



TECNOLÓGICO DE ESTUDIOS SUPERIORES DE ECATEPEC.

INGENIERIA EN SISTEMAS COMPUTACIONALES.

ALUMNOS:

- GARCIA CERVANTES ANGEL RAFAEL
- GARCIA ROJAS RODOLFO (responsable)

PROFESOR: CORTES BARRERA GRISELDA.

MATERIA: BASE DE DATOS PARA DISPOSITIVOS
MOVILES

GRUPO: 5801

“MANUAL DE USUARIO.”

MANUAL DE USUARIO

Índice

Análisis.....	3
Modulo Login/Recuperar/Registro	3
Requerimientos y necesidades	3
Requerimientos Login	3
Requerimientos recuperar contraseña	3
Requerimientos y necesidades registro empleado.....	3
Requerimiento y necesidades registro invitado.	3
Problemática	4
Objetivos generales.....	4
Objetivos específicos.....	4
Modelo Entidad-Relación.....	5
Base de Datos.....	5
Pruebas Postman	16
Diseño.....	21
Login	21
Recuperar contraseña	22
GitHub	24

Análisis

Modulo Login/Recuperar/Registro

Requerimientos y necesidades

- El sistema permitirá la visualización del módulo Login mediante un botón o sección el menú.
- El sistema permitirá el ingreso a los usuarios registrados en el sistema, de acuerdo con su perfil.
- El sistema permitirá el cambio de contraseña por medio un correo registrado.
- El sistema permitirá el registro de una cuenta con datos obligatorios completos.

Requerimientos Login

- Se requiere llenar el formulario para el inicio de sesión.
- El usuario ingresara a el sistema por medio de un correo y contraseña registrados en la base de datos.

Requerimientos recuperar contraseña

- Se requiere de un correo registrado en la base de datos para poder recuperar la contraseña.

Requerimientos y necesidades registro empleado

- Se requiere llenar el formulario completo para registrar al empleado.
- Se requiere de un id de empleado (forma única de identificación de empleado por medio de la empresa), este debe de ser dado cuando se contrata al empleado, el cual debe de estar dado de alta en la base de datos con sus datos del empleado.
- Se requiere de un correo electrónico.
- Se requiere de una contraseña de al menos 8 caracteres.

Requerimiento y necesidades registro invitado.

- Se requiere llenar el formulario completo para registrar el invitado.
- Se requiere un correo electrónico.
- Se requiere de una contraseña de al menos 8 caracteres.
- Se requiere de la selección de un estado para que se habiliten los municipios.
- Se requiere de un teléfono con 10 dígitos.

- Se requiere de una Clave única de registro de población (CURP) valido.

Problemática

La falta de un módulo de ingreso al sistema hace que la información pueda ser manipulada por personas ajenas y/o usuarios del sistema, dado que la información es sensible y confidencial ya que se cuenta con datos personales de cada usuario e información relacionada con la empresa.

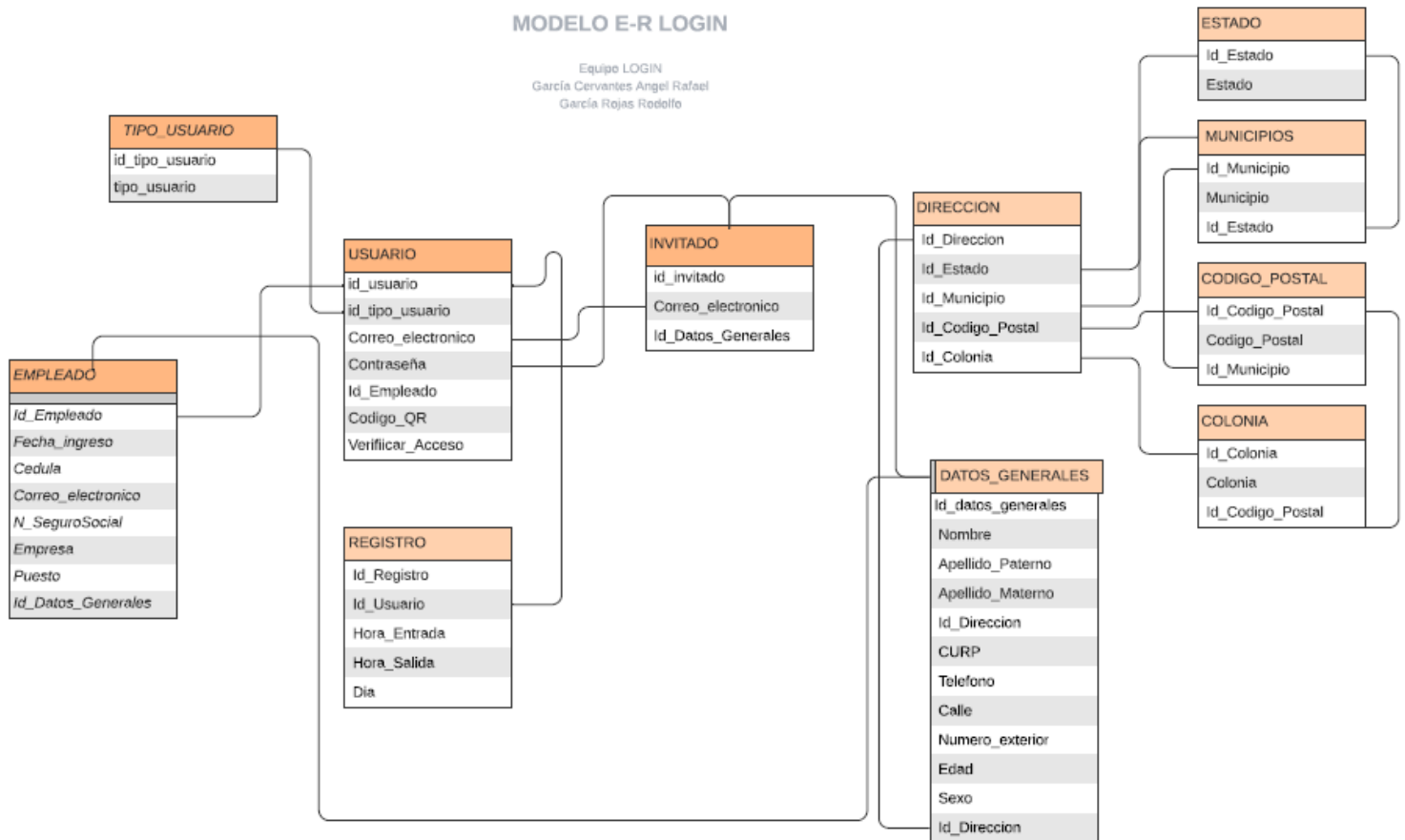
Objetivos generales

1. Desarrollar un componente que autorice el acceso a los usuarios registrados.
2. Elaborar un componente que apruebe la creación de una cuenta.
3. Llevar a cabo un componente que permita recuperar la cuenta.

Objetivos específicos

- Analizar el proceso de inicio de sesión y registro de usuarios.
- Determinar los requerimientos y necesidades.
- Diseñar la interfaz gráfica.
- Implementar la interfaz gráfica.

