**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 estudiante | [Nibaldo Andres Quezada Garay](mailto:nib.quezada@duocuc.cl) |
| Rut | **19.116.353-2** |
| Carrera | **Ingeniería en Informática** |
| Sede | **San Bernardo** |

|  |  |
| --- | --- |
| Nombre estudiante | **Nicolás Gabriel Peña Muñoz** |
| Rut | **18.795.757-5** |
| Carrera | **Ingeniería en Informática** |
| Sede | **San Bernardo** |

|  |  |
| --- | --- |
| Nombre estudiante | [Cristian Mauricio Arroyo Bossardt](mailto:cri.arroyo@duocuc.cl) |
| Rut | **15.893.730-1** |
| Carrera | **Ingeniería en Informática** |
| Sede | **San Bernardo** |

|  |  |
| --- | --- |
| **Contenidos del informe final** | |
| 1. Relevancia del proyecto APT | Nuestro cliente corresponde al condominio vista parque 1 y 2, el cual se encuentra ubicado en la comuna de San Bernardo, en Almirante Riveros #630. El condominio cuenta con 570 departamentos y 30 casas, teniendo un total de 600 domicilios. El condominio presenta dos grandes problemáticas, las cuales se describen a continuación.   1. Debido a la gran cantidad de vehículos en el recinto y la manualidad en la operación diaria de parte de los conserjes, se generan atochamientos tanto en el ingreso al condominio, como en la avenida en la cual se encuentra situado. 2. Producto de los altos índices de delincuencia, la administración necesita de un sistema que restrinja el acceso vehicular, asegurando el acceso exclusivo a residentes y personal autorizado, con la intención de prevenir robos y “portonazos”, lo que ha afectado últimamente al condominio.   En TechApps ponemos como foco principal la calidad de vida y el trabajo de las personas, por esto frente a la problemática planteada por el condominio, decidimos proponerles una solución completa y eficiente que resuelve su problema de gestión de acceso vehicular, ya que como empresa creemos que un proyecto de esta envergadura, se podría aplicar todo el conocimiento adquirido durante el proceso formativo en Duoc, Desde la gestión del proyecto y la calidad del producto, hasta el desarrollo e implementación del software. |
| 2. Objetivos | Objetivos del Proyecto  Objetivo General: Desarrollar un sistema integral de control de acceso vehicular que permita mejorar la seguridad y agilizar el ingreso y salida de vehículos en el condominio Vista Parque, asegurando el control exclusivo de residentes y personal autorizado.  Objetivos Específicos:   1. Implementar un lector automático de patentes vehiculares utilizando Python. 2. Desarrollar una aplicación móvil en Flutter para que los residentes puedan registrar visitas anticipadamente. 3. Crear una API REST en Spring Boot para la gestión de permisos de acceso vehicular e integración con la API externa. 4. Diseñar una base de datos en SQL Server que soporte la gestión de residentes, visitas y vehículos comerciales. 5. Generar reportes de acceso vehicular y monitoreo en tiempo real. 6. Asegurar que el sistema cumpla con estándares de seguridad informática. |
| 3. Metodología | La metodología seguida es Scrum, con 9 esprints de dos semanas. Las actividades incluyen la planificación de requisitos, desarrollo por módulos (lector de patentes, API, aplicación móvil, base de datos), pruebas de integración, y la futura puesta en marcha del sistema.  1. Historias de Usuario  Las historias de usuario son esenciales para definir los requisitos del sistema y asegurar que las funcionalidades desarrolladas se alineen con las necesidades de los usuarios finales. A continuación, se detallan las historias de usuario priorizadas para la aplicación móvil, enfocadas en la mejora del acceso vehicular en el condominio Vista Parque.  01. Registro de Vehículos de Residentes   * Descripción: Como residente, quiero que mi vehículo esté registrado y pueda ingresar automáticamente al condominio para evitar la congestión en la entrada durante las horas punta. * Prioridad: Alta * Justificación: Esta funcionalidad es crítica para agilizar el flujo vehicular y reducir los tiempos de espera en la entrada, mejorando la experiencia diaria de los residentes.   