**Introduction**  
As we embark on the development of our system, it has become evident that incorporating an Identity & Access Management (IAM) provider is essential to manage user-related functionalities. Our system requires robust authentication and authorization mechanisms to control access to specific parts of the front-end application. Additionally, it has been determined through discussions with stakeholders that the system should support multiple roles and organizations. Furthermore, during the design phase, stakeholders expressed the need for an on-premise authorization server within the microservice architecture. It is crucial to note that user data must remain anonymous to preserve privacy and security.

This research aims to identify the most suitable IAM provider. By examining various IAM providers, and looking at their capabilities, and their alignment with our requirements, we can make informed decisions that ensure

**Research Questions**  
The main research question of this research is: “How can we identify and choose the most suitable Identity & Access Management (IAM) provider that aligns with the project's security requirements?”

* What is an IAM-provider?
* What are the project's security requirements?
* Which IAM-providers are available on the market?
* Which IAM-providers are valid options that take the applications requirements into account?
* How can these IAM-providers add value to the project?
* Which IAM-provider is most suitable, taking the project's requirements into account?
* How will the chosen IAM-provider fit in with our software architecture?

**Research Strategies & Methods**  
To help perform practical research, I will use the DOT framework. These are the chosen strategies and methods:

1. **Library strategy**

A lot of base information will be gathered from the internet through google or community forums and documentation. Sources will be included in the research paper. I also want to interview a software expert (teacher) to talk about the research itself. For these reasons, literature study, community research and expert review are chosen as research methods.

1. **Workshop strategy**

Since this research will be practical and include a project demo, I will be performing prototyping as my research method. This prototype is the result of putting all the information that is gathered into practice.

**What is an IAM-provider?**

An IAM (Identity and Access Management) provider is a software service that facilitates management of users, authentication and authorization within an application. It is a set of tools that helps organizations securely control access to their resources. Typically, an IAM-provider has the following main functionality:

* User management
* Authentication
* Authorization
* Single Sign-On
* Monitoring & Logging

By using an IAM-provider or service, organizations can centralize the management of users, authentication and authorization processes.

**What are the project's security requirements?**

By planning meetings with our stakeholders, we got some clarification on the security requirements for the system. The following requirements have been mentioned:

* Must support user roles
* Must securely store users
* Multi-tenancy to support organizations
* Authentication / Authorization of users
* Support on-premise hosting
* Support for JSON Web Tokens
* Low cost / open-source / free

When looking for IAM-providers, we intend to look at these requirements and see which technology is the best fit for the project.

**Which IAM-providers are available on the market?**

* Google Cloud Identity Platorm (Cloud based)
* Amazon Cognito (Cloud Based)
* Okta (Cloud Based)
* Microsoft Active Directory (Cloud Based)
* Auth0 (Cloud Based)
* Firebase Authentication (Cloud Based)
* Keycloak
* Identity Server
* Gluu
* FusionAuth
* Springboot Security
* Ping Identity (Cloud Based)
* IBM Cloud App ID (Cloud Based)
* ForgeRock

**Which IAM-providers are valid options that take the applications requirements into account?**  
From the list of the found IAM-providers, several are strictly cloud based. This means that there is a vendor lock-in and will require additional work to move services to another provider if necessary. The stakeholders have also informed us that they want to be able to host the solution on-premise, which rules out the cloud based IAM-providers. The following providers are chosen for consideration, as they meet most of the company's security requirements:

1. Keycloak

Keycloak is an open source IAM solution that meets most of the project's requirements. It supports user roles and secure user storage. It also has support for multi-tenancy through the concept of realms, which is synonym for tenant in this case. It can be hosted on-premise if needed, avoiding vendor lock-in on the cloud. Keycloak is open-source and free to use without additional costs.

1. Identity Server

Identity Server is an OpenID Connect and OAuth 2.0 framework for ASP.NET Core. While it's built for ASP.NET applications, it can be used on multiple different platforms. It supports single sign-on, uses well defined authentication standards (OpenID and OAuth 2.0), and is highly customizable. Identity server is mostly focussed on authentication and authorization however, which means an extra service would be preferred for storing important user data.

* Gluu

Gluu is an IAM solution that allows you to define security details using their own low-code language: Agama. It offers single sign-on, identity management, authentication and authorization and much more. All of this is easy to configure using their custom low-code language. It also uses industry standards for authentication like OAuth 2.0 and is highly customizable.

* FusionAuth

FusionAuth is an open source authentication and authorization platform. It offers login, registration, single sign-on, multi-factor authentication and a bazillion (according to them) other features. It also offers user management with features like roles and tenants with groups. The service is open source and highly customizable. There is a free plan, but more features are available with a monthly payment plan.

