**INSTITUTO TECNOLÓGICO DE MÉRIDA**

**CAMPUS PONIENTE**

**ASIGNATURA:**

Programación Web

**TEMA:**

"Reporte de investigación"

**MAESTRO:**

RODRIGO FIDEL GAXIOLA SOSA

**ALUMNOS:**

Erick Antonio Chan chin

Kevin Daniel Chan Chale

Juan Manuel Chan Vásquez

Moisés Israel Cornejo Bojórquez

Iván Eduardo vivas Chan

**EQUIPO #Punto y coma**

**7SA**

MÉRIDA, YUCATÁN, MÉXICO,

28 DE ENERO DE 2025

Introducción a la Programación Web

El propósito de este documento es explorar los conceptos básicos de la programación web, los componentes principales, las tecnologías más utilizadas y las tendencias actuales que están dando forma al futuro del desarrollo web.

En nuestra actualidad, en la carrera de sistemas en este documento damos un vistazo a la programación web se ha convertido en una de las mejores opciones para desarrollar aplicaciones y servicios en línea.

A lo largo del documento se plasma conceptos de importancia para la programaciin web.

Internet como plataforma global ha cambiado la forma en que las personas interactúan, acceden a la información y realizan negocios, alimentando las necesidades para todo eso mencionado nos ayuda las páginas web.

Junto con programación web donde abarca el conjunto de tecnologías y herramientas utilizadas para crear y mantener sitios web.

Se mencionara parte los conceptos básicos como diseño de interfaces de usuario hasta la gestión de bases de datos y la implementación de lógica empresarial, este campo combina el desarrollo front-end (que es visible para el usuario) y el desarrollo back-end (que funciona detrás de escena).

Se mencionara en el documento herramientas como HTML, CSS y JavaScript son esenciales para crear interfaces interactivas, mientras que tecnologías como PHP, Python, Ruby o Node.js permiten administrar servidores y conectarse a bases de datos.

**1. ¿Qué es la programación web?**

La programación web es el proceso de crear y desarrollar aplicaciones o sitios web que se ejecutan en navegadores. Implica el uso de lenguajes y tecnologías como HTML, CSS, JavaScript, y lenguajes de servidor como PHP, Python, o Java. Se enfoca en construir interfaces de usuario, gestionar la interacción entre usuarios y servidores, y garantizar la funcionalidad y el diseño adecuados de las aplicaciones web.

**2. ¿Cuál es la diferencia entre el lado del cliente (frontend) y el lado del servidor (backend) en el desarrollo web?**

* **Frontend (Lado del Cliente):** Es la parte de la aplicación web que interactúa directamente con el usuario. Incluye el diseño y la experiencia de usuario (UI/UX), y utiliza tecnologías como HTML, CSS, y JavaScript para mostrar contenido en el navegador.
* **Backend (Lado del Servidor):** Es la parte encargada de manejar la lógica del negocio, procesar solicitudes del cliente y gestionar bases de datos. Funciona detrás de escena utilizando lenguajes como PHP, Python, Java, o Node.js, y se conecta con bases de datos como MySQL o MongoDB.

**3. ¿Qué es HTML y para qué se utiliza en la programación web?**

* HTML (HyperText Markup Language) es un lenguaje de marcado utilizado para estructurar y organizar el contenido de las páginas web. Define la semántica y disposición de los elementos como encabezados, párrafos, imágenes, enlaces, tablas, y formularios. Es la base para construir cualquier página web.

**4. Menciona algunos ejemplos de etiquetas HTML que se utilizan comúnmente:**

* **Encabezados:** <h1>, <h2>, <h3>...
* **Párrafos:** <p>
* **Imágenes:** <img>
* **Enlaces:** <a>
* **Listas:** <ul> (no ordenada), <ol> (ordenada), <li> (elemento de lista).
* **Tablas:** <table>, <tr>, <td>, <th>
* **Formularios:** <form>, <input>, <button>
* **Divisiones:** <div> y <span>

**5. ¿Qué es CSS y cómo se relaciona con HTML?**

* CSS (Cascading Style Sheets) es un lenguaje utilizado para describir la apariencia y el diseño visual de una página web escrita en HTML. Permite definir estilos como colores, fuentes, tamaños, espaciados, y posiciones de los elementos en una página web, separando la estructura (HTML) de la presentación.

**6. ¿Cuál es el propósito de JavaScript en el desarrollo web?**

JavaScript es un lenguaje de programación utilizado para añadir interactividad y dinamismo a las páginas web. Permite realizar tareas como:

* Validar formularios.
* Crear animaciones.
* Actualizar contenido de la página sin recargarla (usando AJAX).
* Manejar eventos del usuario, como clics o desplazamiento.  
   En general, mejora la experiencia del usuario en el navegador.

**7. ¿Cuál es la diferencia entre un sitio web estático y uno dinámico?**

* **Sitio web estático:**
  + Contiene páginas con contenido fijo que no cambia con el tiempo o con la interacción del usuario.
  + Es construido con HTML y CSS, sin lógica de servidor.
  + Ejemplo: Portafolios simples o sitios de información básica.
* **Sitio web dinámico:**
  + Contiene contenido que puede cambiar en función de la interacción del usuario, datos en tiempo real, o procesos del servidor.
  + Utiliza backend, bases de datos y lenguajes de servidor.
  + Ejemplo: Redes sociales, tiendas en línea o aplicaciones web.

.

8. ¿Qué es un navegador web y cómo interpreta HTML, CSS y JavaScript?

Un navegador web es una aplicación de software que permite a los usuarios acceder, visualizar e interactuar con contenido en la web, como páginas web, imágenes, videos

El navegador construye un árbol estructurado llamado DOM (Document Object Model) basado en el HTML.

El navegador aplica las reglas CSS al DOM para darle estilo a los elementos HTML.

El navegador ejecuta el JavaScript en un motor de JavaScript (por ejemplo, V8 en Chrome).

9. ¿Qué es el diseño responsive en el contexto de la programación web?

Es el responsable de que un sitio web se adapte automáticamente al tamaño y orientación de la pantalla del dispositivo, proporcionando una experiencia de usuario óptima sin importar el dispositivo que se utilice.

10. ¿Cuál es el objetivo de la optimización para motores de búsqueda (SEO) en un sitio web?

Se trata de posicionar un sitio web, sino de brindar una experiencia valiosa a los usuarios mientras se obtiene un mejor rendimiento en los motores de búsqueda.

Un sitio web bien optimizado puede aparecer por encima de competidores directos en los resultados de búsqueda, obteniendo una ventaja competitiva.

11. Describe brevemente el proceso de solicitud y respuesta en el modelo cliente-servidor.

l cliente (navegador) envía una solicitud GET al servidor:

GET /index.html HTTP/1.1

El servidor procesa la solicitud y devuelve una respuesta:

Código de estado: 200 OK

Contenido: el archivo index.html.

El navegador interpreta el HTML, solicita recursos adicionales (CSS, imágenes, JavaScript) y renderiza la página.

12. ¿Qué es una URL y cuál es su estructura básica?

es la dirección que se utiliza para acceder a un recurso específico en Internet

https://www.ejemplo.com:443/productos/categoria.html?id=123&categoria=ropa#ofertas

Protocolo: https

Host (dominio): www.ejemplo.com

Puerto: 443 (implícito para HTTPS)

Ruta: /productos/categoria.html

Query: id=123&categoria=ropa

Fragmento: #ofertas

13. Menciona algunos de los lenguajes de programación utilizados en el lado del servidor.

Php. Java, Python, Javascript

14. ¿Qué es un framework en el desarrollo web y cómo puede beneficiar a los desarrolladores?

Es un conjunto de herramientas, bibliotecas y buenas prácticas que facilita el desarrollo de aplicaciones web al proporcionar una estructura base y funciones predefinidas para tareas comunes.

15. Explica el concepto de "caja de modelo de caja" (box model) en CSS.

El modelo de caja es un concepto fundamental en CSS que describe cómo los elementos HTML se estructuran y se renderizan en la página web. Cada elemento se considera como una "caja"

1. ¿Qué son las cookies y para qué se utilizan en la programación web?  
   Las cookies son pequeños archivos de texto almacenados en el navegador del usuario por los sitios web que visita. Se utilizan para:

* *Recordar información del usuario:* como inicios de sesión, preferencias o carritos de compra.
* *Rastrear actividad:* para análisis web o personalización de contenido.
* *Mantener sesiones activas:* evitando que el usuario tenga que iniciar sesión repetidamente.

