



## TECNOLOGICO NACIONAL DE MÉXICO

### INSTITUTO TECNOLÓGICO DE PABELLON DE ARTEAGA

#### **Desarrollo de base de datos y diseño de interfaz para un sistema de control de variables ambientales en una granja vertical para el cultivo de setas**

Ingeniería en Tecnologías de la Información y las Comunicaciones

Valeria Carolina Campos Hernández

Reporte final de residencia profesional

Dra. Pamela Romo Rodríguez

Ing. Oscar Lenin Espinoza Alvarez

Laboratorio de bio tecnología fúngica  
mayo de 2024





## Agradecimientos

Antes de todo, quiero agradecer a mi propia persona, recordarme a mí misma que, gracias a mi fuerza y perseverancia puedo estar escribiendo mi reporte final de proyecto, gracias por no rendirme en ningún momento y por siempre tener en mente que lo que hago es por mí, gracias por dudar y confiar ya que de eso aprendí de muchas personas y situaciones que me pusieron donde estoy.

Agradezco a mi mamá, es mi fuente de vida, mi ejemplo a seguir, la mujer que celebro y en la que pienso todos los días, gracias por darme el sustento para mis estudios y por nunca perder la esperanza en mí, termino este paso en mi vida y comienza uno más en donde espero poder ayudarte más con el sueño que tenemos en casa; gracias por darme paz, confianza, enojos y corajes ya que es por eso que conocemos más de nosotras.

Gracias César, por simplemente estar a mi lado cuanto lo necesito, espero que puedas quedarte mucho tiempo más, fuiste tan vital en este proceso que agradecerte es poco para mí, me ayudaste a no bajar mis estándares y a darme cuenta de que mis esfuerzos valen de verdad.

Celebro tener a mi familia y amigos, son todos parte de mi proceso en la carrera, Abuela Berta, Dana, Vanessa, Rominna, Martín, Astrid, todos mis tíos y tías, Karla, Alejandra, Ian, Anna Paula, Leo, Víctor, Diego, Omar, Armando, Fany.

También aplaudo y agradezco a todos mis profesores por el esfuerzo que pusieron en enseñarme lo que saben, desde mis profesores de primer semestre como Diana García que me convenció de que el cálculo sí te puede gustar y es tan fácil como separar manzanas verdes de las manzanas rojas, hasta Lenin Espinoza que me mostró que se puede ser profesional y exigente sin comprometer la propia salud, todos me mostraron los caminos que hay para poder desenvolverme en el área que más me gusta y también en lo que no me gusta.

Simplemente gracias a todas las personas que conozco por formar parte de mi vida dentro del instituto, sí, también a las personas que me hicieron la vida de cuadritos, por ustedes sé cómo no debo tratar a los demás y que seré mejor que ustedes.

Gracias a todos por existir y hacerme ver que todos llevamos nuestro ritmo en cada etapa, no todos pasamos por lo mismo y que el ritmo que llevo está bien, porque el pasado ya no está, el futuro es incierto y el presente es un regalo que debo disfrutar día a día.





## Resumen

En el siguiente reporte se encuentra detallado el proceso de desarrollo y la programación de la base de datos y la interfaz de control y monitoreo para la granja vertical, que conforma la integración entre el frontend y el backend, mediante el desarrollo de una Api RESTful que gestiona las peticiones del usuario para obtener los datos que se muestran en la interfaz de monitoreo y para enviar los datos del cambio de variables a la granja vertical desde la interfaz de control que se guardarán en la base de datos. Además, se demuestra el proceso para montar el sistema en una Raspberry Pi que se instalará directamente en la granja vertical. Aquí se documenta el proceso de desarrollo e implementación del sistema propuesto, presentando los resultados obtenidos y las lecciones aprendidas, con el objetivo de contribuir al avance y el control climático en cultivos especializados como el de setas.





## Índice

Agradecimientos .....	2
Resumen .....	3
<b>Introducción</b> .....	5
<b>Descripción de la empresa</b> .....	6
<b>Planteamiento del problema</b> .....	7
<b>Objetivos</b> .....	8
<i>Objetivo general</i> .....	8
<i>Objetivos específicos</i> .....	8
<b>Justificación</b> .....	9
<b>Hipótesis</b> .....	10
<b>Alcances y limitaciones</b> .....	10
<i>Alcances</i> .....	10
<i>Limitaciones</i> .....	10
<b>Marco teórico</b> .....	12
<i>Base de datos</i> .....	12
<i>Interfaz</i> .....	13
<b>Desarrollo</b> .....	16
<i>Cronograma de actividades</i> .....	16
<i>Procedimiento y descripción de las actividades realizadas</i> .....	16
<i>Configuración del entorno de desarrollo</i> .....	17
<i>Desarrollo del Backend</i> .....	17
<i>Desarrollo del Frontend</i> .....	20
<i>Integración del backend y el frontend</i> .....	25
<i>Monitoreo y control</i> .....	29
<i>Despliegue en la Raspberry Pi</i> .....	30
<b>Resultados</b> .....	31



<b>Conclusiones</b>	35
<b>Experiencia profesional adquirida</b>	35
<b>Fuentes de información</b>	¡Error! Marcador no definido.
<b>Referencias</b>	36

## Introducción

El presente reporte aborda la problemática de la gestión y control de los parámetros climáticos en una granja vertical para el cultivo de setas, proponiendo el desarrollo e implementación de un sistema de almacenamiento de datos y una interfaz de usuario fácil de manejar por el usuario, todo esto para facilitar la supervisión precisa y oportuna de las condiciones climáticas, lo que es crucial para optimizar el rendimiento y la calidad del cultivo. La metodología propuesta para abordar esta problemática incluye el análisis de requisitos, el diseño del sistema y la optimización y ajustes del mismo. Este enfoque integral busca asegurar que el sistema no solo cumpla con los requisitos técnicos, sino que también sea práctico y beneficioso en un entorno real de producción.





### ***Descripción de la empresa***

Laboratorio de biotecnología fúngica, ubicado en el Instituto Tecnológico de Pabellón de Arteaga, Ags. Tiene como misión, brindar un servicio de educación superior de calidad comprometido con la generación, difusión y conservación del conocimiento científico tecnológico y humanista, a través de programas educativos que permitan un desarrollo sustentable, conservando los principios universales en beneficio de la humanidad





## ***Planteamiento del problema***

Actualmente, los usuarios de una granja vertical para el cultivo de setas enfrentan dificultades significativas debido a la falta de un sistema adecuado que gestione y controle los cambios climáticos y la información emergente del cultivo. Esta carencia impide que los usuarios accedan de manera rápida y organizada a datos críticos sobre las condiciones ambientales dentro de la granja vertical, lo que limita la capacidad para tomar decisiones informadas y optimizar el rendimiento del cultivo.







## **Objetivos**

### **Objetivo general**

Desarrollar e implementar un sistema de almacenamiento y una interfaz eficiente y adecuada que permita la gestión y control de los parámetros climáticos de una granja vertical de cultivo de setas, con el fin de proporcionar a los usuarios un acceso rápido, organizado y fiable a esta información.

### **Objetivos específicos**

- Recopilación de Requisitos: Identificar y documentar exhaustivamente los requisitos del sistema, enfocándose en las necesidades específicas de monitoreo y control de variables ambientales para el cultivo de setas en una granja vertical.
- Diseño y desarrollo de Base de Datos: Desarrollar una base de datos eficiente que permita almacenar y gestionar información relevante sobre variables ambientales en una granja vertical para el cultivo de setas.
- Diseño y desarrollo de Interfaz de Usuario: Crear una interfaz de usuario intuitiva y amigable que permita a los usuarios visualizar, gestionar y controlar los datos recopilados sobre las variables ambientales dentro de la granja vertical.







### ***Justificación***

El cultivo de *Pleurotus ostreatus* en granjas verticales enfrenta desafíos significativos relacionados con la gestión y el control de las condiciones climáticas. Actualmente, existe carencia de infraestructura adecuada que permita la monitorización precisa y el control eficiente de los parámetros ambientales en estas instalaciones especializadas.

La falta de un sistema integrado de almacenamiento de datos y una interfaz de usuario intuitiva dificulta la capacidad de los productores para gestionar exitosamente las variables críticas del entorno, como la temperatura, la humedad y la iluminación. Esta deficiencia puede tener consecuencias negativas en la calidad y el rendimiento del cultivo en cuestión.

