Guia Ágil

TechGuide - Alura

React

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

☐ **Conceptos de SPA**:

- Single-page application o SPA es una aplicación web de una sola página o sitio, que interactúa con el usuario reescribiendo dinámicamente la página web actual con nuevos datos del servidor web, en lugar del método por defecto de un navegador web que carga páginas nuevas enteras. El objetivo

es conseguir transiciones más rápidas que hagan que el sitio web parezca más una aplicación nativa.

- Entender qué es una SPA
- Establecer rutas a otras páginas
- Conocer los frameworks SPA
- Comunicarse con APIs

- [ ] **React - Componentes**:

- React te permite definir componentes como clases o funciones. Los componentes definidos como clases proporcionan más capacidades. Aceptan entradas arbitrarias (llamadas "accesorios") y devuelven elementos React que describen lo que debería aparecer en la pantalla.
- Comprender cómo funcionan los componentes
- Conociendo la biblioteca de componentes con estilo
- Comprender la diferencia entre clase y componentes funcionales

- [ ] **React - Props**:

- Props son objetos que se inyecta en los componentes y proporciona algunos datos que se pueden compartir entre otros componentes en un flujo de datos unidireccional desde un elemento principal a un elemento secundario.
- Las Props son de solo lectura.
- accesorios de paso
- manipulación de accesorios

- [ ] **React Hooks - State**:

- Controlar el estado de los componentes.
- Manipular variables
- Actualizar el valor de los elementos.

- [ ] **Crear una aplicación React**:

- Create React App es una forma oficialmente admitida de crear aplicaciones React de una sola página. Ofrece una configuración de construcción moderna sin configuración.
- Estructuración de un nuevo proyecto React
- Crear una aplicación funcional desde cero

- [ ] **React Hooks - Effect**:

- UseEffect() es un React Hook que le permite manejar los efectos secundarios en sus componentes funcionales de React.
- Ejecutar un componente solo después de renderizar
- Acceso a los accesorios de un elemento
- API de llamadas

Nivel 2

- [ ] **React Hooks - Memo**:

- El React Hook `useMemo` devuelve un valor memorizado. `useMemo` solo volverá a calcular el valor memorizado cuando una de las dependencias haya cambiado.
- Controlar el estado de las variables externas
- Evitar cálculos costosos en cada render

- [ ] **React Hooks - Callback**:

- El Hook `useCallback` de React devuelve una función Callback memorizada.
- Aislamiento de funciones intensivas en recursos para que no se ejecuten automáticamente en cada elemento renderizado

- [ ] **React Hooks - Ref**:

- El Hook `useRef` te permite persistir valores entre renderizaciones.
- Almacenar un valor mutable que no provoca una nueva representación cuando se actualiza
- Acceder a un elemento DOM directamente

- [ ] **React - Bibliotecas de Design System**:
  - Un sistema de diseño es una colección de componentes reutilizables, guiados por estándares claros, que se pueden ensamblar para construir cualquier cantidad de aplicaciones.
- [ ] **React Developer Tools**:
  - React Developer Tools se utiliza para inspeccionar los componentes de React, editar accesorios y estados e identificar problemas de rendimiento.
  - Depuración de aplicaciones
- [ ] **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"
- [ ] **CSS-in-JS**:
  - CSS-in-JS se refiere a un patrón en el que CSS se compone utilizando JavaScript en lugar de definirse en archivos externos.
  - Manejar código CSS usando JavaScript
  - Usar styled-components
  - Conocer Styled-JSX
- [ ] **Styled Components**:
  - Styled Components le permiten escribir código CSS real para diseñar sus componentes usando literales de plantilla etiquetados y el poder de CSS.
  - Manejo de código CSS en componentes
- [ ] **React Router**:
  - Manipulación de la navegación entre interfaces y componentes
- [ ] **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.)

☐ **React Testing Library**:

- React Testing Library se basa en DOM Testing Library al agregar API para trabajar con componentes React. Es un conjunto de ayudantes que le permiten probar los componentes de React sin depender de los detalles de su implementación.
- Probando los componentes de React

☐ **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.

☐ **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 - 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

- [ ] **Babel - Fundamentos**:

- Babel es un compilador de JavaScript. Es una cadena de herramientas que se utiliza principalmente para convertir el código ECMAScript 2015+ en una versión de JavaScript compatible con versiones anteriores en navegadores o entornos actuales y anteriores.

Nivel 3

- [ ] **Lottie**:

- Lottie es una biblioteca que analiza las animaciones de Adobe After Effects exportadas como json con Bodymovin y las renderiza de forma nativa en dispositivos móviles y en la web.

- [ ] **Framer Motion**:

- Framer Motion es una biblioteca de movimiento lista para producción para React de Framer.
- Creando animaciones

- [ ] **Service Workers**:

- Los Service Workers actúan esencialmente como servidores proxy que se ubican entre las aplicaciones web, el navegador y la red (cuando está disponible). Están destinados, entre otras cosas, a permitir la creación de experiencias fuera de línea efectivas, interceptar solicitudes de red y tomar las medidas adecuadas en función de si la red está disponible y actualizar los activos que residen en el servidor. También permitirán el acceso a notificaciones push y API de sincronización en segundo plano.

- [ ] **React Hooks - Form**:

- React Hooks Form es una biblioteca que proporciona validación de formularios.

- [ ] **Lodash**:

- Lodash es una moderna biblioteca de utilidades de JavaScript que ofrece modularidad, rendimiento y extras. Los métodos modulares de Lodash son excelentes para iterar arreglos, objetos y cadenas; manipular y probar valores; y la creación de funciones compuestas.

- [ ] **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

- [ ] **Redux Saga**:

- Redux Saga es una biblioteca que tiene como objetivo hacer que los efectos secundarios de la aplicación (es decir, cosas asincrónicas como la obtención de datos y cosas impuras como acceder al caché del navegador) sean más fáciles de administrar, más eficientes de ejecutar, fáciles de probar y mejores en el manejo de fallas.

- [ ] **NextJS - Fundamentos**:

- Next.js es un marco de desarrollo web de código abierto creado por Vercel que permite aplicaciones web basadas en React con representación del lado del servidor y generación de sitios web estáticos.
- Creación de interfaces web
- Disminución de los tiempos de carga de la página
- Representación de páginas del lado del servidor

- Mejorando el rendimiento en React
- Creación de rutas API con funciones sin servicio

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
