2023 SPRING

CMPE 492 Senior Project II

Test Plan Report

Name of the Project: Receipt Analyzer

The URL of the webpage: https://bzelihagunay.wixsite.com/receipt-analyzer

Team Members' Name: Burçak Zeliha Günay, Ahmet Berat Akdoğan, Başar Aslan, Hüseyin Hikmet Fındık

Section: 01

Supervisor's Name: Aslı Gençtav

Jury Members' Name: Venera Adanova, Gökçe Nur Yılmaz

Date of Submission: 16/05/2023

Table of Contents

1. Introduction

The purpose of this test plan report is to outline the testing approach and strategies for the Receipt Analyzer App. This document provides an overview of the testing objectives, scope, test environments, test deliverables, and testing timeline. It also includes approach, resources, descriptions of the features to be tested, testing methodology, and control procedures.

1.1. Project Overview

The Receipt Analyzer App is a mobile application designed to extract text from images using optical character recognition (OCR) technology. The app then processes the extracted text and generates a graphical representation of the data provided. The primary goal of the application is to provide users with a convenient way to convert visual data into actionable insights.

1.2. Testing Objectives

The main objectives of the testing process for the Receipt Analyzer App are as follows:

- Verify the accuracy and reliability of the text recognition functionality by comparing the extracted text with the original image content.
- Validate the correctness and completeness of the graph generation process by ensuring that the generated graphs accurately represent the extracted data.
- Assess the performance and responsiveness of the application under different scenarios, including varying image sizes, different text layouts, and concurrent user interactions.
- Identify and report any defects or issues encountered during testing, ensuring they are thoroughly documented and tracked for resolution.
- Ensure compatibility and seamless operation across various mobile devices and operating systems, including Android devices running Android OS version 7.0 and above.

1.3. Scope

The scope of this testing effort encompasses both functional and non-functional aspects of the Receipt Analyzer App. It includes the verification of core features such as image capture, OCR processing, text extraction, data analysis, and graph rendering. Additionally, performance testing, unit testing, system and integration testing, and user acceptance testing will be conducted to evaluate the app's responsiveness, user-friendliness, and compatibility with different devices. These testing types will be discussed and explained under the testing methodologies part of this report. Definitions of these testing types are as follows:

- Performance testing is a non-functional software testing technique that detects the levels of stability, speed, scalability, and responsiveness of the app accordingly to the workload.
- Unit testing is a type of testing which acknowledges a methodology that focuses on parting every testable feature to its units,
- User Acceptance Testing (UAT) is an acceptance test carried out by the client following the software development process and is based on actual use cases. This test is performed to ensure that the software performs as anticipated and satisfies customer expectations. Prior to production, UAT is a crucial environment that is advised to be utilized to make sure that the project is accepted by users and that it is prepared for broadcast.
- Beta testing refers to the testing of a software's final version by a small number of users prior to its release to the general public. These evaluations serve as input to find and address software flaws and other issues. Beta testing is frequently the last phase of a product's testing before it is made available to the public.
- System testing and integration testing are two essential types of software testing to guarantee the quality and dependability of software products. A type of testing called integration testing is concerned with examining the interfaces and interactions between different software modules or components. While system testing examines if a whole setup complies with both functional and non-functional requirements.

1.4. Test Environments

The testing environment will be examined under three different categories: Hardware Environment, Software Environment, and Database Environment. The Receipt Analyzer App will be tested on the following environments:

- Test environment consisting of various Android devices with different screen sizes and resolutions.
- Test data, including sample images and datasets to simulate different scenarios for text recognition and graph generation.

1.5 Test Schedule

This report will explain the steps of the testing procedures that will be applied, while each step will be showing the details of different layers and functions of the app. The subsequent sections of this report will provide detailed information about the specific test scenarios, test cases, and test procedures to be followed during the testing phase of the project.

