Estrategia de Pruebas

1. Aplicación Bajo Pruebas

1.1. **Nombre Aplicación:** Sistema de apoyo CCP
1.2. **Versión:** 2.0
1.3. **Descripción**:

La aplicación “Sistema de apoyo CCP” es una aplicación hecha a la medida para la Compañía Comercializadora de Productos (CCP). Esta aplicación se hizo con el objetivo de reducir las pérdidas económicas que estaba teniendo la empresa debido a procesos manuales y descentralizados.

La aplicación tiene como usuarios a los empleados de CCP de las áreas de ventas, logística, compras y también tiene como usuarios a los clientes de CCP.

A los usuarios de las áreas de ventas la aplicación les permite revisar los productos que pueden vender y almacenar además de almacenar información comercial en los sistemas de CCP. A los empleados del área de logística les permite programar rutas de entrega de forma óptima además de monitorear productos y camiones en la medida que van entrando y saliendo de las bodegas de CCP. En cuanto a los usuarios del área de compras, la aplicación les permite administrar productos y pedidos. Finalmente, la aplicación ofrece diferentes consultas y reportes de información a los diferentes usuarios.

1.4. **Funcionalidades Core**:

- Registro y autenticación de los diferentes usuarios.
- Funcionalidades Ventas:
    - Consultar clientes
    - Creación de un pedido de cliente
    - Registrar la visita de un cliente
    - Consultar rutas de visita
    - Procesamiento de video tienda de cliente
    - Consulta de reportes sobre los datos de los vendedores
    - Registro de vendedores
    - Creación planes de venta
- Funcionalidades Clientes:
    - Registrar clientes
    - Registrar Pedidos
    - Consultar estado de Pedidos
    - Consultar entregas programadas
- Funcionalidades logística:
    - Creación rutas de entrega para siguiente día
    - Localización de productos en bodega.
- Funcionalidades área de compras / proveedores
    - Registro individual de fabricantes
    - Registro individual y masivo productos de fabricante

