

iOS PREWORK

- [      INTRODUCTION      ] -



A portrait of Mihir Suvarna, a young man with dark hair and glasses, wearing a blue suit jacket, white shirt, and red tie. He is smiling and standing outdoors with trees and water in the background.

Mihir Suvarna

Sophomore @ UT Austin  
Majoring in CS, with focus on AI/ML

- [      iOS PREWORK: TIP CALCULATOR APP      ] -



## What's The Tip?

Designed to make finding a tip intuitive, and at the tip of your fingers. Created with an optional dark mode, tip slider, and locale-specific currency.

Big takeaway: why calculate a tip manually when an app can do it with one touch?

# SETTING UP XCODE & LEARNING TO USE SWIFT

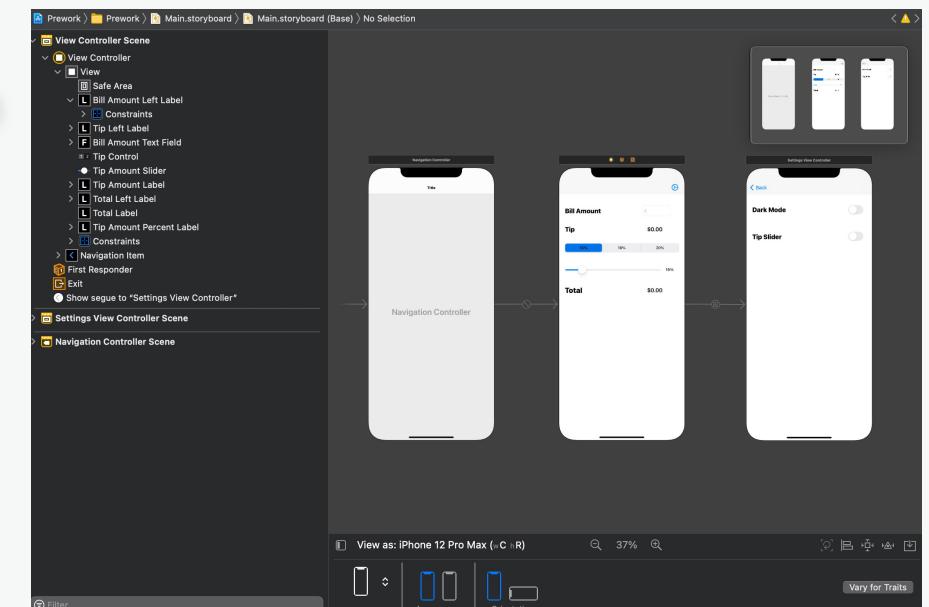


## HOW I DID IT

The first step of my pre-work consisted of setting up Xcode and learning to use Swift.

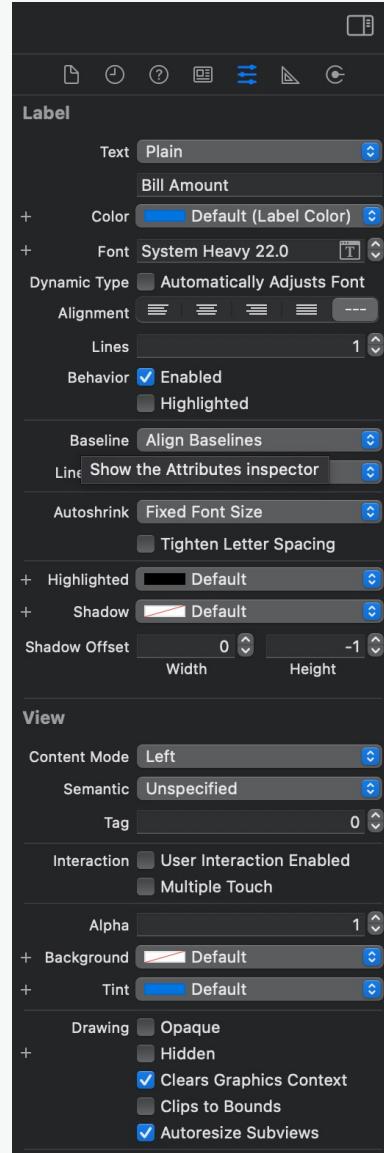
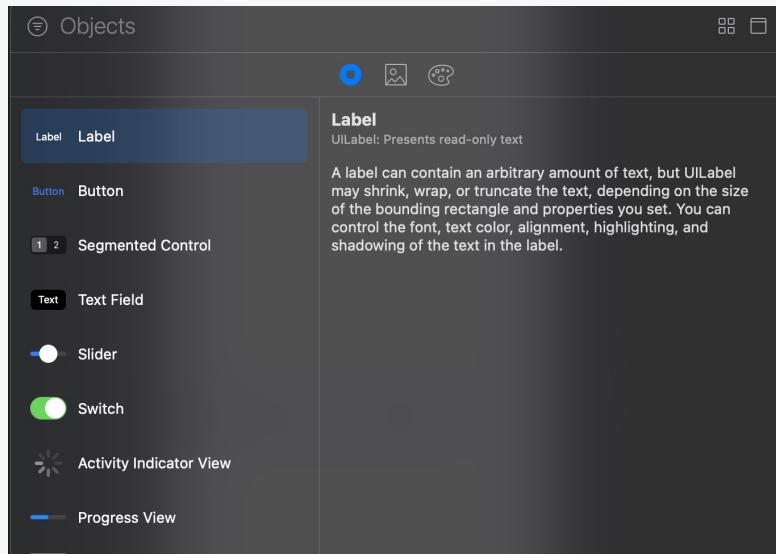
I needed to learn:

- Storyboard
- Connections to View Controller
- Outlets & Verifying Connections
- Functions, Structs



# Basic UI aspects + optional features

Used object library and attribute inspector.



Text Field/Decimal Pad

Responsive Labels

Tip Slider and Seg. Control

Dark Mode

# View Controllers

The screenshot shows the Xcode interface with the following details:

- Project Navigator:** Shows the project structure with files like AppDelegate.swift, SceneDelegate.swift, ViewController.swift, Main.storyboard, SettingsViewController.swift, Assets.xcassets, LaunchScreen.storyboard, and Info.plist.
- Editor:** The main editor area displays the `ViewController.swift` file. The code implements two `@IBAction` methods: `calculateTip` and `calculateTipSlider`. Both methods calculate tip and total amounts based on user input and slider values, using `NumberFormatter` for currency formatting.
- Search Bar:** At the top, it says "Prework | Build Prework: Succeeded | Yesterday at 4:04 PM".
- Log Navigator:** On the right, it shows a log entry from Mihir Suvarna: "Initial Commit" at 6/5/21, with commit hash `dcfbdb95`.

```
83 }
84
85 @IBAction func calculateTip(_ sender: Any) {
86     // Grab the bill amount
87     let bill = Double(billAmountTextField.text!) ?? 0
88
89     // Calculate tip and total using an array
90     let tipPercentages = [0.15, 0.18, 0.2]
91     let tipIndex = tipPercentages[tipControl.selectedSegmentIndex]
92     let tip = bill * tipIndex
93     let total = bill + tip;
94
95     // Format the total
96     let numberFormatter = NumberFormatter()
97     numberFormatter.numberStyle = .currency
98
99     // Update tip and total amount labels
100    tipAmountLabel.text = String(format: "$%.2f", total)
101    totalLabel.text = String(format: "$%.2f", total)
102    tipAmountLabel.text = numberFormatter.string(for: tip)
103    totalLabel.text = numberFormatter.string(for: total)
104
105    // Update the tipSlider with selected segment value
106    if( UserDefaults.standard.object(forKey: "tipSliderOn") != nil) {
107        if UserDefaults.standard.bool(forKey: "tipSliderOn") {
108            tipAmountSlider.setValue(Float(tipIndex), animated: true)
109            tipAmountPercentLabel.text = String(format: "%.2f%%", 100 * tipAmountSlider.value)
110        }
111    }
112
113 }
114
115 @IBAction func calculateTipSlider(_ sender: Any) {
116     tipAmountPercentLabel.text = String(format: "%.2f%%", 100 * tipAmountSlider.value)
117     let bill = Float(billAmountTextField.text!) ?? 0
118
119     // Calculate tip and total using an array
120     let tip = bill * tipAmountSlider.value
121     let total = bill + tip;
122
123     // Format the total
124     let numberFormatter = NumberFormatter()
125     numberFormatter.numberStyle = .currency
```

