Imagen que contiene dibujo, señal, reloj, firmar

Descripción generada automáticamente

**DESARROLLO DE APLICACIONES MÓVILES**

**Proyecto: "Aplicativo móvil en Android para el registro y seguimiento de pedidos en una empresa de transporte"**

Integrantes:

Miguel Angel Huanacchiri Castillo - U17100659

Luis Martin Palacios Morales - U17208273

Docente:  Darwin Manuel Diaz Garrampie

Lima - Perú, 2023

**ÍNDICE**

1.  Introducción.…………………………………………………… 3

2.  Objetivos.……………………………………………………… 3

3.  Alcance.…………………………………………..…………… 4

4.  Gestión del proyecto.………………………………………… 4

4.1     Cronograma.……………………………………..…………… 4

4.2     Recursos.………………………………………...…………… 5

4.3     Actividades.…………………………………………………… 5

4.4     Gestión de riesgos..……………………………..…………… 5

5.  Análisis del Sistema..………………………………………… 7

5.1     Requerimientos Funcionales..………………….…………… 7

5.2     Requerimientos No funcionales..……………………………8

6.  Diseño del Sistema..………………………………………… 9

6.1     Diseño de Arquitectura..………………………..…………… 9

6.2     Diseño Entidad / Relación..………………………………… 10

6.3     Modelos UML..…………………………………..…………… 11

6.4     Prototipos..…………………………………………………… 12

6.5     Vistas de la Aplicación..……………………………………… 14

1. **Introducción**

En la era digital actual, la gestión de pedidos se ha vuelto más que una simple transacción comercial. La competencia en el mercado, la creciente demanda de los consumidores por experiencias personalizadas y la necesidad de agilizar procesos comerciales han llevado a las empresas a buscar soluciones tecnológicas avanzadas para mejorar su eficiencia y competitividad. En este contexto, el desarrollo y la implementación de aplicativos específicos para pedidos se han convertido en un componente crítico para el éxito de las organizaciones. El objetivo principal es explorar cómo un aplicativo innovador puede optimizar los procesos de pedidos, mejorar la toma de decisiones estratégicas y, en última instancia, aumentar la rentabilidad de la empresa.

Examinaremos en detalle las necesidades del mercado, identificamos las tendencias actuales en la gestión de pedidos y las demandas cambiantes de los clientes para comprender mejor las necesidades del mercado. Funcionalidades clave: Analizaremos las características esenciales que deben incluirse en un aplicativo para pedidos eficaz, como la automatización de tareas, la integración con otros sistemas empresariales y la generación de informes. Metodología: Describiremos la metodología que utilizaremos para llevar a cabo esta investigación, incluyendo la recopilación de datos, el análisis y la evaluación de resultados. Contribución esperada: Al finalizar esta investigación, esperamos proporcionar recomendaciones y directrices útiles para la empresa que busca aprovechar al máximo las ventajas de un aplicativo para pedidos.

El mundo empresarial está en constante evolución, y la capacidad de adaptarse a las nuevas tecnologías y tendencias es esencial para el éxito a largo plazo. A medida que avanzamos en esta investigación, esperamos arrojar luz sobre cómo un aplicativo para pedidos puede convertirse en una herramienta fundamental para las empresas que buscan no solo sobrevivir, sino prosperar en un mercado competitivo y en constante cambio.

**2. Objetivos**

Objetivo General:

Diseñar un aplicativo móvil que registra los pedidos del transportista, la cual sirva para realizar un seguimiento de sus pedidos y así subsanar los errores en la gestión de pedidos.

Objetivos Específicos:

Diseñar la aplicación con un módulo de gestión o registro de pedidos para mejorar la gestión de pedidos del transportista.

Diseñar la aplicación con un módulo de información o detalles de pedidos la cual sirva para verificar los pedidos del transportista.

Diseñar un módulo de pedidos finalizados para mostrar los pedidos ya entregados o los que tuvieron algún problema en los enviados por el transportista.

