INTERNSHIP COMPLETION CERTIFICATE

TECHNOHACKS
LET'S GROW TOGETHER

CERTIFICATE OF INTERNSHIP

This Certificate Is Proudly Presented To

Prathmesh Dandage

For Successfully Completing An Internship On

Java Development

From **01/01/2023** To **31/01/2024** At TechnoHacks EduTech. During This Internship, We Found Him/Her Consistent And Hardworking. We Wish All The Best For Future Endeavors.

Sandip Gavit
Founder

CERTIFIED ISO 9001:2015 COMPANY

MSME
MICRO, SMALL & MEDIUM ENTERPRISES
सूक्ष्म, लघु एवं मध्यम उद्यम

02/02/2024
Date

ACKNOWLEDGEMENT

First I would like to thank our Industrial Mentor **Mr. Sandip Gavit**, of TechnoHacks EduTech for giving me an opportunity to do an internship within the organization.

Also would like to thank our entire colleagues that worked with us in **TechnoHacks EduTech** . for their patience and openness, they created an enjoyable working environment. It is indeed with a great sense of pleasure and immense sense of gratitude that we acknowledge thehelp of these individuals.

I would like to thank our guide **Dr.R.V.Babar** of Information Technology Department Sinhgad Institute of Technology Lonavala for providing me the opportunity of internship.

I would like to thank ICEM ‘S Internship Mentor, for their support and advice to get and complete internship in above said organization. I am extremely greatfull to our department staff members and friends who helped me in the successful completion of this internship.

Prathamesh Dandage

(TE-11)

Index

CHAPTER 1:

COMPANY DETAILS

Chapter 1

1. Company Overview

TechnoHacks offers high-quality IT training and advanced tech products to help businesses and individuals stay competitive in the fast-changing tech world. Our expert team is here to support you in achieving your goals and reaching your full potential.

TechnoHacks provide various training programs to cater to different skill levels, from beginners to experienced professionals, helping everyone to grow in their chosen fields. Our tech products include software and hardware solutions designed to boost your performance and enhance your operations.

TECHNOHACKS

LET'S GROW TOGETHER

TechnoHacks belives that with the proper mentorship everyone can reach the goals. They also provide the Internships and Placement training to the students. Their mission is to provide platform to students and working proffessionals to improve their skillset and rock into IT industry. Their vision is to provide world class quality skills which industry needs.

2. IT Services

Target Market

B2B

We provide trainings and mentorship to the students and working proffessionals to improve their skillsets to next level. We belive that with the proper mentorship everyone can reach the goals. We also provide the Internships with Trainings.

Client Segment : AdTech, Enterprise Tech, Software, Technology Target Companies : Startup, Small Enterprise, Medium Enterprise, Large Enterprise Target Geography : India

B2C

We provide trainings and mentorship to the students and working proffessionals to improve their skillsets to next level. We belive that with the proper mentorship everyone can reach the goals. We also provide the Internships

User Age : Less than 18, 18 to 25, 26 to 34

User Income : Lower-middle Income

Location : India

3. Education / Training

- **Trainings & Internships**

  Unlock your potential with our comprehensive IT training and internship programs, designed for students and working professionals to enhance their skills and advance their careers.

- **Consultancy**

  Looking to boost your IT skills and career prospects? Our consultancy services offer personalized guidance for students and professionals seeking to take their expertise to the next level.

- **Software Development**

  Accelerate your digital transformation with our comprehensive software development solutions tailored to your business needs.

1.4 ABSTRACT

I have completed my internship from TechnoHacks which is located in Nashik

An internship will helps to demonstrate your ability to work on real-world projects and prove to your possible future employer what work you can produce.

In these 4 Weeks of Internship, I had a Great experience While learning about Core Java. This allows you to apply practical knowledge you may have learned from a classroom setting while you develop important soft skills, such as time management, organization, adaptability, problem-solving and teamwork.

The internship focused on enhancings advantages of the programming language java and its uses along with the knowledge of Sequential programming language to readers. It helps students understand why we need to study java and what we can achieve through this subject. This Language can be implemented on multiple platforms irrespective of the platform. It's a full-featured object oriented language and has vast usage around the world. Java is a Robust language.I have learnt many major concepts of Java.

This training gave us a complete understanding of Java starting from basic concepts to advanced concepts. Introduction to Java Programming and OOp concepts.In this internship I have learnt, how the web application exactly develop and work.This Internship is designed to equip you with the essential skills and knowledge required to thrive in the dynamic world of software development. Throughout the program, you will dive deep into Java programming fundamentals, explore advanced concepts, and gain practical experience through project-based learning.

I have also design a project in Java programming during this internship training. This was a great experience for all of us to be the part this internship. During this session I have develop a mini project which is basically known as a Spending or a expense tracker.

CHAPTER 2:

Modules

2.1 Introduction to JAVA & Features

Java is a powerful, versatile programming language widely used for developing a variety of applications, from desktop to web and mobile. Here's an introduction to Java along with its key features:

Introduction to Java:

Java was developed by Sun Microsystems (now owned by Oracle Corporation) in the mid-1990s and has since become one of the most popular programming languages globally. It was designed with the principle of "write once, run anywhere" (WORA), meaning that Java code can run on any platform that has a Java Virtual Machine (JVM) installed, without needing to be recompiled.

Key Features of Java:

1. **Platform Independence:** Java programs are compiled into bytecode, which is platform-independent. This bytecode can run on any device or operating system with a Java Virtual Machine (JVM) installed, making Java highly portable.
2. **Object-Oriented:** Java is an object-oriented programming (OOP) language, meaning it organizes code into objects that interact with one another. This approach promotes modularity, reusability, and easier maintenance of code.
3. **Robust and Secure:** Java's strong memory management, exception handling, and type safety features make it robust and less prone to errors. Additionally, Java's security features, such as its sandbox environment and built-in security manager, help protect systems from malicious code.
4. **Multithreading:** Java supports multithreading, allowing concurrent execution of multiple threads within a single program. This feature is essential for developing scalable and responsive applications, particularly in areas like server-side programming and GUI development.
5. **Rich Standard Library:** Java comes with a comprehensive standard library (Java API) that provides classes and methods for common programming tasks, such as input/output operations, networking, database connectivity, and more. This rich set of libraries simplifies development and reduces the need for writing low-level code from scratch.
6. **High Performance:** While Java is often associated with its "interpreted" nature due to its bytecode execution, modern JVM implementations employ sophisticated optimization techniques, such as Just-In-Time (JIT) compilation, to achieve high performance. Additionally, Java's garbage

collection mechanism automatically manages memory, reducing the risk of memory leaks and improving overall performance.

7. particularly enterprise-level systems that need to handle large volumes of data and concurrent users. Its support for distributed computing and integration with technologies like Java EE (Enterprise Edition) enable the development of robust and scalable solutions.

Java Virtual Machine (JVM):

- The JVM is an abstract computing machine that provides the runtime environment in which Java bytecode can be executed.
- It acts as an intermediary between Java bytecode (generated by the Java compiler) and the underlying hardware and operating system.
- The JVM interprets the bytecode or uses just-in-time (JIT) compilation to translate bytecode into native machine code for improved performance.
- It also manages memory, handles exceptions, and facilitates garbage collection.
- JVM implementations are available for various platforms, enabling Java's "write once, run anywhere" (WORA) capability.

Java Runtime Environment (JRE):

- The JRE is a subset of the JDK and includes the JVM along with essential libraries and components required to run Java applications.
- It provides the runtime environment for Java applications without the development tools included in the JDK.
- The JRE consists of the JVM, class libraries, and other supporting files needed to execute Java programs.
- End-users typically install the JRE on their systems to run Java applications or Java applets within web browsers.

Java Development Kit (JDK):

- The JDK is a full-featured software development kit for Java that includes the JRE along with development tools such as compilers, debuggers, and other utilities.
- It provides everything developers need to write, compile, debug, and deploy Java applications.
- In addition to the components of the JRE, the JDK includes the Java compiler (javac), debugger (jdb), documentation tools, and various development APIs.
- Developers use the JDK to create Java applications, applets, and other Java-based software.

2.2&2.3String Class & Array Class :

Sure, let's delve into the **String** class and the **Array** class in Java, along with some of their commonly used methods:

String Class:

Overview:

- The **String** class in Java represents a sequence of characters.
- Strings in Java are immutable, meaning their values cannot be changed after they are created.
- Strings are widely used in Java for representing text and manipulating character data.

Commonly Used Methods:

1. **length():** Returns the length of the string (number of characters).
2. **charAt(int index):** Returns the character at the specified index.
3. **substring(int beginIndex), substring(int beginIndex, int endIndex):** Returns a substring of the original string.
4. **indexOf(String str), lastIndexOf(String str):** Returns the index of the first/last occurrence of the specified substring.
5. **toUpperCase(), toLowerCase():** Returns a new string with all characters converted to uppercase/lowercase.

Array Class:

Overview:

- The **Array** class in Java provides static methods for dynamically creating and manipulating arrays.
- While arrays in Java are objects, they are implemented as special language constructs rather than instances of the **Array** class.

Commonly Used Methods:

1. **static <T> T[] copyOf(T[] original, int newLength):** Copies the specified array, truncating or padding with default values if necessary.
2. **static <T> void fill(T[] array, T value):** Assigns the specified value to each element of the array.

3. **static <T> void sort(T[] array):** Sorts the elements of the array into ascending order.
4. **static boolean equals(Object[] a, Object[] b):** Returns **true** if the two specified arrays are equal to one another.
5. **static String toString(Object[] a):** Returns a string representation of the contents of the specified array.

2.4 Files & Web Application development using Servlets

- **Files & I/O Streams:**

In Java, the **java.io** package provides classes for input and output operations, including reading from and writing to files and streams. Some commonly used classes include **File**, **FileInputStream**, **FileOutputStream**, **BufferedReader**, **BufferedWriter**, etc. These classes allow Java programs to interact with files, perform file manipulation, and handle input/output operations efficiently.

- **Importance of Packages:**

Packages in Java are used to organize classes into namespaces, which helps in avoiding naming conflicts and provides modularity and reusability. They also facilitate better code organization and maintenance. By grouping related classes and interfaces into packages, Java developers can easily manage large codebases and collaborate with others more effectively.

- **Web Application Development using Servlets:**

Servlets are Java classes used to extend the capabilities of web servers to respond to requests from web clients. The Servlet API provides classes and interfaces for writing servlets, handling HTTP requests and responses, managing sessions, and more. Servlets are commonly used in web application development to create dynamic web pages, process form data, interact with databases, and implement business logic on the server-side.

- **Introduction to Servlet API:**

The Servlet API provides a standard way to extend the functionality of web servers. It includes interfaces and classes that servlets use to communicate with the web server. Key components of the Servlet API include **javax.servlet.Servlet**, **javax.servlet.http.HttpServlet**, **javax.servlet.ServletRequest**, **javax.servlet.ServletResponse**, etc.

- **Web Server Steps to Creation & Execution of Servlets using Tomcat server:**

1. **Create Servlet Class:** Write a Java class that extends **HttpServlet** and override the **doGet()** or **doPost()** method to handle HTTP requests.
2. **Compile Servlet:** Compile the servlet class using the Java compiler.
3. **Deploy Servlet:** Place the compiled servlet class file in the appropriate directory structure within the web application directory (e.g., WAR file).
4. **Start Tomcat Server:** Start the Apache Tomcat server, which is a popular servlet container that implements the Servlet API.
5. **Access Servlet:** Access the servlet through a web browser by specifying the servlet URL, which typically includes the servlet's context path and servlet mapping.

- **Installation of Apache Tomcat Server:**

To install Apache Tomcat:

1. Download the Tomcat distribution from the Apache Tomcat website.
2. Extract the downloaded archive to a directory on your system.
3. Set environment variables if necessary.
4. Start the Tomcat server by running the startup script (**startup.sh** for Unix-based systems or **startup.bat** for Windows).

- **Installation of MySQL Database & Database Creation:**

To install MySQL:

1. Download the MySQL installer from the MySQL website.
2. Run the installer and follow the installation instructions.
3. Configure MySQL settings such as root password, port, etc.
4. Start the MySQL server.

To create a database and tables using SQL Yog:

1. Connect to the MySQL server using SQL Yog.
2. Execute SQL queries to create a database (**CREATE DATABASE**) and tables (**CREATE TABLE**).

- **JDBC Examples on Types:**

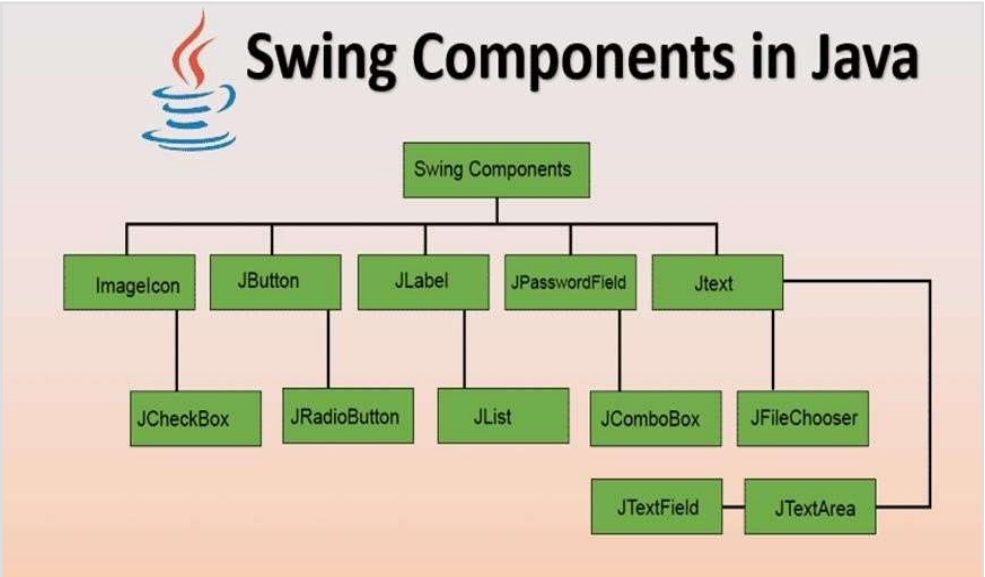
JDBC (Java Database Connectivity) is a Java API for connecting and executing SQL queries against databases. Examples of JDBC types include:

- **Type 1:** JDBC-ODBC Bridge (now deprecated)
- **Type 2:** Native API partly Java technology-enabled driver
- **Type 3:** Network Protocol driver
- **Type 4:** Native-protocol pure Java driver

- **JAVA Swing Packages for GUI**

Java Swing is a powerful framework that allows developers to create graphical user interfaces (GUIs) for their Java applications. It provides a rich set of components, such asbuttons, text fields, checkboxes, and more, which can be arranged and customized to build interactive and visually appealing interfaces. Swing is part of the Java Foundation Classes (JFC)and has been the primary GUI toolkit for Java developers for many years. Swing was introducedas a replacement for the older Abstract Window Toolkit (AWT), which had limited functionality and a less consistent look and feel across different platforms. Swing, on the other hand, was designed to be platform-independent, meaning that applications developed using Swing would have the same look and behavior on different operating systems.

One of the main advantages of Swing is its extensive library of components. Swing provides a wide range of components that can be used to build complex GUIs. These components are highly customizable, allowing developers to modify their appearance, behavior,and layout to suit their specific needs. For example, developers can change the color, font, and size of components, as well as define how they should respond to user interactions. Another key feature of Swing is its support for event-driven programming



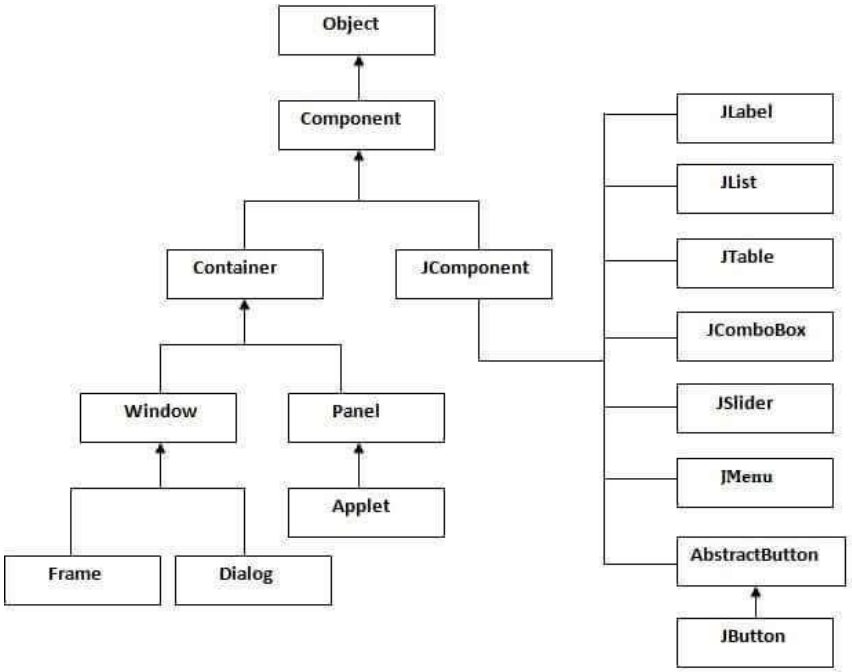


Fig.2.1 Java Swing Classes

2. Java JButton Class

In Java Swing, a JButton is a class that represents a button component. It is partof the Swing package (javax.swing) and is used to create clickable buttons in graphical user interfaces (GUIs).