Modelo Entidad-Relación



Base de Datos

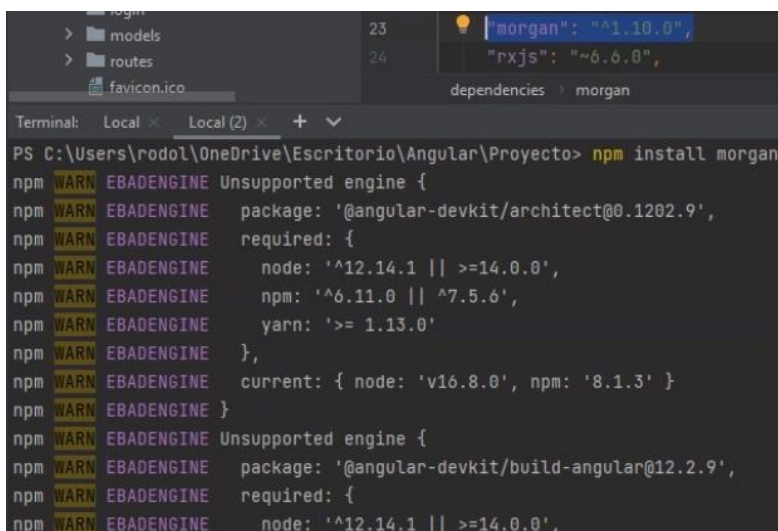
Se instaló Express con la versión 4.17.2

```

PS C:\Users\rodol\OneDrive\Escritorio\Angular\Proyecto> npm install express
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/architect@0.1202.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
npm WARN EBADENGINE     npm: '^6.11.0 || ^7.5.6',
npm WARN EBADENGINE     yarn: '>= 1.13.0'
npm WARN EBADENGINE   },
npm WARN EBADENGINE   current: { node: 'v16.8.0', npm: '8.1.3' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/build-angular@12.2.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
npm WARN EBADENGINE     npm: '^6.11.0 || ^7.5.6',
npm WARN EBADENGINE     yarn: '>= 1.13.0'
npm WARN EBADENGINE   },
npm WARN EBADENGINE   current: { node: 'v16.8.0', npm: '8.1.3' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/build-optimizer@0.1202.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',

```

Se instaló Morgan con la versión 1.10.0

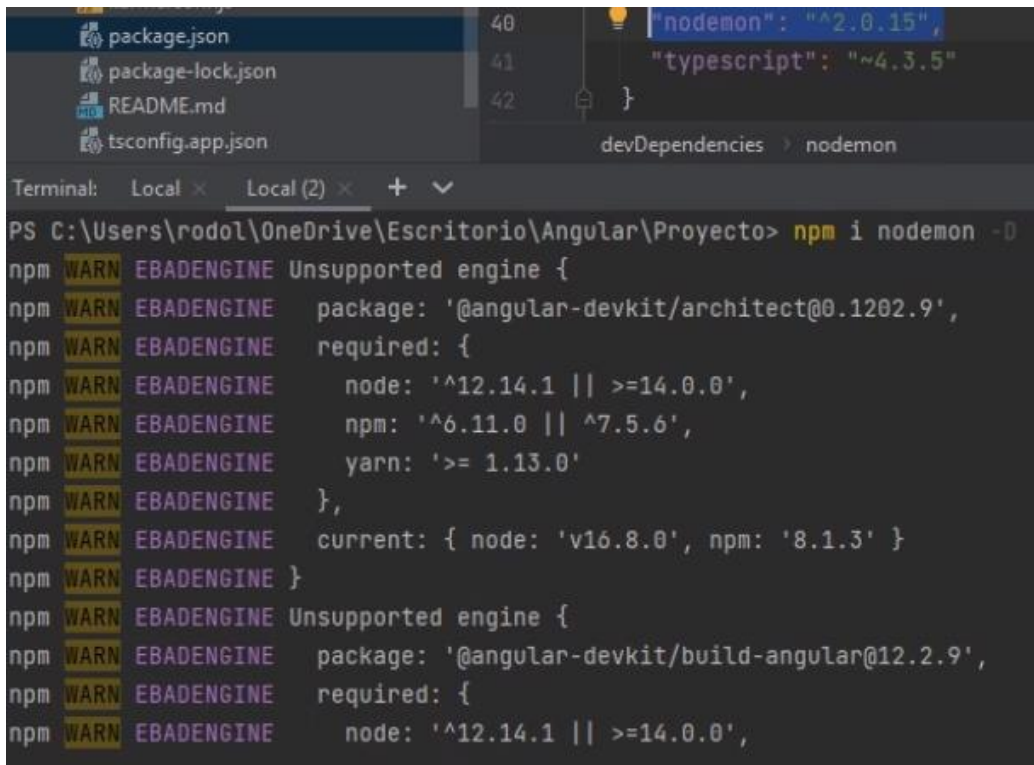


```

PS C:\Users\rodol\OneDrive\Escritorio\Angular\Proyecto> npm install morgan
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/architect@0.1202.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
npm WARN EBADENGINE     npm: '^6.11.0 || ^7.5.6',
npm WARN EBADENGINE     yarn: '>= 1.13.0'
npm WARN EBADENGINE   },
npm WARN EBADENGINE   current: { node: 'v16.8.0', npm: '8.1.3' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/build-angular@12.2.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',

```

Se instaló modemon con la versión 2.0.15



The screenshot shows a VS Code editor with a file explorer on the left containing `package.json`, `package-lock.json`, `README.md`, and `tsconfig.app.json`. The main editor displays the `package.json` file with the following content:

```
{
  "devDependencies": {
    "nodemon": "^2.0.15",
    "typescript": "~4.3.5"
  }
}
```

Below the editor, the terminal shows the command `npm i nodemon -D` being executed. The output consists of several warnings from the npm EBADENGINE:

```
PS C:\Users\rodol\OneDrive\Escritorio\Angular\Proyecto> npm i nodemon -D
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/architect@0.1202.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
npm WARN EBADENGINE     npm: '^6.11.0 || ^7.5.6',
npm WARN EBADENGINE     yarn: '>= 1.13.0'
npm WARN EBADENGINE   },
npm WARN EBADENGINE   current: { node: 'v16.8.0', npm: '8.1.3' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/build-angular@12.2.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
```

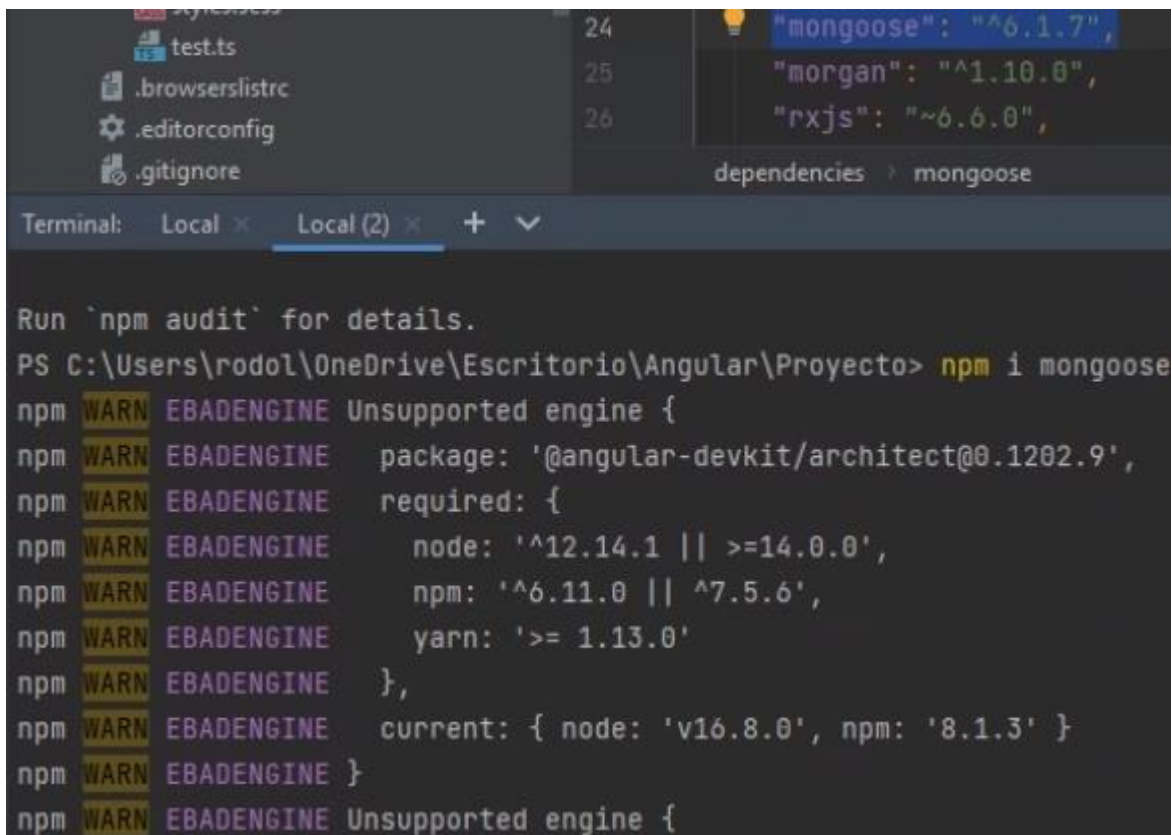
Se agregó el modemon con la ruta `src/index.js`



