

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590018



An Internship Report

on

Game Tracker Pro

Bachelor of Engineering in Computer Science and Engineering
of Visvesvaraya Technological University, Belagavi

Submitted by

CHIRAG N SUNDAR 1RN21CS047

Under the Guidance of:

Coordinator:

Mr. Phanikanth K V

Assistant Professor

Dept. of CSE

Guide:

Dr. Sampada K S

Assistant Professor

Dept. of CSE



Department of Computer Science and Engineering

RNS Institute of Technology

(Accredited by NBA upto 30-06-2025)

Channasandra, Dr.Vishnuvardhan Road, Bengaluru-560098

2022-2023

RNS INSTITUTE OF TECHNOLOGY

Channasandra, Dr. Vishnuvardhan Road, Bengaluru-560098

DEPARTMENT OF COMPUTER SCIENCE ENGINEERING

(Accredited by NBA upto 30-06-2025)



CERTIFICATE

Certified that the Internship work entitled **Game Tracker Pro** has been successfully carried out by **Chirag N Sundar** bearing USN **1RN21CS047** bonafide students of **RNS Institute of Technology** in partial fulfillment of the degree in **Bachelor of Engineering in Computer Science and Engineering of Visvesvaraya Technological University, Belagavi** during academic year **2022-2023**. The internship report has been approved as it satisfies the academic requirements in respect of internship work for the said degree.

Signature of Coordinator

Mr. Phanikanth K V

Assistant Professor
Dept. of CSE

Signature of HOD

Dr. Kiran P

Professor and HoD
Dept. of CSE

Signature of Principal

Dr. M K Venkatesha

Principal
RNSIT



Figure 1: Infosys Springboard Java Programming Fundamentals certificate

Acknowledgement

At the very onset, I would like to place on record my gratitude to all those people who have helped me in making this Internship work a reality. Our Institution has played a paramount role in guiding us in the right direction. I would like to profoundly thank **Sri. Satish R Shetty**, Managing Director, RNS Group of Companies, Bengaluru for providing such a healthy environment for the successful completion of this Internship Project work.

I would like to thank our beloved Principal, **Dr. M K Venkatesha**, for providing the necessary facilities to carry out this work. I am extremely grateful to **Dr. Kiran P**, Professor and Head, Department of Computer Science and Engineering for having accepted to patronize me in the right direction with all his wisdom.

I would like to express my sincere thanks to our Coordinator and guide, **Mr. Phanikanth K V**, Assistant Professor, Department of Computer Science and Engineering and **Dr. Sampada K S**, for his/her constant encouragement that motivated me to the successful completion of this work. Last but not the least, I am thankful for all the teaching and non-teaching staff members of the Computer Science and Engineering Department for their encouragement and support throughout this work.

CHIRAG N SUNDAR

1RN21CS047

Abstract

With the explosive growth of the gaming industry, there is a significant need for tools that can help users keep track of their games and game progress.

The Game Tracker Pro project is a Java-based application that enables users to manage game data stored in a MySQL database. The program implements a 3-tier architecture with a client, server, and database. Users can create, read, update, and delete game data based on the game's title through a login-based menu system. The system allows users to securely manage their data, with the client-side interface designed in Java and the database backend implemented in MySQL. The program is designed for ease of use, and its intuitive interface makes it ideal for both novice and experienced users. Overall, the Game Tracker Pro project provides a powerful yet user-friendly solution for managing game data.

Contents

Acknowledgement	i
Abstract	ii
1 Introduction	1
1.1 Problem Statement	2
1.1.1 Problem Definition	2
1.1.2 Problem Formulation	2
2 Requirement Analysis, Tools & Technologies	3
2.1 Functional Requirements	3
2.2 Non-functional Requirements	3
2.3 Hardware Requirements	4
2.4 Software Requirements	4
2.4.1 Eclipse IDE for Java	4
2.4.2 MySQL database and MySQL workbench	5
3 Design & Implementation	7
3.1 Methodology	7
3.1.1 Code segments:	8
3.2 Libraries	9
3.2.1 java.sql	9
3.2.2 java.util	10
3.3 Algorithms	12

4	Observations & Results	13
5	Conclusion & Future Enhancements	15
5.1	Conclusion	15
5.2	Future Enhancements	15

Chapter 1

Introduction

This project is aimed at implementing a 3-tier architecture, which will be divided into three separate units - client-frontend, server-backend, and database. The front end of the program will be handled by Eclipse in its console, the server will be implemented using Java, and MySQL will be used as the database management system. The objective of the project is to develop a user-friendly and simple application to manage game data. The application will be programmed in Java, which is a popular and widely used programming language. MySQL, on the other hand, is a robust and efficient database management system that will be used to store and manage game data. When you run the program, it will ask the user for their username and password, if the account exists and the password is correct it logs into the software. If wrong credentials are entered it will allow the user for 3 tries to enter the correct credentials, and after 3 tries it will exit the program if the credentials are still incorrect. Once a user logs in, the application will present them with a menu of all possible actions that can be performed. The user can select an option from the menu to perform the desired action. The functionalities of the application will include the ability to add, modify, delete, and view game data. Overall, the project aims to develop a reliable and efficient application that simplifies the management of game data. With its user-friendly interface and comprehensive set of features, the application will provide a hassle-free experience to its users.

1.1 Problem Statement

1.1.1 Problem Definition

Many gamers struggle to keep track of their game collections, leading to frustration and missed opportunities to enjoy their favorite titles. They often lack an organized system to store important information such as the game title, price, and platform, making it difficult to manage and make informed decisions about their purchases.

1.1.2 Problem Formulation

Why we need the project: Gaming is a popular hobby enjoyed by millions worldwide, and having a comprehensive game list is essential for gamers to make informed decisions about their gaming purchases. By developing a user-friendly platform for gamers to store and manage their game collections, we can provide an easy solution to the challenge of keeping track of games. This project has the potential to bring joy and convenience to gamers worldwide, as well as help them make informed decisions about their gaming purchases.

Chapter 2

Requirement Analysis, Tools & Technologies

2.1 Functional Requirements

- Database connection
- User Identification
- Create a new table called "games" if it doesn't exist
- Menu-Based Program
- Read/display data list
- Update data
- Delete data
- Add new data
- Usable Front-end

2.2 Non-functional Requirements

- Reliability: Should be able to run without causing errors, and any error must be aptly handled
- Durability: Should be serviceable for future adjustments
- Secure login system: Should not allow leaking of User Information

- User-friendly interface
- Speed: Must run the Program without much time delays or lag
- Compatibility with MySQL database

2.3 Hardware Requirements

The Hardware requirements are very minimal and the program can be run on most machines. Table 2.1 gives details of hardware requirements.

Table 2.1: Hardware Requirements

Processor	Intel Core i3 processor
Processor Speed	1.70 GHz
RAM	4 GB
Storage Space	40 GB
Monitor Resolution	1024*768 or 1336*768 or 1280*1024

2.4 Software Requirements

The software requirements are a description of the features and functionalities of the system. Table 2.2 gives details of software requirements.

Table 2.2: Software Requirements

Operating System	Windows 7 and above
IDE	Eclipse
Tools	MySQL
Libraries	JDBC, SQL workbench

2.4.1 Eclipse IDE for Java

Eclipse is a popular integrated development environment (IDE) used primarily for developing Java applications but also supports other programming languages. It offers a comprehensive set of tools for software development, including code editing, debugging, testing, and version control. Eclipse also

has a modular architecture that allows developers to add functionality through a vast array of plug-ins. It is open-source and has a large and active community of developers contributing to its development and support. Eclipse is widely used in industry and academia for various software development projects.

2.4.2 MySQL database and MySQL workbench

MySQL is an open-source relational database management system. It is one of the most widely used databases in the world, and it is used by many large and small organizations to store and manage their data. MySQL Workbench is an integrated development environment (IDE) that allows users to design, develop, and manage MySQL databases. It includes features such as visual database design, SQL development and execution, and database administration tools. MySQL and MySQL Workbench are often used together to create, manage, and optimize MySQL databases for various applications.

JDBC Connector

- JDBC Connector for MySQL is a Java-based API that allows Java programs to connect and interact with MySQL databases.
- It provides a standard set of interfaces for developers to work with databases, including connecting to a database, executing SQL queries, and retrieving data.
- To use JDBC Connector for MySQL in a Java project, developers need to add the connector JAR file to the project's classpath.
- Eclipse IDE is a popular integrated development environment for Java development, providing a range of features and tools to streamline the development process.
- Eclipse provides built-in support for JDBC, including the ability to create JDBC connections, execute SQL statements, and view the results of queries.
- Developers can use Eclipse's Database Development perspective and Data Source Explorer to manage JDBC connections and interact with MySQL databases.

