Contents

GOVERNMENT COLLEGE UNIVERSITY FAISALABAD ..... **Error! Bookmark not defined.**
Chapter no. 1 Introduction 5
1.1 Purpose of Project: 5
1.2 How it works? 5
1.3 Project Scope: 5
Chapter No.2 BACKGROUND & PROBLEM DEFINITION 6
2.1 Background Research: 6
2.2 Existing technology: 6
2.3 Area of study: 6
2.4 Reason of the Project: 6
2.5 Objectives of project: 6
2.6 Methodology: 6
Chapter No. 3 SYSTEM REQUIREMENT ANAYLSIS 8
3.1 System Functional Requirements: 8
3.1.1 Use case: 8
3.2 System Non-Functional Requirements: 14
3.3 Hardware & Software Requirements 15
3.3.1 Hardware Requirements: 15
3.3.2 Software Requirements: 15
Chapter No. 4 System Design 16
4.1 Use case-fully dressed: 16
4.2 WBS-fully dressed: 17
4.3 System sequence diagrams: 18
4.4 Class Diagram: 19
4.5 ER Model: 20
4.6 Data Flow Diagram: 21
Chapter No. 5 IMPLEMEMNTATION & TESTING 25
5.1 Testing: 25
5.2 Importance of Software testing: 25
5.3 Objectives of System testing: 26

5.4 Types of Software testing: 27
5.5Test Cases: 34
5.6 Implementation: 39
Chapter No. 6 User Manual 41
6.1 Login page: 42
6.2 Registration Page: 43
6.3 Profile: 44
6.4 Newsfeed & Post: 45
6.5 Messages & Chat: 46
6.6 Friend Request: 47
6.7 Account settings: 48
6.8 Notification Panel: 49
6.9 Search bar: 50

List of Tables:

Table 1:UC_01 9
Table 2: UC_02 10
Table 3:UC_03 11
Table 4:UC_04 12
Table 5:UC_05 13
Table 6: Test case 1 34
Table 7: Test case 2 35
Table 8: Test case 3 35
Table 9: Test case 4 36
Table 10: Test case 5 36
Table 11: Teat case 6 37

Tables of Figures

Figure 1:Use case Diagram 16
Figure 2: WBS 17
Figure 3: System sequence Diagram 18
Figure 4: Class Diagram 19
Figure 5: Entity relation Diagram 20
Figure 6: Data flow Diagram 1 21
Figure 7: Data flow Diagram 2 22
Figure 8: Data flow Diagram 3 23
Figure 9: Data flow Diagram 4 24
Figure 10: Login page 42
Figure 11: Registration page 43
Figure 12: Profile 44
Figure 13: Newsfeed & Post 45
Figure 14: Messages & Chat 46
Figure 15: Friend Request 47
Figure 16: Account setting 48
Figure 17: Notification panel 49
Figure 18: Search bar 50

Chapter no. 1 Introduction

1.1 Purpose of Project:

Purpose of this project is that the social network to helps developer which can connect each other and discuss given projects issues and errors. That can solve with the help of developers.

1.2 How it works?

Since this is a php based project so it works on localhost server such as xampp etc. But after deployment on any local server it could run globally.

1.3 Project Scope:

This Social network for developers can solve developer's problems such as to solve errors, to discuss project relatives' activities. Developers can connect each other, view their profiles and view their projects available on GitHub. Clients can also connect with developers to take projects to developers.

Chapter No.2 BACKGROUND & PROBLEM DEFINITION

2.1 Background Research:

There are not much platforms are available for developer to interact with others. There should a platform for them where they can interact and solve their problems and could help each other.

2.2 Existing technology:

Some technologies are available for this purpose but that are not enough. There should be technology that gives full freedom to developers.

2.3 Area of study:

This project is Web-based that can be accessed by everyone from any device like mobile computer etc.

2.4 Reason of the Project:

Developers faces problems during their project development as like rum time error and different other types of errors and these errors also facing in testing. So main reason is that they can share their problems in Social developer's community for solving their issues.

2.5 Objectives of project:

Developers learn something new and Share their experienceswith each other. Clients connect also with developers and give projects to developer according to their experience, expertise, skillset and technologies they use. Developers also discuss problems of projects to other.

2.6 Methodology:

- HTML
- CSS
- BOOTSTRAP

- JAVASCRIPT
- PHP
- MYSQL
- PWA.

Chapter No. 3 SYSTEM REQUIREMENT ANAYLSIS

3.1 System Functional Requirements:

Functional requirements may involve calculation, technical details, data manipulation and processing, and other functionality that define what a system is supposed to accomplish.

- Internet Connection
- Hardware Software
- User interface
- Privacy
- Create Account
- View Account
- Delete Account
- Update Account
- Login Account
- Logout Account
- Newsfeed
- Home
- Uploading Content.

3.1.1 Use case:

Use case diagrams are usually reffered to as behavior diagrams use to describe a set of actions (use cases) that some system or system should can perform in collaboration with one or more external users of system.

1: Log in

Use case ID: UC_01

Use case Title: Login

Actor user

Pre condition: User should click on login page to open login activity.

Post condition: Login screen will be display and user have to enter his correct e-mail address and password to login.

Summary: On entering correct e-mail address and password user will be authenticated to log on to homepage.

Alternating path: If the e-mail address and password are not correct or user does not have an account the registration button is available.

User action	**System reaction**
The user start with clicking on login page.	System show the system login screen.
User enter the e-mail address and password.	System checked either e-mail address matching or not
clicked on login button	System authenticate the password and check wether it is true or not.

Table 1:UC_01

2: Registration

Use case ID: UC_02

Use case Title: Registration

Actor user

Pre condition: User should click on Registration button to open registration activity.

Post condition: Registration screen will be display and user have to enter his credentials to register an account.

Summary: On entering correct credentials user will be allowed to log on to homepage.

Alternating path: User can leave the activity by clicking exit button

User action	**System reaction**
The user start with clicking on registration page..	System show the system registration screen.
User enter the credentials.	System checked all credentials.
clicked on submit button	After checking all credentials required system will register the account and show an green signal.

Table 2: UC_02

3: Friend request

Use case ID: UC_03

Use case Title: Friend request

Actor user

Pre condition: User should search an account.

Post condition: User will be the friend and can send message.

Summary: On sending friend request to other user they can be friends mutually.

Alternating path: Other User can ignore the friend request.

User action	**System reaction**
The user search an account using search bar.	System show the other user whose name is entered in search bar.
User click on the add friend button to send the friend request	System send the friend request to other user.
Other user clicked on accept button to accept the friend request.	System show an message that you can now chat mutually.

Table 3:UC_03

4: Update password

Use case ID: UC_04

Use case Title: Update password

Actor user

Pre condition: User must have an old password.

Post condition: Password will be changed.

Summary: User can the update his password.

Alternating path: User can exit during the setting and continue with old password.

User action	**System reaction**
User open the setting and clicked on update password.	System show the update password screen and asked user to enter old password.
User entered the old password.	System checked old password true or not and asked entering new password.
User entered new password and click on submit button.	System changed the password.

Table 4:UC_04

5: Delete account

Use case ID: UC_05

Use case Title: Delete account

Actor user

Pre condition: User must have account.

Post condition: Account will be deleted.

Summary: User can delete his account and make new account.

Alternating path: user could exit this process.

User action	**System reaction**
User open the setting and click on delete account button.	System show the delete account.
User click on delete account button.	System asked to confirm to delete account or not.
User clicked on yes button .	System deleted account and show the registration /login screen.

Table 5:UC_05

3.2 System Non-Functional Requirements:

- Usability
- Reliability
- Performance
- Speed
- Design
- Constraints
- Portability
- Maintainability

3.3 Hardware & Software Requirements

3.3.1 Hardware Requirements:
Core i4 processor or above

Atleast 4GB DDR3 RAM

At least 150GB SOLID STATE DRIVE (SSD)

