

American International University-Bangladesh (AIUB)  
Department of Computer Science  
Faculty of Science &Technology (FST)  
Fall 20-21

Section: A/ C  
Software Quality Assurance and Testing

Apartment Service System 1.03

A Report submitted

By

|  |  |  |
| --- | --- | --- |
| SN | Student Name | Student ID |
| 1 | **MD. IER UDDIN RABBI** | 16-32180-2 |
| 2 | **SAMIMA SULTANA** | 16-32012-2 |
| 3 | **OALID MOHAMMAD KHALID BIN** | 17-33622-1 |
| 4 | **MOON MD.IMRAN KABIR** | 17-34106-1 |

Under the supervision of

Faculty Name

ABHIJIT BHOWMIK

Faculty Designation

Software Test Plan

for

<Apartment Service System 1.03>

Version 1.0 approved

Prepared by <MOON MD.IMRAN KABIR, MD.IRE UDDIN RABBI,SAMIMA SULTANA,OALID MOHAMMAD KHALID BIN >

<organization>

<15.12.2020>

Table of Contents

[Revision History 3](#_Toc37271323)

[1. TEST PLAN IDENTIFIER: RS-MTP01.3 3](#_Toc37271324)

[2. REFERENCES 3](#_Toc37271325)

[3. INTRODUCTION 3](#_Toc37271326)

[Background to the Problem 3](#_Toc37271327)

[Solution to the Problem 4](#_Toc37271328)

[4. REQUEIREMNT SPECIFICATION 4](#_Toc37271329)

[4.1 System Features 4](#_Toc37271330)

[4.2 System Quality Attributes 4](#_Toc37271331)

[4.3 System Interface 5](#_Toc37271332)

[4.4 Project Requirements 5](#_Toc37271333)

[5. FEATURES NOT TO BE TESTED 5](#_Toc37271334)

[6. TESTING APPROACH 5](#_Toc37271335)

[6.1 Testing Levels 5](#_Toc37271336)

[6.2 Test Tools 6](#_Toc37271337)

[6.3 Meetings 6](#_Toc37271338)

[7. TEST CASES/TEST ITEMS 6](#_Toc37271339)

[8. ITEM PASS/FAIL CRITERIA 7](#_Toc37271340)

[9. TEST DELIVERABLES 7](#_Toc37271341)

[10. STAFFING AND TRAINING NEEDS 8](#_Toc37271342)

[11. RESPONSIBILITIES 8](#_Toc37271343)

[12. TESTING SCHEDULE 8](#_Toc37271344)

[13. PLANNING RISKS AND CONTINGENCIES 9](#_Toc37271345)

[14. APROVALS 9](#_Toc37271346)

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| Revision | Date | Updated by | Update Comments |
| 0.1 | 2020.12.15 | MD. IER UDDIN RABBI | First Draft |
| 0.2 | 2020.12.17 | SAMIMA SULTANA | Second Draft |
| 0.3 | 2020.12.19 | OALID MOHAMMAD KHALID BIN | Third Draft |
| 0.4 | 2020.12.21 | MOON MD.IMRAN KABIR | Fourth Draft |
| 0.5 | 2020.12.15 | MD. IER UDDIN RABBI | Fifth Draft |
| 0.6 | 2020.12.17 | SAMIMA SULTANA | Sixth Draft |
| 0.7 | 2020.12.19 | OALID MOHAMMAD KHALID BIN | Seventh Draft |
| 0.8 | 2020.12.21 | MOON MD.IMRAN KABIR | Eighth Draft |
|  |  |  |  |

# TEST PLAN IDENTIFIER: Apartment Service System 1.03

# REFERENCES:

SOFTWARE DESIGN SPECIFICATION: Apartment protocol service system is both web-based and desktop-based applications which will be used for the people who lived in a large society and they need to maintain a lot of activities like managing service charge, controlling entrances of different types of people.

