**Beet Seed**

|  |  |  |
| --- | --- | --- |
|  | Functional | Non-functional |
| What is tested? | Tests features/functions: validates the features of the product; functionality of the application is working | Tests non-functional aspects: Performance (stress, volume, server's volume), the software's behavior, reliability factor of the app/software); usability; security |
| When is it used? | Checks the accordance to customer requirements; Tests what the product does; | In accordance with end-user expectations |
| Limitations | ails to account for variables other than those given | Requirements are hard to specify and hard to do manually |
| Features | Requirements are easy to specify; Usually done manually; | depends on the time, accuracy, stability, correctness, and durability of a product under various adverse circumstances |

**Regression Testing vs. Retesting**

Regression Testing is intended to check the stability and functionality of software when modifications are being made. It includes passed and failed test cases and can be done manually or with automation tools. Retesting, on the other hand, is a method that re-executes the failed test cases to check defects, improving the quality and reducing verification time.

**Beet Sprout**

**Functional and Non-functional Testing**

Testing software isn't a case for using the same testing approach. One cannot say that functional testing is better than non-functional, or vice versa since both are important.

Non-functional testing is just as crucial as checking if everything works. Sometimes, teams do not give it as much attention because it does not display those obvious improvements done. If the performance is not great, the users might get annoyed or dissatisfied, but they can still use the software.

But if the basic functions are broken, that's a real problem. Users cannot do what they need and want. That is why functional tests are a big deal. They can be faster and cheaper, and they make sure the basics function as indicated in the requirements.

Still, non-functional tests matter a lot. A good testing setup covers all sorts of non-functional stuff, like how fast it is and how easy it is to use.

**Smoke Testing**

In the early stages of software development, developers do "smoke testing." This is where they check if the code is working right and if the software's core functions are doing their job.

If the code doesn't compile right or the basic stuff isn't working, smoke testing will fail. It won't work if the setup it needs isn't there, the code doesn't have all the necessary parts, or if a big bug messes everything up.

Only after passing smoke testing do they move on to other kinds of testing.

Smoke testing cannot be omitted since its results indicate whether the testing can be done more in-depth or if major flaws need immediate fixing.

**The Mighty Beet**

The 5 functional cases:

<https://shatohinaliz.testrail.io/index.php?/suites/view/12&group_by=cases:section_id&group_order=asc&display_deleted_cases=0>

The 5 functional cases test run: <https://shatohinaliz.testrail.io/index.php?/runs/view/17&group_by=cases:section_id&group_order=asc>

The non-functional requirements.

For the app I would add the following non-functional requirements:

1. The application must be supported on smartphone devices with Android versions 11.0, 12.0, and 13.0.

2. The application must be able to support HEIF, jpg, PNG, and TIFF picture formats.

3. The app must provide a preview window before posting a picture.

To test them I would use the smoke and system integration testing techniques.