The screenshot shows a VS Code editor displaying the `package.json` file. The `scripts` section is expanded, showing the following configuration:

```
{
  "name": "proyecto",
  "version": "0.0.0",
  "scripts": {
    "dev": "nodemon src/index.js",
    "ng": "ng",
    "start": "ng serve",
    "build": "ng build",
  }
}
```

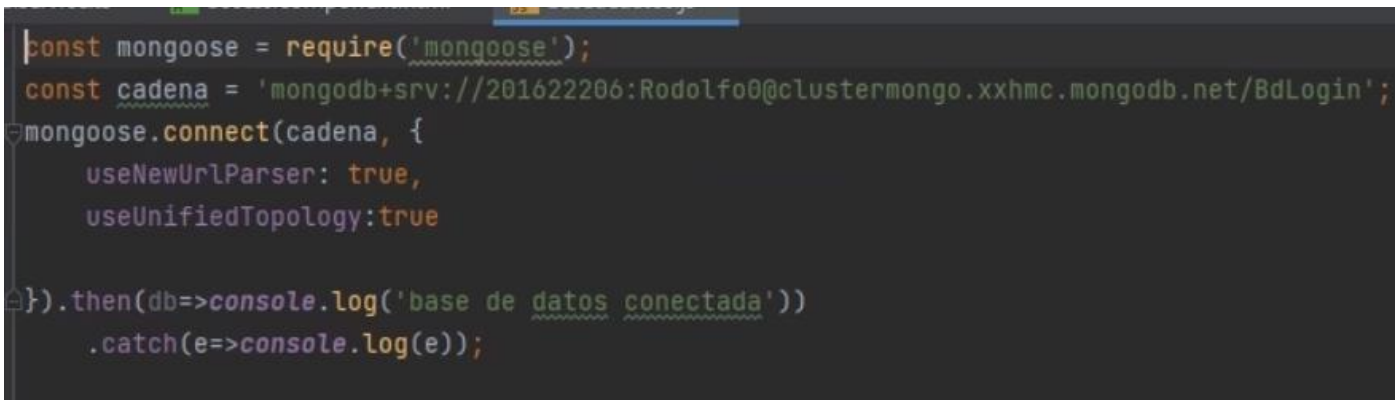
Se instaló la que hace referencia hacia mogodb con la versión 6.1.7



```
test.ts
.browserslistrc
.editorconfig
.gitignore
24 "mongoose": "^6.1.7",
25 "morgan": "^1.10.0",
26 "rxjs": "~6.6.0",
dependencies > mongoose

Terminal: Local x Local (2) x + v
Run 'npm audit' for details.
PS C:\Users\rodol\OneDrive\Escritorio\Angular\Proyecto> npm i mongoose
npm WARN EBADENGINE Unsupported engine {
npm WARN EBADENGINE   package: '@angular-devkit/architect@0.1202.9',
npm WARN EBADENGINE   required: {
npm WARN EBADENGINE     node: '^12.14.1 || >=14.0.0',
npm WARN EBADENGINE     npm: '^6.11.0 || ^7.5.6',
npm WARN EBADENGINE     yarn: '>= 1.13.0'
npm WARN EBADENGINE   },
npm WARN EBADENGINE   current: { node: 'v16.8.0', npm: '8.1.3' }
npm WARN EBADENGINE }
npm WARN EBADENGINE Unsupported engine {
```

Se creo la conexión con la base de datos en caso de tener éxito manda mensaje “base de datos conectada”, si detecta un error manda mensaje con el error encontrado.



```
const mongoose = require('mongoose');
const cadena = 'mongodb+srv://201622206:Rodolfo0@clustermongo.xxhmc.mongodb.net/BdLogin';
mongoose.connect(cadena, {
  useNewUrlParser: true,
  useUnifiedTopology: true
}).then(db=>console.log('base de datos conectada'))
.catch(e=>console.log(e));
```

Se creó un archivo en el cual se hace uso de Express y Morgan, se hace la configuración del puerto que se va a ocupar, se configura el middleware para

el intercambio de información entre la base de datos y la aplicación, por último, se definieron las rutas del api.

```
const express = require('express');
const app = express();
const morgan = require('morgan');
//configuracion
app.set('port', process.env.PORT || 4000);
app.set('json spaces', 2);
require('../configuracion/basededatos');
// middleware
app.use(morgan('dev'));
app.use(express.urlencoded({extended: false}));
app.use(express.json());
//definir rutas
app.use('/api/datos', require('../routes/index'));
app.use('/api/usuario', require('../routes/usuarios'));
//iniciando servidor
app.listen(app.get('port'), ()=>{
  // codigo ascii backstick alt+96 `
  console.log(`El servidor esta en el puerto ${4000}`)
});
```

Se crea el esquema con las variables definiendo su tipo de dato y haciendo mención que el dato es requerido, por último se exporta el esquema con el nombre “usuarios”.

```
1  const {Schema, model} = require('mongoose');
2
3  const esquemausuarios = new Schema( definition: {
4    id_usuario: {
5      type: String,
6      require: true
7    },
8    id_tipo_usuario: {
9      type: String,
10     require: true
11   },
12   correo_electronico: {
13     type: String,
14     require: true
15   },
16   contrasena: {
17     type: String,
18     require: true
19   },
20   id_empleado: {
21     type: String,
22     require: true
23   },
24 }
```

```

        type: String,
        require: true
    },
    contraseña: {
        type: String,
        require: true
    },
    id_empleado: {
        type: String,
        require: true
    },
    fecha_registro: {
        type: Date,
        default: Date.now()
    }
});

module.exports = model( name: 'usuarios', esquemausuarios);

```

Se crea el controlador con el cual se va poder insertar un usuario.

```

const user = require("../models/usuarios");

exports.insertarusuario = async (req,res)=>{
    console.log(req.body);
    const {id_usuario,id_tipo_usuario,correo_electronico,contrasena,id_empleado,fecha_registro} = req.body;
    const nuevousuario = new user( doc: {id_usuario,id_tipo_usuario,correo_electronico,contrasena,id_empleado,fecha_registro});
    await nuevousuario.save();
    res.json(nuevousuario);
}

```

Se va a poder consultar los usuarios, si es que existen, si no, manda un mensaje de error

```
exports.consultausuarios = async (req, res) => {  
  try {  
    const consultarusuario = await user.find();  
    res.json(consultarusuario);  
  }  
  catch (e) { console.log(e);  
    res.status(404).json('error de consulta usuarios'); }  
}
```

