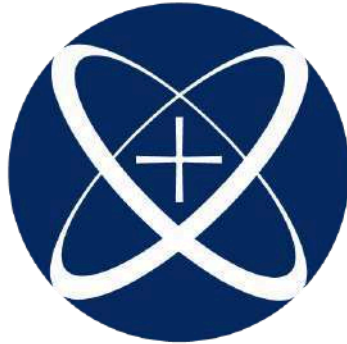


ITESO, UNIVERSIDAD JESUITA DE GUADALAJARA

TECNOLOGÍAS DE DESARROLLO DEL CLIENTE

Profesor: Francisco Sevilla



**ITESO, Universidad
Jesuita de Guadalajara**

Proyecto Integrador

POR

ISAAC ERNESTO VÁZQUEZ SANDOVAL - 745369

MATEO HERNANDEZ GUTIERREZ - 742023

Datos Generales

Link de Proyecto:

<https://neurotype-prod-frontend.onrender.com>

El servidor (no usar este)

<https://neurotype-prod-backend.onrender.com>

Repositorio:

<https://github.com/isaac-evs/neurotype>

(prototipo en angular)

<https://github.com/EurekaTESO/neurotype-angular>

Variables de Entorno: .env

```
SECRET_KEY=t7IzOFk41j736Pmja
ALGORITHM=HS256
ACCESS_TOKEN_EXPIRE_MINUTES=30
SQLALCHEMY_DATABASE_URI=
"postgresql://postgres:rootpostgrespassword@database1.cnmigicey46e.us-east-2.rds.amazonaws.com:5432/postgres"

AWS_ACCESS_KEY_ID=AKIAT4GVRLVWFXD7V5VM
AWS_SECRET_ACCESS_KEY=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
AWS_S3_BUCKET_NAME=neurotype-bucket

OPENAI_API_KEY=sk-proj-E62DvWRzYoXOE2BSNSo47-E89TsuHogI7ZgfQ30ZU3oE4GvI40yHNho4KYRI59BHDJSqY5f6wST3BibkFJTFFD-jGjuXxb1ShXOgrpDNdl3vfNtYQf2ZEsqkX2KpV0AF_2xDtxuWaakwelulLqYJYr6y8T4A

GOOGLE_CLIENT_ID=556952347377-ujs9djrI3t27lbtkvopn3u9fk61ot62j.apps.googleusercontent.com
GOOGLE_CLIENT_SECRET=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
```

Pasos para ejecutar el repositorio:

```
git clone https://github.com/isaac-evs/neurotype.git
```

```
cd neurotype/backend/app
```

```
touch .env (insertar datos)
```

```
cd ..
```

```
cd ..
```

```
docker-compose build -up
```

<http://localhost:3000/> (Vistas) o <http://localhost:8000/> (Servidor)

Nombre: Neuro Type

Descripción: Es una aplicación diseñada para personas interesadas en cuidar y mejorar su salud mental. Los usuarios pueden registrar sus emociones y pensamientos diarios mediante texto, y la app organiza esta información en un calendario personal, identificando tendencias emocionales a lo largo del tiempo.

La aplicación cuenta con un procesador de lenguaje natural que depende que tipo rol tengas, que detecta tendencias en los textos y genera análisis y gráficas en base a ello, también se conecta con un chatbot de soporte que se adapta a las tendencias emocionales del usuario y lo apoya.

¿Qué hace?:

- Detecta patrones en los textos
- Detecta palabras clave
- Detecta tendencias emocionales
- Ofrece asistencia emocional por medio de chat.

¿Qué NO hace?:

- No es un sistema de gestión de pacientes
- No es un CRM, ni planificador de sesiones

Tecnologías

Backend:

FastAPI:

Será el núcleo del backend para procesar textos mediante APIs rápidas y asíncronas, manejando eficientemente las operaciones de análisis de lenguaje natural.

PostgreSQL:

Se utilizará PostgreSQL como base de datos relacional para almacenar de forma estructurada y eficiente las transcripciones, notas y datos del usuario. Es ideal para manejar información que requiere relaciones entre tablas (usuarios, roles, entradas, etc.).

Procesamiento de Lenguaje Natural

SpaCy:

Utilizaremos esta biblioteca robusta para tareas avanzadas de procesamiento de texto, como análisis sintáctico, extracción de palabras clave y detección de emociones.

Hugging Face API:

Como complemento o alternativa, Hugging Face API proporcionará modelos preentrenados avanzados para análisis de textos, especialmente en casos donde se requiera mayor personalización.

Frontend

React:

React permitirá una interfaz dinámica y responsiva, ideal para una experiencia fluida y moderna.

TailwindCSS:

Nos ayudará a mantener estilos modulares y altamente personalizables en la aplicación.

Roles de Usuario y flujos:

Versión Lite (Gratis)

Esta versión está dirigida a usuarios que desean monitorear su bienestar emocional de forma personal y diaria. Ofrece funcionalidades básicas de análisis de texto para detectar y gestionar emociones mediante herramientas sencillas.

Flujo de trabajo:

- El usuario puede escribir notas diarias sobre su estado emocional o importar texto.
- Las notas son procesadas por **spaCy**, que detecta emociones y patrones básicos de sentimientos.
- Los resultados se visualizan en gráficos que muestran tendencias emocionales de manera sencilla.

Roles de usuario

Lite:

Diseñado para usuarios que desean una experiencia simple y accesible. Este rol permite:

- Registrar y analizar textos de manera básica para identificar emociones clave.
- Visualizar tendencias emocionales en un calendario personal.
- Ideal para quienes buscan una introducción al análisis de emociones sin funcionalidades avanzadas.

Plus:

Pensado para usuarios que buscan un análisis más profundo y personalizado. Este rol incluye:

- Acceso a herramientas avanzadas de análisis emocional utilizando modelos de NLP más sofisticados, como Hugging Face.
- Identificación de tendencias emocionales complejas, con detalles como intensidad emocional y palabras clave relacionadas.
- Un chatbot personalizado que adapta sus respuestas y sugerencias a las tendencias emocionales detectadas.
- Gráficas y reportes avanzados que permiten una visión clara de los patrones emocionales a lo largo del tiempo.

Módulos y niveles de acceso:

Módulo	Acción	Lite	Plus
Autenticación			
	Registro	✓	✓
	Inicio de sesión	✓	✓
Gestión de Notas			
	Exportar notas	✓	✓
	Ingresar texto manualmente	✓	✓
Análisis de Texto			
	Análisis básico con spaCy	✓	
	Análisis avanzado con BERT/GPT		✓
	Detección de emociones y sentimientos básicos	✓	✓
	Detección de variaciones sutiles y patrones complejos		✓
Visualización de Datos			
	Gráficos sencillos de evolución emocional	✓	✓
	Gráficos avanzados e interactivos	✓	✓
	Líneas de tiempo de evolución emocional	✓	✓

Diagrama E-R:

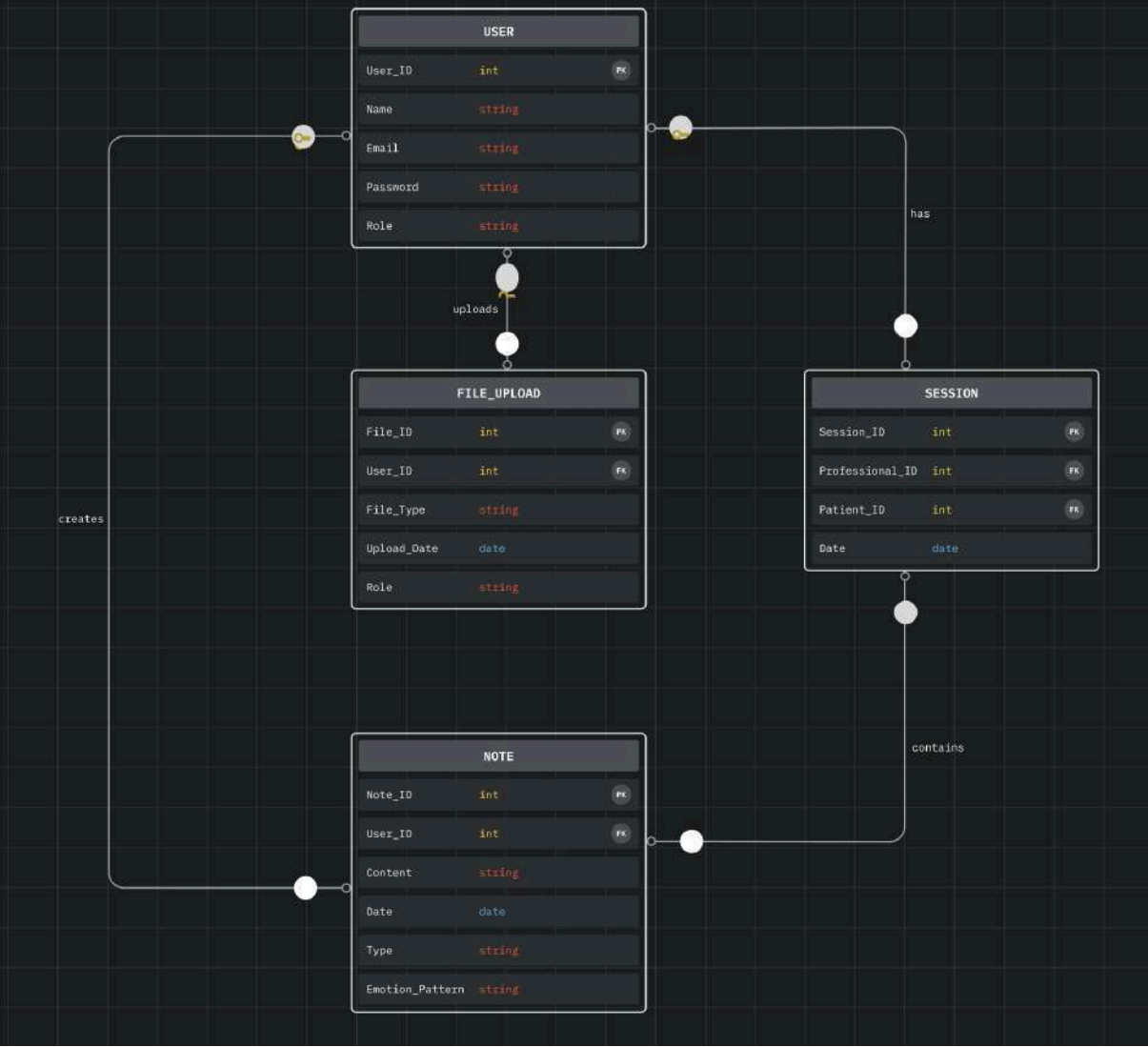
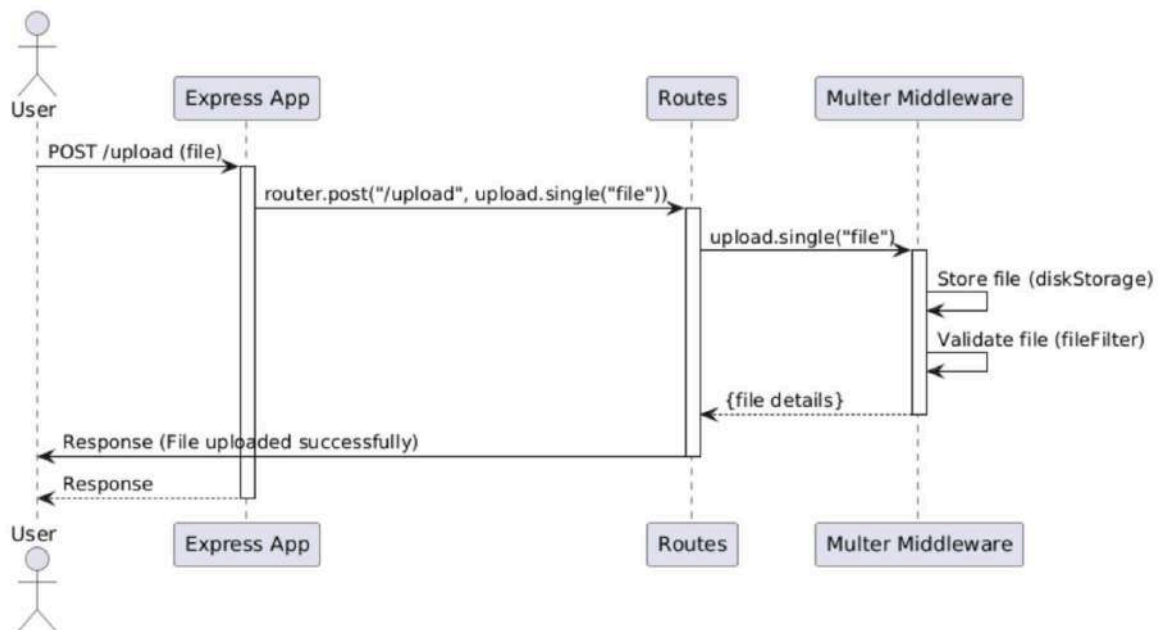
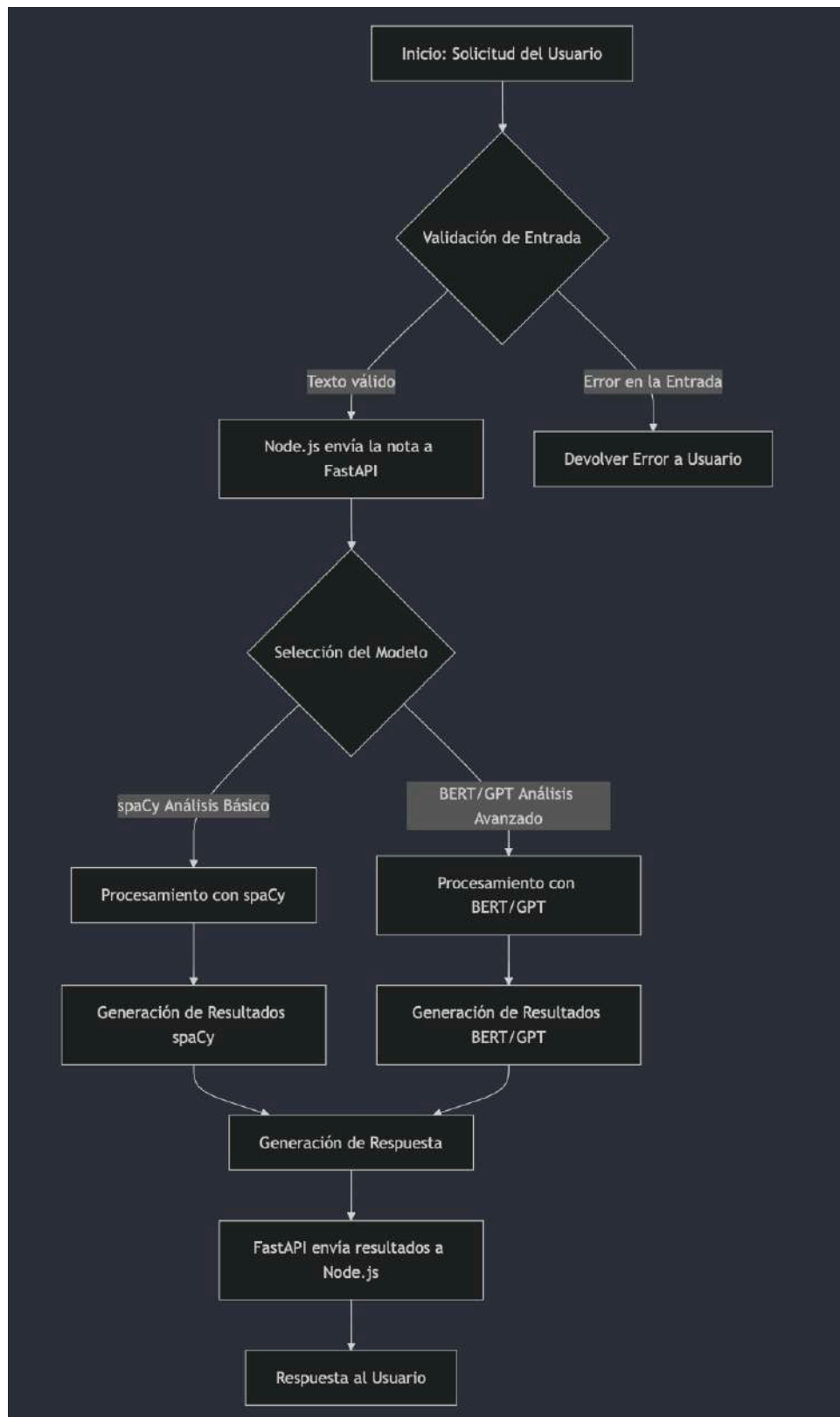


