**Chapter -1**

**INTRODUCTION**

This chapter aims to give a general idea of the overall project. It includes the project proposal, purpose and some objectives which will be reached after completing the project. There is also an overview of all the chapters of the report.

The final product is a Paint Tool.

* 1. **PROJECT PROPOSAL**

Modern Technology brought the use of pictures in various fields. Also a simple picture speaks thousand words. So applications are required to create images and save them. An attempt has been made to do the same using simple algorithms.

The objective of the project is to create an application that can draw simple images. As java is the simple language for coding, java language has been chosen for the purpose. The Paint using javaFX application is created to easily draw images using several tool options. They include options like drawing circles, lines, rectangles, ellipses, polygon and solid fill areas in 2D. However, complex 3D images are not considered for the project.

The images created using this application can be saved. We can further retrieve the images and view them.

* 1. **PROJECT PURPOSE**

The purpose of this project is to create an interactive tool using the JavaFX and scene builder tool kit. There is a tendency to sale and collection to the solution, even before the problem is understood. This often leads to a software solution a wrong problem. To overcome this approach of software development we spent a good amount of time to understand the problem. In fact we identified and listed all the tasks performed manually, and sorted out the tasks which can be accommodated in our software.

**1.3 AIMS AND OBJECTIVES**

The name of the project is “Paint Tool”.

This is a simple javaFX implementation of a paint- brush style program.

We use java language for making this project. In this project we use javaFX and scene builder IDE. Without the use of java language and javaFx we wouldn’t be able to complete this project.

 It is a small example of what java can do. It has the basic drawing tools such as pen, line, eraser, clear, brush, filled and non-filled rectangles and oval and color manipulation options, other than that its really boring, but a good example of how to use java's graphics libraries. This project is simple so it is easy for user to understand.

We need Java Scene builder IDE to run the program of this project. We need java and java javaFX to complete this project.

In this project we make Pen, Line, Rectangle, triangle, Oval, and Brush. If we want to draw line then we can click on line option and we can click on color which we want to take. After this drag the cursor then the operation will be performed. Like this we can draw rectangle, oval etc. If we want to draw something free hand then we can use pen. We can use brush to paint anything.

**1.4 SCOPE OF THE PROJECT**

* This Paint tool can be used to draw color and edit pictures, including imported pictures from a digital camera for example.
* It can save creations in standard formats such as .jpg and .bmp
* Paint can enable ‘painting’ by dragging the mouse and using different types of artistic brushes or pens that can give for example a watercolor or oil effect. The paint also ‘runs out’ on certain brushes after a period of time so that the brush needs to be put back into the color again, as if it was a real painting brush.
* Features included in paint are pencil, brush and airbrush tool. The ability to add text, lines and shapes. Also included are an eraser, magnifier, and fill color tools.
* Many complex graphics software applications have concepts that are included in this Paint tool and the same principals can be applied in these from learning within Paint tool.

**Chapter - 2**

**BACKGROUND THEORY**

This chapter provides some background information.

**2.1 INTRODUCTION TO JAVA**

**Java: -** Java is a programming language originally developed by[James Gosling](http://en.wikipedia.org/wiki/James_Gosling) at[Sun](http://en.wikipedia.org/wiki/Sun_Microsystems) [Microsystems](http://en.wikipedia.org/wiki/Sun_Microsystems) which has since merged into Oracle Corporation and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++ but has a simpler object model and fewer low-level facilities. Java applications are typically compiled to byte code (class file) that can run on any Java Virtual Machine (JVM) regardless of computer architecture. Java is a general purpose, concurrent, class-[based,](http://en.wikipedia.org/wiki/Class-based) object-oriented language that is specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere"(WORA), meaning that code that runs on one platform does not need to be recompiled to run on another. Java is as of 2012 one of the most popular programming languages in use, particularly for client-server web applications, with a reported 10 million users.

**JavaFX: -** JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc.

To develop GUI Applications using Java programming language, the programmers rely on libraries such as Advanced Windowing Tool kit and Swing. After the advent of JavaFX, these Java programmers can now develop GUI applications effectively with rich content.

**Scene Builder: -** JavaFX Scene Builder is a visual layout tool that lets users quickly design JavaFX application user interfaces, without coding. Users can drag and drop UI components to a work area, modify their properties, apply style sheets, and the FXML code for the layout that they are creating is automatically generated in the background. The result is an FXML file that can then be combined with a Java project by binding the UI to the application’s logic.

**Chapter - 3**

**REQUIREMENTS**

This chapter will discuss the project feasibility of the project as well as both the functional and non-functional requirements.

**3.1 PROJECT FEASIBILITY – THE INCEPTION PHASE**

Project feasibility coincides with the inception phase of an iterative methodology such as agile. Inception in one sentence: “Do the stakeholders have basic agreement on the vision of the project, and is it worth investing in serious investigation?”

When thinking similarly in the lines of an experienced software engineer, the first step of developing any project is to investigate whether or not it should take place at all. If it is decided that it is pointless in doing so, then time on further research can be eliminated. Also, asking the question “what other existing systems or programs are out there similar to the intended one, and what scope it will bring to users in comparison with those that are existent” should also be answered before making a decision.

**3.2 THE REQUIREMENTS**

Requirements are everything the system will/should have from the designer’s perspective. As a form of aid, two techniques have been considered and used to assess the requirements. These are:

1. MoSCow – acronyms stand for **MUST**, **SHOULD**, **COULD** and **WON’T** respectively.
2. FURPS – acronyms stand for **FUNCTIONALITY**, **USABILITY**, **RELIABILITY**, **PERFORMANCE**, and **SUPPOTABILITY** respectively. In FURPS, functionality and usability are to be considered for the functional requirements whereas reliability, performance and supportability are to be considered for the non functional requirements.

