Frankfurt University of Applied Sciences

(Master of Engineering)

Information Technology

Course: Agile Development in Cloud Computing Environment

Project: 3b - Access Platform for Providers (APP)

Authors:

Deblina Karmakar (1427965)

Indranil Saha (1427190)

Ankita Talande (1427349)

Nasir Ishaq (1444979)

Pham Tien Son (1318513)

Table of Contents

Abstract

In recent years, the software development industry has been prominently shaped by the adoption of agile methodologies. Agile software development represents a collection of approaches emphasizing iterative development, wherein teams collaboratively craft solutions to problems. This process is facilitated by self-organizing, cross-functional teams. Notably, Scrum and Kanban stand out as two of the most prevalent agile techniques. Our objective is to develop an application using agile methodologies, specifically designed to interact with other applications. This application will access master agreements and available services within the company, facilitating providers in extending bids to agreements and offers to employees, among others.

Keywords: Software Development, Provider Platform, Agile, Sprints, Iterative Methods, Scrum, Kanban.

1. Introduction

"Agile software development" embodies a spectrum of iterative methodologies where requirements and solutions evolve collaboratively within self-organizing, cross-functional teams. Agile software development is a collaborative and iterative approach to software development that prioritizes flexibility, customer feedback, and continuous improvement. Unlike traditional waterfall methodologies, where development progresses linearly through predefined stages, agile embraces change and values customer satisfaction through delivering functional software increments in short cycles, typically referred to as sprints. The Agile Manifesto, established in 2001, outlines key principles such as prioritizing individuals and interactions over processes and tools, responding to change over following a plan, and delivering working software over comprehensive documentation.

In recent times, the adoption of agile methodologies has profoundly influenced the software industry, revolutionizing how teams conceive, plan, and execute software projects. Agile's emphasis on adaptability and collaboration has led to faster development cycles, increased responsiveness to changing requirements, and improved overall project visibility. The iterative nature of agile development allows for continuous feedback from stakeholders, ensuring that the final product aligns closely with customer expectations. Additionally, agile practices such as daily stand-up meetings, frequent testing, and regular retrospectives contribute to enhanced communication, reduced risks, and a more transparent and accountable development process. As organizations continue to prioritize agility in their software development practices, the impact of agile methodologies on the industry is likely to endure and evolve in response to the dynamic nature of modern technology.

Scrum, a widely embraced subset within the Agile methodology, stands out as a popular and nimble framework for software development. What sets Scrum apart are its distinctive roles, artifacts, and time-boxed iterations, offering a well-defined and structured approach that distinguishes it from other Agile methodologies. The framework's emphasis on clear roles, tangible artifacts, and fixed time frames contributes to its efficiency in managing complex projects, promoting collaboration, and ensuring a focused and iterative development process.

Kanban boards, on the other hand, serve as visual workflow tools engineered to streamline work processes and bolster productivity by curbing work-in-progress. This visualization tool enhances transparency, enabling the team to identify and address problematic work phases promptly. By fostering increased visibility, Kanban facilitates prompt solutions, ultimately enhancing team efficiency and effectiveness.

2. Scrum Methodology

In Scrum, three distinct roles — the Scrum Master, the Product Owner, and the Development Team — collectively steer the process. These roles collaborate closely, ensuring seamless information exchange and swift problem resolution among their responsibilities.

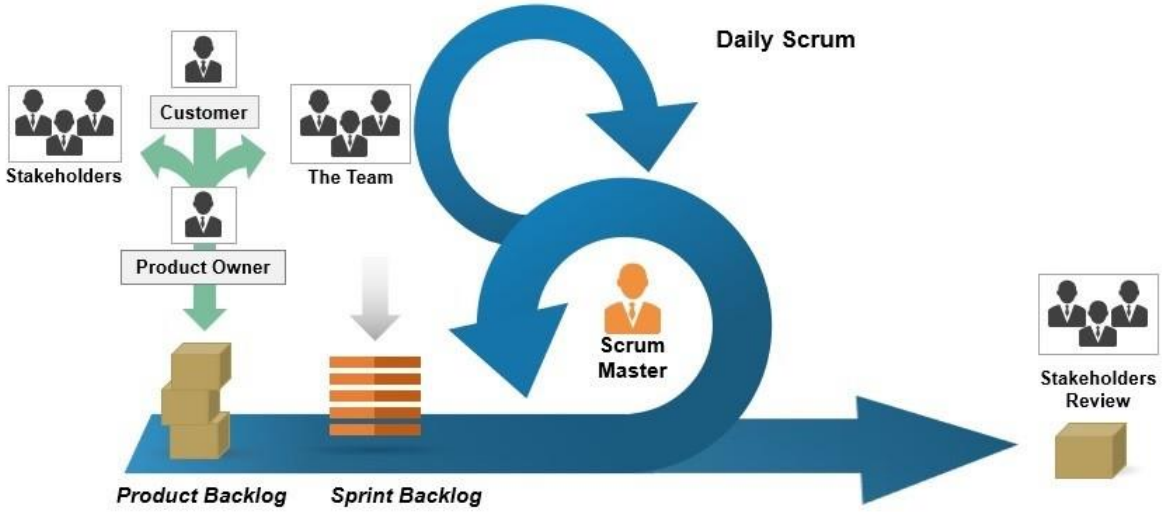


Figure 1: Scrum methodology including all stakeholders

2.1 ScrumMaster

The Scrum Master shoulders multiple responsibilities aimed at streamlining the process, eliminating obstacles hindering productivity, and orchestrating crucial meetings. Encouraging empowerment and innovation among the development team to enhance their quality of work is a pivotal job of Scrum Master. Additionally, improving the team's productivity, maintaining everyone's awareness regarding the team progress, and keeping stakeholders updated fall under the purview of the ScrumMaster. Notably, the ScrumMaster acts as an educator and supporter for other roles, possessing in-depth Scrum knowledge.

The Scrum Master remains vigilant about the project's status, comparing it to anticipated progress, resolving obstacles, and adapting swiftly to emerging challenges. As a liaison between stakeholders, the ScrumMaster shields the team from external interferences, though task allocation remains the responsibility of the team themselves.

2.2 Product Owner

The Product Owner stands as the custodian of criteria and requirements. Serving as the "single source of truth," they delineate requirements and their intended execution sequence for the team. Acting as a bridge between the Team and business stakeholders, clients, and their product-related demands, the Product Owner manages inquiries about product requirements. They collaborate closely with the team to articulate user-facing and technical needs, document requirements, determine their implementation sequence, and curate an up-to-date and detailed Product Backlog. In addition to it, the Product Owner

assesses the readiness of implementations for release, ensuring the requisite functionality and quality, while also setting release timelines for finished work.

2.3 Development Team

Comprising self-organizing, cross-functional individuals, the development team undertakes the hands-on creation and testing of the product. Empowered to decide how to execute tasks, the team is responsible for delivering the final output. Hence, the team self-organizes, determining task allocation and division during Sprints. Ideally, the team size ranges between five to ten members to ensure optimal collaboration and productivity.

2.4 Sprint

Sprints define scrum's iterative work periods, fostering consistency and swift feedback cycles. The shorter duration facilitates continual inspection and adaptation, crucial for efficient workflow management and reducing risks associated with longer cycles. Sprints follow a sequential pattern, immediately commencing after the conclusion of the preceding Sprint, maintaining a cadence for frequent iterations and feedback.

2.4.1 Sprint Planning

The project plan has been divided into 5 sprints with 2 weeks of sprint intervals as dated below:

AGILE PLANING					
10.11.2023	24.11.2023	08.12.2023	22.12.2023	12.01.2024	28.1.2024
Sprint planning (Sprint 1)	Review and retrospective (Sprint 1) Sprint planning (Sprint 2)	Review and retrospective (Sprint 2) Sprint planning (Sprint 3)	Review and retrospective (Sprint 3) Sprint planning (Sprint 4)	Review and retrospective (Sprint 4) Sprint planning (Sprint 5)	Submission

Table1: Agile Planing

Sprint 1: First major task in the sprint is architecture design and pattern discussion. Setup of communication channels to facilitate cross-team discussions.

Sprint 2: Implementation of navigation bar, design of front-end screens, including login, employee, master agreement, agreement bids, open service, and logout.

Sprint 3: Sharing of APIs from group 1, 2 to group 3 and fetching of the same by group 3 to incorporate it in codebase.

Sprint 4: Sharing of APIs from group 4 to group 3 for providing the open service requests and incorporating the same by group 3, updating documentation based on the same.

Sprint 5: Generation and sharing of API from group 3 to group 2 for providing bids to the master agreement, from group 3 to group 4 for providing offers to open service request, intensive testing of the functionalities, updating report based on the results, updating presentation for better readability, and understanding.

2.4.2 Best Practices

After in-depth discussions between team members the decision has been made to adopt a sprint interval of 2 weeks. This choice, driven by the advantages of managing smaller stories with limited requests, has fostered the delivery of small, achievable goals, allowing for increased iteration flexibility as needed.

Key outcomes included brief daily sprint meetings, two dedicated Product Backlog Refinement (PBR) sessions involving the Product Owner (PO), Backlog Owner (BO), and Team Members (TM). Additionally, a Team Demo was conducted to ensure that completed stories not only aligned with the initial requirements but also satisfied the PO, facilitating code review within the team. Sprint Reviews were crucial to ensure proper closure of all stories and addressing any potential spillover stories.

Distinct responsibilities were assigned, encompassing the maintenance of sprint velocity and the management of appropriate backlog stories. The calculation of story points (SP) was collaboratively performed by the team during PBR sessions. The 2 weeks were further broken down as following:

Day 1 (Friday)	Day 2 (Saturday)	Day 3 (Sunday)	Day 4 (Monday)	Day 5 (Tuesday)	Day 6 (Wednesday)	Day 7 (Thursday)
Sprint Review	Product Backlog Refinement	Internal Group Meeting	-	Kanban Board update	Development work	Team Retrospective of sprint
Day 8 (Friday)	Day 9 (Saturday)	Day 10 (Sunday)	Day 11 (Monday)	Day 12 (Tuesday)	Day 13 (Wednesday)	Day 14 (Thursday)
-	Product Backlog Refinement Session2	Team Retrospective ofSprint	-	Development work	Meeting with other groups	Team Demo

Table2: Sprint Planning

2.4.3 Push and Pull Techniques

Implementing a pair programming strategy involving two team members provided the team with a distinctive application of the push and pull principle. Every Monday within the sprint was designated as a deployment day, where the code was introduced to the User Acceptance Testing (UAT) Environment and subjected to a single round of testing, deviating from the conventional Continuous Integration and Continuous Development (CI/CD) pipeline. Following the sprint review, if everything proceeded as intended, the code was then pushed to the Production (PROD) Environment. In an effort to prevent merge conflicts, the responsibility of deployment rested with one team member, while others focused on testing and providing feedback. Despite adhering to traditional practices, this approach maintained a noteworthy level of flexibility.

2.4.4 Collaboration

Cross-functional collaboration is actively promoted through dedicated sessions where a representative from each team is encouraged to participate. These sessions serve as a platform for teams to disseminate comprehensive updates on their ongoing work, identify potential obstacles, and engage in collaborative problem-solving. One prominent example of such interactive meetings is the Scrum session, strategically employed to enhance collaboration among teams. Through these sessions, teams synchronize their efforts, align goals, and foster a collective understanding of the project's progress. This proactive approach not only facilitates efficient communication but also creates an environment where teams can offer and receive assistance, promoting a culture of mutual support and shared responsibility.

