

**FACULTAD DE INGENIERÍA**

**Ingeniería de Sistemas e Informática**

**Título:**

**Proyecto: “Plataforma Interna de Gestión de Pedidos y Ventas para Sol y Mar Comida Marina”**

**(Tercer avance)**

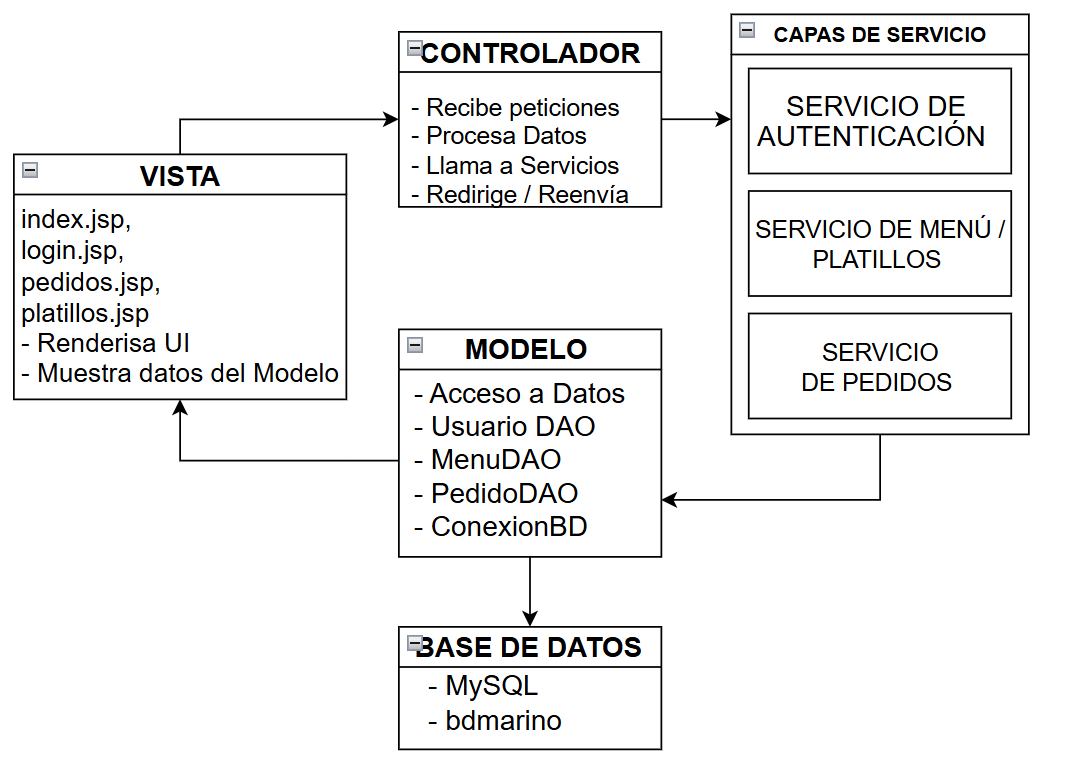
**Alumnos:**

* **Pérez Tovar, Elian Jeanpierr U23220848**
* **Ramírez Gallegos, Braulio Joel** **U20305755**

**2025**

**Lima-Perú**

CAPITULO III

3.1 Arquitectura y Construcción del Sistema.

Componentes y Principios SOLID:

VISTA (JSP / Interfaz de Usuario):

* Responsabilidad Única (SRP): Su única responsabilidad es mostrar la interfaz de usuario y los datos. No contiene lógica de negocio ni acceso a datos.
* Ventanas involucradas:
  + login.jsp: Inicio de Sesión
  + opciones.jsp: Ventana de Opciones / Menú Principal
  + platillos.jsp: Ventana de Platillos para Escoger
  + pedidos.jsp: Solicitud

CONTROLADOR (Servlets):

* Responsabilidad Única (SRP): Recibe las peticiones del usuario, valida la entrada, delega la lógica de negocio a la Capa de Servicios, y decide qué vista mostrar.
* Abierto/Cerrado (OCP): Con una buena estructura (usando un Servlet principal que delega acciones), puedes añadir nuevas acciones sin modificar su código principal.
* Ventana de Opciones y Platillos: Un único Servlet (MainController o GestionController) podría manejar varias acciones (accion=login, accion=mostrarMenu, accion=hacerPedido).

CAPA DE SERVICIOS (Lógica de Negocio):

* Inversión de Dependencias (DIP): Los controladores dependen de abstracciones de servicios (interfaces), no de implementaciones concretas, lo que facilita las pruebas y el cambio de implementaciones.
* Segregación de Interfaz (ISP): Si los servicios tienen muchas responsabilidades, se pueden dividir en interfaces más pequeñas y específicas.

