[Introduction: 2](#_Toc135493689)

[1.1 Rationale and Structure 2](#_Toc135493690)

[1.2 User Scenario 3](#_Toc135493691)

[1.3 Approach 3](#_Toc135493692)

[2 Use Case Model 3](#_Toc135493693)

[3 Business Process Model 4](#_Toc135493694)

[4 Technical Framework 6](#_Toc135493695)

[4.1 Geographical Distribution 6](#_Toc135493696)

[4.2 Overall Architecture 6](#_Toc135493697)

[5 Overall Design 9](#_Toc135493698)

[5.1 System Components and Layering 9](#_Toc135493699)

[5.2 Front-End Design 9](#_Toc135493700)

[6 Technology and Tools 13](#_Toc135493701)

[7 User Manual 14](#_Toc135493702)

[8 Work Log 17](#_Toc135493703)

[Postman HTTP Client: 17](#_Toc135493704)

[9 Summary 19](#_Toc135493705)

[9.1 Major Learnings 19](#_Toc135493706)

[9.2 Future Work 20](#_Toc135493707)

[10 URL’s 20](#_Toc135493708)

[11 REST API Methods 20](#_Toc135493709)

Cloud Project

AIMS-B-CONNECT

# Introduction:

## 1.1 Rationale and Structure

This report details the AIMS-b-Connect chat web application, which was developed by a group of students as part of their Cloud Computing coursework at AUT (Auckland University of Technology). The application is created to highlight their understanding of cloud computing by implementing a moderately complex cloud application that uses organization across several web services to fulfil requests.

The report begins by introducing the AIMS-b-Connect chat web app, its purpose, and the methodology used to scope, plan, and manage the project. Following this, the Requirements and Analysis section provides a concise overview of the solution requirements and the approach used to analyse them. The Design section provides an overview of the chat app activities and the process model used to develop it.

The Architecture and Technical Design section investigates into the overall architecture of the application, including the back end, middleware, and front-end designs, as well as the various technologies and tools utilized in designing, delivering, and documenting the solution. Additionally, the User Manual section describes how users can interact with the solution.

Finally, the Lessons Learnt section details the key insights gleaned from the development process, as well as planned further work.

## 1.2 User Scenario

Aims-b-connect is a chat app that allows users to communicate with each other via text messages. This app is designed for anyone who wants to easily connect with their friends, family, or colleagues.

1. One scenario involves a user who has already registered for the Aims-b-connect app and can be found in search items by other registered users.
2. The other scenario involves a user who is not yet registered on Aims-b-connect and therefore cannot be found in search items by other users.

## 1.3 Approach

Before starting the app's development, the group members met to discuss the approach. This meeting's aim was to ensure that all members were aligned on the project's goals and the development methodology used.   
Figure 1 shows a little of the rough work proof of the meeting.

## 2 Use Case Model

To offer a comprehensive visual representation of the planned functionalities, we crafted a Use Case Model in Fig. 2, using the creativity of Creately. This model provides an organized and graphical overview of the features to be delivered. The highlighted areas within the diagram signify the specific scope that has been carefully chosen for development in our initial release. By focusing on these highlighted sections, we aim to deliver a refined and impactful user experience, showcasing the core capabilities of our chat software right from the start.

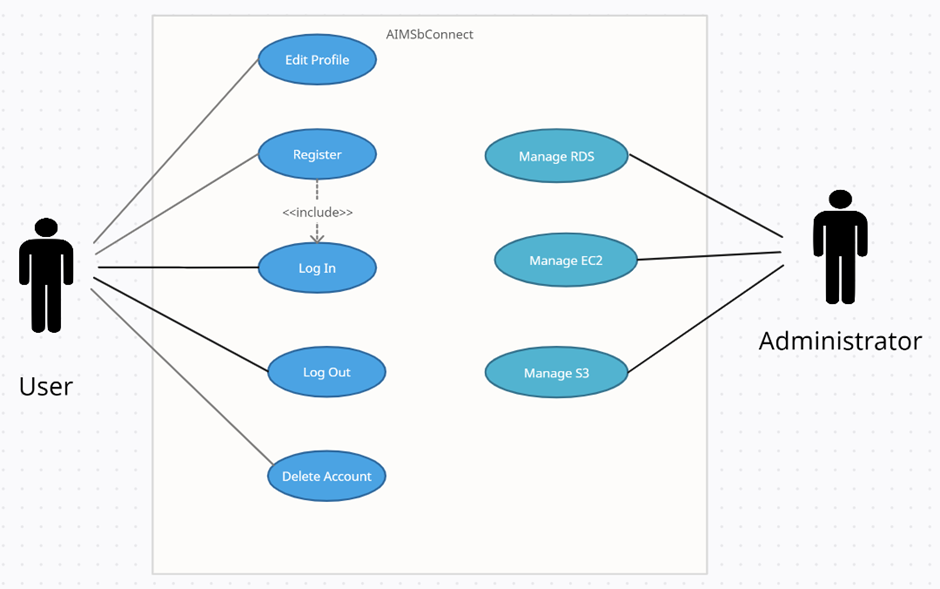


Fig 2. User case model for AIMSbConnect.

# Business Process Model

The BPMN Diagram depicted in Figure 3 illustrates the comprehensive design of the business logic for the AIMSbConnect app's functionality.

A computer screen shot of a computer screen

Description automatically generated with low confidenceThis model effectively visualizes the message flows between different entities involved in the chat process. the BPMN Diagram provides a clear representation of the chat app's workflow, enabling a better understanding of the message exchange and interaction between different components.

Fig 3 BPMN Diagram for AIMSbConnect

# Technical Framework

## 4.1 Geographical Distribution

Table 1 showcases the geographical distribution of the system components for AIMSbConnect across various datacentre locations worldwide.

|  |  |
| --- | --- |
| **System Modules** | **Location** |
| **Database** | N. Virginia |
| **S3 Service** | N. Virginia |
| **Backend REST API services** | Sydney |
| **Web Server** | Sydney |

Table 1

## 4.2 Overall Architecture

The high-level architectural framework implemented for AIMSbConnect is visually represented in Figure 4. This diagram provides an overview of the structured design and interrelationships of the system's key components and modules. It serves as a blueprint for the overall system architecture of AIMSbConnect, enabling efficient communication and seamless functionality between its various elements.

