



# Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

B.Tech. CE Semester – VI

Subject: (CE-619) SERVICE ORIENTED COMPUTING

Project Title:

## BIKE DEKHO

By:

Parth Baudhanwala

(Roll No: CE012 Id: 17CEUBS013)

Meet Charola

(Roll No: CE017 Id: 17CEUOS071)

Guided By:

Prof. Ankit P. Vaishnav



# Dharmsinh Desai University, Nadiad

Faculty of Technology, Department of Computer Engineering

## CERTIFICATE

This is to certify that Service Oriented Computing project entitled “BIKE DEKHO” is the benefited report of work carried out by

1) **Parth Baudhanwala** (17CEUBS013)

2) **Meet Charola** (17CEUOS071)

Of Department of Computer Engineering, Semester VI, academic year 2019-20, under your supervision and guidance.

---

**Guide**

**Prof. Ankit P. Vaishnav**  
Assistant Professor of  
Department of Computer  
Engineering, Dharmsinh Desai  
University, Nadiad.

**HOD**

**Dr. C. K. Bhensdadia**  
Head of the Department of  
Department of Computer  
Engineering, Dharmsinh Desai  
University, Nadiad.

## Table of Contents

1 Abstract.....	4
2 Introduction.....	5
2.1 Project Details: Brief Introduction .....	5
2.2 Technology and Tools Used.....	6
3 Software Requirement Specifications .....	7
3.1 Scope .....	7
3.2 System Functional Requirements .....	7
3.3 Other Non- Functional Requirements.....	8
4 Design.....	9
4.1 Use Case-Diagram.....	9
4.2 Class-Diagram.....	10
4.3 Sequence-Diagram.....	10
4.4 Activity-Diagram.....	11
4.5 E-R Diagram .....	12
4.6 Data Dictionary.....	12
5 Implementation Details .....	13
5.1 Functional Prototypes .....	15
6 Testing .....	18
6.1 Testing Method.....	18
6.4 Test Cases .....	18
7. Screen-shots of the System .....	19
8. Conclusion .....	21
9. Limitations and Future Extensions of System .....	22
10. Bibliography .....	22

“Bike Dekho” is an online platform where you can sell and buy second hand bikes. It is similar to OLX but with limited scope of only bikes. People sometimes not able to find a satisfied buyer with broker. So here we provide facility to add description of that bike with price tag that is favourable to you. You can negotiate for buying and selling both. This gives you agility, global scale and durability, with “anytime, anywhere” data access.

## 2. Introduction

---

Have you seen OLX or Car Dekho??

This is almost same kind of website. Every bike that is on sell around your nearer location is now at your screen!!

You can explore every client description and make a good deal with him. Clients will upload photos too so you can see your dream bike from your home and negotiate at your preferable amount. And our platform is completely secure so no worries at all.

## **2.2 Tools/Technologies Used**

### **Technologies:**

Windows Communication Foundation (WCF) for  
Service ASP.NET for client  
Entity Framework (6.x) for database

### **Tools:**

Visual Studio 2017

### **Platforms:**

localhost/8000 for Service  
localhost/ 57449 for Client

# 3. Software Requirement Specifications

---

## **3.1 Product Scope**

It provides User to sell and buy bikes from uploading details of their bikes from home.

### **Users: -**

1. User

### **Description: -**

You can explore all bikes in surrounding you which are on sell at your home!

### **High level requirement: -**

#### R.1) Login Module

##### R1.1) Login

Description: Provide a unique identity to users.

Input: Credentials

##### R1.2) Register

Description: User can Register.

Input: User details.

##### R1.3) Log out

Description: User Can Logging out.

Input: User Click.

#### R.2) Bike Operation

##### R2.1) Upload Image

Description: User can upload an image.

Input: Image URL.

Output: Byte stream of image.

#### R2.2) Remove Image

Description: User can Remove an image.

Input: Image URL.

Output: Removed from User list.

#### R2.3) Update Bike

Description: User can change Price, Model, Company, Description, Image of bikes that is uploaded by him.

Input: values according to field.

Output: Update will be shown in user list.

#### R2.4) My Bikes

Descriptions: User can view all bikes list that are uploaded by him.

Input: click.

Output: List of all bike uploaded by user.

### **3.4 Other Nonfunctional Requirements**

#### **1. Performance**

The system must be interactive and the delays involved must be less. So, in every action-response of the system, there are no immediate delays.

#### **2. Safety**

User details should be securely stored to the server. The main security concern is for user account hence proper login mechanism should be used to avoid hacking.

#### **3. Reliability**

As the system provides the right tools for discussion, problem solving it must be made sure that the system is reliable in its operations and for securing the sensitive details.