**3.2.1 THE FUNCTIONAL REQUIRMENT**

***“Functional requirements*** capture the intended behavior of the system. This behavior may be expressed as services, tasks or functions the system is required to perform.”

Figure 3.2.1.1 shows all the functional requirements of the end product. Some of these are derived from the objectives of the project from chapter 1.

Requirements are all the options the program will contain once it has been opened by a user, and the application implementation requirements of the system.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **CATEGORY** | **REQUIRMENT** | **PRIORITY** |
| FU01 | THE PROGRAM | Open a new empty project. | HIGH |
| FU02 | THE PROGRAM | User should be able to open a previous created file or to insert an image. | MEDIUM |
| FU03 | THE PROGRAM | User should be able to save his project in jpeg, png, gif and other formats of image etc. | VERY HIGH |
| FU04 | THE PROGRAM | User should be able to save his file again under a different name. | MEDIUM |
| FU05 | THE PROGRAM | With the use of Exit function user should be able to close the whole application. | HIGH |
| FU06 | THE PROGRAM | The user should be able to draw different shapes. | HIGH |
| FU07 | THE PROGRAM | User should be able to insert Text. | HIGH |
| FU08 | THE PROGRAM | With the color tool the user can choose the preferred fill color and border color for the shapes and brush. | VERY HIGH |

**FIGURE 3.2.1.1: Functional requirements of the program**

**3.2.2 THE NON-FUNCTIONAL REQUIRMENTS/SUPPLEMENATRY SPECIFICATION**

***Non functional requirements*** are the requirements that do not have any link with how the system actually works, but are still important. One way to look at them is by using the ‘RPS’ of the FURPS model.

|  |  |  |  |
| --- | --- | --- | --- |
| **ID** | **FURPS** | **REQUIRMENT** | **PRIORITY** |
| NF01 | RELIABILITY | The system will have no financial constraints. | HIGH |
| NF02 | PERFORMANCE | Fast response time | HIGH |
| NF03 | USABILITY | Attractive, user friendly interface | HIGH |
| NF04 | SUPPOTABILITY | Should be able to further develop | MEDIUM |

**3.3 SYSTEM REQUIREMENTS:**

System requirements will describe the functionality of the system in a detailed and technical way. System requirements should outline exactly what functional process the system must do and detail these processes logical and highly detailed manner. System requirements can be sub-categorized into functional and non-functional requirements.

**3.3.1 THE FUNCTIONAL REQUIRMENT**

Functional requirements for the system should describe the services which the system should provide in a technical way. They should include all the services the system needs to carry out and this should be done in a consistent and non–ambiguous manner.

* The system should notify the user which tool is used for which purpose.
* The system should display a user friendly interface.
* The system should notify the user before exiting from unsaved file.
* The system should give you many options for color selection.
* The system should save the file with different image formats like jpg, png etc.

**3.3.2 THE NON-FUNCTIONAL REQUIRMENT**

• The Application should be portable.

**3.4 HARDWARE AND SOFTWARE SPECIFICATIONS**

**3.4.1 HARDWARE REQUIREMENTS:**

Hardware is the physical components of a computer which includes CPU, internal storage unit’s output unit & other peripherals or any device that can be used in conjunctions with this unit. Broadly, hardware can be termed can be termed as the electronic circuit used in building the computer that executes the software.

**Minimum Requirements:**

|  |  |  |
| --- | --- | --- |
| S.No | Name | Specification |
| 1 | Computer | IBM PC ATX |
| 2 | Processor | Pentium IV |
| 3 | Main Memory | 256 RAM |
| 4 | Hard Disk | 10GB |
| 5 | Keyboard | Logitech |

**Recommended Specification**:

|  |  |  |
| --- | --- | --- |
| S.No | Name | Specification |
| 1 | Computer | IBM PC ATX |
| 2 | Processor | Pentium IV Dual core |
| 3 | Main Memory | 512 RAM |
| 4 | Hard Disk | 20GB |
| 5 | Keyboard | Any Multimedia Keyboard |
| 6 | Mouse | PS2/USB Mouse |

**3.4.2 SOFTWARE REQUIREMENTS:**

Software is the list of instructions to be executed by computer. Software is the part of computer system enables the hardware to operate software is an essential requirement of the computer.

**Minimum Requirements:**

|  |  |  |
| --- | --- | --- |
| S.No | Name | Specification |
| 1. | Operating System | Linux, Windows XP professional / Vista / Window-7 |
| 2. | Language | Java 7,8 |
| 3. | Application | NetBeans |
| 4. | Application | Scene Builder |

**Recommended Requirements:**

|  |  |  |
| --- | --- | --- |
| S.No | Name | Specification |
| 1 | Operating System | Windows XP professional/Window-7/8 |
| 2. | Language | Java 8 |
| 3. | Application | NetBeans 8.2 |
| 4. | Application | Scene Builder |

**Chapter - 4**

**DESIGN**

Design describes the way in which a system is to be developed according to its requirements. Preece (1994) states that “’design’ refers to both the process of developing a product, artifact or system and to the various representations (simulations or models) of the product that are produced during the design process” (Preece et al, 1994, p.352).

The design stage incorporates a lot of different representations of the system that would be particularly suitable for the specific stage. It is important for designers to remember this in that they must understand and know how functions are available at various different stages of design.

**4.1 SYSTEM ARCHITECTURE**

**4.1.1 PRESENTATION LAYER**

The interface of the “Paint Tool” will be designed using the Scene Builder. Graphics and images for the shapes, tools and emojis will be downloaded from the internet. The images will be imported into the scene Builder using ImageView control to create the interface so the application is as realistic as possible.