Se va poder consultar el usuario por medio de su ID, en caso de que no exista, manda un mensaje de error.

```
exports.consultausuarioid = async (req, res) => {  
  try {  
    const {id_usuario, id_tipo_usuario, correo_electronico, contrasena, id_empleado, fecha_registro} = req.body;  
    console.log(req.params.id);  
    let usuarioid = await user.findById(req.params.id);  
    console.log('Los datos del usuario son:', usuarioid);  
    res.json(usuarioid);  
  }  
  catch (e) { res.send('error de consulta usuario id', e); }  
}
```

Se va a poder actualizar el usuario, siempre y cuando exista, de lo contrario mandará un mensaje de error.

```
exports.actualizarusuario = async (req, res) => {
  try {
    const {id_usuario, id_tipo_usuario, correo_electronico, contrasena, id_empleado, fecha_registro} = req.body;
    console.log(req.params.id);
    let usuarioid = await user.findById(req.params.id);
    console.log('Los datos del usuario son:', usuarioid);
    res.json(usuarioid);
    if (!usuarioid) {
      res.status(404).json('no existe usuario');
    } else {
      usuarioid.id_usuario = id_usuario;
      usuarioid.id_tipo_usuario = id_tipo_usuario;
      usuarioid.correo_electronico = correo_electronico;
      usuarioid.contrasena = contrasena;
      usuarioid.id_empleado = id_empleado;
      usuarioid = await user.findOneAndUpdate(
        { _id: req.params.id },
        usuarioid,
        { new: true }
      );
      res.json(usuarioid);
    }
  }
}
```

Se va a poder eliminar el usuario siempre y cuando exista, de lo contrario mandará un mensaje de error.

```
exports.eliminarusuario = async (req, res) => {
  try {
    console.log(req.params.id);
    let usuarioid = await user.findById(req.params.id);
    console.log('Los datos del usuario son:', usuarioid);
    res.json(usuarioid);
    if (!usuarioid) {
      res.status(404).json('no existe usuario');
    } else {
      usuarioid = await user.findOneAndDelete(
        { _id: req.params.id }
      );
      res.json('usuario eliminado');
    }
  } catch (e) {
    console.log(e);
    res.status(404).json('error de consulta usuario id');
  }
}
```

Se va a mandar llamar al método consultar (get).

```
const {Router} = require('express');
const router = Router();
/*const d = require('../routes/datos.json');*/
const d = require('../routes/login.json');
const _ = require('underscore');
console.log(d)

//Consultar
router.get( path: '/', handlers: (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> ) =>{
    res.json(d);
});
```

Se va a mandar llamar al método agregar (post).

```
//Agregar registro
router.post( path: '/', handlers: (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> )=>{
    console.log(req.body );
    const {id_usuario, id_tipo_usuario, correo_electronico, contrasena} = req.body;
    if (id_usuario && id_tipo_usuario && correo_electronico && contrasena)
    {
        const nuevoregistro = {...req.body };
        d.push(nuevoregistro);
        res.json(d);
    }
    else
    {
        res.send( body: 'Error al hacer la peticion');
    }
});
```

Se va a mandar llamar al método eliminar (delete).

```
//eliminar
router.delete ( path:('/:id_usuario', handlers: (req : Request<P, ResBody, ReqBody, ReqQuery, Locals> , res : Response<ResBody, Locals> )=>{
    const {id_usuario} = req.params;
    _.each(d, iteratee: (registros,i)=>{
        if(registros.id_usuario == id_usuario)
        {
            d.splice(i,1);
            res.json(d);
        }
        else
        {
            res.send( body: 'Error al hacer la peticion');
        }
    });
    res.send( body: 'Eliminado');
});
```


Se va a mandar llamar al método eliminar (put).

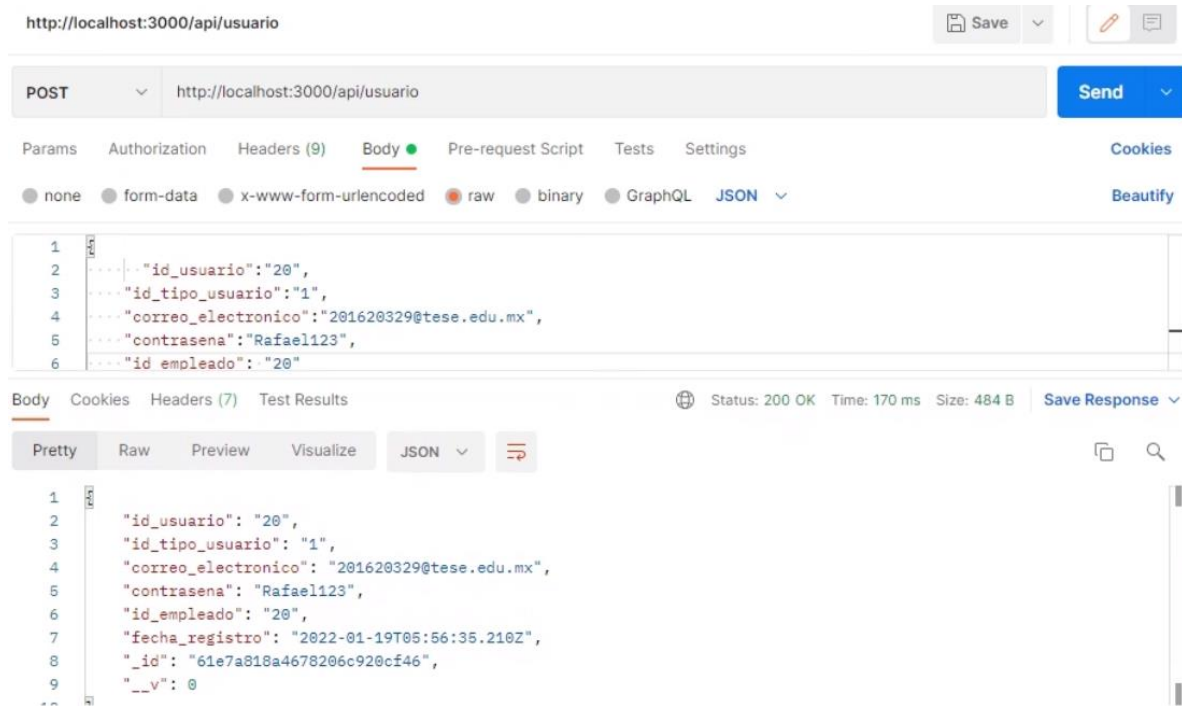
```
//modificar
router.put( path:('/:id_usuario'), handlers: (req : Request<P, ResBody, ReqBody, ReqQuery, Locals>, res : Response<ResBody, Locals>) =>{
  const {id_usuario} = req.params;
  const {id_tipo_usuario, correo_electronico, contrasena} = req.body;
  if(id_tipo_usuario && correo_electronico && contrasena)
  {
    _.each(d, iteratee: (registros,i)=>{
      if(registros.id_usuario == id_usuario)
      {
        registros.id_tipo_usuario = id_tipo_usuario;
        registros.correo_electronico = correo_electronico;
        registros.contrasena = contrasena;
        res.json(d);
      }
    })
  }
});
module.exports = router;
```

