**Tarea - TIA-03**

* **Modalidad de Evaluación: En grupo**
* **Peso: 10% (de la nota final)**
* **Metodología**: Aprendizaje Basado en Problemas (ABR)

**EQUIPO: A**

**MIEMBROS DEL EQUIPO:**

* Angelo Alexander Arango Graciano
* Joimar Danilo Urrego David
* Sebastian Puerta Mejía
* Paulina Cardona Cardona
* Luis Miguel Sosa Bohorquez

**Caso de Estudio**

El Caso de Estudio está relacionado con los Proyectos PA/PIA. Debe tomar en consideración todo el material que se le ha suministrado como el enunciado que se le entregó en la Tarea 2 (TIA-2), el formato de registro de proyecto, la información que socializó la Profesora Vesna Srdanovic en clases, la revisión de los diferentes diccionarios de datos de los otros grupos y toda la información relacionada con los proyectos que Ud. puede recolectar (puede consultar a otros docentes en relación al tema)

**Material de ayuda:**

* Puede utilizar el contenido del ejercicio del Taller Rest-HTTP del 28-03-2025 relacionado con piscinas y adecuarlo a los datos de esa tarea

**Criterios de desempeño / Indicadores de Resultado de Aprendizaje**

* Utilizar los códigos de respuesta HTTP/HTTPS para comprender el estado de las solicitudes y respuestas en una aplicación web.
* Aplicar el uso de APIs RESTful para la comunicación con un servidor backend y el consumo de dichas APIs para la obtención y manipulación de datos.
* Desarrollar habilidades de trabajo en equipo y comunicación efectiva en la resolución de ejercicios prácticos relacionados con tecnologías web.
* El pensamiento sistémico y complejo
* La visión global
* La multi e interdisciplinariedad
* La vinculación solidaria
* Las emociones y los comportamientos
* La autonomía

**Actividad:**

* Analizar el Caso de Estudio suministrado por el Profesor.
* Diseñar e implementar una página Web responsiva que establezca una interacción con un Servidor
* implementar llamadas APIs RESTful para la comunicación con un servidor Backend
* Implementar respuestas con códigos HTTP.
* Conectar y actualizar una base de datos con las cuatro (4) operaciones básicas de un CRUD: INSERT, UPDATE, DELETE, SELECT

**Propósito:**

* Comprender el contexto de un problema o necesidad real y presentar una propuesta de solución
* Comprender cómo funciona un sistema de interacción FrontEnd / BackEnd
* Profundizar los conocimientos de una arquitectura REST
* Practicar con un sistema de Solicitud/Respuesta con el uso de códigos HTTP
* Aprender a conectar y actualizar bases de datos con sistemas de información Web

**Modalidad de entrega:**

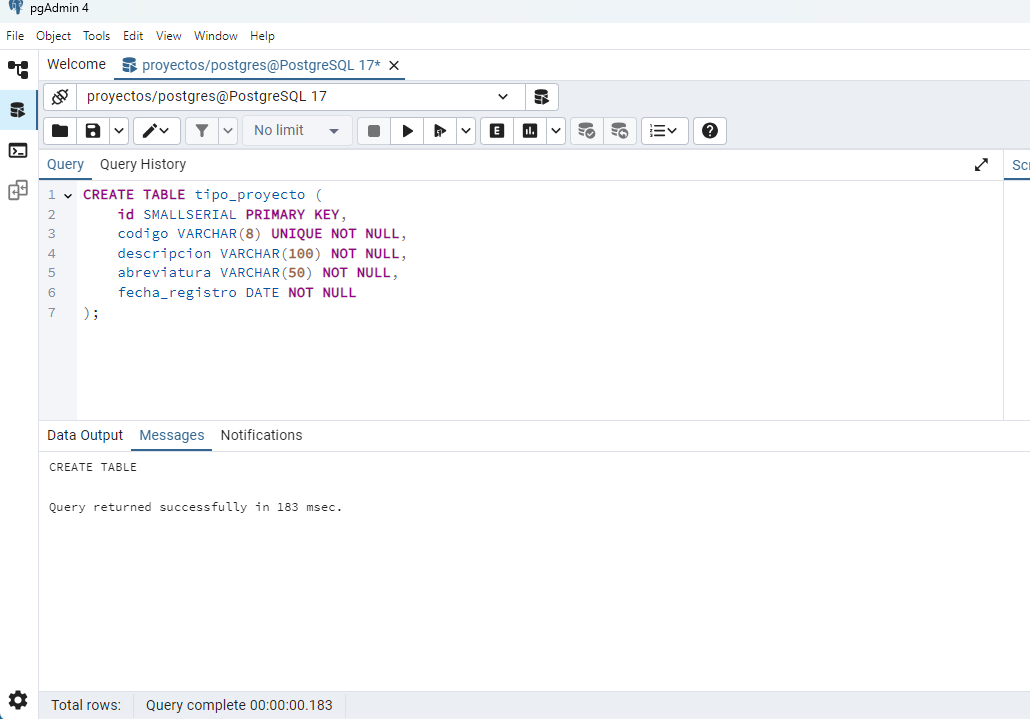
* Los resultados de la Tarea se deben subir al repositorio GIT de tareas de la asignatura en la carpeta Tarea-3 (TIA-3). Nota. Permite el acceso al público y al docente para acceder y ver el contenido.
* En el Classroom solamente colocarán UN SOLO LINK al repositorio. Lo colocará el líder del grupo

**Productos (entregables):**

* Plantilla de Informe con los resultados
* Proyecto Node con conexión y actualización de base de datos (carpetas y sub carpetas)
* Repositorio GIT con las Tareas: TIA-1, TIA-2 y TIA-3
* Equipo de miembros en la página principal del GIT
* Breve descripción del Proyecto PIA en el GIT

**1.- Base de Datos “proyectos” y Tabla de Tipos de Proyecto**

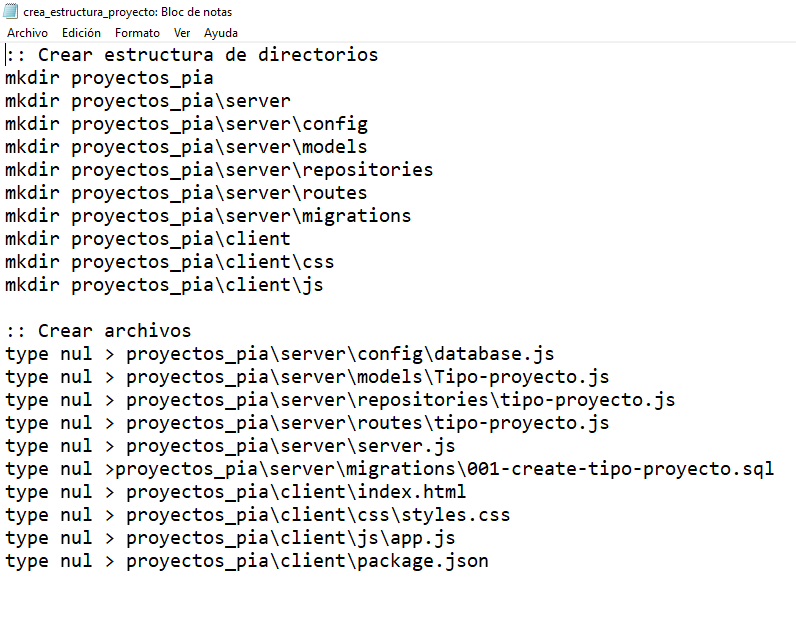
|  | **Tabla** | **tipo\_proyecto** |  | | | |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Descripción** | Tipos de proyecto de Aula que desarrollan los estudiantes durante el semestre | | | | |
| # | **Campo** | **Descripción** | **Tipo Dato** | **Tamaño** | **Restricciones** | **Tablas**  **Relacionada** |
| 1 | **id** | Id de identificación | smallserial |  | Primary Key (PK) |  |
| 2 | **codigo** | Código tipo proyecto | varchar | 8 | Unique Key (UK) |  |
| 3 | **descripcion** | Descripción del tipo de proyecto | varchar |  | Not Null |  |
| 4 | **abreviatura** | Nombre abreviado | varchar | 6 | Not Null |  |
| 5 | **fecha\_registro** | Fecha/hora de inserción |  |  |  |  |

