

**Dharmsinh Desai University, Nadiad**

**Faculty of Technology**

**Department of Computer Engineering**

**B. Tech. CE Semester – VI**

Subject: (CE – 619) **SERVICE ORIENTED COMPUTING**

Project Title**: *Instagram Replica WCF Service***

**Submitted by**:

Name: **Swar Patel**

Roll No: **CE096**

ID: **17CEUOG132**

Name: **Priyank Chaudhari**

Roll No: **CE104**

ID: **17CEUTG028**

Name: **Vyom Pathak**

Roll No: **CE099**

ID: **17CEUON038**

Guided By:

**Prof. Ankit P. Vaishnav**

Assistant Professor, CE Dept.,

Dharmsinh Desai University



Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

**CERTIFICATE**

This is to certify that Service Oriented Computing’s project entitled “**Instagram Replica WCF Service**” is the bonafied report of work carried out by

1. **Patel Swar (17CEUOG132)**
2. **Priyank Chaudhari (17CEUTG028)**
3. **Pathak Vyom (17CEUON038)**

Of Department of Computer Engineering, Semester VI, academic year 2019-20,

under our supervision and guidance.

|  |  |
| --- | --- |
| Guide | HOD |
|  |  |
| **Prof. Ankit P. Vaishnav** | **Dr. C. K. Bhensdadia** |
| Assistant Professor of | Head of the Department of |
| Department of Computer | Department of Computer |
| Engineering, Dharmsinh Desai University, Nadiad. | Engineering, Dharmsinh Desai University, Nadiad. |

|  |  |  |
| --- | --- | --- |
| **No.** | **Content** | **Page No.** |
| **1** | Introduction | **5** |
| **2** | Software Requirement Specifications | **6** |
| **3** | Design Documents | **12** |
| **4** | Implementation Details | **18** |
| **5** | Functionality Prototype | **20** |
| **6** | Testing Details | **28** |
| **7** | Conclusion | **40** |
| **8** | Limitations and Future Extensions | **41** |
| **9** | Bibliography | **42** |

**Overview:**

Instagram Replica WCF Service was made to provide basic functionality of instagram i.e. users can create their accounts, post photos and comments on different posts. User can also follow other users and like/dislike-e the posts. The basic application is made using the Windows Form Application. The business logic is compartmentalized into different services which are essentially WCF services which are hosted on a host application and the proxy of the host is used whenever any operation is to be called from the original application.

**Contribution by Each Member:**

**Swar Patel -** Created the Host Application. Created the User Post Service and its related operation’s business logic.

**Priyank Chaudhari -** Created the Windows form application that will include the host proxy and will contain all the UI related to the application. Created the User Comment Post Service and its related operation’s business logic.

**Vyom Pathak –** Created the User Management Service and all its related operation’s business logic.

**Introduction**

The main object of this project is learn how to build a WCF service and host it on a host application and call it from different application. The project definition is ‘Instagram Replica WCF Service’. The functionalities include the following: User can create, update and delete their account. User can follow/unfollow other users. User can search other users using different parameters. User can post photos and add different parameters to it. User can also update an existing post as well as delete an existing post. User can like/dislike on his/her own post as well on other user’s post that he/she is following. User can comment on posts and can also update an already create comment by that particular user on some post. User can also delete a comment from a particular post.

**Technology Used:**

* Windows Form Application to create application.
* WCF Service for handling user, post and comment CRUD operations.
* Microsoft Sql Server as database. [MSSQL]
* UMLET for Designing Different UML Diagram.

**Tools Used:**

* We used Github as a version control method and to manage the project.
* We used Visual Studio for development of the code.

**Software Requirement Specification**

**Instagram Replica WCF Service**

**FUNCTIONAL REQUIREMENTS:**

**R1. Manage User:**

DESCRIPTION: Users can login and can create new account and view others users by searching using the username and using email address and delete his/her account. User can follow other users. User can update his/her profile details.

**R1.1 Sign-Up User:**

DESCRIPTION: Users can register to the application by providing his/her details like password, username, email address and date of birth information.

INPUT: Users information.

OUTPUT: Registration successful message.

**R1.2 Sign-In User:**

DESCRIPTION: Users can sign-in to his/her account using his/her email and password or using his/her username.

INPUT: User information.

OUTPUT: Sign-In to respective user profile.

**R1.3 Delete User:**

DESCRIPTION: User can delete his/her account from the application by selecting the delete account option.

Input: User selection

Output: Success message is displayed.

**R1.4 Search User:**

DESCRIPTION: User can search other user using username or using user email-address.

**R1.4.1 Search using username:**

Input: User gives username as input

Output: User with the given username is displayed

**R1.4.2 Search using Email-address:**

Input: User gives email address as input.

Output: User with the given email-address is displayed.

**R1.5 Update User Details:**

DESCRIPTION: User provides the necessary details to be updated and selects the update option.

Input: User information

Output: Success message is displayed.

**R2 Manage Posts:**

DESCRIPTION: User can create new post, user can delete a post and user can update an existing post. User can see all the posts of the users he/she is following. User can like and unlike his/her posts and all the user’s posts that he/she is following.

**R2.1 Create Post:**

DESCRIPTION: User inputs information like post caption, post location, post image.

Input: Post Information.

Output: Confirmation Message will be displayed.

**R2.2 Update Post:**

DESCRIPTION: User can update his/her post and give the new post information to be updated and select the update option.

Input: Post Information.

Output: Confirmation Message will be displayed.

**R2.3 Delete Post:**

DESCRIPTION: User selects the post to be deleted and select the delete option to delete it.

Input: User selection

Output: Confirmation Message will be displayed.

**R2.4 View Posts:**

DESCRIPTION: User can see all other user’s posts as soon he/she sign-ins.

Input: User selection

Output: All the posts of the following users are displayed.

**R2.5 Like/Unlike Posts:**

DESCRIPTION: Users can select the like button to like/ dislike other user’s posts.

Input: User selection

Output: Post is liked/disliked according to user selection.

**R3. Manage Comments:**

Description: This functionality allows users to comment on other user’s that he/she is following. It also allows users to delete his/her comments. It also allows users to update comments done by him/her on the posts. User can also view all the comments on the selected post.

**R3.1 Create Comment:**

DESCRIPTION: User selects the appropriate post to comment on and enters the comment text and selects the create option to create comment on that post.

Input: User selection and comment information.

Output: Created comment will be displayed.

**R3.2 Update Comment:**

DESCRIPTION: User can update the comment text that he/she wishes to change on a particular post.

Input: User selection and Comment information.

Output: Update message will be prompted and updated comment will be displayed.

**R3.3 View Comment:**

DESCRIPTION: User can select the post and see the comments on the selected post.

Input: User selection.

Output: Comments on the selected post will be displayed.

**R3.4 Delete Comment:**

DESCPRTION: User can select the post and select the comment which he/she wants to delete.

Input: User selections

Output: Confirmation message will be displayed.

**R4. Manage Followers:**

DESCRIPTION: User can follow other users. User can unfollow other users. User can also see the list of users he/she is following.

**R4.1 Follow User:**

DESCRIPITION: User selects the user to be followed by him/her.

Input: User Selection.

Output: Success message is displayed.

**R4.2 Unfollow User:**

DESCRIPITION: User can select the user to be unfollowed by him/her.

Input: User selection.

Output: Success message is displayed.

**R4.3 View Following Users:**

DESCIPITION: User can see other users which he/she is following.

Input: User selection

Output: List of users he/she is following is displayed.