Diagrama de Flujo:

Subir notas



Análisis de notas:



Swagger:

users			^
POST	/register/	Register a new user	🔒
POST	/login/	Login and get access token	🔒
POST	/reset-password/	Reset a user's password	🔒
POST	/profile/	Update user profile	🔒
POST	/auth/google/	Authenticate with Google	🔒
GET	/users/me/	Get current user	🔒
notes			^
GET	/notes/	Find notes	🔒
POST	/notes/	Create note	🔒
PUT	/notes/{note_id}/	Update note	🔒
DELETE	/notes/{note_id}/	Delete note	🔒
GET	/notes/{slug}/	Get note by slug	🔒
GET	/notes/{slug}/	Get note by slug	🔒
dashboard			^
GET	/dashboard/	Get Dashboard	🔒
recommendations			^
GET	/recommendations/	Get personalized recommendations	🔒
data			^
GET	/data/export	Export user data	🔒

+ /Chat con Socket.IO

link:

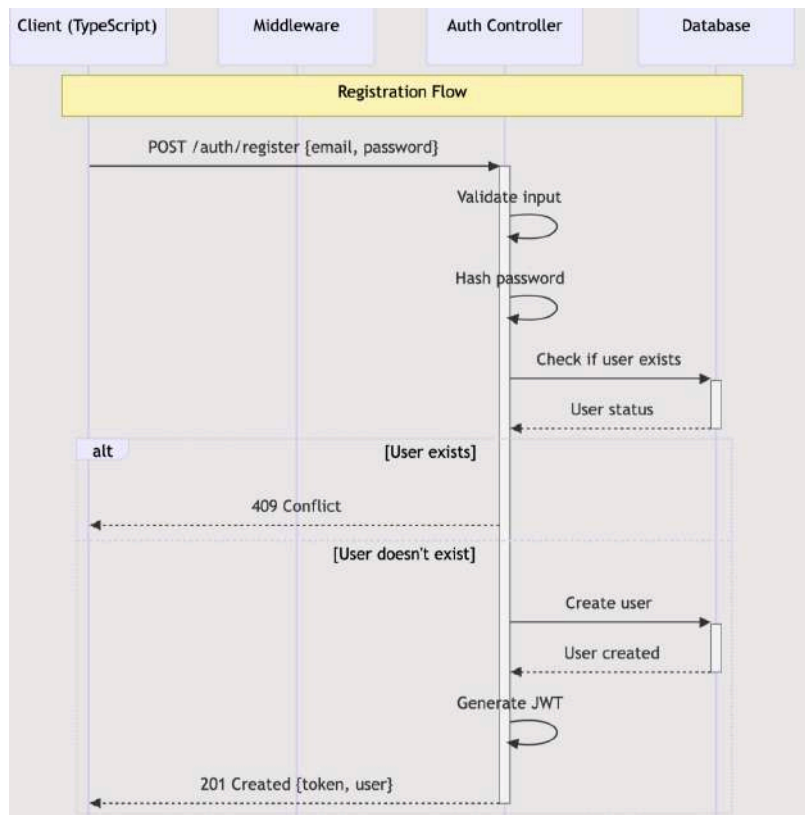
<https://neurotype-prod-backend.onrender.com/docs>

Explicación de cómo se realizó la configuración de Swagger

Swagger ya viene configurado por defecto con FastAPI, se añadieron descripciones y detalles a cada endpoint

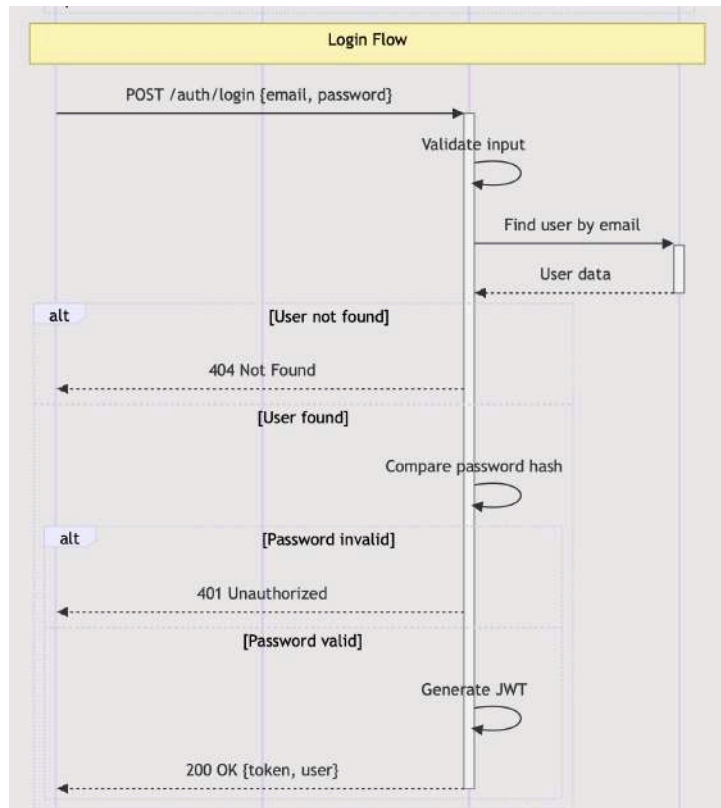
Autenticación de Servicios:

Registro:



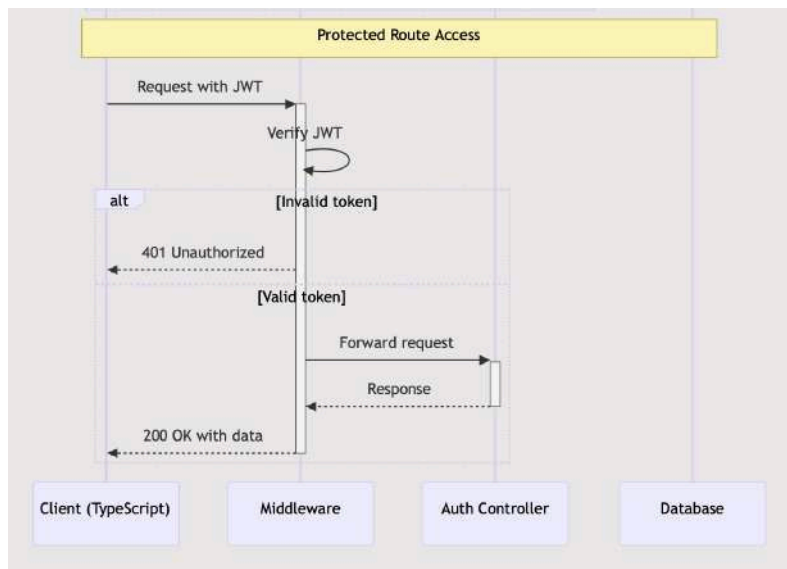
- El usuario crea mediante POST un formulario su cuenta, con
 - email
 - password
- El input es validado
- La contraseña es hasheada
- El usuario se guarda en la base de datos
- Se genera un token de sesión

Inicio de sesión:



- El usuario se loguea mediante un POST con los datos:
 - email
 - password
- El input es extraído y validado
- Se busca de manera asincrónica en la base de datos un usuario con el mail proporcionado
- Si el usuario no se encuentra aparece "Invalid login credentials"
- Si se encuentra se compara el hash de la contraseña, si no coincide da lo siguiente: "Invalid login credentials"
- Si se encuentra, genera un token y se envía como respuesta

Rutas Protegidas:



Autenticación:

- Extrae JWT
 - Verifica la existencia del token, si no hay: "No token provided"
 - Decodifica el token
 - Busca el usuario en la base de datos
 - Verifica la existencia del usuario
 - Asigna el usuario a req.user
-

