



CODE COFFEE

DOCENTE: Remigio Santos Huarcaya Almeyda

CURSO: Herramientas de Desarrollo

INTEGRANTES (GRUPO 10)

- Anthony Miguel Cotrina Vasquez
- Brian Danny Silva Davila
- Nataly Noelia Ramos Meza

2025 - JULIO

Especialidad	Ingeniería de Sistemas e Informática
Participantes	<ul style="list-style-type: none">- Silva Davila, Brian Danny- Cotrina Vasquez, Anthony Miguel- Ramos, Nataly <i>Pagina Ecommerce para la cafeteria Code Coffee</i>
Fecha	22 de julio 2025

Objetivo

Definir la estructura del informe técnico del proyecto final para la evaluación final del curso de **Herramientas de desarrollo**.

Actividades

1. Descripción del proyecto y sus componentes en el repositorio github

El proyecto Code Coffee es una plataforma web de e-commerce para una cafetería ubicada en el distrito de Pachacamac, desarrollada como aplicación full-stack con arquitectura cliente-servidor. El repositorio en GitHub (<https://github.com/anthonyss71/codecoffe.git>) contiene la siguiente estructura:

Componentes del Backend (/backend):

- config/: Configuraciones de base de datos y variables de entorno
- controllers/: Lógica de negocio y manejo de rutas API
- middleware/: Funciones intermedias para autenticación y validación
- models/: Modelos de datos para interacción con MySQL
- routes/: Definición de endpoints de la API REST
- server.js: Punto de entrada del servidor Express.js
- testMacDB.js: Archivo de pruebas de conectividad de base de datos

Componentes del Frontend (/frontend):

- src/components/: Componentes React reutilizables incluyendo:
- About.jsx, Admin.jsx: Páginas informativas y panel administrativo

- bebidas.jsx, postres.jsx: Catálogos de productos
- Login.jsx, Register.jsx: Sistema de autenticación
- Dashboard.jsx, Menu.jsx, Navbar.jsx: Navegación principal
- tiendas.jsx, Ubicaciones.jsx: Información de establecimientos

src/services/: Servicios para comunicación con APIs

src/styles/: Hojas de estilo CSS

public/: Recursos estáticos y archivos públicos

2. Informe técnico final del proyecto

a. Resumen Ejecutivo

El proyecto "Code Coffee – Plataforma E-commerce para Cafetería" se desarrolló a lo largo de 8 semanas bajo una dinámica de trabajo colaborativa utilizando la metodología ágil SCRUM. Esta aplicación full-stack permite a los usuarios explorar un catálogo dinámico de productos, realizar pedidos en línea, visualizar ubicaciones físicas y contar con soporte automatizado a través de un chatbot. El desarrollo se dividió en dos sprints principales, que abarcaron desde la implementación de las funcionalidades core hasta la integración de herramientas complementarias y mejoras visuales.

Durante el desarrollo, se priorizaron historias de usuario mediante la técnica MoSCoW, permitiendo diferenciar entre funciones críticas y mejoras deseables. El enfoque ágil facilitó entregas incrementales y retroalimentación constante entre los miembros del equipo. La integración continua mediante GitHub garantizó control de versiones y colaboración fluida, mientras que el uso de herramientas como ClickUp, Notion, Slack y LandBot permitió un ecosistema de desarrollo profesional, imitando un entorno real de trabajo remoto.

Al finalizar el proyecto, se logró una implementación funcional y estable, cumpliendo con el 90% de las funcionalidades críticas planteadas inicialmente. La experiencia permitió identificar áreas clave de mejora, como la optimización de rendimiento, incorporación de pasarelas de pago reales y mayor

adaptabilidad móvil. Sin embargo, se sentaron bases sólidas en cuanto a diseño de arquitectura, colaboración interfuncional y despliegue efectivo de soluciones digitales orientadas a PyMEs locales.

Metodologías utilizadas:

- Scrum con sprints de 2 semanas
- Desarrollo ágil con historias de usuario
- Priorización MoSCoW para épicas
- Integración continua con Git/GitHub

Herramientas utilizadas:

- Desarrollo: React.js, Express.js, MySQL, Node.js
- Gestión de proyecto: ClickUp
- Comunicación: Slack, WhatsApp
- Documentación: Notion
- Diseño: Figma
- Control de versiones: GitHub
- Chatbot: LandBot

Duración del proyecto: 8 semanas (2 sprints de desarrollo + planificación)

Porcentaje de tareas completas: 90% de las funcionalidades core implementadas

Puntos de mejora: Optimización de performance, implementación de sistema de pagos real, mejoras en la interfaz móvil.

b. Estructura del equipo SCRUM y las funciones realizadas

INTEGRANTES	ROL	FUNCIONES REALIZADAS
Anthony Cotrina Vásquez	Scrum Master	Lideró reuniones, facilitó sprints, gestionó tareas
Brian Danny Silva Davila	Desarrollador Backend	Gestión de base de datos y lógica en ExpressJS
Nataly Ramos Meza	Desarrollador Frontend	Implementó componentes en React, conexión API

Product Owner y Scrum Master: Anthony Miguel Cotrina Vasquez

- Definición de épicas y historias de usuario
- Priorización del product backlog
- Coordinación de sprints y retrospectivas

