**TALLER ARQUITECTURA DE APLICACIONES WEB**

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?

R//: Es la estructura organizativa que define cómo se interconectan y comunican los diferentes componentes de una aplicación web (frontend, backend, base de datos, etc.). Establece patrones de diseño, tecnologías y protocolos para asegurar eficiencia, escalabilidad, seguridad y mantenibilidad.

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

R//: La arquitectura de aplicaciones web es importante porque garantiza que el sistema sea escalable, seguro, mantenible y eficiente, permitiendo que la aplicación soporte un crecimiento en usuarios y funcionalidades sin perder rendimiento, que se puedan realizar actualizaciones y correcciones sin afectar todo el sistema, que los datos estén protegidos mediante buenas prácticas de seguridad y protocolos de comunicación, y que además se optimicen los recursos de infraestructura para asegurar disponibilidad y costos controlados en el despliegue de la solución.

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

R//:

* **Frontend:** interfaz de usuario en navegadores (HTML, CSS, JavaScript, frameworks como React, Angular, Vue).
* **Backend:** lógica de negocio y servicios (Node.js, Django, ASP.NET, Spring Boot).
* **Base de datos:** almacenamiento estructurado o no estructurado (MySQL, MongoDB, PostgreSQL).
* **Servicios externos:** APIs, microservicios, pasarelas de pago.
* **Infraestructura:** hosting, balanceadores de carga, CDN, almacenamiento de archivos (Amazon S3, Azure Blob).
* **Protocolos de comunicación:** HTTP, HTTPS, WebSockets.

1. Mencione las Mejores Prácticas para el desarrollo web

R//: Las arquitecturas modulares y desacopladas, de modo que cada componente (frontend, backend, base de datos) pueda evolucionar sin afectar a los demás. También es recomendable implementar principios de diseño SOLID y patrones de arquitectura que permitan un código limpio y mantenible. El uso de APIs RESTful o GraphQL facilita la comunicación estandarizada entre clientes y servidores, mientras que la integración continua y despliegue continuo (CI/CD) ayuda a automatizar pruebas, compilaciones y despliegues, reduciendo errores humanos y acelerando entregas. En cuanto a infraestructura, es aconsejable trabajar con contenedores (Docker) y orquestadores (Kubernetes) para mejorar la portabilidad y escalabilidad de las aplicaciones. En temas de seguridad, se deben seguir las recomendaciones del OWASP Top 10, incluyendo autenticación robusta, cifrado de datos sensibles, validación de entradas y protección contra ataques como inyección SQL o XSS. Asimismo, es importante implementar pruebas automatizadas (unitarias, de integración y end-to-end) para garantizar la calidad del software, junto con un adecuado control de versiones (Git) que permita trabajo colaborativo. Finalmente, la aplicación debe optimizar su rendimiento mediante el uso de CDN, caching, lazy loading y buenas prácticas en frontend, logrando así una experiencia de usuario más rápida y fluida.

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

R//:

**1. Monolítica:** Toda la aplicación (frontend, backend y lógica de negocio) se desarrolla y despliega como un solo bloque de código.

**Ventajas:** Fácil de iniciar, menor complejidad al comienzo, adecuado para proyectos pequeños.

**Desventajas:** Difícil de escalar, poco flexible para cambios, un error puede afectar a toda la aplicación.

**2. Cliente-Servidor:** El cliente (navegador o aplicación) solicita información y el servidor responde procesando la lógica de negocio y datos.

**Ventajas:** Separación clara de responsabilidades, facilita mantenimiento básico.

**Desventajas:** Puede volverse ineficiente si el servidor central es único, generando dependencia y posibles cuellos de botella.

**3. N-Capas** (3 capas típicas: presentación, lógica de negocio y datos):La aplicación se organiza en capas independientes, cada una con funciones específicas.

**Ventajas:** Modularidad, escalabilidad, facilita mantenimiento y trabajo en equipos grandes.

**Desventajas:** Mayor complejidad de diseño requiere más recursos y puede aumentar la latencia en comunicación entre capas.

**4. Microservicios:** La aplicación se divide en múltiples servicios independientes que se comunican mediante APIs.

**Ventajas:** Escalabilidad, despliegue independiente, resiliencia (fallos en un servicio no afectan a todo el sistema).

**Desventajas:** Complejidad en la gestión, requiere infraestructura más avanzada (Docker, Kubernetes), comunicación más costosa.

**5. Serverless** (Sin servidor dedicado):El backend se implementa como funciones en la nube gestionadas por un proveedor (ej. AWS Lambda, Azure Functions).

**Ventajas:** Bajo costo inicial, autoescalado, sin necesidad de administrar servidores.

**Desventajas:** Dependencia del proveedor, limitaciones de tiempo de ejecución y control, difícil migración entre plataformas.

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, ..

R//:

* Frontend: React, Angular, Vue.js
* Backend: ASP.NET Core, Node.js (Express), Spring Boot, Django
* Base de Datos: SQL Server, MySQL, PostgreSQL, MongoDB
* Almacenamiento de Imágenes: Amazon S3, Google Cloud Storage, Azure Blob Storage

1. Que es un protocolo de comunicación ¿

R//: Es un conjunto de reglas que definen cómo se transmiten y reciben datos entre dos sistemas conectados en red. Ejemplos: HTTP, HTTPS, FTP, SMTP, WebSockets.

1. Explique los protocolos http, https

R//:

* HTTP (HyperText Transfer Protocol): protocolo para transmisión de información en la web, sin cifrado.
* HTTPS (HTTP Secure): versión segura de HTTP, utiliza SSL/TLS para cifrar la comunicación y proteger datos sensibles (contraseñas, tarjetas, etc.).

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

R//: El hosting es el servicio que provee la infraestructura necesaria para almacenar y ejecutar aplicaciones web en Internet, permitiendo que un sitio o aplicación esté disponible de manera pública para los usuarios.

**Tipos de Hosting**

* Compartido: Varios sitios comparten un mismo servidor. Es económico, pero con rendimiento limitado.
* VPS (Servidor Privado Virtual): Divide un servidor físico en entornos virtualizados, con más control y recursos dedicados.
* Dedicado: Un servidor completo asignado a un solo cliente, alto rendimiento, pero costoso.
* Cloud Hosting: Infraestructura escalable en la nube, paga por uso, ideal para proyectos que pueden crecer en usuarios y requieren flexibilidad.

Tabla comparativa

|  |  |  |  |
| --- | --- | --- | --- |
| **Proveedor** | **Tipo de Hosting** | **Ventajas** | **Desventajas** |
| **Hostinger** | Compartido, VPS, Cloud | Económico, fácil de usar, soporte accesible | No tan robusto como los grandes proveedores de nube |
| **AWS (Amazon Web Services)** | Cloud Hosting | Alta escalabilidad, seguridad avanzada, incluye CDN (CloudFront) | Costo variable, curva de aprendizaje alta |
| **Google Cloud** | Cloud Hosting | Excelente rendimiento, integración con IA y big data | Complejo de configurar para principiantes |
| **GoDaddy** | Compartido, VPS, Dedicado | Gestión sencilla, incluye dominios | Menos flexible para proyectos con alto tráfico o escalabilidad |

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

R//: Un servidor de dominio o DNS es el servicio encargado de traducir un nombre de dominio en una dirección IP que los navegadores entienden para acceder al sitio web. En otras palabras, es el "director telefónico" de Internet que conecta la dirección escrita por el usuario con el servidor donde está alojada la aplicación.

Dominio para el proyecto: BlashGames.com (si está disponible)

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.

Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

El contenido generado por IA puede ser incorrecto.