**NON FUNCTIONAL REQUIREMENTS:**

SOFTWARE: Visual Studio, Github Desktop

OS: Windows

**Design Documents**

1. Use Case Diagram:

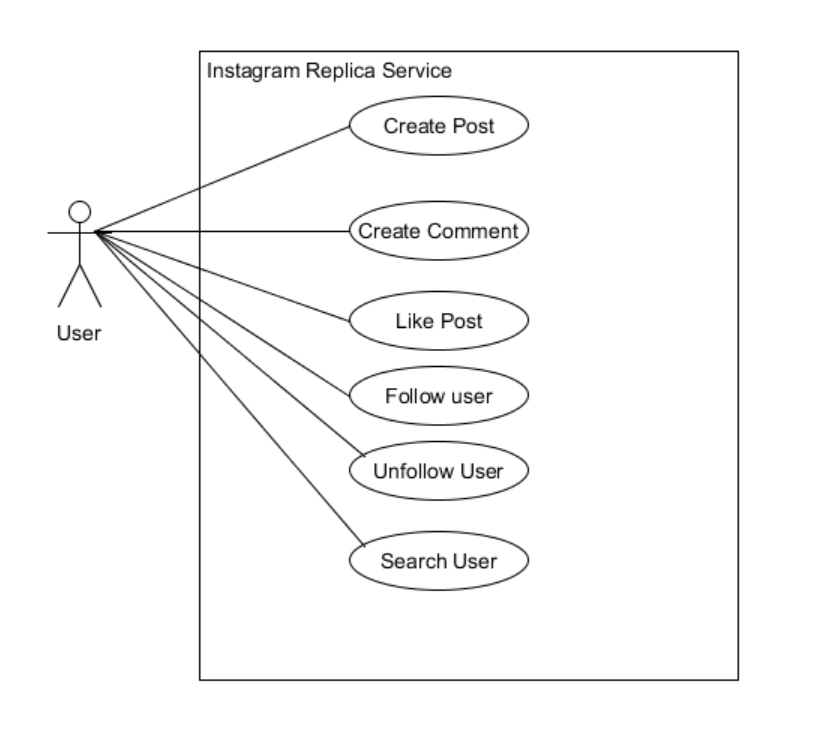


Figure 1 : Use Case Diagram

1. Class Diagram:

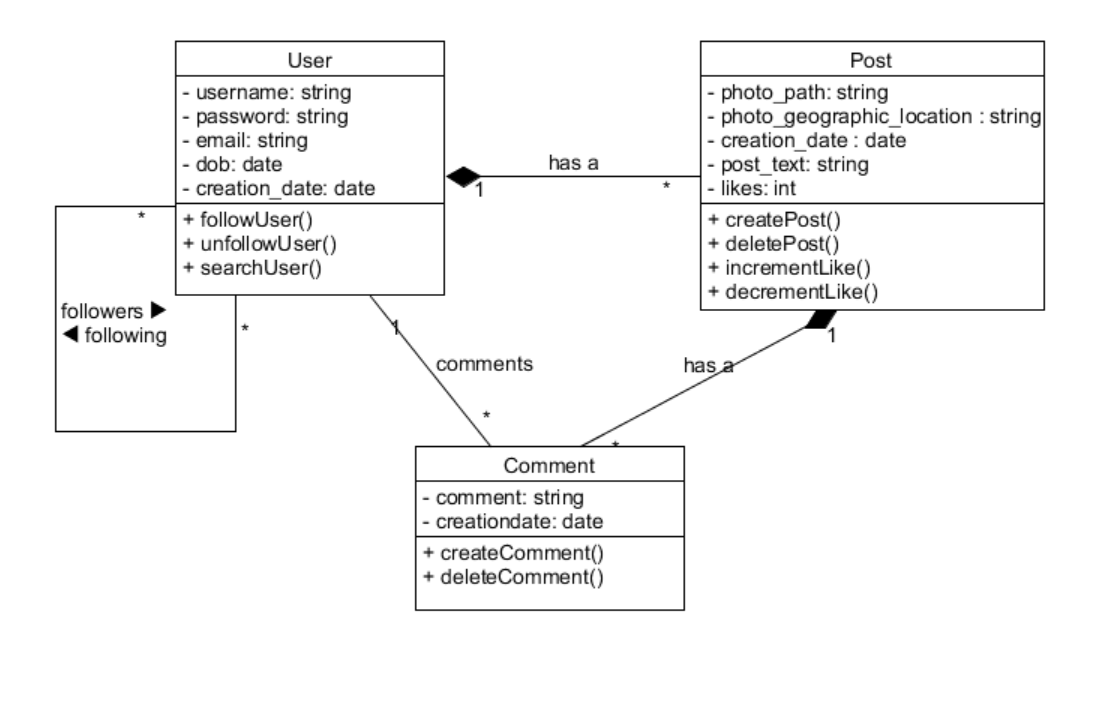


Figure 2 : Class Diagram

1. Sequence Diagram:

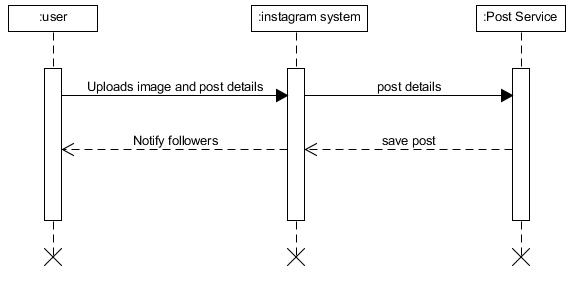


Figure 3 : Sequence Diagram

Creating post

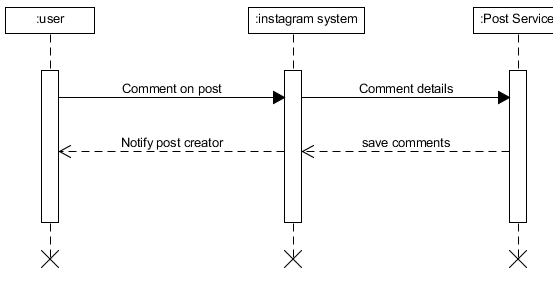


Figure 4 : Sequence Diagram

Commenting on post

1. Activity Diagrams :

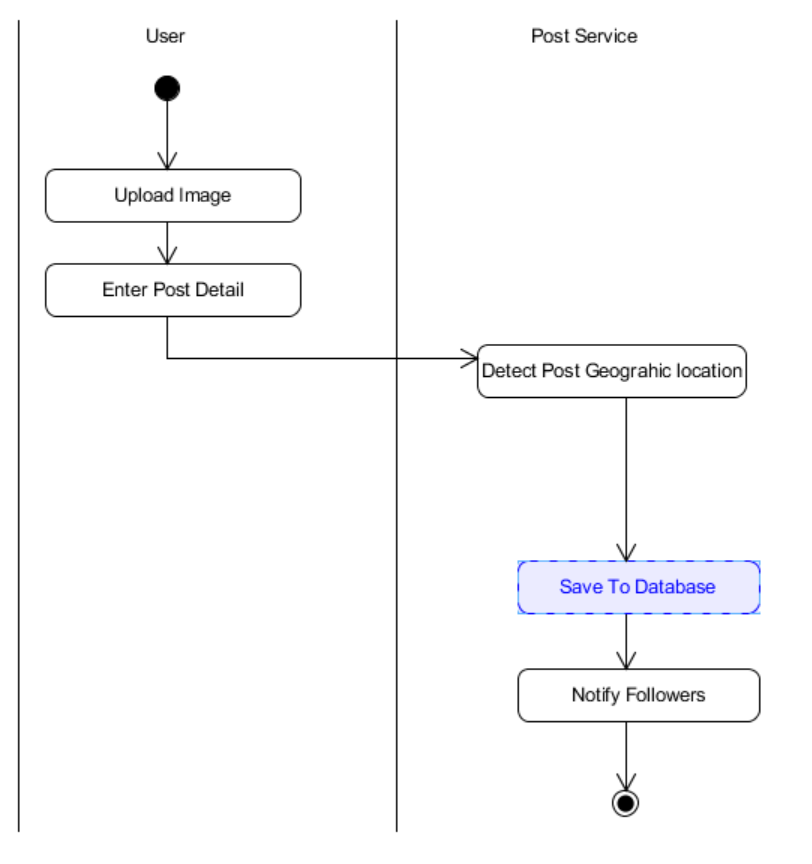
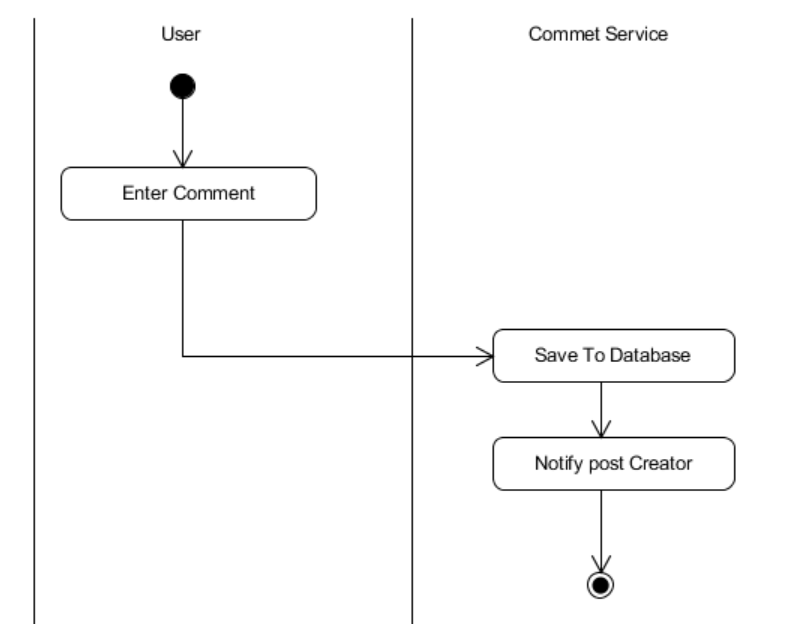


Figure 5 : Activity Diagram

Create Post



Create Comment

Figure 6 : Activity Diagram

5. Data Dictionary:

|  |  |  |  |
| --- | --- | --- | --- |
|  | User |  |  |
| Attributes | Datatype | Size | Feature |
| userId | Int | 20 | AUTO\_INC [PK] |
| username | nvarchar | 50 |  |
| email | nvarchar | 50 |  |
| dob | Datetime | 20 |  |
| creation date | Datetime | 20 |  |
| password | nvarchar | 50 |  |

Table 1 : Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
|  | Post |  |  |
| Attributes | Datatype | Size | Feature |
| postId | Int | 20 | AUTO\_INC [PK] |
| userId | Int | 20 | FK |
| photopath | nvarchar | MAX |  |
| location | nvarchar | 50 |  |
| creation date | Datetime | 20 |  |
| post text | nvarchar | MAX |  |
| likes | Int | 20 |  |

Table 2 : Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
|  | Comment |  |  |
| Attributes | Datatype | Size | Feature |
| commentId | Int | 20 | AUTO\_INC [PK] |
| userId | Int | 20 | FK |
| postId | Int | 20 | FK |
| comment | nvarchar | MAX |  |
| creation date | Datetime | 20 |  |

Table 3 : Data Dictionary

|  |  |  |  |
| --- | --- | --- | --- |
|  | User Follow |  |  |
| Attributes | Datatype | Size | Feature |
| userId1 | Int | 20 | FK |
| userId2 | Int | 20 | FK |

Table 4 : Data Dictionary

