Java Full Stack Coding Challenge

Baton Rouge – Client Innovation Center

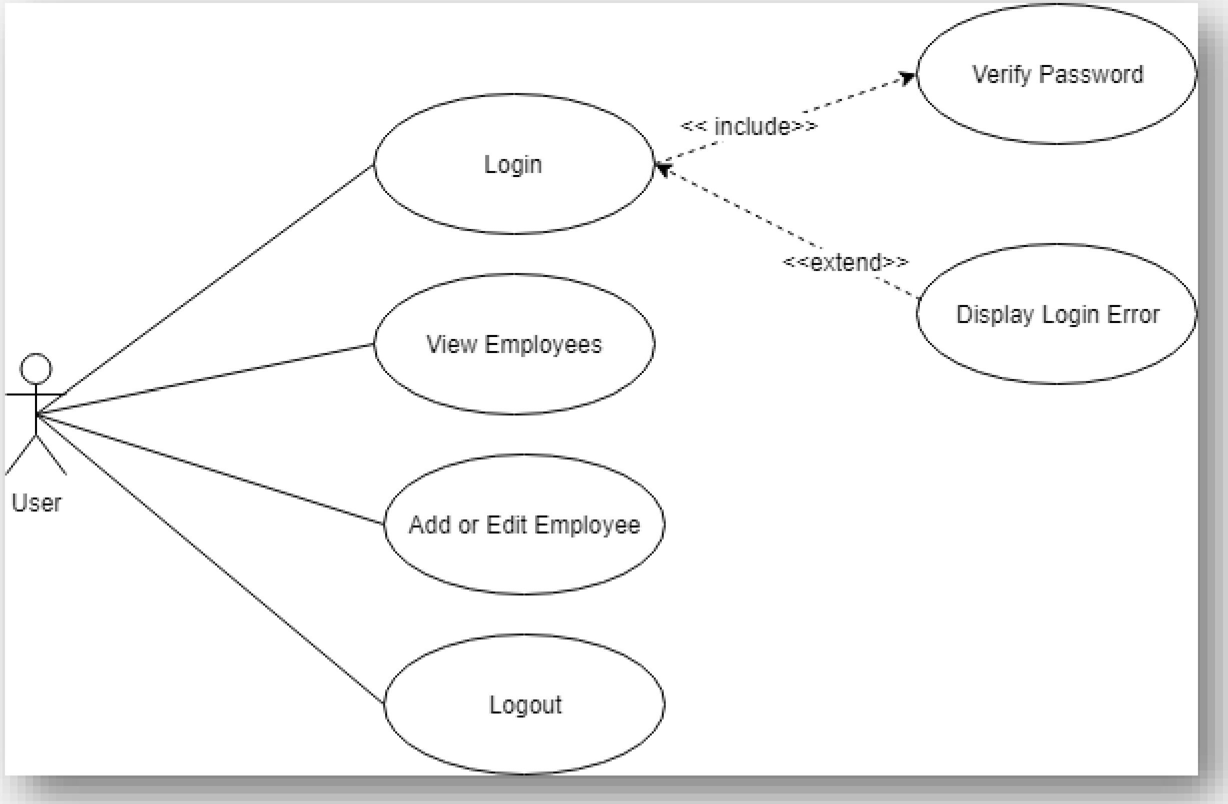
Use Case

- Create a login page for user authentication.

  ❖ Validate the user exists.

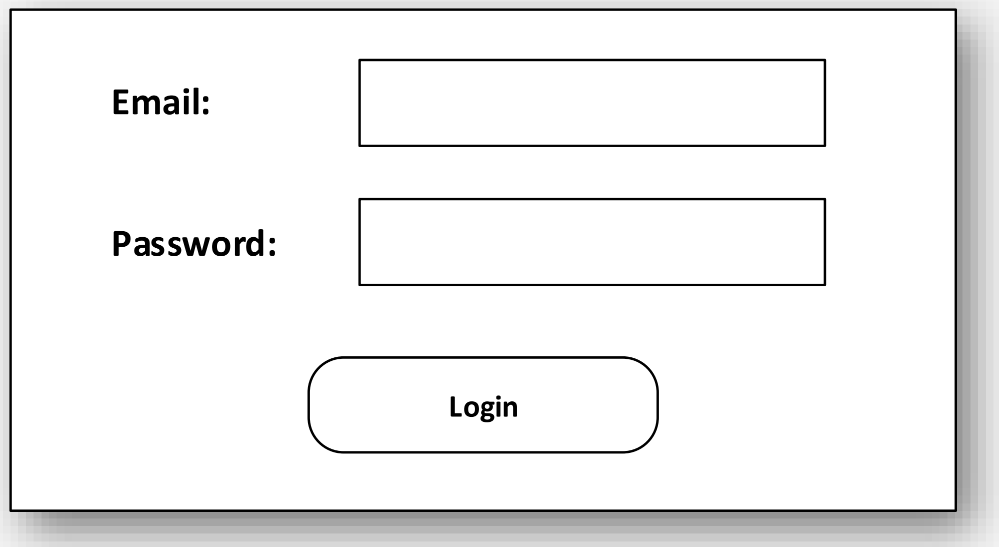
  ❖ Display error if the user does not.