Autorización:

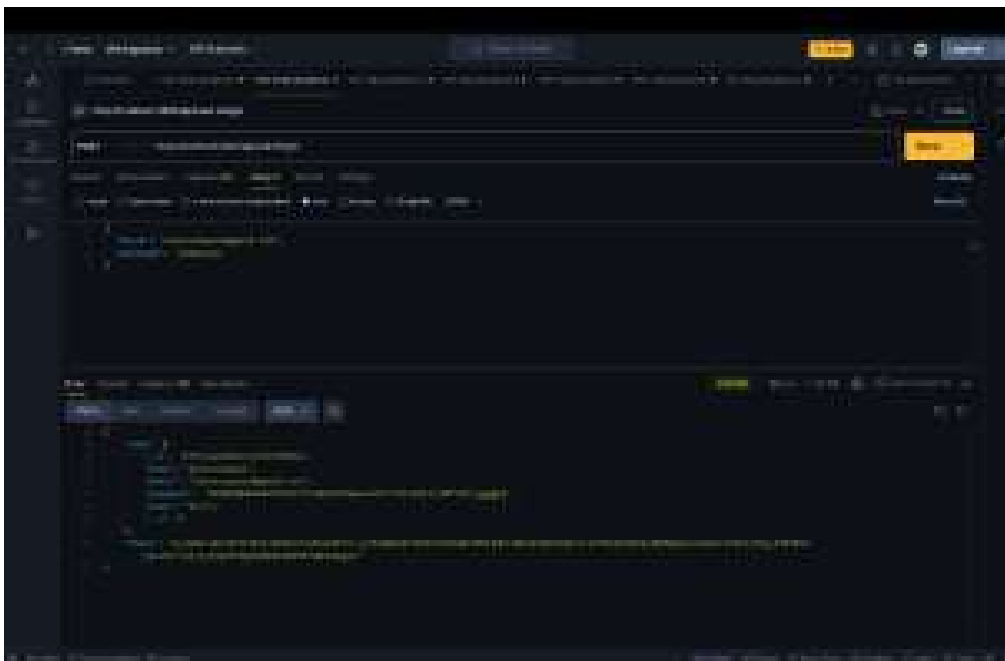
- Verifica si el usuario está autenticado
- Si no está autorizado da = 401 Unauthorized
- Si está autenticado, verifica que el rol del usuario este en la lista de permitidos, si no lo está da = 403 Forbidden
- Si está autenticado, sigue con next()

Ejemplo de validación y capturas:

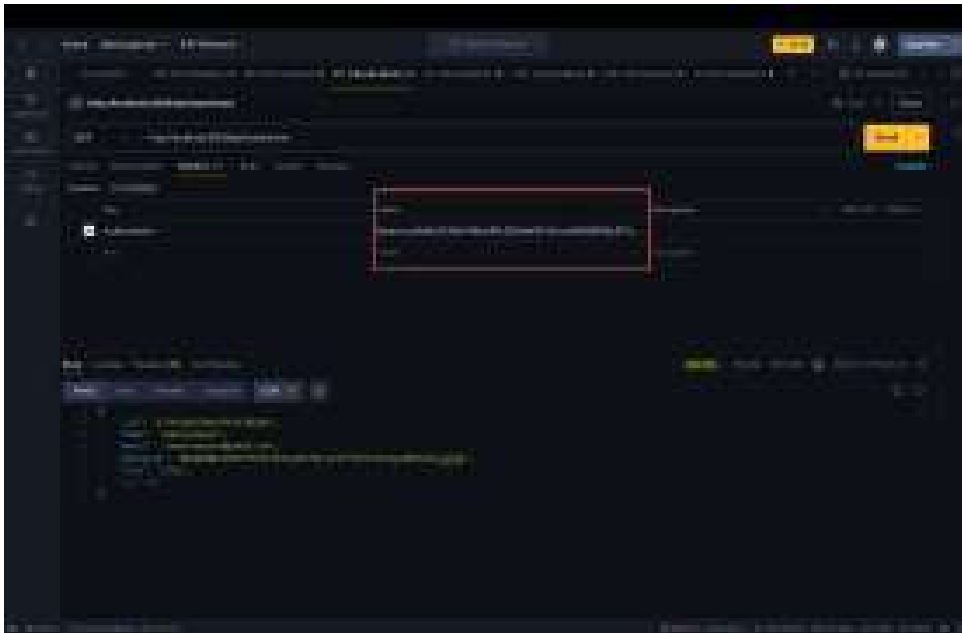
1. El usuario "Isaac", crea su cuenta:



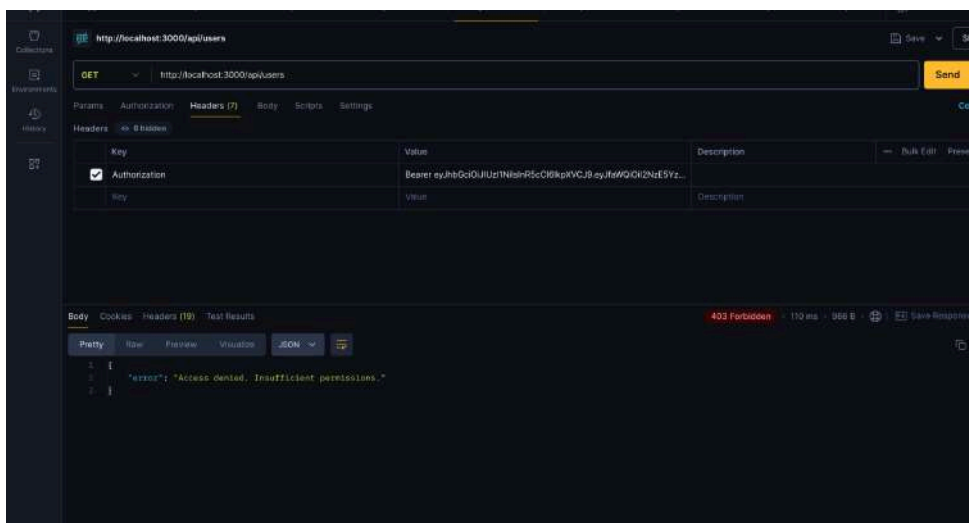
2. El usuario "Isaac", se logea:



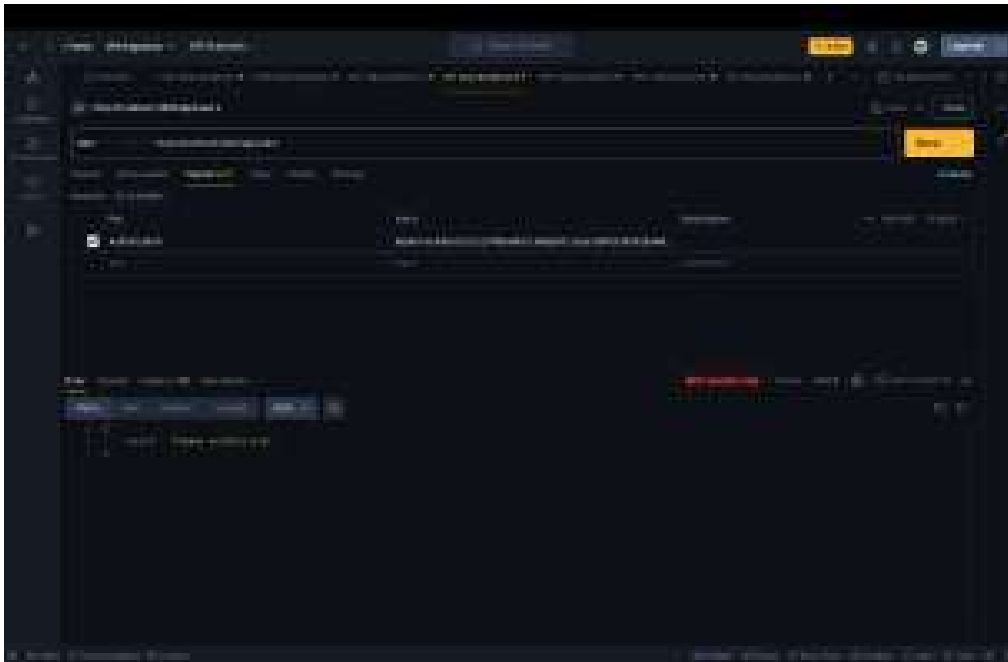
3. El usuario "Isaac"; entra a la sección home donde es **autenticado** mediante su token



