SOURCE CODE METRICS VISUALIZER OF JAVA

By

Farzeen Shahzad

Uzair Zia

Ghulam Abbas

DEGREE

BS Information Technology

Department / Degree of Computing & Information Technology

University of Gujrat

Session 2017-2021

ACKNOWLEDGEMENTS

First and foremost, we would like to thanks Almighty Allah who showers his countless blessing always on us. Then our parents for their love and support throughout. Thanks parents, for giving us strength and opportunity to fulfill our dreams.

We would like to genuinely thank to our supervisor, **Mr. Muhammad Ikram ul Haq** and Faculty of Computing and Information Technology for their guidance and support. His comments and questions are very beneficial for us for completion of this project.

Finally, many thanks to friends, who have helped and given us suggestions, supports and corrections throughout the project.

Farzeen Shahzad	**Uzair Zia**	**Ghulam Abbas**
(17631556-039)	**(17631556-037)**	**(17631556-038)**
_____	_____	_____

DEDICATION

This project is dedicated to our parents and siblings for their endless love, without their support and encouragements we will never be able to achieve our goals.

DECLARATION

We Farzeen Shahzad S/O Muhammad Arif Roll # 17631556-039, Uzair Zia S/O Zia Mumtaz Roll # 17631556-037, Ghulam Abbas S/O Muhammad Shakeel Roll #17631556-038,, student of Bachelor of Information Technology, Department of Computing & Information Technology, University of Gujrat Pakistan, hereby solemnly declare that the data quoted in this thesis titled "Source Code Metrics Visualizer of Java" is based on our original work and has not yet been submitted or published elsewhere.

Farzeen Shahzad _____

Uzair Zia _____

Ghulam Abbas _____

I certify that Farzeen Shahzad S/O Muhammad Arif Roll # 17631556-039, Uzair Zia S/O Zia Mumtaz Roll # 17631556-037, Ghulam Abbas S/O Muhammad Shakeel Roll # 17631556-038, student of Bachelor of Information Technology, Department of Computing & Information Technology, University of Gujrat Pakistan, worked under my supervision and the above stated declaration is true to the best of my knowledge.

Supervisor: Mr. Muhammad Ikram Ul Haq _____

Department of Information Technology

University of Gujrat, Punjab, Pakistan.

Email: ikramulhaq032@gmail.com

Dated: _____

PROJECT COMPLETION CERTIFICATE

It is verified that this thesis titled “Source Code Metrics Visualizer of Java” by Farzeen Shahzad S/O Muhammad Arif Roll # 17631556-039, Uzair Zia S/O Zia Mumtaz Roll # 17631556-037, Ghulam Abbas S/O Muhammad Shakeel Roll # 17631556-038, student of Bachelor of Information Technology, Department of Computing & Information Technology, University of Gujrat Pakistan, contains sufficient material required for the award of above said degree.

Mr. Muhammad Ikram Ul Haq (Supervisor)
Department of Information Technology
University of Gujrat, Punjab, Pakistan.
Email: ikramulhaq032@gmail.com _____

Mr. Muhammad Ikram Ul Haq
Head of IT Department,
Govt. Postgraduate College Jhelum, Punjab, Pakistan.
Email: ikramulhaq032@gmail.com _____
Dated: _____

Contents

List of Figures:

List of Tables:

LIST OF APENDICES

Appendix I. …. Chapter 1

i. CyVis is abbreviated as software complexity visualizer; it is free software metrics collection, analysis and visualization tool for java-based software.

ii. SCMV is abbreviated as Source Code Metrics Visualizer.

Appendix II. …. Chapter 2

i. UC is abbreviated as use cases that are used to describe the functionality of application.

Appendix III. …. Chapter 3

i. UD is abbreviated as use case diagram that is made with the help of use cases.

ii. CD is abbreviated as collaboration diagram that is made with the help of sequences diagram.

iii. SD is abbreviated as sequence diagram that contains the sequences of steps according to system functionality.

CHAPTER 1

INTRODUCTION

CHAPTER-1

INTRODUCTION

1.1 Introduction

This is an era of technology all over in the world a huge number of software are developed and created. Thousands and millions of developers and coders working hard to make efficient and reliable software. In this challenging environment, the size and the number of software increase enormously day by day.

A crucial issue which every developer had to face is maintenance and quality issue regarding their software. Software development projects are difficult to manage, due to the friction between completing system feature and at the same time, obtaining a high degree of code quality to ensure the maintainability of the system in the future. A major challenge of this optimization problem is that code quality is less visible to stakeholders in the development process, particularly to the management [1]

Computer programs are written in different programming languages like Java, Python, C, C++, C# etc. Every programming language has its own rules in which source code is written is known as syntax. The syntax provides the description how developers can write set of source code in different programming languages [2].

Source code is human readable instructions that is written by programmers while writing a program. Source code run through compiler to convert source code into machine code. Machine code is not human readable [3].

Following the statement “If you cannot measure it, you cannot manage it” [4].

Overtime it has been observed that trying to maintain a software system that is not well documented poses serious challenges. A huge amount of time and effort is expended on iteratively reviewing a large body of code to extract the structural information such as data members and methods. This research describes "Source Code Metrics Visualizer of Java (SCMV)". This prototype considers JAVA source code and provides users with many source code metrics information.

Source Code visualization refers to the ability to scan an arbitrary set of source code and map metrics or static structure in graphical terms. The Source code Metrics Visualizer of java offers many capabilities to visualize an application code metrics. According to this, code flaws and code patterns become more apparent. Such perspectives aim to enhance

your understanding of the code base, to let you take better decision to increase code quality and code maintainability.

Varieties of JAVA source files are accepted by SCMV, which analyze them to produce object-oriented information. It begins by analyzing every single file to extract class information like attributes, methods and information like data types.

Additionally, SCMV provides several types of object-oriented software metrics for each class like number of child and parent class. It further calculates metrics like number of the executable lines, declarations and comments. SCMV (e.g. See Appendix I-ii) can count coupled class name and the name of class from which the class coupled and no of time each child class coupled with its parent class in a package. It describes the complexity of source code or programs. In SCMV Complexity can be defined according to the approach that is no of conditional statements, no of loop statements, no of cases and switch statements in each program to calculate. In order to reduce user's cerebral load, the output of the analysis is visualized in 2 dimensional and 3 dimensional graphs, charts and also generate, print, view and save a report of that information.

Coupling is the indication of relationship between modules. It shows the relative independence among modules. It is a degree to which module is connected with others. The minimum degree of coupling is obtained by making modules as independent as possible. [5]

Cohesion is the indication of relationship within modules. It shows the module's relative functional strength. Cohesion is a degree (quality) to which a module focuses on the single thing [6].

Visualization is useful for displaying the information of classes for developer without having to read the whole source code that is difficult task for him [7].

Software metrics and their visualization are two important features of measurement systems. Companies have developed measurement systems to guide them to monitor and control the status and progress of their products.

Software visualization are useful for analyzing multiple aspects of complex software systems. Software visualization tools have been proposed to help analysts to make sense of multivariate data [8].

1.2 Literature Review

Recently, it is difficult to understand and control the development of source code because the length of source code is increasing day by day. So, generally metrics are used to reduce the complexity and maintain the quality of code [9].

Due to sheer length of source code it is difficult to analyze and understand the structure of source code. IEEE defines metric as 'a quantitative measure of the degree to which a system, component, or process possesses a given attribute.' The goal of software metrics is to identify and control essential parameters that affect software development. when metrics are applied in a consistent manner, it helps in project planning and project management activity. For example, schedule-based resource allocation can be effectively enhanced with the help of metrics [10].

Visualization is a process of presenting the information with a purpose of providing the user with a qualitative understanding of that information. It is also the process of transforming objects, concepts, and numbers into a form that is visible to the human eyes [11].

Visualization is useful in a way to identify the complex information due to fact visual presentation that allows human to look at complicated aspects of code. When the actual software code is visualized in both static or dynamic analysis this is called source code visualization [12].

Data visualization is the process of presenting the data and information in graphical form. It is also the process of understanding relationship ratios between numbers [11].

Software visualization systems can be used in teaching to help students understand how algorithms work, and they can be used in program development as a way to help programmers understand their code better. Computer graphics and animation are used in software visualization to present how computer programs, processes, and algorithm works [13].

Cyclomatic Complexity Visualizer (CyVis) is a java-based software tool that is used for metrics collection, analyze and visualization.

It collects raw data from java class or java files and certain metrics like number of methods, lines, classes, statements, and packages are obtained. When the metrics are collected then they are visualized using visualization techniques [14].

1.2.1 Data Visualization

Data visualization is the process of presenting the data and information in graphical form.

It is also the process of understanding relationship ratios between numbers.

When visualizing we have the following attributes and entities.

Entity: point, line (curve), polyline, surface, solid, image, text.

Attribute: color/intensity, location, style, size, relative position.

“Data”, have some special characters, and often can be divided into following groups:

Numeric, symbolic (or mix): 123, or @

Scalar, vector, or complex structure:

Various units: meters, inch.

Discrete or continuous: 1, 2, 3, or p

Spatial, quantity, category, temporal, relational, structural

Accurate or approximate

Dense or space

Ordered or non-ordered

Disjoint or overlapping

Binary, enumerated, multilevel

Independent or dependent

Multidimensional, etc.

We assume that data is properly visualized if it has the following characteristics:

Effective: viewer can easily understand.

Accurate: Data should be correct for fruitful evaluation.

Efficient: Reduce redundancy and show data.

Some of the data visualization techniques are as follows:

Charts: bar or pie

Graphs: good for structure, relationships

Plots: 1- to n-dimensional

Maps: one of most effective

Images: use color/intensity instead of distance (surfaces) [11].

1.2.2 Software Visualization

Refers to the visualization of information of and related to software systems either the architecture of its source code or metrics of their runtime behavior and their development process by means of static, interactive or animated 2-D or 3-D visual representations of their structure, execution, behavior, and evolution.

The field of software visualization (SV) investigates approaches and techniques for static and dynamic graphical representations of algorithms, programs (code), and processed data. SV is concerned primarily with the analysis of programs and their development. The goal is to improve our understanding of inherently invisible and intangible software, particularly when dealing with large information spaces that characterize domains like software maintenance, reverse engineering, and collaborative development. The main challenge is to find effective mappings from different software aspects to graphical representations using visual metaphors. This paper provides an overview of the SV research, describes current research directions, and includes an extensive list of recommended readings.

1.2.3 Cyclomatic Complexity Visualizer

CyVis (e.g. See Appendix I-i) is a free software metrics collection, analysis and visualization tool for Java based software. CyVis collects data from java class or jar files. Once the raw data is collected, certain metrics like number of lines, statements, methods, classes and packages are obtained. Other metrics like cyclomatic complexity etc. Once the metrics are collected, the statistical information can be viewed as charts, graphs and tables. Lots of importance has been given to how the information is shown on the charts. Though CyVis is written in java it has only been tested for windows machines, so we do know how it might behave on other platforms [14].

1.3 Visualization Techniques

To make the prodigious data and information convenient and perspicuous we used the procedure of visualization. Visualization is extremely effective method to understand the software and its entities.

There are many techniques which used to represent the visualization results in a more productive way.

We use 2 dimensional and 3 dimensional graphs, charts to represent output of source code.

1.3.1 2D and 3D Visualization

The world around us is full of shapes. While some shapes exist only on flat surfaces, others exist everywhere else. These shapes are classified as either 2D or 3D.

2D refers to the term “two-dimensional.”

3D refers to the term “three-dimensional.”

A 2D shape is a figure that has only length and height as its dimensions. Because 2D shapes lie on a flat surface, they are also known as plane figures or plane shapes. While they have areas, 2D shapes have no volume.

Apart from length and height, a 3D shape also has width or depth as its third dimension [15].

There are many conventional data visualization techniques some are as follows:

1.3.2 3D Pie Chart

A pie chart is also called circle graph. The pie chart is used to control the size of data wedge as compare to other data wedges. In pie chart a wedge represents the part of data that has common features and characteristics.

Pie Charts are circular charts, each of which sectors represents a proportion of the whole. Note that there can be no zero or negative values and pies are not reliable for displaying a big array of data- it’s recommended to have no more than seven points in a pie.

Pie chart are useful for cross sectional visualizations, or for viewing a snapshot of categories at a single point in time. Pie chart divide categories into slices to illustrate numerical proportions of a whole [16].

A pie chart displays data, information, and statistics in an easy-to-read ‘pie-slice’ format with varying slice sizes. The main use of a pie chart is to show comparison. When items are presented on a pie chart, you can easily see which item is the most popular and which is the least popular. The pie chart is the perfect choice to illustrate data in a percentage format [17].

The given pie chart shows the involvement of people among various social media apps Facebook has got more attraction from people while google plus reached on second rank among public. Only 17% masses used twitter and LinkedIn utilized by almost 15% multitudes. However, Tumbir and Pinterest magnetized only 8.5% and 10.6% people respectively.

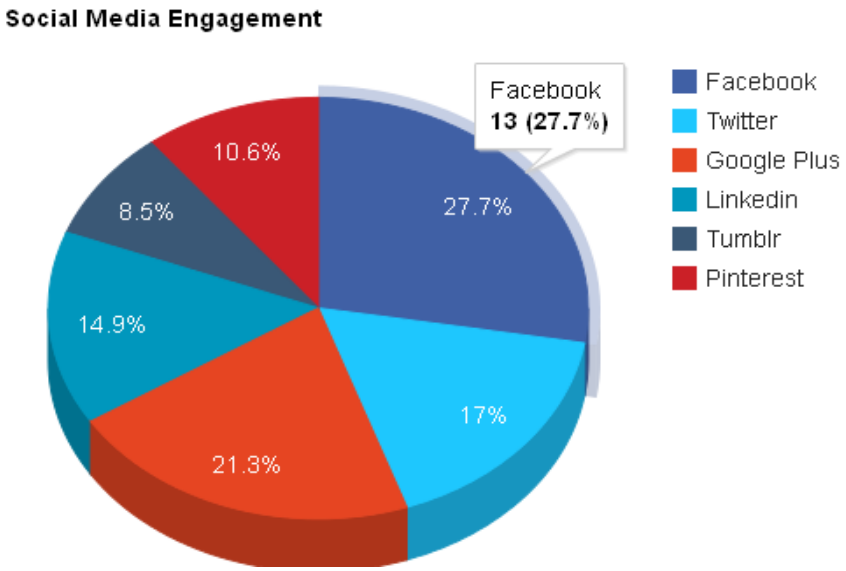


Figure 1.1: 3D Pie Chart

1.3.3 3D Line Chart

Line Charts are used to visually compare values to each other. Data for a 3D line chart are entered in columns. Each numeric data value becomes a point. The simple 3D line chart procedure gives a 3D line chart for each column of data. The two-factor 3D line chart combines columns of data into a single chart [18].

A 3D line chart emphasizes the amount of change over a period of time or compares multiple items. Data points are plotted in series using evenly-spaced intervals and connected with a line to emphasize the relationships between the points. The 3D style is used to add visual emphasis to the chart. It does not provide an extra dimension for plotting data. However, the 3D line chart can help eliminate visual confusion where data sets are similar [19].

Line graph are commonly used visualization technique that uses horizontal (X) and vertical (Y) axes to map quantitative, independent or dependent variables [20].

Here the 3D line graph connects each data point together to determine the monthly sales summary from one point to the next during two consecutive years 2008 and 2009 on Y axes. On X axes the graph often displays time series relationship in month by tracking

changes in continuous data, sing equal interval of time between each data point. Red color shows the data of 2008 while blue shade presents 2009.

The graph bellow shows that in both years there accord bit variation in monthly wages. However, in July and October there was a drastic fall in monthly wages but during such two-month wages lied between 4500 to 4700. In November it raised slightly and then in December increment in salaries although slow but reached on its peak up to the end of December 2008 and cross the salaries level of 4800. In the next year (2009) at the beginning of March level of salaries equalized to the previous year. In this trend a noticeable decreased appeared during various months (march-April, May-June, August to September and October-November). At the beginning of May and October the wages level crossed the increment record and in October wages reached on its highest level (4600).at the end of 2009 wages level reached to 4400.

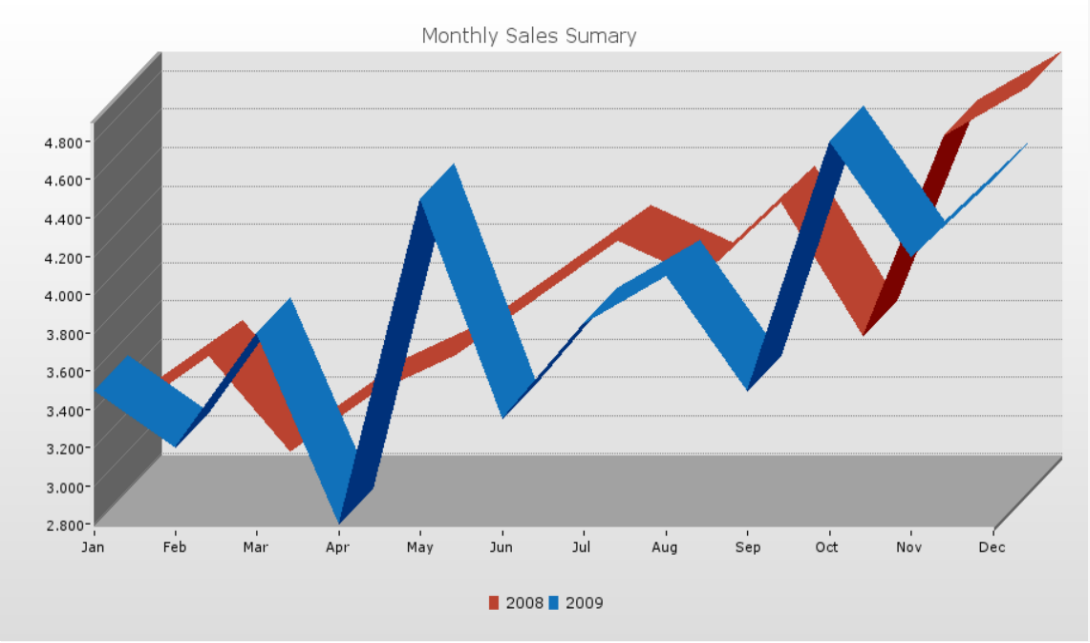


Figure 1.2: 3D Line Chart

1.3.4 3D Stacked Bar Chart

A stacked bar graph (or stacked bar chart) is a chart that uses bars to show comparisons between categories of data, but with ability to break down and compare parts of a whole. Each bar in the chart represents a whole, and segments in the bar represent different parts or categories of that whole.

Stacked bars do a good job of featuring the total and also providing a hint as to how the total for each category value is divided into parts. Stacked Bar graph can have one category

axis and up to two numerical axes. Category axis describes the types of categories being compared, and the numerical axes represent the values of the data.

Stacked Bar graph can be used to represent: Ranking, Nominal Comparisons, Part-to-whole, Deviation, or Distribution [21].

The bellow 3D Stacked Bar Chart is an illustration of ice cream, chocolate and coke sale in four different seasons. In winter demand of such products was lowest (30000) while in summer their consumption reached about 60000. While in spring its sale was 20000 low than summer. In autumn their sale is almost 48000 only.

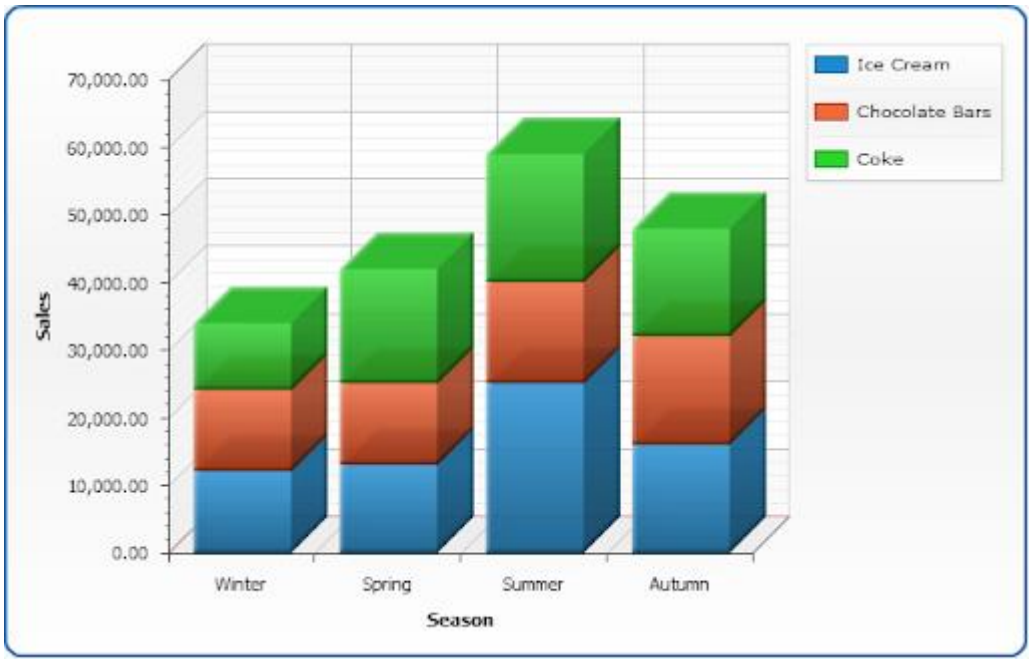


Figure 1-3: 3D Stacked Bar Chart

1.3.5 2D Stacked Area Chart

The stacked area chart typically uses three data dimensions, or variables, to compare trends or changes over time for two or more categories. Categories are non-quantitative data elements such as customers, vendors, departments, branches, merchant category codes, transaction types, statuses, and so on. A single, quantitative category can also be plotted using a stacked area chart.

In addition to comparing changes over time, the stacked area chart also shows the proportion of the total that each category represents at any given point in time. Each

category is represented by an area, differentiated by color. The areas are stacked, and for each increment on the horizontal axis they vary in width in accordance with the percentage of the vertical-axis total they represent. The top edge of each stacked area represents the cumulative total of that category and all the categories beneath it [22].

The bellow 2D Stacked Area Chart shows the targets of two products that a company want to achieve. Months are shown on X-axis and Revenues shown on Y-axis. From product A company failed to achieve its target. However, product B in February and April reached on the target line and from June this product touched the target line and then move upward with the rest of given years.

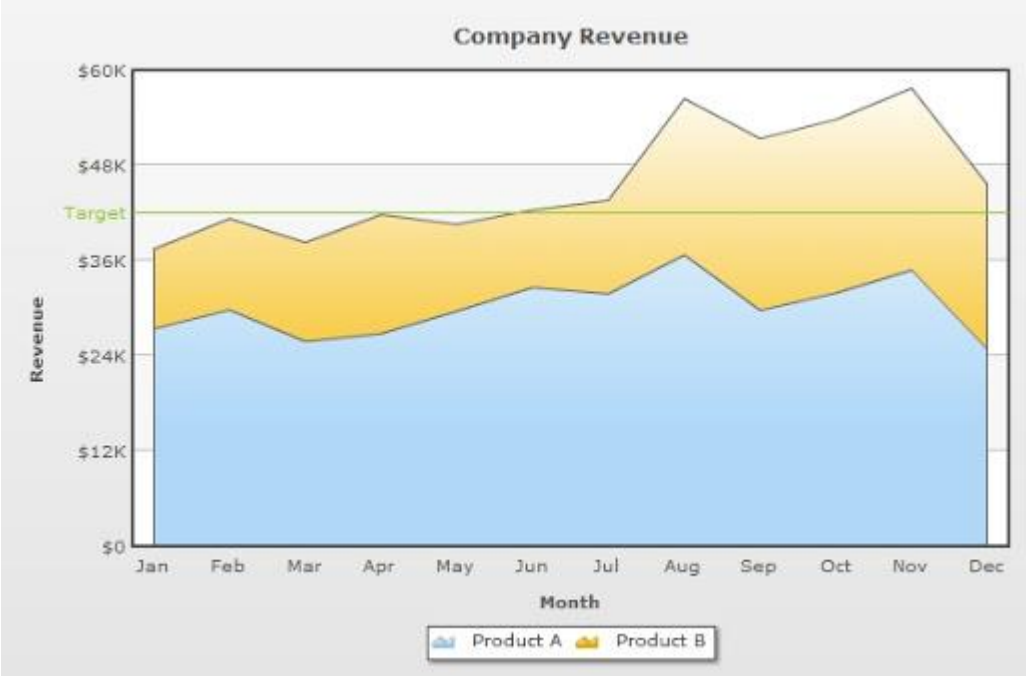


Figure 1-4: 2D Stacked Area Chart

1.3.6 2D Dual Axis Chart

A dual axis chart or multiple axes chart uses two axes to easily illustrate the relationships between two variables with different magnitudes and scales of measurement.

The relationship between two variables is referred to as correlation. Dual axis chart combines a column chart with a line chart. Dual axis chart compares two-line charts. There can be more than two lines if need [23].

The graph bellow depicts production and consumption of Canada natural gas during a decade (2003-12). Years are illustrated on X-axis and revenues are given on Y-axes. From the beginning of the decade the production was bit above than 6000 billion cubic feet than

from 2007 to onward its production decreased slowly and up to 2012 its production reached on 5000 billion cubic feet. However, consumption remained below throughout the whole decade and stayed near and equal to 3000 billion cubic feet.

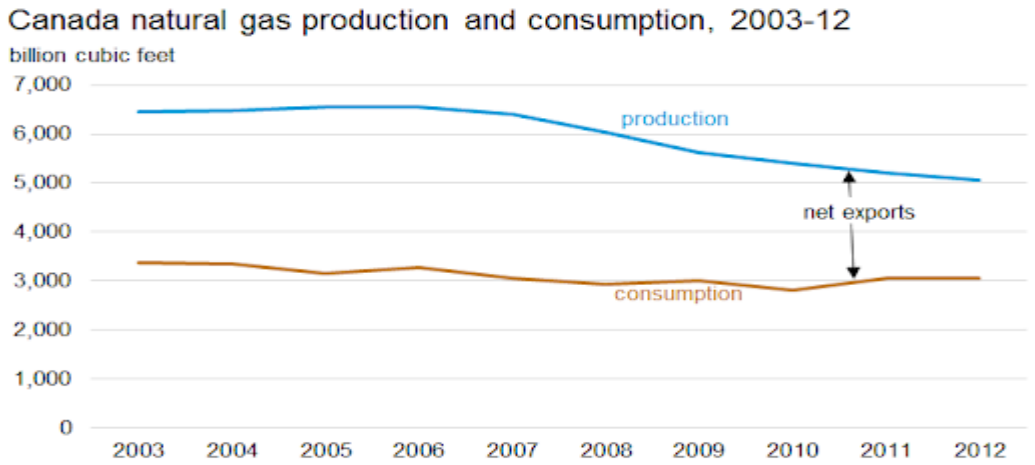


Figure 1-5: 2D Dual Axis Chart

1.3.7 Waterfall Chart

A waterfall chart is a form of data visualization that helps in understanding the cumulative effect of sequentially introduced positive or negative values. These intermediate values can either be time based or category based. The waterfall chart is also known as a flying bricks chart or Mario chart due to the apparent suspension of columns (bricks) in mid-air. Often in finance, it will be referred to as a bridge.

Waterfall charts were popularized by the strategic consulting firm McKinsey & Company in its presentations to clients.

Complexity can be added to waterfall charts with multiple total columns and values that cross the axis. Increments and decrements that are sufficiently extreme can cause the cumulative total to fall above and below the axis at various points. Intermediate subtotals, depicted with whole columns, can be added to the graph between floating columns.

The waterfall, also known as a bridge or cascade, chart is used to portray how an initial value is affected by a series of intermediate positive or negative values. Usually the initial and the final values (end points) are represented by whole columns, while the intermediate values are shown as floating columns that begin based on the value of the previous column.

The columns can be color-coded for distinguishing between positive and negative values [24].

The bellow waterfall chart is an illustration of demand for meat and potato in four different directions. Directions are given on X-axis and demand shown on Y-axis. The demand for these products is 40,30,20,10 in north, east, south and west respectively.

