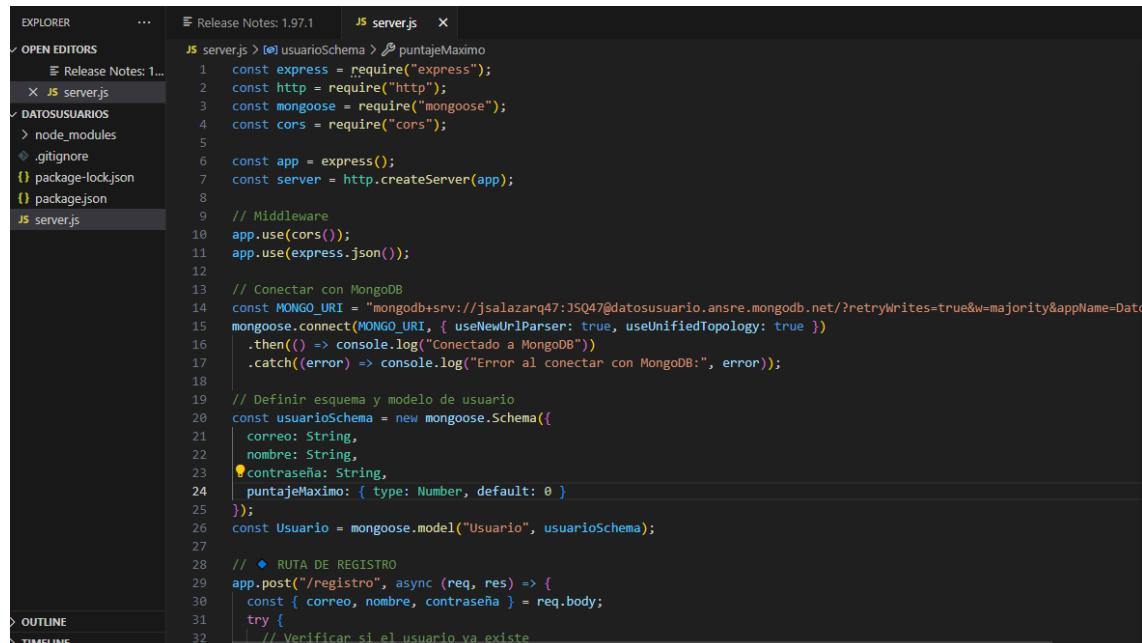


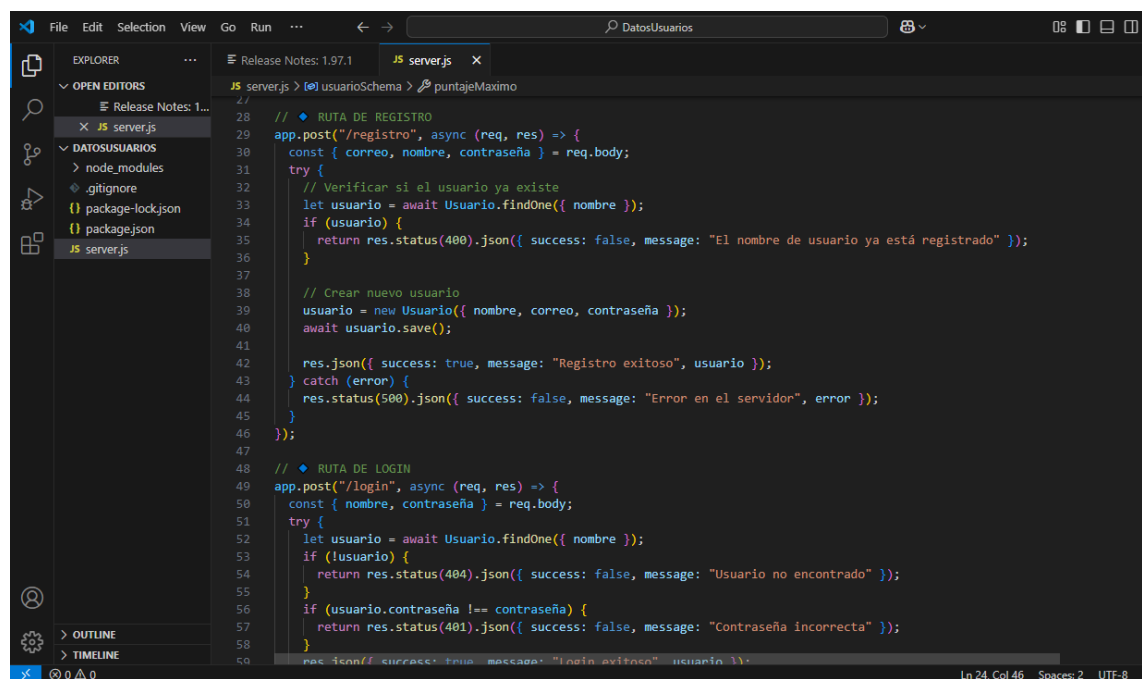
Interfaz

Conexión de interfaz con MongoDB para el registro e inicio de sesión de usuarios y también envía información como el puntaje del usuario. También el guardado de preguntas en un Excel (Google Sheets)

Server.js



```
1 const express = require("express");
2 const http = require("http");
3 const mongoose = require("mongoose");
4 const cors = require("cors");
5
6 const app = express();
7 const server = http.createServer(app);
8
9 // Middleware
10 app.use(cors());
11 app.use(express.json());
12
13 // Conectar con MongoDB
14 const MONGO_URI = "mongodb+srv://jsalazarq47:JSQ47@datosusuario.ansre.mongodb.net/?retryWrites=true&w=majority&appName=DatosUsuario";
15 mongoose.connect(MONGO_URI, { useNewUrlParser: true, useUnifiedTopology: true })
16   .then(() => console.log("Conectado a MongoDB"))
17   .catch((error) => console.log("Error al conectar con MongoDB:", error));
18
19 // Definir esquema y modelo de usuario
20 const usuarioSchema = new mongoose.Schema({
21   correo: String,
22   nombre: String,
23   contraseña: String,
24   puntajeMaximo: { type: Number, default: 0 }
25 });
26 const Usuario = mongoose.model("Usuario", usuarioSchema);
27
28 // RUTA DE REGISTRO
29 app.post("/registro", async (req, res) => {
30   const { correo, nombre, contraseña } = req.body;
31   try {
32     // Verificar si el usuario ya existe
```



```
33     let usuario = await Usuario.findOne({ nombre });
34     if (usuario) {
35       return res.status(400).json({ success: false, message: "El nombre de usuario ya está registrado" });
36     }
37
38     // Crear nuevo usuario
39     usuario = new Usuario({ nombre, correo, contraseña });
40     await usuario.save();
41
42     res.json({ success: true, message: "Registro exitoso", usuario });
43   } catch (error) {
44     res.status(500).json({ success: false, message: "Error en el servidor", error });
45   }
46 });
47
48 // RUTA DE LOGIN
49 app.post("/login", async (req, res) => {
50   const { nombre, contraseña } = req.body;
51   try {