#### **4. Database**

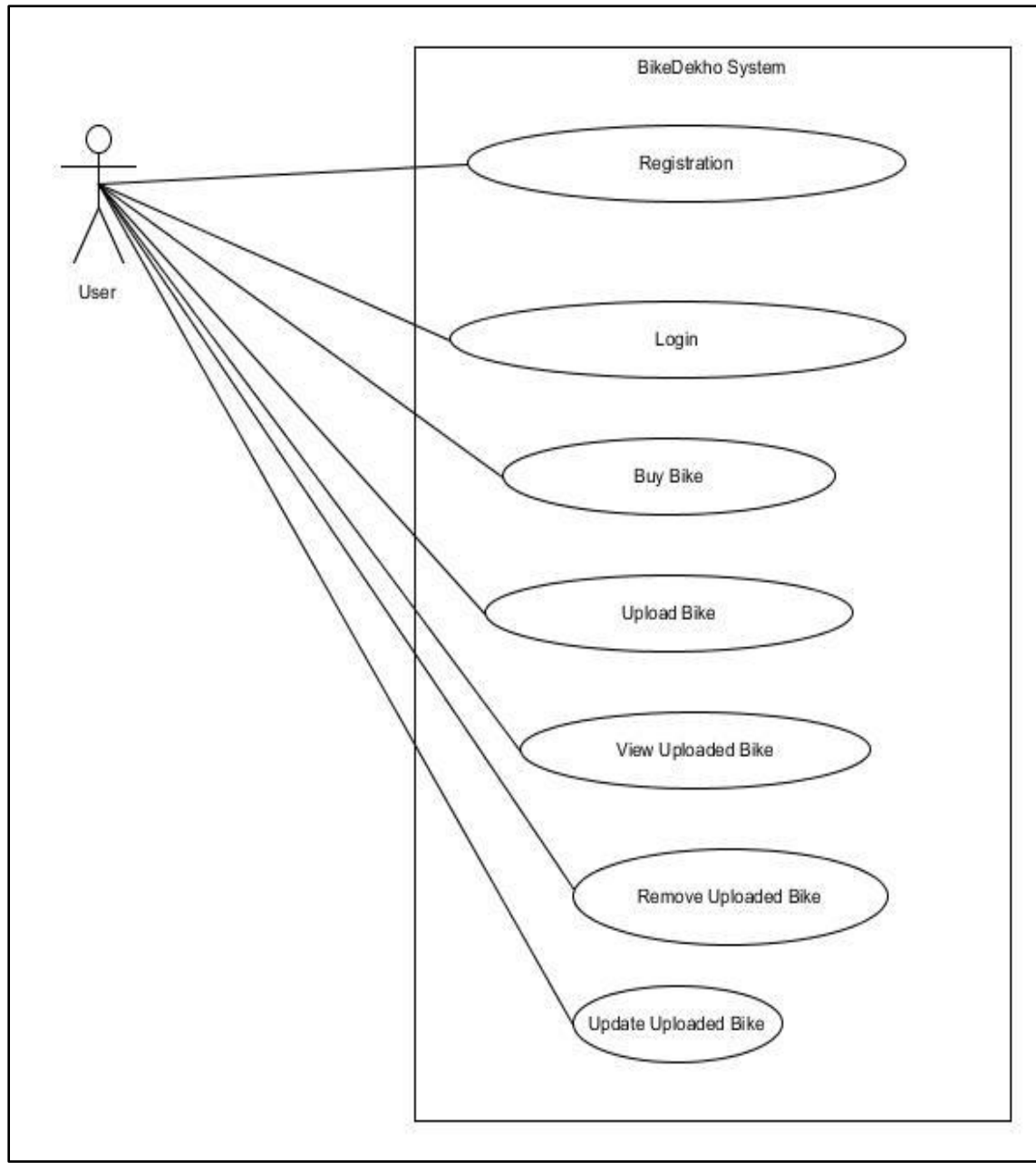
System requires to access user's data fast to maintain the performance.