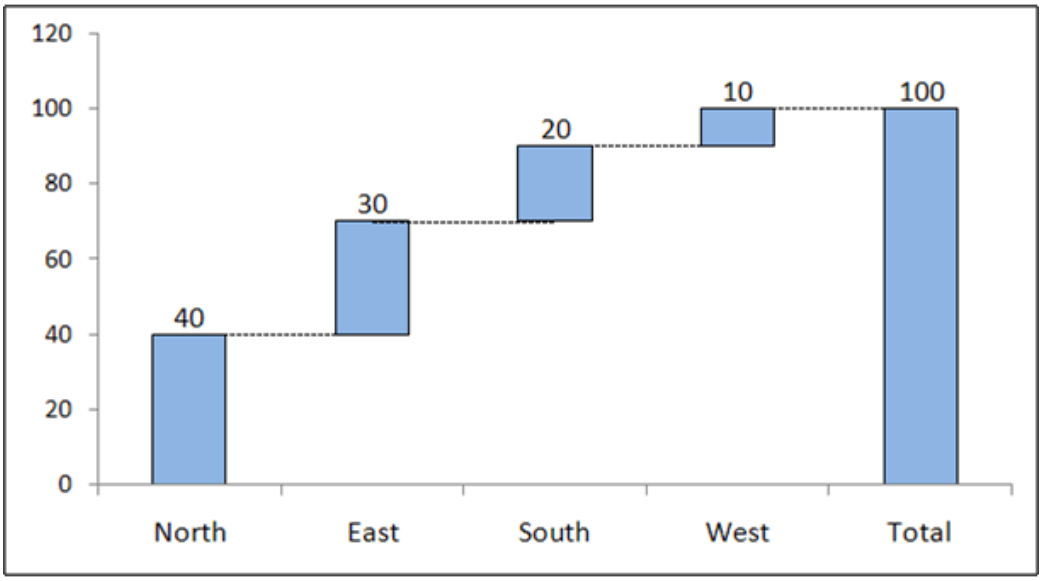


Figure 1-6: Waterfall Chart

1.4 Problem in Existing System

CyVis (e.g. See Appendix I-i) cannot count data types, commented lines, physical lines, logical lines, inheritance of class, blank lines, and also not provide 3dimensional visualization techniques. It cannot facilitate the user to generate report, print and save report in PDF format. To understand the source code for non-programmers was still very hectic job, so we count the level of cyclomatic complexity and visualize source code metrics in 2 dimensional and 3 dimensional charts and graphs.

McCabe refers to some typical software analysis metrics for static structure as being, lines of code, number of functions or classes, or complexity regarding graph theoretical measures, such as cyclomatic complexity.

1.5 Proposed Methodology

Types of Methodology

- Basic vs. Applied
- Qualitative vs. Quantitative

Our methodology which we use in our prototype is

- Applied
- Qualitative

Our proposed research methodology is Applied and Qualitative. Applied because through this product solve the problem facing a non-programmer or even sometime a programmer. Qualitative because we explore the problem and gave the ideas to develop desktop application that has functionality to overcome the problem facing a non-programmer as well as programmers.

Qualitative Research is primarily exploratory research. It is used to gain an understanding of underlying reasons, opinions, and motivations. It provides insights into the problem or helps to develop ideas or hypotheses for potential quantitative research.

Applied research aims at finding a solution for an immediate problem facing a society or an industrial/business organization.

1.6 Proposed System

The proposed system will perform the following functionalities:

- **Browse Java package**

  SCMV provide user friendly environment that helps user to browse a package that contained bundle of .java extension files. User can browse heterogeneous .java extension files from a package. After selecting package, package path will be shown on related field.

- **Browse .java file**

  SCMV (e.g. See Appendix I-ii) also have facility to browse individual .java extension file form bunch of .java extension files in source package.

- **Choose .java file**

  When user click on browse java package or browse java file so all .java files will be appears and user can choose java file and count the metrics of selected java file.

- **Read File line by line**

SCMV (e.g. See Appendix I-ii) read individual java file line by line that are selected by the user. At each line read the metrics such as total line of code, commented lines, blank lines, data types, conditional statements, loop statements, inheritance of classes, cyclomatic complexity, no of method, name of method and no of method call when user request to count metrics.

- **Count Metrics**

  SCMV (e.g. See Appendix I-ii) scan and count metrics from individual selected java file that are mention above. SCMV categorize the metrics in multiple tab panes. It can show counted metrics into their related fields of that specified tab panes. These tab panes include:

  - *LOC:*

    It includes LOCs such as

No. of commented lines	No. of physical lines
No. of blank lines	No. of closes braces
No. of logical lines	No. of open braces

  - *Data types:*

    It includes metrics such as

No. of int	No. of short int
No. of char	No. of long int
No. of String	No. of double
No. of float	No. of Boolean

  - *Conditional Statements:*

    It includes metrics such as

No. of if	No. of catch block
No. of else if	No. of try block

No. of else	No. of cases
No. of switch	No. of finally block

- *Loop Statements:*

  It includes metrics such as

No. of for	No. of while	No. of do while

- *Inheritance:*

It includes metrics such as

No. of classes	No. of child class
No. of parent class	

- *Cyclomatic Complexity:*

  It describes the complexity of source code or programs. In SCMV Complexity can be defined according to the approach that is no of conditional statements, no of loop statements, no of cases and switch statements in each program to calculate we used formula:

  CC = no of conditional Stat + no_ of Loops + no of switch + no of cases + 1;

- *Refactoring Suggestion:*

  After extracting metrics it also provides refactoring suggestion that shows to resize or reengineer programs line of code. If a program has large number of lines of codes. SCMV shows according to LOCs status that this program has large number of codes you can use small number of lines of code so that refine the program and well understood for non-programmers.

- *Coupling:*

  SCMV (e.g. See Appendix I-ii) can count coupled class name and the name of class from which the class coupled and no of time each child class coupled with

its parent class in a package. We are analyzing every component of a class to find the existence of coupling.

- **Establish connection with Database**

  We use Xampp Server v3.2.2 for establishing a connection with database.

- **Insert counted metrics into the Database**

  SCMV (e.g. See Appendix I-ii) get counted metrics (that are above mentions) from their related fields and save these metrics into their related tables that are maintained in database using connection string, if user is connected to database otherwise receive message not connected to database.

- **Extract metrics from Database**

  SCMV (e.g. See Appendix I-ii) extract saved metrics that are set into their related rows in tables, if user is connected to database otherwise receive message not connected to database.

- **Apply visualization Techniques**

  After analyzing metrics, SCMV visualize these metrics in different 2 dimensional and 3-dimensional visualization techniques such as 3D Pie chart, 3D Line chart, 3D Stacked Area chart, 2D Waterfall chart and 2D Dual Axis chart when user made a request to do so.

- **Generate Report**

  When user click on generate report button System get counted source code metrics from database then system will create the report of source code metrics. Make sure system connected with database and metrics must be stored in database.

- **Print Report**

  When user click on print button the system loads the generated report of source code metrics that are extract from the database are send it to printer. System will check printer connection then execute command if printer is connected.

- **View Report**

  User can view report after clicking on generate report button.

- **Save Report**

  When user will click on save report button the system will execute command. User will select the drive or folder to save the report user will type desired filename with

desired extension. System will save report in user's desired location.

1.7 Frame Work of Source Code Metrics Visualizer

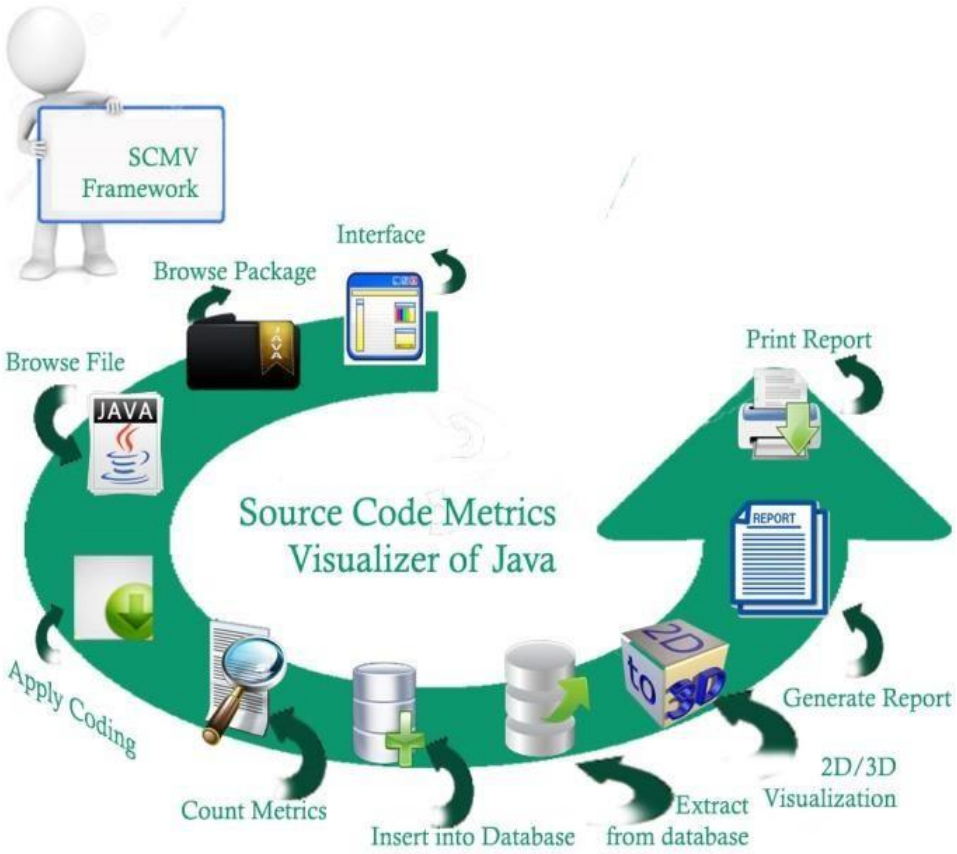


Figure 1-7: Frame work of SCMV

1.8 Main Modules

The main modules of SCMV are as follows:

- **Browser:**

  SCMV have functionality to browse individual Java file as well as Java package.

- **Analyzer:**

  SCMV have functionality to read source code line by line and analyses the metrics from selected java source file.

- **Metrics Counter:**

SCMV have functionality to count metrics from individual JAVA file as well as JAVA package.

- **Database Management:**
  SCMV have functionality to establish connection with database and insert source code metrics into database.
- **Extractor:**
  It can extract source code metrics from database.
- **Visualizer:**
  SCMV have functionality to perform different visualization techniques in 2D and 3D graphs and charts.

1.9 Expected Outcomes

- SCMV (e.g. See Appendix I-ii) reads the individual java file as well as package.
- It extracts source code metrics and stores them into database.
- Then it retrieves stored metrics form database.
- Visualize them in 2 dimensional & 3 dimensional graphs and charts.
- It generates report and also provides facility to print that report.

1.10 Tools & Technology

The tools and Technology are as follows:

- **Technology:**
  - Java Technology
- **Tools:**
  - **Design Diagrams:**
    - MS office Visio 2007 (for UML 2.0 design diagrams)
    - Star UML v.5.0 (for design diagrams)
  - **Implementation/Coding:**
    - NetBeans 7.4,8.1 or 8.0.2 (for java interface development)
  - **Database Management:**
    - Xampp Server v3.2.2 (for database development)
  - **Text Editor:**
    - MS word 2010 and latest (for creating documentation)

- **Libraries of Java Technology:**
  - Mysql-connector-java-5.1.38-bin
  - Java Free Charts
  - Mail.jar
  - Jaspersoft

1.11 Activity Index

Table 1.1: Activity Index

No.	***Activity***	***Duration***	***Deliverables***
1.	Project selection	2 weeks	
2.	Feasibility Report	2 weeks	Feasibility Report
3.	Making Proposal	2 weeks	Proposal Documentation
4.	Defend Proposal	2 weeks	Acceptance Certificate
5.	Acquire Requirement	3 weeks	Requirement Report
6.	Analysis of Requirement	3 weeks	Analysis Report
7.	Identify Scope	1 week	Scope Documentation
8.	Write Specification	2 weeks	Specification Documentation
9.	Design Interface (main links)	3 weeks	Interface
10.	Architecture Design	1 week	Architecture Design
11.	Make detailed Diagram (UML 2.0 and above)	3 weeks	Make detailed design
12.	Design Interface	3 weeks	Interface
13.	Modules Coding + Integration Coding	4 weeks	Coding Package
14.	Unit/Modules Testing	1 week	Unit Testing Report
15.	Integration Testing	1 week	Integration Test Report
16.	Final Project Defense		

CHAPTER 2

REQUIREMENT ANALYSIS

REQUIREMENT ANALYSIS

2.1 Requirement Analysis

In this phrase we will investigate the requirement of source code metrics Visualizer of Java. We will try to elicit the functional and non-functional requirements of the source code metrics Visualizer of Java. We will find the all possible requirements of this project and write it down in separate section according to their types

Some of the requirements are critical or functional, which defines the basic goal of its users and objectives of source code metrics Visualizer of Java. Other than functional requirements there are some other requirements as well which are equally important as other requirements. Let's discuss other requirements.

2.1.1 Functional Requirements

List of all functional requirements in proposed system

- **Interface Requirement**

There are the following interface requirements:

- **Signup**

  Fill the registration form to create an account.

- **Login**

  Enter Username and Password into appropriate fields and press login button.

- **Logout**

  Press Logout button to exit.

- **Open Java File**

  "JFileChooser" is used to open java files.

- **Open Java Package**

  "JFileChooser" is used to open java Packages.

- **2D Visualization**

  Button is pressed to visualize 2 dimensional graphs in java chart frame in x- axis shows total line of code in y-axis shows Cyclomatic complexity, no of method, Inheritance of classes, no. of keywords, no. of conditional statements, no of loops, no of method call, no of empty lines, no of commented lines.

- **3D Visualization**

  Button is pressed to visualize 3 dimensional graphs in java chart frame in x axis shows total line of code in y axis shows Cyclomatic complexity, no of method, Inheritance of classes, no of keywords, no of conditional statements, no of loops, no of method call, no of empty lines, no of commented lines.

- **Count Metrics**

  Button is used to count metrics from selected Java individual file or package.

- **View Counted Metrics**

  Text Fields are used to show record of counted metrics from java file.

- **View Path of File**

  Text Pane is used to display the path of file that you open and also show the path of java package.

- **Report Generation**

  Button is used to generate source code metrics report.

- **Print Report**

  Button is used to print generated source code metrics report.

- **Save Report**

  Button is used to save generated report in customized format.

- **View Report**

  Button is used to view generated report.

2.1.2 Non-Functional Requirements

List of all non-functional requirements in proposed system

- **Usability Requirement**

  There are the following usability requirements:

  - Provide user help so that easy to understand your system at first glance.
  - Once users have learned the design, they perform task quickly.
  - Its design is easy to memories.
  - Dialogue pane is used to show error.

- **Performance Requirement**

  There are the following performance requirements:

- It can support only one user to perform actions at a time.
- Its operations can perform within 5 second.
- It can store record of 100 java files in database.
- It can store record of counted metrics from each java file.

- **Product Requirement**

  There are the following product requirements:
  - It can support windows 7 and latest.
  - It can support Netbeans 7.4.1 and latest version source code.
  - It can support on 5 GB hard disk.

- **Safety Requirement**

  Following safety requirement is:
  - System should not work in the case of power failure.

- **Process Requirement**

  There are the following process requirements:
  - System must develop using Netbeans 7.4.1.
  - The development process to be used must be explicitly defined.
  - It must use mysql-connector-java-5.1.38.

- **Security Requirement**

  There are the followings security requirements:

- Only Java files can read.
- User can only open java file but not count metrics as well as store them.
- User can visualize the counted metrics into 2 dimensional and 3 dimensional graphs and charts but cannot extract these metrics from database.

2.2 Use Cases

A use case is a methodology used in system analysis to identify, clarify and organize system requirements. Use cases define interactions between external actors and the system to attain particular goals.

Following are the written description of tasks that user as well as system performs.

Table 2-1: UC 1: Sign Up

User Case Name	Sign Up
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholder & their Interests	**User:** wants fast and accurate response form sign-up **System:** wants exact name and password for quick verification
Preconditions	Running Application
Success Guarantee	User must be registered
Main Success Scenario	• User will run the application • User request for registration • System will show registration form • User fill all the information to related field • All the information store in database • User account will be created
Alternate Scenario	• Make sure all fields are filled • Invalid email • Name field contains only string • Number field contains only numbers
Frequency of Occurrences	Depends on occurrence of users.

Table 2-2: UC 2: Login

User Case Name	Log in
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholder & their Interests	**User:** typed User name and password in their related field **System:** validate the user name and password if match then login to system interfaces and provide access to the user
Preconditions	User must have an account
Success Guarantee	Valid user name and password
Main Success	• User enter user name and password
Scenario	• Verify user name and password • Login to system
Alternate Scenario	• Message shows invalid user name and password
Frequency of Occurrences	Continues

Table 2-3: UC 3: Log out

User Case Name	Log out
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User

Supporting Actor	System
Off Stage Actor	N/A
Stakeholder & their Interests	**User:** terminate the system functionalities **System:** logs the user out from interface
Preconditions	User must logged in
Success Guarantee	System resource are fully finished
Main Success Scenario	• User Press log out button • User will exit from system
Alternate Scenario	• Message shows are you sure to log out
Frequency of Occurrences	Continues

Table 2-4: UC 4: Browse Java Package

Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A

Stakeholder & their Interests	**User:** want to browse Java package Or include browsing location of java package **System:** want to open the java package that contained the java file
Preconditions	User must Login
Success Guarantee	Browse only Java package
Main Success Scenario	• User click on browse java package button • Option pane will show in the system • User select package from option pane • Path of package shown in JTextArea
Alternate Scenario	• Only Java package will be granted
Frequency of Occurrences	Continues

Table 2-5: UC 5: Count Java files in package

Use Case Name	Count Java files in package
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	User
Off Stage Actor	N/A
Stakeholders & Interests	**System:** read Java package and count no of files **User:** want to get information related to package
Preconditions	**A Java package should be selected**
Success Guarantee	**Package length must be examine in efficient period of time**

Main Success Scenario	• **User click on count metrics button** • **System reads whole package** • **Count files** • **Display file names and package length** • **Store file names and package length into the database**
Alternate Scenario	• **Make sure connected to database**
Frequency of Occurrences	**Continues**

Table 2-6: UC 6: Browse Java file

Use Case Name	Browse java file
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholders & Interests	**User:** browse java file Or include browsing location of java file **System:** store the path of selected java file
Preconditions	User must login
Success Guarantee	Browse only java file
Main Success Scenario	• User browse java file • User select java file • Path of file is stored in database
Alternate Scenario	• Only java file will be granted
Frequency of Occurrences	Continues

Table 2-7: UC 7: Read java file

Use Case Name	Read java file
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	N/A
Off Stage Actor	N/A
Stakeholders & Interests	**System:** read the selected java file line by line and each line of code can be categorized by one of the following categories: o Total no of LOCs o Blank lines o Comment lines o Physical lines o Logical lines o Data types o No. of Variables o Inheritance of classes o Cyclomatic Complexity o Loop Statements

	○ Conditional Statements ○ Coupling b/w Classes
Preconditions	**A java file should be selected**
Success Guarantee	**System should read java accurately**
Main Success Scenario	• **User will request to read file** • **System get file path** • **System will read the file**
Alternate Scenario	**None**
Frequency of Occurrences	**Continues**

Table 2-8: UC 8: Count metric from Java file

Use Case Name	Count metrics from Java file
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	User
Off Stage Actor	N/A
Stakeholders & Interests	User: want to view counted metrics in related text fields System: scan the java file line by line and find the source code metrics from each line according to category that previously defined then count the source code metrics from each line
Preconditions	A java file should be selected and read
Success Guarantee	A java file must be read by the system
Main Success Scenario	• User Press Count metrics button • System will examine java file line by line • System will find specified metrics from java file • System will count the metrics
Alternate Scenario	None
Frequency of Occurrences	Continues

Table 2-9: UC 9: Insert Counted metrics into database

Use Case Name	Insert counted metrics into database
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	N/A
Off Stage Actor	N/A
Stakeholder & Interests	**System:** take the counted metrics from their appropriate text field and store them into their suitable table field in database, if connection established from the system interface with the database
Preconditions	Source code Metrics should be counted
Success	Metrics should be counted in efficient period of time
Guarantee	
Main Success Scenario	• User Press save metrics Button • System will take counted metrics from related text fields • System will store counted metrics into database
Alternate Scenario	• Make sure connected to the database
Frequency of Occurrences	Continues

Table 2-10: UC 10: Extract metrics from database

Use Case Name	Extract metrics from database
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	N/A
Off Stage Actor	N/A
Stakeholders & Interests	**System:** extract the record of counted source code metrics from database, connection must establish between system interface and database for extraction
Preconditions	Source code metrics should be stored in database
Success Guarantee	Metrics should be stored in efficient period of time
Main Success Scenario	• User generate command to extract metrics • System will extract the record of counted metrics from database
Alternate Scenario	• Connection must establish with database
Frequency of Occurrences	Continues

Table 2-11: UC 11: Analyze metrics

Use Case Name	Analyze metrics
Scope	Source Code Metrics Visualizer of Java
Level	Sub-function
Primary Actor	System
Supporting Actor	N/A
Off Stage Actor	N/A
Stakeholders & Interests	**System:** analyze source code metrics that are extracted from the Database
Preconditions	Source code metrics should be stored and retrieve from database
Success Guarantee	Metrics should be analyzed in efficient period of time
Main Success Scenario	• User generate command to analyze metrics • System analyze the metrics
Alternate Scenario	None
Frequency of Occurrences	Continues

Table 2-12: UC 12: Visualize metrics

Use Case Name	Visualize metrics
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A

Stakeholders & Interests	**User:** view 2 dimensional and 3 dimensional graphs and charts **System:** visualize source code metrics in 2 dimensional and 3 dimensional graphs and charts
Preconditions	Source code metrics should be stored and extract from database
Success Guarantee	Source code metrics must retrieve from database in efficient period of time
Main Success Scenario	• User click on Visualize metrics button • System shows option • System will visualize metrics in 2D or 3D graphs and charts
Alternate Scenario	• System must be connected with database • Metrics should be stored in database
Frequency of Occurrences	Continues

Table 2-13: UC 13: Generate Report

Use Case Name	Generate report
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholders & Interests	**User:** generate report **System:** create the report of source code metrics
Preconditions	Source code metrics should be extract from database and graphically visualize
Success Guarantee	Run appropriate query

Main Success Scenario	• User click on view Report button
	• System get counted source code metrics from database • System will generate report
Alternate Scenario	• Make sure system connected with database • Metrics must be stored in database
Frequency of Occurrences	Continuous
	• System get counted source code metrics from database • System will generate report
Alternate Scenario	• Make sure system connected with database • Metrics must be stored in database
Frequency of Occurrences	Continues

Table 2-14: UC 14: Print Report

Use Case Name	Print report
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholders & Interests	**User:** print report **System:** load the generated report of source code metrics that are extract from the database are send it to printer to print
Preconditions	Report must be generated
Success Guarantee	Printer port should be initialized

Main Success Scenario	• User will click print button • System will check printer connection • System will execute command if printer is connected
Alternate Scenario	• Make sure printer is plug in
Frequency of Occurrences	Continues

Table 2-15: UC 15: Save Report

Use Case Name	Save report
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholders & Interests	**User:** save report **System:** save the generated report of source code metrics that are extracting from the database in customized PDF view.
Preconditions	Report must be generated
Success Guarantee	Report should be written in pdf format.
Main Success Scenario	• User will click save PDF button • System will execute command • System will write the report in PDF document • User will select the drive or folder to save report • User will type desired filename with .pdf extension • System will save report in users desired location
Alternate Scenario	• Make sure report is generated

Frequency of Occurrences	Continues

Table 2-16: UC 16: View report

Use Case Name	View report
Scope	Source Code Metrics Visualizer of Java
Level	User goal
Primary Actor	User
Supporting Actor	System
Off Stage Actor	N/A
Stakeholders & Interests	User: view report in pdf format. System: write the generated report of source code metrics that are extracting from the database and create pdf view.
Preconditions	Report must be generated
Success Guarantee	Report should be written in pdf format.
Main Success Scenario	• User will click view PDF button • System will execute command • System will write the report in PDF document • System will show the generated report in pdf view.
Alternate Scenario	• Make sure report is generated
Frequency of Occurrences	Continues

CHAPTER 3

DESIGN

CHAPTER-3

DESIGN

3.1 Design Diagram

This section contains the overview of design diagram tools that we used for creating UML diagrams. UML is abbreviated as Uniformed Modeling Language that was created to forge a common, semantically and syntactically rich visual modeling language for architecture, design, and implementation of complex software systems. UML is not a programming language. For developing UML diagrams used different tools, we used Star UML v.5.0 to design sequences and collaboration diagrams and MS Visio 2010 is used along with UML 2.0 or above to design use case diagrams [25]. MS Visio 2010 is Microsoft suit product that is used for design diagrams. UML 2.0 has formal and completely defined semantics. Its infrastructure defines the basic contracts of language on which UML is based. The main focus of infrastructure is for developers of modeling tools. Its superstructure defines user constructs of UML 2.0. The main focus of superstructure is for user community [26].

3.2 UML Diagrams

3.2.1 Use Case Diagram

Figure 3-1: UD: Source Code Metrics Visualizer of Java

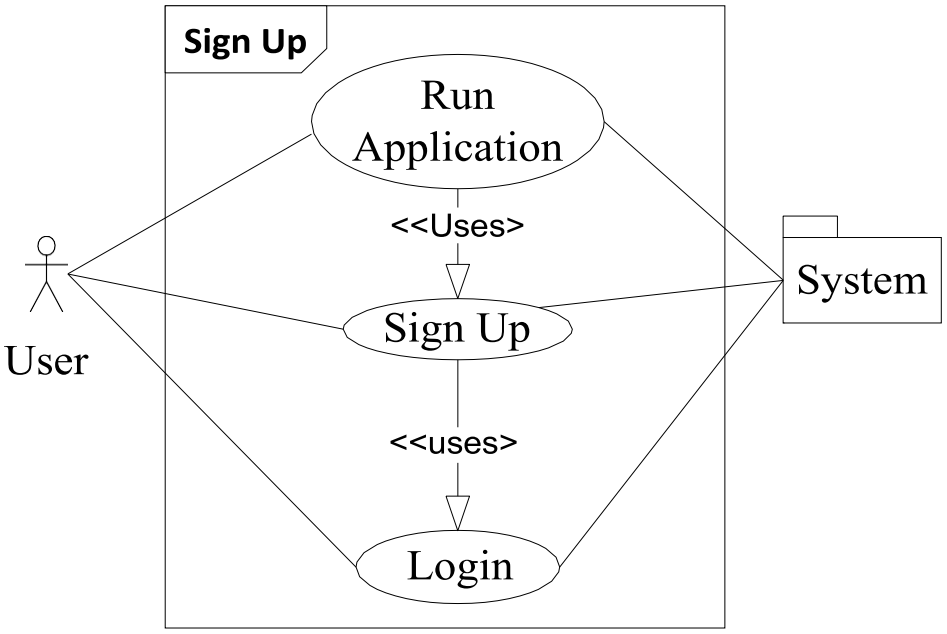


Figure 3-2: UD 1: Sign Up

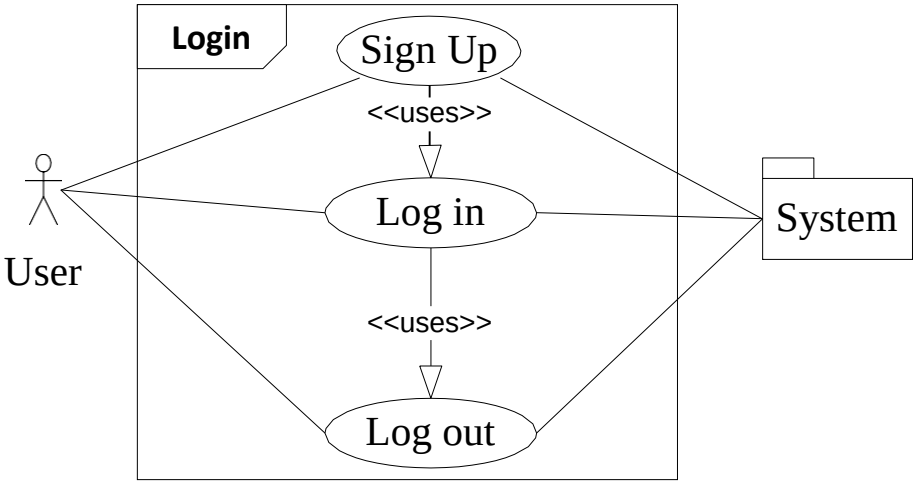


Figure 3-3: UD 2: Login

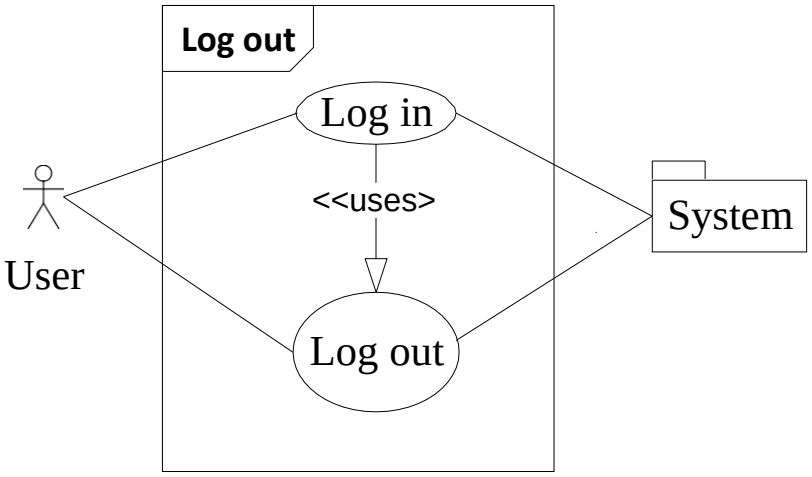


Figure 3-4: UD 3: Log out

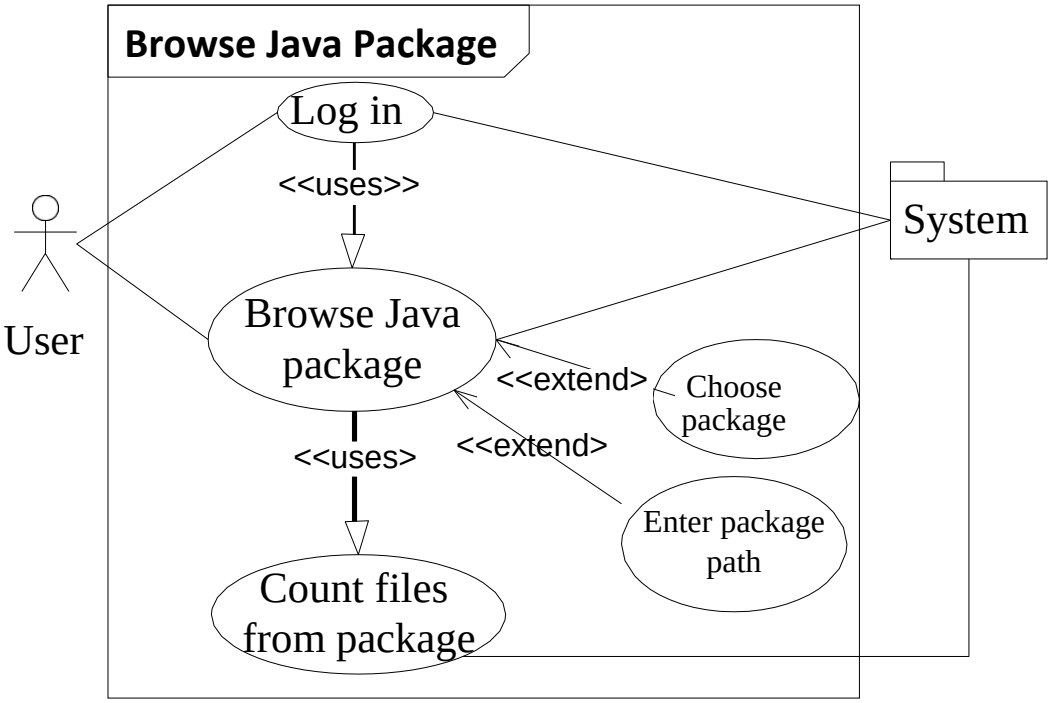


Figure 3-5: UD 4: Browse Java Package

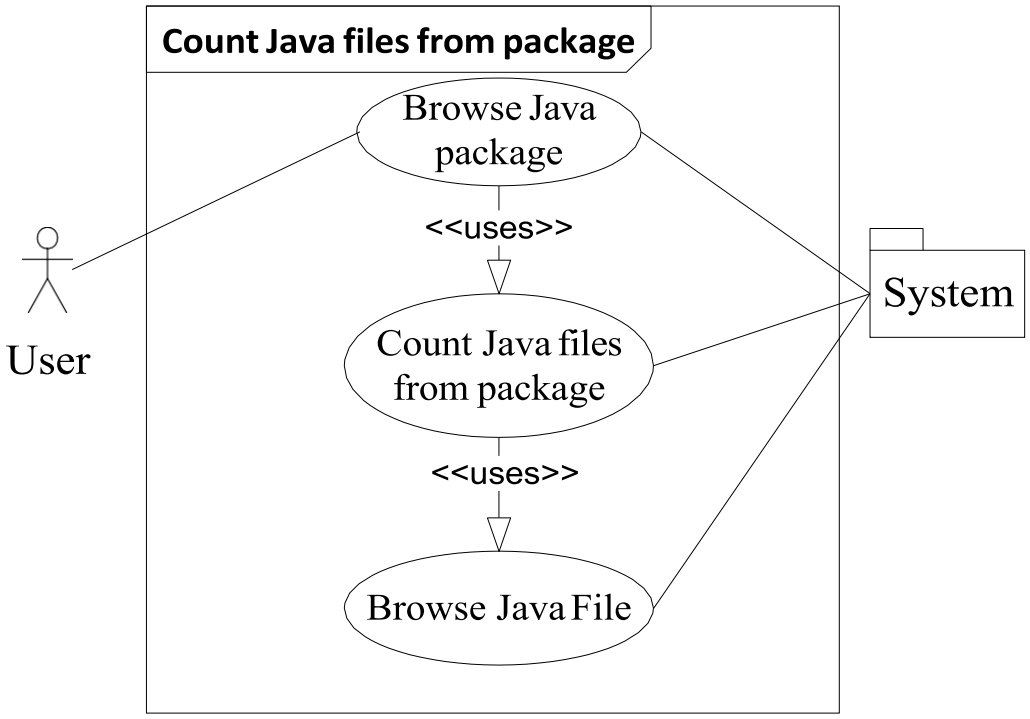


Figure 3-6: UD 5: Count Java files from package

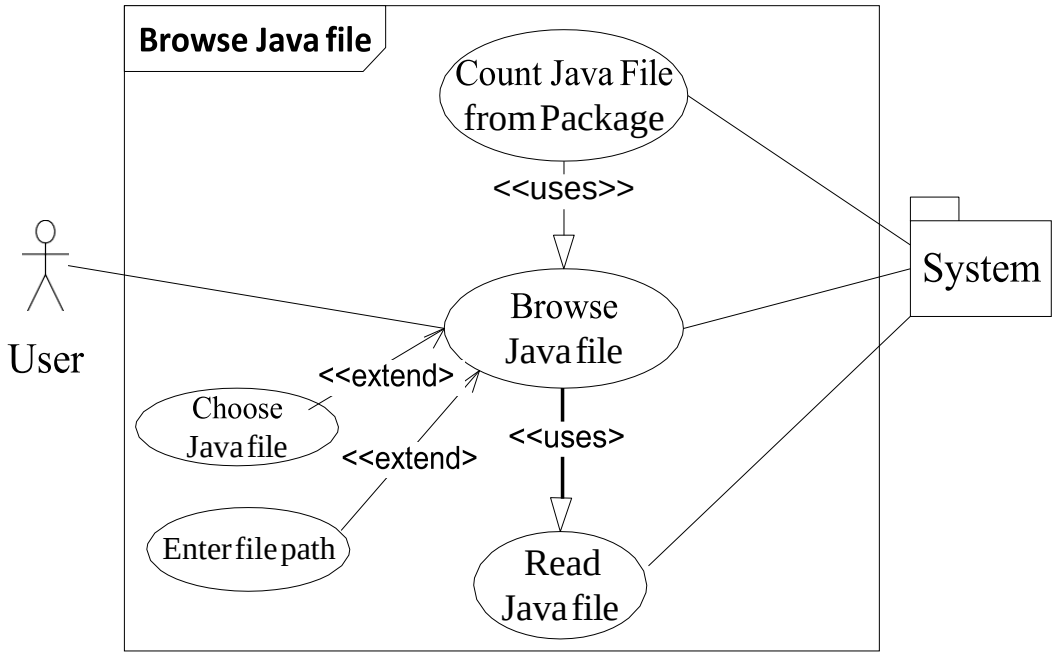


Figure 3-7: UD 6: Browse Java file

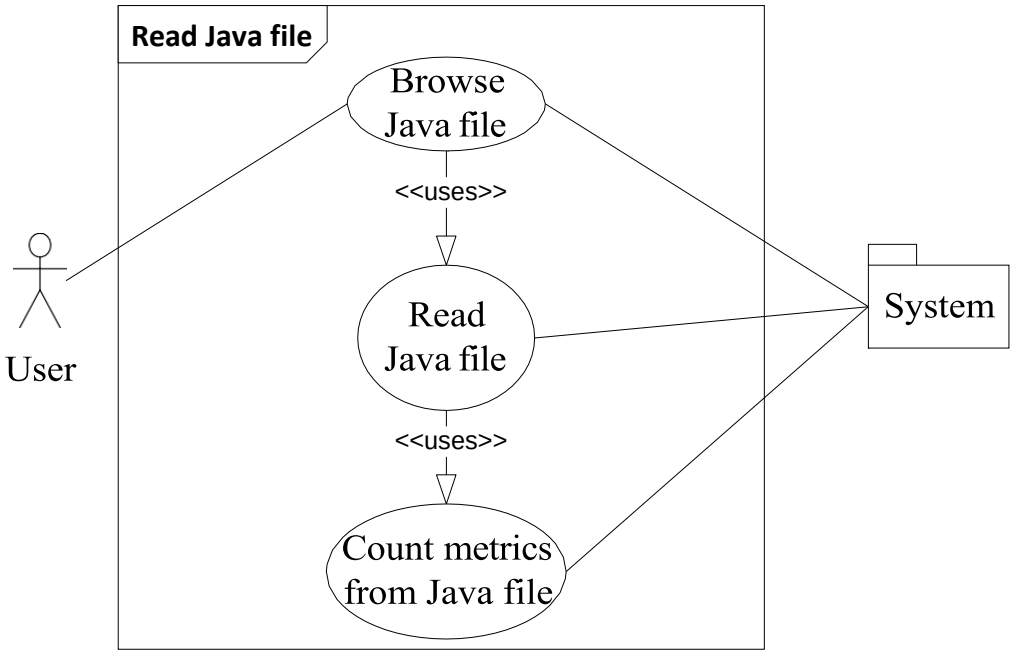


Figure 3-8: UD 7: Read java file

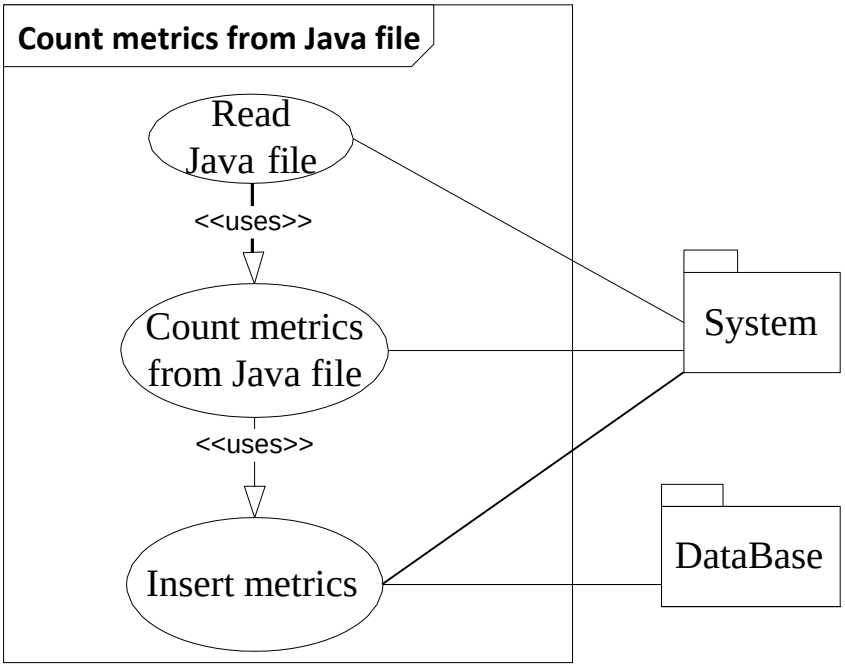


Figure 3-9: UD 8: Count metric from Java file

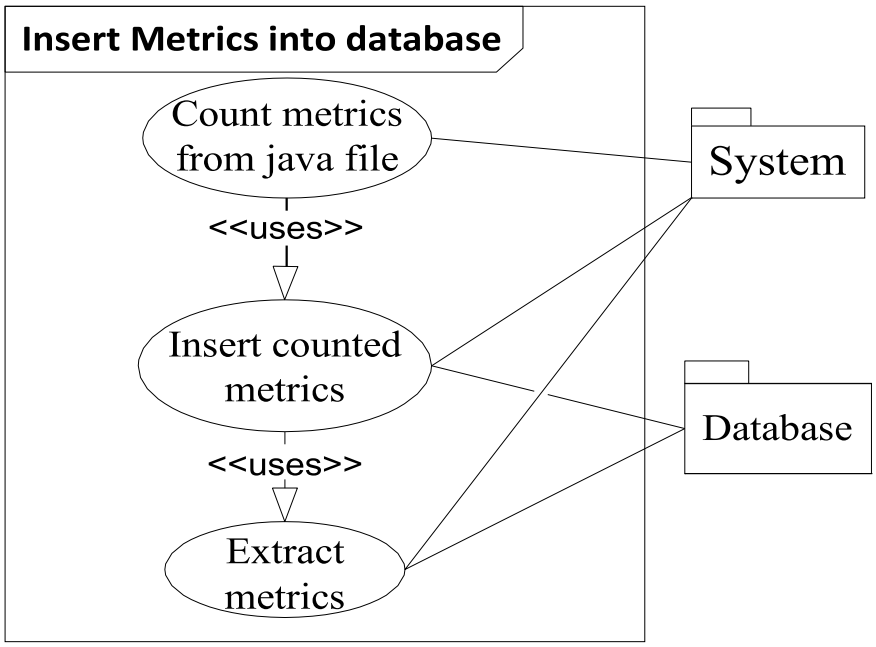


Figure 3-10: UD 9: Insert metrics into database

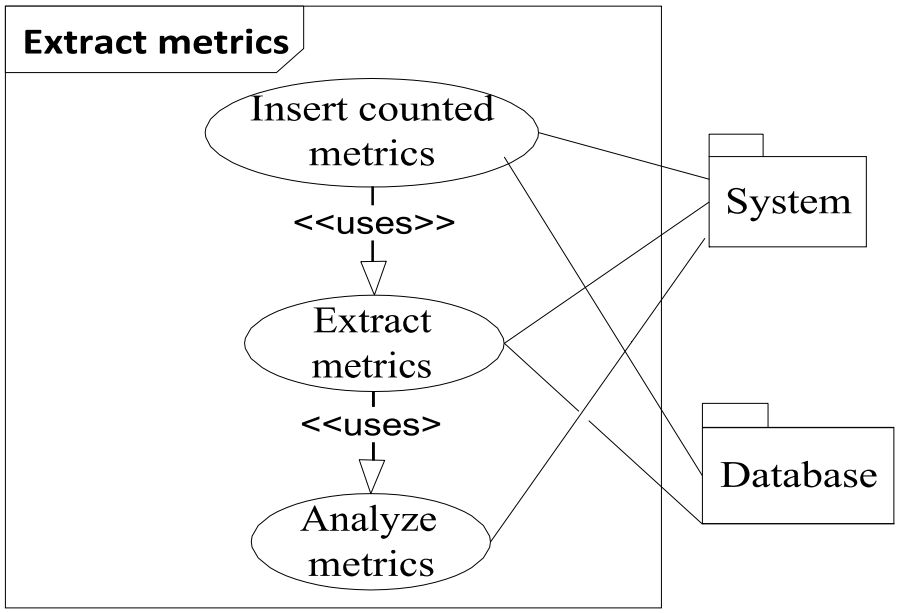


Figure 3-11: UD 10: Extract metrics from database

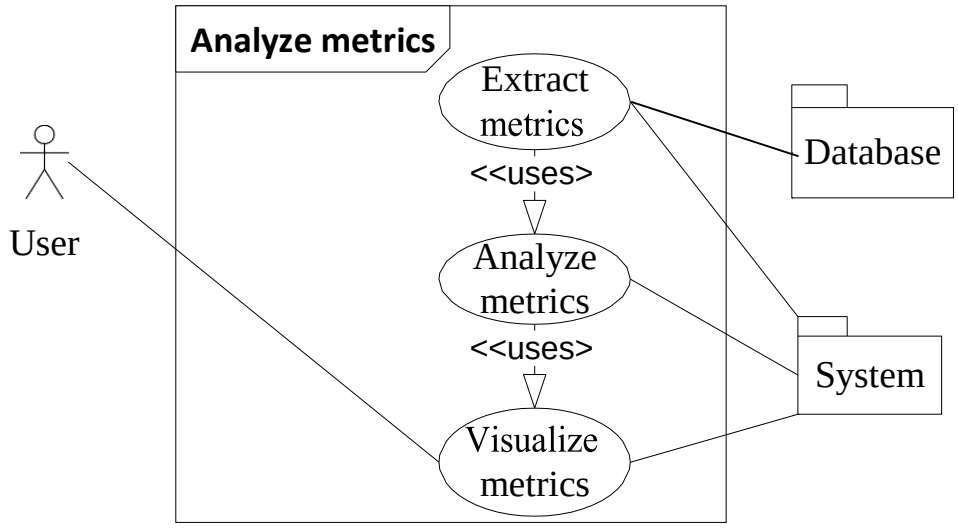


Figure 3-12: UD 11: Analyze metrics

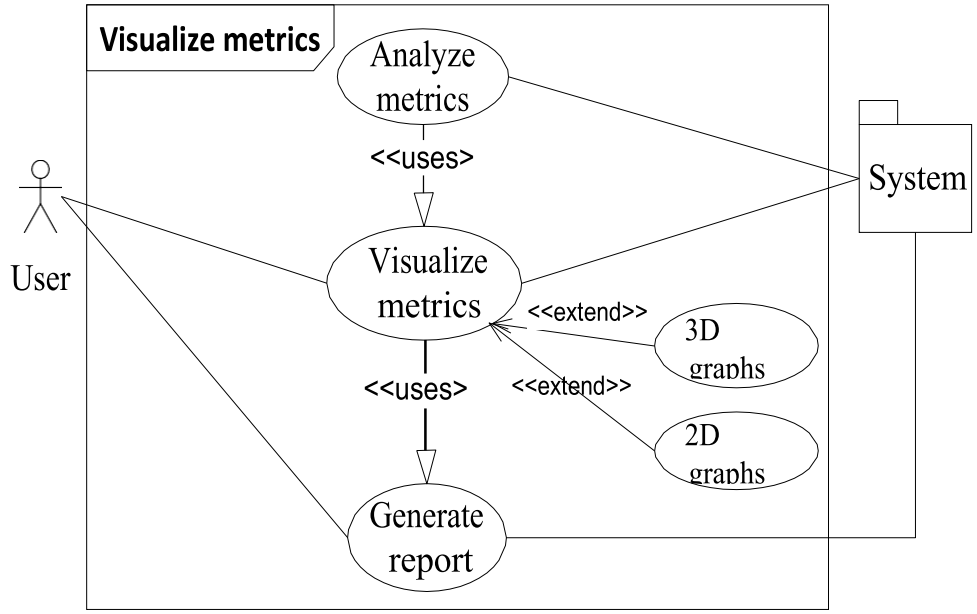


Figure 3-13: UD 12: Visualize metrics

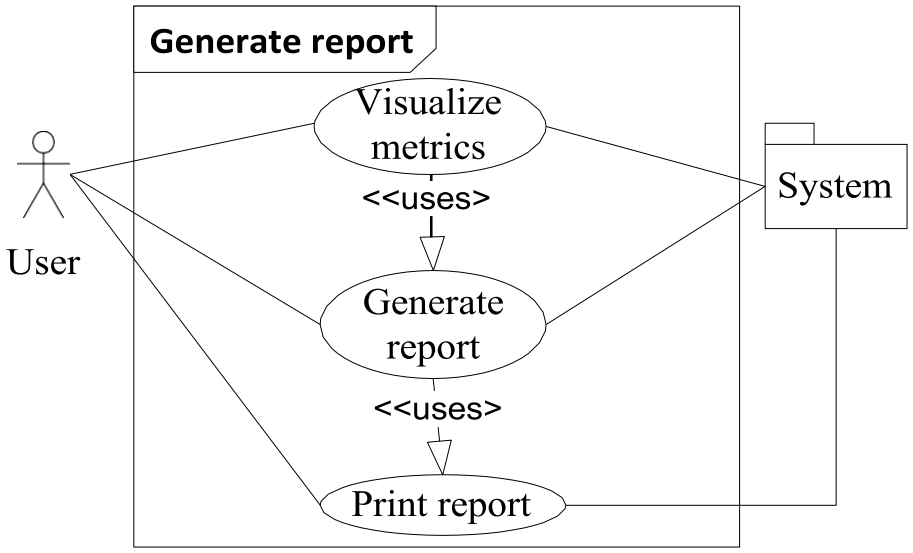


Figure 3-14: UD 13: Generate report

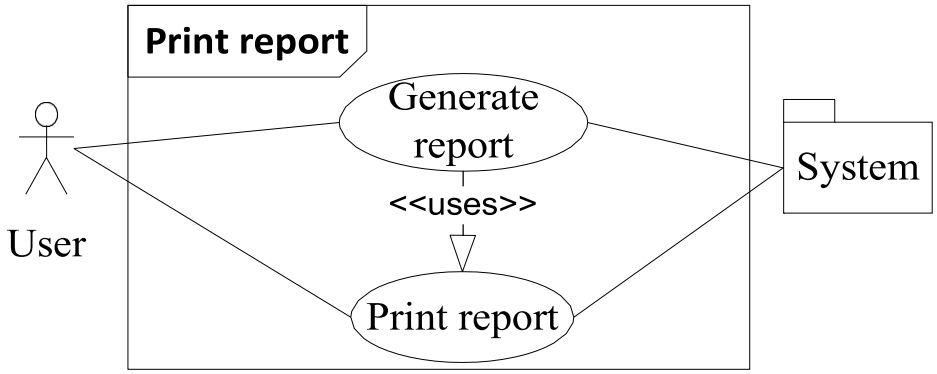


Figure 3-15: UD 14: Print report

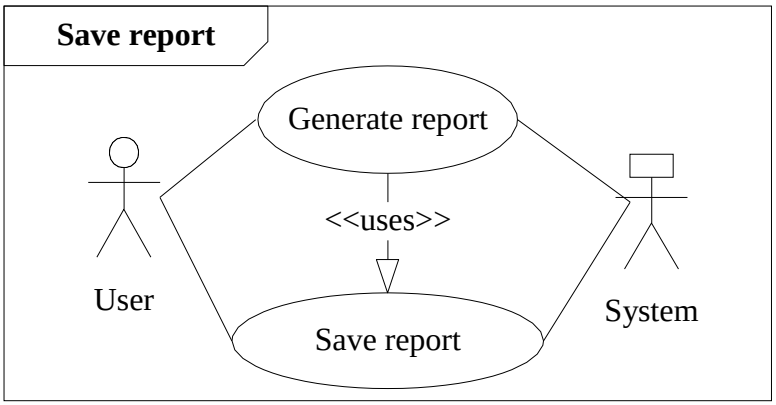


Figure 3-16: UD 15: Save Report

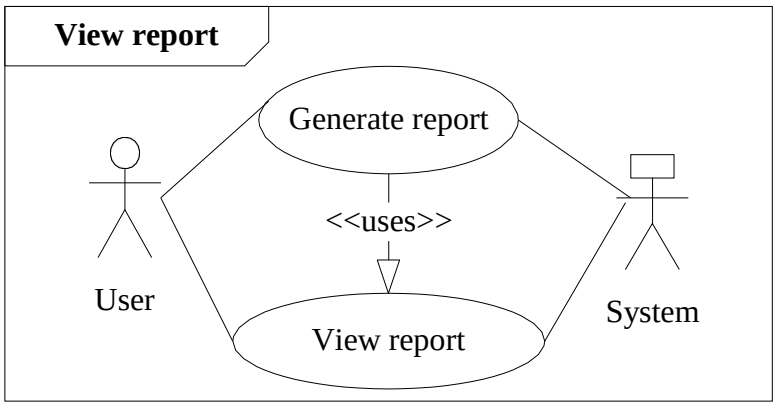


Figure 3-17: UD 16: View Report

3.2.2Sequence Diagrams

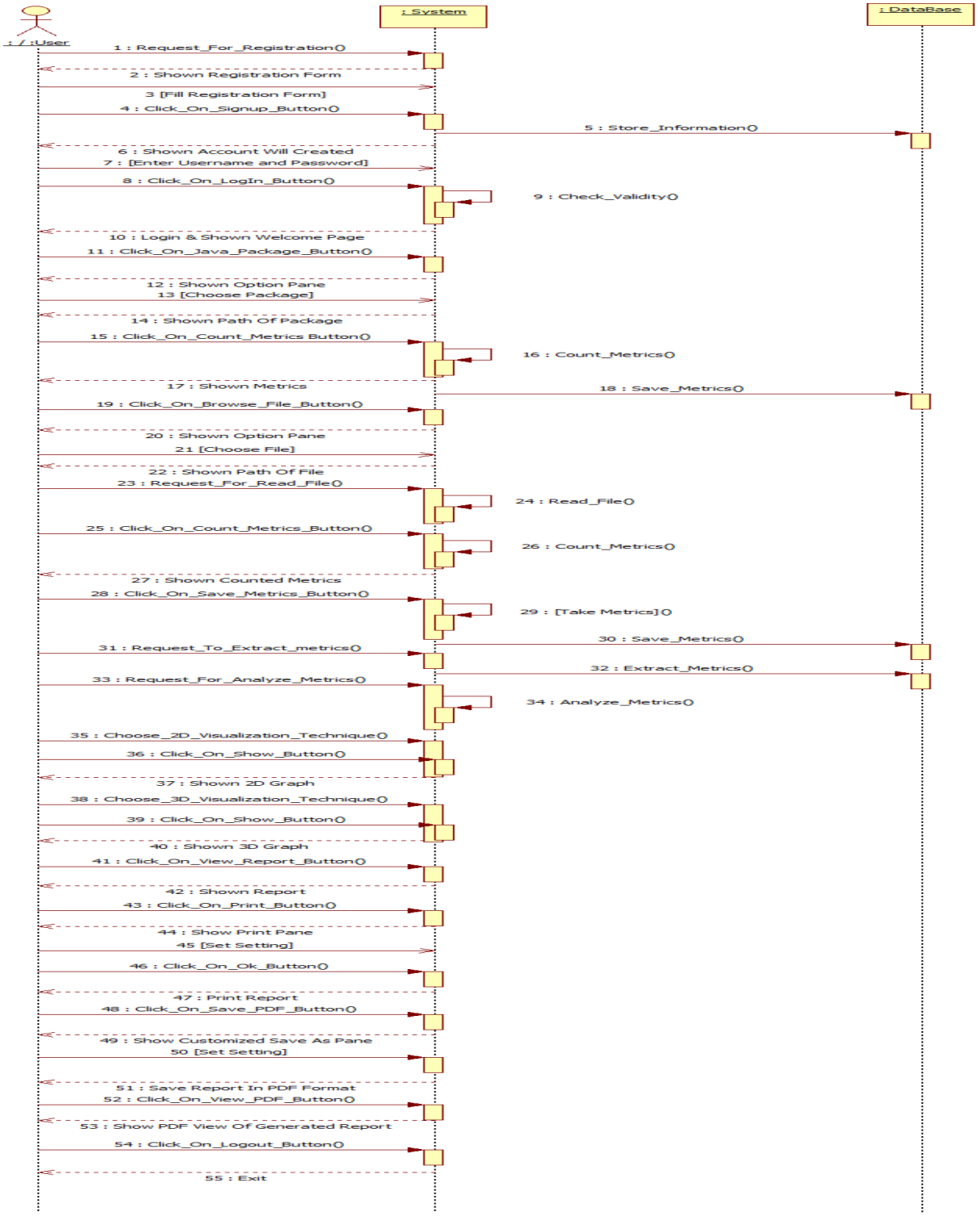


Figure 3.18: SD: Source Code Metrics Visualizer of Java

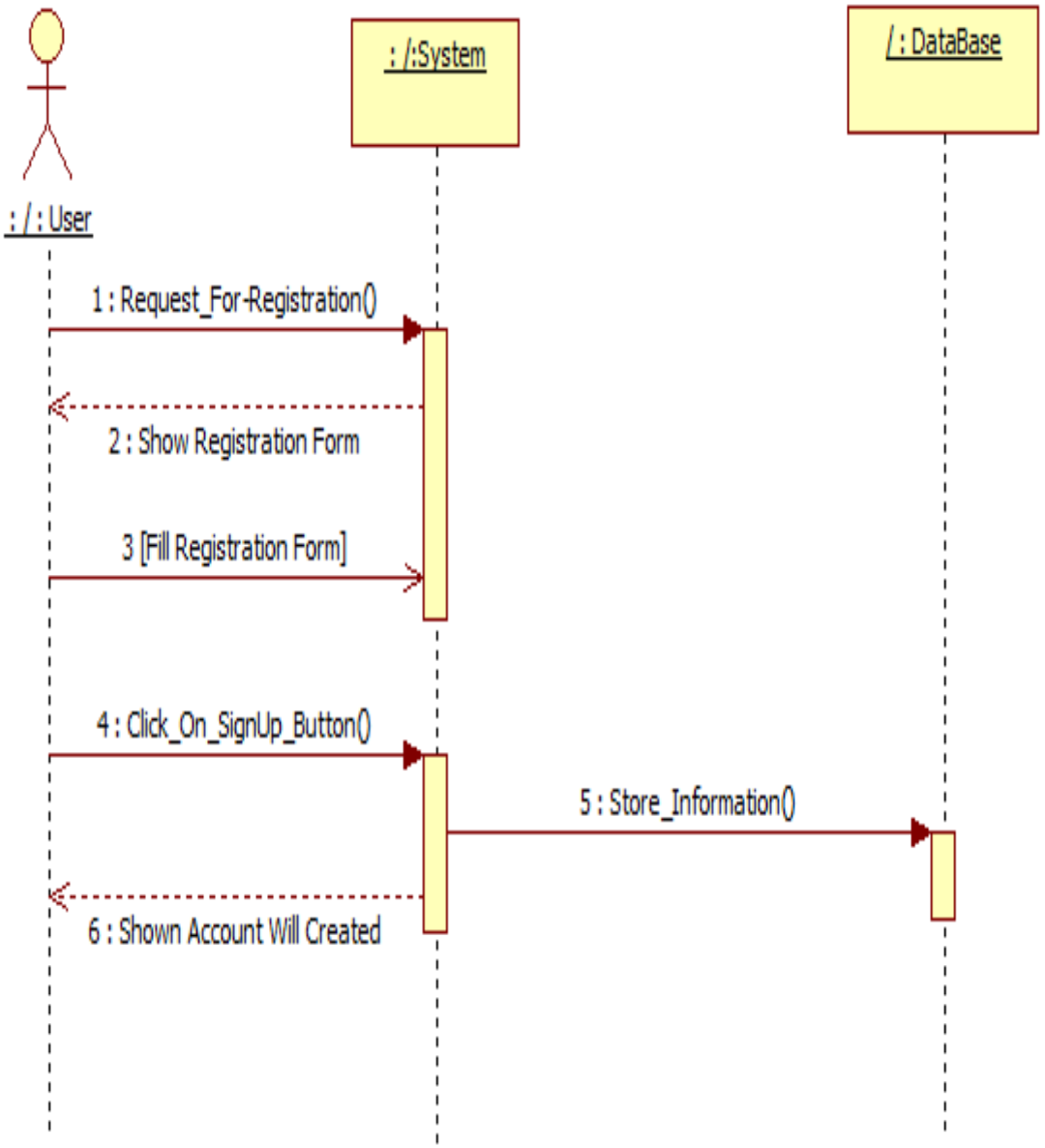


Figure 3-19: SD 1: Sign Up

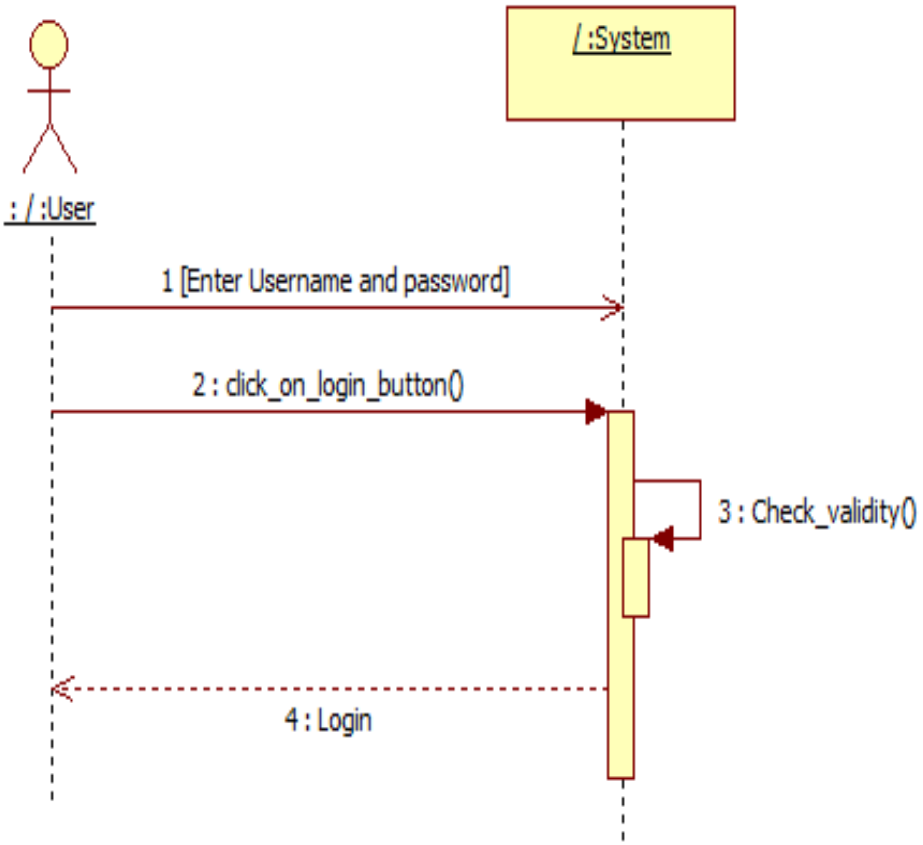


Figure 3-20: SD 2: Login

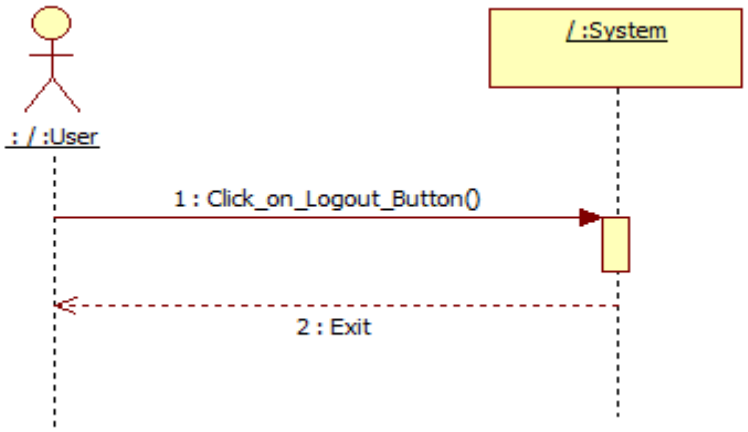


Figure 3-21: SD 3: Log out

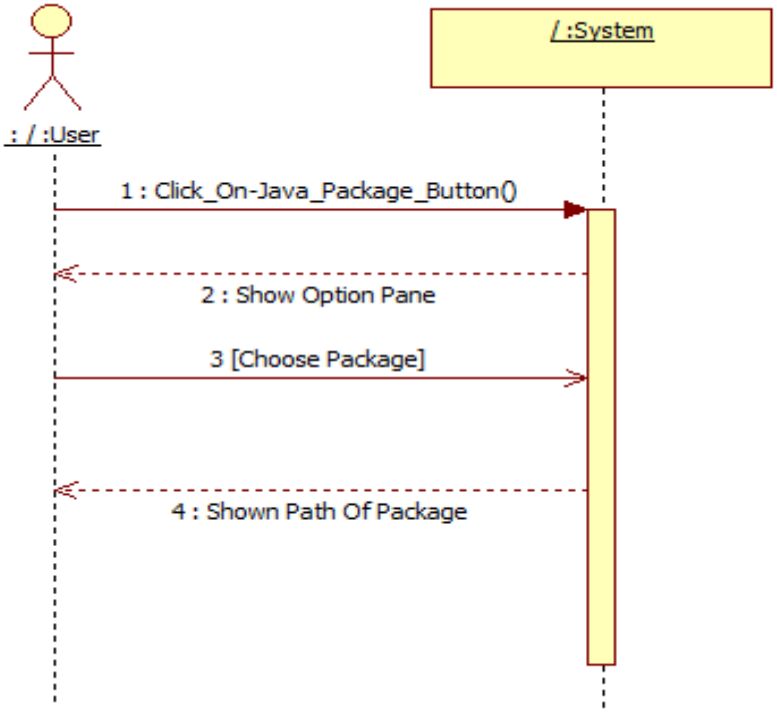


Figure 3-22: SD 4: Browse Java Package

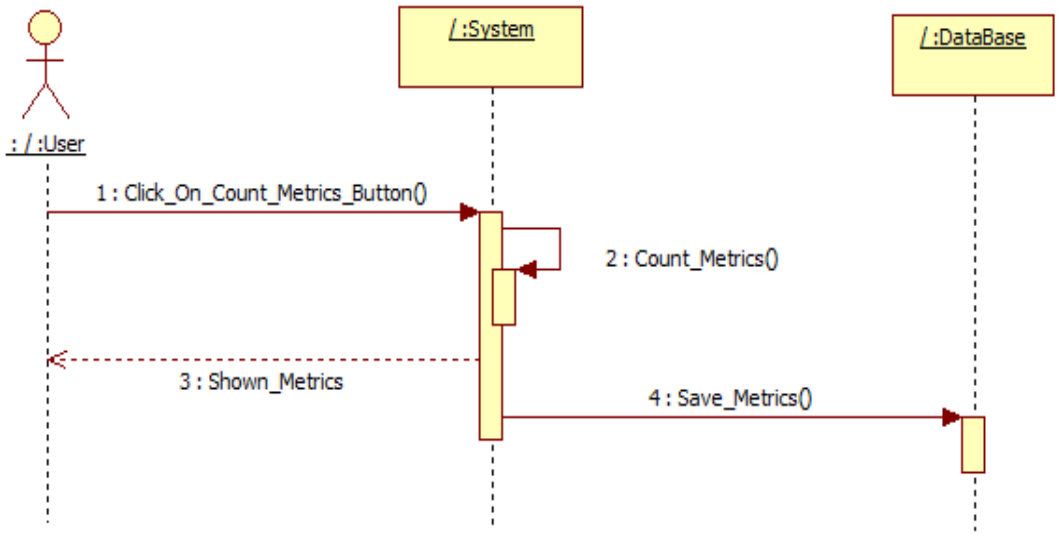


Figure 3-23: SD 5: Count Java files in package

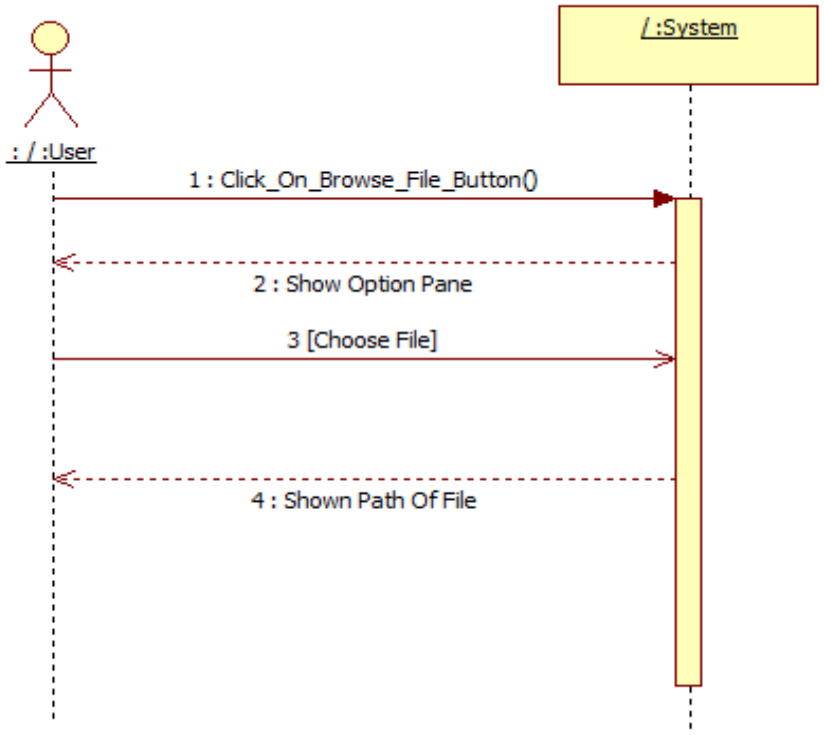


Figure 3-24: SD 6: Browse Java file

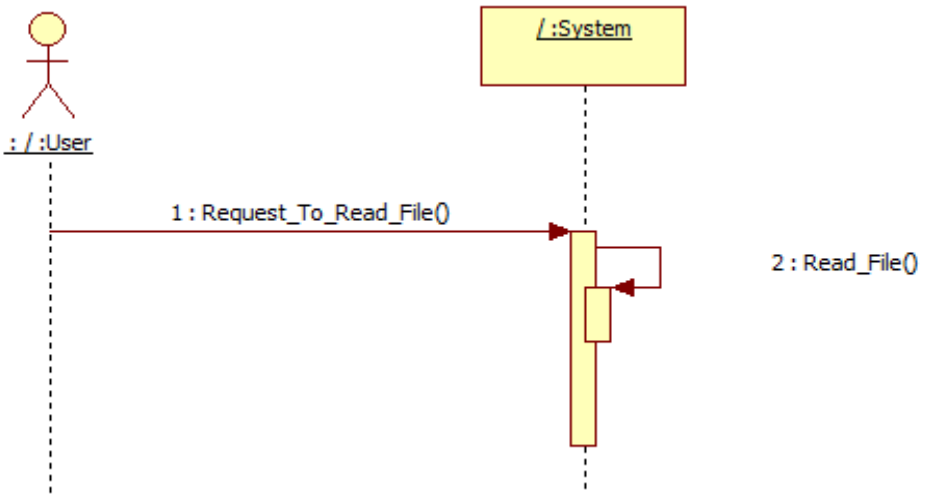


Figure 3-25: SD 7: Read java file

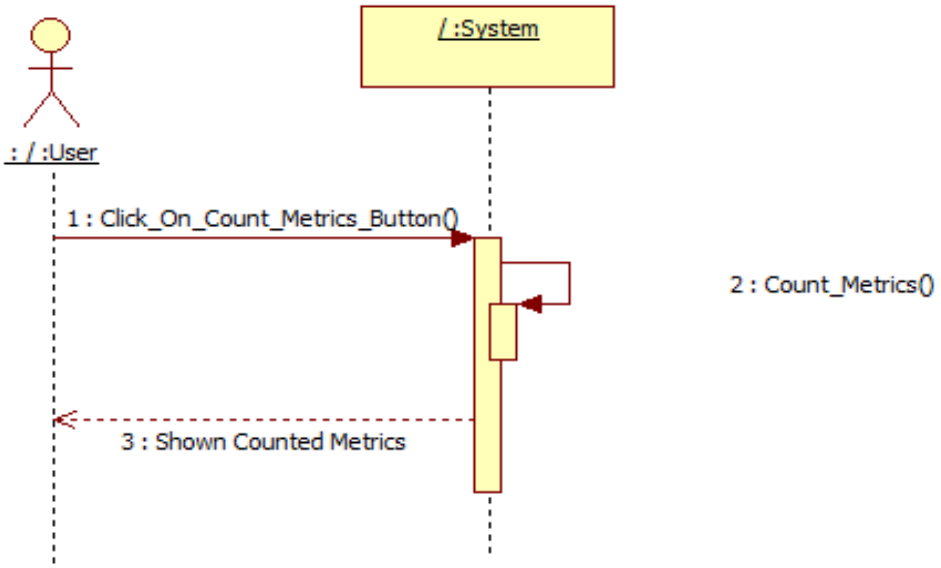


Figure 3-26: SD 8: Count metric from java file

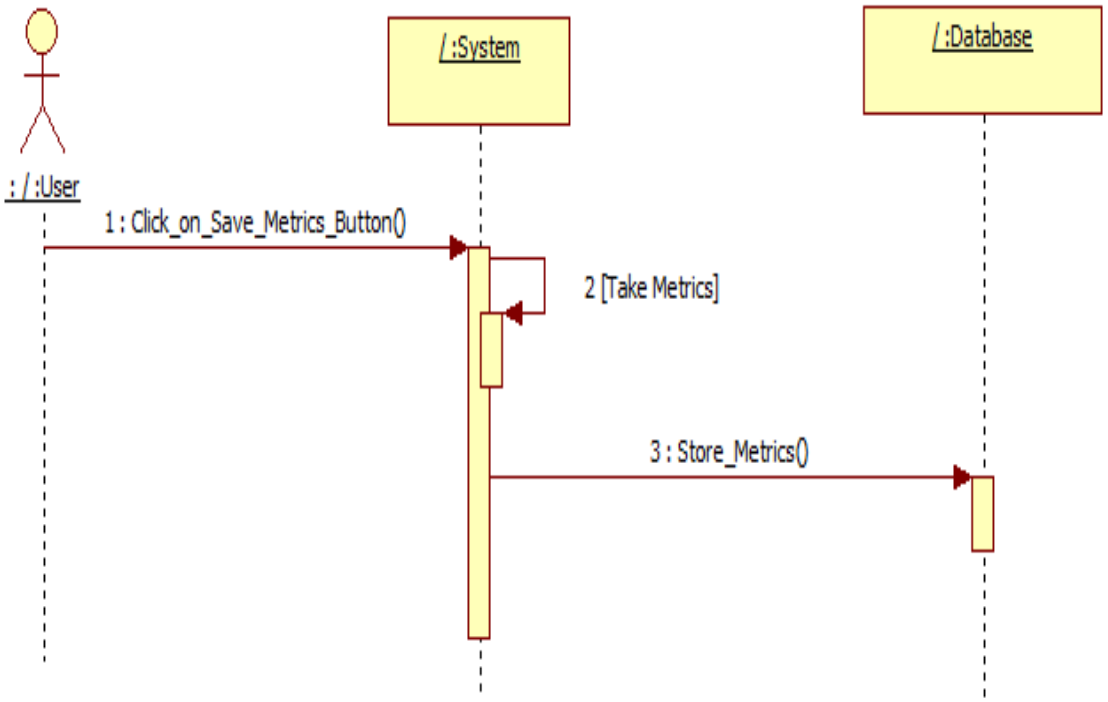


Figure 3-27: SD 9: Insert counted metrics into database

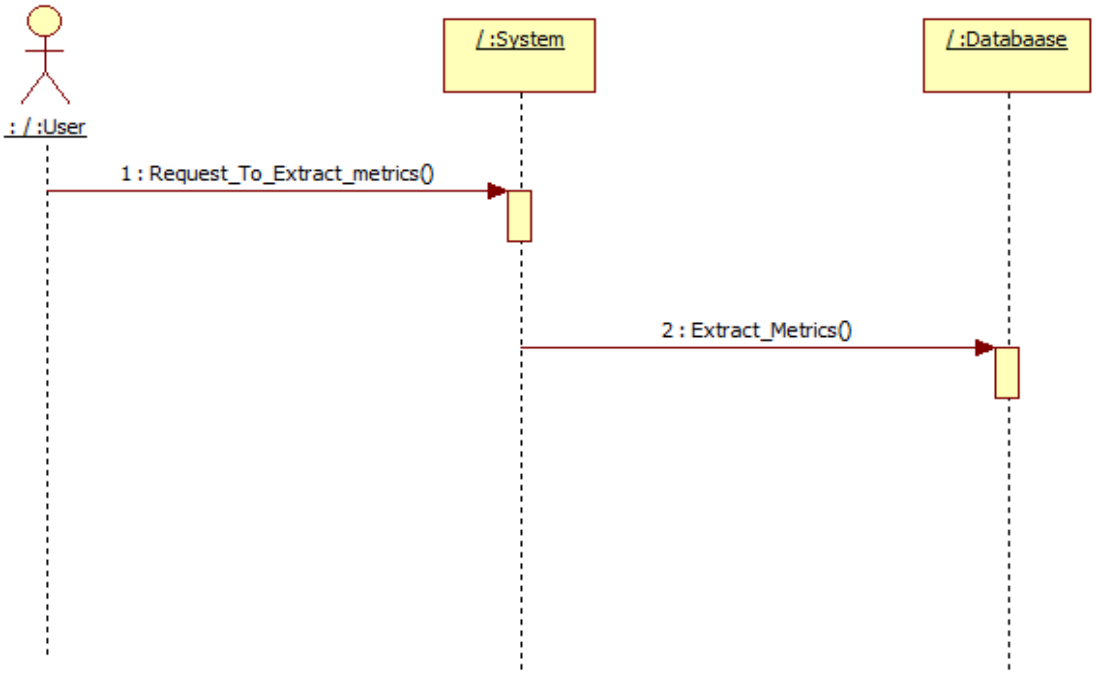


Figure 3-28: SD 10: Extract metrics from database

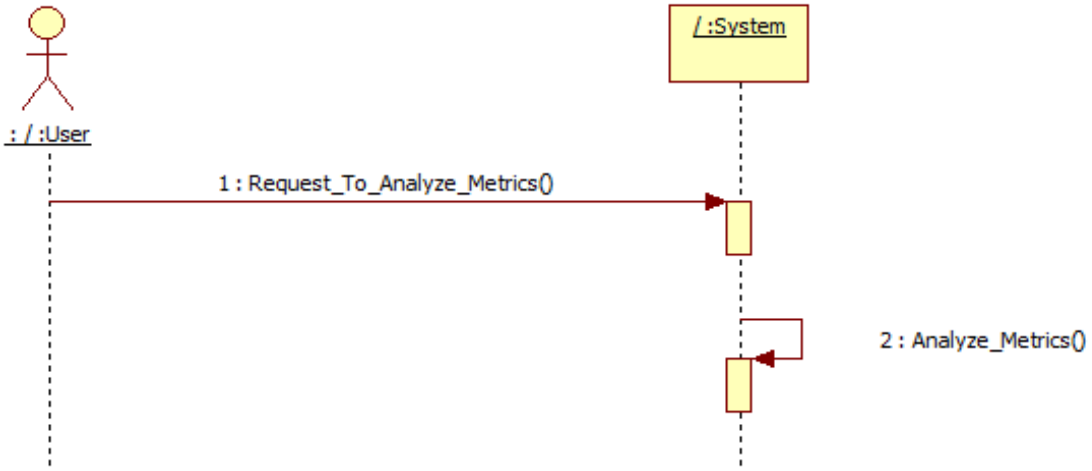


Figure 3-29: SD 11: Analyze metrics

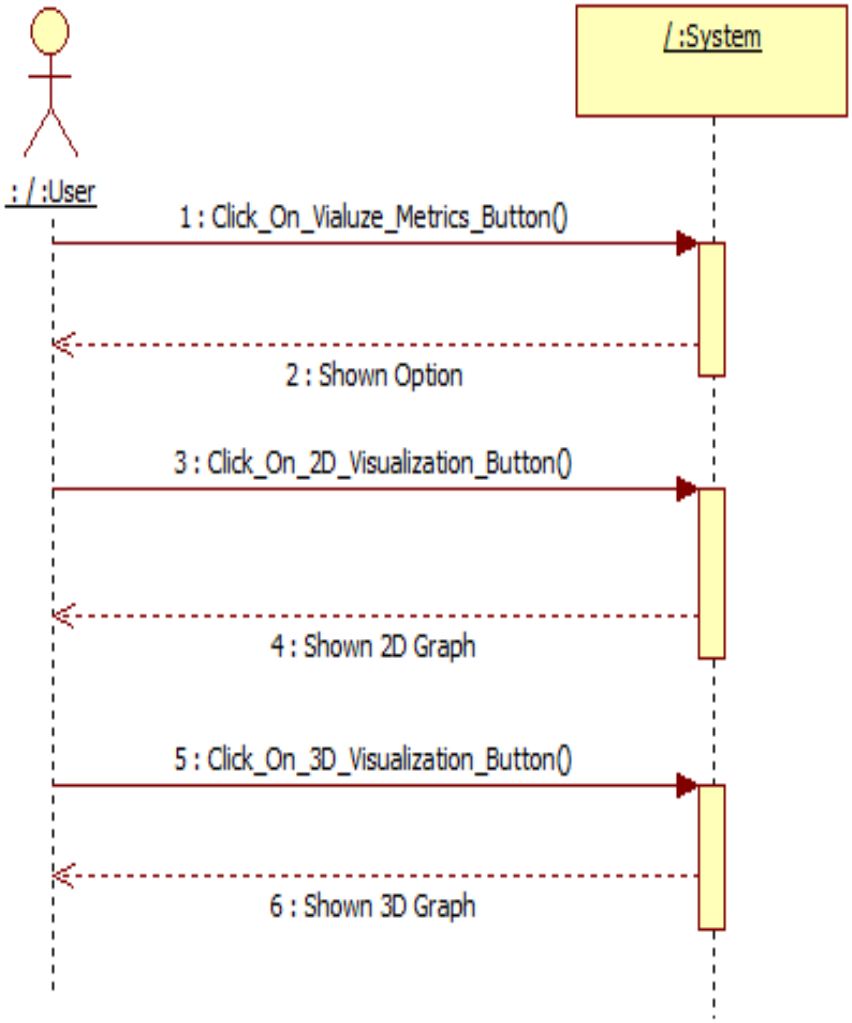


Figure 3-30: SD 12: Visualize metrics

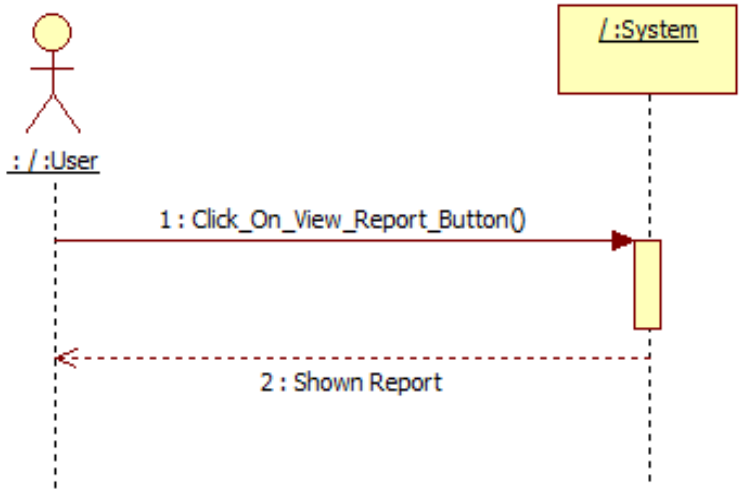


Figure 3-31: SD 13: Generate report

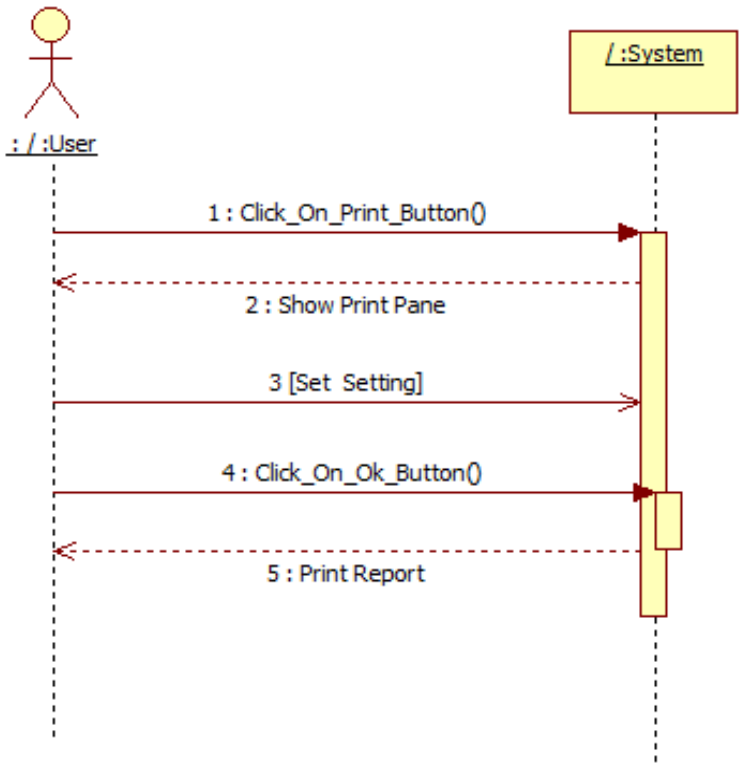


Figure 3-32: SD 14: Print report

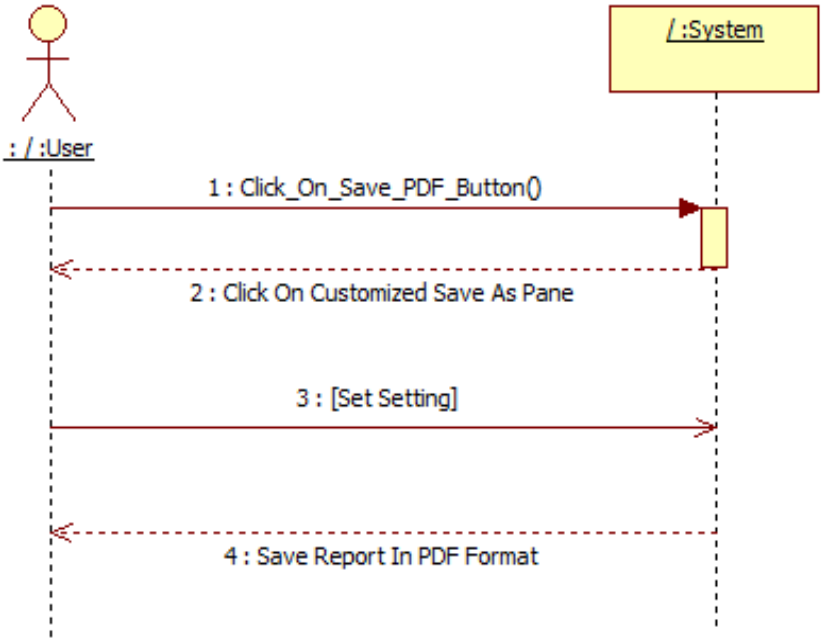


Figure 3-33: SD 15: Save report

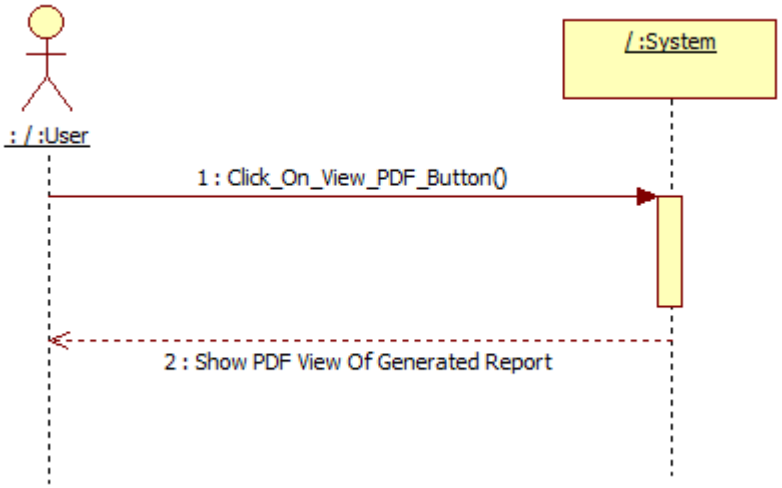


Figure 3-34: SD 16: View report

3.2.3 Collaboration Diagrams

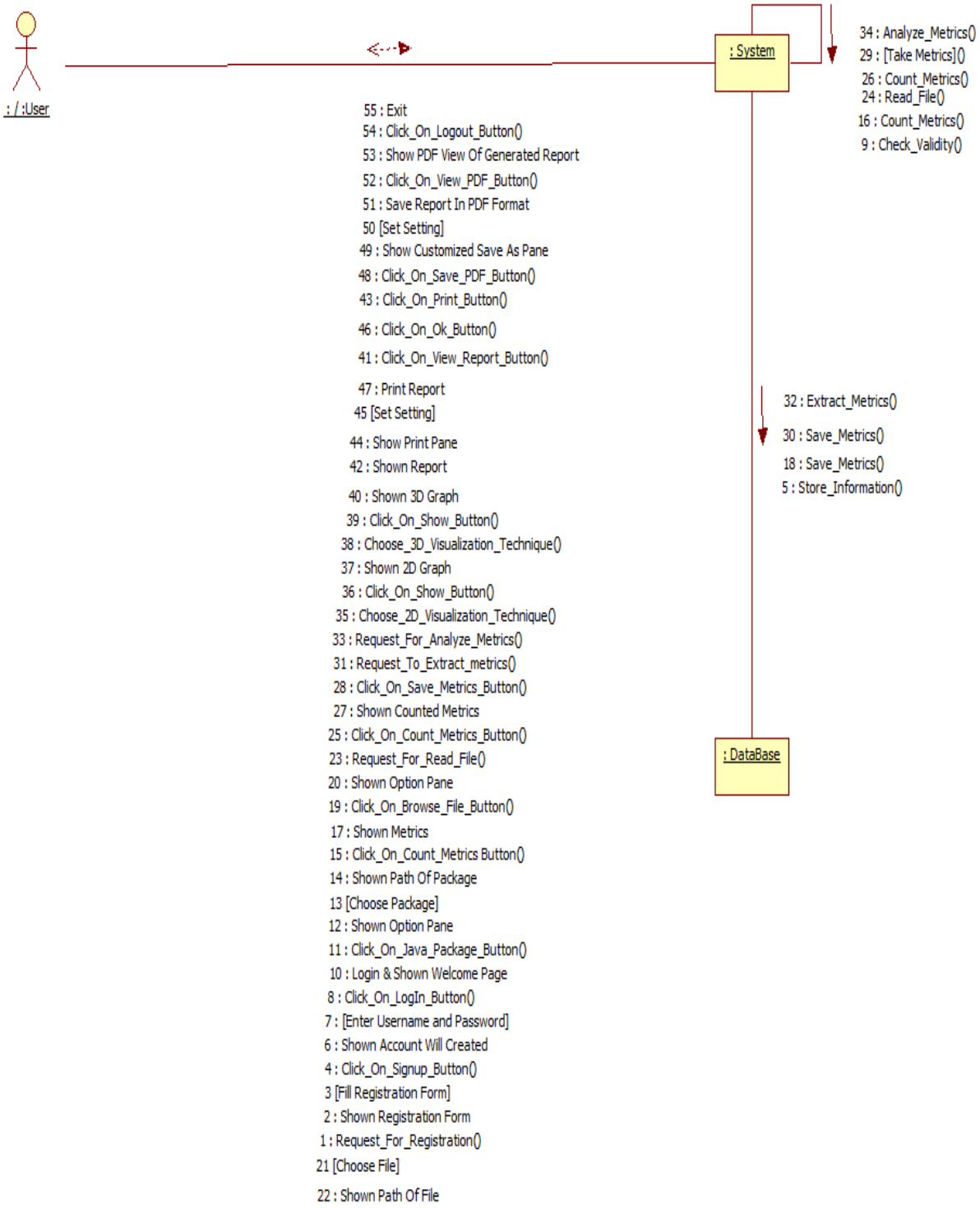


Figure 3-35: CD: Source Code Metrics Visualizer of Java

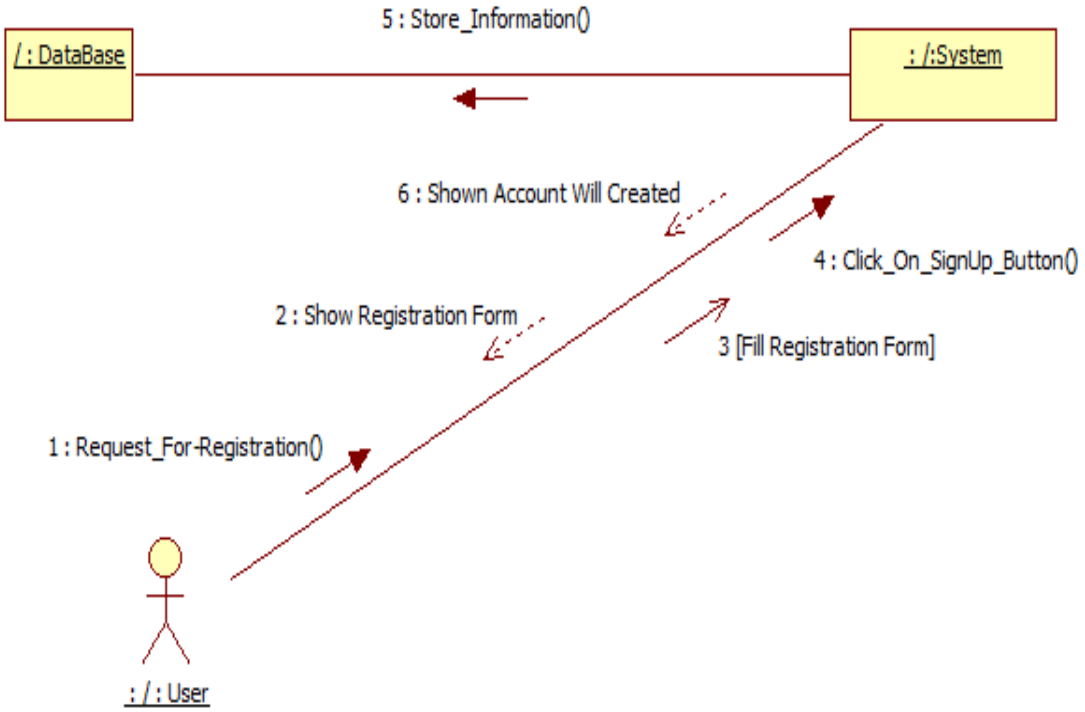


Figure 3-36: CD 1: Sign Up

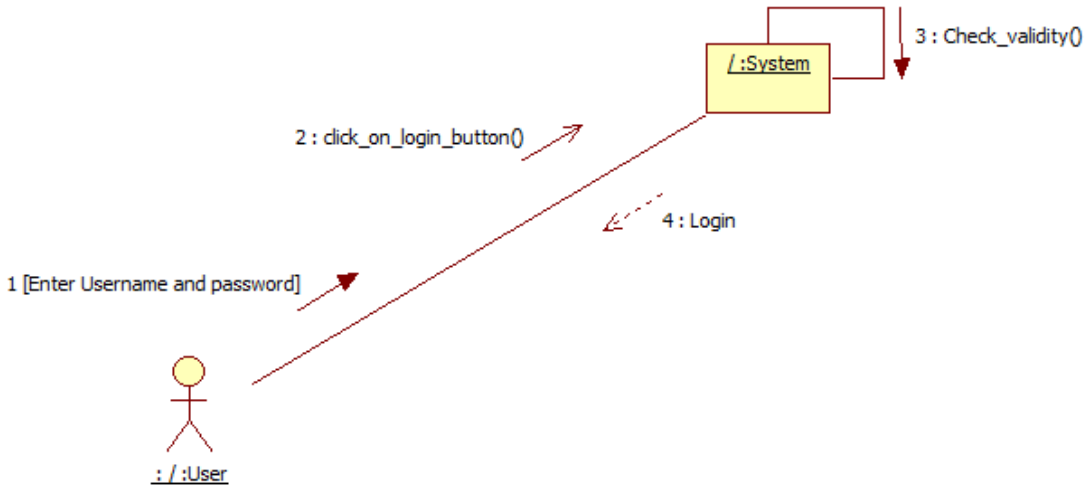


Figure 3-37: CD 2: Login

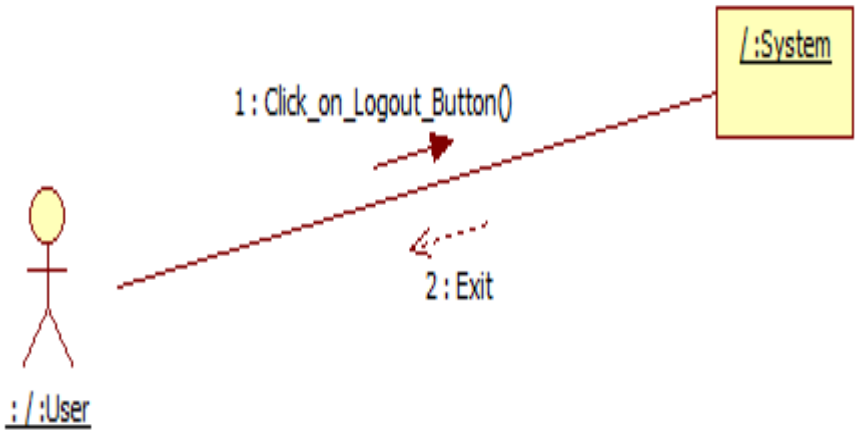


Figure 3-38: CD 3: Log out

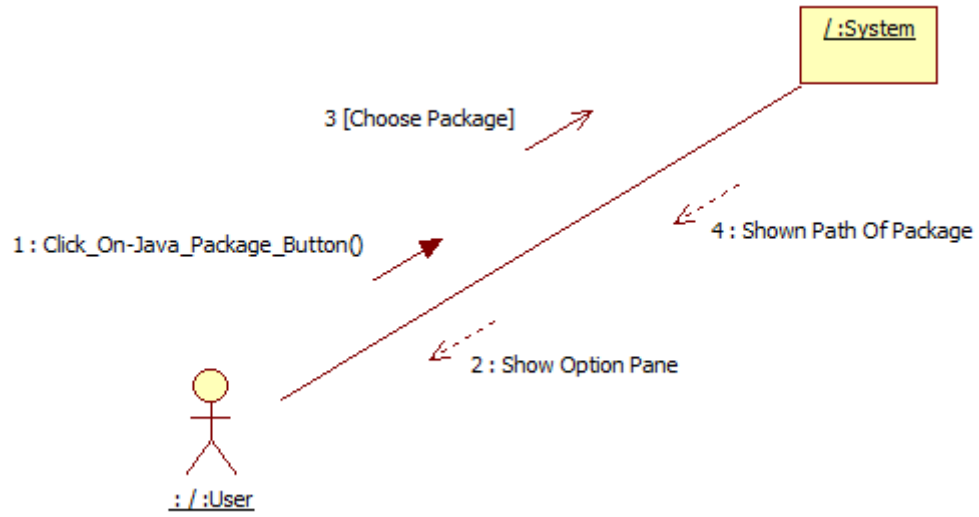


Figure 3-39: CD 4: Browse Java Package

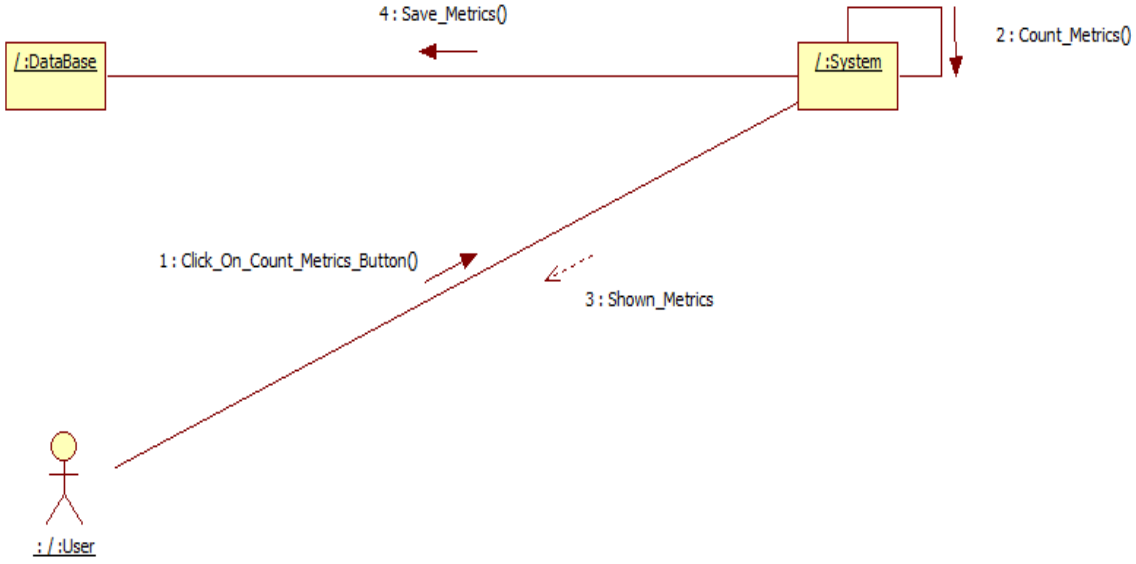


Figure 3-40: CD 5: Count files from package

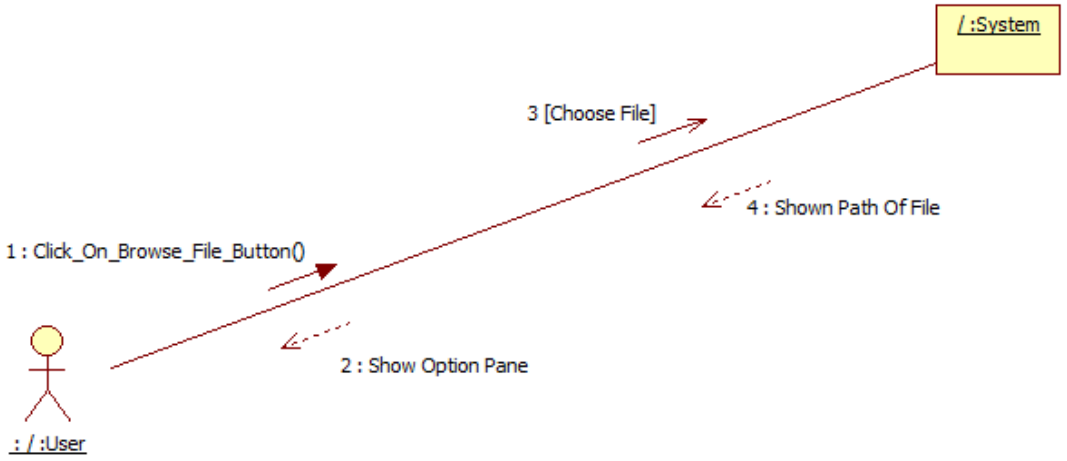


Figure 3-41: CD 6: Browse Java file

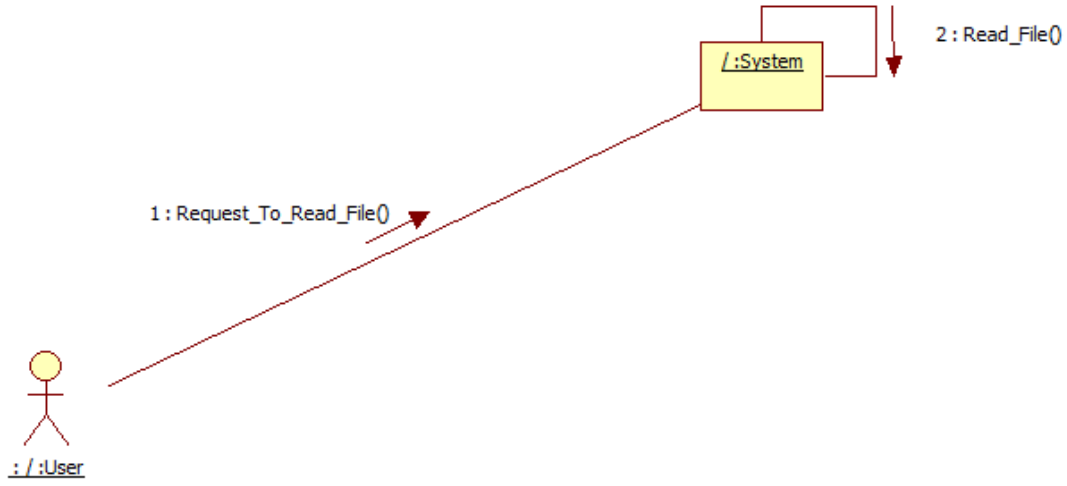


Figure 3-42: CD 7: Read Java file

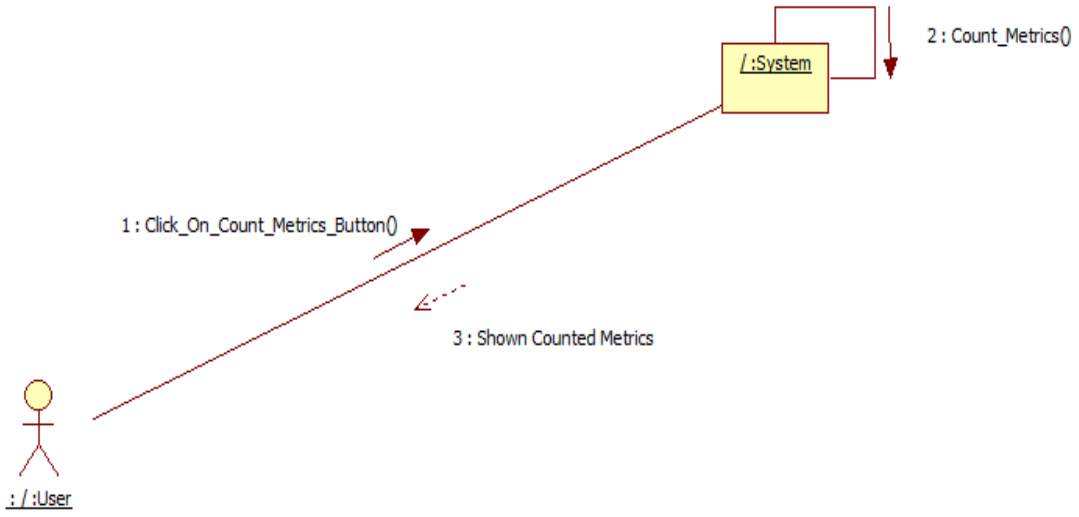


Figure 3-43: CD 8: Count Metrics from java file

