

OBJECT ORIENTED PROGRAMMING

Assignment-2

Group 03 members: UIT/2019/FEB/0060- Rochana Fernando

USE/2019/FEB/0062- Shalan Dulanja

UIT/2019/FEB/0064- Harshani Cooray

USE/2019/FEB/0066-Tharindu Dhanushka

Lanka Nippon Bizatch Institute

Maharagama

2019/12/19

Contents

[**1.** **Introduction** 1](#_Toc31918893)

[**2** **Task-02** 2](#_Toc31918894)

[2.1 Discuss the features in Java programming 2](#_Toc31918895)

[2.2 Evaluate JVM environment and its environmental flexibility in building machine independent java programs 5](#_Toc31918896)

[**3** **Task-03** 8](#_Toc31918897)

[3.1 UML Diagrams 8](#_Toc31918898)

[3.1.1 Use case Diagram 8](#_Toc31918899)

[3.1.2 Class Diagram 9](#_Toc31918900)

[3.2 Use case narratives 10](#_Toc31918901)

[3.3 Database Diagram 16](#_Toc31918902)

[3.3.1 ER Diagram 16](#_Toc31918903)

[3.3.2 Write a short description in plain English for each entity in ER Diagram. 17](#_Toc31918904)

[**4** **Task-04** 18](#_Toc31918905)

[4.1 Implement java solution 18](#_Toc31918906)

[4.1.1 Define relationship between objects and object-oriented programming methods concepts which are used in the project implementation. 18](#_Toc31918907)

[4.2 Control structures used in the project implementation. 23](#_Toc31918908)

[4.3 Error handling and reporting 24](#_Toc31918909)

[**5** **Test and document java solution** 25](#_Toc31918910)

[5.1 Test plan and test cases 25](#_Toc31918911)

[5.1.1 Test plan 25](#_Toc31918912)

[5.1.2 Test cases and result 27](#_Toc31918913)

[6 Testing methods 35](#_Toc31918914)

[7 User manual for INC Company 40](#_Toc31918915)

[7.1 General Information 40](#_Toc31918916)

[7.1.1 System overview 40](#_Toc31918917)

[7.2 System summary 41](#_Toc31918918)

[7.2.1 Configuration 41](#_Toc31918919)

[7.2.2 Access levels 41](#_Toc31918920)

[7.3 System functions 42](#_Toc31918921)

[7.4 Equipment and maintenance 42](#_Toc31918922)

[8 Technical document for INC Company 43](#_Toc31918923)

[8.1 Introduction 43](#_Toc31918924)

[8.2 System overview and system design 43](#_Toc31918925)

[8.3 Hardware requirements 48](#_Toc31918926)

[8.4 Software requirements 48](#_Toc31918927)

[8.5 Database design 48](#_Toc31918928)

[8.6 System security 50](#_Toc31918929)

[8.7 Roles and responsibilities 51](#_Toc31918930)

[9 References 52](#_Toc31918931)

[**10** **Individual contribution** 53](#_Toc31918932)

# **Introduction**

Java language is an Object-Oriented programming language. It is mainly depending on classes and objects.

There are main features of Java language such as,

1. Object Oriented

2. Simple and Familiar

3. Robust and Secure

4. Architecture-neutral

5. High performance

6. Interpreted

7. Dynamic

Java language is compiled through the representation of Java byte code. It is executed by the Java Virtual Machine (JVM). There are primitive data types in Java language. Boolean, Char, Long, Short, String, Float and Double. There are many types of Java Applications such as Java applets, Java Servlets, Java Swing application, JavaFX application etc. Java language can be used or supplies with Oracle and Swing application, JavaFX application etc. Java language can be used or supplies with Oracle and Android. There are main four concepts of Object-Oriented programming. **Abstraction, Encapsulation, Inheritance and Polymorphism** [1].

Java VS C++

|  |  |
| --- | --- |
| **Java** | **C++** |
| **1.Compiler and interpreter both are used. So, it is Platform independent.** | **1.Only uses the compiler. So, it is Platform dependent.** |
| **2.Pointers are not used.** | **2. Pointers are used.** |
| **3.Does not support Multiple Inheritance.** | **3.Provides Multiple Inheritance.** |
| **4.Mainly used for application programming** | **4.Used for system Programming.** |
| **5. Supports in documentation comment** | **5.Does not support for comments in documentation.** |

# **Task-02**

## Discuss the features in Java programming

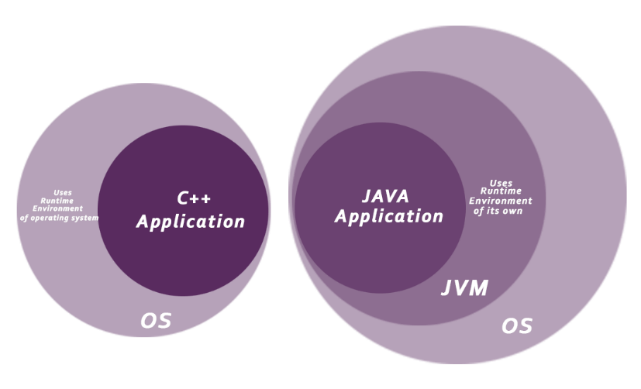
Java is a simple, secure and portable programming language. There are several excellent features which are performed an important role in java programming. These features have been caused for the popularity of this programming language.

**Simple**

As a popular programming language java is very easy to learn because the syntaxes are very simple. It is easy to because java has removed very complicated and rarely used features (ex: operator overloading). Java is Easy to compose and more readable and eye getting. Java has a compact, durable arrangement of highlights that makes it simple to learn and utilize. The vast most of the concept’s ideas are drew from C++ in this way making Java learning less complex. Java considers as a simple programming language because of there is no need of removing unreferenced objects.

**Secure**

Java program harm other system and framework accordingly making it secure. Java gives a safe method for making Internet applications. Java gives secure approach to get to web applications. Java does not enhance express pointers because clear pointers were considered a security risk. There is still a verifiable pointer, but the client cannot characterize the pointer, so the language can be gradually made safe. By using java, we can develop virus-free system. Class loader, bytecode verifier and security manager are some other features which are increased security of java [2].

.Figure 1:JVM Secure

**Portable**

Java projects can execute in any condition for which there is a Java run-time system. (JVM). Java projects can be run on any stage (Linux, Window, Mac). Java programs can be transferred on the World Wide Web. Java Byte code can be carried on any platform. There are no implementation-dependent features.

**Object-oriented**

Java is a fully object-oriented language. In java, every functionality and all operations are based on objects. An object consists with both attributes and properties. We use object-oriented method to arrange our software program as an aggregate of different forms of items that incorporates both statistics and behavior (methods or functions). Like C++ java gives the majority of the object-oriented features. Followings are the basic object-oriented concepts which ae used in java;

Ex: Object, class, inheritance, polymorphism, abstraction, encapsulation

**Platform independent**

Not like other programming languages such as C, C++ java doesn’t need any specific machine platform to compile the code. So, code which is written java can run anywhere without any restriction.

