

CONVO – THE MESSAGING APP

INTERDISCIPLINARY PROJECT

Submitted in partial fulfilment of the
requirements for the award of
Bachelor of Technology degree in Information Technology
by

ANGELIN E TANYA (41120020)



**DEPARTMENT OF INFORMATION TECHNOLOGY
SCHOOL OF COMPUTING**

SATHYABAMA

**INSTITUTE OF SCIENCE AND TECHNOLOGY
(DEEMED TO BE UNIVERSITY)**

**Category – 1 University by UGC
Accredited with Grade “A++” by NAAC |
12B Status by UGC | Approved by AICTE
JEPPIAAR NAGAR, RAJIV GANDHI SALAI,
CHENNAI - 600 119.**

MAY 2024



SATHYABAMA

INSTITUTE OF SCIENCE AND TECHNOLOGY

(DEEMED TO BE UNIVERSITY)

Accredited "A++" Grade by NAAC | 12B Status by UGC | Approved by AICTE

www.sathyabama.ac.in

DEPARTMENT OF INFORMATION TECHNOLOGY

BONAFIDE CERTIFICATE

This is to certify that this Project Report is the bonafide work of **ANGELIN E TANYA (41120020)** who carried out the project entitled **CONVO – THE MESSAGING APP** under my supervision from January 2024 to May 2024.

Internal Guide

Dr. KAMATCHI K S, M.E., Ph.D.,

Head of the Department

Dr. R. SUBHASHINI, M.E., Ph. D.,

Submitted for Viva voce Examination held on 08/05/24

Internal Examiner

External Examiner

DECLARATION

I **ANGELIN E TANYA (41120020)** hereby declare that the project report entitled **CONVO – THE MESSAGING APP** done by me under the guidance **Dr. KAMATCHI K S, M.E., Ph.D., (Internal)** is submitted in partial fulfilment of the requirements for the award of Bachelor of Technology degree in Information Technology.

DATE: 08 - 05 - 24

PLACE: **CHENNAI**

Angelin Tanya

SIGNATURE OF THE CANDIDATE

ACKNOWLEDGEMENT

I am pleased to acknowledge my sincere thanks to Board of Management of **SATHYABAMA INSTITUTE OF SCIENCE AND TECHNOLOGY** for their kind encouragement in doing this project and for completing it successfully. I am grateful to them.

I convey my thanks to **Dr. T. Sasikala M.E., Ph.D.**, Dean, School of Computing **Dr. R. Subhashini M.E., Ph.D.**, Head of the Department of Information Technology for providing me necessary support and details at the right time during the progressive reviews.

I would like to express my sincere and deep sense of gratitude to my Project Guide **Dr. Kamatchi K S, M.E., Ph.D.**, for her valuable guidance, suggestions and constant encouragement paved way for the successful completion of my project work.

I wish to express my thanks to all Teaching and Non-teaching staff members of the **Department of Information Technology** who were helpful in many ways for the completion of the project.

CERTIFICATE OF COMPLETION



149, 1C, 2nd Floor, Mount Poonamallee Rd,
Ramapuram, Chennai - 600089.
admin@hackwittechnologies.com
www.hackwittechnologies.com

10.02.2024

CERTIFICATE FOR INTERNSHIP

TO WHOMSOEVER IT MAY CONCERN

This is to Certify that **ANGELIN E TANYA (Reg No: 41120020)** student of Sathyabama Institute of Science and Technology, B. Tech Information Technology, has successfully completed the Internship at Hackwit Technologies from 22nd January to 30th January 2024.

During the internship, **ANGELIN E TANYA** has closely worked as a part of the **React Full stack** by carrying out the project entitled "**CONVO**".

She demonstrated good developing and learning skill with self-motivated attitude to learn new things. Her contribution to implement the React Full stack was good. We wish her all the best for her future endeavors.

Best Regards,

Ravikumar S
CEO & Managing Director
Hackwit Technologies Pvt Ltd

CERTIFICATE OF COMPLETION



THIS CERTIFICATE IS PRESENTED TO

ANGELIN E TANYA

has successfully completed the 40 Hrs hands-on certification
courses on MERN STACK Development at
Sathyabama Institute of Science and Technology
from 22.01.2024 to 30.01.2024.

A handwritten signature in black ink, appearing to be 'S. Sathya', written over a horizontal line.

DIRECTOR

A handwritten signature in black ink, appearing to be 'H. Gub', written over a horizontal line.

INSTRUCTOR

ABSTRACT

CONVO represents a paradigm shift in online communication, leveraging the power of modern technology to provide users with a seamless and dynamic chatting experience. Built on the MERN (MongoDB, Express.js, React.js, Node.js) stack, CONVO harnesses the latest advancements in web development to deliver a robust and feature-rich platform. At its core, CONVO utilizes MongoDB, a flexible and scalable NoSQL database, to efficiently store and manage user data in JSON format, ensuring optimal performance and reliability.

The project encompasses the development of both frontend and backend components, ensuring a seamless user experience while incorporating robust functionality and security measures.

The frontend of CONVO is developed using React.js, a popular JavaScript library known for its component-based architecture and declarative syntax. By incorporating React Vite, a next-generation build tool, CONVO achieves lightning-fast load times and an ultra-responsive user interface. This combination of technologies enables CONVO to deliver a smooth and immersive chatting experience, empowering users to communicate effortlessly across various devices and screen sizes.

MongoDB serves as the database to store user information, menu items, orders, and transaction details, facilitating efficient data management and retrieval.

TABLE OF CONTENTS

CHAPTER NO	TITLE	PAGE NO
	ABSTRACT	vii
	LIST OF FIGURES	x
1	INTRODUCTION	1
	1.1 Problem Statement	4
2	METHODOLOGY	6
	2.1 Project Objectives	6
	2.2 Requirement Analysis	7
	2.2.1 Hardware Requirements	7
	2.2.2 Software Requirements	7
	2.3 Proposed Model	8
	2.3.1 Overview	8
	2.3.2 Key Features	8
	2.4 System Architecture	9
	2.4.1 Frontend	9
	2.4.2 Backend	9
	2.4.3 Database	10
	2.4.4 Server	10

2.5	Modules	11
2.5.1	User Registration Module	11
2.5.2	User Registration Module Algorithm	11
2.5.3	User Login Module	12
2.5.4	User Login Module Algorithm	13
2.5.5	Message Module	14
2.5.6	Message Module Algorithm	15
3	RESULTS AND DISCUSSION	17
3.1	Frontend Results	17
3.2	Backend Results	19
4	CONCLUSION AND FUTURE WORKS	22
4.1	Conclusion	22
4.2	Future Works	23
	REFERENCES	26
	APPENDICES	27
A	Sample code	27
B	Screenshots	34

LIST OF FIGURES

FIGURE NO.	FIGURE NAME	PAGE NO.
1.1	Web Chat Application	5
2.1	System Architecture	10
3.1	User Registration Page	17
3.2	User Login Page	18
3.3	Chat History Page	18
3.4	Chat Page	19
3.5	Text message in database	20
3.6	Shared image in Database	20
3.7	Shared pdf file in Database	21
3.8	User information in Database	21
3.9	Shared pdf file	34
3.10	Shared png file	35
3.11	Database connection	35
3.12	Database collection	36

