A

PROJECT REPORT

ON

HYDRO YOUR SOURCE FOR WATER INFORMATION

Submitted To

Osmania University

In partial fulfillment of the requirements for the award of the degree of

MASTER OF COMPUTER APPLICATIONS

BY

Mr. K SAI KIRAN REDDY

(1304-22-862-152)

AURORA'S PG COLLEGE – (MCA)

(Affiliated to Osmania University)

Peerzadiguda, Uppal, Hyderabad. - 500039

(2022 -2024)

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompanies the successful completion of any task would be incomplete without the mention of the people who made it possible and whose encouragement and guidance have crowned out efforts with success.

I am extremely thankful to the Principal, **AURORA'S PG COLLEGE (MCA), UPPAL**, for providing me with infrastructural facilities to work in, without which this work would not have been possible.

I would like to express my gratitude to the Head, **Department of Computer Applications, Aurora's PG College (MCA),** Uppal, for his stimulating guidance, continuous encouragement, and supervision during the course of the present work.

I would like a place on record my deep sense of gratitude to Associate professor, Aurora's PG College (MCA), Uppal, for his continuous guidance and valuable suggestions in completing the project right from its conception to conclusions.

Above all I express thanks to the faculty, my parents, and my friends without whose support, I could not have made this project.

K SAI KIRAN REDDY

1304-22-862-152

DECLARATION

I hereby declare that the work presented in this project report titled "**HYDRO YOUR SOURCE FOR**

WATER INFORMATION" is done by me in Master of Computer Applications Aurora's PG College (MCA), Uppal. No part of the dissertation is copied from books/Journals/internet and whenever the portion is taken, this has been duly referred in the text. The report is based on the project work done entire by me not copied from any other source.

K SAI KIRAN REDDY

1304-22-862-152

ABSTRACT

Access to safe and clean drinking water is a fundamental human right, and its availability and quality are of paramount importance. To ensure the efficient management of drinking water resources, a comprehensive and user-friendly information system is essential. This abstract outline the development of a web portal for a drinking water details system (dwds) aimed at providing real-time access to critical information related to drinking water sources, quality, distribution, and consumption.

The development of the DWDS web portal represents a significant step towards ensuring the sustainable management of drinking water resources. It empowers stakeholders with the tools and information needed to make informed decisions,enhances water quality monitoring, and promotes efficient water resource management practices. This system contributes to the broader goal of ensuring safe and reliable access to clean drinking water for all, thereby improving the well-being and health of communities.

Hydro Hub emerges as a pioneering online platform dedicated to providing a comprehensive and accessible source of water-related information. In an era characterized by growing concerns over water scarcity, quality, and sustainable management, Hydro Hub stands as a crucial resource for individuals, communities, policymakers, and businesses seeking reliable data and insights on all aspects of water. HydroHub's mission is to empower individuals and organizations with the knowledge and tools needed to make informed decisions about water resources. By promoting data- driven, sustainable water management practices and fostering a sense of shared responsibility, hydro Hub serves as a catalyst for positive change in how we approach one of our most precious and vital resources. In a world where access to clean and reliable water is increasingly essential, hydro Hub stands as a beacon of knowledge and collaboration in the pursuit of water sustainblity.

INDEX

List of Contents	Page No
1.INTRODUCTION	
1.1 Overview	
1.2 Company Profile	
2. LITERATURE OVERVIEW	
2.1 The Software Development Life Cycle (SDLC)	
2.1.1 Requirement Analysis And Design	
2.1.2 Implementation	
2.1.3 Testing	
2.1.4 Maintenance	
2.2 Software Developing Process Model	
2.3 Architecture	
2.4 Object Oriented System Development	
2.4.1 Data Flow Diagram	
2.4.2 Class Diagram	
2.4.3 Use – Case Diagram	
2.4.4 Sequence Diagram	
2.4.5 Activity Diagram	
2.5 Data Dictionary	
2.6 ER - Diagram	
2.7 Technology	
2.8 Content Reference To Project	
3. ANALYSIS	
3.1 System Requirement Specification	
3.1.1 Introduction	
3.1.2 Existing system	
3.1.3 Proposed system	
3.1.4 Scope	
3.2 Software / Hardware Requirements	
3.2.1 Software Requirements	
3.2.2 Hardware Requirements	
3.3 Modules	
3.3.1 Attribute Authority (AA)	
3.3.2 Data Owner (DO)	

3.3.3 Cloud Server (CS)	
3.3.4 Data Consumer (DC)	
3.4 Functional Requirements	
3.5 Feasibility Study	
3.5.1 Economical Feasibility	
3.5.2 Technical Feasibility	
3.5.3 Operational Feasibility	
4. Design	
4.1 Architecture	
4.2 System Design	
4.2.1 Use case Diagram	
4.2.2 Class Diagram	
4.2.3 Activity Diagram	
4.2.4 Sequence Diagram	
4.2.5 Collaboration Diagram	
4.2.6 Component Diagram	
4.2.7 E.R. Diagram	
4.2.8 Data Dictionary	
5. System Implementation	
5.1 Organization Impact	
5.2 User Manger Assessment	
5.3 Code	
5.4 Screen Shots	
6. Testing	
6.1 Software Testing	
6.2 Testing Approaches	
6.3 Testing methods	
6.4 Testing Types	
6.5 Test Cases	
7. Conclusion	
8. Future Enhancement	
9. Bibliography	
10. Gantt Chart	

1 INTRODUCTION

Hydro is a comprehensive platform that provides a wealth of data and resources related to water. Hydro offers a wide range of information, including water quality reports, conservation tips, consumption data, infrastructure updates, and much more. Whether you're a part of the public, municipal authorities, or administrators, Hydro can be a valuable tool for accessing essential water- related information to make informed decisions and promote water sustainability.

It provides a ton of valuable data and resources to help everyone from the public to municipal authorities make informed decisions about water.

Hydro can give you insights into water quality reports, conservation tips to save water, data on how much water is being used, updates on water infrastructure, and even emergency response protocols. It's a great tool to understand water supply and demand, improve infrastructure, and respond quickly to any water-related issues in your community.

1.2 Company profile

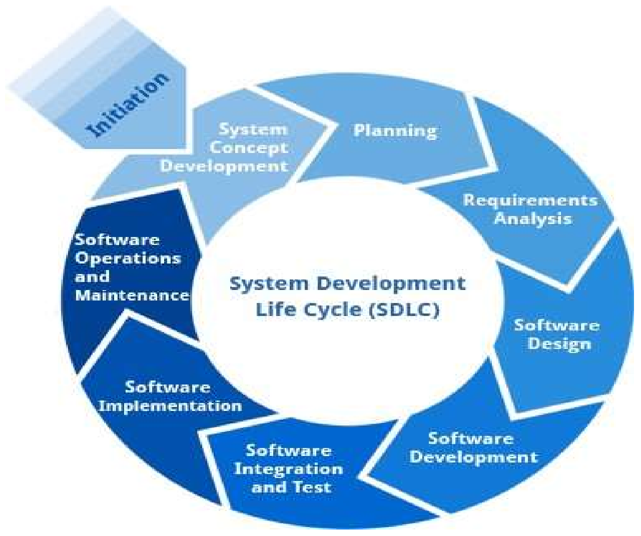
An IIM Alumnus Company started in 1998, as a training partnerof CMC LTD, located at Door No: 105, 1st Floor, Above Airtel Office, Near Metro Pillar no: A-1515, Hyderabad, "PRIME TECHNOLOGIES" brings transformational high-end and deep-tech learning programs to emergingand experienced professionals in partnership with top academic institutions and global corporations. Our vision is to be a globally recognized centre of excellence in education and training. We aspire to transform lives by equipping our students with the knowledge and skills needed to excel in their chosen fields. We envision a future where our graduates drive positive change, contribute to the advancement of society, and lead with integrity.

Our mission is to empower individuals with knowledge, skills, and confidence to thrive in a dynamic world. We are dedicated to providing high-quality education and training that fosters personal growth, professional development, and lifelong learning. Through innovative teaching methods, industry-relevant curriculum, and a commitment to excellence, we aim to nurture the leaders, innovators, and problem solvers of tomorrow. In the training arena PRIME TECHNOLOGIES has an unenviable track record of grooming several hundreds of students, building their skills and launching their careers. Alumni of PRIME TECHNOLOGIES are in senior IT positions across the globe. PRIME TECHNOLOGIES also offers a host of training programs tailormade for IT companies.

2 . OVERVIEW LITERATURE

The Systems Development Life Cycle (SDLC):

The Systems Development Life Cycle (SDLC), or Software Development Life Cycle in systems engineering, information systems and software engineering, is the process of creating or altering systems, and the models and methodologies use to develop these systems.


Initiation
System Concept Development
Planning
Requirements Analysis
Software Design
Software Development
Software Integration and Test
Software Implementation
Software Operations and Maintenance
System Development Life Cycle (SDLC)

2.1.1 Requirement Analysis and Design

Analysis gathers the requirements for the system. This stage includes a detailed study of the business needs of the organization. Options for changing the business process may be considered. Design focuses on high level design like, what programs are needed and how are they going to interact, low-level design (how the individual programs are going to work), interface design (what are the interfaces going to look like) and data design (what data will be required). During these phases, the software's overall structure is defined. Analysis and Design are very crucial in the whole development cycle. Any glitch in the design phase could be very expensive to solve in the later stage of the software development. Much care is taken during this phase. The logical system of the product is developed in this phase.

2.1.2 Implementation

In this phase the designs are translated into code. Computer programs are written using a conventional programming language or an application generator. Programming tools like Compilers, Interpreters, and Debuggers are used to generate the code. Different high level programming languages like PYTHON 3.6, Anaconda Cloud are used for coding. With respect to the type of application, the right programming language is chosen.

2.1.3 Testing

In this phase the system is tested. Normally programs are written as a series of individual modules, this subject to separate and detailed test. The system is then tested as a whole. The separate modules are brought together and tested as a complete system. The system is tested to ensure that interfaces between modules work (integration testing), the system works on the intended platform and with the expected volume of data (volume testing) and that the system does what the user requires (acceptance/beta testing).

2.1.4 Maintenance

Inevitably the system will need maintenance. Software will definitely undergo change once it is delivered to the customer. There are many reasons for the change. Change could happen because of some unexpected input values into the system. In addition, the changes in the system could directly affect the software operations. The software should be developed to accommodate changes that could happen during the post implementation period.

Software Developing Process Model Waterfall Model:

The Waterfall Model was the first Process Model to be introduced. It is also referred to as a linear-sequential life cycle model. It is very simple to understand and use. In a waterfall model, each phase must be completed before the next phase can begin and there is no overlapping in the phases.

The Waterfall model is the earliest SDLC approach that was used for software development. The waterfall Model illustrates the software development process in a linear sequential flow. This means that any phase in the development process begins only if the previous phase is complete. In this waterfall model, the phases do not overlap.

Waterfall Model – Design

Waterfall approach was first SDLC Model to be used widely in Software Engineering to ensure success of the project. In "The Waterfall" approach, the whole process of software development is divided into separate phases. In this Waterfall model, typically, the outcome of one phase acts as the input for the next phase sequentially. The following illustration is a representation of the different phases of the Waterfall Model.

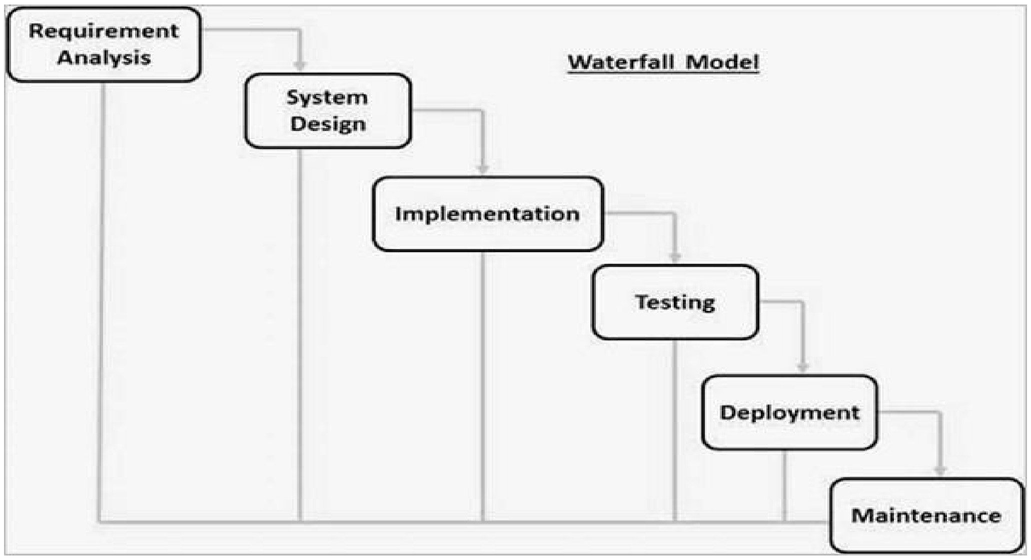


Figure1.Different phases of waterfall model

The sequential phases in Waterfall model are

- **Requirement Gathering and analysis** − All possible requirements of the system to be developed are captured in this phase and documented in a requirement specification document.
- **System Design** − The requirement specifications from first phase are studied in this phase and the system design is prepared. This system design helps in specifying hardware and system requirements and helps in defining the overall system architecture.
- **Implementation** − With inputs from the system design, the system is first developed in small programs called units, which are integrated in the next phase. Each unit is developed and tested for its functionality, which is referred to as Unit Testing.
- **Integration and Testing** − All the units developed in the implementation phase are integrated into a system after testing of each unit. Post integration the entire system is tested for any faults and failures.
- **Deployment of system** − Once the functional and non-functional testing is done; the product is deployed in the customer environment or released into the market.

- **Maintenance** − There are some issues which come up in the client environment. To fix those issues, patches are released. Also to enhance the product some better versions are released. Maintenance is done to deliver these changes in the customer environment.

Waterfall Model – Application

Every software developed is different and requires a suitable SDLC approach to be followed based on the internal and external factors. Some situations where the use of Waterfall model is most appropriate are

- Requirements are very well documented, clear and fixed.
- Product definition is stable.
- Technology is understood and is not dynamic.
- There are no ambiguous requirements.
- Ample resources with required expertise are available to support the product.
- The project is short.

Waterfall Model – Advantages

The advantages of waterfall development are that it allows for departmentalization and control. A schedule can be set with deadlines for each stage of development and a product can proceed through the development process model phases one by one.

Development moves from concept, through design, implementation, testing, installation, troubleshooting, and ends up at operation and maintenance. Each phase of development proceeds in strict order.

Some of the major advantages of the Waterfall Model are as follows –

- Simple and easy to understand anduse
- Easy to manage due to the rigidity of the model. Each phase has specific deliverables and a reviewprocess.
- Phases are processed and completed one at atime.

- Works well for smaller projects where requirements are very wellunderstood.
- Clearly definedstages.
- Well understoodmilestones.
- Easy to arrangetasks.
- Process and results are welldocumented

Waterfall Model – Disadvantages

The disadvantage of waterfall development is that it does not allow much reflection or revision. Once an application is in the testing stage, it is very difficult to go back and change something that was not well-documented or thought upon in the concept stage.

The major disadvantages of the Waterfall Model are as follows – No

working software is produced until late during the life cycle.

- Not suitable for the projects where requirements are at a moderate to high risk of changing. So, risk and uncertainty is high with this process model.
- Cannot accommodate changing requirements.

Architecture:

Client/server architecture:

- Client/server architecture is a producer/consumer computing architecture where the server acts as the producer and the client as a consumer. The server houses and provides high-end, computing-intensive services to the client on demand. These services can include application access, storage, file sharing, printer access

- Client/server architecture works when the client computer sends a resource or

processed and delivered to the client. A server computer can manage several clients simultaneously, whereas one client can be connected to several servers at

- In its simplest form, the internet is also based on client/server architecture where web servers serve many simultaneous users with website data.

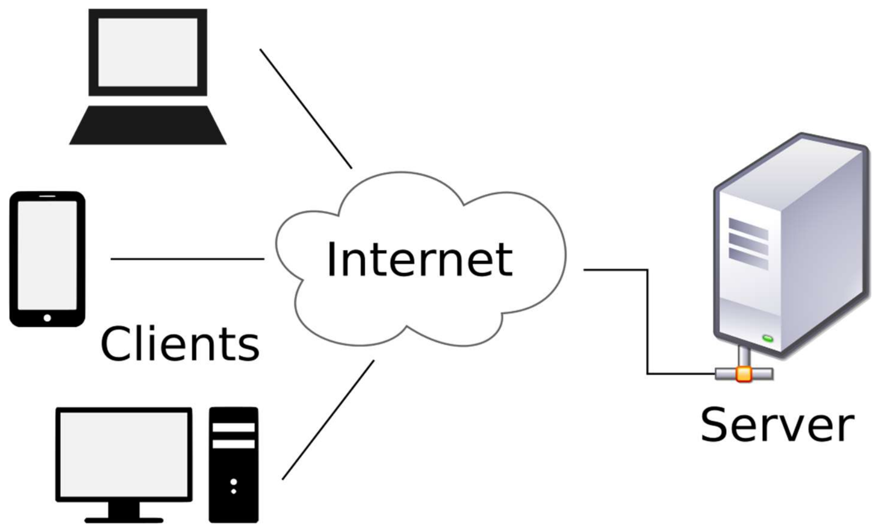


Figure 2: Client Server Architecture

Object Oriented System Development:

The Unified Modeling Language (UML) is used to specify, visualize, modify, construct and document the artifacts of an object-oriented software intensive system under development. UML offers a standard way to visualize a system's architectural blueprints, including elements such as:

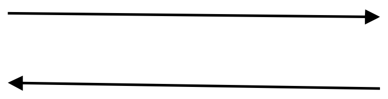
- Actors
- business processes
- components(logical)
- activities
- programming language statements
- database schemas, and
- Reusable software components.