Developer: Brian Danny Silva Davila

- Desarrollo del backend con Express.js
- Implementación de APIs REST
- Configuración de base de datos MySQL
-

Developer: Nataly Noelia Ramos Meza

- Desarrollo del frontend con React.js
- Diseño de interfaces de usuario
- Integración frontend-backend

c. **Protocolos de Comunicación establecidas**

Comunicación Diaria

- Daily Standup virtual vía Slack a las 10:00 AM
- Reporte de progreso en Slack
- Resolución de impedimentos inmediatos vía WhatsApp

Comunicación Semanal

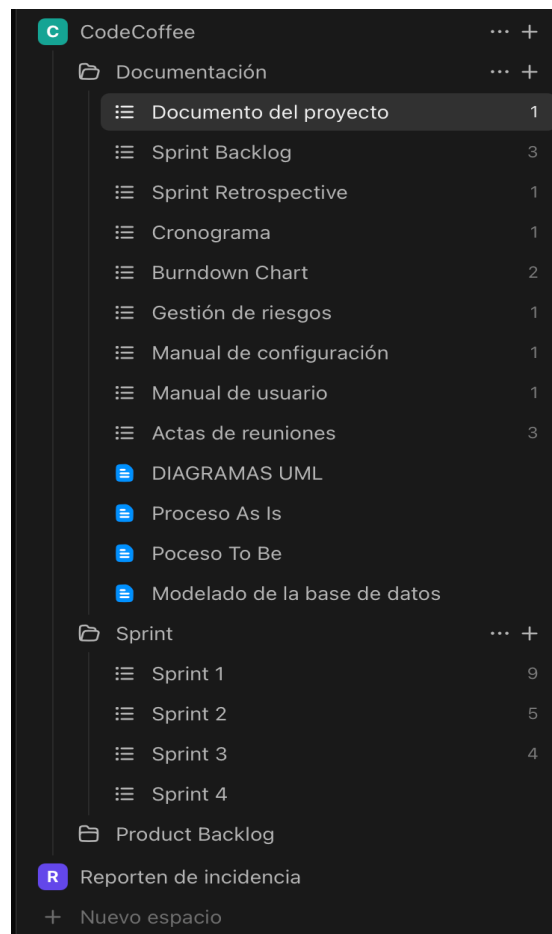
- Sprint Planning los lunes vía videollamada
- Sprint Review los viernes con demo funcional
- Retrospectiva y documentación en Notion
- Actualización de dashboards en ClickUp

Comunicación de Emergencia

- Canal Slack #emergencias para issues críticos
- WhatsApp para contacto inmediato fuera de horario

d. **Estructura jerárquica implementada**

La estructura organizacional del proyecto en ClickUp se estableció de la siguiente manera:



e. Tabla de estados y campos personalizados

- **PENDIENTE (0):** Tareas programadas en el backlog que estan en progreso para esta entrega final.
- **EN PROGRESO (3 tareas):** Tareas actualmente en desarrollo por los miembros del equipo.
- **COMPLETADAS (36 tareas):** Tareas finalizadas y validadas, incluyendo todas las HU desarrolladas durante los 3 Sprint.

HU01: HU-Módulo de Login de usuario Diseño de interfaz de login Implementación del backend de autenticación JWT Integración frontend-backend login Pruebas unitarias	HU03: HU - Como docente, quiero editar mi perfil con información básica y áreas de interés para que los estudiantes conozcan mi especialidad. Implementar funcionalidad en el backend para actualizar perfil Crear endpoint para actualizar perfil y probarlo en postman Implementar la interfaz en el frontend Probar edición del perfil	HU05: HU - Como administrador, quiero crear cursos y al momento de crearlo asignarle un docente Crear la lógica en el backend para crear curso Crear endpoint para crear curso y probarlo en postman Diseñar un formulario en el frontend Actualizar vista de cursos del docente
HU02: HU-Registro de nuevos usuarios Diseño de formulario de registro Implementación del endpoint de registro Validación de datos y pruebas	HU04: HU - Como estudiante, quiero visualizar mi información básica y poder editarla Implementar funcionalidad de editar perfil en el backend Crear endpoint para actualizar perfil y probarlo en postman Diseñar la interfaz de usuario para editar información básica	HU6 - Como docente, quiero subir archivos o enlaces educativos y organizarlos por curso, para que mis estudiantes encuentren fácilmente el material correspondiente. Crear lógica del backend que permita que el docente suba sus archivos a cierto curso el docente pueda subir recursos en el frontend
HU03: HU-Vista Menú principal con navbar y footer Diseño y maquetado del navbar Implementación del menú principal Diseño y maquetado del footer	HU06: HU-Vista Establecimientos Maquetado de pagina de establecimientos HU07: HU-Carrito de Compras funcional Diseño de interfaz de carrito Logica de agregar/eliminar productos Persistencia del carrito en BD HU08: HU-Cerrar Sesión Implementación de logout HU09: HU-Proceso de pago simulado (checkout) Diseño de interfaz de checkout Implementación de proceso de pago simulado	HU015: HU-Modal de detalle de productos con efecto expansivo Diseño del modal Implementación y pruebas del efecto expansivo
HU04: HU-Categoría de bebidas Diseño de vista bebidas Creación de endpoints para bebidas Integración y carga de datos		HU016: HU-Dashboard administrativo con autenticación de administrador Maquetación del dashboard Lógica de autenticación de administrador Estilo y validación del dashboard HU017: HU-Rediseño y mejora del chatbot Rediseño visual del chatbot Ajustes de flujo conversacional y pruebas
HU05: HU-Categoría de postres Diseño de vista postres Creación de endpoints para postres Integración y carga de datos		HU018: HU-Rediseño del carrito de compras Pruebas de funcionalidad del carrito