Java JButton Example

```
import javax.swing.*;
public class ButtonExample {
public static void main(String[] args) {
    JFrame f=new JFrame("Button Example");
    JButton b=new JButton("Click Here");
    b.setBounds(50,100,95,30);
    f.add(b);
    f.setSize(400,400);
    f.setLayout(null);
    f.setVisible(true);
}
}
```

3. Java JLabel

In Java Swing, a JLabel is a class that represents a component used to display text or an image in a graphical user interface (GUI). It is part of the Swing package (javax.swing) and provides various methods and properties to customize its appearance and content.

JLabel label = new JLabel("Hello, World!");

Java JLabel Example

```
import javax.swing.*;
class LabelExample
{
public static void main(String args[])
   {
   JFrame f= new JFrame("Label Example");
   JLabel l1,l2;
   l1=new JLabel("First Label.");
   l1.setBounds(50,50, 100,30);
   l2=new JLabel("Second Label.");
   l2.setBounds(50,100, 100,30);
   f.add(l1); f.add(l2);
   f.setSize(300,300);
   f.setLayout(null);
   f.setVisible(true);
   }
```

Output:

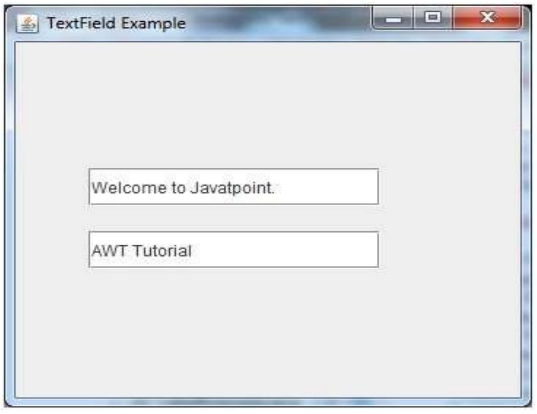
3.1.4.Java JTextField

In Java Swing, a JTextField is a class that represents a text input component in a graphical userinterface (GUI).

syntax:JTextField textField = new JTextField();

Java JTextField Example

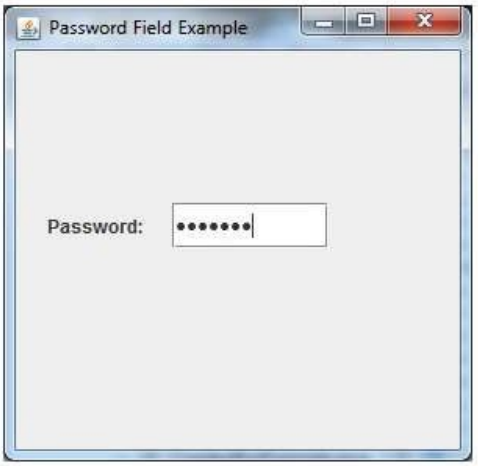
```
import javax.swing.*;
class TextFieldExample
{
public static void main(String args[])
   {
   JFrame f= new JFrame("TextField Example");
   JTextField t1,t2;
   t1=new JTextField("Welcome to Javatpoint.");
   t1.setBounds(50,100, 200,30);
   t2=new JTextField("AWT Tutorial");
   t2.setBounds(50,150, 200,30);
   f.add(t1); f.add(t2);
   f.setSize(400,400);
   f.setLayout(null);
   f.setVisible(true);
   }
   }
```



3.1.5.JTextArea

In Java, JTextArea is a class that represents a multi-line text component in the Swing library. It is a subclass of the JTextComponent class and provides functionality for displaying and editingmultiple lines of text.

```
import javax.swing.*;
public class PasswordFieldExample {
   public static void main(String[] args) {
   JFrame f=new JFrame("Password Field Example");
    JPasswordField value = new JPasswordField();
    JLabel l1=new JLabel("Password:");
      l1.setBounds(20,100, 80,30);
       value.setBounds(100,100,100,30);
          f.add(value);  f.add(l1);
          f.setSize(300,300);
          f.setLayout(null);
          f.setVisible(true);
}
}
```



2. What is JDBC?

JDBC stands for **J**ava **D**ata**b**ase **C**onnectivity, which is a standard Java API for database independent connectivity between the Java programming language and a wide range of databases.

The JDBC library includes APIs for each of the tasks mentioned below that arecommonly associated with database usage.

i. Making a connection to a database.
ii. Creating SQL or MySQL statements.
iii. Executing SQL or MySQL queries in the database.
iv. Viewing & Modifying the resulting records.
v. Fundamentally, JDBC is a specification that provides a complete set of interfaces that allows for portable access to an underlying database. Java can be used to write different types of executables, such as

   a. Java Applications
   b. Java Applets
   c. Java Servlets
   d. Java Server Pages (JSPs)
   e. Enterprise JavaBeans (EJBs).

JDBC (Java Database Connectivity) is a widely-used API in the Java ecosystemthat provides a standard way for Java applications to interact with databases. It offers a consistent and reliable approach to connect, query, and manipulate data in various database systems. JDBC follows a driver-based architecture, where each database vendor provides a JDBC driver that acts as a bridge between the Java application and the underlying database.

JDBC also supports transaction management, allowing you to group multiple database operations into a single atomic unit. By disabling auto-commit on the connection andusing the commit() and rollback() methods, you can control the transaction boundaries and ensure data integrity. Proper exception handling is crucial when working with JDBC. Database operations can encounter errors, such as connection failures, SQL syntax errors, or concurrency issues. JDBC provides a set of checked exceptions, such as SQLException, that you need to handle in your code. By catching and handling exceptions appropriately, you can provide informative error messages to the user and take corrective actions if necessary.

2.1. JDBC Architecture:

The JDBC architecture consists of several key components. The core components include the JDBC API, JDBC drivers, and the Java Database Connectivity Manager. The JDBC API provides interfaces and classes that define the standard methods for database operations. JDBC drivers, provided by database vendors, implement these interfaces and handle communication with the specific databases. The JDBC Manager handles the management of connections, statements, and result sets. Together, these components allow Java applications tointeract with databases seamlessly.

2.2. Establishing a Connection:

To establish a connection to a database using JDBC, you need to provide the necessary connection details such as the database URL, username, and password. The JDBC driver uses this information to establish a connection to the database server. Once the connection is established, you can perform various database operations, such as executing queries, updating records, or calling stored procedures.

2.3. Executing SQL Statements:

JDBC provides different types of statements for executing SQL queries and updates. The most basic type is the `Statement` interface, which allows you to execute simple SQL statements. For more complex queries or statements that require input parameters, you can use`PreparedStatement`. It provides the ability to parameterize SQL queries, enhancing security and performance. Additionally, JDBC supports the execution of stored proceduresthrough the`CallableStatement` interface.

2.4. Handling Result Sets:

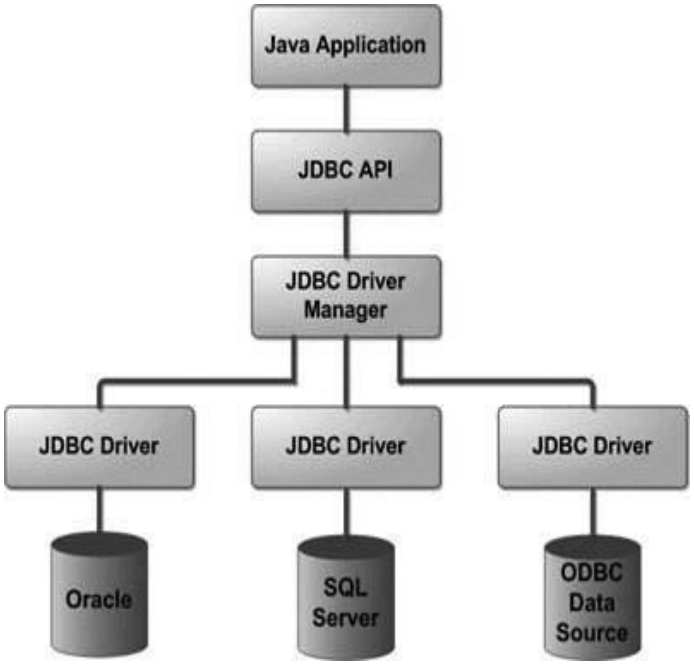
When executing a SQL query that returns a result set, JDBC provides the `ResultSet` interface to retrieve and process the query results. The result set allows you to navigate through the returned rows, access column values, and perform operations such as filtering, sorting, and aggregation. You can use methods like `next()`, `getString()`, `getInt()`, and others to extract data from the result set and work with it in your Java code.

2.5. Exception Handling and Resource Cleanup:

Proper exception handling is essential when working with JDBC. Database operations can encounter errors, such as connection failures, SQL syntax errors, or concurrency issues. JDBC provides a set of exceptions that you can catch and handle appropriately, allowing you to take corrective actions or provide meaningful error messages to the users. Additionally, it's crucial to release database resources properly. Closing connections, statements, and result sets using the `close()` method is necessary to free up system resources and avoid potential memory leaks.