52     let usuario = await Usuario.findOne({ nombre });
53     if (!usuario) {
54       return res.status(404).json({ success: false, message: "Usuario no encontrado" });
55     }
56     if (usuario.contraseña !== contraseña) {
57       return res.status(401).json({ success: false, message: "Contraseña incorrecta" });
58     }
59     res.json({ success: true, message: "Login exitoso", usuario });
60   }
```

```
48 // RUTA DE LOGIN
49 app.post("/login", async (req, res) => {
50   const { nombre, contraseña } = req.body;
51   try {
52     let usuario = await Usuario.findOne({ nombre });
53     if (!usuario) {
54       return res.status(404).json({ success: false, message: "Usuario no encontrado" });
55     }
56     if (usuario.contraseña !== contraseña) {
57       return res.status(401).json({ success: false, message: "Contraseña incorrecta" });
58     }
59     res.json({ success: true, message: "Login exitoso", usuario });
60   } catch (error) {
61     res.status(500).json({ success: false, message: "Error en el servidor", error });
62   }
63 });
64
65 // RUTA PARA ACTUALIZAR EL PUNTAJE MÁXIMO
66 app.post("/actualizar-puntaje", async (req, res) => {
67   const { nombre, nuevoPuntaje } = req.body;
68   try {
69     let usuario = await Usuario.findOne({ nombre });
70     if (!usuario) return res.status(404).json({ success: false, message: "Usuario no encontrado" });
71
72     if (nuevoPuntaje > usuario.puntajeMaximo) {
73       usuario.puntajeMaximo = nuevoPuntaje;
74       await usuario.save();
75     }
76     res.json({ success: true, message: "Puntaje actualizado", usuario });
77   } catch (error) {
78     res.status(500).json({ success: false, message: "Error en el servidor", error });
79   }
80 });
```

```
76   res.json({ success: true, message: "Puntaje actualizado", usuario });
77   catch (error) {
78     res.status(500).json({ success: false, message: "Error en el servidor", error });
79   }
80 });
81
82 // Iniciar servidor
83 const PORT = 3000;
84 server.listen(PORT, () => console.log(`Servidor corriendo en http://localhost:${PORT}`));
85
```

