



Frankfurt University of Applied Sciences
Master of Engineering
Information Technology

Course:
Agile Development in Cloud Computing Environment

Authors:

Aneeta Antony	(1431461)
Aqib Javed	(1427145)
Gaurav Kale	(1427213)
Poonam Dashrath Paraskar	(1427297)
Pratik Desai	(1438367)
Sonam Priya	(1427129)

Index

Abstract	4
Project Overview	4
Goals	4
Interaction with other Scrum Teams:.....	4
Introduction.....	5
Why Agile Model over Waterfall?	5
Agile Software Development:.....	6
Kanban Board:	6
Benefits of Agile, Scrum, and Kanban:	6
Agile scrum methodologies	7
Agile Principles:	7
Scrum Framework:.....	7
Roles:	7
Artifacts:	8
Time Boxes:	8
Advantages of Agile Scrum:	8
Scrum Team.....	9
Scrum Master	9
Product Owner (POC)	10
Development Team	11
Sprint	11
Sprint Planning.....	11
Sprint Retrospective.	12
Best Practices.....	12
Development Environment and DevOps Techniques	13
GitHub and Deployment	13
Kanban Board	13
Software Architecture.....	15
AWS Deployment Strategy: Orchestrating Seamless Application Deployment:	15
Use Case Diagrams/Flow charts	16
Master Agreement Type:.....	16
Provider API Details:	17
Offer Workflow:.....	18
API Implementation	18
Master Agreement Type APIs:.....	18

GET API	19
POST API:	19
Provider API.....	20
GET API:	20
POST API:	20
Offers:	20
GET API:	21
Database tables:	21
Final Application Output	22
Login Page	22
Registration page.....	22
Provider Access Management Landing Page	23
Landing page with user menu	23
MasterAgreementType: Creation and Listing Page.....	24
Master Agreement creation page	24
Provider: Creation and Listing Page	25
Provider creation page	25
Offers Page:	25
Rating:.....	26
Conclusion:	27
Reference	27

Provider Management Component

Abstract

The phrase "agile software development" refers to a set of software development approaches based on the notion of iterative development, in which problems and solutions are developed in concert by self-organizing cross-functional teams. Two of the most popular Agile techniques are Scrum and Kanban. Our objective is to develop an application using an agile methodology based on the given requirement and communicate with other applications. As per the requirement, the application we developed can create and modify master agreements, also create providers and establishing offers from Providers to master agreements.

Keywords: Agile, Scrum, Kanban, Software Development, Sprint, Provider Platform.

Project Overview

The company "Future-X" wants to establish a new IT landscape for collaboration with their suppliers/providers (PMP=Provider Management Platform). As they do not have enough developers and experts, they mandate a new upcoming company ADCCE.

Our Project Group 2a delivers the Provider Management component of the final project.

Goals

- The development of an API responsible for helping the company future X in creating master agreement types.
- Secondly, enabling of a Provider API. This provider API helps in creating a new provider who will supply corresponding domains and roles.
- The next section in our application provider management component we have a section offers. In the offer speech the company future X will receive the bided offer coated amount for each roll will be displayed to the company future X and they will decide on which provider to accept 4 their master agreement type.
- In the Offer page the company future X can provide their personal experience for each provider as ratings.
- User will figure out provider quality according to the best price offered.

Interaction with other Scrum Teams:

- Integrate with group 1 API for login credentials for the application and login as an authorized user.
- Deliver APIs to Group 3a and 4a (Fetch all: Master Agreement Type API, Fetch all Provider API).
- Use Offer API from Group 3a to display in offer page and do next steps.

Introduction

Agile software development encompasses a range of iterative approaches wherein self-organizing cross-functional teams collaboratively develop requirements and solutions. Agile methods advocate for a disciplined project management process, emphasizing frequent inspection and adaptation. This methodology embraces a leadership philosophy that fosters teamwork, self-organization, and accountability, along with engineering best practices aimed at swift delivery of high-quality software. The business approach of agile aligns development with client needs and company goals.

Within the realm of agile, Scrum stands out as a popular and lightweight process framework. It introduces specific concepts and procedures, grouped into roles, artifacts, and time boxes, distinguishing it from other agile methodologies.

The Kanban board serves as a tool for visualizing workflows, enhancing work process efficiency, and reducing work-in-progress. This visualization fosters increased transparency, allowing teams to readily identify and address problematic work phases. Consequently, this heightened openness facilitates effective adjustments, leading to a more efficient and productive team.

Why Agile Model over Waterfall?

- The Agile methodology follows an iterative process in which projects are divided into sprints of shorter duration. Unlike the traditional approach, in Agile the time spent on initial planning and prioritization of activities is reduced because this methodology is more flexible with respect to changes in initial requirements.
- The process in Agile is customer focused. As a result, each stage involves the user or client constantly, seeing their progress, and giving feedback. So, the chances of the product not meeting their expectations are reduced compared to the waterfall model, in which a customer orders a product and knows nothing about it until its completion. With this, the scenario of delivering a final product is eliminated, which, if there is negative feedback from the user, could take months to update, causing a significant impact on the project and the client.
- That Agile is based on successive releases is of great importance for the client since they can start using parts of the product as soon as they are released; it is not necessary to wait until the end of the project to use them.
- In an incremental methodology like Agile, both the client and the team involved in the project know the progress status, what is already available and what is still missing. The risk associated with the development process is reduced and provides visibility and transparency, which is impossible with other traditional methodologies such as waterfall.
- Agile is well-focused on complex projects, where all the requirements and the deliverables are not fully known from the start. With successive approximations, it is possible to get closer to the desired product, something that is not possible with waterfall.
- Agile provides mechanisms to manage uncertainty adequately; since the project is divided into several sections or sprints, it is possible to resolve doubts, clarify the vision

and improve work, which translates into a gradual reduction of uncertainty and cost of the project.

- Waterfall is not prepared when business processes are changing while the project is developing. Agile, in contrast, not only can adapt to changing needs, but also anticipates and plans to address them at the appropriate time.

Agile Software Development:

Definition: Agile is an iterative and incremental approach to software development that emphasizes flexibility and collaboration. It values customer feedback, adapts to changing requirements, and focuses on delivering small, functional pieces of software in short development cycles.

Principles: The Agile Manifesto outlines core values such as individuals and interactions over processes and tools, working software over comprehensive documentation, customer collaboration over contract negotiation, and responding to change over following a plan.

Scrum: Scrum is a specific framework within the agile methodology that provides a structure for teams to work collaboratively. It is based on the principles of transparency, inspection, and adaptation.

Roles: Scrum defines specific roles, including Product Owner, Scrum Master, and Development Team, each with distinct responsibilities.

Artifacts: Scrum uses artifacts like Product Backlog, Sprint Backlog, and Increment to facilitate communication and transparency.

Time Boxes: Scrum introduces time-boxed events like Sprint Planning, Daily Standup, Sprint Review, and Sprint Retrospective to ensure regular inspection and adaptation.

Kanban Board:

Definition: Kanban is another agile methodology, and the Kanban board is a visual tool used to manage and optimize workflows. It visualizes the work process and helps in identifying bottlenecks, managing work-in-progress, and improving efficiency.

Columns and Cards: The Kanban board typically consists of columns representing different stages of the workflow (e.g., To Do, In Progress, Done). Cards represent individual tasks or user stories.

Limiting WIP: Kanban emphasizes limiting the work-in-progress to avoid overloading the team and to ensure a smoother flow of work.

Continuous Improvement: With a Kanban board, teams can easily identify areas for improvement and adjust enhance productivity and delivery speed.

Benefits of Agile, Scrum, and Kanban:

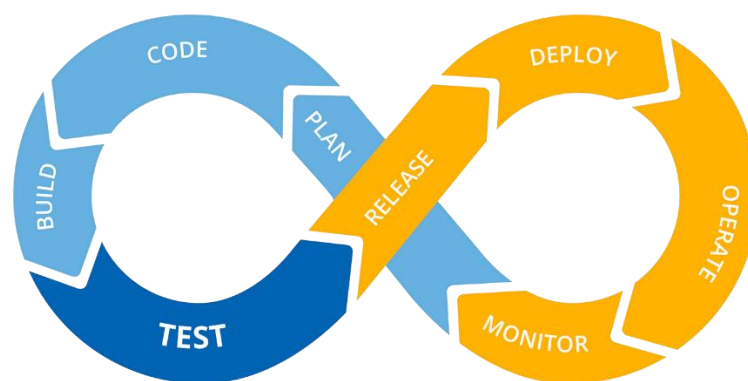
Flexibility: Agile methodologies allow teams to adapt quickly to changing requirements.

Transparency: Scrum and Kanban provide transparency into the development process, fostering collaboration and communication.

Continuous Improvement: Both Scrum and Kanban emphasize continuous improvement through regular inspection and adaptation.

Customer Satisfaction: Agile methodologies aim to align development with customer needs, leading to higher customer satisfaction.

In summary, agile methodologies, including Scrum and Kanban, have become popular approaches for software development due to their flexibility, collaborative nature, and emphasis on delivering value to customers.



Agile scrum methodologies

Agile Scrum methodologies represent a collaborative and iterative approach to software development, emphasizing flexibility and responsiveness to changing requirements. The term "Agile Scrum" specifically refers to the integration of the broader Agile principles with the Scrum framework. Here's an overview of key aspects:

Agile Principles:

Flexibility: Agile emphasizes adaptability to changing requirements throughout the development process.

Customer Collaboration: Prioritizing customer feedback and involvement in the development process.

Iterative Development: Delivering small, functional increments of software in short cycles

Scrum Framework:

Roles:

Product Owner: Represents the customer, defines features, and prioritizes the product backlog.

Scrum Master: Facilitates the Scrum process, removes obstacles, and ensures adherence to Scrum principles.

Development Team: Self-organizing and cross-functional, responsible for delivering the product increment.