## Connecting UI to Code

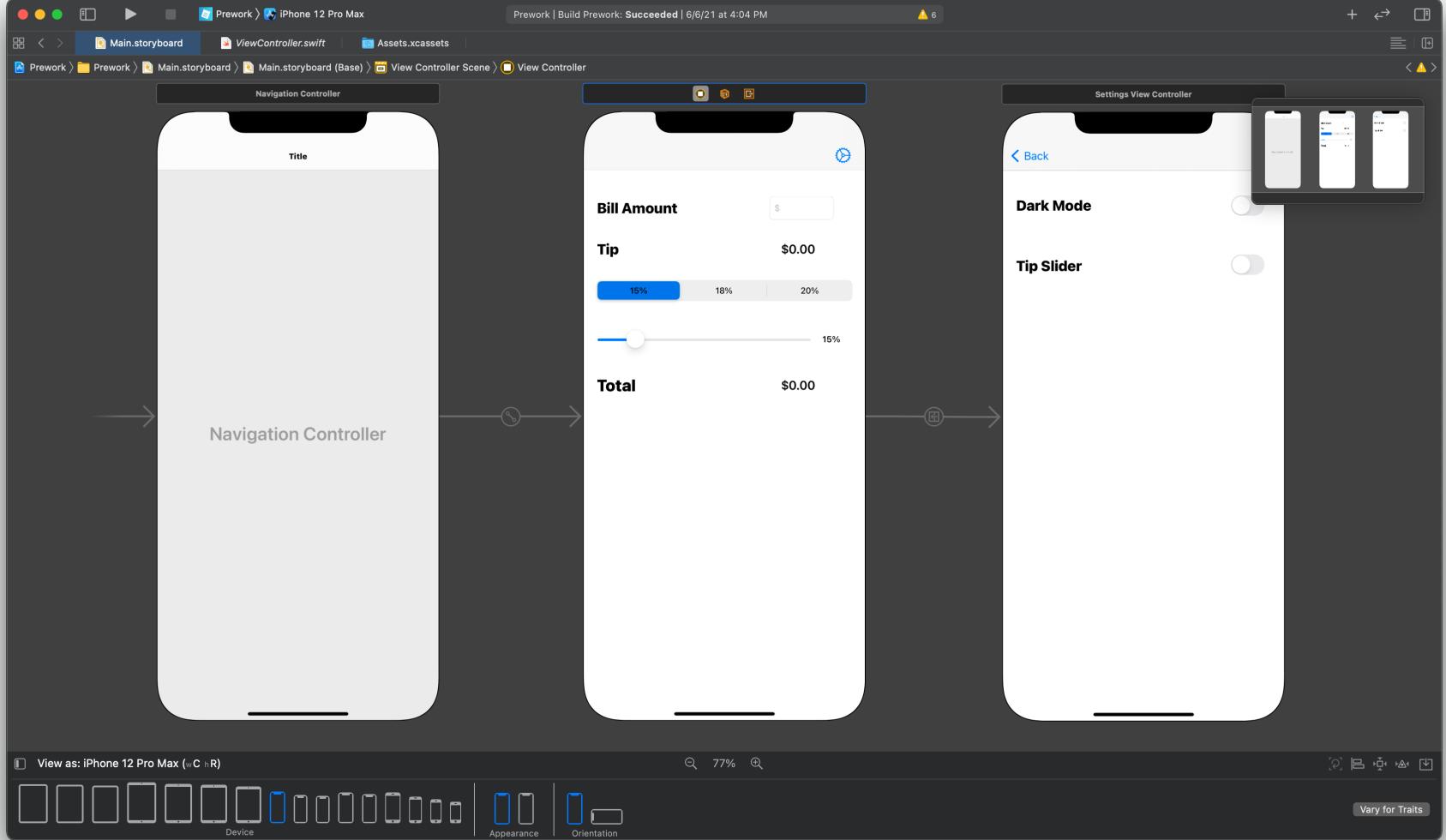
Once all objects are in place, I needed to use outlets and actions to make my UI functional.