- Upon a successful login, the user shall view a list of employees.
- The user shall have the ability to add new employees.
- The user shall have the ability to update existing employees.



Employee Login Page

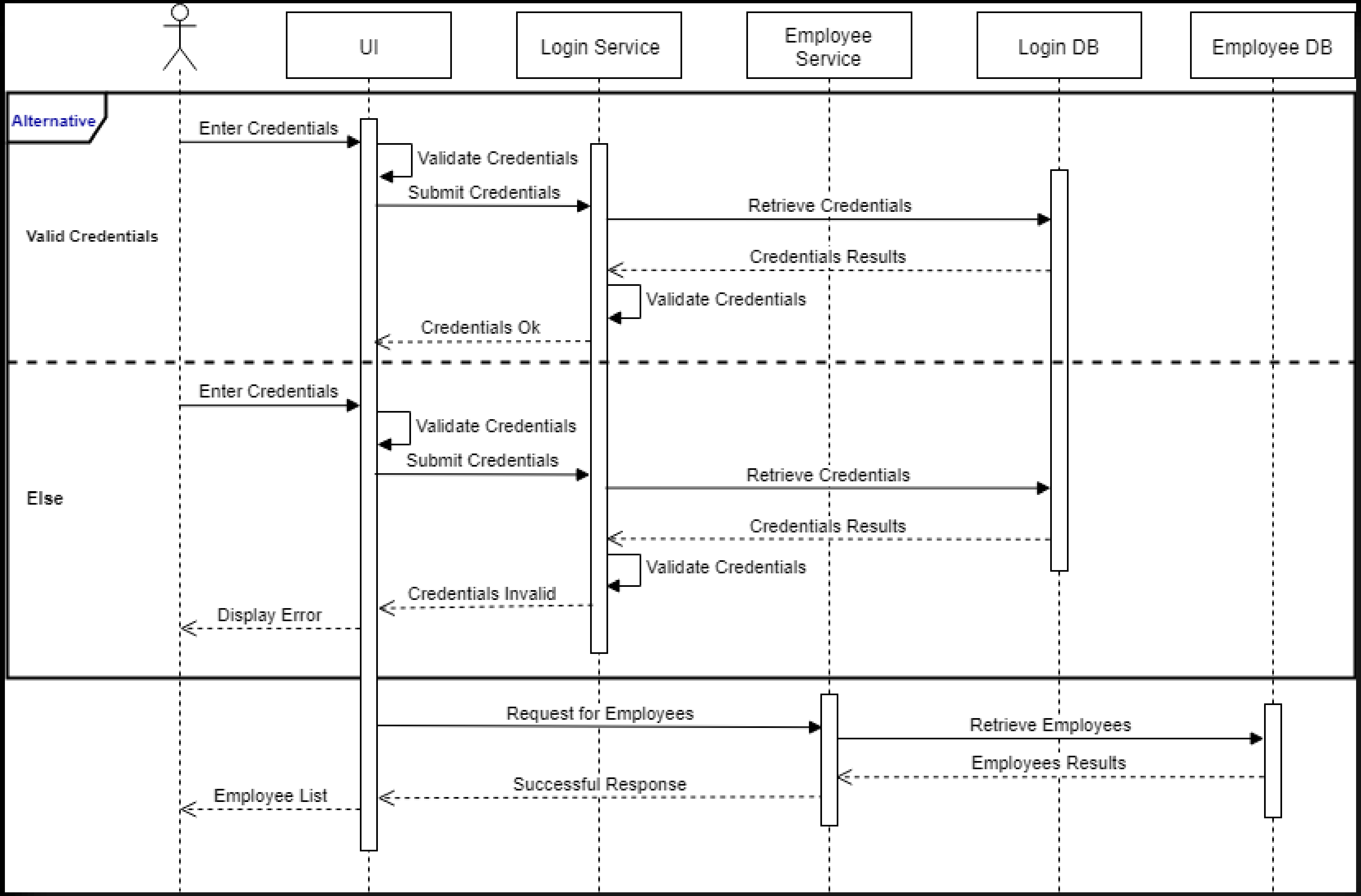
- The user shall not login if the email and password do not meet minimal requirements.
- If the user login is unsuccessful, an error shall display on the login page.



