Olimpiada de Programación – 2024

- CUE de la escuela:060593300
- Nombre completo de la escuela: Escuela Educativa Secundaria Técnica n2
- Localidad: Mar Del Plata
- Provincia: Buenos Aires
- Profesor responsable: Gabriel Ángel Pimentel
- Apellido y nombre: Gabriel Ángel Pimentel
- Título de grado o profesorado: Analista de sistemas
- Cargo docente: Profesor
- Correo electrónico: pimentel.@icloud.com
- Teléfono celular (con prefijo jurisdicción):2235510343

- Apellido y Nombre: Lautaro Aguirre
- Especialidad que cursa: Programación
- Ciclo del curso: 2024
- Año de cursada: 7mo2da

- Apellido y Nombre: Damián Martínez
- Especialidad que cursa: Programación
- Ciclo del curso: 2024
- Año de cursada: 7mo2da

- Apellido y Nombre: Sebastian Gonzalez
- Especialidad que cursa: Programación
- Ciclo del curso: 2024
- Año de cursada: 7mo2da

Contenido

Planilla de distribución de tareas:

ID	Descripción de la tarea	Responsable	Fecha de inicio	Fecha de entrega	Criterios de asignación
T01	Gestión y mantenimiento de planilla de tareas	Sebastian Gonzalez	12/8	16/8	Responsabilidad administrativa
T02	Diagrama de contexto	Damián Martínez	12/8	12/8	Experiencia de diseño
T03	Caso de uso	Damián Martínez	12/8	12/8	Experiencia de diseño
T04	Diagrama de entidad relación	Lautaro Aguirre	12/8	12/8	Conocimiento en diseño de bases de datos
T05	Planificación de los tiempos de desarrollo	Sebastian Gonzalez	12/8	12/8	Responsabilidad administrativa
T06	Selección e instalación del servidor de archivos y aplicaciones	Sebastian Gonzalez	12/8	13/8	Experiencia en infraestructura de servidores
T07	Selección e instalación del motor de base de datos	Sebastian Gonzalez	13/8	14/8	Experiencia en bases de datos
T08	Diseño de la página (incluye wireframes y diseño visual)	Damián Martínez	13/8	13/8	Experiencia en diseño y desarrollo frontend
T09	Desarrollo frontend completo (incluye diseño adaptable a móviles)	Damián Martínez	13/8	16/8	Experiencia en diseño y desarrollo frontend
T10	Desarrollo e implementación del módulo de inicio de sesión	Lautaro Aguirre	14/8	14/8	Experiencia en bases de datos
T11	Desarrollo del módulo de pago	Lautaro Aguirre	14/8	14/8	Experiencia en bases de datos
T12	Implementación del módulo de pago	Sebastian Gonzalez	15/8	15/8	Experiencia en bases de datos

T13	Documentación	Todo el equipo	17/8	17/8	

Diagrama Entidad Relacion:

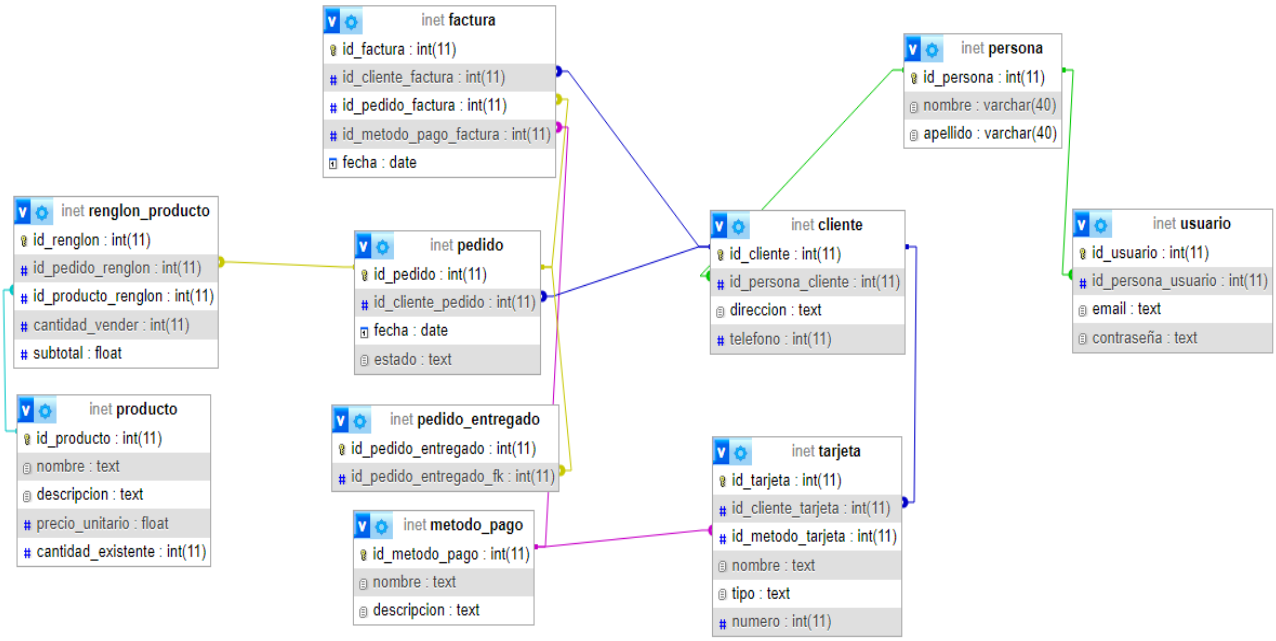
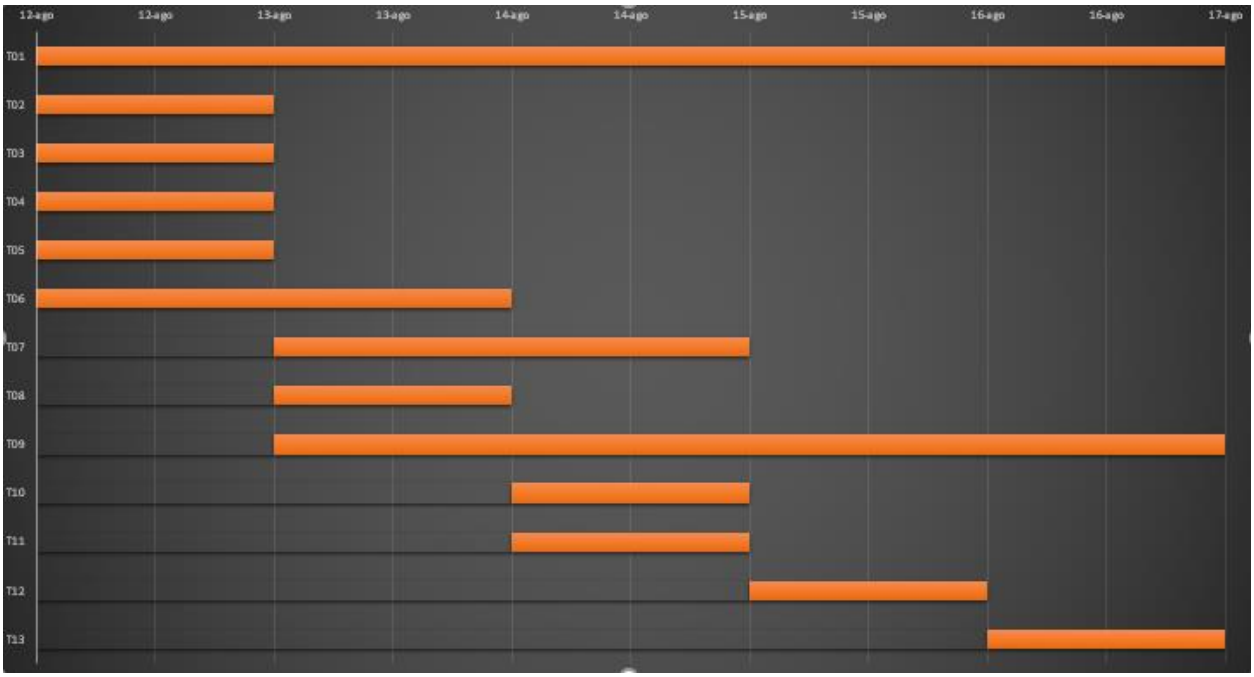


