API.  Facilitadores: La API de Zoom cuenta con documentación extensa, ejemplos de integración y un entorno de pruebas gratuito para desarrolladores. | No iniciado |  |
| Construir programas y rutinas de variada complejidad para dar solución a requerimientos de la organización, acordes a tecnologías de mercado y utilizando buenas prácticas de codificación. | Ampliación BD (pagos, historial, soporte y calificaciones) | -Proyecto con el frontend y backend avanzados (vistas para el pago, historial y soporte en un estado básico) | Sprint 5 | Fernando Cavada, Joaquín Madariaga | Riesgo: Problemas de obtención de datos al usar servicios externos (el diseño de la base de datos no coincide con los datos entregados por las APIs). | No iniciado |  |
| *Implementar soluciones sistémicas integrales para automatizar u optimizar procesos de negocio de acuerdo a las necesidades de la organización.* | Funcionalidades: videollamadas completas y sistema de pago | -Infraestructura general del proyecto, incluyendo el backend en Django y la base de datos PostgreSQL.  -Pasarela de pagos Stripe  -API Key de zoom  -Certificados SSL | Sprint 5 | Joaquín Madariaga, Lucas Cisternas | Dificultades: La principal dificultad es cumplir con normas de seguridad financiera (PCI-DSS) y manejar errores en transacciones.  Facilitadores: Stripe ofrece SDKs y APIs bien documentadas. | No iniciado |  |
| Implementar soluciones sistémicas integrales para automatizar u optimizar procesos de negocio de acuerdo a las necesidades de la organización. | Modulo de administracion + soporte | -Infraestructura tecnológica general del proyecto (backend y frontend). | Sprint 5 | Joaquín Madariaga, Lucas Cisternas | Dificultades: Riesgo de dar seguimiento a tickets de ayuda duplicados y riesgo de dar seguimiento a los tickets de forma ineficiente  Facilitadores: Django Admin permite configurar un panel básico rápidamente y ampliarlo según las necesidades. Existen librerías y plantillas open source para sistemas de soporte. | En curso |  |
| Implementar soluciones sistémicas integrales para automatizar u optimizar procesos de negocio de acuerdo a las necesidades de la organización. | Exportación a móvil | -Android Studio  -Infraestructura tecnológica general del proyecto (backend y frontend) | Sprint 5 | Lucas Cisternas | Riesgo: diferencias en versiones de SO. Se recomienda pruebas continuas en emuladores y dispositivos físicos. | No iniciado |  |
| Asegurar la calidad del software mediante la validación del código, pruebas automatizadas y verificación de la integración entre componentes. | Pruebas unitarias a los módulos móviles | -Frameworks de testing: JUnit  -Plataforma con todos los módulos desarrollados. | Sprint 5 | Lucas Cisternas,  Joaquín Madariaga,  Fernando Cavada | Riesgo: cambios en requisitos pueden afectar casos de prueba. Se recomienda mantener pruebas automatizadas. | No iniciado |  |
| Implementar soluciones sistémicas integrales para automatizar u optimizar procesos de negocio de acuerdo a las necesidades de la organización. | Pruebas de integración final | -Usuarios finales.  -Plataforma con todas sus funciones integradas. | Sprint 5 | Lucas Cisternas,  Joaquín Madariaga,  Fernando Cavada | Importante incluir Marcha Blanca como validación con usuarios reales. | No iniciado |  |
| Gestionar proyectos informáticos, ofreciendo alternativas para la toma de decisiones de acuerdo a los requerimientos de la organización. | Cierre del proyecto | -Documentación final  -Acta de cierre  -Retrospectiva Scrum  -Repositorio Git | Sprint 5 | Lucas Cisternas, Fernando Cavada, Joaquín Madariaga | Importante incluir retrospectiva final y validación con stakeholders. | No iniciado |  |

| **3. Ajustes a partir del monitoreo** |
| --- |

| Facilitadores: coordinación efectiva con el equipo de desarrollo, claridad en los requerimientos funcionales, uso de metodologías ágiles con revisiones semanales, y disponibilidad de entornos de prueba funcionales.  Dificultadores: retrasos iniciales en la integración entre frontend (Ionic/Angular) y backend (Django), diferencias en versiones de dependencias y errores de comunicación con el servidor. Se resolvió mediante revisión del entorno virtual, actualización de dependencias y ajuste del flujo de consumo de API. Se planificó una ventana técnica adicional para pruebas de despliegue, además del limitado tiempo para realizar las actividades. |
| --- |

| Se ajustó el calendario de pruebas funcionales y de validación de usuarios, desplazando su inicio una semana para permitir estabilización del backend. Se eliminaron tareas duplicadas relacionadas con documentación técnica ya cubierta en la primera entrega. El ajuste se justificó por la necesidad de mantener coherencia entre módulos de registro y búsqueda de profesionales. |
| --- |

| El módulo de agenda profesional presenta retraso por priorización del flujo de autenticación y conexión con la base de datos. La causa principal fue la resolución de errores en endpoints y la reconfiguración de permisos en el servidor. Se aplicará una estrategia de avance paralelo: desarrollo de la interfaz de agenda mientras se realizan pruebas automatizadas para poder detectar errores lo más pronto posible. |
| --- |

1. En caso de que el Proyecto APT sea grupal, en esta columna deben indicar el nombre de los responsables de cada tarea o actividad. Esto posteriormente permitirá diferenciar la evaluación por cada integrante. [↑](#footnote-ref-0)