2.6. Transactions and Batch Processing:

JDBC supports transaction management, allowing you to group multiple database operations into a single atomic unit. By starting a transaction, executing multiple statements, and then committing or rolling back the transaction, you can ensure data integrity and consistency. JDBC also provides batch processing capabilities, enabling you to execute a set of SQL statements as a batch, reducing network round-trips and improving performance.



3.SQL:

SQL (Structured Query Language) is a programming language designed for managing and manipulating relational databases. It is used to create, modify, and query databases, and is commonly used in web applications and enterprise software. SQL allows usersto retrieve and manipulate data stored in a relational database, as well as manage the structure and schema of the database itself.

SQL is used to perform a wide range of tasks, including creating and managing tables, inserting and updating data, and querying data from multiple tables. SQL also includes advanced features such as subqueries, joins, and aggregations, which allow users to perform complex queries and analysis.

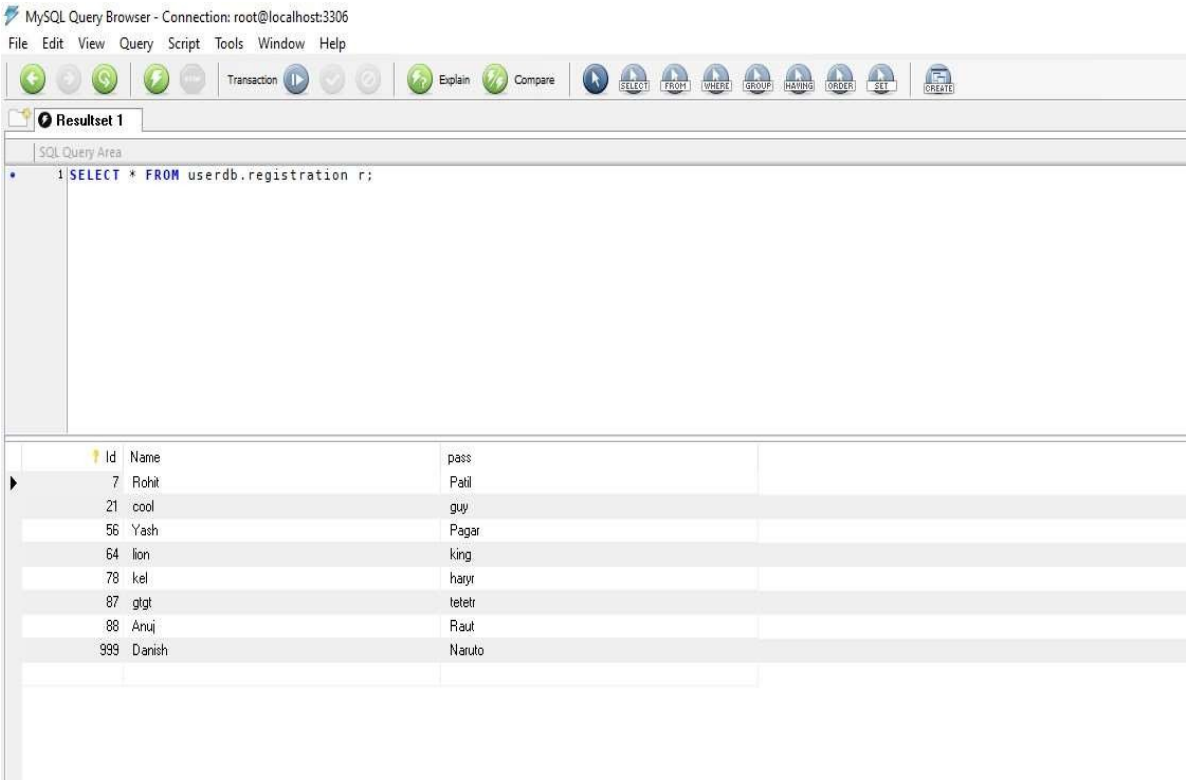
Some popular database management systems that support SQL include MySQL, Oracle, SQL Server, PostgreSQL, and SQLite.

MySQL is the world's most popular open-source database. According to DB-Engines, MySQL ranks as the second-most-popular database, behind Oracle Database. MySQL powers many of the most accessed applications, including Facebook, Twitter, Netflix, Uber, Airbnb, Shopify, and Booking.com.

Since MySQL is open source, it includes numerous features developed in close cooperation with users over more than 25 years. So it's very likely that your favorite applicationor programming language is supported by MySQL Database.

Fig. 2.3.1

Fig. 2.3.2



4. Use of MySQL

It is one of the most popular open-source relational database management systems (RDBMS) used worldwide. It provides a powerful and scalable platform for storing and managing structured data efficiently. It is known for its reliability, performance, and ease of use, making it a preferred choice for a wide range of applications, from small-scale web applications to large enterprise systems.

One of the key advantages of It is its wide platform compatibility. It runs on various operating systems, including Windows, macOS, Linux, and UNIX, making it accessible to developers on different platforms. Additionally, It supports multiple programming languages, including Java, PHP, Python, and more, making it versatile and suitable for a variety of development environments.

It offers a rich set of features that enable efficient data management. It supports a wide range of data types, including numeric, string, date/time, and spatial data types, allowingdevelopers to store and manipulate different types of information. It also provides various indexing techniques to optimize data retrieval, including B-tree, hash, and full-text indexing. Indexes improve query performance by facilitating faster data access and reducing the need for full table scans.It provides a range of tools and utilities to assist developers and administrators.

To use MySQL in JDBC, you need to follow these steps:

4.1. Load the MySQL JDBC Driver:

Before establishing a connection to a MySQL database, you need to load the MySQL JDBC driver class using the `Class.forName()` method. The driver class name for MySQL is typically `com.mysql.jdbc.Driver` or `com.mysql.cj.jdbc.Driver`, depending on theversion of MySQL and the driver you are using. Loading the driver class ensures that it is available for establishing a database connection.

4.2.Establish a Connection to MySQL:

Once the MySQL JDBC driver is loaded, you can establish a connection to the MySQL database using the `DriverManager.getConnection()` method. You need to provide the database URL, username, and password as parameters to this method. The database URL for MySQL typically follows the format `jdbc:mysql://hostname:port/databaseName`, where you replace `hostname`, `port`, and `databaseName` with the appropriate values.

4.3.Execute SQL Statements:

After successfully establishing a connection, you can create a `Statement` or `PreparedStatement` object to execute SQL statements. The `Statement` interface allows you toexecute simple SQL statements, while the `PreparedStatement` interface is used for executing parameterized SQL queries. You can create these objects by calling the `createStatement()` or
`prepareStatement()` method on the `Connection` object.

4.4.Process the Result:

If executing a query, you can retrieve the result in the form of a `ResultSet` object. The `ResultSet` provides methods to navigate through the result set, retrieve column values, and perform other operations on the returned data. You can use methods like `next()`,
`getString()`, `getInt()`, etc., to iterate over the result set and extract the data.

4.5.Handle Exceptions and Close Resources:

It's important to handle any exceptions that may occur during database operations. JDBC provides a set of exception classes, such as `SQLException`, that you can catch and handle appropriately. Additionally, it's necessary to close the database resources, including the

`ResultSet`, `Statement`, and `Connection`, using the `close()` method. This step ensures properresource cleanup and prevents memory leaks.

By using the MySQL JDBC driver in conjunction with the JDBC API, you can establish a connection to a MySQL database, execute SQL statements, retrieve results, and perform other database operations seamlessly within your Java application. This integration allows you to leverage the power and capabilities of MySQL in your Java projects, enabling efficient and reliable data management.

Fig. 2.4 MySQL

- **FrontEnd Technologies :**

HTML Basics:

HTML (Hypertext Markup Language) is the standard markup language used to create web pages. It consists of a series of elements (tags) that define the structure and content of a web page.

Commonly Used HTML Tags:

1. **<html>**: Defines the root element of an HTML document.
2. **<head>**: Contains metadata about the document, such as title, links to stylesheets, etc.
3. **<title>**: Defines the title of the document.
4. **<body>**: Contains the content of the document.
5. **<h1>**, **<h2>**, ..., **<h6>**: Define headings of different sizes.
6. **<p>**: Defines a paragraph.
7. **<a>**: Defines a hyperlink.
8. **<img>**: Embeds an image in the document.
9. ** **, ****: Defines divisions or spans for styling and layout purposes.
10. **<ul>**, **<ol>**, **<li>**: Defines unordered and ordered lists and list items.

HTML5 Validations:

HTML5 introduced built-in form validation features that can be used to validate user input before submitting a form. This includes attributes like **required**, **pattern**, **min**, **max**, etc., which can be added to form inputs to enforce validation rules.