Artifacts:

Product Backlog: A prioritized list of features and tasks representing the product requirements.

Sprint Backlog: A subset of the product backlog selected for a specific sprint.

Increment: The potentially shippable product version created during a sprint.

Time Boxes:

Sprint: A fixed time frame (usually 2-4 weeks) during which a potentially shippable product increment is developed.

Sprint Planning: A meeting at the beginning of the sprint to plan the work.

Daily Scrum: A daily stand-up meeting for the team to discuss progress and plan the day.

Sprint Review: A meeting at the end of the sprint to review and demonstrate the product increment.

Sprint Retrospective: A meeting at the end of the sprint to reflect on the process and make improvements.

Advantages of Agile Scrum:

Adaptability: Ability to respond quickly to changing requirements.

Transparency: Clear visibility into the development process through regular reviews and reflections.

Customer Satisfaction: Involvement of the customer throughout the development cycle ensures alignment with their needs.

Continuous Improvement: Regular retrospectives promote a culture of continuous learning and enhancement.

Implementation:

Cross-Functional Teams: Teams are composed of individuals with diverse skills to handle various aspects of development.

User Stories: Requirements are often expressed as user stories, small, comprehensible features from an end-user perspective. Summary, Agile Scrum methodologies merge the flexibility of Agile with the specific roles, artifacts, and time-boxed events defined by the Scrum framework, fostering a collaborative and efficient approach to software development.

Scrum Team

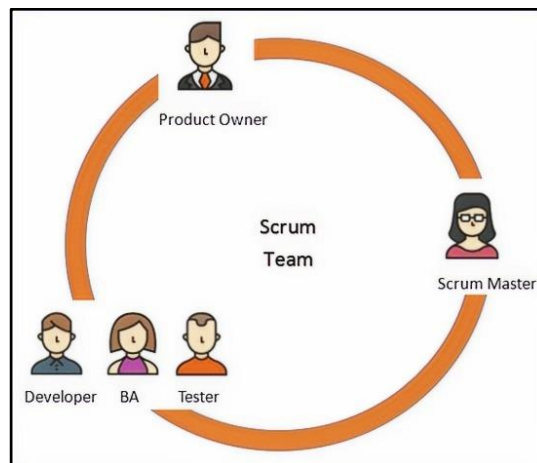
The fundamental unit of Scrum is a small team of people, a Scrum Team. The Scrum Team consists of one Scrum Master, one Product Owner, and Developers. Within a Scrum Team, there are no sub-teams or hierarchies. It is a cohesive unit of professionals focused on one objective at a time, the Product Goal.

Scrum Teams are cross-functional, meaning the members have all the skills necessary to create value for each Sprint. They are also self-managing, meaning they internally decide who does what, when, and how.

The Scrum Team is small enough to remain nimble and large enough to complete significant work within a Sprint, typically 10 or fewer people. In general, we have found that smaller teams communicate better and are more productive. If Scrum Teams become too large, they should consider reorganizing into multiple cohesive Scrum Teams, each focused on the same product. Therefore, they should share the same Product Goal, Product Backlog, and Product Owner.

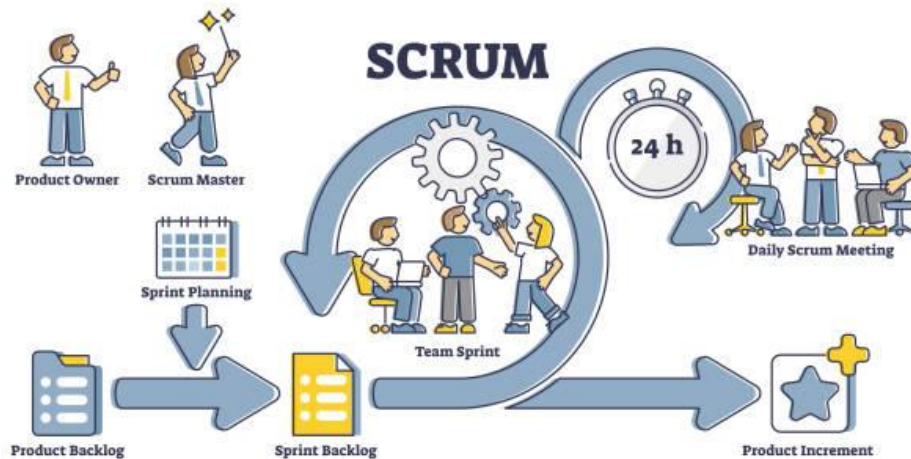
The Scrum Team is responsible for all product-related activities from stakeholder collaboration, verification, maintenance, operation, experimentation, research and development, and anything else that might be required. They are structured and empowered by the organization to manage their own work. Working in Sprints at a sustainable pace improves the Scrum Team's focus and consistency.

The entire Scrum Team is accountable for creating a valuable, useful Increment every Sprint. Scrum defines three specific accountabilities within the Scrum Team: the Developers, the Product Owner, and the Scrum Master.



Scrum Master

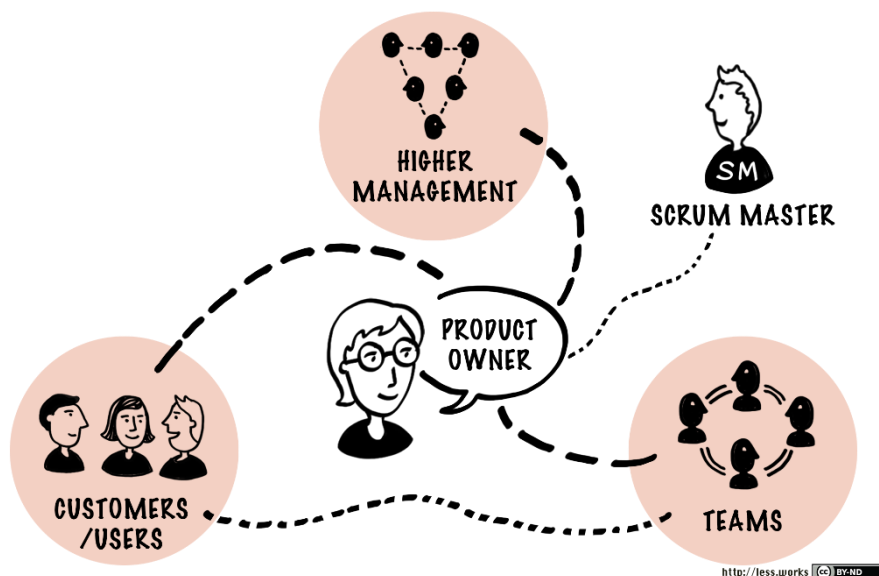
The Scrum Master plays a crucial role in ensuring the smooth operation of the process, eliminating obstacles to productivity, and facilitating key meetings. Responsibilities include educating the product owner on effective Scrum utilization to achieve goals and maximize return on investment (ROI). Fostering empowerment and creativity, the Scrum Master strives to enhance the quality of life for the development team and finds ways to boost their productivity. Implementing improvements in engineering procedures and methods is essential to make each capability increment potentially shippable. Keeping the team well-informed about the latest developments is part of the Scrum Master's duties.



A proficient Scrum Master should possess in-depth knowledge of Scrum to coach and train other roles and provide support to various process stakeholders. Constant awareness of the project's state, comparing current progress to anticipated progress, and addressing any obstacles to progress are vital tasks. The Scrum Master must be adaptable and proactive in recognizing and resolving difficulties as they arise. Acting as a liaison between different parties, the Scrum Master shields the team from external interference. It's important to note that task distribution is the responsibility of the team, and the Scrum Master does not assign tasks to individual team members.

Product Owner (POC)

The responsibility for defining criteria lies with the Product Owner, who acts as the ultimate authority on requirements and their prioritization for the Team. The Product Owner essentially serves as the "single source of truth," maintaining a crucial link between the Team and the broader business, clients, and their product-related needs. Acting as the central point of contact, the Product Owner handles all inquiries regarding product requirements and acts as a buffer between the Team and requests for features and bug fixes from various sources.



In close collaboration with the team, the Product Owner works to define both user-facing and technical requirements, documents them as needed, and determines the optimal sequence for implementation.

Keeping the Product Backlog up to date at a level of detail and quality expected by the Team, the Product Owner serves as the repository for all this information. Additionally, the Product Owner assesses whether implementations meet the required functionality and quality for release, and they set timelines for delivering completed work to customers.

Development Team

The product's creation and testing are done by the Team, a self-organizing and cross-functional group of individuals. The Team is responsible for the hands-on work, giving them the authority to determine the approach to tasks and ensuring accountability for delivering the final output. This self-organizing aspect means that, during the Sprint, Team members independently decide how to allocate work tasks and assign responsibilities among themselves. Ideally, the team size should range from five to nine members.

Sprint

To ensure consistency and facilitate rapid iterations for feedback, work periods are defined to last one month or less. This shorter timeframe enables both ongoing work and tasks to be regularly inspected and adapted. It's crucial to maintain shorter cycles, as longer ones may compromise the essence of frequent feedback and potentially escalate the difficulty and risk involved. As soon as one Sprint concludes, a new Sprint promptly commences in succession.

Sprint Planning

Sprint	Start	End
Sprint 1	11/10/2023	24/11/2023
Sprint 2	24/11/2023	08/12/2023
Sprint 3	08/12/2023	22/12/2023
Sprint 4	22/12/2023	12/01/2024
Sprint 5	12/01/2024	21/01/2024

Sprint 1: First major task in the sprint is Architecture design and pattern discussion. Edit profile and Logout-Frontend.

Sprint 2: Design of Login page and its sub fields on Frontend and Blackened, implementation of Nav-bar.

Sprint 3: Sharing of APIs created between group 1 and group 2 for open services agreement.

Sprint 4: Update the offered employee status with the response from project A (Profile Accepted/Rejected)

Sprint 5: Testing of data between group 1 and 2. Updating selected profile in database and fixing file path object.

