**Guía3. Informe final Proyecto APT**

**Asignatura Capstone**

|  |
| --- |
| **1. Informe final Proyecto APT** |
| El objetivo de este informe es que describas los aspectos más relevantes de tu Proyecto APT. Es importante que fundamentes las decisiones que tuviste que tomar a lo largo del proceso.  A continuación, encontrarás distintos campos que deberás completar con la información solicitada, los que dan cuenta del resumen de tu proyecto APT y sus principales resultados. |

|  |  |
| --- | --- |
| Nombre del proyecto | *NeoTotem Retail: Sistema inteligente de recomendación de moda con voz y visión por computadora.* |
| Área (s) de desempeño(s) | * **Desarrollo de Software**: implementación de backend con FastAPI, base de datos en SQLAlchemy/PostgreSQL y prototipo frontend en Flutter Web. * **Inteligencia Artificial Aplicada**: uso de visión por computadora (detección de prendas y colores), análisis de voz (NLU) y generación de recomendaciones personalizadas. * **Gestión de Proyectos de TI**: planificación, cronograma, gestión de riesgos y validación de avances según rúbrica. |
| Competencias | * **Diseño e implementación de software modular**: Se aplicó FastAPI con una arquitectura basada en routers independientes (productos, sesiones, recomendaciones, visión por computadora y análisis de voz), lo que permite una clara separación de responsabilidades y prepara el terreno para una futura migración hacia microservicios. * **Gestión de bases de datos relacionales**: Se diseñó y modeló una base de datos en PostgreSQL a partir de un MER y script DDL, utilizando SQLAlchemy para el ORM y manteniendo integridad en entidades como productos, variantes, sesiones y detecciones. * **Desarrollo de APIs RESTful y WebSockets**: Se construyeron endpoints documentados automáticamente con Swagger UI, además de un canal WebSocket para la comunicación en tiempo real con el frontend. * **Integración de IA aplicada**: Se incorporaron modelos de visión por computadora (MediaPipe/OpenCV/YOLO) y análisis de voz (NLU heurístico) para generar recomendaciones de moda personalizadas basadas en prendas detectadas y consultas por voz. * **Desarrollo frontend en Flutter Web**: Se implementó un prototipo de totem interactivo con pantallas de bienvenida, consulta por voz y recomendaciones, permitiendo visualizar el flujo completo de interacción usuario–backend. * **Pruebas y aseguramiento de calidad**: Se usaron herramientas como Swagger UI y Postman para validar los endpoints, y pruebas unitarias en Python para validar detección e interacción con los módulos. * **Gestión de proyectos tecnológicos**: Se cumplió con las fases definidas en la guía APT (planificación, cronograma, monitoreo de avances, ajustes), aplicando control de versiones con GitHub y registro de evidencias. |

|  |  |
| --- | --- |
| **Contenidos del informe final** | |
| 1. Relevancia del proyecto APT | * El proyecto **NeoTotem Retail** buscó **simular un sistema inteligente de recomendación de moda en un punto de venta (tótem interactivo)**, que integra visión por computadora y análisis de voz para entregar sugerencias personalizadas de prendas, tallas y colores a los clientes. * Este problema es relevante para el **campo laboral de la Ingeniería en Informática**, ya que aborda la integración de **inteligencia artificial, APIs RESTful, bases de datos y desarrollo frontend/backend** en un contexto real de retail, sector altamente demandante de soluciones tecnológicas innovadoras. * La situación se ubica en **Chile, en el rubro del retail y comercios de moda**, donde la experiencia de compra personalizada es cada vez más valorada por consumidores y empresas. La propuesta responde a un escenario en que las tiendas físicas buscan diferenciarse frente al comercio electrónico mediante el uso de **tecnologías interactivas y de análisis inteligente de clientes**. * Los principales afectados o beneficiados son **clientes jóvenes y adultos que realizan compras en retail** y **las empresas del sector moda**, ya que la solución les permitiría mejorar la experiencia de compra, aumentar la satisfacción del cliente y optimizar las recomendaciones de productos. * El aporte de valor de este Proyecto APT fue **la simulación de un tótem inteligente que combina IA y APIs RESTful para retail**, integrando tecnologías de análisis en tiempo real de imagen (ropa, colores, edad estimada) y procesamiento de voz (intenciones y consultas). Aunque en esta etapa es un prototipo académico, aporta una **base tecnológica replicable** que puede aplicarse en proyectos reales de retail, alineándose con las tendencias actuales de **IA aplicada al comercio, automatización y experiencia de usuario personalizada**. |