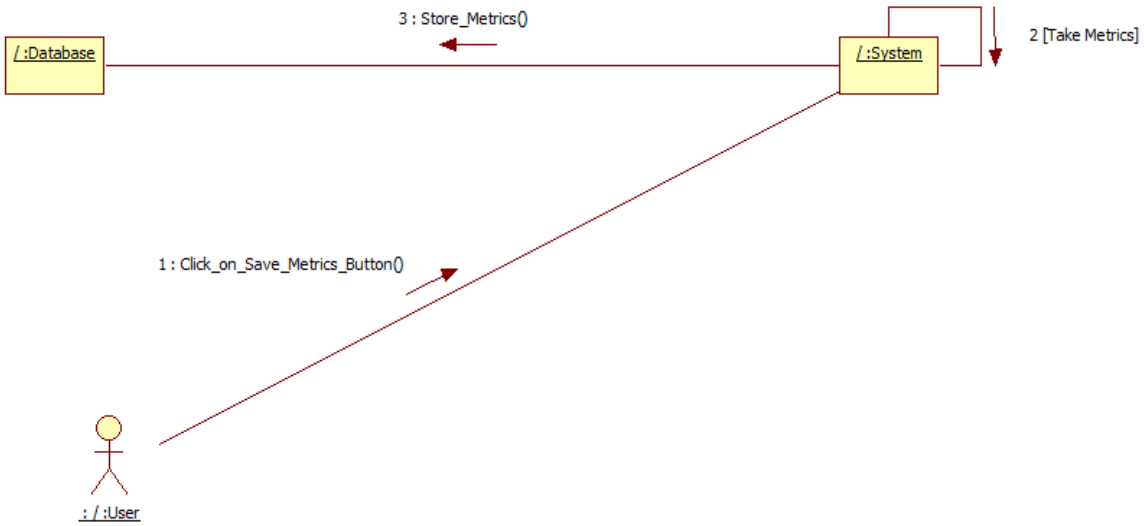


Figure 3-44: CD 9: Insert counted metrics into database

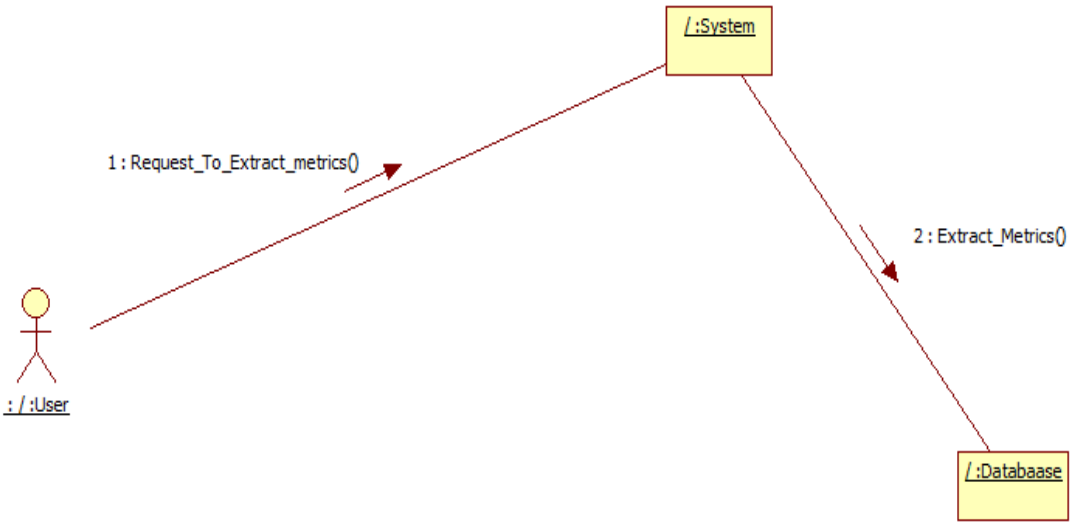


Figure 3-45: CD 10: Extract metrics from database

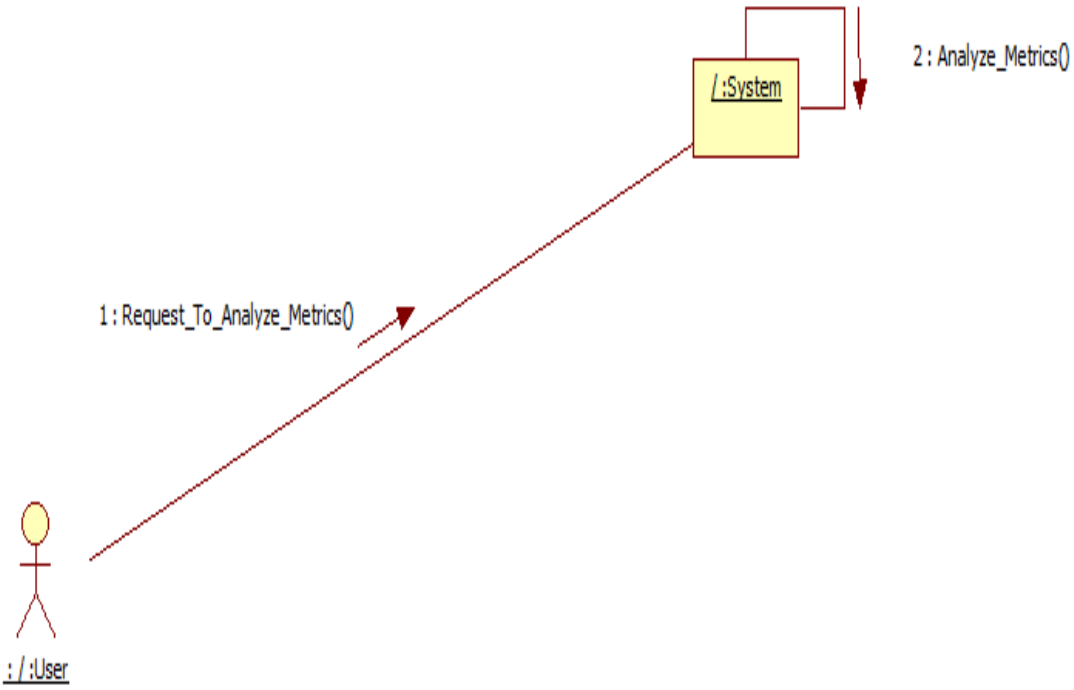


Figure 3-46: CD 11: Analyze metric

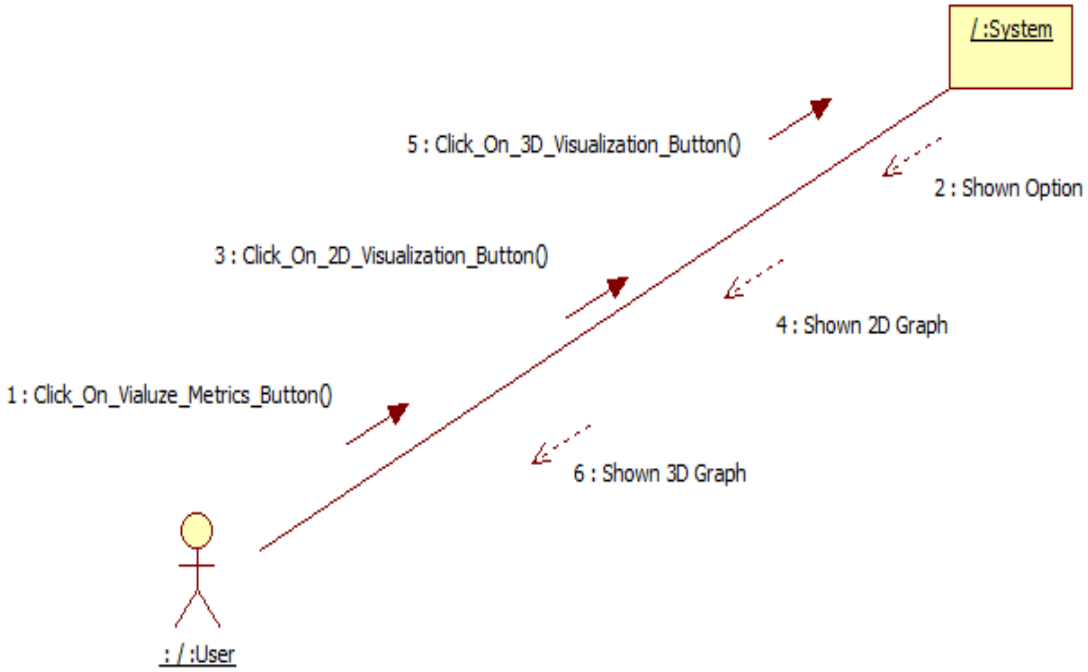


Figure 3-47: CD 12: Visualize metrics

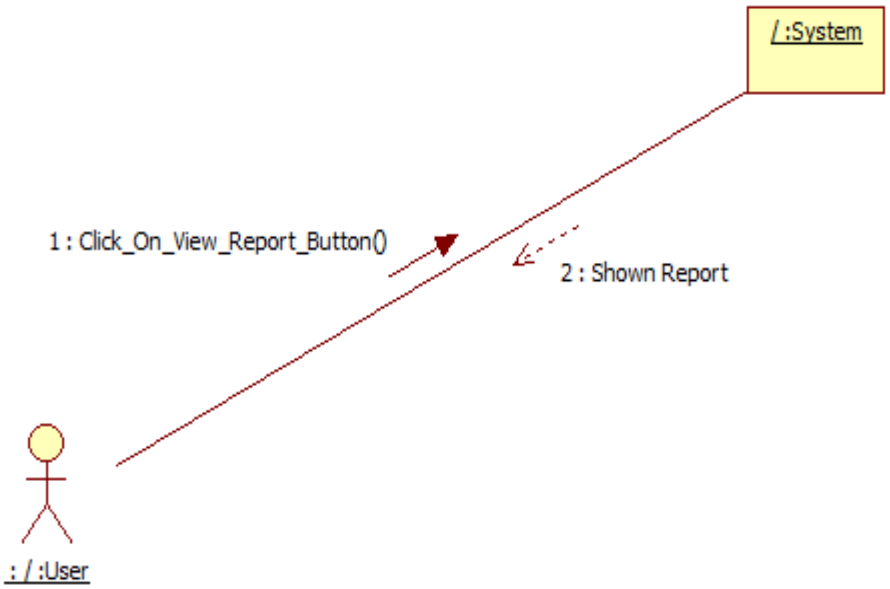


Figure 3-48: CD 13: Generate report

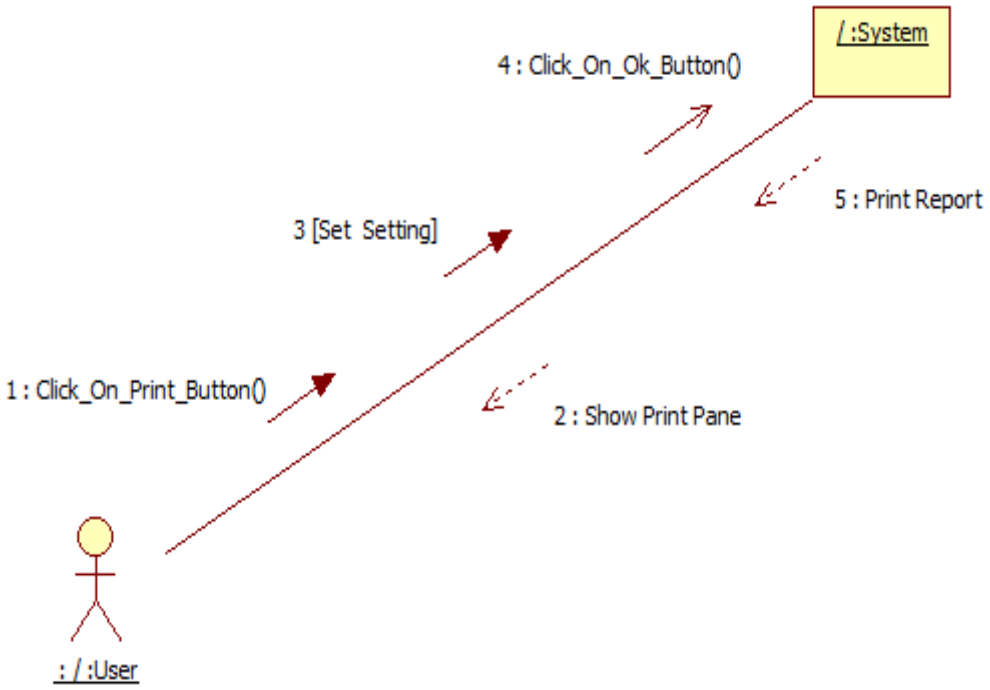


Figure 3-49: CD 14: Print report

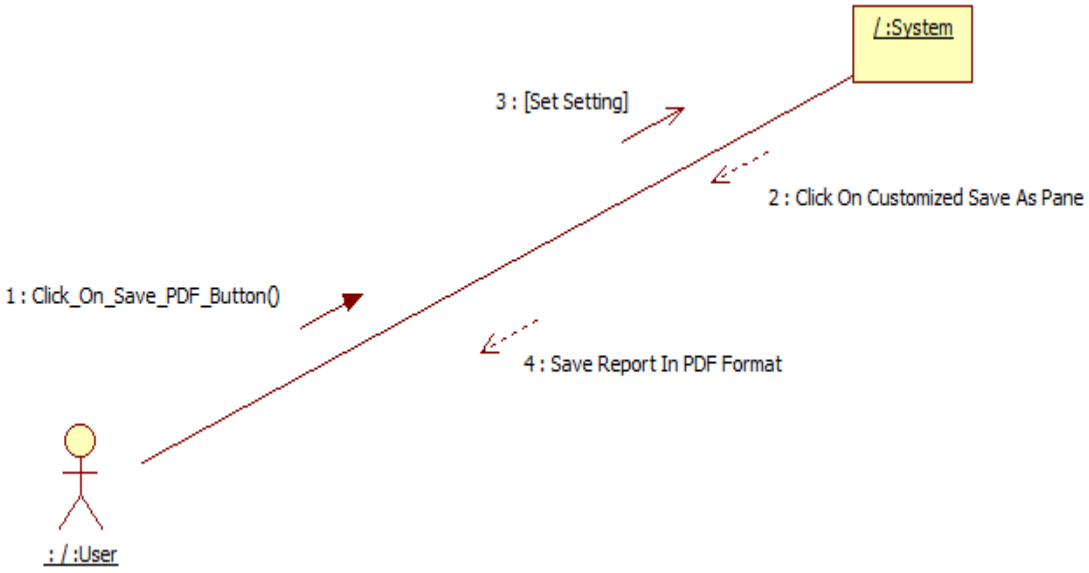


Figure 3-50: CD 15: Save report

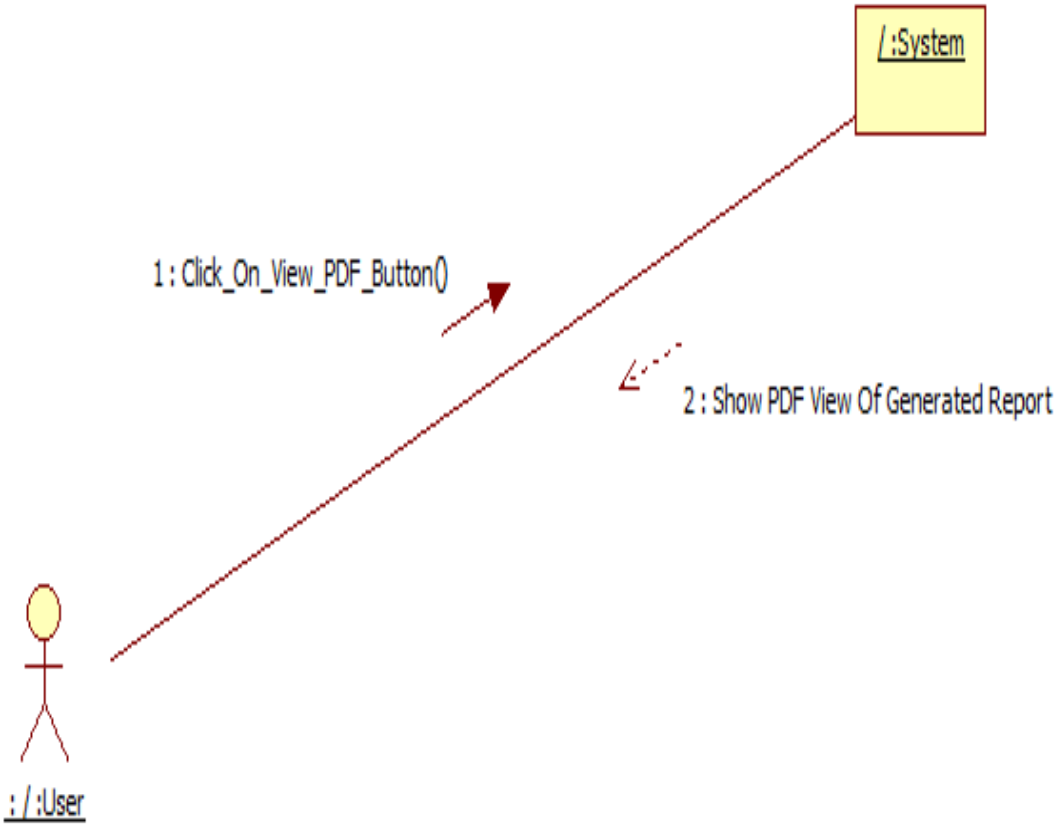


Figure 3-51: CD 16: View report

3.2.4 Class Diagram

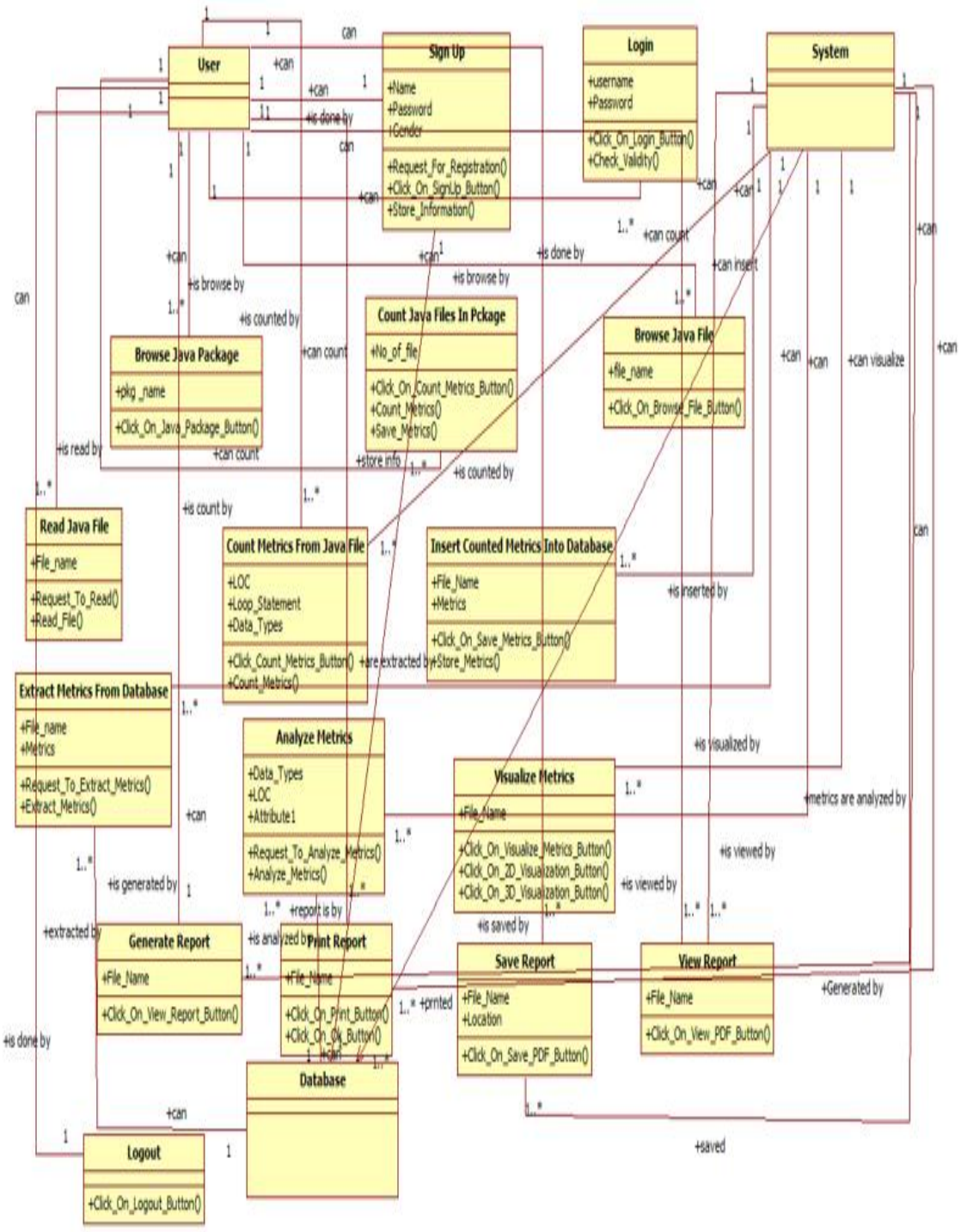


Figure 3-52: CD: SOURCE CODE METRICS VISUALIZER OF JAVA

ChAPTER4

TESTING

CHAPTER-4

TESTING

4.1 Testing

Testing is the process of evaluating a system or its components with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements.

4.2 Test Cases

Following are the test cases of our application. Following are the testers who perform test cases: M1 =Uzair Zia, M2 = Farzeen Shahzad, M3 = Ghulam Abbas.

4.2.1 Sign UP

Sign Up	
Test Engineer:	M1
Test Case ID:	TC1
Related UC:	UC1
Date:	01-07-2020
Purpose:	Create account
Pre-Req:	Application must run
Test Data:	Username: admin, Password: admin, Email: uzairzia143@gmail.com,
Steps:	• Visit Sign up Page • Enter Username, Password, Email • Click on ok Button Navigate to Login Page
Status:	Pass

Sign Up	
Test Engineer:	M1
Test Case ID:	TC2
Related UC:	UC1
Date:	01-07-2020
Purpose:	Create account
Pre-Req:	Application must run
Test Data:	Username: admin , Password: admin, Email:
	ghulam.abbas4065@gmail.com
Steps:	• Visit Sign up Page • Enter Username, Password, Email • Click on ok Button Invalid Username, Password, Email
Status:	Fail

4.2.2 Log in

Log in	
Test Engineer:	M1
Test Case ID:	TC3
Related UC:	UC3
Date:	01-07-2020
Purpose:	Login to application
Pre-Req:	Must have Username and Password
Test Data:	Username: farzeen_18 , Password: farzeen123

Steps:	• Visit Login Page • Enter Username and Password • Click on ok Button Navigate to Main Page
Status:	Pass

Log in	
Test Engineer:	M1
Test Case ID:	TC4
Related UC:	UC3
Date:	01-07-2020
Purpose:	Login to application
Pre-Req:	Must have Username and Password
Test Data:	Username: Uzair_zia, Password: uzair12
Steps:	• Visit Login Page • Enter Username and Password • Click on ok Button Invalid Username and Password
Status:	Fail

4.2.3 Log out

Log out	
Test Engineer:	M1
Test Case ID:	TC5
Related UC:	UC4
Date:	01-07-2020
Purpose:	Exit from application
Pre-Req:	Must run application
Test Data:	Logout option from File menu

Steps:	• Click on File option from menu • Select Log out option Exit from Application
Status:	Pass

Log out	
Test Engineer:	M1
Test Case ID:	TC6
Related UC:	UC4
Date:	01-07-2020
Purpose:	Exit from application
Pre-Req:	Must run application
Test Data:	Logout option from File menu
Steps:	• Click on File option from menu • Select Log out option
	Not exit from Application
Status:	Fail

4.2.4 Browse Java Package

Browse Java Package	
Test Engineer:	M1
Test Case ID:	TC7
Related UC:	UC5
Date:	01-07-2020
Purpose:	Browse Java Package
Pre-Req:	Must login
Test Data:	Browse Java Package button

Steps:	• Click on File option from menu • Select Browse java Package option • New frame will appear • Click on Browse Package Button • Choice Java Package • Click on Open button Package Path shown in JTextArea
Status:	Pass

Browse Java Package	
Test Engineer:	M1
Test Case ID:	TC8
Related UC:	UC5
Date:	01-07-2020
Purpose:	Browse Java Package
