Capstone

informe

LIZA . MOLINA VENEGAS – Jaime malhue olmoS

2024

Tabla de contenido

[Proyecto APT – CitaConecta 2](#_Toc180403861)

[Descripción del Proyecto APT - CitaConecta 2](#_Toc180403862)

[Objetivo del Proyecto 2](#_Toc180403863)

[Propósito del Proyecto 3](#_Toc180403864)

[Informe de Mejoras en la Gestión del Proyecto 3](#_Toc180403865)

[Abstract 3](#_Toc180403866)

[Nuevas Mejoras 4](#_Toc180403867)

[1. Reorganización de la Planificación de Trabajo en Jira 4](#_Toc180403868)

[2. División del Trabajo de Documentación según PMBOK 7 5](#_Toc180403869)

[3. Fases del Proyecto: Inicio, Planificación y Ejecución 5](#_Toc180403870)

[4. Procesos Clave de Gestión Integrados 6](#_Toc180403871)

[Opciones de Arquitectura de software analizadas para nuestro proyecto. 7](#_Toc180403872)

[1. Vista Lógica (Logical View) 7](#_Toc180403873)

[2. Vista de Desarrollo (Development View) 8](#_Toc180403874)

[3. Vista de Procesos (Process View) 9](#_Toc180403875)

[4. Vista Física (Physical View) 9](#_Toc180403876)

[5. Vista de Casos de Uso (Use Case View) 10](#_Toc180403877)

[Arquitectura Basada en Capas (Layered Architecture). 11](#_Toc180403878)

[1. Capa de Presentación (Front-End Layer) 12](#_Toc180403879)

[2. Capa de Servicios y API (Service/API Layer) 12](#_Toc180403880)

[3. Capa de Lógica de Negocio (Business Logic Layer) 12](#_Toc180403881)

[4. Capa de Acceso a Datos (Data Access Layer) 13](#_Toc180403882)

[5. Capa de Integración (Integration Layer) 13](#_Toc180403883)

[6. Capa de Infraestructura (Infrastructure Layer) 14](#_Toc180403884)

[Beneficios de la Arquitectura en Capas: 14](#_Toc180403885)

# Proyecto APT – CitaConecta

## Descripción del Proyecto APT - CitaConecta

El proyecto CitaConecta tiene como objetivo desarrollar una plataforma web enfocada en la automatización de reservas de citas para pequeñas y medianas empresas (pymes) en Santiago de Chile. Esta solución permitirá a las pymes gestionar eficientemente las citas de sus clientes, optimizando la planificación y el uso de recursos. La plataforma facilitará la interacción entre las empresas y sus clientes mediante un sistema intuitivo de gestión de reservas, recordatorios automáticos y control de disponibilidad en tiempo real.

## Objetivo del Proyecto

El objetivo principal de CitaConecta es proporcionar una herramienta tecnológica que permita a las pymes organizar de manera eficaz sus agendas, mejorando su eficiencia operativa y atención al cliente. Además, la plataforma integrará funciones como la gestión de clientes, generación de reportes y el análisis de datos, permitiendo a las empresas tomar decisiones informadas basadas en el comportamiento de sus usuarios.

## Propósito del Proyecto

El propósito de CitaConecta es mejorar la experiencia tanto para las empresas como para sus clientes, facilitando la gestión de citas a través de una plataforma amigable y accesible. El proyecto no solo optimiza procesos internos, sino que también contribuye a la digitalización de las pymes, ayudándolas a adaptarse a un entorno cada vez más competitivo y digitalizado.

Este proyecto aborda áreas clave del plan de estudios de Ingeniería en Informática de Duoc UC, tales como desarrollo de software, gestión de proyectos informáticos, integración de sistemas, seguridad informática, análisis de datos, y soporte técnico. La sinergia entre el proyecto y la formación académica asegura una aplicación de conocimientos integrales, contribuyendo tanto al éxito del proyecto como al perfil profesional del estudiante.