○ PLANEANDO 0 + Agregar Tarea	○ EN PROGRESO 0 ... +	● COMPLETADAS 36 ... +
	HU01-Módulo de Login de usuarios BD A abr. 17 Alta 4 subtareas	
	HU02-Registro de nuevos usuarios BD A abr. 18 Alta 3 subtareas	
	HU03-Vista Menú principal con navbar y footer A abr. 19 Alta 3 subtareas	
	HU04-Categoría de bebidas BD A abr. 21 Alta	

3. Gestión de tareas

a. Metodologías Implementadas y Características

Sprint de 2 semanas:

- Sprint 1: Funcionalidades core (autenticación, catálogo, carrito)
- Sprint 2: Mejoras arquitectónicas y herramientas complementarias

Daily Virtuales:

- Reuniones diarias de 15 minutos vía Slack
- Check-in de progreso y identificación de impedimentos
- Actualización en tiempo real del tablero Kanban

Sprint reviews con dashboards personalizados:

- Métricas de velocidad del equipo
- Revisión de burn-down charts

b. Dashboards implementados

Progreso General:

- Vista consolidada de todas las épicas
- Porcentaje de completitud por sprint
- Timeline de entregables

Velocidad del Equipo:

- Promedio de 8-10 story points por sprint
- Tracking de capacity vs. Commitment
- Identificación de bottlenecks

c. Descripción de herramientas usadas:

Vista Gantt:

- Planificación temporal de sprints
- Dependencias entre tareas

Alertas Automáticas:

- Notificaciones de tareas vencidas
- Recordatorios de daily standups
- Escalamiento de issues bloqueados

Tablero Kanban:

- **Visualización del flujo de trabajo**

4. Arquitectura de Software Relaciones entre:

La arquitectura del proyecto “Code Coffee” fue diseñada teniendo en cuenta la escalabilidad, accesibilidad multiplataforma y una correcta separación entre el

frontend, backend y la base de datos. A continuación, se describen los elementos clave que conforman esta arquitectura:

a. Sistema operativo

El sistema está diseñado para ser accesible desde cualquier dispositivo que cuente con un navegador web moderno, independientemente del sistema operativo. Esto incluye equipos con Windows, macOS, Linux, así como dispositivos móviles Android e iOS. La plataforma es 100% web responsive, lo que permite su correcto funcionamiento en navegadores móviles y de escritorio, sin necesidad de instalación local, garantizando una experiencia de usuario fluida desde cualquier sistema operativo con acceso a internet.

b. Navegadores

La aplicación web fue desarrollada siguiendo estándares modernos de compatibilidad y rendimiento, asegurando su correcto funcionamiento en los navegadores más utilizados, entre ellos:

- Google Chrome (principal navegador utilizado durante el desarrollo)
- Mozilla Firefox
- Microsoft Edge
- Safari (tanto en macOS como en iOS)

c. Componentes de nube de computación

Para el despliegue y funcionamiento en línea de la plataforma, se utilizaron diversos componentes en la nube, esenciales para garantizar disponibilidad y continuidad del servicio:

- **Dominio y Hosting Web:** Se utilizó Hostinger para el hospedaje del sitio web y la gestión del dominio. Esto permitió desplegar tanto el frontend como el backend de manera accesible en la web pública.
- **Repositorio en la nube:** Se implementó GitHub como repositorio remoto para el control de versiones y la colaboración entre los integrantes del equipo de desarrollo.
- **Gestión de tareas en la nube:** Se utilizó ClickUp, una herramienta alojada en la nube, para la gestión de sprints, tareas y responsabilidades del equipo.

- **Chatbot en la nube:** Se integró un ChatBot informativo utilizando LandBot, también alojado en servidores cloud, para dar soporte a consultas frecuentes de los usuarios.

d. Software (Base de datos y software de desarrollo)

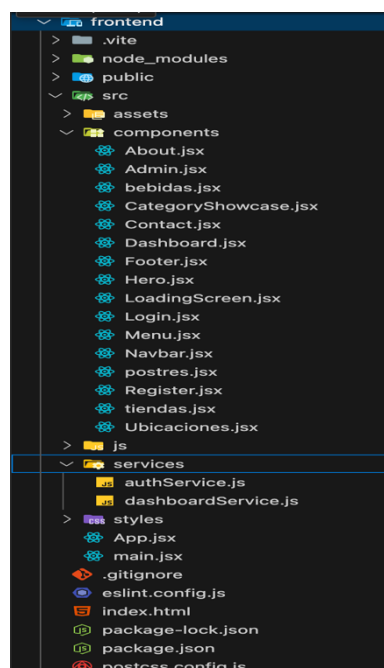
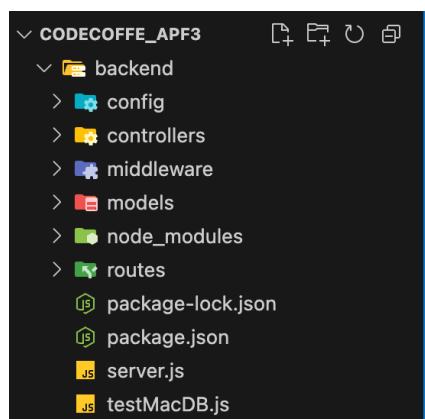
Para el desarrollo del backend se utilizó Node.js con el framework ExpressJS, lo que permitió crear un servidor robusto y modular. En cuanto a la base de datos, se optó por MySQL, gestionada localmente a través de phpMyAdmin usando XAMPP como entorno de desarrollo. Esta elección facilitó la configuración, gestión y pruebas de la base de datos en entornos de desarrollo locales antes del despliegue en el servidor web.