Por tanto, la implementación de un sistema de almacenamiento y control climático diseñado para granjas verticales de cultivo de setas es una necesidad clara. Este proyecto tiene como objetivo abordar esta problemática desarrollando una solución integral que permita la recopilación, almacenamiento y gestión de datos sobre variables ambientales clave.

Al desarrollar una base de datos y una interfaz de usuario eficiente e intuitiva, los productores podrán acceder de manera rápida y organizada a la información sobre las condiciones climáticas dentro de la granja vertical. Esto no solo facilitará la toma de decisiones en tiempo real, sino que también optimizará las condiciones del cultivo, mejorando la calidad y el rendimiento.

La implementación exitosa de esta solución no solo beneficiará a los productores, si no también sentará las bases para futuras investigaciones y prácticas en contextos educativos.

En resumen, el desarrollo de este sistema es fundamental para optimizar las condiciones del cultivo, mejorar la productividad y garantizar condiciones idóneas en el futuro.





## ***Hipótesis***

Si se desarrolla e implementa un sistema de almacenamiento eficiente y una interfaz de usuario adecuada para el control de los parámetros climáticos dentro de la granja vertical de cultivo de setas, entonces se observará una mejora significativa en la gestión y el monitoreo de las condiciones ambientales, lo que resultará en un aumento de la calidad y el rendimiento del cultivo de setas

## ***Alcances y limitaciones***

### ***Alcances***

- Desarrollar y completar el código de los controles climáticos para la granja vertical.
- Realizar pruebas exhaustivas de los controles climáticos para garantizar su funcionalidad y precisión.
- Programar la conexión de la interfaz de usuario con el código de los controles climáticos.
- Diseñar y desarrollar la interfaz de usuario para permitir el monitoreo y control de los parámetros climáticos de la granja vertical.
- Realizar pruebas de integración para asegurar que la interfaz de usuario funcione correctamente con los controles climáticos y el sistema de monitoreo.
- Capacitar a los usuarios finales sobre el uso adecuado de la interfaz de usuario y el sistema de control climático.

### ***Limitaciones***

- Retraso en el desarrollo del código de los controles: El retraso en la disponibilidad del código de los controles climáticos afectará el progreso del desarrollo, ya que la integración de estos controles es fundamental para el funcionamiento del sistema. Esto puede requerir ajustes en el





plan de implementación y la priorización de otras tareas mientras se espera la disponibilidad del código.

- Pruebas prolongadas de los controles: Dado que las pruebas de los controles climáticos pueden llevar más tiempo del previsto, existe el riesgo de que se retrase aún más el desarrollo y la integración del sistema en su conjunto.
- Dificultades en la programación de la conexión de la interfaz: La programación de la conexión entre la interfaz de usuario y el código de los controles climáticos puede resultar compleja y requerir un tiempo considerable. Se deben asignar recursos adicionales y expertos en programación para abordar estos desafíos de manera efectiva y minimizar el impacto en el cronograma general del proyecto.
- Falta de desarrollo de la interfaz y la codificación del monitoreo: la falta de desarrollo de la interfaz de usuario y la codificación del monitoreo representa una limitación significativa, ya que estas son partes clave del sistema que requieren tiempo y recursos adicionales para su implementación. Se debe priorizar estas tareas y asignar recursos adecuados para complementarlas dentro del plazo establecido.





## **Marco teórico**

### **Base de datos**

**Una base de datos** es un conjunto de información Registrada, organizada y estructurada dentro de tablas relacionadas entre sí de maneras específica para que su contenido pueda ser tratado y analizado de forma rápida y sencilla. Estas bases de datos en ocasiones se utilizan para almacenar y asegurar la integridad de los datos, ya sea de una empresa o sistema electrónico. Existen varios tipos de bases de datos, como las estáticas, que son de solo lectura y se utilizan para análisis, y las dinámicas, que permiten realizar consultas de datos, actualizaciones e incluso eliminar los datos.

**SQL** es un lenguaje estándar para acceder y manipular bases de datos, se utiliza para realizar operaciones como consultas, inserciones, actualizaciones y eliminaciones en los datos almacenados, con una estructura simple, permite crear y modificar la estructura de una base de datos con comandos como CERATE TABLE, ALTER TABLE y DROP TABLE, así como manipular los datos dentro de las tablas con los comandos SELECT, INSERT, UPDATE y DELETE. SQL se utiliza en sistemas como MySQL para almacenar y recuperar datos en sitios web y aplicaciones.

**MySQL** es un sistema de gestión de bases de datos relacionales y su objetivo principal es almacenar y administrar datos, esta herramienta funciona para crear bases de datos y organizarlos mediante relaciones entre las tablas, recibe solicitudes de los usuarios, estos envían instrucciones SQL a las tablas de la base de datos, las procesa y devuelve la información solicitada a los usuarios. Es ampliamente utilizado en aplicaciones web, sistemas empresariales y más.



**Apache** es un servidor web que responde a las solicitudes de contenido de los usuarios. Funciona como un intermedio entre el navegador del usuario y los archivos o aplicaciones alojadas en el servidor y permite ejecutar aplicaciones escritas en PHP, Python y más. Es fundamental para alojar aplicaciones web de forma local y brindar contenido a los usuarios.

**PHP** (Procesador de Hipertexto) es un lenguaje de programación interpretado diseñado específicamente para el desarrollo web puede incluirse en HTML para hacer las webs dinámicas, su sintaxis se basa en Java, con algunas características propias. PHP facilita la conexión con bases de datos como MySQL, procesando datos enviados desde formularios web.

**Laravel** es un marco de aplicación Web construido en PHP que proporciona una gestión de autenticación de usuarios y permisos, además, permite crear y modificar la estructura de la base de datos de manera controlada (migraciones), ofrece herramientas para validar que los formularios estén completados de forma correcta en los datos de entrada, envía notificaciones y correos electrónicos de manera sencilla y ejecuta tareas programadas en intervalos específicos.

**HTML** (Lenguaje de Etiquetas de Hipertexto) no es un lenguaje de programación; es un lenguaje de marcado que define la estructura del contenido de una página web. Se trata de una serie de elementos que contienen diferentes partes de la página web para que se vean o se comporten de una manera determinada.





**JavaScript** es un lenguaje de programación interpretado y orientado a objetos que se utiliza principalmente para añadir características interactivas a un sitio web, como ejemplo, para aplicar efectos de estilo; esto es fundamental para que los sitios web sean interactivos y llamativos.

- **Interpretado:** No necesitas compilarlo; los navegadores lo ejecutan directamente.
- **Orientado a objetos:** Permite crear objetos y manipularlos.
- **Versatilidad:** Puede interactuar con HTML para crear experiencias dinámicas en la web.
- **Dinámico:** Algo que cambia o se adapta en tiempo real.

**React** es una biblioteca de JavaScript para construir interfaces de usuario. Te permite crear componentes reutilizables y combinarlos para formar aplicaciones web interactivas. React se basa en la idea de componentes. Un componente es una pieza aislada de la interfaz de usuario que puede contener lógica y representación visual.

**TypeScript** es un lenguaje de programación que se basa en JavaScript que detecta errores tempranos en el editor de texto antes de ejecutar el código y se puede ejecutar en navegadores, Node.js; Además facilita la comprensión del código por parte de otros desarrolladores.

**Node.js** es un entorno de ejecución de JavaScript de código abierto gratuito que permite crear aplicaciones web y scripts, permite ejecutar código fuera del navegador, lo que lo hace ideal para el desarrollo del lado del servidor. Es utilizado para crear servidores web, APIs y aplicaciones en tiempo real y gracias a NPM (Node Package Manager - Manejador de paquetes de Node) es posible tener acceso a una gran cantidad de paquetes y bibliotecas esenciales para la programación, ya que permiten reutilizar código y acelerar el desarrollo de aplicaciones.







**Composer** es un gestor de dependencias para el PHP que ayuda a administrar las bibliotecas necesarias para proyectos y es muy utilizado en proyectos con Laravel.

**Axios** permite realizar solicitudes y recibe respuestas HTTP desde el navegador o el servidor.

**CORS** (Intercambio de recursos de origen cruzado) es un mecanismo basado en HTTP que permite a un servidor qué dominios o puertos puedan realizar solicitudes seguras entre navegadores; CORS se aplica a API como Fetch.

Un puerto es una interfaz a través de la cual se pueden enviar y recibir diferentes tipos de datos. Estos puertos son puntos de conexión que permiten la transferencia de información entre distintos dispositivos y redes, ya sea mediante cables o conexiones inalámbricas.