Aunque útiles, deben manejarse con cuidado por razones de privacidad y seguridad.

1. ¿Cuál es la importancia de la seguridad en el desarrollo web?  
   La seguridad en el desarrollo web protege los datos de los usuarios y la funcionalidad del sitio contra amenazas como:

* Ataques de inyección SQL.
* Cross-Site Scripting (XSS).
* Cross-Site Request Forgery (CSRF).
* Robo de datos o accesos no autorizados.

Una buena seguridad genera confianza en los usuarios y asegura el cumplimiento de leyes y normativas.

1. ¿Qué es control de versiones y cómo puede ayudar en el desarrollo web colaborativo?

El control de versiones es un sistema que registra los cambios realizados en un proyecto, permitiendo:

* Guardar múltiples versiones del código.
* Colaborar en equipo sin sobrescribir el trabajo de otros.
* Retroceder a versiones anteriores si ocurre un error.

Herramientas como *Git* son comunes y facilitan la colaboración en equipos distribuidos.

1. ¿Qué es un dominio y cómo se relaciona con una dirección IP?  
   Un dominio es un nombre legible por humanos que identifica un sitio web, como [www.ejemplo.com](http://www.ejemplo.com/). Está relacionado con una dirección IP, que es el identificador numérico único que usan los servidores para comunicarse en Internet. Los sistemas DNS (Domain Name System) traducen los dominios a direcciones IP.
2. Describe brevemente cómo funciona el protocolo HTTP en la navegación web.

HTTP (HyperText Transfer Protocol) es el protocolo que permite la comunicación entre el cliente (navegador) y el servidor:

El cliente envía una solicitud HTTP al servidor.

El servidor procesa la solicitud y responde con el contenido solicitado (como una página HTML) o un código de estado (por ejemplo, 404 si no se encuentra la página).  
HTTP es sin estado, lo que significa que cada solicitud es independiente.

21. ¿Cuál es la evolución histórica que llevó a la creación de las aplicaciones web modernas?

* **Años 90**: Páginas estáticas con HTML y CSS.
* **2000s**: Aplicaciones dinámicas con PHP, JavaScript y bases de datos.
* **2010s**: Web 2.0 con interactividad (AJAX, frameworks como React, Angular).
* **Actualidad**: Aplicaciones web progresivas (PWA), inteligencia artificial y computación en la nube

22. ¿Qué diferencias clave existen entre las aplicaciones de escritorio y las aplicaciones web en términos de accesibilidad y distribución?

* **Accesibilidad**: Las apps web pueden ejecutarse en cualquier dispositivo con navegador, mientras que las de escritorio requieren instalación.
* **Distribución**: Las apps web no necesitan descargas ni actualizaciones manuales; en cambio, las de escritorio requieren instalación en cada dispositivo.

23. ¿Puedes mencionar ejemplos de tecnologías o servicios que impulsaron el avance de las aplicaciones web? Arquitectura de las aplicaciones web:

* **AJAX**: Permite actualizaciones dinámicas sin recargar la página.
* **Cloud Computing**: Facilita el almacenamiento y procesamiento de datos online.
* **Frameworks JS (React, Angular, Vue)**: Mejoran la experiencia del usuario.
* **PWA (Progressive Web Apps)**: Apps web con funciones similares a las nativas.

24. Explica la diferencia entre la arquitectura cliente-servidor y la arquitectura peer-to-peer en el contexto de las aplicaciones web.

* **Cliente-servidor**: Un cliente (navegador) solicita recursos a un servidor central que responde con datos. Ejemplo: una tienda online.
* **Peer-to-peer (P2P)**: Cada nodo actúa como cliente y servidor al mismo tiempo, intercambiando información directamente. Ejemplo: Torrents.

25. Describe brevemente la arquitectura de tres capas utilizada en muchas aplicaciones web. ¿Cuáles son esas capas y cuál es su función?

