COMPREHENSIVE DATA INTEGRATION AND QUERY SYSTEM FOR ENHANCED DATA MANAGEMENT AND SCIENTIFIC ANALYSIS

A PROJECT REPORT

Submitted by

LOGASUBRAMANI S M [211420104148]

MOHANA KRISHNAN S [211420104163]

NAVIN DURAI S M [211420104182]

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

APRIL 2024

PANIMALAR ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

BONAFIDE CERTIFICATE

Certified that this project report **"COMPREHENSIVE DATA INTEGRATION AND QUERY SYSTEM FOR ENHANCED DATA MANAGEMENT AND SCIENTIFIC ANALYSIS"** is the bonafide work of "**LOGASUBRAMANI S M (211420104148), MOHANA KRISHNAN S (211420104163) NAVIN DURAI SM (211420104182)"** who carried out the project work under my supervision.

SIGNATURE

Dr.L.JABASHEELA,M.E.,Ph.D.,
HEAD OF THE DEPARTMENT

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING COLLEGE,
COLLEGE,NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

SIGNATURE

Dr. MOHANA PRAKASH TA,M.Tech.,Ph.D.,
SUPERVISOR
ASSOCIATE PROFESSOR

DEPARTMENT OF CSE,
PANIMALAR ENGINEERING
NASARATHPETTAI,
POONAMALLEE,
CHENNAI-600 123.

Certified that the above candidate(s) was examined in the End Semester Project

Viva-Voce Examination held on.....

INTERNAL EXAMINER

EXTERNAL EXAMINER

DECLARATION

We **Logasubramani sm (211420104148), Mohana Krishanan S (211420104163), Navin Durai.SM(211420104182)** hereby declare that this project report titled "**Comprehensive Data Integration and Query System for Enhanced Data Management and scientific analysis**", under the guidance of **Dr.Mohana Prakash TA..,** is the original work done by us and we have not plagiarized or submitted to any other degree in any university by us.

LOGASUBRAMANI SM

MOHANA KRISHNAN.S

NAVIN DURAI SM

ACKNOWLEDGEMENT

We would like to express our deep gratitude to our respected Secretary and Correspondent **Dr.P.CHINNADURAI, M.A., Ph.D.** for his kind words and enthusiastic motivation, which inspired us a lot in completing this project.

We express our sincere thanks to our beloved Directors **Tmt.C.VIJAYARAJESWARI**, **Dr.C.SAKTHI KUMAR,M.E.,Ph.D** and **Dr.SARANYASREE SAKTHI KUMAR B.E.,M.B.A.,Ph.D.,** for providing us with the necessary facilities to undertake this project.

We also express our gratitude to our Principal **Dr.K.MANI, M.E., Ph.D.** who facilitated us in completing the project.

We thank the Head of the CSE Department, **Dr. L.JABASHEELA , M.E.,Ph.D.,** for the support extended throughout the project.

We would like to thanks to **Project Coordinator Dr.G.SENTHIL KUMAR, M.C.A., M.Phil., M.E., M.B.A., Ph.D,** and **Project Guide Dr.T.A MOHANAPRAKASH M.Tech, Ph.D.,** and all the faculty members of the Department of CSE for their advice andencouragement for the successful completion of the project.

LOGASUBRAMANI SM

NAVIN DURAI SM

MOHANA KRISHNAN S

ABSTRACT

In today's dynamic landscape of software development, collaboration among developers spanning various domains is essential for driving innovation and project success. However, existing collaboration platforms often lack the seamless integration and user-friendly interfaces necessary to facilitate efficient communication and task management. These limitations result in increased touchpoints and inefficiencies, hindering productivity and project progress.

This platform addresses these challenges by providing a comprehensive solution for seamless collaboration among developers. Central to our platform is the creation of shared to do boards, initiated by project owners, where collaborators can easily contribute by completing tasks and earning recognition for their contributions. By emphasizing user-friendliness and efficiency, our platform minimizes touchpoints and streamlines workflows, enabling developers to focus on their core tasks without unnecessary distractions.A key feature of our platform is its ability to accommodate data input in multiple formats, including HTML, CSV, PDF, and Excel. This data is automatically fetched and organized into a structured user interface, eliminating the need for manual data entry and organization. This automation not only enhances productivity but also reduces the likelihood of errors, thereby fostering innovation and success in projects across organizations and global communities.

TABLE OF CONTENTS

LIST OF FIGURE

LIST OF TABLES

CHAPTER-1
INTRODUCTION

CHAPTER-1
INTRODUCTION

1.1 OVERVIEW

In today's fast-paced world, collaboration among developers from various domains has become increasingly vital for the success of projects, both within organizations and across the globe. Our platform serves as a centralized hub where developers can seamlessly integrate and communicate with peers from different domains, facilitating effective collaboration and teamwork. One of the platform's key features is the creation of a shared todo board, initiated by the project owner, where collaborators can contribute by completing tasks and earning recognition for their contributions.

A primary focus of our platform is to minimize touchpoints and provide a user-friendly experience, ensuring that developers can easily navigate and engage with the project tasks. To streamline the workflow, users can input data in various formats such as HTML, CSV, PDF, and Excel, which is then automatically fetched and organized into a structured user interface. This automation not only saves time but also adds significant value by enhancing efficiency and reducing manual effort.

By offering a platform that promotes seamless collaboration, reduces touchpoints, and provides a user-friendly experience, we aim to empower developers to efficiently work together on projects, ultimately driving innovation and success in organizations and communities worldwide.

1.2 PROBLEM DEFINITION

Collaboration among developers across different domains presents challenges in communication, integration, and task management, hindering project completion and innovation. Existing platforms lack seamless integration and user-friendly interfaces, resulting in inefficiencies and increased touchpoints. Manual data input and organization further compound these challenges, leading to delays and decreased productivity. As a result, there is a pressing need for a comprehensive platform that streamlines collaboration, minimizes touchpoints, and offers a user-friendly experience. This platform should enable developers to easily communicate, integrate, and manage tasks, while automating data input and organization from various formats. By addressing these challenges, the platform aims to enhance collaboration, foster innovation, and drive success in projects across organizations and global communities.

CHAPTER-2
LITERATURE SURVEY

CHAPTER-2
LITERATURE SURVEY

1. **TITLE**: Data Management Architecture for Service-Oriented Maritime Testbeds

YEAR: 2022

AUTHORS: Julius Möller, Dennis Jankowski, Arne Lamm

ABSTRACT:

In recent years, numerous new approaches that rely on data-intensive methods have been developed for maritime assistance systems, leading to a compelling need for more elaborate verification and validation procedures. Modern testbeds that can meet these demands are often developed separately from the system itself and provided as generically usable services. However, the joint usage of such testbeds by multiple stakeholders from research and industry confronts them with various challenges in terms of data management: Data control and protection is required to preserve possible competitive advantages or comply with legal framework conditions. The resulting decentralization in data management complicates collaboration, especially in the joint processing and analysis of testbed data. In this paper, we present a decentralized software system, which can deal with these challenges by modelling interrelationships between the stakeholders in a data space, considering their various interests. With the help of a modular data management architecture, the organization of a testbed data basis, as well as the support of verification and validation processes and the evaluation of data streams is made possible. This is achieved with a workflow model for mapping complex and distributed data processing steps. We demonstrate the applicability of the system in an application scenario for the development of a maritime assistance system.

2. TITLE: Exploring the Fusion Potentials of Data Visualization and Data Analytics in the Process of Mining Digitalization

YEAR: 2023

AUTHORS: Ruiyu Liang, Chaoran Huang, Chengguo Zhang

ABSTRACT:

Mining digitalisation have been receiving significant attention due to the utilisation of advanced technologies, such as IoT, automation, and sensing. However, maximising the potential value of collected data in the mining industry remains a challenge. Therefore, this paper aims to review timely concern topics to facilitate the fusion implementation in mining engineering. Specifically, this review covers recent popular topics, such as, data visualisation, data management, data analytics,data fusion, visual analytics, and mining digital twin construction. In this paper, weaim to draw a comprehensive picture about the fusion of data visualisation and analytics in the big data context, by examining the recent academic research relatedto these topics. Therefore, this paper reviews the visualisation domain by conventional classification, including scientific visualisation, information visualisation, and visual analytics, associated with the analysis of current digital twin development. Next, according to the challenges and issues related to visualisation development, this paper reviews the data management and data analytics domains as well. Incorporating with the fusion concept, machine learning-oriented fusion applications and potential scenarios in the mining industry have been discussed. In addition, based on the observation across various domains, this paper presents challenges and future potentials of data fusion in mining.

3. TITLE: A Big Data Stream-Driven Risk Recognition Approach for Hospital Accounting Management Systems

YEAR: 2023

AUTHORS: Yining Wang, Bin Liang, Tian Wang

ABSTRACT:

This work is confronted with hospital accounting management systems where business volume is usually large and trivial. While designing system prototype and processing algorithms, it is required to integrate realistic big data stream as the main factors for consideration. Because of such point, currently, there still lacks mature solutions for accounting risk recognition in such scenes. Combined with the micro service management technology of data flow, this paper puts forward the risk identification mode and cloud Data integrity verification algorithm for the purpose. Compared with traditional single user authentication techniques, this method has a significantly higher accuracy in hospital data analysis compared to comparative algorithms. At the same time, its error has been reduced. The multi-user parallel authentication algorithm further improves the computational efficiency of the authentication process while ensuring the integrity of data files and reducing the average time. Finally, we also make some empirical analysis on realistic data to testify performance of the proposed technical framework. The results show that the proposal is well suitable for digital risk recognition in hospital accounting management systems. And the recognition accuracy of the proposal can achieve 98%, and is about 22% higher than comparison methods.

4. **TITLE:** Data Enrichment Toolchain: A Data Linking and Enrichment Platform for Heterogeneous Data.

YEAR: 2023

AUTHORS: Luis Sánchez, Jorge Lanza, Juan Ramón Santana

ABSTRACT:

Proliferation of data sources associated to Internet of Things (IoT) deployment as well as those bound to Open Data Portals (e.g. European Data Portal, Municipalities Open Data Portals, etc.) and Social Media platforms is creating an abundance of information that is called to bring benefits for both the private and public sectors, through the development of added-value services, increasing administrations' transparency and availability or fostering efficiency of public services. However, pieces of information without a context are significantly less valuable. Raw data lacks semantics and it is highly heterogeneous from one data-source to another. This poses a challenge to make it useful. To turn all this data into valuable information it is necessary to enable its combination so that meaningful context can be created. Moreover, it is fundamental to define the mechanisms enabling the adoption and orchestration of advanced (typically AI-enabled) data processing techniques to be applied over the harmonized datasets and data-streams. This paper presents the Data Enrichment Toolchain (DET) that provides the necessary harmonization and enrichment to datasets and data-streams coming from heterogeneous sources. The value of the enriched data lies on the one hand in the transfer of the data into a semantically grounded knowledge graph and, on the other hand, in the creation of new data through linking, aggregating and reasoning on the data. In both cases, the benefit of employing linked-data modelling and semantics comes from the extension of the metadata that is associated to every piece of information. Furthermore, the experimental evaluation of the DET implementation that we have carried out is also presented in the paper.

CHAPTER-3
SYSTEM ANALYSIS

CHAPTER-3
SYSTEM ANALYSIS

3.1 EXISTING SYSTEM

GitHub serves as a centralized version control system tailored for software development, enabling code management, change tracking, collaboration, and code review through features like pull requests and issue tracking. Its robust capabilities and integrations make it indispensable for streamlining workflows and fostering collaborative coding environments.

In contrast, Notion provides a versatile workspace for organizing information, managing tasks, and facilitating collaboration across teams. Its flexible structure allows users to create customizable pages, databases, and workflows, catering to various needs from note-taking to project management. Despite its versatility, Notion may pose challenges like a steep learning curve and performance issues in larger workspaces. However, it remains a unified platform where users can consolidate digital activities and collaborate seamlessly.

Overall, GitHub excels in version control and software development workflows, while Notion offers a comprehensive workspace solution for organizing information and fostering collaboration across projects and teams, adapting to the evolving needs of users in the digital age.

DISADVANTAGES:

- Complexity: Both GitHub and Notion present steep learning curves for new users due to their complex interfaces and feature sets, potentially hindering adoption.
- Performance Issues: Users of both platforms encounter performance

problems, such as slow loading times and lags, especially in larger workspaces or with complex content.

3.2 PROPOSED WORK

A comprehensive data parsing system encompasses various stages and functionalities to effectively process and analyze data from diverse sources. It begins with identifying and connecting to data sources like databases, APIs, files, web scraping, and streaming platforms. Upon connection, the system extracts relevant data based on predefined criteria or user-defined queries in its original format. Subsequently, raw data undergoes transformation, including cleaning, normalization, and formatting, to become structured and suitable for analysis. Parsing structured data involves extracting specific fields, values, or patterns according to predefined rules or user configurations. Robust error handling mechanisms ensure the management of exceptions and data quality checks maintain accuracy and consistency. Parsed data is then stored in databases or data warehouses for further analysis and visualization using varioustools. A user-friendly interface allows for easy configuration, scheduling of automated tasks, and report generation. Scalability, security, compliance, and monitoring mechanisms ensure efficient and secure data processing, maintenance, and performance optimization of the system.