CSS Basics:

CSS (Cascading Style Sheets) is a style sheet language used to describe the presentation of a document written in HTML. It allows developers to control the layout, appearance, and styling of web pages.

Commonly Used CSS Properties:

1. **color**: Sets the text color.
2. **font-family**: Defines the font family for text.
3. **font-size**: Sets the font size.
4. **margin**, **padding**: Controls the spacing around elements.

5. **border**: Defines the border around elements.
6. **background-color**: Sets the background color of elements.
7. **display**: Specifies how an element is displayed.
8. **position**: Controls the positioning of elements.

Introduction to JSP (JavaServer Pages) Technology:

JSP is a technology used to create dynamic web pages by embedding Java code within HTML pages. It allows developers to write server-side logic directly within the HTML markup, enabling the creation of dynamic content.

Importance of JSP over Servlets:

1. **Simplicity:** JSP simplifies the development process by allowing developers to embed Java code directly into HTML, reducing the need for separate Java files.
2. **Separation of Concerns:** JSP promotes the separation of presentation logic (HTML) from business logic (Java code), making code maintenance and collaboration easier.
3. **Rapid Development:** With JSP, developers can quickly create dynamic web pages without writing extensive Java code.
4. **Reuse of Components:** JSP allows the reuse of components such as custom tags and JavaBeans, enhancing code modularity and reusability.

JSP Life Cycle Methods:

JSP Life Cycle Phases:

1. **Translation:** JSP file is translated into a servlet.
2. **Compilation:** Servlet code is compiled into bytecode.
3. **Initialization:** Servlet instance is initialized.
4. **Request Handling:** Servlet services client requests.
5. **Destroy:** Servlet instance is destroyed.

Execution Flow of JSP Pages:

1. Client sends a request to the web server for a JSP page.
2. Web server passes the request to the JSP container.
3. JSP container translates the JSP page into a servlet.
4. Servlet is compiled into bytecode.
5. Servlet instance is initialized and executed to generate dynamic content.
6. Generated HTML content is sent back to the client by the web server.

Various Tags in JSP:

JSP provides several standard tags for performing common tasks within JSP pages, such as including other resources, controlling flow, iterating over collections, etc.

Commonly Used JSP Tags:

1. **<%@ include file="filename.jsp" %>**: Includes the content of another JSP file.
2. **<jsp:include page="filename.jsp" />**: Includes the content of another JSP file at runtime.
3. **<jsp:forward page="filename.jsp" />**: Forwards the request to another resource.
4. **<c:forEach>**: Iterates over a collection.
5. **<c:if>**, **<c:choose>**, **<c:when>**, **<c:otherwise>**: Conditional processing tags.
6. **<jsp:useBean>**, **<jsp:setProperty>**, **<jsp:getProperty>**: Used for working with JavaBeans.

JSP provides a powerful mechanism for creating dynamic web content and simplifying the development of web applications by combining the power of Java with the simplicity of HTML.

CHAPTER 3 :

Training Activity

Requirements Analysis:

Testing begins in the requirements phase of the software development life cycle.

Design Analysis:

During the design phase, testers work with developers in determining what aspects of a design ar testable and under what parameter those tests work.

Test Planning:

Test Strategy, Test plan(s), and Test Bed creation.

Test Development:

Test procedures, Test Scenarios, and Test Cases & Test Scripts to use in testing software.

Test Execution:

Testers execute the software based on the plans and tests and report any errors found to the development team.

Test Reporting:

Once testing is completed, testers generate metrics and make final reports on their test effort whether or not the software tested is ready for release.

CHAPTER 4 :

WEEKLY OVERVIEW OF INTERNSHIPACTIVITIES

Week 1:

Day	Date	Work done
Monday	01/01/2024	Introduction to Java Programming ,Features of Java. Application development in JAVA –understanding JDK,JRE, JVM WALK: Contest on - Data types, Operators, Control Structures:- Conditionals statements, Loops, Arrays - 2 to 4 Problems, Hands On – Data types,Conditionals statements, Loops, Arrays - 6 to 8 Pr
Tuesday	02/01/2024	Concepts – String Class and Methods of String Class handling Unicode characters, Comparing Strings, Concatenation of String Concepts – String Buffer, String Builder CRAWL: Hands On – String Class - 6 to 8Problems.
Wednesday	03/01/2024	Concepts - Arrays Class and Methods CRAWL: Hands On – Arrays Class Concepts - Properties Class and Objects Concepts – Exception handling, try-catch, nested exceptions, finally, throw, throws. CRAWL: mini-app using Collections WALK: Contest on – Collections - 4 to 6 Problems
Thursday	04/01/2024	Concepts – Classes & Objects, Real world connections Concepts - Inheritance (IS-A), Relationships & Collaboration (HAS-A), Polymorphism (Overloading and Overriding), Abstraction (Interface & Abstract Class), Encapsulation(Packages & Access modifiers
Friday	05/01/2024	Concepts :- Packages, Advantages of using Packages, Types of Packages,Creating our own packages

Week 2:

Day	Date	Work done
Monday	08-01-202	Concepts: Wrapper Class and Methods of Wrapper Class, Collection Framework : ArrayList,LinkedList,Vector, Hashset class and Methods.
Tuesday	09-01-202	Files & I/O Streams. Files & Streams (About all I/O streams with various operations)Packages & Importance of packages
Wednesday	10-01-202	Web Application development using Servlets Introduction of Servlet API, Web Server (Tomcat 8.0) Configuration), Steps to Creation & Execution of Servlets using Tomcat server. Installation of apache software, programs on servlets (Home Page, Login, Registrations and update profile)

Thursday	11-01-202	Installation of MYSQL Data Base & Data Base Creation & Tables in SQL Yog. JDBC examples on types, stages, statement and prepared statement.
Friday	12-01-202	Guest lecture on Web application security assessment

Week 3 :

Day	Date	Work done
Monday	15/01/2024	HTML, HTML-5- validations & CSS Various tags of html with examples, different ways, Creation of Web Pages using html.
Tuesday	16/01/2024	Introduction of JSP technology, Importance of JSP over Servlets, JSP Life Cycle methods, Execution flow of JSP pages, Various tags in JSP.
Wednesday	17/01/2024	JSP Life Cycle methods, execution flow of JSP pages, Various tags in JSP. Scripting Tags, Directive Tags, Action Tags with all possible attributes
Thursday	18/01/2024	NO WORK DONE
Friday	19/01/2024	Guest lecture on Manual testing and automation testing

Weak 4:

Day	Date	Work done
Monday	22/01/2024	Introduction of JDBC API, types of drivers, how to connect with DB using jdbc drivers
Tuesday	23/01/2025	Installation of apache software, Intoduction to servlet, Life cycle of servlet HTTP Protocol, GET and POST method. Program on Servlet.
Wednesday	24/01/2026	Development of Projects : Creating Portfolio WebPage , Library Management System , Atm InterFace
Thursday	25/01/2027	Development of Project : Simple Calculator Using java, Tic Tac Toe Game, Number Guessing Game.

CHAPTER 5 :

PLAN OF THE INTERNSHIP PROGRAM

a) Duration of the Internship: 4 Weeks

b) Start Date of Internship: 1 Jan 2024

c) End Date of Internship: 31 Jan 2024

CHAPTER 6:

PROBLEM STATEMENT

Title of Project:
1. “Library Management System”
2. “Adding stocks of Books in Library”
3. “Display the collection of books”
4. “Editing the records of Books”
5. “Deleting the Records”

