**Test Plan for Conjugo:**

I Introduction

The test plan outlines the testing strategy and approach for the Conjugo app. The goal is to ensure the app functions as expected, providing an effective and user-friendly experience for users. The testing will cover various aspects of the app, including navigation, game functionality, vocabulary display, verb-related information, settings, error handling, performance, compatibility, and usability.

II Test strategy

1. Tab Navigation:
   1. Verify that all tabs are displayed correctly.
   2. Test switching between different tabs.
   3. Ensure the active tab is highlighted properly.
2. GameScreen:
   1. Test the generation of random verbs for conjugation.
   2. Verify that the user input is accepted and processed correctly.
   3. Check if the app provides appropriate feedback based on the user's input.
   4. Test the streak and highest streak functionality.
   5. Verify that the UI is responsive and updates in real-time.
3. VocabScreen:
   1. Test the display of vocabulary words.
   2. Verify that the UI is properly formatted and easy to navigate.
   3. Test any filtering or sorting functionality.
   4. Ensure that the app handles a large number of vocabulary words efficiently.
4. VerbsScreen:
   1. Test the display of verb-related information.
   2. Verify that the UI is properly formatted and easy to navigate.
   3. Test any filtering or sorting functionality.
   4. Ensure that the app handles a large number of verbs efficiently.
5. SettingsScreen:
   1. Test the selection and modification of tenses.
   2. Verify that the selected tenses are updated correctly.
   3. Test saving and loading of selected tenses using AsyncStorage.
   4. Ensure that the app handles different scenarios, such as selecting multiple tenses or deselecting all tenses.
6. Error Handling:
   1. Test the error UI when the Octicon library is not available.
   2. Verify that an appropriate error message is displayed when deselecting last tense in SettingsScreen.
7. Performance:
   1. Test the app's performance on different devices and network conditions.
   2. Verify that the app loads quickly and responds promptly to user interactions.
8. Compatibility:
   1. Test the app on different screen sizes.
   2. Verify that the UI adapts properly to different devices.
   3. Test the app on both Android and iOS platforms.
9. Usability:
   1. Perform usability testing to ensure a smooth and intuitive user experience.
   2. Test the app with a group of users and gather feedback on usability issues or areas for improvement.
   3. Implement any necessary changes based on user feedback.

III Justification of the strategy

The chosen testing strategy covers a wide range of functionality and aspects of the Conjugo app. By testing navigation, game functionality, vocabulary and verb display, settings, error handling, performance, compatibility, and usability, we can ensure that the app meets the requirements and provides a seamless user experience. The strategy includes both functional and non-functional testing, addressing different scenarios and potential issues that may arise during app usage.

IV Test Environment

* Expo app for iPhone on iPhone 11
* Android Studio Android device simulator/virtual machine

