**1. Project Overview**

**Purpose**:

Build **YouSnoozeYouWin**, an iOS app that helps users limit screen time by allowing them to lock specific apps on their device for specified durations. Users can unlock early by using skip coins (which can be purchased or earned through achievements). The goal is to create a fun yet psychologically persuasive way to curb screen addiction.

**Objectives**:

1. Provide a simple flow for selecting an app and scheduling a lock duration.

2. Persuade the user to stick to the lock through playful UI, minimalistic design, and small friction points (skip coins).

3. Gamify the experience so users feel a sense of accomplishment when they do not unlock an app prematurely.

**Target Audience**:

• Anyone looking to reduce phone/app usage.

• Users conscious about digital well-being and seeking a fun, modern solution.

**2. Features**

1. **App Lockbox & Search**

• Users can search among apps available on their device.

• The user adds chosen apps to the “lockbox.”

2. **Lock Scheduling**

• Users set a time interval (e.g., 1 hour, 3 hours, or until a specific time) during which the app is locked.

3. **Lock Reminder & Restriction Interface**

• When an app is locked, attempts to open it surface the YouSnoozeYouWin UI stating: “App locked until [time].”

• User is nudged to keep the lock or to “Snooze longer” (extend lock).

4. **Skip Coins & Unlock**

• Each user starts with 3 skip coins.

• Unlocking early consumes 1 skip coin.

• Purchasing additional coins costs $1 each (to keep the MVP simple) or in future increments (bulk deals, see below).

5. **Payment & Monetization**

• **In-app purchase**: Skip Coins for real money.

6. **UI & Gamification**

• Minimalistic but modern design.

• Friendly messages celebrating success, encouraging them to “Snooze.”

• Gentle guilt triggers if they break locks.

7. **Future Enhancements (Do not implement now)**

• **Advanced analytics**: Show how much screen time was saved, daily streaks.

• **Achievement-based coins**: Earn coins by maintaining streaks or referring friends.

• **Tiered Pricing**:

• 1 coin for $0.99

• 5 coins for $3.99 (20% savings)

• 10 coins for $6.99 (30% savings)

• **Visual Feedback**: Animations for awarding or spending coins.

**3. Requirements for Each Feature**

**3.1 App Lockbox & Search**

• **Functional Requirements**:

1. Users must be able to see a list of installed apps (within iOS constraints).

2. Users can add any app to the lockbox.

3. If the user tries to add the same app twice, show an error or block it.

• **Non-Functional Requirements**:

1. Search functionality should be near-instant (<200 ms).

2. The UI must remain responsive, even with many apps installed.

• **Dependencies**:

• iOS APIs for listing installed apps. Use Screen Time API to find which apps they’d like to track  
<https://developer.apple.com/documentation/screentime>

**3.2 Lock Scheduling**

• **Functional Requirements**:

1. A user can set a lock duration (e.g., 1 hour, 3 hours, or a user-defined end time).

2. The system must store the lock start time and end time.

3. The system must check if the current time is >= end time to determine if the lock is still active.

• **Data Fields**:

• lock\_start\_time: Date

• lock\_end\_time: Date

• app\_identifier: String (unique ID for each locked app)

• **Dependencies**:

• Local storage to store locked app info.

• System clock to compare current time with lock\_end\_time.

**3.3 Lock Reminder & Restriction Interface**

• **Functional Requirements**:

1. If the user attempts to open a locked app, display a blocking screen (or a popup) with the lock status.

2. Provide an option to “Extend Lock” or “Skip by paying.”

• **Non-Functional Requirements**:

1. The block screen should appear quickly (<500 ms).

2. The UI theme matches the minimalistic design guidelines.

• **Dependencies**:

• Possibly leveraging iOS notifications or custom URL handlers to intercept app opens.

• SwiftUI views or UIKit views for the “locked” overlay screen.

**3.4 Skip Coins & Unlock**

• **Functional Requirements**:

1. Each user starts with initial\_skip\_coins = 3.

2. Unlocking an app early deducts skip\_coins -= 1 from the user.

3. If skip\_coins == 0, prompt the user to buy more coins.

• **Data Fields**:

• skip\_coins: Int stored at the user level.

• user\_id: String or some unique user identifier.

• **Edge Cases**:

1. If skip\_coins < 1, block the unlock or open the purchase flow.

2. If the lock is already expired, do not consume coins.

**3.5 Payment & Monetization**

• **Functional Requirements**:

1. In-app purchase flow for skip coins.

2. purchase\_coins(quantity: Int) updates skip\_coins += quantity.

• **Dependencies**:

• Apple’s StoreKit / StoreKit2.

• Product IDs for coin purchases (e.g., com.yousnoozeyouwin.coins.single, com.yousnoozeyouwin.coins.bulk5, com.yousnoozeyouwin.coins.bulk10).

• **Variable Names**:

• coinPriceSingle = 1.00

• coinPriceBundle5 = 2.99

• coinPriceBundle10 = 5.99

**3.6 UI & Gamification**

• **Functional Requirements**:

1. Show a simple timer indicating how long until unlock.

2. Display subtle confetti or “winning” animation if the user keeps the lock until it expires.

• **Non-Functional Requirements**:

1. Minimal color palette, consistent typography, friendly messaging.

2. Use SwiftUI for dynamic, smooth animations.

**4. Data Models**

You may store these models locally.

**Relationships**:

• A User can have multiple LockedApp entries.

**Additional Considerations**

• **Privacy & Compliance**: Make sure you’re transparent about what data is collected (app usage data is sensitive).

• **App Store Review Guidelines**: Apple can be strict about apps that replicate or circumvent Screen Time. Frame this as a motivational, user-consent-based usage tracking tool rather than forcibly blocking usage.

• **Marketing**: The name “YouSnoozeYouWin” is catchy, but test if Apple’s guidelines are okay with referencing other apps usage in your marketing descriptions.