

# SkillSeeker Documentation

Salih Ekici

Mediaan Internship Project

2023-2024

Abstract .....	4
Functional Documentation .....	5
Requirements.....	5
Use Case Diagram .....	6
Technical Documentation .....	7
Cloud Architecture .....	7
Frontend .....	12
Setup.....	12
Pipeline.....	13
Structure.....	15
Components .....	15
Models .....	15
Service .....	15
Assets .....	15
Environments.....	15
Final Product.....	16
Login page .....	16
Employee list page .....	17
Searching for Employees by Name .....	17
Searching for Employees by Skill Set.....	18
Employee profile page .....	19
Editing Profile Details .....	20
Managing Skills .....	20
Project list page.....	21
Viewing Projects.....	21
Filtering Projects .....	22
Navigation Options.....	22
Project manager page – add/edit project .....	23
Adding/Editing Project Details .....	23

Soft Deleting (Archiving) Projects.....	24
Employee assign page .....	25
Viewing Assigned Employees .....	25
Managing Assignments .....	26
Viewing and Filtering Unassigned Employees .....	26
Skill manager page .....	27
Adding New Skills .....	27
Exploring Existing Skills and Related Skills .....	28
Skill manager – View similar skills of a skill .....	29
Assigning New Similar Skills .....	29
Backend.....	30
Technologies.....	30
Setup.....	30
Folder Structure .....	31
Pipeline.....	32
Database .....	34
What is a Graph Database.....	34
Setup.....	35
Conclusion.....	36

## Abstract

Mediaan, an IT consultancy firm recently integrated into the Conclusion group, faced challenges in efficiently assigning employees to projects due to the manual process of sifting through numerous CVs and matching required skills. To address this, a solution named SkillSeeker was developed. This application simplifies employee management by allowing sales teams to enter project requirements and receive recommendations for the best-suited employees based on their skills.

The application employs a graph database, Neo4j, to effectively manage and query the complex relationships between skills, enabling it to suggest related competencies when necessary (e.g., recommending C# developers for Java projects). The backend is built with Java Spring Boot, and the frontend uses Angular, both hosted on Azure's serverless container apps for optimal scalability and maintenance.

SkillSeeker not only enhances the speed and accuracy of employee assignments but also aligns with Mediaan's goal of leveraging advanced technology to improve business processes. This modernization ensures that Mediaan remains competitive in the IT consultancy industry by optimizing their project staffing workflow.

## Functional Documentation

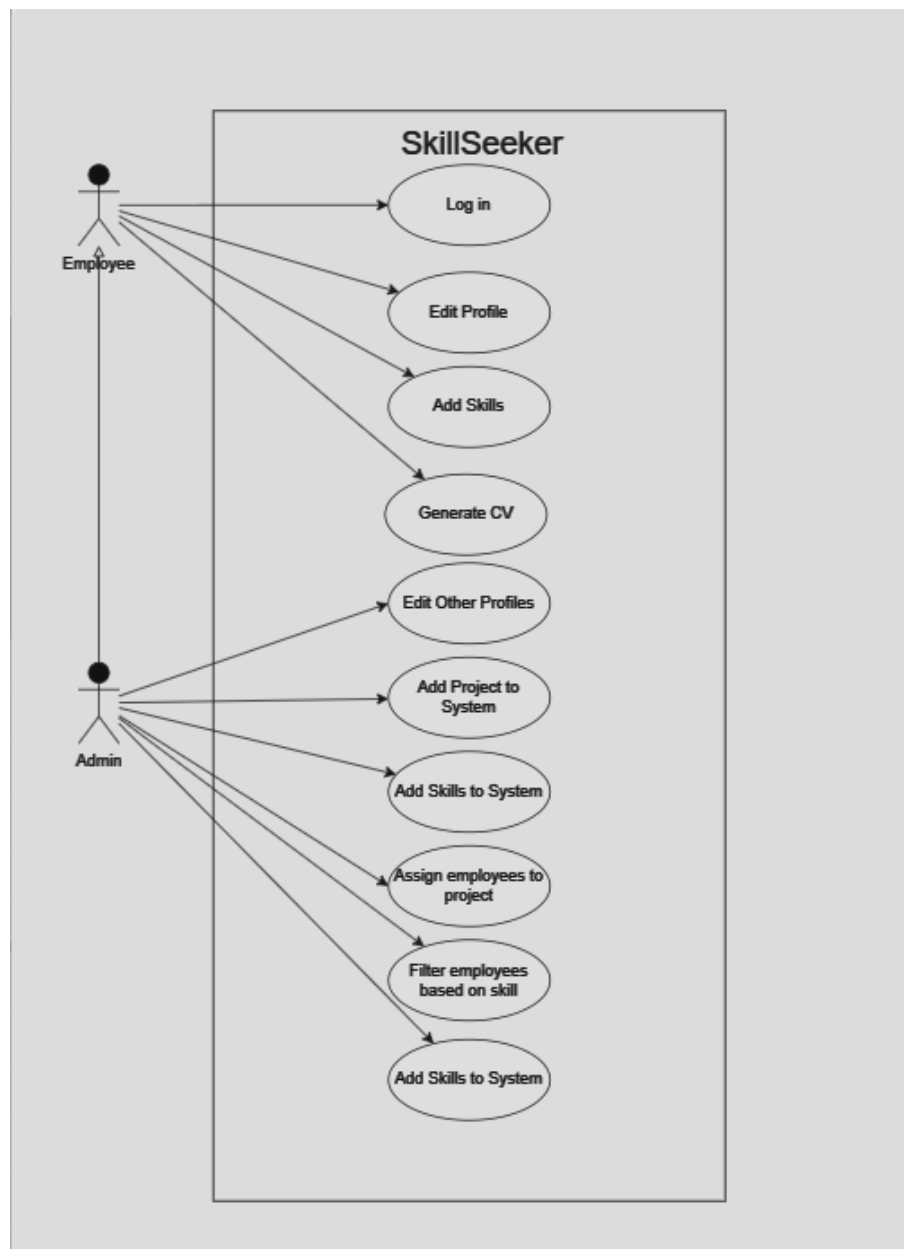
This part of the documentation includes the project's requirements.

### Requirements

Table of features the application must, should and could have:

Must Have	Should Have	Could Have
Admin can filter employee on skill.	Admin can add projects.	Active Directory login.
Admin can filter employee on name.	Admin can edit projects.	Admin can alter the relationship between skills.
Admin can edit the info from other employees.	Admin can filter projects based on active state. This state is determined by end and start date.	Admin can add new skills.
Employee and Admin can edit own information	Admin can view project details.	Admin can download an employee CV.
		Employees can upload their CV.
		Downloaded CV is dynamic and alters based on needs.
		Admin can filter employees based on if they are available.

