# TALLER ARQUITECTURA DE APLICACIONES WEB

SEBASTIAN ALBIS

PIRATOBA VELOZA NICK

THOMAR RAMIREZ

FRANCK JORDAN

UNIVERSIDAD DE PAMPLONA

FACULTAD DE INGENIERÍAS Y ARQUITECTURA

PROGRAMA INGENIERÍA DE SISTEMAS

DESARROLLO ORIENTADO A PLATAFORMAS

VILLA DEL ROSARIO

2025

La arquitectura de aplicaciones web es fundamental para diseñar sistemas eficientes y escalables. Permite que los distintos componentes de una aplicación interactúen de manera coherente.

1. ¿Qué es la Arquitectura de las Aplicaciones Web?

* Es la estructura que define cómo se organizan los componentes de una aplicación web (frontend, backend, base de datos, almacenamiento, red, etc.) y cómo interactúan entre sí para brindar servicios de forma eficiente, segura y escalable.

1. ¿Por qué es importante la Arquitectura de las Aplicaciones Web?

* Porque garantiza rendimiento, escalabilidad, seguridad y mantenibilidad. Una buena arquitectura permite crecer en número de usuarios, integrar servicios externos (pasarelas de pago, mensajería, APIs), reducir riesgos de fallos y mejorar la experiencia del usuario.

1. ¿Cuáles son los Componentes de la Aplicación Web?

Frontend (cliente / navegador): interfaz gráfica y lógica de presentación.

* Backend (servidor / API): lógica de negocio, autenticación, seguridad, procesos.
* Base de datos: persistencia de información.
* Almacenamiento de archivos: imágenes, documentos, archivos estáticos.
* CDN / caché: optimizar tiempos de respuesta.
* Servicios externos: pagos, correo, mensajería.
* Infraestructura: servidores, balanceadores, DNS.
* Monitoreo y logs: seguimiento del rendimiento y errores.

1. Mencione las Mejores Prácticas para el desarrollo web

* Uso de HTTPS obligatorio.
* Arquitectura en capas para separar responsabilidades.
* Aplicar metodologías ágiles y CI/CD (automatización de despliegues).
* Validar y sanitizar entradas de usuario para evitar ataques (SQL injection, XSS).
* Diseñar interfaces accesibles y responsivas.
* Backups y planes de recuperación.
* Monitoreo de rendimiento y errores en producción.

1. Escriba y explique brevemente los Tipos de Arquitectura de Aplicaciones Web, sus ventajas y desventajas

**1. Arquitectura Monolítica**

La aplicación completa (frontend, backend y base de datos) se desarrolla y despliega como una única unidad.

**Ventajas:**

* Sencilla de desarrollar y mantener en proyectos pequeños.
* Menor costo inicial de infraestructura.
* Fácil de probar y desplegar.

**Desventajas:**

* Dificultad para escalar por componentes.
* Los fallos en un módulo pueden afectar toda la aplicación.
* Código más complejo de mantener a medida que crece el proyecto.

**2. Arquitectura de Microservicios**

Divide la aplicación en servicios pequeños e independientes, comunicados generalmente mediante APIs.

**Ventajas:**

* Escalabilidad independiente por cada servicio.
* Despliegues y actualizaciones más flexibles.
* Favorece el trabajo en equipos distribuidos.

**Desventajas:**

* Requiere mayor complejidad de infraestructura.
* Necesita monitoreo, orquestación y comunicación entre servicios.
* Aumenta la dificultad de pruebas e integración.

**3. Arquitectura Serverless (Functions as a Service – FaaS)**

El backend se implementa en funciones que se ejecutan bajo demanda en la nube.

**Ventajas:**

* Pago por uso, lo que reduce costos para cargas variables.
* Escalado automático sin intervención del equipo de desarrollo.
* Ideal para procesos puntuales y eventos.

**Desventajas:**

* Limitaciones de tiempo de ejecución y recursos.
* Dependencia fuerte del proveedor (vendor lock-in).
* No adecuada para aplicaciones de gran carga constante.

**4. Arquitectura SPA (Single Page Application) con Backend-for-Frontend (BFF)**

El frontend funciona como una aplicación dinámica de una sola página (React, Vue, Angular), que consume un backend especializado adaptado a sus necesidades.

**Ventajas:**

* Experiencia de usuario más fluida y rápida.
* Optimización de APIs para diferentes clientes (móvil, web).
* Mejora la usabilidad y reduce recargas innecesarias.

**Desventajas:**

* Mayor complejidad en la coordinación entre frontend y backend.
* Dificultades en SEO si no se maneja bien el renderizado.
* Requiere mayor control de seguridad en el intercambio de datos.

1. Escriba Ejemplos de Tecnologías aplicada a:
   1. Frontend: React, …
   2. Backend: ASP.NET …
   3. Base de Datos: SQL Server, …
   4. Almacenamiento de Imágenes: Amazon S3, ..

**a. Frontend (interfaz de usuario):**

* React (JavaScript, librería para SPAs).
* Angular (framework completo de Google).
* Vue.js (framework progresivo, ligero y flexible).
* Svelte (framework moderno con gran rendimiento).
* Next.js (framework basado en React con renderizado del lado del servidor).