2.4.1. Data flow diagram:

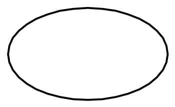
A graphical tool used to describe and analyze the moment of data through a system manual or automated including the process, stores of data, and delays in the system. Data Flow Diagrams are the central tool and the basis from which other components are developed. The transformation of data from input to output, through processes, may be described logically and independently of the physical components associated with the system. The DFD is also known as a data flow graph or a bubble chart.

DFDs are the model of the proposed system. They clearly should show the requirements on which the new system should be built. Later during activity this is taken as the basis for drawing the systems structure charts. The Basic notation used to create a DFD's are as follows:

Dataflow: Data movie in a specific direction from an origin to a destination

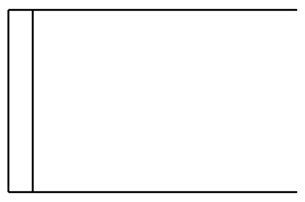


Process: People, Procedures, or devices that use or produce (Transform) Data. The physical component is not identified.



Source: External sources or destination of data, which may be people, programs, organizations or other entities.

Data store: Here data are stored or referenced by a process in the system.



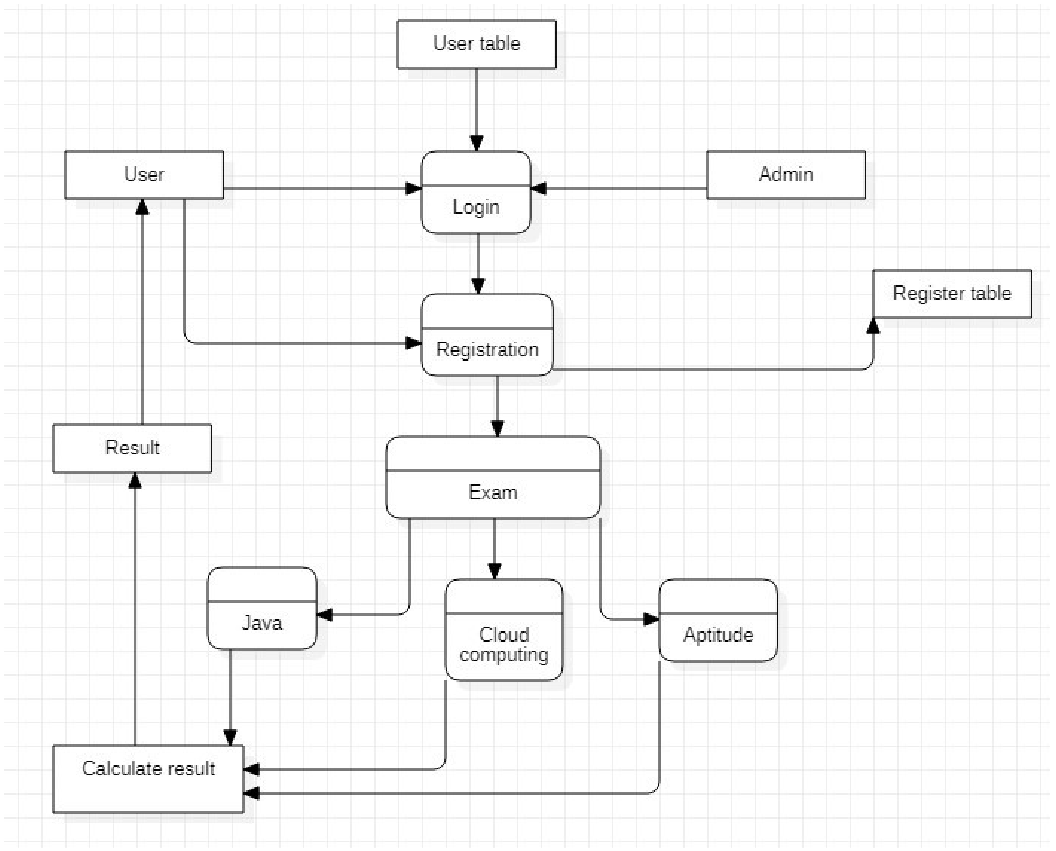


Figure 3: Data flow Diagram for Online Examination.

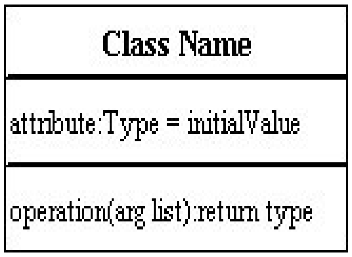
2.4.2 Class Diagram:

Class diagrams are the backbone of almost every object-oriented method including UML. They describe the static structure of a system.

Basic Class Diagram Symbols and Notations:

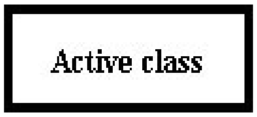
Classes represent an abstraction of entities with common characteristics. Associations represent the relationships between classes. Illustrate classes with rectangles divided into compartments. Place the name of the class in the first partition (centered, bolded, and capitalized), list the attributes in the second partition, and

Write operations into the third.


Class Name
attribute:Type = initialValue
operation(arg list):return type

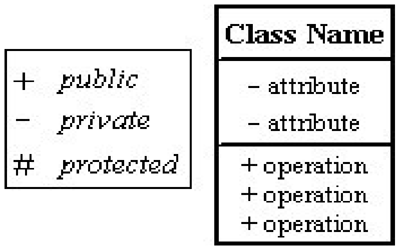
Active Class

Active classes initiate and control the flow of activity, while passive classes store data and serve other classes. Illustrate active classes with a thicker border.



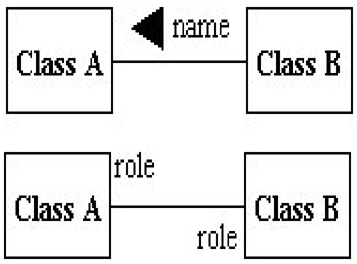
Visibility

Use visibility markers to signify who can access the information contained within a class. Private visibility hides information from anything outside the class partition. Public visibility allows all other classes to view the marked information. Protected visibility allows child classes to access information they inherited from a parent class.



Associations

Associations represent static relationships between classes. Place association names above, on, or below the association line. Use a filled arrow to indicate the direction of the relationship. Place roles near the end of an association. Roles represent the way the two classes each other.



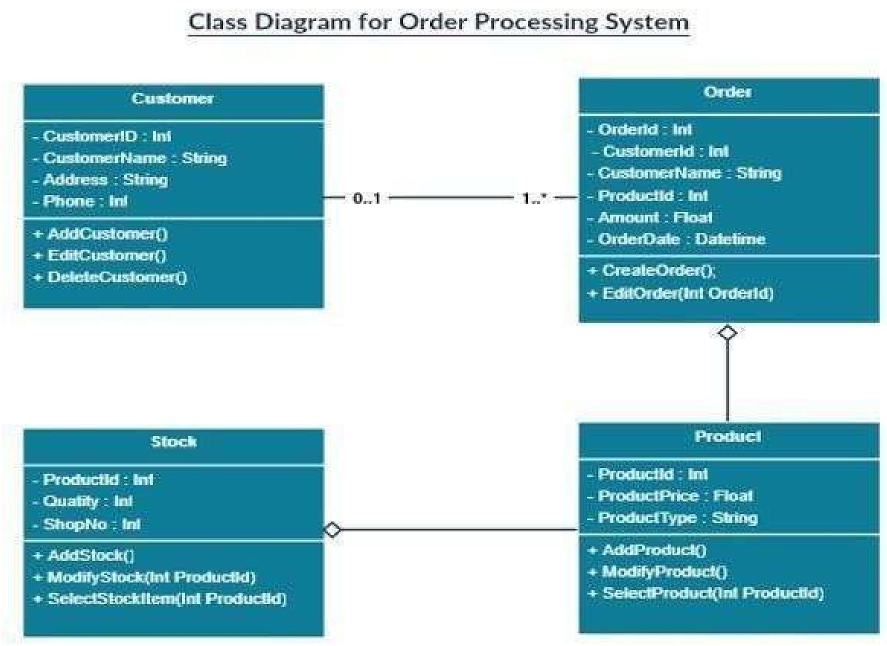


Figure 4: Class Diagram

2.4.3 Use case Diagram:

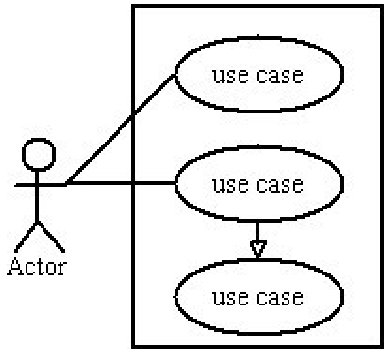
The purpose of a use case diagram in UML is to demonstrate the different ways that a user might interact with a system. Create a professional diagram for nearly any use case using our UML diagram tool.

What is a use case diagram?

In the Unified Modeling Language (UML), a use case diagram can summarize the details of your system's users (also known as actors) and their interactions with the system. To build one, you'll use a set of specialized symbols and connectors. Scenarios in which your system or application interacts with people, organizations, or external systems.

- Goals that your system or application helps those entities (known as actors) achieve

- The scope of your system.



When to apply use case diagrams

A use case diagram doesn't go into a lot of detail—for example; don't expect it to model the order in which steps are performed. Instead, a proper use case diagram depicts a high-level overview of the relationship between use cases, actors, and systems. Experts recommend that use case diagrams be used to supplement a more descriptive textual use case.UML is the modeling toolkit that you can use to build your diagrams. Use cases are represented with a labeled oval shape. Stick figures represent actors in the process, and the actor's participation in the system is

modeled with a line between the actor and use case. To depict the system boundary, draw a box around the use case itself.

UML use case diagrams are ideal for:

- Representing the goals of system-user interactions
- Defining and organizing functional requirements in a system
- Specifying the context and requirements of a system
- Modeling the basic flow of events in a use case

Diagramming is quick and easy with Lucid chart. Start a free trial today to start creating and collaborating.

Use case diagram components:

Common components include:

- **Actors:** The users that interact with a system. An actor can be a person, an organization, or an outside system that interacts with your application or system. They must be external objects that produce or consume data.
- **System:** A specific sequence of actions and interactions between actors and the system. A system may also be referred to as a scenario.
- **Goals:** The end result of most use cases. A successful diagram should describe the activities and variants used to reach the goal.

Use case Diagram Symbols and Notation:

The notation for a use case diagram is pretty straightforward and doesn't involve as many types of symbols as other UML diagrams. Here are all the shapes you will be able to find in Lucid chart:

- **Use cases:** Horizontally shaped ovals that represent the different uses that a user might have.
- **Actors:** Stick figures that represent the people actually employing the use cases.
- **Primary Actors:** The Actor(s) using the system to achieve a goal. The Use Case documents the interactions between the system and the actors to achieve the goal of the primary actor.
- **Secondary Actors:** Actors that the system needs assistance from to achieve the primary actor's goal. Secondary actors may or may not have goals that they expect to be satisfied by the use case, the primary actor always has a goal, and the use case exists to satisfy the primary actor.
- **Associations:** A line between actors and use cases. In complex diagrams, it is important to know which actors are associated with which use cases.
- **System boundary boxes:** A box that sets a system scope to use cases. All use cases outside the box would be considered outside the scope of that system. For example, Psycho Killer is outside the scope of occupations in the chain saw example found below.

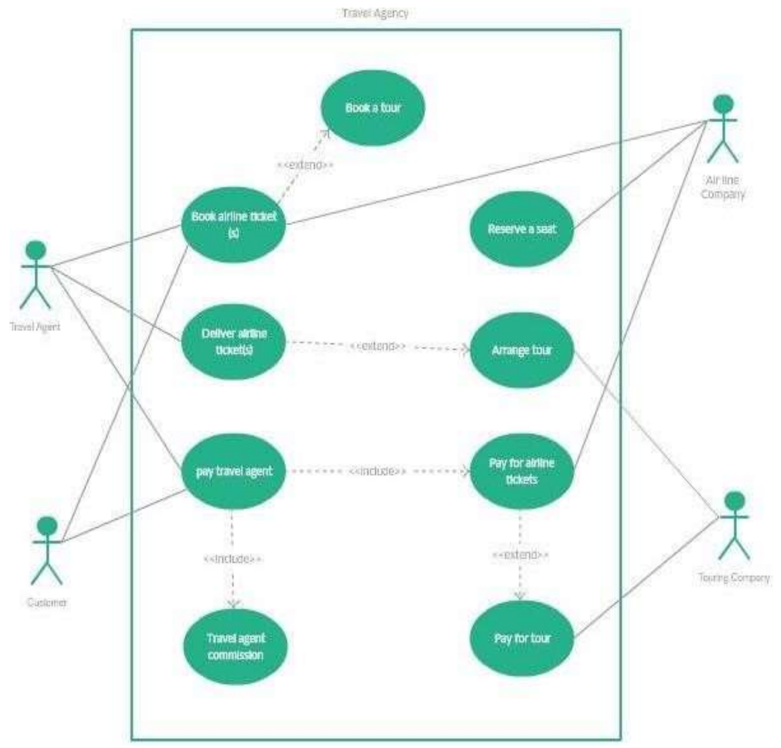


Figure 5: Use case Diagram

2.4.4 Sequence Diagram:

UML Sequence diagrams are interaction diagrams that detail how operations are carried out. As sequence diagrams can be used to capture the interaction between objects in the context of collaboration, one of the primary uses of sequence diagrams is in the transition from requirements expressed as use cases to the next and more formal level of refinement. Use cases are often refined into one or more sequence diagrams. Sequence diagrams are time focus and they show the order of the interaction visually by using the vertical axis of the diagram to represent time what messages are sent and when.

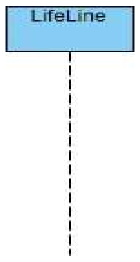
Sequence Diagrams Captures Interaction In Different Level Of Granularity:

- High-level interactions between user of the system and the system, between the system and other systems, or between subsystems (sometimes known as system sequence diagrams).
- The interaction that takes place in collaboration that either realizes a use case or an operation (instance diagrams or generic diagrams).
- Represent objects interact in (Model, View / Controller) MVC pattern of software framework.

Sequence Diagram Notations:

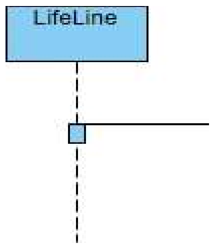
Lifeline:

A lifeline represents an individual participant in the Interaction.



Activation:

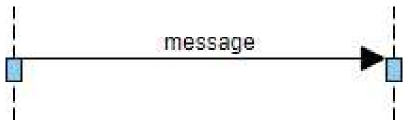
An activation is represented by a thin rectangle on a lifeline) represents the period during which an element is performing an operation. The top and the bottom of the of the rectangle are aligned with the initiation and the completion time respectively



Messages:

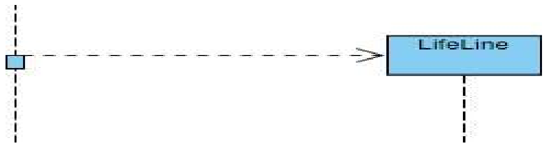
Call Message:

A call message defines a particular communication between lifelines of an interaction, which represents an invocation of operation of target lifeline.



Create Message:

A create message defines a particular communication between lifelines of an interaction, which represents the instantiation of (target) lifeline.



When to draw sequence diagram:

- Model high-level interaction between active objects in a system
- Model the interaction between object instances within a collaboration that realizes a use case
- Model the interaction between objects within a collaboration that realizes an operation
- Either model generic interactions (showing all possible paths through the interaction) or specific instances of a interaction (showing just one path through the interaction

How to draw a sequence diagram:

1. Identify a set of objects that will participate in the general collaboration (or use case scenario)

   a. If you derive the sequence diagram based on a scenario of a use case, select the normal scenarios first

   b. You should know the primary actor(s) who activates the use case

2. Consider what the system need to be done in order to response to the actor, when the actor send the message to the system

   a. What the system need to be handled before the return message response back from the system?

   b. E.g. A customer inserted an ATM card to the machine, the system will display "input pin number" in the normal scenario, right?

   c. Guess, what will to be handled inside the ADM by a set of objects at the "back" of the system? Something like, read and verify the ATM card (card reader), read the card information of the card holder (by the bank) and ask for the pin, or, return "invalid card type, insert another card", and etc.

   d. By this way, you will identify the candidate objects and operations of the target application for that particular scenario and you can also use this information as a basis to derive the class diagram incrementally.

3. Repeat each of the point of the scenario (or flow of event) and until you complete all the points in the scenario.

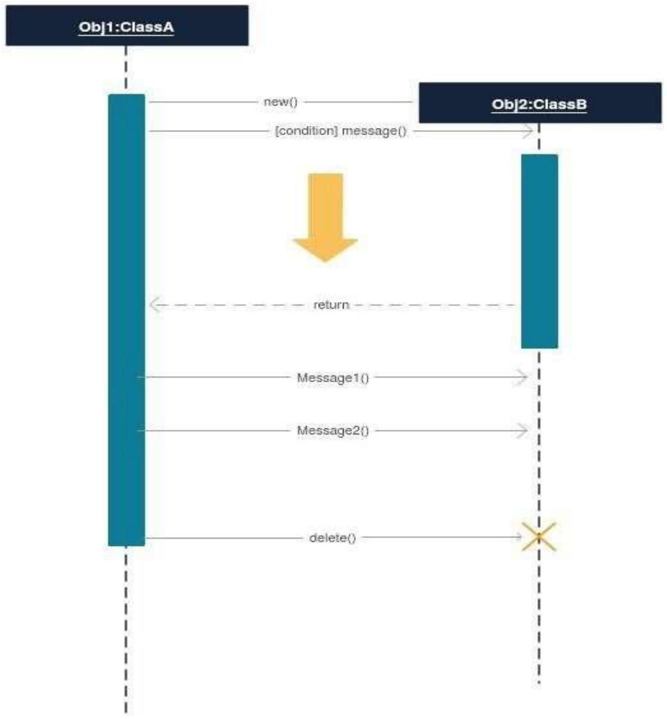


Figure 6: Sequence Diagram

2.4.5 Activity Diagram:

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.