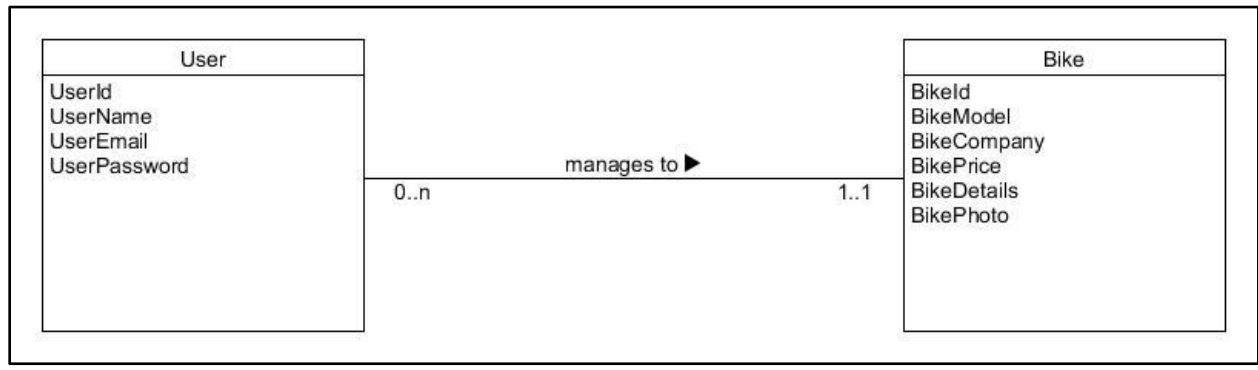
# 4. Design

---

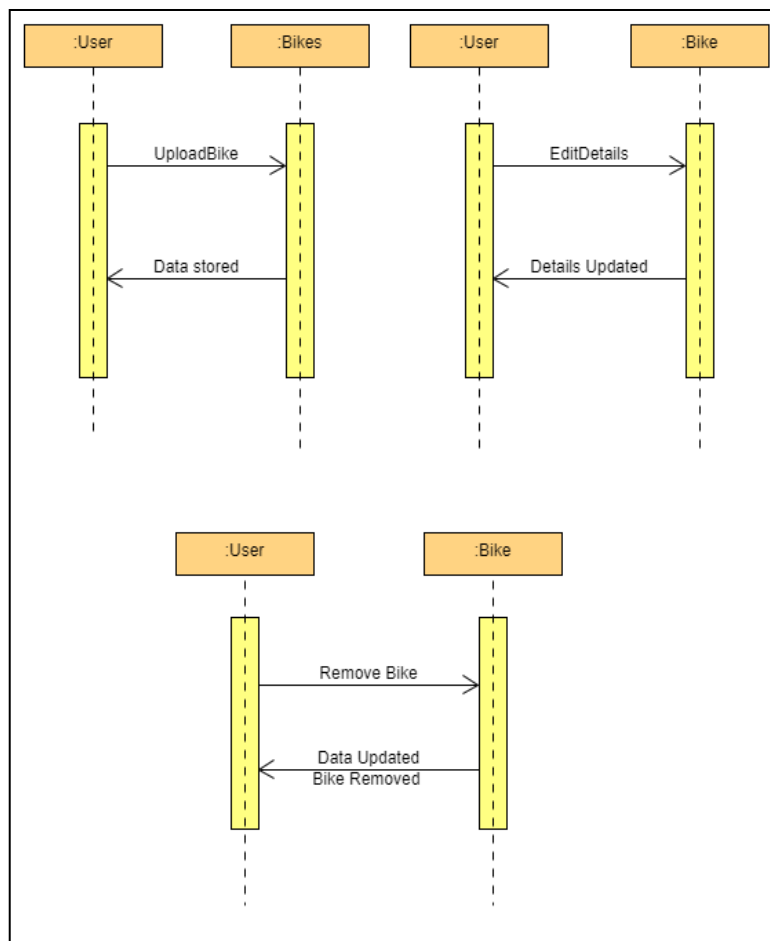
## 4.1 Use Case Diagram



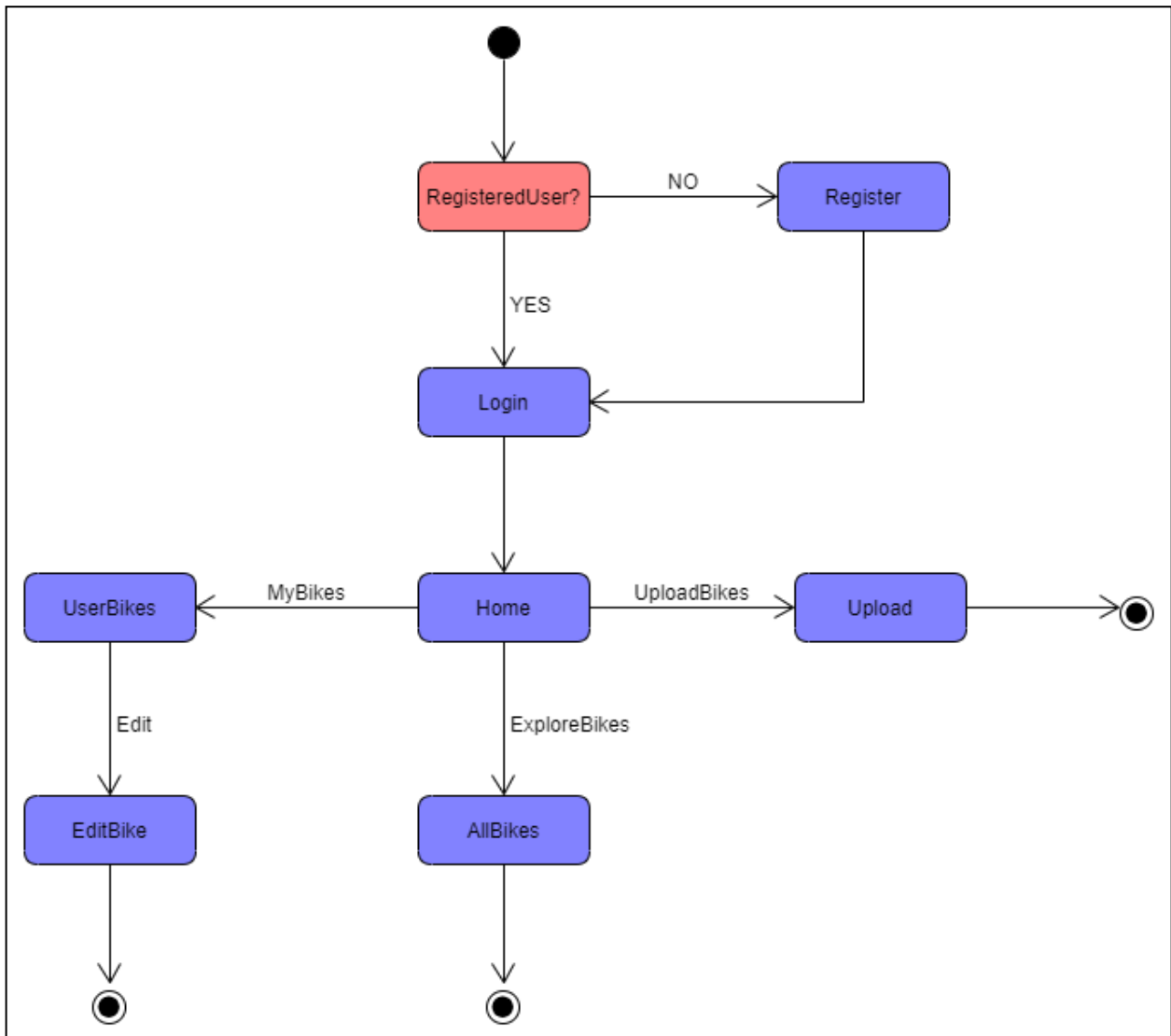
## 4.2 Class Diagram



## 4.3 Sequence Diagram

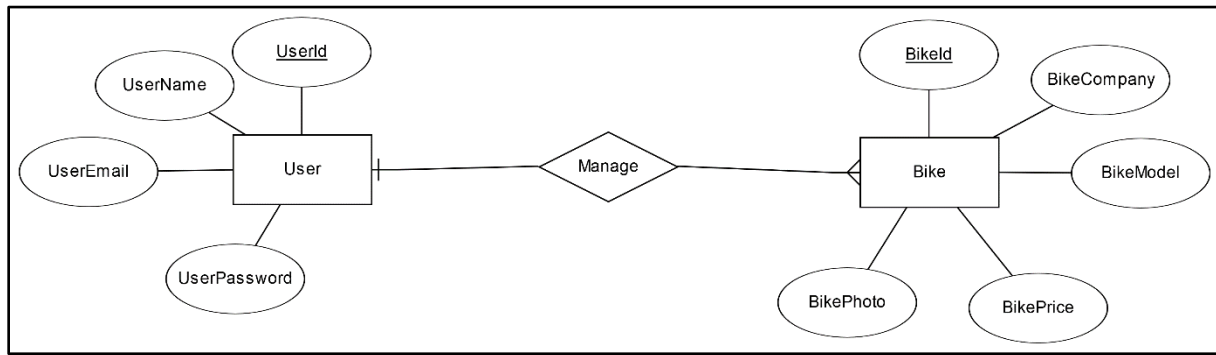


#### 4.4 Activity Diagram



*Activity Diagram*

## 4.5 E-R Diagram



## 4.6 Data Dictionary

User								
Sr. No	Field Name	Data Type	Width	Required	Unique	PK/FK	Referenced Table	Description
1	UserId	numeric	15	Yes	Yes	PK	-	Id of User
2	UserName	varchar	30	Yes	No	-	-	Name of User
3	UserEmail	varchar	30	Yes	Yes	-	-	Email ID of User
4	UserPassword	varchar	30	Yes	Yes	-	-	Password of User

Bike								
Sr. No	Field Name	Data Type	Width	Required	Unique	PK/FK	Reference Table	Description
1	BikeId	varchar	15	Yes	Yes	PK	-	Id of Bike
2	BikeModel	varchar	30	Yes	Yes	-	-	Model of Bike
3	BikeCompany	numeric	30	Yes	Yes	-	-	Company of Bike
4	BikePrice	varchar	30	Yes	Yes	-	-	Price of Bike
5	BikeDetails	varchar	150	No	Yes	-	-	Details of Bike
6	BikePhoto	byte	255	Yes	Yes	-	-	Photo of Bike

## 5. Implementation Details

---

- This all modules are implemented in WCF service.

