**CHAPTER 1**

**INTRODUCTION**

**1.1 COURSE OBJECTIVES**

The aim of this project is develop banking interface for the bank users. This project is developed using Java with JDBC connection for back-end and Swing and Awt for front-end. The application is very useful for bank users. The application reduces lot of work load for customer and as well as for owner.

This project works as Stand-alone system. In this project, Users can login into the application by using username and password. This project provides secure transaction and can be trusted by the users. The project consists of bank details, transaction details, and balance enquiry and withdrawal. It is easy and simple to use this application. As the language used is java, which secure and nobody can hack or misuse the application. The transactions are safe and consist of good interfaces.

The agenda of this project is designing an application to manage the user account in the application and also help the customers to do transactions without having to visit the bank physically. The project contains many modules like account details, balance enquiry, withdrawal, loan details, etc. This projects also aims in reducing the man power and atomize the existing banking system.

**1.2 PROBLEM STATEMENT**

The Current Banking system is under manual power and requires lot of paper work and manual records. In order to search the detail of a particular person the bank has to undergo huge documents and people have to stand in queue for banking transaction, Any work related to bank should be done in bank by visiting the bank. This will require time consumption, manual work consumption and there is no atomization. So to avoid this problems Net banking project is developed to reduce man power, time and hard work. This will act as Smart Work. People can sit in the place where they are and can check their account details and banking details without interrupting to the bank.

**1.3 OUTCOMES OF THE PROJECT WORK CARRIED OUT**

The outcomes of this project are:

* To automate the existing system..
* User-friendly.
* Multiple Modules.
* Use of interface, frames and button.
* To make banking available on mobile.
* Help customers.
* In case of emergency, people need not go to bank and waste their time.
* Secure registration and profile management facilities for Customers.
* Secured mechanism for checking out the account details.
* Updates to customers about the current bank details.

**CHAPTER 2**

**JAVA FEATURES AND OOPS CONCEPT**

# 2.1 Features of Java

The primary objective of Java Programming language creation was to make it portable, simple and secure programming language. Apart from this, there are also some excellent features which play an important role in the popularity of this language. The features of Java are also known as java buzzwords.

A list of most important features of Java language is given below.



### Simple

Java is very easy to learn, and its syntax is simple, clean and easy to understand. According to Sun, Java language is a simple programming language because:

* Java syntax is based on C++ (so easier for programmers to learn it after C++).
* Java has removed many complicated and rarely-used features, for example, explicit pointers, operator overloading, etc.
* There is no need to remove unreferenced objects because there is an Automatic Garbage Collection in Java.

### Object-oriented

Java is an [object-oriented](https://www.javatpoint.com/java-oops-concepts) programming language. Everything in Java is an object. Object-oriented means we organize our software as a combination of different types of objects that incorporates both data and behavior.

Object-oriented programming (OOPs) is a methodology that simplifies software development and maintenance by providing some rules.

Basic concepts of OOPs are:

1. [Object](https://www.javatpoint.com/object-and-class-in-java)
2. Class
3. [Inheritance](https://www.javatpoint.com/inheritance-in-java)
4. [Polymorphism](https://www.javatpoint.com/runtime-polymorphism-in-java)
5. [Abstraction](https://www.javatpoint.com/abstract-class-in-java)
6. [Encapsulation](https://www.javatpoint.com/encapsulation)

### Platform Independent

Java is platform independent because it is different from other languages like [C](https://www.javatpoint.com/c-programming-language-tutorial), [C++](https://www.javatpoint.com/cpp-tutorial), etc. which are compiled into platform specific machines while Java is a write once, run anywhere language. A platform is the hardware or software environment in which a program runs.

There are two types of platforms software-based and hardware-based. Java provides a software-based platform.

### Secured

Java is best known for its security. With Java, we can develop virus-free systems. Java is secured because:

* **No explicit pointer**
* **Java Programs run inside a virtual machine sandbox.**
* Class loader:  Class loader in Java is a part of the Java Runtime Environment (JRE) which is used to load Java classes into the Java Virtual Machine dynamically. It adds security by separating the package for the classes of the local file system from those that are imported from network sources.
* Byte code Verifier: It checks the code fragments for illegal code that can violate access right to objects.
* Security Manager: It determines what resources a class can access such as reading and writing to the local disk.

### Robust

Robust simply means strong. Java is robust because:

* It uses strong memory management.
* There is a lack of pointers that avoids security problems.
* There is automatic garbage collection in java which runs on the Java Virtual Machine to get rid of objects which are not being used by a Java application anymore.

### Architecture-neutral

Java is architecture neutral because there are no implementation dependent features, for example, the size of primitive types is fixed.

In C programming, int data type occupies 2 bytes of memory for 32-bit architecture and 4 bytes of memory for 64-bit architecture. However, it occupies 4 bytes of memory for both 32 and 64-bit architectures in Java.

### Portable

Java is portable because it facilitates you to carry the Java byte code to any platform. It doesn't require any implementation.

### High-performance

Java is faster than other traditional interpreted programming languages because Java byte code is "close" to native code. It is still a little bit slower than a compiled language (e.g., C++). Java is an interpreted language that is why it is slower than compiled languages, e.g., C, C++, etc.

### Distributed

Java is distributed because it facilitates users to create distributed applications in Java. RMI and EJB are used for creating distributed applications. This feature of Java makes us able to access files by calling the methods from any machine on the internet.

### Multi-threaded

A thread is like a separate program, executing concurrently. We can write Java programs that deal with many tasks at once by defining multiple threads. The main advantage of multi-threading is that it doesn't occupy memory for each thread. It shares a common memory area. Threads are important for multi-media, Web applications, etc.

### Dynamic

Java is a dynamic language. It supports dynamic loading of classes. It means classes are loaded on demand. It also supports functions from its native languages, i.e., C and C++.

Java supports dynamic compilation and automatic memory management (garbage collection).

**2.2 OOPS CONCEPT**

Object-Oriented Programming or OOPs refers to languages that use objects in programming. Object-oriented programming aims to implement real-world entities like inheritance, hiding, polymorphism etc in programming. The main aim of OOP is to bind together the data and the functions that operate on them so that no other part of the code can access this data except that function.

OOPs Concepts:

* [Polymorphism](https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/#Polymorphism)
* [Inheritance](https://www.geeksforgeeks.org/inheritance-in-java/)
* [Encapsulation](https://www.geeksforgeeks.org/encapsulation-in-java/)
* [Abstraction](https://www.geeksforgeeks.org/abstraction-in-java-2/)
* [Class](https://www.geeksforgeeks.org/classes-objects-java/)
* [Object](https://www.geeksforgeeks.org/classes-objects-java/)
* [Method](https://www.geeksforgeeks.org/methods-in-java/)
* [Message Passing](https://www.geeksforgeeks.org/object-oriented-programming-oops-concept-in-java/#Message%20Passing)

**Polymorphism:** Polymorphism refers to the ability of OOPs programming languages to differentiate between entities with the same name efficiently. This is done by Java with the help of the signature and declaration of these entities.

* Overloading in Java
* Overriding in Java

[**Inheritance**](https://www.geeksforgeeks.org/inheritance-in-java/)**:** Inheritance is an important pillar of OOP (Object Oriented Programming). It is the mechanism in java by which one class is allows inheriting the features of another class.

* **Super Class:**The class whose features are inherited is known as super class (or a base class or a parent class).
* **Sub Class:** The class that inherits the other class is known as subclass (or a derived class, extended class, or child class). The subclass can add its own fields and methods in addition to the super class fields and methods.
* **Reusability:**Inheritance supports the concept of “reusability”, i.e. when we want to create a new class and there is already a class that includes some of the code that we want, we can derive our new class from the existing class. By doing this, we are reusing the fields and methods of the existing class.

[**Encapsulation**](https://www.geeksforgeeks.org/encapsulation-in-java/)**:** Encapsulation is defined as the wrapping up of data under a single unit. It is the mechanism that binds together code and the data it manipulates. Other way to think about encapsulation is, it is a protective shield that prevents the data from being accessed by the code outside this shield.

* Technically in encapsulation, the variables or data of a class is hidden from any other class and can be accessed only through any member function of own class in which they are declared.
* As in encapsulation, the data in a class is hidden from other classes, so it is also known as **data-hiding**.
* Encapsulation can be achieved by: Declaring all the variables in the class as private and writing public methods in the class to set and get the values of variables.

[**Abstraction**](https://www.geeksforgeeks.org/abstraction-in-java-2/)**:** Data Abstraction is the property by virtue of which only the essential details are displayed to the user. The trivial or the non-essentials units are not displayed to the user. Ex: A car is viewed as a car rather than its individual components.

Data Abstraction may also be defined as the process of identifying only the required characteristics of an object ignoring the irrelevant details. The properties and behaviors of an object differentiate it from other objects of similar type and also help in classifying/grouping the objects.

[**Class**](https://www.geeksforgeeks.org/classes-objects-java/)**:** A class is a user defined blueprint or prototype from which objects are created.  It represents the set of properties or methods that are common to all objects of one type. In general, class declarations can include these components, in order:

**Modifiers**: A class can be public or has default access (Refer [this](https://www.geeksforgeeks.org/access-specifiers-for-classes-or-interfaces-in-java/) for details).

**Class name:** The name should begin with a initial letter (capitalized by convention).

**Super class (if any):** The name of the class’s parent (super class), if any, preceded by the keyword extends. A class can only extend (subclass) one parent.

**Interfaces (if any):** A comma-separated list of interfaces implemented by the class, if any, proceeded by the keyword implements. A class can implement more than one interface.

**Body:** The class body surrounded by braces, { }.

[**Method**](https://www.geeksforgeeks.org/methods-in-java/)**:** A method is a collection of statements that perform some specific task and return result to the caller. A method can perform some specific task without returning anything. Methods allow us to **reuse** the code without retyping the code. In Java, every method must be part of some class which is different from languages like C, C++ and Python.  
Methods are **time savers**and help us to**reuse** the code without retyping the code.

**Message Passing:** Objects communicate with one another by sending and receiving information to each other. A message for an object is a request for execution of a procedure and therefore will invoke a function in the receiving object that generates the desired results. Message passing involves specifying the name of the object, the name of the function and the information to be sent.

**CHAPTER 3**

**REQUIREMENTS AND DESIGN**

System Analysis is a process by which we attribute process or goals to a human activity, determine how well those purpose are being achieved and specify the requirements of the various tools and techniques that are to be used within the system if the system performances are to be achieved.

### Non-Functional Requirements

1. Secure access of confidential data (user’s details). SSL can be used.
2. 24 X 7 availability
3. Browser testing and support for IE, NN, Mozila, and Firefox
4. Reports exportable in .XLS, .PDF
5. Create a detailed UML diagram (Component, Sequence, Class) for the system and its sub-components

**3.1 HARDWARE SPECIFICATION**

* Operating System: Windows 7, 8,9,10.
* Hard disk: 1TB
* RAM: 5GB

**3.2 SOFTWARE SPECIFICATION**

* Programming Language: Java, JavaScript
* Back End: JDBC
* Front End: Swing, Awt.
* You need to install an IDE Eclipse / Myeclipse / Netbeans.
* MySql database.

**3.3 CLASS DIAGRAM**

**3.4 DATA FLOW DIAGRAM**

**3.6 SEQUENCE DIAGRAM**

**CHAPTER 4**

**IMPLEMENTATION**

**CHAPTER 5**

**OUTPUT SNAPSHOTS**

**CHAPTER 6**

**CONCLUSION AND FUTURE SCOPE**

This project was successfully completed within the time span allotted. The project

NET BANKING has been developed in java. All the modules are tested separately and put together to form the main system. Finally the system is tested with real data and everything worked successfully. Thus the system has fulfilled the entire objective identified. The system had been developed in an attractive dialogs fashion.

So user with minimum knowledge about computers can also operate the system easily. It will make easy interactions between users and store. The speed and accuracy are maintained in proper way.

**CHAPTER 7**

**REFERENCES**

 Website Links

* <https://www.scribd.com/doc/80911858/REPORT-OF-ONLINE-BANKING-SYSTEM>
* <https://en.wikipedia.org/wiki/Online_banking>
* <https://finteknews.com/online-banking-report-millennial-trends/>

* <https://issuu.com/sanjaykumarguptaa/docs/a-research-report-on-e-banking>

Text Books

## [JavaScript Bible, 4th Edition](http://target.georiot.com/Proxy.ashx?TSID=5036&GR_URL=http%3A%2F%2Fwww.amazon.com%2Fs%2Fref%3Das_li_ss_tl%3Furl%3Dsearch-alias%253Dbooks%26field-keywords%3DJavaScript+Bible%2C+4th+Edition+&amazon-ids-by-cc=US%3DRankernode32467-20)Danny Goodman