

Universidad ORT Uruguay

Facultad de Ingeniería

Escuela de Tecnología

PROGRESSIVE WEB AP PARA CENTRO TIFÉRET

**Entregado como requisito para la obtención del título
de Analista Programador**

Facundo Orihuela – 293326

Juan Núñez – 293652

Diego Mazas – 294061

N5C

Tutor: Carlos Berruti

2024


Declaración de autoría

Nosotros, Juan Núñez, Diego Mazas y Facundo Orihuela, declaramos que el trabajo que se presenta en esa obra es de nuestra propia mano. Podemos asegurar que:

- La obra fue producida en su totalidad mientras realizábamos el proyecto integrador de la carrera de Analista Programador de 2024.
- Cuando hemos consultado el trabajo publicado por otros, lo hemos atribuido con claridad.
- Cuando hemos citado obras de otros, hemos indicado las fuentes. Con excepción de estas citas, la obra es enteramente nuestra.
- En la obra, hemos acusado recibo de las ayudas recibidas.
- Cuando la obra se basa en trabajo realizado conjuntamente con otros, hemos explicado claramente qué fue contribuido por otros, y qué fue contribuido por nosotros.
- Ninguna parte de este trabajo ha sido publicada previamente a su entrega, excepto donde se han realizado las aclaraciones correspondientes.

Aclaración de firma: Facundo Orihuela

Fecha del día de la entrega: 21/10/2024

Firma: 


Aclaración de firma: Juan Núñez

Fecha del día de la entrega: 21/10/2024

Firma: 

Aclaración de firma: Diego Mazas

Fecha del día de la entrega: 21/10/2024

Firma: 

Agradecimientos

Queremos expresar nuestro más sincero agradecimiento a Daniel Bentancor, quien nos dedicó su tiempo y conocimientos, ayudándonos comprender las motivaciones y expectativas del cliente a lo largo del desarrollo de este proyecto. Su compromiso y paciencia fueron fundamentales para nuestro avance.

Asimismo, agradecemos a nuestro tutor, Carlos Berruti, por su constante apoyo, orientación y confianza en nosotros. Su experiencia y consejos nos permitieron mejorar continuamente y alcanzar los objetivos planteados. A ambos, les debemos gran parte del éxito de este trabajo.

Abstract

El proyecto busca renovar por completo la página web del Centro Terapéutico Tiféret, incorporando nuevas funcionalidades para mejorar su presencia online y hacerla más funcional. El cliente nos solicitó modernizar el diseño, integrar una tienda online y un sistema de membresías que brinden una experiencia completa a los usuarios.

La tienda ofrecerá una amplia gama de productos, desde libros físicos y digitales hasta cursos, podcasts y material terapéutico. Los usuarios podrán comprar fácilmente desde la web. Además, el sistema de membresías proporcionará beneficios exclusivos como descuentos y acceso a ofertas especiales. También se habilitará el registro y reserva online para eventos y servicios.

El rediseño no solo busca modernizar la apariencia, sino también optimizar la usabilidad, garantizando una navegación fluida y sencilla para los usuarios.

Palabras clave

Tienda online

E-commerce

Membresías

Rediseño web

Centro Terapéutico Tiféret

Wix

Venta de libros

Cursos online

Podcasts

Audiolibros

Contenido digital

Reservas online

Eventos

Grupos interactivos

Experiencia de usuario

Modernización web

Progressive Web App

Índice

| | |
|--|----|
| 1. Introducción | 9 |
| 2. Descripción del cliente | 10 |
| 2.1 Gestión de stakeholders | 10 |
| 2.1.1 Product Owner (Diego Bentancor)..... | 10 |
| 2.1.2 Scrum Master (Juan Núñez) | 11 |
| 2.1.3 Desarrolladores (Facundo Orihuela, Juan Núñez y Diego Mazas) | 11 |
| 2.1.4 Centro Terapéutico Tiféret (Cliente)..... | 11 |
| 2.1.5 Usuarios Finales del Centro Terapéutico Tiféret | 12 |
| 2.1.6 Equipo de Administración del Centro | 12 |
| 2.1.7 Estrategia General de Gestión | 12 |
| 3. Descripción del problema | 13 |
| 4. Lista de necesidades | 13 |
| 5. Objetivos | 15 |
| 5.1 Implementar un sistema de e-commerce..... | 15 |
| 5.2 Crear un sistema de membresías pagas..... | 16 |
| 5.3 Establecer un sistema de reservas online..... | 16 |
| 5.4 Desarrollar un panel administrativo para la gestión de contenido | 16 |
| 5.5 Desarrollar la web como una Progressive Web App (PWA) | 17 |
| 6. Requerimientos funcionales | 17 |
| 7. Requerimientos no funcionales: | 22 |
| 8. Tecnologías Seleccionadas y Justificación | 23 |
| 8.1 Progressive Web App (PWA)..... | 23 |
| 8.1.1 Razones para utilizar una PWA:..... | 24 |
| 8.2 React..... | 24 |
| 8.2.1 Razones para usar React: | 24 |
| 8.3 Node.js con Express | 24 |
| 8.3.1 Razones para usar Node.js con Express: | 24 |
| 8.4 Tailwind CSS | 25 |
| 8.4.1 Razones para usar Tailwind: | 25 |
| 8.5 Jira | 25 |
| 8.5.1 Razones para usar Jira:..... | 25 |
| 8.6 MySQL..... | 26 |
| 8.6.1 Razones para usar MySQL: | 26 |
| 9. Tecnologías alternativas y justificación de la selección | 27 |

| | | |
|--------|---|----|
| 9.1.1 | Angular o Vue.js (en lugar de React): | 27 |
| 9.1.2 | PHP con Laravel (en lugar de Node.js con Express):..... | 27 |
| 9.1.3 | Bootstrap (en lugar de Tailwind CSS): | 27 |
| 9.1.4 | Trello o Asana (en lugar de Jira): | 27 |
| 9.1.5 | Comparación entre CMS y Tecnologías Personalizadas | 27 |
| 9.1.6 | Limitaciones de los CMS: | 28 |
| 9.1.7 | Ventajas de las Tecnologías Personalizadas (React, Node.js, Tailwind, etc.) 28 | |
| 9.1.8 | Conclusión | 29 |
| 10. | Metodología de Trabajo: Scrum | 29 |
| 10.1 | Roles del equipo..... | 29 |
| 10.2 | ¿Por qué elegimos Scrum? | 30 |
| 10.3 | Otras ventajas de Scrum | 30 |
| 10.4 | Conclusión | 31 |
| 11. | Plan de SCM..... | 31 |
| 11.1 | Gestión de Cambios | 31 |
| 11.2 | Versionado de Código | 32 |
| 11.3 | Herramientas a Utilizar | 32 |
| 12. | Plan de Aseguramiento de la Calidad del Software (SQA) | 32 |
| 12.1 | Estándares y Buenas Prácticas de Desarrollo | 33 |
| 12.2 | Plan de Testing | 33 |
| 12.3 | Herramientas a Utilizar | 34 |
| 13. | Plan de Capacitación..... | 35 |
| 13.1 | Capacitación en Tecnologías:..... | 35 |
| 13.2 | Capacitación en Jira: | 35 |
| 13.3 | Esfuerzo estimado:..... | 36 |
| 13.4 | Capacitación al Cliente:..... | 36 |
| 14. | Gestión de riesgos | 36 |
| 14.1 | Identificación de Riesgos | 36 |
| 14.2 | Plan de Mitigación y Contingencia | 37 |
| 15. | Supuestos | 39 |
| 16. | Arquitectura | 39 |
| 17. | Alcance del producto | 39 |
| 18. | Cronograma | 40 |
| 18.1 | Inicio de anteproyecto | 41 |
| 18.1.1 | Sprint 1 | 41 |

| | | |
|---------|---|----|
| 18.1.2 | Sprint 2 | 41 |
| 18.1.3 | Sprint 3 | 42 |
| 18.2 | Inicio de desarrollo | 43 |
| 18.2.1 | Sprint 4 | 43 |
| 18.2.2 | Sprint 5 | 43 |
| 18.2.3 | Sprint 6 | 44 |
| 18.2.4 | Sprint 7 | 44 |
| 18.2.5 | Sprint 8 | 45 |
| 18.2.6 | Sprint 9 | 45 |
| 18.2.7 | Sprint 10 | 46 |
| 18.2.8 | Sprint 11 | 46 |
| 18.2.9 | Sprint 12 | 47 |
| 18.2.10 | Sprint 13 | 48 |
| 18.2.11 | Sprint 14 | 48 |
| 18.2.12 | Sprint 15 | 48 |
| 18.2.13 | Sprint 16 | 49 |
| 18.2.14 | Sprint 17 | 49 |
| 18.2.15 | Sprint 18 | 50 |
| 18.2.16 | Sprint 19 | 50 |
| 19. | Bibliografía | 52 |
| 20. | Anexos | 53 |
| 20.1 | Anexo 1: Guía de implementación de PWA | 53 |
| 20.1.1 | Crear la aplicación React con el soporte PWA activado | 53 |
| 20.1.2 | Instalar los archivos necesarios | 53 |
| 20.1.3 | Configurar `manifest.json` | 53 |
| 20.1.4 | Habilitar el Service Worker | 54 |
| 20.1.5 | Asegurarte de que tu app sea responsive | 55 |
| 20.1.6 | Construir y probar tu PWA | 55 |
| 20.1.7 | Testing PWA | 55 |

1. Introducción

El Centro Terapéutico Tiféret es una organización comprometida con el bienestar integral de las personas, combinando prácticas terapéuticas tradicionales con enfoques innovadores como la psicoterapia, los círculos terapéuticos, la Kabala viva y las integraciones grupales. Su objetivo principal es proporcionar herramientas y recursos que promuevan el crecimiento personal y espiritual, adaptándose a las necesidades individuales y colectivas de sus usuarios. El centro ha logrado consolidarse como un espacio de referencia para aquellos que buscan un equilibrio emocional, mental y espiritual, apoyando a sus clientes a través de diversas terapias y actividades que fomentan el autoconocimiento y la conexión con los demás.

Actualmente, Tiféret cuenta con una página web alojada en la plataforma Wix. Esta página, aunque funcional en términos de presentación de información, se limita a ofrecer contenido estático, donde los usuarios pueden leer sobre los servicios y actividades del centro, pero no interactuar de manera activa con la página. No existe la posibilidad de realizar reservas, comprar productos, o inscribirse en actividades directamente desde el sitio web. Esto representa una limitación significativa en un mundo cada vez más digital, donde los usuarios esperan poder interactuar de manera más dinámica y efectiva con las plataformas online.

Ante esta situación, el Centro Terapéutico Tiféret ha solicitado la creación de una nueva página web que no solo mantenga la información ya existente, sino que también expanda sus capacidades para permitir una mayor interacción con los usuarios. La propuesta del proyecto incluye la incorporación de un sistema de e-commerce, donde los usuarios puedan comprar productos y servicios ofrecidos por el centro, tales como libros, cursos online, grabaciones de clases, podcasts, audiolibros y más. Además, se integrará un sistema de membresías pagas, lo que permitirá a los usuarios acceder a contenido exclusivo y ofertas especiales. Este sistema de membresías busca no solo monetizar algunos de los servicios ofrecidos por Tiféret, sino también fomentar la fidelización y el compromiso de los usuarios con el centro.

Uno de los elementos clave de la nueva página será la implementación de un sistema de usuarios registrados. Los usuarios, ya sean con membresía paga o no, podrán crear un perfil personal donde gestionarán sus actividades dentro del centro. Esto incluye la capacidad de registrarse a eventos, realizar reservas online, gestionar sus compras y acceder a sus actividades pasadas. Este sistema no solo permitirá a los usuarios tener un control más organizado de sus interacciones con el centro, sino que también facilitará la comunicación entre Tiféret y sus clientes, brindando un servicio más personalizado y eficiente.

Además, se incluirá un panel administrativo para los responsables del centro, desde donde podrán gestionar el contenido de la página de manera autónoma. Este panel permitirá agregar noticias, eventos, productos, precios, así como gestionar las reservas y grupos. La capacidad de modificar el contenido en tiempo real es esencial para mantener la página actualizada y relevante, permitiendo que Tiféret se adapte rápidamente a las necesidades de sus usuarios y a las novedades en su oferta de servicios. Asimismo, el centro podrá comunicarse de manera más directa y efectiva con

sus clientes, ya sea para enviar notificaciones sobre eventos, ofertas, o cualquier otra información relevante.