02. Registro de Vehículos de Visitas   * Descripción: Como residente, quiero poder registrar visitas anticipadamente para que puedan ingresar sin demoras, en cualquier horario. * Prioridad: Alta * Justificación: Facilitar el acceso de visitas previamente autorizadas mejora la seguridad y eficiencia, permitiendo a los residentes gestionar sus visitas de forma autónoma.   03. Historial de Visitas   * Descripción: Como residente, quiero poder consultar el historial de mis entradas y salidas, así como de las visitas al condominio, para controlar mis movimientos vehiculares y evitar multas. * Prioridad: Media * Justificación: Esta función permite a los residentes tener un mayor control sobre el acceso de sus vehículos y visitas, contribuyendo a la transparencia y a la prevención de infracciones.   2. Product Backlog  El Product Backlog define las funcionalidades y módulos clave que el sistema debe incluir para optimizar el control de acceso vehicular y la seguridad en el condominio Vista Parque. A continuación, se describen las funcionalidades priorizadas para el equipo de desarrollo.  01. Monitoreo en Tiempo Real   * Descripción: Como personal de seguridad (Portería), quiero monitorear en tiempo real el flujo vehicular en la entrada para agilizar el acceso y atender incidentes en situaciones de emergencia. * Prioridad: Alta * Justificación: Esta funcionalidad es esencial para que el personal de seguridad tenga una visión clara del acceso vehicular y pueda tomar decisiones rápidas en caso de emergencias.   02. Gestor de Multas   * Descripción: Como personal de seguridad (Portería), quiero que el sistema gestione y registre automáticamente las multas por visitas que exceden el tiempo autorizado. * Prioridad: Media * Justificación: Automatizar el control de multas ayuda a reducir la carga administrativa y asegura que las reglas de acceso sean cumplidas.   03. Consulta de Vehículos Comerciales   * Descripción: Como personal de seguridad (Portería), quiero poder consultar rápidamente el acceso de vehículos comerciales (Uber, correos, pedidos) para darles acceso de manera rápida al condominio. * Prioridad: Media * Justificación: Optimiza el acceso de vehículos de servicio, mejorando la eficiencia sin comprometer la seguridad del condominio.   04. Control de Acceso para Emergencias   * Descripción: Como personal de seguridad (Portería), quiero tener el control total del acceso en situaciones de emergencia para garantizar la seguridad de los residentes y el condominio. * Prioridad: Alta * Justificación: Esta funcionalidad es crítica para asegurar un acceso controlado durante emergencias, evitando que personas no autorizadas ingresen en momentos de crisis.   El desarrollo de las funcionalidades descritas en las historias de usuario y el backlog tiene un enfoque claro en mejorar la eficiencia del acceso vehicular, garantizar la seguridad de los residentes y facilitar la labor del personal de seguridad. Al priorizar las necesidades tanto de los residentes como del personal administrativo, se busca crear un sistema integral y escalable que responda a los desafíos del condominio Vista Parque.  El proyecto TechApps SCAV ha sido planificado para desarrollarse en un total de 9 sprints, con una duración de 2 semanas cada uno. A continuación, se describe el desglose detallado de las actividades a realizar en cada fase del proyecto, así como los objetivos específicos de cada sprint.  **Resumen del Plan de Trabajo**  El proyecto TechApps SCAV ha sido planificado para desarrollarse en un total de 9 sprints, con una duración de 2 semanas cada uno. A continuación, se describe el desglose detallado de las actividades a realizar en cada fase del proyecto, así como los objetivos específicos de cada sprint.  **1. Sprint 1 y 2** **Gestión de Proyecto**   * Actividades: Planificación de los sprints, creación del backlog y definición de prioridades. * Duración: Semana 1 a la 4   **2. Sprint 3 Desarrollo del Sistema de Reconocimiento de Patentes**   * Actividades: Implementar el sistema de reconocimiento de patentes usando Python y OpenCV para capturar y validar patentes vehiculares. * Duración: Semana 5 y 6   **3. Sprint 4 Análisis y Evaluación de la Base de Datos**   * Actividades: Diseñar la estructura de la base de datos y su modelo relacional utilizando Google Cloud SQL. * Duración: Semana 7 y 8   **4. Sprint 5 Desarrollo de la API REST en Spring Boot**   * Actividades: Implementar la API REST en Spring Boot para gestionar la comunicación entre el frontend y el backend, utilizando Google Kubernetes Engine (GKE) para el despliegue. * Duración: Semana 9 y 10   **5. Sprint 6 y 7 Desarrollo de la Aplicación Móvil (Flutter)**   * Actividades: Crear una aplicación móvil en Flutter que se conecte con la API para que los residentes puedan registrar visitas, gestionar vehículos y recibir notificaciones. * Duración: Semana 11 a la 14   **6. Sprint 8 Pruebas de Calidad de Software**   * Actividades: Realizar pruebas funcionales, de integración y seguridad para garantizar que el sistema sea robusto y confiable. * Duración: Semana 15 y 16   **7. Sprint 9 Implementación, Capacitación y Marcha Blanca**   * Actividades: Realizar la marcha blanca, capacitación a los usuarios finales (personal de seguridad y administradores) y despliegue final en producción. * Duración: Semana 17 y 18   **Plan de Trabajo**  El cronograma estructurado en 9 sprints, con una duración de 2 semanas cada uno, está diseñado para asegurar que el proyecto TechApps SCAV cumpla con los plazos establecidos y se entregue con la calidad esperada. La integración de tecnologías en la nube y el uso de metodologías ágiles han sido claves para optimizar tanto el desarrollo como la implementación del sistema.  Este enfoque permitirá mejorar la seguridad y eficiencia del control de acceso vehicular en el condominio Vista Parque, beneficiando tanto a los residentes como al personal de seguridad. |
| 4. Desarrollo | |  |  |  |  |  | | --- | --- | --- | --- | --- | | Actividad | Estado | Dificultades | Materiales/Recursos Físicos | Lenguajes/Entorno | | Lector de Patentes | Completado | Dificultades para hacer pruebas en vivo con la cámara en la calle debido a condiciones climáticas y tráfico | Cámara de video, PC | Python, OpenCV | | Diseño de Base de Datos | Completado | N/A | |  | | --- | | PC, Google Cloud SQL |  |  | | --- | |  | | PostgreSQL | | API REST | En curso | N/A | PC, GCP (Google Cloud Platform) | Java, Spring Boot | | Aplicación Móvil en Flutter | En curso | N/A | PC, smartphone para pruebas | Flutter, Dart | | Pruebas de Integración | No iniciado | Coordinación para pruebas con cámara en tiempo real en un entorno controlado | Cámara de video, PC, red local para la integración | Python, Java, SQL, Flutter |   Actividades Futuras   * Realizar pruebas de integración entre los módulos. * Preparar el sistema para la puesta en marcha y capacitación a usuarios finales.   Factores que han facilitado el desarrollo del plan de trabajo:   * Coordinación del equipo: La implementación de reuniones diarias mediante la metodología Scrum ha permitido una mejor comunicación y resolución rápida de problemas. * Herramientas tecnológicas adecuadas: Contamos con un entorno de desarrollo bien configurado, incluyendo Python, SQL Server y Spring Boot, lo que ha facilitado el trabajo de los desarrolladores. * Prototipo funcional temprano: El lector de patentes fue completado en las primeras fases del proyecto, lo que permitió avanzar rápidamente en la integración con otros módulos.   Factores que han dificultado el desarrollo del plan de trabajo:   1. **Cambio en la Estructura de la Aplicación**   **Dificultad Identificada**: Mencionado como cambio anterior, se planificó desarrollar dos aplicaciones independientes: una para **administración** y otra para **residentes**. Sin embargo, esta separación generaba redundancia, complicaba la gestión y aumentaba la carga de mantenimiento.  **Acción Tomada**: En lugar de continuar con dos aplicaciones separadas, se tomó la decisión de **unificar** todas las funcionalidades en **una sola aplicación móvil**.  