CENTRO DE ESTUDIOS CIENTÍFICOS Y TECNOLÓGICOS

“JUAN DE DIOS BATIZ”

Laboratorio de Desarrollo de Software III

MANUAL DE DESAR

5IV8

PROFESOR:Roman Alvarado

Equipo:

- **Camargo Huertero Jesus Saul**
- **Esquivel Zarate Leonardo**
- **Flores Raya Cesar Damian**
- **Hernandez Hernandez Cesar Eduardo**
- **Lopez Garcia Miguel Angel**
- **Rueda Gutierrez Adrian Everardo**

Índice

Introducción

Este manual está dirigido a los desarrolladores encargados de la programación y mantenimiento de **Mech-Record**, una plataforma orientada a mecánicos, que permite gestionar el historial de vehículos, compras, y adeudos. A través de este software, se podrá interactuar con un sistema de tickets, ver el flujo de vehículos que entran y salen del taller, llevar un registro detallado en una bitácora, y realizar compras dentro de la tienda del taller. Además, incluye un sistema de chat en tiempo real, conectado a la base de datos y gestionado por un backend en Node.js y MySQL.

Objetivo, propósito y alcance

Objetivo del manual

El objetivo de este manual es proporcionar a los desarrolladores una guía clara sobre cómo trabajar con el sistema Mech-Record, cómo importar la base de datos, y cómo interactuar con las tecnologías y las funcionalidades específicas, como la validación de sesión, la integración de la tienda, el sistema de chat en tiempo real, y el manejo de vehículos.

Propósito del manual

El propósito de este manual es permitir una correcta implementación de las funciones del software, asegurar que los desarrolladores puedan manejar el sistema de forma eficiente, y facilitar el mantenimiento y la evolución del proyecto.

Alcance del manual

Este manual cubre la arquitectura y las funcionalidades clave del sistema, la configuración de la base de datos, y el uso de las tecnologías involucradas (Node.js, React Native, MySQL, WebSockets, etc.).

Arquitectura del software

El software Mech-Record está compuesto por las siguientes capas:

1. **Frontend (React Native):**
   La interfaz de usuario se desarrolla con React Native, permitiendo que la aplicación sea compatible con dispositivos móviles. El frontend se conecta con el backend para interactuar con los datos y realizar operaciones en tiempo real.
2. **Backend (Node.js):**
   Utiliza Node.js para manejar las solicitudes del cliente, validar la sesión, gestionar el carrito de compras, y realizar operaciones con la base de datos.
3. **Base de datos (MySQL):**
   Almacena toda la información de los clientes, vehículos, tickets de compra, y chat. MySQL Workbench se usa para gestionar la base de datos.
4. **Comunicaciones en tiempo real (WebSockets):**
   Se utiliza WebSockets para implementar un chat en tiempo real entre los usuarios, permitiendo una interacción instantánea.

Base de datos y configuración

Importación de la base de datos en MySQL Workbench

Para trabajar con la base de datos en **MySQL Workbench**, sigue estos pasos:

1. **Instalación de MySQL Workbench:**
   - Si aún no tienes MySQL Workbench instalado, puedes descargarlo desde aquí.
2. **Crear la base de datos en MySQL:**
   - Inicia MySQL Workbench y conéctate a tu servidor de base de datos.

Crea una nueva base de datos llamada `mech_record` usando el siguiente comando:
sql
Copiar código
```
CREATE DATABASE mech_record;
```

   -
3. **Importar la base de datos:**

    - Descarga el archivo `.sql` que contiene la estructura de la base de datos para el proyecto.
    - En MySQL Workbench, ve a **File** → **Open SQL Script**, y selecciona el archivo `.sql`.
    - Ejecuta el script para crear las tablas necesarias (usuarios, vehículos, tickets, bitácora, etc.).
4. **Verificación de tablas:**

Después de ejecutar el script, asegúrate de que las tablas se hayan creado correctamente mediante:
sql
Copiar código
```
SHOW TABLES;
```

    -
5. Esto debería mostrar las tablas correspondientes para el manejo de los vehículos, tickets, usuarios, entre otros.

