



# IIITB Summer Internship Program (ISIP) 2022

Rapid Software Engineering Process, Tools and Techniques

Professor Chandrashekar Ramanathan

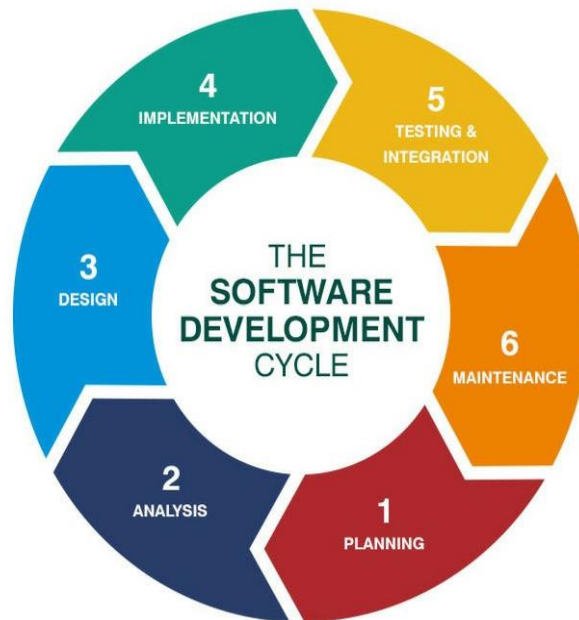
Aman Gupta (MT2021012)

Gopal Goyal (MT2021048)

Vishal Govil (MT2021152)

## Motivation

Majority of the applications in the real world follows the SDLC (Software Development Life Cycle) i.e. The applications go through different phases of the life cycle model.



Although the current SDLC process has been through substantial changes for the better, there are still certain areas which we found that can be further improved: -

- As the development progresses it has been observed that there is a significant variance between the initial plan and the final product.
- When the project moves forward, if there are any new features to be added or altered. The process can be slow due to interdependency among different phases of the lifecycle which required a lot of human interaction.

## Approach

To minimise the human interaction, we discovered that a proper knowledge of internal designing is required for the development of the application. So, we found that among the most popular tools available in the market, UML was the best fit.

## What is UML?

The UML stands for Unified modelling language, is a standardized general-purpose visual modelling language in the field of Software Engineering. It is used for specifying, visualizing,

constructing, and documenting the primary artifacts of the software system. It helps in designing and characterizing, especially those software systems that incorporate the concept of Object orientation. It describes the working of both the software and hardware systems.

## **Diagrams in UML?**

### **Structure Diagrams**

- Class Diagram
- Component Diagram
- Deployment Diagram
- Object Diagram
- Package Diagram
- Profile Diagram
- Composite Structure Diagram

### **Behavioural Diagrams**

- Use Case Diagram
- Activity Diagram
- State Machine Diagram
- Sequence Diagram
- Communication Diagram
- Interaction Overview Diagram
- Timing Diagram

### **Class Diagram**

Class diagrams are the main building block of any object-oriented solution. It shows the classes in a system, attributes, and operations of each class and the relationship between each class.

### **Component Diagram**

A component diagram displays the structural relationship of components of a software system. These are mostly used when working with complex systems with many components. Components communicate with each other using interfaces.

### **Deployment Diagram**

A deployment diagram shows the hardware of your system and the software in that hardware. Deployment diagrams are useful when your software solution is deployed across multiple machines with each having a unique configuration.

## Object Diagram

Object Diagrams, sometimes referred to as Instance diagrams are very similar to class diagrams. Like class diagrams, they also show the relationship between objects but they use real-world examples.

## Package Diagram

As the name suggests, a package diagram shows the dependencies between different packages in a system.

## Profile Diagram

Profile diagram is a new diagram type introduced in UML 2. This is a diagram type that is very rarely used in any specification.

## Composite Structure Diagram

Composite structure diagrams are used to show the internal structure of a class.

## Use Case Diagram

A **use case diagram** is a graphical depiction of a user's possible interactions with a system. A use case diagram shows various use cases and different types of users the system has and will often be accompanied by other types of diagrams as well. The use cases are represented by either circles or ellipses.

## Activity Diagram

Activity diagrams represent workflows in a graphical way. They can be used to describe the business workflow or the operational workflow of any component in a system.

## State Machine Diagram

State machine diagrams are similar to activity diagrams, although notations and usage change a bit. They are sometimes known as state diagrams or state chart diagrams as well.

## Sequence Diagram

Sequence diagrams in UML show how objects interact with each other and the order those interactions occur. It's important to note that they show the interactions for a particular scenario.

## Communication Diagram

In UML 1 they were called collaboration diagrams. Communication diagrams are similar to sequence diagrams, but the focus is on messages passed between objects.

## Interaction Overview Diagram

Interaction overview diagrams are very similar to activity diagrams. While activity diagrams show a sequence of processes, Interaction overview diagrams show a sequence of interaction diagrams.

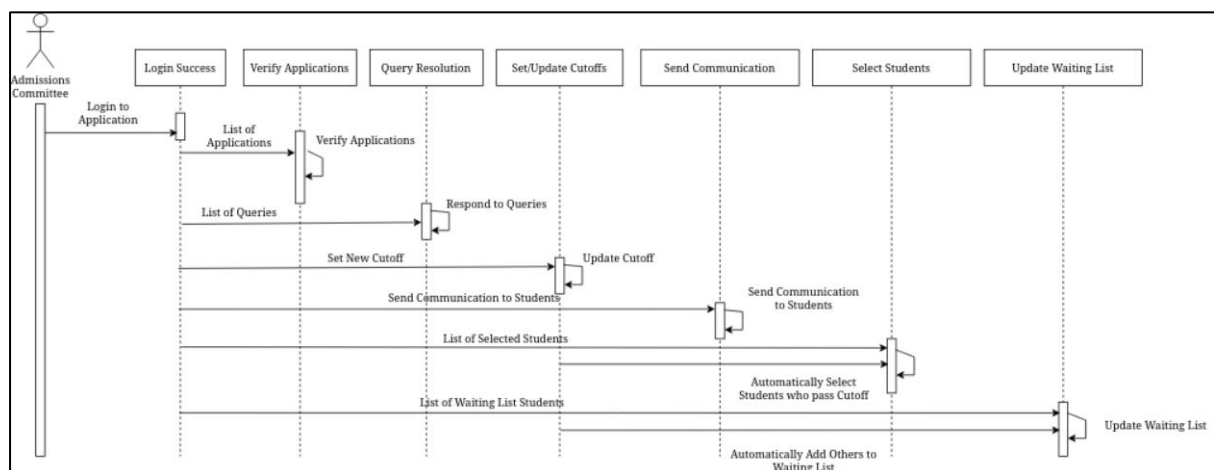
## Timing Diagram

Timing diagrams are very similar to sequence diagrams. They represent the behaviour of objects in a given time frame. If it's only one object, the diagram is straightforward.

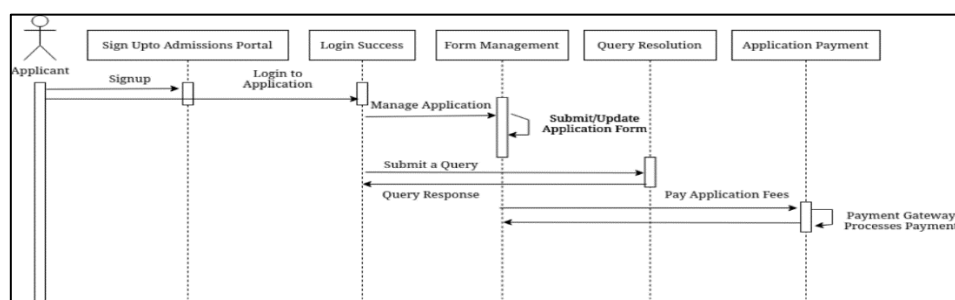
## Experiment

After having some understanding over the UML we decided to apply our knowledge with a demo project 'Student Admission System'.

To understand the use-cases for this project we designed Sequence diagrams and Use Case diagram as shown below.



Sequence Diagram for Admission Committee



Sequence Diagram for Applicants



Use Case Diagram

## Result

After going through this exercise, we realised which UML component is useful for which aspect of the SDLC process.

Further more as per our original motivation we wanted to see if it's possible to Forward Engineer our UML diagrams to code and used tools have been discussed further.

## Micro Services and traditional SDLC

We wanted to divide a full stack application into its smaller micro services where each of the microservices follows the same traditional SDLC phases. This was done to reduce the total time taken as of each these development process will run parallel to each other.

**Project: - Course Enrolment System**

<https://github.com/bolleyboll/course-enroll-microservices>

The Project is a simple course enrolment system with the following microservices.

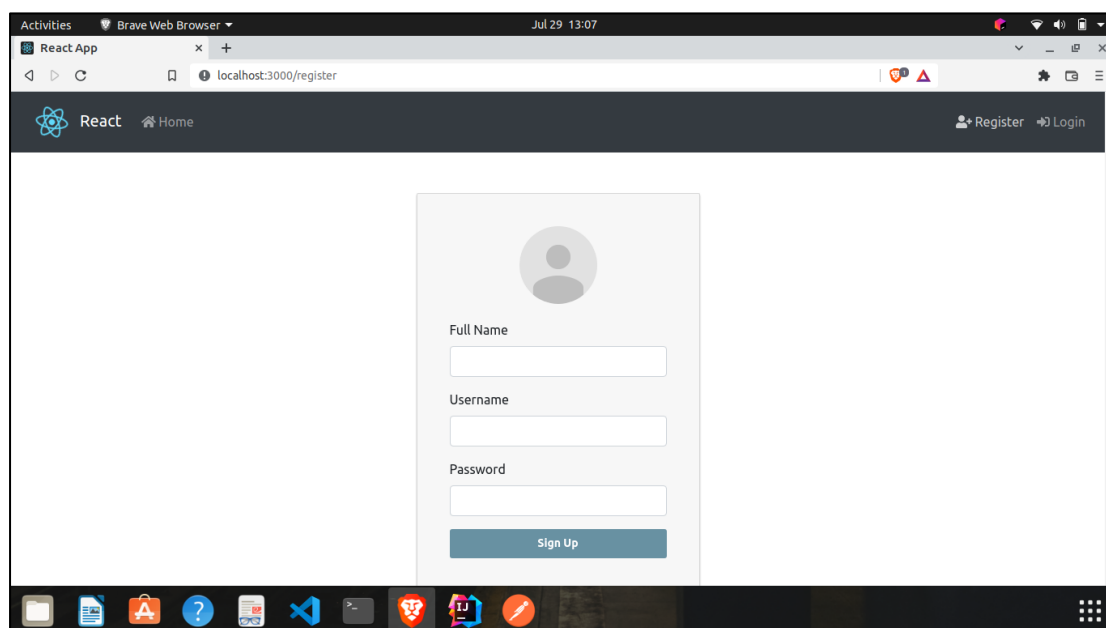
- **microservice-user-management** - Microservice implemented using Spring boot.
- **microservice-course-management** - Microservice implemented using Spring boot.
- **eureka-discovery-service** - Microservice implemented using Spring Eureka.
- **zuul-gateway-service** - Microservice implemented using Spring Zuul.
- **client-side** - A NodeJs application implemented using React.

**Technologies used:-** Spring Cloud, Spring Boot, React, MySQL, Hibernate .

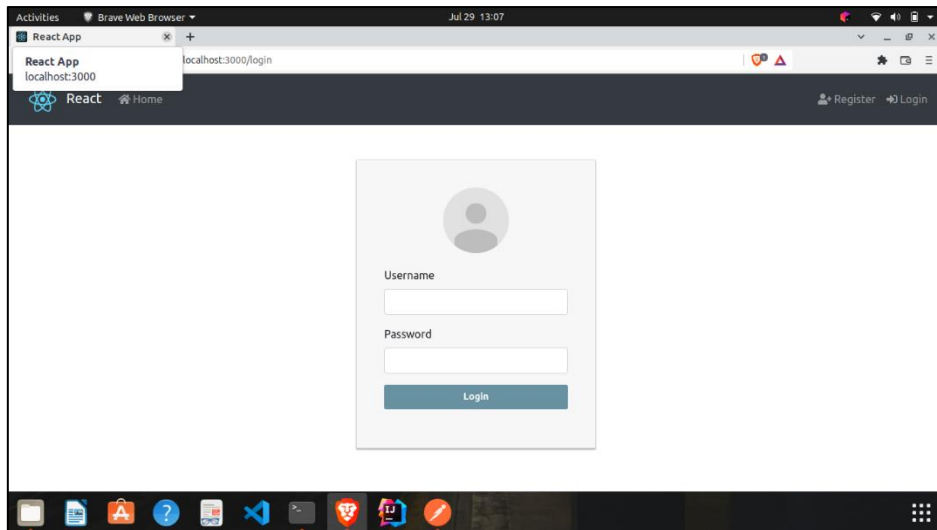
**Version 0.1:-** Supported student registration and enrolment in the subjects.

**Version 0.2:-** Added unenrolment facility in the application.

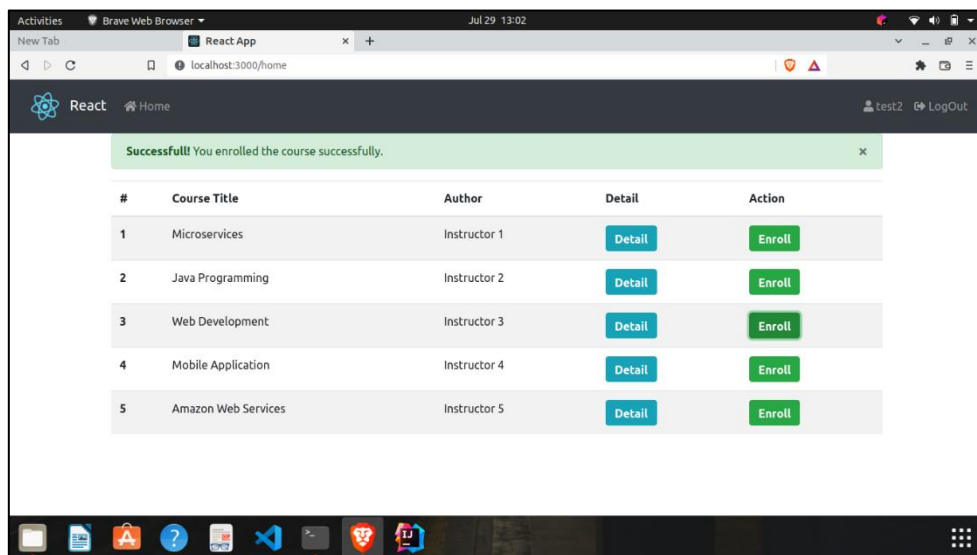
**Version 0.3:-** Added functionality to upload grades of a student in as a CSV file.



Login Page



Registration page



Dashboard

	A	B	C	D	E
	Use Case	Artifact	Artifact Path	Artifact Type	Remarks
1	Register	App.js	client-side/src/App.js	frontend-page	React Start Script
2	Register	register.page.jsx	client-side/src/pages/register/register.page.jsx	frontend-page	Registration Page
3	Register	register.page.css	client-side/src/pages/register/register.page.css	frontend-stylesheet	Registration Page Stylesheet
4	Register	user.js	client-side/src/models/user.js	frontend-model	User Model
5	Register	user.service.js	client-side/src/services/user.service.js	frontend-service	Frontend User Service(API Endpoints)
6	Register	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/shah/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
7	Register	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
8	Register	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/shah/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
9	Register	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
10	Register	UserController.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/controller/UserController.java	backend-controller	Backend API Endpoint
11	Register	User.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/model/User.java	backend-model	Backend DB Model
12	Register	UserService.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/service/UserService.java	backend-service	Backend Services
13	Register	UserServiceImpl.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/service/UserServiceImpl.java	backend-service-impl	Backend Services Implementation
14	Register	UserRepository.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/repository/UserRepository.java	backend-db-repo	JPA DB Repo
15	Register	login.page.jsx	client-side/src/pages/login/login.page.jsx	frontend-page	Login Page
16	Register	login.page.css	client-side/src/pages/login/login.page.css	frontend-stylesheet	Login Page Stylesheet
17	Login	App.js	client-side/src/App.js	frontend-page	React Start Script
18	Login	login.page.jsx	client-side/src/pages/login/login.page.jsx	frontend-page	Login Page
19	Login	login.page.css	client-side/src/pages/login/login.page.css	frontend-stylesheet	Login Page Stylesheet
20	Login	user.js	client-side/src/models/user.js	frontend-model	User Model
21	Login	user.service.js	client-side/src/services/user.service.js	frontend-service	Frontend User Service(API Endpoints)
22	Login	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/shah/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
23	Login	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
24	Login	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/shah/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
25	Login	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
26	Login	UserController.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/controller/UserController.java	backend-controller	Backend API Endpoint
27	Login	User.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/model/User.java	backend-model	Backend DB Model
28	Login	UserService.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/service/UserService.java	backend-service	Backend Services
29	Login	UserServiceImpl.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/service/UserServiceImpl.java	backend-service-impl	Backend Services Implementation
30	Login	UserRepository.java	microservice-user-management/src/main/java/com/shah/microserviceusermanagement/repository/UserRepository.java	backend-db-repo	JPA DB Repo

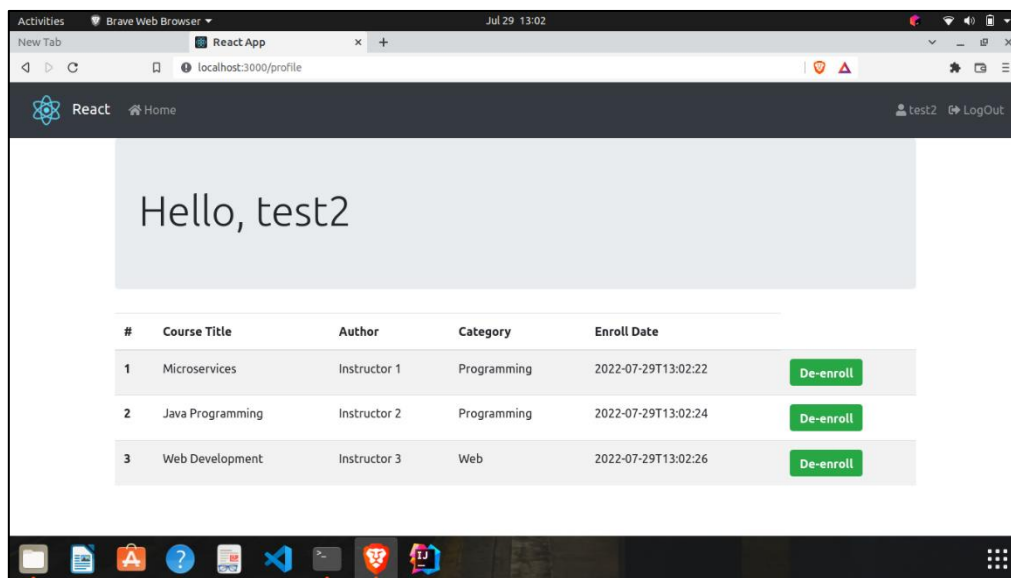
Version 0.1: Artifacts



Use Case	Artifact	Artifact Path	Artifact Type	Remarks
Show Course	App.js	client-side/src/App.js	frontend-page	React Start Script
Show Course	home.page.jsx	client-side/src/pages/home/home.page.jsx	frontend-page	Home Page
Show Course	detail.page.jsx	client-side/src/pages/detail/detail.page.jsx	frontend-page	Detail Page
Show Course	course.service.js	client-side/src/services/course.service.js	frontend-service	Frontend Course Service(API Endpoints)
Show Course	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/sha/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
Show Course	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
Show Course	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/sha/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
Show Course	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
Show Course	CourseController.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/controller/CourseController.java	backend-controller	Backend API Endpoint
Show Course	CourseService.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseService.java	backend-service	Backend Services
Show Course	TransactionRepository.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/repository/TransactionRepository.java	backend-db-repo	JPA DB Repo
Show Course	UserClient.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/intercomm/UserClient.java	backend-controller	Backend API Endpoint
Show Course	UserController.java	microservice-user-management/src/main/java/com/sha/microserviceusermanagement/controller/UserController.java	backend-controller	Backend API Endpoint
Show Course	User.java	microservice-user-management/src/main/java/com/sha/microserviceusermanagement/model/User.java	backend-model	Backend DB Model
Show Course	UserService.java	microservice-user-management/src/main/java/com/sha/microserviceusermanagement/service/UserService.java	backend-service	Backend Services
Show Course	UserServiceImpl.java	microservice-user-management/src/main/java/com/sha/microserviceusermanagement/service/UserServiceImpl.java	backend-service-impl	Backend Services Implementation
Show Course	UserRepository.java	microservice-user-management/src/main/java/com/sha/microserviceusermanagement/repository/UserRepository.java	backend-db-repo	JPA DB Repo
Enroll	App.js	client-side/src/App.js	frontend-page	React Start Script
Enroll	home.page.jsx	client-side/src/pages/home/home.page.jsx	frontend-page	Home Page
Enroll	user.js	client-side/src/models/user.js	frontend-model	User Model
Enroll	transaction.js	client-side/src/models/transaction.js	frontend-model	Course Enrollment Model
Enroll	course.service.js	client-side/src/services/course.service.js	frontend-service	Frontend Course Service(API Endpoints)
Enroll	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/sha/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
Enroll	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
Enroll	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/sha/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
Enroll	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
Enroll	CourseController.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/controller/CourseController.java	backend-controller	Backend API Endpoint
Enroll	Transaction.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/model/Transaction.java	backend-model	Backend DB Model
Enroll	CourseService.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseService.java	backend-service	Backend Services

Version 0.1 Artifacts

## Version 0.2 Changes:

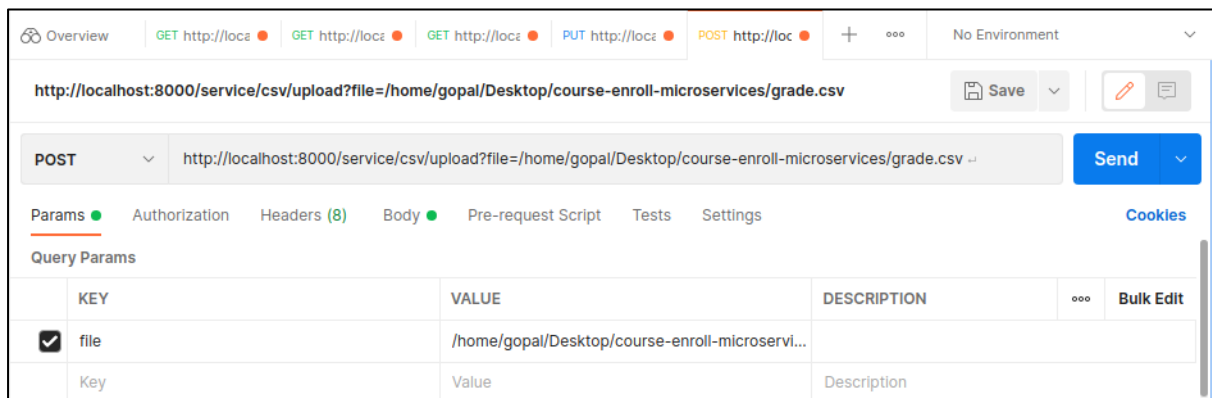


User Profile Page

Use Case	Artifact	Artifact Path	Artifact Type	Remarks
Enroll	CourseRepository.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/repository/CourseRepository.java	backend-db-repo	JPA DB Repo
Enroll	TransactionRepository.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/repository/TransactionRepository.java	backend-db-repo	JPA DB Repo
Logout	App.js	client-side/src/App.js	frontend-page	React Start Script
View Profile	App.js	client-side/src/App.js	frontend-page	React Start Script
View Profile	profile.page.jsx	client-side/src/pages/profile/profile.page.jsx	frontend-page	Profile Page
View Profile	user.service.js	client-side/src/services/user.service.js	frontend-service	Frontend User Service(API Endpoints)
View Profile	course.service.js	client-side/src/services/course.service.js	frontend-service	Frontend Course Service(API Endpoints)
View Profile	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/sha/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
View Profile	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
View Profile	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/sha/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
View Profile	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
View Profile	CourseController.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/controller/CourseController.java	backend-controller	Backend API Endpoint
View Profile	CourseService.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseService.java	backend-service	Backend Services
View Profile	CourseServiceImpl.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseServiceImpl.java	backend-service-impl	Backend Services Implementation
View Profile	TransactionRepository.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/repository/TransactionRepository.java	backend-db-repo	JPA DB Repo
Un-Enroll	App.js	client-side/src/App.js	frontend-page	React Start Script
Un-Enroll	home.page.jsx	client-side/src/pages/home/home.page.jsx	frontend-page	Home Page
Un-Enroll	user.js	client-side/src/models/user.js	frontend-model	User Model
Un-Enroll	transaction.js	client-side/src/models/transaction.js	frontend-model	Course Enrollment Model
Un-Enroll	course.service.js	client-side/src/services/course.service.js	frontend-service	Frontend Course Service(API Endpoints)
Un-Enroll	ZuulGatewayServiceApplication.java	zuul-gateway-service/src/main/java/com/sha/zuulgatewayservice/ZuulGatewayServiceApplication.java	backend-gateway	Gateway
Un-Enroll	application.properties	zuul-gateway-service/src/main/resources/application.properties	backend-gateway-props	Gateway Props
Un-Enroll	EurekaDiscoveryServiceApplication.java	eureka-discovery-service/src/main/java/com/sha/eurekadiscoveryservice/EurekaDiscoveryServiceApplication.java	backend-service-discovery	Service Discovery
Un-Enroll	application.properties	eureka-discovery-service/src/main/resources/application.properties	backend-service-discovery-props	Service Discovery Props
Un-Enroll	CourseController.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/controller/CourseController.java	backend-controller	Backend API Endpoint
Un-Enroll	Transaction.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/model/Transaction.java	backend-model	Backend DB Model
Un-Enroll	CourseService.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseService.java	backend-service	Backend Services
Un-Enroll	CourseServiceImpl.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/service/CourseServiceImpl.java	backend-service-impl	Backend Services Implementation
Un-Enroll	CourseRepository.java	microservice-course-management/src/main/java/com/sha/microservicecoursemanagement/repository/CourseRepository.java	backend-db-repo	JPA DB Repo

Version 0.2 Artifacts

## Version 0.3 Changes



Upload Grades as a CSV File

```
mysql> show tables;
+-----+
| Tables_in_micro_user |
+-----+
| DATABASECHANGELOG    |
| DATABASECHANGELOGLOCK |
| grade                |
| user                 |
+-----+
4 rows in set (0.00 sec)

mysql> Select * from grade;
+----+-----+-----+-----+
| id | grade | semester | username |
+----+-----+-----+-----+
| 1  | A     | 3        | test1    |
| 2  | B     | 4        | test2    |
+----+-----+-----+-----+
2 rows in set (0.00 sec)
```

Database Updates upon CSV Upload

## **Reverse Engineering Process**

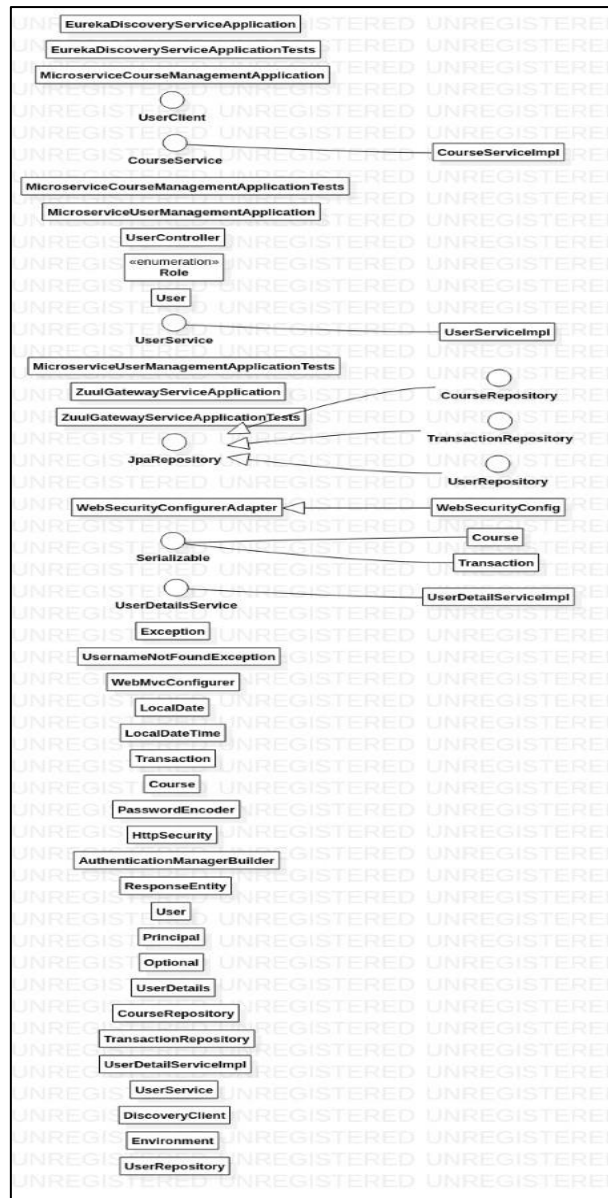
For the reverse engineering we have tool called StarUML. StartUML is a software engineering tool for system modeling using the Unified Modeling Language, as well as Systems Modeling Language, and classical modeling notations. It is published by MKLabs and is available on Windows, Linux and MacOS.

### **Giving code as input to generate UML Diagram : -**

Diagram as can be seen below is not able to make proper relation between classes, and yet not can be used for further analysis.

### **Giving UML diagram generated by StarUML as input to generate code : -**

The code generated is able to maintain the basic structure of the code but not enough to capture the complete code hence cannot be used for further exploration. Diagram regarding this can be seen below.



UML Diagram generated by StarUML

```

File Edit Selection View Go Run Terminal Help
User.java - JavaReverse - Visual Studio Code
1 package com.sha.microserviceusermanagement.model;
2
3 import java.util.*;
4
5 /**
6  *
7  */
8
9 public class User {
10
11     /**
12      * Default constructor
13      */
14     public User() {
15
16     }
17
18     /**
19      *
20      */
21     private Long id;
22
23     /**
24      *
25      */
26     private String name;
27
28     /**
29      *
30      */
31     private String username;
32
33     /**
34      *
35      */
36     private String password;
37
38 }
  
```

Code Structure generated by StarUML

## No-Code / Low-Code approach:-

Low-code/no-code development platforms are types of visual software development environments that allow enterprise developers and citizen developers to drag and drop application components, connect them together and create mobile or web apps.

Low-code/no-code software development approaches support a variety of application types. Small business transactional systems are perhaps the most common. These are applications that process business transactions — tools such as human resource management (e.g., performance appraisal), reservation management for restaurants or other experiences, order quote creation, field service management, and so forth. Some LC/NC tools now make it easier for marketers to automate marketing activities such as website personalization, email marketing and digital ad trafficking.

## No-Code / Low-Code tools:-

- Bubble
- GlideApps
- Appy Pie
- Webflow

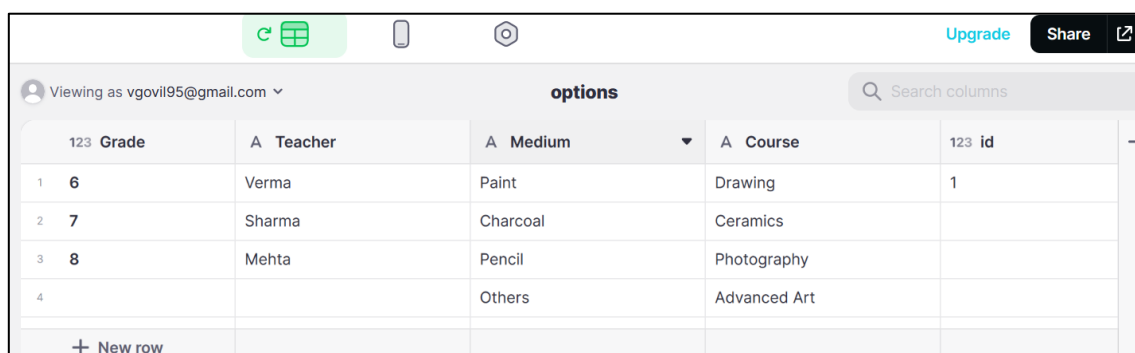
## GlideApps

GlideApps is an online platform that instantly transforms data from spreadsheets (Google Sheet, Excel) into mobile apps that fit on iOS, Android, phones and tablets. Glide doesn't offer a complicated visual editor, just scalable visual blocks to display data and make it interactive with drag and drop. Instant updating. Changes to data immediately update the application.

Customizing the app once complete, it can be added to our home screen for access at any time and shared with anyone as a QR code to scan or via a simple link.

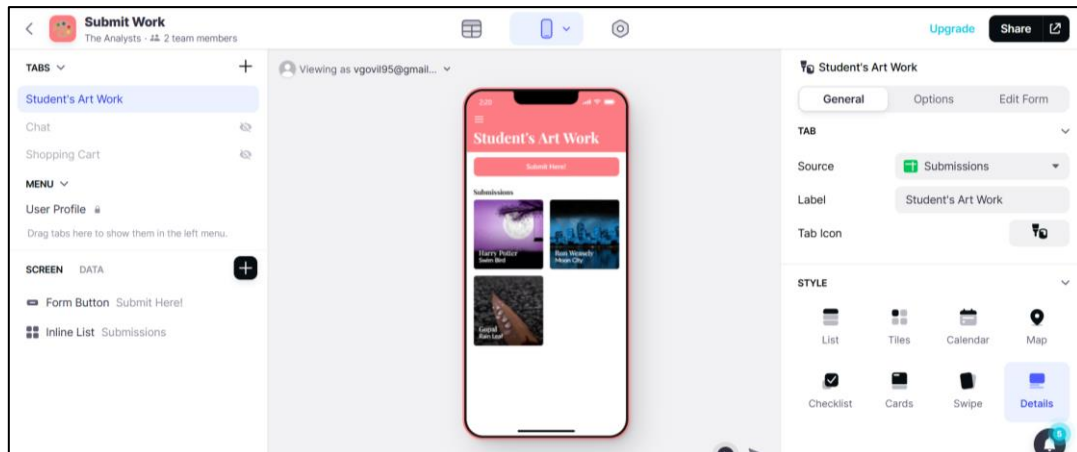
In order to try out Glideapps, we built a sample application which allows a student to their artwork.

Below are the screenshots from the application:

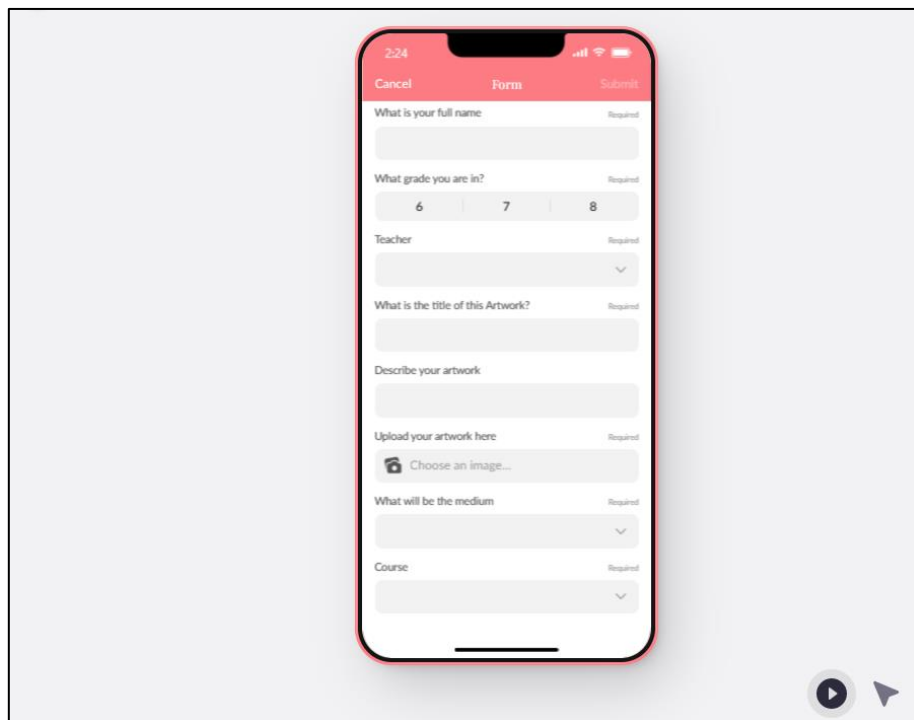


options				
123 Grade	A Teacher	A Medium	A Course	123 id
1 6	Verma	Paint	Drawing	1
2 7	Sharma	Charcoal	Ceramics	
3 8	Mehta	Pencil	Photography	
4		Others	Advanced Art	
+ New row				

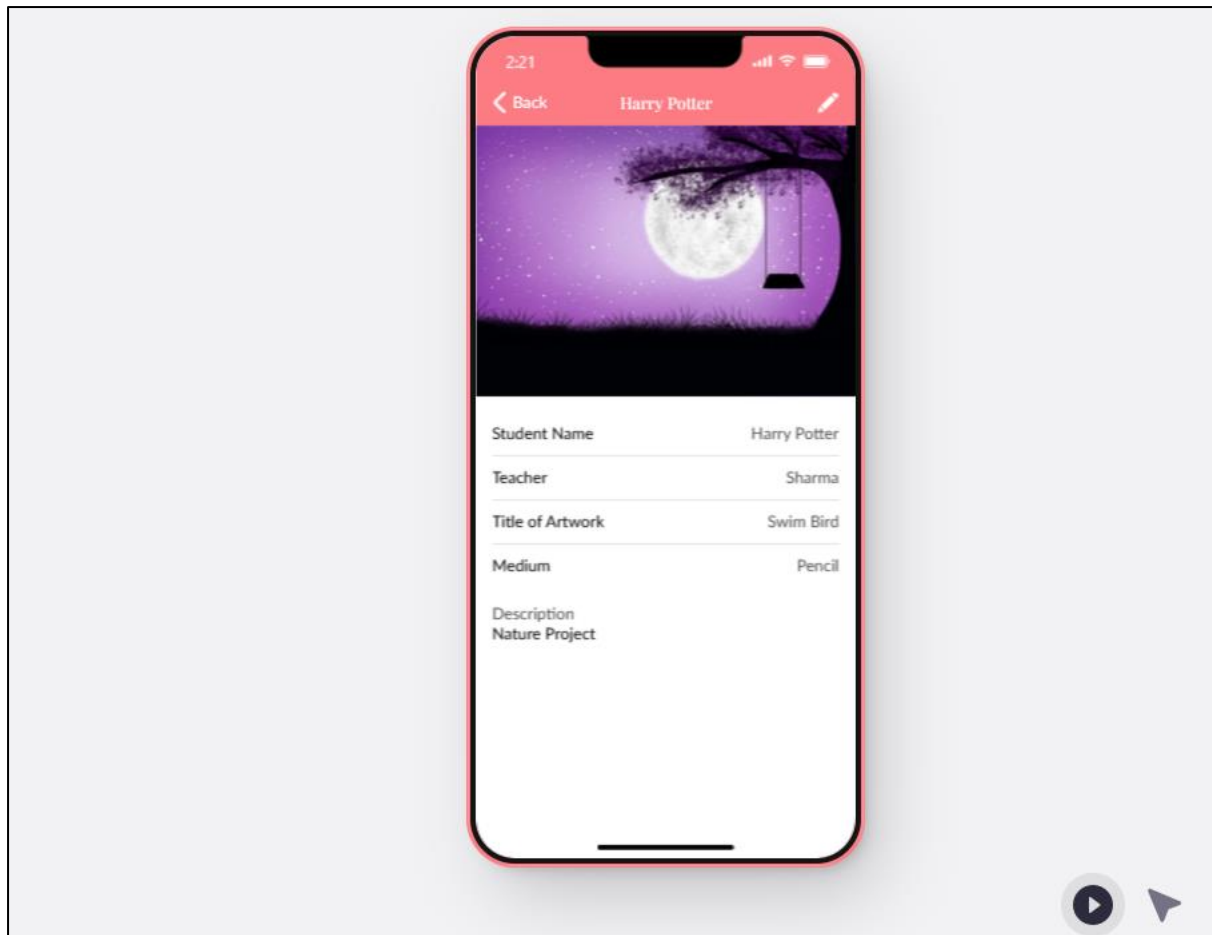
Options Table Database



GlideApps Layout Page



Submit Artwork Page






Submissions Page

Submit Work  
The Analysts - 2 team members

Upgrade Share

Viewing as vgovil95@gmail.com

Search columns

	A Full Name	123 Grade	A Teacher	A Title of Artwork	A Description	Link to Image	
1	Harry Potter	7	Sharma	Swim Bird	Nature Project		
2	Ron Weasley	8	Mehta	Moon City	Moon City Art Work		
3	Gopal	7	Sharma	Rain Leaf	Leaf in the rain		
	+ New row						

GlideApps Data Page: Submissions

### Advantages of GlideApps:

- **Modern themes:** You don't need to be a designer to create a beautiful app.
- **User profiles:** Data is stored in a user's profile to customize the experience.
- **Customized actions:** Ability to chain multiple steps with logical conditions.
- **Spreadsheet synchronization:** Data resides in Google Spreadsheet, while keeping everything collaborative.
- **Tablet mode:** Apps automatically expand when opened on larger screens.
- **Open collaboration:** Ability to add collaborators to work on apps together.

### Disadvantages of GlideApps:

- Limited data capacity, GlideApps for Free only provides 500 rows, and Business plans support upto 50000 rows, which is extremely limited.
- Only supports Spreadsheets, GlideTable and Airtable(Premium) which are different from standard databases, which makes it very difficult to migrate to GlideApps.
- Doesn't support Single-Sign-On.

### **Conclusion:-**

During this period, we had learned and explored about various UML diagrams, modelling application through various UML diagrams, tried reverse engineering on microservice architecture to understand the internal artifacts required to build the sub-modules necessary to generate the functionality for each microservices, along with all the necessary artifacts that are common among all the three versions. We also discovered about the No-code/Low-code methodology to understand the generic functionalities that can be implemented through No-code/Low-code along with its limitations, that determines the dynamic aspects of any application i.e. functionalities for which human interaction is still required.

### **References:-**

<https://www.omg.org/>

<https://www.geeksforgeeks.org/unified-modeling-language-uml-introduction/>

<https://staruml.io/>

<https://www.glideapps.com/>

<https://github.com/bolleyboll/course-enroll-microservices>