### **Registration module:**

This module is used to store user's data to the database and enables the user to login to the system. All the fields in system module contains required validations. User can also navigate to login page if he/she has already registered

Input: User's Information

Output: User Registered and redirect to login page

Processing: Validating user's data and then storing them to database

**Login:**

This module takes user credentials and then verifies it with registered users, if user is not registered the invalid credentials is shown else if they match with database then login user.

Input: User Credentials

Output: Logging user.

Processing: Verify user credentials with the database.

**Bike Module:**

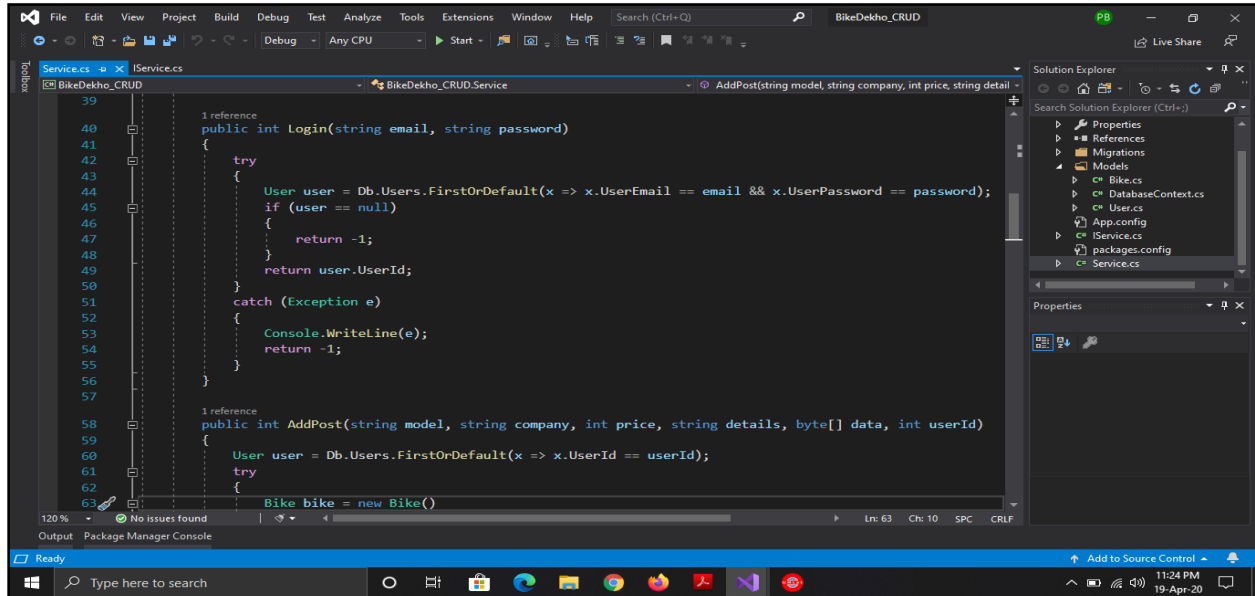
This Module is running on WCF service. It provides facility of uploading and updating of bike details (Model No, Company name, Price, Image, Description) that has uploaded by user before.

Input: Details of bike

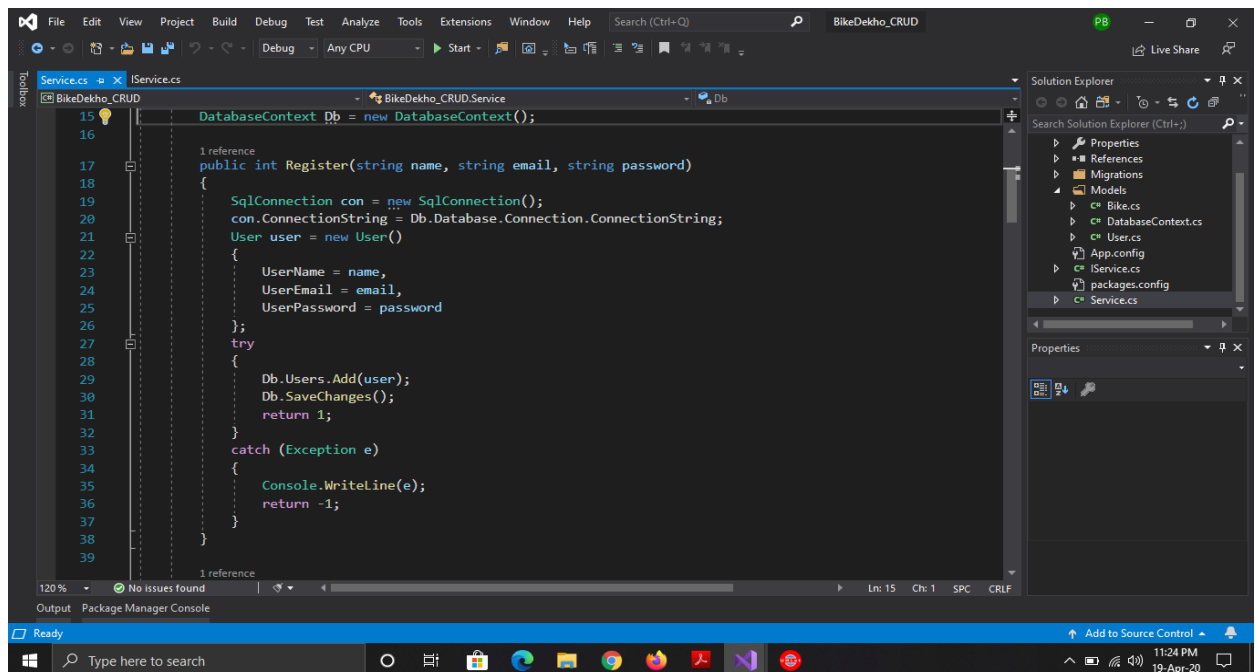
Output: Bikes will be shown in list.