En resumen, la modernización de la página web del Centro Terapéutico Tiféret representa un paso crucial para la expansión de sus servicios y la mejora de su presencia digital. A través de la incorporación de un sistema de e-commerce, membresías pagas, y usuarios registrados, la nueva página no solo ofrecerá una mayor funcionalidad a sus usuarios, sino que también permitirá al centro gestionar de manera más eficiente su oferta de servicios. Esta transformación digital permitirá a Tiféret continuar cumpliendo su misión de proporcionar bienestar integral a las personas, pero ahora a través de una plataforma moderna, interactiva y accesible, que ampliará su alcance y fortalecerá su relación con su comunidad de usuarios tanto a nivel local como global.

La nueva página web del Centro Terapéutico Tiféret será desarrollada como una Progressive Web App (PWA), lo que ofrecerá una experiencia similar a la de una aplicación nativa, pero accesible desde cualquier navegador sin necesidad de descargarla. Las PWAs permiten uso sin conexión, mejorando la accesibilidad al contenido, como grabaciones o podcasts. Además, son altamente responsivas, adaptándose a diferentes dispositivos para asegurar una navegación fluida. La PWA también se beneficiará de carga rápida, optimizando el rendimiento y la interacción del usuario, lo que garantiza una experiencia eficiente. Al ser una aplicación segura y actualizable automáticamente, los usuarios siempre contarán con la versión más reciente sin interrupciones.

2. Descripción del cliente

Centro Tiféret es una empresa unipersonal que consiste en un centro terapéutico que funciona como arrendamiento de espacios para distintos profesionales ya sea como consultorios psicoterapéuticos y como salones para trabajos en grupo (talleres, clases de yoga, eventos, etc.)

En Tiféret trabajan alrededor de cuarenta profesionales de diferentes disciplinas y cuenta con un alcance de entre trescientos y quinientos clientes mensuales.

Tiféret cuenta a su vez con una directora general, un equipo de coordinación integrado por tres personas y un equipo encargado del área de comunicación.

2.1 Gestión de stakeholders

2.1.1 Product Owner (Diego Bentancor)

Rol: Diego es el representante del cliente y es responsable de definir los requerimientos del producto. También prioriza las funcionalidades y toma decisiones clave sobre la dirección del proyecto.

Intereses y expectativas: Asegurarse de que el producto final refleje las necesidades y expectativas del Centro Terapéutico Tiféret. Buscará que las funcionalidades prioritarias estén bien implementadas y que se cumplan los plazos.

Nivel de influencia: Alto. Es quien toma decisiones finales sobre la priorización y aprobación de entregables.

Gestión: Se mantendrá una comunicación constante con él mediante reuniones bisemanales para revisar el progreso, discutir cambios y ajustar el backlog según sea necesario.

2.1.2 Scrum Master (Juan Núñez)

Rol: Facilita el proceso Scrum, garantiza el cumplimiento de la metodología y ayuda a remover impedimentos que puedan afectar el desarrollo.

Intereses y expectativas: Asegurar que el equipo esté alineado con las prácticas ágiles, mantenga una comunicación fluida y que las ceremonias Scrum se lleven a cabo correctamente.

Nivel de influencia: Medio. Aunque no toma decisiones de producto, su influencia en la organización del equipo es crucial para el éxito del proyecto.

Gestión: A través de reuniones diarias, planificación de sprints y retrospectivas semanales, Juan se encargará de monitorear el estado del proyecto y asegurar el cumplimiento de los objetivos.

2.1.3 Desarrolladores (Facundo Orihuela, Juan Núñez y Diego Mazas)

Rol: Son los encargados de implementar las funcionalidades del producto, siguiendo las indicaciones del Product Owner y los lineamientos del Scrum Master.

Intereses y expectativas: Desarrollar el producto de manera eficiente y con alta calidad, utilizando las tecnologías seleccionadas. También esperan tener un entorno de trabajo ágil y organizado que facilite el desarrollo.

Nivel de influencia: Medio. Aportan conocimientos técnicos y propuestas para mejorar el desarrollo, aunque las decisiones finales recaen en el Product Owner.

Gestión: Se mantendrán reuniones de planificación de sprint y reuniones diarias (Daily Scrum) para asegurar que las tareas se estén realizando según lo planeado y se discutan posibles problemas o mejoras.

2.1.4 Centro Terapéutico Tiféret (Cliente)

Rol: El cliente final que utilizará y beneficiará de la solución desarrollada. Está representado por el Product Owner en la toma de decisiones clave.

Intereses y expectativas: Que la solución final cumpla con sus necesidades de ofrecer servicios digitales, expandir su oferta a usuarios remotos, y mejorar la experiencia de usuario en línea.

Nivel de influencia: Alto. Aunque las decisiones se delegan al Product Owner, es esencial cumplir con sus expectativas generales sobre el producto final.

Gestión: Se proporcionarán demostraciones periódicas del progreso del proyecto, recibiendo su retroalimentación indirectamente a través del Product Owner.

2.1.5 Usuarios Finales del Centro Terapéutico Tiféret

Rol: Son los individuos que utilizarán la nueva plataforma digital, ya sea para acceder a productos, servicios, o contenidos exclusivos del centro.

Intereses y expectativas: Tener una experiencia de usuario fluida y sin fricciones, con acceso rápido y sencillo a los servicios del centro, incluyendo reservas, compras y suscripciones.

Nivel de influencia: Bajo. Aunque no están directamente involucrados en el desarrollo del proyecto, la experiencia del usuario final será el indicador clave del éxito del producto.

Gestión: El equipo realizará pruebas de usuario y encuestas (en colaboración con el cliente) para asegurar que las funcionalidades cumplen con las expectativas de los usuarios finales.

2.1.6 Equipo de Administración del Centro

Rol: Son quienes gestionarán la plataforma desde el backend, añadiendo contenido, administrando productos y servicios, y gestionando las membresías y usuarios.

Intereses y expectativas: Tener un panel administrativo sencillo de usar, eficiente y con acceso rápido a la gestión de las actividades del centro.

Nivel de influencia: Medio. Aunque no están directamente involucrados en el desarrollo diario, su feedback será fundamental en la definición y mejora de las funcionalidades administrativas.

Gestión: Se proporcionarán capacitaciones al equipo administrativo sobre el uso del panel, y su feedback será recogido y discutido con el Product Owner para ajustes posteriores.

2.1.7 Estrategia General de Gestión

Comunicación Regular: Se mantendrán reuniones bisemanales con el Product Owner para revisar el progreso, discutir cambios y obtener feedback. Estas reuniones serán esenciales para la toma de decisiones.

Reuniones Diarias: Con el equipo de desarrollo, se realizarán reuniones diarias cortas para revisar avances, impedimentos y ajustar prioridades si es necesario.

Demostraciones y Feedback: Cada sprint concluirá con una demostración del incremento del producto al Product Owner, permitiendo obtener feedback temprano y realizar ajustes antes de la siguiente fase.

Capacitación al Cliente: Se organizarán sesiones de capacitación para el equipo administrativo del centro, quienes utilizarán el panel de gestión. Esto se llevará a cabo en

las fases finales del proyecto, garantizando que estén preparados para utilizar el producto final de manera eficiente.

3. Descripción del problema

El problema planteado por el centro Tiféret radica principalmente en la escasez de funcionalidades digitales para el Centro y la falta de modernización que brindan éstas, como la posibilidad de hacer agendas mediante la web, centralización de la información, tienda online, etc. lo cual repercute directamente tanto en el alcance de Tiféret como en los ingresos mensuales del Centro.

Problemas para resolver:

- Problemas de alcance: Centro Tiféret estaba teniendo problemas para llegar a más gente y en diferentes partes del mundo ya que hasta ahora, funciona todo a nivel local, tanto las diferentes actividades, podcast o mismo la tienda funcionan muy centralizadas en el centro físico y ellos quisieran globalizarse más y poder llegar a diferentes partes del mundo.
- Problemas de gestión: Actualmente el funcionamiento de las diferentes agendas es principalmente mediante el boca a boca o mensaje de texto, por lo tanto, estarían necesitando una forma más automatizada y que les requiera menos tiempo para las agendas.
- Problemas de comunidad: Centro Tiféret no cuenta con una base de datos centralizada donde se aprecian todos los interesados en las propuestas del centro, por lo que muchas veces promocionar distintos eventos y/o clases se puede hacer un poco tedioso y complicado.

4. Lista de necesidades

| ID | Necesidad | Descripción |
|-----|--------------------------|---|
| N01 | Mayor Visibilidad Online | El centro necesita mejorar su presencia en Internet para llegar a más personas. Actualmente, tienen una web limitada que no atrae a nuevos clientes de forma eficiente. Buscan que la nueva web les permita captar un público más amplio, tanto local como internacional. |
| N02 | Tienda Online Eficiente | Quieren integrar una tienda online completa que permita a los usuarios comprar productos y servicios de manera sencilla. Esto incluye desde libros y podcasts, hasta cursos y talleres. Hoy en día, el centro no tiene una forma fluida de vender productos en línea. |
| N03 | Sistema de Membresías | Desarrollar un sistema de membresías que ofrezca contenido exclusivo a quienes se suscriban. Este sistema busca crear una comunidad de miembros que accedan a descuentos, eventos especiales y contenido adicional como podcasts o grabaciones. |

| | | |
|-----|---|---|
| N04 | Fácil Registro en Eventos y Reservas | Los usuarios deben poder registrarse y pagar sus eventos de forma rápida y directa desde la web. Actualmente, el proceso de inscripción es manual y tedioso, lo que puede generar pérdidas de usuarios y errores. |
| N05 | Automatización de la Gestión de Usuarios | Necesitan un sistema que permita a los usuarios crear perfiles, gestionar sus actividades, y acceder a sus compras o eventos. El control de acceso debe ser simple y automatizado para que los administradores puedan gestionar todo sin complicaciones. |
| N06 | Creación de Grupos Interactivos | Los grupos interactivos serán claves para fomentar la participación de los miembros. A través de ellos, los usuarios podrán compartir experiencias, acceder a recursos como documentos y participar en discusiones. Esto será fundamental para la comunidad de miembros pagos del centro. |
| N07 | Mejora de la Experiencia de Usuario (UX) | La página necesita ser más fácil de usar, tanto para los clientes como para los administradores del centro. La navegación debe ser fluida y accesible desde cualquier dispositivo, eliminando las barreras actuales que hacen que la web sea difícil de usar. |
| N08 | Facilitar el Contacto Directo | Implementar un sistema de contacto efectivo para que los usuarios puedan comunicarse fácilmente con el centro. Esto incluye soporte en tiempo real o un sistema de mensajería, algo que actualmente no existe y dificulta la resolución rápida de problemas o dudas. |
| N09 | Seguridad en las Transacciones | Aumentar la seguridad de los pagos y de los datos personales. Necesitan implementar medidas como autenticación multifactor (MFA) y encriptación para asegurar que la información sensible esté protegida, algo especialmente crítico en transacciones de pagos o registros de usuarios. |
| N10 | Herramienta para la Gestión de Contenidos | Los administradores deben poder actualizar el contenido de la web (eventos, productos, artículos) sin depender de terceros o conocimientos técnicos. Esto permitirá que la web esté siempre al día y que la oferta del centro pueda adaptarse rápidamente a las necesidades del momento. |
| N11 | Sección de Noticias y Blog | Crear un espacio donde se puedan publicar noticias y artículos relacionados con el bienestar personal y terapéutico. Esta sección ayudará a fidelizar a los usuarios con contenido relevante y actualizado que enriquezca su experiencia más allá de los servicios comprados. |
| N12 | Registro de Asistencia y Participación | Necesitan un sistema que lleve registro de la asistencia a eventos y actividades para poder analizar cómo están funcionando y ajustar la oferta futura. Actualmente, no |

| | | |
|-----|---|---|
| | | hay un método sistemático para medir cuántas personas asisten o cómo interactúan en las actividades del centro. |
| N13 | Personalización del Usuario | Los usuarios deben tener una experiencia adaptada a sus intereses y necesidades. Esto significa que el contenido mostrado debe ser relevante a su perfil, ya sea un visitante, un miembro registrado, o un miembro con suscripción paga. |
| N14 | Calendario para Reservas | Implementar Google calendar para que los usuarios puedan ver la disponibilidad de los eventos y reservar fácilmente. Esto evitará errores en la gestión de plazas y permitirá a los usuarios inscribirse sin problemas en sus actividades favoritas. |
| N15 | Notificaciones Personalizadas | El sistema debe enviar notificaciones automáticas a los usuarios sobre eventos, productos o contenidos relevantes. Esto ayudará a aumentar la participación en eventos y la compra de productos, al mantener a los usuarios siempre informados de las novedades que les interesen. |
| N16 | Gestión Automática de Pagos | Facilitar los pagos online, tanto para los eventos como para la compra de productos y suscripciones, de forma automática y sencilla. Actualmente, todo el proceso es manual, lo que genera demoras y problemas en la gestión. |
| N17 | Compatibilidad con Múltiples Dispositivos | La web debe ser completamente funcional desde cualquier dispositivo, ya sea un smartphone, tablet o computadora. La accesibilidad móvil es esencial para garantizar que los usuarios puedan interactuar con el centro desde cualquier lugar y en cualquier momento. |
| N18 | Análisis de Datos para Administradores | Los administradores necesitan herramientas que les permitan ver estadísticas sobre el uso del sitio web, comportamiento de los usuarios, ventas y participación en eventos. Esta información será clave para tomar decisiones basadas en datos y mejorar continuamente la oferta de servicios del centro. |

