**M S Ramaiah Institute of Technology**

(An Autonomous Institute, Affiliated to VTU) MSR nagar, MSRIT post, Bangalore-54

A Dissertation Report on

**Real time Response Capture and Analytics of Intelligent Tutoring System**

# *Bachelor of Engineering in Computer Science & Engineering*



**DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING**

**Under the guidance of:**

**Asst Prof Ganeshayya I Shidaganti**

**Submitted by:**

Namratha Ravishankar 1MS11CS067

Ashwij S 1MS11CS152

Nuthana Kumari M 1MS12CS070

Aishwarya S S 1MS12CS142

**1.0 project overview**

* Our main objective is to provide the rating parameters such as Syllabus Coverage, Usage of Examples, Understandability and Practical Knowledge to learner for better response collection. The learner can rate the performance of educator’s video session any number of times within the time limit.
* Total performance of the session will be analyzed with the help of ratings from all learners. This analysis will be given to educators to improve the performance in next sessions.
* Helps to improve learning, instruction and student satisfaction.

**2.0 External Interface Requirements**

**2.1 User Interfaces**

* All the contents in the project are implemented using Graphical User Interface (GUI) in Java through JavaFX concepts.
* Every conceptual part of the projects is reflected using the JavaFX.
* System gets the input and delivers through the GUI

**2.2 Hardware Interfaces**

**ISDN:** We can connect AS/400 to an Integrated Services Digital Network (ISDN) for faster, more accurate data transmission. An ISDN is a public or private digital communications network that can support data, fax, image, and other services over the same physical interface. Also, you can use other protocols on ISDN, such as IDLC and X.25.

**2.3 Software Interfaces**

This software is interacted with the TCP/IP protocol, Socket and listening on unused ports. Server Socket and listening on unused ports and JDK 1.6. This software is also interacted with the SMTP protocol, sending and receiving on SMTP protocol.

**2.4 Communication Interfaces**

* TCP/IP protocol.
* SMTP.

**3.0 Functional Requirements**

An intelligent tutoring system is computer software designed to**simulate a human tutor’s behavior** and guidance. Because these systems are able to **interpret complex student responses** and can learn as they operate, they are able to discern where and why a student’s understanding has gone astray and to **offer hints to help the student understand the material** at hand. Intelligent tutors provide many of the benefits of a human tutor to very large numbers of students. Intelligent tutoring systems can also provide real-time data to instructors and developers looking to refine teaching methods.

**User Classes and Characteristics**

* + - Student: when student views the video, gives comment and rating of that particular video
    - tutor: goes through the video analysis rating and comments

**4.0 Software System Attributes**

**4.1 Reliability**

Reliability ensures the integrity and consistency of the application and all its transactions. As the load increases on your system, your system must continue to process requests and handle transactions as accurately as it did before the load increased. Reliability can have a negative impact on scalability. If the system cannot maintain the reliability as the load increases, then the system is really not scalable. So, for a system to truly scale it must be reliable.

**4.2 Availability**

Availability ensures that a service/resource is always accessible. Reliability can contribute to availability, but availability can be achieved even if components fail. By setting up an environment of redundant components and failover, an individual component can fail and have a negative impact on reliability, but the service is still available due to the redundancy.

**4.3 Security**

Security is the ability to ensure that the system cannot be compromised. Security is by far the most difficult systemic quality to address. Security includes not only issues of confidentiality and integrity, but also relates to Denial-of-Service (DoS) attacks that impact availability. Creating an architecture that is separated into functional components makes it easier to secure the system because you can build security zones around the components. If a component is compromised, then it is easier to contain the security violation to that component.

**4.4 Portability**

This software will be designed to run on windows OS and forward compatibility

**4.5 Maintainability**

Maintainability is the ability to correct flaws in the existing functionality without impacting other components of the system. This is another of those systemic qualities that you cannot measure at the time of deployment. When creating an architecture and design, you should consider the following to enhance the maintainability of a system: low coupling, modularity, and documentation.

