MANUAL TÉCNICO CLOUD-ARCH POR:

Erick Daniel Morales Xicara

Carné: 201930699

Manual Técnico

El software puede ser utilizado en cualquier sistema operativo ya que se utilizó una aplicación web, también se podrá acceder en todas las computadoras que tengan internet . Se utilizó nodejs como backend, y el Visual Studio Code. A su vez para el soporte de frontend se utilizó el framework Angular 15.2.4, npm 9.8.1

El software consiste en el manejo de información, mediante el almacenamiento en la nube, el cual permite la gestión de carpeta/directorios y archivos de texto los cuales pueden ser de tipo .txt o .html

El software a su vez contiene varias clases y manejadores que permiten iniciar, ejecutar y realizar las peticiones en el sistema antes mencionado.

FRONTEND

Carpeta models:

Esta contiene los modelos utilizados para poder convertir los json de la base de datos a objetos en el frontend y el manejo de cada uno, los cuales son:

- Archivo
- Carpeta
- Papelera
- Path
- Respuesta
- Shared
- User

Las clases sólo poseen los atributos y métodos constructores, y sus getters y setters

Carpeta Views:

- Admin Option: Esta carpeta tiene las vistas utilizadas para las opciones exclusivas del administrador
    - Crear Nuevo empleado
        - newEmpleado(): método utilizado para poder guardar un nuevo empleado, el cual verifica que no

exista el nombre de usuario anteriormente en el sistema

- saveEmpleado(): método utilizado para enviar a guardar el usuario al sistema
- clear(): método utilizado para limpiar el formulario

- Papelera (acceso a ella):
  - pedirPapelera(): método utilizado para pedir todos los archivos de la papelera
  - verContenido(): método para ver el contenido del archivo seleccionado
  - salir(): método utilizado para salir de ver el contenido del documento seleccionado
- Ver Empleados
  - obtenerUsuarios(): método utilizado para pedir todos los usuarios del sistema para mostrarlos en una tabla
  - ocultarPassword(): método utilizado para poder "dibujar" y mostrar un número de asteriscos dependiendo del tamaño de la contraseña del usuario
  - deleteUser(): método utilizado para eliminar al usuario del sistema
  - updateUser(): método utilizado para pedir toda la nueva informacion del usuario
  - updateFinalUser(): método utilizado para realizar la actualización de todos los datos del usuario

Carpeta Home:

- Esta carpeta solamente contiene un método para obtener la opción seleccionada por el usuario para poder así cambiar la ventana según la opción del menú seleccionado, controla las opciones exclusivas para un usuario.

Carpeta Home-Admin:

- Esta carpeta solamente contiene un método para obtener la opción seleccionada por el administrador para poder así cambiar la ventana según la opción del menú seleccionado, controla las opciones exclusivas para un usuario, añadiendo las opciones exclusivas de un administrador.

Carpeta Login:

- userAuth():Esta carpeta solamente contiene un método para poder validar que el ingreso del usuario y contraseña sean correctos y redireccionarlo dependiendo si es usuario o administrador

Carpeta Menu:

- Carpeta menu: Esta carpeta contiene los métodos utilizados para controlar las opciones exclusivas de un usuario

    - cambiarHojaUser(): método utilizado para cambiar el valor de la variable alojada en el servicio el cual servirá para cambiar de componente según sea la opción
    - cerrarSesion(): método utilizado para borrar todos los componentes guardados en el local storage y reenviar a la vista de inicio de sesion:
    - validarSesion(): método utilizado para validar la sesión del usuario, si existe el usuario en el local storage puede regresar sin problemas, sino solamente lo redirigirá a la página de inicio de sesión
- Carpeta menú-admin: contiene exactamente los mismos métodos que el menú normal, solamente que las variables que se modifican son del servicio utilizado exclusivamente para el administrador

Carpeta User-Options: Esta carpeta contiene los metodos utilizados para las acciones exclusivas de un usuario.