5. Objetivos

5.1 Implementar un sistema de e-commerce

Específico: Incorporar una plataforma que permita la compra de productos y servicios ofrecidos por el centro, como libros, cursos, podcasts y más.

Mensurable: El objetivo es alcanzar un mínimo de 50 ventas online en los primeros tres meses.

Alcanzable: Se desarrollará utilizando tecnologías integradas en la nueva plataforma web.

Relevante: El comercio electrónico amplía el acceso a los productos y servicios del centro, aumentando los ingresos.

Temporal: Este sistema estará completamente funcional desde el lanzamiento de la nueva página web.

5.2 Crear un sistema de membresías pagas

Específico: Desarrollar un sistema que permita a los usuarios suscribirse a membresías para acceder a contenido exclusivo y ofertas especiales.

Mensurable: Se pretende captar al menos 100 usuarios con membresías pagas en el primer semestre de operación.

Alcanzable: Será desarrollado como parte de la nueva web, permitiendo a los usuarios gestionar sus membresías y acceder a contenido exclusivo.

Relevante: Esto fomenta la fidelización y monetización de los servicios.

Temporal: El sistema estará disponible desde el lanzamiento de la nueva página web.

5.3 Establecer un sistema de reservas online

Específico: Implementar una herramienta que permita a los usuarios registrarse y reservar actividades y eventos directamente desde la web.

Mensurable: La meta es tener un 60% de las reservas realizadas a través del sistema online en los primeros seis meses.

Alcanzable: Esto será posible gracias a la integración de la plataforma con un sistema de gestión de usuarios.

Relevante: Facilitará la organización y planificación de actividades tanto para el centro como para los usuarios.

Temporal: El sistema estará operativo desde el lanzamiento de la nueva página web.

5.4 Desarrollar un panel administrativo para la gestión de contenido

Específico: Crear un panel de control que permita al personal del centro actualizar el contenido de la página web, como eventos, precios, noticias, y reservas.

Mensurable: El equipo del centro podrá gestionar el 100% del contenido sin necesidad de asistencia técnica externa en el primer mes de uso.

Alcanzable: El panel será diseñado para ser intuitivo y de fácil uso por el personal del centro.

Relevante: Mejorará la eficiencia operativa y la capacidad de respuesta del centro.

Temporal: El panel estará disponible desde el lanzamiento de la nueva página web.

5.5 Desarrollar la web como una Progressive Web App (PWA)

Específico: Crear la nueva página web como una PWA para ofrecer una experiencia optimizada y accesible desde cualquier dispositivo.

Mensurable: Se espera un aumento del 20% en la tasa de retención de usuarios debido a la mejora en la experiencia de navegación y acceso al contenido.

Alcanzable: Se utilizarán tecnologías web modernas que permitan la creación de una PWA, con funcionalidades offline.

Relevante: Esto asegura accesibilidad y mejora la interacción de los usuarios con la plataforma.

Temporal: La PWA estará lista en los primeros seis meses tras el inicio del proyecto.

6. Requerimientos funcionales

| ID | Requerimiento Funcional | Prioridad | Descripción Humanizada |
|--------|--|-----------|--|
| RF01 | Registro de usuarios | Alta | Los usuarios podrán registrarse en la plataforma proporcionando su información básica (nombre, email, contraseña). Este registro permitirá acceder a diferentes servicios y contenido según su rol en la plataforma. |
| RF01.1 | Validación de datos en el registro | Alta | Durante el proceso de registro, se validarán los datos para asegurar que el email tenga el formato correcto y que la contraseña cumpla con los requisitos de seguridad establecidos (longitud mínima, uso de caracteres especiales, etc.). |
| RF01.2 | Confirmación de registro mediante correo electrónico | Media | Una vez completado el registro, el sistema enviará un correo de confirmación al usuario. El usuario deberá hacer clic en un enlace de verificación para activar su cuenta y completar el proceso de registro. |

| | | | |
|---------------|---|-------|---|
| RF02 | Inicio de sesión y autenticación | Alta | Los usuarios podrán iniciar sesión en la plataforma con su correo electrónico y contraseña. Dependiendo de su rol (visitante, miembro, miembro pago o administrador), tendrán acceso a distintas funcionalidades. |
| RF02.1 | Recuperación de contraseña | Media | Si un usuario olvida su contraseña, podrá solicitar un enlace de recuperación que le será enviado a su correo electrónico, permitiéndole establecer una nueva contraseña. |
| RF03 | Asignación de roles y permisos | Alta | Se definirán varios roles de usuario: <ul style="list-style-type: none"> - Visitante: Puede ver productos y noticias. - Miembro: Acceso a eventos y perfil personal. - Miembro Pago: Acceso a contenido premium y beneficios adicionales. - Administrador: Control total del sitio. |
| RF03.1 | Gestión de permisos para cada rol | Alta | Cada rol tendrá permisos específicos para acceder a determinadas funcionalidades, como la tienda online, eventos, y contenido premium. Los administradores tendrán el control total del sistema, mientras que los usuarios pagarán para acceder a contenido exclusivo. |
| RF04 | Gestión de la tienda online | Alta | La plataforma incluirá una tienda donde los usuarios podrán comprar productos físicos y digitales (como libros, cursos o podcasts). Cada producto tendrá su descripción, imágenes y precio para facilitar la decisión de compra. |
| RF04.1 | Visualización del catálogo de productos | Alta | Los usuarios podrán explorar el catálogo de productos, ver las descripciones detalladas |

| | | | |
|---------------|--|------|---|
| | | | y precios, y seleccionar productos de su interés. |
| RF04.2 | AB carrito | Alta | Los usuarios podrán agregar o quitar productos a un carrito de compras virtual y proceder a la finalización de la compra cuando estén listos. |
| RF04.3 | Finalización de la compra y métodos de pago | Alta | Una vez que el usuario decida finalizar la compra, podrá elegir entre diferentes métodos de pago, como tarjeta de crédito o débito, o incluso utilizar su saldo en la billetera virtual, si ha decidido usar esta opción. |
| RF05 | Sistema de membresías pagadas | Alta | Se ofrecerán diferentes tipos de membresías pagas, permitiendo a los usuarios acceder a contenido exclusivo, eventos especiales y otros beneficios. Estas membresías podrán ser gestionadas y actualizadas por los administradores. |
| RF05.1 | Gestión de tipos de membresías | Alta | Los administradores podrán crear y gestionar distintos tipos de membresías, definiendo los beneficios y el acceso exclusivo que cada una otorga. |
| RF06 | Registro y reserva de eventos | Alta | Los usuarios podrán registrarse en eventos como talleres, conferencias o clases que se ofrezcan en la plataforma. Estos eventos estarán organizados en el calendario de Google donde podrán hacer sus reservas. |
| RF06.1 | Visualización de eventos en calendario de google | Alta | La plataforma se comunicará con el calendario de Google donde los usuarios podrán ver los próximos eventos. ABM de calendario controlado con la API |

| | | | |
|---------------|---|-------|--|
| RF07 | Grupos interactivos para miembros pagos | Media | Los usuarios con membresías pagas tendrán acceso a grupos interactivos donde podrán compartir experiencias, interactuar con otros miembros, y acceder a contenido exclusivo, como documentos y archivos relacionados con las actividades del centro. |
| RF07.1 | Chat grupal para miembros | Media | Los miembros pagos podrán participar en un chat grupal estilo foro dentro de los grupos interactivos, donde podrán intercambiar ideas y participar en discusiones sobre temas de interés en común. |
| RF07.2 | Compartir archivos dentro del grupo | Media | Los usuarios podrán compartir archivos, como documentos y videos, dentro de los grupos interactivos, permitiendo una mayor colaboración y participación en temas relacionados. |
| RF08 | Gestión del perfil personal | Alta | Cada usuario tendrá un perfil personal donde podrá ver su historial de compras, modificar sus datos, y gestionar sus reservas de eventos. Esta funcionalidad ayudará a los usuarios a tener control sobre su actividad en la plataforma. |
| RF08.1 | Ver historial de compras | Media | El usuario tendrá acceso a los detalles de todas sus compras en orden descendiente. |
| RF08.2 | Modificación de datos personales | Media | Los usuarios podrán actualizar su información personal (nombre, dirección, correo electrónico, etc.) en cualquier momento a través de su perfil en la plataforma. |

| | | | |
|---------------|---|-------|---|
| RF08.3 | Gestor de reservas | Alta | El usuario podrá visualizar todos los eventos a los cuales está inscrito y ver el detalle de los mismos. |
| RF9 | Panel de administración para gestión de contenido | Alta | Los administradores tendrán acceso a un panel que les permitirá gestionar toda la plataforma, desde la creación de eventos y productos, hasta la gestión de miembros, suscripciones y contenido general, como noticias y actualizaciones. |
| RF9.1 | ABM de productos y eventos | Alta | Los administradores podrán agregar, modificar o eliminar productos y eventos directamente desde el panel de administración, asegurando que siempre se ofrezcan los servicios más actualizados. |
| RF9.2 | Enviar productos de pago presencial | Alta | Los administradores podrán habilitar el uso de un artículo previamente agregado a la tienda online para un usuario que haya realizado un pago presencial en el centro. |
| RF9.3 | Envío de notificaciones | Media | Los administradores podrán enviar notificaciones y boletines informativos a todos los usuarios registrados, informándoles sobre eventos próximos, ofertas en la tienda, o cualquier actualización importante en la plataforma. |
| RF10 | Soporte y contacto | Media | Los usuarios podrán ponerse en contacto con el equipo del centro para resolver cualquier duda o problema que tengan con respecto al uso de la plataforma. Esta funcionalidad incluirá un sistema de mensajes directos o formulario de contacto. |

| | | | |
|---------------|-----------------------------------|-------|--|
| RF10.1 | Sistema de respuestas automáticas | Media | Se implementará un sistema de respuestas automáticas para consultas frecuentes (FAQ), lo que permitirá a los usuarios obtener respuestas rápidas a las preguntas más comunes sin necesidad de esperar la asistencia de un administrador. |
| RF11 | ABM Noticias y blog informativo | Media | Se publicarán artículos y noticias relevantes sobre los eventos, productos, y temas relacionados con el centro terapéutico, ofreciendo contenido útil y actualizado a los usuarios para mejorar su experiencia y conocimiento. |
| RF12 | Seguridad de datos y encriptación | Alta | La información personal de los usuarios y los datos relacionados con pagos estarán protegidos mediante la encriptación y uso de protocolos de seguridad avanzados (SSL), garantizando la seguridad y privacidad de toda la información. |
| RF12.1 | Autenticación multifactor (MFA) | Media | Los usuarios, especialmente los administradores, tendrán la opción de activar la autenticación multifactor para asegurar aún más sus cuentas, protegiéndolas de posibles accesos no autorizados. |