**Implementation Details**

**Sign In And Sign Up Module:** User sign-In is done by calling the user service to check for the provided email and password from the database and if it correct than the User’s object is created and used for that particular session. For sign-up functionality, user provides his/her information and the details will be stored into the database using the user service operation.

**Search User Module:** User can search other user by typing in the email address or by typing username and the query will call the user service operation appropriately to find the user detail from the database and will return the user detail.

**Delete User Module:** Whenever user wishes to delete his/her account the appropriate user service operation is called and that particular user information is removed from the database.

**Update User Module:** Whenever user wants to update his/her details appropriate user service operation is called and the user information is updated into the database.

**Post CRUD Module:** User can create, view other people’s post, update his/her own post and delete his/her own post. All the functionalities was completed using the Post Service and by calling the appropriate operation of the service to complete the task and the database was updated accordingly.

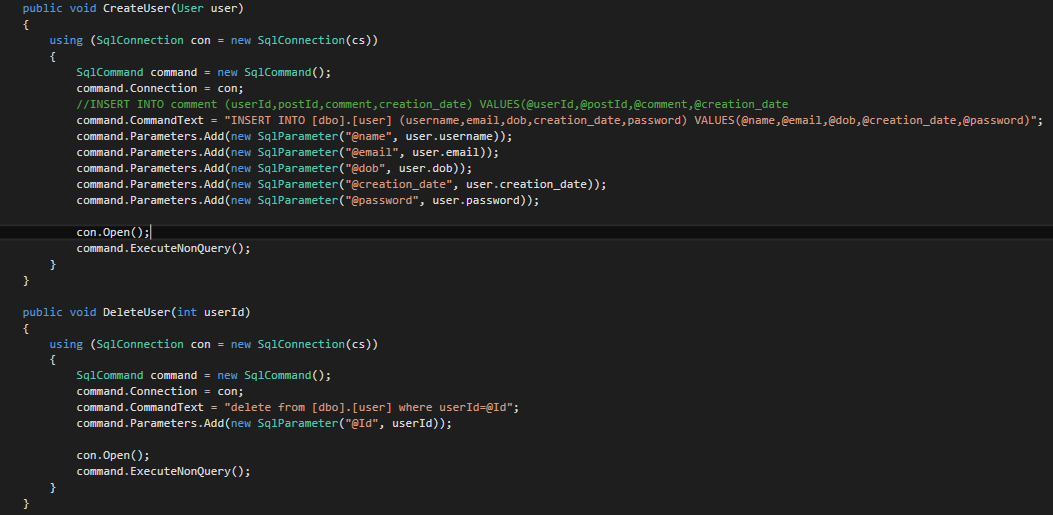
**Comment CRUD Module:** User can comment on posts, user can update already created comment, user can view all the comments of other user’s on his/her post and user can also delete his/her own post and these all functionality was achieved by calling the respective operations from the comment service by the users according to the task and the database was updated accordingly.

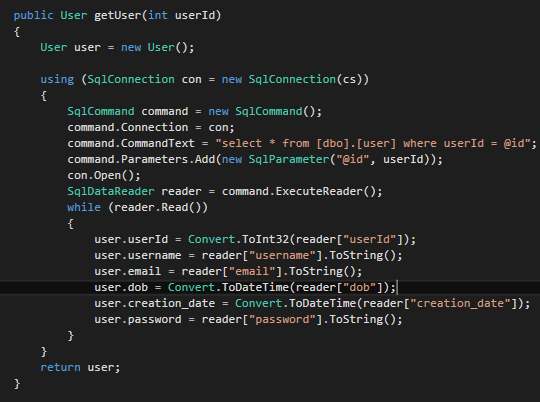
**Follow/Unfollow User:** User can follow or unfollow user and this functionality was achieved by using appropriate operations of the UserFolow Service which will complete the task and update the database accordingly.