3.3.2 Software Requirements:
XAMPP or any local server sever s/w

Chapter No. 4 System Design

4.1 Use case-fully dressed:

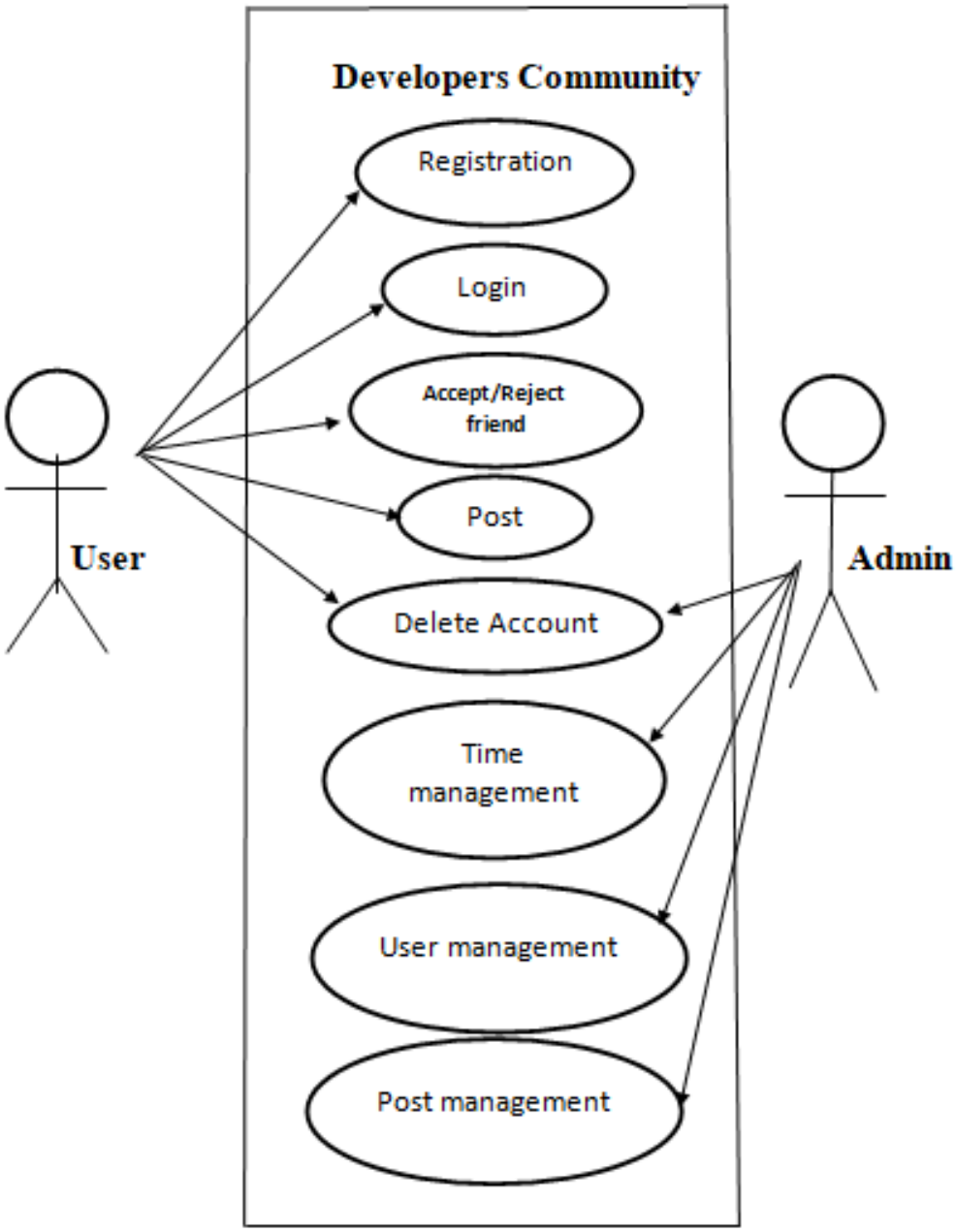

Figure 1:Use case Diagram

4.2 WBS-fully dressed:

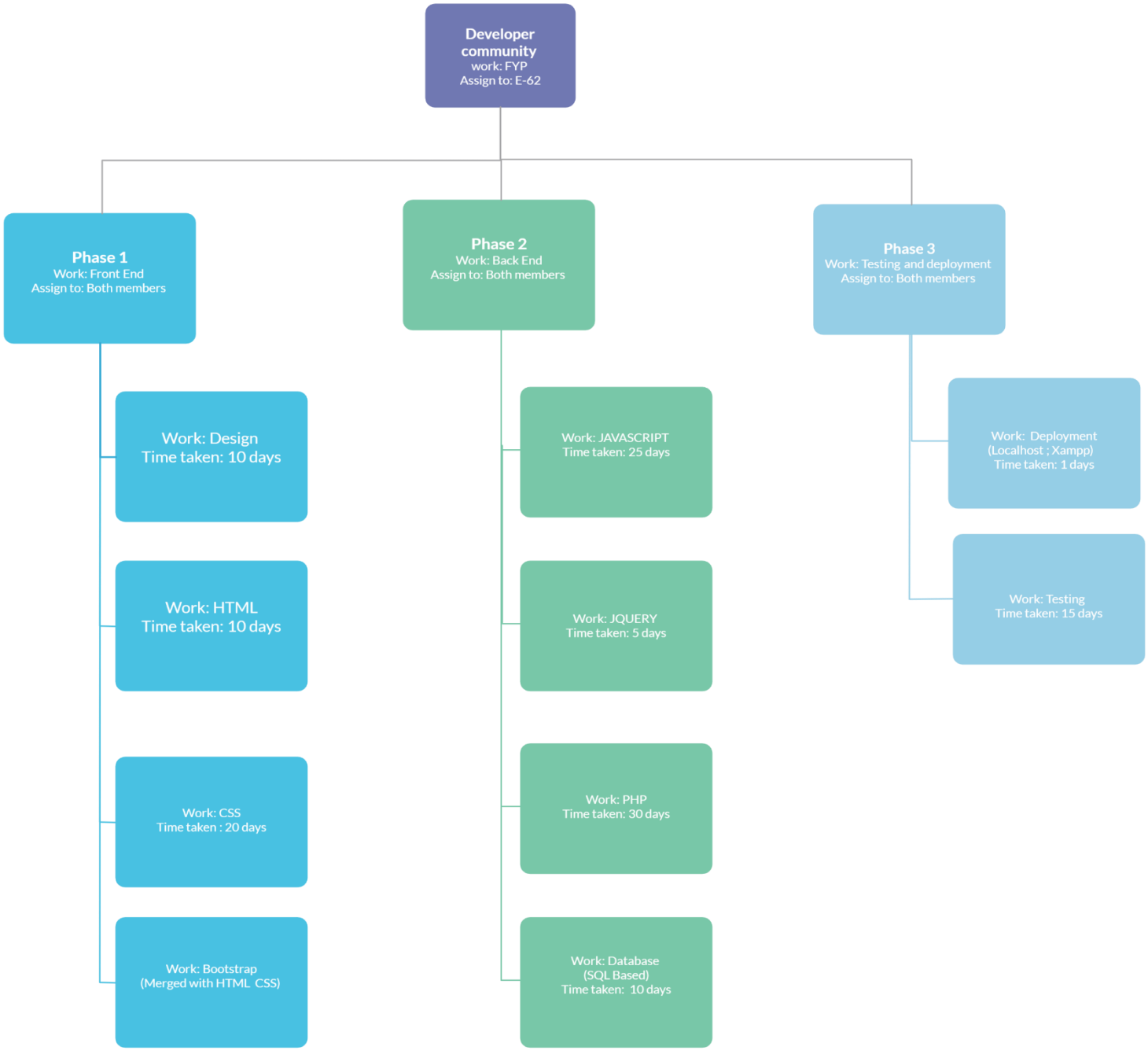

Figure 2: WBS

4.3 System sequence diagrams:

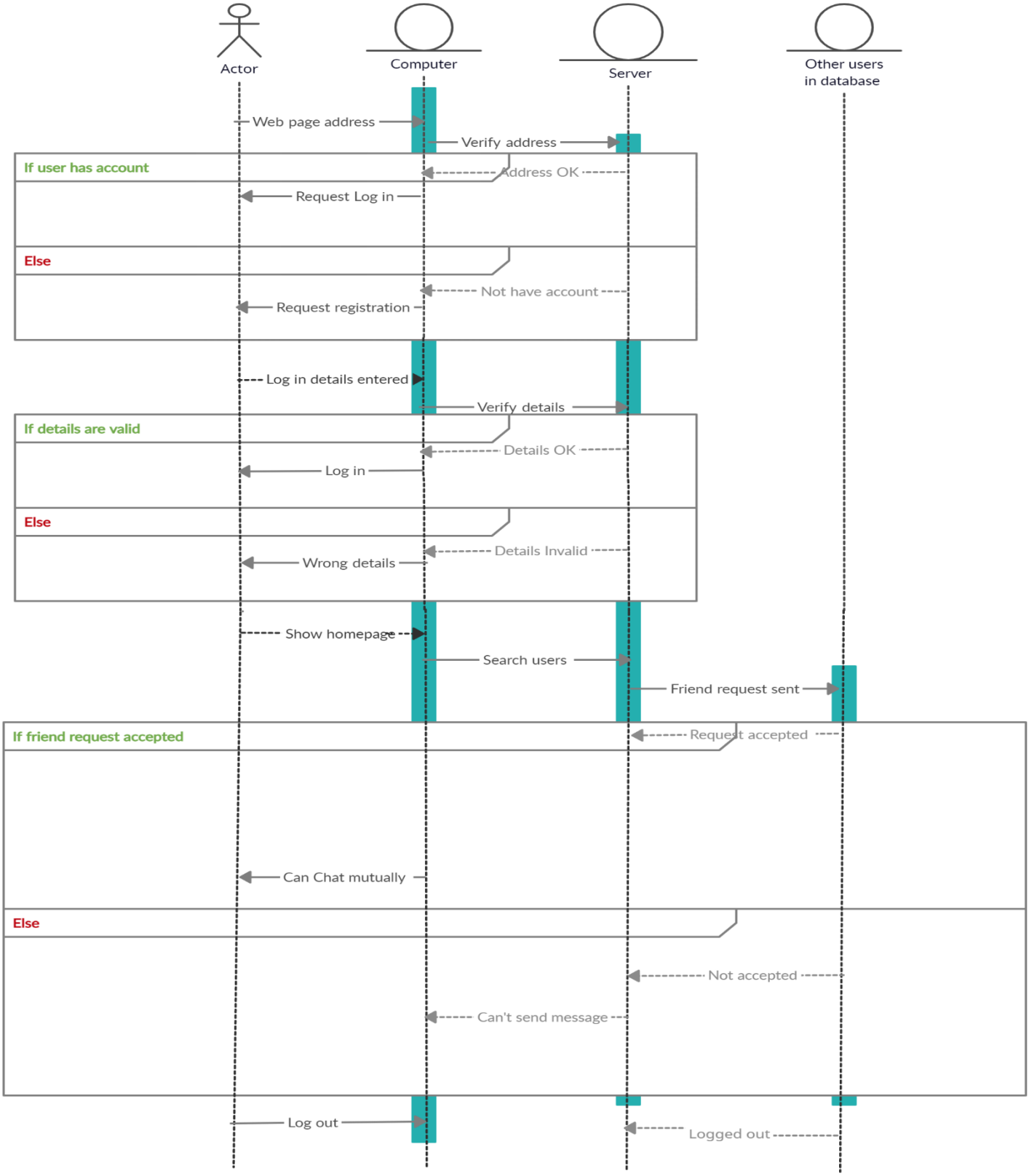

Figure 3: System sequence Diagram

4.4 Class Diagram:

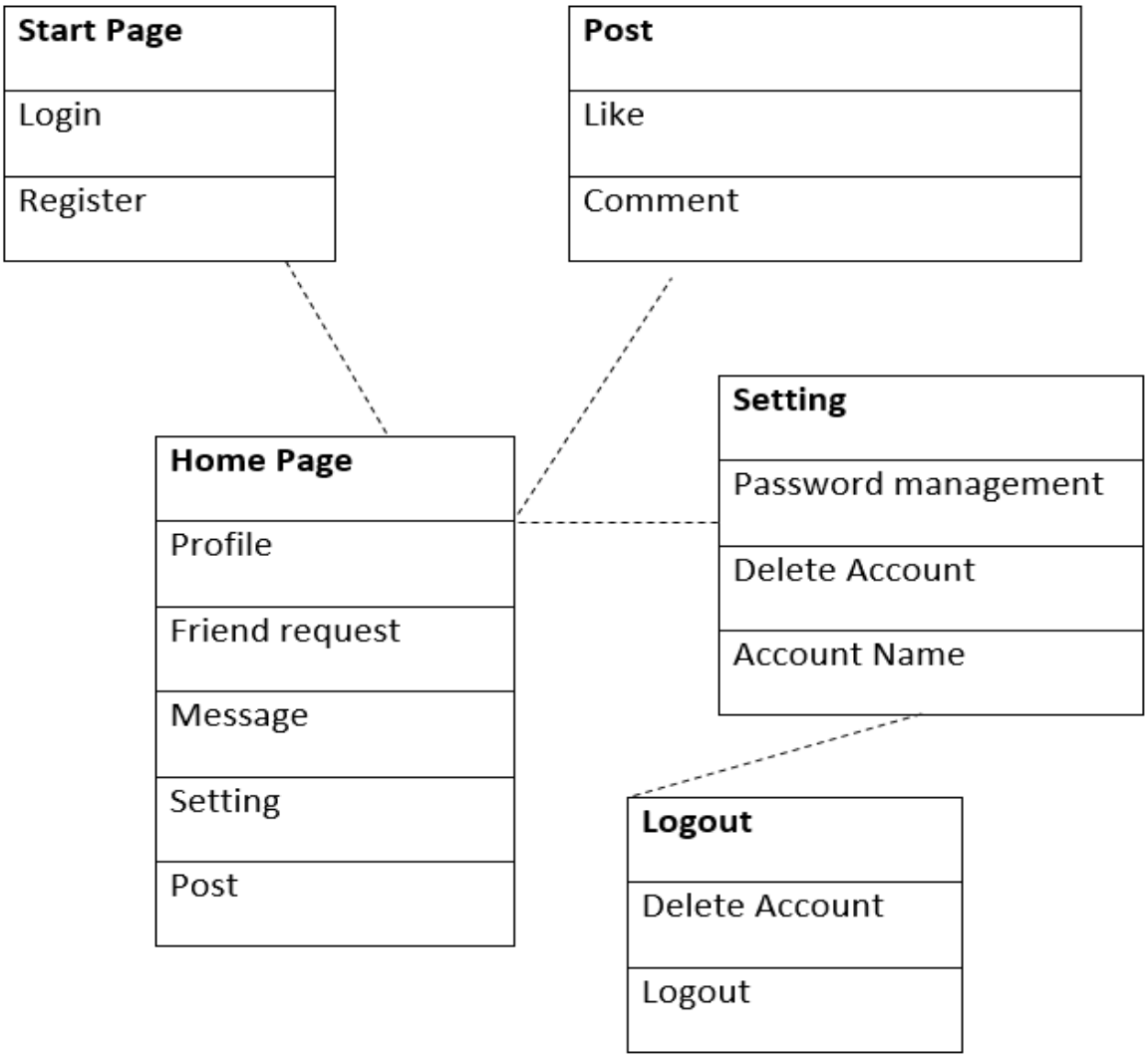

Figure 4: Class Diagram

4.5 ER Model:

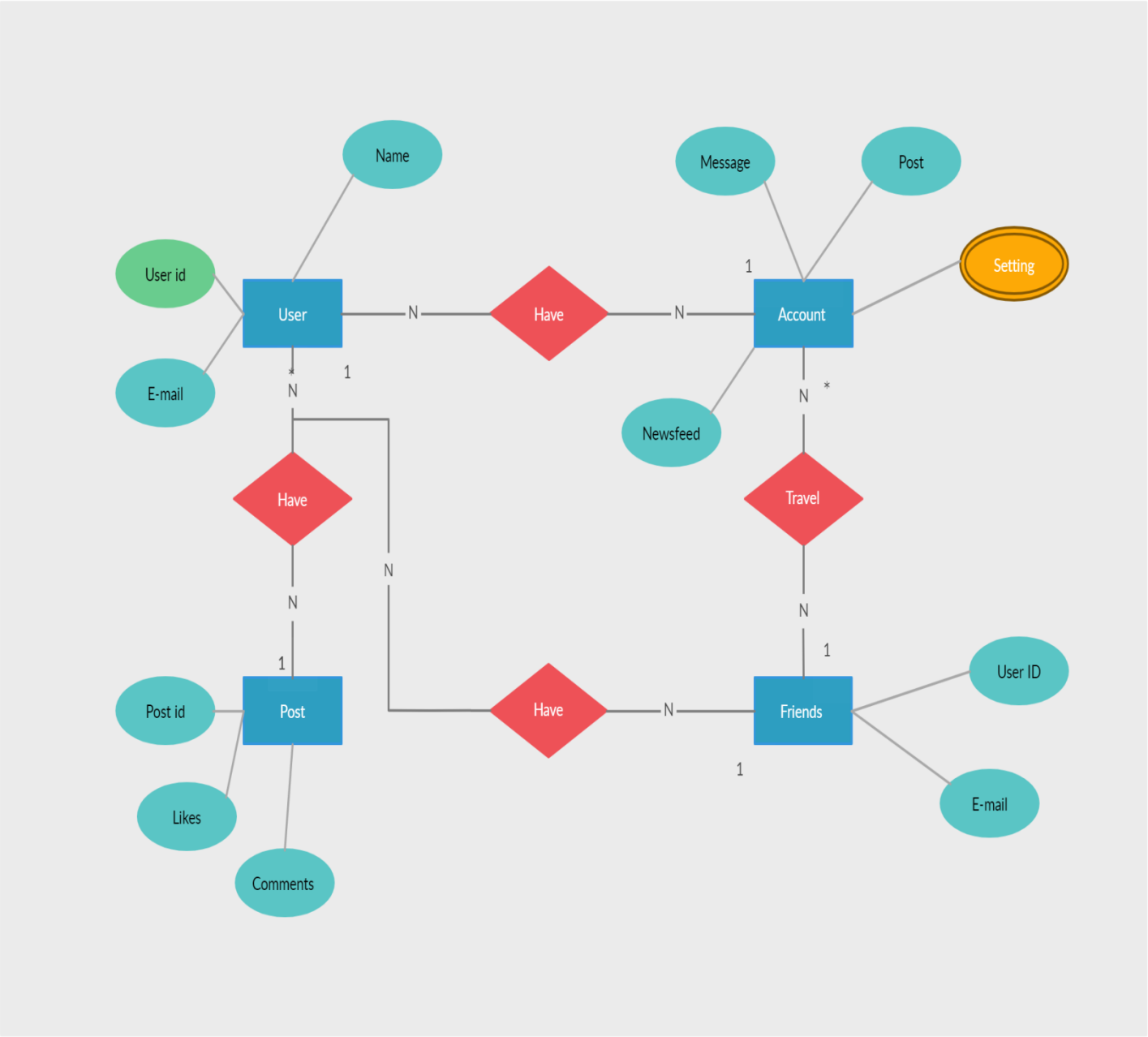

Figure 5: Entity relation Diagram

4.6 Data Flow Diagram:

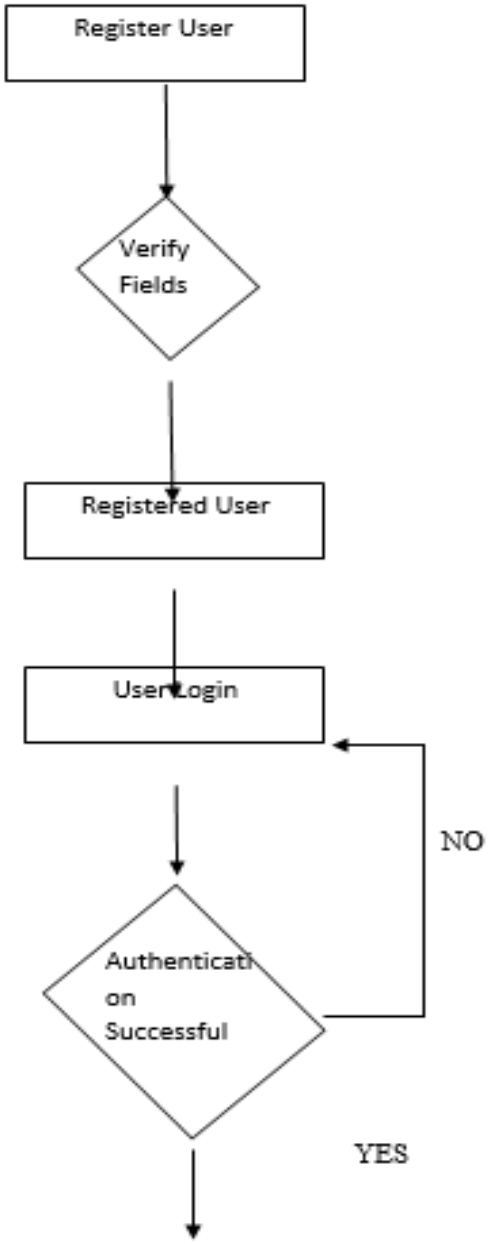

Figure 6: Data flow Diagram 1

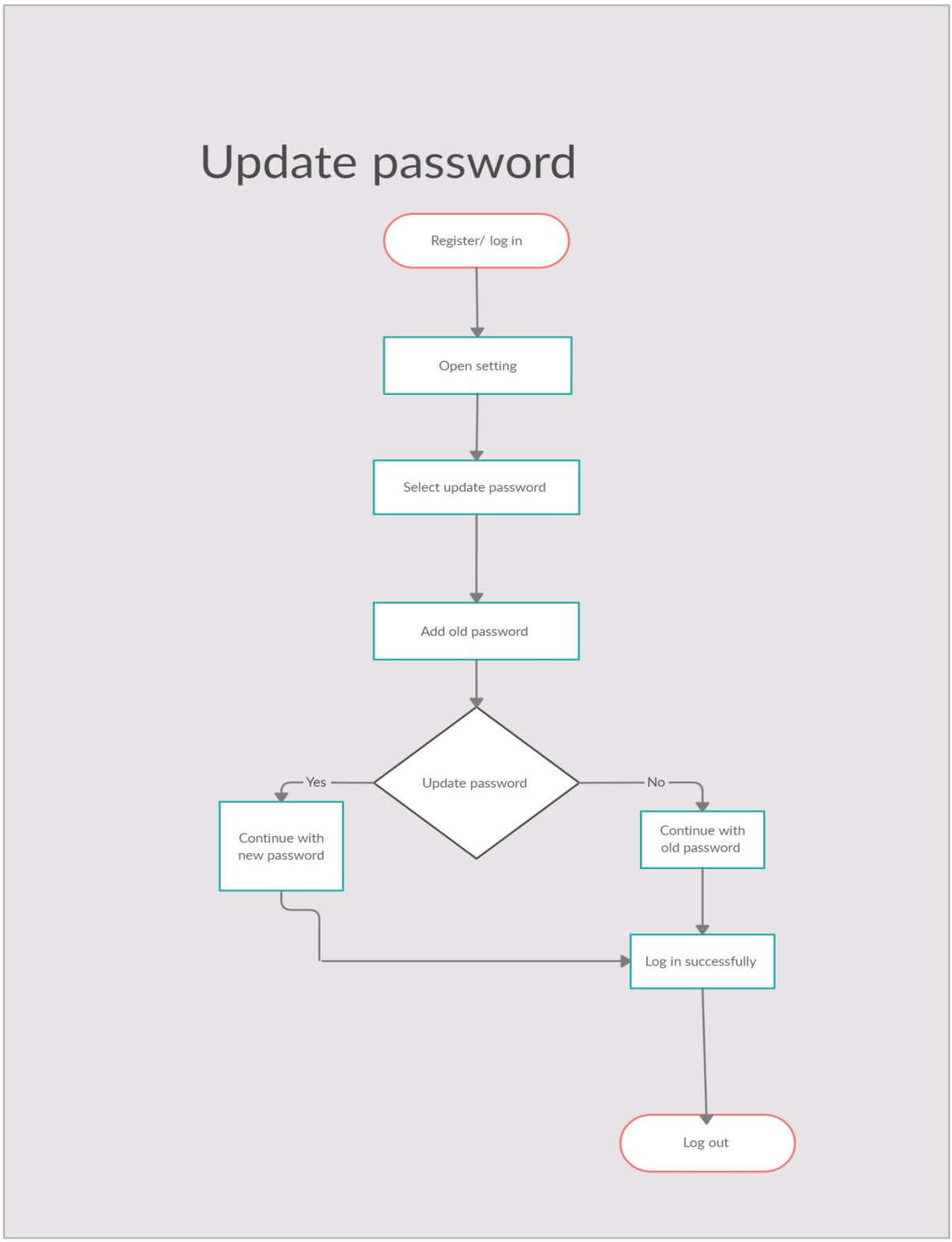

Figure 7: Data flow Diagram 2

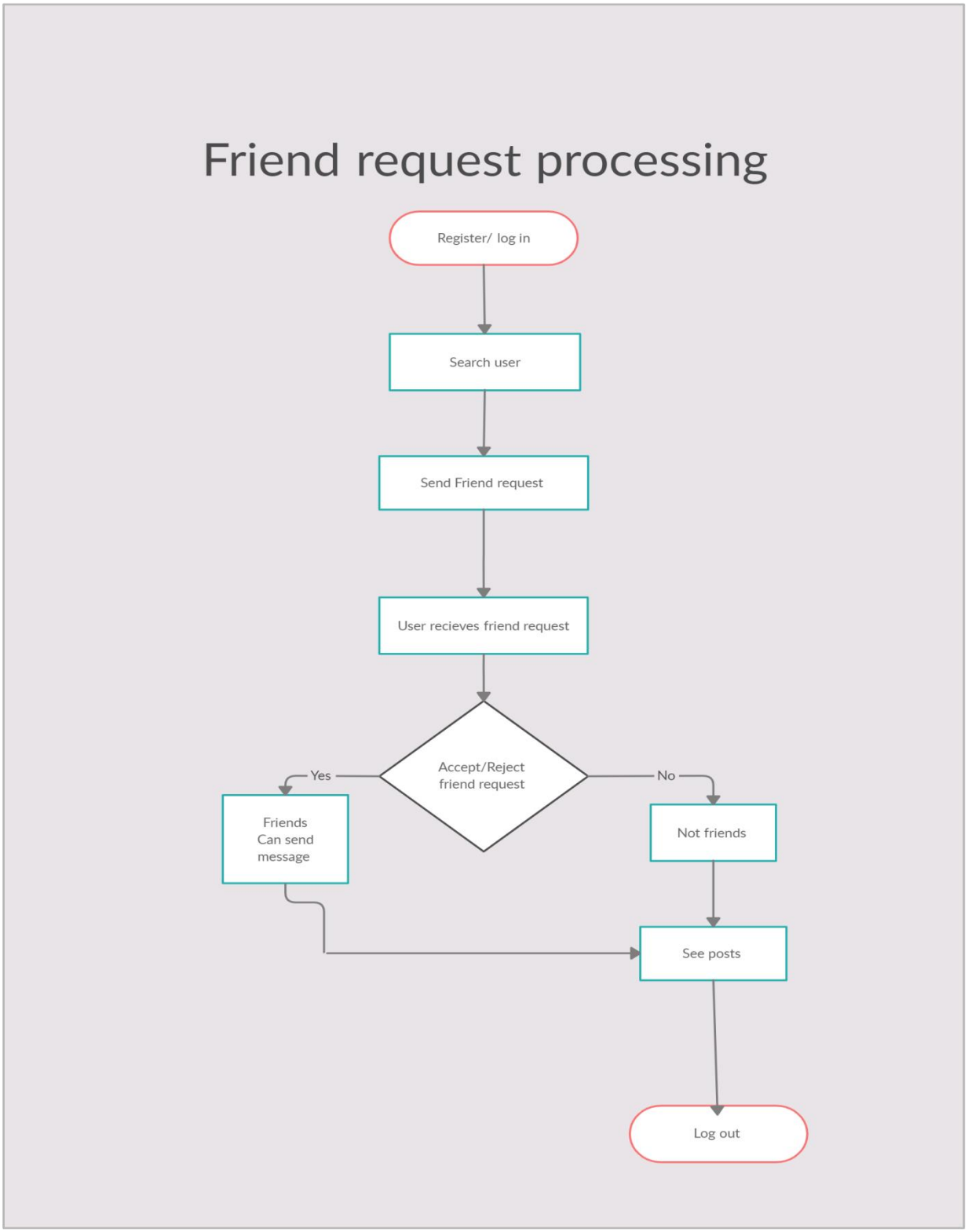

Figure 8: Data flow Diagram 3

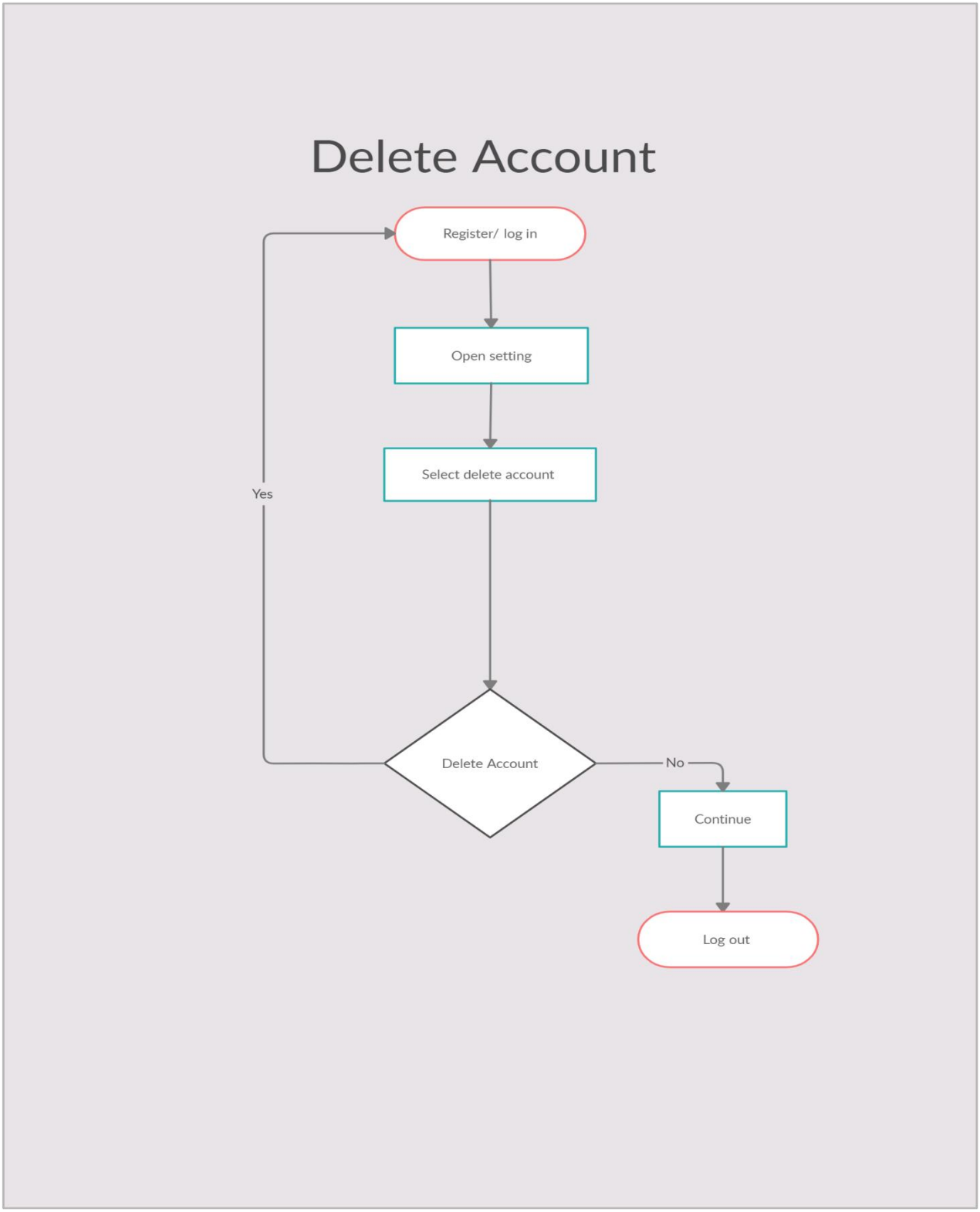

Figure 9: Data flow Diagram 4

Chapter No. 5 IMPLEMEMNTATION & TESTING

5.1 Testing:

Software testing is defined as an activity to check whether the actual results match the expected results and to ensure that the software system is Defect free. It involves execution of a software component or system component to evaluate one or more properties of interest.

Software testing also helps to identify errors, gaps or missing requirements in contrary to the actual requirements. It can be either done manually or using automated tools. Some prefer saying Software testing as a White Box and Black Box Testing.

In simple terms, Software Testing means Verification of Application Under Test (AUT).

5.2 Importance of Software testing:

Testing is important because software bugs could be expensive or even dangerous. Software bugs can potentially cause monetary and human loss, and history is full of such examples.

- In April 2015, Bloomberg terminal in London crashed due to software glitch affected more than 300,000 traders on financial markets. It forced the government to postpone a 3bn pound debt sale.
- Nissan cars have to recall over 1 million cars from the market due to software failure in the airbag sensory detectors. There has been reported two accident due to this software failure.
- Starbucks was forced to close about 60 percent of stores in the US and Canada due to software failure in its POS system. At one-point store served coffee for free as they unable to process the transaction.
- Some of the Amazon's third-party retailers saw their product price is reduced to Ip due to a software glitch. They were left with heavy losses.
- Vulnerability in Window 10. This bug enables users to escape from security sandboxes through a flaw in the win32k system.
- In 2015 fighter plane F-35 fell victim to a software bug, making it unable to detect targets correctly. China Airlines Airbus A300 crashed due to a software bug on April 26, 1994, killing 264 innocents live.
- In 1985, Canada's Therac-25 radiation therapy machine malfunctioned due to software bug and delivered lethal radiation doses to patients, leaving 3 people dead and critically injuring 3 others.
- In April of 1999, a software bug caused the failure of a $1.2 billion military satellite launch, the costliest accident in history