**3. Alcance**

-La aplicación solo es para celulares android

-La aplicación requiere conexión a internet

-Está diseñada solo con objetivo a transportistas

-Los datos de las entregas son ingresados de manera manual

**4.Gestión del proyecto**

**4.1 Cronograma**

| **semana** | **Descripción** | **Duración** | **Resp.** |
| --- | --- | --- | --- |
| 1 | Plan de trabajo | 4 días | Miguel |
| 1 | Objetivos | 4 días | Willy |
| 2 | Alcance | 3 días | GianCarlo |
| 2 | Gestión de proyecto | 3 días | Manuel |
| 3 | Análisis de sistema | 4 días | Willy |
| 3 | Diseño de sistema | 7 días | Luis |
| 4 | Diseño de interfaz | 7 días | Miguel |
| 5 - 6 | Codificación de vista de registro de pedido | 10 días | Luis |
| 7 - 8 | Codificación de vista de detalle de pedido | 10 días | GianCarlo |
| 9 - 10 | Codificación de vista de lista de pedidos | 10 días | Manuel |
| 11 - 15 | Codificación backend | 30 días | Todos |
| 16 | Pruebas Funcionales | 7 días | Todos |
| 17 | Despliegue | 7 días | Todos |
| 18 | Cierre | 1 día | Todos |

**4.2 Recursos**

**Android Studio**

Para este proyecto usaremos todos los recursos que podemos utilizar en Android Studio que es el entorno de desarrollo integrado (IDE) oficial para el desarrollo de apps para Android y está basado en Intellij IDEA. Además del potente editor de códigos y las herramientas para desarrolladores de IntelliJ. Fue anunciado el 16 de mayo de 2013 en la conferencia Google I/O y desde entonces es un programa clave para el desarrollo de aplicaciones para Android. También Android Studio nos da las herramientas para poder diseñar las apps para Android.

Android Studio developers nos dice que su programa aumenta la productividad como las siguientes funciones:

* Un sistema de compilación flexible basado en Gradle
* Un emulador rápido y cargado de funciones
* Un entorno unificado donde puedes desarrollar para todos los dispositivos Android
* Aplicación de cambios para insertar cambios de código y recursos a la app en ejecución sin reiniciarla
* Integración con GitHub y plantillas de código para ayudarte a compilar funciones de apps comunes y también importar código de muestra
* Variedad de marcos de trabajo y herramientas de prueba
* Herramientas de Lint para identificar problemas de rendimiento, usabilidad y compatibilidad de versiones, entre otros
* Compatibilidad con C++ y NDK
* Compatibilidad integrada con Google Cloud Platform, que facilita la integración con Google Cloud Messaging y App Engine.

**Otros Recursos: GitHub, Postman, SQL Server**

**4.3 Actividades**

**4.4 Gestión de riesgos**

La gestión riesgo del proyecto nos ayuda a identificar posibles fallas que puedan aparecer en el transcurso del proyecto.

| **Riesgo identificado** | **Causas de los riesgos** | **Categoria de Riesgo** | **Nivel de Riesgo** | **Reserva Contingencia** |
| --- | --- | --- | --- | --- |
| Mala Elección de diseño de la interfaz | Falla en el área de diseño | Interno | Medio | Cambio a diseño mas amigable |
| Falla en pruebas | Problema en el Codigo | Interno | Medio | Correción oportuna de errores |
| Falla de algún entregable por parte de integrantes en el grupo | Posible problema personal, conexión, etc | Interno | Medio | Cambio de integrante asignado |
| No respuesta de un integrante de grupo | Retiro del curso | Interno | Medio | Comunicación al profesor |