La nueva aplicación es **dinámica** y adapta sus funcionalidades según el **rol del usuario** que inicia sesión (Administrador o Residente).   1. Error en la aplicación de la metodología Scrum:    * Dificultad: Aunque se planificó trabajar con Scrum, en las primeras semanas se adoptó por error un enfoque tradicional, afectando la planificación y las entregas.    * Acción tomada: Se realizó una corrección inmediata para retomar la metodología Scrum, estableciendo sprints de 2 semanas con reuniones diarias y entregas incrementales. 2. Ajuste en el cronograma de sprints:    * Dificultad: La planificación inicial de los sprints no asignaba suficiente tiempo para algunos módulos clave, como la aplicación en Flutter.    * Acción tomada: Se redistribuyeron las tareas en 9 sprints, dando 2 sprints completos para el desarrollo en Flutter y ajustando las fechas del resto de los módulos.  Actividades ajustadas o eliminadas Ajustes realizados:   1. Unificación de aplicaciones en dos versiones (Administrativa y Residentes):    * Motivo: Originalmente, se planificó desarrollar dos aplicaciones separadas: una para **administración** y otra para **residentes**. Sin embargo, durante el desarrollo, se identificó que esta separación generaba redundancias y aumentaba la complejidad del sistema.    * Ajuste: Se decidió volver a la idea original de desarrollar una sola aplicación unificada que integre todas las funcionalidades, adaptando su interfaz y opciones según el rol del usuario (Administrador o Residente) que inicie sesión.    * Impacto: Esta unificación simplifica el mantenimiento y actualizaciones, mejora la experiencia del usuario, y reduce la carga de trabajo del equipo de desarrollo. Ahora, todos los esfuerzos se centran en **optimizar la aplicación unificada**, que garantiza una gestión más eficiente y escalabilidad del sistema. 2. Ajuste en el cronograma de sprints:    * Motivo: La planificación inicial asignaba un sprint por módulo, lo que no proporcionaba el tiempo adecuado para el desarrollo de algunos componentes más complejos, como la aplicación móvil en Flutter.    * Ajuste: Se realizó una redistribución del cronograma, incrementando a dos sprints el tiempo para el desarrollo de la aplicación en Flutter y manteniendo un sprint para los módulos más simples. Esta nueva estructura asegura una mejor planificación y evita retrasos.   Actividades eliminadas: Por el momento, no se han eliminado actividades, ya que todas las fases definidas inicialmente siguen siendo relevantes para alcanzar los objetivos del proyecto. |
| 5. Evidencias | Módulo Lector de patentes  A continuación, describiremos el funcionamiento del sistema de control de acceso vehicular desarrollado para optimizar la seguridad en recintos privados, tales como condominios, oficinas o estacionamientos. El sistema permite el acceso automatizado de vehículos autorizados mediante la detección de patentes (placas de vehículos), y registra tanto la entrada como la salida en una bitácora a través de una API. Adicionalmente, se integra un servomotor controlado por un Arduino para manejar una barrera que se eleva automáticamente al detectar una patente válida.  **Descripción General del Sistema**  El sistema consta de varios componentes que interactúan entre sí:   1. **Cámara**: Captura en tiempo real las imágenes del vehículo al acercarse al punto de control. 2. **Lector de Patentes (Python)**: Detecta la patente del vehículo a partir del flujo de video y realiza consultas a una API para verificar si el vehículo está registrado. 3. **API (Backend)**: Se encarga de manejar la información sobre los vehículos autorizados y de registrar los accesos en una base de datos. 4. **Control de Barrera (Arduino)**: Utiliza un servomotor para controlar la apertura y cierre de una barrera física, activada por comandos enviados desde el sistema Python.   **Funcionamiento del Sistema**  **1. Flujo de Entrada de Vehículos**  El proceso de entrada al recinto sigue estos pasos:   1. **Captura de Imagen**: La cámara captura continuamente imágenes en tiempo real. 2. **Detección de la Patente**:    * Utilizando la biblioteca OpenCV en Python, se detecta la presencia de una patente en el flujo de video.    * El número de la patente se extrae utilizando técnicas de reconocimiento óptico de caracteres (OCR). 3. **Consulta a la API**:    * El número de la patente detectada se envía a un endpoint específico (/api/v2/vehiculo/patente/{patente}) para verificar si el vehículo está registrado.    * La API devuelve información sobre el vehículo, incluyendo su estado (residente, visita, etc.). 4. **Registro en la Bitácora**:    * Si el vehículo está registrado, se crea un nuevo registro de entrada en la bitácora mediante una solicitud POST a la API (/api/v2/bitacora).    * La bitácora almacena la hora de entrada y deja el campo de salida como null hasta que el vehículo salga. 5. **Activación de la Barrera**:    * Si la patente es válida, se envía un comando al Arduino a través de la comunicación serial para que el servomotor eleve la barrera.   **2. Flujo de Salida de Vehículos**  El proceso de salida sigue una lógica similar, con algunas diferencias:   1. **Captura y Detección**:    * La cámara detecta la patente del vehículo cuando intenta salir del recinto. 2. **Verificación y Actualización en la Bitácora**:    * El número de la patente se consulta en la API para obtener el ID del registro de entrada correspondiente.    * Se envía una solicitud PUT a la API (/api/v2/bitacora/salida/{vehiculoId}) para actualizar el campo fechaout con la hora de salida. 3. **Cálculo del Tiempo de Permanencia**:    * Una vez registrado el horario de salida, se calcula la duración total del tiempo de permanencia del vehículo en el recinto. 4. **Apertura de la Barrera para Salida**:    * Al confirmar que el registro ha sido actualizado correctamente, se envía un comando al Arduino para abrir la barrera y permitir la salida del vehículo.   **Conexión con el Arduino para Control de Barrera**  El sistema utiliza un Arduino conectado a un servomotor que controla la barrera. El Arduino se comunica con el sistema Python mediante un puerto serial (COM6, en este caso). La integración se realiza de la siguiente forma:   1. **Inicialización**:    * El Arduino se configura para escuchar comandos desde el puerto serial. 2. **Comando de Apertura**:    * Cuando se detecta una patente válida, se envía el comando subir al Arduino, el cual activa el servomotor para elevar la barrera. 3. **Cierre Automático**:    * Después de un breve intervalo (5 segundos), el Arduino baja automáticamente la barrera.   Módulo API REST    Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto  Descripción generada automáticamenteInterfaz de usuario gráfica, Texto  Descripción generada automáticamenteInterfaz de usuario gráfica  Descripción generada automáticamente con confianza mediaMódulo Flutter  Interfaz de usuario gráfica  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Correo electrónico  Descripción generada automáticamente  Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación, Word  Descripción generada automáticamente  Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto  Descripción generada automáticamente  Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  Interfaz de usuario gráfica, Texto, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  Interfaz de usuario gráfica, Aplicación  Descripción generada automáticamenteInterfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto  Descripción generada automáticamente  Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto  Descripción generada automáticamenteInterfaz de usuario gráfica, Aplicación  Descripción generada automáticamente  MER    Diagrama arquitectura |
| 6. Intereses y proyecciones profesionales | * + Dentro del equipo contamos con intereses diversos, algunos enfocados en la gestión de proyectos, como otros en el desarrollo de software netamente. Por el motivo mencionado anteriormente, es que creemos que el proyecto se adapta a nuestro equipo y sus intereses profesionales, ya que abarca todos los ámbitos. |