TRIBHUVAN UNIVERSITY

Prime College

Nayabazar, Kathmandu, Nepal

An Internship Report On "Generative AI Chatbot" At AMNIL Technologies Pvt. Ltd.

Submitted By

Basanta Rai (23473/076)

An Internship Report Submitted in partial fulfillment of the requirement of **Bachelor of Science in Computer Science & Information Technology (BSc.CSIT) 8th Semester** of Tribhuvan University, Nepal

7th Ashadh, 2081

Generative AI Chatbot

[CSC 412]

An internship report submitted for the partial fulfillment of the requirement for the degree of Bachelor of Science in Computer science & Information Technology awarded by Tribhuvan University.

Submitted By

Basanta Rai (23473/076)

Submitted To

Prime College

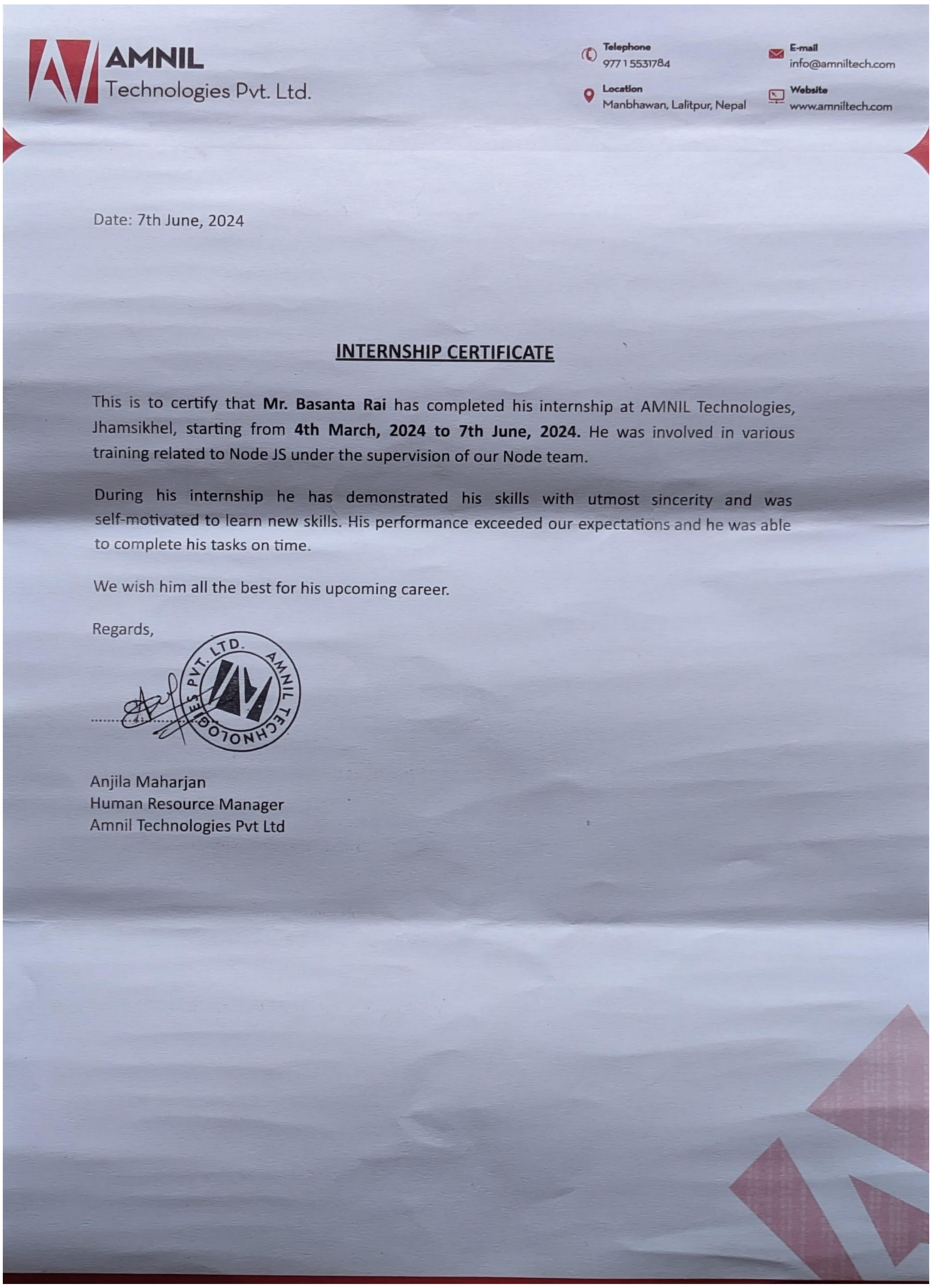
Department of Computer science & IT

Affiliated to Tribhuvan University

Khusibun, Nayabazar, Kathmandu

7th Ashadh, 2081

MENTOR'S RECOMMENDATION



AMNIL
Technologies Pvt. Ltd.

Telephone
977 1 5531784

E-mail
info@amniltech.com

Location
Manbhawan, Lalitpur, Nepal

Website
www.amniltech.com

Date: 7th June, 2024

INTERNSHIP CERTIFICATE

This is to certify that **Mr. Basanta Rai** has completed his internship at AMNIL Technologies, Jhamsikhel, starting from **4th March, 2024 to 7th June, 2024.** He was involved in various training related to Node JS under the supervision of our Node team.

During his internship he has demonstrated his skills with utmost sincerity and was self-motivated to learn new skills. His performance exceeded our expectations and he was able to complete his tasks on time.

We wish him all the best for his upcoming career.

Regards,

AMNIL TECHNOLOGIES PVT. LTD.

[illegible]

Anjila Maharjan
Human Resource Manager
Amnil Technologies Pvt Ltd

Date: 7th Ashadh

SUPERVISOR’S RECOMMENDATION

It is my pleasure to recommend that an internship report on “**Generative AI Chatbot**” has been prepared under my supervision by **Basanta Rai** in partial fulfillment of the requirement of the degree of Bachelor of Science in Computer Science and Information Technology (BSc.CSIT). His report is satisfactory and is an original work done by him to process for the future evaluation.

…………………………………

Er. Sravan Ghimire
Supervisor
Department of Computer Science & IT
Prime College

Date: 7th Ashadh

CERTIFICATE OF APPROVAL

The undersigned certify that he has read and recommended to the Department of Computer Science and Information Technology for acceptance of an internship report entitled "**Generative AI Chatbot**" submitted by **Basanta Rai** in partial fulfillment for the degree of Bachelor of Science in Computer Science and Information Technology (BSc.CSIT), Institute of Science and Technology, Tribhuvan University.

……………………………….	………………………………
Mr. Narayan Prasad Sharma	**Ms. Rolisha Sthapit**
Principal	**Program Coordinator**

………………………………	………………………………
Er. Sravan Ghimire	**Mr. Sarbin Sayami**
Supervisor	**External Examiner**

ACKNOWLEDGEMENT

I extend my heartfelt gratitude to Prime College for affording me the opportunity to explore my passion for Software Development through this internship program. I am immensely thankful to my supervisor, **Er. Sravan Ghimire**, whose unwavering support and guidance have been invaluable in shaping my journey and aiding in the completion of this report.

I also wish to express my deep appreciation to **Mr. Amit Joshi**, the CEO of **AMNIL Technologies Pvt. Ltd.**, for extending the internship opportunity within the organization and for his consistent support and coordination throughout the duration of my internship. Additionally, I am indebted to **Mr. Aaditya Jha**, my mentor and a skilled Software Engineer at AMNIL Technologies Pvt. Ltd., whose assistance and encouragement propelled me to delve deeper into the realm of Software Development.

I am grateful to the entire team at **AMNIL Technologies Pvt. Ltd.** for their warm welcome and for generously sharing their expertise and time with me. Their collective efforts created an environment conducive to learning and growth, without which my internship experience would not have been as enriching.

With respect,

Basanta Rai (23473/076)

ABSTRACT

This comprehensive report encapsulates my enriching journey as a Node.js intern at AMNIL Technologies Pvt. Ltd., shedding light on the tasks and responsibilities I undertook during my tenure. It provides valuable insights into the projects I contributed to and the multifaceted roles I assumed within the organization, along with the methodologies and cutting-edge tools employed in the realm of backend development. Throughout my internship, my primary focus was on API development using Node.js. I had the privilege to work on a transformative project involving the development of a Generative AI chatbot. Leveraging advanced technologies such as OpenAI, Langchain, and vector databases, I honed my skills in backend development and gained hands-on experience in harnessing Large Language Models (LLMs) for real-world applications. This report not only highlights the practical learning experiences and challenges encountered during my internship but also underscores the significance of collaborative teamwork in driving innovation within the domain of backend/API. It serves as a testament to the invaluable skills acquired and lessons learned during my internship journey at AMNIL Technologies Pvt. Ltd.

KEYWORDS: *Node.js, Internship, Backend Development, API, Generative AI Chatbot, OpenAI, Langchain, Chroma DB, Vector Databases, RAG*

TABLE OF CONTENTS

LIST OF ABBREVIATIONS

AI	Artificial Intelligence
API	Application Programming Interface
CEO	Chief Executive Officer
CFO	Chief Finance Officer
CSDO	Chief Service Delivery Officer
COO	Chief Operating Officer
CTO	Chief Technology Officer
DevOPS	Development Operations
GPT	Generative Pre-Trained Transformer
ISO	International Organization for Standardization
LLM	Large Language Models
NLU	Natural Language Understanding
PM	Project Manager
QA	Quality Assurance
RAG	Retrieval Augmented Generation
REST	Representational State Transfer
UI	User Interface
UX	User Experience

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Introduction

The project the intern worked on was part of the Generative AI Chatbot project. The Generative AI Chatbot project leverages advanced technologies such as LangChain, OpenAI GPT-3.5 (a Large Language Model), and ChromaDB (a vector database) to facilitate seamless communication and knowledge extraction using Retrieval Augmented Generation (RAG) capabilities.

The intern focused on building the share conversation feature, an integral component of this project. This feature aimed to enhance user interaction by enabling users to generate, share, and manage conversation links. The intern was responsible for developing the functionality to generate these links, making it easy for users to copy and share them, and ensuring that recipients could access the shared conversations. Additionally, the intern implemented features to update link expiration times and manage link expiration as needed.

This experience provided the intern with valuable hands-on experience in API development, allowing them to apply their skills to solve real-world problems. Overall, the internship significantly contributed to the intern’s growth as a backend developer, equipping them with the practical skills and knowledge necessary for a successful career in the field.

1.2 Problem Statement

Previously, the chatbot at AMNIL Technologies was developed using Botpress, an open-source conversational platform designed for building and managing chatbots. While Botpress provided a basic framework for chatbot development, it presented several drawbacks. The primary limitations included restricted natural language understanding (NLU) capabilities, limited scalability, and challenges in handling complex queries and large datasets. These constraints hindered the chatbot's ability to provide accurate, context-aware responses and limited its effectiveness in dynamic, information-rich environments.

To overcome these limitations and enhance user interaction, the GenerativeAI Chatbot project was initiated. This revolutionary platform aims to provide an interactive and intelligent interface for extracting, ingesting, and querying information. By leveraging advanced technologies such as LangChain, OpenAI GPT-3.5 (a Large Language Model), and ChromaDB (a vector database), the GenerativeAI Chatbot addresses the shortcomings of Botpress.

The GenerativeAI Chatbot project specifically focuses on enabling users to create tenant profiles, ingest data from various sources (PDFs, web pages), and leverage advanced LLM capabilities for interactive communication. Users can send prompt messages, and the GPT-3.5 model, integrated with LangChain and ChromaDB, generates responses based on the ingested data, ensuring a seamless and efficient user experience.

By addressing the limitations of Botpress and integrating cutting-edge technologies, the GenerativeAI Chatbot project aims to revolutionize how users interact with AI-driven systems, providing a more robust, scalable, and intelligent solution.

1.3 Objectives

The main objectives of the project are:

- To enhance user interaction to provide intelligent, context-aware responses for improved user engagement.
- To implement efficient data ingestion to create a robust system for seamless data integration from PDFs, web pages, and other sources.
- To optimize information retrieval to enable rapid and accurate data retrieval to enhance performance and usability.
- To facilitate seamless integration and customization to ensure easy customization and integration with existing systems for flexible use.

1.4 Scope and Limitation

The scope of the project is:

- **High Accuracy**: Utilizing advanced technologies such as OpenAI GPT-3.5 and LangChain ensures intelligent, context-aware responses with high accuracy.

- **Efficient Data Ingestion**: Robust system for seamless integration of data from various sources like PDFs and web pages.
- **Enhanced Performance**: Optimized for rapid and accurate information retrieval, enhancing the chatbot's usability and effectiveness.
- **Flexibility in Integration**: Easily customizable and integrable with existing systems, providing flexibility for various use cases and user requirements.
- **Scalability**: Designed to handle large datasets and complex queries efficiently, ensuring scalability and robust performance.

The limitations are:

- **Dependent on Data Quality**: The performance and accuracy of the chatbot are highly dependent on the quality and comprehensiveness of the ingested data.
- **Computational Resources**: Requires substantial computing resources for optimal performance, which may limit deployment on resource-constrained devices.
- **Complexity of Setup**: Integrating and fine-tuning the advanced technologies may require significant time and expertise.
- **Language and Context Limitations**: While advanced, the chatbot's ability to understand and generate responses may still be limited by the nuances of certain languages and specific contexts not well-represented in the training data.
- **Real-Time Processing**: The need for real-time data processing and response generation may impose additional constraints on system performance and resource allocation.

1.5 Report Organization

Chapter 1

It shows the overall description of the project. It contains an introduction, objective, reason to take the project, to whom the project favors, and limitations.

Chapter 2

It contains the details of the organization in which the internship was done. It describes the organization, the type of organization, its hierarchy, its working domain, and the departmental unit within an organization.

Chapter 3

It covers the actual activity performed during the internship. It describes the role and responsibilities of the student, the work the particular student performed weekly, the project in which the student was involved, and the depth of work he/she has done within the internship period. It focuses on the student's technical work within the specified internship period.

Chapter 4

It describes what a student learns within the internship period and what conclusion can be drawn from the project they were involved in.

CHAPTER 2
ORGANIZATION DETAILS AND LITERATURE REVIEW

2.1 Introduction to Organization

AMNIL Technologies Pvt. Ltd., established in 2009 A.D, is a leading company in the tech industry, offering a variety of web services, including website development, design, and graphic design. The company has two branches in Jhamsikhel and Dhobighat and employs nearly 200 people. AMNIL Technologies serves a diverse range of clients from various industries such as banking, pharmaceuticals, automotive, tourism, and government investment boards.

AMNIL Technologies is known for providing high-quality web solutions tailored to meet the unique needs of both national and international clients. Their services help businesses improve their digital presence and drive growth. The company is ISO certified, which reflects their commitment to quality and excellence. Transparency, collaboration, and customer satisfaction are key aspects of their approach, making them a trusted partner in the tech industry.

Apart from their core services, AMNIL Technologies focuses on innovation and client satisfaction. They aim to remain at the forefront of the tech industry by consistently delivering advanced web solutions that exceed client expectations. They strive to build long-term partnerships with clients based on trust, transparency, and reliability.

Table 1: Company Details

Official Name	AMNIL Technologies Pvt. Ltd.
Location	Dhobighat, Lalitpur, Nepal
Email Address	info@amniltech.com
Website	https://www.amniltech.com/
Office Hours	9:00 am – 6:30 pm
Establishment Year	2009 A.D

The table 1 summarizes the key details of AMNIL Technologies Pvt. Ltd., including its name, location, contact information, office hours, and establishment year.

2.2 Organizational Hierarchy

An organization is usually organized according to the different activities they perform to the extent possible so that the company can run smoothly and achieve its mission, vision and goals. AMNIL Technologies also has a functional organizational structure and the figure 1 shows its organizational hierarchy.

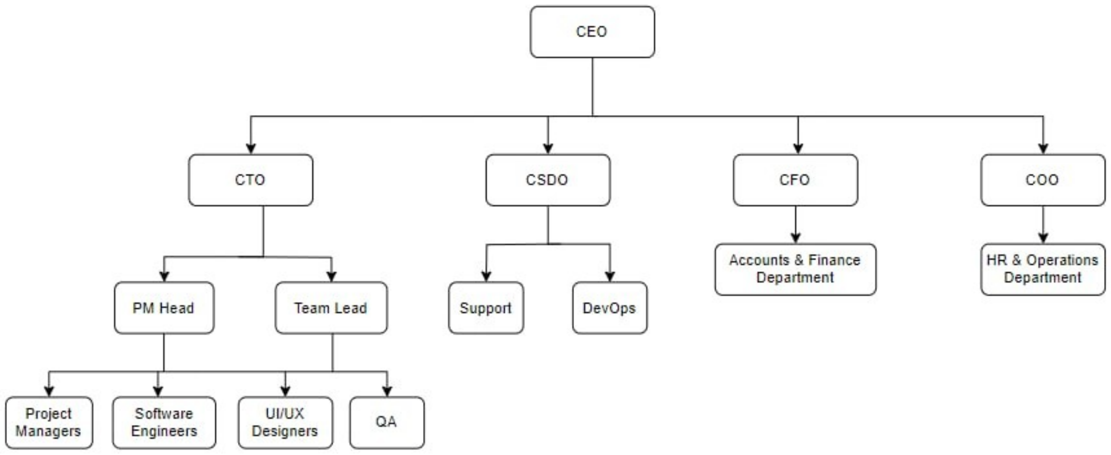


Figure 1: Organization Hierarchy

2.3 Working Domains of Organization

AMNIL Technologies Pvt. Ltd. offers a broad spectrum of services in application and system development, delivering customized tech solutions tailored to meet the unique needs of each client. With a foundation rooted in creative and technical expertise, AMNIL is driven by design and guided by strategy, committed to providing top-tier service to all customers.

2.3.1 Speciality

Web and Mobile Development

The organization has extensive experience in developing dynamic web and mobile applications using modern frameworks such as Vue.js, AngularJS, ReactJS, React Native, and Flutter. These applications are designed to enhance user engagement and deliver seamless experiences across devices.

API Development

The organization specializes in building powerful server-side applications and APIs with technologies like Node.js, PHP/Laravel, .NET Core, and Python. These backend systems are designed to be scalable, secure, and high-performing.

UI/UX Design and Implementation

With a strong focus on user experience, the organization uses tools like Figma to design intuitive and visually appealing interfaces. The team collaborates closely with clients to ensure that the final product aligns with their vision and enhances user satisfaction.

Integration Services

The organization provides integration services for SMS, Viber, and WhatsApp messaging, enabling businesses to enhance their communication strategies and reach their audience more effectively.

Desktop Application Development

The organization develops robust and user-friendly desktop applications using technologies such as Electron, ensuring that applications are cross-platform and feature-rich to meet diverse business needs.

2.4 Description of Intern Department/Unit

The department in which the author interned is the Web Development Department at AMNIL Technologies. This department plays a pivotal role in the company's technological ecosystem, focusing on the development of robust and efficient web-based solutions.

Within the Web Development Department, two primary teams drive the development process: Front-End Development and Back-End Development.

Front-End Development: This team specializes in crafting user-centric interfaces that prioritize usability and aesthetics. Their responsibilities include translating design mockups into interactive web elements, optimizing user experience, and ensuring cross-browser compatibility.

They work collaboratively with the UI/UX department to create visually compelling and intuitive interfaces that resonate with end-users.

Back-End Development: The Back-End Development team is responsible for building the backbone of web applications, comprising server-side logic, databases, and APIs. They architect scalable and secure systems, handle data management, and implement complex business logic to support the functionality of the application. Their expertise ensures seamless communication between the front-end interface and the underlying infrastructure, facilitating smooth data flow and efficient application performance.

Both teams operate under the guidance of dedicated team leads who oversee project execution, manage resources, and ensure alignment with organizational goals. This collaborative structure fosters synergy between front-end and back-end development efforts, resulting in cohesive and impactful web solutions that meet the needs of clients and end-users alike.

2.4.1 Internship Period Details

According to the requirements of the B.Sc. CSIT program at Tribhuvan University, the intern successfully completed a 3-month internship from March 4, 2024, to June 7, 2024.

Table 2: Internship Duration Details

Start Date	4th March, 2024
End Date	7th June, 2024
Duration	3 months
Working Days	Monday to Friday
Office Hours	9:00 am – 6:30 pm
Position	Node.js Intern
Mentor	Aaditya Jha

The table 2 outlines the details of an internship. The internship starts on March 4, 2024, and ends on June 7, 2024, spanning a duration of 3 months. The working days are from Monday to Friday, and the office hours are from 9:00 am to 6:30 pm.

The position held during the internship is "Node.js Intern," and the mentor assigned is Aaditya Jha. This schedule provides a structured and consistent work environment for the intern.

2.5 Literature Review

AI chatbots are advanced artificial intelligence technologies designed for conversational interactions, using natural language processing (NLP) and machine learning (ML) to generate human-like responses. These chatbots perform tasks like answering questions and providing personalized recommendations. Generative AI utilizes large language models (LLMs) like OpenAI and Gemini to create sophisticated chatbots. These advanced chatbots maintain contextual awareness and engage in complex conversations, representing a significant leap in conversational AI.

The introduction of the Transformer model by (Vaswani et al., 2023) marked a breakthrough in generative AI. Transformers utilize a self-attention mechanism that processes entire sentences simultaneously, significantly improving context handling and text generation. OpenAI's Generative Pre-trained Transformer (GPT) models, particularly GPT-3, have demonstrated exceptional capabilities in generating coherent and contextually appropriate responses across various domains (Brown et al., 2020).

Generative AI chatbots now leverage frameworks like LangChain and vector databases to further enhance their capabilities. LangChain integrates large language models (LLMs) with external data sources, and when combined with vector databases like ChromaDB, it implements Retrieval Augmented Generation (RAG) techniques. These advancements improve the chatbot's ability to retrieve and generate relevant information efficiently (Lewis et al., 2021).

Generative AI chatbots represent a significant advancement in conversational AI, leveraging sophisticated models like GPT-3 and frameworks such as LangChain to deliver highly interactive and contextually aware user experiences. While these technologies offer substantial benefits in terms of accuracy and scalability, they also present challenges related to resource demands and integration complexity. Ongoing research and development are essential to address these challenges and fully realize the potential of generative AI chatbots in various applications.

CHAPTER 3
INTERNSHIP ACTIVITIES

3.1 Roles and Responsibilities

The intern was employed by **AMNIL Technologies Pvt. Ltd.** as a Node.js intern. Under the mentorship of **Mr. Aditya Jha**, the intern's role was to learn and apply various technologies related to backend and API development. During the internship, the intern gained extensive knowledge and skills through direct guidance and hands-on experience. The major tasks and responsibilities assigned included:

- API development with Express.js
- Database management with PostgreSQL and MongoDB
- Implementing unit and integration testing
- Working with TypeScript for enhanced type safety
- Developing and managing RESTful services

3.2 Weekly Log

Following table shows the weekly activities the intern performed throughout their internship period.

Table 3: Weekly Log

Week	Activities
Week 1	• Learned basics of Javascript • Understood variable hoisting, lexical scope, and execution context • Explored the event loop and callback queue • Learned about Promises, Closures, async/await, and first-class functions • Understood event listeners and event handlers
Week 2	• Assignment:

	o Completed basic JavaScript exercises o Performed DOM manipulation tasks • Reviewed and incorporated feedback on JavaScript assignments
Week 3	• Introduction to Node.js • Set up and installed Node.js • Explored core modules and created a simple HTTP server • Practiced creating simple server applications with http module • Introduced to asynchronous programming in Node.js.
Week 4	• Introduced to Express.js and its core features • Learned how to set up a basic Express.js server • Understood middleware and routing • Reviewed relational (PostgreSQL) and non-relational (MongoDB) databases
Week 5	• Connected Express.js to PostgreSQL and MongoDB • Created a basic Express.js server with database integration • Implemented the Model-View-Controller pattern using Express. • Learned best practices for error handling in Express.js • Assignments: o Implemented an MVC pattern with error handling. • Received and incorporated feedback on MVC and error handling assignment
Week 6	• Introduction to REST API development • Designed RESTful services • Implemented token-based authentication and role-based authorization • Created API documentation and database design • Developed REST API endpoints and documentation • Assignments: o Developed REST API and documentation • Received and incorporated feedback on REST API assignment
Week 7	• Learned the basics of TypeScript and its integration with Node.js • Rewrote API components using TypeScript for type safety

	• Implemented DTOs for structured data exchange • Documented APIs with swagger • Learned the principles and best practices of unit testing • Wrote unit tests for individual API components using jest
Week 8	• Introduced to Loopback and built APIs with it • Understood Loopback's features and benefits • Developed a simple API using Loopback
Week 9	• Learned and practiced version control with Git for managing code changes • Studied the source code and features of the existing Generative AI Chatbot
Week 10	• Created database and API design documents for share conversation feature • Received and incorporated feedback on database and API design
Week 11	• Created database entities for share conversation as per the design document • Developed POST and PATCH APIs for generating and updating share conversation links • Developed GET API for accessing shared conversations • Reviewed and integrated feedback on API development
Week 12	• Implemented functionality to update or expire shared conversation links • Ensured secure handling and storage of conversation data • Performed code reviews and refactored code for optimization • Conducted final testing and bug fixes • Prepared final documentation and deployment of the feature

3.3 Description of Project Involved During Internship

During the internship from March 4th to June 7th, 2024, the intern was deeply involved in the development of the share conversation feature as part of the Generative AI Chatbot project at AMNIL Technologies.

Generative AI Chatbot

The Generative AI Chatbot project at AMNIL Technologies aims to revolutionize conversational AI capabilities by harnessing advanced technologies such as OpenAI, LangChain, and vector databases. The GenerativeAI Chatbot project is specifically designed to empower users in creating tenant profiles, extracting data from diverse sources like PDFs and web pages, and leveraging advanced Large Language Model (LLM) capabilities for interactive communication. Through prompt messages, users can engage with the GPT-3.5 model, which, integrated with LangChain and ChromaDB, generates contextually relevant responses based on ingested data, ensuring a seamless and effective user experience.

3.3.1 System Requirements

Requirement analysis can be carried out in following ways:

3.3.1.1 Functional Requirement

Functional requirements provide an overview of how the Share Conversation feature operates. The following functionalities are required:

- **Share Conversation Button:** The system should include a user interface element, such as a button, to initiate the sharing process.
- **Expiration Time Setting:** Users should be able to set the desired expiration time for the shared conversation link.
- **Generate and Copy Link:** Functionality must be provided for users to generate and copy a unique link for the shared conversation.
- **Link Management:** The system should manage the lifecycle of shared conversation links, including updating expiration times and expiring links.
- **Secure Link Generation:** Unique, secure links should be generated for each shared conversation to ensure data privacy and security.
- **Link Tracking:** The system must track the status of shared links, indicating whether they are active, expired, or updated.

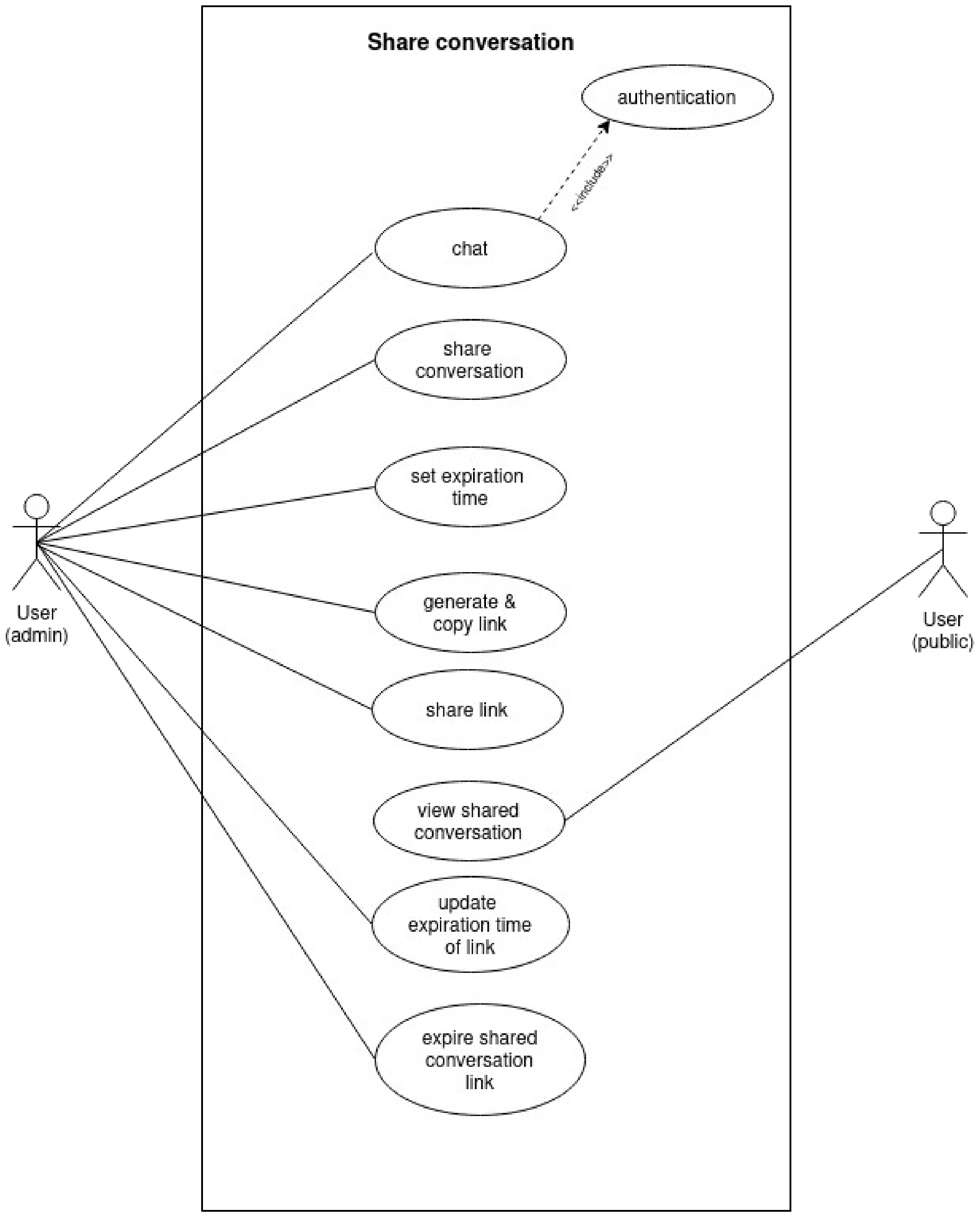


Figure 2: Use Case Diagram Of Share Conversation

Figure 2 illustrates the fundamental functional requirement of the Share Conversation feature. Users engage in conversation with the Chatbot and generate a share conversation link with a specified expiration time. This lik can then be shared with recipients, allowing them to access the shared conversation. The generator retains the ability to update or expire the shared link as needed.

3.3.1.2 Non-Functional Requirement

Non-functional requirements focus on how the system should perform and behave. It describes the characteristics, constraints, and qualities that a system must possess. The points below focus on the non-functional requirement of the system:

Performance

The system should respond promptly to user requests for generating, updating, or expiring share conversation links. Share conversation links should be generated and accessed within acceptable time limits, even under peak loads.

Scalability

The system should be capable of handling a large number of concurrent users without significant degradation in performance. It should be designed to scale horizontally to accommodate increased traffic and user demand.

Reliability

Share conversation links should remain accessible and functional throughout their specified expiration period. The system should minimize the occurrence of errors and ensure robustness against failures.

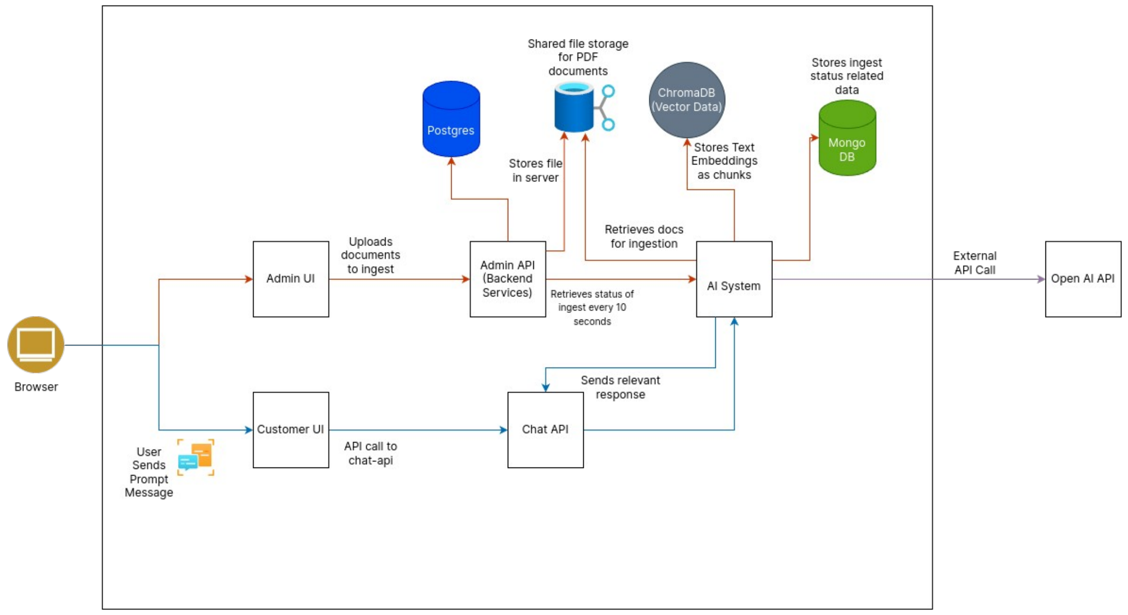
Security

Share conversation links should be securely generated and transmitted to prevent unauthorized access.

3.3.2 System Design

System design refers to the process of creating a detailed plan or blueprint for the architecture, components, and functionality of a system. It involves defining how various parts of a system will work together to achieve the desired objectives and meet specific requirements.

3.3.2.1 Architectural Design



Shared file storage for PDF documents
Postgres
ChromaDB (Vector Data)
Stores ingest status related data
Mongo DB
Stores file in server
Stores Text Embeddings as chunks
Retrieves docs for ingestion
Admin UI
Uploads documents to ingest
Admin API (Backend Services)
Retrieves status of ingest every 10 seconds
AI System
External API Call
Open AI API
Browser
Sends relevant response
Customer UI
Chat API
API call to chat-api
User Sends Prompt Message

Figure 3: Architecture Diagram of Generative AI Chatbot

The figure 3 illustrates the comprehensive Generative AI Chatbot system designed for both user and admin interactions. The system consists of two primary user interfaces: the Customer UI and the Admin UI.

Customers use the Customer UI to interact with the chat API, enabling question-and-answer conversations facilitated by the AI system. As part of the Customer UI, the intern developed the share conversation feature, which allows users to generate, share, and manage conversation links. This functionality makes it easy for users to copy and share these links, ensuring that recipients can access the shared conversations. Additionally, the intern implemented features to update link expiration times and manage link expiration as needed.

Admins utilize the Admin UI to upload documents, which are stored in PostgreSQL and processed by the AI system for ingestion. The Admin API handles document management and ingestion status updates, ensuring the smooth operation of the system. The AI system leverages ChromaDB to store text embeddings and MongoDB to track ingestion status. Additionally, the AI system uses the OpenAI API to generate responses based on the processed data. This architecture ensures efficient and secure data handling, robust backend operations, and a seamless user experience.

3.3.2.2 Analysis

Object modeling using class diagrams

The class diagram for shared conversations is shown in figure 4. It consists of several classes, each with specific attributes and relationships. The User class has attributes like username, email, and phone for user identification. The Conversation class includes attributes like username and userId to track conversations for specific users. The entire conversation history is tracked by the ConversationHistory class. The SharedConversation class contains information about shared conversations, such as the user who shared the conversation, the type of shared conversation, the conversation itself along with its history, its link, and its expiration time.