Se ejecuta el proyecto y se muestra los mensajes siguientes

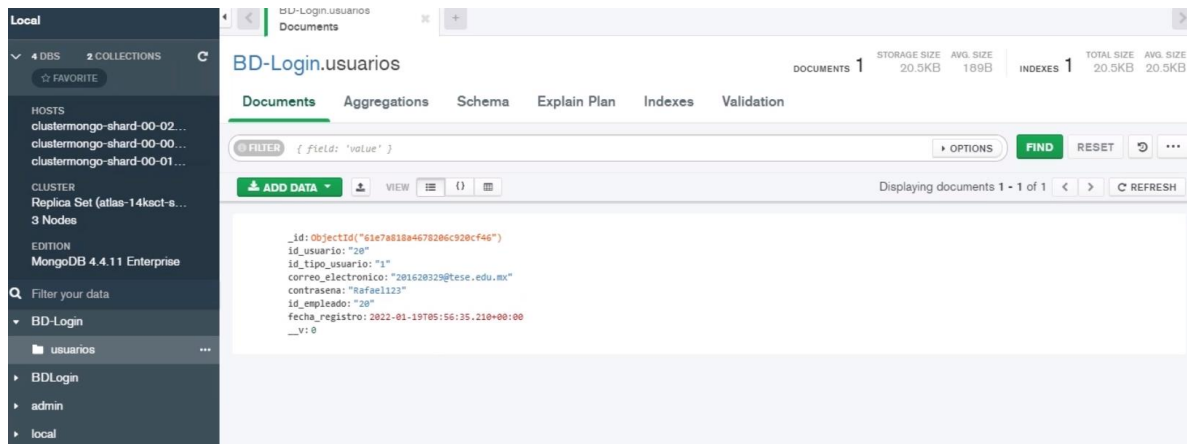
```
]
El servidor esta en el puerto 3000
base de datos conectada
```

Pruebas Postman

Se realizó la prueba con post (insertar), donde se agregó con éxito el registro.



Dentro de MongoDB se visualiza con éxito que se agregó el registro.



Se realizó la prueba con get (consultar usuarios), donde se visualizan con éxito los registros.

The screenshot displays two applications used for testing a REST API. The top application is MongoDB Compass, showing the 'BD-Login.usuarios' collection with two documents. The bottom application is Postman, showing a GET request to 'http://localhost:3000/api/usuario/' returning a 200 OK status with a JSON response containing two user records.

MongoDB Compass Data:

Document	Content
1	<pre>{ "_id": "61e7a818a4678206c920cf46", "id_usuario": "21", "id_tipo_usuario": "2", "correo_electronico": "201622206@tese.edu.mx", "contrasena": "Rodolfo123", "id_empleado": "21", "fecha_registro": "2022-01-19T05:56:35.210+00:00", "__v": 0 }</pre>
2	<pre>{ "_id": "61e7a9eaa4678206c920cf49", "id_usuario": "21", "id_tipo_usuario": "2", "correo_electronico": "201622206@tese.edu.mx", "contrasena": "Rodolfo123", "id_empleado": "21", "fecha_registro": "2022-01-19T05:56:35.210+00:00", "__v": 0 }</pre>

Postman Request:

Method: GET
URL: http://localhost:3000/api/usuario/

Postman Response (JSON):

```
[
  {
    "_id": "61e7a818a4678206c920cf46",
    "id_usuario": "20",
    "id_tipo_usuario": "1",
    "correo_electronico": "201620329@tese.edu.mx",
    "contrasena": "Rafael123",
    "id_empleado": "20",
    "fecha_registro": "2022-01-19T05:56:35.210Z",
    "__v": 0
  },
  {
    "_id": "61e7a9eaa4678206c920cf49",
    "id_usuario": "21",
    "id_tipo_usuario": "2",
    "correo_electronico": "201622206@tese.edu.mx",
    "contrasena": "Rodolfo123",
    "id_empleado": "21",
    "fecha_registro": "2022-01-19T05:56:35.210Z",
    "__v": 0
  }
]
```

Se realizó la prueba con get (consultar usuario por id), donde se visualiza con éxito el registro.

```
http://localhost:3000/api/usuario/61e7a818a4678206c920cf46
```

GET http://localhost:3000/api/usuario/61e7a818a4678206c920cf46

Params Authorization Headers (9) Body ● Pre-request Script Test:

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON ↻

```
1  {
2    "_id": "61e7a818a4678206c920cf46",
3    "id_usuario": "20",
4    "id_tipo_usuario": "1",
5    "correo_electronico": "201620329@tese.edu.mx",
6    "contrasena": "Rafael123",
7    "id_empleado": "20",
8    "fecha_registro": "2022-01-19T05:56:35.210Z",
9    "__v": 0
10 }
```

Se realizó la prueba con put (modificar), donde se visualiza con éxito el cambio en el registro.

http://localhost:3000/api/usuario/61e7a818a4678206c920cf46

PUT http://localhost:3000/api/usuario/61e7a818a4678206c920cf46

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "id_usuario": "21",
3   "id_tipo_usuario": "1",
4   "correo_electronico": "rodolfo@tese.edu.mx",
5   "contrasena": "Rodolfo_Garcia",
6   "id_empleado": "21"
7 }
```

Body Cookies Headers (7) Test Results Status: 200 OK

Pretty Raw Preview Visualize JSON

```
1 {
2   "_id": "61e7a818a4678206c920cf46",
3   "id_usuario": "21",
4   "id_tipo_usuario": "2",
5   "correo_electronico": "201622206@tese.edu.mx",
6   "contrasena": "Rodolfo123",
7   "id_empleado": "21",
8   "fecha_registro": "2022-01-19T05:56:35.210Z",
9   "__v": 0
10 }
```

Local BD-Login.usuarios Documents

BD-Login.usuarios DOCUMENTS 2

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

```
_id: ObjectId("61e7a818a4678206c920cf46")
id_usuario: "21"
id_tipo_usuario: "1"
correo_electronico: "rodolfo@tese.edu.mx"
contrasena: "Rodolfo_Garcia"
id_empleado: "21"
fecha_registro: 2022-01-19T05:56:35.210+00:00
__v: 0
```

```
_id: ObjectId("61e789e8a4678206c920cf49")
id_usuario: "21"
id_tipo_usuario: "2"
correo_electronico: "201622206@tese.edu.mx"
contrasena: "Rodolfo123"
id_empleado: "21"
fecha_registro: 2022-01-19T05:56:35.210+00:00
__v: 0
```

Se realizó la prueba con delete (eliminar), donde se visualiza con éxito la eliminación del registro.

http://localhost:3000/api/usuario/61e7a818a4678206c920cf46

DELETE http://localhost:3000/api/usuario/61e7a818a4678206c920cf46

Params Authorization Headers (9) **Body** Pre-request Script Tests Settings

none form-data x-www-form-urlencoded raw binary GraphQL **JSON**

```
1 {
2   ... "id_usuario": "21",
3   ... "id_tipo_usuario": "1",
4   ... "correo_electronico": "rodolfo@tese.edu.mx",
5   ... "contrasena": "Rodolfo_Garcia",
6   ... "id_empleado": "21"
```

Body Cookies Headers (7) Test Results Status: 200 OK Tin

Pretty Raw Preview Visualize **JSON**

```
1 {
2   "_id": "61e7a818a4678206c920cf46",
3   "id_usuario": "21",
4   "id_tipo_usuario": "1",
5   "correo_electronico": "rodolfo@tese.edu.mx",
6   "contrasena": "Rodolfo_Garcia",
7   "id_empleado": "21",
8   "fecha_registro": "2022-01-19T05:56:35.210Z",
9   "__v": 0
10 }
```

Local BD-Login.usuarios

4 DBS 2 COLLECTIONS

☆ FAVORITE

HOSTS
clustermongo-shard-00-02...
clustermongo-shard-00-00...
clustermongo-shard-00-01...

CLUSTER
Replica Set (atlas-14ksc-t...
3 Nodes

EDITION
MongoDB 4.4.11 Enterprise

Q Filter your data

BD-Login

usuarios

BD-Login.usuarios DOCUMENTS 1

Documents Aggregations Schema Explain Plan Indexes Validation

FILTER { field: 'value' }

ADD DATA VIEW