Advantages:

- Integration Capabilities: Both platforms offer integration with third-party tools and services, enabling users to connect with a wide range of applications and services to enhance functionality and streamline workflows.

- Community and Support: Our project boast large and active communities of users, providing resources, documentation, and support to help users get the most out of the platforms and troubleshoot any issues they encounter.

3.3 PROJECT REQUIREMENTS

General:

Requirements are the basic constrains that are required to develop a system. Requirements are collected while designing the system. The following are the requirements that are to be discussed.

1. Functional requirements
2. Non-Functional requirements
3. Environment requirements
    A. Hardware requirements
    B. software requirements

1. Functional requirements:

The software requirements specification is the first step in the requirements analysis process. It lists requirements of a particular software system. The following details to follow the special libraries like Python, Mongodb, ReactJs, SpringBoot ,Vscode.

2. Non-Functional Requirements:

Process of functional steps,

1) Get the required fields
2) Add data
3) Processing data using function
4) push data into DB
5) Display to the UI

3. Environmental Requirements: A.Hardware system configuration:

Processor - Intel i3,i5,i7, AMD ProcessorRAM - Minimum 500mb

Hard Disk - Minimum 2gb

B.Software system configuration:

Operating System - Windows 7/8/10/11Front End - React JS

Scripts - Javascript and Java language

Tool -SpringBoot

SOFTWARE DESCRIPTION

JAVASCRIPT

JavaScript is a versatile programming language primarily used for front-end web development. It enables dynamic and interactive content on websites, allowing users to engage with elements like forms, animations, and interactive maps. JavaScript interacts with HTML and CSS to manipulate webpage elements, update content, and respond to user actions in real-time without needing to reload the page. With frameworks like React, Angular, and Vue.js, JavaScript facilitates the creation of complex and responsive user interfaces. Its widespread support across browsers and extensive libraries make it a fundamental tool for building modern, interactive web applications and enhancing user experiences.

JAVASCRIPT INSTALLING PACKAGE:

To install npm packages, use the npm install command followed by the package name. Optionally, specify a version or use npm install to download dependencies listed in package. Json. For global installation, add the -g flag. Use npm uninstall to remove packages. Dependencies are managed through package. Json.

JAVA

Java is a widely-used programming language, particularly in the context of

Spring Boot, a popular framework for building enterprise-level, scalable, and efficient applications. Spring Boot simplifies Java development by providing a comprehensive suite of tools and libraries for rapid application development. It emphasizes convention over configuration, reducing boilerplate code and allowing developers to focus on business logic. Spring Boot utilizes the Java Virtual Machine (JVM), making it platform-independent and compatible with various operating systems. With its modular design, Spring Boot enables the creation of microservices architecture, facilitating scalability and maintainability. Additionally, it integrates seamlessly with other Spring projects, such as Spring Data, Spring Security, and Spring Cloud, offering extensive functionality for building robust and secure applications. Overall, Java's versatility combined with Spring Boot's features makes it a powerful choice for developing modern web and enterprise applications.

SPRING BOOT

To install Spring Boot, first, ensure you have Java Development Kit (JDK) installed on your system. Then, you can set up Spring Boot by either downloading the Spring Boot CLI (Command Line Interface) or using a build tool like Maven or Gradle.For Maven, you can add the Spring Boot starter dependencies in your project's pom.xml file. Alternatively, with Gradle, you configure your build.gradle file to include the necessary Spring Boot dependencies.

Additionally, you can create a Spring Boot project using Spring Initializr, a web - based tool that generates a project structure with all necessary dependencies.

Once your project is set up, you can start building Spring Boot applications and take advantage of its features for rapid development and deployment.

CHAPTER-4
SYSTEM DESIGN

CHAPTER-4

SYSTEM DESIGN

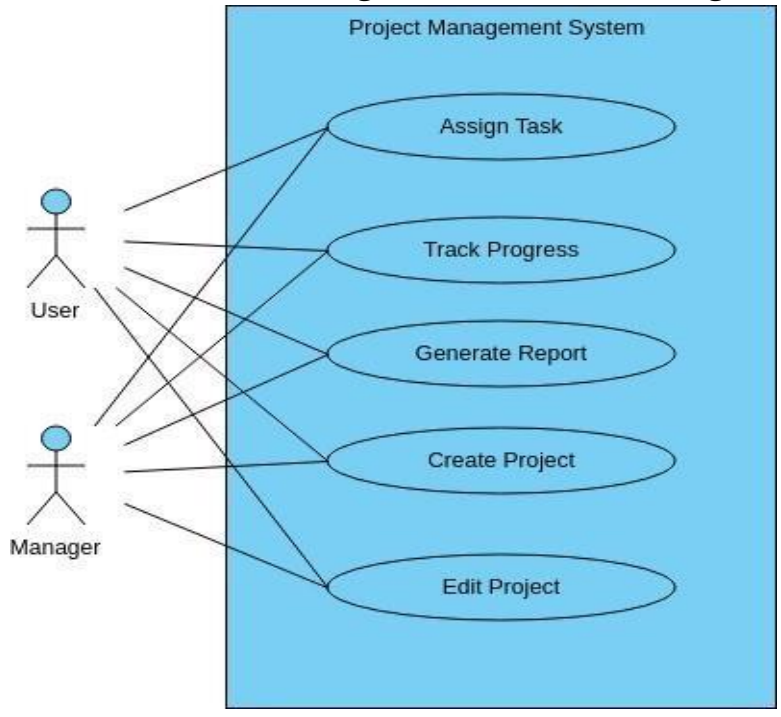
4.1 UML DIAGRAMS

Unified Modeling Language (UML) is a general purpose modelling language. The main aim of UML is to define a standard way to visualize the way a system has been designed. It is quite similar to blueprints used in other fields of engineering.

4.1.1 USE CASE DIAGRAM:

Use case diagrams are considered for high level requirement analysis of a system. When the requirements of a system are analyzed the functionalities are captured in use cases. So, it can say that uses cases are nothing but the system functionalities written in an organized manner.

Figure 4.1.1 Use Case Diagram



4.1.2 ACTIVITY DIAGRAM:

A graphical representation of an executed set of procedural system activities and considered a state chart diagram variation. Activity diagrams describe parallel and conditional activities, use cases and system functions at a detailed level.

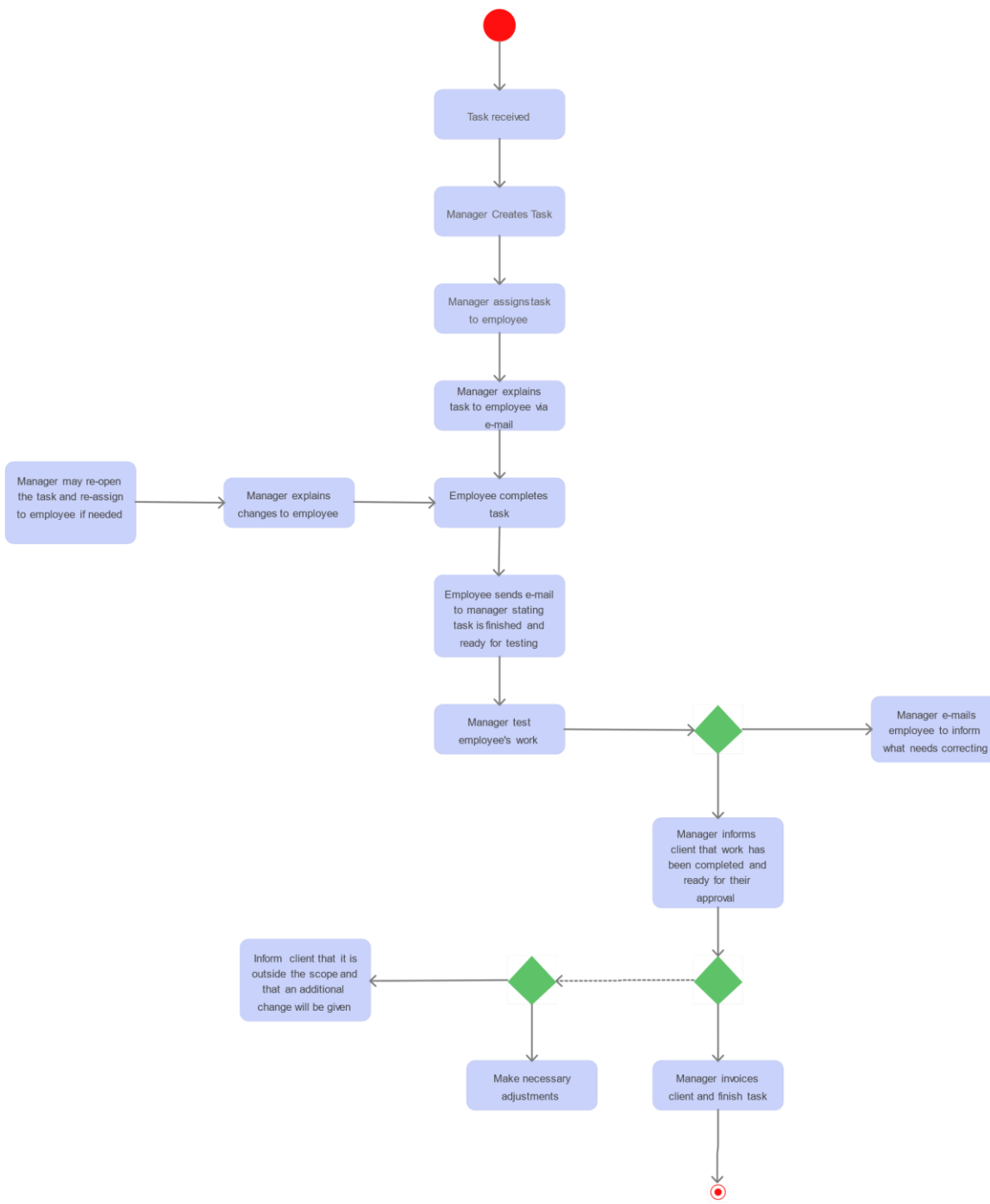


Figure 4.1.2 Activity Diagram

4.1.3 CLASS DIAGRAM:

Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. The name of the class diagram should be meaningful to describe the aspect of the system. Each element and their relationships should be identified in advance.

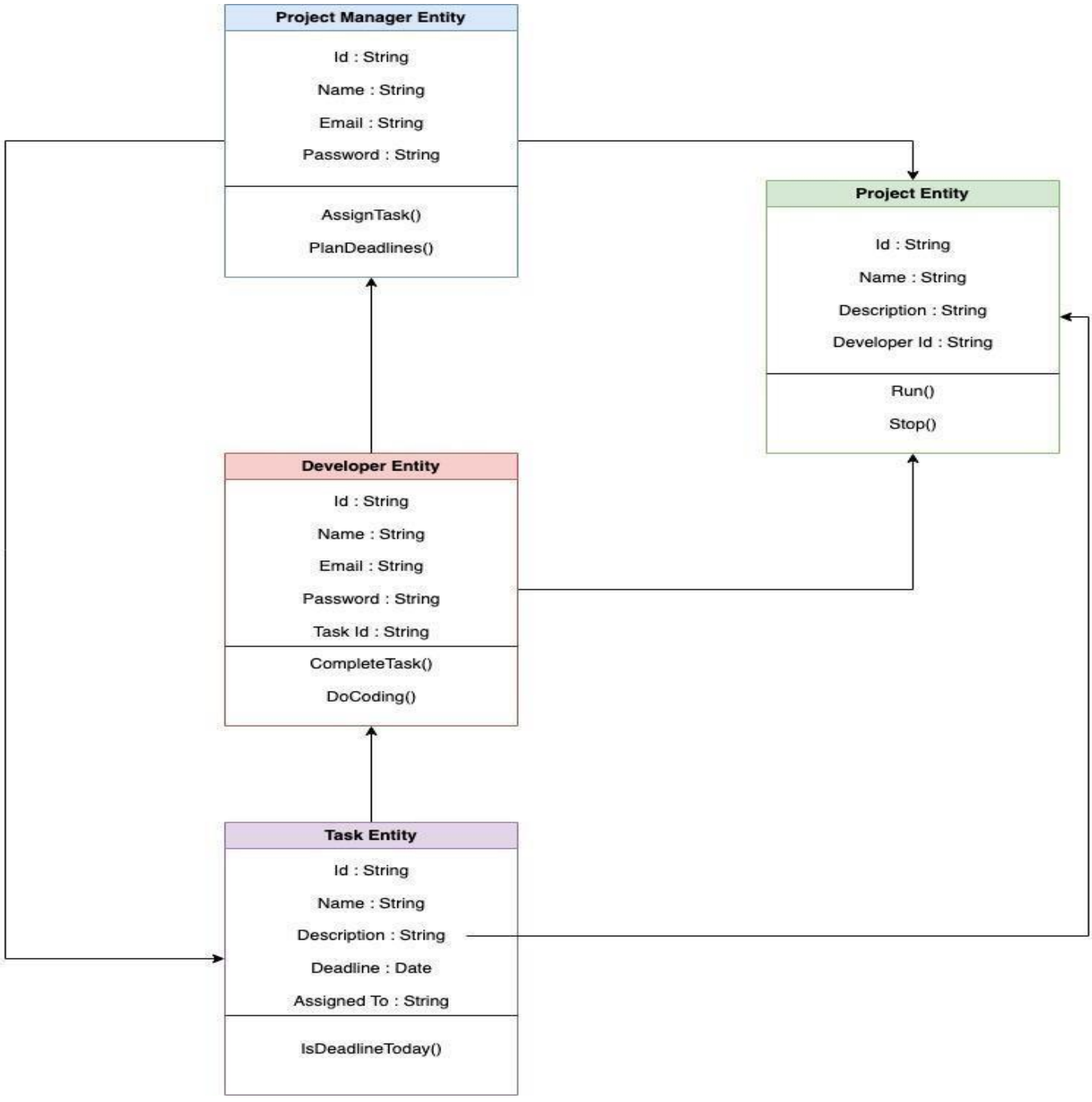


Figure 4.1.3 Class Diagram

4.2 DATA FLOW DIAGRAM:

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects.It can be used for the visualization of data processing (structured design).Data flow diagrams are also known as bubble charts. DFD is a designing tool used in the top down approach to Systems Design. DFD levels are numbered 0, 1 or 2, and occasionally go to even Level 3 or beyond. DFD Level 0 is also called a Context Diagram.