El frontend fue desarrollado con React.js como biblioteca principal, complementado con TailwindCSS para el diseño de la interfaz de usuario. Esto permitió mantener una arquitectura desacoplada entre la interfaz visual y la lógica del servidor, promoviendo buenas prácticas de desarrollo moderno.

e. Patrón de diseño

Durante el desarrollo de Code Coffee, se adoptaron varios patrones de diseño para garantizar la escalabilidad y el mantenimiento del sistema:

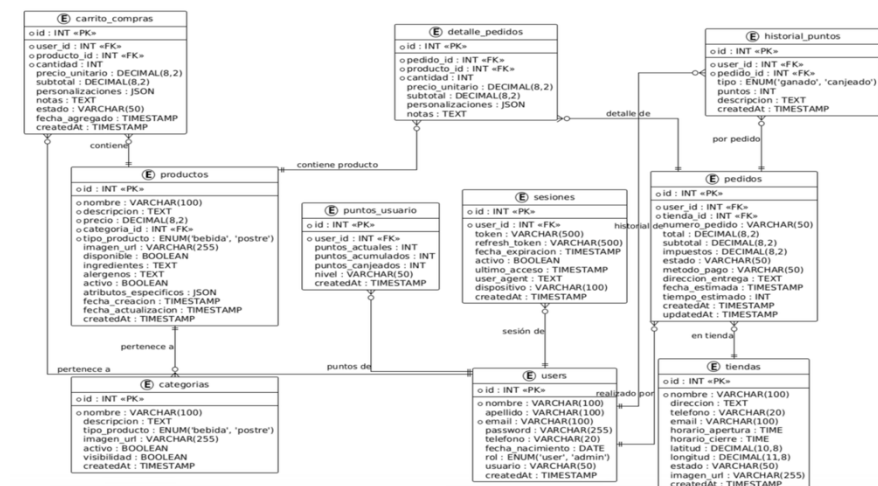
- **Modelo MVC (Modelo-Vista-Controlador):** aplicado en el backend, para separar la lógica de negocio (controladores), la gestión de datos (modelos) y la presentación (vistas o endpoints).



- **Componentización en React:** el frontend está estructurado en componentes reutilizables, lo que permite escalar nuevas funcionalidades sin afectar otras partes del sistema.
- **Arquitectura de Casos de Uso:** se diseñaron historias de usuario y épicas claras, que definieron cada funcionalidad desde el punto de vista del cliente.



- **Arquitectura de Datos:** la base de datos fue modelada con un enfoque relacional, garantizando la integridad y relaciones correctas entre entidades como usuarios, productos y pedidos.



- Arquitectura del Software: se estructuró el sistema en capas (frontend, backend, base de datos), cada una con responsabilidades bien definidas y comunicadas a través de APIs RESTful.

5. Integración con otras herramientas

HERRAMIENTA	USO EN EL PROYECTO
GitHub	Control de versiones y código
Slack	Comunicación técnica
Google Drive	Almacenamiento compartido de recursos
Notion	Actas de reunión, documentación, cronograma de actividades, enlaces a las herramientas.
ClickUp	Gestión de tareas, documentación y estructura SCRUM en la organización
LandBot	Chatbot creado para atender las necesidades del cliente

6. Desafíos enfrentados

a. Configuración del proyecto en clickUp

Uno de los primeros desafíos fue la configuración inicial del entorno de gestión de tareas en ClickUp, una herramienta nueva para el equipo. Al inicio del Sprint 1, la curva de aprendizaje representó una barrera, ya que no todos los miembros estaban familiarizados con la creación de espacios de trabajo, la asignación de tareas por prioridad ni la utilización de campos personalizados. Sin embargo, con el avance del proyecto, se logró una apropiación progresiva de la herramienta, permitiendo aprovechar funcionalidades como vistas Gantt, etiquetas por sprint y paneles Kanban personalizados.

b. Resistencia del equipo

En las primeras semanas, el equipo enfrentó dificultades relacionadas con la disponibilidad horaria, debido a compromisos académicos y

laborales individuales. Esto ocasionó retrasos en la sincronización de tareas y limitó la asistencia simultánea a reuniones. A través de acuerdos internos y un calendario común, se logró establecer rutinas de coordinación semanales, y se asignaron responsables por funcionalidades para garantizar el cumplimiento de objetivos.

c. Falta de compromiso puntual

En momentos puntuales del proyecto, se evidenció una falta de seguimiento riguroso por parte de algunos miembros respecto al avance de tareas asignadas. Esto derivó en retrasos en entregables intermedios, afectando el ritmo del equipo y la interdependencia funcional. Para mitigar este problema, se implementaron controles diarios de avance mediante Slack y se reforzó la importancia de la responsabilidad individual en el cumplimiento de los objetivos grupales.

d. Estimación inexacta

Durante la planificación de los primeros sprints, algunas tareas fueron subestimadas en cuanto a complejidad técnica y tiempo requerido. Esto se debió a una falta de experiencia en la evaluación de story points y a imprevistos técnicos en la integración de APIs o fallos de compatibilidad en navegadores. Como respuesta, se ajustaron los cronogramas en tiempo real y se replantearon estimaciones más conservadoras para los sprints finales.