- Change-Pass: contiene los metodos necesarios para el cambio de contraseña del usuario:
    - ngOnInit(): método utilizado para solicitar al usuario en "sesion" que vuelva a ingresar la contraseña actual, y si es correcta le muestra el formulario de cambio de contraseña y sino lo redirige al inicio:
    - updatePass(): método utilizado para enviar la peticion para actualizar la nueva contraseña

- Editor: contiene metodos necesarios para ver un archivo ya en el sistema o crear uno nuevo.
    - OnCodeChange(): método utilizado para llamar al método que obtendra la linea y columna en que se encuentra el usuario.
    - ObtenerPosicion(): método utilizado para obtener la linea y columna donde se encuentra situado el usuario
    - saveFile(): método utilizado para guardar el nuevo archivo creado, verificando que el nombre no contenga caracteres especiales
    - salir(): método utilizado para salir de la creacion/edicion del archivo sin guardar los cambios realizados
    - updateFile(): método utilizado para sobreescribir el contenido y cambiar el nombre del archivo si el usuario asi lo hiciese
    - obtenerNameSinExtension(): método utilizado para saber solamente el nombre del archivo, dado que se guarda como nombre del archivo'.'tipo de archivo
- Inicio-User: contiene solamente los metodos utilizados para poder cambiar de componente según la opcion seleccionada por el usuario de la parte de documentos y compartidos
- MyDocuments: contiene los metodos necesarios para el manejo de archivos y carpetas del usuario
    - newDirectory(): método utilizado para crear una nueva carpeta, la cual solamente se solicita el nombre

- buscarCarpetasa(): método utilizado para obtener las carpetas del usuario dado el path actual o el path donde se encuentra
- clickCarpeta(): método utilizado para mostrarle al usuario las opciones que puede realizar con la carpeta, abrir, mover copiar y eliminar, el cual verifica la opcion y envia al método según la opcion.
- copiarCarpeta(): método utilizado para copiar una carpeta el cual solamente envia la carpeta a copiar al backend y ahi se realiza la logica.
- abriCarpeta(): método utilizado para cambiar el “path” actual del usuario que simula el cambio entre carpetas.
- eliminarCarpeta(): método utilizado para eliminar una carpeta el cual solamente envia la carpeta a eliminar al backend y ahi se realiza la logica:
- regresar(): método utilizado para poder manejar el boton de regresar el cual solamente va cambiando el path anterior
- edit_new-File(): método utilizado para mostrar el editor de codigo
- buscarArchivos(): método utilizado para buscar los archivos del usuario dado el path donde se encuentra actualmente

    - clickArchivo(): método utilizado para mostrar las opciones al usuario que se pueden realizar con los archivos, y enviar a su respectivo método
    - abrirArchivo(): método utilizado para mostrar el editor de codigo con el contenido del archivo que se selecciono
    - eliminarArchivo(): método utilizado para eliminar el archivo
    - compartirArchivo(): método utilizado para compartir un archivo
    - añadirAPapelera: método utilizado para para añadir a papelera de reciclaje el archivo eliminado
    - hacerCopia(): método utilizado para hacer la copia del archivo
    - moveFile(): método utilizado para guardar el archivo que se va a mover en el local storage y activar el boton de mover aqui:
    - moverArchivo(): método utilizado para mover el archivo seleccionado o puesto en el local storage al path actual
    - moverCarpeta(): igual que el mover archivo pero con la carpeta
    - moveDirectory(): método utilizado para guardar los datos de la carpeta en el local storage y activar el boton
- Shared: contiene metodos necesarios para manejar los archivos compartidos:

    - formatearFecha(): método utilizado para formatear la fecha de la bd y que se vea en formato yyyy-MM-dd
    - VerArchivoCompartido(): método utilizado para mostrar el contenido del archivo en el editor, solamente para verlo
    - EliminarArchivoCompartido(): método utilizado para eliminar de forma permanente el archivo compartido

Carpeta Service: contiene los servicios para realizar todas las distintas peticiones a la base de datos:

- Admin-Options: Servicio utilizado para las opciones (peticiones) exclusivas del administrador
- User-Options: Servicio utilizado para las opciones (peticiones) exclusivas del usuario, y las peticiones utilizadas para la gestión de archivos y carpetas

BACKEND

Carpeta Controllers: contiene los metodos utilizados para la gestion de las peticiones a la base de datos de:

- ArchivosController
- CarpetasController
- PapeleraController
- SharedController
- UsersController

Carpeta Models: contiene los modelos utilizados para la gestion de los objetos para la base de datos siendo (JS):

- Archivo
- Carpeta
- Papelera
- Shared
- User

Carpeta Routes: contiene las rutas de las peticiones para la gestion de la base de datos:

- Archivos_Routes
- Carpetas_Routes
- Papelera_Routes
- Shared_Routes
- User_Routes