In addition to interactive meetings, the use of Kanban boards plays a pivotal role in facilitating collaborative planning and visualizing progress across teams. These boards act as visual aids that allow teams to collectively plan their work, track ongoing tasks, and maintain focus on individual and shared goals. By providing a real-time snapshot of workflow status, Kanban boards empower teams to streamline processes, identify bottlenecks, and make informed decisions to optimize overall project performance. The combination of interactive sessions and visual tools like Kanban boards contributes to a collaborative and transparent working environment, fostering effective cross-team coordination and alignment.

2.4.5 Retrospective

Following the conclusion of Sprint #1, the team exhibited adaptability by strategically selecting a Story Point (SP) allocation that optimized the full utilization of both Team Members' bandwidth. This decision emerged from a comprehensive postmortem analysis conducted after the initial sprint, where the team critically evaluated their performance and identified areas for improvement. A key takeaway from this reflective discussion was the establishment and clarification of both the Definition of Done (DOD) and Definition of Readiness (DOR), ensuring a shared understanding of task expectations and prerequisites, thereby enhancing overall team cohesion and project efficiency.

One notable outcome of this continuous improvement initiative was the team's commitment to refining operational practices. Traditionally, retrospectives were a routine held on the second Friday of each sprint. However, the postmortem analysis demonstrated the team's proactive approach to adjusting their processes. This commitment is a testament to their dedication to iterative feedback, showcasing their willingness to adapt and optimize collaboration for improved performance.

In addition to the flexible SP allocation, the team's dedication to continuous improvement was further exemplified through the regular scheduling of retrospectives. By ingraining these reflective sessions into their routine, the team fostered a culture of learning and adaptability, reinforcing their commitment to refining practices and achieving optimal outcomes in subsequent sprints.

3. Kanban Board

Teams can effectively visualize and control their operations using Kanban boards, employing cards to represent tasks visually. Each Kanban card includes comprehensive details about the task, such as the due date, assigned personnel, and a description of the work in progress.

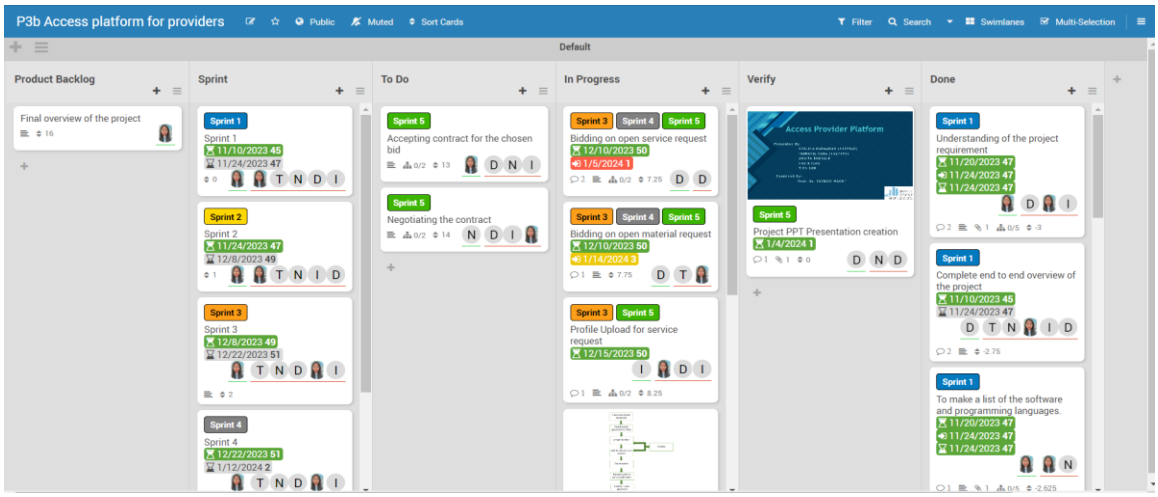


Figure 2: Kanban Board

A product backlog is a dynamic, prioritized list of tasks, features, and enhancements that a development team plans to work on. It evolves over time, reflecting changing priorities and business needs. The backlog helps to prioritize work, estimate effort, and provides visibility into upcoming tasks for efficient product development.

Task Name: Clarity in task labels is crucial for understanding the nature of the work. Tasks that begin with a verb, like "Add functionality to web app," provide a high-level overview of the required work.

Key Dates: Depending on the team's process, certain dates may hold significance for both project managers and developers. For example, a project manager might need the task's start date, while a developer would require the task's deadline.

Task Owner: The designated task owner is responsible for ensuring that the job is completed. Any queries regarding the task should be directed to this individual.

Task Status: In the Kanban system, task status is often indicated by the card's location on the Kanban board. It is essential for everyone to be familiar with the various stages of the team's Kanban board, as each team may have a distinct process for tracking work in progress.

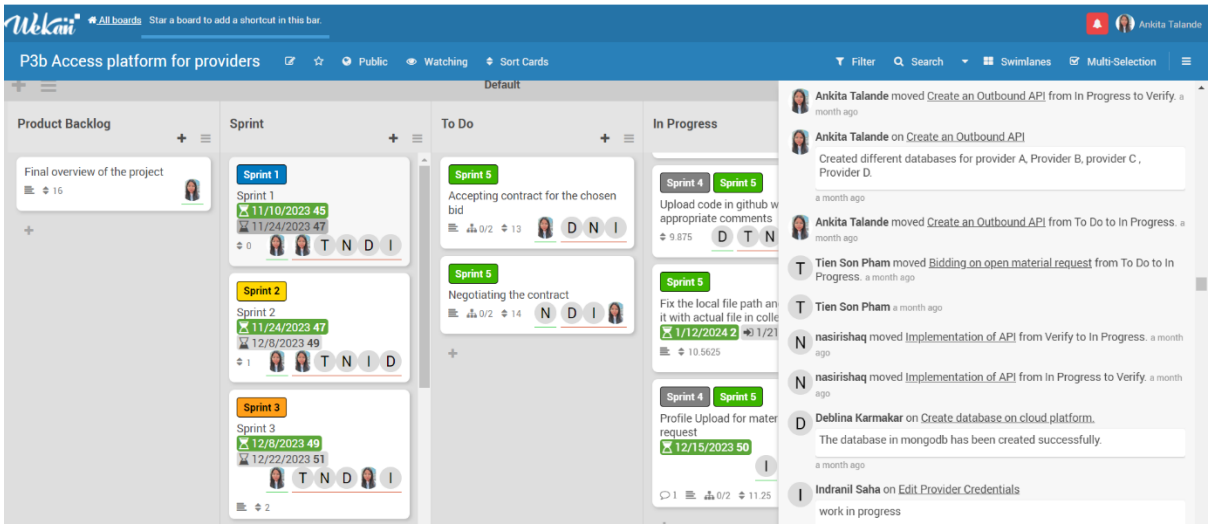


Figure 3: Task Status in Kanban Board

Task Priority: Illustrating the significance of an activity in relation to others is crucial. Setting priorities can help maximize lead time for tasks that demand it, assisting the development team in organizing their work effectively.

Subtasks: Some tasks may be sufficiently labour-intensive to warrant division into smaller subtasks. These subtasks can be linked to a parent task, contributing to a more organized workflow.

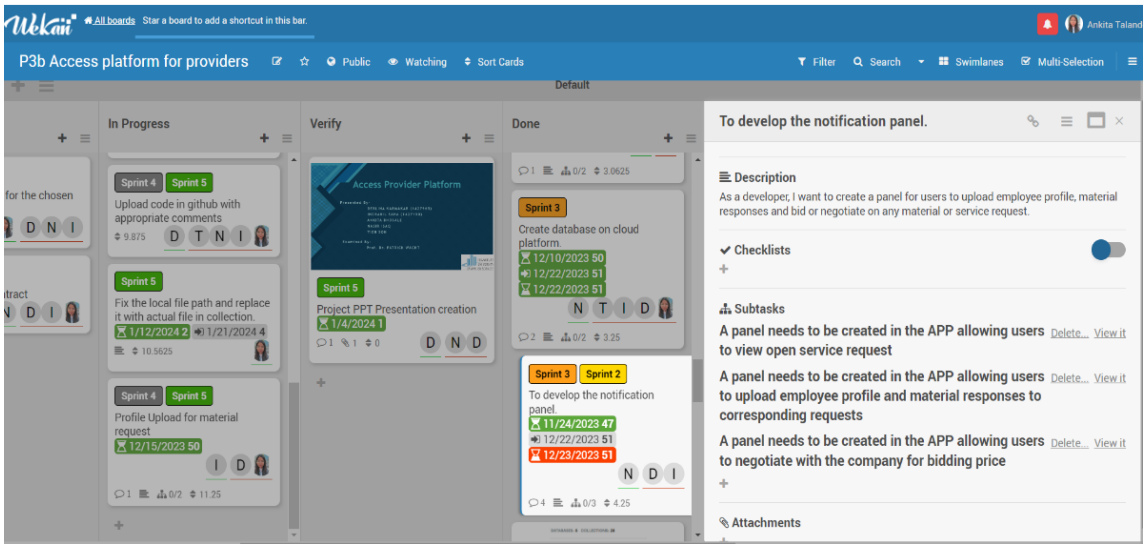


Figure 4: Subtask in Kanban Board

4. Requirement

Below is the list of general requirements provided by the Product Owner:

- User wants to login using authentication component.
- Provider Admin wants to edit provider credentials (name).
- Provider Admin wants to configure user management for provider (register new user, deregister user).
- User wants to see the offers of the company “future-X” regarding master agreement types (example: technology-based).
- User wants to provide an offer to establish a master agreement.
- User wants to bid on open service requests and material request.
- User wants to upload profiles of employees for service requests and responses to material requests.
- User wants to make suggestions based on the knowledge of each offered employee.
- User accepts contract if his offer has been chosen. Negotiation can take place.
- Further specification takes place during collaboration with the customer.
- APIs need to be provided to other groups.

At the beginning of sprint 1, all the requirements have been provided by the product owner. The development team, along with the Scrum Master has been notified of the respective responsibilities. Each and every requirement has been discussed in detail and the expectation from the product owner has been conveyed in depth to all the involved stakeholders. The development team, in the following sprint, discussed the feasibility of the requirements and presented the roadmap in scrum call to the product owner before the start of the developmental activity.

As listed in the requirement, the focal point of activity for group 3 has been the development of the website, Access Platform for Providers, where users of the provider shall be able to add employees, accept or reject the master agreement, provide bids to the master agreement, and offers to the open service request. Due to these objectives, group 3 has worked closely with other groups, especially with group 2 and group 4 to realise the goals.

5. Software Architecture

For the project activity, we utilized Node.js as the backend for our provider platform application, and for crafting the user interface, we employed JavaScript along with Pug CSS. The following table outlines the packages and their respective versions employed in our Node.js environment. To install these packages, we executed the 'npm install' command, and to launch the application in the terminal, we initiated the 'npm start' command.

Name	Version
Bcryptjs	**^2.4.3**
connect-flash	**^0.1.1**
cookie-parser	**~1.4.4**
Cors	**^2.8.5**
Debug	**~2.6.9**
Express	**~4.16.1**
express-ejs-layouts	**^2.5.1**
express-session	**^1.17.3**
express-validator	**^6.14.2**
http-errors	**~1.6.3**
Mongoose	**^6.8.1**
Morgan	**~1.9.1**
Multer	**^1.4.5-lts.1**
Passport	**^0.6.0**
passport-local	**^1.0.0**
Pug	**2.0.0-beta11**
node-fetch	**^2.6.7**
serve-favicon	**^2.5.0**

Table 3: Packages list

6. Database Model

Mongodb Atlas is being utilized for the database part in this project. A separate database is maintained for each provider. The collections available for each provider include Agreements, Bids, Employees, Offers, OpenServiceRequests, and Users.