**b. Backend (servidor / lógica de negocio):**

* ASP.NET Core (framework de Microsoft para aplicaciones robustas).
* Node.js (JavaScript en el servidor, junto con Express o NestJS).
* Django (framework en Python, orientado a rapidez y seguridad).
* Spring Boot (framework en Java para aplicaciones empresariales).
* Ruby on Rails (framework en Ruby, rápido para prototipado).

**c. Base de Datos (gestión de información):**

* SQL Server (Microsoft, relacional)
* MySQL (muy usado, open source).
* PostgreSQL (relacional, robusto y avanzado).
* MongoDB (NoSQL, orientado a documentos).
* Oracle Database (relacional empresarial).

d. **Almacenamiento de Imágenes (archivos estáticos/objetos):**

* Amazon S3 (servicio de almacenamiento en la nube de AWS).
* Google Cloud Storage (servicio de Google para archivos).
* Azure Blob Storage (almacenamiento de objetos de Microsoft).
* Cloudinary (servicio especializado en gestión y optimización de imágenes).
* Firebase Storage (integrado al ecosistema Firebase de Google).

1. ¿Qué es un protocolo de comunicación?

Un protocolo de comunicación es un conjunto de reglas y estándares que permiten el intercambio de datos entre dos o más sistemas dentro de una red.

Define aspectos como:

* Formato de los mensajes (cómo se estructuran los datos).
* Secuencia de transmisión (orden en que se envían y reciben los mensajes).
* Mecanismos de control de errores y confirmación (para garantizar que la información llegue completa y correcta).
* Velocidad y sincronización de la comunicación.

En el contexto de aplicaciones web, los protocolos de comunicación son esenciales porque permiten que un cliente (navegador) y un servidor (backend) puedan entenderse e intercambiar información de manera confiable y segura.

1. Explique los protocolos http, https

**HTTP (HyperText Transfer Protocol):**

* Es el protocolo base de la web, utilizado para la comunicación entre un cliente (navegador) y un servidor.
* Funciona bajo el modelo petición–respuesta: el cliente solicita un recurso (página, imagen, API) y el servidor responde.

Características principales:

* Opera en el puerto 80 por defecto.
* Es sin estado (stateless), lo que significa que cada petición es independiente de las anteriores.
* Los datos viajan sin cifrar, lo que los hace vulnerables a ataques como robo de información o manipulación de datos.

**HTTPS (HyperText Transfer Protocol Secure):**

Es la versión segura de HTTP, ya que añade una capa de cifrado TLS/SSL para proteger la comunicación.

Garantiza:

* Confidencialidad: los datos viajan cifrados, evitando que terceros los lean.
* Integridad: asegura que los datos no sean modificados en el camino.
* Autenticidad: confirma que el cliente se comunica con el servidor legítimo (gracias al certificado digital)
* Opera en el puerto 443 por defecto.
* Es esencial en aplicaciones que manejan información sensible, como contraseñas, pagos en línea o datos personales.

1. Que es hosting, investigue los tipos de hosting y haga una tabla comparativa de mínimo 4 proveedores de este servicio, elija uno apropiado para su proyecto

**¿Qué es hosting?**

El hosting es el servicio que permite alojar una aplicación o sitio web en servidores conectados a internet, para que los usuarios puedan acceder desde cualquier lugar. En el caso de GESGAN, el hosting es indispensable porque la plataforma debe estar disponible para ganaderos, administradores y técnicos que consulten datos en tiempo real sobre el ganado.

**Tipos de hosting**

1. **Hosting compartido:** varios proyectos comparten un mismo servidor. Económico, pero limitado en rendimiento.
2. **Hosting VPS (Servidor Privado Virtual):** recursos dedicados dentro de un servidor físico compartido, con mayor control y rendimiento.
3. **Hosting dedicado:** un servidor completo para un solo proyecto, ideal para grandes organizaciones.
4. **Cloud hosting:** se basa en servidores en la nube, permite escalar de acuerdo a la demanda.
5. **PaaS (Platform as a Service):** el proveedor gestiona la infraestructura y el desarrollador solo despliega la aplicación.

### Tabla comparativa de proveedores de hosting

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| proveedor | tipo principal | ventajas | desventajas | precio inicial aprox. |
| **hostinger** | compartido / cloud | económico, fácil de usar, ssl gratis | limitado para apps grandes con muchos usuarios | desde 2.5 usd/mes |
| **digitalocean** | vps / cloud | escalable, control total, documentación clara | requiere conocimientos técnicos de administración | desde 5 usd/mes |
| **aws lightsail** | cloud simplificado | alta disponibilidad, integración con servicios de amazon | costes mayores al escalar, curva de aprendizaje | desde 5 usd/mes |
| **render** | paas(apps web) | deploy directo desde github/gitlab, ssl y cdn incluidos | plan gratuito con límites, menos robusto que aws | desde 7 usd/mes |

1. Que es un servidor de Dominio, escojan un dominio para su proyecto e investiga si está disponible, agregue capturas para comprobar su investigación

### Recomendación para GESGAN

Dado que **GESGAN** es un proyecto académico con potencial de escalar en el futuro:

* Para la **fase inicial/prototipo**, lo ideal es **Render** → porque permite desplegar fácilmente la app desarrollada en Spring Boot, integrarse con GitHub y olvidarse de la configuración del servidor.
* A mediano/largo plazo, si el sistema crece en usuarios y datos, la mejor opción es **DigitalOcean** → porque ofrece escalabilidad, flexibilidad y buen balance entre costo y rendimiento.