Sprint 6: The extra week to finalize the Project and complete documentation.

Sprint Retrospective.

Inspection	Adaptation	Participants	Timebox
Discussion on technology stack Architecture design and pattern discussion	We have finalized the technology stack we are going to use in the project. Finalized layout for profile and logout Front end	All team members	13/11/2023 2 hours 30 minutes.
Distribution of work and responsibilities	Team members divided work among themselves according to their skillset	All team members	20/11/2023 2 hours
Finalize the UI	We have created the mock UI and made the necessary changes by discussing in the retrospective	All team members	27/11/2023 3 hours
Deploying application on AWS	Deployed the application on cloud and solved the problem of data overlapping	All team members	29/12/2023
Collaboration with other teams and testing	Testing of the platform	All team members	19/01/2024

Following the conclusion of Sprint #1, the team exercised the freedom to select Story Points (SP) that enabled both Team Members (TMs) to optimize their bandwidth. This determination arose during the postmortem meeting of the first sprint. An important outcome on Day 14 (Tuesday) was the establishment of criteria for Definition of Done (DOD) and Definition of Readiness (DOR). Traditionally, the retrospective used to take place on the second Friday of each sprint.

Best Practices

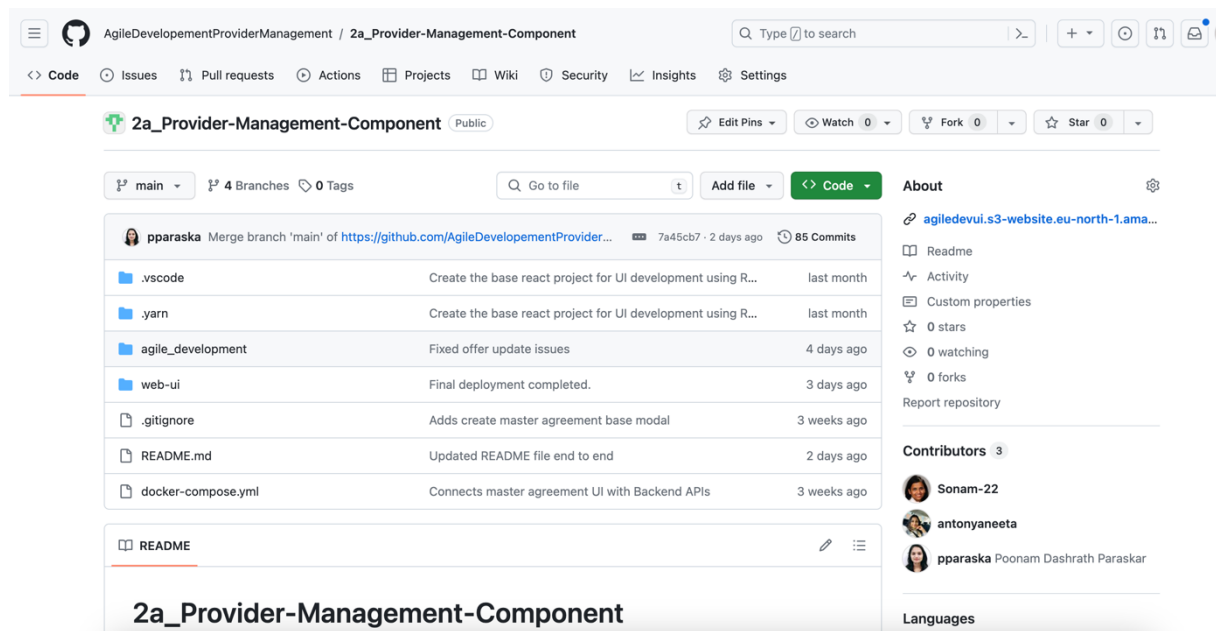
After in-depth discussions between Team Members 1 and 2, the decision was made to adopt a two-week interval per sprint. This choice was influenced by the manageable size, allowing for limited requests in stories, small deliverables, and the flexibility to increase iterations if needed.

Key outcomes included keeping daily sprint meetings concise, not exceeding 5 minutes. Additionally, two sessions of Product Backlog Refinement (PBR) occurred between the Product Owner (PO), Backlog Owner (BO), and Team Members (TM). A Team Demo was conducted to ensure that completed stories not only met the specified requirements but also satisfied the PO, aiding in team code reviews. The Sprint Review ensured proper closure of all stories and addressed any potential spillover stories from the sprint. Distinct responsibilities were assigned, such as maintaining sprint velocity and handling backlog stories appropriately. The calculation of story points (SP) was collaboratively performed by the team during PBR sessions.

Development Environment and DevOps Techniques

GitHub and Deployment

[GitHub Repository Link](#)



Kanban Board

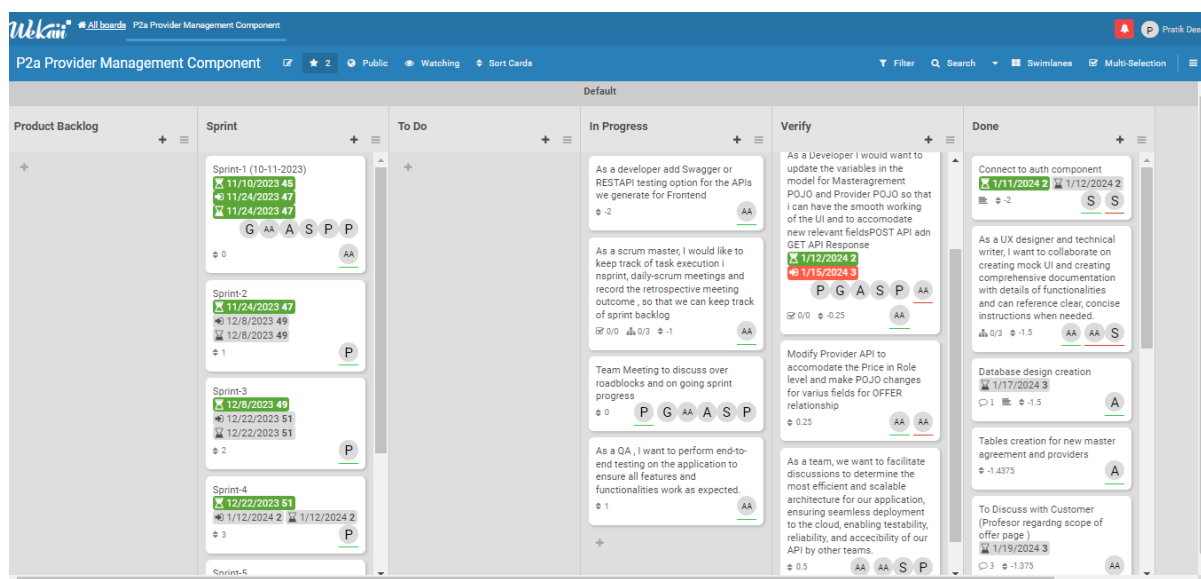


Figure 1

Task name: It is easier to grasp what task needs to be done when task labels begin with a verb. For instance, the task "Add functionality to web app" makes it obvious what sort of work will be required to achieve it on a high level.

Key dates: There are some dates that are significant to project managers and developers alike, depending on your team's process. For instance, a project manager might require knowledge of a task's start date, whereas a developer would require knowledge of the task's deadline.

Task owner: This person oversees getting the job done. This is the person to get in touch with for clarification if anyone has any queries concerning this task.