* **Capa de presentación (Front-end)**: Interfaz de usuario (HTML, CSS, JS).
* **Capa lógica de negocio (Back-end)**: Procesamiento de datos y lógica de la aplicación (Node.js, Python, Java).
* **Capa de datos (Base de datos)**: Almacenamiento y gestión de información (MySQL, MongoDB).

26. ¿Cómo se comunican el cliente y el servidor en una arquitectura web típica de tres capas?

1. El cliente envía una solicitud HTTP/HTTPS desde el navegador a la capa de presentación.
2. La capa de presentación pasa la solicitud a la capa de lógica de negocio.
3. La capa de lógica de negocio interactúa con la capa de datos para obtener o guardar información.
4. La respuesta se regresa al cliente a través de la capa de presentación.

**Tecnologías para el desarrollo de aplicaciones web**

27. ¿Qué es HTML5 y qué mejoras introdujo con respecto a versiones anteriores de HTML?

HTML5 es la quinta versión principal del lenguaje de marcado HTML (HyperText Markup Language), que se utiliza para estructurar y presentar contenido en la web. Fue oficialmente publicado por el World Wide Web Consortium (W3C) en octubre de 2014, aunque muchas de sus características ya estaban siendo implementadas por los navegadores antes de su lanzamiento oficial.

### Mejoras introducidas por HTML5:

1. **Nuevos elementos semánticos**:
   1. HTML5 introdujo elementos semánticos como <header>, <footer>, <article>, <section>, <nav>, y <aside>, que ayudan a definir mejor la estructura y el significado del contenido, facilitando la accesibilidad y el SEO.
2. **Soporte multimedia nativo**:
   1. HTML5 incluye elementos como <audio> y <video> para integrar contenido multimedia directamente en las páginas web sin necesidad de plugins externos como Flash.
3. **Canvas y gráficos vectoriales**:
   1. El elemento <canvas> permite la creación de gráficos en 2D y animaciones directamente en el navegador, lo que es útil para juegos y visualizaciones dinámicas.
   2. También se introdujo el soporte para gráficos vectoriales escalables (SVG) integrados en HTML.
4. **Formularios mejorados**:
   1. HTML5 añadió nuevos tipos de entrada (<input>) como email, date, time, number, range, url, y search, que mejoran la experiencia del usuario y la validación de datos.
   2. También se incluyeron atributos como placeholder, required, autofocus, y pattern para facilitar la creación de formularios más interactivos y accesibles.
5. **Almacenamiento local**:
   1. HTML5 introdujo localStorage y sessionStorage, que permiten almacenar datos en el navegador del usuario de manera persistente o durante la sesión, respectivamente, sin necesidad de cookies.
6. **APIs nuevas**:
   1. **Geolocalización**: Permite a las aplicaciones web acceder a la ubicación geográfica del usuario.
   2. **Drag and Drop**: Facilita la implementación de funcionalidades de arrastrar y soltar.
   3. **Web Workers**: Permite ejecutar scripts en segundo plano, mejorando el rendimiento de aplicaciones web complejas.
   4. **WebSockets**: Facilita la comunicación bidireccional en tiempo real entre el cliente y el servidor.
   5. **IndexedDB**: Una base de datos NoSQL en el navegador para almacenar grandes cantidades de datos estructurados.
7. **Mejoras en la accesibilidad**:
   1. HTML5 incluye mejoras en la accesibilidad, como atributos ARIA (Accessible Rich Internet Applications) integrados, que ayudan a los usuarios con discapacidades a interactuar mejor con las páginas web.
8. **Compatibilidad con dispositivos móviles**:
   1. HTML5 fue diseñado teniendo en cuenta la creciente importancia de los dispositivos móviles, ofreciendo una mejor experiencia de usuario en pantallas pequeñas y táctiles.
9. **Doctype simplificado**:
   1. HTML5 simplificó la declaración del tipo de documento a <!DOCTYPE html>, lo que hace que sea más fácil de recordar y usar.
10. **Mejoras en el rendimiento**:
    1. HTML5 optimizó el rendimiento de las páginas web, reduciendo la necesidad de recursos externos y mejorando la velocidad de carga.

28. Explica la importancia de CSS en la presentación de una página web**.**

CSS (Cascading Style Sheets) es un lenguaje de hojas de estilo utilizado para controlar la presentación y el diseño de una página web. Su importancia radica en varios aspectos clave que mejoran la experiencia del usuario, la mantenibilidad del código y la eficiencia en el desarrollo web.

