



CENTRO NACIONAL COLOMBO ALEMAN.

Barranquilla, Atlántico.

TECNOLOGO EN ANALISIS Y DESARROLLO DE SOFTWARE.

Ficha: (2547385).

SENA

Instructor

BELKIS MILENA GUELL MUÑOZ

Aprendiz

SEBASTIAN VELEZ BLANDON

URRAO ANTIOQUIA

2023

Contenido

Actividad a realizar	3
GA7-220501096-AA2-EV01	3
Solución de la actividad	4
Introducción	4
Repositorio github.com	4
fotografía de código fuente	4
 Captura de pantalla 1 código de la conexión a la base de datos	5
Captura de pantalla 2 código de conexión al api de Google drive	5
Captura de pantalla 3 código de inserción de empleado nuevo a la base de datos, la fotografía almacenada en Google drive	6

Actividad a realizar

GA7-220501096-AA2-EV01

codificación de módulos del software según requerimientos del proyecto

Teniendo en cuenta las características del software a desarrollar realizar la codificación del módulo del proyecto realizando conexiones con bases de datos por medio de JDBC tomando como referencia lo visto en el componente formativo “Construcción de aplicaciones con JAVA”.

Elementos para tener en cuenta:

- Para la codificación del módulo debe tener en cuenta los artefactos del ciclo del software realizados con anterioridad: diagrama de clases, diagramas de casos de uso, historias de usuario, diseños, prototipos, Informe técnico de plan de trabajo para construcción de software con tecnologías seleccionadas etc.
- Se debe crear el proyecto utilizando herramientas de versionamiento.
- El código debe cumplir con estándares de codificación como:
 - Nombramiento de variables
 - Nombramiento de métodos
 - Nombramiento de clases
 - Nombramiento de paquetes
 - Debe tener funcionalidades de inserción, consulta, actualización y eliminación.

Lineamientos generales para la entrega de la evidencia:

- **Productos para entregar:** carpeta comprimida que debe tener los siguientes archivos:
 - archivos del proyecto,
 - archivo con enlace del repositorio, la carpeta comprimida debe tener el nombre del aprendiz y número de la evidencia así: NOMBREAPELLIDO_AA2_EV01
 - **Extensión:** ZIP, RAR.
- Para hacer el envío de la evidencia remítase al área de la actividad correspondiente y acceda al espacio:
 - codificación de módulos del software según requerimientos del proyecto GA7-220501096-AA2-EV01.

Solución de la actividad

Introducción

En este documento comparto parte del código fuente del proyecto que estoy realizando que se trata de la administración de los empleados en las empresas, este software pretende gestionar los procesos de administrativos de los empleados como lo es registro de las incapacidades. El software se esta realizando en angular.js y en node.js frameworks del lenguaje de JavaScript.

Repositorio github.com

A continuación, comparto el repositorio del proyecto EmpleAdminis, en este repositorio se encuentra todo el código fuente del software que estoy desarrollando.

Frontend en angular.js

https://github.com/SEBASVELEZBLANDON07/EmpleAdminis_angularjs

backend en node.js

https://github.com/SEBASVELEZBLANDON07/EmpleAdminis_nodejs

fotografía de código fuente

A continuación, se muestran imágenes del código fuente más importantes como es la conexión a la base de datos, la conexión con la api de Google drive, la inserción de los empleados a la base de datos.

```

JS conexion_dbjs
database > JS conexion_dbjs > ...
1 //se llaman las variables globales
2 require('dotenv').config();
3
4 const mysql = require('mysql');
5
6 //se configura la conexión a la base de datos
7 var conexiondb=mysql.createConnection({
8   port: process.env.DB_PORT,
9   host: process.env.DB_HOST,
10  user: process.env.DB_USERNAME,
11  password: process.env.DB_PASSWORD,
12  database: process.env.DB_NAME
13 });
14
15 conexiondb.connect(function(error){
16   if(error){
17     throw error;
18   }else{
19     console.log('conexión exitosa');
20   }
21 });
22
23
24 module.exports = conexiondb;
25
26
27
28
29
30
31
32
Ln 38, Col 1 Spaces: 4 UTF-8 CRLF JavaScript Go Live II Ninja Prettier
05:06 p. m. 01/12/2023

