

CMPE 282 - Project

University: [San Jose State University \(SJSU\)](#)

Course: [Cloud Services](#)

Professor: [Andrew Bond](#)

Team: [Techno Spartan](#)

Name	Student ID
Parvathi Pai	015293460
Shreya Ghotankar	015304393
Sania Gonsalves	015313974

TECHNO SPARTAN	VERSION: 2.0
PROJECT PLAN	DATE: 05/14/2021

Revision History

Date	Version	Description	Author
02/03/2021	0.1	Initial Draft	Shreya Ghotankar
02/10/2021	0.2	Kubernetes Architecture and workflow	Parvathi Pai Sania Gonsalves
02/14/2021	1.0	Minor edits and published	Shreya Ghotankar Parvathi Pai Sania Gonsalves
05/05/2021	1.1	Architecture Diagram updated	Parvathi Pai
04/15/2021	1.2	Implemented CRUD operations backend	Sania Gonsalves
05/10/2021	1.3	Updated Implementation, Deployment, CI/CD Information	Parvathi Pai Shreya Ghotankar
05/14/2021	2.0	Peer Review and Final publish	

Table of Contents

Introduction	4
Project Idea	4
Presentation and Demo	4
Architecture Diagram	5
High Level Design	5
Technology and Services Stack	7
Project Group	7
Implementation	7
Deployment	11
Application Screenshots	17
References	24

Introduction

Purpose of this document

The purpose of this document is to provide a detailed project description of the application called “Smart Salary”, which will have three different roles: HR, Manager and Employee. Through this application, the employee can tweet his salary raise; Manager can list their reportee, update, view their salaries; and HR can visualize the salary change and predict the growth of both employee and manager. This document includes details about the application idea, development, deployment, and deliverables.

Scope:

This document defines the project plan of the “Smart Salary” application. This includes project idea, individual contribution, development process, deployment process and deliverables.

Project Idea

We have used the default project database and created application called **Smart Salary** where the employees can view their information like salary and reporting manager on a dashboard. Managers can see employees reporting to them. HR can see the list of all employees and HR department specific dashboard that provides insights into each employee’s growth over the years. This project majorly focuses on data analysis and data visualization to generate salary insights. We have used Amazon Elastic Container Service for our application deployment.

Our responsive web application can be used by any enterprise organization and can be extended to any levels. Initially we are creating a three-tier application. This can be extended to multiple tiers depending on the depth of the organization.

3 roles: Employee, Manager and HR.

Our application has following features –

- ❖ Employee Dashboard – Employee information like name, salary, reporting manager
- ❖ Manager Dashboard – Own Information, List of employees reporting to her/him.
- ❖ HR Dashboard- List of all Employees, perform actions like update, delete and add employees, HR department specific dashboard to view Employee Trends – insights into employee demographics based on gender, age and salary difference by age.

Presentation and Demo

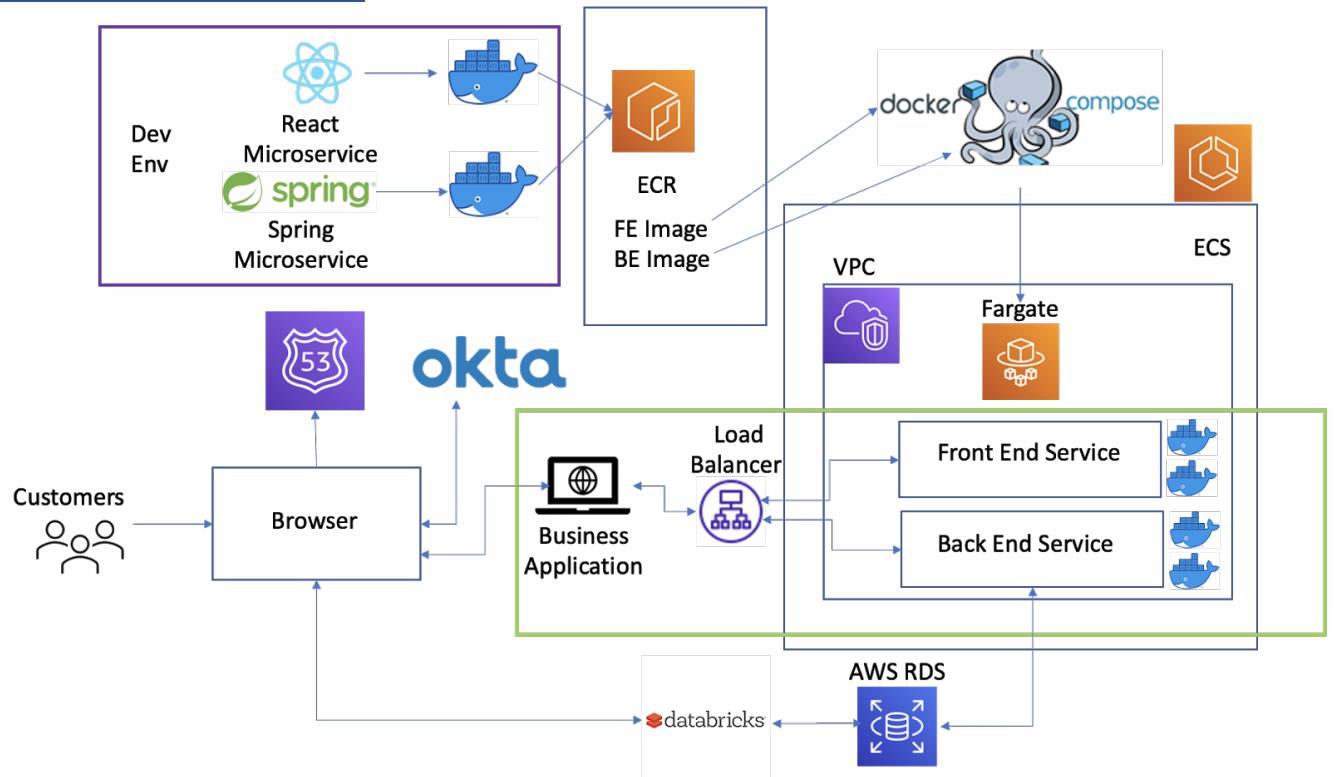
Presentation Slides -

<https://drive.google.com/file/d/1CyfWanUG60JKxU6DU8v4uYsFyqU60lDd/view?usp=sharing>

Demo -

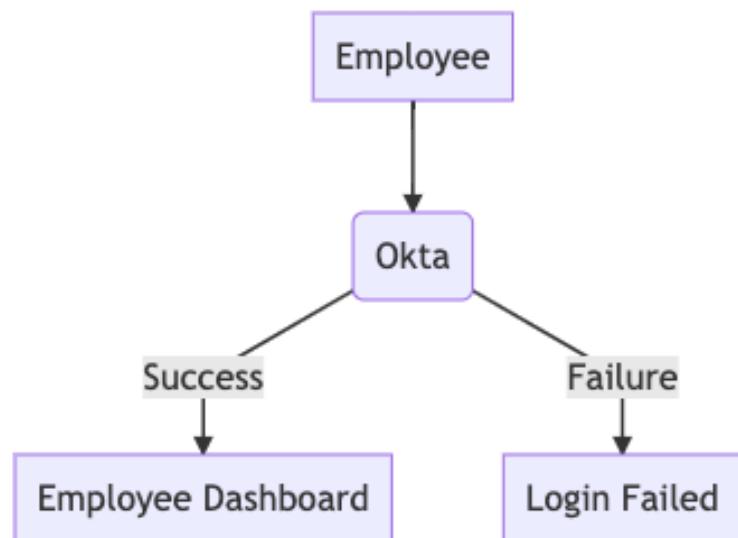
<https://drive.google.com/file/d/1VClkp9VRymb8yTUKTCLD4FyIA2mEpK8I/view?usp=sharing>