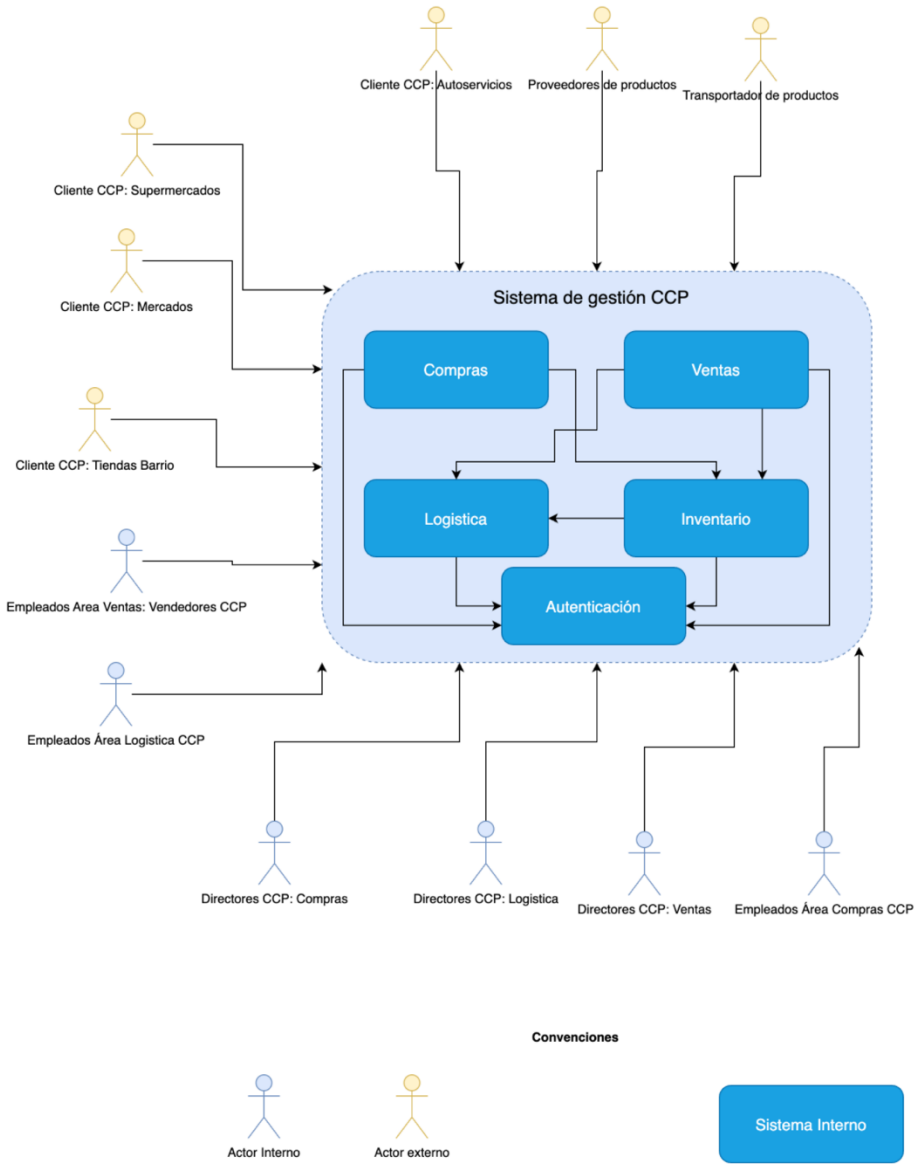
1.5. **Diagrama de Arquitectura:**

Ver README del Proyecto: 2025-proyecto-final-grupo2/README.md at main · DaGamez/2025-proyecto-final-grupo2