**PLAN DE RESPUESTA A LOS RIESGOS**

| **Riesgo identificado** | **Categoria de Riesgo** | **Estrategias de Respuest** | **Planes de contingencia y de respaldo** | **Posibles riesgos secundarios** |
| --- | --- | --- | --- | --- |
| Falla de algún entregable por parte de integrantes en el grupo | interno | Se asignará la tarea a algún integrante del equipo | Contar con la probabilidad que un integrante falle | Que ningún integrante del equipo cuente con tiempo libre |
| No respuesta de un integrante del grupo | interno | No se considerará en el avance | Contar con la probabilidad que un integrante se retire del curso |  |

**5. Análisis del Sistema**

* 1. **Requerimientos Funcionales**

El aplicativo debe permitir registrar la información de un pedido(nro de pedido, empresa, evento y fotos de cargo y referencia), en un formulario reactivo e interactivo.

El aplicativo debe permitir el uso del scanner para el código de barras, la cámara y/o galería para las fotos de cargo y referencia.

El aplicativo debe permitir ver el detalle de un pedido(empresa, número de piezas, dirección, número de contacto).

El aplicativo debe permitir ver el seguimiento de sus eventos previos con el detalle de sus ubicaciones y fotos.

El aplicativo debe permitir ver el historial de sus pedidos, con la información del número de pedido, el evento, la empresa y la fecha en la que se registró cada pedido.

El aplicativo debe permitir registrar pedidos de forma offline, guardarlos para registrarlo una vez que tenga una conexión óptima de señal.

1. **Requerimientos No funcionales**

El aplicativo debe tener una capa de seguridad para validar qué sólo usuarios con autorización tengan el permiso de poder registrar la información de un pedido.

El aplicativo será alojado en un servidor gratuito(heroku).

El aplicativo tendrá una estructura modular, para facilitar una futura escalabilidad si el negocio lo requiere.

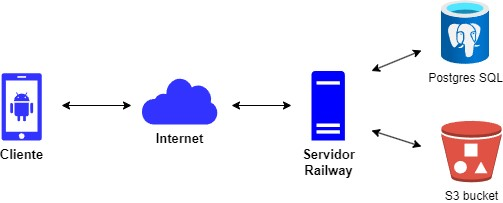
El tiempo de respuesta y carga luego de cada acción dentro del aplicativo debe ser rápido.

Se usará una base de no relacional y/o relacionales (SQL MongoDB)

El aplicativo debe funcionar en todos los dispositivos móviles.

**6. Diseño del Sistema**

**6.1 Diseño de Arquitectura**



**6.2 Diseño Entidad / Relación**

Diagrama

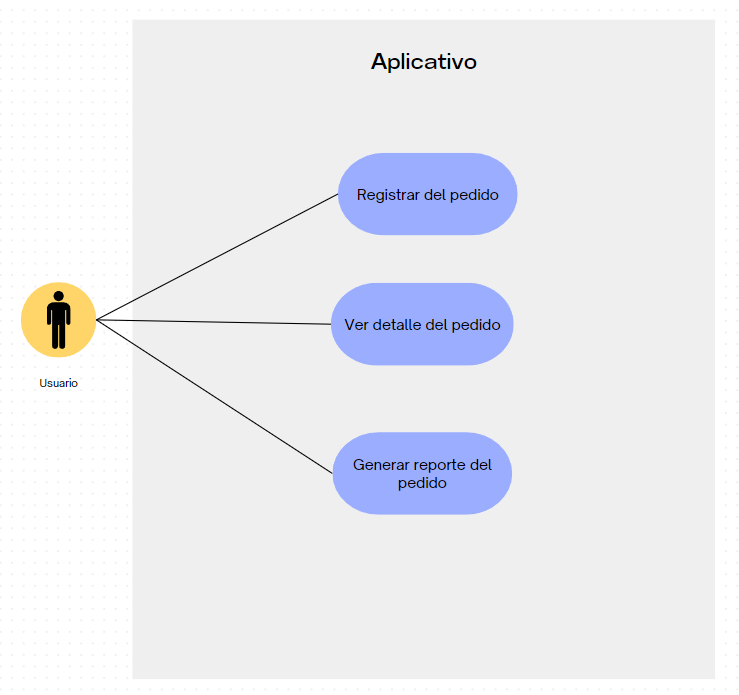
Descripción generada automáticamente

**6.3 Modelos UML**

Diagrama, Esquemático

Descripción generada automáticamente

**6.4 Casos de Uso**



**6.5 Prototipos**

Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamenteInterfaz de usuario gráfica

