



BSCS-S19-017

03-134161-054 NAJEEB AHSAN KHAN

03-134161-057 ASAD MAHMOOD

Automated Software Testing

In partial fulfilment of the requirements for the degree of
Bachelor of Science in Computer Science

Supervisor: Dawood Akram

Department of Computer Sciences
Bahria University, Lahore Campus

January 2020

Certificate



We accept the work contained in the report titled

“Automated Software Testing”

Written by

Najeeb Ahsan Khan

Asad Mahmood

as a confirmation to the required standard for the partial fulfilment of the degree of
Bachelor of Science in Computer Science.

Approved by:

Supervisor: Dawood Akram

(Signature)

January 27, 2020

DECLARATION

We hereby declare that this project report is based on our original work except for citations and quotations which have been duly acknowledged. We also declare that it has not been previously and concurrently submitted for any other degree or award at Bahria University or other institutions.

Enrolment	Name	Signature
03-134161-054	NAJEEB AHSAN KHAN	
03-134161-057	ASAD MAHMOOD	

Date : January 27, 2020

Specially dedicated to
My beloved grandmother, mother and father
(Najeeb Ahsan Khan)
My beloved grandmother, mother and father
(Asad Mahmood)

ACKNOWLEDGEMENTS

We would like to acknowledge our gratitude to our supervisor, Dawood Akram for his guidance, constant attention and personal concern towards the completion of this project. Furthermore, our appreciation also goes to our lab engineers from the department of computer science for their time, attention and guidance. We are also thankful to the evaluating team who evaluated our project with full dedication pinpointed our mistakes and guided as in right way during whole project.

In addition, we would also like to say thanks our loving parents and friends who had helped and given encouragement.

NAJEEB AHSAN KHAN

ASAD MAHMOOD

AUTOMATED SOFTWARE TESTING

ABSTRACT

Before diving into the structural behaviour of software industry like Arfa technology park it is required that software testing techniques plays vital role, just to make a distinguish environment testing was kind of a difficult in manual manners like making excel sheet and all. We will be making a system to overcome this problem, that will be marking a sign of automation system. SQA's department will be able to manage projects, test suits and test cases online and clients will be able to check deliverable's quality. Real time reporting will be available along with historical data. Rigid systems like windows and Ubuntu will be the upfront platform to use it. Sometimes, manual testing may not be so much effective and efficient due to its inconsistency, lack of coverage, lack of reporting time which may have bad impact on the cost of the product. In Pakistan, software industry is using excel sheets to manage test cases and this is really time consuming to design separate test cases and then generate the results on excel sheets. There is no proper way of archiving previous test execution results. This tool is required in our industry and as per our survey, people need this. We have gathered requirements from different software houses.

TABLE OF CONTENTS

DECLARATION	ii
ACKNOWLEDGEMENTS	iv
ABSTRACT	v
TABLE OF CONTENTS	vi
LIST OF TABLES	ix
LIST OF FIGURES	x
LIST OF SYMBOLS / ABBREVIATIONS	xii

CHAPTERS

1	INTRODUCTION	1
	1.1 Background	1
	1.2 Problem Statements	1
	1.3 Aims and Objectives	2
	1.4 Scope of Project	2
2	LITERATURE REVIEW	3
	2.1 Overall Description	3
	2.2 Operating Environment	3
	2.3 Design and implementation Constraints	3
	2.4 Assumptions and dependencies	3
	2.5 Other Non-functional Requirements	4

2.6	Other Requirements	4
2.7	System Requirements Chart	5
3	DESIGN AND METHODOLOGY	6
3.1	Design	6
3.2	Use Case Diagram	12
3.3	Domain Model	17
3.4	Sequence Diagrams	18
3.5	Class Diagram	22
3.6	Activity Diagram	22
3.7	Fully Attributed ERD	23
4	DATA AND EXPERIMENTS (and/or IMPLEMENTATION)	24
4.1	Methodology	24
4.2	Tools / Technology	25
5	RESULTS AND DISCUSSIONS (or USER MANUAL)	26
5.1	Register User	26
5.2	Login User	27
5.3	Main Page	27
5.4	Create New Project	28
5.5	View Test Suites	28
5.6	Create New Test Suite	29
5.7	View all Test Cases	29
5.8	Create New Test Case	30
5.9	New Test Run	30
5.10	Passed or Failed Test Cases	31
5.11	Graphical Report of Test Cases	31

6	CONCLUSION AND RECOMMENDATIONS	32
6.1	Recommendation	32
6.2	Conclusion	32
	REFERENCES	33

LIST OF TABLES

TABLE	TITLE	PAGE
	Table 2-1: System Requirement Chart	5
	Table 3-1: Registration	6
	Table 3-2: Login	7
	Table 3-3: Logout	7
	Table 3-4: Create Project	8
	Table 3-5: View All Projects	8
	Table 3-6: Create Suite	9
	Table 3-7: Visit Suite	9
	Table 3-8: Running of Suite	10
	Table 3-9: Edit Test Case	10
	Table 3-10: Remove test case	11

LIST OF FIGURES

FIGURE	TITLE	PAGE
Figure 3-1:	System Use Case	12
Figure 3-2:	Register	12
Figure 3-3:	Login	13
Figure 3-4:	Logout	13
Figure 3-5:	Create Project	14
Figure 3-6:	View All Projects	14
Figure 3-7:	Create Suite	15
Figure 3-8:	Visit Suite	15
Figure 3-9:	Running of Test Suite	16
Figure 3-10:	Edit Test Case	16
Figure 3-11:	Remove Test Case	17
Figure 3-12:	Domain Model	17
Figure 3-13:	Register Sequence	18
Figure 3-14:	Login Sequence	18
Figure 3-15:	Logout Sequence	19
Figure 3-16:	Creating Project Sequence	19
Figure 3-17:	Create test suite Sequence	20
Figure 3-18:	Creating test case Sequence	20
Figure 3-19:	Execute Test Case Sequence	21

Figure 3-20: Generate Report Sequence	21
Figure 3-21: Class	22
Figure 3-22: Activity	22
Figure 3-23: Fully Attributed ERD	23
Figure 4-1: Agile	24
Figure 5-1: Registration	26
Figure 5-2: Login	27
Figure 5-3: Main Page	27
Figure 5-4: Create Project	28
Figure 5-5: View Test Suites	28
Figure 5-6: Create New Test Suite	29
Figure 5-7: View all Test Cases	29
Figure 5-8: Create Test	30
Figure 5-9: Test Run	30
Figure 5-10: Test Cases Execution results	31
Figure 5-11: Graph Execution Report	31

LIST OF SYMBOLS / ABBREVIATIONS

<i>XLS</i>	Excel Spreadsheet
<i>ROR</i>	Ruby on Rails
<i>QA</i>	Quality Assurance
<i>GUI</i>	Graphical User Interface
<i>HW</i>	Hardware
<i>SW</i>	Software

CHAPTER 1

INTRODUCTION

1.1 Background

Testing is an important part of the software development cycle. It is not uncommon for developers to make errors when writing code. These mistakes can cause the web server to crash impacting thousands of clients. To detect these kinds of errors we are building an automated testing software in which we can perform testing of software's both manually and automatically. The previous versions of the software are manually operated, and they generated results on Excel (XLS) sheets that is the major fault. Manual system has two major drawbacks one is costly and the second one is more time consuming and most of these testing tools have domains on the internet. But our software will be totally free of cost and it will not generate results on XLS sheets, but it will generate results on one click automatically. We will use ROR on mvc framework for backend and for database connectivity we will use MySQL and for frontend we will use react/bootstrap/GUI.

1.2 Problem Statements

To automate the manual testing software for the QA'S

1.3 Aims and Objectives

The objectives of the thesis are shown as following:

- i) Our objective is to make such a web base software for testing purpose that can make environment so feasible that the testing of software can be easily done automatically and that can maintain test cases and their execution to check their functionalities and all.
- ii) Our target is to provide free testing service in minimum time span. We can also maintain historical data.

1.4 Scope of Project

Our project is purely based on software testing program or simulation which will do at the end software testing automatically and manually. Before In this Era it is a kind of difficult for every QA job to do testing manually as there may come errors in the website and QA may be unable to detect these errors and may lead to website crashes. There are some software available to test or find bugs in the websites or software but the available testing tools are not free of cost. So, we decided to work it that will find bugs, track record, testing automation and this software must be free cost and it will generate our result automatically on our one click.

CHAPTER 2

LITERATURE REVIEW

2.1 Overall Description

2.1.1 User Classes and Characteristics

We have different classes in this project like user, project, test suit, test cases, test run, execution class inheritance will use extensively use in this project because of the test cases. And the main user in this case is tester.

2.2 Operating Environment

A Web based platform which can run easily on every operating system including PC and laptops.

2.3 Design and implementation Constraints

Automated software testing is developed in react JS/bootstrap language in sublime platform. In our project we use react for frontend and for the backend we will use ruby on rails and for other Web content to a user language like Bootstrap, JavaScript, and jQuery. We will provide a testing platform.

2.4 Assumptions and dependencies

We use react for frontend and for the backend we will use ruby on rails, and we will be deployed it on the heroku cloud. All these tools are dependencies and we are depending on this these tools

2.5 Other Non-functional Requirements

2.5.1 Performance Requirements

Load time should be under approximately 5-10 seconds

2.5.2 Safety Requirements

Project data should not viewable without Authorization.

2.5.3 Security Requirements

Data should be same no sql injection no vulnerability.

2.5.4 Software Quality Attributes

Software quality attributes that needs to be addressed are:

- Testing should become flexible.
- Less time consuming and more efficient.
- Accurate result generator.

2.6 Other Requirements

Use my sql and all open source technologies.

2.7 System Requirements Chart

Table 2-1: System Requirement Chart

ID	Priority	Type	Used use cases	Description
1	High	Functional	U1	This user needs to register for testing the test suits.
2	High	Functional	U2	Only registered users can login in order to use the tool.
3	High	Functional	U3	Create Project
4	Medium	Functional	U4	User should be able to view the previous projects.
5	High	Functional	U5	Create Suite
6	Medium	Functional	U6	Visit Suite
7	High	Functional	U7	Running of Suite
8	High	Functional	U8	Edit Test Case
9	High	Functional	U9	Remove Test Case
10	High	Functional	U10	Logout

CHAPTER 3

DESIGN AND METHODOLOGY

3.1 Design

3.1.1 Use case description

3.1.1.1 Registration <U1>

Table 3-1: Registration

Unique Identifier	U1
Brief description	This use case describes that the tester needs to register for testing the test suits.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	<ul style="list-style-type: none"> • User will open the tool. • User will fill the required fields. • Click on the register button.
ii) Alternative Flow 1	<ul style="list-style-type: none"> • User cannot register again with same data. • If user already exist, then user will move to login screen.
iii) Alternative Flow 2	If all the required fields are not filled, an error message will be displayed.
Preconditions	User must have connectivity with the internet.
Post conditions	User successfully registered.