**4.1.2 FILE SYSTEM LAYER**

The File System Layer is concerned with the design and development of the project using Scene builder and Netbeans IDE. The project will be programmed using JavaFX. Through the file System we can save and open creations or images in standard formats such as .jpg and .bmp.

**4.1.3 THE PROCESSING LAYER**

The processing layer is the combination of the File System layer and the presentation layer showing how these two layers interact with each other to provide a working system. The Net Beans was chosen as the software to develop and implement the “Paint Tool”.

**4.2 BLOCK DIAGRAM**

**Application**

**Open**

**Tool Box**

**Color Box**

**Shapes**

**Line**

**Circle**

**Rectangle**

**Polygon**

**Curve**

**Other Shapes**

**Red**

**Green**

**Blue**

**Other Colors**

**Save**

**Text**

**Eraser**

**Fig: - Block Diagram of Paint Tool**

**Chapter - 5**

**IMPLEMENTATION AND TESTING**

The implementation phase involves translating the logical design of the system as laid down in the design phase into an actual working “Paint Tool” application that should fulfill the requirements of the system both functional and non-functional.

**5.1 HOW THE SYSTEM WAS IMPLEMENTED**

The system was implemented using the Net Beans editor to create a version of the “Paint Tool”. The user interacts with the application via Graphical User Interface.

It was implemented using the following tools and menus: -

**5.1.1 Menu Bar**

It should consist of the following menus:

1. File menu – For creating a new file, saving, opening a file, etc.
2. Help menu – This provides comprehensive help about the paint program.

**5.1.2 The Tool Box**

The tool box is very useful because it contains the tools you will use in order to come up with your drawing. It contains the following tools:

1. Eraser – this is for erasing your drawing or the color you have applied.
2. Pick color – this picks a specific color you want and makes it the active color, meaning that whatever you do next will have that color.
3. Line tool – for drawing a line. To make your line straight, hold down the shift key and then drag your mouse holding the left button. Select the thickness of the line below the toolbox.
4. Rectangle tool – for making rectangles. You can also draw a square by holding the shift key.
5. Ellipse – for drawing an ellipse, you can also draw a perfect circle by holding the shift key before you start dragging your mouse.
6. Fill with color – as the name suggests, you use this tool to fill an object with color at once.
7. Brush – use this for painting, just like you would paint your house.
8. Text – you will use this for entering text.
9. Curve tool – for drawing a curve.
10. Polygon tool – for drawing a polygon but still you can draw a lot of other objects.
11. Rounded rectangle – this tool helps you to draw a rectangle with rounded corners.
12. Emojis - This tool helps to add emojis.

**5.1.3 The Color Picker**

The color picker contains the colors that you will use for painting your drawings.

**5.2 TESTING**

Testing has a dual function within a system. It is used to ascertain the presence of defects in a system and it is used to help judge whether or not the system is usable. The testing process can demonstrate the presence of errors but cannot show that there are no errors within the system. Within this different testing techniques will be discussed and the preferred testing method for the Paint Tool will be detailed.

**5.2.1 BLACK BOX**

Black box testing is defined by Sommerville as “testing which relies on the specification of the system or component which is being tested to derive test cases” (Sommerville, 2001, p.466). The system is a ‘black box’ in that the internal workings of the item being tested are not known to the tester. The tester will only know the inputs to the system and the expected outcomes, but will not know how the system arrives at those outcomes.

**5.2.2 WHITE BOX TESTING**

White box testing is a method which requires the tester to have explicit knowledge of the internal workings of a system. Sommerville cites that “the tester can analyze the code and use knowledge about the structure of a component to derive test data”.(Sommerville, 2001, p.471). In this case the tests are only accurate if the tester knows what the program is supposed to do.