Diagrama de Gantt:



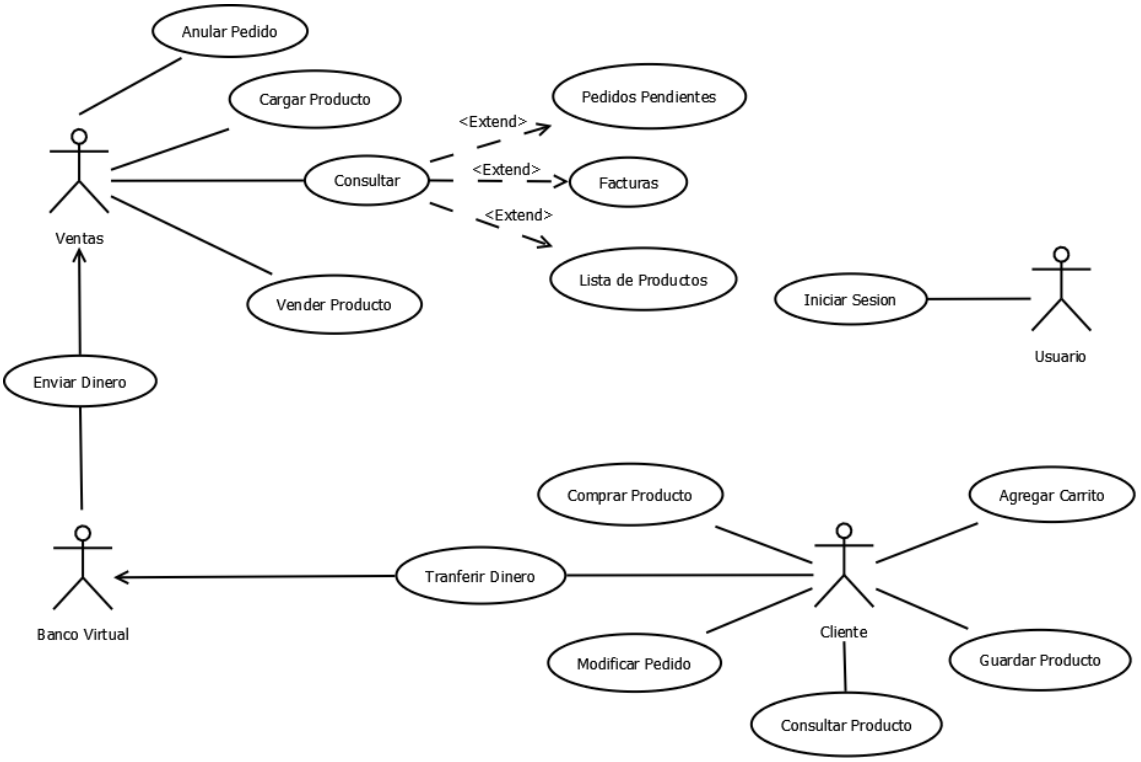
Resumen del Relevamiento

En la primera etapa del proyecto, nos reunimos con el Jefe de Ventas y los vendedores más experimentados de la empresa para entender lo que necesitaban. Después de varias charlas, decidimos que la página del carrito de compras se enfocará en la venta de productos deportivos.

Queremos que los usuarios puedan encontrar y comprar fácilmente todo lo que necesitan para deportes al aire libre, como ciclismo, senderismo y escalada. Además, pensamos en agregar un sistema de seguridad sencillo pero efectivo para que los usuarios puedan crear sus cuentas y gestionar sus compras sin complicaciones.

También escuchamos que es importante que los empleados de la empresa puedan manejar los productos, ver el estado de los pedidos y gestionar las entregas desde la aplicación. Así, no solo ayudamos a los clientes, sino también al equipo de ventas a mantener todo bajo control.

Casos de uso:



Nombre del Caso de Uso	Iniciar Sesión
Actores	Usuario
Descripción	El usuario inicia sesión en el sistema
Precondición	El usuario debe haber creado una cuenta anteriormente
Flujo Principal	El usuario ingresa su correo y contraseña en el sistema
Postcondición	El usuario ahora posee acceso al sistema
Excepciones	1. El usuario no posee una cuenta 2. Los datos ingresados no son correctos.

Nombre del Caso de Uso	Comprar Producto
Actores	Cliente - Ventas
Descripción	El cliente compra un producto en el sistema
Precondición	1. El cliente debe tener acceso al sistema. 2. Debe tener una billetera virtual disponible.
Flujo Principal	1. Selecciona un producto. 2. Selecciona la Opción “Comprar”. 3. Elige una Billetera Virtual/Tarjeta. 4. Finaliza la operación.
Postcondición	Se imprime una factura de compra y una orden de envío.
Excepciones	1. El producto no está disponible. 2. La información de la tarjeta es errónea. 3. No cantidad suficiente para comprar el producto.

Nombre del Caso de Uso	Agregar Carrito
Actores	Cliente
Descripción	El Cliente agrega un producto a un “carrito de compras” virtual
Precondición	El producto debe estar disponible en la página.
Flujo Principal	1. El cliente selecciona un producto. 2. Selecciona la opción “agregar al carrito”.
Postcondición	El producto se agrega al carrito para luego comprarlo o agregar otros.
Excepciones	El cliente no esté interesado.

