

# Self-Assessment and Personalized Learning (SAPL) System

## Team Members

Alfeshkhi, Hassan

Islam Riad, A B M Kamrul

Islam, Md Saiful

Kommineni, Prathyusha

IT 7933 Capstone project  
KSU & Digi Safari



11/25/2020

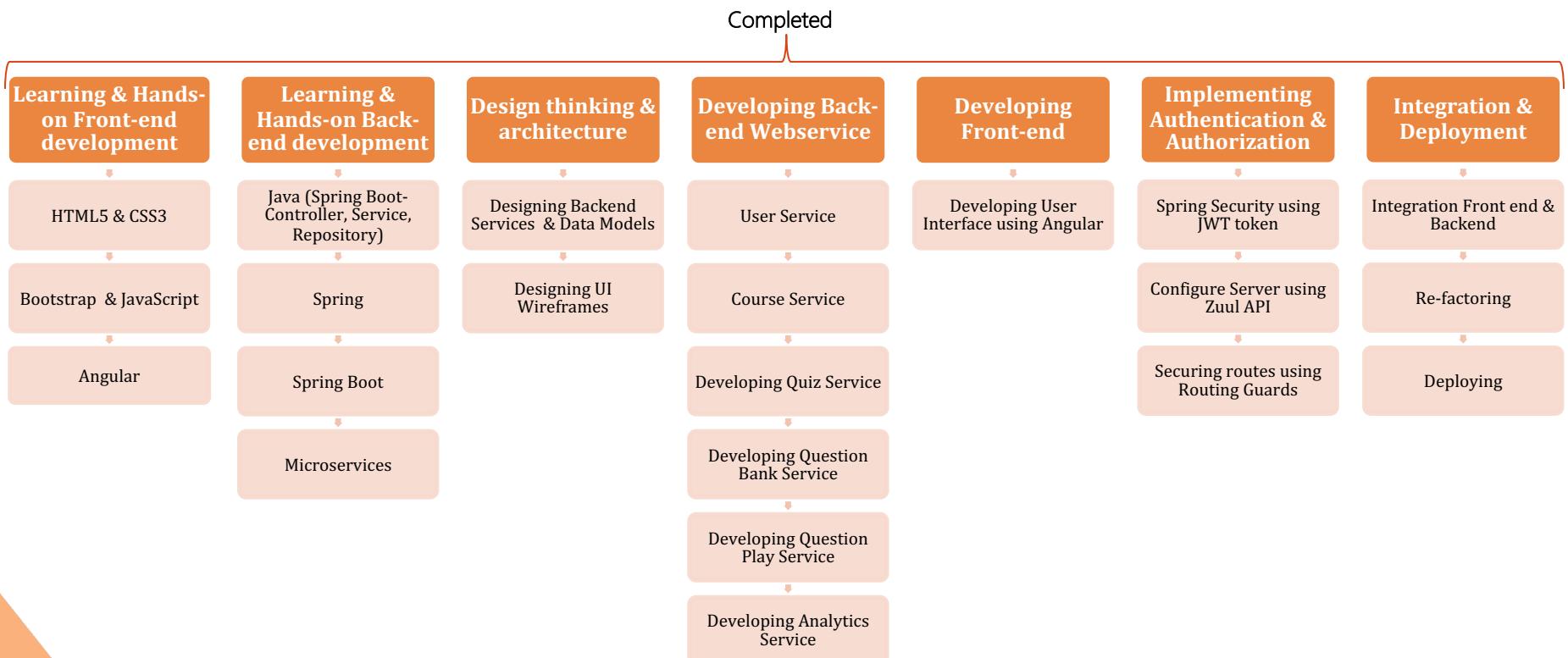
# Digi-Safari (DS)

- Digi-Safari (DS) provides software training and services to corporate clients looking to re-skill/ Up-skill existing/ new hires under the B2B (Business-Business) model. DS wants to extend the service for the B2C (Business-Consumer) segment, where individual learners can also directly register for self-paced training.
- They also provide training programs in different universities like us and help students to boost programming knowledge by working in a real-world project. That helps to kick start their career in the IT industry.