Level 0:

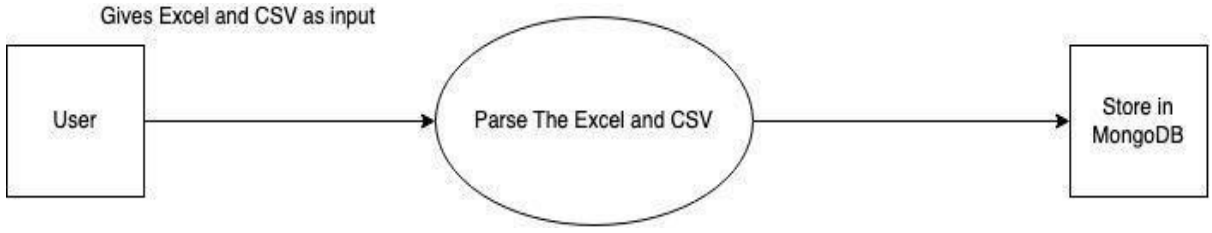


Figure 4.2.1 Level 0 DFD Diagram

LEVEL 1:

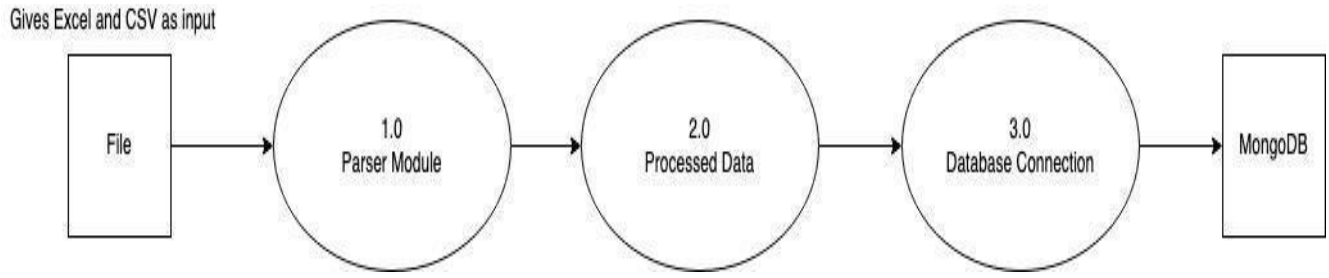


Figure 4.2.1 Level 1 DFD Diagram

LEVEL 2:

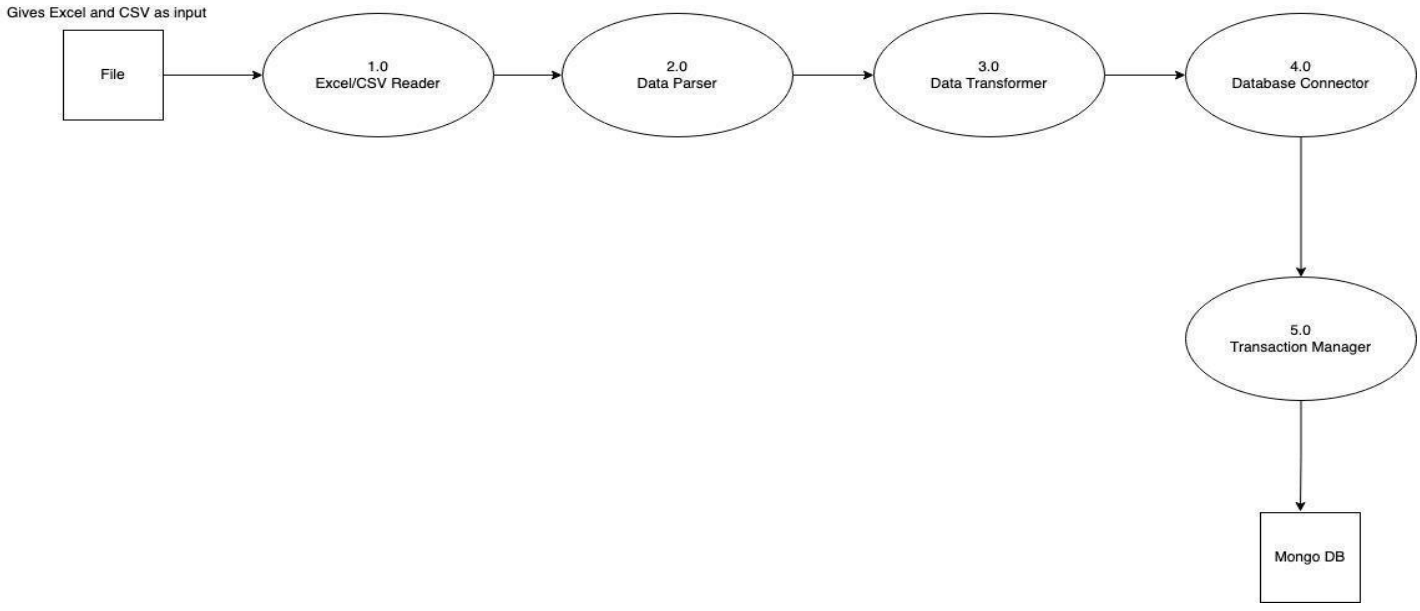


Figure 4.2.2 Level 2 DFD Diagram

4.3 SYSTEM ARCHITECTURE

4.3.1 SYSTEM ARCHITECTURE OVERVIEW

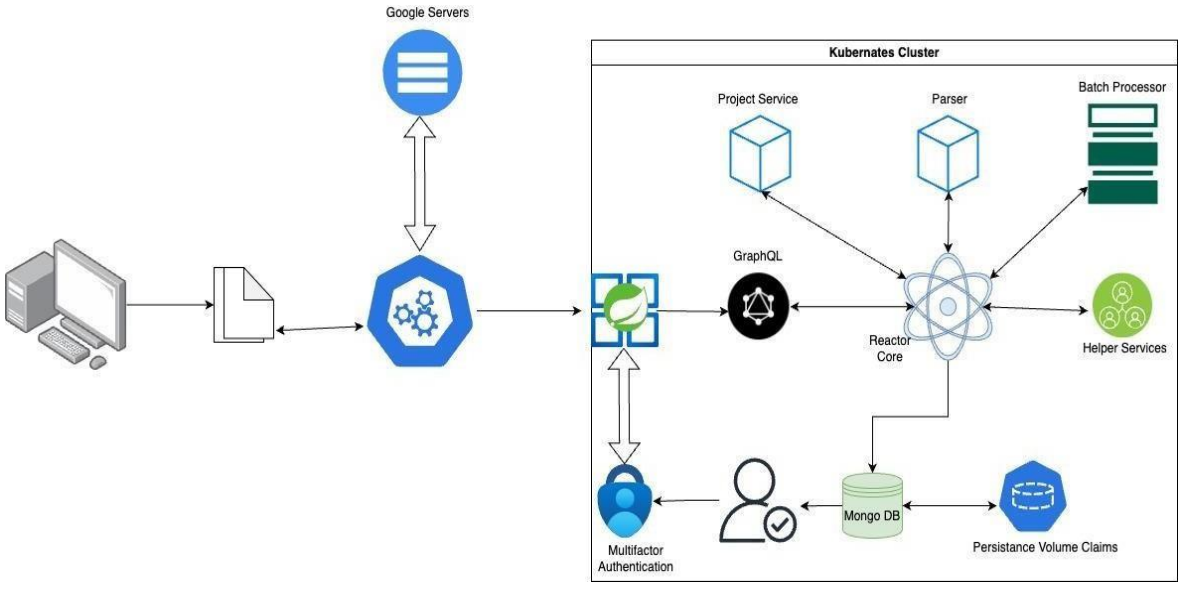


Figure 4.3.1 System Architecture

The architecture diagram shows the processes involved for building the project.This System architecture shows all the possible functionality of the project through the different frameworks and database. Some of the tools ,Frameworks, Database, Packages used in the project are listed below.

- Spring Boot
- React JS
- MongoDb
- Node
- Graphql

CHAPTER-5

SYSTEM IMPLEMENTATION

CHAPTER-5

SYSTEM IMPLEMENTATION

5.1 ALGORITHMS:

1. Batch Processing

2. Synchronous Multithreading

3. Non-Blocking /Parallel Processing4.Multi-Factor Authentication.

5.1.1 BATCH PROCESSING

Three key components need to be configured in order to process Excel/CSV files in batches using Batch: the ItemReader, ItemProcessor, and ItemWriter. Using Spring Batch's FlatFileItemReader and providing the file path and column mappings, the ItemReader reads data from the files. Every item that is read is subjected to business logic or data transformations by the ItemProcessor, and after Spring Batch's JdbcBatchItemWriter is set up with SQL statements for insertion, the ItemWriter uses it to send the processed data to the database. These parts are combined into a batch task that is defined in a single step and includes a chunk size specification for processing effectiveness. Testing guarantees that the Spring Boot application runs successfully and offers a scalable approach to managing big datasets with parallel processing and efficient resource use

5.1.2 SYNCHRONOUS MULTITHREADING:

By enabling concurrent data processing inside each batch and utilizing the processing capabilities of multi-core machines, multithreading in the context of batch improves performance. Although multithreading is not supported by Batch by default, it may be enabled by setting the Step to run in parallel. This is accomplished by configuring a Task Executor, such a ThreadPoolTaskExecutor, to oversee several threads for the simultaneous execution of batch processing jobs. Parallel processing is made possible by each thread, which completes a step

of data at a time and maintains thread safety via suitable synchronization techniques. Multithreading in Spring Batch takes careful consideration of parameters like as thread pool size, resource consumption, and potential concurrency concerns, but when done successfully, it may dramatically enhance throughput and minimize processing time.

5.1.3 NON BLOCKING / PARALLEL PROCESSING:

WebFlux and Project Reactor enable reactive programming paradigms that may be used to process jobs in parallel without blocking. Reactive libraries, like Flux and Mono, let you to handle I/O-bound operations like network requests and database transactions by enabling you to execute tasks simultaneously without stopping threads. You may provide endpoints in WebFlux that return reactive types, such as Flux or Mono, which enables the application to process requests in parallel. You may use operators like flatMap, merge, or zip to process jobs in parallel and integrate their results asynchronously by having them run simultaneously. Schedulers may also be used to manage thread pools and the execution context.

5.1.4 MULTIFACTOR AUTHENTICATION :

Use Google Authenticator to implement two-factor authentication (2FA). The system asks users for their password and username as the initial authentication factor. When validation is completed, the system creates and shows a QR code with a special secret key on it. Using the Google Authenticator app on their smartphone, the user scans this QR code. The software then uses the shared secret key to construct a time-based one-time password (TOTP). As the second factor of authentication, the user inputs this TOTP. The shared secret key is used to produce an expected value, which the system uses to validate the entered TOTP. The user is given access if the TOTP matches; if not, access is refused.

5.2 MODULE DESIGN SPECIFICATION

1. Data Parsing
2. Data Structuring
3. Collaboration Featurues
4. Data Analysis and Manipulation

5.3 MODULE DESCRIPTION

5.3.2 DATA PARSING:

- Ability to parse data from common file formats such as CSV, JSON, XML, Excel, etc.
- Support for parsing data from external sources like databases, APIs, and web scraping.
- Data normalization and validation to ensure consistency and integrity.

