TECNICATURA SUPERIOR EN

Telecomunicaciones

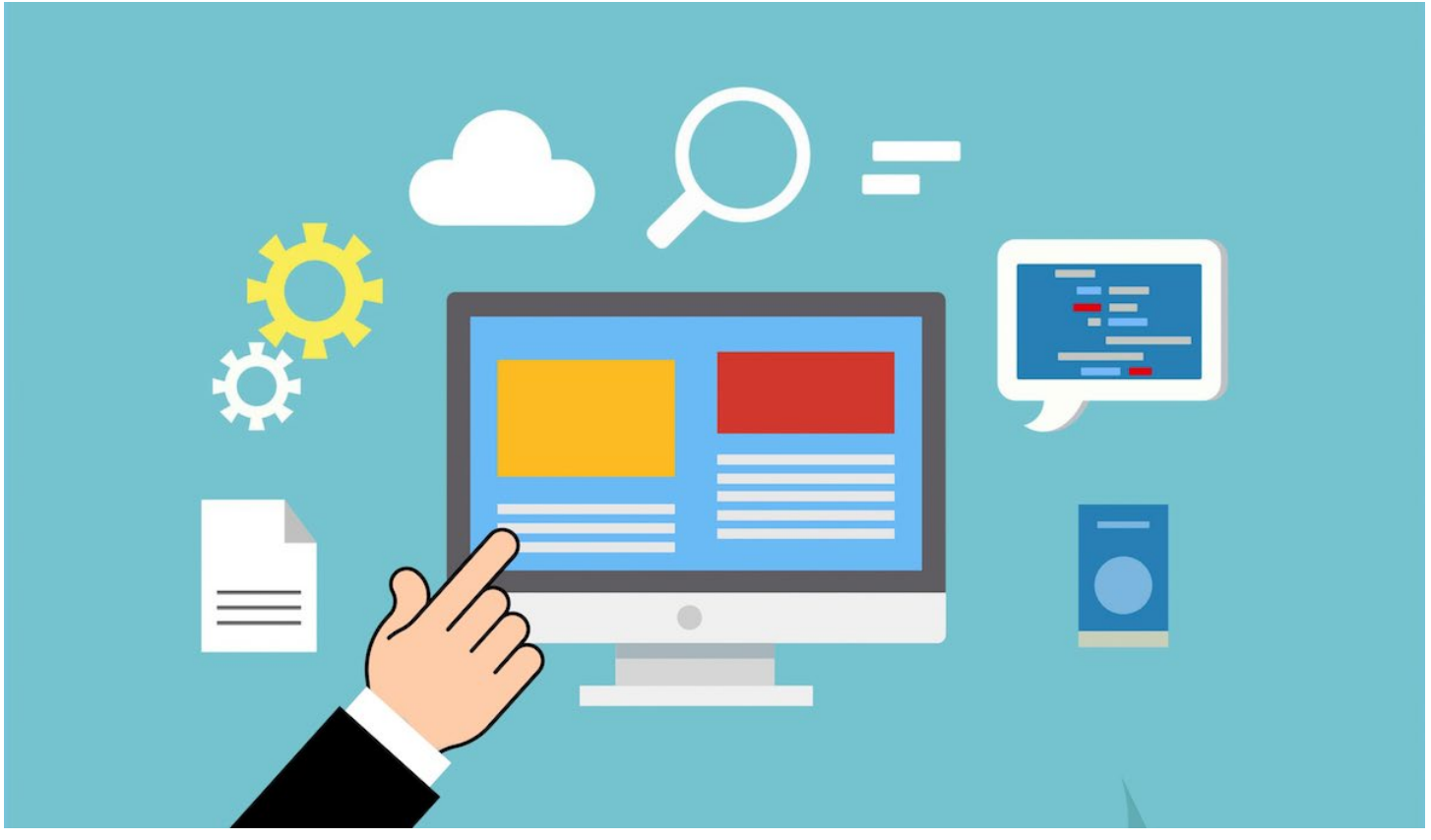
Proyecto Integrador I

Desarrollo Web Básico

ISPC
INSTITUTO SUPERIOR POLITÉCNICO CÓRDOBA

Dirección General de EDUCACIÓN TÉCNICA Y FORMACIÓN PROFESIONAL
Ministerio de EDUCACIÓN
CÓRDOBA entre todos

