Página Web para subir y descargar Archivos

Resumen

En esta documentación redactare como hice la pagina web, para que propósito será utilizado y que herramientas utilice.

Objetivo

El objetivo principal es permitir que los usuarios dentro de la red de Colun puedan tener un fácil acceso a los manuales y aplicaciones que la empresa ofrecerá en distintos casos. Donde también los administradores podrán subir los manuales y/o aplicaciones e ir dándole el respectivo mantenimiento (actualizar/borrar archivos).

Arquitectura y Tecnologías utilizadas

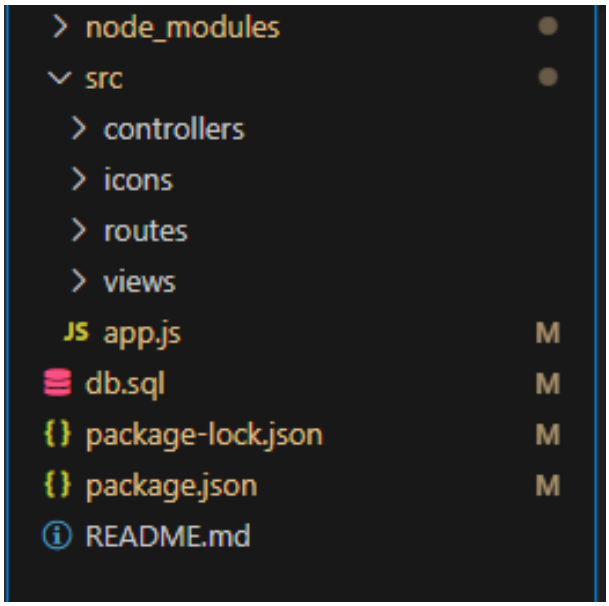
Utilizo como frontend: HandleBars el cual es un popular sistema de plantillas en JavaScript que te permite crear y formatear código HTML de una forma sencilla.

Para el backend utilizo MariaDB, un conocido gestor de base de datos relacionales, el cual es gratuito y de

código abierto, similar al MySQL por no decir igual. Y a través del mismo Node.js el cual es un entorno que trabaja en tiempo de ejecución de código abierto, comúnmente utilizado en el uso de servidores y aplicaciones, uso varias de sus librerías donde puedo conectarme con la base de datos (Ejemplo: Express).

Estructura de Archivos

El manejo de archivos se vera de la siguiente estructura en la imagen:



Todo lo que node_modules será simples extensiones que se instalan para poder ejecutar el servidor de manera local y muchas otras funcionalidades.

En src tendremos los archivos que contendrán los principales componentes que componen a la página.

La app.js se encarga de cargar las extensiones de node, como MySQL, , que es una herramienta que permiten la conexión de la base de datos con el código en si o como fileUpload que permite el manejo de subir archivos a la base de datos.

Controllers tendrá todas las funciones que realizará la página, como por ejemplo verificar que el usuario sea administrador, que el usuario al apretar el botón de descarga se ejecute esta función permitiendo que sea descargada en el la carpeta de descargas.

Icons será más que nada las imágenes que se utilizaron para que la pagina sea mas vistosa y cumpla con el propósito de representar de buena mera a la empresa.

Routes tendrá las rutas entre los controladores y la interfaz gráfica.

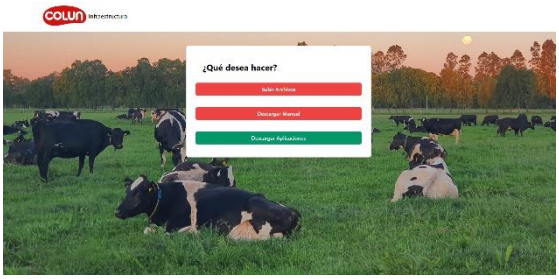
Views tendrá a los que anteriormente mencione, los Handlebars, que se usaran para controlar todo lo visual dentro de la página.

El archivo db.sql tendrá la estructura de la base de datos, en formato mariadb.

Funcionalidades y uso

Uno podrá ingresar a través de un ip a la pagina web, donde ese ip es: ...

Una vez dentro de la pagina podra ver lo siguiente:

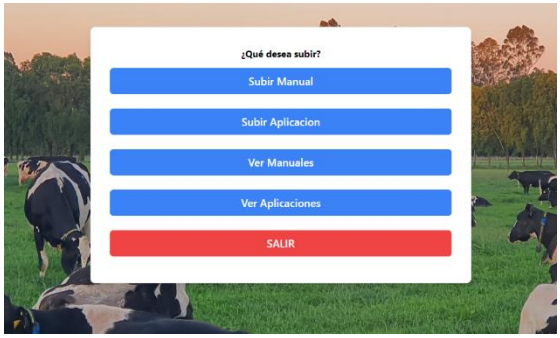


Aquí el usuario tendrá 3 opciones:

Subir Archivos: necesitara una cuenta de administrador para poder subir archivos.

Descargar Manuales/Aplicaciones: dependiendo de lo que necesita podrá ver el archivo y con un botón de descargar lo descargará automáticamente.

Para los que tienen acceso como administrador:



Verán todas estas opciones donde podrá subir los Manuales/Aplicaciones o bien los podrá visualizar los que están subidos a la base de datos, donde podrá eliminarlos si en algún futuro ya no necesitan ese manual o actualizar si tienen uno mas reciente para la fecha.

Lógica utilizada

```
const express = require('express');
const { engine } = require('express-handlebars');
const myconnection = require('express-myconnection');
const mysql = require('mysql');
const fileUpload = require('express-fileupload');
const session = require('express-session');
const bodyParser = require('body-parser');
const loginRoutes = require('./routes/login');
const { redirect } = require('express/lib/response');

const app = express();
app.set('port', 5000);

app.set('views', __dirname + '/views');
app.engine('.hbs', engine({
  extname: '.hbs',
}));
app.set('view engine', 'hbs');

app.use(express.static('src'));

app.use(bodyParser.urlencoded({
  extended: true
}));
app.use(bodyParser.json());
app.use(fileUpload()); // Habilitar el middleware de fileUpload
app.use(myconnection(mysql, {
 host: 'localhost',
 user: 'root',
 password: 'colun24',
 port: 3306,
 database: 'nodel'
}, 'single'));

app.use(session({
  secret: 'secret',
  resave: true,
  saveUninitialized: true
}));
```

Archivo app.js, aquí esta toda la lógica de los paquetes utilizados para el uso de esta página, como también, como se usan y como se conectan con la base de datos, dándole todas las indicaciones necesarias.

Visualización de los archivos:

```
function verManuales(req, res){
  req.getConnection((err, conn) => {
    conn.query('SELECT * FROM aarchivos', (err, archivos) => {
      if(err) {
        res.json(err);
      }
        res.render('login/verManuales', {archivos});
    });
  });
};
```

Archivo LoginController.js en la carpeta controllers, generalmente para ver los archivos. borrarlos o descargarlos, se va hacer consultas SQL para poder entrar a la base de datos y realizar la respectiva descarga.

Futuras Mejoras

Se tendrá la idea de agregar un nuevo botón, donde los llevara directamente a la página de descargar, y se descargar el correspondiente archivo requerido.

Personal Presente en la operación
Benjamín Cea