A picture containing text, diagram, screenshot, plan

Description automatically generated

Fig 4 AIMSbConnect Framework

Below, Figure shows instances of the REST services that were utilized by the AIMSbConnect.

A picture containing text, diagram, screenshot, line

Description automatically generated

Figure 5: REST Services

# Overall Design

## 5.1 System Components and Layering

**Architecture Layers  
Backend Layer**: The backend layer of the AIMSbConnect is responsible for handling the server-side logic and data management. It is implemented using JavaScript, specifically using Express.js. The backend layer incorporates CRUD functionality, allowing users to create, read, update, and delete messages. For data storage, PostgreSQL is chosen as the database solution, ensuring efficient and reliable data management.

**Frontend Layer:** The front-end layer is the presentation layer it focuses on the user interface and user experience of the AIMSbConnect. It utilizes HTML and JavaScript, with a primary emphasis on React.js, a popular JavaScript library for building dynamic and responsive user interfaces.

**Middleware Layer:** The middleware layer acts as a bridge between the frontend and backend layers, facilitating communication and data exchange. It ensures seamless integration and interoperability between the client-side and server-side components. Additionally, the middleware layer incorporates various AWS services to enhance the app's functionality and performance. AWS's Simple Storage Service (S3) is employed for storing static images, while the Relational Database Service (RDS) is utilized for hosting databases, enabling efficient data storage and retrieval.

## 5.2 Front-End Design

The AIMSbConnect app presents users with an engaging and user-friendly front page, serving as the entry point to the application. The initial focus during the development process was to design an aesthetically pleasing user interface that provides a seamless and intuitive experience for users. The front page prominently features a login section, where users can enter their credentials, including a username and password, to access the app's functionalities.

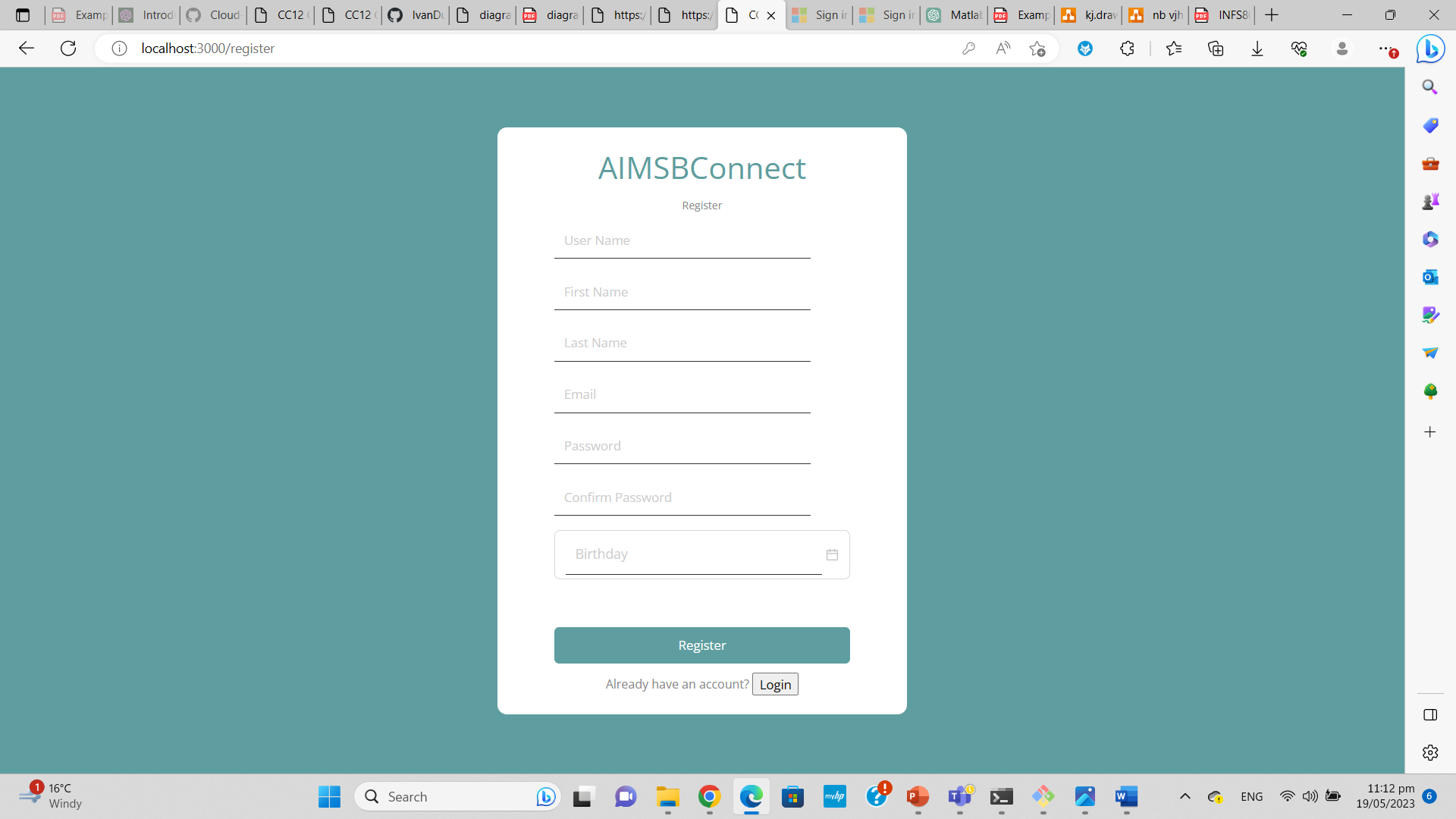
Additionally, the interface offers a convenient registration option for new users who wish to create an account. This feature is easily accessible through a dedicated Register tab, enabling a straightforward and hassle-free registration process.

The design of the user interface prioritizes both visual appeal and usability, ensuring that users are greeted with an attractive and welcoming interface that facilitates effortless navigation and interaction. By prioritizing a user-cantered approach, the AIMSbConnect app aims to deliver a positive and enjoyable user experience right from the initial login page. Figures 6 and 7 below show the login and Register page, respectively.

