Estrategia de Pruebas

1. Aplicación Bajo Pruebas

1.1. Nombre Aplicación: Vinyls

1.2. Versión: v3.0.0

1.3. Descripción: Consiste en una aplicación móvil nativa de Android, que permite a los melómanos coleccionar sus álbumes favoritos y seguir a otros coleccionistas. Adicional permite la navegación de los álbumes a usuarios visitantes, interesados en conocer el contenido de la aplicación.

1.4. Funcionalidades Core: A continuación, se relacionan las funcionalidades Core de la ABP:

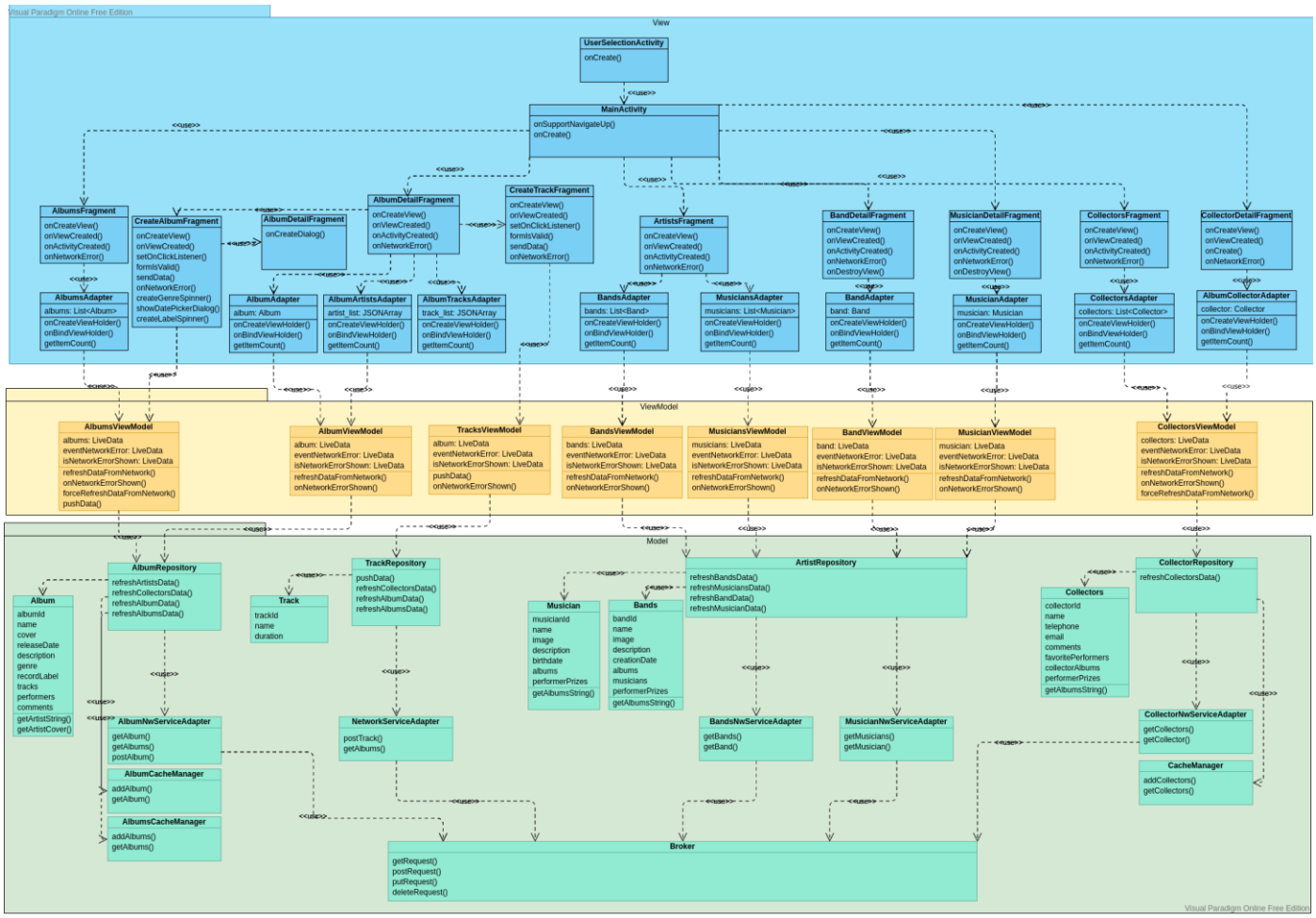
- **Administración de Álbumes:**
  Consultar el catálogo de álbumes.
  Ver información detalle de un álbum seleccionado.
  Crear un álbum.
  Asociar canciones a un álbum.

