**Contents**

**Topic**

1. Introduction …………………………………..……………………………………… 1
   1. Purpose………………………………………………………………………. 2
   2. Task Scope…..………………………………...…………………………….. 3
2. Software and Hardware Requirement…..……………………………………………. 4
3. Screenshots…………………………………………………………………………… 5
4. System Implimentation……………………………………………………………….. 9
   1. Processing Environment……………………………………………………… 11
   2. Software Description…………………………………………………………. 11
5. Testing………………………………………………………………………………... 24
6. Output Screen.………………………………………………………………………… 28

**Introduction**

**Assignment No. 1:**

**JAVA SYNC SERVICE**

Technology Used: **Java Desktop Technology**

Language: **Java, MySql**

Model Type: **Individual**

**Report Format**

Version : 1.0

Description : First Draft

Created By : AKASH SINGH YADAV

Date : 12-May-22 to 13-May-22

1. **Introduction**

## Purpose

This task is to create a **Java** service that syncs data from CSV to MySQL. It consists of two parts.

Part 1: You will be given a CSV file that should keep updating over the time. You have to right a Java program that adds few rows of data in the CSV File.

Part 2: You have to populate the MySQL table with the data present in CSV. Your service should run in every 2 mins and check for any new data in CSV. If any new data is found, add that new data to MySQL. For running the service at scheduled intervals, you can read about Runnable interfaces and ScheduledExecutorService in Java.

## Intended Audience and Reading Suggestions

This document is intended for any developers, project manager or documentation writer that needs to understand the basic system architecture and its specifications for development.

* **Developer**: The developer who wants to read, change, modify or add new requirements into the existing program, must firstly consult this document and update the requirements with appropriate manner so as to not destroy the actual meaning of them and pass the information correctly to the next phases of the development process.
* **Tester**: The tester needs this document to validate that the initial requirements of this web portal actually corresponds to the executable code correctly.

For each one of the reader types to better understand this document, here is a suggestion to read in this document:

* Overall description
* System Features
* External Interface Requirement
* Non Functional Requirement

## Task Scope

This task is to create a **Java** service that syncs data from CSV to MySQL. It consists of two parts.

Part 1: You will be given a CSV file that should keep updating over the time. You have to right a Java program that adds few rows of data in the CSV File.

Part 2: You have to populate the MySQL table with the data present in CSV. Your service should run in every 2 mins and check for any new data in CSV. If any new data is found, add that new data to MySQL. For running the service at scheduled intervals, you can read about Runnable interfaces and ScheduledExecutorService in Java.