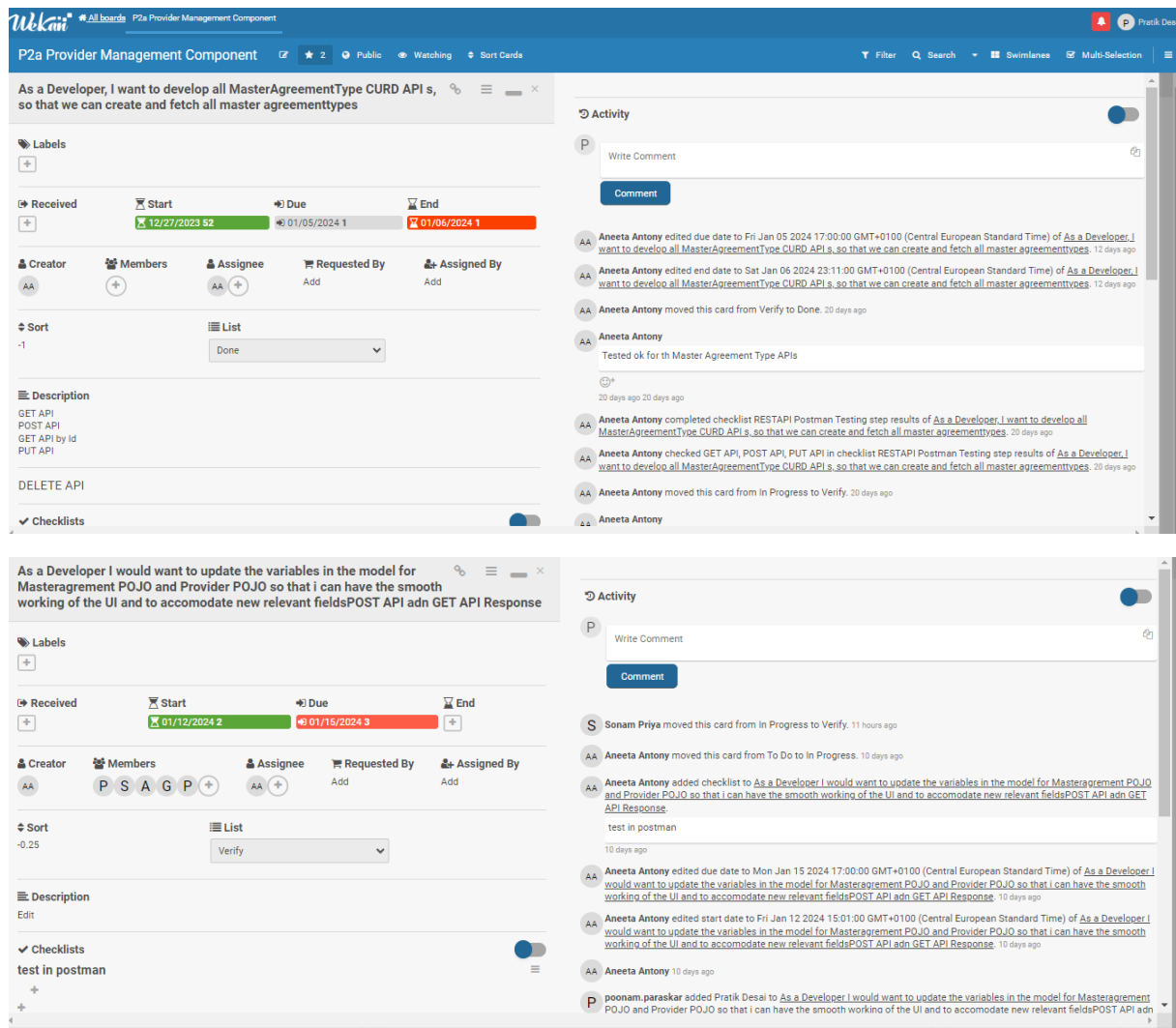


Figure 2

Task status: This is frequently expressed in the Kanban system by the location of the card on a Kanban board. It is crucial that everyone is aware of the various stages of your team's Kanban board because every team has a different process for keeping track of work in progress.

Task priority: This illustrates how significant an activity is in relation to those surrounding it. Setting priorities can aid in maximizing lead time for jobs that demand it. This might assist your development team in prioritizing their work.

Subtasks: Some tasks might be labor-intensive enough to be divided into smaller subtasks. These tasks can be linked to a parent task to keep the work structured.

Software Architecture

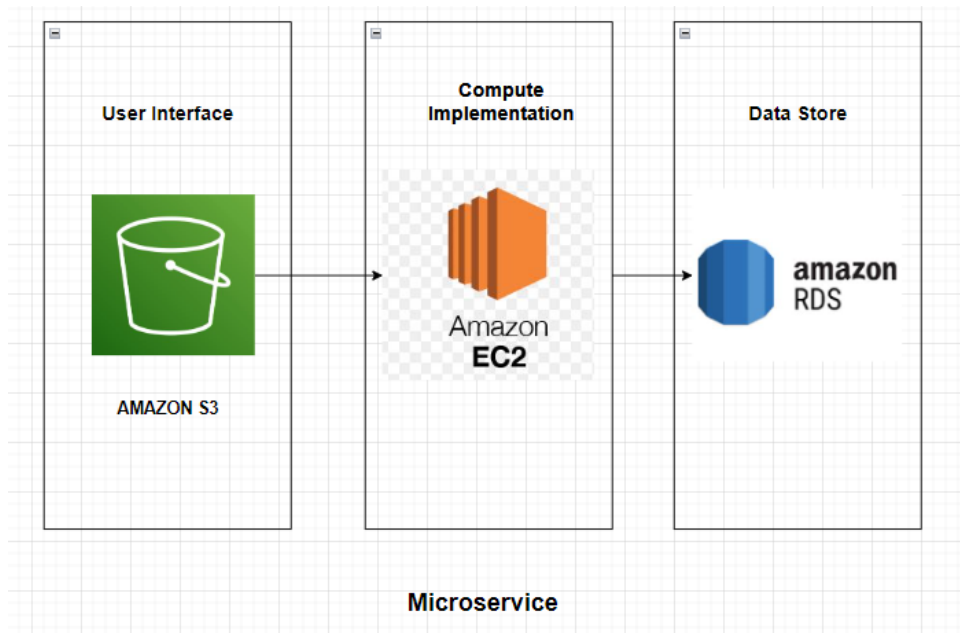
The software architecture we used is micro service based modularized architecture. Our UI is React Vite template-based module. The backend code stack is in Java programming language, and a spring boot application is generated as backend module. The database we use is MySQL. All 3 modules, frontend, backend and the database are deployed in AWS.

Dependencies	Version
React	18.2.0
React Material UI	5.15.1
Vite	5.0.8
Axios	1.6.3
Yarn	4.0.2
Node Js	18.19.0
TypeScript	5.2.2
Java	17
SpringBoot	3.2.1
Oracle MySQL	8.0

AWS Deployment Strategy: Orchestrating Seamless Application Deployment:

AWS Backend Application Deployment Steps:

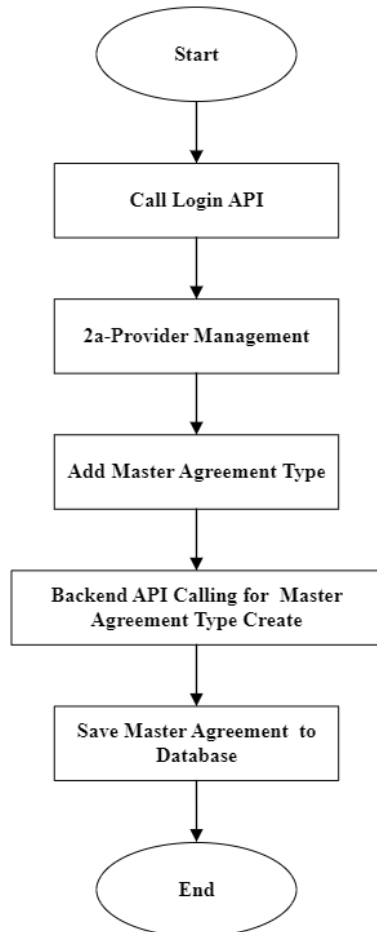
1. Begin by setting up an RDS instance with the MySQL engine in AWS, ensuring that public access is enabled.
2. For the Spring Boot application, execute the following command on your local machine to package the Maven project and generate a .jar file:
mvn package
3. Once the .jar file is created, upload it to an S3 bucket. While uploading, grant public read access to the object.
4. Create an Ubuntu EC2 instance to host the Java application. Ensure that the inbound rule for port 3306 (SQL) is enabled in the security group of the EC2 instance.
5. Once the EC2 instance is up and running, install the Java Development Kit (JDK) on it.
6. Retrieve the .jar file from the S3 bucket to the EC2 instance using the following command:
wget object_url
7. Run the .jar file on the EC2 instance with the following command.
java -jar jar_file_name



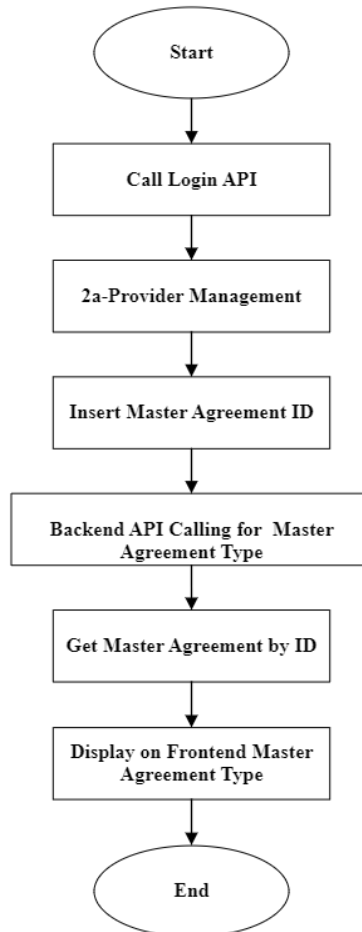
Use Case Diagrams/Flow charts

Master Agreement Type:

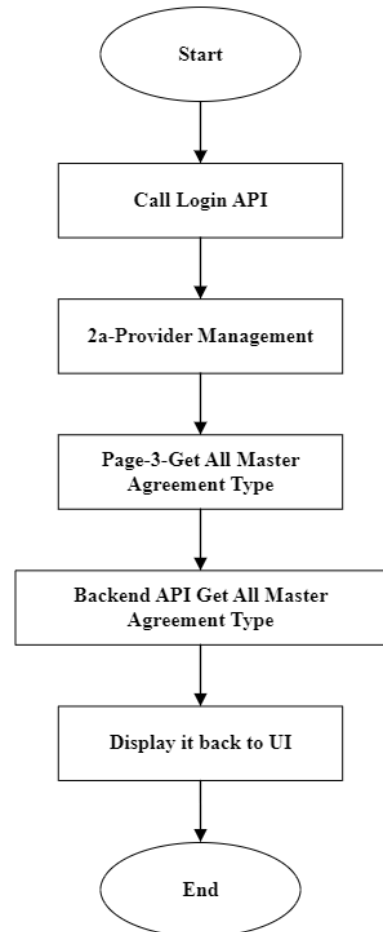
1.Create Master Agreement Type API



2.Find Master Agreement Type by ID API

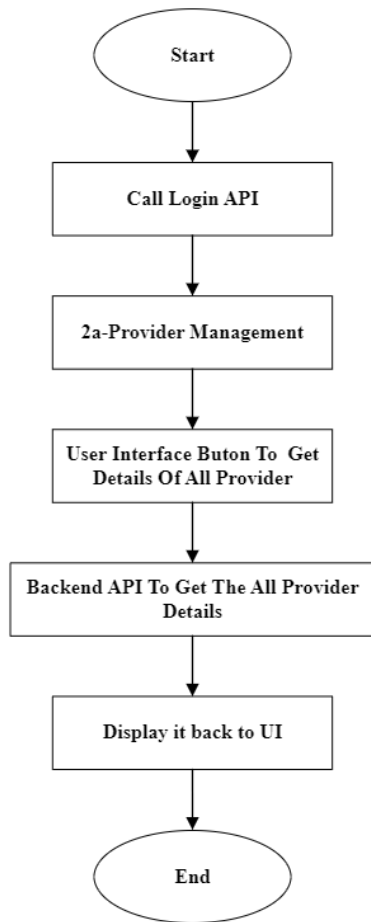


3.Get All Master Agreements API:

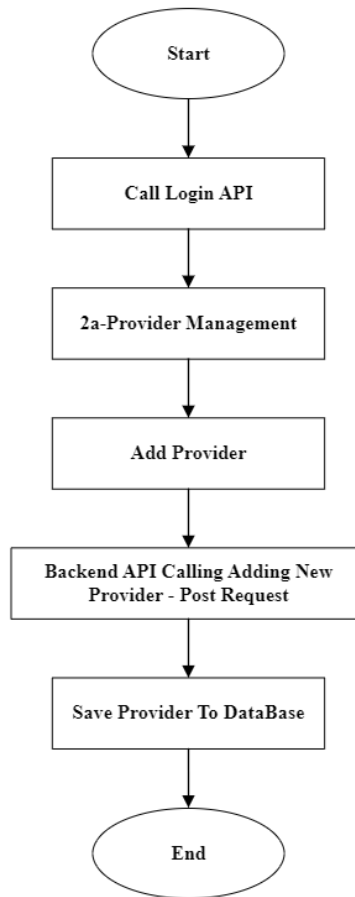


Provider API Details:

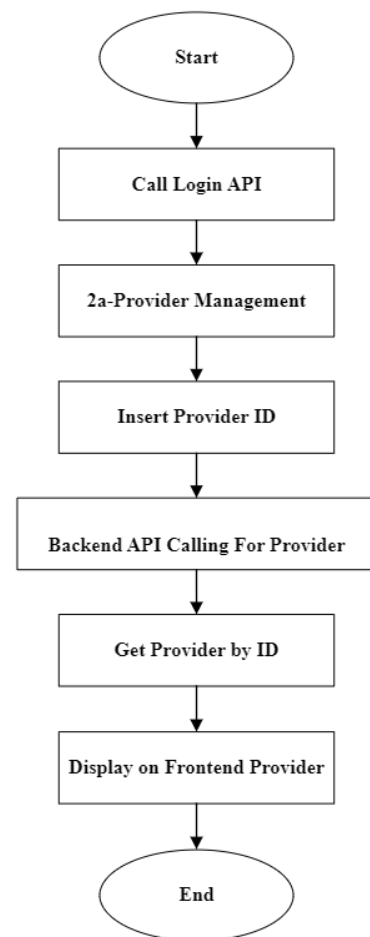
1. Create New Provider API



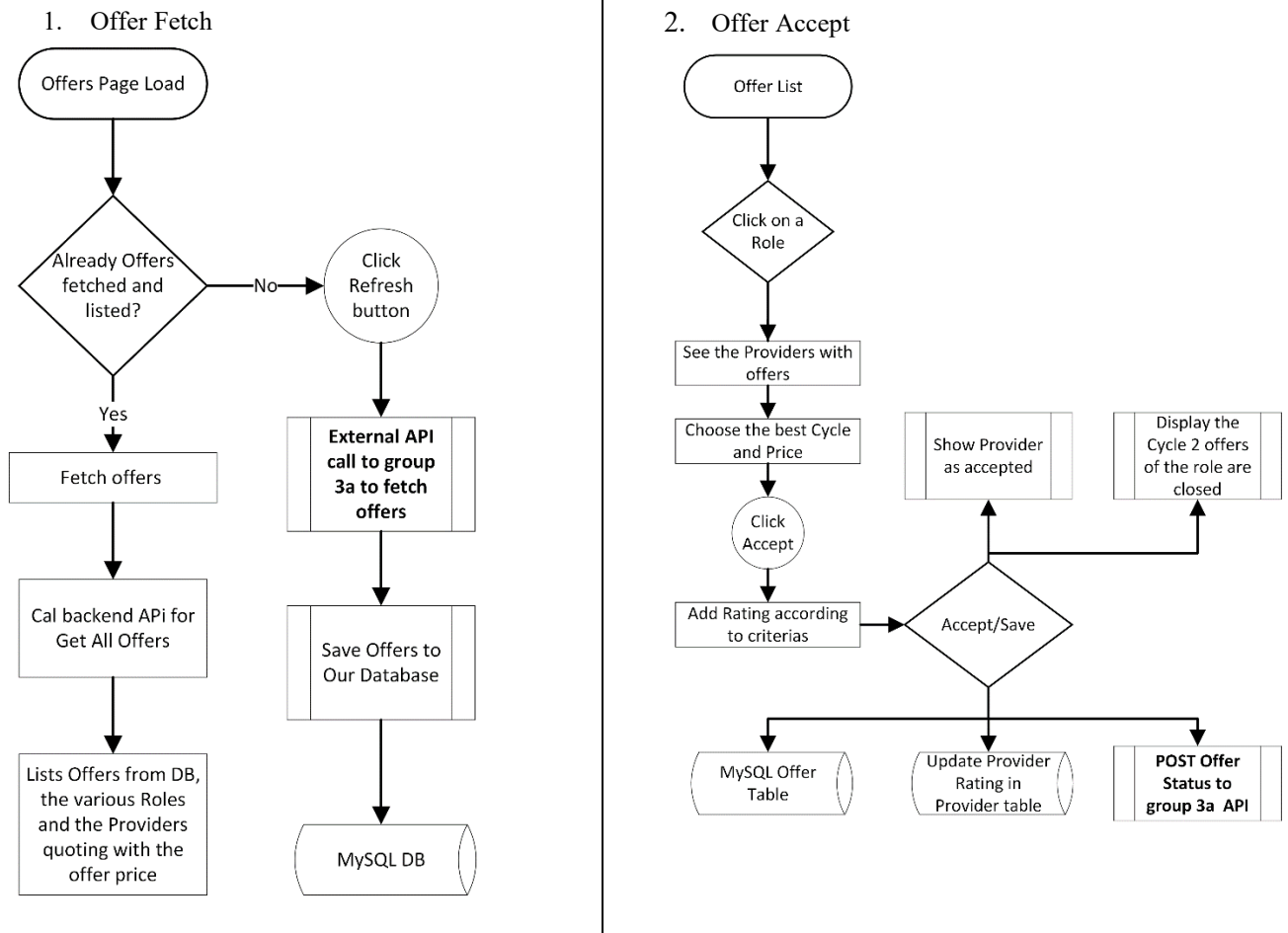
2. Find Provider by ID API



3. Get All Provider API:



Offer Workflow:



API Implementation

Master Agreement Type APIs:

The Main functionality our application is Master Agreement Type creation. The Backend APIs for Creation and manipulation for the Master Agreement is made namely, master agreement create, get all master agreement and modify master agreement.

- Master agreement POST API – Requires the mandatory fields like name, validFrom, validUntil, deadline, and the list of domains and specific roles in it
- Master agreement GET API: this helps the UI component to list out all agreement created.
- Master agreement GET by ID: Fetch details of a specific Agreement created.
- Master agreement PUT API: Modify defined Fields like name of the agreement, Validity dates and deadlines.

GET API

<http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/mastertype/all>

GET <http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/mastertype/all> Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

ody Cookies Headers (8) Test Results Status: 200 OK Time: 120 ms Size: 2.82 KB Save as example

Pretty Raw Preview Visualize

```
[{"masterAgreementTypeId":1,"masterAgreementTypeName":"MAT1","validFrom":"2024-09-15","validUntil":"2024-11-30",
"dailyRateIndicator":{"dailyRateIndicator1","deadline":"2024-11-30","teamDeadline":"2024-11-30",
"worksContractDeadline":"2024-11-30","isAccepted":"Accepted","providerNames":[{"provider_b"],"userName":null,"userType":null,
"domains":[{"id":1,"domainId":"4","domainName":"Operations","roles":[{"id":1,"roleName":"Service operator",
"experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","roleprice":null},{id":2,"roleName":"Infrastructure Engineer",
"experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null}]},"id":2,"domainId":"1",
"domainName":"DevAndConsult","roles":[{"id":3,"roleName":"Developer","experienceLevel":"INTERMEDIATE",
"technologiesCatalog":"Common","roleprice":null},{id":4,"roleName":"Solution Architect","experienceLevel":"ADVANCED",
"technologiesCatalog":"Rare","roleprice":null},{id":5,"roleName":"Test Engineer","experienceLevel":"JUNIOR",
"technologiesCatalog":"Uncommon","roleprice":null}]},"id":3,"domainId":"2","domainName":"Data","roles":[{"id":6,
"roleName":"Data Architect","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Uncommon","roleprice":null},{id":7,
"roleName":"Data Eng.", "experienceLevel":"JUNIOR","technologiesCatalog":"Common","roleprice":null},{id":8,"roleName":"Data
Analyst","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null}]}]},"masterAgreementTypeId":2,
"masterAgreementTypeName":"MAT2","validFrom":"2024-03-01","validUntil":"2024-06-01","dailyRateIndicator":{"dailyRateIndicator
2","deadline":"2024-06-01","teamDeadline":"2024-06-01","worksContractDeadline":"2024-06-01","isAccepted":"Accepted",
"providerNames":[{"provider_b"],"userName":null,"userType":null,"domains":[{"id":4,"domainId":"4","domainName":"Operations",
"roles":[{"id":9,"roleName":"DevOps Eng","experienceLevel":"JUNIOR","technologiesCatalog":"Common","roleprice":null},{id":10,
"roleName":"Service operator","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","roleprice":null},{id":11,
"roleName":"Infrastructure Engineer","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null}]},"id":5,
"domainId":"3","domainName":"Security","roles":[{"id":12,"roleName":"Cyber Security Engineer","experienceLevel":"INTERMEDIATE",
"technologiesCatalog":"Common","roleprice":null},{id":13,"roleName":"Information security manager",
```

POST API:

<http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/addmastertype>

GET <http://ec2-16-171-169-38.eu-north-1.compute.amazonaws.com:5000/api/mastertype/all> Save Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

ody Cookies Headers (8) Test Results Status: 200 OK Time: 76 ms Size: 3.69 KB Save as example

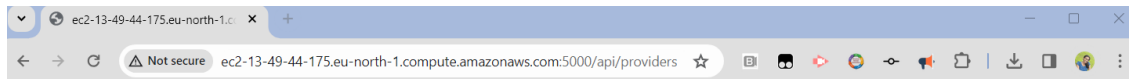
Pretty Raw Preview Visualize JSON

```
{
  "masterAgreementTypeId": 5,
  "masterAgreementTypeName": "MAT6",
  "validFrom": "2024-09-15",
  "validUntil": "2024-11-30",
  "dailyRateIndicator": "dailyRateIndicator6",
  "deadline": "2024-11-30",
  "teamDeadline": "2024-11-30",
  "worksContractDeadline": "2024-11-30",
  "domains": [
    {
      "id": 13,
      "domainId": "4",
      "domainName": "Operations",
      "roles": [
        {
          "id": 31,
          "roleName": "Service operator",
          "experienceLevel": "INTERMEDIATE",
          "technologiesCatalog": "Common"
        }
      ]
    }
  ]
}
```

Provider API

GET API:

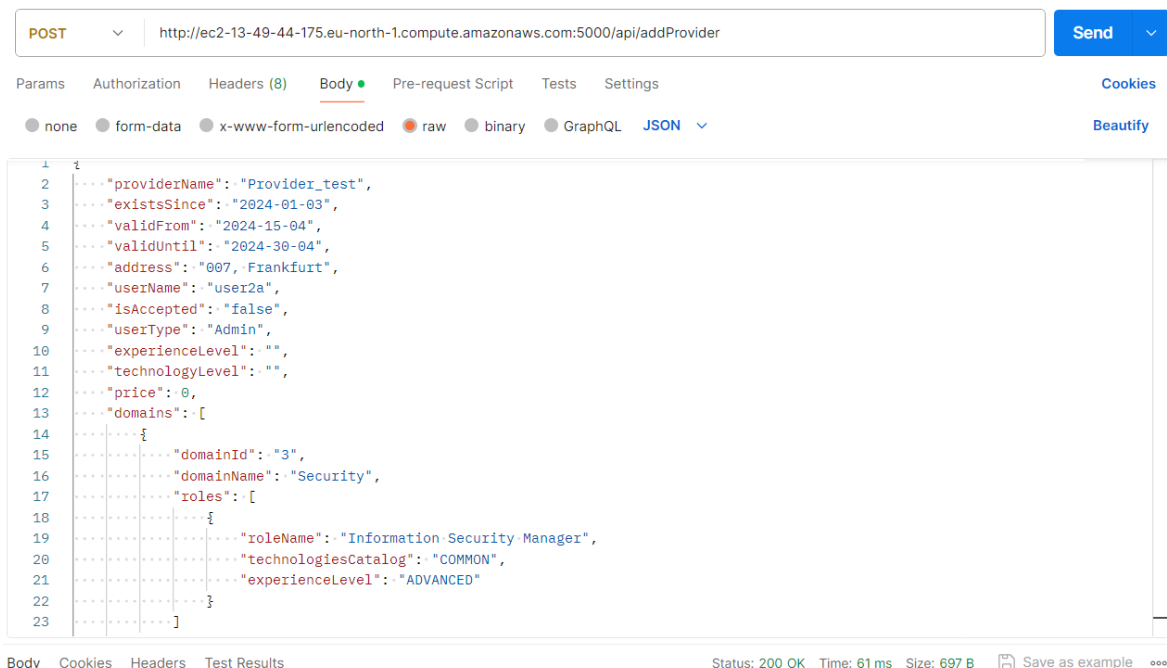
<http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/providers>



```
[{"providerId":1,"providerName":"provider_x","domains":[{"id":1,"domainId":"4","domainName":"Operations","roles":[{"id":1,"roleName":"Service operator","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","roleprice":null},{id:2,"roleName":"Infrastructure Engineer","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null}]}],"id":2,"domainId":"1","domainName":"DevAndConsult","roles":[{"id":3,"roleName":"Developer","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","roleprice":null},{id:4,"roleName":"Solution Architect","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null},{id:5,"roleName":"Test Engineer","experienceLevel":"JUNIOR","technologiesCatalog":"Uncommon","roleprice":null}]}],"address":"Stuttgart","existsSince":"2020-01-01","validFrom":"2024-01-01","validUntil":"2024-03-01","experienceLevel":"Level 1","technologyLevel":"Common","price":233.76,"providerRating":null,"isAccepted":null,"userName":null,"userType":null},{providerId":2,"providerName":"provider_y","domains":[{"id":3,"domainId":"2","domainName":"Data","roles":[{"id:6,"roleName":"Data Eng.","experienceLevel":"JUNIOR","technologiesCatalog":"Common","roleprice":null},{id:8,"roleName":"Data Analyst","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","roleprice":null}]}],"id":4,"domainId":"4","domainName":"Operations","roles":[{"id":9,"roleName":"DevOps Eng","experienceLevel":"JUNIOR","technologiesCatalog":"Common","roleprice":null},{id:10,"roleName":"Service operator","experienceLevel":"ADVANCED","technologiesCatalog":"Common","roleprice":null},{id:11,"roleName":"Infrastructure Engineer","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","roleprice":null}]}],"address":"Frankfurt","existsSince":"2020-01-01","validFrom":"2024-01-01","validUntil":"2024-03-01","experienceLevel":"Level 1","technologyLevel":"Common","price":233.76,"providerRating":null,"isAccepted":null,"userName":null,"userType":null},{providerId":52,"providerName":"Provider a","domains":[{"id":7,"domainId":"2","domainName":"Data","roles":[{"id":19,"roleName":"Data Scientist","experienceLevel":"Junior","technologiesCatalog":"Common","roleprice":null}]}],"address":null,"existsSince":null,"validFrom":null,"validUntil":null,"experienceLevel":"Junior","technologyLevel":"Common","price":0.0,"providerRating":null,"isAccepted":null,"userName":null,"userType":null},{providerId":102,"providerName":"Provider_pmp","domains":[{"id":8,"domainId":"2","domainName":"Data","roles":[{"id:20,"roleName":"Data Architect","experienceLevel":"INTERMEDIATE","technologiesCatalog":"UNCOMMON","roleprice":null}]}],"address":"123, Frankfurt, Germany","existsSince":"2024-27-01","validFrom":"2024-23-01","validUntil":"2024-23-01","experienceLevel":"","technologyLevel":"","price":0.0,"providerRating":null,"isAccepted":"false","userName":"pmp","userType":"Admin"}, {"providerId":103,"providerName":"provider_b","domains":[{"id":9,"domainId":"3","domainName":"Security","roles":[{"id":21,"roleName":"Cyber Security Engineer","experienceLevel":"INTERMEDIATE","technologiesCatalog":"COMMON","roleprice":null}]}],"address":"221B Baker Street","existsSince":"2024-02-02","validFrom":"2024-02-02","validUntil":"2024-02-06","experienceLevel":"","technologyLevel":"","price":0.0,"providerRating":"4","isAccepted":"accepted","userName":"user2a","userType":"Admin"}]
```

POST API:

<http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/addProvider>



POST <http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/addProvider> Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify

```
1 {
2   "providerName": "Provider_test",
3   "existsSince": "2024-01-03",
4   "validFrom": "2024-15-04",
5   "validUntil": "2024-30-04",
6   "address": "007, Frankfurt",
7   "userName": "user2a",
8   "isAccepted": "false",
9   "userType": "Admin",
10  "experienceLevel": "",
11  "technologyLevel": "",
12  "price": 0,
13  "domains": [
14    {
15      "domainId": "3",
16      "domainName": "Security",
17      "roles": [
18        {
19          "roleName": "Information Security Manager",
20          "technologiesCatalog": "COMMON",
21          "experienceLevel": "ADVANCED"
22        }
23      ]
24    }
25  ]
26 }
```

Body Cookies Headers Test Results Status: 200 OK Time: 61 ms Size: 697 B Save as example

Offers:

- For Offers are received from Group 3a by the backend for external API call to https://agiledev3a.pythonanywhere.com/p3aplatfrom/api/agreement_offers (Group 3a API)
- Offers are fetched and Saved to our database and listed on the offer page.
- Roles are quoted by different providers, and the user is displayed with automatically filtered best price offers for two cycles.
- The users' own experience (according to best price offer and best match) rating for the Providers.