7. Lecciones aprendidas

A lo largo del proceso de desarrollo, nuestro equipo logró identificar diversos aprendizajes clave que resultaron fundamentales para la ejecución exitosa del proyecto. Estas lecciones pueden ser aplicadas en proyectos futuros para mejorar la organización, productividad y colaboración:

a. Factores críticos de éxito

La adopción disciplinada de la metodología SCRUM, junto con el uso constante de herramientas de apoyo como ClickUp, permitieron mantener una estructura clara de trabajo. La definición de roles, las historias de usuario bien planteadas y los objetivos de sprint definidos facilitaron el avance progresivo del sistema. Además, las reuniones semanales ayudaron a mantener al equipo alineado y con visibilidad del estado general del proyecto.

b. Implementación exitosa del sistema

Gracias a la buena coordinación técnica entre frontend y backend, así como una arquitectura bien definida, se logró completar una integración funcional de todos los componentes del sistema. El proyecto fue desplegado satisfactoriamente en la web, cumpliendo los requisitos técnicos y funcionales establecidos al inicio.

c. Errores evitados tras experiencias tempranas

Tras enfrentar los primeros retrasos y descoordinaciones, el equipo logró adaptar su metodología para mejorar la distribución del trabajo y ajustar los tiempos de entrega. Se reforzó la comunicación y se estandarizó el uso de herramientas colaborativas, evitando así repetir errores relacionados con la desorganización y las tareas sin seguimiento.

8. Recomendaciones para futuros proyectos

A partir de la experiencia obtenida en el desarrollo de Code Coffee, se plantean las siguientes recomendaciones para equipos que inicien proyectos similares, especialmente en entornos académicos o colaborativos remotos:

a. Configuración inicial del clickUp

Es fundamental invertir tiempo al comienzo del proyecto en configurar adecuadamente la herramienta de gestión de tareas. Esto incluye definir claramente los estados de cada tarea (pendiente, en progreso, revisión, completado), personalizar campos según los sprints, estimaciones y responsables, y establecer flujos de trabajo acordes con la metodología SCRUM. Una buena configuración permite ahorrar tiempo a largo plazo y da visibilidad completa del avance del proyecto.

b. Asegurar horarios compatibles desde el inicio

Antes de comenzar las actividades técnicas, se recomienda coordinar con todos los integrantes del equipo horarios compatibles para reuniones semanales y revisión de tareas. Establecer un calendario común desde el inicio permite mayor sincronización, previene malentendidos y fortalece la colaboración continua.

c. Integrar pruebas desde el primer sprint

Una de las recomendaciones más importantes es implementar pruebas unitarias y funcionales desde el inicio del desarrollo. Esto permite identificar errores tempranamente, asegurar calidad en el código y evitar acumulación de fallos que pueden escalar a problemas mayores en etapas más avanzadas del proyecto.