1. **Software and Hardware requirements**

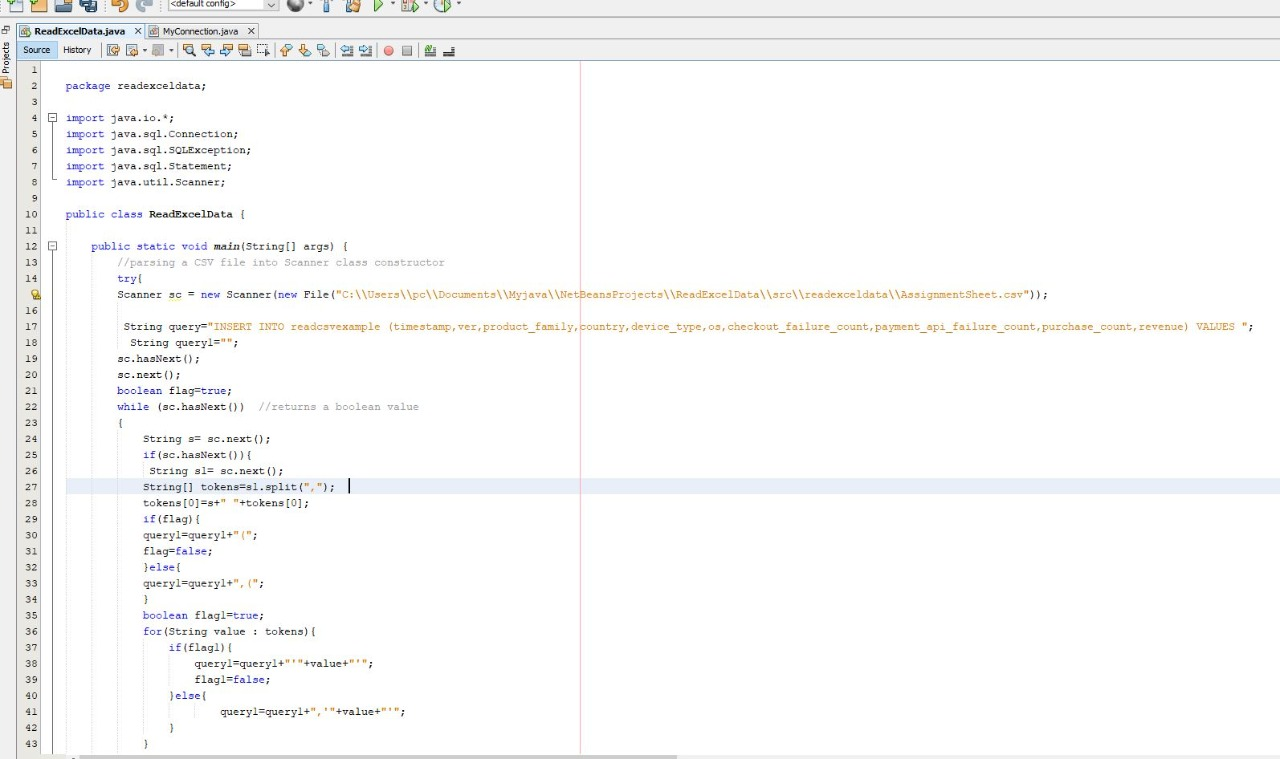
**SOFTWARE REQUIREMENTS SPECIFICATION**

* **Operating System** : Windows-XP,7,8,10,11
* **Programming Language** : Java 2, MySql

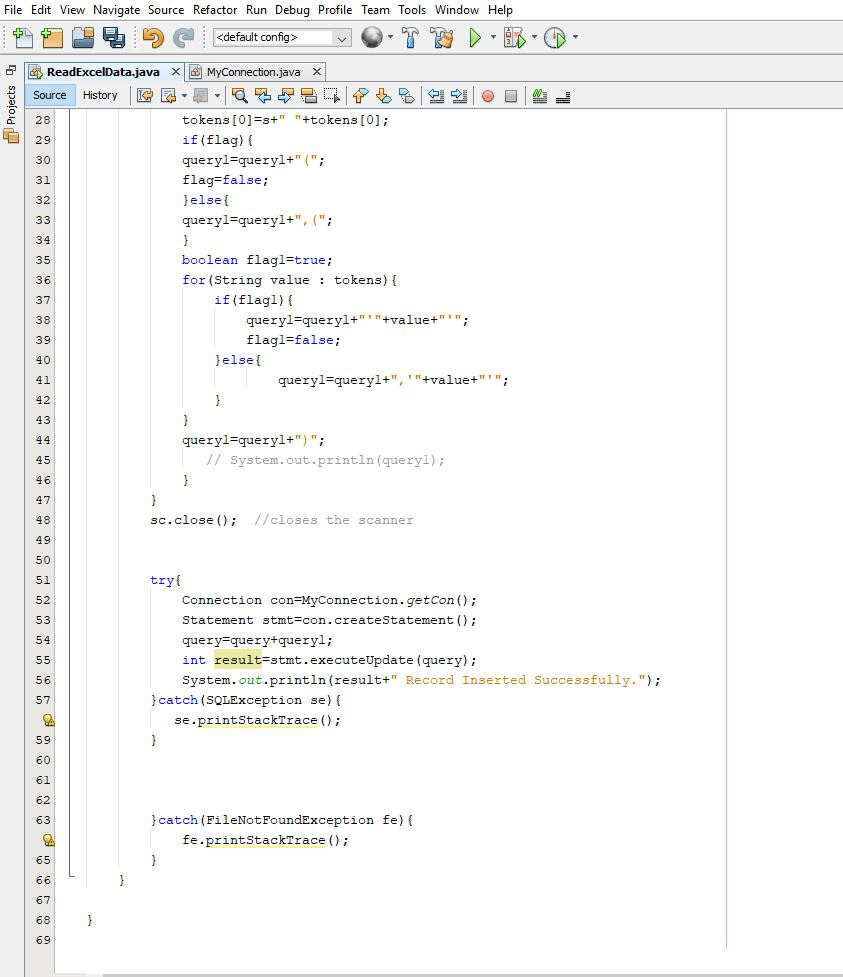
**HARDWARE REQUIREMENTS SPECIFICATION**

* **Processor :** Pentium IV processor or higher
* **RAM**  **:** Minimum of 2GB RAM
* **Memory :** 500 MB or higher

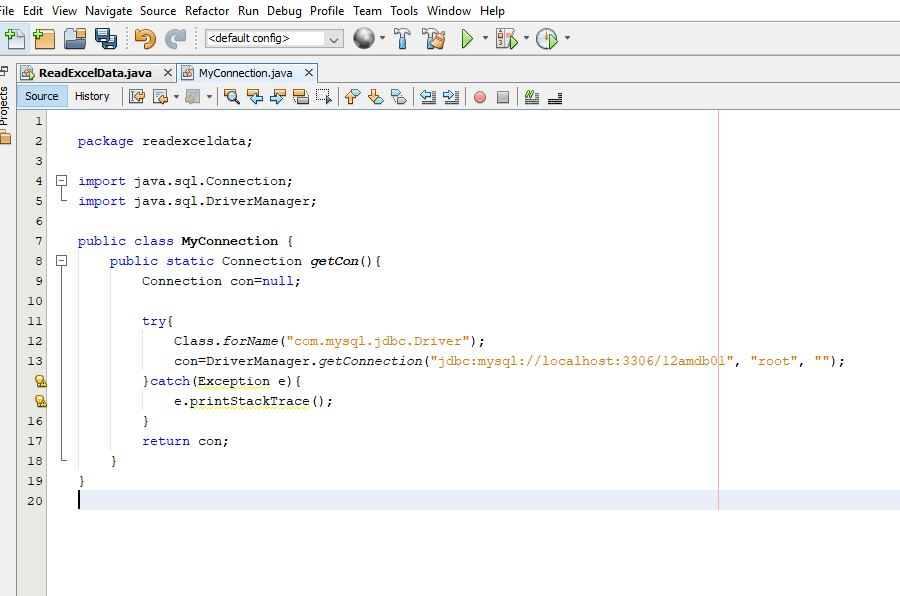
1. **SCREENSHOTS**

**Fig.1**

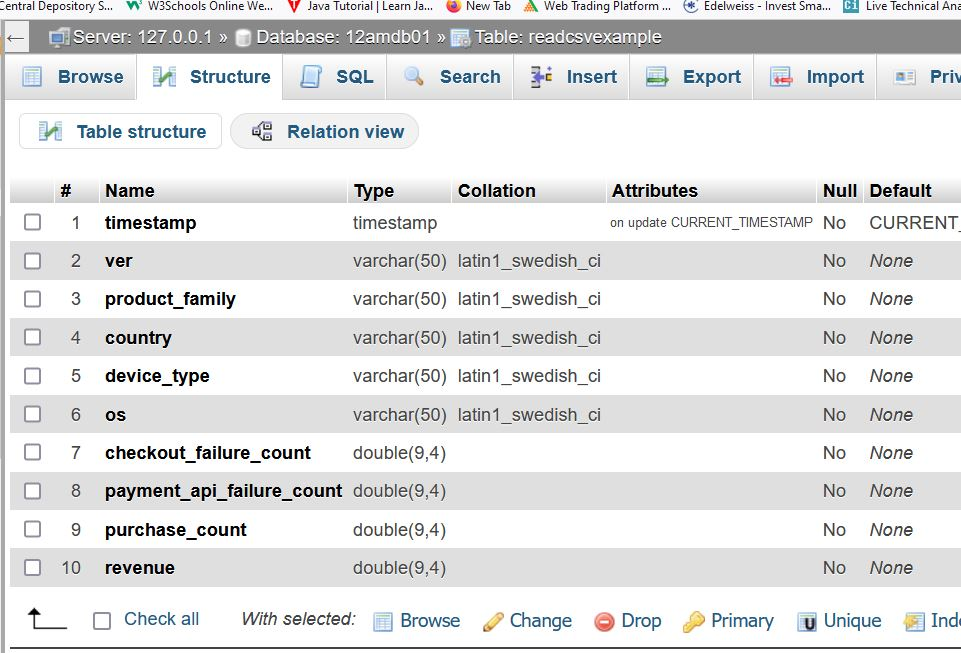
**Fig.2**

****

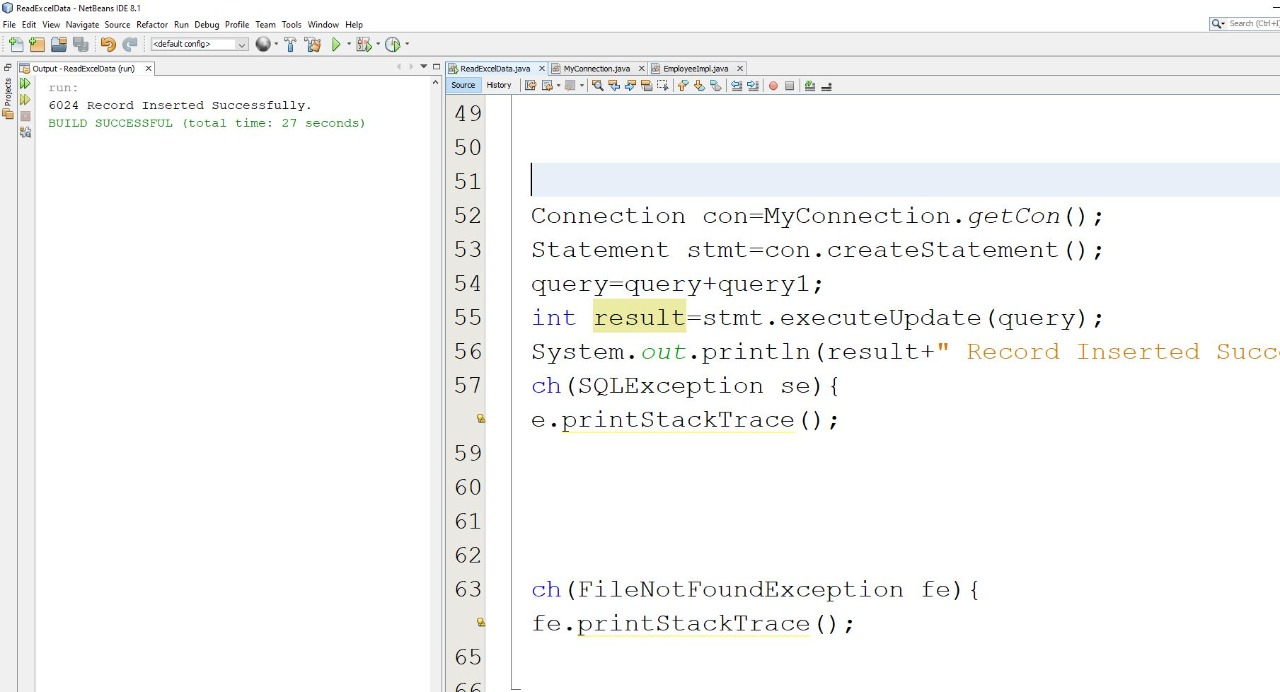
**Fig. 3**

****

**Fig.4**

****

**Fig.5**

****

**4.SYSTEM IMPLEMENTATION**

The term Implementation has different meanings ranging from the conversation of a basic application to a complete replacement of a computer system. The procedures however, are virtually the same. Implementation includes all those activities that take place to convert from old systems to new.

The new system may be totally new replacing an existing manual or automated system or it may be major modification to an existing system. The method of implementation and time scale to be adopted is found out initially. Neat the system is test properly and at the same time the users are trained in the new procedure. Proper implementation is essential to provide a reliable system to meet organization requirement.

Successful and efficient utilization in the system can be achieved only through proper implementation of the system in the organization. So implementation phase is also important like other phases such as analysis, design, coding and testing.

* Careful planning
* Investigation of the system and its constraints
* Design the methods to achieve the change over
* Training the staff in the changed phase
* Ensuring the user has understood and accepted the changes
* Getting complete feedback during test run and ensuring everything in perfect for the final change over.

**4.1 PROCESSING ENVIRONMENT**

**Hardware Specification**

Machine : ACER

Processor : Intel Pentium

Clock speed: 120 MHz Higher

RAM : 4GB

Hard Disk : 1TB

Mother board: Intel mother board

**Software Specification**

Operating System: Windows XP, 7, 8,10, 11

Language: JAVA

Front End: NetBeans 6.9

**4.2 SOFTWARE DESCRIPTIONS**

**JAVA**

Java was developed at Sun Microsystems. Work on Java initially began with the goal of creating a platform- independent language and OS for consumer electronics. The original intend was to use C++, but as work progressed in this direction, developers identified that creating their own language would serve them better. The effort towards consumer electronics led the Java team, then known as First Person Inc., towards developing b/w and s/w for the delivery of video- on- demand with Time Warner.

Unfortunately (or fortunately for us) Time Warner selected Silicon Graphics as the vendor for video-on-demand project. This set back left the First Person team with an interesting piece of s/w (Java) and no market to place it. Eventually, the natural synergies of the Java language and the www were noticed, and Java found a market.

