PROJECT REPORT
ON

Online Matrimonial System

By

SHAH SHAIL (CE-142) (19CEUON088)
THAKKAR ADITYA M. (CE-130) (19CEUOS131)
THAKKAR SAGAR D. (CE-131) (19CEUOG026)

B.Tech-CE Semester-V
Subject: Advanced Technologies

Guided by:
Prof. Prashant Jadhav
Prof. Siddharth Shah
Dept. of Computer Engineering.

Faculty of Technology
Department of Computer Engineering
Dharmsinh Desai University

Table of Contents

1. Abstract

Online Matrimonial System is an online match making system.
The software is proposed to develop a system to find the users desire match for marriage purpose. A random user can also search for his/her desired match. Also the user can view his profile, look into others profile, can request a meeting etc.

2. Introduction

2.1 Brief Introduction

Any user who wants to use this system can first of all register themselves or else he/she can only search the required candidate, if already registered then the user can directly go to the login page and provide his/her details there.

When the user registers for the first time he/she has to fill his/her details such as name, age, data of birth, city, state, gender etc. if the user details are not validated, then an error is shown or if the details has been empty then also an error of field empty has been thrown. After the registration has been done successfully then the user is been redirected to the login page.

In the login page the user has to enter his/her email and password provided during the registration. After the login details has been entered then the email and the password has been checked in the database and is redirected to the home page.

After login the user can check his/her profile as well as edit/update it as per required. The user can also delete his/her profile by clicking on the delete button provided in the user profile page.

The user or a random visitor can search his/her partner by filtering his/her required mentioned criteria. If the match('s) has been found successfully then all the users that has been registered is been displayed or else no match found has been displayed.

2.2 Tools/Technologies Used

Technologies:

- Mongo Db
- Bootstrap
- JavaScript
- HTML
- CSS
- Express JS
- React JS
- Node JS

Tools:

Visual Studio Code

Platform

Local development server

3. Software Requirement Specifications

3.1 End Users

- Customer
- Guest User

3.2 System Functional Requirements

R.1. USER MANAGEMENT

R.1.1 : USER REGISTRATION

Description : The details of the new customer are asked for such as name, age, Email-Id, date of birth, marital-status, gender, mother tongue, religion, city, pin code etc.

Input : User selection

Output : Unique user id.

Processing: If all details are authentically filled then user is registered OR if the details are invalid then message accordingly should be displayed.

R.1.2 : VIEW PROFILE

Description : User can view his/her current details .

Input : User selection

Output: User details.

R.1.3 : UPDATE PROFILE

Description : User can update his/her details and can keep the profile up to date.
Input : Updated user details

Output: Confirmation message with updated details of the user.

Processing: If all details are authentically filled then profile of the user is updated OR if the details are invalid then message accordingly should be displayed.

R.1.4 : DELETE PROFILE

Description : The user is permanently deleted from the portal.

Input : User selection.

Output: Confirmation message.

R.1.4 : USER LOGIN

Description : User can login by entering username and password . If username and password match correctly then confirmation message should be displayed.

Input : Username and Password of user.

Output: User Profile.

Processing: Password validation

R.2. SEARCH MODULE

R.2.1 : Search for Partner

Description : The user can search for their partner based on specified criteria and can view the matches based on that.

Input : User selection.

Output : Profile Matches based on user selection.

R.3. PARTNER MODULE

R.3.1 : REQUEST PARTNER

DESCRIPTION : User can send a friend request to the selected partner.

INPUT : User selection.

OUTPUT: Confirmation message.

PROCESSING: Request is sent to the specific user.

R.3.2 : CHAT WITH PARTNER

DESCRIPTION :The user can chat with their friends.

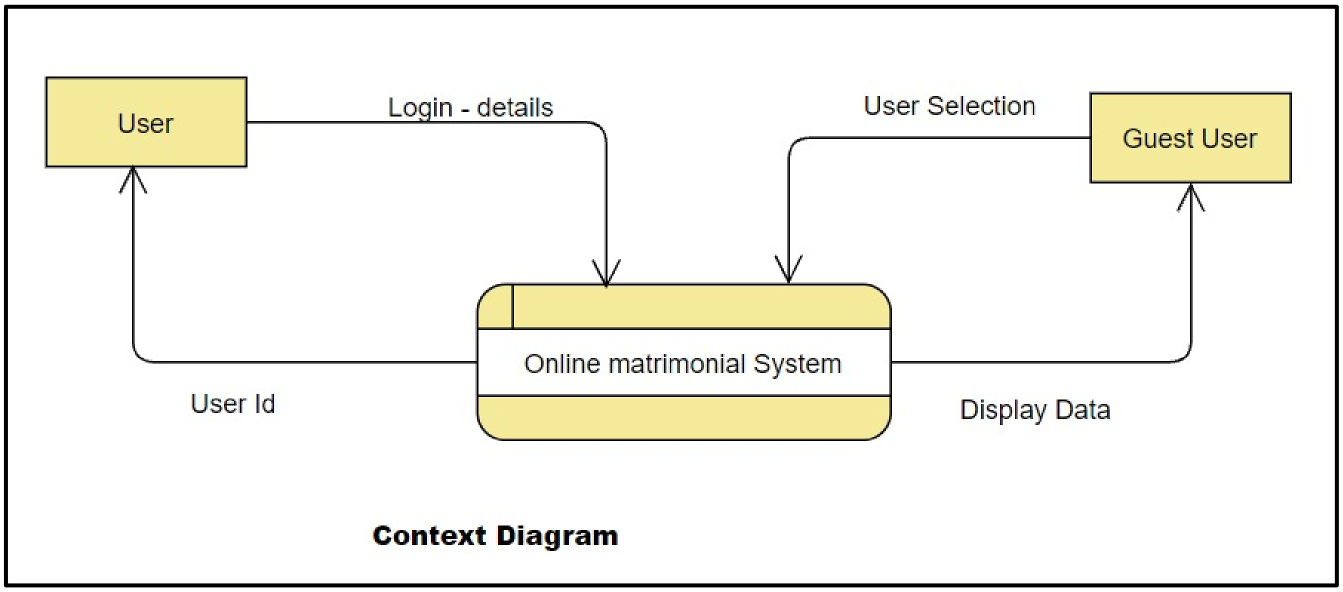
INPUT : User messages .

OUTPUT: Confirmation message.

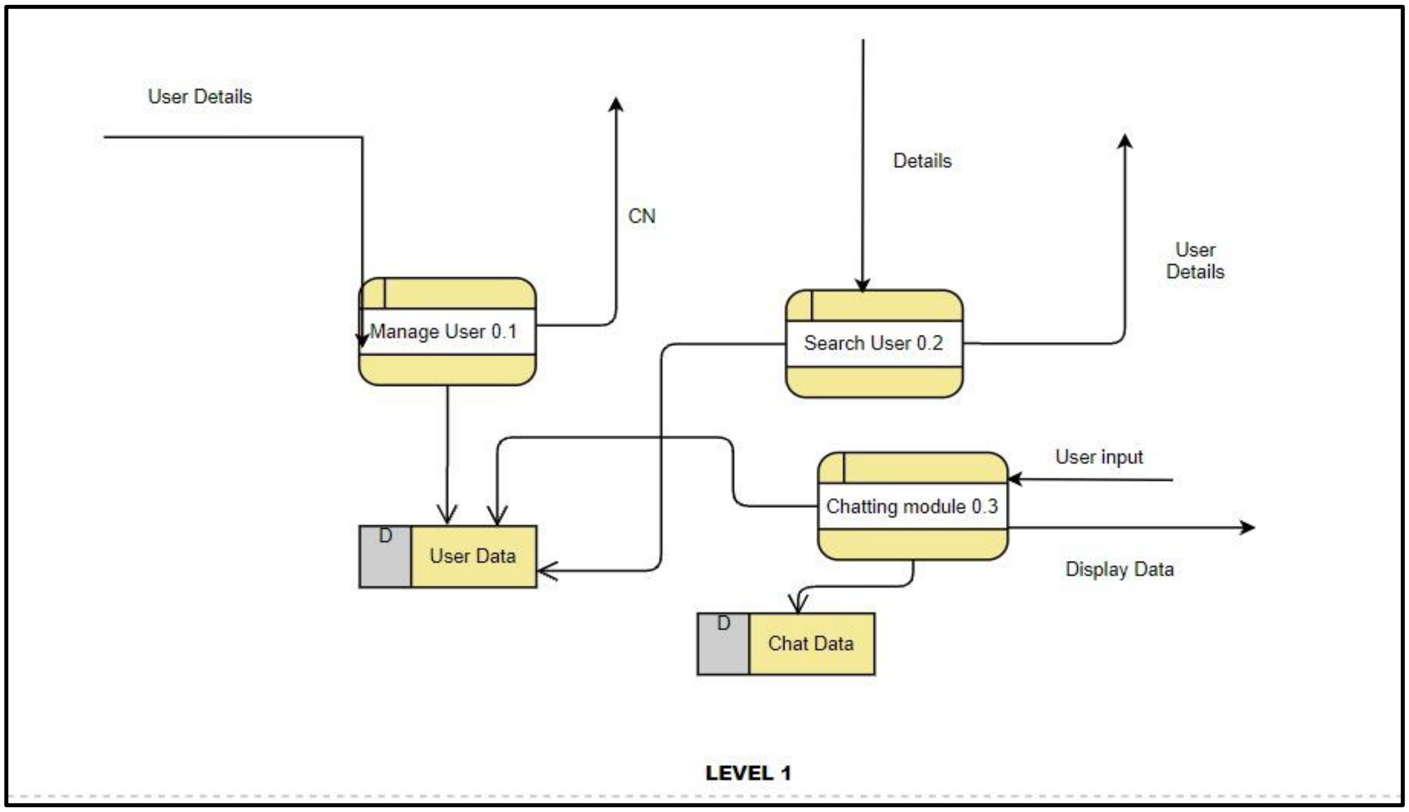
4. Design

4.1 Data Flow Diagram

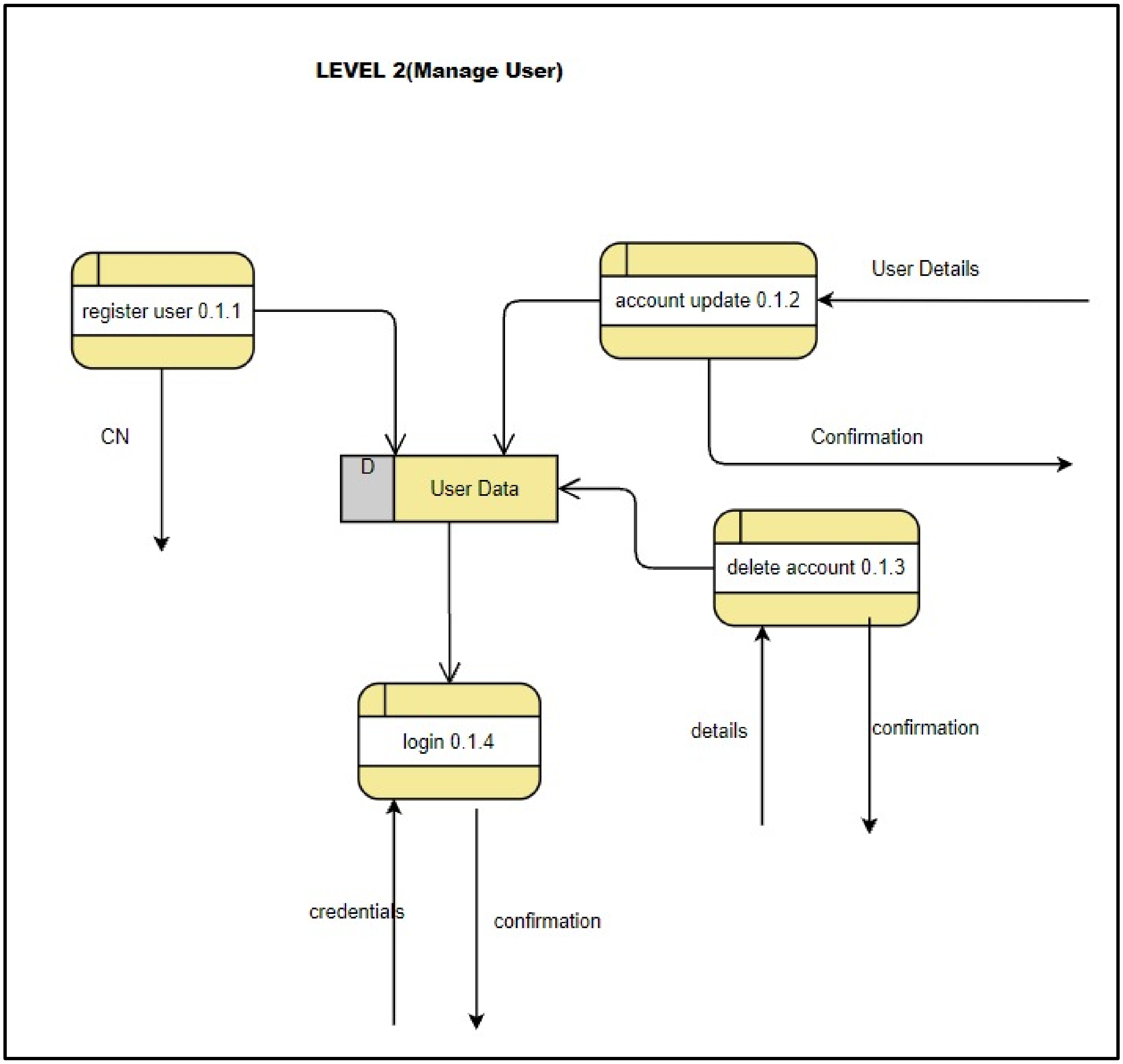
4.1.1 Context Diagram for Matrimonial Website



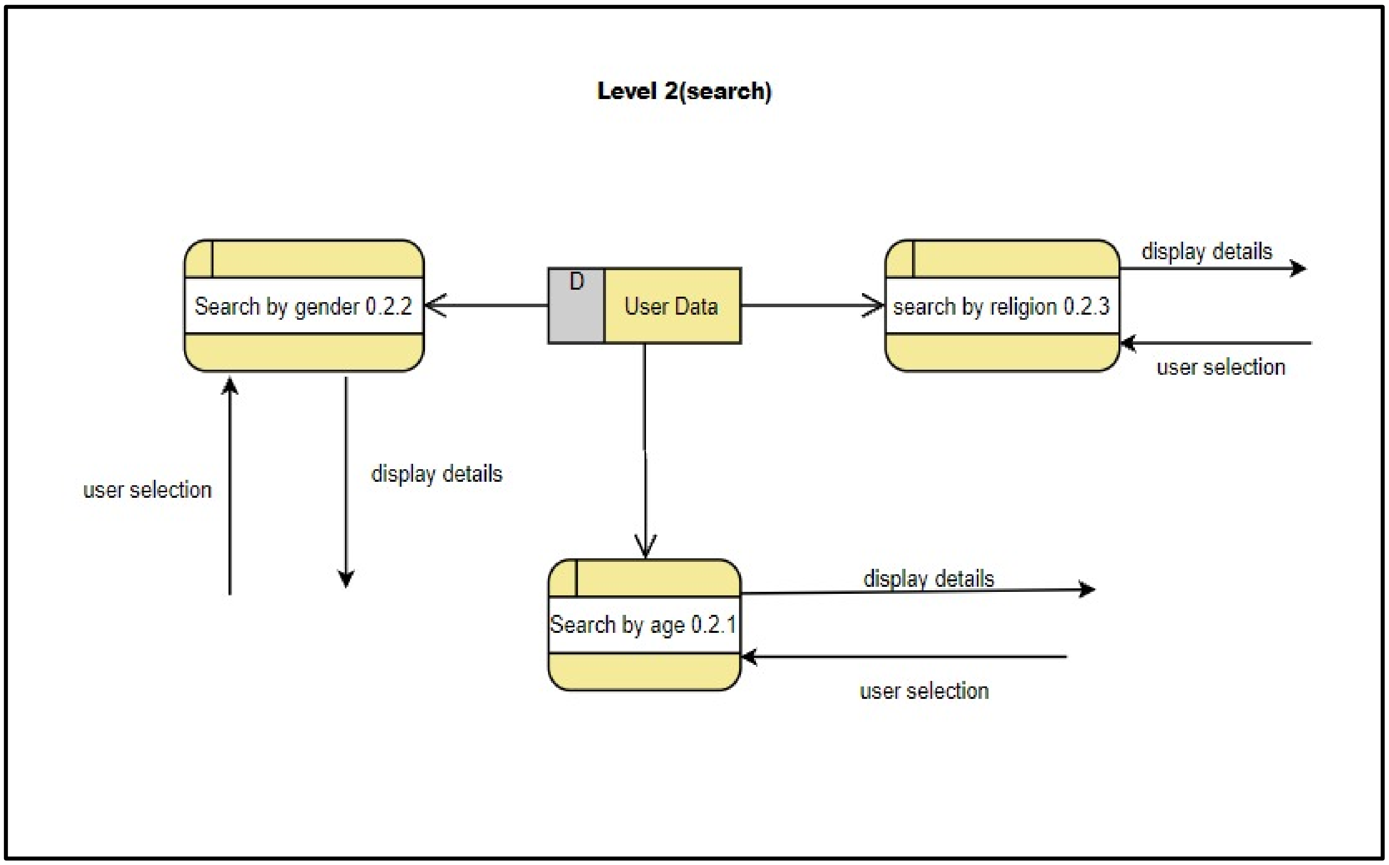
4.1.2 Level-1 Diagram for Matrimonial Website



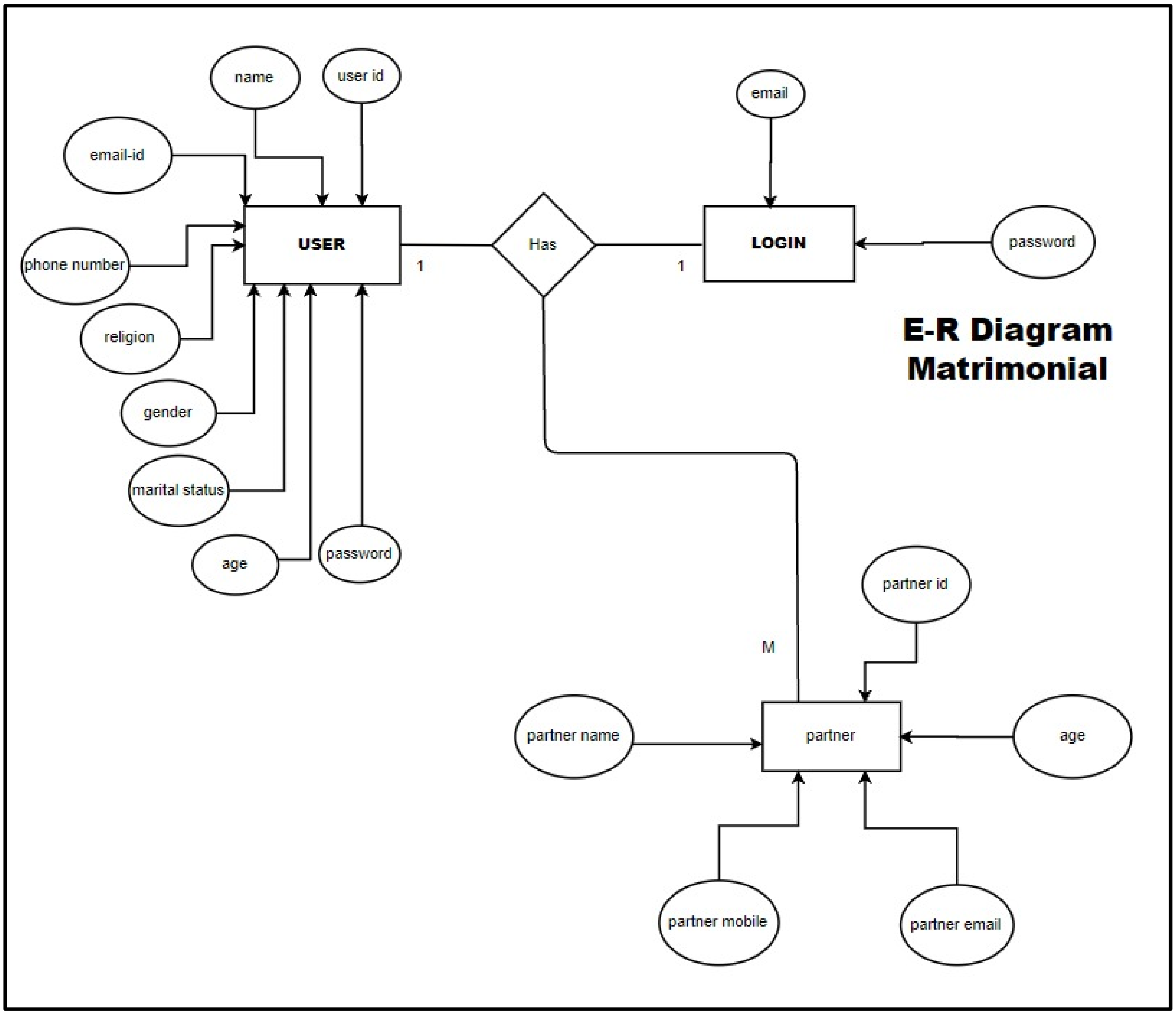
4.1.3 Level-2 Diagram for Matrimonial Website



4.1.4 Level-2 Diagram for Matrimonial Website



4.2 E-R Diagram



4.3 Data Dictionary

User Schema			
Sr no.	Field Name	Data Type	Description
1	id	Object ID	Unique id generated by mongo db
2	name	String	-
3	Email	String	-
4	Password	String	Hashed password
5	Phone number	Integer	-
6	Gender	String	-
7	Date of birth	Date	-
8	Age	String	-
9	Marital Status	String	-
10	Mother tongue	String	-
11	Religion	string	-
12	City	String	-
13	Pincode	String	-
14	Tokens	String	Use to store in cookies for authentication and authorization

5. Implementation Details

The system consists of 3 basic modules namely

1. User Module
2. Search Module
3. Partner Module

Each module consists of several methods to implement the required functionality. Implementation is done using Node JS and front end is done using React. Database used in these modules is MongoDB.

User Module

This module allows user to become a new member in the portal. Customer is required to fill necessary details related for Registration. After successful registration user can login with their email and password. User can also view their details on their profile and can edit them too. User can also delete the profile permanently.

Search Module

Any user i.e. Registered or Unregistered/Guest user can search based on criteria and can find the matches based on the criteria selected.

Partner Module

Registered users have access to the features like **send request** to other users, **connect** with them and can **chat** with partner.

5.1 Function prototypes

5.1.1 User Registration

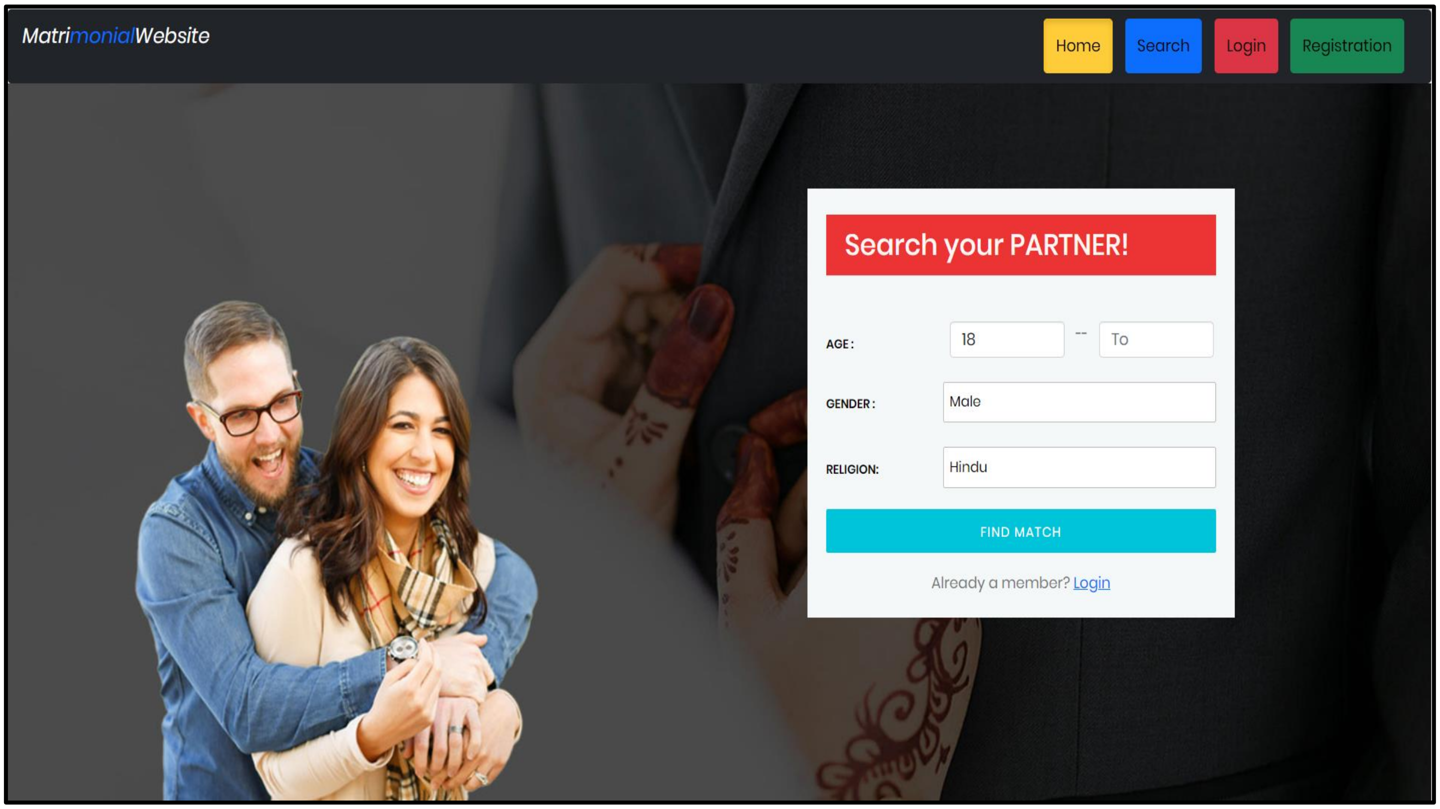
```
router.post('/register', upload.single('profile'), async (req, res) => {
    const { name, age, dob, gender, phone, email, password, marital_status,
mother_tongue, religion, city, pincode } = req.body;
    let profile = (req.file) ? req.file.filename : null;
    var date = new Date(dob);

    const modified_dob = date.toISOString().split('T')[0];

    if (!name || !email || !phone || !password || !gender || !age || !dob ||
!marital_status || !mother_tongue || !religion || !city || !pincode) {
        console.log("hi");
        return res.status(422).json({ error: "Please fill all fields properly." })
    }
    try {
        const userExist = await User.findOne({ email: email });
        if (userExist) {
            return res.status(422).json({ error: "User with entered email already
exist." })
        }
        else {
            const user = new User({ name, age, dob: modified_dob, gender, phone,
email, password, profile, marital_status, mother_tongue, religion, city, pincode });
            const userRegistration = await user.save();
            if (userRegistration) {
                res.status(201).send({ message: "User registered successfully." })
            }
            else {
                res.status(500).send({ error: "Registration failed due to some
internal error.Plz try again" })
            }
        }
    }
    catch (err) {
        console.log(err);
    }

});
```

5.1.2 Search For Partner based on criteria

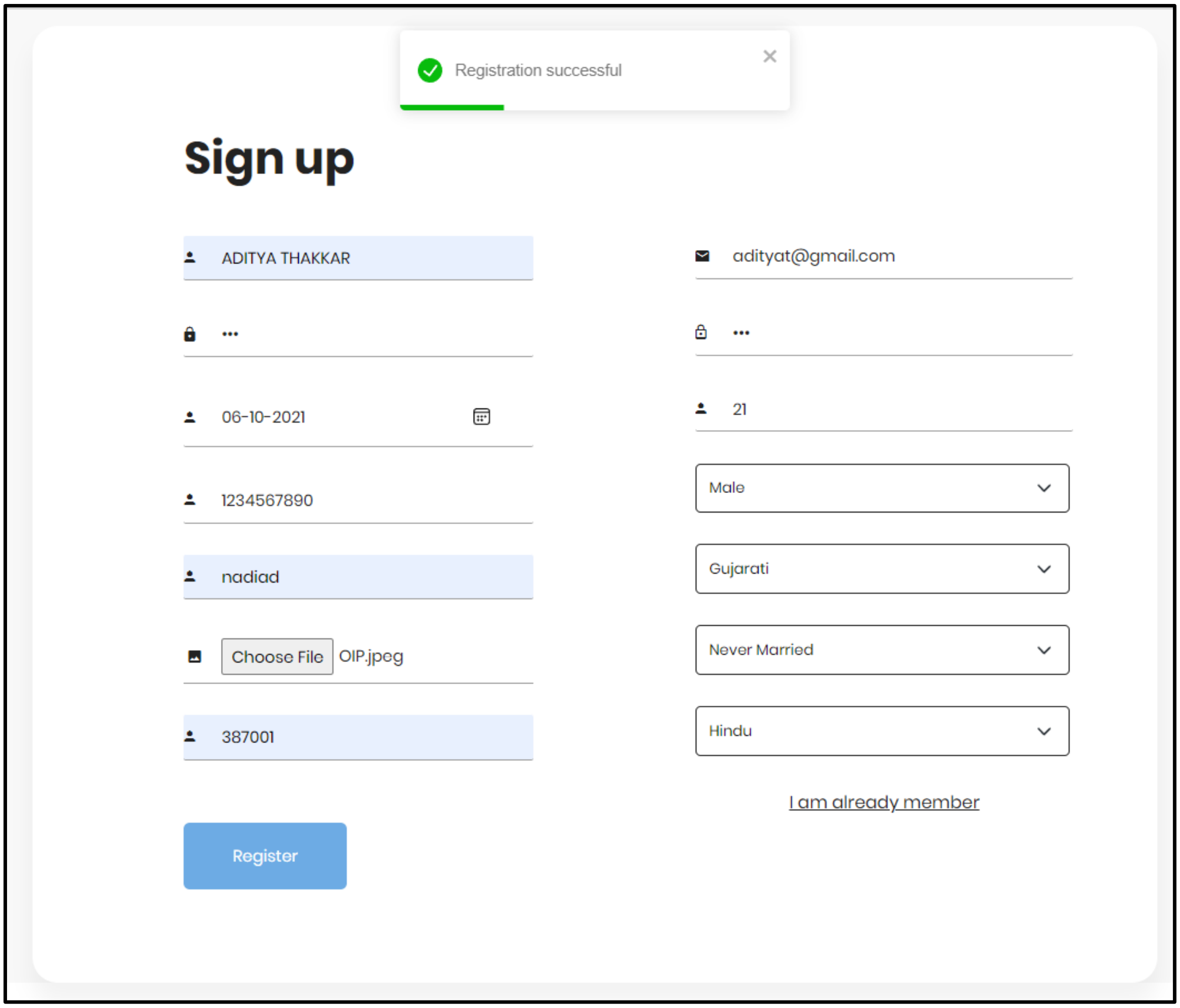
```
router.post('/filter', async (req, res) => {
    const gtage = req.body.gtage;
    const ltage = req.body.ltage;
    const gender = req.body.gender;
    const religion = req.body.religion;
    console.log(gtage + ltage + gender + religion);

    let fltparam;

    if (gtage && ltage && gender && religion) {
        fltparam = { $and: [{ age: { $gt: gtage } }, { age: { $lt: ltage } }, { gender
}, { religion }] }
    }
    else if (gtage && gender && religion && !ltage) {
        fltparam = { $and: [{ age: { $gt: gtage } }, { gender }, { religion }] }
    }
    try {

        const user = await User.find(fltparam);
        console.log(user);
        if (user.length != 0) {
            return res.status(201).json(user);
        }
        return res.status(422).json({ error: "No matches found." })
    }
    catch (err) {
        console.log(err);
    }
});
```

6. Screenshots

Home Page



User Registration



Registration successful

Sign up

ADITYA THAKKAR

•••

06-10-2021

1234567890

nadiad

Choose File OIP.jpeg

387001

adityat@gmail.com

•••

21

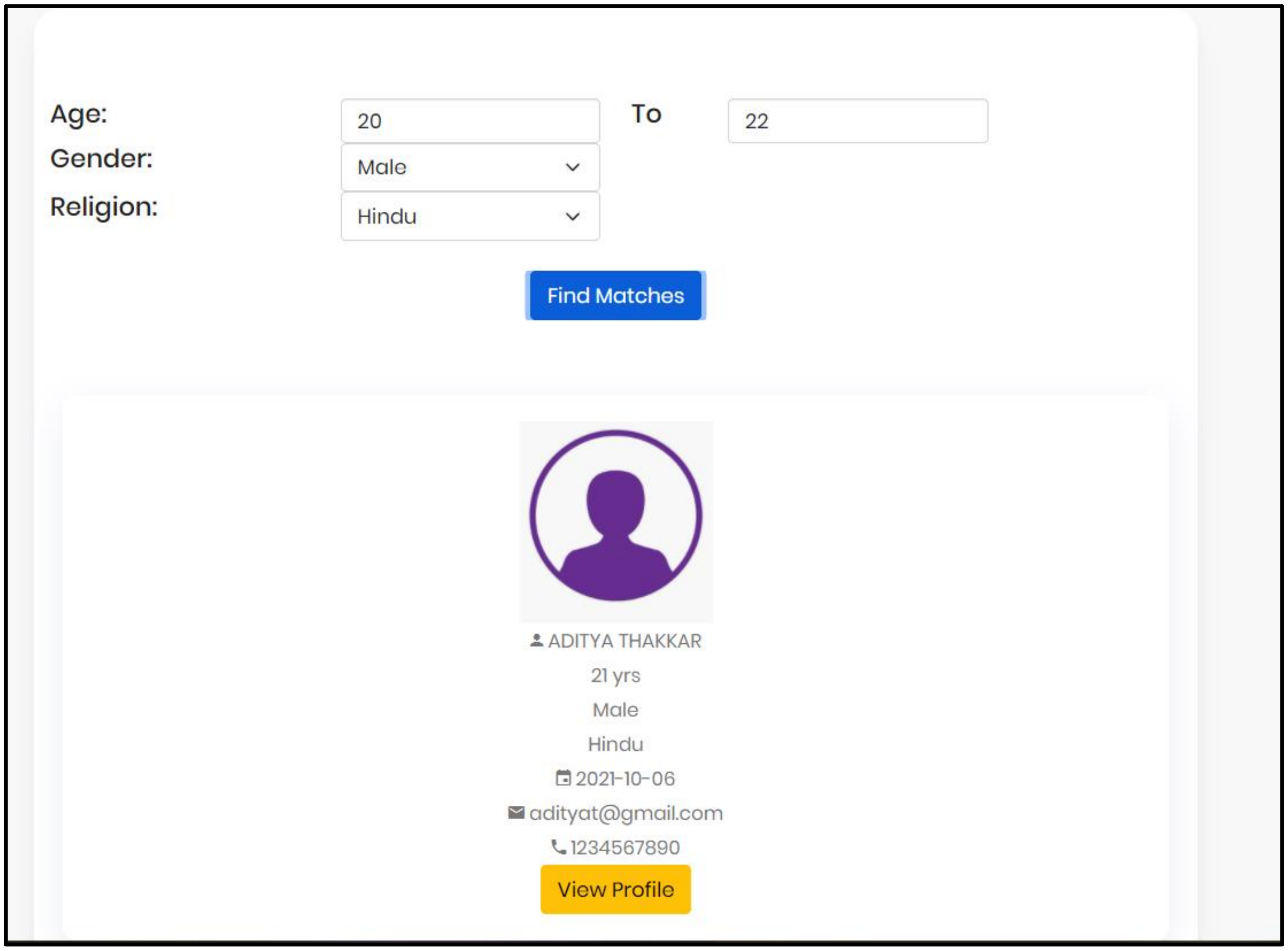
Male

Gujarati

Never Married

Hindu

Search Feature



Age: 20 To 22

Gender: Male

Religion: Hindu

ADITYA THAKKAR

21 yrs

Male

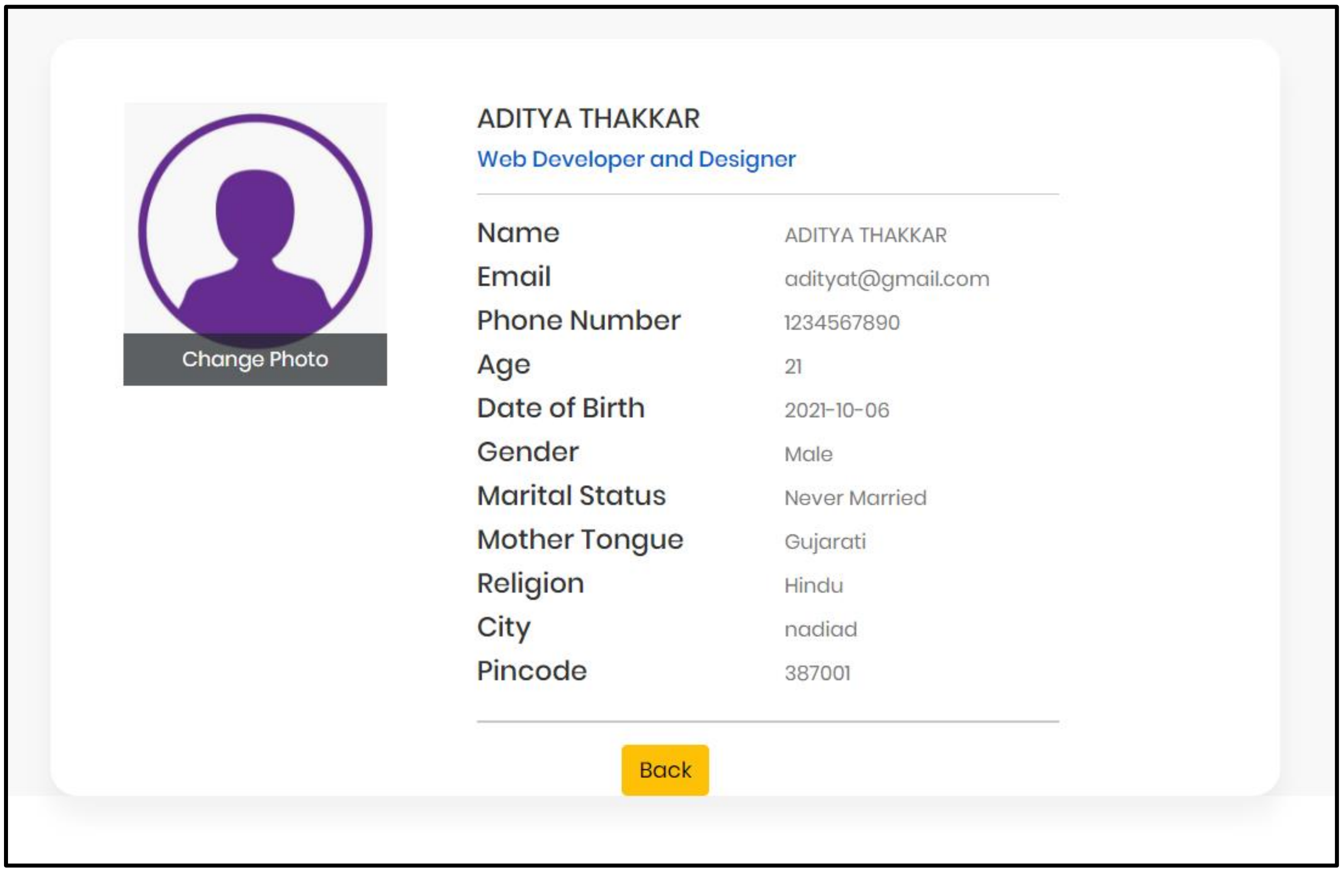
Hindu

2021-10-06

adityat@gmail.com

1234567890

View Profile of other Users(Only Registered Users)



ADITYA THAKKAR

Web Developer and Designer

Name	ADITYA THAKKAR
Email	adityat@gmail.com
Phone Number	1234567890
Age	21
Date of Birth	2021-10-06
Gender	Male
Marital Status	Never Married
Mother Tongue	Gujarati
Religion	Hindu
City	nadiad
Pincode	387001

7. Conclusion

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

- User Registration
- Login
- User authentication
- Logout
- Manage Profile
- Search based on criteria
- View Partner based on matches found
- Delete Profile

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

8. Limitations and Future Enhancements

We are able to implement the functionality model of the "Matrimonial Website". We aim to make this product ready to be used in live markets.

Limitations:

1) Currently the user can only view the profile of other users, they are unable to chat with others.
2) There is no personal validation and verification of the user because there is no administration involved.

Future Enhancements:

1) The project can be extended to support **chat feature** which help the user to know more about their selected partner.
2) We can add a **Friend-Request** feature with helps user to select their interested partners.
3) There can be administration for validation and verification of users in future version.

We can enhance more as per future requirements.

9. Bibliography

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

- https://stackoverflow.com/

- React official site:

  https://reactjs.org/

- Express JS official site:

  https://expressjs.com/

- npm:

  https://www.npmjs.com/

- mongoose docs:

  https://mongoosejs.com/docs/

- mongo DB

  https://www.mongodb.com/