AISSMS
INSTITUTE OF INFORMATION TECHNOLOGY
ADDING VALUE TO ENGINEERING

Department of Computer Engineering

Object-Oriented Modeling and Design

Project Report On

UML DIAGRAMS ON RECOMMENDATION SYSTEM

SUBMITTED TO THE DEPARTMENT OF COMPUTER ENGINEERINGAISSMS IOIT

BE Computer Engineering

SUBMITTED BY

STUDENT NAME	ERP No:
Onasvee Banarse	**09**
Kaustubh Kabra	**37**
Akash Mete	**50**
Harsh Shah	**65**

2022 -2023

Department of Computer Engineering

CERTIFICATE

This is to certify that the project report

"UML DIAGRAMS ON RECOMMENDATION SYSTEM"

Submitted by

STUDENT NAME	ERP No:
Onasvee Banarse	09
Kaustubh Kabra	37
Akash Mete	50
Harsh Shah	65

is a bonafide students at this institute and the work has been carried out by them under the supervision of **Prof. Prajwal Gaikwad** and it is approved for the partial fulfillment of the Department of Computer Engineering AISSMS IOIT.

(**Prof. Prajwal Gaikwad**) (**Dr. S.N.Zaware**)

Mini-Project Guide Head of Computer Department

Place: Pune Date:

Abstract

The Unified Modeling Language (UML) was created to forge a common, semantically, and syntactically rich visual modeling language for the architecture, design, and implementation of complex software systems both structurally and behaviourally. UML has applications beyond software development, such as process flow in manufacturing. UML diagrams make it easier to identify the requirements and scopes of systems and applications by providing visual models. The main findings showed that UML diagrams were mostly used for the purpose of design and modeling, and class diagrams were the most used ones.

Object-oriented languages dominate the programming world because they model real-world objects. UML is a combination of several object-oriented notations: Object-Oriented Design, Object Modeling Technique, and Object-Oriented Software Engineering. UML represents best practices for building and documenting different aspects of software and business system modeling.

Our aim is to provide a comprehensive summary of UML diagrams and implement all types of UML diagram to our project based on Content Recommendation System.

Contents

1. Introduction

UML is a truly large notation offering many different diagrams, and for each diagram it provides a large set of constructs covering any possible need of any modeller for any possible task. The UML metamodel is large and quite complex, and the definition and the understanding of its static and dynamic semantics is a truly difficult task, with also the consequence to make difficult to teach it both at the school/university level or in the industry. Moreover, the large number of constructs and the consequent very large metamodel make complex and time-consuming developing transformation of UML models and building tools for the UML. Clearly, these features of the UML have a negative impact on how the UML is perceived by the modellers hindering its adoption and leading in some cases to replace the UML by ad-hoc lean and simple DSLs (Domain Specific Languages).

OOA, OOD, and UML-based diagrams typically used in the software development approach. OOA, OOD approaches present an excellent framework for eventualities. A use case is a widespread description of a whole transaction involving a few objects of the procedure. The UML language supplies a suitable framework for situation acquisition making use of use case diagrams like sequence, collaboration diagrams. A Scenarios is defined as an illustration of a given use case. Scenarios viewed in two exclusive approaches via sequence diagrams and collaboration diagrams. Each variety of diagrams depends on the equal underlying semantics. Conversion from one to the opposite is possible. In this paper, we propose a requirement engineering process that composes UML eventualities for receive a global description of a given service of the procedure and implementation code from the UML use case (service).

2. Problem Statement

Draw all UML diagrams for your project work. Drawing all UML Diagrams for Content Recommendation System.

In this Project we will focus on drawing UML diagrams based on our project which is used to recommend content (movies, books, etc) to user. This is the list of UML diagrams for the project:

- State Diagram
- Class Diagram
- Use Case Diagram
- Flowchart Diagram
- Activity Diagram
- Component Diagram
- Deployment Diagram

3. Software Requirement Specification

Software Used:

- **StarUML –**

  StarUml is a software engineering tool for system modeling using the Unified Modeling Language, as well as Systems Modeling Language, and classical modeling notations.

- **Visual paradigm enterprise-**

  Visual Paradigm is a software application designed for software development teams to model business information systems and manage development processes. In addition to modeling support, this technology provides report generation and code engineering capabilities including code generation.

- **Lucidchart (online)–**

  Lucidchart is a web-based diagramming application that allows users to visually collaborate on drawing, revising and sharing charts and diagrams, and improve processes, systems, and organizational structures.

4. Theory

1. Object-oriented programming

Object-oriented programming (OOP) is a programming paradigm based on the concept of "objects", which can contain data and code: data in the form of fields (often known as attributes or properties), and code, in the form of procedures (often known as methods). The object-oriented programming is basically a computer programming design philosophy or methodology that organizes/ models software design around data, or objects rather than functions and logic. An object is referred to as a data field that has unique attributes and behaviour. Everything in OOP is grouped as self-sustainable objects.

1.1 Pillars of OOPs

The major concepts that we have discussed above are known as pillars of OOPs. There are four pillars on which OOP rests.

- Abstraction
- Encapsulation
- Inheritance
- Polymorphism

2. OBJECT ORIENTED MODELING

Object oriented modeling is entirely a new way of thinking about problems. This methodology is all about visualizing the things by using models organized around real-world concepts. Object oriented models help in understanding problems, communicating with experts from a distance, modeling enterprises, and designing programs and database. We all can agree that developing a model for a software system, prior to its development or transformation, is as essential as having a blueprint for large building essential for its construction. Object oriented models are represented by diagrams. OOM approach is an encouraging approach in which software developers must think in terms of the application domain through most of the software engineering life cycle.

In OOM the modeling passes through the following processes:

- ❖ System Analysis
- ❖ System Design
- ❖ Object Design, and
- ❖ Final Implementation

3. UML (Unified Modeling Language)

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG). It was initially started to capture the behaviour of complex software and non-software system and now it has become an OMG standard. UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.

OMG is continuously making efforts to create a truly industry standard.

- UML stands for **Unified Modeling Language**. UML is a pictorial language used to make software blueprints.
- UML is different from the other common programming languages such as C++, Java, COBOL, etc.
- UML can be described as a general-purpose visual modeling language to visualize, specify, construct, and document software system.

UML is not a programming language, but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object-oriented analysis and design.

3.1 Role of UML in OO Design

UML is a modeling language used to model software and non-software systems. Although UML is used for non-software systems, the emphasis is on modeling OO software applications. If we investigate class diagram, object diagram, collaboration diagram, interaction diagrams all would basically be designed based on the objects. Hence, the relation between OO design and UML is very important to understand. The OO design is transformed into UML

diagrams according to the requirement. Before understanding the UML in detail, the OO concept should be learned properly.

3.2 UML - Modeling Types

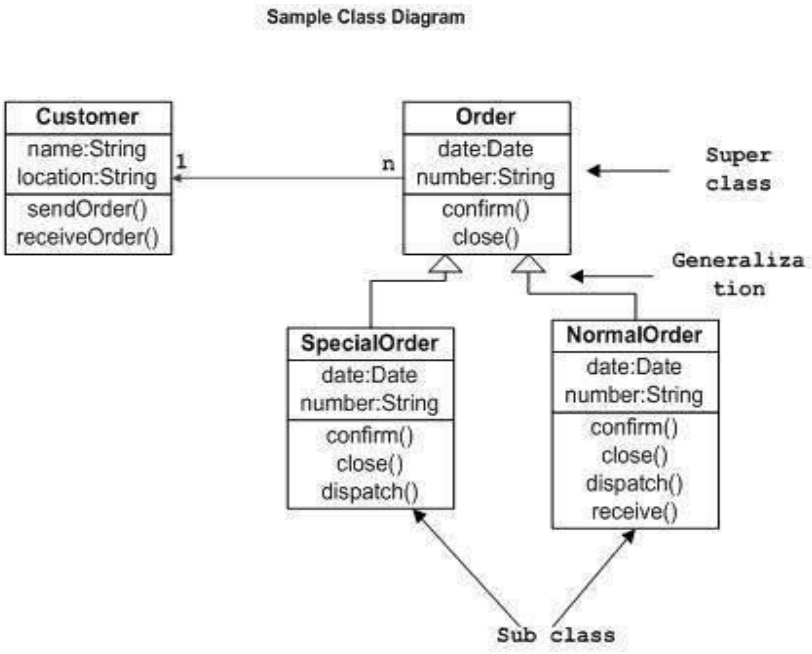
There are three important types of UML modelling:

1. Structural Modeling
    - ❖ Classes diagrams
    - ❖ Objects diagrams
    - ❖ Deployment diagrams
    - ❖ Component diagram
2. Behavioural Modeling
    - ❖ Activity diagrams
    - ❖ Interaction diagrams
    - ❖ Use case diagrams
3. Architectural Modeling
    - ❖ Package diagram

4. UML - Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.



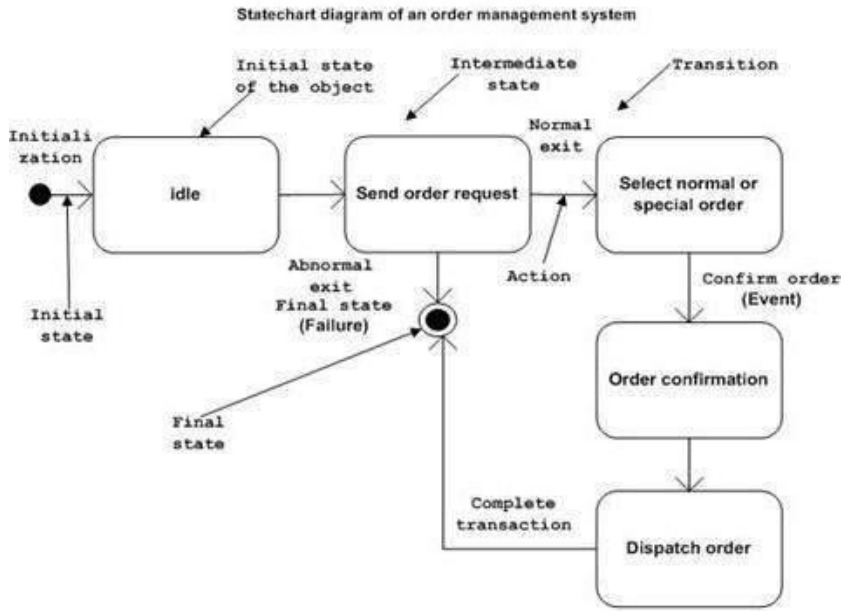
Class diagrams are used for −

- Describing the static view of the system.
- Showing the collaboration among the elements of the static view. Describing the functionalities performed by the system.
- Construction of software applications using object-oriented languages.

5. UML - State Diagrams

The name of the diagram itself clarifies the purpose of the diagram and other details. It describes different states of a component in a system. The states are specific to a component/object of a system.

A Statechart diagram describes a state machine. State machine can be defined as a machine which defines different states of an object, and these states are controlled by external or internal events. As Statechart diagram defines the states, it is used to model the lifetime of an object.



The main usage of State Diagram −

- To model the object states of a system. To model the reactive system. Reactive system consists of reactive objects.
- To identify the events responsible for state changes. Forward and reverse engineering.

6. UML - Use Case Diagrams

To model a system, the most important aspect is to capture the dynamic behaviour. Dynamic behaviour means the behaviour of the system when it is running/operating.

Only static behaviour is not sufficient to model a system rather dynamic behaviour is more important than static behaviour. In UML, there are five diagrams available to model the dynamic nature and use case diagram is one of them. Now as we must discuss that the use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction.

These internal and external agents are known as actors. Use case diagrams consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system.

Use case diagrams can be used for −

- Requirement analysis and high-level design.
- Model the context of a system. Reverse and Forward engineering.

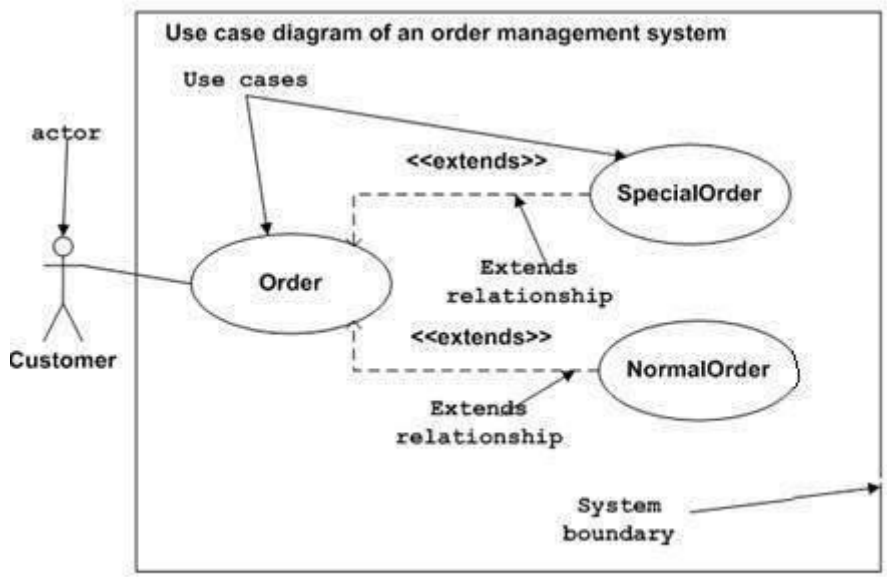
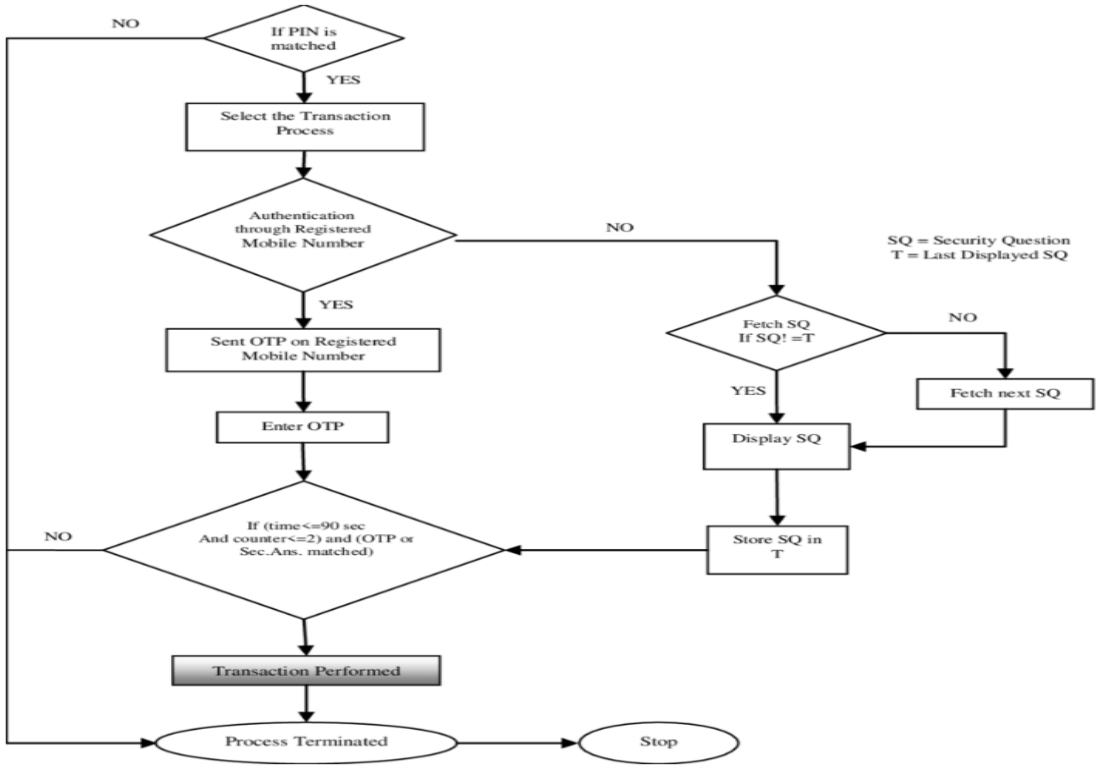


Figure: Sample Use Case diagram

7. UML- Flowchart

A flowchart is a diagram that depicts a process, system, or computer algorithm. They are widely used in multiple fields to document, study, plan, improve and communicate often complex processes in clear, easy-to-understand diagrams. Flowcharts, sometimes spelled as flow charts, use rectangles, ovals, diamonds, and potentially numerous other shapes to define the type of step, along with connecting arrows to define flow and sequence.

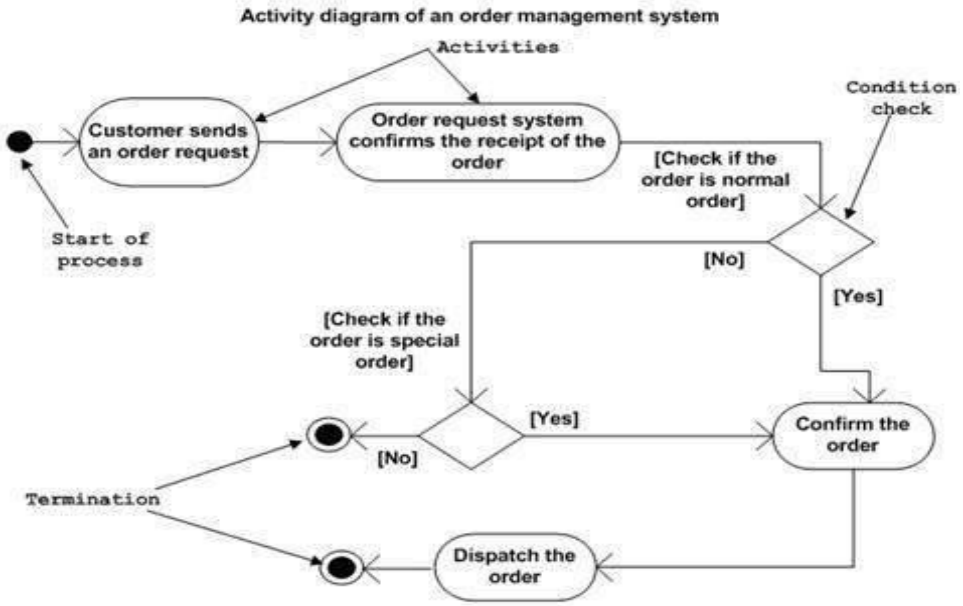


-Flowchart for ATM

8. UML - Activity Diagrams

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system. Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.

The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etc. The basic purposes of activity diagrams are like other four diagrams. It captures the dynamic behaviour of the system.



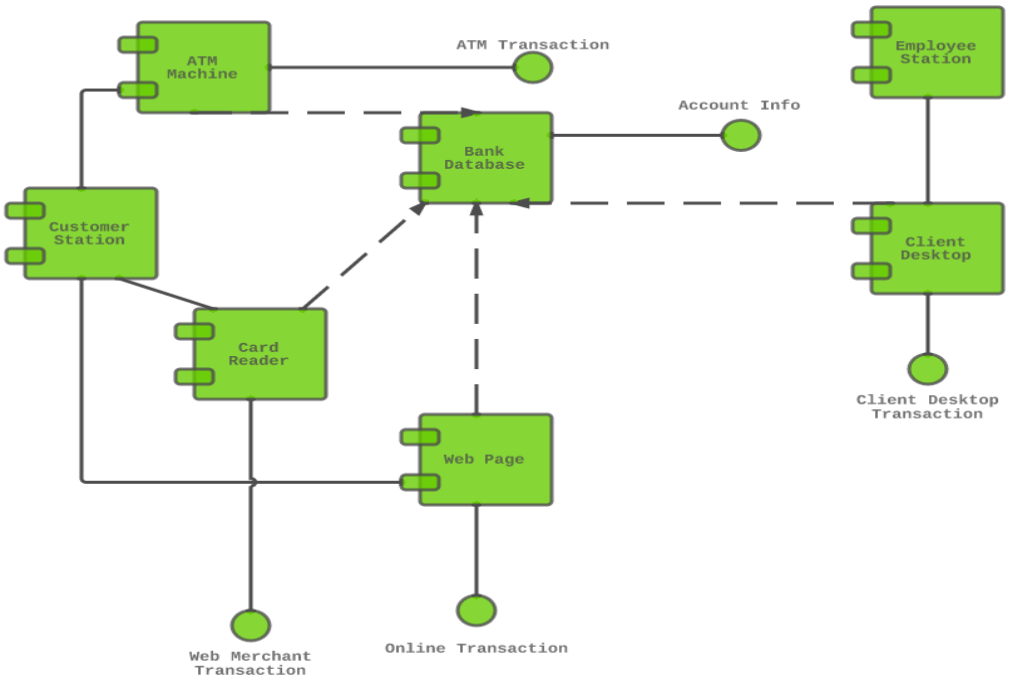
Activity diagram can be used for −

- Modeling workflow by using activities. Modeling business requirements.
- High level understanding of the system's functionalities. Investigating business requirements at a later stage.

9. UML - Component Diagrams

Component diagrams are different in terms of nature and behaviour. Component diagrams are used to model the physical aspects of a system. Now the question is, what are these physical aspects? Physical aspects are the elements such as executables, libraries, files, documents, etc. which reside in a node.

Component diagrams are used to visualize the organization and relationships among components in a system. These diagrams are also used to make executable systems. Component diagram is a special kind of diagram in UML. The purpose is also different from all other diagrams discussed so far. It does not describe the functionality of the system, but it describes the components used to make those functionalities.



-ATM Transaction Component Diagram

Component diagrams can be used to −

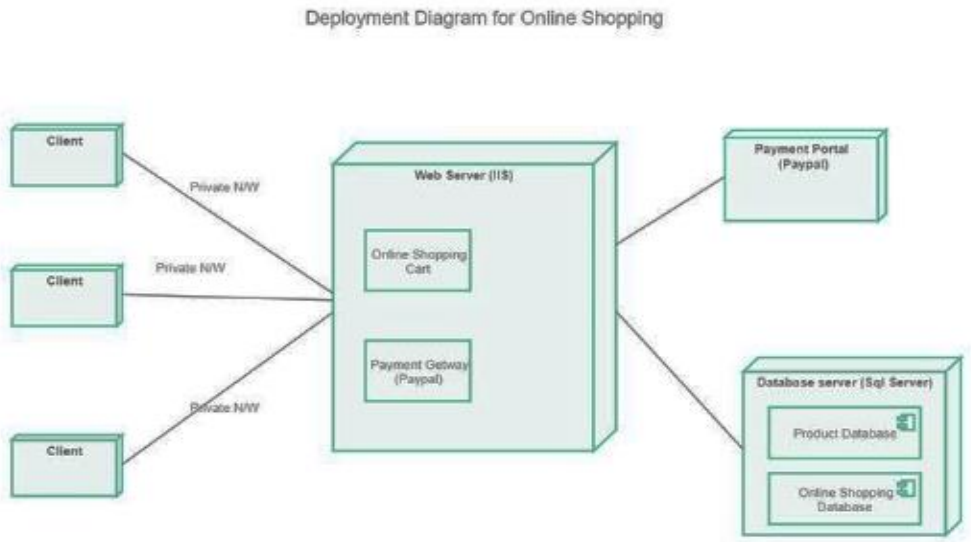
- Model the components of a system.
- Model the database schema.
- Model the executables of an application.
- Model the system's source code.

10. UML - Deployment Diagrams

Deployment diagrams are used to visualize the topology of the physical components of a system, where the software components are deployed. Deployment diagrams are used to describe the static deployment view of a system. Deployment diagrams consist of nodes and their relationships.

The term Deployment itself describes the purpose of the diagram. Deployment diagrams are used for describing the hardware components, where software components are deployed. Component diagrams and deployment diagrams are closely related. Component diagrams are used to describe the components and deployment diagrams shows how they are deployed in hardware.

UML is mainly designed to focus on the software artifacts of a system. However, these two diagrams are special diagrams used to focus on software and hardware components.

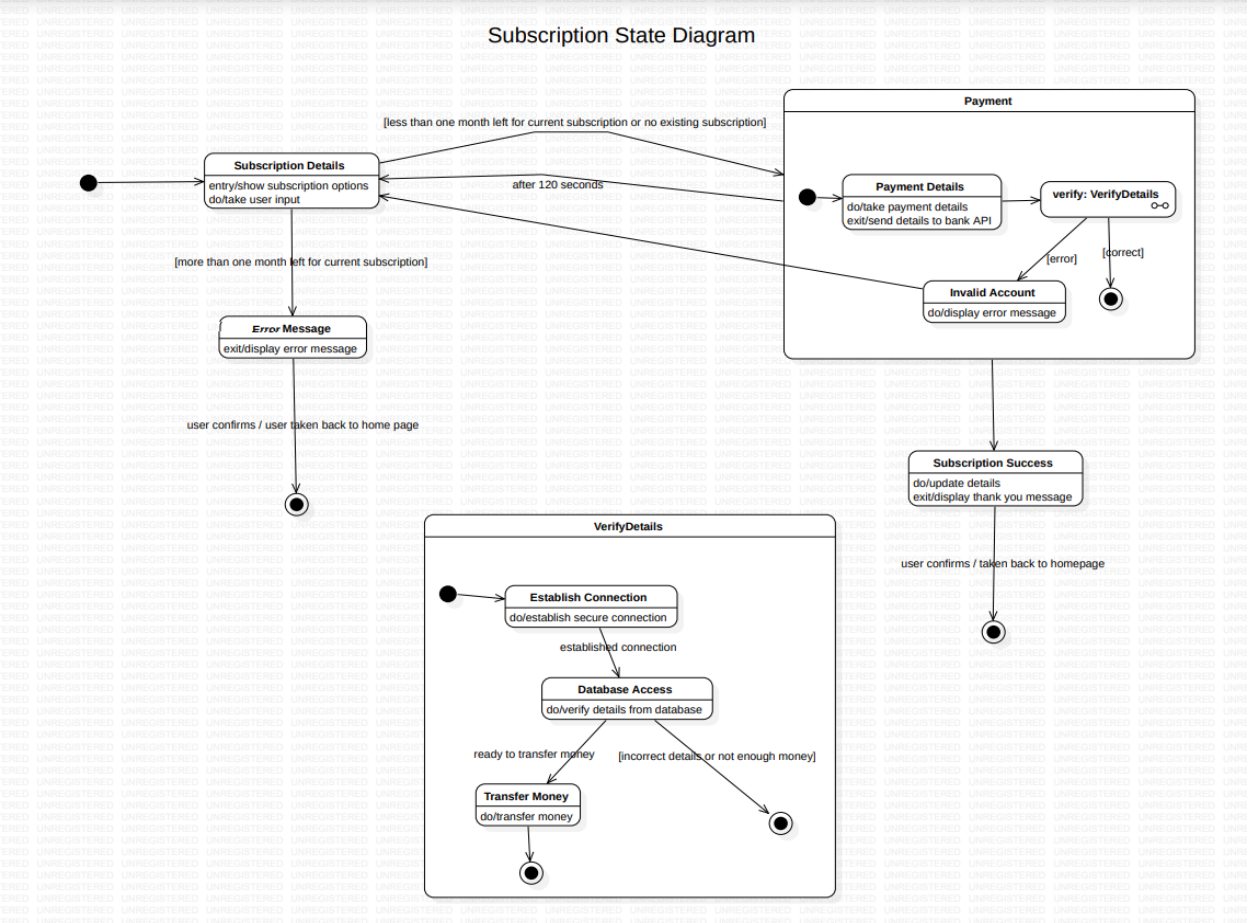


Deployment diagrams can be used −

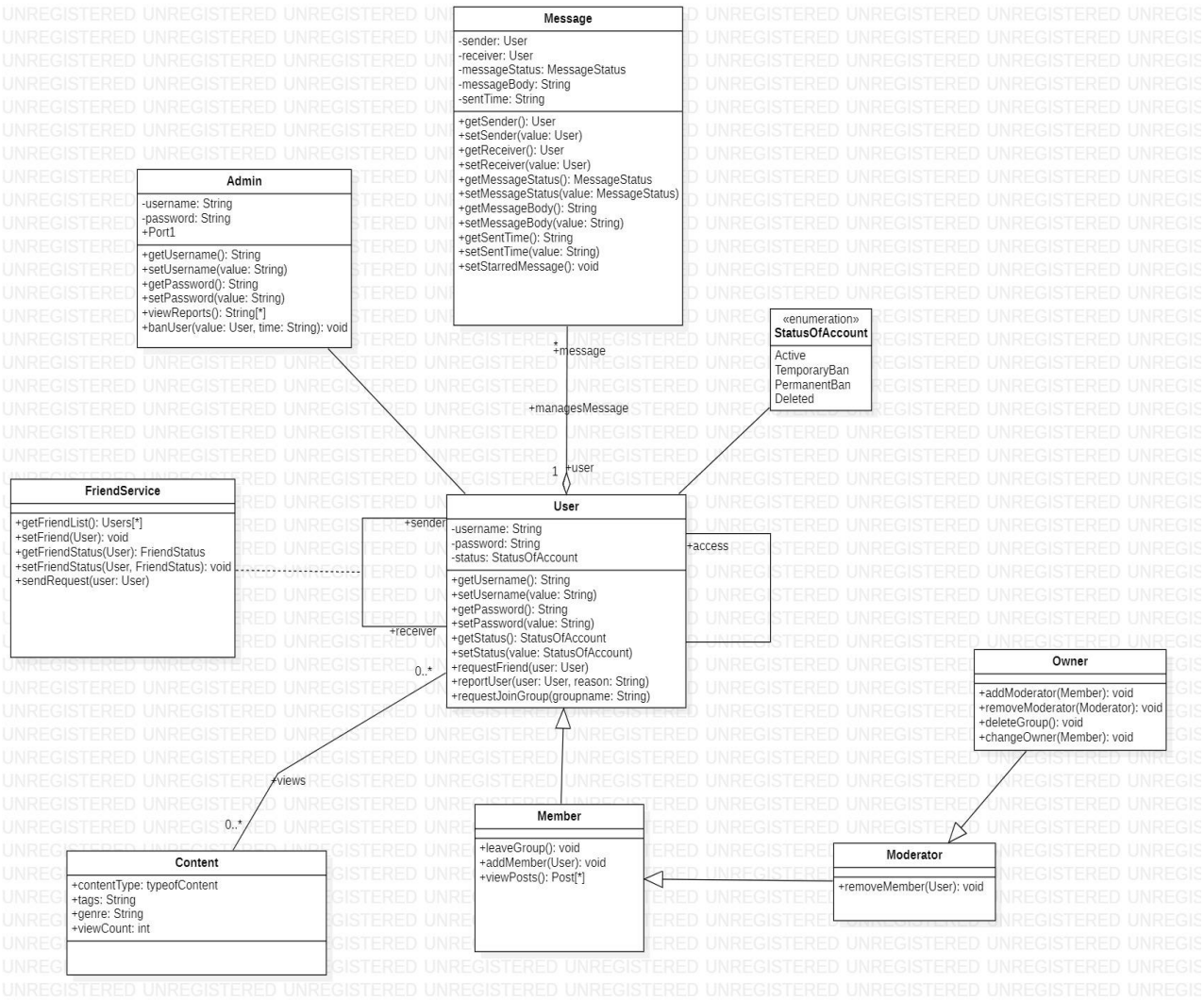
- To model the hardware topology of a system. To model the embedded system.
- To model the hardware details for a client/server system.
- To model the hardware details of a distributed application. Forward and Reverse engineering.

5. UML Diagrams

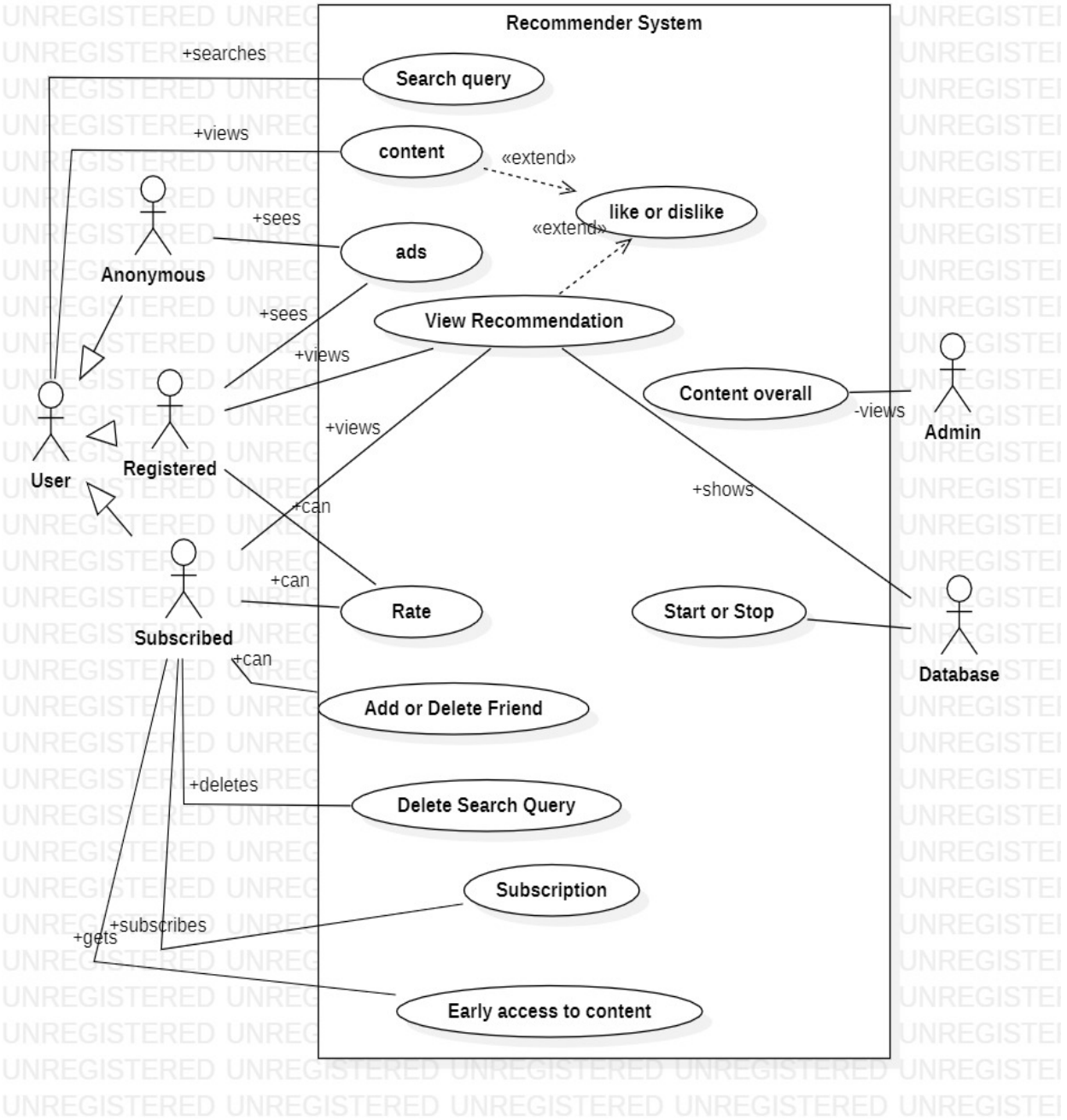
1. State Diagram:



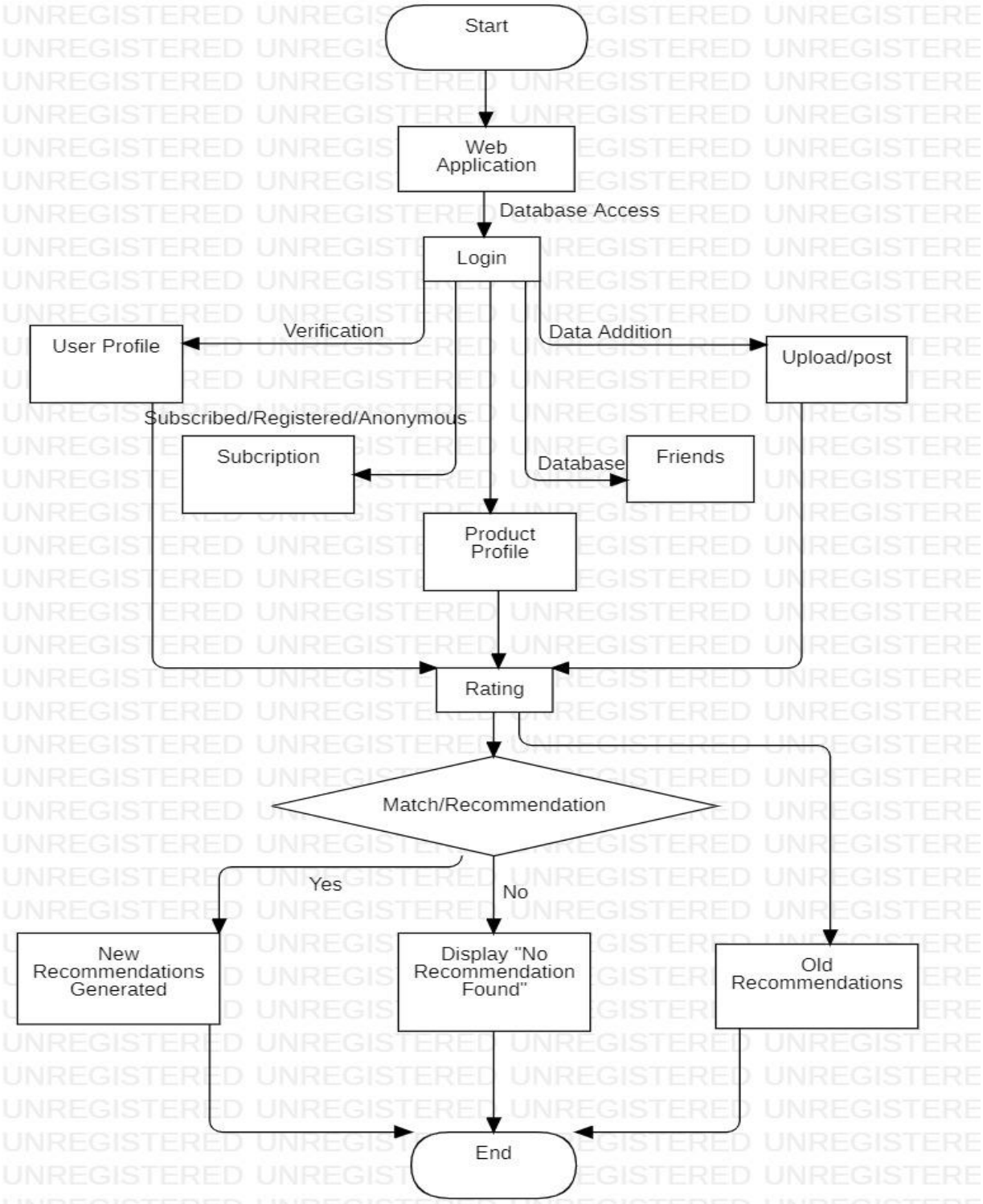
2. Class Diagram:



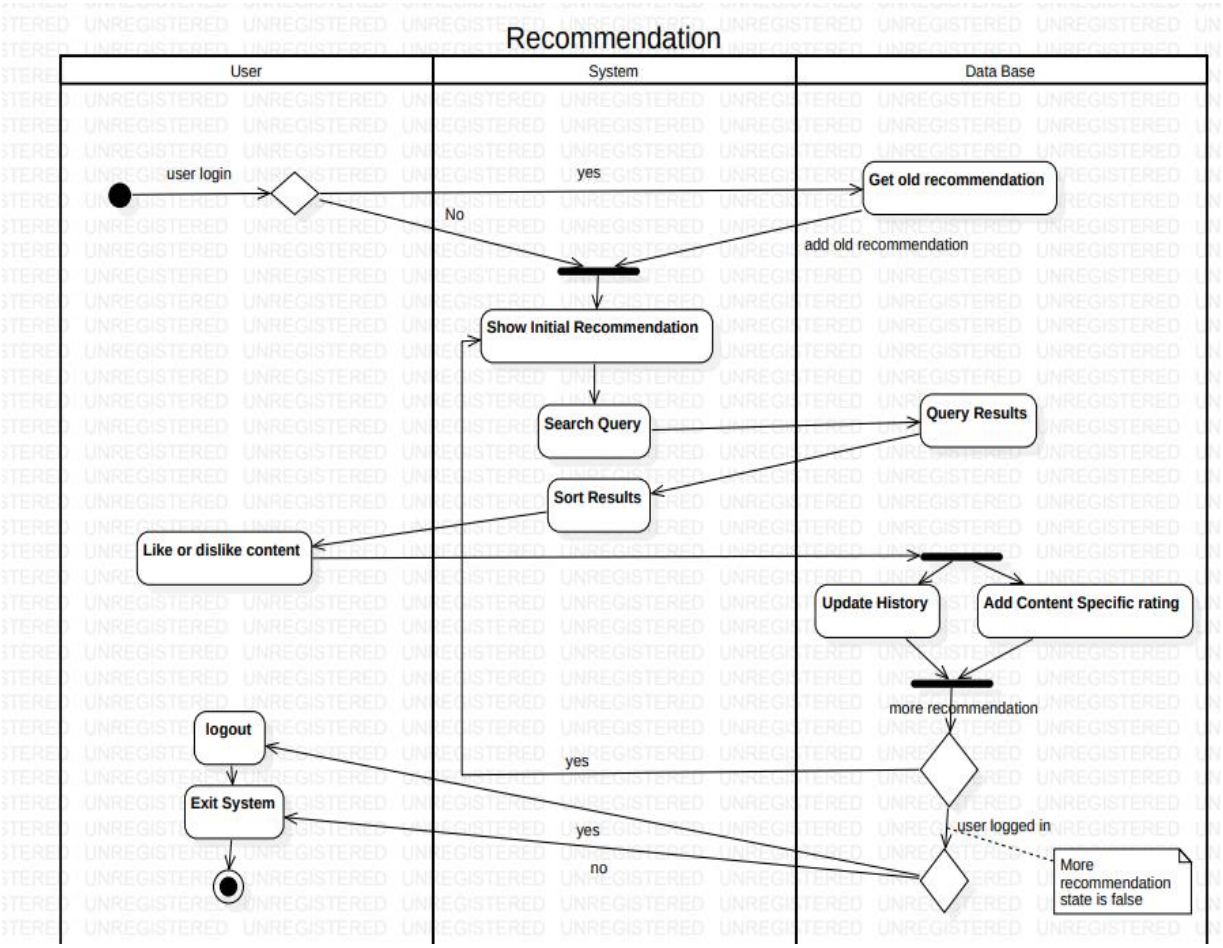
3. Use Case Diagram:



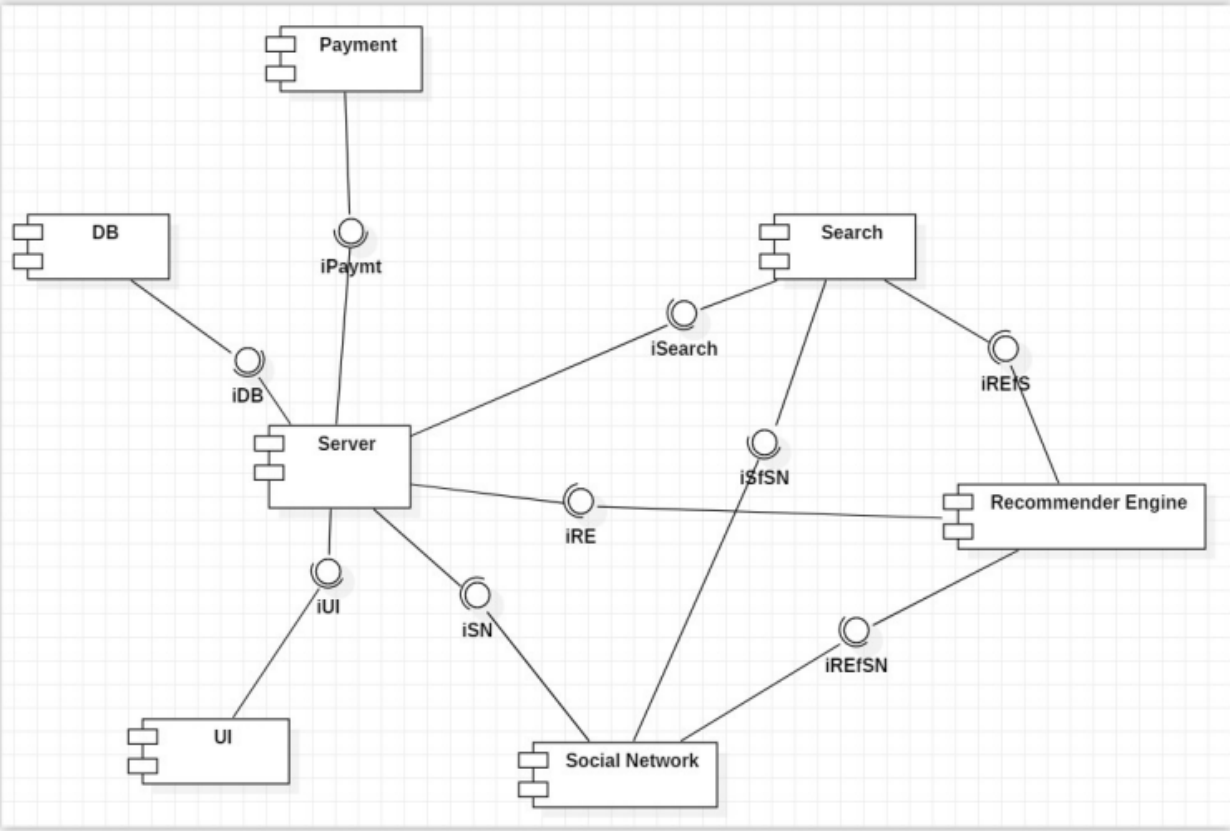
4. FlowChart Diagram:



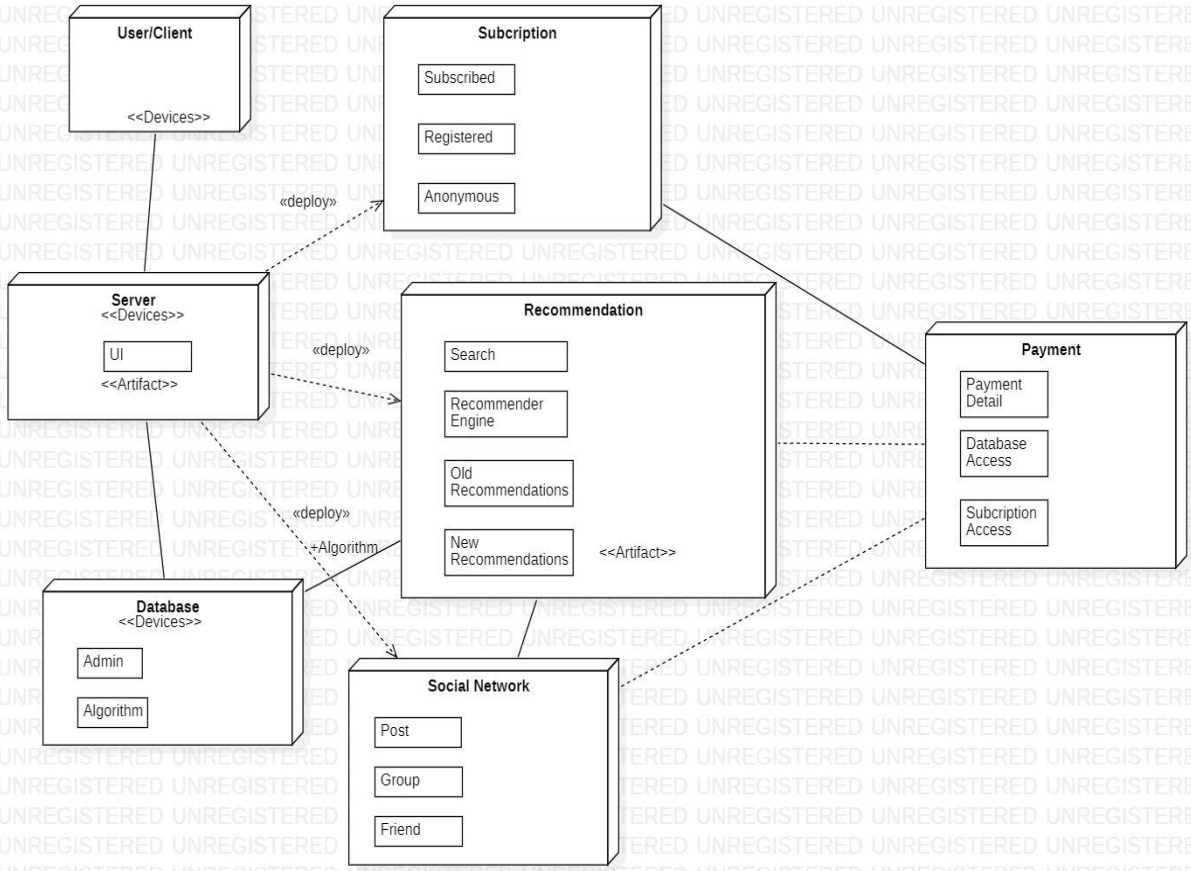
5. Activity Diagram:



6. Component Diagram:



7. Deployment Diagram:



6. Conclusion

UML class diagrams are useful when modeling business data. By accurately modeling attributes and associations of class entities, we can easily map these class diagram specifications to elements in Content Recommendation System. Class attributes map to abstract access methods for persistent fields, and association roles map to abstract access methods for relationship fields.

For system requirements to be easily understood by implementation groups, it’s important to employ UMLs since they simplify the system and make reusability and maintainability simpler. Hence we studies and implemented the concept of UML diagrams on project during its software development lifecycle , which help us to better understand and design optimized models for project.

7. References

[1] A Study of Importance of UML diagrams: With Special Reference to Very Large-sized Projects. March 2013. https://www.researchgate.net/publication/322991896_A_Study_of_Importance_of_UML_diagrams_With_Special_Reference_to_Very_Large-sized_Projects

[2] UML Diagrams in Software Engineering Research: A Systematic Literature Review † Hatice Koç *, Ali Mert Erdoğan, Yousef Barjakly and Serhat Peker

[3] UML – Overview https://www.tutorialspoint.com/uml/uml_overview.htm

[4] UML diagram synthesis techniques: a systematic mapping study. Authors: Damiano Torre, Yvan Labiche, Marcela Genero, Maria Teresa Baldassarre, Maged Elaasar.

[5] What are the used UML diagrams? A Preliminary Survey Gianna Reggio, Maurizio Leotta, Filippo Ricca, Diego Clerissi DIBRIS – Universit`a di Genova, Italy.

[6] A Study: UML for OOA and OOD D. Rajagopal, K. Thilakavalli

[7] Abrial, J.-R.: Modeling in Event-B: System and Software Engineering. Cambridge University Press, Cambridge (2010)