Descripción generada automáticamenteTabla

Descripción generada automáticamente

**6.6 Vistas de la Aplicación**

<https://www.figma.com/proto/F0qIr9IuMQttfFsaV26YYD/Tracking?node-id=115-19&starting-point-node-id=2%3A4&mode=design&t=D5wNNUn3yUSFPUC4-1>

**6.7 Diagrama de Navegabilidad**Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**7.-**

**Catálogo de Servicios**

| **Item** | **Servicio Web** | **Parámetros de Entrada** | **Parámetros de Salida** |
| --- | --- | --- | --- |
| WS1 | /ObtenerListaPedidos | offset, limit, filters | **lista**{nro\_pedido, observaciones, piezas, id\_empresa, id\_evento, id\_usuario, nombre\_empresa, nombre\_evento, nombre\_usuario} |
| WS2 | /ObtenerPedido | nro\_pedido, id\_empresa | **consultora\_info**{nombre, codigo, telefono, direccion nro\_piezas, latitude, longitude}, **trazabilidad\_lista**{id\_evento, id\_usuario, observaciones, imagen\_cargo, imagen\_referencia, latitude, longitude} |
| WS3 | /CrearPedido | nro\_pedido, id\_empresa, id\_evento, id\_usuario, observaciones, imagen\_cargo, imagen\_referencia, latitude, longitude | ok/error |
| WS4 | /ActualizarPedido | nro\_pedido, id\_empresa, id\_evento, id\_usuario, observaciones, imagen\_cargo, imagen\_referencia, latitude, longitude | ok/error |
| WS5 | /CrearUsuario | nombre\_usuario, contrasena | ok/error |
| WS6 | /AutenticarUsuario | nombre\_usuario, contrasena | id\_usuario, nombre\_usuario |
| WS7 | /ObtenerEmpresas |  | **list**{id, descripcion} |
| WS8 | /ObtenerEventos |  | **list**{id, descripcion} |
| WS9 | /ObtenerStadisticas |  | **list**{count, descripcion} |

**8.- Documentación del Desarrollo del Proyecto**

**Versión de Android:** Android 8.0.1.

**Distribución de Carpetas**.

**Carpeta Adapters**: Este paquete contiene adaptadores que se utilizan para gestionar la presentación de datos en la interfaz de usuario.

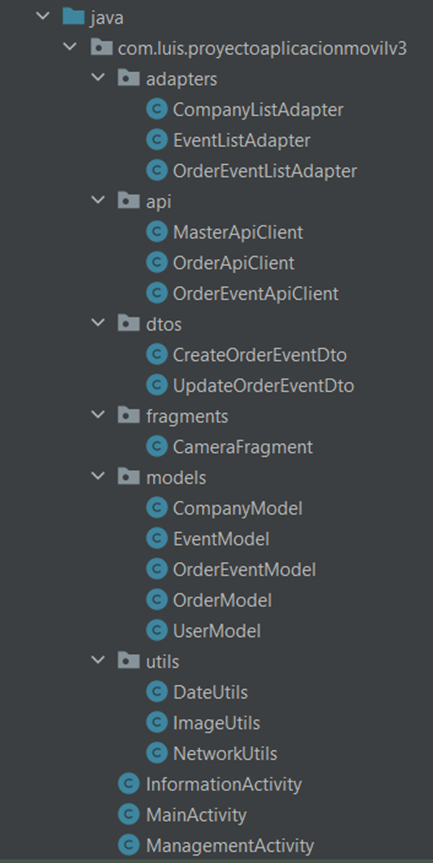
**Carpeta Api:** Este paquete contiene clases que se encargan de las interacciones con los servicios web a través de retrofit

**Carpeta Dtos:** Con este paquete se utiliza para transferir la información que se envía al api. Sirve para estructurar la data que vamos a enviar al api.