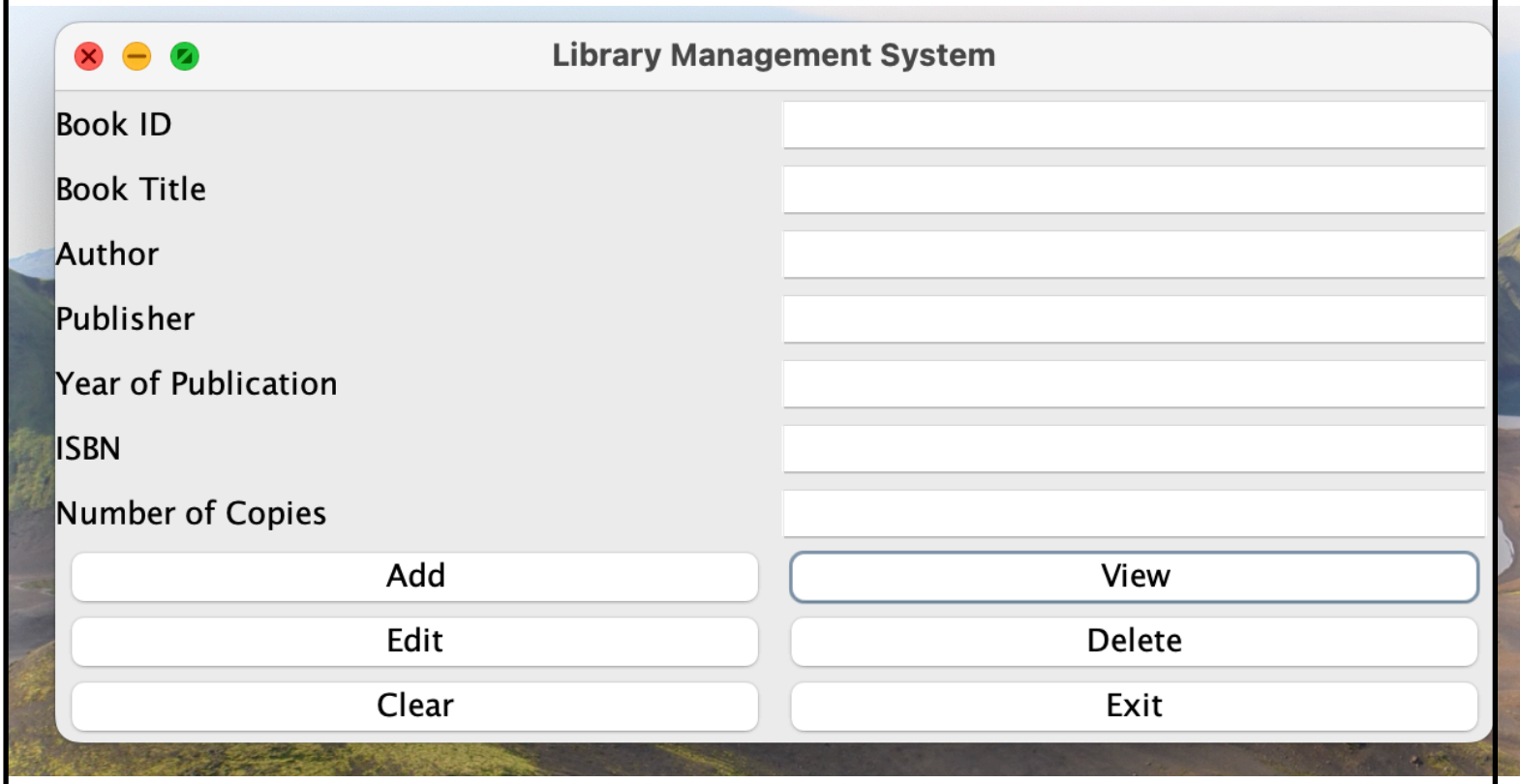
Aim of project:
1. “Adding stocks of Books in Library”
2. “Display the collection of books”
3. “Editing the records of Books”
4. “Deleting the Records”

Software : For Front-End : Eclipse

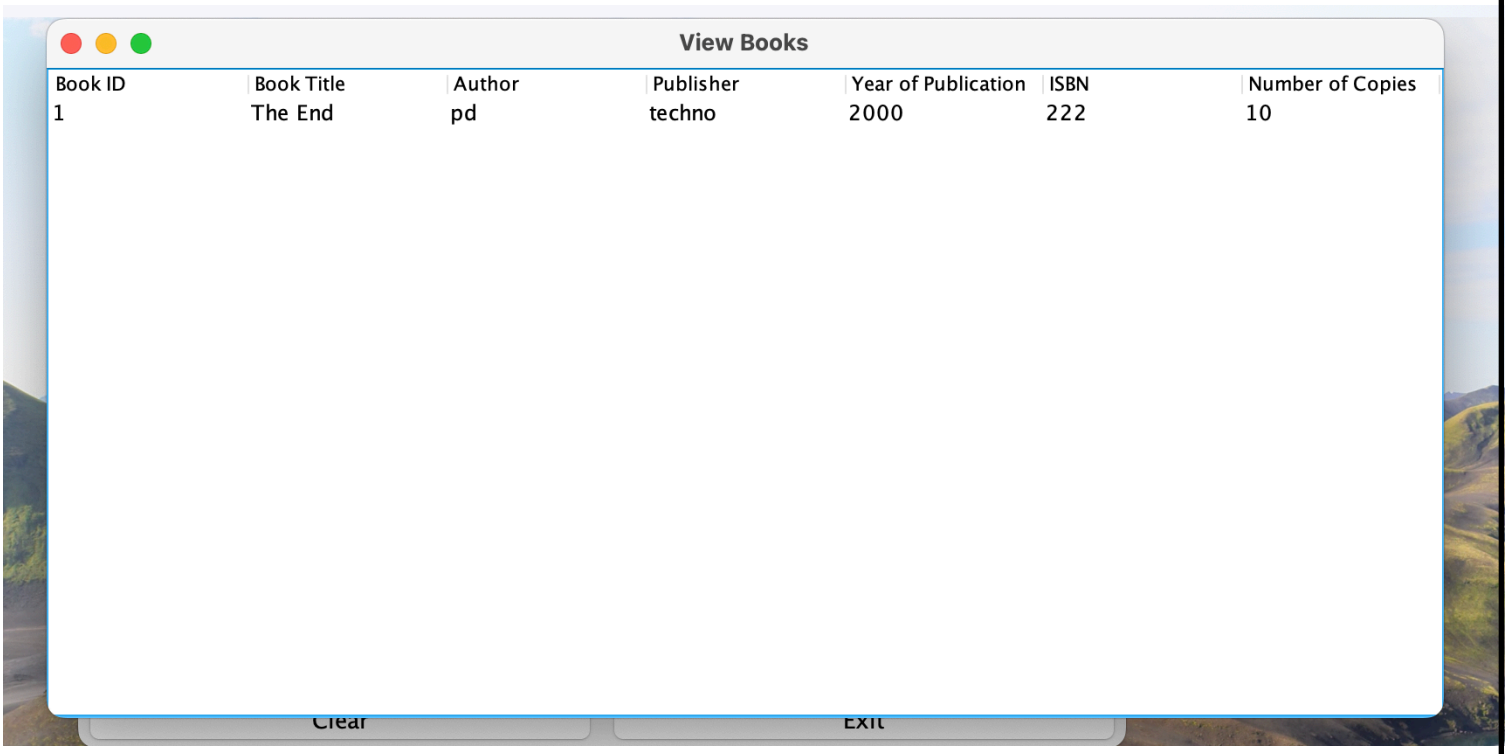
For Backend : MySQL

CHAPTER 7: RESULT

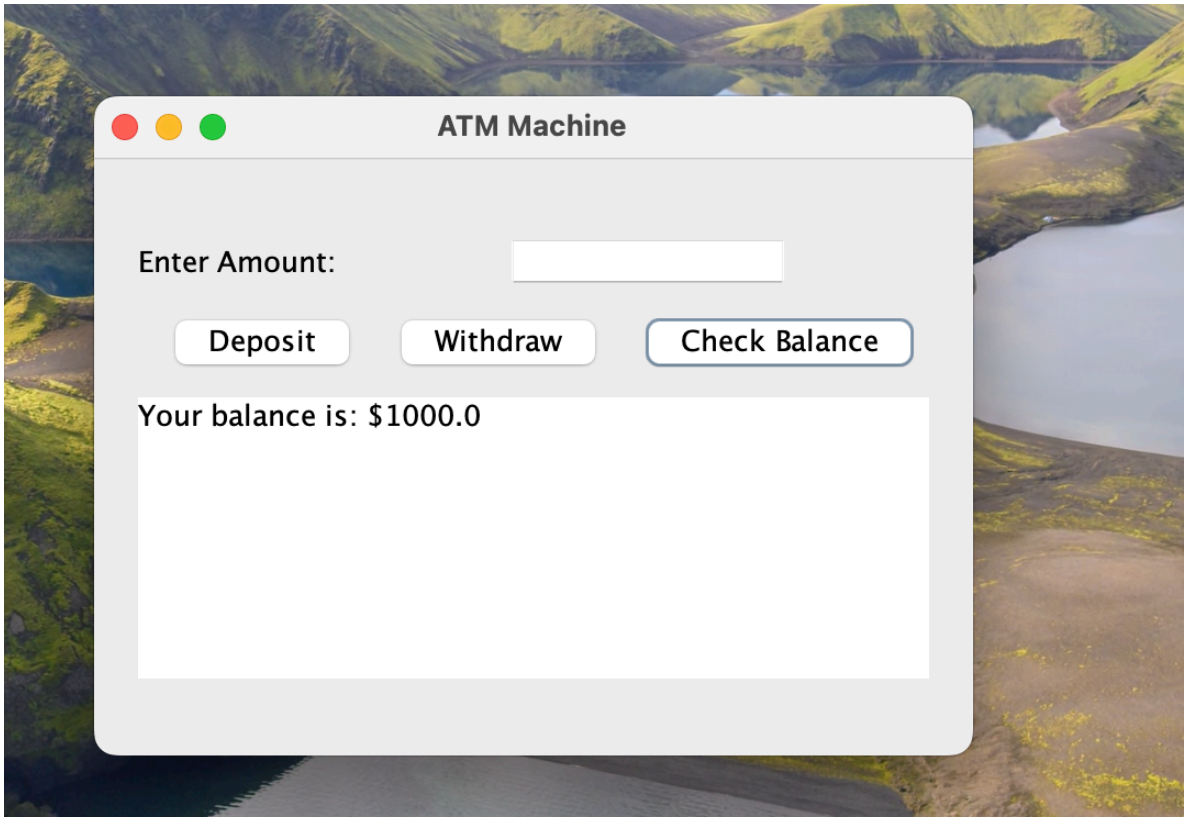
"Library Management System"