5.3.3 DATA STRUCTURING:

- Transforming parsed data into a standardized format for further processing.
- Handling nested and complex data structures appropriately.
- Support for custom mappings and transformations based on user requirements.

5.3.4 COLLABORATION FEATURES:

- User authentication and authorization mechanisms to control access to data and collaboration features.
- Real-time collaboration capabilities for multiple users to work on the same dataset simultaneously.
- Version control functionality to track changes, revert to previous versions, and resolve conflicts.

5.3.5 DATA ANALYSIS AND MANUPULATION:

- Basic data analysis functionalities such as filtering, sorting, aggregation, and visualization.
- Support for advanced data manipulation operations including joins, transformations, and calculations.
- Integration with third-party libraries and tools for specialized data analysis tasks.

CHAPTER-6

RESULTS AND DISCUSSIONS

6. TESTING

To keep the system error free during the phases of development and during the time when new features are added, the following testing strategies are applied:

6.1 Unit Testing

Unit Testing is done on individual modules as they are completed and become executable. It is confined only to the designer's requirements.

TEST CASE ID	SCENARIO	TEST CASE	PRECONDITION	EXPECTED RESULT	ACTUAL RESULT	STATUS
1	Store data to block chain	Connect application to blockchain wallet	Valid project title and description is filled	Success message project reserved for copyright	Project stored in blockchain with success logs	PASS
2	Authenticate user with app	Enter registered email and password and OTP at time of login	Multifactor authentication should be enabled or should be registered in authenticator app	On Successful authentication user redirects to home page	User logged in successfully and redirect to home page	PASS
3	Show errors in upload files if any primary key mismatch	Navigate to upload page and upload files	Files should have invalid developer id's that does not quardinate to project id	Should navigate to validation page and show errors to user	Errors shown in validation screen and not allowed to move forward	PASS

CHAPTER-7
CONCLUSION

7.1 CONCLUSION

In conclusion, the development and implementation of a comprehensive data integration and query system have significantly enhanced data management and scientific analysis capabilities. Through this project, we have successfully addressed the challenges associated with disparate data sources and complex data structures. By integrating various data sources into a unified platform, we have streamlined the process of accessing and analyzing data, thereby improving efficiency and productivity in scientific research and analysis.The system's robust query capabilities have empowered researchers and analysts to retrieve relevant information quickly and efficiently, facilitating more informed decision-making processes. Additionally, the integration of advanced data management techniques has enhanced data integrity, security, and accessibility, ensuring that data remains reliable and protected throughout its lifecycle.

7.2 FUTURE ENHANCEMENT

The future enhancements for the comprehensive data integration and query system encompass various aspects to enhance functionality and utility. Integration of advanced machine learning algorithms enables predictive analytics, while real-time data processing supports instantaneous analysis, especially in time-sensitive sectors like healthcare and finance. Strengthening security features ensures compliance and protects sensitive data from unauthorized access. Scalability and performance optimization through distributed computing enable efficient handling of increasing data volumes and user demands. Broadening integration to include diverse external data sources enriches analysis breadth. Intuitive data visualization tools facilitate seamless exploration and communication of insights. Leveraging semantic technologies enhances interoperability and knowledge discovery. Collaboration features streamline teamwork with version control and task tracking capabilities. Cloud

integration optimizes resource utilization while maintaining data control. Gathering user feedback and monitoring system performance drive continuous improvement, ensuring adaptability to changing needs. Together, these enhancements make the system more adaptive, secure, and user-friendly, fostering better data-driven decision-making and advancing scientific analysis across domains.

APPENDICES

A.1 SDG GOALS

SDG 3

Good Health and Well-being: Data integration and analysis can play a crucial role in public health by facilitating disease surveillance, monitoring health trends, and identifying areas for intervention, thus contributing to improving healthcare outcomes and overall well-being.

SDG 4

Quality Education: Enhanced data management and scientific analysis can improve educational systems by providing educators and policymakers with valuable insights into learning outcomes, educational needs, and effective teaching strategies.

SDG 9

Industry, Innovation, and Infrastructure: Developing comprehensive data integration and query systems requires innovation in technology and infrastructure. Achieving this goal can contribute to building resilient infrastructure, promoting inclusive and sustainable industrialization, and fostering innovation.

SDG 11

Sustainable Cities and Communities: Comprehensive data systems can support urban planning, infrastructure development, and resource management in cities, promoting sustainability, resilience, and inclusive growth.

SDG 13

Climate Action: Effective data management and analysis are essential for understanding climate change patterns, assessing environmental impacts, and formulating strategies for mitigation and adaptation.

SDG 17

Partnerships for the Goals: Collaboration between governments, private sector entities, research institutions, and civil society organizations is crucial for developing and implementing comprehensive data integration and query systems. Such partnerships can enhance data sharing, improve analytical capabilities, and accelerate progress across all SDGs.