3.1.1.2 Login <U2>

Table 3-2: Login

Unique Identifier	U2
Brief description	Now registered users can login in order to use the tool.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	<ul style="list-style-type: none"> • User enter the name and password. • If the enter information is correct, then user move to main screen of their account.
ii) Alternative Flow 1	If the user record does not exist, then the user asked to register first.
iii) Alternative Flow 2	If the user provides the wrong information, then the user asked to reenter the information.
Preconditions	<ul style="list-style-type: none"> • User must be registered. • User must have connectivity with the internet.
Post conditions	User successfully logged in.

3.1.1.3 Logout <U3>

Table 3-3: Logout

Unique Identifier	U3
Brief description	After login if the user has done with testing and want to end the session then user can logout.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	User clicks on the logout.
Preconditions	<ul style="list-style-type: none"> • User must be registered. • User must have connectivity with the internet.
Post conditions	User successfully logout.

3.1.1.4 Create Project <U4>

Table 3-4: Create Project

Unique Identifier	U4
Brief description	<ul style="list-style-type: none"> • Asked the project details like name and description of the project. • After entering the project click on add project button. If the user wants to cancel the project, then the user may select the cancel button.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	<ul style="list-style-type: none"> • User enters the project title and then briefly describe the project. • If the provided data is correct, then the user will allow to move on the next phase.
ii) Alternative Flow	<ul style="list-style-type: none"> • User need to provide to both name and description. • If the user misses any one of these then the process will not move further.
Preconditions	User must be logged in and having connection with the internet.
Post conditions	User successfully create the project.

3.1.1.5 View All Projects <U5>

Table 3-5: View All Projects

Unique Identifier	U5
Brief description	User should be able to view the previous projects.
Priority	Medium
Actors	Tester
Flow of events	
i) Basic Flow	User also search the specific project by entering the project title
ii) Alternative Flow	If user wants to view specific project, then user need to enter the title of the specific project if he wants to view all the project then must click on all project option.
Preconditions	The searching project needs to be existed in the tool.
Post conditions	Project successfully searched.

3.1.1.6 Create Suite <U6>

Table 3-6: Create Suite

Unique Identifier	U6
Brief description	User first create the test suits and then enter the test cases in the test suits.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	User select the test case option and then select the new option from menu bar, then select the test suits and create new test cases.
ii) Alternative Flow	If the test cases are defective, then test suits will not be created.
Preconditions	If the test case hierarchy does not create properly then it will generate error message.
Post conditions	Test suite created successfully.

3.1.1.7 Visit Suite <U7>

Table 3-7: Visit Suite

Unique Identifier	U7
Brief description	User can visit the created suite.
Priority	Medium
Actors	Tester
Flow of events	
i) Basic Flow	User can visit the created test suits.
Preconditions	Before visiting the test, suits need to be created.
Post conditions	User successfully visits the test suits.

3.1.1.8 Running of Suite <U8>

Table 3-8: Running of Suite

Unique Identifier	U8
Brief description	User can run the created test suits by selecting them one by one
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	Selected suits will appear one by one on the screen and will execute them.
ii) Alternative Flow	Mark as pass, failed or skip by the user.
Preconditions	Test suits need to be created.
Post conditions	Shows result that how many test cases have been passed, failed or skip by the user.

3.1.1.9 Edit Test Case <U9>

Table 3-9: Edit Test Case

Unique Identifier	U9
Brief description	User will edit the test case in which error have been find during the time of running
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	User will remove errors one by one from the test case that have been occurred while at the time running
ii) Alternative Flow	Marked them as a passed or skip after removing errors
Preconditions	Test case must have run before the editing of the test case
Post conditions	Results will come out

3.1.1.10 Remove Test Case <U10>

Table 3-10: Remove test case

Unique Identifier	U10
Brief description	User will remove the extra use case or that have been skipped at the time of running.
Priority	High
Actors	Tester
Flow of events	
i) Basic Flow	If test case will not working even after removing the errors or user feels that the test case is extra, then it has the authority to remove that test case.
ii) Alternative Flow	If test case will start working properly then marked the test case as passed
Preconditions	Test case must have any error
Post conditions	Successful should marked as a passed and the test cases which have error or not working properly should marked as failed and remove that test case

