Software Requirements Specification

For
Web Parser System

Eman Hamdy (Team Leader)

Yassmin Elnageh

Aml Hamdy

Nourhan AbdelMageed

Lmiaa Hamdy

Table of Contents

1. Introduction

1.1. Purpose

The purpose of this Software Requirements Documentation is to provide high-level and detailed descriptions of the Real Estate Web Site and to parse all available web sites working on cars. This Software Requirements Documentation will provide quantifiable requirements of the web site for use by the designer and the users of the Real Estate Web Site.

1.2. Scope

The Web Parser Web Site shall provide my client the ability to showcase and manage their property listings while allowing potential clients a concise and structured way of searching properties. The web site will be set up in a way the Admin will have total Administrative rights to their listings.

1.3. Overview

The main purpose of this site is to serve as a web presence for the realtor's properties. The realtor will serve as the administrator for this site with the ability to create, modify, or delete admin or users. Once all administrative tasks are in place, potential clients can interact with the site via Twitter or Facebook. Also, detailed and specific search capabilities will be available.

Overall Description

2.1 Product perspective

This system will be a web application to get data about cars and real-estate from other websites and show in a friendly way and user can search for cars or real-estate and also can search by the range of price, type, region, government and country. User can share products on social media.

Admin can see users and modify, delete and ban user data and see statistics and analysis for user's search, also see suggestion from users of website.

2.2 Product functions

1. Every user can register in the website.
2. Every user can login in the website.
3. Get data from other websites.
4. User can search for product by some criteria.
5. User can share product or more on social media.
6. User has a profile contains user information and can modify it.
7. Visitors can suggest any thing about the product.
8. Admin can see user information.
9. See statistics of users and search result.

2.3 User characteristics

There are two types of users use application: users of the website and administrators.

Users can see data that come from other websites, search about specific products and see their profile to modify if want.

Admin can modify, delete and ban users also see statistics about site and suggestions from visitors.

2.4 Constraints

The Internet connection is a constraint for the application.

2.5 Assumptions and dependencies

Software:
One assumption about the product is that it will always be responsive for all devices.

Internet:

The end user must have Internet access. High - speed Internet access is preferred but not required.

2.6 Apportioning of requirements

In the case that the project is delayed, there are some requirements that could be canceled that will depend on the opinion of the client.

Specific Requirements

3.1 Functional Requirements

1. Every user can register in the website.
   Information provided by the user in the registration process is inserted into the database.
2. Every user can login in the website.
   Information provided by the user in the login process is verified from the database and retrieve the latest search result and user can login using social media.

3. Get data from other websites.
   Can get updated data from other websites and store URLs in database and view this data about cars and real estate in friendly way to users.
4. User can search for product by some criteria.
   User can search by some criteria such as: minimum price, maximum price, type of product, region, government, country, text and also can search by one or more criteria.
5. User can share product or more on social media.
   User can share product details and photo in his profile in social media.
6. User has profile contain user information and can modify it.
   Every user has profile that contain his information that registered in database and can modify or add any missing information and change his/her image.
7. Visitors can suggest any thing about product.
   Every visitor can say his/her opinion about the website and other suggestion. This visitor may be registered or not registered and can rate the website.
8. Admin can see user information.
   Admin for website can see all information of registered users, modify, delete and ban users.
9. Admin can see statistics of users and search result.
   System can generate statistics related to a specific user, a specific URL or users and URLs together.

3.2 Quality Requirements

Reliability:

- The web-server should always be available no matter what time a user attempts to access the website.
- The database should maintain data normalization by implementing a primary and foreign key system so that discrepancies do not occur within the data.

Efficiency:

- Search queries or page loads should be returned and formatted by the ASP.net parser in a timely fashion depending on the request being made.

