Test Plan: Account Creation UI

Version 1.0

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description** | **Author** |
| 10/FEB/17 | 1.0 | Creation of Test Plan | Vaibhav Patil |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[1. Introduction 4](#_Toc473793178)

[1.1 Purpose 4](#_Toc473793179)

[1.2 Background 4](#_Toc473793180)

[1.3 Scope 4](#_Toc473793181)

[1.4 Project Identification 5](#_Toc473793182)

[2. Requirement for Test 5](#_Toc473793183)

[2.1 Functional Testing 5](#_Toc473793184)

[2.2 User Interface Testing 5](#_Toc473793187)

[3. Test Strategy 6](#_Toc473793191)

[3.1 Testing Type 6](#_Toc473793192)

[3.1.1 Functional testing 6](#_Toc473793193)

[3.1.2 User Interface 8](#_Toc473793194)

[3.2 Decision Table 9](#_Toc473793195)

[3.3 Tool 10](#_Toc473793196)

[4. Resources 10](#_Toc473793197)

[4.1 Roles 10](#_Toc473793198)

[4.2 Systems 11](#_Toc473793199)

[5. Project Milestone 11](#_Toc473793200)

[Appendix A 11](#_Toc473793201)

**Test Plan**

# Introduction

## Purpose

This Test Plan document for the Account Creation UI supports the following objectives:

* Explains the testing methodology for a mobile application - Account Creation UI.
* List of recommended requirements for test (Highly level)

Intended Audience

* The Project Manager
* Product Development team members
* Test Engineers

## Background

The account creation UI for the ABC app needs to be tested. The mobile account creation page is mainly responsible to let a new user to create an account. The user has to type in valid the username of his/her choice, an email address and a password and then hit Create Account button. If the all the fields are valid, then proceed with next steps.

The Account creation page is starting point of the part in application where a user wants to create a new account on the app. The main fields are the Username text field, where the user will type in a unique user name; an Email address text field to type in the email address that will link the username to the email address thus making account valid and credible; a password text field to put in the password to make it secure and safe.

## Scope

This test plan will mainly focus on Manual testing of the Account creation UI part of app by doing functional testing and UI testing. The main targets are to test the Username, email and password text field input and the error handling when an invalid or incorrect input is entered and stable and consistent UI across different devices and

Assumptions made:

* The text boxes are highlighted when a user is select it or is entering the text.
* At the end of the text box
  + A green check mark appears at the end of the text field if the value in the field is valid and/or available
  + A red error icon appears for the invalid text entry or a blank text field and when tapped displays the error message
* There are proper error messages displayed on the screen for invalid inputs in the respected text field.
* The tests related to use of the application in a different language other than that of English will not be covered in this test plan.

Any change it the color scheme or in the error messages and any UI changes to the Account Creation page should be updated to the tests too.

General Assumptions

* Performance testing is not considered for this estimation.
* The UI defects will be tracked through JIRA only and steps to simulate along with screenshots if appropriate.
* An error icon appears for the fields with invalid entry and gives a small detailed popup error text when tapped.
* Create account remains inactive till all the fields are filled correctly
* Any change it the color scheme or in the error messages and any UI changes to the Account Creation page and any defect fixes planned will be shared with Test Team prior to applying the fixes on the Test environment
* Any UI defect noticed should be reported with a snapshot attached to it.
* The network connection is perfect without any drop in network.
* The app supports iOS and Android on multiple devices.

## Project Identification

The table below identifies the documentation and availability used for developing the *test plan*:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Document  (and version / date)** | **Created or Available** | **Received or Reviewed** | **Author or Resource** | **Notes** |
| Requirements Specification | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Functional Specification | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Use-Case Reports | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Project Plan | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Design Specifications | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Prototype | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| User’s Manuals | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Business Model or Flow | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |
| Business Functions and Rules | 🞏 Yes 🞏 No | 🞏 Yes 🞏 No |  |  |

# Requirement for Test

The listing below identifies those items (use cases, functional requirements, non-functional requirements) that have been identified as targets for testing. This list represents what will be tested.

## Functional Testing

* Checking the data/text entry in the field.
* Checking of error messages generated with invalid inputs in the field.
* Checking of account creation button functionality with valid and invalid data.

## User Interface Testing

* Checking for the UI components.
* Checking of the UI components when device rotated.
* Checking of the actions on selecting the fields.
* Checking of a Loading/Processing icon/Activity Indicator if the process of account creation is initiated.

# Test Strategy

The Test Strategy presents the recommended approach to the testing of the software applications. The previous section on Test Requirements described what will be tested; this describes how it will be tested.

The main considerations for the test strategy are the techniques to be used and the criterion for knowing when the testing is completed.

In addition to the considerations provided for each test below, testing should only be executed using known, controlled databases, in secured environments.