Purpose of Activity Diagrams:

The basic purpose of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.

It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

The purpose of an activity diagram can be described as

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

How to draw an activity diagram:

Activity diagrams are mainly used as a flowchart that consists of activities performed by the system. Activity diagrams are not exactly flowcharts as they have some additional capabilities. These additional capabilities include branching, parallel flow, swimlanes, etc. Before drawing an activity diagram, we must have a clear understanding about the elements used in activity diagram. The main element of an activity diagram is the activity itself. An activity is a function performed by the system. After identifying the activities, we need to understand how they are associated with constraints and conditions.

Before drawing an activity diagram, we should identify the following elements −

- Activities
- Association
- Conditions
- Constraints

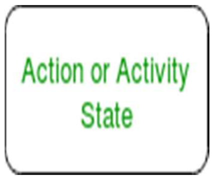
Once the above-mentioned parameters are identified, we need to make a mental layout of the entire flow. This mental layout is then transformed into an activity diagram.

Activity Diagram Notations:

1. **Initial State –** The starting state before an activity takes place is depicted using the initial state.

A process can have only one initial state unless we are depicting nested activities. We use a black filled circle to depict the initial state of a system. For objects, this is the state when they are instantiated. The Initial State from the UML Activity Diagram marks the entry point and the initial Activity State.

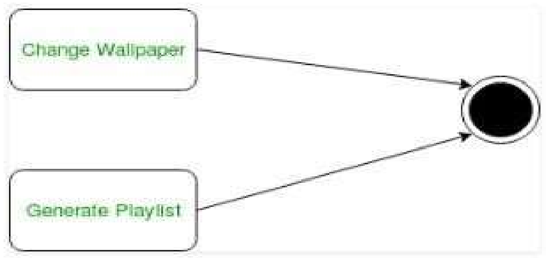
2. **Action or Activity State –** An activity represents execution of an action on objects or by objects. We represent an activity using a rectangle with rounded corners. Basically any action or event that takes place is represented using an activity.



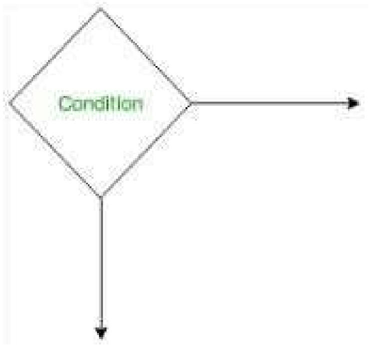
3. **Action Flow or Control flows –**Action flows or Control flows are also referred to as paths and edges. They are used to show the transition from one activity state to another.

An activity state can have multiple incoming and outgoing action flows. We use a line with an arrow head to depict a Control Flow. If there is a constraint to be adhered to while making the transition it is mentioned on the arrow. Consider the example – Here both the states transit into one final state using action flow symbols

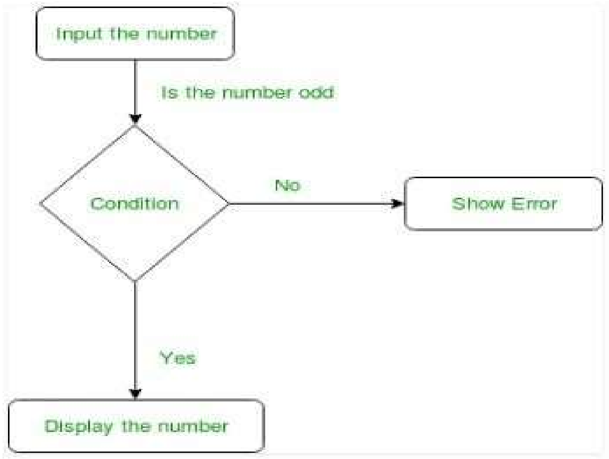
i.e. arrows.



4. **Decision node and Branching –** When we need to make a decision before deciding the flow of control, we use the decision node.



The outgoing arrows from the decision node can be labeled with conditions or guard expressions. It always includes two or more output arrows.



5. **Fork –** Fork nodes are used to support concurrent activities.

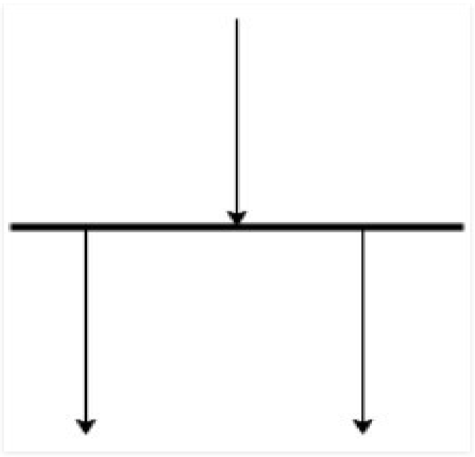
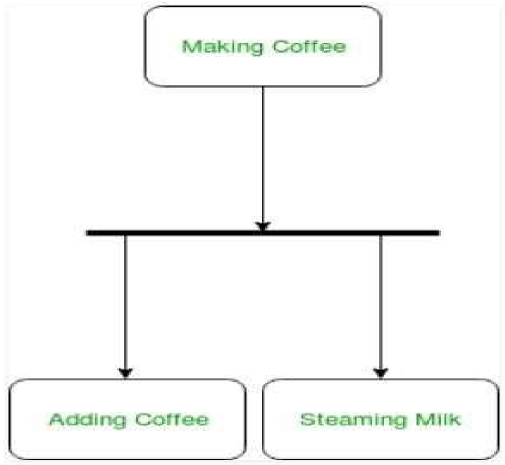
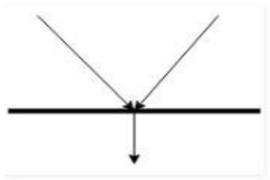


Figure – fork notation when we use a fork node when both the activities get executed concurrently i.e. no decision is made before splitting the activity into two parts. Both parts need to be executed in case of a fork statement. We use a rounded solid rectangular bar to represent a Fork notation with incoming arrow from the parent activity state and outgoing arrows towards then newly created activities.

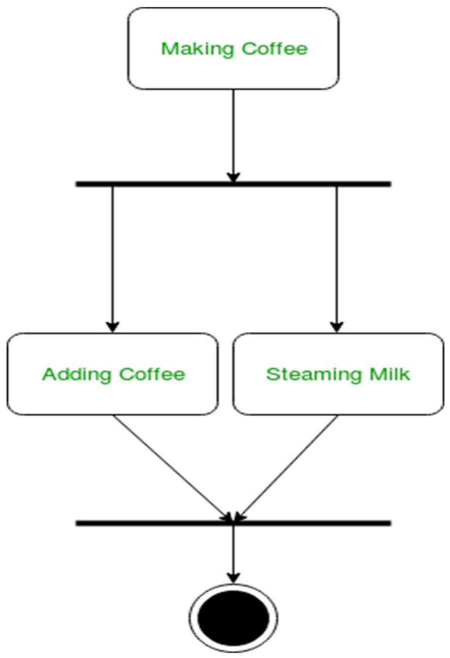
For example: In the example below, the activity of making coffee can be split into two concurrent activities and hence we use the fork notation.



6. **Join –** Join nodes are used to support concurrent activities converging into one. For join notations we have two or more incoming edges and one outgoing edge.



For example – When both activities i.e. steaming the milk and adding coffee get completed, we converge them into one final activity.



7. **Final State or End State –** The state which the system reaches when a particular process or activity ends is known as a Final State or End State. We use a filled circle within a circle notation to represent the final state in a state machine diagram. A system or a process can have multiple final states.

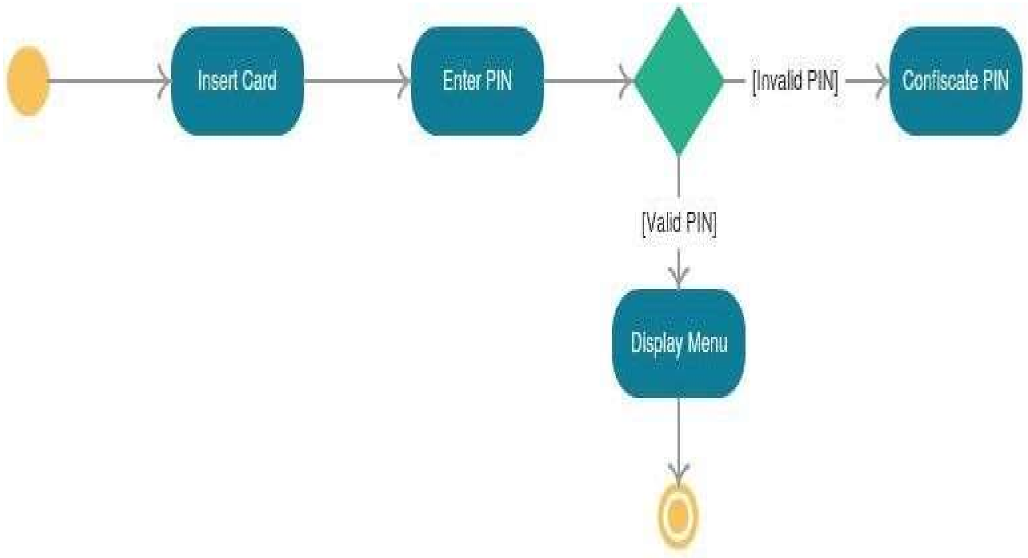


Figure 7: Activity Diagram

Data dictionary:

Data Dictionary, also called a Data Definition Matrix, provides detailed information about the business data, such as standard definitions of data elements, their meanings, and allowable values. While a conceptual or logical Relationship Diagram will focus on the high-level business concepts, a Data Dictionary will provide more detail about each attribute of a business concept. Essentially, a data dictionary provides a tool that enables you to communicate business stakeholder requirements in such a way that your technical team can more easily design a relational database or data structure to meet those requirements. It helps avoid project mishaps such as requiring information in a field that a business stakeholder can't reasonably be expected to provide, or expecting the wrong type of information in afield.

Normalization:

Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multistep process that puts data into tabular form, removing duplicated data from the relation tables.

First Normal Form (1NF):

- A relation will be 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only singlevalued attribute.
- First normal form disallows the multi-valued attribute, composite attribute, and their combinations.

Example: Relation EMPLOYEE is not in 1NF because of multi-valued attribute EMP_PHONE.

Second Normal Form (2NF)

- In the 2NF, relational must be in 1NF.
- In the second normal form, all non-key attributes are fully functional dependent on the primary key

Example: Let's assume, a school can store the data of teachers and the subjects they teach. In a school, a teacher can teach more than one subject.

Third Normal Form (3NF)

- A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.
- 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.
- If there is no transitive dependency for non-prime attributes, then the relation must be in third normal form.

A relation is in third normal form if it holds at least one of the following conditions for every non-trivial function dependency X → Y.

- X is a super key.
- Y is a prime attribute, i.e., each element of Y is part of some candidate key.

Example: EMP_STATE & EMP_CITY dependent on EMP_ZIP and EMP_ZIP dependent on EMP_ID. The non-prime attributes (EMP_STATE, EMP_CITY) transitively dependent on super key (EMP_ID). It violates the rule of third normal form. That's why we need to move the EMP_CITY and EMP_STATE to the new <EMPLOYEE_ZIP> table, with EMP_ZIP as a Primary key.

Boyce Codd normal form (BCNF)

- BCNF is the advance version of 3NF. It is stricter than 3NF.
- A table is in BCNF if every functional dependency X → Y, X is the super key of the table.
- For BCNF, the table should be in 3NF, and for every FD, LHS is super key.

Example: Let's assume there is a company where employees work in more than one department.

The Key Elements of a Data Dictionary:

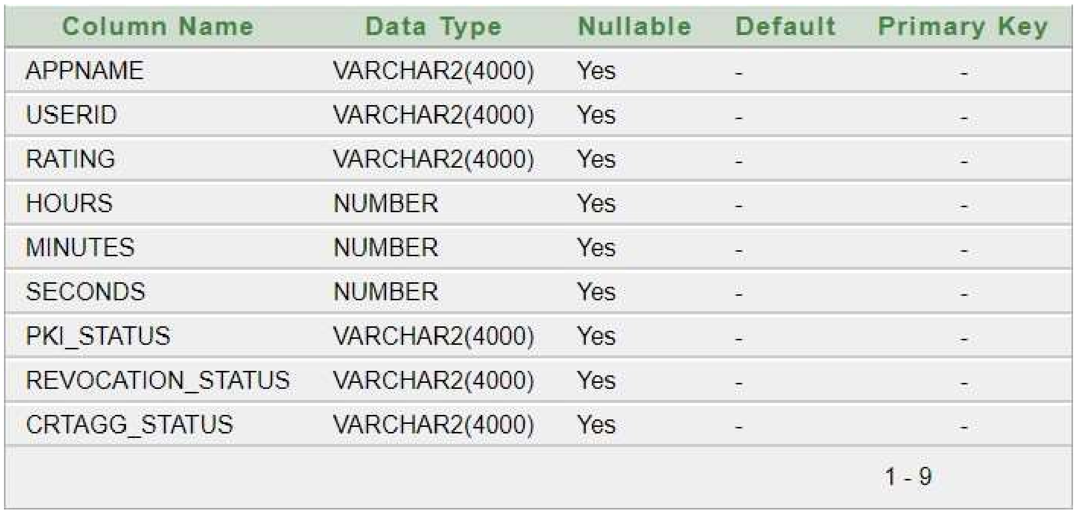
A Data Dictionary provides information about each attribute, also referred to as fields, of a data model. An attribute is a place in the database that holds information. For example, if we were to create a Data Dictionary representing the articles here on Bridging the Gap, we'd potentially have attributes for article title, article author, article category, and the article content itself. A Data Dictionary is typically organized in a spreadsheet format. Each attribute is listed as a row in the spreadsheet and each column labels an element of information that is useful to know about the attribute.

Let's look at the most common elements included in a data dictionary.

- **Attribute Name** – A unique identifier typically expressed in business language that labels each attribute.

- **Optional/Required** – Indicates whether information is required in an attribute before a record can be saved.

- **Attribute Type** – Defines what type of data is allowable in a field. Common types include text, numeric, date/time, enumerated list,look- ups,Booleansandunique identifiers. While these are the core elements of a data dictionary, it's not uncommon to document additional information about each element, which may include the source of the information, the table or concept in which the attribute is contained, the physical database field name, the field length, and any default values.



Column Name	Data Type	Nullable	Default	Primary Key
APPNAME	VARCHAR2(4000)	Yes	-	-
USERID	VARCHAR2(4000)	Yes	-	-
RATING	VARCHAR2(4000)	Yes	-	-
HOURS	NUMBER	Yes	-	-
MINUTES	NUMBER	Yes	-	-
SECONDS	NUMBER	Yes	-	-
PKI_STATUS	VARCHAR2(4000)	Yes	-	-
REVOCATION_STATUS	VARCHAR2(4000)	Yes	-	-
CRTAGG_STATUS	VARCHAR2(4000)	Yes	-	-
				1 - 9

Figure 8: Data Dictionary

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system.ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also Known as ERDs or ER models , they use a defined set of symbols such as rectangles , diamonds ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs.

ER diagrams are related to data structure diagrams (DSDs) which focus on the relationships of elements within instead f relationships between entities themselves.ER diagrams also are often used in conjunction with data flow diagrams (DFDs), which map out the flow of information for processes or systems.

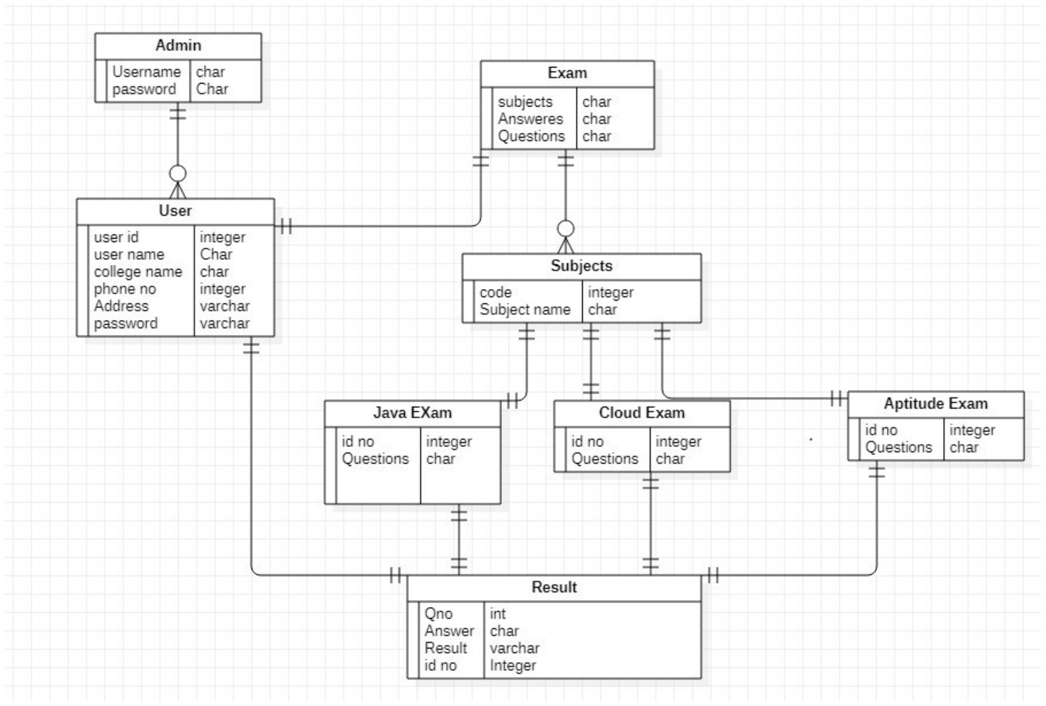


Figure 9: ER Diagram for Online Examination

Uses of entity relationship diagrams:

1. **Database design:**

ER diagrams are used to model and design relational databases, in terms of logic and business rules (in a logical data model). In software engineering, an ER diagram is often an initial step in determining requirements for an information systems project .It's also later used to model a particular database or databases. A relational database has an equivalent relational table and can potentially be expressed that way as needed.

