Mauricio Londoño Puche T00049903 **Romario Marimon Romero T00049321**
Luis Carlos Oviedo Talaigua T00046275 **Maria Fernanda Campo T00049516**

Documentación
PokeBattle War

Imagen 1: Pantalla principal de la APP - Prototipo

Descripción de la App:

PokeBattleWar es una App móvil desarrollada para Android, dicha aplicación fue construida en el entorno de desarrollo de Android Studio, y con el lenguaje de programación Kotlin, además utilizamos la plataforma de desarrollo colaborativo de GitHub para alojar nuestro proyecto. El propósito de su realización es aplicar todos los conocimientos adquiridos en el curso de ingeniería de Software, por otra parte se desea poder consumir los datos de una API llamada PokéAPI, y con éstos poder recrear el popular juego de batalla Pokémon .

Objetivos

- Escoger una Plataforma que nos ayude a almacenar el código fuente del proyecto.
- Escoger una herramienta que nos permita organizar y gestionar el proyecto.
- Hacer La captura de requerimientos para la construcción del software.
- Hacer un modelado del funcionamiento de la App.

Captura de Requerimientos

1) Requerimientos Funcionales

1.1) Requerimientos de usuario:

- El usuario (el jugador) creará un nickname.
- Al iniciar el juego solo podrá escoger entre tres pokemones (Pikachu - Squirtle - Charmander), y podrá ver las estadísticas de estos.
- El Usuario podrá escoger entre dos modos de juego (Historia y versus), en el primer modo este podrá escoger su pokémon aliado, y su oponente será escogido de forma aleatoria, mientras que en el segundo modo el usuario tiene la opción escoger su pokémon aliado y su oponente o que tanto el oponente como el aliado sean escogidos de forma aleatoria.
- Los enfrentamientos se harán por un sistema de turnos, por lo cual el usuario no podrá efectuar un movimiento, si no es su respectivo turno.

- El usuario será ganador de la contienda sólo cuando la vida de su oponente llegue a cero.
- El usuario puede elegir el tiempo por enfrentamiento.

1.2) Requerimientos del Sistema:

- El sistema tomará la información acerca los pokemones de la API (PokéAPI)
- El sistema le restara a la vida de los oponentes dependiendo del daño recibido por cada ataque.
- El sistema dará como ganador a quien logre reducir la vida de su oponente a cero.

2) Requerimientos no funcionales

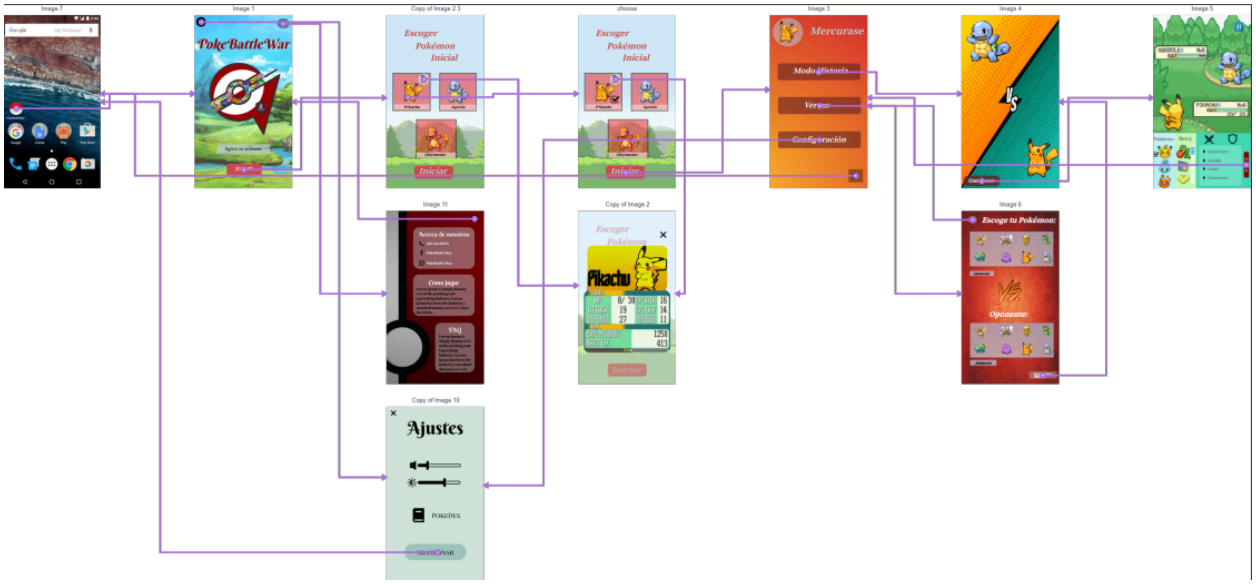
Eficiencia

- El juego debe ser capaz de procesar la información de los pokemones en un tiempo máximo de 10 segundos.