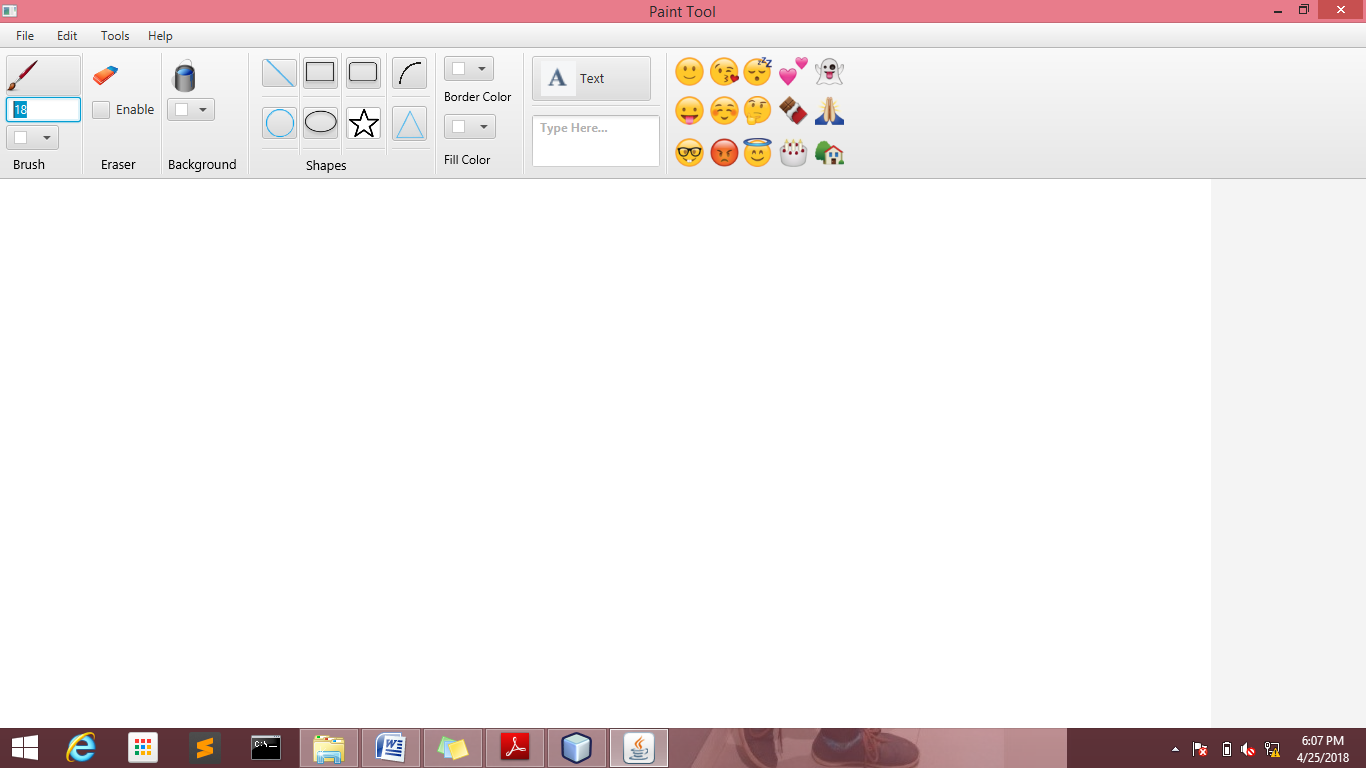
**5.2.3 UNIT TESTING**

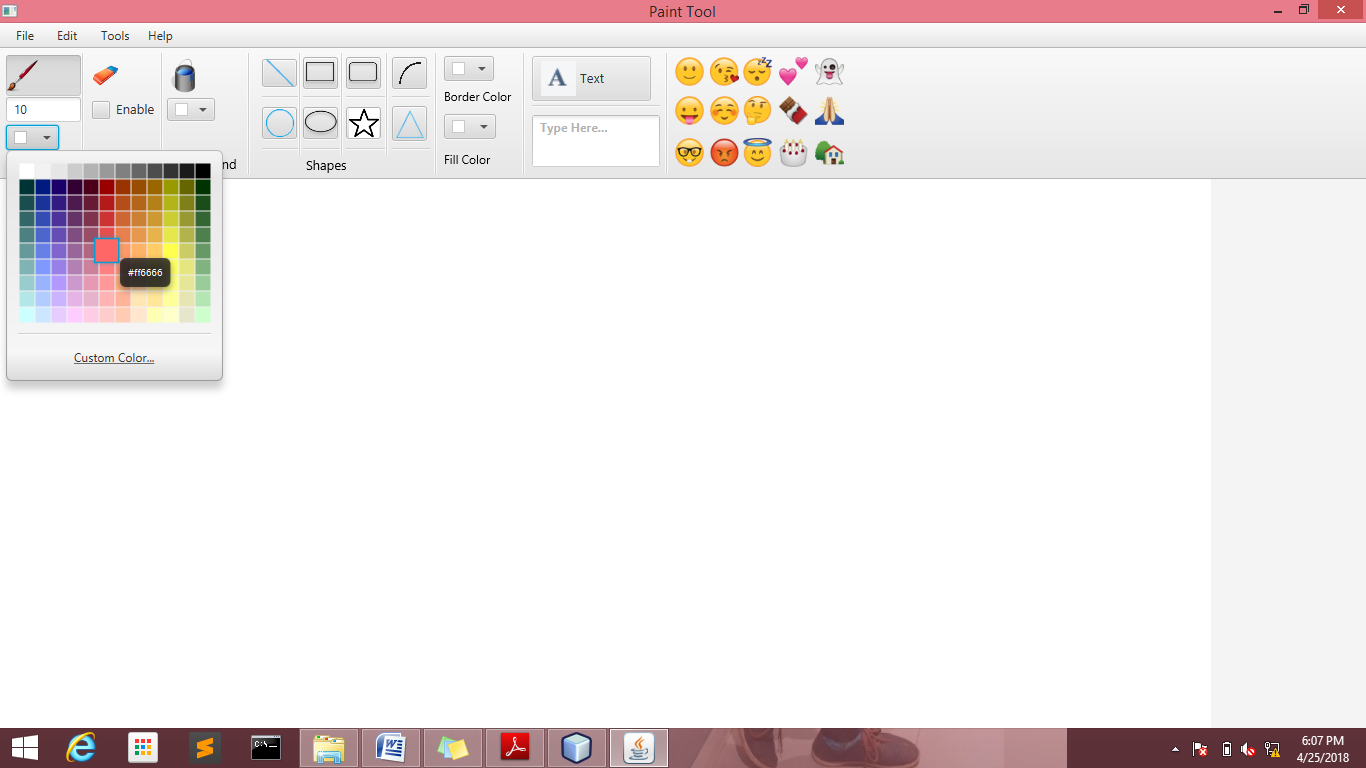
Testing of the Paint Tool was carried out following the Unit Testing methodology. Each individual function of the system was thoroughly tested and evaluated to ensure that the working system meets user requirements. Sommerville (2001, p.449) states that “individual components are tested to ensure that they operate correctly. Each component is tested independently, without other system components.”