1.6. Diagrama de Contexto:

1.7. Modelo de Datos:

2. Ver README del Proyecto: 2025-proecto-final-grupo2/README.md at main · DaGamez/2025-proyecto-final-grupo2

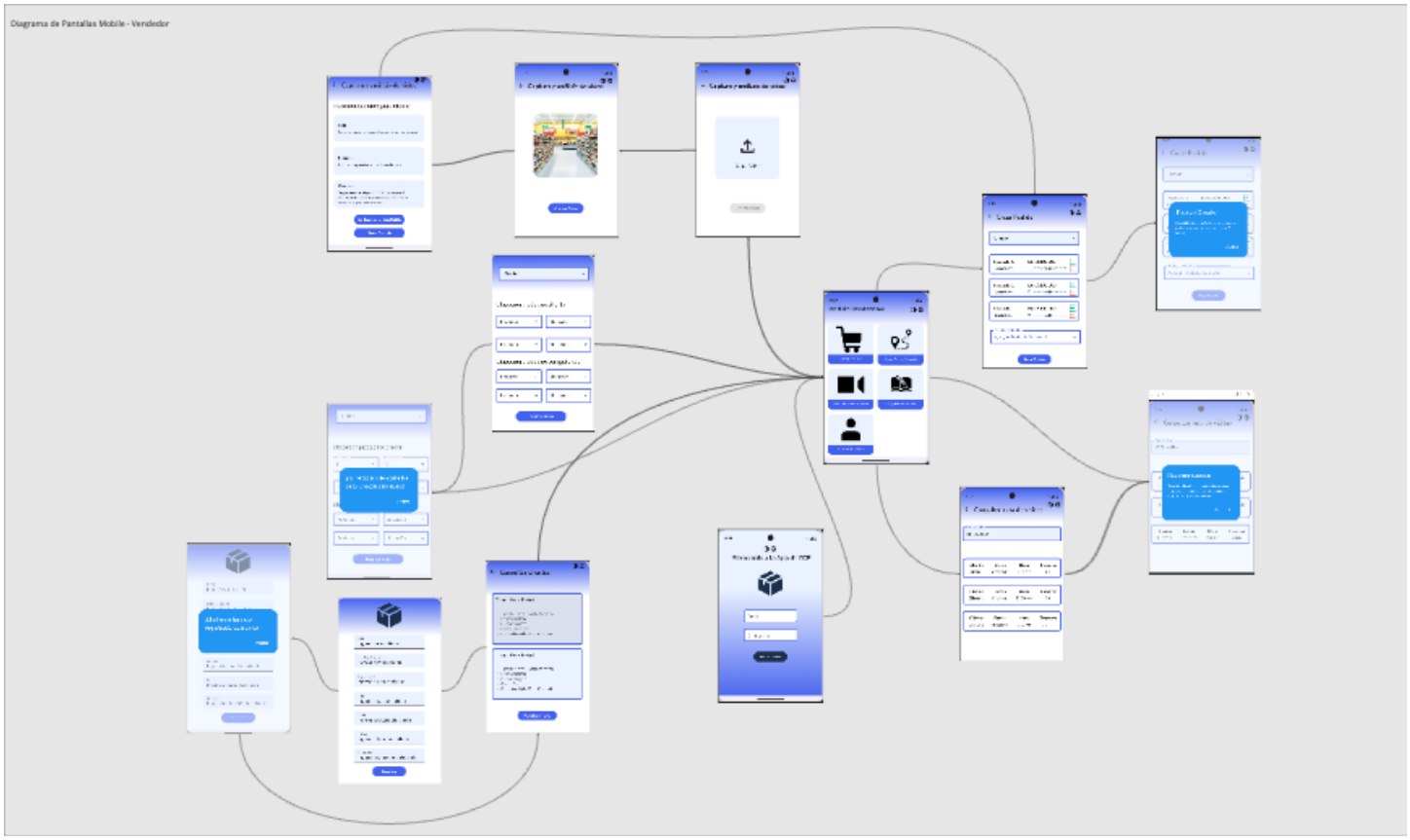