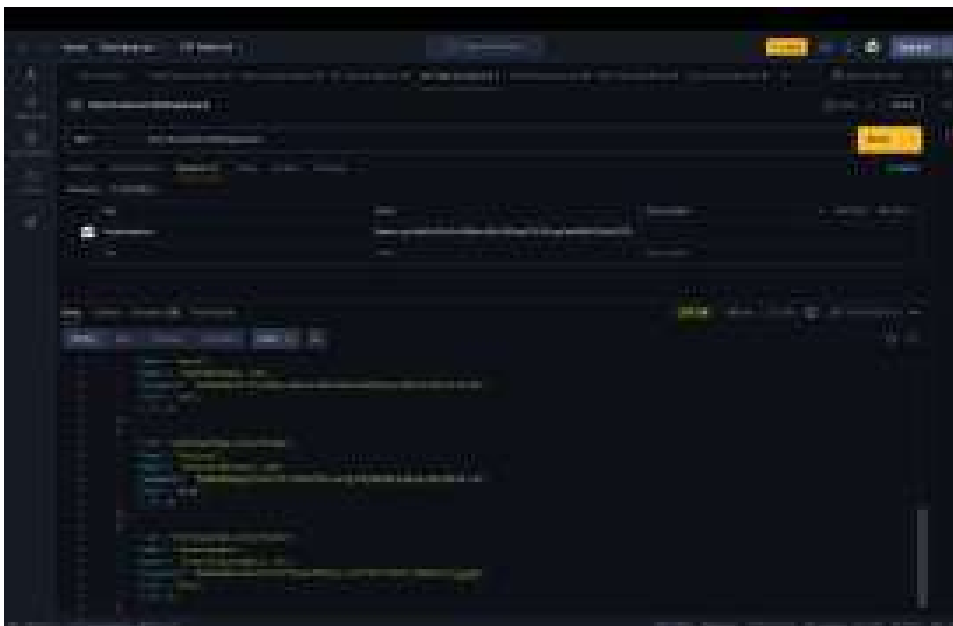
4. El usuario "Isaac"; desea conseguir la lista de todos los pacientes en la base de datos mediante su token, pero al ser versión "lite" no puede, por lo que es **autenticado sin permisos**



5. El usuario "Mateo" al ser usuario "pro" que posee los permisos necesarios, desea conseguir la lista de todos los pacientes en la base de datos , pero no ha iniciado sesión en mucho tiempo por lo que **no es autenticado**.



6. El usuario "Mateo" decide volver a iniciar sesión, por lo que se le otorga un nuevo token válido y vuelve a pedir los pacientes de la base de datos si ningun problema, lo que significa que está **autenticado y con permisos**.



Envio de Imagenes por medio de Formularios:

Dashboard



Welcome, Usuario

Total Notes: 0

Notes

Calendar

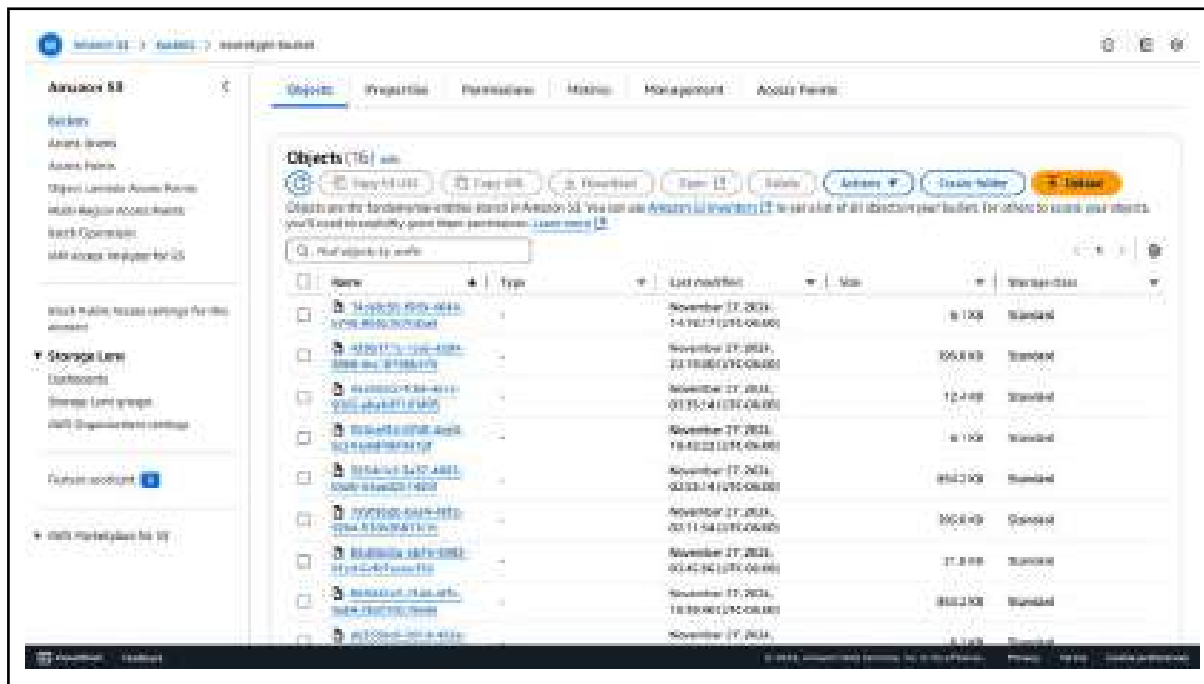
Export Data

C

Prevalent Emotion Today

happy

Objetos en el Bucket S3:



Ruta:

<https://neurotype-bucket.s3.us-east-2.amazonaws.com/439b171c-1ce6-4594-8888-4cc10768b576>

Captura de pantalla uso de Sockets (Socket.IO - REACT) FrontEnd:

Evento de conexión Main.py

```
@sio.event
async def connect(sid, environ):
    query_params = environ.get('asgi.scope').get('query_string', b'').decode()
    token = None
    for param in query_params.split('&'):
        if param.startswith('token='):
            token = param.split('=')[1]
            break
    if token is None:
        await sio.disconnect(sid)
        return
    user = await authenticate_user(sid, token)
    if user is None:
        await sio.disconnect(sid)
        return
    connected_users[sid] = user
    print(f"User {user.id} connected with SID {sid}")
```