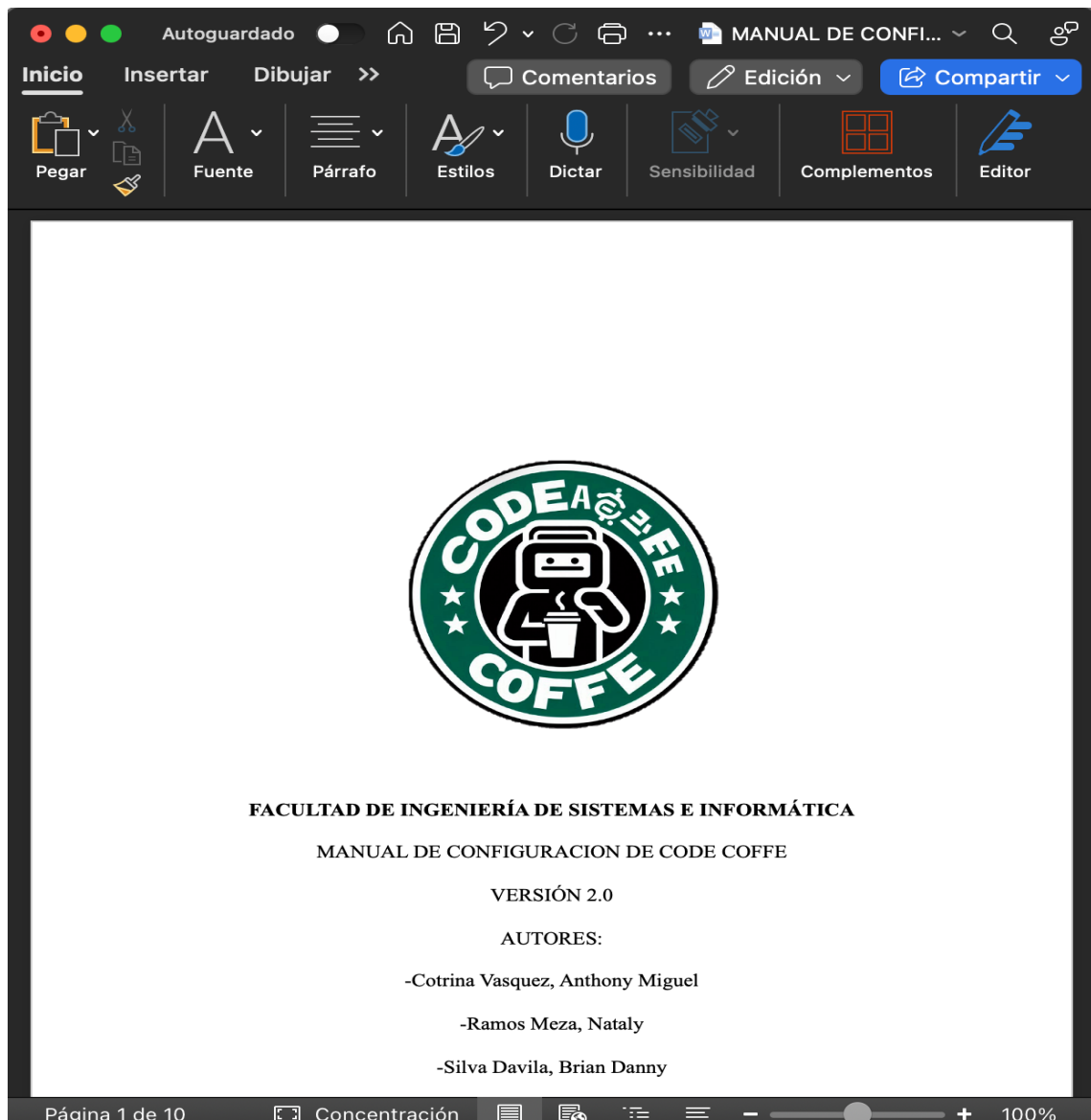
9. Conclusiones

- El desarrollo de la plataforma "Code Coffee" representó una experiencia integral de aplicación de conocimientos técnicos y metodológicos en un entorno colaborativo simulado de desarrollo profesional. El uso de la metodología SCRUM con sprints bien definidos y herramientas modernas como ClickUp, React, Express.js y MySQL permitió no solo cumplir con los objetivos técnicos del proyecto, sino también fortalecer habilidades clave como la organización, la planificación ágil y la colaboración en equipo. El proyecto culminó con una plataforma funcional y escalable, capaz de responder a las necesidades básicas de una cafetería moderna en el entorno digital.
- Uno de los aprendizajes más significativos fue la importancia de una arquitectura de software bien estructurada y la división clara de responsabilidades entre frontend y backend. Esta separación facilitó el trabajo paralelo del equipo, la integración gradual de funcionalidades y el control de calidad. Asimismo, se evidenció el valor de integrar herramientas de diseño (Figma), documentación (Notion) y automatización (LandBot), lo cual permitió ofrecer una experiencia de usuario más completa y profesional. La experiencia adquirida en este proceso será esencial para futuros desarrollos tanto en entornos académicos como laborales.
- Finalmente, el proyecto dejó valiosas lecciones respecto al manejo de tiempos, comunicación efectiva y compromiso del equipo. Las dificultades enfrentadas, como la falta de seguimiento en algunas tareas o la estimación imprecisa de esfuerzos, ayudaron a ajustar y fortalecer los procesos de trabajo. Este proceso de mejora continua, impulsado por la reflexión durante las retrospectivas, demuestra que una buena gestión ágil no solo se enfoca en entregar software funcional, sino en formar equipos más resilientes, organizados y capaces de adaptarse a los desafíos del desarrollo real.