Pre-Req:	Must login
Test Data:	Browse Java Package button
Steps:	• Click on File option from menu
	• Select Browse java Package option • New frame will appear • Click on Browse Package Button • Choice Java Package • Click on Open button Please Choice Java Package
Status:	Fail

4.2.5 Count Java Files from Package

Count Java files from package	
Test Engineer:	M1
Test Case ID:	TC9
Related UC:	UC6
Date:	01-07-2020
Purpose:	count java files from package
Pre-Req:	Browse java package
Test Data:	Count metrics button
Steps:	• Click on Count Metrics Button • No of Files and Name of Files Shown from Package
Status:	Pass

Count Java files from package	
Test Engineer:	M1
Test Case ID:	TC10
Related UC:	UC6
Date:	01-07-2020
Purpose:	count java files from package
Pre-Req:	Browse java package
Test Data:	Count metrics button
Steps:	• Click on Count Metrics Button
	• First Choice Java Package
Status:	Fail

4.2.6 Browse Java File

Browse Java file	
Test Engineer:	M2
Test Case ID:	TC11
Related UC:	UC7
Date:	01-07-2020
Purpose:	Browse Java file
Pre-Req:	Must login
Test Data:	Browse java file button
Steps:	• Click on File option from menu • Select Browse java File option • New frame will appear • Click on Browse File Button • Choice Java File • Click on Open button File Path is shown into JTextArea

Status:	Pass

Browse Java file	
Test Engineer:	M2
Test Case ID:	TC12
Related UC:	UC7
Date:	01-07-2020
Purpose:	Browse Java file
Pre-Req:	Must login
Test Data:	Browse java file button
Steps:	• Click on File option from menu • Select Browse java Package option • New frame will appear • Click on Browse Package Button • Choice Java Package • Click on Open button Please Choice Java File
Status:	Fail

4.2.7 Count Metrics

Count Metrics	
Test Engineer:	M2
Test Case ID:	TC13
Related UC:	UC8
Date:	01-07-2020
Purpose:	Count metrics from java file
Pre-Req:	Browse Java File
Test Data:	Count metrics button

Steps:	• Click on Count Metrics Button File Metrics shown in related fields
Status:	Pass

Count Metrics	
Test Engineer:	M2
Test Case ID:	TC14
Related UC:	UC8
Date:	01-07-2020
Purpose:	Count metrics from java file
Pre-Req:	Browse Java File
Test Data:	Count metrics button
Steps:	• Click on Count Metrics Button First Choice Java File
Status:	Fail

4.2.8 Insert metrics into database

Insert metrics into database	
Test Engineer:	M2
Test Case ID:	TC15
Related UC:	UC9
Date:	01-07-2020
Purpose:	Insert metrics into database
Pre-Req:	Count metrics from java file
Test Data:	Button
Steps:	• Click on Button • Establish connection with database • get values from related field Insert counted metrics into database

Status:	Pass

Insert metrics into database	
Test Engineer:	M2
Test Case ID:	TC16
Related UC:	UC9
Date:	01-07-2020
Purpose:	Insert metrics into database
Pre-Req:	Count metrics from java file
Test Data:	Button
Steps:	• Click on Button • Establish Connection with database • Get values from related fields Connection not establish or Fail to access file data
Status:	Fail

4.2.9 Extract metrics from database

Extract metrics from database	
Test Engineer:	M2
Test Case ID:	TC17
Related UC:	UC10
Date:	01-07-2020
Purpose:	Extract metrics from database
Pre-Req:	Insert metrics into database
Test Data:	Button
Steps:	• Click on Button • Establish connection Extract inserted metrics from database
Status:	Pass

Extract metrics from database	
Test Engineer:	M2
Test Case ID:	TC18
Related UC:	UC10
Date:	01-07-2020
Purpose:	Extract metrics from database
Pre-Req:	Insert metrics into database
Test Data:	Button
Steps:	• Click on Button • Establish connection Connection not establish or Fail to access file data