Coded:

- Bill Text Field
  - Tip/Total ‘logic’
  - Slider Functionality

Added:

- Clean Dark UI
  - Currency Locale
  - Settings w/ Switches



## Navigation Controller

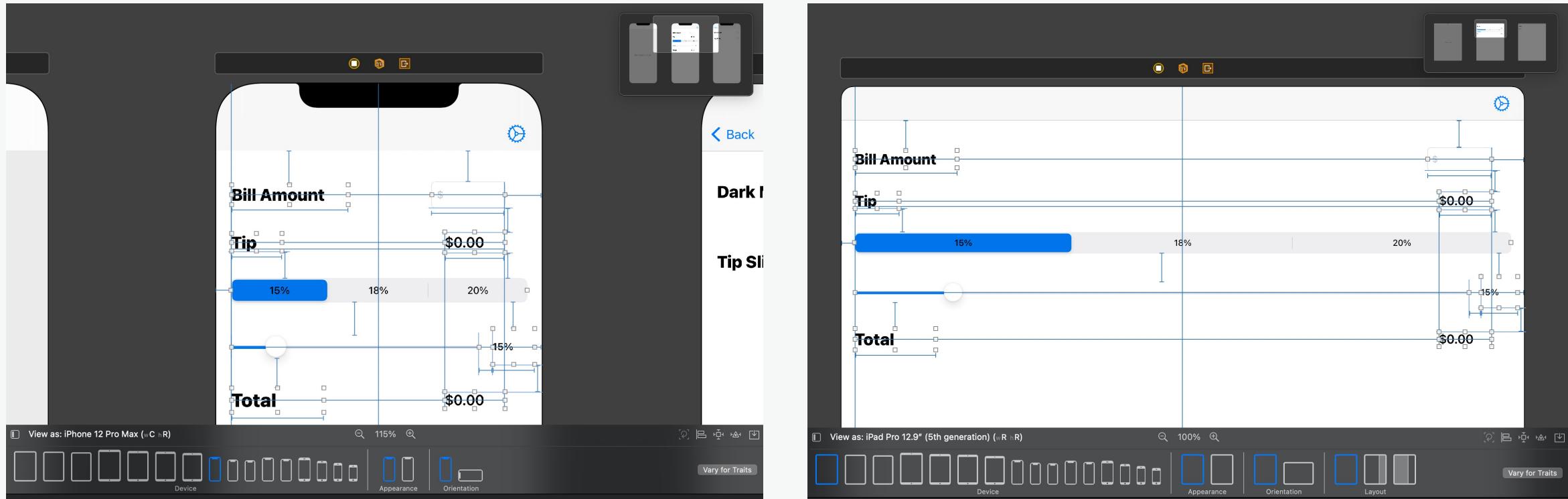
Simple top-bar navigation, settings icon included title and back button.

## Tip View Controller

Main UI, responsive elements, first responder.

## Settings View Controller

Optional settings screen, includes switches for tip slider & app-wide dark mode.



# Constraints: Love/Hate

Shown above are iOS constraints and iPadOS constraints for '*What's the Tip?*'.

Notice the # needed to ensure proper resizing across different devices; even more painful for watchOS & macOS.

With various screen sizes, constraints are a *must*. Adding them can be a pain.

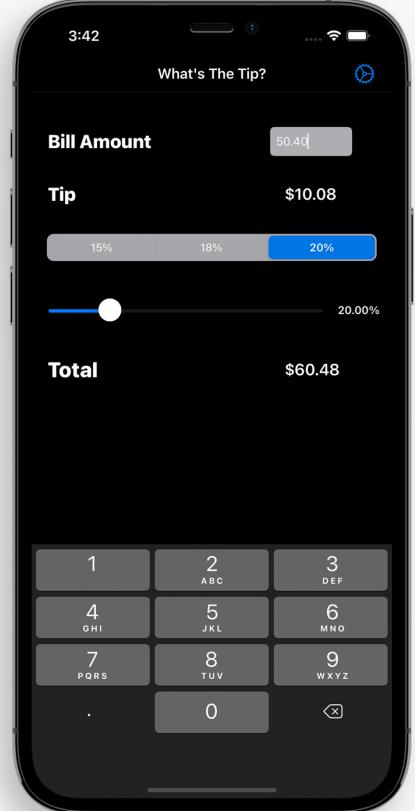
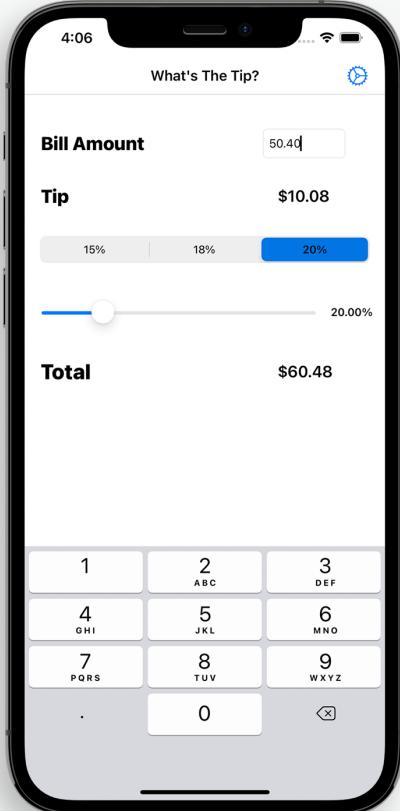
Xcode's auto-layout is not as reliable; manual adjustment is necessary.

