



**SCHOOL OF INFORMATION TECHNOLOGY AND ENGINEERING**

**SWE-2020 SOFTWARE METRICS**

**D1**

**J-COMPONENT**

**FINAL REVIEW**

**METRICS BASED ASSESSMENT OF MOBILE GAME  
APPLICATION: CONNECT 3**

*Under the guidance of*

**Prof Uma Maheswari G.**

**Submitted by**

**Sabrina Manickam 19MIS0137**

**Laasya Yarlagaadda 19MIS0138**

**Sirigiri Sri Sai Sharanya 19MIS0266**

# **Metrics-based Assessment of a Mobile Gaming Application: Connect 3**

## **Abstract**

The software game creation industry is experiencing intense rivalry as a result of the quick expansion in the development of mobile games to meet varied consumer needs. Usability is a crucial component of mobile game products that has a big impact on player happiness. The necessity for earlier corrections that might assist reduce development costs and time led to the usability assessment of the mobile game product, which is a game prototype created throughout the production. In this study, a method for defining an metrics for a mobile game prototype that was created during the design process is proposed.

This document focuses on the game Connect 3 which helps in making strategic decisions and various metrics that can be applied to make the connect 3 application work better. The paper contains the introduction which provides a brief introduction about the connect 3 game, software metrics, and various software metrics that have to be applied to measure the various required features of the application such as efficiency, reliability, etc. Then the Goal-question-Metric approach is applied which helps us to define goals, trace those goals to the data that are intended to define those goals with the help of questions, and finally provides us a framework for interpreting the data with respect to the stated goals. After GQM, the demonstration of the application is presented.

## **1. Introduction**

According to a Portio Research [1] estimate, the number of people using mobile applications worldwide is expected to rise by 29.8% every year, reaching 4.4 billion people in 2017.

According to statistics [2], there are already more than 2.1 billion mobile applications available. This is an increase from the previous year [3], and it is anticipated that this number will continue to rise. More crucially, the category for games continues to receive the most downloads. The aforementioned data shows how the mobile application market, which includes mobile games, is expanding. As more operators enter the market, there is increased rivalry. It is crucial that programmes, particularly games, are developed to satiate user needs. The usability of the selected game is one of the elements influencing users' choices. Because of this, the development of mobile games should place a greater emphasis on quality and meeting customer expectations.

The usability quality of games, however, differs significantly from that of general software [5] due to the fact that games have specific characteristics that are utilized to gauge the usability quality, such as content, storyline, challenge, and mechanisms. All of these factors influence user happiness and enjoyment. The final stage of the development process [6], or the finalized product, is typically when mobile games or applications are evaluated for quality [8][9][10]. If a clear indicator exists that the finished game product does not adhere to user criteria, a rectification process must be initiated. Typically, this takes more work than early-stage repairs, which can usually be made. As a result, this study suggests a strategy for

evaluating the usability of mobile games during the preproduction stage. The game prototype is what comes out of the design phase of the development process for mobile games [7] [11]. The game used in this project is Connect 3, which is a two-player Android mobile application. This game is based on the famous age-old game tic-tac-toe. The players in this game have to connect 3 coins of their suite in a straight line horizontal, vertical or diagonal. The game does not support the use of artificial intelligence and is solely based on two players playing the game. Due to the nature of the game, the game is constructed on a 2D layout which feels similar to the original pen and paper game. The game Connect 3 is one of the most commonly known games. This game does not allow one to win all the time and a significant proportion of games played result in a draw. Thus, the best player can hope is to not lose the game.

## 2. Literature Survey

S.No	Title of the article (authors involved)	Name of the journal, year, issue number, page no	Contribution in that work	Reason for choosing this article for your reference
1.	Towards a Reliable Identification of Deficient Code with a Combination of Software Metrics  (Tina Beranič, Vili Podgorelec and Marjan Heričko)	Applied Sciences, 2018, Vol.8, 10, 1902,	The method for identifying defective source code in the research is innovative and is based on applying a majority function to a mix of software metrics. The evaluation of programme entities takes into account the statistical distributions of the values of the software metrics, which are used to evaluate a collection of software metrics and their corresponding threshold values	This paper came up with an innovative approach to identify a deficient code considering different software metrics. This was very useful for us as we wanted to focus on a better source code for our Connect 3 game. It helped us to calculate metrics like LOC, Avg Complexity, Max Nesting and other important metrics.

			<p>obtained from benchmark data. The suggested strategy was put into practise and validated on Java, C++, and C# applications. Software developers and architects with many years of expertise evaluated the calibre of the software classes as part of the validation of the suggested strategy. The number of possibly flawed object-oriented programme entities was shown to be decreased when a mix of software metrics was used as the standard for identifying defective source code. The outcomes demonstrate the accuracy of the quality ratings arrived at using the suggested identification method, and more crucially, they prove the absence of false positive entities.</p>	
2.	A Consolidated and Comparative	IJARCCE, 2019, Vol. 8, 4, 160-167	The integration of eight separate object-oriented,	In this paper the authors have compared 8

	<p>Analysis of Software Metrics Tools for Systems Performance Evaluation: A Survey</p> <p>(Er. Ashish Gupta , Dr. Rajat Kumar Mehta , Er. Mahendra Rai)</p>		<p>free metric tools is attempted. The metrics values for software projects were measured in a study using the same set of standard metrics. The results, which demonstrated the disparities in results from different tools for the same metrics, have already been explored. Measurements demonstrate the tool dependence of the metrics values for the same software system and metrics. We compared the CCCC, CKJM, Dependency Finder, Semmle, VizAnalyzer, JHawk Tool, Analyst 4j, and OOMeter tools in this study.</p>	<p>different software metric tools to measure the values and they found out that values are tools dependent. We have also tried to use some of the mentioned tools like JHawk to calculate different metrics like Maintainability, Halstead Metrics which helped us in comparison with other tools.</p>
3.	<p>Quality Measurement for Serious Games</p> <p>(Suryapranata, Louis Khrisna Putera and Soewito, Benfano and Kusuma, Gede Putra and Gaol, Ford Lumban and Warnars, Harco Leslie</p>	<p>International Conference on Applied Computer and Communication Technologies (ComCom), 2017, 1-4</p>	<p>The study suggests a brand-new game assessment metric to assess game quality features, particularly to assess serious game materials that can inspire players. The output of this study is a metric table with the test types and factors</p>	<p>This paper suggested a new assessment metric to increase the quality of a code and this helped us in our Connect 3 game. Better and faster performance was achieved with the proposed metric.</p>

	Hendric Spits)		that can be measured.	
4.	<p>A Size Estimation Model for Board-Based Desktop Games</p> <p>(NOSHEEN SABAHAAT , ALI AFZAL MALIK , AND FAROOQUE AZAM</p>	IEEE Access, 2017, Vol.5, 4980-4990	<p>When developing new projects, estimating the amount of the software involved is crucial. In this research, a parametric model for determining the size of board-based PC games is developed, validated, and put to use. On a data set made up of more than 60 open source board games that were gathered from several open source repositories, this model was created using forward stepwise multiple linear regression. This model is evaluated using a number of prediction accuracy measures, including MMRE, PRED(x), MdMRE, and others, and is validated using K-fold cross-validation. Exercises in model validation and assessment produce encouraging</p>	<p>The parametric model was useful in determining the size of the game. This helped in developing a better version of Connect 3. Multiple linear regression was helpful in measuring accuracy and other important software metrics.</p>

			outcomes. By showing a worked-out game size estimation example and then some size-related what-if analysis, this model's usefulness is shown.	
5.	<p>What Concerns Game Developers? A Study on Game Development Processes, Sustainability and Metrics</p> <p>(Kasurinen, J., Palacin-Silva, M., &amp; Vanhala, E.)</p>	2017 IEEE/ACM 8th Workshop on Emerging Trends in Software Metrics (WETSoM), 2017, 15-21	<p>One may argue that the process models for game creation and software development are similar in a number of ways, such as the requirement to design, build, and test software functionalities. Although the software engineering (SE) models successfully support the development of software, they usually fail when applied practically to the development of video games. They conducted a poll with gaming organizations to better understand this issue, as well as a number of other factors such as green IT and business strategies. According to their observations of</p>	<p>The survey of this paper helped us to understand what the concerns of a game developer are and how to overcome them. Since our project was also a game development, the paper was relatable and it solved many of our concerns. The paper discussed different metrics like code complexity, LOC, usability, maintainability, software cost, etc, which are very important for a game's development. It also discussed the business concerns that a game developing company can face. So all these discussions have helped in developing a good game.</p>

			SE practices, the game industry mainly uses agile process models or does not use any at all. Their main business concerns were mobile development, digital marketing, hiring people with specific skills, and maintaining innovation in their business. Their less important concerns were eco-impact factors, software reusability, and financing. Based on our findings, we identified areas that may benefit from additional research to enhance commercial, environmental, and game industry practices.	
6.	An investigation of strength analysis metrics for game-playing programs: A case study in Chinese dark chess  (Chu-Hsuan Hsueh , I-Chen Wu, Tsan-sheng Hsu and Jr-Chang Chen)	ICGA Journal, 2, 77--104	In this study, a simplified version of the non-deterministic game Chinese dark chess (CDC), known as 24 CDC, is utilized as a testbed to assess the strengths of game-playing software. The win percentages vs the ideal opponent are then used to	This paper came up with a version of the Chinese Dark Chess that was used as a testing platform for the strength of the gaming software. This is highly relevant to our project as we have tried to assess the quality of the game, Connect 3. The selected paper has



			<p>establish absolute strength. Results from experiments demonstrate that win rates versus predetermined baseline programmes are a desirable metric because they are strongly connected with absolute strengths.</p>	<p>come up with metrics such as mean squared errors and prediction rates that can be applied to our project as well.</p>
7.	<p>Evaluating Software Metrics of Gaming Applications using Code Counter Tool for C and C++ (CCCC)</p>	<p>International conference of Electronics, Communication, and Aerospace Technology (ICECA), Vol. 2, 180-184</p>	<p>This article uses the metric device code counter for C and C++ to show the various programming measures of gaming apps. This provides three C++ apps for keeping track of money, each with its own set of classes. This tool evaluates each application's measurements and provides limit estimates for each measurement.</p>	<p>This paper was chosen because it proposed three tools that were used to measure gaming software. In our project, we also propose a tool-based approach for measuring the Connect 3 game. This paper brought forth the various methods used by the tools to evaluate the game in addition to enumerating the ways to analyze the results produced by the tools.</p>
8.	<p>Perfecting A Video Game with Game Metrics  (Junaidi, Andy Julianto, Nizirwan Anwar, Safrizal, Harco</p>	<p>TELKOMNIKA (Telecommunication Computing Electronics and Control) 16(3), 1324-1331</p>	<p>Video game makers have been working toward creating the ideal state for their products, but in order to get there, a number of standards and techniques must</p>	<p>This paper provides an in-depth analysis of the to tools used to measure gaming application. This paper helped as understand how tools can be used</p>

	Leslie Hendric Spits Warners , Kiyota Hashimoto)		be used. We determined that this is a must-have tool or method to be implanted in the development of a game because it will boost your standards quickly and be able to tell you about your own progress in development. These standards and methods are unique in that they are both verifiable and quantifiable, to clarify their action and end goal.	to understand that game better.
9.	Software Metrics Classification for Agile Scrum Process: A Literature Review	2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI) (pp. 174-179). IEEE.	The essential function of software measuring metrics in the agile software development life cycle are discussed in this paper. We examine software metrics in the context of scrum using 13 carefully chosen academic works. Software metrics are organized according to measurement goals and scrum event objectives.	34 software metrics that are dispersed among the four scrum activities of sprint planning, daily scrum meetings, sprint review, and sprint retrospective are covered.
10.	Using software metrics for	Journal of Software: Evolution and Process, 33, pages={e2303}	When an attacker successfully uses the insecure code	This paper talks about the vulnerabilities

	<p>predicting vulnerable classes and methods in Java projects: A machine learning approach (Kazi Zakia Sultana, Vaibhav Anu, Tai-Yin Chong)</p>		<p>to exploit the vulnerability and expose it, the software vulnerability becomes dangerous for the software. The entire piece of software may be at risk due to a single code flaw. As a result, development teams have the vital and at the same time difficult duty of preserving software security throughout the product life cycle. Additionally, this may enable susceptible code to develop throughout subsequent versions.</p>	<p>present that go undetected if software metrics are not used correctly. This paper tells us about the importance of software metrics.</p>
11.	<p>Comparison of Learning Software Architecture by Developing Social Applications versus Games on the Android Platform</p> <p>Authors: Bian Wu and Alf Inge Wang</p>	<p>Hindawi Publishing Corporation International Journal of Computer Games Technology Volume 2012, Article ID 494232, 10 pages doi:10.1155/2012/494232</p>	<p>This paper describes the how traditional software domain and game development is different and how the learning process varies and, the relevant tools used, the phases involved, the methods used in each approach and how the results vary depending on the factors that were mentioned.</p>	<p>To develop a game application, it is necessary to know the significant methods, tools and technologies used for game development. For developing a game, it is important the techniques needed to correctly develop as well as what are the methodologies that are not.</p>

12.	<p>Revisiting prior empirical findings for mobile apps: An empirical case study on the 15 most popular open-source Android apps.</p> <p>Authors: Syer, M. D., Nagappan, M., Hassan, A. E., &amp; Adams, B</p>	<p>In Proceedings of the 2013 Conference of the Center for Advanced Studies on Collaborative Research, pp. 283-297. 2013, November.</p>	<p>This paper was an exploratory study which compares mobile apps to desktop or server-side application where most of the software engineering research is performed today. It also describes about how effectively the existing “best practices” apply to mobile application development. The hardware limitation and diversity present in mobile devices, distribution channels, source code size and time taken to execute, defect fix time, platform usage, threats to validity were compared and the results were presented.</p>	<p>This paper was chosen to know the best practices of mobile development as compared to traditional software development processes. The various limitation and advantages present in mobile development. How the software source code for mobile development projects can be measured, by using different size metrics such as size of the code base, size of the development team, platform usage, the defect fix times, distribution of defects and many others.</p>
13.	<p>Identifying Development-Metrics for Use in a Gamified Mobile Web Application to Support Software Development</p> <p>Authors: Tim Wenzel, Thomas Franke, Rainer Wasinger, and</p>	<p>MOBILITY 2020 : The Tenth International Conference on Mobile Services, Resources, and Users</p>	<p>This paper tries to provide insights about the gamification of software development and to also to identify the relevant metrics that lead to successful software development and to filter out these metrics that can be suitable for</p>	<p>Gamification of software helps us to provide better experience for the users. It enhances the usability by including a playful way to do tasks. For any software or game development process this can be an important feature to be included and the</p>

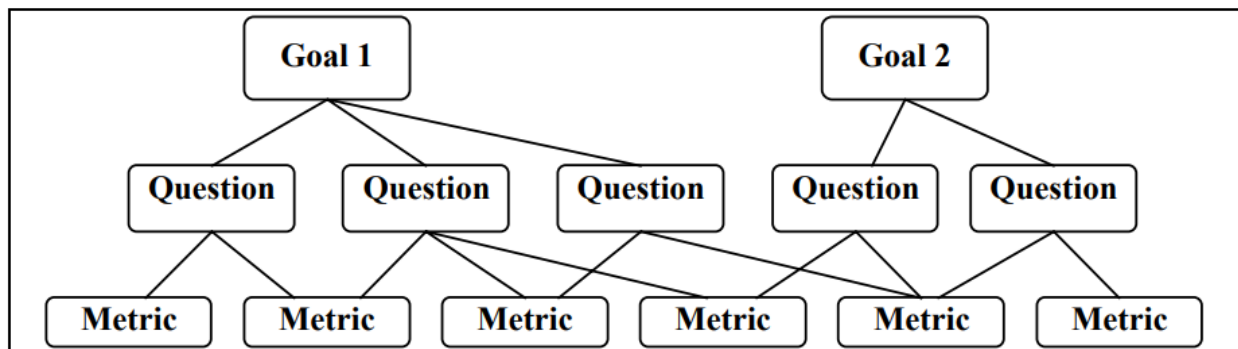
	Michael Korner		gamification. Gamification is a process by which a service can be improved in a playful manner to provide better user experience. The identification of evaluation criteria used in software development are presented in the form of quantitative metrics and qualitative metrics and finally the software design issues and technologies were mentioned.	paper presented various way of how gamification of software can be done.
14.	Game Industry Metrics Terminology and Analytics Case Study  Author: Game Industry Metrics Terminology and Analytics Case Study	Game Analytics. Springer, London, 2013. 53-71.	This paper mentions about how online games can be benefitted by tracking information about the game and user behaviour. How the data can be gathered and converted into metrics, and in turn how to use the metrics obtained to compare the performance of one game to another. It also provides a briefing about the commonly used metrics and using these metrics how the developers can	Using the metrics defined in the paper such as the daily active users, monthly active users, the user conversion rate, the peak concurrent users, user acquisition cost etc, a good understanding about the interface of the game, the current state of the game, and also about the business analytics of the game can be obtained which helps the developers as well as the organization to improve and

			improve their game. A case study was considered and various aspects such as player population, monetization, advertising were considered to provide general terms and measures for metrics and analytics.	enhance their current methodologies that are being followed to obtain better results.
15.	<p>Game development software engineering process life cycle: a systematic review</p> <p>Authors: Saiqa Aleem, Luiz Fernando Capretz, and Faheem Ahmed</p>	<p>Journal of Software Engineering Research and Development</p> <p>Aleem et al. Journal of Software Engineering Research and Development (2016) 4:6</p>	<p>The research aims at finding the underline techniques that help game development to achieve maintainability, flexibility, lower effort and cost, and better design. It emphasizes about the necessity of a development method as a systemized procedure to achieve the goal of producing a working product withing budget and schedule. And also, a development methodology was proposed which consists of three steps Planning, Conducting and Documenting with detailed steps. The phase wise game</p>	<p>This article would help us to manage the development of the application within schedule and also provides insight on different development strategies, methodologies, phases involved in the game development life cycle and the distinction it would have with the traditional software development methodologies.</p>

			development life cycle was proposed.	
--	--	--	--------------------------------------	--

### 3. Goal-Question-Metric

The Goal Question Metric (GQM) approach is predicated on the idea that in order for an organization to measure with purpose, it must first define the goals for itself and its projects, then link those goals to the data that are meant to operationalize those goals, and finally provide a framework for interpreting the data in light of the stated goals. In order for this information needs to be quantified wherever possible and for the quantified information to be assessed to determine whether or not the goals are accomplished, it is crucial to identify, at the very least in basic terms, what the organization's informational needs are.[12]



#### Goal Question Metric Table

Goal	Quality Attributes	Questions	Metric
		Is the goal of the game easy to understand?	Goal's Description Understandability

<b>Content</b>	Understandability	Does player understand role of the character?	Total number of clear characters/Total number of characters
	Learnability	-----	-----
	Operability	Can the player customize their character?	Character Customisability
		Is the main storyline and goal of the game conflicting?	Number of changes in rules/Total number of rules specified
	Attractiveness	How is the character design?	Character Design Attractive Interaction
		How about game story attractiveness?	Event Attractive Interaction



	Compliance	-----	-----
Device	Understandability	Is game menu comprehensive?	Total number of menu items
		Is player able to understand purpose of the buttons?	Evident Buttons
	Learnability	Does the game provide help facility of how to input?	Completeness of Input Help Facility
	Operability	Can the game handle errors?	Total number of errors recovered/Total number of errors found
		Can the game function on the device with no real button?	Physical button accessibility
	Attractiveness	Does player customize their interface	User Interface Appearance Customisability

		appearance?	
<b>Gameplay</b>	Understandability	Does the player know how to play the game?	Game Guidance Total number of guided modules Total number of modules
		Are the rules easy to understand?	Rules Understandability Total number of comprehended rules Total number of rules
	Learnability	Is the help system comprehensive?	Completeness of Help System Total number of topics covered
	Operability	Are the rules clear?	Rules Clarity Number of changes in rules Total number of rules specified

	Attractiveness	How is the game play methodology?	Gamely Method Attractive Interaction
	Compliance	Are the game criterions too hard?	Game criterion strictness  Number of changeable rules  Total number of rules specified

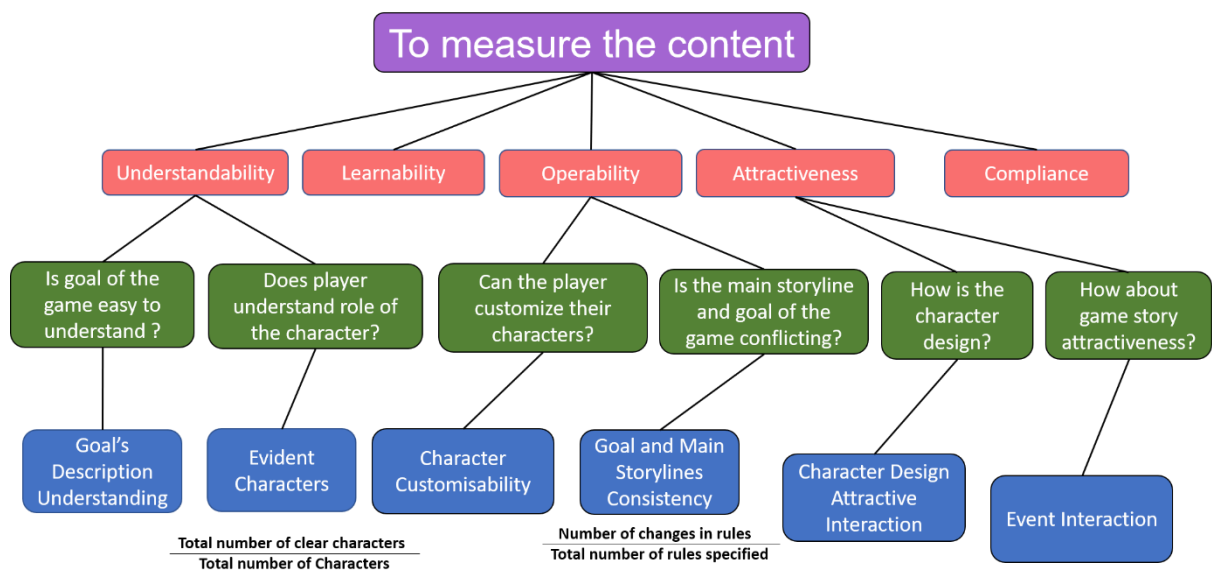


Fig 1. GQM to measure the content

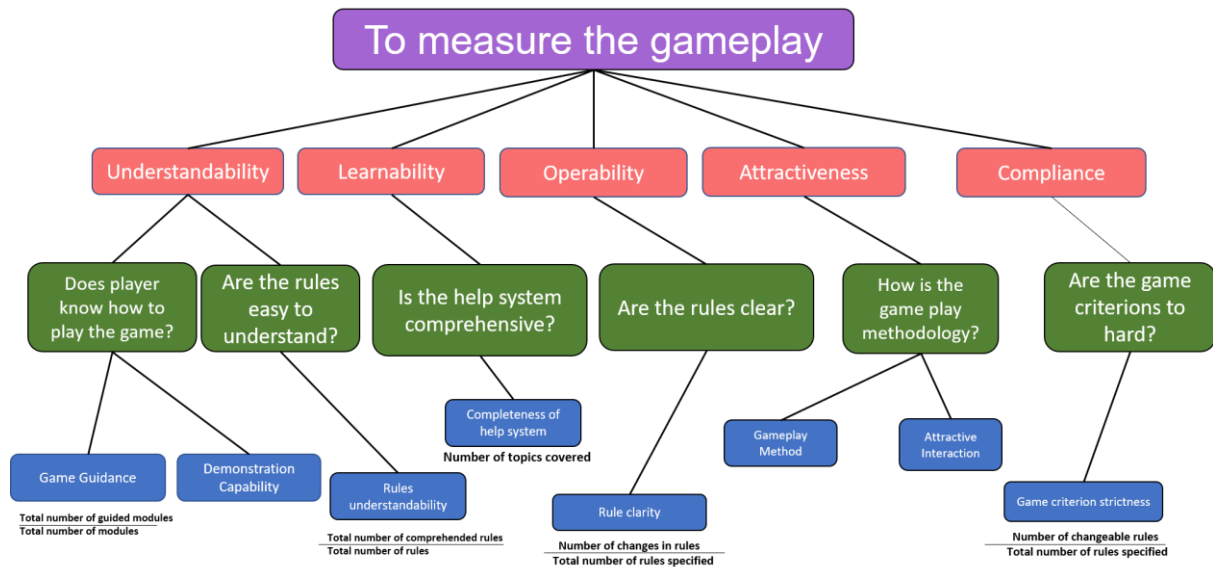


Fig 2. GQM to measure gameplay

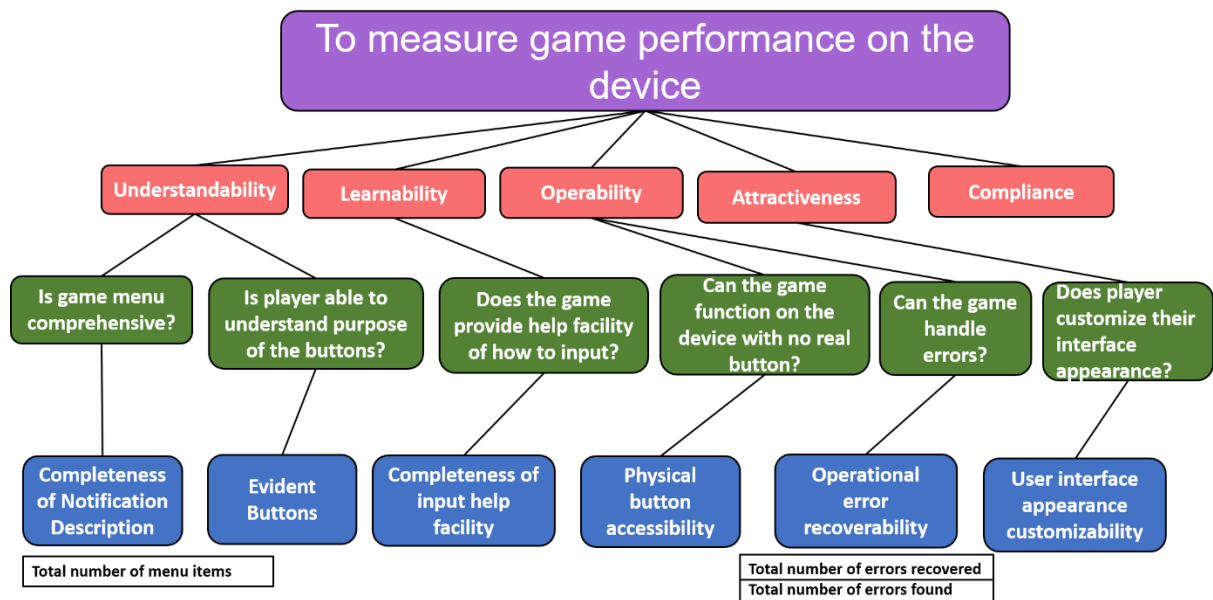


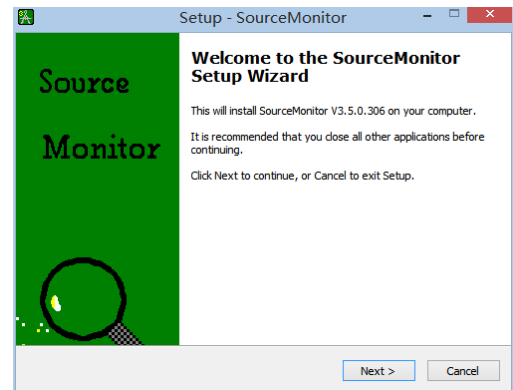
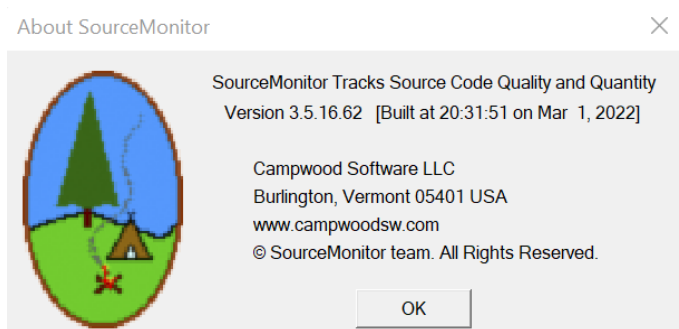
Fig 3. GQM to measure the game performance

## 4. Implementation Details: Tools & Approaches Used

### 1) Source Monitor

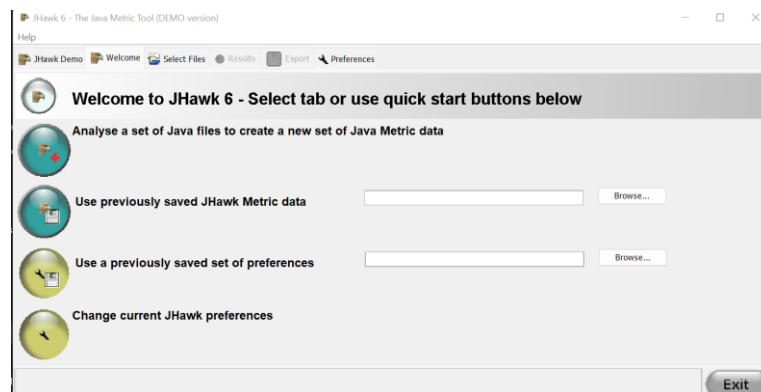
SourceMonitor is a freeware program that collects software metrics in a fast, single pass through the source files. It lets the user see how much code they have inside the source code and makes it easy to identify the relative complexity of different code modules. SourceMonitor is written using C++ programming language and can measure metrics which includes method and function level metrics for source code written in C, C++, and C #, Java, Delphi or HTML. This program can be operated within a standard Windows GUI or by using XML commands inside the scripts. It can display and print the results of metrics in tables and

charts and also has the ability to export the results to XML (Extensible Markup Language) or CSV (comma separated value) files for further processing with other tools. The problem with this tool is that it is not a web-based application and hence a user need to download and install the software before using it. The other issue is that the users are not permitted to modify or customize the software according to their needs even though the tool is free to download and use.



## 2) Jhawk

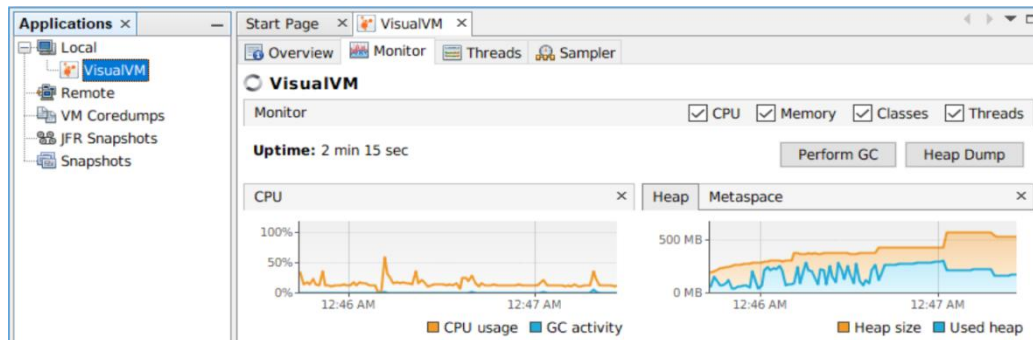
Hawk is a static code analysis tool - i.e. it takes the source code of your project and calculates metrics based on numerous aspects of the code - for example volume, complexity, relationships between class and packages and relationships within classes and packages. From the start JHawk has been the leader in Java metrics tools.



## 3) VisualVM

VisualVM is a tool that provides a visual interface for viewing detailed information about Java applications while they are running on a Java Virtual Machine (JVM). VisualVM organizes JVM data that is retrieved by the Java Development Kit (JDK) tools and presents the information in a way that allows data on multiple Java applications to be quickly viewed—both local applications and applications that are running on remote hosts. Programmers can also capture data about the JVM software and save the data to the local system, and then view the data later or share it with others. VisualVM is built on the NetBeans Platform; its architecture is modular and easy to extend with plugins. Java

VisualVM can be used by Java application developers to troubleshoot applications and to monitor and improve the applications' performance. Java VisualVM can allow developers to generate and analyze heap dumps, track down memory leaks, browse the platform's MBeans and perform operations on those MBeans, perform and monitor garbage collection, and perform lightweight memory and CPU profiling.



## 4. Metric Analysis: Detailed Analysis

### 1. Metrics calculated using SourceMonitor

SourceMonitor - [Java Checkpoints In Project 'Connect3GameSOURCEMONITOR']

File Edit View Checkpoint Window Help

Checkpoint Name	Created...	Files	Lines	Statements	% Branches	Calls	% Comments	Classes	Methods/Class	Avg Strmts/Method	Max Complexity	Max Depth	Avg Depth	Avg Complexity
Baseline	6 Nov 2022	7	384*	260	5.8	75	0.8	14	1.79	5.76	11*	6	1.61	2.28*

Table for SourceMonitor Results

S.No	Metric Type	Definition	Metric Value
1.	Files	It shows the total number of files or modules present in the app.	7
2.	Lines	This shows the total number of Lines of Code(LOC) in the app.	384
3.	Statements	It shows the group of code that is produced to create an expected output.	260

4.	<b>%Branch</b>	It gives the % of branch statements present in the source code.	<b>5.8</b>
5.	<b>Method Call Statements</b>	Shows the total number of method call statements present in the code.	<b>75</b>
6.	<b>%Comments</b>	It shows the % of lines with comments in the source code.	<b>0.8</b>
7.	<b>Classes and Interfaces</b>	Total number of classes and interfaces in the source code.	<b>14</b>
8.	<b>Average Methods per Class</b>	Shows the average methods present per class in the code.	<b>1.79</b>
9.	<b>Average Statements per Method</b>	This shows the average statements per method in the entire source code.	<b>5.76</b>
10.	<b>Max Complexity</b>	It shows the max complexity of the source code.	<b>11</b>
11.	<b>Max Block Depth</b>	Shows the max block depth in the entire code.	<b>6</b>
12.	<b>Average Block Depth</b>	Shows the average block depth.	<b>1.61</b>
13.	<b>Average Complexity</b>	Shows the average complexity.	<b>2.28</b>

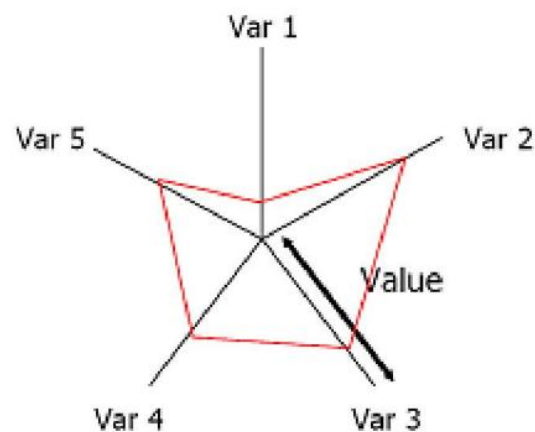
SourceMonitor - [All Methods in Java Project 'Connect3GameSOURCEMONITOR', Checkpoint 'Baseline']

File Edit View Window Help

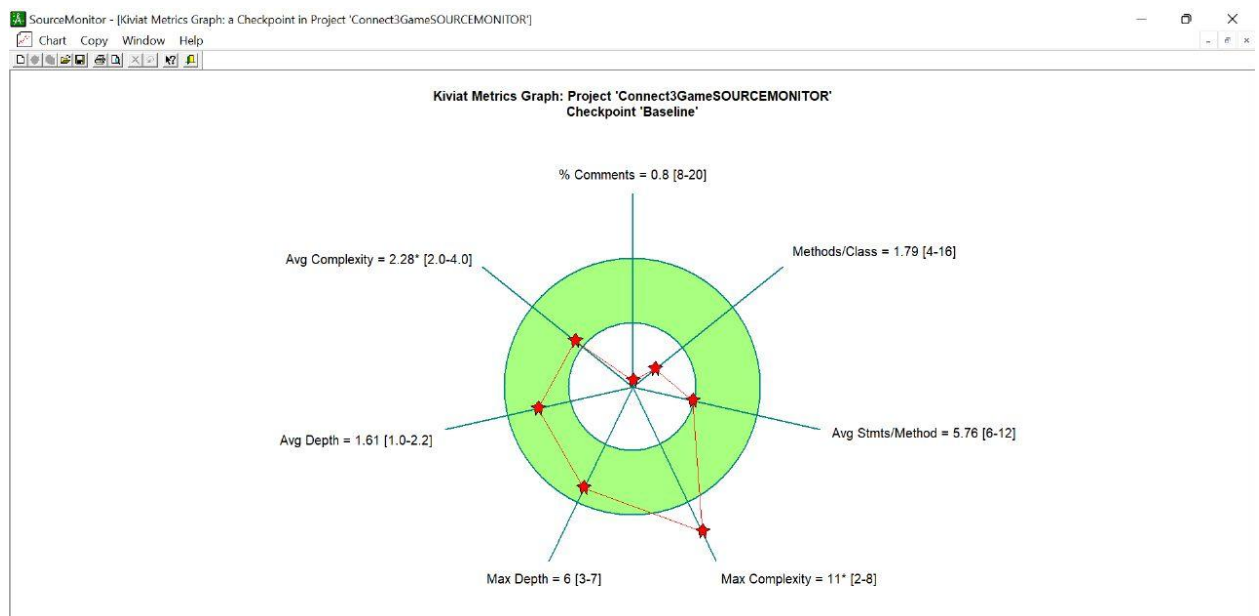
Class	Method Name	Complexity	Statements	Maximum Depth	Calls
DataBaseHelper	addOne()	3*	8	2	2
DataBaseHelper	DataBaseHelper()	1*	1	2	1
DataBaseHelper	deleteOne()	3*	7	3	3
DataBaseHelper	getAll()	3*	13	4	5
DataBaseHelper	onCreate()	1*	2	2	1
DataBaseHelper	onUpgrade()	1*	0	0	0
EndGame	onCreate()	9*	17	2	9
EndGame	showWinnerOnListView()	1*	2	2	2
Login	onCreate()	3*	4	2	4
Login	openMainActivity()	1*	8	2	7
MainActivity	draw()	1*	3	2	2
MainActivity	dropin()	11*	30	6	14
MainActivity	endGame()	3*	2	2	2
MainActivity	onCreate()	1*	11	2	6
MainActivity	playAgain()	3*	14	3	5
MainActivity	player1wins()	1*	3	2	1
MainActivity	player2wins()	1*	3	2	1
MainActivity	updatePointsBoard()	1*	2	2	2
savedData	onCreate()	1*	6	2	4
welcome	onCreate()	3*	4	2	4
WinnerModel	getName()	1*	1	2	0
WinnerModel	setName()	1*	1	2	0
WinnerModel	toString()	1*	1	2	0
WinnerModel	WinnerModel()	1*	0	0	0
WinnerModel	WinnerModel()	1*	1	2	0

### Kiviat Graph:

Generally, one single software metric has a two-dimensional data set, including the value and the time it was measured. Kiviat diagrams (or Star Glyphs, Star Plots) are widely used in two-dimensional visualization. They have axes for each attribute, and the axes are arranged like a star. The values of attributes are dots on the corresponding axes. Each dot is connected to another two dots nearby. Such diagrams can not only represent the value of each attribute exactly, but also can help us to have an overview of the whole object. In 2D Kiviat diagrams, the area of the shape indicates the integrated performance of an object. Generally, bigger area means better integrated performance.



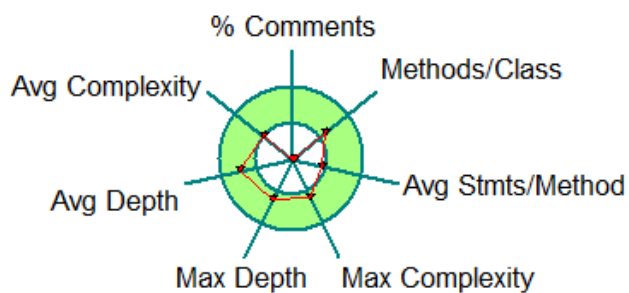
**Figure 1.1: A typical 2D Kiviat diagram**



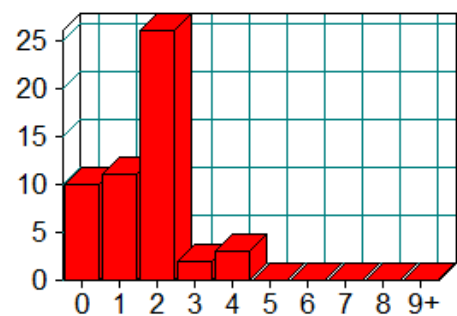


## DataBaseHelper.class

Kiviati Graph:

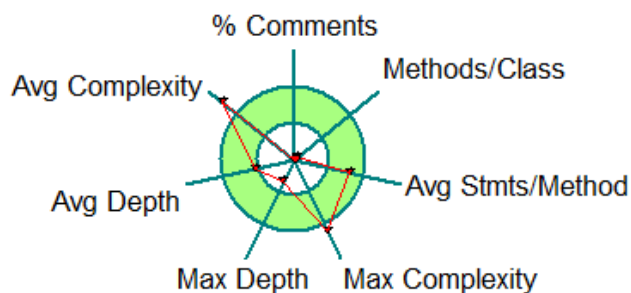


Block Histogram (statements vs. depth):

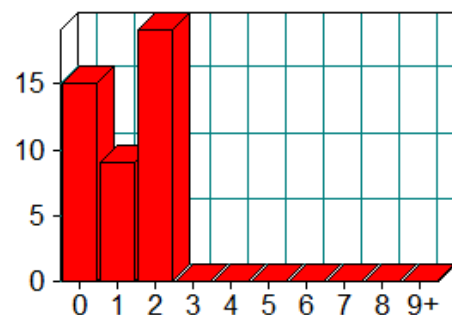


## EndGame.java

Kiviati Graph:

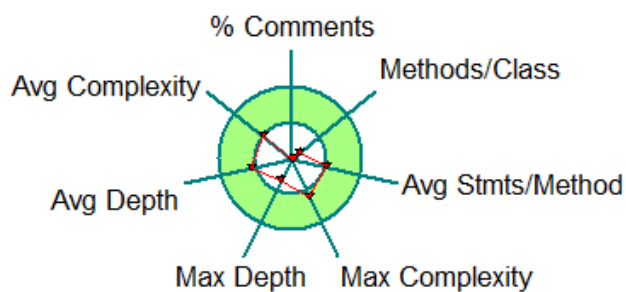


Block Histogram (statements vs. depth):

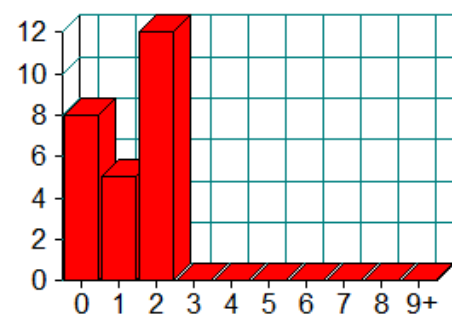


## Login.java

Kiviati Graph:

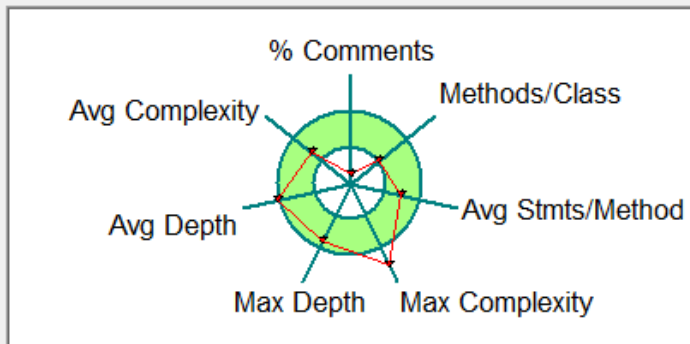


Block Histogram (statements vs. depth):

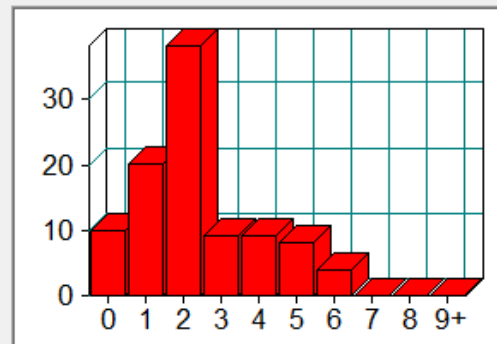


## MainActivity.java

Kiviati Graph:

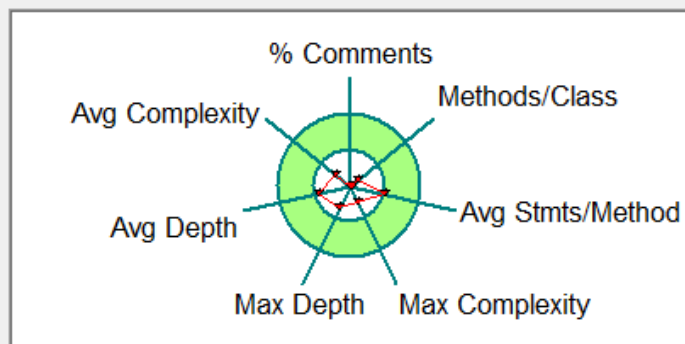


Block Histogram (statements vs. depth):

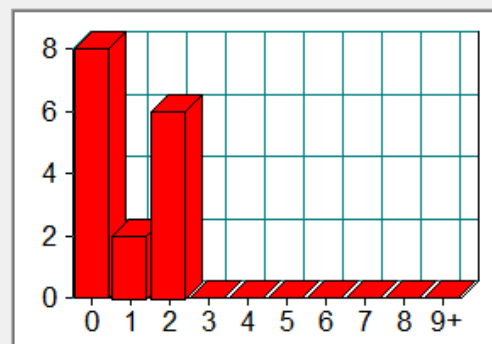


## savedData.java

Kiviati Graph:

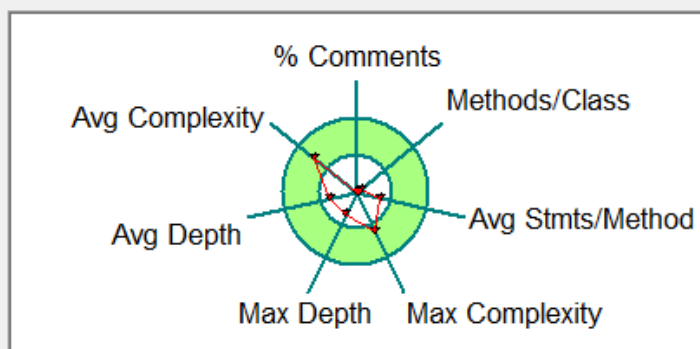


Block Histogram (statements vs. depth):

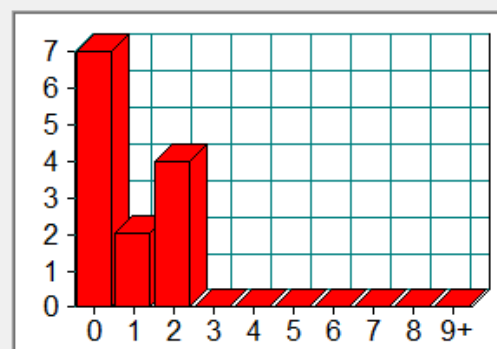


## Welcome.java

Kiviati Graph:

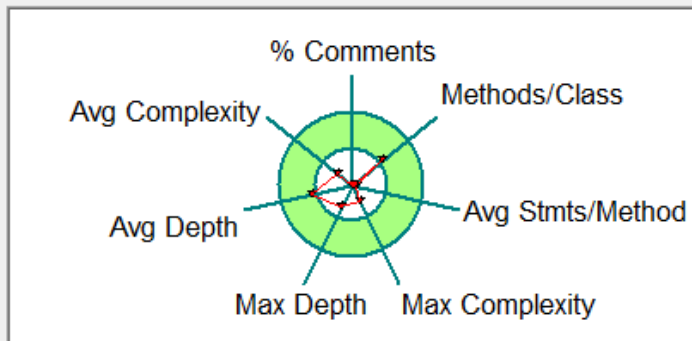


Block Histogram (statements vs. depth):

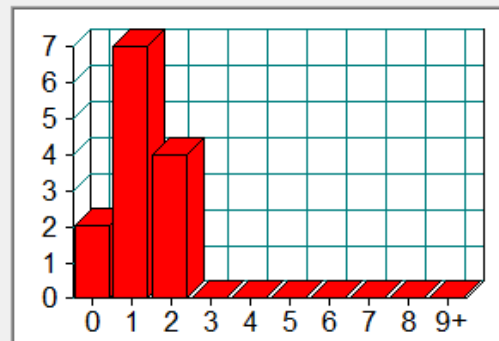


## WinnerModel.java

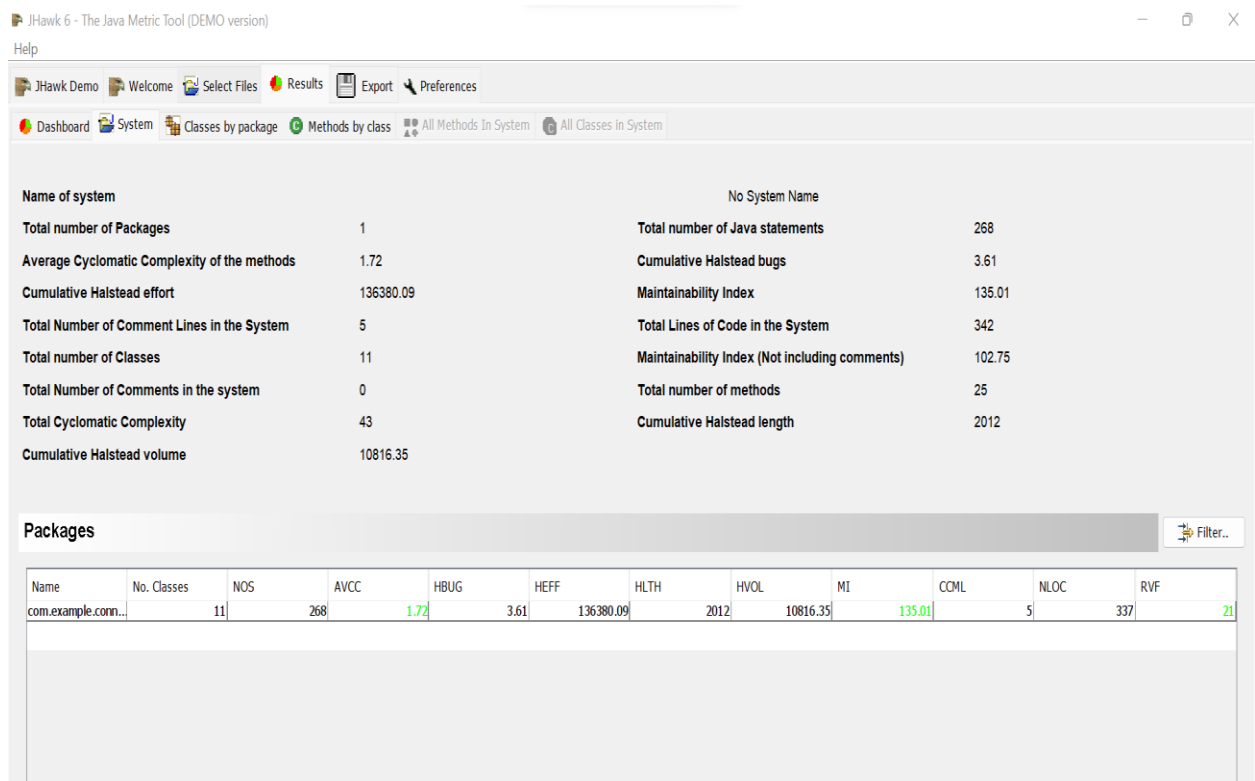
Kiviat Graph:

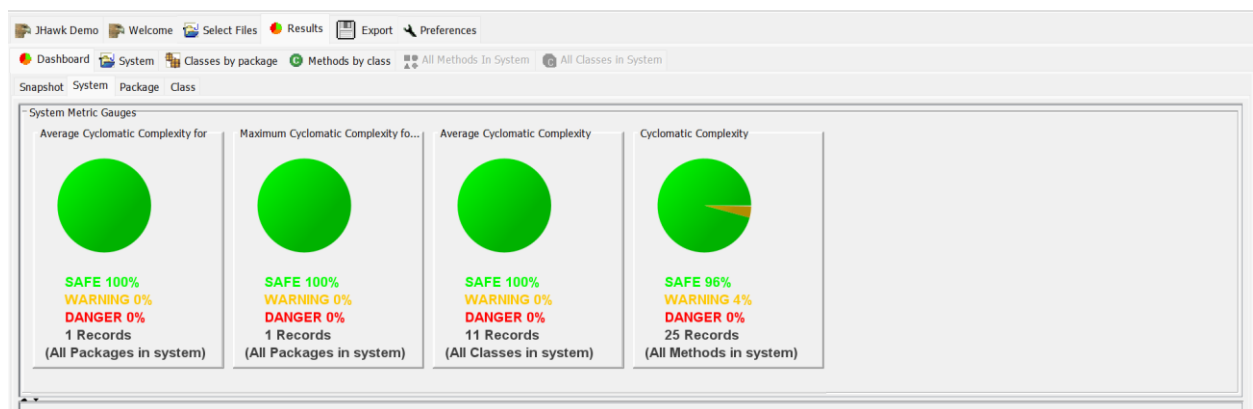
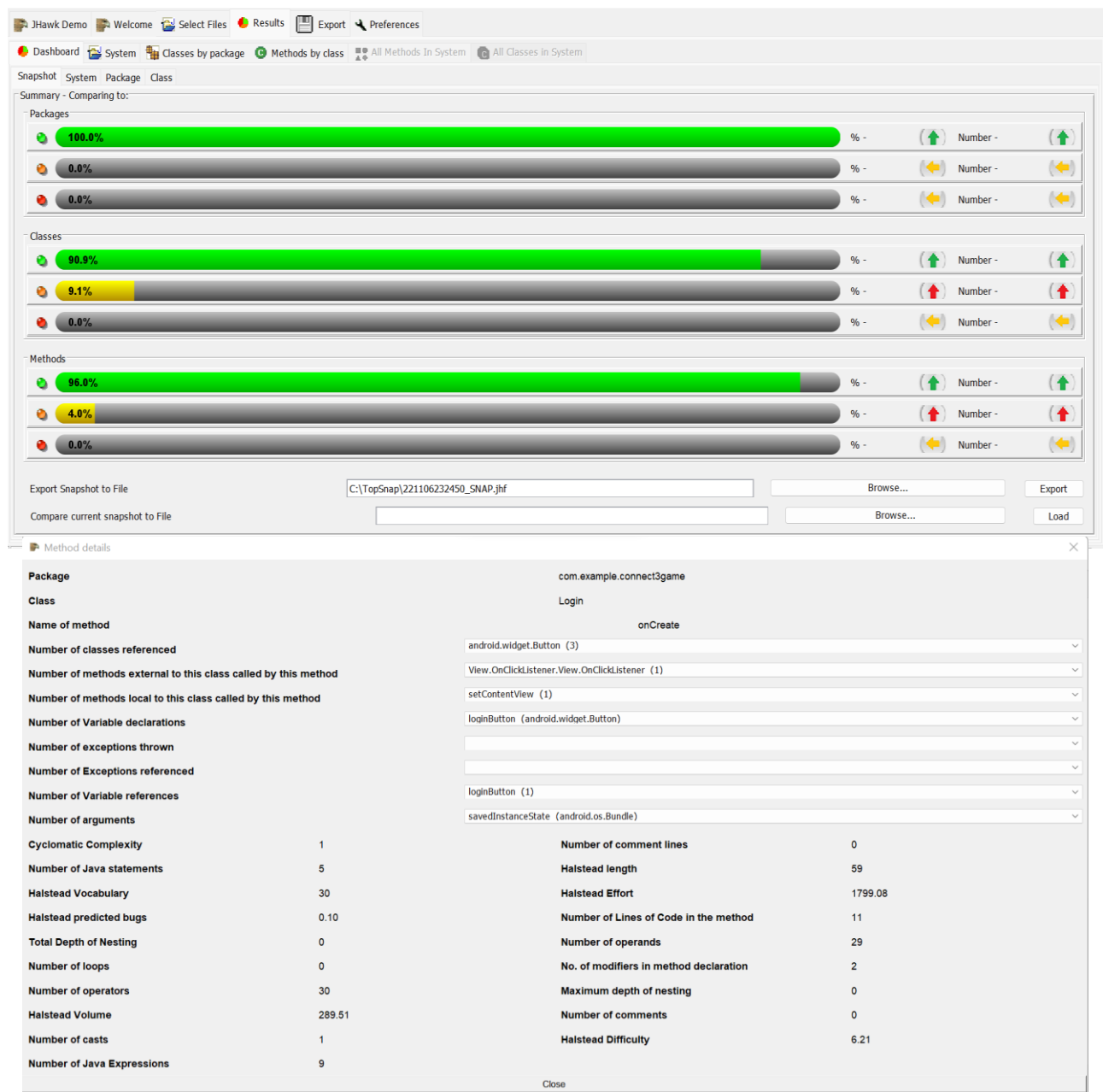


Block Histogram (statements vs. depth):



## 2. Metrics calculated using JHAWK tool



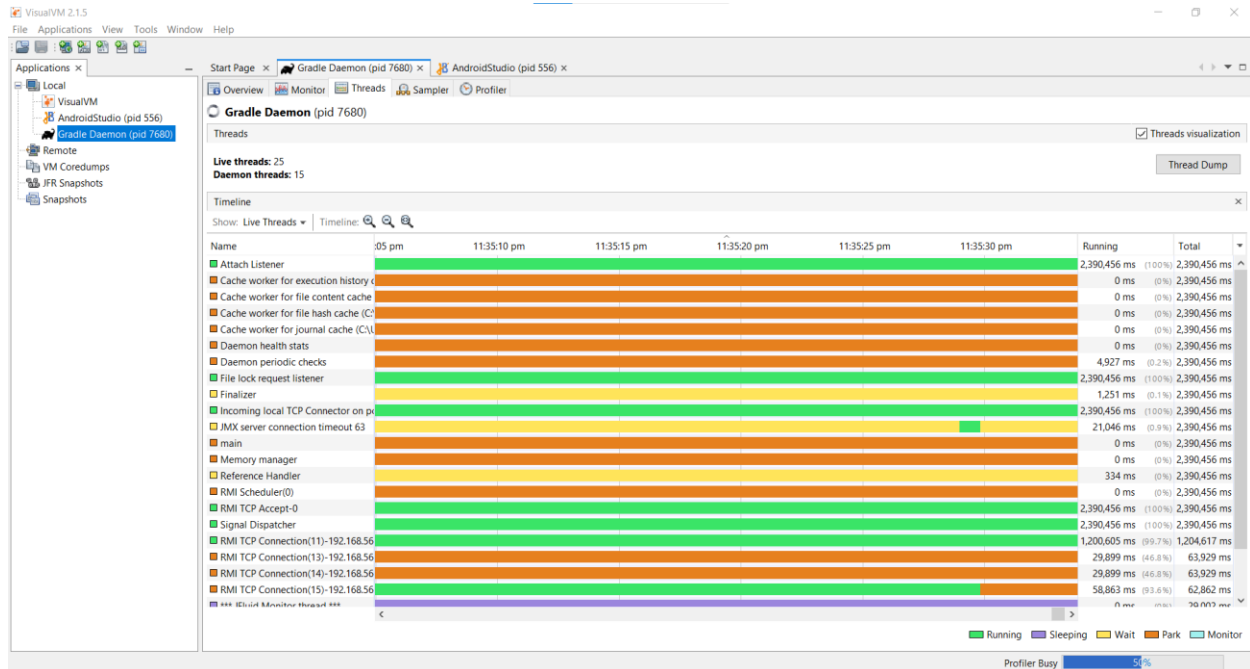


## Explanation of Metrics used

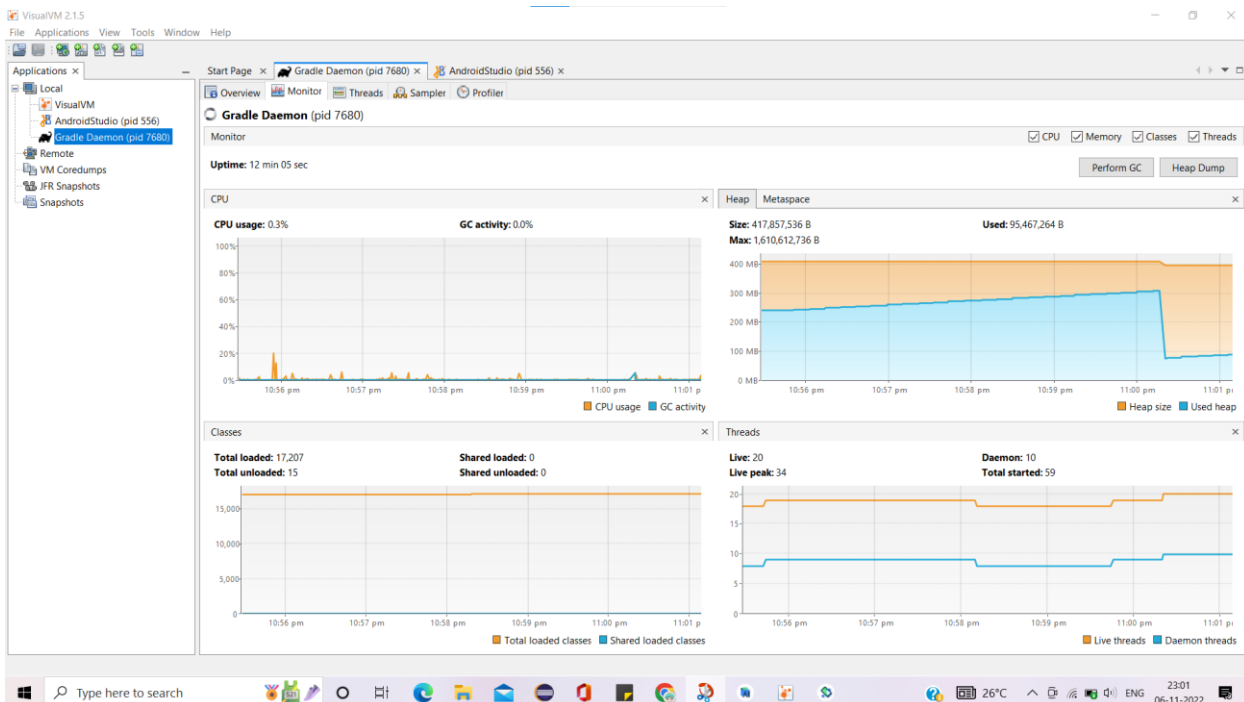
<b>Name</b>	The name of the class	Login
<b>Number of methods</b>	The number of methods in the class (WMC - one of the Chidamber and Kemerer metrics)	2
<b>Total Cyclomatic Complexity</b>	The Total McCabes cyclomatic Complexity for the class	1
<b>Average Cyclomatic Complexity</b>	The Average McCabes cyclomatic Complexity for all of the methods in the class	1.72
<b>Number of statements</b>	The number of statements in the class	5
<b>Cumulative Halstead length</b>	The Halstead length of the code in the class plus the total of all the Halstead lengths of all the methods in the class .	59
<b>Cumulative Halstead volume</b>	The Halstead volume of the code in the class plus the total of all the Halstead volumes of all the methods in the class .	289.51
<b>Cumulative Halstead effort</b>	The Halstead effort of the code in the class plus the total of all the Halstead efforts of all the methods in the class	1799.08
<b>Cumulative Halstead bugs</b>	The Halstead prediction of the number of bugs in the code of the class and all of its methods	0.10
<b>External method calls</b>	The external methods called by the class and by methods in the class	1
<b>Local method calls</b>	The local methods called by the class and by methods in the class that are defined in the hierarchy of the class	1
<b>Modifiers</b>	The modifiers (public,	2

	protected etc) applied to the declaration of the class	
<b>Maintainability Index (MI) (including comments)</b>	The Maintainability Index for the class, including the adjustment for comments	102.75
<b>Maintainability Index (MI)</b>	The Maintainability Index for the class without any adjustment for comments	135.01
<b>Depth of Inheritance Tree</b>	The value of the Depth of Inheritance Tree (DIT) metric for the class.	0
<b>Number of Comments</b>	The number of Comments associated with the class	0
<b>Number of Comment Lines</b>	The number of Comment Lines associated with the class	0
<b>Cumulative Number of Comments</b>	The number of Comments associated with the class and its method	0
<b>CumulativeNumber of Comment Lines</b>	The number of Comment Lines associated with class and its methods	0
<b>Lines of Code</b>	The number of lines of code in the class and its methods	342

### 3. Metrics using VisualVM



It shows the status of threads at the point of using the app.



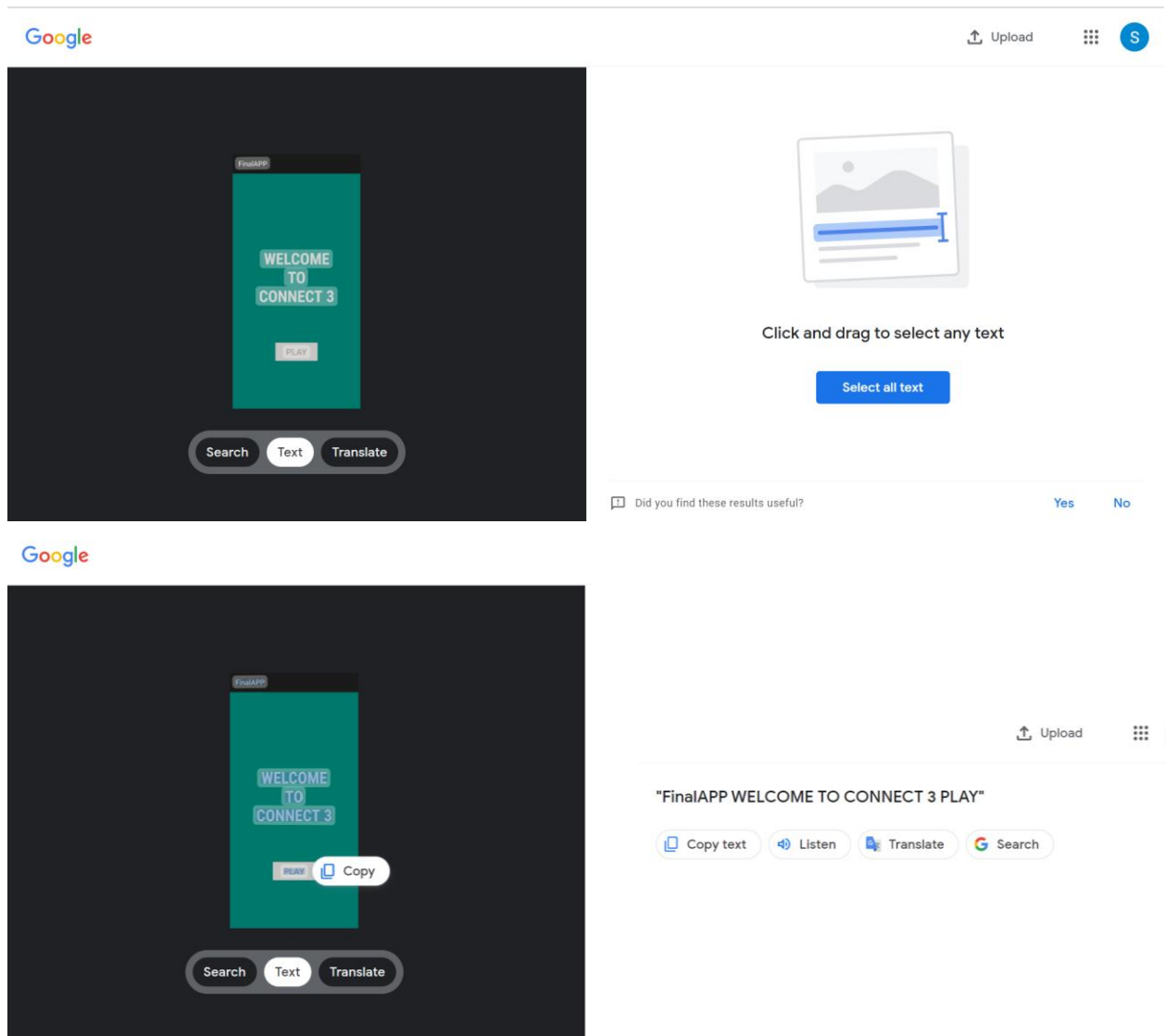
This shows the CPU usage of the app.

GC (Garbage Collection) is an automatic process of claiming unused allocated memory which was previously allocated

## 5. Manual Calculation of Metrics

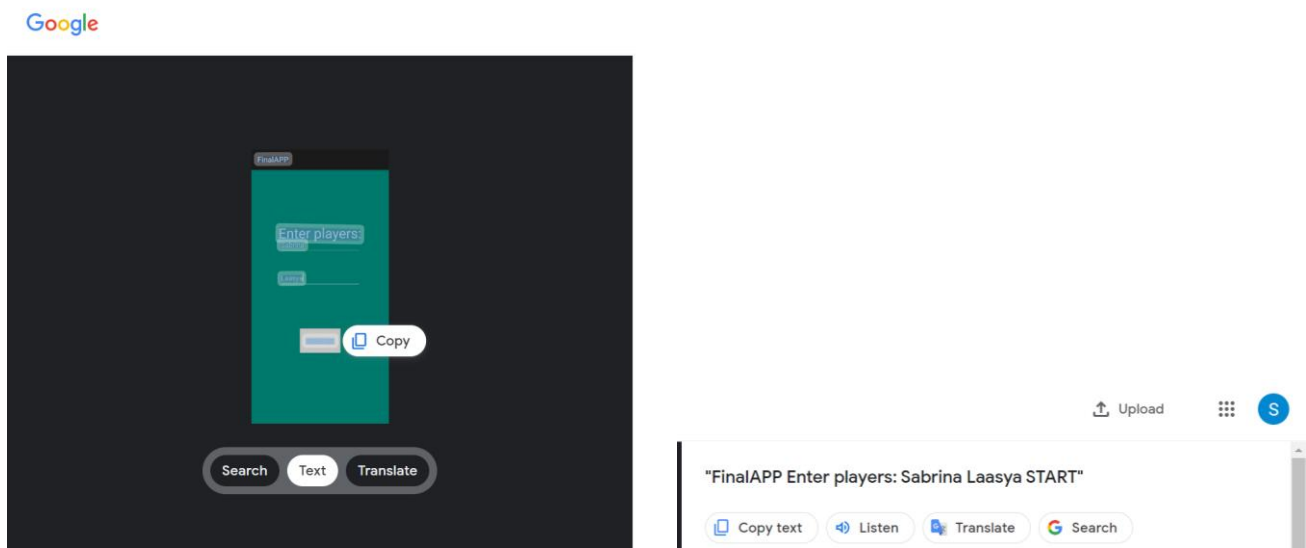
### 1) Understandability

a) Metric used: Evident Characters = Total number of clear characters/Total number of characters \*100

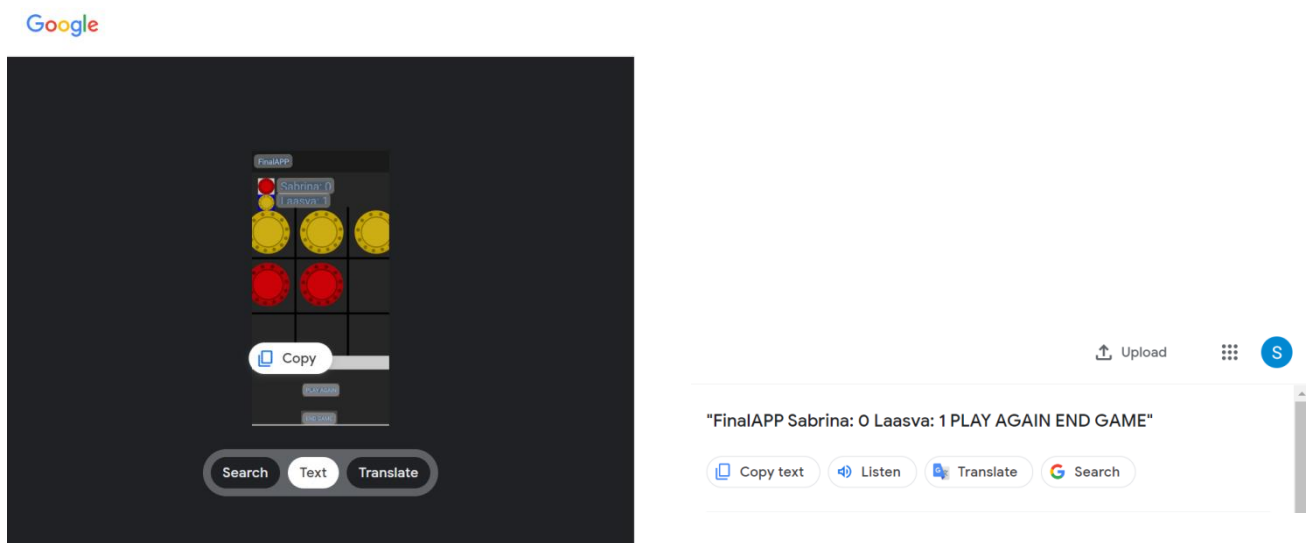


Total number of clear characters/Total number of characters \*100= (37/37) \*100 =100%

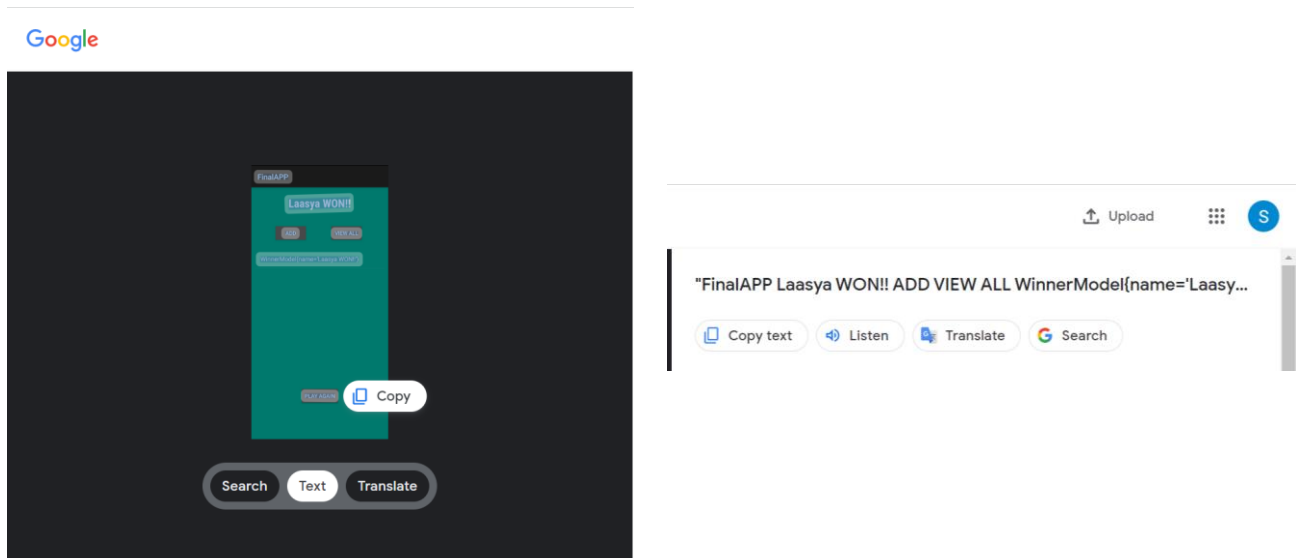




**Total number of clear characters/Total number of characters \*100= (44/44) \*100 =100%**

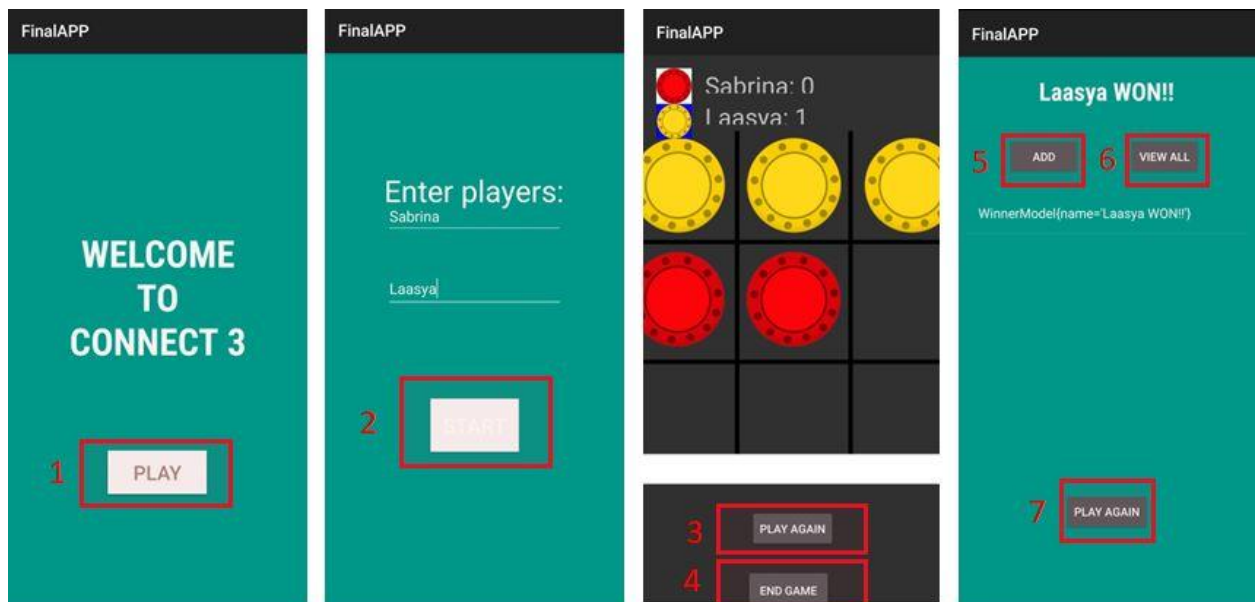


**Total number of clear characters/Total number of characters \*100= (46/47) \*100  
=97.87%**



Total number of clear characters/Total number of characters \*100= (79/79) \*100 =100%