Security:

- Administrators can only perform administrative tasks on pages they are privileged to access. Clients will not be allowed to access the administrators' areas**.**
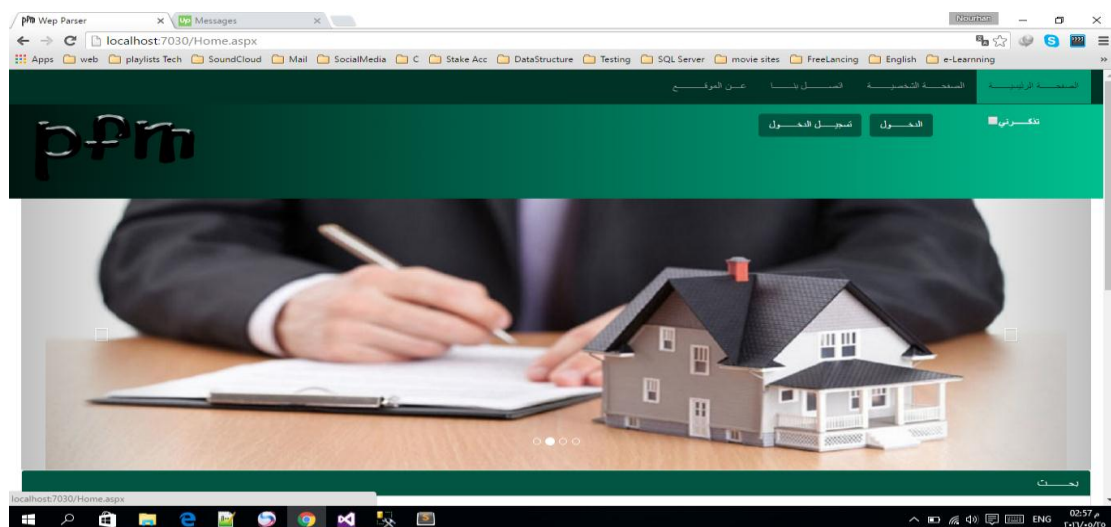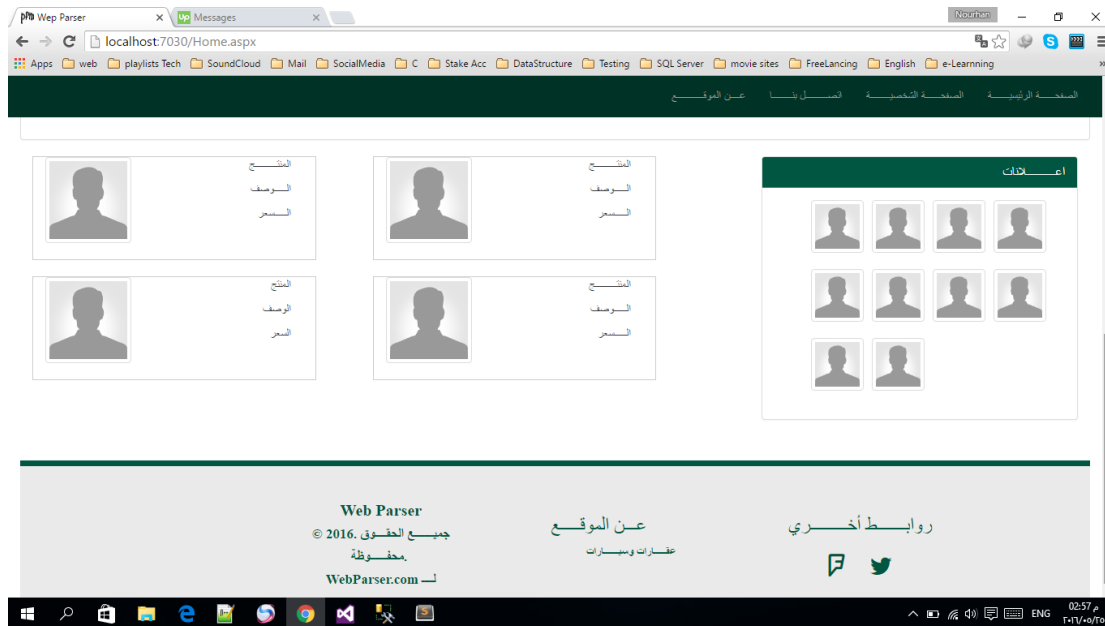
3.3. Tools

- MS SQL
- ASP.net

3.4. User Interface Requirements

Home:

User should be able to access and change user account information as well as logout with ease.