**4.6 Performance**

The performance requirement is usually measured in terms of response time for a given screen transaction per user. In addition to response time, performance can also be measured in transaction throughput, which is the number of transactions in a given time period, usually one second. For example, you could have a performance measurement that could be no more than three seconds for each screen form or a transaction throughput of one hundred transactions in one second. Regardless of the measurement, you need to create an architecture that allows the designers and developers to complete the system without considering the performance measurement.

**5.0 Performance Requirements**

In WMNs, network-layer handoff signalling packets, including the signalling messages for obtaining a new IP address for the MC (e.g., Agent Solicitation, Agent Advertisement), finding a new route to the new gateway, and updating the new IP address, may be transmitted over the multi-hop wireless mesh backbone with data packets on the same backhaul channel in traditional designs. In this scenario, signalling packets compete with data packets for the same wireless resources. Therefore, the more data packets are generated in the mesh backbone, the more possible collisions may occur between the two types of packets, which results in long handoff delay. In addition, the contention of the two types of packets on the same channel increases the end to end (ETE) delay of both packets.

**6.0 Database Requirement**

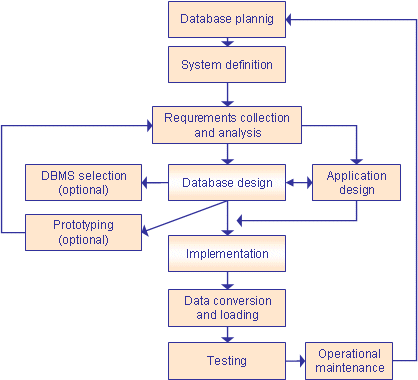
**Oracle**

An Oracle **database** is a collection of data treated as a unit. The purpose of a database is to store and retrieve related information. A database server is the key to solving the problems of information management. In general, a [server](https://docs.oracle.com/cd/B19306_01/server.102/b14220/glossary.htm#i432724) reliably manages a large amount of data in a multiuser environment so that many users can concurrently access the same data. All this is accomplished while delivering high performance. A database server also prevents unauthorized access and provides efficient solutions for failure recovery.

Oracle Database is the first database designed for enterprise grid computing, the most flexible and cost effective way to manage information and applications. Enterprise grid computing creates large pools of industry-standard, modular storage and servers. With this architecture, each new system can be rapidly provisioned from the pool of components. There is no need for peak workloads, because capacity can be easily added or reallocated from the resource pools as needed.

The database has **logical structures** and **physical structures**. Because the physical and logical structures are separate, the physical storage of data can be managed without affecting the access to logical storage structures

**DATABASE DESIGN**

A database is usually a fundamental component of the information system, especially in business oriented systems. Thus database design is part of system development. The following picture shows how database design is involved in the system development lifecycle.  
  
The phases in the middle of the picture (Database Design, Database Implementation) are the phases that you concentrate on in the Database Design course. The other phases are briefly described. They are part of the contents of the Systems Analysis and Design courses, for example.  
  


### Database Planning

The database planning includes the activities that allow the stages of the database system development lifecycle to be realized as efficiently and effectively as possible. This phase must be integrated with the overall Information System strategy of the organization.  
  
The very first step in database planning is to define the mission statement and objectives for the database system. That is the definition of  
- the major aims of the database system  
- the purpose of the database system  
- the supported tasks of the database system  
- the resources of the database system

### Systems Definition

In the systems definition phase, the scope and boundaries of the database application are described. This description includes:  
- links with the other information systems of the organization  
- what the planned system is going to do now and in the future  
- who the users are now and in the future.  
  