**Robust**

Robust honestly ability strong. Java makes use of strong memory management. There is lack of pointers that avoids safety problem. There is automated garbage collection in java. There is exception coping with and kind checking mechanism in java. All these factors make java robust. But the principal areas which Java extended were mishandled Exceptions by way of introducing automated Garbage Collector and Exception Handling. Java encourages error-free programming by means of being strictly typed and performing run-time checks.

**Multithreaded**

Java offers built-in help for multithreaded programming. Threads are impartial small applications or subprograms of a program.java allows to write java programs that deals with many tasks at once by defending multiple threads. When these threads are run concurrently, the process is called multithreading. The importance of the multi threads is it doesn’t take separate memory for each thread and it shares a common memory space to perform it tasks.

**High-performance**

Java is an interpreted language, so it will by no means be as quick as a compiled language like C or C++. But Java allows high performance with the use of just-in-time compiler.

**Distributed**

Java language is designed for the allotted environment of the internet. We may also get admission to files via calling the strategies from any computing device on the internet by using java.

**Architecture Neutral**

We have seen in C and C++ that the measurement of integers, go with the flow and other variables varies in accordance to the architecture of the device in bits. Java has the same dimension of any type of variable across all possible architectures.

## Evaluate JVM environment and its environmental flexibility in building machine independent java programs

**What is JVM**

The JVM is stands for a software program that executes code and provides the runtime environment [1]

JVM is stands for Java Virtual Machine. JVM consider as a part of JVM (Java Run Time Environment) because it acts as a run time engine to run java applications. The specialty of java is java compiler generate bytecode for the JVM but in other programming languages are generated byte code and directly connect with the system. Byte can be considered as an intermediary language between java source and host system. This byte code can be used to run in any flatform because it makes its own platform to interpret on a different machine [4].

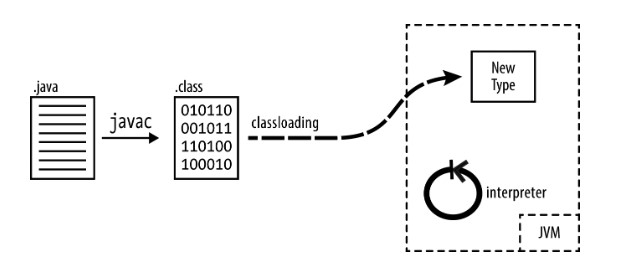


Figure 2:how JVM works

Figure 3:jvm

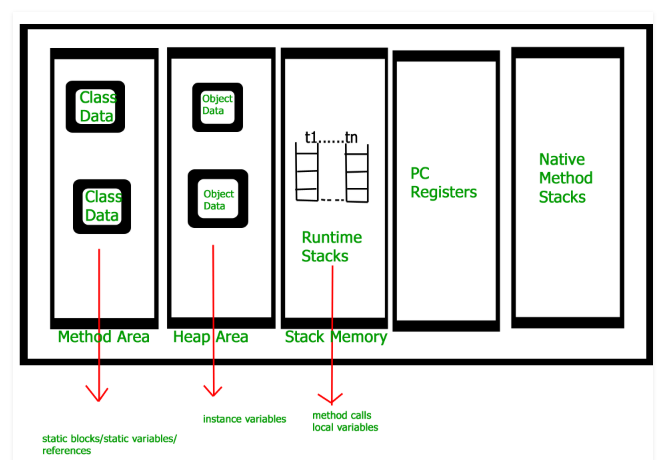


Figure 4:JVM architecture

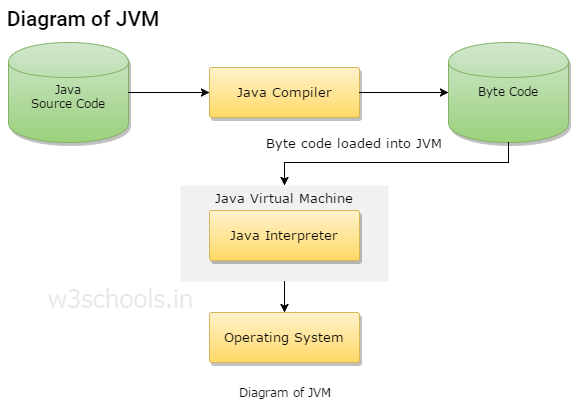


Figure 5:JVM diagram

**How JVM works**

There are three main activates performed by the JVM.

1. Loading bytecode
2. Linking the code with library
3. initialization

* Loading bytecode

Class loader reads the class file and generates the byte code and save it inside a method.

Linking the code with library.

* ‘Linking’ is consists with 3 actions
  + Verification – ensure the correctness of the byte code (ex: check whether files are properly performed)
  + Preparation – allocates memory for class variables and initialized the memory to the default values.
  + Resolution – search method areas
* Initialization

Assign values to static variables an execute the code from top to bottom.

**JVM flexibility in building machine independent java programs**

A platform is the hardware or software environment which a java program runs. There are two types of software based and hard ware based. Java provides software-based platform and it runs on the hardware-based platforms. it has two components:

1. Runtime environment
2. API (Application Programming Interface)

As above mentioned, compiler compiles the java code and convert it in to bytecode. JVM provides performing environment to byte code and byte code is a platform-independent code because it can be run on multiple platforms. Because of the JVM and the byte code java has become a programming language which proof “once write and Run any ware” concept.

# **Task-03**

## UML Diagrams

### Use case Diagram

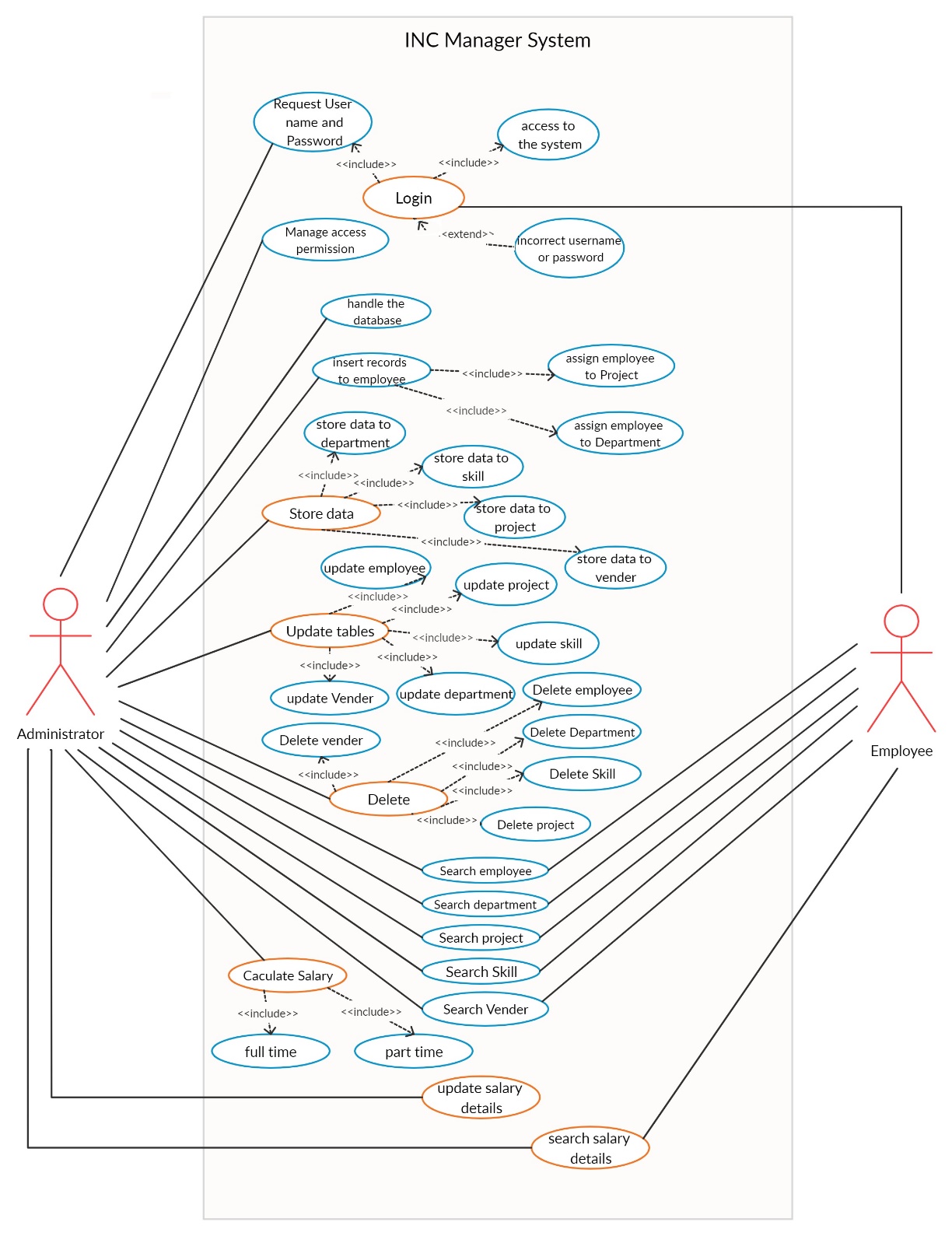


Figure 6:Use Case Diagram

### Class Diagram

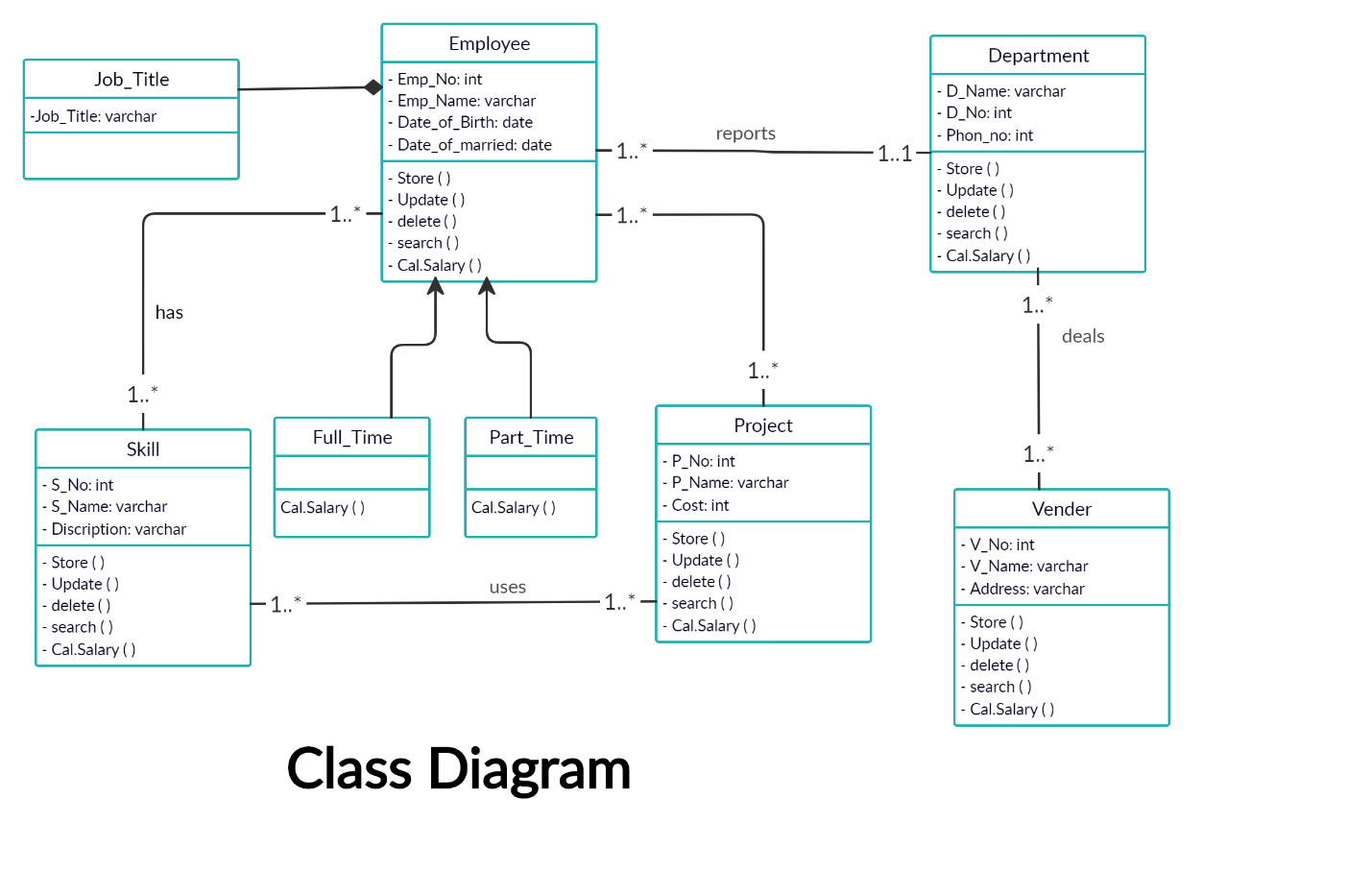


Figure 7:Class Diagram

## Use case narratives

Select three use cases and provide use case narratives for the selected use cases.

1.Login

Pre-condition: user must have

* + - * Username
      * Password

Post-condition: Access to the INC Company Page

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1.Access to the login page |  |
|  | 2.Dispaly the login form |
| 3.Enter username |  |
| 4.Enter password |  |
| 5.click on login button |  |
|  | 6.get entered username and password and send it to database |
|  | 7.check whether username and password correct or not |
|  | 8. if both are correct give access to the INC company page |
|  | 9.if one of username or password or both are incorrect popup an error message on display. |

Alternative flow

A1: When user enter invalid username or password more than 3 times

Popup an error message “invalid username and password”

Revoke permission to enter username and password

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

2.Add

Pre-condition: user should be logged to the system

Post-condition: All data should be insert into the database

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1.enter the all data which are corresponded to each text box |  |
| 2.click on the add button |  |
|  | 3. get all entered data and send it to database |
|  | 4.Inserte data into table in the database. |
|  | 5.popup a message “recodes are inserted successfully” |

Alternative flow

A1: When all data will not insert properly or any inconsistency with data

Popup an error message “all fields are not inserted”

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

3.Update

Pre-condition: user should be logged to the system

Post-condition: requested records of database should be updated

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1.enter data into corresponded fields which should be updated |  |
| 2.click on the update button |  |
|  | 3.first check that is there any id in the database which is similar to id which we entered trough the interface. |
|  | 4. get all entered data and send it to database |
|  | 5.update records in the table in the database. |
|  | 6.popup a message “recodes are updated successfully” |

Alternative flow

A1: When ID will not match with recodes in the database

Popup an error message “ID cannot be found”

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

4.Search

Pre-condition: user should be logged to the system

Post-condition: searched data should be viewed in form

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1.enter the id which we want to search records |  |
| 2.click on the search button |  |
|  | 3. first check that is there any id in the database which is similar to id which we entered through the interface. |
|  | 4.get data from the database which are corresponded to given ID |
|  | 5.display data on grid view |

Alternative flow

A1: When ID will not be found

Popup an error message “ID cannot be found”

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

5.Delete

Pre-condition: user should be logged to the system

Post-condition: delete records from the database which are corresponded to given ID

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1.enter the ID which we want to delete records |  |
| 2.click on the delete button |  |
|  | 3.first check that is there any id in the database which is similar to id which we entered through the interface. |
|  | 4.delete recode which are corresponded to the ID |
|  | 5.popup a message “recodes are deleted successfully” |

Alternative flow

A1: When ID will not be found

Popup an error message “ID cannot be found”

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

6. Clear button

Pre-condition: user should be logged to the system

Post-condition: clear data in the textbox

Normal flow

|  |  |
| --- | --- |
| User Action | System Action |
| 1. click on the clear button |  |
|  | 2.clear data in the text boxes |

Alternative flow

A1: When the button click event will

System should be allowed to clear textboxes values by using keyboard (delete or backspace button)

Exception flow

E1: When internal code doesn’t exist or if there is any connection error with debase connectivity

Popup an error message “a system error has been accrued please contact your system administrator”

## Database Diagram

### ER Diagram

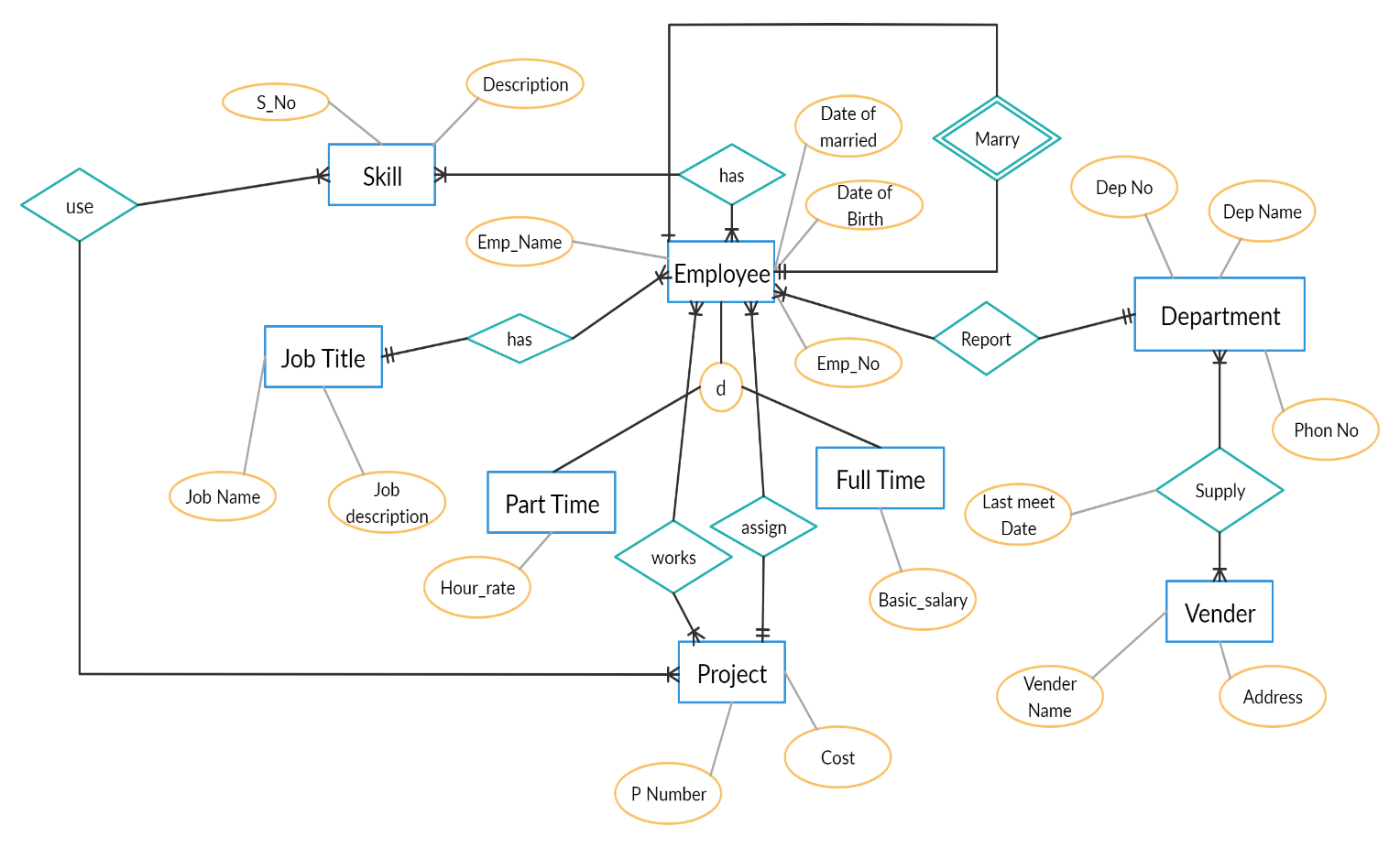


Figure 8:ER diagram

### Write a short description in plain English for each entity in ER Diagram.

**Employee**

In the INC Company, each employee is identified with a unique number, name, Date of Birth

If an employee marries other in the same company, we have to store the Married Date. Each employee works as a full-time or part-time worker in the company.

Primary Key of this entity is the Employee ID.

**Job Title**

Each employee has a job title. It is recognized by a job-ID and a job-name. Primary Key of this entity is the Job-ID.

**Department**

Each employee has a department. It is identified by a unique number, name and a phone number. Each department deals with many vendors. Primary Key of this entity is the Department Number.

**Vendor**

Vendor Supplies equipment to many departments. It is recognized by a unique number, name and an address. Date of the last meeting department and the vendor should be stored. Primary Key of this entity is the Vendor’s number.

**Job Skill**

Each employee has a special skill. He/she can use their skill on projects. Job skills are identified by a number, name and a description. Primary Key of this entity is the unique number.

**Project**

An employee can work on many projects. But one employee can be assigned to one project at a given time. A project has a unique number, name and cost of each project in the company. Primary Key of this entity is the project number.

# **Task-04**

## Implement java solution

### Define relationship between objects and object-oriented programming methods concepts which are used in the project implementation.

* **relationship between objects**

mainly there are 3 types of cardinalities. In this project consists with all these types.

1. one to one relationship

ex: an employee marries another employee who works in INC company

1. one to many relationship

ex: many employees are worked for one department

many employees can have same job title

1. many to many relationship

ex: many skills can be used for many projects

many employees work on many project

many vendors are supplied equipment to many departments

* **object-oriented programming methods**

There are basic components of object-oriented programming

1. Class

A class is a group of objects which have similar behaviors and attributes. Class is a logical entity and it can be known as blue print of objects.

According to INC company system, there are many classes such as;

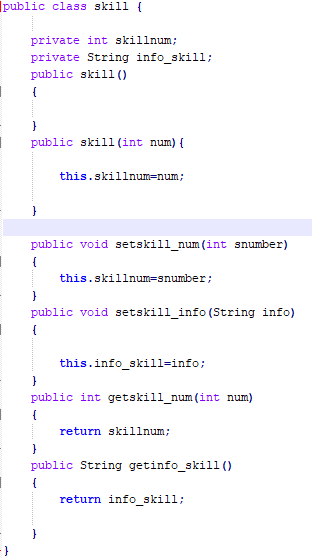
Ex: employee class

Department class

Project class

Vender class

Skills class



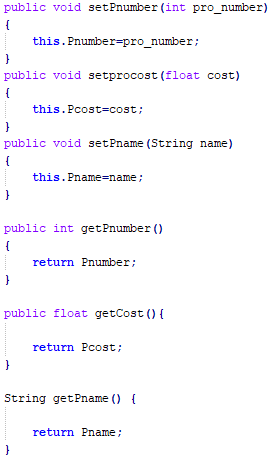
1. Object

An object can be defined as an instance of a class. Object is a real-world thing which is consists with attributes and behaviors. It can be tangible or nontangible. In INC company system also have both tangible objects like employee, department and nontangible objects like skill, job title.

1. Methods

A block of code which only run when it is called. Methods can be defined as the behaviors of a class. If someone wants to call a method it should be created an object from particular class and call the method through object.

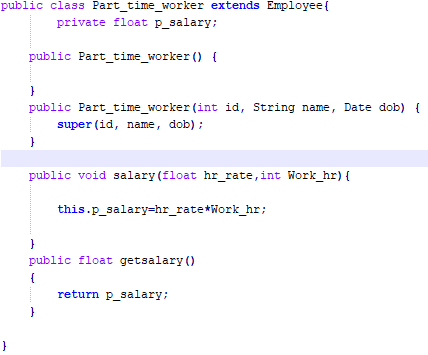
Ex: method uses to get employee name



1. Inheritance

Inheritance means creating a parent-child relationship between two classes. We use ‘extends’ keyword to defined inherit classes. When two classes are inherited, all attributes and behaviors which are corresponded to the parent class will come to the child class.

Ex: full time worker and part time worker classes inherit employee class



1. Abstraction

Abstract class refers to the ability of a variable, object or function to take on multiple forms. It is defined by using abstract keyword. Abstract classes can have abstract or non-abstract methods but if any class consists with abstract method by default that class became an abstract class. We cannot create objects from an abstract class directly.

1. Polymorphism

an act of representing essential features without including background details. There are two main concepts of polymorphism,

1.overiding- same name same signature (both parent class and child class are having method in same name and same para meters)

2.overloading-same name different signature (both child and parent classes are having method in same name but different parameters)

Ex: part time employee class overrides salary calculation method in employee class

1. Encapsulation

Technique of wrapping the data and code. To hide data and methods we use access modifiers,

* Private- can be accessed only inside the created class
* Public-can be accessed anywhere
* Protected-can be accessed inside the same package and only child class in another package by using child object
* Default- can be accessed only inside the same package

1. association

Relationship between two objects. It can be one to one, one to many or many to many relationships.

1. aggregation

All objects have their separate lifecycle. Changes of another class or objects will not be affected.

1. Composition

Child objects do not have their lifecycle so when parent object deletes. Changes of parent class and performance of objects which are made by using parent class will be directly affected on the child class and child objects.

## Control structures used in the project implementation.

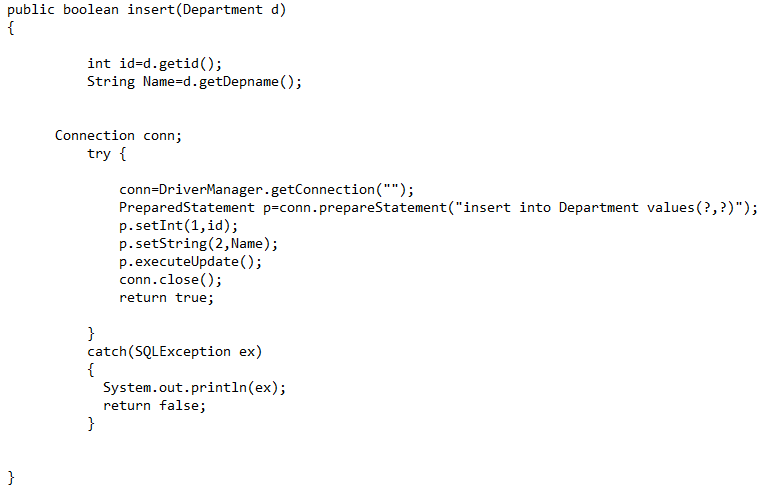
Ex: Using loop to check condition in login page