Data Entry Lengths and Validation Information

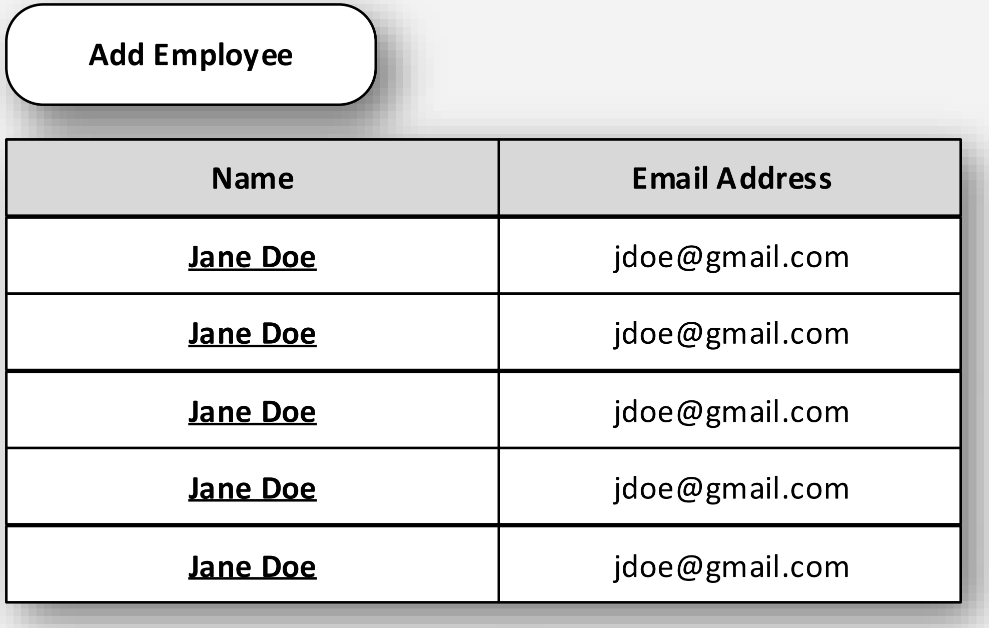
- Email: Minimum 8, Maximum 35 (alpha numeric)
- Password: Minimum 8, Maximum 35 (alpha numeric)