Architecture Diagram

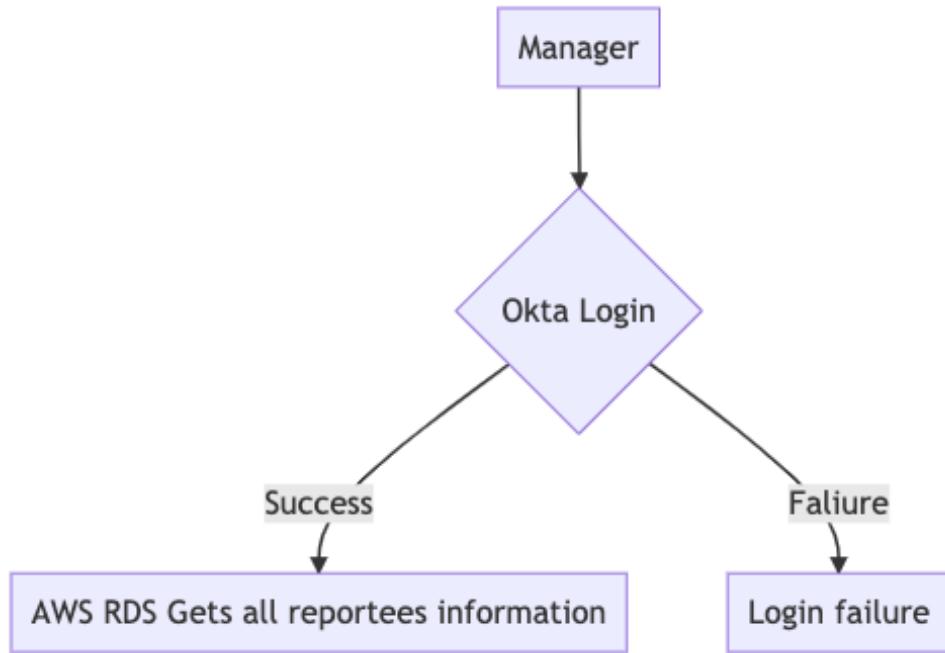


High Level Design

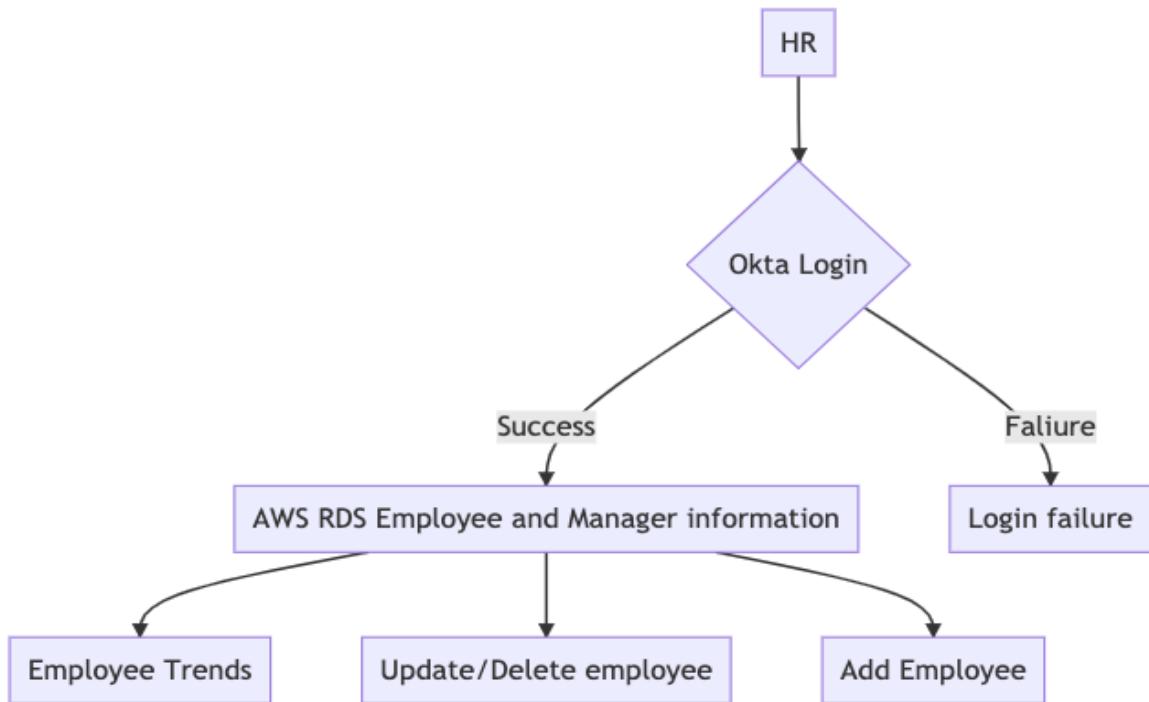
Use Case 1: Employee login and lands on Dashboard



Use Case 2: Manager can see list of employees reporting to him/her.



*Use Case 3: Add, update and delete employees.
Data visualization for Employee Trends.*



Technology and Services Stack

- ❖ Front-end – CSS, JavaScript, REACT JS, Material Design
- ❖ Spring Boot
- ❖ Authentication and Authorization – Okta
- ❖ Database - AWS RDS MYSQL
- ❖ Data Insights– Python, Databricks
- ❖ Deployment – Docker, Docker Compose, Amazon ECS, Amazon ECR, AWS Fargate

Project Group

Name	Responsibility
Shreya Ghotankar	Authentication and Authorization, Role-based access, UI, Documentation Add, Update, Delete Employee - Frontend
Parvathi Pai	UI, Employee Trends using Databricks, Deployment of frontend and backend microservices using Docker, Docker compose, Amazon ECS.
Sania Gonsalves	Implemented Add, Update, Delete Employee -Backend

Implementation

GitHub URL: https://github.com/ParvathiRPai/Cloud_Services

1. Authentication and Authorization

We have used OKTA for our authentication and authorization feature. User can Login or Sign Up using the Okta Sign-in Widget integrated in our React JS application. We have also implemented role-based access feature. There are 3 roles – Employee, Manager and HR. Based on the role the specific dashboard is made available to the user.

Okta Configuration:

The screenshot shows the Okta application configuration interface for an application named "SmartSalary2". The left sidebar has sections like Dashboard, Directory, Applications (selected), Self Service, Security, Workflow, Reports, and Settings. The main area is divided into three sections: APPLICATION, USER CONSENT, and LOGIN.