7. Requerimientos no funcionales:

| ID | Requerimiento No Funcional | Prioridad | Descripción Humanizada |
|-------------|--|-----------|---|
| RNF1 | Compatibilidad en diferentes plataformas | Alta | La página web debe ser compatible tanto en los navegadores más utilizados como Chrome, Safari, Firefox y Edge; como en los distintos dispositivos móviles y sus diferentes marcas: Apple, Samsung, Huawei, Xiaomi, Motorola, etc. |

| | | | |
|-------------|--|-------|---|
| RNF2 | Arquitectura escalable para futura expansión | Alta | La arquitectura del sistema debe permitir la expansión futura para agregar más servicios, usuarios y funcionalidades sin necesidad de rediseñar la infraestructura. |
| RNF3 | Soporte para usuarios sin impacto crítico | Media | El sistema debe poder soportar hasta 1,000 usuarios concurrentes sin afectar el rendimiento de las funciones críticas (e-commerce, reservas, membresías). |
| RNF4 | Respaldo automático | Media | Debe contar con un sistema de respaldo automático diario de la base de datos y el contenido para evitar pérdidas de información. |
| RNF5 | Logs de transacciones | Baja | Se deben generar logs detallados sobre transacciones de e-commerce y actividades de usuarios para estar al tanto de posibles fallas y resolución de problemas. |
| RNF6 | Interfaz intuitiva | Alta | La interfaz debe ser intuitiva y fácil de usar tanto para los usuarios finales como para los administradores del panel de control. |
| RNF7 | Diseño responsivo | Media | Debe implementarse un diseño responsivo que garantice una experiencia óptima en dispositivos móviles y tablets, además de los navegadores de escritorio. |

8. Tecnologías Seleccionadas y Justificación

8.1 Progressive Web App (PWA)

Una Progressive Web App (PWA) es una aplicación web que utiliza las capacidades modernas de los navegadores para ofrecer una experiencia similar a la de una aplicación nativa. Las PWAs pueden instalarse en dispositivos móviles o computadoras, funcionar

sin conexión (gracias al uso de Service Workers), y ofrecen una experiencia de usuario optimizada, como notificaciones push y tiempos de carga más rápidos.

8.1.1 Razones para utilizar una PWA:

Experiencia de usuario nativa: Las PWAs combinan lo mejor de las aplicaciones web y nativas, permitiendo a los usuarios tener una experiencia fluida tanto en navegadores como en dispositivos móviles sin tener que descargar una aplicación de una tienda de apps.

Funcionamiento offline: Gracias al uso de caché y Service Workers, los usuarios pueden interactuar con la aplicación incluso cuando no tienen conexión a Internet.

Velocidad y rendimiento: Al almacenar partes de la aplicación en caché, las PWAs cargan más rápido que las páginas web tradicionales.

Menor costo de desarrollo: Una PWA permite una experiencia unificada en diferentes dispositivos sin la necesidad de desarrollar aplicaciones separadas para iOS, Android y web.

8.2 React

React es una biblioteca de JavaScript desarrollada por Facebook, usada principalmente para construir interfaces de usuario (UI) interactivas. React se basa en el uso de componentes, lo que permite la creación de interfaces dinámicas, reutilizables y eficientes.

8.2.1 Razones para usar React:

Componentización: React permite dividir la UI en componentes pequeños, lo que facilita el mantenimiento, la reutilización de código y el desarrollo modular.

Rendimiento optimizado: React utiliza el Virtual DOM, que actualiza solo los elementos necesarios en lugar de renderizar toda la página de nuevo, lo que mejora el rendimiento.

Amplia comunidad y soporte: Al ser una de las bibliotecas más populares, React cuenta con una gran comunidad de desarrolladores, recursos, y bibliotecas que aceleran el desarrollo.

8.3 Node.js con Express

Node.js es un entorno de ejecución de JavaScript que permite ejecutar código JavaScript del lado del servidor. Express es un framework minimalista para Node.js, diseñado para construir aplicaciones web y APIs robustas.

8.3.1 Razones para usar Node.js con Express:

Unificación del lenguaje: Utilizar JavaScript tanto en el frontend (React) como en el backend (Node.js) facilita el desarrollo, ya que se emplea un único lenguaje en toda la aplicación.

Asincronía y rendimiento: Node.js está optimizado para manejar múltiples conexiones simultáneas de manera eficiente gracias a su modelo basado en eventos y asincronía.

Escalabilidad: Node.js con Express permite construir APIs rápidas y escalables, ideales para aplicaciones que requieren manejar un número significativo de usuarios y datos.

8.4 Tailwind CSS

Tailwind CSS es un framework de CSS que permite construir interfaces de usuario de forma rápida y eficiente mediante la aplicación de clases predefinidas directamente en el HTML. A diferencia de otros frameworks como Bootstrap, Tailwind es altamente personalizable y no impone estilos predeterminados.

8.4.1 Razones para usar Tailwind:

Control total sobre los estilos: Tailwind permite diseñar componentes con precisión sin necesidad de sobrescribir clases predefinidas.

Desarrollo rápido: Las utilidades de Tailwind permiten aplicar estilos directamente a los elementos HTML, lo que acelera significativamente el proceso de desarrollo.

Consistencia: Al usar una misma fuente de estilos, Tailwind garantiza una apariencia coherente en todo el proyecto.

8.5 Jira

Jira es una herramienta de gestión de proyectos desarrollada por Atlassian. Es ampliamente utilizada en equipos de desarrollo de software para la planificación, seguimiento y gestión de proyectos en metodologías ágiles, como Scrum y Kanban.

8.5.1 Razones para usar Jira:

Gestión de tareas eficiente: Jira permite dividir el proyecto en tareas pequeñas, asignarlas a miembros del equipo y hacer un seguimiento de su progreso.

Metodologías ágiles: Soporta frameworks como Scrum y Kanban, lo que facilita la planificación de sprints y la asignación de prioridades.

Integración: Jira se integra con muchas otras herramientas como GitHub, Slack y Bitbucket, lo que facilita la colaboración y el seguimiento del código.

Ventajas sobre otras herramientas de gestión: Aunque existen otras opciones como Trello o Asana, Jira está especialmente diseñado para proyectos de desarrollo de software, ofreciendo una gran variedad de herramientas para el seguimiento de issues, integración continua y colaboración entre equipos.

8.6 MySQL

MySQL es un sistema de gestión de bases de datos relacional (RDBMS) ampliamente utilizado y de código abierto. Es una de las bases de datos más populares y estables, especialmente en proyectos de desarrollo web. Su facilidad de uso, robustez y velocidad la convierten en una excelente opción para proyectos como el del Centro Terapéutico Tiféret, donde se manejarán datos de usuarios, productos y transacciones.

8.6.1 Razones para usar MySQL:

Fiabilidad y estabilidad: MySQL es conocido por su rendimiento confiable y capacidad para manejar grandes volúmenes de datos, lo que es esencial en aplicaciones que requieran una base de datos robusta.

Compatibilidad: MySQL se integra fácilmente con Node.js y otras tecnologías del stack de desarrollo, permitiendo la creación de aplicaciones eficientes y bien conectadas.

Escalabilidad: Es lo suficientemente flexible para gestionar desde pequeñas bases de datos hasta grandes volúmenes de información, lo que permite su escalabilidad a medida que crece el centro y sus usuarios.

Comunidad y soporte: MySQL cuenta con una amplia comunidad de desarrolladores, lo que facilita el acceso a recursos, tutoriales y soluciones a posibles problemas.

Licencia Open Source: Al ser de código abierto, MySQL es una opción económica que no requiere costosas licencias, a diferencia de otros sistemas propietarios.

Ventajas sobre otras bases de datos:

PostgreSQL: Aunque PostgreSQL es una base de datos avanzada con soporte para funciones complejas, MySQL es más fácil de configurar y tiene un mejor rendimiento en aplicaciones web que no requieren características avanzadas como procedimientos almacenados o tipos de datos complejos.

MongoDB: MongoDB es una base de datos NoSQL adecuada para manejar datos no estructurados. Sin embargo, para un proyecto que requiere relaciones definidas entre datos (como usuarios y sus actividades, productos y reservas), MySQL, al ser relacional, es una opción más apropiada.

SQLite: SQLite es ligera y fácil de implementar, pero está diseñada para proyectos más pequeños y no tiene la capacidad de escalar como MySQL, lo que es importante para la futura expansión de Tiferet.

9. Tecnologías alternativas y justificación de la selección

9.1.1 Angular o Vue.js (en lugar de React):

Angular es un framework completo que ofrece una estructura más rígida que React, lo que puede ser útil para equipos grandes. Sin embargo, React es más flexible y tiene una curva de aprendizaje más baja, además de una comunidad más extensa.

Vue.js es una opción más ligera y también muy utilizada. No obstante, React tiene una comunidad más grande y mejores integraciones con bibliotecas y herramientas específicas para PWAs, lo que lo hace preferible para este proyecto.

9.1.2 PHP con Laravel (en lugar de Node.js con Express):

Laravel es un framework de PHP muy popular para la creación de aplicaciones web. Aunque es potente y ofrece muchas características, Node.js con Express permite trabajar con JavaScript tanto en el backend como en el frontend, lo que unifica el desarrollo y simplifica la integración entre ambas partes.

9.1.3 Bootstrap (en lugar de Tailwind CSS):

Bootstrap es uno de los frameworks CSS más populares. Sin embargo, ofrece componentes predefinidos que pueden hacer que las interfaces parezcan genéricas. Tailwind, en cambio, permite un diseño completamente personalizado, lo que es ideal para una aplicación con necesidades específicas como la del Centro Tiferet.

9.1.4 Trello o Asana (en lugar de Jira):

Trello es más simple y visual que Jira, pero carece de la profundidad en la gestión de tareas y sprints que se necesita para un proyecto de desarrollo de software estructurado. Asana es otra opción, pero Jira ofrece más integraciones y herramientas especializadas para proyectos ágiles.

9.1.5 Comparación entre CMS y Tecnologías Personalizadas

Ventajas de los CMS como WordPress, Wix, o Squarespace:

Fácil de usar: Permiten a los usuarios sin experiencia técnica crear sitios web de forma rápida a través de plantillas y editores drag-and-drop.

Implementación rápida: Al contar con plantillas prediseñadas, los CMS permiten poner en marcha un sitio web en poco tiempo.

Plugins y extensiones: Ofrecen un gran número de plugins que agregan funcionalidad adicional sin necesidad de escribir código.

Mantenimiento simplificado: El hosting, las actualizaciones de seguridad y las copias de seguridad están generalmente automatizados.

9.1.6 Limitaciones de los CMS:

Flexibilidad limitada: Aunque ofrecen muchas funcionalidades predefinidas, la personalización avanzada suele ser limitada o compleja. Las plantillas y plugins tienen restricciones y, a veces, no permiten crear experiencias completamente a medida.

Rendimiento: Los CMS tienden a ser más lentos debido a su estructura, ya que cargan muchas funciones genéricas que no siempre son necesarias.

Escalabilidad: A medida que un sitio web crece en tamaño y complejidad, los CMS pueden tener problemas de rendimiento y escalabilidad.

Dependencia de terceros: Los usuarios dependen de plugins de terceros para agregar funcionalidades adicionales, lo que puede generar problemas de compatibilidad y seguridad.

9.1.7 Ventajas de las Tecnologías Personalizadas (React, Node.js, Tailwind, etc.)

Flexibilidad Total: El uso de tecnologías como React y Node.js permite construir aplicaciones web y Progressive Web Apps (PWA) totalmente personalizadas. No estás limitado por plantillas ni plugins, lo que significa que puedes adaptar la funcionalidad y el diseño a las necesidades específicas del proyecto.

Rendimiento Optimizado: Al construir una PWA desde cero, es posible optimizar el rendimiento al máximo, utilizando solo el código necesario y aplicando técnicas modernas como el Virtual DOM en React. Esto permite que el sitio cargue más rápido y funcione de manera más eficiente, especialmente en dispositivos móviles y en entornos con mala conectividad.