**b) Metric used: Completeness of application = Total Number of Menu items (buttons)**



**Completeness of Notification Description: Total Number of Menu items (buttons)= 7**

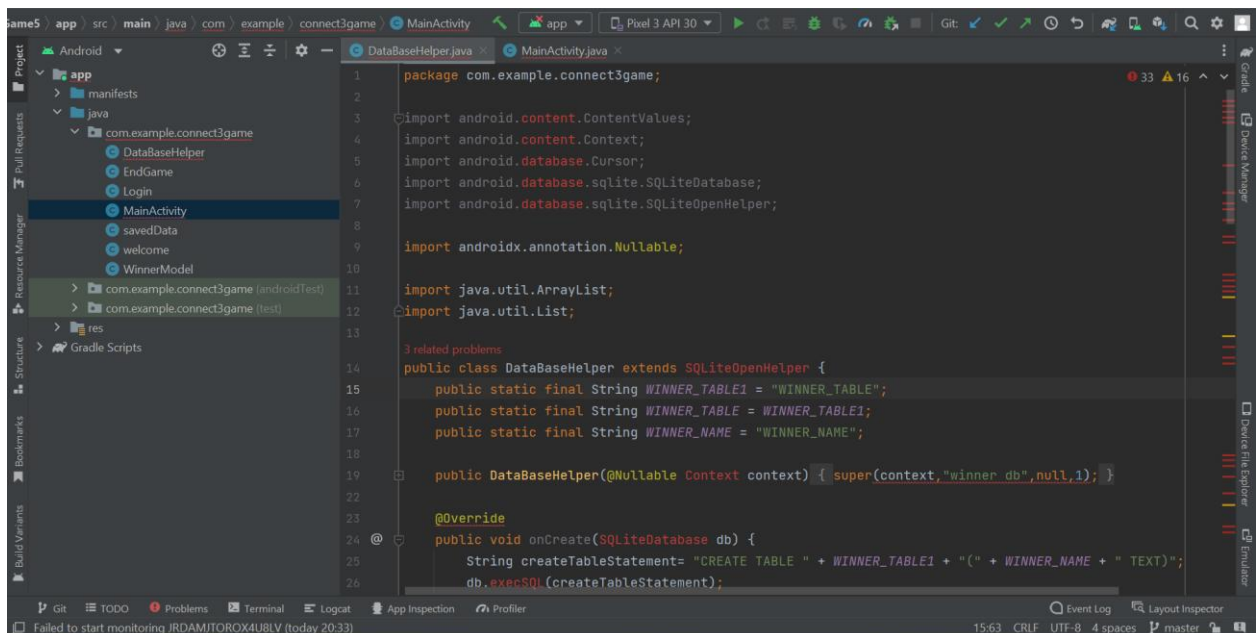
## c) Metric used : Game Guidance

Number of guided modules/ Total number of modules = 7/7= 100%

 DataBaseHelper	23-08-2022 20:26	JAVA File	3 KB
 EndGame	23-08-2022 20:26	JAVA File	4 KB
 Login	23-08-2022 20:26	JAVA File	2 KB
 MainActivity	23-08-2022 20:26	JAVA File	6 KB
 savedData	23-08-2022 20:26	JAVA File	1 KB
 welcome	23-08-2022 20:26	JAVA File	1 KB
 WinnerModel	23-08-2022 20:26	JAVA File	1 KB

## 2) Operatability

a) Metric used: Operational Error Recoverability= Total number of errors recovered/Total number of errors found



```
1 package com.example.connect3game;
2
3 import android.content.ContentValues;
4 import android.content.Context;
5 import android.database.Cursor;
6 import android.database.sqlite.SQLiteDatabase;
7 import android.database.sqlite.SQLiteOpenHelper;
8
9 import androidx.annotation.Nullable;
10
11 import java.util.ArrayList;
12 import java.util.List;
13
14 3 related problems
15 public class DataBaseHelper extends SQLiteOpenHelper {
16     public static final String WINNER_TABLE1 = "WINNER_TABLE";
17     public static final String WINNER_TABLE = "WINNER_TABLE1";
18     public static final String WINNER_NAME = "WINNER_NAME";
19
20     public DataBaseHelper(@Nullable Context context) { super(context, "winner db", null, 1); }
21
22     @Override
23     public void onCreate(SQLiteDatabase db) {
24         String createTableStatement= "CREATE TABLE " + WINNER_TABLE1 + "(" + WINNER_NAME + " TEXT)";
25         db.execSQL(createTableStatement);
26     }
27 }
```

```

1 package com.example.connect3game;
2
3 import android.content.ContentValues;
4 import android.content.Context;
5 import android.database.Cursor;
6 import android.database.sqlite.SQLiteDatabase;
7 import android.database.sqlite.SQLiteOpenHelper;
8
9 import androidx.annotation.Nullable;
10
11 import java.util.ArrayList;
12 import java.util.List;
13
14 public class DataBaseHelper extends SQLiteOpenHelper {
15     public static final String WINNER_TABLE1 = "WINNER_TABLE";
16     public static final String WINNER_TABLE = "WINNER_TABLE";
17     public static final String WINNER_NAME = "WINNER_NAME";
18
19     public DataBaseHelper(@Nullable Context context) { super(context, name: "winner db", factory: null, version: 1); }
20
21     @Override
22     public void onCreate(SQLiteDatabase db) {
23         String createTableStatement= "CREATE TABLE " + WINNER_TABLE1 + "(" + WINNER_NAME + " TEXT)";
24         db.execSQL(createTableStatement);
25     }
26 }

```

Total number of errors recovered/ Total number of errors found=  $23/33 \times 100 = 69.69\%$

## 6. Conclusion

Metric	Value measured by tool 1: Source Meter	Value measured by tool 2: JHAWK	Conclusion
Lines	384	342	Small project/organic in nature
Comments	0.8%~4	5 lines	Low commenting found
Classes identified	14	11	Avg classes= 12.5~13
Methods Identified	-	25	Method/class ratio=2.27
Max Complexity	11	43	Avg Complexity=27 Very Complex, Medium risk
Avg Complexity	2.28	1.72	

After conducting a thorough analysis of the Connect3 android game on three metrics tools, namely SourceMeter, JHAWK and VisualVM, we found that JHAWK gave the most in depth analysis.

From our analysis we have found that the system developed has around 350 lines of code, with 13 classes and 25 methods. From this we conclude that it can be taken as the organic project for development. Its complexity is 27 on average which makes it a complex system posing medium risks.

## 7. References

- [1] Beranič, T., Podgorelec, V., & Heričko, M. (2018). Towards a reliable identification of deficient code with a combination of software metrics. *Applied Sciences*, 8(10), 1902.
- [2] Gupta, E. M., Mehta, R. K., & Rai, E. M. (2019). A consolidated and comparative analysis of software metrics tools for systems performance evaluation: a survey. *IJARCCCE*, 8(4), 160-167.
- [3] Suryapranata, L. K. P., Soewito, B., Kusuma, G. P., Gaol, F. L., & Warnars, H. L. H. S. (2017, May). Quality measurement for serious games. In *2017 International Conference on Applied Computer and Communication Technologies (ComCom)* (pp. 1-4). IEEE.
- [4] Sabahat, N., Malik, A. A., & Azam, F. (2017). A size estimation model for board-based desktop games. *IEEE Access*, 5, 4980-4990.
- [5] Kasurinen, J., Palacin-Silva, M., & Vanhala, E. (2017, May). What concerns game developers? a study on game development processes, sustainability and metrics. In *2017 IEEE/ACM 8th Workshop on Emerging Trends in Software Metrics (WETSoM)* (pp. 15-21). IEEE.
- [6] Kurnia, R., Ferdiana, R., & Wibirama, S. (2018, November). Software metrics classification for agile scrum process: a literature review. In *2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)* (pp. 174-179). IEEE.
- [7] Thirumalai, C., Reddy, P. A., & Kishore, Y. J. (2017, April). Evaluating software metrics of gaming applications using code counter tool for C and C++ (CCCC). In *2017 International conference of Electronics, Communication and Aerospace Technology (ICECA)* (Vol. 2, pp. 180-184). IEEE.
- [8] Hsueh, C. H., Wu, I., Hsu, T. S., & Chen, J. C. (2018). An investigation of strength analysis metrics for game-playing programs: A case study in Chinese dark chess. *ICGA Journal*, 40(2), 77-104.

- [9] Junaidi, J., Julianto, A., Anwar, N., Safrizal, S., Warnars, H. L. H. S., & Hashimoto, K. (2018). Perfecting a video game with game metrics. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 16(3), 1324-1331.
- [10] Demyanova, Y., Pani, T., Veith, H., & Zuleger, F. (2017). Empirical software metrics for benchmarking of verification tools. *Formal Methods in System Design*, 50(2), 289-316.
- [11] Wu, B., & Wang, A. I. (2012). Comparison of learning software architecture by developing social applications versus games on the android platform. *International Journal of Computer Games Technology*, 2012.
- [12] Syer, M. D., Nagappan, M., Hassan, A. E., & Adams, B. (2013, November). Revisiting prior empirical findings for mobile apps: An empirical case study on the 15 most popular open-source Android apps. In *Proceedings of the 2013 Conference of the Center for Advanced Studies on Collaborative Research* (pp. 283-297).
- [13] Wenzel, T., Franke, T., Wasinger, R., & Körner, M. Identifying Development-Metrics for Use in a Gamified Mobile Web Application to Support Software Development.
- [14] Fields, T. V. (2013). Game industry metrics terminology and analytics case study. In *Game Analytics* (pp. 53-71). Springer, London.
- [15] Aleem, S., Capretz, L. F., & Ahmed, F. (2016). Game development software engineering process life cycle: a systematic review. *Journal of Software Engineering Research and Development*, 4(1), 1-30.
- [16] Portioresearch.com [Internet]. Mobile Applications Futures 2013-2017. 2014 [cited 2014 Nov 20]. Available from: <http://www.portioresearch.com/en/mobile-industry-reports/mobileindustry-research-repots/mobile-applications-futures-2013-2017.aspx>.
- [17] Statisticbrain.com [Internet]. Mobile Phone App Store Statistics. 2014 [cited 2011 Mar 29]. Available from: <http://www.statisticbrain.com/mzmobile-phone-app-store-statistics>.
- [18] Mobilestatistics.com [Internet]. Total Apps Available. 2012 [cited 2014 Apr 15]. Available from: <http://www.mobilestatistics.com/mobilestatistics>.
- [19] Simon Khalaf. Flurry Blog [Internet]. Apps Solidify Leadership Six Years into the Mobile Revolution. 2014 [cited 2014 Apr 5]. Available from: <http://blog.flurry.com>.
- [20] Srisuriyasavad, A. and N. Prompoon, Defining Usability Quality Metric for Game Prototype using Software Attributes, IMECS 2013, Hong Kong.
- [21]Caldiera, V. R. B. G., & Rombach, H. D. (1994). The goal question metric approach. *Encyclopedia of software engineering*, 528-532.