**

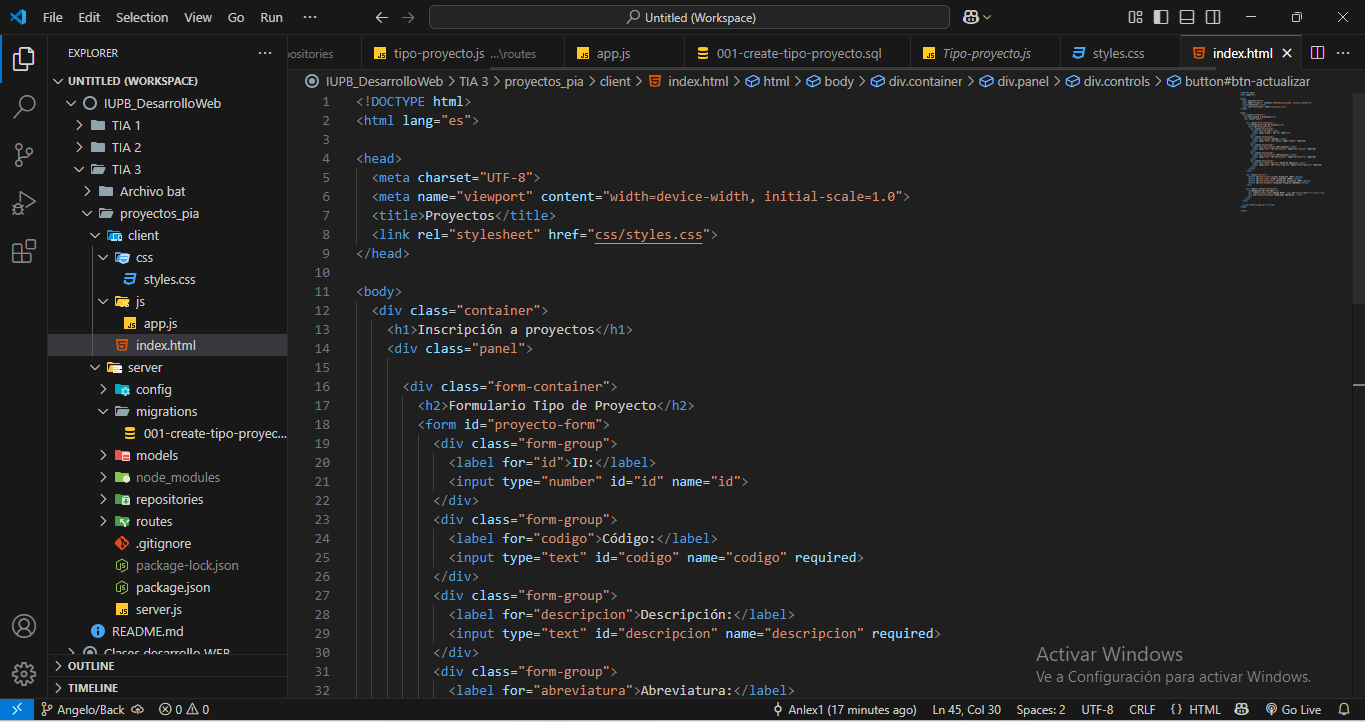
**2.- Estructura del proyecto FrontEnd/BackEnd**

**Estructura archivo .bat:**

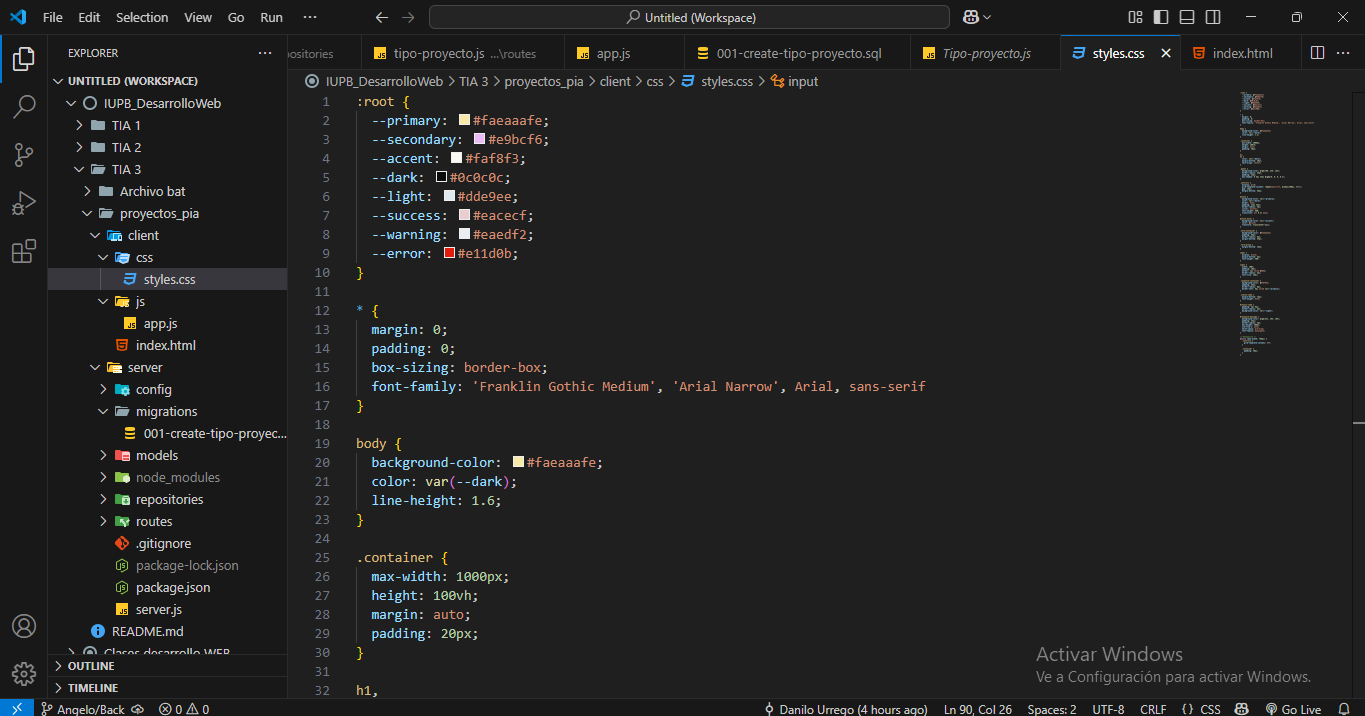
Este archivo tiene la función de crear el directorio de archivos que contendrá nuestro proyecto, tanto de la parte del frontend(client) como del backend(server), además de crear la estructura de carpetas, también creará los archivos de diferentes extensiones como lo son js, css, html y json.