- APPLICATION:**
 - App integration name: SmartSalary2
 - Application type: Single Page App (SPA)
 - Grant type: Client acting on behalf of a user
 - Authorization Code (checked)
 - Implicit (checked)
 - Allow ID Token with implicit grant type (checked)
 - Allow Access Token with implicit grant type (checked)
- USER CONSENT:**
 - User consent (checkbox checked)
 - Require consent (checkbox checked)
 - Terms of Service URI (link)
 - Policy URI (link)
 - Logo URI (link)
- LOGIN:**
 - Sign-in redirect URIs:
 - http://127.0.0.1:5000/login/callback
 - http://testlb1-ad60a16c04b0615e.elb.us-east-1.amazonaws.com:3000/login/callback
 - http://technospartan.parvathipai.com:3000/login/callback
 - Sign-out redirect URIs:
 - http://testlb1-ad60a16c04b0615e.elb.us-east-1.amazonaws.com:3000/
 - http://127.0.0.1:5000/

On the left, there's a message about upgrading to the Advanced plan and a blue "Upgrade" button.

2. We have used AWS RDS – MYSQL as our database, where we imported the data from the default project database.

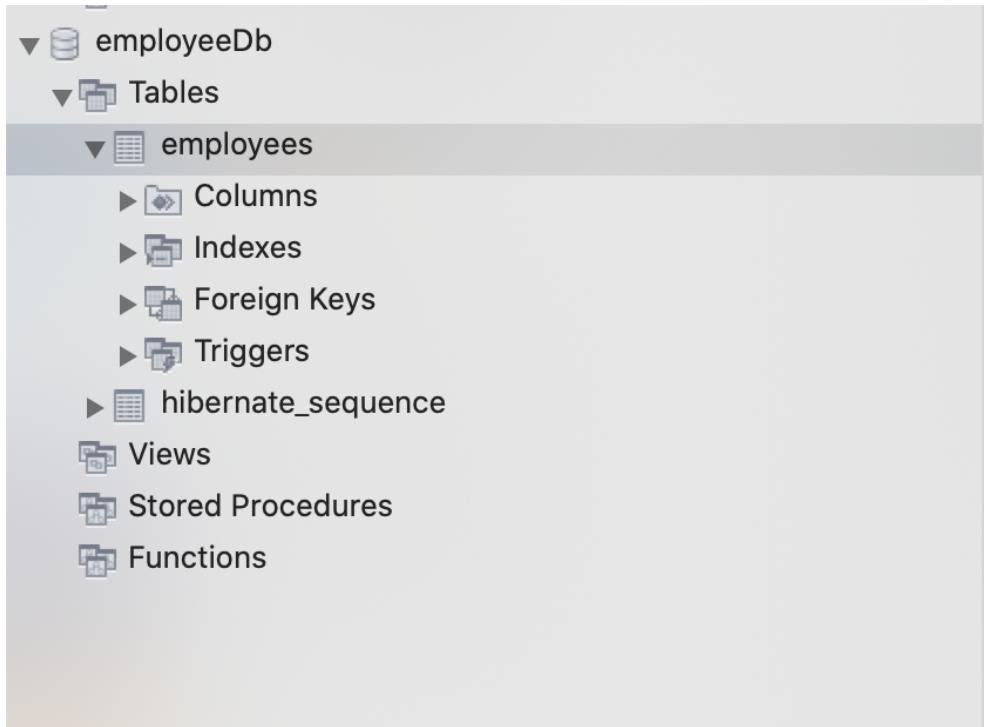
The screenshot shows the AWS RDS MySQL instance configuration page for "employeedb-1".

- Summary:**

DB identifier: employeedb-1	CPU: 1.97%	Status: Starting	Class: db.t2.micro
Role: Instance	Current activity	Engine: MySQL Community	Region & AZ: us-west-2b
- Connectivity & security:**

Endpoint: employeedb-1.c511y206r0j.us-west-2.rds.amazonaws.com	Networking: Availability zone: us-west-2b, VPC: vpc-f6fa2887, Subnet group: default, Subnets: subnet-18f92660, subnet-9eb0a8d4, subnet-9976db4, subnet-eb6a4b0c	Security: VPC security groups: default (sg-b206779f) (active), Public accessibility: Yes, Certificate authority: rds-ca-2019, Certificate authority date: August 22, 2024 10:08
--	---	---
- Security group rules (2):**

Security group	Type	Rule
default (sg-b206779f)	EC2 Security Group - Inbound	sg-b206779f



3. Using DB to fetch employee information to display on the dashboard. Everyone is an employee including Manager and HR. So, they also see the employee dashboard.
4. Manager has a manager specific dashboard which shows manager the list of employees reporting to them.
5. HR has access to perform actions like Add, update and delete employees which are implemented as CRUD operations.
6. HR has a specific dashboard to view the Employee Trends. These trends are generated using the [test data](#). We have made use of Databricks community edition for data analysis. We are using Python with libraries like NumPy, Pandas and Matplotlib.

Databricks Screenshots

Cloud_services (Python)

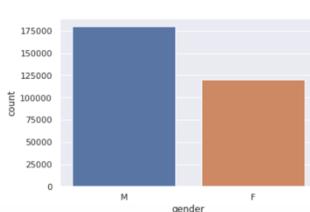
```
Cmd 1
def CreateSchemaToPandas(file_location):    file_type = "csv"  # CSV options  i ...
Show cell

Cmd 2
file_location_salaries = "/FileStore/tables/salaries.csv" file_location_dept_ma ...
Show cell

Cmd 3
df_salaries=CreateSchemaToPandas(file_location_salaries) display(df_salaries) df ...
Show cell

Cmd 4
import seaborn as sns import matplotlib.pyplot as plt ...
Show cell

Cmd 5
1 sns.set(style="darkgrid")
2 ax = sns.countplot(x="gender", data=df_employees.gender != 28791)
3 # Through this analysis we can conclude that there are more male employees.
```



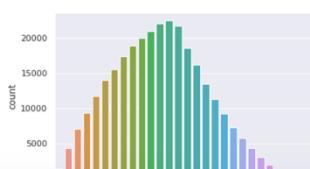
Cloud_services (Python)

```
Cmd 1
1 import pandas as pd
2 import numpy as np
3 # Type casting object type to string type
4 df_employees['hire_date']=pd.to_datetime(df_employees.hire_date)
5 df_employees['birth_date']=pd.to_datetime(df_employees.birth_date)
6 df_employees['age_when_hired'] = (df_employees['hire_date'] - df_employees['birth_date'])/np.timedelta64(1, 'Y')
7 df_employees['age_when_hired'] = df_employees.age_when_hired.astype(int)

Command took 0.22 seconds -- by parvathi.pai@sjtu.edu at 3/29/2021, 1:58:56 PM on cloud_services
Cnd 7
1 df_employees.head()

Out[7]:
emp_no birth_date first_name last_name gender hire_date age_when_hired
0 10001 1953-09-02 Georgi Facello M 1986-06-26 32
1 10002 1964-06-02 Bezalel Simmel F 1985-11-21 21
2 10003 1958-12-03 Parto Bamford M 1986-08-28 26
3 10004 1954-05-01 Christian Koblick M 1986-12-01 32
4 10005 1955-01-21 Kyochi Maliniak M 1989-09-12 34

Command took 0.07 seconds -- by parvathi.pai@sjtu.edu at 3/29/2021, 1:58:57 PM on cloud_services
Cnd 8
1 yx = sns.countplot(x='age_when_hired', data=df_employees)