2.1. Modelo de GUI:

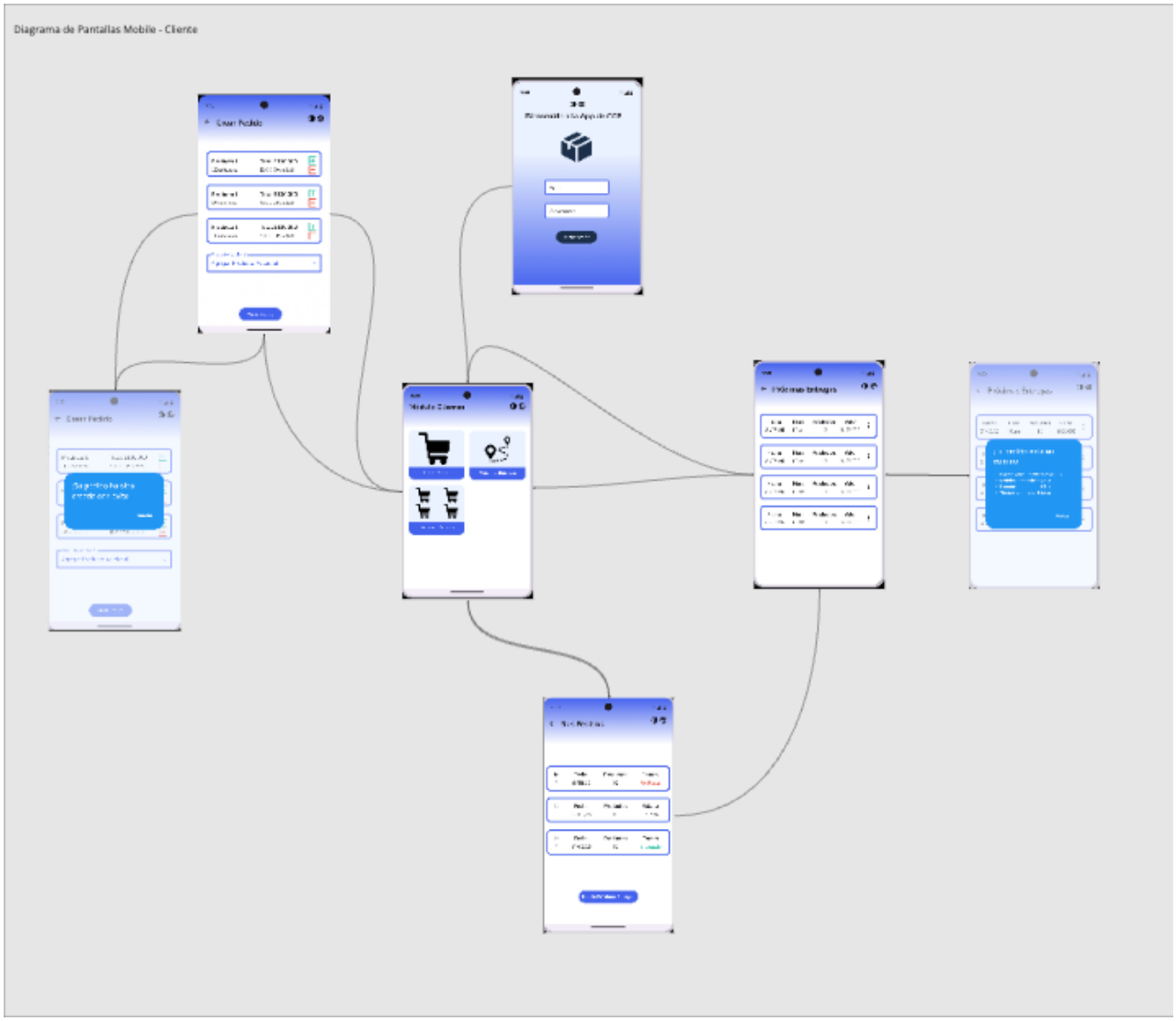
A continuación se presenta el modelo de navegación para dispositivos mobile, tanto del área de vendedores como del área de clientes, y para dispositivos web