**JSON** (Notación de objeto de JavaScript) es un formato de texto sencillo para el intercambio de datos y se utiliza comúnmente para transmitir datos entre el servidor y el cliente en aplicaciones web.

**Python** es un lenguaje de programación versátil y poderoso que te permite trabajar de manera rápida e integrar sistemas de manera más efectiva y tiene una amplia gama de bibliotecas y marcos para diversos propósitos, como desarrollo web.







## Desarrollo

### Cronograma de actividades

Tabla 1 Cronograma de actividades

Actividades	Enero	Febrero	Marzo	Abril	Mayo	Junio
Planificación y Recopilación de Requisitos						
Diseño de Base de Datos e Interfaz de Usuario						
Desarrollo de interfaz y del Sistema unificando la Base de Datos						
Implementación del Sistema						
Integración y Pruebas						
Validación en Ambiente Real y Documentación Final						

### Procedimiento y descripción de las actividades realizadas

De acuerdo a los objetivos específicos, lo primero a realizar para el desempeño adecuado del proyecto es el levantamiento de requerimientos por parte de los usuarios de la interfaz, ya que por medio de una serie de preguntas cuidadosamente formuladas, se llegó a la conclusión de que el sistema sería montado en una microcomputadora Raspberry Pi para ser integrado a la granja vertical por lo que, a continuación se revisaron las posibles tecnologías factibles a implementar para el correcto funcionamiento del proyecto, dando como resultado la elección de MySQL para gestionar la base de datos y JavaScript con React como lenguaje para el desarrollo de la interfaz.

Teniendo el conocimiento de las herramientas a utilizar, el proceso para iniciar toma lugar con la instalación de estas herramientas desde la terminal de la Raspberry Pi con comandos para asegurarse de que ninguna paquetería fuera extraviada en la instalación.





### **Configuración del entorno de desarrollo**

Se configuró un entorno de desarrollo adecuado en la Raspberry Pi, incluyendo la instalación de apache y MySQL; Composer para gestionar las dependencias de PHP y Laravel.

### **Desarrollo del Backend**

Para el backend, se eligió Laravel debido a su capacidad para simplificar tareas comunes como la autenticación, el enrutamiento, las sesiones y el almacenamiento. Laravel se configuró para interactuar con MySQL y este a su vez almacena datos, como las configuraciones de la interfaz de control y los registros del monitoreo de las variables de temperatura y humedad.

A continuación, se muestra el código utilizado para la configuración de la conexión del backend con la base de datos desde el archivo. env y las migraciones para crear las tablas con sus respectivos campos, además de la programación de lo que realizarán las peticiones de cada tabla y sus rutas.





Ilustración 1 Configuración de conexión a base de datos

```
1 APP_NAME=Laravel
2 APP_ENV=local
3 APP_KEY=base64:yzM3c4Qzlo338Ycw+CF3M5vV1cgYI8XuQ7S3fhEJ6ws=
4 APP_DEBUG=true
5 APP_URL=http://localhost
6
7 LOG_CHANNEL=stack
8 LOG_DEPRECATIONS_CHANNEL=null
9 LOG_LEVEL=debug
10
11 DB_CONNECTION=mysql
12 DB_HOST=localhost
13 DB_PORT=3306
14 DB_DATABASE=BDIGVCS
15 DB_USERNAME=labbiofun
16 DB_PASSWORD=laboratoriid
17
18 BROADCAST_DRIVER=log
19 CACHE_DRIVER=file
20 FILESYSTEM_DISK=local
21 QUEUE_CONNECTION=sync
22 SESSION_DRIVER=file
23 SESSION_LIFETIME=120
24
25 MEMCACHED_HOST=127.0.0.1
26
27 REDIS_HOST=127.0.0.1
28 REDIS_PASSWORD=null
```

Ilustración 2 Migración de las tablas principales a la base de datos

```
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('fase', function (Blueprint $table) {
15             $table->id();
16             $table->string('fase')->nullable();
17             $table->timestamp('created_at')->nullable();
18             $table->timestamp('updated_at')->nullable();
19         });
20     }
21
22     /**
23      * Reverse the migrations.
24      */
25     public function down(): void
26     {
27         Schema::dropIfExists('fase');
28     }
29 }
```

```
1 <?php
2
3 use Illuminate\Database\Migrations\Migration;
4 use Illuminate\Database\Schema\Blueprint;
5 use Illuminate\Support\Facades\Schema;
6
7 return new class extends Migration
8 {
9     /**
10      * Run the migrations.
11      */
12     public function up(): void
13     {
14         Schema::create('parametro', function (Blueprint $table) {
15             $table->id();
16             $table->dateTime('dia')->nullable();
17             $table->integer('temperatura')->nullable();
18             $table->integer('humedad')->nullable();
19             $table->integer('iluminacion')->nullable();
20             $table->timestamp('created_at')->nullable();
21             $table->timestamp('updated_at')->nullable();
22             $table->unsignedBigInteger('fase_id');
23             $table->foreign('fase_id')->references('id');
24         });
25     }
26
27     /**
28      * Reverse the migrations.
29      */
30 }
```





Ilustración 3 Programación de la tabla de fases

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Fase;
6 use Illuminate\Http\Request;
7
8 class FaseController extends Controller
9 {
10     /**
11      * Display a listing of the resource.
12      */
13     public function index()
14     {
15         $fas = Fase::all();
16         return response()->json($fas, 200);
17     }
18
19     /**
20      * Display the specified resource.
21      */
22     public function show(string $id)
23     {
24         $showfas = Fase::find($id);
25         return response()->json($showfas, 200);
26     }
27 }
28
```

Ilustración 4 Programación de la tabla de parametros

```
1 <?php
2
3 namespace App\Http\Controllers;
4
5 use App\Models\Parametro;
6 use Illuminate\Http\Request;
7
8 class ParametroController extends Controller
9 {
10     /**
11      * Display a listing of the resource.
12      */
13     public function index(Request $request)
14     {
15         // Obtener las fechas proporcionadas por el usuario desde la solicitud
16         $fechaInicio = $request->input('fecha_inicio');
17         $fechaFin = $request->input('fecha_fin');
18
19         // Filtrar los datos por el rango de fechas especificado
20         $parametros = Parametro::whereBetween('dia', [$fechaInicio, $fechaFin])->get();
21
22         // Retornar los datos filtrados
23         return response()->json($parametros);
24     }
25
26     /**
27      * Display the specified resource.
28      */
29 }
30
```





Ilustración 5 Rutas que procesan las peticiones

```
1 <?php
2
3 use App\Http\Controllers\AuthController;
4 use App\Http\Controllers\FaseController;
5 use App\Http\Controllers\UserController;
6 use App\Http\Controllers\ParametroController;
7 use App\Http\Controllers\ValuesController;
8 use App\Models\Parametro;
9 use Illuminate\Http\Request;
10 use Illuminate\Support\Facades\Route;
11
12 /*
13  * API Routes
14  *
15  * Here is where you can register API routes for your application. These
16  * routes are loaded by the RouteServiceProvider and all of them will
17  * be assigned to the "api" middleware group. Make something great!
18  */
19
20 Route::get('/parametro/{id}', [ParametroController::class, 'show']);
21 Route::get('/parametro', [ParametroController::class, 'index']);
22 Route::get('/fase/{id}', [FaseController::class, 'show']);
23 Route::get('/fase', [FaseController::class, 'show']);
24 Route::post('/update-settings', [ValuesController::class, 'update']);
```

## Desarrollo del Frontend

El frontend se desarrolló usando React JS y TypeScript para asegurar una experiencia de usuario dinámica. React JS permitió crear componentes reutilizables y administrar el estado de la aplicación de manera eficiente.

Enseguida se muestran los componentes y funciones que hacen posible la interfaz de control de variables de temperatura y humedad y la interfaz de monitoreo.





Ilustración 6 Funcionamiento del botón Modo oscuro

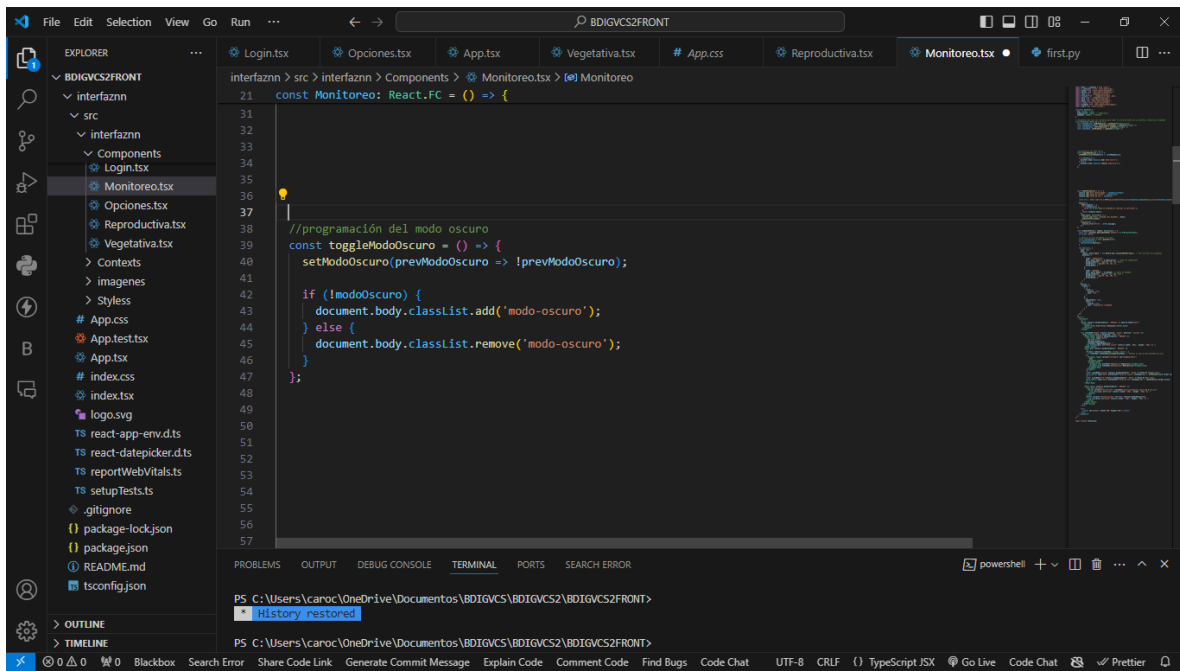






Ilustración 7 Primera parte de la composición de la página de control