Nombre del Caso de Uso	Consultar Producto
Actores	Cliente
Descripción	El cliente consulta la información del producto.
Precondición	El producto debe estar disponible.
Flujo Principal	1. Selecciona el producto. 2. Realiza la consulta.
Postcondición	Se muestra la información del producto en pantalla.
Excepciones	

Nombre del Caso de Uso	Guardar Producto
Actores	Cliente
Descripción	El Cliente guarda el producto
Precondición	El producto debe estar disponible en la pagina
Flujo Principal	1. Selecciona el producto. 2. Selecciona la opción “Guardar Producto”.
Postcondición	El producto se guarda en una lista para el usuario.
Excepciones	El producto no esté disponible.

Nombre del Caso de Uso	Modificar Pedido
Actores	Cliente
Descripción	El cliente modifica la información del pedido.
Precondición	El cliente debe comprar el producto.
Flujo Principal	1. El cliente compra el producto. 2. Se le envían información sobre el pedido. 3. Pide modificar la información del mismo. 4. Modifica la información. 5. Se la envía al Vendedor.
Postcondición	La información del pedido es modificada, afectando el proceso del envío.
Excepciones	1. El pedido sea anulado. 2. El pedido no puede ser modificado al estar en la zona.

Nombre del Caso de Uso	Transferir Dinero
Actores	Cliente – Banco Virtual
Descripción	El cliente manda una orden de transferencia para el eCommerce al Banco Virtual.
Precondición	1. El cliente debe seleccionar una forma de pago. 2. El cliente debe ejecutar la compra del producto.
Flujo Principal	1. El cliente selecciona la forma de pago. 2. Rellena la información del Banco Virtual seleccionado. 3. Efectúa la compra.
Postcondición	Confirma la compra del producto
Excepciones	1. La información dada por el cliente es errónea. 2. No hay cantidad suficiente

Nombre del Caso de Uso	Enviar Dinero
Actores	Banco Virtual – Ventas
Descripción	El Banco realiza la transferencia al vendedor.
Precondición	El cliente autorizar la transferencia.
Flujo Principal	1. El cliente transfiere el pago. 2. El banco autoriza la transferencia. 3. El banco mueve la paga al vendedor.
Postcondición	El vendedor recibe la transferencia.
Excepciones	El Banco no lo autoriza por diversas razones.

Nombre del Caso de Uso	Anular Pedido
Actores	Ventas
Descripción	El vendedor anula el pedido del cliente.
Precondición	Debe haber un pedido por parte del cliente.
Flujo Principal	1. Pasa una irregularidad. 2. El vendedor anula el pedido.
Postcondición	Se anula él envió y se da un reembolso.
Excepciones	No hay irregularidades durante el tiempo del pedido.

Nombre del Caso de Uso	Cargar Producto
Actores	Ventas
Descripción	El Vendedor carga un producto al sistema.
Precondición	El vendedor debe tener acceso suficiente para realizar la acción.
Flujo Principal	1. El vendedor entra a la Base de Datos. 2. Rellena con información del producto. 3. Acepta los cambios.
Postcondición	Se agrega y se lanza un producto nuevo al sistema.
Excepciones	1. No tenga información suficiente sobre el producto. 2. El usuario no tenga acceso suficiente.

Nombre del Caso de Uso	Consultar
Actores	Ventas
Descripción	El vendedor consulta la información en el sistema.
Precondición	El vendedor debe iniciar Sesión en el sistema.
Flujo Principal	1. El vendedor consulta: Pedidos, Facturas, Productos. 2. El sistema muestra los resultados.
Postcondición	La información solicitada en la consulta es mostrada en pantalla.
Excepciones	

Nombre del Caso de Uso	Vender Producto
Actores	Ventas – Cliente
Descripción	El Vendedor vende el producto al cliente
Precondición	1. Se necesita que el cliente acceda a comprar el producto. 2. Se necesita que el producto esté disponible.
Flujo Principal	1. El cliente compra el producto. 2. El sistema emite una factura. 3. Se emite el pedido.
Postcondición	Se efectúa la transferencia y procede a realizar el envío.
Excepciones	1. El pedido en anulado. 2. El cliente cancela el pedido.

Codigo Fuente:

App.js

```
import express from 'express';

import bodyParser from 'body-parser';
import session from 'express-session';
import path from 'path';
import { fileURLToPath } from 'url';
import { MercadoPagoConfig, Preference } from 'mercadopago';
import conexion from './conexion.js'; // Importar conexión de la base de datos

const __filename = fileURLToPath(import.meta.url);
const __dirname = path.dirname(__filename);

const app = express();

// Configurar Mercado Pago
const client = new MercadoPagoConfig({ accessToken: 'APP_USR-1273616427086786-
081416-6c677cf05d5375601b97d922415fb61c-1947370360' });

// Configurar body-parser para manejar datos del formulario
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

// Configuración de sesiones
app.use(session({
    secret: 'mi-secreto', // Cambia esto por un valor más seguro
    resave: false,
    saveUninitialized: true,
    cookie: { secure: false } // Cambia a true si usas HTTPS
}));

// Servir archivos estáticos desde la carpeta 'public'
app.use(express.static(path.join(__dirname, '..')));

// Página de inicio con opciones para registrarse o iniciar sesión
app.get('/', (req, res) => {
    res.sendFile(path.join(__dirname, '..', 'index.html'));
});

// Renderizar el formulario de registro
app.get('/registro', (req, res) => {
    res.sendFile(path.join(__dirname, '..', 'secciones', 'formulario',
'registro.html'));
```

```
});

// Manejar el registro
app.post('/registro', (req, res) => {
    const { nombre, apellido, email, contraseña } = req.body;

    conexion.beginTransaction(err => {
        if (err) throw err;

        conexion.query(
            'SELECT id_persona_usuario FROM usuario WHERE email = ?',
            [email],
            (err, results) => {
                if (err) {
                    return conexion.rollback(() => {
                        throw err;
                    });
                }

                if (results.length > 0) {
                    return res.send(`
                        <p>Este email ya está registrado.</p>
                        <a href="/login">Inicia sesión aquí</a>
                    `);
                }

                conexion.query(
                    'INSERT INTO persona (nombre, apellido) VALUES (?, ?)',
                    [nombre, apellido],
                    (err, result) => {
                        if (err) {
                            return conexion.rollback(() => {
                                throw err;
                            });
                        }

                        const personaId = result.insertId;

                        conexion.query(
                            'INSERT INTO usuario (id_persona_usuario, email,
contraseña) VALUES (?, ?, ?)',
                            [personaId, email, contraseña],
                            (err, result) => {
                                if (err) {
                                    return conexion.rollback(() => {
```

```
                                    throw err;
                                });
                            }

                            conexion.commit(err => {
                                if (err) {
                                    return conexion.rollback(() => {
                                        throw err;
                                    });
                                }

                                res.redirect(`/login`);
                            });
                        }
                    );
                }
            );
        }
    );
});

// Renderizar el formulario de inicio de sesión
app.get('/login', (req, res) => {
    res.sendFile(path.join(__dirname, '..', 'secciones', 'formulario',
'login.html'));
});

// Manejar el inicio de sesión y verificar si el usuario es cliente
app.post('/usuario', (req, res) => {
    const { email, contraseña } = req.body;

    conexion.query(
        'SELECT p.id_persona, u.email, u.contraseña, c.id_cliente FROM usuario u
JOIN persona p ON u.id_persona_usuario = p.id_persona LEFT JOIN cliente c ON
c.id_cliente = p.id_persona WHERE u.email = ?',
        [email],
        (err, results) => {
            if (err) throw err;

            if (results.length === 0) {
                return res.send(`
                    <p>Usuario o contraseña incorrectos.</p>
                    <a href="/login">Inténtalo de nuevo</a>
                `);
```

```
            }

            const user = results[0];
            if (user.contraseña !== contraseña) {
                return res.send(`
                    <p>Contraseña incorrecta.</p>
                    <a href="/login">Inténtalo de nuevo</a>
                `);
            }

            req.session.clienteId = user.id_cliente; // Guardar el ID del cliente
en la sesión
            res.sendFile(path.join(__dirname, '..', 'index.html'));
        }
    );
});

app.get('/usuario', (req, res) => {
    const id = req.session.clienteId;
    connection.query(
        'SELECT id_persona FROM usuario WHERE id_cliente= ?', [id],
        (err, results) => {
            if (err) throw err;

            if (results.length === 0) {
                return res.send(`
                    <p>Email o contraseña incorrectos.</p>
                    <a href="/login">Intenta de nuevo</a>
                `);
            }
            id_persona = results[0];
            connection.query(
                `SELECT * FROM usuario WHERE id= ?`, [id_persona], (err, results)
=> {
                    if (err) throw err;
                    const persona = results[0].persona;

                    req.session.user = {
                        email: persona.email,
                        id_usuario: persona.id_usuario
                    };
                }
            );
```

```
            res.sendFile(path.join(__dirname, '..', 'secciones', 'Usuario',
'Usuario.html'));
        }
    );
});

// Renderizar la página de productos
app.get('/productos', (req, res) => {
    conexion.query('SELECT * FROM producto', (err, productos) => {
        if (err) throw err;
        res.json(productos);
    });
});

app.get('/productos-pagina', (req, res) => {
    res.sendFile(path.join(__dirname, '..', 'secciones', 'producto',
'producto.html'));
});

// Manejar la creación de un nuevo pedido
app.post('/iniciar-pedido', (req, res) => {
    const clienteId = req.session.clienteId;

    if (!clienteId) {
        return res.status(401).send('No estás autenticado');
    }

    conexion.query(
        'INSERT INTO pedido (id_cliente_pedido, fecha, estado) VALUES (?, NOW(),
"En proceso")',
        [clienteId],
        (err, result) => {
            if (err) throw err;
            req.session.id_pedido_actual = result.insertId; // Guardar el ID del
pedido en la sesión
            res.json({ id_pedido: req.session.id_pedido_actual });
        }
    );
});

// Manejar la inserción de productos en el carrito (renglón de producto)
app.post('/agregar-al-carrito', (req, res) => {
    const { id_producto, cantidad } = req.body;
    const id_pedido = req.session.id_pedido_actual;
```

```
    if (!id_pedido) {
        return res.status(400).send('No hay un pedido en curso');
    }

    conexion.query(
        'INSERT INTO renglon_producto (id_pedido_renglon, id_producto_renglon,
cantidad_vender, subtotal) VALUES (?, ?, ?, (SELECT precio_unitario FROM producto
WHERE id_producto = ?) * ?)',
        [id_pedido, id_producto, cantidad, id_producto, cantidad],
        (err, result) => {
            if (err) throw err;
            res.json({ success: true });
        }
    );
});

// Manejar la finalización de la compra
app.post('/finalizar-compra', (req, res) => {
    const id_pedido_actual = req.session.id_pedido_actual;

    if (id_pedido_actual) {
        conexion.query(
            'UPDATE pedido SET estado = "Finalizado" WHERE id_pedido = ?',
            [id_pedido_actual],
            (err, result) => {
                if (err) throw err;
                req.session.id_pedido_actual = null; // Resetear el ID del pedido
actual
                res.json({ success: true });
            }
        );
    } else {
        res.status(400).json({ error: 'No hay pedido en curso' });
    }
});

// Crear una preferencia de pago con Mercado Pago
app.post("/create_preference", async (req ,res)=> {
    try{
        const body = {
            items: [
                {
                title: req.body.title,
                quantity: Number(req.body.quantity),
                unit_price: Number(req.body.price),
```

```
                currency_id: "ARS",
                },
            ],
            back_urls: {
                success:"https://www.youtube.com/@onthecode",
                failure:"https://www.youtube.com/@onthecode",
                pending:"https://www.youtube.com/@onthecode ",
            },
            auto_return: "approved",
        };

        const preference = new Preference(client);
        const result = await preference.create({body});
        res.json({
            id: result.id,
        });
    }catch (error) {
        console.log(error)
        res.status(500).json({
            error: "Error al crear la preferencia"
        })
    }
});

// Escuchar en el puerto 3000
app.listen(3000, () => {
    console.log('Servidor iniciado en http://localhost:3000');
});
```

Checkout.js

```
// Configura Mercado Pago

const mp = new MercadoPago('APP_USR-405752fa-83cd-43b1-bd30-352314bc1954', {
    locale: "es-AR",
});

// Maneja el evento de clic en el botón de finalizar compra
document.getElementById("checkout-btn").addEventListener("click", async () => {
    try {
        const orderData = {
            title: document.querySelector(".name")?.innerText || 'Producto
Genérico',
            quantity: 1,
            price: 100,
```

```
        };

        // Solicita la creación de una preferencia de pago en el servidor
        const response = await fetch("/create_preference", {
            method: "POST",
            headers: {
                "Content-Type": "application/json",
            },
            body: JSON.stringify(orderData),
        });

        const preference = await response.json();
        createCheckoutButton(preference.id);
    } catch (error) {
        alert("Error al procesar el pago.");
        console.error(error);
    }
});

// Crea el botón de pago con Mercado Pago
const createCheckoutButton = (preferenceId) => {
    const bricksBuilder = mp.bricks();

    const renderComponent = async () => {
        if (window.checkoutButton) {
            try {
                window.checkoutButton.unmount();
            } catch (error) {
                console.error("Error al desmontar el botón:", error);
            }
        }

        try {
            window.checkoutButton = await bricksBuilder.create("wallet",
"wallet_container", {
                initialization: {
                    preferenceId: preferenceId,
                },
                customization: {
                    texts: {
                        valueProp: 'smart_option',
                    },
                },
            });
        } catch (error) {
```

```
            console.error("Error al crear el botón:", error);
        }
    };

    renderComponent();
};
```