```
_id: ObjectId("61e7a818a4678206c920cf49")
id_usuario: "21"
id_tipo_usuario: "2"
correo_electronico: "201622206@tese.edu.mx"
contrasena: "Rodolfo123"
id_empleado: "21"
fecha_registro: 2022-01-19T05:56:35.210+00:00
__v: 0
```

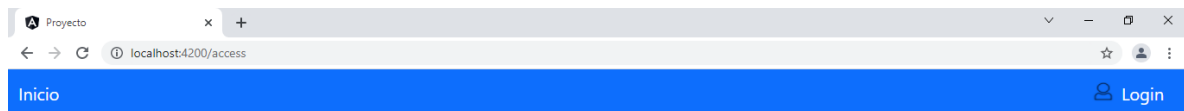

Diseño

Para visualizar el inicio de sesión (Login), mediante un botón colocado en la sección del menú en la parte derecha, se debe hacer clic para mostrar el contenido.



Login

Se cuenta con un título Iniciar sesión, un formulario que solicita el correo electrónico y contraseña, un botón para iniciar sesión, después del formulario se tiene un enlace ¿Has olvidado tu contraseña? El cual permite acceder a Recuperar contraseña, debajo de este, se encuentra el enlace para Crear cuenta el cual permite acceder a Registro.



Iniciar sesión

Correo electronico
Contraseña
Iniciar sesión

[¿Has olvidado tu contraseña?](#)

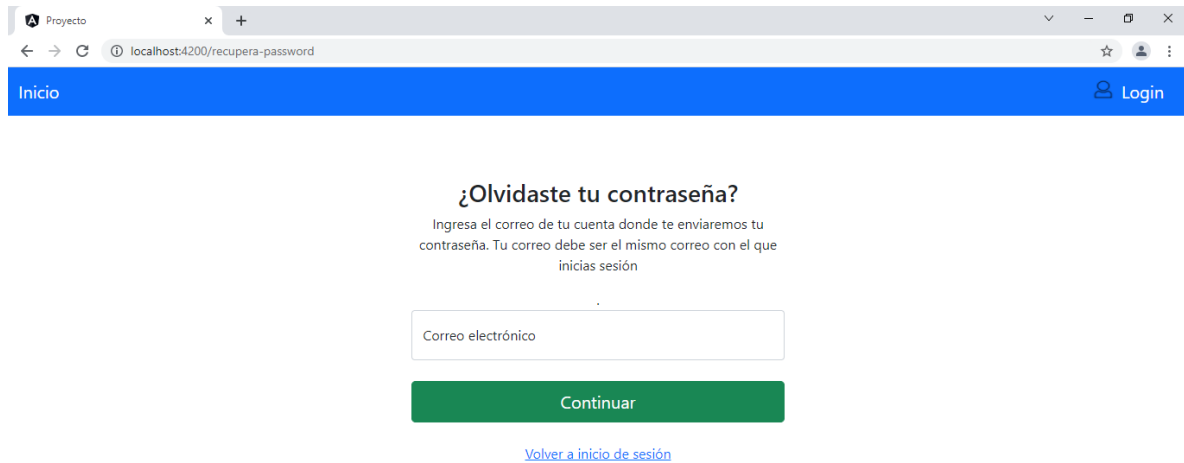
[¿No tienes cuenta?](#)

[Crear cuenta como empleado](#)

[Crear cuenta como invitado](#)

Recuperar contraseña

Se cuenta con un título para distinguir la pantalla **¿Olvidaste tu contraseña?**, texto con instrucciones, un formulario para ingresar el correo electrónico, un botón para continuar y un enlace para volver a la página de inicio de sesión.

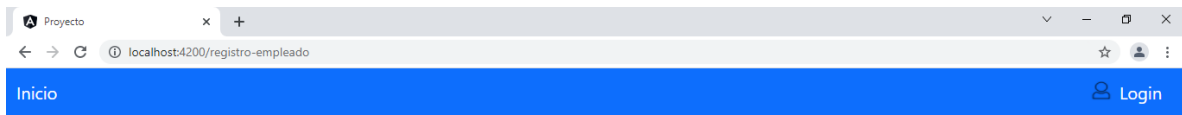


The screenshot shows a web browser window with the title 'Proyecto'. The address bar displays 'localhost:4200/recupera-password'. The page has a blue header with 'Inicio' on the left and a 'Login' button on the right. The main content area is white and contains the following elements:

- ¿Olvidaste tu contraseña?**
- Text: 'Ingresa el correo de tu cuenta donde te enviaremos tu contraseña. Tu correo debe ser el mismo correo con el que inicias sesión'
- A text input field with the placeholder 'Correo electrónico'.
- A green button labeled 'Continuar'.
- A blue link labeled 'Volver a inicio de sesión'.

Registro Empleado

Se cuenta con un título 'Crear una cuenta', un formulario donde se debe ingresar el id empleado, el correo electrónico, contraseña y un botón para registrarse.



Crear una cuenta

Empleado

ID empleado
Correo electronico
Contraseña
Registrarse

Registro Invitado

Se cuenta con un título Crear una cuenta, un subtítulo indicando que, para invitado, un formulario donde se debe ingresar el correo electrónico, contraseña, datos generales como nombre(s), apellido paterno, apellido materno, una dirección en donde se tiene que elegir el estado para que muestre el municipio, el código postal, colonia, calle, número exterior e interior, un número de teléfono, un CURP y un botón para registrarse.

Crear una cuenta	
Invitado	
Correo electronico	
Contraseña	
Nombre(s)	
Apellido Paterno	
Apellido Materno	
Dirección	
Estados	Selecciona un estado...
Municipio	
Código Postal	

Proyecto x +

localhost:4200/registro

Estado:

Municipio:

Código Postal

Colonia

Calle

Número exterior

Número interior

Teléfono

CURP

Instrucciones:

Para iniciar sesión se debe ingresar el correo electrónico y la contraseña, en caso de no tener estos datos, deberá hacer clic en Crear cuenta; en caso de no recordar la contraseña puede hacer clic en ¿Has olvidado tu contraseña?

En la sección de Crear cuenta, se debe de elegir si es para invitado o empleado, una vez que se elige una opción se muestra un formulario donde se tiene que llenar toda la información que se requiere, finalmente dar clic en el botón de registrarse.

En la sección de Recuperar cuenta, se debe ingresar el correo electrónico y dar clic al botón para recuperar la contraseña.

GitHub

Actualización de versión.

Se conecta con GitHub mediante el nombre del equipo en este caso es Equipo Login y el correo que esta dado de alta por el equipo de control de versiones.

```
MINGW64:/c/Users/rodol

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ git config --global user.name "Equipo Login"

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ git config -global user.email 201622206@tese.edu.mx
error: did you mean '--global' (with two dashes)?

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ git config --global user.email 201622206@tese.edu.mx

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ |
```

Se genera la clave ssh