Register:

New Users must have easy access to registration links and be able to place Information upon first login.
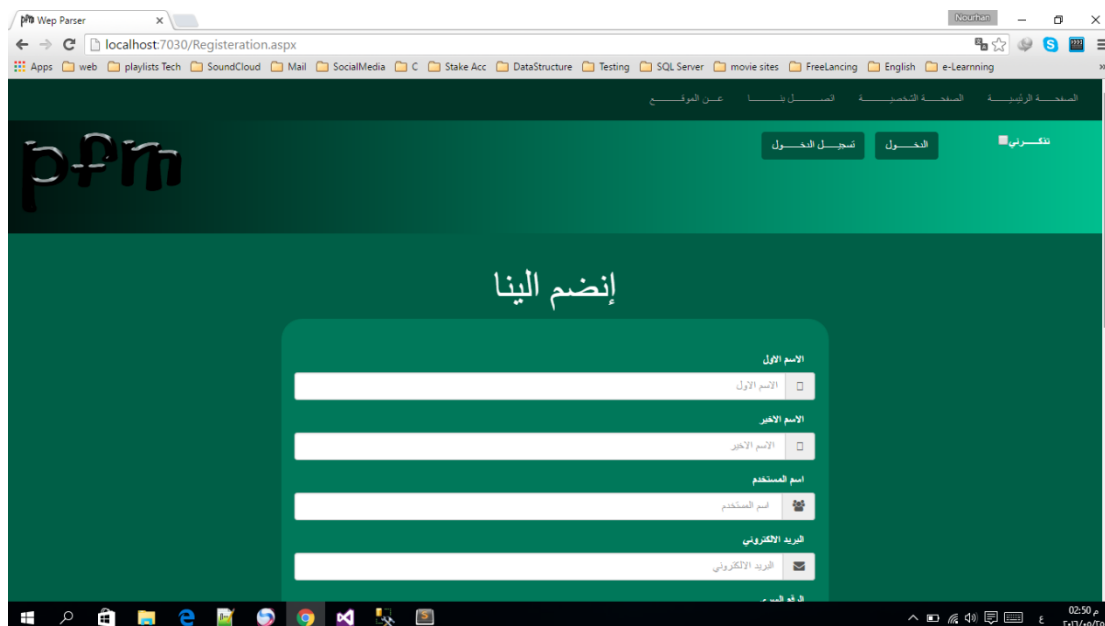
Login:

Users and admins should be able to login to their account without having to scroll on the web page. User should be able to login within minutes of registering.
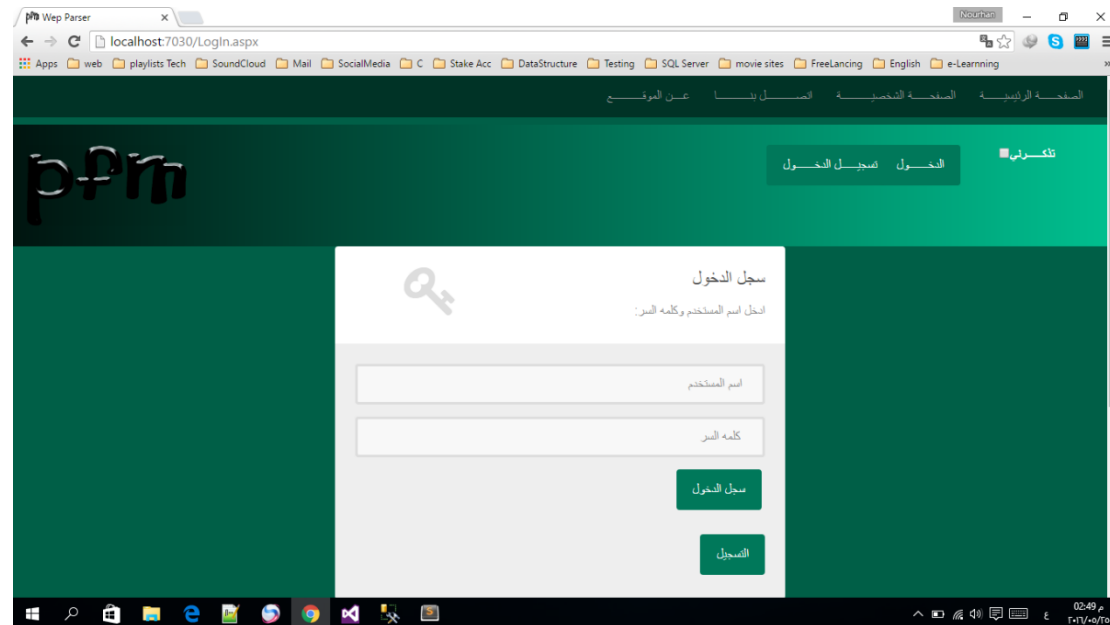
Password Reminder:

Users/Admins should be able to retrieve access info via e-mail. Process should be quick and simple.

View Products:

The user must be able to view a listing Products contained within a property listing in less than one minute once they have accessed the web site.

Administrator:

The administrator must be able to add a URL in less than five minutes once they have logged into the web site. The administrator must be able to remove a URL in less than one minute once they have logged into the web site.

3.5. Logical Database Requirements

Users:
This table will store the accounts of registered users.
The Users table will be created based on information such as username,password,government,region,country,firstname,lastname,mobile,birthdate,picture,gender and ID. Once a user provides an email or username the website will ensure that they are registered before the user is allowed to login. Each user will have an e-mail, which should ideally be their actual name, perhaps split into first and last names. The user will also have a password. The user will also be required to enter a valid email or username.

	Column Name	Data Type	Allow Nulls
🔑	ID	int	☐
	FirstName	nvarchar(50)	☐
	LastName	nvarchar(50)	☐
	UserName	nvarchar(50)	☐
	Password	char(10)	☐
	Picture	varbinary(MAX)	☑
	Email	nvarchar(100)	☐
	Gender	char(6)	☐
	BirthDate	date	☐
	Mobile	char(11)	☐
	RegionID	int	☐

Admins:

This table will store admins information.
The Users table will be created based on information such as ID, First name, last name and email.

Once an admin provides an email the web site will ensure that they are registered before the admin is allowed to login. Each admin will have an e-mail, which should ideally be their actual name, perhaps split into first and last names. The admin will also have a password. The admin will also be required to enter a valid email address.

Column Name	Data Type	Allow Nulls
UserName	nvarchar(50)	☐
Password	nvarchar(50)	☐
ID	int	☐
FName	nvarchar(50)	☐
LName	nvarchar(50)	☐
		☐

Contact Info:

This table will store the contact information of registered or unregistered users. The Users table will be created based on information such as ID, Name, Subject, details, mobile, email and time.

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	nvarchar(50)	☐
Subject	nvarchar(50)	☑
Details	nvarchar(MAX)	☐
Mobile	char(11)	☑
Email	nvarchar(100)	☐
Time	datetime	☐
		☐

Country:

This table will store all country information.

The country table will be created based on information such as ID and name

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	nvarchar(50)	☐

Governorate:

This table will store all Governorates that belong to countries in country table.

The Governorate table will be created based on information such as ID and name.

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	nvarchar(50)	☐
CountryID	int	☐

Region:

This table will store all regions that belong to Governorate in Governorate table. The region table will be created based on information such as ID and name.