MODELO (DAO - Data Access Object):

* Responsabilidad Única (SRP): Cada DAO (UsuarioDAO, MenuDAO, PedidoDAO) tiene la única responsabilidad de interactuar con la base de datos para una entidad específica. No contiene lógica de negocio.
* Abierto/Cerrado (OCP): Puedes añadir nuevos métodos de acceso a datos sin modificar el DAO existente.
* Inversión de Dependencias (DIP): La Capa de Servicios tiene que depender de interfaces de DAO, para facilitar el cambio de bases de datos o la simulación en pruebas.

BASE DE DATOS (MySQL):

Interfaz de usuario gráfica, Aplicación

El contenido generado por IA puede ser incorrecto.

FASE 1: Login de Usuario y Módulo de Pedido para Mesa

1. Módulo de Autenticación (Inicio de Sesión)

Vistas:

* login.jsp: Formulario para ingresar usuario y contraseña.
* opciones.jsp: La página a la que se redirige después de un login exitoso.

Modelo (DAO):

* Necesitarás una tabla de usuarios en bdmarino.
* UsuarioDAO.java: Con métodos como validarUsuario (String username, String password).

Capa de Servicios:

* ServicioAutenticacion.java: Con un método autenticar (String username, String password). Este servicio usará UsuarioDAO.

Controlador:

* LoginController.java (MainController que maneje la acción login):
  + Recibe los datos del formulario de login.jsp.
  + Llama al ServicioAutenticacion.
  + Si es exitoso, redirige a opciones.jsp.
  + Si falla, vuelve a login.jsp con un mensaje de error.

1. Módulo de Pedido para Mesa (Ventana de Platillos)

* Vistas:

- opciones.jsp: Para el menú principal, que tendrá un enlace para "Hacer Pedido" o "Ver Platillos".

- platillos.jsp: Muestra la lista de platillos disponibles (la tabla que ya intentaste mostrar).

* Modelo (DAO):

- MenuDAO.java: Con el método listar () que ya tienes.

- Tabla menú en bdmarino: Asegúrate de que tiene columnas como id, nombre, descripción, precio, fecha\_creacion, imagen.

Capa de Servicios:

* ServicioMenu.java: Con un método obtenerTodosLosPlatillos(). Este servicio usará MenuDAO.
* Controlador:

MenuController.java (o tu MainController que maneje la acción mostrarPlatillos):

* + - Llama al ServicioMenu.
    - Obtiene la lista de platillos.
    - Pasa la lista a platillos.jsp (usando request.setAttribute y RequestDispatcher.forward).

**Texto

El contenido generado por IA puede ser incorrecto.3.2 Estructura del Proyecto**

Esta estructura del proyecto está organizada en una aplicación web en Java con una clara separación entre las capas de presentación (JSP), lógica de negocio (Controlador), acceso a datos (DAO), y las entidades del modelo de datos. También se utiliza un servidor GlassFish y se conecta a una base de datos MySQL.

**3.3 Herramientas de desarrollo y sistemas de gestión de bases de datos**

Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

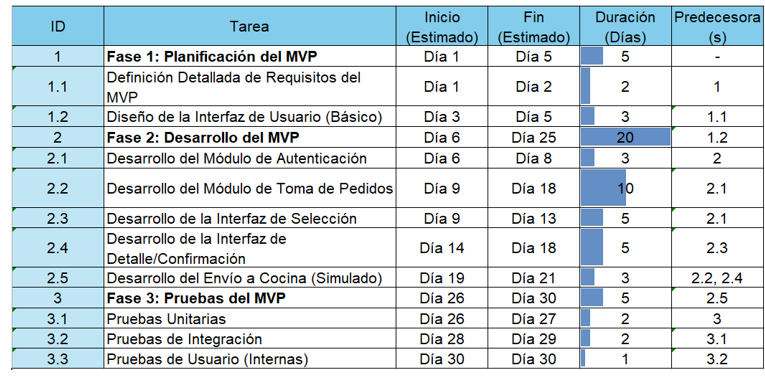
Interfaz de usuario gráfica, Texto

El contenido generado por IA puede ser incorrecto.

Texto

El contenido generado por IA puede ser incorrecto.Texto, Carta

El contenido generado por IA puede ser incorrecto.

**Diagrama GANTT**

Bibliografía

* All Pet Food Magazine. (2021). El Perú de las mascotas: el mercado Pet Food en 2021. Edición N° 9.
* Semana Económica. (2022). Industry Insights. pp. 29–36.
* Sodabathina, R., Singh, A., Stehle, J. (2022). Architecting near real-time personalized recommendations with Amazon Personalize. Amazon Web Services.
* Archenti, A. (2012). Metodología de las Ciencias Sociales. Buenos Aires: Cengage Learning.
* ISO/IEC/IEEE 29148:2018. Systems and software engineering — Life cycle processes — Requirements engineering. International Organization for Standardization.
* Balsamiq. (2023). Mockup Tools for UI Design. Disponible en: <https://balsamiq.com>