- In may of 1996, a software bug caused the bank accounts of 823 customers of a major U.S.bank to be credited with 920 million US dollars.

5.3 Objectives of System testing:

The goals and objectives of software testing are numerous, which when achieved help developers build a defect less and error free software and application that has exceptional performance, quality, effectiveness, security, among other things. Though the objective of testing can vary from company to company and project to project, there are some goals that are similar for all. These objectives are:

Verification: A prominent objective of testing is verification, which allows testers to confirm that the software meets the various business and technical requirements stated by the client before the inception of the whole project. These requirements and specifications guide the design and development of the software, hence are required to be followed rigorously. Moreover, compliance with these requirements and specifications is important for the success of the project as well as to satisfy the client.

Validation: Confirms that the software performs as expected and as per the requirements of the clients. Validation involves checking the comparing the final output with the expected output and then making necessary changes if there is a difference between the two.

Defects: The most important purpose of testing is to find different defects in the software to prevent its failure or crash during implementation or go live of the project. Defects if left undetected or unattended can harm the functioning of the software and can lead to loss of resources, money, and reputation of the client. Therefore, software testing is executed regularly during each stage of software development to find defects of various kinds. The ultimate source of these defects can be traced back to a fault introduced during the specification, design, development, or programming phases of the software.

Providing Information: With the assistance of reports generated during the process of software testing, testers can accumulate a variety of information elated to the software and the steps taken to prevent its failure. These, then can be shared with all the stakeholders of the project for better understanding of the project as well as to establish transparency between members.

Preventing: During the process of testing the aim of testes to identify defects and prevent them from occurring aging in the future. To accomplish this goal, software is tested rigorously by a independent testers, who are not responsible for software development.

Quality Analysis: Testing helps improve the quality of the software by constantly measuring and verifying its design and coding. Additionally, various types of testing techniques are used by testers, which help them achieve the desired software quality.

Compatibility: It helps validate application's compatibility with the implementation environment, various devices, Operating Systems, user requirements, among other things.