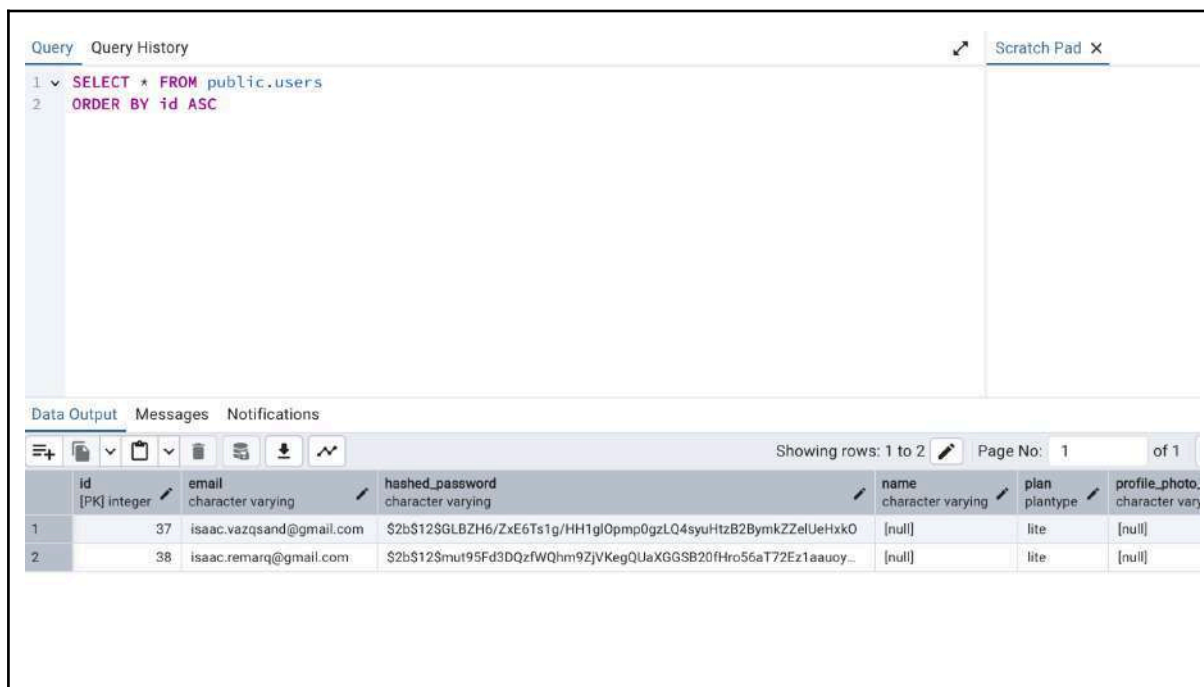
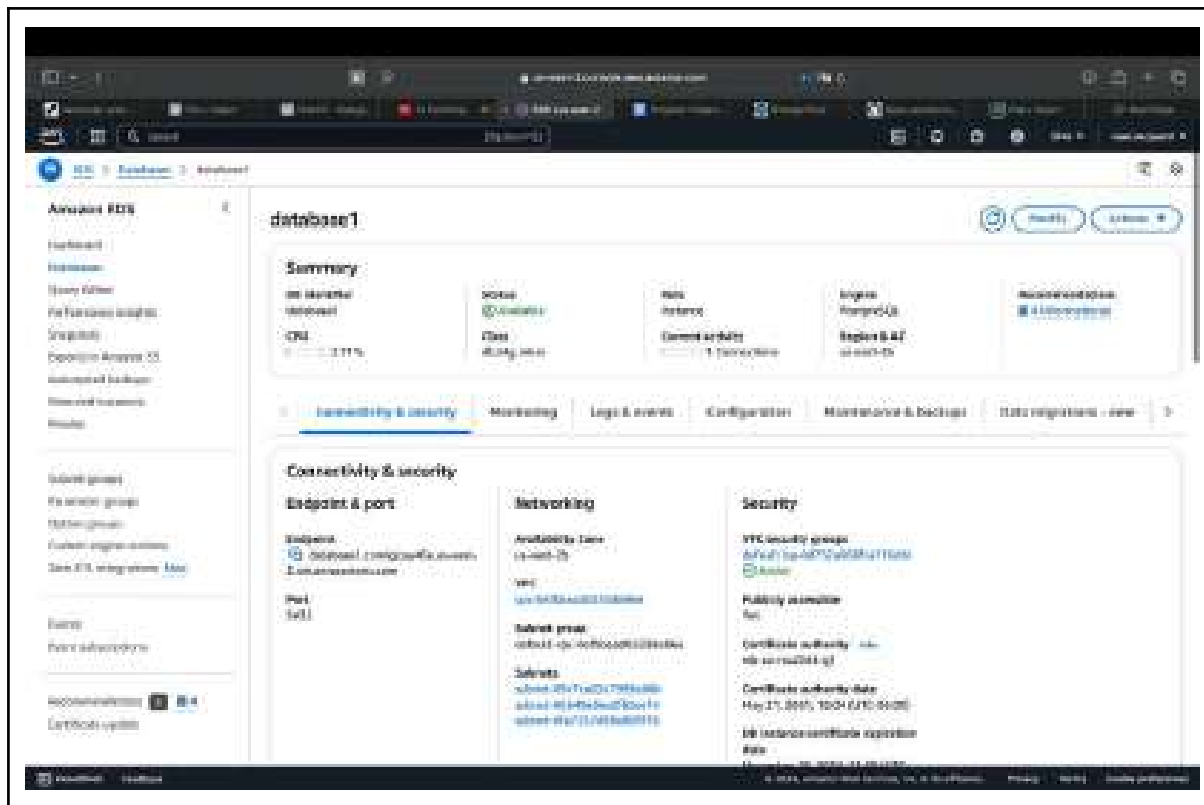
Conexión desde react ChatPage.jsx

```
// Login Handler
const handleLogin = async (e: React.FormEvent<HTMLFormElement>) => {
    e.preventDefault();
    const form = e.target as HTMLFormElement;
    const username = (form.login_username as HTMLInputElement).value;
    const password = (form.login_password as HTMLInputElement).value;

    try {
        const formData = new URLSearchParams();
        formData.append("username", username);
        formData.append("password", password);

        const response = await fetch("/token", {
            method: "POST",
            body: formData,
        });
        const result = await response.json();
        if (response.ok) {
            setToken(result.access_token);
            setCurrentUser(username);
        } else {
            alert("Login failed");
        }
    } catch (error) {
        console.error("Login error:", error);
    }
}
```

Base de Datos (RDS):



Componentes:



Directivas:

ejemplo:

```
{dashboardData.plan === "plus" && (  
  <motion.button  
    onClick={() => navigate("/chat")}  
    className="flex flex-col items-center bg-yellow-50  
    whileHover={{ scale: 1.05 }}  
    whileTap={{ scale: 0.95 }}  
  >  
    <FaComments className="text-3xl mb-2" />  
    <span className="font-semibold">Chat</span>  
  </motion.button>  
)}  
</motion.div>
```

Servicio:



Guardias:

```
2 import React, { useContext } from "react";
3 import { AuthContext } from "../context/AuthContext";
4 import { Navigate } from "react-router-dom";
5
6 export const PrivateRoute = ({ children }) => {
7   const { token } = useContext(AuthContext);
8   return token ? children : <Navigate to="/login" />;
9 };
10
```

Pruebas:

Todas las pruebas, tanto frontend como backend, se encuentran en el readme.MD del repositorio, ya que requieren de configuración adicional debido a Docker:

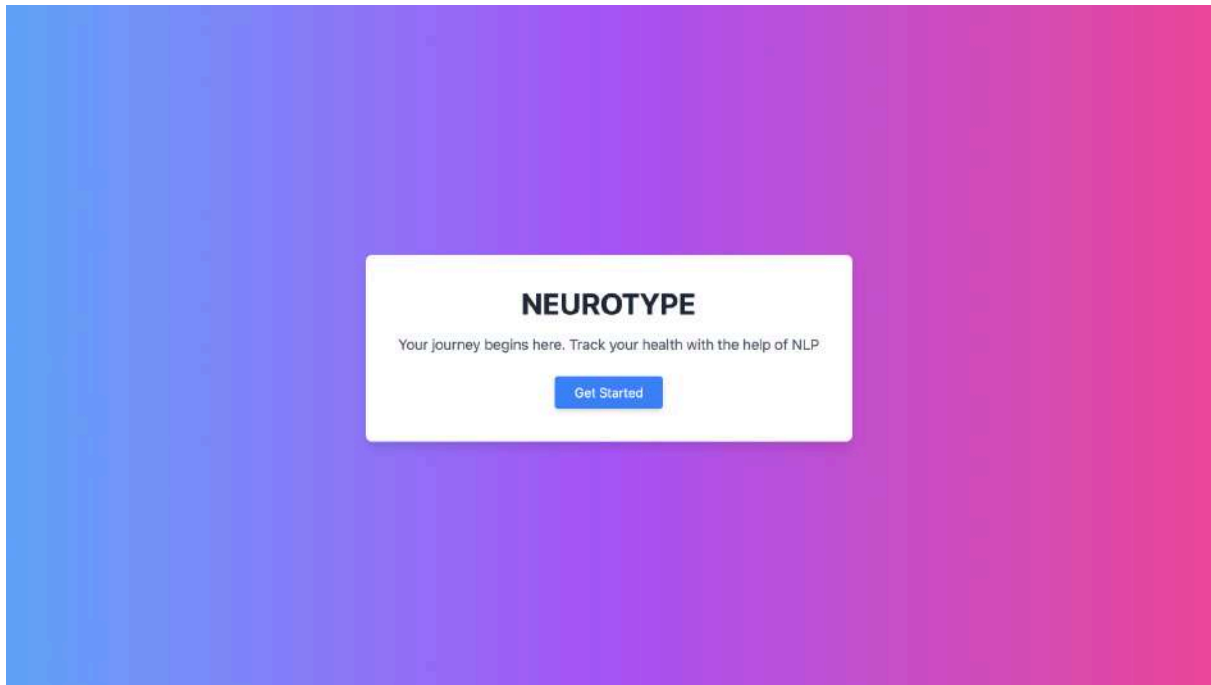
<https://github.com/isaac-evs/neurotype>