Interfaz de Login

Inicia sesión

Nombre

Contraseña

Iniciar

Código de la interfaz de Login

```
1  using System.Collections;
2  using UnityEngine;
3  using TMPro;
4  using UnityEngine.Networking;
5
6  public class LoginManager : MonoBehaviour
7  {
8      public TMP_InputField nombreInput;
9      public TMP_InputField contrasenaInput;
10     public TMP_Text mensajeLoginText;
11     public TMP_Text nombreJugadorText; // Para mostrar el nombre del jugador logueado
12     public TMP_Text nombreJugadorText2; // El nuevo TMP_Text donde también mostraremos el nombre
13     public GameObject botonJugar; // El botón de jugar que se activará después del login
14
15     public static string jugadorLogueadoNombre; // Variable estática para el nombre del jugador
16     public GameObject canvasLogin; // El canvas de login
17     public GameObject canvasJuego1; // El primer canvas del juego que se activará
18     public GameObject canvasJuego2; // El segundo canvas del juego que se activará
19
20     private string serverURL = "http://localhost:3000";
21
22     public void OnLoginButton()
23     {
24         if (string.IsNullOrEmpty(nombreInput.text) || string.IsNullOrEmpty(contrasenaInput.text))
25         {
26             mensajeLoginText.text = "❌ Todos los campos deben ser llenados.";
27             return;
28         }
29
30         StartCoroutine(LoginUsuario(nombreInput.text, contrasenaInput.text));
31     }
32 }
```

```
33  IEnumerator LoginUsuario(string nombre, string contrasena)
34  {
35     string jsonData = "{\"nombre\": \"" + nombre + "\", \"contraseña\": \"" + contrasena + "\"}";
36     byte[] jsonBytes = System.Text.Encoding.UTF8.GetBytes(jsonData);
37
38     UnityWebRequest request = new UnityWebRequest(serverURL + "/login", "POST");
39     request.uploadHandler = new UploadHandlerRaw(jsonBytes);
40     request.downloadHandler = new DownloadHandlerBuffer();
41     request.SetRequestHeader("Content-Type", "application/json");
42
43     yield return request.SendWebRequest();
44
45     if (request.result == UnityWebRequest.Result.Success)
46     {
47         var response = JsonUtility.FromJson<LoginResponse>(request.downloadHandler.text);
48         jugadorLogueadoNombre = response.usuario.nombre; // Guardamos el nombre del jugador logueado
49         nombreJugadorText.text = "Bienvenido, " + jugadorLogueadoNombre;
50         nombreJugadorText2.text = "Jugador: " + jugadorLogueadoNombre; // Actualizamos el segundo TMP_Text
51         mensajeLoginText.text = "✅ Login exitoso!";
52
53         // Activamos el botón de Jugar
54         botonJugar.SetActive(true);
55     }
56     else
57     {
58         mensajeLoginText.text = "❌ " + request.downloadHandler.text;
59         botonJugar.SetActive(false); // Desactivamos el botón de Jugar si el login falla
60     }
61 }
```

```

61     }
62
63     public void OnJugarButton()
64     {
65         // Cerrar el canvas de login
66         canvasLogin.SetActive(false);
67
68         // Activar los dos canvas necesarios para el juego
69         canvasJuego1.SetActive(true);
70         canvasJuego2.SetActive(true);
71     }
72
73     [System.Serializable]
74     public class LoginResponse
75     {
76         public bool success;
77         public string message;
78         public Usuario usuario;
79     }
80
81     [System.Serializable]
82     public class Usuario
83     {
84         public string nombre;
85         public string contraseña;
86     }
87 }
88