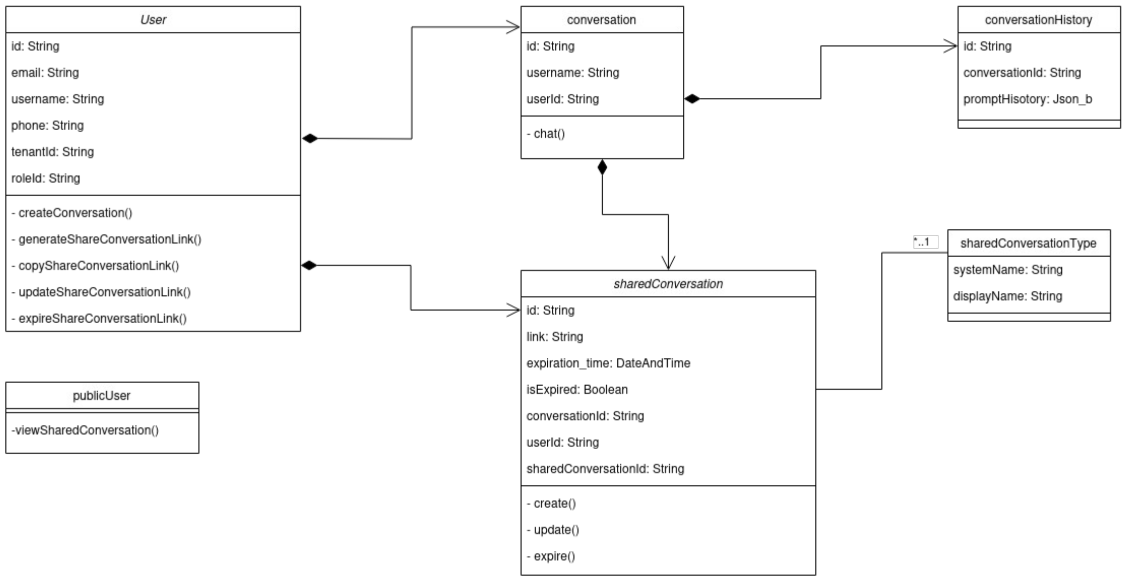


Figure 4: Class Diagram For Share Conversation

Dynamic modeling using sequence diagram

The sequence diagram in Figure 5 illustrates the process of sharing conversations from the admin's perspective. The admin user maintains a history of interactions with the generative AI application, comprising queries and responses. The admin requests the generation of a shareable link for a specific conversation. The system generates this link and provides it to the admin, who can then distribute it to other users for access.

Additionally, the admin can update the link's expiration time or expire the link entirely. By sending a request to the system, the admin can extend the link's validity or invalidate it, ensuring control over the conversation's accessibility. The system processes these requests, updating or expiring the link as needed and confirming the changes to the admin. This sequence ensures secure and manageable sharing of conversation histories.

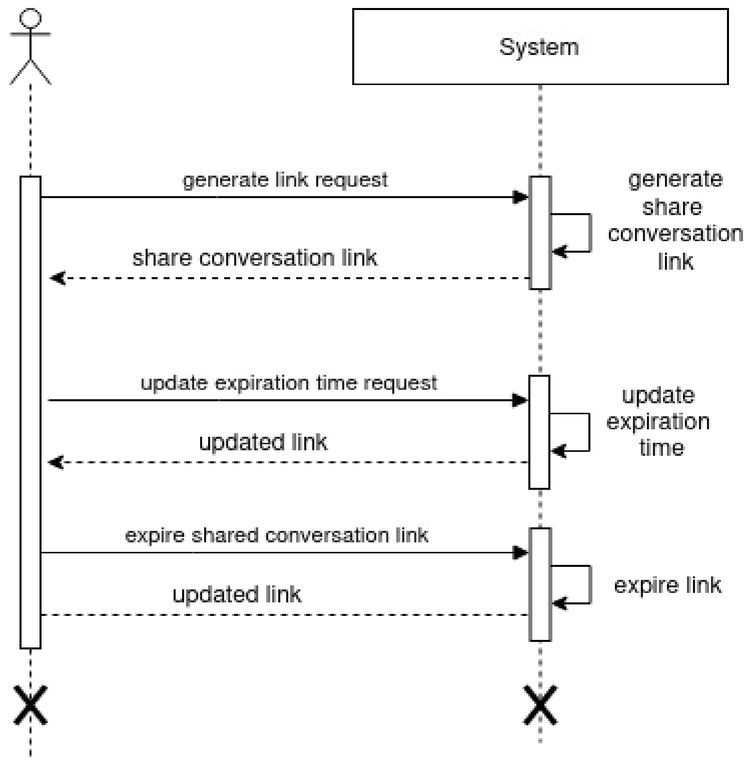


Figure 5: Sequence Diagram For Sharing Conversation

Once the admin shares the link, it can be accessed by public users. Figure 6 illustrates the sequence diagram for accessing the shared conversation. After obtaining the shared conversation link, visitors can use that link to view the conversation without requiring any authentication.

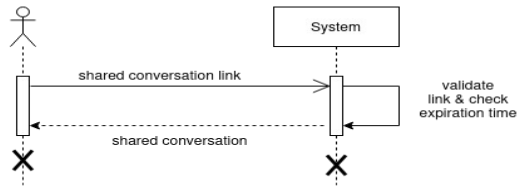


Figure 6: Sequence Diagram For Viewing Shared Conversation

When a visitor clicks on the shared conversation link, the system first validates the link to ensure it is still active and has not expired. If the link is valid, the system retrieves the associated conversation and displays it to the visitor. This allows public users to access and read the conversation seamlessly, without needing any special permissions or login credentials. The streamlined process ensures that shared information is easily accessible while maintaining control over its availability.

Process modeling using activity diagram

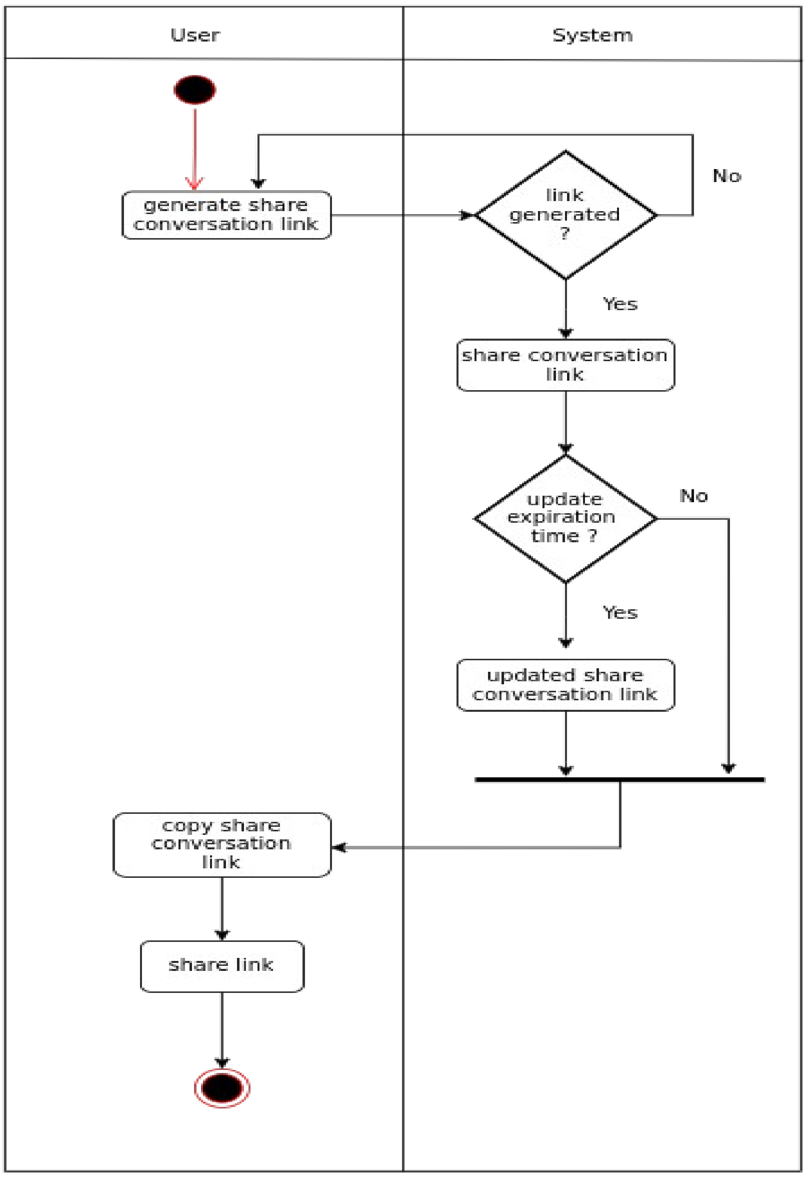


Figure 7: Activity Diagram For Sharing Conversation

The activity diagram for sharing a conversation is shown in the figure 7. The user requests the system to generate a link to share their conversation. Once the link is generated, they are asked if they want to update the expiration time for the link.

They have the option to set a custom expiration time. If they update the expiration time, the link is generated with the user-chosen time; if not, the link is generated with the default expiration time. The user then copies the generated link and shares it with others.

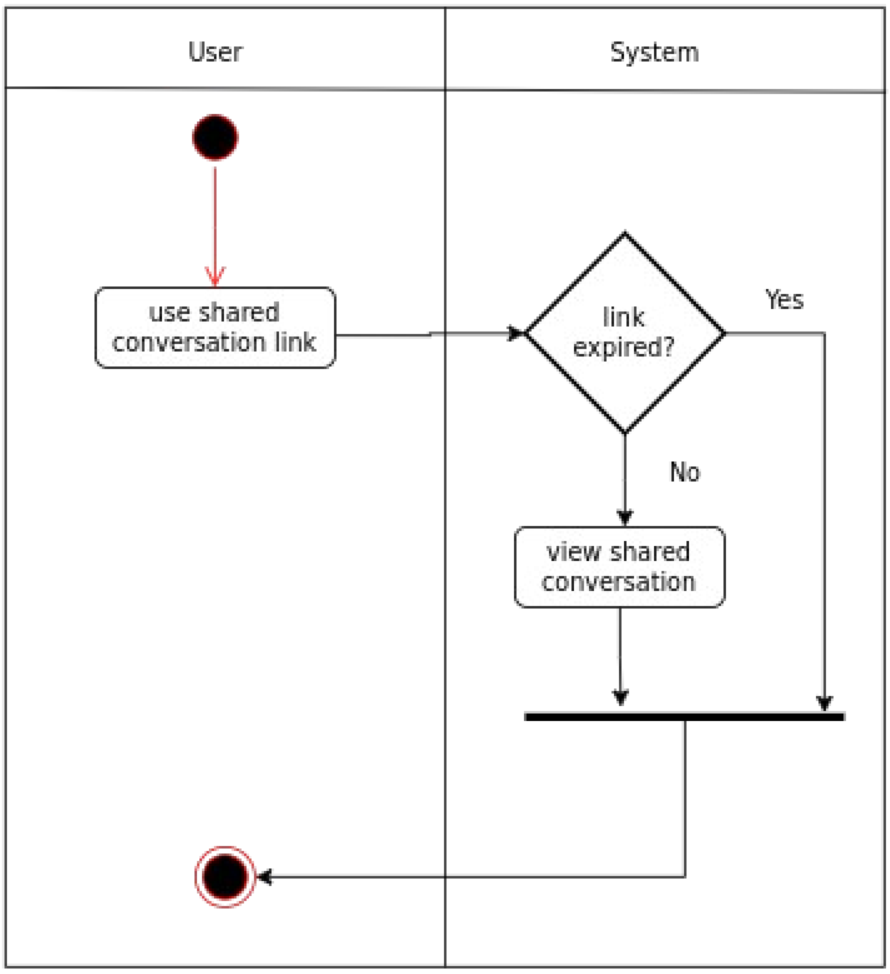


Figure 8: Activity Diagram For Viewing Shared Conversation

The figure 8 shows the activity diagram for viewing a shared conversation. The visitor uses the shared conversation link to access or view the conversation. The system checks the link's validity and expiration time. If the link has expired, the process is terminated; if not, the visitor can view the shared conversation.