Column Name	Data Type	Allow Nulls
ID	int	☐
Name	nvarchar(50)	☐
Governorate ID	int	☐

Source:

This table will store all sources that will be parsing its data.

The source table will be created based on information such as ID and URL.

Column Name	Data Type	Allow Nulls
ID	int	☐
URL	nvarchar(MAX)	☐

Search criteria:

This table will store all criteria that user can select to search depending on it.

The Search criteria table will be created based on information such as ID and criteria.

Column Name	Data Type	Allow Nulls
ID	int	☐
Crieteria	nvarchar(50)	☐
		☐

Search history:

This table will store all search history that users make.

The search criteria table will be created based on information such as ID, User ID and date.

Column Name	Data Type	Allow Nulls
ID	int	☐
UserID	int	☐
Date	datetime	☐
		☐

Search history source:

This table will store all Search history source that user search come from.

The Search history source table will be created based on information such as searchhistoryID, source ID and number.

Column Name	Data Type	Allow Nulls
SearchHistoryID	int	☐
SourceID	int	☐
Number	int	☐
		☐

Search value:

This table will store all Search value of criteria that user search with.

The Search value table will be created based on information such as searchhistoryID, searchcrieteriaID and value.

Column Name	Data Type	Allow Nulls
SearchHistoryID	int	☐
SearchCrieteriaID	int	☐
Value	nvarchar(50)	☐
		☐

ERD:

Class Diagram:

User
```
-ID: int {ordered, unique}
-FName: string
-LName: string
-UserName: string
-Password: string
-Picture: bytes
-Email: string {unique}
-Gender: string
-BirthDate: string
-Mobile: string
```
```
+Register(FName: string, LName: string, UserName: string, Password: string, Picture: bytes, Email: string, Gender: string, BirthDate: Date, Mobile: string): bool
+Login(UserName: string, Password: string): bool
+Logout(UserName: string): bool
+Search(Keys: list<string>): void
+Suggest(Name: string, Subject: string, Details: string, Mobile: string, Email: string, Time: string): void
+Share(URL: string): void
+getFName(): string
+set(FName: string): void
+getLName(): string
+set(LName: string): void
+set(Password: string): void
+getPassword(): string
+getLUserName(): string
+set(UserName: string): void
+getPicture(): bytes
+set(Picture: bytes): void
+getEmail(): string
+set(Email: string): void
+getGender(): string
+set(Gender: string): void
+getBirthDate(): string
+set(BirthDate: string): void
+getMobile(): string
+set(Mobile: string): void
+getID(): int
+set(ID: int): void
```
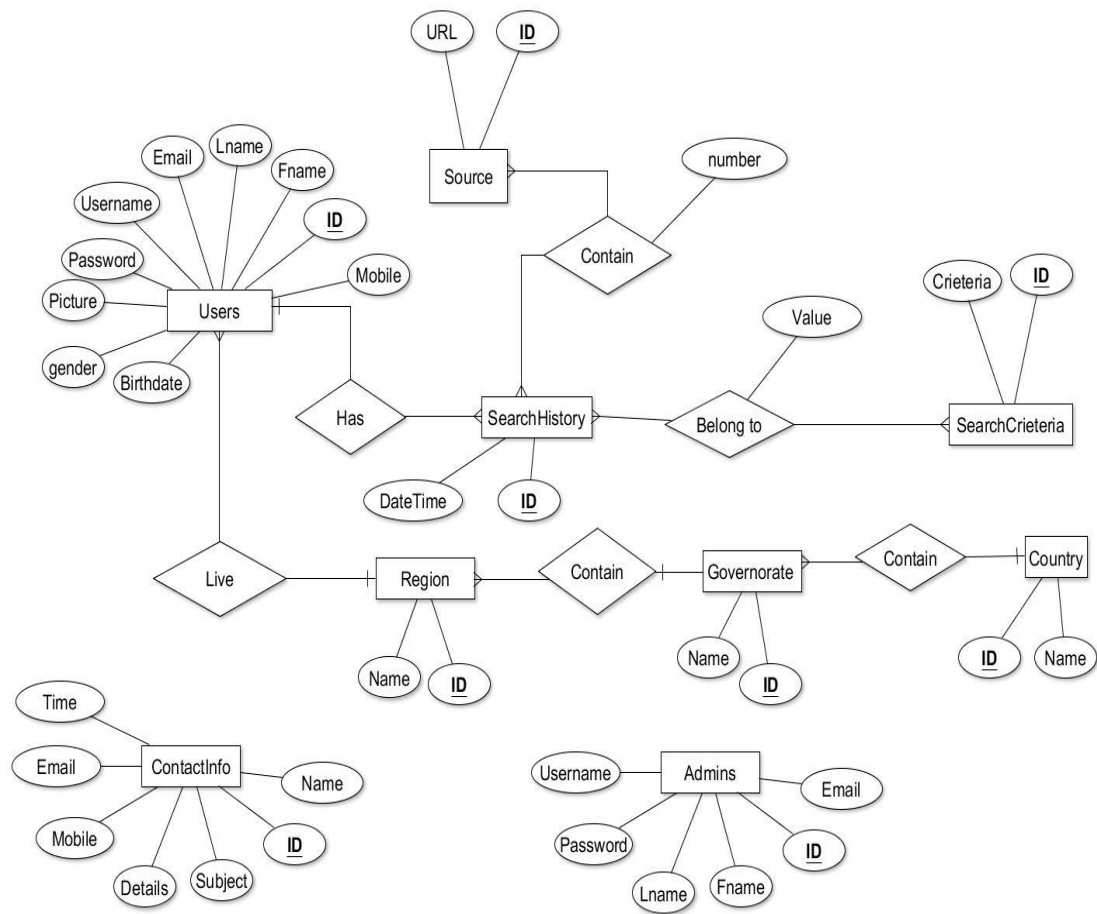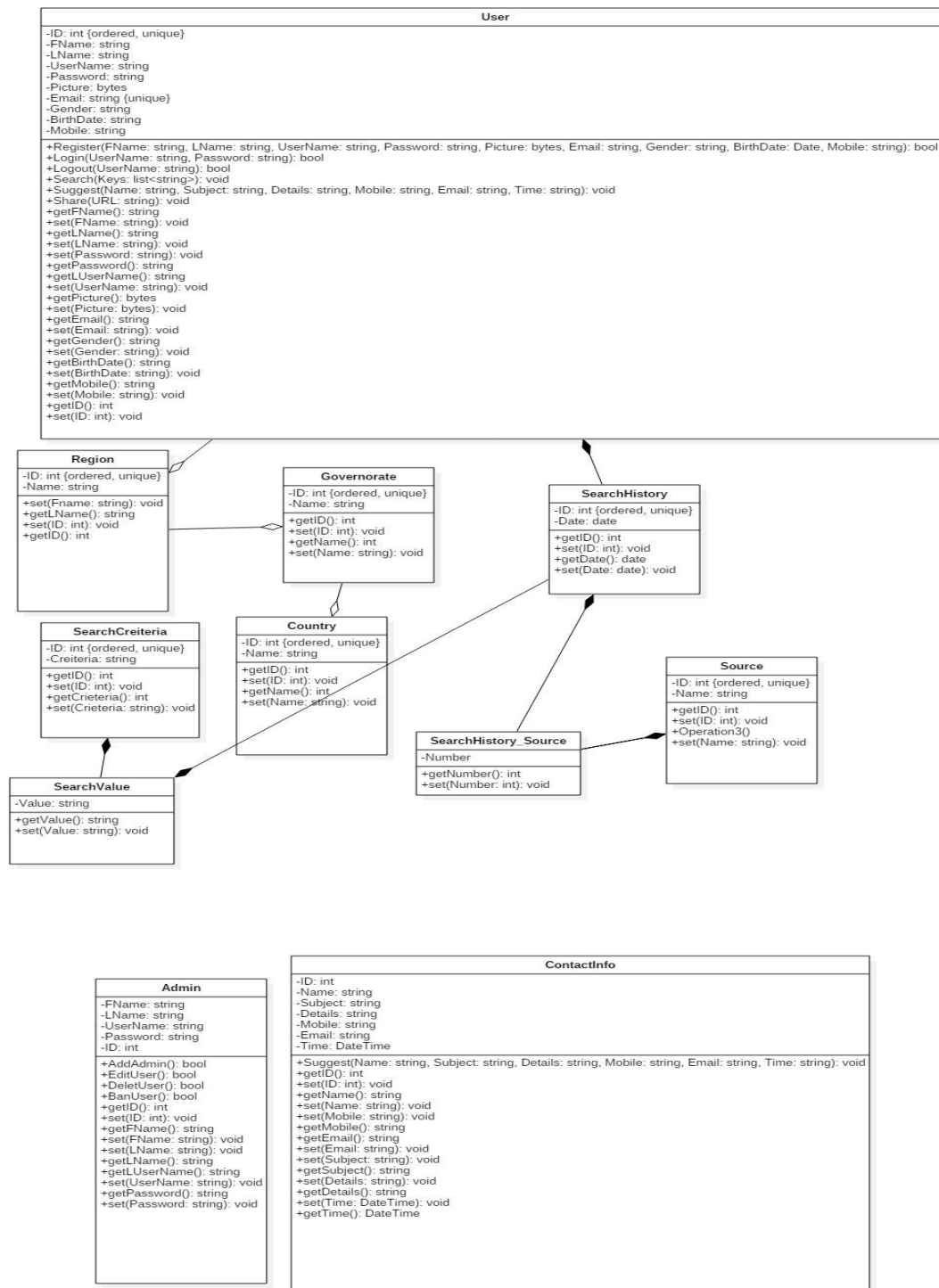
Region
```
-ID: int {ordered, unique}
-Name: string
```
```
+set(Fname: string): void
+getLName(): string
+set(ID: int): void
+getID(): int
```

Governorate
```
-ID: int {ordered, unique}
-Name: string
```
```
+getID(): int
+set(ID: int): void
+getName(): int
+set(Name: string): void
```

SearchHistory
```
-ID: int {ordered, unique}
-Date: date
```
```
+getID(): int
+set(ID: int): void
+getDate(): date
+set(Date: date): void
```

SearchCreiteria
```
-ID: int {ordered, unique}
-Creiteria: string
```
```
+getID(): int
+set(ID: int): void
+getCrieteria(): int
+set(Crieteria: string): void
```

Country
```
-ID: int {ordered, unique}
-Name: string
```
```
+getID(): int
+set(ID: int): void
+getName(): int
+set(Name: string): void
```

Source
```
-ID: int {ordered, unique}
-Name: string
```
```
+getID(): int
+set(ID: int): void
+Operation3()
+set(Name: string): void
```

SearchHistory_Source
```
-Number
```
```
+getNumber(): int
+set(Number: int): void
```

SearchValue
```
-Value: string
```
```
+getValue(): string
+set(Value: string): void
```

Admin
```
-FName: string
-LName: string
-UserName: string
-Password: string
-ID: int
```
```
+AddAdmin(): bool
+EditUser(): bool
+DeletUser(): bool
+BanUser(): bool
+getID(): int
+set(ID: int): void
+getFName(): string
+set(FName: string): void
+set(LName: string): void
+getLName(): string
+getLUserName(): string
+set(UserName: string): void
+getPassword(): string
+set(Password: string): void
```

ContactInfo
```
-ID: int
-Name: string
-Subject: string
-Details: string
-Mobile: string
-Email: string
-Time: DateTime
```
```
+Suggest(Name: string, Subject: string, Details: string, Mobile: string, Email: string, Time: string): void
+getID(): int
+set(ID: int): void
+getName(): string
+set(Name: string): void
+set(Mobile: string): void
+getMobile(): string
+getEmail(): string
+set(Email: string): void
+set(Subject: string): void
+getSubject(): string
+set(Details: string): void
+getDetails(): string
+set(Time: DateTime): void
+getTime(): DateTime
```
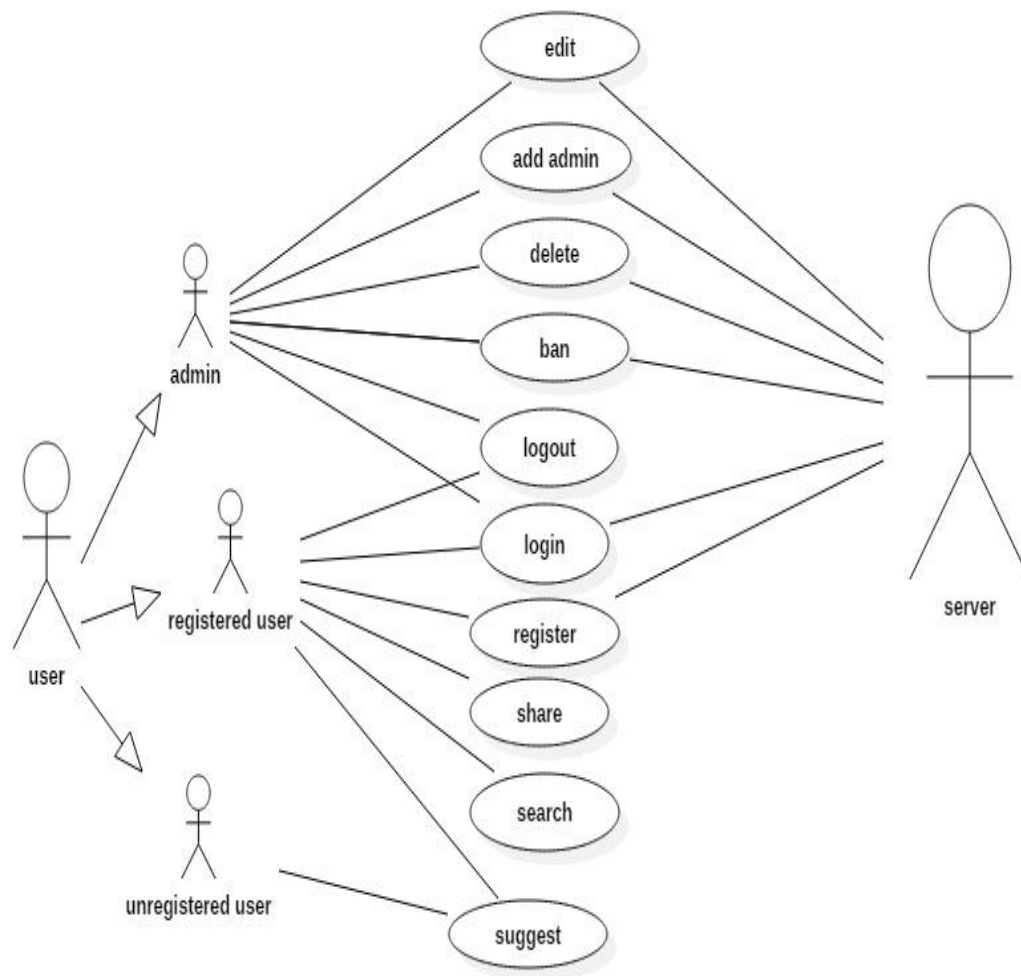
Use case:

User can register and login to make search about cars and real estate.

Admin can login to show, ban, edit and delete users.

And also can delete or add new Admin.

Activity Diagram