Status:	Fail

4.2.10 Analyze metrics

Analyze metrics	
Test Engineer:	M2
Test Case ID:	TC19
Related UC:	UC11
Date:	01-07-2020
Purpose:	Analyze metrics
Pre-Req:	Extract metrics from database
Test Data:	Button
Steps:	• Click on Button Analyze file extracted metrics
Status:	Pass

Analyze metrics	
Test Engineer:	M2
Test Case ID:	TC20
Related UC:	UC11
Date:	01-07-2020
Purpose:	Analyze metrics
Pre-Req:	Extract metrics from database
Test Data:	Button
Steps:	• Click on Button File access error
Status:	Fail

4.2.11 Visualize Metrics

Visualize Metrics	
Test Engineer:	M3
Test Case ID:	TC21
Related UC:	UC12
Date:	01-07-2020
Purpose:	Visualize Metrics
Pre-Req:	Analyze Metrics
Test Data:	JCombobox for Selection, 2D or 3D Visualization button
Steps:	• Select visualization technique from JCombobox • Click on 2D or 3D Button 2D or 3D graphs or charts are shown
Status:	Pass

Visualize Metrics	
Test Engineer:	M3
Test Case ID:	TC22
Related UC:	UC12
Date:	01-07-2020
Purpose:	Visualize Metrics
Pre-Req:	Analyze Metrics
Test Data:	JCombobox for Selection, 2D or 3D Visualization button
Steps:	• Select visualization technique from JCombobox • Click on 2D or 3D Button Fail to access file data
Status:	Fail

4.2.12 Generate Report

Generate Report	
Test Engineer:	M3
Test Case ID:	TC23
Related UC:	UC13
Date:	01-07-2020
Purpose:	Generate Report
Pre-Req:	Analyze metrics
Test Data:	Open interface, Table for show Information
Steps:	• Extend on View option on menu • Select Report option Report of stored metrics will be shown
Status:	Pass

Generate Report	
Test Engineer:	M3

Test Case ID:	TC24
Related UC:	UC13
Date:	01-07-2020
Purpose:	Generate Report
Pre-Req:	Analyze metrics
Test Data:	Open interface, Table for show Information
Steps:	• Extend on View option on menu • Select Report option Fail to access file data or database is not connected
Status:	Fail

4.2.13 Print Report

Print Report	
Test Engineer:	M3
Test Case ID:	TC25
Related UC:	UC14
Date:	01-07-2020
Purpose:	Print Report
Pre-Req:	Generate Report
Test Data:	Print button
Steps:	• Click on Print button • Show Print pane • Set required settings • Click ok button Report is printed
Status:	Pass
Print Report	
Test Engineer:	M3

Test Case ID:	TC26
Related UC:	UC14
Date:	01-07-2020
Purpose:	Print Report
Pre-Req:	Generate Report
Test Data:	Print button
Steps:	• Click on Print button • Show Print pane • Set required settings • Click ok button Printer port is not connected
Status:	Fail

4.2.14 Save Report

Save Report	
Test Engineer:	M3
Test Case ID:	TC27
Related UC:	UC15
Date:	01-07-2020
Purpose:	Save Report
Pre-Req:	Generate Report
Test Data:	Save PDF button
Steps:	• Click on Save PDF button • Show customizes save as pane • Set required settings • Click ok button Report is saved in pdf format
Status:	Pass

Save Report	
Test Engineer:	M3
Test Case ID:	TC28
Related UC:	UC15
Date:	01-07-2020
Purpose:	Save Report
Pre-Req:	Generate Report
Test Data:	Save PDF button
Steps:	• Click on Save button • Show customizes save as pane • Set required settings • Click ok button Save report appropriate path is assigned to report
Status:	Fail

4.2.15 View Report

View Report	
Test Engineer:	M3
Test Case ID:	TC28
Related UC:	UC16
Date:	01-07-2020
Purpose:	View Report
Pre-Req:	Generate Report
Test Data:	View button
Steps:	• Click on View button Report is viewed in format
Status:	Pass

View Report	
Test Engineer:	M3
Test Case ID:	TC29
Related UC:	UC16
Date:	01-07-2020
Purpose:	View Report
Pre-Req:	Generate Report
Test Data:	View button
Steps:	• Click on View button Make sure report is generated.
Status:	Fail