**

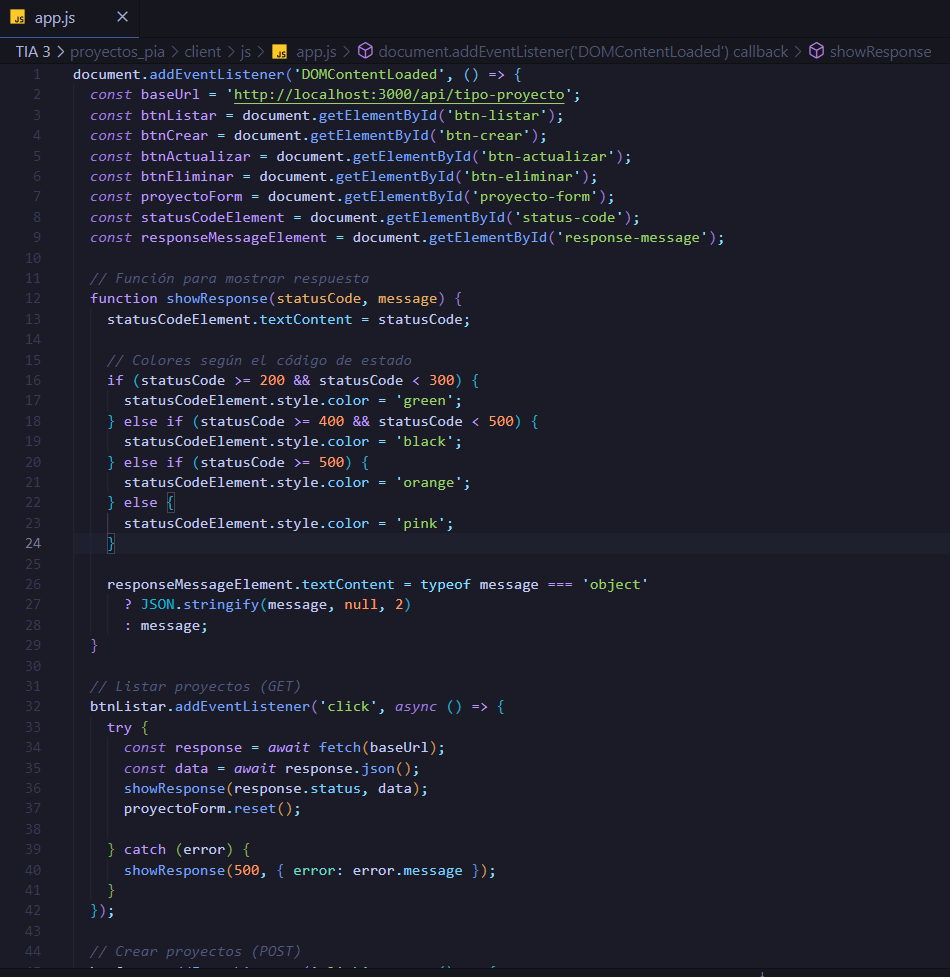
**3.- Código fuente del Proyecto**

**

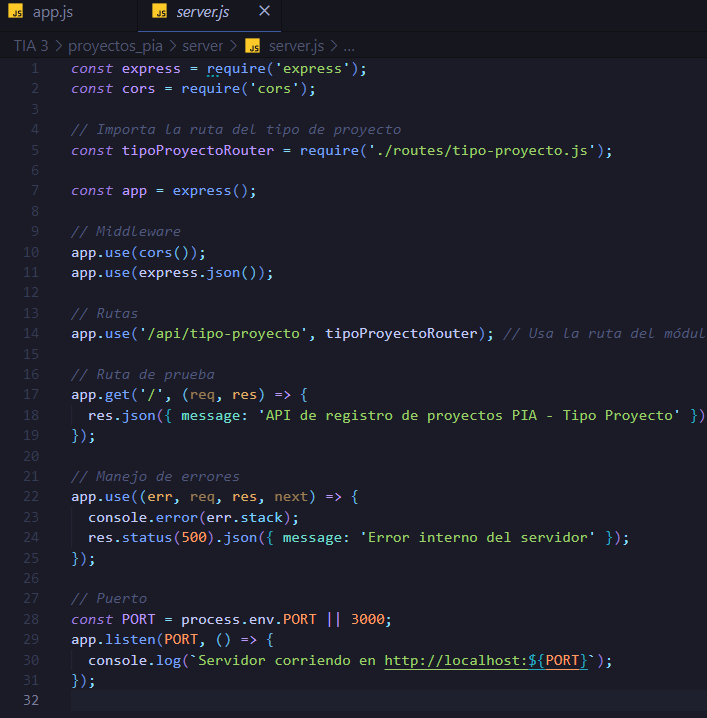
Este código HTML crea una página para registrar y gestionar proyectos. Incluye un formulario donde se ingresan datos como ID, código, descripción, abreviatura y fecha de registro. También hay botones para realizar operaciones tipo API: listar (GET), crear (POST), actualizar (PUT) y eliminar (DELETE) proyectos. Al final, muestra la respuesta del servidor con el código HTTP y un mensaje. Todo se complementa con un archivo CSS para el estilo y un archivo JS para la lógica.

**

El archivo style.css define el estilo visual de una página web del lado del cliente. Usa variables CSS para definir colores reutilizables (como primario, secundario, éxito, error, etc.). Establece una apariencia limpia y organizada para elementos como el cuerpo de la página (body), contenedores (.container, .panel), botones (button), formularios (input, label) y áreas de respuesta (.response-container). También incluye una sección responsive que adapta el diseño para pantallas pequeñas, como las de celulares. En general, este código mejora la presentación y la usabilidad de una interfaz web.

**

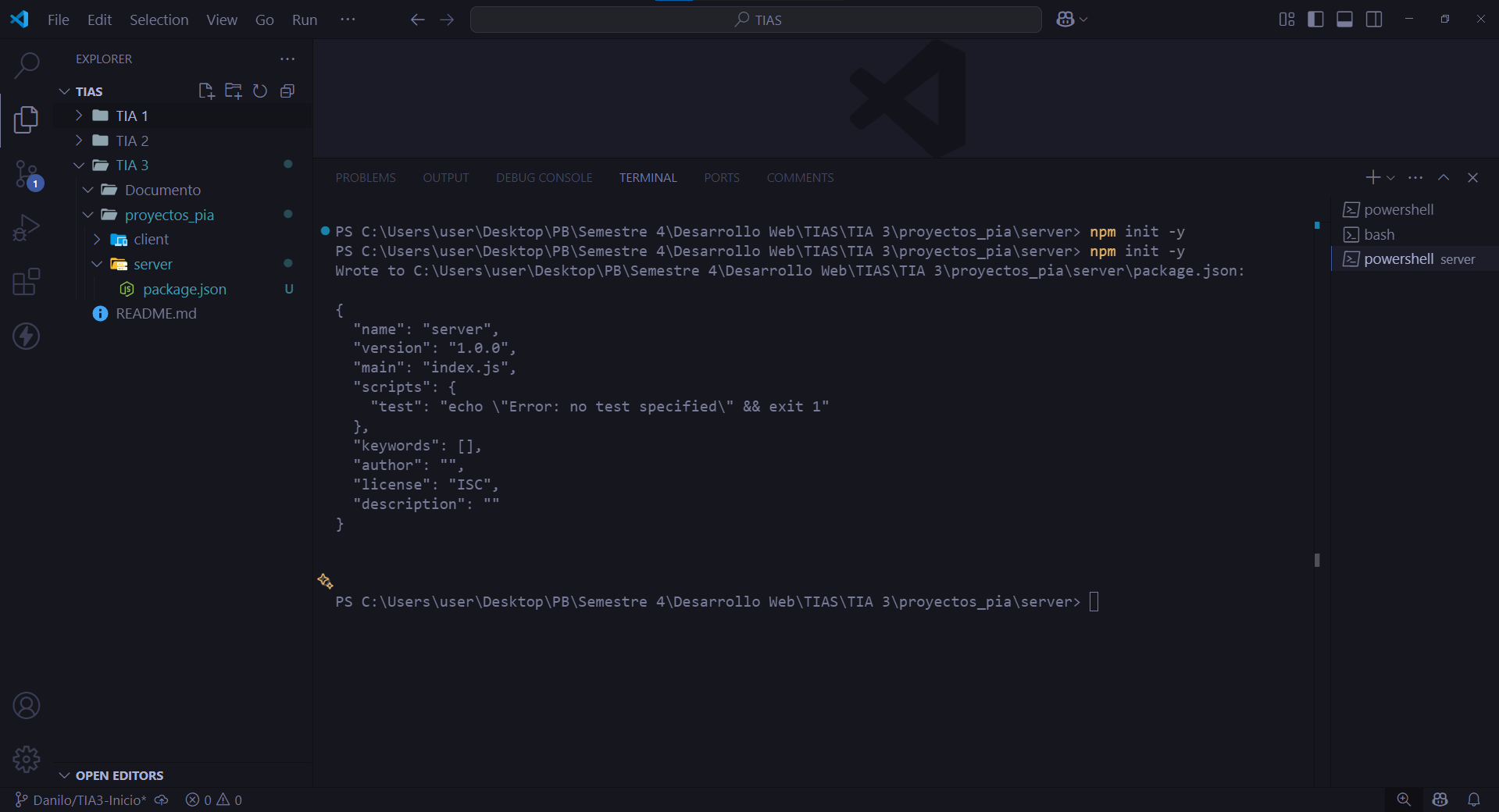
Este código en el frontend realiza la petición al backend usando javascript.