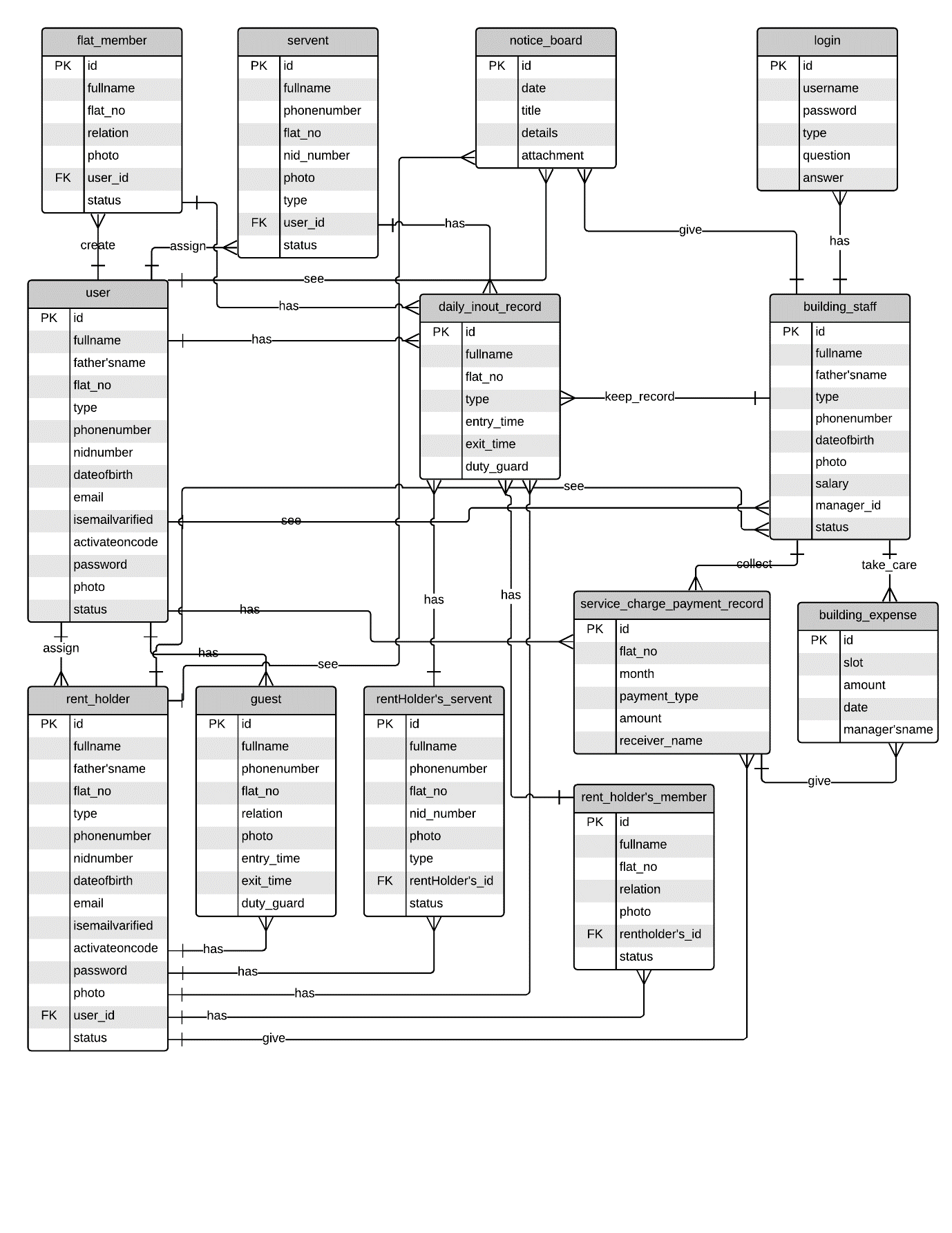


Figure 1: E-R Diagram

# INTRODUCTION

## Background to the Problem:

Over the years apartment authority facing a common problem to maintain a huge record as well as managing the service system. The managing system is getting very difficult day by day because of maintaining a large number of people in a manual way without any mistake.

* **Growth of Data:** Data is increasing day by day and maintaining a huge number of data in this way is totally difficult.
* **Lack of computerized system:** Without a computerized system, it is not possible to calculate this kind of robust balance sheet accurately.
* **The security issue in storing data**: We know data is money. In manual or traditional way data is not secured, it may be stolen or damaged. Losing data can collapse the whole system. The digitalized authentication process will be the best option nowadays if we take the necessary steps.
* **No central database system:** Without a central database system, it will be difficult to access the data and merge the data properly. There may have redundancy in manual data storing systems but the digitalized system is safe, secure, recoverable and there is no risk as well.