Modelo GUI móvil:

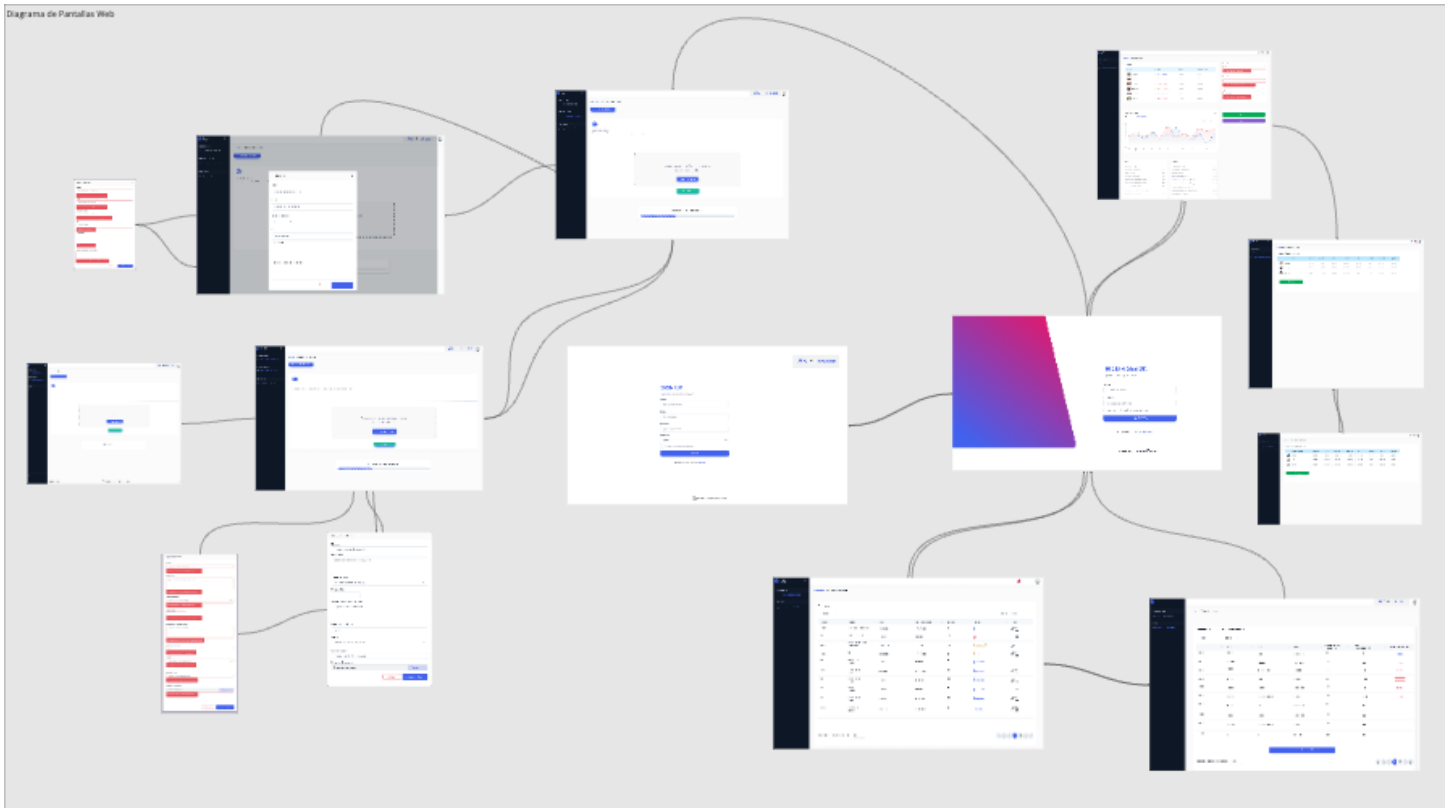
Área Vendedor:



Área Cliente:



Modelo GUI Web:



3. Contexto de la estrategia de pruebas

3.1. Objetivos:

El objetivo principal es garantizar que el sistema a desarrollar cumpla con los requisitos funcionales y no funcionales acordados con el cliente. Para ello se van a realizar pruebas tanto funcionales como no funcionales:

1. Pruebas funcionales
    a. Funcionalidades Módulo Ventas
    b. Funcionalidades Módulo Clientes
    c. Funcionalidades Módulo Área de Compras
    d. Funcionalidades Módulo Logística
2. Pruebas no funcionales y/o de atributos de calidad
    a. Seguridad
    b. Desempeño
    c. Disponibilidad
    d. Escalabilidad
    e. Facilidad de modificación
    f. Pruebas de internacionalización

Y en cuanto a las pruebas funcionales se van a desarrollar en los frentes:

1. Pruebas unitarias

2. Pruebas de integración
3. Pruebas E2E o de sistema
4. Pruebas de carga y estrés

Haciendo un uso equilibrado entre pruebas automatizadas y manuales. De esta forma creemos que se va a brindar una buena revisión de la calidad del producto a entregar.

3.2. Duración de la iteración de pruebas:

Las pruebas van a ejecutarse en paralelo al desarrollo en los 3 sprints planeados y se van a desarrollar por sprint por lo que la duración de las tares de pruebas van a ser 7 semanas comenzando en la semana 9 y terminando en la semana 15:

	Semana 9	Semana 10	Semana 11	Semana 12	Semana 13	Semana 14	Semana 15
Sprint 1							
1.1. Pruebas Funcionales:							
1.1.1 Pruebas Unitarias							
1.1.2. Pruebas Integración							
1.1.3. Pruebas E2E							
1.1.3.1. E2E Flujo Cliente P1							
1.1.3.2. E2E Flujo Vendedor P1							
Sprint 2							
2.1. Pruebas Funcionales:							
2.1.1 Pruebas Unitarias							
2.1.2. Pruebas Integración							
2.1.2.1. P.Integ Proveed y Prod							
2.1.3. Pruebas E2E							
2.1.3.1. E2E Flujo Cliente P2							
2.1.3.2. E2E Flujo Vendedor P2							
2.2 Pruebas no funcionales							
2.2.1. Pruebas de escalabilidad							
2.2.1.1 ASR Scrum 51 P1							
2.2.1.2 ASR Scrum 31 P1							
2.2.2. Pruebas de disponibilidad							
2.2.2.1 ASR Scrum 47 P1							
2.2.2.2 ASR Scrum 30 P1							
Sprint 3							
3.1. Pruebas Funcionales:							
3.1.1 Pruebas Unitarias							
3.1.2. Pruebas Integración							
3.1.2.1. P.Integ Funciones Logistica							
3.1.3. Pruebas E2E							
3.1.3.1. E2E Flujo Cliente P3							
3.1.3.2. E2E Flujo Vendedor P3							
3.2 Pruebas no funcionales							
3.2.1. Pruebas de carga y estres							
3.2.1. Pruebas de escalabilidad							
3.2.1.1 ASR Scrum 51 P2							
3.2.1.2 ASR Scrum 31 P2							
3.2.2. Pruebas de disponibilidad							
3.2.2.1 ASR Scrum 47 P2							
3.2.2.2 ASR Scrum 30 P2							

3.3. Presupuesto de pruebas:

3.3.1. Recursos Humanos

Se cuenta con 4 ingenieros de software para el desarrollo de la estrategia de pruebas. Estos ingenieros tienen conocimientos en pruebas de tipo manuales y automatizadas y en diseño y desarrollo de pruebas tanto funcionales como no funcionales. Se estima una disponibilidad de 2 horas a la semana por ingeniero para el diseño y ejecución de las pruebas.

3.3.2. Recursos Computacionales

Se cuenta con dos equipos portátil con sistema operativo Windows y dos equipos portátiles con sistemas operativos MacOS. Estos 4 equipos son los equipos de pruebas locales de los ingenieros y en caso de requerir ambientes de pruebas uniformes se utilizarían herramientas tipo Docker, maquinas virtuales o conexiones remotas a Gitpod. Se van a realizar pruebas sobre dispositivos Android que son los que se cuentan.

Adicionalmente se va a contar con capacidad computacional en la nube en las plataformas de Github, AWS y GCP. El costo de computación en nube no debe superar los USD 100 en las 7 semanas de ejecución del plan de pruebas.

3.3.3. Recursos Económicos para la contratación de servicios/personal:

No se cuenta con recursos económicos para contratar horas adicionales a los 4 ingenieros anteriormente mencionados.

3.4. TNT (Técnicas, Niveles y Tipos) de pruebas:

Técnica	Nivel	Tipo	Objetivo
Pruebas automatizadas	Sistema/E2E	• Funcionales y no Funcional • Positivas y negativas • Caja negra	Verificar que el sistema como un todo cumpla con los flujos funcionales esperados por los usuarios en las historias de usuario.
Pruebas automatizadas	Sistema/E2E	• No Funcionales • Positivas y negativas • Caja negra	Verificar que el sistema cumpla con los requisitos de calidad definidos.
Pruebas automatizadas	Sistema/E2E	• Internacionalización • Positivas y negativas • Caja negra	Verificar que el sistema cumpla con los requerimientos de internacionalización.
Pruebas automatizadas	Integración	• Funcionales • Positivas y negativas • Caja blanca	Revisar que las funcionalidades trasversales a los diferentes módulos (ventas, proveedores/área compras, clientes y logística) estén funcionando de acuerdo con lo definido en las historias de usuario en su **comportamiento conjunto entre** pares o grupos de módulos sin contemplar aún el sistema como un todo.
Pruebas automatizadas	Unitarias	• Funcionales • Positivas y negativas Caja blanca	Revisar que las funcionalidades más específicas de los diferentes módulos (ventas, proveedores/área compras, clientes y logística) estén funcionando de acuerdo a lo definido en las

			historias de usuario en su **comportamiento individual**.
Pruebas manuales	Sistema	• Funcionales • Positivas y negativas • Caja negra	Identificar problemas funcionales en el sistema que posiblemente no se estén contemplando en las pruebas automatizadas.
Pruebas manuales	Sistema	• No funcionales • Positivas y negativas • Caja negra	Identificar problemas no funcionales en el sistema que posiblemente no se estén contemplando en las pruebas automatizadas.