1.6. Test Deliverables

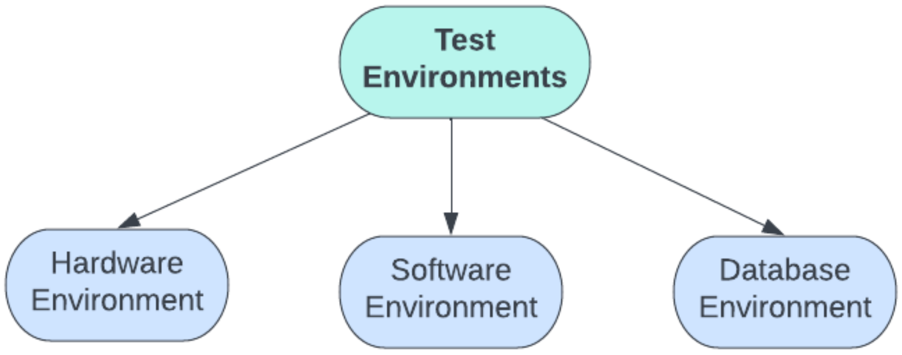
The test deliverables for this project will include:

- Test plan: This document outlines the overall testing approach, scope, objectives, and test environment details.
- Test cases: Detailed test cases covering various scenarios and functionalities of the application.
- Test data: Sample images and datasets used for testing the OCR and graph generation functionality.
- Test bugs and errors: Detailed bugs and errors of test executions, defect reports, and different testing scenarios.

1.7 Test Risks

This report will underline some expected or observed risks. Risks will be explained with details of occurrence times such as before testing, during testing, or after testing. The testing risks part will also consider different conditions that the Receipt Analyzer App can challenge.

2. Test Environment



Since our project includes several steps and tools during the usage, we categorized the testing environments into three subparts:

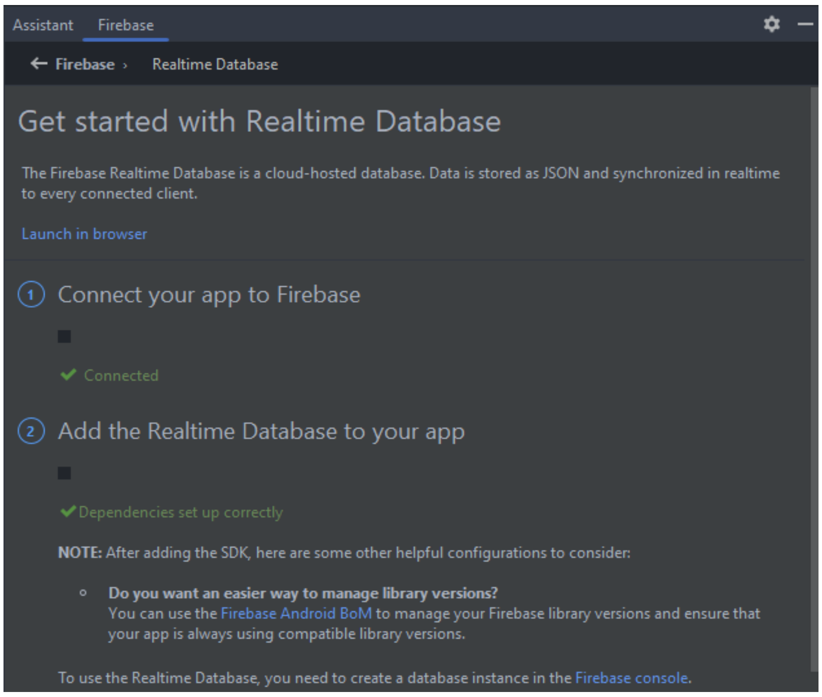
- **Hardware Environment:** This part consists of devices that are needed during the testing procedures. In the project, we are using our smartphones for checking the interface structure of the application. In addition, personal computers running on Windows operating systems are used to organize the code and backend structure. In the hardware environment, it is also important to note that we have used type-C to USB cables to provide a connection between the Java program in the computer and the smartphone.

- **Software Environment:** Since the software environment provides the functionality, architecture, database connection, and features of our application, this environment is the most essential part of our test implementations. The software environment includes Android Studio IDE, Java codes, XML files, and Gradle scripts. The software environment allows us to check errors and make improvements during the testing.

- **Database Environment:** In Receipt Analyzer, the price, product, and date information are retrieved after scanning the text from the receipt. Therefore, as a database environment during the testing, Realtime Database, a cloud-hosted database in a web server is used. We implemented several tests by taking pictures of various receipts to observe data transfer from the application to the Realtime Database.

3. Test Schedule

As a first step, we established the connection between the smartphone and the computer via a cable. We observed that when we run the application without a connection with Integrated Development Environment (IDE), the application works without any communication with the Realtime Database. Therefore, the data will not transfer to the database which causes the end-user to not see the graph. For this reason, it is essential to connect the computer to the smartphone to observe current updates, developments, and features of the program. Before running the program on the integrated development environment, we also checked the connection status of the Firebase Realtime Database shown below:



Get started with Realtime Database

The Firebase Realtime Database is a cloud-hosted database. Data is stored as JSON and synchronized in realtime to every connected client.