A.2 SAMPLE SCREENSHOTS

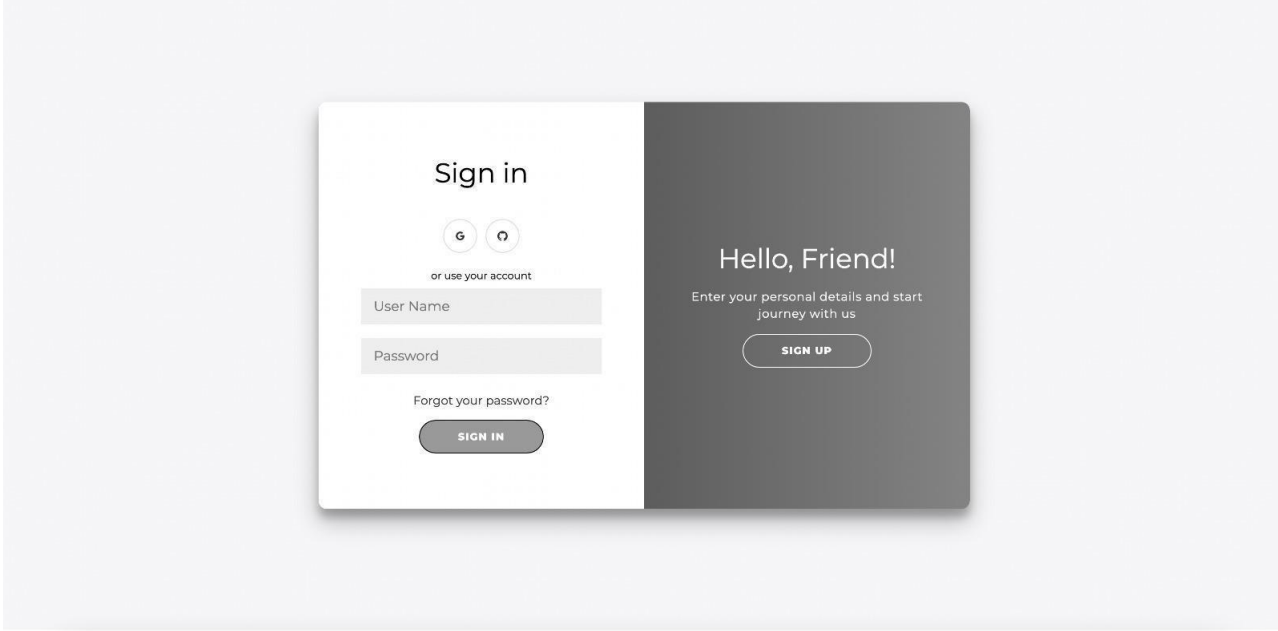


Figure A.2.1 Screenshot of Signup Page

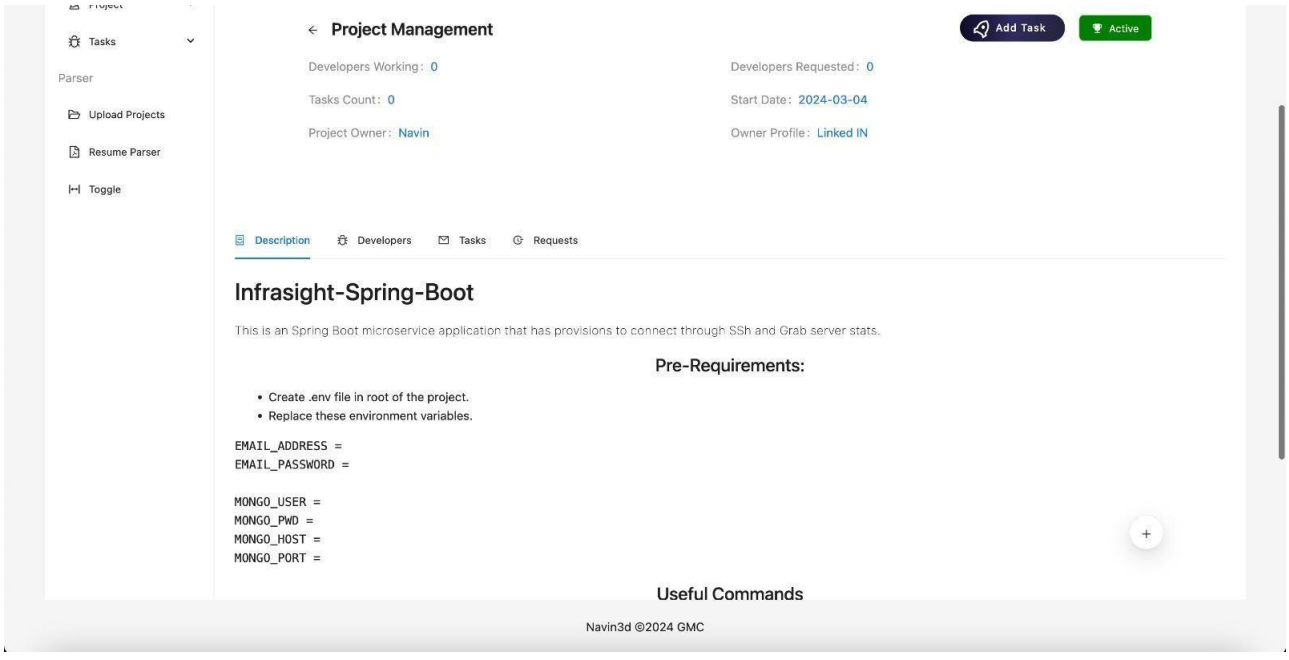


Figure A.2.2 Screenshot of Project Page

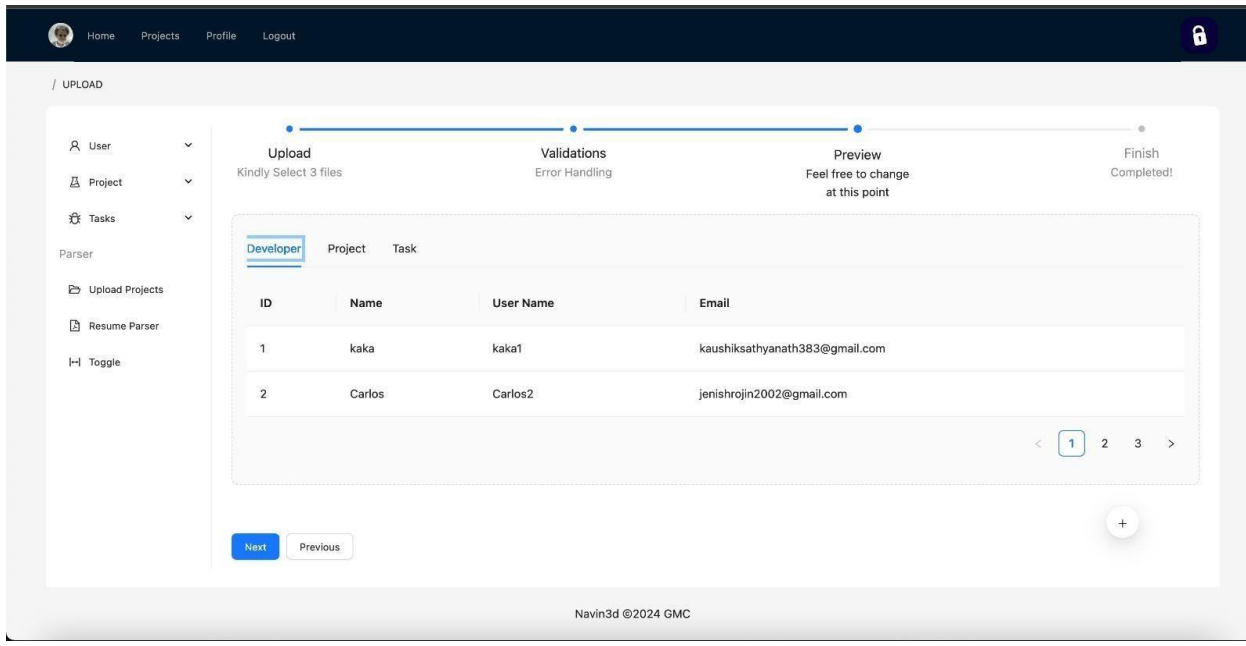


Figure A.2.3 Screenshot of upload page

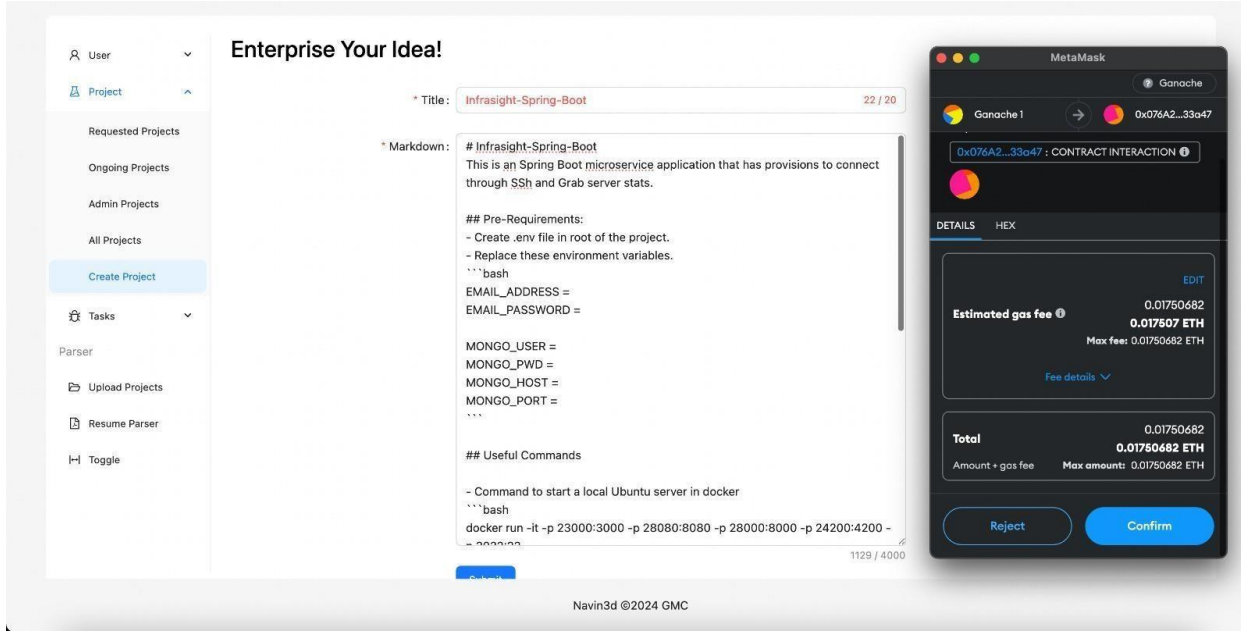


Figure A.2.4 Screenshot of Blockchain part

A.3 SOURCE CODE:

PROJECTFORM.JSX:

```
import { useDispatch } from 'react-redux';
import { useWriteContract } from 'wagmi';
import { Button, Form, Input, notification } from 'antd';
import { editProject } from '../../redux/project-slice';
import { createProject } from '../../redux/auth-slice';
import { createOrUpdateProject } from '../../services/project-service';
import { projectToBlockchain } from '../../services/web3-service';
import { useWeb3ModalAccount } from '@web3modal/ethers5/react';


const ProjectForm = ({ projectData }) => {
const layout = {
labelCol: {
span: 8,
},
wrapperCol: {
span: 16,
},
};
const validateMessages = {
required: '${label} is required!',
types: {
email: '${label} is not a valid email!',
number: '${label} is not a valid number!',
},
number: {
range: '${label} must be between ${min} and ${max}',
},
};
const { isConnected, address } = useWeb3ModalAccount();
const { writeContractAsync } = useWriteContract();
const dispatch = useDispatch();
const [notice, contextHolder] = notification.useNotification();
```

```
const writeDataToblockchain = data => {
if (isConnected) {
projectToBlockchain({ ...data, ownerAddress: address }, writeContractAsync);
notice.success({
message: Project Copyrights reserved.,
});
}
};

const saveProject = project => createOrUpdateProject(project)
.then(res => {
notice.success({
message: Project ${project.tittle} saved.,
});
writeDataToblockchain(res.data);
})
.catch(e => {
notice.error({
message: e.message,
});
console.log(e)
});

const onFinish = (values) => {
const project = { ...values["project"], createdBy: projectData.createdBy };
console.log("project ", project);
if (projectData["id"])
dispatch(editProject(project));
else
dispatch(createProject(project));
saveProject(project);
};
return (
<div>
{contextHolder}
<Form
{...layout}
name="nest-messages"
onFinish={onFinish}
```

```
style={{
maxWidth: '70%',
}}
validateMessages={validateMessages}
>
<Form.Item
name={['project', 'tittle']}
label="Title"
initialValue={projectData.tittle}
rules={[
{
required: true,
},
]}
>
<Input
count={{
show: true,
max: 20,
}}
/>
</Form.Item>
<Form.Item
name={['project', 'description']}
label="Markdown"
initialValue={projectData.description}
rules={[
{
required: true,
},
]}>
<Input.TextArea
style={{
height: 500,
}}
count={{
show: true,
max: 4000,
}}
```

```
/>
</Form.Item>
<Form.Item
wrapperCol={{
...layout.wrapperCol,
offset: 8,
}}
>
<Button type="primary" htmlType="submit">
Submit
</Button>
</Form.Item>
</Form>
</div>
);
}

export default ProjectForm
```

RESUMEPAGE.JSX:

```
import { useState, useEffect } from 'react';
import { useDispatch } from 'react-redux';
import { Tabs, Select, Row, Avatar, List, Button, notification } from "antd";
import UploadBox from "../components/parser/UploadBox";
import SideNavLayout from "../layouts/SideNavLayout";
import { clearAllresumes, filterUsers } from '../services/resume-service';
import { setProfile } from '../redux/profile-slice';

const FilterTabContent = _ => {

const options = [];
const dispatch = useDispatch();
const [data, setData] = useState([]);
const [filterKey, setFilterKey] = useState("");
const [filterValue, setFilterValue] = useState([""]);

for (let i = 10; i < 36; i++) {
options.push({
value: i.toString(36) + i,
label: i.toString(36) + i,
});
}

const handleChangeKey = (e) => {
setFilterKey(e.value);
};

const handleChangeValue = (value) => {
setFilterValue(value);
};

useEffect(_ => {
filterUsers(filterKey, filterValue).then(res => {
setData(res.data);
})
.catch(e => console.log(e));
}, [filterKey, filterValue]);

return (
<div>
<Row
align="left"
```

```
style={{
marginBottom: "3%"
}}
>
<Select
labelInValue
defaultValue={{
value: 'skills',
label: 'Skills',
}}
style={{
width: 140,
}}
onChange={handleChangeKey}
options={[
{
value: 'skills',
label: 'Skills',
},
{
value: 'email',
label: 'Email',
},
{
value: 'firstName',
label: 'First Name',
},
{
value: 'lastName',
label: 'Last Name',
},
{
value: 'place',
label: 'Location',
},
{
value: 'links',
label: 'Social Profiles',
},
{
value: 'education',
label: 'Qualifications',
},
```

```
]}
/>
<Select
mode="tags"
style={{
width: "50%",
marginLeft: "3%"
}}
onChange={handleChangeValue}
tokenSeparators={[',', "  "]}
options={options}
/>
</Row>
<List
pagination={{
position: "top",
align: "end",
}}
dataSource={data}
renderItem={(item, index) => (
<List.Item
actions={[<Button onClick={() => dispatch(setProfile(item))}>Profile</Button>]}
>
<List.Item.Meta
avatar={<Avatar
src={https://api.dicebear.com/7.x/miniavs/svg?seed=${index}} />}
title={<h4 key={item.id} >{item.firstName} {item.lastName}</h4>}
description={item.email}
/>
</List.Item>
)}
/>
</div>
);
}

const UploadTabContent = _ => {
const [notice, contextHolder] = notification.useNotification();
const handleClearResumes = _ => {
clearAllresumes()
.then(res => {
notice.success({
```

```
message: "Resumes cleared from DB..."
});
})
.catch(e => {
console.log(e);
notice.warning({
message: "Problem Clearing resumes...",
description: e.message,
});
});
}
return (
<div>
{contextHolder}
<UploadBox actionURL="http://localhost:3001/pdf/single" />
<Button onClick={handleClearResumes}>Clear All Resumes</Button>
</div>
);
}const PageContents = () => {
const items = [
{
key: '1',
label: 'Parse',
children: <UploadTabContent />,
},
{
key: '2',
label: 'Filter',
children: <FilterTabContent />
},
];
return (
<div>
<Tabs defaultActiveKey="1" items={items} style={{ margin: "1.5%" }} />
</div>
);
}

const ResumePage = _ => {
return <SideNavLayout element={<PageContents />} />
}

export default ResumePage;
```