Today Java is both a programming language and an environment for executing programs written in Java Language. Unlike traditional compilers, which convert source code into machine level instructions, the Java compiler translates Java source code into instructions that are interrupted by the runtime Java Virtual Machine. Java is an interpreted language.

Java is the first programming language designed from ground up with network programming in mind. The core API for Java includes classes and interfaces that provide uniform access to a diverse set of network protocols. As the Internet and network programming have evolved, Java has maintained its cadence. New APIs and toolkits have expanded the available options for the Java network programmer.

**Why Java?**

In one of their early papers about the language, Sun described Java as follows: Java : A simple, object oriented, distributed, interpreted, robust, secure, architecture neutral, portable, high-performance, multi threaded and dynamic language.

Sun acknowledges that this is quite a string of buzzwords, but the fact is that, for the most part, they aptly describe the language. In order to understand why Java is so interesting, let’s take a look at the language features behind the buzzwords.

1. OBJECT – ORIENTED

Java is an object-oriented programming language. As a programmer, this means that you focus on the data in your applications and methods that manipulate that data, rather than thinking strictly in terms of procedures. In an object-oriented system, a class is a collection of data and methods that operate on that data. Taken together, the data and methods describe the state and behavior of an object. Classes are arranged in a hierarchy, so that a sub class can inherit behavior from its super class. Java comes with an extensive set of classes, arranged in packages that you can use in your programs. For example, Java provides classes that create graphical user interfacecomponents (the java.awt package), classes that handle input and output (the java.io package), and classes that support networking functionality (the java.net package). Most things in Java are objects; the primitive numeric, character and the Boolean types are the only exceptions. Strings are represented by objects in Java, as are other important language constructs like threads. A class is the basic unit of compilation and of execution in Java; all Java programs are classes. For a complete description of the object-oriented features of Java, The object oriented language used to create executable contents such as applications and applets.