# Informe de Mejoras en la Gestión del Proyecto

## Abstract

Este informe describe una serie de mejoras implementadas en la gestión del proyecto con el objetivo de optimizar el flujo de trabajo, mejorar la documentación y asegurar una ejecución eficiente. Estas mejoras se centran en la reorganización de la planificación del trabajo utilizando Jira, la división de la documentación basada en el estándar PMBOK 7, y la estructuración del proyecto en fases clave: Inicio, Planificación y Ejecución. Además, se integran procesos esenciales como gestión de la integración, alcance, tiempo, costos, calidad, recursos humanos, comunicación, riesgos, adquisiciones y gestión de interesados para garantizar un desarrollo del proyecto más eficiente y alineado con las mejores prácticas en gestión de proyectos.

# Nuevas Mejoras

## 1. Reorganización de la Planificación de Trabajo en Jira

Para mejorar el manejo de la información y la distribución de tareas, se ha implementado una reorganización completa de la planificación del trabajo dentro de **Jira**. Este cambio permite una visualización más clara de los hitos, el seguimiento del progreso y la asignación de recursos. Las principales mejoras incluyen:

* **Priorización de tareas**: Asignación de prioridades claras y visibles para cada tarea, lo que facilita la toma de decisiones sobre qué actividades deben completarse primero.
* **Uso de sprints**: Implementación de ciclos de trabajo definidos (sprints) para realizar entregas incrementales, optimizando la ejecución de las tareas.
* **Seguimiento de dependencias**: Visualización clara de las dependencias entre tareas para evitar cuellos de botella y asegurar que los equipos puedan trabajar de manera eficiente.

Estas mejoras en la planificación permiten un mejor manejo de los recursos, el seguimiento continuo del progreso y la identificación temprana de problemas.

## 2. División del Trabajo de Documentación según PMBOK 7

Se ha dividido el trabajo de documentación del proyecto siguiendo las pautas del **PMBOK 7** para asegurar una cobertura exhaustiva y alineación con estándares globales de gestión de proyectos. La documentación se ha estructurado en las siguientes áreas:

* **Inicio del Proyecto**: Documentación inicial que define los objetivos, el alcance, los interesados y los riesgos identificados en las primeras etapas del proyecto.
* **Planificación**: Documentos que cubren los planes detallados de gestión del alcance, cronograma, costos, calidad y riesgos.
* **Ejecución**: Reportes y documentación generada durante la ejecución del proyecto, incluyendo actualizaciones sobre la gestión de la calidad, monitoreo de riesgos y avances respecto al cronograma.

Esta división asegura una mejor organización del trabajo de documentación y facilita la trazabilidad de las decisiones y cambios a lo largo del ciclo de vida del proyecto.

## 3. Fases del Proyecto: Inicio, Planificación y Ejecución

El proyecto ha sido estructurado en tres fases principales, asegurando un enfoque claro y eficiente para cada etapa:

* **Fase de Inicio**: Definición del caso de negocio, identificación de los interesados, y establecimiento de los objetivos y el alcance preliminar.
* **Fase de Planificación**: Desarrollo de los planes de trabajo detallados, especificando los cronogramas, recursos, presupuesto y análisis de riesgos. Esta fase también incluye la definición de los criterios de calidad y las métricas de éxito.
* **Fase de Ejecución**: Implementación de las tareas planificadas, con una gestión continua del cronograma, costos, calidad y riesgos. Se implementa un control riguroso de cambios y se llevan a cabo entregas incrementales del proyecto.

Al dividir el proyecto en estas fases, se logra una mayor claridad en los procesos, lo que facilita la asignación de tareas y el monitoreo del progreso.