## Error handling and reporting

Ex: exceptional handling

Ex: department



# **Test and document java solution**

## Test plan and test cases

### Test plan

**INC Engineering Company manager system**

|  |  |
| --- | --- |
| Test plan ID | 01 |
| Brief introduction about the system | This is a manager system for INC engineering company. The system requires to manage all employees (full time and part time employees), their skills, project assign and departments.  Also, the system requires to calculate salary of the employees and the system can be performed store, update, delete and search employee functionalities including the salary details, skills, projects, departments and vendors. |
| Introduction to the testing objectives | Testing objectives of this system are,   * Identify the features should be tested * List the recommended test requirements * Recommend and describe the testing strategies to be employed * Identify defects and correct them * List the deliverable elements of the test activities |
| Testing items | Interface  Functionalities  Database connectivity |
| Features to be tested | Interface   * Login * Textboxes * Buttons   Functionalities   * Curd operations * Insert * Update * Delete * Search * Salary calculation * Full time employee’s salary calculation * Part time employee’s salary calculation   Database connectivity |
| Features not to be tested | Design of interface will not be tested (ex: colours, font styles) |
| Test environment | <tester 1>PC, visual studio 2019 and Microsoft server management studio 2017. |
| Test approach | Black box testing  Whitebox testing |
| Testing task | Test planning, test designing, test development, test execution, test evaluation |
| Test deliverables | Test plan, test environment, test summery, test result, test evaluation report |
| schedule | 18th December 2019 8.00 a.m. to 12.00 p.m. |

### Test cases and result

TEST CASES AND TEST RESULT

|  |  |
| --- | --- |
| **Test case ID** | 1 |
| **Test Unit** | Login page |
| **Test Type** | Black box |
| **Tester** | R. Fernando |
| **Test Designer** | V. Fernando |
| **Venue** | LNBTI |
| **Date** | 2019/12/18 |
| **Time** | 12.00 a.m. – 1.00 a.m. |
| **Number of test cases** | Cases - 4   * 1. test username textbox   2. test password textbox   3. test ‘login button’ click   4. test ‘clear button’ click |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Teat case | Test step | Test data | Expected results | Actual result | Status (Pass / Fail) | Notes |
| 1.1 | Insert userid | 4321 | Textbox should be taken the given value according to given restrictions. If not, it should popup an error message. | Textbox get the value. | pass |  |
| 1.2 | Insert password | tester@123 | Textbox should be taken the given value according to given restrictions. If not, it should popup an error message. | Textbox get the value. | pass |  |
| 1.3 | ‘login button’ click event |  | Button can be clicked  When click the login button if both username and password are correct it should give access to the INC Company page. | Access to the INC company page | pass |  |
| 1.4 | ‘clear button’ click event |  | Button can be clicked  When click the clear button, it should clear the selected text area in relevant test box. | Clear the selected area of the text box | pass |  |

TEST CASES AND TEST RESULT

|  |  |
| --- | --- |
| **Test case ID** | 2 |
| **Test Unit** | INC Company Form |
| **Test Type** | Black box |
| **Tester** | R. Fernando |
| **Test Designer** | V. Fernando |
| **Venue** | LNBTI |
| **Date** | 2019/12/18 |
| **Time** | 12.00 a.m. – 1.00 a.m. |
| **Number of test cases** | Cases - 5  2.1textboxes  2.2test ‘insert button’ click  2.3test ‘update button’ click  2.4test ‘delete button’ click  2.5test ‘search button’ click |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Teat case | Test step | Test data | Expected results | Actual result | Status (Pass / Fail) | Notes |
| 2.1 | Text boxes | ID: 001 | Textbox should be taken the given value according to given restrictions. If not, it should popup an error message. | Get the value | pass |  |
| 2.2 | ‘insert button’ click event | ID:003  Name: Amal | Button can be clicked  When click the insert button, it should access the insert code and pass  Passes relevant values to the code.  Popup a message insert is successful or not. | should access the insert code | pass |  |
| 2.3 | ‘update button’ click event | ID:003  Name: Amal | Button can be clicked  When click the update button, it should access the update code and pass  Passes relevant values to the code.    Popup a message update is successful or not. | access the update code | pass |  |
| 2.4 | ‘delete button’ click event | ID:003 | Button can be clicked  When click the delete button, it should delete the records according to the given ID.  Popup a message deletes successful or not. | delete the records according to the given ID. | pass |  |
| 2.5 | ‘search button’ click event | ID:003 | Button can be clicked  When click the search button, it should show relevant selected data in relevant textboxes.  If cannot search the recodes it should popup a message “not found” | shows relevant selected data in relevant textboxes. | pass |  |

TEST CASES AND TEST RESULT

|  |  |
| --- | --- |
| **Test case ID** | 3 |
| **Test Unit** | Login page |
| **Test Type** | White box testing |
| **Tester** | R. Fernando |
| **Test Designer** | V. Fernando |
| **Venue** | LNBTI |
| **Date** | 2019/12/18 |
| **Time** | 12.00 a.m. – 1.00 a.m. |
| **Number of test cases** | Cases – 4  3.1 Right username and Right password  3.2 Right username and Wrong password  3.3 Wrong username and right password  3.4 Wrong username and Wrong password |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Teat case | Test step | Test data | Expected results | Actual result | Status (Pass / Fail) | Notes |
| 3.1 | Right username and Right password | 4321  tester@123 | Login success | Login success | pass |  |
| 3.2 | Right username and  Wrong password | 4321  tester@12 | Incorrect password | Incorrect password | fail |  |
| 3.3 | Wrong username and right password | 432  tester@123 | Incorrect username | Incorrect username | fail |  |
| 3.5 | Wrong username and  Wrong password | 432  tester@12 | Incorrect username and password | Incorrect username and password | fail |  |

Example for testing(login)

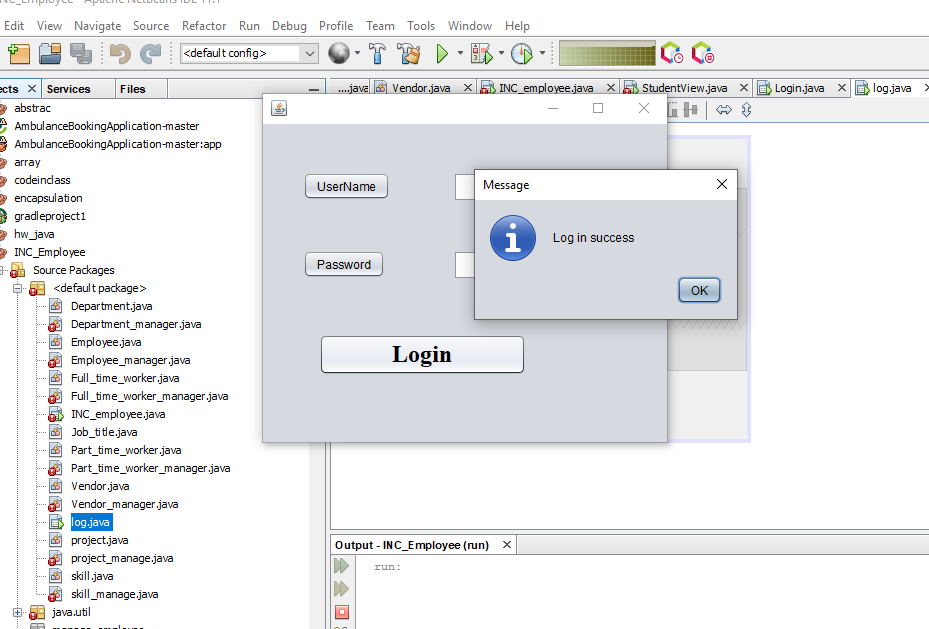


Figure 9:login

TEST CASES AND TEST RESULT

|  |  |
| --- | --- |
| **Test case ID** | 4 |
| **Test Unit** | Functionalities |
| **Test Type** | White box |
| **Tester** | R. Fernando |
| **Test Designer** | V. Fernando |
| **Venue** | LNBTI |
| **Date** | 2019/12/18 |
| **Time** | 12.00 a.m. – 1.00 a.m. |
| **Number of test cases** | Cases - 4  4.1Test insert function  4.2Test update function  4.3Test delete function  4.4Test search function |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Teat case | Test step | Test data | Expected results | Actual result | Status (Pass / Fail) | Notes |
| 4.1 | Insert a new department to the database | Project ID=441  Project name=IOT  Cost=100,000 | Values should be inserted successfully to the database | Insert Successfully  New recode inserted | pass |  |
| 4.2 | Update an address of a vendor | Address  Number line=12  Street: Beach road  City: Galle | Value should be updated successfully in the database. | Update successfully  Table updated | pass |  |
| 4.3 | Delete an employee from the database | Employee ID=001 | Values should be deleted successfully from the database. | Delete recodes from table | pass |  |
| 4.4 | Search a project in the database | Project number=001 | Values should be displayed in the User Interface. | Retrieved data are displayed in the interface | pass |  |

TEST CASES AND TEST RESULT

|  |  |
| --- | --- |
| **Test case ID** | 5 |
| **Test Unit** | Salary Calculation |
| **Test Type** | White box |
| **Tester** | R. Fernando |
| **Test Designer** | V. Fernando |
| **Venue** | LNBTI |
| **Date** | 2019/12/18 |
| **Time** | 12.00 a.m. – 1.00 a.m. |
| **Number of test cases** | Cases - 2  5.1Full time Employee’s salary calculation  5.2Part time Employee’s salary calculation |

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Teat case | Test step | Test data | Expected results | Actual result | Status (Pass / Fail) | Notes |
| 5.1 | Calculate the salary for the Full Time Employee. | Basic Salary=10,000  Deductions=1000  Allowances=5000 | Total Salary of the Full-Time employee should be calculated successfully.  Net salary = 14,000 | Net salary=  14,000 | pass |  |
| 5.2 | Calculate the salary for the Part Time Employee. | Hourly Rate=2000  Number of working hours=40 | Total Salary of the Part Time employee should be calculated successfully.  Net salary=80,000 | Net salary =80,000 | pass |  |

# Testing methods

* Black box testing

Decision tree (for login page)

Correct LOGIN ACCESS

correct

DENIED LOGIN

incorrect ACCESS

correct DENIED LOGIN

incorrect ACCESS

DENIED

LOGIN

incorrect ACCESS

Figure 10:decision tree

Decision table (for logging page)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **USE ID** | Y | Y | N | N |
| **PASSWORD** | Y | N | Y | N |
| **CORRECT** | Y |  |  |  |
| **INCORRECT** |  | Y | Y | Y |

State chart diagram for login page

Assumption: one user has 3 chances to insert username and password

Figure 11state chart diagram

3 try

S4

2 try

S3

1 try

S2

Start

S1

Correct user id and password

Incorrect user id or password

Incorrect use id and password

Denied login access

S7

S6

Login access

S5

|  |  |  |
| --- | --- | --- |
|  | Correct | Incorrect |
| S1 start | S5 | S2 |
| S2 1 try | S5 | S3 |
| S3 2 try | S5 | S4 |
| S4 3 try | S5 | S6 |
| S5 Login access | null | null |
| S6 Denied login access | null | null |

Equivalence partitioning (for part time workers hourly rate)

hours 1 3 6 9 12

hours< 1 – no payment

1<= hours >=3 - 2000

3< hours >=6 - 4000

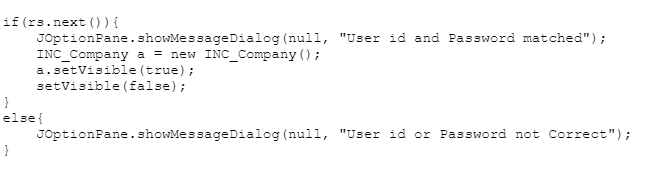
6<= hours >=9 - 6000

9<= hours >=12 - 8000

12<hours – 10 000

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TEST CASE | 1 | 2 | 3 | 4 | 5 | 6 |
| INPUT ITEM | 0 | 2 | 5 | 7 | 10 | 13 |
| HOURLY RATE | No payment | 2000 | 4000 | 6000 | 8000 | 12000 |
| PARTITION | hours<1 | 1<= hours >=3 | 3< hours >=6 | 6<= hours >=9 | 9<= hours >=12 | 12<hours |
| VALID/INVALID | Invalid | valid | valid | valid | valid | valid |
| ACTION | Input 0 | Input 0 | Input 0 | Input 0 | Input 0 | Input 0 |
| EXPECTED OUTPUT | Invalid values | 4000 | 20000 | 42000 | 80000 | 12000 |

* White box testing (login button condition)



* 100% statement coverage
* 100% decision coverage
* 100% condition coverage

By using 2 text cases.

Start

Attempt=0

Enter

Username and password

T If F

Attempt <3

If T

Username && password print

Correct “access denied”

F

print

“invalid username and password” access to INC Company page

attempt = attempt +1

End

Figure 12:flow chart for login

# User manual for INC Company

## General Information

### System overview

Firstly, user should be able to login to the system. Therefore he/she should be able to enter the correct User ID to the first textbox in the login interface. Then he/she should be able to enter the correct password to the second textbox in the login interface. Then user should click the login button in the interface. After that user should be able to login to the system of the INC company.