```

Cloud_services Python

File Edit View Standard Permissions Run All Clear Publish Comments Experiment Revision history

```

Cmd 9
1 grouped_salaries = df_salaries.groupby('emp_no')
2 df_employees['salary_dif'] = np.nan
3 for i, (name, group) in enumerate(grouped_salaries):
4     salary_dif = max(group.salary) - min(group.salary)
5     df_employees.loc[i, 'salary_dif'] = salary_dif

Command took 5.75 minutes -- by parvathi.pai@sjsu.edu at 3/29/2021, 1:56:07 PM on cloud_services

```

```

Cmd 10
1 age_range = range(20, 48)
2 avg_salary = list(np.zeros(len(age_range)))
3 for i, age in enumerate(age_range):
4     avg_salary[i] = np.mean(df_employees[df_employees['age_when_hired'] == age].salary_dif)
5 plt.plot(age_range, avg_salary)
6 plt.xlabel('Salary difference by age')
7 plt.ylabel('Age')

Out[13]:


```

```

Text(0.5, 0, 'Age')
Command took 1.42 seconds -- by parvathi.pai@sjsu.edu at 3/29/2021, 1:56:14 PM on cloud_services

```

7. We have added a JUnit test case.

```

package com.example.smartsalary.backend;

import ...

@SpringBootTest
class BackendApplicationTests {
    @Autowired
    private EmployeeController controller;
    @Test
    void contextLoads() { assertThat(controller).isNotNull(); }
}

```

Deployment

- We are making use of two microservices - Front end using react and back end using spring.
- Locally we are creating two images one for front end and another for back end using docker as shown

v1_frontend-ser	latest	6faa70985b1c	6 days ago	720MB
public.ecr.aws/x4y9x2u7/pavasjsuproj	frontend-latest	6faa70985b1c	6 days ago	720MB
public.ecr.aws/x4y9x2u7/pavasjsuproj	backend-latest	324c3211144c	6 days ago	912MB
v1_backend-ser	latest	324c3211144c	6 days ago	912MB
- The local docker images are registered in ECR as shown. We are making use of public repositories as it is almost free of cost, whereas the private repositories are charged after 750MB.

Amazon Container Services

Amazon ECS

Clusters

Task definitions

Amazon EKS

Clusters

Amazon ECR

Repositories

Images

Gallery detail

Permissions

Tags

Registries

Public gallery

ECR > Repositories > pavasjsuproj

pavasjsuproj

View public listing | View push commands | Edit

Images (2)

Image tag	Pushed at	Size (MB)	Image URI	Digest
backend-latest	May 04, 2021 01:58:12 PM	591.24	Copy URI	sha256:6219df9db344d8...
frontend-latest	May 04, 2021 01:42:23 PM	218.04	Copy URI	sha256:a2621708b641e7...

- The two microservices are deployed using Fargate as shown in the figure

New ECS Experience

Tell us what you think

Amazon ECS

Clusters

Task Definitions

Account Settings

Amazon EKS

Clusters

Amazon ECR

Repositories

AWS Marketplace

Discover software

Subscriptions

Clusters

Clusters

An Amazon ECS cluster is a regional grouping of one or more container instances on which you can run task requests. Each account receives a default cluster the first time you use the Amazon ECS service. Clusters may contain more than one Amazon EC2 instance type.

For more information, see the [ECS documentation](#).

Create Cluster | Get Started

View list card

v1 > CloudWatch monitoring Default Monitoring

FARGATE

Services	Running tasks	Pending tasks
2	2	0

EC2

Services	Running tasks	Pending tasks	CPUUtilization	MemoryUtilization	Container instances
0	0	0	No data	No data	0

New ECS Experience [Tell us what you think](#)

Amazon ECS [Clusters](#)

- Task Definitions
- Account Settings
- Amazon EKS
- Clusters
- Amazon ECR
- Repositories
- AWS Marketplace
- Discover software
- Subscriptions

Clusters > v1

Cluster : v1

Get a detailed view of the resources on your cluster.

Cluster ARN	arn:aws:ecs:us-east-1:274412100151:cluster/v1
Status	ACTIVE
Registered container instances	0
Pending tasks count	0 Fargate, 0 EC2
Running tasks count	2 Fargate, 0 EC2
Active service count	2 Fargate, 0 EC2
Draining service count	0 Fargate, 0 EC2

Services [Create](#) [Update](#) [Delete](#) [Actions](#) Last updated on May 10, 2021 5:53:44 PM (0m ago) [Filter in this page](#) [Launch type](#) ALL [Service type](#) ALL

Service Name	Status	Service type	Task Definition ...	Desired tasks	Running tasks	Launch type	Platform versio...
v1-BackendserService-TIKR2Z0qUvlg	ACTIVE	REPLICA	v1-backend-ser:5	1	1	FARGATE	1.4.0
v1-FrontendserService-QpeT1MInA9xG	ACTIVE	REPLICA	v1-frontend-ser:5	1	1	FARGATE	1.4.0

New ECS Experience [Tell us what you think](#)

Amazon ECS [Clusters](#)

- Task Definitions
- Account Settings
- Amazon EKS
- Clusters
- Amazon ECR
- Repositories
- AWS Marketplace
- Discover software
- Subscriptions

Clusters > v1 > Service: v1-BackendserService-TIKR2Z0qUvlg

Service : v1-BackendserService-TIKR2Z0qUvlg

[Update](#) [Delete](#)

Cluster	v1	Desired count	1
Status	ACTIVE	Pending count	0
Task definition	v1-backend-ser:5	Running count	1
Service type	REPLICA		
Launch type	FARGATE		
Service role	AWSServiceRoleForECS		
Created By	arn:aws:iam::274412100151:user/swTestUser1		

Details [Tasks](#) [Events](#) [Auto Scaling](#) [Deployments](#) [Metrics](#) [Tags](#) [Logs](#)

Load Balancing

Target Group Name	Container Name	Container Port
v1-Backends-NR7R9Y0WD20C	backend-ser	8080

Network Access

Health check grace period 0

Allowed VPC vpc-b75d8dca

Allowed subnets subnet-4544c41a,subnet-b9bd3f98,subnet-b1cf48d7,subnet-b30079bd,subnet-875b90b6,subnet-9f5700d2

Security groups* sg-02ec51b48bca0d3a0

Auto-assign public IP ENABLED

Service discovery

Service : v1-FrontendService-QpeT1MInA9xG

Load Balancing

Target Group Name	Container Name	Container Port
v1-Frontend-AJZ220T26RFE	frontend-ser	3000

Network Access

Health check grace period: 0

Allowed VPC: vpc-b75d8dca

Allowed subnets: subnet-4544c41a, subnet-b9bd3f98, subnet-b1cf48d7, subnet-b30079bd, subnet-875b90b6, subnet-9f5700d2

Security groups*: sg-02ec51b48bca0d3a0

Auto-assign public IP: ENABLED

- The application load balancer is configured as shown in the figure.