**Like/Dislike Post:** User can like/dislike a post and this functionality was achieved using appropriate operations of the Post Service which will complete the task and update the database accordingly.

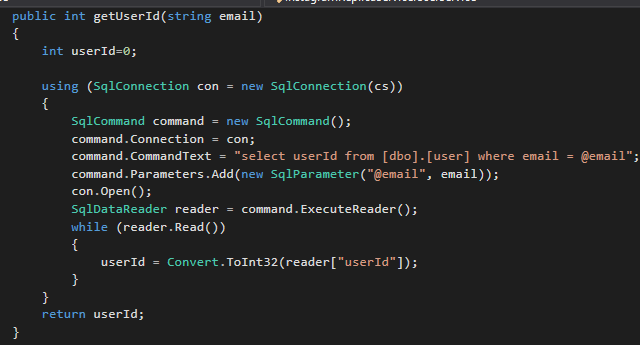
**Function Prototype:**

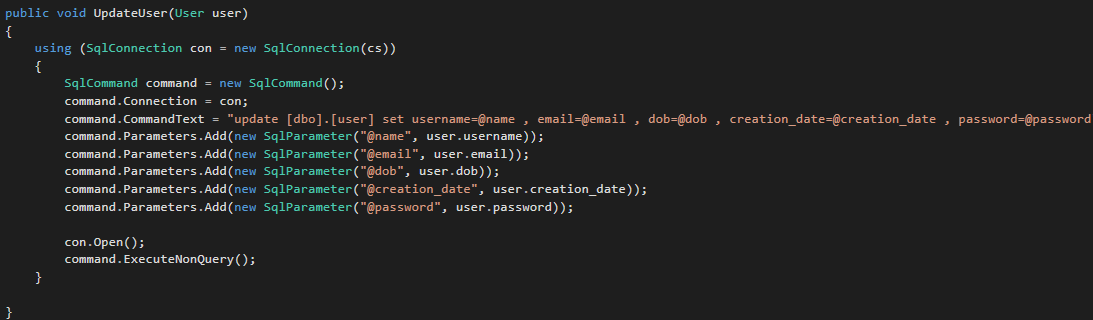
**User Functionality Service [CRUD]:**

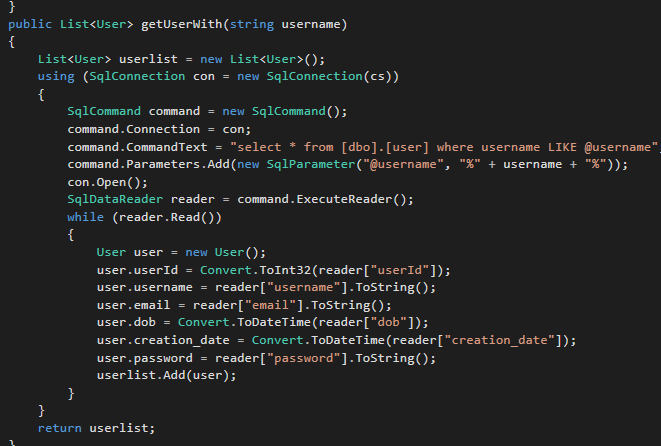
****

****

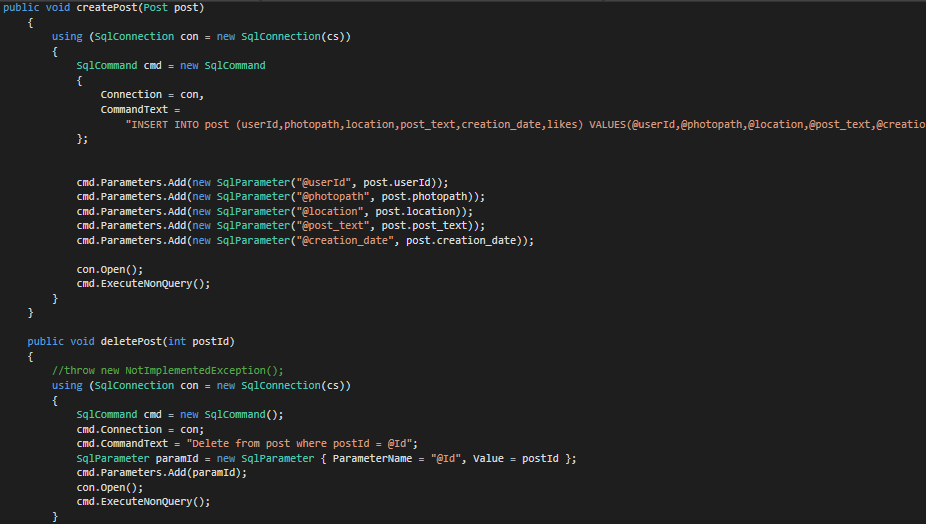