**

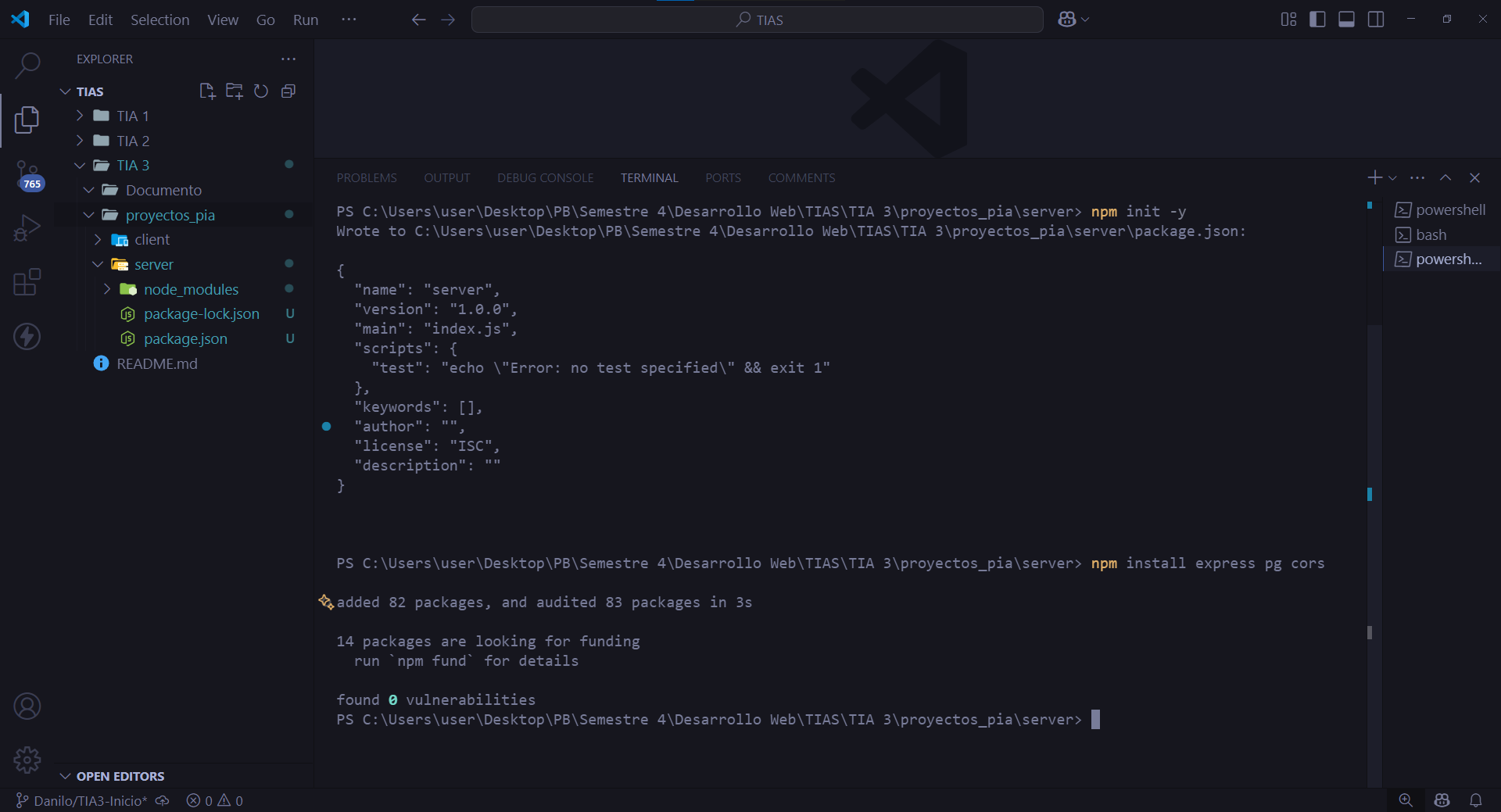
Este código en el backend es el archivo principal donde se define el puerto y se importan librerías como express o cors.

**4.- Creación del proyecto Node e instalación de las dependencias iniciales**

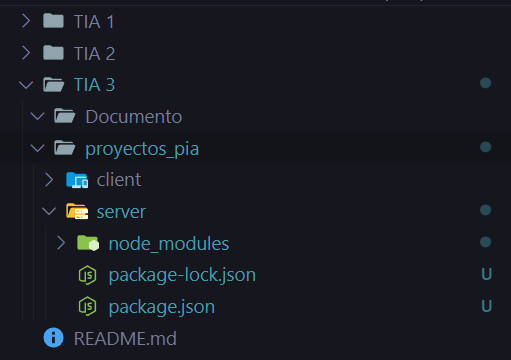
Pantallazo creación del proyecto en consola: Se crea el proyecto de node con el comando npm init -y

