Estrategia de Pruebas

1. Aplicación Bajo Pruebas

1.1. **Nombre Aplicación:** Vinyls
1.2. **Versión:** Develop 1
1.3. **Descripción**:

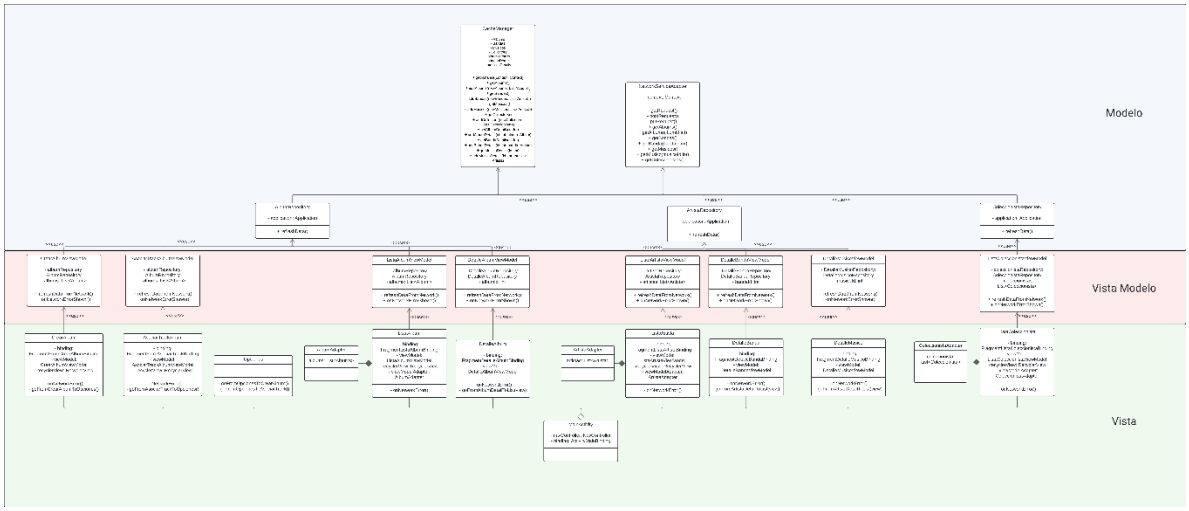
Vinyls es una aplicación diseñada para coleccionistas de música que desean organizar, crear y conocer colecciones de música. Estas colecciones son repositorios de álbumes, canciones y artistas que contienen la información de cada uno.

1.4. **Funcionalidades Core**:

- Gestión de álbumes: Una sección que permite agregar y visualizar álbumes en el repositorio.
- Gestión de artistas: Una sección que permite agregar y visualizar artistas en el repositorio.
- Gestión de colecciones: Una sección que permite agregar y visualizar colecciones musicales en el repositorio. Estas colecciones contienen álbumes, artistas y canciones
- Busqueda de elementos: Permitir al usuario buscar álbumes, artistas, canciones y colecciones que ya se encuentran en el servicio
- Relacionar elementos: El usuario puede crear asociaciones entre los álbumes, artistas y canciones para tener un mejor control de la información, al igual que asociar estos elementos a diferentes colecciones