Basic Configuration

Name	DNS name	State	VPC ID	Availability Zones	Type	Created At
awseb-e-z-AWSEBLba-FN9...	awseb-e-z-AWSEBLba-FN9UN7KX1M8L	active	vpc-b75d8dca	us-east-1d, us-east-1b,...	classic	May 5, 2021 at 1:47:15 PM ...
testlb1	testlb1-ad60a16c04b0615e...	active	vpc-b75d8dca	us-east-1e, us-east-1d,...	network	May 4, 2021 at 8:05:50 PM ...

Port Configuration

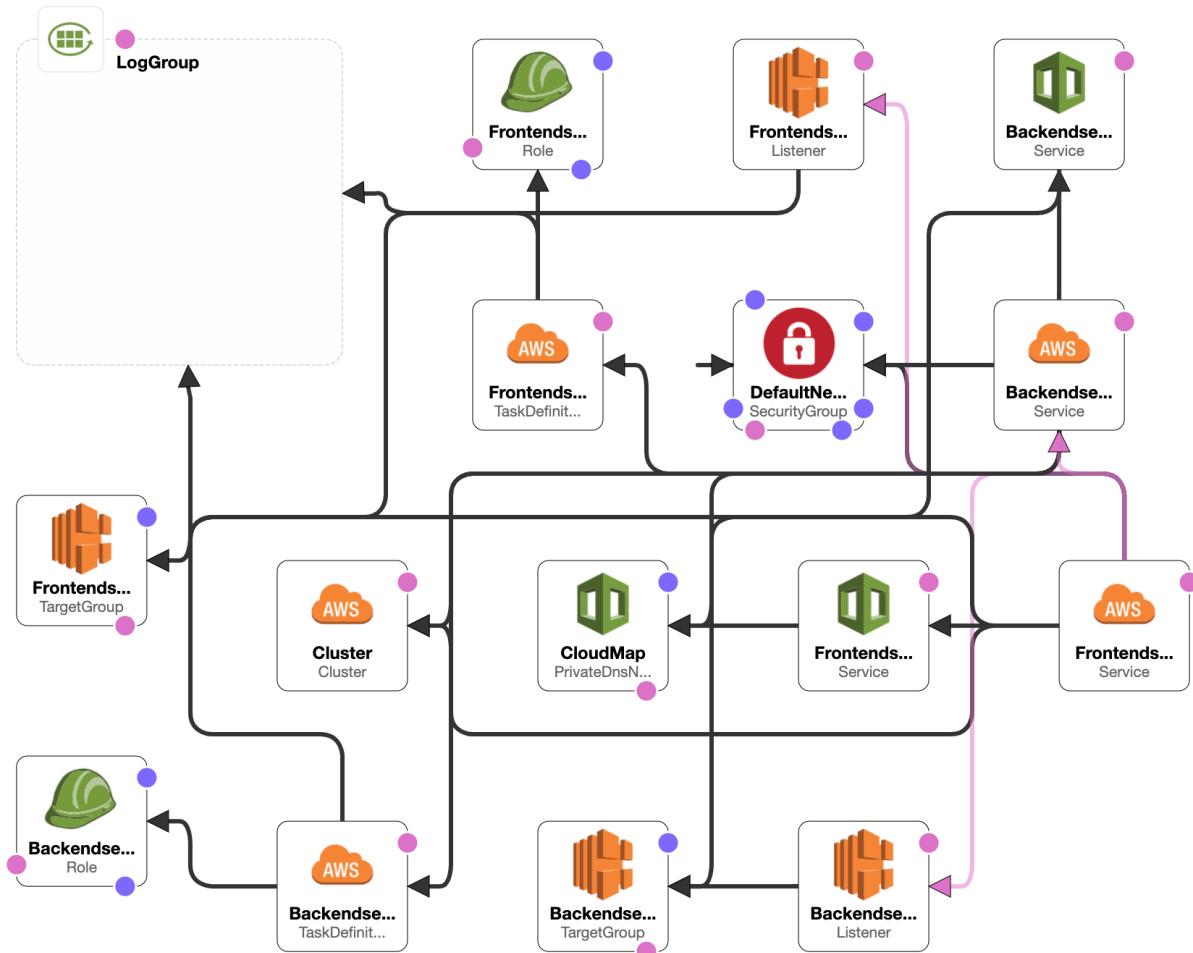
Port Configuration	80 (HTTP) forwarding to 80 (HTTP)
Stickiness:	Disabled
Edit stickiness	

- Route 53 is configured as shown in the figure

Screenshot of the AWS Route 53 console showing the hosted zone details for v1.local. The table lists four records:

Record name	Type	Routing policy	Value/Route traffic to
v1.local	NS	Simple	ns-1536.awsdns-00.co.uk. ns-0.awsdns-00.com. ns-1024.awsdns-00.org. ns-512.awsdns-00.net.
v1.local	SOA	Simple	ns-1536.awsdns-00.co.uk. awsdns-hostmaster.amazon.com. 1 7200 900 1209600 86400
backend-serv1.local	A	Multivalued	172.31.84.149
frontend-serv1.local	A	Multivalued	172.31.47.228

- Cloud formation template for Fargate is shown in the figure



V |

Delete | Update | Stack actions ▾ | Create

Stack info | Events | Resources | Outputs | Parameters | **Template** | Change sets

Template

[View in Designer](#)

```
AWS::TemplateFormatVersion: 2010-09-09
Resources:
  BackendserService:
    Type: AWS::ECS::Service
    Properties:
      Cluster:
        Fn::GetAtt:
          - Cluster
          - Arn
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 100
      DeploymentController:
        Type: ECS
      DesiredCount: 1
      LaunchType: FARGATE
      LoadBalancers:
        - ContainerName: backend-ser
          ContainerPort: 8080
          TargetGroupArn:
            Ref: BackendserTCP8080TargetGroup
      NetworkConfiguration:
        AwsVpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups:
            - Ref: DefaultNetwork
          Subnets:
            - subnet-875b90b6
            - subnet-b9bd3f98
            - subnet-4544r410
```

CI/CD

1. We have 3 environments created for CI/CD as shown in the figure

```
git:(pava/documentation) ✘ docker context ls
NAME      TYPE      DESCRIPTION          DOCKER ENDPOINT      KUBERNETES ENDPOINT      ORCHESTRATOR
default   *  moby    Current DOCKER_HOST based configuration  unix:///var/run/docker.sock  https://kubernetes.docker.internal:6443 (default)  swarm
ecsLocal   ecs-local  ECS local endpoints
myecscontext  ecs
```

2. The *ecslocal* environment is for local ECS CI/CD setup
3. *myecscontext* is for registering our service to ECR then to ECS.
4. The context switch could be done as shown in the figure