- **Administración de Artistas:**
  Listado de artistas.
  Ver detalle de un artista.

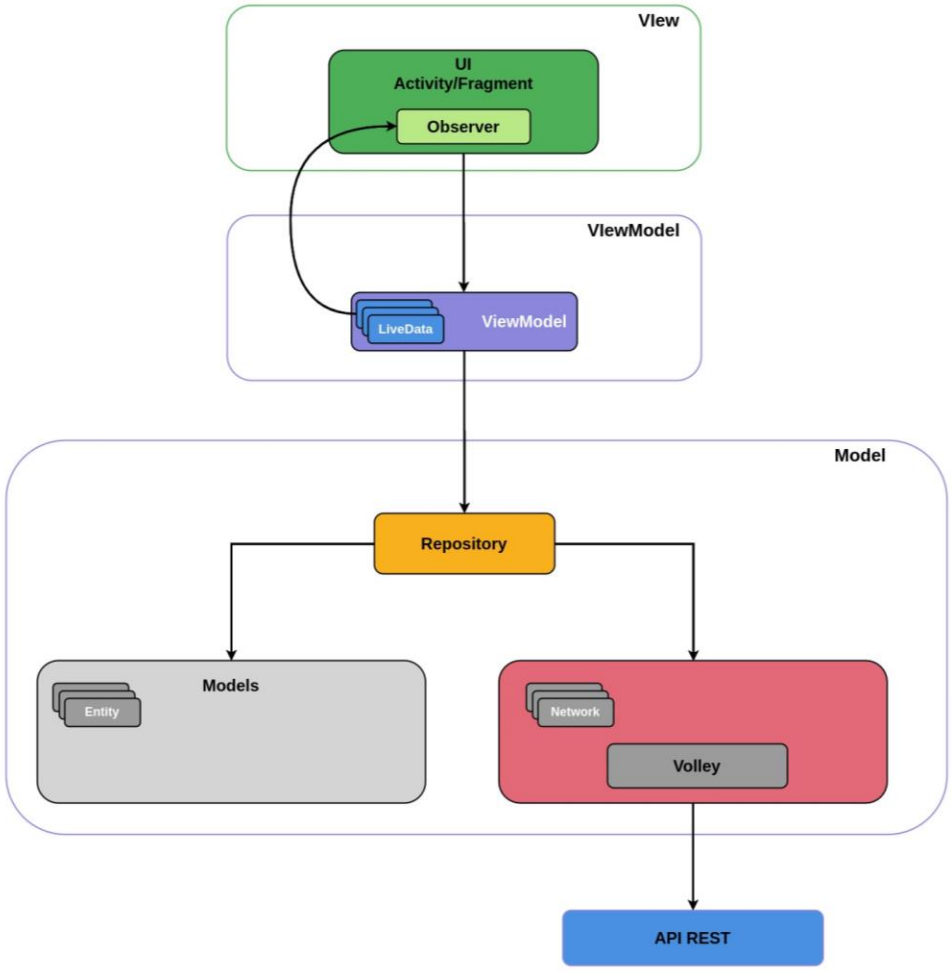
- **Administración de Coleccionistas:**
  Listado de coleccionistas.
  Ver detalle de un coleccionista.