# Project background

- To better assist learners in assessing their current skill level and pick the best-fit learning path, DS planned to develop a Self-Assessment and Personalized Learning-path (SAPL) System.
- SAPL provide an on-line self-assessment tool through a series of questions, which will auto-select the next question based on response from previous questions equivalent to the GRE type test.
- Based on the performance and candidate's career goal, SAPL automatically come up with Personalized Learning-path. Candidates then have the option of enrolling through Digi-Safari training programs.

# SAPL Project Summary



## Learning & Hands-on Front-end development

### HTML 5 & CSS3

- Web Development fundamentals, Creating Simple web page
- CSS3 Flex Box, Responsive Web Design
- Implement Font-Awesome Icons

### Our Team

Our greatest strength is our team spirit & that will remain because a core of us come through the leagues together.



Magna  
Founder & CEO  
[f](#) [in](#) [o](#) [G+](#)



Ellen Hassan  
Founder & COO  
[f](#) [in](#) [o](#) [G+](#)



John Fairley  
Vice President  
[f](#) [in](#) [o](#) [G+](#)

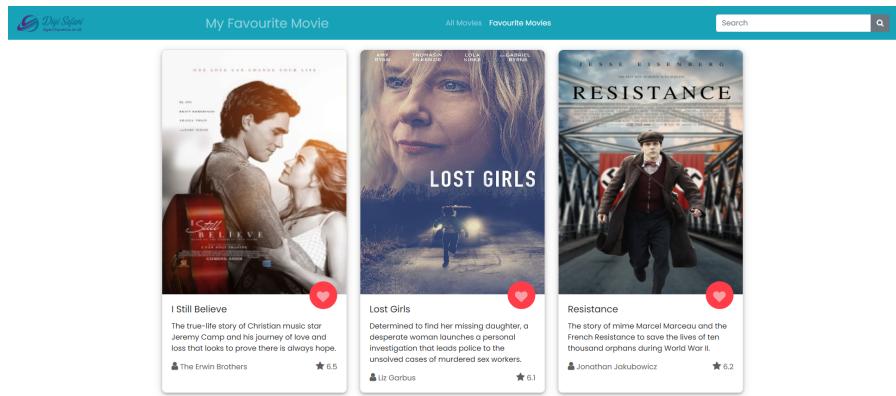


Anni Gudorf  
Marketing Manager  
[f](#) [in](#) [o](#) [G+](#)

## Learning & Hands-on Front-end development

### Bootstrap & JavaScript

- Bootstrap and its components (NavBar, GridSystem, Form, Card)
- Fetch API with GET, POST, DELETE HTTP requests
- Promise API ( usage of resolve, reject, then, catch functions)
- json-server, http-server
- Implement Font-Awesome Icons



# Learning & Hands-on Front-end development

## Angular

- Working with Angular Architecture, Modules, Components, Template Data Binding, Event Handling, Angular Material
- Working With Services, Working with HTTP API calls, GET, POST,PUT,DELETE,PATCH
- Angular Forms, Template Driven, Reactive Forms, Routing, Guards, JWT Security

The screenshot shows a web-based application titled "Google Keep Clone". The interface includes a top navigation bar with a "Menu" icon and the title "Google Keep Clone". On the left, there's a sidebar with icons for "Notes" (highlighted), "Reminder", "Categories" (with "Product", "Meeting", and "News" options), "Log In", and "Sign Up". The main area has a "Create a note" form with fields for "Note title" and "Note description", and a "Create" button. Below the form is a grid of five cards representing tasks:

Project Review Meeting sdf Need to attend Project Review Meeting with Project manager	Unit Test Cases for User Module Writing unit test cases for user module	Updating Angular CLI to 10.0.7 Updating Angular CLI from 8.1.0 to 10.0.7	Read Angular10 blog Needs to read Angular10 new features blog	Project Meeting-completed Project meeting is completed today discussed on architecture
--	--	---	--	---

# Learning & Hands-on Front-end development

## Code Review - Refactoring

- Code Review on Angular
- Improve readability
- Increase scalability

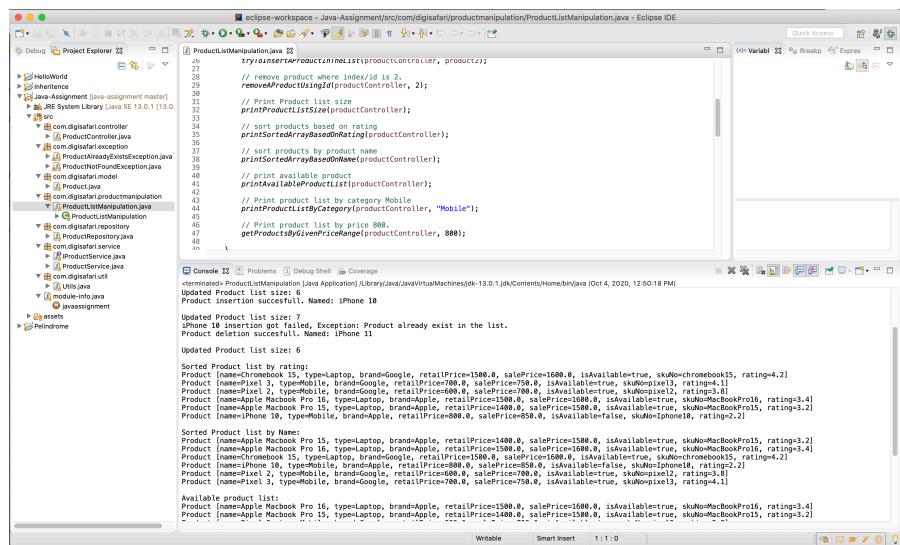
The screenshot shows a user interface for a note-taking application, identified as a "Google Keep Clone". The top navigation bar is blue with the title "Google Keep Clone". On the left, there's a sidebar with icons for "Notes" (yellow), "Reminder" (orange), "Categories" (grey), and lists for "Product", "Meeting", and "News". Below the sidebar are "Log In" and "Sign Up" buttons. The main area has a form titled "Create a note" with fields for "Note title" and "Note description", and a "Create" button. At the bottom, there are five cards representing tasks:

- Project Review Meeting sdf**  
Need to attend Project Review Meeting with Project manager
- Unit Test Cases for User Module**  
Writing unit test cases for user module
- Updating Angular CLI to 10.0.7**  
Updating Angular CLI from 8.1.0 to 10.0.7
- Read Angular10 blog**  
Needs to read Angular10 new features blog
- Project Meeting-completed**  
Project meeting is completed today discussed on architecture

# Learning & Hands-on Front-end development

## Java

- Java Fundamentals, Working with Primitives, Control Statements
- Working with classes & Objects, constructors, static members, methods
- Object Oriented Programming Features, Exception Handling
- Working with Collections



The screenshot shows the Eclipse IDE interface with the following details:

- Project Explorer:** Shows a Java project named "Java-Assessment" with packages like com.digisafarilabs, com.digisafarilabs.productmanipulation, and com.digisafarilabs.productrepository.
- Code Editor:** Displays the file `ProductListManipulation.java` containing Java code for manipulating a list of products based on rating, price, or category.
- Output Console:** Shows the execution results of the code, including product insertion, deletion, and sorting operations.
- Properties View:** Shows the properties of the selected Java file.
- Search View:** Shows search results for the word "Product".
- Help View:** Shows help documentation for the Java Virtual Machine.

```

eclipse-workspace - Java-Assessment/src/com/digisafarilabs/productmanipulation/ProductListManipulation.java - Eclipse IDE

try{
    insertProductInArrayList(productController, product);
} catch(NoSuchElementException e) {
    removeProductWhereIndexIsIn2(productController, 2);
}
removeProductUsingArrayList(productController);
printProductListSize(productController);
printProductListSize(productController);
sort products based on rating
printSortedArrayBasedOnRating(productController);
sort products by product name
printSortedArrayBasedOnName(productController);
print available product
print product list by category Mobile
printProductListByCategory(productController, "Mobile");
Print product list by price 800
getProductsByGivenPriceRange(productController, 800);
}

Updated Product list size: 6
Product insertion successful. Named: iPhone 10
Updated Product list size: 7
Product deletion successful. Named: iPhone 10
Product deletion successful. Named: iPhone 10
Updated Product list size: 6

Sorted Product list by rating:
Product [name=Chromebook 15, type=laptop, brand=Google, retailPrice=1500.0, salePrice=1000.0, isAvailable=true, skuNo=Chromebook15, rating=4.2]
Product [name=Pixel 2, type=Mobile, brand=Google, retailPrice=700.0, salePrice=700.0, isAvailable=true, skuNo=Pixel12, rating=3.0]
Product [name=Pixel 3, type=Mobile, brand=Google, retailPrice=800.0, salePrice=700.0, isAvailable=true, skuNo=Pixel13, rating=3.8]
Product [name=Pixel 3a, type=Mobile, brand=Google, retailPrice=600.0, salePrice=600.0, isAvailable=true, skuNo=Pixel14, rating=3.2]
Product [name=iPhone 10, type=Mobile, brand=Apple, retailPrice=800.0, salePrice=950.0, isAvailable=false, skuNo=iPhone10, rating=2.1]
Product [name=iPhone 11, type=Mobile, brand=Apple, retailPrice=1000.0, salePrice=1000.0, isAvailable=false, skuNo=iPhone11, rating=2.1]

Sorted Product list by Name:
Product [name=Apple MacBook Pro 15, type=laptop, brand=Apple, retailPrice=1500.0, salePrice=1500.0, isAvailable=true, skuNo=MacBookPro15, rating=3.2]
Product [name=Apple MacBook Pro 16, type=laptop, brand=Apple, retailPrice=1500.0, salePrice=1500.0, isAvailable=true, skuNo=MacBookPro16, rating=3.4]
Product [name=Chromebook 15, type=laptop, brand=Google, retailPrice=1500.0, salePrice=1000.0, isAvailable=true, skuNo=Chromebook15, rating=4.2]
Product [name=Pixel 2, type=Mobile, brand=Google, retailPrice=700.0, salePrice=700.0, isAvailable=true, skuNo=Pixel12, rating=3.0]
Product [name=Pixel 3, type=Mobile, brand=Google, retailPrice=800.0, salePrice=700.0, isAvailable=true, skuNo=Pixel13, rating=3.8]
Product [name=Pixel 3a, type=Mobile, brand=Google, retailPrice=600.0, salePrice=600.0, isAvailable=true, skuNo=Pixel14, rating=3.2]
Product [name=iPhone 10, type=Mobile, brand=Apple, retailPrice=800.0, salePrice=950.0, isAvailable=false, skuNo=iPhone10, rating=2.1]
Product [name=iPhone 11, type=Mobile, brand=Apple, retailPrice=1000.0, salePrice=1000.0, isAvailable=false, skuNo=iPhone11, rating=2.1]
Product [name=Apple MacBook Pro 16, type=laptop, brand=Apple, retailPrice=1500.0, salePrice=1500.0, isAvailable=true, skuNo=MacBookPro16, rating=3.4]
Product [name=Apple MacBook Pro 15, type=laptop, brand=Apple, retailPrice=1500.0, salePrice=1500.0, isAvailable=true, skuNo=MacBookPro15, rating=3.2]

```

## Design thinking & architecture of the capstone project

### Design thinking & architecture

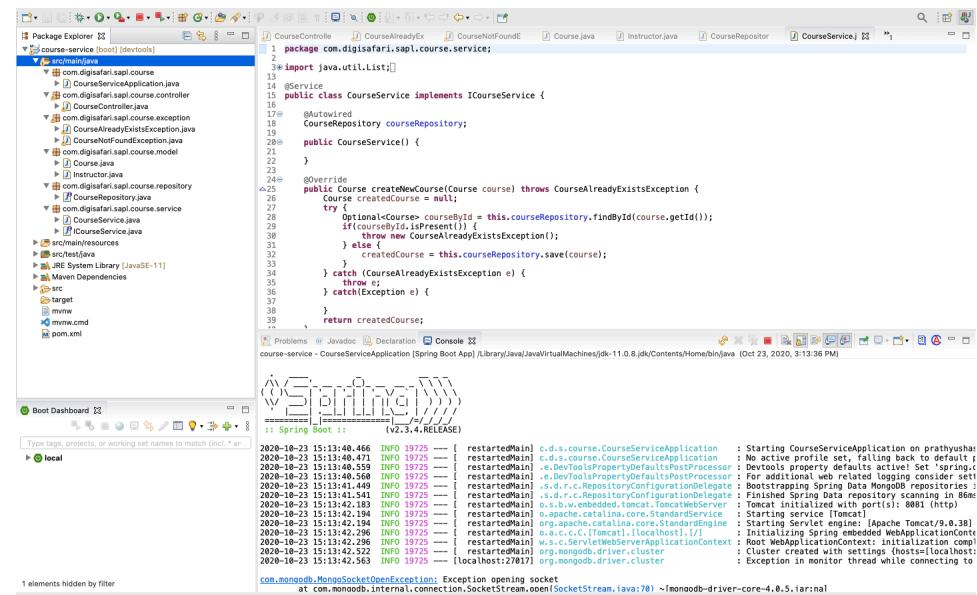
- Design Thinking, Domain Driven Design
- Designing Backend Services, Data Models
- Wireframes

END POINTS	HTTP REQUEST	DESCRIPTION
<a href="http://localhost:8084/api/v1/Quiz">http://localhost:8084/api/v1/Quiz</a>	GET	Should fetch all the quizzes
<a href="http://localhost:8084/api/v1/Quiz/{QuizId}">http://localhost:8084/api/v1/Quiz/{QuizId}</a>	GET	Should fetch specific quiz with provided id
<a href="http://localhost:8084/api/v1/Quiz/{QuizId}/CreateQuiz">http://localhost:8084/api/v1/Quiz/{QuizId}/CreateQuiz</a>	POST	Should create a new quiz
<a href="http://localhost:8084/api/v1/Quiz/{QuizId}/UpdateQuiz">http://localhost:8084/api/v1/Quiz/{QuizId}/UpdateQuiz</a>	POST	Should update the existing quiz
<a href="http://localhost:8084/api/v1/Quiz/{QuizId}/DeleteQuiz">http://localhost:8084/api/v1/Quiz/{QuizId}/DeleteQuiz</a>	GET	Should delete the quiz

## Learning & Hands-on Back-end development(Tools and Coding)

### Spring

- Introduction to Spring Framework, Core Container, Dependency Injection, Spring MVC



```

package com.digisafari.sapi.course;
import java.util.List;
@Service
public class CourseService implements ICourseService {
    @Autowired
    CourseRepository courseRepository;
    public CourseService() {
    }
    @Override
    public Course createNewCourse(Course course) throws CourseAlreadyExistsException {
        try {
            Optional<Course> courseById = this.courseRepository.findById(course.getId());
            if(courseById.isPresent())
                throw new CourseAlreadyExistsException();
            else {
                createdCourse = this.courseRepository.save(course);
            }
        } catch (CourseAlreadyExistsException e) {
            throw e;
        } catch (Exception e) {
            return null;
        }
        return createdCourse;
    }
}

```

The screenshot shows the Eclipse IDE interface with the 'course-service' project open in the Package Explorer. The 'course-controller' package contains the 'CourseController.java' file, which is displayed in the editor. The code implements the 'ICourseService' interface and uses Spring's @Service and @Autowired annotations. It checks if a course with the same ID already exists in the database and saves the new course if not.

### Spring Boot

- Testing RESTful Web Services, Introduction to CI, Implementing Gitlab CI pipelines for RESTful Webservices

1 elements hidden by filter

2020-10-23 15:13:40.466 INFO 19725 --- [ restartedMain] c.d.s.course.CourseServiceApplication : Starting CourseServiceApplication on prathushashish-OptiPlex-5090-10144C with PID 19725 (started by root in /home/prathushashish)
No active profile set, falling back to default: 'dev'
To display the application context's beans, set the 'spring-boot.run.profiles' system property to 'dev'.
For additional web related logging consider setting 'logging.level.web=DEBUG'.

2020-10-23 15:13:40.558 INFO 19725 --- [ restartedMain] o.DevToolsPropertyDefaultsPostProcessor : Bootstrapping Spring Data MongoDB repositories ...

2020-10-23 15:13:41.449 INFO 19725 --- [ restartedMain] s.d.r.c.RepositoryConfigurationDelegate : Bootstrapping Spring Data JPA repositories ...

2020-10-23 15:13:42.183 INFO 19725 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat initialized with port(s): 8081 (http)

2020-10-23 15:13:42.194 INFO 19725 --- [ restartedMain] o.apache.catalina.core.StandardService : Starting service [Tomcat]

2020-10-23 15:13:42.196 INFO 19725 --- [ restartedMain] o.apache.catalina.core.StandardEngine : Starting Servlet Engine: [Apache Tomcat/9.0.38]

2020-10-23 15:13:42.296 INFO 19725 --- [ restartedMain] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat created with settings: [localhost:8081]

2020-10-23 15:13:42.563 INFO 19725 --- [localhost:27017] org.mongodb.driver.cluster : Cluster created with settings: [hosts:[localhost:27017],replicaSetName:<none>]

com.mongodb.MongoSocketOpenException: Exception opening socket
at com.mongodb.internal.connection.SocketStream.open(SocketStream.java:70) ~[mongodb-driver-core-4.0.5.jar:na]

## Learning & Hands-on Back-end development(Tools and Coding)

### Spring MVC

- Spring MVC, Developing a monolithic application using Spring – Hibernate

### Spring Cloud - Microservices

- Introduction to Microservices, Monolith vs Microservice, Netflix Cloud, Spring Cloud, Eureka Service Registry & Discovery,

## Structure of Project Architecture

### Developing User and Quiz Service

- User Login and Sign Up
- User can browse the courses
- User can see the course topics
- User can attempt the Quiz
- User can see the attempted quizzes
- User can access personalized Learning path based on the quiz result

The screenshot shows a Java Spring Boot application structure in the Package Explorer:

- User** package contains the **User.java** model class.
- bootstrap** package contains configuration files.
- controller** package contains controllers for User, Course, Quiz, and APIS.
- entity** package contains entities for User, Course, Quiz, and APIS.
- repository** package contains repositories for User, Course, Quiz, and APIS.
- service** package contains services for User, Course, Quiz, and APIS.
- utils** package contains utility classes.

**User.java** code:

```

1 package com.stackroute.springboot.model;
2
3 import org.springframework.data.annotation.Id;
4
5 @Document(collection = "users")
6 public class User {
7
8     @Id
9     private String id;
10    private String name;
11    private String username;
12    private String password;
13    public String getId() {
14        return id;
15    }
16    public void setId(String id) {
17        this.id = id;
18    }
19    public String getName() {
20        return name;
21    }
22    public void setName(String name) {
23        this.name = name;
24    }
25    public String getUsername() {
26        return username;
27    }
28    public void setUsername(String username) {
29        this.username = username;
30    }
31    public String getPassword() {
32        return password;
33    }
34    public void setPassword(String password) {
35}

```

**Terminal** output:

```

<terminated>: C:\Users\lyzen\Downloads\Programs\spring-tool-suite-4-4.3.0.RELEASE-e4.17.0-win32-x64_64_bit-extracting\contents\sts-4.8.0.RELEASE\plugins\org.eclipse.jdt\openjdk.hotspot.jre.full.win32.x86_64_141
[INFO] Self Assessment Personalized Learning Path ..... SUCCESS [ 0.079 s]
[INFO] SAPL-Course-Service ..... SUCCESS [ 1.508 s]
[INFO] SAPL-User-Service ..... SUCCESS [ 0.174 s]
[INFO] SAPL-Quiz-Service ..... SUCCESS [ 0.163 s]
[INFO] SAPL-Quiz-Play-Service ..... SUCCESS [ 0.163 s]
[INFO] SAPL-APIS-Gateway ..... SUCCESS [ 0.245 s]
[INFO] SAPL-Naming-Server ..... SUCCESS [ 0.218 s]

```

## Structure of Project Architecture

### Developing Question Bank Service (Quiz Service)

- Should allow to create quiz
- Should fetch all the created quizzes
- Should allow to update a quiz
- Should deactivate a quiz
- Should fetch the details for the quiz
- Should fetch the statistics of the quiz

The screenshot shows an IDE interface with the following components:

- Package Explorer:** Shows the project structure with several modules: `sapl-api-gateway`, `sapl-authentication-service`, `sapl-config-server`, `sapl-course-service`, `sapl-naming-server`, `sapl-quiz-play-service`, `sapl-quiz-service`, `sapl-user-service`, `self-assessment-personalized-learning-path`, and `user-service`.
- Code Editor:** Displays the `Quiz.java` class code. The class has fields for id, title, question type, marks, negative marks, taxonomy level, creation date, update date, incorrect answer explanation, and option. It includes a constructor and a no-argument constructor.
- Terminal:** Shows the output of a build command. The logs indicate successful builds for various services: `Sapl-Assessment Personalized Learning Path`, `Sapl-Course-Service`, `Sapl-User-Service`, `Sapl-Quiz-Play-Service`, `Sapl-Quiz-Service`, `Sapl-API-Gateway`, and `Sapl-Naming-Server`. Each log entry shows a timestamp and a success status.

## Developing Back-end Webservice

### Developing Quiz Play Service

- A user should be able to take / attempt quiz for any particular course
- A user should be able to see his score, preview his quiz, and the description for the wrong answer.
- User should fetch the next question based on the previous answered question (It should increase/decrease the bloom's taxonomy level)

### Authentication Service

- Should implement user authentication & authorization

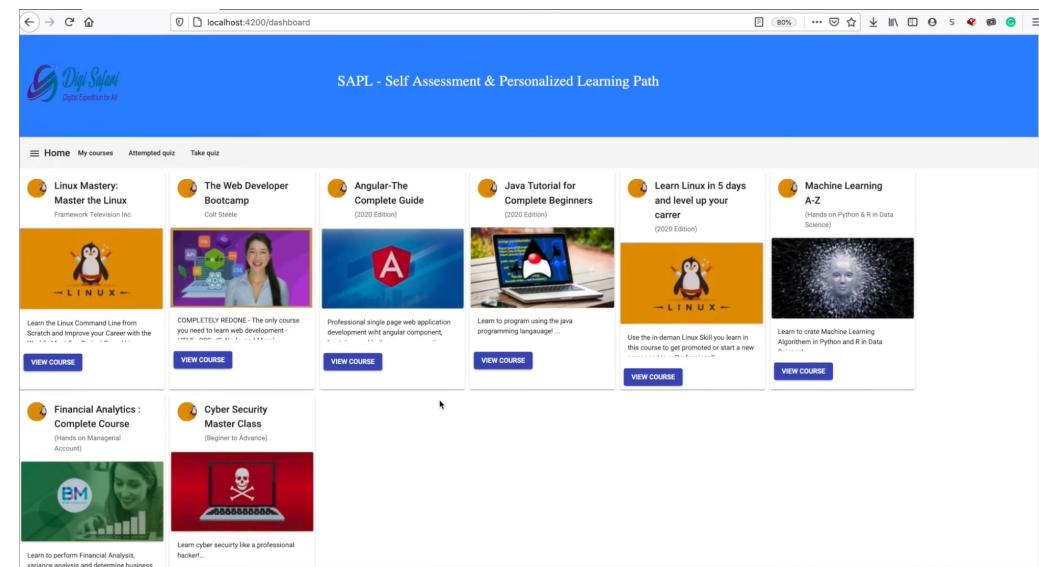
### Course Service

- Should allow the CRUD operation on the course

# Developing Front-end Interface

## Developing UI using Angular part 1

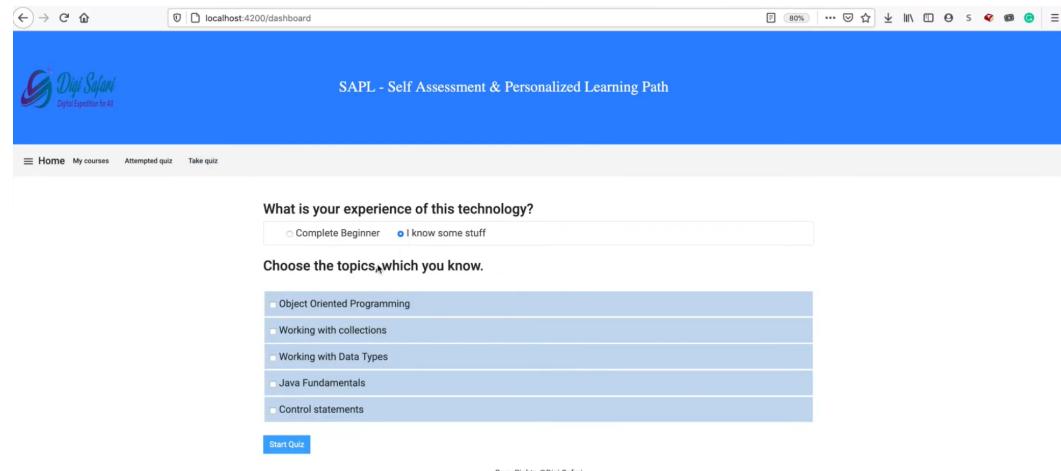
- Developing services to consume backend Quiz Service end points
- Developing services to consume backend Question Bank Service End points
- Developing the Front-end UI screen to replicate the functionality of quiz and quiz play services



## Developing Front-end Interface

### Developing UI using Angular part 2

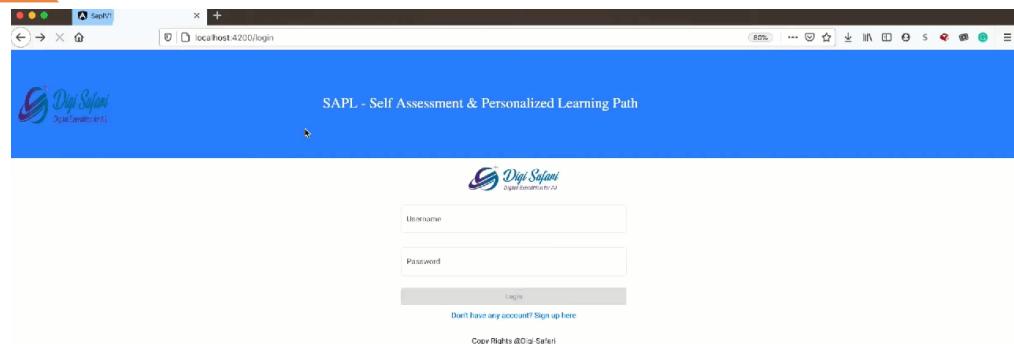
- Developing services to consume backend Quiz Play Service end points
- Developing services to consume backend Analytics Service End points
- Developing the Front-end UI screen to replicate the functionality of Quiz play and Analytics services



# Implementation

## Implementing Authentication & Authorization

- Implement authentication service using JWT token
- Implementing Authentication & Authorization using Zuul API Gateway
- Securing routes in Angular using Routing Guards



## Integration & Deployment

### Integration & Deployment

- Implement unit testing and e2e testing for the whole application and automating through Gitlab CI pipelines
- Containerize the application using Docker
- Finally, package and deploy the application to Kubernetes

# Project Video Demo

The screenshot shows a web browser window titled "SapiV1" with the URL "localhost:4200/login". The page has a blue header with the "Digi Safari" logo and the text "SAPL - Self Assessment & Personalized Learning Path". The main content area contains two input fields for "Username" and "Password", a "Login" button, and a link "Don't have any account? Sign up here".

SAPL - Self Assessment & Personalized Learning Path

Digi Safari  
Digital Expedition for All

Username

Password

Login

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

Copy Rights @Digi-Safari



**Thank You!**