## 4. Procesos Clave de Gestión Integrados

En alineación con las mejores prácticas de gestión de proyectos, se han integrado los siguientes procesos clave en la gestión del proyecto:

* **Gestión de la Integración**: Coordinación de todos los aspectos del proyecto para asegurar que se trabaja de manera coherente hacia los objetivos establecidos.
* **Gestión del Alcance**: Definición y control de lo que está y no está incluido en el proyecto para evitar desviaciones de alcance.
* **Gestión del Tiempo**: Monitoreo y control de los cronogramas para asegurar que el proyecto se mantiene en curso según lo planeado.
* **Gestión de los Costos**: Control de los gastos del proyecto para asegurarse de que se mantengan dentro del presupuesto.
* **Gestión de la Calidad**: Implementación de medidas para asegurar que el proyecto cumple con los estándares de calidad predefinidos.
* **Gestión de los Recursos Humanos**: Asignación eficiente de los recursos humanos para maximizar el rendimiento y asegurar el cumplimiento de las tareas asignadas.
* **Gestión de la Comunicación**: Establecimiento de canales efectivos de comunicación para mantener informados a todos los interesados y miembros del equipo.
* **Gestión de los Riesgos**: Identificación, evaluación y respuesta a los riesgos para mitigar su impacto en el proyecto.
* **Gestión de las Adquisiciones**: Supervisión de la adquisición de recursos externos necesarios para completar el proyecto.
* **Gestión de los Interesados**: Identificación y gestión de las expectativas y necesidades de los interesados a lo largo del ciclo de vida del proyecto.

# Opciones de Arquitectura de software analizadas para nuestro proyecto.

Arquitectura de Software del Proyecto CitaConecta (Modelo 4+1)

El modelo 4+1 es una metodología de arquitectura que organiza la estructura del software en cinco vistas diferentes para cubrir distintos aspectos del sistema. Cada una de estas vistas proporciona una perspectiva única que aborda preocupaciones específicas de diferentes partes interesadas. A continuación, se presenta la arquitectura de CitaConecta basada en este modelo.

## 1. Vista Lógica (Logical View)

Esta vista describe los componentes y módulos del sistema desde una perspectiva orientada a objetos o modular, enfocándose en la estructura lógica del software.

Componentes Principales:

Front-End:

Desarrollado con HTML, CSS y JavaScript, utilizando frameworks como React o Vue.js para mejorar la interactividad y la experiencia del usuario.

Back-End:

Arquitectura basada en servicios desarrollada en Node.js con Express, o Django (si es Python), manejando la lógica del negocio y las solicitudes del cliente.

Base de Datos:

- PostgreSQL o MySQL para la persistencia de datos.

- Redis (opcional) para manejo de sesiones y caché.

Servicios API:

- APIs RESTful para la interacción entre el cliente y el servidor.

- Servicios de autenticación usando JWT (JSON Web Tokens).

Módulos Principales:

- Gestión de Usuarios: Módulo para autenticación, registro y manejo de usuarios.

- Gestión de Reservas: Módulo para la creación, modificación y cancelación de citas.

- Notificaciones: Módulo para el envío de recordatorios de citas (vía email o SMS).

- Panel de Administración: Módulo para que las PYMEs gestionen sus horarios, disponibilidad y clientes.

## 2. Vista de Desarrollo (Development View)

Esta vista refleja la organización del software en módulos desde una perspectiva de desarrollo. Está orientada a los programadores y muestra cómo el sistema está organizado para facilitar su desarrollo.

Estructura del Código:

Front-End:

- Estructura basada en componentes para un desarrollo modular.

- Uso de React para implementar componentes reutilizables y escalables.

- Integración con Axios para gestionar las solicitudes HTTP al back-end.

Back-End:

- Estructura MVC (Modelo-Vista-Controlador) para separar la lógica de negocio, el manejo de datos y la presentación de los resultados.

- Node.js y Express organizados en rutas, controladores, modelos y servicios.

- Base de Datos:

- Tablas organizadas en base a las entidades del sistema: usuarios, reservas, horarios, servicios y PYMEs.

- Estructura relacional para garantizar consistencia y escalabilidad.

Pruebas:

- Jest o Mocha para realizar pruebas unitarias y de integración.

## 3. Vista de Procesos (Process View)

La vista de procesos describe el comportamiento dinámico del sistema, enfocándose en cómo los componentes interactúan y cómo se distribuyen las responsabilidades de procesamiento.