For optimum User Experience: Easy software and application accessibility and optimum user experience are two important requirements that need to be accomplished for the success of any project as well as to increase the revenue of the client. Therefore, to ensure this software is tested again and again by the testers with the assistance of stress testing, load testing, spike testing, etc.

Verifying Performance & Functionality: It ensures superior performance and functionality. This is mainly verified by placing the software under extreme stress to identify and measure. Its

all plausible failure modes. To ensure this, performance testing, usability testing, functionality testing, etc. is executed by the testers.

5.4 Types of Software testing:

Functional testing types:

- Unit testing
- Integration testing
- System testing
- Sanity testing
- Smoke testing
- Interface testing
- Regression testing
- Beta/Acceptance testing

Non-Functional testing types:

- Performance Testing
- Load testing
- Stress testing
- Volume testing
- Security testing
- Compatibility testing
- Install testing
- Recovery testing
- Reliability testing
- Usability testing
- Compliance testing
- Localization testing

Alpha Testing: It is the most common type of testing used in the Software industry. The objective of this testing is to identify all possible issues or defects before releasing it into the market or to the user.

Alpha testing is carried out at the end of the software development phase but before the Beta Testing. Still, minor design changes may be made as a result of such testing. Alpha testing is conducted at the developer's site. In-house virtual user environment can be created for this type of testing.

Acceptance Testing: An acceptance test is performed by the client and verifies whether the end to end the flow of the system is as per the business requirements r not and if it is as per the needs of the end user. Client accepts the software only when all the features and functionalities work as expected.

It is the last phase of the testing, after which the software goes into production. This is also called User Acceptance Testing (UAT).

Ad-hoc Testing: The name itself suggests that this testing is performed on an ad-hoc basis i.e. with no reference to the test case and also without any plan or documentation in place for such type of testing. The objective of this testing is to find the defects and break the application by executing any flow of the application or any random functionality.

Ad-hoc testing is an informal way of finding defects and can be performed by anyone in the project It is difficult to identify defects without a test case but sometimes it is possible that defects found during ad-hoc testing might not have been identified using existing test cases

Accessibility Testing: The aim of accessibility testing is to determine whether the software or application is accessible for disabled people or not. Here disability means deaf, color blind, mentally disabled, blind, old age and other disabled groups. Various checks are performed such as font size for visually disabled, color and contrast for color blindness etc.