## 5.2 Function Prototypes

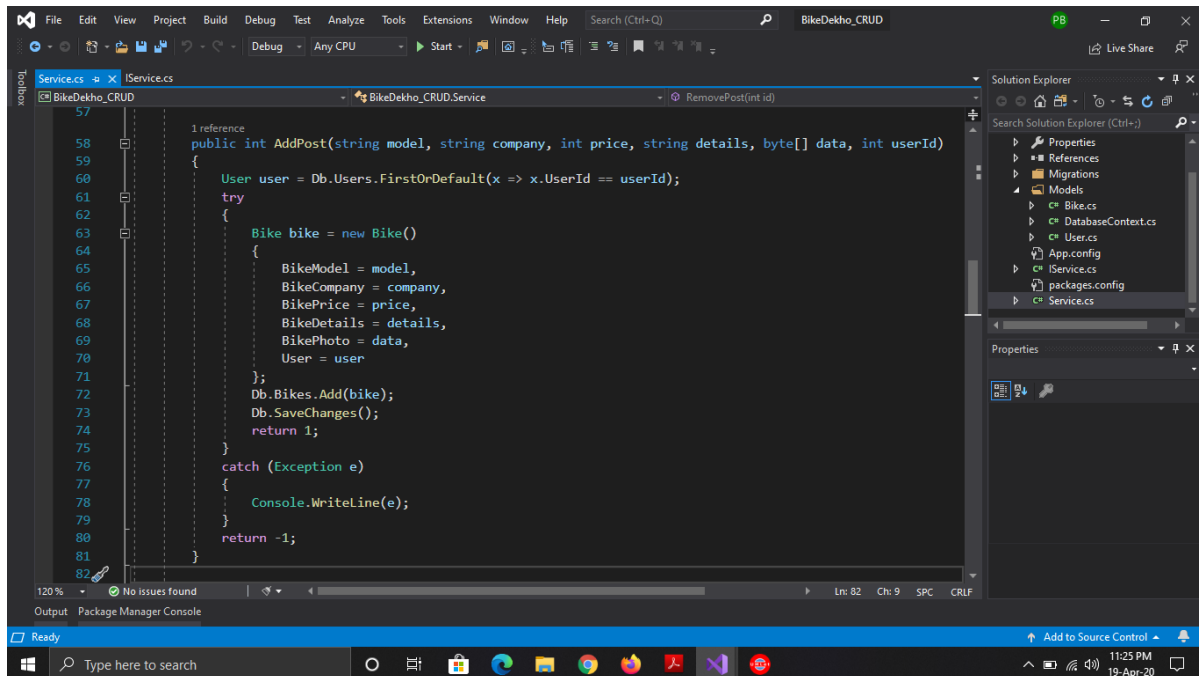
### Login



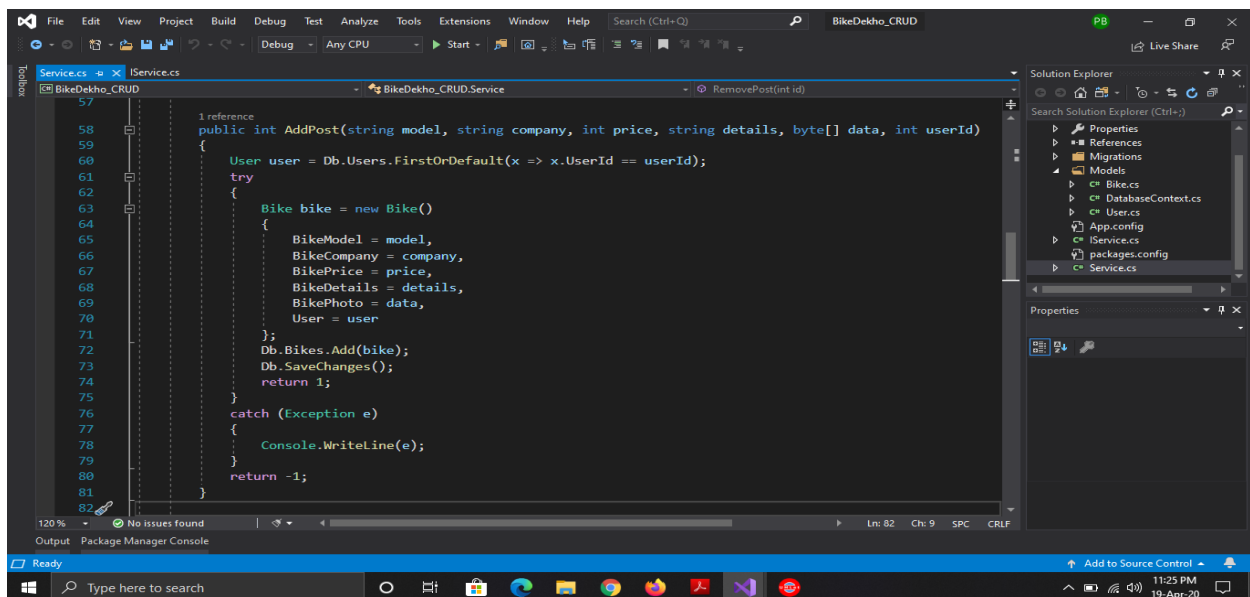
### Register:



## Add Bike:

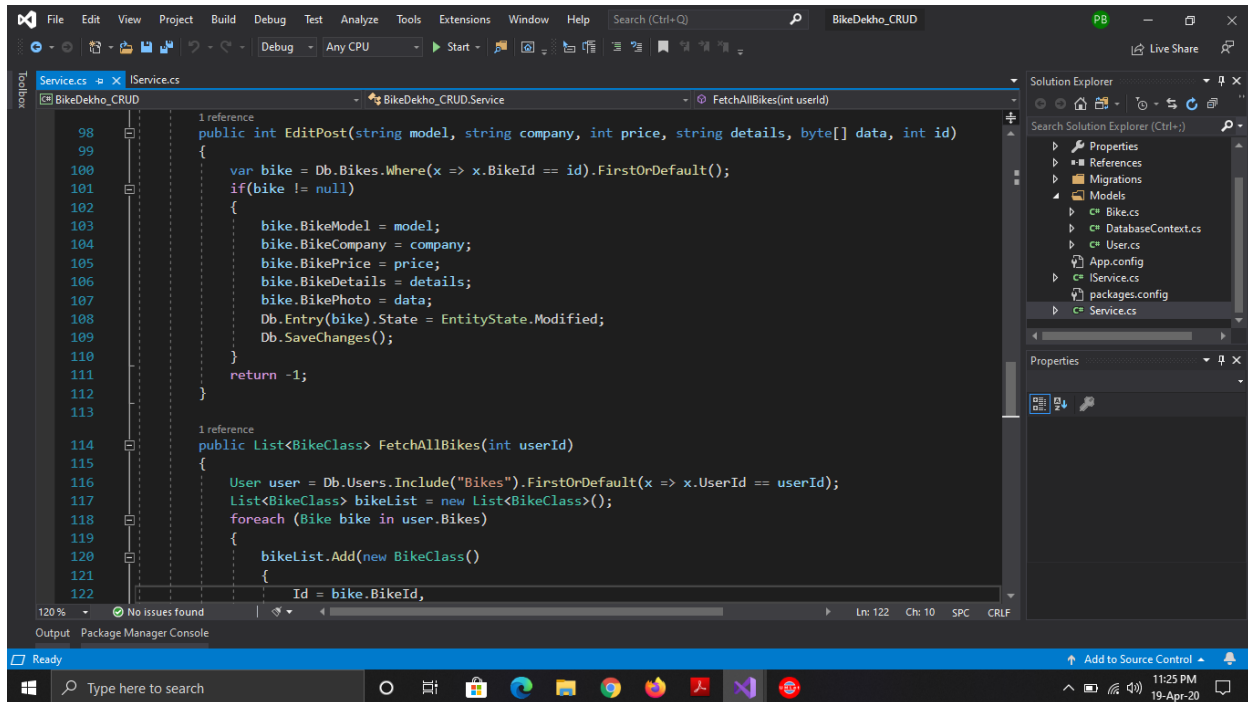


## Remove Bike:





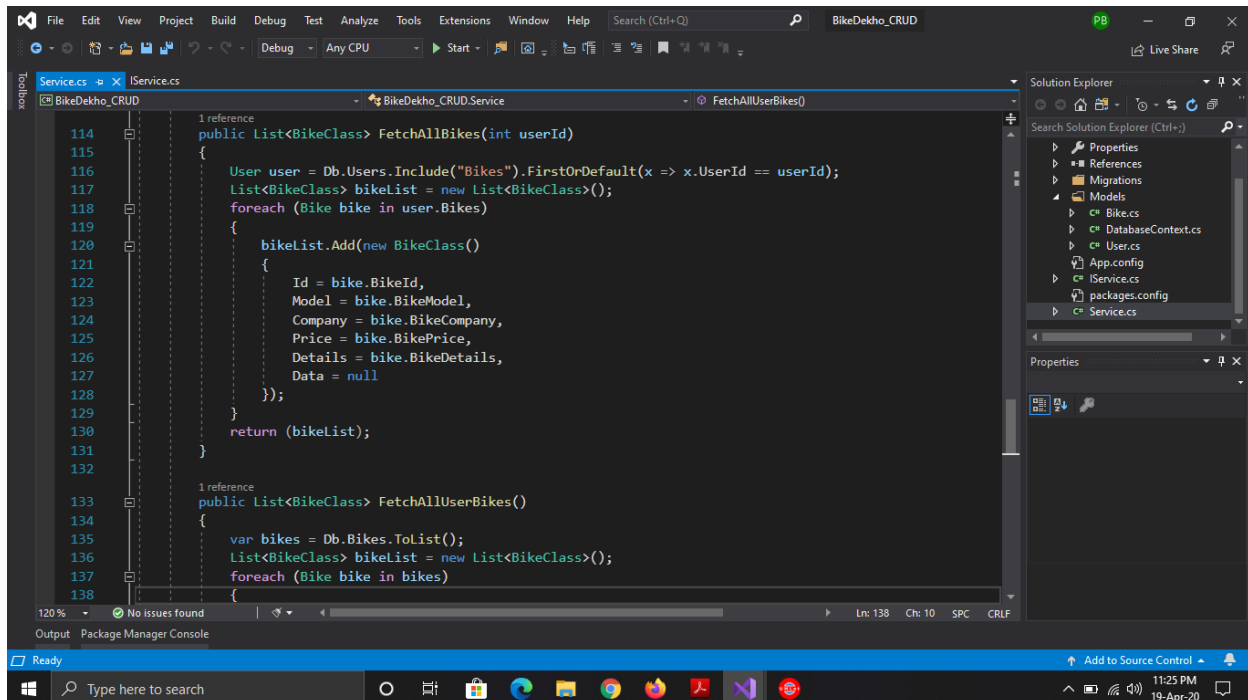
## Edit Bikes:



The screenshot shows the Visual Studio IDE with the file `Service.cs` open. The `EditPost` method is highlighted, which takes parameters for model, company, price, details, data, and id. It uses LINQ to find a bike by ID and updates its details. The `FetchAllBikes` method is also visible, which fetches bikes for a specific user. The Solution Explorer on the right shows the project structure, and the Properties window is empty.

```
1 reference
98 public int EditPost(string model, string company, int price, string details, byte[] data, int id)
99 {
100     var bike = Db.Bikes.Where(x => x.BikeId == id).FirstOrDefault();
101     if(bike != null)
102     {
103         bike.BikeModel = model;
104         bike.BikeCompany = company;
105         bike.BikePrice = price;
106         bike.BikeDetails = details;
107         bike.BikePhoto = data;
108         Db.Entry(bike).State = EntityState.Modified;
109         Db.SaveChanges();
110     }
111     return -1;
112 }
113
114 1 reference
115 public List<BikeClass> FetchAllBikes(int userId)
116 {
117     User user = Db.Users.Include("Bikes").FirstOrDefault(x => x.UserId == userId);
118     List<BikeClass> bikeList = new List<BikeClass>();
119     foreach (Bike bike in user.Bikes)
120     {
121         bikeList.Add(new BikeClass()
122         {
123             Id = bike.BikeId,
```