3.3.3 Refinement Diagram

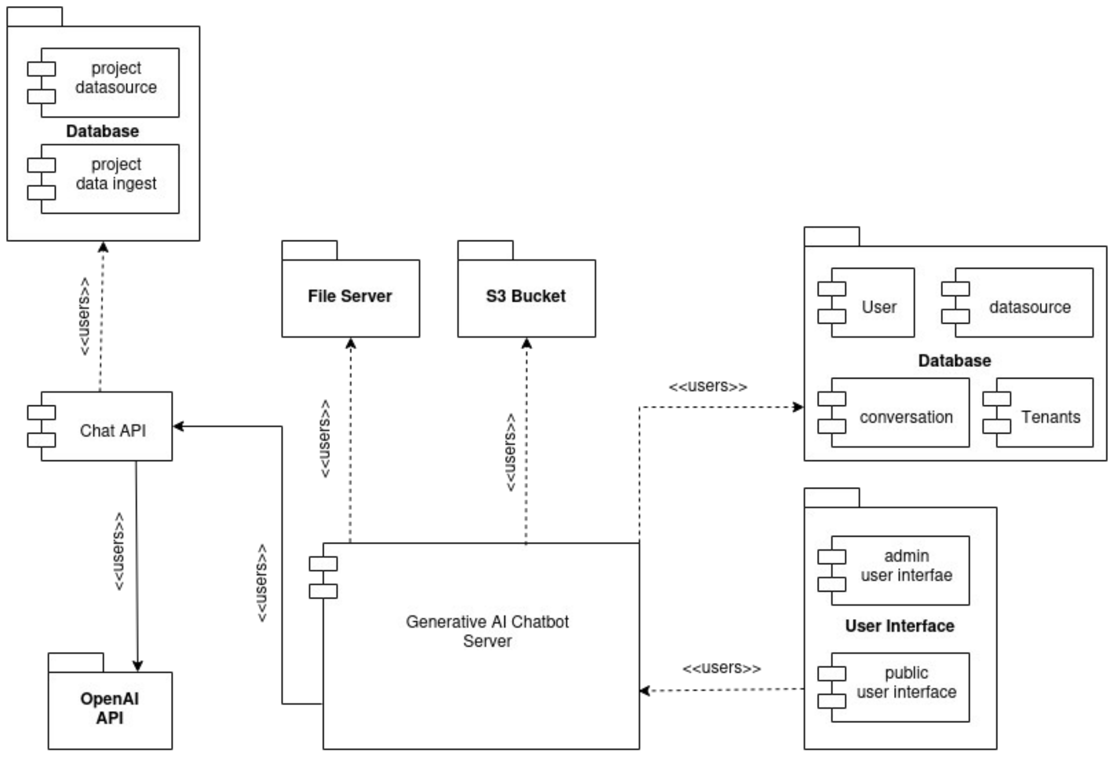


Figure 9: Component Diagram Of The System

The component diagram of the system is shown in the figure 9. The main component of the system is the Generative AI Chatbot Server, which interacts with other components. The system is based on a microservice architecture, with many APIs interacting with each other. The Generative AI Server interacts with the database to retrieve required information and communicates with the chat API. The chat API, in turn, interacts with the OpenAI API and the database to answer user queries requested from the user interface of the GenerativeAI app.

The deployment diagram of the system is shown in the figure 10. In this diagram, users interact with the system through the user interface of the Generative AI Chatbot server. When a user submits a request, the system communicates with the request server to process the request. The request server then interacts with the database server to retrieve and manipulate the necessary data. Once the data has been processed, the response is sent back through the system to the user. The system ensures seamless communication between the user interface, request server, and database server to provide accurate and timely responses to user queries.

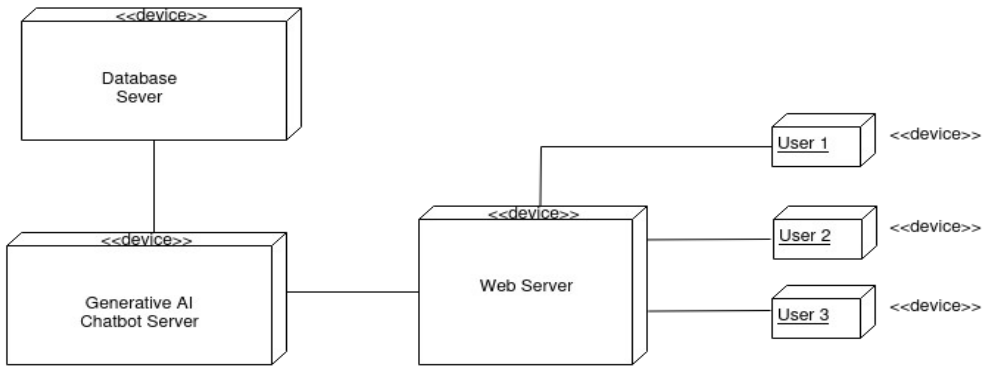


Figure 10: Deployment Diagram Of The System

3.4 Tasks/Activities Performed

Different activities were assigned by the supervisor of the organization. The activities performed as a Node.js intern are as follows:

- Learned the basics of JavaScript, including variable hoisting, lexical scope, and execution context.
- Explored the event loop and callback queue.
- Gained knowledge about Promises, Closures, async/await, and first-class functions.
- Understood event listeners and event handlers.
- Completed basic JavaScript exercises and performed DOM manipulation tasks.
- Set up and installed Node.js.
- Explored core modules and created a simple HTTP server.
- Practiced creating simple server applications using the http module.
- Introduced to asynchronous programming in Node.js.
- Introduced to Express.js and its core features.
- Learned how to set up a basic Express.js server.
- Understood middleware and routing.
- Reviewed relational (PostgreSQL) and non-relational (MongoDB) databases.

- Connected Express.js to PostgreSQL and MongoDB.
- Created a basic Express.js server with database integration.
- Implemented the Model-View-Controller (MVC) pattern using Express.
- Learned best practices for error handling in Express.js.
- Implemented an MVC pattern with error handling as part of an assignment.
- Received and incorporated feedback on MVC and error handling assignments.
- Designed RESTful services.
- Implemented token-based authentication and role-based authorization.
- Created API documentation and database design.
- Developed REST API endpoints and documentation.
- Received and incorporated feedback on REST API assignments.
- Learned the basics of TypeScript and its integration with Node.js.
- Rewrote API components using TypeScript for type safety.
- Implemented Data Transfer Objects (DTOs) for structured data exchange.
- Documented APIs with Swagger.
- Learned the principles and best practices of unit testing.
- Wrote unit tests for individual API components using Jest.
- Introduced to Loopback and built APIs with it.
- Understood Loopback's features and benefits.
- Developed a simple API using Loopback.
- Practiced version control with Git for managing code changes.
- Studied the source code and features of the existing Generative AI Chatbot.
- Created database and API design documents for the share conversation feature.
- Received and incorporated feedback on database and API design.

- Created database entities for share conversation as per the design document.
- Developed POST and PATCH APIs for generating and updating share conversation links.
- Developed GET API for accessing shared conversations.
- Reviewed and integrated feedback on API development.
- Implemented functionality to update or expire shared conversation links.
- Ensured secure handling and storage of conversation data.
- Performed code reviews and refactored code for optimization.
- Conducted final testing and bug fixes.
- Prepared final documentation and deployment of the feature.

3.4.1 System Implementation

The backend and database tools used in development of the system are as follows:

3.4.1.1 Backend Tools

LoopBack

LoopBack is a versatile Node.js framework primarily used for building APIs and connecting applications to data sources. Its robust features and intuitive architecture make it a popular choice for developing scalable and extensible applications. Here's a breakdown of the key features and functionalities that LoopBack offers:

- **Endpoint and Model Configuration**: LoopBack simplifies endpoint and model configuration, allowing developers to define data models and corresponding RESTful API endpoints effortlessly.
- **Built-in Authentication and Authorization**: LoopBack offers built-in support for authentication and authorization mechanisms, including role-based access control (RBAC), OAuth, and integration with third-party authentication providers.
- **Middleware**: Middleware in LoopBack can intercept incoming HTTP requests and responses, enabling developers to implement custom logic such as request validation, logging, and error handling.

- **Data Persistence with Connectors**: LoopBack supports various data connectors, allowing seamless integration with databases such as MongoDB, MySQL, PostgreSQL, and more. Developers can easily configure data sources and interact with them using LoopBack's built-in data access layer.
- **Remote Method Invocation (RMI)**: LoopBack facilitates remote method invocation, enabling applications to expose remote procedures as RESTful endpoints or remote procedure calls (RPCs).
- **Swagger Integration**: LoopBack integrates seamlessly with Swagger, allowing developers to generate API documentation automatically and ensuring consistency and clarity in API design.
- **Command-Line Interface (CLI)**: LoopBack provides a powerful command-line interface (CLI) tool for scaffolding applications, generating models, controllers, and other artifacts, as well as for running tests and managing application dependencies.

With its comprehensive set of features and flexible architecture, LoopBack empowers developers to rapidly build robust APIs and applications, making it an ideal framework for building modern, data-driven systems.

PostgreSQL

PostgreSQL, an advanced open-source relational database management system (RDBMS), is widely utilized in building permit systems for its robust features and reliability across various applications. Here's an overview of how PostgreSQL contributes to the efficiency and effectiveness of permit management:

- **Data Management**: PostgreSQL excels in storing and managing data efficiently, handling permit-related information such as applications, documents, and inspection records with ease. Its architecture ensures data integrity and reliability, crucial for maintaining accurate permit records.
- **Advanced Query Capabilities**: PostgreSQL offers powerful query capabilities, supporting complex queries and indexing strategies for fast data retrieval and optimized database operations. This capability enhances the performance of permit system applications, especially when dealing with large datasets and complex queries.

- **ACID Compliance**: Similar to MySQL, PostgreSQL adheres to the ACID principles (Atomicity, Consistency, Isolation, Durability), guaranteeing reliable transactions and maintaining data consistency. This ensures that permit-related operations are executed reliably, without compromising data integrity.
- **Security Features**: PostgreSQL provides robust security features to safeguard permit system data. It supports user authentication, role-based access control (RBAC), and data encryption, enhancing the confidentiality and integrity of permit-related information. These security measures help ensure that only authorized users have access to sensitive permit data, thereby mitigating security risks and maintaining compliance with data protection regulations.

3.4.1.2 Development Methodology

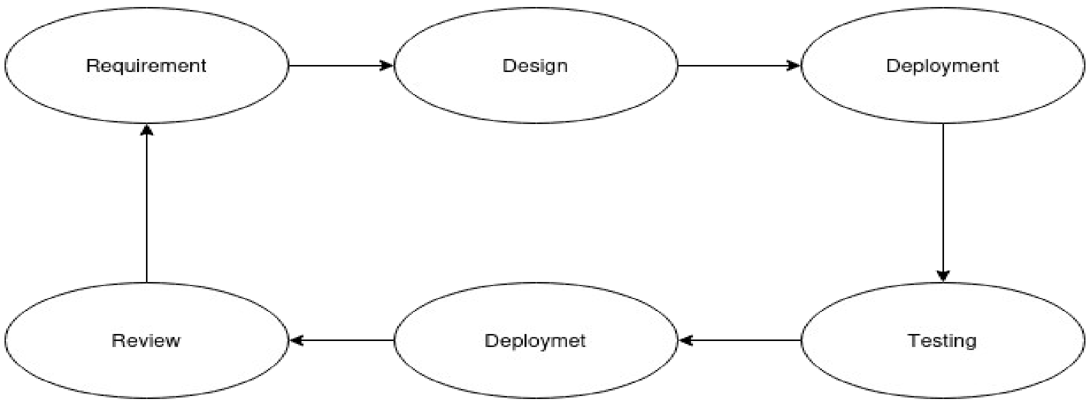


Figure 11: Agile Development Methodology

The Agile development methodology is a cooperative and iterative strategy for software development, highlighting flexibility, adaptability, and customer involvement. Its objective is to provide top-notch software within shorter development cycles by segmenting the project into smaller parts called iterations or sprints. Agile practices prioritize personal connections, functional software, customer engagement, and the ability to adapt to change, rather than adhering strictly to procedures and extensive documentation.