Beta Testing: Beta Testing is a formal type of software testing which is carried out by the customer. It is performed in the Real Environment before releasing the product to the market for the actual end users.

Beta testing is carried out to ensure that there are no major failures in the software or product and it satisfies the business requirements from a end-user perspective. Beta testing is successful when the customer accepts the software.

Usually, this testing is typically done by end-users or others. It is the final testing done before releasing an application for commercial purpose. Usually, the Beta version of the software or product released is limited to a certain number of users in a specific area.

So end user actually uses the software and shares the feedback to the company. Company then takes necessary action before releasing the software to the worldwide.

Back-end Testing: Whenever an input or data is entered on front-end application, it stores in the database and the testing of such database is known as Database Testing or Backend testing.

Thereare different databases like SQL Server, MySQL, and Oracle etc. Database testing involves testing of table structure, schema, stored procedure, data structure and so on.

In back-end testing GUI is not involved, testers are directly connected to the database with proper access and testers can easily verify data by running a few queries on the database. There can be issues identified like data loss, deadlock, data corruption ete during this back-end testing and these issues are critical to fixing before the system goes live into the production environment

Browser Compatibility Testing: It is a subtype of Compatibility Testing (which is explained below) and is performed by the testing team. Browser Compatibility Testing is performed for web applications and it ensures that the software can run with the combination of different browser and operating system. This type of testing also validates whether web application runs on all versions of all browsers or not.

White box testing: This is software testing technique in which internal structure, design and coding of software are tested to verify flow of input-output and to improve design, usability and security. In white box testing, code is visible to testers so it is also called Clear box testing, open box testing. Transparent box testing, Code-based testing and Glass box testing.

White box testing involves the testing of the software code for the following:

- Internal security holes
- Broken or poorly structured paths in the coding processes
- The flow of specific inputs through the code
- Expected output
- The functionality of conditional loops
- Testing of each statement, object, and function on an individual basis

Black box testing:

Black box testing also known as Behavioral Testing, is a software testing method in which the Internal structure/ design/ implementation of the item being tested are not known to the tester. These tests can be functional or non-functional, though usually functional. This method is named So because the software program, in the eyes of the tester, is like a black box; inside which one cannot see.
This method attempts to find errors in the following categories

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behavior or performance errors
- Initialization and termination errors

Backward Compatibility Testing: It is a type of testing which validates whether the newly developed software or updated software works well with older version of the environment or not Backward Compatibility Testing checks whether the new version of the software works properly with file format created by older version of the software; it also works well with data tables, data files, data structure created by older version of that software. If any of the software is updated then it should work well on top of the previous version of that software.

Boundary Value Testing: This type of testing checks the behavior of the application at the boundary level. Boundary is performed for checking if defects exist at boundary values. Boundary value testing is used for testing a different range of numbers. There is an upper and lower boundary for each range and testing is performed on these boundary values. If testing requires a test range of numbers from 1 to 500 then Boundary Value Testing is performed on values at 0, 1, 2, 499, 500 and 501.

Branch Testing: It is a type of white box testing and is carried out during unit testing. Branch Testing, the name itself suggests that the code is tested thoroughly by traversing at every branch.

Comparison Testing: Comparison of a product's strength and weaknesses with its previous versions or other similar products is termed as Comparison Testing.

Compatibility Testing: It is a testing type in which it validates how software behaves and runs in a different environment, web servers, hardware, and network environment. Compatibility testing ensures that software can run on a different configuration, different database, different browsers, and their versions. Compatibility testing is performed by the testing team.

Component Testing: It is mostly performed by developers after the completion of unit testing Component Testing involves testing of multiple functionalities as a single code and its objective is to identify if any defect exists after connecting those multiple functionalities with each other.

End-to-End Testing: Similar to system testing, End-to-end testing involves testing of a complete application environment in a situation that mimics real-world use, such as interacting with a database, using network communications, or interacting with other hardware, applications, or systems if appropriate.

Equivalence Partitioning: It is a testing technique and a type of Black Box Testing. During this equivalence partitioning, a set of group is selected and a few values or numbers are picked up for testing. It is understood that all values from that group generate the same output.

The aim of this testing is to remove redundant test cases within a specific group which generates the same output but not any defect. Suppose, application accepts values between -10 to +10 so using equivalence partitioning the values picked up for testing are zero, one positive value, one negative value. So the Equivalence Partitioning for this testing is: -10 to -1, 0, and 1 to 10.

Example Testing: It means real-time testing. Example testing includes the real-time scenario, it also involves the scenarios based on the experience of the testers

Exploratory Testing: Exploratory Testing is informal testing performed by the testing team. The objective of this testing is to explore the application and looking for defects that exist in the

application. Sometimes it may happen that during this testing major defect discovered can even cause system failure. During exploratory testing, it is advisable to keep a track of what flow you have tested and what activity you did before the start of the specific flow. An is performed without documentation and test cases.

Functional Testing: This type of testing ignores the internal parts and focuses only on the output to check if it is as per the requirement or not. It is a Black-box type testing geared to the functional requirements of an application.

Graphical User Interface (GUI) Testing: The objective of this GUI testing is to validate the GUI as per the business requirement. The expected GUI of the application is mentioned in the Detailed Design Document and GUI mockup screens. The GUI testing includes the size of the buttons and input field present on the screen, alignment of all text, tables and content in the tables. It also validates the menu of the application, after selecting different menu and menu items, it validates that the page does not fluctuate and the alignment remains same after hovering the mouse on the menu or sub-menu

Gorilla Testing: Gorilla Testing is a testing type performed by a tester and sometimes by developer the as well. In Gorilla Testing, one module or the functionality in the module is tested thoroughly and heavily. The objective of this testing is to check the robustness of the application.

Happy Path Testing: The objective of Happy Path Testing is to test an application successfully on a positive flow. It does not look for negative or error conditions. The focus is only on the valid and positive inputs through which application generates the expected output.

Incremental Integration Testing: Incremental Integration Testing is a Bottom-up approach for testing i.e continuous testing of an application when a new functionality is added. Application functionality and modules should be independent enough to test separately. This is done by programmers or by testers.

Install/Uninstall Testing: Installation and uninstallation testing is done on full, partial, or upgrade install / uninstall processes on different operating systems under different hardware or software environment.

Integration Testing: Testing of all integrated modules to verify the combined functionality after integration is termed as Integration Testing. Modules are typically code modules, individual applications, client and server applications on a network, etc. This type of testing is especially relevant to client/server and distributed systems.

Load Testing: It is a type of non-functional testing and the objective of Load testing is to check how much of load or maximum workload a system can handle without any performance degradation.

Load testing helps to find the maximum capacity of the system under specific load and any issues that cause the software performance degradation. Load testing is performed using tools like JMeter, LoadRunner, Web Load, Silk performer etc.

Monkey Testing: Monkey testing is carried out by a tester assuming that if the monkey uses the application then how random input, values will be entered by the Monkey without any knowledge or understanding of the application.

The objective of Monkey Testing is to check if an application or system gets crashed by providing random input values/data. Monkey Testing is performed randomly and no test cases are scripted and it is not necessary to Monkey Testing is performed randomly and no test cases are scripted and it is not necessary to be aware of the full functionality of the system.

Mutation Testing: Mutation Testing is a type of white box testing in which the source code of one of the program is changed and verifies whether the existing test cases can identify these defects in the system. The change in the program source code is very minimal so that it does not impact the entire application, only the specific area having the impact and the related test cases should able to identify those errors in the system.

Negative Testing: Testers having the mindset of "attitude to break" and using negative testing they validate that if system or application breaks. A negative testing technique is performed using incorrect data, invalid data or input. It validates that if the system throws an error of invalid input and behaves as expected.

Non-Functional Testing: It is a type of testing for which every organization having a separate team which usually called as Non-Functional Test (NFT) team or Performance team. Non functional testing involves testing of non-functional requirements such as Load Testing, Stress Testing, Security, Volume, Recovery Testing etc. The objective of NFT testing is to ensure whether the response time of software or application is quick enough as per the business requirement.

It should not take much time to load any page or system and should sustain during peak load.

Performance Testing: This term is often used interchangeably with "stress" and "load" testing. Performance Testing is done to check whether the system meets the performance requirements. Different performance and load tools are used to do this testing.

Recovery Testing: It is a type of testing which validates that how well the application or system recovers from crashes or disasters. Recovery testing determines if the system is able to continue the operation after a disaster. Assume that application is receiving data through the network cable and suddenly that network cable has been unplugged. Sometime later, plug the network cable; then the system should start receiving data from where it lost the connection due to network cable unplugged.

Regression Testing: Testing an application as a whole for the modification in any module or functionality is termed as Regression Testing. It is difficult to cover all the system in Regression Testing, so typically automation testing tools are used for these types of testing.

Risk-Based Testing (RBT): In Risk Based Testing, the functionalities or requirements are tested based on their priority. Risk-based testing includes testing of highly critical functionality, which has the highest impact on business and in which the probability of failure is very high.

The priority decision is based on the business need, so once priority is set for all functionalities then high priority functionality or test cases are executed first followed by medium and then low priority functionalities. The low priority functionality may be tested or not tested based on the available time. The Risk-based testing is carried out if there is insufficient time available to test entire software and software needs to be implemented on time without any delay. This approach is followed only by the discussion and approval of the client and senior management of the organization.

Sanity Testing: Sanity Testing is done to determine if a new software version is performing well enough to accept it for a major testing effort or not. If an application is crashing for the initial use then the system is not stable enough for further testing. Hence a build or an application is assigned to fix it.

Security Testing: It is a type of testing performed by a special team of testers. A system can be penetrated by any hacking way. Security Testing is done to check how the software or application or website is secure from internal and external threats. This testing includes how much software is secure from the malicious program, viruses and how secure and strong the authorization and authentication processes are. It also checks how software behaves for any hacker's attack and malicious programs and how software is maintained for data security after such a hacker attack.

Smoke Testing: Whenever a new build is provided by the development team then the software testing team validates the build and ensures that no major issue exists.

The testing team ensures that the build is stable and a detailed level of testing is carried out further. Smoke Testing checks that no show stopper defect exists in the build which will prevent the testing team to test the application in detail. If testers find that the major critical functionality is broken down at the initial stage itself then testing team can reject the build and inform accordingly to the development team. Smoke Testing is carried out to a detailed level of any functional or regression testing.

Static Testing: Static Testing is a type of testing which is executed without any code. The execution is performed on the documentation during the testing phase. It involves reviews, walkthrough, and inspection of the deliverables of the project. Static testing does not execute the code instead of the code syntax, naming conventions are checked.
The static testing is also applicable for test cases, test plan, design document. It is necessary to perform static testing by the testing team as the defects identified during this type of testing are cost-effective from the project perspective.

Stress Testing: This testing is done when a system is stressed beyond its specifications in order to check how and when it fails. This is performed under heavy load like putting large number beyond storage capacity, complex database queries, continuous input to the system or database load.

System Testing: Under System Testing technique, the entire system is tested as per the requirements. It is a Black-box type testing that is based on overall requirement specifications and covers all the combined parts of a system.

Unit Testing: Testing of an individual software component or module is termed as Unit Testing It is typically done by the programmer and not by testers, as it requires a detailed knowledge of the internal program design and code. It may also require developing test driver modules or test harnesses.

Usability Testing: Under Usability Testing, User-friendliness check is done. Application flow is tested to know if a new user can understand the application easily or not, Proper help documented if a user gets stuck at any point. Basically, system navigation is checked in this testing.

Vulnerability Testing: The testing which involves identifying of weakness in the software, hardware and the network is known as Vulnerability Testing. Malicious programs, the hacker can

take control of the system, if it is vulnerable to such kind of attacks, viruses, and worms. So it is necessary to check if those systems undergo Vulnerability Testing before production. It may identify critical defects, flaws in the security.