## Fetch bikes:



The screenshot shows the Visual Studio IDE with the file `Service.cs` open. The `FetchAllBikes` method is highlighted, which fetches bikes for a specific user. The `FetchAllUserBikes` method is also visible, which fetches all bikes for a specific user. The Solution Explorer on the right shows the project structure, and the Properties window is empty.

```
1 reference
114 public List<BikeClass> FetchAllBikes(int userId)
115 {
116     User user = Db.Users.Include("Bikes").FirstOrDefault(x => x.UserId == userId);
117     List<BikeClass> bikeList = new List<BikeClass>();
118     foreach (Bike bike in user.Bikes)
119     {
120         bikeList.Add(new BikeClass()
121         {
122             Id = bike.BikeId,
123             Model = bike.BikeModel,
124             Company = bike.BikeCompany,
125             Price = bike.BikePrice,
126             Details = bike.BikeDetails,
127             Data = null
128         });
129     }
130     return (bikeList);
131 }
132
133 1 reference
134 public List<BikeClass> FetchAllUserBikes()
135 {
136     var bikes = Db.Bikes.ToList();
137     List<BikeClass> bikeList = new List<BikeClass>();
138     foreach (Bike bike in bikes)
139     {
```

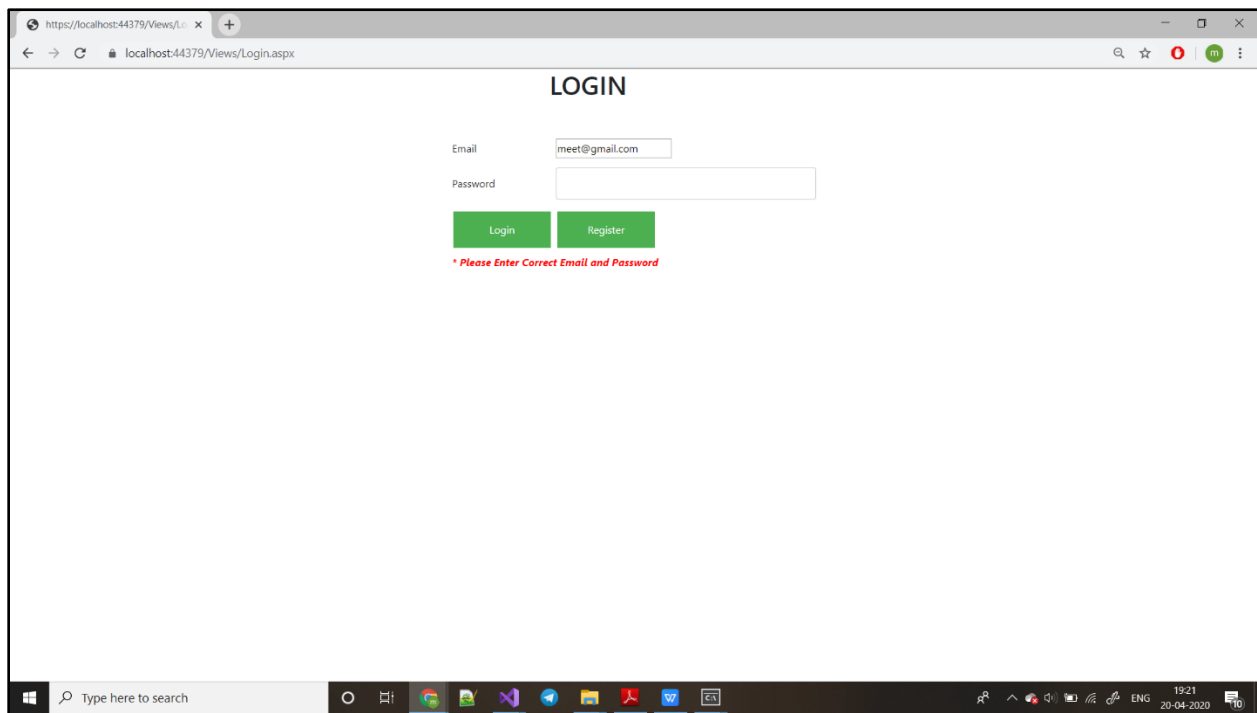
## 6. Testing

---

### 6.1: Testing Methods:

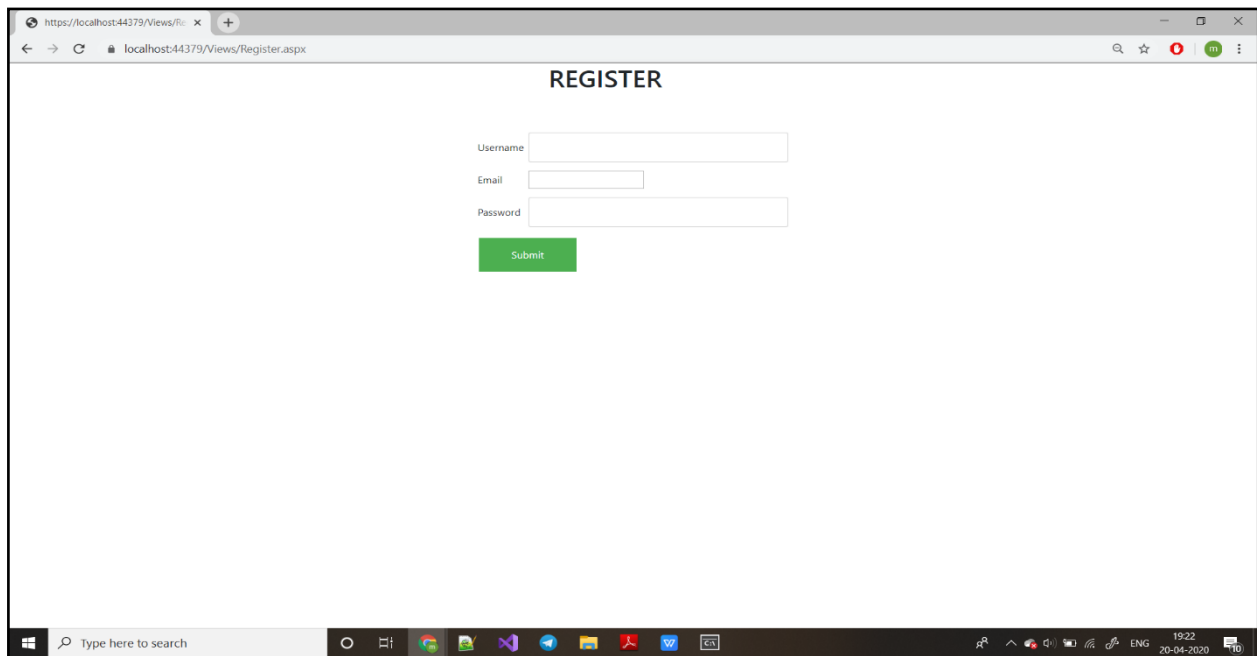
We have performed Black-box testing for the testing purpose.