****

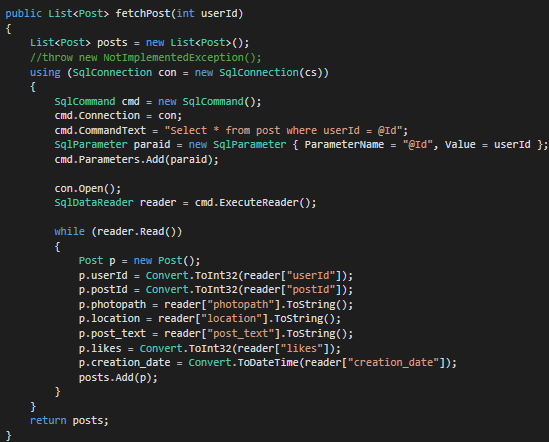
****

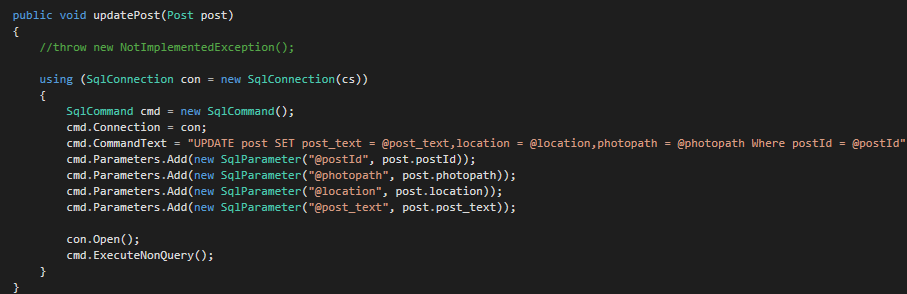
****

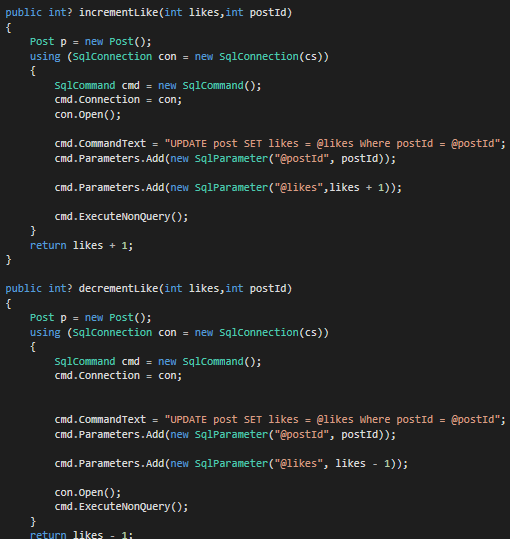
****

**Post Functionality Service [CRUD]:**

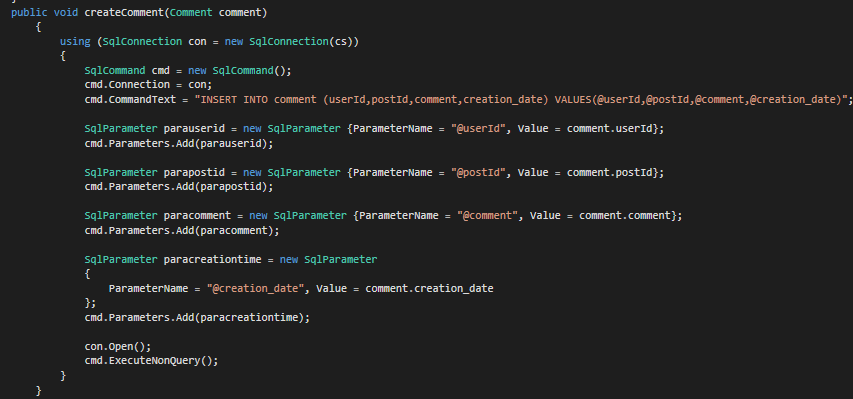
****

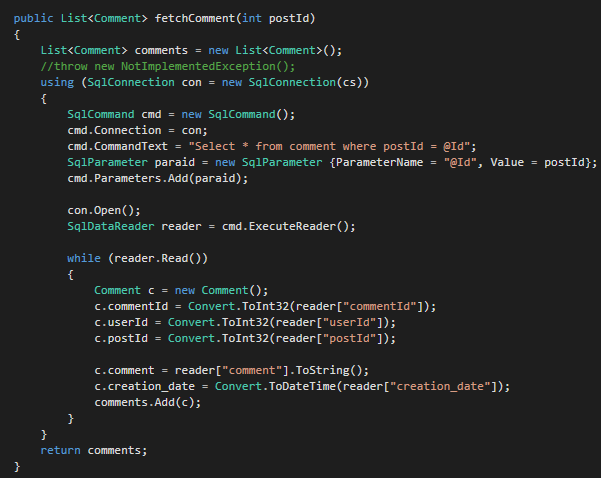
****

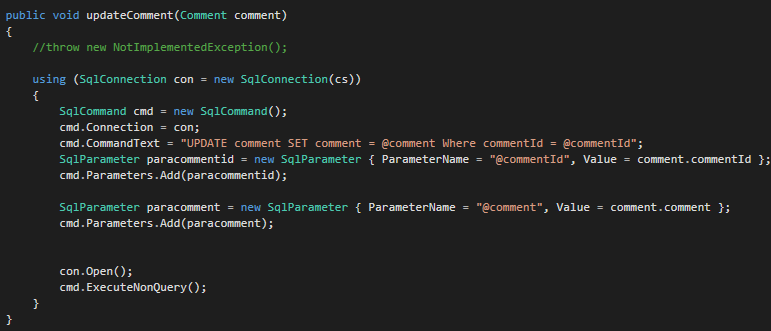
****

****

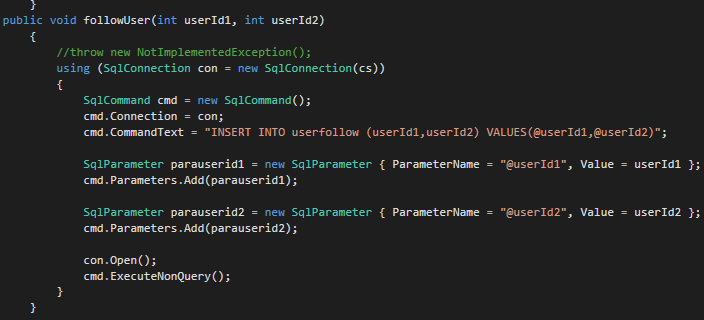
**Comment Functionality Service [CRUD]:**

