Front-end

TechGuide - Alura, FIAP e PM3

Front-end

Nivel 1

- [ ] **HTML - Fundamentos**:

  - HTML es un lenguaje de marcado que define la estructura de su contenido. HTML consta de una serie de elementos que se utilizan para que se vea o actúe de cierta manera. Las etiquetas de archivos adjuntos pueden vincular una palabra o imagen a otro lugar, pueden poner palabras en cursiva, pueden hacer que la fuente sea más grande o más pequeña, etc.
  - Aprender qué etiquetas son necesarias para HTML básico
  - Crear de un párrafo de texto
  - Mostrar una imagen
  - Conocer la diferencia entre 'h1', 'h2', 'h3', etc.
  - Crear de un texto con hipervínculo
  - Crear un formulario con campos relevantes
  - Crear de una lista ordenada o desordenada de elementos
  - Crear de una lista de elementos en una lista desplegable
  - Vincular a un archivo CSS
  - Crear una tabla
  - Adicionar ID y clases

- [ ] **CSS - Fundamentos**:

- Las hojas de estilo en cascada (CSS) son un lenguaje de hoja de estilo usado para descubrir una presentación de un documento escrito en un lenguaje de marca, como HTML o XML. O CSS puede ser usado para estilizar textos de documentos muy básicos — por ejemplo, para alterar a cor e o tamanho de cabeçalhos e links. Ele pode ser usado para criar um layout — por ejemplo, transformando uma única columna de texto en um layout com uma área de conteúdo principal e uma barra lateral para información relacionada. Pode até ser usado para efectos como animação.
- Aprender a estructura visual de uma página, con 'margem' e 'preenchimento'
- Estabelecer o tamanho com 'larga' e 'altura'
- Aprender sobre la posición de un elemento ('estático', 'relativo' o 'absoluto')
- Aprender sobre la exposición de un elemento ('block', 'inline', 'inline-block')
- Aprender a posicionar imágenes en relación con el texto
- Aprender sobre alinhamento
- Aprender sobre estilo de fuente
- Aprender las diferencias y ventajas de usar las diferentes unidades de medida en CSS (%, relativo, etc.)
- Conectar-se a los elementos (IDs, clases) de un archivo HTML
- Cambiar las características de un elemento cuando se pasa el ratón sobre él
- Aprender a dimensionar cajas
- Aprender Flexbox
- Grado de aprendizaje

☐ **Javascript - Fundamentos**:

- Javascript es el lenguaje de programación más popular del mundo y es una de las tecnologías centrales de la World Wide Web, junto con HTML y CSS. Tiene escritura dinámica, orientación a objetos basada en prototipos y funciones de primera clase. Es un paradigma múltiple que admite estilos de programación imperativos, funcionales e impulsados por eventos.
- Conocer los tipos primitivos

- Declarar variables, considerando la diferencia entre 'var', 'let' y 'const
- Uso de estructuras condicionales ('if', 'else')
- Conocer los operadores de asignación y comparación ('=', '==', '===')
- Uso de estructuras de repetición y bucles ('while', 'for')
- Usar funciones, pasar parámetros y argumentos
- Manipulación de arreglos y listas
- Aprender el concepto de Orientación a Objetos
- Realizar un CRUD
- Obtener datos de una API
- Hacer llamadas asincrónicas usando 'Async/Await', 'Promise', etc.

☐ **DOM - Fundamentos**:

- El Document Object Model (DOM) es una interfaz de programación para documentos de la web. Representa la página para que los programas puedan cambiar la estructura, el estilo y el contenido del documento. El DOM representa el documento como nosotros y objetos; de esa forma, los lenguajes de programación pueden interactuar con la página.
- Entender cómo funciona el árbol DOM
- Accesando y manipulando elementos HTML y CSS
- Acceder a los padres e hijos de un elemento
- Insertar un nuevo elemento en el árbol
- Quitar un elemento del árbol
- Esperando un evento en un elemento determinado de la página usando

☐ **Accesibilidad en Javascript**:

- La Accesibilidad Digital es la eliminación de barreras en la Web. El concepto presupone que los sitios y portales sean diseñados de modo que todas las personas puedan percibir, entender, navegar e interactuar de manera efectiva con las páginas.
- Escribir código teniendo en cuenta la accesibilidad.

- [ ] **Estratégias de SEO**:
    - SEO significa optimización para motores de búsqueda y se refiere a las estrategias utilizadas para clasificar un sitio web dentro de los motores de búsqueda como Yahoo, Bing y, por supuesto, el más famoso de todos, Google.
    - Elegir palabras clave
    - Entender cómo Google clasifica las páginas
    - Conocer los factores de clasificación
    - Hacer Link Building
- [ ] **Diseño Responsivo**:
    - Ajustar tus páginas al tamaño de pantalla del usuario
    - Media queries
    - Conocer el concepto de Mobile first

Nivel 2

- [ ] **JavaScript- Callbacks y Promises**:
    - Una Promesa (Promises) es un proxy de un valor que no necesariamente se conoce cuando se crea la promesa. Esto permite que los métodos asíncronos devuelvan valores como los métodos síncronos- en lugar de devolver inmediatamente el valor final, el método asíncrono devuelve la promesa de proporcionar el valor en algún momento en el futuro.
    - Una función de devolución de llamada (Callback) es una función que se pasa a otra función como argumento, que luego se invoca dentro de la función externa para completar algún tipo de rutina o acción.
    - Una función asíncrona es una función declarada con la palabra clave asíncrona y la palabra clave espera está permitida dentro de ella. Las palabras clave async y await permiten que el comportamiento asincrónico basado en promesas se escriba en un estilo más claro, lo que evita la necesidad de configurar explícitamente cadenas de promesas.
    - Comprender el concepto de programación asíncrona

- Escribir código asíncrono entendiendo el concepto de promesas en JavaScript
- Usar métodos, palabras clave y objetos de JavaScript para manejar promesas como 'Async/Await', '.then()', 'Promise', etc.
- Aprender en qué situaciones necesitas usar la programación asíncrona
- Llamar a las API con 'fetch()'

- [ ] **JavaScript - Pruebas**:

- La prueba de software es el proceso de evaluación y verificación de que un software realmente hace lo que debería hacer. Los beneficios de las pruebas incluyen la prevención de errores, la reducción de los costos de desarrollo y la mejora del rendimiento.
- Usar pruebas unitarias
- Usar pruebas de integración
- Usar pruebas de comportamiento (behavior)
- Usar mocks

- [ ] **JavaScript - Manejo de errores**:

- El manejo de errores se refiere a los procedimientos de respuesta y recuperación de las condiciones de error presentes en una aplicación de software. En otras palabras, es el proceso compuesto por la anticipación, detección y resolución de errores de aplicación, programación o comunicación.
- Conocer y manejar las excepciones más comunes
- Conocer los tipos de errores y en qué situaciones se pueden producir
- Comprender cómo Node.js maneja los errores
- Usar 'try' y 'catch' para el manejo de errores
- Aprender en qué ocasiones y cómo usar `throw`
- Creación de excepciones específicas según las necesidades de su aplicación

- [ ] **JavaScript - ES6**:

- Conocer las diferencias de esta versión de JavaScript

- [ ] **JavaScript - Modularización**:

- Dividir partes del código en módulos
- Usar import y export

- [ ] **Versiones semánticas de front-end**:

- SemVer (Semantic Versioning) es un conjunto simple de reglas y requisitos que dictan cómo se asignan e incrementan los números de versión.
- Organización de las dependencias de un proyecto
- Evitar el "infierno de la dependencia"

- [ ] **Jest**:

- Jest es un corredor de pruebas de JavaScript que le permite acceder al DOM a través de jsdom. Proporciona una gran velocidad de iteración combinada con funciones potentes como módulos de simulación y temporizadores para que pueda tener más control sobre cómo se ejecuta el código.
- Componentes de prueba

- [ ] **Cypress**:

- Cypress es una herramienta de prueba Front-end que permite configurar, escribir, ejecutar y depurar pruebas.

Nivel 3

- [ ] **Estructura de Datos**:

- En el contexto de los ordenadores, una estructura de datos es una forma específica de almacenar y organizar los datos en la memoria del ordenador para que esos datos puedan ser fácilmente recuperados y utilizados de forma eficiente cuando sea necesario posteriormente.
- Conocer las principales estructuras de datos
- Implementar las principales estructuras de datos

- [ ] **Conceptos de Orientación a Objetos**:
  - La Programación Orientada a Objetos es un paradigma de programación de software basado en la composición e interacción entre diversas unidades llamadas 'objetos' y las clases, que contienen una identidad, propiedades y métodos. Se basa en cuatro componentes de la programación - abstracción digital, encapsulación, herencia y polimorfismo.
  - Cómo funcionan los objetos
  - Crear y utilizar constructores
  - Qué son las clases
  - Crear y utilizar métodos
  - Cómo funciona la encapsulación
  - Qué es la herencia
  - Qué es el polimorfismo
  - Cómo funcionan las interfaces
  - Qué son las abstracciones
- [ ] **JavaScript - Almacenamiento**:
  - Almacenar datos en el front-end con localStorage
  - Manipular datos almacenados
  - Persistir datos almacenados
- [ ] **JavaScript - Concurrencia**:
  - La programación concurrente es un paradigma de programación para construir programas que utilizan la ejecución simultánea de varias tareas computacionales interactivas, que pueden ser implementadas como programas separados o como un conjunto de hilos creados por un único programa
  - Ejecutar tareas en paralelo
- [ ] **TypeScript - Fundamentos**:

- TypeScript es un lenguaje de programación fuertemente tipado que se basa en JavaScript.
- Comprender en profundidad qué son los tipos y la importancia de la programación tipificada
- Aprender qué es TypeScript, por qué se creó, cómo funciona y su relación con JavaScript
- Conocer las herramientas de TypeScript (integración con el editor de código, verificador estático y compilador)
- Escribir código en TypeScript usando sus herramientas (interfaces, enumeración, decoradores, etc.)