Login Sequence Diagram



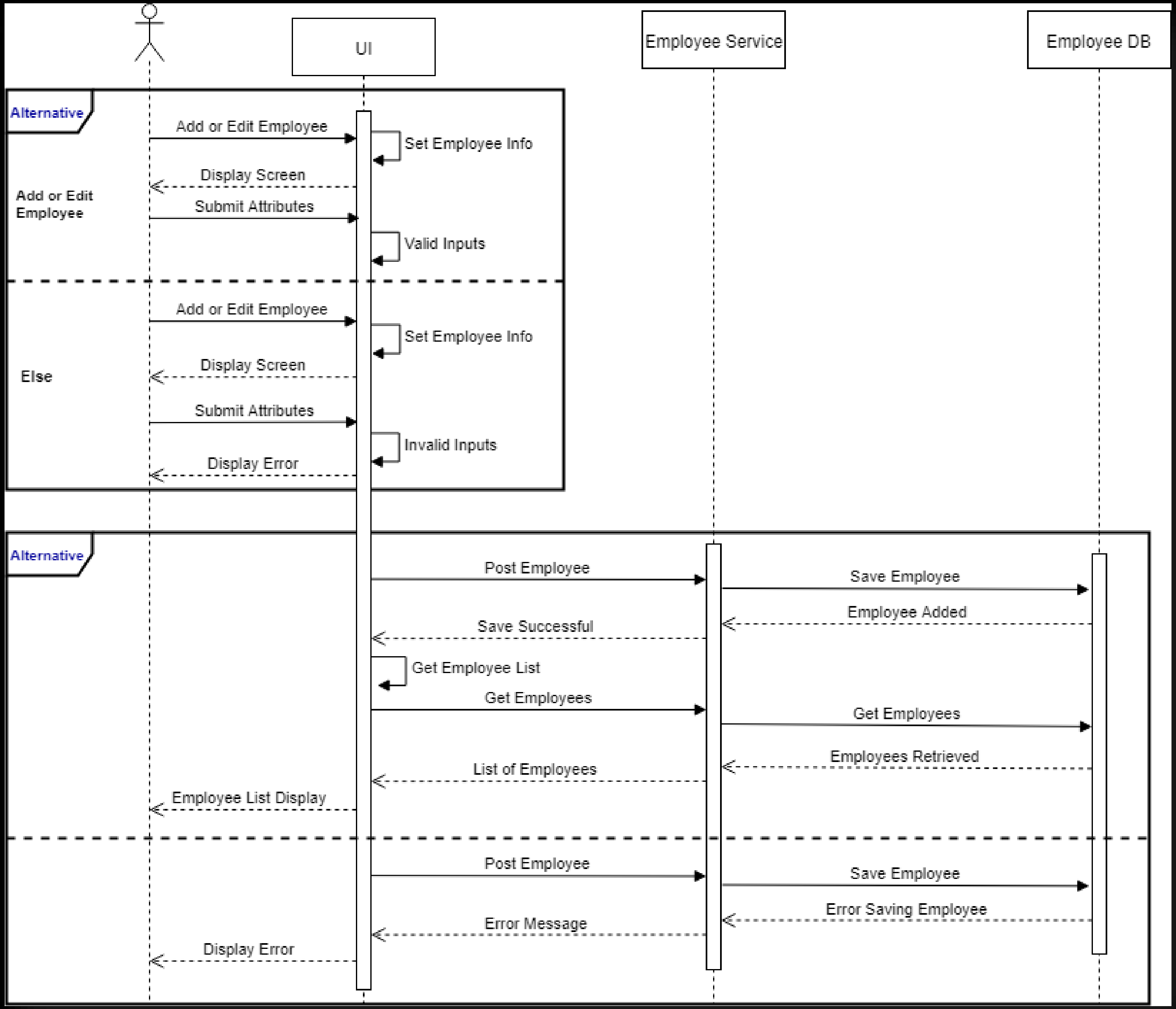
Employee List Page

- Upon a successful login, the application shall display the employee list page.
- The employee list shall be sorted by the employee names.
- The add employee button shall navigate to the employee add/edit page.
- The employee name link shall navigate to the employee add/edit page.



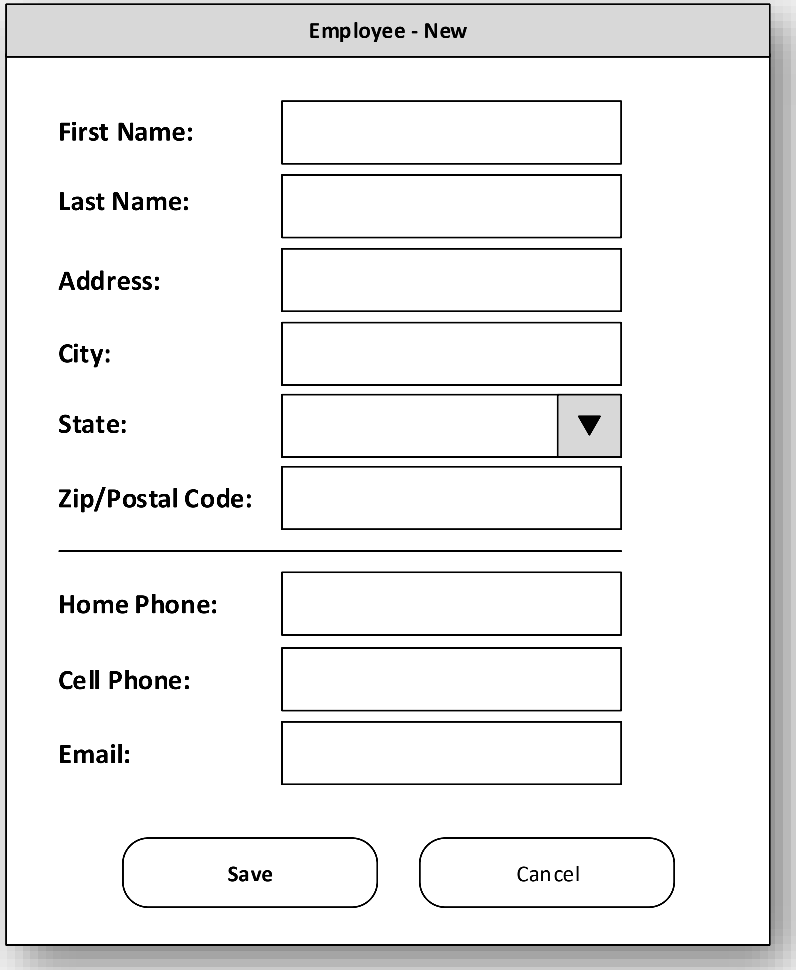
Name	Email Address
Jane Doe	jdoe@gmail.com
Jane Doe	jdoe@gmail.com
Jane Doe	jdoe@gmail.com
Jane Doe	jdoe@gmail.com
Jane Doe	jdoe@gmail.com