- After establishing an offer made for a Role with a Master agreement, saved to DB as well as Saved to Group 3a
(https://agiledev3a.pythonanywhere.com/p3aplatform/api/post_ma_offer_response?offerId=2&isAccepted=True)

GET API:

<http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/getAlloffers>

GET <http://ec2-13-49-44-175.eu-north-1.compute.amazonaws.com:5000/api/getAlloffers> Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Description	...	Bulk Edit
Key	Value	Description		

ody Cookies Headers (8) Test Results Status: 200 OK Time: 218 ms Size: 2.54 KB Save as example

Pretty Raw Preview Visualize

```
[{"id":1,"roleName":"Data Architect","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Uncommon","domainId":2,"domainName":"Data","masterAgreementTypeId":1,"masterAgreementTypeName":"MAT1","provider":[{"id":72,"accepted":true,"offerId":"6","name":"provider_y","quotePrice":2000,"isAccepted":true,"cycle":1}],"id":2,"roleName":"DevOps Eng","experienceLevel":"JUNIOR","technologiesCatalog":"Common","domainId":4,"domainName":"Operations","masterAgreementTypeId":2,"masterAgreementTypeName":"MAT2","provider":[{"id":73,"accepted":true,"offerId":"3","name":"FRAUAS","quotePrice":2000,"isAccepted":true,"cycle":1}],"id":3,"roleName":"Infrastructure Engineer","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","domainId":4,"domainName":"Operations","masterAgreementTypeId":2,"masterAgreementTypeName":"MAT2","provider":[{"id":74,"accepted":true,"offerId":"5","name":"provider_y","quotePrice":4600,"isAccepted":true,"cycle":1}],"id":4,"roleName":"Service operator","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","domainId":4,"domainName":"Operations","masterAgreementTypeId":1,"masterAgreementTypeName":"MAT1","provider":[{"id":75,"offerId":"1","name":"FRAUAS","quotePrice":2500,"cycle":1}],"id":76,"offerId":"7","name":"provider_b","quotePrice":5000,"cycle":2}],"id":77,"accepted":true,"offerId":"4","name":"provider_y","quotePrice":2000,"isAccepted":true,"cycle":1}],"id":5,"roleName":"Solution Architect","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","domainId":1,"domainName":"DevAndConsult","masterAgreementTypeId":1,"masterAgreementTypeName":"MAT1","provider":[{"id":78,"accepted":true,"offerId":"2","name":"FRAUAS","quotePrice":5000,"isAccepted":true,"cycle":1}],"id":6,"roleName":"Cyber Security Engineer","experienceLevel":"INTERMEDIATE","technologiesCatalog":"Common","domainId":3,"domainName":"Security","masterAgreementTypeId":2,"masterAgreementTypeName":"MAT2","provider":[{"id":70,"accepted":true,"offerId":"8","name":"provider_b","quotePrice":4800,"isAccepted":true,"cycle":1}],"id":7,"roleName":"Data Analyst","experienceLevel":"ADVANCED","technologiesCatalog":"Rare","domainId":2,"domainName":"Data","masterAgreementTypeId":1,"masterAgreementTypeName":"MAT1","provider":[{"id":71,"accepted":true,"offerId":"9","name":"provider_b","quotePrice":3900,"isAccepted":true,"cycle":1}]}]
```

Database tables:

MySQL Workbench

agiletestdb x

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

Filter objects

agileproject1

- Tables
 - domain
 - domain_roles
 - master_table
 - offer_domain
 - offer_provider
 - offer_role
 - offer_role_provider
 - provider
 - provider_domains
 - provider_seq
 - role
 - standard_domains
 - standard_domains_standard_roles
 - standard_roles
- Views
- Stored Procedures
- Functions
- sys

Administration Schemas

Information

master_table

1 • SELECT * FROM agileproject1.master_table;

Result Grid

	master_agreement_type_id	dailyRateIndicator	deadline	master_agreement_type_name	teamdeadline	valid_from	valid_until	workcontractdeadline	offer_link
1		dailyRateIndicator 1	2024-03-01	MAT2	2024-03-01	2024-01-01	2024-03-01	2024-05-01	1
3		dailyRateIndicator 1	2024-03-01	MAT2	2024-03-01	2024-01-01	2024-03-01	2024-05-01	1

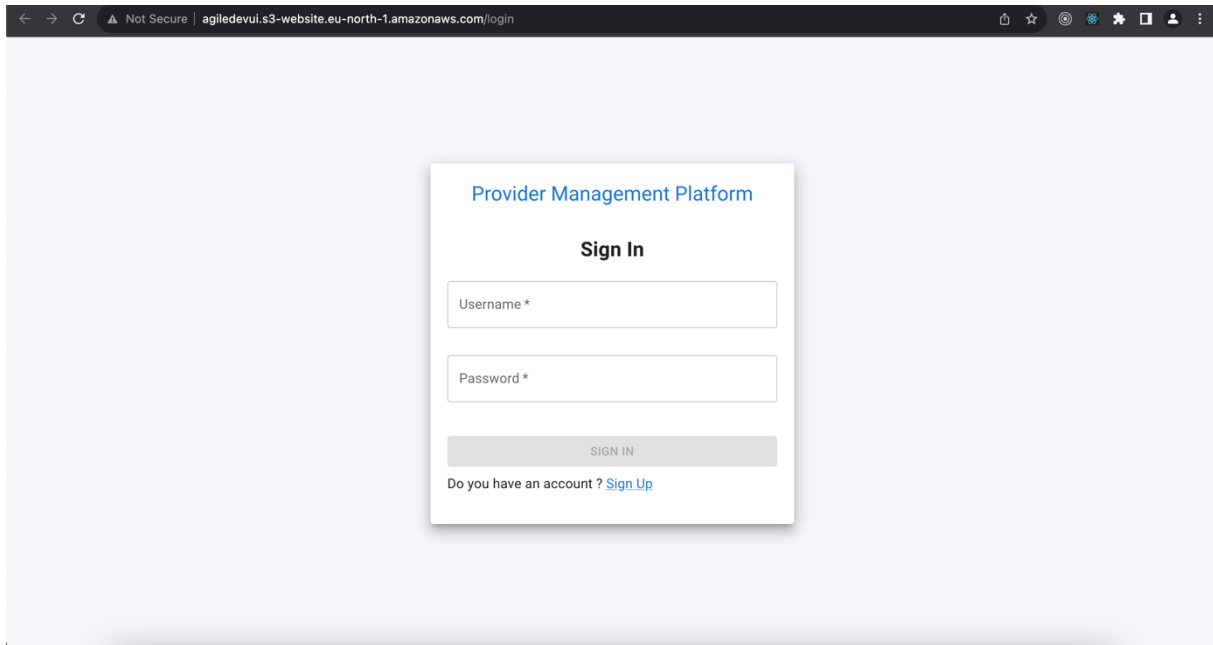
master_table 1 x

Apply Revert

Final Application Output

Login Page

The Authorized user can login providing the username and password which they provided during registration process.

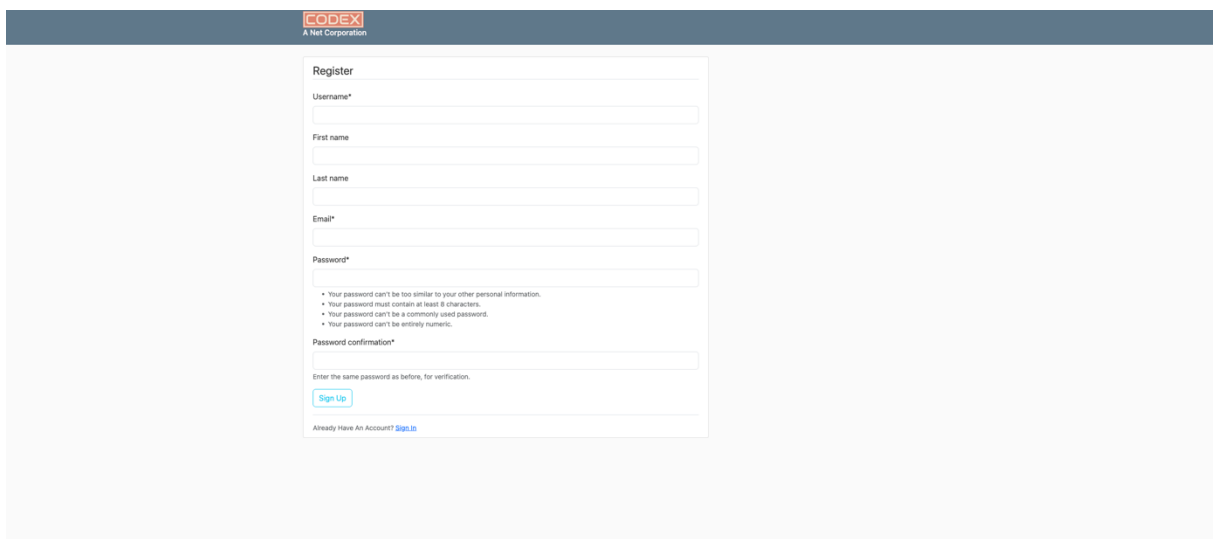


The screenshot shows a web browser window with the address bar displaying "Not Secure | agiledevui.s3-website.eu-north-1.amazonaws.com/login". The main content area features a light blue background with a central white card titled "Provider Management Platform" and "Sign In". The card contains two input fields: "Username *" and "Password *". Below these fields is a grey "SIGN IN" button. At the bottom of the card, it says "Do you have an account ? [Sign Up](#)".

Registration page

Users can register if they do not have an account by providing the basic information like first name, last name, username, email, and password etc.

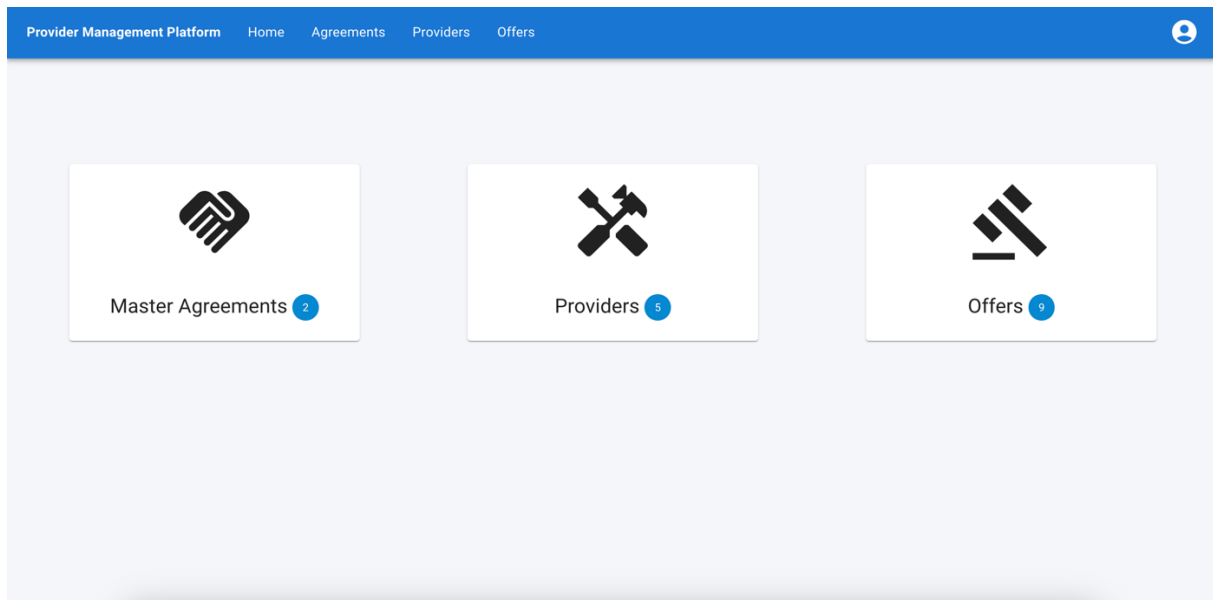
Note: We are using “Group 1” registration page.