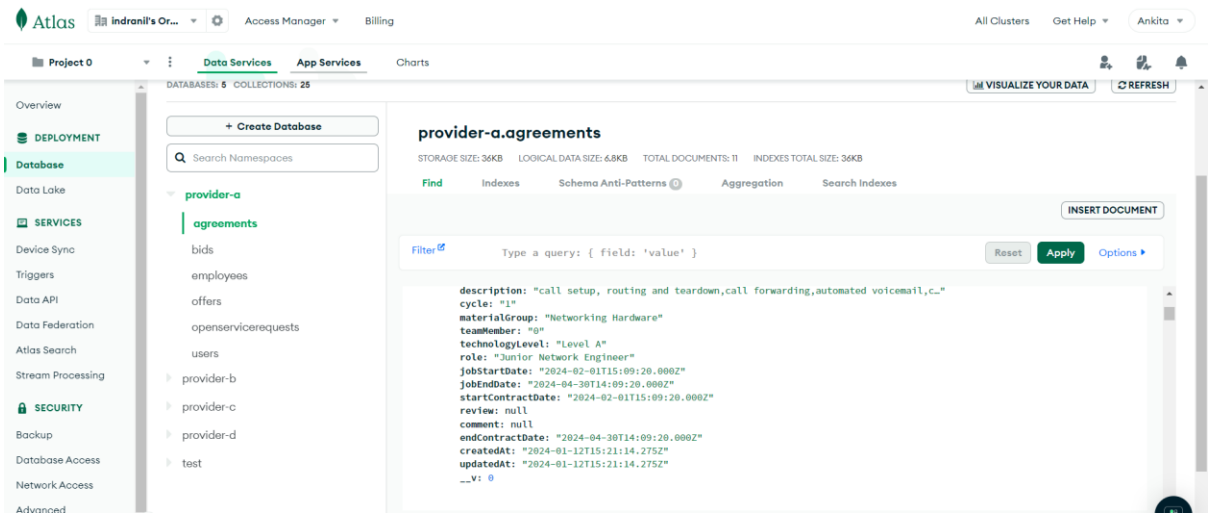


Figure 5: Database and collection snapshot

7. Use case Diagram

This diagram illustrates the use case for the provider in phase 1. Following registration, the provider can log in. Once logged in, the provider can modify personal details and access the master agreement provided by the company. The provider can request daily rates and has the option to accept or reject master agreements. After accepting, the provider can then bid for active master agreements.

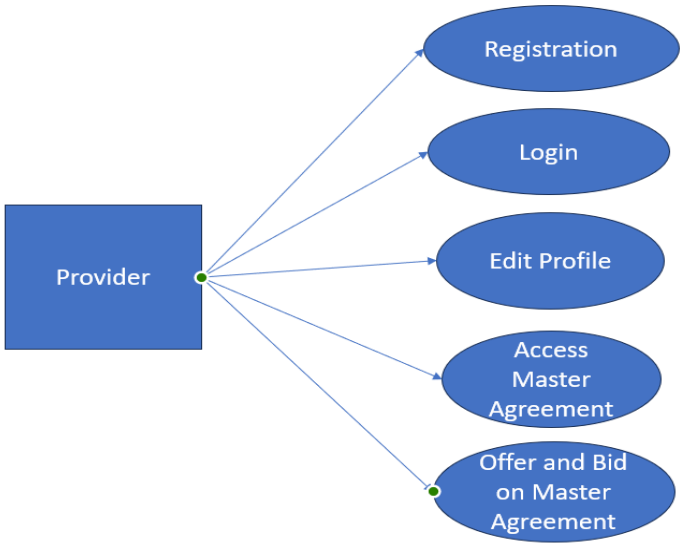


Figure 6: Use case diagram for the provider in phase 1

This use case diagram represents the provider's functionalities in phase 2. During this phase, the provider can access open service requests from the company, aligning with the accepted master agreements. The provider is equipped with the capability to create employees and extend offers to them based on the existing open requests.

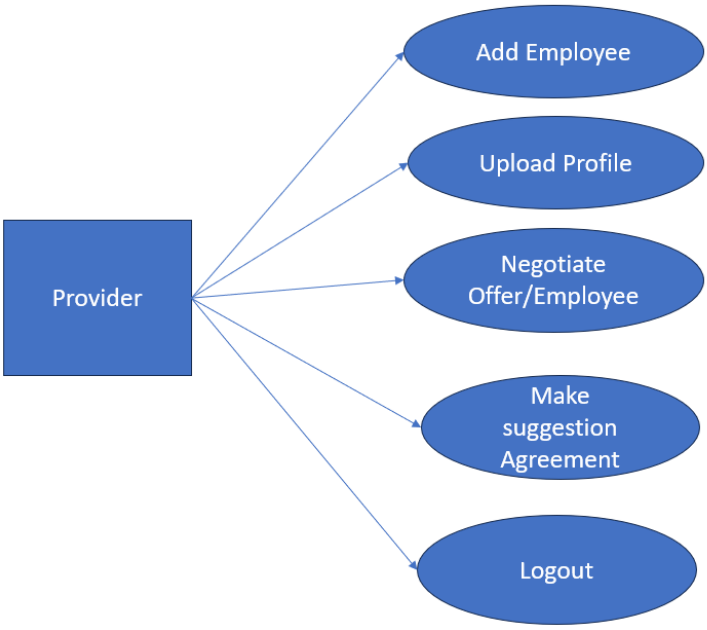


Figure 7: Use case diagram for the provider in phase 2

8. Workflow

The project is structured around two key phases. In the initial phase, the company is dedicated to generating and publishing new master agreements. It is to be noted that master agreements can be published in several quantities, depending on the domain and role. Providers have the privilege of accessing these master agreements, enabling them to make decisions by either accepting or rejecting the offers. Within this phase, providers can also define parameters for salary limits, and the master agreements automatically conclude after a specified time interval.

Moving into the second phase of the project, the company focuses on creating service requests for employees to be fulfilled by providers. Following the service evaluation, providers extend offers to employees, and a negotiation process ensues.

9. API Implementation

In compliance with the project requirement, there shall be 2 sets of APIs from the perspective of group 3 – Inbound API, Outbound API. Group 3 shall receive the inbound APIs from group 2 and group 4, whereas group 3 shall provide the outbound APIs to both the other groups.

9.1 API from others group

Below are the inbound APIs from group 2 and group 4 as provided by the respective teams –

Group 1 API - https://codexauth.onrender.com/api/register/

Group 2b API - https://dg4gi3uw0m2xs.cloudfront.net/agreement

Group 4b API - http://13.48.42.106:8000/request-details/

9.2 Group 3b API

Below are the outbound APIs from group 3 to the other groups of the project -

Offers to Open Service Request -

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/offers?provider=A

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/offers?provider=B

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/offers?provider=C

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/offers?provider=D

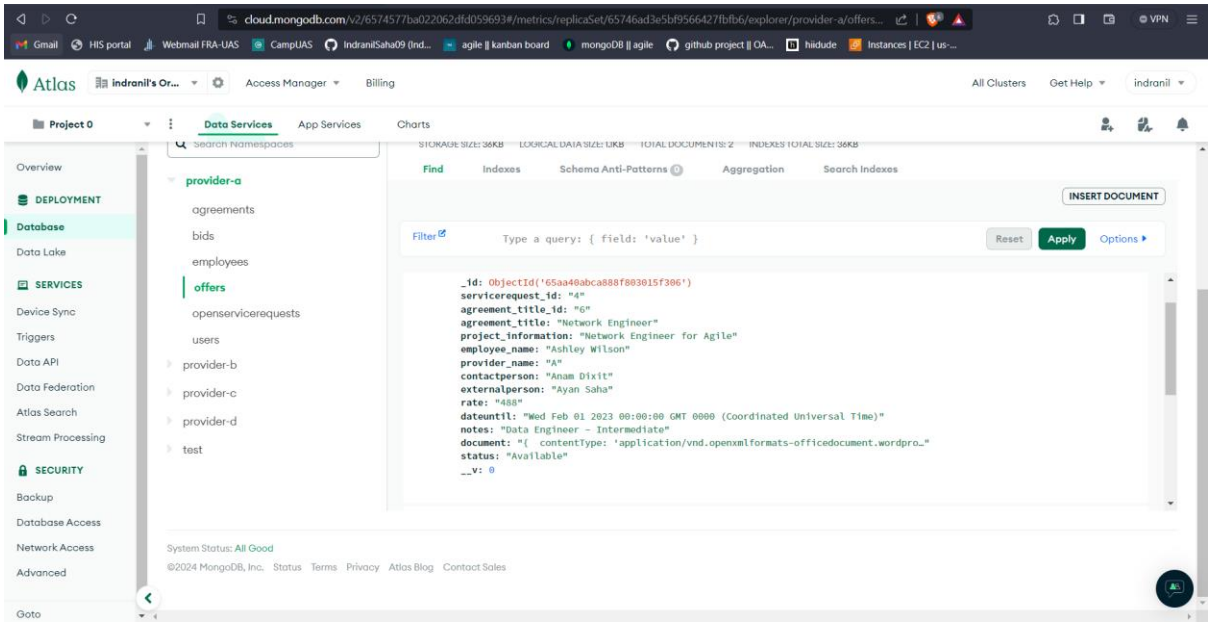


Figure 8: Offers of the Provider a

Master Agreement Bids –

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/Agreementbids?provider=A

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/Agreementbids?provider=B

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/Agreementbids?provider=C

http://ec2-52-90-1-48.compute-1.amazonaws.com:4000/users/Agreementbids?provider=D

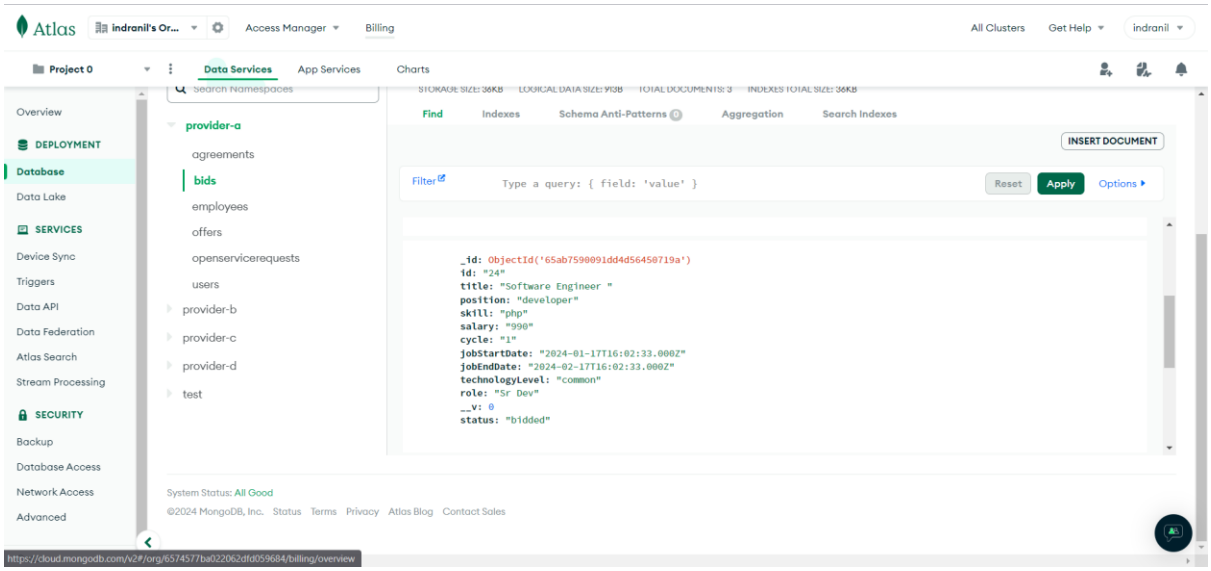


Figure 9: Bids of provider A to master agreement

9.3. Output

In this section, the application Access Platform for Providers has been run and the result is being shown below.