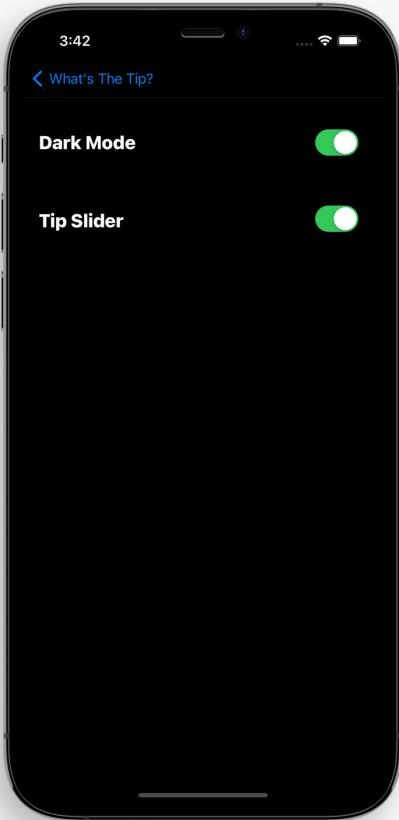
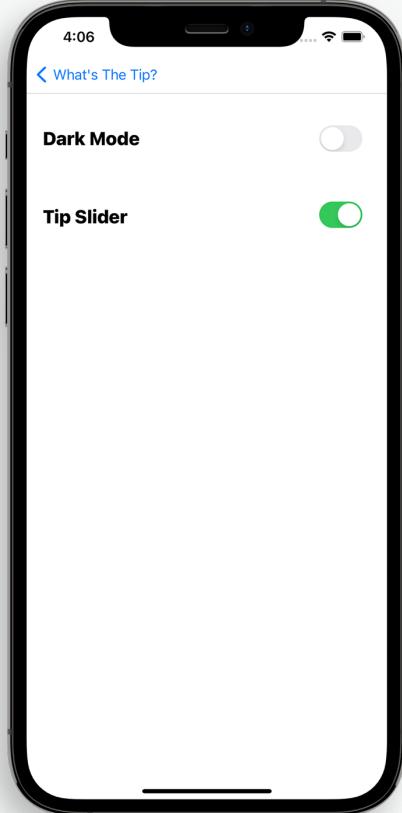
- [ D A R K   M O D E    ] -

## Dark Mode: An Interesting Obstacle

App-wide dark mode was quite difficult.

Many UI elements individually customized, special colors needed from Apple's color set.





- [ DARK MODE ] -

## Other View Controllers?

Had to add dark mode to settings view controller.

Think about scalability; can be *way* harder for more than just two view controllers.

Each UI has to be custom-designed; no easy Apple 'dark' switch exists.

- [        W H A T ' S    T H E    T I P ?    ] -

# iOS Preview

All UI elements in place, full product mockup.

Displayed on: iPhone 12 Pro Max





X

- [        W H A T ' S    T H E    T I P ?    ] -

# iPadOS Preview

Added support for iPadOS; here is all UI elements in place, full mockup.

Displayed on: 11-inch iPad Pro 3<sup>rd</sup> Generation

# Development Summary

**1 hr**

## DESIGN THE UI

Create a flexible design, main view.

**2.5 hrs**

## CONNECT AND CODE

Connect all elements and code a functional UI.

**3 hrs**

## OPTIONAL FEATURES

Implement other features (dark mode, tip slider, locale).

**1.5 hrs**

## PRODUCT REVIEW

Ensure product is complete and works on all devices.



THANK YOU.

QUESTIONS/COMMENTS?

WTT