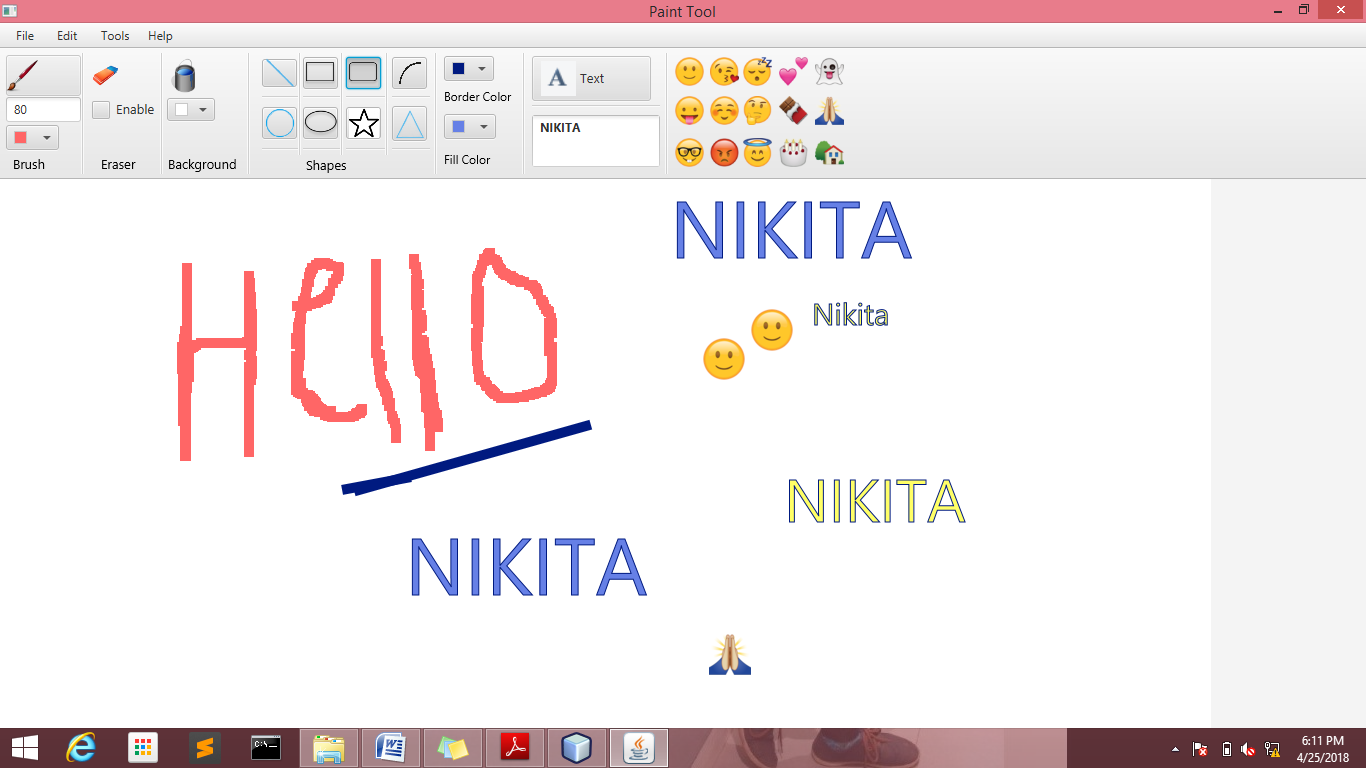
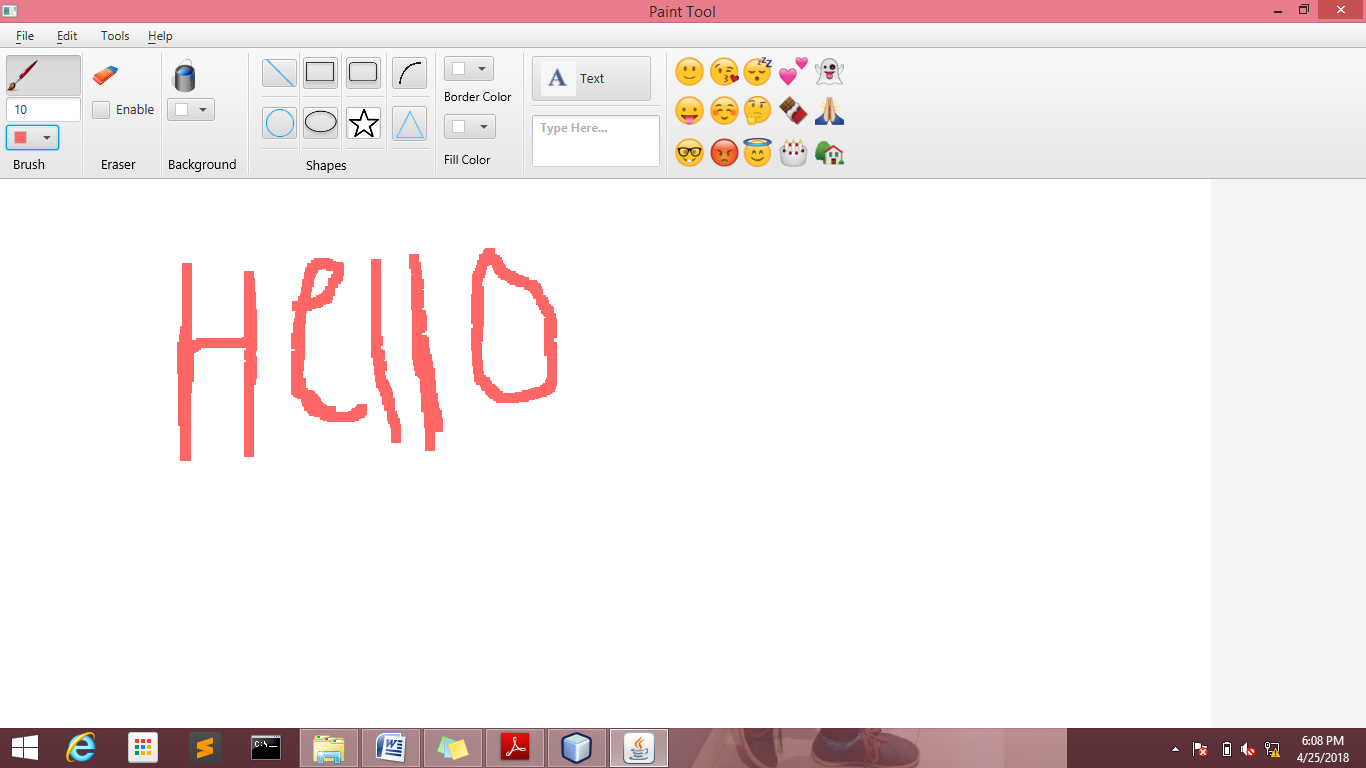
**Chapter - 6**