3.2 Use Case Diagram

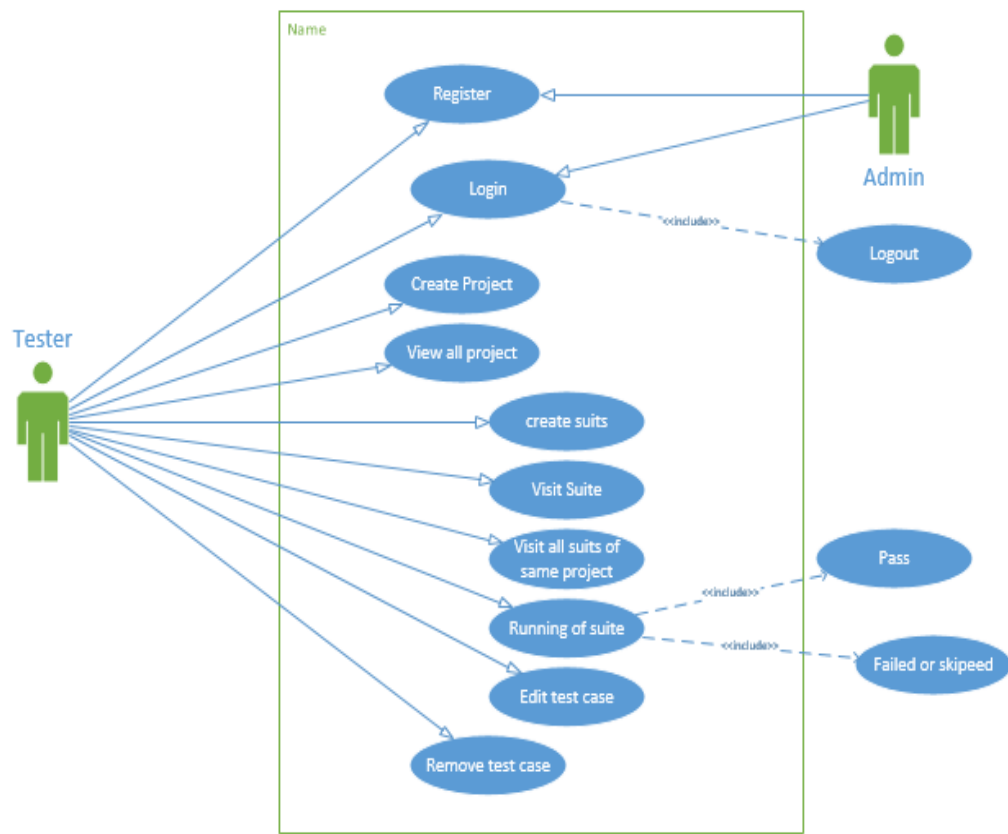


Figure 3-1: System Use Case

3.2.1 Register (U1)

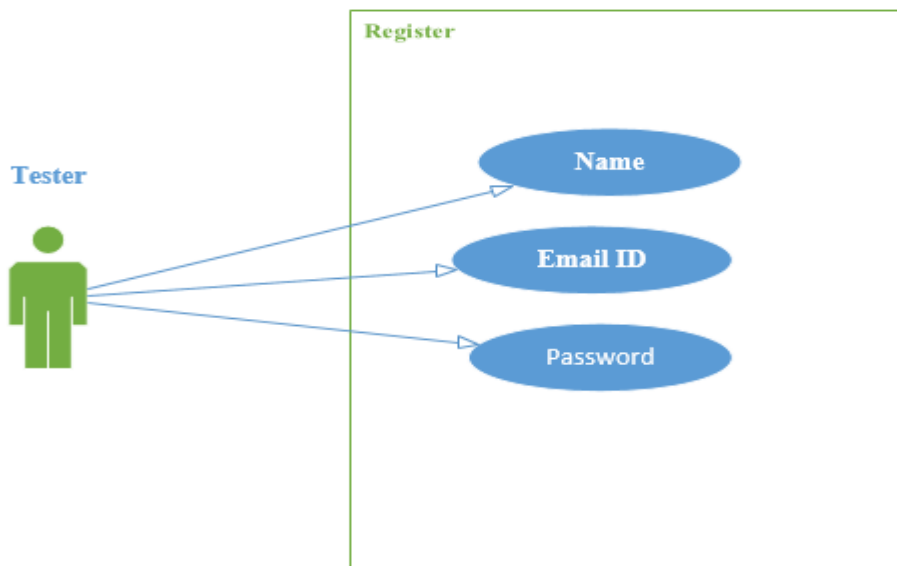


Figure 3-2: Register

3.2.2 Login (U2)

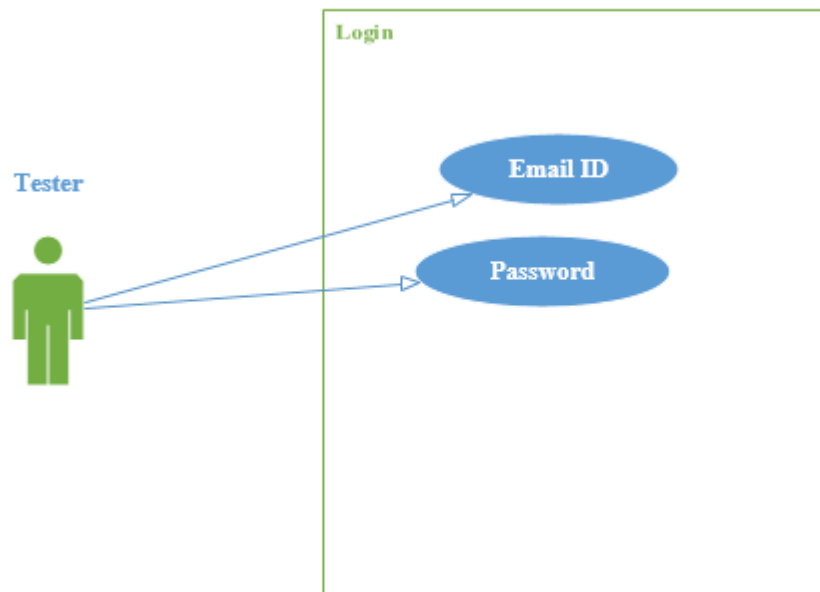


Figure 3-3: Login

3.2.3 Logout (U3)

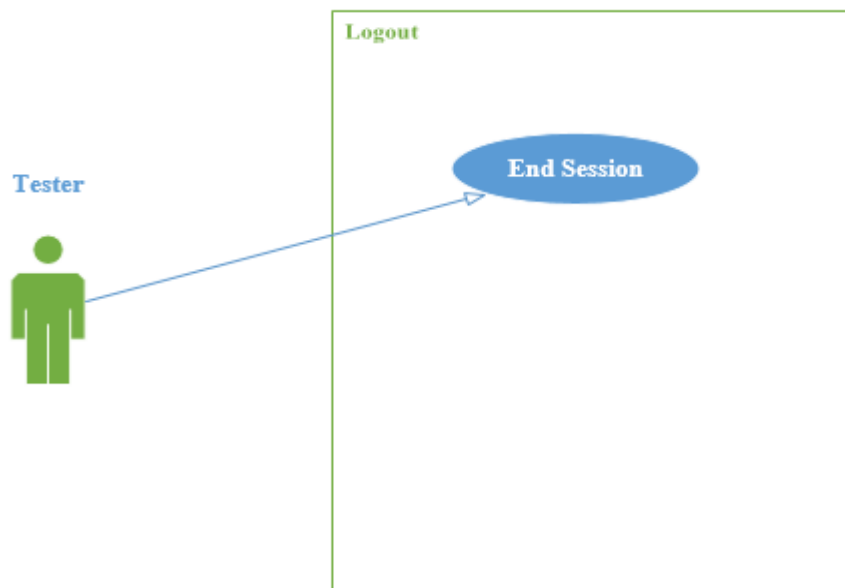


Figure 3-4: Logout

3.2.4 Create Project (U4)

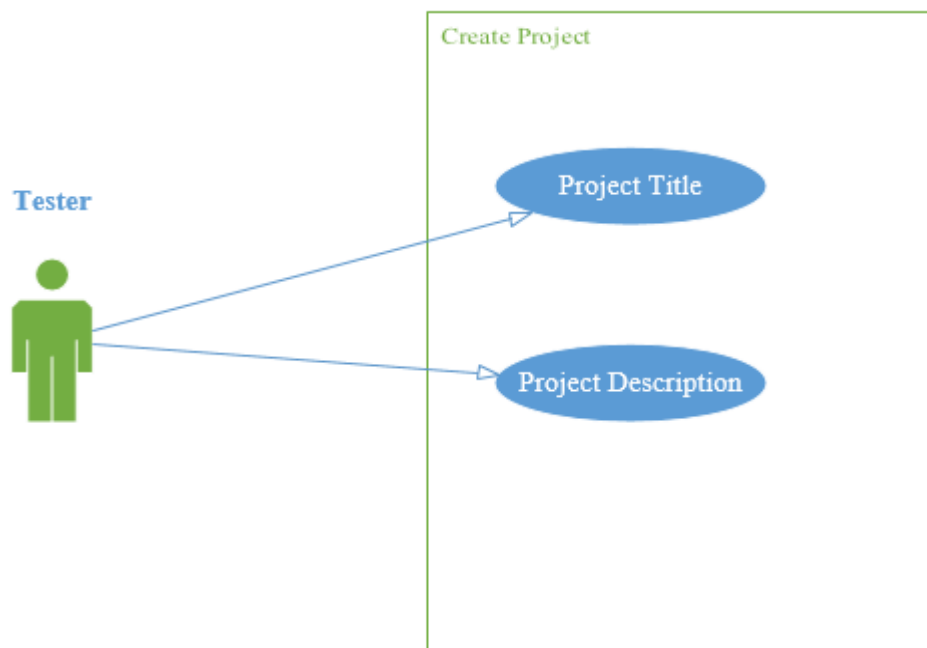


Figure 3-5: Create Project

3.2.5 View All Projects (U5)

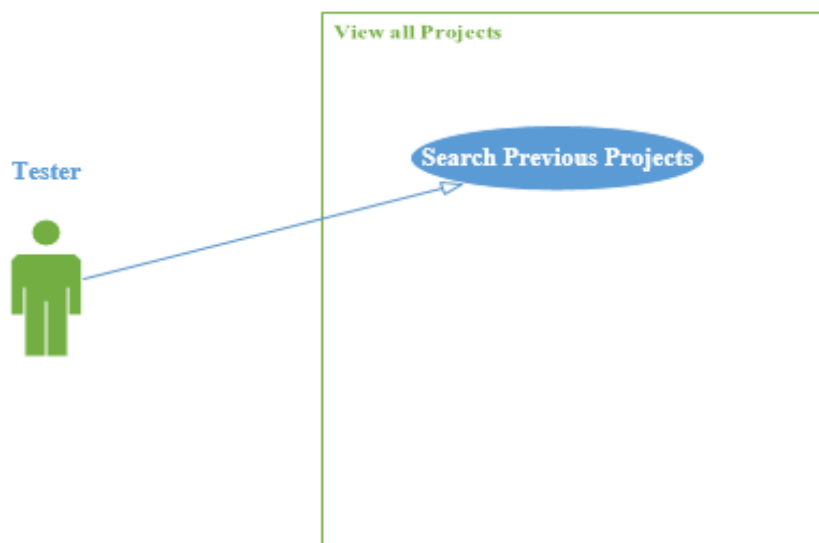


Figure 3-6: View All Projects

3.2.6 Create Suite (U6)