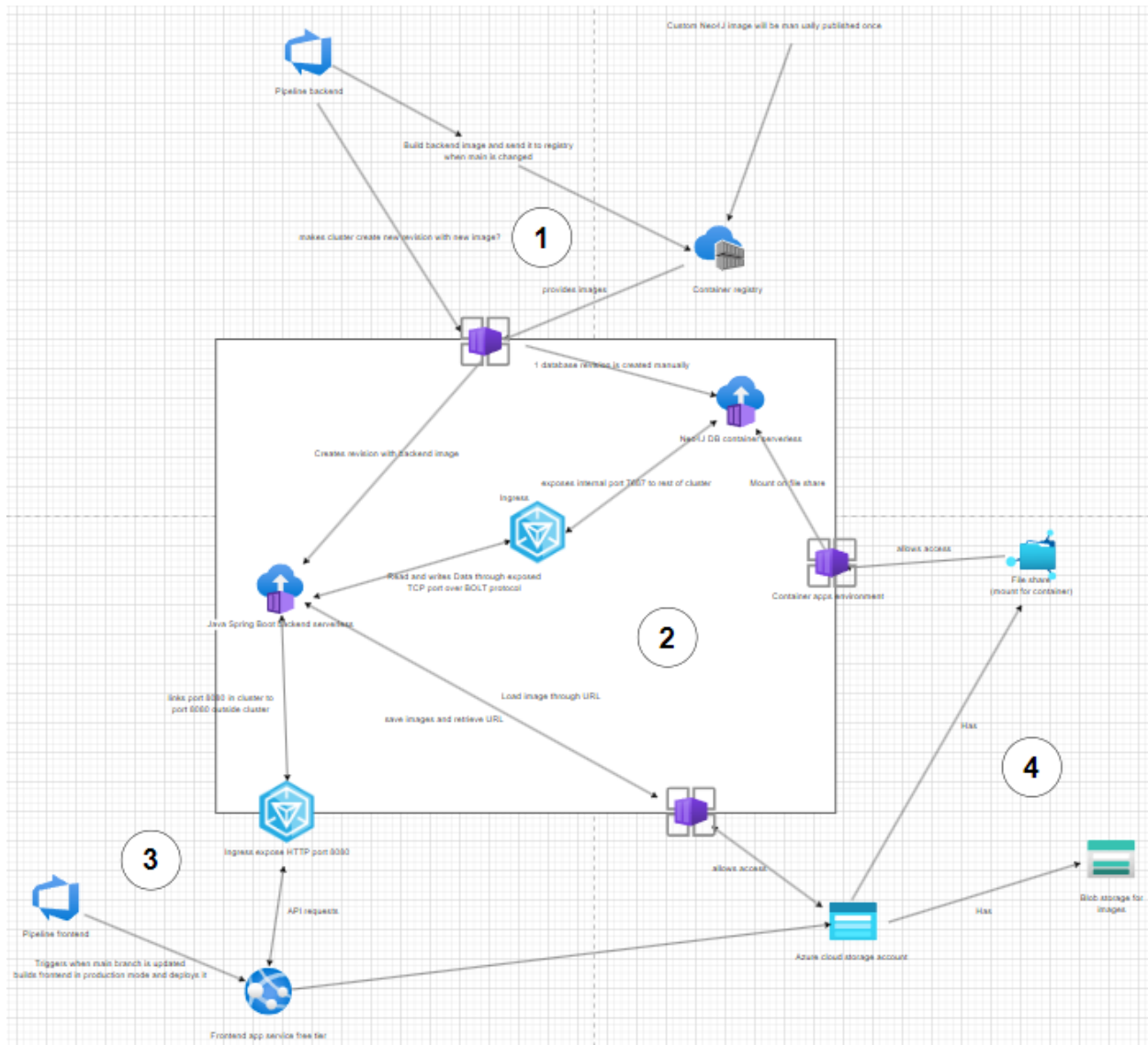
## Use Case Diagram



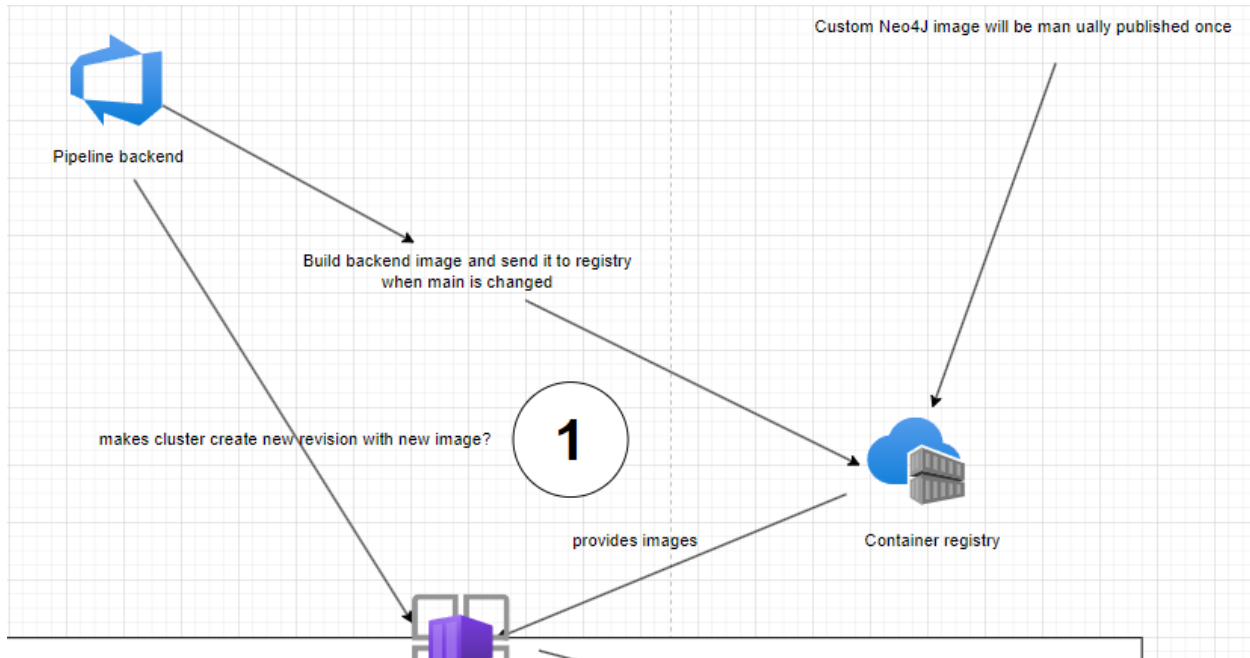
# Technical Documentation

This portion of the documentation includes the following information regarding the cloud architecture, backend, and frontend technologies with also screenshots of the latest version of the application.

## Cloud Architecture



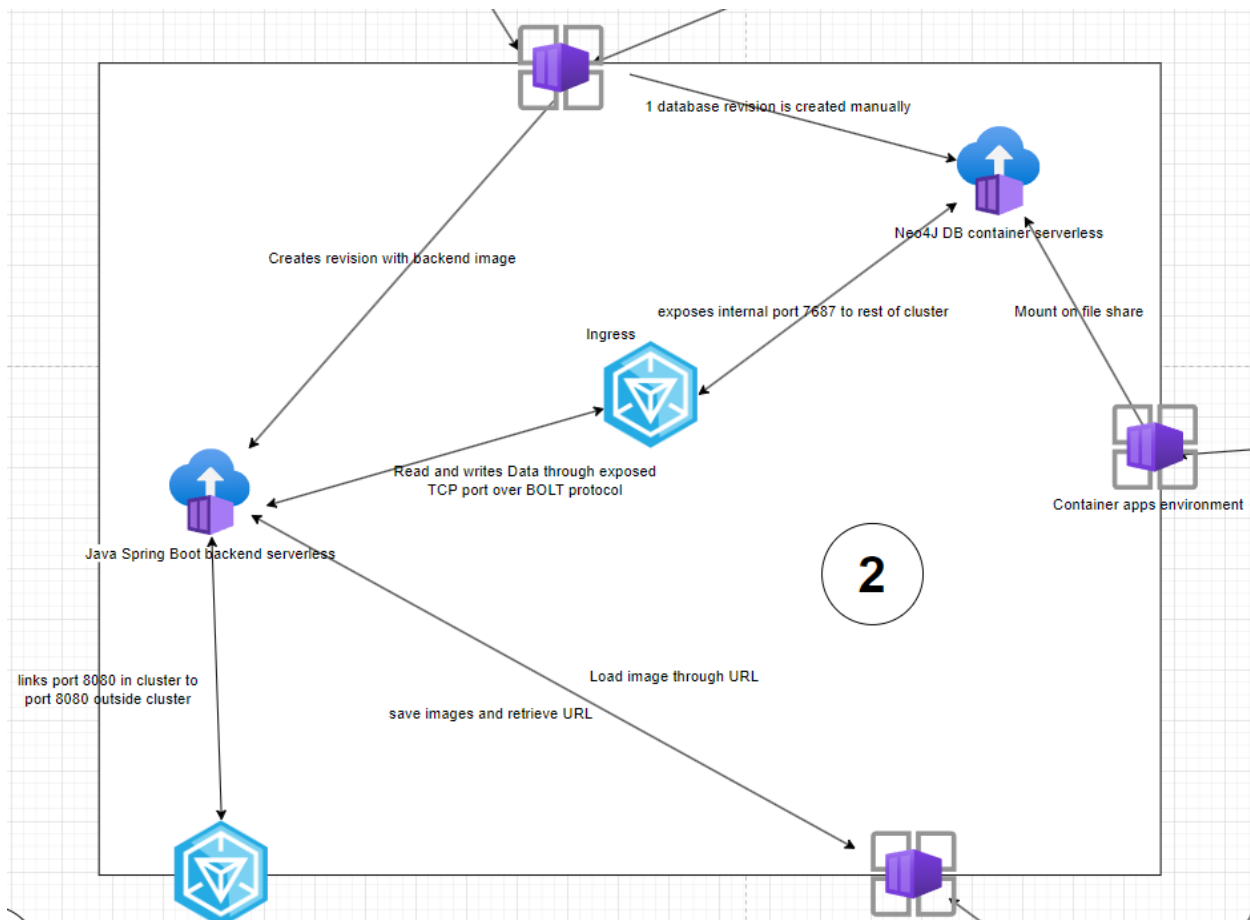
The image above provides an overview of the cloud infrastructure utilized for deploying SkillSeeker. A detailed explanation of each section is provided below.



The first part illustrates the provisioning of container images. The Azure Container Registry stores an image for the backend and one for the database. The Neo4j database image, including the APOC plugin, is manually added to the Registry, as it does not require subsequent updates.