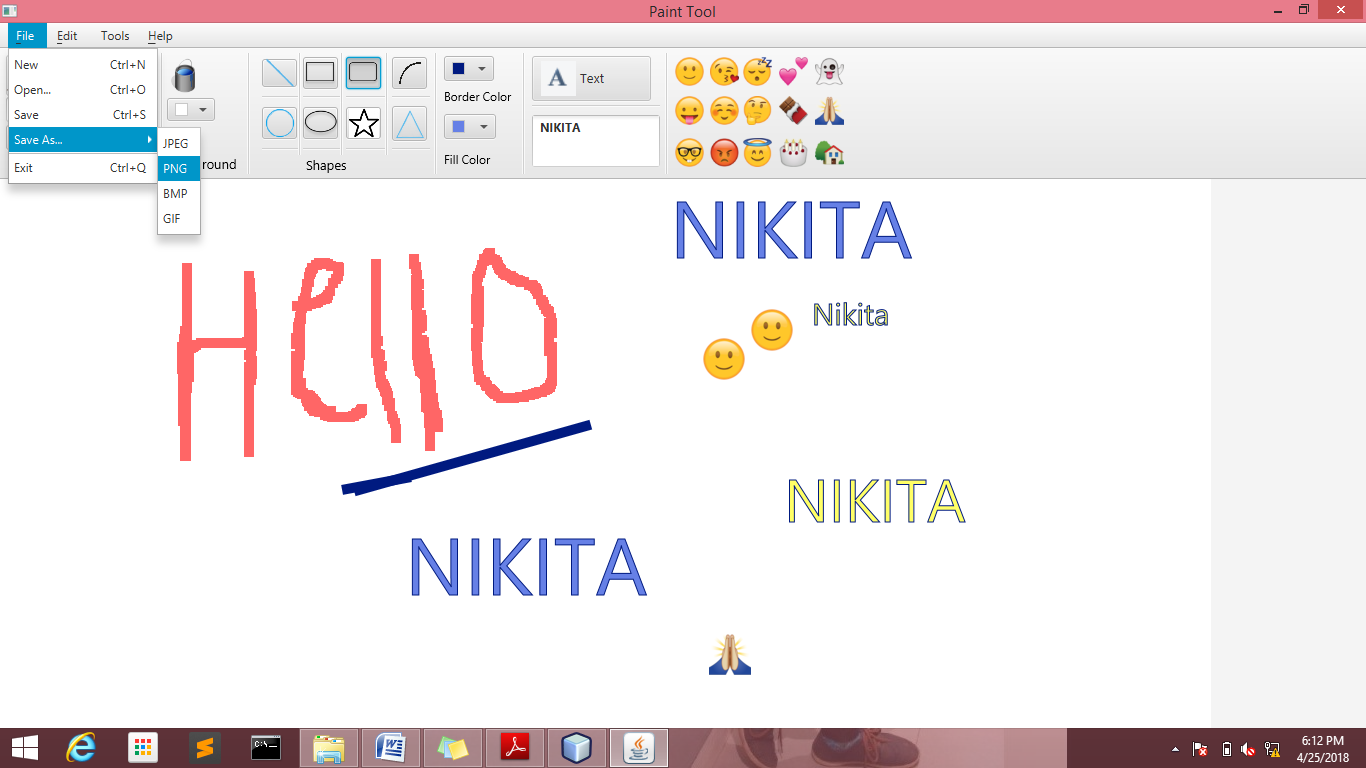
**RESULTS**

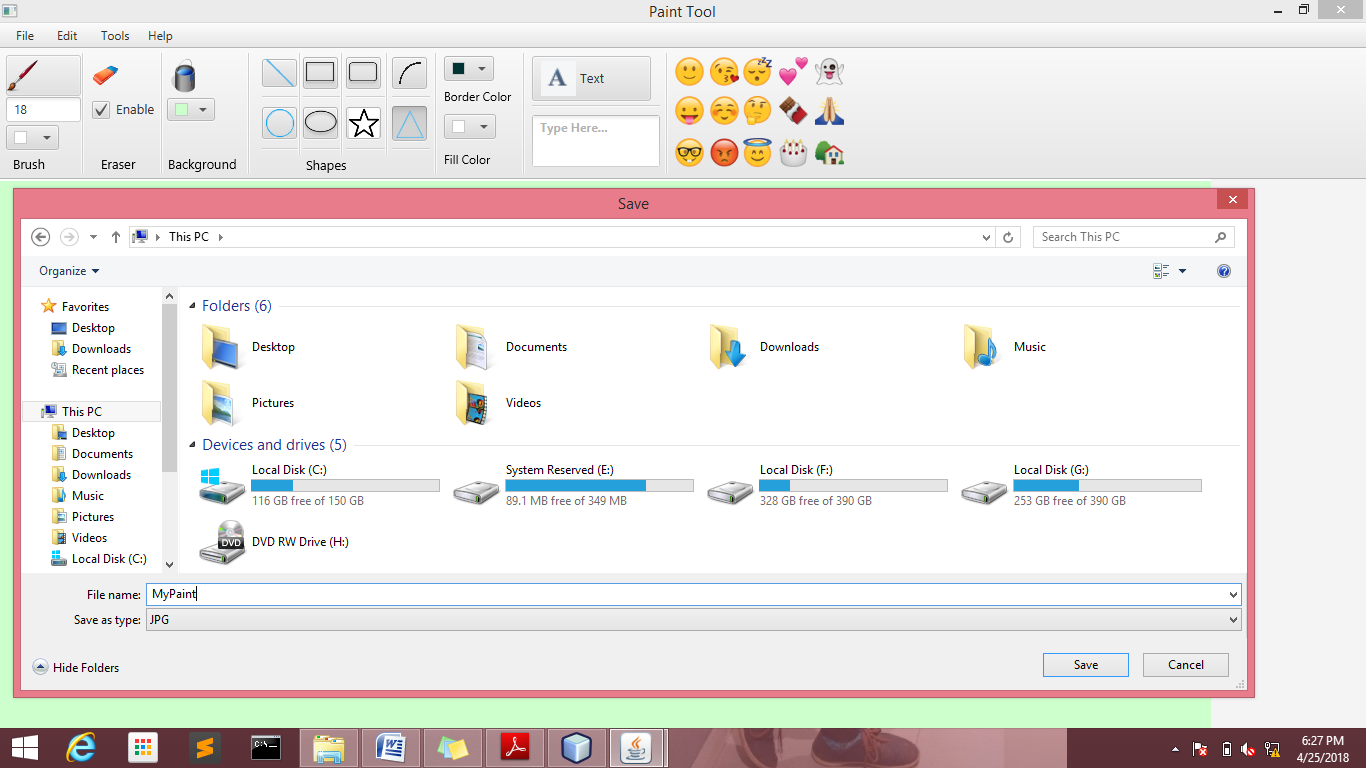
This chapter will discuss everything that has been achieved after the implementation. It shows how the final product works in practice. A simple step by step method with the aid of screen shots is used to explore everything the application has to offer with descriptions of what is occurring.