1. INTERPRETED

Java is an interpreted language: the Java compiler generates byte-codes for the Java Virtual Machine (JVM), rather that native machine code. To actually run a Java program, you use the Java Interpreter to execute the compiled byte-codes. Because Java byte-codes are platform-independent, Java programs can run on any platform that the JVM (the interpreter and run-time system) has been ported to. In an interpreted environment, the standard “link” phase of program development pretty much vanishes. If Java has a link phase at all, it is only the process of loading the new classes into the environment, which is an incremental, light weight process that occurs at run-time.

1. ARCHITECTURE NEUTRAL AND PORTABLE

Because Java programs are compiled to an architecture neutral byte-code format, a Java application can run on any system, as long as that system implements the Java Virtual Machine. This is a particularly important for applications distributed over the internet or heterogeneous networks. But the architecture neutral approach is useful beyond the scope of network-based applications. As an application developer in today’s software market, you probably want to develop versions of your application that can run on PCs, Macs and UNIX workstations. The fact that Java is interpreted and defines a standard, architecture neutral, byte-code format is one big part of beingportable. But Java goes even further, by making sure that there are no “implementation-dependent” aspects of the language specification.

1. DYNAMIC AND DISTRIBUTED

Java is a dynamic language. Any Java class can be loaded into a running Java interpreter at any time. These dynamically loaded classes can be the dynamically instantiated. Native code libraries can also be dynamically loaded. Classes in Java are represented by the Class; you can dynamically obtain information about a class at run-time. Java is also called a distributed language. This means, simply, that it provides a lot of high-level support for networking. In Java 1.1, the Remote Method Invocation (RMI) API allows a Java program to invoke methods of remote Java objects, as if they were local objects. The distributed nature of Java really shines when combined with its dynamic class loading capabilities. Imagine. When multi-media word processor written in Java program is asked to display some type of data that it has never encountered before, it might dynamically download a class from the network that can parse the data, and then dynamically download another class that can display the data within a compound document. A program like this uses distributed resources on the network to dynamically grow and adapt to the needs of its user.

1. SIMPLE

Java is a simple language. The java designers were trying to create a language that a programmer could learn quickly, so the number of language constructs has been kept relatively small. In order to keep the language both small and familiar, the Java designers removed a number of features available in C and C++. Java does not use header files and it eliminates the C preprocessor. Java also eliminates the operator overloading and multiple inheritance features of C++. Java does not use pointers. Since Java does not have structures, and arrays and strings are objects, there’s no need for pointers. Java automatically handles the referencing and dereferencing of objects for you. Java also implements automatic garbage collection, so you don’t have to worry about dangling pointers, invalid pointer references, and memory leaks, so you can spend your time developing the functionality of your programs.

1. ROBUST

Java has been designed for writing highly reliable or robust software. Java certainly doesn’t eliminate the need for software quality assurance; it’s still quite possible to write buggy software in Java. Java is strongly typed language, which allows for extensive compile-time check for potential type-mismatch problems. Java requires explicit method declarations. These stringent requirements ensure that the compiler can catch method invocation errors, which leads to more reliable programs. Lack of pointers and pointer arithmetic increases the robustness of Java programs by abolishing an entire class of pointer – related bugs. Similarly, all accesses to arrays and strings are checked at run-time to ensure that they are in bounds, eliminating the possibility of over writing memory and corrupting data. Casts of objects from one type to another are also checked at runtime to ensure that they are legal. Java’s automatic garbage collection prevents memory leaks and another pernicious bug related to memory allocation and de location. Exception handling is another feature in Java that makes for more robust programs. An exception is a signal that some sort of exceptional condition.

1. SECURE

One of the most highly touted aspects of Java is that it’s a secure language. This is especially important because of the distributed nature of Java. Without an assurance of security, you certainly couldn’t want to download code from a random site on the internet and let it run on your computer. Yet this is exactly what people do with Java applets every day. Java was designed with security in mind, and provides several layers of security controls that protect against malicious code, and allow user to comfortably run untrusted programs such as applets. At the lowest level, security goes hand in hand with robustness. As we have already seen, Java programs cannot forgepointers to memory, or overflow arrays, or read memory outside of the bounds of an array or string. This feature is one of Java’s main defenses against malicious code. By totally disallowing any direct access to memory, an entire huge, messy class of security attacks is ruled out.

The second line of defense against malicious code is the byte-code verification process that the Java interpreter performs on any untrusted code it loads. This verification steps ensure that the code is well-formed that it doesn’t over flow or under flow the stack or contain illegal byte-codes. Another layer of security protection is commonly referred to as the “sandbox model”: untrusted code is placed in a “sandbox”, where it can play safely, without doing any damage to the “real world”, or full Java environment. When an applet, or another untrusted code, is running in the sandbox.

Finally, in Java 1.1, there is another possible solution to the problem of security. By attaching a digital signature to Java code, the origin of that code can be established in a cryptographically to secure and unforgivable way. If u has specified that you trust a person or organization, then code that bears the digital signature of that trusted entity is trusted, even when loaded over the network, and may be run without the restriction of the sandbox model.

