**Guía Informe de Proyecto**

**Unidad 1: Levantamiento de requerimientos y diseño de interfaces**

**Asignatura: Taller de Diseño y desarrollo de Soluciones**

**Sección:** D-B50-N4-P12-C1/D

**Nombre del docente:** Geraldo Valenzuela Polanco

**Nombre de los integrantes del grupo: Pablo Leon Ponce**

**Issa Peña Mondaca**

**Eduardo Rapimán Acuña**

**Fecha de entrega: 25/09/2024**

**Contenido**

[I. Introducción 3](#_heading=h.gjdgxs)

[II. Objetivo 3](#_heading=h.30j0zll)

[III. Desarrollo 3](#_heading=h.1fob9te) al 12

[IV. Conclusiones](#_heading=h.3znysh7)  13

[V. Referencias bibliográficas 13](#_heading=h.2et92p0)

1. **Introducción**

Este informe es la segunda parte del proyecto de creación de una aplicación web para la gestión de citas en una peluquería. En esta etapa, se abordarán los patrones de arquitectura, los estándares de programación segura, la definición de un sistema de almacenamiento en la nube y el diseño de un plan de pruebas. Estas mejoras buscan asegurar que la solución sea robusta, segura y eficiente, garantizando una experiencia de usuario optimizada y la satisfacción de los clientes.

1. **Objetivo**

El objetivo de este informe es avanzar en el desarrollo del sistema, detallando la estructura de su arquitectura, aplicando prácticas de seguridad efectivas, eligiendo un sistema de almacenamiento en la nube adecuado y elaborando un plan de pruebas que asegure su correcto funcionamiento.

**III. Desarrollo**

1. Definir los patrones de arquitectura de la solución.

2. Incluir estándares de programación segura.

3. Definir almacenamiento de datos en cloud.

4. Diseñar el plan de pruebas.

### 1. Definir los Patrones de Arquitectura de la Solución

#### Selección del Patrón de Arquitectura

Se eligió el patrón **Vista-Vista-Modelo (VVM)** para estructurar el sistema. Esta arquitectura permite una separación clara entre la lógica de negocio, la interfaz de usuario y el estado de la aplicación, proporcionando una mayor flexibilidad para manejar la presentación y los datos dinámicos de la interfaz. El uso de VVM facilita el desarrollo de interfaces modernas y reactivas, así como el mantenimiento futuro del sistema.

**Propuesta de Arquitectura: Arquitectura en 3 Capas (Cliente-Servidor, Vista-Vista-Modelo - VVM)**

La arquitectura propuesta se basa en tres capas principales:

1. **Capa Cliente (Frontend):**
   * Implementa la **Vista** y el **ViewModel**, asegurando que la interfaz sea interactiva y responda dinámicamente a las interacciones del usuario.
   * Utiliza frameworks modernos como **Vue.js**, **React**, o **Angular** para el desarrollo de componentes de interfaz.
   * El **ViewModel** se encarga de manejar el estado de la interfaz, las validaciones y las interacciones con la API del servidor.
2. **Capa Servidor (Backend):**
   * Implementa el **Modelo**, gestionando la lógica de negocio y el acceso a la base de datos.
   * Exposición de datos y funcionalidades mediante una API REST o GraphQL, proporcionando flexibilidad para integrarse con múltiples clientes.
3. **Capa de Datos:**
   * Gestiona la persistencia de información, como los clientes, servicios y reservas.
   * Implementada con una base de datos relacional o no relacional según las necesidades del sistema.

Esta arquitectura basada en VVM permite un desarrollo modular y escalable, mientras mejora la experiencia del usuario mediante la integración de interfaces dinámicas y modernas.

**Capa de Acceso a Datos (Base de Datos)**:

Esta capa se encarga de interactuar directamente con la base de datos para manejar y persistir la información necesaria para el sistema. En el contexto del patrón **Vista-Vista-Modelo (VVM)**, la capa de datos es accesible a través de la lógica del **Modelo**, que centraliza las operaciones de negocio y la gestión de datos.

**Funciones de la Capa de Acceso a Datos:**

* **Almacenamiento y recuperación de información:**
  + Gestiona los datos de citas, clientes, servicios ofrecidos, horarios disponibles y usuarios del sistema.
* **Estructuración de datos:**
  + Garantiza que la información esté organizada y optimizada para consultas rápidas y eficientes.

#### Justificación

Se eligió el modelo **Vista-Vista-Modelo (VVM)** para desarrollar el sistema porque ofrece muchas ventajas, como su diseño modular, flexible y fácil de usar. Esto lo hace ideal para una aplicación como un sistema de reservas en línea para una peluquería. Este enfoque permite desarrollar el sistema de manera más sencilla, crear una experiencia de usuario moderna y asegurar que sea fácil de actualizar o ampliar en el futuro.