## Solution to the Problem:

If we consider the current recording system of data and user activities of an apartment is totally in a manual way with lots of papers. Each house may one file to store user’s data and service charge information and other lists of expenses. Every month or after a certain period, the authority needs to merge all the calculations and make a balance sheet for maintaining the service charge system which is totally time-consuming. Imagine a society with 1000 (thousand) family or more, there might be a thousand files. It is quite impossible to manage this huge data and doing the calculation without any error. On the other hand, if the whole process will maintain by a computerized system with a graphical user interface, all the things can be handled very easily without any error and fast as well. Authority can break all the limitations and the whole system will be secured and easy to access. So a graphical user interface software with central database connection can solve the problem

# REQUEIREMNT SPECIFICATION:

## System Features:

## System Login

**Functional Requirements**

* This functionality describes how to enter into the System.
* The user must be valid before logging in.
* User should give username and password to enter the system.
* After clicking the login button user will be validated by the system.
* If he is validated successfully, he will be redirected to his particular interface or he will receive an error.
* If he is not able to make himself validate within three times, his account will be blocked for a certain period.

**Priority Level:** High  
**Cross-reference:** none

## Identity verification

**Functional Requirements**

* This function is for verifying the ingoing persons by the guards.
* The guard will have an interface to check the person that he is valid or not for entering the premises.
* The process will be simulated by taking fingerprint or id card scanning from the person.
* If his biometric sample is matched with the system, only then he will get access to enter into the building.

**Priority Level:** High  
**Cross-reference:** If the user failed to authenticate himself several times, he will be blocked.

## Guest entry-exit record

**Functional Requirements**

* This function is for verifying the ingoing guests by the guards.
* Guests will be verified by giving flat information and visit of purpose and this will be cross-checked to make sure that he gives valid information.
* If the given information is correct, the access for entering the apartment will be granted and his information will be recorded in the system and while leaving the premises his information will be updated as well.

**Priority Level:** High  
**Cross-reference:** If the info is not matched, he cannot enter.

## Service charge collection

**Functional Requirements**

* This functionality is for taking service charge information from the flat owners by a valid manager and giving a printed document after a successful transaction.
* The manager will take the money and give input to the system.
* Print a receipt of the service charge for the flat owner.

**Priority Level:** High  
**Cross-reference:** none

## Building’s expense management

**Functional Requirements**

* This functionality is for maintaining the building’s expenses.
* A valid manager will give necessary input for the particular expenses (Gas,

Electricity, Water, Dustbin, Staff Salary).

* While distributing the salary of security guards and other stuff, the manager needs admin approval for granting access.
* The manager can print any information at any time.

**Priority Level:** High  
**Cross-reference:** none

## Building notice board management

**Functional Requirements**

* This functionality if for maintaining the notice section.
* Flat owners can preview the notices.
* Notice will be provided by a valid manager, only he can add notices in the system.
* If he feels the necessity to upload any attachment like pdf or doc file, he has the privilege to upload it.

**Priority Level:** High  
**Cross-reference:** Notice will be updated only with a valid format

## Building staff creation and deletion

**Functional Requirements**

* This functionality will help admin to add and discard building staff.
* Admin can create an account for the staff as well as he can remove the staff at any time.
* After a successful operation, the admin will be notified by the system message.

**Priority Level:** High  
**Cross-reference:** It will be created only after filling up all the necessary information.

## Give approval to managers request

**Functional Requirements**

* This functionality is for permitting salary, expense, and notice requested from the building manager’s account.
* Admin will verify the manager’s request to give permission and the manager will be notified accordingly.
* Managers' activity depends on admin approval in the system.

**Priority Level:** High  
**Cross-reference:** Every decision will be pending for admin approval.

## Flat member and servant addition, deletion and altering

**Functional Requirements**

* This function will privilege the flat owners and rent holders to add, update, or even delete their member and servant account.
* Both the users have member and servant creation option in portal account.
* After a successful operation, the user will be notified by the system.

**Priority Level:** High  
**Cross-reference:** none

## Service Charge Payment and payment details

**Functional Requirements**

* This functionality is for flat owners to make payments and observe payment history by himself.
* Flat owners and rent holders can use these functions to make the transaction with their suitable method.
* They can watch their previous payment information and print necessary things as well.

**Priority Level:** High  
**Cross-reference:** none

1. **Displaying Notice for Particular User**

**Functional Requirements**

* This function is for portal users, flat owners, rent holders. Easily they can preview the notices that are provided by the manager.
* They can see text notices along with the attachment.

**Priority Level:** High  
**Cross-reference:** none

1. **Displaying Necessary Information of Flat Owners**

**Functional Requirements**

* One flat can see limited information from other flat owners. This is just because of an emergency contact.
* User privacy will be maintained highly.

**Priority Level:** High  
**Cross-reference:** none

1. **Maintain Log System of staffs**

**Functional Requirements**

* There will be a log system for staff where their entry and exit time will be recorded.
* Guards, servants, drivers of flat owners are included in this operation.
* Partially every entry-exit time will be recorded because flat owners also need a fingerprint system to take entry into the apartment.

**Priority Level:** High  
**Cross-reference:** none

1. **Distribute Expense Sheet to the Flat Owners**

**Functional Requirements**

* Every month an expense sheet will be generated by the system.
* There will be mentioned clearly the expenses that are included by the manager.
* Flat owners can easily check or even print the expense list.
* This will be provided as a balance sheet format.

**Priority Level:** High  
**Cross-reference:** none

## System Quality Attributes:

Non-functional requirements refer to the quality assurance parameters that are rudimentary for software to be successful. The non-functional requirements are given below:

**Availability:** Availability Testing which is also called Durability Testing is a kind of performance testing in which the application runs for a set period of time and collects failure events and repair times, and compares the availability percentage to the service level agreement. It is related to reliability. This attribute is concerned with failures. Failures must be corrected or masked by the system. It is also related to system performance.

**Modifiability:** Modifiability encompasses two aspects: “Maintainability. (1) The ease with which a software system or component can be modified to correct faults, improve performance or other attributes, or adapt to a changed environment. This attribute is concerned with what can change and how easy it can be to change functionalities. The system shall have the ability to adapt to changes easily.

**Performance:** In [software quality assurance](https://en.wikipedia.org/wiki/Software_quality_assurance), performance testing is in general a [testing](https://en.wikipedia.org/wiki/Software_testing) practice performed to determine how a [system](https://en.wikipedia.org/wiki/System) performs in terms of responsiveness and stability under a particular workload. It can also serve to investigate measure, validate or verify other [quality](https://en.wikipedia.org/wiki/Quality_(business)) [attributes](https://en.wikipedia.org/wiki/Attribute_(computing)) of the system, such as [scalability](https://en.wikipedia.org/wiki/Scalability), [reliability](https://en.wiktionary.org/wiki/reliability) and resource usage. The system response time is measured to quantify performance. Moreover, availability is also focused to measure performance.

**Security:** Security Testing is a type of Software Testing that uncovers vulnerabilities of the system and determines that the data and resources of the system are protected from possible intruders. It ensures that the software system and application are free from any threats or risks that can cause a loss. This is one of the most crucial attributes. All user information must remain secure and integrated. There should be no denial of service.

**Testability:** Software testability is the degree to which a software artifact (i.e. a software system, software module, requirements- or design document) supports testing in a given test context. ... Many software systems are untestable, or not immediately testable. As the test cost for every project is nearly 40%, this attribute is efficient to reduce test costs.

**Usability:** Usability Testing also known as User Experience(UX) Testing, is a testing method for measuring how easy and user-friendly a software application is. A small set of target end-users, use software application to expose usability defects. The software will be for blind individuals. So, it must maintain the utmost user-friendliness. The user must feel ease to use the software and the system must be quick to interact with the user.

**Reliability:** Reliability refers to the consistency of a measure. A test is considered reliable if we get the same result repeatedly. Software Reliability is the probability of failure-free software operation for a specified period of time in a specified environment.The system must start to work after facing maximum failure.

## System Interface:

Figure 2: Use Case Diagram

## Project Requirements:

Tentative/Approximate budgeting of the project is shown below:

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Rate Per day(taka)** | **Quantity/Unit/day** | **Total** |
| 1. **Selection Costs** |  |  |  |
| 1. Internal Resources/ staff costs | 15,000 | 10 days | 150,000 |
| 1. Information System department (IS) | 10,000 | 10 days | 100000 |
| 1. Legal assistance | 6000 | 10 days | 60,000 |
|  |  |  | **Total = 310,000** |
| 1. **Software Costs** |  |  |  |
| 1. User licenses |  |  | 100000 |
| 1. Database user licenses |  |  | 50000 |
| 1. Operating system | 0 | 0 | 0 |
|  |  |  | **Total = 150,000** |
| 1. **Hardware Costs** |  |  |  |
| 1. Servers | 200,000 | 1 | 2,00,000 |
| 1. PCs’ | 50,000 | 4 | 2,00,000 |
| 1. Printers | 6,000 | 3 | 18,000 |
| 1. Scanners | 4,000 | 2 | 8,000 |
| 1. Cameras | 50,000 | 4 | 2,00,000 |
|  |  |  | **Total = 626,000** |
| 1. **Network Costs** |  |  |  |
| 1. Cabling, LAN or other networks |  |  | 50,000 |
| 1. Routers | 3000 | 3 | 9,000 |
| 1. Dedicated lines |  |  | 10,000 |
|  |  |  | **Total = 69,000** |
| 1. **Labor Costs** |  |  |  |
| 1. Implementation, system installation, report writing, documentation |  |  | **Total = 70,000** |
| 1. **Project team** |  |  |  |
| 1. Project manager |  |  | 170,000 |
| 1. Project leader |  |  | 150,000 |
| 1. Programmers | 30,000 | 8 persons | 240,000 |
| 1. Testers | 30,000 | 4 persons | 120,000 |
|  |  |  | **Total = 680,000** |
| 1. **Support Costs** |  |  |  |
| 1. Client manager | 35,000 | 2 | **Total = 70,000** |
| 1. **Office Costs** |  |  |  |
| 1. Office rent | 30,000 | 3 months | **90,000** |
| 1. furniture | 40,000 |  | **40,000** |
| 1. Training staff | 30,000 |  | **30,000** |
|  |  |  | **Total = 1,00000** |
|  |  | **Sub Total =** | **2,135,000** |
|  |  |  |  |

# FEATURES NOT TO BE TESTED:

Coming up next is a rundown of the territories that won't be explicitly tended to. All testing in these territories will be roundabout because of other testing endeavors.

1. Building Entry/leave log information investigation. Since this information are totally leveled out of building specialists and it is outside task scope. The essential information will be given to the specialists so they can utilize this information for their utilization.

2. Administration Charge information. These information esteems are totally subject to building rules and costs. Consequently, this component won't be tried by the product group. Be that as it may, it will be clarified to the structure specialists by telling them the best way to embed these qualities. Furthermore, the month to month information will likewise be put away in an arrangement with the goal that it tends to be removed for future ease of use.

# TESTING APPROACH

## Testing Levels:

**Test approach** is a document which contains a summary of all test activities and final test results of a testing project. Test approach is an assessment of how well the testing is performed. Based on the test report, stakeholders can evaluate the quality of the tested product and make a decision on the software release.

* **System testing:**  In this testing we test individual component. The aim is to test each part of the software by separating it. We check that component are fulfilling functionalities or not. This kind of testing is performed by developers.
* **Integrational testing:** In this testing we test integrated component. In this testing phase we use different software modules that are combined and tested as a group to make sure that integrated system is ready for system testing. We check the data flow from one module to other modules. This kind of testing is performed by testers.
* **System testing:**  In this testing phase we test the entire system. It tests the overall interaction of components. It this testing phase we check load, performance, reliability and security testing of the system.
* **Acceptance testing:** In this testing we test the final system. Acceptance testing is basically done by the user or customer. However, other stockholders can be involved in this process.

## Test Tools:

In our system we do different types of testing and those are unit testing, integration testing, System testing, acceptance testing. We use an automated testing tool do perform all the testing which is **selenium**.

## Meetings:

The test team will meet once every two weeks to evaluate progress to date and to identify error trends and problems as early as possible. The test team leader will meet with development and the project manager once every two weeks as well. These two meetings will be scheduled on different weeks. Additional meetings can be called as required for emergency situations.