Of course, security isn’t a black & white thing. Just as a program can never be guaranteed to be 100 % bug-free, no language or environment can be guaranteed 100 % secure. With that said, however, Java does seem to offer a practical level of security for most applications. It anticipates and defends against most of the techniques.

viii) HIGH- PERFOMANCE

Java is an interpreted language, so it is never going to be as fast as a compiled language like C. This speed is more than adequate to run interactive, GUI and network-based applications, where the application is often idle, waiting for the user to do something, or waiting for data from the network. Furthermore, the speed-critical sessions of Java run-time environment, that do things like string concatenation and comparison, are implemented with efficient native code. As a further performance boost, many Java interpreters now include “just in time” compilers that can translate Java Byte- codes into machine codes for a particular CPU at runtime. The Java byte- code format was designed with these “just in time” compilers in mind, so the process of generating machine code is fairly efficient and it produces reasonably good code. If you are willing to sacrifice code probability to gain speed, you can also write portions of your program in C or C++ and use Java native methods to interface with this native code. When you are considering performance, its important to remember where Java falls in the spectrum of available programming languages. Java falls in the middle of the spectrum. The performance of Java’s interpreted byte-codes is much better than the high-level scripting languages (even Perl), but it still offers the simplicity and probability of those languages.

1. MULTITHREADED

Java is a multi threaded language; it provides support for multiple threads of execution (sometimes called light weight processes) that can handle different tasks. An important benefit of multithreading is that it improves the interactive performance of graphical applications for the user. Java makes programming with threads much easier, by providing built-in-language support threads. The Java language syntax also supports threads directly with the synchronized keyword. This keyword makes it extremely easy to mark sections of code or entire methods that should only be run by single thread at a time. Java makes threads so easy to use; the Java class libraries require their use in a number of places. For example, any applet that performs animation does so with a thread. Similarly, Java does not support asynchronous, non-blocking I/O with notification through signals or interrupts—you must instead create a thread that blocks on every I/O channel you are interested in.

1. JAVA RUNTIME ENVIRONMENT

The runtime environment used to execute the code. It is made up of the java language and java virtual machine. It is portable and its platform neutral.

1. JAVA TOOLS

It is used by the developers to create java code. They include java compiler, java interpreter, classes, libraries and applet viewer.

1. JAVA APPLICATION

Compiling the source code into byte code using javac.

Executing byte code program using java interpreter.

1. JAVA APPLETS

Java applets are pieces of java code that are embedded in HTML document using the applet tag. When the browser encounters such code it automatically download it and execute it.

1. JAVA VIRTUAL MACHINE

It is a specification to which java codes must be written. All java code is to be compiled to be used in this nonexistent virtual machine. Writing the code which compiles in JVM ensures platform independence.

1. BUILT-IN NETWORKING

Java was designed with networking in mind and comes with many classes to develop sophisticated Internet communications. Features such as eliminating memory pointers and by checking array limits greatly help to remove program bugs. The garbage collector relieves programmers of the big job of memory management. These and the other features can lead to a big speed up in program development compared to C/C++ programming.

ADVANTAGES OF JAVA

1. JAVA IS ROBUST

Java is relaxed in terms of type checking in terms of programming errors. Java is strict about type declarations and does not allow automatic type casting. Also it uses a pointer model that does not over write memory or corrupt data.

1. JAVA IS PLATFORM INDEPENDENCE

The write-once-Run-Anywhere ideal has not been achieved (tuning for different platforms usually required), but closer than with other languages.

1. COMPILER/INTERPRETER COMBO

Code is compiled to byte codes that are interpreted by Java virtual machines (JVM). This provides portability to any machine for which a virtual machine has been written. The two steps of compilation and interpretation allow for extensive code checking.

1. JAVA IS SECURE

Pointers and memory allocations are removed during compile time.

All byte codes are verified by the interpreter before executing.

All Java applets are treated as untrusted code executing in trusted environment.

Programs run inside the virtual machine sandbox.

Array index limit checking

Java was written to support distributed applications over the computer networks, it can be used with a variety of CPU and operating system architecture-neutral object files from Java code.

1. JAVA IS PORTABLE
2. Java primitive data types and the behavior of arithmetic operations on these data types are explicitly specified.
3. The Java libraries include portable interfaces for each platform on which the runtime environment is available,
4. The entire Java system itself is portable.
5. JAVA IS SMALL

Because java was designed to run on small computers, java system is relatively small for a programming language. It can run efficiently on PCs with 4MB RAM or more. The Java interpreter takes up only a few hundred kilo bytes.

1. JAVA IS GARBAGE COLLECTED

Java programs don’t have to worry about memory management. The java system has a built in program called the garbage collector, which scans the memory and automatically frees the memory chunks that are not in use.

1. JAVA IS DYNAMIC

Fundamentally distributed computer environments must be dynamic. Java is capable of dynamically linking new libraries, methods and instance variables as it goes without breaking.