Escalabilidad: Las tecnologías personalizadas permiten construir soluciones escalables que pueden crecer en funcionalidad y complejidad a medida que lo haga el negocio. No existen limitaciones de infraestructura o rendimiento como en los CMS tradicionales, ya que puedes adaptar el backend y el frontend según las demandas.

Funcionalidad Avanzada: Al usar React y Node.js, se pueden crear aplicaciones web complejas que incluyan características avanzadas, como interacciones en tiempo real, notificaciones push, caché offline, e-commerce personalizado, sistemas de membresías, y mucho más. Estas funcionalidades son difíciles o costosas de implementar en un CMS.

Control Total sobre el Diseño: Con frameworks como Tailwind CSS, el diseño no se ve restringido por plantillas. Puedes construir interfaces visuales completamente a medida, garantizando una experiencia de usuario única y acorde a los valores y estilo de la organización.

Mejor Experiencia de Usuario (UX): Las PWAs ofrecen una experiencia más parecida a una aplicación nativa, lo que es ideal para usuarios que acceden desde dispositivos móviles. Además, la funcionalidad offline y las notificaciones push mejoran la

interacción continua con el usuario, algo que un sitio basado en un CMS tradicional no puede ofrecer de manera fluida.

9.1.8 Conclusión

Elegir tecnologías personalizadas como React, Node.js, y Tailwind CSS, en lugar de un CMS como WordPress, Wix o Squarespace, ofrece una clara ventaja en términos de flexibilidad, escalabilidad, rendimiento y personalización. Si bien los CMS pueden ser una solución rápida y económica para sitios web simples, proyectos que requieren funcionalidades avanzadas, un diseño único y una experiencia de usuario optimizada se benefician enormemente de una PWA hecha a medida. Estas tecnologías permiten crear soluciones que están completamente alineadas con las necesidades del proyecto, sin depender de las limitaciones de una plataforma cerrada o plugins de terceros.

La combinación de React, Node.js con Express, Tailwind CSS, y Jira permitirá crear una Progressive Web App robusta, escalable y personalizable para el Centro Terapéutico Tiféret. Esta elección garantiza flexibilidad, velocidad, una experiencia de usuario moderna y una capacidad de gestión eficiente del proyecto, superando las limitaciones de plataformas como WordPress o soluciones cerradas como Shopify.

Al seleccionar las tecnologías y herramientas para este proyecto, también se tuvo en cuenta el nivel de experiencia de nuestro equipo y el desafío que buscamos enfrentar. Contamos con una base de conocimiento en React gracias a lo aprendido en la carrera, y uno de nuestros objetivos es profundizar y afianzar nuestra competencia en esta tecnología tan popular. Respecto a las demás tecnologías y herramientas, el equipo está en busca de retos que no solo amplíen nuestros conocimientos, sino que también potencien nuestro aprendizaje, permitiéndonos avanzar en habilidades clave para el desarrollo web moderno.

10. Metodología de Trabajo: Scrum

Para el desarrollo de nuestro proyecto hemos optado por utilizar Scrum, una metodología ágil que nos permite trabajar de forma organizada, flexible y eficiente. Nuestro equipo está compuesto por tres integrantes, sin contar al Product Owner (PO), con quien mantenemos contacto continuo y nos reunimos de manera bisemanal para revisar avances y ajustar prioridades. Los ciclos de trabajo se estructuran en sprints de una semana, lo que nos permite realizar entregas frecuentes y asegurar una evolución constante del proyecto.

10.1 Roles del equipo

En este proyecto, los roles están claramente definidos para asegurar una correcta gestión y ejecución:

Diego Bentancor como Product Owner (PO), responsable de representar los intereses del centro terapéutico Tiféret.

Juan Núñez como Scrum Master, encargado de guiar al equipo en la metodología Scrum y desarrollo de la aplicación.

Facundo Orihuela y Diego Mazas como desarrolladores, responsables del desarrollo técnico y funcional de la aplicación.

10.2 ¿Por qué elegimos Scrum?

Experiencia previa: Como equipo, ya hemos trabajado con Scrum en otros proyectos, lo que nos ha permitido familiarizarnos con sus dinámicas. Esta experiencia nos facilita la organización interna y la ejecución de tareas dentro de cada sprint, ya que conocemos bien cómo estructurar el trabajo y gestionar las expectativas del cliente.

Facilidad de organización: Scrum establece un marco claro para la planificación y asignación de tareas. Al dividir el trabajo en sprints cortos, podemos enfocarnos en objetivos específicos y gestionar mejor las prioridades. Esto también nos ayuda a tener un seguimiento continuo del avance del proyecto, asegurando que no se acumulen retrasos o tareas pendientes.

Flexibilidad: Una de las principales razones por las que Scrum es ideal para este proyecto es la flexibilidad que ofrece, en este proyecto el cliente presenta incertidumbre sobre lo que será el producto final. A través de las reuniones bisemanales con el Product Owner, podemos adaptarnos a cambios en los requisitos o prioridades. Scrum permite una respuesta ágil ante cualquier cambio que pueda surgir durante el desarrollo, lo que es fundamental en proyectos donde los requisitos pueden evolucionar.

Involucración activa del cliente: Scrum destaca por fomentar la participación activa del cliente, en este caso el Product Owner quien trasmite el entusiasmo del cliente en este proyecto participando en cada etapa del mismo con gran presencia. Esto nos permite mantener un flujo de comunicación constante y directo, ajustando el proyecto en función de las necesidades y expectativas del cliente. El involucramiento del cliente en cada etapa del proceso garantiza que el producto final esté alineado completamente con su visión, lo que minimiza la posibilidad de malentendidos o entregas fuera de sus expectativas.

10.3 Otras ventajas de Scrum

Colaboración constante: Scrum fomenta la colaboración entre el equipo y el Product Owner. Las reuniones periódicas nos permiten recibir feedback inmediato y ajustar rápidamente el rumbo del proyecto, lo que mejora la calidad del resultado final. Este enfoque asegura que el producto esté alineado con las expectativas del cliente en todo momento.

Mejora continua: Al finalizar cada sprint, el equipo realiza una retrospectiva en la que se analizan los aspectos positivos y los desafíos encontrados. Esta práctica fomenta la mejora continua, ya que se pueden identificar áreas para optimizar tanto en la parte técnica como en la organización del equipo.

Transparencia y visibilidad: Scrum proporciona una gran visibilidad del estado del proyecto en todo momento. Las herramientas de gestión como Jira, que usamos para planificar y monitorear el trabajo, permiten que tanto el equipo como el Product Owner

tengan una visión clara de las tareas en curso, pendientes y completadas. Esto ayuda a mantener una comunicación abierta y fluida.

Enfoque en entregas incrementales: La naturaleza incremental de Scrum permite entregar valor de manera continua a través de pequeños incrementos del producto. Esto nos permite validar constantemente el trabajo con el Product Owner, asegurando que estamos en el camino correcto antes de avanzar demasiado.

Gestión de riesgos: Al realizar entregas regulares y mantener una comunicación constante con el Product Owner, los riesgos de desviación en cuanto a tiempos, recursos o resultados se minimizan. Cualquier problema o dificultad puede identificarse y abordarse de manera temprana, lo que reduce la posibilidad de sorpresas en fases avanzadas del proyecto.

10.4 Conclusión

La elección de Scrum como metodología de trabajo para este proyecto no solo se debe a nuestra experiencia previa con esta forma de organización, sino también a las claras ventajas que ofrece en términos de flexibilidad, colaboración y mejora continua. La capacidad de adaptarse a cambios, mantener un contacto constante con el Product Owner y gestionar el proyecto de manera eficiente hacen de Scrum la opción ideal para asegurar el éxito de nuestro proyecto.

Para llevar a cabo la implementación de esta metodología, utilizaremos Jira como herramienta de gestión, la cual nos permitirá organizar y planificar los sprints, asignar tareas, y hacer un seguimiento detallado del progreso, facilitando la visibilidad y control en todo momento.

11. Plan de SCM

Para asegurar una correcta gestión de los cambios y del versionado de código en el desarrollo del proyecto, utilizaremos GitHub como la plataforma principal para la gestión de versiones, con un enfoque en la colaboración eficiente y la protección del código en producción. A continuación, se describen los procedimientos y herramientas que utilizaremos para la gestión de cambios, versionado y control de código.

11.1 Gestión de Cambios

La gestión de cambios será llevada a cabo mediante un proceso estructurado basado en la división del repositorio en tres ramas principales:

Develop: Esta será la rama de desarrollo activo donde se integrarán las nuevas características. Cada desarrollador trabajará en ramas individuales basadas en develop para luego fusionarlas una vez revisadas y aprobadas.

Testing: Una vez que las nuevas funcionalidades o cambios sean completados en la rama develop, se fusionarán en la rama testing para llevar a cabo pruebas exhaustivas y asegurar que no se generen errores o conflictos.

Main: Es la rama principal, que contendrá únicamente el código listo para producción. Solo se fusionarán cambios en esta rama después de pasar todas las pruebas en testing, garantizando la estabilidad y calidad del código que se despliega en producción.

Este enfoque permitirá tener un flujo de trabajo organizado, donde los cambios siempre se desarrollan y prueban antes de ser lanzados oficialmente, minimizando los riesgos de introducir errores en la versión final de la aplicación.

11.2 Versionado de Código

El control de versiones seguirá el modelo GitFlow, en el que las nuevas funcionalidades se desarrollan en ramas específicas creadas desde develop y luego se integran siguiendo el proceso descrito anteriormente. Cada commit deberá estar acompañado de un mensaje claro y descriptivo que explique el cambio realizado, lo cual facilitará el seguimiento y la revisión del historial de versiones.

Para asegurar la trazabilidad y el control, se aplicará versionado semántico, que define las versiones del código en función de actualizaciones importantes (mayores), nuevas funcionalidades (menores) o correcciones (parches). De esta forma, cualquier cambio en la aplicación quedará claramente documentado y controlado.

11.3 Herramientas a Utilizar

El equipo utilizará GitHub Desktop como la herramienta principal para interactuar con el repositorio de GitHub. GitHub Desktop facilita la sincronización del código, permitiendo clonar, enviar y recibir cambios de manera sencilla, y revisar los commits de manera gráfica, lo que simplifica la gestión del código y su histórico. Esta interfaz más intuitiva reduce la curva de aprendizaje y ayuda a los desarrolladores a centrarse en el código sin complicarse con comandos de terminal.

Además, GitHub Desktop será fundamental para la revisión de cambios antes de integrarlos a las ramas de desarrollo, ya que permite visualizar las diferencias entre versiones y resolver conflictos de manera más amigable. Esto asegura que los cambios sean revisados y aprobados antes de ser fusionados en las ramas correspondientes, manteniendo la calidad y consistencia del proyecto.

12. Plan de Aseguramiento de la Calidad del Software (SQA)

El plan de aseguramiento de calidad del software (SQA) para el proyecto se centrará en garantizar que el código desarrollado cumpla con los estándares de desarrollo definidos y que el producto final cumpla con los requisitos funcionales a través de un proceso exhaustivo de testing. A continuación, se detallan los estándares a utilizar, el plan de testing y las herramientas seleccionadas.

Para la documentación nos apoyaremos en los estándares brindados por la ORT, el documento 302.

12.1 Estándares y Buenas Prácticas de Desarrollo

Para asegurar la calidad del código y su mantenibilidad a largo plazo, seguiremos los siguientes estándares de desarrollo:

Nombres Mnemotécnicos y en inglés: Todos los nombres de variables, funciones y clases deben ser claros, descriptivos y estar en inglés. Esto facilitará la lectura y comprensión del código tanto para los miembros actuales del equipo como para futuros colaboradores. Ejemplo: `calculateTotalPrice()` es preferible a `calcular_precio_total()`.

Segregación de Responsabilidades (SRP): Cada función o módulo debe tener una única responsabilidad claramente definida. Esto facilita la comprensión, el testing y el mantenimiento del código.

No Dejar Comentarios Innecesarios: El código debe ser auto explicativo. Los comentarios innecesarios o excesivos que "ensucien" el código no serán permitidos. Sin embargo, se permitirán comentarios breves en inglés para explicar decisiones complejas o poco evidentes.