Flujo de Reservas:

1. El usuario inicia sesión o se registra.

2. El usuario navega por los servicios disponibles de las PYMEs.

3. El usuario selecciona una fecha y hora para una cita.

4. El sistema verifica la disponibilidad a través de un servicio API y confirma la reserva.

5. El sistema notifica al usuario y a la PYME vía email/SMS.

Procesos Asíncronos:

Notificaciones Automáticas:

- Servicio asíncrono encargado de enviar recordatorios de citas (por ejemplo, utilizando Node.js con cron jobs o Celery en el caso de Django).

Manejo de Carga:

- Servidor con capacidad para manejar múltiples usuarios concurrentes a través de balanceo de carga y escalabilidad horizontal (si se implementa en la nube, como AWS o Heroku).

## 4. Vista Física (Physical View)

Esta vista refleja la distribución física del software en el hardware. Se enfoca en la infraestructura de despliegue, la configuración de servidores, bases de datos y otros componentes físicos.

Servidor Web:

- Nginx o Apache configurado como proxy inverso para servir la aplicación y manejar el tráfico entre el cliente y el servidor.

Base de Datos:

- Base de datos relacional en un servidor separado o en un servicio administrado como AWS RDS o Heroku Postgres.

Despliegue:

- Uso de Docker para contenerización y despliegue consistente en cualquier entorno.

- Despliegue en la nube utilizando plataformas como AWS, Heroku o Google Cloud.

Backup y Seguridad:

- Implementación de políticas de backup de la base de datos y manejo seguro de credenciales a través de servicios como AWS Secrets Manager.

## 5. Vista de Casos de Uso (Use Case View)

Esta vista muestra los casos de uso más importantes y cómo los usuarios interactúan con el sistema. Es una vista clave para los interesados no técnicos, como los clientes y las PYMEs.

Casos de Uso Principales:

1. Registro de Usuario:

- Un usuario se registra proporcionando información personal y de contacto.

2. Reserva de Cita:

- Un cliente selecciona un servicio, fecha y hora, y realiza una reserva.

3. Notificación de Confirmación:

- El cliente recibe una notificación por correo electrónico o SMS confirmando la cita.

4. Cancelación de Cita:

- El cliente puede cancelar o reprogramar una cita antes de la fecha programada.

5. Administración de Horarios:

- Las PYMEs pueden actualizar su disponibilidad y gestionar sus citas desde el panel de administración.

El modelo 4+1 ofrece una visión integral de la arquitectura de CitaConecta, proporcionando una base sólida para el desarrollo y despliegue de la plataforma.

Esta estructura asegura que todos los aspectos clave desde el diseño lógico hasta la interacción con los usuarios y la infraestructura física están bien definidos y alineados con los objetivos del proyecto.

# Arquitectura Basada en Capas (Layered Architecture).

Otra metodología común para diseñar y describir la arquitectura de un sistema de software es Arquitectura Basada en Capas (Layered Architecture). Este enfoque organiza el sistema en capas que interactúan entre sí de manera jerárquica, donde cada capa tiene una responsabilidad específica y sólo puede comunicarse con las capas adyacentes. A continuación te explico cómo podría estructurarse el proyecto CitaConecta usando este método.

Arquitectura Basada en Capas de CitaConecta

La arquitectura en capas se compone de varios niveles, cada uno con responsabilidades bien definidas, lo que permite un desarrollo modular y facilita el mantenimiento y escalabilidad del sistema. Las capas más comunes que podemos identificar para el proyecto CitaConecta son:

## 1. Capa de Presentación (Front-End Layer)

Responsabilidad: Proporcionar la interfaz gráfica con la que los usuarios interactúan. Se encarga de recibir las entradas de los usuarios y mostrar los resultados de las operaciones del sistema.

Tecnologías:

- HTML/CSS/JavaScript para la construcción de la interfaz de usuario.

- Frameworks como React o Vue.js para una experiencia interactiva y dinámica.

- Axios o Fetch API para interactuar con el backend a través de solicitudes HTTP.

Funcionalidades Clave:

- Formularios de registro y login.

- Interfaz para la selección de citas.

- Panel de administración para las PYMEs.

## 2. Capa de Servicios y API (Service/API Layer)

Responsabilidad: Manejar la lógica de negocio y servir de intermediario entre la capa de presentación y la capa de acceso a datos. En esta capa, las reglas de negocio, validaciones, y operaciones con los datos son ejecutadas.