| 2. Objetivos | 1. **Objetivo general** Desarrollar un prototipo funcional de **tótem inteligente para retail**, que combine visión por computadora y análisis de voz mediante una **API RESTful en FastAPI y un frontend en Flutter Web**, con el fin de **ofrecer recomendaciones personalizadas de prendas, tallas y colores** a los clientes. 2. **Objetivos Específicos:**  * **Diseñar y modelar la base de datos** (MER y DDL en PostgreSQL/SQLite) para almacenar productos, categorías, variantes y sesiones de interacción. * **Implementar servicios backend** en **FastAPI**, desarrollando endpoints RESTful para gestionar productos, sesiones y recomendaciones. * **Integrar algoritmos de IA simulados** que permitan el **análisis en tiempo real de prendas, colores y estimación de edad** a través de visión por computadora. * **Incorporar un módulo de análisis de voz (NLU)** que procese las consultas del usuario y detecte intenciones para enriquecer la experiencia de interacción. * **Desarrollar un prototipo de frontend en Flutter Web**, que actúe como la interfaz de usuario del tótem y muestre productos y recomendaciones de manera interactiva. * **Realizar pruebas de funcionamiento** utilizando **Swagger UI y Postman** para validar la correcta interacción entre frontend, backend y base de datos. * **Documentar el proceso de desarrollo y resultados obtenidos**, destacando aprendizajes, dificultades y posibles mejoras futuras para un escenario real de retail. |
| 3. Metodología | Para el desarrollo del Proyecto APT se utilizó la metodología **CRISP-DM**, la cual resultó pertinente dado que integra procesos de **análisis de datos, modelamiento con inteligencia artificial y validación en entornos aplicados**, lo que corresponde al alcance del prototipo del tótem inteligente.  Las fases de CRISP-DM aplicadas en el proyecto fueron las siguientes:   1. **Comprensión del negocio** Se definió el problema dentro del contexto del **retail en Chile**, específicamente en la experiencia de compra de clientes en tienda física. El desafío identificado fue la necesidad de mejorar la interacción cliente-producto mediante un sistema automatizado que detecte ropa, colores y preferencias, y entregue recomendaciones personalizadas de moda. 2. **Comprensión de los datos** Se analizaron las fuentes de datos necesarias:    * **Datos estructurados**: tablas de productos, categorías, sesiones e interacciones, definidas en el **MER** y el **DDL.sql**.    * **Datos no estructurados**: imágenes capturadas desde cámara y transcripciones de voz del cliente. Esto permitió diseñar un modelo de datos robusto (models.py) que respalda las pruebas del backend. 3. **Preparación de los datos** Se implementaron los scripts de inicialización y conexión a la base de datos (init\_db.py, database.py), normalizando atributos como categorías, productos, variantes y sesiones. Se emplearon validaciones con **Pydantic v2** y endpoints en **FastAPI** para asegurar consistencia en entradas y salidas. 4. **Modelado** Se integraron modelos de IA y visión por computadora:    * **MediaPipe y YOLO** (mediapipe\_engine.py, yolo\_clothing\_detector.py) para la detección de ropa y accesorios.    * **NLU heuristics** (heuristics.py) para extraer intenciones y entidades desde comandos de voz. Además, se simularon escenarios realistas con real\_detection.py y simple\_ai.py, garantizando la coherencia del flujo de datos. 5. **Evaluación** Los endpoints fueron probados con **Swagger UI** y **Postman**, documentando los resultados y asegurando que las operaciones CRUD de productos y sesiones funcionaran correctamente. También se desarrollaron **tests automatizados** (test\_real\_detection.py, test\_frontend\_endpoint.py) que verificaron la respuesta del sistema en distintos escenarios. 6. **Despliegue** El sistema fue configurado en un entorno local utilizando **FastAPI + Uvicorn** para el backend y **Flutter Web** para el frontend, funcionando en un notebook como prototipo de tótem de retail. Esto permitió simular la experiencia real de uso e integrar los módulos de cámara y voz en un flujo único. |
