Aplicaciones Informáticas 2

- Nombre: Jennifer Yambay – 6916
- Fecha: 17 – 05 – 2025
- Docente: Ing. Julio Santillán

ARQUITECTURA DEL SISTEMA

La arquitectura del sistema es una arquitectura de software de tipo cliente-servidor, donde una aplicación web interactúa con una base de datos PostgreSQL. Esta arquitectura permite separar responsabilidades, mejorar la escalabilidad, facilitar el mantenimiento y garantizar una experiencia de usuario fluida y segura.

1. Descripción General de la Arquitectura

El sistema se divide en tres capas principales que trabajan de forma integrada para ofrecer un servicio completo:

- Capa de Presentación (Frontend):

Esta capa representa la interfaz gráfica con la que interactúan los usuarios del sistema. Es desarrollada generalmente con tecnologías como React, HTML, CSS y JavaScript. Permite mostrar datos, recibir entradas del usuario y comunicarse con el backend a través de peticiones HTTP o AJAX.

- Capa de Lógica de Negocio (Backend):

Aquí se implementan todas las reglas del sistema, validaciones y control de datos. Utiliza Django como framework web, que incluye características como autenticación de usuarios, ORM para interactuar con la base de datos, y generación de APIs RESTful mediante Django REST Framework. El backend recibe las peticiones del frontend, procesa la lógica correspondiente y responde con datos o acciones adecuadas.

- Capa de Datos (Base de Datos):

Es el lugar donde se almacena toda la información persistente del sistema. Utiliza PostgreSQL y está compuesta por un conjunto de tablas relacionales que representan entidades como usuarios, productores, entregas, pagos, reportes y precios. La estructura está normalizada y optimizada para integridad y rendimiento.

2. Componentes del Sistema

A continuación se detallan los módulos funcionales que constituyen el sistema:

- Gestión de Usuarios y Autenticación:

Utiliza las tablas proporcionadas por Django como auth_user, auth_group, django_session, entre otras. Esto permite una gestión robusta de acceso a través de usuarios, permisos y grupos, así como el seguimiento de sesiones activas. Además, se incluye funcionalidad para restablecimiento de contraseñas y autenticación segura mediante tokens y sesiones.

- Módulo de Productores:

Registra la información básica de cada productor, incluyendo nombre, contacto y dirección. Esta información es utilizada para relacionar entregas y pagos con los productores registrados en el sistema.

- Módulo de Entregas:

Administra las entregas de productos (posiblemente leche o productos agrícolas). Cada entrega está asociada a un productor y a un usuario del sistema que la registra. Contiene datos como fecha y cantidad entregada, fundamentales para la gestión logística y financiera.

- Módulo de Pagos:

Controla los pagos efectuados a los productores por sus entregas. Incluye fecha de pago, monto total y productor asociado. Puede estar vinculado con los módulos de entregas y precios para automatizar cálculos.

- Módulo de Reportes:

Genera reportes sobre el funcionamiento del sistema o el desempeño de productores y entregas. Estos reportes se vinculan con los usuarios que los elaboran, y pueden incluir datos consolidados o exportables.

- Módulo de Gestión de Precios:

Permite definir y registrar los precios por unidad entregada en diferentes fechas. Esto permite la actualización dinámica de precios y el cálculo automático de pagos según la fecha de entrega.

3. Diagrama Conceptual de la Arquitectura

La interacción entre las capas puede representarse de la siguiente forma lógica:

Usuario (Navegador Web) → Interfaz React → API REST de Django → Lógica de negocio → Base de Datos PostgreSQL

- El usuario realiza acciones en la interfaz web.
- React realiza solicitudes al backend usando HTTP.
- Django interpreta estas solicitudes, aplica lógica de negocio y valida datos
- Finalmente, Django accede a la base de datos PostgreSQL para obtener o guardar información.

4. Seguridad y Escalabilidad

La seguridad se gestiona principalmente desde Django, aprovechando sus herramientas de autenticación, autorización y gestión de sesiones. Se puede utilizar HTTPS, cifrado de contraseñas con hashing, validaciones, y control de accesos para roles diferenciados.

En términos de escalabilidad, el sistema puede escalar horizontalmente mediante la separación de servicios (frontend/backend), uso de contenedores (Docker), balanceadores de carga y replicación de bases de datos. Esto permite soportar un mayor número de usuarios concurrentes y asegurar la disponibilidad del sistema.