Launch in browser

1 Connect your app to Firebase

✓ Connected

2 Add the Realtime Database to your app

✓ Dependencies set up correctly

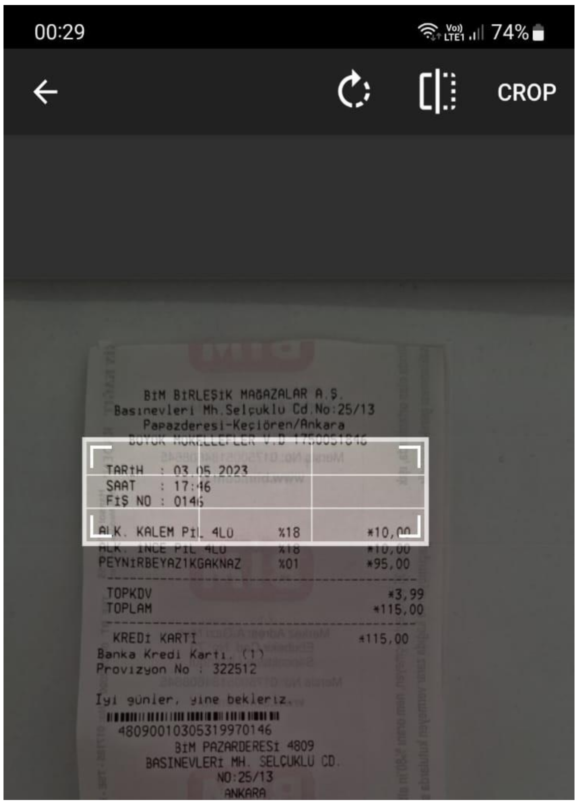
NOTE: After adding the SDK, here are some other helpful configurations to consider:

- **Do you want an easier way to manage library versions?**
  You can use the Firebase Android BoM to manage your Firebase library versions and ensure that your app is always using compatible library versions.

To use the Realtime Database, you need to create a database instance in the Firebase console.

Figure-1: Connecting Database.

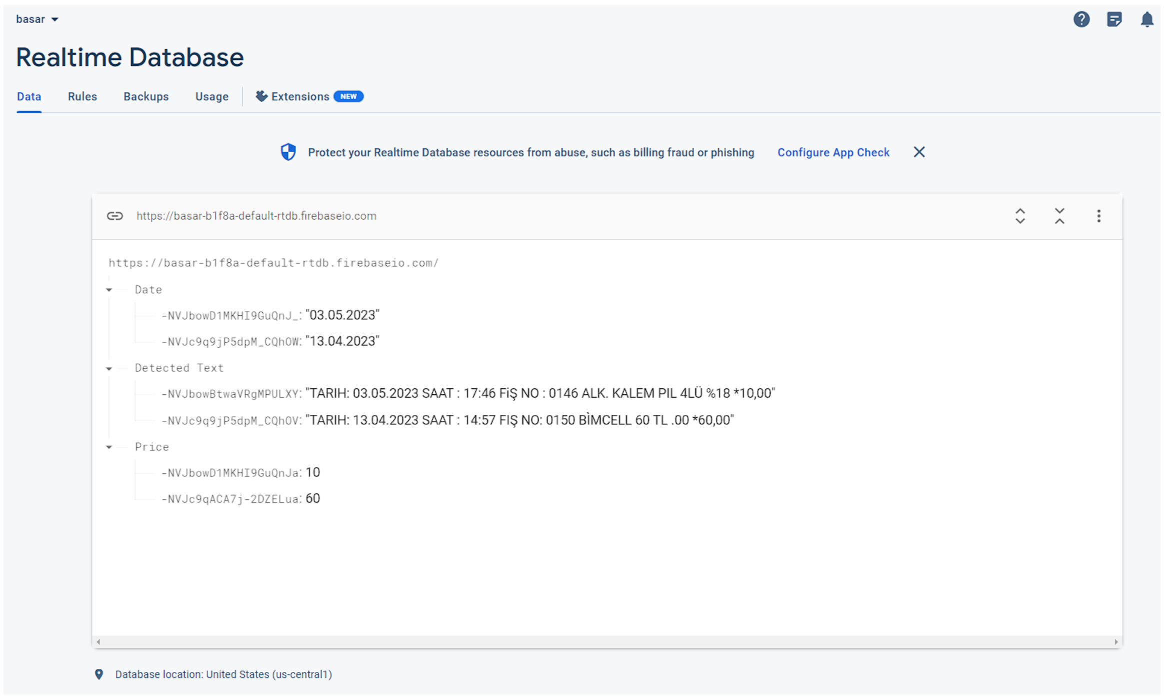
When we run the application, the home page is displayed on the screen. After pressing the "CAPTURE" button, we tested the camera and gallery options on the smartphone. During our tests, we observed that the application only allows us to use pictures from the gallery for devices that have Android 13 and above versions. After selecting/taking the receipt picture, we implemented several tests to check the crop functionality as below:



```
        BİM BİRLEŞİK MAĞAZALAR A.Ş.
    Basınevleri Mh.Selçuklu Cd.No:25/13
       Papazderesi-Keçiören/Ankara
    BÜYÜK MÜKELLEFLER V.D 1750051846

 TARİH  : 03.05.2023
 SAAT   : 17:46
 FİŞ NO : 0146

ALK. KALEM PİL 4LU       %18      *10,00
ALK. İNCE PİL 4LU        %18      *10,00
PEYNİRBEYAZ1KGAKNAZ      %01      *95,00
-----------------------------------------
 TOPKDV                            *3,99
 TOPLAM                          *115,00
-----------------------------------------
  KREDİ KARTI                    *115,00
Banka Kredi Kartı. (1)
Provizyon No : 322512

Iyi günler, yine bekleriz.
  48090010305319970146
           BİM PAZARDERESİ 4809
     BASINEVLERİ MH. SELÇUKLU CD.
                NO:25/13
                 ANKARA
```

