Design Document

HoopMaster

|  |  |
| --- | --- |
| Date | 07/09/2024 |
| Version | 1 |
| State | Inception Phase |
| Authors | Valentin Morisca & Victor Maftei |

# Introduction

In the following pages you will find a description of the software architecture for our project, detailing the design choices made, the constraints we operated under, and the principles we have followed.

# Architecture Constraints and Design Decisions

## Platform-Specific Considerations:

* **iOS Ecosystem:** Swift and SwiftUI are tightly integrated within the Apple ecosystem. Utilizing and developing with Swift ensures that the application can fully use platform-specific devices such as the Apple Watch, or specific accessibility optimizations such as VoiceOver, which can assist different users.
* **Performance Optimization:** Swift’s performance is optimized for Apple’s hardware, making it more than ideal for creating resource-intensive applications such as AR (Augmented Reality).
* **Smooth Animations with SwiftUI:** SwiftUI provides declarative syntax, which makes it easier to create smooth, responsive, and user-friendly UIs, which is crucial for providing an enjoyable user experience.

## Development Speed

* **Declarative UI with SwiftUI:** SwiftUI’s declarative syntax significantly speeds up the development process, allowing us to describe the UI and its behavior in a fast-paced manner.
* **Hot Reload:** SwiftUI’s live preview and Xcode’s hot-reloading reduce iteration times, making the development more efficient.
* **SwiftUI compared to Flutter:** Flutter can also allow iOS development, but in general lacks the speed and different features that enhance the development process, and the user experience. Overall, SwiftUI integrates better within Apple devices (in our case, iPhone) than Flutter. Another information that has to be pointed out is that Flutter allows for cross-platform development (iOS and Android) with a single codebase, making it more cost-effective for multiple platform applications. However, as agreed with the stakeholders, the development platform will be iOS, so SwiftUI is the clear choice for this application.

## UI/UX Considerations:

* **Native Look and Feel:** SwiftUI seamlessly integrates with the other iPhone apps, thus giving a native feel to the users.