```

14 function App() {
70
71   return (
72     <Container>
73       <br />
74       <Navbar style={{ backgroundColor: '#032634' }} data-bs-theme="dark">
75         <Container>
76           <Navbar.Brand href="#home">BioSystem</Navbar.Brand>
77         </Container>
78       </Navbar>
79
80       <div className="modal" style={{ display: 'block', position: 'initial' }}>
81         <Modal.Dialog style={{ margin: '10vh' }}>
82           <Modal.Header style={{ backgroundColor: '#032634' }}>
83             <Form.Check type="checkbox"
84               id="modo-oscuro"
85               checked={modoOscuro}
86               onChange={toggleModoOscuro} />
87             <img src={eclipse} alt="Modo oscuro" style={{ width: '20px', height: '20px' }} />
88           </Modal.Header>
89           <Modal.Body style={{ backgroundColor: '#032634' }}>
90             <br />
91             <div>
92               <label className="temperature-label">{temperature}°</label>
93             <br />
94             <Button variant="secondary" onClick={increaseTemperature}>+</Button>
95             <Button variant="secondary" onClick={decreaseTemperature}>-</Button>
96           </div>
97           <img src={temperatura} alt="temperatura" style={{ width: '20px', height: '20px' }} />
98           <br />
99           <br />
100          <div>
101            <label className="temperature-label">{humidity}%</label>
102          </div>
103        </Modal.Body>
104      </Modal.Dialog>
105    </div>
106  );
107 }
108
109 export default App;

```

Ilustración 8 Segunda parte de la composición de la página de control

```

109
110   <Modal.Footer style={{ backgroundColor: '#032634' }}>
111     <Link to="/opciones">
112       <Button variant="outline-info" className="position-absolute top-0 end-0 m-2 p-1">
113         <img src={casa} alt="casa" style={{ width: '20px', height: '20px' }} />
114       </Button>
115     </Link>
116     <Button variant="outline-success" onClick={applyChanges}>
117       <img src={play} alt="play" style={{ width: '20px', height: '20px' }} />
118     </Button>
119   </Modal.Footer>
120 </Modal.Dialog>
121 </div>
122 </Container>
123
124 );
125 }
126
127 export default App;
128

```







Ilustración 9 Configuración para la correcta visualización de la gráfica en el componente del monitoreo

```
interfazn > src > interfazn > Components > Monitoreo.tsx > Monitoreo
21 const Monitoreo: React.FC = () => {
84 const dibujarGrafica = (datos: DataItem[]) => {
93
94   new Chart(ctx, {
95     type: 'bar',
96     data: {
97       labels: datos.map(d => new Date(d.dia).toLocaleDateString()), // Usar día para las etiquetas
98       datasets: [
99         {
100           label: 'Temperatura',
101           data: datos.map(d => d.temperatura), // Datos de temperatura
102           backgroundColor: 'rgba(255, 99, 132, 0.2)',
103           borderColor: 'rgba(255, 99, 132, 1)',
104           borderWidth: 1
105         },
106         {
107           label: 'Humedad',
108           data: datos.map(d => d.humedad), // Datos de humedad
109           backgroundColor: 'rgba(75, 192, 192, 0.2)',
110           borderColor: 'rgba(75, 192, 192, 1)',
111           borderWidth: 1
112         }
113       ]
114     },
115     options: {
116       scales: {
117         x: {
118           title: {
119             display: true,
120             text: 'Dia'
121           }
122         },
123         y: {
124           beginAtZero: true,
125           title: {
126             display: true,
127             text: 'Temperatura'
128           }
129         }
130       }
131     }
132   });
133 }
```





Ilustración 10 Parte 1 de componente de monitoreo

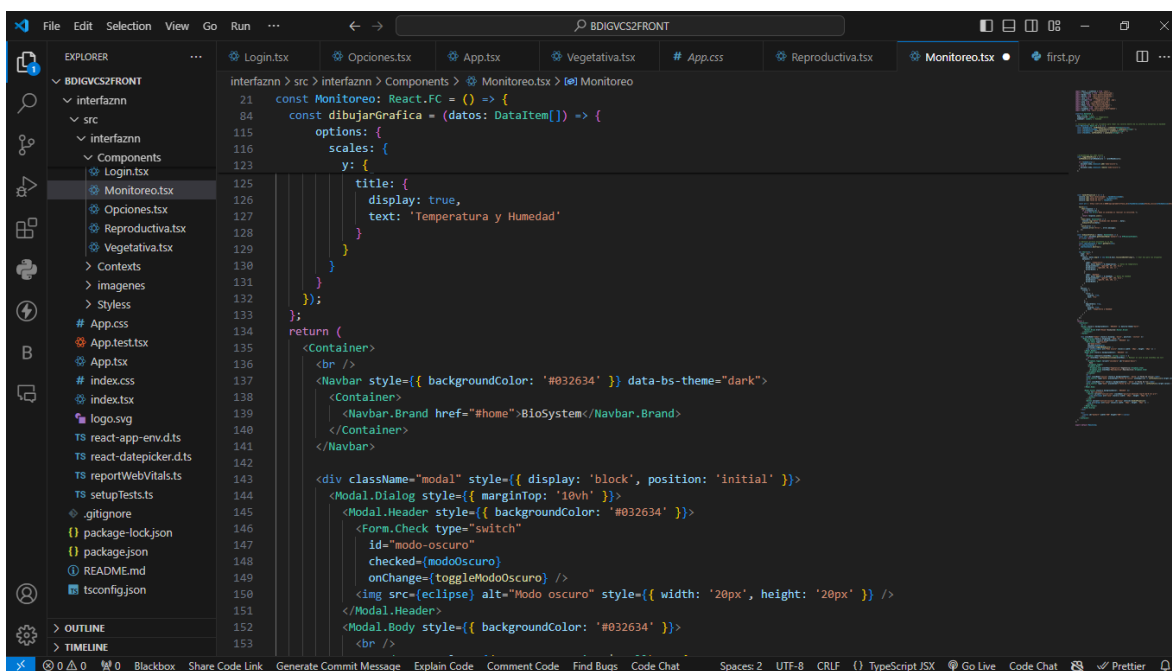
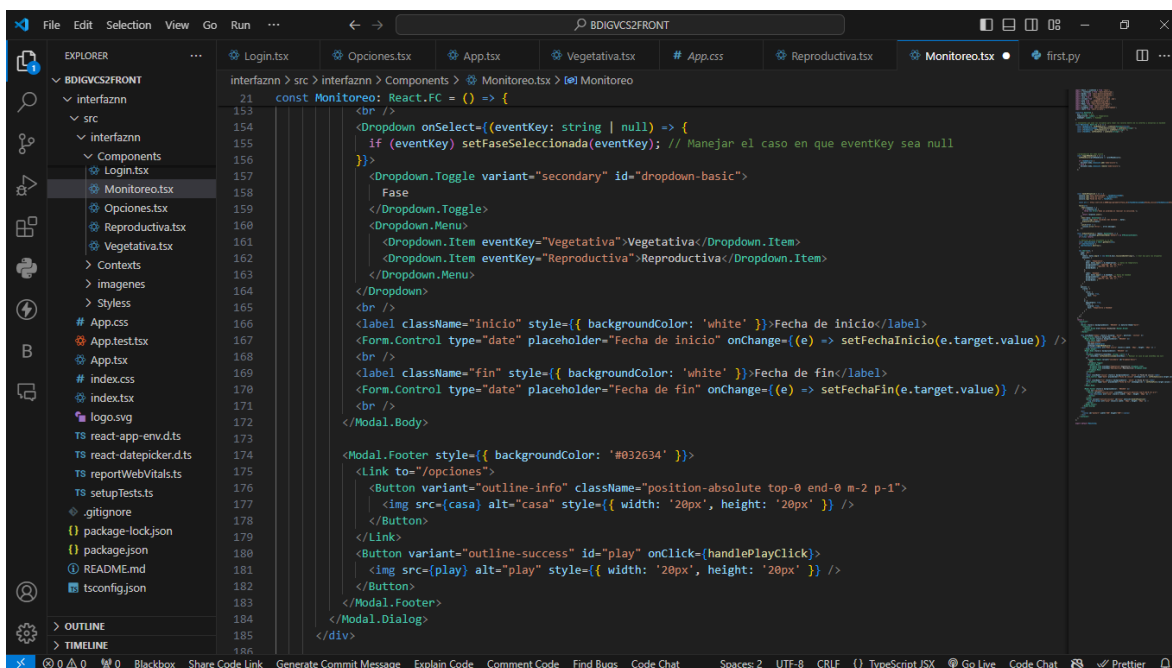


Ilustración 11 Parte 2 de componente de monitoreo





### *Integración del backend y el frontend*

Para la integración entre el frontend y el backend, se desarrollaron APIs RESTful con Laravel. Estas APIs permitieron la comunicación entre React JS y Laravel, gestionando las operaciones para obtener y enviar datos de control y monitoreo necesarias para la aplicación.

Se anexan las partes del código que se programaron y configuraron para poder lograr la integración.





Ilustración 12 programación de constantes para el funcionamiento de los controles