**Carpeta Fragments:** Contiene fragmentos de la interfaz de usuario como CameraFragment se usa en la vista de registro.

**Carpeta Models:** Contiene modelos que representan entidades de datos en la lógica de la aplicación.

**Carpeta Utils:** Este paquete contiene clases de utilidad que ofrecen utilidades relacionadas con manipulación de fechas, imágenes y gestión de la red.



**Librerías Utilizadas**

**Retrofit:** Esta librería simplifica la comunicación son los servicios web.



Documentación: https://square.github.io/retrofit/

**Picasso :** Simplifica el proceso de carga y visualización de imágenes en Android.



Documentacion: <https://square.github.io/picasso/>

**MPAndroidChart:**Generacion de Charts



**Google Maps Services:** Para la marcacion de las ubicaciones en el mapa.

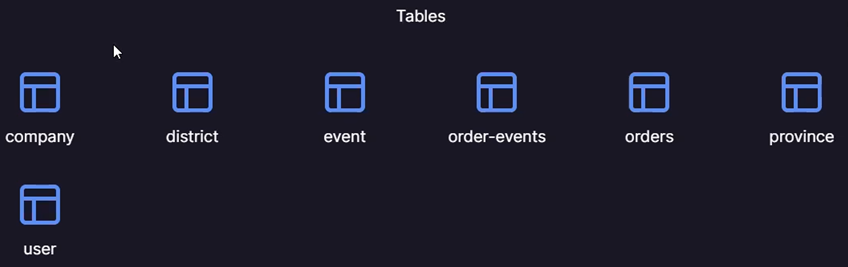


**Despliegue de la Base de Datos.**

**Host utilizado:** Railway.

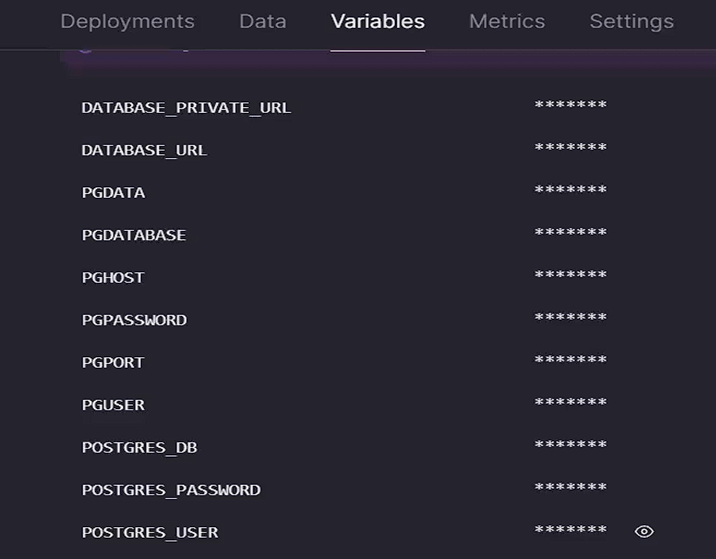
**Base de datos:** Postgres.

Sección Data: Muestra las tablas con la información inicial.



**Sección Variables**:

Muestra las variables de la base de datos donde se especifica los accesos o claves que requiere guardar.