CHAPTER 5
CONCLUSION & FUTURE WORK

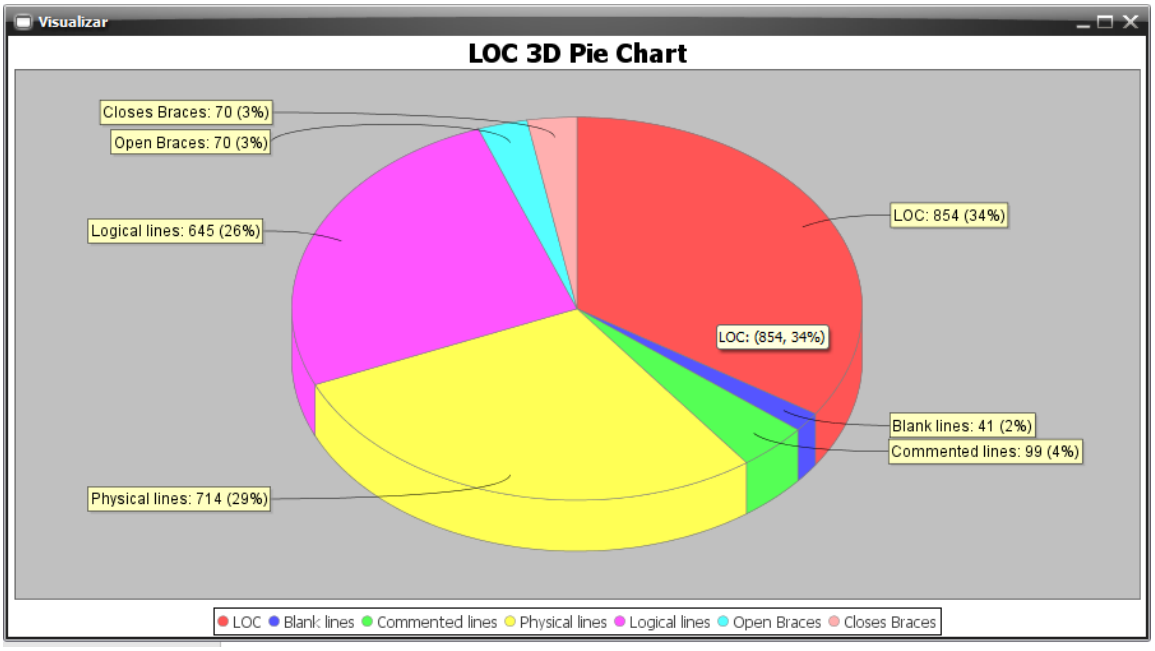
CHAPTER-5

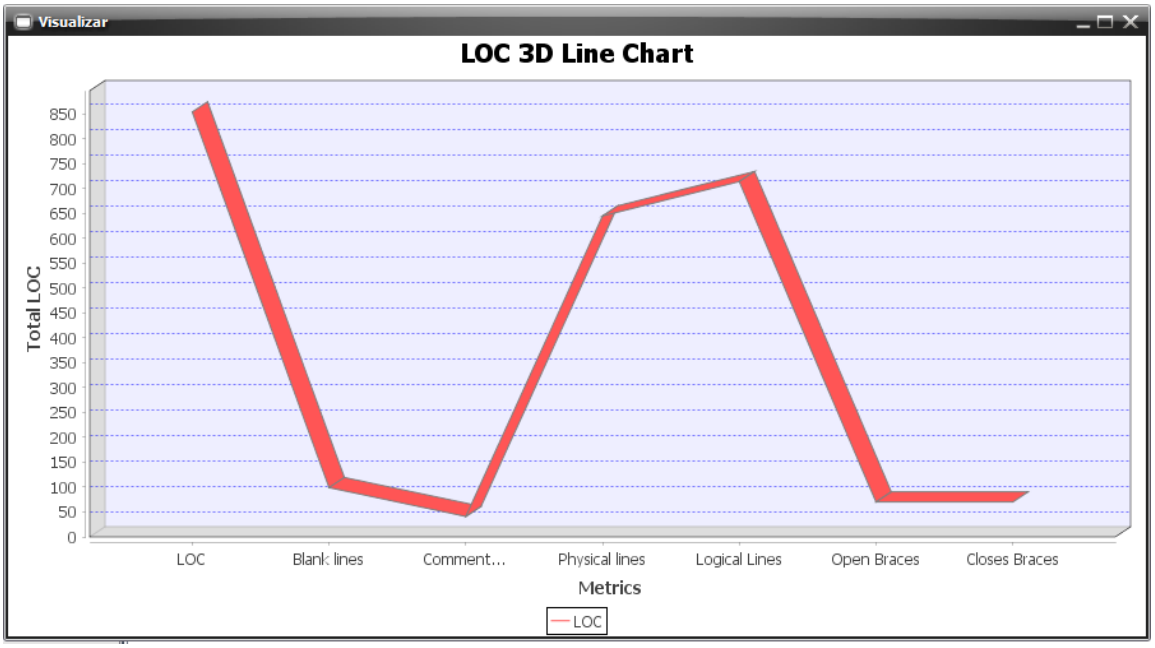
CONCLUSION & FUTURE WORK

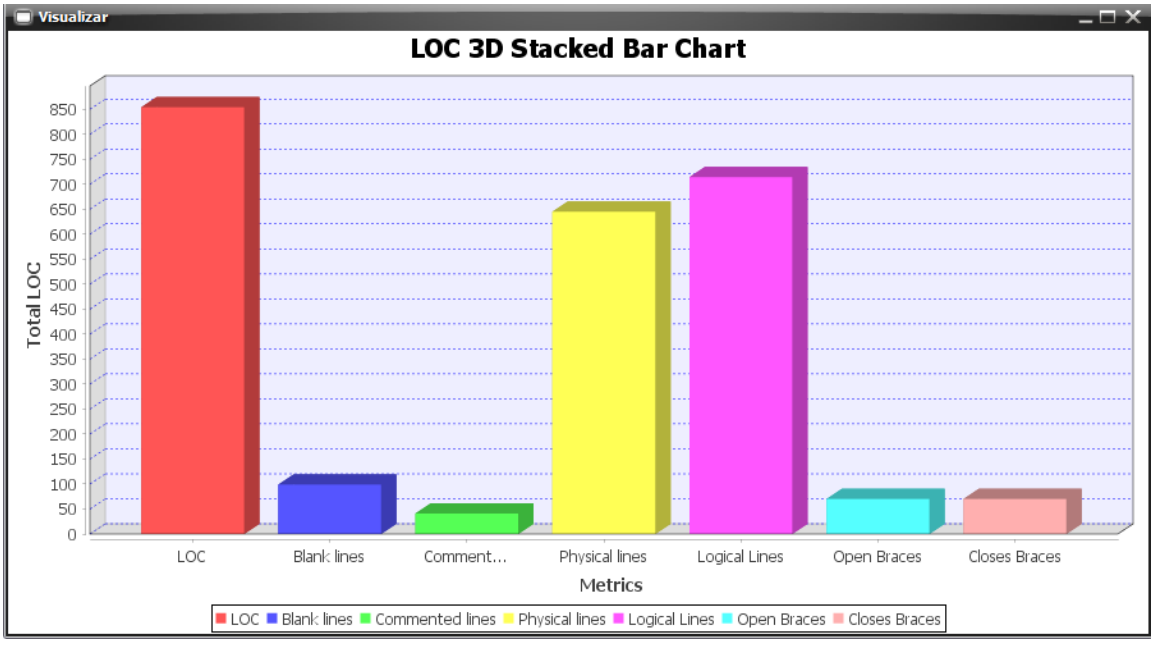
5.1 Successes Guarantee

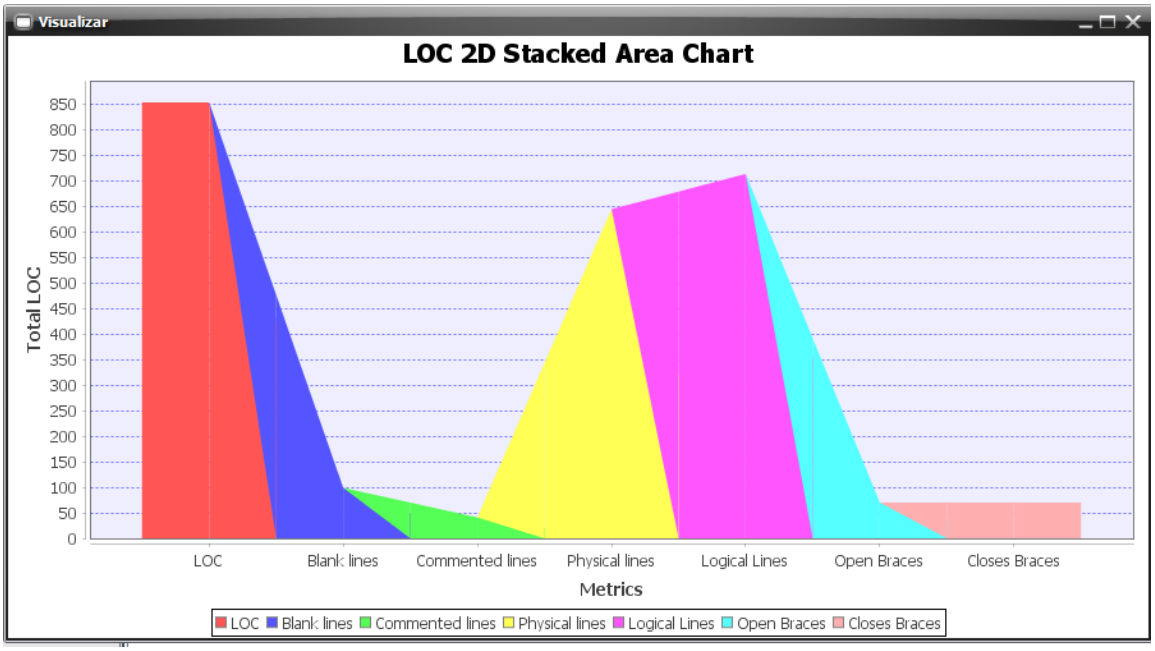
Source Code Metrics Visualizer of Java successfully registered user if he/she have no account and also provide login facility. It efficiently browser Java package as well as individual Java files, count metrics from package as well as from individual file such as total line of code, blank lines, empty lines, logical lines, physical lines, data types, inheritance of classes, cyclomatic complexity, name of methods, no of arguments in each method, coupled classes, no of method, loop statements and conditional statements. Then it efficiently stores all counted metrics into database. After insertion extract metrics from database and visualize them in 2 dimensional and 3dimention visualization such as 3D Line chart, 3D Pie chart, 3D Stacked Area chart, 2D Dual Axis chart and 2D Waterfall chart. It also efficaciously generates, print, view and save report in PDF format in customized view.

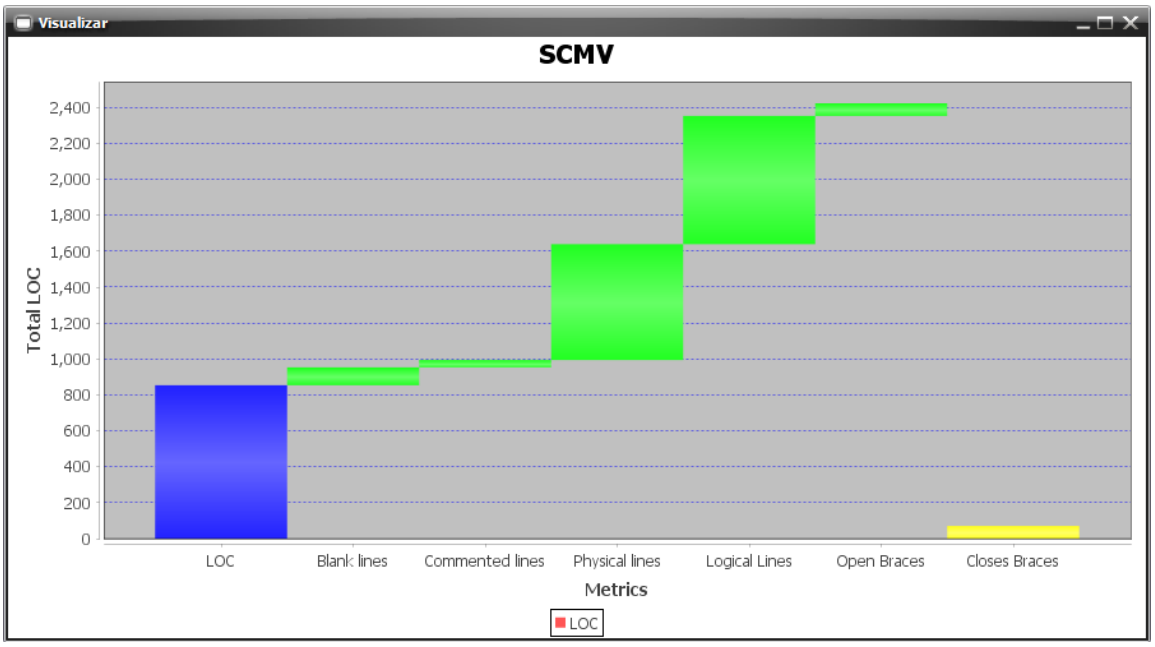
5.1.1 Result of SCMV




Visualizar
LOC 3D Line Chart
Total LOC
850
800
750
700
650
600
550
500
450
400
350
300
250
200
150
100
50
0
LOC
Blank lines
Comment...
Physical lines
Logical Lines
Open Braces
Closes Braces
Metrics
LOC


Visualizar
LOC 3D Stacked Bar Chart
Total LOC
850
800
750
700
650
600
550
500
450
400
350
300
250
200
150
100
50
0
LOC
Blank lines
Comment...
Physical lines
Logical Lines
Open Braces
Closes Braces
Metrics
LOC
Blank lines
Commented lines
Physical lines
Logical Lines
Open Braces
Closes Braces


Visualizar
LOC 2D Stacked Area Chart
Total LOC
850
800
750
700
650
600
550
500
450
400
350
300
250
200
150
100
50
0
LOC
Blank lines
Commented lines
Physical lines
Logical Lines
Open Braces
Closes Braces
Metrics
LOC Blank lines Commented lines Physical lines Logical Lines Open Braces Closes Braces


Visualizar
SCMV
Total LOC
2,400
2,200
2,000
1,800
1,600
1,400
1,200
1,000
800
600
400
200
0
LOC
Blank lines
Commented lines
Physical lines
Logical Lines
Open Braces
Closes Braces
Metrics
LOC

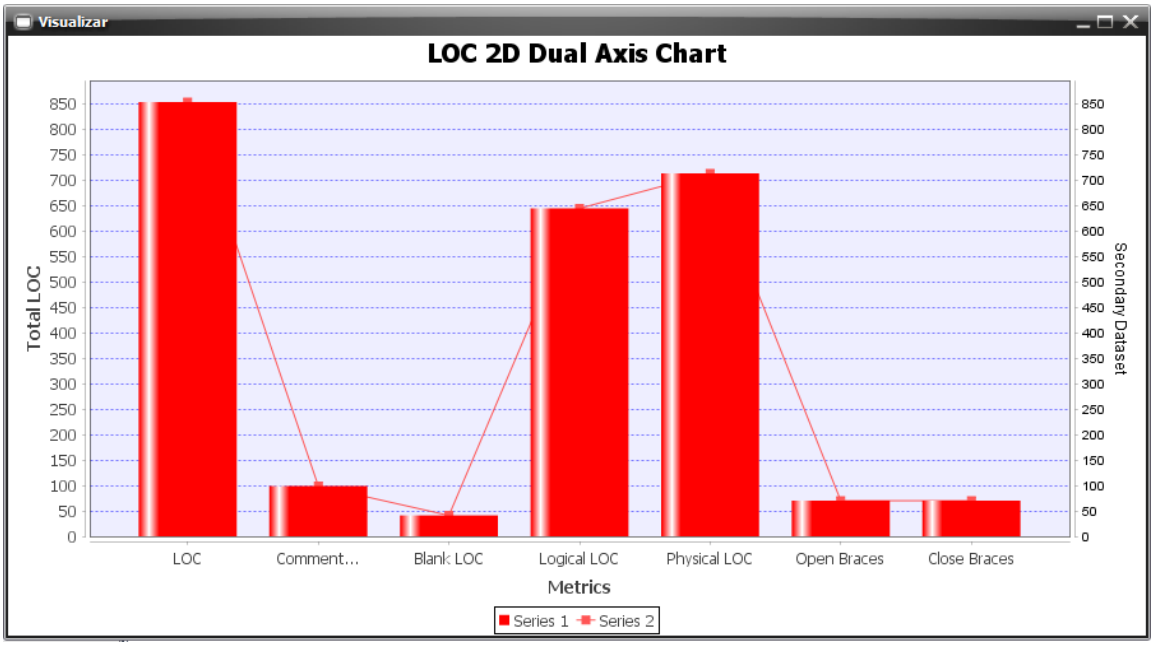


Figure 5-1: Various 2D and 3D visualization techniques

5.2 Future Recommendations

However, the world gain success development in software technology increases. In order to face these situations, we upgrade this system. Futures recommendations are as follows:

- **For Developers**

  It can help programmers as well as non-programmers to understand source code effectively by using different 2D and 3D visualization techniques.

- **For Refactoring teams**

  In this application we identified the bad small in source code and send to re-factoring teams to redesign the software that will redesign source code in such a way that it does not alter the external behavior of the code yet improve its internal structure.

- **Use for reusability**

  In this application we identified the bad small in source code and send to software engineering teams to solve the errors, redesign and reuse their

code to improve their product performance and maintain the product budget.

- **Add more source code metrics**

In future, add more source code metrics such as Cohesion for the improvement of application functionality.

- **Add more visualization techniques**

In future, add more 2D and 3D visualization techniques so that end users can understand their source code well.

- **Enhance Application**

SCMV is application developed by desktop users, in future developed web-based application. We used Java language for developing application. SCMV support. Java extension files or JAVA package for identifying different datasets and visualized them. In future, you have to use various languages such as C#, python and C++ for improve the development of application and visualization of numerous datasets.

Chapter 6

USER MANUAL

USER MANUAL

6.1 Login

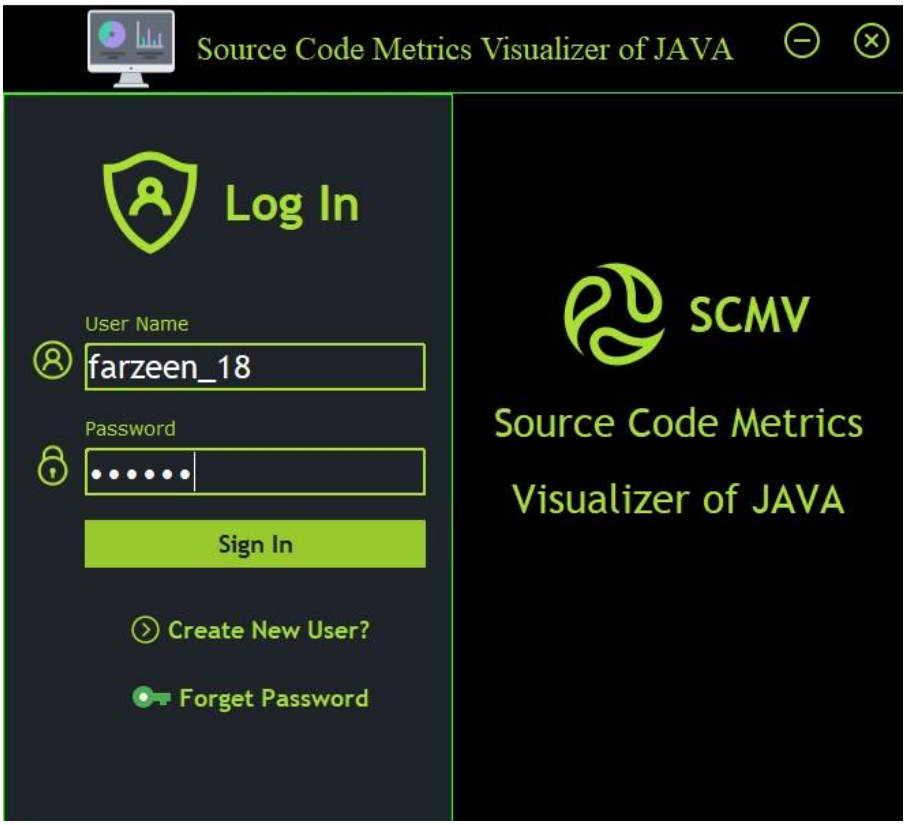


Figure 6-1 : Login

Step 1: Enter user name.

Step 2: Enter password.

Step 3: Click on 'Sign In' button.

If the user is authentic then system will login successfully and **Home Screen** will be popup. If the user is not authentic then system will display a message stating that 'Invalid Username or Password'.

6.2 Sign Up

Figure 6-2: Sign up

Step 1: Enter Username.
Step 2: Enter email address.
Step 3: Enter password.
Step 4: Enter confirm password.
Step 5: Click on 'Sign up' button.

When the user clicked on Sign up button after filling all fields system will first check user name in database. If the user name is already registered then system will ask the user to change the user name. If the user name is not registered then system will send a One-Time-Password on the email address which the user has provide on Sign up field. After sending a One-Time-Password system will ask the user to enter One-Time-Password to get successfully register. After successfully register the user can login to system.

6.3 Main Page

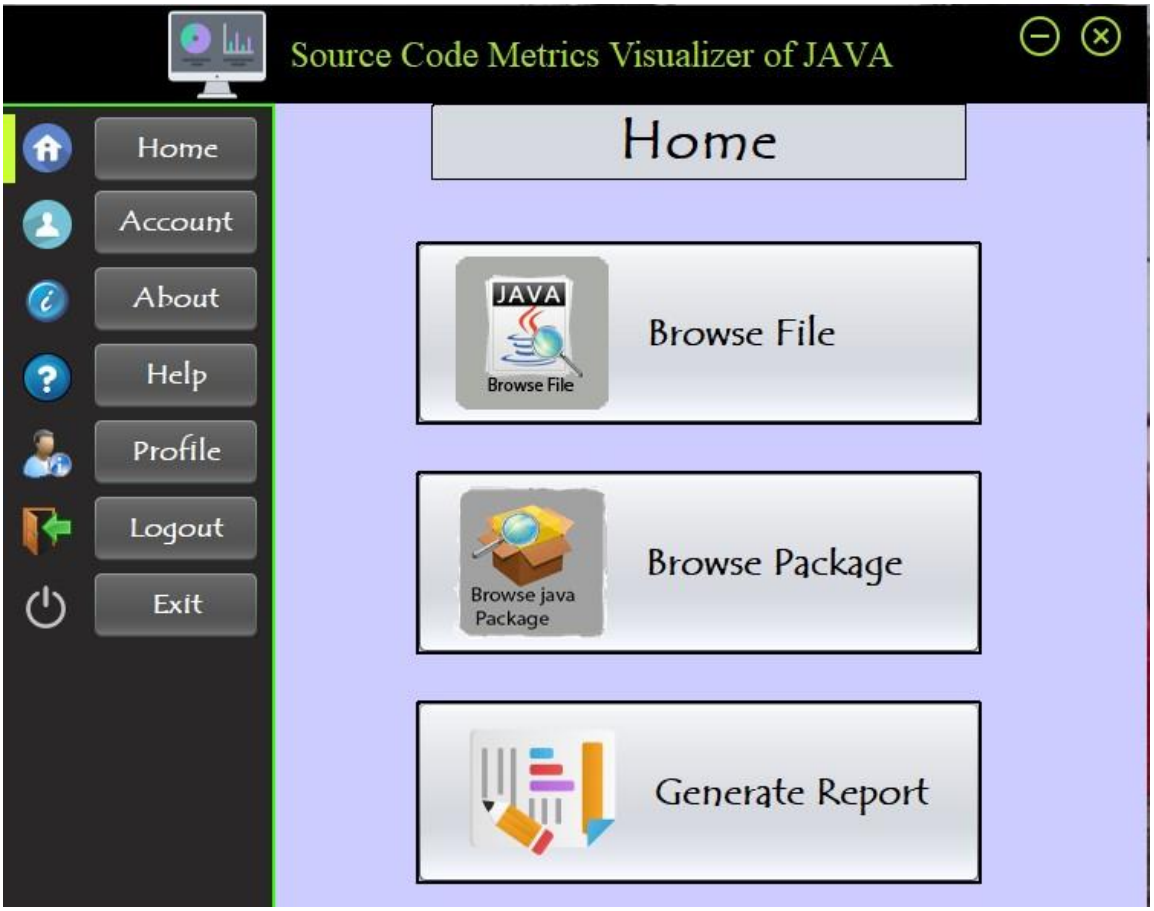


Figure 6-3: Main Page

If the user wants to browse a Java file then the user will click on ‘Browse File’ button. And if the user want to browse a Java package then the user will click on ‘Browse Package’ button. If the user has already browsed a file or package then the user can click on ‘Generate Report’ button to generate the report.

6.4 Browse Java Package

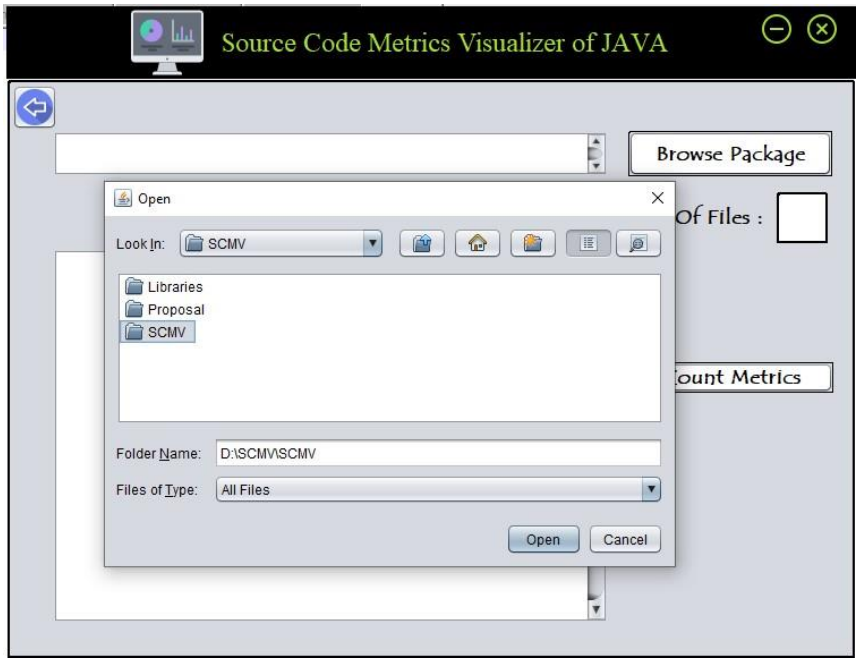


Figure 6-4: Browse java package

Step 1: Click on 'Browse Package' button.

Step 2: Select Package.

Step 3: Click 'Open' button.

After clicking on open button system will open each folder of selected package and finds files having .java extension. After finding all Java files system displays all files name on screen.

6.5 Count Metrics from Java Package

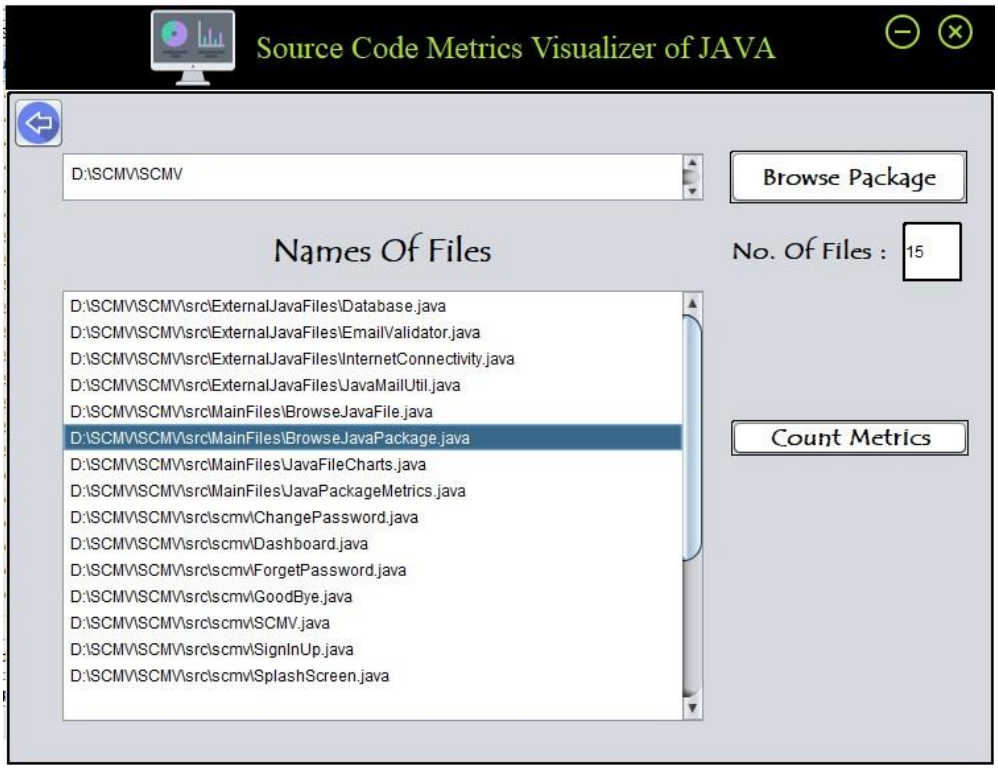


Figure 6-5: Count Metrics from Java package

Step 1: Select file name.