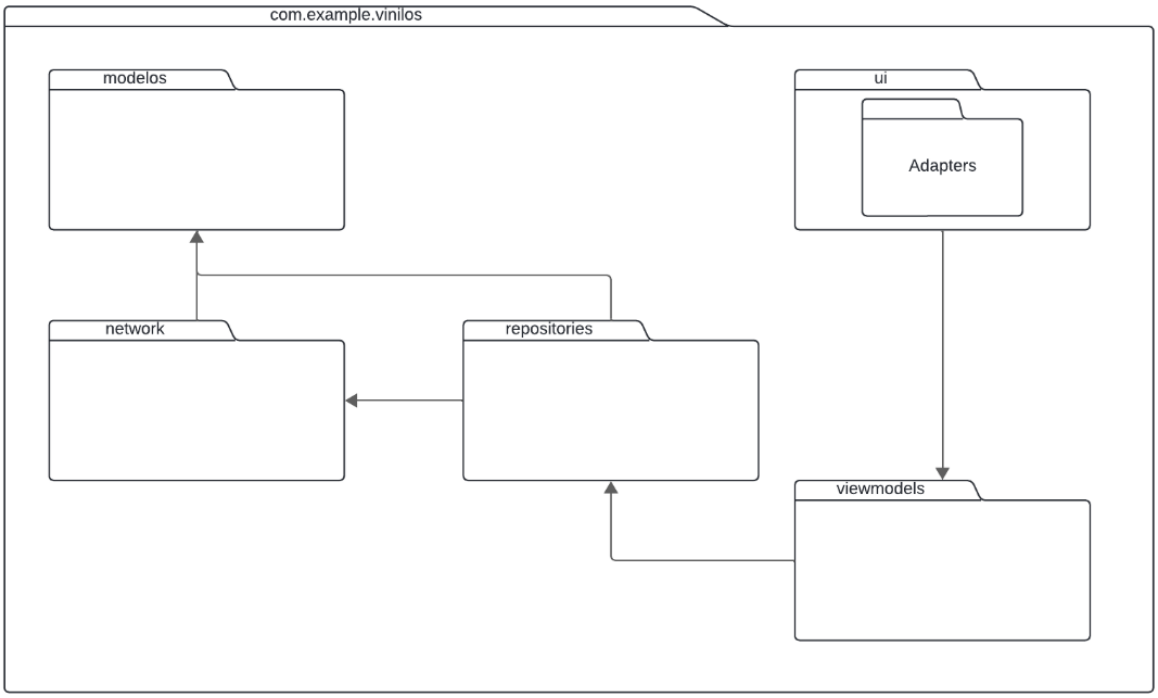
1.5. **Modelo de Datos:**

Los modelos construidos se pueden visualizar en el siguiente enlace: https://lucid.app/lucidchart/78982b77-2bc1-4022-8a4f-0699cd562113/edit?viewport_loc=-115%2C-547%2C5644%2C2632%2Cf7Ww5lQIr17v&invitationId=inv_ecf5ff0f-6c5c-4733-a3c4-e839afded220