The backend image is generated through a pipeline. When changes are made to the main branch, the pipeline builds the Java code using Maven, creates a new image, and uploads it to the Registry. Upon uploading the image, the pipeline instructs the Container Apps Environment to create a new revision with the updated image.

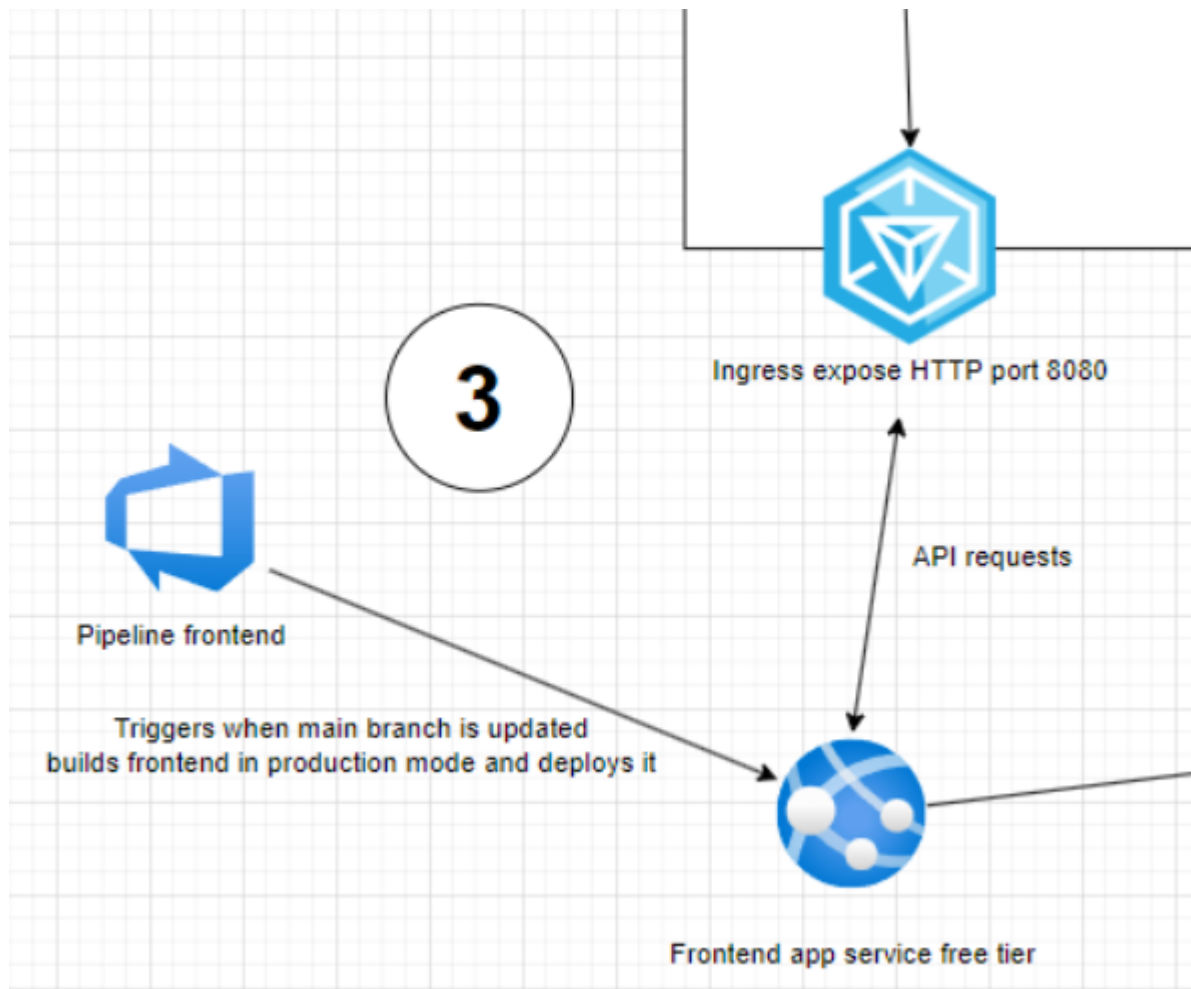




The second part of the infrastructure comprises the Container Apps Environment. Although this environment operates within a Kubernetes cluster, the use of Azure Container Apps obviates the need for direct cluster management. Within the cluster, there are two container apps: one for the backend and one for the database.

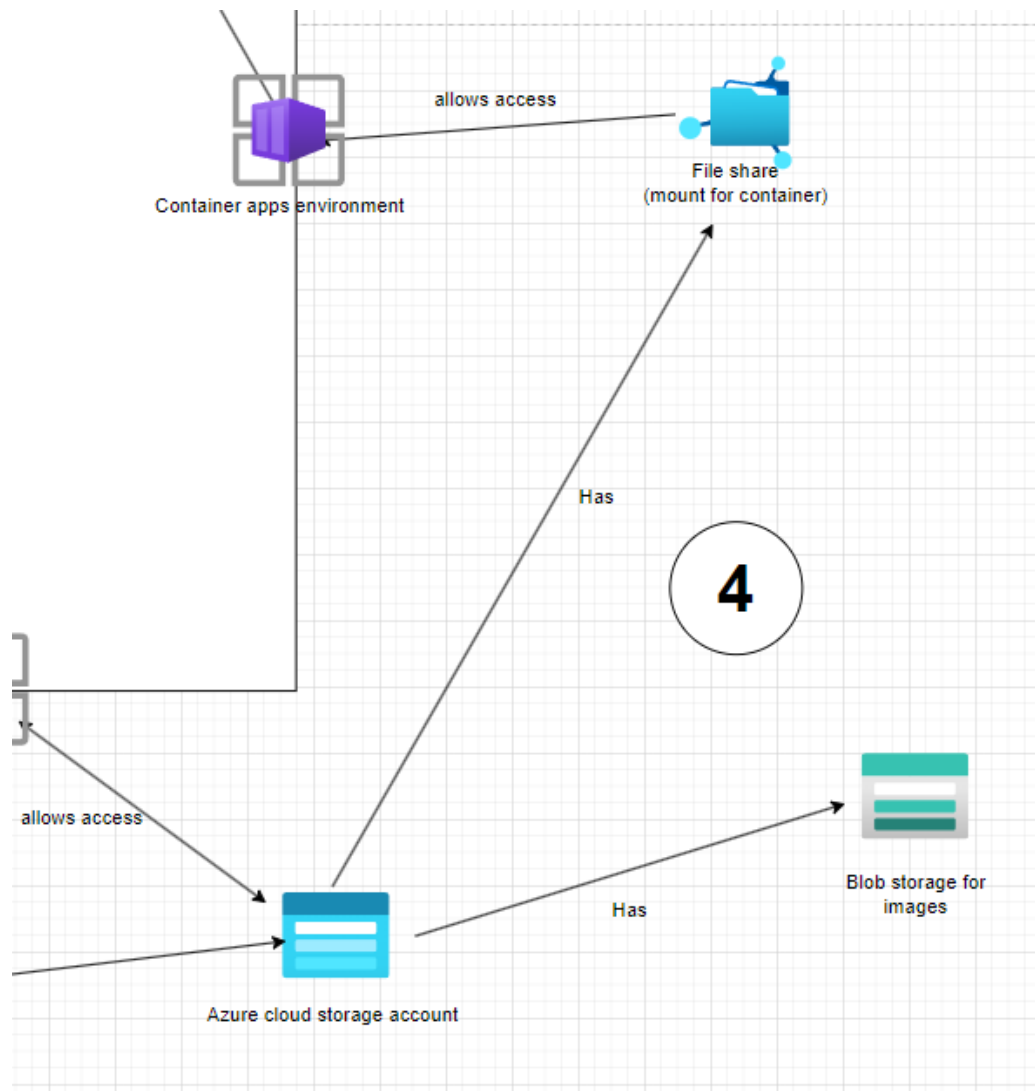
The decision to use container apps stems from the application's infrequent usage, typically by a single user at a time, making it resource- and cost-inefficient to keep the application running continuously. SkillSeeker prioritizes cost efficiency over optimal functionality, acknowledging the significant load times associated with starting Container Apps.

The backend is hosted in a Container App rather than an Azure Web App because TCP communication is necessary, given that the database uses the Bolt protocol to receive requests from the backend. TCP communication is only possible within the cluster environment.



The third part of the infrastructure details the frontend of the SkillSeeker application. The frontend is hosted on an Azure Web App using the free tier, which is sufficient for SkillSeeker's basic requirements.