The linking of data and methods to where they are located is done at run-time.

New classes can be loaded while program is running. Linking is done on the fly. Even if libraries are recompiled, there is no need to recompile code that uses classes in those libraries. This differs from C++, which use static binding. This can result in fragile classes for cases where linked code is changed and memory pointers then point to wrong addresses.

NETBEANS

The NetBeans IDE is a free, open-source Integrated Development Environment for software developers. The IDE runs on many platforms including Windows, Linux, Solaris and the MacOS. The NetBeans IDE provides developers with all the tools they need to create professional cross-platform desktop, enterprise, web and mobile applications.

Using an Integrated Development Environment (IDE) for developing applications saves you time by managing windows, settings, and data. Drag-and-drop features make creating graphical user interface (GUI) components or accessing databases easy, and highlighted code and debugging features alert you to errors in your code.

The NetBeans IDE is open source and is written in the Java programming language. It provides the services common to creating desktop applications such as window and menu management, settings storage and is also the first IDE to fully support JDK 5.0 features. The NetBeans platform and IDE are free for commercial and non commercial use, and they are supported by Sun Microsystems.

1. SWING GUI BUILDER (Formerly Project Matisse)

Design Swing GUIn intuitively by dragging and dropping GUI components from a palette onto a canvas, the click into JLabels, JButtons, JTextFields and edit the text directly in place. You can even use the GUI builder to prototype GUIs right in front of customers. The NetBeans IDE comes with built-in support for GUI localization and accessibility.

1. WEB APPLICATION DEVELOPMENT

Create a complete functional JSF page for manipulating data from a database in a few steps. Supports Java Server Faces (JSF), Persistence Units, Struts, and the Java Server Pages Standard Tag Library (JSTL). Comes with a visual editor for development descriptors and HTTP monitor to debug web applications.

1. ENTERPRISE DEVELOPMENT

Use Blueprints, templates and wizards to quickly create Enterprise Java Beans (EJB 3) and JAX-WS web services. Full support for Java EE 1.3, 1.4 and 1.5 standards.

1. VERSION CONTROL

Version control is tightly integrated into the IDE’s workflow: The IDE recognizes your existing CVS working directories automatically. Use the built-in CVS or get the Subversion module from the Update Center.

1. DEVELOPER COLLABORATION

With NetBeans Developer Collaboration (available from the Update Center), you share whole projects and files in real time over the network. Allow your co-workers to make remote changes and watch them type and run your application. Review your partners work and send instant messages in plain text, XML, HTML, or Java complete with syntax highlighting.

1. ADVANCED SOURCE CODE EDITOR

The language-aware editor indents, completes, and syntax-highlights your source code. It parses your code live, matches words and brackets, marks errors, and displays hints and javadoc. The editor can be fully customized and split vertically or horizontally, and offers well integrated Refactoring, Debugging and JUnit testing.

Enterprise manager is a window management console to create new data base, users, security restrictions triggers, procedures, rules etc.. it gives a GUI based design with security. It allows login for valid users, user can create data bases and users using wizard with good ease. It gives a tree structure like windows explorer and navigation is very easy. It also allows to creating a new roles and logins which can be used for security aspects using enterprise manager we import and export data base from or to their RDBMS.

Query analyzer is another tool with SQL Server, which extends the capabilities of ANSI standard for SQL. If is an application that recognize and execute SQL commands and the specialized T-SQL commands that can be used to create data base objects using SQL command

* We can use query analyzer commands to
* Enter, Edit, Solve, Retrieve, and Run SQL commands
* Format, Perform, Calculation, Store and Print Query resolves
* List column definition for any table

**5.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 test. 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 input produces 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 satisfaction, as shown by successfully 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.

**System Testing**

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.

**Functional Testing**

Functional tests provide a systematic demonstration that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be 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.

There are two basic approaches of functional testing:

a. Black box or functional testing.

b. White box testing or structural testing.

**(a) Black box testing**

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

In **black box testing,** we try various inputs and examine the resulting outputs. The black box testing can also be used for scenarios based test .In this test we verify whether it is taking valid input and producing resultant out to user. It is imaginary box testing that hides internal workings. In our project valid input is image resultant output well structured image should be received.

**White box testing**

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

1. Statement testing

2. Branch testing

**Statement Testing**

The main idea of statement testing coverage is to test every statement in the objects method by executing it at least once. However, realistically, it is impossible to test program on every single input, so you never can be sure that a program will not fail on some input.

**Branch Testing**

The main idea behind branch testing coverage is to perform enough tests to ensure that every branch alternative has been executed at least once under some test. As in statement testing coverage, it is unfeasible to fully test any program of considerable size.