### 6.2: Testing Demo:



### *Validation at Login Page*

## 7. Screenshots



A screenshot of a web browser displaying a registration form titled "REGISTER". The form is centered on a white background. It contains three input fields: "Username", "Email", and "Password", each with a corresponding label to its left. Below these fields is a green "Submit" button. The browser's address bar shows the URL "https://localhost:44379/Views/Register.aspx". The Windows taskbar is visible at the bottom, showing the search bar and various application icons.

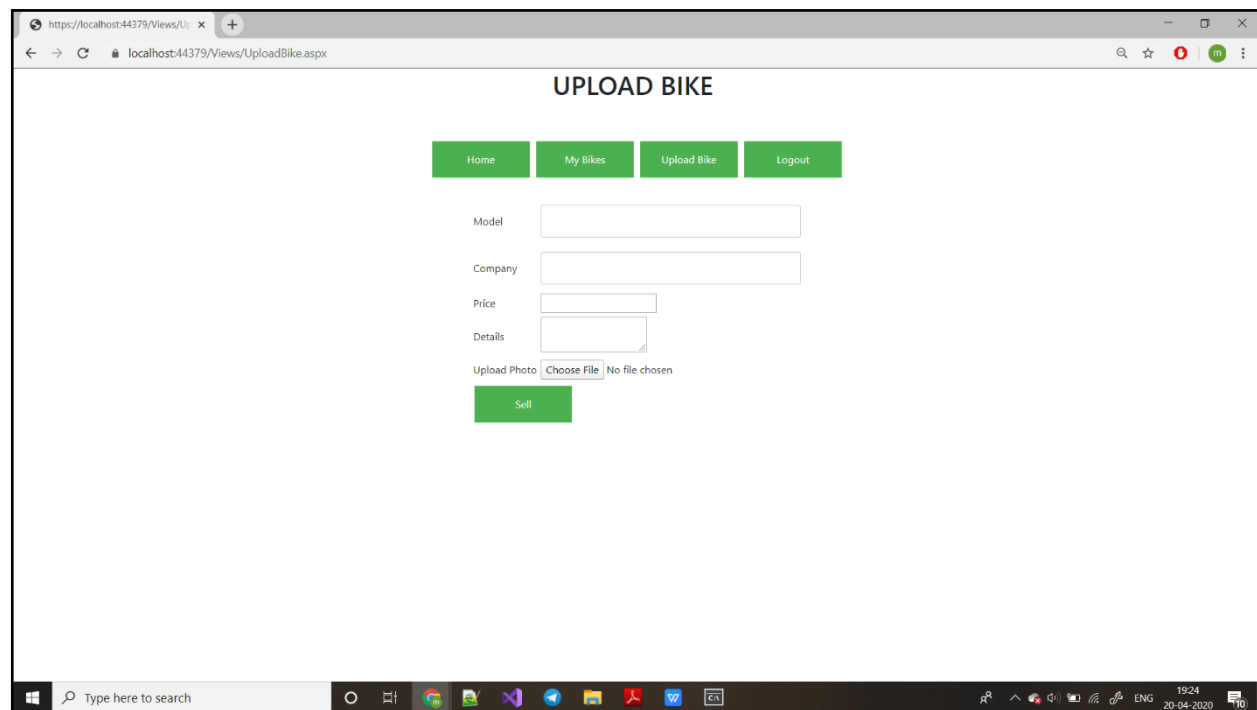
REGISTER

Username

Email

Password

**1.Register Page**



A screenshot of a web browser displaying a page titled "UPLOAD BIKE". At the top, there is a navigation bar with four green buttons: "Home", "My Bikes", "Upload Bike", and "Logout". Below the navigation bar, there are four input fields: "Model", "Company", "Price", and "Details", each with a corresponding label to its left. Below these fields is a section for "Upload Photo" with a "Choose File" button and the text "No file chosen". At the bottom of the form is a green "Sell" button. The browser's address bar shows the URL "https://localhost:44379/Views/UploadBike.aspx". The Windows taskbar is visible at the bottom, showing the search bar and various application icons.

UPLOAD BIKE

Model

Company

Price

Details

Upload Photo  No file chosen

**2.Upload Bike Page**

### 3.EditBike Page

EDIT YOUR BIKE DETAILS

Model

Company

Price

Details

Upload Photo  No file chosen

### 4.YourBike Page

Home My Bikes Upload Bike Logout

YOUR BIKES

Model	Company	Price	Details	Operation
40	yamaha	999999	sexy	<a href="#">View Photo</a>   <a href="#">Edit</a>   <a href="#">Remove</a>

## 8. Conclusion

---

The functionalities implemented in system after understanding all the system modules according to the requirements. Functionalities That are successfully implemented in the system are:

- User registration containing all the necessary validation on field
- Login
- User authentication
- Logout
- Bike Upload
- Bike Download
- Bike Remove

After the implementation and coding of system comprehensive testing was performed on the system to determine the errors and possible flaws in the system.

## 9. Limitations and Future Enhancements

---

The System has adequate scope for modification in future if it is necessary.

Development and launching of Mobile app and refining existing services and adding more service, System security, data security and reliability are the main feature.

In the existing system user will be able to chat also. So, in Future scope this will be resolve.

## 10. Reference / Bibliography

---

Following links and websites were referred during the development of this project.

<https://www.google.co.in>

<https://www.olx.in>

<http://www.getbootstrap.com>

<http://www.tutorialspoint.com>

<http://www.stackoverflow.com>