****

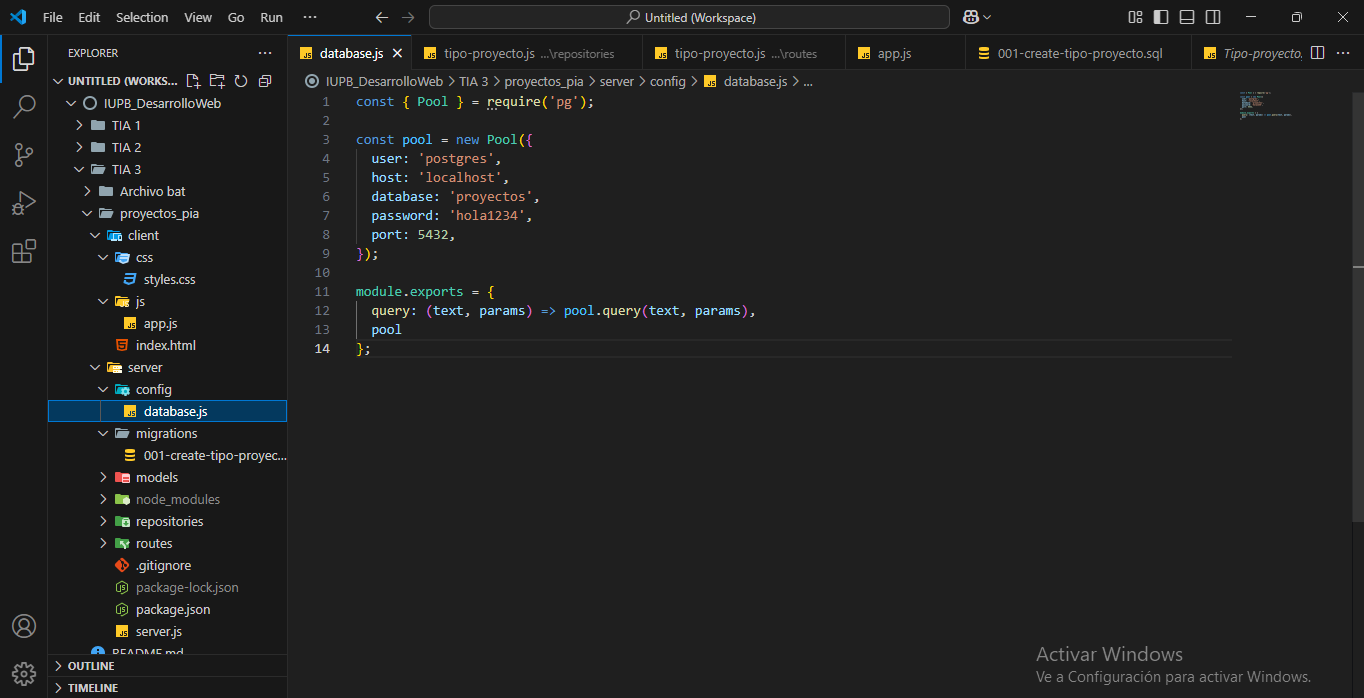
Pantallazo de instalación de dependencias del proyecto: Se instalan las dependencias express, pg (postgres) y cors.

****

Contenido de carpeta luego de las instrucciones: Se crea la estructura de directorios.

****

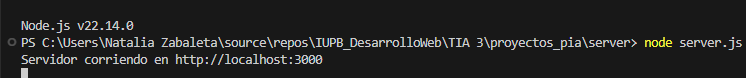
**5.- Configuración de la base de datos**

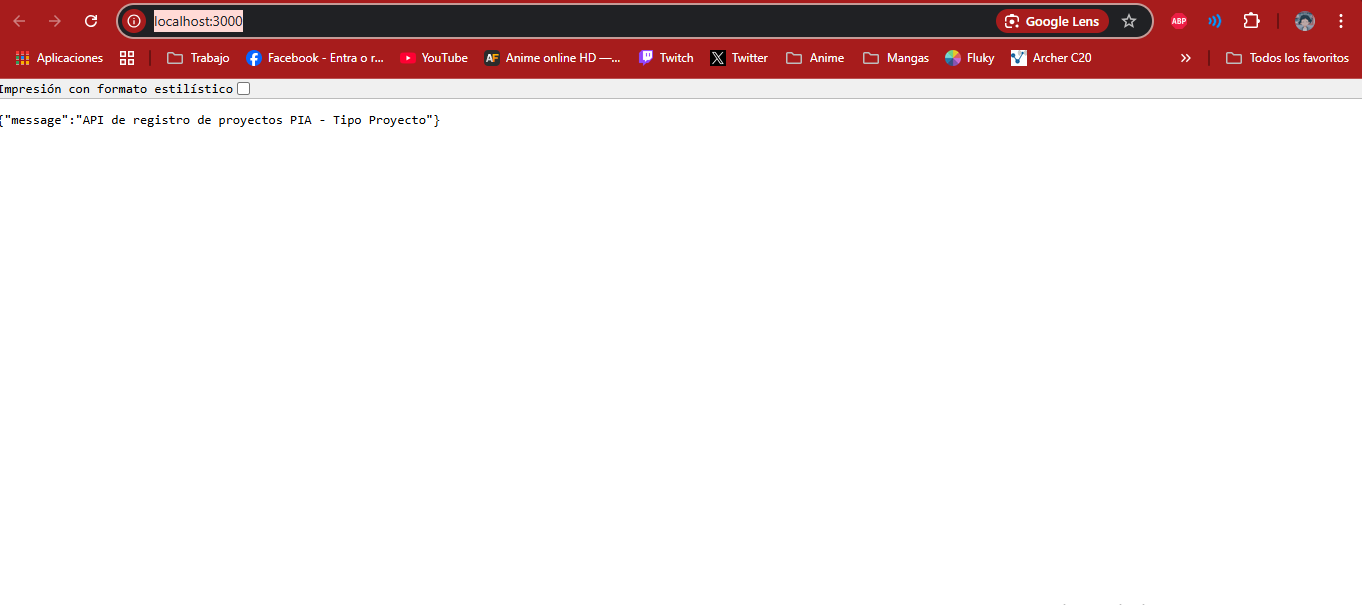
****

Este código permite conectar una aplicación Node.js con una base de datos PostgreSQL. Define los datos de acceso (usuario, contraseña, base de datos, etc.) y exporta funciones para hacer consultas. Es importante porque sin esta conexión no se podría guardar ni obtener información desde la base de datos.

**6.- Arranque del Servidor Node**

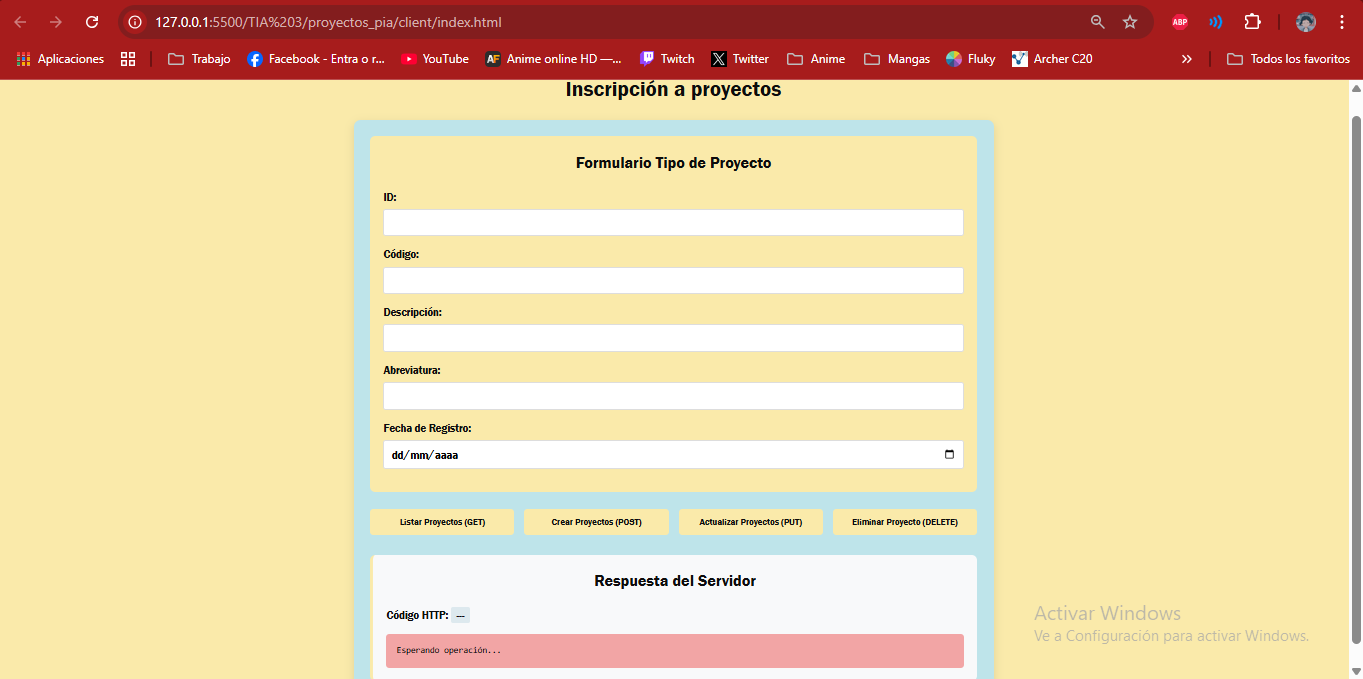
Se ejecuta el servidor a través de la consola de comandos usando “node server.js”, podemos acceder al server a través de la url localhost:3000.