Volume Testing: Volume testing is a type of non-functional testing performed by the performance testing team.

The software or application undergoes a huge amount of data and Volume Testing checks the system behavior and response time of the application when the system came across such a high volume of data. This high volume of data may impact the system's performance and speed of the processing time.

5.5Test Cases:

A test case is a document, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values , the application has a definitive outcome and leaves the system at some end point or also known as execution post condition.

Login:

Test case 1:	Login
Precondition:	User must have internet connection, local host server
Action:	User entered the correct e-mail address and password.
Expected result:	User will be logged in.
Tested by:	Muhammad usman
Results:	Successful

Table 6: Test case 1

Login:

Test case 2:	Login
Precondition:	User must have internet connection, local host server
Action:	User entered the wrong e-mail address and password.
Expected result:	User will be logged in.
Tested by:	Muhammad usman
Results:	Fail

Table 7: Test case 2

Friend request:

Test case 3:	Friend request
Precondition:	User must have internet connection, local host server
Action:	User search an other user and send him friend request. Other user receive the friend request and clicked on accept button.
Expected result:	They will be friends.
Tested by:	Muhibullah
Results:	Successful

Table 8: Test case 3

Friend request:

Test case 4:	Friend request
Precondition:	User must have internet connection, local host server
Action:	User search an other user and send him friend request. Other user receive the friend request and clicked on ignore button.
Expected result:	They will be friends.
Tested by:	Muhibullah
Results:	Fail

Table 9: Test case 4

Update password:

Test case 5:	Update password
Precondition:	User must have internet connection, local host server
Action:	User open the setting and click on update password button. User entered the old and new password correctly.
Expected result:	Password will be changed.
Tested by:	Muhibullah
Results:	Successful

Table 10: Test case 5

Update password:

Test case 6:	Update password
Precondition:	User must have internet connection, local host server
Action:	User open the setting and click on update password button. User entered the wrong old password.
Expected result:	Password will be changed.
Tested by:	Muhibullah
Results:	Fail

Table 11: Teat case 6

Positive Test Case:

ID TC_LOGIN_SUCCESS

Priority High

Description to verify user authentication to system

. Reference Functional Requirement reference

Users Administrator.

Pre-requisites A: System is online. B User must have active login credentials provided by system

Administrator. C User has internet access.

Steps A: Open the web link to system. B Enter login id C Enter Password. D Press Login. Input

Login id and password Expected result successfully enters the system and main home page

Opens. Status Tested, passed.

Negative Test Case :

ID TC_LOGIN_FAILURE if they put wrong password or email

Priority High for all developers

Description to verify user authentication to system.

Reference Functional Requirement reference

Users Administrator and User.

Pre-requisites A: System is online. B User must have active login credentials provided by system

Administrator. C User has internet access.

Steps A: Open the web link to system. B Enter login id. C Enter Password. D Press Login.

Input Incorrect Login id or password or deactivated credentials. Expected result does not allows

Access to system features and notifies the error.

5.6 Implementation:

Structured programming: Structured programming (sometimes known as modular programming) is a subset of procedural programming that enforces a logical structure on the program being written to make it more efficient and easier to understand and modify. Certain languages such as Ada, Pascal, and dBase are designed with features that encourage or enforce a logical program structure.

Top-down analysis-: In a Nutshell. Top-down programming starts at the top and then works its way down. Or, as Wikibooks puts it. "A top-down approach (also known as stepwise design) is essentially the breaking down of a system to gain insight into the sub-systems that make it up.

Modular-programming: Modular programming is a software design technique that emphasizes separating the functionality of a program into independent, interchangeable modules, such that each contains everything necessary to execute only

Function programming: In computer science, functional programming is a programming paradigm-a style of building the structure and elements of computer programs that treats computation as the evaluation of mathematical functions and avoids changing state and mutable data. It is a declarative programming paradigm in that programming is done with expressions or declarations instead of statements. Functional code is idempotent a function's return value depends only on its arguments, so calling a function with the same value for an argument always produces the same result. This is in contrast to imperative programming where, in addition to a function's arguments, global program state can affect a function's resulting value. Eliminating side effects, that is, changes in state that do not depend on the function inputs, can make understanding a program easier, which is one of the key motivations for the development of functional programming

Coding guideline: It is used by physicians, other health care providers, and payers to classify diseases, injuries, health encounters and inpatient procedures. The "CM" stands for Clinical Modifications and is used for medical diagnoses. The "PCS" is a new Procedure Coding System developed for inpatient procedures

- **Indentation**

Use tabs (and configure your IDE to show a size of 8 spaces for them) for writing your code (hopefully we can keep this consistent). If you are modifying someone else's code, try to keep the coding style similar. Since we are using 8-space tabs, you might want to consider the Linux Torvalds trick to reduce code nesting. Many times, in a loop, you will find yourself doing a and if the test is true, you will nest. Many times, this can be changed.

- **Naming convention**

This section defines how name functions, variables, constants and global variables.

- **Whitespace**

It is generally omitted at the end of line

- **Operators**

Defines the rules of writing mathematical, assignment and logical operators for example, assignment operator should have space before and after it, as in "

Control structures the rule of writing if-then-else, cave-switch, while until and for control flow statements solely and in nested fashion.

- **Variables**

This defines how functions should be declared and invoked, with and without parameters

- **Comments**

This is one of the important coding components, as the comments included in the code descrite what the code actually dose and all other associated descriptions. This section also helps creating help documentations for other developers

Software documentation: Software documentation is written text or illustration that accompanies computer software or is embedded in the search code. The documentation either explains how the software operates or how to use it or may not different things to people in

Chapter No. 6 User Manual

This is the login authentication screen from where the user can log in to the system. User must have:

1. **Open login webpage.**

2. **Enter login details.**

3. **Click Login.**

4. **Done.**

6.1 Login page:

This is log in page.If user do not have account he can select register button to make an account.

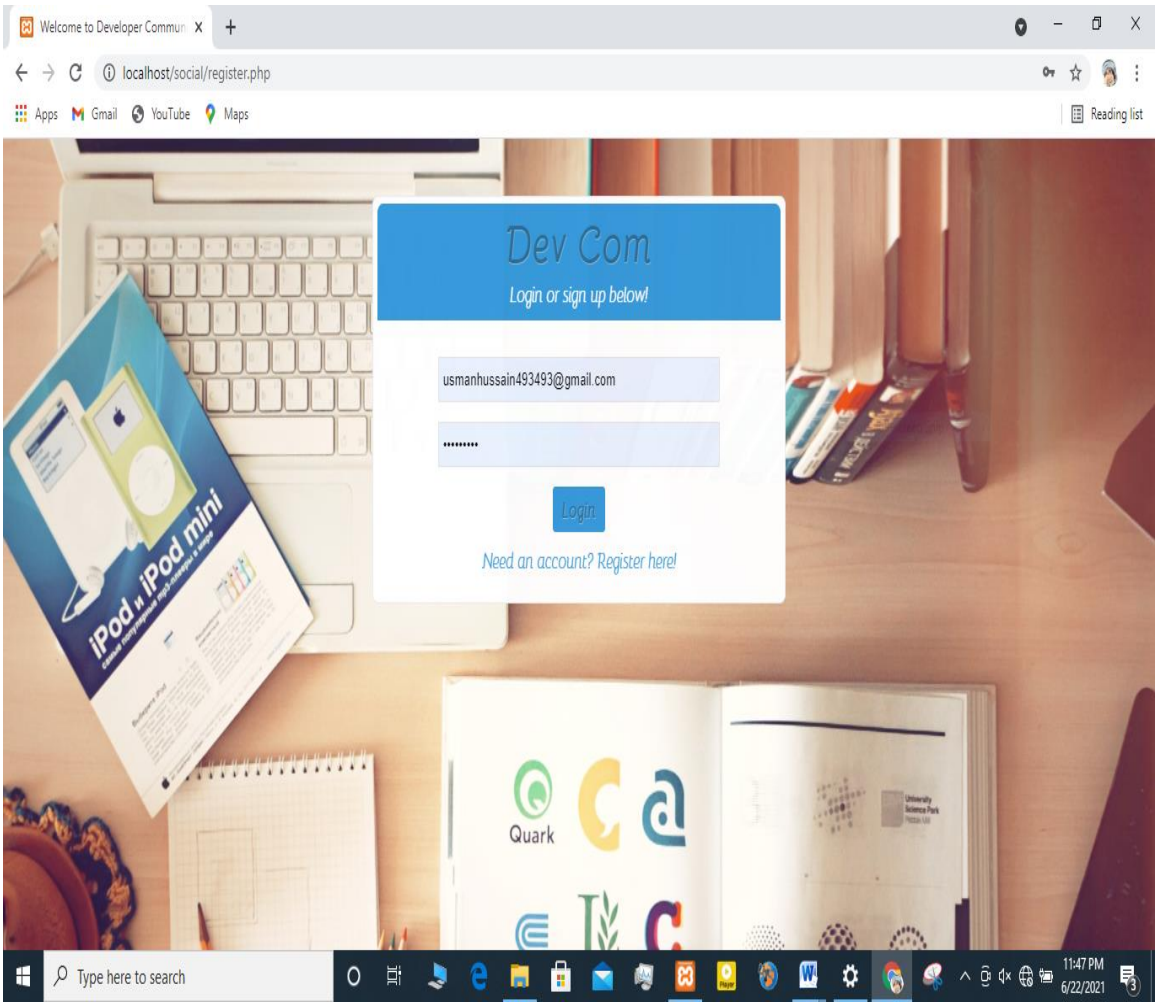

Figure 10: Login page

6.2 Registration Page:

Registration page. After entering credentials user can make the account.

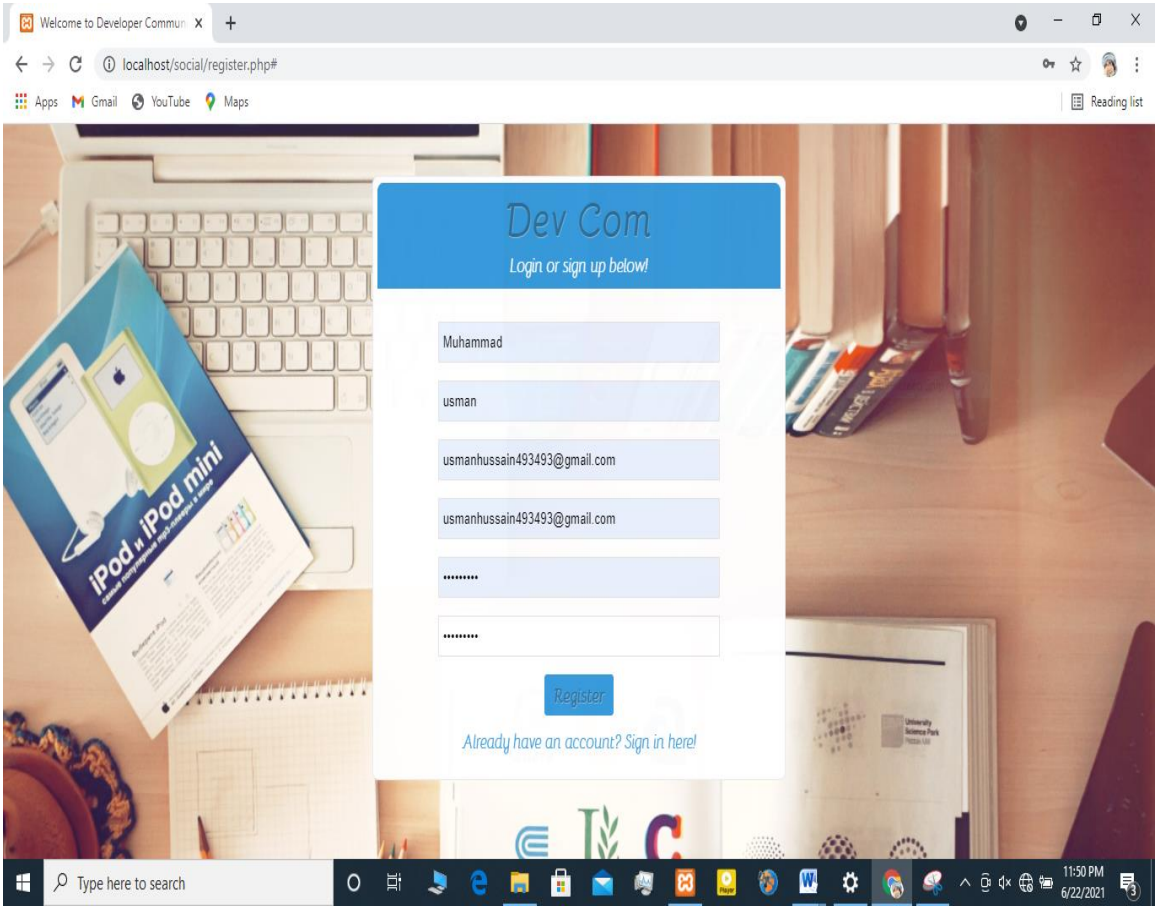

Figure 11: Registration page

6.3 Profile:

This is the user profile page post, likes and friends will show here.