SERVER.PY:

```
from fastapi import FastAPI, File, UploadFile
import uvicorn, re, pathlib, os, shutil
import pandas as pd
from fastapi.middleware.cors import CORSMiddleware
from fastapi.encoders import jsonable_encoder
from datetime import datetime


base = pd.to_datetime("2022-10-10")
app = FastAPI()
app.add_middleware(
CORSMiddleware,
allow_origins=['*'],
allow_credentials=True,
allow_methods=['*'],
allow_headers=['*']
)

FILE_BASEPATH: str = os.path.join(pathlib.Path().resolve(), "files/prod/")


@app.get("/")
def read_root() -> dict:
return {"status": "App running in port 8000"}


@app.post("/uploadfile/{bactchId}/")
async def upload_file(bactchId: str, files: UploadFile = File(...)):
batch_path = FILE_BASEPATH + bactchId
if(not os.path.isdir(batch_path)):
os.mkdir(batch_path)
path = os.path.join(batch_path, files.filename)
try:
contents = await files.read()
with open(path, 'wb') as f:
f.write(contents)
except Exception as e:
print(e)
return False
finally:
files.close()
return True
```

```
@app.get("/clearbatch/{bactchId}/")
async def upload_file(bactchId: str):
batch_path = FILE_BASEPATH + bactchId
if os.path.isdir(batch_path):
shutil.rmtree(batch_path)
return True

def return_list_ifempty(df):
try:
if df.empty:
return []
return df.to_dict(orient="records")
except Exception:
return []

@app.get("/process-batch/{bactchId}/")
async def proces_batch(bactchId: str) -> dict:
batch_path = FILE_BASEPATH + bactchId

parsed: list = []
developer_file = filter_file(batch_path, "developer")
project_file = filter_file(batch_path, "project")
task_file = filter_file(batch_path, "task")

project_df = []
developer_df = []
task_df = []
if ".xlsx" in project_file:
project_df = await excel_to_df(project_file)
else:
project_df = await csv_to_df(project_file)

if ".xlsx" in developer_file:
developer_df = await excel_to_df(developer_file)
else:
developer_df = await csv_to_df(developer_file)

if ".xlsx" in task_file:
task_df = await excel_to_df(task_file)
else:
task_df = await csv_to_df(task_file)

parsed.append(handle_developer(developer_df))
parsed.append(handle_project(project_df, developer_df))
```

```
parsed.append(handle_task(task_df, project_df, developer_df))

returnValue = {
"isValid": True,
"errors": parsed
}

if parsed[0]['isDeveloperValid'] == False or parsed[1]['isProjectValid'] == False
or parsed[2]['isTaskValid'] == False:
returnValue["isValid"] = False

if returnValue["isValid"] == True:
returnValue["data"] = {
"developer": return_list_ifempty(developer_df),
"project": return_list_ifempty(project_df),
"task": return_list_ifempty(task_df),
}
returnValue.pop("errors")

print(returnValue)
return jsonable_encoder(returnValue)


def filter_file(folder_path, file_name):
for file in os.listdir(folder_path):
print(file)
if file_name in file:
return os.path.join(folder_path, file)
return None

async def csv_to_df(csv_file):
return pd.read_csv(csv_file)

async def excel_to_df(excel_file):
return pd.read_excel(excel_file)

async def handle_csv(csv_file):
file_name = csv_file.filename
dataframe = await csv_to_df(csv_file)
if "developer" in file_name:
return handle_developer(dataframe)
elif "project" in file_name:
return handle_project(dataframe)
elif "task" in file_name:
```

```
return handle_task(dataframe)


async def handle_excel(exce_file):
file_name = exce_file.filename
dataframe = await excel_to_df(exce_file)
if "developer" in file_name:
return handle_developer(dataframe)
elif "project" in file_name:
return handle_project(dataframe)
elif "task" in file_name:
return handle_task(dataframe)


def handle_developer(developer_dataframe) -> dict:
is_valid = True
developer_dataframe_cleaned = developer_dataframe.where(pd.notnull(developer_dataframe), None)
# Get rows with null values
null_rows = developer_dataframe_cleaned[developer_dataframe_cleaned.isnull().any(axis=1)]
# Exclude the first row
null_rows = null_rows[null_rows.index != 1]
dict_converted = developer_dataframe.where(pd.notnull(developer_dataframe), "null").to_dict(orient="records")

invalid_emails = validate_email(dict_converted)
duplicate_emails = developer_dataframe[developer_dataframe['email'].duplicated(keep=False)]
duplicate_user_ids = developer_dataframe[developer_dataframe['id'].duplicated(keep=False)]
if len(null_rows) > 0 or len(invalid_emails) > 0 or len(duplicate_emails) > 0 or len(duplicate_user_ids) > 0:
is_valid = False
return {
"isDeveloperValid": is_valid,
"invalidEmails": invalid_emails,
"nullRows": return_list_ifempty(null_rows),
"duplicateEmailEntry": return_list_ifempty(duplicate_emails),
"duplicateIdEntry": return_list_ifempty(duplicate_user_ids)
}
```

```
def handle_project(project_dataframe, developer_dataframe):
is_valid = True
null_rows = project_dataframe[project_dataframe.isnull().any(axis=1)]
null_rows = null_rows[null_rows.index != 1]
missing_developer_ids =
project_dataframe[~project_dataframe['createdBy'].isin(developer_dataframe['i
d'])]
duplicate_project_ids =
project_dataframe[project_dataframe['id'].duplicated(keep=False)]
if len(null_rows) > 0 or len(duplicate_project_ids) > 0:
is_valid = False
return {
"isProjectValid": is_valid,
"nullRows": return_list_ifempty(null_rows),
"duplicateIdEntry": return_list_ifempty(duplicate_project_ids),
"missingDevelopers":   return_list_ifempty(missing_developer_ids),
}

def handle_task(task_dataframe, project_dataframe, developer_dataframe):
is_valid = True
today_date = datetime.now().date()
null_rows = project_dataframe[project_dataframe.isnull().any(axis=1)]
null_rows = null_rows[null_rows.index != 1]
missing_project_ids =
task_dataframe[~task_dataframe['projectId'].isin(project_dataframe['id'])]
missing_assigned_to =
task_dataframe[~task_dataframe['assignedTo'].isin(developer_dataframe['id'])]
#                 invalid_deadlines =
task_dataframe[~pd.to_datetime(task_dataframe['deadline'],
errors='coerce').dt.date.isnull() & (pd.to_datetime(task_dataframe['deadline']) <
today_date)]
invalid_deadlines = []
invalid_assigned_to = []

for index, row in task_dataframe.iterrows():
project_id = row['projectId']
assigned_to = row['assignedTo']
# Check if project_id exists in projects DataFrame
if project_id in project_dataframe['id'].values:
# Filter projects DataFrame for the specific project_id
project_row = project_dataframe[project_dataframe['id'] == project_id]
if not project_row.empty:
# Extract the developers associated with the project
project_developers = project_row['developers'].iloc[0]
```

```
# Check if project_developers is not NaN (missing)
if not pd.isna(project_developers):
# Convert to list if not already
if not isinstance(project_developers, list):
project_developers = [project_developers]
# Check if assigned_to is not in the list of project developers
if assigned_to not in project_developers:
invalid_assigned_to.append({ "projectId": project_id, "assignedId": assigned_to })

try:
task_dataframe['deadline'] = pd.to_datetime(task_dataframe['deadline'])
invalid_deadlines = task_dataframe[task_dataframe['deadline'] < pd.Timestamp.today()]
except ValueError as e:
print("Invalid deadline format:", e)
if len(null_rows) > 0 or len(missing_project_ids) > 0 or len(missing_assigned_to) > 0 or len(invalid_deadlines) > 0:
is_valid = False
return {
"isTaskValid": is_valid,
"nullRows": return_list_ifempty(null_rows),
"missingProjects": return_list_ifempty(missing_project_ids),
"missingDevelopers": return_list_ifempty(missing_assigned_to),
"invalidDeadlines": return_list_ifempty(invalid_deadlines),
"inValidAssignment": return_list_ifempty(invalid_assigned_to)
}

def validate_email(objects):
invalid_emails = []
for obj in objects:
for key, value in obj.items():
if key == 'email':
if value != value:
invalid_emails.append(value)
else:
# print(value)
if not re.match(r'^[a-zA-Z0-9._%+-]+@[a-zA-Z0-9.-]+\.[a-zA-Z]{2,}$', value):
invalid_emails.append({ "id": obj["id"], "email": value })
return invalid_emails

if _name_ == "_main_":
uvicorn.run("server:app", host="0.0.0.0", port=8000, log_level="info", reload=True
```

SECURITY.CONFIGJAVA:

```
package gmc.learning.reactive.management.project.security;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Value;
import org.springframework.context.annotation.Bean;
import org.springframework.context.annotation.Configuration;
import org.springframework.http.HttpMethod;
import
org.springframework.security.authentication.ReactiveAuthenticationManager;
import
org.springframework.security.authentication.UserDetailsRepositoryReactiveA
uthenticationManager;
import
org.springframework.security.config.web.server.SecurityWebFiltersOrder;
import org.springframework.security.config.web.server.ServerHttpSecurity;
import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.security.web.server.SecurityWebFilterChain;
import
org.springframework.security.web.server.context.NoOpServerSecurityContext
Repository;
import org.springframework.stereotype.Component;
import  org.springframework.web.cors.CorsConfiguration;
import org.springframework.web.cors.reactive.CorsConfigurationSource;
import
org.springframework.web.cors.reactive.UrlBasedCorsConfigurationSource;

@Component
@Configuration
public class SecurityConfig {

@Value("${settings.app.disableSecurity}")
private Boolean disableSecurity;

@Autowired
private AuthConfig authConfig;

@Bean
SecurityWebFilterChain springWebFilterChain(ServerHttpSecurity http,
JwtTokenProvider tokenProvider,
AuthService authService, ReactiveAuthenticationManager
reactiveAuthenticationManager) {
```

```
if (disableSecurity)
return http.cors(cors -> cors.configurationSource(corsConfigurationSource())).csrf(ServerHttpSecurity.CsrfSpec::disable).httpBasic(ServerHttpSecurity.HttpBasicSpec::disable)
.authenticationManager(reactiveAuthenticationManager)
.securityContextRepository(NoOpServerSecurityContextRepository.getInstance())
.authorizeExchange(it -> it.pathMatchers("", "/*").permitAll()).build();
final String SWAGGER = "/webjars/swagger-ui/**";
return http.cors(cors -> cors.configurationSource(corsConfigurationSource())).csrf(ServerHttpSecurity.CsrfSpec::disable).httpBasic(ServerHttpSecurity.HttpBasicSpec::disable)
.authenticationManager(reactiveAuthenticationManager)
.securityContextRepository(NoOpServerSecurityContextRepository.getInstance())
.authorizeExchange(it -> it.pathMatchers(HttpMethod.GET, SWAGGER).permitAll()
.pathMatchers(HttpMethod.GET, "/auth/*").permitAll().pathMatchers(HttpMethod.POST, "/auth")
.permitAll().pathMatchers(HttpMethod.POST, authConfig.getAuthUrl()).permitAll()
.pathMatchers(authConfig.getOauthPath()).permitAll().pathMatchers("/**").authenticated()
.anyExchange().permitAll())
.oauth2Login(oauth -> {
oauth.authenticationSuccessHandler(new OAuthSuccessHandler(tokenProvider, authService));
}).addFilterAt(new JwtTokenAuthenticationFilter(tokenProvider), SecurityWebFiltersOrder.HTTP_BASIC)
.build();
}

@Bean
BCryptPasswordEncoder bCryptPasswordEncoder() {
return new BCryptPasswordEncoder();
}

@Bean
ReactiveAuthenticationManager reactiveAuthenticationManager(BCryptPasswordEncoder bCryptPasswordEncoder,
AuthService authService) {
var authenticationManager = new UserDetailsRepositoryReactiveAuthenticationManager(authService);
authenticationManager.setPasswordEncoder(bCryptPasswordEncoder);
return authenticationManager;
```

```
}

private CorsConfigurationSource corsConfigurationSource() {
CorsConfiguration configuration = new CorsConfiguration();
configuration.setAllowedOrigins(List.of("http://localhost:3000"));
configuration.setAllowedMethods(List.of("GET", "POST", "PUT"));
configuration.setAllowedHeaders(List.of("Access-Control-Allow-Origin",
"Authorization", "Content-Type"));
configuration.setAllowCredentials(true);
UrlBasedCorsConfigurationSource source = new
UrlBasedCorsConfigurationSource();
source.registerCorsConfiguration("/**", configuration);
return source;
}

}
```

GOOGLEAUTHENTICATIONHANDLER.JAVA:

```
package gmc.learning.reactive.management.project.security;

import java.net.URI;
import java.util.ArrayList;
import java.util.Collection;
import  java.util.concurrent.CompletableFuture;
import  java.util.concurrent.ExecutionException;
import java.util.function.Supplier;

import org.springframework.http.server.reactive.ServerHttpRequest;
import org.springframework.security.core.Authentication;
import org.springframework.security.core.GrantedAuthority;
import
org.springframework.security.web.server.DefaultServerRedirectStrategy;
import      org.springframework.security.web.server.ServerRedirectStrategy;
import org.springframework.security.web.server.WebFilterExchange;
import
org.springframework.security.web.server.authentication.ServerAuthenticationS
uccessHandler;
import org.springframework.web.server.ServerWebExchange;

import  gmc.learning.reactive.management.project.models.DeveloperModel;
import reactor.core.publisher.Mono;

public class OAuthSuccessHandler implements
ServerAuthenticationSuccessHandler {

private JwtTokenProvider jwtTokenProvider;

private AuthService authService;

private ServerRedirectStrategy serverRedirectStrategy;

public OAuthSuccessHandler( JwtTokenProvider jwtTokenProvider,
AuthService authService) {
this.jwtTokenProvider = jwtTokenProvider;
this.authService = authService;
```

```
this.serverRedirectStrategy = new DefaultServerRedirectStrategy();
}

@Override
public Mono<Void> onAuthenticationSuccess(WebFilterExchange
webFilterExchange, Authentication authentication) {
DeveloperModel signinUser = new
DeveloperModel(authentication.getPrincipal().toString());
CompletableFuture<DeveloperModel> saved =
authService.registerUser(signinUser).toFuture();
ServerWebExchange exchange = webFilterExchange.getExchange();
Authentication newAuthentication = new Authentication() {
private static final long serialVersionUID = -39092530541194180 51L;
@Override
public String getName() {
try {
return saved.get().getId();
} catch (InterruptedException e) {
// TODO Auto-generated catch block
e.printStackTrace();
return "";
} catch (ExecutionException e) {
// TODO Auto-generated catch block
e.printStackTrace();
return "";
}
}
@Override
public void setAuthenticated(boolean isAuthenticated) throws
IllegalArgumentException {
}
@Override
public boolean isAuthenticated() {
// TODO Auto-generated method stub
return true;
}
@Override
public Object getPrincipal() {
return null;
```

```
}
@Override
public Object getDetails() {
return null;
}
@Override
public Object getCredentials() {
return null;
}
@Override
public Collection<? extends GrantedAuthority> getAuthorities() {
return new ArrayList<>();
}
};
String token = jwtTokenProvider.createToken(newAuthentication);;
Supplier<Mono<Void>> responseSuplier = () -> serverRedirectStrategy
.sendRedirect(exchange,
resolveRedirectUri(exchange.getRequest(), token)
);
return responseSuplier.get();
}

private URI resolveRedirectUri(ServerHttpRequest httpRequest, String token)
{
//                                          String encodedUrlSafeState =
httpRequest.getQueryParams().getFirst("state");
//      if (!StringUtils.hasText(encodedUrlSafeState))
//          return URI.create(httpRequest.getURI().getHost());
//                                                  byte[] redirectUriByte =
Base64.getDecoder().decode(encodedUrlSafeState);
```

Enhancing Data Security and Patient Care with Blockchain in Healthcare

Mohana Prakash TA
Dept of computer science
Panimalar engineering college
Chennai , India
tamohanaprakash@gmail.com

Navin Durai SM
Dept of computer science
Panimalar engineering college
Chennai , India
smnavin65@gmail.com

Logasubramani SM
Dept of computer science
Panimalar engineering college
Chennai , India
smlogasubramani@gmail.com

Mohana Krishnan S
Dept.of computer science
Panimalar engineering college
Chennai , India
crmonish2103@gmail.com

Abstract: ***Data security and safety are top priorities in next-generation computing, especially in areas like financial transactions and medicine. Since medical reports include sensitive patient data, protecting them from data breaches is critical to the healthcare industry. Our platform, which makes use of blockchain technology, transforms the sharing and storing of medical reports by guaranteeing safe, decentralized storage that is only accessible by those with permission. In order to reduce fatalities, blockchain also makes medical transactions easier by giving patients' treatments priority based on severity. Enhancing security and accessibility, biometric authentication—such as fingerprint and retinal scanning—allows emergency access to medical records. Furthermore, our platform offers instructional materials to enable users to comprehend illnesses and encourage well-being.***

Keywords: Blockchain technology, decentralized storage, next-generation computing, safety, security, data, medication, financial transactions, healthcare, and medical reports, authorized parties, medical transactions, severity, fatalities, biometric authentication, fingerprint scanning, retinal scanning, emergency access, educational resources, diseases, wellness.