CHAPTER 1

INTRODUCTION

In today's digital age, where information flows at breakneck speed and staying connected is crucial, effective communication is no longer a luxury, it's a necessity. However, traditional methods often fall short, leaving us feeling frustrated and isolated. We juggle through endless text message threads, struggling to convey complex ideas or share important documents. Sometimes, words simply aren't enough. We yearn for a way to express ourselves with more nuance, to capture the full spectrum of human emotion and experience in our interactions. This is where CONVO steps in, a revolutionary online chat web application designed to bridge the gap and redefine how we connect and collaborate. CONVO goes beyond the limitations of text messaging, offering a comprehensive platform that empowers richer, more meaningful communication. Imagine a world where sharing a critical work document is as easy as sending a message, where cherished memories can be captured and shared with loved ones in the very moments they happen, and where complex ideas or emotions can be conveyed through the power of video. CONVO unlocks this potential, fostering a new era of connection and collaboration that reflects the richness and complexity of human interaction.

CONVO shatters the confines of text-based messaging, offering a robust platform that empowers communication that's as rich and multifaceted as human experience itself. Forget juggling endless text messages or struggling to explain complex ideas with mere words. With CONVO, sharing critical documents becomes effortless. Imagine instantly sending PDFs, presentations, or even spreadsheets directly within a chat, ensuring everyone on your team has the information they need at their fingertips. Capture life's precious moments as they unfold – share photos with loved ones the very second the memory is made.

Need to explain a technical process with more clarity? Upload a quick screen recording to visually demonstrate each step. Want to add a dash of humor to a birthday greeting? Send a funny video message that will leave them laughing. CONVO transcends the limitations of text, allowing you to leverage the power of various media formats – photos, videos, and screen recordings – to express yourself with unparalleled depth and nuance. It's communication that reflects the full spectrum of human emotion and experience, fostering genuine connections that go far beyond the boundaries of words.

CONVO's power extends far beyond its ability to share various file formats. We understand that fostering genuine connection hinges on an interface that's as user-friendly as it is powerful. That's why CONVO boasts effortless registration – no jumping through hoops or navigating complex sign-up processes. Instead, you'll be up and running in a matter of minutes. But user-friendliness doesn't stop there. The design prioritizes intuitive navigation, with features and functionalities that are clear and easy to understand. No more deciphering cryptic menus or wasting precious time trying to figure out how things work. CONVO feels natural and familiar, like an extension of your everyday communication. This focus on user experience ensures that everyone, regardless of technical background, can leverage the full potential of the platform. With CONVO, the technical barriers melt away, allowing you to focus on what truly matters – building meaningful connections with the people who matter most in your life. The result? Communication that's smooth, efficient, and fosters a sense of genuine connection that transcends the limitations of traditional chat applications.

CONVO goes beyond the realm of simply sending messages. It aspires to be the cornerstone for fostering genuine connection. At the heart of this philosophy lies our robust user management system. This powerful feature allows for multiple user logins, ensuring you can stay connected with everyone in your life, from close friends and family to colleagues and fellow community members. No more wondering if someone will see

your message – real-time updates on users' online/offline status keep you informed and in the loop. Never miss a heartfelt message from a loved one again, or a crucial update from a teammate.

This functionality extends beyond simple one-on-one communication. CONVO empowers seamless collaboration through the creation of dedicated chat groups. Imagine a central hub for your work team, where brainstorming ideas, sharing resources, and staying connected in real-time becomes effortless. Family gatherings can take on a whole new dimension with dedicated chat groups, fostering a sense of togetherness that transcends physical distance. Whether you're catching up with relatives across the globe or planning the next family reunion, CONVO keeps everyone on the same page, ensuring a sense of closeness, even when miles apart. CONVO is a platform that fosters community.

Project CONVO collapses physical distance and fosters real-time collaboration, creating a seamless flow of ideas and information. CONVO eliminates the frustration of sharing files that often plagues traditional communication methods. Say goodbye to the days of wrestling with email attachment size limits or clunky file transfer services that require separate logins and uploads. With CONVO, sharing files becomes a seamless extension of the conversation itself. Imagine collaborating on a project and needing to quickly share a document with your team. No need to switch applications or navigate a separate platform. CONVO integrates seamlessly with popular cloud storage solutions like Dropbox and Google Drive. This allows you to directly access and share files stored in your cloud storage without ever leaving the chat window. Need to share a specific spreadsheet or presentation from your Dropbox? Simply navigate to your Dropbox within CONVO, select the file, and it's instantly shared with your team members. This effortless integration ensures information flows freely and efficiently, keeping everyone on the same page and eliminating the time wasted wrestling with outdated file sharing methods. With CONVO, sharing

becomes as natural and intuitive as the conversation itself, fostering a collaborative environment where ideas and information can be exchanged quickly and effortlessly.

1.1 PROBLEM STATEMENT

Problem Statement: Streamlining Communication in an Era of Overloaded Chat Applications.

In today's digital landscape, the proliferation of chat applications has led to an inundation of unnecessary features, updates, and distractions, diluting the primary focus of these platforms: communication. Existing chat applications are burdened with an array of superfluous functionalities, ranging from social media integrations to complex multimedia capabilities, ultimately detracting from the essence of seamless interaction between users. Navigating through a maze of extraneous user interfaces with menus, notifications, and prompts, users often struggle to locate basic communication tools amidst the noise of peripheral features.

Consequently, the fundamental purpose of chat applications - facilitating meaningful and efficient communication - becomes obscured, diminishing the user experience and hindering productivity. There is a pressing need for a streamlined and focused chat platform that prioritizes simplicity, clarity, and functionality, enabling users to connect and converse with ease, without the distraction of extraneous bells and whistles.

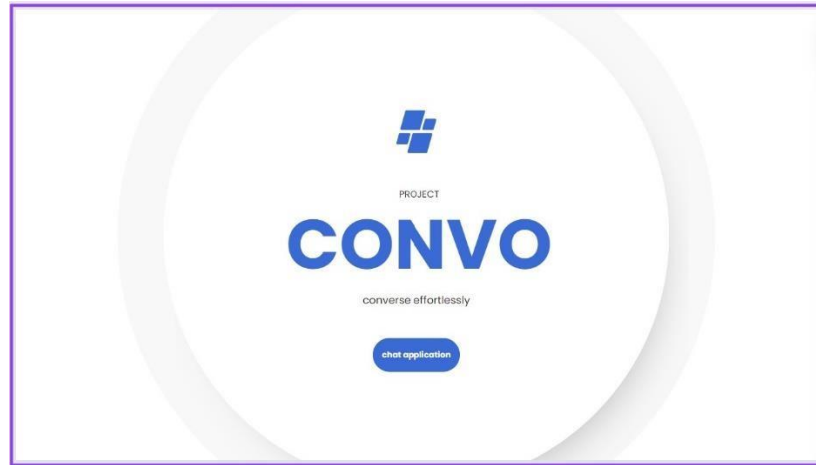


Fig:1.1: Web Chat Application

CHAPTER 2

METHODOLOGY

2.1 PROJECT OBJECTIVES

Overall Objective: To develop and launch CONVO, a revolutionary online chat web application that redefines how users connect and collaborate.

