

University of Economics and Human Sciences  
in Warsaw

Field of study: Computer Engineering



Zaur Jabrayilov

36507

**Title: Course Management System within the LMS**

Bachelor's thesis written under the supervision of:  
Mikołaj Aleksiejuk

Warsaw, 2022

## Table of Contents

Index of Figures .....	Ошибка! Закладка не определена.
Index of Abbreviations .....	3
Abstract.....	3
I. Introduction.....	6
A. Goals .....	6
B. Objectives.....	7
II. System Analysis.....	8
A. Existing Solutions .....	8
B. Feasibility Analysis .....	9
III. Software Requirements.....	9
C. Requirement Analysis .....	9
D. Functional Requirements .....	10
E. Non-Functional Requirements.....	11
IV. System Design .....	12
A. System Architecture.....	12
B. Diagram.....	14
F. Essential Components of the project .....	Ошибка! Закладка не определена.
V. Implementation .....	20
A. Software Analysis .....	20
B. Used Technologies .....	33
C. Timeline or Gantt chart .....	Ошибка! Закладка не определена.
VI. Testing.....	Ошибка! Закладка не определена.
Conclusion .....	39
Source Code.....	40
References.....	41
Appendices.....	43

## Index of Abbreviations

### Abbreviations

LMS

UI

UX

ML

AI

API

OCR

MVC

JWT

HTML

CSS

SQL

CRUD

UML

WORA

PC

OS

OOP

JSF

### Long Text

Learning Management System

User Interface

User Experience

Machine Learning

Artificial Intelligence

Application Programming Interface

Optical Character Recognition

Model View Controller

JSON Web Token

Hyper Text Markup Language

Cascading Style Sheets

Structured Query Language

Create Read Update Delete

Unified Modeling Language

Write Once Run Everywhere

Personal Computer

Operating System

Object Oriented Programming

JavaServer Faces

JEE	Java Enterprise Edition
AOP	Aspect Oriented Programming
ORM	Object-Relational Mapping
DAO	Data Access Object
JPA	Java Persistence API
JSP	JavaServer Pages
IO	Input Output
JSON	JavaScript Object Notation
SSH	Secure Shell
XML	Extensible Markup Language
POJO	Plain Old Java Object

**Abstract**

Course management systems are so important for this era of our lives when it comes to managing student and their course. The number of students is increasing each day and it makes to management harder. Increasing number of students force the course management to switch from paper work to systems which makes it a bit easier. Those systems are called LMS which means Learning Management System. LMS software should provide simple user interface where everything will be clear for its users. There should be students, teachers where each teacher will be assigned to one course and a lot of students assigned to one course. In this dissertation work, I try to describe and introduce my project idea called “Ingress LMS” developed by me. LMS software is basically a software that provides a lot of functionalities such as tracking students, setting grades to the students, assigning courses and teachers a lot of such functionalities. We are aiming to explain all the differences from traditional choices such as using papers for student management which is old school way.

## I. Introduction

### A. Goals

One of the main purposes of this software is that to provide reliable LMS platform for its users such as students, teachers. Reliable website means that users of the software can easily interact and benefit out if it. The platform needs to comfortable to use on any device such as laptop, mobile device in order to make it comfortable for its users. For a small update it would be easier to just launch the software on mobile device rather than laptop. But our main goal is to replace the traditional way of management which is based on paperwork. We are trying to replace it with the coded way where it can be easier to manage. Hence, it leads to better management of students. Such as we can warn students throughout the platform where we don't have to talk to them face to face. There are considerable a lot of advantages and disadvantages of LMS software that I would love to list them down so we have clear understanding on the project. Here we will be identifying the points which are the best for the future of students and such management system:

- **Eco-friendly:** We can basically avoid using a lot of papers and just throwing away at the end of the work.
- **Accurate data:** If we compare the traditional ways of tracking learning management systems, we can see the huge difference. Because paperwork can lead to mistakes but well-coded software won't have any mistakes unless there is no attack to the software. LMS softwares solve these issues and provide more accurate data for each party.
- **Backed up data:** It's quite standard that each software which is serving for users need to have database where the software can store some information. If we have databases we can easily clone and backup those databases which will prevent future data loss. But if look at the situation from paperwork perspective, it would be real loss if we would lose one of the main papers where it's written payments of the students which is crucial when it comes to course management. Papers only can be protected by physical measures such as locked shelves but the databases are easy to back up and save them. But databases are not always safe because databases can face attacks as well such as SQL injection [1] but for those reasons there is such approach of hashing sensitive data of users such as password, address which can be hashed which means that the given data from user let's say password will be changed by an algorithm and a keyword into something random numbers and letters and only if you know the keyword and algorithm you can decrypt it. Otherwise, it's kind of impossible to decrypt such data. That's why such approach is one of the safest.
- **Easy management:** LMS softwares provide an easy management for teachers where they don't have to deal with huge paperwork but rather just clicking one or two buttons in order to set attendance for students. In such way students cannot

change those data manually. And also, based on those data we can extract a monthly analysis of students giving information about each student who is not joined lectures the most. Such way we can track students. Also, from the student's perspective it's also easier to use LMS softwares because such way they can access the resources from the software rather than browsing in the internet to find a particular resource. Also, it's one of the best approaches of LMS softwares that we can also upload the recorded videos of the lectures. Where students can easily download and start watching if they haven't joined a lecture.

- **Data analysis:** One of the most important advantages and features of such softwares is that we can easily retrieve particular data and analyze it. For example, we want to know how many students haven't joined a lecture in last month. It would be really hard process when it comes to paperwork. Let's imagine not last month but 6 months ago which would be possibly impossible to retrieve these data because that paper which contains data can be lost or thrown away. But LMS softwares allow to retrieve any data from any period of time. And it can be even done within minutes which is the best part of such approach. And it can be even done at night time if you have enough privileges in the system because with the paperwork you might need to wait for workers and get lost in the papers. As we have already mentioned that we are just few clicks away from those data in the softwares.

The basic advantages of the software are listed above of the section. As well as we can see the main differences between traditional approaches and my software here, we can gain better understanding of the project.

## **B. Objectives**

Our main objective to lower the use of paperwork and make an efficient platform where you can manage students easily and in a timely manner. And we need to keep it as simple as possible. Meaning of keeping simple is that we want to make the software look simply so we can do the processes within a few clicks without going through a lot of steps. For an instance, attendance is one of the most important things when it comes to student management and it's one of the things that is probably used every lecture by teachers. So, saying that, this process needs to be as simple as possible in order to make it in a really less amount of time. This can save a lot of time of each party both students and teachers so it's one of the most crucial and such way both parties can be more focused on the lecture rather than these legworks.

Our one of the main objectives is security for both parties. On LMS softwares there can be a lot of courses ongoing and a lot of teachers. So, we need to protect each course from both students and teachers. Which means we need to prevent students seeing every course unless they are assigned to otherwise it would be nonsense letting students to see every course without

enrolling and paying the tuition fee. As well as, we need to prevent teachers seeing all the courses unless they have permissions.

The main purpose of our software is to provide to the users of the software user-friendly and high usability software. User-friendly and high usability terms can mean a lot things under the hood. For example, fast load times which is one of the most important things when deploying a software. For sure nothing is more annoying than a software taking too long to load a page. Of course, in this point if you are using cheap hosting or servers, there is nothing much to do with the architecture or code. On the other hand, one of the main aspects of designing user-friendly website is clear structure and effective navigation. Moreover, keeping clean the home page is best way to achieve this approach. More important giving the full control to user is crucial which means adding informative and useful links in the navbar such as Home, Categories and so on so forth. We should avoid adding too many options to the navigation bar as well because simplicity is also important. Users should get to know the website by 3-4 clicks it shouldn't exceed.

The design and everything related to that part will be discussed in the future sections of the paper briefly.

## II. System Analysis

### A. Existing Solutions

Existing solutions analysis is basically we analyze the software that are already exists and serving for people. We can find a lot of alternatives in the market. The logic of those software is quite similar with my idea and I will list them below and talk about them in details. This section will briefly cover all the alternative software in the market, they are as follows:

1. **Google classroom** – [2] One of the most known software by courses and teachers as it comes from Google and provides a lot of features. It's one of the best when it comes to having courses for only small to medium topics but when it comes to maintain the huge course company that would be a lot to maintain for such courses it's better to develop their own platform where they can use. The software can be used on a lot of devices such as mobile, laptop and etc. Which is a big advantage for the software. And one of the quite important things about google classroom is that it's free to use and gives a lot of functionalities.

#### *Best features:*

- The maintainability of assignments and accuracy of the setting dates so it's also a bit elastic that we may need to assign an extra time so it's a lot easier to do it here.
- The usability of the grading systems for its students where it can be by category grading or no overall grading based on the need.



- The best feature and the feature that must be implemented of the LMS software is live classes that we can have live classes to teach students and being able to record those meetings afterwards we can share them on google classroom.
- Language support.
- Nice-looking user interface.

***Devices Supported:***

- Web-based
- Mobile based

The extensive research gained me a lot of understating about the existing LMS softwares which are provide a lot of functionalities just like my software. I managed to compare my software to the listed ones throughout the research and there are a lot of common points and things which are different. That makes each software unique for its functionalities. And each platform can differ by its interface and features.

**B. Feasibility Analysis**

The purpose of feasibility analysis is that it checks the whether the existing idea will be a good fit for the market or not which means it worth to implement or not. Basically, means that here we are analyzing the merits and viability of the proposed project. As we have already done the market analysis, we can see that there is already website which makes really good revenue. This analysis contains various factors. The main purpose of the feasibility analysis is to test and examine the proposed project including financial, environmental, auditory statistics. Let's get started with some of factors and discuss them in details:

**Economic Environment.** Based on the current economic in the current country is it advisable to launch such web application or not. This factor aims to check whether the statistics of the existing e-learning softwares if they are functional right now.

**Financial.** As the project will be a web application it requires a hosting cost, domain cost which are the most important factors to be a website. A hosting is as the name refers something that hosts the website in the server and if we want to allocate some memory in the server, we need pay for it. A domain is the unique name and identifier for the website. [3]

### **III. Software Requirements**

**C. Requirement Analysis**

This is one of the most important parts when it comes to writing about the software and documenting it [4]. The main purpose of this documentation is to show the functionalities and non-functionalities to the end-user so we can prevent future misunderstandings. Requirement analysis is also defined as requirement engineering and we will talk about the details about this analysis in a minute. The definition of this analysis is to identify what are the expectation and

expected behavior of the software. The documentation needs to be done considering those five factors which are

- The file needs to be well documented
- The file needs to be measurable
- The file afterwards of the development needs to be testable as well as traceable
- The file needs to be actionable

This analysis has classified by functional a non-functional requirement which we are going to talk about it in the below sections.

#### **D. Functional Requirements**

The first one and the quite important one is functional requirement. This is all about what software should do and how it should look based on those characteristics we document it. Here we are mostly trying to explain what clients want and what we are going to develop after clear understanding. We should be careful that all the logics have listed in this file should met with the final products because these are the main requirements of the product. As well as how the interface should look like such as fonts and colors of the design which are quite important. Basically, we list everything here related to the after development which is final product and its behaviors.

