

INTRODUCTION

Real estate is considered real property that includes land and anything permanently attached to it or built on it whether natural or manmade. A type of real property with tangible assets. This includes land, buildings on the land, and improvements to the land, such as a roadway or private well. You might want to invest in real estate because of its potential for an excellent return. It's a much less volatile investment than the stock market. And it has real value. Real estate always has value, unlike stocks which can sink to pennies on the dollar. It covers residential housing, commercial offices, and trading spaces such as theatre, hotels, and restaurant retail outlets, industrial buildings such as factories and government buildings.

The real estate market is characterized by diverse segments, including residential, commercial, industrial, and agricultural properties. Residential real estate caters to housing needs, ranging from single-family homes to apartments and condominiums. Commercial real estate comprises office buildings, retail spaces, hotels, and warehouses, facilitating business activities and commerce. Industrial real estate encompasses factories, manufacturing plants and logistics centers



LITERATURE SURVEY

TITLE: Research and Development of Real Estate Information Registration Platform
Based on Web GIS

AUTHOR: Xiaoli Shan, Chengxia Zhang, Mingyu Dong

ABSTRACT:

For the purpose of managing the registration process and results, this paper uses Web GIS to develop the real estate information registration platform to realize the functions of user login, data query, data editing, data output and data analysis, meet the work requirements of real estate registration, and help the registration department to perform the unified registration responsibilities of real estate. It effectively solves the difficult problem of information integration after the integration of registration institutions, and highlights the accuracy, openness and ease of use of the platform, which brings great convenience to the real estate registration in Qingzhou area.

TITLE: Design and application of real estate market monitoring platform based on spatio-temporal big data

AUTHOR: YinXue Huang

ABSTRACT:

When the traditional visualization application platform visually analyzes large-scale real estate spatio-temporal data, there are problems such as slow loading and poor functional scalability. In order to solve these problems, this paper proposes a design scheme of a visual analysis platform for real estate market monitoring. First, carry out conceptual analysis and feature induction of spatio-temporal big data to realize the collection processing and organizational structure analysis of real estate spatio-temporal big data. Then a four-tier platform architecture consisting of data sources layer, data resource management layer, interface service engine layer and visual application layer is designed. The visualization platform designed by this scheme has good scalability, and can provide technical guidance for the construction of a visualization analysis platform in the field of industry spatio-temporal big data. Finally, a visual application design is carried out based on the real estate market data of Hubei Province, which proves the validity and feasibility of the research scheme in this paper.

TITLE : Real Estate Management System of Personal Seekers for Investment Decision

AUTHOR: Waralak Chongdarakul, Nikorn Rongbuttri

ABSTRACT:

Real estate for residence consists of various types such as houses, town houses, condominiums, and vacant land. The residence for sale can be found by the buyer who searches for their preferred information through websites, social media, visiting the real places and more. To find candidate properties has been so complicated and time-consuming since they are large amount of data and many factors to be considered. The residence seeker and investors also have problems of storing plenty of real estate and organizing their property information. The complicated property information causes the seeker difficult to compare many property lists and select the best choice for investment. Many existing websites offer the real estate markets for people who looking for property investment or residence and support real estate agencies and owners to advertise their properties to the right target audiences. However, these websites do not support for personally collecting the buyers' preferred estates found from various sources i.e. property for sale at the real location, or from other web sites. As this motivation, this paper introduces the real estate management system for buyers to personally collect their property information related to the investment decision making. The mobile based application is proposed for collecting house and land data with geographical position at the real place. Meanwhile, the web application is created to handle the backend services, and provides all functions of the property management. Unlike other real estate applications, the proposed system offers features of the property information sharing as the personal group formed by investors. The evaluation of user satisfaction shows that our proposed system is practically used and satisfied.

TITLE: Study on Real Estate Project Investment Decision-Making Based on Principal Component Analysis and Adaptive Network-Based Fuzzy Inference System

AUTHOR: Hui Zhao, Xue-qing Wang

ABSTRACT:

A new method based on the integration of principal component analysis (PCA) and adaptive network-based fuzzy inference system ANFIS) is put forward for selecting the real estate project. Firstly, principal component analysis (PCA) is used to reduce the evaluation index dimensions. And then, adaptive network-based fuzzy inference system (ANFIS) is used to evaluate the real estate projects. In order to grasp this method better, finally, the paper provides a case to demonstrate the application of this method in selecting the real estate project. The case has shown that the method applied to the real estate project investment decision-making is feasible and reliable.

TITLE: Real Estate Management System based on Blockchain

AUTHOR: Ankit Mittal,Bhavyansh Sharma,Pinku Ranjan

ABSTRACT:

Real Estate management in India as well as in many parts of the world is a very inefficient process. Developing a secure central system that not only accelerates the process of land registration but also makes it efficient will be effective. This paper presents a blockchain-powered real estate management system that will provide a transparent, secure, and efficient system for real estate management. This system will include all the departments related to real estate management. It will store all the transactions on a distributed permissioned blockchain which will be very secure and will not be prone to hacking and this can be automated to a great extent. The system will be centralized connecting all the departments and will be decentralized for data storage. It is a practical solution to the real estate management problem.

EXISTING SYSTEM:

The existing system of real estate involves the buying, selling, and renting of properties such as land, buildings, and homes. It typically operates through a network of real estate agents, brokers, and agencies who facilitate transactions between buyers and sellers. Property values are determined by various factors including location, market demand, property condition, and economic trends. Legal frameworks, regulations, and financing options also play significant roles in the real estate system, which varies considerably from one country to another.

DISADVANTAGES:

- **High Cost**

The biggest disadvantage with real estate investment is the high capital requirement.

- **Long Term Investment**

Real estate investments are always made as a part of a long-term strategy.

- **Legal Difficulties**

Investing in real estate is tedious as it involves a lot of paperwork as well as cumbersome formalities, more so in the case of commercial real estate.

- **Liquidity Constraints**

Real estate property is very illiquid. A huge amount of money gets locked up as it is difficult to readily find buyers and sellers.

PROPOSED SYSTEM:

Real Estate Management System (REMS) is an online real estate software application that manages the overall operational activities and processes, starting from the management of the property, to the management of real estate agencies, agents, clients and financial transactions.

ADVANTAGES:

- **Finances Are Easier to Monitor**

With Easy estate real estate management software, you will be able to better monitor finances and they will be organized.

- **Property Management Is Efficient**

The correct software provides you with better and efficient tools to accomplish the job so that you can convince both the tenants and the buyers with agreeable deals.

- **Save Valuable Time**

Property management software is designed to save you time with features like sending automated notifications, such as when rent is due and balances.

FEASIBILITY STUDY

The feasibility of the project is analysed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

FEASIBILITY ANALYSIS

Three key considerations involved in the feasibility analysis are

ECONOMICAL FEASIBILITY

TECHNICAL FEASIBILITY

SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

SYSTEM REQUIREMENTS

FUNTIONAL REQUIREMENTS:

Hardware Requirments:

Processor : Intel core I5

RAM : 16.0 GB

Software Requirments:

Operating system : windows 11

Web Browser : Chrome and Mozilla Firefox

Languages Used :

Front End : HTML,CSS

Back End : Java

Database : SQL

SOFTWARE REQUIREMENT SPECIFICATION:

This Chapter describes about the requirements. It specifies the hardware and software requirements that are required in order to run the application properly. The Software Requirement Specification (SRS) is explained in detail, which includes overview of this dissertation as well as the functional and non-functional requirement of this dissertation.

SRS

Functional	service provider browse the file and sends to end user, router sends file from source to destination by selecting shortest path, router drops the packets at nodes if they have less energy, Auditor stores all dropped packets details, Auditor may discovers the traffic pattern details, end user receives the files from router, end users can also gets dropped packets and attacker make changes of nodes energy.
Non- Functional	The Sender and Receiver never Find the attackers.
External interface	LAN , Routers
Performance	Browse, upload, assign energy, assign distances, view distances, view energy, view files, view attackers, verify, refresh, discover traffic pattern, get dropped packets, modify energy and exit.
Attributes	Anonymous Communication, Mobile Ad Hoc Networks, Statistical Traffic Analysis, Service Provider, End Users, Attackers.

Table: 3.1 Summaries of SRS

Functional Requirements

Functional Requirement defines a function of a software system and how the system must behave when presented with specific inputs or conditions. These may include calculations, data manipulation and processing and other specific functionality. In this system following are the functional requirements:-

- The service provider browses the file and sends to the particular end users via router.
- The service provider can also assign energy and assigns distance for nodes.
- The router will send the all files from source to particular destination, by selecting shortest path & node energy.
- The packet dropper in router may drops the packets from file, suppose if node has the less energy than file size.
- The router can also view distances, view energy, view files, view attackers, verify, refresh.
- The Auditor stores the dropped packet details on it and also it can discovers the traffic pattern details.
- The Remote user has to receive the file from router by without changing the file Contents.
- If packets are dropped from file then end users will gets dropped packets from point to point manager.
- Attacker is one who makes changes the energy sizes of nodes in router.
- The Attributes are Anonymous Communication, Mobile Ad Hoc Networks, Statistical Traffic Analysis, Service Provider, End Users and Attackers.

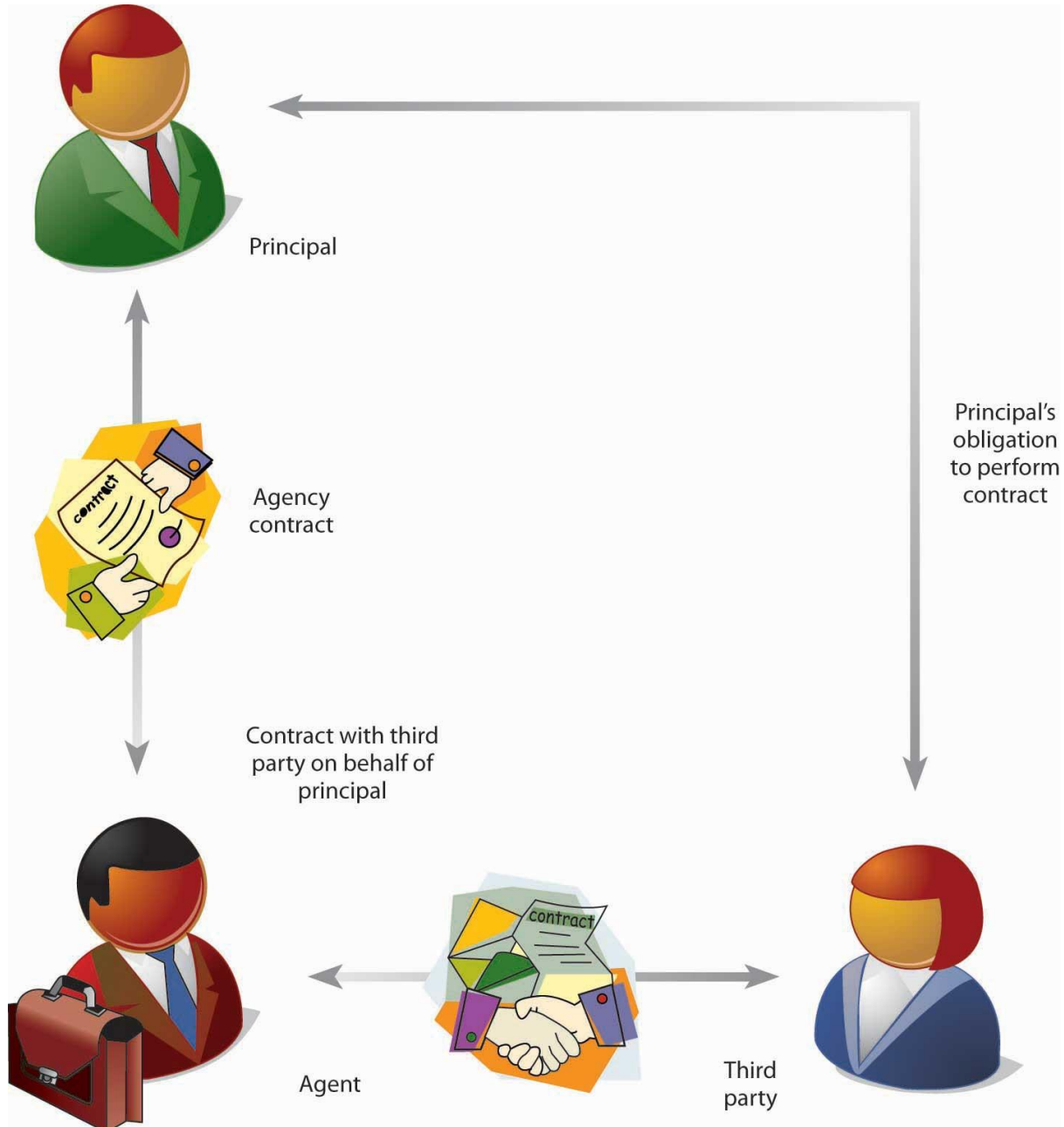
Non – Functional Requirements

Non – Functional requirements, as the name suggests, are those requirements that are not directly concerned with the specific functions delivered by the system. They may relate to emergent system properties such as reliability response time and store occupancy. Alternatively, they may define constraints on the system such as the capability of the Input Output devices and the data representations used in system interfaces. Many non-functional requirements relate to the system as whole rather than to individual system features. This means they are often critical than the individual functional requirements. The following non-functional requirements are worthy of attention.

The key non-functional requirements are:

- Security: The system should allow a secured communication between Service provider and Router and Receiver.
- Energy Efficiency: The Time consumed by the Router, transfer the File's to the Receiver.
- Reliability: The system should be reliable and must not degrade the performance of the existing system and should not lead to the hanging of the system.

SYSTEM ARCHITECTURE:



UML DIAGRAMS

UML stands for Unified Modeling Language. UML is a standardized general-purpose modeling language in the field of object-oriented software engineering. The standard is managed, and was created by, the Object Management Group.

The goal is for UML to become a common language for creating models of object-oriented computer software. In its current form UML is comprised of two major components: a Meta-model and a notation. In the future, some form of method or process may also be added to; or associated with, UML.

The Unified Modeling Language is a standard language for specifying, Visualization, Constructing and documenting the artifacts of software system, as well as for business modeling and other non-software systems.

The UML represents a collection of best engineering practices that have proven successful in the modeling of large and complex systems.

The UML is a very important part of developing objects-oriented software and the software development process. The UML uses mostly graphical notations to express the design of software projects.

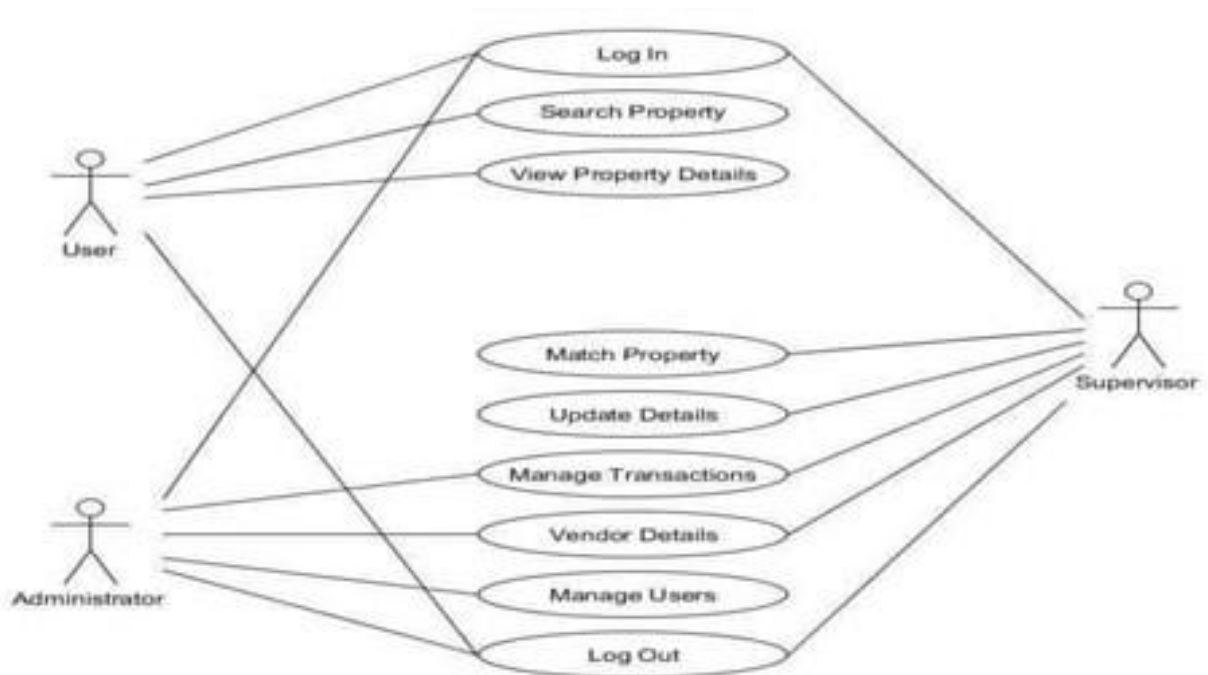
GOALS:

The Primary goals in the design of the UML are as follows:

1. Provide users a ready-to-use, expressive visual modeling Language so that they can develop and exchange meaningful models.
2. Provide extendibility and specialization mechanisms to extend the core concepts.
3. Be independent of particular programming languages and development process.
4. Provide a formal basis for understanding the modeling language.
5. Encourage the growth of OO tools market.
6. Support higher level development concepts such as collaborations, frameworks, patterns and components.
7. Integrate best practices.

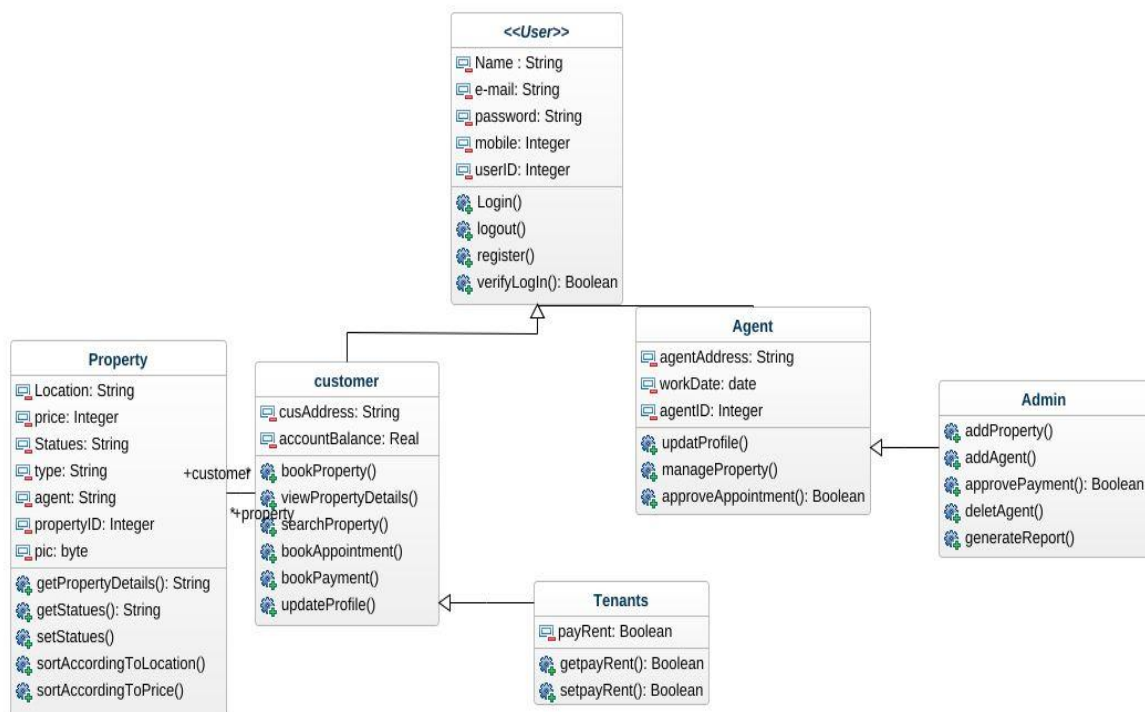
USE CASE DIAGRAM:

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.



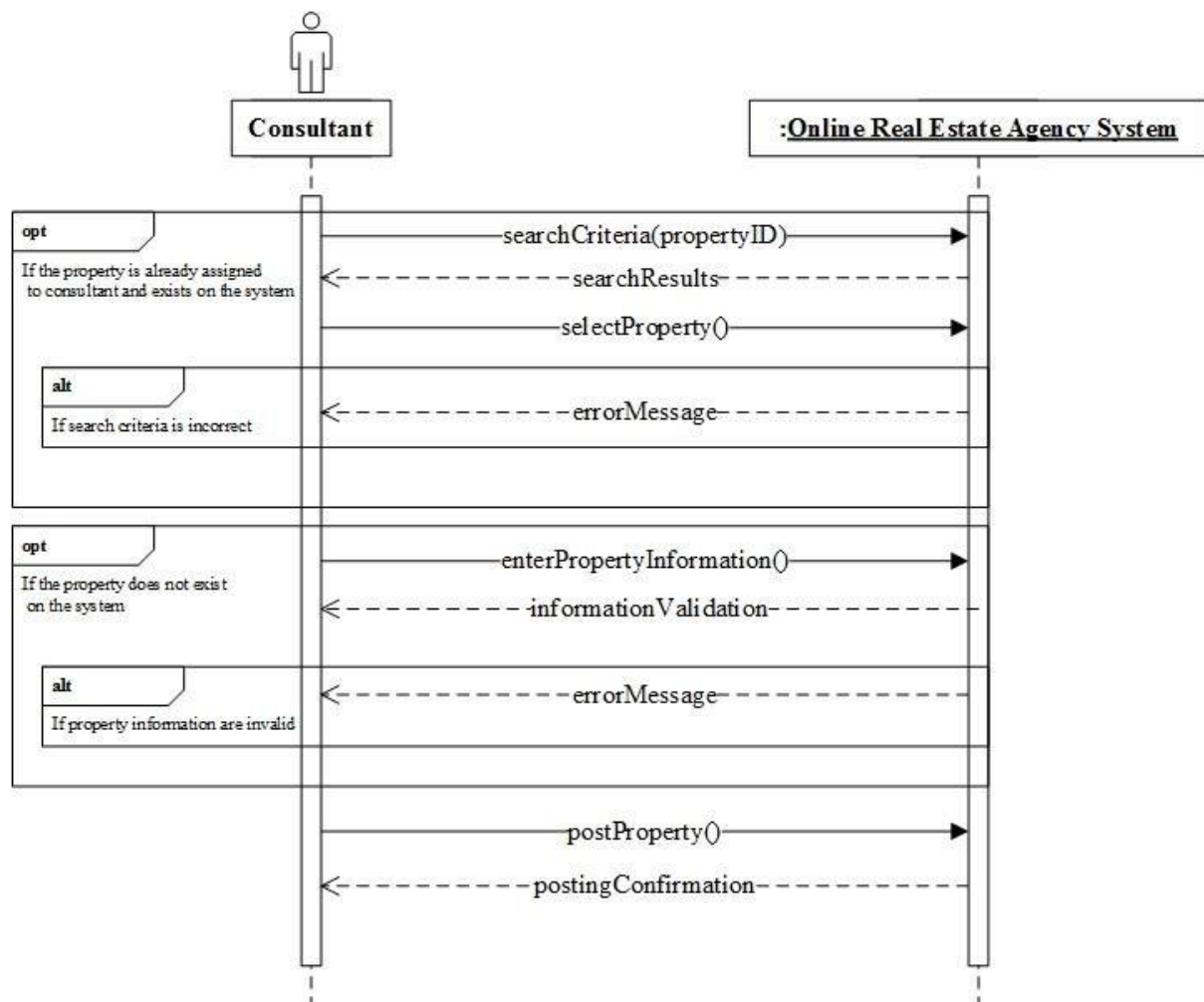
CLASS DIAGRAM:

In software engineering, a class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing the system's classes, their attributes, operations (or methods), and the relationships among the classes. It explains which class contains information. Class diagrams are a fundamental tool in object-oriented modeling and are widely used in software development.



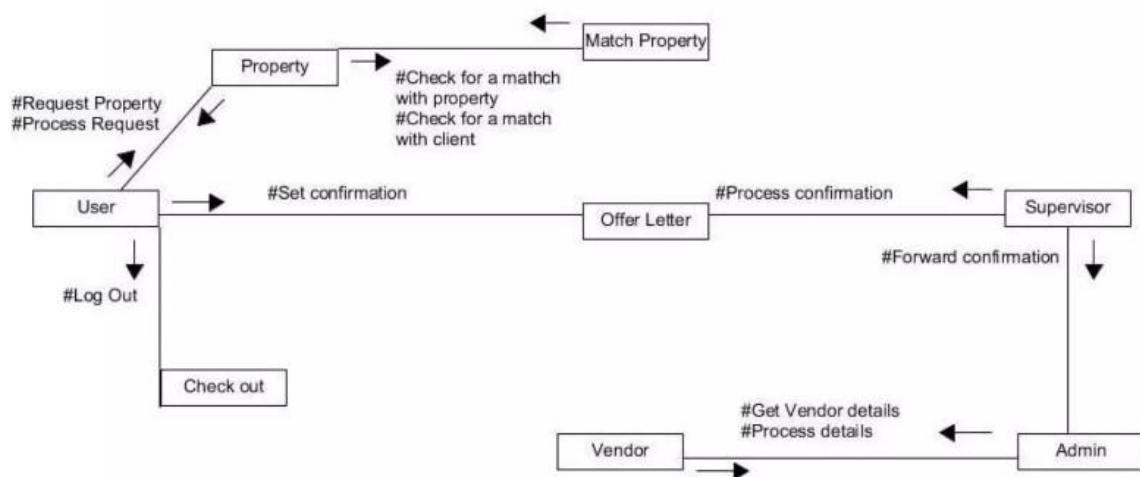
SEQUENCE DIAGRAM:

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequence diagrams are sometimes called event diagrams, event scenarios, and timing diagrams.



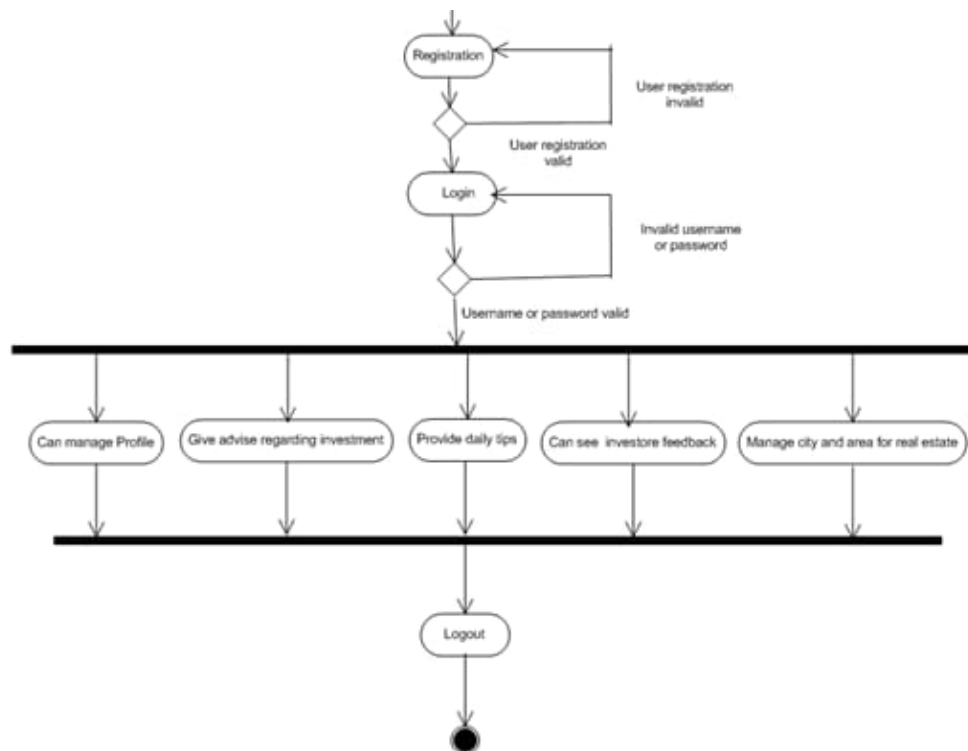
COLLABARATION DIAGRAM :

A Collaboration diagram, also known as a Communication diagram, is a type of Unified Modeling Language (UML) diagram used to visually represent the interactions and communication between objects or components in a system or software application. It focuses on showing how various objects or components collaborate to achieve specific functionality or a particular use case. Collaboration diagrams are particularly useful for understanding the dynamic aspects of a system and how objects interact over time.



ACTIVITY DIAGRAM :

An Activity diagram is a type of Unified Modeling Language (UML) diagram that is used to model the dynamic aspects of a system, particularly the flow of activities and actions within a business process, use case, or other system behaviors. Activity diagrams provide a graphical representation of how different activities and actions are organized and interact to achieve a specific goal. Activity diagrams help stakeholders, including analysts, designers, and developers, to understand the sequential and parallel flows of activities and make them a valuable tool for system design and documentation.



INPUT AND OUTPUT DESIGN

INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

- What data should be given as input?
- How the data should be arranged or coded?
- The dialog to guide the operating personnel in providing input.
- Methods for preparing input validations and steps to follow when error occur.

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user

will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output, they should Identify the specific output that is needed to meet the requirements.

2. Select methods for presenting information.

3. Create document, report, or other formats that contain information produced by the system.

The output form of an information system should accomplish one or more of the following objectives.

- ❖ Convey information about past activities, current status or projections of the
- ❖ Future.
- ❖ Signal important events, opportunities, problems, or warnings.
- ❖ Trigger an action.
- ❖ Confirm an action.

IMPLEMENTATION

MODULES ON REAL ESTATE

Residential Real Estate: This involves properties intended for private living, such as single-family homes, apartments, condominiums, and townhouses.

Commercial Real Estate: This sector involves properties used for business purposes, including office buildings, retail spaces, industrial warehouses, and hotels.

Industrial Real Estate: Industrial real estate includes properties used for manufacturing, distribution, storage, and research purposes, such as factories, warehouses, and industrial parks.

Land Development: Land development involves acquiring, improving, and subdividing land for various purposes, such as residential, commercial, or mixed-use developments.

SOFTWARE ENVIRONMENT

Java Technology

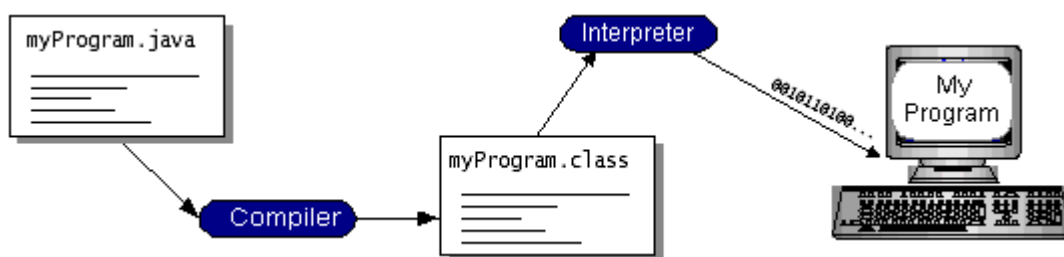
Java technology is both a programming language and a platform.

The Java Programming Language

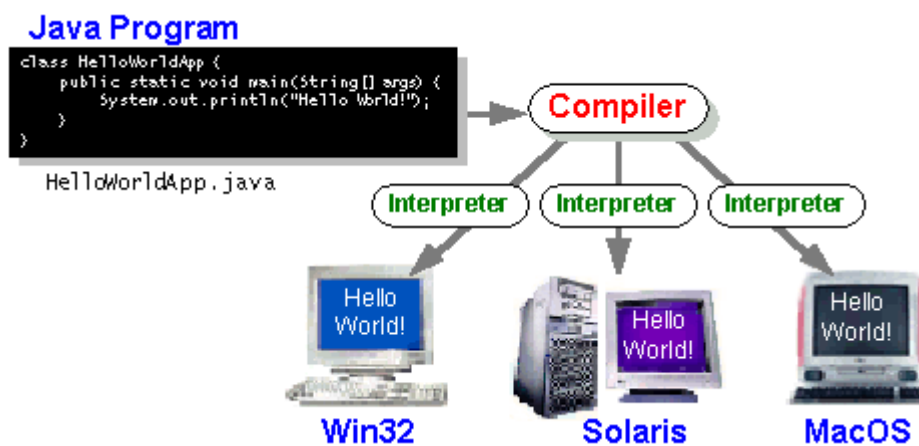
The Java programming language is a high-level language that can be characterized by all of the following buzzwords:

- Simple
- Architecture neutral
- Object oriented
- Portable
- Distributed
- High performance
- Interpreted
- Multithreaded
- Robust
- Dynamic
- Secure

With most programming languages, you either compile or interpret a program so that you can run it on your computer. The Java programming language is unusual in that a program is both compiled and interpreted. With the compiler, first you translate a program into an intermediate language called *Java byte codes* —the platform-independent codes interpreted by the interpreter on the Java platform. The interpreter parses and runs each Java byte code instruction on the computer. Compilation happens just once; interpretation occurs each time the program is executed. The following figure illustrates how this works.



You can think of Java byte codes as the machine code instructions for the *Java Virtual Machine* (Java VM). Every Java interpreter, whether it's a development tool or a Web browser that can run applets, is an implementation of the Java VM. Java byte codes help make “write once, run anywhere” possible. You can compile your program into byte codes on any platform that has a Java compiler. The byte codes can then be run on any implementation of the Java VM. That means that as long as a computer has a Java VM, the same program written in the Java programming language can run on Windows 2000, a Solaris workstation, or on an iMac.



The Java Platform

A *platform* is the hardware or software environment in which a program runs. We've already mentioned some of the most popular platforms like Windows 2000, Linux, Solaris, and MacOS. Most platforms can be described as a combination of the operating system and hardware. The Java platform differs from most other platforms in that it's a software-only platform that runs on top of other hardware-based platforms.

The Java platform has two components:

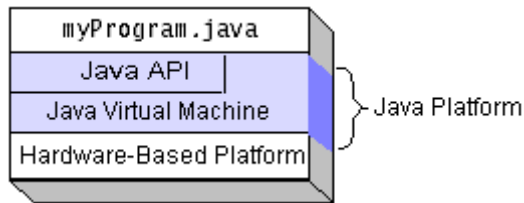
- The *Java Virtual Machine* (Java VM)
- The *Java Application Programming Interface* (Java API)

You've already been introduced to the Java VM. It's the base for the Java platform and is ported onto various hardware-based platforms.

The Java API is a large collection of ready-made software components that provide many useful capabilities, such as graphical user interface (GUI) widgets. The Java API is grouped

into libraries of related classes and interfaces; these libraries are known as *packages*. The next section, What Can Java Technology Do? Highlights what functionality some of the packages in the Java API provide.

The following figure depicts a program that's running on the Java platform. As the figure shows, the Java API and the virtual machine insulate the program from the hardware.



Native code is code that after you compile it, the compiled code runs on a specific hardware platform. As a platform-independent environment, the Java platform can be a bit slower than native code. However, smart compilers, well-tuned interpreters, and just-in-time byte code compilers can bring performance close to that of native code without threatening portability.

What Can Java Technology Do?

The most common types of programs written in the Java programming language are *applets* and *applications*. If you've surfed the Web, you're probably already familiar with applets. An applet is a program that adheres to certain conventions that allow it to run within a Java-enabled browser.

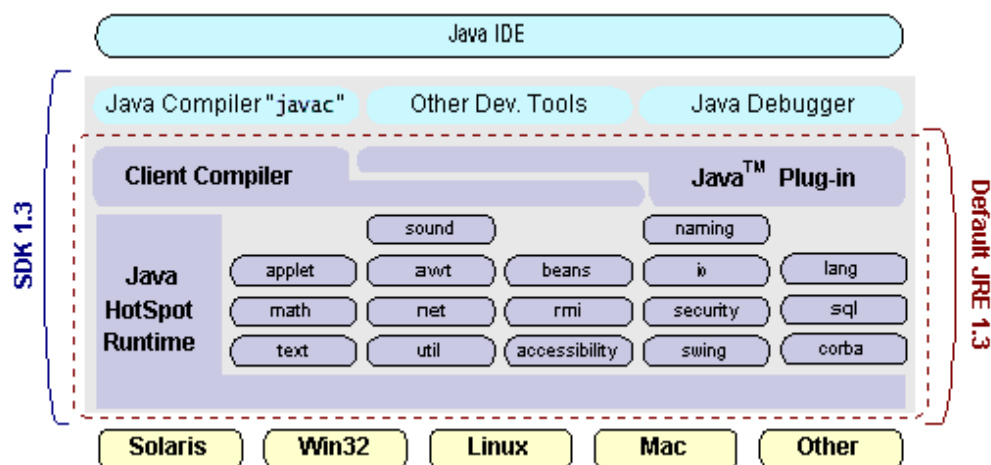
However, the Java programming language is not just for writing cute, entertaining applets for the Web. The general-purpose, high-level Java programming language is also a powerful software platform. Using the generous API, you can write many types of programs.

An application is a standalone program that runs directly on the Java platform. A special kind of application known as a *server* serves and supports clients on a network. Examples of servers are Web servers, proxy servers, mail servers, and print servers. Another specialized program is a *servlet*. A servlet can almost be thought of as an applet that runs on the server side. Java Servlets are a popular choice for building interactive web applications, replacing the use of CGI scripts. Servlets are similar to applets in that they are runtime extensions of applications. Instead of working in browsers, though, servlets run within Java Web servers, configuring or tailoring the server.

How does the API support all these kinds of programs? It does so with packages of software components that provides a wide range of functionality. Every full implementation of the Java platform gives you the following features:

- **The essentials:** Objects, strings, threads, numbers, input and output, data structures, system properties, date and time, and so on.
- **Applets:** The set of conventions used by applets.
- **Networking:** URLs, TCP (Transmission Control Protocol), UDP (User Datagram Protocol) sockets, and IP (Internet Protocol) addresses.
- **Internationalization:** Help for writing programs that can be localized for users worldwide. Programs can automatically adapt to specific locales and be displayed in the appropriate language.
- **Security:** Both low level and high level, including electronic signatures, public and private key management, access control, and certificates.
- **Software components:** Known as JavaBeans™, can plug into existing component architectures.
- **Object serialization:** Allows lightweight persistence and communication via Remote Method Invocation (RMI).
- **Java Database Connectivity (JDBC™):** Provides uniform access to a wide range of relational databases.

The Java platform also has APIs for 2D and 3D graphics, accessibility, servers, collaboration, telephony, speech, animation, and more. The following figure depicts what is included in the Java 2 SDK.



How Will Java Technology Change My Life?

We can't promise you fame, fortune, or even a job if you learn the Java programming language. Still, it is likely to make your programs better and requires less effort than other languages. We believe that Java technology will help you do the following:

- **Get started quickly:** Although the Java programming language is a powerful object-oriented language, it's easy to learn, especially for programmers already familiar with C or C++.
- **Write less code:** Comparisons of program metrics (class counts, method counts, and so on) suggest that a program written in the Java programming language can be four times smaller than the same program in C++.
- **Write better code:** The Java programming language encourages good coding practices, and its garbage collection helps you avoid memory leaks. Its object orientation, its JavaBeans component architecture, and its wide-ranging, easily extendible API let you reuse other people's tested code and introduce fewer bugs.
- **Develop programs more quickly:** Your development time may be as much as twice as fast versus writing the same program in C++. Why? You write fewer lines of code and it is a simpler programming language than C++.
- **Avoid platform dependencies with 100% Pure Java:** You can keep your program portable by avoiding the use of libraries written in other languages. The 100% Pure Java™ Product Certification Program has a repository of historical process manuals, white papers, brochures, and similar materials online.
- **Write once, run anywhere:** Because 100% Pure Java programs are compiled into machine-independent byte codes, they run consistently on any Java platform.
- **Distribute software more easily:** You can upgrade applets easily from a central server. Applets take advantage of the feature of allowing new classes to be loaded "on the fly," without recompiling the entire program.

ODBC

Microsoft Open Database Connectivity (ODBC) is a standard programming interface for application developers and database systems providers. Before ODBC became a *de facto* standard for Windows programs to interface with database systems, programmers had to use proprietary languages for each database they wanted to connect to. Now, ODBC has

made the choice of the database system almost irrelevant from a coding perspective, which is as it should be. Application developers have much more important things to worry about than the syntax that is needed to port their program from one database to another when business needs suddenly change.

Through the ODBC Administrator in Control Panel, you can specify the particular database that is associated with a data source that an ODBC application program is written to use. Think of an ODBC data source as a door with a name on it. Each door will lead you to a particular database. For example, the data source named Sales Figures might be a SQL Server database, whereas the Accounts Payable data source could refer to an Access database. The physical database referred to by a data source can reside anywhere on the LAN.

The ODBC system files are not installed on your system by Windows 95. Rather, they are installed when you setup a separate database application, such as SQL Server Client or Visual Basic 4.0. When the ODBC icon is installed in Control Panel, it uses a file called ODBCINST.DLL. It is also possible to administer your ODBC data sources through a stand-alone program called ODBCADM.EXE.

From a programming perspective, the beauty of ODBC is that the application can be written to use the same set of function calls to interface with any data source, regardless of the database vendor. The source code of the application doesn't change whether it talks to Oracle or SQL Server. We only mention these two as an example. There are ODBC drivers available for several dozen popular database systems. Even Excel spreadsheets and plain text files can be turned into data sources. The operating system uses the Registry information written by ODBC Administrator to determine which low-level ODBC drivers are needed to talk to the data source (such as the interface to Oracle or SQL Server). The loading of the ODBC drivers is transparent to the ODBC application program. In a client/server environment, the ODBC API even handles many of the network issues for the application programmer.

The advantages of this scheme are so numerous that you are probably thinking there must be some catch. The only disadvantage of ODBC is that it isn't as efficient as talking directly to the native database interface. ODBC has had many detractors make the charge that it is too slow. Microsoft has always claimed that the critical factor in performance is the quality of the driver software that is used. In our humble opinion, this is true. The availability of good ODBC drivers has improved a great deal recently. And anyway, the criticism about

performance is somewhat analogous to those who said that compilers would never match the speed of pure assembly language. Maybe not, but the compiler (or ODBC) gives you the opportunity to write cleaner programs, which means you finish sooner. Meanwhile, computers get faster every year.

JDBC

In an effort to set an independent database standard API for Java; Sun Microsystems developed Java Database Connectivity, or JDBC. JDBC offers a generic SQL database access mechanism that provides a consistent interface to a variety of RDBMSs. This consistent interface is achieved through the use of “plug-in” database connectivity modules, or *drivers*. If a database vendor wishes to have JDBC support, he or she must provide the driver for each platform that the database and Java run on.

To gain a wider acceptance of JDBC, Sun based JDBC’s framework on ODBC. As you discovered earlier in this chapter, ODBC has widespread support on a variety of platforms. Basing JDBC on ODBC will allow vendors to bring JDBC drivers to market much faster than developing a completely new connectivity solution.

JDBC was announced in March of 1996. It was released for a 90-day public review that ended June 8, 1996. Because of user input, the final JDBC v1.0 specification was released soon after.

The remainder of this section will cover enough information about JDBC for you to know what it is about and how to use it effectively. This is by no means a complete overview of JDBC. That would fill an entire book.

JDBC Goals

Few software packages are designed without goals in mind. JDBC is one that, because of its many goals, drove the development of the API. These goals, in conjunction with early reviewer feedback, have finalized the JDBC class library into a solid framework for building database applications in Java.

The goals that were set for JDBC are important. They will give you some insight as to why certain classes and functionalities behave the way they do. The eight design goals for JDBC are as follows:

1. **SQL Level API**

The designers felt that their main goal was to define a SQL interface for Java. Although not the lowest database interface level possible, it is at a low enough level for higher-level tools and APIs to be created. Conversely, it is at a high enough level for application programmers to use it confidently. Attaining this goal allows for future tool vendors to “generate” JDBC code and to hide many of JDBC’s complexities from the end user.

2. **SQL Conformance**

SQL syntax varies as you move from database vendor to database vendor. In an effort to support a wide variety of vendors, JDBC will allow any query statement to be passed through it to the underlying database driver. This allows the connectivity module to handle non-standard functionality in a manner that is suitable for its users.

3. **JDBC must be implemental on top of common database interfaces**

The JDBC SQL API must “sit” on top of other common SQL level APIs. This goal allows JDBC to use existing ODBC level drivers by the use of a software interface. This interface would translate JDBC calls to ODBC and vice versa.

4. **Provide a Java interface that is consistent with the rest of the Java system**

Because of Java’s acceptance in the user community thus far, the designers feel that they should not stray from the current design of the core Java system.

5. **Keep it simple**

This goal probably appears in all software design goal listings. JDBC is no exception. Sun felt that the design of JDBC should be very simple, allowing for only one method of completing a task per mechanism. Allowing duplicate functionality only serves to confuse the users of the API.

6. **Use strong, static typing wherever possible**

Strong typing allows for more error checking to be done at compile time; also, less error appear at runtime.

7. **Keep the common cases simple**

Because more often than not, the usual SQL calls used by the programmer are simple SELECT's, INSERT's, DELETE's and UPDATE's, these queries should be simple to perform with JDBC. However, more complex SQL statements should also be possible.

Finally we decided to proceed the implementation using Java [Networking](#).
And for dynamically updating the cache table we go for MS [Access](#) database.

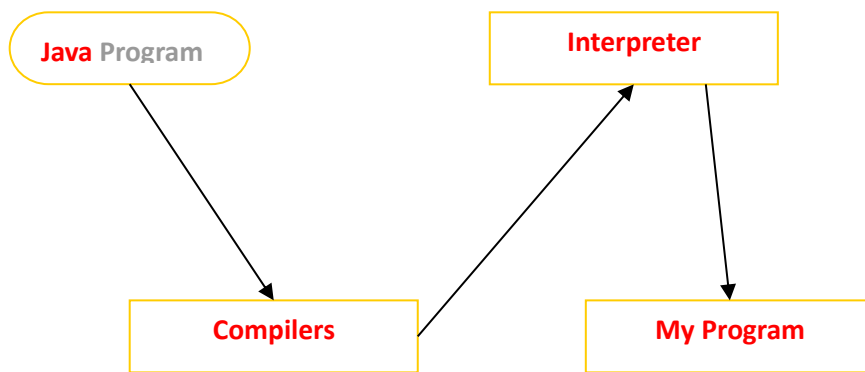
Java ha two things: a programming language and a platform.

Java is a high-level programming language that is all of the following

Distributed	Simple	Architecture-neutral
	Object-oriented	Portable
	High-performance	
	Interpreted	multithreaded
	Robust	Dynamic
	Secure	

Java is also unusual in that each Java program is both compiled and interpreted. With a compile you translate a Java program into an intermediate language called Java byte codes the platform-independent code instruction is passed and run on the computer.

Compilation happens just once; interpretation occurs each time the program is executed. The figure illustrates how this works.



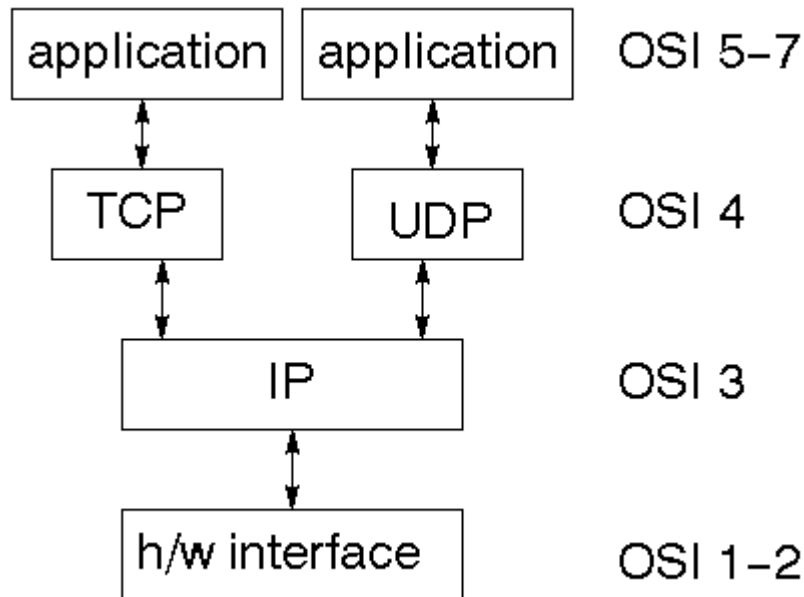
You can think of Java byte codes as the machine code instructions for the Java Virtual Machine (Java VM). Every Java interpreter, whether it's a Java development tool or a Web browser that can run Java applets, is an implementation of the Java VM. The Java VM can also be implemented in hardware.

Java byte codes help make “write once, run anywhere” possible. You can compile your Java program into byte codes on my platform that has a Java compiler. The byte codes can then be run any implementation of the Java VM. For example, the same Java program can run Windows NT, Solaris, and Macintosh.

Networking

TCP/IP stack

The TCP/IP stack is shorter than the OSI one:



TCP is a connection-oriented protocol; UDP (User Datagram Protocol) is a connectionless protocol.

IP datagram's

The IP layer provides a connectionless and unreliable delivery system. It considers each datagram independently of the others. Any association between datagram must be supplied by the higher layers. The IP layer supplies a checksum that includes its own header. The header includes the source and destination addresses. The IP layer handles routing through an Internet. It is also responsible for breaking up large datagram into smaller ones for transmission and reassembling them at the other end.

UDP

UDP is also connectionless and unreliable. What it adds to IP is a checksum for the contents of the datagram and port numbers. These are used to give a client/server model - see later.

TCP

TCP supplies logic to give a reliable connection-oriented protocol above IP. It provides a virtual circuit that two processes can use to communicate.

Internet addresses

In order to use a service, you must be able to find it. The Internet uses an address scheme for machines so that they can be located. The address is a 32 bit integer which gives the IP address. This encodes a network ID and more addressing. The network ID falls into various classes according to the size of the network address.

Network address

Class A uses 8 bits for the network address with 24 bits left over for other addressing. Class B uses 16 bit network addressing. Class C uses 24 bit network addressing and class D uses all 32.

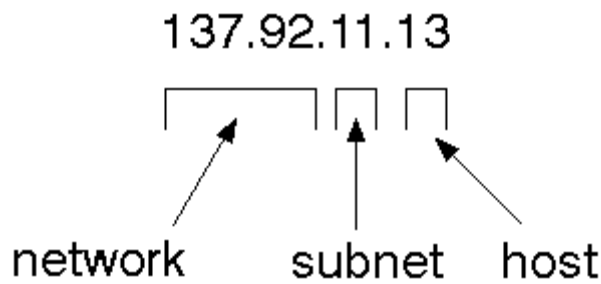
Subnet address

Internally, the UNIX network is divided into sub networks. Building 11 is currently on one sub network and uses 10-bit addressing, allowing 1024 different hosts.

Host address

8 bits are finally used for host addresses within our subnet. This places a limit of 256 machines that can be on the subnet.

Total address



The 32 bit address is usually written as 4 integers separated by dots.

Port addresses

A service exists on a host, and is identified by its port. This is a 16 bit number. To send a message to a server, you send it to the port for that service of the host that it is running on. This is not location transparency! Certain of these ports are "well known".

Sockets

A socket is a data structure maintained by the system to handle network connections. A socket is created using the call `socket`. It returns an integer that is like a file descriptor. In fact, under Windows, this handle can be used with `Read File` and `Write File` functions.

```
#include <sys/types.h>
#include <sys/socket.h>
int socket(int family, int type, int protocol);
```

Here "family" will be `AF_INET` for IP communications, `protocol` will be zero, and `type` will depend on whether TCP or UDP is used. Two processes wishing to communicate over a network create a socket each. These are similar to two ends of a pipe - but the actual pipe does not yet exist.

JFree Chart

JFreeChart is a free 100% Java chart library that makes it easy for developers to display professional quality charts in their applications. JFreeChart's extensive feature set includes:

A consistent and well-documented API, supporting a wide range of chart types;

A flexible design that is easy to extend, and targets both server-side and client-side applications;

Support for many output types, including Swing components, image files (including PNG and JPEG), and vector graphics file formats (including PDF, EPS and SVG);

JFreeChart is "open source" or, more specifically, [free software](#). It is distributed under the terms of the [GNU Lesser General Public Licence](#) (LGPL), which permits use in proprietary applications.

1. Map Visualizations

Charts showing values that relate to geographical areas. Some examples include: (a) population density in each state of the United States, (b) income per capita for each country in Europe, (c) life expectancy in each country of the world

2. Time Series Chart Interactivity

Implement a new (to JFreeChart) feature for interactive time series charts --- to display a separate control that shows a small version of ALL the time series data, with a sliding "view" rectangle that allows you to select the subset of the time series data to display in the main chart.

3. Dashboards

There is currently a lot of interest in dashboard displays. Create a flexible dashboard mechanism that supports a subset of JFreeChart chart types (dials, pies, thermometers, bars, and lines/time series) that can be delivered easily via both Java Web Start and an applet.

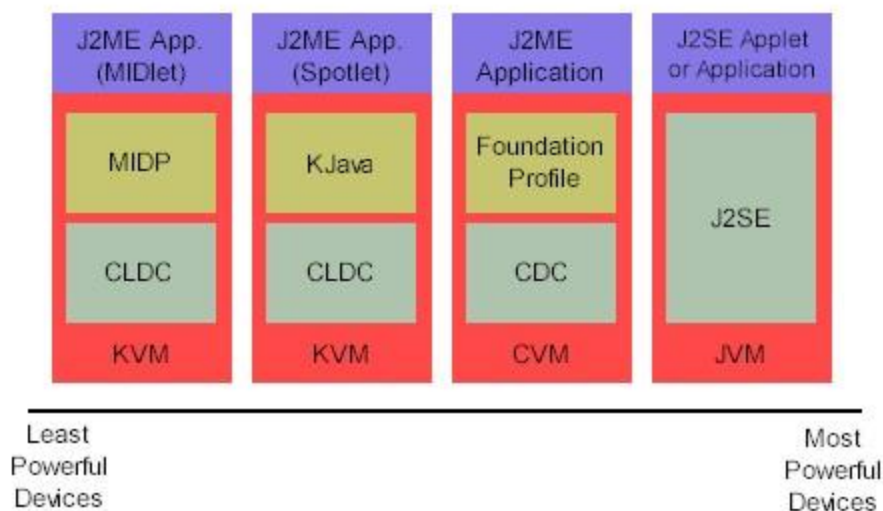
4. Property Editors

The property editor mechanism in JFreeChart only handles a small subset of the properties that can be set for charts. Extend (or reimplement) this mechanism to provide greater end-user control over the appearance of the charts.

J2ME (Java 2 Micro edition): -

Sun Microsystems defines J2ME as "a highly optimized Java run-time environment targeting a wide range of consumer products, including pagers, cellular phones, screen-phones, digital set-top boxes and car navigation systems." Announced in June 1999 at the JavaOne Developer Conference, J2ME brings the cross-platform functionality of the Java language to smaller devices, allowing mobile wireless devices to share applications. With J2ME, Sun has adapted the Java platform for consumer products that incorporate or are based on small computing devices.

1. General J2ME architecture



J2ME uses configurations and profiles to customize the Java Runtime Environment (JRE). As a complete JRE, J2ME is comprised of a configuration, which determines the JVM used, and a profile, which defines the application by adding domain-specific classes. The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. We'll discuss configurations in detail in the The profile defines the application; specifically, it adds domain-specific classes to the J2ME configuration to define certain uses for devices. We'll cover profiles in depth in the The following graphic depicts the relationship between the different virtual machines, configurations, and profiles. It also draws a parallel with the J2SE API and its Java virtual machine. While the J2SE virtual machine is generally referred to as a JVM, the J2ME

virtual machines, KVM and CVM, are subsets of JVM. Both KVM and CVM can be thought of as a kind of Java virtual machine -- it's just that they are shrunken versions of the J2SE JVM and are specific to J2ME.

2.Developing J2ME applications

Introduction In this section, we will go over some considerations you need to keep in mind when developing applications for smaller devices. We'll take a look at the way the compiler is invoked when using J2SE to compile J2ME applications. Finally, we'll explore packaging and deployment and the role preverification plays in this process.

3.Design considerations for small devices

Developing applications for small devices requires you to keep certain strategies in mind during the design phase. It is best to strategically design an application for a small device before you begin coding. Correcting the code because you failed to consider all of the "gotchas" before developing the application can be a painful process. Here are some design strategies to consider:

- * Keep it simple. Remove unnecessary features, possibly making those features a separate, secondary application.
- * Smaller is better. This consideration should be a "no brainer" for all developers. Smaller applications use less memory on the device and require shorter installation times. Consider packaging your Java applications as compressed Java Archive (jar) files.
- * Minimize run-time memory use. To minimize the amount of memory used at run time, use scalar types in place of object types. Also, do not depend on the garbage collector. You should manage the memory efficiently yourself by setting object references to null when you are finished with them. Another way to reduce run-time memory is to use lazy instantiation, only allocating objects on an as-needed basis. Other ways of reducing overall and peak memory use on small devices are to release resources quickly, reuse objects, and avoid exceptions.

4.Configurations overview

The configuration defines the basic run-time environment as a set of core classes and a specific JVM that run on specific types of devices. Currently, two configurations exist for J2ME, though others may be defined in the future:

* **Connected Limited Device Configuration (CLDC)** is used specifically with the KVM for 16-bit or 32-bit devices with limited amounts of memory. This is the configuration (and the virtual machine) used for developing small J2ME applications. Its size limitations make CLDC more interesting and challenging (from a development point of view) than CDC. CLDC is also the configuration that we will use for developing our drawing tool application. An example of a small wireless device running small applications is a Palm hand-held computer.

* **Connected Device Configuration (CDC)** is used with the C virtual machine (CVM) and is used for 32-bit architectures requiring more than 2 MB of memory. An example of such a device is a Net TV box.

J2ME profiles

What is a J2ME profile?

As we mentioned earlier in this tutorial, a profile defines the type of device supported. The Mobile Information Device Profile (MIDP), for example, defines classes for cellular phones. It adds domain-specific classes to the J2ME configuration to define uses for similar devices. Two profiles have been defined for J2ME and are built upon CLDC: KJava and MIDP. Both KJava and MIDP are associated with CLDC and smaller devices. Profiles are built on top of configurations. Because profiles are specific to the size of the device (amount of memory) on which an application runs, certain profiles are associated with certain configurations.

A skeleton profile upon which you can create your own profile, the Foundation Profile, is available for CDC.

Profile 1: KJava

KJava is Sun's proprietary profile and contains the KJava API. The KJava profile is built on top of the CLDC configuration. The KJava virtual machine, KVM, accepts the same

byte codes and class file format as the classic J2SE virtual machine. KJava contains a Sun-specific API that runs on the Palm OS. The KJava API has a great deal in common with the J2SE ABSTRACT Windowing Toolkit (AWT). However, because it is not a standard J2ME package, its main package is com.sun.kjava. We'll learn more about the KJava API later in this tutorial when we develop some sample applications.

Profile 2: MIDP

MIDP is geared toward mobile devices such as cellular phones and pagers. The MIDP, like KJava, is built upon CLDC and provides a standard run-time environment that allows new applications and services to be deployed dynamically on end user devices. MIDP is a common, industry-standard profile for mobile devices that is not dependent on a specific vendor. It is a complete and supported foundation for mobile application

development. MIDP contains the following packages, the first three of which are core CLDC packages, plus three MIDP-specific packages.

- * java.lang
- * java.io
- * java.util
- * javax.microedition.io
- * javax.microedition.lcdui
- * javax.microedition.midlet
- * javax.microedition.rms

SOURCE CODE:

HOME PAGE:

```
<html>
<head>
<TITLE>REAL ESTATE</TITLE>
```

```

<style>

#heading

{
height:100px;
width:100%;
border:1px solid black;
display:flex;
justify-content: space-between;
align-items:center;
}

#links

{
height:70px;
width:100%;
}

ul li

{
width:70px;
list-style:none;
float: left;
font-size: 20px;
}

a

{
text-decoration:none;
color:blue;
}

body

```

```

{

background-image:url("https://www.indiabullsrealestate.com/wp-
content/uploads/images/commercial-real-estate-projects.jpg");

background-repeat:no-repeat;

background-size:cover;

margin: 0px;

}

h1

{

margin-left:300px;

}

.image

{

height:100px;

width:120px;

padding:10px;

}

#menu

{

    display:flex;

}

</style>

</head>

<body>

<div id="heading">

    <div id="menu">



```

```

<center>

<h1 style="color:white; background-color:black; width:fit-content; ">REAL
ESTATE</h1>

</div>

<ul>

<li><a href="index.html">Home</li>

<li><a href="adminlogin.html">Admin</li>

<li><a href="userlogin.html">user</li>

<li><a href="dept.html">Department</li>

</ul>

</div>

</body>

</html>

```

Userpage:

```

<html>
<head>
<style>
#body
{
background-
image:url("https://static.vecteezy.com/system/resources/thumbnails/007/1
64/537/small/fingerprint-identity-sensor-data-protection-system-podium-
hologram-blue-light-and-concept-free-vector.jpg");
background-repeat:no-repeat;
background-size:cover;
height:100vh;
width:100%;
}

```

```

</style>
</head>
<body>
<div id="body">
<center>
  <div id="box">
    <h1 style="color:white;">user login</h1>
    <table>
    <tr>
      <th style="color:white;">username:</th>
      <td><input type="text" name="username" placeholder="username"
required></td>
    </tr>
    <tr>
      <th style="color:white;">password:</th>
      <td><input type="text" name="password" placeholder="password"
required></td>
    </tr>
    <tr>
      <td>
        <Input type="submit" value="Login">
        <Input type="Reset" value="Reset">
      </td>
    </tr>
    <tr>
      <td style="color:white;">Don't have an account?<a
href="register.html">signup</a>
    </td>
    </tr>
  </table>
</div>
</center>
</div>
</body>
</html>

```

Admin login:

```
<html>

<head>

<TITLE>Admin login page</TITLE>

<style>

body{

background-
image:url("https://t3.ftcdn.net/jpg/00/21/70/82/240_F_21708280_RFKz4O7Im
QluB9FgX2mUYFUNDmlLokX6.jpg");

background-repeat:no-repeat;

background-size:cover;

height:100px;

width:100%;

}

#p1
{
text-align:center;
color:white;
}

.bg{
color:red;

height:100vh;
width:100vw;
}

.imgcls{
height:400px;
width:800px;
}

.bg1{
height:250px;
width:350px;
```

```
padding:40px;
background-color:white;
}

</style>
</head>
<body>
<div id="bg">
<center>
<div id="bg1">
<h1 id="p1">Admin Login</h1>
<table>
  <tr>
    <td style="color:white">username:</td>
    <td><input
name="user" type="text" placeholder="username"
/></td>
  </tr>
  <tr>
    <td style="color:white">password:</td>
    <td><input
name="user_pass" type="password" placeholder="*****"
/></td>
  </tr>
  <tr>
    <td>
      <button type="submit">login</button></a>
    </td>
  </tr>
</table>
</div>
</center>
</div>
</body>
</html>
```


Registration page :

```
<DOCTYPE! html>

<html>

<head>

<style>

.register{

background-image:url("https://img.freepik.com/free-vector/geometric-
gradient-futuristic-background_23-2149116406.jpg");

width:100vw;

height:100vh;

}

.reg2{

text-align:center;

background-color:white;

height:350px;

width:500px;

padding:40px;

}

.reg1{

text-align:center;

color:white;

}

</style>

<TITLE>register form</TITLE>

</head>

<div id="section1">

<body>

<div class="register">

<h1 class="reg1">Registration Form</h1>

<form id="register" method="post">

<center>

<div class="reg2">

<table>
```

```

<tr>
<td>Name:</td>
<td><Input type="text" placeholder="Name" name=""></td>
</tr>
<tr>
<td>Phone number:</td>
<td><Input type="phone" placeholder="8978*****"></td>
</tr>
<tr>
<td>Email id:</td>
<td><Input type="email" placeholder="example1@*8"></td>
</tr>
<tr>
<td>Password:</td>
<td><Input type="password" placeholder="password"></td>
</tr>
<tr>
<td>Gender:</td>
<td><Input type="radio" name="gender">Male
      <Input type="radio" name="gender">Female
</td>
</tr>
<tr>
<td>Date of Birth:</td>
<td><Input type="date"></td>
</tr>
<tr>
<td>Address:</td>
<td>
<textarea row="4" col="15" placeholder="Address"></textarea></td>
</tr>
<tr>
<td>
<a href="kuserlogin.html" <button type="submit">submit</button></a>

```

```
</td>
</tr>
</table>
</div>
</center>
</form>
</div>
</body>
</div>
</html>
```

RESULTS/DISCUSSIONS

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub-assemblies, assemblies and/or a finished product. It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of tests. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application. It is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

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 centred 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 invoked.

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 Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in

which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing:

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects.

The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

TEST CASES:

Test case1 for Login form:

FUNCTION:	LOGIN
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

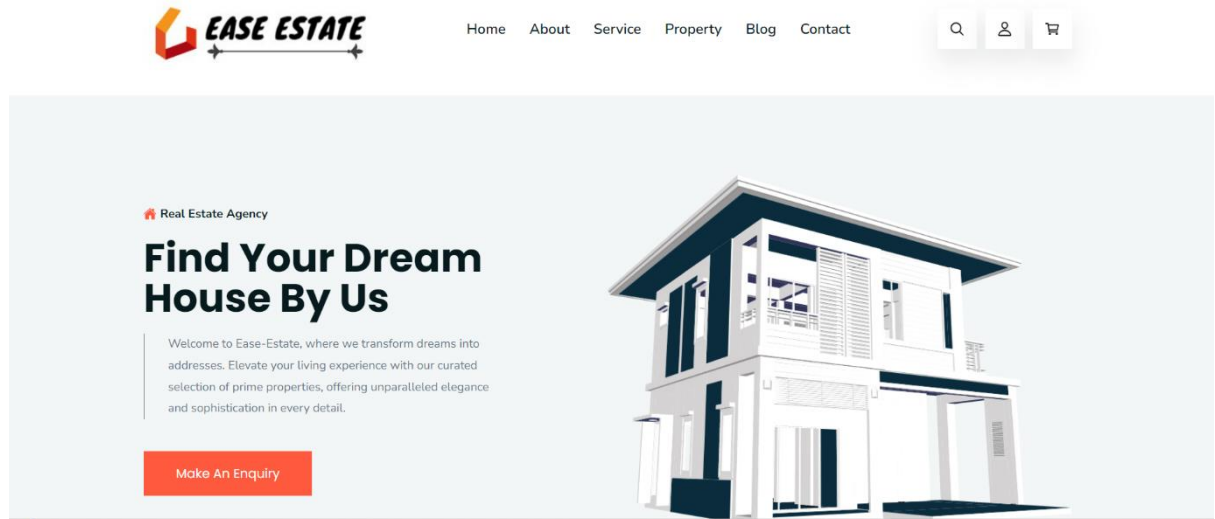
Test case2:

Test case for User Registration form:

FUNCTION:	USER REGISTRATION
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

SCREENSHOTS:

HOME PAGE:





USER PAGE:


user login entails accessing a system or application by entering a unique combination of credentials, typically a username and password. Security may be reinforced through additional measures such as Multi-Factor Authentication (MFA).

Create an account

☐ I accept the terms & conditions.

 Connect with facebook

 Connect with Twitter


 Connect with Google


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


REGISTRATION:

Create an account

☐ I accept the terms & conditions.

 Connect with facebook

 Connect with Twitter

 Connect with Google

I already have an account? [Login Now](#)

CONCLUSION

CONCLUSION

In conclusion, the real estate industry is a dynamic and multifaceted sector that plays a crucial role in the economic development of nations. Through its impact on job creation, infrastructure development, and overall economic growth, real estate proves to be a significant driver of prosperity. The evolving landscape of the industry, economic factors, necessitates adaptability and innovation.

The Indian real estate market is currently experiencing a period of growth, with housing prices rising in several cities across the country. Factors such as low interest rates, government initiatives, infrastructure development, and favorable demographics have contributed to this upward trend.

The Commercial Real Estate Investment Market is a complex system of firms based in different sectors of the industry. The future of companies in this sector are most directly affected and tied to changes in global economic conditions. Depending on what the firms focus on, from Land and Development, Retail Shopping Centers, Industry/Office Space, or Apartments and Multi-family homes, there are many factors affecting their strategy, overall performance, and future.

FUTURE SCOPE

The real estate market in India has been experiencing dynamic shifts and changes over the years, driven by technological advancements, demographic shifts, and economic fluctuations after the pandemic. In 2023, home buyers need to be aware of estate industry in India. The Market research analysts at concorde have gathered industry trends and real estate predictions for the next five years. In this blog, let's look at the top real estate trends and predict, and find out whether real estate is still a good investment to make

India's real estate market is expected to undertake a growth rate (CAGR) of 9.2% during the five years from 2023 to 2028. Looking ahead, the future of the real estate market in India appears promising, with continued growth and evolution.

BIBLIOGRAPHY

1. Akaaboune, O., R. Quarles and R. D. Burnett. 2021. Assessing the relevance of county-level eco-efficiency to single-family housing prices. *Accounting and the Public Interest* (21): 137-155.
2. Alcock, J. and E. Steiner. 2017. The interrelationships between REIT capital structure and investment. *Abacus* 53(3): 371-394.
3. Allen, W. E. III. and M. B. Foster. 2005. The best of both worlds: A combination of cost segregation and like-kind exchanges can save on real estate taxes. *Journal of Accountancy* (August): 43-46.
4. Amarante, M. and A. P. Curatola. 2022. Taxes: Converting a primary residence into a rental property: Renting out your personal residence for a short period or converting it into rental property can raise federal income tax issues. *Strategic Finance* (July): 17-18.
5. Apgar, M. IV. 2009. What every leader should know about real estate. *Harvard Business Review* (November): 100-107.
6. Baik, B., B. K. Billings and R. M. Morton. 2008. Reliability and transparency on non-GAAP disclosures by real estate investment trusts (RIETs). *The Accounting Review* (March): 271-301.
7. Banham, R. 2011. Space race: The weak real estate market offers CFO's a great opportunity. Here's how to maximize it. *CFO* (January/February): 59-63.
8. Park, Tae H., and Lorne N. Switzer. "Risk management of real estate: The case of real estate swaps." *Journal of Real Estate Finance and Economics* 11, no. 3 (November 1995): 219-33.
9. Bon, Ranko. "Corporate Real Estate Management." *Facilities* 10, no. 12 (December 1992): 13-17.
10. unoz Cabanes, Alberto, Alfonso Herrero de Egana, and Arturo Romero. "Real option analysis. The viability of real estate projects." *Investment Management and Financial Innovations* 17, no. 4 (December 8, 2020): 271-84.