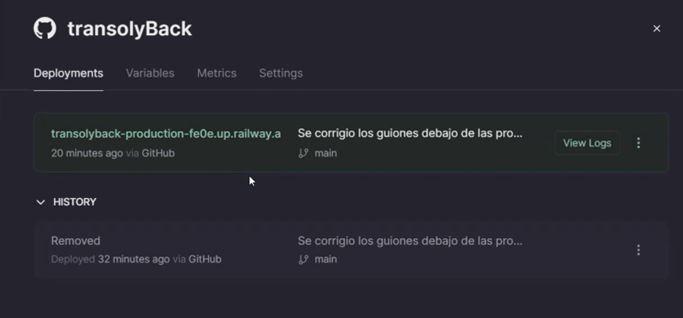


**Despliegue del Servidor**

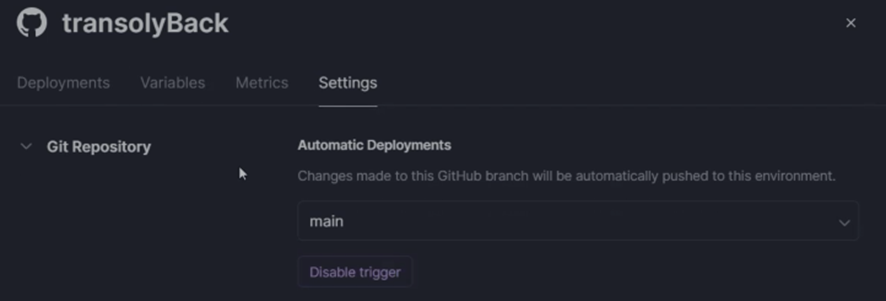
**Entorno de ejecución**: NodeJS.

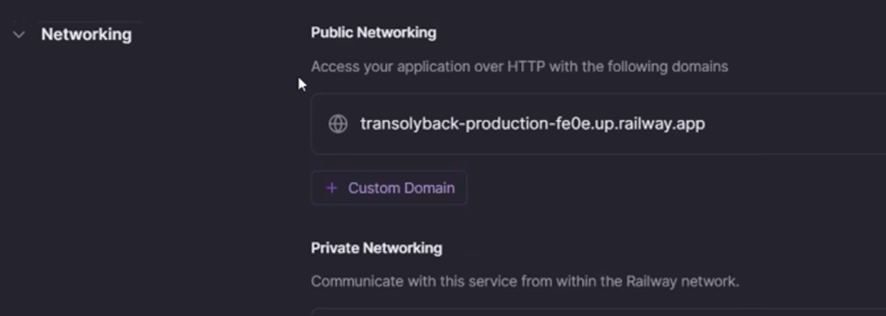
**Framework**: NestJS.

**Sección Deployments**: Muestra el historial de todos los despliegues realizados.

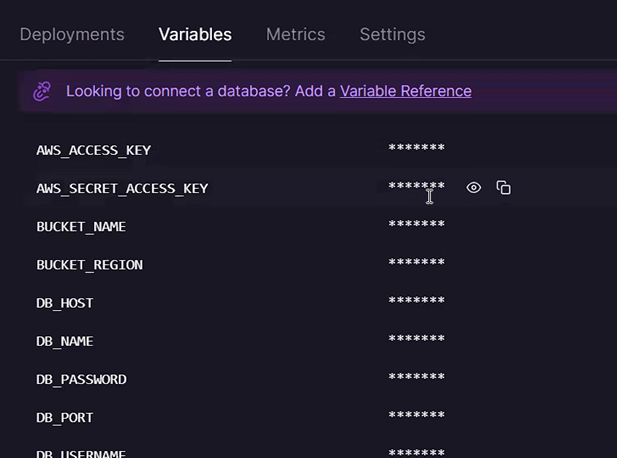


**Sección Settings**: Muestra el repositorio git y el dominio del servidor.





**Sección Variables**: Se encuentran las variables de entorno, estas las cadenas de conexión, las credenciales y puertos para el acceso a la base de datos y los accesos para los buckets.



**LINKS A LOS REPOSITORIOS:**

**App: https://github.com/LDacce/ProyectoAplicacionMovilv3**

**Backend:** [**https://github.com/gremmil/transolyBack**](https://github.com/gremmil/transolyBack)