**How can these IAM-providers add value to the project?**  
The IAM providers listed can provide functionalities like user management and authentication to our project. When a manager wants to create a survey, there has to be certainty from the system’s site that the manager is who he says he is so that they can create the survey. Otherwise, anyone could create surveys which would not make any sense. On top of that, it would mean a lot of extra data could be created and other unintended computing power could be wasted. User identity verification is also useful in other parts of the application, like when submitting responses. If there is no authentication, anyone could claim they made a response as someone else.

Another advantage that the IAM provider can give is more advanced user management based on groups. Whenever a manager creates a survey, their entire team is supposed to fill it in. With a well defined user management service, this process can be simplified. With these groups you have managers and members. If the manager creates a survey, the system can just look at their group’s members and add them to the survey.

**Which IAM-provider is most suitable, taking the project's requirements into account?**  
Looking at the requirements that have been set, and the options that have been chosen, we think that Keycloak can add the most value for us. Keycloak uses the concept of realms, which are essentially tenants that can be used for managing users for different organizations. These tenants are isolated from each other.

Keycloak also supports user roles out of the box, allowing you to define and manage these roles for your applications. Keycloak also provides a secure way to store users, ensuring that credentials and other senstive data is stored safely. There is support for multiple types of databases, LDAP protocol and other identity providers such as Active Directory or Google.

Another key factor is that Keycloak provides out-of-the-box authentication and authorization functionalities like user login, SSO, MFA and these features can be tuned to protect your resources based on user roles. The interaction between clients will work by leveraging JWT (JSON Web Tokens) to enable secure communication across different services in the system.

Keycloak is also easily deployed and hosted on-premise, allowing full control over the identity and access management. There are no additional costs to this, as Keycloak is open-source and free software which can be customized to fit the application needs.

By looking at the requirements that Keycloak meets, it is a strong choice for implementing a robust identity and access management solution.

**How will the chosen IAM-provider fit in with our software architecture?**  
Keycloak’s services can be utilized in various parts of our architecture.

For the front end we can secure pages so they can only be viewed by logged in members with certain roles. Pages like ‘create survey’ for example should only be accessible by people with the ‘Manager’ role. Personalized pages where the user can see all the answers, they have submitted should also have a authorization mechanism in place so that users only see the answers they submitted, and not someone else’s.

For the backend, we can secure the endpoints by validating the user’s identity using Keycloak with every incoming request. This way we can be sure that the incoming data is always valid, and no bad actor has interfered. For example, it shouldn’t be possible for someone to send a request with someone else their user id and have the system process it like normal. There should be some authentication mechanism in place that makes sure the incoming user id is valid. Keycloak does this by following the OAuth2.0 standard, and authenticating requests using token-based authentication. When a request is made, it sends a token with it in the authorization header. When the request is received, this token is received and checked with a secret to determine if the token is valid or not. If it's not valid, the request doesn’t get processed.

This process ensures that nobody outside of the application can make unauthorized requests. If someone were to make a request to the endpoint themselves using someone else’s user id and a fake token, the authorization mechanism in the backend recognizes it’s an unauthorized request and doesn’t let it through. The only way for an authorized request to get sent is through the normal application’s flow.

**Demo**  
To demonstrate the authentication of users with Keycloak, a demo is created to show the stakeholders what value Keycloak can bring to the project. Specifically, this demo shows the flow from the end-user's perspective.

The goal of this demo is to provide the stakeholders with context on how Keycloak can help solve the security challenges surrounding the project infrastructure. The demo will be discussed in the next sprint delivery.

**Environment Setup**

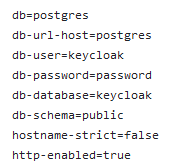
In order to host a Keycloak instance locally, we decided to use Docker as it makes the development environment easier to setup. To facilitate this, docker-compose is used to create the required services using only one Docker command:



In this file two services are created using the following images:

* Latest Keycloak image (the one on Docker Hub is outdated)
* Postgres database to persist user data from Keycloak

They are connected together using the environment variables, Keycloak uses an additional config file that will be loaded on startup.