To run the application – npm install, npm start

To run the application locally - http://localhost:4000/

To run application in cloud – host it in cloud platform and run it

9.3.1 Screen: Login

Login page –

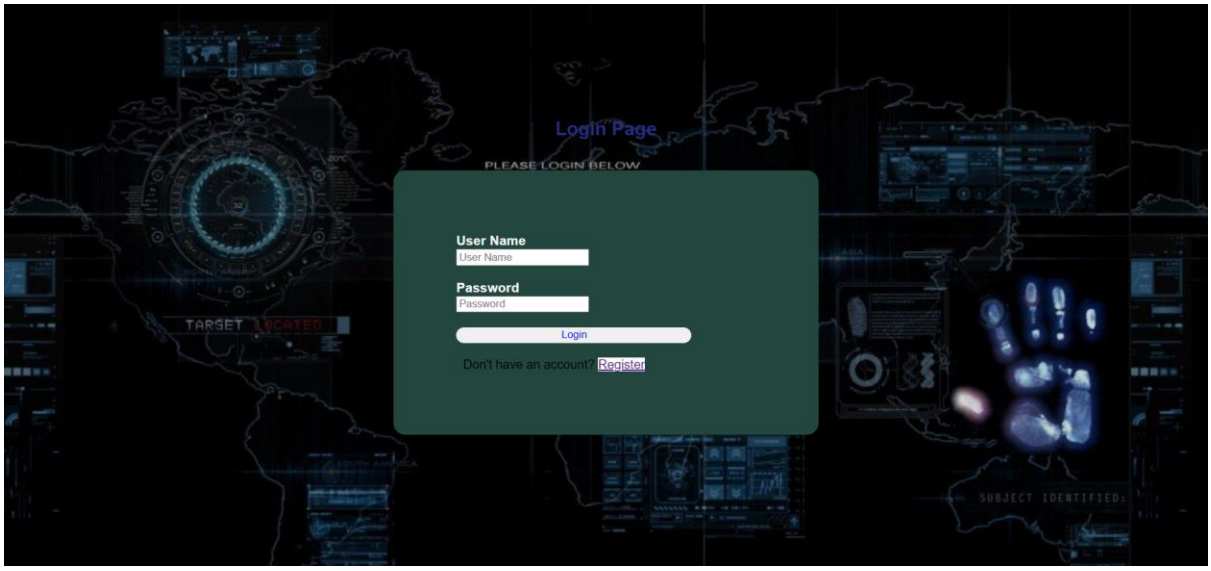


Figure 10: Login page

Enter proper credentials to login.

Logging in as user of provider-d

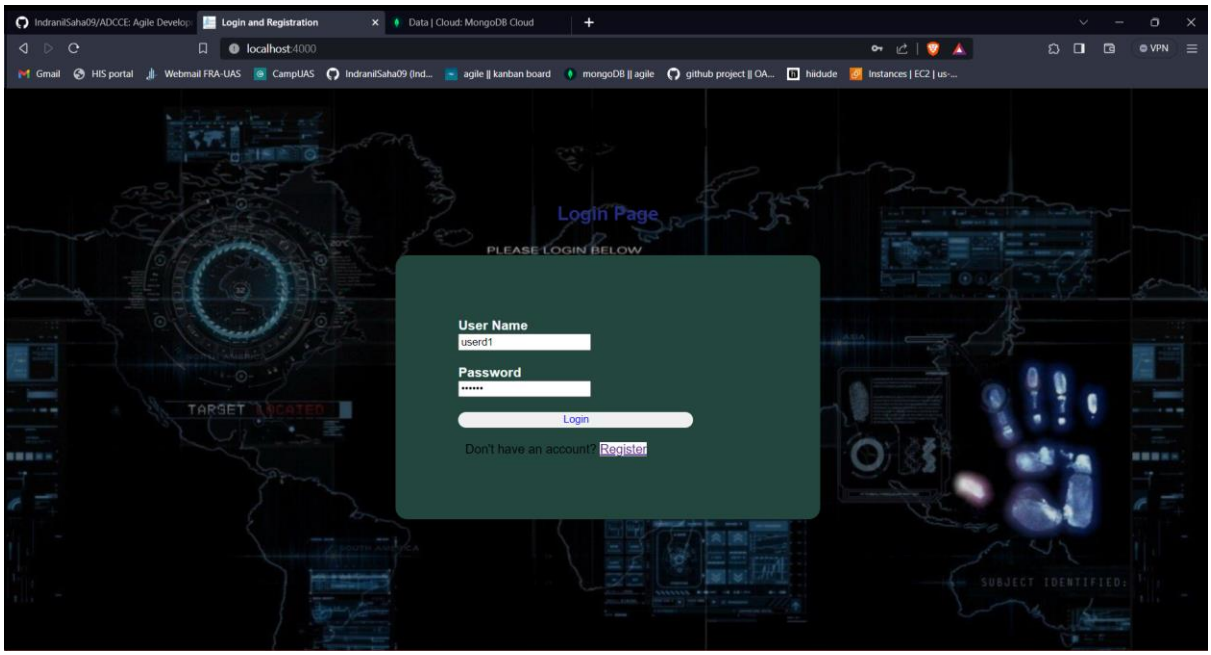


Figure 11: Logging in as user of provider-d

Home page -

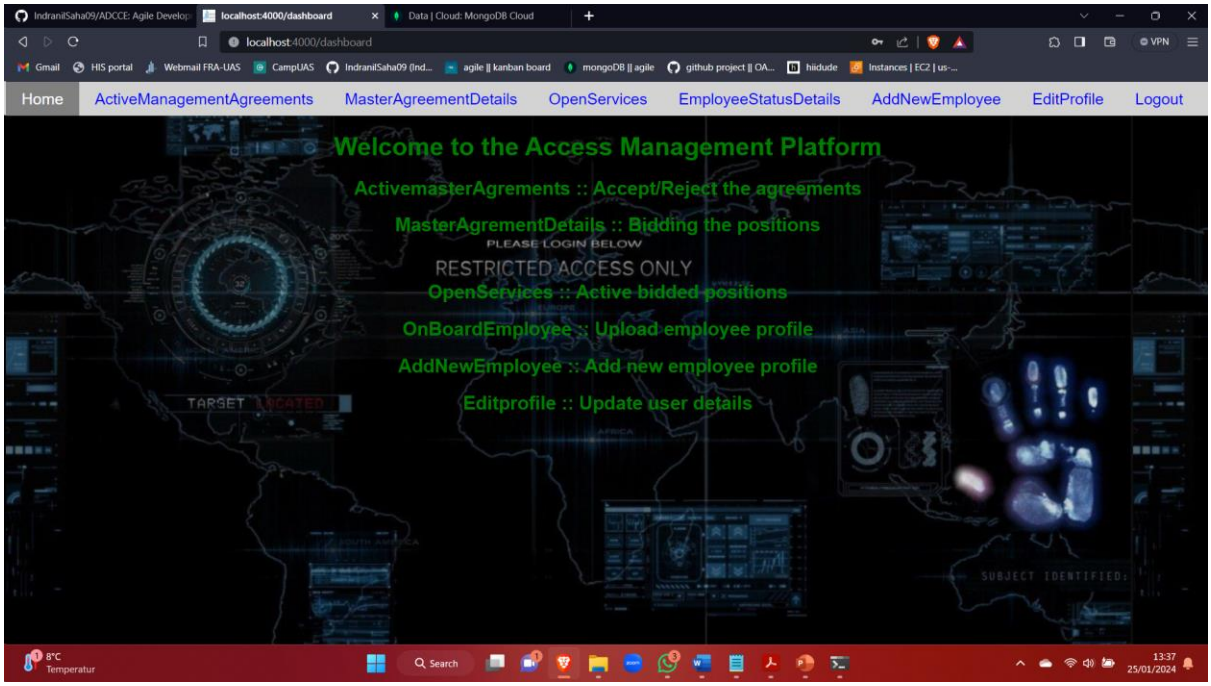


Figure 12: Home page

9.3.2 Screen: Edit Profile

To edit user profile - click on Edit Profile in dashboard and Changing phone number .