10. Anexos

Anexo A: Estructura de los manuales

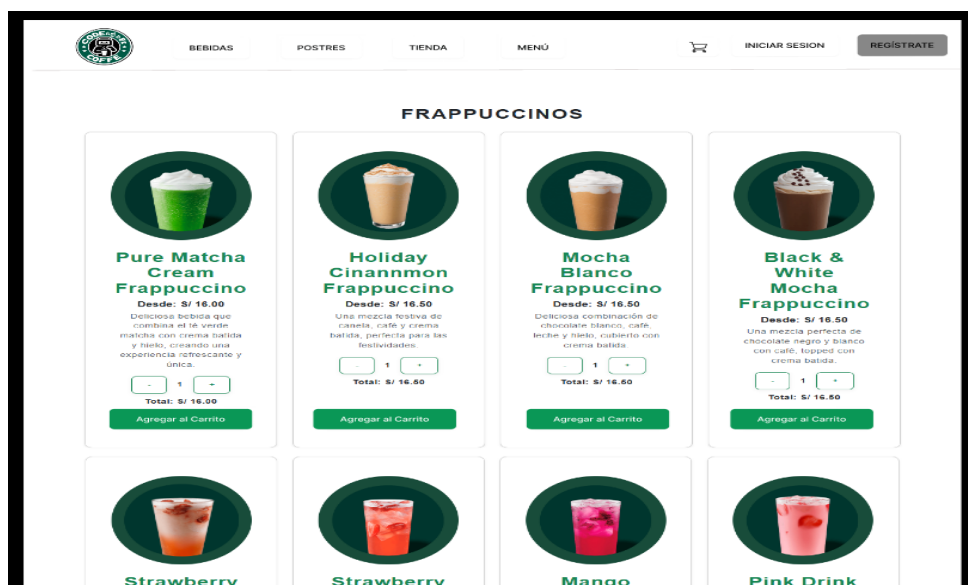
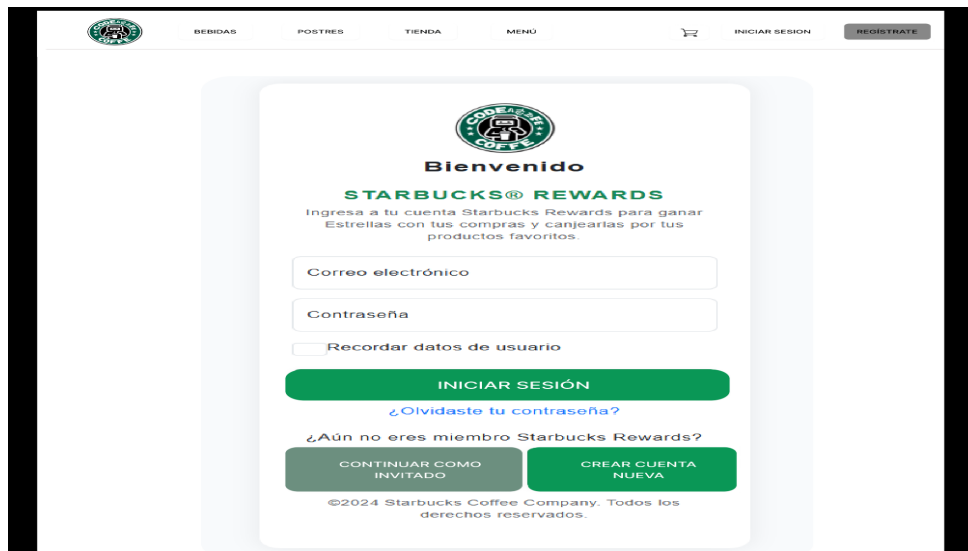
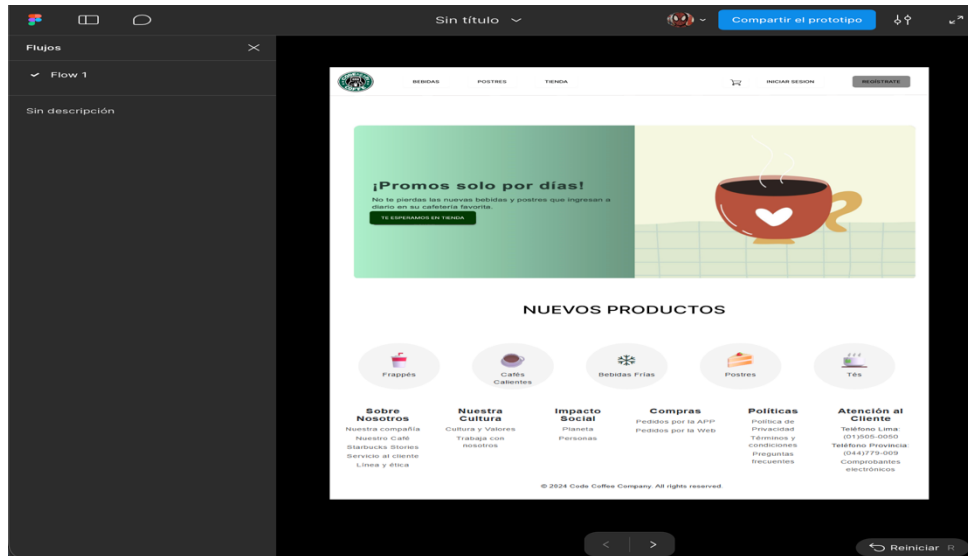
El manual cuenta con una introducción, requisitos del servidor, editor de código, frameworks, pasos de configuración, requisitos del cliente, configuración de funcionalidades y contacto de soporte.