- MySQL Workbench is a graphical tool for working with MySQL databases, providing a range of features for database design, modeling, and administration.
- MySQL Workbench also includes built-in support for JDBC, allowing developers to connect to MySQL databases and execute SQL queries directly from the tool.

Chapter 3

Design & Implementation

3.1 Methodology

The project is implemented using a structured approach. The program establishes a connection to the database(MYSQL) using JDBC and prompts the user for their username and password, if the account exists and the password is correct it logs into the software. If wrong credentials are entered it will allow the user for 3 retries to enter the correct credentials, and after 3 retries if the user still doesn't enter the correct credentials, it will exit the program. If the credentials entered are correct, the application starts. If the "games" table does not exist, the program creates it. The program presents the user with a menu of options, including reading/displaying the current game list, updating data, deleting data, adding new data, and exiting the program.

- To read data from the database, the program executes a SELECT statement and prints the results to the console.
- To update data, the program prompts the user for the game title to be updated, the new platform, and the new price. Then it executes an UPDATE statement.
- To delete data, the program prompts the user for the game title to be deleted and executes a DELETE statement.
- To add new data, the program prompts the user for the game title, platform, and price then executes an INSERT statement.

The program handles errors and exceptions gracefully, such as checking if the game title already exists before adding new data, checking whether the user inputs are empty, and validating if the user enters a valid price. The program uses prepared statements to execute SQL queries and handle user input to prevent SQL injection attacks. It also catches SQLE exceptions and displays informative error messages to the user. Overall, the project provides a simple example of how to interact with a MySQL database using Java and JDBC.

3.1.1 Code segments:

Establishing connection:

```
String url = "jdbc:mysql://localhost:3306/game_list"; // game_list is the
database name

Scanner scanner = new Scanner(System.in);
int attempts = 0;
while (attempts < 3) {
    // providing the user only 3 max attempts to login to database:
    System.out.print("Enter_username:_");
    String username = scanner.nextLine();
    System.out.print("Enter_password:_");
    String password = scanner.nextLine();
    try(Connection conn=DriverManager.getConnection(url, username, password))
    {
        System.out.println();
        System.out.println("Connected_to_database");

        // create a new table called "games" if it doesn't exist
        String createTableSql = "CREATE_TABLE_IF_NOT_EXISTS_games_(id_INT_PRIMARY
        _KEY_AUTO_INCREMENT, _title_VARCHAR(255), _platform_VARCHAR(255),
        _price_FLOAT_NULL)";

        try (Statement stmt = conn.createStatement()) {
            stmt.executeUpdate(createTableSql);
```

```
    } catch (SQLException e) {  
        System.out.println("Error creating table: " + e.getMessage());  
    }  
}  
}
```

Taking user input and handling empty inputs

```
private static String getUserInput(String message) {  
    Scanner scanner = new Scanner(System.in);  
    String input = "";  
    while (input.trim().isEmpty()) {  
        System.out.print(message);  
        input = scanner.nextLine();  
        if (input.trim().isEmpty()) {  
            System.out.println("Input cannot be empty.  
            Please try again.");  
        }  
    }  
    return input;  
}
```

3.2 Libraries

3.2.1 java.sql

The 'java.sql' library in Java provides a set of classes and interfaces for working with databases and managing data persistence.

Here are some of the main features of the 'java.sql' library:

Main Features

- Connection management: The 'Connection' interface provides methods for establishing and managing connections to a database.
- Statement execution: The 'Statement' interface provides methods for executing SQL statements and retrieving results.
- ResultSet management: The 'ResultSet' interface provides methods for iterating over the results of a query and accessing individual rows and columns.
- Transaction management: The 'Transaction' interface provides methods for managing transactions, including committing and rolling back changes.
- Data types: The 'java.sql' library provides a set of classes for working with various data types, including integers, dates, and times.
- Prepared statements: The 'PreparedStatement' interface provides a mechanism for precompiling SQL statements and efficiently executing them multiple times with different parameter values.
- Metadata retrieval: The 'DatabaseMetaData' interface provides methods for retrieving information about the structure and capabilities of a database.

Overall, the java.sql library provides a comprehensive set of tools for working with databases in Java, making it a crucial component of many enterprise-level applications.

3.2.2 java.util

The 'java.util' library in Java provides a set of classes and interfaces for working with collections, date and time, and other utility functions.

Here are some of the main features of 'java.util' library:

Main Features

1. Collection framework: The `java.util` library provides a set of classes and interfaces for working with collections, including `ArrayList`, `LinkedList`, `HashSet`, and `TreeMap`.