Before change

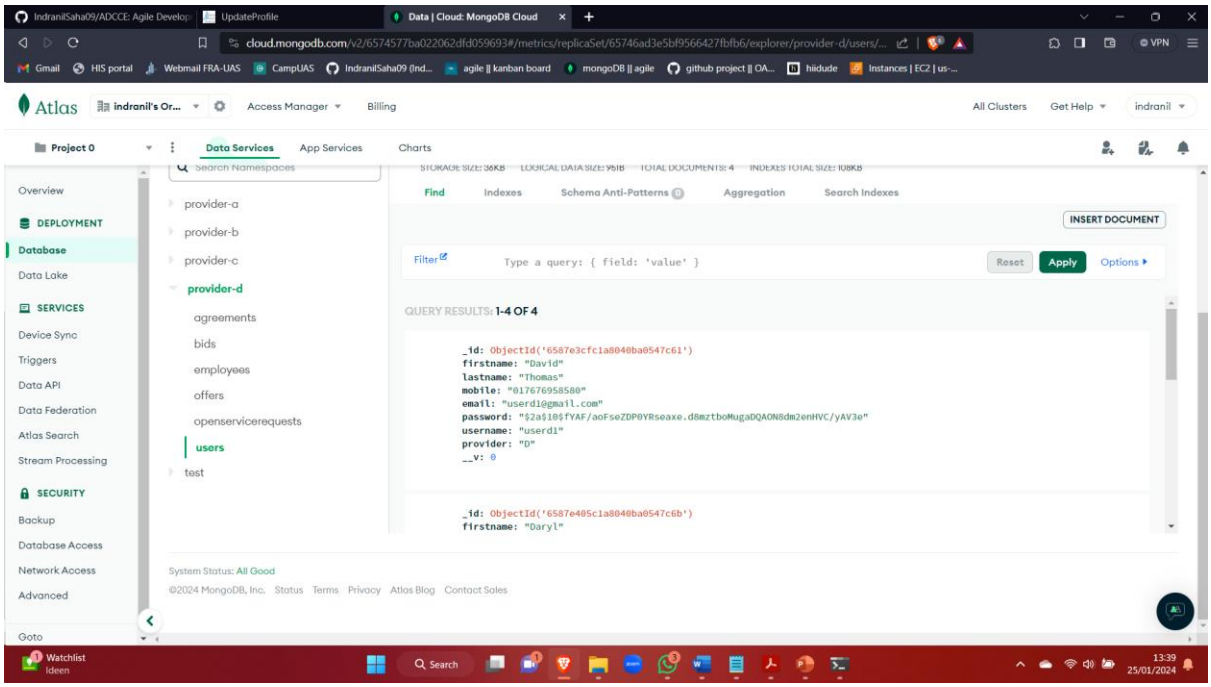


Figure 13: edit user profile before change

Changed mobile number field, enter username and password and click on update

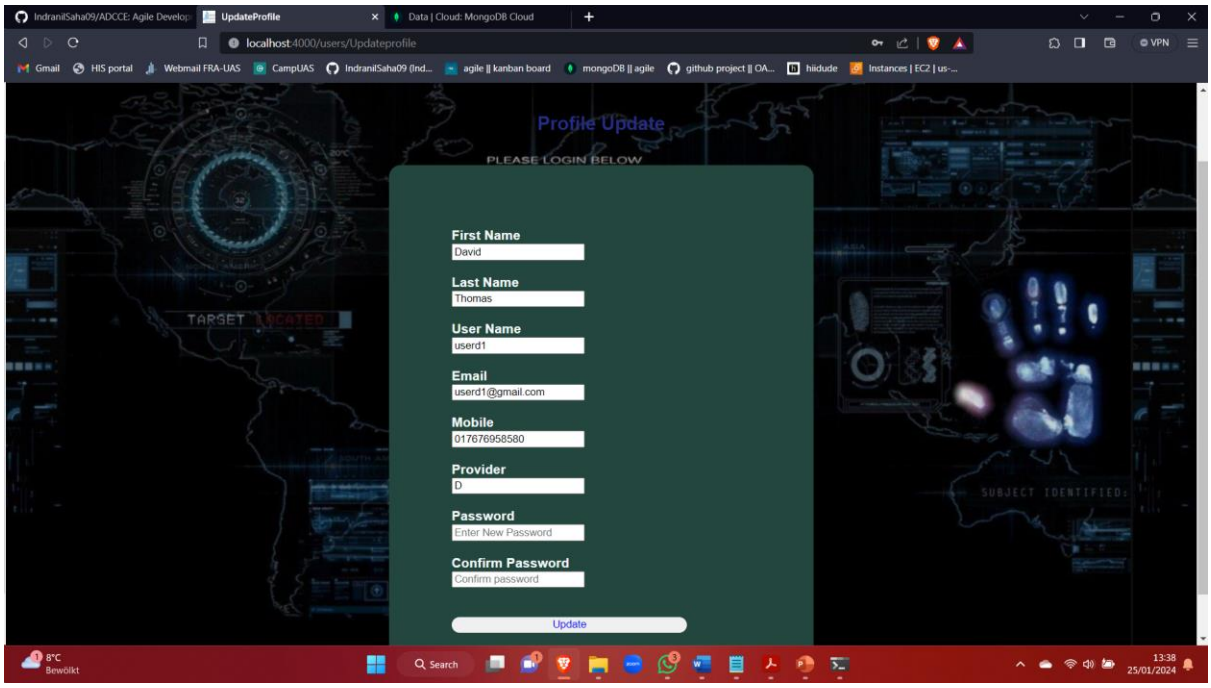


Figure 14: edit user profile

Entering correct password and clicking on update

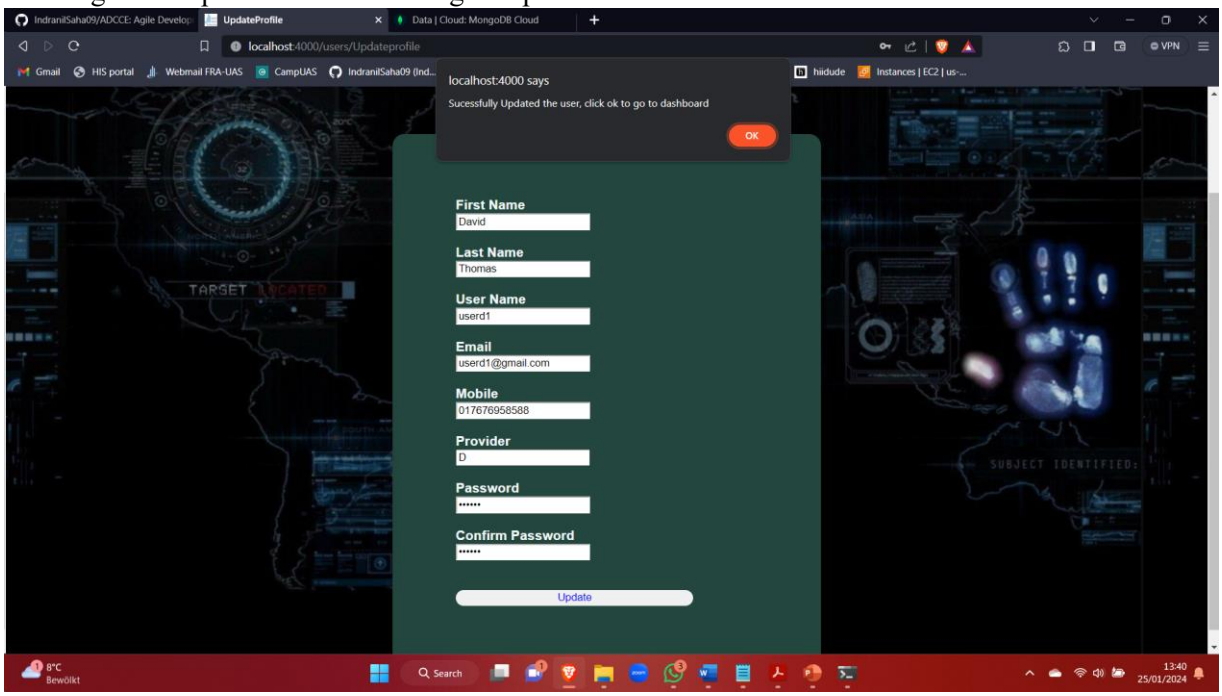


Figure 15: edit user profile - updated

After change,

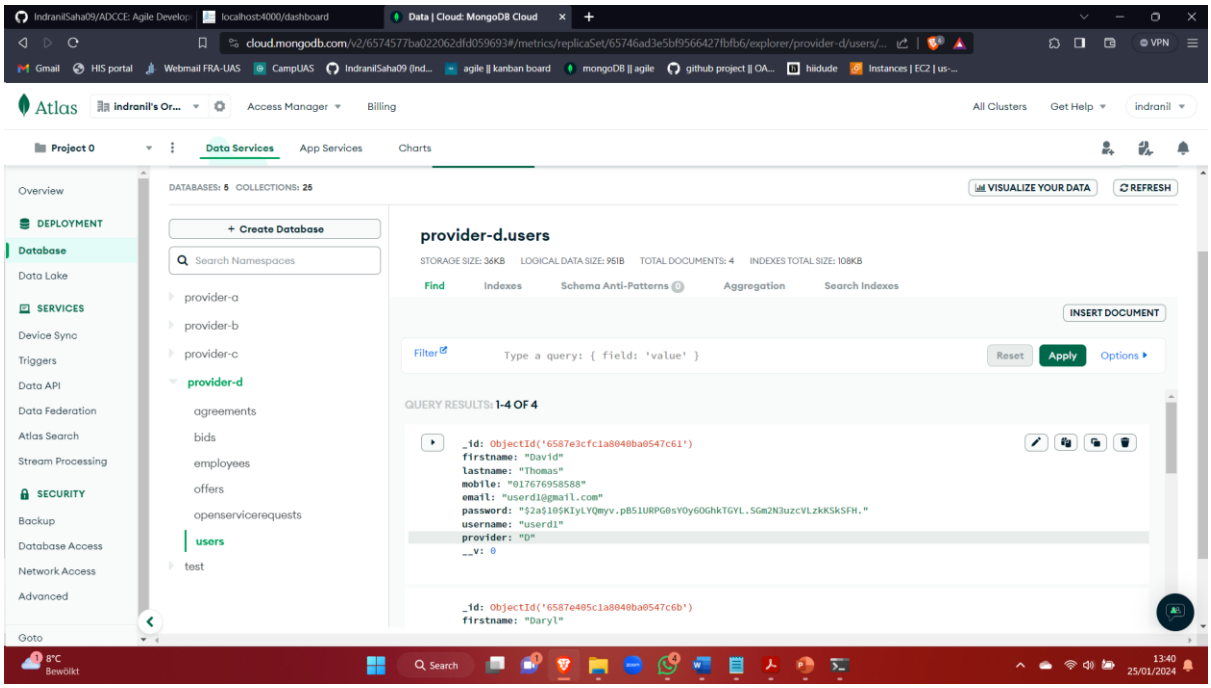


Figure 16: edit user profile -Database updated

9.3.3 Screen: Master agreement

Click on ActiveManagementAgreements to get the master agreements published from group 2b.

Access Management Platform

Agreement

ID	Agreement Name	Details
6	Network Engineer	Full Agreement Details
8	Software Engineer	Full Agreement Details
7	Network Engineer	Full Agreement Details
9	Software Engineer	Full Agreement Details
18	string	Full Agreement Details
17	string	Full Agreement Details
16	test 2	Full Agreement Details
19	string	Full Agreement Details
20	string	Full Agreement Details
11	Front End Dev	Full Agreement Details
10	test	Full Agreement Details