CSS es fundamental para la presentación de una página web, ya que no solo mejora la apariencia visual, sino que también contribuye a la usabilidad, accesibilidad, rendimiento y mantenibilidad del sitio. Su capacidad para crear diseños responsivos y su flexibilidad en la aplicación de estilos lo convierten en una herramienta indispensable en el desarrollo web moderno.

29. ¿Cuál es el propósito principal de JavaScript en el desarrollo web? ¿Puedes mencionar algunas de sus características clave?

El propósito principal de JavaScript en el desarrollo web es añadir interactividad, dinamismo y funcionalidad a las páginas web. Mientras que HTML se encarga de la estructura del contenido y CSS de su presentación, JavaScript permite crear experiencias de usuario más ricas y complejas al hacer que las páginas web respondan a las acciones del usuario, actualicen contenido dinámicamente y se comuniquen con servidores en tiempo real.

### Propósito principal de JavaScript:

1. **Interactividad**:
   1. JavaScript permite que las páginas web respondan a eventos del usuario, como clics, movimientos del ratón, entradas de teclado, y más. Esto incluye validaciones de formularios, menús desplegables, sliders, y otros elementos interactivos.
2. **Manipulación del DOM**:
   1. JavaScript puede manipular el Document Object Model (DOM), lo que permite cambiar el contenido, la estructura y el estilo de una página web dinámicamente sin necesidad de recargar la página.
3. **Comunicación Asíncrona**:
   1. Con tecnologías como AJAX (Asynchronous JavaScript and XML) y Fetch API, JavaScript permite realizar solicitudes asíncronas al servidor, lo que posibilita la actualización de partes específicas de una página sin recargarla completamente.
4. **Animaciones y Efectos Visuales**:
   1. JavaScript puede trabajar junto con CSS para crear animaciones complejas y efectos visuales que mejoran la experiencia del usuario.
5. **Almacenamiento en el Cliente**:
   1. JavaScript permite almacenar datos en el lado del cliente usando tecnologías como localStorage, sessionStorage, y IndexedDB, lo que es útil para aplicaciones web que necesitan guardar información temporalmente o de manera persistente.
6. **APIs del Navegador**:
   1. JavaScript puede interactuar con diversas APIs del navegador, como la API de Geolocalización, la API de Canvas para gráficos, la API de WebSockets para comunicación en tiempo real, y muchas más.

**30. Dentro del desarrollo del lado del servidor, nombra al menos dos lenguajes de programación comúnmente utilizados y describe en qué contextos son preferibles.**

PHP y Java

* **PHP**: Es preferible para proyectos pequeños o medianos, desarrollo rápido, bajo costo y aplicaciones que no requieren alta complejidad o escalabilidad extrema.
* **Java**: Es preferible para aplicaciones empresariales, de gran escala, con altos requisitos de rendimiento, seguridad y escalabilidad.

31. ¿Cuáles son los principales youtubers/tiktores que enseñan sobre el desarrollo web (3 en español y 3 en inglés)?

**En español**

1. YouTube: Fazt

2. YouTube: Dorian Desings

3. TikTok: Midudev (Miguel Ángel Durán)

**En ingles**

* YouTube: Traversy Media
* YouTube: The Net Ninja
* TikTok: Codecademy

**Planificación de aplicaciones web**

32. **¿Qué implica la planificación de aplicaciones web en términos de diseño y desarrollo?**

La planificación de aplicaciones web es una fase crítica que define el éxito del proyecto. Implica una serie de pasos y consideraciones tanto en el diseño como en el desarrollo para garantizar que la aplicación sea funcional, escalable, segura y satisfaga las necesidades del usuario. A continuación, te explico los aspectos clave de esta planificación:

### **1. Análisis de Requisitos**

* **Objetivos del proyecto**: Definir el propósito de la aplicación, los problemas que resuelve y los objetivos empresariales.
* **Requisitos funcionales**: Detallar las características y funcionalidades que debe tener la aplicación (por ejemplo, autenticación de usuarios, procesamiento de pagos, etc.).
* **Requisitos no funcionales**: Considerar aspectos como rendimiento, escalabilidad, seguridad y compatibilidad con diferentes dispositivos y navegadores.
* **Público objetivo**: Identificar a los usuarios finales y sus necesidades para diseñar una experiencia adecuada.