```
interfaznn > src > interfaznn > Components > Vegetativa.tsx > App
1 import React, { useState } from 'react';
2 import { Container } from 'react-bootstrap';
3 import { Navbar } from 'react-bootstrap/Navbar';
4 import Button from 'react-bootstrap/Button';
5 import casa from '../imagenes/casa.png';
6 import eclipse from '../imagenes/eclipse.png';
7 import temperatura from '../imagenes/temperatura.png';
8 import humedad from '../imagenes/humedad.png';
9 import play from '../imagenes/play.png';
10 import Form from 'react-bootstrap/Form';
11 import Modal from 'react-bootstrap/Modal';
12 import { Link } from 'react-router-dom';
13
14 function App() {
15   const [temperature, setTemperature] = useState(28);
16   const [humidity, setHumidity] = useState(70);
17
18   const increaseTemperature = () => {
19     setTemperature(prevTemperature => prevTemperature + 1);
20   };
21
22   const decreaseTemperature = () => {
23     setTemperature(prevTemperature => prevTemperature - 1);
24   };
25
26   const increaseHumidity = () => {
27     setHumidity(prevHumidity => prevHumidity + 1);
28   };
29
30   const decreaseHumidity = () => {
31     setHumidity(prevHumidity => prevHumidity - 1);
32   };
33
34 }
```





Ilustración 13 Programación para la conexión con el backend en la parte de controles

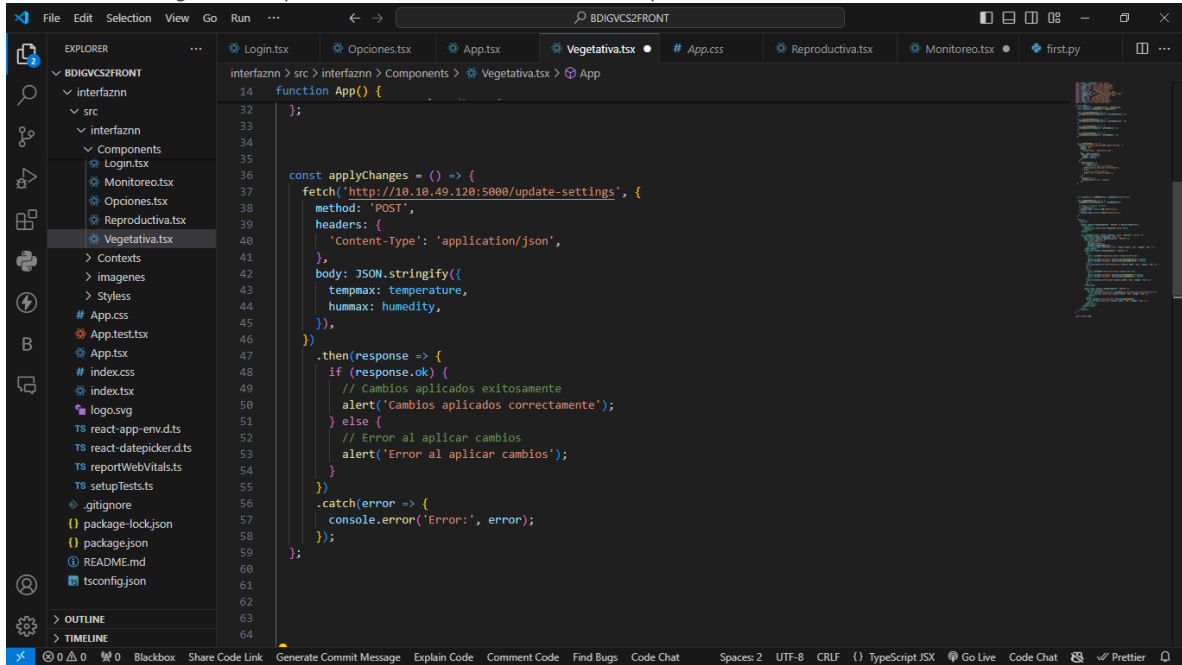


Ilustración 14 Programación de constantes para el funcionamiento del componente de monitoreo

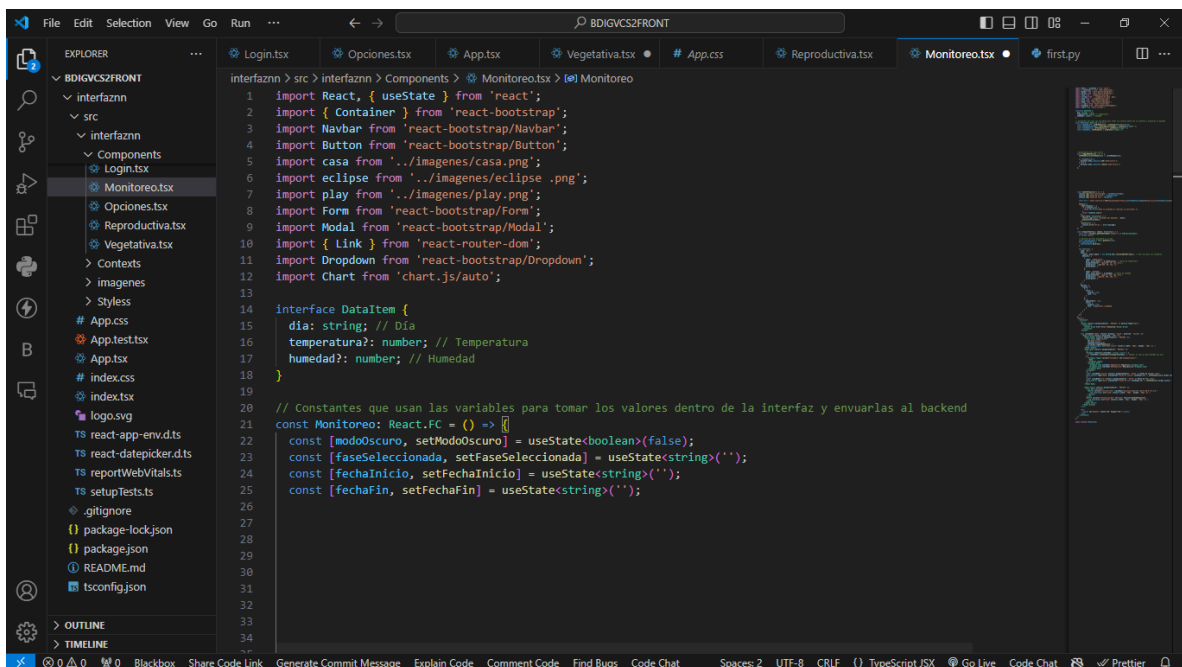




Ilustración 15 Programación de la conexión con el backend para la parte del monitoreo