Figure 17: Master agreements published from group 2b

Click on Full Agreement Details to view the total agreement details

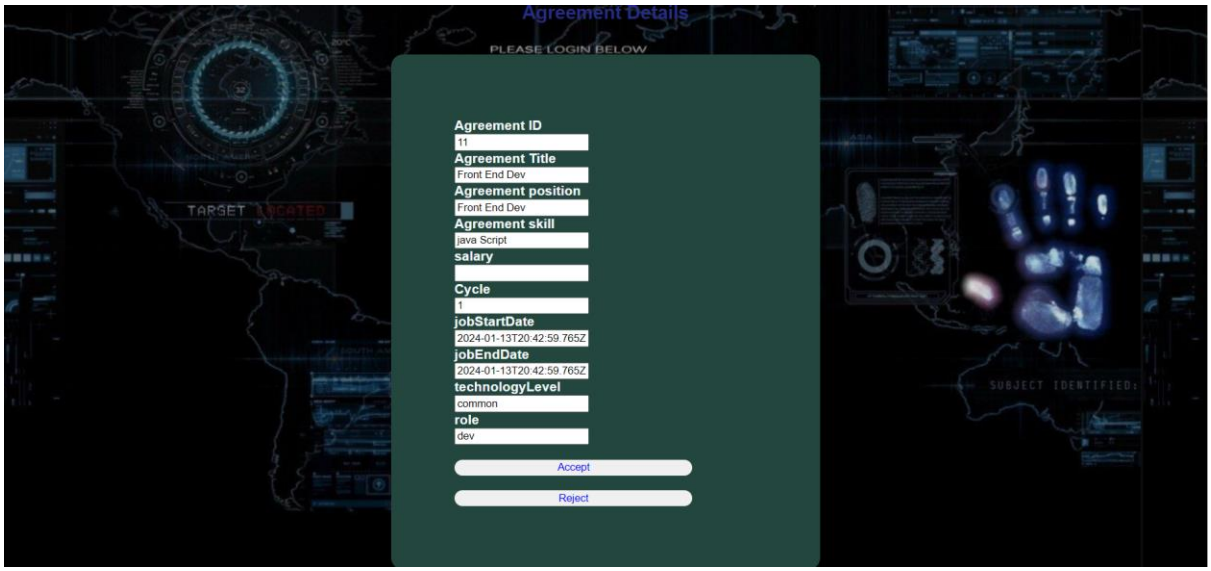


Figure 18: Agreement Details

Enter the salary field and click on accept

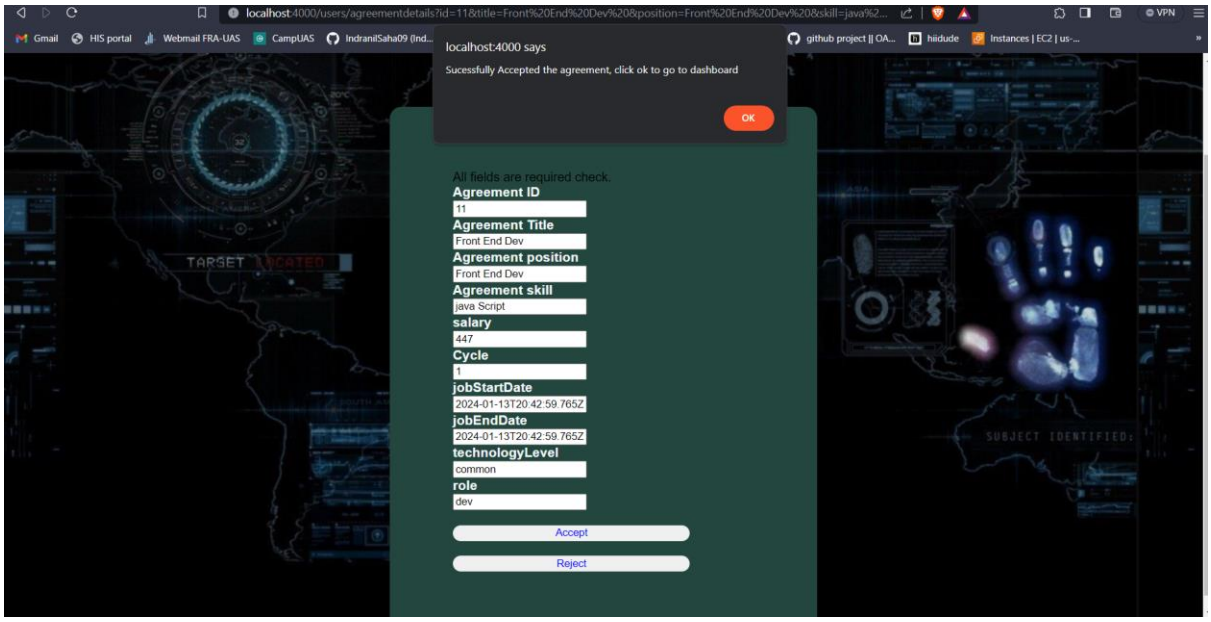


Figure 19: Enter Agreement Details

Click ok to go to dashboard

Click on MasterAgreementDetails to view the accepted master agreement

9.3.4 Screen: Master agreement bidding

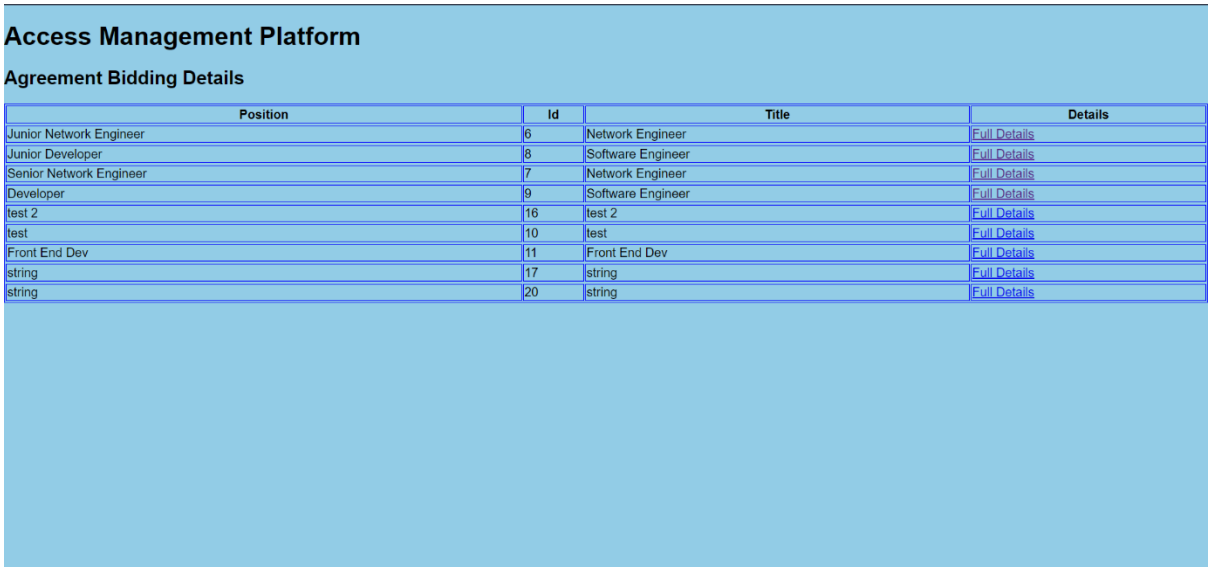


Figure 20: Master agreement bidding

Click on full details to view details

Fill in proper details and bid for the agreement.