Employee Update Sequence Diagram



Employee Add/Edit Page

- All fields are required.
- User cannot save the employee information unless all fields are populated.
- Upon clicking the Cancel button, the user shall navigate to the employee list page.
- Upon clicking the Save button, the entered information shall update the database and navigate to the employee list page.



Employee - New

First Name:

Last Name:

Address:

City:

State: ▼

Zip/Postal Code:

Home Phone:

Cell Phone:

Email:

Data Entry Lengths and Validation Information

- First and Last Names: Minimum 2, Maximum 35 (alpha, spaces allowed)
- Address: Minimum 10, Maximum 50 (alpha numeric , spaces allowed)
- City: Minimum 5, Maximum 50 (alpha , spaces allowed)
- State: Dropdown, value 2 characters (Dropdown)
- Zip/Postal Code: Minimum 5, Maximum 9 (numeric only)
- Home/Cell Phone: 10 characters (numeric only)
- Email: Minimum 10, Maximum 50 (alpha numeric , email validation)

Stacks to Deliver

Angular Front End

- **A**ngular 2+
- **N**odeJS

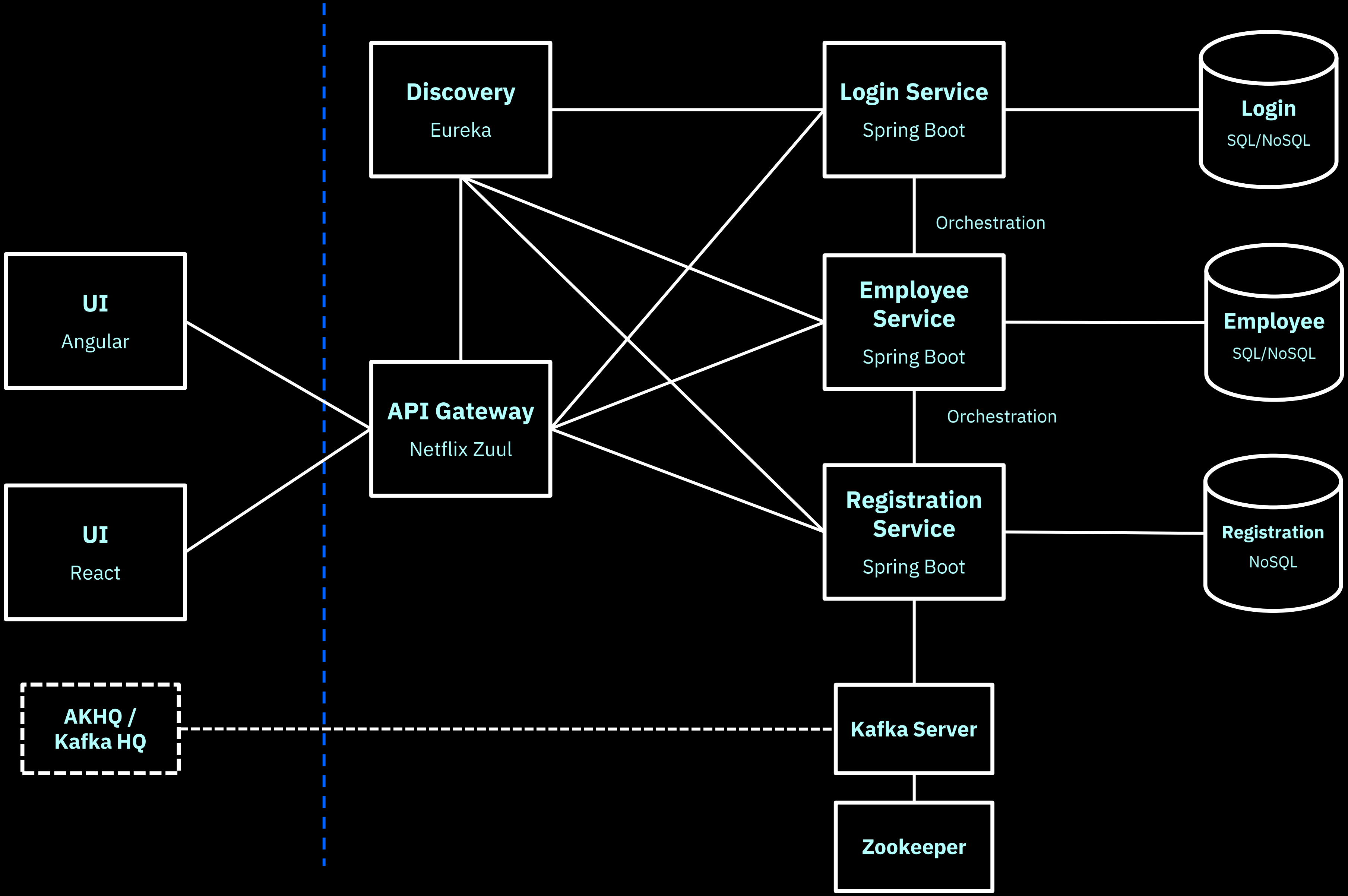
React Front End

- **R**eact
- **N**odeJS

Java Backend

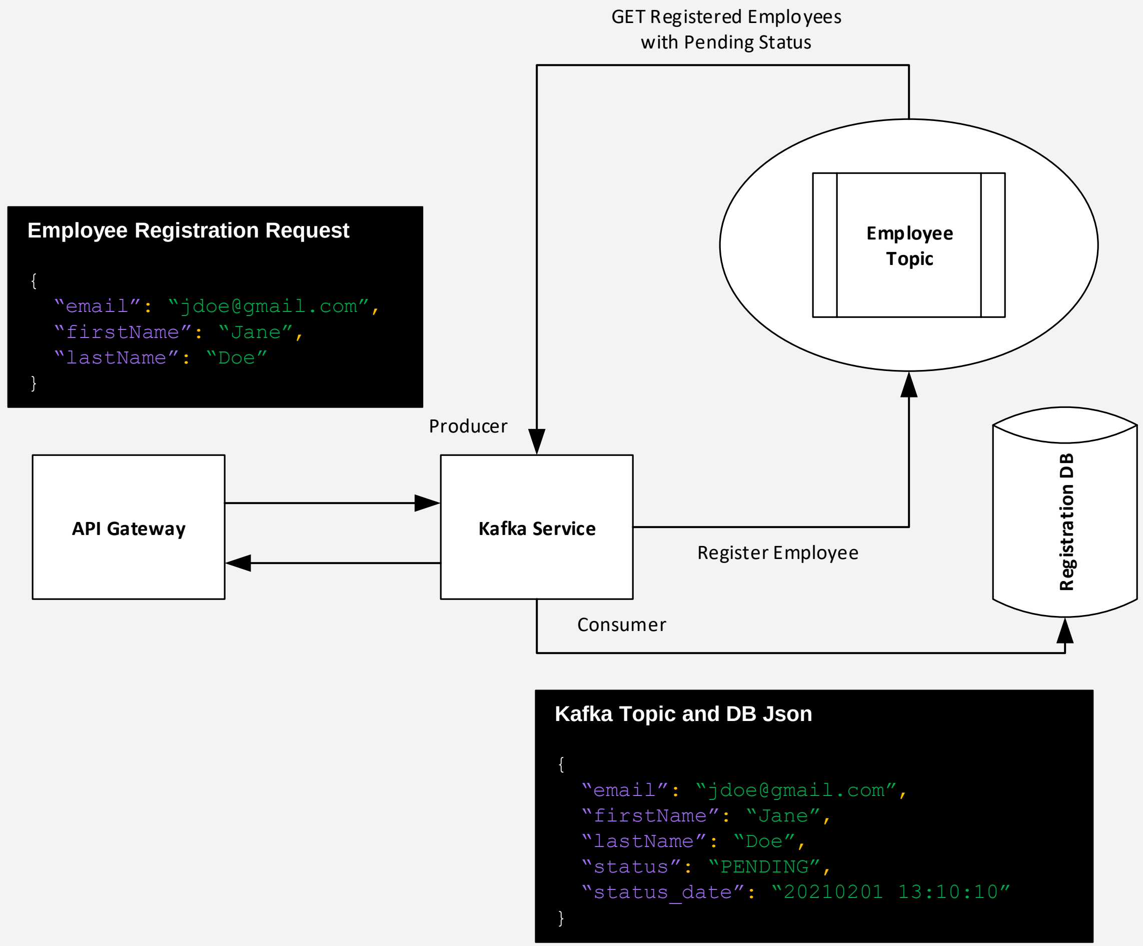
- SQL / NoSQL DB
- Spring Boot
- Apache Kafka
- JPA/Hibernate
- API Gateway
- Service Discovery
- Microservices
  - ❖ Login Service
  - ❖ Employee Service
  - ❖ Kafka Service