3.5. Distribución de Esfuerzo

A continuación, describimos la distribución de esfuerzo que se planea utilizar durante esta etapa de pruebas. Identificamos los momentos en los que se planea ejecutar esta estrategia. Se describe la asignación de recursos del presupuesto de pruebas y tiempo a las actividades mencionadas en 2.4. De igual forma se espera detallar más la distribución de esfuerzo al inicio de cada sprint:

Descripción del Hito	Asignado a	Inicio	Días	Herramientas
Sprint 1				
Pruebas unitarias para revisión de requisitos funcionales en sprint 1	4 Ingenieros	Semana 9	10 (2 semanas)	Suites de pruebas unitarias, por ejemplo: • Angular ng test • Python pytest
Pruebas de integración para revisión de requisitos funcionales en sprint 1	4 Ingenieros	Semana 9	10 (2 semanas)	• Docker • Maquinas virtuales • Github • GCP
Pruebas de sistema/E2E para revisión de requisitos funcionales en sprint 1	4 Ingenieros	Semana 9	10 (2 semanas)	• Docker • Maquinas virtuales • Github GCP
Pruebas no funcionales sprint 1	4 Ingenieros	Semana 9	10 (2 semanas)	• Docker

				• Maquinas virtuales • Github GCP
Sprint 2				
Pruebas unitarias para revisión de requisitos funcionales en sprint 2	4 Ingenieros	Semana 11	10 (2 semanas)	Suits de pruebas unitarias, por ejemplo: • Angular ng test Python pytest
Pruebas de integración para revisión de requisitos funcionales en sprint 2	4 Ingenieros	Semana 11	10 (2 semanas)	• Docker • Maquinas virtuales • Github GCP
Pruebas de sistema/E2E para revisión de requisitos funcionales en sprint 2	4 Ingenieros	Semana 11	10 (2 semanas)	• Docker • Maquinas virtuales • Github GCP
Pruebas no funcionales sprint 2	4 Ingenieros	Semana 11	10 (2 semanas)	• Docker • Maquinas virtuales • Github GCP
Sprint 3				
Pruebas unitarias para revisión de requisitos funcionales en sprint 3	4 Ingenieros	Semana 13	15 (3 semanas)	Suits de pruebas unitarias, por ejemplo: • Angular ng test • Python pytest
Pruebas de integración para revisión de requisitos funcionales en sprint 3	4 Ingenieros	Semana 13	15 (3 semanas)	• Docker • Maquinas virtuales

				• Github GCP
Pruebas de sistema/E2E para revisión de requisitos funcionales en sprint 3	4 Ingenieros	Semana 13	15 (3 semanas)	• Docker • Maquinas virtuales • Github GCP
Pruebas no funcionales sprint 3	4 Ingenieros	Semana 13	15 (3 semanas)	• Docker • Maquinas virtuales • Github GCP

Definición de Framework para pruebas

Framework para el componente web

- Framework: Angular
- Unit testing: TestBed
- Integration Testing: Jasmine
- End to end: Cypress

Framework para el componente móvil

- Framework: Jetpack Compose
- Lenguaje: Kotlin
- Unit testing: JUnit
- Integration Testing: Espresso
- End to end: Espresso

Ambiente cloud para backend

- Nube: Google Cloud
- Lenguaje Programación: Python

Herramientas para test para el Backend

- Lenguaje: Python
- Unit testing: Pytest