```
MINGW64:/c/Users/rodol

$ ssh-keygen -o
Generating public/private rsa key pair.
Enter file in which to save the key (/c/Users/rodol/.ssh/id_rsa): Login
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in Login
Your public key has been saved in Login.pub
The key fingerprint is:
SHA256:Nr3IWf6Qx/2bphLm+xpAZ6tlaCVA2of+uQhXpdqR/PA rodol@LAPTOP-VH79P3BE
The key's randomart image is:
+----[RSA 3072]-----+
|
|  .o
| o o o .
| . o * B .
| . + & . .
| . S * .
| % #o. o
| . + OoE. +
| o . .o. o
| . . o=o
+----[SHA256]-----+

rodol@LAPTOP-VH79P3BE MINGW64 ~
$
```

Se obtiene la clave publica

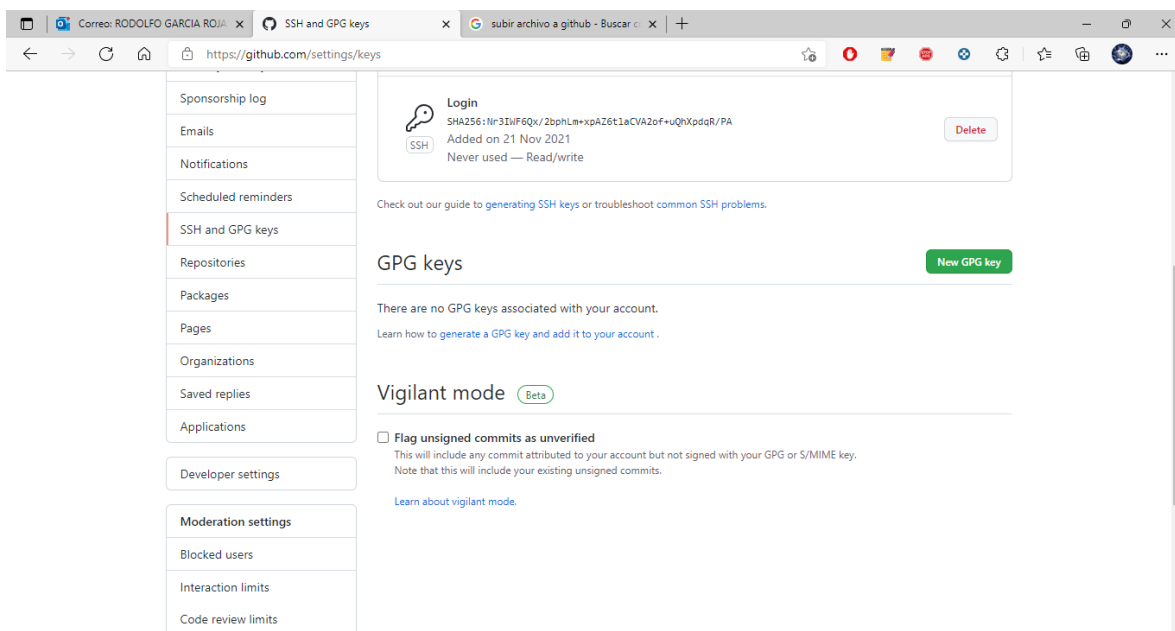
```
MINGW64:/c/Users/rodol

.0
o o o .
. o * B .
. + & . .
. S * .
% #o. o
. + OoE. +
o . .o. o
. . o=o
+----[SHA256]-----+

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ cat Login.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC3Lh7J6aIYSBgw1QV00ASKhZSNhU/v5Z9eV4CQn0sm
ZnWdd7TdcRAvNsVDw9HC9800rb8X8mD2Q7LtQ4R2jJbwzKhR0p/cFrHAJhRen2x62GdmpEadsk/Q60mN
OCbPi6G7MhcQdNcvhYRFm6Wp7za5ax2IoN/4p7mC1L6TUUKN4GvsK3qZGLA5cvmqqBFM0i1KbF/6+UyP
1VYiFpuM02SpC9+y1MiE+iG2HVDuqZt3CQt1817c84z1CZAbQjS7DMGqnD+GaVKUU5E7oJnYY92mgjxf
OYD1eedv2mah6Kv5Q2VdZGAbHLnoZXpiEQ/U004wm256ORfAwd0kWFMZVrC0kdj1e53oOZ9AwF1/DhSj
IQW2t9A5f5Ni0wghuQ7o+dVx/+1mySUGrKDbSv8xEkP9kwwAZTnhOCgt4b5YfBxQ6sejI2BLUbzxsABK
OJ8dezqgbmzts12z49LL9bOG5Xro5FvP2T5kwoDu6gofTFUTtV02snciUYfHXbE9Qefqv4k= rodol@L
APTOP-VH79P3BE

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ |
```

Se registra la clave publica en GitHub.



Se accede mediante la clave generada y registrada.


```
MINGW64:/c/Users/rodol
rodol@LAPTOP-VH79P3BE MINGW64 ~
$ cat Login.pub
ssh-rsa AAAAB3NzaC1yc2EAAAADAQABAAQGC3Lh7J6aIYSBgw1QV00ASKhZSNhU/v5Z9eV4CQn0sm
ZnWdd7TdcRAvNsVDw9HC9800rb8X8mD2Q7LtQ4R2jJbwzKhR0p/cFrHAJhRen2x62GdmpEadsk/Q60mN
OCbPi6G7MhcQdNcvhYRFm6Wp7za5ax2IoN/4p7mC1L6TUUKN4GvsK3qZGLA5cvmqqBFM0i1KbF/6+UyP
lVYiFpuM02SpC9+y1MiE+iG2HVDuqZt3CQtI817c84z1CZAbQjS7DMGqnD+GaVKUU5E7oJnYY92mgjxf
OYD1eedv2mah6Kv5Q2VdZGAbHLnoZXpiEQ/U004wm256ORfAwd0kWFmZVrC0kdj1e53oOZ9AwF1/DhSJ
IQW2t9A5f5Ni0wgHuQ7o+dVx/+lmySUGrKDbSv8xEkP9kwwAZTnhOCgt4b5YfBxQ6sejI2BLUbzxsabK
OJ8dezqgbmzts12z49LL9bOG5Xro5FvP2T5kwoDu6gofTFUTtV02snciUYfHXbE9Qefqv4k= rodol@L
APTOP-VH79P3BE

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ ^C

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ eval "$(ssh-agent -s)"
Agent pid 627

rodol@LAPTOP-VH79P3BE MINGW64 ~
$ ssh-add Login
Identity added: Login (rodol@LAPTOP-VH79P3BE)

rodol@LAPTOP-VH79P3BE MINGW64 ~
$
```

Se clona el archivo en GitHub en la ruta especificada.

```
MINGW64:/c/Users/rodol/OneDrive/Escritorio/Angular
$ cd Escritorio/

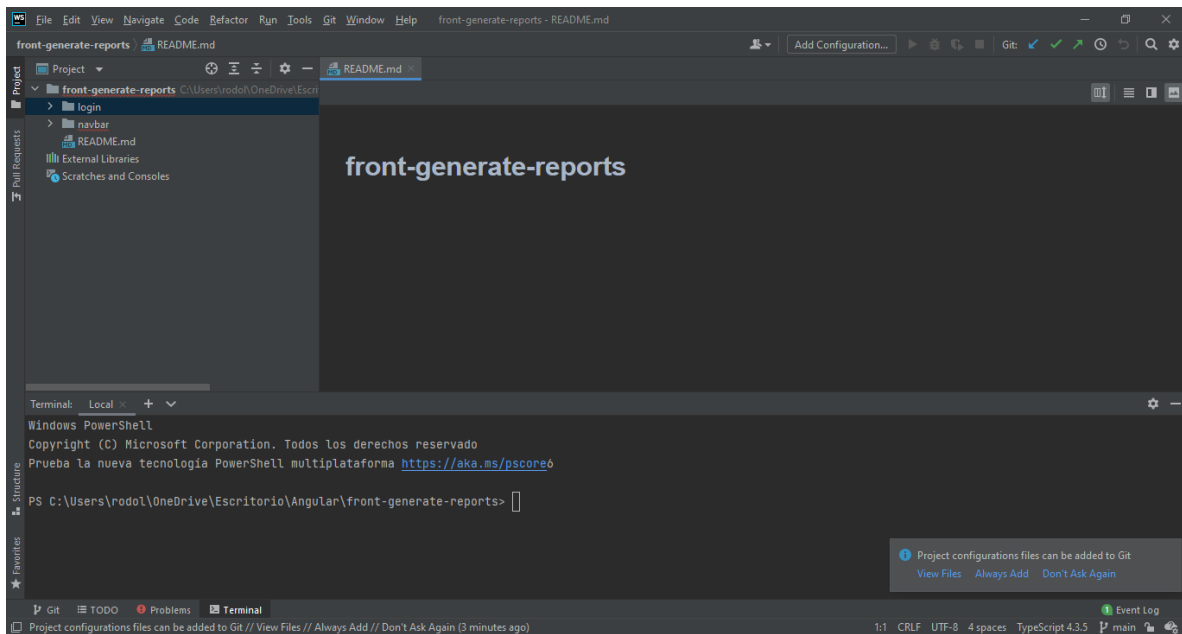
rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio
$ ls
Angular/                                'Visual Studio 2019 (2).lnk'*
'Cisco Packet Tracer Student.lnk'*      'Visual Studio Code.lnk'*
'Epson Print and Scan.lnk'*              'XAMPP Control Panel.lnk'*
'Microsoft SQL Server Management Studio 18.lnk'*  Zoom.lnk*
'Microsoft Teams.lnk'*                   desktop.ini
'Microsoft Visual Studio 2005.lnk'*

rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio
$ cd Angular/

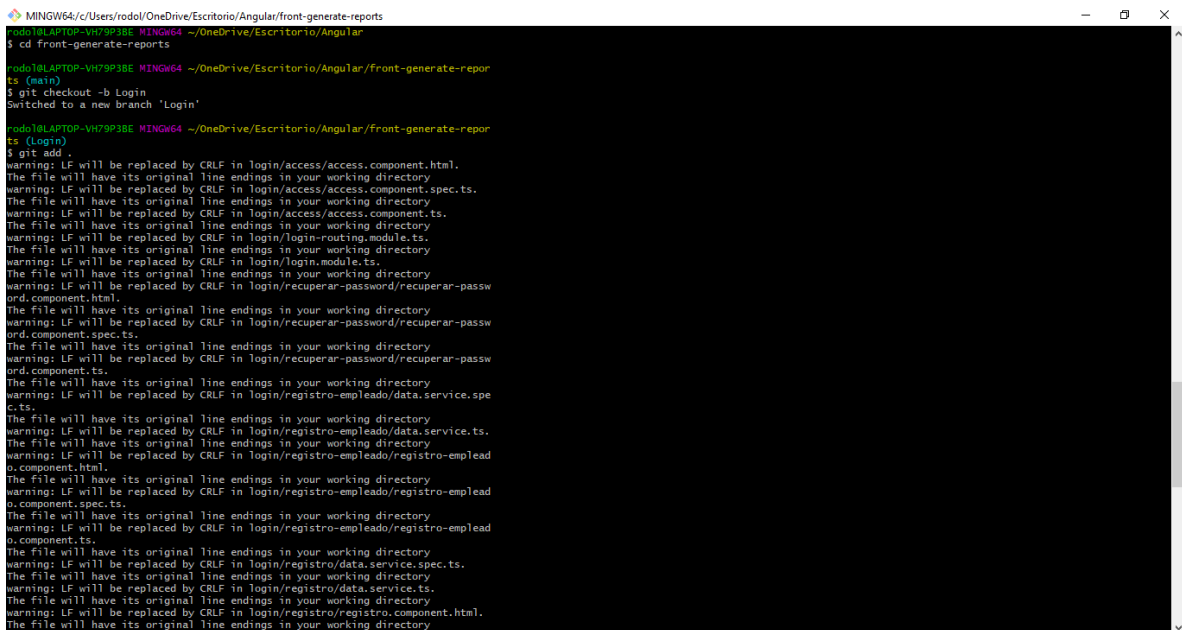
rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular
$ git clone git@github.com:sandyCortes/front-generate-reports.git
Cloning into 'front-generate-reports'...
remote: Enumerating objects: 3, done.
remote: Counting objects: 100% (3/3), done.
remote: Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (3/3), done.

rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular
$ |
```

Una vez clonado el proyecto se puede abrir en WebStorm.



Se hace el cambio de rama a Login para poder subir el proyecto generado por el equipo Login.



```
MINGW64:/c/Users/rodol/OneDrive/Escritorio/Angular/front-generate-reports
The file will have its original line endings in your working directory

rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular/front-generate-reports (Login)
$ git commit -m "Navbar con Login"
(Login 4301fca) Navbar con Login
32 files changed, 1010 insertions(+)
create mode 100644 .idea/.gitignore
create mode 100644 .idea/front-generate-reports.iml
create mode 100644 .idea/modules.xml
create mode 100644 .idea/vcs.xml
create mode 100644 login/access/access.component.html
create mode 100644 login/access/access.component.scss
create mode 100644 login/access/access.component.spec.ts
create mode 100644 login/access/access.component.ts
create mode 100644 login/login-routing.module.ts
create mode 100644 login/login.module.ts
create mode 100644 login/recuperar-password/recuperar-password.component.html
create mode 100644 login/recuperar-password/recuperar-password.component.scss
create mode 100644 login/recuperar-password/recuperar-password.component.spec.ts
create mode 100644 login/recuperar-password/recuperar-password.component.ts
create mode 100644 login/registro-empleado/data.service.spec.ts
create mode 100644 login/registro-empleado/data.service.ts
create mode 100644 login/registro-empleado/interface.ts
create mode 100644 login/registro-empleado/registro-empleado.component.html
create mode 100644 login/registro-empleado/registro-empleado.component.spec.ts
create mode 100644 login/registro-empleado/registro-empleado.component.ts
create mode 100644 login/registro/data.service.spec.ts
create mode 100644 login/registro/data.service.ts
create mode 100644 login/registro/interface-registro.ts
create mode 100644 login/registro/registro.component.html
create mode 100644 login/registro/registro.component.scss
create mode 100644 login/registro/registro.component.spec.ts
create mode 100644 login/registro/registro.component.ts
create mode 100644 navbar/navbar.component.html
create mode 100644 navbar/navbar.component.scss
create mode 100644 navbar/navbar.component.spec.ts
create mode 100644 navbar/navbar.component.ts

rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular/front-generate-reports (Login)
$ git push
fatal: The current branch Login has no upstream branch.
To push the current branch and set the remote as upstream, use

    git push --set-upstream origin Login
```

Se sube el archivo a la rama creada, que en este caso es la rama Login.

```
MINGW64:/c/Users/rodol/OneDrive/Escritorio/Angular/front-generate-repo...
git push --set-upstream origin Login


rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular/front-generate-reports (Login)
$ git push origin Login
Enumerating objects: 37, done.
Counting objects: 100% (37/37), done.
Delta compression using up to 4 threads
Compressing objects: 100% (34/34), done.
Writing objects: 100% (36/36), 29.13 KiB | 962.00 KiB/s, done.
Total 36 (delta 8), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (8/8), done.
remote:
remote: Create a pull request for 'Login' on GitHub by visiting:
remote:   https://github.com/sandyCortes/front-generate-reports/pull/new/Login
remote:
To github.com:sandyCortes/front-generate-reports.git
 * [new branch]      Login -> Login










rodol@LAPTOP-VH79P3BE MINGW64 ~/OneDrive/Escritorio/Angular/front-generate-reports (Login)
$
```

Por último, se visualiza que en GitHub en la rama de Login, los archivos subidos.

[Login](#) [8 branches](#) [0 tags](#) [Go to file](#) [Add file](#) [Code](#)

This branch is 5 commits ahead, 21 commits behind main. [Contribute](#)

 **rodolfogarciarojas** Add files via upload ed53a67 now [6 commits](#)

 .idea	Navbar con Login	2 months ago
 configuracion	Add files via upload	now
 login	Login	2 months ago
 navbar	Navbar con Login	2 months ago
 src/app	Add files via upload	now
 MANUAL DE USUARIO-LOGIN.pdf	Add files via upload	2 months ago
 Modelo ER.pdf	Add files via upload	last month
 README.md	Initial commit	3 months ago
 package.json	Add files via upload	now