2. **Database troubleshooting:**

ER diagrams are used to analyze existing databases to find and resolve problems in logic or deployment .Drawing the diagram should reveal where its going wrong.

3. **Business information systems:**

The diagrams are used to design or analyze relational databases used in business processes. Any business process that uses fielded data involving entities, actions and interplay can potentially benefit from a relational database. It can streamline processes, uncover information more easily and improve results.

4. **Business process re-engineering (BPR):**

ER diagrams help in analyzing databases used in business process re- engineering and in modeling in a new database setup.

5. **Education:**

Databases are today's method of storing relational information for educational purposes and later retrieval, so ER Diagrams can be valuable in planning those data structures.

6. **Research:**

Since so much research focuses on structured data, ER diagram scan play a key role in setting up useful databases to analyze the data.

JAVA

Initially the language was called as “oak” but it was renamed as “java” in 1995.The primary motivation of this language was the need for a platform-independent (i.e. architecture neutral) language that could be used to create software to be embedded in various consumer electronic devices.

- Java is a programmer’s language
- Java is cohesive and consistent
- Except for those constraint imposed by the Internet environment. Java gives the programmer, full control

Finally Java is to Internet Programming where c was to System Programming.

Importance of Java to the Internet

Java has had a profound effect on the Internet. This is because; java expands the Universe of objects that can move about freely in Cyberspace. In a network, two categories of objects are transmitted between the server and the personal computer. They are passive information and Dynamic active programs. in the areas of Security and probability. But Java addresses these concerns and by doing so, has opened the door to an exciting new form of program called the Applet.

Applications and applets.

An application is a program that runs on our computer under the operating system of that computer. It is more or less like one creating using C or C++.Java’s ability to create Applets makes it important. An Applet I San application, designed to be transmitted over the Internet and executed by a Java-compatible web browser. An applet I actually a tiny Java program, dynamically downloaded across the network, just like an image but it is an program, not just a media file. It can be reacted to the user input and dynamically change

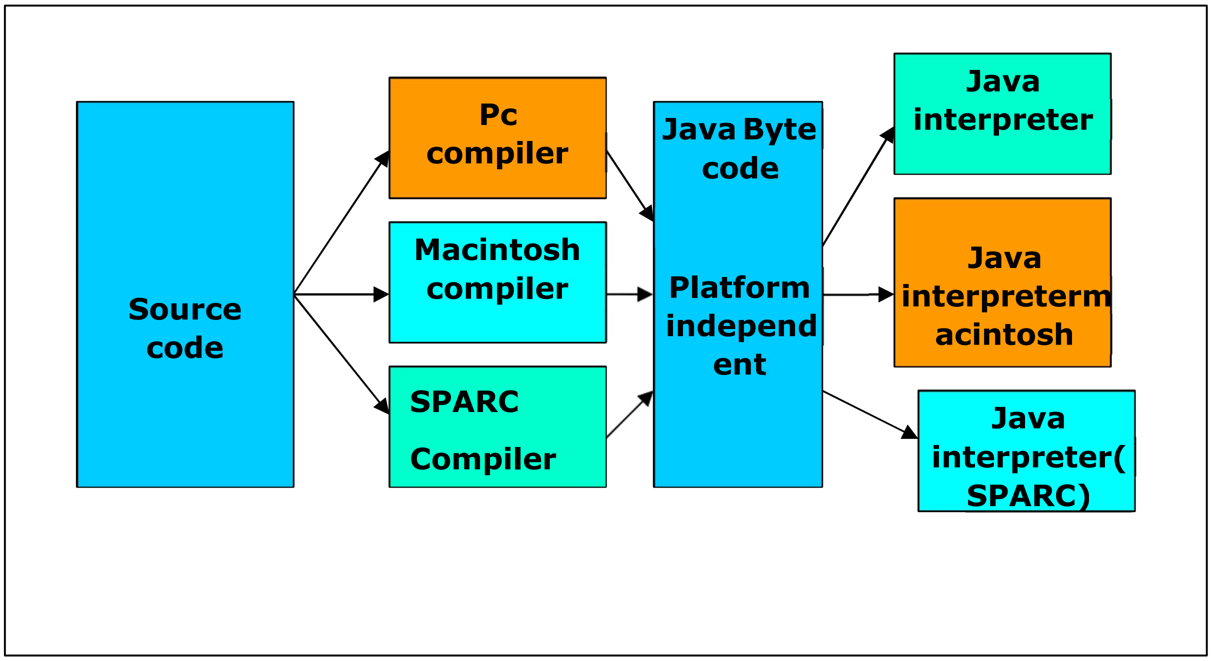
Java Architecture

Java architecture provides a portable, robust, high performing environment for development. Java provides portability by compiling the byte codes for the Java Virtual Machine, which is then interpreted on each platform by the run-time environment. Java is a dynamic system, able to load code when needed from a machine in the same room or across the planet.

Compilation of code

When you compile the code, the Java compiler creates machine code (called byte code) for a hypothetical machine called Java Virtual Machine (JVM). The JVM is supposed t executed the byte code. The JVM is created for the overcoming the issue of probability. The code is written and compiled for one machine and interpreted on all machines. This machine is called Java Virtual Machine.

Compiling and interpreting java source code.



During run-time the Java interpreter tricks the byte code file into thinking that it is running on a Java Virtual Machine. In reality this could be an Intel Pentium windows 95 or sun SPARCstation running Solaris or Apple Macintosh running system and all could receive code from any computer through internet and run the Applets.

Simple:

Java was designed to be easy for the Professional programmer to learn and to use effectively. If you are an experienced C++ Programmer. Learning Java will oriented features of C++. Most of the confusing concepts from C++ are either left out of Java or implemented in a cleaner, more approachable manner. In Java there are a small number of clearly defined ways to accomplish a given task.

Object oriented

Java was not designed to be source-code compatible with any other language. This allowed the Java team the freedom to design with a blank state. One outcome of this was a clean usable, pragmatic approach to objects. The object model in Java is simple and easy to extend, while simple types, such as integers, are kept as high-performance non-objects.

Robust

The multi-platform environment of the web places extraordinary demands on a program, because the program must execute reliably in a variety of systems. The ability to create robust programs. Was given a high priority in the design of Java. Java is strictly typed language; it checks your code at compile time and runtime.

Java virtually eliminates the problems of memory management and deal location, which is completely automatic. In a well-written Java program, all run-time errors can and should be managed by your program.

4.4.2 Servlets/JSP INTRODUCTION

A Servlet Is a generic server extension. a Java class that can be loaded

Dynamically to expand the functionality of a server. Servlets are commonly used with web servers. Where they can take the place CGI scripts.
A servlet is similar to proprietary server extension, except that it runs inside a Java Virtual Machine (JVM) on the server, so it is safe and portable

Servlets operate solely within the domain of the server.

Unlike CGI and Fast CGI, which use multiple processes to handle separate program or separate requests, separate threads within web server process handle all servlets. This means that servlets are all efficient and scalable.

Servlets are portable; both across operating systems and also across web servers. Java Servlets offer the best possible platform for web application development.

Servlets are used as replacement for CGI scripts on a web server, they can extend any sort of server such as a mail server that allows servlet's extend its functionality perhaps by performing a virus scan on all attached documents or handling mail filtering tasks.

Servlets provide a Java-based solution used to address the problems currently associated with doing server-side programming including inextensible scripting solutions platform- specific API's and incomplete interface.

Servlets are objects that conform to a specific interface that can be plugged into a Java- based server. Servlets are to the server-side what applets are to the server-side what applets are to the client-side-object byte codes that can be dynamically loaded off the net. They differ from applets in than they are faceless objects (without graphics or a GUI component).They serve as platform independent, dynamically loadable, pluggable helper byte code objects on the server side that can be used to dynamically extend server-side functionality.

For example, an HTTP servlet can be used to generate dynamic HTML content when you use servlets to do dynamic content you get the following advantages:

➤ They're faster and cleaner than CGI scripts ➤ They use a standard API (the servlet API)

➤ They provide all the advantages of Java (run on a variety of servers without needing to be rewritten)

Attractiveness of Servlets:

They are many features of servlets that make them easy and attractive to use these include:

➤ Easily configure using the GUI-based Admin tool]

- Can be Loaded and Invoked from a local disk or remotely across the network.
- Can be linked together or chained, so that on servlet can call another servlet, or several servlets in sequence.
- Can be called dynamically from within HTML, pages using server-side include- tags.
- Are secure-even when downloading across the network, the servlet security model and servlet and box protect your system from unfriendly behavior.,

Advantages of the servlet API

One of the great advantages of the servlet API is protocol independent. It assumes nothing about: The protocol being used to transmit on the net How it is loaded

- The server environment it will be running in
- These quantities are important, because it allows the Servlet API to be embedded in many different kinds of servers. There are other advantages to the servlets API as well These include:
- It's extensible-you can inherit all your functionality from the base classes made available to you It's simple small, and easy to use.

Servlets are persistent. Servlet are loaded only by the web server and can maintain services between requests.

- Servlets are fast. Since servlets only need to be l\loaded once, they offer much better performance over their CGI counterparts.
- Servlets are platform independent.
- Servlets are extensible Java is a robust, object-oriented programming language, which easily can be extended to suit your needs.
- Servlets are secure
- Servlets are used with a variety of client.

Servlets are classes and interfaces from tow packages, javax. servlet and javax. servlet. Http. The java. Servlet package contains classes t support generic, protocol-independent servlets. The classes in the javax. servlet. Http package To and HTTP specific functionality extend these classes

Every servlet must implement the javax. servlets interface. Most servlets implement it by extending oneoftwoclasses. javax. Servlet. Generic Servlet or javax. servlet. Http. HTTP Servlet. A protocol-independent servlet should subclass Generic-Servlet. while an Http servlet should subclass HTTP Servlet, which is itself a subclass of Generic-servlet with added HTTP-specific functionality.

Unlike a java program, a servlet does not have a main () method, Instead the server in the process of handling requests invoke certain methods of a servlet. Each time the server dispatches a request to a servlet, it invokes the servlet's Service () method,

A generic servlet should override its service () method to handle requests as appropriate for the servlet. The service() accepts two parameters a request object and a response object .The request object tells the servlet about the request, while the response object is used to return a response

InContrast.anHttp servlet usually does not override the service () method. Instead it
overrides doGet() to handle GET requests and do Post() to handle Post requests. An Http servlet can override either or both of these modules the service () method of HTTP Servlet handles the setup and dispatching to all the doXXX () methods. Which is why it usually should not be overridden

The remainders in the javax. servlet and javax. servlet. Http. Package are largely support classes. The Servlet Request and Servlet Response classes in javax. servlet provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse classes in javax.servelt provide access to generic server requests and responses while HttpServletRequest and HttpServletResponse in javax.servlet.http provide access a HTTP requests and responses . The javax. servlet. Http provide contains an HTTP Session class that provides built-in session tracking functionality and Cookie class that allows quickly setup and processing HTTP Cookies.

Loading Servlets:

Servlets can be loaded from their places. From a directory that is on the CLASSPATH. The CLASSPATH of the Java Webserver includes service root/classes/, which is where the system classes reside

From the <SERVICE_ROOT/servlets/directory. This is not in the server's class path. A class loader is used to create servlets form this directory. New servlets can be added-existing servlets can be recompiled and the server will notice these changes. From a remote location. For this a code base like http://nine.eng/classes/foo/ is required in addition to the servlet's class name. Refer to the admin Gui docs on servlet section to see how to set this up.

Loading Remote Servlets

Remote servlets can be loaded by:

➤ Configuring the admin Tool to setup automatic loading of remote servlets. ➤ Selecting up server side include tags in .html files

➤ Defining a filter chain Configuration

Invoking Servlets

A servlet invoker is a servlet that invokes the "server" method on a named servlet. If the servlet is not loaded in the server, then the invoker first loads the servlet (either form local disk or from the network) and the then invokes the "service" method. Also like applets, local servlets in the server can be identified by just the class name. In other words, if a servlet name is not absolute.it is treated as local.

A Client can Invoke Servlets in the Following Ways:

➤ The client can ask for a document that is served by the servlet.

➤ The client(browser) can invoke the servlet directly using a URL, once it has been mapped using the

SERVLET ALIASES Section of the admin GUI

- The servlet can be invoked through server side include tags.

- The servlet can be invoked by placing it in the servlets/directory - The servlet can be invoked by using it in a filter chain

The Servlet Life Cycle: -

The Servlet life cycle is one of the most exciting features of Servlets. This life cycle is a powerful hybrid of the life cycles used in CGI programming and lower-level NSAPI and ISAPI programming.

The servlet life cycle allows servlet engines to address both the performance and resource problems of CGI and the security concepts of low-level server API programming.

Servlet life cycle is highly flexible Servers have significant leeway in how they choose to support servlets. The only hard and fast rule is that a servlet engine must conform to the following life cycle contact: destroyed an garbage collected without handling any client request or after handling just one request

The most common and most sensible life cycle implementations for HTTP servlets are: Single java virtual machine and astatine persistence.

Init and Destroy:-

Just like Applets servlets can define init () and destroy () methods, A servlets init(ServiceConfig) method is called by the server immediately after the server constructs the servlet's instance. Depending on the server and its configuration, this can be at any of these times

- When the server states

- When the servlet is first requested, just before the service() method is invoked - At the request of the server administrator

In any case, nit() is guaranteed to be called before the servlet handles its first request

The init() method is typically used to perform servlet initialization creating or loading objects that are used by the servlet in handling of its request. In order to providing a new servlet any information about itself and its environment, a server has to call a servlets in it() method and pass an object that implement the ServletConfig interface.

This ServletConfig object supplies a servlet with information about its initialization parameters. These parameters are given to the servlets and are not associated with any single request. They can specify initial values, such as where a counter should begin counting, or default values, perhaps a template to use when not specified by the request,

The server calls a servlet's destroy() method when the servlet is about to be unloaded. In the destroy() method, a servlet should free any resources it has acquired that will not be garbage collected. The destroy() method also gives a servlet a chance to write out its unsaved. cached information or any persistent information that should be read during the next call to init().

Session Tracking:

HTTP is a stateless protocol, it provides no way for a server to recognize that a sequence of requests is all from the same client. This causes a problem for application such as shopping

cart applications. Even in chat application server can't know exactly who's making a request of several clients.

The solution for this is for client to introduce itself as it makes each request, Each clients needs to provide a unique identifier that lets the server identify it, or it needs to give some information that the server can use to properly handle the request, There are several ways to send this introductory information with each request Such as:

USER AUTHORIZATION:

One way to perform session tracking is to leverage the information that comes with User authorization. When a web server restricts access to some of its resources to only those clients that log in using a recognized username and password. After the client logs in, the username is available to a servlet through getRemoteUser ()

Wean use the username to track the session. Once a user has logged in, the browser remembers her username and resends the name and password as the user views new pages on the site. A servlet can identify the user through her username and they're by

Track her session.

The biggest advantage of using user authorization to perform session tracking is that it's easy to implement. Simply tell the protect a set of pages, and use getRemoteUser () to identify each client. Another advantage is that the technique works even when the user accesses your site form or exists her browser before coming back.

The biggest disadvantage of user authorization is that it requires each user to register for an account and then log in in each time the starts visiting your site. Most users will tolerate registering and lagging in as a necessary evil when they are accessing sensitive information, but its all overkill for simple session tracking. Other problem with user authorization is that a user cannot simultaneously maintain more than one session at the same site.

Hidden Form Fields:

One way to support anonymous session tracking is to use hidden from fields. As the name implies, these are fields added to an HTML, form that are not displayed in the client's browser, they are sent back to the server when the form that contains them is submitted.

In a sense, hidden form fields define constant variables for a form. To a servlet receiving a submitted form, there is no difference between a hidden fields and a visible filed.

As more and more information is associated with a client session. It can become burdensome to pass it all using hidden form fields. In these situations, it's possible to pass on just a unique session ID that identifies as particular client's session.

That session ID can be associated with complete information about its session that is stored on the server.

The advantage of hidden form fields is their ubiquity and support for anonymity. Hidden fields are supported in all the popular browsers, they demand on special server requirements, and they can be used with clients that haven't registered or logged in.

The major disadvantage with this technique, however is that works only for a sequence of dynamically generated forms, The technique breaks down immediately with static documents, emailed documents book marked documents and browser shutdowns.

URL Rewriting:

URL rewriting is another way to support anonymous session tracking, With URL rewriting every local URL the user might click on is dynamically modified. or rewritten, to include extra, information. The extra information can be in the deform of extra path information, added parameters, or some custom, server-specific.URL change. Due to the limited space available in rewriting a URL, the extra information is usually limited to a unique session.

Each rewriting technique has its own advantage and disadvantage

Using extra path information works on all servers, and it works as a target for forms that use both the Get and Post methods. It does not work well if the servlet has to use the extra path information as true path information

The advantages and disadvantages of URL. Rewriting closely match those of hidden form fields, The major difference is that URL rewriting works for all dynamically created documents, such as the Help servlet, not just forms. With the right server support, custom URL rewriting can even work for static documents.

Persistent Cookies:

A fourth technique to perform session tracking involves persistent cookies. A cookie is a bit of information. sent by a web server to a browser that can later be read back form that browser. When a browser receives a cookie, it saves the cookie and there after sends the cookie back to the server each time it accesses a page on that server, subject to certain rules. Because a cookie's value can uniquely identify a client, cookies are often used for session tracking.

Persistent cookies offer an elegant, efficient easy way to implement session tracking. Cookies provide as automatic an introduction for each request as we could hope for. For each request, a cookie can automatically provide a client's session ID or perhaps a list of client's performance. The ability to customize cookies gives them extra power and versatility.

The biggest problem with cookies is that browsers don't always accept cookies sometimes this is because the browser doesn't support cookies. More often it's because

The browser doesn't support cookies. More often it's because the user has specifically configured the browser to refuse cookies.

The power of serves:

The power of servlets is nothing but the advantages of servlets over other approaches, which include portability, power, efficiency, endurance, safety elegance, integration, extensibility and flexibility.

Portability:

As servlets are written in java and conform to a well-defined and widely accepted API. They are highly portable across operating systems and across server implementation

We can develop a servlet on a windows NT machine running the java web server and later deploy it effortlessly on a high-end Unix server running Apache. With servlets we can really "write once, serve everywhere"

Servlet portability is not the stumbling block it so often is with applets, for two reasons

First, Servlet portability is not mandatory i.e. servlets has to work only on server machines that we are using for development and deployment

Second, servlets avoid the most error-prone and inconstancy implemented portion of the java languages.

Power:

Servlets can harness the full power of the core java. API's: such as Networking and Url access, multithreading, image manipulation, data compression, data base connectivity, internationalization, remote method invocation(RMI) CORBA connectivity, and object serialization, among others.

Efficiency And Endurance:

Servlet invocation is highly efficient, once a servlet is loaded it generally remains in the server's memory as a single object instance, there after the server invokes the servlet to handle a request using a simple, light weighted method invocation. Unlike the CGI, there's no process to spawn or interpreter to invoke, so the servlet can begin handling the request almost immediately, Multiple, concurrent requests are handled the request almost immediately. Multiple, concurrent requests are handled by separate threads, so servlets are highly scalable.

Servlets in general are enduring objects. Because a servlet stays in the server's memory as a single object instance. it automatically maintains its state and can hold onto external resources, such as database connections.

Safety:

Servlets support safe programming practices on a number of levels.

As they are written in java, servlets inherit the strong type safety of the java language. In addition, the servlet API is implemented to be type safe. Java's automatic garbage collection and lack of pointers mean that servlets are generally safe from memory management problems like dangling pointers invalid pointer references and memory leaks.

Servlets can handle errors safely, due to java's exception – handling mechanism. If a servlet divides by zero or performs some illegal operations, it throws an exception that can be safely caught and handled by the server.

A server can further protect itself from servlets through the use of java security manager. A server can execute its servlets under the watch of a strict security manager.

Elegance:

The elegance of the servlet code is striking. Servlet code is clean, object oriented modular and amazingly simple one reason for this simplicity is the served API itself. Which includes methods and classes to handle many of the routine chores of servlet development. Even advanced to operations like cookie handling and session tracking are abstracted int convenient classes.

Integration:

Servlets are tightly integrated with the server. This integration allows a servlet to cooperate with the server in two ways. for e.g.: a servlet can use the server to translate file paths, perform logging, check authorization, perform MIME type mapping and in some cases even add users to the server's user database.

Extensibility and Flexibility:

The servlet API is designed to be easily extensible. As it stands today the API includes classes that are optimized for HTTP servlets. But later it can be extended and optimized for another type of servlets. It is also possible that its support for HTTP servlets could be further enhanced.

Servlets are also quite flexible, Sun also introduced java server pages. which offer a way to write snippets of servlet code directly with in a static HTML page using syntax similar to Microsoft's Active server pages (ASP)

JDBC What is

JDBC?

any relational database. One can write a single program using the JDBC API, and the JDBC is a Java Api for executing SQL, Statements (As a point of interest JDBC is trademarked name and is not an acronym; nevertheless, Jdbc is often thought of as standing for Java Database Connectivity. It consists of a set of classes and interfaces written in the Java Programming language. JDBC provides a standard API for tool/database developers and makes it possible to write database applications using a pure Java API

Using JDBC, it is easy to send SQL statements to virtually program will be able to send SQL. statements to the appropriate database. The Combination of Java and JDBC lets a programmer writes it once and run it anywhere.

What Does JDBC Do?

Simply put, JDBC makes it possible to do three things

○ Establish a connection with a database ○ Send SQL

statements ○ Process the results ○ JDBC Driver Types

○ The JDBC drivers that we are aware of this time fit into one of four categories ○

JDBC-ODBC Bridge plus ODBC driver

○ Native-API party-java driver

- JDBC-Net pure java driver o Native-protocol pure Java driver

An individual database system is accessed via a specific JDBC driver that implements the java. SQL. Driver interface. Drivers exist for nearly all-popular RDBMS systems, through few are available for free. Sun bundles a free JDBC-ODBC bridge driver with the JDK to allow access to a standard ODBC, data sources, such as a Microsoft Access database, Sun advises against using the bridge driver for anything other than development and very limited development.

JDBC drivers are available for most database platforms, from a number of vendors and in a number of different flavours. There are four driver categories

Type 01-JDBC-ODBC Bridge Driver

Type 01 drivers use a bridge technology to connect a java client to an ODBC database service. Sun's JDBC-ODBC bridge is the most common type 01 driver. These drivers implemented using native code.

Type 02-Native-API party-java Driver

Type 02 drivers wrap a thin layer of java around database-specific native code libraries for Oracle databases, the native code libraries might be based on the OCI (Oracle call Interface) libraries, which were originally designed for **C/ C++** programmers, because type- 02 drivers are implemented using native code. in some cases, they have better performance than their all-java counter parts. They add an element of risk, however, because a defect in a driver's native code section can crash the entire server

Type 03-Net-Protocol All-Java Driver

Type 03 drivers communicate via a generic network protocol to a piece of custom middleware. The middleware component might use any type of driver to provide the actual database access. These drivers are all java, which makes them useful for applet deployment

and safe for servlet deployment

Type-04-native-protocol All-java Driver

Type o4 drivers are the most direct of the lot. Written entirely in java, Type 04 drivers understand database-specific networking. protocols and can access the database directly without any additional software

JDBC-ODBC Bridge

If possible, use a Pure Java JDBC driver instead of the Bridge and an ODBC driver. This completely eliminates the client configuration required by ODBC. It also eliminates the potential that the Java VM could be corrupted by an error in the native code brought in by the Bridge (that is, the Bridge native library, the ODBC driver manager library, library, the ODBC driver library, and the database client library)

WHAT IS The JDBC-ODBE Bridge?

The JDBC-ODBC Bridge is a Jdbc driver, which implements JDBC operations by translating them into ODBC operations. To ODBC it appears as a normal application program. The Bridge is implemented as the sun. Jdbc. ODBC Java package and contains a native library used to access ODBC. The Bridge is joint development of Innersole and Java Soft

Oracle

Oracle is a relational database management system, which organizes data in the form of tables. Oracle is one of many database servers based on RDBMS model, which manages a seer of data that attends three specific things-data structures, data integrity and data manipulation.

With oracle cooperative server technology, we can realize the benefits of open, relational systems for all the applications. Oracle makes efficient use of all systems resources, on all hardware architecture; to deliver unmatched performance, price performance and scalability. Any DBMS to be called as RDBMS has to satisfy Dr.E.F. Codd's rules.

Features of Oracle:

Portable

The Oracle RDBMS is available on wide range of platforms ranging from PCs to super computers and as a multi user loadable module for Novel NetWare, if you develop application on system, you can run the same application on other systems without any modifications.

Compatible

Oracle commands can be used for communicating with IBM DB2 mainframe RDBMS that is different from Oracle, which is Oracle compatible with DB2. Oracle RDBMS is a high- performance fault tolerant DBMS, which is specially designed for online transaction processing and for handling large database applications.

Multithreaded Server Architecture

Oracle adaptable multithreaded server architecture delivers scalable high performance for very large number of users on all hardware architecture including symmetric multiprocessors (sumps) and loosely coupled multiprocessors. Performance is achieved by eliminating CPU, I/O, memory and operating system bottlenecks and by optimizing the Oracle DBMS server code to eliminate all internal bottlenecks. Oracle has become the most popular RDBMS in the market because of its ease of use

- Client/server architecture.
- Data independence.
- Ensuring data integrity and data security.
- Managing data concurrency.
- Parallel processing support for speed up data entry and online transaction processing used for applications.
- DB procedures, functions and packages.

Dr.E.F. Codd's Rules

These rules are used for valuating a product to be called as relational database management systems.

Out of 12 rules, a RDBMS product should satisfy at least 8 rules + rule called rule 0 that must be satisfied.

RULE 0: Foundation Rule

For any system to be advertised as, or claimed to be relational DBMS should manage database with in itself, without using an external language.

RULE 1: Information Rule

All information in relational database is represented at logical level in only one way as values in tables.

RULE 2: Guaranteed Access

Each and every data in a relational database is guaranteed to be logically accessibility by using to a combination of table name, primary key value and column name.

RULE 3: Systematic Treatment of Null Values

Null values are supported for representing missing information and inapplicable information.

They must be handled in systematic way, independent of data types.

RULE 4: Dynamic Online Catalog based Relation Model

The database description is represented at the logical level in the same way as ordinary data so that authorized users can apply the same relational language to its interrogation as they do to the regular data.

RULE 5: Comprehensive Data Sub Language

A relational system may support several languages and various models of terminal use. However, there must be one language whose statement can express all of the following:

Data Definitions, View Definitions, Data Manipulations, Integrity, Constraints, Authorization and transaction boundaries.

RULE 6: View Updating

Any view that is theoretical can be updatable if changes can be made to the tables that effect the desired changes in the view.

RULE 7: High level Update, Insert and Delete

The capability of handling a base relational or derived relational as a single operand applies not only retrieval of data also to its insertion, updating, and deletion.

RULE 8: Physical Data Independence

Application program and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access method.

RULE 9: Logical Data Independence

Application programs and terminal activities remain logically unimpaired whenever any changes are made in either storage representation or access methods.

RULE 10: Integrity Independence

Integrity constraints specific to particular database must be definable in the relational data stored in the catalog, not in application program.

RULE 11: Distributed Independence

Whether or not a system supports database distribution, it must have a data sub-language that can support distributed databases without changing the application program.

RULE 12: Non-Sub-Version

If a relational system has low level language, that low language cannot use to subversion or by pass the integrity rules and constraints expressed in the higher-level relational language.

Oracle supports the following Codd's Rules

Rule 1: Information Rule (Representation of information)-YES. Rule 2: Guaranteed Access-YES. Rule 3: Systematic treatment of Null values-YES.

Rule 4: Dynamic on-line catalog-based Relational Model-YES. Rule 5: Comprehensive data sub language-YES.

Rule 6: View Updating-PARTIAL.

Rule 7: High-level Update, Insert and Delete-YES. Rule 8: Physical data Independence- PARTIAL.

Rule 9: Logical data Independence-PARTIAL. Rule 10: Integrity Independence-PARTIAL. Rule 11: Distributed Independence-YES.
Rule 12: Non-subversion-YES.

HTML

Hypertext Markup Language (HTML), the languages of the world wide web (WWW), allows users to produces web pages that included text, graphics and pointer to other web pages (Hyperlinks).
HTML is not a programming language but it is an application of ISO Standard 8879, SGML (Standard Generalized Markup Language), but
Specialized to hypertext and adapted to the Web. The idea behind Hypertext one point to another point. We can navigate through the information based on out interest and preference. A markup language is simply a series of items enclosed within the elements should be displayed.
Hyperlinks are underlined or emphasized works that load to other documents or some portions of the same document.
Html can be used to display any type of document on the host computer, which can be geographically at a different location. It is a versatile language and can be used on any platform or desktop

HTML provides tags (special codes) to make the document look attractive.

HTML provides are not case-sensitive. Using graphics, fonts, different sizes, color, etc... can enhance the presentation of the document. Anything That is not a tag is part of the document itself.

Basic Html Tags:

<! -- -->Specific Comments.

<A>………</A>Creates Hypertext links.

<B>………</B>Creates hypertext links.

<Big>……. </Big>Formats text in large-font

<Body>……. </Body>contains all tags and text in the Html-document

<Center>……</Center> Creates Text

<DD>………. </DD>Definition of a term.

<TABLE>……</TABLE>creates table

<Td>………. </Td>indicates table data in a table.

<Tr>………. </Tr>designates a table row

<Th>………. </Th>creates a heading in a table. **ADVANTAGES:**

-

A HTML document is small and hence easy to send over the net. It is small because it does not include formatted information.

HTML is platform independent HTML tags are not case-sensitive.

JAVA SCRIPT

The Java Script Language

JavaScript is a compact, object-based scripting language for developing client and server internet applications. Netscape Navigator 2.0 interprets JavaScript statements embedded similar to common gateway interface (CGI) programs.

In a client application for Navigator, JavaScript statements embedded in an HTML Page can recognize and respond to user events such as mouse clicks form
Input, and page navigation.

For example, you can write a JavaScript function to verify that users enter valid information into a form requesting a telephone number or zip code. Without any network transmission, an Html page with embedded Java Script can interpret the entered text and alert the user with a message dialog if the input is invalid or you can use JavaScript to perform an action (such as play an audio file, execute an applet, or communicate with a plug-in) in response to the user opening or exiting a page.

4.2.1 SOFTWARE REQUIREMENTS

Operating System	:	Windows Family or higher version
Techniques	:	JDK 1.8
Data Bases	:	MySQL
Server	:	Apache Tomcat

4.2.2 HARD WARE REQUIREMENTS:

Processor	:	Pentium-III (or) Higher
Ram	:	64MB (or) Higher
Cache	:	512MB
Hard disk	:	10GB

3.3 MODULES

- **ADMIN**
- **MUNICIPAL AUTHORITIES**
- **PUBLIC MODULES DESCRIPTION Admin.**

In this application the admin is one of the Modules and here the admin can directly login with the application. And the admin can add the authorities Authorize public and view all complaints.

Municipal Authorities

Here the municipal authorities can login with the application and the municipal authorities perform some actions like post water details, and check complaints.

Public

Here the public is also one of the modules the public can register with the application and public should authorize by the admin, then only the public can login into the application, and the public can perform the following operations such as check the water details, complaint on water and check the complaint status.

3.4 FUNCTIONAL REQUIREMENTS

ADMIN

Hydro can be a great tool for aceesing water-related data and resources for administrative purposes.
In the admin module, Hydro can provide valuable information such as water quality reports, consumption data, infrastructure updates, and conservation tips. By leveraging Hydro, administrators can make informed decisions regarding water management, maintenance, and sustainability initiatives.

PUBLIC

In the public module, Hydro can offer valuable insights into water conservation tips, local water quality reports, educational resources on water usage, and updates on water-related events in the community. By utilizing Hydro, the public can stay informed about water conservation practices, understand the importance of water sustainability, and access relevant information to make informed decisions about their water usage.

MUNICIPAL AUTHORITIES

In the municipal authorities module, Hydro can provide crucial information such as water usage patterns, infrastructure maintenance reports, water quality monitoring data, and emergency response protocols. By leveraging Hydro, municipal authorities can better understand water supply and demand dynamics, identify areas for improvement in water infrastructure, and respond promptly to water-related issues in the community.

FEASIBILITY STUDY:

Preliminary investigation examines project feasibility, the likelihood the system will be useful to the organization. The main objective of the feasibility study is to test the Technical, Operational and Economical feasibility for adding new modules and debugging old running system. All systems are feasible if they are given unlimited resources and infinite time. There are aspects in the feasibility study portion of the preliminary investigation:

• Technical Feasibility

• Operation Feasibility

• Economic Feasibility

TECHNICAL FEASIBILITY

The technical issue usually raised during the feasibility stage of the investigation includes the following:

• Does the necessary technology exist to do what is suggested?

• Do the proposed equipment's have the technical capacity to hold the data required to use the new system?

• Will the proposed system provide adequate response to inquiries, regardless of the number or location of users?

• Can the system be upgraded if developed?

Are there technical guarantees of accuracy, reliability, ease of access and data security?

OPERATIONAL FEASIBILITY

User-friendly

Customer will use the forms for their various transactions i.e., for adding new routes,

viewing the routes details. Also, the Customer wants the reports to view the various transactions based on the constraints. These forms and reports are generated as user-friendly to the Client.

Reliability

The package wills pick-up current transactions on line. Regarding the old transactions, User will enter them in to the system.

Security

The web server and database server should be protected from hacking, virus etc **Portability**

The application will be developed using standard open-source software (Except Oracle) like Java, tomcat web server, Internet Explorer Browser etc. this software will work both on Windows and Linux o/s. Hence portability problems will not arise.

Availability

This software will be available always.

Maintainability

The system called the wheels uses the 2-tier architecture. The 1st tier is the GUI, which is said to be frontend and the 2nd tier is the database, which uses My-Sql, which is the back-end. The front-end can be run on different systems (clients). The database will be running at the server. Users access these forms by using the user-ids and the passwords.

ECONOMIC FEASILITY

The computerized system takes care of the present existing system's data flow and procedures completely and should generate all the reports of the manual system besides a host of other management reports.

It should be built as a web-based application with separate web server and database server. This is required as the activities are spread throughout the organization customer wants a centralized database. Further some of the linked transactions take place in different

locations.Open-source software like TOMCAT, JAVA, MySQL and Linux is used to minimize the cost for the Customer.

FEASIBILITY ANALYSIS

Identifies Feasibility analysis is a critical phase in the project planning process, serving as the compass that guides decision-makers. It encompasses a comprehensive evaluation of the proposed project's potential, considering various aspects such as technical, economic, operational, scheduling, legal, and market feasibility.

From a technical perspective, it addresses whether the project can be successfully executed given the available technology, expertise, and infrastructure. It examines compatibility with existing systems, potential technical challenges, and assesses the feasibility of integrating with external systems.

Economic feasibility delves into the financial aspects of the project. It scrutinizes development, operational, and maintenance costs, compares these costs against anticipated benefits through cost-benefit analysis, and calculates the projected return on investment (ROI).

Operational feasibility investigates whether the project can be smoothly integrated into the organization's operations. It assesses alignment with organizational goals, staff readiness, potential impact on existing processes, change management requirements, and long-term sustainability.