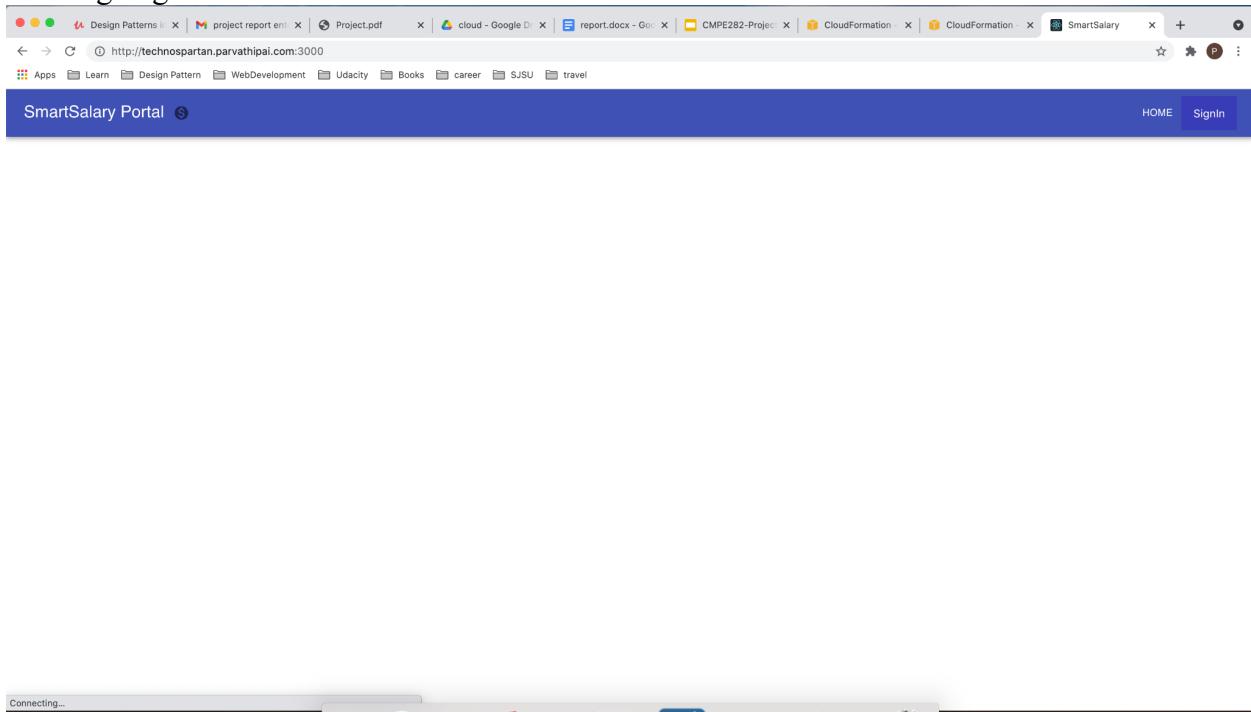
```
git:(pava/documentation) ✘ docker context use myecscontext
myecscontext
git:(pava/documentation) ✘ docker context ls
NAME      TYPE      DESCRIPTION          DOCKER ENDPOINT      KUBERNETES ENDPOINT      ORCHESTRATOR
default   *  moby    Current DOCKER_HOST based configuration  unix:///var/run/docker.sock  https://kubernetes.docker.internal:6443 (default)  swarm
ecsLocal   ecs-local  ECS local endpoints
myecscontext  *  ecs
```

5. Docker push pushes the images to ECR and ECS as shown in the figure

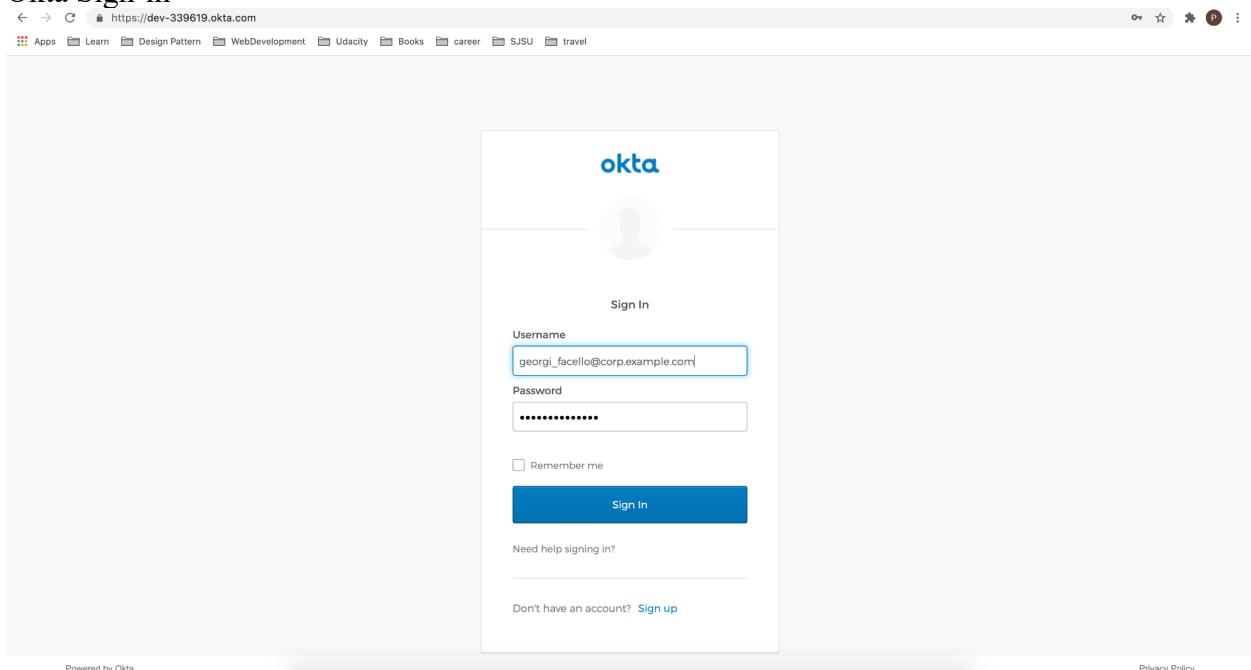
```
git:(pava/documentation) ✘ docker push public.ecr.aws/x4y9x2u7/pavajsproj:frontend-latest && docker push public.ecr.aws/x4y9x2u7/pavajsproj:backend-latest
```