```

Interfaz de Registro

Registrate

Correo

Correo

Nombre y apellido

Nombre y apellido

Contraseña

Contraseña

Registrate

Código de la interfaz de registro

```
RegisterManger.cs X
RegisterManger.cs
1  using System.Collections;
2  using UnityEngine;
3  using UnityEngine.UI;
4  using TMPro;
5  using UnityEngine.Networking;
6
7  public class RegistroManager : MonoBehaviour
8  {
9      public TMP_InputField correoInput;
10     public TMP_InputField nombreInput;
11     public TMP_InputField contrasenaInput;
12     public TMP_Text estadoRegistroText;
13
14     private string serverURL = "http://localhost:3000";
15
16     public void OnRegistroButton()
17     {
18         // Verificar si los campos no están vacíos
19         if (string.IsNullOrEmpty(correoInput.text) || string.IsNullOrEmpty(nombreInput.text) || string.IsNullOrEmpty(contrasenaInput.text))
20         {
21             estadoRegistroText.text = "❌ Todos los campos deben ser llenados.";
22             return;
23         }
24
25         StartCoroutine(RegistrarUsuario(correoInput.text, nombreInput.text, contrasenaInput.text));
26     }
27
28     IEnumerator RegistrarUsuario(string correo, string nombre, string contrasena)
29     {
30         string jsonData = "{\"correo\": \"" + correo + "\", \"nombre\": \"" + nombre + "\", \"contraseña\": \"" + contrasena + "\"}";
31         byte[] jsonBytes = System.Text.Encoding.UTF8.GetBytes(jsonData);
32
33         UnityWebRequest request = new UnityWebRequest(serverURL + "/registro", "POST");
34         request.uploadHandler = new UploadHandlerRaw(jsonBytes);
35         request.downloadHandler = new DownloadHandlerBuffer();
36         request.SetRequestHeader("Content-Type", "application/json");
37
38         yield return request.SendWebRequest();
39
40         if (request.result == UnityWebRequest.Result.Success)
41         {
42             estadoRegistroText.text = "✅ Registro exitoso!";
43         }
44         else
45         {
46             estadoRegistroText.text = "❌ " + request.downloadHandler.text; // Muestra el mensaje de error
47         }
48     }
49 }
50
51
```

Uso de las interfaces

The image shows a registration form titled "Registrate" in a pixelated font. It has three input fields: "Correo" with the text "jsq@est.ups.edu.ec", "Nombre y apellido" with the text "Cristiano Ronaldo", and "Contraseña" with masked characters "*****". Below the fields is a button labeled "Registrate". To the right of the form, there is a message "✅ Registro exitoso!" (Successful registration!).

Jugador:
Cristiano
Ronaldo

Tu puntaje:

Inicia sesión

Nombre

Cristiano Ronaldo

Contraseña

Iniciar

Login exitoso!

Ingresaste como:
Bienvenido, Cristiano Ronaldo

Interfaz de puntaje

Llegaste a la
puntuacion de

790

Enviar datos

Volver al menu

□ Puntaje
actualizado!