## 2. Definición de Privilegios de Acceso

### Los privilegios de acceso están basados en los roles previamente definidos, asegurando que cada usuario tenga acceso únicamente a la información y funcionalidades necesarias para cumplir con sus responsabilidades. Esto previene accesos no autorizados y garantiza que los datos sensibles se mantengan protegidos.

### Privilegios por Rol

#### Administrador

### CRUD Completo: El administrador tiene acceso completo a todos los datos del sistema, con privilegios de creación, lectura, actualización y eliminación (CRUD).

#### Empleado

### Acceso Limitado: Solo puede acceder a su propia información y citas.

### Gestión Propia: Permiso para modificar su disponibilidad y gestionar las citas que tiene asignadas.

### Sin Acceso Administrativo: No tiene acceso a la información de otros empleados ni a la gestión de usuarios.

#### Cliente

### Gestión de citas: Puede agendar y cancelar citas de acuerdo con la disponibilidad de los empleados.

### Sin Acceso Administrativo: No tiene acceso a información administrativa ni a los datos de otros usuarios.

### 3. Almacenamiento de Datos en la Nube

#### Investigación de Servicios Cloud

Se investigaron varias opciones de almacenamiento en la nube, incluyendo Google Cloud, Amazon Web Services (AWS) y Microsoft Azure. Cada opción ofrece características únicas, pero también tiene sus particularidades en cuanto a precios, escalabilidad y facilidad de integración.

##### Ejemplo Comparativo

* Google Cloud: Es una opción accesible con precios basados en el uso, lo que lo hace ideal para proyectos pequeños y en crecimiento. Además, es fácil de integrar con aplicaciones web modernas gracias a su API amigable.
* AWS: Aunque es altamente escalable, la estructura de precios es más compleja, lo que podría ser un reto para un proyecto con un equipo pequeño. Sin embargo, AWS ofrece más servicios especializados para empresas que necesitan funciones avanzadas.
* Microsoft Azure: Es especialmente útil para empresas que ya usan otros productos de Microsoft, pero requiere más esfuerzo para integrar en proyectos que no dependen de este ecosistema.

#### Selección del Servicio

Se eligió Google Cloud como el servicio de almacenamiento debido a su modelo de precios flexible, su facilidad de uso y su escalabilidad para proyectos pequeños como este. Además, su integración sencilla con sistemas web hace que sea una solución rápida y eficiente para la gestión de datos en este contexto.

##### Justificación

Google Cloud fue seleccionado porque su modelo de pago por uso es más accesible para proyectos que aún están en fase de desarrollo. Por ejemplo, si en el futuro la peluquería decide expandir su base de clientes y ofrecer más servicios, Google Cloud puede escalar sin necesidad de grandes cambios en la infraestructura. AWS, aunque poderoso, tiene una curva de aprendizaje más pronunciada y sería innecesariamente costoso para el tamaño actual del proyecto.

### 4. Identificar los Criterios de Aceptación de Cada Historia de Usuario

#### Historia de Usuario 1: Ver fechas y horas disponibles para agendar una cita

* El cliente puede ver un calendario con fechas y horas disponibles.
* Solo se muestran fechas y horas no reservadas.
* Las fechas pasadas no son seleccionables.

#### Historia de Usuario 2: Agendar una cita

* El cliente puede seleccionar una fecha y hora y completar los detalles del servicio.
* El sistema registra la cita en la base de datos.
* El cliente recibe una confirmación visual de que su cita ha sido agendada con éxito.

#### Historia de Usuario 3: Ver, modificar o cancelar citas agendadas

* El cliente puede visualizar todas sus citas agendadas.
* El cliente puede modificar o cancelar citas.
* Se muestra una confirmación después de modificar o cancelar una cita.

#### Historia de Usuario 4: Gestión de citas por parte del administrador

* El administrador puede ver todas las citas agendadas.
* El administrador puede modificar o eliminar citas.
* Las modificaciones del administrador son visibles para los clientes.

#### Historia de Usuario 5: Definir la disponibilidad del calendario

* El administrador puede definir horarios y días laborables.
* Las actualizaciones de disponibilidad se reflejan inmediatamente para los clientes.

### Identificar las Funcionalidades que Dan Solución a las Historias de Usuario

* Calendario interactivo: Mostrar fechas y horas disponibles.
* Formulario de agendamiento: Permitir al cliente seleccionar una fecha y registrar sus datos para agendar la cita.
* Gestión de citas (cliente): Visualización, modificación y cancelación de citas.
* Panel de administración: Gestión de citas y disponibilidad por parte del administrador.
* Base de datos: Almacenamiento de citas, clientes y horarios.

