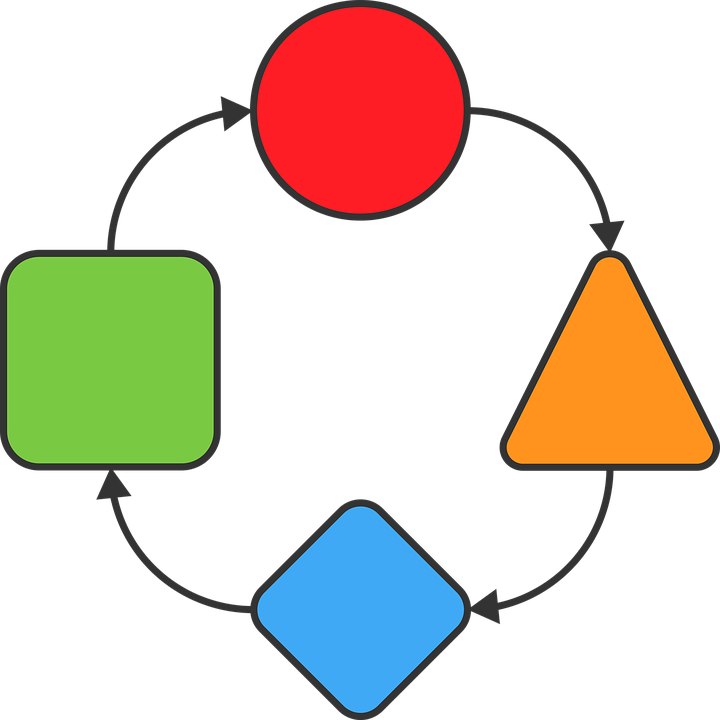
**Test cases Document**



**WQC**

**Version History**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Data** | **Versão** | **Descrição** | **Autor** | **Revisor** |
| 25/10/19 | 1.0 | Initial version | Richard Cardoso | - |
| 29/10/19 | 1.1 | 1. Changed document title from ‘Requirements Document: WQC’ to ‘Test cases Document: WQC’ | Richard Cardoso |  |

|  |  |
| --- | --- |
| **Client** | Weger Walter GmbH |
| **Document** | Test Cases Document: *WQC* |
| **Date** | 25 October 2019 |
| **Author** | **Richard Cardoso**  richard.cardoso@weger.com.br |

**Signatures**

|  |  |  |
| --- | --- | --- |
| Reviewed and Approved by: |  |  |
|  |  |  |

**Summary**

[1. Document goal 4](#_Toc23182107)

[2. Test cases 4](#_Toc23182108)

[2.1. TC001 4](#_Toc23182109)

[2.2. TC002 5](#_Toc23182110)

Document goal

This document describes some WQC system’s test cases.

1. Test cases
   1. TC001

|  |  |
| --- | --- |
| **ID** | TC001 |
| **Actors** | Mobile user |
| **Type** | Primary, essential |
| **Description** | A qr-code of a device is scanned and, after loading the data, the app shows the list of existing reports |
| **Pre-conditions** | 1. An authorized user with assigned roles already stored at the database 2. App opened at the welcome screen which must be showing:  * the name of the current user * the roles of the current user * the device’s id |
| **Post-conditions** | Reports list screen shown |
| **Associations** | FR001 |
| **Default flow** | 1. At the welcome screen, the user taps the qr-scan button 2. The user points the camera to a device’s qr-code 3. The app goes back to the welcome screen 4. The app translates the qr-code and starts to retrieve all related information from the server 5. The app shows the reports list screen filled with the retrieved reports list |
| **Alternative flow 1** | Default flow, step 2 – Back button pressed:   1. With the camera opened, the user presses the system’s back button 2. The app goes back to the welcome screen |
| **Alternative flow 2** | Default flow, step 4 – Translation failed:   1. The app fails to translate the qr-code for a given reason (incorrect / invalid / non-standard qr-code) 2. The app shows an error message with the code QR\_TRANSLATION\_FAILED and an ‘OK’ button 3. The user taps the ‘OK’ button and the app again shows the welcome screen |

* 1. TC002

|  |  |
| --- | --- |
| **ID** | TC002 |
| **Actors** | Mobile user |
| **Type** | Primary, essential |
| **Description** | After scanning a qr-code, the user opens the general pictures list, captures some pictures, remove one of them to cancel it and them uploads the remaining ones. |
| **Pre-conditions** | 1. An authorized user with assigned roles already stored at the database 2. App opened at the welcome screen which must be showing:  * the name of the current user * the roles of the current user * the device’s id  1. A valid qr-code |
| **Post-conditions** | General pictures list showing the new pictures after they were successfully captured and uploaded. |
| **Associations** | FR001, FR0010 |
| **Default flow** | 1. At the welcome screen, the user taps the qr-scan button 2. The user points the camera to a device’s qr-code 3. The app goes back to the welcome screen 4. The app translates the qr-code and starts to retrieve all related information from the server 5. The app shows the reports list screen filled with the retrieved reports list 6. The user taps the ‘general pictures list’ button 7. The app checks for outdated pictures and shows the filled pictures list screen 8. The user taps the ‘new’ button 9. The app shows up the camera preview 10. The user taps the ‘capture’ button 3 times, after ‘focusing’ the camera at different points (before each tap at the capture button) 11. At each tap, the app shows at the top of the screen a number that corresponds to the sum of took pictures 12. The user taps the ‘back’ button 13. The app shows a list of the captured pictures 14. The user taps the remove button that is shown over the second picture of the list 15. The app shows a question asking if the user wants to remove the picture 16. The user taps the ‘yes’ button 17. The app removes the picture of the list 18. The user taps the ‘upload’ button 19. The app asks the user to confirm the action 20. The user taps the ‘yes’ button 21. The app starts to upload the pictures and, for each finished one, it shows, in the list, an ‘ok’ check over the picture 22. When finished uploading all pictures, the app shows a success message 23. The user taps the ‘ok’ button 24. The app goes back to the pictures list from the default flow, step 12 and shows an updated pictures list |
| **Alternative flow 1** | Default flow, step 7. Back button pressed:   1. The user presses the screen’s back button 2. The app goes back to the reports list screen from default flow, step 5. |
| **Alternative flow 2** | Default flow, step 8 – Back button pressed:   1. With the camera opened and without taking any pictures, the user taps the system’s back button 2. The app goes back to the pictures list from default flow, step 7, |

|  |  |
| --- | --- |
| **Alternative flow 3** | Default flow, step 15 – ‘No’ button pressed:   1. The user taps the ‘no’ button 2. The app closes the question dialog without removing the picture |
| **Alternative flow 4** | Default flow, step 18 – Back button pressed   1. The user presses the screen’s back button 2. The app shows a question asking if the user wants to upload the pictures 3. The user taps the ‘yes’ button 4. The app executes the default flow starting at step 21 |
| **Alternative flow 5** | Alternative flow 4, steps 3 and 20 – ‘No’ button pressed   1. The user taps the ‘no’ button 2. The app goes back to the pictures list from default flow, step 12 and the flow ends. |