A screenshot of a computer

Description automatically generated

Fig 6 Sows Login page of AIMSbConnect

Fig 7 Shows the register page of the app AIMSbConnect

A computer screen shot of a blue car

Description automatically generated with medium confidenceFigure 8 below illustrates the post-login page of the application, displaying a well-structured layout. The left section showcases the list of friends, while the right side exhibits the ongoing chat with the selected contact. At the top left corner, users can find their profile picture alongside their name for easy identification. Additionally, the chat history with the chosen contact is conveniently displayed. The list of contacts includes both the name and a corresponding profile picture. By clicking on a user's name, there is an option to modify the displayed name, and clicking on the profile picture enables users to update their profile picture as well.

Fig 8 Shows the user interface of the app AIMSbConnect

## 5.3 Middleware

## 5.4 Back-end Design

## EC2

Fig – Fig shows all backend design.

We have selected an Amazon EC2 instance of type t2. micro, running on Amazon Linux, to host our backend service. The choice of t2. micro is advantageous because it falls under the free tier eligibility, providing cost-effective hosting capabilities.

To meet the storage requirements of our application while maintaining optimal performance, we have attached an 8 GiB Amazon EBS gp3 volume as the root volume.

To securely manage the EC2 instance, we have generated a key-pair in PEM format. This key-pair enables us to establish secure SSH connections for administration and maintenance tasks.

Overall, this setup takes advantage of the benefits offered by the t2. micro instance type's free tier eligibility, while ensuring efficient storage management through the attached Amazon EBS gp3 volume. The generated PEM key-pair adds an extra layer of security and simplifies access to the EC2 instance for seamless administration and maintenance.

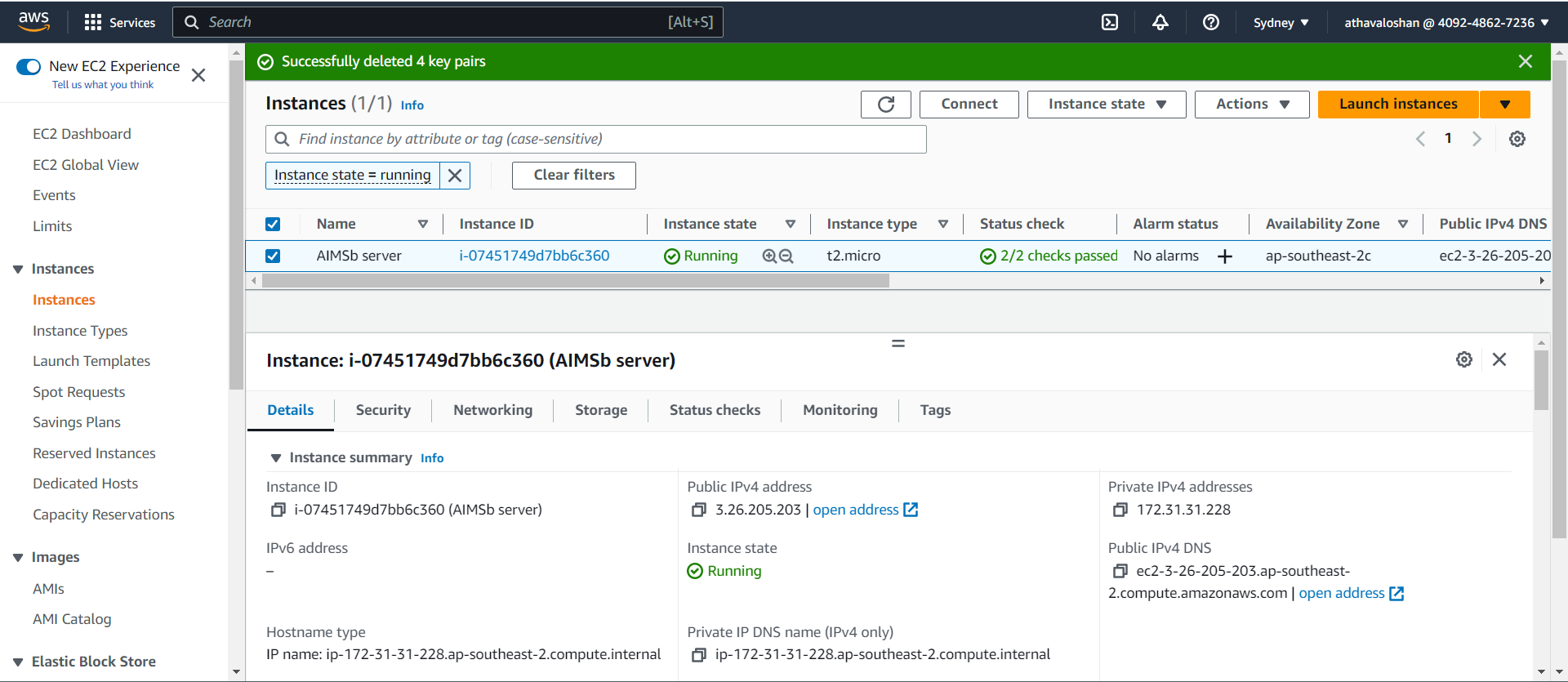


Fig 9

We have adjusted the security group of our EC2 instance to permit incoming TCP traffic on port 3001. This modification is specifically intended to enable access to the REST service of our application, which is listening on that port. By allowing TCP traffic on this specific port, we ensure smooth communication with the REST service, facilitating the proper handling of incoming requests.