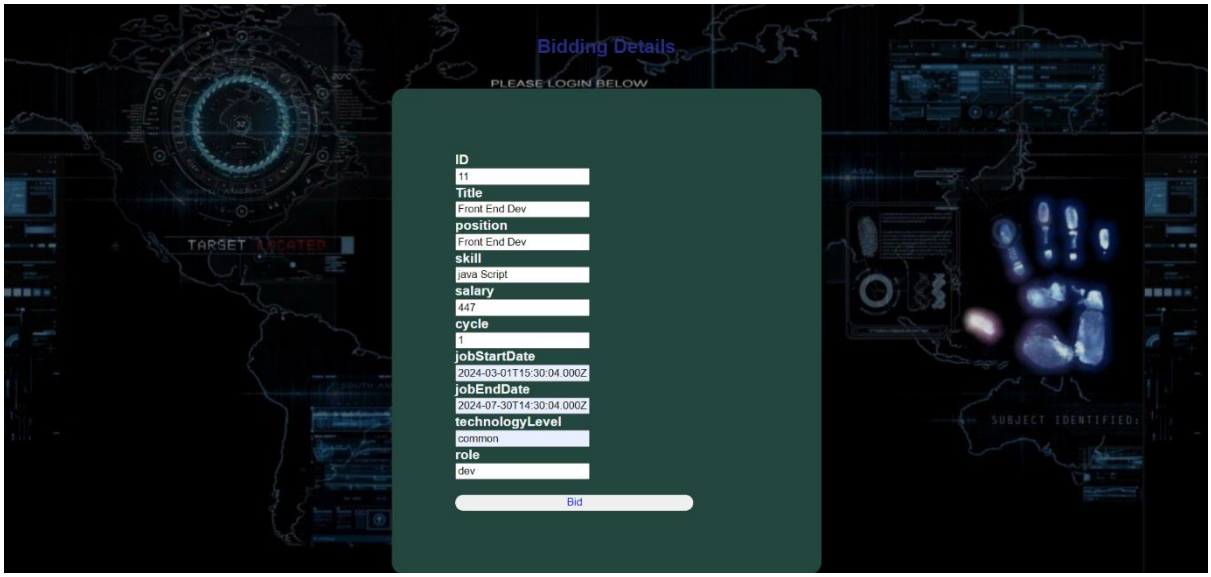


Figure 21: Bidding Details

Click on OK to dashboard

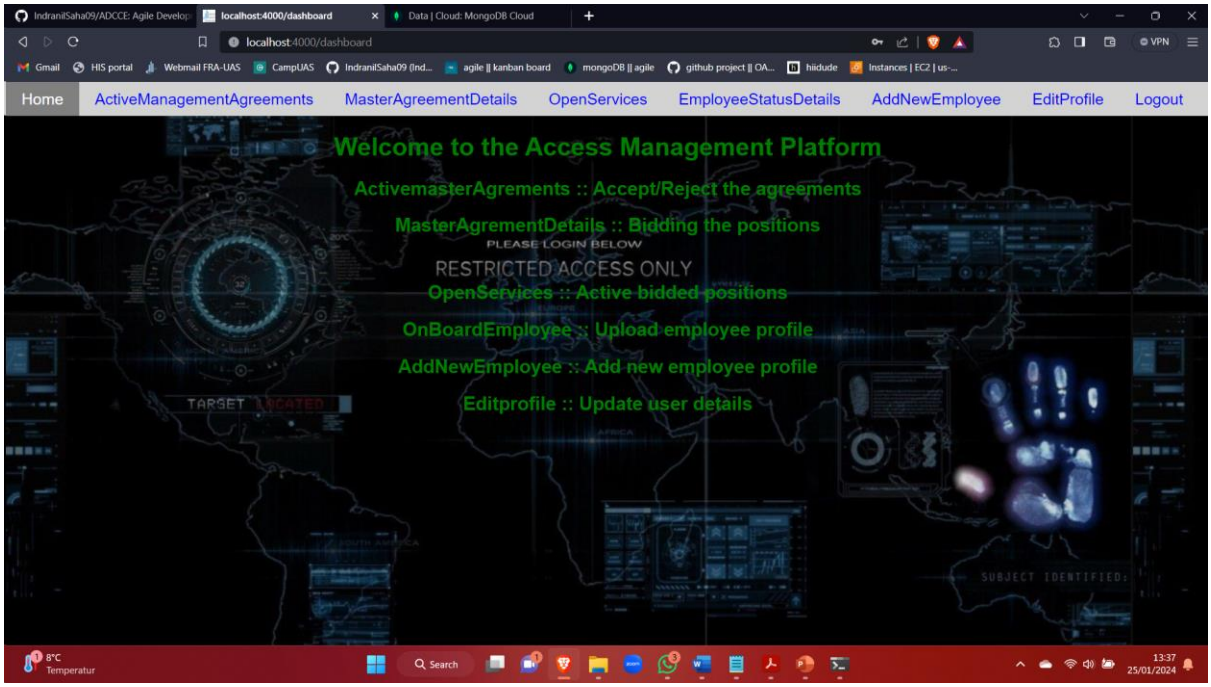


Figure 22: Dashboard

9.3.5 Screen: Open Service

Before fetching API of group 4b

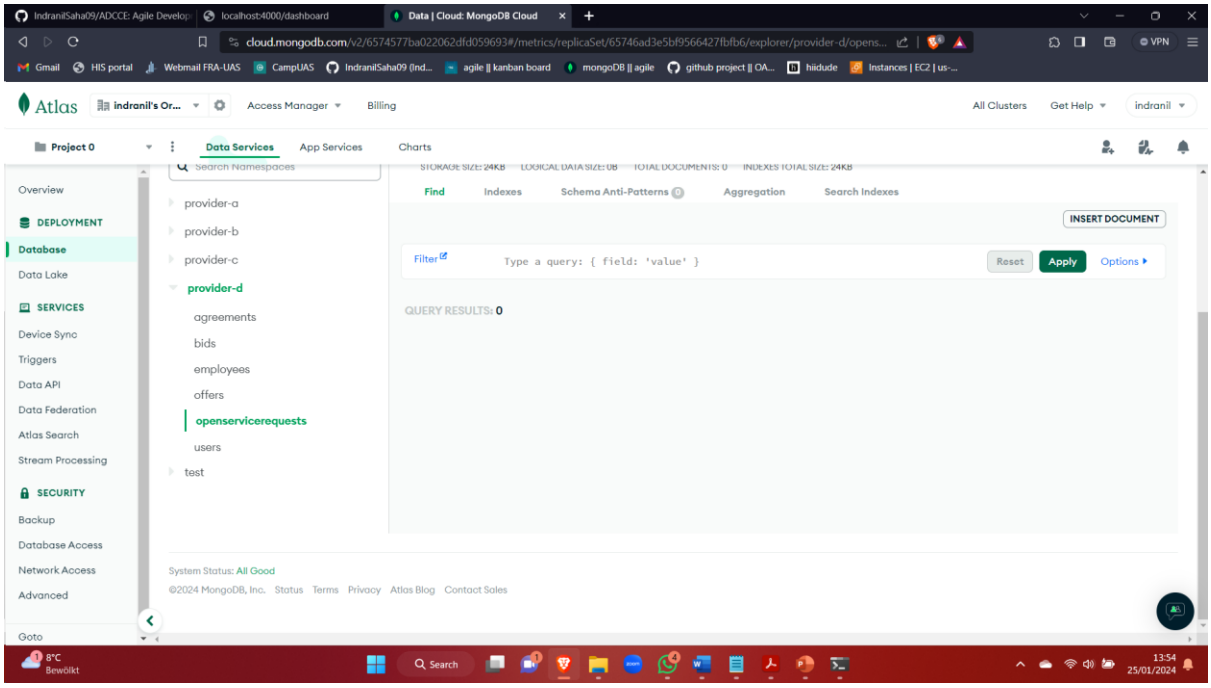


Figure 23: Database before fetching API of group 4b

Click on OpenServices

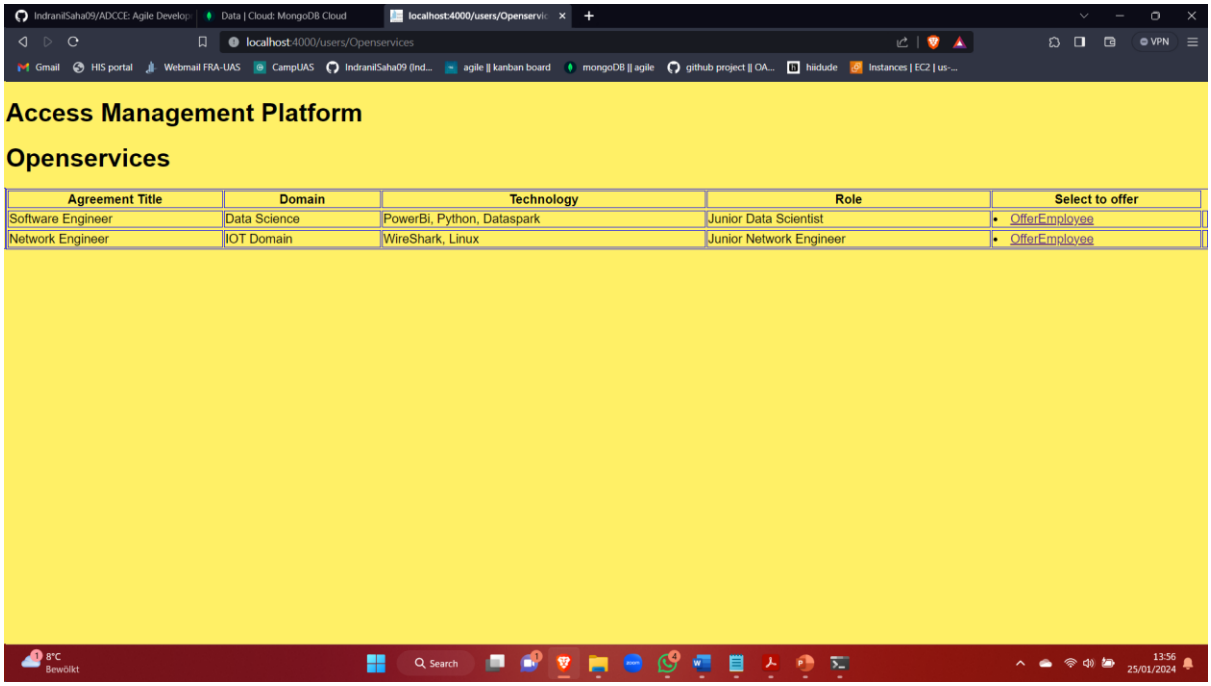


Figure 24: OpenServices

Click on offerEmployee to offer an employee against an open service request.

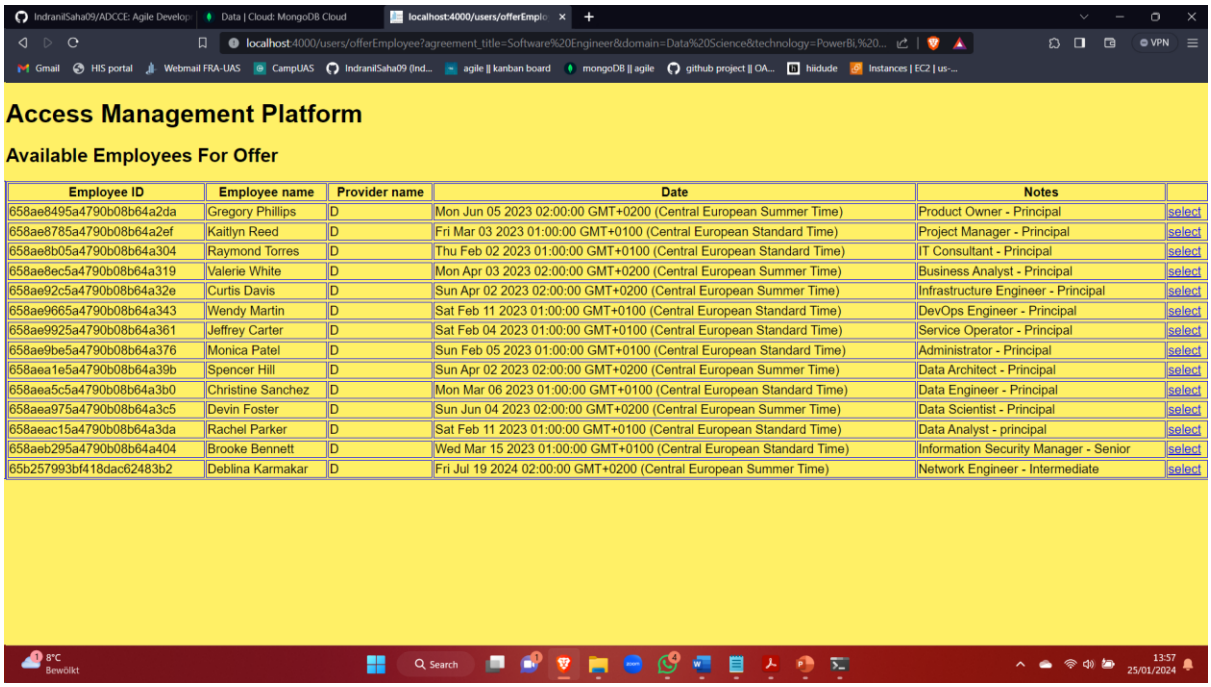


Figure 25: Available Employees for Offer

Click on select. Fill proper details like salary and submit the offer.

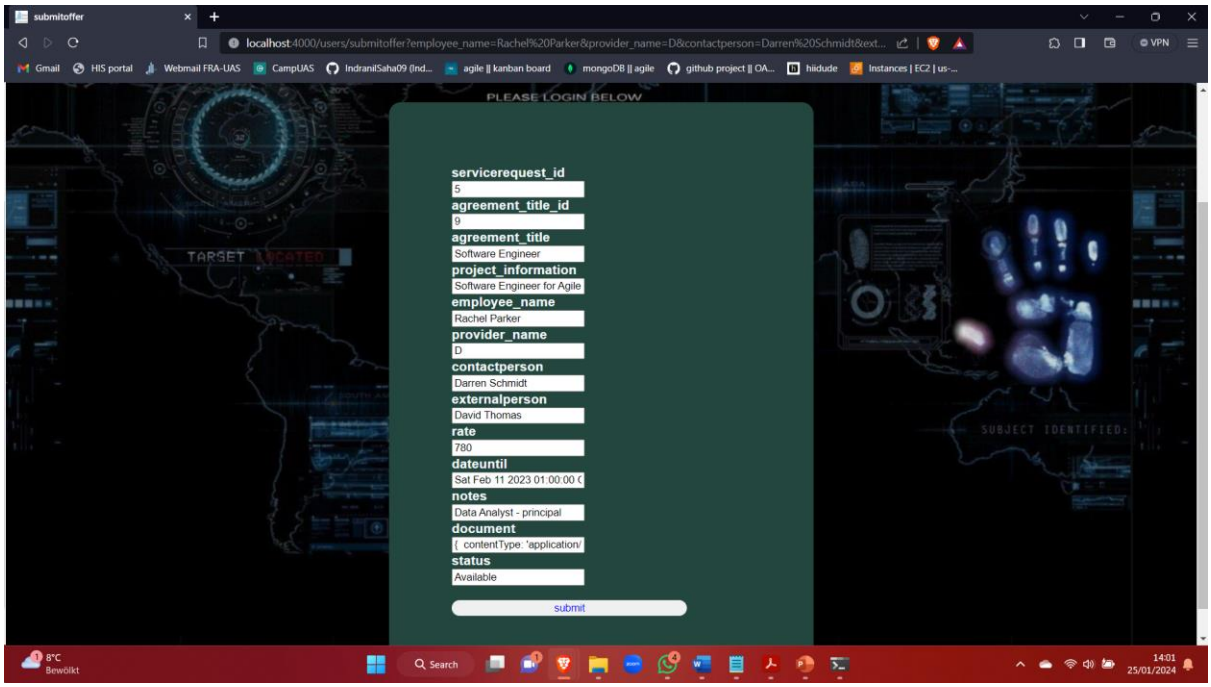


Figure 26: Available Employees for Offer – details

The data will get reflect in database.