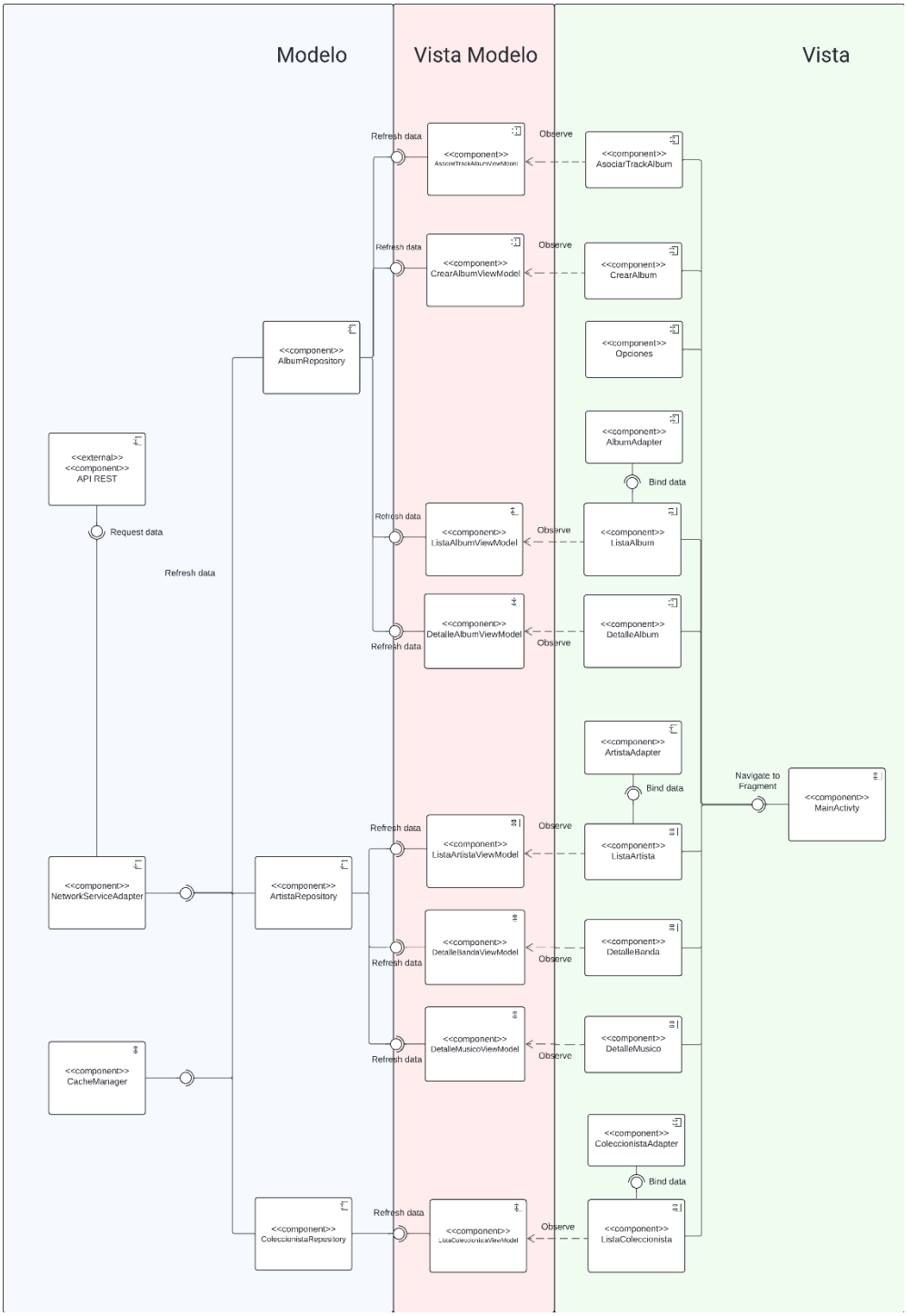
1.6. **Modelo de Paquetes:**

Los modelos construidos se pueden visualizar en el siguiente enlace:
https://lucid.app/lucidchart/78982b77-2bc1-4022-8a4f-0699cd562113/edit?viewport_loc=-115%2C-547%2C5644%2C2632%2Cf7Ww5lQIr17v&invitationId=inv_ecf5ff0f-6c5c-4733-a3c4-e839afded220



