PRUEBAS BUS ROUTRACK

1. Pruebas Unitarias

Objetivo: Verificar que cada componente y función de la aplicación funcione correctamente de manera independiente. Áreas clave a probar incluyen:

- Backend: Endpoints de la API (por ejemplo, /getRoutes, /getBusLocation).
- Lógica de cálculo para tiempos estimados y distancias.
- Frontend: Comportamiento de botones y campos de búsqueda.
- Renderización de datos en el mapa.

Herramientas:

- Backend: Mocha, Jest (Node.js) o Pytest (Python).
- Frontend: Jest, Enzyme o React Testing Library.

Ejemplo de caso de prueba:

Prueba: Verificar que el endpoint /getRoutes devuelva una lista de rutas en menos de 500 ms.

Resultado esperado: Respuesta con código 200 y datos de rutas en formato JSON.

2. Pruebas de Integración

Objetivo: Asegurar que los diversos módulos de la aplicación interactúen de forma adecuada. Áreas clave a probar:

- Comunicación entre el frontend y el backend: Validar que las solicitudes al servidor devuelvan datos correctos y actualicen la interfaz.

- Integración con APIs externas (como Google Maps API y Firebase): Comprobar que los mapas se carguen sin problemas y que las notificaciones se envíen correctamente.
- Base de datos: Confirmar que las actualizaciones de rutas y ubicaciones se reflejen en tiempo real.

Herramientas:

- Postman para realizar pruebas manuales de APIs.
- Cypress para pruebas automáticas de integración entre frontend y backend.

Ejemplo de caso de prueba:

Prueba: Comprobar que el mapa interactivo muestre la ubicación de los buses después de consultar el backend.

Resultado esperado: Los buses se visualizan correctamente con datos precisos en el mapa.

3. Pruebas de Usuario

Objetivo: Identificar problemas de usabilidad y mejorar la experiencia del usuario. Metodología:

- Reclutar usuarios reales, como quienes usan frecuentemente el transporte público.
- Pedirles que realicen tareas comunes, como buscar una ruta o verificar la llegada de un bus.
- Recopilar retroalimentación sobre la navegación, diseño y funcionalidad.

Herramientas:

- Grabaciones de sesiones con Hotjar o Lookback para observar el comportamiento de los usuarios.
- Encuestas rápidas para medir la satisfacción.

Ejemplo de caso de prueba:

Prueba: Un usuario busca una parada y selecciona un bus.

Resultado esperado: El usuario encuentra la información deseada en menos de 3 clics.

4. Pruebas de Rendimiento

Objetivo: Evaluar la velocidad, estabilidad y capacidad de respuesta de la aplicación. Pruebas clave:

- Tiempo de carga del mapa en la pantalla principal.
- Latencia en la actualización de ubicaciones de buses.
- Manejo de carga con múltiples usuarios simultáneos (pruebas de estrés).

Herramientas:

- JMeter o K6 para pruebas de carga en el backend.
- Lighthouse para evaluar el rendimiento del frontend.

Ejemplo de caso de prueba:

Prueba: Simular 500 usuarios solicitando datos de buses al mismo tiempo.

Resultado esperado: El sistema responde en menos de 2 segundos en el 95% de las solicitudes.

5. Pruebas de Seguridad

Objetivo: Asegurar que la aplicación sea segura contra posibles ataques y proteja los datos del usuario. Pruebas clave:

- Validación de entradas para prevenir inyecciones SQL o scripts maliciosos.
- Cifrado adecuado de la comunicación (HTTPS).
- Manejo seguro de tokens de autenticación y credenciales de usuario.

Herramientas:

- OWASP ZAP o Burp Suite para análisis de vulnerabilidades.
- Pruebas manuales para detectar problemas como fugas de datos o accesos no autorizados.