The test environment consists of the Expo app running on an iPhone 11 device and the Android Studio Android device simulator/virtual machine. This allows for testing on both iOS and Android platforms, covering a wider range of devices. The Expo app provides an easy way to run and test React Native applications on iOS devices, while the Android Studio simulator/virtual machine enables testing on Android devices.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test Log** | | | | | | |
| **Test case** | **Requirement no. / Item being tested** | **Test Data / how to test** | **Expected Result** | **Actual Result** | **Test Outcome** | **Comments** |
| 1a | Tab Navigation - Tabs displayed properly. | Start the app and check the tabs | Tabs displayed | Tabs displayed | OK | - |
| 1b | Tab Navigation - Test switching between different tabs. | Switch from tab to tab | Tabs switch properly | Tabs switch properly | OK | - |
| 1c | Tab Navigation - Ensure the active tab is highlighted properly. | See if chosen tab is highlighted | Tab is highlighted | Tab is highlighted | OK | - |
| 2a | GameScreen - Test the generation of random verbs for conjugation. | Open the game screen and generate new verb using the submit button | Verb is generated along with any other variables | OK | OK | - |
| 2bc | GameScreen:  Verify that the user input is accepted and processed correctly.  Check if the app provides appropriate feedback based on the user's input. | Input some text and see if the app behaves properly when a correct, incorrect, an almost correct answer is submitted | As described | OK | OK | Proper testing of this functionality can take a lot of time, since in the current version there are 100 verbs, 19 different tense-mood pairs and 6 performers for almost every tense, except for two imperative tenses with 5 performers each. That gives a bit more than 11,000 cases to test altogether. |
| 2d | GameScreen - Test the streak functionality. | Streak should be incremented with every correct and ‘almost correct’ answer and go back to zero with each mistake. | As described | OK | OK | - |
| 2e | GameScreen - Test the highest streak functionality | Highest streak should be incremented every time the last highest streak is exceeded.  If streak > highest streak then highest streak = streak | As described | OK | OK | - |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case** | **Requirement no. / Item being tested** | **Test Data / how to test** | **Expected Result** | **Actual Result** | **Test Outcome** | **Comments** |
| 3a | VocabScreen - Test the display of vocabulary words. | Open the vocabulary tab and select a part of speech. | A list of words along with translations | OK | OK | - |
| 3b | VocabScreen - Verify that the UI is properly formatted and easy to navigate. | See if data is readable and within the bounds of the screen, if navigation buttons lead to lists of words for each part of speech and if it’s possible to return to the previous screen | As described | OK | OK | - |
| 3c | VocabScreen - Test any filtering or sorting functionality. | See if filtering/sorting works | Filtering functionality lets filter and render the filtered list on given requirements, sorting sorts the list based on chosen requirements | Not implemented yet | N/A | To be implemented |
| 3d | VocabScreen - Ensure that the app handles a large number of vocabulary words efficiently. | See if there are any performance issues while loading the list of words | No performance issues | OK | OK | FlatList element doesn’t render all of data at once, so this is covered by the system. |
| 4a | VerbsScreen - Test the display of verb-related information. | See if verb-related info is displayed correctly | As described | Not implemented yet | N/A | - |
| 4b | VerbsScreen - Verify that the UI is properly formatted and easy to navigate. | See if data is readable and within the bounds of the screen, if list entries are pressable and navigate to conjugation tables | As described | Conjugation tables not implemented | OK except for conjugation tables | - |
| 4c | VerbsScreen - Test any filtering or sorting functionality. | See if filtering/sorting works | Filtering functionality lets filter and render the filtered list on given requirements, sorting sorts the list based on chosen requirements | Not implemented yet | N/A | To be implemented |
| 4d | VerbsScreen - Ensure that the app handles a large number of verbs efficiently. | See if there are any performance issues while loading the list of words | No performance issues | OK | OK | FlatList element doesn’t render all of data at once, so this is covered by the system. |
| 5a | SettingsScreen - Test the selection and modification of tenses. | See if entries in SettingsScreen are pressable and if checkboxes are checked/unchecked then | Checkboxes switch their boolean state with every press | OK | OK | At least one checkbox has to remain checked therefore unchecking the only checked box is not possible |
| 5b | SettingsScreen - Verify that the selected tenses are updated correctly. | SettingsScreen contains multiple checkboxes, check some of them to see if the generated content in GameScreen changes accordingly | GameScreen generated mood-tense pairs include only those chosen in the SettingsScreen | OK | OK | - |
| 5c | SettingsScreen - Test saving and loading of selected tenses using AsyncStorage. | AsyncStorage should hold an array of chosen mood-tense pairs | AsyncStorage holds an array intact after restarting the application | Not implemented yet | N/A | AsyncStorage is used only for highestStreak at this point |
| 5d | SettingsScreen - Ensure that the app handles different scenarios, such as selecting multiple tenses or deselecting all tenses. | Select many different sets of mood-tense pairs, including all of one mood, each single mood-tense pair, all mood-tense pairs, random 5 mood-tense pairs | Mood-tense pairs are generated accordingly | OK | OK | - |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Test case** | **Requirement no. / Item being tested** | **Test Data / how to test** | **Expected Result** | **Actual Result** | **Test Outcome** | **Comments** |
| 6a | Error Handling - Test the error UI when the Octicon library is not available. | Corrupt the Octicon library link | Error message shows up | OK | OK | Eventually should switch to an alternative instead of showing an error message |
| 6b | Error Handling - Verify that an appropriate error message is displayed when deselecting last tense in SettingsScreen | Deselect the last selected tense in SettingsScreen | Error message shows up (popup) | OK | OK | - |
| 7a | Performance - Test the app's performance on different devices. | Test the app on iPhone’s Expo and 3 different virtual android devices (Google Pixel, Samsung, Sony) | App is usable in all cases | OK | OK | - |
| 7b | Performance - Verify that the app loads quickly and responds promptly to user interactions. | Check if screens load in a reasonable time | Every screen should take no more than half a second to load | OK | OK | - |
| 8a | Compatibility - Test the app on different screen sizes. | Test the app on virtual android devices with screen sizes ranging from ~4.5” to ~6.5” | App is usable on those screens | OK | OK | - |
| 8b | Compatibility - Verify that the UI adapts properly to different devices. | See if UI elements like the keyboard, menu bar work well with other elements if UI | They do | OK | OK | - |
| 8c | Compatibility - Test the app on both Android and iOS platforms. | Test on expo and android studio | App works on both | OK | OK | - |
| 9a | Usability - Perform usability testing to ensure a smooth and intuitive user experience. | Let other party test the app | Intuitive and smooth UX | OK | OK | Two friends tested the app, decided it’s easy to use |
| 9b | Usability - Test the app with a group of users and gather feedback on usability issues or areas for improvement. | Gather a group of potential users and let them test the app and look for issues | Gathered information to move forward with development | The app was not tested this way yet | N/A | So far I’ve not managed to find too many people eager to test the app |
| 9c | Usability - Implement any necessary changes based on user feedback. | Gather feedback from testing | Changed implemented | No user feedback | No changes based on user feedback made | - |