

**A PROJECT REPORT ON**  
**SMART CONTACT MANAGER**

**By**

**Jwalit Shah (CE121) (18CEUOS057)**  
**Mahammadayan Shaikh (CE124) (18CEUOG004)**

**B. Tech CE Semester - VI**  
**Subject : Service Oriented Computing**

**Guided By:**

**Prof. Apruva A. Mehta**  
**Assistant Professor**  
**Dept. Of Computer**  
**Engineering**

**Prof. Prashant M. Jadav**  
**Associate Professor**  
**Dept. Of Computer**  
**Engineering**



**Faculty of Technology**  
**Department of Computer Engineering**  
**Dharmsinh Desai University**



**Faculty of Technology  
Department of Computer Engineering  
Dharmsinh Desai University**

**CERTIFICATE**

This is to certify that the practical / term work carried out in the subject of  
**Service Oriented Computing** and recorded in this journal is the  
bonafide work of

**Jwalit Shah (CE121) (18CEUOS057)  
Mahammadayan Shaikh (CE124) (18CEUOG004)**

of B.Tech semester **VI** in the branch of **Computer Engineering** during the  
academic year **2020-2021**.

Prof. Apurva A. Mehta  
Assistant Professor,  
Dept. of Computer  
Engg.,  
Faculty of Technology,  
Dharmsinh Desai  
University, Nadiad

Prof. Prashant M. Jadav  
Associate Professor,  
Dept. of Computer  
Engg.,  
Faculty of Technology,  
Dharmsinh Desai  
University, Nadiad

Dr. C. K. Bhensdadia,  
Head,  
Dept. of Computer  
Engg.,  
Faculty of Technology,  
Dharmsinh Desai  
University, Nadiad

## Table Of Contents

<b>Abstract</b>	4
Purpose	4
Scope	4
<b>Introduction</b>	5
Brief Introduction	5
Technology/Platform/Tools used	5
<b>Software Requirements Specification - SRS</b>	6
Functional Requirements	6
1 Account module	6
2 Contact module	7
3 Group module	9
<b>Design</b>	11
Class Diagram	11
State Diagram	12
Use case Diagram	13
Sequence Diagram	14
Activity Diagram	15
ER Diagram	16
Data Dictionary	16
<b>Implementation Detail</b>	18
Modules	18
Function prototypes which implements major functionality	18
<b>Testing</b>	20
<b>Screenshots</b>	22
<b>Conclusion</b>	27
<b>Limitation and Future Extensions</b>	28
Limitations	28
Functionalities not implemented	28
Possible future extensions	28
<b>Bibliography</b>	29

## **Abstract**

This Smart Contact Manager System is a web application that provides the platform for users to manage their contacts individually or group wise. Users can create and manage contacts. Users can create and manage groups by adding or removing contacts to the group.

## **Purpose**

The purpose of this document is to collect and present ideas, requirements & analysis done in order to develop this system. This document provides a detailed overview of the system, details of the functionalities provided to users, target audience of the system and their user interfaces.

## **Scope**

The scope of the system is providing a handy web application to users for contact management. Any user with a registered account on the system can use the functionalities & get benefits of the provided features.

## **Introduction**

### **- Brief Introduction**

The Smart Contact Manager is a web application for users to minimize their day to day contact management. The system works around only on End Users. Users can Register and login to the system. Users can create, update, view and delete the contacts. Users can create, update and delete the Groups. Users can manage the groups by adding or removing their contacts from contact list to the group contact list. System checks all the corner cases for authentication, authorization of the users, validation of input data, duplication of data and gives appropriate error response messages according to the situation.

### **- Technology/Platform/Tools used**

#### **Technology :**

- WCF framework.
- Dot net core WEB API framework
- ASP.NET framework
- SQL Server database.
- Javascript
- Bootstrap

#### **Platform :**

- Windows

#### **Tools :**

- Visual Studio

## **Software Requirements Specification - SRS**

Total 1 type of users are there in system:

1. User

## **Functional Requirements**

### **R.1 Account module**

#### **R.1.1 Login**

Description: User can login to the system

Exception Flow: If credentials are incorrect or insufficient data is provided

Input: User data

Output: Redirects to Dashboard

#### **R.1.2 Register**

Description: User can login to the system

Exception flow: If username is already taken or insufficient data is provided

Input: User data

Output: Redirects to Login page with success message

#### **R.1.3 Update profile**

Exception flow: If username is already taken or insufficient data is provided

Input: User input

Output: Success message

#### **R.1.4 Forgot Password**

Input: username, old password, new password

Output: Response message

#### **R.1.5 Logout**

Input: User selection

Output: Redirects to Login page with success message

### **R.2 Contact module**

#### **R.2.1 Create contact**

Exception flow: If contact number is already taken or insufficient data is provided

Input: Contact data

Output: Success message

#### **R.2.2 View contact**

Exception flow: 404 If contact not found or Access denied when data is not allowed to access

Input: User selection

Output: Contact details

### **R.2.3 Update contact**

Exception flow: If new contact number is already taken or insufficient data is provided

Input: Contact data

Output: Success message

### **R.2.4 Delete contact**

Input: User selection

Output: Success message

### **R.2.5 View Recently added contacts**

Description: Last 3 recently added contact list can be viewed

Input: User selection

Output: Contacts list

### **R.2.6 Import contacts**

Description: Contacts can be imported in by .vcf file

Input: .vcf file

Output: Success message

### **R.2.7 Export contacts**



Description: Contacts can be exported in .vcf file

Input: User selection

Output: .vcf file

## **R.3 Group module**

### **R.3.1 Create group**

Exception flow: If group name is already taken or insufficient data is provided

Input: Group data

Output: Success message

### **R.3.2 View group**

Exception flow: 404 If group is not found or Access denied when data is not allowed to access

Input: User selection

Output: Group details

### **R.3.3 Update group**

Exception flow: If new group name is already taken or insufficient data is provided