## Testing Type

### Functional testing

The goals of these tests are to verify proper data acceptance, processing and the appropriate implementation of the input rules. This type of testing is based upon black box techniques, that is, verifying the application (and its internal processes) by interacting with the application via the GUI and analyzing the output (results).

#### Text Fields

The main aim of testing the Text field is to make sure the input that system gets from the user is accurate and valid

|  |  |
| --- | --- |
| Test Objective: | Checking the data/text entry in the field |
| Technique: | * Text field is highlighted, with cursor blinking and respective placeholder disappears as soon as the user starts typing. * Enter valid and invalid input. |
| Completion Criteria: | * A green check mark appears at the end of the text field if the value in the field is valid and/or available * A red error icon appears for the invalid text entry or a blank text field and when tapped displays the error message. |
| Special Considerations: | * Create account button remains inactive for the whole time till all fields are filled with valid inputs * Valid Inputs for the fields:   1. User Name: A valid username must be      + Minimum 3 characters long.      + Maximum 30 characters long.      + Can contain a combination of characters like a-z, A-Z,0-9, and underscore (“\_”), without space. All characters apart from these should be considered invalid.      + Must be unique.   2. Email Address: A valid email address must be * In the format Local-Part@domain-part.com * “@” Symbol should be there between the local part and the domain part   E.g.: jack@gmail.com   * + - Local part and domain part Can contain a combination of characters like a-z, A-Z,0-9,”.” and underscore (“\_”), without space. All characters apart from these should be considered invalid.   E.g.: jack.sparrow\_22@gmail.com   * Domain name may contain a hyphens (“-”) but it should be between alphanumeric characters.   E.g.: jack.sparrow\_22@g-mail.com   * Minimum one dot(.) should be there in the domain part * Maximum of two dots(.) are allowed.   E.g.: jack.sparrow\_22@husky.neu.edu   * Total length of email address should not be greater than 30 characters   1. Password      + Minimum 8 characters long.      + Maximum 30 characters long.      + Can contain a combination of characters like a-z, A-Z,0-9, and (\_, @, ! ), without space. All characters apart from these should be considered invalid.      + Cases sensitive |
| Related Test Cases |  |

#### Error Messages

An error icon appears when the user enters invalid inputs. When the user presses the error Icon, an error message pops up showing more details about the respective text field. The error message may differ in different situation for the same text field.

|  |  |
| --- | --- |
| Test Objective: | * Checking of error messages generated with invalid inputs in the field. |
| Technique: | * Text field is highlighted, with cursor blinking and respective placeholder disappears as soon as the user starts typing. * Enter an invalid input. * Click on the error symbol to know more about it. |
| Completion Criteria: | * A red error icon appears for the invalid text entry and when tapped displays the error message. * The error message should correspond the invalid entry field indicating the error caused |
| Special Considerations: | * A text field can generate multiple error messages depending on different invalid input given by the uses |
| Related Test Cases |  |

#### Create Account button

|  |  |
| --- | --- |
| Test Objective: | * Checking of account creation button functionality with valid and invalid data. |
| Technique: | * Text field is highlighted, with cursor blinking and respective placeholder disappears as soon as the user starts typing. * Enter valid and invalid inputs. |
| Completion Criteria: | * The “Create Account” button becomes active if all the input fields are filled with valid entry * The “Create Account” button remains inactive if all the input fields are:   + Empty   + Partially filled   + Filled with invalid text |
| Special Considerations: | The Activity indicator icon should be displayed once the Create Account button is displays and make the whole screen unresponsive to the user input except double taping on the activity indicator |
| Related Test Cases |  |

### User Interface

User Interface testing verifies a user’s interaction with the Account Creation page. The goal of UI Testing is to ensure that the User Interface provides the user with the appropriate access and navigation through the functions of the applications. In addition, UI Testing ensures that the objects within the UI function as expected and conform to corporate or industry standards.

#### UI components

|  |  |
| --- | --- |
| Test Objective: | * Checking for the UI components. |
| Technique: | * When on the Account creation UI page, check if all the elements are loaded and displayed as designed in the wireframe. |
| Completion Criteria: | * All UI components are loaded and displayed as designed in the wireframe. |
| Special Considerations: | The UI components should be consistent and same across different devices and platforms (Android and iOS)  All images and labels present on the Account Creation page are not clickable |
| Related Test Cases |  |

#### Device Rotation

|  |  |
| --- | --- |
| Test Objective: | * Checking for the UI components when device Rotated. |
| Technique: | * When on the Account creation UI page, rotate device by 90°, 180° and 270° in both clockwise and counter clockwise direction (Landscape and Portrait Mode) |
| Completion Criteria: | * All UI components are loaded and displayed as designed in the wireframe in both portrait and landscape mode |
| Special Considerations: | The UI components should be consistent and same across different devices and platforms (Android and iOS) |
| Related Test Cases |  |

