# Proposal for LED Control iOS App using SwiftUI and Arduino

## Introduction

This proposal outlines the development of an iOS application in SwiftUI designed to control an LED connected to an Arduino over the internet. The app will offer functionalities such as turning the LED on/off, making it blink, switching colors, and providing visual feedback and notifications to the user. The app will support both light and dark modes and will feature an animation to reflect the current state of the LED.

## Project Overview

### Objectives

- **LED Control**: Enable the user to turn the LED on and off.  
- **Blink Functionality**: Allow the user to make the LED blink.  
- **Color Switching**: Provide options to change the LED color.  
- **User Feedback**: Display animations based on the LED's state and send notifications to inform the user of actions taken.  
- **Light/Dark Mode**: Support both light and dark mode themes.

### Technology Stack

- **iOS Development**: Swift, SwiftUI  
- **Network Communication**: URLSession  
- **Microcontroller**: Arduino  
- **LED** **Control**: Arduino code to handle requests and control the LED  
- **User** **Notifications**: UserNotifications framework

## Features

### 1. LED Control

The app will have buttons to turn the LED on and off. These buttons will send HTTP requests to the Arduino's IP address, which will process these requests and control the LED accordingly.

### 2. Blink Functionality

A button will allow the user to make the LED blink at a preset interval. This will involve sending a specific request to the Arduino, which will handle the blinking logic.

### 3. Color Switching

The app will provide a color picker to change the LED's color. This functionality will require sending RGB values to the Arduino, which will adjust the LED's color.

### 4. User Feedback

- **Animation**: An animation in the center of the main UI will visually represent the LED's state (e.g., on, off, blinking, color change).  
- **Notifications**: The app will display notifications confirming the user's actions (e.g., "LED turned on", "LED is blinking", "LED color changed to red").

### 5. Light/Dark Mode

The app will adapt its appearance based on the system's light or dark mode settings, providing a seamless user experience.

## UI/UX Design

### Main Screen

- **Animation** **View**: Centered on the screen to show the current state of the LED.  
- **Control** **Buttons**: Placed at the bottom, including "Turn On", "Turn Off", "Blink", and "Change Color".

### Secondary Screens

- **Color** **Picker**: A dedicated screen or modal to select the LED color.

### Notifications

Inform the user about the success or failure of their actions with timely notifications.

## Technical Implementation

### Network Requests

Use `URLSession` to send HTTP GET/POST requests to the Arduino's IP address. The Arduino will be programmed to listen for these requests and control the LED accordingly.

### Arduino Code

The Arduino will run a web server, listening for specific endpoints (e.g., `/on`, `/off`, `/blink`, `/color`). Upon receiving requests, it will execute the corresponding actions on the LED.

### SwiftUI Animation

Create custom SwiftUI animations to represent the LED's state changes. Use `Lottie` or built-in SwiftUI animation capabilities to create smooth and visually appealing animations.

### Notifications

Use the `UserNotifications` framework to create and schedule notifications. Ensure permissions are requested and handled appropriately.

## Project Timeline

### Phase 1: Planning and Design (1 week)

- Define requirements and create wireframes.  
- Set up project structure in Xcode.

### Phase 2: Basic Functionality (2 weeks)

- Implement basic UI components in SwiftUI.  
- Set up network communication with Arduino.  
- Develop basic LED control features (on/off).

### Phase 3: Advanced Features (3 weeks)

- Implement blink and color change functionalities.  
- Develop animations and integrate them with LED states.  
- Add light/dark mode support.

### Phase 4: Notifications and Testing (1 week)

- Implement user notifications.  
- Conduct thorough testing and debugging.  
- Gather user feedback and make necessary adjustments.

### Phase 5: Finalization and Deployment (1 week)

- Finalize the app, ensuring all features work as intended.  
- Prepare for submission before deadline.

### Conclusion

This iOS app will provide a user-friendly interface to control an LED connected to an Arduino, offering a range of functionalities from basic on/off control to advanced features like blinking and color changes. The app will enhance user experience through animations, notifications, and support for light/dark modes. With a structured timeline and clear objectives, this project aims to deliver a high-quality, functional app within the proposed timeframe.