Tecnologías utilizadas

1. **Node.js:**
   Backend basado en Node.js para manejar las peticiones y las conexiones a la base de datos.
2. **React Native:**
   Framework para el desarrollo de la aplicación móvil, permitiendo que la misma aplicación sea compatible con plataformas Android e iOS.
3. **MySQL Workbench:**
   Para la gestión de la base de datos MySQL, que almacena la información crítica como los registros de los clientes, vehículos, y tickets.
4. **WebSockets (Socket.io):**
   Usado para implementar el sistema de chat en tiempo real, permitiendo la comunicación instantánea entre los usuarios del sistema.

Flujo de trabajo y funcionalidades

Validación de sesión

La validación de sesión se realiza en el backend usando **JSON Web Tokens (JWT)**. Al iniciar sesión, el sistema valida las credenciales del usuario y genera un token JWT que se almacena en el dispositivo del cliente. Cada vez que el usuario realiza una solicitud, el token JWT se envía al servidor para verificar que la sesión esté activa.

Carrito de compras con ticket

- **Flujo:** Los usuarios pueden agregar productos al carrito de compras (repuestos o servicios) y, al finalizar, generar un ticket que incluya un resumen de la compra.
- **Tecnología:** El ticket es generado en el frontend, y los datos de la compra se almacenan en la base de datos de MySQL.

Bitácora de vehículos

- **Funcionalidad:** Cada vez que un vehículo entra o sale del taller, se registra automáticamente en la bitácora con la fecha, hora, y el cliente relacionado.
- **Almacenamiento:** La información de la bitácora se guarda en la base de datos, y los usuarios pueden consultar el historial de los vehículos.

Sistema de chat en tiempo real (WebSockets)

- **Flujo:** Los usuarios pueden interactuar mediante un chat en tiempo real utilizando **Socket.io**.
- **Conexión:** El cliente se conecta al servidor de WebSockets (Node.js) y puede enviar y recibir mensajes instantáneamente.

Configuración del entorno de desarrollo

Node.js y React Native

1. **Instalar Node.js:**
   Asegúrate de tener instalado **Node.js** en tu máquina. Puedes instalarlo desde aquí.
2. **Crear el proyecto de Node.js:**

Ejecuta los siguientes comandos en tu terminal para crear un proyecto de Node.js:
bash
Copiar código

```
mkdir mech-record-backend

cd mech-record-backend

npm init -y

npm install express mysql2 jsonwebtoken socket.io
```

-

Iniciar servidor Node.js:
Para iniciar el servidor, crea un archivo `server.js` que maneje las conexiones y la autenticación de sesión:

js
Copiar código

```
const express = require('express');

const mysql = require('mysql2');

const jwt = require('jsonwebtoken');

const socketIo = require('socket.io');

const app = express();


// Base de datos

const db = mysql.createConnection({

  host: 'localhost',

  user: 'root',

  password: 'password',

  database: 'mech_record'

});


// Autenticación y WebSocket en el servidor

// (Código de conexión y manejo de WebSockets)


app.listen(3000, () => console.log('Servidor corriendo en
http://localhost:3000'));
```

3.

Instalar React Native: Para el frontend móvil, instala React Native CLI:
bash
Copiar código

```
npx react-native init MechRecordApp

cd MechRecordApp
```

4. Luego puedes usar el código de React Native para conectar el frontend con el backend de Node.js.

Consideraciones para futuras actualizaciones

1. **Mejoras en seguridad:**
   Asegurar que las contraseñas sean encriptadas correctamente, y que las sesiones estén protegidas.
2. **Optimización del rendimiento:**
   Mejorar el manejo de la base de datos y las consultas para garantizar que el sistema sea escalable.
3. **Interfaz de usuario (UI/UX):**
   Continuar con la mejora de la experiencia de usuario, tanto en la aplicación móvil como en el sistema de administración.