2. Date and time management: The `java.util` library provides a set of classes for working with dates and times, including `Date`, `Calendar`, and `SimpleDateFormat`.
3. Random number generation: The `java.util` library provides a `Random` class for generating random numbers.
4. Iterator and Enumeration: The `java.util` library provides a set of classes for iterating over collections, including `Iterator` and `Enumeration`.
5. Comparator and Comparable: The `java.util` library provides interfaces for sorting collections, including `Comparator` and `Comparable`.
6. Properties: The `java.util` library provides a `Properties` class for working with properties files.
7. Scanner: The `Scanner` class in Java is a versatile class that can be used for parsing text input. It can read input from various sources like console, files, and other input streams. It provides many methods to read different types of data such as integers, floats, doubles, strings, and more.

Here are some of the main features of the `Scanner` class:

- Reading input from different sources: The `Scanner` class can read input from various sources, including console, files, and other input streams.
- Parsing different types of data: The `Scanner` class provides methods for reading different types of data, such as integers, floats, doubles, strings, and more.
- Delimiter customization: The `Scanner` class allows you to customize the delimiter used to separate tokens in the input. The default delimiter is whitespace.
- Error handling: The `Scanner` class throws exceptions when it encounters errors while parsing input. These exceptions can be caught and handled in your code.
- Localization support: The `Scanner` class supports localization, which means it can read input in different languages and parse numbers in different formats.
- Convenient API: The `Scanner` class provides a convenient API that makes it easy to parse the input and perform different operations on it.

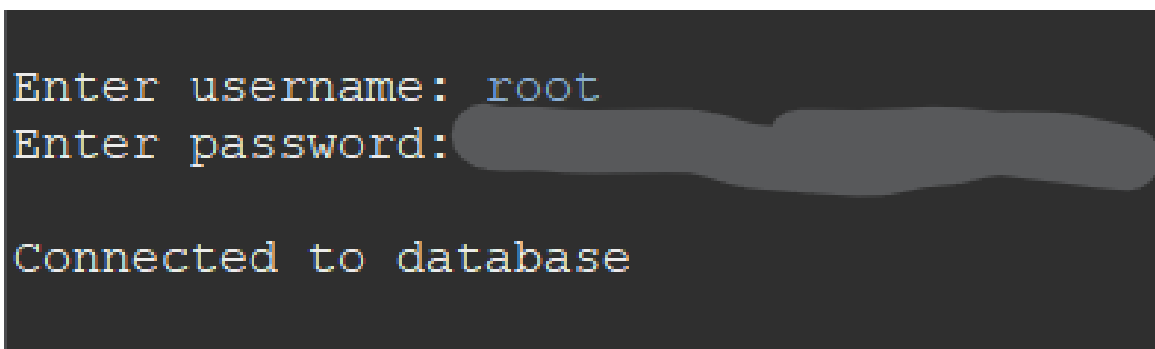
Overall, 'java.util' library provides a rich set of utility classes and interfaces for working with collections, date and time, and other utility functions in Java. It is widely used in many types of Java applications.

3.3 Algorithms

1. Set up the JDBC URL for the database connection.
2. Prompt the user for their username and password and attempt to establish a connection to the database.
3. Create a table called "games" if it doesn't already exist.
4. Present the user with a menu of options:
 - (a) Read/display data from the table.
 - (b) Update data in the table.
 - (c) Delete data from the table.
 - (d) Add new data to the table.
 - (e) Exit the program.
5. Depending on the user's choice, perform one of the following actions:
 - (a) Read data from the table and display it to the user.
 - (b) Prompt the user for the title, platform, and price of a game, and update that game's data in the table.
 - (c) Prompt the user for the title of a game and delete that game's data from the table.
 - (d) Prompt the user for the title, platform, and price of a new game, and add that game's data to the table.
 - (e) Exit the program.
6. Handle any exceptions that occur during the program's execution.

