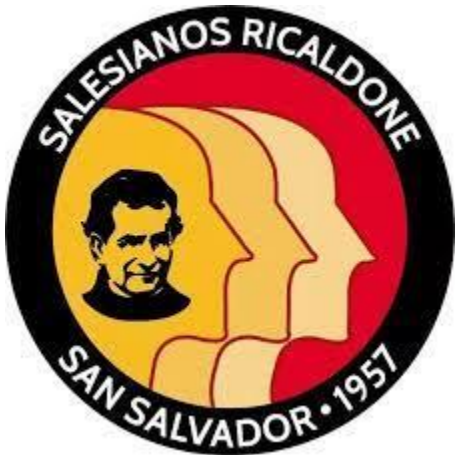
Instituto técnico ricaldone



Alumno: Carlos Mateo Amaya Meléndez

Docente: Daniel Wilfredo Granados

Especialidad: Desarrollo de software

Sección: A

“A manera de repaso, de forma individual se deberá de crear una guía paso a paso, explicando los pasos que se deben de seguir para la creación de una aplicación móvil con react native utilizando expo, como se deben de organizar las carpetas del proyecto y como se hace uso de los custom hooks a la hora de crear una aplicación que utilice FETCH de Javascript para comunicarse con una API del Backend.

Deberás de crear una app de ejemplo con un crud y subirla a github junto con la guía de su creación

paso a paso.”

Guía paso a paso: CRUD de Empleados con React Native + Expo

1. Se crea la carpeta donde se estará trabajando, dentro de esta pondremos todo lo necesario para hacer un crud

   Creamos el backend y descargamos sus dependencias
   Empezaremos por estas:
   npm install express mongoose cors dotenv
   npm install --save-dev nodemon

   ahora que tenemos el backend, creamos su estructura, esta quedaría algo asi:
   backend-simple/
   ├── src/
   │ ├── controllers/
   │ ├── models/
   │ ├── routes/
   │ └── app.js
   ├── .env
   └── package.json

   en el archivo .env pondremos el puerto y la conexión a la base de datos, esta quedaría algo asi:

```
PORT=4000
MONGO_URI="mongodb://127.0.0.1:27017/employeesDB"
```

Luego en models, definimos los campos que habran, para simplificar solo usaremos el name, quedaría algo asi:

```
import mongoose from "mongoose";

const employeeSchema = new mongoose.Schema({
  name: { type: String, required: true },
});

export default mongoose.model("Employee", employeeSchema);
```

luego, en app, pondremos las rutas donde definimos al empleado y también conectaremos con mongoDB, quedando algo asi:

```
import express from "express";
import mongoose from "mongoose";
import cors from "cors";
import dotenv from "dotenv";
import employeeRoutes from "./routes/employeeRoutes.js";

dotenv.config();

const app = express();
app.use(cors());
app.use(express.json());

// Conectar a MongoDB
mongoose.connect(process.env.MONGO_URI)
  .then(() => console.log("DB connected"))
  .catch(err => console.log(err));

// Rutas
app.use("/api/employees", employeeRoutes);

// Iniciar servidor
```

```
const PORT = process.env.PORT || 4000;
app.listen(PORT, () => console.log(`Server running on port
${PORT}`));
```

Ahora en routes creamos la ruta de nuestro empleado, en donde cada ruta tiene un controlador, quedando algo asi:

```
import express from "express";

import { getEmployees, createEmployee, updateEmployee, deleteEmployee }
from "../controllers/employeeController.js";


const router = express.Router();


router.get("/", getEmployees);

router.post("/", createEmployee);

router.put("/:id", updateEmployee);

router.delete("/:id", deleteEmployee);


export default router;
```

por ultimo creamos el controlador de empleado, donde pondremos cada funcionamiento del crud, GET, POST, PUT y DELETE, quedaría algo asi:

```
import Employee from "../models/employee.js";


export const getEmployees = async (req, res) => {
  const employees = await Employee.find();
  res.json(employees);
};
```

```
export const createEmployee = async (req, res) => {
  const newEmp = new Employee(req.body);
  await newEmp.save();
  res.json(newEmp);
};

export const updateEmployee = async (req, res) => {
  const updated = await Employee.findByIdAndUpdate(req.params.id,
req.body, { new: true });
  res.json(updated);
};

export const deleteEmployee = async (req, res) => {
  const deleted = await Employee.findByIdAndDelete(req.params.id);
  res.json(deleted);
};
```

Ahora en package.json ponemos en scripts el “dev” ("dev": "nodemon src/app.js")para poder arrancar el servidor con npm run dev

Ahora conectamos el backend con el frontend, la línea de código para hacer esto es asi:

const BASE_URL = "http://(la ip de su porpia pc):4000/api/employees";

ahora solo queda trabajar en el frontend, para esto crearemos plantillas que hagan funcionar al crud que esta en backend, como:

Lista de empleados: donde se podrá editar y eliminar empleados

Formulario de empleados: donde se podrán agregar y editar empleados

Estas pantallas están en el app navigator que se usa para poder moverse libremente de pantalla a pantalla

Ahora que tenemos listo el backend y el frontend sin que nos de alguna clase de errores, arrancamos el backend y el frontend para que los dos esten funcionales y dependediendo como lo queramos probar seguiremos distintos pasos, en este caso, conectaremos nuestro proyecto a un emulador de Android studio, estos son los pasos a seguir:

Abrimos Android studio y nos vamos a devices, encendemos cualquier dispositivo previamente descargado

Nos vamos a el backend con cd backend y lo corremos con npm run dev

Nos vamos a el frontend con cd frontend y lo corremos con npx expo start

Desde el mismo panel se nos enseñaran los siguientes pasos, si tienes un celular Android con el código QR que se otorga puedes ver como es que te ha quedado, si no, le daremos a la tecla “a” para que el proyecto se conecte con el emulador de Android studio que tengamos abierto se comenzara a enseñar el proyecto

Ahora probamos que todo lo que hicimos funcione, y eso es la guía paso a paso de como hacer un crud con React Native y Expo

Link de github: