**NITTE MEENAKSHI INSTITUTE OF TECHNOLOGY**

(AN AUTONOMOUS INSTITUTION, AFFILIATED TO

VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM,

APPROVED BY AICTE & GOVT.OF KARNATAKA)

****

**JAVA & J2EE LABPROJECT REPORT (14CSL67)**

**ON**

**BLOOD BANK MANAGEMENT SYSTEM**

*Submitted in partial fulfilment of the requirement for the award of Degree of*

*Bachelor of Engineering*

*In*

*Computer Science and Engineering*

**Submitted by:**

|  |  |
| --- | --- |
| PRAMOD SUDI | 1NT16CS077 |
| VIJAY GOUDA S Y | 1NT16CS131 |
| AKSHAT SINGH | 1NT16CS139 |
|  |  |



**Department of Computer Science and Engineering**

2018-19

|  |  |  |
| --- | --- | --- |
|  | **Nitte Meenakshi Institute of Technology**  (AN AUTONOMOUS INSTITUTION AFFILIATED TO VISVESVARAYA TECHNOLOGICAL UNIVERSITY, BELGAUM)  PB No. 6429, Yelahanka, Bangalore 560-064, Karnataka  Telephone: 080- 22167800, 22167860  Fax: 080 – 22167805 |  |

**Department of Computer Science and Engineering**

**CERTIFICATE**

This is to certify that the Course Project titled “Project title”is an authentic work carried out by **Pramod Sudi (1NT16CS077), Vijay Gouda S Y(1NT16CS131),Akshat Singh(1NT16CS139),** bonafide students of **Nitte Meenakshi Institute of Technology**, Bangalore in partial fulfilment for the award of the degree of ***Bachelor of Engineering*** in COMPUTER SCIENCE AND ENGINEERING of Visvesvaraya Technological University, Belagavi during the academic year ***2018-2019.***

|  |  |  |
| --- | --- | --- |
| **Name Signature of the Faculty Incharge**  MRS. SHOBHA P |  | **Name and Signature of the**  **HOD** |

**DECLARATION**

We hereby declare that

(i) This Presentation does not contain text, graphics or tables copied and pasted from the Internet, unless specifically acknowledged, and the source being detailed in the report and in the References sections.

(ii) All corrections and suggestions indicated during the internal presentation have been incorporated in the report.

(iii) Content of the report has been checked for the plagiarism requirement

Name USN Signature

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

**Date: 11/04/2019**

**ACKNOWLEDGEMENT**

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible, whose constant guidance and encouragement crowned our effort with success. we express our sincere gratitude to our Principal **Dr. H. C. Nagaraj**, Nitte Meenakshi Institute of Technology for providing facilities.

We wish to thank our HoD**, Dr. Thippeswamy M.N** for the excellent environment created to further educational growth in our college.

We also thank our faculty and for the invaluable guidance provided which has helped in the creation of a better project and the technical report. We also thank all our friends, teaching and non-teaching staff at NMIT, Bangalore, for all the direct and indirect help provided in the completion of the presentation.

|  |  |  |
| --- | --- | --- |
| PRAMOD SUDI | 1NT16CS077 |  |
| VIJAY GOUDA S Y | 1NT16CS131 |  |
| AKSHAT SINGH | 1NT16CS137 |  |

Date: 11/04/19

**ABSTRACT**

We have implemented a Blood Bank Management system which uses the concepts of JavaFX and MySQL. The project aims to understand the concepts of using JavaFX tools to create a stage and move to various stages etc. and also to understand the concepts of creation, insertion, updating and deletion of various objects in the MySQL database through Java interface.

The Main aim of this project is to ease the process of blood donation and allowance at blood bank. We aim to demonstrate the use of create, read, update and delete MySQL operations through this project. This project starts by adding details of doctor. Once, registration is done doctor can add blood donor details with blood group, quantity, age etc. Now, when patient is in need of blood, doctors can easily check the blood group available and provide it to the blood receiver. We will provide the addition of doctors separately so as to know the details of doctors available.

**TABLE OF CONTENTS**

**CHAPTER 1 ...................................................................................page-7**

* 1. **Background**
  2. **Key Points**
  3. **History of JavaFX**
  4. **Need for JavaFX**
  5. **Research Objectives**

**CHAPTER 2....................................................................................page-11**

**2.1 Brief Overview/Working of Project**

**2.2 Technology Used 2.3Output**

**CHAPTER 3....................................................................................page-16**

**3.1 Concept of Blood Bank management 3.2Applications used for/by Blood Bank management**

**CHAPTER 4....................................................................................page 17**

**4 Conclusion**

**CHAPTER 5....................................................................................page 18**

**5 References**

# Chapter 1

**INTRODUCTION**

* 1. **BACKGROUND**

When it comes to the Java ecosystem, Swing act as a toolkit for GUI widget. Swing is responsible for providing the APIs, for creating the user interface for Java programs. Before the advent of Swing, AWT i.e. Abstract Window Toolkit was responsible for providing an advanced form of user interface components. But Swing supersedes the AWT library and come up with a look and feel which resemble with most of the platform. These created UI components are not only advance in terms of appearance and feel, but they are also pluggable in nature. That means the underlying platform is not bounded with a specific set of UI components. UI components like buttons, labels, and checkbox can be easily created with Swing APIs. Thus, Swing is more flexible in nature than AWT.

JavaFX act as a standard GUI library, having extensive support for desktop computer and different web browsers on a different operating system like Windows, Linux etc. Desktop applications can be created efficiently using JavaFX, which act as a software platform. In the earlier edition of JavaFX, scripts were being used to build JavaFX applications, these scripts were declarative and static in nature. But with the advent of JavaFX 2.0 version, it is implemented as Java library, means applications now can be written using native Java code instead of scripts. With JavaFX, Java developers can address all the issues which come along with modern UI design. A complex set of controls are required in the modern UI, the responsiveness of UI is highly dependent upon concurrency, but Java multi-threaded code requires a lot of boilerplate code addition.

**1.2 KEY POINTS**

Both Java Swing vs Java FX performance is recommended options in the business. Let us examine some of the key difference between Java Swing vs Java FX:-

* Swing is the standard toolkit for Java developer in creating GUI whereas JavaFX provides platform support for creating desktop applications.
* Swing has a more sophisticated set of GUI components whereas JavaFX has a decent number of UI components available but lesser than what Swing provides.
* Swing is a legacy library which fully features and provide the pluggable UI components whereas JavaFX has UI components which are still evolving with a more advanced look and feel.
* Swing can provide UI components with a decent look and feel whereas JavaFX can provide rich internet application having a modern UI.
* Swing related classes can be found in Java API guide with complete documentation whereas JavaFX doc is available in a various format with a comprehensive detailing and file support.
* Swing since its advent can create UI component using standard Java component classes whereas Java FX initially uses a declarative language called JavaFX script.
* Swing has a UI component library, and act as a legacy whereas JavaFX has several components built over Swing.
* Swing has support for MVC, but it is not consistent across component whereas JavaFX support is very friendly with MVC.
* Swing has various IDEs which offer a tool for rapid development whereas JavaFX has also support from various IDEs as well, but it is not as mature as Swing.
* A swing was renamed from Java Foundation Classes and sun microsystems announced it in the year 1997 whereas JavaFX was initially released in December 2008, by Sun microsystem and now acquired by Oracle.

**1.3 HISTORY OF JAVAFX**

JavaFX Script, the scripting component of JavaFX, began life as a project by [Chris Oliver](https://en.wikipedia.org/wiki/Chris_Oliver_(software_engineer)) called F3. [Sun Microsystems](https://en.wikipedia.org/wiki/Sun_Microsystems) first announced JavaFX at the [JavaOne](https://en.wikipedia.org/wiki/JavaOne" \o "JavaOne) Worldwide Java Developer conference on May 2007. In May 2008 Sun Microsystems announced plans to deliver JavaFX for the browser and desktop by the third quarter of 2008, and JavaFX for mobile devices in the second quarter of 2009. Sun also announced a multi-year agreement with [On2 Technologies](https://en.wikipedia.org/wiki/On2_Technologies) to bring comprehensive video capabilities to the JavaFX product family using the company's [TrueMotion](https://en.wikipedia.org/wiki/TrueMotion" \o "TrueMotion)[Video codec](https://en.wikipedia.org/wiki/Video_codec). Since end of July 2008, developers could download a preview of the JavaFX SDK for Windows and Macintosh, as well as the JavaFX plugin for [NetBeans 6.1](https://en.wikipedia.org/wiki/NetBeans). Major releases since JavaFX 1.1 have a release name based on a street or neighborhood in [San Francisco](https://en.wikipedia.org/wiki/San_Francisco). Update releases typically do not have a release name.

Oracle announced their intention to stop shipping JavaFX with [JDK](https://en.wikipedia.org/wiki/JDK) 11 and later,and it's no longer bundled with the latest version.

**1.4 NEED FOR JAVAFX**

To develop **Client Side Applications** with rich features, the programmers used to depend on various libraries to add features such as Media, UI controls, Web, 2D and 3D, etc. JavaFX includes all these features in a single library. In addition to these, the developers can also access the existing features of a Java library such as **Swing**.

JavaFX provides a rich set of graphics and media API’s and it leverages the modern **Graphical Processing Unit** through hardware accelerated graphics. JavaFX also provides interfaces using which developers can combine graphics animation and UI control.

One can use JavaFX with JVM based technologies such as Java, Groovy and JRuby. If developers opt for JavaFX, there is no need to learn additional technologies, as prior knowledge of any of the above-mentioned technologies will be good enough to develop RIA’s using JavaFX.

**1.5 Research Objectives**

Following are some of the important features of JavaFX −

* **Written in Java** − The JavaFX library is written in Java and is available for the languages that can be executed on a JVM, which include − **Java, Groovy and JRuby**. These JavaFX applications are also platform independent.
* **FXML** − JavaFX features a language known as FXML, which is a HTML like declarative markup language. The sole purpose of this language is to define a user Interface.
* **Scene Builder** − JavaFX provides an application named Scene Builder. On integrating this application in IDE’s such as Eclipse and NetBeans, the users can access a drag and drop design interface, which is used to develop FXML applications (just like Swing Drag & Drop and DreamWeaver Applications).
* **Swing Interoperability** − In a JavaFX application, you can embed Swing content using the **Swing Node** class. Similarly, you can update the existing Swing applications with JavaFX features like embedded web content and rich graphics media.
* **Built-in UI controls** − JavaFX library caters UI controls using which we can develop a full-featured application.
* **CSS like Styling** − JavaFX provides a CSS like styling. By using this, you can improve the design of your application with a simple knowledge of CSS.
* **Canvas and Printing API** − JavaFX provides Canvas, an immediate mode style of rendering API. Within the package **javafx.scene.canvas** it holds a set of classes for canvas, using which we can draw directly within an area of the JavaFX scene. JavaFX also provides classes for Printing purposes in the package **javafx.print**.
* **Rich set of API’s** − JavaFX library provides a rich set of API’s to develop GUI applications, 2D and 3D graphics, etc. This set of API’s also includes capabilities of Java platform. Therefore, using this API, you can access the features of Java languages such as Generics, Annotations, Multithreading, and Lambda Expressions. The traditional Java Collections library was enhanced and concepts like observable lists and maps were included in it. Using these, the users can observe the changes in the data models.
* **Integrated Graphics library** − JavaFX provides classes for **2d** and **3d** graphics.

# Chapter 2

**2.1 Brief Overview/Working of Blood Bank Management system**

The Blood Bank Management System is an interactive user interface where blood banks can maintain a database that holds the details of donors who donated the blood, doctors available for donation and also details of the quantity of blood available for each blood group at any given point of time. Here, at first the details of doctors are added, then details of donors ready to donate are added, when there is need of blood, we can check the quantity of blood available for each blood group and according to it, if ready can take details of the receiver. At any point of time, updating and deleting the details are possible.

**2.2 Technology Used:**

JavaFX and MySQL.

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. MySQL an open source Relational database management system is used.

**Advantage/Disadvantages of the Technology used:**

**JavaFX**

**Advantages:**

1.JavaFX is built-in Java 8 and above and offers a much simpler way to develop GUI code (more features and less complexity than Swing).

2.FXML (similar to xml or html )

3. CSS (for animation and style)

4.Drag and Drop Builder (scene builder)

5.It can be embedded in swing application.

**Disadvantages:**

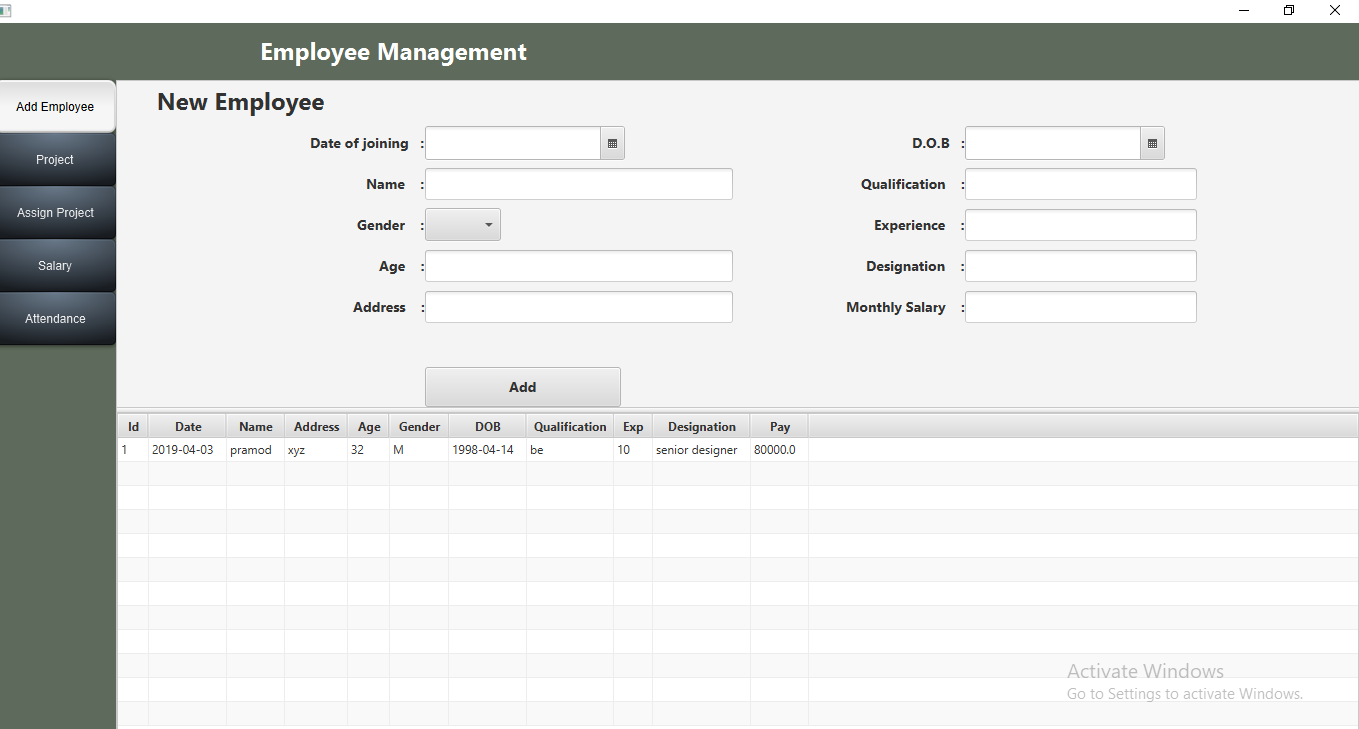
1.wouldn't feel comfortable developing a Web-based application for JavaFX, just because you can't assume that the user has the framework already installed on their machine. If they don't, you don't necessarily know if the user has permissions on their machine to even download and install the latest version of Java.

2.It is very immature technology. While everything, as others have noted, must have a start somewhere, JavaFX is very immature technology *in a field that has several more mature competitors*. That makes all the difference. Flex/AIR and Silverlight are out there already and in active use all over the place. To tackle an entrenched market you need tools that are far superior to what's out there, not just a little superior (if at all).

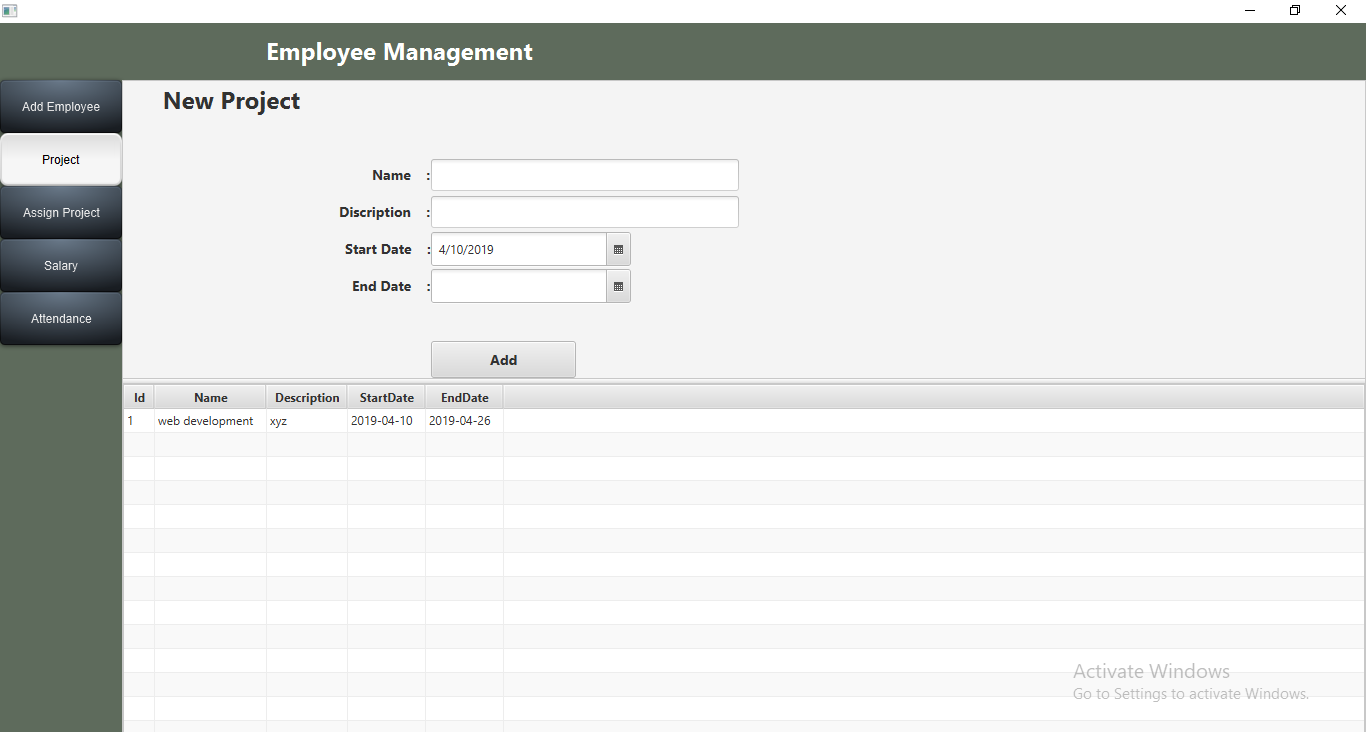
3.It is stacked on top of the rather large JRE. The JRE is available on every major platform (and many minor ones) but it is not ubiquitously installed. The same is true of Flex/AIR and Silverlight, of course, but Flash is a lighter-weight solution than the whole JRE+JavaFX and the latter is basically a default on the target platform anyway.

4.JavaFX is heavily reliant upon the huge infrastructure that surrounds Java. This is a strength, of course, given that if you need it there's a library for it somewhere, but it is also a weakness in that it involves even more bloat than the JRE+JavaFX.

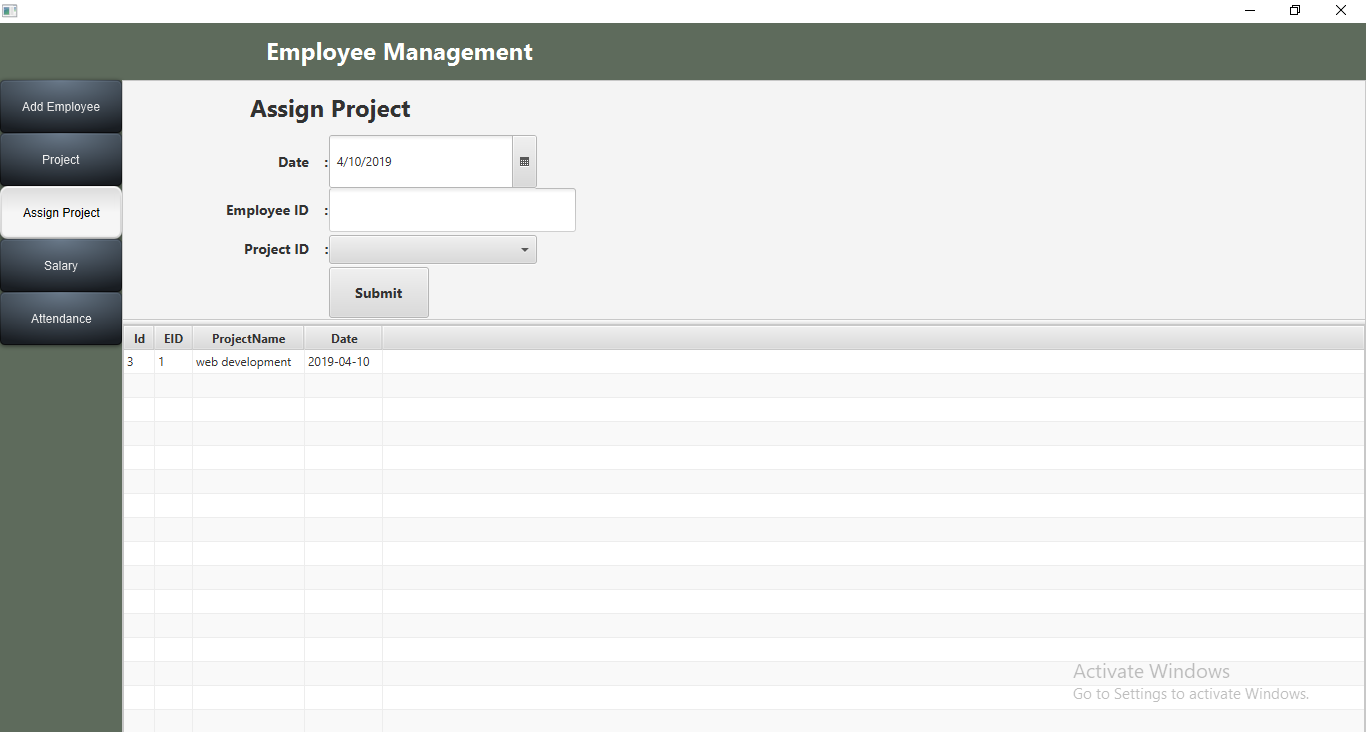
**2.3 OUTPUT:-**

****

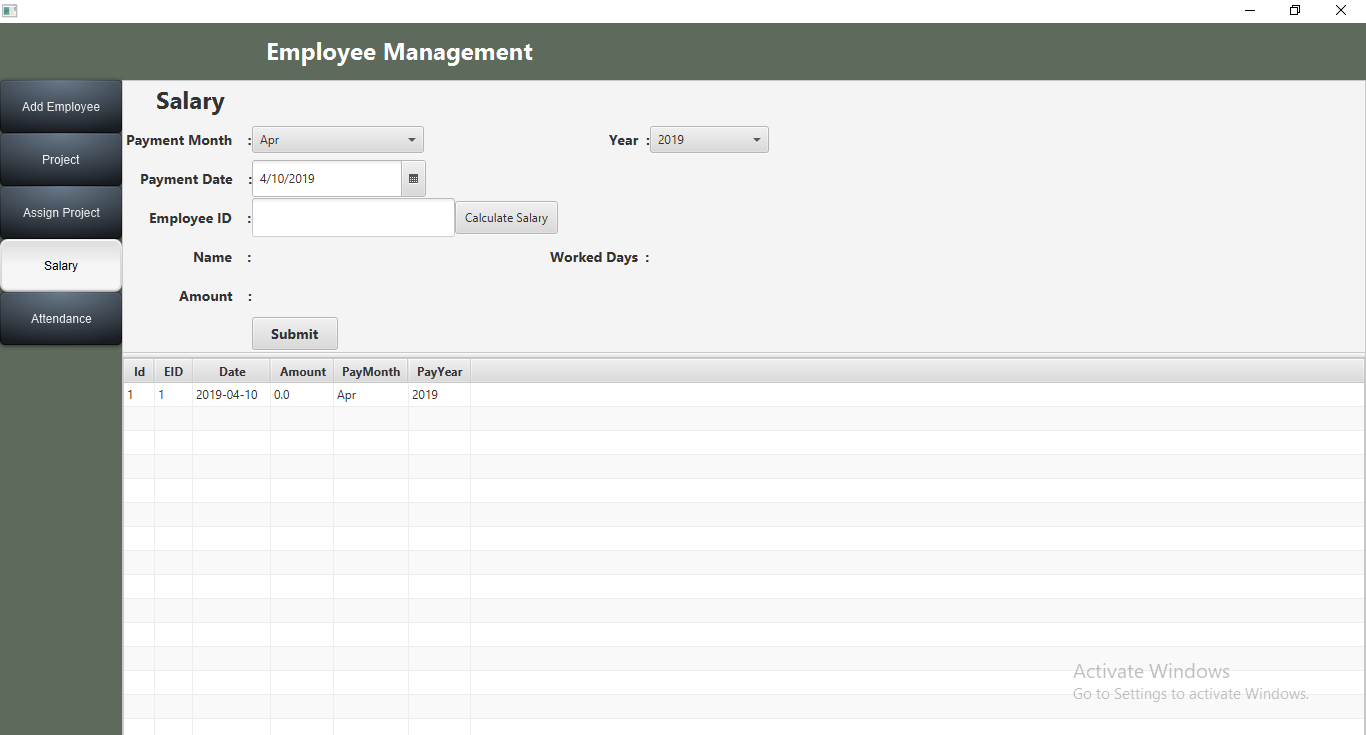
**Fig 2.3.1**

****

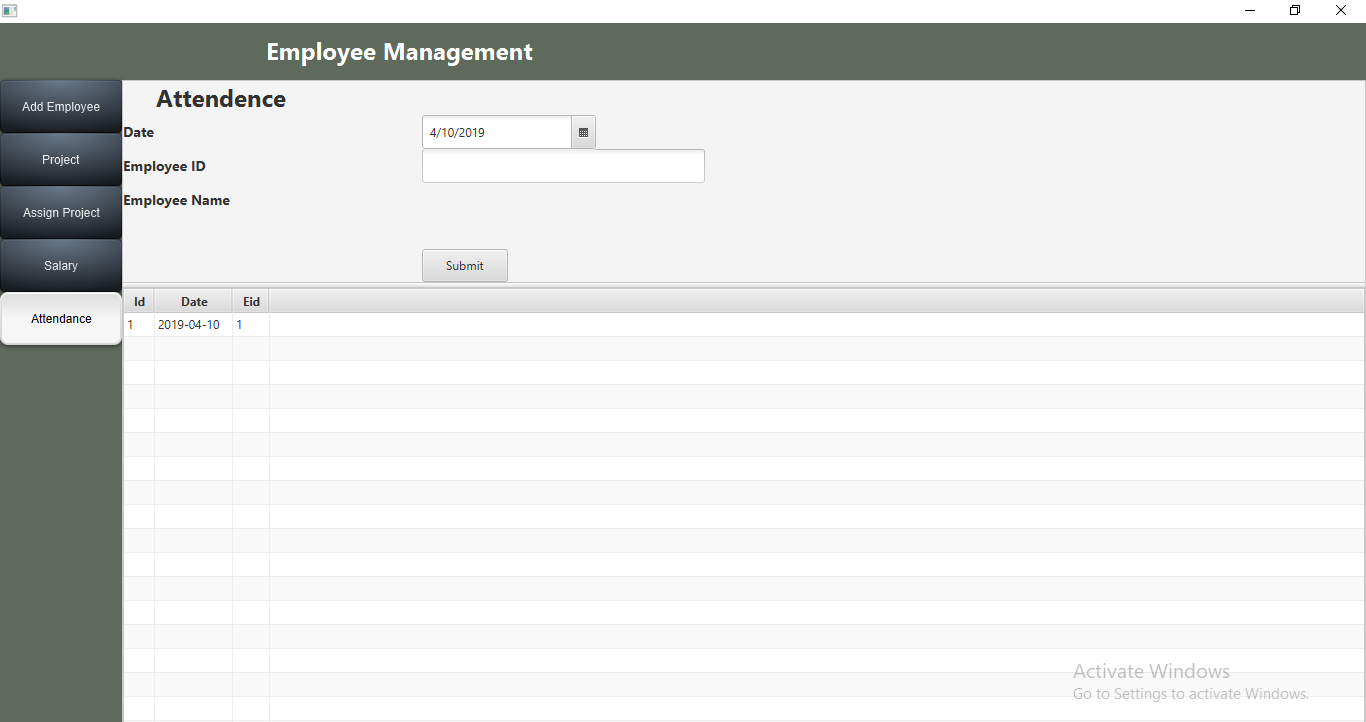
**Fig 2.3.2**

****

**Fig 2.3.3**

****

**Fig 2.3.4**

****

**Fig 2.3.5**

**Chapter 3**

**3.1 Concept of Blood Bank Management System.**

Any blood bank needs to maintain the details of the activities that takes place as a part of blood donation. This is important to avoid the mistake of wrong donation and also loss of data. So, our project aims to eliminate this by creating an interactive user interface that holds and manipulates data of every possible aspect of blood bank. This systematic management allows for ease of blood donation and proper working of blood bank.

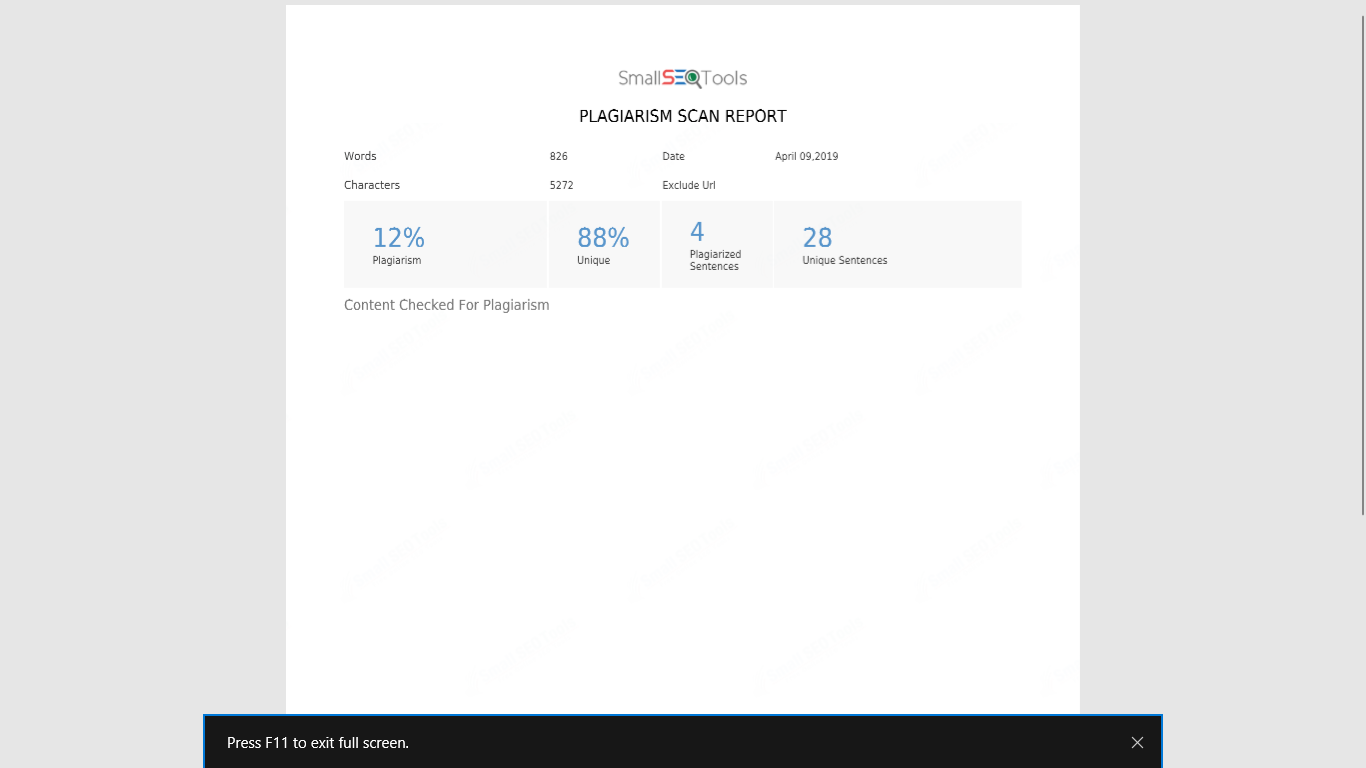
Modules of Blood Bank Management System

1. Donor: Details of donor donating blood.
2. Blood Receiver: Details of patients in need of blood.
3. Doctor Details: Details of doctor managing blood bank.
4. Employee: Employee details managing blood bank patients.
5. Available Blood: Details of blood groups available in blood bank.

**3.2 Applications of project**

Blood banks have a need for efficient, long-term storage of records and quick access to the stored information. Database management systems make it easy to retrieve and review large amounts of data. They provide such functions as data entry, searching capability, update and delete operations, report availability in one file to those in another file. Privately developed programs can be shared by blood banks. Doctors can easily access this data and allow blood donation with ease.

**Chapter 4: Conclusion and Future work**

JAVAFX is well suited to support software for any management system, and the project was manageable given the tools we’ve accrued throughout the semester. Understanding javafx tools, database connection, and if/else statements was essential for creating this project. Debugging was necessary throughout the coding process, but often the problems we encountered required only logic to understand how our code had room for misinterpretation. We could further improve the project by also maintaining the details of the destination where blood reaches in case it is transferred to any other blood bank. We can create a workspace for donors, once donation is done they can access as to where his/her blood is reached.

**Chapter 5: References**

1. Java - The Complete Reference, 9th Edition - Herbert Schildt.
2. [www.wikipedia.com/](http://www.wikipedia.com/) Requirements of management system.
3. [www.tutorialspoint.com/javafx/](http://www.tutorialspoint.com/javafx/)
4. [www.javatpoint/javafx-tutorial/](http://www.javatpoint/javafx-tutorial/)
5. Youtube.com/javafx for beginners.