# TEST CASES/TEST ITEMS:

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Apartment Service System | | | **Test Designed by:** Khalid | | |
| **Test Case ID:** TC\_1 | | | **Test Designed date:** 15 Dec 2020 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Imran | | |
| **Module Name:** System Login | | | **Test Execution date:**20 Dec 2020 | | |
| **Test Title:** verify login with valid username and password | | |  | | |
| **Description:** Test the software’s login page | | |  | | |
| **Precondition:** user has valid username and password  **Dependencies:** if any | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open the software. 2. Enter username 3. Enter password 4. Click submit | Username: admin Password: admin | User should login into the application | | User logged into the application | Pass |
| **Post Condition:** User is validated with database and successfully login to account. The account session details are logged in the database | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Apartment management system | | | **Test Designed by:** Khalid | | |
| **Test Case ID:** TC\_2 | | | **Test Designed date:** 15 Dec 2020 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Imran | | |
| **Module Name:** System Login | | | **Test Execution date:**20 Dec 2020 | | |
| **Test Title:** verify login with valid username and password | | |  | | |
| **Description:** Test the software’s login page | | |  | | |
| **Precondition:** user don’t have valid username and password  **Dependencies:** none | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open the software. 2. Enter username 3. Enter password 4. Click submit | Username: admin12 Password: admin12 | User shouldn’t login into the application | | User didn’t log into the application | Pass |
| **Post Condition:** User is validated with database and successfully login to account. The account session details are logged in the database | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Apartment management system | | | **Test Designed by:** Khalid | | |
| **Test Case ID:** TC\_3 | | | **Test Designed date:** 15 Dec 2020 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Imran | | |
| **Module Name:** Guest entry-exit record | | | **Test Execution date:**20 Dec 2020 | | |
| **Test Title:** verify guest info | | |  | | |
| **Description:** Test the software’s module | | |  | | |
| **Precondition:** user has valid username and password  **Dependencies:** none | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open the software. 2. Enter flat information 3. Click check | Flat: 2a-Shimu | Match found | | Match found | Pass |
| **Post Condition:** User is validated with database and successfully login to account. The account session details are logged in the database | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Apartment management system | | | **Test Designed by:** Khalid | | |
| **Test Case ID:** TC\_4 | | | **Test Designed date:** 15 Dec 2020 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Imran | | |
| **Module Name:** Guest entry-exit record | | | **Test Execution date:**20 Dec 2020 | | |
| **Test Title:** verify guest info | | |  | | |
| **Description:** Test the software’s module | | |  | | |
| **Precondition:** user has valid username and password  **Dependencies:** none | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open the software. 2. Enter flat information 3. Click check | Flat: 2a-Robin | Match not found | | Match found | Fail |
| **Post Condition:** User is validated with database and successfully login to account. The account session details are logged in the database | | | | | |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Project Name:** Apartment management system | | | **Test Designed by:** Khalid | | |
| **Test Case ID:** TC\_5 | | | **Test Designed date:** 15 Dec 2020 | | |
| **Test Priority (Low, Medium, High):** High | | | **Test Executed by:** Imran | | |
| **Module Name:** Service charge payment | | | **Test Execution date:**21 Dec 2020 | | |
| **Test Title:** check successful payment | | |  | | |
| **Description:** Test the payment gateway | | |  | | |
| **Precondition:** User has valid bank account and balance  **Dependencies:** none | | | | | |
| **Test Steps** | **Test Data** | **Expected Results** | | **Actual Results** | **Status (Pass/Fail)** |
| 1. Open the software. 2. Enter bank account info 3. Enter amount 4. Click pay | Name: DBBL  Acc:156985445333  Amount: 10000 | Success | | Success | Pass |
| **Post Condition:** User is validated with database and successfully login to account. The account session details are logged in the database | | | | | |

# ITEM PASS/FAIL CRITERIA:

The test process will be completed once the initial set of distributors have successfully sent in reassigned sales data for a period of one month and the new EDI data balances with the old ZIP/FAX data received in parallel. When the sales administration staff is satisfied that the data is correct the initial set of distributors will be set to active and all parallel stopped for those accounts.

# TEST DELIVERABLES:

**Test Deliverables in Software Testing:**

Test Deliverables play an important role in Software Testing. This article discusses all about Test Deliverables in detail.

Some of the important test deliverables are enlisted below for your reference:

**1. Test strategy**

**2. Test plan**

**3. Estimation**

**4. Test scenario**

**5. Test cases**

**6. Test data**

**7. Test summary report**

**8. Test closure report**

**9. Conclusion**

**Test Strategy:**

Test strategy will be decided based on the Business requirement specification. It is a vital document that contains all the details of the testing work to be carried on. It is a complete management document.

**Test Plan:**

The granular level details for each step of testing should be mentioned here. In general, a proper plan leads to a proper work structure. Similarly, a good plan leads to good testing.