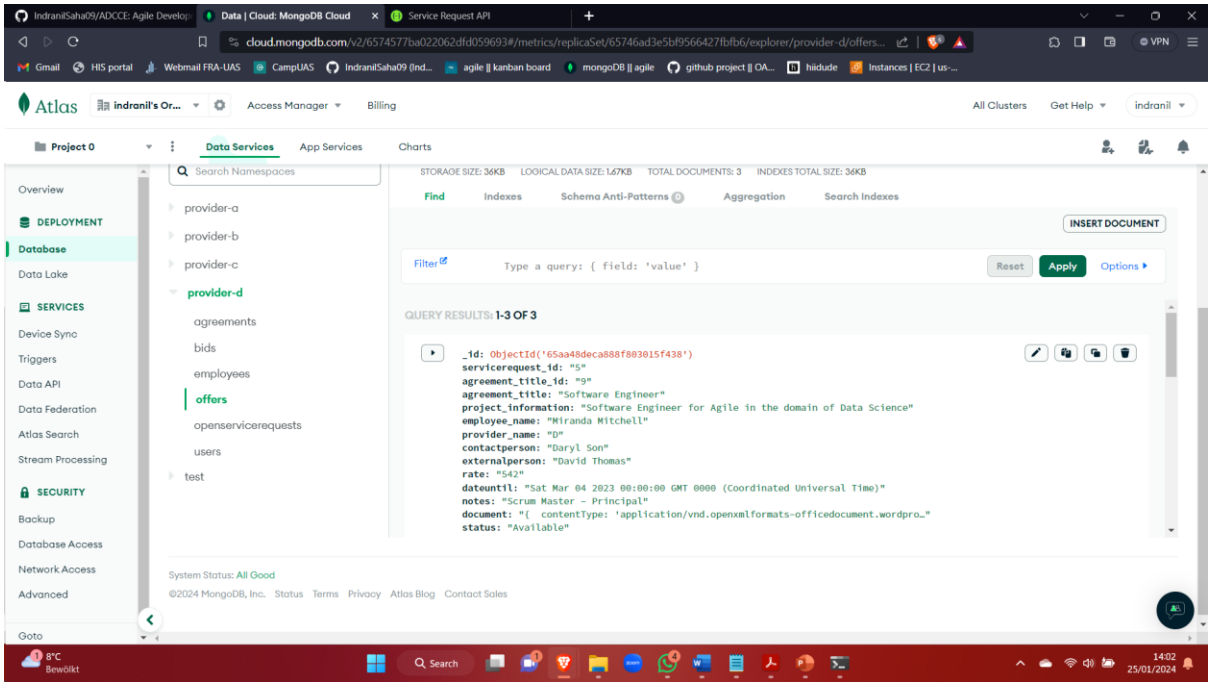


Figure 27: Database Entry

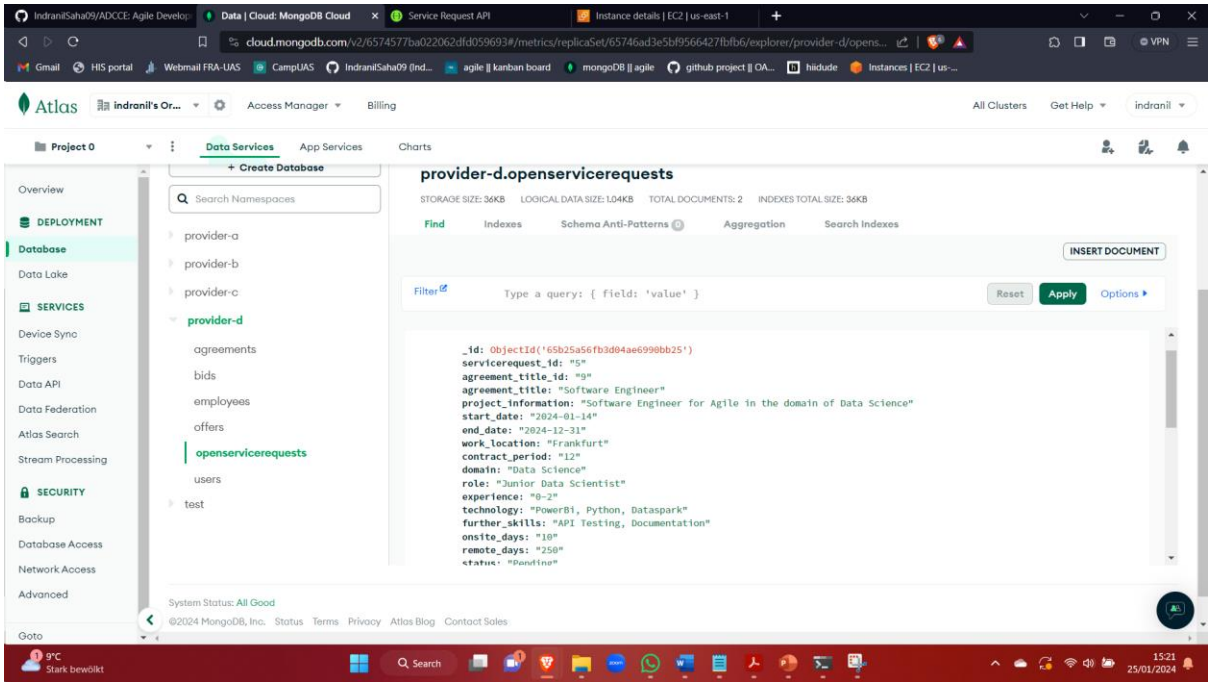


Figure 28: Database Entry – Open Service Request

9.3.6 Screen: Onboard Employee

For Adding new employee for the provider. Fill up with proper details and click on save. On clicking Save, it redirects to dashboard.

Figure 28: Add new employee

Details are also added in database.

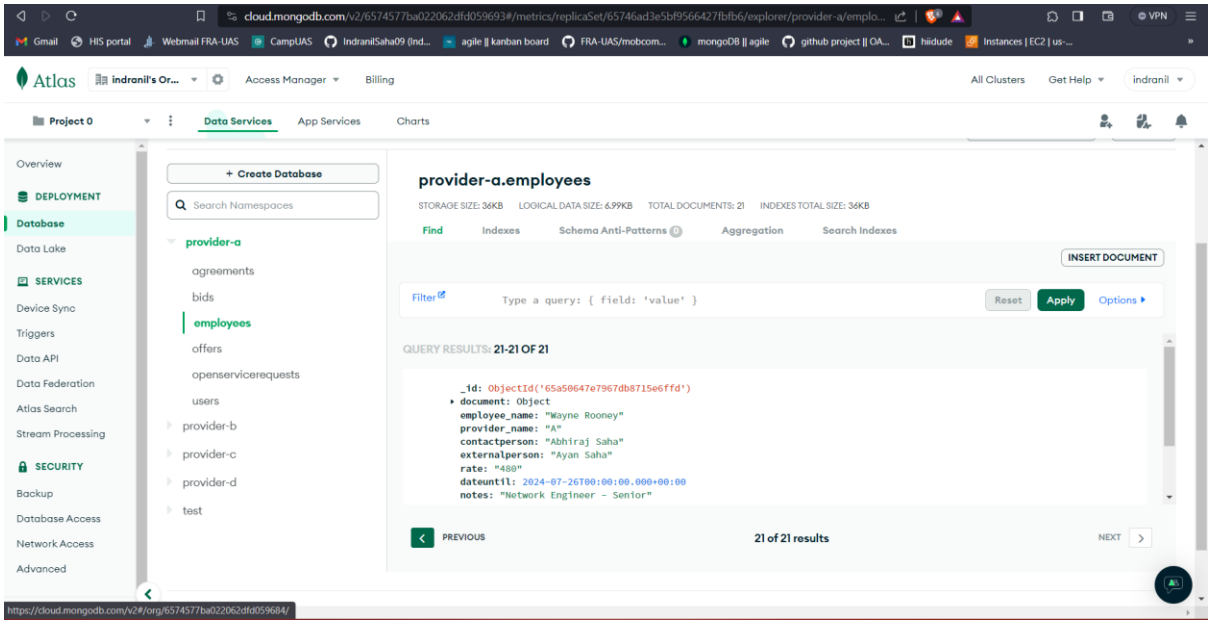
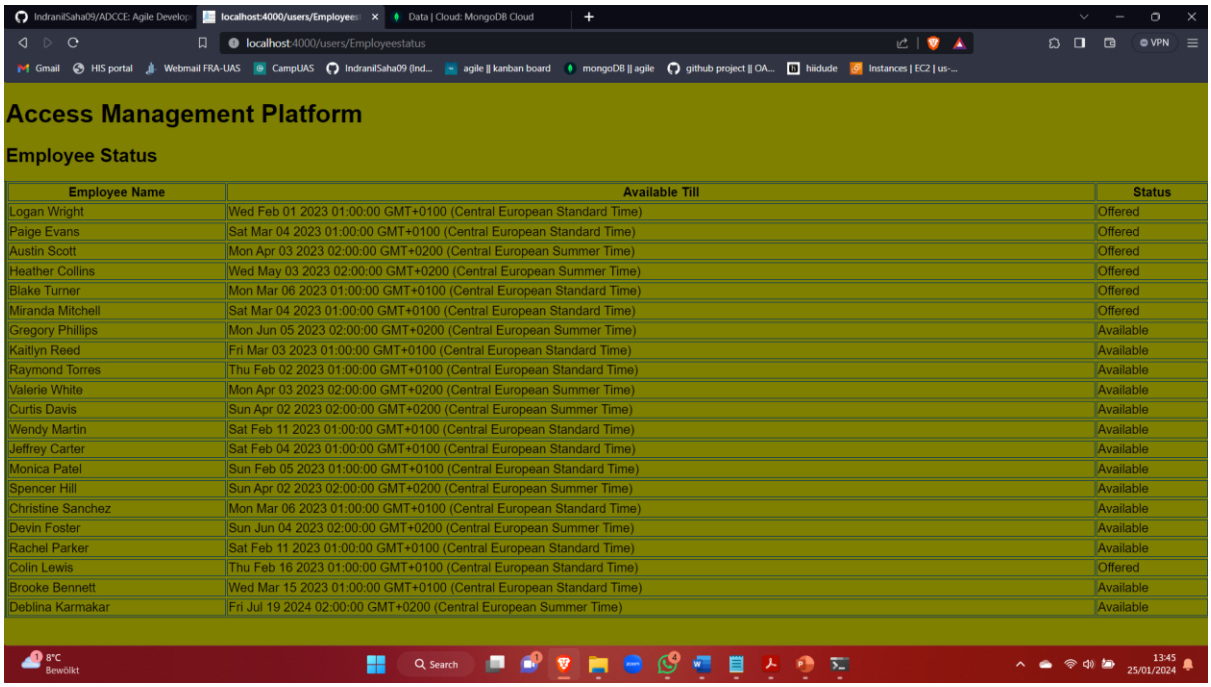


Figure 29: Database Entry

9.3.7 Screen: Employee status details

Click on EmployeeStatusDetails



Access Management Platform

Employee Status

Employee Name	Available Till	Status
Logan Wright	Wed Feb 01 2023 01:00:00 GMT+0100 (Central European Standard Time)	Offered
Paige Evans	Sat Mar 04 2023 01:00:00 GMT+0100 (Central European Standard Time)	Offered
Austin Scott	Mon Apr 03 2023 02:00:00 GMT+0200 (Central European Summer Time)	Offered
Heather Collins	Wed May 03 2023 02:00:00 GMT+0200 (Central European Summer Time)	Offered
Blake Turner	Mon Mar 06 2023 01:00:00 GMT+0100 (Central European Standard Time)	Offered
Miranda Mitchell	Sat Mar 04 2023 01:00:00 GMT+0100 (Central European Standard Time)	Offered
Gregory Phillips	Mon Jun 05 2023 02:00:00 GMT+0200 (Central European Summer Time)	Available
Kaitlyn Reed	Fri Mar 03 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Raymond Torres	Thu Feb 02 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Valerie White	Mon Apr 03 2023 02:00:00 GMT+0200 (Central European Summer Time)	Available
Curtis Davis	Sun Apr 02 2023 02:00:00 GMT+0200 (Central European Summer Time)	Available
Wendy Martin	Sat Feb 11 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Jeffrey Carter	Sat Feb 04 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Monica Patel	Sun Feb 05 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Spencer Hill	Sun Apr 02 2023 02:00:00 GMT+0200 (Central European Summer Time)	Available
Christine Sanchez	Mon Mar 06 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Devin Foster	Sun Jun 04 2023 02:00:00 GMT+0200 (Central European Summer Time)	Available
Rachel Parker	Sat Feb 11 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Colin Lewis	Thu Feb 16 2023 01:00:00 GMT+0100 (Central European Standard Time)	Offered
Brooke Bennett	Wed Mar 15 2023 01:00:00 GMT+0100 (Central European Standard Time)	Available
Deblina Karmakar	Fri Jul 19 2024 02:00:00 GMT+0200 (Central European Summer Time)	Available

Figure 30: EmployeeStatusDetails

10. Conclusion

In the process of developing the application, the primary objective was to streamline provider interaction with the master agreement and provide a seamless platform for initiating offer to the service requests. Adopting agile and scrum methodologies in the development process ensured an efficient and adaptable approach. Embracing agile practices allowed for continuous collaboration, quick adjustments to evolving requirements, and the delivery of incremental updates. Concurrently, the scrum framework provided a structured yet flexible environment, with iterative cycles ensuring that user feedback was promptly incorporated. Through this development approach, we not only optimized efficiency but also cultivated a user-centric mindset.

Commitment to understanding user requirements in-depth has resulted in the delivery of comprehensive services that effectively cater to the needs. The current state of the service capabilities reflects robust functionality, allowing users of any providers to navigate the master agreement effortlessly and initiate service requests with ease. Recognizing the dynamic nature of technology and user expectations, we acknowledge the potential for expansion. This forward-looking perspective underlines our anticipation of ongoing enhancements, reflecting our dedication to staying at the forefront of user satisfaction and technological advancements. By continually refining and expanding our application, we aim to ensure its long-term relevance and effectiveness in meeting the evolving needs of our user base.

11. Reference

1. https://nodejs.org/en/docs/
2. https://docs.openml.org/React/
3. https://devdocs.io/javascript/
4. https://phlow.de/magazin/webdesign/html/html-css-javascript-erklaert/
5. https://reactjs.org/docs/thinking-in-react.html