iOS

TechGuide - Alura, FIAP e PM3

iOS

Nivel 1

- [ ] **Swift - Fundamentos**:

- Swift é uma linguagem de programação de código aberto desenvolvida pela Apple Inc. e pela comunidade de desenvolvedores. É projetada para ser segura, rápida e intuitiva. Swift é a linguagem preferida para o desenvolvimento de aplicativos iOS, macOS, watchOS e tvOS. Veja as principais habilidades que você precisa desenvolver:
- Conhecer as palavras chaves do Swift;
- Entender como funciona o controle de fluxo e as condicionais;
- Construir aplicações de terminal usando Swift;
- Conhecer os protocolos de Encodable e Decodable;
- Compreender e aprender a usar coleções no Swift.

- [ ] **iOS - UIKit - Fundamentos**:

- O UIKit é a biblioteca de código padrão para aplicativos iOS. Ele fornece uma ampla gama de recursos para criar interfaces de usuário, incluindo views, controles e animações. O UIKit é uma ótima maneira de criar aplicativos iOS que sejam visualmente atraentes e fáceis de usar. Veja as principais habilidades que você precisa desenvolver:

- Conhecer Swift ou Objective-C que são as linguagens para desenvolver apps iOS;
- Entender como o SDK do iOS funciona (Core Foundation e Cocoa Touch);
- Conhecer os componentes de interface do UIKit para criação de layots;
- Dominar a criação de interfaces utilizando o UIKit, tanto através de código (ViewCode) quanto pelo uso de Storyboards;
- Explorar os princípios de Auto Layout e como criar layouts flexíveis e responsivos para diferentes tamanhos de tela;
- Implementar a navegação entre telas e a passagem de dados utilizando o sistema de view controllers do UIKit.

☐ **Conceitos de Orientação a Objetos**:

- A Programação Orientada a Objetos é um paradigma de programação de software baseado na composição e interação entre diversas unidades chamadas de 'objetos' e as classes, que contêm uma identidade, propriedades e métodos). Ela é baseada em quatro componentes da programação: abstração digital, encapsulamento, herança e polimorfismo.
- Como funcionam objetos
- Criar e utilizar construtores
- O que são classes
- Criar e utilizar métodos
- Como funciona encapsulamento
- O que é herança
- O que é polimorfismo
- Como funcionam interfaces
- O que são abstrações

☐ **iOS - Storyboard**:

- Um storyboard no iOS é um arquivo de XML que contém um diagrama visual de uma interface do usuário. Os storyboards são usados para criar aplicativos iOS que são visualmente atraentes e fáceis de navegar. Os

storyboards contêm uma hierarquia de views, que são os elementos visuais que compõem a interface do usuário. As views podem ser adicionadas aos storyboards arrastando-as da paleta de views para o diagrama. As views podem ser personalizadas alterando suas propriedades, como tamanho, cor e fonte. Veja as principais habilidades que você precisa desenvolver:

- Conhecer o conceito de Storyboards e como eles são usados para criar aplicativos iOS;
- Criar e usar Storyboards para criar layouts de interface do usuário;
- Implementar a navegação entre telas em aplicativos iOS;
- Aprofundar os princípios de Auto Layout e como criar layouts flexíveis e responsivos para diferentes tamanhos de tela.

- [ ] **iOS - ViewCode**:

- O método de viewcode no UIKit é uma abordagem para a criação de interfaces de usuário que envolve a codificação da interface de usuário diretamente em Swift, em vez de usar um storyboard ou nib. O método de viewcode pode ser uma ótima maneira de criar interfaces de usuário mais personalizadas e eficientes, mas é importante notar que ele pode ser mais complexo do que usar um storyboard ou nib. Veja as principais habilidades que você precisa desenvolver:
- Conhecer o conceito de ViewCode e como ele é usado para criar aplicativos iOS;
- Criar e usar ViewCode para criar layouts de interface do usuário;
- Explorar os princípios de Auto Layout com ViewCode.

- [ ] **iOS - Imagens e Ícones**:

- Os recursos do app em iOS são os ativos visuais que são usados para representar o seu app. Estes recursos incluem ícones, imagens, fontes e outros tipos de arquivos. Os recursos do app são importantes porque eles ajudam os usuários a identificar e se conectar com o seu app. Eles também podem ser usados para melhorar a estética do seu app e torná-lo mais atraente para os usuários. Veja as principais habilidades que você precisa desenvolver.

- Conhecer os diferentes tipos de recursos de aplicativos iOS, como imagens, ícones e outros recursos;
- Adicionar e usar imagens, ícones e outros recursos em aplicativos iOS;
- Personalizar recursos de aplicativos iOS para diferentes dispositivos e resoluções de tela.

- [ ] **iOS - Persistência de dados**:

- A persistência de dados em iOS é o processo de armazenar e recuperar dados em um dispositivo iOS. Isso pode ser feito usando uma variedade de métodos, incluindo arquivos, Core Data e Realm. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os diferentes tipos de persistência de dados no iOS, como arquivos, Core Data, SwiftData e Realm;
- Armazenar e recuperar dados em um aplicativo iOS usando arquivos, Core Data, SwiftData e Realm;
- Personalizar a persistência de dados em um aplicativo iOS para diferentes necessidades.

- [ ] **iOS - SwiftUI**:

- SwiftUI é uma biblioteca de interface de usuário declarativa para criar aplicativos iOS, iPadOS, macOS, watchOS e tvOS. Ele é construído sobre o UIKit e o AppKit, mas fornece uma maneira mais declarativa de criar interfaces de usuário. Isso torna mais fácil criar interfaces de usuário bonitas e envolventes. Veja as principais habilidades que você precisa desenvolver:
- Conhecer SwiftUI e como usá-lo para criar interfaces de usuário para aplicativos iOS;
- Usar combinações de componentes para criar layouts;
- Usar os métodos de navegação para adicionar mais versatilidade para seu app.

- [ ] **Xcode**:

- O Xcode é a ferramenta que os desenvolvedores usam para criar apps para o ecossistema da Apple – MacOS, iOS e todos os produtos da Apple.
- Conhecer a ferramenta Xcode;
- Criar um novo projeto;
- Usar o emulador;
- Debugar seu código;
- Debugar é o processo de encontrar e corrigir erros em um aplicativo. É uma parte essencial do desenvolvimento de software e é importante saber como fazer debugging de forma eficaz. O Xcode é o IDE oficial da Apple para desenvolvedores iOS. Ele inclui uma série de ferramentas integradas que podem ser usadas para debugging de aplicativos, incluindo um depurador, um analisador de memória e um analisador de desempenho. Veja as principais habilidades que você precisa desenvolver:
- Conhecer as ferramentas e técnicas mais comuns para debugging de aplicativos iOS;
- Usar as ferramentas e técnicas mais comuns para debugging de aplicativos iOS para encontrar e corrigir erros.

☐ **Estruturas de Dados**:

- No contexto dos computadores, uma estrutura de dados é uma forma específica de armazenar e organizar os dados na memória do computador para que esses dados possam ser facilmente recuperados e utilizados de forma eficiente quando necessário posteriormente.
- Conhecer as principais estruturas de dados
- Implementar as principais estruturas de dados

Nivel 2

☐ **iOS - Assíncrono**:

- Assincronicidade é uma técnica de programação que permite que as tarefas sejam executadas em segundo plano, enquanto a interface do usuário continua a ser executada normalmente. Isso pode melhorar o desempenho

dos aplicativos, pois evita que eles travem ou fiquem lentos enquanto aguardam a conclusão de tarefas demoradas. Uma das maneiras mais comuns de usar assincronicidade em iOS é usar os métodos async e await. Os métodos async e await permitem que você execute código em segundo plano e retorne para o código principal antes que a tarefa seja concluída. Isso pode ser útil para tarefas que demoram muito para serem concluídas, como baixar um arquivo grande ou processar uma imagem grande. Veja as principais habilidades que você precisa desenvolver:

- Conhecer os fundamentos da assincronicidade em iOS;
- Usar as classes OperationQueue e async e await para executar tarefas em segundo plano.

☐ **iOS - Comunicação com APIs**:

- As comunicações API em iOS são o processo de comunicação entre um aplicativo iOS e um serviço externo. Isso pode ser feito usando uma variedade de métodos, incluindo HTTP, HTTPS e Sockets. As comunicações API são importantes porque elas permitem que os aplicativos iOS acessem dados e recursos que não estão disponíveis no dispositivo local. Elas também podem ser usadas para compartilhar dados e recursos entre aplicativos iOS e serviços externos. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os diferentes tipos de APIs que podem ser usadas em aplicativos iOS;
- Conhecer e utilizar bibliotecas/ferramentas para facilitar a comunicação a APIs, como o Alamofire, URLSession e Codable;
- Escolher e usar APIs que sejam adequadas para as necessidades do seu aplicativo;
- Testar comunicações API em diferentes condições para garantir que elas funcionem corretamente.

☐ **iOS - Gerenciamento de Estado**:

- Gerenciamento de estado é o processo de rastrear e gerenciar a condição de um aplicativo. Isso é importante para aplicativos que são complexos ou

que têm muitos estados diferentes. No UIKit, o gerenciamento de estado é feito manualmente pelo desenvolvedor. Isso significa que o desenvolvedor é responsável por rastrear o estado do aplicativo e atualizar a interface do usuário conforme o estado muda. Existem várias maneiras diferentes de gerenciar o estado no UIKit. Uma maneira comum é usar uma estrutura de dados para armazenar o estado. No SwiftUI, o gerenciamento de estado é feito automaticamente pelo framework. Isso significa que o desenvolvedor não precisa se preocupar em rastrear o estado do aplicativo ou atualizar a interface do usuário conforme o estado muda. O SwiftUI usa um sistema de gerenciamento de estado baseado em propriedade para gerenciar o estado dos aplicativos. Isso significa que o desenvolvedor pode definir propriedades que armazenam o estado do aplicativo e o framework atualizará automaticamente a interface do usuário conforme o estado muda. Veja as principais habilidades que você precisa desenvolver:

- Conhecer os fundamentos do gerenciamento de estado em iOS;
- Conhecer programaçao reativa para frameworks que usam o padrão declarativo (UIKit);
- Usar estruturas de dados e bibliotecas de gerenciamento de estado para armazenar e gerenciar o estado de um aplicativo.

☐ **iOS - Recursos do sistema**:

- O iOS fornece acesso a uma variedade de recursos do sistema, incluindo localização, câmera e sensores. Esses recursos podem ser usados para criar aplicativos que são mais envolventes e informativos. Por exemplo, um aplicativo de jogos pode usar o recurso de localização para rastrear a localização do usuário e adaptar o jogo de acordo. Um aplicativo de fotografia pode usar o recurso da câmera para tirar fotos e vídeos. E um aplicativo de saúde pode usar o recurso de sensores para rastrear a atividade física do usuário. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os recursos do sistema em iOS;
- Acessar a localização, a câmera e os sensores;
- Criar aplicativos que usam recursos do sistema.

- [ ] **iOS - Testes**:

- Os testes são uma parte importante do desenvolvimento de software. Eles ajudam a garantir que o software esteja funcionando corretamente e que atenda aos requisitos do usuário. Existem vários tipos de testes que podem ser usados em iOS, incluindo testes de unidade, testes de integração e testes de sistema. Os testes de unidade são usados para testar unidades individuais de código, os testes de integração são usados para testar como diferentes unidades de código interagem umas com as outras e os testes de sistema são usados para testar todo o aplicativo como um todo. O teste é uma parte importante do processo de desenvolvimento de software e pode ajudar a garantir que os aplicativos iOS sejam de alta qualidade e confiáveis. Veja as principais habilidades que você precisa desenvolver:
- Conhecer as ferramentas e técnicas mais comuns para debugging de aplicativos iOS;
- Usar as ferramentas e técnicas mais comuns para debugging de aplicativos iOS para encontrar e corrigir erros.

- [ ] **iOS - Vazamento de Memória**:

- Uma memory leak é um problema que ocorre quando um aplicativo não libera a memória que não está mais usando. Isso pode levar a uma série de problemas, como o aplicativo ficar lento, travar ou até mesmo ser fechado pelo sistema operacional. Para evitar memory leaks, é importante entender como o gerenciamento de memória funciona no iOS e usar as ferramentas e técnicas adequadas para gerenciar a memória do seu aplicativo. Veja as principais habilidades que você precisa desenvolver:
- Conhecer as causas de memory leaks em iOS;
- Identificar os sintomas de memory leaks e resolvê-los.