Desarrollo Web Básico



Desarrollo Web Básico

El desarrollo web es un campo muy amplio y tiene muchas tecnologías involucradas, dependiendo de los requisitos del proyecto.
El **stack tecnológico** es el conjunto de herramientas, lenguajes y frameworks que eliges para construir una aplicación web. Generalmente, un stack se divide en tres partes:

1. Frontend
2. Backend
3. Infraestructura (DevOps)

Desarrollo Web Básico

1. **Frontend (lado del cliente):**

- Aquí se trabaja en la interfaz que los usuarios ven y con la que interactúan. Las tecnologías más comunes son:

- **HTML:** Estructura básica de la web.
- **CSS:** Para el diseño y estilo.
- **JavaScript:** Para la interactividad y la lógica en el lado del cliente.
- **Frameworks/Librerías de JavaScript para el front:**
  - **React:** Librería para construir interfaces interactivas.
  - **Vue.js:** Framework progresivo, fácil de integrar en proyectos existentes.
  - **Angular:** Framework robusto y completo desarrollado por Google para aplicaciones más grandes y estructuradas.

Desarrollo Web Básico

2. Backend (lado del servidor):

- Es la parte que maneja la lógica de negocio, la autenticación, la interacción con la base de datos, etc. Algunas tecnologías populares:
- **Lenguajes y Frameworks:**
  - **Node.js con Express:** JavaScript también en el backend, ideal para aplicaciones en tiempo real.
  - **Python con Django/Flask:** Django es un framework completo, mientras que Flask es más ligero y flexible.
  - **Ruby on Rails:** Ideal para desarrollo rápido, con convenciones claras.
  - **PHP con Laravel:** Laravel es un framework moderno y popular para PHP.
  - **Java con Spring:** Para aplicaciones empresariales robustas.
  - **Go:** Lenguaje ligero y eficiente, popular para microservicios.
- **Bases de Datos:**
  - **Relacionales:** MySQL, PostgreSQL, SQL Server (ideal para datos estructurados).
  - **NoSQL:** MongoDB, Cassandra, Redis (más flexibles para datos no estructurados).

Desarrollo Web Básico

3. Infraestructura/DevOps:

- Aquí entra el despliegue, escalabilidad, automatización y gestión de servidores:
- **Servidores/Almacenamiento:**
  - **AWS, Google Cloud, Azure:** Plataformas de nube más populares, ofrecen infraestructura escalable.
  - **Docker/Kubernetes:** Para contenerización y orquestación de microservicios, asegurando que el código se ejecute de la misma manera en cualquier entorno.
  - **CI/CD Pipelines (Integración Continua/Despliegue Continuo):** Jenkins, GitLab CI, CircleCI para automatizar pruebas y despliegues.

Monitorización y Seguridad: Herramientas como Prometheus, Grafana, o servicios como Datadog para monitoreo y observabilidad, y medidas de seguridad como el uso de certificados SSL, firewalls y controles de acceso.

Desarrollo Web Básico

Stack Tecnológico Común

- Un ejemplo de stack completo es el **MERN Stack**, que incluye:
- **MongoDB** (base de datos NoSQL).
- **Express** (framework backend).
- **React** (librería frontend).
- **Node.js** (entorno para ejecutar JavaScript en el backend).

Otras combinaciones comunes:

- **LAMP:** Linux, Apache, MySQL, PHP.
- **MEAN:** MongoDB, Express, Angular, Node.js.
- **JAMstack:** JavaScript, APIs y Markup (HTML estático servido desde una CDN, usando APIs para la lógica dinámica).