****

****

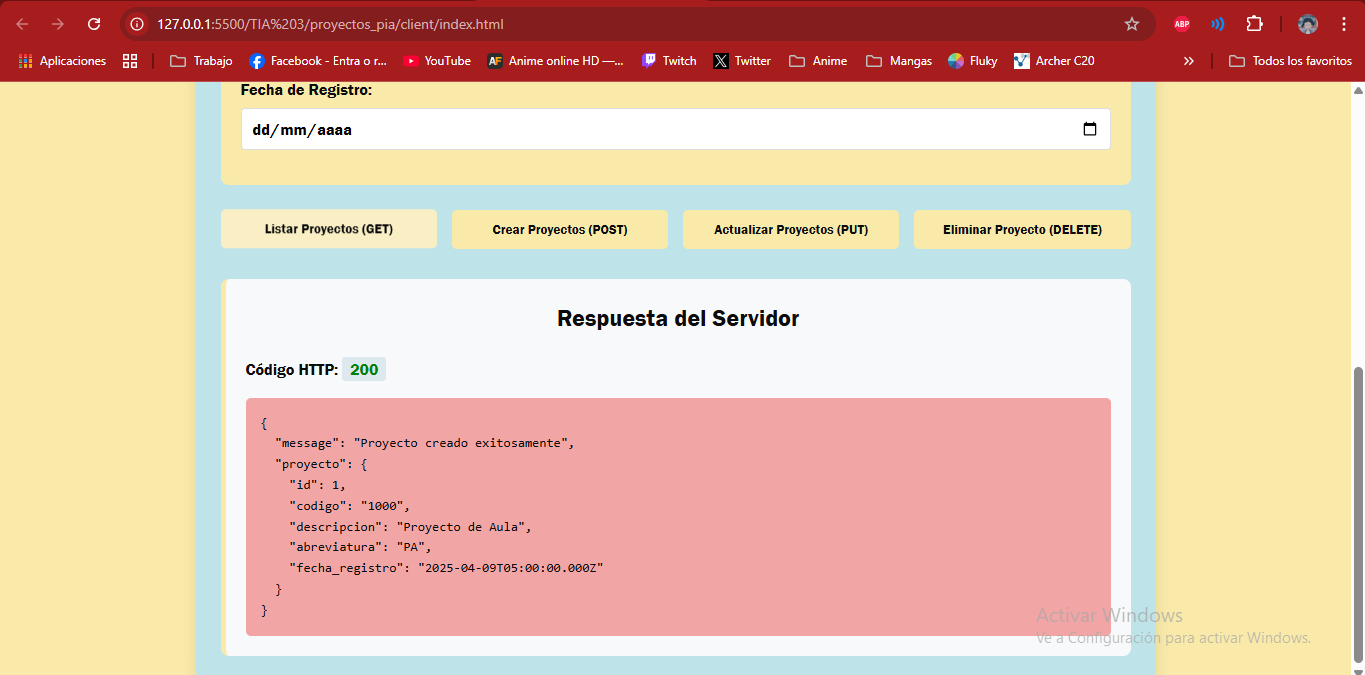
**7.- Arrancar la página Web**

La página inició en un puerto local y se ve visualmente lo realizado con html y css.

****

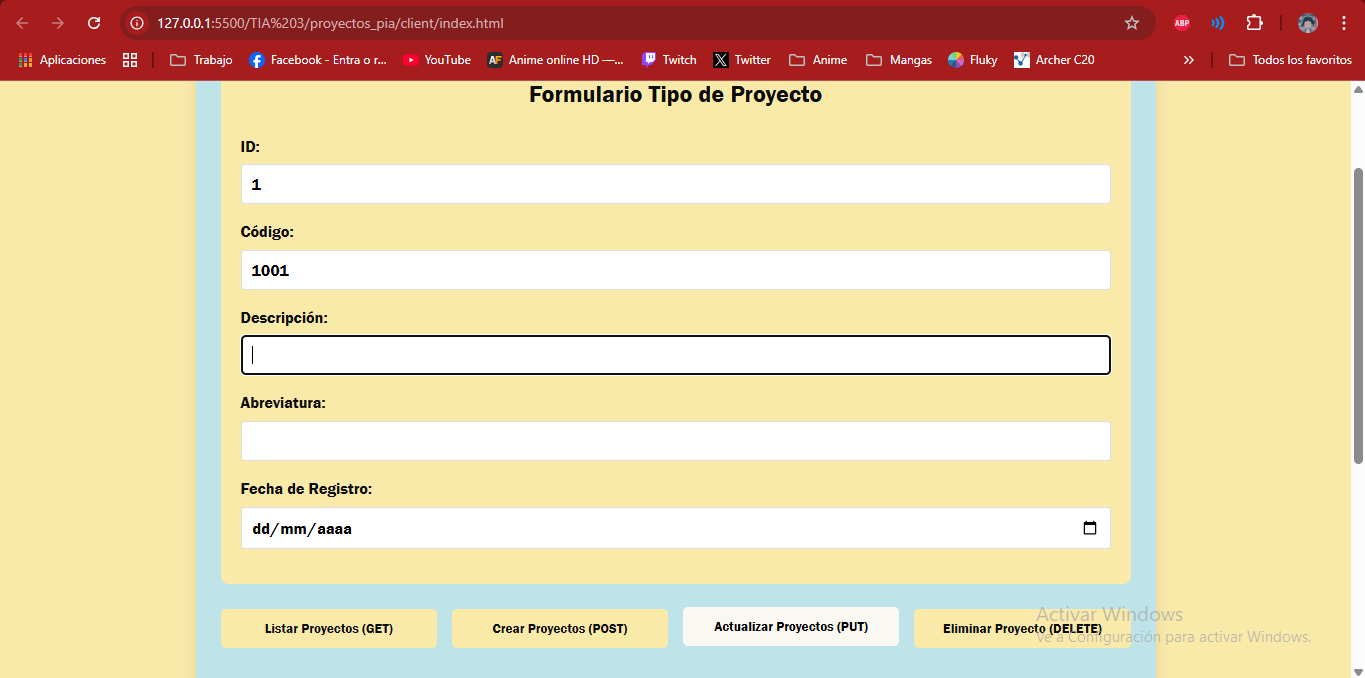
**8.- Inserción de registros (POST/INSERT)**

****

****

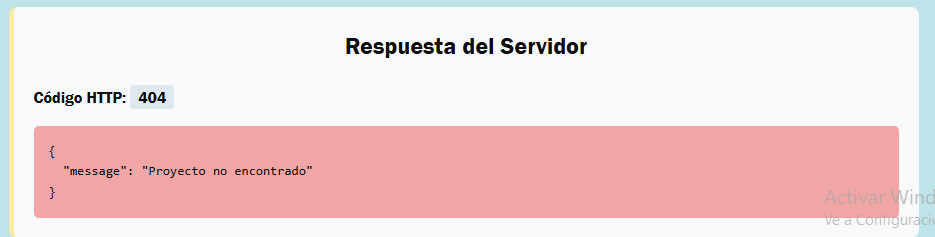
Se agregan los registros solicitados en el enunciado a la base de datos a través del cliente (página web)

