Versión 1.0.0.0.0.0
Fecha de Publicación: 27/11/2024
Huancayo, Junín, Perú

MANUAL DE TÉCNICO
APLICACIÓN TRAINUP

ÍNDICE

INTRODUCCIÓN

Este manual técnico está diseñado para proporcionar una visión general sobre el sistema MERN (MongoDB, Express, React, Node.js) de la aplicación. La aplicación permite a los usuarios gestionar rutinas de entrenamiento y realizar sus rutinas en conjunto a la aplicación.

REQUISITOS DEL SISTEMA

REQUISITOS DE SOFTWARE

Windows 10
Visual studio code: Visual Studio Code
MERN(MongoDB, Express, React, Node.js)
docker: Install Docker Desktop on Windows | Docker Docs
Postman: Install Postman Desktop on Windows.
github/git

REQUISITOS MÍNIMOS DE HARDWARE

- Procesador: Intel core i5
- Memoria RAM: 8GB
- Disco Duro:128 GB
- Resolución de pantalla: 1280 x 720 píxeles
- Conexión a Internet: Necesaria para sincronización y acceso a datos en tiempo real.

TÉCNICAS

La aplicación está construida utilizando las siguientes tecnologías:

1. MongoDB: Base de datos NoSQL utilizada para almacenar los datos del usuario, rutinas y otras configuraciones. Se utilizó Mongoose como ODM (Object Data Modeling) para interactuar con MongoDB.
2. Express: Framework de Node.js utilizado para construir la API RESTful que

conecta el front-end con el back-end.
3. React: Biblioteca de JavaScript utilizada para construir la interfaz de usuario en el lado del cliente. Utiliza React Router para la navegación entre páginas.
4. Node.js: Entorno de ejecución de JavaScript para ejecutar el back-end.

5. Postman: es una herramienta que permite desarrollar, probar y documentar API de manera sencilla. Facilita la realización de solicitudes HTTP, la verificación de respuestas y la colaboración entre equipos de desarrollo.
INSTALACIÓN:

Para comenzar instalaremos el Visual STudio Code, y luego Mongodb,node.js

INSTALACIÓN DE Mongo

```
npm install mongoose
```

INSTALACIÓN DE Express

npm install express

INSTALACIÓN DE React

npx create-react-app nombre-de-tu-aplicacion

INSTALACIÓN DE React router

npm install react-router-dom

LEVANTAMIENTO DE BASE DE DATOS

docker compose up -d

INICIALIZACIÓN DEL PROYECTO

npm run dev

ESTRUCTURA DEL PROYECTO

La estructura del proyecto sigue el patrón de arquitectura de tres capas. A continuación se describe la organización de los directorios principales:

1. Backend (Servidor)**:
    - `index.js`: Archivo principal donde se configura el servidor Express y se conecta a MongoDB.
    - `models/`: Contiene los modelos de Mongoose, incluyendo `User.js`, `Rutina.js` y otros.
    - `routes/`: Contiene las rutas de la API para gestionar las solicitudes HTTP,
    - `controllers/`: Contiene la lógica para manejar las rutas, como la creación o actualización de un usuario.

2. Frontend (Cliente):
    - `src/`: Carpeta principal con todos los componentes de React.
    - `components/`: Componentes reutilizables como `FragmentPerfil.js`, `FrameRunEjercicio.js` y otros.
    - `App.js`: Componente principal donde se maneja la navegación con React Router.

3. Base de Datos:
    - `MongoDB`: Utiliza MongoDB como base de datos NoSQL para almacenar información sobre rutinas y usuarios.

INICIAR CON GIT

CONFIGURACIÓN INIC

CONFIGURACIÓN GENERAL DEL USUARIO GIT

```
git config --global user.name "tu usuario"
git config --global user.email "tu correo"
```

CONFIGURACIÓN DEL FRONTEND

```
git clone https://github.com/HieysonS/trainup
```

CARACTERÍSTICAS PRINCIPALES

1. Autenticación de Usuarios: Los usuarios pueden registrarse e iniciar sesión usando JWT (JSON Web Tokens) para autenticar sus solicitudes.
2. Gestión de rutinas: Los usuarios pueden ver, crear, editar y eliminar rutinas.
3. Autenticación en 2 pasos: Los usuarios necesitan realizar una autenticación en 2 pasos para realizar el login o cambiar de contraseña.

MODELADO DE DATOS

La base de datos de la aplicación contiene varias colecciones importantes como `Usuarios`, `Rutinas` y `Sesiones`.

1. **User**Contiene los datos del usuario como nombre, email, fecha de nacimiento y otros detalles relevantes.
2. **Rutina**Almacena la información sobre cada rutina de ejercicio.
3. **Sesión Rutina**: Registra las sesiones realizadas por cada usuario para cada rutina.

ENDPOINTS Y API

Los principales endpoints de la API son los siguientes:

1. `POST /api/users`: Registro de un nuevo usuario.
2. `GET /api/users`: Obtener todos los usuarios.
3. `GET /api/rutinas`: Obtener todas las rutinas.
4. `POST /api/rutinas`: Crear una nueva rutina.
5. `DELETE /api/rutinas/:id`: Eliminar una rutina.

ACTUALIZACIONES Y MANTENIMIENTO

Procedimientos para realizar actualizaciones.

- **Verificación de la Versión Actual:**
  - Identificar la versión actual instalada.
  - Comprobar la disponibilidad de nuevas actualizaciones.
- **Copia de Seguridad:**
  - Realizar una copia de seguridad completa de la base de datos y archivos relevantes antes de comenzar la actualización.
- **Acceso a Recursos de Actualización:**
  - Consultar los contactos oficiales de "Trainup" o el portal de soporte para obtener acceso a las últimas actualizaciones.
- **Desactivación Temporal del Sistema:**
  - Notificar a los usuarios sobre la inminente actualización.
  - Desactivar temporalmente el acceso al sistema para evitar cambios concurrentes durante la actualización.

SEGURIDAD

La seguridad de la aplicación se asegura mediante el uso de JWT para la autenticación de los usuarios y la protección de las rutas del backend mediante middleware que valida el token del usuario.