Input: Group data

Output: Success message

#### **R.3.4 Delete group**

Input: User selection

Output: Success message

#### **R.3.5 View Recent created groups**

Description: Last 3 recently created groups list can be viewed

Input: User selection

Output: Groups list

#### **R.3.6 Add contacts to the group**

Description: User can add multiple contacts to the specific group

Input: User selection of contacts

Output: Success message

#### **R.3.7 Remove contacts from the group**

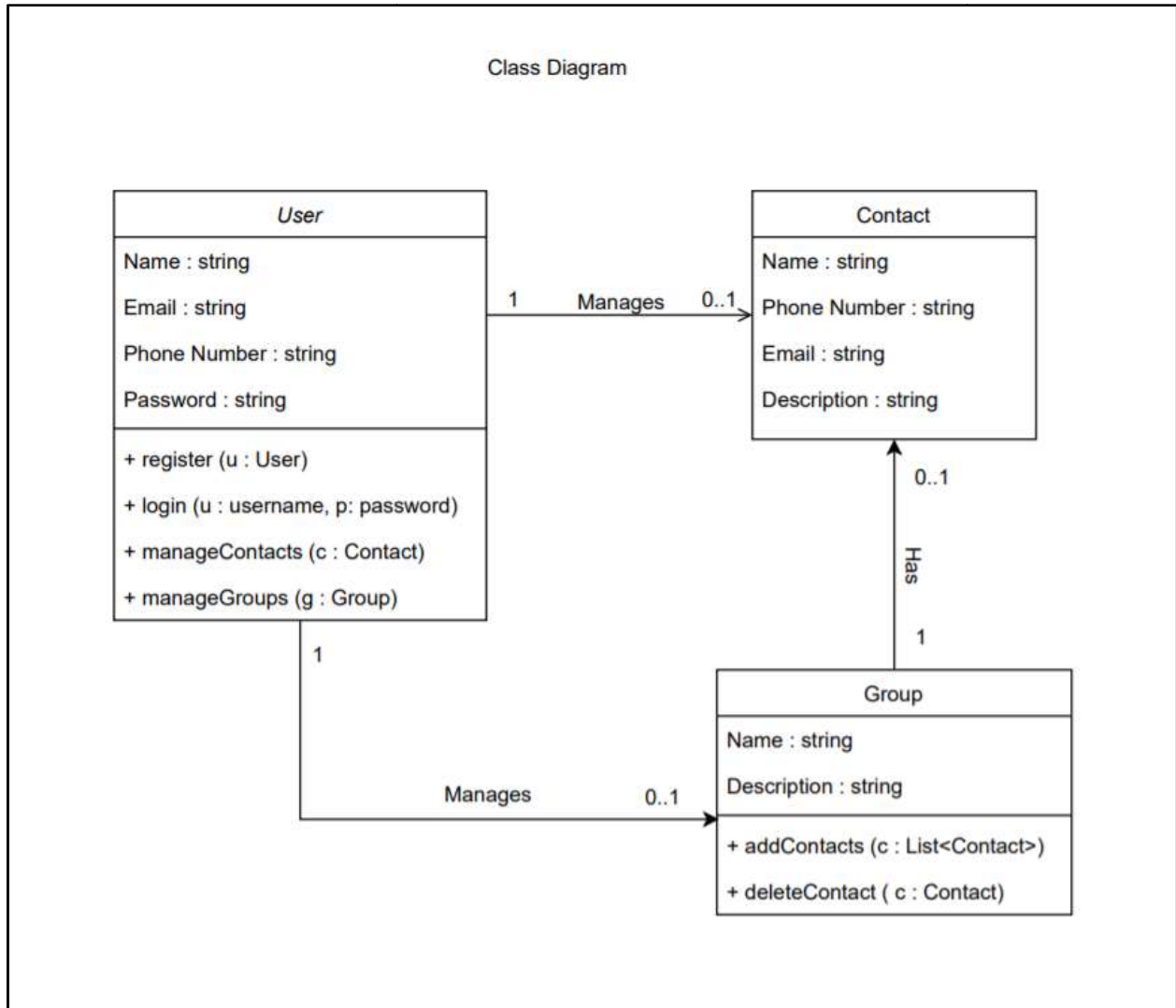
Description: User can remove contact from the specific group's contact list

Input: User selection

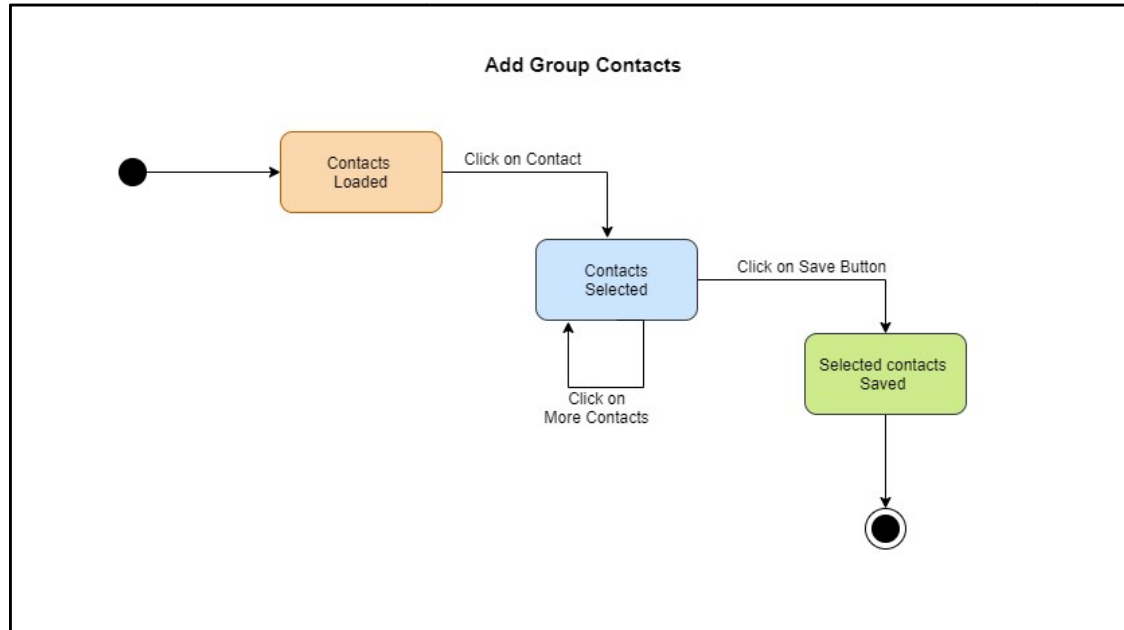
Output: Success message

## Design

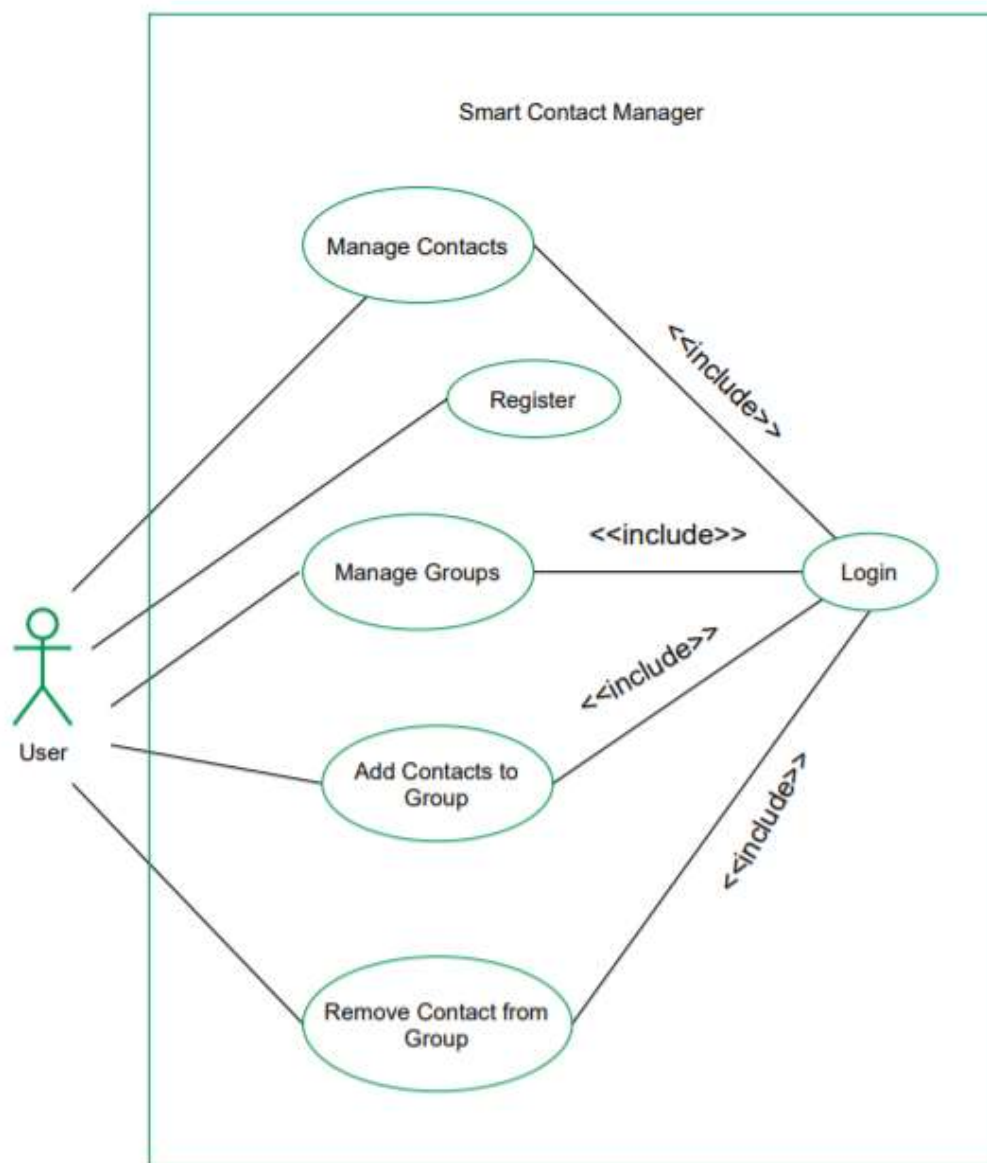
### Class Diagram



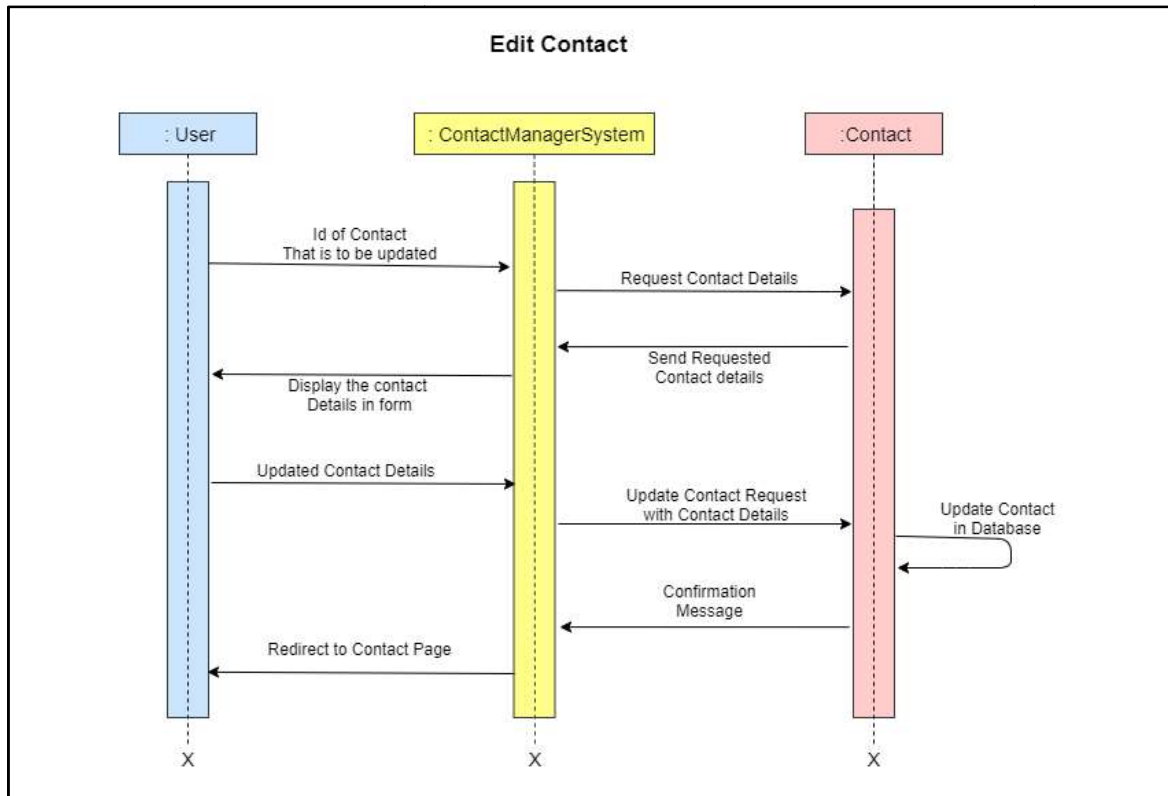
## State Diagram



## Use case Diagram

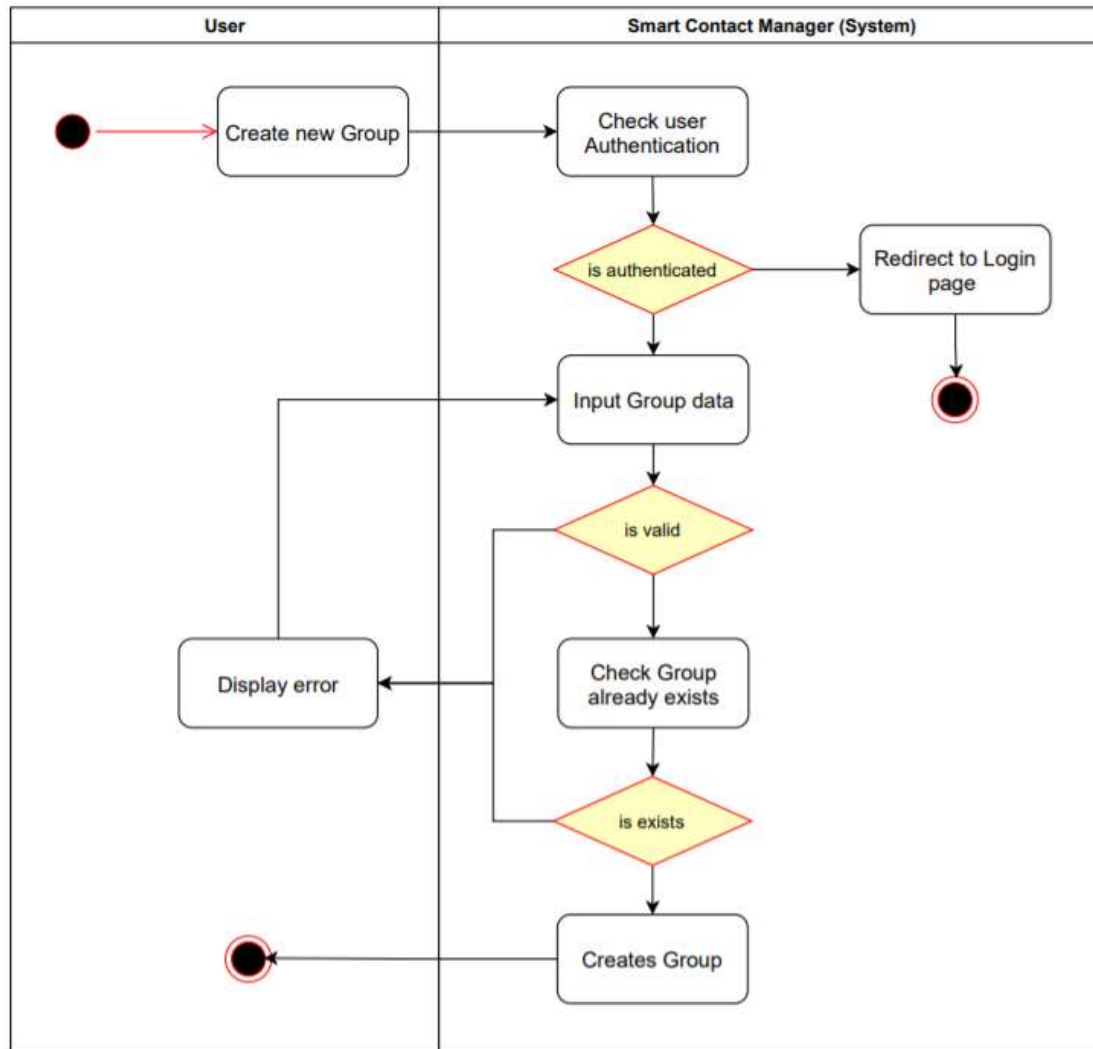


## Sequence Diagram

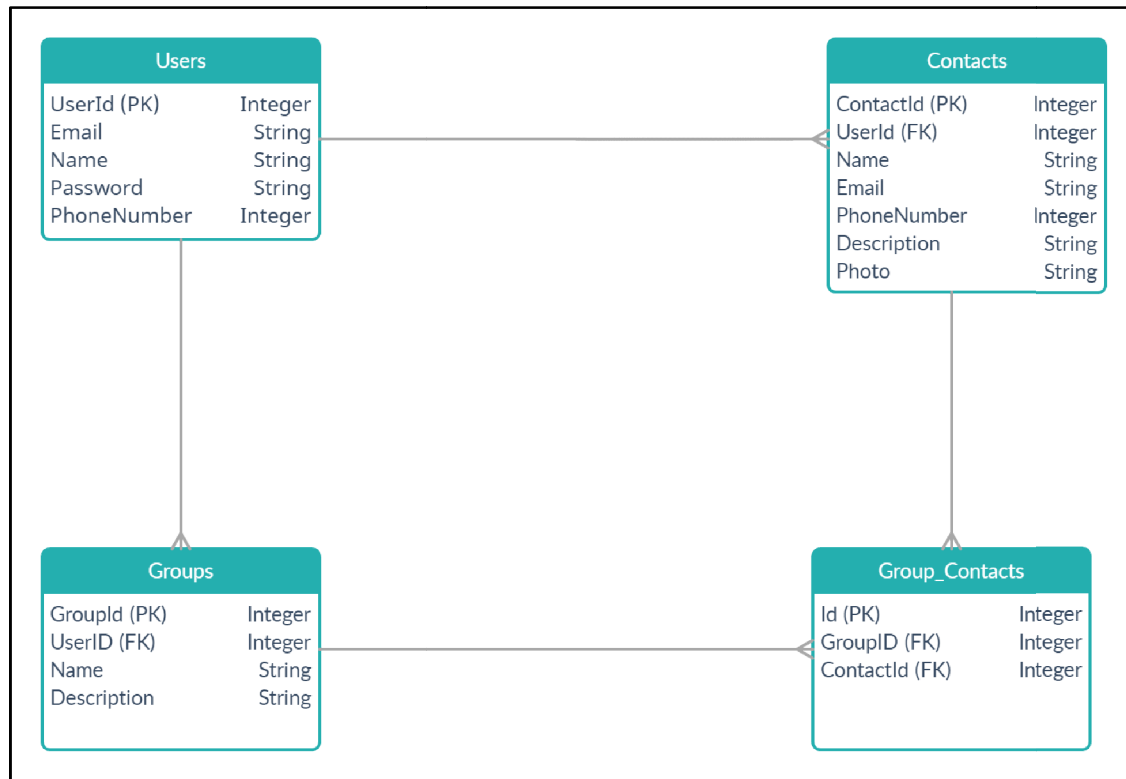


# Activity Diagram

Activity Diagram - Create Group



## ER Diagram



## Data Dictionary

### User

No	Field name	Data type	Required	Unique	PK / FK	Ref. Table
1	UserId	int	true	true	PK	-
2	Email	string	true	true	-	-
3	Name	string	true	false	-	-
4	Password	string	true	false	-	-
5	PhoneNumber	int	true	false	-	-



## Contact

No	Field name	Data type	Required	Unique	PK / FK	Ref. Table
1	ContactId	int	true	true	PK	-
2	UserId	int	true	false	FK	User
3	Name	string	true	false	-	-
4	Email	string	false	false	-	-
5	PhoneNumber	int	true	false	-	-
6	Description	string	false	false	-	-
7	Photo	image	false	false	-	-

## Group

No	Field name	Data type	Required	Unique	PK / FK	Ref. Table
1	GroupId	int	true	true	PK	-
2	UserId	int	true	false	FK	User
3	Name	string	true	true	-	-
4	Description	string	false	false	-	-

## GroupContact

No	Field name	Data type	Required	Unique	PK / FK	Ref. Table
1	Id	int	true	true	PK	-
2	UserId	int	true	false	FK	Classroom
3	ContactId	int	true	false	FK	Contact

## Implementation Detail

### Modules

Total modules :- 3

#### 1. Account Module :-

This module manages operations related to users' accounts like login, signup and forgot password. Existing users can login to the system by providing their email-id and password. New users can create their account by providing the required details.

#### 2. Contact Module :-

This module manages operations related to contacts. Users can add new contacts. They can view a list of existing contacts, detailed information of existing contacts. They can update & delete the contacts as well.

#### 3. Group Module :-

This module manages all group related operations. Users can create a new group. They can add contacts into a group & they can remove contacts from a group. They can view a list of contacts which are added in a group. They can update & delete the groups as well.

## Function prototypes which implements major functionality

### - Function prototypes of Account Module

- User GetUserByUsernameAndPassword(string username, string password);
- User GetUserByUsername(string username);
- User FindUserById(int id);
- User UpdateUser(User user);

- User CreateUser(RegisterUser user);

#### **- Function prototypes of Contact Module**

- IEnumerable<Contact> GetAllContactsByUserId(int id);\*/
- Contact GetContactById(int id);
- Contact UpdateContact(Contact contact);
- Contact CreateContact(CreateContact contact);
- Contact DeleteContact(Contact contact);

#### **- Function prototypes of Group Module**

- Group AddGroup(Group group);
- Group GetGroupById(int id);
- IEnumerable<Group> GetAllGroups(int userId);
- void UpdateGroup(Group group);
- void DeleteGroup(Group group);
- void AddGroupContacts(IEnumerable<GroupContact> groupContacts);
- IEnumerable<GroupContact> GetGroupContactsByGroupId(int groupId);
- void DeleteGroupContact(int id);

## Testing

Unit testing of each module was done after successfully completing the module. Each module was tested individually before integrating them with the whole system. After integrating each module with the system, integration testing was done in order to check if modules are working properly together. After completing all integrations, black-box testing of the whole system was carried out to ensure the system works in a correct manner.

### Black box testing of Major functions of the system

#### 1. Log in to the system.

**Case 1 :** Invalid Username or password entered by the user.

**Output :** Error message on the screen saying "Invalid credentials"

**Case 2 :** Valid credentials.

**Output :** The user is redirected to the Dashboard page.

#### 2. Edit Contact

**Case 1 :** Contact number already exists.

**Output :** Error message on the screen saying "Contact already exists"

**Case 2 :** Some of required fields missing in input.

**Output :** Model validation errors will be displayed to the user.

**Case 3 :** All input data are valid.

**Output :** Contact updated successfully.

#### 3. View Group.

**Case 1 :** User is not logged in.

**Output :** Redirected to the login page with error message "Please login..!".

**Case 2 :** If a group exists.

**Output :** All contacts and group information will be displayed

**Case 3 :** Provided group ID in url not exists at all.

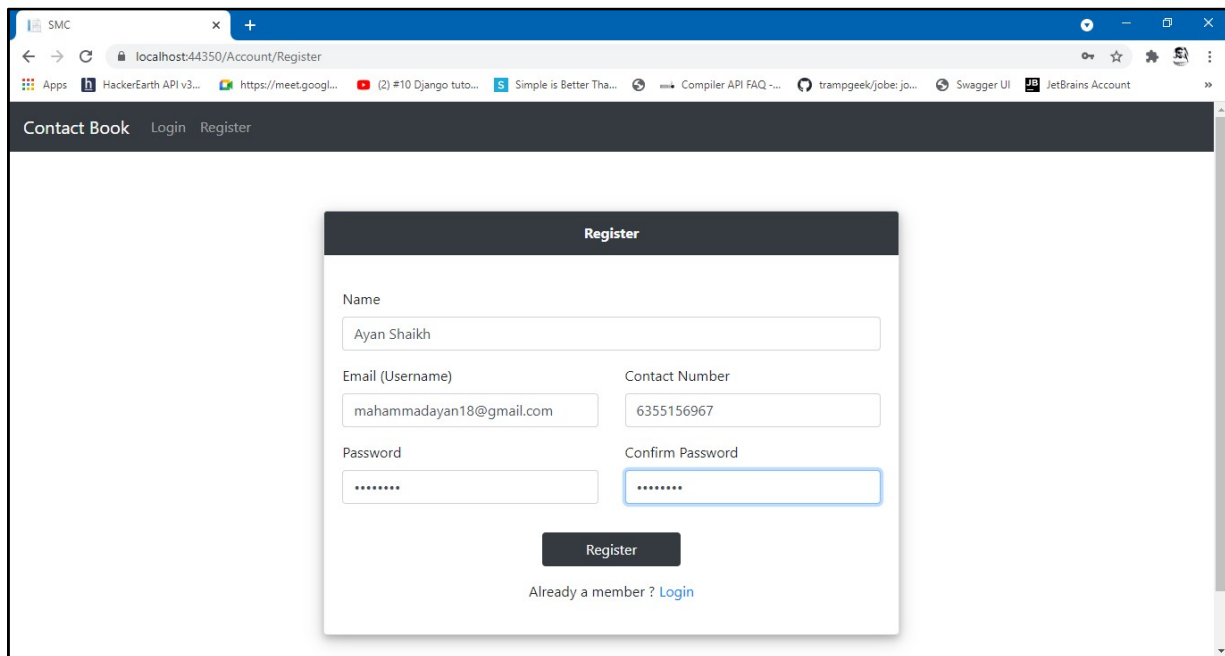
**Output :** 404 Error page displayed.

**Case 4 :** Provided group ID in url does not belong to the logged in user.

**Output :** 404 Error page displayed.

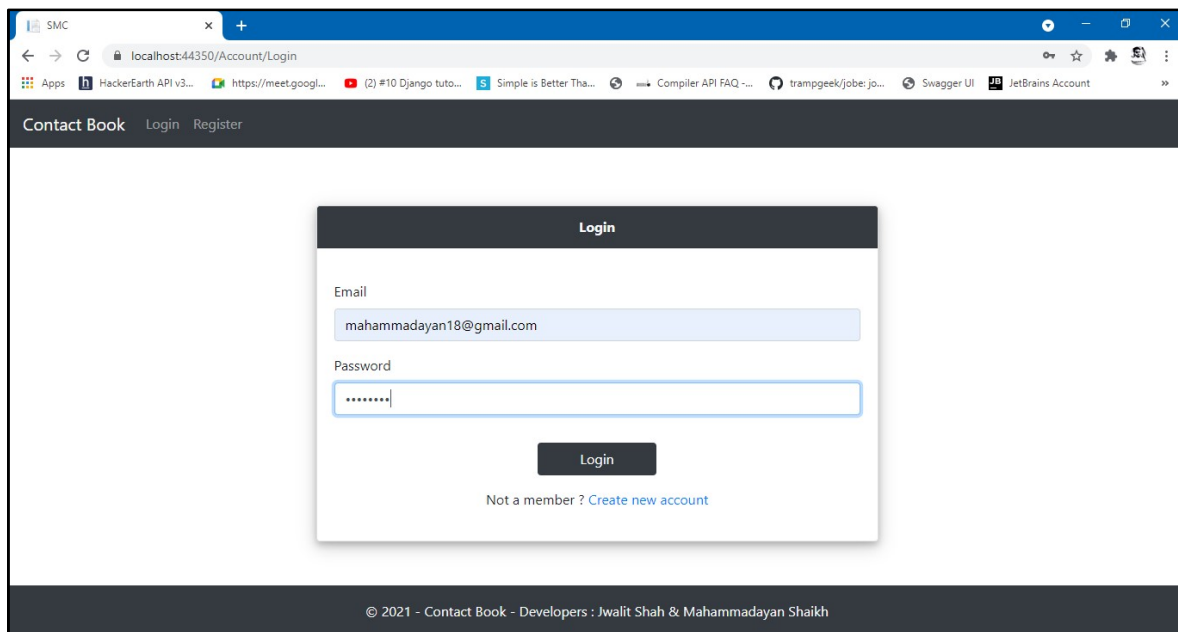
# Screenshots

## 1. Signup



The screenshot shows a web browser window with the URL `localhost:44350/Account/Register`. The browser's address bar and tabs are visible at the top. Below the browser window, there is a dark navigation bar with the text "Contact Book" and links for "Login" and "Register". The main content area displays a "Register" form. The form has a dark header with the word "Register" in white. It contains four input fields: "Name" (with the value "Ayan Shaikh"), "Email (Username)" (with the value "mahammadayan18@gmail.com"), "Contact Number" (with the value "6355156967"), and "Password" (with masked characters "\*\*\*\*\*"). There is also a "Confirm Password" field with masked characters. A "Register" button is located below the fields. At the bottom of the form, there is a link: "Already a member ? [Login](#)".

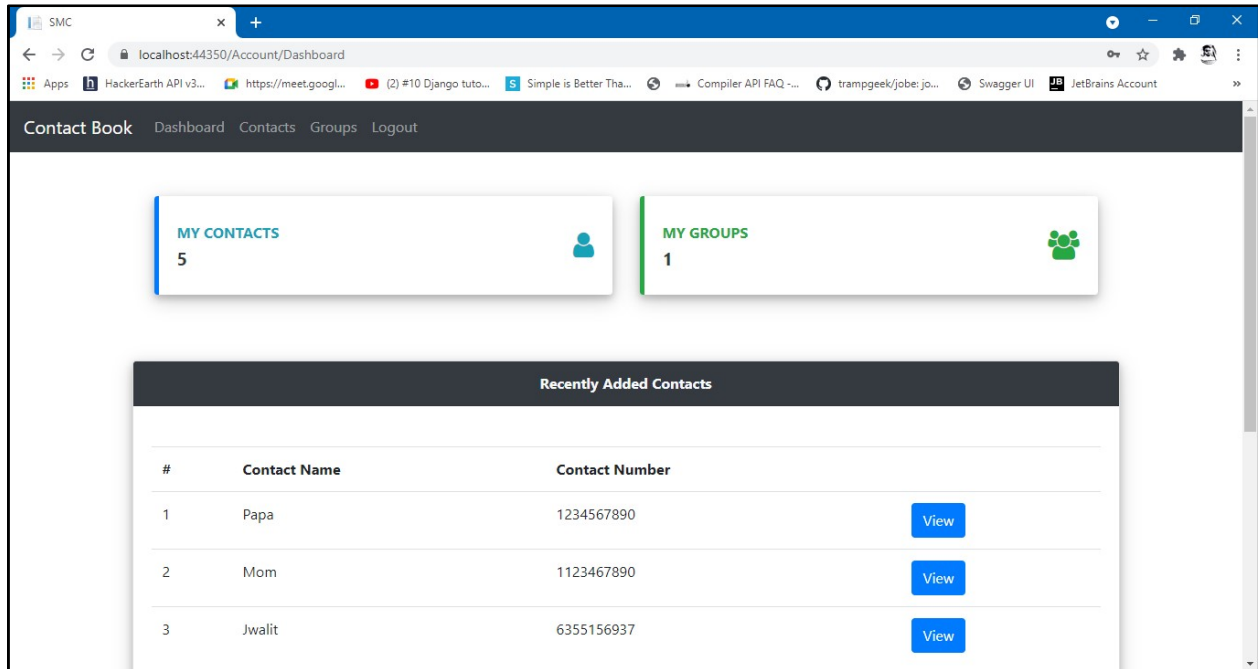
## 2. Login



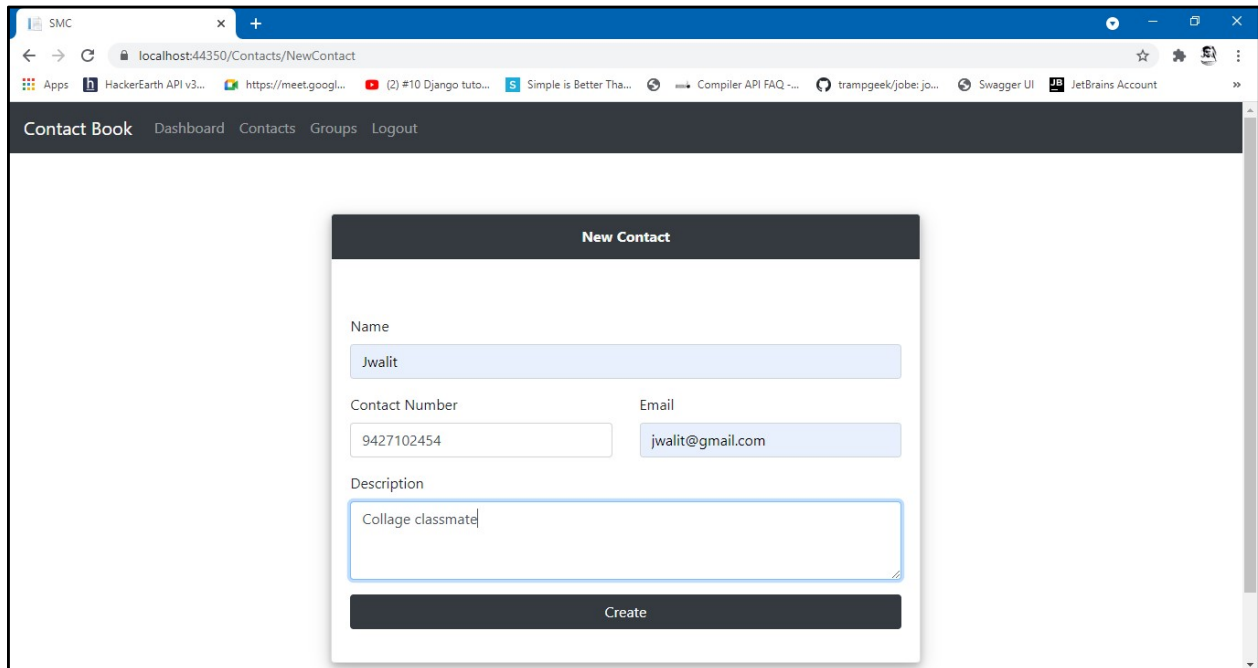
The screenshot shows a web browser window with the URL `localhost:44350/Account/Login`. The browser's address bar and tabs are visible at the top. Below the browser window, there is a dark navigation bar with the text "Contact Book" and links for "Login" and "Register". The main content area displays a "Login" form. The form has a dark header with the word "Login" in white. It contains two input fields: "Email" (with the value "mahammadayan18@gmail.com") and "Password" (with masked characters "\*\*\*\*\*"). A "Login" button is located below the fields. At the bottom of the form, there is a link: "Not a member ? [Create new account](#)".

© 2021 - Contact Book - Developers : Jwalit Shah & Mahammadayan Shaikh

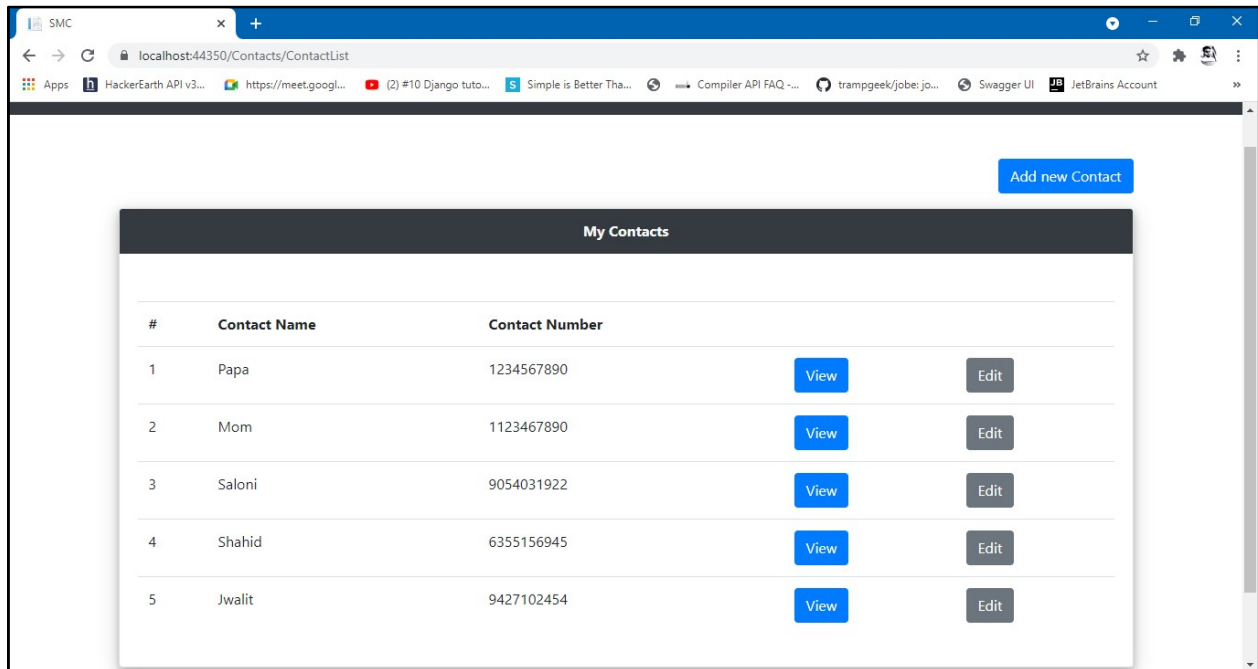
### 3. Dashboard



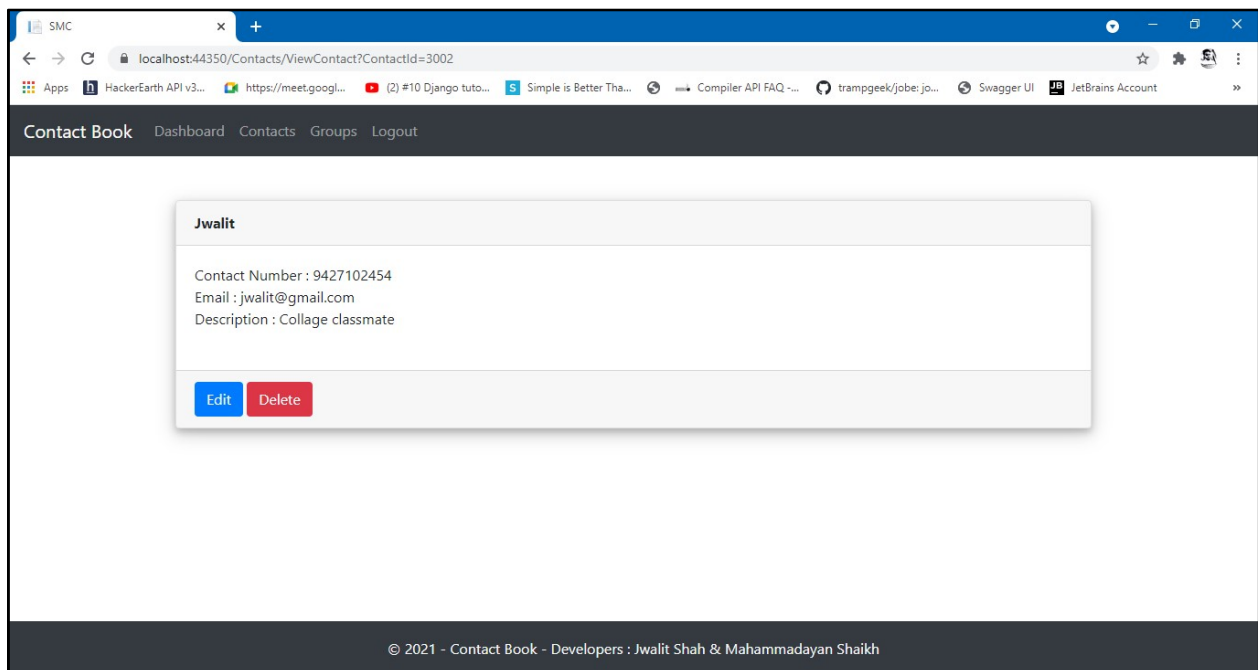
### 4. New Contact



## 5. Contact List

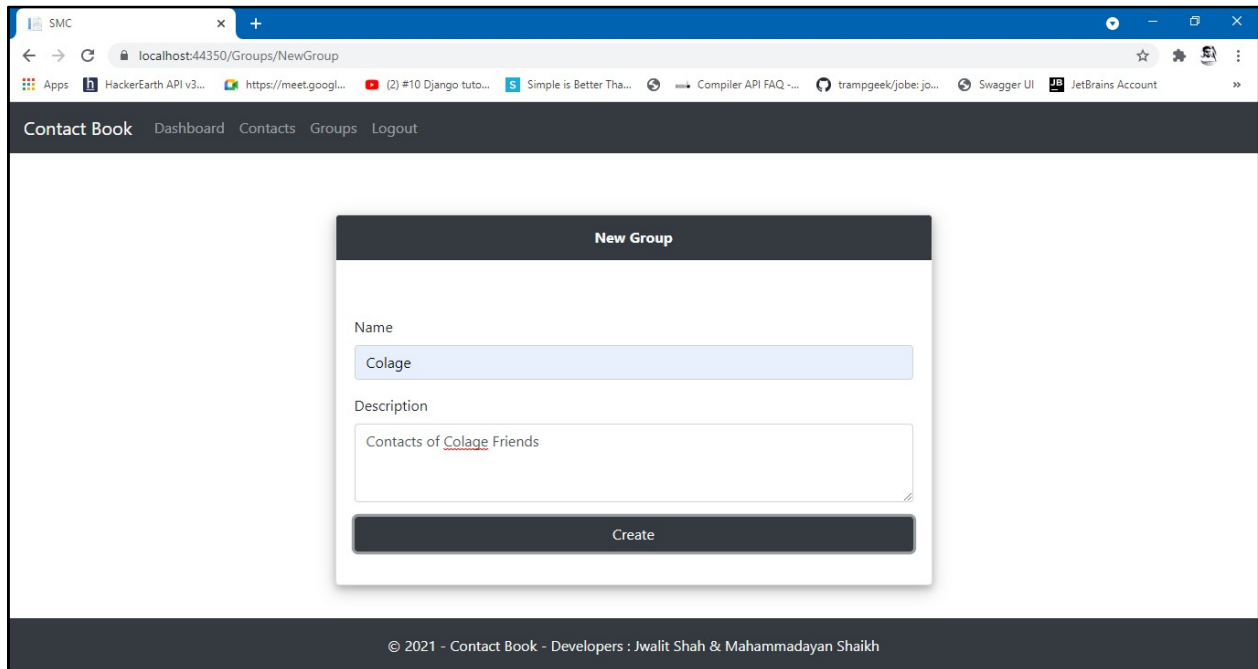


## 6. View Contact





## 7. New Group



The screenshot shows a web browser window with the URL `localhost:44350/Groups/NewGroup`. The application has a navigation bar with links: **Contact Book**, [Dashboard](#), [Contacts](#), [Groups](#), and [Logout](#). The main content area displays a modal titled "New Group". Inside the modal, there is a "Name" field with the value "Colage" and a "Description" field with the value "Contacts of Colage Friends". A "Create" button is at the bottom of the modal. The footer of the application reads "© 2021 - Contact Book - Developers : Jwalit Shah & Mahammadayan Shaikh".

**New Group**

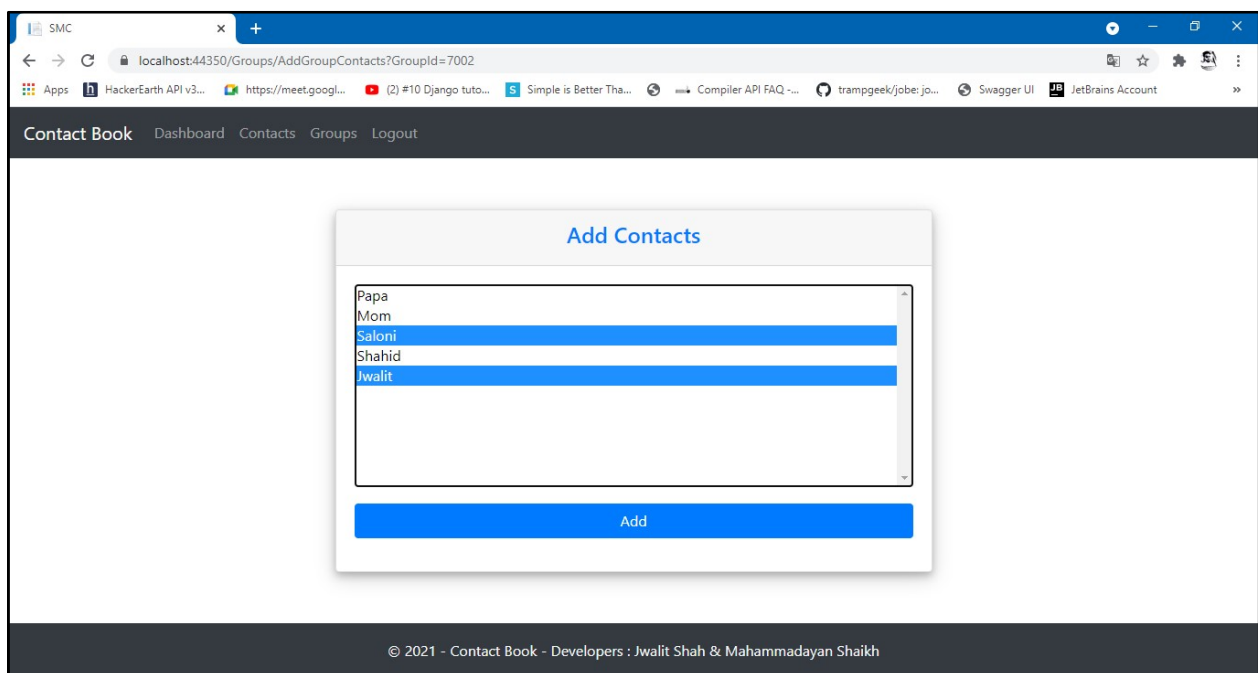
Name  
Colage

Description  
Contacts of Colage Friends

Create

© 2021 - Contact Book - Developers : Jwalit Shah & Mahammadayan Shaikh

## 8. Add Contacts in group



The screenshot shows a web browser window with the URL `localhost:44350/Groups/AddGroupContacts?GroupId=7002`. The application has a navigation bar with links: **Contact Book**, [Dashboard](#), [Contacts](#), [Groups](#), and [Logout](#). The main content area displays a modal titled "Add Contacts". Inside the modal, there is a list of contacts: Papa, Mom, Saloni, Shahid, and Jwalit. The contact "Jwalit" is selected and highlighted in blue. An "Add" button is at the bottom of the modal. The footer of the application reads "© 2021 - Contact Book - Developers : Jwalit Shah & Mahammadayan Shaikh".

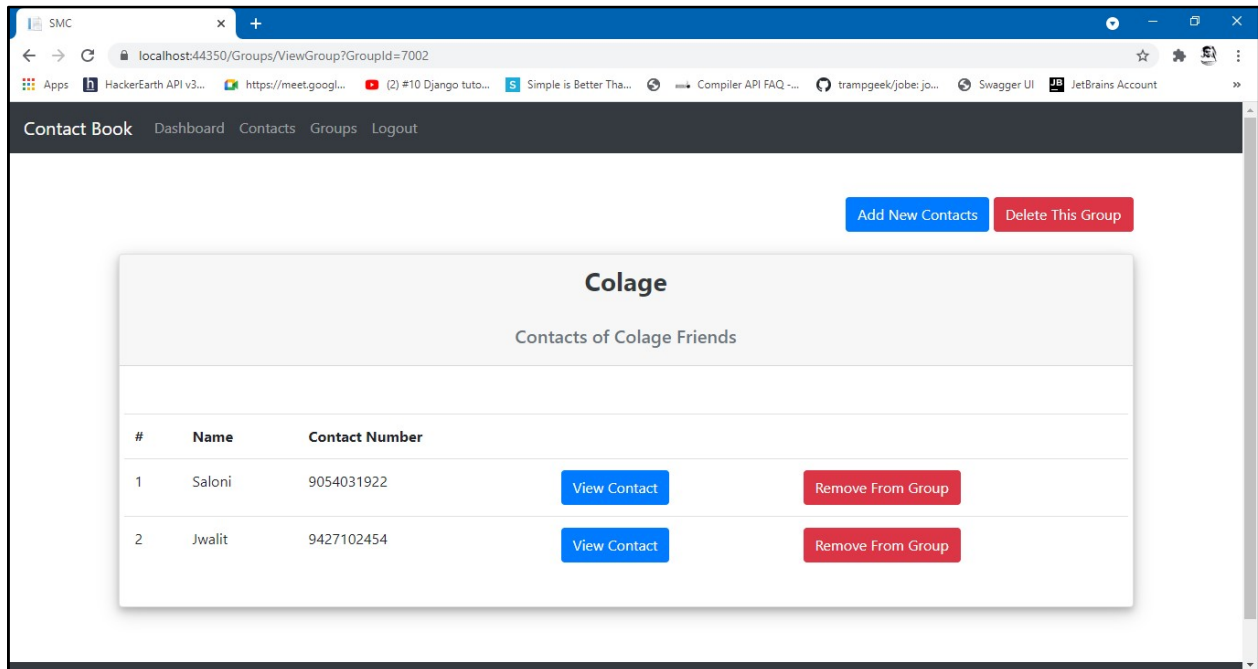
**Add Contacts**

Papa  
Mom  
Saloni  
Shahid  
Jwalit

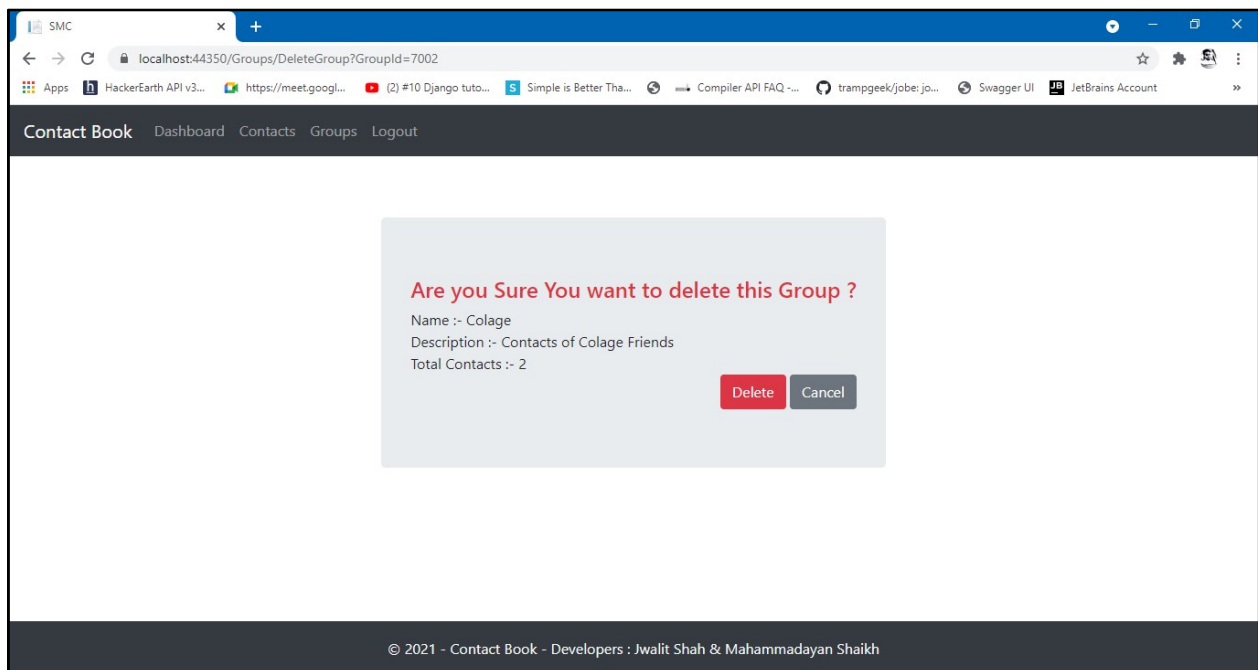
Add

© 2021 - Contact Book - Developers : Jwalit Shah & Mahammadayan Shaikh

## 9. View Group



## 10. Delete Group



## **Conclusion**

### **Functionalities implemented successfully :**

- Login by user.
- Registration by user.
- Create contact
- Update contact details
- View contacts list
- View contact details
- Delete contact
- Create group
- Update Group details
- Add Contacts to the group
- View Contact list of specific group
- Remove contact from group
- Delete group
- Logout from the system

## **Limitation and Future Extensions**

### **● Limitations**

- User can't login in case if password is not known

### **● Functionalities not implemented**

- Forgot password
- Update profile
- Import contacts
- Export contacts

### **● Possible future extensions**

- Users can upload their profile picture & update profile.
- Users can update their old password to the new password using forgot password in case if password is not known.
- Users can add the contacts by Importing contacts as .vcf file and can export the contacts in .vcf file.

## Bibliography

To build this project we have taken references from following websites,

- Frameworks used for development :-  
<https://docs.microsoft.com/en-us/dotnet/framework/wcf/> (For backend service)  
<https://docs.microsoft.com/en-us/aspnet/core/web-api/?view=aspnetcore-5.0>  
(For backend API)  
<https://docs.microsoft.com/en-us/dotnet/framework/wcf/>(For frontend)
- CSS framework for frontend designing :-  
<https://getbootstrap.com/>
- For debugging :-  
<https://stackoverflow.com/>