Desarrollo Web Básico

El tiempo para aprender **frontend** y avanzar de un nivel junior a senior depende de varios factores como la dedicación, experiencia previa en programación, y la complejidad de los proyectos en los que trabajes. A continuación, una idea aproximada de los tiempos en función del camino común para muchos desarrolladores.

1. Aprender Frontend a Nivel Junior:

- Para llegar a ser un desarrollador frontend junior, generalmente se logra en un plazo de **6 meses a 1 año**, si sigues una ruta de aprendizaje intensivo y prácticas constantemente.
- **Ruta de aprendizaje**:
- **Mes 1-3**: Fundamentos sólidos de **HTML, CSS y JavaScript**.
  - Aprende a crear páginas web estáticas.
  - Entiende conceptos básicos como el modelo de caja, flexbox, y responsive design.
  - Practica la manipulación del DOM y eventos en JavaScript.

Desarrollo Web Básico

- **Mes 3-6**: Inicia con un framework o librería de JavaScript, como **React** o **Vue.js**.
  - Aprende sobre componentes, estado, propiedades y ciclos de vida.
  - Trabaja con APIs para hacer fetch de datos (REST, GraphQL).
  - Comienza a crear proyectos más dinámicos e interactivos.
- **Mes 6-12**: Participa en proyectos colaborativos o haz prácticas profesionales.
  - Aprende sobre control de versiones (Git) y desarrollo colaborativo.
  - Cubre algunos aspectos de optimización como manejo de assets (imágenes, scripts, CSS) y performance básico.

Para consolidar tu nivel junior:

Trabajar en proyectos reales es clave. Crear un portafolio con aplicaciones que demuestren tu capacidad para crear interfaces funcionales y atractivas. También es importante familiarizarte con herramientas de construcción como **Webpack** y preprocesadores como **SASS/SCSS**.

Desarrollo Web Básico

2. Transición a Desarrollador Senior:

- Convertirse en un desarrollador senior toma más tiempo, generalmente entre **4 a 6 años**. Esto se debe a que no es solo cuestión de dominar tecnologías, sino de adquirir experiencia práctica, resolver problemas complejos y asumir responsabilidades técnicas más avanzadas.

Factores clave en el nivel senior:

- **Experiencia Profunda:** Conoce a fondo varios frameworks frontend (no solo uno) y tecnologías asociadas.
- **Optimización de rendimiento:** Mejora las aplicaciones en términos de velocidad de carga, optimización de recursos y experiencia del usuario.

Desarrollo Web Básico

- **Arquitectura:** Ser capaz de diseñar y estructurar aplicaciones complejas desde cero, siguiendo buenas prácticas de modularidad, separación de responsabilidades y diseño escalable.
- **Pruebas y mantenimiento:** Experiencia con pruebas unitarias, integración continua (CI/CD), y metodologías ágiles (Scrum, Kanban).
- **Mentoría y liderazgo:** Los desarrolladores senior a menudo guían a equipos, revisan código y ayudan a implementar mejores prácticas.

Desarrollo Continuo y Proyectos Reales

- Durante este tiempo es fundamental trabajar en **proyectos desafiantes**, ya que enfrentarte a problemas reales y usuarios te ayuda a madurar como desarrollador. También es importante mantenerse actualizado con las tendencias y nuevas herramientas, ya que el campo del desarrollo frontend cambia rápidamente.

Desarrollo Web Básico

Recursos y Estrategia

- **Cursos Online**: Plataformas como **FreeCodeCamp**, **Udemy**, **Platzi** o **Scrimba** son ideales para aprender.
- **Documentación Oficial**: La documentación de librerías como **React**, **Vue.js** o **Svelte** es excelente para profundizar.
- **Proyectos personales**: Dedica tiempo a crear proyectos desde cero, ya que te ayudará a desarrollar un portafolio y a resolver problemas más allá del tutorial.

Resumen de tiempos estimados:

- **Junior**: 6 meses a 1 año, con aprendizaje continuo y práctica.
- **Senior**: 4 a 6 años, con experiencia en proyectos, habilidades avanzadas, y mentoría de otros desarrolladores.

Desarrollo Web Básico

La estructura de un proyecto frontend

Esta varía dependiendo de la complejidad del proyecto y de las herramientas o frameworks que utilices. Sin embargo, existen buenas prácticas y convenciones que se suelen seguir para mantener el proyecto organizado, escalable y fácil de mantener.

¿Qué considerar al estructurar un proyecto frontend?

1. **Modularidad:** Los archivos deben organizarse de manera que las responsabilidades estén claramente separadas, como estilos, componentes y lógica.

Desarrollo Web Básico

2. **Componentes reutilizables:** Especialmente en frameworks como React, Vue o Angular, es importante dividir la interfaz en componentes pequeños, modulares y reutilizables.
3. **Pruebas:** Siempre es útil organizar un directorio para pruebas unitarias o de integración, asegurando que la funcionalidad se mantenga correctamente.
4. **Escalabilidad:** A medida que el proyecto crece, se puede dividir el código en más carpetas como utils, services, models, etc.
5. **Mantenibilidad:** Siguiendo una estructura bien organizada, el proyecto será más fácil de mantener y de escalar en el futuro.

Desarrollo Web Básico

```
/mi-proyecto/
|
├── /assets/              # Recursos estáticos como imágenes, fuentes, etc.
|   ├── /img/             # Imágenes
|   └── /fonts/           # Fuentes personalizadas
|
├── /css/                 # Estilos CSS
|   ├── main.css          # Archivo principal de estilos
|   └── reset.css         # Estilos para reiniciar valores predeterminados
|
├── /js/                  # Archivos JavaScript
|   └── main.js           # Archivo principal de lógica JavaScript
|
├── /vendor/              # Librerías de terceros (por ejemplo, Bootstrap, jQuery)
|
├── index.html            # Página principal del proyecto
├── about.html            # Otra página del proyecto
└── contact.html          # Página de contacto
```

Desarrollo Web Básico

Desarrollo Web Básico

Creación de la página web

Ahora hablemos un poco sobre el desarrollo de la creación de una página web. Supongamos que tengo una empresa que se dedica a realizar páginas web. Y llega un cliente solicitando asesoramiento porque necesita una página para su negocio. ¿Cuáles son los pasos que seguiría para darle una solución ?

A continuación, los pasos que serían útiles para asegurarte de que el proyecto sea exitoso y que cumpla con las necesidades del cliente:

Desarrollo Web Básico

1. Reunión Inicial y Recopilación de Requisitos
2. Propuesta Inicial y Planificación del Proyecto
3. Diseño de la UX/UI
4. Desarrollo del Sitio Web
5. Pruebas y Control de Calidad (QA)
6. Presentación al Cliente y Revisión Final
7. Lanzamiento del Sitio Web
8. Capacitación y Documentación
9. Mantenimiento y Soporte

Desarrollo Web Básico

1. Reunión Inicial y Recopilación de Requisitos

- **Objetivo**: Entender la visión del cliente y los objetivos que tiene para su página web.
- **Acción**: Reúnete con el cliente (presencial o virtualmente) para conocer su negocio, el tipo de clientes que tiene, y lo que quiere lograr con la página web (por ejemplo, ventas, branding, comunicación). Es fundamental preguntar sobre:
  - **Objetivo del sitio**: ¿Es un sitio informativo, un catálogo de productos, una tienda online, un blog?
  - **Público objetivo**: ¿A quién está dirigida la página?
  - **Funcionalidades necesarias**: Formularios de contacto, tienda en línea, integración con redes sociales, etc.
  - **Estilo y diseño preferido**: Si tienen ejemplos de otros sitios que les gustan, eso ayuda a entender la estética deseada.

Desarrollo Web Básico

2. Propuesta Inicial y Planificación del Proyecto

- **Objetivo**: Presentar una solución adaptada a las necesidades del cliente y planificar cómo se llevará a cabo.
- **Acción**: Desarrolla una propuesta que incluya:
  - **Esquema del sitio**: Propuesta del número de páginas (Inicio, Nosotros, Servicios, Contacto, etc.).
  - **Wireframes iniciales**: Esbozos básicos del diseño para mostrar cómo se organizará la información.
  - **Tecnologías a utilizar**: Explica si usarás CMS como WordPress, desarrollos personalizados con HTML/CSS/JS, o frameworks como React, según la necesidad.
  - **Cronograma**: Indica fases del proyecto, desde el diseño hasta el lanzamiento.
  - **Presupuesto y alcance**: Presenta el costo estimado, dependiendo del trabajo necesario y las funcionalidades requeridas.

Desarrollo Web Básico

3. Diseño de la UX/UI

- **Objetivo**: Crear una experiencia visual y de navegación atractiva y sencilla para los usuarios.
- **Acción**: Trabaja en los **mockups y prototipos**:
  - **Diseño visual**: Desarrolla un prototipo que muestre los elementos visuales como colores, tipografía, disposición de elementos.
  - **Pruebas de usabilidad**: Comparte estos prototipos con el cliente para asegurarte de que la estética y navegación coincidan con sus expectativas. Se recomienda también obtener opiniones de los usuarios objetivo.

Desarrollo Web Básico

4. Desarrollo del Sitio Web

- **Objetivo**: Convertir el diseño aprobado en un sitio web funcional.
- **Acción**:
  - **Front-end**: Escribir el código HTML, CSS y JavaScript para implementar el diseño.
  - **Back-end** (si aplica): Si la página necesita funcionalidades específicas como bases de datos, carritos de compras, o autenticación de usuarios, desarrollar el back-end con tecnologías adecuadas (Node.js, Django, PHP, etc.).
  - **Sistema de Gestión de Contenidos (CMS)**: Si el cliente necesita actualizar el contenido por sí mismo, implementa un CMS para hacer la gestión más simple.

Desarrollo Web Básico

5. Pruebas y Control de Calidad (QA)

- **Objetivo**: Asegurar que el sitio esté libre de errores y sea fácil de usar en diferentes dispositivos y navegadores.
- **Acción**:
  - **Pruebas de funcionalidad**: Asegúrate de que todas las características del sitio funcionen según lo esperado (formularios, enlaces, carrito de compras).
  - **Pruebas de compatibilidad**: Verifica que el sitio sea **responsive** y se vea correctamente en dispositivos móviles, tablets y diferentes navegadores (Chrome, Firefox, Safari, etc.).
  - **Pruebas de velocidad**: Utiliza herramientas como Google PageSpeed Insights para optimizar la carga del sitio.
  - **Pruebas de accesibilidad**: Asegúrate de que la página sea accesible para personas con discapacidades, cumpliendo con pautas como WCAG.

Desarrollo Web Básico

6. Presentación al Cliente y Revisión Final

- **Objetivo**: Obtener la aprobación final del cliente antes del lanzamiento.
- **Acción**:
  - Muestra el sitio completo al cliente y repasa cada sección para asegurarse de que todos los elementos y funcionalidades sean las esperadas.
  - Haz los **ajustes finales** según las sugerencias del cliente.

7. Lanzamiento del Sitio Web

- **Objetivo**: Hacer que la página esté disponible para el público.
- **Acción**:
  - **Hosting y Dominio**: Configura el dominio y el alojamiento (hosting) donde se desplegará la página.
  - **Subida al servidor**: Sube todos los archivos y bases de datos al servidor, asegurándote de que todo funcione correctamente en el entorno de producción.
  - **Configuración de SEO**: Configura los elementos básicos de SEO (títulos, descripciones, metadatos, sitemap) para que la página sea fácil de encontrar en los motores de búsqueda.

Desarrollo Web Básico

8. Capacitación y Documentación

- **Objetivo**: Asegurar que el cliente pueda administrar el sitio web.
- **Acción**:
  - **Capacitación**: Si el sitio tiene un CMS, capacita al cliente para que sepa cómo actualizar el contenido.
  - **Documentación**: Proporciona documentación que describa cómo realizar tareas básicas en el sitio.

9. Mantenimiento y Soporte

- **Objetivo**: Mantener el sitio seguro y actualizado a largo plazo.
- **Acción**:
  - **Soporte Técnico**: Ofrece un plan de mantenimiento que incluya actualizaciones de seguridad, backups periódicos, y posibles mejoras.
  - **Optimización**: A medida que se recopilan datos de uso, propone optimizaciones para mejorar la experiencia del usuario o el rendimiento del sitio.

Desarrollo Web Básico

Conceptos de Marketing y Comunicación

1. **Branding**: Proceso de construir una marca, creando una identidad reconocible y confiable. En el contexto de una página web, branding implica la presentación visual (colores, logo, diseño) y el mensaje que se transmite a los visitantes para representar la empresa de manera consistente.
2. **Tienda Online**: Plataforma de comercio electrónico que permite a los usuarios comprar productos o servicios directamente desde el sitio web. Incluye funcionalidades como carrito de compras, gestión de productos, y sistemas de pago.
3. **Blog**: Sección del sitio web dedicada a la publicación de artículos o entradas de contenido. Los blogs son útiles para comunicar novedades, compartir conocimientos, mejorar el SEO y atraer tráfico al sitio web.

Desarrollo Web Básico

Conceptos Relacionados con Diseño y Desarrollo Web

4. **Wireframe**: Esquema visual simple del diseño de un sitio web que muestra la estructura básica y la disposición de los elementos antes de que se decidan los detalles del diseño gráfico. Sirve para planificar la disposición de contenido y navegación.
5. **UX/UI (User Experience/User Interface)**: UX se refiere a la **experiencia de usuario** y cómo interactúan con el sitio, asegurándose de que sea agradable y sencilla. UI se refiere a la **interfaz de usuario** y es la parte visual y estética del sitio, incluyendo botones, tipografía, y colores.

Desarrollo Web Básico

4. **Mockups**: Prototipos visuales detallados que muestran cómo se verá el sitio web después de aplicar los elementos de diseño como colores, imágenes y tipografía. Representan el aspecto final del producto antes de desarrollarlo.

5. **Front-end**: Parte del sitio web con la que los usuarios interactúan directamente. Incluye el desarrollo de la interfaz visual utilizando tecnologías como HTML, CSS y JavaScript.

6. **Back-end**: Parte del sitio web responsable de la lógica del servidor, bases de datos y procesamiento de datos. Se encarga de que los datos que ve el usuario estén correctos y se ejecuten las acciones solicitadas.

Desarrollo Web Básico

Conceptos de Pruebas y Optimización

9. **QA (Quality Assurance)**: Proceso de asegurar la calidad del sitio web a través de pruebas exhaustivas para detectar errores o problemas, asegurando que todas las funcionalidades trabajen correctamente.
10. **Responsive**: Técnica de diseño que permite que un sitio web se adapte a diferentes tamaños de pantalla y dispositivos (móviles, tablets, desktops) ofreciendo una buena experiencia al usuario en cualquier dispositivo.
11. **PageSpeed Insights**: Herramienta de Google que analiza la velocidad de carga de una página web y sugiere mejoras para optimizar el rendimiento.
12. **WCAG (Web Content Accessibility Guidelines)**: Directrices que buscan asegurar que el contenido web sea accesible para todas las personas, incluidas aquellas con discapacidades (visual, auditiva, cognitiva, etc.).