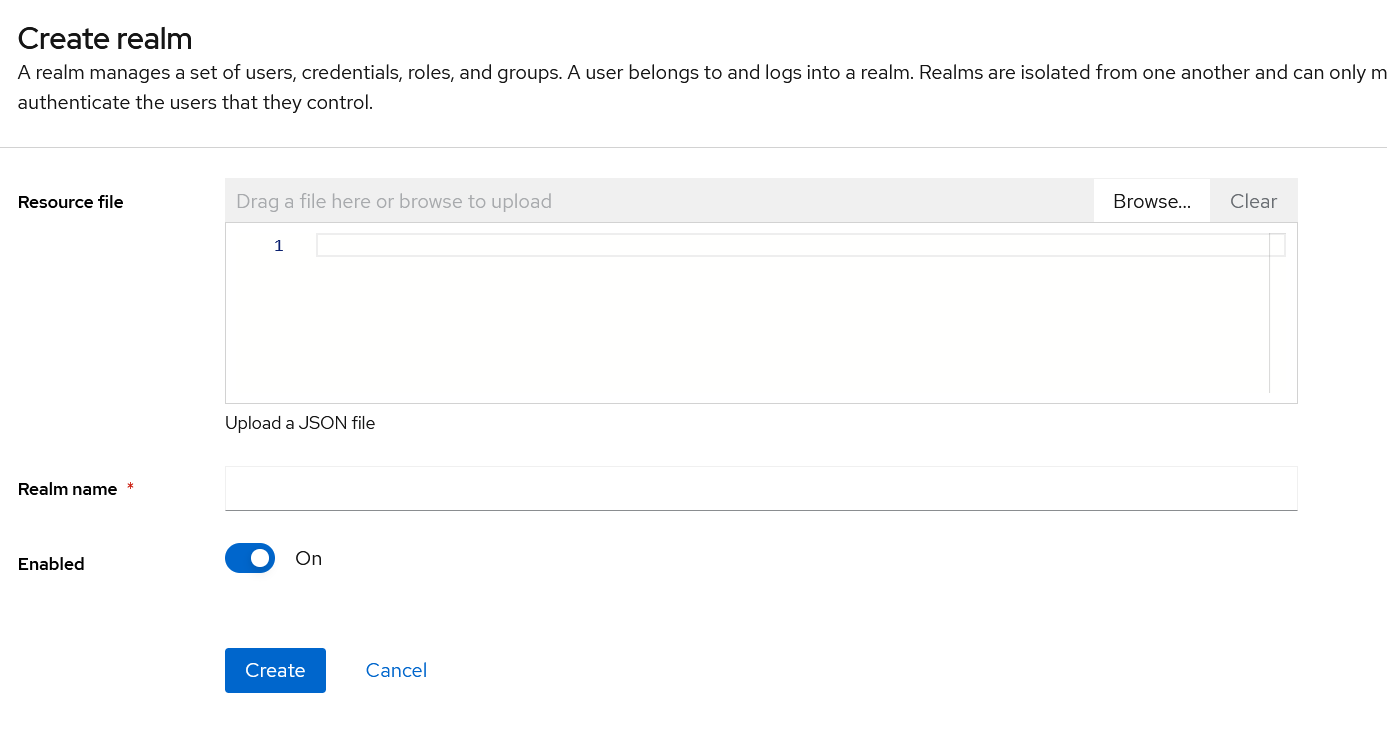
With the command: docker-compose up –d you can start these services at the same time. Keycloak will be automatically configured to use the postgres database, as shown in the image above.

**Configuration**

When the services are running, you can access the admin dashboard of Keycloak. In this example, the dashboard lives at localhost:8080. The login credentials are admin / admin as specified in the docker-compose file.

The next step is configuring a realm. A realm in Keycloak is synonymous with a tenant. You can see each realm as its own isolated environment that could belong to a different organization. In order to use Keycloak with other applications, several things need to be configured. To make this process easier, we have generated a JSON file that will import most of the import settings. This file is called “realm-export.json”.

In order to import this configuration file, it is required to create a new realm. Within the setup screen of a new realm, you have the option to import a .JSON configuration file:



This configuration file does not import any existing users. For this reason, these have to be created manually for testing. In our demo we simulated 3 roles, which are the employee, manager and researcher. Every user has been created in the same way. When a user is created, it can be added to a group. A group can hold roles within a realm. The role of the user will be included in the access token, which is required to access certain resources on the front-end and back-end.

**Front-end**

For setting up the front-end with keycloak, we created a separate setup guide that explains this process in more detail. The Angular front-end application uses the keycloak-angular and keycloak-js libraries to connect to the Keycloak instance. It makes heavy use of an authentication guard to protect certain resources. It also adds the access token obtained from keycloak to every HTTP request using the Angular HTTP client.

**Conclusion**  
After evaluating the requirements and considering the capabilities of the technologies, Keycloak stands out as the most suitable choice. Keycloak provides support for user rules, secure user storage, multi-tenancy, authentication and authorization, on-premise hosting and is low in cost because its free to use and open source.

The other contenders such as Identity Server, Gluu and FusionAuth offer similar functionalities, but Keycloak still comes up on top because it is easy to set up and use. It is clear that it should be the recommended option due to its ability to fulfill all the specified requirements effectively while providing a low-cost solution for identity and access management.