- [ ] **iOS - Bibliotecas de terceiros**:

- As bibliotecas de terceiros mais utilizadas em iOS são aquelas que fornecem funcionalidades adicionais ao SDK da Apple. Essas bibliotecas podem ser usadas para acessar dados da web, trabalhar com imagens e vídeos,

desenvolver jogos, criar animações e muito mais. Veja as principais habilidades que você precisa desenvolver:

- Conhecer os diferentes tipos de bibliotecas que podem ser usadas em aplicativos iOS;
- Escolher e usar bibliotecas que sejam adequadas para as necessidades do seu aplicativo.

Nivel 3

- [ ] **iOS - Arquitetura**:

- As arquiteturas MVVM, MVC e VIPER são maneiras populares de estruturar aplicativos iOS. Ambas as arquiteturas se baseiam no conceito de separar a lógica de negócios da interface do usuário. Isso torna os aplicativos mais fáceis de manter e estender. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os conceitos básicos de arquiteturas;
- Entender as diferenças entre as arquiteturas e suas vantagens;
- Implementar alguma delas em um aplicativo iOS.

- [ ] **iOS - Design System**:

- A Apple oferece ferramentas e recursos para que os aplicativos sigam as melhores práticas de design e interação no iOS.
- Entender as Human Interface Guidelines (HIG) da Apple;
- Criar componentes reutilizáveis e escaláveis para iOS;
- Aplicar o Design System em diferentes dispositivos iOS.

- [ ] **iOS - Injeção de dependências**:

- A injeção de dependências (DI) é um padrão de projeto que permite que os componentes de um aplicativo recebam as dependências de que precisam em tempo de execução. Isso torna os componentes mais testáveis e flexíveis. Existem várias maneiras de implementar a DI em iOS. Uma maneira comum é usar um contêiner de injeção de dependências. Um contêiner de

injeção de dependências é um objeto que armazena as dependências de um aplicativo e as fornece aos componentes conforme necessário. A DI é uma ferramenta poderosa que pode ajudar os desenvolvedores a criar aplicativos mais testáveis e flexíveis. Ao aprender como usar a DI, você pode criar aplicativos que são mais fáceis de manter e estender. Veja as principais habilidades que você precisa desenvolver:

- Conhecer e utilizar o padrão de projeto de injeção de dependências.

☐ **iOS - Animações**:

- As animações são uma parte importante do desenvolvimento de aplicativos iOS. Elas podem ser usadas para melhorar a experiência do usuário, tornando os aplicativos mais atraentes e envolventes. Existem várias maneiras de adicionar animações aos aplicativos iOS. Uma maneira é usar o framework UIKit, que fornece uma variedade de métodos para animar componentes da interface do usuário. Outra maneira é usar um framework de animação de terceiros, como o Lottie ou o Anima. Veja as principais habilidades que você precisa desenvolver:
- Entender como usar o framework UIKit para animar componentes da interface do usuário;
- Criar animações com o SwiftUI, como mostrar transições entre telas, atualizar a interface do usuário em resposta a eventos e criar efeitos visuais interessantes.

☐ **iOS - Implantação (Deployment)**:

- O deployment ou implantação é o processo de disponibilizar um aplicativo iOS para os usuários. Existem várias maneiras de fazer isso, um deles é o uso da App Store. A App Store é a maneira mais comum de distribuir aplicativos iOS. Ela permite que os desenvolvedores distribuam seus aplicativos para um público global. O TestFlight é uma ferramenta beta privada que permite que os desenvolvedores compartilhem seus aplicativos com um grupo limitado de usuários para feedback. O Distribute é uma ferramenta que permite que os desenvolvedores distribuam seus aplicativos para seus próprios servidores. Veja as principais habilidades que você precisa desenvolver:

- Conhecer os conceitos básicos de deployment de aplicativos iOS;
- Criar uma conta de Apple Developer;
- Usar a App Store para distribuir aplicativos iOS para um público global.

☐ **Entrega e integração contínuas (CI/CD)**:

- CI/CD é a abreviação de Continuous Integration/Continuous Delivery, traduzindo para o português "entrega e integração contínuas". Trata-se de uma prática de desenvolvimento de software que visa tornar a integração de código mais eficiente por meio de builds e testes automatizados.
- Automatizar a integração de código entre varias partes da equipe se tornou cada vez mais importante, ja que assim é possivel acelerar o desenvolvimento e diminiur o tempo de entrega de software.
- Executar testes automatizados da aplicação para verificar seu funcionamento.
- Realizar a entrega de atualizações de forma automatica e com segurança.
- Realizar testes de conexão e testes de carga para evitar que a aplicação apresente problemas ao ser atualizada.