### **2. Diseño de la Arquitectura**

* **Arquitectura del sistema**: Decidir si la aplicación será monolítica, basada en microservicios o serverless, según los requisitos de escalabilidad y mantenimiento.
* **Tecnologías a utilizar**: Elegir el stack tecnológico (frontend, backend, base de datos, servidores, etc.).
* **API y integraciones**: Planificar si la aplicación necesitará integrarse con servicios externos o APIs de terceros.
* **Flujo de datos**: Diseñar cómo se moverán los datos entre el frontend, el backend y la base de datos.

### **3. Diseño de la Interfaz de Usuario (UI)**

* **Wireframes y prototipos**: Crear esquemas básicos y prototipos interactivos para visualizar la estructura y el flujo de la aplicación.
* **Experiencia de usuario (UX)**: Asegurar que la aplicación sea intuitiva, fácil de usar y accesible para todos los usuarios.
* **Diseño visual**: Definir la paleta de colores, tipografía, iconos y otros elementos visuales que reflejen la identidad de la marca.

### **4. Planificación del Desarrollo**

* **Metodología de trabajo**: Elegir una metodología ágil (como Scrum o Kanban) para gestionar el desarrollo de manera iterativa y flexible.
* **División de tareas**: Asignar responsabilidades al equipo (frontend, backend, diseño, testing, etc.).
* **Cronograma**: Establecer plazos realistas para cada fase del proyecto (diseño, desarrollo, pruebas, lanzamiento).

### **5. Desarrollo del Frontend**

* **Maquetación**: Implementar el diseño visual usando HTML, CSS y frameworks como Bootstrap o Tailwind.
* **Interactividad**: Añadir funcionalidades dinámicas con JavaScript y frameworks/librerías como React, Angular o Vue.js.
* **Responsividad**: Asegurar que la aplicación sea compatible con diferentes dispositivos y tamaños de pantalla.

### **6. Desarrollo del Backend**

* **Lógica del servidor**: Implementar la lógica de negocio y las funcionalidades del servidor usando lenguajes como PHP, Python, Java, Node.js, etc.
* **Base de datos**: Diseñar y configurar la base de datos (SQL o NoSQL) para almacenar y gestionar la información.
* **APIs**: Crear APIs RESTful o GraphQL para la comunicación entre el frontend y el backend.

### **7. Pruebas y Control de Calidad**

* **Pruebas unitarias**: Verificar el funcionamiento de cada componente por separado.
* **Pruebas de integración**: Asegurar que los diferentes módulos de la aplicación funcionen correctamente juntos.
* **Pruebas de usabilidad**: Validar que la aplicación sea fácil de usar y cumpla con las expectativas del usuario.
* **Pruebas de rendimiento**: Evaluar la velocidad, escalabilidad y estabilidad de la aplicación bajo diferentes cargas.

### **8. Despliegue y Lanzamiento**

* **Configuración del servidor**: Elegir un proveedor de hosting (como AWS, Google Cloud, o Heroku) y configurar el entorno de producción.
* **Despliegue continuo**: Implementar pipelines de CI/CD (Integración Continua/Despliegue Continuo) para automatizar el proceso de despliegue.
* **Monitoreo**: Configurar herramientas para monitorear el rendimiento y detectar errores en tiempo real.

### **9. Mantenimiento y Actualizaciones**

* **Soporte técnico**: Resolver problemas y bugs reportados por los usuarios.
* **Actualizaciones**: Implementar nuevas funcionalidades y mejoras basadas en feedback y cambios en los requisitos.
* **Escalabilidad**: Asegurar que la aplicación pueda crecer y manejar un mayor tráfico o carga de trabajo.

### **10. Consideraciones Adicionales**

* **Seguridad**: Implementar medidas como HTTPS, validación de entradas, protección contra inyecciones SQL y autenticación segura.
* **SEO**: Optimizar la aplicación para motores de búsqueda si es un sitio web público.
* **Accesibilidad**: Asegurar que la aplicación cumpla con estándares de accesibilidad (WCAG) para usuarios con discapacidades.