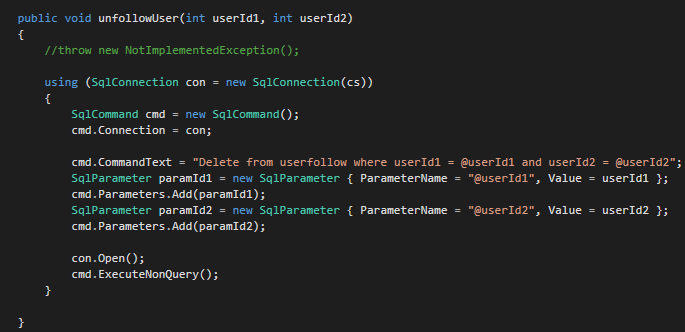
****

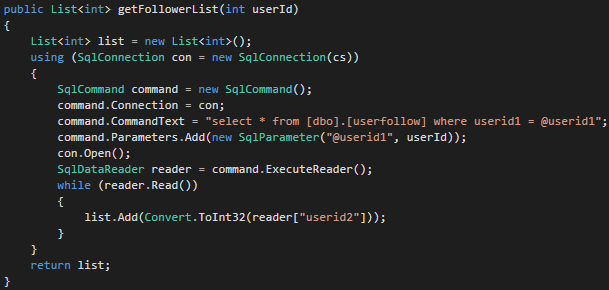
****

****

**User Follow Functionality Service [CRUD]:**

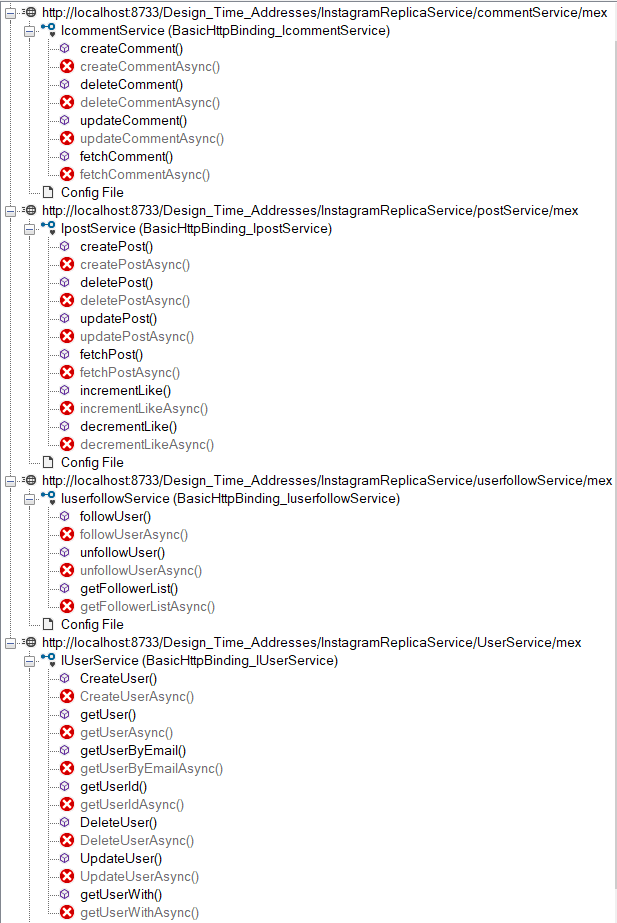
****

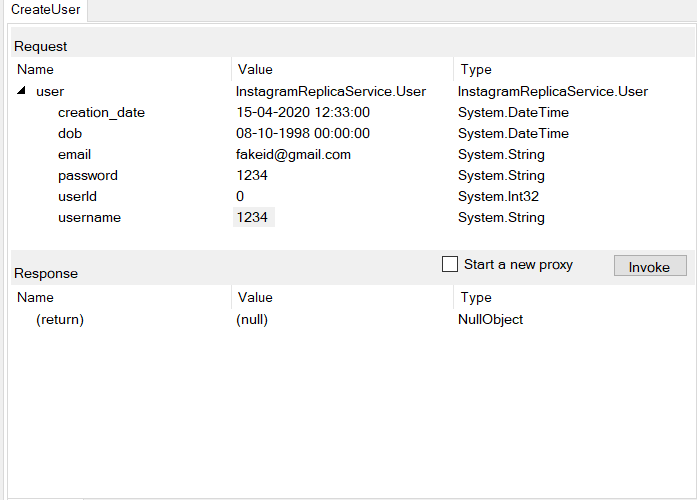
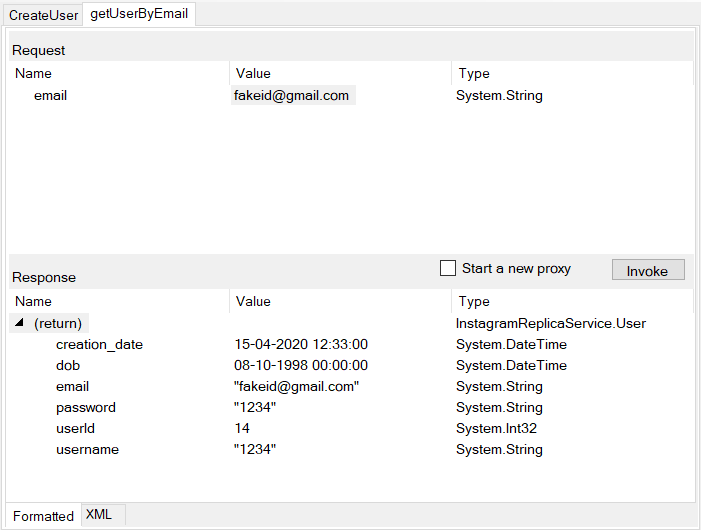
****

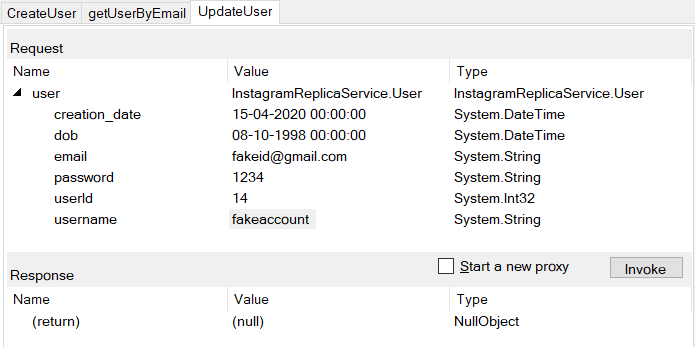
****

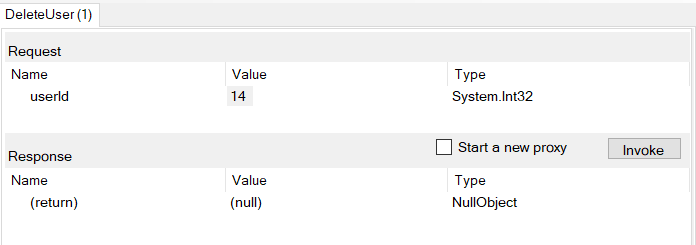
**Testing Details**

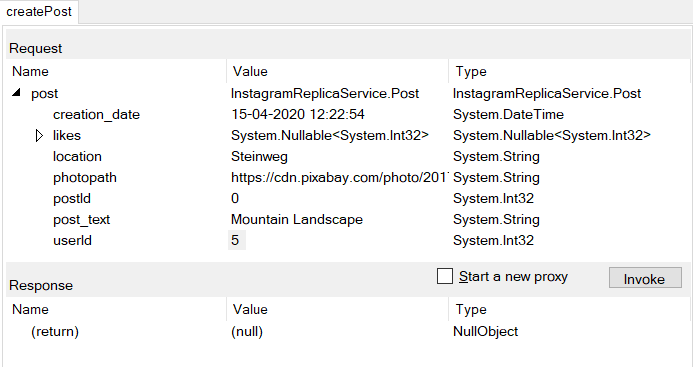
Testing of services was done in WCF test client. Below are some important test scenarios