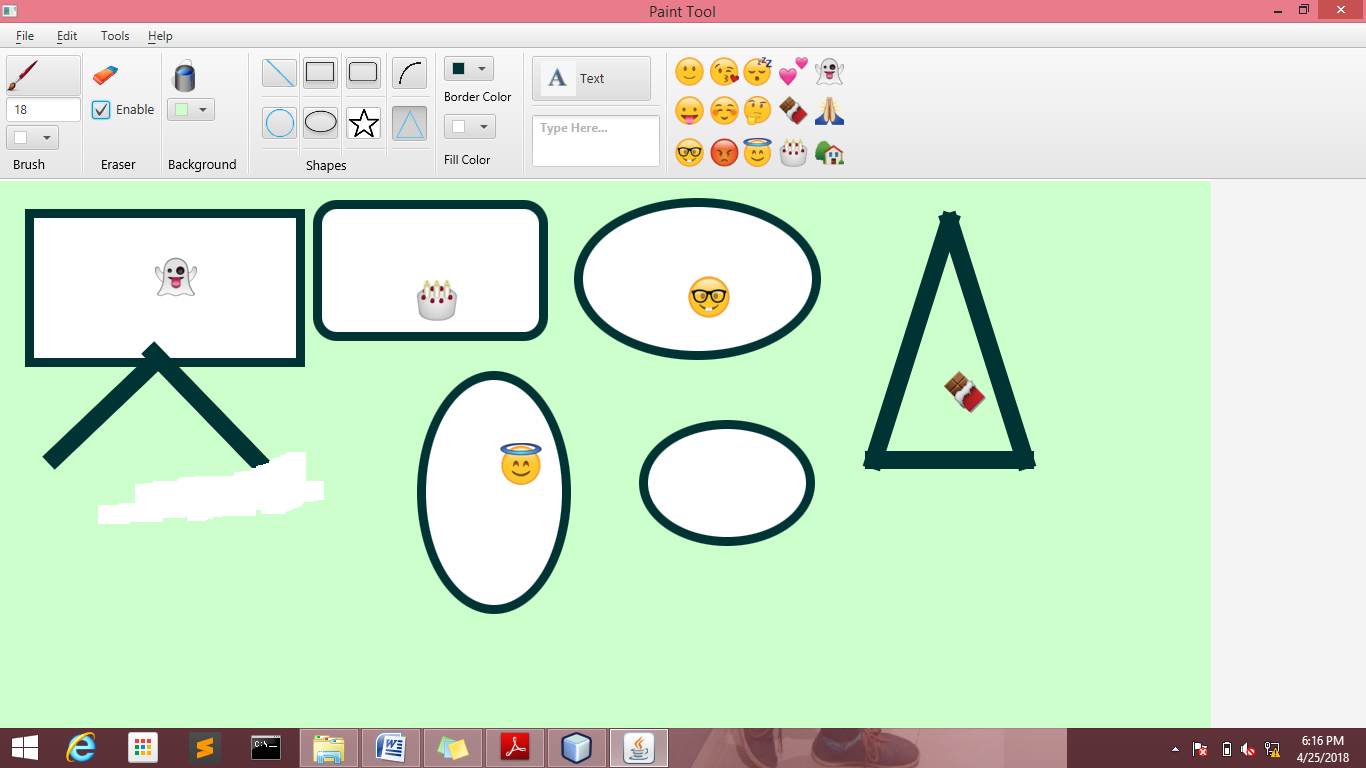


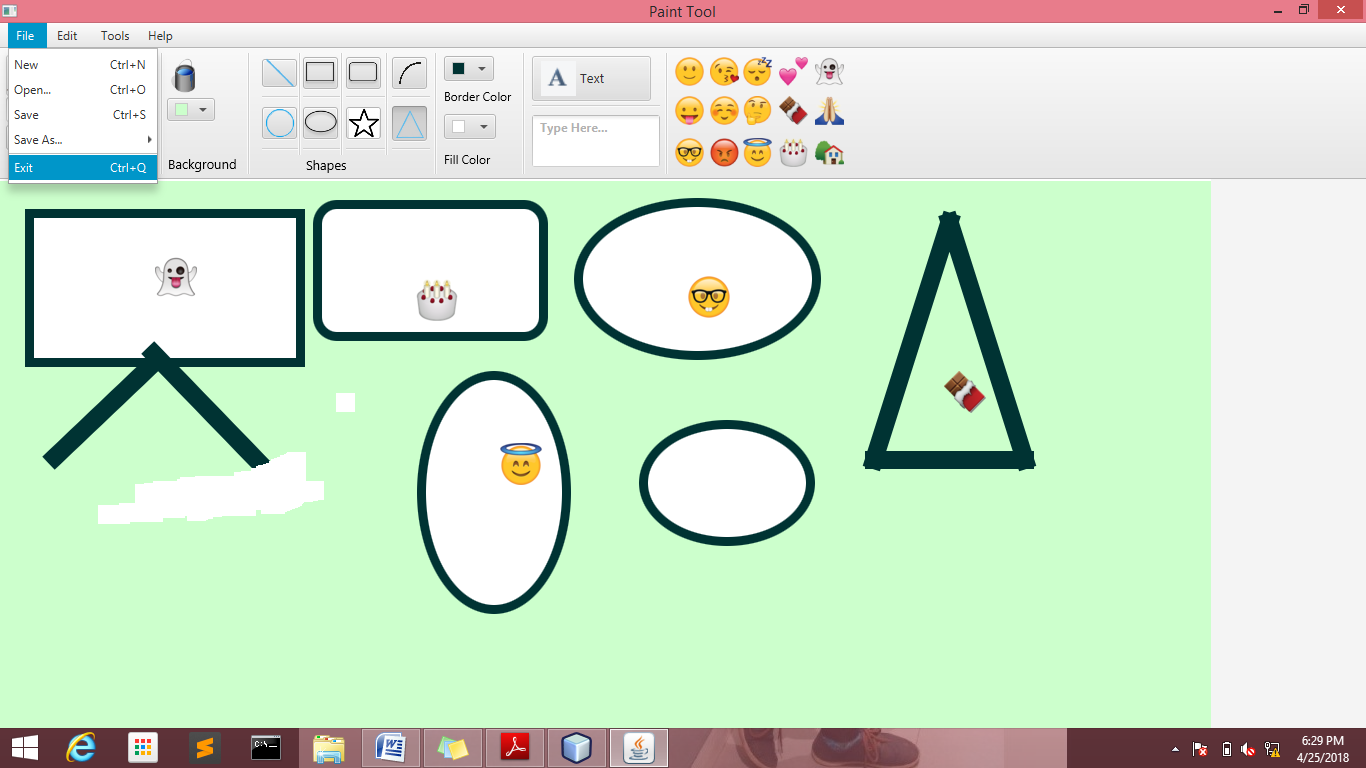












**Chapter - 7**

**CONCLUSION**

This chapter is a conclusion of everything that has been achieved after considering the results from the testing.

**7.1 PROJECT EVALUATION**

* This section will detail whether initial objectives of the project were met and will suggest any future recommendations for the system, the background section discussed the java language, Net Beans and scene builder IDE which was used to implement the system.
* To fully implement a successfully system it was necessary to find out what the system should require. Functional and non – functional user and system requirements were detailed. These requirements showed what the user and system should be able to do and were specified in the requirements specification. Block diagram and descriptions were used to illustrate the functions of the users and the system as a whole.
* In the design of the “Paint Tool” application, a storyboard was designed to show the initial layout of the interface to be developed.
* The implementation of the “Paint Tool” application was carried out using the scene builder toolkit.
* Testing of the system was carried out using the Unit Testing methodology. Each function of the system was taken in turn and tested. During the testing of the system all of the tests conducted were successful.

**7.2 FURTHER EXTENSIONS AND IMPROVEMENTS**

In this project we made drawing tools like rectangle, oval, pen, eraser, brush etc. In our free time or boring time we can use it. This project is based on javaFX.

There are many decisions that would have been changed if I get the chance to do the project again. Different options would have been added given that more time was available. We may include the following:-

* In future we can make this project totally differ from the window based paint or many other kind of painting software.
* We will make this as a kind of application which can also run on our mobile.
* We can use it as our professional work and for entertainment.
* In future we can extend this project by making more tools.
* We can make more polygon tool, other editing tools, filter options and many other tools.
* We may extend it to draw 3D images.
* The program can be made user friendly by adding more help options.

Finally we are able to produce an application that would allow us to draw images using several options provided and save them and retrieve them whenever necessary.

After completing the project many things have been achieved and learnt, and many of them have not been related to software engineering.

Being able to manage and priorities time properly is a positive factor that has been learnt.

**7.3 REFERENCES**

1. Sumita Arora, year of publication2006, Information Practice
2. www.Google.com
3. S.Chand, year of publication 2012, Programming in Java
4. [JAVA][The Complete Reference Java Ninth Edition]