1.5. Diagramas de Arquitectura:

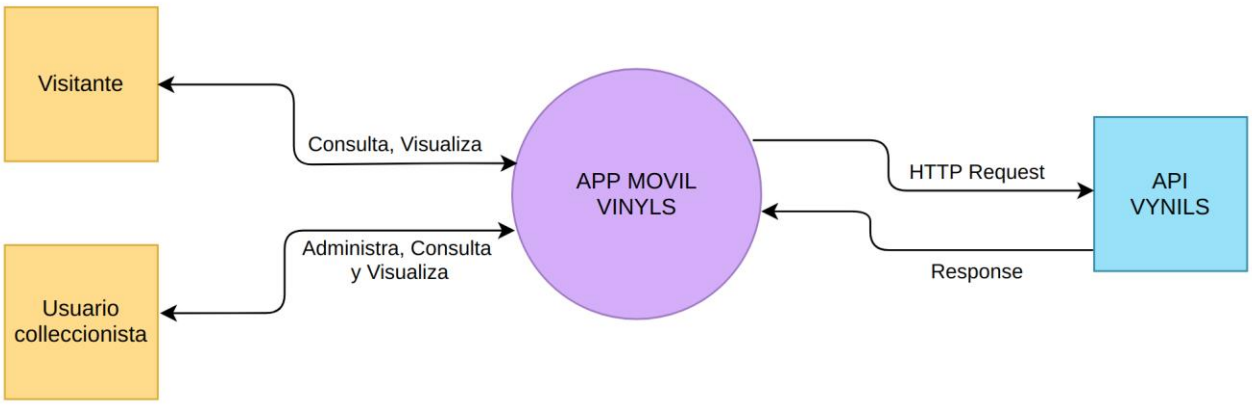
1.5.1. Diseño arquitectónico:



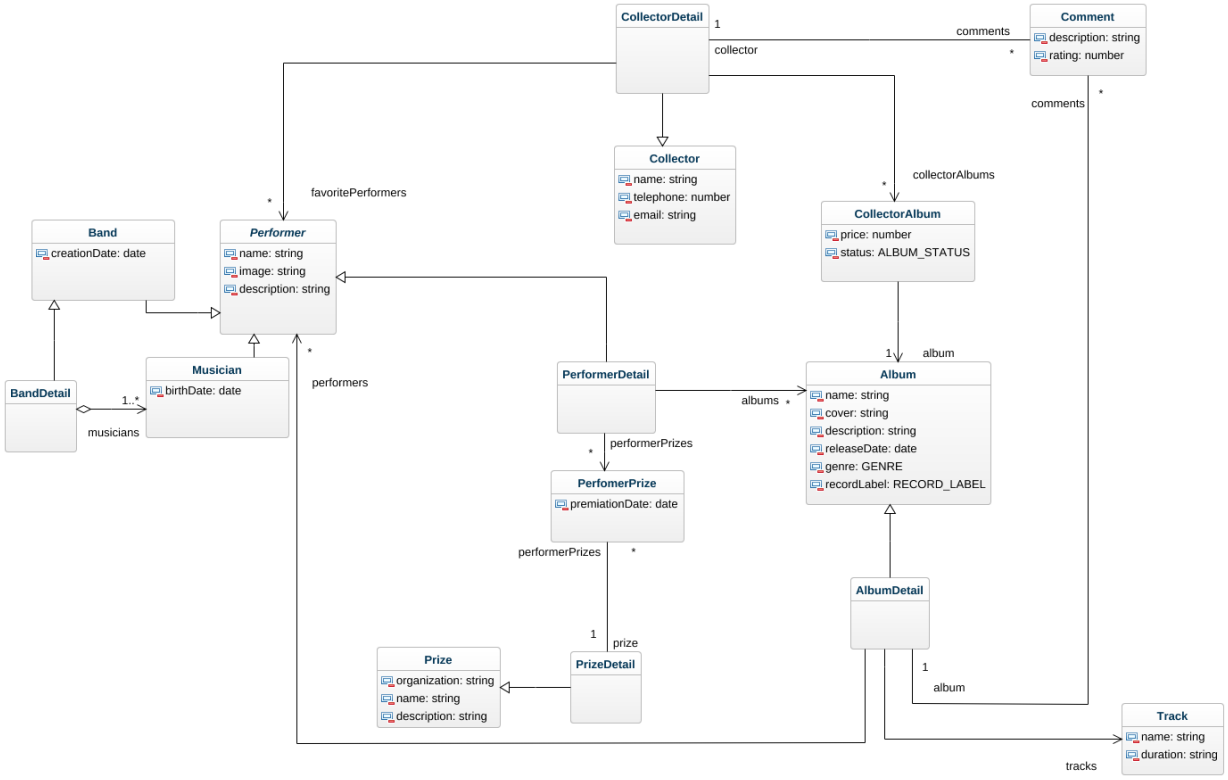
1.5.2. Diagrama de Componentes:



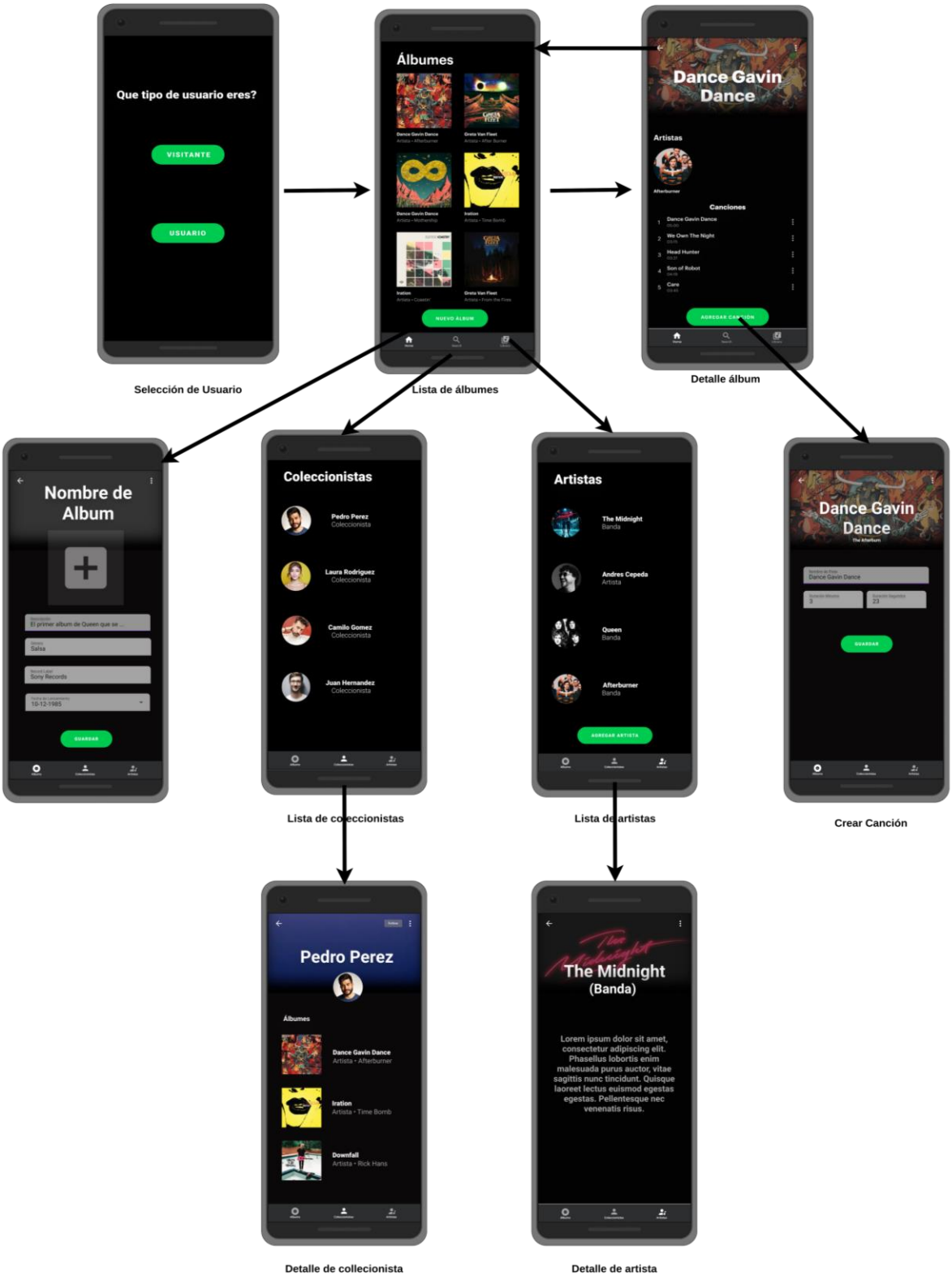
1.5.3. Diagrama de Contexto:



1.5.4. Modelo de Datos:



1.5.5. Modelo de GUI:



2. Contexto de la estrategia de pruebas

2.1. Objetivos:

- Construcción y ejecución de pruebas E2E con Espresso, para las funcionalidades implementadas durante la iteración del Sprint (HU04, HU06 y HU07), correspondientes a consultar información detallada de un artista, consultar información detallada de un coleccionista y crear un álbum.

- Garantizar que la implementación de nuevas funcionalidades no se inserten errores en la aplicación, realizando pruebas de regresión con los scripts ya creados en iteraciones previas.

- Ejecución de pruebas E2E funcionales de aceptación, para las funcionalidades construidas durante la iteración del Sprint, (HU04, HU06 y HU07), correspondientes a consultar información detallada de un artista, consultar información detallada de un coleccionista y crear un álbum. Estas prueban se realizarán de forma manual, haciendo uso de emuladores y dispositivos físicos.

- Realizar análisis de desempeño de la aplicación, a través de ejecución de pruebas de perfilamiento, haciendo uso de herramientas propias de Android Studio como profiler. El alcance de estas pruebas será el de analizar los componentes que tienen mayor consumo de recursos e identificar si es natural al funcionamiento de la aplicación o si es posible realizar una mejora en el performance.

- Realizar pruebas de reconocimiento aleatorias y de exploración sistemática, para todas las funcionalidades implementadas de la aplicación.

- Realizar pruebas de accesibilidad para analizar las sugerencias reportadas por la aplicación y tomar acciones que permitan a la aplicación tener una mayor accesibilidad y garantizar el uso a usuarios con algún tipo de limitación. Para estas pruebas se usara Lint de Android Studio, herramientas externas tales como la aplicación de prueba de accesibilidad (Accessibility Scanner).

2.2. Duración de la iteración de pruebas:

La iteración tendrá una duración de una semana. Con una intensidad horaria de 8 horas distribuidas durante la semana.

2.3. Presupuesto de pruebas:

2.3.1. Recursos Humanos:

Contaremos con un equipo de pruebas para llevar a cabo esta estrategia, que será el mismo equipo que realizo la implementación de las funcionalidades, que corresponde a cuatro Ingenieros de Desarrollo, que cuentan con los conocimientos básicos y con experiencia no mayor a 6 meses en pruebas automatizadas. Los cuatro recursos tendrán una disponibilidad del 100% de asignación para esta iteración.