Requirement

Agile starts with collecting and prioritizing stakeholder needs, expectations, and constraints, often through collaborative workshops and discussions. This ensures a clear understanding of project objectives and user requirements.

Design

Instead of a single, extensive design phase, Agile encourages incremental design and prototyping. Developer teams work closely with stakeholders to create prototypes or minimal viable products (MVPs) that can be tested and refined iteratively.

Development

Agile promotes incremental development, where software is built in small, manageable increments called iterations or sprints. Each iteration typically lasts from one to four weeks and results in a potentially shippable product increment.

Testing

Testing is integrated throughout the development process in Agile. Quality Assurance (QA) teams continuously test the software to identify and address defects early. Automated testing tools are often used to streamline the testing process and ensure consistent quality.

Deployment

Agile encourages frequent and incremental deployment of software updates. As features are completed and tested, they are deployed to a staging environment for user feedback and validation. This allows for rapid iteration and adaptation based on real-world usage.

Review

Agile emphasizes regular reflection and review sessions, such as sprint reviews and retrospectives. These meetings bring together the development team, stakeholders, and project managers to evaluate progress, identify areas for improvement, and adapt the approach as needed.

3.4.2 System Testing

System testing entails the comprehensive validation of the integrated system to confirm its alignment with specified requirements. This encompasses evaluating all facets of the system, including functionality, performance, security, and usability.

3.4.2.1 Unit Testing

Table 4: Unit testing for share conversation

Test ID	Description	Test Input Data	Expected Output	Actual Output	Status
TC001	Generate link functionality	N/A	User generates a link for the conversation	Link is generated successfully.	Pass

TC002	Access shared conversation with the link	Shared link	Recipient clicks on the shared link to access the conversation	Shared conversation is accessed successfully	Pass
TC003	Update expiration time of link	New expiration time	User updates the expiration time of the generated link.	Expiration time of the link is updated as per input.	Pass
TC004	Expire link functionality	N/A	User selects to expire the generated link.	The link is no longer accessible after the expiration time.	Pass

CHAPTER 4
CONCLUSION AND LEARNING OUTCOMES

4.1 Conclusion

Throughout the internship, the Node.js intern underwent a comprehensive learning journey, immersing themselves in the intricacies of backend development and API design. They began by mastering fundamental JavaScript concepts, gradually progressing to advanced Node.js topics under the guidance of structured assignments.

As the internship unfolded, the intern transitioned to more complex subjects, such as setting up Node.js environments, utilizing the Express.js framework, and managing databases with PostgreSQL and MongoDB. Their hands-on experience extended to building RESTful APIs, implementing secure authentication mechanisms, and architecting resilient backend systems.

In addition to core technologies, the intern explored cutting-edge tools like TypeScript for enhanced type safety, Swagger for streamlined API documentation, and Jest for robust unit testing. These skills were applied meticulously while developing a share conversation feature, ensuring scalability, security, and codebase maintainability.

Throughout their journey, the Node.js intern actively sought and incorporated feedback on their assignments, leveraging it to refine their skills and deepen their understanding of backend development best practices. By the end of the internship, they emerged equipped with a solid foundation in Node.js development, ready to tackle real-world challenges in the field.

4.2 Learning Outcome

As a backend Node.js intern working on the development of a share conversation feature, they had a transformative learning experience throughout the internship. It provided a unique opportunity to bridge theoretical knowledge with practical application, particularly in the realm of backend development and API design.

The challenges encountered during the project, such as ensuring secure data handling and implementing efficient database management, honed their problem-solving skills and fostered resilience in overcoming obstacles. Collaborating closely with experienced mentors and peers enriched their learning journey, offering invaluable insights and guidance.

Moreover, the internship underscored the importance of project management and effective communication in a professional setting. It equipped them with the necessary skills to work under supervision, meet deadlines, and deliver high-quality outputs.

From selecting appropriate datasets to designing RESTful services and documenting APIs, they gained technical proficiency in various aspects of backend development. Additionally, exposure to different testing methodologies and techniques broadened their understanding of software quality assurance practices.

Overall, the internship provided a comprehensive learning experience that prepared them for future endeavors in backend development and equipped them with the skills and confidence to tackle real-world challenges in this domain.

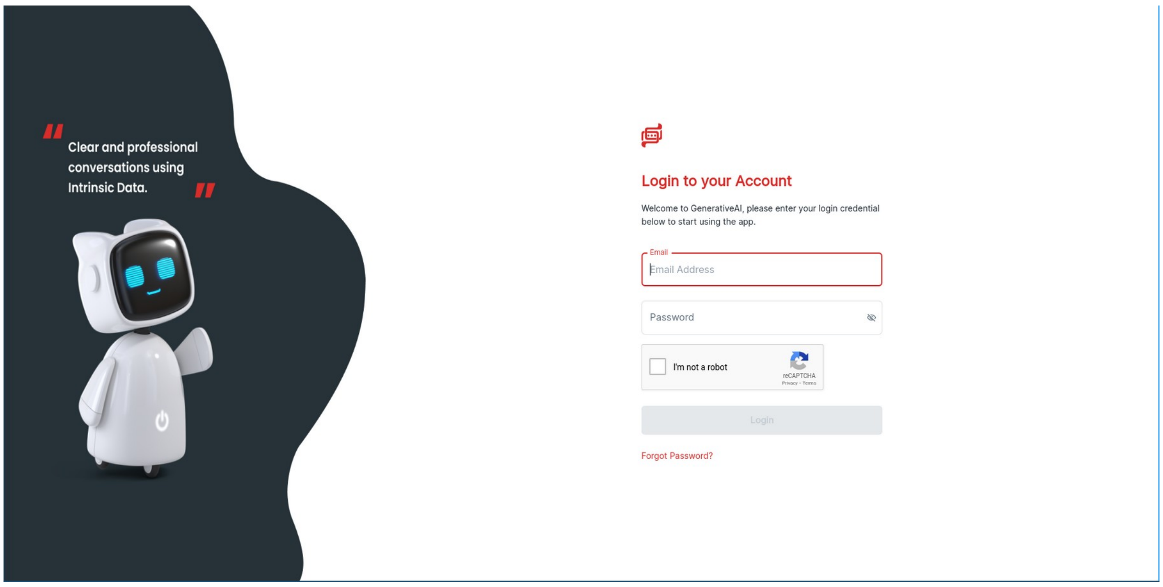
REFERENCES

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., ChiComment: 40+32 pagesld, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., … Amodei, D. (2020). *Language Models are Few-Shot Learners* (arXiv:2005.14165). arXiv. http://arxiv.org/abs/2005.14165

Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W., Rocktäschel, T., Riedel, S., & Kiela, D. (2021). *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks* (arXiv:2005.11401). arXiv. http://arxiv.org/abs/2005.11401

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. http://arxiv.org/abs/1706.03762

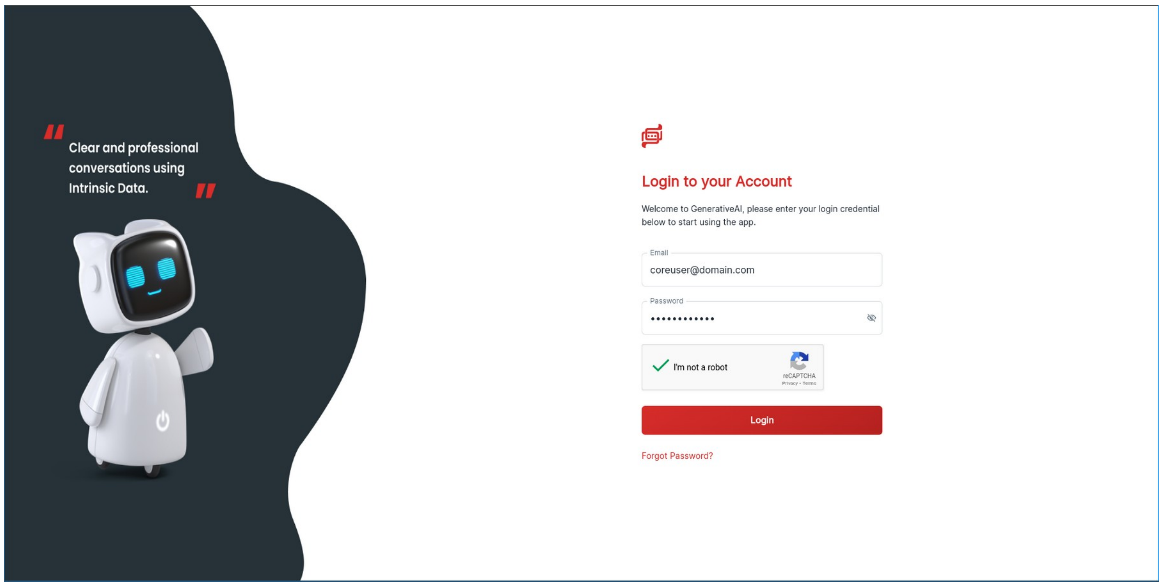
APPENDICES



Login to your Account

Welcome to GenerativeAI, please enter your login credential below to start using the app.

Login Page



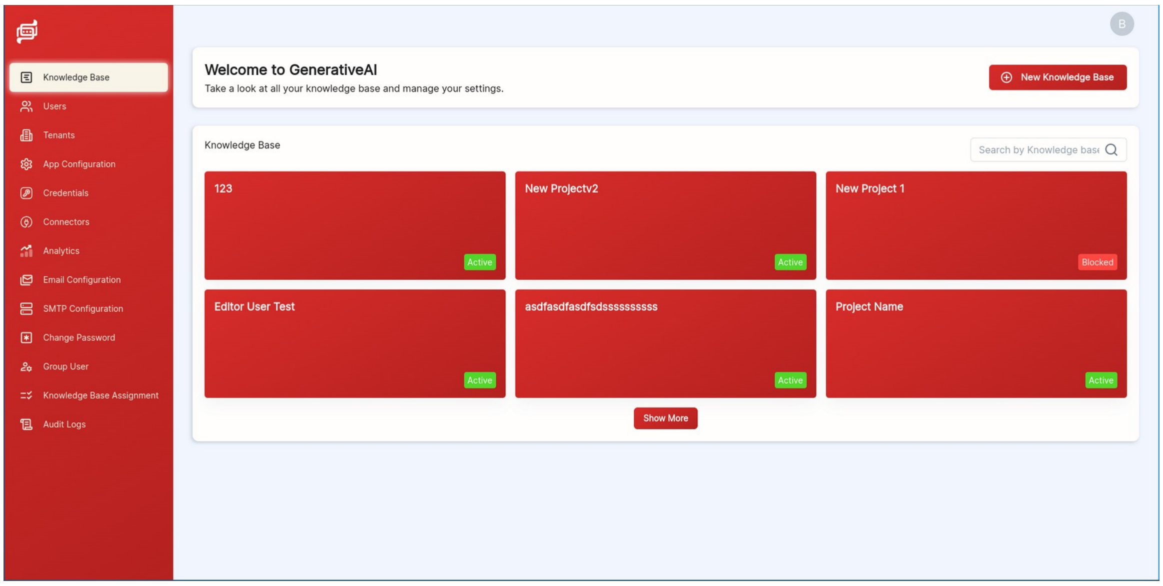
Login to your Account

Welcome to GenerativeAI, please enter your login credential below to start using the app.