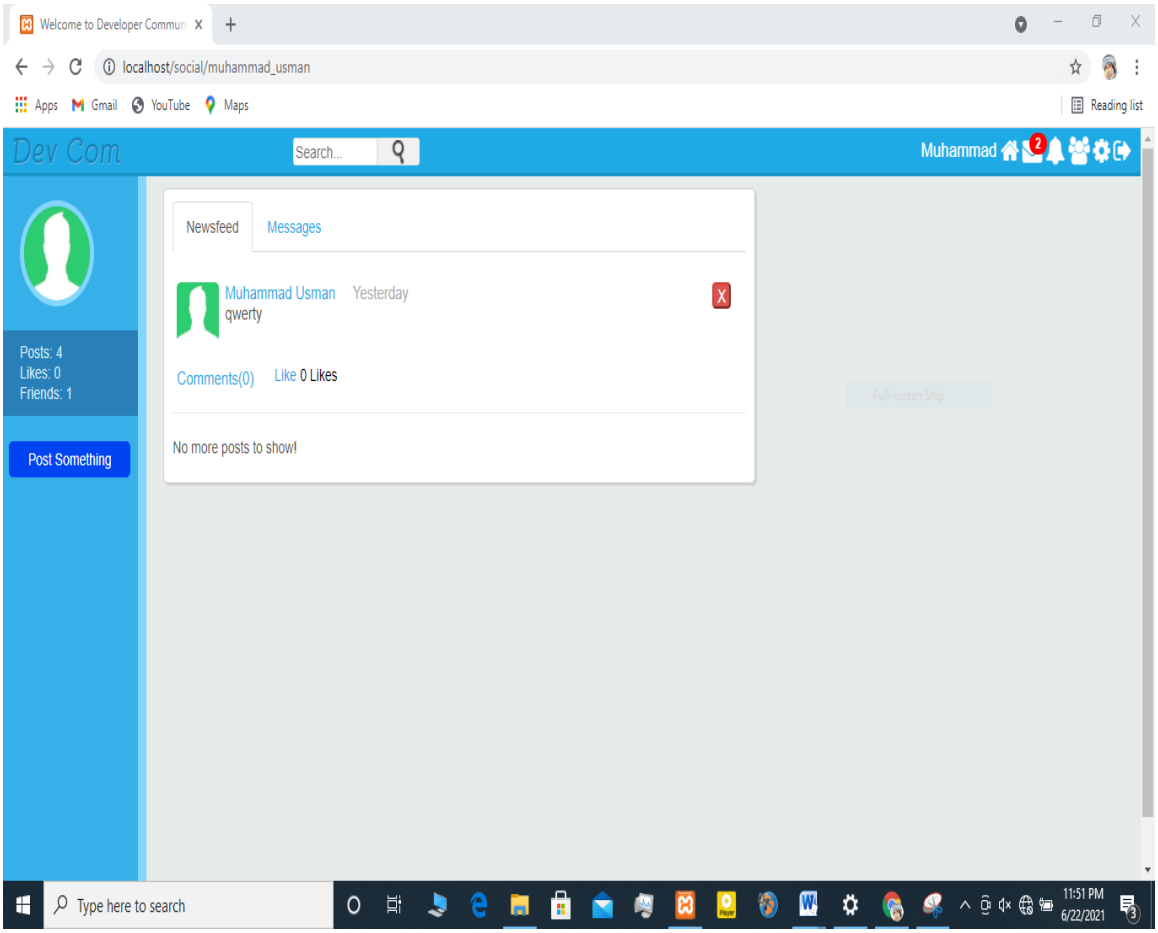

Figure 12: Profile

6.4 Newsfeed & Post:

This is the newsfeed section. All post from friends will show here.

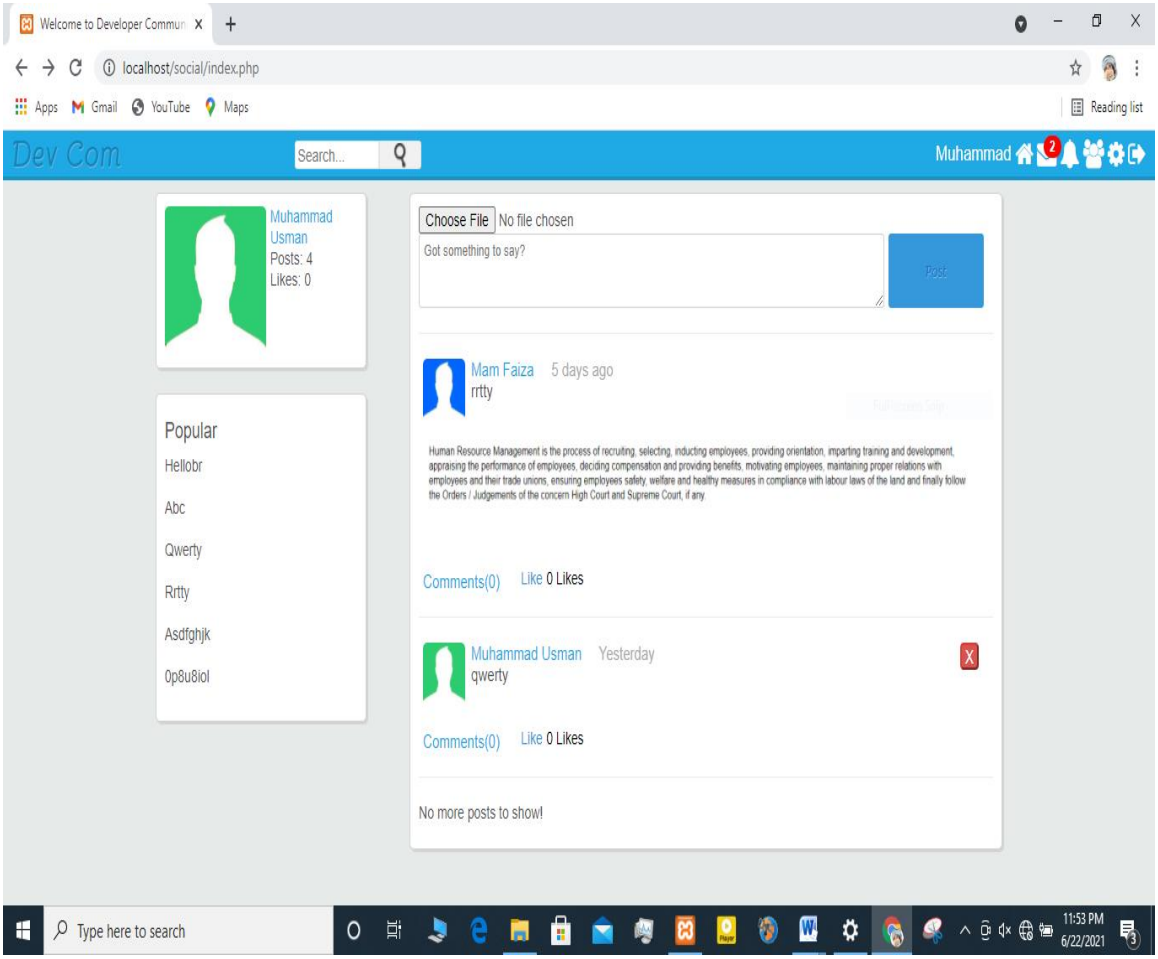

Figure 13: Newsfeed & Post

6.5 Messages & Chat:

This is the chat section. User can receive and send message from here.

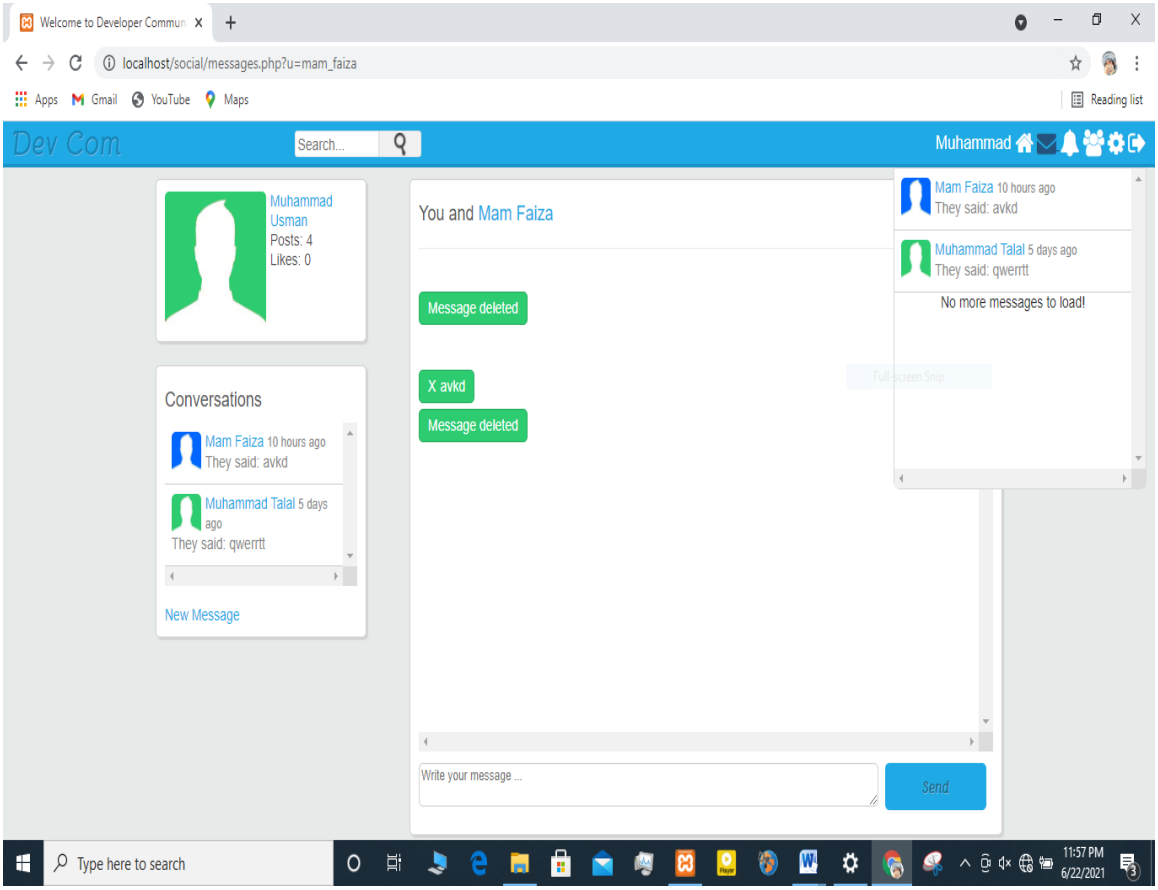

Figure 14: Messages & Chat

6.6 Friend Request:

User can accept or ignore friend request.