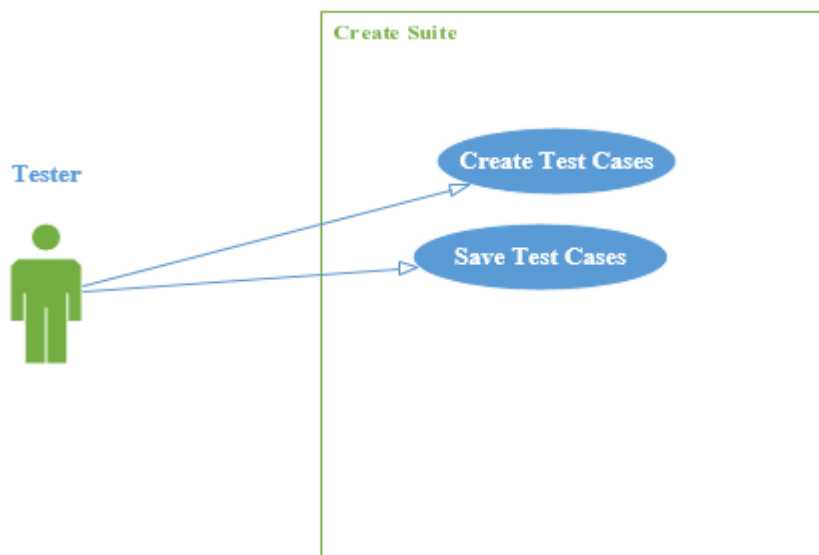


Figure 3-7: Create Suite

3.2.7 Visit Suite (U7)

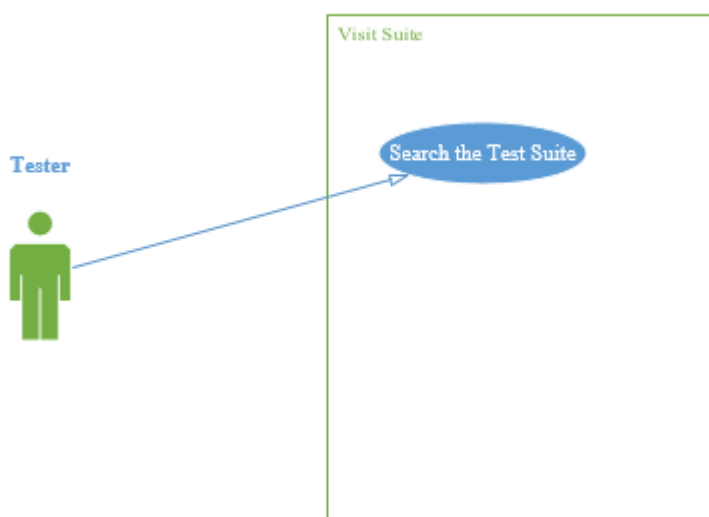


Figure 3-8: Visit Suite

3.2.8 Running of Test Suite (U8)

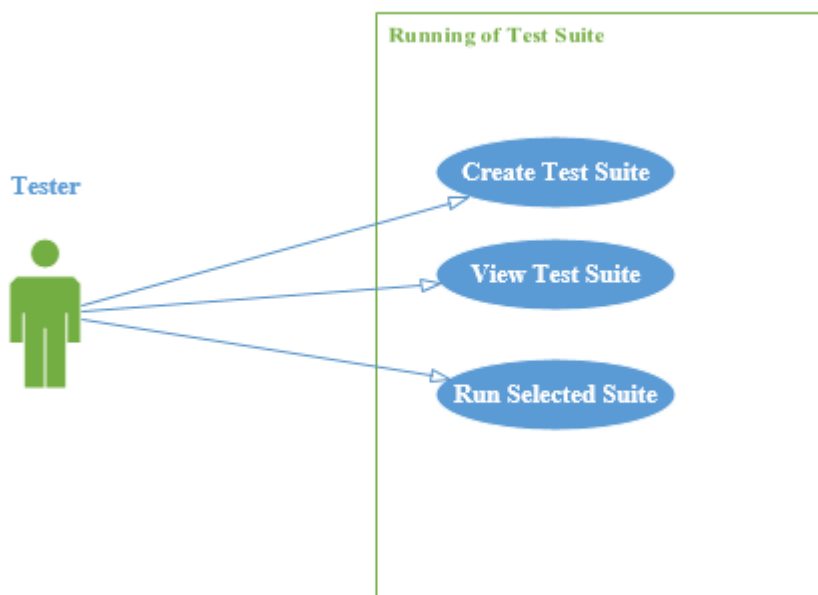


Figure 3-9: Running of Test Suite

3.2.9 Edit Test Case (U9)

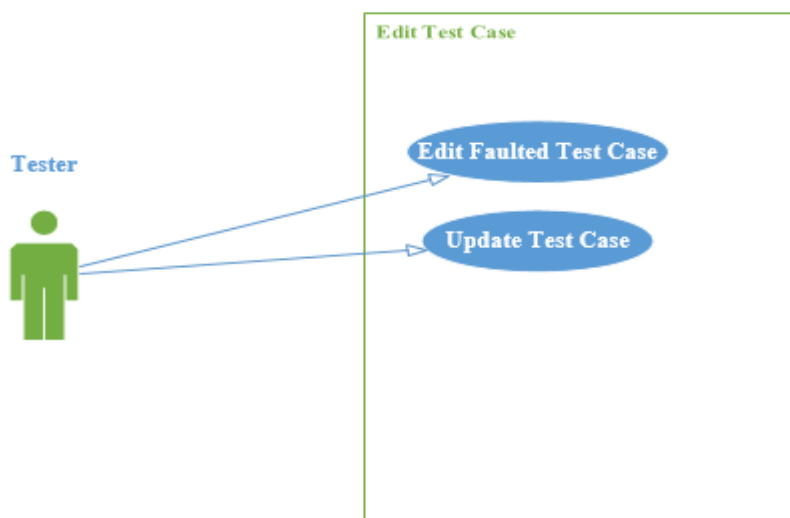


Figure 3-10: Edit Test Case

3.2.10 Remove test case (U10)

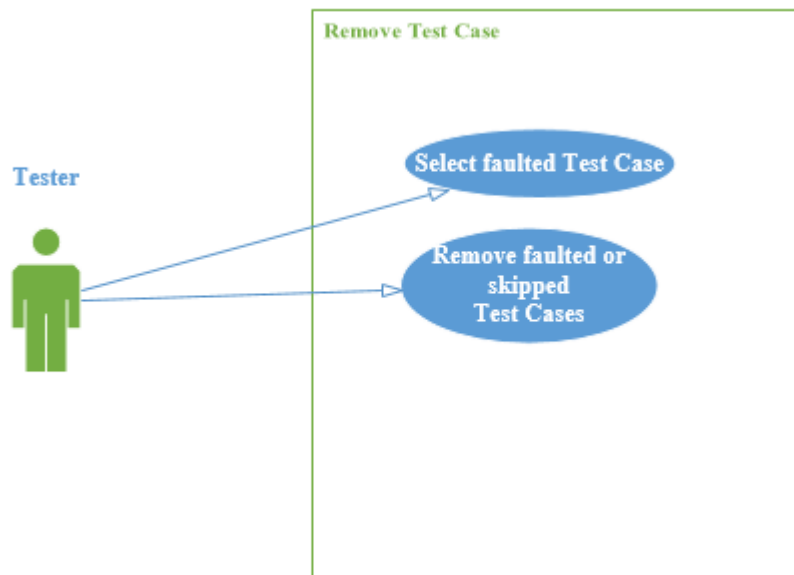


Figure 3-11: Remove Test Case

3.3 Domain Model

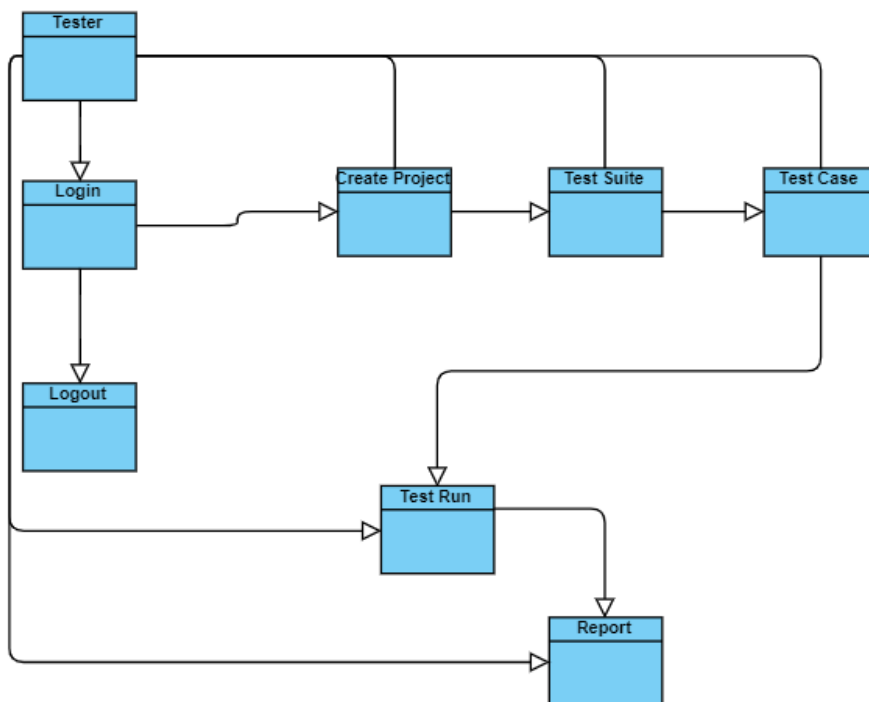


Figure 3-12: Domain Model

3.4 Sequence Diagrams