Desarrollo Web Básico

Conceptos Relacionados con Hosting y Posicionamiento

13. **Hosting**: Servicio que permite alojar el sitio web en un servidor para que esté disponible en internet. El hosting almacena todos los archivos necesarios para que el sitio funcione.
14. **SEO (Search Engine Optimization)**: Conjunto de prácticas para mejorar la visibilidad del sitio web en motores de búsqueda como Google. Incluye optimización de palabras clave, estructura del sitio, velocidad de carga, entre otros.
15. **Metadatos**: Información sobre el sitio web que ayuda a los motores de búsqueda a entender su contenido. Incluyen títulos, descripciones y etiquetas para cada página.
16. **Sitemap**: Archivo XML que enumera todas las páginas de un sitio web para facilitar la indexación por parte de los motores de búsqueda. Ayuda a los motores a rastrear todo el contenido de manera más eficiente.

Desarrollo Web Básico

Desarrollo Web Básico

1. Análisis de Requisitos Específicos del Proyecto IoT

- **Reunión Inicial y Objetivos**:
  - **Identificar usuarios clave**: Determinar los tipos de usuarios: usuarios finales que interactúan con los dispositivos y administradores de la empresa que gestionan las características del dispositivo.
  - **Funcionalidades necesarias**:
    - **Panel de Usuario Final**: Para monitorear y controlar dispositivos remotamente, obtener información sobre su estado, y ver datos como temperatura, consumo de energía, etc.
    - **Panel de Administración (Backend Interno)**: Para la empresa, que permita administrar actualizaciones de firmware, configurar los dispositivos, y realizar diagnósticos.

Desarrollo Web Básico

2. Propuesta de Solución y Arquitectura del Sitio

- **Estructura del Sitio**:
  - **Portal de Usuario Final**: Panel accesible donde los usuarios puedan:
    - Ver información de sus dispositivos conectados (datos en tiempo real).
    - Controlar aspectos del dispositivo (encender/apagar, cambiar configuraciones, etc.).
  - **Portal de Administración de Empresa**:
    - Gestión de dispositivos (registro, actualización de firmware, monitoreo de estado).
    - Herramientas de soporte (diagnóstico remoto, mensajes a usuarios).
- **Arquitectura del Sistema**:
  - Propón utilizar una arquitectura cliente-servidor, donde los dispositivos IoT se conecten a un servidor central y los usuarios accedan a este servidor para interactuar con los dispositivos.

Desarrollo Web Básico

3. Selección de Tecnologías

- **Backend y APIs**:
  - Desarrollar un **backend** que pueda gestionar los dispositivos. Aquí se podrían usar tecnologías como **Node.js**, **Django** o **Flask**.
  - Utilizar **APIs RESTful** para permitir la comunicación entre los dispositivos IoT y los servicios web.
  - **MQTT** para la comunicación entre los dispositivos y el servidor, ya que es un protocolo ideal para IoT por ser liviano y eficiente.
- **Frontend**:
  - **React**, **Vue.js** o **Angular** para desarrollar la interfaz de usuario interactiva tanto para los usuarios finales como para el panel de administración.
- **Base de Datos**:
  - Utilizar una **base de datos relacional** (MySQL, PostgreSQL) para almacenar la información de los usuarios y dispositivos.
  - Adicionalmente, **base de datos NoSQL** (como MongoDB) puede ser útil para almacenar datos en tiempo real generados por los dispositivos.

Desarrollo Web Básico

4. Diseño UX/UI Adaptado a los Usuarios

- **Panel de Usuario Final**:
  - **Dashboard de Control**: Desarrollar una interfaz sencilla que muestre los datos en tiempo real (por ejemplo, gráficos y tablas).
  - **Interacciones Simples**: Botones para controlar el dispositivo (encendido, cambios de modo, etc.).
- **Panel de Administración**:
  - **Control Detallado**: Pantalla que permita a la empresa gestionar dispositivos, actualizar el firmware y acceder a registros históricos.
  - **Alertas y Notificaciones**: Un sistema que notifique si hay dispositivos con problemas o si se requiere una actualización de firmware.