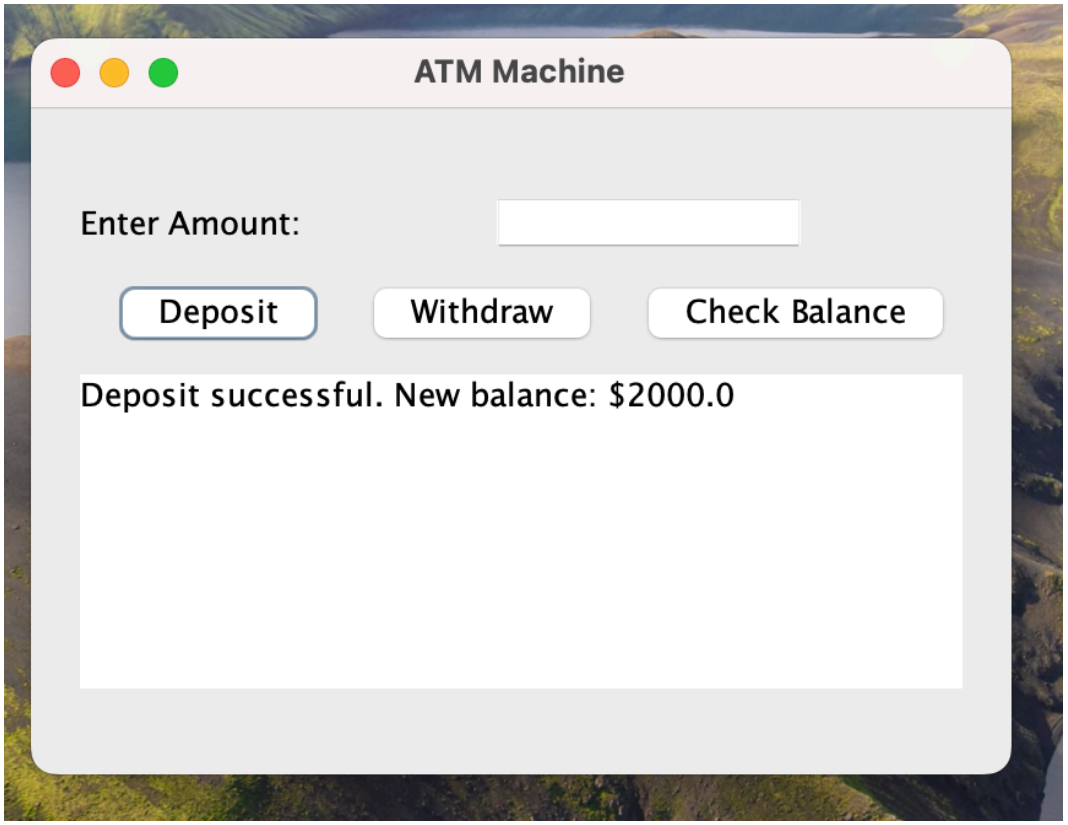
View:



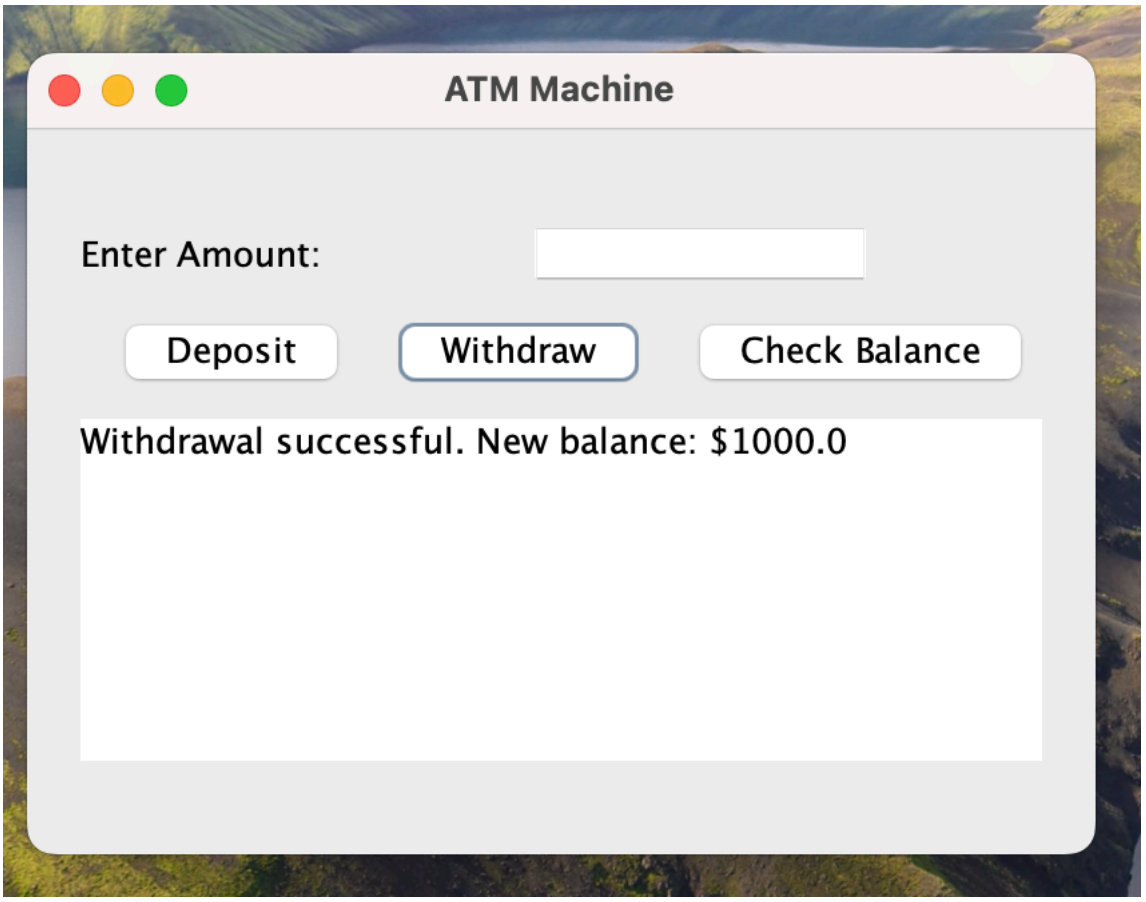
“Atm InterFace”
1.Check Balance



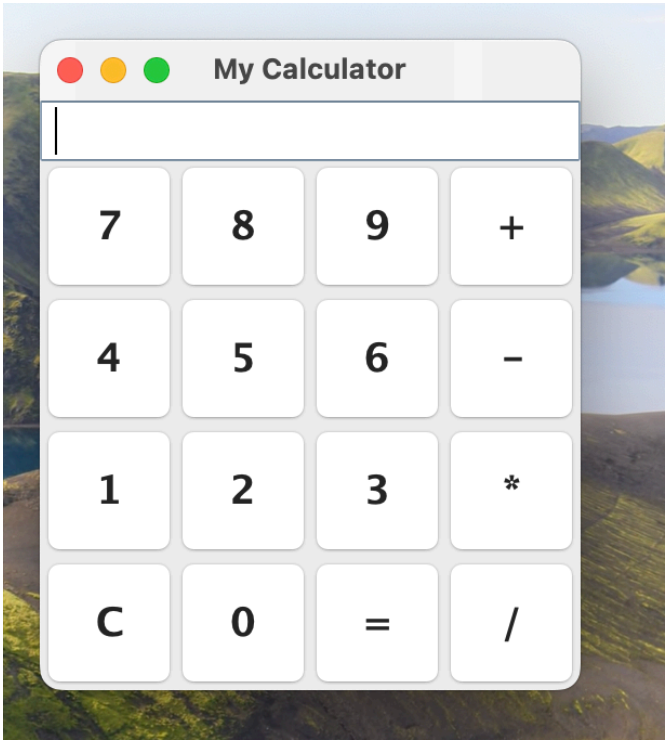
2.Deposit:



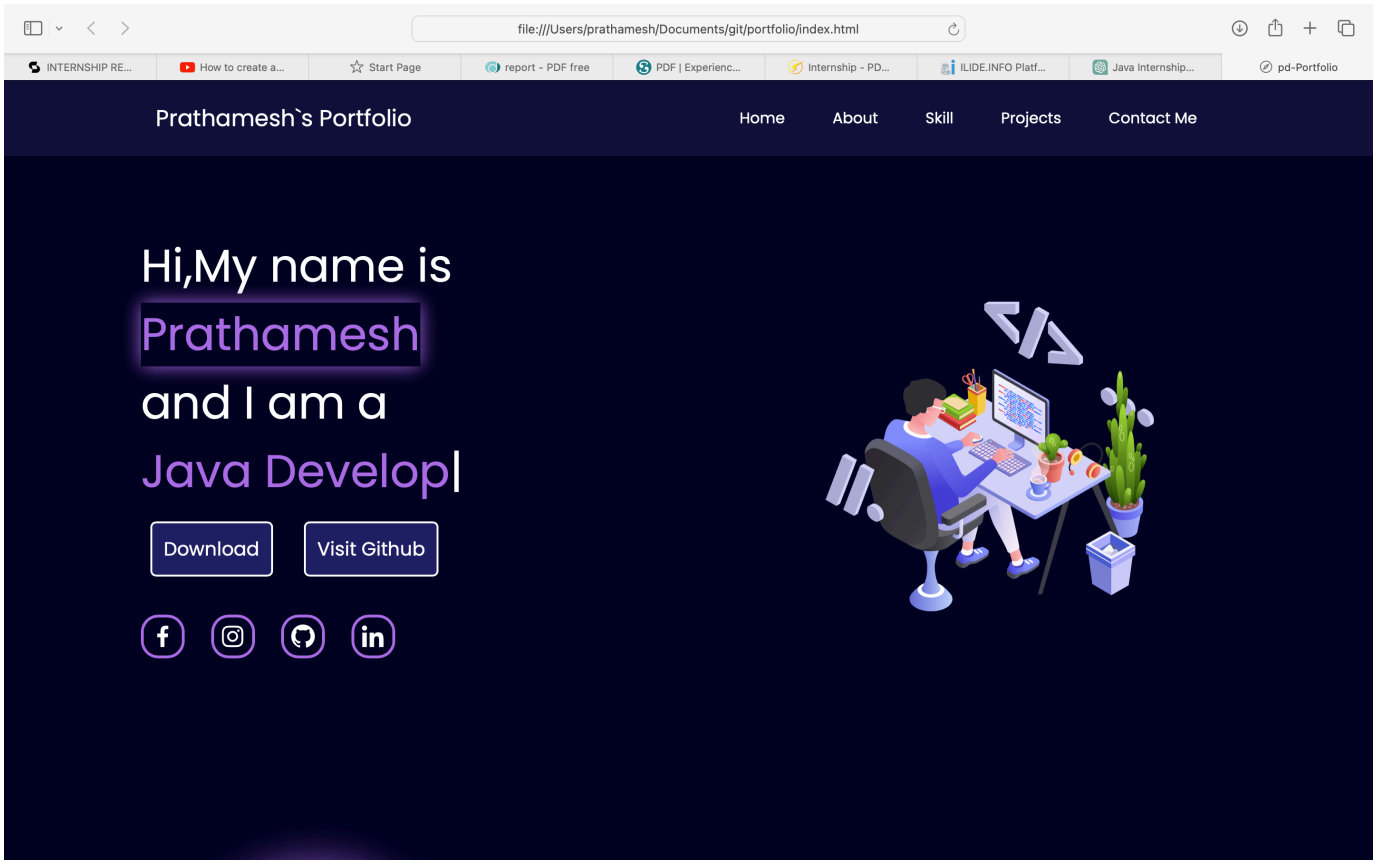
3.Withdraw:

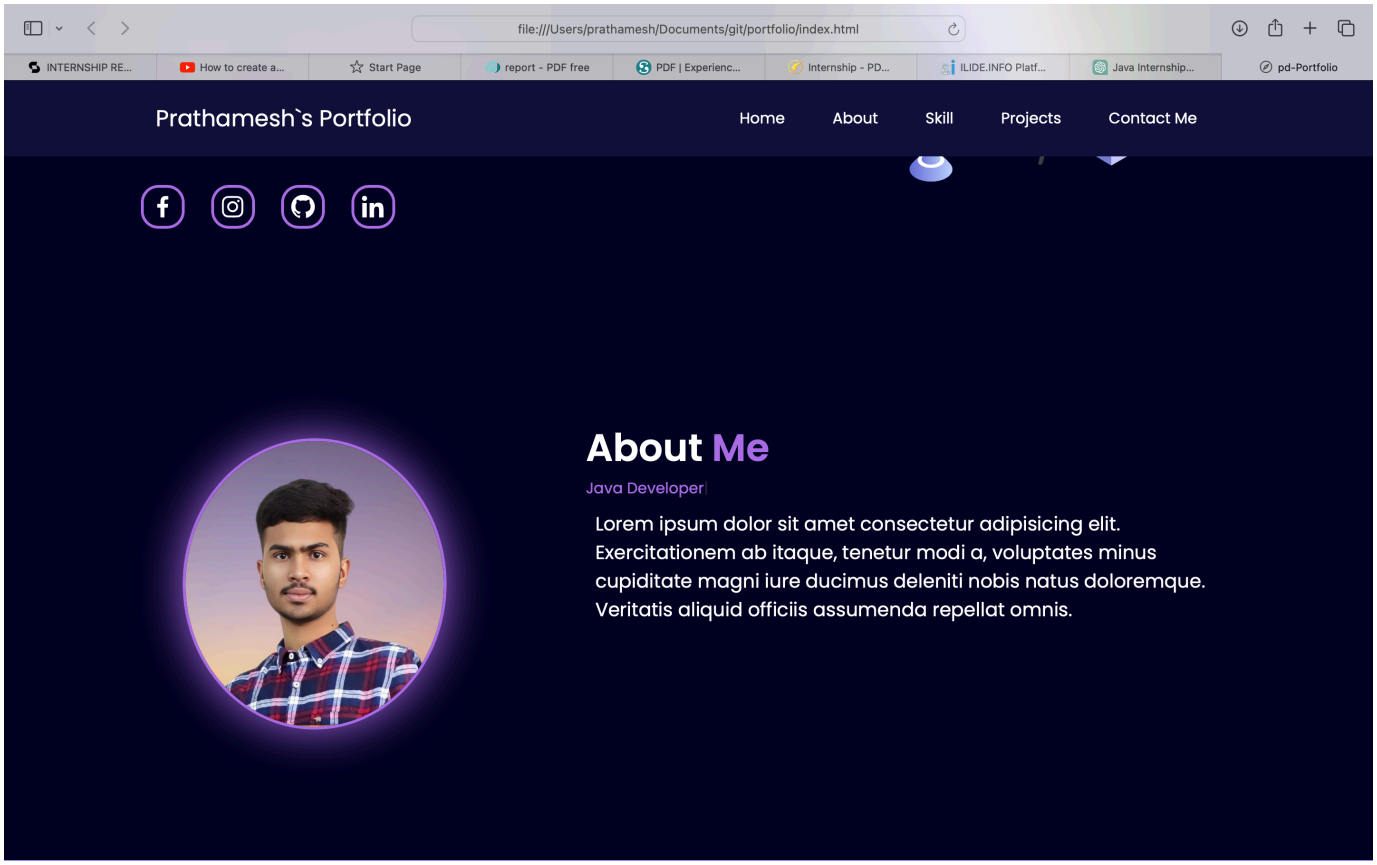


“Simple Calculator”



“Portfolio by Using HTML,CSS, JS”





CHAPTER 8:

CONCLUSION

In this internship I gained Java knowledge and experience in real-life situations**.**

Java internships are relevant because being able to prove your experience in the tech industry is critical. A background in practical work in Java has a lot of significance when finding a job in Java or in IT fields.

Experience always adds up into your path of getting a successful career. Hence, trying something new adds up into your experience which eventually turns out to be beneficial for your career.

My Java development internship has been an invaluable learning journey. Throughout this experience, I've had the opportunity to delve deep into Java programming, gaining practical insights into software development processes, problem-solving, and teamwork. From developing robust applications to troubleshooting intricate bugs, each task has enhanced my skills and expanded my knowledge base.

In closing, I want to express my gratitude to TechnoHacks EduTech for providing me with this enriching opportunity. It has been a privilege to be part of such a dynamic and innovative team. I look forward to staying connected and continuing to contribute to the world of Java development."

CHAPTER 9:

REFERENCES

1. Java Development Internship Study material :
   Website : https://www.geeksforgeeks.org/best-full-stack-developer-courses/

2. Introduction of Java Development
   https://www.coursera.org/specializations/java-fullstack

3. . https://www.w3schools.com/java/java_intro.asp

4. Database Connectivity :
   https://www.mysql.com/downloads/

5. Software Used For Internship Projects:
   eclipse : https://eclipseide.org/
   mysql : https://www.mysql.com/downloads/