3.4.1 Registration

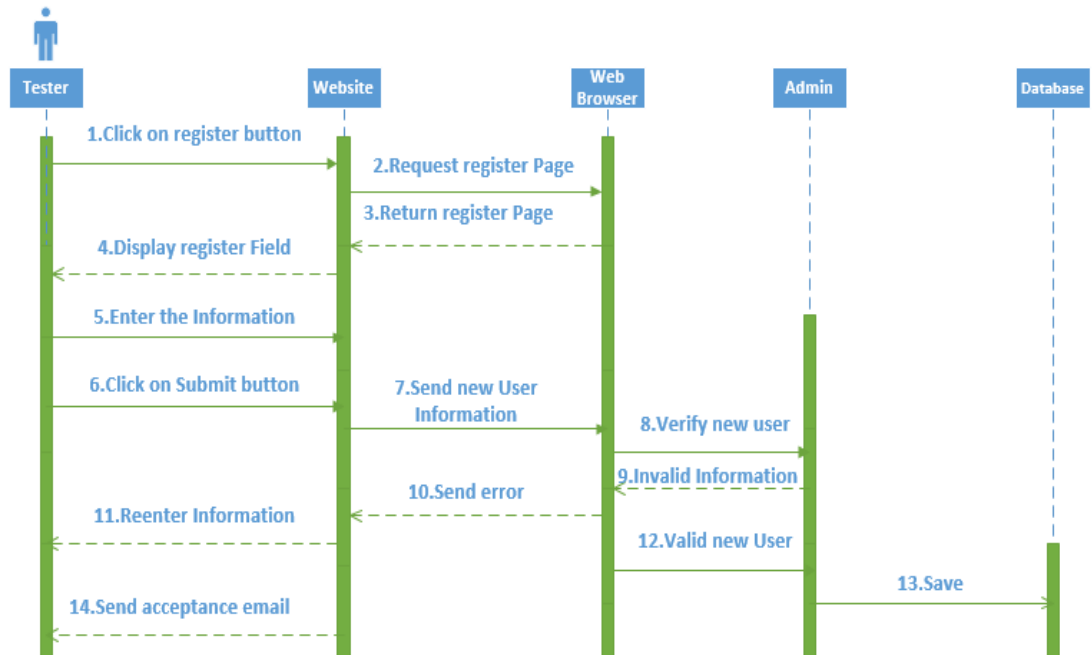


Figure 3-13: Register Sequence

3.4.2 Login

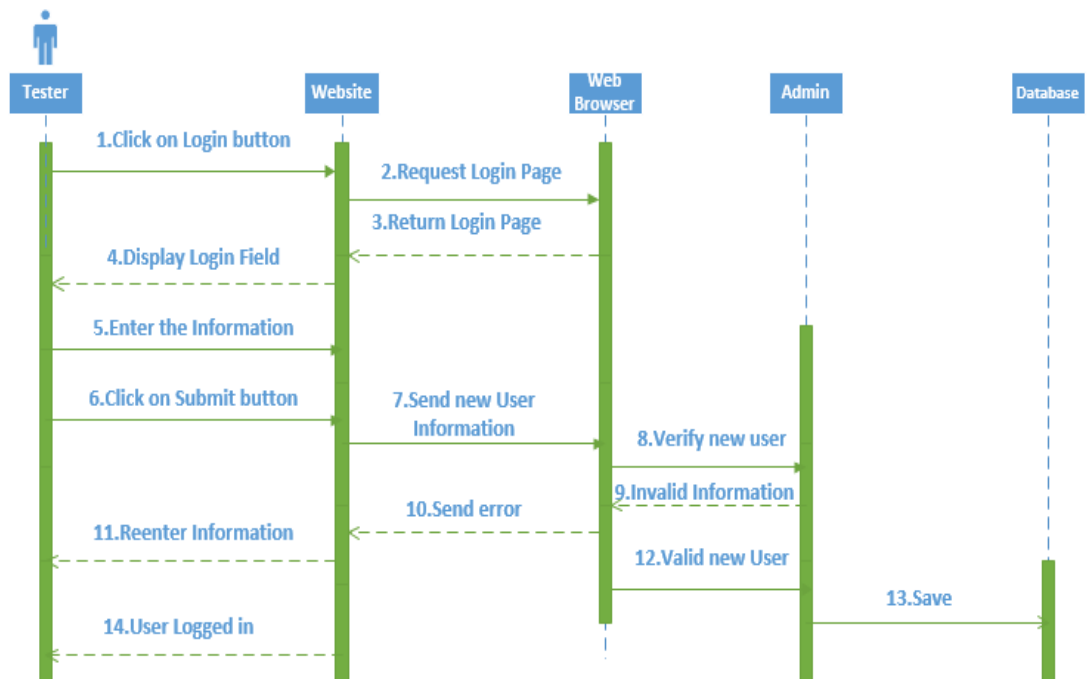


Figure 3-14: Login Sequence

3.4.3 Logout

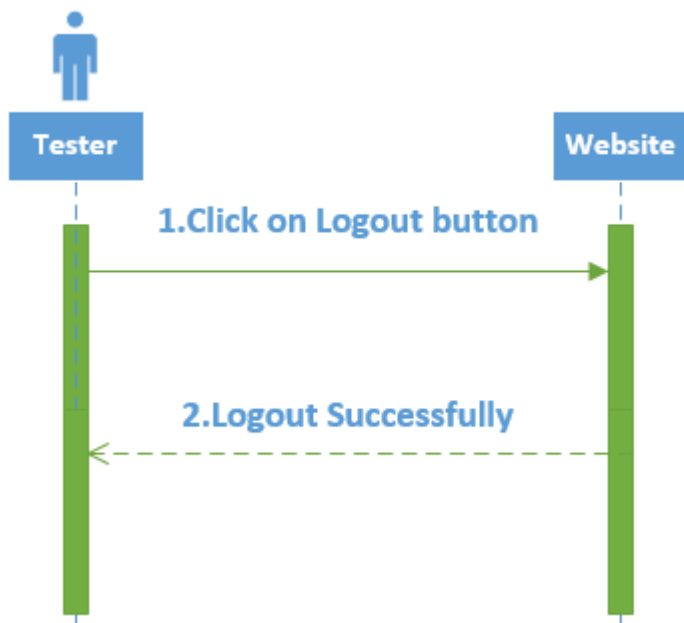


Figure 3-15: Logout Sequence

3.4.4 Creating Project

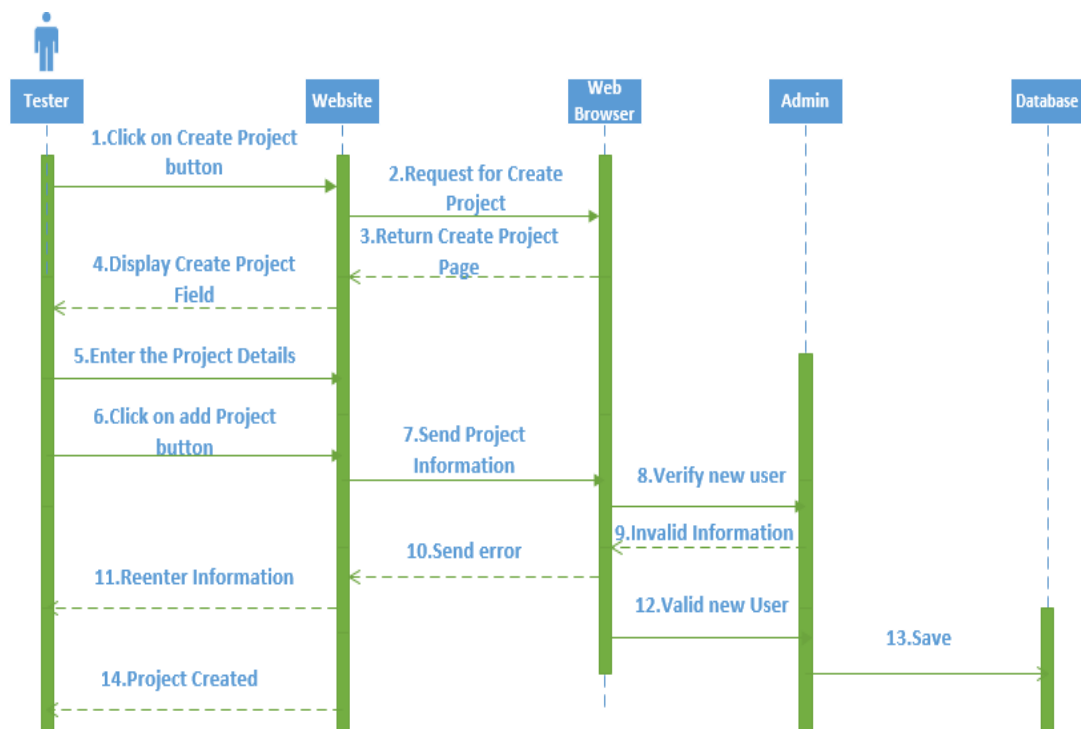


Figure 3-16: Creating Project Sequence

3.4.5 Create Test Suite

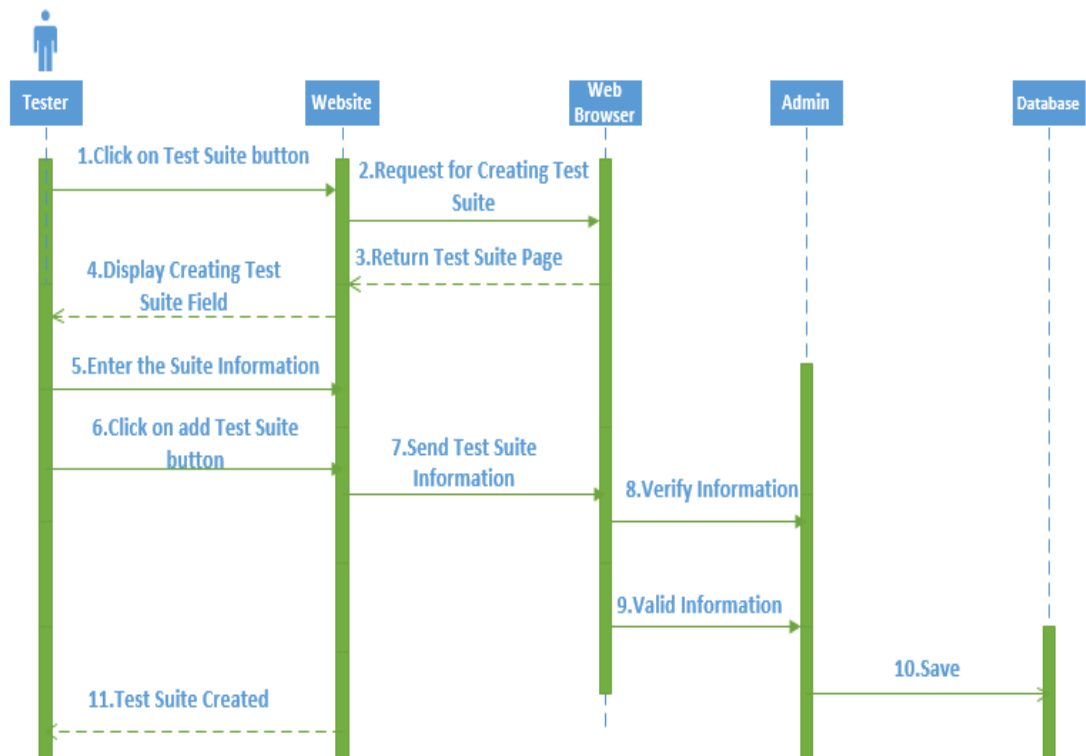


Figure 3-17: Create test suite Sequence

3.4.6 Create Test Case

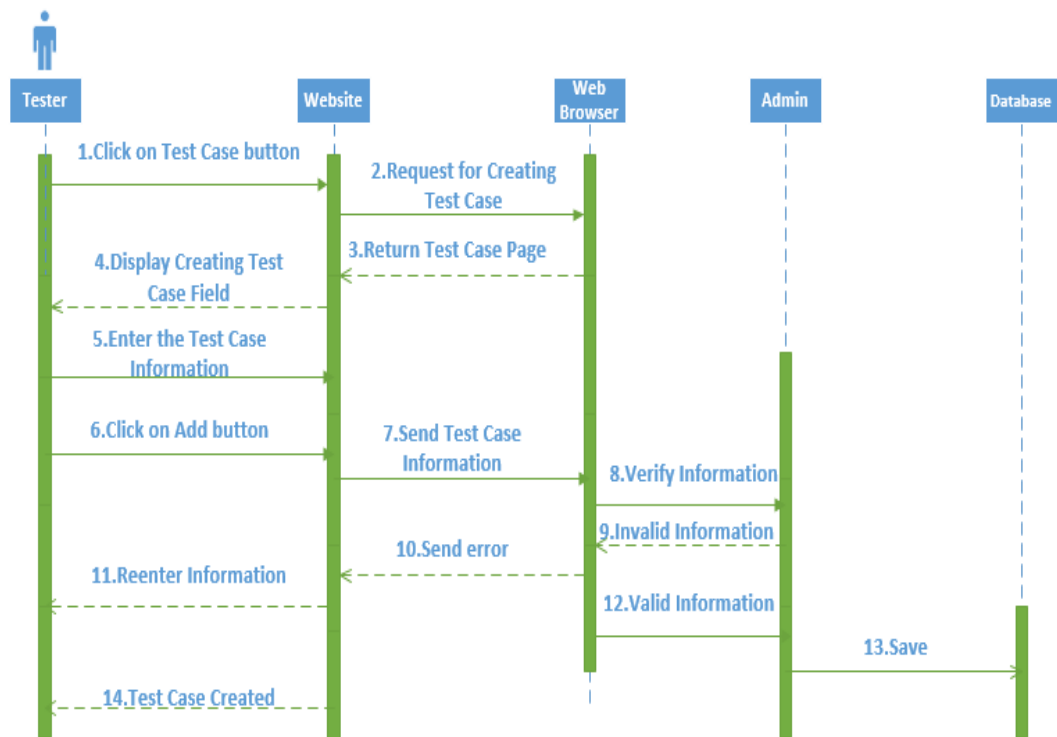