A pipeline builds the Angular code in production mode and deploys it to the Web App. This pipeline is triggered whenever the backend pipeline completes successfully. This approach ensures that if the frontend receives the latest changes, which might include new pages requiring additional backend APIs, the backend build must succeed to prevent users from encountering non-functional buttons. Conversely, a successful backend update and a failed frontend update would not cause issues, as users would not be aware of new features if they are not visible on the frontend.



The final part of the infrastructure provides details on data storage. Storing data inside the containers is not feasible as it gets wiped whenever the container scales down to zero or turns off. Therefore, persistent storage is necessary, achieved by mounting the container to a volume.

For persistent storage, an Azure File Share is created within our Azure cloud storage account. This file share is mounted on the database to ensure data persistence.

Additionally, the account includes Blob storage, intended for storing user profile pictures. When the backend stores an image in the Blob storage, it returns the image's URL, which is then added to the corresponding Neo4j object to load the image. However, this image functionality is currently conceptual and not implemented in the application.

## Frontend

This part of the documentation includes information regarding the frontend.

### Setup

To start the application locally, use these steps:

1. First step to run the frontend locally is make sure node is installed on your device with the following link:  
<https://nodejs.org/en/download>
2. To set up the frontend, clone the repository and navigate inside the SkillSeeker-Frontend/skillseeker folder.
3. After node is installed, use following command to install the Angular CLI globally:

```
npm install -g @angular/cli
```

4. Next use following command to install all the required packages:

```
npm install
```

5. Lastly use one of these commands to start the application:

```
ng serve || npm start
```

6. To run the tests with coverage, simply run one of these commands:

```
npm test || ng serve --code-coverage
```

## Pipeline

A **pipeline** in technology is a way to automate a series of steps that need to happen in a specific order to achieve a result.

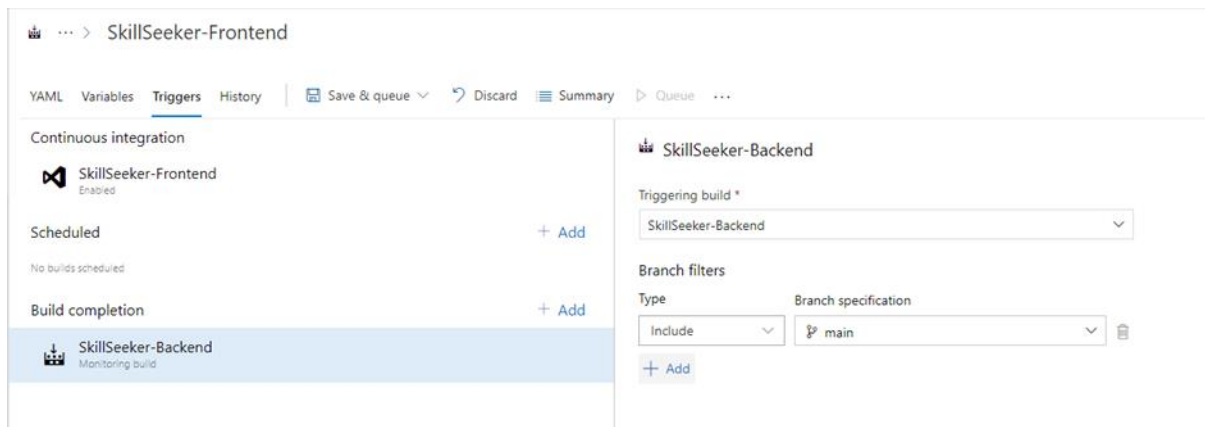
```
1 pool:
2   vmImage: ubuntu-latest
3
4 steps:
5   - task: NodeTool@0
6     inputs:
7       versionSpec: '20.x'
8     displayName: 'Install Node.js'
9
10  - script: |
11      cd SkillSeeker
12      npm install -g @angular/cli
13      npm install
14      ng build --configuration production
15    displayName: 'npm install and build'
16
17  - task: AzureWebApp@1
18    inputs:
19      azureSubscription: 'Interns BE'
20      appName: 'SkillSeeker-Frontend'
21      package: '$(System.DefaultWorkingDirectory)/SkillSeeker/dist/skill-seeker/browser'
22      deploymentMethod: 'zipDeploy'
23      appType: 'webApp'
24      resourceGroupName: 'TE_Interns_Belgie'
25    displayName: 'Deploy to Azure Web App'
26
```

As shown in the image above, the pipeline starts by specifying that the code be run on an Ubuntu system with the latest version.

Next, Node.js is installed as a package manager on the system. Then the pipeline navigates to the application and uses the package manager to install the Angular command-line interface. After this command-line is installed, the remaining packages can be installed.

Finally, this part of the pipeline builds the application with the “ng build” command and specifies that it should be built for production, so the application uses different settings. The frontend application contains two files with settings, one for production and one for development. So, in the pipeline, the correct file must be chosen. This command results in a folder called “browser” that contains all the files needed to run the application.

The pipeline's second task is to send this folder's contents to the App Service in the Azure Portal. Authentication is performed using a service connection created for the pipeline.



The screenshot above illustrates the trigger that activates the pipeline. The pipeline starts each time the backend pipeline completes successfully. The logic behind this is that if both the frontend and backend pipelines are started at the same time, but the backend pipeline fails, users may see new frontend features that then cause an error message because the backend endpoint is not up to date. Therefore, the backend should always be up to date before the frontend is updated.

## Structure

We selected Angular 17 for the frontend of our application due to its modern framework architecture and widespread adoption in numerous projects. This choice was influenced by our preference for Angular and its extensive documentation and libraries, which augment its already robust functionality.

Despite our prior experience with Angular 16, we opted for Angular 17 to adhere to best practices of maintaining applications with the latest versions for enhanced security and maintainability. Upgrading ensures access to new features and improvements that contribute to the overall performance and longevity of our application.

For our package manager, we chose to go with node twenty which was the latest version at the time of creating the project. We decided to go with a developer friendly folder structure using components, models, services, and assets.

### *Components*

The component folder contains all the pages along with the functionalities that the page needs. Also, the test of each individual component is visible within the components folder.

### *Models*

The model's folder is where we define how objects will look and what properties they have. For example, we define that an employee object has the properties first name, last name etcetera.

### *Service*

In the service folder we define service classes that do most of the complex logic and all the communication with the backend API.

### *Assets*

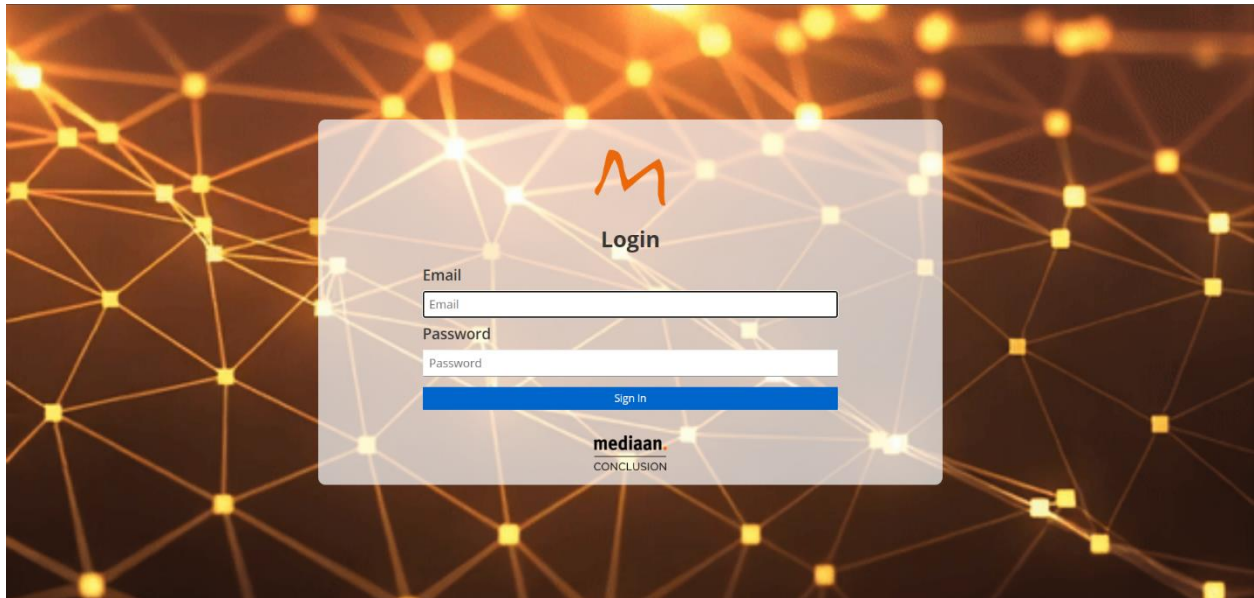
The asset folder contains all the images that are used inside the application.