I. INTRODUCTION

It is more important than ever to protect patient data and optimize the delivery of care in the constantly changing healthcare environment. Next-generation computing presents a rare chance to take advantage of state-of-the-art tools to successfully tackle these problems. Ensuring data integrity and confidentiality is essential to this endeavor, especially when it comes to medical reports that contain sensitive information that is essential for diagnosis and treatment.

Given this, blockchain technology becomes a revolutionary force that provides unmatched decentralization and security. Blockchain promises to revolutionize data storage and transaction methods, thereby revolutionizing healthcare systems, ensuring the integrity and confidentiality of patient information while facilitating seamless access for authorized parties.

This abstract examines how blockchain technology is being creatively applied in the healthcare industry, with an emphasis on how it can improve patient care and strengthen data security. By utilizing a customized platform that is outfitted with sophisticated biometric authentication features like fingerprint and retinal scanning, patients can safely retrieve their medical records in real-time, even during emergencies. Furthermore, the platform reduces the risk of fraud and data breaches by utilizing blockchain technology to guarantee the integrity and traceability of each interaction in the medical field.

Additionally, the platform breaks through traditional barriers by offering educational materials that help users gain a deeper comprehension of different illnesses and wellness techniques. Through the promotion of a proactive and informed culture, people are empowered to take control of their health journey.

This abstract explores the relationship between blockchain technology and healthcare, emphasizing how it could transform patient care delivery and redefine data security standards. Data integrity and patient empowerment are key components of this new era of innovation in healthcare, which is ushered in by a multifaceted approach that includes security, accessibility, and education. [1],[2],[3].

II. EXISTING SYSTEM

This Centralized Electronic Health Records (EHR) Systems: Centralized EHR systems were put in place by numerous healthcare organizations in order to digitize and store patient medical records. These systems usually include a central database that is run by the healthcare organization or provider and contains patient information like medical history, diagnosis, Medication and treatments are kept in storage. User authentication procedures are typically used to restrict access to EHRs, and security

measures like encryption and access controls are put in place to protect patient privacy and adhere to laws like HIPAA. [17],[18],[19].

Health Information Exchanges (HIEs): The sharing of medical information between different healthcare organizations and providers is facilitated by health information exchanges. Licensed healthcare providers can access patient data from various sources through these exchanges, enhancing care coordination and continuity. Federated models and centralized databases—which store data across multiple systems within participating organizations and enable access to it through a single platform—are commonly used by HIEs.. [20],[21],[22].

Cloud-Based Storage Solutions: Since the inception of cloud computing, numerous healthcare organizations have adopted cloud-based storage solutions for the management and archiving of medical data. Cloud storage companies offer scalable, secure platforms for storing data related to health including electronic medical records and imaging data. Data security measures, such as encryption, access controls, and regular backups, are implemented to safeguard patient data stored on cloud platforms.. [23],[24],[25].

Data Warehousing and Analytics Platforms: Healthcare organizations use data warehousing and analytics platforms to gather, analyze, and derive insights from vast volumes of medical data. These systems facilitate clinical decision-making, research, and quality-improvement initiatives by combining data from various sources, including laboratory systems, medical devices, and electronic h h records. Data security procedures are implemented to ensure the confidentiality, availability, and integrity of patient data processed and stored on these platforms.

Customized Software Solutions: Some healthcare organizations develop or employ customized software programs that are tailored to their particular needs for managing and storing patient data. Practice management software, electronic health record systems, and patient portals that enable secure access to medical records and provider communication are some examples of these solutions. In order to protect patient privacy and comply with regulatory requirements, these systems are equipped with security features such as encryption, audit trails, and role-based access controls.

Demerits:

Complexity and Technical Expertise: Implementing and managing a blockchain-based system like Secure Health requires significant technical expertise and resources. Organizations may face challenges in understanding and integrating blockchain technology into their existing infrastructure, as well as maintaining and updating the system over time.

Scalability: Despite the promise of blockchain technology for enhancing security and decentralization, scalability remains a significant concern. As the volume of patient data and transactional activity increases, blockchain networks may struggle to handle the growing demand, leading to slower processing times and higher transaction fees.

Regulatory Compli ce: Compliance with existing healthcare regulations, such as HIPAA in the United States, can be challenging in a blockchain-based system like SecureHealth. Ensuring that patient data stored on the blockchain complies with privacy and security regulations while still maintaining transparency and accessibility requires careful navigation of legal and regulatory frameworks.

Privacy Concerns: While blockchain offers transparency and immutability, it also raises privacy concerns, particularly regard the exposure of sensitive health information. While access to patient data on the blockchain may be restricted through encryption and authentication mechanisms, there is still a risk of unintended data exposure or unauthorized access.

Integration Challenges: Integrating SecureHealth with existing healthcare systems and workflows may be complex and time-consuming. Ensuring amless interoperability between SecureHealth and other electronic health record (EHR) systems, medical devices, and healthcare applications requires careful planning and coordination.

Cost: Building and maintaining a blockchain-based system like SecureHealth can be costly, particularly in terms of infrastructure, development, and ongoing operational expenses. Organizations must carefully evaluate the return on investment and weigh the costs against the benefits of implementing a blockchain solution.

User Adoption: User adoption and acceptance of SecureHealth may be a challenge, particularly among healthcare professionals and patients who are unfamiliar with blockchain technology. Education and training initiatives may be necessary to promote under anding and confidence in using SecureHealth effectively.High capital expenditure required to invest in automation (It can cost around millions to design, fabricate, and install).

III. PROPOSED SYSTEM

Client Interface: The system's user-friendly interface facilitates interaction between users, including physicians. They can safely upload health data with this interface.

NLP Model Service: The client in ace sends the uploaded data to this service. makes use of a Natural Language Processing (NLP) model to assess how serious the health data is.adds the data and severity score to the message queue.

Message Queue (Kafka): serves as a barrier between the blockchain microservice and the NLP model service. obtains information and severity ratings from the NLP model provider. temporarily keeps the data until the blockchain microservice has had a chance to process it.

Blockchain Microservice (Spring Boot): receives data, including severity scores, from the message queue. calculates and appends the block hash to the blockchain. manages the blockchain network, adding new blocks and approving transactions. ensures the accuracy and confidentiality of the data stored on the blockchain.

Security and Consensus Mechanism: Use Proof of Security (PoS), your own consensus method, to guarantee the safety and integrity of the blockchain network.Establish guidelines for adding blocks, verifying transactions, and achieving node consensus.

Load Balancer: For scalability and fault tolerance, the NLP model service, blockchain microservice, and other components divide up incoming requests among several instances.

Database: keeps extra information about transactions, blocks, users, and other details for analysis and auditing needs.

Monitoring and Logging: Establish logging and monitoring systems to keep tabs on the system's security, health, and performance. Use logging and monitoring tools such as Prometheus, Grafana, ELK stack, and so on.

Authentication and Authorization: To guarantee that only authorized users (administrators, doctors, etc.) can access and modify data on the blockchain network, implement strong authentication and authorization procedures..

Encryption: Implement encryption techniques to secure data both in transit and at rest to prevent unauthorized access or tampering.

Compliance: Ensure compliance with relevant regulations such as HIPAA (Health Insurance Portability and Accountability Act) for handling sensitive health data.

Doctor Access Interface: Develop a separate interface for doctors to access patient data stored on the blockchain during emergencies. Implement features such as search functionality, patient identification, and secure access controls.

Blockchain Query Service: Create a module or service that handles blockchain-based patient data queries. During emergencies, make sure that data is retrieved quickly and securely using the different search parameters that doctors provide.

Data Encryption and Decryption: Before putting patient data on the blockchain, encrypt it to protect privacy and confidentiality. Decrypted data can be safely retrieved and presented to authorized doctors during emergencies by implementing decryption mechanisms within the query service.

Access Control Mechanism: Put in place a strong access control system to limit access to patient information kept on the blockchain. Make sure that in an emergency, patient data can only be accessed by authorized physicians who possess the necessary credentials and permissions.

Real-time Notifications: Put in place real-time notification systems to inform physicians when new patient data is uploaded to the blockchain or is updated. Make sure physicians are notified in a timely manner about important patient information.

Emergency Data Retrieval Protocol: Establish a uniform procedure that specifies how physicians should ask for and obtain patient data in an emergency. Incorporate protocols for authorization, data retrieval, and authentication to optimize workflow and guarantee adherence to legal requirements.

Audit Trail and Logging: Keep a record of every action taken by doctors to access and retrieve data during emergencies. Keep track of pertinent data, including timestamps, user names, and actions taken, to make compliance auditing and accountability easier.

Testing and Simulation: To confirm the system's functionality, dependability, and responsiveness in emergency situations, carry out extensive testing and simulation exercises. Make use of simulated emergencies and realistic test cases to find and fix any possible problems or bottlenecks.

Fingerprint Authentication Service: Provide a fingerprint authentication service that uses the patient's fingerprint to confirm their identity. To guarantee that access to patient data is only authorized after the patient's fingerprint is authenticated, integrate this service with the doctor access interface.

Smart Contracts: Utilize blockchain smart contracts to control data visibility and access control. Establish rules in smart contracts to impose access restrictions on patient data through fingerprint authentication.

Biometric Data Storage: Save the patient's fingerprint data off-chain in an encrypted format or safely on the blockchain. Make sure that all rules pertaining to the handling and storage of biometric data are followed.

Access Control Policies: Create policies for access control in smart contracts so that doctors can access patient data

only after the patient's fingerprint is verified. Provide procedures for withdrawing access in the event that the patient modifies their authorization or withdraws their consent.

Transaction Verification: To verify that fingerprint authentication transactions are authentic, include transaction verification mechanisms in smart contracts. Make sure that the blockchain records only transactions with approved fingerprint authentication.

Event Logging: On the blockchain, record fingerprint authentication events for audit and traceability purposes. Add pertinent data, such as patient IDs, timestamps, and the outcomes of fingerprint authentication.

User Experience: seamless integration of the fingerprint authentication process with the doctor access interface. To guarantee a seamless user experience, give doctors and patients clear instructions and feedback throughout the authentication process.

Security Considerations: Protect sensitive data, such as patient identifiers and fingerprint authentication records, by implementing encryption techniques (such as AES). To reduce possible risks and vulnerabilities, evaluate and update security measures on a regular basis.

With the additional layer of biometric authentication to confirm the patient's identity during data access by doctors, you can guarantee safe and auditable access to patient data by integrating fingerprint authentication with the blockchain implementation. This is not automation; rather, it is semi-automation.

Merits:

"SecureHealth" offers several advantages over traditional methods of storing data in a database or using Proof of Work (PoW) based blockchain implementations:

Enhanced Security: Modern encryption techniques like AES are used by SecureHealth to safeguard patient data. Private health information is protected from tampering or unauthorized access thanks to this.Because a blockchain creates an immutable and tamper-evident ledger, storing data on it offers an additional layer of security over traditional databases. To further increase security, only authorized individuals can access patient data thanks to SecureHealth's fingerprint authentication.

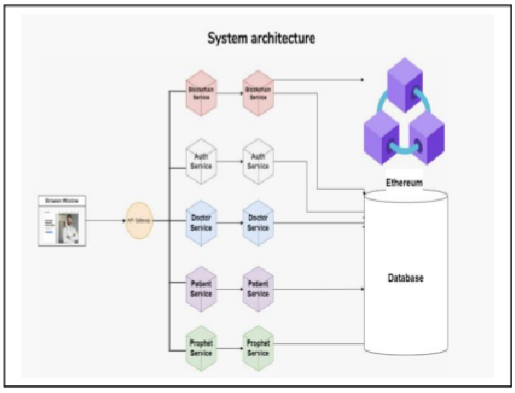
Data Integrity and Immutability: SecureHealth uses a blockchain to ensure data immutability and integrity. Once information is saved on the blockchain, it cannot be altered or removed without the consent of all network users. As a result, the system is very impervious to manipulation or corrupted data.In conventional databases, data breaches, corruption, or unapproved changes could compromise the integrity of medical records.

Transparent and Auditable Transactions: Blockchain technology encourages transparency by making transaction histories visible to all network users. This transparency boosts trust in the system and accountability. SecureHealth ensures that all patient data-related transactions, including requests for access and modifications, are recorded on the network through the use of blockchain technology. This creates an open, verifiable trail of data usage and access..

Decentralization and Resilience: SecureHealth's blockchain implementation most likely makes use of a decentralized network of nodes, which reduces the likelihood of a single point of failure and increases the system's resilience to hardware issues or cyberattacks. Conversely, traditional database systems might be built upon centralized server architecture, which is vulnerable to interruptions, data breaches, and outages.