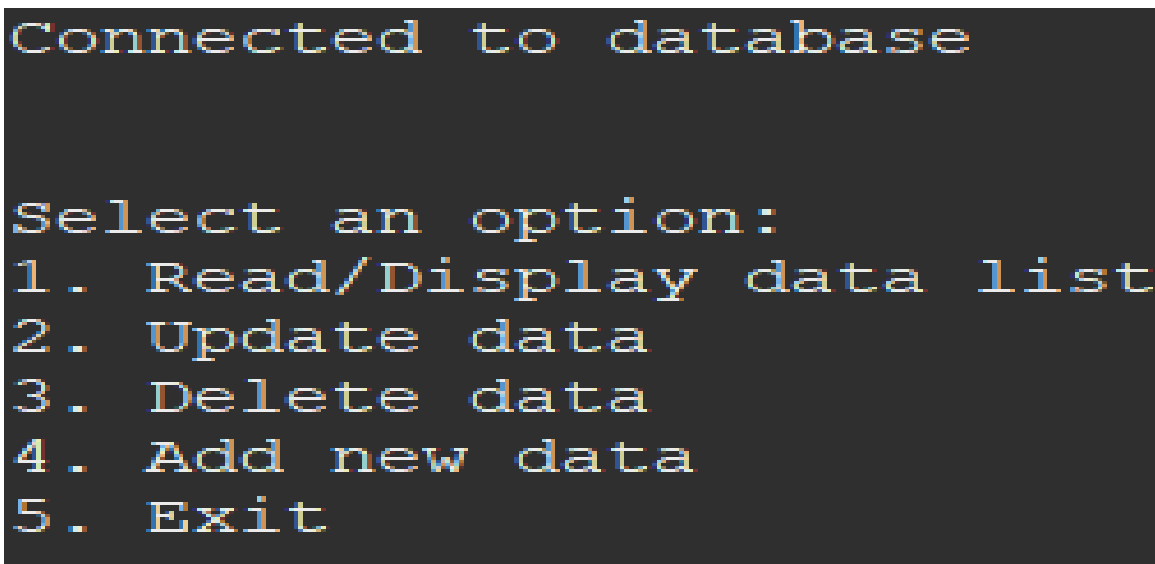
Chapter 4

Observations & Results

A terminal window with a dark background. The text 'Enter username: root' is displayed in a light blue monospace font. Below it, 'Enter password:' is followed by a greyed-out area representing a masked password. At the bottom, the text 'Connected to database' is shown in a light blue monospace font.

```
Enter username: root
Enter password: 
Connected to database
```

Figure 4.1: Establishing connection with the database

A terminal window with a dark background. The text 'Connected to database' is at the top. Below it, 'Select an option:' is followed by a numbered list of five options: '1. Read/Display data list', '2. Update data', '3. Delete data', '4. Add new data', and '5. Exit'. All text is in a light blue monospace font.

```
Connected to database

Select an option:
1. Read/Display data list
2. Update data
3. Delete data
4. Add new data
5. Exit
```

Figure 4.2: Main menu of operations available

```
Enter game title:  
Input cannot be empty. Please try again.  
Enter game title:cs go  
Enter platform:pc  
Enter new price400  
Data inserted successfully.
```

(a) Empty input

```
Enter game title:fallguy  
Enter platform:pc  
Enter new pricehi  
This is not a price value,Not adding this to database
```

(b) Improper price value

```
Enter game title:cs go  
This game data already exists
```

(c) Repeated title

```
Enter username: root  
Enter password: hi  
Error connecting to database:  
Enter correct username and password ,  
Access denied for user 'root'@'localhost'
```

(d) Improper login credentials

```
Retrieving the game list from MYSQL database....  
Nothing found in database.
```

(e) No data case

```
Enter game title to update:cs  
Enter new platform:pc  
Enter new price400  
No rows updated.
```

(f) Improper updation

```
Enter game title to delete:cs  
No rows deleted.
```

(g) Improper deletion

Figure 4.3: Exceptions handled

Chapter 5

Conclusion & Future Enhancements

5.1 Conclusion

The project is a simple and user-friendly application for managing game data. It provides efficient and error-free data management through a secure login system and a user-friendly interface. It is compatible with the MySQL database and has been implemented using a structured approach with proper exception handling. Overall, the project provides a practical and reliable solution for game data management. It is an excellent tool for game enthusiasts, developers, and researchers who require a simple yet effective application for managing their game data.

5.2 Future Enhancements

I have planned to improve the program in various ways. Firstly, I intend to create a graphical user interface (GUI) which would make it more user-friendly and simple to navigate. Secondly, I aim to add a sorting functionality to the program to enable users to easily organize the games in the database based on criteria such as titles, platforms, or prices. Lastly, the program currently works only with a local database, but I intend to enhance it to function with cloud-based databases. This upgrade will allow easy access to the database from any location and facilitate multiple users to work with the same database concurrently.