```

Captura de pantalla 1 código de la conexión a la base de datos

```

JS api_driverjs
services > JS api_driverjs > [unknown]
1 require('dotenv').config();
2
3 const { google } = require("googleapis");
4 const path = require("path");
5 const stream = require("stream");
6 const mime = require('mime');
7
8 // Módulo para operaciones de sistema de archivos
9 const fs = require("fs");
10
11 //variables globales
12 const KEYFILEPATH = process.env.GOOGLE_DRIVE_KEYFILEPATH;
13 const SCOPES = process.env.GOOGLE_DRIVE_SCOPES;
14 const ID_folder = process.env.GOOGLE_DRIVE_FOLDER_ID;
15 //const filePath = './controllers/logo_logo.png';
16
17 //se autentifica la verificación
18 const auth = new google.auth.GoogleAuth({
19   keyFile: KEYFILEPATH,
20   scopes: SCOPES,
21 });
22
23 //función para llamar a la api de driver para subir la imagen a driver
24 const uploadFile = async (fileObject) => {
25
26   const bufferStream = new stream.PassThrough();
27   bufferStream.end(fileObject.buffer);
28   try {
29     const { data } = await google.drive({ version: "v3", auth }).files.create({
30       media: {
31         mimeType: fileObject.mimeType,
32         body: bufferStream,
33       },
34     });
35   } catch (error) {
36     console.log(error);
37   }
38 }
Ln 156, Col 16 Spaces: 2 UTF-8 CRLF JavaScript Go Live II Ninja Prettier
05:07 p. m. 01/12/2023

```

Captura de pantalla 2 código de conexión al api de Google drive



```
File Edit Selection View Go Run ... backend
insertEmpleados.js M X
routes > JS insertEmpleados.js > router.post('/incapacidad') callback > conexion.query() callback
70
71 //ingresar los datos personales del empleado con la fotografia del empleado
72 router.post('/InEmpleado', auth.authenticateToken, checkRole.check_Role, uploadimg.single('imagen'), (req, res) => {
73   let inset_empleado = req.body;
74
75   //se verifica si no hay otro empleado con ese id
76   var query = "SELECT id_cedula FROM empleado WHERE id_cedula = ?";
77   conexion.query(query, [inset_empleado.id_cedula], (err, results) =>{
78
79     if(!err){
80
81       if(results.length <= 0){
82
83         //se busca la id de la empresa a donde se va a ingresar el empleado
84         query = "SELECT id_empresa FROM empresa WHERE nom_empresa = ?";
85         conexion.query(query, [inset_empleado.nom_empresa], (err, results) =>{
86
87           if(!err){
88             //console.log("paso el error ")
89
90             if(results.length >= 0){
91               const imagen = req.file;
92
93               //se verifica si hay un file llamado imagen
94               if (!imagen) {
95                 return res.status(403).json({ message: "No se subió ningún archivo" });
96               }else{
97
98                 //se guarda la imagen en filePath
99                 const filePath = imagen.path;
100
101                 //se define el fileobject
102                 const fileData = {
103                   buffer: fs.readFileSync(filePath),
104                   mimeType: 'image/jpeg',
105                   originalname: inset_empleado.id_cedula + '-fotografia-Nombre-' + inset_empleado.nombre,
106                 };
107
108                 //se procesa la imagen, se envia asia la api de driver para subirla a driver
109                 apiDriver.uploadFile(fileData).then((imageUrl) => {
110
111                   //se guarda la url de la imagen guardada
112                   const fotografia = imageUrl;
113                   console.log("URL de la imagen en Google Drive:", imageUrl);
114
115                   //se relaciona el id de la empresa
116                   const id_empresa_e = results[0].id_empresa;
117                   console.log(id_empresa_e);
118
119                   //se insertan los datos del empleado a la base de datos
120                   query = "INSERT INTO 'empleado'('id_cedula', 'tipo_documento', 'nombre', 'apellidos', 'fecha_nacimiento', 'pais', 'num_contacto') VALUES (?, ?, ?, ?, ?, ?, ?)";
121                   conexion.query(query, [inset_empleado.id_cedula, inset_empleado.tipo_documento, inset_empleado.nombre, inset_empleado.apellidos, inset_empleado.fecha_nacimiento, inset_empleado.pais, inset_empleado.num_contacto], (err, results) =>{
122
123                     if (!err){
124                       return res.status(200).json({message:"Función realizada con éxito"});
125                     }else{
126                       return res.status(500).json(err);
127                     }
128                   })
129                 })
130               .catch((error) => {
131                 console.error('Error al cargar la imagen:', error);
132                 return res.status(500).json(err);
133               });
134             }else{
135               return res.status(402).json({message: "Empresa no ingresada"});
136             }
137           }else{
138             return res.status(500).json(err);
139           }
140         }
141       }else{
142         return res.status(401).json({message: "Empleado ya ingresada"});
143       }
144     }else{
145       return res.status(500).json(err);
146     }
147   });
148 });
149
150
151 //se ingresa los datos de la cuenta bancaria
152 router.post('/cuenBancaria', auth.authenticateToken, checkRole.check_Role, (req, res) =>{
```

Captura de pantalla 3 código de inserción de empleado nuevo a la base de datos, la fotografía almacenada en Google drive