



MARRET CAFÉ

PROYECTO FINAL N°6 — DISEÑO Y DESARROLLO DE SOFTWARE

PROFESOR: CRISTIAN IGLESIAS VERA

EQUIPO: TEAM MARRET
INTEGRANTES

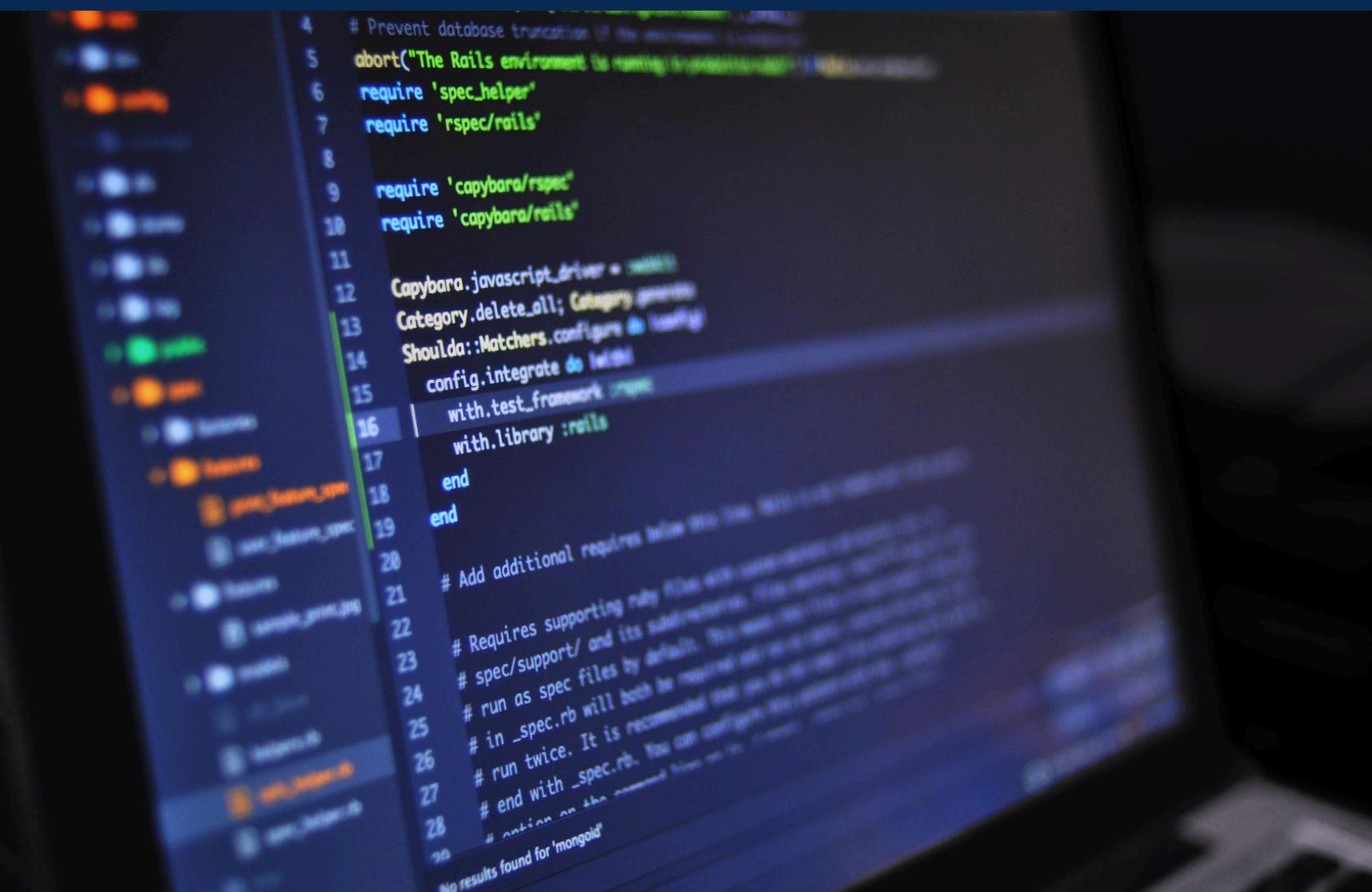
PRESENTES: NICOLÁS HUENCHUAL, FELIPE SALAZAR,
MICHAEL FLORES.

PROYECTO: CAFETERÍA Y PASTELERÍA MARRET.

CLIENTE/SOLICITANTE: EMPRENDIMIENTO MARRET CAFÉ.

INTRODUCCIÓN

ESTE PROYECTO SE BUSCA DIGITALIZAR LA GESTIÓN DE PRODUCTOS, RESERVAS Y PEDIDOS, UTILIZANDO PATRONES DE DISEÑO, ARQUITECTURA MVC Y UNA APP WEB FUNCIONAL QUE FACILITE LA ATENCIÓN Y MEJORE LA EXPERIENCIA DEL CLIENTE



```
4 # Prevent database truncation if the migration fails
5 abort("The Rails environment is running in production mode!
6 require 'spec_helper'
7 require 'rspec/rails'

8 require 'capybara/rspec'
9 require 'capybara/rails'

10
11 Capybara.javascript_driver = :webkit
12 Category.delete_all; Category.create!(name: "Electronics")
13 Shoulda::Matchers.configure do |config|
14   config.integrate do |sp|
15     sp.with.test_framework :rspec
16     sp.with.library :rails
17   end
18 end
19
20
21 # Add additional requires below this line if necessary
22
23 # Requires supporting ruby files with custom matchers and helpers
24 # in spec/support/ and its subdirectories. This directory also
25 # contains supporting files for this generator. Custom matchers and
26 # helpers can be added by placing them into this directory. They are
27 # run as spec files by default. You can also run:
28 # --require ./support/**/*.rb when you run the tests. If you
29 # want to run a specific file from support, specify it in the
30 # --require option.
```



1/ PROBLEMA Y CONTEXTO

Problemas detectados:

- Largas esperas en cafeterías tradicionales.
- Poca variedad y falta de frescura en productos.
- Mala comunicación entre negocio y clientes. Usuario

Impacto esperado: una atención ágil, productos frescos y mayor fidelización

REQUERIMIENTOS PRINCIPALES:

- GESTIÓN DE PRODUCTOS (CAFÉS, PASTELES).

GESTIÓN BÁSICA DE PRODUCTOS Y VISUALIZACIÓN DE MENÚ.

- RESERVAS Y COMUNICACIÓN BÁSICA CON CLIENTES.

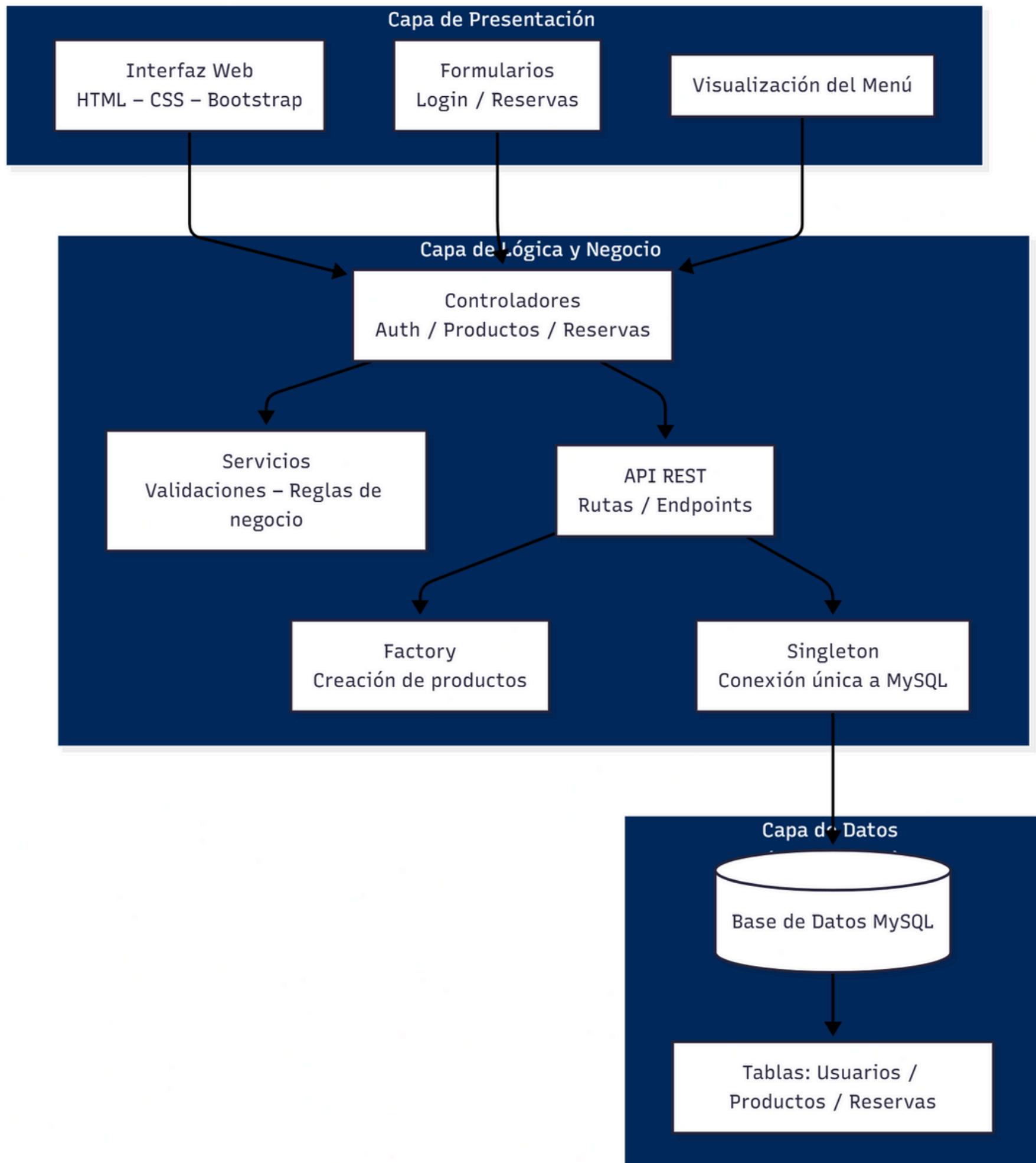
ALCANCE Y REQUISITOS

CRITERIOS DE ACEPTACIÓN IMPLEMENTADOS:

- CRUD completo de productos.
- Conexión única estable a MySQL (Singleton).
- Creación dinámica de productos (Factory).
- Respuestas correctas de la API vista en área local con Postman.



3 / ARQUITECTURA Y DISEÑO



IMPLEMENTACIÓN Y LÓGICA

STACK TECNOLÓGICO:

- PHP PARA BACKEND Y LÓGICA PRINCIPAL.

- MYSQL PARA PERSISTENCIA.

- POSTMAN PARA PRUEBAS DE ENDPOINTS.

ASPECTOS CRÍTICOS:

- PERSISTENCIA USANDO CONSULTAS PARAMETRIZADAS PARA EVITAR INYECCIONES SQL.

- SERVICIOS SEPARADOS PARA PRODUCTOS, USUARIOS Y RESERVAS.

- API REST CON RUTAS CLARAS Y VALIDACIONES.

- PLANTILLAS Y ORGANIZACIÓN MVC PARA MANTENER SEPARACIÓN DE CAPAS.

4 / DEMOSTRACIÓN FUNCIONAL (DEMO EN LOCAL)



PRUEBAS, MÉTRICAS Y MANTENIMIENTO

PRUEBAS REALIZADAS:

- PRUEBAS FUNCIONALES EN LOGIN Y VISUALIZACIÓN DEL MENÚ.
- VALIDACIÓN DE ENDPOINTS BÁSICOS EN POSTMAN (/PRODUCTOS, /USUARIOS/LOGIN).
- VERIFICACIÓN DE CONEXIÓN ESTABLE Y RESPUESTAS CORRECTAS DESDE MYSQL.

MÉTRICAS APLICADAS:

- REGISTRO Y REVISIÓN DE ERRORES DURANTE EL DESARROLLO.
- COMPROBACIÓN DE TIEMPOS DE RESPUESTA DE LA API.
- VALIDACIÓN DE CONSISTENCIA DE DATOS ENVIADOS Y RECIBIDOS.

MANTENIMIENTO:

- DOCUMENTACIÓN DEL PROYECTO EN GITHUB.
- VERSIONADO DEL CÓDIGO MEDIANTE GIT.
- VERIFICACIÓN MANUAL DEL ESTADO DE LA BASE DE DATOS.

Uso de IA y Ética

IA utilizada:

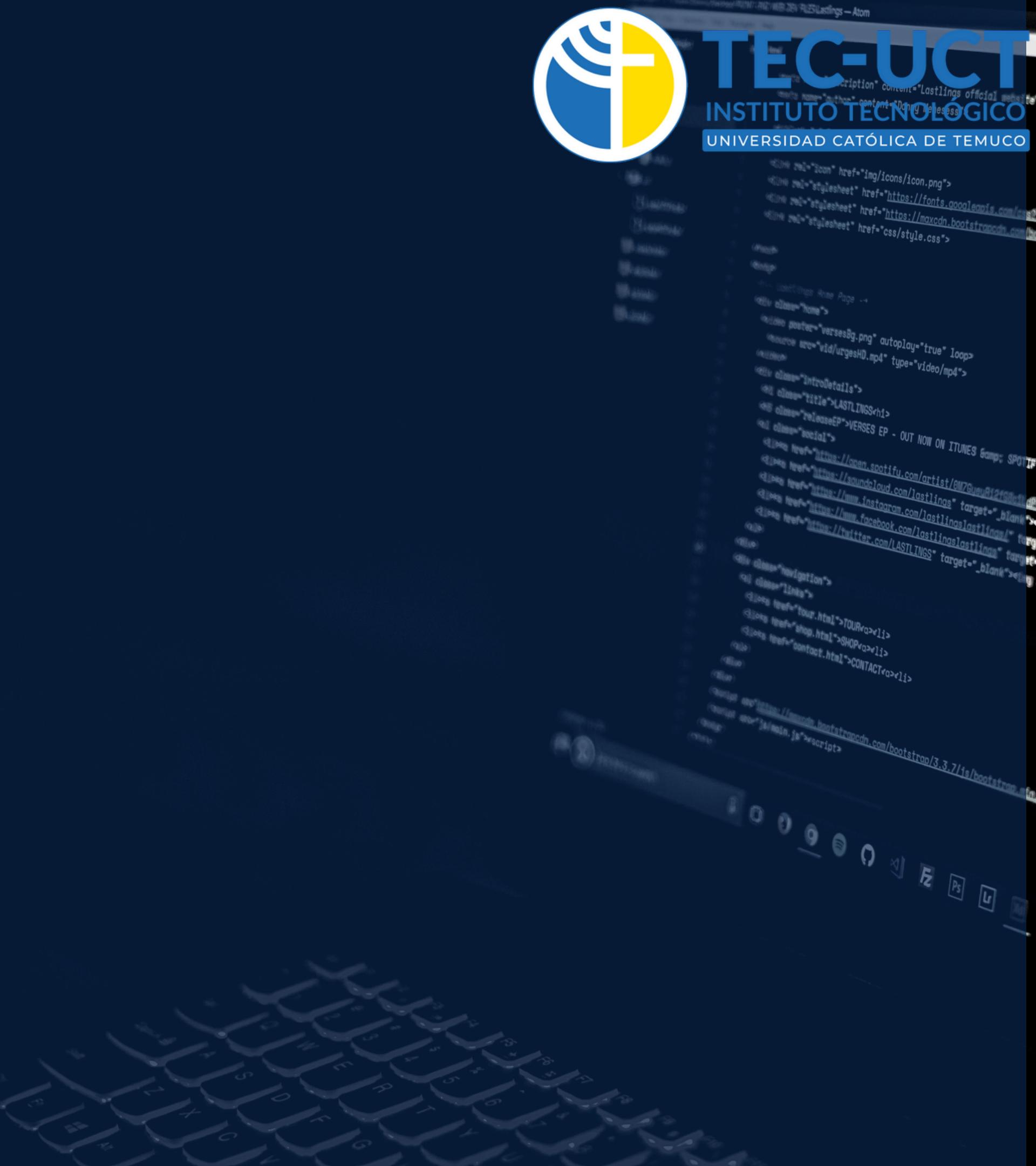
- ChatGPT (GPT-5) (Parte técnica.)
- DeepSeek (apoyo en generación y Diseños)