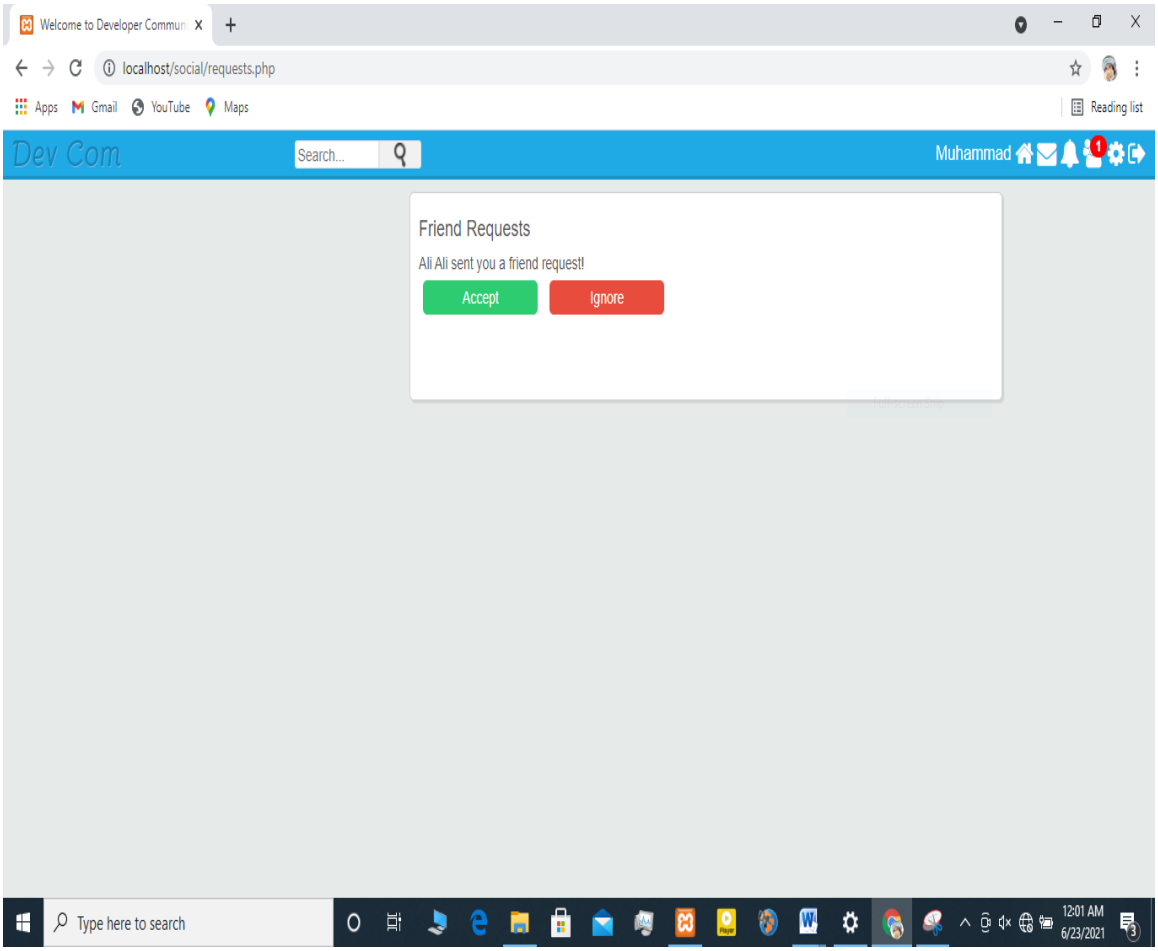

Figure 15: Friend Request

6.7 Account settings:

User can update their e-mail account password and delete their account form here.

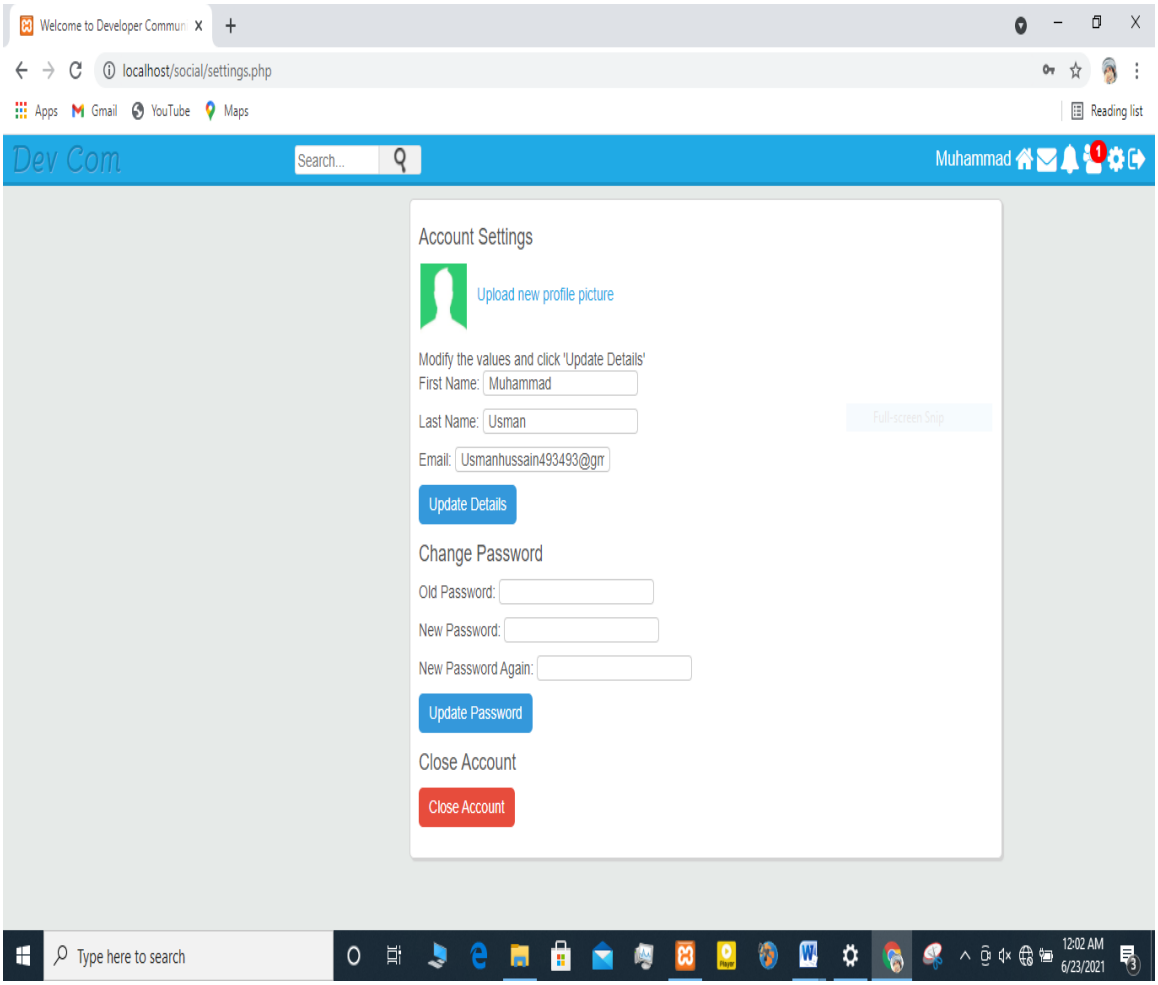

Figure 16: Account setting

6.8 Notification Panel:

The user receive the notification of message or any friend request from any other user.

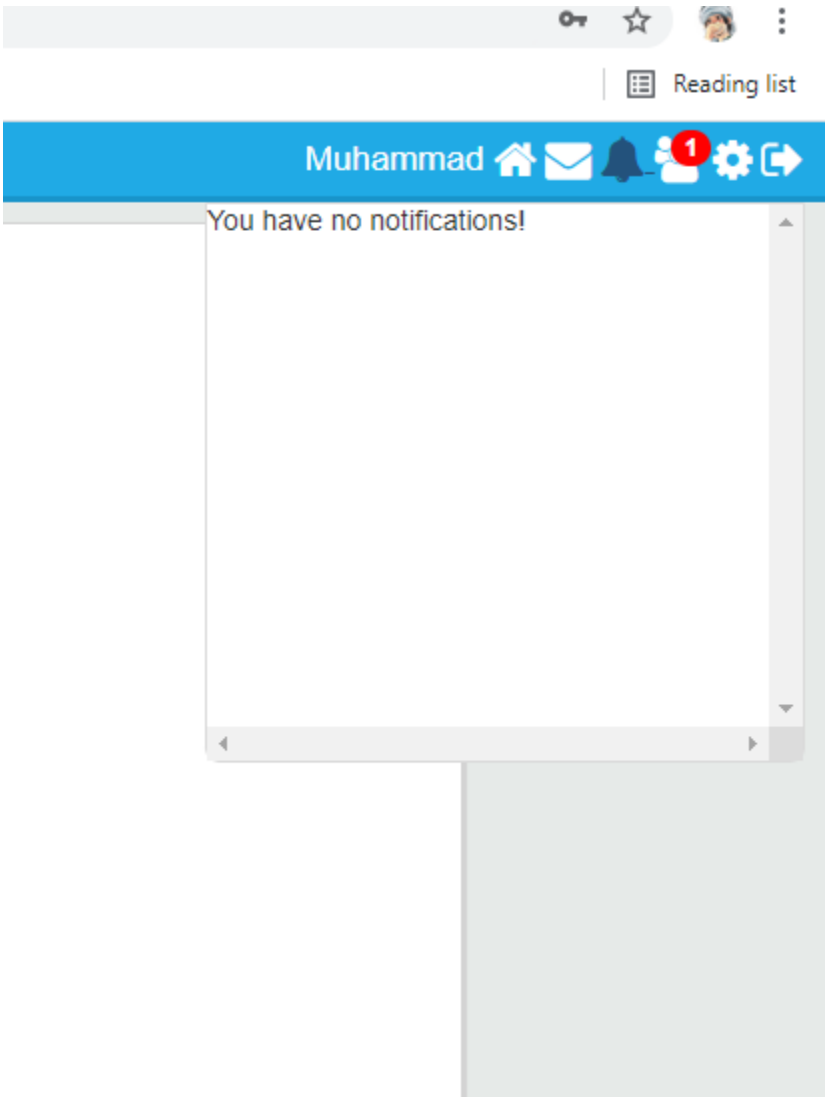

Figure 17: Notification panel

6.9 Search bar:

User can search any user account from here. And user also search anything about this web site from this search bar

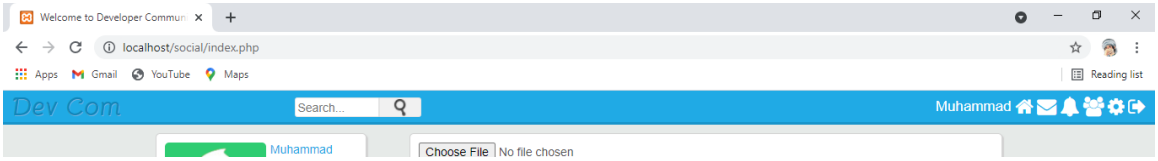

Figure 18: Search bar