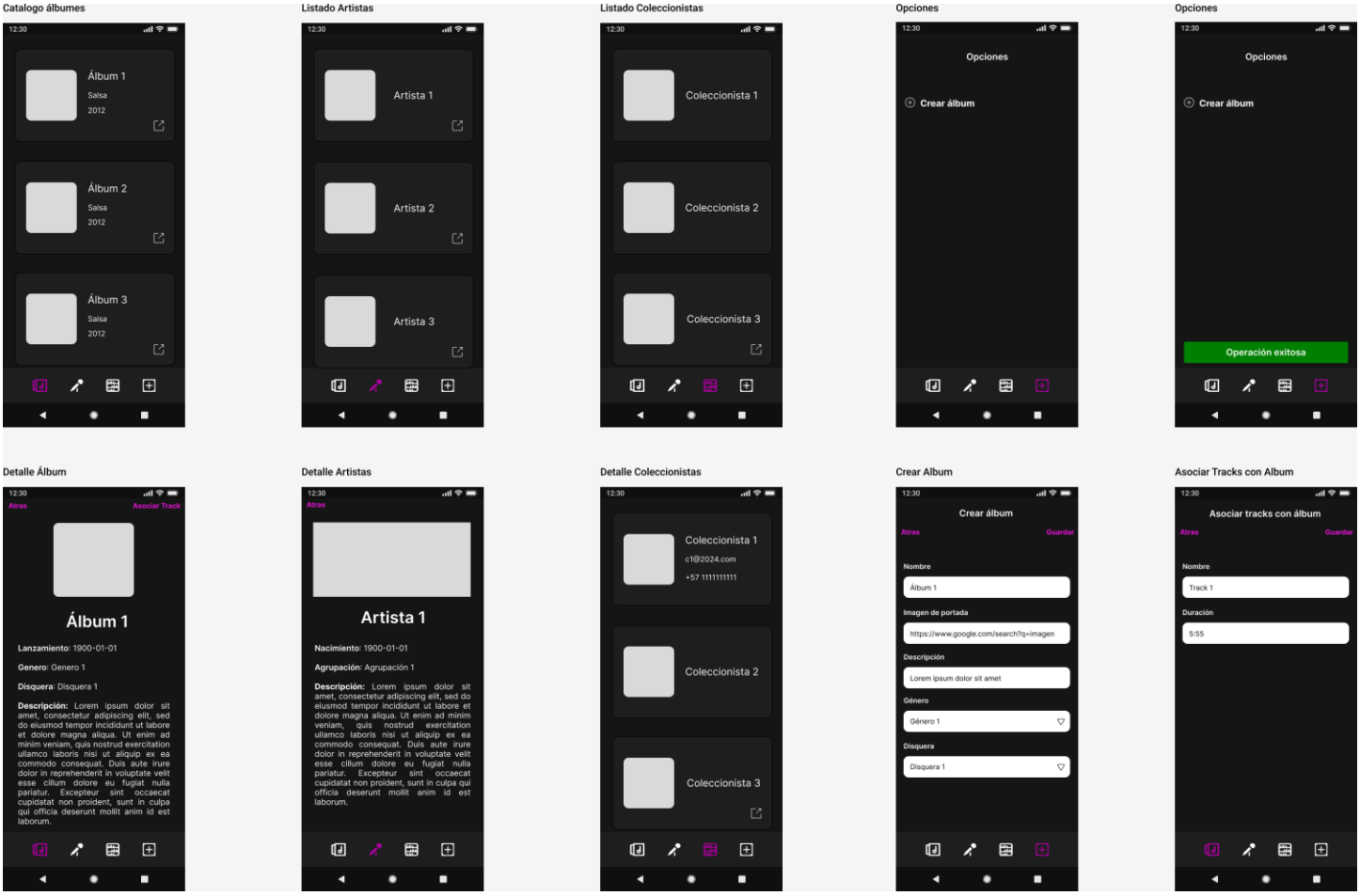
1.7. Modelo de Componentes:

Los modelos construidos se pueden visualizar en el siguiente enlace:
https://lucid.app/lucidchart/78982b77-2bc1-4022-8a4f-0699cd562113/edit?viewport_loc=-115%2C-547%2C5644%2C2632%2Cf7Ww5lQIr17v&invitationId=inv_ecf5ff0f-6c5c-4733-a3c4-e839afded220



1.8. Modelo GUI:

El diseño de UI/UX detallado se puede ver en el siguiente enlace: https://www.figma.com/proto/ccoVjGodLwHAOhVlm1RNHt/Proyecto_misw4203?type=design&node-id=104-1538&t=ryetbmZwjy9YwX6P-1&scaling=min-zoom&page-id=104%3A1027&starting-point-node-id=104%3A1538&mode=design



2. Contexto de la estrategia de pruebas

2.1. Objetivos:

1. Identificar fallos visuales en la maquetación de la interfaz de usuario.
2. Explorar la respuesta del sistema ante datos positivos y datos negativos que puedan generar excepciones que bloqueen o terminen la aplicación.
3. Evaluar el rendimiento y la eficiencia de la aplicación mediante pruebas de perfilamiento para identificar áreas de mejora en el uso de recursos y tiempos de respuesta.
4. Mejorar las condiciones de accesibilidad de la aplicación para permitir el acceso a usuario con o sin discapacidades
5. Diagnosticar por medio de pruebas automatizadas la respuesta de la aplicación frente a estímulos aleatorios

2.2. Duración de la iteración de pruebas:

Tiempo iteración: 2 Semanas

2.3. Presupuesto de pruebas:

2.3.1. Recursos Humanos

Para la estrategia de pruebas propuesta se cuenta con 4 ingenieros de software, con alrededor de 8 horas efectivas para el diseño, construcción y ejecución de estas. Estos ingenieros cuentan con experiencia en ejecución de pruebas E2E en ambientes configurados en Node.JS y en la configuración de APIs de automatización escritas en el lenguaje JavaScript.

Recurso	Cantidad	Horas disponibles	Horas planeadas	Costo ($/hr)
Ingeniero de software	4	8	18	$31530 COP

2.3.2. Recursos Computacionales

Para la consecución de las pruebas se requiere de Android Studio para la compilación y carga de las pruebas, al igual que para la creación de los dispositivos de prueba, en este caso emuladores y dispositivos físicos. También se requiere del uso de git para la descarga de los repositorios. Así mismo, Android Studio debe tener configurado el framework “Espresso” para la ejecución de las pruebas

Por el lado del backend de la aplicación, se requiere de un presupuesto para ejecutar 64 horas/máquina en servicios en la nube. Adicionalmente, se agregó un servicio de Cloud SQL para Esto corresponde a un único servicio configurado para ejecutar el API REST del servicio de Vinyls:

Servicio	vCPUs (unidades)	RAM (Gb)	Costo ($/hr)	Horas planeadas	Costo final
GCP Compute Engine	2	1	0.01	64	$0.64 USD
GCP Cloud SQL	1	1.7	0.05	64	$3.2 USD

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

Nivel	Tipo	Técnica	Objetivo
Sistema	Caja negra	E2E (APIs Automatización)	1
Sistema	Mixtas	Pruebas manuales exploratorias	2
Sistema	Caja negra	Pruebas de perfilamiento	3
Sistema	Caja negra	Pruebas de accesibilidad	4
Sistema	Caja negra	Monkey Testing	5

2.5. Distribución de Esfuerzo

Para distribuir el esfuerzo de la estrategia se realizó una distribución aleatoria de todas las actividades de pruebas a realizar para garantizar que la ejecución de las pruebas sea menos sesgada. Esta distribución se puede observar en la figura 1:

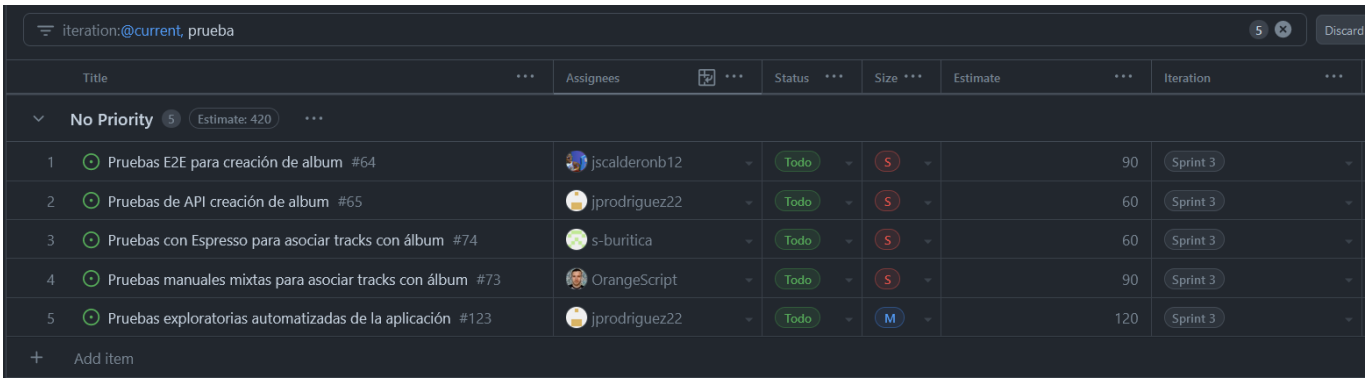


Figura 1 - Distribución de tareas de pruebas