Scheduling feasibility ensures that the project can be completed within a reasonable timeframe. It outlines the project timeline, resource availability, dependencies on external factors, and contingency plans for potential delays.

Legal and regulatory feasibility verifies the project's compliance with relevant laws, regulations, and industry standards. It addresses data privacy and security, environmental and safety regulations, intellectual property considerations, and any necessary permits and approvals.

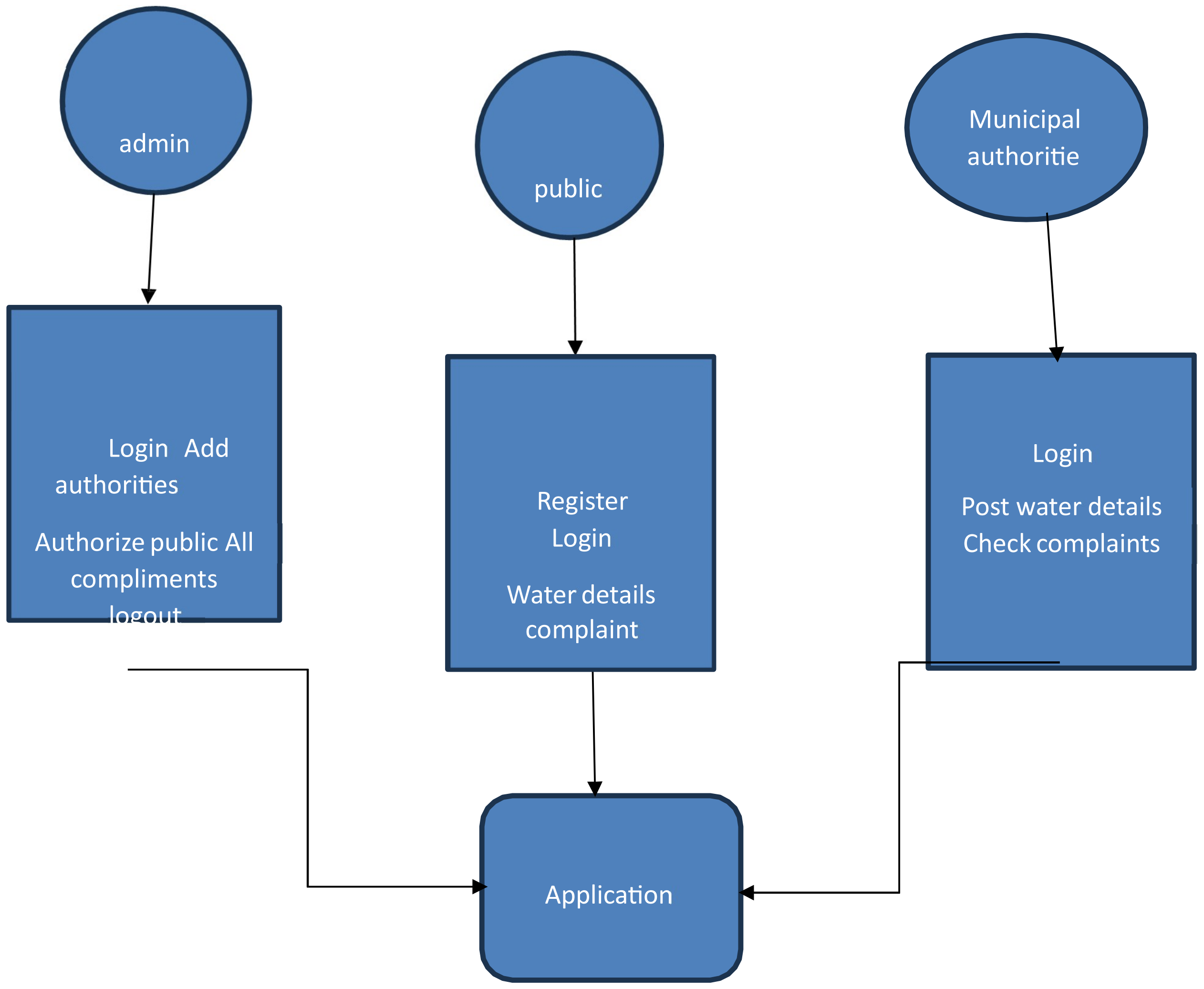
In some cases, market feasibility evaluates whether there is demand for the project's outcomes or products. It involves market research, competitive analysis, assessment of market size and growth potential, and the formulation of pricing and **revenue strategies.**

Risk analysis identifies potential risks and uncertainties associated with the project, from technical challenges to regulatory changes. It also entails the development of risk mitigation strategies.

The culmination of this analysis is a comprehensive feasibility report, summarizing findings and providing recommendations. It equips stakeholders with the information needed to make an informed decision about the project's viability, considering all aspects of feasibility, potential risks, and alignment with strategic goals. Ultimately, a well-conducted feasibility analysis is an essential tool for sound decision- making in project planning.

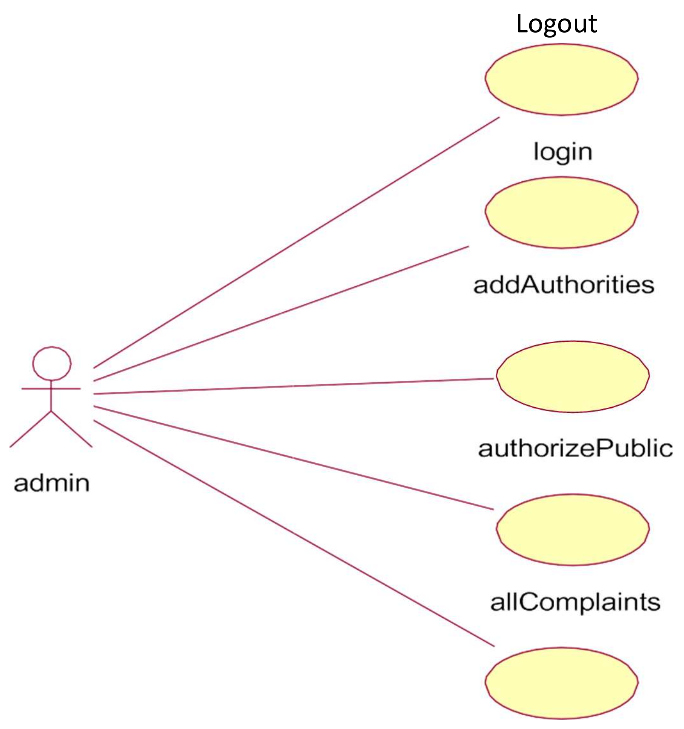
DESIGN

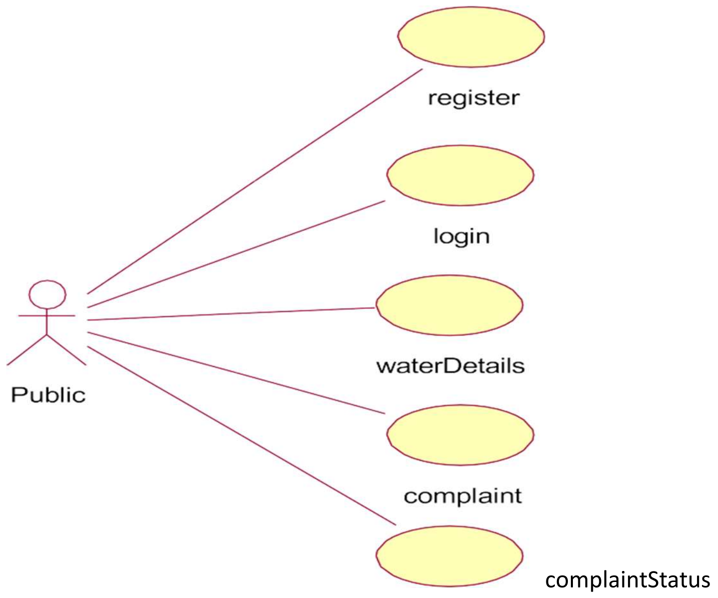
6.1 DATA FLOW DIAGRAM

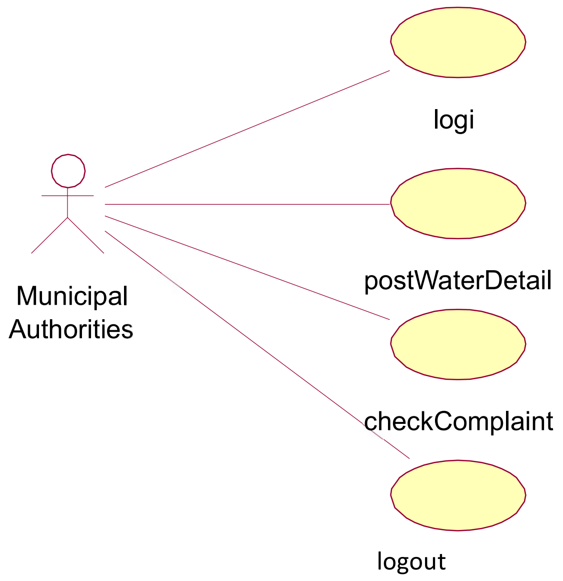


6.2.1 USE CASE DIAGRAM:

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. The main purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted

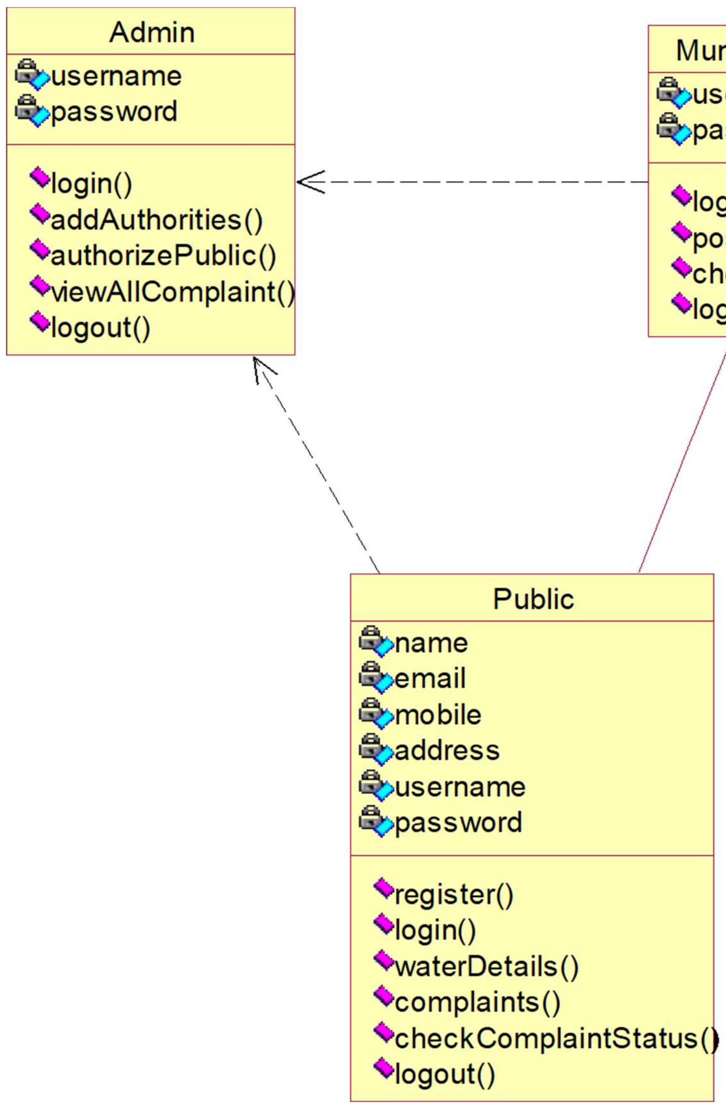






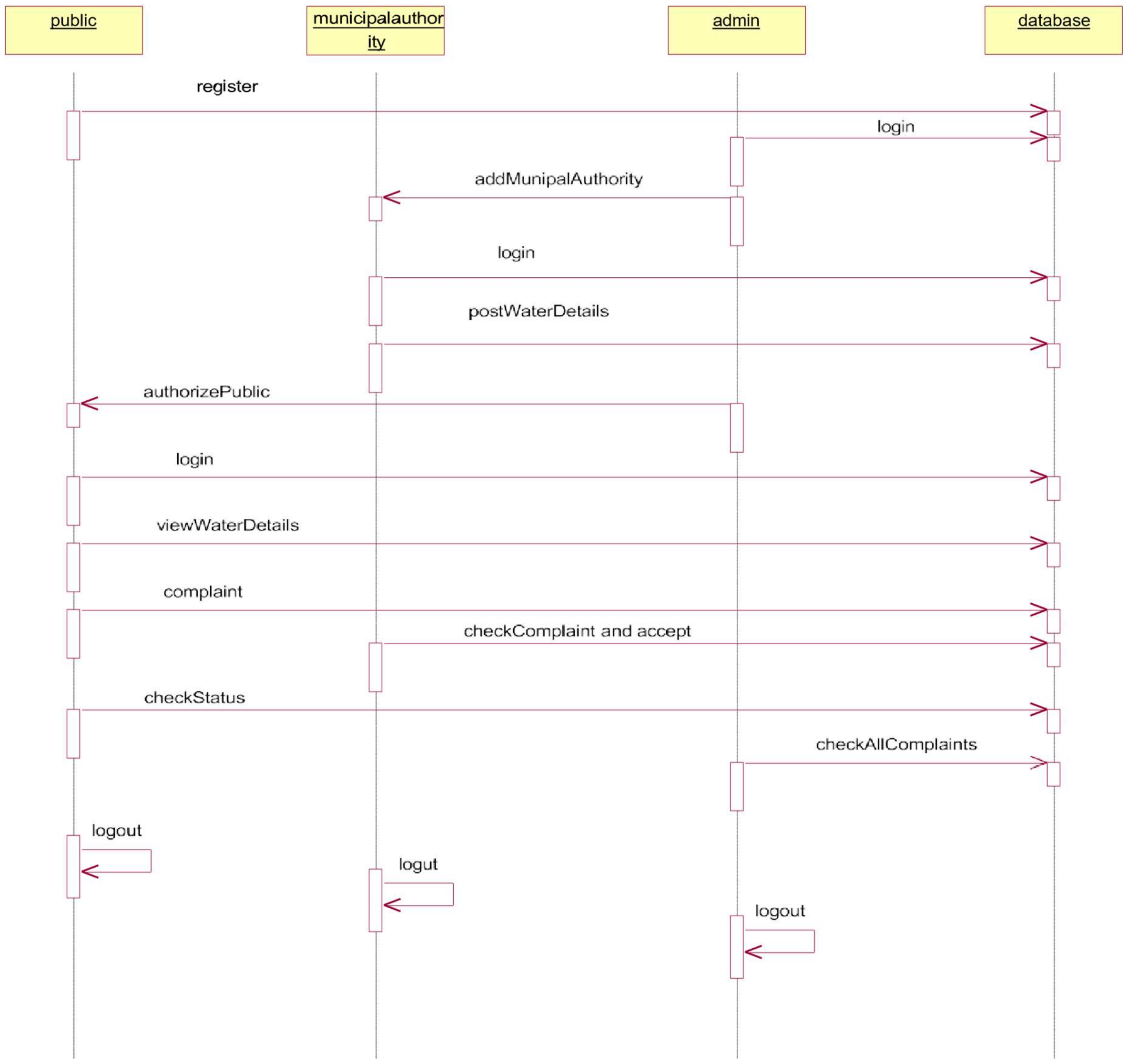
6.2.2 CLASS DIAGRAM

A class diagram is a type of diagram in the Unified Modeling Language (UML) that represents the structure and relationships of classes in a system. It is a fundamental tool used in software engineering and object-oriented modeling to visually depict the various classes, their attributes, methods, and associations within a software application or system. Class diagrams are essential for designing and understanding the architecture of software systems.



6.2.3 SEQUENCE DIAGRAM

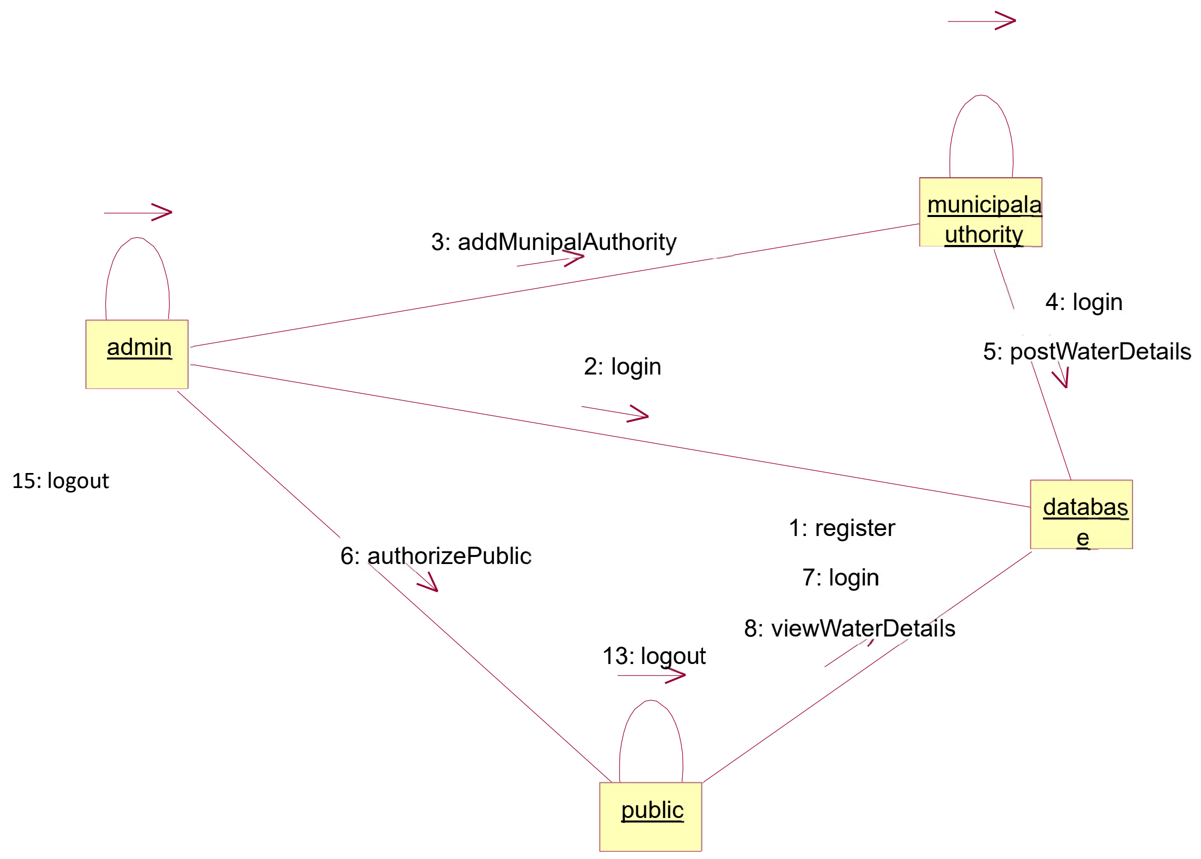
A sequence diagram is a type of Unified Modeling Language (UML) diagram that visualizes the interactions and communication among objects or components in a system over a specific period of time. Sequence diagrams are particularly useful for modeling the dynamic aspects of a system, showing how objects or actors collaborate to achieve a particular functionality or scenario.