Tecnologías:

- Node.js con Express (o Django si se usa Python) para la creación de la API REST.

- JWT (JSON Web Tokens) para la autenticación y manejo de sesiones de usuario.

Funcionalidades Clave:

- Gestión de usuarios (registro, autenticación, recuperación de contraseñas).

- Creación, modificación y cancelación de citas.

- Envío de notificaciones automáticas (e-mail/SMS).

## 3. Capa de Lógica de Negocio (Business Logic Layer)

Responsabilidad: Procesar y aplicar las reglas de negocio, asegurar la integridad de las transacciones y manejar las operaciones de datos requeridas por el sistema.

Tecnologías:

- Node.js/Django, donde se escriben las reglas de negocio.

- Middleware para la gestión de la autenticación y autorización.

Funcionalidades Clave:

- Validación de la disponibilidad de citas.

- Gestión de políticas de negocio como el tiempo mínimo de cancelación o cambios en citas.

- Enrutamiento de notificaciones según el tipo de usuario (PYME o cliente).

## 4. Capa de Acceso a Datos (Data Access Layer)

Responsabilidad: Acceder, modificar y gestionar los datos persistentes almacenados en la base de datos. Actúa como una abstracción que proporciona la interacción con las bases de datos y asegura que el sistema esté desacoplado de los detalles de implementación de la base de datos.

Tecnologías:

- ORM (Object-Relational Mapping) como Sequelize (para Node.js) o Django ORM (en caso de usar Django).

- SQL/NoSQL: Bases de datos relacionales como PostgreSQL o MySQL, o una base de datos NoSQL como MongoDB si se busca más flexibilidad.

Funcionalidades Clave:

- Gestión de reservas, usuarios, y horarios.

- Actualización de la base de datos de manera eficiente y segura.

- Manejo de transacciones para garantizar la integridad de los datos.

## 5. Capa de Integración (Integration Layer)

Responsabilidad: Facilitar la integración con servicios externos o sistemas de terceros, como proveedores de SMS, servicios de pago o plataformas de correo electrónico.

Tecnologías:

- API Externas para el envío de notificaciones (por ejemplo, Twilio para SMS o SendGrid para correos electrónicos).

- Servicios de Pago si el sistema requiere pagos en línea para las reservas.

Funcionalidades Clave:

- Comunicación con sistemas de notificaciones externas.

- Integración con plataformas de pago (si se requiere).

## 6. Capa de Infraestructura (Infrastructure Layer)

Responsabilidad: Proveer los servicios y la infraestructura necesaria para que el sistema funcione, como la gestión de servidores, almacenamiento y redes.

Tecnologías:

- Docker para contenerización del entorno.

- Servicios de la nube como AWS, Heroku o Google Cloud para el despliegue.

- Nginx como proxy inverso para servir las solicitudes web y balancear la carga.

Funcionalidades Clave:

- Despliegue y escalabilidad del sistema.

- Monitoreo y gestión de servidores y redes.

- Seguridad y gestión de acceso a los recursos.

## Beneficios de la Arquitectura en Capas:

1. Modularidad: Cada capa se desarrolla de manera independiente, lo que permite un mejor mantenimiento y actualización del sistema.

2. Reutilización: Los componentes de cada capa pueden ser reutilizados por otros sistemas o servicios, mejorando la eficiencia del desarrollo.

3. Separación de responsabilidades: Cada capa tiene una función clara, lo que facilita el diseño y desarrollo colaborativo, y permite la delegación de tareas entre equipos especializados.

4. Escalabilidad: Esta arquitectura permite que el sistema sea fácilmente escalable horizontalmente, agregando más servidores o servicios en capas específicas cuando sea necesario.

La arquitectura en capas proporciona una estructura sólida y flexible para el desarrollo del proyecto CitaConecta. A través de la separación clara de responsabilidades en cada capa, el sistema es más fácil de mantener, escalar y extender en el futuro, lo que lo convierte en una elección ideal para este tipo de plataformas web de reservas automatizadas.