Figure 3-18: Creating test case Sequence

3.4.7 Execute Test Case

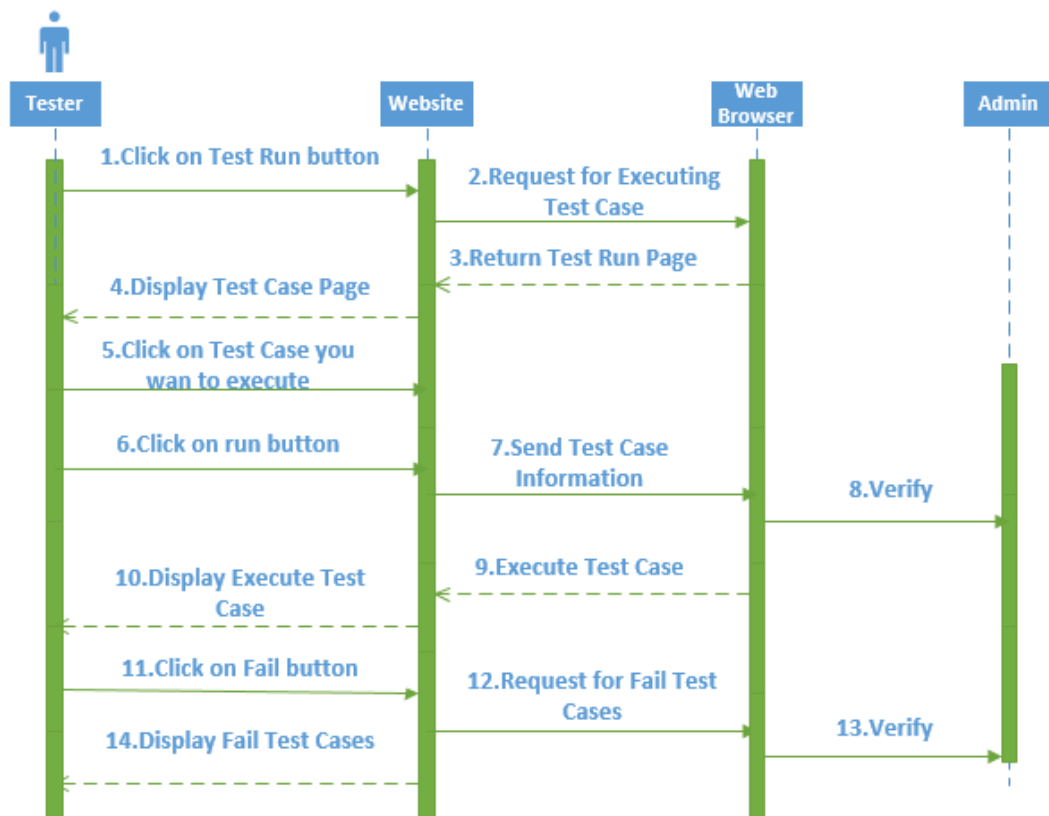


Figure 3-19: Execute Test Case Sequence

3.4.8 Generate Report

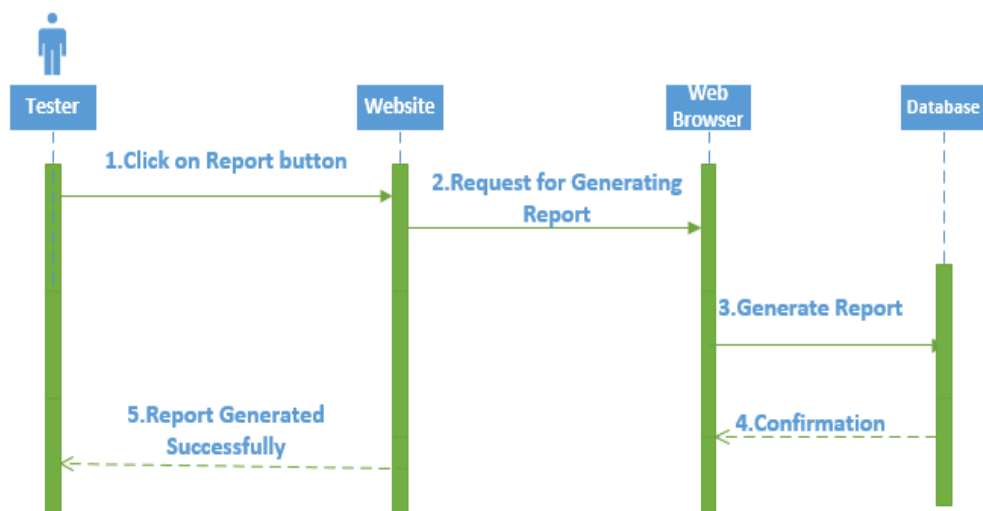


Figure 3-20: Generate Report Sequence

3.5 Class Diagram

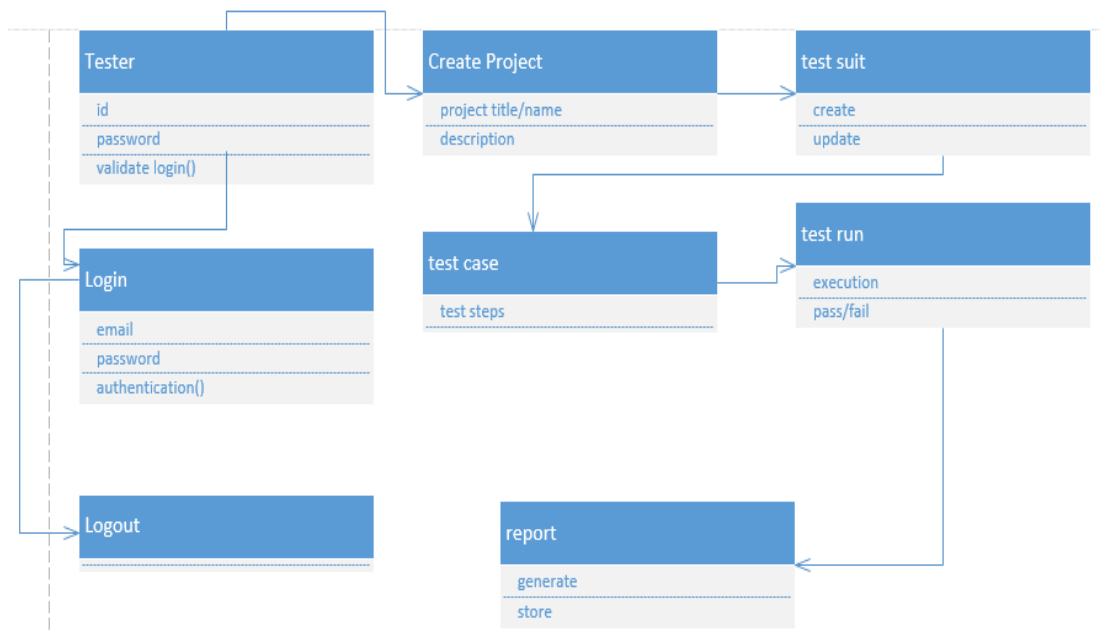


Figure 3-21: Class

3.6 Activity Diagram

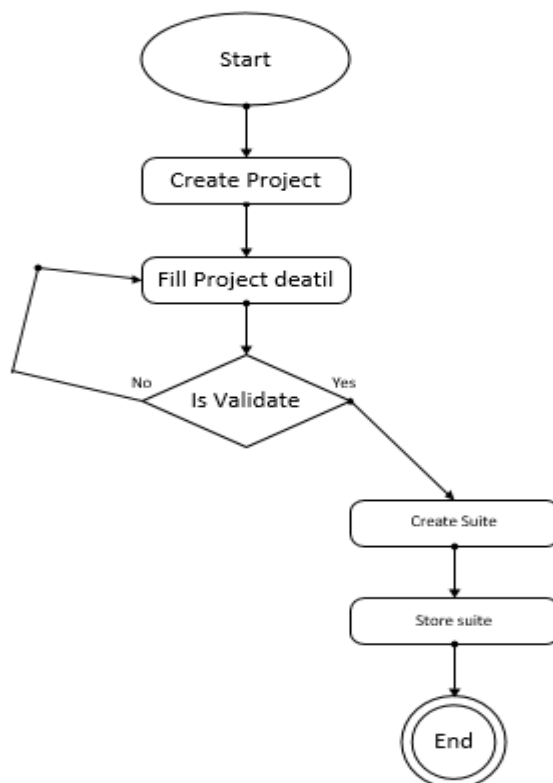


Figure 3-22: Activity

3.7 Fully Attributed ERD

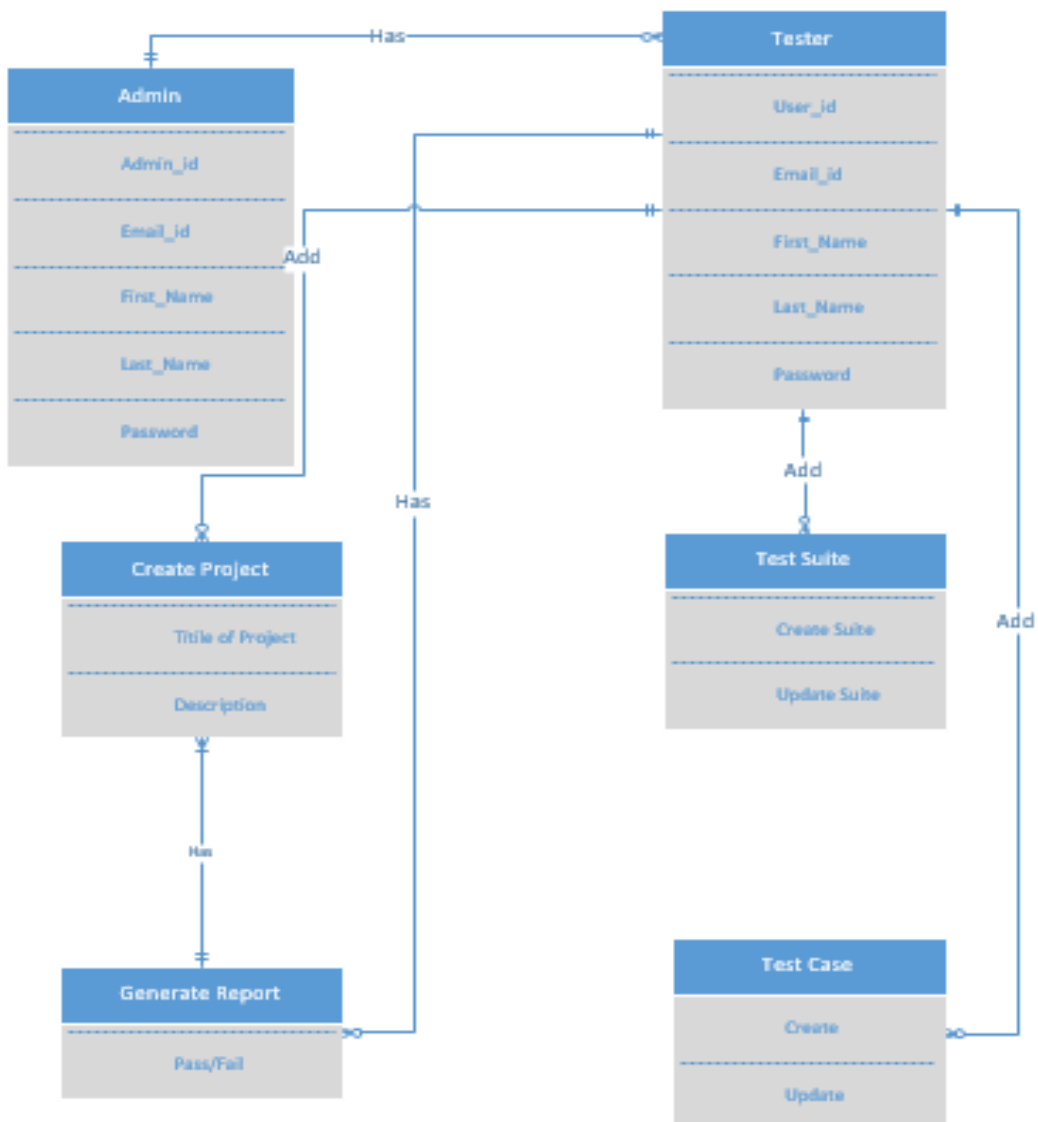


Figure 3-23: Fully Attributed ERD

CHAPTER 4

DATA AND EXPERIMENTS (and/or IMPLEMENTATION)