Conexión.js

```
import mysql from 'mysql2';

// Crear la conexión a la base de datos
const conexion = mysql.createConnection({
    host: 'localhost',
    database: 'inet',
    user: 'root',
    password: ''
});

// Conectar a la base de datos
conexion.connect(err => {
    if (err) {
        console.error('Error al conectar a la base de datos:', err);
        process.exit(1);
    }
    console.log('Conexión a la base de datos establecida con éxito');
});

// Exportar la conexión
export default conexion;
```

perfil.js

```
document.addEventListener('DOMContentLoaded', () => {

    fetch('/usuario')
        .then(response => response.json())
        .then(data => {
            const enlace = document.getElementById('inicio-sesion');

            // Modifica el contenido del enlace
            enlace.textContent = 'Mi Perfil';

            // Modifica el atributo href del enlace
            enlace.href = '/secciones/usuario/usuario.html';
```

```
        })
        .catch(error => console.error('Error:', error));
});
```

sesion.js

```
const conexion = require('../Js/Conexion');

const usuario = require('../JS/app');

if(usuario==true){
    console.log('sesion exitosa');
}
```

Slides.js

```
// Función para obtener los productos y construir los slides

async function cargarSlides() {
try {
    const response = await fetch('/productos');
    const productos = await response.json();
    const container = document.getElementById('slides-container');
    // Construye el HTML de los slides
    container.innerHTML = productos.map(producto => `
        <div class="swiper-slide">
            <h2>${producto.nombre}</h2>
            <p>${producto.descripcion}</p>
            <p>Precio: $${producto.precio_unitario}</p>
            <p>Cantidad: ${producto.cantidad_existente}</p>
        </div>
    `).join('');
    new Swiper('.swiper', {
        centeredSlides: true,
        slidesPerView: 'auto',
        effect: 'coverflow',
        pagination: {
            el:".swiper-pagination",
            clickable: true,
        },
        navigation: {
            nextEl: ".swiper-button-next",
            prevEl: ".swiper-button-prev",
        },
    });
    } catch (error) {
        console.error('Error al cargar los slides:', error);
```


2

```
    }
}

// Cargar los slides cuando se cargue la página
window.onload = cargarSlides;
```

ajuste_menu.js

```
window.addEventListener('resize', function(){

    var menuHeight = this.document.querySelector('header').offsetHeight;
    this.document.querySelector('main').style.paddingTop = menuHeight + 'px';
})

window.dispatchEvent(new Event('resize'));
```

index.html

```
<!DOCTYPE html>

<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>eCommerce</title>
    <link rel="stylesheet" href="CSS/style.css">
</head>

<body>
    <header>
        <nav>
            <input type="checkbox" name="" id="check">
            <label for="check" class="checkbtn">
                <img class="fas fa-bars" src="Imagenes/Iconos/menu.png"
width="30">
            </label>
            <ul class="contenidos">
                <li><a class="active" href="#">Inicio</a></li>
                <li><a href="http://localhost:3000/login">Iniciar Sesion</a></li>
                <li><a href="/Secciones/Usuario/Usuario.html">Tu Perfil</a></li>
                <li class="productos-lista"><a
href="http://localhost:3000/productos-pagina"  class="Productos">Productos</a>
            </ul>
        </nav>
    </header>

    <main>
```

```
        <section class="Novedades">
            <article class="Contenedor">
                <h1>Novedades del Dia</h1>

                <!-- Slider main container -->
                <div class=" swiper">
                    <!-- Additional required wrapper -->
                    <div class="slides-container swiper-wrapper" id="slides-
container">
                        <!-- Slides -->
                    </div>

                    <!-- If we need pagination -->
                    <div class="swiper-pagination"></div>
                    <!-- If we need navigation buttons -->
                    <div class="swiper-button-prev"></div>
                    <div class="swiper-button-next"></div>
                </div>

            </article>
        </section>

        <script src="JS/node_modules/swiper/swiper-bundle.min.js"></script>
        <script src="JS/perfil.js"></script>

        <script src="JS/slides.js"></script>
        <script src="JS/ajuste_menu.js"></script>
    </main>

    <footer>
        <section class="Datos">
            <div>

            </div>
        </section>
        <section class="Derechos">
            <div>
                <p>&copy; 2024 Deportes Pirulo. Todos los derechos
reservados.</p>
            </div>
        </section>
    </footer>
</body>
</html>
```

Login.html

```
<!DOCTYPE html>

<html lang="en">
<head>
    <title>Iniciar Sesión</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <link rel="stylesheet" href="../../css/style.css">
</head>
<body>
    <br>
    <main>
        <h1>Iniciar Sesion</h1>

        <form action="http://localhost:3000/usuario" method="post">
            <br>
            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required><br>

            <br>
            <label for="contraseña">Contraseña:</label>
            <input type="password" id="contraseña" name="contraseña"
required><br>

            <input type="hidden" value="1" name="activo">

            <br>
            <input type="submit" value="Iniciar Sesion">
        </form>
        <br>
        <a href="http://localhost:3000/registro"> ¿No tenes cuenta? ingresa aca
</a>
    </main>
    <script src="../../Js/app.js"></script>
</body>
</html>
```

Registro.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Registro</title>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
    <link rel="stylesheet" href="../../css/style.css">
</head>
<body>
    <br>
    <main>
        <h1>Registro</h1>
        <form action="http://localhost:3000/registro" method="post">
            <label for="nombre">Nombre:</label>
            <input type="text" id="nombre" name="nombre" required><br>

            <br>
            <label for="apellido">Apellido:</label>
            <input type="text" id="apellido" name="apellido" required><br>

            <br>
            <label for="email">Email:</label>
            <input type="email" id="email" name="email" required><br>

            <br>
            <label for="contraseña">Contraseña:</label>
            <input type="password" id="contraseña" name="contraseña"
required><br>

            <br>
            <input type="submit" value="Registrar">
        </form>
        <p>¿Ya tienes una cuenta? <a href="http://localhost:3000/login">Inicia
sesión aquí</a></p>
    </main>
    <script src="../../Js/app.js"></script>
</body>
</html>

Producto.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>eCommerce - Productos</title>
    <link rel="stylesheet" href="../../CSS/style.css">
    <link rel="stylesheet" href="https://cdn.jsdelivr.net/npm/swiper@11/swiper-
bundle.min.css"/>
    <style>
        .card-container {
```

```
        display: flex;
        flex-wrap: wrap;
        gap: 16px;
        justify-content: space-around;
    }

    .card {
        border: 1px solid #ccc;
        border-radius: 8px;
        padding: 16px;
        background-color: #f9f9f9;
        width: 200px;
        height: 200px;
        display: flex;
        flex-direction: column;
        justify-content: space-between;
    }

    .name {
        font-size: 1.2em;
        margin-bottom: 8px;
        text-align: center;
    }