Efficient and Scalable Consensus Mechanism: Proof of Sceviarity (PoS), a proprietary consensus mechanism developed by SecureHealth, is likely more efficient and scalable than Proof of Work (PoW) consensus, which is used in many public blockchains. As PoS typically consumes less energy and computational resources than PoW, it is a more cost-effective and environmentally responsible approach to blockchain network security.

System Architecture



IV. Conclusion And Future Work:

Using blockchain technology, SecureHealth offers a viable way to improve the security, integrity, and transparency of healthcare data management. SecureHealth provides strong protection against unauthorized access, tampering, and data breaches by utilizing cutting-edge encryption techniques, immutable ledger technology, and decentralized consensus mechanisms. Improved patient outcomes and healthcare delivery are fostered by the platform's emphasis on data integrity, transparency, and auditability, which guarantees trust and accountability in the healthcare ecosystem. To fully realize SecureHealth's potential in transforming

healthcare data management, however, issues like scalability, regulatory compliance, privacy concerns, integration complexities, cost, and user adoption must be carefully addressed. SecureHealth has the potential to usher in a new era of safe, effective, and patient-centered healthcare services with the right preparation, cooperation, and investment.

Advanced Data Analytics: By integrating machine learning algorithms and advanced analytics, SecureHealth may be able to provide real-time insights and predictive analytics to help with clinical decision-making, spot trends, and enhance patient outcomes. Healthcare organizations could find important insights for research and population health management by utilizing the massive amount of data stored on the blockchain.

Improved Privacy Functionalities: Sensitive health data stored on the blockchain may be further protected from prying eyes by integrating privacy-enhancing technologies like homomorphic encryption and zero-knowledge proofs. By using these methods, data could be shared or securely analyzed without disclosing the patient data underneath.

Patient-Centric Features: Developing patient-centric features within SecureHealth, such as personalized health profiles, health tracking tools, and interactive educational resources, would empower individuals to actively manage their health and engage in informed decision-making. This emphasis on patient empowerment and education would promote proactive healthcare management and preventive care initiatives.

REFERENCES

[1] Stanfill, M.H.; Marc, D.T. Health information management: Implications of artificial intelligence on healthcare data and information management. Yearb. Med. Inform. 2019, 28, 56–64. [Google Scholar] [CrossRef] [PubMed] [Green Version]

[2] Adamu, J.; Hamzah, R.; Rosli, M.M. Security issues and framework of electronic medical record: A review. Bull. Electr. Eng. Inform. 2020, 9, 565–572. [Google Scholar]

[3] Enaizan, O.; Zaidan, A.A.; Alwi, N.; Zaidan, B.B.; Alsalem, M.A.; Albahri, O.; Albahri, A. Electronic medical record systems: Decision support examination framework for individual, security and privacy concerns using multi-perspective analysis. Health Technol. 2020, 10, 795–822. [Google Scholar]

[4] Hulsen, T. Sharing is caring—Data sharing initiatives in healthcare. Int. J. Environ. Res. Public Health 2020, 17, 3046. [Google Scholar] [PubMed]

[5] Ghafur, S.; Van Dael, J.; Leis, M.; Darzi, A.; Sheikh, A. Public perceptions on data sharing: Key insights from the UK and the USA. Lancet Digit. Health 2020, 2, e444–e446. [Google Scholar] [PubMed]

[6] Schwalbe, N.; Wahl, B.; Song, J.; Lehtimaki, S. Data sharing and global public health: Defining what we mean by data. Front. Digit. Health 2020, 2, 612339. [Google Scholar]

[7] Kish, L.J.; Topol, E.J. Unpatients—Why patients should own their medical data. Nat. Biotechnol. 2015, 33, 921–924. [Google Scholar]

[8] Wang, Y.; Li, P.-F.; Tian, Y.; Ren, J.-J.; Li, J.-S. A shared decision-making system for diabetes medication choice utilizing electronic health record data. IEEE J. Biomed. Health Inform. 2016, 21, 1280–1287. [Google Scholar] [CrossRef]

[9] Singh, C.; Chauhan, D. IoT–Blockchain Integration-Based Applications Challenges and Opportunities. Mob. Radio Commun. 5g Netw. Proc. MRCN 2020, 2020, 87–116. [Google Scholar]

[10] Lin, B.; Huang, Y.; Zhang, J.; Hu, J.; Chen, X.; Li, J. Cost-driven off-loading for DNN-based applications over cloud, edge, and end devices. IEEE Trans. Ind. Inform. 2019, 16, 5456–5466. [Google Scholar] [CrossRef] [Green Version]

[11] Thilakanathan, D.; Chen, S.; Nepal, S.; Calvo, R.; Alem, L. A platform for secure monitoring and sharing of generic health data in the Cloud. Future Gener. Comput. Syst. 2014, 35, 102–113. [Google Scholar] [CrossRef]

[12] Yang, J.-J.; Li, J.-Q.; Niu, Y. A hybrid solution for privacy preserving medical data sharing in the cloud environment. Future Gener. Comput. Syst. 2015, 43, 74–86. [Google Scholar] [CrossRef]

[13] Zhu, H.; Liu, X.; Lu, R.; Li, H. Efficient and privacy-preserving online medical prediagnosis framework using nonlinear SVM. IEEE J. Biomed. Health Inform. 2016, 21, 838–850. [Google Scholar] [CrossRef] [PubMed]

[14] Michalas, A.; Weingarten, N. Healthshare: Using attribute-based encryption for secure data sharing between multiple clouds. In Proceedings of the 2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS), Thessaloniki, Greece, 22–24 June 2017; pp. 811–815. [Google Scholar]

[15] Fang, H.S.A.; Tan, T.H.; Tan, Y.F.C.; Tan, C.J.M. Blockchain personal health records: Systematic review. J. Med. Internet Res. 2021, 23, e25094. [Google Scholar] [CrossRef]

[16] Westphal, E.; Seitz, H. Digital and decentralized management of patient data in healthcare using blockchain implementations. Front. Blockchain 2021, 4, 732112. [Google Scholar] [CrossRef]

[17] Kuo, T.-T.; Kim, H.-E.; Ohno-Machado, L. Blockchain distributed ledger technologies for biomedical and health care applications. J. Am. Med. Inform. Assoc. 2017, 24, 1211–1220. [Google Scholar] [CrossRef] [PubMed] [Green Version]

[18] Soltanisehat, L.; Alizadeh, R.; Hao, H.; Choo, K.-K.R. Technical, temporal, and spatial research challenges and opportunities in blockchain-based healthcare: A systematic literature review. IEEE Trans. Eng. Manag. 2020, 70, 353–368. [Google Scholar] [CrossRef]

[19] Abu-Elezz, I.; Hassan, A.; Nazeemudeen, A.; Househ, M.; Abd-Alrazaq, A. The benefits and threats of blockchain technology in healthcare: A scoping review. Int. J. Med. Inform. 2020, 142, 104246. [Google Scholar] [CrossRef]

[20] Saha, A.; Amin, R.; Kunal, S.; Vollala, S.; Dwivedi, S.K. Review on "Blockchain technology based medical healthcare system with privacy issues". Secur. Priv. 2019, 2, e83. [Google Scholar] [CrossRef]

[21] Hasselgren, A.; Kralevska, K.; Gligoroski, D.; Pedersen, S.A.; Faxvaag, A. Blockchain in healthcare and health sciences—A scoping review. Int. J. Med. Inform. 2020, 134, 104040. [Google Scholar] [CrossRef]

[22] Jin, H.; Luo, Y.; Li, P.; Mathew, J. A review of secure and privacy-preserving medical data sharing. IEEE Access 2019, 7, 61656–61669. [Google Scholar] [CrossRef]

[23] Dubovitskaya, A.; Novotny, P.; Xu, Z.; Wang, F. Applications of blockchain technology for data-sharing in oncology: Results from a

RE-2022-220545-plag-report

ORIGINALITY REPORT

9%	3%	6%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Higher Education Commission Pakistan Student Paper	1%
2	todaysmeet.com Internet Source	1%
3	Submitted to University of Liverpool Student Paper	1%
4	Rashmi Pathak, Badal Soni, Naresh Babu Muppalaneni. "Chapter 74 Significance and Challenges in Blockchain-Based Secure Sharing of Healthcare Data", Springer Science and Business Media LLC, 2024 Publication	1%
5	Submitted to Champlain College Student Paper	1%
6	Submitted to Purdue University Student Paper	1%
7	Submitted to University of Northampton Student Paper	1%

8	Submitted to Polytechnic Institute Australia Student Paper	<1%
9	Sanjay Kumar Jena, Ram Chandra Barik, Rojalina Priyadarshini. "A systematic state-of-art review on digital identity challenges with solutions using conjugation of IOT and blockchain in healthcare", Internet of Things, 2024 Publication	<1%
10	"Sustainable Communication Networks and Application", Springer Science and Business Media LLC, 2020 Publication	<1%
11	www.devx.com Internet Source	<1%
12	www.mdpi.com Internet Source	<1%
13	0-www-mdpi-com.brum.beds.ac.uk Internet Source	<1%
14	Amina Kessentini, Ibtissem Wali, Mayssa Jarray, Nouri Masmoudi. "Enhancing E-Health with Secure IoT Architecture Leveraging Blockchain and AI", 2023 IEEE 11th International Conference on Systems and Control (ICSC), 2023 Publication	<1%

15	Chetna Tiwari, Anuradha. "Potential of Blockchain Technology in Healthcare, Finance, and IoT", Wiley, 2023 Publication	<1%
16	Chaoran Li, Jusheng Liu, Guanyu Qian, Ziyi Wang, Jingti Han. "Double chain system for online and offline medical data sharing viaprivate and consortium blockchain: A system design study", Frontiers in Public Health, 2022 Publication	<1%
17	Tarun Kumar Vashishth, Vikas Sharma, Kewal Krishan Sharma, Bhupendra Kumar, Sachin Chaudhary, Rajneesh Panwar. "chapter 14 Serverless Computing Real-World Applications and Benefits in Cloud Environments", IGI Global, 2024 Publication	<1%
18	link.springer.com Internet Source	<1%
19	Hamed Taherdoost. "The Role of Blockchain in Medical Data Sharing", Cryptography, 2023 Publication	<1%
20	Niu Muqing, Liu Wenting, Gao Xiaole, Zhao Zhiyuan, Wang Jingjuan. "IoT, Heterogeneous Data Processing, and AI Automation Synergy for Improved Efficiency and Maintenance:	<1%

Revolutionizing Hospital Operations", 2023 IEEE International Conference on Signal Processing, Communications and Computing (ICSPCC), 2023
Publication

21 "Computer Security. ESORICS 2023 International Workshops", Springer Science and Business Media LLC, 2024 <1%
Publication

22 Mohammed K. Elghoul, Sayed F. Bahgat, Ashraf S. Hussein, Safwat H. Hamad. "Management of medical record data with multi-level security on Amazon Web Services", SN Applied Sciences, 2023 <1%
Publication

Exclude quotes	On	Exclude matches	Off
Exclude bibliography	On		

REFERENCES

[1] Luis Sánchez, Jorge Lanza, Juan Ramón Santana, Pablo Sotres, Víctor González, Laura Martín, " Data Enrichment Toolchain: A Data Linking and Enrichment Platform for Heterogeneous Data" in *IEEE Access*, date of publication Sep 20, 2023. https://ieeexplore.ieee.org/document/10256189

[2] Ruiyu Liang , Chaoran Huang, Chengguo Zhang, Binghao, "Exploring the Fusion Potentials of Data Visualization and Data Analytics in the Process of Mining Digitalization" in *IEEE Access*, date of publication April 17, 2023. https://ieeexplore.ieee.org/document/10103681

[3] Julius Möller, Dennis Jankowski, Arne Lamm L. "Data Management Architecture for Service-Oriented Maritime Testbeds" https://ieeexplore.ieee.org/document/9893899

[4] Yining Wang, Bin Liang, Tian Wang, "A Big Data Stream-Driven Risk Recognition Approach for Hospital Accounting Management Systems" https://ieeexplore.ieee.org/document/10320321

[5] Dong Wang, Zhigang Zhang, Chenyang Xu and Zhuohao Wang."Data Query Method of Science and Technology Management Based on Relational Engine" https://ieeexplore.ieee.org/document/9442601Enda O'Shea,Rafflesia Khan,Ciara Breathnach, and Tiziana Margaria."Towards Automatic Data Cleansing and Classification of Valid Historical Data An Incremental Approach Based on MDD". https://ieeexplore.ieee.org/document/9378148

[6] Siyuan Qi, Baoxiong Jia,Siyuan Huang, Ping Wei, and Song-Chun Zhu." A Generalized Earley Parser for Human Activity Parsing and Prediction". https://ieeexplore.ieee.org/document/9018126

[7] Kang Le and Lei Wang."Research and design of Metadata management system of Party Building Information in universities based on Big data". https://ieeexplore.ieee.org/document/10314271