A screenshot of a computer

Description automatically generated with medium confidence

Fig 9

S3

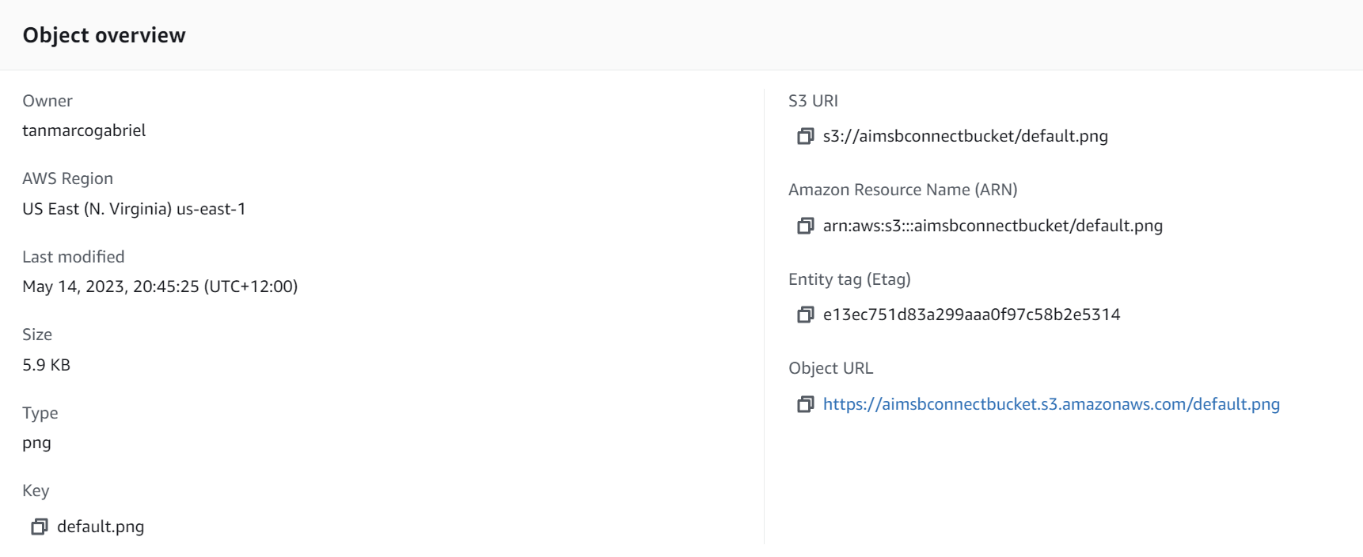