Consistencia en el Formato de Código: Se adoptarán convenciones claras de formato, como usar indentación consistente, limitación de la longitud de las líneas, y el uso de espacios en lugar de tabs. Para esto, se podrán utilizar herramientas de formateo automático como Prettier o ESLint.

Principio DRY (Don't Repeat Yourself): Evitaremos duplicar código. Si hay fragmentos de código que realizan tareas similares, serán encapsulados en funciones reutilizables o clases. Esto no solo reduce errores, sino que facilita las actualizaciones en el futuro.

Manejo Eficiente de Errores: Implementaremos un sistema adecuado de captura y manejo de errores que no solo intercepte fallos, sino que también brinde retroalimentación útil para los desarrolladores y los usuarios. Evitaremos exponer detalles sensibles de errores al usuario final.

Código Modular y Desacoplado: Dividiremos el código en módulos o componentes independientes que puedan ser mantenidos, probados y reutilizados por separado. Esto hace que el código sea más escalable y facilita la localización de errores.

12.2 Plan de Testing

El proceso de testing será clave para asegurar que la aplicación funcione correctamente antes de su despliegue en producción. Se centrará en pruebas exhaustivas de los endpoints del backend y se extenderá a pruebas funcionales en el frontend.

Testing de Endpoints (Backend)

Antes de que cualquier endpoint sea integrado en el frontend, se realizará una prueba exhaustiva para asegurarse de que cumpla con los requisitos funcionales y no presente errores. Este proceso incluye:

Validación de Respuestas: Se verificarán las respuestas del servidor para asegurarse de que devuelvan los datos esperados o los mensajes de error correctos.

Pruebas de Carga y Rendimiento: Se evaluará el comportamiento del servidor bajo condiciones de carga para garantizar que los endpoints mantengan un rendimiento óptimo incluso con múltiples solicitudes concurrentes.

Pruebas de Seguridad: Se probarán los endpoints para asegurar que cumplan con las políticas de seguridad, validando el acceso a los datos solo a usuarios autorizados.

Casos de Prueba: Se utilizarán plantillas de casos de prueba para documentar los inputs, outputs y resultados esperados de cada endpoint. Cada prueba será registrada junto con su evidencia en capturas de pantalla y logs de solicitudes/respuestas.

Pruebas Funcionales (Frontend)

Una vez los endpoints estén validados, se integrarán en el frontend, donde se realizarán pruebas funcionales para asegurar que los flujos de usuario funcionen correctamente.

Validación de Funcionalidades Principales: Se probarán las funcionalidades clave como el sistema de reservas, compras, registro de usuarios, entre otros, para asegurarse de que no presenten errores.

Pruebas de Compatibilidad: La PWA se probará en diferentes navegadores y dispositivos para asegurar que sea responsiva y compatible en diferentes entornos.

12.3 Herramientas a Utilizar

Postman: Utilizaremos Postman como la herramienta principal para probar los endpoints del backend. Postman permite realizar solicitudes HTTP fácilmente, facilitando la validación de respuestas y el seguimiento de pruebas. Además, permite crear colecciones de pruebas automáticas y exportar resultados, lo que nos permitirá automatizar parte del proceso de testing.

Prettier: Para asegurar que el código mantenga un formato limpio y consistente, se utilizarán estas herramientas de linters y formateadores automáticos. De este modo, el equipo se asegura de cumplir los estándares sin la necesidad de revisión manual constante.

Material UI: Material UI es una librería de componentes que incorporaremos al proyecto para agilizar el diseño de la página.

GitHub y GitHub Desktop: Para gestionar la revisión de código, los pull requests y la documentación de casos de prueba. Las pruebas pasarán por revisiones exhaustivas antes de ser aprobadas y fusionadas en la rama principal.

13. Plan de Capacitación

El equipo ha diseñado un plan de capacitación para familiarizarse a fondo con las tecnologías que se utilizarán en el desarrollo del proyecto. Estas tecnologías incluyen React, Tailwind, Node.js, Progressive Web App (PWA) y MySQL. Se ha asignado un total de 3 horas semanales dedicadas a la formación, con el objetivo de asegurar un manejo adecuado y fluido de cada una de estas herramientas a lo largo del desarrollo.

13.1 Capacitación en Tecnologías:

React: Dado que el equipo ya cuenta con experiencia básica adquirida en cursos previos, se planificarán sesiones avanzadas que se enfoquen en el desarrollo de componentes complejos, hooks avanzados, y gestión del estado a gran escala. Además, se dedicará tiempo a estudiar buenas prácticas y patrones de diseño comunes en React para garantizar un desarrollo limpio y escalable.

Tailwind: Se proporcionarán recursos y guías sobre cómo integrar y utilizar Tailwind CSS en proyectos React, con énfasis en su funcionalidad para crear interfaces responsivas y escalables. Las sesiones de capacitación incluirán la personalización de estilos, la creación de temas reutilizables y cómo optimizar el uso de Tailwind para evitar duplicación de código CSS.

Node.js: Se estructurará la formación en dos fases: una inicial donde se cubrirán conceptos básicos de Node.js y una segunda enfocada en la creación de APIs RESTful utilizando Express. La capacitación también incluirá técnicas de optimización de rendimiento en servidores y el uso de bases de datos con SQL, además de la integración con React en el frontend.

Progressive Web Apps (PWA): Se brindarán recursos y se realizarán prácticas sobre cómo convertir la aplicación en una PWA, centrándose en los aspectos de caching, service workers, manifest files, y cómo lograr que la aplicación funcione de forma offline. Se estudiarán casos de éxito en la implementación de PWA para comprender mejor los beneficios y su aplicación en el contexto de este proyecto.

MySQL: El equipo tiene experiencia previa trabajando con SQL Server Express, por lo que la transición a MySQL será fluida debido a las similitudes en las consultas SQL y la estructura general de bases de datos relacionales. La capacitación en MySQL se centrará en la creación y gestión de bases de datos, optimización de consultas y la integración con Node.js en el backend. Dado que MySQL es una de las bases de datos más utilizadas en entornos web, se estudiarán las mejores prácticas de configuración y seguridad, así como técnicas de optimización para garantizar el rendimiento eficiente de la aplicación.

13.2 Capacitación en Jira:

La capacitación en Jira no será necesaria, ya que Juan Núñez ha realizado un curso previo sobre esta herramienta. Él ha sido el responsable de crear el entorno de trabajo en Jira y ha capacitado al equipo en su uso básico, cubriendo la creación y gestión de tareas, la planificación de sprints, y el seguimiento del progreso del proyecto.

13.3 Esfuerzo estimado:

El equipo dedicará 3 horas semanales a la capacitación en las tecnologías mencionadas, lo que permitirá a todos los integrantes profundizar en el uso de estas herramientas sin comprometer el desarrollo del proyecto. Las sesiones se dividirán en pequeños módulos de acuerdo con las necesidades del sprint actual y los objetivos de cada etapa del proyecto.

13.4 Capacitación al Cliente:

Una vez que el producto esté finalizado, se brindará una capacitación al cliente, enfocada en el uso del panel administrativo del sistema. Este entrenamiento incluirá la gestión de contenidos, productos, reservas, y usuarios. Se organizará una sesión presencial o virtual para demostrar las funcionalidades principales, acompañada de manuales detallados y grabaciones de video para futuras consultas.

14. Gestión de riesgos

14.1 Identificación de Riesgos

- Riesgo 1: Retraso en la integración de la tienda online y el sistema de membresías. Esto podría generar demoras en la entrega final del producto, afectando tanto nuestra reputación como la experiencia de los usuarios.
- Riesgo 2: Cambios en los requisitos del cliente durante el desarrollo. A veces, surgen nuevas necesidades o cambios en las condiciones del mercado que pueden complicar nuestro proceso y requerir ajustes inesperados.
- Riesgo 3: Problemas de hardware o software en nuestros equipos. Fallos críticos en nuestro entorno de desarrollo o incompatibilidades con herramientas necesarias pueden frenar nuestro progreso y causar inconvenientes.
- Riesgo 4: Alto nivel de complejidad técnica en la integración de sistemas. Integrar APIs externas, como las de MercadoPago para gestionar pagos o las de plataformas de autenticación como Google y Facebook, puede ser un desafío y requerir más tiempo del esperado.
- Riesgo 5: Estimación inadecuada del tiempo de ejecución del proyecto. La falta de experiencia con ciertas tecnologías (como PWA o Tailwind CSS) o con funcionalidades adicionales (como el modo offline) puede llevarnos a subestimar el tiempo necesario para completar el proyecto.
- Riesgo 6: Vulnerabilidades de seguridad en el manejo de datos sensibles. Es fundamental proteger la información de los usuarios y asegurar las transacciones; cualquier brecha podría resultar en fugas de información o ataques malintencionados.

- **Riesgo 7:** Problemas de compatibilidad con navegadores y dispositivos. Si nuestra plataforma no funciona correctamente en diferentes entornos, podríamos comprometer la accesibilidad y la experiencia de los usuarios, lo que es inaceptable.
- **Riesgo 8:** Falta de capacitación del equipo administrativo. Si el equipo que gestionará la plataforma no está bien preparado para utilizar el panel de control, podrían surgir problemas operativos al administrar eventos, productos o usuarios.
- **Riesgo 9:** Sobrecarga en nuestro equipo de desarrollo. La presión para cumplir plazos estrictos y la falta de recursos adicionales pueden afectar la calidad del producto y llevar a errores que podrían haberse evitado.

A continuación, se ofrece un análisis de la probabilidad e impacto de cada riesgo, lo que facilitará la priorización de nuestras acciones correctivas:

| ID | Riesgo | Probabilidad (1-5) | Impacto (1-5) | Severidad (P x I) |
|----|---|-----------------------|------------------|-----------------------|
| R1 | Retraso en la integración de la tienda online y sistema de membresías | 4 (Probable) | 5 (Alto) | 20 (Crítico) |
| R2 | Cambios en los requisitos del cliente | 3 (Posible) | 5 (Alto) | 15 (Importante) |
| R3 | Problemas de hardware o software | 3 (Posible) | 4 (Moderado) | 12 (Moderado) |
| R4 | Complejidad técnica por integración de APIs | 3 (Posible) | 5 (Alto) | 15 (Importante) |
| R5 | Estimación inadecuada del tiempo | 3 (Posible) | 4 (Moderado) | 12 (Moderado) |
| R6 | Vulnerabilidades de seguridad en datos | 2 (Bajo) | 5 (Alto) | 10 (Significativo) |
| R7 | Incompatibilidad con navegadores y dispositivos móviles | 3 (Posible) | 3 (Moderado) | 9 (Bajo) |
| R8 | Falta de capacitación del equipo administrativo | 2 (Bajo) | 3 (Moderado) | 6 (Bajo) |
| R9 | Sobrecarga del equipo de desarrollo | 4 (Probable) | 3 (Moderado) | 12 (Moderado) |

14.2 Plan de Mitigación y Contingencia

Para cada uno de los riesgos que hemos identificado, hemos desarrollado acciones para prevenirlos y planes de contingencia en caso de que se materialicen.