Basically, Functional Requirement is the backbone of the system where we define all the expectations. It's quite important to consider functional requirement here is why and the advantages:

- Bugs found in this stage of development is cheap to fix
- Avoiding meetings which can consume money and time. Because if the purpose and everything is clear then there is no need for regular meetings except monthly or weekly reports.
- Avoids misunderstandings between both parties

Functional requirements need to be clear and understandable for each side. Following section discuss functional requirements for the website separately:

- **Design**
  - Body fonts in the application shall be Comfortaa.
  - Descriptive texts in the application shall be in Times New Roman.
  - The website should display the logo on every page
  - Primary color of the website shall be in hex code #FFFFFF

- **Core**

- The website must allow users to verify their accounts using their email
- The website must allow users to sign in using Github
- The website must allow users to sign in using Twitter
- The website must allow to create courses and manage them
- The website must allow to assign students into courses
- The website must allow to set name, title and description for the courses as well as the picture for courses
- The website must allow users to change personal information such as phone number.
- The website must allow users to change their password by using their email.
- The website must allow users to reset their password in case of forgetting.
- The website must allow the user to create an account by using their email and password.

- **Database**

- The website shall be able to store user information.
- The website shall be able to store user's cake list.
- The website should encrypt the users' password.
- Tables should be connected to each other with foreign key.
- Database shall be protected for SQL injection attacks.

## **E. Non-Functional Requirements**

We might consider non-functional requirements as the parts that are not listed in the functional requirements part but other than we also list here kind of important parts of the documentation. The definition of the non-function requirement part is that we try to describe how software should work like considering system downtime, security and scalability which are also quite important parts. We may encounter the speed of certain functionalities and system downtimes. These are just some basic examples of what Non-functional requirements consider. Let's see what are the non-functional requirements of the software that I have built:

- **Privacy**

- The website shall prevent users from viewing other users' personal information.

- Admin is not allowed to change users' personal information.
- The website shall protect the users' privacy and data such as name, address and etc.
- Users shall not be able to change other users' data such as name.
- Users shall not be able to see other users' transactions details such as address.
- **Availability**
  - The website shall be available every day of the year.
  - The website might have downtime only during low-intensity hours such as at night.
  - If there is a fix going on in the website then website should let users know by giving expected time of getting things back.
- **Speed**
  - Verification emails shall be sent with a latency of no greater than 1 hour.
  - Each request shall be processed within 10 seconds
  - Adding new cake to the card shall take not more than 10 seconds.

## IV. System Design

### A. System Architecture

There are two mainly known web development architectures which are Microservices and MVC. Each of them is offering different maintain abilities. I will talk about each of them in details. The architecture of the project considered to be on MVC architecture which is quite popular since 2009 [5]. The MVC architecture introduced in 2009 by Microsoft. Since that time there are a lot of updates in order to bring new features. Using this architecture, we can easily achieve the goals throughout the development process. MVC makes it easier to develop a web project and gives us possibility to easily maintain it. MVC architecture will make it easier to satisfy the requirements alongside.

Microservices architecture first used by Peter Rodger in 2005 [6] but very first introduction was on 2011. And since that time the first users started to be Netflix and Amazon [7]. There are considerable a lot differences between MVC and Microservices architecture which makes each of them unique in the industry. Using Microservices architecture we are dividing the whole system into pieces which are called services. Each service has its own responsibilities. Let's take a look at an example. For example, most of the flight reservation systems use Microservices architecture because there are a lot of services that system provides. The services can be car rental, hotel booking, tours and so on so forth. It would be hard to maintain all of these things under the MVC architecture. Imagine you might need to change something or disable one service in one of the services in such case you might need to dive into MVC architecture code base and look for the exact logic. Or even we might need to delete car-rental service which is a possible scenario in such cases might lead to big problems. But in

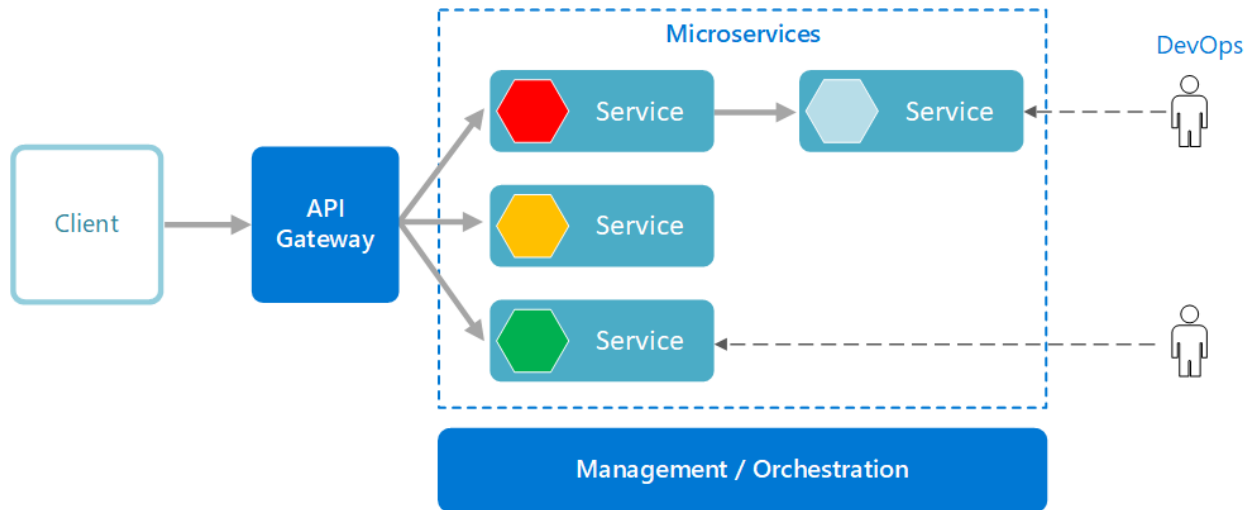
microservices architecture they are already separated and easy to maintain. There are also some major disadvantages of this architecture. For small web applications like mine, such architecture can be really slow and hard to implement. Further than that, testing gets harder with such architecture and gives us plenty of debugging problems and as well as we might face with the deployment issues.

MVC architecture helps along the way for a lot things such as performance, maintainability, security and other crucial requirements for web development. MVC architecture holds three main components and each of them is implemented to handle specific development logic. This architecture is the best fit for small systems which gives us a lot of advantages. One of the main advantages of MVC architecture is that way faster development process and ability to collaborate with other developers. Faster development means that there is no complex implementation such as dividing system into services. And the reason that I am mentioning ability to collaborate with other developers is that one programmer can work on Controller during the other programmer could work on the View component. Such approach also makes the development faster and easier to maintain.

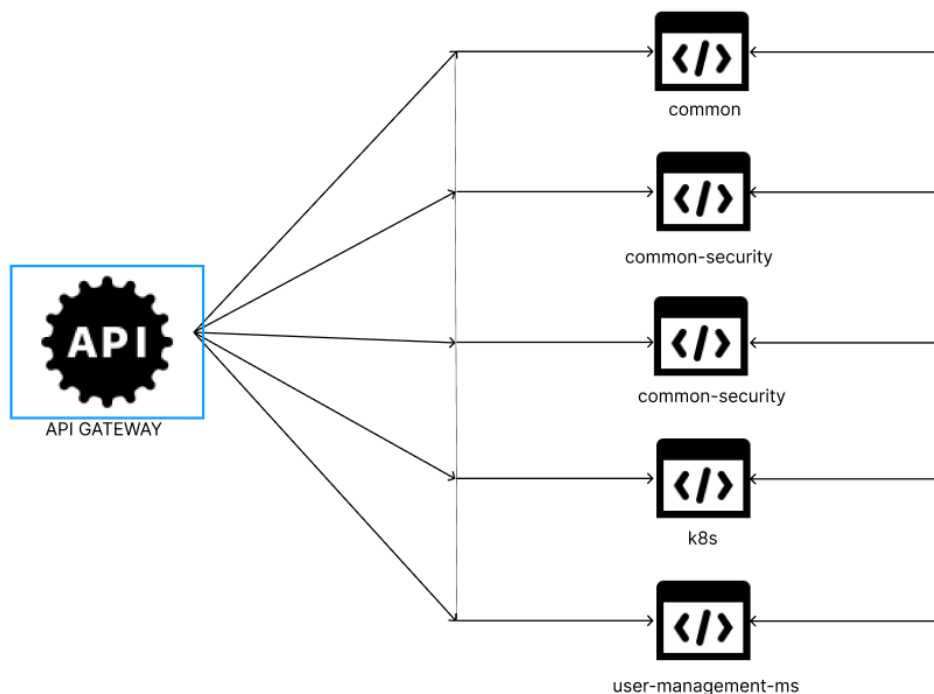
Performance is the key factor for the website. Therefore, we managed to divide the most complex logics into pieces. Additionally, we managed to add caching data-layer which helps to prevent sending requests to the database because there are such data that never changes or changes every year such as the name of countries or name of cakes. Caching data-layer keeps the original data till we destroy it and acts like database in Controller layer which helps to improve the performance.

Maintainability is also one of the crucial requirements of the website. Maintainability term means that we are able to maintain the project easily so we can add new features or update it. As we have already considered to follow SOLID [8] principles this gives us possibility to maintain the project easily. One of the first principle of SOLID is that Single Responsibility which means each class, method should only and only for one responsibility. We have avoided to share data all over every pieces. But only to those which might really need so we can also prevent security leak. Furthermore, small pieces make security a bit harder but for that reason we have used layered structure to the most critical pieces protected in the innermost layers. And all of the models are secured by validation which is high-level of security validation.

Our architecture is going to be Microservices [9] architecture which can give better understanding by taking a look at below figure. Where we can see a client which is a user sending requests by surfing the website or calling APIs. The requests will be handled by API Gateway and sent throughout the services. Here we may have more than 4-5 services.



Our Microservices architecture looks like the same on the below attached figure.



## B. Diagram

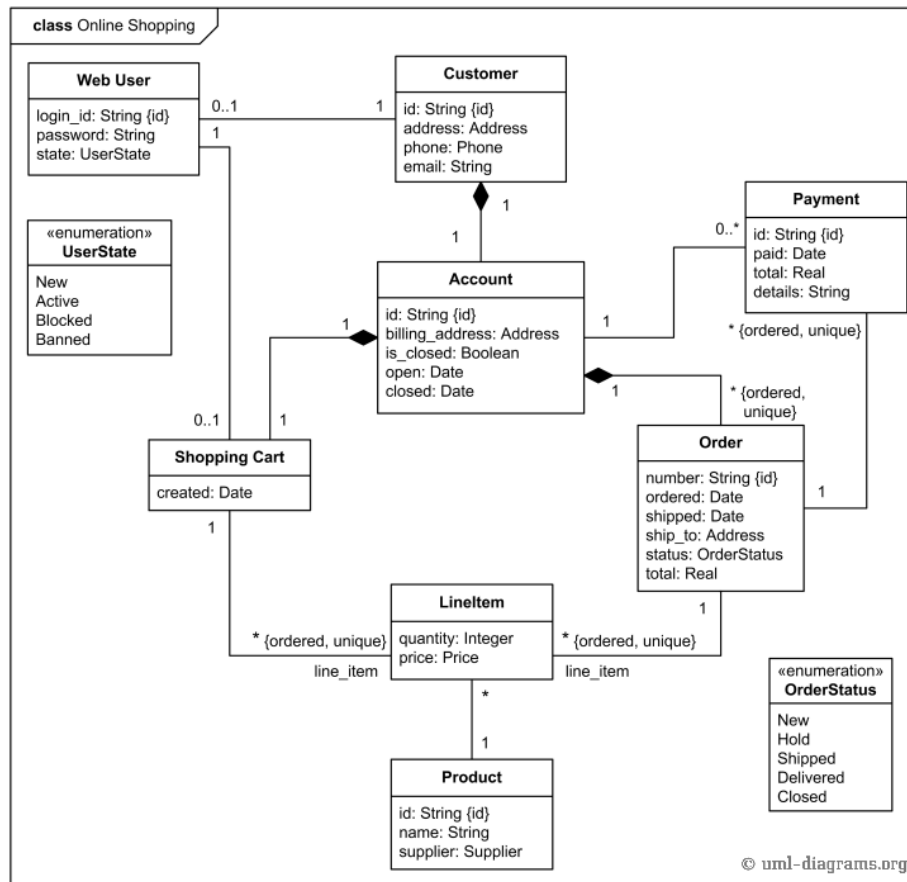
**UML.** UML means Unified Modeling Language. The definition of this term is that it's one of the most important modeling languages known by all software engineers. It's most of the time used for modeling the software and visualizing. The main advantages of UML are that it is understandable by even non-programmer which can be a stakeholder and it would be much easier that it understandable by them as well. UML gives common understanding and makes it easier to show the final product visualization which can be helpful to understand the main

features of the software such as login, registration and other features. In simple words UML is a visualized language that can be understood by everyone

The main purpose of the UML diagrams is that it tries to explain the logic by pictures. It's well known that 'A picture worth a thousand words. By the help of pictures and diagrams we have better understanding of the whole logic. The creation of the UML diagrams also the reason of software documentation which means the documentations are not enough clear for stakeholders. Because of those concerns in 1994 UML [10] diagrams introduced. In the below section of the paper, you can refer to the Figure 9 in order to understand how UML diagrams look like. This is the example of online shopping management diagram which is quite understandable by non-programmer person. Here we have entities and each entity has its own attributes. And each entity is connected to each other with some type of relation. There are 4 types of relations [11] between entities and they are like:

- ManyToOne
- ManyToMany
- OneToMany
- OneToOne

Each of them has their own usage based on the entities in order to keep connectivity between them and such way we can access the data linked and easily.



**Figure 3 – Example UML diagram of Online shopping system**

UML is classified by two diagrams which are behavioral and structural. The below section of the paper we will dive into more details of these diagrams. Each diagram consists 7 diagrams inside of it and let's take a closer look:

**Behavioral diagram** – As the name refers, it's used to describe what is the behavior of the software. It interacts with the entities. This is more about the data and how it moves around the system and entities. As we can see in the Table 1, it consists 7 diagrams and each of them plays different roles for diagrams. Those diagrams include approaches such as case, state, activity and complexity of timing in the software. All of them define the interaction within the system.

**Structural diagram** – Also some people name it as static view. This type of diagrams is emphasizing the static parts of the software such as entities, fields, methods and mapping. Structure diagrams are more about the static structure of the elements in the software which means why and how one object relates to another. The main purpose of this diagram is to identify and show clearly particular logics. For an instance, let's look at an example in order to understand better, let's think about a just static aspects of a home, it encompasses the existence



of such elements; door, lights, stairs, garden and etc. Since structure diagrams represent the structure, they are used extensively in documenting the software architecture of software systems. If we look at the Table 1, we can see that Structural diagrams consist of 7 diagrams in itself.

Behavioral UML Diagram	Structural UML Diagram
Activity diagram	Class diagram
Use case diagram	Object diagram
Interaction overview diagram	Component Diagram
Timing Diagram	Composite structure diagram
State machine diagram	Deployment diagram
Communication diagram	Package diagram
Sequence diagram	Profile diagram

Table 1 – Behavioral and Structural diagrams' subdiagrams

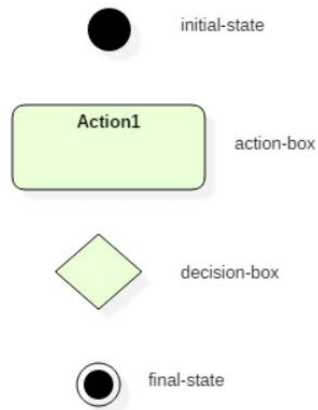
Based on the requirements on my project implementation I went through with Behavioral diagrams such as activity and use case diagrams. In order to make good documentation they are one of the essential. In the below section of the paper, I will dive into more details of each of them.

**Activity diagram** – As we have seen in the table that Activity diagrams are coming along with the Behavioral diagrams [12]. The main purpose of activity diagram is that to provide the better understanding one subsystem or system by visualizing. Apart from visualizing it's quite understandable by non-programmer people. It contains some components which each of them has their own meaning. Each of them serves for some other functionality we can't use those components randomly. Let's see and understand what are those components, let's take a look at the Figure 4 and discuss its components:

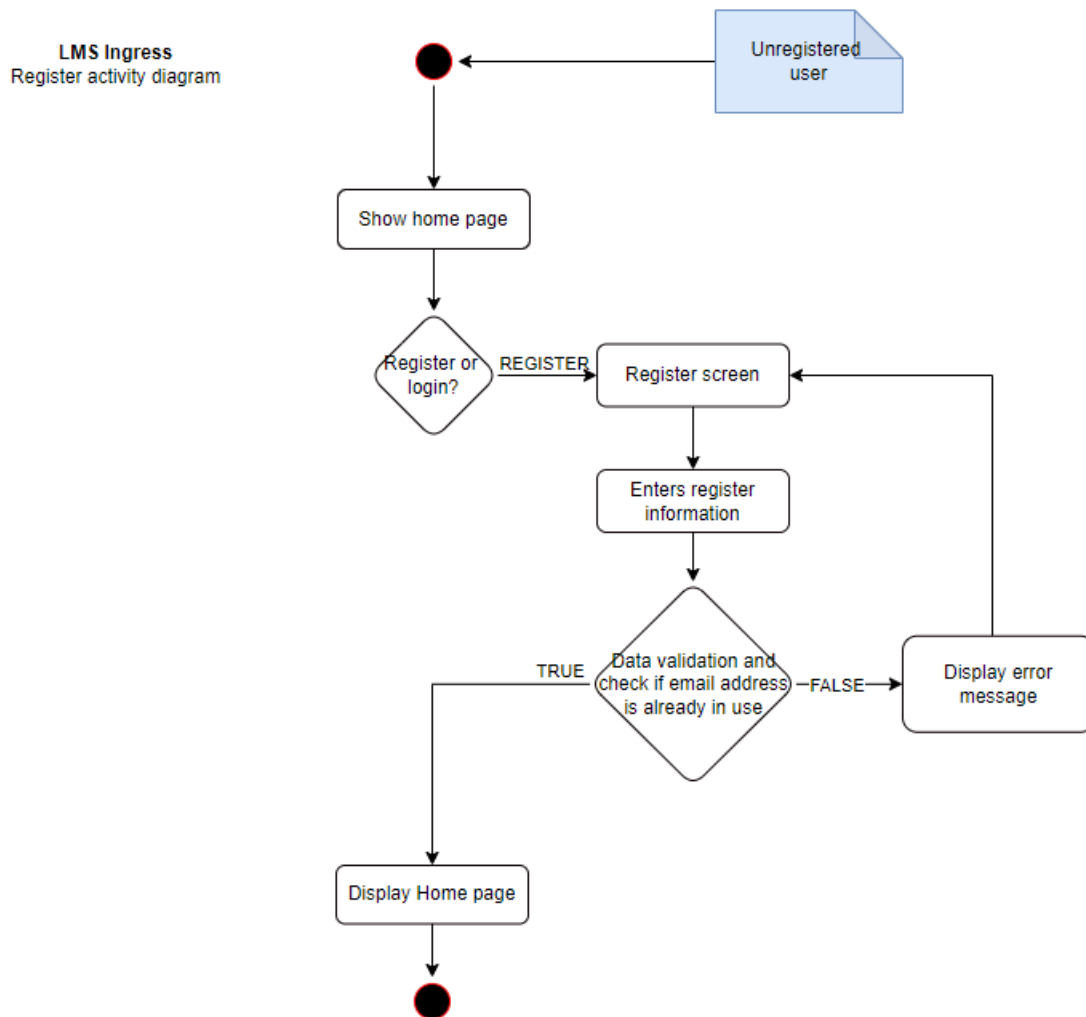
- **Circle (initial-state).** As we can easily see on the FIGURE that it's filled small circle which represents the starting point of the process. It's also called initial-state of the process where all the process starts and this needs to be connected at least an action.
- **Bordered square.** The figure shows all the component of the activity diagram and we can easily see the bordered square which is action-box. Action-box is basically an action that software should handle such as go to main page or do some process. Such movements go into action-boxes and using that we are moving to decision-boxes based on decision boxes we proceed with the action-boxes.
- **Rhombus.** Rhombus is one of the most crucial one and we can't use it everywhere, it's like an if condition meaning that let's say user provides his/her credentials in order to login account this activity requires a decision prior in order to move to text activity. That's why it's also called decision-box. There are two scenarios one is let's say credentials are correct then it means TRUE and we are good to proceed with the next activity. But the second scenario is let's say the credentials are incorrect which is FALSE

state then in such cases we are returning it back by pointing ac activity box before written the rhombus.

- **Circle(final-state).** The final one is a circle small filled as the initial state but this is for representing final-state of the project.



**Registration process in activity diagram.**



**Login process in activity diagram.**

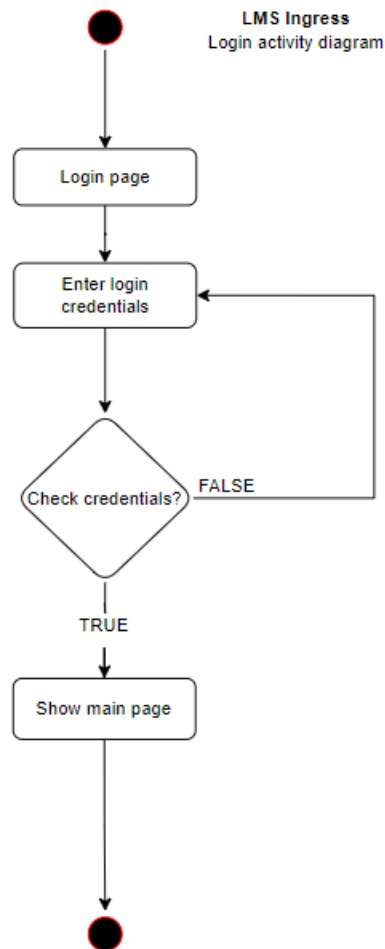


Figure 6 - Activity diagram of login process

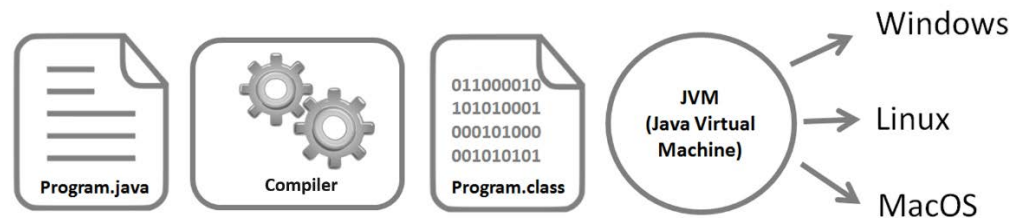
## V. Implementation

### A. Software Analysis

**-Core.** The core side of the project has developed using Java programming language which is one of the most popular languages introduced by James Gosling. In order to develop web applications in java such as mine we have used Spring framework which is also one of the most powerful and popular frameworks of all time. It's also called EE version of Java which means Enterprise Edition here we can develop web applications, cross-platforms and much more web services. There are plenty advantages of using both Java and this framework called Spring. But we may also encounter the disadvantages of Java as well. Let's take a closer look at them and get to know better about them:

**Advantages:**

- **WORA** – Java is so popular by this approach which means write once run anywhere you want. As we can see in the below figure that java class is converted to bytecode by compiler and it's compatible to run it any OS independently such as Linux MacOS.



- **OOP** – Java is quite well known by its OOP approach as well. OOP means Object oriented programming and java has a lot of concepts on it such as Encapsulation, Abstraction, Polymorphism and much more as we can see on the figure. Each of those concepts play a huge role to make the development maintainable. The encapsulation says that we need to encapsulate the whole fields so we are setting all the variables private in the class. Inheritance is like obtaining something from super class and using it in the subclass. Polymorphism lets us to use one object in different ways. There are a lot of examples from the world based on the OOP concepts.



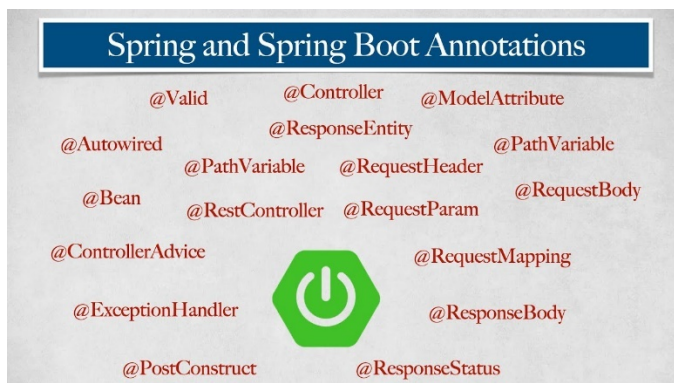
- **Quite enough tutorials**– One of the best advantages of java is that it's so widely used by programmers and it means there are a lot of users and there are a lot of free tutorials based on Java which makes it easier to learn. When we compare it to other languages such as Python, Kotlin there might not be enough tutorials in the internet because those languages are kind of new in the industry.

**Disadvantages:**

- **Desktop application problem** – Building web application and web-services can be really easy and straightforward. Even it can be easier and fun when we compare to other programming languages. But building desktop applications in java can be a bit frustrating. Developing desktop applications can be done using Swing, JavaFX or JSF but they are not suitable for creating complicated UI. There are many problems faced by developers. Statistically, most of the developers don't prefer building desktop application using Java.
- **Complex codes** – It's no secret that writing 5 lines of code in Python is easier than writing 20 lines of code in Java. When we compare complexity of code snippets in java, we can see the difference clearly. Java is kind of old language and there are a lot of versions to keep it up-to-date. But the preferred version of the Java is 1.8. It's the most compatible one for developers. In further versions of java, they are trying to add lambda and stream functions which helps to write less code but do a lot.

In conclusion, in this section of the paper I have covered cons and pros of the programming language that I have used to build the whole logic and processes. There are more advantages over disadvantages in Java.

**-Spring Framework.** Spring framework has introduced in 2003 [13]. It's an open-source and easy to use for web application development. It's also one of the most popular Java EE supported framework in Java that we can build enterprise applications. Java provides a lot of frameworks such as Hibernate, JSF, Servlet, Maven and much more. The main goal here in Spring is to gather all of them together and build structured and fully-functional applications. Spring is considered to be lightweight which means it helps to improve the coding time so we can code efficiently. Mostly in spring we will be using annotations such as visible in the Figure. Here we can encounter with a lot of annotations and each of them provides different functionality. Here in this figure, we have listed the ones which are frequently used and we can give more information about them.

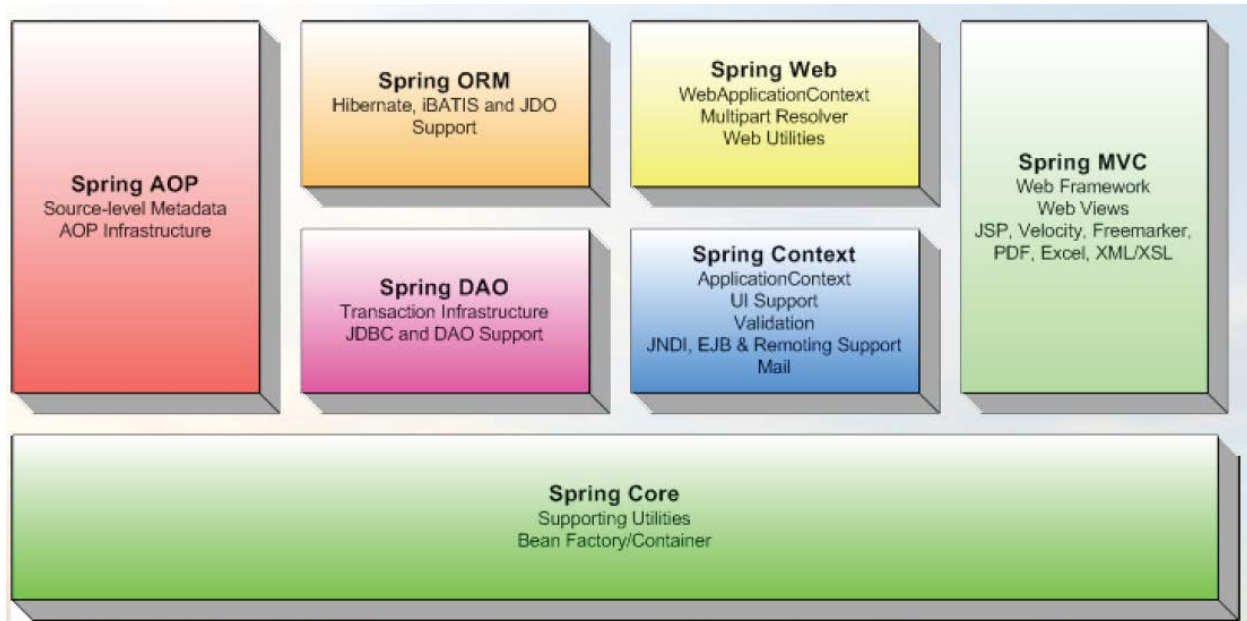


The main core logic of Spring framework is DI and IOC which means Dependency Injection and Inversion of Control. Dependency Injection is mainly used in Spring framework the goal here is that to say spring to create the instance of the class by actually not creating it. All we have to do just put an annotation top of the class and Spring will provide the bean of this class. All we have to do is use `@Autowired` annotation when we declare that class. We can see this example in figure below. As we can see we have created the class `CourseManagement` and put an annotation called `@Component` which will create the instance of this class as soon as Spring context starts. Later on in service class all we have to put is `@Autowired` annotation at the top of instantiation.

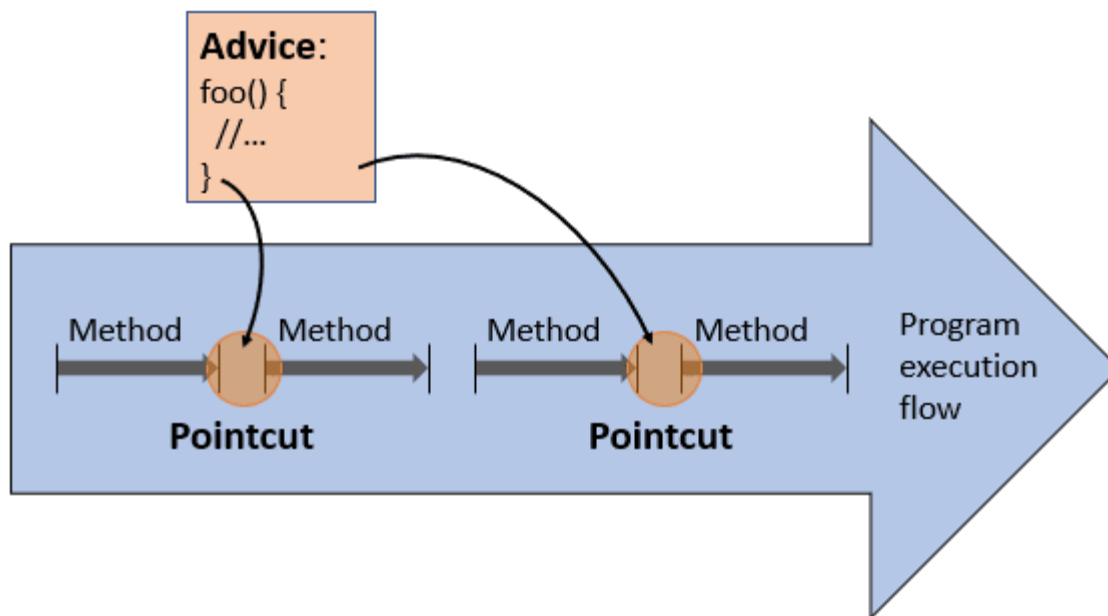
```
@Component("fooFormatter")
public class CourseManagement {
    public String format() {
        return "foo";
    }
}

@Component
public class FooService {
    @Autowired
    private CourseManagement courseManagement;
}
```

There are a lot of parts of Spring framework that helps to achieve many goals. Each of them is playing different role and has different concept in order to create a fully-functional website. We have almost used all of them throughout the development process. There are Spring AOP, ORM, WEB, MVC, DAO and much more that we can see in the figure attached below. We will give brief information about each of them and the place where we have used.



**Spring AOP** – AOP means that Aspect Oriented Programming. The main goal here is to add other functionality to the method or code without actually modifying it. It's quite required when it comes to log, or do something before a particular logic that has executed. There are core concepts of AOP which are essential such as Joint point, advice, pointcut. Advice also has its components which are the making whole logic here. They are like @After, @Before, @Around as the name refers for example when we use @After it's like doing the proper logic after the execution of method which can be logging or different logic based on the functionality. [14]





**ORM** – ORM basically means Object Relational Mapping which is used so frequently in the development process. It's actually a framework which brings with itself Hibernate, JPA, JDO and iBatis we will be using JPA throughout the development. Spring JPA provides a lot of functionalities for the development and makes it easier to develop fast and easy to maintain applications. JPA means Java Persistence API here we are accessing data with the less effort. And we can achieve CRUD operations much easier. JPA helps us to not to write any SQL queries but instead all we have to do is create the POJO class and create the Repository of this class let's take a look at below example. As we can see we have User class where we have a lot of fields. In order to create the table in database for such class would require quite long query where in Spring all we have to do is put an annotation called @Entity. On the other hand, @Data annotation helps to not to deal with the legworks such as creating the getters and setters of this class's fields. @Builder is a design pattern that we can implement just only using by one annotation which is quite time saving. Furthermore, when it comes to accessing data from the database for this entity it's quite easier.

```
@Data
@Entity
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Table(name = User.TABLE_NAME)
//PMD.TooManyFields/
public class User implements UserDetails {

    ...

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private String password;

    private String name;

    private String about;

    private String phone;

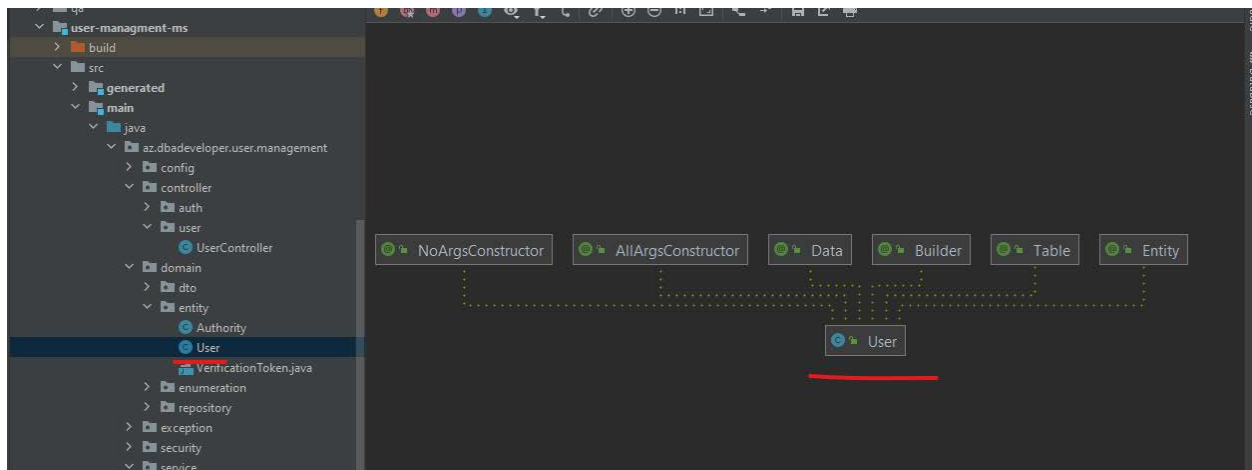
    @Column(nullable = false, unique = true)
    private String username;

    private String avatar;

    private String title;

    private String background;
```

Here on the FIGURE, we can see all the used annotations on the class level.



In order to access the data of this class all we have to do is create an interface and extend it from JpaRepository class also this class takes two generics which are the first one is the class itself the second one is the actual id in the class such way this class will serve to manage the data access. All we have to do is just create methods with the proper names as we can see we have created findByUsername method where we are expecting User. But we haven't written any SQL queries.

```

@Repository
public interface UserRepository extends JpaRepository<User, Long>, JpaSpecificationExecutor<User> {

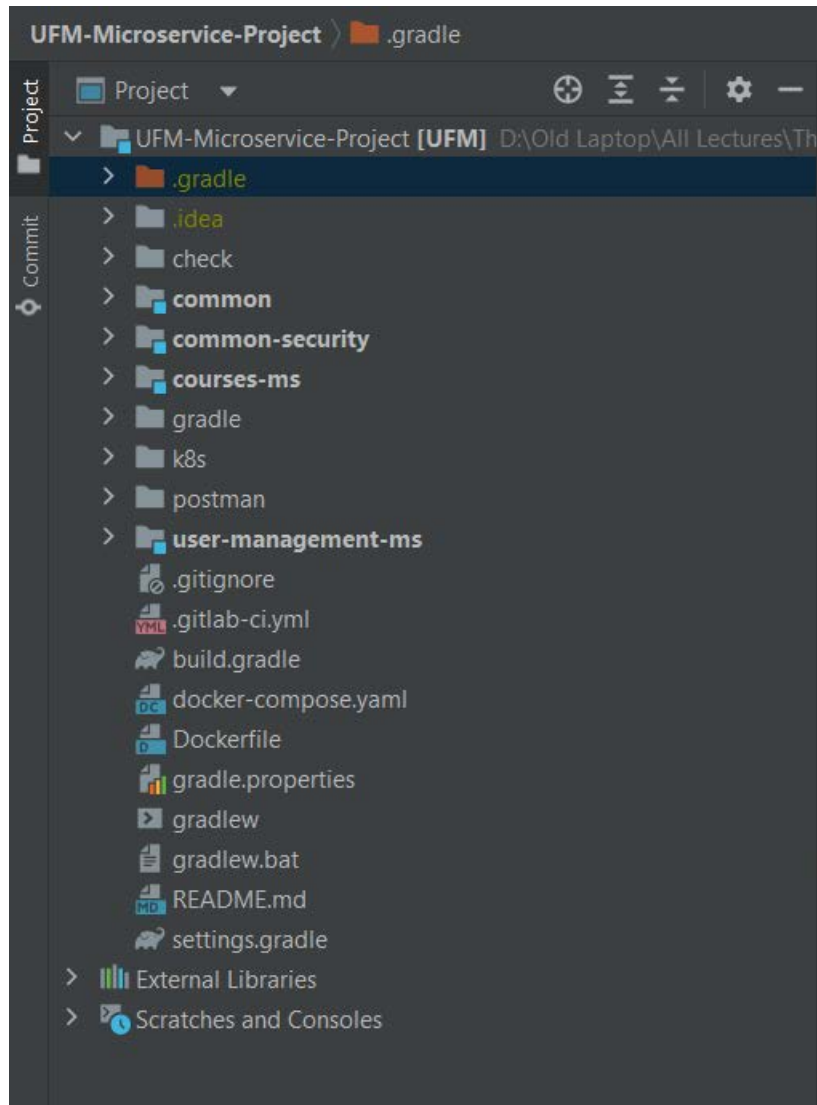
    // @EntityGraph(attributePaths = "authorities")
    Optional<User> findByUsername(String email);

    @Override
    @EntityGraph(attributePaths = {"authorities"})
    Optional<User> findById(Long id);

    @Override
    @EntityGraph(attributePaths = {"authorities"})
    void deleteById(Long id);
}

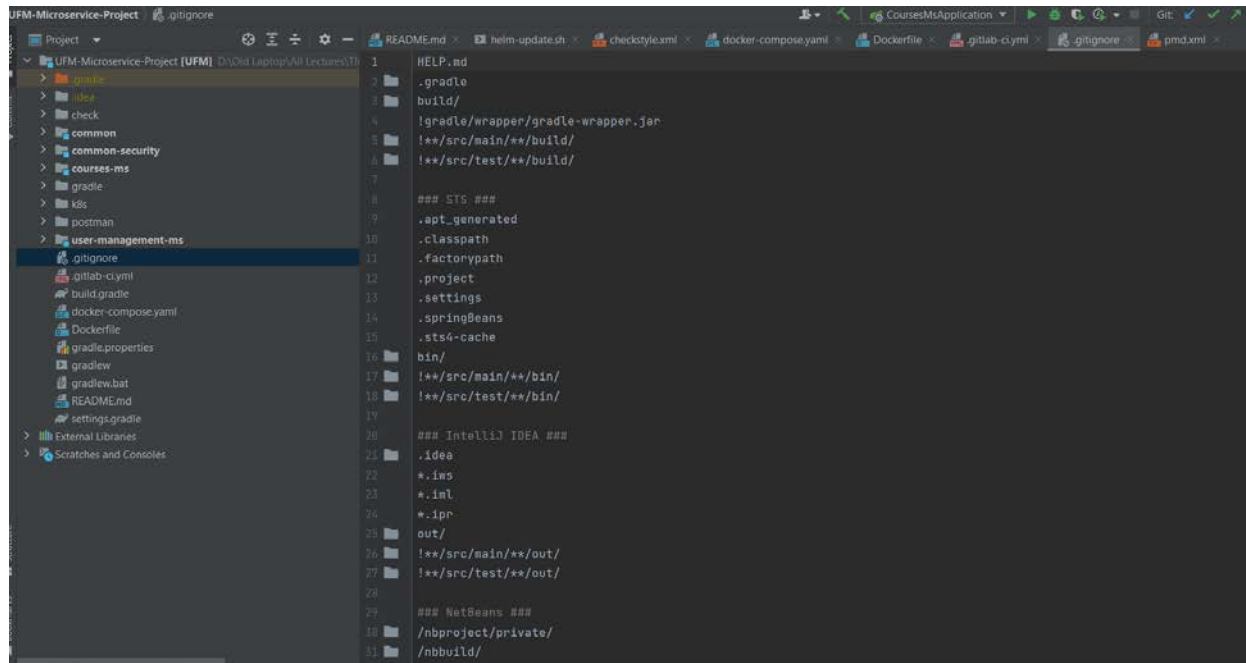
```

In the figure we can see the project folders and microservices. There are all separated from each other and each of them serves for a particular goal. The structure is quite important when it comes to develop a high usability application such as mine and maintaining microservices is quite challenging. As we can see figure below there are a lot of packaging and files which will be explained in the below section of this paper.

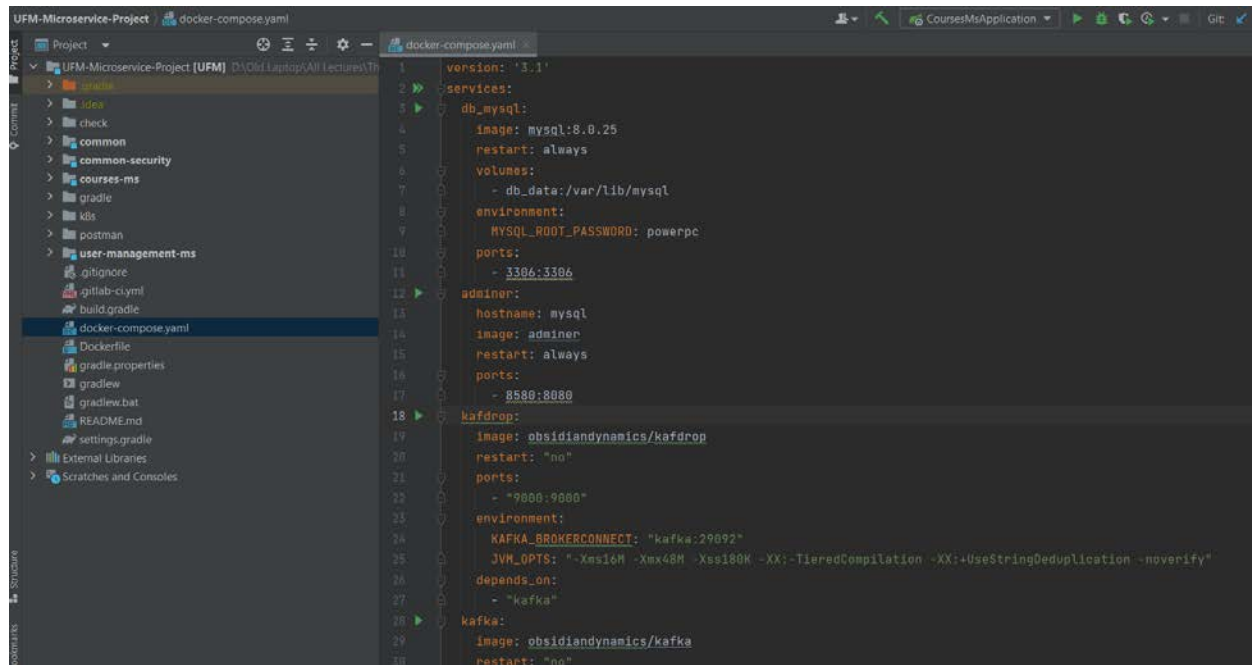


As we can see inside of the project there are a lot of subfolders and files which are doing a bunch of work. Let's talk about each of them separately and briefly:

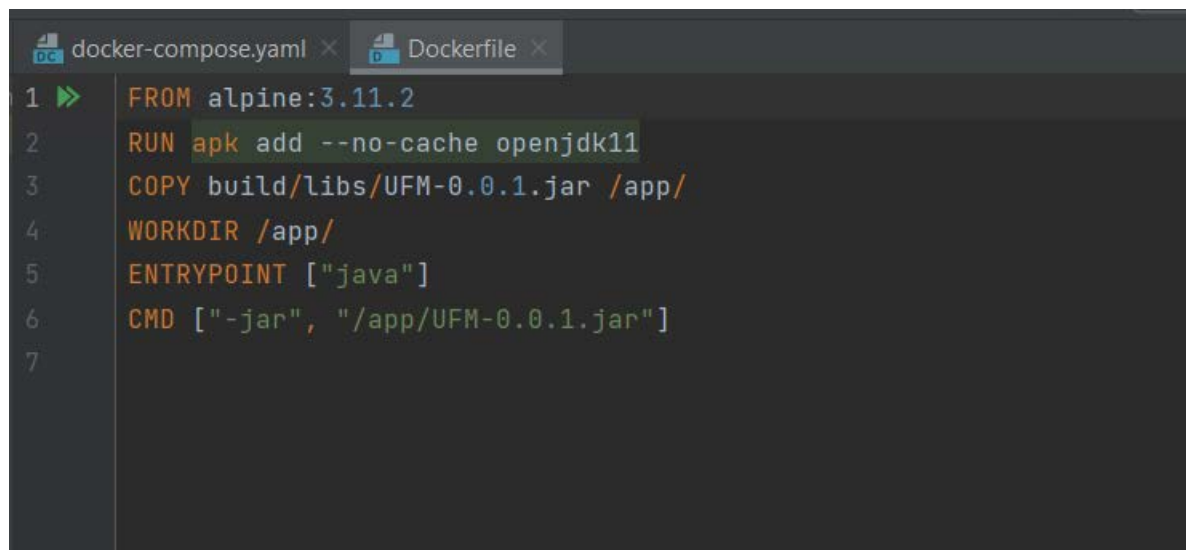
**Gitignore.** The file called **.gitignore** is used to ignore the files which not needed to push to the repository as we can see in the below figure there are several folders which will be ignored by git and won't be tracked in such way it won't be visible in the repository. This is quite important when it comes to ignore some files called idea which is only important for the local machine and belong to the IDE.



**Docker-compose.** The next file is **docker-compose** file which is quite important when it comes to building scaled microservices application. Docker-compose file contains a lot of services and images which means for example we are using MySQL database and in order to connect the database locally we need to have it in our machine. And installing database to our machine can take a lot of time if we would do it manually. Docker gives us such opportunity to do it within minutes. Docker-compose file gives us possibility to run a lot of services and images at one time for example by one command we can bring up MySQL, Kafka, Adminer, and much more. The command to bring them up is “docker-compose up” just basically downloads and installs the images services into machine and gives us possibility to use them. Docker-compose file looks like as in the figure attached below. [15]

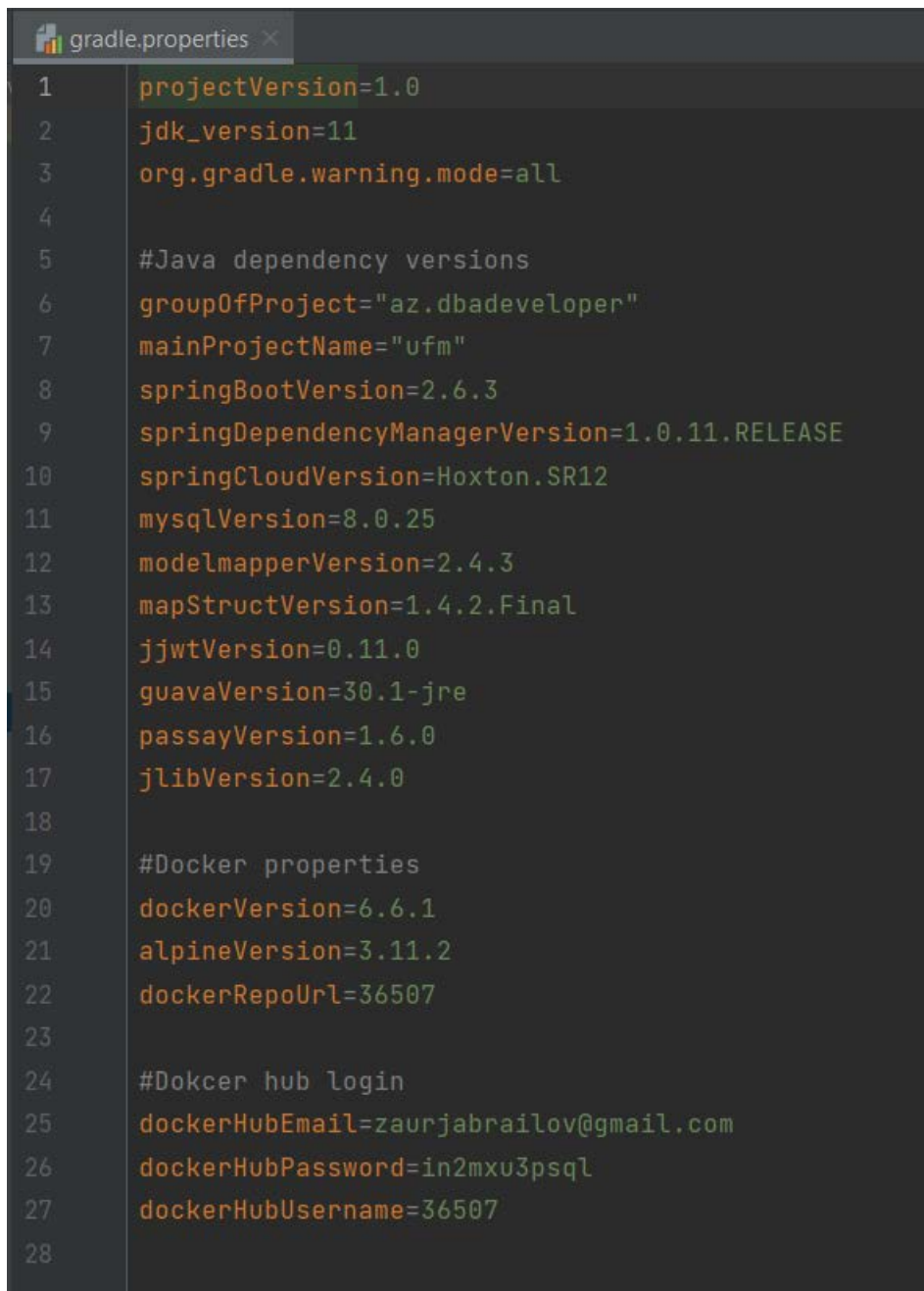


**Dockerfile.** A Dockerfile is quite connected with the docker-compose file as it all comes from the Docker. It's the same type of file as it is text file and helps to build the application and we can just call it by one command and build the image of the project. In our case it would be JAR file because it's a java-based project. The Dockerfile of the project is attached below in FIGURE which can be executed by one click or command.



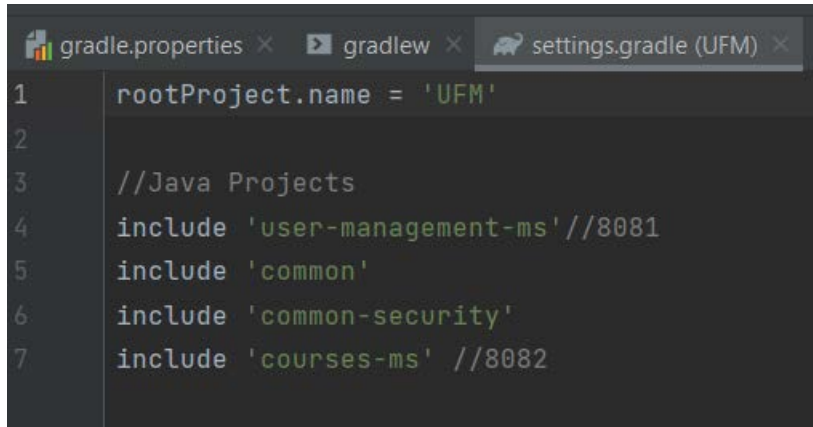
**Gradle-properties.** This file is a text file and contains some needed information for whole microservices application such as here we can declare the java version which is JDK 11. Most of the time this file is used to declare and store versions of such files. Let's take a look at

the below figure and see what are the use-cases. As we can see a lot of versioning and the login credentials of docker.



```
1 projectVersion=1.0
2 jdk_version=11
3 org.gradle.warning.mode=all
4
5 #Java dependency versions
6 groupOfProject="az.dbadeveloper"
7 mainProjectName="ufm"
8 springBootVersion=2.6.3
9 springDependencyManagerVersion=1.0.11.RELEASE
10 springCloudVersion=Hoxton.SR12
11 mysqlVersion=8.0.25
12 modelmapperVersion=2.4.3
13 mapStructVersion=1.4.2.Final
14 jjwtVersion=0.11.0
15 guavaVersion=30.1-jre
16 passayVersion=1.6.0
17 jlibVersion=2.4.0
18
19 #Docker properties
20 dockerVersion=6.6.1
21 alpineVersion=3.11.2
22 dockerRepoUrl=36507
23
24 #Dokcer hub login
25 dockerHubEmail=zaurjabrailov@gmail.com
26 dockerHubPassword=in2mxu3psql
27 dockerHubUsername=36507
28
```

**Settings-gradle.** This file is quite important when it comes to execute the microservice application successfully because here we say the build tool that we have got particular microservices. As we can see in the below figure there are 4 microservices and also we have declared the name of the root project which is UFM.



```
1 rootProject.name = 'UFM'
2
3 //Java Projects
4 include 'user-management-ms' //8081
5 include 'common'
6 include 'common-security'
7 include 'courses-ms' //8082
```

The rest of the files are related to configuration stuff of the project except the README.md file which is used to give essential information and guidelines throughout the project. It's like a text file but there are a lot of formatting tags such as HTML but it's not written like in the html instead here, we use some symbols in order to make space or make the header big.

**-UI.** The term UI means User-interface which is quite important when it comes to present a user-friendly website to the end-user. This part is crucial and essential part of implementation process. UI is the heart of project where user interacts with the website. Think about bad designed without any navigation website but well-designed backend if the User-interface is not well designed enough it won't succeed. Therefore, UI plays an essential role in the web development. This is where we present all the things that we have developed throughout the process to the user. UI is totally important for us and plays a huge role in the development. The main part of the UI is login page where you can see on the figure below which we have divided the page into two sections in the first tab we are letting user to sign in and on the other side we are giving user information about the platform such as values and passions.



## Sign in

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

Email address

zaurjabrailov@gmail.com

Password

.....

☐ Remember me

[Forgot password?](#)

Sign in

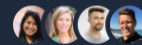
Or continue with



INGRESS ACADEMY  
empowered by innovation

## Welcome to Ingress Academy

Ingress Learning Management System (LMS), by the students, for the to students, to organize and manage the learning process to maximize knowledge sharing & collaboration. Join us and start learning with us today.



More than 250 people joined us, it's your turn

After successful registration we will see such page where we can categorize courses and manage it truthfully. Here we can do a lot of actions. Let's say user wants to change the profile picture which can be done by a few clicks. Or we can add new course into our platform and update the details which is fully functional and can serve for a lot of purposes.

The screenshot displays the Ingress Academy dashboard. On the left is a dark sidebar with the user's profile 'Zaur Jabrailov' and a menu with 'Categories', 'Courses', 'Course Sessions', and 'Lessons'. The main content area is titled 'Categories' and shows a list of 7 categories: Software Development, DevOps & Automation, Linux System Administration, Computer Networking, Cyber Security, IT Essentials, and Database. Each category has a corresponding icon and a brief description of the training programs. A search bar and an 'Add' button are located at the top right of the categories list.

Category	Training Programs
Software Development	Software Development Training Programs
DevOps & Automation	Software and Automation Training Programs
Linux System Administration	Linux System Administration Training Programs
Computer Networking	Computer Networking (CISCO) Training Programs
Cyber Security	Cyber Security Training Programs
IT Essentials	IT Essentials Training Programs
Database	Database Administration & Development



**Database design.** Database design is very crucial when it comes to developing a fully functional website. Database helps us to store our data such as user information, cake names and everything related to data. For our web application we are using MySQL which is one of the best. There are lots of advantages of using such database:

- Here we can run the queries in less amount time when we compare to other database.
- Combability of entity framework and almost all ORMs.
- It complies with ACID structure.

When it comes to database it should be also secured and maintained. For example, it should be protected from database attacks such as SQL injection, packet sniffing. Even admin of the website shouldn't be able to access customers' sensitive data such as their passwords. For those reasons, it's always recommended to encrypt the passwords. Encrypting can be done with one key word and algorithm which is known by the software and all the letter and numbers will be changed to some random letters and numbers based on that keyword and algorithm. There are a lot of password encryption methods such AES, DES and etc. So, such way even the admin of the website won't be even able to see the users' passwords.

Furthermore, if we take a look at the below Figure, we can see the Entity relation where we can see all the relations between entities and fields of entities.

Figure 16 – Entity relations table

## B. Used Technologies

**Kubernetes.** Kubernetes is one of the best tools that we have to use when we are developing Microservices application [16]. The main purposes of Kubernetes are automating manual processes such as deploying, scaling, redeploying and much more related to Network admin. By using Kubernetes, we are doing a lot of work that would require a lot of people to do. If we look at the below figures we can see the workloads and as well as services that we have deployed to google cloud platform.

Free trial status: z1930.20 credit and 77 days remaining - with a full account, you'll get unlimited access to all of Google Cloud Platform.

Google Cloud Platform My First Project Search products and resources

Kubernetes Engine

Workloads REFRESH DEPLOY DELETE

Cluster Namespace RESET SAVE

OVERVIEW COST OPTIMIZATION PREVIEW

Filter Is system object: False Filter workloads

Name	Status	Type	Pods	Namespace	Cluster
courses-ms-deployment	OK	Deployment	3/3	ingress-ms-ns	ingressms
courses-ms-deployment	OK	Deployment	3/3	ingress-ms-dev	ingressms
grafana	OK	Deployment	1/1	istio-system	ingressms
istio-egressgateway	OK	Deployment	1/1	istio-system	ingressms
istio-ingressgateway	OK	Deployment	1/1	istio-system	ingressms
istiod	OK	Deployment	1/1	istio-system	ingressms
jaeger	OK	Deployment	1/1	istio-system	ingressms
kiali	OK	Deployment	1/1	istio-system	ingressms
mysql-deployment	OK	Deployment	1/1	ingress-ms-ns	ingressms
mysql-deployment	OK	Deployment	1/1	ingress-ms-dev	ingressms
prometheus	OK	Deployment	1/1	istio-system	ingressms
management-ms-deployment	OK	Deployment	3/3	ingress-ms-ns	ingressms

Google Cloud Platform My First Project Search products and resources

Kubernetes Engine

Services & Ingress REFRESH CREATE INGRESS DELETE

Cluster Namespace RESET SAVE

SERVICES INGRESS

Services

Filter Is system object: False Filter services and ingresses

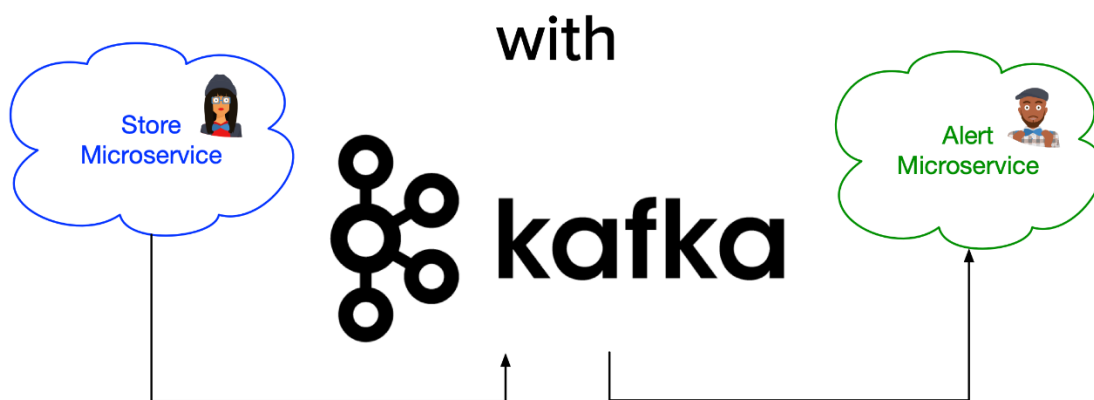
Name	Status	Type	Endpoints	Pods	Namespace	Clusters
courses-ms-service	OK	Node Port	10.36.12.224:8082 TCP	3/3	ingress-ms-ns	ingressms
dev-ingressmstemp-courses-ms	OK	Cluster IP	10.36.9.160	3/3	ingress-ms-dev	ingressms
dev-ingressmstemp-mysql	OK	Cluster IP	10.36.1.212	1/1	ingress-ms-dev	ingressms
dev-ingressmstemp-user-management-ms	OK	Cluster IP	10.36.14.168	3/3	ingress-ms-dev	ingressms
grafana	OK	Cluster IP	10.36.1.109	1/1	istio-system	ingressms
istio-egressgateway	OK	Cluster IP	10.36.10.236	1/1	istio-system	ingressms
istio-ingressgateway	OK	External load balancer	34.72.125.105:15021	1/1	istio-system	ingressms
istiod	OK	Cluster IP	10.36.14.245	1/1	istio-system	ingressms
jaeger-collector	OK	Cluster IP	10.36.4.251	1/1	istio-system	ingressms
kiali	OK	Cluster IP	10.36.8.98	1/1	istio-system	ingressms
mysql-service	OK	Cluster IP	10.36.8.219	1/1	ingress-ms-ns	ingressms
prometheus	OK	Cluster IP	10.36.1.35	1/1	istio-system	ingressms
tracing	OK	Cluster IP	10.36.12.122	1/1	istio-system	ingressms
user-managment-ms-service	OK	Node Port	10.36.5.142:8081 TCP	3/3	ingress-ms-ns	ingressms
zipkin	OK	Cluster IP	10.36.6.144	1/1	istio-system	ingressms

istio-egressgateway OK Cluster IP 10.36.10.236

istio-ingressgateway OK External load balancer 34.72.125.105:15021

**Kafka.** Kafka in microservices architecture is used to communicate services with each other [17]. It's like messaging system that receives message and keeps it on real time for the target. We will take a look of the implementation of Kafka in our project. Before that we can take a look at the below figure and see the points of Kafka as we can see two microservices talk to each other using Kafka.

## Communicate Between Microservices



We need to main classes in order to make Kafka work the first one is `KafkaTopicConfig` and the other one is `KafkaProducerConfig`. Each of them has some type of configuration in order to make Kafka work as we can see on the figure below.

```
@Configuration
public class KafkaTopicConfig {

    1 usage
    @Value("${spring.kafka.topics.signUp}")
    private String signUp;

    1 usage
    @Value("${spring.kafka.topics.update}")
    private String update;

    zaurjabrailov
    @Bean
    public NewTopic topicUserSignUp() { return new NewTopic(signUp, numPartitions: 10, (short) 1); }

    zaurjabrailov
    @Bean
    public NewTopic topicUpdate() { return new NewTopic(update, numPartitions: 10, (short) 1); }

}
```

**Gitlab.** Gitlab is just like a platform like Github but the main difference is just UI. In order to manage our code, we are using Gitlab and its features such as GitlabCI. At the below figure we can see how Gitlab looks like and we can see the repository version of our code. Basically, Gitlab is a code repository where we can manage our code and store it. This is the best when it comes to collaboration with teams or group of people. There are a lot of cool technologies for companies. The main benefit of using GitLab is that it allows all the team members to collaborate in every phase of the project. GitLab offers tracking from planning to creation to help developers automate the entire DevOps lifecycle and achieve the best possible results. More and more developers have started to use GitLab because of its wide assortment of features and brick blocks of code availability.

Zaur Jabrayilov &gt; UFM-Microservice-Project

**UFM-Microservice-Project**

Project ID: 33451002

Star 0

122 Commits 4 Branches 0 Tags 246 KB Project Storage

master

UFM-Microservice-Project

Find file

Clone



Merge branch 'feature/devZaur-CheckStyle-Step1' into 'master'

Zaur Jabrayilov authored 2 months ago

2c504b9d



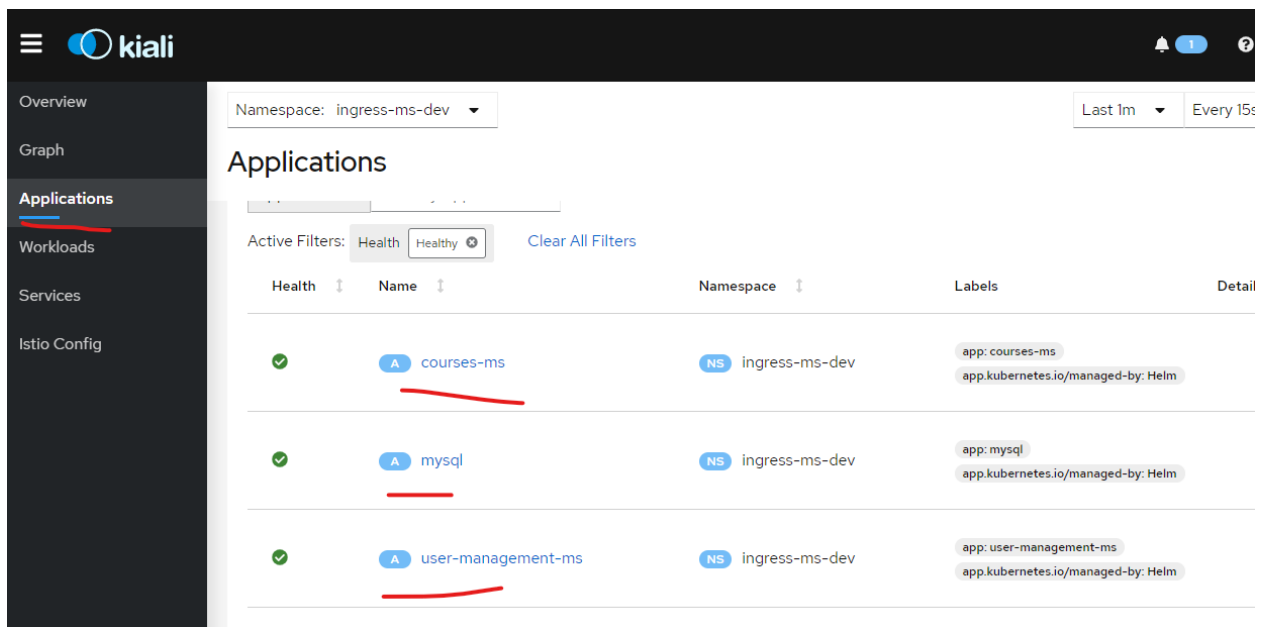
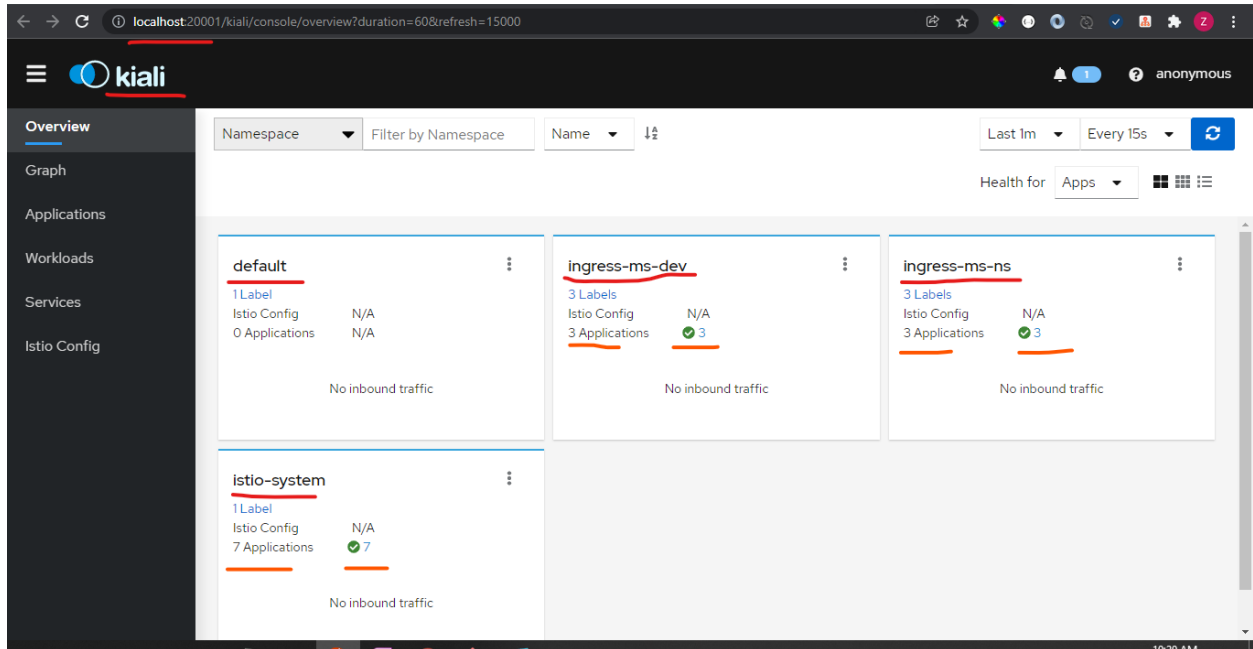
README

CI/CD configuration

No license. All rights reserved

Name	Last commit	Last update
check	Add Delete From UserManMs	4 months ago
common-security	Add CheckStyle and PMD and first check all ...	4 months ago
common	Add CheckStyle and PMD and first check all ...	4 months ago
courses-ms	check ci/cd	3 months ago
gradle/wrapper	Create Common Module, change something...	6 months ago
k8s	check ci/cd	3 months ago
postman	add findByUsername to UserMS and Change...	2 months ago
user-management-ms	add findByUsername to UserMS controller	2 months ago
.gitignore	Create UFM Microservice Project	6 months ago
.gitlab-ci.yml	Comment docker push in .gitlab-ci.yml	2 months ago

**Kiali.**



**Figma.** During the UI development process Figma is very essential because it makes things easier. It helped a lot to draw wireframes rather than on paper. Not only wireframes but also mockups, even it can be used to create logo for the website. Figma provides features animations, slides. For example, we want to build a web application and we want to create a prototype. We can easily create and manage it here because there is such functionality that pressing on one design of the button or something will open other page which is very useful. This feature is also called Prototype in Figma.

**Bootstrap.** It's so popular for front-end developers. It's like a framework that we can use to build standalone and user-friendly designs. One of the best features of it is that it's an open-source framework which provides a lot of functionalities. I have used some of the libraries in order to speed up the front-end design of the website. It can be helpful when you are in rush and you don't want to play with CSS for long time. In order to use the bootstrap and start implementing it in our website all we have to do is that to add a link into our html code into our head tags which will be containing bootstrap framework library. Which is just one line of code and tons of benefits.

**IntelliJ Ultimate.** This is the IDEA that we have used throughout the development process and it's one of the best when it comes to developing Java EE applications. There are two versions of this IDEA there are like Ultimate version and community edition. [18]

**Community Edition:** open source and available free of charge. The Community Edition is covered by the Apache 2.0 license, and is built together with the open community around jetbrains.org. You can do all the things such as development here but you won't be getting premium features.

**Ultimate Edition:** professional and fully-featured commercial IDE provided by JetBrains. The Ultimate Edition is built on top of the Community Edition with many extra features for web and enterprise development. You can download IntelliJ IDEA Ultimate from the website of JetBrains. You can evaluate it for free for 30 days, purchase a license, or get a free license if you're a contributor to an established open-source project.

## Conclusion

In conclusion, creating such website was really challenging and also helpful to reuse my knowledges that I have gained during the studies.

The ideas presented in this paper associates the system design, architecture, and testing are helpful for researchers who would have an intention to develop other kinds of web applications.

I would like to conclude that my web project aims to help and serve people to manage their courses easily and in fast manner of time. Developing such website needs quite effort and in-depth understanding of the project so in order to make it work. As by myself I have went through a lot of researches and diagrams in order to make the development process as simple as possible. But apart from everything the testing phrase is quite important that we haven't done yet but we are planning to make it on prod level which is the most dangerous level and expensive. But in the future process we are also planning to add some unit tests where we can test some scopes such as methods and if a new change comes so we won't break the existing code. Furthermore, we are planning to develop a mobile application of this software so it can be easier

to use rather the laptop. We would like to implement those features in future updates and also add additional functionalities as the system has room for improvement

### **Source Code**

The source code of the project can be found here: <https://gitlab.com/zaurjabrailov/UFM-Microservice-Project>




## References

1. Examples of SQL injection attacks. More information: <https://brightsec.com/blog/sql-injection-attack/>
2. Google classroom. Taken from: <https://www.epicurious.com/expert-advice/types-of-cake-glossary-article>
3. Domain prices. Can be found here: <https://www.godaddy.com/pl-pl>
4. Why requirement analysis is important the taken article can be found here: <https://invozone.com/blog/importance-of-requirement-analysis-process-in-software-development/#:~:text=Requirements%20analysis%20helps%20to%20make,matters%20more%20than%20anywhere%20else.>
5. MVC architecture detailed information can be found here: <https://www.interviewbit.com/blog/difference-between-spring-mvc-and-spring-boot/>
6. About Peter Rodger and his first application. Can be found here: <https://www.sumerge.com/what-is-microservices-architecture/>
7. Amazon and Netflix usage of microservice. Taken from: <https://techsur.solutions/why-companies-like-netflix-uber-and-amazon-are-moving-towards-microservices/>
8. More information about SOLID principles. Can be found here: <https://www.baeldung.com/solid-principles>
9. The infrastructure of the Microservices architecture and detailed information can be found here: <https://www.martinfowler.com/articles/microservices.html?ref=wellarchitected>
10. Introduction date and more information about UML diagrams can be found here: [https://www.gfa-group.de/web-archive/inspire/www.inspiration-westernbalkans.eu/5/9/5/3/7/7/Introduction\\_to\\_the\\_Unified\\_Modeling\\_Language\\_UML\\_.pdf](https://www.gfa-group.de/web-archive/inspire/www.inspiration-westernbalkans.eu/5/9/5/3/7/7/Introduction_to_the_Unified_Modeling_Language_UML_.pdf)
11. Detailed and in-depth understanding of relationship between entities. Can be found here: <https://beginnersbook.com/2015/04/e-r-model-in-dbms/>
12. More information about activity diagrams. Can be found here: <https://creately.com/blog/diagrams/activity-diagram-tutorial/>

13. Spring framework. More information taken from: <https://docs.spring.io/spring-framework/docs/current/reference/html/overview.html>
14. Spring AOP approach. Got information from: <https://docs.spring.io/spring-framework/docs/2.5.4/reference/aop.html>
15. Docker general information about docker-compose as well as Dockerfile. Can be found here: <https://medium.com/bb-tutorials-and-thoughts/15-useful-docker-compose-commands-for-everyday-development-8d5340d641c7>
16. The book about the Kubernetes technologies. Can be found here: <https://www.oreilly.com/library/view/kubernetes-up-and/9781098110192/ch16.html>
17. Kafka messaging queue information can be found here: <https://www.alibabacloud.com/help/en/message-queue-for-apache-kafka/latest/message-queue-for-apache-kafka-sla>
18. The versions of IntelliJ IDEA and details of each version. Can be found here: <https://www.jetbrains.com/help/idea/supported-java-versions.html>

## Appendices




Name

Title

Category

Description

 Server Configuration, Enterprise Manager Grid Control, Managing Database Availability, Data Management, Data Warehouse Management, Performance Management, Grid Infrastructure, and ASM, Real Application Clusters, Data Guard.

Delete

Cancel

Save

The screenshot displays the Ingress Academy LMS interface. The top header includes the Ingress Academy logo and a search icon. The main content area is divided into three sections: 'Students', 'Users', and 'Attendances'.

**Students Section:** The title 'Students' is followed by '14 students'. Below this is a search bar labeled 'Search students' and a blue '+ Add' button.

**Users Section:** The title 'Users' is followed by 'No users'. Below this is a search bar labeled 'Zaur Jabrailov'.

**Attendances Section:** The title 'Attendances' is followed by '1 attendance'. Below this is a search bar labeled 'Search attendances' and a blue '+ Add' button. A table below the search bar shows a single entry: 'M Microservices 6'.

The left sidebar contains the Ingress Academy logo and a list of navigation items: 'Categories', 'Courses', 'Course Sessions', and 'Lessons' (which is highlighted). The user's name 'Zaur Jabrailov' is displayed at the top of the sidebar.

## Lessons

13 lessons

 Search lessons

 Add

2

2

### 29. Kubernetes Introduction

Pods, Nodes, Kubernetes Cluster Components and Core K8S Concepts

3

3

### 30. Kubernetes 2

NodePort, ClusterIP, Service, Auto Discovery

3

### 31. Kubernetes

Networking, Subnet, Pod to Pod, Container to Container, Service to Pod communication

N


 Edit

## New Lesson

Name

 ELK

Notes

 Elasticsearch, Log stash, Kibana Zaur Jabrailov ☒ Yes