    .price {
        font-size: 1em;
        margin-top: 8px;
        margin-bottom: 8px;
        text-align: center;
    }

    #checkout-btn, .add-to-cart-btn {
        background-color: #4CAF50;
        color: white;
        padding: 8px 16px;
        border: none;
        border-radius: 4px;
        cursor: pointer;
        align-self: center;
    }

    #checkout-btn:hover, .add-to-cart-btn:hover {
        background-color: #45a049;
    }
```

```
        #carrito {
            position: fixed;
            top: 20px;
            right: 20px;
            background-color: white;
            border: 1px solid #ccc;
            padding: 16px;
            border-radius: 8px;
            box-shadow: 0 0 10px rgba(0,0,0,0.1);
        }

        #carrito ul {
            list-style-type: none;
            padding: 0;
        }
    </style>
</head>
<body>
    <header class="menu">
        <!-- Tu código del menú aquí -->
    </header>

    <main>
        <section class="Productos">
            <h1>Nuestros Productos</h1>
            <div id="lista-productos" class="card-container"></div>
        </section>

        <aside id="carrito">
            <h2>Carrito de Compras</h2>
            <ul id="carrito-items"></ul>
            <button id="checkout-btn">Finalizar Compra</button>
            <div id="wallet_container"></div>
        </aside>
    </main>

    <footer>
        <!-- Tu código del footer aquí -->
    </footer>

    <!-- Incluir el SDK de Mercado Pago -->
    <script src="https://sdk.mercadopago.com/js/v2"></script>
    <!-- Archivo JavaScript que contiene la lógica -->
    <script src="../../JS/checkout.js" defer></script>
```

```
    <script>
        let carrito = [];
        let id_cliente = 1; // Cambia esto según el cliente actual (esto debería
ser dinámico en una aplicación real)

        function cargarProductos() {
            fetch('/productos')
                .then(response => response.json())
                .then(productos => {
                    const listaProductos = document.getElementById('lista-
productos');
                    productos.forEach(producto => {
                        const card = document.createElement('div');
                        card.className = 'card';

                        const name = document.createElement('h3');
                        name.className = 'name';
                        name.textContent = producto.nombre;
                        card.appendChild(name);

                        const descripcion = document.createElement('p');
                        descripcion.textContent = producto.descripcion;
                        card.appendChild(descripcion);

                        const price = document.createElement('div');
                        price.className = 'price';
                        price.textContent = `Precio:
$${producto.precio_unitario}`;
                        card.appendChild(price);

                        const button = document.createElement('button');
                        button.className = 'add-to-cart-btn';
                        button.textContent = 'Agregar al Carrito';
                        button.onclick = () => agregarAlCarrito(producto);
                        card.appendChild(button);

                        listaProductos.appendChild(card);
                    });
                })
                .catch(error => console.error('Error:', error));
        }

        function agregarAlCarrito(producto) {
            fetch('/iniciar-pedido', {
                method: 'POST',
```

```
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({ id_cliente })
            })
            .then(response => response.json())
            .then(data => {
                const id_pedido = data.id_pedido;

                fetch('/agregar-al-carrito', {
                    method: 'POST',
                    headers: { 'Content-Type': 'application/json' },
                    body: JSON.stringify({
                        id_producto: producto.id_producto,
                        cantidad: 1, // Puedes ajustar la cantidad según sea
necesario
                        id_pedido
                    })
                })
                .then(response => response.json())
                .then(() => {
                    carrito.push(producto);
                    actualizarCarrito();
                })
                .catch(error => console.error('Error:', error));
            })
            .catch(error => console.error('Error:', error));
        }

        function eliminarDelCarrito(index) {
            carrito.splice(index, 1);
            actualizarCarrito();
        }

        function actualizarCarrito() {
            const carritoItems = document.getElementById('carrito-items');
            carritoItems.innerHTML = '';
            carrito.forEach((producto, index) => {
                const li = document.createElement('li');
                li.textContent = `${producto.nombre} -
$${producto.precio_unitario}`;
                const removeBtn = document.createElement('button');
                removeBtn.textContent = 'Eliminar';
                removeBtn.onclick = () => eliminarDelCarrito(index);
                li.appendChild(removeBtn);
                carritoItems.appendChild(li);
            });
```

```
        }

        document.getElementById('checkout-btn').addEventListener('click', () => {
            // Función para manejar el proceso de pago
            handleCheckout();
        });

        function handleCheckout() {
            // Aquí se inicializa el SDK de Mercado Pago
            const mp = new MercadoPago('APP_USR-405752fa-83cd-43b1-bd30-
352314bc1954', {
                locale: "es-AR",
            });

            // Crear una preferencia de pago
            fetch('/create_preference', {
                method: 'POST',
                headers: { 'Content-Type': 'application/json' },
                body: JSON.stringify({
                    title: 'Pedido en Carrito', // Puedes personalizar el título
                    quantity: carrito.length,
                    price: carrito.reduce((total, item) => total +
item.precio_unitario, 0)
                })
            })
            .then(response => response.json())
            .then(preference => {
                // Renderizar el botón de pago de Mercado Pago
                const bricksBuilder = mp.bricks();
                bricksBuilder.create("wallet", "wallet_container", {
                    initialization: {
                        preferenceId: preference.id,
                    },
                    customization: {
                        texts: {
                            valueProp: 'smart_option',
                        },
                    },
                }).then((button) => {
                    // Aquí puedes manejar el botón de pago si es necesario
                }).catch(error => console.error('Error al crear el botón:',
error));
            })
            .catch(error => console.error('Error al crear la preferencia:',
error));
```

```
        }

        // Llamar a la función cuando se carga la página
        window.onload = cargarProductos;
    </script>
</body>
</html>
```

Usuario.html

```
<!DOCTYPE html>
<html lang="es">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>eCommerce</title>
        <link rel="stylesheet" href="../../CSS/style.css">
        <script>
            document.addEventListener('DOMContentLoaded', () => {
                fetch('/usuario')
                    .then(response => response.json())
                    .then(data => {
                        document.getElementById('email').textContent =
data.email;
                        document.getElementById('usuario').textContent =
data.id_cliente;
                    })
                    .catch(error => console.error('Error:', error));
            });
        </script>
    </head>
    <body>
        <header>
            <nav>
                <input type="checkbox" name="" id="check">
                <label for="check" class="checkbtn">
                    <img class="fas fa-bars" src="/Imagenes/Iconos/menu.png"
width="30">
                </label>
                <ul class="contenidos">
                    <li><a class="active" href="/Index.html">Inicio</a></li>
                    <li class="productos-lista"><a
href="#"  class="Productos">Productos</a>
                        <ul class="lista-productos">
                            <li><a href="#">Futbol</a></li>
                            <li><a href="#">Bascket</a></li>
```