Fig 10

## RDS

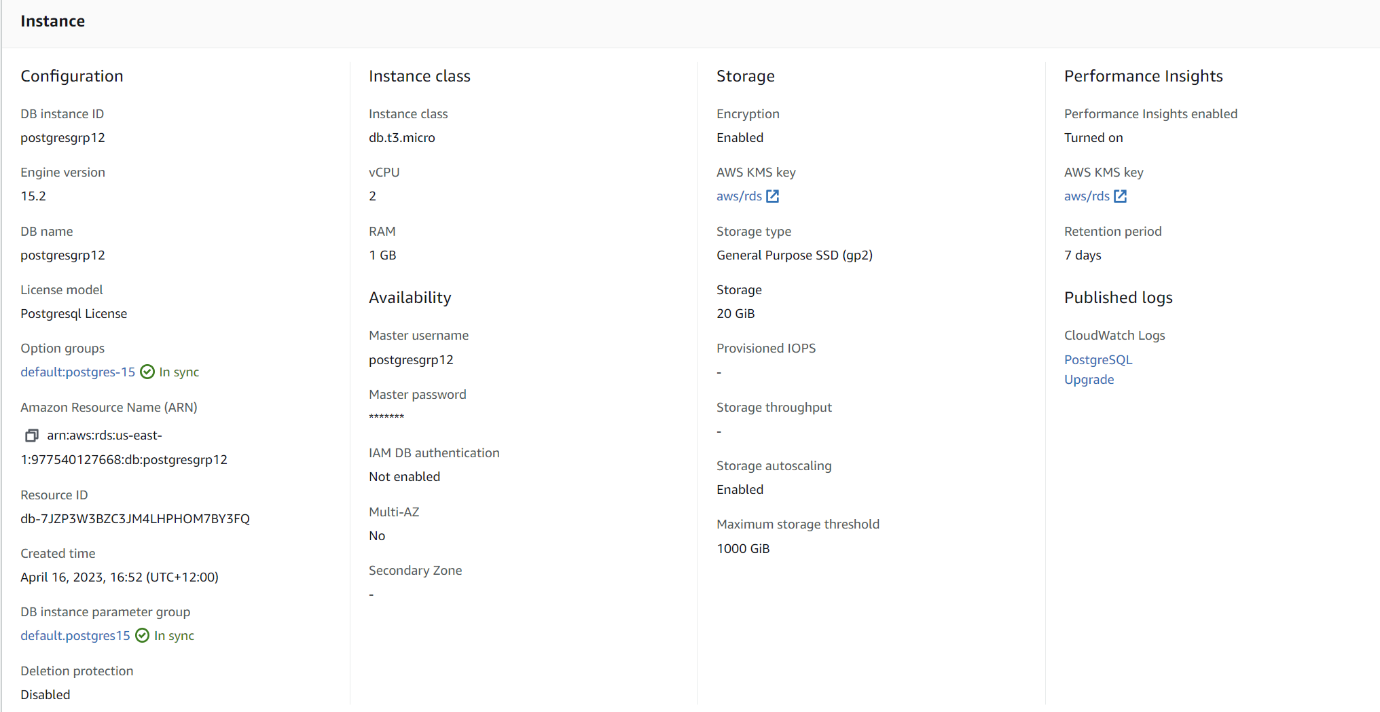
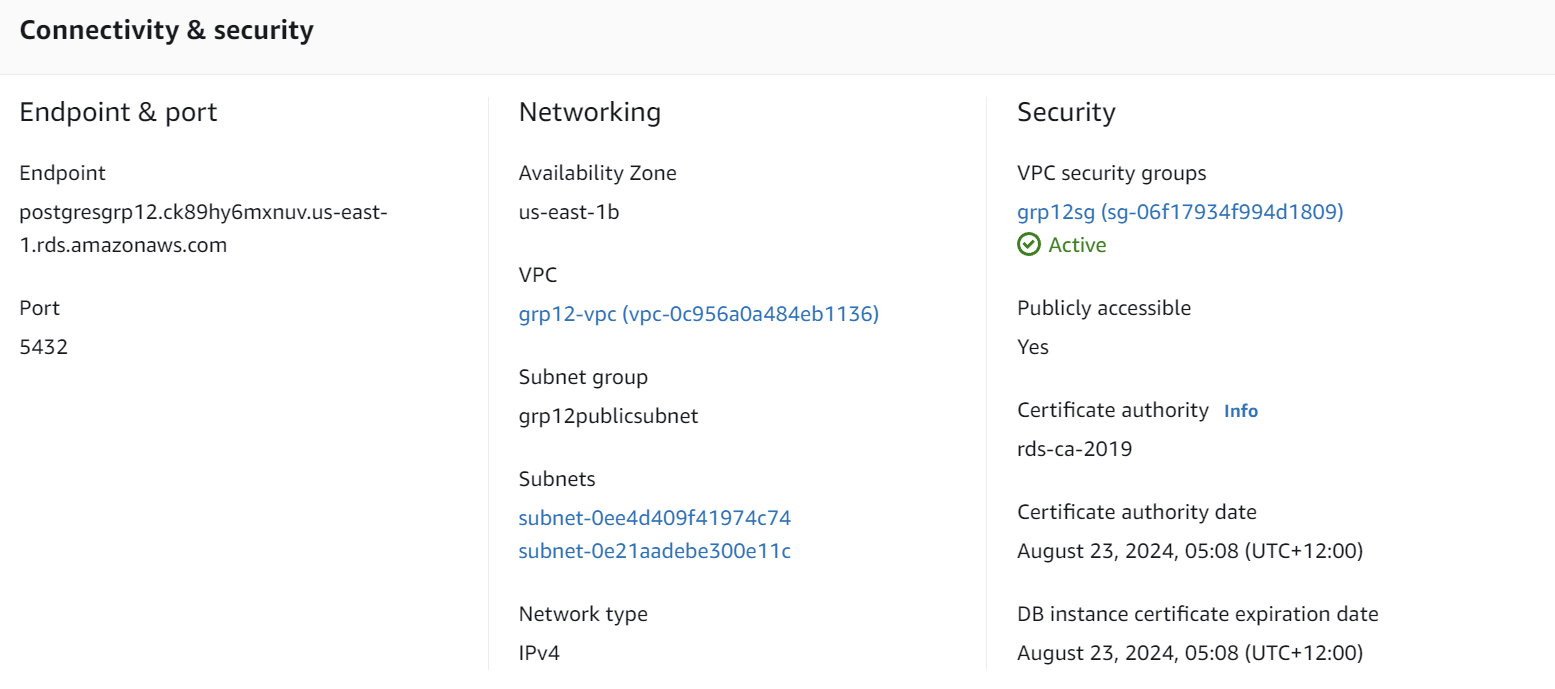