Application Screenshots

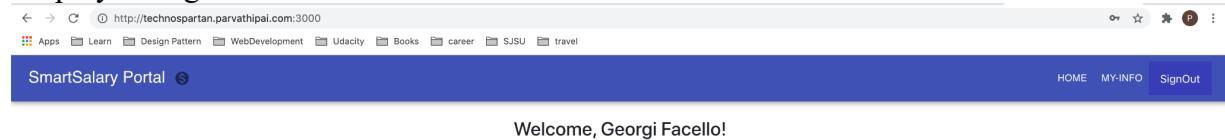
Landing Page



Okta Sign-in

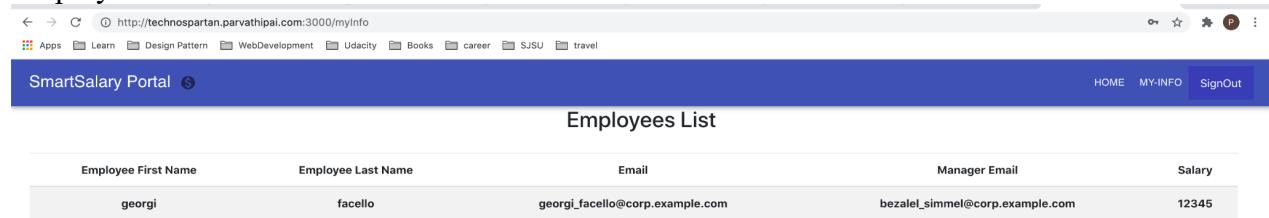


Employee Login



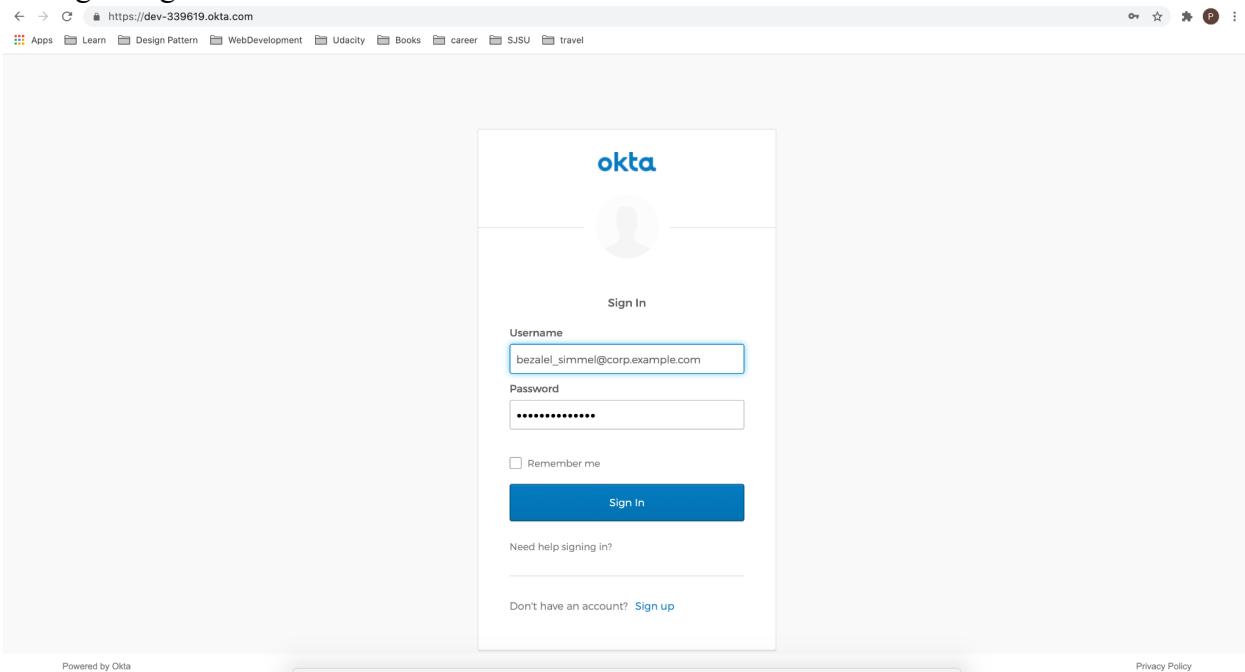
The screenshot shows a web browser window with the URL <http://technospartan.parvathipai.com:3000>. The page title is "SmartSalary Portal". The main content area displays the message "Welcome, Georgi Facello!". The top navigation bar includes links for "HOME", "MY-INFO", and "SignOut". A horizontal menu bar at the top of the page contains items like "Apps", "Learn", "Design Pattern", "WebDevelopment", "Udacity", "Books", "career", "SJSU", and "travel".

Employee Dashboard



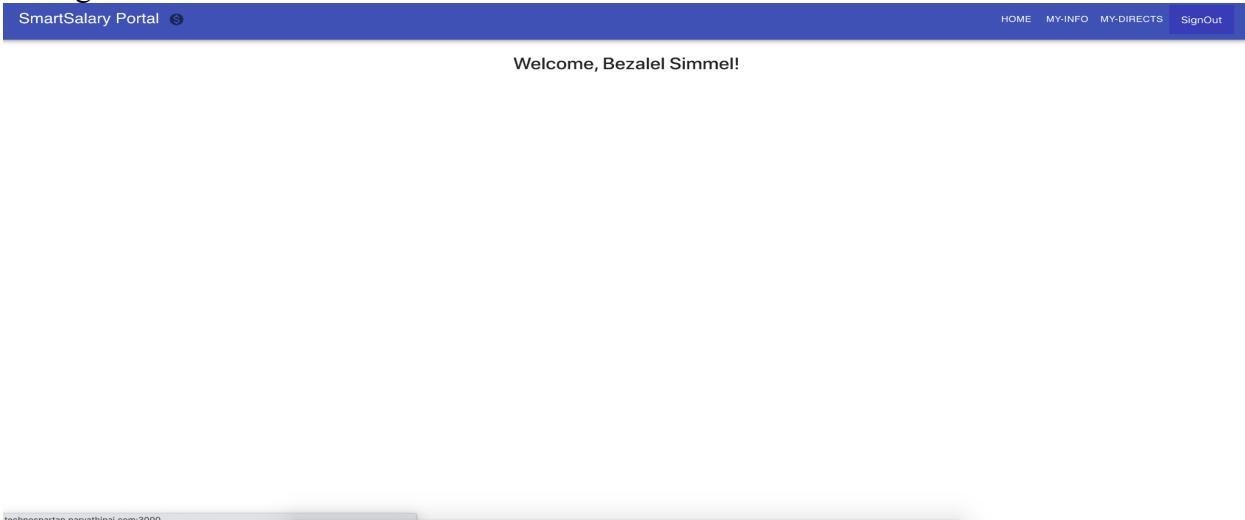
The screenshot shows a web browser window with the URL <http://technospartan.parvathipai.com:3000/myinfo>. The page title is "SmartSalary Portal". The main content area displays a table titled "Employees List" with one row of data. The table columns are "Employee First Name", "Employee Last Name", "Email", "Manager Email", and "Salary". The data row contains "georgi", "facello", "georgi_facello@corp.example.com", "bezalel_simmel@corp.example.com", and "12345". The top navigation bar includes links for "HOME", "MY-INFO", and "SignOut". A horizontal menu bar at the top of the page contains items like "Apps", "Learn", "Design Pattern", "WebDevelopment", "Udacity", "Books", "career", "SJSU", and "travel".

Manager Login



The screenshot shows the Okta sign-in interface. At the top, the URL is https://dev-339619.okta.com. Below the header, there's a placeholder user icon. The main title is "Sign In". The "Username" field contains "bezalel_simme@corp.example.com". The "Password" field is filled with a series of dots. There's a "Remember me" checkbox followed by a "Sign In" button. Below the button, links for "Need help signing in?" and "Sign up" are visible. At the bottom of the form, there are "Powered by Okta" and "Privacy Policy" links.

Manager Dashboard



The dashboard has a dark blue header bar. On the left, it says "SmartSalary Portal". In the center, it displays a welcome message: "Welcome, Bezalel Simme!". On the right, there are navigation links: "HOME", "MY-INFO", "MY-DIRECTS", and "SignOut". A small watermark at the bottom left reads "technospartan.parvathipai.com:3000".