```
                        <li><a href="#">Tenis</a></li>
                    </ul>
                </li>
            </ul>
        </nav>
    </header>
    <main>
        <section class="informacion">
            <h1>Bienvenido <span id="email"></span></h1>
            <p>ID Usuario: <span id="usuario"></span></p>
            <p>ID Cliente: <span id="cliente"></span></p>
            <p>Telefono: <span id="direccion"></span></p>
            <p>Direccion: <span id="telefono"></span></p>
        </section>
        <br>
        <a href="/Secciones/Usuario/Cliente.html">Accede como Cliente</a>
    </main>
    <script src="/JS/app.js"></script>
    <script src="/JS/ajuste_menu.js"></script>
</body>
</html>

Cliente.html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="../../CSS/style.css">
</head>
<body>
    <main>
        <br>
        <h1>Registros de Clientes</h1>
        <form action="http://localhost:3000/cliente" method="post">
            <input type="hidden" name="id_persona" value= 1>
            <br>
            <br>
            <label for="direccion">Dirección:</label>
            <input type="text" id="direccion" name="direccion" required>
            <br>
            <br>
            <label for="telefono">Teléfono:</label>
            <input type="text" id="telefono" name="telefono" required>
```


3

```
            <br>
            <br>
            <input type="submit" value="Guardar Cliente">
        </form>
    </main>
    <script src="../../Js/app.js"></script>
</body>
</html>
```

Style.css

```
@import '../JS/node_modules/swiper/swiper-bundle.css';
@import
url('https://fonts.googleapis.com/css2?family=Montserrat:ital,wght@0,100..900;1,100..900&display=swap');

body{
    padding: 0;
    margin: 0;
    box-sizing: border-box;
    background-color: rgb(203, 66, 66);
}

a{
    text-decoration: none;
    color: aquamarine;
}

p, a, h1{
    font-family: "Montserrat", sans-serif;
}

h1{
    font-weight: 400;
    font-size: 50px;
}

/* Menu */

header{
    height: 75px;
    width: 100%;
}

nav{
    max-height: 100px;
```

```
    width: 100%;

    z-index: 1000;
    position: fixed;

    background: rgb(68, 51, 51);
}

.enlace{
    position: absolute;
    padding: 20px 50px;
}

nav ul{
    float: right;
    margin-right: 20px;
}

nav ul li{
    display: inline-block;
    line-height: 80px;
}

nav ul li a{
    font-size: 18px;
    padding: 20px;

    border-radius: 3px;
    text-transform: uppercase;
}

li a:hover{
    background-color: rgb(220, 154, 55);
    transition: .5s;
}

.lista-productos{
    max-width: 150px;

    display: none;
    position: absolute;
    top: 100%;
    right: 0;
}
```

```css
.productos-lista:hover > .lista-productos{
    display: block;
    flex-direction: column;
    background: rgb(68, 51, 51);
}

.checkbtn{
    font-size: 30px;
    color: #fff;
    float: right;
    line-height: 80px;
    margin-right: 40px;
    cursor: pointer;
    display: none;
}

#check{
    display: none;
}

/* Contenido */

main{
    width: 100%;
    max-height: 3000px;
}

.Novedades{
    width: 100%;
    max-height: 1000px;
}

.Contenedor{
    width: 100%;
    height: 600px;
}

.swiper{
    width: 100%;
    height: 500px;
}

.slides-conteiner{
    width: 100%;
    height: 500px;
```

```
}

.swiper-slide{
    width: 450px;
    height: 450px;
    visibility: visible;

    padding: 25px;
    border-radius: 100px;
    overflow: hidden;
    background-color: rgb(219, 202, 202);
}


/* Pie de PAgina */

footer{
    width: 100%;
    max-height: 1000px;
    display: flex;
}

.Datos{
    width: 40%;
    height: 100px;
}

.Derechos{
    width: 60%;
    height: 100px;
}

.secciones{
    width: 100%;
    height: 100px;
}

@media (max-width: 952px){
    .enlace{
        padding-left: 20px;
    }
    nav ul li a{
        font-size: 16px;
    }
}
```

```
@media screen and (max-width: 800px) {
    .Contenedor{
        width: 100%;
        height: 750px;
    }

    .swiper{
        width: 100%;
        height: 700px;
    }

    .slides-conteiner{
        width: 100%;
        height: 650px;
    }

    .swiper-slide{
        width: 500px;
        height: 550px;
    }
}

@media screen and (max-width: 700px) {
    h1{
        font-size: 25px;
    }

    /*Contenidos*/

    .Novedades{
        max-height: 600px;
    }

    .Contenedor{
        max-height: 550px;
    }

    .swiper{
        height: 400px;
    }

    .swiper-slide{
        width: 250px;
        height: 300px;
```

```
    }
}

@media screen and (max-width: 450px){
    /*Menu*/

    .checkbtn{
        display: block;
    }

    ul{
        width: 100%;
        height: 100%;

        bottom: 100%;
        position: fixed;
        background: #2c3e50;

        text-align: center;
        transition: all .5s;
    }

    .lista-productos{
        display: block;
        text-align: left;
    }

    nav ul li{
        display: block;
        margin: 50px 0;
        line-height: 30px;
    }