☐ **iOS - Modularização**:

- A modularização é o processo de dividir um grande aplicativo em módulos menores, mais gerenciáveis. Isso pode tornar o código mais fácil de entender e manter, e também pode facilitar a reutilização de código. Existem várias maneiras de modularizar um aplicativo iOS. Uma maneira é usar frameworks. Os frameworks são bibliotecas de código que podem ser usadas para adicionar funcionalidade a um aplicativo. Outra maneira de modularizar um aplicativo é usar módulos de projeto. Os módulos de projeto são coleções de arquivos de código que podem ser usados para agrupar o código relacionado. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os conceitos básicos de modularização de código.

☐ **Objective-C - Fundamentos**:

- Objective-C é uma linguagem de programação orientada a objeto que foi criada em 1984 por Brad Cox e Tom Love. Ela é usada para desenvolver aplicativos para o macOS, iOS e tvOS. Objective-C é uma linguagem de programação poderosa e flexível, e é usada por muitos desenvolvedores experientes. Veja as principais habilidades que você precisa desenvolver:
- Conhecer os tipos primitivos;
- Declarar e usar variáveis e constantes;
- Usar estruturas condicionais (if, else);
- Usar estruturas de repetição e laços (while, for);
- Usar funções, passando parâmetros e argumentos;
- Implementando métodos e reutilizando eles;
- Exceptions e Throwables;
- Convenções de código;
- Compreensão de Orientação a objetos.

☐ **iOS - Jogos**:

- A plataforma iOS é uma ótima opção para o desenvolvimento de jogos. Ela possui uma grande base de usuários, com bilhões de dispositivos ativos, e uma série de recursos que tornam o desenvolvimento de jogos fácil e acessível. Para desenvolver jogos para iOS, você pode usar uma variedade de linguagens e ferramentas, incluindo Swift, Objective-C e Unity. Você também pode usar uma variedade de frameworks, como SpriteKit e SceneKit. O desenvolvimento de jogos para iOS é uma ótima maneira de criar experiências envolventes e divertidas para seus usuários. Se você está interessado em desenvolver jogos para iOS, há uma série de recursos disponíveis para ajudá-lo a começar, incluindo tutoriais, cursos e comunidades online. Veja as principais habilidades que você precisa desenvolver:
- Conhecer as linguagens e ferramentas usadas para desenvolver jogos para iOS, como por exemplo C# e Unity;
- Explorar recursos de desenvolvimento de jogos, como o uso de gráficos 3D e animação.

Habilidade Auxiliar: Infraestrutura e boas práticas

- [ ] **Git e GitHub - Fundamentos**:
  - Git é um sistema de controle de versão distribuído gratuito e de código aberto projetado para lidar com tudo, desde projetos pequenos a muito grandes com velocidade e eficiência.
  - GitHub é um serviço de hospedagem para desenvolvimento de software e controle de versão usando Git.
  - Criar um repositório
  - Clonar um repositório
  - Fazer commit, push e pull de e para o repositório
  - Reverter um commit
  - Criar branches e pul requests
  - Lidar com merge e conflitos
- [ ] **HTTP - Fundamentos**:
  - HTTP significa Hyper Text Transfer Protocol. A comunicação entre computadores cliente e servidores web é feita enviando solicitações HTTP e recebendo respostas HTTP.
  - Entender a diferença dos verbos HTTP
  - Testar os requests e ver os status codes no navegador
  - Saber fazer uma requisição HTTP na linha de comando com WGET
  - Baixar uma imagem com WGET
  - Fazer um post
- [ ] **JSON**:
  - JSON significa JavaScript Object Notation (notação de objeto JavaScript). É um formato de texto para armazenar e transmitir dados.
  - Criar um objeto
  - Transformar um objeto em uma string

- Transformar uma string em objeto
- Manipular um objeto

- [ ] **Linha de comando - Fundamentos**:

- CLI é um programa de linha de comando que aceita entradas de texto para executar funções do sistema operacional.
- Conhecer os principais comandos

- [ ] **Cloud - Fundamentos**:

- Cloud, ou computação em nuvem é a distribuição de serviços de computação pela Internet usando um modelo de preço pago conforme o uso. Uma nuvem é composta de vários recursos de computação, que abrangem desde os próprios computadores (ou instâncias, na terminologia de nuvem) até redes, armazenamento, bancos de dados e o que estiver em torno deles. Ou seja, tudo o que normalmente é necessário para montar o equivalente a uma sala de servidores, ou mesmo um data center completo, estará pronto para ser utilizado, configurado e executado.
- Conhecer a diferença entre IaaS, PaaS e SaaS
- Conhecer os maiores provedores de cloud
- Especializar-se em algum provedor

- [ ] **SOLID**:

- O Solid possui cinco princípios considerados como boas práticas no desenvolvimento de software que ajudam os programadores a escrever os códigos mais limpos, separando as responsabilidades, diminuindo acoplamentos, facilitando na refatoração e estimulando o reaproveitamento do código.

- [ ] **Clean Code**:

- Aplicar técnicas simples que visam facilitar a escrita e leitura de um código
- Refatorar seu código para que fique mais claro

- [ ] **Clean Architecture**:

- A Clean Architecture (Arquitetura Limpa) é uma forma de desenvolver software, de tal forma que apenas olhando para o código fonte de um programa, você deve ser capaz de dizer o que o programa faz.

- [ ] **Firebase**:

- O Firebase é uma plataforma de desenvolvimento de aplicativos Backend-as-a-Service (BaaS) que fornece serviços de backend hospedados, tais como banco de dados em tempo real, armazenamento em nuvem, autenticação, relatórios de falhas, aprendizado de máquina, configuração remota e hospedagem para seus arquivos estáticos.
- Entender como Instalar o Firebase
- Conhecer a documentação do Firebase
- Conhecer as ferramentas do Firebase disponíveis

Habilidade Auxiliar: UX & Design

- [ ] **iOS - Diretrizes de Interface Humana (HIG)**:

- As Diretrizes de Interface Humana contêm orientações e práticas recomendadas que podem ajudá-lo a criar uma ótima experiência para qualquer plataforma Apple. Veja as principais habilidades que você precisa desenvolver:
- Aprender os fundamentos das Diretrizes de Interface Humana (HIG) para iOS;
- Usar cores, fontes, layouts, ícones e animação de acordo com as diretrizes HIG.

- [ ] **Design System**:

- Um Design System (sistema de design) é uma coleção de componentes reutilizáveis, guiados por padrões claros, que podem ser colocados juntos para construir aplicações.
- Criar e manter bibliotecas que serão consumidas e usadas como padrão para a construção de um projeto
- Design tokens

- Estilos fundamentais
- Construção de componentes
- Microinterações
- Documentação

- [ ] **Sistemas de cores**:

- Definir uma paleta de cores que faça sentido para determinada interface

- [ ] **Como usar fontes**:

- Escolher a fonte mais adequada para determinado projeto

- [ ] **Design Responsivo**:

- Ajustar suas páginas para o tamanho da tela do usuário
- Media queries
- Conhecer o conceito de Mobile first

- [ ] **iOS - Acessibilidade**:

- A acessibilidade é a capacidade de todos os usuários de acessar e usar um aplicativo, independentemente de suas capacidades. Isso inclui pessoas com deficiências visuais, auditivas, motoras e cognitivas. Existem muitas maneiras de tornar um aplicativo acessível em iOS. Uma maneira é usar os recursos de acessibilidade do iOS, como VoiceOver, Zoom e Switch Control. Outra maneira é projetar o aplicativo de forma que seja fácil de usar por pessoas com deficiências. É importante tornar seus aplicativos acessíveis para que todos possam usá-los. Isso é bom para os negócios, pois pode aumentar sua base de clientes. Também é bom para a sociedade, pois pode ajudar a promover a inclusão e a igualdade. Veja as principais habilidades que você precisa desenvolver:
- Conhecer e utilizar os recursos de acessibilidade do iOS.

- [ ] **Figma - Fundamentos**:

- Figma é uma aplicação web colaborativa para design de interfaces. O conjunto de recursos do Figma se concentra na interface do usuário e no

design da experiência do usuário, com ênfase na colaboração em tempo real, utilizando uma variedade de editores de gráficos vetoriais e ferramentas de prototipagem.

- Criar layouts de páginas e componentes