33. ¿Qué es una metodología ágil y cuáles son las más populares?

Una **metodología ágil** es un enfoque de gestión de proyectos que se centra en la entrega incremental, la colaboración continua y la adaptación a los cambios. A diferencia de los métodos tradicionales (como el modelo en cascada), las metodologías ágiles buscan dividir el trabajo en pequeños incrementos o iteraciones, permitiendo que los equipos respondan rápidamente a los cambios en los requisitos o en el entorno del proyecto.

Metodologías agile más populares:

1. **Scrum**
   1. **Descripción**: Scrum es una de las metodologías ágiles más utilizadas. Se basa en iteraciones cortas llamadas **sprints** (generalmente de 2 a 4 semanas), durante las cuales el equipo trabaja en un conjunto de tareas definidas.
2. **Kanban**
   1. **Descripción**: Kanban se centra en la visualización del flujo de trabajo y la limitación del trabajo en progreso (WIP). Utiliza un tablero con columnas (como "Por hacer", "En progreso" y "Terminado") para gestionar tareas.
3. **Extreme Programming (XP)**
   1. **Descripción**: XP es una metodología ágil que se enfoca en la calidad del software y la satisfacción del cliente. Promueve prácticas como la programación en parejas, el desarrollo guiado por pruebas (TDD) y la integración continua.
4. **Lean**
   1. **Descripción**: Lean se basa en los principios de la manufactura esbelta (Toyota Production System) y se enfoca en eliminar el desperdicio, maximizar el valor y optimizar el flujo de trabajo.

34. ¿Por qué es importante considerar el público objetivo al planificar una aplicación web?

Considerar al público objetivo al planificar una aplicación web es crucial porque permite que el producto se adapte mejor a las necesidades y expectativas de quienes lo van a utilizar. Al tener en cuenta a los usuarios, se pueden tomar decisiones más acertadas sobre el diseño, funcionalidades, y experiencia de usuario.

entender quiénes usarán tu aplicación te ayuda a construir una experiencia más relevante, efectiva y atractiva, lo cual se traduce en mayor satisfacción

35. Menciona algunos pasos esenciales en la planificación de la arquitectura de una aplicación web.

* **Definir los requisitos funcionales y no funcionales**: Antes de comenzar con la arquitectura, es fundamental entender las necesidades del negocio y del usuario. Esto incluye tanto los requisitos funcionales (qué debe hacer la aplicación) como los no funcionales (rendimiento, seguridad, escalabilidad, etc.).
* **Elegir el modelo de arquitectura adecuado**: Dependiendo de la complejidad del proyecto, se debe decidir entre diferentes tipos de arquitectura, como arquitectura monolítica, basada en microservicios, arquitectura de servidor sin estado (serverless), entre otras.
* **Seleccionar tecnologías y herramientas**: Elegir las tecnologías de desarrollo, bases de datos, frameworks y librerías que mejor se adapten a los requisitos del proyecto y al perfil del equipo de desarrollo. Esto incluye la elección de lenguajes de programación, bases de datos (SQL o NoSQL), servidores y herramientas de integración continua.
* **Definir la estructura de la base de datos**: Planificar cómo se organizarán los datos, qué tipo de bases de datos se utilizarán, y cómo se manejarán las relaciones entre los datos. Esto también implica pensar en la escalabilidad de la base de datos, la seguridad de los datos y cómo se realizará el respaldo.
* **Establecer la gestión de la seguridad**: Definir cómo se manejarán aspectos de seguridad, como la autenticación, autorización, encriptación de datos y protección contra ataques comunes (como inyecciones SQL, XSS, CSRF, etc.).
* **Diseñar la arquitectura de la red y comunicación**: Decidir cómo se manejarán las comunicaciones entre los diferentes componentes de la aplicación (servidores, bases de datos, APIs). Esto incluye la elección de protocolos como HTTP/HTTPS, WebSockets, REST, GraphQL, y si será una arquitectura orientada a eventos o de mensajería.
* **Planificar la escalabilidad**: Prever cómo la aplicación se manejará a medida que crezca en términos de tráfico y datos. Esto podría implicar el uso de balanceadores de carga, clusters, caché, y la posibilidad de escalar de forma horizontal o vertical.
* **Diseñar la infraestructura de despliegue**: Determinar cómo y dónde se desplegará la aplicación, considerando opciones como servidores locales, servicios en la nube (AWS, Google Cloud, Azure), contenedores (Docker), o plataformas de orquestación (Kubernetes).
* **Implementar la monitorización y mantenimiento**: Planificar cómo se realizará el monitoreo de la aplicación, la recopilación de métricas, la detección de errores y la implementación de registros (logs) para una gestión proactiva de la aplicación a lo largo del tiempo.
* **Establecer un plan de pruebas**: Definir las pruebas que se van a realizar (unitarias, integración, funcionales, de rendimiento) y cómo se integrarán en el ciclo de desarrollo. La automatización de pruebas es esencial para garantizar que la aplicación se mantenga confiable y eficiente.
* **Definir un plan de gestión de versiones y CI/CD**: Implementar una estrategia de control de versiones y establecer un pipeline de integración continua y entrega continua (CI/CD) para asegurar un flujo de trabajo ágil y sin interrupciones.