Figure-2: Cropping the market receipt.

Then, we observed the retrieved data on the displayed page which shows the scanned text from the receipt. We implemented a push() methodology for the scanned data to prevent data loss. After that, we pressed the "DETECTED TEXT" button to save the retrieved data to the Realtime Database. Additionally, we implemented Java methods to extract price and date information before sending the data to the Realtime Database. For now, we only focused on only one product and price in one crop to analyze the success of extraction operations better in our testing scenarios. The data on the Firebase can be seen below:


basar
Realtime Database
Data
Rules
Backups
Usage
Extensions
NEW
Protect your Realtime Database resources from abuse, such as billing fraud or phishing
Configure App Check
https://basar-b1f8a-default-rtdb.firebaseio.com
Database location: United States (us-central1)

```
https://basar-b1f8a-default-rtdb.firebaseio.com/
  Date
    -NVJbowD1MKHI9GuQnJ_: "03.05.2023"
    -NVJc9q9jP5dpM_CQhOW: "13.04.2023"
  Detected Text
    -NVJbowBtwaVRgMPULXY: "TARIH: 03.05.2023 SAAT : 17:46 FiŞ NO : 0146 ALK. KALEM PIL 4LÜ %18 *10,00"
    -NVJc9q9jP5dpM_CQhOV: "TARIH: 13.04.2023 SAAT : 14:57 FIŞ NO: 0150 BİMCELL 60 TL .00 *60,00"
  Price
    -NVJbowD1MKHI9GuQnJa: 10
    -NVJc9qACA7j-2DZELua: 60
```

Figure-3: Scanned data on the Realtime Database.

As a last step, we tested the graph functionality of the application which allows us to compare prices of the same product in different time intervals. Although we attained a chart that represents prices on the y-axis as below, we observed that the graph methodology we used affected the visualization of the data negatively in some cases. Therefore, during the following weeks, we aim to make adjustments to better display price and date values.

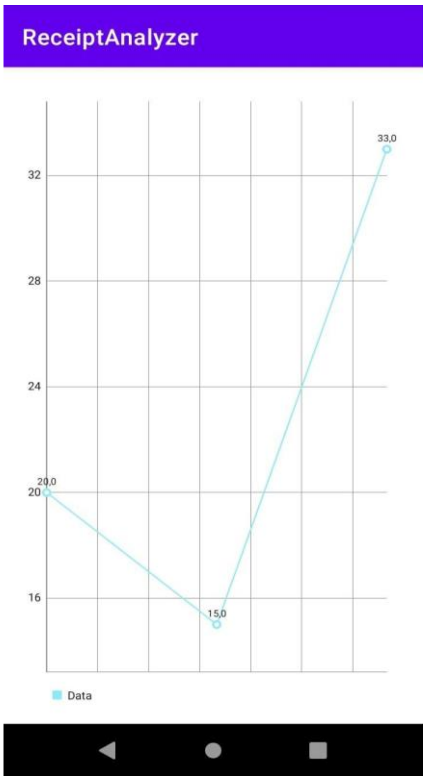


Figure-4: Graph to illustrate price changes.