| ID | Riesgo | Mitigación | Contingencia |
|----|--|---|------------------------------------|
| R1 | Retraso en la integración de la | Haremos revisiones constantes y pruebas | Si encontramos problemas, podremos |

| | | | |
|----|--|--|--|
| | tienda online y sistema de membresías | anticipadas para asegurarnos de que todo funcione correctamente. También nos informamos previamente sobre librerías que pueden agilizar el proceso | dedicar más horas a la integración y utilizar las semanas que reservamos como holgura. |
| R2 | Cambios en los requisitos del cliente | Tendremos reuniones bisemanales con el Product Owner para hacer ajustes en los requisitos de manera oportuna. | Ajustaremos el backlog y los sprints según los nuevos requisitos para mantener todo en orden. |
| R3 | Problemas de hardware o software | Implementaremos copias de seguridad y redundancias para proteger nuestro trabajo. | Usaremos los recursos necesarios para arreglar el problema, como contactar personas capacitadas, etc. |
| R4 | Complejidad técnica por integración de APIs | Vamos a capacitarnos antes de comenzar con las tareas más complejas e integraremos las APIs de manera gradual. | Si la complejidad supera nuestras capacidades, doblaremos el tiempo para capacitarnos en estas tecnologías. |
| R5 | Estimación inadecuada del tiempo | Haremos revisiones frecuentes del cronograma y asignaremos márgenes de tiempo para tareas críticas. Además, tendremos dos sprints dedicados a testing que representan la holgura del proyecto para casos de incertidumbre. | Si es necesario, redistribuiremos las tareas en diferentes sprints con menor carga y dedicaremos más tiempo a realizar las tareas. |
| R6 | Vulnerabilidades de seguridad en datos | Implementaremos auditorías de seguridad y usaremos protocolos para proteger la información. | Si identificamos problemas, aplicaremos parches de seguridad rápidamente. |
| R7 | Incompatibilidad con navegadores y dispositivos móviles | Realizaremos pruebas exhaustivas en diferentes dispositivos y navegadores desde el inicio del desarrollo. | Si surgen problemas, daremos prioridad a las soluciones en los navegadores y dispositivos más utilizados. |
| R8 | Falta de capacitación del equipo administrativo | Planificaremos sesiones de capacitación en grupo practicando y demostrando cómo utilizar la página. | Si persisten dudas o problemas operativos, organizaremos más sesiones para asegurarnos de el entendimiento de esta. |

| | | | |
|----|--|---|---|
| R9 | Sobrecarga en el equipo de desarrollo | Asignaremos tareas según su prioridad y revisaremos los avances utilizando la metodología ágil Scrum. | Si el equipo no puede cumplir con las tareas críticas, dividiremos el trabajo para hacerlo más manejable. |
|----|--|---|---|

15. Supuestos

- **Involucramiento del cliente:** Se prevé una participación constante y comprometida del Centro Terapéutico Tiféret a lo largo de todo el proyecto. Esto abarca su colaboración en el establecimiento de los requerimientos, la revisión y validación de los resultados, así como su disposición para ofrecer opiniones y tomar decisiones fundamentales.
- **Acceso a recursos:** Suponemos que contaremos con acceso a toda la información necesaria por parte del equipo de Tiféret, incluyendo contenido existente, productos, y detalles sobre los servicios para integrarlos en el nuevo sitio web de manera eficiente.
- **Cumplimiento de normativas:** Se asume que la aplicación desarrollada cumplirá con las normativas legales aplicables, en especial en lo que respecta a la protección de datos personales y la seguridad de las transacciones que se realicen a través de la plataforma.
- **Accesibilidad del cliente al panel administrativo:** Se espera que el equipo de Tiféret gestione el contenido del sitio de manera autónoma tras la implementación, por lo que el panel de administración será intuitivo y de fácil acceso para el personal no técnico.

16. Arquitectura

El proyecto consta de un sistema web con implementación de Progressive Web App (PWA) desarrolladas en React, que se comunicará con una API RESTFUL implementada en Node.js con Express y conectada a una base de datos MySQL en una nube.

17. Alcance del producto

En este análisis preliminar, el alcance propuesto para el proyecto se percibe como factible y asequible, considerando el nivel de complejidad de los requisitos. Los desafíos adicionales podrían surgir por el empleo de herramientas y funcionalidades que no se profundizaron durante la carrera, lo que demanda un proceso de investigación y adquisición de conocimientos sobre ellas.

Obligatorios:

- Registro de usuarios
- Inicio de sesión
- Asignación de roles
- Tienda online
- Sistema de membresías pagas
- Registro y reserva de eventos
- Panel de administración para gestión de contenido
- Seguridad de datos y encriptación

Deseables:

- Visualización de eventos en calendario
- Grupos interactivos para miembros
- Chat grupal para miembros
- Compartir archivos dentro del grupo
- Gestión del perfil personal

Opcionales:

- Envío de notificaciones
- Sistema de respuestas automáticas
- Noticias y blog informativo
- Mapa interactivo para ubicación de profesionales

18. Cronograma

El equipo se maneja con sprints de una semana a excepción del primero (Sprint 0) y el ultimo.

Se considera sprint 0 a todo el tiempo previo a la aprobación y clarificación del proyecto tanto con el cliente como con la universidad ORT. La duración de este sprint fue desde el veintidós de octubre al treinta de noviembre, el equipo dedico este tiempo a hablar con Diego Bentancor, el Product Owner de nuestro proyecto y cara visible del Centro Tiféret y también a capacitarnos en las tecnologías que creímos relevantes para este proyecto.

18.1 Inicio de anteproyecto

18.1.1 Sprint 1

| Inicio: 30/09/24 | Fin: 06/10/24 |
|---|---------------------|
| Tarea | Estimación en horas |
| Creación de jira | 1 |
| Introducción al proyecto | 2 |
| Investigación de PWA | 5 |
| Abstract | 1 |
| Palabras clave | 1 |
| Descripción del cliente | 2 |
| Descripción del problema | 2 |
| Requerimientos funcionales | 10 |
| Requerimientos no funcionales | 3 |
| Estudio de alternativas y selección de herramientas | 8 |
| Total | 35 |

Gantt



18.1.2 Sprint 2

| Inicio: 07/10/24 | Fin: 13/10/24 |
|--|---------------------|
| Tarea | Estimación en horas |
| Metodología del proyecto | 4 |
| Objetivos específicos | 4 |
| Listado de necesidades | 4 |
| Refinamiento de requerimientos funcionales | 8 |
| Plan SCM | 3 |
| Plan de capacitación | 3 |

| | |
|--------------|-----------|
| Plan SQA | 4 |
| Total | 30 |

Gantt



18.1.3 Sprint 3

| | |
|--|----------------------------|
| Inicio: 14/10/24 | Fin: 20/10/24 |
| Tarea | Estimación en horas |
| Gestión de riesgos | 10 |
| Cronograma | 14 |
| Arquitectura | 3 |
| Gestión de stakeholders | 2 |
| Supuestos | 2 |
| Bibliografía | 2 |
| Anexos | 4 |
| Conformación del documento de anteproyecto | 8 |
| Total | 45 |

Gantt



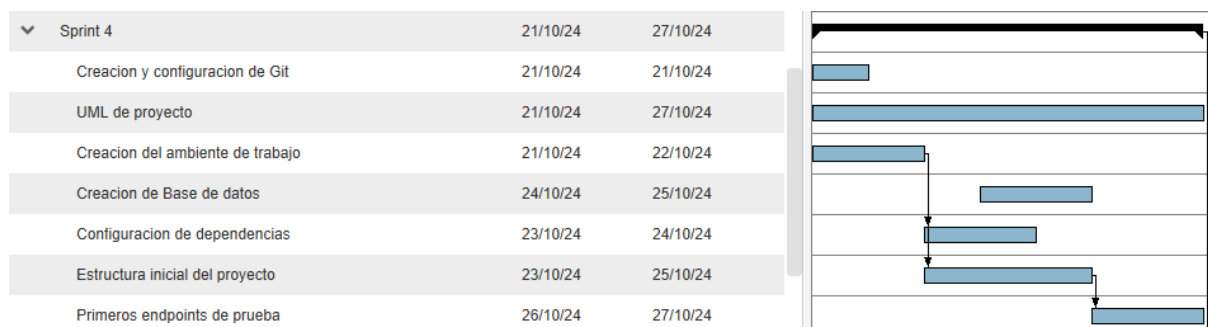
18.2 Inicio de desarrollo

18.2.1 Sprint 4

Para este sprint buscaremos tener todo el entorno de trabajo configurado para comenzar la etapa de desarrollo. A la vez también haremos un diagrama UML para facilitarnos la construcción de la solución y algunos endpoints básicos de prueba para comprobar que funcione la conexión con la base de datos.

| Inicio: 21/10/24 | Fin: 27/10/24 |
|---------------------------------|---------------------|
| Tarea | Estimación en horas |
| Creación y configuración de Git | 1 |
| UML de proyecto | 14 |
| Creación de ambiente de trabajo | 4 |
| Creación de Base de datos | 5 |
| Configuración de dependencias | 4 |
| Estructura inicial del proyecto | 6 |
| Primeros endpoints de prueba | 6 |
| Total | 40 |

Gantt



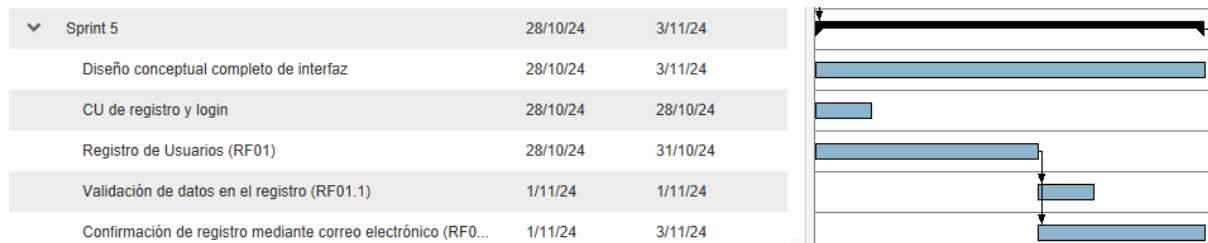
18.2.2 Sprint 5

Este sprint esta enfocado en diseñar bosquejos de todas las interfaces del proyecto que nos servirán de guía para el diseño final y para mostrarle al cliente una visión previa del proyecto para que apruebe. Desarrollaremos también en paralelo los primeros requerimientos del programa. En cada sprint que se desarrolle requerimientos se hará también el documento de casos de uso del mismo.

| Inicio: 28/10/24 | Fin: 03/11/24 |
|--|---------------------|
| Tarea | Estimación en horas |
| Diseño conceptual completo de interfaz | 20 |
| CU de registro | 1 |
| Registro de usuarios (RF01) | 14 |

| | |
|---|-----------|
| Validación de datos en el registro (RF01.1) | 4 |
| Confirmación de registro mediante correo electrónico (RF01.2) | 6 |
| Total | 45 |

Gantt

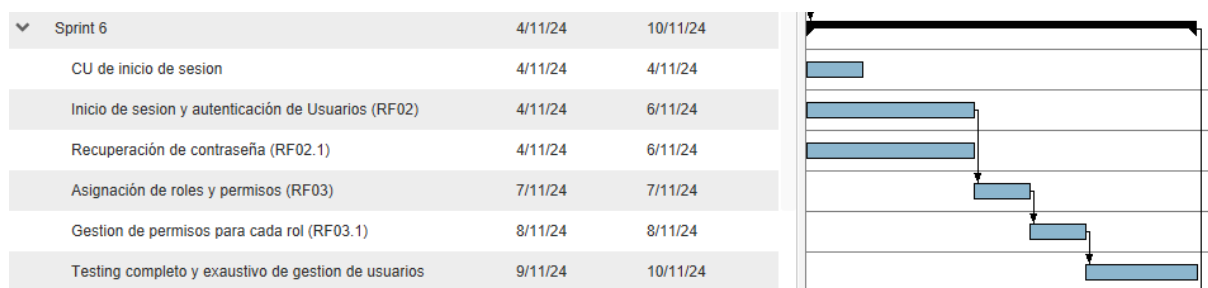


18.2.3 Sprint 6

Teniendo ya del anterior sprint la parte del registro de usuario, continuaremos con el inicio de sesión y la autenticación del usuario en el sistema. Finalizaremos este sprint testeando todo lo avanzado hasta el momento.

| Inicio: 04/11/24 | Fin: 10/11/24 |
|---|---------------------|
| Tarea | Estimación en horas |
| CU de inicio de sesión | 1 |
| Inicio de sesión y autenticación de Usuarios (RF02) | 12 |
| Recuperación de contraseña (RF02.1) | 8 |
| Asignación de roles y permisos (RF03) | 4 |
| Gestión de permisos para cada rol | 4 |
| Testing completo de gestión de usuarios | 10 |
| Total | 39 |