Fig 11

# Technology and Tools

The AIMSbConnect web service utilizes a diverse range of programming languages and technologies during its design, development, testing, and documentation phases. The following table 2 provides a comprehensive list of these tools and technologies employed in the creation of AIMSbConnect:

|  |  |
| --- | --- |
| Technology | How Used |
| Express.js | Back-end web application framework |
| PostgreSQL | Database engine |
| Node.js | Server environment |
| JavaScript | Programming language |
| HTML | Displaying front end in web browser |
| CSS | Front end styling language |
| Sass | Cascading style sheets |
| React | Building user interface |
| Redux Store | Stores all the states of an application |
| Axios | Promise-based HTTP Client |
| pgAdmin | Manage database |
| Postman | Test APIs |
| MS Visio | Modelling and diagramming Tool |
| bpmn.io | Business process model diagram |

Table 2 shows range of programming languages and technologies during AIMSbConnect design

# 7 User Manual

Below is a step-by-step user manual explaining the user interface of AIMSbConnect, a user-friendly and organized chatting software:

1. Registration: (Fig 9)
   * Visit the AIMSbConnect website and click on the "Register" button.
   * Fill in the required information, such as your name, DOB, email address, and password.
   * Click "Register" to create your account.

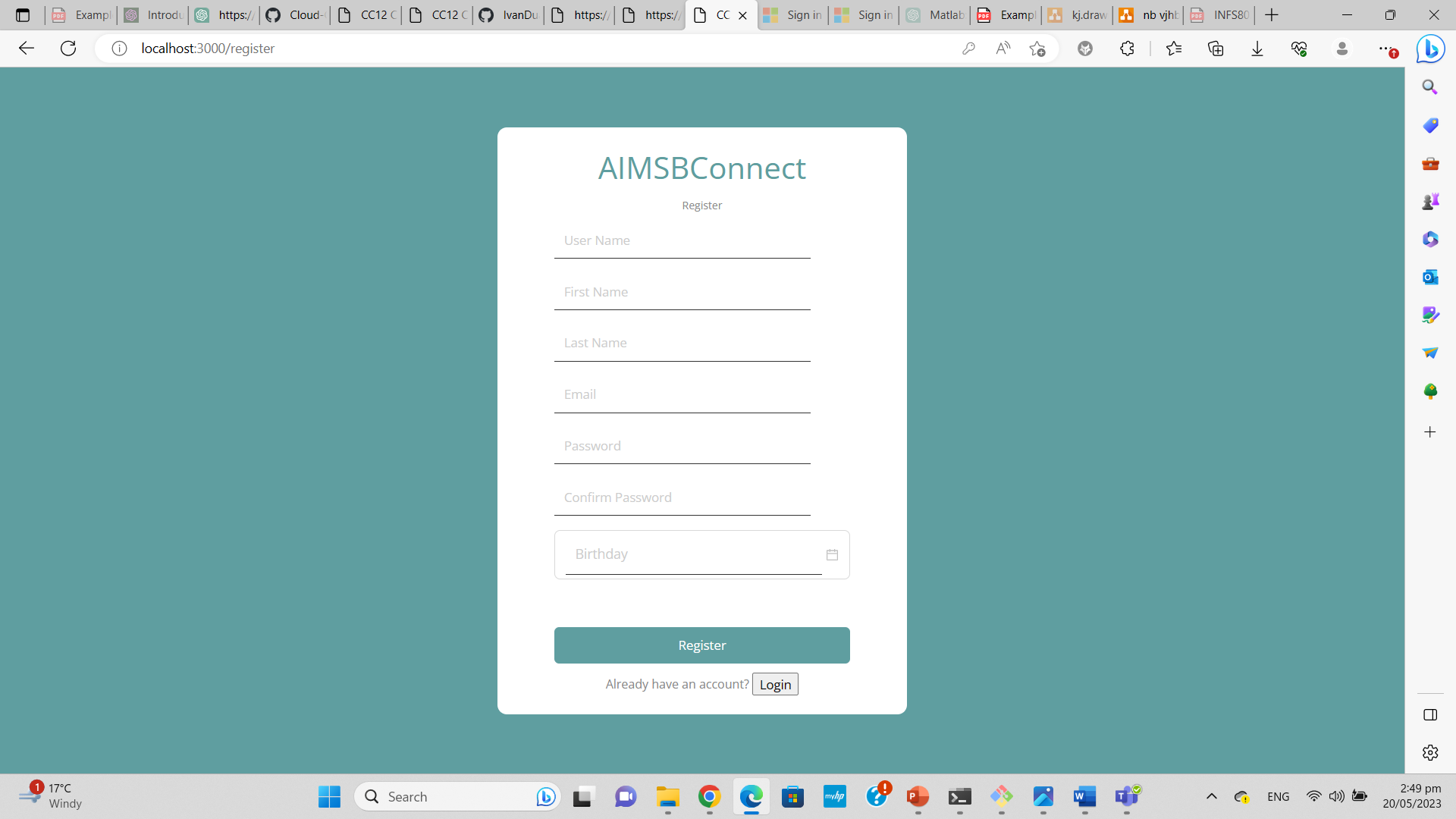


Fig 9 Sign up page of AIMSbConnect

1. Login: (Fig 10)
   * After registering, go to the AIMSbConnect website and click on the "Login" button.
   * Enter your registered email address and password.
   * Click "Login" to access your account.

A screenshot of a computer

Description automatically generated

Fig 10 Login page of AIMSbConnect

1. Dashboard: (Fig 11)
   * Once logged in, you will be directed to the AIMSbConnect dashboard.
   * The dashboard provides an overview of your chats, contacts, and recent activity.
   * You can navigate through different sections of the software using the menu or sidebar.
2. Chatting:
   * To initiate a chat, click on a contact's name from the left panel chat list.
   * A chat window will open, displaying your conversation.
   * Type your message in the input field at the bottom and press “Enter” to send.
   * You can also send files, emojis, or multimedia by using the respective icons in the chat window.
3. Logout:
   * When you're finished using AIMSbConnect, click on the "Logout" button to securely sign out of your account.

