

DOST (Friend Finder)

*A Project Report submitted in partial fulfilment of the requirements for the
award of the degree of*

Bachelor of Technology

In

Computer Science and Engineering

(Hons.)

By

Name of Students :-

Luvkush Sharma (2115800015)

Dheeraj Sharma (2115800009)

Group No. :- 04

Under the Guidance of

Mr. Saksham Mishra

Department of Computer Engineering & Applications

Institute of Engineering & Technology



GLA University

Mathura- 281406, INDIA

May 2024



**Department of computer Engineering and
Applications**

GLA University, Mathura

17 km. Stone NH#2, Mathura-Delhi Road, P.O. – Chaumuha

Mathura – 281406

Declaration

Certified that this project report “**DOST (Friend Finder)**” is the bonafide work of “**Luvkush Sharma, Dheeraj Sharma**” who carried out the project work under our supervision.

(Signature of Mentor)

Acknowledgement

It gives us a great sense of pleasure to present the report of the BTech(H) project undertaken during the **B.Tech.(Hons.)CS III Year**. This project is going to be an acknowledgment of the inspiration, drive, and technical assistance that will be contributed to it by many individuals.

We owe a special debt of gratitude to **MR. SAKSHAM MISHRA**, for providing us with an encouraging platform to develop this project, which thus helped us in shaping our abilities towards a constructive goal, and for his constant support and guidance to our work. His sincerity, thoroughness, and perseverance have been a constant source of inspiration for us. We believe that he will shower us with all his extensively experienced ideas and insightful comments at different stages of the project & also teach us about the latest industry-oriented technologies.

We also do not like miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind guidance and cooperation.

GROUP MEMBERS NAMES :

LUVKUSH SHARMA

2115800015(13)

DHEERAJ SHARMA

2115800009(08)

Abstract

In today's fast-paced digital world, social connections play a crucial role in our well-being and personal growth. The DOST application emerges as a revolutionary platform designed to facilitate meaningful connections and foster friendships in an increasingly interconnected society. Drawing inspiration from its name, which means "friend" in several languages, DOST aims to break down geographical barriers and connect individuals with like-minded friends, irrespective of location.

Key features of the DOST application include a user-friendly interface, matchmaking algorithm, real-time messaging functionality, personalized event recommendations, and robust privacy controls. These features collectively contribute to creating a vibrant and inclusive community where users can explore common interests, engage in meaningful conversations, and forge lasting friendships.

By leveraging cutting-edge technology and a user-centric approach, DOST empowers individuals to expand their social networks, combat social isolation, and enhance their overall well-being. Through its innovative features and commitment to fostering genuine connections, DOST strives to redefine the way people connect and build friendships in the digital age.

In an age where digital interfaces have become the primary mode of communication, DOST serves as a testament to the enduring power of human connection. As we navigate the complexities of modern life, DOST stands as a steadfast companion, dedicated to fostering friendships that transcend the virtual realm and leave a lasting impact on the lives of its users.

In summary, the DOST application represents a paradigm shift in the way individuals approach social interactions in the digital era. By leveraging technology to facilitate authentic connections, DOST stands poised to revolutionize the landscape of online friendship, one meaningful connection at a time.

Content

1. CHAPTER 1 :- Introduction.....	1-6
1.1 Overview & Motivation.....	1
1.2 Objective.....	2
1.3 Survey of Similar Application.....	3-5
1.4 Organization of the Project.....	6
2. CHAPTER 2 :- Software Requirement Analysis.....	7-8
2.1 Functional Requirements.....	7
2.2 Non-Functional Requirements.....	7
2.3 Technical Requirements.....	7
2.4 Technical Feasibility.....	8
3. CHAPTER 3 :- Implementation and User Interface.....	9-18
3.1 Chat Model.js.....	9
3.2 Friend Request Model.js.....	9-10
3.3 OTP Model.js.....	10
3.4 User Model.js.....	10-14
4. CHAPTER 4 :- Software Testing.....	19
5. CHAPTER 5 :- Conclusion.....	20
6. CHAPTER 6 :- Summary.....	21
Reference.....	22

CHAPTER 1 :- Introduction

1.1 Overview :- In today's fast-paced digital world, social connections play a vital role in our well-being and personal growth. As people increasingly turn to technology to meet new friends and expand their social circles, the Dost application emerges as a revolutionary Friend Finder platform designed to foster meaningful connections and build lasting friendships.

Dost, meaning “friend” in several languages, encapsulates the essence of its mission — to be a reliable companion in helping individuals discover like-minded friends, irrespective of geographical boundaries. This application is not merely about expanding one's social network but about creating genuine connections that enrich lives and provide a sense of belonging in an interconnected world.

Motivation :- . The motivation behind the DOST project stems from a deep-seated understanding of the human need for connection and belonging. In an era characterized by digital communication and virtual interactions, genuine human connections have become increasingly elusive. We envision DOST as a solution to this challenge, providing users with a platform where they can discover like-minded individuals, engage in meaningful conversations, and forge lasting friendships.

Moreover, the prevalence of social isolation and loneliness in today's society further underscores the importance of initiatives like DOST. By creating a welcoming and inclusive space for individuals from all walks of life, we aim to alleviate feelings of loneliness and foster a sense of community among our users. Through DOST, we aspire to empower individuals to build meaningful connections, enrich their lives, and ultimately, experience the joy of genuine friendship in the digital age.

In the contemporary digital landscape, where social interactions are often confined to virtual platforms, the significance of fostering genuine connections and meaningful friendships becomes increasingly vital. The Dost project emerges as a response to the inherent challenges and limitations present in existing social networking platforms, offering a unique and transformative approach to the way individuals connect and build relationships.