☐ **GraphQL**:

- GraphQL es un nuevo estándar API que proporciona una alternativa más eficiente, potente y flexible a REST. Fue desarrollado y de código abierto por Facebook y ahora lo mantiene una gran comunidad de empresas e individuos de todo el mundo.
- Comprender cómo se utiliza GraphQL en el desarrollo de API
- Creación de API utilizando bibliotecas y marcos GraphQL

☐ **Apollo Client**:

- Apollo Client es una biblioteca integral de administración de estado para JavaScript que le permite administrar datos locales y remotos con GraphQL.
- Utilizar Apollo para crear un servidor GraphQL
- Conectar a una API

Habilidade Auxiliar: Infraestructura y Back-end

☐ **Git y GitHub - Fundamentos**:

- Git es un sistema de control de versiones distribuido gratuito y de código abierto diseñado para manejar todo, desde proyectos pequeños hasta proyectos muy grandes, con rapidez y eficiencia.

- GitHub es un servicio de hosting para el desarrollo de software y el control de versiones mediante Git.
- Crear un repositorio
- Clonar un repositorio
- Comprometerse, empujar y tirar hacia y desde el repositorio
- Revertir un commit
- Crear de ramas y Pull requests
- Manejar fusiones y conflictos

☐ **HTTP - Fundamentos**:

- HTTP significa Protocolo de transferencia de hipertexto. La comunicación entre las computadoras cliente y los servidores web se realiza mediante el envío de solicitudes HTTP y la recepción de respuestas HTTP.
- Comprender la diferencia entre los verbos HTTP
- Probar solicitudes y verificar los códigos de estado en el navegador
- Aprendiendo a hacer una solicitud HTTP en la línea de comando con WGET
- Descargar una imagen con WGET
- Realización de una POST

☐ **JSON**:

- JSON significa Notación de objetos de JavaScript. Es un formato de texto para almacenar y transportar datos.
- Crear un objeto
- Transformar un objeto en una cadena
- Transformar una cadena en un objeto
- Manipular un objeto

☐ **Línea de Comando - Fundamentos**:

- CLI es un programa de línea de comandos que acepta la entrada de texto para ejecutar funciones del sistema operativo.

- Conocer los comandos más importantes

- [ ] **Cloud - Fundamentos**:

- La computación en nube, o cloud computing, es la distribución de servicios informáticos a través de Internet mediante un modelo de tarificación de pago por uso. Una nube se compone de varios recursos informatizados, desde los propios ordenadores (o instancias, en terminología de nube) hasta las redes, el almacenamiento, las bases de datos y todo lo que les rodea. En otras palabras, todo lo que normalmente se necesita para montar el equivalente a una sala de servidores, o incluso un centro de datos completo, estará listo para usar, configurar y ejecutar.
- Conocer la diferencia entre IaaS, PaaS y SaaS
- Conocer los mayores proveedores de nube
- Especializarse en un proveedor específico de su preferencia

- [ ] **YARN**:

- Yarn es un administrador de paquetes para su código. Le permite usar y compartir código con otros desarrolladores. El código se comparte a través de algo llamado paquete (a veces denominado módulo). Un paquete contiene todo el código que se comparte, así como un archivo package.json que O describe.
- Administrar paquetes
- Administrar dependencias
- Instalar paquetes sin conexión
- Comandos
- El archivo yarn.lock

Habilidade Auxiliar: UX y Design

- [ ] **Sistemas de Diseño**:

- Un sistema de diseño es una colección de componentes reutilizables, guiados por estándares claros, que se pueden ensamblar para crear

aplicaciones.

- Creación y mantenimiento de bibliotecas que se consumirán y utilizarán como estándar para construir un proyecto.

- [ ] **Figma - Fundamentos**:

- Figma es una aplicación web colaborativa para el diseño de interfaces. El conjunto de funciones de Figma se centra en la interfaz de usuario y el diseño de la experiencia del usuario, con énfasis en la colaboración en tiempo real, utilizando una variedad de herramientas de creación de prototipos y editor de gráficos vectoriales.
- Crear diseños de página y componentes

- [ ] **Sistemas de color**:

- Definición de una paleta de colores que tenga sentido para una interfaz dada

- [ ] **Cómo usar fuentes**:

- Elegir la fuente más adecuada para un proyecto determinado