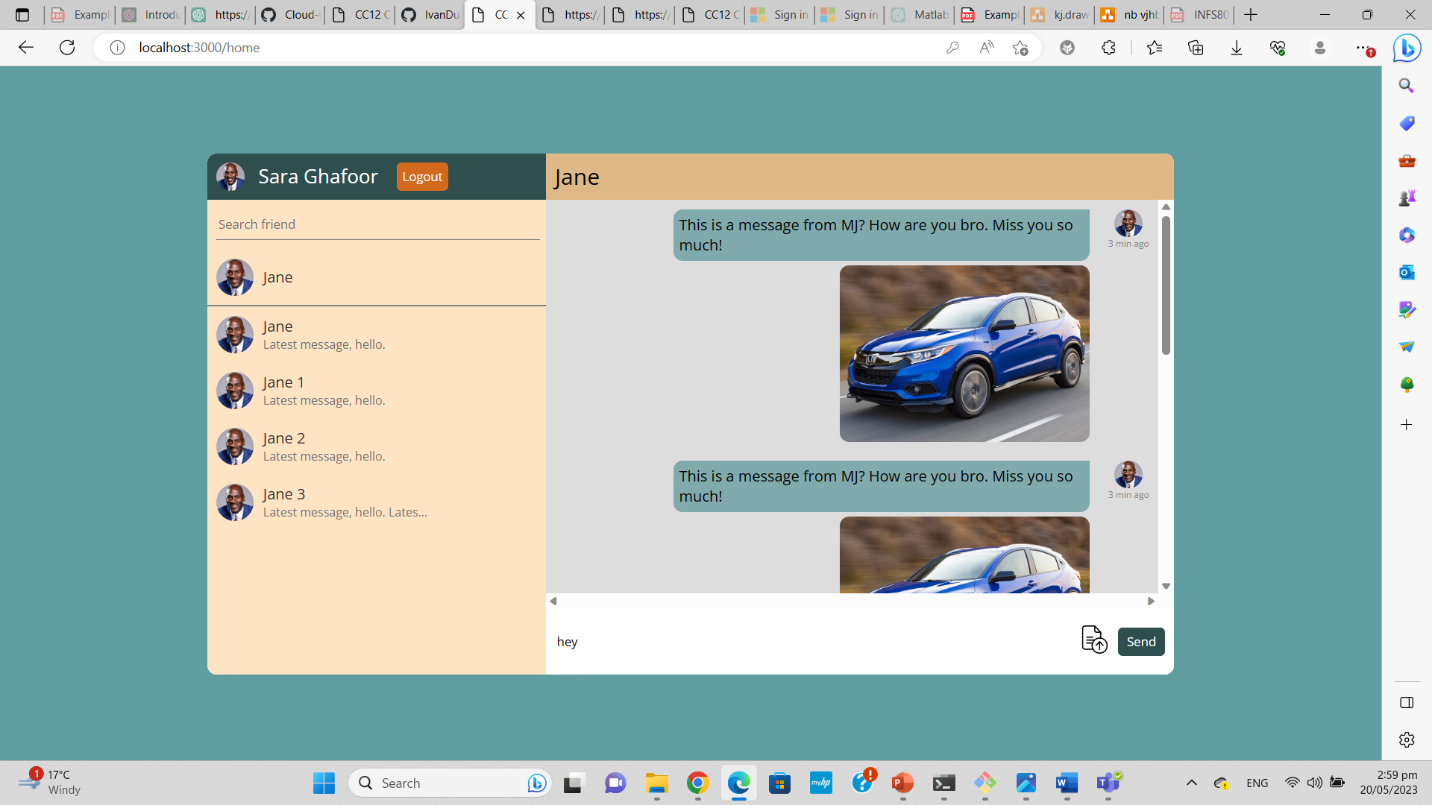


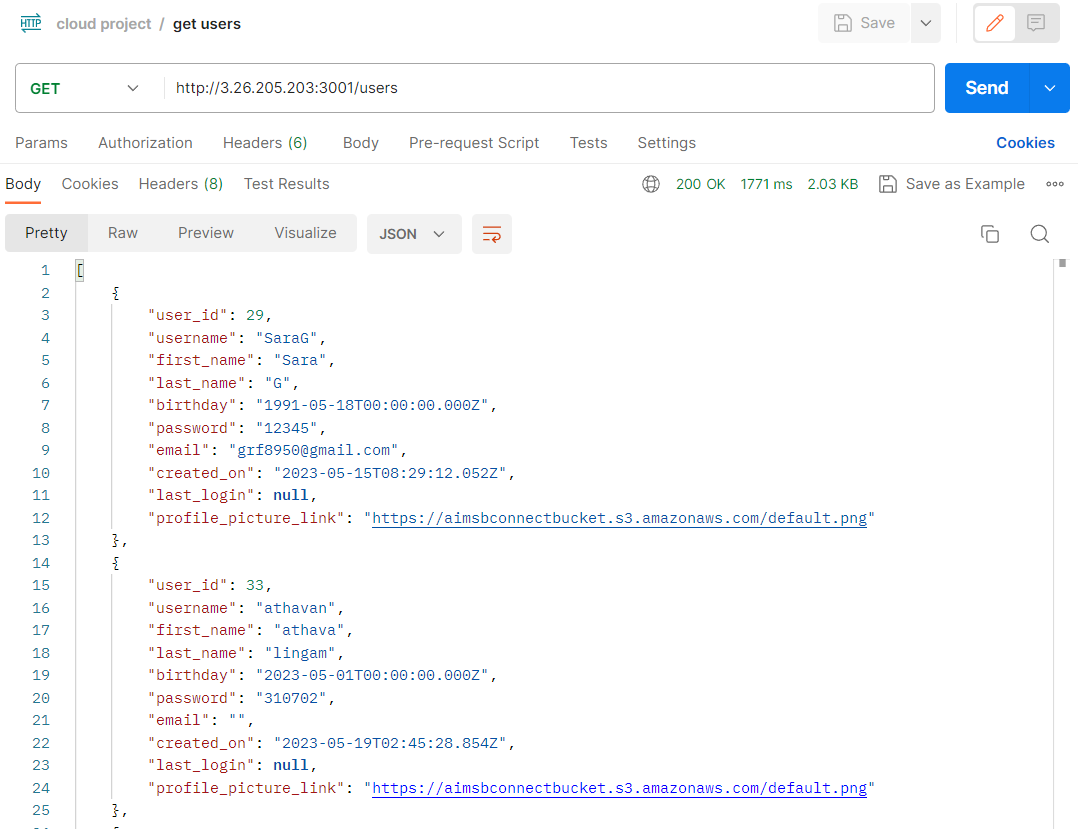
Fig 11 Dashboard of AIMSbConnect

# Work Log

## 8.1 Postman HTTP Client:

Below are the API specifications for the Postman HTTP client, displaying the relative URL, REST method, and description.

|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /users | GET | Get all the users from db |
|  |  |  |