Specific Objectives:

1. Communication Richness:

- a. Enable users to send and receive a variety of media formats beyond text messages, including PDFs, photos, and videos.
- b. Facilitate the sharing of critical documents, cherished memories, and complex ideas through multimedia.

2. User Experience:

- a. Design a user-friendly interface with effortless registration and intuitive navigation.
- b. Ensure the platform is accessible to users of all technicalities.

3. Real-Time Connection:

- a. Implement a robust user management system with real-time online/offline status updates.
- b. Enable seamless creation of dedicated chat groups for work teams, families, and online communities.
- c. Increase user base through effective marketing and outreach strategies.
- d. Facilitate the sharing of critical documents.

4. Collaboration:

- a. Foster real-time collaboration through effortless information sharing and group communication.
- b. Integrate with popular cloud storage platforms (e.g., Dropbox, Google Drive) for easy access to stored files.

5. Security & Privacy:

- a. Prioritize user privacy and data security with encryption protocols.
- b. Provide users with control over data sharing within the platform.

6. Growth & Development:

- a. Continuously develop new features based on user feedback and market trends (e.g., video conferencing, voice chat).
- b. Increase user base through effective marketing and outreach strategies.

2.2 REQUIREMENT ANALYSIS

2.2.1 HARDWARE REQUIREMENTS

Processor	:	Processor Intel CORE i3 and above
RAM	:	16 GB
Mouse	:	Digital
Keyboard	:	105 Digital Keys

2.2.2 SOFTWARE REQUIREMENTS

Operating System	:	Windows
Framework	:	REACT, REACT VITE, JAVASCRIPT
Development GUI	:	Visual Studio (VS CODE).
Database	:	MongoDB(MongoDB Compass).

2.3 PROPOSED MODEL

2.3.1. Overview:

In today's digital world, CONVO bridges the gap for everyone. This user-friendly chat platform empowers rich communication with photos, videos, and documents, making it perfect for families to share memories and stay connected. With a focus on security and real-time updates, CONVO ensures a safe space for kids to connect with loved ones. For senior citizens, the intuitive interface and effortless file sharing make it easy to stay in touch with family and friends, fostering a sense of closeness that transcends distance.

2.3.2. Key Features:

1. Rich Communication: Send and receive a variety of media formats beyond text messages:

Photos: Share cherished memories and life moments instantly.

Videos: Express complex ideas, emotions, or celebrate occasions with video messages.

Documents (PDFs, presentations, etc.): Effortlessly share critical work documents or important files.

Screen recordings: Explain technical processes or workflows visually.

2. User-Friendly Design:

Effortless registration and intuitive navigation for all technical background. Also, the clean interface prioritizes ease of use.

CONVO's interface avoids clutter, with clear navigation elements and well-defined sections for chats, messages, and user information.

3. Real-Time Connection:

Robust user management system with real-time online/offline status updates. See who's available for a chat instantly, never miss a message again.

4. Future-Proof Design:

Commitment to continuous development based on user feedback and market trends. Video conferencing for face-to-face interactions. Voice chat for hands-free communication. New features to enhance the user experience.

2.4 SYSTEM ARCHITECTURE

The system architecture of the proposed chat application website encompasses various components working together to deliver the desired functionalities. Here's an explanation of each component:

2.4.1. Frontend:

- 1 The frontend likely utilizes React.js framework to deliver an interactive user interface.
- 2 Users can leverage React Vite for potentially faster development workflows.

2.4.2. Backend:

The backend likely relies on Node.js runtime environment.

Express.js framework is likely used to construct the RESTful API endpoints that facilitate communication between the frontend and backend. CONVO's well-architected backend lays the foundation for a reliable, secure, and scalable communication platform. By leveraging the power of Node.js, Express.js, and a robust database solution, CONVO empowers users to connect and collaborate seamlessly.

2.4.3. Database:

A NoSQL database is used on the backend, potentially to store user data and message history due to its scalability for large amounts of data.

2.4.4. Server:

The central hub of the system, responsible for receiving requests from the user interface (client application) and providing responses, can be implemented using high-performance web servers such as Nginx or Apache. Alternatively, Node.js can be employed for this purpose. This component handles user authentication and authorization, manages sessions, serves static content like images and JavaScript code, and routes user requests to appropriate backend services. While the provided system architecture offers a general overview, the specific technologies and configurations utilized may vary depending on the actual implementation. Notably, the image does not depict components for functionalities like video chat or voice chat, which might be incorporated in future iterations of CONVO.

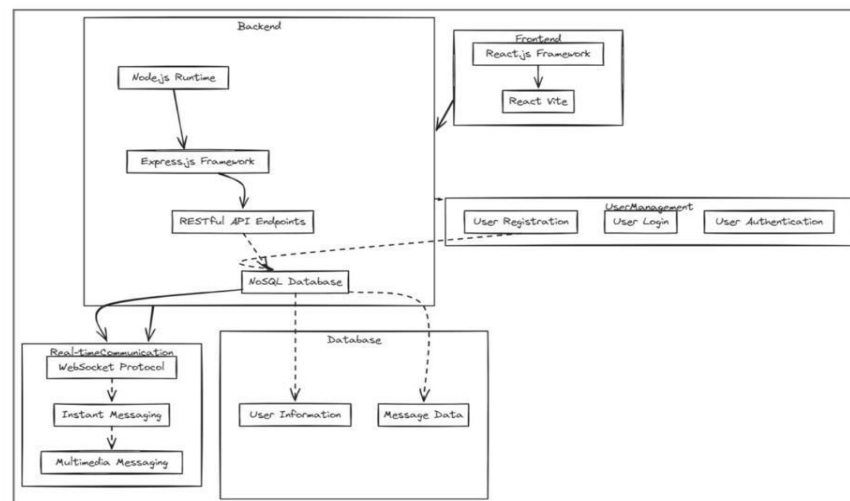


Fig 2.1 System Architecture

2.5 MODULES

In this project there are three modules. They are:

1. User Registration Module
2. User Login Module
3. Message Module

2.5.1. User Registration Module:

The user registration module is the gateway for users to join the CONVO community. It should be a streamlined and user-friendly process that gathers essential information while ensuring data security. Here's a breakdown of the functionalities:

1. Registration Form:

Username: Users choose a unique username for identification within CONVO.

Email Address: A valid email address is required for account creation and password recovery.

Password: Users create a secure password following best practices (e.g., minimum length, combination of characters). Password confirmation ensures users enter their password correctly.

Data Security: All data transmitted during registration should be encrypted using HTTPS to protect user privacy. Strong password hashing algorithms should be used to store passwords securely.

2.5.2 User Registration Module Algorithm:

Input: Admin username and password.