High Level Architecture



Employee Registration Service

- Register an employee and add to Kafka Topic with a Pending status.
- Retrieve all employees by status (Approved or Pending) from Kafka topic.
- Retrieve employee by email from Kafka Topic.
- Approve employees with Pending status.
- Decline employees and remove from Kafka Topic
- If the employee status is Approved, the Consumer updates the Registration DB and sends a post request to the Employee Services.



Employee Registration Request

```
{
  “email”: “jdoe@gmail.com”,
  “firstName”: “Jane”,
  “lastName”: “Doe”
}
```

Kafka Topic and DB Json

```
{
  “email”: “jdoe@gmail.com”,
  “firstName”: “Jane”,
  “lastName”: “Doe”,
  “status”: “PENDING”,
  “status_date”: “20210201 13:10:10”
}
```

Technology Exposure

- Docker / Docker Compose
- SpringBoot
- .Net
- JPA
- Hibernate
- NodeJS
- ReactJS
- Angular 2+
- Javascript ES5/ES6
- Jenkins
- Kubernetes
- Apache Kafka
- Cloud Platform: IBM Cloud, AWS, Azure, Google Cloud, RedHat OS.

Deliverables

Before you start …

Prerequisites

- IDEs
  - ➢ Spring Tool Suite IDE
  - ➢ Visual Studio Code
  - ➢ Or, your choice
- JDK 1.8 or later
- NodeJS
- Docker Hub Account: https://hub.docker.com/
- IBM Cloud Account: https://cloud.ibm.com
- IBM GitHub Account: https://github.ibm.com/
- Udemy Course - Git for Geeks: Quick Git Training for Developers (3.5 Hours)
- Fork the Git repository: https://github.ibm.com/jeffreys/full-stack-coding-challenge

Coding stubs are included in git repository.

Deliverables

<table>
<tr><th>Week</th><th>Udemy</th><th>Hours</th><th>Task</th></tr>
<tr><td>1</td><td>• Docker for the Absolute Beginner – Hands On – DevOps
• Microservices with Spring Cloud</td><td>3.5
4.5</td><td>• Create DB Docker Images (Login and Employee)
• Complete Login Service and Containerize
• Complete Employee Service and Containerize
• Create and Run Images with **Docker Compose**
• Test Docker Images (Postman and MySQL Workbench)</td></tr>
<tr><td>2 &amp; 3</td><td>• Microservices with Spring Cloud (continued)</td><td>4.5</td><td>• Implement Eureka Discovery and Zuul API Gateway Services
• Validate Eureka Discover Service identified: Login, Employee, and API Gateway Services.
• Implement Security: oAuth, JWT, etc.
• Create and Run images with **Docker Compose**
• Test Services via Zuul API Gateway</td></tr>
<tr><td>4</td><td>• Kubernetes for the Absolute Beginner – Hands On</td><td>5.5</td><td>• Deploy backend to a Cloud using Kubernetes (or use minikube).</td></tr>
<tr><td colspan="4">**BACKEND DEMONSTRATION**
Backend Service components must be running on a Cloud platform via Kubernetes (or use minikube).</td></tr>
<tr><td>5 &amp; 6</td><td>• Angular Step by Step for beginners
• Hello React - React Training for JavaScript Beginners</td><td>8
6</td><td>• Implement and Containerize Angular UI
• Ensure screen requirements are implemented
• Test Angular UI against service components
• Repeat above steps for the React UI
• Create and Run images with **Docker Compose**</td></tr>
<tr><td>7</td><td>• Kubernetes for the Absolute Beginner – Hands On</td><td>5.5</td><td>• Deploy backend to a Cloud using Kubernetes (or use minikube).</td></tr>
<tr><td colspan="4">**COMPLETE APPLICATION DEMONSTRATION**
UI and Service components must be running on a Cloud platform via Kubernetes (or use minikube).</td></tr>
<tr><td>8</td><td>• Kafka & Kafka Stream With Java Spring Boot - Hands-on</td><td>16.0</td><td>• Refer to the Employee Registration slide for implementation steps</td></tr>
<tr><td colspan="4">**APACHE KAFKA DEMONSTRATION**
UI and Service components must be running on a Cloud platform via Kubernetes (or use minikube).</td></tr>
</table>