```
interfaznzn > src > interfaznzn > Components > Monitoreo.tsx > Monitoreo
21 const Monitoreo: React.FC = () => {
60
61 const handlePlayClick = () => {
62   console.log('Fase seleccionada:', faseSeleccionada);
63   console.log('Fecha de inicio:', fechaInicio);
64   console.log('Fecha de fin:', fechaFin);
65
66   const url = `http://127.0.0.1:8000/api/parametro?fase_id=${faseSeleccionada}&fecha_inicio=${fechaInicio}&fecha_fin=${fechaFin}`;
67
68   fetch(url)
69     .then(response => {
70       if (!response.ok) {
71         throw new Error('Hubo un problema al realizar la solicitud.');
```

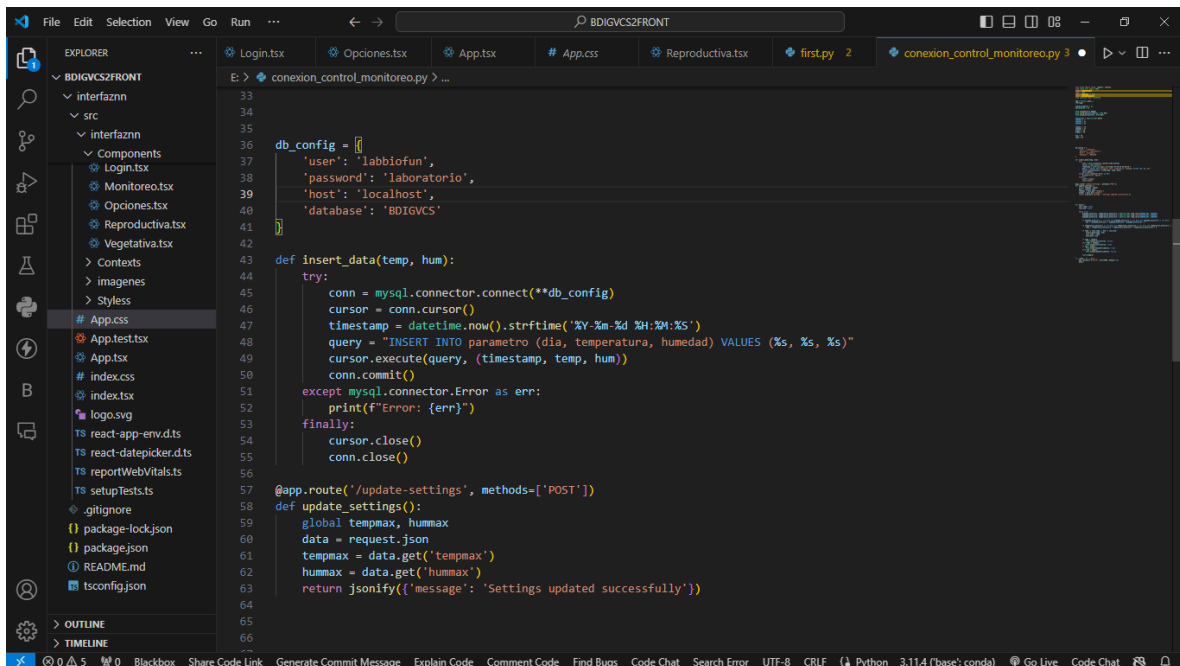




## Monitoreo y control

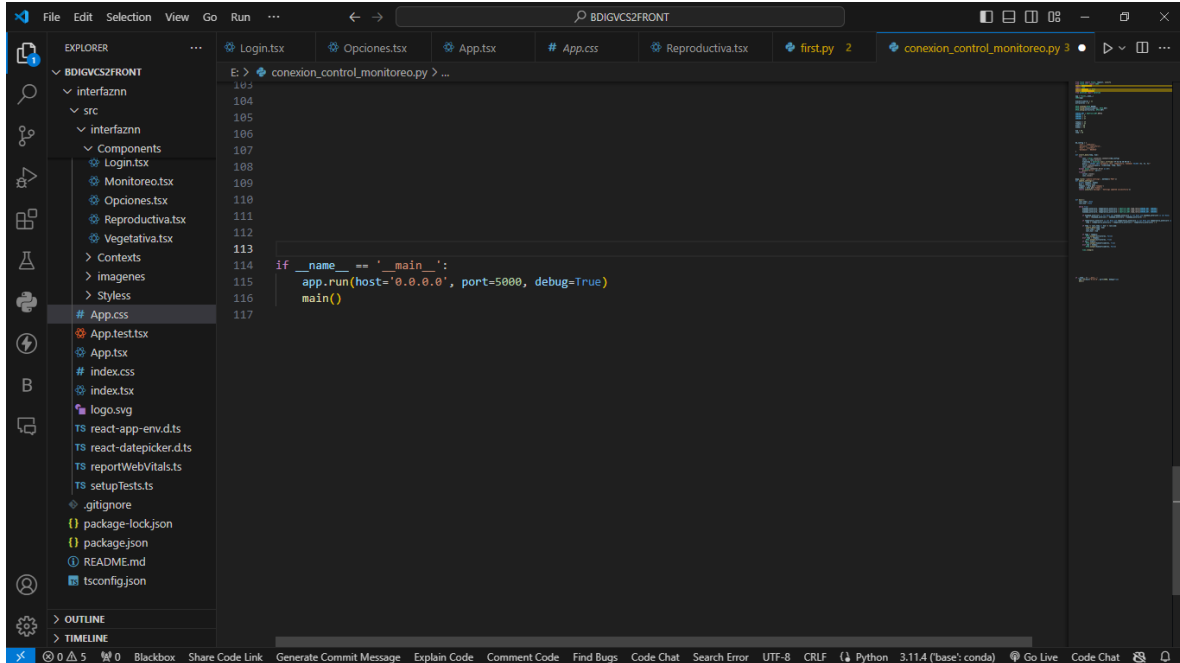
Se implementaron funcionalidades de monitoreo y control en tiempo real utilizando Python para manejar la comunicación con sensores y actuadores en la granja vertical. Estos datos se almacenaron en MySQL para una rápida consulta y procesamiento.

*Ilustración 16 Programación y configuración de la base de datos que envía los datos nuevos que se necesitaran para el monitoreo*