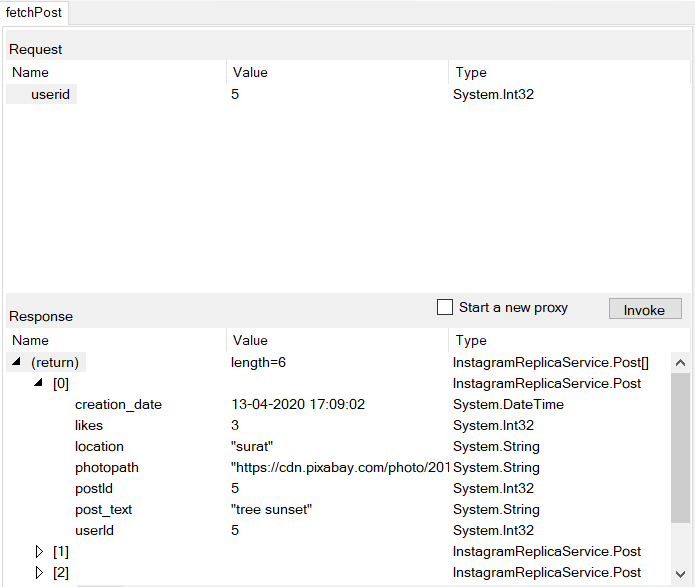


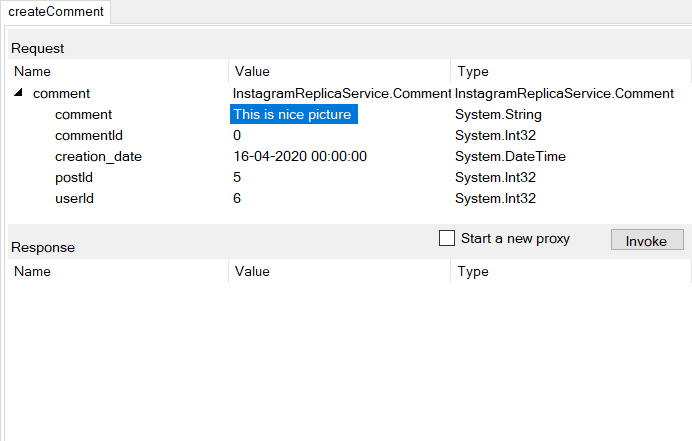


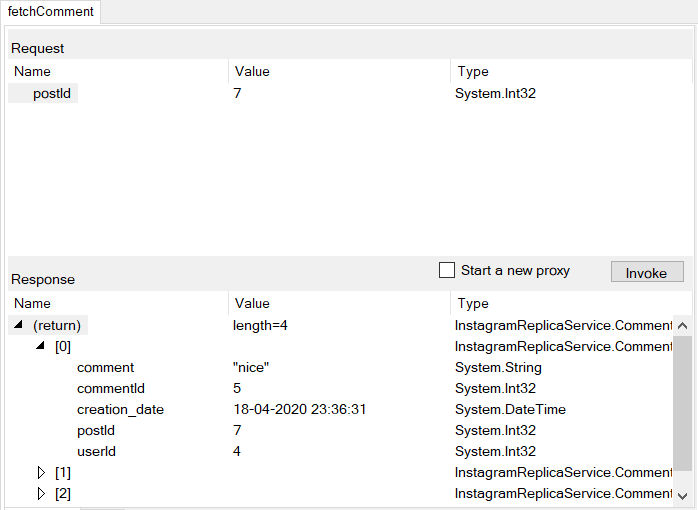


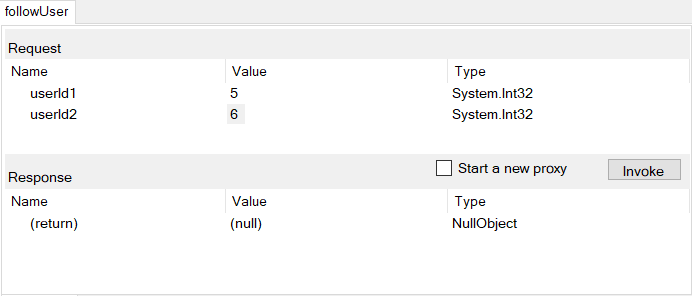


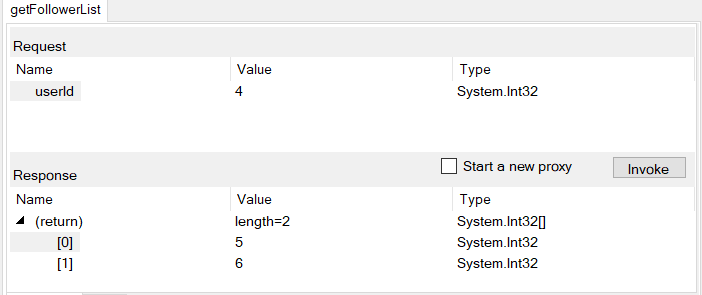












**Screen-Shots**

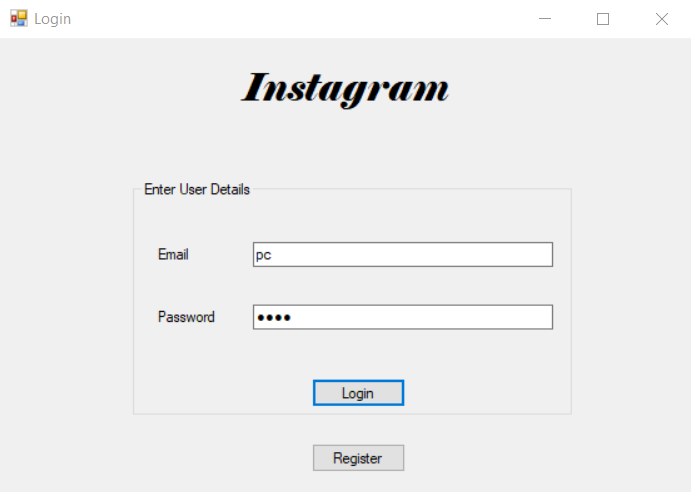
****

Figure : Login Module

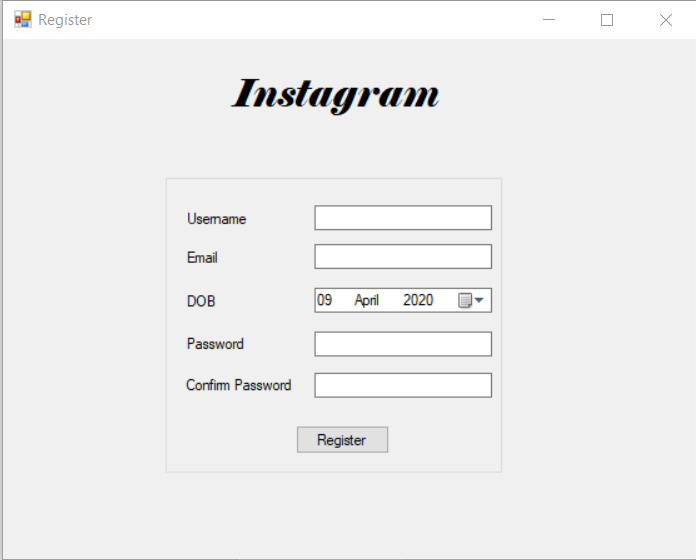


Figure : Register Module

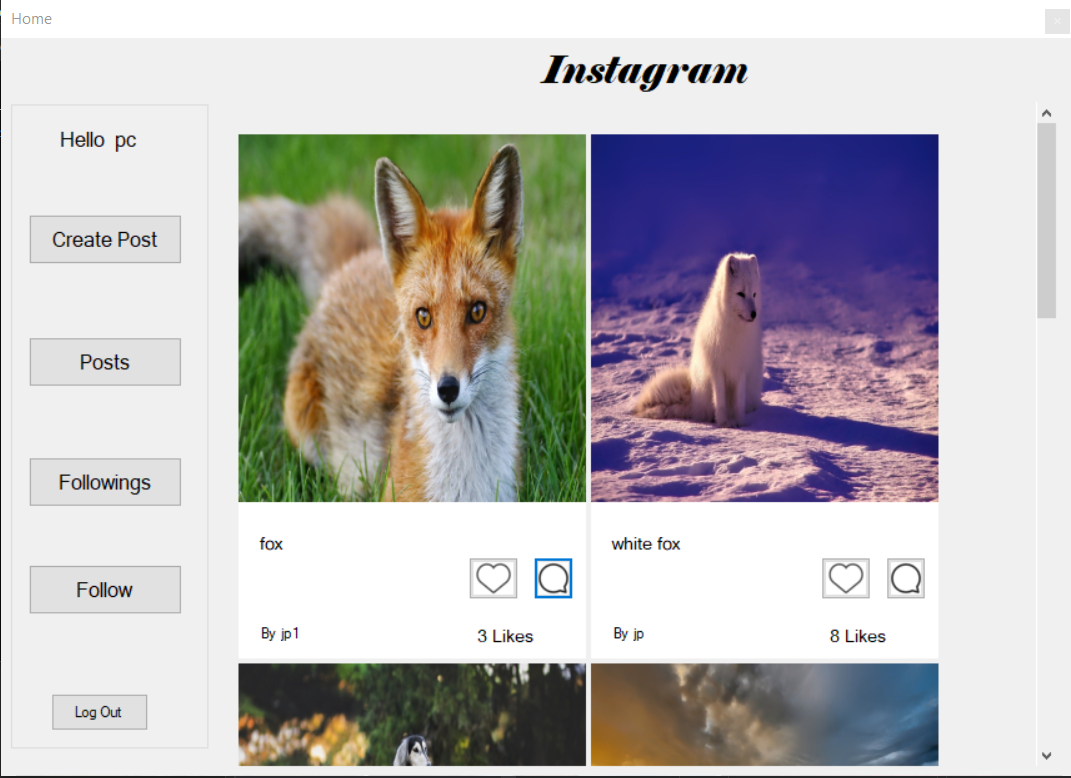


Figure : Home Dashboard of user

(You can also see likes and title of post along with the name of user who posted)