4. Testing Methodology

4.1 System and Integration Testing

Since we wanted to ensure the quality and reliability of our app, we wanted to verify the interfaces and interactions between different components and modules of our app. We have focused on it right after unit testing and tested the welfare of individual components. We wanted to see if there are any unexpected behaviors or non-functionalities. Our test cases and attempts showed us that for one individual occasion, our application cannot process our current cropping interface and fails to proceed to the text detection phase. There was no error message at our Android Studio console through this malfunction. This has occurred on a device that had an Android version 12, and it should be noted that for that exception, the option of picking an image from the gallery of the device results well and does not give any error or non-functionality. Besides this specific example, none of our testing attempts and test cases showed an error in our app's functionality.

4.2. Beta Testing

In beta testing, the security features of the application are tested while ensuring the reliability of the mobile application. In this context, the mobile application is not 100% consistent and reliable. Because the project is basically based on OCR technology, it has been tested that sometimes scenarios where unwanted or wrong words are detected while taking photos of the market receipts. In particular, while the OCR algorithm is running, it has been tested that the sizes and placements of different characters on that paper cause inconsistency and unreliability in terms of detection in the application. When it is considered from a security point of view, it has been determined that there is no security vulnerability since the information written to the database consists of the product price, date, and product name. However, in the future, when simple authentication is added to the application, tests such as blurring the passwords during login and writing this information into the database in an encrypted way can be performed.

4.3. Performance Testing

Since the developed mobile application is not a hybrid or web-based mobile application, instead it is a native application, response times are quite low. During our tests, a slight delay was observed in factors such as menu transitions, button click times, response times, and screen display time. The transition time between the menus is very short (0-2 seconds). On the backend side, the execution time of the code does not exceed 4-5 seconds. However, during the tests, it was observed that this time increased to 10-15 seconds on phones with lower Android versions.

When revisions are made to the code, on some occasions, the application may crash. This situation is also observed during the Performance Test phase. Besides, when the "Detected Text" button is pressed, the saving time of the data to the Firebase Realtime Database may change depending on the traffic on Google's servers. Generally, these differences do not cause a problem in terms of performance, but when the "Detected Text" button is pressed, the data is written to the database in approximately 2-3 seconds.

4.4. Unit testing

In general, Unit Testing is divided into manual and automated testing. However, in the mobile application that is developed, the scenarios and the execution of the project do not follow a certain path. The reason is that the cropping mechanism is implemented with the limitations imposed by the developers according to the demands of the users. As an example of this, you need to crop the receipt so that only the date, price and product information are taken. Considering these situations, it is not easy to automate the system or the app. Therefore, the tests have been run manually. For instance, when the unit of the system is tested, the control of whether the crop and OCR mechanism works without errors is made by the user and the developer. In addition, various tests are used to check whether the text on the back of the receipt affects text detection while the receipt is being scanned.

4.5. User Acceptance Test

User acceptance test (UAT) is the test that the developers are testing the main system together with the users before the system is ready and goes to a live environment. In the graduation presentation that will be made at the end of the semester at the university, an example can be given to the testers to be done with the jury members and supervisor during the presentation. For instance, just like an end user, all of the tests that they will do in accordance with certain rules can be regarded as user acceptance tests. To test the functionality and operability of the application, an example can be given of the process of running the system from scratch and displaying the graph. OCR and crop features can be given as an example of these details of the processes.

5. Risks

Although there is no serious risk in the project, there are some situations that will prevent the basic work of the project. The quality of the photograph taken on the receipt paper, especially the lighting in the shooting environment and the unnecessary writings on the back of the receipt, can cause inconsistency in the OCR detection according to the tests performed. Furthermore, wrong information can be sent to the database. In addition, it is observed in some tests that the graphic displays wrong results to the end-end user in scenarios with unsuccessful crop (not getting the rectangle in the frame with full borders) and low quality of the selected receipt photo. Therefore, there is a risk that the application will not work as desired. The reason is that in some of the actions like the user's wrong crop operation and the inconsistency in OCR cause the graphic to show incorrect results to the user.