Email: coreuser@domain.com

Password: ••••••••••••

Login Page With Credentials



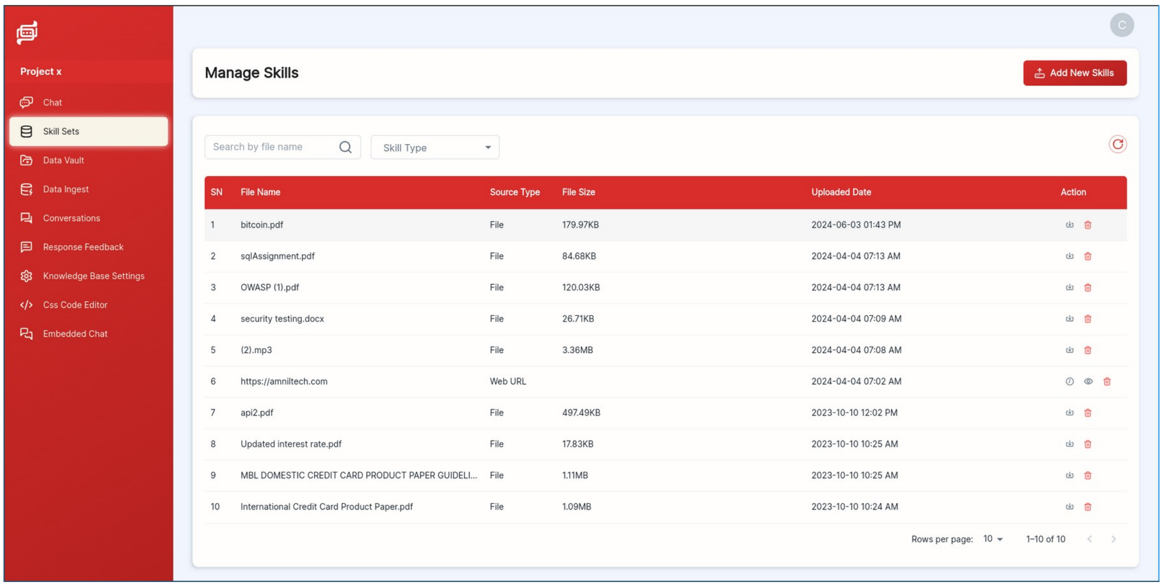
Welcome to GenerativeAI

Take a look at all your knowledge base and manage your settings.

Knowledge Base

123 — Active	New Projectv2 — Active	New Project 1 — Blocked
Editor User Test — Active	asdfasdfasdfsdsssssssss — Active	Project Name — Active

Dashboard

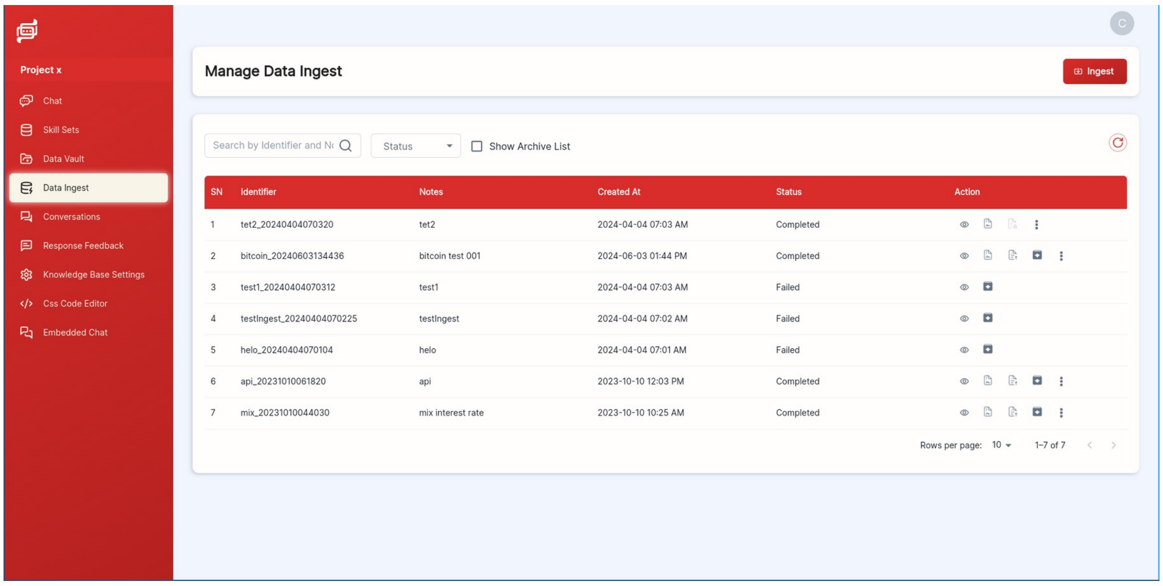


Manage Skills

SN	File Name	Source Type	File Size	Uploaded Date	Action
1	bitcoin.pdf	File	179.97KB	2024-06-03 01:43 PM	
2	sqlAssignment.pdf	File	84.68KB	2024-04-04 07:13 AM	
3	OWASP (1).pdf	File	120.03KB	2024-04-04 07:13 AM	
4	security testing.docx	File	26.71KB	2024-04-04 07:09 AM	
5	(2).mp3	File	3.36MB	2024-04-04 07:08 AM	
6	https://amniltech.com	Web URL		2024-04-04 07:02 AM	
7	api2.pdf	File	497.49KB	2023-10-10 12:02 PM	
8	Updated interest rate.pdf	File	17.83KB	2023-10-10 10:25 AM	
9	MBL DOMESTIC CREDIT CARD PRODUCT PAPER GUIDELI...	File	1.11MB	2023-10-10 10:25 AM	
10	International Credit Card Product Paper.pdf	File	1.09MB	2023-10-10 10:24 AM	

Rows per page: 10 1–10 of 10

Skill Sets

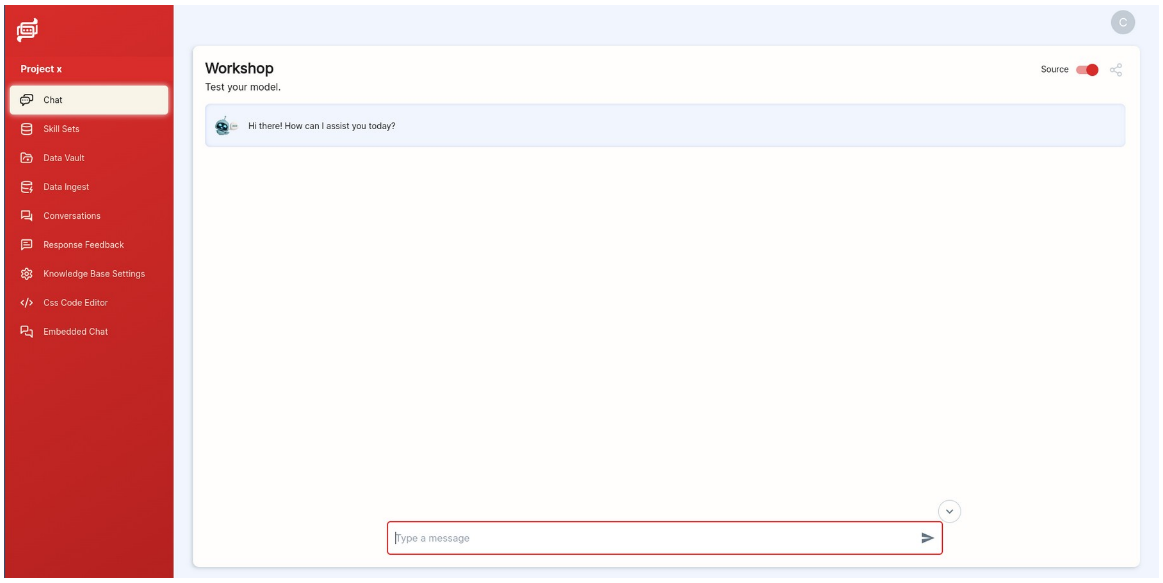


Manage Data Ingest

SN	Identifier	Notes	Created At	Status	Action
1	tet2_20240404070320	tet2	2024-04-04 07:03 AM	Completed	
2	bitcoin_20240603134436	bitcoin test 001	2024-06-03 01:44 PM	Completed	
3	test1_20240404070312	test1	2024-04-04 07:03 AM	Failed	
4	testIngest_20240404070225	testIngest	2024-04-04 07:02 AM	Failed	
5	helo_20240404070104	helo	2024-04-04 07:01 AM	Failed	
6	api_20231010061820	api	2023-10-10 12:03 PM	Completed	
7	mix_20231010044030	mix interest rate	2023-10-10 10:25 AM	Completed	

Rows per page: 10 1–7 of 7

Data Ingests

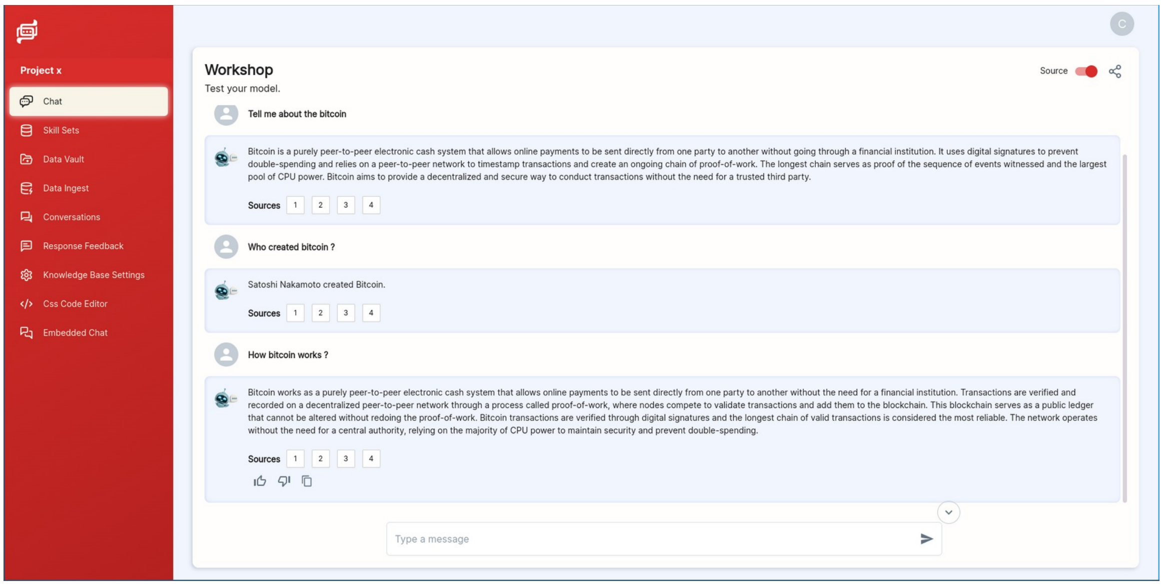


Workshop

Test your model.

Hi there! How can I assist you today?

Chat Interface



Workshop

Test your model.

Tell me about the bitcoin

Bitcoin is a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without going through a financial institution. It uses digital signatures to prevent double-spending and relies on a peer-to-peer network to timestamp transactions and create an ongoing chain of proof-of-work. The longest chain serves as proof of the sequence of events witnessed and the largest pool of CPU power. Bitcoin aims to provide a decentralized and secure way to conduct transactions without the need for a trusted third party.

Sources 1 2 3 4

Who created bitcoin ?

Satoshi Nakamoto created Bitcoin.

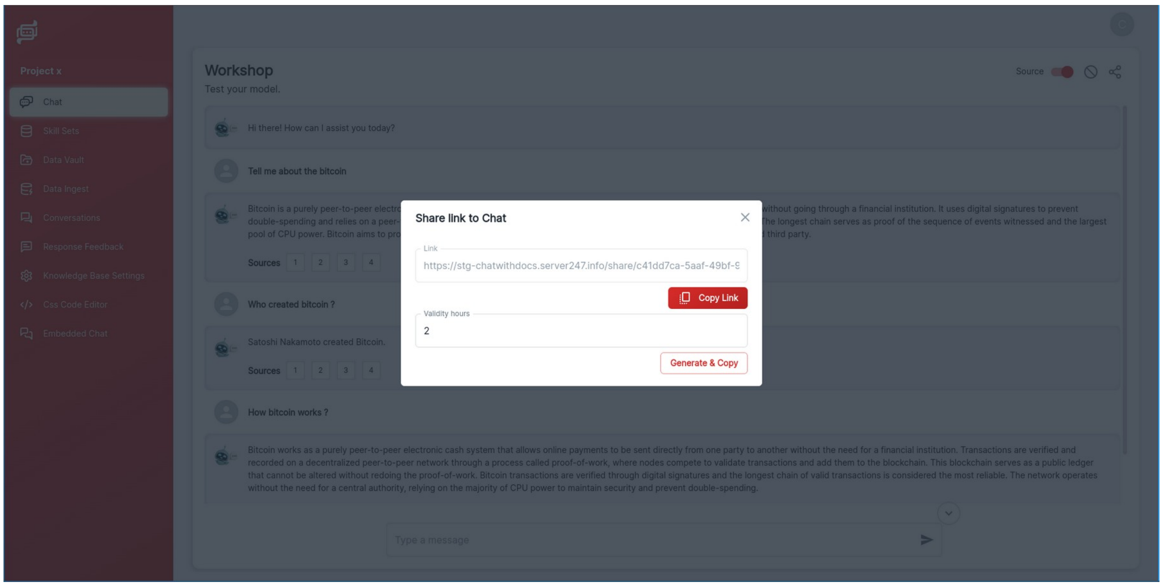
Sources 1 2 3 4

How bitcoin works ?