Usabilidad

- El juego debe contar con alguna información de contacto donde los jugadores puedan comunicarse en caso de tener alguna inquietud (FAQ)

Vista de interacción entre las pantallas de prototipo hecho en Marvel



Modelado de Contexto

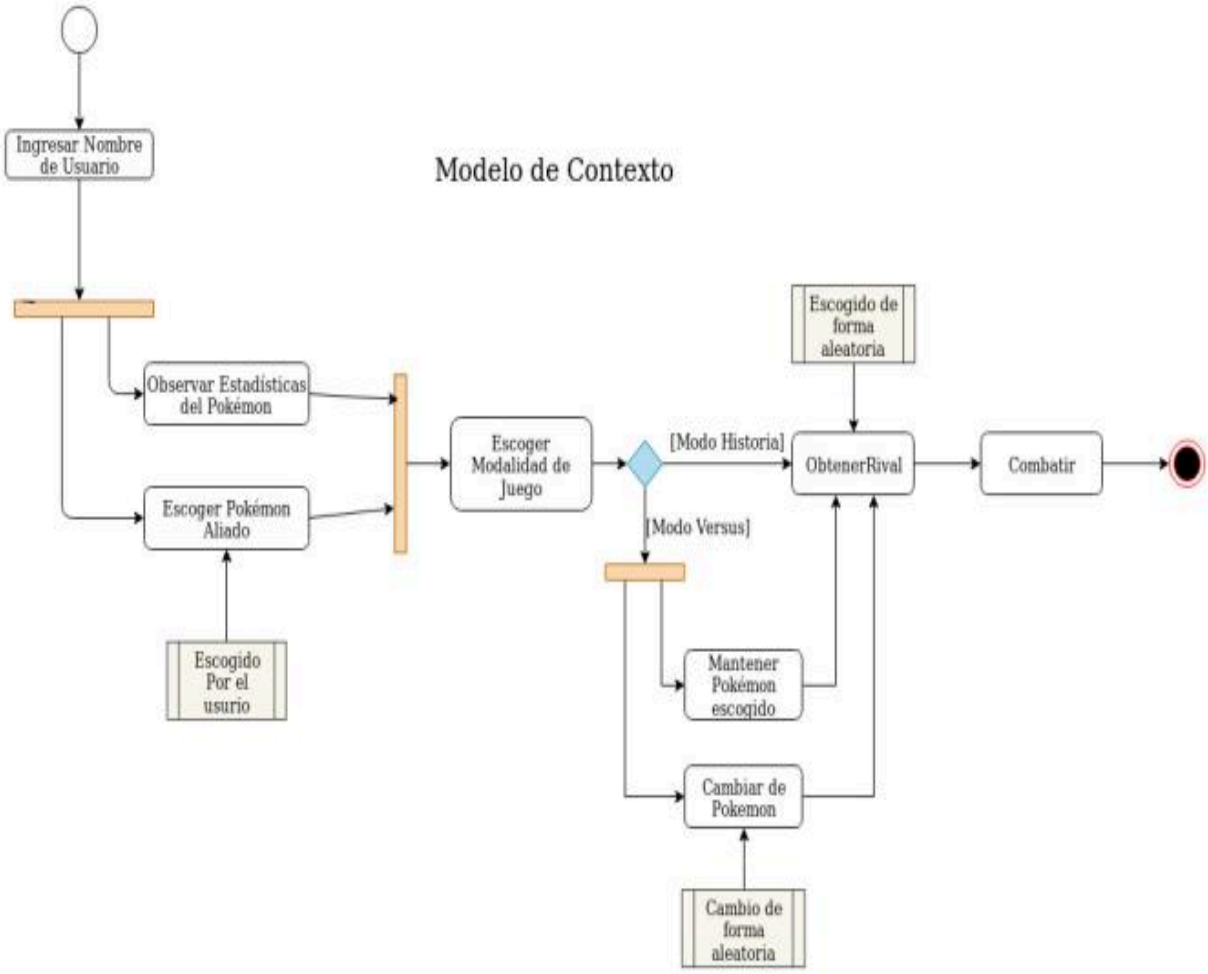
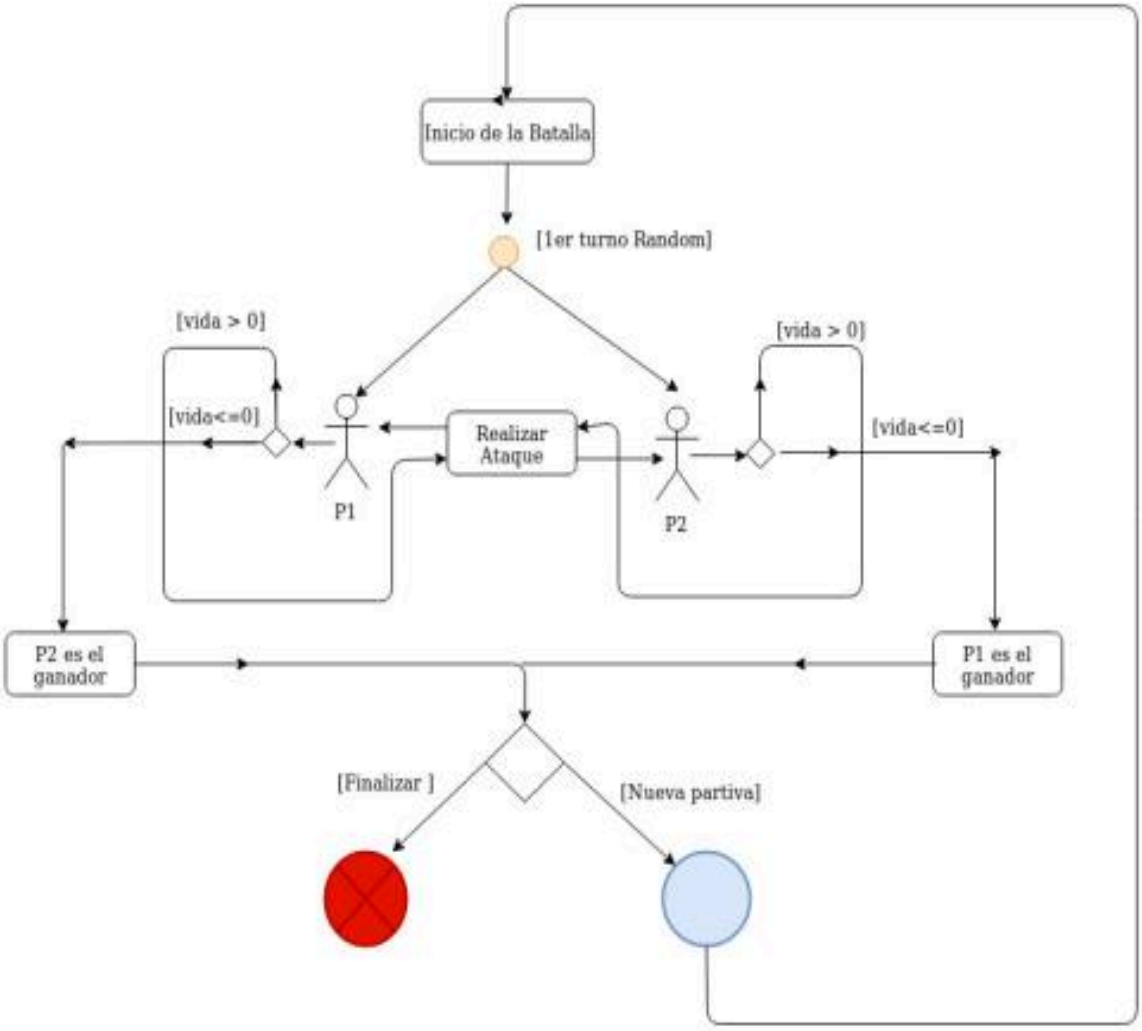


Diagrama de interacción



Obstáculos de cara al desarrollo:

- Actualmente no contamos con mucho conocimiento acerca del entorno de desarrollo usado, Android Studio, y el lenguaje de programación Kotlin, eso nos dificultó el desarrollo por los tiempos de entrega.
- Nuestros equipos de cómputo no tienen los requisitos suficientes para que el entorno de desarrollo funcionara de forma fluida, de hecho, como plan de contingencia, usamos nuestros teléfonos celulares como emulador.
- Poca información y tutoriales útiles que nos ayudaran con el desarrollo de la App.