1.2 Objective :-

- i. **Responsive Design for All Devices:** Ensure that the DOST application is easily accessible and user-friendly across various devices, including smartphones, tablets, and desktops.
- ii. **Enable users to find others:** With similar preferences and experiences. This facilitates meaningful connections and builds a more engaging community within your application.
- iii. **User-Friendly Chat Features:** Implement intuitive chat features to facilitate seamless communication and interaction between users.
- iv. **Simplified User Feedback System:** Create a straightforward and user-friendly feedback system to encourage user input and suggestions.
- v. **Enable Friend Requests:** Implement a friend request system to allow users to send and receive friend requests, facilitating connections between users.
- vi. **Build User Profiles:** Create a user profile system where users can input and update their personal details, including favourite programming languages, frameworks, libraries, and hackathon experiences.
- vii. **Implement User Authentication:** Develop a secure authentication system to allow users to sign up, log in, and manage their accounts securely.
- viii. **Favourite Tech Stack Selection:** Allow users to specify their favourite programming languages, frameworks, and libraries during the signup process.
- ix. **Security Measures:** Implement basic security practices such as input validation, password hashing, and protecting against common web vulnerabilities.
- x. **Database Connectivity:** Establish a connection between the application and a MongoDB database using Mongoose, allowing seamless storage and retrieval of user data.

1.3 Survey of Similar Application :- The fast pace of today's world can make it quite challenging to find a friendship. Fortunately, the modern age has produced accessible technology designed to help us meet kindred spirits and create lasting connections at any age.

Some of the existing applications related to finding a friend are listed below :-

- i. **Televeda:-** Televeda is an award-winning platform that's created a friendly, safe, and welcoming community for lifelong learners. We make it simple and easy to find expert workshops and events hosted by local and national community centers. Televeda is where older adults go when they want more perspectives, more exploration, and more connections in a like-minded community.

Target Audience: Older adults and lifelong learners.

Key Features:

Virtual classes, workshops, and events.

Safe and welcoming community for exploration and connections.

Community Centric.

- ii. **Meetup:** Meetup is a social networking platform that focuses on using shared interests as a basis for facilitating in-person gatherings and events. The app has millions of members across the globe, allowing users with similar interests to meet up at local events, gatherings, and social activities. It is up to users to then build a friendship off of these connections.

Target Audience: Users with diverse interests.

Key Features:

Social networking based on shared interests.

In-person gatherings and events.

User-friendly interface with extensive search options.

- iii. **Bumble BFF:** A branch of Bumble, a popular dating app, Bumble BFF is an app designed for individuals seeking friendships. Users build a profile that exhibits their interests and personality, which other users then get to observe. Similar to Bumble, on Bumble BFF, users then swipe left or right—no or yes—on the profiles that Bumble BFF sends their way.

Target Audience: Individuals seeking friendships.

Key Features:

Profiles showcase interests and personality.

Swiping mechanism for mutual matches.

Chatting functionality upon mutual interest.

- iv. **Nextdoor:** A branch of Bumble, a popular dating app, Bumble BFF is an app designed for individuals seeking friendships. Users build a profile that exhibits their interests and personality, which other users then get to observe. Similar to Bumble, on Bumble BFF, users then swipe left or right—no or yes—on the profiles that Bumble BFF sends their way.

Target Audience: Individuals seeking friendships.

Key Features:

Private social platform for local discussions.
Arranging meetups and events within neighbourhoods.
Features like crime alerts for enhanced community engagement.

- v. **Friender:** Combining multiple features of other popular apps, Friender matches its users based on common interests and personality traits. Users create profiles highlighting their hobbies, preferences, and personality traits and then get to browse other profiles and strike up conversation with potential friends.
Target Audience: Users seeking friends based on common interests.
Key Features:
Matching based on hobbies and personality traits.
User profiles highlight preferences and traits.
Filters and search options for specific criteria.
- vi. **Facebook Groups:** Created by Facebook, Facebook Groups are online interest-based communities where users with common interests can connect and engage in discussions. Facebook Groups are not just limited to hobbies: users can explore a vast range of groups covering professional networking, parenting, relationships, and many other aspects of life in which users may want to find fellowships.
Target Audience: Facebook users with shared interests.
Key Features:
Online interest-based communities.
Wide range of groups covering various aspects of life.
Professional networking, parenting, relationships, etc.
- vii. **Hey! VINA:** Hey! VINA is a social networking app designed exclusively for women to find new friends and expand their social circles in a woman-only setting. The app has fun quizzes and profile categories that cater to women, so users can enjoy creating a unique and in-depth profile of themselves. The app interface is swipe-based like many popular dating apps, so users can browse profiles and initiate conversation with other women.
Target Audience: Women seeking new friends.
Key Features:
Social networking exclusively for women.
Fun quizzes and profile categories.
Swipe-based interface for browsing profiles.
- viii. **Nearify:** Nearify alerts users to local events, concerts, and activities at which users can meet new people with shared interests. With its user-friendly interface and personalized recommendations, Nearify makes it easy to find exciting events related to a range of interests and hobbies.
Target Audience: Users interested in local events.
Key Features:
Alerts users to local events and activities.
Personalized recommendations based on interests.

Facilitates meeting new people with shared interests.

- ix. We3:** We3 gets a sense of users' personalities in order to match compatible users in groups of three so that users can chat and meet up. Users get started by setting up a profile consisting of their values, interests, life experiences, and more, so that the app's intelligent algorithm can match users to similar profiles. The app is specifically designed to match users in groups of three, referred to as "Tribes."
Target Audience: Users looking for group connections.
Key Features:
Personality-based matching in groups of three.
Icebreakers and guided questions for discussions.
Deep compatibility exploration through algorithm.
- x. Wink:** Another app on which users swipe left or right on profiles, Wink presents user profiles who share at least one common interest. With Wink, users can create profiles, browse other profiles, and initiate conversations with potential friends through text or voice chat.
Target Audience: Users looking for connections with shared interests.
Key Features:
Profiles with common interests presented for swiping.
Conversation options within Wink or via Snapchat.
Facilitates genuine connections based on interests.
- xi. LMK:** Perhaps catering more towards extroverts, LMK is a social networking app that prompts users to text and join voice chats within the app so that users can engage in verbal discussion without sharing phone numbers. This app can be a great option for those who prefer phone calls to texting.
Target Audience: Social networking for engaging discussions.
Key Features:
App prompts users to text and join voice chats.
Verbal discussions without sharing phone numbers.
Real-time interactions for extroverted individuals.
- xii. Hoop:** Hoop, also connected with Snapchat, helps users meet others with shared interests and activities. This social networking app, like many others, allows users to make new friends and connect with people worldwide through a swipe-based interface. The app also offers users the ability to chat and video call so users can expand their social circles.
Target Audience: Users looking to connect with shared interests.
Key Features:
Connected with Snapchat for expanded interactions.
Swiping interface for making new friends.
Chat and video call features for communication.

1.4 Organization of the Project :-

i. Project Management :-

- **Project Kickoff :-** Conduct a kickoff meeting to discuss project goals, objectives, and timelines. Assign roles and responsibilities to team members.
- **Project Planning :-** Develop a comprehensive project plan outlining tasks, milestones, and deadlines. Create a communication plan to ensure effective collaboration among team members.
- **Progress Monitoring :-** Regularly monitor project progress, identify any potential risks or issues, and implement mitigation strategies as needed.

ii. Development :-

- **Backend Development :-** Develop the backend infrastructure of the DOST application, including database design, server setup, and API development.
- **Frontend Development :-** Design and develop the user interface of the DOST application, focusing on usability, accessibility, and responsiveness across different devices.
- **Integration :-** Integrate third-party services and APIs, such as authentication services and email notifications, to enhance the functionality of the DOST application.

iii. Deployment :-

- **Staging Environment Setup :-** Configure a staging environment to deploy and test the DOST application before final deployment.
- **Production Deployment :-** Deploy the DOST application to a production environment, ensuring seamless transition and minimal downtime.

iv. Support :-

- **Customer Support :-** Establish a support system to address user inquiries, troubleshoot issues, and provide assistance as needed.
- **Feedback Collection :-** Collect feedback from users to identify areas for improvement and inform future iterations of the DOST application.

CHAPTER 2 :- Software Requirement Analysis

2.1 Functional Requirements :-

- i. **User Registration and Authentication :-** Users should be able to create accounts, log in securely, and manage their profiles.
- ii. **Profile Creation :-** Users should have the ability to create detailed profiles, including personal information, interests, and preferences.
- iii. **Messaging System :-** Provide a messaging system for users to communicate with each other securely within the application.
- iv. **Matching Algorithm :-** Implement a sophisticated matching algorithm to suggest potential friends based on user profiles, interests, and preferences. The algorithm should take into account factors like mutual interests.
- v. **Privacy Controls :-** Implement robust privacy controls to allow users to manage their privacy settings. Users should be able to control who can send them messages.

2.2 Non-Functional Requirements :-

- i. **Security :-** Ensure the security of user data through encryption, secure authentication mechanisms, and adherence to data protection regulations.
- ii. **Usability :-** Design an intuitive and user-friendly interface that is easy to navigate and accessible to users of all ages and technological proficiency levels.
- iii. **Compatibility :-** Ensure compatibility with various devices, operating systems, and web browsers.
- iv. **Accessibility :-** Ensures that all sections of users can access our application.
- v. **Performance :-** Optimize application performance to ensure fast response times, minimal latency, and scalability.

2.3 Technical Requirements :-

- i. **Backend Framework :-** Choose a suitable backend framework for developing the application's server-side logic, database management, and API development.
- ii. **Frontend Framework :-** Select a frontend framework for building the user interface of the application, ensuring responsiveness and cross-browser compatibility.
- iii. **Database :-** Choose a reliable and scalable database solution for storing user data, preferences, messages, and event information securely.
- iv. **Hosting :-** Select a hosting provider and infrastructure that can accommodate the application's scalability, performance, and security requirements.
- v. **Messaging Service :-** Integrate a messaging service or API for real-time communication between users within the application.

2.4 Technical Feasibility :-

- i. **MERN Stack :-** The MERN stack (MongoDB, Express.js, React.js, Node.js) is well-suited for developing dynamic and interactive web applications. MongoDB provides a flexible and scalable NoSQL database solution, while Express.js and Node.js enable server-side logic and API development. React.js facilitates the creation of responsive and user-friendly frontend interfaces.
- ii. **Socket.IO :-** Socket.IO is a JavaScript library that enables real-time, bidirectional communication between web clients and servers. It is ideal for implementing features like instant messaging and live updates in the DOST application.
- iii. **Server-Side Rendering :-** The MERN stack offers performance optimizations such as server-side rendering (SSR) with React.js, caching mechanisms with Express.js, and asynchronous I/O operations with Node.js. These optimizations help minimize latency, improve page load times, and enhance overall application performance.
- iv. **Security :-** The DOST application can implement security measures such as data encryption, input validation, authentication, and authorization using middleware and libraries available in the MERN stack ecosystem. Socket.IO supports secure WebSocket connections over HTTPS and enables implementation of authentication mechanisms and message encryption to ensure data privacy and integrity.
- v. **Resources :-** The MERN stack has a large and active community of developers, extensive documentation, and a wide range of libraries and frameworks available for building web applications. This ensures ample availability of resources, tutorials, and support for development and maintenance of the DOST application. Socket.IO is a widely adopted library with comprehensive documentation, community forums, and online resources available for developers. This facilitates the integration and implementation of real-time communication features in the DOST application.

CHAPTER 3 :- Implementation and User Interface

3.1 Chat Model.js :-

```
const mongoose = require('mongoose');
const chatSchema = new mongoose.Schema({
  sender: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true
  },
  receiver: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true
  },
  message: {
    type: String,
    required: true
  },
  timestamp: {
    type: Date,
    default: Date.now
  }
});

const Chat = mongoose.model('Chat', chatSchema);
module.exports = Chat;
```

3.2 Friend Request Model.js :-

```
const mongoose = require('mongoose');
const friendRequestSchema = new mongoose.Schema({
  recipient: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true,
  },
  sender: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'User',
    required: true,
  },
  status: {
    type: String,
    enum: ['accepted', 'pending'],
    default: 'pending',
  },
});
```

```
}, { timestamps: true });
```

```
const FriendRequest = mongoose.model('FriendRequest', friendRequestSchema);  
module.exports = FriendRequest;
```

3.3 OTP Model.js :-

```
const mongoose = require("mongoose");  
const otpSchema = new mongoose.Schema({  
  user: {  
    type: mongoose.Schema.Types.ObjectId,  
    ref: "User",  
  },  
  phone: {  
    type: String,  
    required: [true, "Phone number is required"],  
  },  
  otp: {  
    type: String,  
    required: [true, "OTP is required"],  
  },  
  otpExpiration: {  
    type: Date,  
    required: [true, "ExpiresAt is required"],  
  },  
});  
const OTP = mongoose.model("OTP", otpSchema);  
module.exports = OTP;
```

3.4 User Model.js :-

```
const mongoose = require("mongoose");  
const validator = require("validator");  
const bcrypt = require("bcryptjs");  
const crypto = require("crypto");  
const userSchema = new mongoose.Schema({  
  name: {  
    type: String,  
    required: [true, "Please tell us your name!"],  
  },  
  email: {  
    type: String,  
    required: [true, "Please provide your email"],  
    unique: true,  
    lowercase: true,  
    validate: [validator.isEmail, "Please provide a valid email"],  
  },  
  photo: String,
```

```

password: {
  type: String,
  required: [true, "Please provide a password"],
  minlength: 8,
  select: false,
},
passwordConfirm: {
  type: String,
  required: [true, "Please confirm your password"],
  validate: {
    validator: function (pc) {
      return this.password === pc;
    },
    message: "Password and ConfirmPassword are not the same",
  },
},
programmingLanguages: {
  type: [String],
  required: [
    true,
    "Please provide programming languages in which you feel comfortable.",
  ],
  enum: [
    "JavaScript",
    "Python",
    "Java",
    "C/C++",
    "C#",
    "PHP",
    "Dart",
    "Other",
  ],

  default: [], // Allow empty array
  validate: {
    validator: (v) => Array.isArray(v),
    message: "Programming languages must be an array of strings",
  },
},

frameworks: {
  type: [String],
  enum: ["React", "Angular", "Vue.js", "Express", "Django", "Flask", "Other"],

  default: [], // Allow empty array
  validate: {

```



```

    validator: (v) => Array.isArray(v),
    message: "Frameworks must be an array of strings",
  },
},
libraries: {
  type: [String],
  enum: [
    "Pandas",
    "Numpy",
    "Matplotlib",
    "Seaborn",
    "Scikit-learn",
    "Tensorflow",
    "Keras",
    "Pytorch",
    "React-router",
    "Material-UI",
    "Redux",
    "Axios",
    "Other",
  ],
  default: [], // Allow empty array
  validate: {
    validator: (v) => Array.isArray(v),
    message: "Frameworks must be an array of strings",
  },
},
hobbies: {
  type: [String],
  required: [true, "Please provide your hobbies"],
  trim: true,
  default: [],
},
participatedInHackathon: {
  type: Boolean,
  default: false,
},
role: {
  type: String,
  enum: ["user", "admin"],
  default: "user",
},
title: {
  type: String,
  enum: [
    "Data Scientist",

```

```

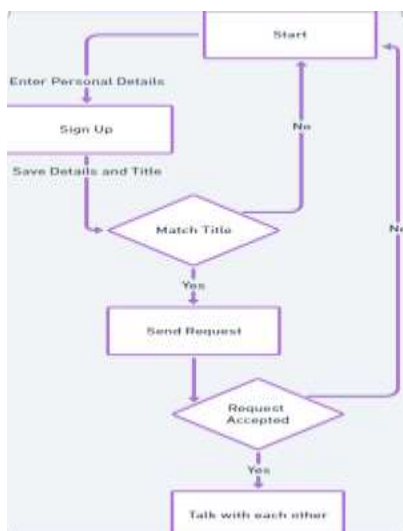
    "Full Stack Developer",
    "Frontend Developer",
    "Backend Developer",
    "Mobile App Developer",
    "DevOps Engineer",
    "UI/UX Designer",
    "QA",
    "Cloud Engineer",
    "Other",
  ],
  default: "Other",
},
cloudinaryImageUrl: {
  type: String,
  default: "https://res.cloudinary.com/dx2vel6vy/image/upload/v1710573655/default_uv0cmg.png",
},
rejectedUsers: {
  type: [mongoose.Schema.Types.ObjectId],
  ref: "User",
  default: [],
},
passwordChangedAt: Date,
passwordResetToken: String,
passwordResetExpires: Date,
active: {
  type: Boolean,
  default: true,
  select: false,
},
});
userSchema.pre("save", async function (next) {
  if (!this.isModified("password")) return next();
  if (this.isModified("rejectedUsers")) {
    return next();
  }
  this.password = await bcrypt.hash(this.password, 12);
  if (this.passwordConfirm && this.isModified('password')) {
    this.passwordConfirm = undefined;
  }
  next();
});
userSchema.pre("save", function (next) {
  if (!this.isModified("password") || this.isNew) return next();
  this.passwordChangedAt = Date.now() - 1000;
  next();
});

```

```

});
userSchema.methods.correctPassword = async function (
  candidatePassword,
  userPassword
) {
  return await bcrypt.compare(candidatePassword, userPassword);
};
userSchema.methods.changedPasswordAfter = function (JWTTimestamp) {
  if (this.passwordChangedAt) {
    const changedTimestamp = parseInt(
      this.passwordChangedAt.getTime() / 1000,
      10
    );
    return JWTTimestamp < changedTimestamp;
  }
  return false;
};
userSchema.methods.createPasswordResetToken = function () {
  const resetToken = crypto.randomBytes(32).toString("hex");
  this.passwordResetToken = crypto
    .createHash("sha256")
    .update(resetToken)
    .digest("hex");
  console.log({ resetToken }, this.passwordResetToken);
  this.passwordResetExpires = Date.now() + 1000 * 60 * 1000;
  return resetToken;
};
userSchema.pre(/^find/, function (next) {
  this.find({ active: { $ne: false } });
  next();
});
const User = mongoose.model("User", userSchema);
module.exports = User;

```



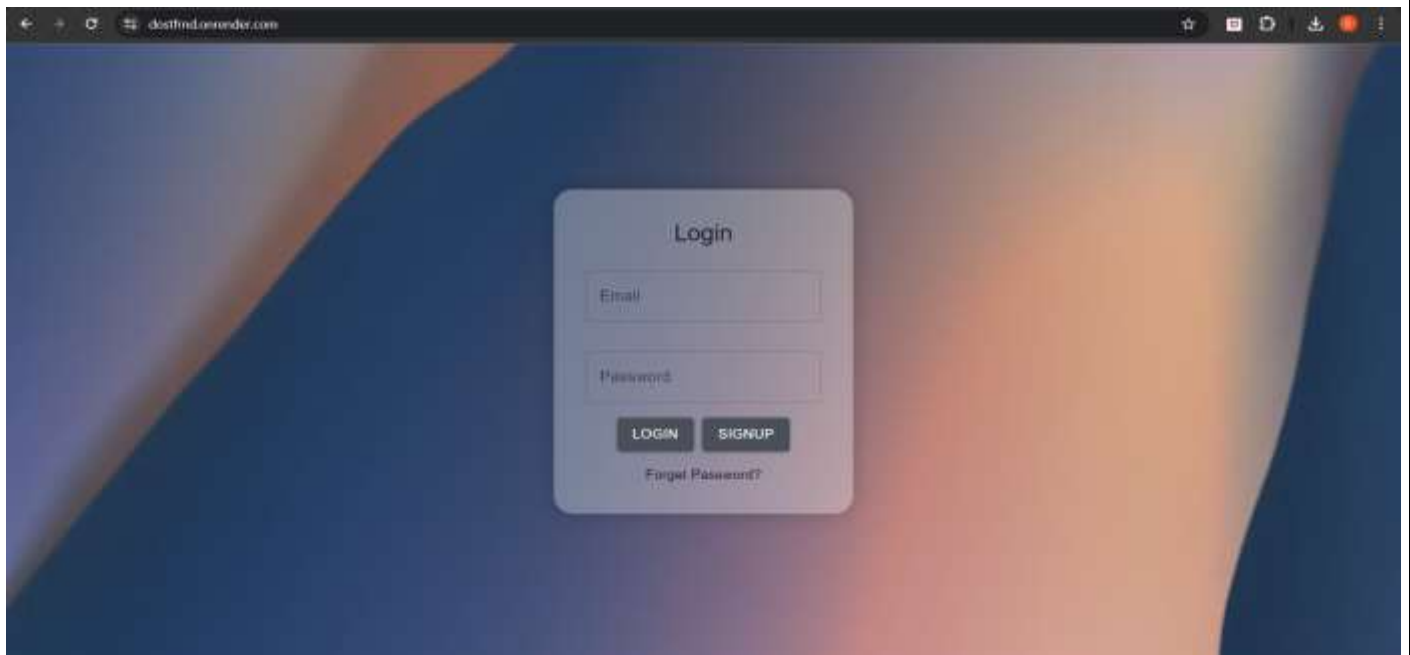


Figure 1 Login Screen

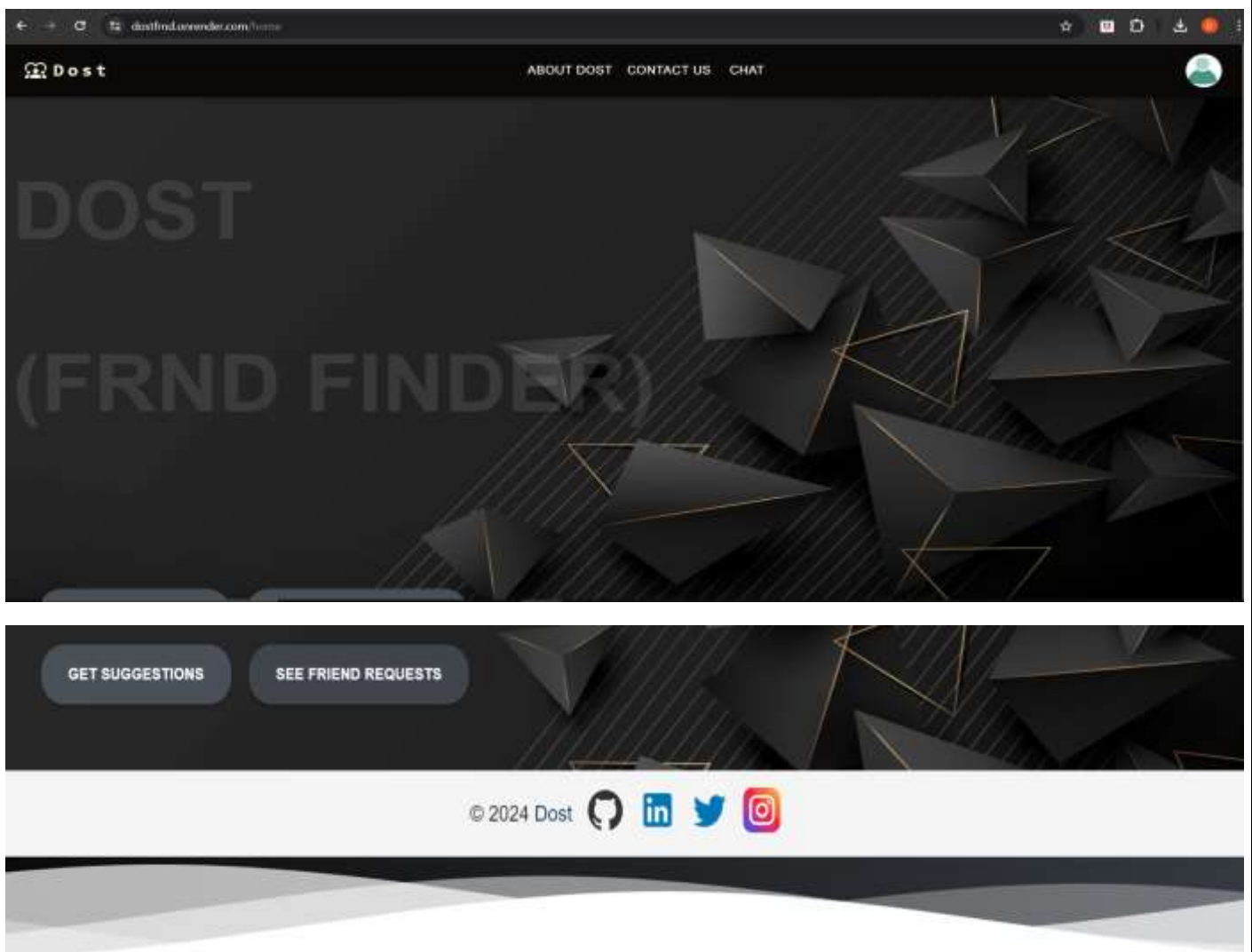
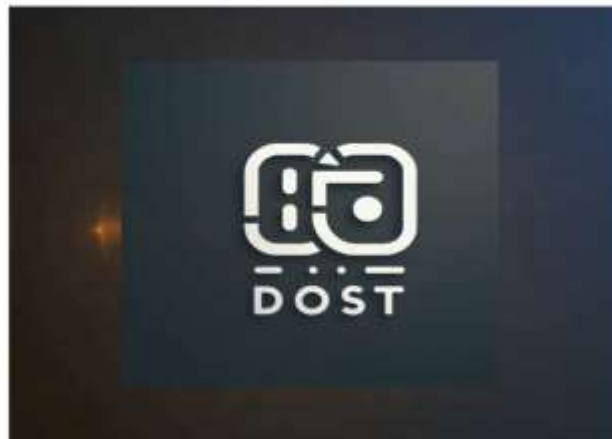


Figure 2 Home Screen



Dost: Find Your Tech Buddies!

Dost is your one-stop platform to connect with fellow tech enthusiasts. Network with developers, programmers, and like-minded individuals who share your passion for the coding world.

Whether you're a seasoned professional or just starting your coding journey, Dost helps you find friends who can motivate, collaborate, and learn together.

Why Dost?

Find Your Tribe

Connect with people who share your interests in programming languages, frameworks, and libraries.

Expand Your Network

Build meaningful connections that can lead to friendships, collaborations, or even professional opportunities.

Learn and Grow Together

Motivate and inspire each other, share knowledge, and tackle coding challenges together.

User Stories

Hear what our users are saying about Dost:

Hear what our users are saying about Dost:

"Dost helped me connect with a group of developers who share my passion for Python. We now work on freelance projects together, and it's been a fantastic learning experience!"

- Sarah, Web Developer

"I found my coding buddy on Dost! We motivate each other to stay on track and celebrate our coding milestones together. It's made learning to code so much more fun."

- John, Aspiring Programmer

Join the Dost Community!

Over 10,000 tech enthusiasts have already found their coding buddies on Dost. Sign up today and start building your network!

Over 10,000 tech enthusiasts have already found their coding buddies on Dost. Sign up today and start building your network!

[WANTS TO CREATE A NEW ACCOUNT ?](#)



Figure 3 About Us Page


Contact Us

Name*

Message*

SUBMIT

Figure 4 Contact Us Page



Nikhil Sharma
Full Stack Developer
yos8760@gmail.com

Technical Skills

Programming Languages:
JavaScriptPythonJavaDart

Frameworks:
ReactExpressFlask

Libraries:
PandasNumpyMatplotlibSeaborn

Choose FileNo file chosen

UPLOAD IMAGE

Figure 5 Profile Page

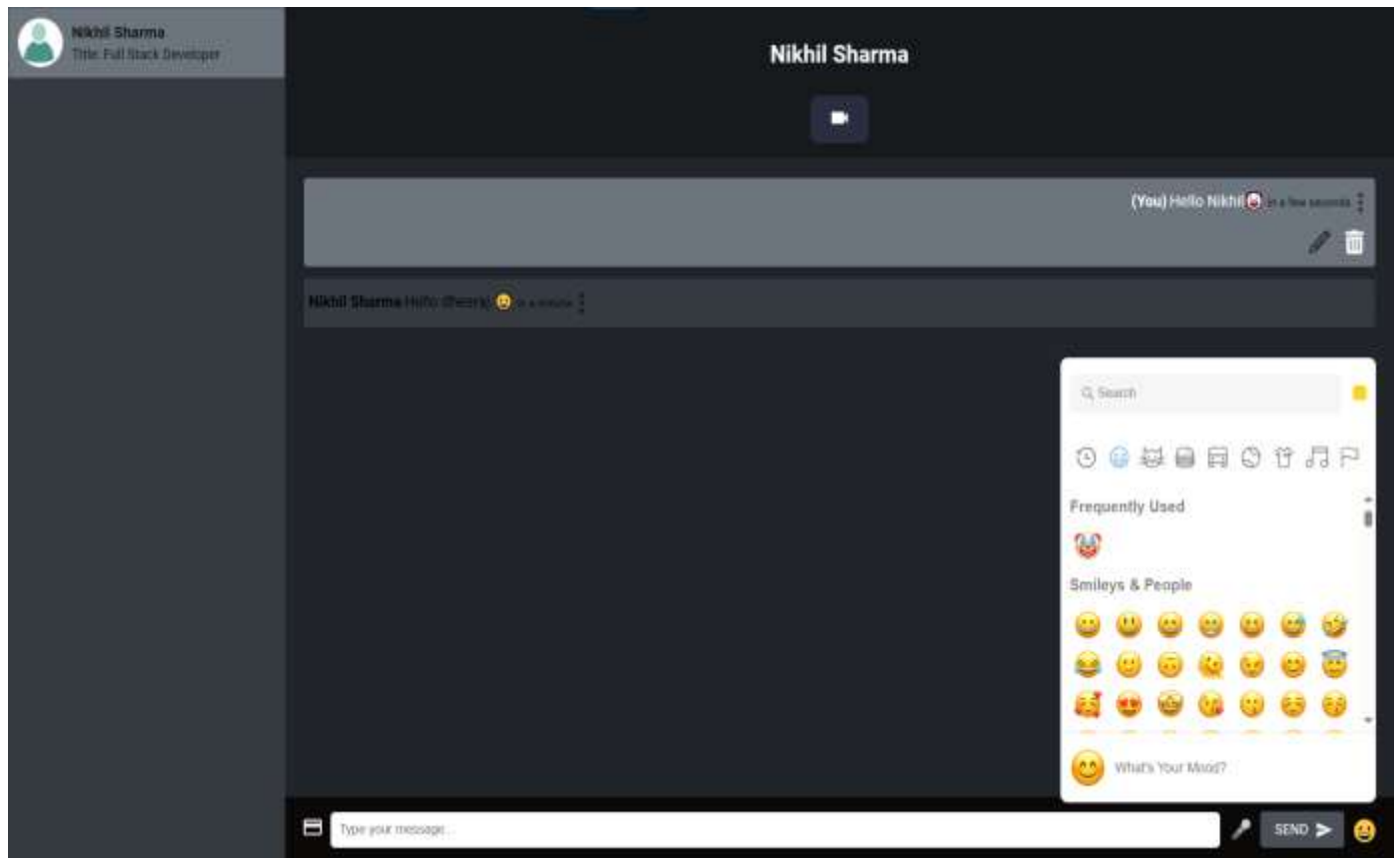


Figure 6 Chatting Page Sender

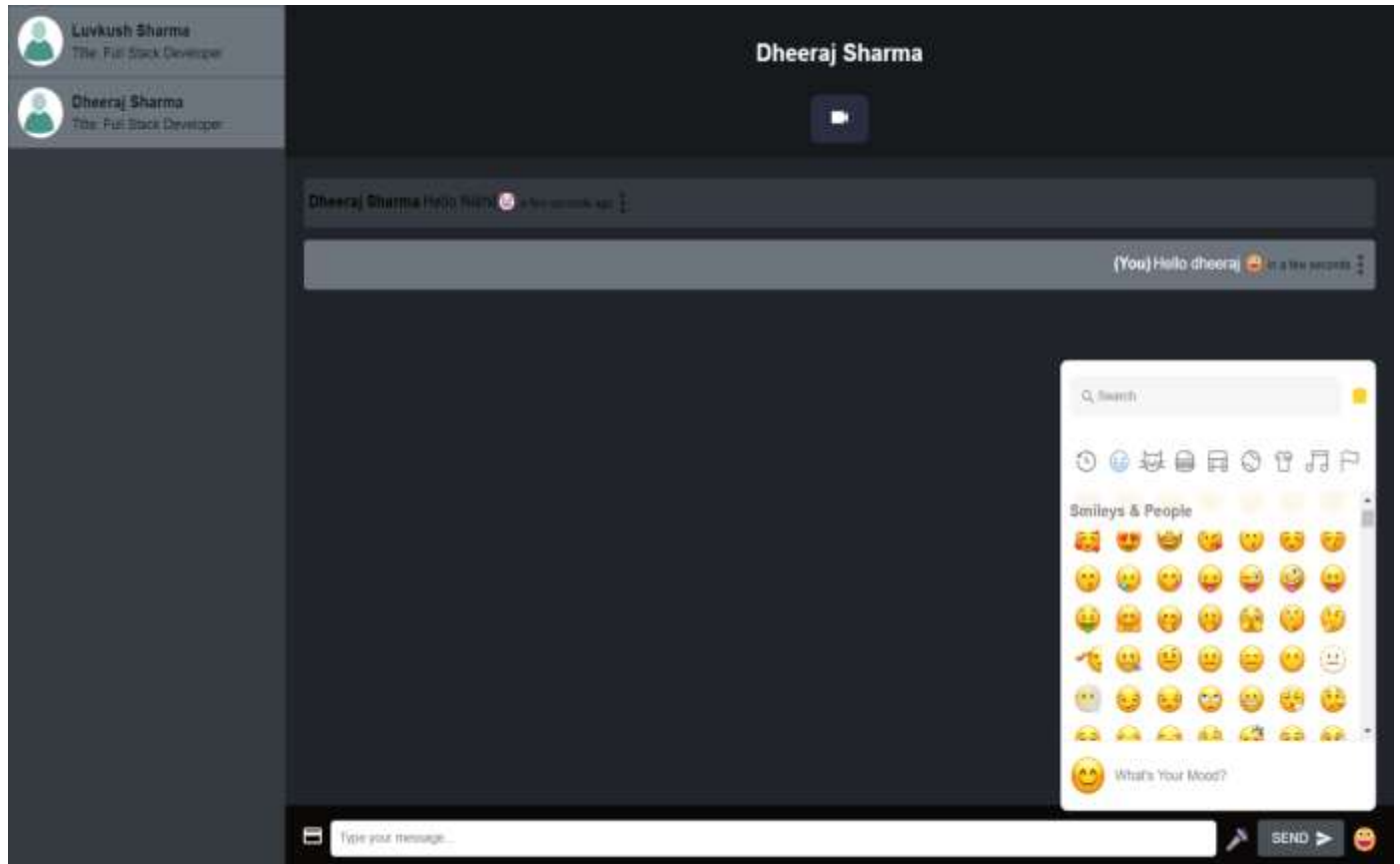


Figure 7 Chatting Page Receiver

CHAPTER 4 :- Software Testing

Software testing is a crucial process in the software development lifecycle aimed at identifying defects, errors, or bugs in the software to ensure its quality, reliability, and performance. It involves executing the software system or components under controlled conditions and evaluating the results to determine whether they meet specified requirements.

Here are some key aspects of software testing:

- i. Functional Testing :-** Functional testing focuses on verifying that each feature and functionality of the DOST App performs as intended according to the specified requirements. This includes testing individual components, user interactions, and system behaviours to ensure that they meet user expectations and business objectives.
- ii. Usability Testing :-** Usability testing evaluates the ease of use, intuitiveness, and overall user experience of the DOST App. This involves gathering feedback from real users through user interviews, surveys, and usability tests to identify any usability issues or pain points and make iterative improvements to the application's design and functionality.
- iii. Performance Testing :-** Performance testing assesses the responsiveness, scalability, and stability of the DOST App under various load conditions. This includes stress testing, load testing, and endurance testing to determine how the application performs under normal usage, peak usage, and prolonged usage scenarios, ensuring optimal performance and reliability.
- iv. Capability Testing :-** Compatibility testing ensures that the DOST App functions correctly across different devices, browsers, and operating systems. This involves testing the application on various platforms and configurations to identify any compatibility issues or inconsistencies and optimize the application for a seamless user experience across all environments.

Through a comprehensive testing approach encompassing functional testing, usability testing, performance testing, compatibility testing, and security testing, we aim to ensure that the DOST App meets the highest standards of quality, reliability, and security. By identifying and addressing any issues or deficiencies early in the development process, we strive to deliver an application that exceeds user expectations and delivers exceptional value.

CHAPTER 5 :- Conclusion

The DOST application epitomizes the essence of modern connectivity, offering a platform where individuals can transcend physical boundaries and forge meaningful connections with like-minded individuals. Through its intuitive interface and advanced features, DOST empowers users to navigate the complexities of social interaction in an increasingly digital landscape.

At its core, DOST is a testament to the human need for belonging and companionship. In a world where loneliness and isolation are prevalent, DOST serves as a beacon of hope, providing a safe and welcoming space where users can find solace, support, and camaraderie.

One of DOST's greatest strengths lies in its technical prowess. Leveraging the robustness of the MERN stack and Socket.IO, DOST ensures seamless performance, scalability, and security. By prioritizing user experience and reliability, we aim to cultivate a platform that users can trust and rely on for their social networking needs.

But DOST is more than just a software application—it's a community. Through shared experiences, interests, and conversations, DOST fosters bonds that transcend the digital realm. Whether it's connecting with a fellow enthusiast, finding a study buddy, or simply sharing a laugh, DOST facilitates moments of genuine connection that enrich lives.

As we embark on this journey, our commitment to inclusivity and diversity remains unwavering. DOST is a space where everyone is welcome, regardless of age, gender, race, or background. By embracing the richness of human diversity, we aim to create a community that celebrates individuality and fosters mutual respect and understanding.

Looking ahead, the future of DOST is bright and promising. With each iteration and update, we strive to enhance the user experience, introduce new features, and adapt to the evolving needs of our community. Our dedication to continuous improvement ensures that DOST remains at the forefront of social networking innovation.

But perhaps the most remarkable aspect of DOST is its ability to make a difference in people's lives. From forging lifelong friendships to providing a source of comfort during challenging times, DOST has the power to create meaningful connections that leave a lasting impact.

In the end, DOST is more than just an application—it's a movement. It's about breaking down barriers, bridging divides, and fostering a sense of belonging in an increasingly fragmented world. Together, let's embrace the power of connection and make the world a friendlier and more compassionate place, one connection at a time.

Join us on this journey. Together, we can build a community where everyone feels seen, heard, and valued. With DOST, the possibilities for connection are limitless.

CHAPTER 6 :- Summary

The DOST application is a revolutionary platform designed to facilitate meaningful connections and foster friendships in today's digital age. With its user-friendly interface, advanced matching algorithm, and real-time communication features, DOST empowers users to discover like-minded individuals, engage in enriching conversations, and build lasting relationships.

At its core, DOST aims to address the pervasive issue of social isolation by providing a safe and inclusive space for individuals to connect and interact. Through shared experiences, interests, and conversations, DOST cultivates a sense of belonging and camaraderie among its users, bridging geographical boundaries and fostering a global community of friends.

DOST boasts an intuitive and easy-to-navigate interface, ensuring a seamless user experience for individuals of all ages and backgrounds. With built-in chat and messaging features, DOST enables users to engage in real-time conversations with their matches, fostering deeper connections and interactions. DOST prioritizes user privacy and security, implementing robust measures to safeguard personal information and ensure a safe and secure environment for interaction.

It's about breaking down barriers, bridging divides, and fostering a sense of belonging in an increasingly fragmented world. Together, let's embrace the power of connection and make the world a friendlier and more compassionate place, one connection at a time.

In summary, the DOST application represents a beacon of hope for individuals seeking genuine connections in an increasingly fragmented world. By embracing the power of technology to bring people together, DOST aims to redefine the way we connect and build friendships, making the world a friendlier and more compassionate place, one connection at a time.

References

- <https://expressjs.com/>
- <https://nodejs.org/docs/latest/api/>
- <https://mongoosejs.com/docs/documents.html>
- <https://react.dev/learn>
- <https://socket.io/docs/v4/>
- <https://jwt.io/introduction>