The major user views are also described. i.e., what is required of a database system from the perspectives of particular job roles or enterprise application areas

### Requirements Collection and Analysis

During the requirements collection and analysis phase, the collection and analysis of the information about the part of the enterprise to be served by the database are completed. The results may include eg:  
- the description of the data used or generated  
- the details how the data is to be used or generated  
- any additional requirements for the new database system

### Database Design

The database design phase is divided into three steps:  
- conceptual database design  
- logical database design  
- physical database design  
  
In the conceptual database design phase, the model of the data to be used independent of all physical considerations is to be constructed. The model is based on the requirements specification of the system.  
  
In the logical database design phase, the model of the data to be used is based on a specific data model, but independent of a particular database management system is constructed. This is based on the target data model for the database e.g. relational data model.  
  
In the physical database design phase, the description of the implementation of the database on secondary storage is created. The base relations, indexes, integrity constraints, security, etc. are defined using the SQL language.

### Database Management System Selection

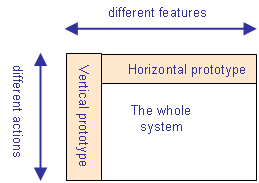
This in an optional phase. When there is a need for a new database management system (DBMS), this phase is done. DBMS means a database system like Access, SQL Server, MySQL, Oracle.  
  
In this phase the criteria for the new DBMS are defined. Then several products are evaluated according to the criteria. Finally the recommendation for the selection is decided.

### Application Design

In the application design phase, the design of the user interface and the application programs that use and process the database are defined and designed.

### Protyping

The purpose of a prototype is to allow the users to use the prototype to identify the features of the system using the computer.  
  
There are horizontal and vertical prototypes. A horizontal prototype has many features (e.g. user interfaces) but they are not working. A vertical prototype has very few features but they are working. See the following picture.



### Implementation

During the implementation phase, the physical realization of the database and application designs are to be done. This is the programming phase of the systems development.

### Data Conversion and Loading

This phase is needed when a new database is replacing an old system. During this phase the existing data will be transferred into the new database.

### Testing

Before the new system is going live, it should be thoroughly tested. The goal of testing is to find errors! The goal is not to prove the software is working well.

### Operational Maintenance

The operational maintenance is the process of monitoring and maintaining the database system.  
Monitoring means that the performance of the system is observed. If the performance of the system falls below an acceptable level, tuning or reorganization of the database may be required.   
Maintaining and upgrading the database system means that, when new requirements arise, the new development lifecycle will be done.

**7.0 Design and Implementation Constraints**

**7.1 Constraints in Analysis**

* Constraints as Informal Text
* Constraints as Operational Restrictions
* Constraints Integrated in Existing Model Concepts
* Constraints as a Separate Concept
* Constraints Implied by the Model Structure

**7.2 Constraints in Design**

* Determination of the Involved Classes
* Determination of the Involved Objects
* Determination of the Involved Actions
* Determination of the Require Clauses
* Global actions and Constraint Realization

**7.3 Constraints in Implementation**

A hierarchical structuring of relations may result in more classes and a more complicated structure to implement. Therefore it is advisable to transform the hierarchical relation structure to a simpler structure such as a classical flat one. It is rather straightforward to transform the developed hierarchical model into a bipartite, flat model, consisting of classes on the one hand and flat relations on the other. Flat relations are preferred at the design level for reasons of simplicity and implementation ease. There is no identity or functionality associated with a flat relation. A flat relation corresponds with the relation concept of entity-relationship modeling and many object oriented methods.

**8.0 Other Requirements**

**8.1 Safety Requirements**

* The software may be safety-critical. If so, there are issues associated with its integrity level
* The software may not be safety-critical, although it forms part of a safety-critical system. For example, software may simply log transactions.
* If a system must be of a high integrity level and if the software is shown to be of that integrity level, then the hardware must be at least of the same integrity level.
* There is little point in producing 'perfect' code in some language if hardware and system software (in widest sense) are not reliable.
* If a computer system is to run software of a high integrity level then that system should not at the same time accommodate software of a lower integrity level.
* Systems with different requirements for safety levels must be separated. Otherwise, the highest level of integrity required must be applied to all systems in the same environment.