**9.- Modificación de registros (PUT)/UPDATE)**

****

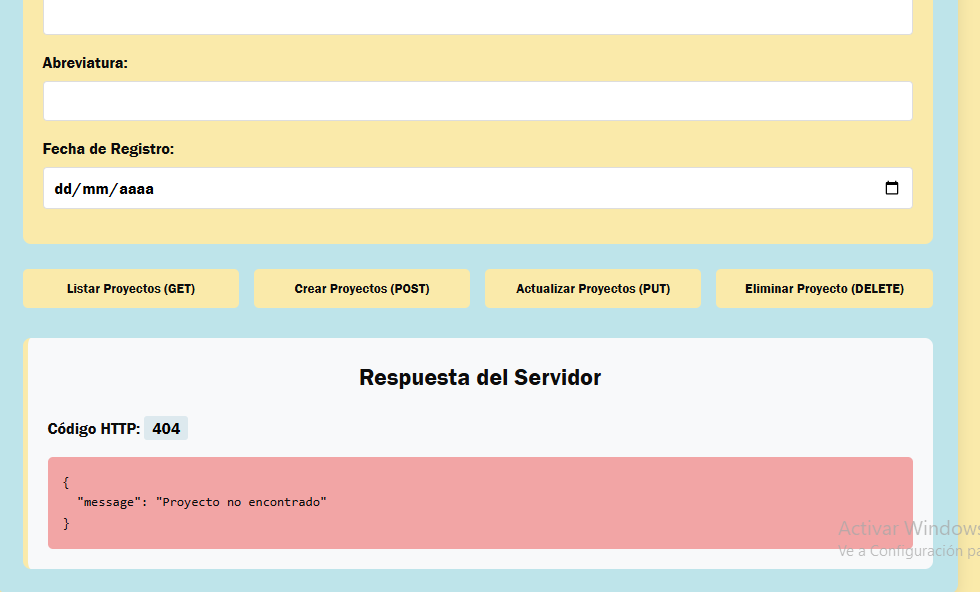
****

****

****

Se realiza una modificación usando el método PUT, la cual según el id del tipo de proyecto realizará los cambios según los campos en el formulario. A su vez validará si existe o no un proyecto.

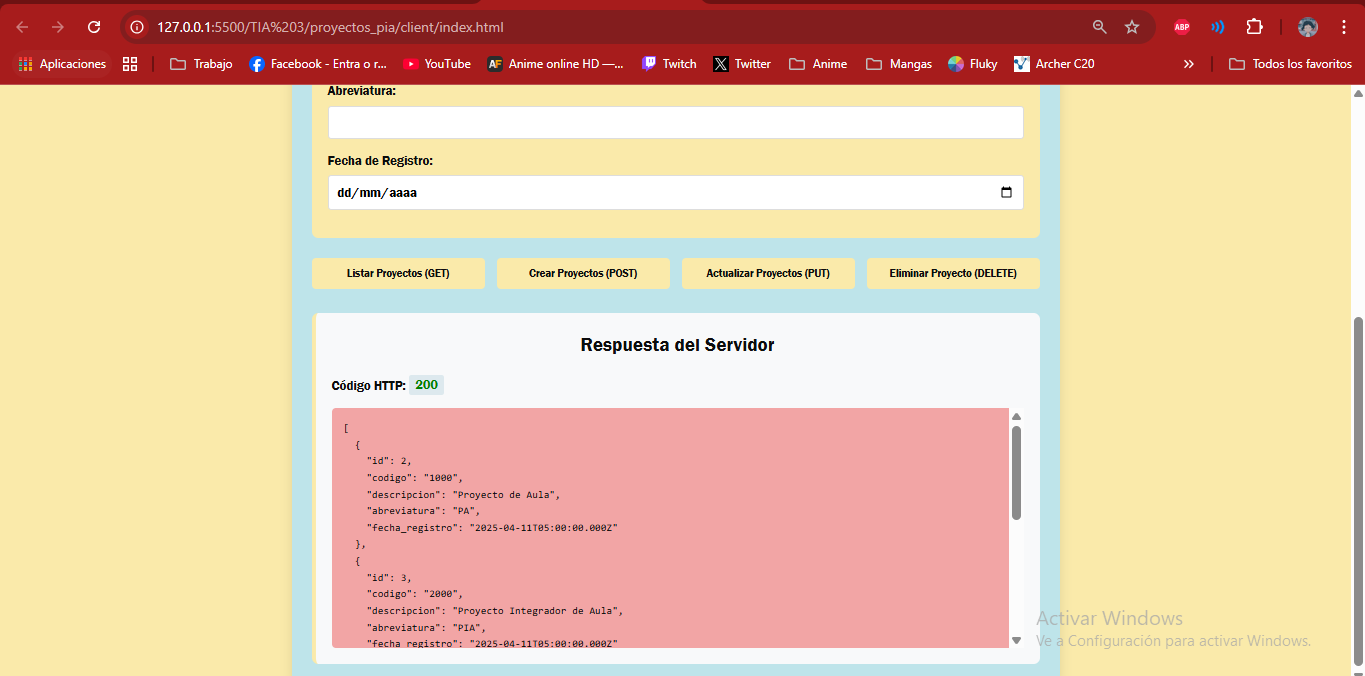
**10.- Eliminación de un registro (DELETE/DELETE)**

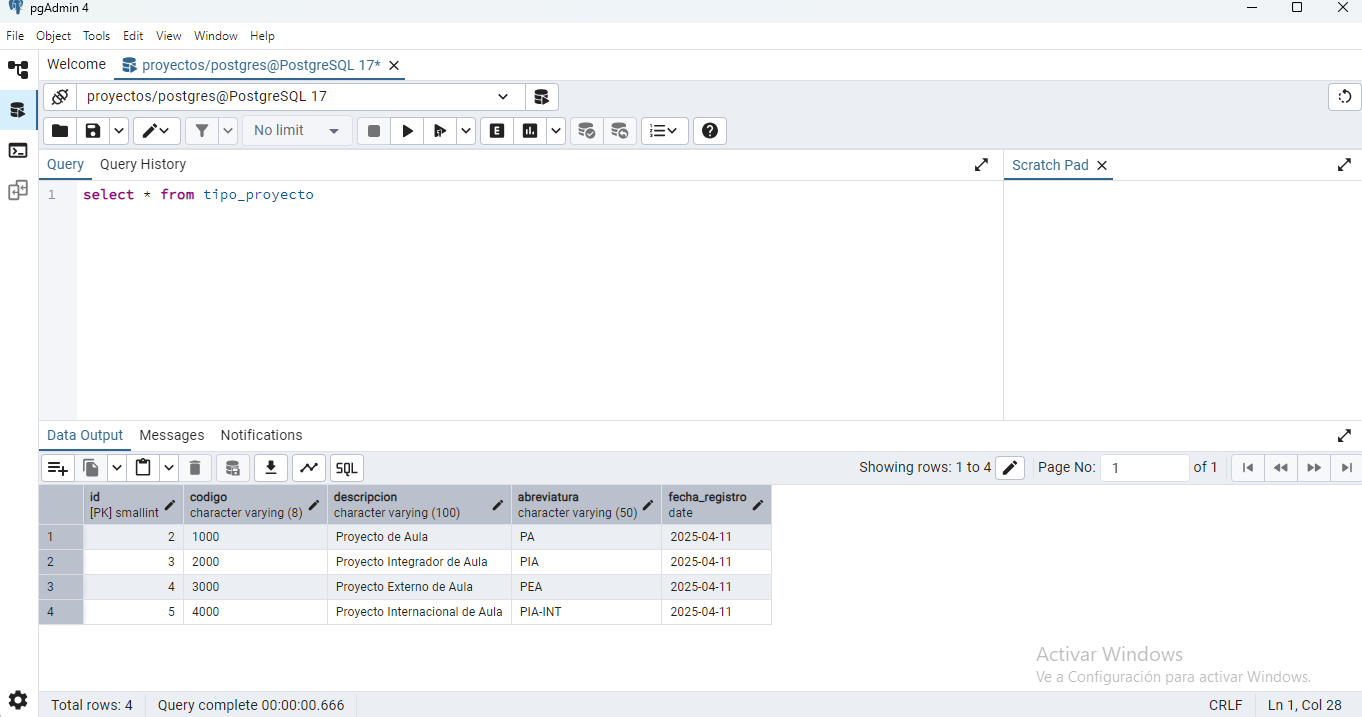
****

****

Se realiza una modificación usando el método DELETE, la cual según el id del tipo de proyecto eliminará el registro de la base de datos. A su vez validará si existe o no un proyecto con el id.