| 4. Desarrollo | El Proyecto APT se estructuró en las siguientes etapas principales:   1. **Definición y análisis inicial**    * Identificación del problema en el contexto del retail físico en Chile.    * Diseño del **Modelo Entidad-Relación (MER)** y creación del script **DDL.sql** para la base de datos.    * Establecimiento del alcance: detección de prendas, interacción por voz y recomendaciones personalizadas.  * **Diseño de la arquitectura**   + Desarrollo del backend en **FastAPI** con endpoints RESTful (productos, sesiones, recomendaciones).   + Creación de modelos en **SQLAlchemy** (models.py) y esquemas de validación con **Pydantic v2**.   + Definición de servicios de IA: detección de ropa con **MediaPipe/YOLO**, análisis de voz con heurísticas NLU. * **Implementación**   + Construcción de módulos de detección de ropa y accesorios (mediapipe\_engine.py, real\_detection.py).   + Desarrollo de endpoints en FastAPI (productos.py, sesiones.py, recomendaciones.py).   + Creación de prototipo de **frontend en Flutter Web** para simular el tótem interactivo. * **Pruebas y validación**   + Ejecución de pruebas con **Swagger UI** y **Postman** para validar los endpoints CRUD.   + Pruebas automatizadas (test\_real\_detection.py, test\_frontend\_endpoint.py) para verificar la detección en escenarios controlados.   + Evidencias de integración en el prototipo funcionando en entorno local con **Uvicorn**. * **Entrega del prototipo**   + Montaje del sistema en un notebook como simulador de tótem.   + Generación de capturas y documentación de resultados para evidenciar funcionamiento del sistema.   **Facilitadores**   * Uso de **FastAPI**: permitió construir rápidamente una API clara y documentada. * Librerías como **MediaPipe, YOLO y OpenCV**, que agilizaron la implementación de visión por computadora. * **Swagger UI y Postman**, que facilitaron la validación de endpoints. * Trabajo en equipo con roles definidos (Backend, CV & Voz, Frontend & UX), lo que distribuyó las cargas de manera eficiente.   **Dificultades**   * **Integración de módulos de IA (voz e imagen)**: requirió múltiples pruebas para lograr coherencia en los resultados. * **Problemas de configuración** en entornos de desarrollo (IntelliJ, Visual Studio, Flutter) que retrasaron algunas fases. * **Compatibilidad de dependencias** (ej. Pydantic v2 con SQLAlchemy, instalación de paquetes de visión por computadora). * Limitaciones de hardware: la simulación en notebook no refleja la velocidad de procesamiento de un tótem real.   **Ajustes realizados**   * Se decidió **simular algunos análisis de prendas** (ej. colores, estilos) en lugar de implementar modelos entrenados desde cero, para asegurar la entrega en el plazo establecido. * Se ajustó el alcance del **frontend Flutter** a un prototipo de navegación y recomendaciones básicas, dejando pendiente la conexión directa con el backend para futuras fases. * Se documentaron las pruebas con capturas de **Swagger y Postman** como evidencia para suplir la falta de despliegue en la nube. |
| 5. Evidencias | **1.Script DDL en SQL (DDL.sql)** → demuestra la **implementación técnica de la base de datos**.   * Incluye las tablas, constraints, claves foráneas, triggers y secuencias. * Sirve como **prueba de que el modelo lógico fue implementado en un motor real** (Oracle 11g).     2**.** **Modelo Entidad-Relación (MER)** en formato PDF, que permite visualizar la estructura conceptual y relacional de la base de datos.    **3.Evidencia del Endpoint Raíz** → demuestra que la API está levantada y respondiendo (<http://127.0.0.1:8000/>).  **4.Evidencia de Operaciones (Sesiones, ASR, CV, Recomendaciones, Productos)** → capturas de pruebas ejecutadas en Swagger/Postman mostrando que los módulos funcionan.    **5.Evidencia Swagger UI** → garantiza que la API está correctamente documentada y lista para consumir. |
| 6. Intereses y proyecciones profesionales | Durante el desarrollo del Proyecto APT pude darme cuenta de que este trabajo fue un gran aporte para clarificar mis intereses profesionales. Al inicio tenía inquietudes en varias áreas: desarrollo backend, inteligencia artificial y la nube, además de un interés personal en seguridad informática. Con el proyecto confirmé que todas estas disciplinas se pueden integrar en un mismo sistema, lo que refuerza mi motivación por seguir creciendo en esas líneas.  Mis intereses profesionales siguen siendo los mismos que tenía al inicio, pero ahora con más claridad y confianza: quiero profundizar en el **desarrollo de software con integración de IA**, en **arquitecturas que puedan escalar en la nube (AWS, GCP, Azure)** y en **seguridad informática aplicada a los sistemas distribuidos y en la nube**, ya que la protección de los datos y de la infraestructura tecnológica es un aspecto clave en la industria actual.  Después de este proyecto me proyecto laboralmente en áreas relacionadas al **desarrollo backend, soluciones fullstack y aplicaciones de IA aplicada al retail y la experiencia de usuario**, pero también con una proyección hacia **ciberseguridad en entornos cloud** y **arquitecturas seguras y escalables**. Me interesaría trabajar en empresas tecnológicas, de retail o en bancos, donde pueda aportar en el diseño de soluciones seguras y con valor para los usuarios. |