The screenshot shows a web browser window with the address bar displaying "CODEX A Net Corporation". The main content area features a light blue background with a central white card titled "Register". The card contains several input fields: "Username*", "First name", "Last name", "Email*", "Password*", and "Password confirmation*". Below the "Password*" field, there are four bullet points: "Your password can't be too similar to your other personal information.", "Your password must contain at least 8 characters.", "Your password can't be a commonly used password.", and "Your password can't be entirely numeric.". Below the "Password confirmation*" field, it says "Enter the same password as before, for verification.". At the bottom of the card, there is a blue "Sign Up" button and a link "Already Have An Account? [Sign In](#)".

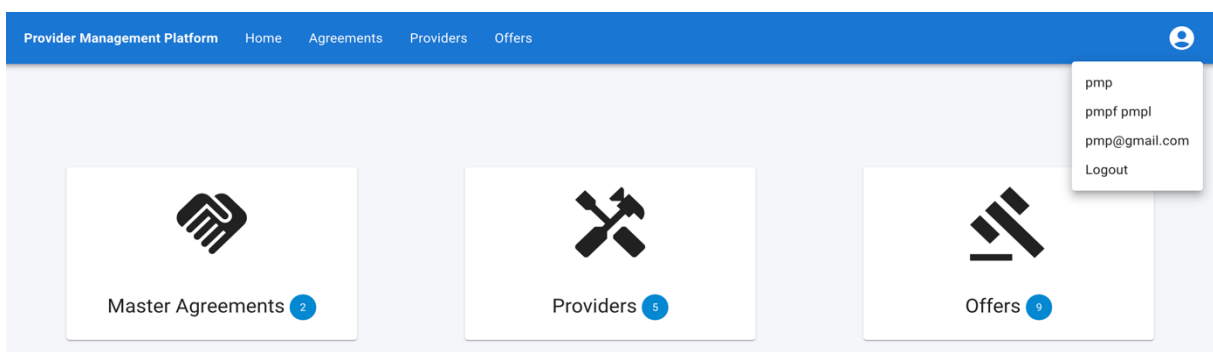
Provider Access Management Landing Page

The following page shows the landing page of our app. Once the user has logged in successfully, they will be redirected here. The page shows a simple dashboard. Each tile displays the total count of the available entities. Moreover, each tile is a link to its respective module.



Landing page with user menu

The landing page boasts a user-friendly dropdown menu, conveniently displaying key details such as first name, last name, username, and email for quick reference. Additionally, a prominent logout option has been integrated to facilitate smooth user interaction and enhance overall accessibility.



MasterAgreementType: Creation and Listing Page

The following pages shows all the Master Agreement list which includes fields master agreement id, agreement name, domain, valid from, valid until, daily rate indicator, deadline, team deadline and contract deadline. The results are paginated by default we show 10 rows.

Provider Management Platform

Home

Agreements

Providers

Offers

Master Agreements

+ CREATE MASTER AGREEMENT

ID	Agreement Name	Domains	Valid From	Valid Until	Daily Rate Indicator	Deadline	Team Deadline	Contract Deadline
1	MAT1	4-Operations						
		Service operator INTERMEDIATE Common Infrastructure Engineer ADVANCED Rare						
		1-DevAndConsult						
		Developer INTERMEDIATE Common Solution Architect ADVANCED Rare						
		Test Engineer JUNIOR Uncommon						
		2-Data						
		Data Architect INTERMEDIATE Uncommon Data Eng. JUNIOR Common						
		Data Analyst ADVANCED Rare						
		2024-09-15						
		2024-11-30						
dailyRateIndicator1								
2024-11-30								
2024-11-30								
2024-11-30								

Rows per page: 10

1-2 of 2

<

>

Master Agreement creation page

The Master Agreement creation page allows users to easily set up agreements by entering details like the agreement name, daily rate indicator, valid from, valid until, deadline, team deadline, work contract deadline and, domains and roles.

Create Master Agreement		X			
Master Agreement Name					
Daily Rate Indicator					
Valid From					
2024-26-01					
Valid Until					
2024-26-01					
Deadline					
2024-26-01					
Team Deadline					
2024-26-01					
Work Contract Deadline					
2024-26-01					
Domain and Roles					
SAVE		CANCEL			

Provider: Creation and Listing Page

The following pages shows all the Provider list which includes fields master provider id, provider name, domain, address, rating, valid from, valid until, and exists since. The results are paginated by default we show 10 rows.

Provider Management Platform							
Home Agreements <u>Providers</u> Offers							
Providers							
ID	Provider Name	Domains	Address	Rating	Valid From	Valid Until	Exists Since
52.00	Provider a	2-Data Data Scientist Junior Common		☆☆☆☆			
102.00	Provider_pmp	2-Data Data Architect INTERMEDIATE UNCOMMON	123, Frankfurt, Germany	☆☆☆☆	2024-23-01	2024-23-01	2024-27-01
103.00	provider_b	3-Security Cyber Security Engineer INTERMEDIATE COMMON	221B Baker Street	★★★★☆	2024-02-02	2024-02-06	2024-02-02
Rows per page: 10 1-5 of 5 < >							

Provider creation page

The provider creation page streamlines the addition of new providers by prompting users to enter key information such as provider name, domain, roles, address, and important dates like the start and end of validity and the establishment date.

Create Provider

Provider Name

Domain and Roles

Address

Valid From
2024-26-01

Valid Until
2024-26-01

Exist Since
2024-26-01

SAVE

CANCEL

Offers Page:

The offer page displays the offers provided by the Group 3A. Group 3A's offers are saved in our SQL database, which is finally displayed in our UI. The offers table displays a row for each role, domain, experience level, catalog, and master agreement. Under each row, we display all the offers provided by each provider. Offers are further grouped by cycles. From here, user could accept an offer. Once the offer is accepted, all other offers for that particular role, domain, experience level, catalog, and master agreement are marked as closed.

The offer page also has a refresh button which fetches the latest offers and updates the UI.

Provider Management Platform

Home

Agreements

Providers

Offers

Offers

REFRESH

ID	Role	Experience Level	Catalog	Domain	Master Agreement
1	Data Architect	INTERMEDIATE	Uncommon	Data	MAT1

Providers

Cycle 1

Id	Offer Id	Provider Name	Quote Price (EUR)	Actions
72	6	provider_y	2000	Accepted

2	DevOps Eng	JUNIOR	Common	Operations	MAT2
---	------------	--------	--------	------------	------

Rows per page: 101-7 of 7

Rating:

Ratings modal facilitates the user to rate a provider while he is accepting an offer. Ratings are based on the criteria mentioned below in the page. Saved ratings are displayed in provider’s page.

Provider Management Platform

Home

Agreements

Providers

Offers

Offers

REFRESH

ID	Role	Experience Level	Catalog	Domain	Master Agreement
Providers					
Cycle 1					
Id	Offer Id	Provider Name			Actions
157	6	provider_y			<div>ACCEPT</div>
Cycle 2					
Id	Offer Id	Provider Name	Quote Price (EUR)		Actions
156	8	Provider_pmp	5900		<div>ACCEPT</div>

Rows per page: 10 1-8 of 8

Please rate the provider provider_y

Please rate the provider based on

1. Best price offered

2. Maximum roles provided

ACCEPT

CANCEL

Conclusion:

In conclusion, agile Development in Cloud Computing Environments (Project) enabled us to work in teams like any professional environment. The Agile project for the Provider Management Component has successfully addressed the user requirements through a collaborative and iterative development approach. The cross-functional team of 6 students, guided by our professor, prioritized user stories, and implemented features in short, incremental cycles.

The project achieved key functionalities including user authentication, creation of master agreement types, providers, and material groups. Users can efficiently manage master agreements, open offers, and automatically filter the best price offers. The system facilitates the evaluation of providers based on a scale, allowing users to assess and specify measures for provider quality.

Throughout the project, close collaboration with the customer ensured that evolving requirements were accommodated, and feedback was incorporated promptly. The use of APIs enables seamless integration with other groups, promoting interoperability and scalability.

By adhering to these Agile best practices, the Provider Management Component project not only delivered a robust solution but also established a foundation for ongoing collaboration, adaptability, and extension of our minimum viable product to a full-scale project.

Delivered OUTPUT:

1. Fully functional Provider Management Component.
2. Two functional REST APIs (Provider & Master Agreement) for group 3a and 4a Delivered.
3. Cross Connectivity in using Group 1 Login/User Authentication API integration to UI login.
4. Use of Offers API from Group 3a to Fetch offers and later update status after offer establishment.

Reference

1. <https://mui.com/material-ui/>
2. <https://react.dev/learn>
3. <https://learn.microsoft.com/en-us/devops/plan/what-is-agile>
4. <https://vitejs.dev/guide/>
5. <https://spring.io/guides/gs/spring-boot/>
6. <https://spring.io/guides/gs/accessing-data-mysql/>
7. <https://dev.mysql.com/doc/>
8. https://docs.aws.amazon.com/ec2/?nc2=h_q1_doc_ec2
9. <https://github.com/git-guides/git-push>