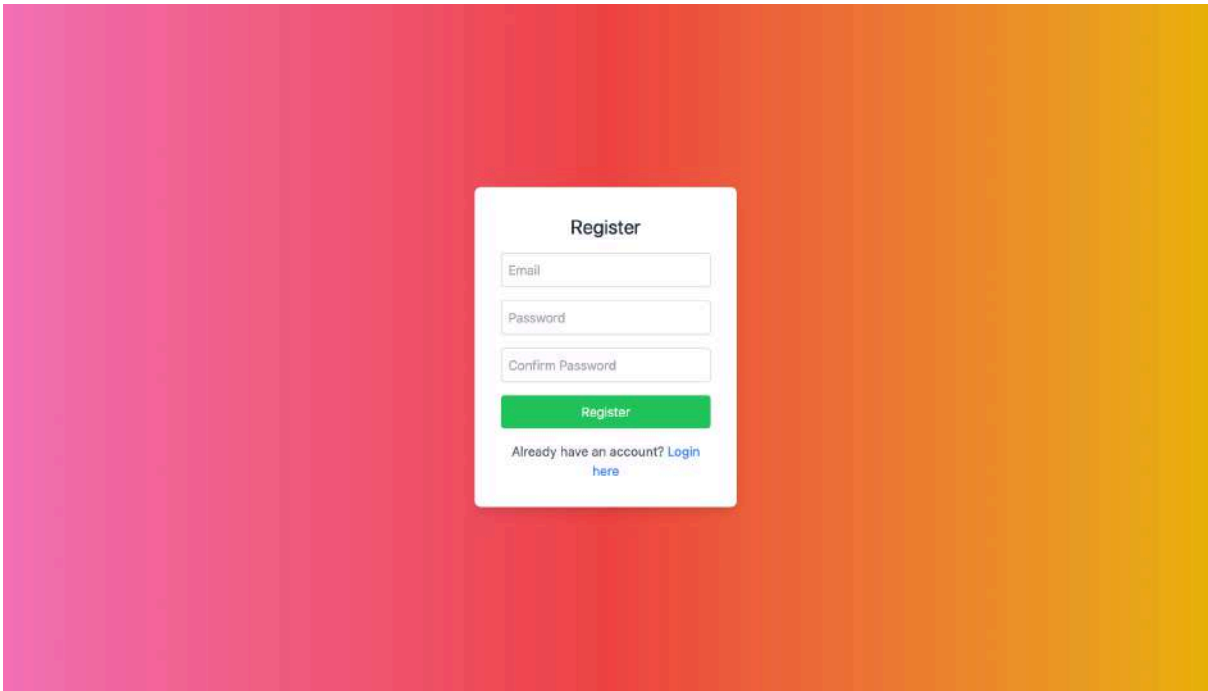
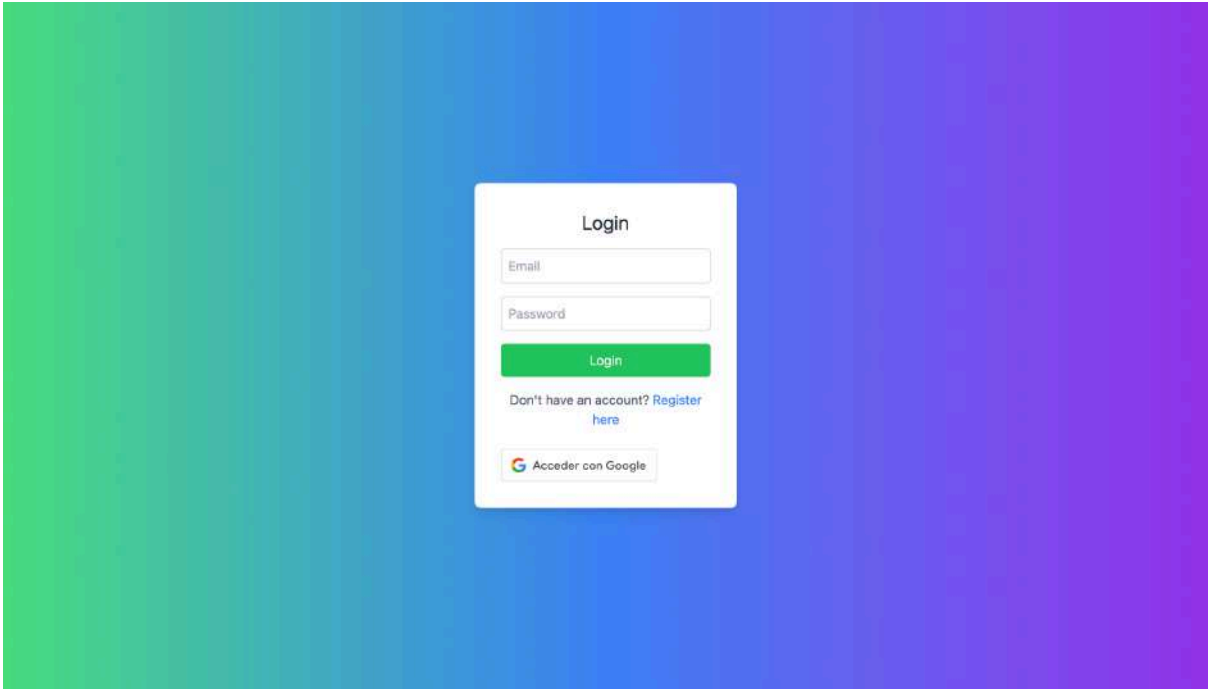
Diseño Vistas:

Prototipo:

<https://www.figma.com/proto/T6WKODdgy5h3B1kq2WIXuC/NeuroType-Wireframe?node-id=1-596&t=2MyA7oZ3kuaVTfNg-1>

Producción:





Select Your Plan

Choose a Plan

Plan: 100 episodes

Choose Plan

[Back to Dashboard](#)

Update Your Profile

Full Name

Profile Photo

Choose File or file selected

Save Profile

Dashboard

Logout



Welcome, Isaac
Total Notes: 4



Notes



Calendar



Export Data



Chat

Prevalent Emotion Today

happy

Emotion Counts This Week

2

Happy

1

Calm

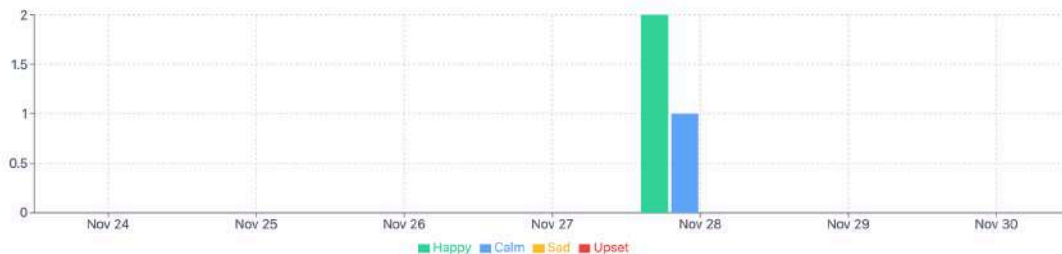
0

Sad

0

Upset

Weekly Emotion Chart



[← Back to Dashboard](#)

[+ New Note](#)

Your Notes

Untitled Note

I am so happy today!

Untitled Note

I feel really calm too!

Untitled Note

I am joyfull

Untitled Note

It feels so refreshing

← Back to Dashboard

Chat

How do I feel today?

You seem to be feeling content today. How are you doing?