Deliverables – Week 1

Databases Implementation

- Create DB Docker Images:
  - Login DB
  - Employee DB
- Run Images in Docker Container
- Test Connectivity to DBs with MySQL Workbench

Services

- Implement Login and Employee Services
- Create Docker Images per Service
- Run services in Docker Container
- Test services with Postman or tool of choice

Push Images to Docker Hub

Weekly deliverables should be committed to your code repository and added to the deployment of the entire application stack using Docker Compose.

Udemy Courses:

- Docker for the Absolute Beginners – Hands On – DevOps
- Microservices with Spring Cloud

Deliverables – Week 2 - 3

API Gateway and Discovery

- Implement Discovery Service
- Implement API Gateway Service and configure to interface with the Discovery Service
- Modify Login and Employee Services to interface with:
  - Discovery Service
  - API Gateway
- Create/Modify Docker Images of Services
- Run services in Docker containers
- Test services with Postman or tool of choice

Push Docker images to Docker Hub

Weekly deliverables should be committed to your code repository and added to the deployment of the entire application stack using Docker Compose.

employee.sql

Udemy Courses:

- Microservices with Spring Cloud (continued)
- Kafka & Kafka Stream With Java Spring Boot - Hands-on Coding

Deliverables – Weeks 4

Now let’s **Journey to the Cloud**

By now, you should have successfully implemented your backend services.

- Implement the Kubernetes yaml files for the backend services.
- Deploy backend services to Minikube.
- Deploy backend services to a Cloud platform using Kubernetes.
- Test services using Postman or tool of choice.

Udemy Courses:

- Kubernetes for the Absolute Beginner – Hands-on

Backend Demonstration

Deliverables – Weeks 5 to 6

UI Development

- Implement UI using Angular
- Create Docker Image of UI
- Run Image in Container

- Implement UI using React
- Create Docker Image of UI
- Run Image in Container

Push Docker images to Docker Hub

At this point, all code should be committed to your code repository. The entire application stack (DB, Services, and UIs) should be deployed by running Docker images and using Docker Compose.

Udemy Courses:
- Full Stack: Angular and Spring Boot (12.5 Hours)
- Go Java Full Stack with Spring Boot and React (11.5 Hours)

Deliverables – Weeks 7

Now let’s **Journey to the Cloud**

By now, you should have successfully accomplished delivering a full-stack application.

- Implement the Kubernetes yaml files for the frontend: Angular and React.
- Deploy the frontend and backend to Minikube.
- Deploy the frontend and backend to a Cloud platform using Kubernetes.
- Test application via the frontend. Angular and React frontends should be running in parallel.

Udemy Courses:

- Kubernetes for the Absolute Beginner – Hands-on

Complete Application Demonstration

Deliverables – Week 8

Kafka Service

- Create Docker Images using Docker Compose:
  - Implement Kafka Server
    - Topics / Partitions
    - Producer / Consumer
  - Implement Zookeeper
  - Implement AKHQ / KafkaHQ (optional)
  - Kafka Service
  - Kafka DB
- See Employee Registration slide for steps
- Run services in Docker containers
- Test services with Postman or tool of choice

Push Docker images to Docker Hub

Weekly deliverables should be committed to your code repository and added to the deployment of the entire application stack using Docker Compose.

employee.sql

Udemy Courses:

- Kafka & Kafka Stream With Java Spring Boot - Hands-on Coding (continued)

Bonus Deliverables

Jenkins Deliverable

With Jenkins, build a pipeline to:

- Checkout code base from Git Repository
- Compile Code (Java projects)
- Build Docker Images from code base
- Launch application with Docker Compose or Kubernetes
  - Databases
  - Services
  - UI (Angular or React)

Udemy Courses:

- Jenkins 2 Bootcamp: Fully Automate Builds to Deployment 2019

Other Tutorials and References

Angular: https://angular.io/tutorial/

ReactJS: https://reactjs.org/

Spring Initializer: https://start.spring.io/

Spring Boot: https://spring.io/projects/spring-boot/

Tutorials Point: http://www.tutorialspoint.com/

W3 Schools: https://w3schools.com/