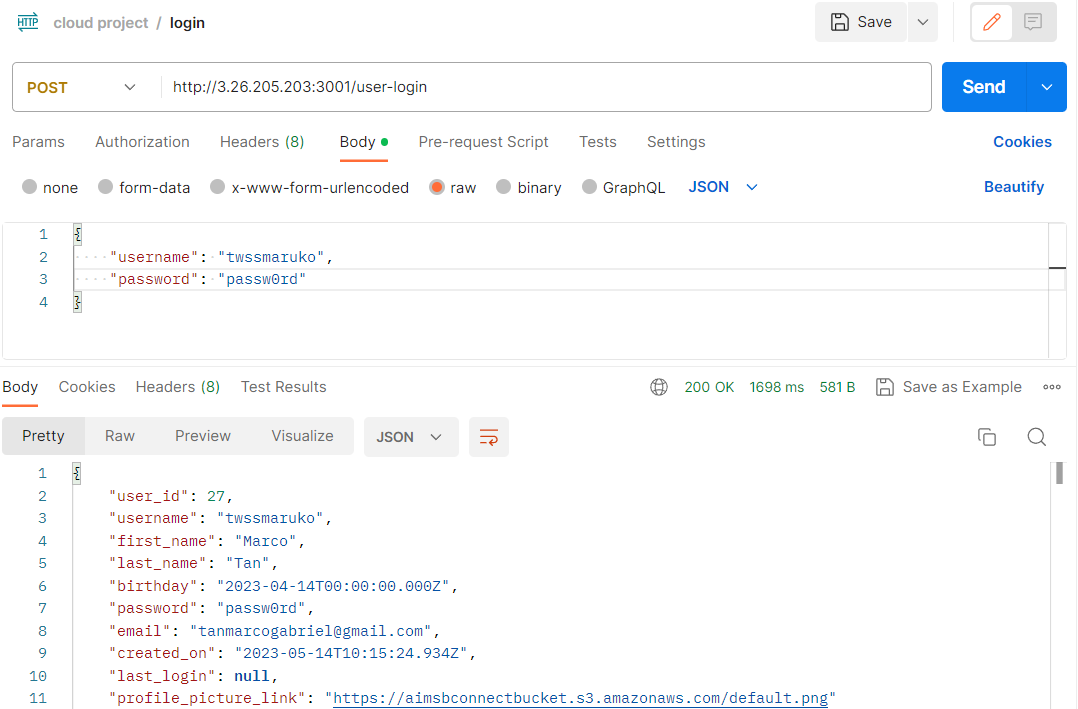


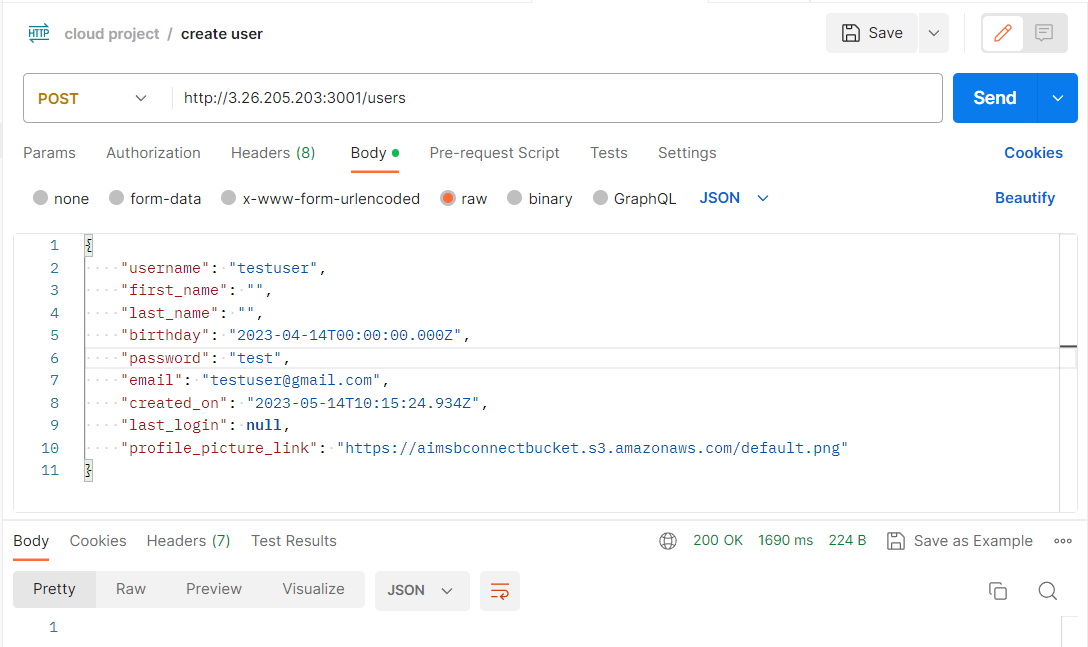
|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /users/:id | GET | Get user by id from db |

A screenshot of a computer

Description automatically generated with medium confidence

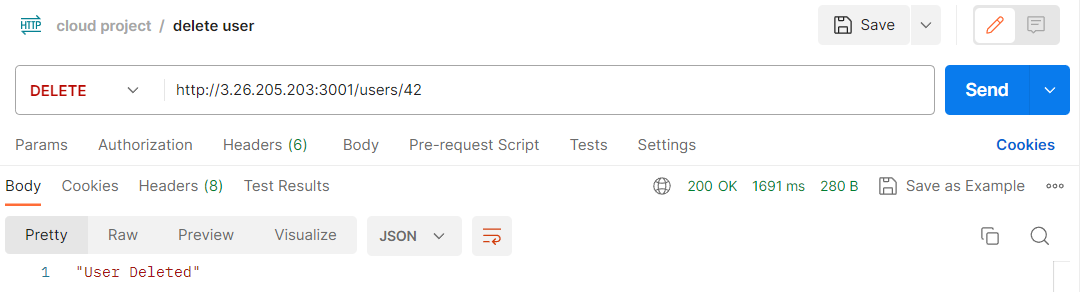
|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /user-login | POST | Validation |





|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /users | POST | Create new user |

|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /users | Delete | Delete User |



# 9 Summary

## 9.1 Major Learnings

## 9.2 Future Work

## 10 URL’s

## 10.1 REST API Methods

|  |  |  |
| --- | --- | --- |
| Relative URL | REST Method | Description |
| /users | GET | Get all the users from db |
| /users/:id | GET | Get user by id from db |
| /user-login | POST | Validation |
| /users/:id | PUT | Update user info by id |
| /users | POST | Create new user |
| /users/:id | DELETE | Delete user by id |
| /iam | GET | Get IAM users |
| /friends | GET | Get all friends |
| /friends | POST | Add friends |

## 10.3 XML Output

[https://aimsbconnectbucket.s3.amazonaws.com](https://aimsbconnectbucket.s3.amazonaws.com/)

This XML file does not appear to have any style information associated with it. The document tree is shown below.

<ListBucketResult xmlns="http://s3.amazonaws.com/doc/2006-03-01/">

<Name>aimsbconnectbucket</Name>

<Prefix/>

<Marker/>

<MaxKeys>1000</MaxKeys>

<IsTruncated>false</IsTruncated>

<Contents>

<Key>2B.png</Key>

<LastModified>2023-05-14T07:46:45.000Z</LastModified>

<ETag>"aaca46309685618a03920ffc6ee823c4"</ETag>

<Size>412061</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>default.png</Key>

<LastModified>2023-05-14T08:45:25.000Z</LastModified>

<ETag>"e13ec751d83a299aaa0f97c58b2e5314"</ETag>

<Size>5997</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>gojo.png</Key>

<LastModified>2023-05-14T08:15:35.000Z</LastModified>

<ETag>"41b14cae7933bfacf6f8086eb5c73a49"</ETag>

<Size>89471</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>name.png</Key>

<LastModified>2023-05-14T08:16:02.000Z</LastModified>

<ETag>"aaca46309685618a03920ffc6ee823c4"</ETag>

<Size>412061</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>tanjiro.png</Key>

<LastModified>2023-05-14T04:50:06.000Z</LastModified>

<ETag>"c21c99f538ad7bad88e25b436abe171c"</ETag>

<Size>323285</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>test\_user.png</Key>

<LastModified>2023-05-15T04:58:45.000Z</LastModified>

<ETag>"aaca46309685618a03920ffc6ee823c4"</ETag>

<Size>412061</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>timothee.png</Key>

<LastModified>2023-05-14T07:45:17.000Z</LastModified>

<ETag>"92371f79aa3914d93918fd8a0a910780"</ETag>

<Size>322620</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

<Contents>

<Key>twssmaruko.png</Key>

<LastModified>2023-05-17T01:56:59.000Z</LastModified>

<ETag>"aaca46309685618a03920ffc6ee823c4"</ETag>

<Size>412061</Size>

<Owner>

<ID>3f148a3d0b63dbf1f18531506ffe2f80bc3f77f4a4f621bcb5cf88d0f91332a4</ID>

<DisplayName>tanmarcogabriel</DisplayName>

</Owner>

<StorageClass>STANDARD</StorageClass>

</Contents>

</ListBucketResult>