### Definir la Estrategia de Pruebas

#### 1. Pruebas Funcionales

Estas pruebas verificarán que las funcionalidades del sistema cumplan con los requisitos establecidos en las historias de usuario.

* Tipos de Pruebas Funcionales:
  + Pruebas de unidad: Probar individualmente las funciones o métodos del sistema (p. ej., verificar que una función que calcula la disponibilidad de citas devuelva resultados correctos).
  + Pruebas de integración: Asegurar que los módulos (p. ej., calendario, base de datos, y formularios) funcionen correctamente al interactuar entre sí.
  + Pruebas de sistema: Verificar que todo el sistema funcione de extremo a extremo, desde la visualización de citas hasta el agendamiento y la gestión por parte del administrador.

#### 2. Pruebas No Funcionales

Evaluarán aspectos de la calidad del sistema que no estén directamente relacionados con una funcionalidad específica.

* Tipos de Pruebas No Funcionales:
  + Pruebas de rendimiento: Evaluar el tiempo de respuesta del sistema bajo diferentes cargas (p. ej. cuando varios usuarios están agendando citas simultáneamente).
  + Pruebas de usabilidad: Verificar que el sistema sea fácil de usar y que los clientes puedan agendar y gestionar sus citas sin dificultad.

#### 3. Pruebas Integrales

Se evaluará la integración de las distintas partes del sistema. Estas pruebas se realizarán después de que todos los módulos hayan sido desarrollados e integrados.

* Tipos de Pruebas Integrales:
  + Pruebas de integración del calendario y la base de datos: Verificar que las citas se guarden y actualicen correctamente.
  + Pruebas de flujo completo: Comprobar que todo el flujo desde la selección de una cita hasta su registro en la base de datos funcione sin errores.

#### 4. Pruebas de Arquitectura

Estas pruebas validarán que la arquitectura del sistema esté correctamente implementada y cumpla con los principios de diseño definidos.

* Tipos de Pruebas de Arquitectura:
  + Pruebas de separación de capas: Asegurar que la lógica de negocio esté bien separada de la capa de presentación y que las APIs REST sean el único punto de comunicación entre el front-end y el back-end.
  + Pruebas de modularidad: Comprobar que cada módulo (p. ej., gestión de citas, calendario, panel de administración) sea independiente y que pueda ser probado aisladamente.

#### 5. Pruebas de Sistema

Se realizan para verificar que todo el sistema, con todas sus componentes, funcionen correctamente como una unidad.

* Tipos de Pruebas de Sistema:
  + Pruebas de extremo a extremo: Validar que todo el flujo del usuario, desde la visualización del calendario hasta la modificación de citas, funcione sin errores.

1. **Conclusiones**

* El desarrollo de una aplicación web de agendamiento de citas, requiere un enfoque estructurado y bien pensado que abarca tres componentes fundamentales: la arquitectura de la solución, el almacenamiento de datos y el plan de pruebas. A lo largo de este trabajo se han definido estas áreas clave, asegurando un sistema escalable, funcional y seguro.
* Modularidad y Escalabilidad: El sistema debe estar diseñado de manera que sea fácil de mantener, escalar y agregar nuevas funcionalidades conforme crece el negocio de la peluquería.
* Seguridad y Privacidad: Dado que el sistema maneja datos sensibles, las pruebas de seguridad y la selección adecuada de soluciones en la nube son aspectos fundamentales para garantizar la confianza de los usuarios.
* Experiencia del Usuario: La usabilidad y accesibilidad son pilares esenciales. Un sistema que ofrezca una interfaz amigable, rápida y accesible para todo tipo de usuarios será clave para su adopción.
* La arquitectura, almacenamiento y plan de pruebas diseñados para el sitio web de agendamiento de citas ofrecen una solución sólida y escalable, alineada con las necesidades del negocio. La adopción de herramientas modernas como el almacenamiento en la nube y la automatización de pruebas contribuirá a mejorar la calidad y eficiencia del sistema. Al seguir evolucionando la plataforma, se recomienda explorar nuevas tecnologías y enfoques que garanticen que el sistema continúe siendo robusto, seguro y fácil de usar para los clientes y administradores.

1. **Referencias bibliográficas**

[**https://www.watchguard.com/es/wgrd-news/blog/consejos-para-practicar-la-codificacion-segura-y-ser-un-programador-responsable-1**](https://www.watchguard.com/es/wgrd-news/blog/consejos-para-practicar-la-codificacion-segura-y-ser-un-programador-responsable-1)

[**https://es.abstracta.us/blog/guia-crear-estrategia-pruebas-software-adecuada/**](https://es.abstracta.us/blog/guia-crear-estrategia-pruebas-software-adecuada/)