36. ¿Cuál es la importancia de establecer una estructura de carpetas y archivos bien organizada en el desarrollo de aplicaciones web?

### **1. Facilita la colaboración en equipo:**

Una estructura clara ayuda a los miembros del equipo a encontrar rápidamente los archivos que necesitan, evitando confusión y reduciendo la posibilidad de cometer errores al trabajar en diferentes partes del proyecto. Es especialmente útil en equipos grandes o cuando hay nuevos desarrolladores que se unen al proyecto.

### **2. Mejora la escalabilidad del proyecto:**

A medida que una aplicación web crece, tener una estructura bien organizada facilita añadir nuevas funcionalidades sin que el proyecto se vuelva caótico. Una jerarquía lógica permite que el código se mantenga modular, facilitando la extensión o modificación de componentes sin afectar a todo el sistema.

### **3. Fomenta la reutilización de código:**

Al seguir una estructura consistente, es más fácil identificar y extraer componentes reutilizables. Por ejemplo, si tienes carpetas separadas para controladores, vistas, y modelos (en una arquitectura MVC), puedes reutilizar esos componentes en diferentes partes de la aplicación sin duplicar código.

### **4. Facilita la depuración y mantenimiento:**

Una estructura bien definida permite que los errores sean más fáciles de identificar y corregir. Si todo está en su lugar, puedes rápidamente ubicar los archivos correspondientes a una funcionalidad o módulo específico y solucionarlo de manera eficiente.

### **5. Sigue las mejores prácticas y convenciones:**

Muchas veces, seguir una estructura de carpetas comúnmente aceptada (como las utilizadas en frameworks populares como React, Angular o Django) facilita el aprendizaje y la integración de nuevos desarrolladores al proyecto, pues estarán familiarizados con la disposición de los archivos.

### **6. Optimiza el control de versiones:**

Una organización adecuada de archivos también facilita el uso de sistemas de control de versiones (como Git). Si tienes una estructura coherente, es más sencillo realizar cambios sin crear conflictos, ya que cada componente está claramente definido y ubicado en su lugar correspondiente.

### **7. Mejora la accesibilidad y documentación:**

Una estructura bien organizada también facilita la documentación del proyecto. Si los archivos y las carpetas están lógicamente distribuidos, se hace más fácil entender la arquitectura del sistema, lo que ayuda tanto a la documentación interna como a los posibles usuarios externos que necesiten interactuar con el código.

Conclusión:

En conclusión la programación web es una materia de suma importancia en nuestra situación actual, permitiéndonos la creación de aplicaciones y sitios web necesarios para comunicarse, comerciar e interactuar en línea.

Junto con las tecnologías front-end y back-end, además de utilizar marcos y herramientas modernos, los programadores pueden crear soluciones muy buenas para un futuro.

Al igual que la evolución constante del campo, para dar una buena experiencia al usuario y la capacidad de adaptarse a varios dispositivos, hace que la programación web siga siendo un campo interesante. Por último, llegar a dominar las habilidades de programación web nos abre las puertas a muchas oportunidades profesionales, igual aporta a la innovación y el progreso en la era digital.

REFERENCIAS: