**OOSD ASSIGNMENT**

Report On Full OOA On Railway Ticket Reservation System

**Submitted By,**

|  |  |  |  |
| --- | --- | --- | --- |
| Roll no.: | Name | Sec | Branch |
| 1605087 | Bishal Tiwari | Cs-2 | CSE |
| 1605088 | Aadarsh Gupta | Cs-2 | CSE |
| 1605099 | Rahul Rauniyar | Cs-2 | CSE |
| 1605144 | Rajeev | Cs-2 | CSE |

**OOA Of Railway Ticket Reservation System**

Assume that domain analysis is complete and DAD is ready for reference. The analyst also has a fair knowledge of the system and the system environment. For the sake of convenience and to reduce complexity at this stage in OOA, we are ignoring certain conditions, constraints and features that the real system may have. For example, we are not considering cancellation of tickets as a requirement. We assume all passengers irrespective of their type (senior citizen, military personnel, special category passengers etc.) are the same, and therefore are charged with the same fare. We begin with the statement of requirement of the system.

**System Requirement**

* The passenger has a prior knowledge of the reservation and ticketing system. The passenger arrives at the railway ticket counter and interacts with the counter clerk first through an enquiry and then follows the process of form filling, tendering, payment and collecting the tickets.
* Passenger accepts the ticket or leaves the counter.
* Passenger seeks information on fare, train timings and availability of tickets.
* Passenger can have single ticket or multiple tickets.
* Journey begins on a day and will be over with one break in between.
* Passenger is identified by name, age, sex and address.
* Trains are identified by name and number.
* No receipt is issued for money transacted.
* Output of the system is ticket(s) with details.
* The process is triggered through a form filled by the passenger detailing the requirements of tickets, train, date, etc.
* A form is used for each train. If the journey requires use of multiple trains, separate forms are used for each train.

**Identify the Actors**

The actors in the system are the passenger, the counter clerk and the reservation system consisting of form processing, reservation, fare computation, ticket processing, ticket printing, collection of fare amount and posting as sub-systems. The passenger is a passive user–actor who initiates the process and obtains the ticket(s), a goal of measurable value. The counter clerk is an active user–actor, who triggers the system and has the role of issuing the tickets with the responsibility of collecting the correct fare amount from the passenger, which is a measurable value. Predesigned and deployed ticket reservation system at the back end is a system actor–user to ensure that ticket processing is done correctly and different system statuses are updated on issuing of tickets. This actor has an active role and responsibility at the back end.

**Develop Business Process Model – Reservation and Issue of Tickets**

Based on the system observation by the analyst, a high-level use-case diagram is drawn modelling the process of reservation and issue of tickets to the passenger. The use-case diagram brings everybody concerned with the system on the ground to a common understanding of the system as it functions.

**Functional Requirements:**

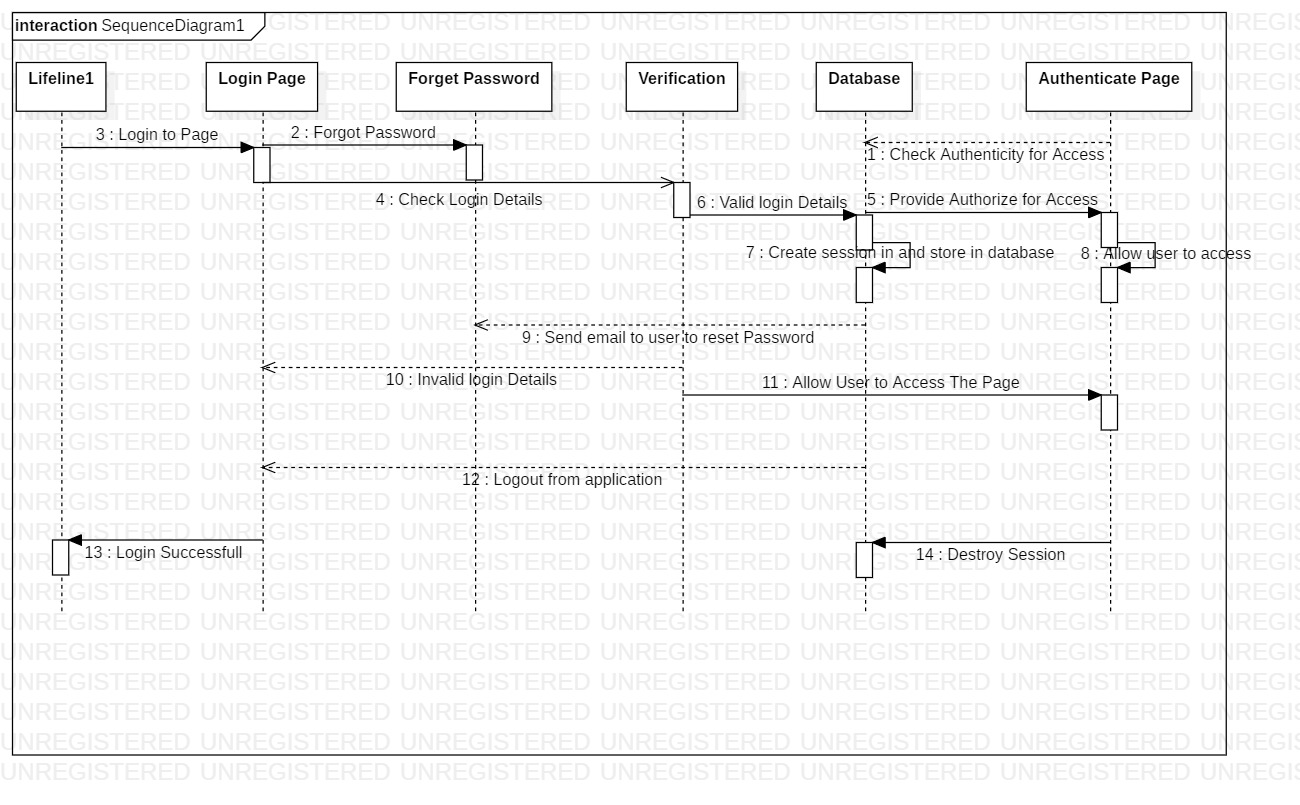
* Performance Requirements
* User Satisfaction
* Response Time
* Error Handling
* Safety and robustness
* Portable
* User Friendly
* Design Constrain
* Standard Compliance
* Hardware Limitations
* Reliability and Fault Tolerance
* Security
* Hardware Requirements
* Software Requirements
* Correctness
* Flexibility
* Reusability

**Non-Functional Requirements**:

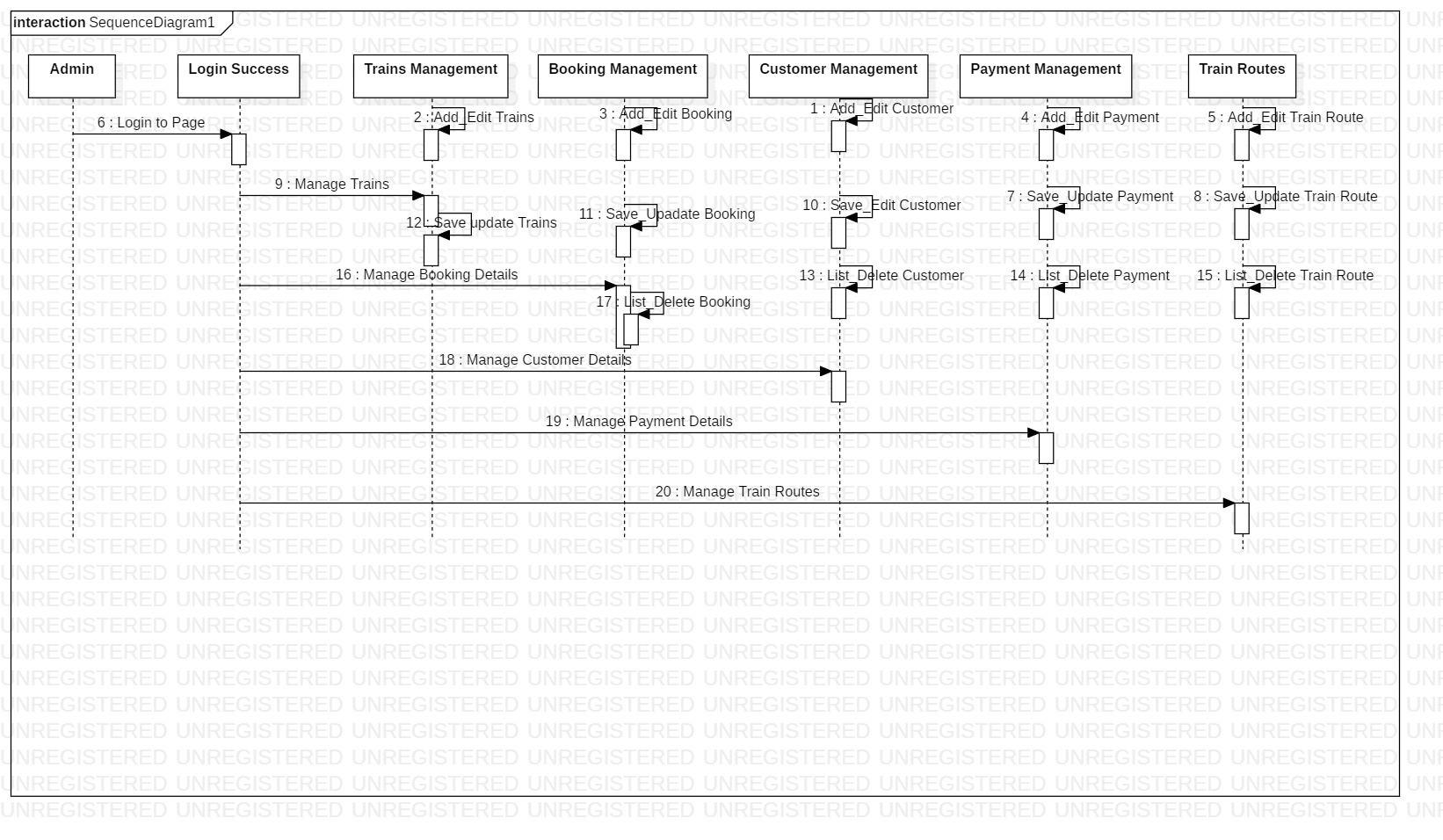
* Security
* Reliability
* Availability
* Maintainability
* Supportability

**UML DIGRAMS:**

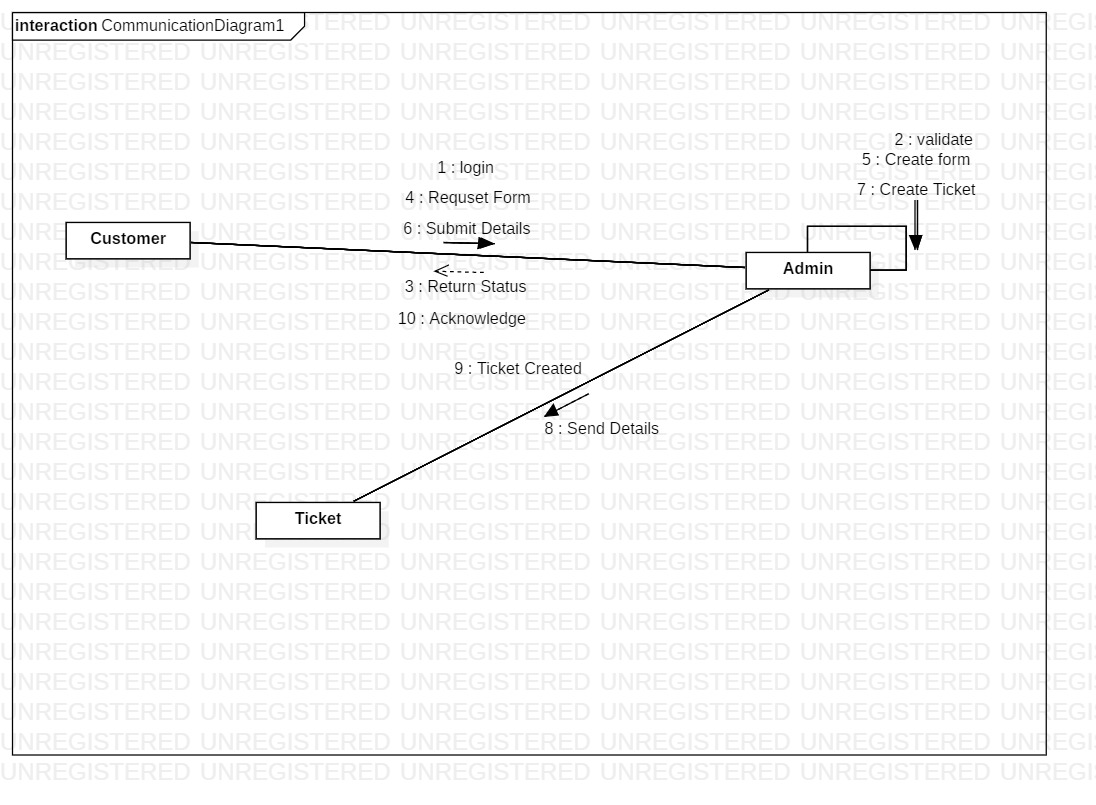
1. **USE-CASE DIAGRAM:**
2. **Sequence Diagram for Login:**



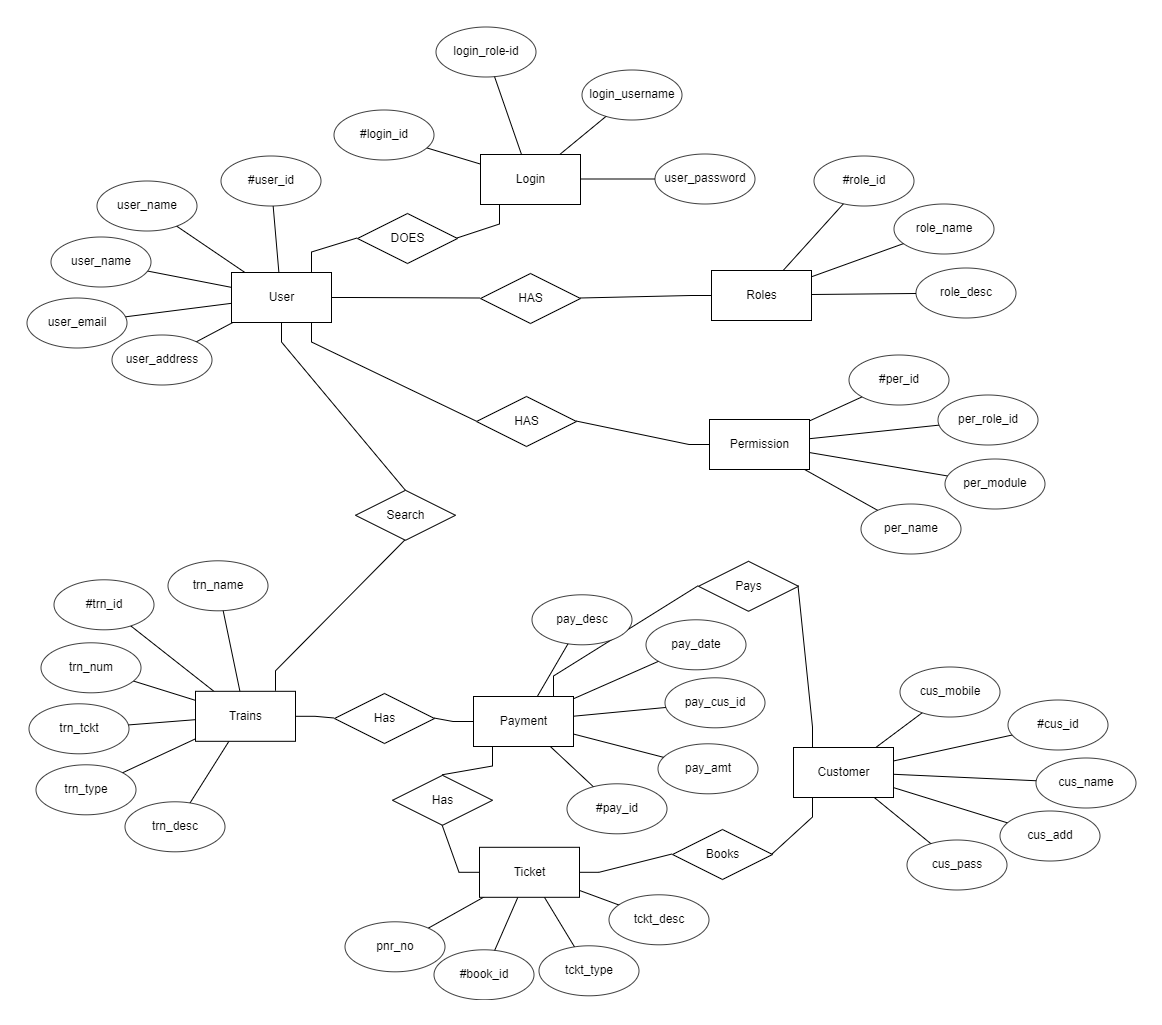
1. **Sequence Diagram for Railway Reservation System**

****

1. **Collaboration Diagram:**

****

1. **Class Diagram:**
2. **Activity Diagram:**
3. **ER Diagram:**

****

**Conclusion:**

We here by conclude, after completion of this assignment that OOSD is the most proper, easy and efficient way for software development. It is fast and provides users easier view of the problem through the diagrams. It also helps developers to manage the product’s integrity and make changes whenever needed.

**Reference:**

* [www.google.com](http://www.google.com)
* [www.youtube.com](http://www.youtube.com)
* [www.irctc.com](http://www.irctc.com)
* [www.scribd.com](http://www.scribd.com)
* [www.wikipedia.com](http://www.wikipedia.com)