Desarrollo Web Básico

5. Desarrollo de la Plataforma

- **Back-end**:
  - **Autenticación y Autorización**: Implementar un sistema de roles para asegurar que los usuarios finales solo puedan acceder a sus propios dispositivos y los administradores puedan acceder a todo.
  - **Conexión con Dispositivos**: Usar APIs para permitir que los dispositivos IoT envíen datos al servidor y recibir comandos.
  - **Control de Firmware**: Desarrollo de una funcionalidad que permita cargar nuevos firmwares al servidor y desplegarlos a los dispositivos.
- **Front-end**:
  - **Dashboard del Usuario Final**: Desarrollo del panel de usuario que incluya todas las funciones de control y monitoreo de los dispositivos.
  - **Panel Administrativo**: Herramienta que permita administrar dispositivos, recibir alertas, ver estadísticas de uso, etc.

Desarrollo Web Básico

6. Pruebas Específicas para IoT

- **Pruebas de Funcionalidad**: Asegurarse de que los comandos enviados desde el panel web realmente lleguen a los dispositivos y funcionen correctamente.
- **Pruebas de Seguridad**:
  - **Autenticación y Autorización**: Pruebas para garantizar que los usuarios finales no puedan acceder a áreas administrativas.
  - **Encriptación**: Asegurarse de que las comunicaciones entre el servidor, la web y los dispositivos estén cifradas (por ejemplo, usando HTTPS y MQTT con SSL/TLS).
- **Pruebas de Conectividad**: Probar la estabilidad de la conexión entre los dispositivos IoT y el servidor.

Desarrollo Web Básico

7. Implementación de Seguridad

- **Cifrado de Comunicación**: Todo el tráfico entre el dispositivo, el servidor y la aplicación web debe estar cifrado.
- **Actualizaciones Seguras**: Para las actualizaciones de firmware, implementar una verificación para asegurarse de que solo firmwares autorizados puedan ser instalados.

8. Presentación al Cliente y Capacitación

- **Demostración del Sistema**: Mostrar cómo se pueden administrar y controlar los dispositivos.
- **Capacitación al Personal**: Capacitar al equipo del cliente sobre cómo actualizar los firmwares, monitorear los dispositivos y utilizar el portal administrativo.
- **Documentación**: Proporcionar manuales de usuario tanto para el personal administrativo como para los usuarios finales.

Desarrollo Web Básico

- **9. Lanzamiento y Mantenimiento**

- **Configuración del Hosting y Servidor**: Asegurarse de que el servidor donde se alojarán el backend y la base de datos tenga buena capacidad y seguridad.
- **Mantenimiento y Actualizaciones**:
  - **Monitoreo**: Ofrecer monitoreo continuo del sistema y alertas de fallas.
  - **Mantenimiento de Seguridad**: Realizar actualizaciones periódicas para asegurar que la infraestructura sea segura y esté libre de vulnerabilidades.

Desarrollo Web Básico

Conclusión

- En este caso, no solo se requiere una página web para mostrar información, sino una plataforma que permita tanto a los usuarios finales como a la empresa interactuar con los dispositivos. Esto implica el desarrollo de dos áreas diferenciadas: un panel de usuario y un panel administrativo. Además, es crucial considerar la comunicación eficiente y segura entre los dispositivos IoT y la plataforma web.
- La solución debe tener una buena infraestructura técnica, además de una experiencia de usuario sencilla y efectiva para ambas partes: los usuarios finales que interactúan con los dispositivos y el personal administrativo que gestiona su funcionamiento.

Desarrollo Web Básico

¡Muchas gracias!