6.2.4 COLLABORATION DIAGRAM

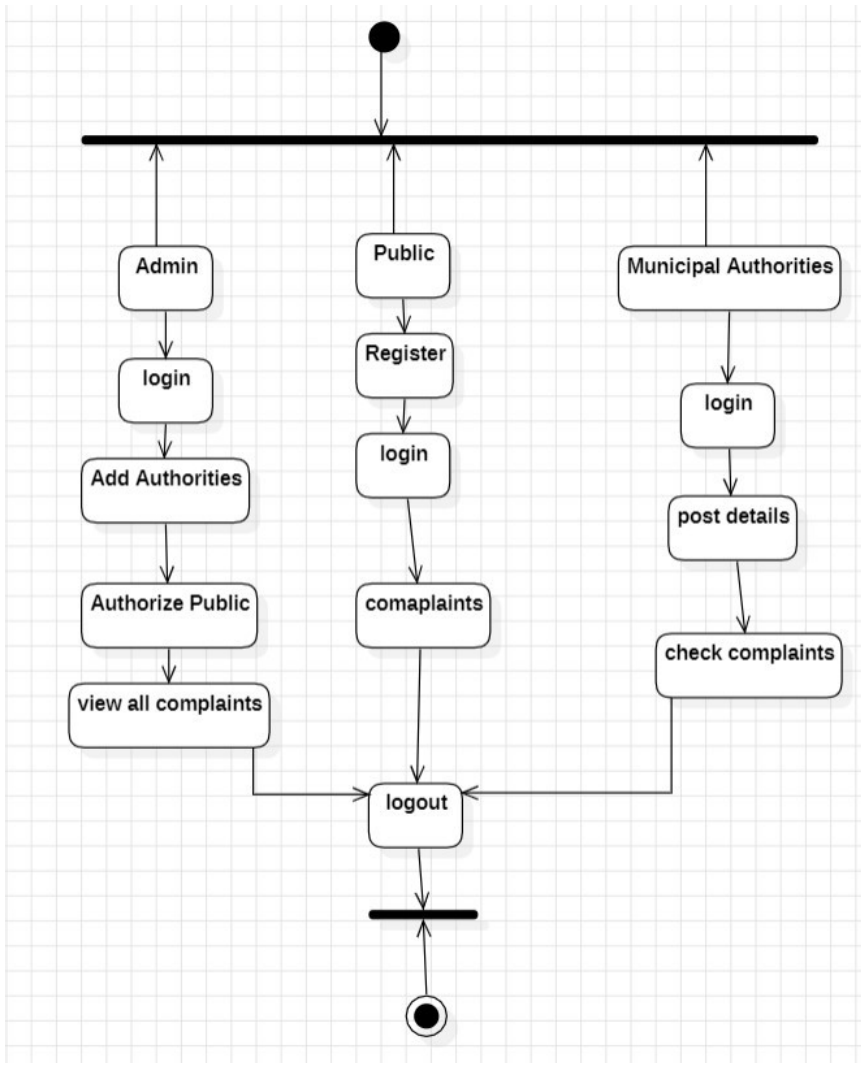
Collaboration diagrams, also known as communication diagrams, are a type of Unified Modeling Language (UML) diagram that focus on visualizing the interactions and collaborations among objects or components in a system. Collaboration diagrams provide an alternative view to sequence diagrams for understanding the dynamic behavior of a system, emphasizing the relationships between objects and their interactions.

14: logut



6.2.5 ACTIVITY DIAGRAM

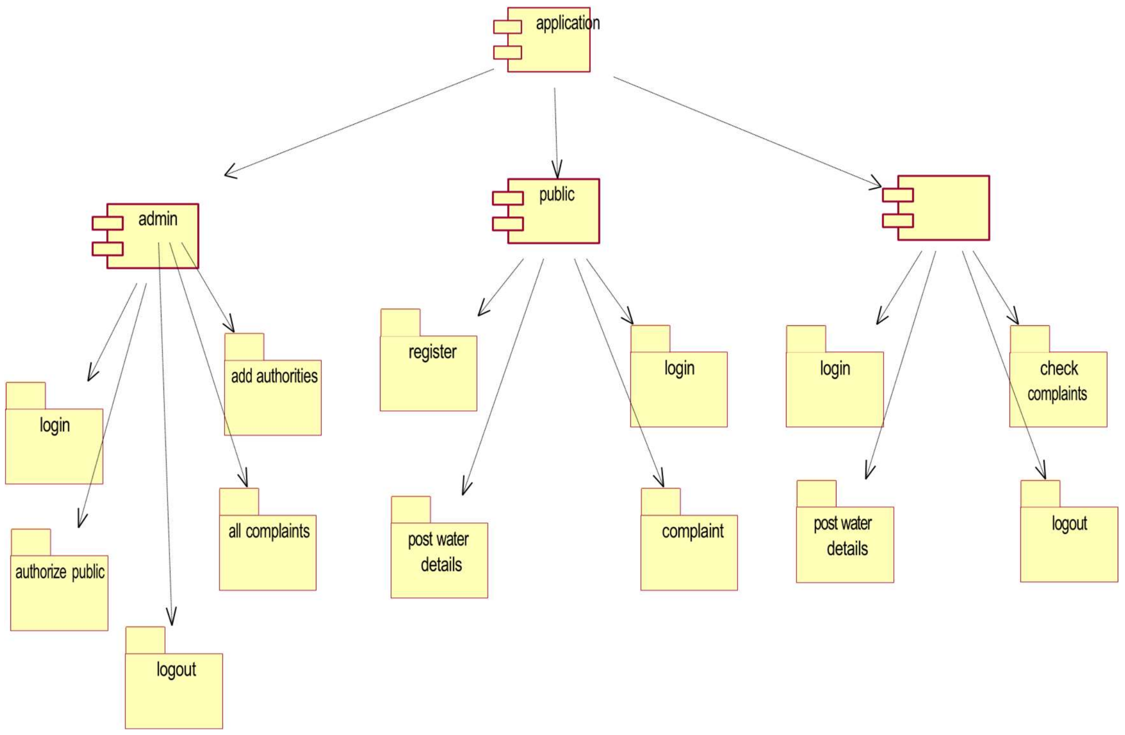
Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe the business and operational step-by-step workflows of components in a system. An activity diagram shows the overall flow of control.



6.2.6 COMPONENT DIAGRAM

A component diagram is used to break down a large object-oriented system into the smaller components, so as to make them more manageable. It models the physical view of a system suchas executables, files, libraries, etc. that resides within the node.

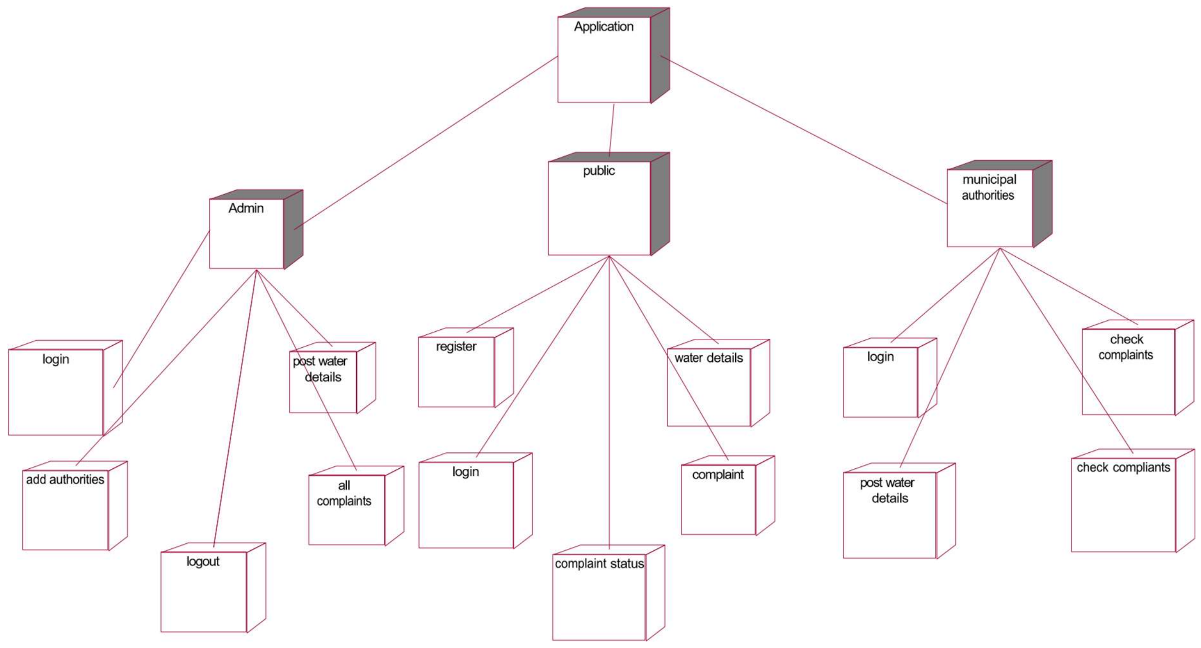
It visualizes the relationships as well as the organization between the components present in the system. It helps in forming an executable system. A component is a single unit of the system, which is replaceable and executable. The implementation details of a component are hidden, andit necessitates an interface to execute a function. It is like a black box whose behavior is explainedby the provided and required interfaces.



6.2.7 DEPLOYMENT DIAGRAM

A deployment diagram is a UML diagram type that shows the execution architecture of a system, including nodes such as hardware or software execution environments, and the middleware connecting them.

Deployment diagrams are typically used to visualize the physical hardware and software of a system. Using it you can understand how the system will be physically deployed on the hardware.Deployment diagrams help model the hardware topology of a system compared to other UML diagram types which mostly outline the logical components of a system.



DATA DICTIONARY

UREG (PUBLIC REGISTRATION)

Column	**Datatype**	**Constraints**	**Description**
Id	int	PRIMARY KEY	To enter id
Name	Varchar (100)	NOTNULL	To mention name
E-mail	Varchar (100)	NOTNULL	To enter E-mail
Mobile	int	NOTNULL	To enter mobile
Location	Varchar (100)	NOTNULL	To enter location
Area	Varchar (100)	NOTNULL	To enter area
Username	Varchar (100)	NOTNULL	To enter username
Password	Varchar (100)	NOTNULL	To view password

PUBLIC LOGIN

Column	**Data Type**	**Constraints**	**Description**

username	Varchar (100)	Not Null	Username of public
password	Varchar (100)	Primary Key, Unique Value	Password of public

ADMIN LOGIN

Column	Data Type	Constraints	Description
username	Varchar (100)	Not Null	Username of admin
password	Varchar (100)	Primary Key, Unique Value	Password of admin

MUNICIPAL AUTHORITIES REGISTRATION

Column	Datatype	Constraints	Description
Id	int	PRIMARY KEY	To enter id
Name	Varchar (100)	NOTNULL	To mention name
E-mail	Varchar (100)	NOTNULL	To enter E-mail
Mobile	int	NOTNULL	To enter mobile
Location	Varchar (100)	NOTNULL	To enter location
Area	Varchar (100)	NOTNULL	To enter area
Username	Varchar (100)	NOTNULL	To enter username
Password	Varchar (100)	NOTNULL	To view password

MUNICIPAL AUTHORITIES LOGIN

Column	**Data Type**	**Constraints**	**Description**
username	Varchar (100)	Not Null	Username of municipal authorities
password	Varchar (100)	Primary Key, Unique Value	Password of municipal authorities

SYSTEM IMPLEMENTATION

5.1 Organization Impact

Identification and measurement of benefits to the organization in such areas as financial concerns operational efficiency and competitive impact on the internal and external information flows.

5.2 User Manager Assessment

Evaluation of the development process on accordance with such yardsticks as overall development time and effort conformance to budgets and standards and other project management criteria, includes assessment of development and tools.

Unfortunately system evaluation does not always receive the annotation it merits were properly managed. However it provides a great deal of information that can improve the effectiveness of subsequent application.

Implementation

Implementation is the stage where the theoretical design is turned in to working system. The most crucial stage is achieving a new successful system and in giving confidence on the new system for the users that it will work efficiently and effectively.

The system can be implemented only after through testing is done and if it found to work according to the specification. It involves careful planning, investigation of the current system and its constraints on implementation, design of methods to achieve the changeover and an evaluation of change over methods a part from planning. Two major tasks of preparing the implementation are education and training of the users and testing of the system.

The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation. The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

Implementation is the process of having systems personnel check out and put new equipment in to use, train users, install the new application, and construct any files of data needed to it.

Depending on the size of the organization that will be involved in using the application and the risk associated with its use, system developers may choose to test the operation in only one area of the firm, say in one department or with only one or two persons. Sometimes they will run the old and new systems together to compare the results. In still other situation, developers will stop using the old system one-day and being using the new one the next. As we will see, each implementation strategy has its merits, depending on the business situation in which it is considered. Regardless of the implementation strategy used, developers strive to ensure that the system's initial use in trouble-free   The more complex the system being implemented, the more involved will be the systems analysis and design effort required just for implementation. The implementation phase comprises of several activities. The required hardware and software acquisition is carried out. The system may require some software to be developed. For this, programs are written and tested. The user then changes over to his new fully tested system and the old system is discontinued.

Implementation is the process of having systems personnel check out and put new equipment in to use, train users, install the new application, and construct any files of data needed to it.

Depending on the size of the organization that will be involved in using the application and the risk associated with its use, system developers may choose to test the operation in only one area of the firm, say in one department or with only one or two persons. Sometimes they will run the old and new systems together to compare the results. In still other situation, developers will stop using the old system one-day and being using the new one the next. As we will see, each implementation strategy has its merits, depending on the business situation in which it is considered. Regardless of the implementation strategy used, developers strive to ensure that the system's initial use in trouble-fre

1 SOURCE CODE

SOURCE CODE

INDEX CODE

```
<!DOCTYPE html>
```

```
<html lang="en">

<head>

<meta charset="utf-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-scale=1">

<title> HYDROHUB

</title>

<!--

Pipeline

http://www.templatemo.com/tm-496-pipeline

-->

<!-- load stylesheets -->

<link rel="stylesheet" href="//fonts.googleapis.com/css?family=Open+Sans:300,400"> <!-- Google web
font "Open Sans", https://fonts.google.com/ -->

<link rel="stylesheet" href="font-awesome-4.6.3/css/font-awesome.min.css"> <!-- Font Awesome,
http://fontawesome.io/ -->
<link     rel="stylesheet"     href="css/bootstrap.min.css"><!-- Bootstrap style, http://v4-
alpha.getbootstrap.com/ -->

<link rel="stylesheet" href="css/magnific-popup.css">    <!-- Magnific pop up style,
http://dimsemenov.com/plugins/magnific-popup/ -->
<link rel="stylesheet" href="css/templatemo-style.css"> <!-- Templatemo style -->

<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries -->

<!-- WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>

<script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>

<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>

<![endif]-->

</head>

<body>

<div class="container-fluid">

<section id="welcome" class="tm-content-box tm-banner margin-b-10">

<div class="tm-banner-inner">

<h1 class="tm-banner-title">

HYDROHUB: YOUR SOURCE FOR WATER INFORMATION
```

```
</h1>

}

<tr><th>EMAIL<th><td><input type="text" name="email" value="<%=email%>"></td></tr>

<tr><th>LOCATION<th><td><input type="text" name="location" value="<%=location%>"></td></tr>

<tr><th>AREA<th><td><input type="text" name="area" value="<%=area%>"></td></tr>

<tr><th>FROM DATE<th><td><input type="date" name="from" required=""></td></tr>

<tr><th>TO DATE<th><td><input type="date" name="to" required=""></td></tr>

<tr><th>SESSION<th><td><input type="text" name="session" required=""></td></tr>

<tr><th>HOURS<th><td><input type="text" name="hours" required=""></td></tr>

<tr><th><th><td><input type="submit" value="ADD DETAILS"></td></tr>

</table>

</form>

</center>

</div>

<div id="gallery" class="tm-content-box" style="margin- bottom:500px;">

</div>

</body>

</html>
```

POST ACTION CODE:

```
String email=request.getParameter("email");

String location=request.getParameter("location"); String area=request.getParameter("area"); String
from=request.getParameter("from"); String to=request.getParameter("to");

String ses=request.getParameter("session"); String hour=request.getParameter("hours");


try{

Connection con=DBCon.getCon();

Statement pst=con.createStatement(); ResultSet r=null; PreparedStatement st=null;

st=con.prepareStatement("insert into water values(null,?,?,?,?,?,?,?)");

st.setString(1,email); st.setString(2,location); st.setString(3,area); st.setString(4,from); st.setString(5,to);

st.setString(6,ses); st.setString(7,hour); int i=st.executeUpdate(); if(i>0){

response.sendRedirect("PostDetails.jsp?msg=success");

}else{ response.sendRedirect("PostDetails.jsp?msg=failed");
}
}catch(Exception e){ out.println(e);
}
</html>
```

COMPLAINT STATUS CODE:

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1">
<title>HYDROHUB</title>
<!-- load stylesheets -->
<link rel="stylesheet" href="//fonts.googleapis.com/css?family=Open+Sans:300,400"> <!-- Google web
font "Open Sans", https://fonts.google.com/ -->
<link rel="stylesheet" href="font-awesome-4.6.3/css/font-awesome.min.css"> <!-- Font Awesome,
http://fontawesome.io/ -->
<link     rel="stylesheet"     href="css/bootstrap.min.css"><!-- Bootstrap style, http://v4-
alpha.getbootstrap.com/ -->
<link rel="stylesheet" href="css/magnific-popup.css">    <!-- Magnific pop up style,
http://dimsemenov.com/plugins/magnific-popup/ -->
<link rel="stylesheet" href="css/templatemo-style.css"> <!-- Templatemo style -->
<link rel="stylesheet" href="tablestyle.css">
<!-- HTML5 shim and Respond.js for IE8 support of HTML5 elements and media queries --> <!--
WARNING: Respond.js doesn't work if you view the page via file:// -->

<!--[if lt IE 9]>
<script src="https://oss.maxcdn.com/html5shiv/3.7.2/html5shiv.min.js"></script>
<script src="https://oss.maxcdn.com/respond/1.4.2/respond.min.js"></script>
<![endif]-->
</head>
<body>
<div class="container-fluid">
```

```
<section id="welcome" class="tm-content-box tm-banner margin-b-10">
<div class="tm-banner-inner">
<h1 class="tm-banner-title">HYDROHUB: YOUR SOURCE FOR WATER INFORMATION</h1>
</div>
</section>
<div class="tm-body">
<div class="tm-sidebar">
<nav class="tm-main-nav">
<ul class="tm-main-nav-ul">
<li class="tm-nav-item"><a href="User_Home.jsp" class="tm-nav-item-link
```

3. SYSTEM TESTING

6.1. Software Testing Testing

- The process of executing a system with the intent of finding an error.
- Testing is defined as the process in which defects are indentified, isolated, subjected for rectification and ensured that product is defect free in order to produce the quality product and hence customer satisfaction.
- Quality is defined as justification of the requirements
- Defect is nothing but deviation from the requirements
- Defect is nothing but bug.

- Testing---The presence of bugs
- Testing can demonstrate the presence of bugs, but not their absence
- Debugging and Testing are not the same thing!
- Testing is a systematic attempt to break a program or the AUT
- Debugging is the art or method of uncovering why the script/program did not execute properly.

6.2 Testing Approaches

Testing can be done in two ways

- Bottom up approach
- Top down approach

Bottom up Approach

Testing can be performed starting from smallest and lowest level modules and proceeding ne at a time. For each module in bottom up testing a short program executes the module and provides the needed data so that the module is asked to perform the way it will when embedded within the larger system. When bottom level modules are tested attention turns to those on the next level that use the lower level ones they are tested individually and then linked with the previously examined lower level modules.

Top down Approach

This type of testing starts from upper level modules. Since the detailed activities usually performed in the lower level routines are not provided stubs are written. A stub is a module shell called by upper level module and that when reached properly will return a message to calling module indicating that proper interaction occurred. No attempt is made to verify the correctness of the lower level module. The system has been tested and implemented successfully and thus ensured that all the requirements as listed in the software requirements specification are completely fulfilled. In case of erroneous input corresponding error messages are displayed.

6.3 Testing Methods

1. Black box or functional testing
2. White box testing or structural testing

Black Box Testing

This method is used when knowledge of the specified function that a product has been designed to perform is known. The concept of black box is used to represent a system whose inside workings are not available to inspection. In a black box the test item is a ”Black”, since its logic is unknown, all that is known is what goes in and what comes out, or the input and output.

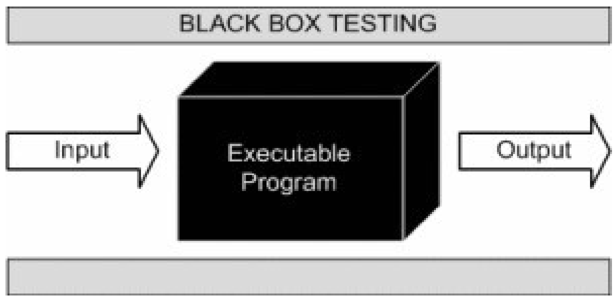
Black box testing attempts to find errors in the following categories:

a. Incorrect or missing functions
b. Interface errors
c. Errors in data structure
d. Performance errors
e. Initialization and termination errors

As shown in the following figure of Black box testing, we are not thinking of the internal workings, just we think about

What is the output to our system?

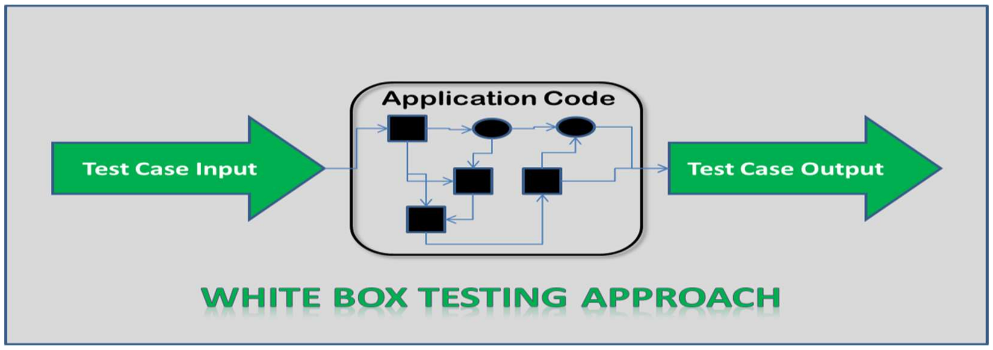
What is the output for given input to our system?



The Black box is an imaginary box that hides its internal workings

White Box Testing

White box testing is concerned with testing the implementation of the program. The intent of structural is not to exercise all the inputs or outputs but to exercise the different programming and data structure used in the program. Thus structural testing aims to achieve test cases that will force the desire coverage of different structures. Two types of path testing are statement testing coverage and branch testing coverage.



The white box testing strategy, the internal workings

6.4. Testing Types

Different levels of testing are used in the testing process, each level of testing aims to test different aspects of the system. The basic levels are:

- Unit testing
- Integration testing
- Functionality testing
- System testing
- User Acceptance testing

Unit Testing

Unit testing focuses on the building blocks of the software system, that is, objects and sub system. There are three motivations behind focusing on components. First, unit testing reduces the complexity of the overall tests activities, allowing us to focus on smaller units of the system. Second, unit testing makes it easier to pinpoint and correct faults given that few components are involved in this test. Third, unit testing allows parallelism in testing activities that is each component can be tested independently of one another. Hence the goal is to test the internal logic of module.

Integration Testing

In the integration testing, many test modules are combined into sub systems, which are then tested. The goal here is to see if the modules can be integrated properly, the emphasis being on testing module interaction.

After structural testing and functional testing we get error free modules. These modules are to be integrated to get the required results of the system. After checking a module, another module is tested and is integrated with the previous module. After the integration, the test cases are generated and the results are tested.

Functionality Testing

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals. Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted. Invalid Input : identified classes of invalid input must be rejected. Functions : identified functions must be exercised.

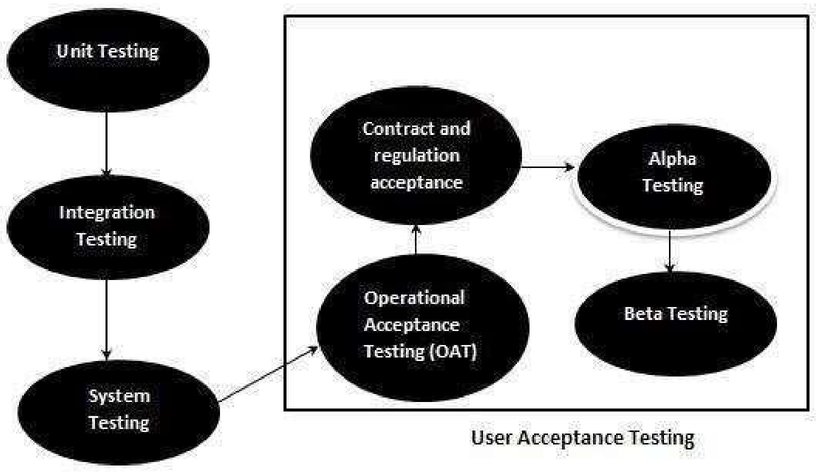
Output : identified classes of application outputs must be exercised. Systems/Procedures: interfacing systems or procedures must be involve Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Testing

In system testing the entire software is tested. The reference document for this process is the requirement document and the goal is to see whether the software meets its requirements. The system was tested for various test cases with various inputs.

User Acceptance Testing

Acceptance testing is sometimes performed with realistic data of the client to demonstrate that the software is working satisfactory. Testing here focus on the external behavior of the system, the internal logic of the program is not emphasized. In acceptance testing the system is tested for various inputs.



Testing is the process of finding difference between the expected behavior specified by system models and the observed behavior of the implemented syste

TEST CASES

Test case1:

Test case for Login form:

FUNCTION:	ADMIN
EXPECTED RESULTS:	Should Validate the user and check his existence in database
ACTUAL RESULTS:	Validate the user and checking the user against the database
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case2:

Test case for User Registration form:

FUNCTION:	PUBLIC
EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case3:

Test case for User Registration form:

FUNCTION:	MUNICIPAL AUTHORITIES

EXPECTED RESULTS:	Should check if all the fields are filled by the user and saving the user to database.
ACTUAL RESULTS:	Checking whether all the fields are field by user or not through validations and saving user.
LOW PRIORITY	No
HIGH PRIORITY	Yes

Test case3:

Test case for Change Password:

When the old password does not match with the new password, then this results in displaying anerror message as “OLD PASSWORD DOES NOT MATCH WITH THE NEW PASSWORD”.

Test case 4:

Test case for Forget Password:

When a user forgets his password, he is asked to enter Login name, ZIP code, Mobile number. If these are matched with the already stored ones then user will get his original password.

Module	**Functionality**	**Test Case**	**Expected Results**	**Actual Results**	**Resut**	**Prioriy**

User	Login Use case	1. Navigate To Www.Sample.Com 2. Click On Submit Button Without Entering Username and Password	A Validation Should Be as Below "Please Enter Valid Username & Password"	A Validation Has Been Populated as Expected	Pass	High

User	Login Use case	1. Navigate To Www.Sample. Com 2. Click On Submit Button with Out Filling Password AndWith Valid Username	A Validation Should Be as Below “Please Enter Valid Password or Password Field CanNot Be Empty “	A Valida tion Is Shown as Expected	Pass	High
User	Login Use case	1. Navigate To Www.Sample. Com 2. Enter Both Username and Password Wrong And Hit Enter	A Validation Shown as Below “The Username Entered Is Wrong”	A Valida tion Is Shown as Expec ted	Pass	High

User	Login Use case	1. Navigate To Www.Sample. Com 2. Enter Validate Username and Password And Click On Submit	Validate Usernameand Password In Database And Once If They Correct Then ShowThe Main Page	Main Page/ Home PageH as Been Displayed	Pass	High

CONCLUSION

1. A web portal for safe drinking water is an essential resource for ensuring the well-being and health of communities around the world. In conclusion, such a portal plays a crucial role in addressing various aspects related to safe drinking water, including information dissemination, access to critical resources, and raising awareness. It serves as a centralized platform for:
2. Information Dissemination: Providing comprehensive information on water quality standards, water treatment processes, and water-related regulations to educate the public and empower them with knowledge about safe drinking water.
3. Access to Resources: Offering access to resources such as water quality reports, testing kits, and contact information for relevant authorities, enabling individuals and communities to monitor and address water quality concerns effectively.
4. Community Engagement: Facilitating community engagement through forums, discussion boards, and educational materials to foster a sense of shared responsibility for safe drinking water and encourage collaboration on water-related issues.
5. Emergency Response: Providing real-time updates on water quality in case of emergencies, natural disasters, or contamination incidents, helping communities take immediate action to safeguard their health.

In summary, a web portal for safe drinking water is a vital tool in the effort to ensure that people have access to clean and safe water. It promotes transparency, awareness, and community involvement, all of which are crucial in maintaining and improving the quality of drinking water, thereby contributing to the health and well-being of individuals and communities.

FUTURE ENHANCEMENT

The future scope of a web portal for safe drinking water is filled with potential for addressing global water quality challenges and improving access to clean and potable water. Here are some key areas of development and opportunities for such a portal:

1. Data Analytics and AI Integration: Implementing advanced data analytics and artificial intelligence (AI) can help predict and detect water quality issues in real- time. AI algorithms can analyse data from

various sources, such as sensors, environmental data, and water quality reports, to identify trends and anomalies, allowing for more proactive responses to water contamination and quality concerns.

2. Mobile Applications and IoT Devices: Developing mobile applications and integrating Internet of Things (IoT) devices can make it easier for individuals to monitor their own water quality and contribute to a broader database. Users can use smartphone apps to check water quality, report issues, and receive alerts, creating a more participatory and engaged user base.
3. Geographic Information System (GIS) Mapping: Implementing GIS technology can help users visualize water quality data on interactive maps. This feature can be particularly useful for identifying areas with poor water quality, tracking pollution sources, and planning infrastructure improvements.
4. Water Quality Certification and Standards Tracking: Creating a section on the portal dedicated to tracking and updating water quality certifications, standards, and regulations can help users stay informed about changes and ensure compliance with the latest safety guidelines.
5. Collaboration with NGOs and Government Bodies: Collaborating with non- governmental organizations (NGOs), governmental agencies, and water experts can enhance the portal's credibility and effectiveness. These partnerships can help with data collection, quality assessment, and policy advocacy.
6. Water Quality Testing Kits: Offering affordable and easy-to-use water quality testing kits through the portal can empower individuals to assess their own water quality and contribute to a comprehensive database. These kits can be integrated with the portal, allowing users to input their test results for analysis.
7. Educational Resources: Expanding the educational resources on the portal to include interactive content, virtual tours of water treatment facilities, and case studies on successful water quality improvement initiatives can enhance user engagement and awareness.
8. Global Expansion: Scaling the portal to serve communities worldwide can have a significant impact on ensuring safe drinking water in regions facing water quality challenges. Translating content into multiple languages and customizing resources for specific regions will be crucial.

9. Water Conservation and Sustainability: Integrating content on water conservation, sustainability practices, and eco-friendly water treatment methods can align the portal with broader environmental goals and support long-term water resource management.

10. Disaster Preparedness and Response: Enhancing the portal's capabilities for disaster preparedness and response by providing emergency guidelines, resources, and real- time updates during natural disasters or contamination incidents is crucial for safeguarding public health.

The future scope of a web portal for safe drinking water is promising, with opportunities for leveraging technology, data, and community involvement to improve water quality, accessibility, and public health. By staying adaptable and continuously innovating, such a portal can play a significant role in addressing the evolving challenges of water quality and safety

7. BIBLIOGRAPHY

1. System analysis and Design - Alan Dennis, Barbara and Wixom, Roberta M. Roth
2. Software Engineering – A Practitioner's Approach- Roger SPressman.
3. Software Engineering- A precise Approach – Pankaj Jalote –2010
4. The Unified Software Development Process – Ivor Jacbson, Grady Booch, James Rumbaugh, Pearson Education, India, 2008.
5. Database Management Systems - Raghu Ramakrishnan, and Johannes Gehrke.
6. Patidar Vishnu & Kadam Vishal (2016). Analysis Process of Design and Development of Online Examination System, IJIRCCE, Vol. 4, Issue 2
7. Kotwal Deepankar Vishwas, Bhadke Shubham Rajendra, Gunjal Aishwarya Sanjay & Biswas Pushpendu (2016). Online Examination System, IRJET, Vol. 3, Issue 6
8. Huiping Wang, RuowuZhong Department of Computer and Information Design and Implementation of Online Exam System Based on Data Mining, IEEE paper.
9. Wang Jingjing, Department of Computer and Information, Research On Intelligent Test Paper Of WEB-Based, IEEE paper.
10. Nijaz. (2000) Dynamic Web-based Application Development. New York:Prentice

10 GANT CHART

S.NO	TASK	Week 1, 2 Apr 15- May 03	Week 3,4 May 04- May 17	Week 5 May 18-May 24	Week 6 May 25- May 31	Week 7 Jun 01 - Jun 09	Week 8 Jun 10 - Jun 16	Week 9,10 Jun 17- Jun 28	Week 11 Jun 29- Jul 05	Week 12 Jul 06- Jul 12	Week 13, 14 Jul 13- Jul 22	Week 15 Jul 23- Jul 27
1	**REQURIMENT ANALYSIS**											
	• Fact Finding	🟥										
	• Requirements specification		🟥									
2	**DESIGN**											
	• Class			🟨								
	• Use case			🟨								
	• Sequence			🟨								
	• Activity			🟨	🟨							
	• Architecture Diagram				🟨							
	• ER Diagram				🟨							
	• Data Dictionary				🟨	🟨						
3	**CODING**											
	• Admin						🟩					
	• Municipal Authorities						🟩	🟩				
	• Public							🟩	🟩			
4	**TESTING**											
	• Unit Testing									■		
	• Integration Testing									■		
	• Functionality Testing									■		
	• System Testing										■	
	• User Acceptance Testing										■	
5	**DOCUMENTATION**								🟦	🟦	🟦	🟦