### *Environments*

This folder contains the environments for development and production. Environments are used to manage different deployment stages. By using environments, we can easily switch between different configurations without having to manually change settings in our code.

## Final Product

### *Login page*



This is the login page that is generated with Keycloak. You have to login with a user that is known in the realm of Keycloak. In the main branch of the application this page will not be visible due to Keycloak not being deployed. Instead, you get redirected to the employee list page after login in with your own account of Mediaan.



## Employee list page

Name	Skills	Working for	Total hours per week
Robin Lux	Python: 6 Java: 7 CSharp: 5	UrbanTech Corp, SunPower Belgium, Eco Innovations	40
Samuel Wouters	Nagios: 6 Puppet: 5 Docker: 7	UrbanTech Corp, SunPower Belgium, Eco Innovations	40
Salih Ekici	Gradle: 5 Python: 6 Heroku: 7	HealthTech NV, Eco Innovations	40
Zoe Viola	CSS: 6 Arduino: 5 Flutter: 7	HealthTech NV, TechSolutions Ltd.	40
Anna Janssens	Javascript: 6 CSS: 7 HTML5: 8	HealthTech NV, SunPower Belgium, TechSolutions Ltd.	40
Lucas Vermeulen	CSS: 6 Python: 5 HTML5: 7	GreenTransit BVBA	40
Emma Lefevre	Python: 8 CSS: 7 Java: 6	SecureIT Solutions, GreenTransit BVBA	40
Max Garcia	CSS: 6 HTML5: 7 Python: 5	SecureIT Solutions, GreenTransit BVBA	40

This page is designed to empower the sales manager with the capability to efficiently search for employees based on either their name or specific skill sets. This dual-search functionality ensures that the sales manager can quickly find the right personnel for any given task or requirement.

### Searching for Employees by Name

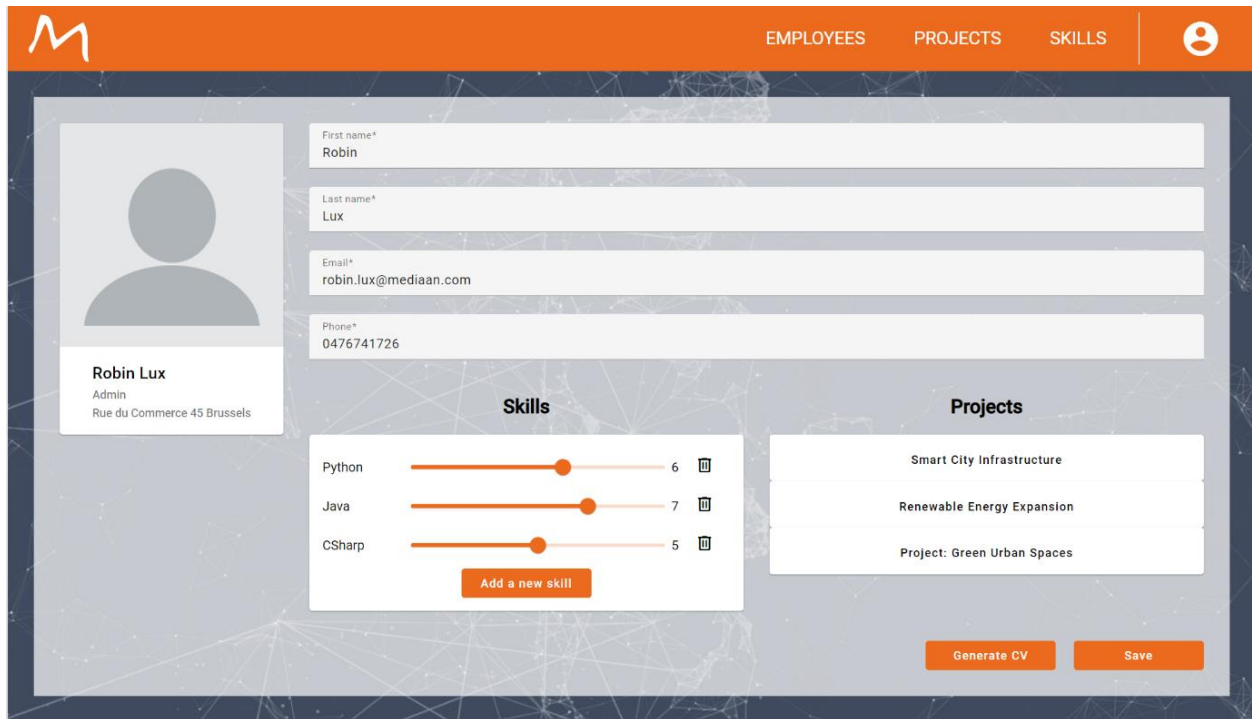
The first search option allows the sales manager to locate an employee by entering their name. This feature is particularly useful when the manager knows exactly which employee they are looking for. By simply typing in the name, the system will retrieve and display the employee's details, including their contact information, current role, and availability. This straightforward search method saves time and simplifies finding and managing employees individually.

### Searching for Employees by Skill Set

The second, more advanced search functionality enables the sales manager to search for employees based on specific skill sets. This feature is designed to identify employees whose skills match the requirements of a particular task or project. When using this search option, the system will:

- **Input the Skill Set:** The sales manager can enter one or more skills required for the job. These skills can range from technical abilities, such as proficiency in a software tool, to soft skills, like team leadership or customer communication.
- **Ranking System:** Upon initiating the search, the system analyzes the employees' skills and ranks them based on how well their skills match the input criteria. This not only ranks employees who have the skills but also the ones that have the potential to learn the skills quickly. This ranking is done from best to worst, ensuring that the most qualified employees appear at the top of the list.
- **Display Results:** The results are displayed in an organized list, showing the employees in descending order of suitability for the specified skill set. Each employee's profile in the list includes their skill ratings, experience level, and other relevant information that can help the sales manager make an informed decision.

## Employee profile page



The image shows a web interface for an employee profile. At the top is an orange navigation bar with a logo 'M' on the left and links for 'EMPLOYEES', 'PROJECTS', and 'SKILLS' on the right, followed by a user icon. The main content area has a dark blue background with a white grid pattern. On the left, there's a profile card for 'Robin Lux', 'Admin', with the address 'Rue du Commerce 45 Brussels'. To the right of the card are four input fields: 'First name\*' (Robin), 'Last name\*' (Lux), 'Email\*' (robin.lux@mediaan.com), and 'Phone\*' (0476741726). Below these is a 'Skills' section with three rows: 'Python' with a slider at 6, 'Java' with a slider at 7, and 'CSharp' with a slider at 5. Each row has a trash icon and an 'Add a new skill' button at the bottom. To the right of the skills is a 'Projects' section with three rows: 'Smart City Infrastructure', 'Renewable Energy Expansion', and 'Project: Green Urban Spaces'. At the bottom right are two orange buttons: 'Generate CV' and 'Save'.

Field	Value
First name*	Robin
Last name*	Lux
Email*	robin.lux@mediaan.com
Phone*	0476741726

Skill	Level	Action
Python	6	Trash
Java	7	Trash
CSharp	5	Trash

Project
Smart City Infrastructure
Renewable Energy Expansion
Project: Green Urban Spaces

The employee profile page provides a user-friendly interface where employees can manage and update their personal and professional information. This page is a central hub for employees to keep their profiles current, ensuring that their skills and details accurately reflect their capabilities and achievements.

## Editing Profile Details

Employees can edit various aspects of their profile directly from the employee profile page. The editable sections include:

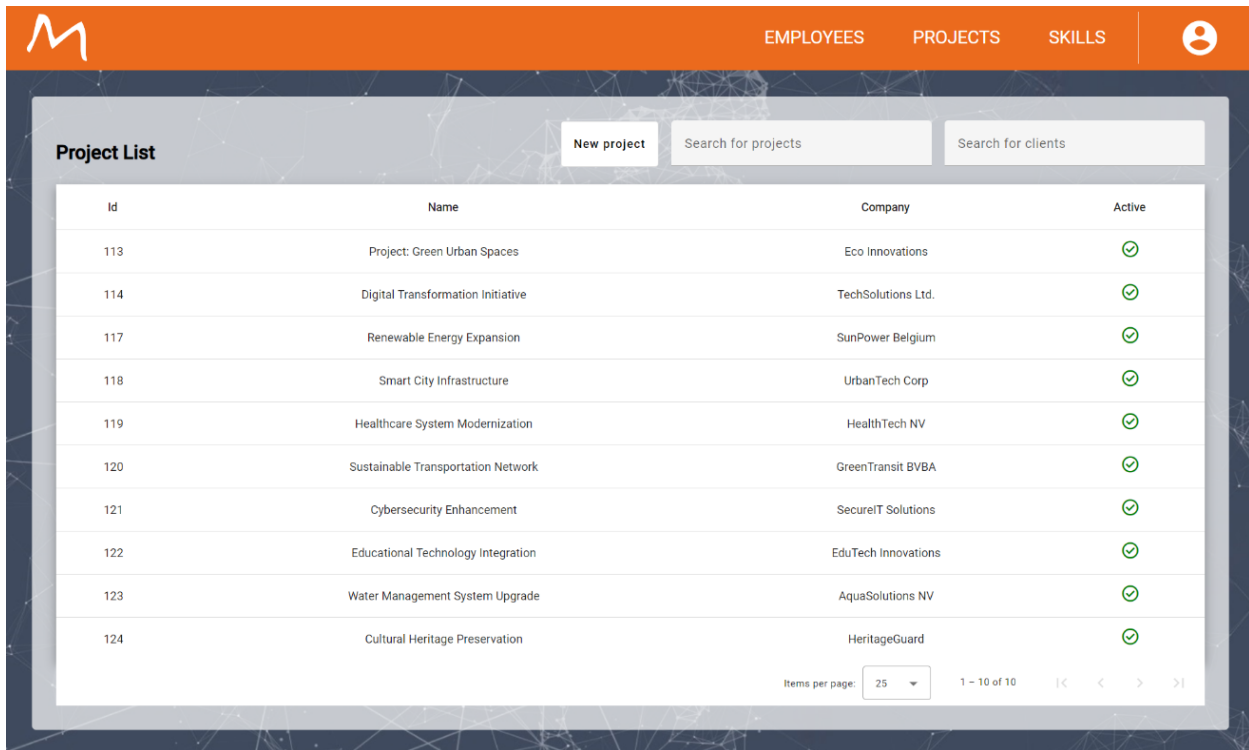
**Personal Information:** Employees can update their name, contact information, address, and other personal details. This ensures that all contact information remains accurate and up to date.

## Managing Skills

The skills management feature is a critical aspect of the employee profile page, enabling employees to dynamically update their skill sets. This functionality includes:

- **Adding Skills:** Employees can add new skills to their profile by selecting from a predefined list of skills or by entering custom skills. This allows them to highlight new competencies they have acquired through training, projects, or self-study.
- **Removing Skills:** If an employee no longer wishes to highlight certain skills, they can easily remove them from their profile. This ensures that their skill set remains relevant and focused on their current abilities and career goals.
- **Rating Skills:** Employees can rate their proficiency in each skill, providing a self-assessment that can be useful for managers and colleagues to understand their expertise level. These ratings can be periodically reviewed and updated as the employee's proficiency grows.

## Project list page



Id	Name	Company	Active
113	Project: Green Urban Spaces	Eco Innovations	✓
114	Digital Transformation Initiative	TechSolutions Ltd.	✓
117	Renewable Energy Expansion	SunPower Belgium	✓
118	Smart City Infrastructure	UrbanTech Corp	✓
119	Healthcare System Modernization	HealthTech NV	✓
120	Sustainable Transportation Network	GreenTransit BVBA	✓
121	Cybersecurity Enhancement	SecureIT Solutions	✓
122	Educational Technology Integration	EduTech Innovations	✓
123	Water Management System Upgrade	AquaSolutions NV	✓
124	Cultural Heritage Preservation	HeritageGuard	✓

The project list page is a crucial tool for the sales employees, particularly those with admin privileges, providing a comprehensive overview of all the projects within the system. This page is designed to facilitate efficient project management through robust filtering options and seamless navigation to related pages.

### Viewing Projects

The project list page presents a detailed list of all projects currently managed within the system. Key features include:

**Comprehensive Project Information:** Each project entry in the list displays essential details such as the project name, company name, project status, start and end dates, and the project manager's name. This information provides a quick snapshot of the project's key attributes.

## Filtering Projects

To streamline the process of locating specific projects, the project list page includes advanced filtering options:

- **Filter by Project Name:** Sales employees can filter the list by entering the project name. This is particularly useful when searching for a specific project out of many.
- **Filter by Company Name:** Employees can also filter projects based on the company name. This helps in quickly identifying all projects associated with a particular client, facilitating better client management and follow-up.

## Navigation Options

From the project list page, sales employees have direct access to additional functionalities, enhancing their ability to manage projects effectively:

- **Navigate to Employee Assign Page:** By selecting a project, sales employees can navigate to the employee assign page. This page allows them to view and manage the team members assigned to the project, ensuring that the right skills are matched with project requirements. It includes features such as adding or removing employees, viewing their roles, and assessing their availability.
- **Navigate to Project Edit Page:** Sales employees can also access the project's edit page from the project list. This page enables them to update project details, such as modifying timelines, changing the project scope, updating the status, or revising any other critical information. This functionality ensures that project data remains current and accurate, supporting effective project management and reporting.

## Project manager page – add/edit project

The screenshot shows the 'Edit Project: Green Urban Spaces' form. The form is divided into several sections: Project name, Company name, Location, and Description. To the right of the form are two calendar pickers for 'Start Date' and 'End Date'. The 'Start Date' calendar is set to March 2023, with the 1st highlighted. The 'End Date' calendar is set to December 2024, with the 31st highlighted. At the bottom right of the form are three buttons: 'Delete' (red), 'Save' (grey), and 'Cancel' (grey).

**Edit Project: Green Urban Spaces**

Project name\*  
Project: Green Urban Spaces

Company name\*  
Eco Innovations

Location\*  
Rue de la Loi 175, 1048 Brussels, Belgium

Description  
Developing urban green spaces to enhance biodiversity and community well-being in Brussels.

**Start Date**

MAR 2023

S M T W T F S

MAR

1 2 3 4

5 6 7 8 9 10 11

12 13 14 15 16 17 18

19 20 21 22 23 24 25

26 27 28 29 30 31

**End Date**

DEC 2024

S M T W T F S

DEC

1 2 3 4 5 6 7

8 9 10 11 12 13 14

15 16 17 18 19 20 21

22 23 24 25 26 27 28

29 30 31

Delete Save Cancel

The project manager page provides a robust set of tools for the admin to manage and update project details effectively. This page is essential for ensuring that project information is current, accurate, and easily accessible.

### Adding/Editing Project Details

The admin can use the project manager page to edit a variety of project details. This functionality includes:

- **Project Name:** Update the name of the project to reflect any changes or corrections.
- **Company Name:** Modify the associated company name if there has been a rebranding or if the project is reassigned to a different client.
- **Project Description:** Edit the detailed description of the project, ensuring it accurately describes the project's scope, objectives, and deliverables.
- **Start and End Dates:** Adjust the project timeline by modifying the start and end dates as necessary to reflect updated schedules.

## Soft Deleting (Archiving) Projects

The project manager page also allows the admin to soft delete, or archive, projects. This feature is crucial for maintaining a clean and organized project list without permanently removing important historical data. Key aspects of this functionality include:

- **Soft Delete Button:** By pressing the delete button, the admin can archive the project. This action does not permanently delete the project but moves it to an archived state.
- **Archived Projects:** Archived projects are no longer displayed in the active project list but can be accessed from an archive section if needed. This ensures that the project data remains available for future reference or auditing purposes.
- **Reactivation:** If a project needs to be reinstated, the admin can reactivate it from the archived state, restoring it to the active project list and making it editable again.



## Employee assign page

**TechSolutions Ltd. - Digital Transformation Initiative**

**Assigned Employees**

Name	Skills	Total hours per week	Rate per hour	Start Date	End Date	
Sophia Martin	CSS: 6 Python: 8 Java: 7	32	70	2024-05-09	2024-07-31	
Anna Janssens	Javascript: 6 CSS: 7 HTML5: 8	16	60	2024-05-09	2024-07-12	
Zoe Viola	CSS: 6 Arduino: 5 Flutter: 7	8	50	2024-05-09	2024-05-31	

Items per page: 5 1 - 3 of 3

**Available Employees**

Search by name Search skills

Name	Skills	Working for	Total hours per week	
Robin Lux	Python: 6 Java: 7 CSharp: 5	UrbanTech Corp SunPower Belgium Eco Innovations	40	+
Samuel Wouters	Nagios: 5 Puppet: 5 Docker: 7	UrbanTech Corp SunPower Belgium Eco Innovations	40	+
Salih Ekici	Gradle: 5 Python: 6 Heroku: 7	HealthTech NV Eco Innovations	40	+

Items per page: 3 1 - 3 of 12

The employee assign page is a critical tool for admins, enabling comprehensive management of employee assignments for specific projects. This page is designed to provide a detailed overview of current assignments and facilitate efficient assignment operations through intuitive and powerful features.