Gantt

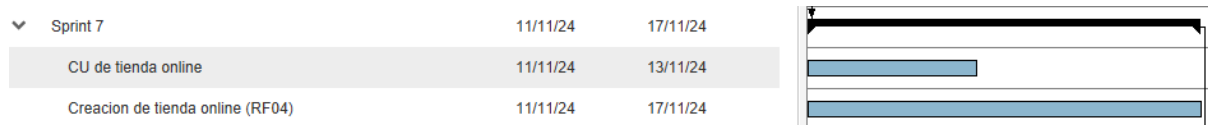


18.2.4 Sprint 7

En este sprint comenzaremos a desarrollar la tienda online. Esta etapa es una etapa de gran incertidumbre por lo que puede darse variaciones en los tiempos de desarrollo.

| | |
|----------------------------------|----------------------------|
| Inicio: 11/11/24 | Fin: 17/11/24 |
| Tarea | Estimación en horas |
| CU de tienda online | 1 |
| Creación de tienda online (RF04) | 44 |
| Total | 45 |

Gantt



18.2.5 Sprint 8

Continuaremos con el desarrollo de la tienda, incorporaremos un carrito y sus funcionalidades.

| | |
|--|----------------------------|
| Inicio: 18/11/24 | Fin: 24/11/24 |
| Tarea | Estimación en horas |
| Visualización del catálogo de productos (RF04.1) | 25 |
| AB carrito (RF04.2) | 20 |
| Total | 45 |

Gantt



18.2.6 Sprint 9

Idealmente en este sprint se finalizara la creación de la tienda implementando las funcionalidades relacionadas a la compra y métodos de pago con la API de MercadoPago.

| | |
|---|----------------------------|
| Inicio: 25/11/24 | Fin: 01/12/24 |
| Tarea | Estimación en horas |
| Finalización de compra y métodos de pago (RF04.3) | 14 |
| Pruebas de tienda online | 25 |
| Total | 39 |

Gantt



18.2.7 Sprint 10

Para esta etapa implementaremos el sistema de membresías solicitado por el cliente, además del registro y reserva de eventos usando la API de Google Calendar.

| Inicio: 02/12/24 | Fin: 08/12/24 |
|---|---------------------|
| Tarea | Estimación en horas |
| CU de membresías | 1 |
| Sistema de membresías pagas (RF05) | 15 |
| Gestión de tipos de membresías (RF05.1) | 10 |
| Registro y reserva de eventos (RF06) | 10 |
| Visualización de eventos en calendario de Google (RF06.1) | 6 |
| Total | 42 |

Gantt



18.2.8 Sprint 11

En este sprint paralelizaremos las tareas en dos grandes requisitos para dividirnos mejor el trabajo. Comenzaremos con los grupos para miembros y la gestión del perfil de usuario.

| Inicio: 09/12/24 | Fin: 15/12/24 |
|--|---------------------|
| Tarea | Estimación en horas |
| CU de grupos | 1 |
| CU de perfil | 1 |
| Grupos interactivos para miembros (RF07) | 22 |
| Gestión del perfil personal (RF08) | 20 |
| Total | 44 |

Gantt



18.2.9 Sprint 12

Continuaremos con lo planteado en el anterior sprint con el chat grupal y sus funcionalidades adjuntas. Además también incorporaremos un historial de compras y continuaremos con la gestión del perfil y sus datos.

| Inicio: 16/12/24 | Fin: 22/12/24 |
|-------------------------------------|---------------------|
| Tarea | Estimación en horas |
| Chat grupal para miembros | 28 |
| Compartir archivos dentro del grupo | 4 |
| Ver historial de compras | 6 |
| Modificación de datos personales | 6 |
| Gestor de reservas | 15 |
| Total | 59 |

Gantt



Aclaración: En el sprint 12 se realiza una ligera sobrecarga para compensar la falta de trabajo durante las fiestas

18.2.10 Sprint 13

Para este sprint se planea dar foco a la pantalla de administración para administradores de Tiféret.

| Inicio: 23/12/24 | Fin: 29/12/24 |
|--|---------------------|
| Tarea | Estimación en horas |
| CU de administración | 1 |
| Panel de administración para gestión de contenido (RF09) | 30 |
| Total | 31 |

Gantt



18.2.11 Sprint 14

Esta semana se priorizará el ABM de productos y eventos, además de también incorporar el envío de estos productos o eventos pagos a los usuarios que hayan pagado de manera presencial en el centro. También se añadirá funcionalidad nativa de notificaciones donde se requiera.

| Inicio: 30/12/24 | Fin: 05/01/25 |
|--|---------------------|
| Tarea | Estimación en horas |
| ABM de productos y eventos (RF09.1) | 14 |
| Enviar productos de pago presencial (RF09.2) | 6 |
| Envío de notificaciones (RF09.3) | 10 |
| Total | 30 |



Aclaración: Tanto el sprint 13 como el sprint 14 se cuentan como dos semanas de bajo rendimiento dado que coinciden con fechas como navidad y fin de año.

18.2.12 Sprint 15

Este sprint añadiremos un área de soporte y contacto que se maneja vía mail. Además crearemos cualquier documento de caso de uso que nos haya faltado y también los que

están por venir. Haremos pruebas de todo lo avanzado hasta la fecha y añadimos también una tarea opcional de desarrollar el sistema de respuestas automáticas (el desarrollo de esta se hará teniendo en consideración los tiempos del equipo ya que no es una funcionalidad clave para el cliente)

| | |
|--|----------------------------|
| Inicio: 06/01/25 | Fin: 12/01/25 |
| Tarea | Estimación en horas |
| Soporte y contacto (RF10) | 14 |
| Creación de CU restante | 2 |
| Pruebas funcionales | 16 |
| Sistema de respuestas automáticas (RF10.1) | 15 |
| Total | 47 |

Gantt



18.2.13 Sprint 16

Para este sprint vamos a desarrollar el sitio de noticias del centro en el cual podrán agregar, editar o eliminar noticias que aparecerán en la pagina principal. Además también agregaremos la información estática presentando al centro Tiféret con la que cuentan actualmente en su pagina hecha en Wix.

| | |
|--|----------------------------|
| Inicio: 06/01/25 | Fin: 12/01/25 |
| Tarea | Estimación en horas |
| ABM Noticias y blog informativo (RF11) | 35 |
| Página principal estática | 8 |
| Total | 43 |

Gantt



18.2.14 Sprint 17

Priorizaremos la seguridad de la pagina esta semana, nos aseguraremos de que los datos de los usuarios estén correctamente encriptados para solidificar el registro.

| | |
|--|----------------------------|
| Inicio: 20/01/25 | Fin: 26/01/25 |
| Tarea | Estimación en horas |
| Seguridad de datos y encriptación (RF12) | 24 |
| Autenticación multifactor (MFA) (RF12.1) | 30 |
| Total | 54 |

Gantt



18.2.15 Sprint 18

Esta semana es considerada una semana de holgura en caso de atraso. En caso de que el cronograma se este cumpliendo en tiempo y forma esta semana se utilizara para hacer pruebas completas del sistema y pulir el código.

| | |
|-------------------------------------|----------------------------|
| Inicio: 27/01/25 | Fin: 02/02/25 |
| Tarea | Estimación en horas |
| Testing y reparación – Primer etapa | 49 |
| Total | 49 |

Gantt



18.2.16 Sprint 19

Este sprint será similar al anterior, testaremos y puliremos código. Además de eso también haremos un deploy provisional de la pagina y documentaremos lo requerido para la primera entrega solicitada por la universidad.

| | |
|--------------------------------------|----------------------------|
| Inicio: 03/02/25 | Fin: 11/02/25 |
| Tarea | Estimación en horas |
| Testing y reparación – Segunda etapa | 20 |
| Deploy | 10 |
| Documentación | 20 |
| Total | 50 |

Gantt



El total estimado de horas de esfuerzo de los sprints del 1 al 19 será de 812 horas

Se incluyeron todos los requerimientos en los sprints, esto incluye los requerimientos obligatorios, los deseables y los opcionales definidos en el alcance del producto.

19. Bibliografía

Guías de desarrollo de Progressive Web Apps (PWA):

- Google Developers. (2020). Introducción a las aplicaciones web progresivas (PWA). Recuperado de <https://developers.google.com/web/progressive-web-apps/>

Norma ISO para la gestión de riesgos:

- Organización Internacional de Normalización. (2018). ISO 31000: Gestión del riesgo - Directrices. ISO.

Guía de iniciación de Material UI

- <https://mui.com/material-ui/getting-started/>. Copyright © 2024 Material UI SAS, trading as MUI.

Manual de API de integración de pagos:

- MercadoPago. (2020). Guía de integración API MercadoPago. Recuperado de <https://www.mercadopago.com.ar/developers/es/guides/>

ChatGPT:

- OpenAI. (2024). ChatGPT. <https://openai.com/chatgpt>

20. Anexos

20.1 Anexo 1: Guía de implementación de PWA

20.1.1 Crear la aplicación React con el soporte PWA activado

Si vas a crear una nueva aplicación React, puedes habilitar el soporte PWA desde el inicio usando `create-react-app` con el flag `--template pwa`.

```
npx create-react-app my-app --template pwa
```

Esto generará una aplicación React preconfigurada con lo necesario para ser una PWA (manifest, service workers, etc.).

Si ya tienes una aplicación React existente, puedes hacer los siguientes pasos para agregar soporte PWA manualmente.

20.1.2 Instalar los archivos necesarios

Si no lo tienes ya, asegúrate de tener los siguientes archivos en tu proyecto:

- `manifest.json`: Este archivo describe los metadatos de la aplicación, como el nombre, iconos y colores de tema.
- Service Worker: Este archivo permite a tu aplicación funcionar offline y manejar recursos en segundo plano.

Si usas `create-react-app`, ya deberías tener estos archivos configurados. Vamos a verificar y personalizar algunas partes.

20.1.3 Configurar `manifest.json`

En la carpeta `public/`, busca o crea un archivo `manifest.json`. Aquí es donde defines los detalles de cómo se verá la aplicación cuando se "instale" en un dispositivo móvil o escritorio.

Ejemplo de archivo `manifest.json`:

json

```
{  
  "short_name": "MyApp",  
  "name": "My Progressive Web App",  
  "icons": [  
    {  
      "src": "favicon.ico",
```

```

    "sizes": "64x64 32x32 24x24 16x16",
    "type": "image/x-icon"
  },
  {
    "src": "logo192.png",
    "type": "image/png",
    "sizes": "192x192"
  },
  {
    "src": "logo512.png",
    "type": "image/png",
    "sizes": "512x512"
  }
],
"start_url": ".",
"display": "standalone",
"theme_color": "#000000",
"background_color": "#ffffff"
}

```

20.1.4 Habilitar el Service Worker

Si usas create-react-app, el archivo `service-worker.js` ya está presente y puedes encontrarlo en la carpeta `src/`. Para que esté activo en tu aplicación, simplemente asegúrate de registrar el service worker.

Abre el archivo `index.js` en la carpeta `src/` y busca esta línea:

```
serviceWorkerRegistration.unregister();
```

Cámbiala a:

```
serviceWorkerRegistration.register();
```

Esto activará el service worker para que funcione en segundo plano, lo que permitirá funcionalidades como el soporte offline y la carga rápida de la app.

20.1.5 Asegurarte de que tu app sea responsive

Si tu sitio aún no es responsive, es importante que uses CSS flexbox, grid y media queries para que se adapte a diferentes tamaños de pantalla, especialmente móviles.

Ejemplo básico de media queries en CSS para móviles:

```
@media (max-width: 768px) {  
  
  .container {  
  
    flex-direction: column;  
  
  }  
  
}
```

20.1.6 Construir y probar tu PWA

Una vez que tengas todo configurado, puedes compilar tu aplicación y probarla.

- Ejecuta `npm run build` para crear una versión optimizada de tu aplicación.
- Sube esta versión a un servidor y visita la aplicación en tu navegador.
- Prueba si tu PWA es "instalable" y si funciona offline:
 - En Chrome, abre las herramientas de desarrollador (Ctrl+Shift+I), ve a la pestaña Application y en el menú lateral busca Service Workers.
 - Aquí podrás ver si tu service worker está registrado correctamente y también tendrás opciones para simular el estado offline.

20.1.7 Testing PWA

- Para probar si tu aplicación cumple con los estándares PWA, puedes usar Lighthouse en las DevTools de Chrome. Solo ve a la pestaña de "Lighthouse" y genera un reporte. Esto te dará una puntuación y sugerencias para mejorar tu aplicación.