**Estimation:** Estimation is defining how long each step will occur in testing along with the overall cost.

**Test Scenario:**

We will understand this with an example now. Let us take the train reservation as an example here. All the functionalities that we need to test are mentioned in high-level forms in the test scenario document. In simple words, it means a group of similar activities to be performed.

Two Techniques for the scenario:

**1) Use Case**

It is the goal-oriented method which is a set of interactions between the external factors and system. Its components include primary flow, alternate flow, triggers or activities, exception flows, pre-conditions, post-conditions, etc.

**2) ACE (Activity Component Element)**

The Activity Component Element process breaks the business requirements into activities.

In general, we book a ticket by filling the passenger details, gender, etc. Hence we need to validate the following fields which thereby become scenarios.

**Reservation:** Check the reservation functionality.

**Passenger details:** Check the gender, age, and sex fields functionality.

**Modify:** Check if the modify functionality works properly.

**Concession:** Check if the concession functionality works properly.

**View:** Check if the view functionality works properly.

**Cancel:** Check if the cancel functionality works properly.

**Test Case:**

By taking the same above example of the reservation page, the test cases are written as follows:

**Reservation:**

Check if the user can book a ticket by filling valid details in all the fields.

Check if the user can book a ticket by filling invalid details in all the fields.

Check if the user can book a ticket by leaving any blank field.

**Passenger details:**

Check if the user can book a ticket by entering a valid name.

Check if the user can book a ticket by entering an invalid name.

Check if the user can book a ticket by choosing any one gender at a time.

Check if the user can book a ticket by entering age greater than 60.

Check if the user can book a ticket by entering age less than 60.

Check if the user can book a ticket by entering any valid age greater than 5.

Check if the user is not able to book by entering age less than 5.

**Modify:**

Check if the user can modify the name field.

Check if the user can modify the gender field.

Check if the user can modify the age field.

**Concession:**

Check if the user can get concession by selecting the “Senior citizen” option.

Check if the user can get concession by selecting the “Handicapped/Disabled” option.

**View:**

Check if the user can view the ticket reserved.

**Cancel:**

Check if the user can cancel the ticket.

**Test Data:** Some project needs prior data from the client before proceeding with the execution of the test case. Test data need to be applied to perform testing.

**Test Summary Report:**

Test Summary Report summarizes all the test activities done and the test results are compiled in it. All the test information such as members involved in testing, objectives, scope, client details, test approach used, test results, defect report, etc should be mentioned here.

**Test Closure Report:**

It means that we are going to close the project after testing and defect fixing. Thus here we have to provide a detailed analysis of the execution of the tests.

**Conclusion:**

The artifacts that are sent to the stakeholders of a software project during the STLC are known as Test Deliverables. We had a look at the most important Test Deliverables in this article.

# STAFFING AND TRAINING NEEDS:

The stuffing of this project is also a vital things-To fulltime tester is preferable for the project for the system/integration and acceptance testing phases of the project .There can be an additional guy for the reviews. Five to six months is considered the approximate time for the project to be finished .If a separate test person is not available another test person/ project manager /test manager will perform the role .For the best service and properties the following areas are needed to be trained.

* The developers and tester(s) will need to be trained on the basic operations of the EDI interface. Prior to final acceptance of the project the operations staff will also require complete training on the EDI communications process.
* The sales administration staff will require training on the reports.

# RESPONSIBILITIES:

Tester to be responsible for designing and running tests on software usability. Tester will conduct tests, analyze the results, and report observations to the design team. Interact with clients in order to understand the requirements of the product.

To be successful as a Tester, should have a working knowledge of software and test design, the capability to run through tests, and the ability to analyze the results. Ultimately, the Tester should be result-driven, have good communication skills, and up-to-date knowledge of software programming and software test design.