6. Errors

Various errors were encountered during the testing process. For instance, when the number of detected products is more than one, problems are encountered in retrieving the prices of the products. The reason for this is that the data does not transfer to the database (Realtime Database) in a certain order since there is no such kind of fixed mythology or algorithm that finds the price on a database. In addition, as mentioned in the test schedule section, when trying to use the camera functionality on devices with high Android version, the application returns the user to the home page instead of the crop screen. In the current system, after the crop operation is done, the date information on the receipt is not displayed on the graph as the X-axis. Therefore, currently there is a problem on the X-axis of the date/price graph. For this reason, as future improvements aim, it is aimed to increase the number of products detected at the time of a single crop operation and to integrate the application with devices with high Android versions. Plus, it is aimed to make improvements to better present the date and price values with respect to the x and y axes of the graph to be displayed to the end user.

7. Test Case

7.1 Test Case 1

The process from starting the program from scratch to the end of the crop process.

- The user or developer connects his/her android-based phone to the computer with the help of a type-c or micro-USB cable,
- Click the run button to execute the Java code,
- Then the user/developer sees the mobile application to begin,
- Then it is seen that the main screen on the phone like welcoming word and the name of the application and the relevant button like CAPTURE,
- User selects the supermarket name that the application supports.
- User decides to take a photo/or choose from a gallery to perform OCR and crop action.
- After the desired decision is made, the application offers the crop feature to the user.
- After making the crop, the user observes which part of the market receipt is cropped on the mobile screen.

According to the version of the mobile application used, the above scenarios can be done both by selecting a photo from the gallery and by taking a photo with the help of the camera from scratch.

7.2 Test Case 2

The control process and test of whether the data after the crop, displays to the phone screen without data loss and whether the data is successfully saved to the db with a single button or not.

- In order to write the detected data to the database, it is checked whether the project is connected to Firebase successfully.
- Then, the user clicks the DETECTED TEXT button and examines the data written to the database on the Firebase console screen with the help of a second web-based device.

7.3 Test Case 3

Charting and plotting the obtained data. Checking the correctness of the elements on the X-axis and Y-axis of the graph.

- After observing the detailed extracting of the data in the database, the user clicks the plot button on the mobile screen,
- Then, the user/developer sees the plotted date/price graph in terms of the x and y coordinates.

8.Roles and Responsibilities

Role	Responsibilities
Burçak Zeliha Günay Hüseyin Hikmet Fındık	Describing the scope, approach, resources, and schedule of the testing activities for the system.
Ahmet Berat Akdoğan Başar Aslan	Creating and Running Test Scenarios and Scripts on a Mobile Application (Tester)

Figure-5: Roles and Responsibilities Table.

9. References

1. BlazeMeter. (n.d.). Mobile App Performance Testing: Best Practices, Tools, and Tips. Retrieved from https://www.blazemeter.com/blog/mobile-app-performance-testing

2. ArtofTesting.com. (n.d.). Performance Testing. Retrieved from https://artoftesting.com/performance-testing#:~:text=Performance%20testing%20is%20a%20type,performance%20testing%20tools%20like%20JMeter.

3. Guru99. (n.d.). User Acceptance Testing (UAT) - Complete Guide. Retrieved from https://www.guru99.com/user-acceptance-testing.html

4. nFocus Limited. (n.d.). Testing Methodologies - Five Core Components of Testing. Retrieved from https://blog.nfocus.co.uk/testing-methodologies-five-core-components-of-testing

5. The QA Lead. (n.d.). Best Unit Testing Tools in 2021. Retrieved from https://theqalead.com/tools/best-unit-testing-tools/

6. GeeksforGeeks. (n.d.). Difference between System Testing and Integration Testing. Retrieved from https://www.geeksforgeeks.org/difference-between-system-testing-and-integration-testing/

7. TechTarget. (n.d.). Unit Testing. Retrieved from https://www.techtarget.com/searchsoftwarequality/definition/unit-testing#:~:text=Unit%20testing%20is%20a%20software,tests%20during%20the%20development%20process.

8. Micro Focus. (n.d.). Performance Testing. Retrieved from https://www.microfocus.com/en-us/what-is/performance-testing#:~:text=Performance%20testing%20is%20a%20non,up%20under%20a%20given%20workload.