Studocu Mobile Automation Project Documentation

1. Introduction

This document provides a detailed overview of the mobile automation project aiming to automate testing for Studocu app.

2. Analysis and Requirements

Application Overview

Studocu app allow student to share study notes, courses etc. It has features to search books , you can add book to wishlist . Also book can be added to currently reading which you can access & track.

Test Scenarios and Use Cases

- **Scenario 1:** Search for Books with title
- **Scenario 2:** Add books to Currently reading
- **Scenario 3:** Add/remove books to wishlist
- **Scenario 4:** Open book and verify book author , summary , book name , year etc details.
- .
-

Non-Functional Requirements

- **Performance:** Verify app handle projected load volume(load testing/stress testing).
- Verify response time of app is as expected.
-
- **Security:** Verify mobile app withstand any brute force attack

Device and Platform Compatibility

[List of supported devices, OS versions, and browsers.]

3. Task Planning

Project Timeline

[Tentative timelines] (Responsible: [QA Lead], Timeline: [Dates])

Resource Allocation

- **Automation Engineers:** [<Name>]
- **Testing Devices:** [List Android/IOS devices supported devices]

4. Technical Solution and Tools Selection

- **Appium** with **WebDriverIO:** Chosen for its cross-platform compatibility and robust capabilities in automating mobile apps.
- **JavaScript:** Language of choice due to its ease of use and extensive support for WebDriverIO.
- Developed hybrid testing framework setup using page object model design pattern.
- Tools used (e.g., Mocha, Visual Studio Code,Appium inspector, Appium Desktop, github).

5. Design Patterns and Best Practices

Design Patterns

- **Page Object Model (POM):** Chosen for better test maintenance and readability.

6. Implementation

Test Scripts

- Search Scenario
- Book Details scenario
- Add/Remove book to wishlist
- Add/Remove book to Currently reading
- Currently reading sceanrio

- Test data is used from constant class for data-driven approach .
- Verify ,Interact , helper classes are used for generic reuable components & assertion.

7. Execution and Reporting

Running Tests

- Instructions for running tests on different devices/simulators/emulators are added in Readme.MD file
- Framework can be integrated with CI/CD pipeline

Reporting

- Deafult reporter Spec is used , it shows detailed summary without adding any third party plugins.
- HTML based reporter plugins can be implemented.

8. Challenges and Lessons Learned

Challenges Faced

- **Challenge 1:** Building app binaries & finding locators of elements
- **Challenge 2:** App is very slow , it takes more than minute to show search results

Lessons Learned

- App should have added testID for automation .

- Framework can be optimised Parallel testing , running tests in cloud on device farm , fancy reporter to add.

9. Conclusion

- Automation framework is completed for android platform .
- As framework is ready we have to keep adding new application featues to regression suite with minimal efforts