### Viewing Assigned Employees

The main section of the employee assign page displays a table of employees who are currently assigned to the selected project. This table includes crucial information for effective project management:

- **Employee Name:** The name of the assigned employee.
- **Hours Worked:** The number of hours the employee is scheduled to work on the project.
- **Hourly Rate:** The payment rate per hour for each employee.
- **Assignment Start and End Dates:** The duration of the employee's assignment to the project, including both the start and end dates.

## Managing Assignments

Admins have robust capabilities to manage assignments directly from this page:

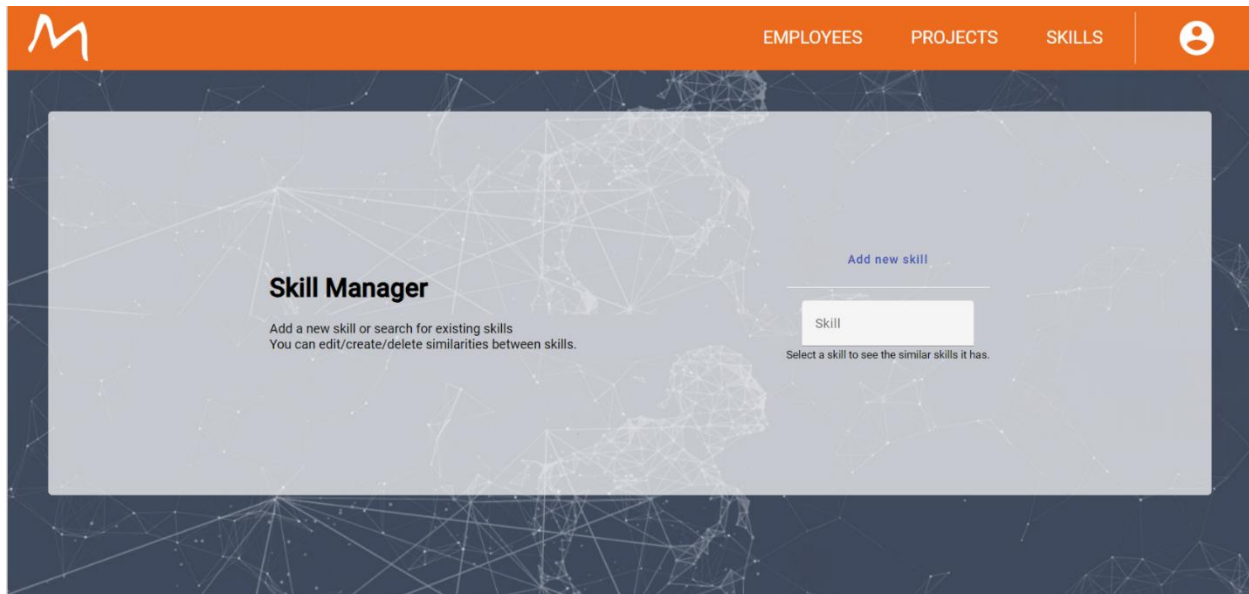
- **Edit Assignments:** Admins can edit the details of an existing assignment, such as adjusting the hours worked, changing the hourly rate, or modifying the start and end dates to reflect new project requirements or changes in availability.
- **Remove Assignments:** If an employee is no longer needed on the project, the admin can remove their assignment. This ensures that the project team is always composed of the currently required personnel.
- **Create Assignments:** Admins can create new assignments by selecting employees from the available pool and specifying their hours, pay rate, and assignment dates. This function allows for quick and efficient staffing adjustments as project needs evolve.

## Viewing and Filtering Unassigned Employees

Below the table of assigned employees, another table lists employees who are not currently assigned to the selected project. This table provides a resource pool from which admins can draw to fill project roles. Key features include:

- **Filter by Name:** Admins can filter the list of unassigned employees by name to quickly locate specific individuals.
- **Filter by Skill Set:** Admins can filter employees based on their skill sets. This filter is particularly powerful, returning a list of employees who not only possess the specified skill set but also those with similar skills. The results are ranked from best to worst match, ensuring that the most qualified candidates are displayed at the top of the list.

## Skill manager page



The skill management page provides administrators with powerful tools for managing the organization's skill database. This page is designed to add new skills and explore existing skills, along with their related and similar skills. This functionality enhances the comprehensiveness and usability of the skill database, ensuring it remains up-to-date and relevant.

### Adding New Skills

One of the primary features of the skill management page is the ability for administrators to add new skills to the database. This process includes:

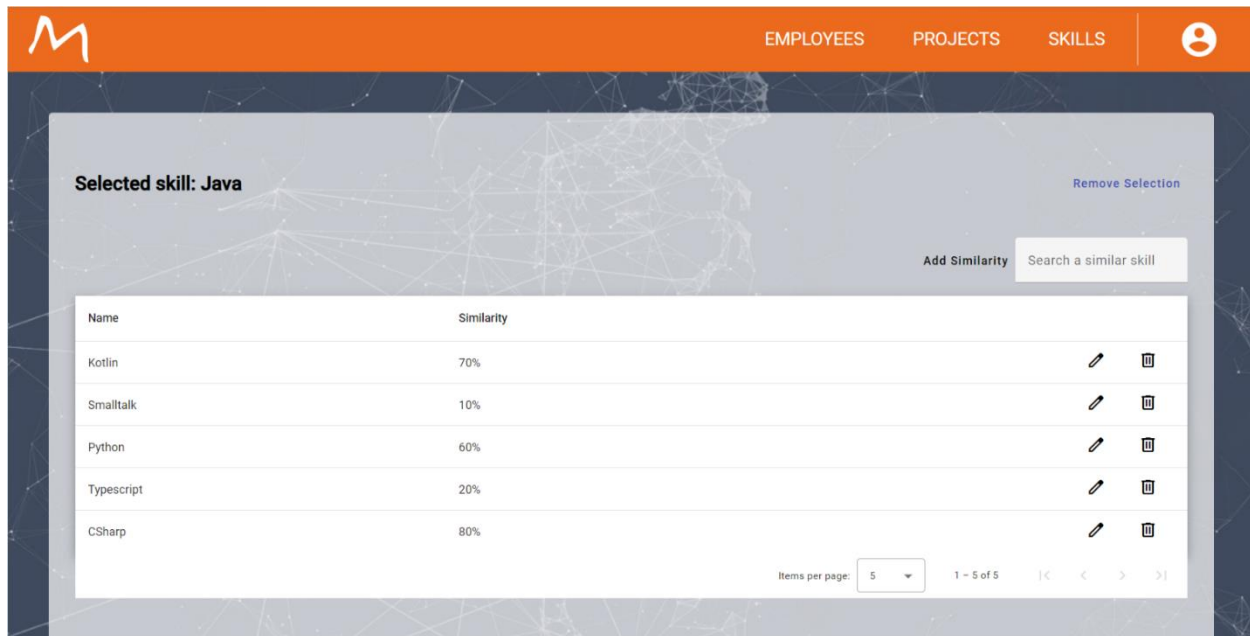
- **Skill Name Entry:** Administrators can input the name of the new skill they wish to add.

## Exploring Existing Skills and Related Skills

The skill management page also allows administrators to select an existing skill and explore its related and similar skills. This feature includes:

- **Selecting an Existing Skill:** Admins can choose from a list of existing skills in the database to view detailed information about each one.
- **Viewing Related Skills:** Once a skill is selected, the page displays a list of related skills that are similar or complementary. This list helps in understanding the overlaps and connections between different skills.
- **Detailed Skill Information:** For each related skill, detailed information is provided, including descriptions, categories, and the context in which the skill is used. This information helps administrators make informed decisions about training needs, role assignments, and skill development.

## Skill manager – View similar skills of a skill



The skill management page includes functionality that allows administrators to assign new similar skills to a selected skill. This feature is crucial for maintaining a well-organized and comprehensive skill database, ensuring that the relationships between skills are accurately reflected and easily accessible.

### Assigning New Similar Skills

The process of assigning new similar skills to a selected skill is streamlined and efficient, involving the following steps:

- **Selecting a Skill:** Administrators begin by selecting a skill from the existing database. This can be done through a search function or by browsing a categorized list of skills.
- **Viewing Current Similar Skills:** Once a skill is selected, the page displays a list of currently assigned similar skills. This provides context and helps administrators understand existing relationships.
- **Adding New Similar Skills:** Administrators can then add new similar skills by either selecting from a list of existing skills or by adding completely new skills to the database. This process includes:
  - **Search and Select:** A search bar allows administrators to quickly find and select skills that are like the selected skill.
  - **Manual Addition:** If the desired similar skill is not already in the database, administrators can manually add it, providing necessary details such as the skill name, description, and category.

## Backend

### Technologies

For the backend we used the Spring Boot framework to develop Java API's, this choice was made because of the popularity of this framework and the experience within the team. Spring Boot makes it easier to develop API's by providing the user with tools using the proxy pattern.