Información obtenida en la base de datos:


```
▶ _id: ObjectId('67acce69f037473b656ab17c')
  correo : "jsq@est.ups.edu.ec"
  nombre : "Cristiano Ronaldo"
  contraseña : "123456"
  puntajeMaximo : 790
  v : 0
```

Preguntas

La intención del juego es que dentro hay enemigos con ejercicios matemáticos correctos e incorrectos, el juego accede a un Excel (google sheets) publicado como .csv que toma lo escrito en la primera columna en este caso el juego está orientado a matemáticas básicas, aunque se puede cambiar en este caso esta de ejemplo las tablas del 1 al 10, un archivo para los correctos y otro para los incorrectos.

	Ejercicios Incorrectos	yo	10 feb 2025	⋮
	Ejercicios correctos 	yo	10 feb 2025	⋮

	Ejercicios Incorrectos	Archivo	Editar	Ver	Inse
<div>🔍 ↶ ↷ 🖨 🗣 100%</div>					
A1	fx 1 x 1 = 2				
	A	B			
1	1 x 1 = 2				
2	1 x 2 = 3				
3	1 x 3 = 4				
4	1 x 4 = 5				
5	1 x 5 = 6				
6	1 x 6 = 7				
7	1 x 7 = 8				
8	1 x 8 = 9				
9	1 x 9 = 10				
10	1 x 10 = 11				
11	2 x 1 = 3				
12	2 x 2 = 5				
13	2 x 3 = 7				
14	2 x 4 = 9				
15	2 x 5 = 11				
16	2 x 6 = 13				
17	2 x 7 = 15				
18	2 x 8 = 17				
19	2 x 9 = 19				
20	2 x 10 = 21				
21	3 x 1 = 4				
22	3 x 2 = 7				
23	3 x 3 = 10				
24	3 x 4 = 13				
25	3 x 5 = 16				
26	3 x 6 = 19				
27	3 x 7 = 22				
28	3 x 8 = 25				
29	3 x 9 = 28				
30	3 x 10 = 31				




Ejercicios correctos ☆


Archivo


Editar


Ver


Inserta













100%

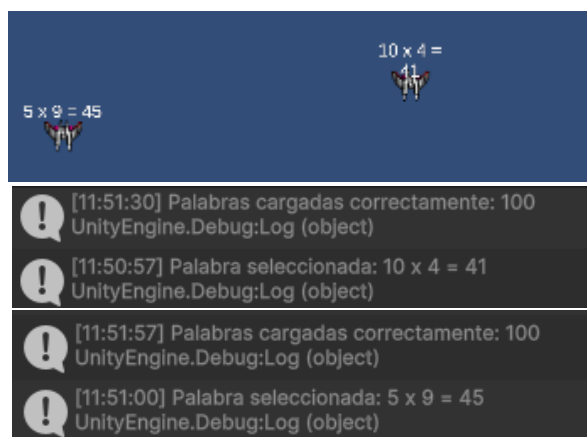


A1

 1 x 1 = 1

	A	B	
1	1 x 1 = 1		
2	1 x 2 = 2		
3	1 x 3 = 3		
4	1 x 4 = 4		
5	1 x 5 = 5		
6	1 x 6 = 6		
7	1 x 7 = 7		
8	1 x 8 = 8		
9	1 x 9 = 9		
10	1 x 10 = 10		
11	2 x 1 = 2		
12	2 x 2 = 4		
13	2 x 3 = 6		
14	2 x 4 = 8		
15	2 x 5 = 10		
16	2 x 6 = 12		
17	2 x 7 = 14		
18	2 x 8 = 16		
19	2 x 9 = 18		
20	2 x 10 = 20		
21	3 x 1 = 3		
22	3 x 2 = 6		
23	3 x 3 = 9		
24	3 x 4 = 12		
25	3 x 5 = 15		
26	3 x 6 = 18		
27	3 x 7 = 21		
28	3 x 8 = 24		
29	3 x 9 = 27		
30	3 x 10 = 30		

Lo que se muestra dentro del juego:



GitHub: [TIC-InnovaEdu/mathspace](https://github.com/TIC-InnovaEdu/mathspace)