    nav ul li a{
        font-size: 20px;
    }

    li a:hover{
        background: none;
        color: red;
    }

    #check:checked ~ ul{
        bottom: -95px;
    }
```

```
    .Productos:hover ~ .lista-productos{
        bottom: -95px;
    }
}
```

Menu.png

Registro de experiencia:

El desarrollo de una aplicación web para una tienda especializada en productos de deportes al aire libre ha sido una experiencia enriquecedora y desafiante en múltiples aspectos. Este proyecto no solo nos permitió aplicar nuestros conocimientos técnicos, sino que también nos confrontó con retos relacionados con el tiempo y la organización.

Aspecto Técnico: En el ámbito técnico, la aplicación demandó el uso de tecnologías avanzadas como JavaScript y Node.js. JavaScript, como lenguaje de programación fundamental para el desarrollo web, nos permitió implementar funcionalidades interactivas en el front-end. Por otro lado, Node.js fue crucial para el desarrollo del back-end, facilitando la creación de un servidor robusto y eficiente. A lo largo del proyecto, tuvimos que aprender y adaptar nuestras habilidades a estas tecnologías, lo que implicó una curva de aprendizaje significativa. La integración de estos dos componentes nos obligó a entender mejor cómo interactúan el front-end y el back-end, así como a resolver problemas relacionados con la sincronización de datos y la gestión de sesiones.

Aspecto Temporal: El tiempo fue un factor crítico en este proyecto. Con un plazo de solo cinco días y unas pocas horas disponibles por día, cada minuto contaba. La necesidad de trabajar bajo presión y cumplir con los plazos ajustados nos obligó a optimizar nuestro flujo de trabajo y a tomar decisiones rápidas, pero bien fundamentadas. Este desafío nos enseñó a priorizar tareas, gestionar el tiempo de manera eficiente y mantener un enfoque en los objetivos clave para garantizar que todas las funcionalidades básicas de la aplicación estuvieran operativas dentro del plazo establecido.

En conclusión, el desarrollo de esta aplicación web fue una experiencia multifacética que nos desafió a nivel técnico y temporal. Nos permitió fortalecer nuestras habilidades en tecnologías web, mejorar nuestra capacidad para trabajar bajo presión y perfeccionar nuestras competencias organizativas. A pesar de las dificultades, el éxito del proyecto ha sido un testimonio de nuestra capacidad para enfrentar desafíos y lograr resultados efectivos en un entorno dinámico.

Experiencia del trabajo grupal:

En el desarrollo de este proyecto web para una tienda de productos deportivos al aire libre, tuvimos la oportunidad de trabajar juntos tres compañeros que no solo somos colegas, sino también amigos. Esta relación cercana fue fundamental para el éxito del proyecto, ya que conocíamos las fortalezas y debilidades de cada uno y esto nos permitió tener una conexión fluida y efectiva durante el trabajo.

Nuestro conocimiento previo de cómo cada uno aborda las tareas nos facilitó una colaboración más armoniosa y eficiente. Esta familiaridad no solo mejoró nuestra comunicación, sino que también ayudó a asignar tareas de manera que aprovecháramos al máximo nuestras habilidades individuales. A pesar de enfrentar varios desafíos, como trabajar con tecnologías nuevas y cumplir con un plazo muy ajustado, siempre supimos manejar las adversidades con respeto y compañerismo. Este enfoque positivo y colaborativo nos permitió mantener la motivación alta, incluso cuando los requisitos del proyecto parecían difíciles de cumplir.

A lo largo del proceso, nuestra capacidad para enfrentar problemas juntos y aprender sobre la marcha demostró que, a pesar de las dificultades y las limitaciones, éramos capaces de lograr resultados significativos. Aunque no pudimos cumplir con todos los requisitos iniciales debido al tiempo limitado y a la complejidad de las nuevas tecnologías, consideramos que el verdadero logro para nuestro pequeño grupo fue haber enfrentado un proyecto que combinaba tareas desconocidas y desafíos técnicos en un tiempo récord. Esta experiencia no solo amplió nuestras habilidades, sino que también reforzó el valor del trabajo en equipo y la perseverancia.

Aclaraciones:

Durante el desarrollo de la aplicación web para la tienda de productos deportivos, hubo ciertos aspectos que no se completaron como se había planeado inicialmente. A continuación, se detallan las principales aclaraciones relacionadas con los entregables y el desarrollo de la página:

1. **Funcionalidades No Implementadas:** Lamentablemente, no logramos implementar todas las funcionalidades requeridas para la página. Las funcionalidades que quedaron pendientes incluyen:
    - Logueo del equipo de ventas.
    - Añadir productos a la base de datos.
    - Generación de pedidos y facturas.
    - Gestión del carrito de compras, incluyendo la adición y eliminación de productos. Cada una de estas funcionalidades está vinculada a la base de datos y su falta de implementación afecta la operatividad completa de la aplicación.
2. **Entregables Incompletos:** En relación con los entregables específicos, los siguientes elementos no están presentes o no se completaron en su totalidad:
    - **Credenciales de Usuario Interno (Jefe de Ventas):** No se proporcionaron credenciales para un usuario interno de la compañía con acceso completo a la aplicación, incluyendo los permisos necesarios para las funciones asignadas a este perfil.
    - **URL de la Aplicación:** La URL de la aplicación, instalada y funcionando, no está disponible. Esto significa que no se puede acceder a la aplicación en línea en su forma actual.
    - **Imágenes de Pantallas:** No se han proporcionado imágenes de pantallas que representen las actividades de los dos actores principales: clientes y jefe de ventas. Estas imágenes deben ser capturas de pantalla con datos reales para ilustrar las funciones de la aplicación.
    - **Vídeo de Funcionamiento:** El vídeo que muestra el funcionamiento del sistema no ha sido subido a la nube (como Google Drive, Microsoft OneDrive, etc.). Este vídeo debería mostrar el uso de la aplicación y proporcionar una demostración de sus características.
    - **Ayudas en Línea y Manuales:** No se han incluido ayudas en línea ni manuales de uso para las aplicaciones del jefe de ventas. Estos documentos son esenciales para guiar a los usuarios sobre cómo utilizar las funciones específicas del sistema.

Estas aclaraciones son importantes para comprender el alcance del proyecto y las limitaciones actuales. A pesar de los desafíos y las funcionalidades no completadas, el proceso de desarrollo ha sido una valiosa experiencia de aprendizaje. Agradecemos su comprensión y estamos dispuestos a discutir cualquier aspecto adicional relacionado con el proyecto.

Bibliografía:

Mailchimp. (13 de mayo de 2021). ¿Qué es un sitio web de E-Commerce? *Mailchimp*. https://mailchimp.com/es/marketing-glossary/ecommerce-website/#%C2%BFQu%C3%A9_es_un_sitio_web_de_E-Commerce%3F

Euroinnova. (s. f.). Requisitos para un gerente de ventas. *Euroinnova*. https://www.euroinnova.com/blog/requisitos-para-un-gerente-de-ventas

Brandnlabel. (7 de septiembre de 2020). 10 requisitos básicos de un E-Commerce para los clientes. *Brandnlabel*. https://www.brandnlabel.com/customer-experience/10-requisitos-basicos-de-un-ecommerce-para-los-clientes/

IUSPORT. (10 de abril de 2024). ¿Cuáles son los deportes más populares en Argentina? *IUSPORT*. https://iusport.com/art/130909/cuales-son-los-deportes-mas-populares-en-argentina

Offshore Company Corp. (s. f.). ¿Cuáles son los requisitos mínimos de un sitio web de comercio electrónico? *Offshore Company Corp*. https://www.offshorecompanycorp.com/es/es/faq/what-is-minimum-requirements-of-e-commerce-website?creative=608159718924&keyword=&matchtype=&network=g&device=c&idCampaign=17652218806&&idGroupAds=142077240887&idAds=608159718924&idAccount=9223406146&gad_source=1&gclid=Cj0KCQjw5ea1BhC6ARIsAEOG5pxvFp-zjVL-x_MTKshTTM6WDJoQIu6ZFA8dqMS5IpEBUKzVOGYdWFMaArskEALw_wcB

Mercado Pago. (n.d.). *Integrar checkout pro: Documentación de checkout pro*. Mercado Pago. https://www.mercadopago.com.ar/developers/es/docs/checkout-pro/integrate-checkout-pro/web

on the code. (7 dic. 2023). *Integracion de Mercado Pago con Javascript* [Video]. YouTube.https://www.youtube.com/watch?v=vEXwN9-tKcs&t=183s