```
33
34
35
36 db_config = {
37     'user': 'labbiofun',
38     'password': 'laboratorio',
39     'host': 'localhost',
40     'database': 'BDIGVCS'
41 }
42
43 def insert_data(temp, hum):
44     try:
45         conn = mysql.connector.connect(**db_config)
46         cursor = conn.cursor()
47         timestamp = datetime.now().strftime('%Y-%m-%d %H:%M:%S')
48         query = "INSERT INTO parametro (dia, temperatura, humedad) VALUES (%s, %s, %s)"
49         cursor.execute(query, (timestamp, temp, hum))
50         conn.commit()
51     except mysql.connector.Error as err:
52         print(f"Error: {err}")
53     finally:
54         cursor.close()
55         conn.close()
56
57 @app.route('/update-settings', methods=['POST'])
58 def update_settings():
59     global tempmax, hummax
60     data = request.json
61     tempmax = data.get('tempmax')
62     hummax = data.get('hummax')
63     return jsonify({'message': 'Settings updated successfully'})
64
65
66
```



*Ilustración 17 Programación del puerto que escucha la solicitud de los controles*

### Despliegue en la Raspberry Pi

Finalmente, se procedió al despliegue del sistema completo en la Raspberry Pi. Apache sirvió como servidor web, mientras que los servicios de base de datos MySQL y MongoDB se configuraron para ejecutarse de manera eficiente en el hardware limitado de la Raspberry Pi. El proceso incluyó:

- Configuración del servidor Apache para servir la aplicación frontend construida con React JS.
- Despliegue de la aplicación Laravel en el mismo servidor.
- Configuración de Node JS para ejecutar servicios de monitoreo en segundo plano.

### **Resultados**

Finalmente, los resultados son satisfactorios, ya que el sistema en conjunto fue integrado en la raspberry con éxito y esta a su vez fue integrada a la granja vertical.

*Ilustración 18 Resultado de la interfaz*

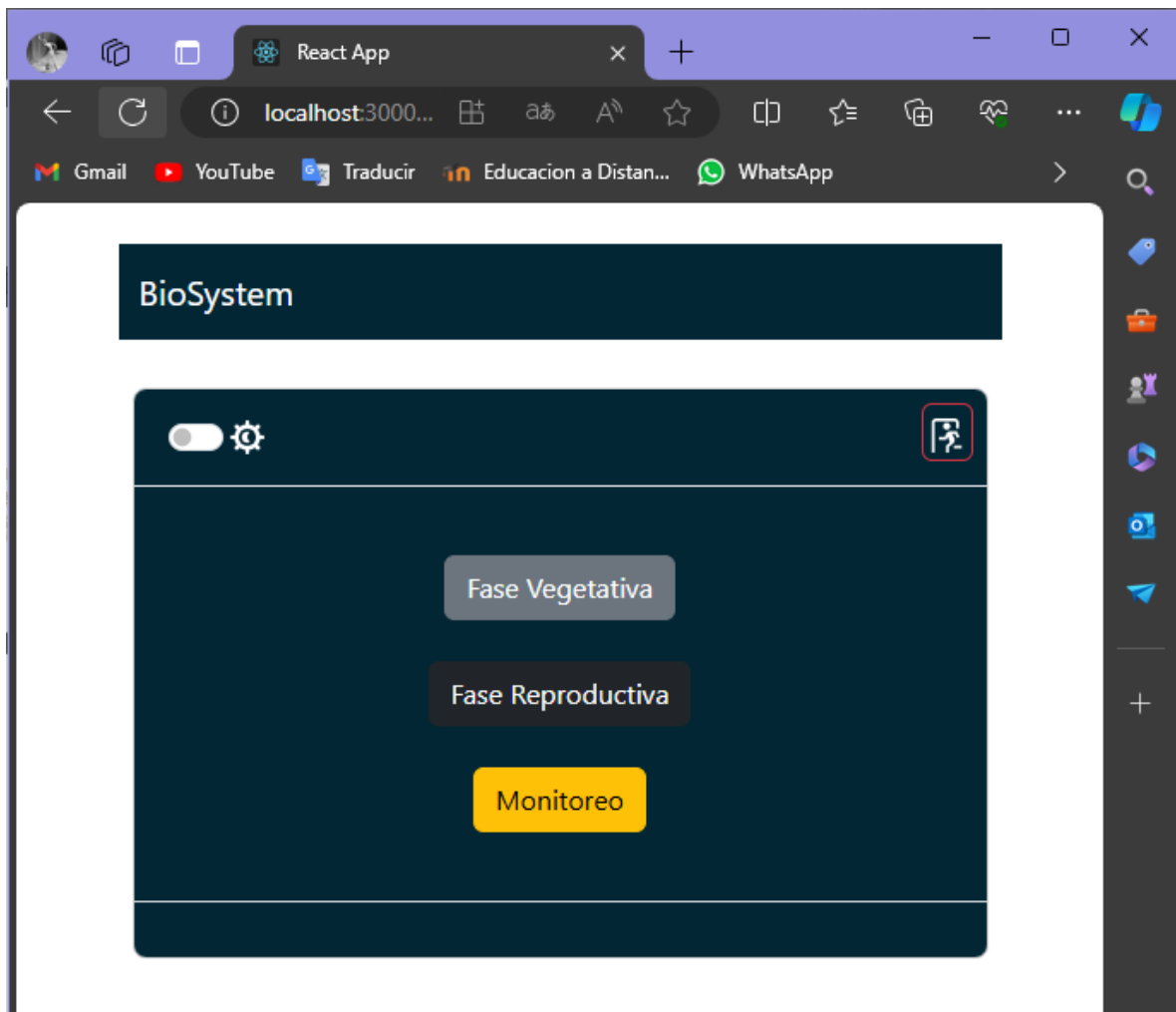




Ilustración 19 Resultado de la interfaz de control

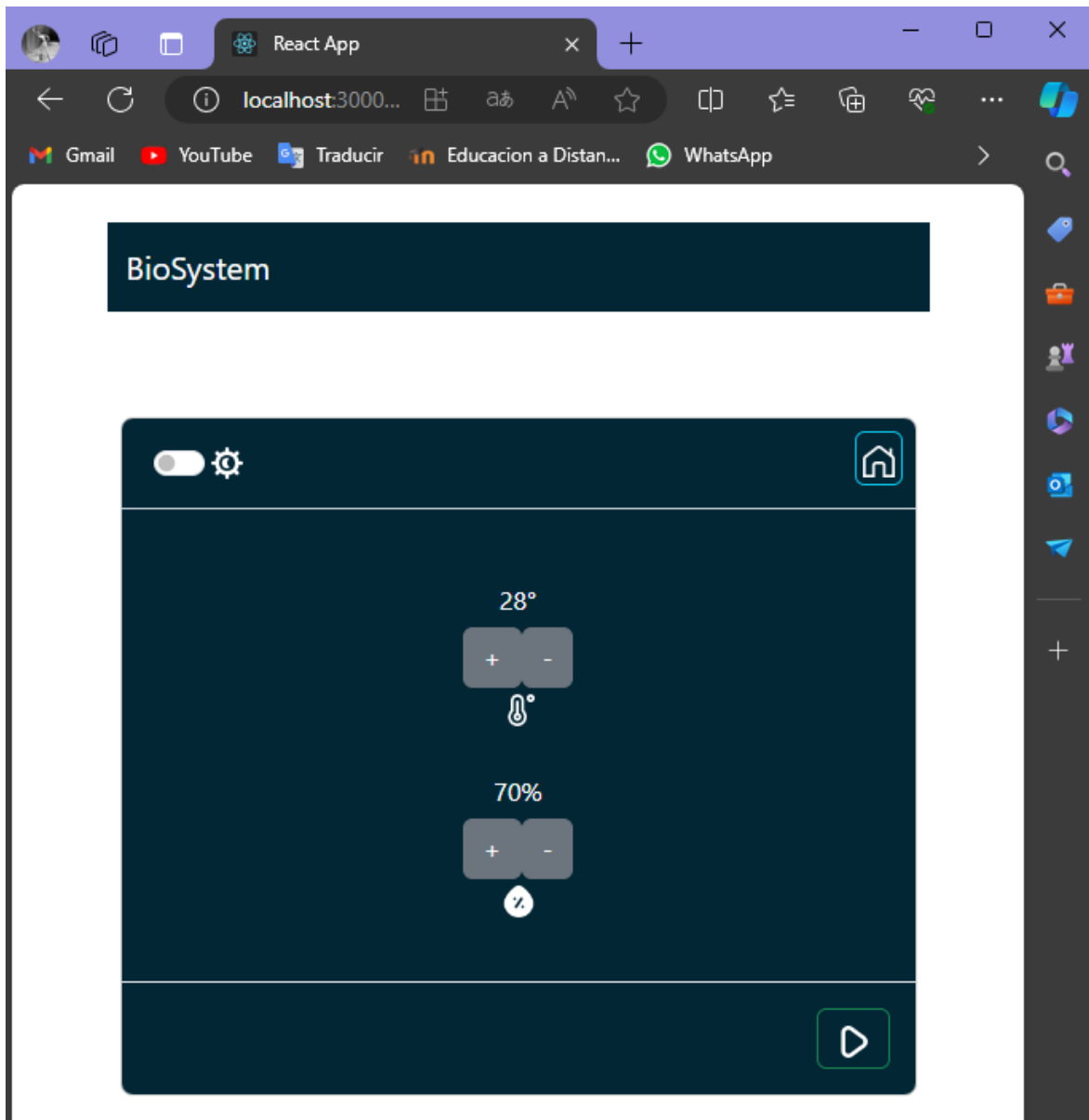




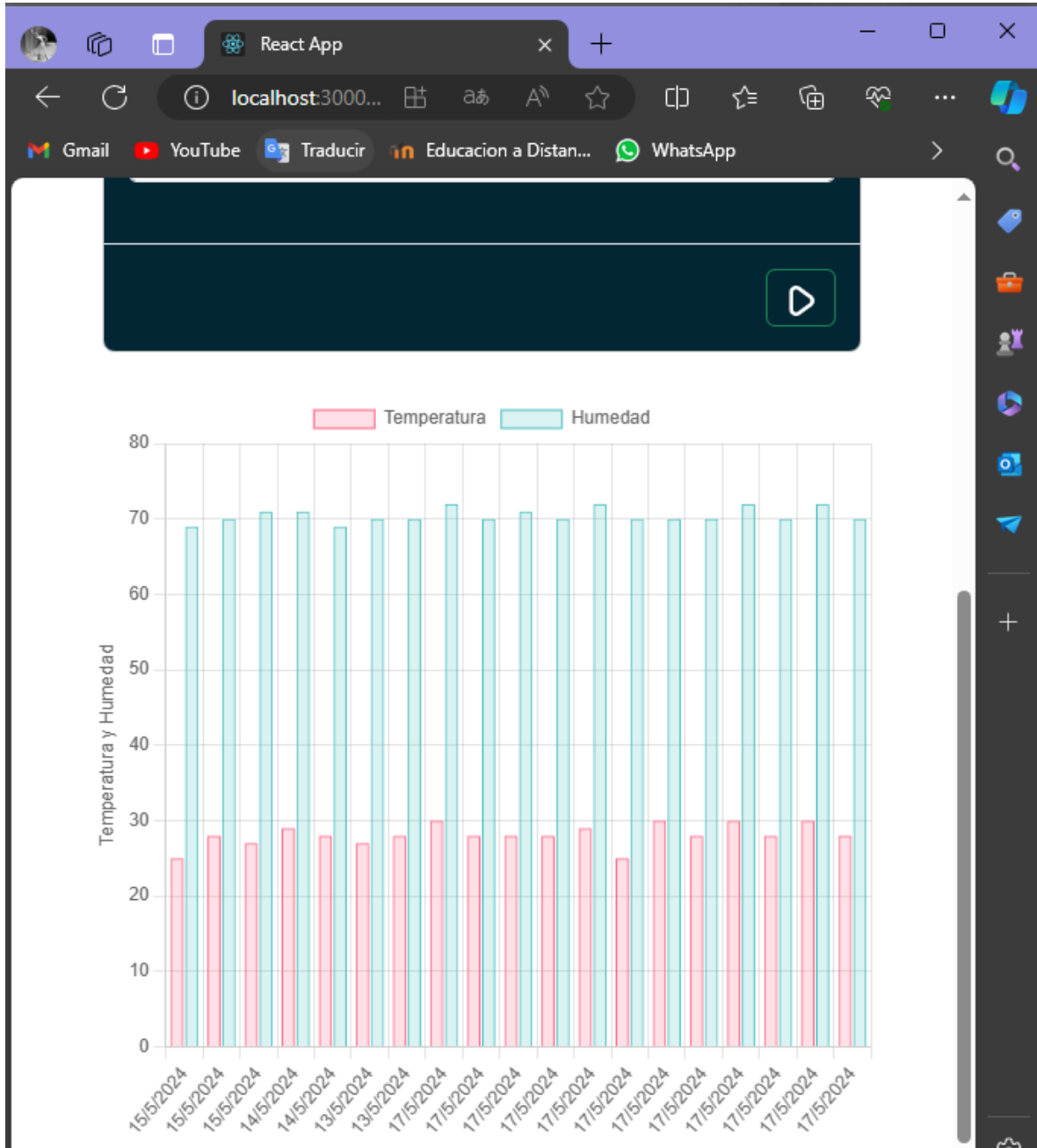
Ilustración 20 Resultado de la interfaz de monitoreo

The screenshot shows a web browser window with a single tab titled 'React App'. The address bar displays 'localhost:3000...'. Below the browser window, a dark-themed web application interface is visible. At the top, the word 'BioSystem' is displayed in white. Below this, there is a dark blue header bar containing a toggle switch and a gear icon on the left, and a home icon on the right. The main content area features a dropdown menu labeled 'Fase' with a downward arrow. Below the dropdown are two white input fields. The first field is labeled 'Fecha de inicio' and contains the date '12/05/2024'. The second field is labeled 'Fecha de fin' and contains the date '23/05/2024'. Both input fields have a calendar icon on the right. At the bottom right of the main content area, there is a green play button icon.





Ilustración 21 Resultado de la interfaz de monitoreo con grafica





### ***Conclusiones***

Dando seguimiento a la Hipótesis definida para este proyecto se puede concluir con la correcta y exitosa implementación de este sistema lo que representa una mejora significativa en la gestión y el monitoreo de las condiciones ambientales, para mejorar la calidad y el rendimiento del cultivo de setas y se espera que se pueda implementar este sistema en una mayor cantidad de proyectos que se enfoquen en este ámbito ya sea dentro del Instituto Tecnológico de Pabellón de Arteaga o en cualquier institución que lo requiera, además de que el sistema esta hecho para poder recibir actualizaciones y ser mejorado con el paso del tiempo, adaptándose a las necesidades del proyecto y el usuario.

### ***Experiencia profesional adquirida***

La habilidad de trabajar en quipo es fundamental para una excelente comunicación y un menor numero de errores que recaigan en el desarrollo del proyecto.

Se desarrollaron habilidades comunicativas entre los integrantes del equipo para poder manejar el orden de los factores y las tareas correspondientes.

Se aplicaron habilidades de programación para el manejo de errores emergentes de los componentes en conexión con el backend y la base de datos.

Además, se obtuvieron conocimientos sobre la correcta instalación y verificación de versiones de sistemas y aplicaciones entorno a una Raspberry Pi.





## Referencias

(s.f.). Obtenido de php: <https://www.php.net/manual/es/faq.general.php>

(s.f.). Obtenido de TusClases: <https://www.tusclases.co/blog/conceptos-basicos-empezar-entender-javascript>

(s.f.). Obtenido de TypeScript: <https://www.typescriptlang.org/>

(s.f.). Obtenido de Laravel: <https://laravel.com/>

(s.f.). Obtenido de NODEjs: <https://nodejs.org/en/learn/getting-started/introduction-to-nodejs>

(s.f.). Obtenido de Composer : <https://getcomposer.org/download/>

(s.f.). Obtenido de MDN web docs: <https://developer.mozilla.org/es/docs/Web/HTTP/CORS>

(30 de 04 de 2021). Obtenido de Codecademy:  
<https://www.codecademy.com/resources/blog/what-is-sql/>

(25 de 05 de 2023). Obtenido de ITD: [https://informatecdigital.com/bases-de-datos/que-es-mysql-y-como-  
funciona/#:~:text=%C2%BFQu%C3%A9%20es%20MySQL%20y%20c%C3%B3mo%20funciona%3F%201%20Definici%C3%B3n,8%20Funciones%20populares%20de%20MySQL%20...%20M%C3%A1s%20elementos](https://informatecdigital.com/bases-de-datos/que-es-mysql-y-como-funciona/#:~:text=%C2%BFQu%C3%A9%20es%20MySQL%20y%20c%C3%B3mo%20funciona%3F%201%20Definici%C3%B3n,8%20Funciones%20populares%20de%20MySQL%20...%20M%C3%A1s%20elementos)

Etcé, E. (19 de 11 de 2023). Obtenido de Concepto: <https://concepto.de/base-de-datos/>

Marotel, A. (15 de 07 de 2022). Obtenido de TWAINO: <https://www.twaino.com/es/blog/crear-un-sitio-web/servidor-apache-una-guia-completa-para-principiantes/>

MDN web docs. (s.f.). Obtenido de  
[https://developer.mozilla.org/es/docs/Learn/Getting\\_started\\_with\\_the\\_web/JavaScript\\_basics](https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/JavaScript_basics)

Patricia, R. (s.f.). Obtenido de conectemos: <https://conectemos.com/que-es-php/>







**EDUCACIÓN**  
SECRETARÍA DE EDUCACIÓN PÚBLICA



TECNOLÓGICO  
NACIONAL DE MÉXICO®

Instituto Tecnológico de Pabellón de Arteaga



Carretera a la Estación de Rincón Km. 1 C.P. 20670 Pabellón de Arteaga, Aguascalientes  
Tel. 465 958-2482 Ext. 105 e-mail: [dep\\_parteaga@tecnm.mx](mailto:dep_parteaga@tecnm.mx) [tecnm.mx](http://tecnm.mx) | [pabellon.tecnm.mx](http://pabellon.tecnm.mx)



**2024**  
FELIPE CARRILLO  
PUERTO  
MANIFIESTO DEL PUEBLABUENO,  
DESARROLLO Y DEFENSA  
DEL MAYOR