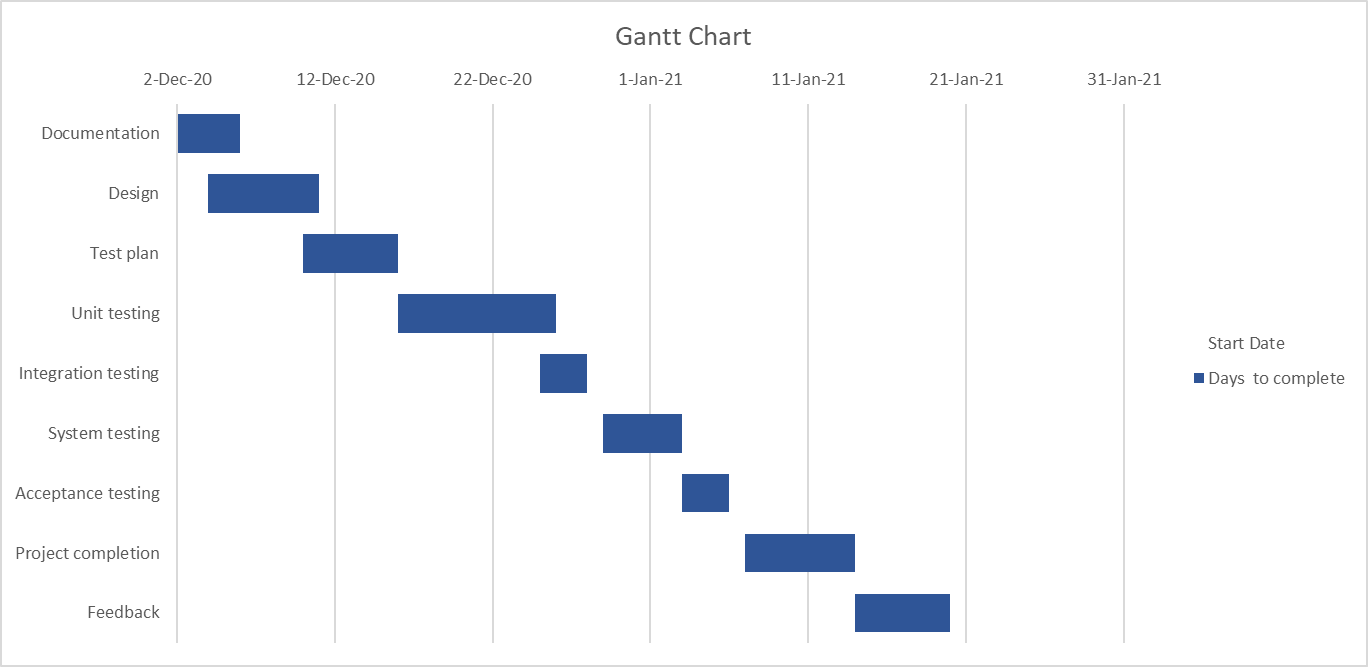
Tester Responsibilities:

* Reviewing software requirements and preparing test scenarios.
* Executing tests on software usability.
* Analyzing test results on database impacts, errors or bugs, and usability.
* Preparing reports on all aspects related to the software testing carried out and reporting to the design team.
* Interacting with clients to understand product requirements.

Participating in design reviews and providing input on requirements, product design, and potential problems.

# TESTING SCHEDULE:

Time has been allocated within the project plan for the following testing activities. The specific dates and times for each activity are defined in the project plan timeline. The persons required for each process are detailed in the project timeline and plan as well. Coordination of the personnel required for each task, test team, development team, management and customer will be handled by the project manager in conjunction with the development and test team leaders.



# PLANNING RISKS AND CONTINGENCIES:

* Lack of personnel resources when testing is to begin.
* Lack of availability of required hardware, software, data or tools.
* Late delivery of the software, hardware or tools.
* Delays in training on the application and/or tools.
* Changes to the original requirements or designs.
* Complexities involved in testing the applications

Specify what will be done for various events, for example: Requirements definition will be complete by December 20, 2020, and, if the requirements change after that date, the following actions will be taken:

* The test schedule and development schedule will move out an appropriate number of days. This rarely occurs, as most projects tend to have fixed delivery dates.
* The number of tests performed will be reduced.
* The number of acceptable defects will be increased.
* Resources will be added to the test team.
* The test team will work overtime.
* The scope of the plan may be changed.
* There may be some optimization of resources. This should be avoided, if possible, for obvious reasons.

Management is usually reluctant to accept scenarios such as the one above even though they have seen it happen in the past.

The important thing to remember is that, if you do nothing at all, the usual result is that testing is cut back completely, neither of which should be an acceptable option.

# APROVALS:

|  |  |
| --- | --- |
| Project Sponsor – A Sponsor |  |
| Development Management – R Manager |  |
| RS Test Manager – P Project |  |
| RS Development Team Manager – J Test |  |
| Reassigned Sales - C Sales |  |
| Order Entry EDI Team Manager – J Order |  |