Bitcoin works as a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without the need for a financial institution. Transactions are verified and recorded on a decentralized peer-to-peer network through a process called proof-of-work, where nodes compete to validate transactions and add them to the blockchain. This blockchain serves as a public ledger that cannot be altered without redoing the proof-of-work. Bitcoin transactions are verified through digital signatures and the longest chain of valid transactions is considered the most reliable. The network operates without the need for a central authority, relying on the majority of CPU power to maintain security and prevent double-spending.

Sources 1 2 3 4

Chat responses



Workshop

Test your model.

Hi there! How can I assist you today?

Tell me about the bitcoin

Share link to Chat

Link: https://stg-chatwithdocs.server247.info/share/c41dd7ca-5aaf-49bf-9

Copy Link

Validity hours: 2

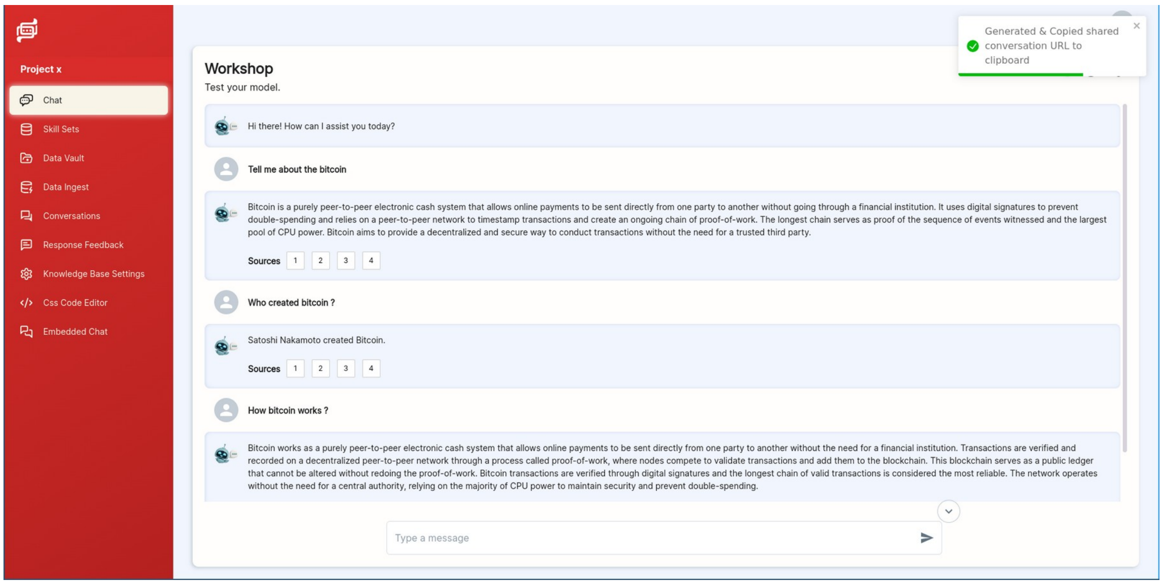
Generate & Copy

Who created bitcoin ?

Satoshi Nakamoto created Bitcoin.

How bitcoin works ?

Share Conversation Popup



Generated & Copied shared conversation URL to clipboard

Workshop

Test your model.

Hi there! How can I assist you today?

Tell me about the bitcoin

Bitcoin is a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without going through a financial institution. It uses digital signatures to prevent double-spending and relies on a peer-to-peer network to timestamp transactions and create an ongoing chain of proof-of-work. The longest chain serves as proof of the sequence of events witnessed and the largest pool of CPU power. Bitcoin aims to provide a decentralized and secure way to conduct transactions without the need for a trusted third party.

Sources 1 2 3 4

Who created bitcoin ?

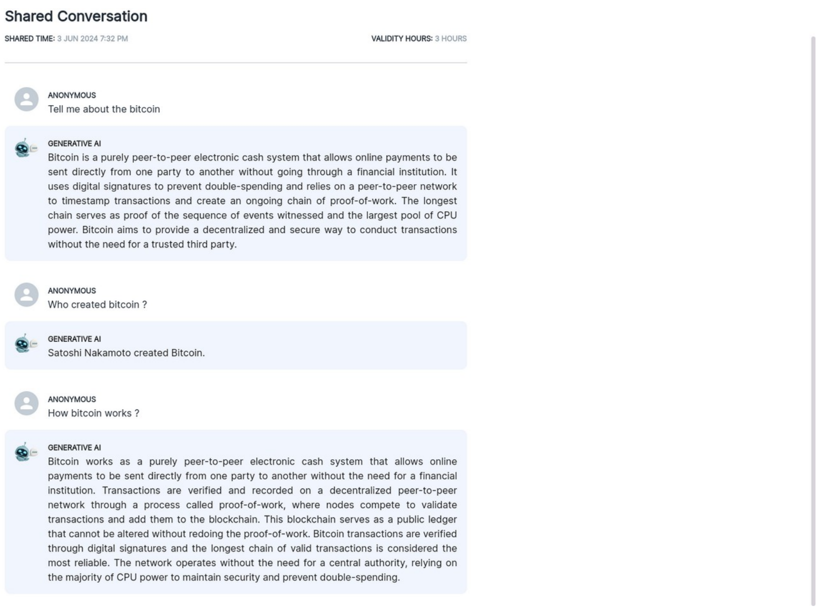
Satoshi Nakamoto created Bitcoin.

Sources 1 2 3 4

How bitcoin works ?

Bitcoin works as a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without the need for a financial institution. Transactions are verified and recorded on a decentralized peer-to-peer network through a process called proof-of-work, where nodes compete to validate transactions and add them to the blockchain. This blockchain serves as a public ledger that cannot be altered without redoing the proof-of-work. Bitcoin transactions are verified through digital signatures and the longest chain of valid transactions is considered the most reliable. The network operates without the need for a central authority, relying on the majority of CPU power to maintain security and prevent double-spending.

Copy share conversation link



Shared Conversation

SHARED TIME: 3 JUN 2024 7:32 PM VALIDITY HOURS: 3 HOURS

ANONYMOUS
Tell me about the bitcoin

GENERATIVE AI
Bitcoin is a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without going through a financial institution. It uses digital signatures to prevent double-spending and relies on a peer-to-peer network to timestamp transactions and create an ongoing chain of proof-of-work. The longest chain serves as proof of the sequence of events witnessed and the largest pool of CPU power. Bitcoin aims to provide a decentralized and secure way to conduct transactions without the need for a trusted third party.

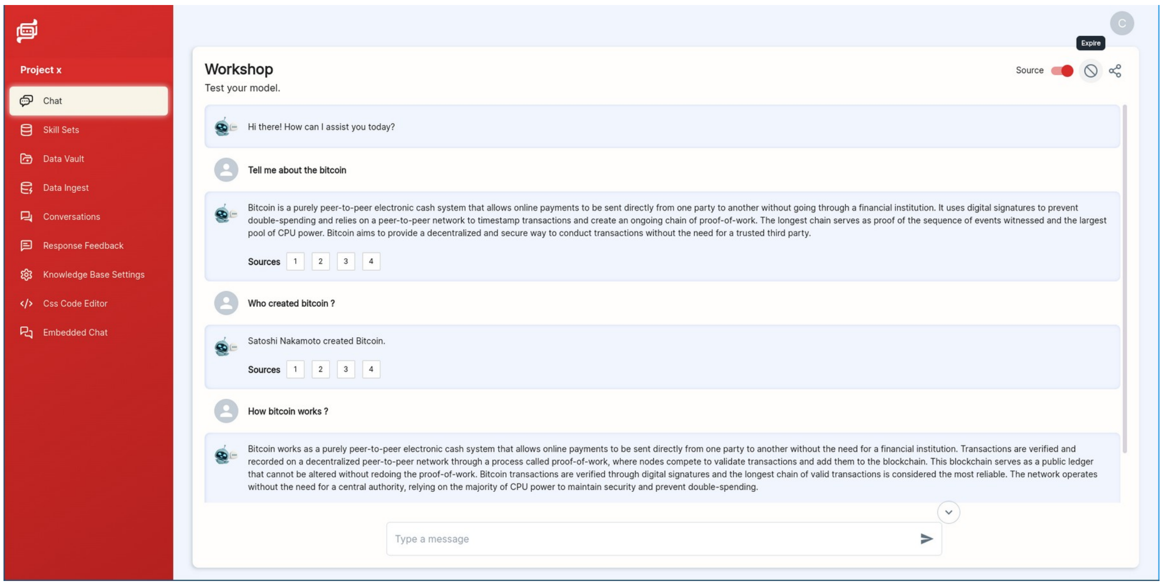
ANONYMOUS
Who created bitcoin ?

GENERATIVE AI
Satoshi Nakamoto created Bitcoin.

ANONYMOUS
How bitcoin works ?

GENERATIVE AI
Bitcoin works as a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without the need for a financial institution. Transactions are verified and recorded on a decentralized peer-to-peer network through a process called proof-of-work, where nodes compete to validate transactions and add them to the blockchain. This blockchain serves as a public ledger that cannot be altered without redoing the proof-of-work. Bitcoin transactions are verified through digital signatures and the longest chain of valid transactions is considered the most reliable. The network operates without the need for a central authority, relying on the majority of CPU power to maintain security and prevent double-spending.

View Shared Conversation



Workshop

Test your model.

Hi there! How can I assist you today?

Tell me about the bitcoin

Bitcoin is a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without going through a financial institution. It uses digital signatures to prevent double-spending and relies on a peer-to-peer network to timestamp transactions and create an ongoing chain of proof-of-work. The longest chain serves as proof of the sequence of events witnessed and the largest pool of CPU power. Bitcoin aims to provide a decentralized and secure way to conduct transactions without the need for a trusted third party.

Sources 1 2 3 4

Who created bitcoin ?

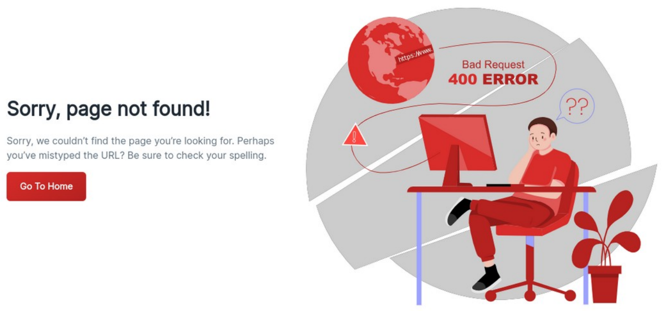
Satoshi Nakamoto created Bitcoin.

Sources 1 2 3 4

How bitcoin works ?

Bitcoin works as a purely peer-to-peer electronic cash system that allows online payments to be sent directly from one party to another without the need for a financial institution. Transactions are verified and recorded on a decentralized peer-to-peer network through a process called proof-of-work, where nodes compete to validate transactions and add them to the blockchain. This blockchain serves as a public ledger that cannot be altered without redoing the proof-of-work. Bitcoin transactions are verified through digital signatures and the longest chain of valid transactions is considered the most reliable. The network operates without the need for a central authority, relying on the majority of CPU power to maintain security and prevent double-spending.

Expire Share Conversation Link



Sorry, page not found!

Sorry, we couldn't find the page you're looking for. Perhaps you've mistyped the URL? Be sure to check your spelling.

Viewing Shared Conversation Using Expired Link