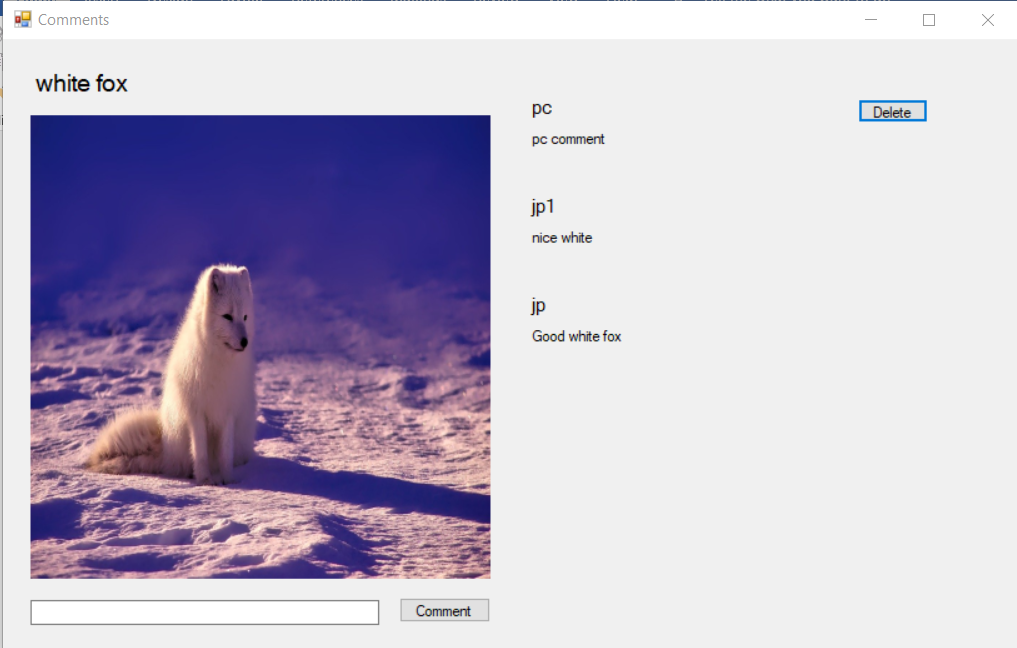


Figure : Comment on a post and user can delete its own comment

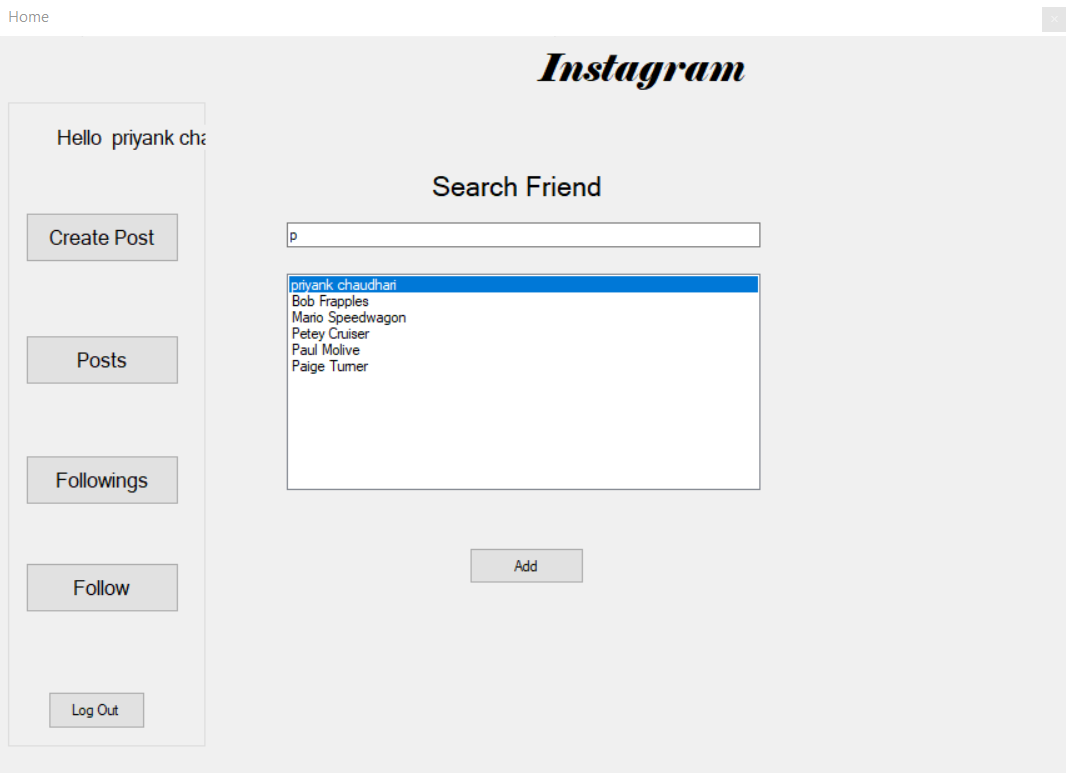
****

Figure : Finding Friends on application

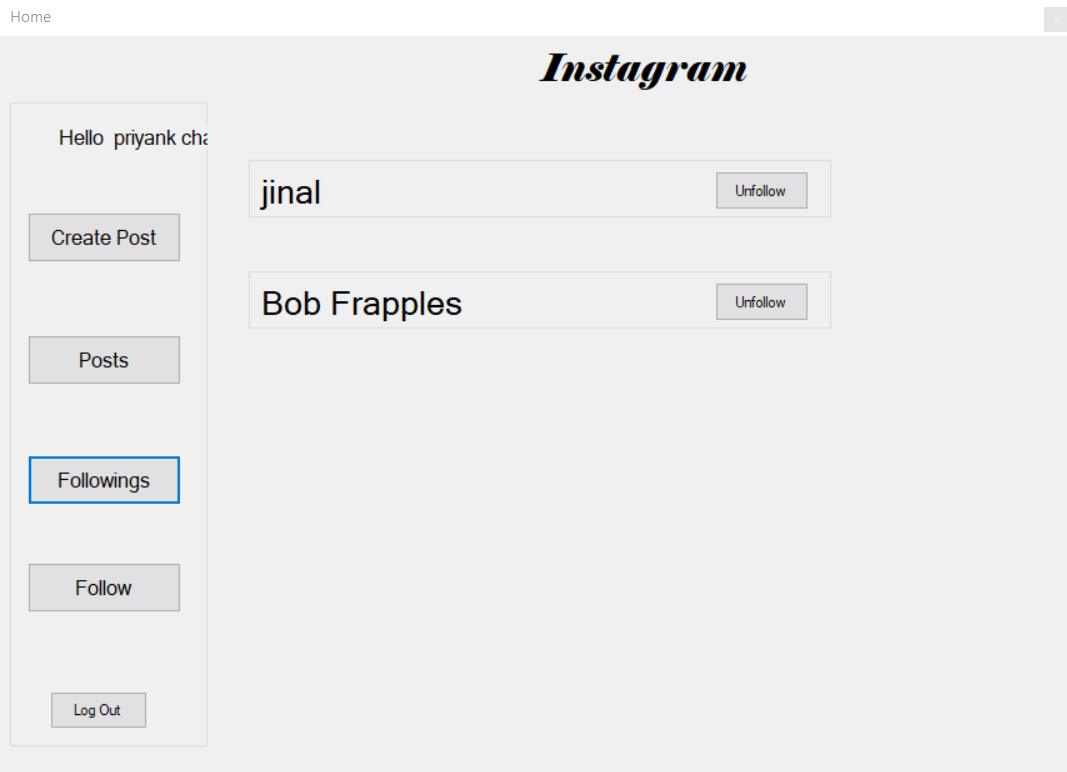


Figure : Friends you follow

**Conclusion**

We successfully created Instagram replica service which follows principal of service oriented architecture. We tested our application using windows form application. We learned benefits of Service Oriented Architecture. We learned how to divide task to different service and get benefited from that.

We learned in which condition we should use service oriented architecture and in which condition we should not.

**Limitation and Future Extension**

**Limitations:**

* While creating this project we created post which are already hosted on internet. We are not storing image at server side. Because we don’t know how to pass images using soap which is xml and there is only one way to communicate with service is soap.

**Future Extension:**

* Maintaining who has given likes and comments.
* Use multiple database for scalability. (e.g. Store user data in one database and use this database via user service and userfollow service, create other database for storing post data and use this database via post service and comment service. )
* Receive request only one service and redirect to particular service for fetching data after receiving data aggregate that and then give back to the user.

**Bibliography**

**References:**

<https://stackoverflow.com/>

<https://www.youtube.com/>

<https://www.geeksforgeeks.org/>

<https://docs.microsoft.com/en-us/dotnet/>

<https://www.codeproject.com/>

<https://www.c-sharpcorner.com/>