**11.- Consulta de registros (GET/SELECT)**

****

****

Con este último método (GET), podemos acceder a todos los registros de la base de datos listandolos en un array de objetos de “tipo proyecto”, podemos ver los mismos resultados en pgAdmin.

**12.- Conclusiones Individuales**.

**Conclusiones**:

**Danilo Urrego**: El trabajo se postergó mucho por algún motivo después de haber hablado hacer las cosas con tiempo y más organizados, aún así creo que nos separamos mejor las tareas pero falta mucho por mejorar.

**Angelo Arango**: En comparación al trabajo anterior esta vez nos pusimos dividir las tareas más equitativamente. pero aún nos falta más compromiso y responsabilidad al momento de realizar las entregas, en cuando al aprendizaje fue buena práctica ya que puse en práctica todo lo visto anteriormente en clase.

**13.- Calidad del Informe de la Tarea. Rúbrica Informe**

| **#** | **Características del Informe** | **Puntos** | **Calificación** |
| --- | --- | --- | --- |
| 1 | * Presenta informe en la plantilla suministrada de forma correcta * Coloca el informe en la raíz de la carpeta TIA3 en el repositorio GIT * Se presentan todos los miembros del equipo con su nombre * Cubre la totalidad de los ítems de la tarea | **20** |  |
| 2 | * Escribe el nombre del video correctamente (colocando la letra del equipo en la “X”): ***20251-et0179-TIA3-equipo-x-informe.docx*** | **5** |  |
| 3 | * Presenta cuadros centrados, sin “dividirlos” entre saltos de página, sin distorsiones, letra ni tan grande ni tan pequeñita, con colores agradables. * Presenta figuras (pantallazos) centradas y de buena calidad | **50** |  |
| 4 | * Calidad general (tipo de letra, redacción, organización, figuras) * Redacta el informe con buena ortografía, gramática y expresión de ideas | **25** |  |
|  | **Total** | **100** |  |

**14.- Video de Sustentación**. **Rúbrica Video de Sustentación**

| **#** | **Características del Video de Sustentación** | **Puntos** | **Calificación** |
| --- | --- | --- | --- |
| **1** | * Se presentan todos de manera individual mostrando su cara, diciendo su nombre y describiendo la actividad que van a presentar * Se cubre la totalidad de los ítems de la tarea | **10** |  |
| **3** | Muestra de código en ejecución en PostgreSQL. Script SQL (pgAdmin4) | **5** |  |
| **3** | Muestra de código en ejecución del lado del cliente | **25** |  |
| **4** | Muestra de código en ejecución del lado del servidor | **25** |  |
| **5** | Describe la estructura del proyecto Node FrontEnd/BackEnd | **10** |  |
| **6** | Presenta breve conclusión individual sobre la tarea | **10** |  |
| **7** | Presenta video con calidad tanto de sonido como de visualización. Cada participante muestra su rostro claramente, se escucha bien el audio y se ve lo que presenta. | **10** |  |
| **8** | * Presenta enlace de acceso al video en el informe y en el repositorio * Escribe el nombre del video correctamente (colocando la letra del equipo en la “X”) ***20251-et0179-TIA3-equipo-X-video*** | **5** |  |
|  | Total | **100** |  |
|  | *NOTA: El estudiante que no aparezca, no tendrá evaluación de la tarea. ES OBLIGATORIA LA PARTICIPACIÓN EN EL VIDEO. Si por razones de “fuerza mayor” no puede aparecer en el video en conjunto, presentará un video individual exponiendo parte del trabajo.* | **0** | **0** |

**15.- Repositorio GIT. Rúbrica Repositorio GIT**

| **#** | **Características del Video de Sustentación** | **Puntos** | **Calificación** |
| --- | --- | --- | --- |
| **1** | Crea y presenta un repositorio con un nombre que identifique fácilmente el equipo de estudiantes | **20** |  |
| **2** | Presenta breve descripción del contenido del repositorio (Proyectos PA/PIA) | **5** |  |
| **3** | Se presentan los miembros del equipo en la primera página del repositorio | **5** |  |
| **4** | Repositorio organizado por carpetas de Tareas de esta manera  Tarea-1 o TIA-1  Tarea-2 o TIA-2  Tarea-3 o TIA-3  Tarea-4 o TIA-4  Tarea-5 o TIA-5  Tarea-5 o TIA-5 | **10** |  |
| **5** | Tarea 1 (TIA1) subida al repositorio | **10** |  |
| **6** | Tarea 2 (TIA2) subida al repositorio | **10** |  |
| **7** | Tarea 3 (TIA3) subida al repositorio (esta tarea) | **20** |  |
| **8** | Repositorio permite el acceso al público, el docente y todos los estudiantes  (modo lectura) | **5** |  |
| **9** | Cada tarea presenta un enlace al video de sustentación. | **5** |  |
| **10** | La Tarea-3 debe estar organizada en subcarpetas y archivos de esta manera:  Archivos en la raíz de la carpeta Tarea-4   * Informe * Enlace video * Subcarpetas “proyecto\_pia” dentro de la Tarea-4   + Cliente   + Servidor | **10** |  |
|  | **Total** | **100** |  |

**Rúbrica de la Tarea: Criterios de Evaluación**

| **ítem** | **Criterio** | | | **Peso** | **Cal** |
| --- | --- | --- | --- | --- | --- |
| **1** | **Base de Datos “proyectos” y Tabla de Tipos de Proyecto** | | | **20** |  |
| **2** | **Estructura del proyecto FrontEnd/BackEnd** | | | **30** |  |
| **3** | **Código fuente del Proyecto. Cambiar estilo Página Web** | | | **200** |  |
| **4** | **Creación del proyecto Node e instalación de las dependencias iniciales** | | | **10** |  |
| **5** | **Configuración de la base de datos** | | | **10** |  |
| **6** | **Arranque del Servidor Node** | | | **15** |  |
| **7** | **Arrancar la página Web** | | | **15** |  |
| **8** | **Mostrar ejecución operación Rest - Inserción de registros (POST)** | | | **75** |  |
| **9** | **Mostrar ejecución operación Rest- Modificación de registros (PUT)** | | | **75** |  |
| **10** | **Mostrar ejecución operación Rest - Eliminación de un registro (DELETE)** | | | **75** |  |
| **11** | **Mostrar ejecución operación Rest - Consulta de registros (GET)** | | | **75** |  |
| **12** | **Conclusiones Individuales** | | | **50** |  |
| **13** | **Calidad del Informe** | | | **100** |  |
| **14** | **Video de sustentación** | | | **200** |  |
| **15** | **Repositorio** | | | **50** |  |
|  | **NOTA** |  | **TOTAL** | **1000** |  |