4.1 Methodology

We will use agile methodology in this project because we are working on review-based teaching method, so we need to actively involve our customer (user) after every iteration. After every development iteration, customer can see the result and understand if he/she is satisfied with that functionality or not. It advocates planning, development, early delivery, and continual improvement, and it encourages rapid and flexible response to change. It is a lightweight process framework and the most widely used one. We will gather requirements from industry survey, document all requirements, implement using ROR, MYSQL, Bootstrap and jQuery/GUI.

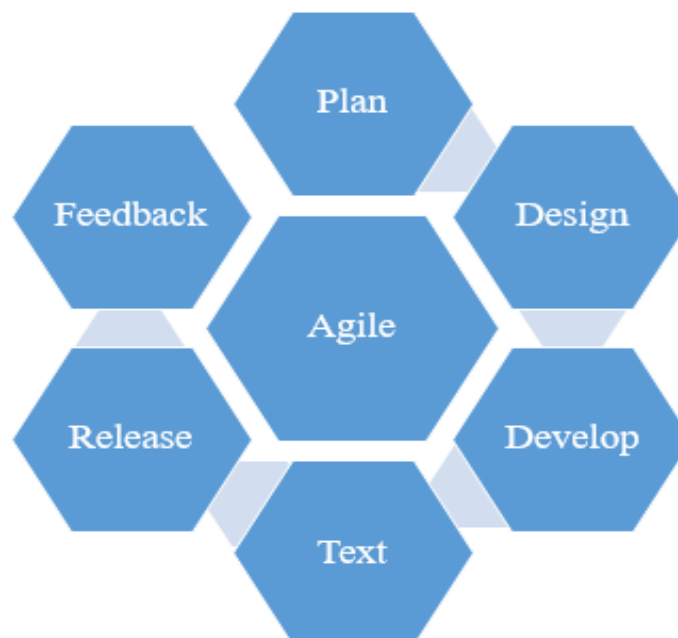


Figure 4-1: Agile

4.2 Tools / Technology

Mention all the HW/SW tools/technologies required for the project along with their availability.

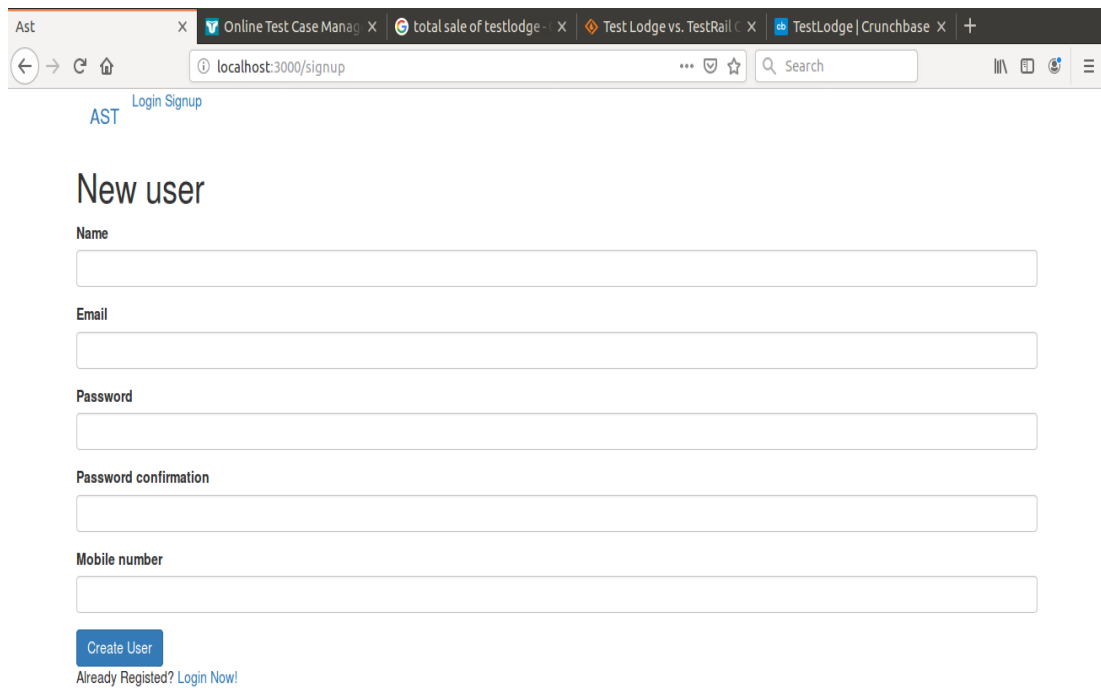
These are the tools required for the completion of this project

- MySQL (For Database)
- ROR (For backend)
- Bootstrap (For frontend)
- Ram 8GB
- Hardisk 500GB
- Processor: Intel(R) Core (TM) i7-7500U CPU @ 2.70GHz 2.90GHz

CHAPTER 5

RESULTS AND DISCUSSIONS (or USER MANUAL)

5.1 Register User



The screenshot shows a web browser window with the address bar displaying "localhost:3000/signup". The page content includes a "Login Signup" link, the text "AST", and a "New user" heading. Below the heading are five input fields labeled "Name", "Email", "Password", "Password confirmation", and "Mobile number". At the bottom of the form is a blue "Create User" button and a link that says "Already Registered? Login Now!".

AST [Login Signup](#)

New user

Name

Email

Password

Password confirmation

Mobile number

[Create User](#)

[Already Registered? Login Now!](#)

Figure 5-1: Registration

5.2 Login User

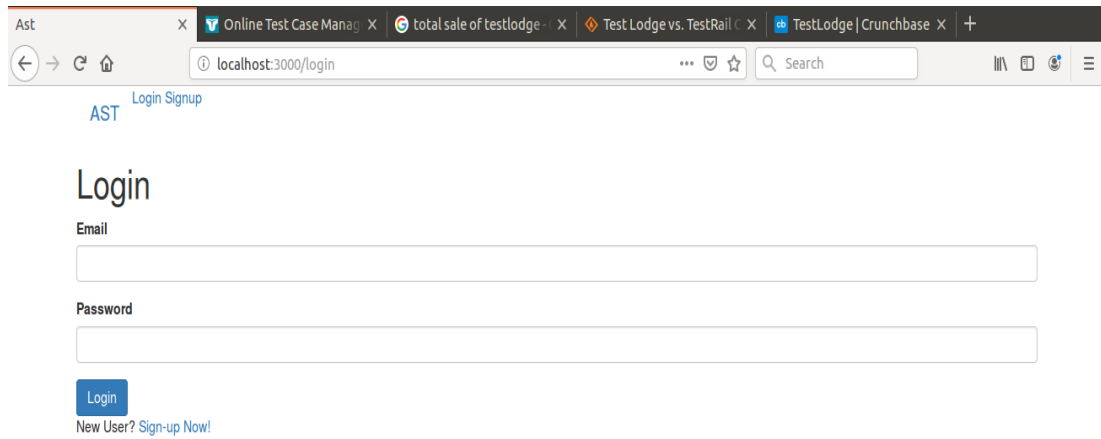


Figure 5-2: Login

5.3 Main Page

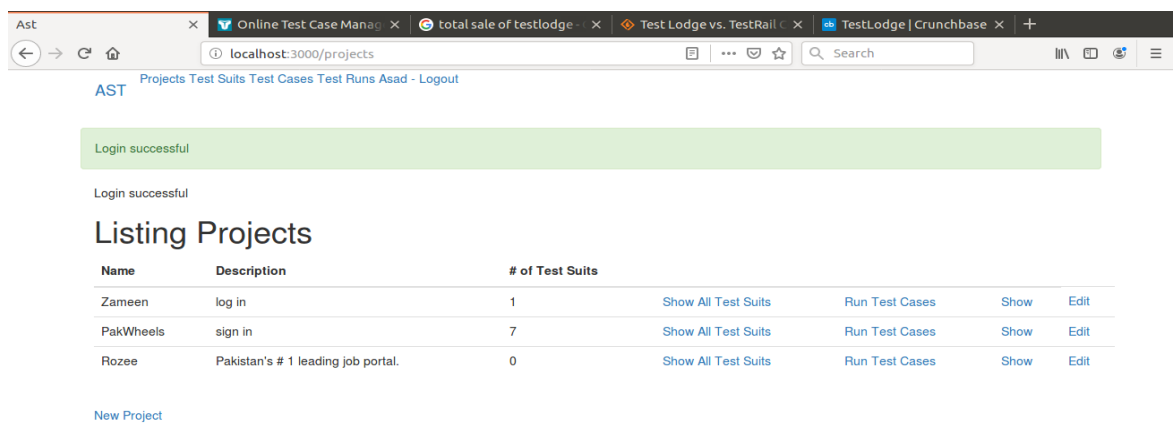


Figure 5-3: Main Page

5.4 Create New Project

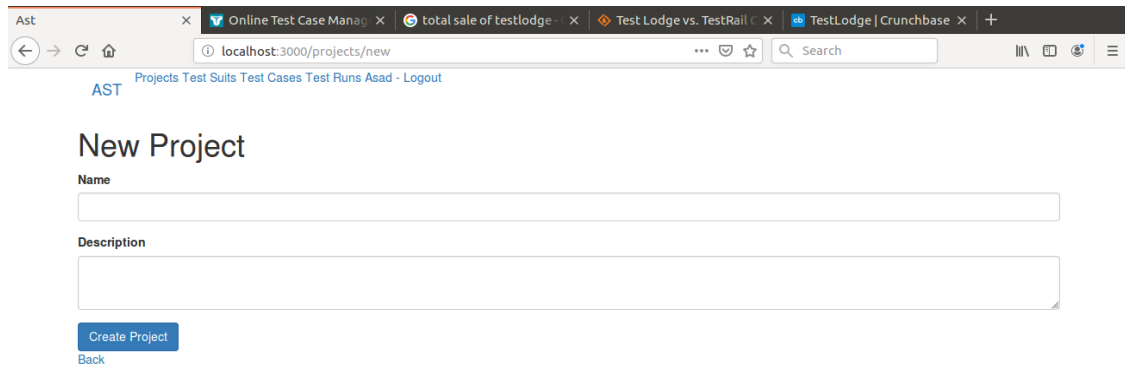


Figure 5-4: Create Project

5.5 View Test Suites

AST [Projects](#) [Test Suites](#) [Test Cases](#) [Test Runs](#) [Asad](#) - [Logout](#)