Tareas con IA:

- Redacción técnica del informe.
- Generación de código base para patrones.
- Estandarización del lenguaje del documento.

Consideración ética:

La IA se utilizó solo como apoyo; el diseño, la programación y la validación final fueron realizados completamente por el equipo.





CIERRE Y APRENDIZAJES

```
4 # Prevent database truncation if the database needs clearing.
5 abort("The Rails environment is running in production mode!
6 require 'spec_helper'
7 require 'rspec/rails'
8
9 require 'capybara/rspec'
10 require 'capybara/rails'
11
12 Capybara.javascript_driver = :webkit
13 Category.delete_all; Category.create(name: "Electronics")
14 Shoulda::Matchers.configure do |config|
15   config.integrate do |with|
16     with.test_framework :rspec
17     with.library :rails
18   end
19 end
20
21 # Add additional requires below this line to append them to all features
22 # require 'page-object'
23 # require 'factory_girl_rails'
24
25 # Requires supporting ruby files with custom matchers and helpers
26 # in spec/support/ and its subdirectories. This folder is not included
27 # in spec/rails by default. You can add it to your own application
28 # by following this guide: https://relishapp.com/rspec/rspec-rails/docs
29
30 # run as spec files by default. You can override this using --format
31 # or --color options (refer to the documentation). To run tests in
32 # the browser, specify :ui before the spec file name.
33 # in _spec.rb will both be required. If you want to yield a value
34 # between the steps, you can use a block, e.g.
35 # end with _spec.rb. You can configure the
36 # options which are used by default in
37 # configuration or in the .rspec file
38
39 #民族
40 # mongoid
41
42 # No results found for 'mongoid'
```

Aprendizaje:

A lo largo del proyecto aprendimos a crear y configurar un sitio web, programar sus funcionalidades y desarrollar una base de datos completa. También incorporamos el uso de IA para apoyar en tareas técnicas y en la generación de diseños. Este proceso fortaleció nuestros conocimientos y nos permitió integrar mejor la tecnología en el desarrollo del proyecto.

Pasos a seguir:

Aún como equipo nos falta completar las secciones de compras, gestión de stock y el panel de administrador, que serán parte del siguiente avance a futuro de este proyecto.

MUCHAS GRACIAS

```
    render() {
      return (
        <React.Fragment>
          <div className="py-5">
            <div className="container">
              <Title name="our" title="product">
              <div className="row">
                <ProductConsumer>
                  {(value) => {
                    |   |   |   console.log(value)
                  }}
                </ProductConsumer>
              </div>
            </div>
          </div>
        <React.Fragment>
```