Metas alcanzadas

- Se escogió la plataforma de desarrollo colaborativo de **Github ,** para alojar nuestro proyecto, lo cual nos brinda un doble beneficio, ya que nos brinda la opción gestionar nuestro proyecto.
- se escogió el entorno de desarrollo de android studio y el lenguaje de programación Kotlin.
- Se realizó la captura de requerimientos, la cual se encuentra en :https://github.com/MauricioALP/PokeBattle_War/blob/master/README.md
- Se realizó el maquetado o demo de nuestra App en la herramienta de MarvelApp, el cual se encuentra en : https://marvelapp.com/5bcaa1g/
- Se completó la construcción de las vistas.

Metodología de trabajo

Empezamos asignando tareas por hacer a cada miembro del equipo en el tablero Github, luego se analizaron las dependencias que existían entre dichas actividades con el fin de paralelizar la máxima cantidad de actividades posibles.

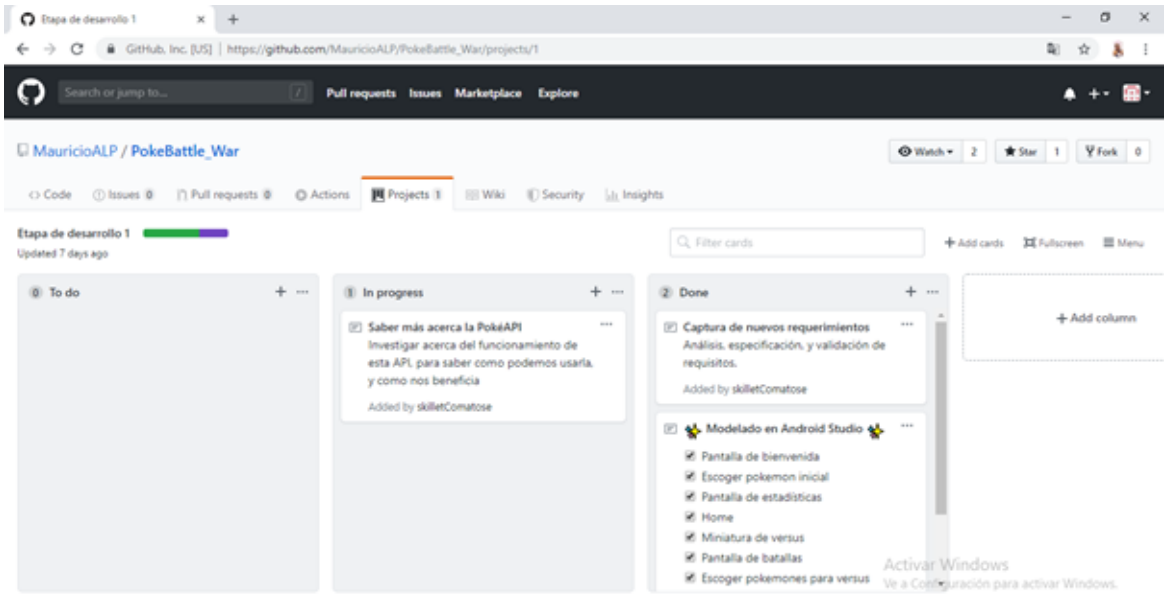


Imagen 2: Tablero de actividades en GitHub

Una vez realizadas todas las actividades pasamos a realizar las pruebas de lo que teníamos. Nos encontramos con errores como: incompatibilidad entre las pantallas, uso de gadgets inapropiados, diferentes dominios para cada pantalla, entre otros. Luego de dar solución a estos inconvenientes y conseguir pasar la prueba de funcionamiento, empezamos a trabajar en la conexión a la API, que lamentablemente aún no hemos podido lograr.