Step 2: Click on ‘Count Metrics’ button.

When the user select a file and click on ‘Count Metrics’ button system reads selected file line by line and extracts all metrics from selected file. After extracting all metrics system stores all metrics in database and opens another screen and displays all those extracted metrics.

6.6 Browse Individual Java File

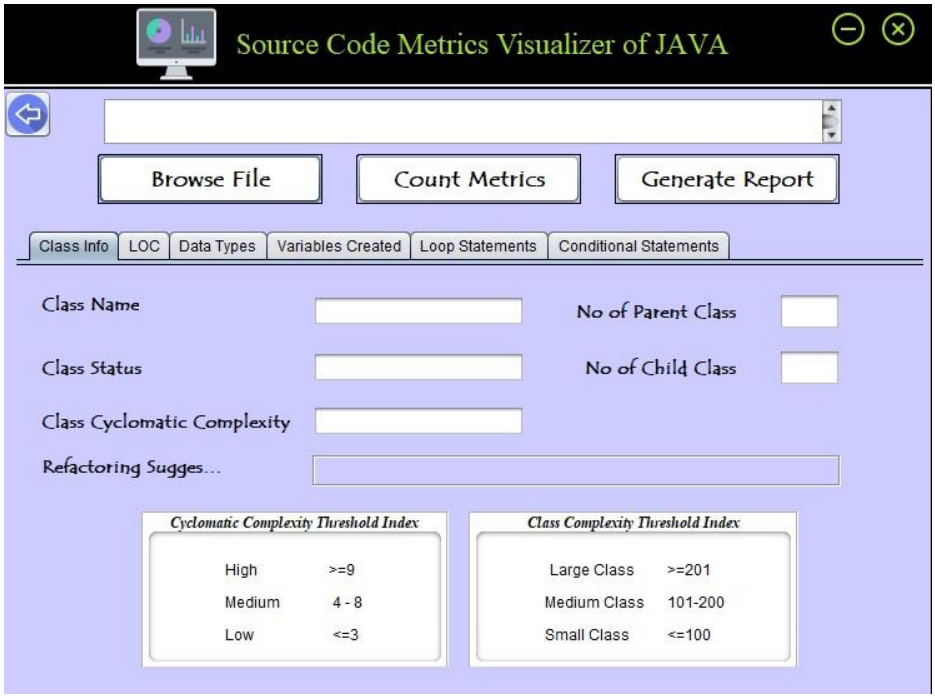


Figure 6-6: Browse Java file

Step 1: Click on ‘Browse File’ button.

Step 2: Select java file.

Step 3: Click on ‘Open’ button.

After clicking on ‘Browse File’ button system will open a file chooser dialogue box from where the user can select a Java file and will click on open button.

6.7 Count Metrics from File

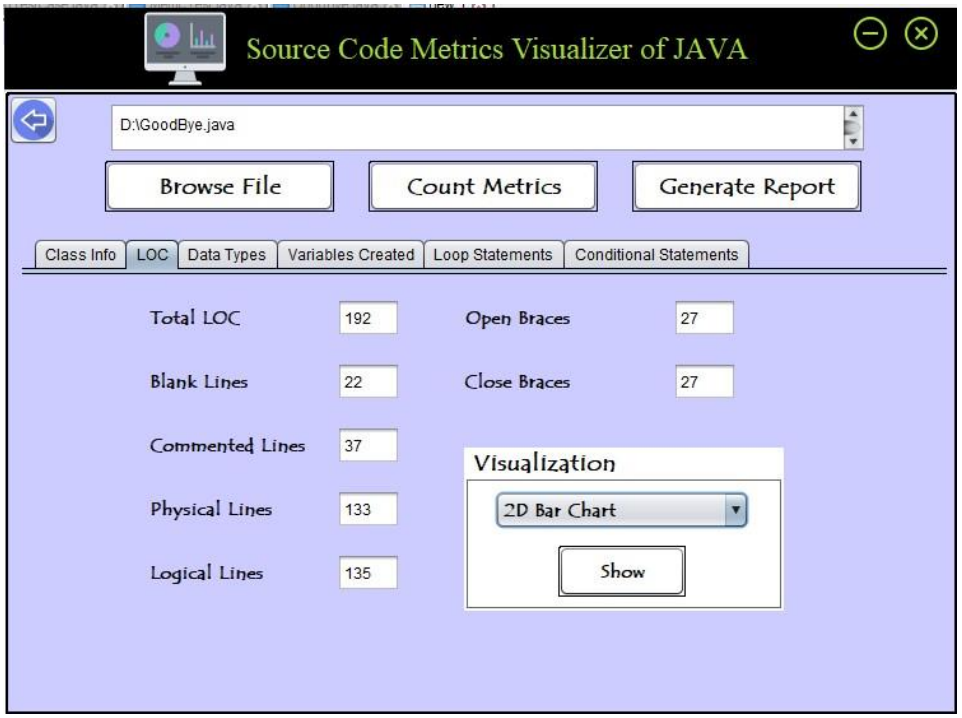


Figure 6-7: Count Metrics from file

Step1: Click on 'Count Metrics' button.

When the user click on 'Count Metrics' button system reads selected file line by line and extracts all metrics from selected file. After extracting all metrics system stores all metrics in database and then displays on screen.

6.8 Visualize Metrics

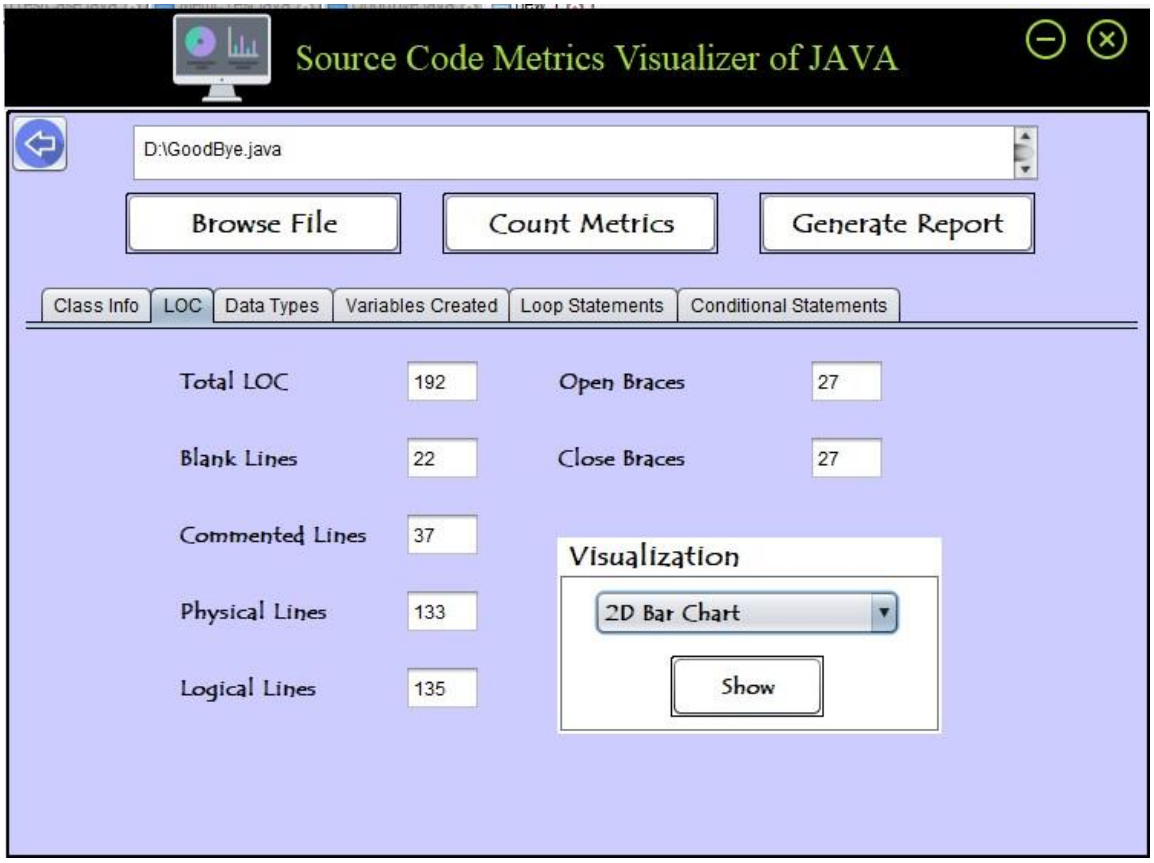


Figure 6-8: Visualize metrics

Step 1: Click on dropdown arrow.

Step 2: Select 2D or 3D visualization chart.

Step 3: Click on 'Show' button.

When the user click on 'Show' button after selected a desired chart system will extract all stored metrics from database and visualize it in selected chart.

6.9 Generate Report

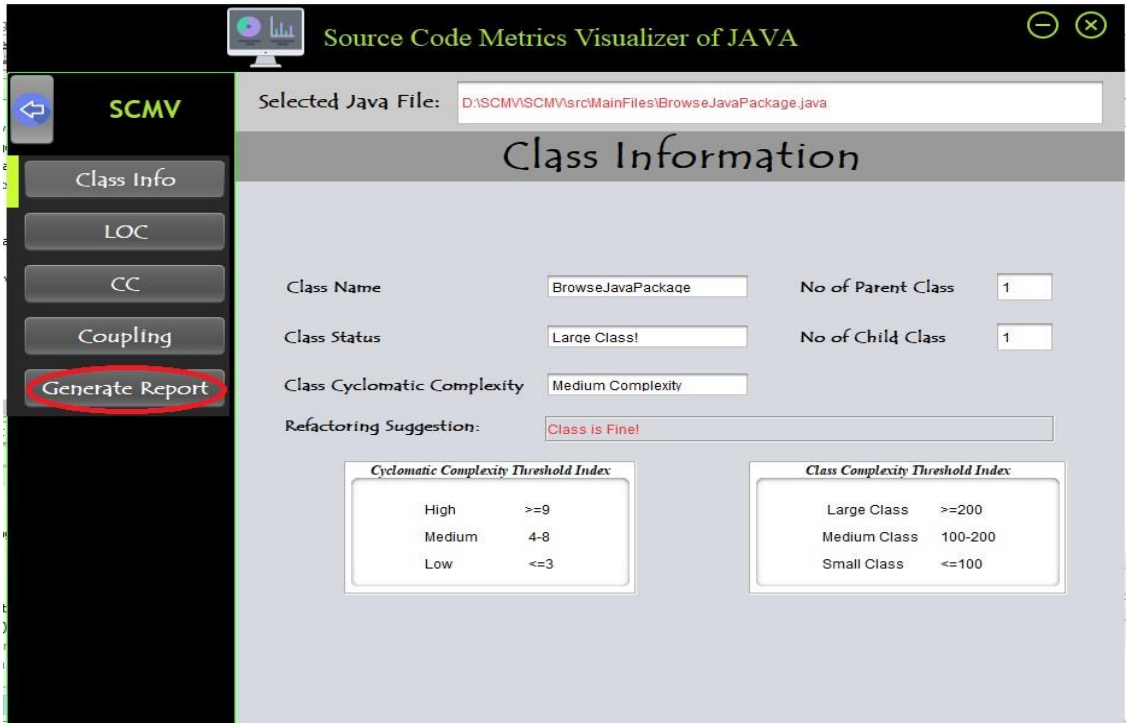


Figure 6-9: Report metrics

Step 1: Click on 'Generate Report' Button to generate and view report.

When the user click on generate report button system extracts all stored metrics from database and creates a report and opens another window to view the report.

6.10 Print Report

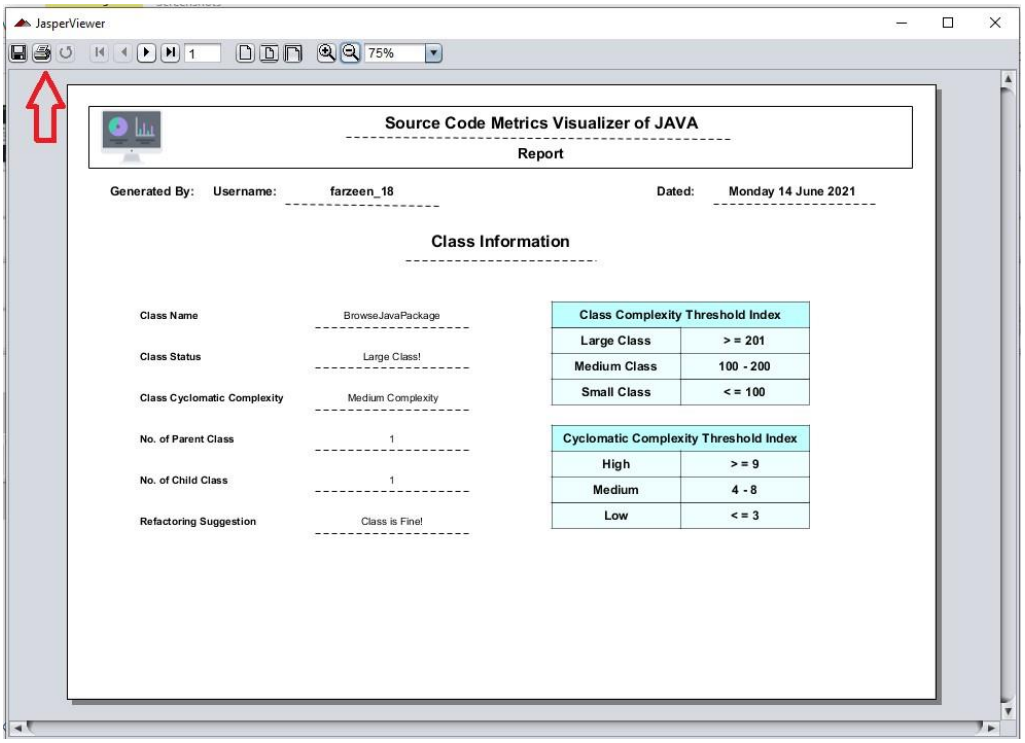


Figure 6-10: Print report

Step 1: Click on ‘print’ button.

Step 2: Set appropriate setting.

Step 3: Click on ‘print’ from print pane.

User can print report from print pane and can customize print setting.

6.11 Save Report

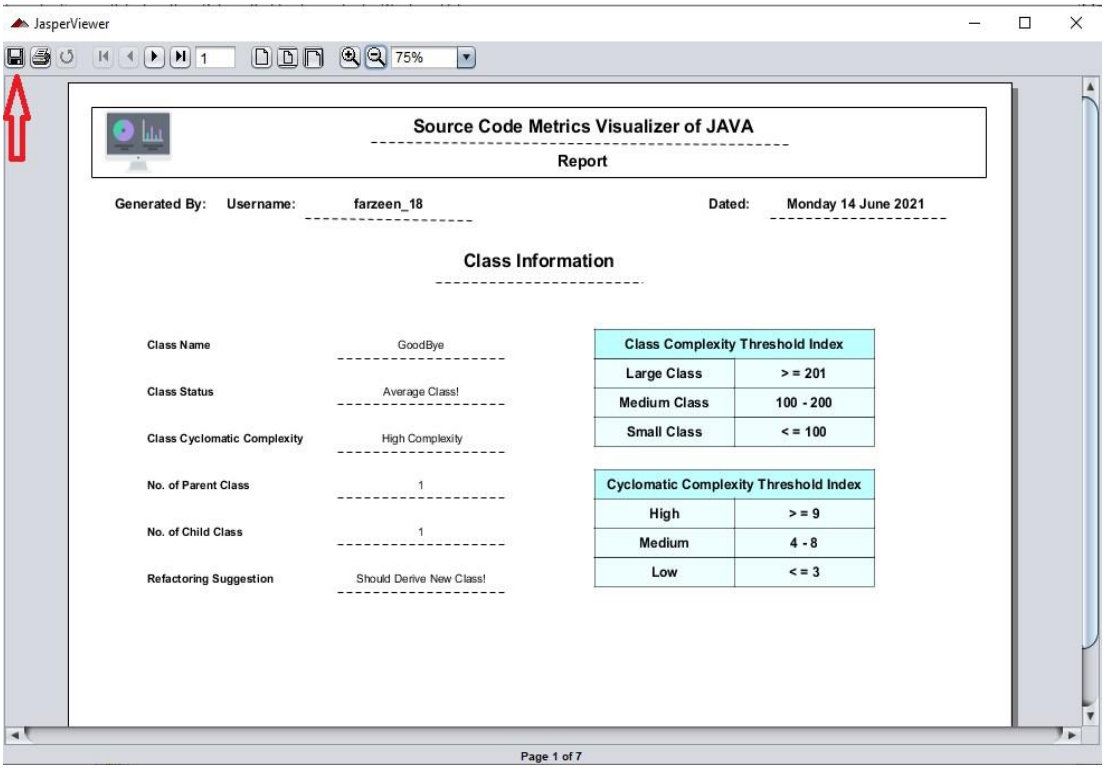


Figure 6-11: Save report in PDF

Step 1: Click on ‘Save’ button.

Step 2: Write file name and give file path.

Step 3: Click on ‘Save’ button.

User can save report by clicking on save report icon. When the user click on save report button system will ask user to enter file name, select desire destination, and desire extension of report type.

6.12 View Report

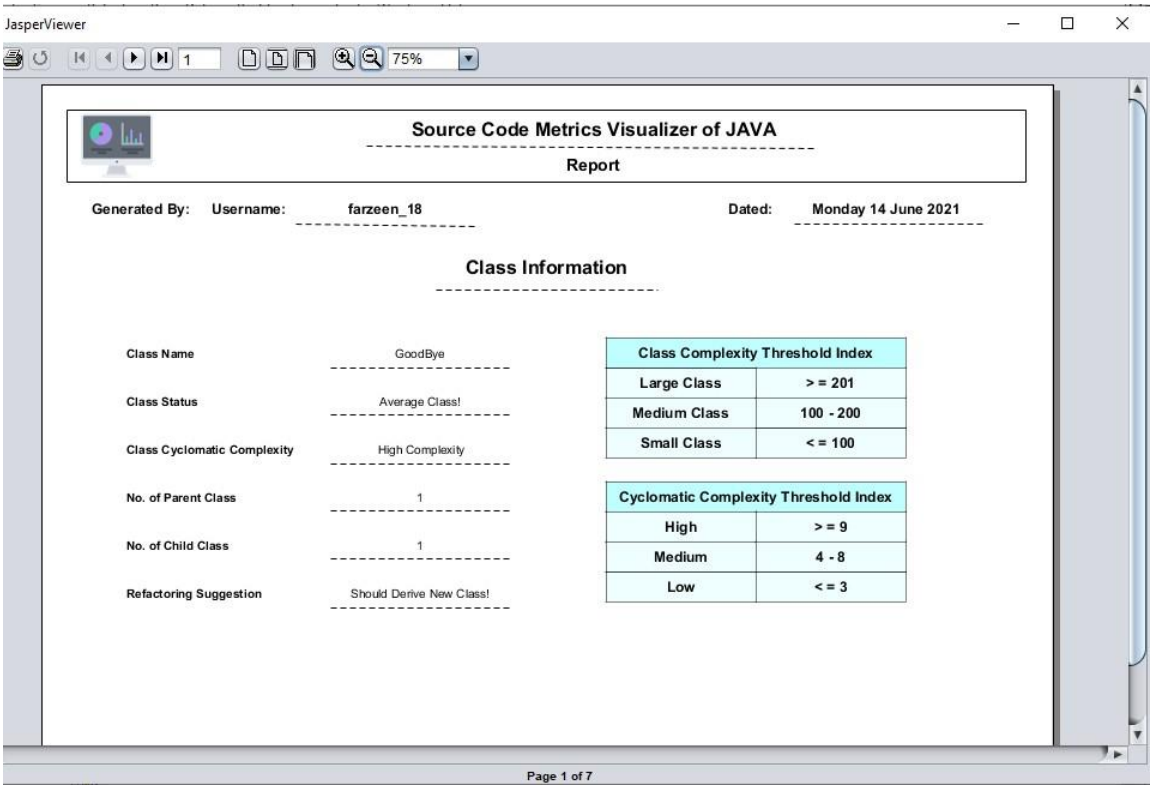


Figure 6-12: View report in PDF

Step 1: Click on arrow on top to 'View Report".

User can also view report by clicking on top arrow indicating next page icon. User can also zoom in or out the report.

6.13 Help Window

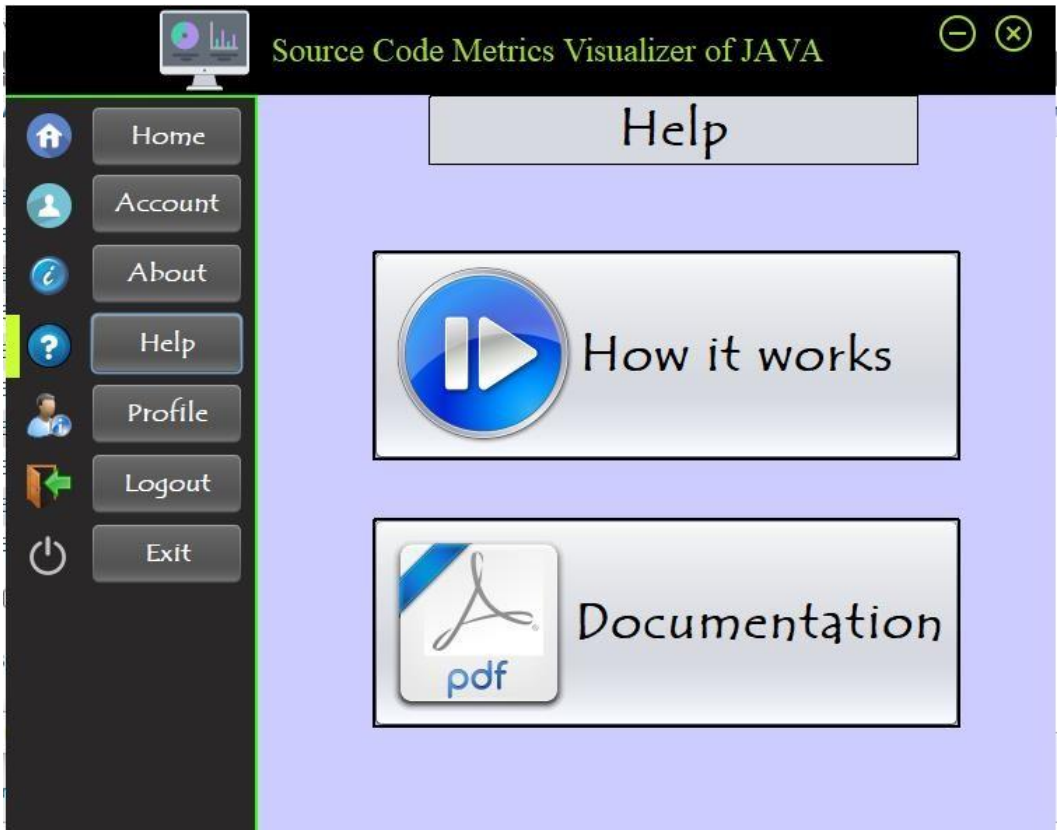


Figure 6-13: Help window

Step 1: Click on 'Help' on menu.

When the user click on Help menu button. New panel will be open. From help menu the user can watch a short video tutorial of using the tool by clicking on 'How it works' button. And user can also view the documentation report of tool.

6.14 Log Out

Figure 6-14: Log out

Step 1: Click on 'Logout' button.

When the user click on logout button system logout the current login user and displays the login screen to the user.

REFERENCES

REFERENCES

[1] “program Verification System”, our first practical research in the sphere of metrics calculation. Available at: http://www.viva 64.com

[2] about tech, [Online]. Available at: https://java.about.com/od/s/g/sourcecode.htm

[3] Source code, [Online]. Available at: https://www.thoughtco.com/source-code-definition-958200

[4] Technology Conversation [Online]. Available at: https://technologyconversations.com/2013/11/11/black_box_vs_white_box_test

[5] Boukari Souley and Baba Bata, “A Class Coupling Analyzer for Java Programs”, Mathematical Sciences Programme Abubakar Tafawa Balewa University (ATBU), Bauchi, Nigeria bsouley2001@yahoo.com +2348069667696, +2347034568741.

[6] https://www.google.com/search?q=cohesion+and+coupling+in+ooad&tbm=isch&source=iv&ictx=1&fir=oLKEPcgIHo3zCM%253A%252Cx1x-T7-3#imgrc=cY1iepTXMJu2uM&imgdii=oLKEPcgIHo3zCM

[7] Haneen Abu Alfeilat.Visualizing class information, [Online]. Available at: https://arxiv.org/ftp/arxiv/papers/1602/1602.07838.pdf

[8] Software Metrics Visualization, [Online]. Available at: https://blog.hubspot.com/marketing/great/data/visualization-examples

[9] Jaekwon Lee, Woosung Jung+, “Automated Metrics Visualizations for Analyzing Source Code Representation,” Department of Computer Engineering Chungbuk National University
Cheongju, Republic of Korea
{exatoa, wsjung}@cbnu.ac.kr.

[10] Dinesh Thakur, “Software metrics in software engineering”, [Online]. Available at: http://ecomputernotes.com/software-engineering/software-metrics

[11] Zhao Kaidi, “Data visualization,” School of Computing., National University of Singapore, Iscp0075@nus.edu.sg zhaokaidi@hotmail.com,Matrix Number:HT00-6177E (Document Version 1.0).

[12] Danny Hubertus Rosalia Holten, “Visualization of Graphs and Trees for Software Analysis,” Published by Technische Universiteit Eindhoven, ISBN: 978-90-386-1882-1.

[13] Cc.gatech.edu. Software Visualization research at GVU. 2016 [Online]. Available at: http://www.cc.gatech.edu/gvu/ii/softvis/

[14] CyVis-Software Complexity Visualizer", Cyvis.sourceforge.net, 2016 [Online]. Available at: http://cyvis.sourceforge.net/index.html. [Accessed: 29-Mar-2016].

[15] Difference Between 2D and 3D [Online]. Available at: http://www.differencebetween.net/language/difference-between-2d-and-3d/

[16] Muzammil Khan and Sarwar Shah Khan. (November, 2011). Data and Information Visualization Methods and Interactive Mechanisms (Volume 34). [Online]. Available at: https://pdfs.semanticsscholar.org/4ff1/f2fff62e899f4b9f507b2eb4bb297b7febc2.pdf

[17] 3D Pie Chart the Data Visualization Technique [Online]. Available at: https://unitedwebsoft.in/blog/create-dynamic-pie-chart-php-google-charts/

[18] 3D Line Chart Visualization Technique [Online]. Available at: https://ncss-wpengine.netdna-ssl.com/wp-content/themes/ncss/pdf/Procedures/NCSS/3D_Line_Charts.pdf

[19] 3D Line Chart Visualization Technique [Online]. Available at: https://www.infragistics.com/help/winforms/chart-about-3d-line-charts

[20] Carolyn Talmadge and Jonathon Gale. (January 28, 2016). Introduction to Data Visualization Techniques. [Online]. Available at: https://sites.tufts.edu/gis/files/2016/02/Introduction_to_Data_Visualization.pdf

[21] 3D Stacked Bar Chart [Online]. Available at: https://businessq-software.com/2017/02/21/stacked-bar-chart-definition-and-examples-businessq/

[22] 2D Stacked Area Chart the Data Visualization Technique [Online]. Available at: https://help.highbond.com/acl/11/index.jsp?topic=%2Fcom.acl.user_guide.help%2Fanalysis_app%2Fc_stacked_area_charts.html

[23] 2D Dual Axis Chart [Online]. Available at: https://infogram.com/create/dual-axis-chart

[24] Waterfall Chart [Online] Available at: https://en.wikipedia.org/wiki/Waterfall_chart

[25] Lucidechart, "Unified Modeling Language Basics", Available at: https://www.lucidchart.com/pages/what-is-UML-unified-modeling-language.

[26] TutorialsPoints, "UML 2.0", [online]. Available at: https://www.tutorialspoint.com/uml/uml_2_overview.htm