Process: User fills out registration form.

The form captures user information such as username, email address, password, which will be utilised for logging in.

Form validation: The system validates the user input.

Email address: Verify valid email format.

Password: Ensure password meets minimum length and complexity

Account creation: Upon successful validation and verification (if applicable), the system creates a new user account in the database.

User data is securely stored using hashing algorithms for passwords.

Output:

Login redirection: The system redirects the user to the login page where they can enter their credentials and access the CONVO platform.

2.5.3. User Login Module:

The login module is the entry point for existing CONVO users. It should be secure, efficient, and offer a seamless transition into the platform. Here's a breakdown of the functionalities:

Login Form: Username or Email Address: Users can log in using either their username or registered email address.

Password: The password field allows users to enter their login credentials.

"Remember Me" Checkbox (): This option allows users to keep themselves logged in for a predetermined period (on the same device) for faster access in the future.

Login Button: Clicking the button initiates the login process.

Login Validation: The system validates the user's credentials against the stored data in the database:

Username/Email: Verify the entered username or email address exists in the system.

Password: Check if the entered password matches the hashed password stored for that user account.

Login Success: Upon successful login, the system: Grants the user access to the CONVO platform.

Redirects them to their personalized CONVO homepage or dashboard.

Login Failure: In case of incorrect login credentials, the system provides clear and concise error messages.

"Invalid Username or Email Address"

"Incorrect Password"

The system should avoid displaying specific details about which credential is incorrect for security reasons (e.g., "Username or password incorrect").

2.5.4. User Login Module Algorithm

1. User initiates login:

User accesses the CONVO login page or form.

2. User enters credentials:

User enters their username or email address and password in the designated fields.

3. Form validation:

The system performs basic validation on the user input:

Username/Email: Ensure a value is entered.

Password: Ensure a value is entered.

4. Credential submission:

User clicks the "Login" button to submit their credentials.

5. Server-side validation and processing:

The system retrieves the entered username/email address and password. The username/email address is checked against the database to verify if it exists. If the username/email address is not found,

the system returns a "Username/Email Not Found" error message and redirects the user back to the login page.

6. Password verification:

If the username/email address is found, the entered password is compared against the hashed password stored for that user account in the database.

7. Login success:

If the password matches:

The system creates a login session for the user (depending on implementation). The user is granted access to the CONVO platform. The system redirects the user to their personalized CONVO homepage or dashboard.

8. Login failure:

If the password does not match:

The system increments a login attempt counter (, for security measures). The system displays a clear error message:

"Incorrect Password". The system redirects the user back to the login page.

ERROR HANDLING:

This algorithm provides a basic framework for user registration and user login in CONVO. Additional functionalities can be integrated based on specific requirements, such as social login options or more complex user data fields.

2.5.5. Message Module

The message module forms the heart of CONVO, enabling real-time communication and seamless exchange of information. Here's a breakdown of its potential functionalities:

1. **Real-Time Communication:** Type and send text messages, and they appear instantly for the recipient(s), fostering a sense of live interaction. No more waiting for emails or delayed replies.
2. **Rich Media Sharing:** Go beyond text! Users can share photos, videos, and documents directly within chats. File size limitations and supported file types can be defined to ensure smooth message delivery and server storage management.
3. **Chat History:** Conversations are automatically saved, allowing users to access past messages and search for specific information within chat history. Message timestamps and indicators for read/unread messages can enhance conversation flow.

2.5.6. Message module Algorithm

This algorithm outlines the core functionalities of the message module in CONVO:

1. **Sending a Message:** User initiates a chat with a contact (individual) or opens an existing chat conversation. User composes their message in the text box. User clicks the "Send" button.
2. **Message Processing:** The system validates the message content: Check for empty messages (prevent sending blank messages). Enforce character limits.
3. **Media Handling:** If the message includes media (photos, videos, documents), the system performs the following: Upload the media files to the CONVO server or a designated storage service (e.g., cloud storage integration). Resize or compress large media files to optimize transmission and storage.
4. **Message Delivery:** The system identifies the recipient(s) of the message (individual). The message content (text, formatted text, media links) is packaged for transmission.

5. Real-Time Delivery (if recipient(s) are online): If the recipient(s) are currently online, the system utilizes real-time communication protocols (e.g., Web Sockets) to deliver the message instantly.

6. Offline Delivery (if recipient(s) are offline): If the recipient(s) are offline, the system stores the message in a queue or database. Message delivery will be attempted upon the recipient(s) next login to CONVO.

7. Conversation History: All sent and received messages are stored in the conversation history. Messages are displayed chronologically, allowing users to revisit past conversations and search for specific information.

CHAPTER 3

RESULTS AND DISCUSSION

3.1 FRONTEND RESULTS

The frontend translates into a user interface that feels responsive and adapts to your actions. Whether it's navigating through menus, composing messages, or viewing media content, CONVO's frontend strives for a smooth and intuitive experience.

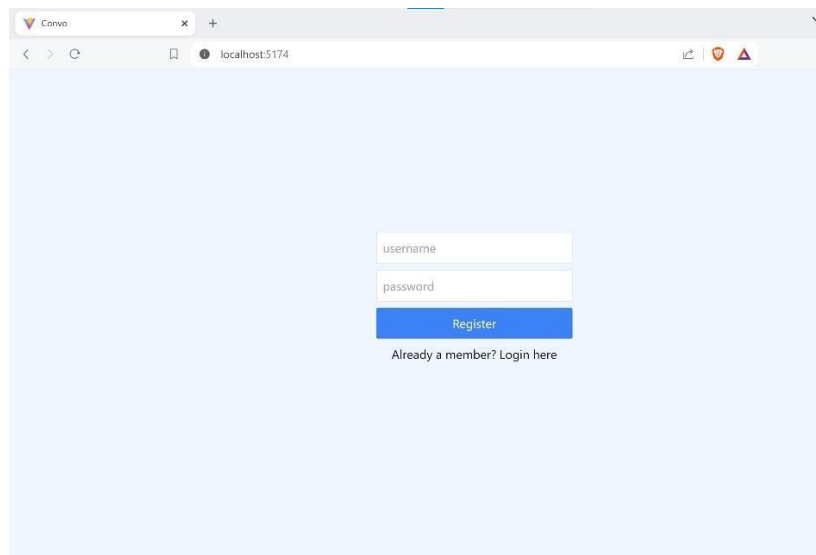


Fig 3.1 User Registration Page

The Registration Page welcomes you to create your account. This page likely involves fields for username selection (unique identifier within CONVO), email address (for verification), and password creation (adhering to security best practices). A clear and concise introduction to CONVO welcomes new users, highlighting the platform's core benefits. This could be a brief tagline or a short sentence explaining CONVO's purpose (e.g., "Connect seamlessly with friends, family, and colleagues").

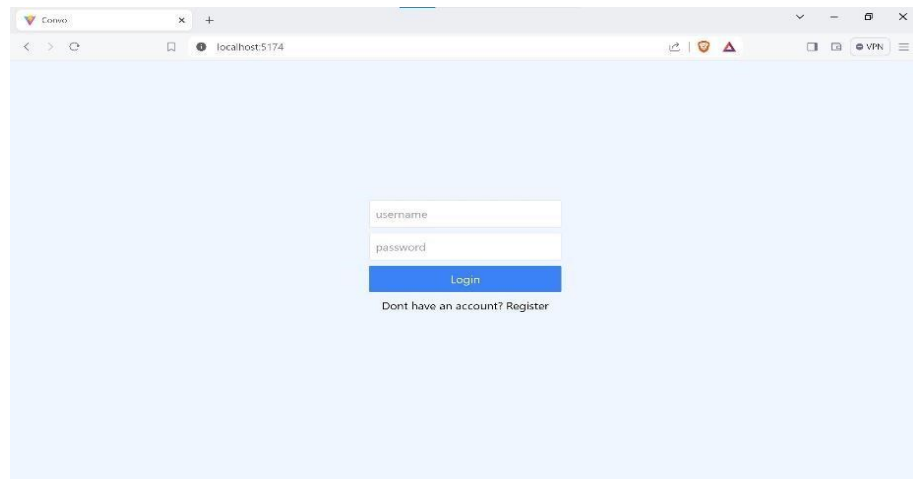


Fig 3.2 User Login Page

The Login Page

Your interaction with CONVO begins at the login page. This is where you use your registered credentials (username or email address and password) to gain access to the platform. The login page should be clear, concise, and secure. Imagine a clean and uncluttered interface with designated fields for username/email and password entry.

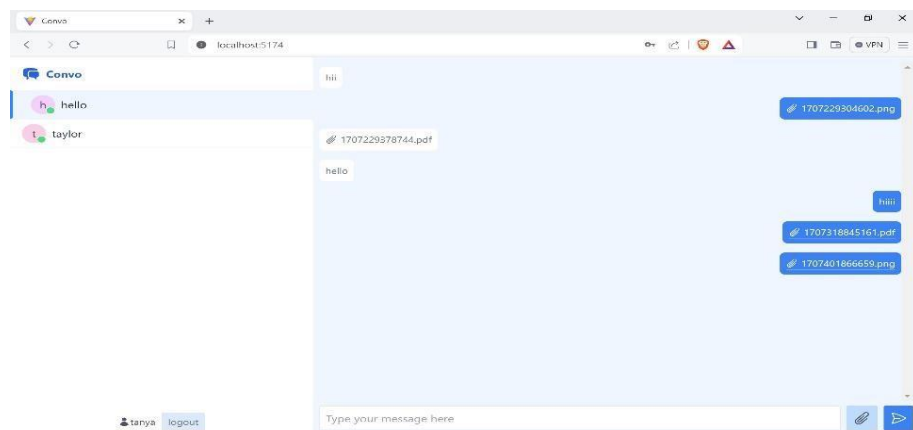


Fig 3.3 Chat history page

The Chat History Page

The chat history page serves as your central hub for accessing past conversations. Imagine a neatly organized list displaying your chats with individuals and groups. Each entry might showcase a snippet of the latest message, along with a timestamp and potentially an indicator for unread messages. This allows you to easily pick up where you left off in your conversations.

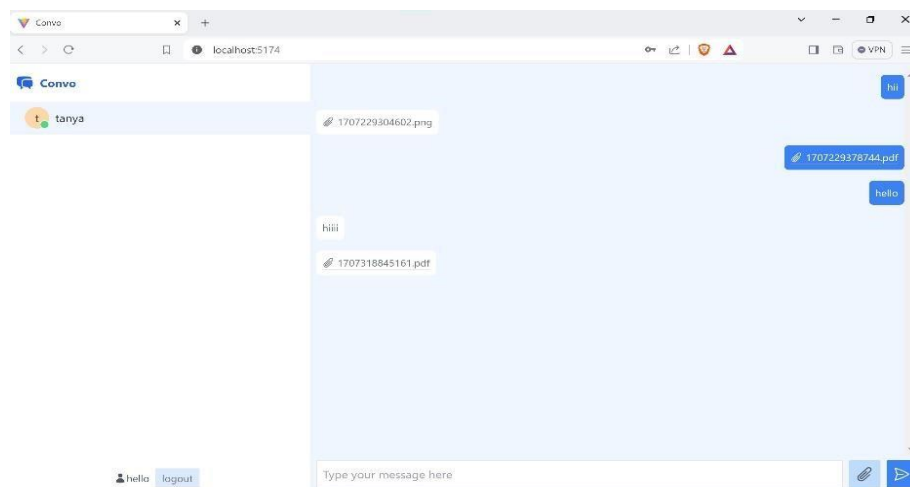


Fig 3.4 Chat page

The Chats/Messages Page

The chats/messages page is the platform where you actively engage in communication. This page likely displays the conversation thread with your chosen contact (individual). Text messages appear chronologically, potentially with visual cues for message formatting and media content (photos, videos, documents).

3.2 BACKEND RESULTS

CONVO emerges as a versatile and user-friendly platform designed to bridge communication gaps and foster connections. Its modular

architecture allows for a feature rich experience, catering to various communication needs.

From secure user management and real-time messaging to intuitive chat history and online/offline status indicators.

CONVO empowers users to stay connected and collaborate seamlessly.

The potential integration of functionalities like group chats, media sharing, and accessibility features further broadens its appeal, making it a valuable tool for both personal and professional interactions.

The backend, potentially built upon MongoDB, ensures efficient data storage and management. Security measures like password hashing and access controls safeguard user privacy. Additionally, MongoDB's scalability empowers CONVO to adapt and grow effortlessly as its user base expands.

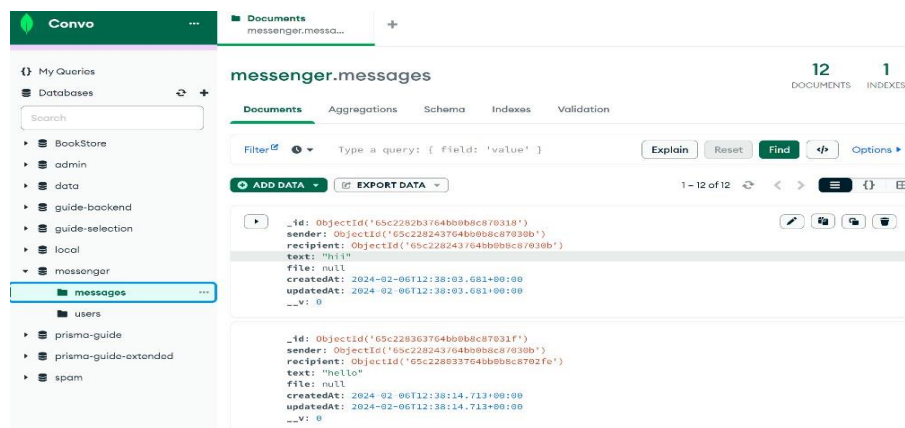


Fig 3.5 Text message in Database

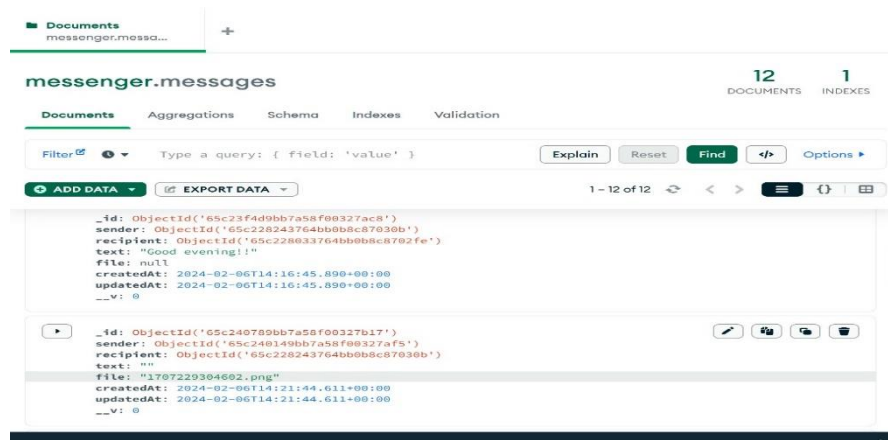


Fig 3.6 Shared image in Database

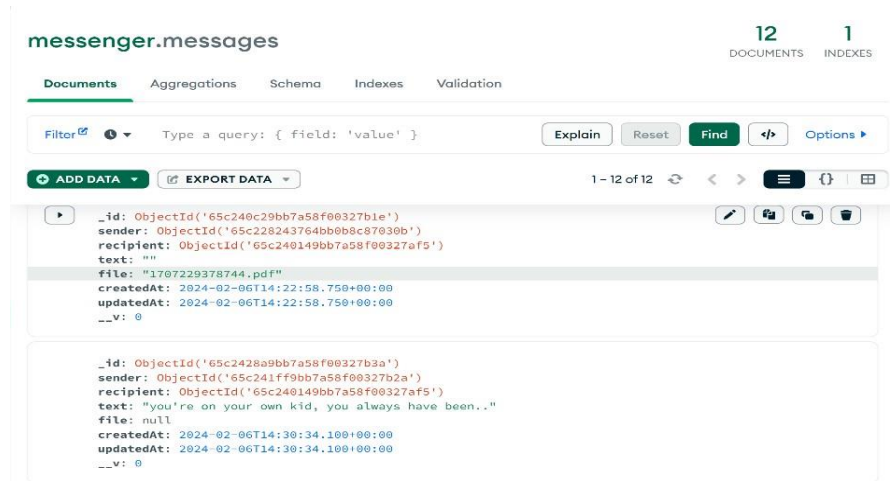


Fig 3.7 Shared pdf file in Database

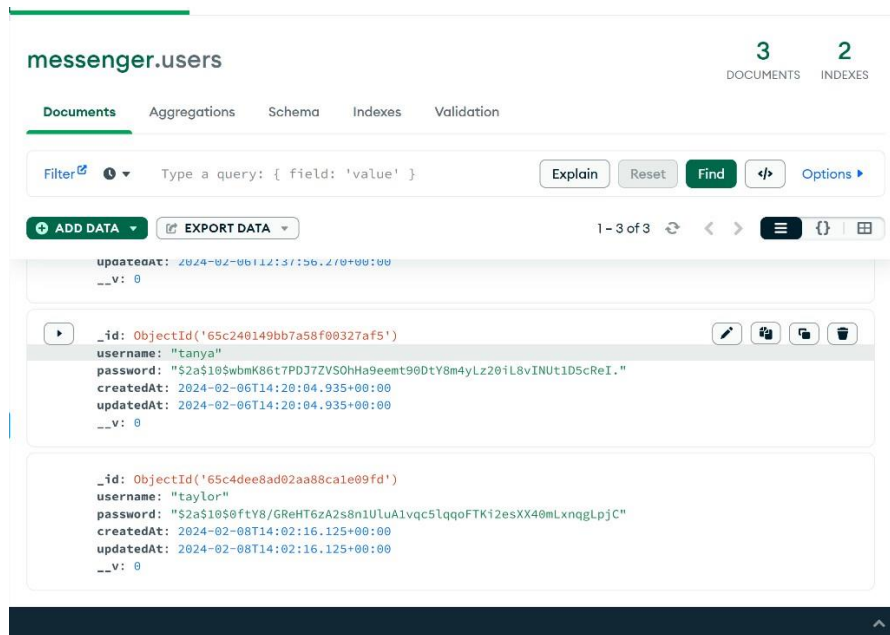


Fig 3.8 User information in Database

CHAPTER 4

CONCLUSION AND FUTURE WORKS

4.1 CONCLUSION

CONVO emerges as a versatile and user-friendly platform designed to bridge communication gaps and foster connections. Its modular architecture allows for a feature rich experience, catering to various communication needs.

From secure user management and real-time messaging to intuitive chat history and online/offline status indicators,

CONVO empowers users to stay connected and collaborate seamlessly. The potential integration of functionalities like group chats, media sharing, and accessibility features further broadens its appeal, making it a valuable tool for both personal and professional interactions.

The backend, potentially built upon MongoDB, ensures efficient data storage and management. Security measures like password hashing and access controls safeguard user privacy. Additionally, MongoDB's scalability empowers CONVO to adapt and grow effortlessly as its user base expands.

In conclusion, CONVO positions itself as a well-rounded communication platform with a commitment to user-friendliness, security, and scalability. Its potential to streamline communication across diverse scenarios, from casual interactions to professional collaborations, positions CONVO as a strong contender in the evolving realm of online interaction. As CONVO continues to evolve, incorporating new features and functionalities, it has the potential to become a preferred choice for users seeking a feature-rich and secure communication experience.

4.2 FUTURE WORKS

Here are points outlining potential future work directions for CONVO:

1. Advanced Communication Features:

Implement functionalities like group video calls and screen sharing to facilitate richer collaboration experiences.

Explore voice message recording and playback for a more natural communication flow.

2. Enhanced Security and Privacy:

Integrate end-to-end encryption for messages (optional) to provide an extra layer of security for highly sensitive communication.

Implement multi-factor authentication (MFA) to strengthen user account security.

Develop granular privacy controls allowing users to manage how their data is shared and used within CONVO.

3. Advanced Search and Content Management:

Integrate powerful search functionalities to enable users to easily locate specific messages, files, or contacts within their chat history.

Explore options for message tagging or categorization to further enhance message organization.

Consider offering message pinning or starring features for prioritizing important conversations.

4. Artificial Intelligence Integration (Optional):

Introduce chatbots with natural language processing capabilities to provide customer service support or answer frequently asked questions.

Explore sentiment analysis features to understand the overall tone of conversations, potentially aiding communication strategies.

Implement message translation features (optional) to break down language barriers and facilitate communication across borders.

5. Customization and Integrations:

Allow users to personalize the CONVO interface with custom themes, notification sounds, or chat wallpapers.

Explore integration possibilities with other productivity tools or project management platforms for a more streamlined workflow.

Consider API development to allow third-party applications to interact with CONVO's functionalities.

6. Mobile Application Development:

Develop a dedicated mobile application for CONVO, allowing users to stay connected and access their communication channels on the go.

Ensure seamless integration between the web and mobile platforms for a consistent user experience.

7. Analytics and Data Insights:

Implement analytics dashboards to provide users with insights into their communication patterns and preferences.

Offer features for monitoring team communication or gauging project collaboration effectiveness.

8. Content Moderation and Safety:

Implement robust content moderation tools and policies to ensure a safe and respectful communication environment for all users.

Develop mechanisms for users to report inappropriate content or flag suspicious activity, promoting a collaborative approach to maintaining a positive platform experience.

9. Ephemeral Messaging:

Explore the possibility of offering "disappearing messages" functionality, allowing users to send messages that automatically delete after a set time. This feature can cater to specific communication needs where temporary information sharing is desired.

10. Accessibility Enhancements:

Continuously evaluate and improve accessibility features to ensure CONVO is inclusive for users with disabilities. Explore advanced text-to-speech and voice control functionalities for a more hands-free experience.

REFERENCES

- A Comprehensive Study of MERN Stack - Architecture, Popularity and Future Scope (2021). International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), 7(6), 236-240.
- Applications using Node.js and MongoDB (2017). International Journal of Advanced Research in Computer and Communication Engineering, 6(2), 200-203.
- Express supervision system based on NodeJS and MongoDB (2018). 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO).
- Performance Optimization using MERN stack on Web Application (2021). International Journal of Engineering Research & Technology, 10(6), 102-105.
- Real-Time Data Visualization in Web Applications: A Case Study of the MERN Stack. Proceedings of the IEEE International Conference on Big Data (2017).
- Performance Evaluation of MERN Stack for Large-Scale Web Applications. Journal of Web Engineering (JWE)(2016).
- Secure Web Application Development Using the MERN Stack. ACM Transactions on Internet Technology (TOIT)(2018).

APPENDICES

A. Sample code:

Message.js

```
const mongoose = require('mongoose');
```

```
const MessageSchema = new mongoose.Schema({  sender: {type:
mongoose.Schema.Types.ObjectId, ref:
'User'},  recipient: {type: mongoose.Schema.Types.ObjectId, ref:
'User'},  text: String,  file: String,
}, {timestamps:true});
```

```
const      MessageModel      =      mongoose.model('Message',
MessageSchema);
```

```
module.exports = MessageModel;
```

```
User.js const mongoose = require('mongoose');
```

```
const MessageSchema = new mongoose.Schema({  sender: {type:
mongoose.Schema.Types.ObjectId, ref:
'User'},  recipient: {type: mongoose.Schema.Types.ObjectId, ref:
'User'},  text: String,  file: String,
}, {timestamps:true});
```

```
const MessageModel = mongoose.model('Message',
MessageSchema);
```

```
module.exports = MessageModel;
```

```
Index.js const express = require('express'); const mongoose =
require('mongoose'); const cookieParser = require('cookie-parser');
const dotenv = require('dotenv'); const jwt = require('jsonwebtoken');
const cors = require('cors'); const bcrypt = require('bcryptjs'); const
User = require('./models/User'); const Message =
require('./models/Message'); const ws = require('ws'); const fs =
require('fs'); const { log } = require('console'); const morgan =
require('morgan')
```

```
dotenv.config();
```

```
mongoose.connect(process.env.MONGO_URL, (err) => {
  if (err) throw err;
});
```

```
const jwtSecret = process.env.JWT_SECRET; const bcryptSalt =
bcrypt.genSaltSync(10);
```

```
const allowedOrigin = ['http://localhost:5173', 'http://localhost:5174',
'http://localhost:5175',
'http://localhost:5176', 'http://localhost:5177']
```

```
const app = express(); app.use('/uploads',
  express.static(__dirname +
    'uploads')); app.use(express.json()); app.use(cookieParser());
app.use(cors({ credentials: true, origin: allowedOrigin
}));
app.use(morgan('dev'))
```

```
console.log('listening');
```

```
async function getUserDataFromRequest(req) { return new
Promise((resolve, reject) => {    const token = req.cookies?.token;    if
(token) {
        jwt.verify(token, jwtSecret, {}, (err, userData) => {        if (err) throw
err;        resolve(userData);
    });
    } else {    reject('no token');
    }
});
}
```

```
app.get('/test', (req,res) => { res.json('test ok');
});
```

```
app.get('/messages/:userId', async (req,res) => {    const {userId} =
req.params;
    const userData = await getUserDataFromRequest(req); const
ourUserId = userData.userId; const messages = await Message.find({
sender:{$in:[userId,ourUserId]},    recipient:{$in:[userId,ourUserId]},
}).sort({createdAt: 1}); res.json(messages);
});
```

```
app.get('/people', async (req,res) => { const users = await
User.find({}, {'_id':1,username:1}); res.json(users);
});
```

```

app.get('/profile', (req,res) => {  const token = req.cookies?.token;  if
(token) {
    jwt.verify(token, jwtSecret, {}, (err, userData) => {    if (err) throw err;
res.json(userData);
    });
  } else {
    res.status(401).json('no token');
  }
});

```

```

app.post('/login', async (req,res) => {  const {username, password} =
req.body;  const foundUser = await User.findOne({username});  if
(foundUser) {
    const    passOk    =    bcrypt.compareSync(password,
foundUser.password);

    if (passOk) {
        jwt.sign({userId:foundUser._id,username}, jwtSecret,
{}, (err, token) => {
            res.cookie('token',    token,    {sameSite:'none',
secure:true}).json({    id: foundUser._id,
                });
        });
    }
  }
});

```

```

app.post('/logout', (req,res) => {

    res.cookie('token', "",    {sameSite:'none', secure:true}).json('ok');

```

```
});
```

```
app.post('/register', async (req,res) => {  const {username,password} =
req.body;
  try {
    const hashedPassword = bcrypt.hashSync(password, bcryptSalt);
    const createdUser = await User.create({    username:username,
password:hashedPassword,
    });
    jwt.sign({userId:createdUser._id,username}, jwtSecret,
    {}, (err, token) => {    if (err) throw err;
      res.cookie('token',    token,    {sameSite:'none',
secure:true}).status(201).json({    id: createdUser._id,
    });
    });
  } catch(err) {    if (err) throw err;    res.status(500).json('error');
  }
});
```

```
const server = app.listen(4040);
```

```
const wss = new ws.WebSocketServer({server}); wss.on('connection',
(connection, req) => {
```

```
  function notifyAboutOnlinePeople() {    [...wss.clients].forEach(client
=> {    client.send(JSON.stringify({
      online:    [...wss.clients].map(c    =>
({userId:c.userId,username:c.username})),
```

```

    });
  });
}

```

```

connection.isAlive = true;

```

```

connection.timer = setInterval(() => { connection.ping();
  connection.deathTimer = setTimeout(() => { connection.isAlive =
false; clearInterval(connection.timer); connection.terminate();
notifyAboutOnlinePeople(); console.log('dead');
  }, 1000);
}, 5000);

```

```

connection.on('pong', () => { clearTimeout(connection.deathTimer);
});

```

```

// read username and id form the cookie for this connection
const cookies = req.headers.cookie; if (cookies) {
  const tokenCookieString = cookies.split(';').find(str =>
str.startsWith('token=')); if (tokenCookieString) {
    const token = tokenCookieString.split('=')[1]; if (token) {
      jwt.verify(token, jwtSecret, {}, (err, userData) => { if (err) throw
err;
        const {userId, username} = userData; connection.userId =
userId; connection.username = username;
      });
    }
  }
}

```

```
}
```

```
connection.on('message', async (message) => {  
    const messageData =  
JSON.parse(message.toString());    const {recipient, text, file} =  
messageData;    let filename = null;  
    if (file) {  
        console.log('size', file.data.length);    const parts =  
file.name.split('.');    const ext = parts[parts.length - 1];    filename =  
Date.now() + '.'+ext;    const path = __dirname + '/uploads/' + filename;  
const bufferData = new Buffer(file.data.split(',')[1],  
'base64');  
  
        fs.writeFile(path, bufferData, () => {    console.log('file  
saved:'+path);  
        });  
    }  
    if (recipient && (text || file)) {  
        const messageDoc = await Message.create({  
sender:connection.userId,    recipient,    text,  
        file: file ? filename : null,  
        });  
        console.log('created message');  
        [...wss.clients]  
            .filter(c => c.userId === recipient)    .forEach(c =>  
c.send(JSON.stringify({    text,  
        sender:connection.userId,    recipient,  
        file: file ? filename : null,  
        _id:messageDoc._id,
```

```
}}));
```

```
}
```

```
});
```

```
// notify everyone about online people (when someone connects)
```

```
notifyAboutOnlinePeople();
```

```
});
```

B. Screenshots

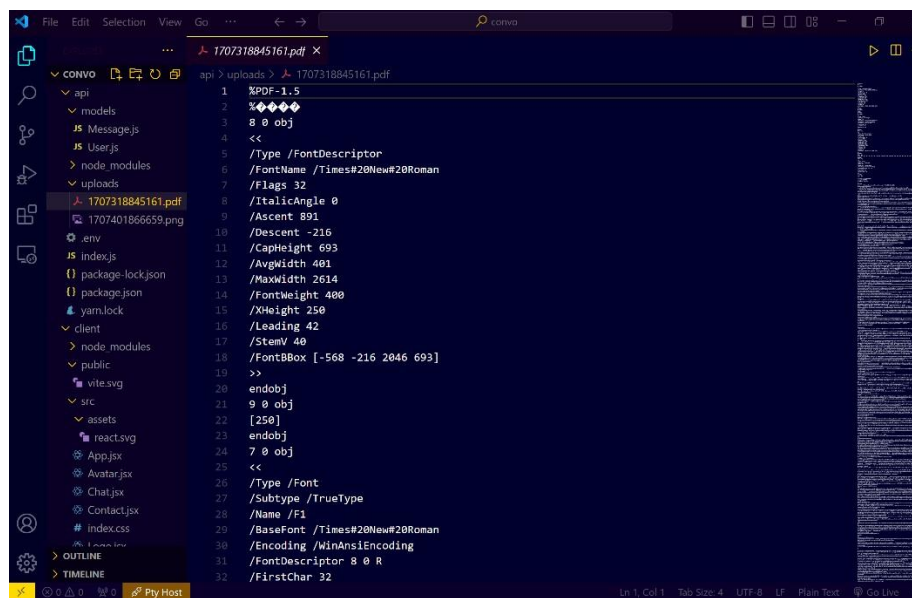


Fig 3.9 Shared pdf file



Fig 3.10 Shared png file

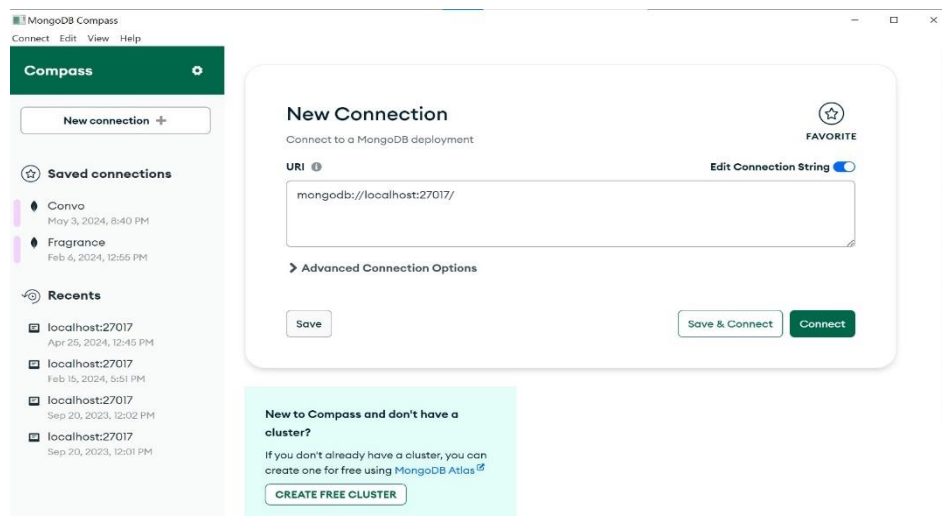


Fig 3.11 Database Connection

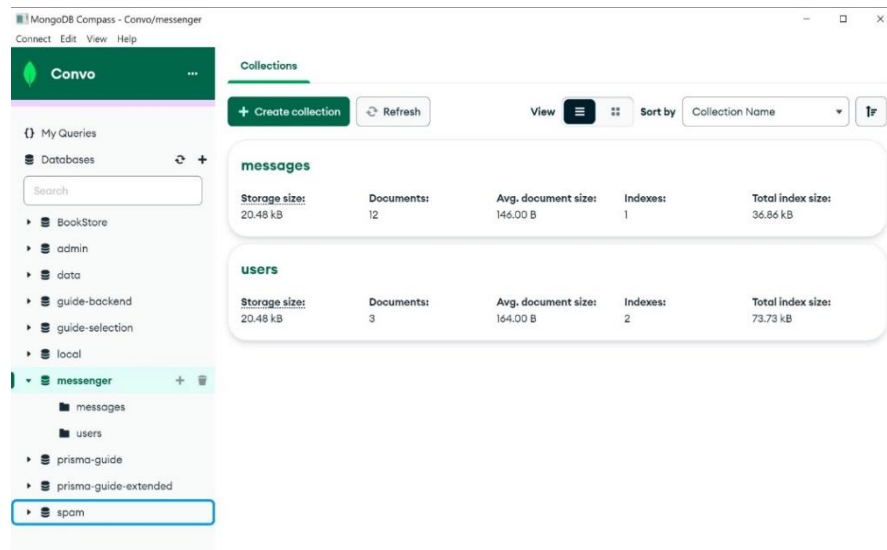


Fig 3.12 Database Collections