Thanks, I have been feeling great lately, my therapy is going great and i feel like life is getting better day by day

That's wonderful to hear! It's great that you're feeling positive and that therapy is going well for you. Remember to continue practicing self-care and reaching out for support if you ever need it. Keep up the good work!

However, yesterday I had a panick attack, what can i do if i have another one?

I'm sorry to hear that you experienced a panic attack. If you have another one, here are some tips that may help: 1. Practice deep breathing: Focus on taking slow, deep breaths to help calm your mind and body. 2. Use grounding techniques: Try to focus on your surroundings by using your senses (e.g., noticing the texture of an object, listening to sounds around you). 3. Challenge negative thoughts: Remind yourself that panic attacks are temporary and that you have the strength to get through it. 4. Practice mindfulness or meditation: Engaging in mindfulness exercises can help you stay present and reduce anxiety. 5. Reach out for support: Talk to a trusted friend, family member, or mental health professional for support and

Thank you! :)

You're welcome! If you ever need support or someone to talk to, feel free to reach out. Take care!

Type your message...

Send

Registro e Inicio de Sesión:

Registro

Register

username@email.com

password

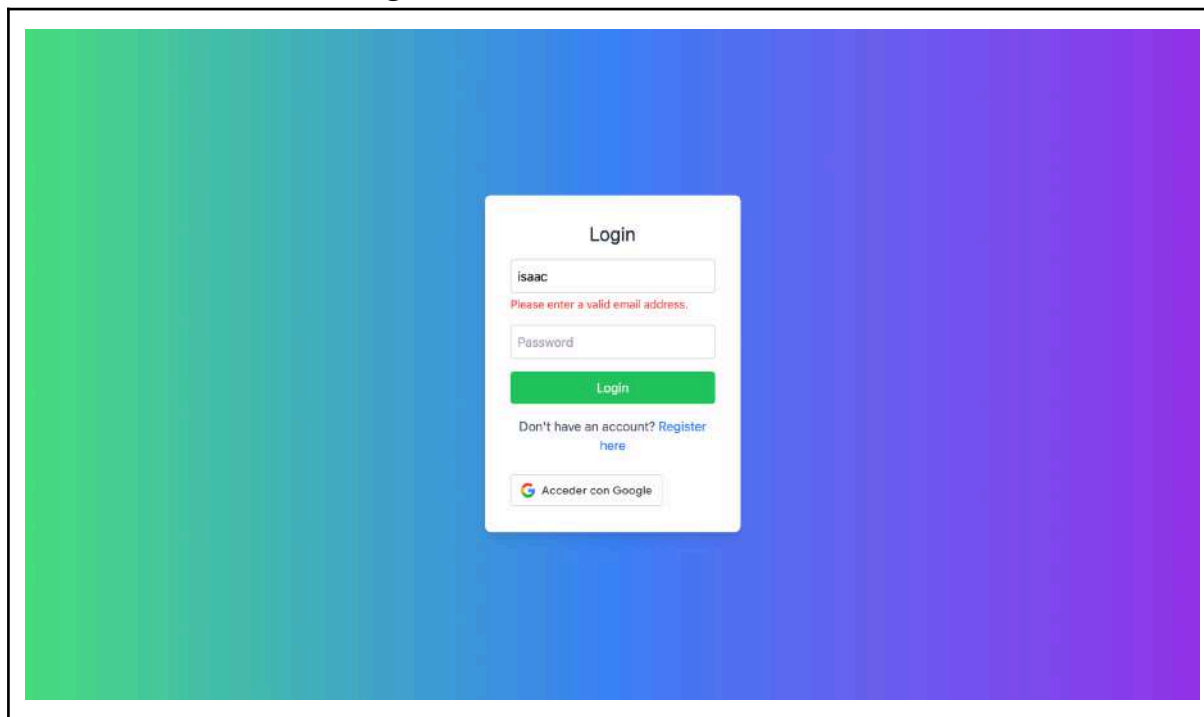
confirm password

Register

Already have an account? Login

here

Inicio de Sesión (Google o mail)



The login form is centered on a background with a horizontal gradient from green on the left to purple on the right. The form itself is white with a subtle shadow. It contains a title 'Login', an email input field with the text 'isaac' and a red error message 'Please enter a valid email address.', a password input field, a green 'Login' button, a link to 'Register here' for users without an account, and a button for 'Acceder con Google'.

Login


isaac

Please enter a valid email address.

Password

Login

Don't have an account? [Register here](#)

 Acceder con Google

Dashboard una vez logueado:

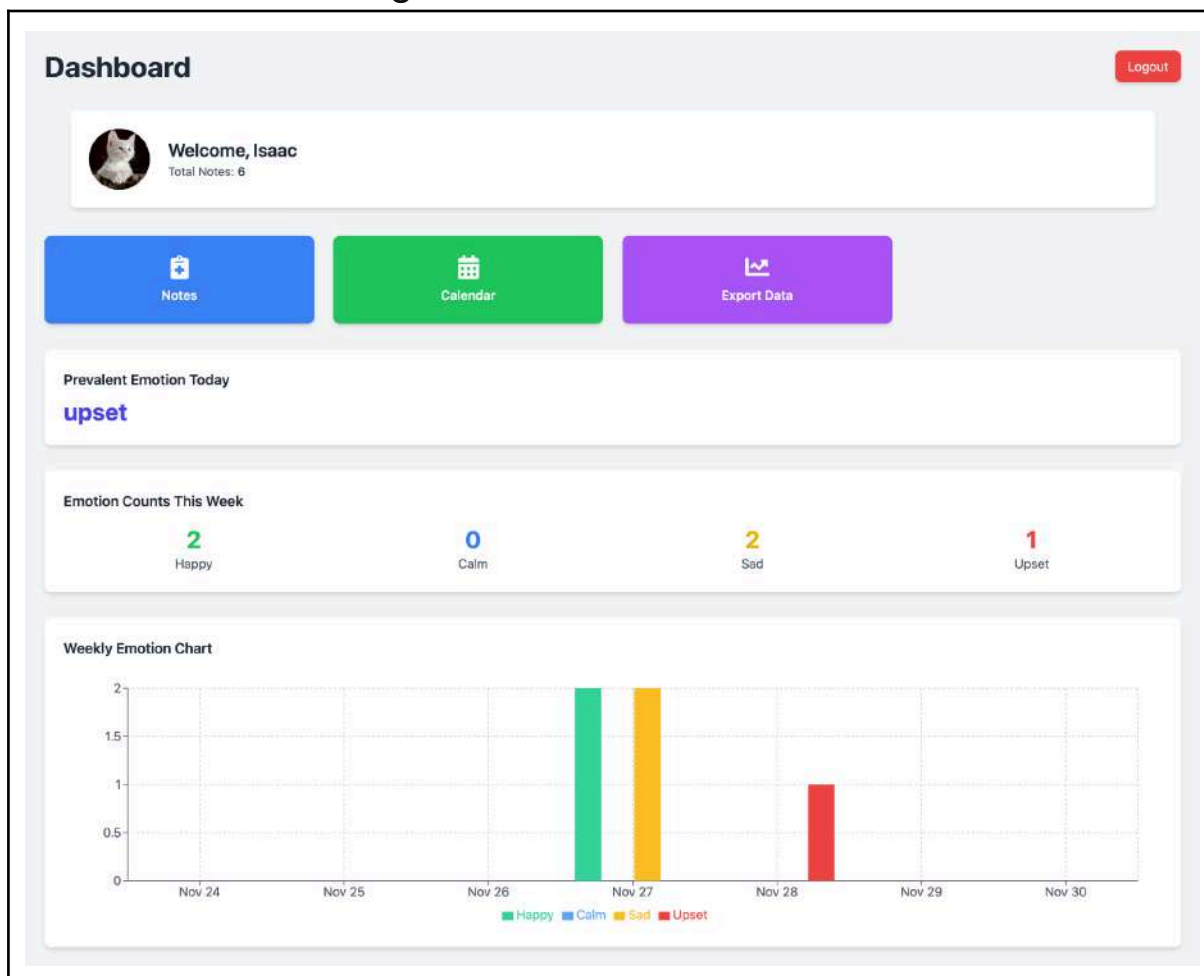


Diagrama de login/register:

