# Software Requirements Specification

**for**

# MindMatch

**Version 1.0 approved**

**Prepared by AKHIL (18CS10070)**

**Indian Institute of Technology, Kharagpur**

**12 February, 2020**

# Table of Contents



# Revision History

| Name | Date | Reason for Changes | Version |
|---|---|---|---|
| | | | |
| | | | |

# 1. Introduction

## 1.1 Purpose

The purpose of this document is to show the detailed explanation of the objectives, features, user interface and application of **MindMatch**. Document provides the detailed profile of the external interfaces, performance considerations and design constraints imposed on the implementation.

## 1.2 Need of this Project

Many times, we don't know the other people of our community who have similar interests, goals and may be working on similar projects. So, if we can get to know about such people as early as possible this would definitely increase our productivity and will lead us to our goal at a faster rate.
We also face difficulty when we want to find a person with some particular interests.
For example, there are 2 persons in an Institute who are preparing for a competitive exam but don't know each other because of their different departments and hostels etc., then this application will help them know and connect to each other.

## 1.3 Addressing the Need

As the name suggests this application will be aimed at connecting like-minded people in the community by giving them suggestions and % match with each of the suggested persons.
Unlike many social/professional networking sites like FB, Instagram, LinkedIn this application is made especially for the college community so that they can easily find friends with similar goals, habits and academic interests. Studying/working together with such friends will definitely increase their productivity and will keep them motivated.
This software takes in the personal data, educational details and the professional information as input through a **user-details form** and then calculates % of things common between two users and suggests a match.
In addition to that the user will also get the option to search for people in the community with filters of their choice, then they can connect and will also get the facility to chat to each other inside the application itself.
Some of the fields of the user details will have higher weightage in calculating the % match like their academic goals and hobbies.

## 1.4 Prospective Users

Major types of users are:

- **Students:** The users who are attending a regular school/college/university.
- **Developer:** (Admin)

## 1.5 Issues/challenges to be overcome

As this application holds an important role in connecting the community so the % match calculating algorithm should be very accurate and should focus more on some of the user details by assigning proper weights to them.

These weights should be changed according to the class of the user.

# 2. Plan of Working

**2.1 Week 1 (26.02.2020 – 04.03.2020):** This period will focus on developing the basic design of the software, how to execute the functional requirements and an idea of the classes and objects to be used. Also, we try to learn where to collect data from and different tools like HTML, JavaScript and Java.

**2.2 Week 2 (04.03.2020 - 11.03.2020):** Frontend will be developed. Various user interaction buttons and links will have to be integrated in the Web App. The concept of GUI-based programming will be required.

**2.3 Week 3 - 4 (11.03.2020 - 25.03.2020):** Backend of the website will be developed using Java and database tools. Different functionalities as proposed in the SRS, would be added. Implementation part would be done by now and our software would be ready for testing and debugging.

**2.4 Week 5-6 (15.04.2020 - 1.05.2020):** Testing will begin. Efforts would be made to fix bugs, if found, and make the software as robust as possible. And final submission will be done.

# 3. Functional Requirements

## 3.1 Creating new account

Step-1: SignUp by user (through some unique identifiers like username/email ID).
He/She will fill the information asked by the application such as username, email ID, first name and password.
Step-2: By completing the above procedure users will get a successful registration message.
Step-3: All the user data will be saved in the database.

## 3.2 Matching/Connection-Suggestion

Step-1: A new user registers by filling the **user-details form.** Database gets updated.
Users can use this **get suggestions** option only after filling user-details form.

Step-2: Now this algorithm will be implemented(when user chooses the option of **get suggestions)** and the % match of this user will be calculated with all existing users.

Step-3: The suggested users will be shown in decreasing order of % match.

### 3.3 Privacy settings

Step-1: Any existing user can choose which information of his profile can be seen by the people in the community.
Step-2: Database will be updated about which information to show.
Step-3: Now whenever anyone uses the search option the privacy will be maintained.

### 3.4 Searching in the database

Step-1: Current user can search others by their first name, last name and username or any substring of these.
Step-2: Database will be searched and the users will be shown in decreasing order of % match.

### 3.5 Admin login and monitoring

**Admin has the following options:**

Option-1: See the newly registered users on a daily/weekly/monthly basis (GRAPH).
Option-2: See the total count of Users.
Option-3: See the total count of Connections.
Option-4: See the total count of Messages.

### 3.6 Chatting

Step-1: Two users will connect to each other either by suggestion or by searching.
Step-2: Now they have the option to chat with each other.
(Message can be sent even if they are not connected).

# 4. Non-Functional Requirements

## 4.1 Security Requirements

- As this application uses a large amount of data of users, so security of the data is a major concern.
- All the personal data stored for a particular user will be encrypted by using the standard encryption algorithm. This needs to be protected from any possible data theft.

## 4.2 Performance Requirements

- The response of the server should be fast when provided with high speed internet.
- The data should be stored in the database in such a way which makes it fast to search.

### 4.3 Software Quality Attributes

- The software must work with at least Chrome and Firefox browser (these two are the most user web browsers).
- Software must operate major operating systems like windows and ubuntu.

### 4.4 Business Rules

- This will be a free to use application.
- This will be an open source application so that anyone can contribute.

## 5. External Interface Requirements

### 5.1 User Interfaces

It will be a Web-Application. All the main menu and settings options such as account settings, profile management etc. will always be available to the user. The interface will be user friendly

### 5.2 Hardware Interfaces

There are no special hardware interface requirements.
Any device with an internet connectivity and internet browser will be suitable for using the system

### 5.3 Communication Interfaces

- There are no special communication interfaces requirements. All communication with the user takes place within the application itself.
- The major communication between the system and the user will be when he fills the user-details form.

## 6. Tentative hardware/software environment

- The Graphical user interface will be constructed upon HTML5, CSS and JavaScript. (and some other required softwares.)
- The Server will be hosted on an online hosting site (heroku in our case).

# 7. Publicizing the system

- The webpage will be attractive and on the homepage we will clearly mention the basic idea of the application.
- This application is first of the kind, so we can also spread the information about this application in the student community of a university.
- We can compare this application with existing social and professional websites and will show how this application is different from them. Also how this application is made specifically for the college community.

# Appendix A: Glossary

No special terminology is used.

# Software Design Document

**for**

# MindMatch

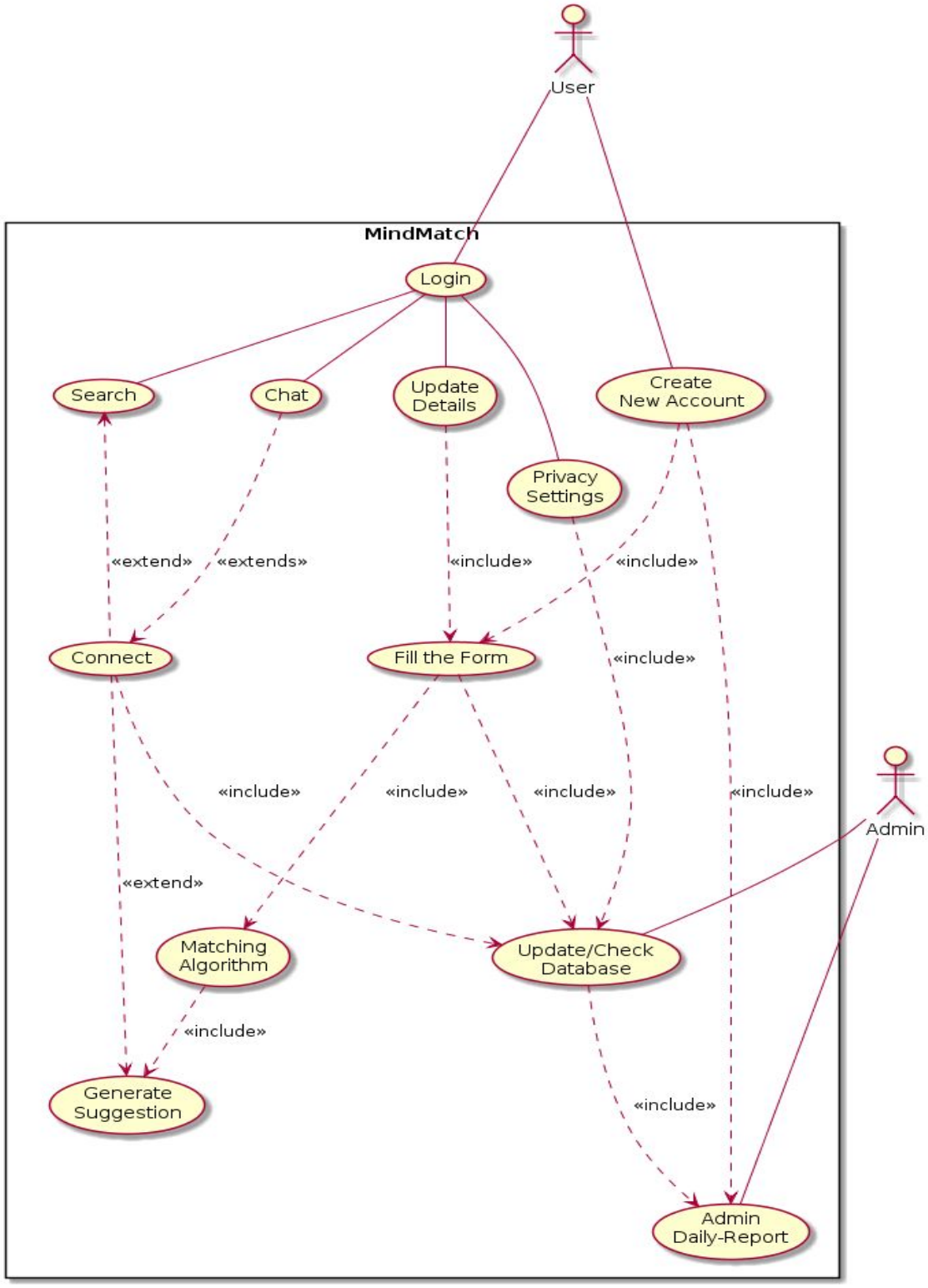
**Version 1.0 approved**

**Prepared by AKHIL (18CS10070)**

**Indian Institute of Technology, Kharagpur**

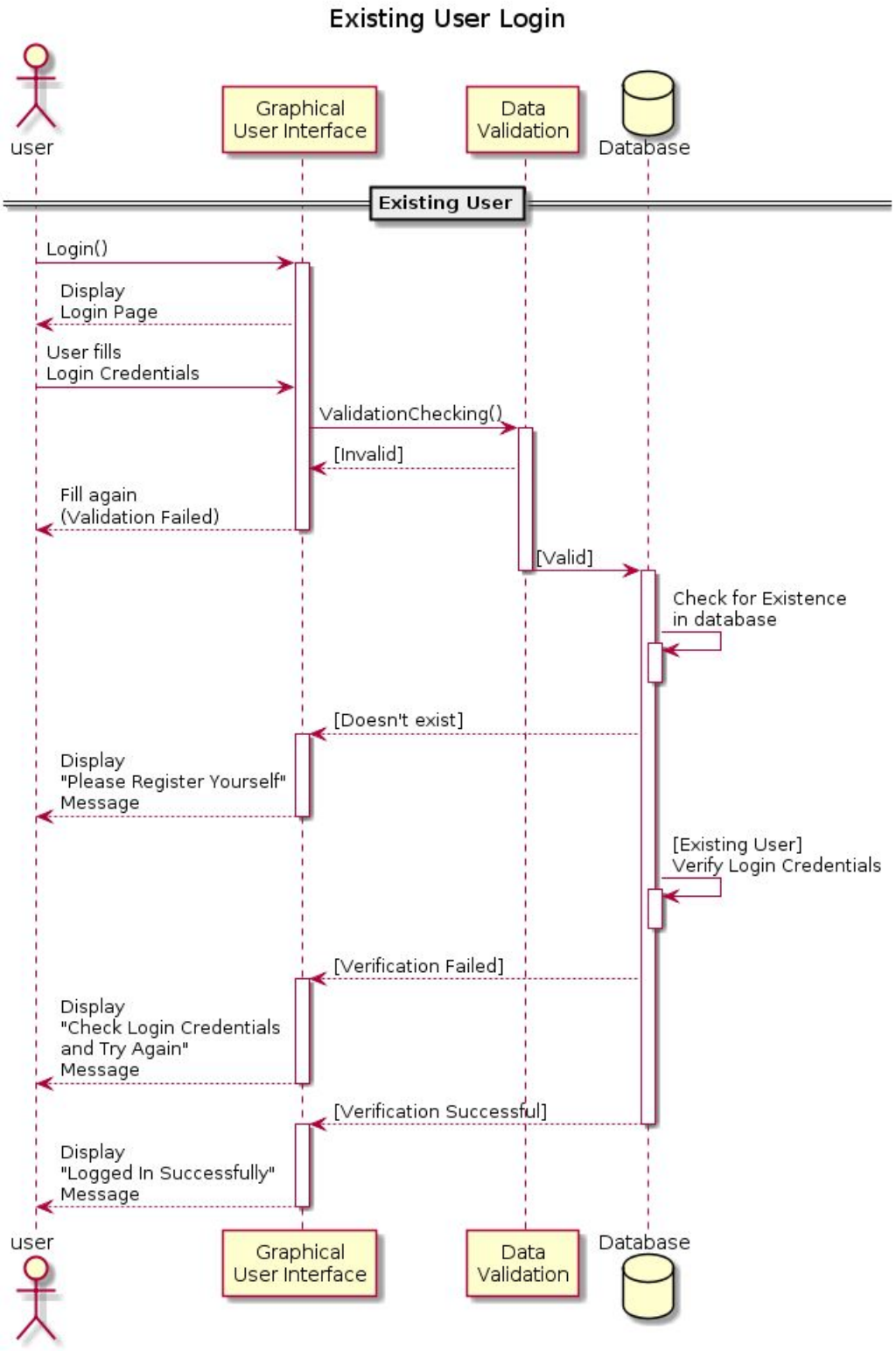
**26 February, 2020**

# Use Case Diagram

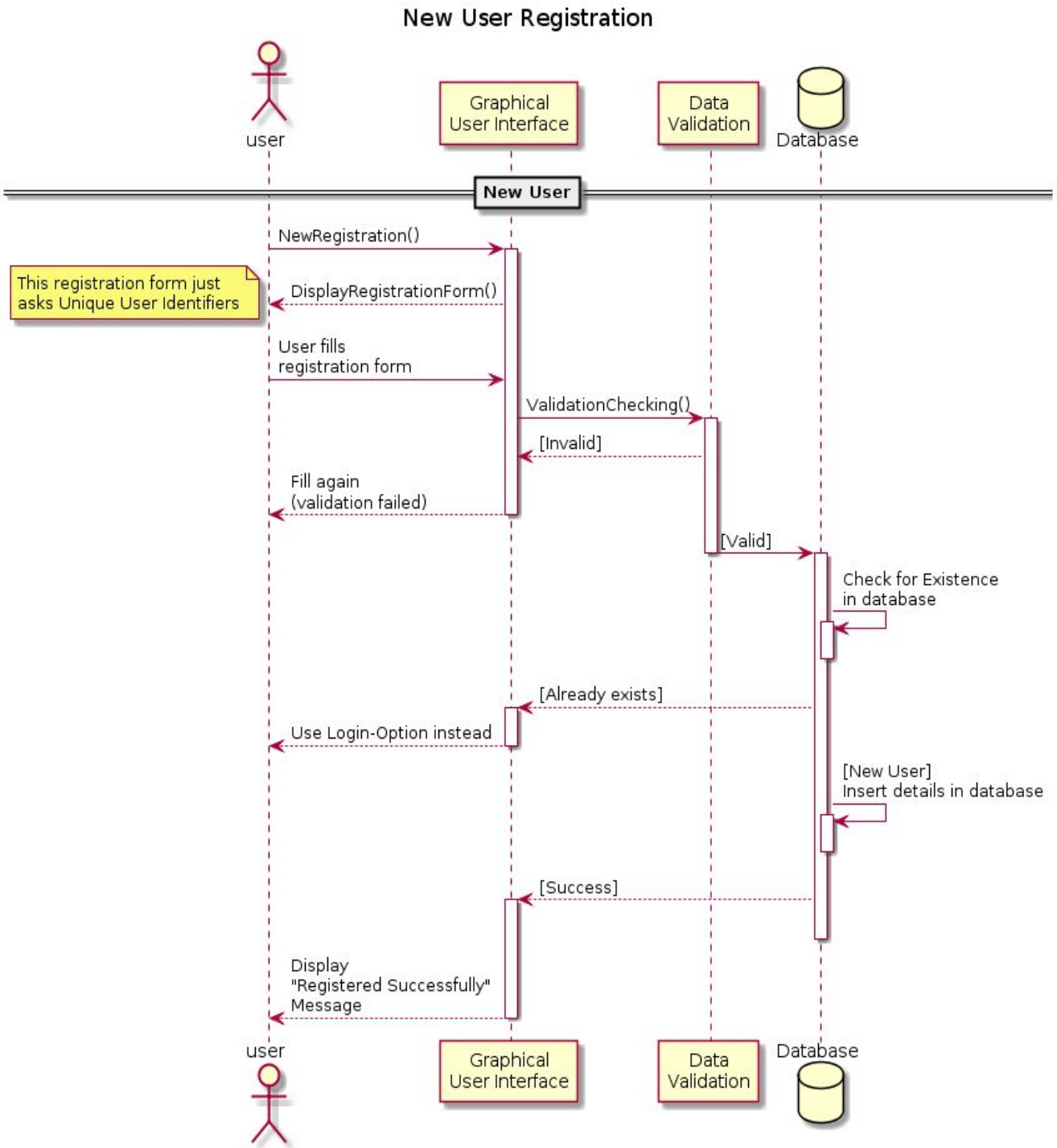


# Sequence Diagrams

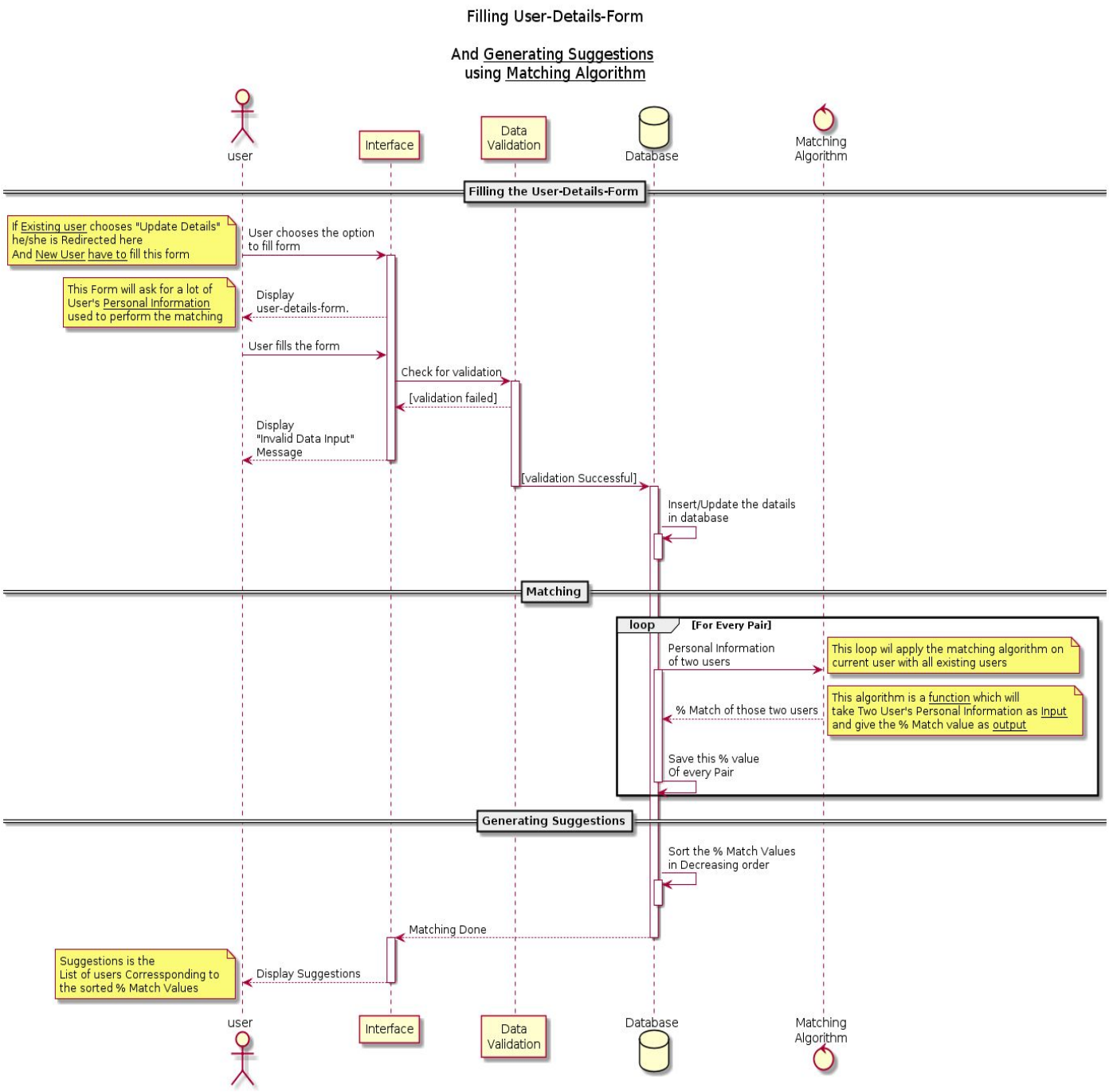
## (1 of 6)



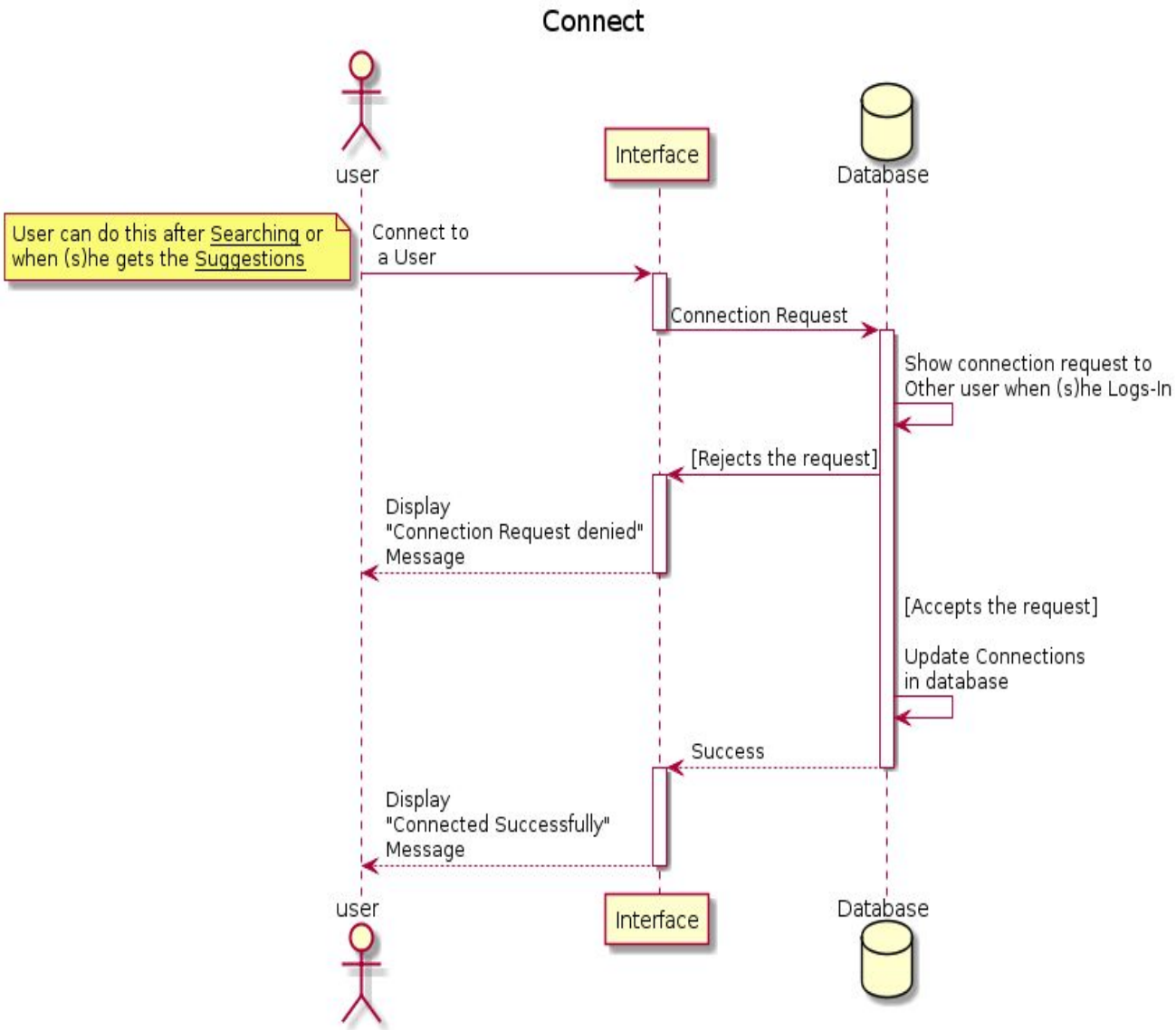
# (2 of 6)


New User Registration
user
Graphical User Interface
Data Validation
Database
New User
NewRegistration()
This registration form just asks Unique User Identifiers
DisplayRegistrationForm()
User fills registration form
ValidationChecking()
[Invalid]
Fill again (validation failed)
[Valid]
Check for Existence in database
[Already exists]
Use Login-Option instead
[New User] Insert details in database
[Success]
Display "Registered Successfully" Message
user
Graphical User Interface
Data Validation
Database

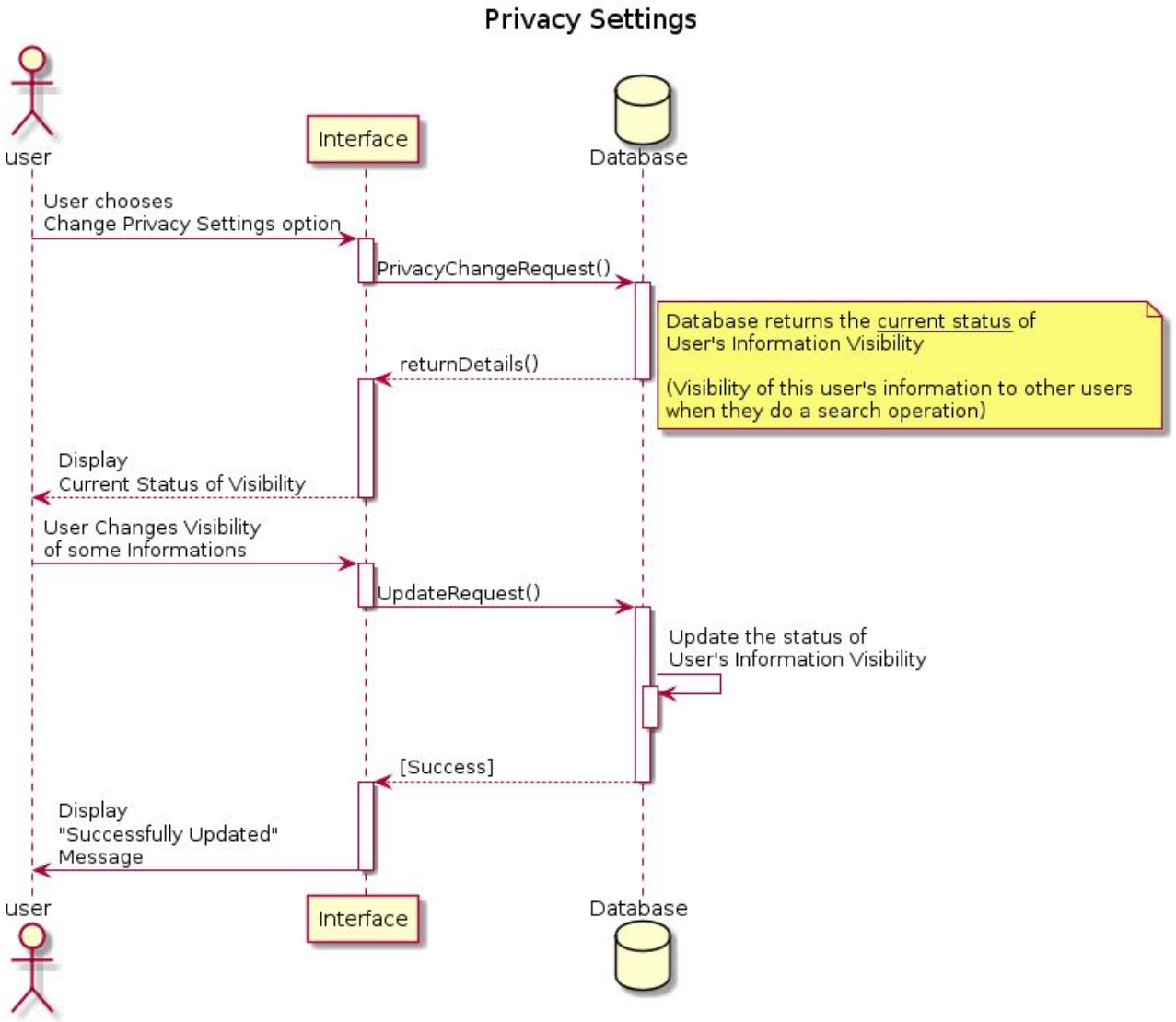
# (3 of 6)


Filling User-Details-Form
And Generating Suggestions
using Matching Algorithm
user
Interface
Data Validation
Database
Matching Algorithm
Filling the User-Details-Form
If Existing user chooses "Update Details" he/she is Redirected here And New User have to fill this form
User chooses the option to fill form
This Form will ask for a lot of User's Personal Information used to perform the matching
Display user-details-form.
User fills the form
Check for validation
[validation failed]
Display "Invalid Data Input" Message
[validation Successful]
Insert/Update the datails in database
Matching
loop
[For Every Pair]
Personal Information of two users
This loop wil apply the matching algorithm on current user with all existing users
% Match of those two users
This algorithm is a function which will take Two User's Personal Information as Input and give the % Match value as output
Save this % value Of every Pair
Generating Suggestions
Sort the % Match Values in Decreasing order
Matching Done
Suggestions is the List of users Corressponding to the sorted % Match Values
Display Suggestions
user
Interface
Data Validation
Database
Matching Algorithm

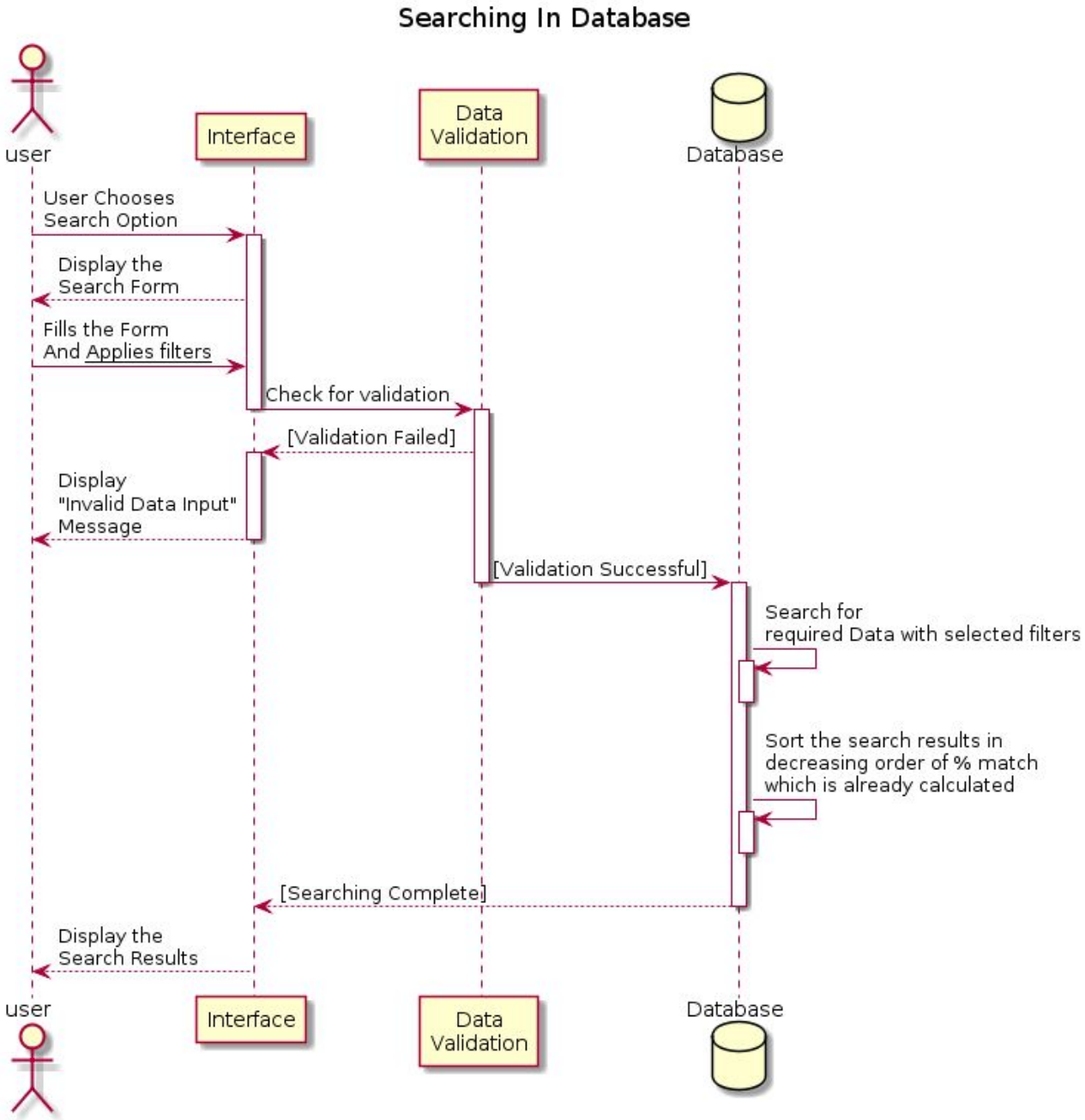
# (4 of 6)


Connect
user
Interface
Database
User can do this after Searching or when (s)he gets the Suggestions
Connect to a User
Connection Request
Show connection request to Other user when (s)he Logs-In
[Rejects the request]
Display "Connection Request denied" Message
[Accepts the request]
Update Connections in database
Success
Display "Connected Successfully" Message
user
Interface
Database

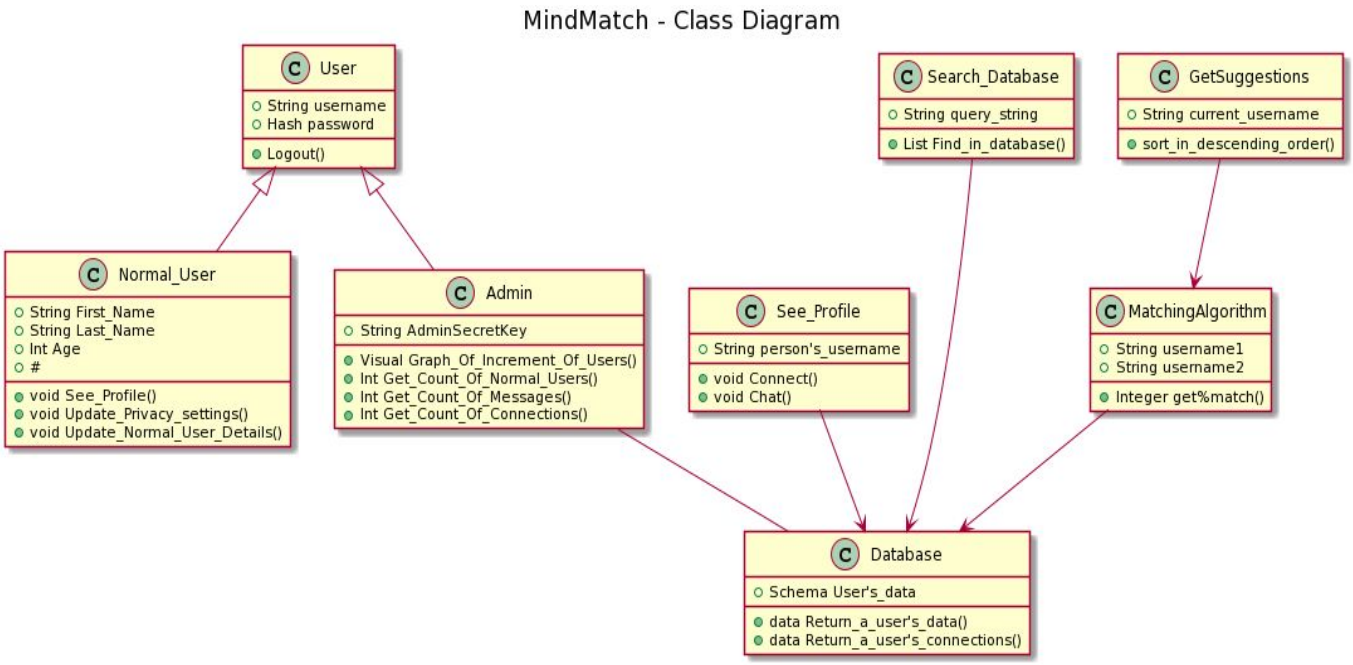
# (5 of 6)


Privacy Settings
user
Interface
Database
User chooses
Change Privacy Settings option
PrivacyChangeRequest()
Database returns the current status of
User's Information Visibility
(Visibility of this user's information to other users
when they do a search operation)
returnDetails()
Display
Current Status of Visibility
User Changes Visibility
of some Informations
UpdateRequest()
Update the status of
User's Information Visibility
[Success]
Display
"Successfully Updated"
Message
user
Interface
Database

# (6 of 6)


Searching In Database
user
Interface
Data Validation
Database
User Chooses Search Option
Display the Search Form
Fills the Form And Applies filters
Check for validation
[Validation Failed]
Display "Invalid Data Input" Message
[Validation Successful]
Search for required Data with selected filters
Sort the search results in decreasing order of % match which is already calculated
[Searching Complete]
Display the Search Results
user
Interface
Data Validation
Database

# CLASS DIAGRAM




**#** as the user have a lot of properties/data, so it's not convenient to represent in diagram

| **Name** | **Data Type** |
|---|---|
| department | String |
| hometown | String |
| sports interest value | number |
| tech interest value | number |
| social activities interest | number |
| hobbies | [String] |
| personality type | string |

NOTE : As this project is based totally on **node.js** which is not strictly based on OOPS (object oriented paradigm) as well as I didn't Implement a lot of classes in my code. So the Class diagram may not be perfect. (https://qr.ae/pNySPJ)