2.3.2. Recursos Computacionales y Herramientas:

- Cada recurso contará con su pc de dotación.
  - Portátil – Linux Ubuntu 20.04 – 32 GB Ram – Intel Core 7 8th Generación
  - Portátil – Linux Ubuntu 20.04 – 8 GB Ram – 2 CPU
  - Portátil – Linux Ubuntu 20.04 – 8 GB Ram
  - Portátil Macbook Pro

- Emulador de dispositivos móviles de Android Studio con las siguientes versiones.
  - Pixel 3a – API30 x86

- Dispositivos Físicos Android
  - Samsung A21S – Android 10
  - Samsung Galaxy J7 – Android 8.1
  - Xiaomi mi 10t Pro – Android 11

- API del backend desplegada en Heroku.
- BD PostgreSQL

- Usaremos GitHub como repositorio y documentación para las fuentes y los Sripts.
- Android Studio como IDE de desarrollo.
- Lint de Android Studio para análisis de código estático.
- Profile de Android Studio para el análisis de perfilamiento.
- Espresso para las pruebas automatizadas
- Accessibility Scanner para las pruebas de accesibilidad.

2.3.3. Recursos Económicos para la contratación de servicios/personal: Para esta estrategia, no se contemplan recursos económicos destinados a la tercerización de tareas y contratación de servicios externos.

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

Nivel	Tipo	Técnica	Objetivo

Aceptación	Funcionales	Manuales	Ejecutar pruebas de extremo a extremo (E2E) manuales de las funcionalidades de listar y detallar álbumes.
Sistema	E2E	Basadas en API de Automatización	Crear los scripts de pruebas haciendo usando de Espresso para automatizar las funcionalidades de listar y detallar álbumes.
Aceptación	Caja negra	Manuales	Ejecución de las pruebas funcionales en diferentes tamaños de dispositivos y versiones de Android. Pruebas se harán para validar y flujos y criterios de aceptación de Historia de Usuario.
Sistema	E2E	Perfilamiento	Identificar mejoras de rendimiento y uso de los recursos de la aplicación, haciendo uso de la herramienta Profiler de Android Studio.
Sistema	Caja negra	Monkey Testing	Realizar pruebas de reconocimiento aleatorias y de exploración sistemática, para todas las funcionalidades implementadas de la aplicación.
Sistema	No funcionales	Accesibilidad	Identificar la accesibilidad de la aplicación realizando análisis de accesibilidad con herramientas tales como Lint y Accessibility Scanner.

2.5. Distribución de Esfuerzo: La distribución de esfuerzo que se busca con esta estrategia. es orientar la ABP al patrón de la pirámide de Testing. En donde la distribución inicialmente seria:

- 50% de pruebas Automatizadas.
- 50% de pruebas manuales.

Actividad	Horas/Hombre	Horas/Maquina
Construcción del script de pruebas para la funcionalidad ver detalle de un artista. Usando Espresso	2 Horas / Recurso 1	0
Construcción del script de pruebas para la funcionalidad de ver detalle de un coleccionista. Usando Espresso	2 Horas/ Recurso 2	0

Construcción del script de pruebas para la funcionalidad crear un álbum. Usando Espresso	2 Horas/ Recurso 3	0
Análisis de perfilamiento de la aplicación haciendo uso de herramientas como profiler.	2 Horas/ Recurso 4	0
Pruebas de regresión de la aplicación haciendo uso de los scripts generados en las anteriores iteraciones.	1 Hora / Todos	0
Ejecución de los scripts generados por la herramienta en diferentes versiones de Android.	1 Hora / Todos	0
Ejecución de pruebas funcionales en dispositivos físicos con la APK generado.	1 Hora / Todos	0
Realizar pruebas de reconocimiento aleatorias y de exploración sistemática, para todas las funcionalidades implementadas de la aplicación.	1 Hora / Todos	0
Realizar pruebas de accesibilidad de la aplicación usando Lint y Accessibility Scanner	1 Hora / Todos	0
Reporte de incidencias encontradas durante la ejecución de las pruebas, en la sección de issues de Github.	1 Hora / Todos	0