Fuente: Elaboración propia

Anexo B: Estructura de las interfaces

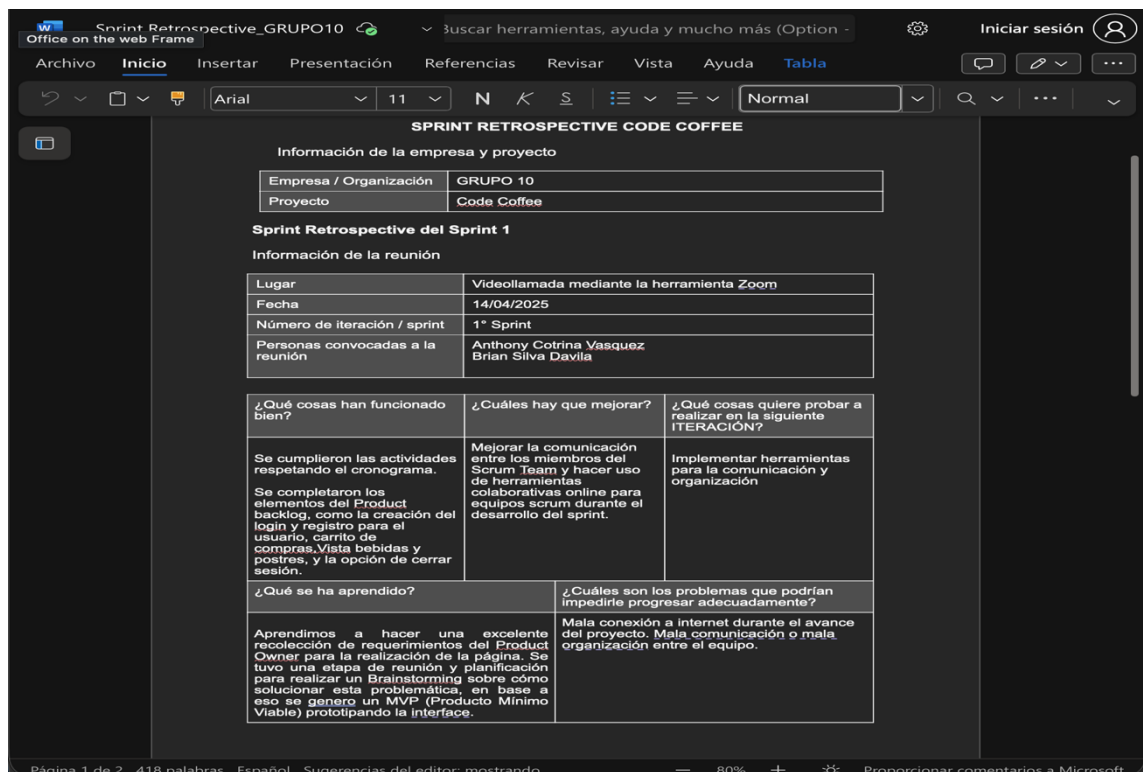
Se diseñó un prototipo de la vista del usuario de cómo será interactuar con la página de la cafetería usando la herramienta de Figma.



Fuente: Elaboración propia

Anexo C: Reportes semanales generados:

Los reportes semanales y Sprint Retrospectives fueron esenciales para el seguimiento del proyecto Code Coffee, permitiendo identificar logros como el cumplimiento del cronograma y la implementación exitosa de funcionalidades core (login, carrito, catálogos), así como áreas de mejora en comunicación del equipo y uso de herramientas colaborativas.



SPRINT RETROSPECTIVE CODE COFFEE

Información de la empresa y proyecto

Empresa / Organización	GRUPO 10
Proyecto	Code Coffee

Sprint Retrospective del Sprint 1

Información de la reunión

Lugar	Videollamada mediante la herramienta Zoom
Fecha	14/04/2025
Número de iteración / sprint	1° Sprint
Personas convocadas a la reunión	Anthony Cotrina Vasquez Brian Silva Davila

¿Qué cosas han funcionado bien?	¿Cuáles hay que mejorar?	¿Qué cosas quiere probar a realizar en la siguiente ITERACIÓN?
Se cumplieron las actividades respetando el cronograma. Se completaron los elementos del Product backlog, como la creación del login y registro para el usuario, carrito de compras, Vista bebidas y postres, y la opción de cerrar sesión.	Mejorar la comunicación entre los miembros del Scrum Team y hacer uso de herramientas colaborativas online para equipos scrum durante el desarrollo del sprint.	Implementar herramientas para la comunicación y organización
¿Qué se ha aprendido?	¿Cuáles son los problemas que podrían impedirle progresar adecuadamente?	
Aprendimos a hacer una excelente recolección de requerimientos del Product Owner para la realización de la página. Se tuvo una etapa de reunión y planificación para realizar un Brainstorming sobre cómo solucionar esta problemática, en base a eso se generó un MVP (Producto Mínimo Viable) prototipando la interface.	Mala conexión a internet durante el avance del proyecto. Mala comunicación o mala organización entre el equipo.	

Anexo D: Comparación con otras herramientas

Se realizó una tabla de comparación entre 3 herramientas que conocemos (ClickUp, Notion y Slack), para aplicar el modelo SCRUM para la organización de nuestro proyecto, tal tabla lo podemos observar a continuación:

Criterio	ClickUp	Notion	Slack
Gestión de tareas	★★★★★	★★★★	★★
Comunicación	★★★★	★★	★★★★★★
Documentación	★★★★	★★★★★★	★★
Integración SCRUM	★★★★★	★★★★	★★
Curva de aprendizaje	Media	Baja	Baja
Costo	Freemium	Freemium	Freemium
Colaboración tiempo real	★★★★★	★★★★	★★★★★★
Reportes y dashboards	★★★★★	★★	★
Kanban boards	★★★★★	★★★★	★
Sprint planning	★★★★★	★★	★

La comparación entre herramientas reveló que ClickUp ofreció la mejor integración para metodologías ágiles SCRUM, destacando en gestión de tareas, reportes automatizados y funcionalidades específicas como Kanban boards y sprint planning. Aunque Slack sobresalió en comunicación instantánea y Notion en documentación colaborativa, ClickUp proporcionó el ecosistema más completo para coordinar sprints, trackear el progreso del equipo y generar métricas de velocidad, convirtiéndose en la herramienta principal que permitió al equipo mantener la estructura organizacional y el seguimiento efectivo durante los 8 semanas de desarrollo del proyecto Code Coffee.