Test Suites

Name	Description	Project			
Search properties		Zameen	Show All Test Cases	Show	Edit
Authentication	To test authentication	PakWheels	Show All Test Cases	Show	Edit
LOGIN		PakWheels	Show All Test Cases	Show	Edit
login with irrelevant data		PakWheels	Show All Test Cases	Show	Edit
Check set Role CanExportAllCarrierChoices		PakWheels	Show All Test Cases	Show	Edit
Check Template of Excel when Export all Carriers		PakWheels	Show All Test Cases	Show	Edit
Check value of Notes/COS Selected column		PakWheels	Show All Test Cases	Show	Edit
Check value of columns that provider supports		PakWheels	Show All Test Cases	Show	Edit

[New Test suit](#)

Figure 5-5: View Test Suites

5.6 Create New Test Suite

AST [Projects](#) [Test Suites](#) [Test Cases](#) [Test Runs](#) [Asad](#) - [Logout](#)

New Test Suit

Name

Description

Project

[Create Test suit](#)

[Back](#)

Figure 5-6: Create New Test Suite

5.7 View all Test Cases

AST [Projects](#) [Test Suites](#) [Test Cases](#) [Test Runs](#) [Asad](#) - [Logout](#)

Listing Test Cases

Name	Testing steps	Test suit	
Splash screen should appear	- close the app if already opened. 2. re open app and splash should appear.	Search properties	Show Edit Destroy
login should work fine	1. go to login screen. 2. give valid username and password as input in login form 3. press login button output should be login state with success message	Authentication	Show Edit Destroy
User should be able to sign up	1: Go to the system TestProEngine with Classic or Current Mode that has to set role "CanExportAllCarrierChoices" 2: Submit for the quote 3: Open the quote at the home 4: Go to Quick Links	Authentication	Show Edit Destroy
Check Customer Login with valid Data	1.Enter User Id 2.Enter Password 3.Click login	LOGIN	Show Edit Destroy
Check Customer Login with invalid Data	1.Enter User Id 2.Enter Password 3.Click Submit	login with irrelevant data	Show Edit Destroy
Checking new function is added in Classic or Current Mode	1: Go to the system Test ProEngine with Classic or Current Mode that has to set role "Can Export All Carrier Choices" 2: Submit for the quote	Check set Role CanExportAllCarrierChoices	Show Edit Destroy

Figure 5-7: View all Test Cases

5.8 Create New Test Case

AST [Projects](#) [Test Suites](#) [Test Cases](#) [Test Runs](#) Asad - Logout

New Test Case

Name

Testing steps

Test suit

[Create Test case](#)

[Back](#)

Figure 5-8: Create Test

5.9 New Test Run

AST [Projects](#) [Test Suites](#) [Test Cases](#) [Test Runs](#) Asad - Logout

Test Case Executions

Execution ID	Project	Start time	End time	Time Spent	Status	
30	PakWheels	01 December, 2019 05:53:04 PM	01 December, 2019 05:56:17 PM	3 minutes	Completed	Show
31	PakWheels	01 December, 2019 06:00:19 PM	01 December, 2019 06:01:50 PM	2 minutes	Completed	Show
32	PakWheels	01 December, 2019 07:08:14 PM	01 December, 2019 07:08:37 PM	less than a minute	Completed	Show
33	Zameen	03 December, 2019 12:34:29 PM	03 December, 2019 12:34:33 PM	less than a minute	Completed	Show
34	Zameen	03 December, 2019 12:34:47 PM		-	In-Progress	Show
35	PakWheels	03 December, 2019 12:34:58 PM	03 December, 2019 12:35:26 PM	less than a minute	Completed	Show
36	Zameen	05 December, 2019 02:23:36 PM	05 December, 2019 02:23:41 PM	less than a minute	Completed	Show
37	Zameen	05 December, 2019 06:06:34 PM	05 December, 2019 06:06:38 PM	less than a minute	Completed	Show
38	Rozee	05 December, 2019 06:06:50 PM	05 December, 2019 06:06:50 PM	less than a minute	Completed	Show
39	PakWheels	05 December, 2019 06:06:58 PM		-	In-Progress	Show
40	PakWheels	05 December, 2019 06:07:17 PM	05 December, 2019 06:07:40 PM	less than a minute	Completed	Show
41	Zameen	05 December, 2019 06:11:13 PM		-	In-Progress	Show
42	Zameen	05 December, 2019 07:24:05 PM	05 December, 2019 07:24:18 PM	less than a minute	Completed	Show
43	PakWheels	06 December, 2019 05:05:43 PM	06 December, 2019 05:06:27 PM	1 minute	Completed	Show

Figure 5-9: Test Run

5.10 Passed or Failed Test Cases

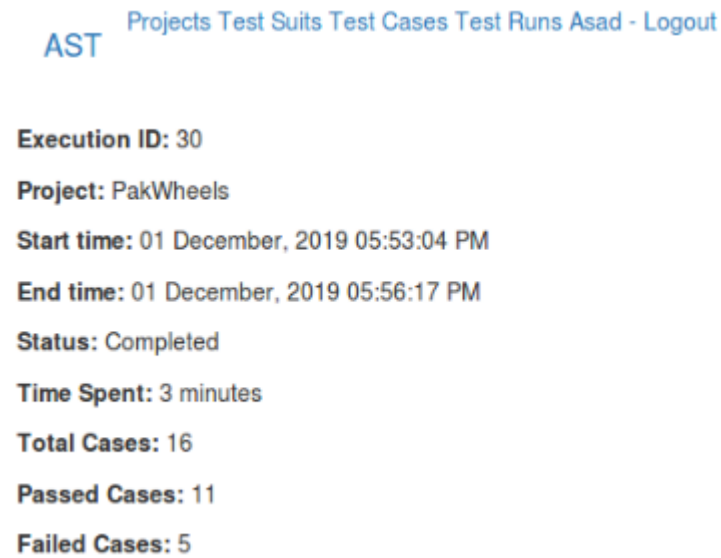


Figure 5-10: Test Cases Execution results

5.11 Graphical Report of Test Cases

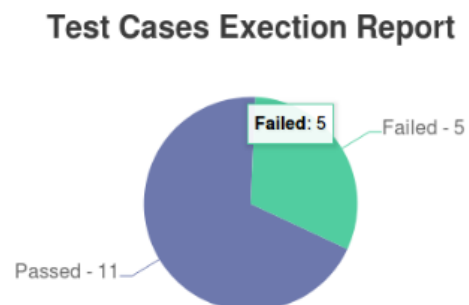
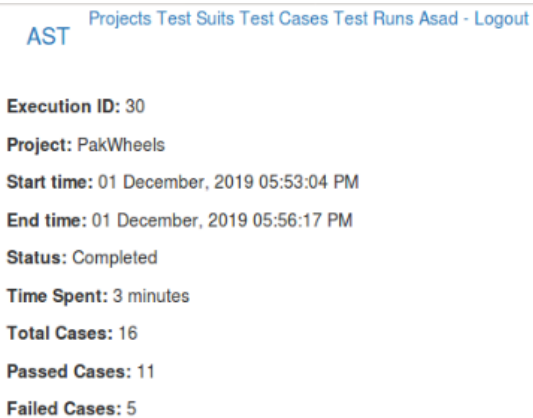


Figure 5-11: Graph Execution Report

CHAPTER 6

CONCLUSION AND RECOMMENDATIONS

6.1 Recommendation

Automated software testing is basically recommended for QA's. Testers can build test suites of tests that covers every feature in software application. Every software tester's group tests its products, yet delivered software always has defects. Test engineers strive to catch them before the product is released but they always become puzzled in and they often reappear, even with the best manual testing processes. Test Automation software is the best way to increase the effectiveness, efficiency and coverage of your software testing and also time efficient. Automation testing basically runs tests significantly faster than human users.

6.2 Conclusion

Automated software testing is a necessity if QA's are to meet the fast-paced development deadlines of today's software markets. More and more development organizations are following Microsoft Corporation's lead and implementing the build-a-day approach. In the past, build dates were set and builds occurred at specific intervals that were measured in days, weeks, or months. That cost the industries lots of time and also coast effective. But the automated software testing will provide a platform that will help them to resolve their issues and generate reports automatically.

REFERENCES

Journal Papers:

- [1] Grano G, Titov TV, Panichella S, Gall HC. How high will it be? Using machine learning models to predict branch coverage in automated testing. In: Fontana FA, Walter B, Ampatzoglou A, Palomba F, eds. 2018 IEEE Workshop On Machine Learning Techniques for Software Quality Evaluation (MaLTeSQuE). Campobasso, Italy: IEEE Computer Society; 2018:19-24.
- [2] Grano G, Titov TV, Panichella S, Gall HC. Replication Package - Branch Coverage Prediction in Automated Testing. <https://doi.org/10.5281/zenodo.2548323>; doi 10.5281/zenodo.254832; 2019.

Conference Papers:

- [3] Claus, K and R. Rudolf. (2017). A Journey from Manual Testing to Automated Test Generation in an Industry Project. 2017 IEEE International Conference on Software Quality, Reliability and Security Companion (QRS-C) 591--592.