**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.

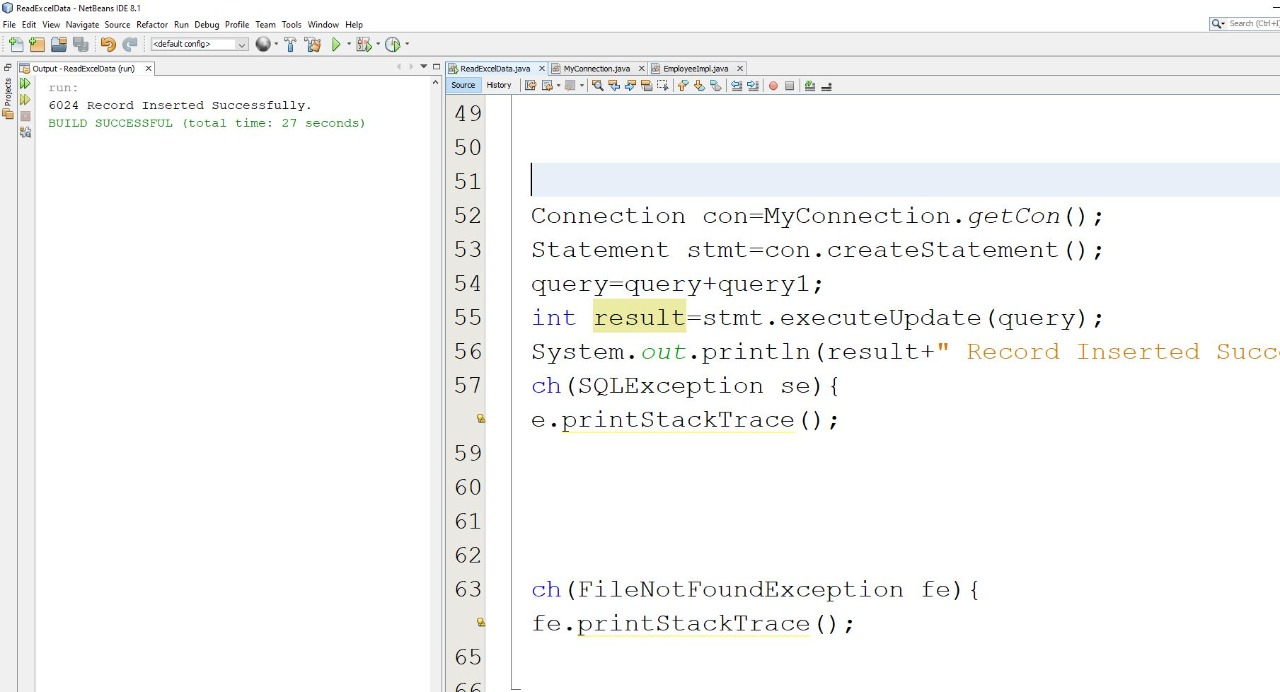
The task of the integration test is to check that components or software applications, ex. components in a software system or one step up software applications at the company level - interact without error.

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

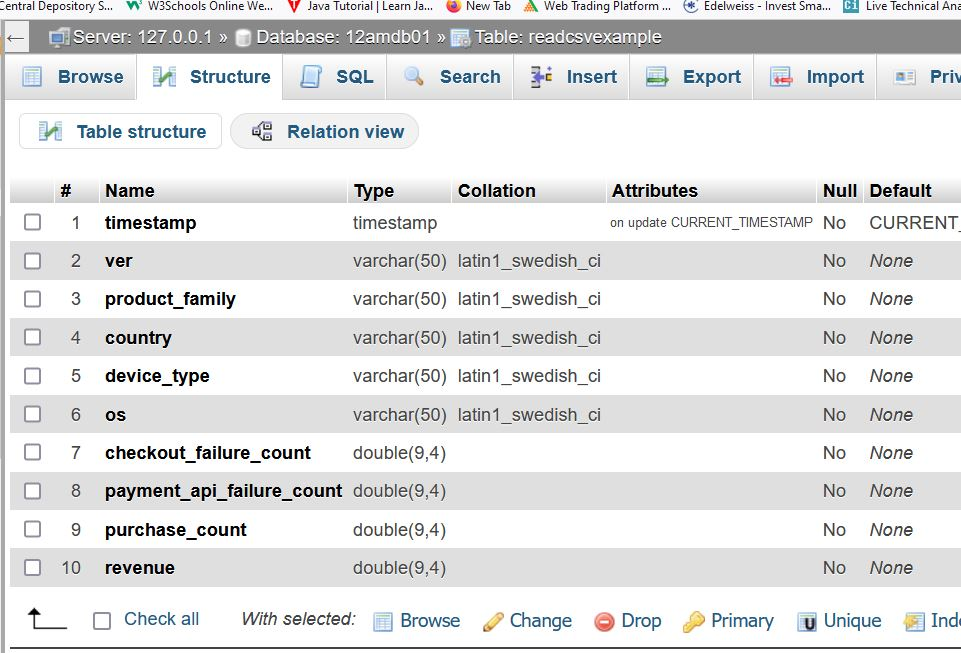
1. **Output Screens**

The following shows the output screens and how the actual process of implementing looks like.

The output is as shown below:-

****

**Database:-**

****