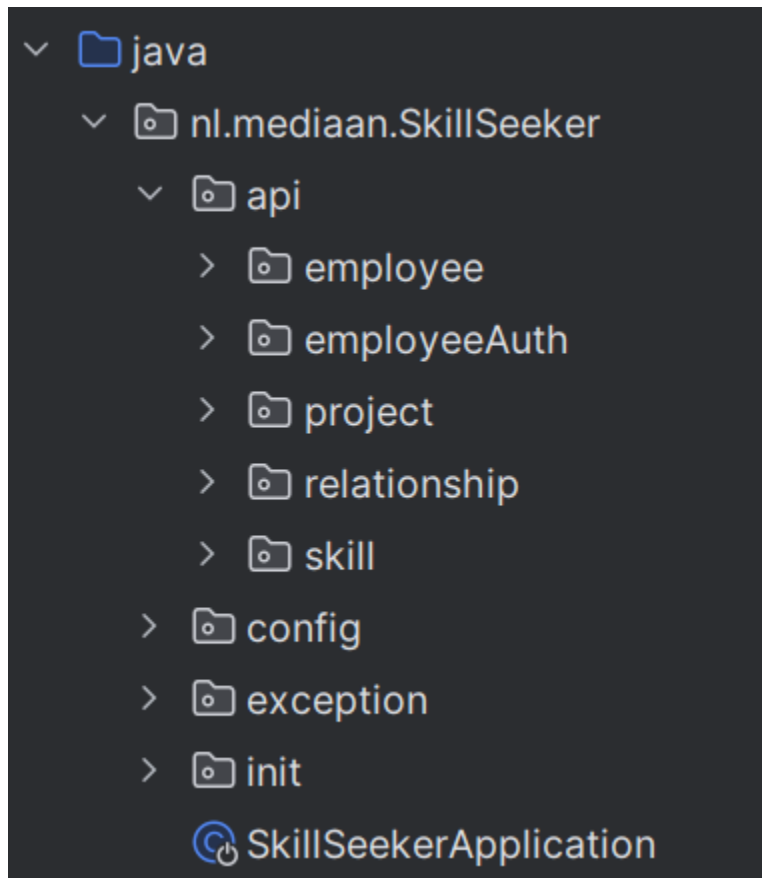
### Setup

- First, make sure you followed the steps to set up the [database](/SkillSeeker-Documentation/Technical-Documentation/Database) first!
- To set up the local backend you need to make sure u have Java 17 and maven installed and configured correctly.
- Next navigate to the root directory of the project so go inside the SkillSeeker-backend folder.
- Then use the following command:

```
mvn spring-boot:run|
```

- This will start up the backend and populate the database if it is empty.

## Folder Structure



## Pipeline

A **pipeline** in technology is a way to automate a series of steps that need to happen in a specific order to achieve a result.

```
1 trigger:
2   - main
3
4 pool:
5   vmImage: 'ubuntu-latest'
6
7 variables:
8   dockerRegistryServiceConnection: 'SkillSeekerACR'
9   imageRepository: 'skillseeker-backend'
10  containerRegistry: 'skillseekercontainerregistry.azurecr.io'
11  tag: '$(Build.BuildId)'
12  dockerfilePath: '$(Build.SourcesDirectory)/Dockerfile'
13
14 steps:
15   - task: CopyFiles@2
16     inputs:
17       SourceFolder: '$(System.DefaultWorkingDirectory)/src/main/resources/application-production.properties'
18       Contents: '**'
19       TargetFolder: '$(System.DefaultWorkingDirectory)/src/main/resources/application.properties'
20
21   - task: Maven@3
22     inputs:
23       mavenPomFile: 'pom.xml'
24       goals: 'package'
25       javaHomeOption: 'JDKVersion'
26       jdkVersionOption: '1.17'
27
28   - task: Docker@2
29     displayName: Build and push an image to container registry
30     inputs:
31       command: buildAndPush
32       repository: $(imageRepository)
33       dockerfile: $(dockerfilePath)
34       containerRegistry: $(dockerRegistryServiceConnection)
35       tags: |
36         latest
37         $(tag)
38
39   - task: AzureContainerApps@1
40     inputs:
41       imageToDeploy: '$(containerRegistry)/skillseeker-backend:$(tag)'
42       azureSubscription: 'Interns BE'
43       acrName: 'skillseekercontainerregistry'
44       containerAppName: 'skillseeker-backend'
45       resourceGroup: 'TE_Interns_Belgie'
46       containerAppEnvironment: 'managedEnvironment-TEInternsBelgie-a2a7'
47       targetPort: '8080'
48       ingress: 'external'
```

The backend pipeline is activated whenever there is a change on the main branch. In addition, the main branch is secured so that no direct commits can be performed on it. Changes can only be made via a pull request reviewed by the entire team.

The pipeline starts by using an up-to-date Ubuntu image as the operating system. Next, some variables are defined that can be used throughout the pipeline.



In Java, it is not possible to explicitly specify whether it is a production or development environment. Therefore, a different method is used. The Spring Boot application contains an `application.properties` file that specifies the URL of the database. In development, this should point to `localhost`, but in production it should point to the address of the container within the cluster. Therefore, there is a second file called `application-production.properties` that contains the production URL of the database. The first step in the pipeline swaps the contents of these files so that the correct URL is set in the `application.properties` file.

Next, the Maven task is used to download the necessary libraries defined in the `pom.xml` file. After these are downloaded, the Docker command can be used to create an image of the code and then push it to the registry. The registry is defined by an Azure Container Registry connection that also provides immediate access.

Finally, the `AzureContainerApp` task is used to signal the creation of a new container with the image just pushed to the registry. This completes the deployment from the backend to the production environment.

## Database

For SkillSeeker, we decided to utilize a graph database named [Neo4j](#).

### What is a Graph Database

A graph database is a type of database that uses graph structures with nodes, edges, and properties to represent and store data. In a graph database:

- **Nodes:** Represent entities in the system.
- **Edges:** Represent the relationships between nodes.
- **Properties:** Contain information or attributes associated with both nodes and edges.

The choice of a graph database, in this case, is driven by the nature of the data and the specific requirements of the application.

In the context of SkillSeeker, where the goal is to efficiently assign employees to projects based on their skills, the relationships between skills play a crucial role.

Here are some reasons why a graph database is well-suited for this scenario:

- **Complex Relationships:** Skills often have complex relationships with each other. For example, one skill might be a prerequisite for another, or certain skills might complement each other. Representing these relationships in a graph database allows for a more natural and intuitive model compared to traditional relational databases.
- **Efficient Querying:** Graph databases excel at traversing relationships between entities. This is particularly useful when searching for the best employees for a specific skill. With a graph database, queries can efficiently navigate the skill graph to find employees who possess the required skill directly or indirectly through related skills.
- **Scalability:** As the number of employees and skills grows, the complexity of the relationships between them also increases. Graph databases are designed to handle highly interconnected data and can scale efficiently as the dataset expands.
- **Flexibility:** Graph databases offer flexibility in modeling data. New skills can be added, relationships between skills can evolve, and new types of queries can be accommodated without significant changes to the database schema.

- Recommendation and Analysis: By leveraging the rich network of relationships between skills and employees, graph databases enable advanced recommendation algorithms and data analysis techniques. This can help in identifying patterns, making informed decisions, and optimizing the assignment of employees to projects based on skill similarities and other factors.

In summary, the choice of a graph database for SkillSeeker is driven by the need to effectively model and leverage the relationships between skills, which are central to the application's functionality of matching employees to projects based on their skill sets.

## Setup

Run the following command in your terminal to pull the image and run the Docker container: <https://hub.docker.com/repository/docker/salihekici/neo4j-with-apoc/general>

```
docker run -d -p 7474:7474 -p 7687:7687 salihekici/neo4j-with-apoc
```

## Conclusion

The SkillSeeker application significantly streamlines the process of assigning employees to projects at Mediaan. By leveraging the power of a graph database, specifically Neo4j, the application can efficiently manage and query complex relationships between different skills. This ensures that when a project requires a specific skill, the application can recommend not only the employees who possess that skill but also those with related competencies, such as C# developers for Java projects.

The use of Java Spring Boot for the backend, Angular for the frontend, and serverless container apps in the Azure cloud provides a robust, scalable, and maintainable solution. This architecture ensures that the application is accessible online, making it user-friendly for the non-technical sales department.

With the implementation of SkillSeeker, Mediaan can now quickly and accurately match the best employees to new projects, reducing the time spent manually sifting through CVs and enhancing overall efficiency. This modernization aligns with Mediaan's commitment to leveraging advanced technology to optimize their business processes and maintain a competitive edge in the IT consultancy industry.