#### Active text field Indicator

|  |  |
| --- | --- |
| Test Objective: | * Checking of the actions on selecting the fields. |
| Technique: | * When a user hits the create Account button, Activity indicator appears. This indicate that the process of account creation is being carried on and the user has to wait till it finishes to go to next step. * It disables user input to button and text fields so as to avoid hitting of server with the same call and thus causing the errors |
| Completion Criteria: | * Active field is highlighted and text can be entered in the field |
| Special Considerations: | * The UI components should be consistent and same across different devices and platforms (Android and iOS) * When the system keyboard pops up, the active field should not be covered by the keyboard. |
| Related Test Cases |  |

#### Activity Indicator

An activity indicator is a very important UI component. It shows the system is processing the request that was just made. It helps to prevent getting multiple calls for the same service from the user. It is necessary that the activity indicator stays up on the top of the Account creation view so as to disable the user to make any further requests by clicking the Create Account button multiple times

|  |  |
| --- | --- |
| Test Objective: | * Checking of the actions on selecting the fields. |
| Technique: | * The active text field is highlighted when selected and system keyboard pops up with the cursor blinking in the active field and the place holder disappears as the user starts to type in. * To go to the next field a user taps in to the field UI component and it becomes active. |
| Completion Criteria: | * Activity indicator appears on the middle of the screen as an overlay, disabling all the text fields and buttons. |
| Special Considerations: | * The UI components should be consistent and same across different devices and platforms (Android and iOS) * When the system keyboard pops up, the active field should not be covered by the keyboard. |
| Related Test Cases |  |

## Decision Table

A decision table gives a glimpse of the different cases possible for a combination of input for a logic and their possible outcomes.

|  |  |  |  |
| --- | --- | --- | --- |
| **Username** | **Email address** | **Password** | **Outcome** |
| Valid | Valid | Valid | Successful |
| Valid | Valid | Invalid | Unsuccessful |
| Valid | Invalid | Valid | Unsuccessful |
| Invalid | Valid | Valid | Unsuccessful |
| Valid | Invalid | Invalid | Unsuccessful |
| Invalid | Valid | Invalid | Unsuccessful |
| Invalid | Invalid | Valid | Unsuccessful |
| Invalid | Invalid | Invalid | Unsuccessful |

## Tool

The list of tools and their respective vendors and version numbers

|  |  |  |  |
| --- | --- | --- | --- |
|  | Tool | Vendor/In-house | Version |
| Test Management | TestRails | Gurock | 5.0 |
| Defect Tracking | Jira | Atlassian | 6.4 |

# Resources

This section presents the recommended resources for the ABC app project, their main responsibilities and their knowledge or skills set

## Roles

This table shows the staffing assumptions for the project

|  |  |  |
| --- | --- | --- |
| Human Resources | | |
| Worker | Minimum Resources Recommended  (number of full-time roles allocated) | Specific Responsibilities or Comments |
| Test Manager,  Test Project Manager |  | Provides management oversight.  Responsibilities:   * provide technical direction * acquire appropriate resources   provide management reporting |
| Mobile QA Tester |  | Provides management oversight.  Responsibilities:   * Buffer for the engineering teams: handle bug reports from employees and users * Write and execute test plans * Perform manual testing of our iOS and Android apps * Find and report bugs and defects |
| Test System Administrator |  | Ensures test environment and assets are managed and maintained.  Responsibilities:   * administer test management system * install and manage access to test systems |
| Database Administrator, Database Manager |  | Ensures test data (database) environment and assets are managed and maintained.  Responsibilities:   * administer test data (database) |

## Systems

The following table sets forth the system resources for the testing project.

|  |  |
| --- | --- |
| System Resources | |
| Resource | Name / Type |
| Database Server |  |
| —Network or Subnet | TBD |
| —Server Name | TBD |
| —Database Name | TBD |
| Test Repository |  |
| —Network or Subnet | TBD |
| —Server Name | TBD |
| Test Development PC's | TBD |

# Project Milestone

|  |  |  |  |
| --- | --- | --- | --- |
| **Milestone Task** | **Effort** | **Start Date** | **End Date** |
| Plan Test |  |  |  |
| Design Test |  |  |  |
| Implement Test |  |  |  |
| Execute Test |  |  |  |
| Evaluate Test |  |  |  |

Appendix A **Project Tasks**

Below are the test-related tasks:

• Plan Test

* + identify requirements for test
  + assess risk
  + develop test strategy
  + identify test resources
  + create schedule
  + generate Test Plan

• Design Test

- prepare workload analysis

- identify and describe test cases

- identify and structure test procedures

- review and assess test coverage

• Implement Test

* + record or program test scripts
  + identify test-specific functionality in the Design and Implementation Model
  + establish external data sets

• Execute Test

- execute Test procedures

- evaluate execution of Test

- recover from halted Test

- verify the results

- investigate unexpected results

- log defects

• Evaluate Test

- evaluate Test-case coverage

- evaluate code coverage

- analyze defects

- determine if Test Completion Criteria and Success Criteria has been achieved