SmartSalary Portal					
Employees List					
Employee First Name	Employee Last Name	Email	Manager Email	Salary	
bezalel	simmel	bezalel_simmel@corp.example.com	None	129999	

SmartSalary Portal					
Employees List					
Employee First Name	Employee Last Name	Email	Manager Email	Salary	
georgi	facello	georgi_facello@corp.example.com	bezalel_simmel@corp.example.com	12345	
kyoichi	maliniak	kyoichi_maliniak@corp.example.com	bezalel_simmel@corp.example.com	1250	
Mary	Sluis	Mary_Sluis@corp.example.com	bezalel_simmel@corp.example.com	123666	
Anneke	Preusig	Anneke_Preusig@corp.example.com	bezalel_simmel@corp.example.com	12400	
Kazuhide	Peha	Kazuhide_Peha@corp.example.com	bezalel_simmel@corp.example.com	30000	
Chirstian	Koblick	Chirstian_Koblick@corp.example.com	bezalel_simmel@corp.example.com	50000	

HR login

← → C https://dev-339619.okta.com

Apps Learn Design Pattern WebDevelopment Udacity Books career SJSU travel



Sign In

Username
kyoichi_maliniaik@corp.example.com

Password

Remember me

Sign In

Need help signing in?

Don't have an account? [Sign up](#)

Powered by Okta

Privacy Policy

← → C http://technospartan.parvathipai.com:3000

Apps Learn Design Pattern WebDevelopment Udacity Books career SJSU travel

SmartSalary PortalHOMEMY-INFOSALARY-INSIGHTSSignOut

Welcome, Kyoichi Maliniak!

SmartSalary Portal

Employees List

Employee First Name	Employee Last Name	Email	Manager Email	Salary	Actions	
georgi	facello	georgi_facello@corp.example.com	bezalel_simmel@corp.example.com	12345	UPDATE	DELETE
kyoichi	maliniak	kyoichi_maliniak@corp.example.com	bezalel_simmel@corp.example.com	1250	UPDATE	DELETE
Mary	Sluis	Mary_Sluis@corp.example.com	bezalel_simmel@corp.example.com	123666	UPDATE	DELETE
Anneke	Preusig	Anneke_Preusig@corp.example.com	bezalel_simmel@corp.example.com	12400	UPDATE	DELETE
Kazuhide	Peha	Kazuhide_Peha@corp.example.com	bezalel_simmel@corp.example.com	30000	UPDATE	DELETE
Chirstian	Koblick	Chirstian_Koblick@corp.example.com	bezalel_simmel@corp.example.com	50000	UPDATE	DELETE
bezalel	simmel	bezalel_simmel@corp.example.com	None	129999	UPDATE	DELETE

Add Employee

SmartSalary Portal

HOME MY-INFO SALARY-INSIGHTS SignOut

Add Employee

First Name	<input type="text"/>
Last Name	<input type="text"/>
First Name	<input type="text"/>
Email Address	<input type="text"/>
Manager Email	<input type="text"/>
Salary	<input type="text"/>
<input type="button" value="SAVE"/> <input type="button" value="CANCEL"/>	

Update Employee

SmartSalary Portal

HOME MY-INFO SALARY-INSIGHTS SignOut

Update Employee

First Name
georgi

Last Name
facello

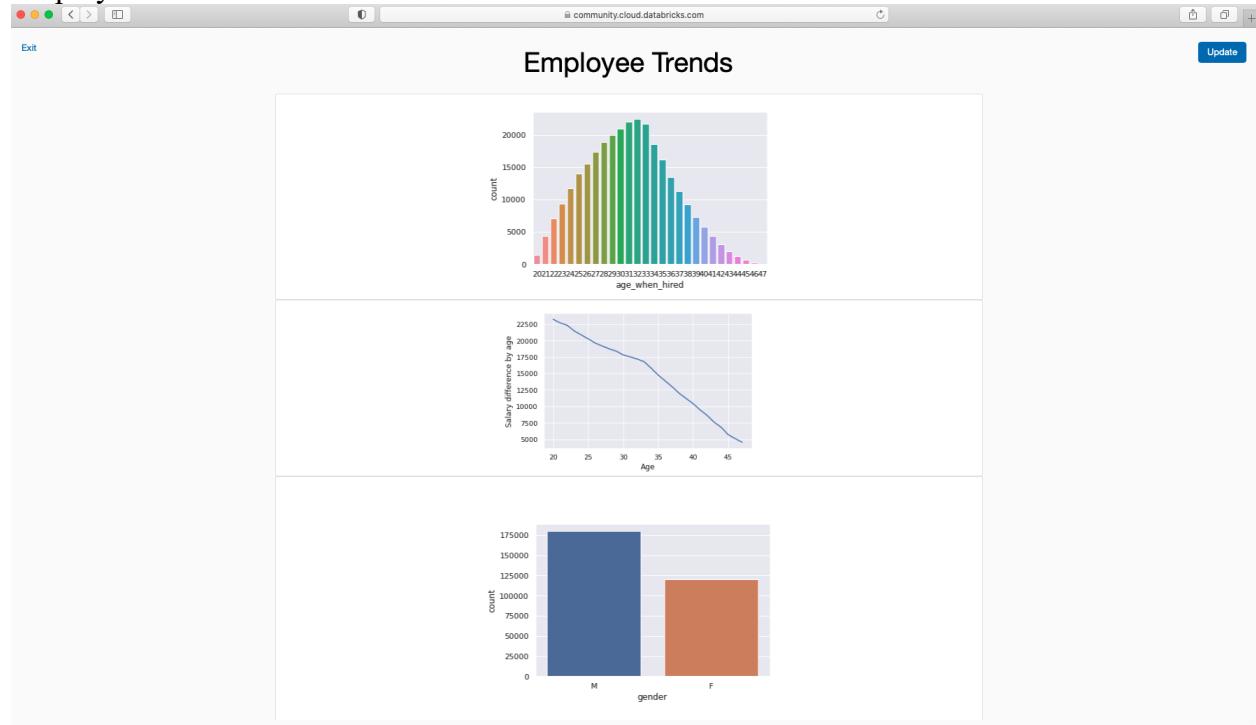
Email Address
georgi_facello@corp.example.com

Manager Email
bezalel_simmel@corp.example.com

Salary
12345

SAVE CANCEL

Employee Trends



References

- ❖ <https://www.vectorlogo.zone/util/preview.html?image=/logos/databricks/databricks-ar21.svg>
- ❖ <https://mermaid-js.github.io/mermaid/>
- ❖ https://github.com/datacharmer/test_db
- ❖ <https://community.cloud.databricks.com/>
- ❖ <https://github.com/awslabs/amazon-ecr-credential-helper>
- ❖ <https://aws.amazon.com/blogs/containers/deploy-applications-on-amazon-ecs-using-docker-compose/>
- ❖ <https://reactjs.org/>
- ❖ [https://developer.okta.com/code/react\(okta_react_sign_in_widget/](https://developer.okta.com/code/react(okta_react_sign_in_widget/)
- ❖ <https://spring.io/>