Then the User should be able to move to the second interface of the system which includes the Company details. If the user wants to Add an employee, he/she should be able to include the employee ID, Employee name and married status. If the user wants to update employee name, he/she should be able to enter the correct value for the suitable position in the interface. If the user wants to search an employee, he/she should enter the Employee ID and search the relevant employee. IF the user wants to delete an employee from the database, he/she should be entering the correct employee ID and delete the record of the relevant employee.

If the user wants to add a project to the database, he/she should enter the project ID, project name and the cost. If the user wants to update the cost of the project, he/she should enter the correct value and update it in the database. IF the user wants to search a project, he/she should enter the correct project ID and search the relevant project. If the user wants to delete a project from the database, he/she must enter the project ID and delete the record from the database.

If the user wants to add a department to the database, he/she add the department ID, Department name and the phone number. If the user wants to update the phone number of the department, he/she should enter the correct value and update it in the database. IF the user wants to search a department, he/she should enter the correct department ID and search the relevant department. IF the user wants to delete a department from the database, he/she must enter the department ID and delete the record from the database.

If the user wants to add a skill to the database, he/she add the skill ID, skill name. If the user wants to update the skill ID, he/she should enter the correct value and update it in the database. IF the user wants to search a skill, he/she should enter the correct skill ID and search the relevant skill. IF the user wants to delete a skill from the database, he/she must enter the skill ID and delete the record from the database.

## System summary

### Configuration

The system consists with several parts

1. Interfaces
2. Database

There is separate interface for each entity including the login interface. The interfaces are login interface, employee details, department details, vendor details, project details and skills details. All interfaces are liked together. Before access to the other interfaces it should be access the login interface.

Above interfaces are connect with the database for manipulate data which we want to enter or retrieve from the interfaces.

### Access levels

Access permissions are granted according to levels of each user

There are three types of access levels

1. Access as an administrator

Administrator has access to all interfaces internal codes and database. he is the person who has granted access to add, delete and update data in all fields.

1. Access as a tester

Tester can access to the data base for testing purposes. He is not allowed to access actual data in the system or do any changes.

1. Access as other user

There are two types of other users

Employee- employee can search data from employee, projects and skills, venders and department. They allow to add and update only few data such as marriage states, working hours. Employee revoke access to delete recodes.

Vendor- vendor can access only few fields in database. he not allows to access data such as employee, skills and projects.

## System functions

There are only few functions in the system

* Login

In the login page there is a login button. Before user enters to the other interfaces, he should be login to the system by providing correct username and password. User have three attempts to give username and password but if he unable to give correct username and password within tree attempts access will be denied.

* Add

Insert data which are in the text boxes into database

* Update

Update recodes in the database according to given id by the user

* Delete

Delete recodes from the database according to the given id

* Search

Retrieve data from the database which are corresponded to given id and display in the interface.

Except above functions system calculates salary separately for full time employee by using basic salary constraints and for part time employees by using hourly rate.

## Equipment and maintenance

Suitable computer which is having enough storage and good performance to run developed system.

If there is any problem with system, recommended to contact system administrator

# Technical document for INC Company

## Introduction

The “INC company” system consists with four main classes and two sub classes. Employee, Department, Project, Skills, Vendor, Job title are the main classes Full time worker and the Part time workers are the sub classes of the system.

Both Full time worker and the Part time workers class inherit Employee class and salary calculation method is overriding.

Inheritance, encapsulation, polymorphism, abstraction, aggregation, association and composition are the object-oriented programming methods used in this system.

## System overview and system design

This system mainly consists with interfaces, classes and database.

Interfaces

There are separate interfaces for each entity in the system. Login, employee, department, vendor, project and skills.

Classes

As well as interfaces, every entity has a its own class and manager class.

Ex: Employee class and employee manager class, department class and department manager class

Database

Database is connected with interfaces to complete insert, update, delete and search functionalities.

System functions

* Login

In the login page there is a login button. Before user enters to the other interfaces, he should be login to the system by providing correct username and password. User have three attempts to give username and password but if he unable to give correct username and password within three attempts access will be denied.

* Add

Insert data which are in the text boxes into database

* Update

Update recodes in the database according to given id by the user

* Delete

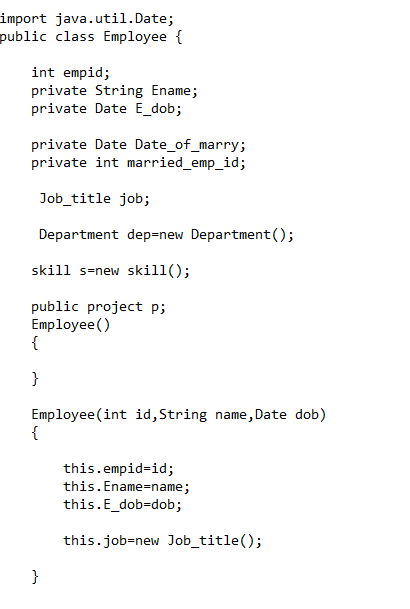
Delete recodes from the database according to the given id

* Search

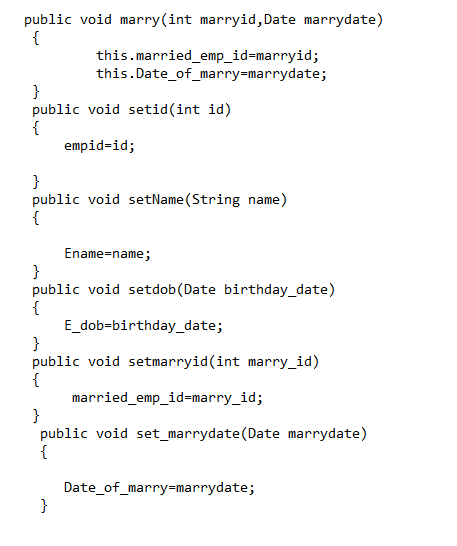
Retrieve data from the database which are corresponded to given id and display in the interface.

Except above functions system calculates salary separately for full time employee by using basic salary constraints and for part time employees by using hourly rate.

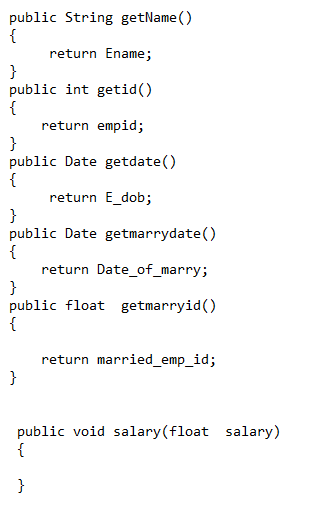
Following cords will show how the internal developing has been done



* Above code shows how the class id declared and how to create an object by using references.

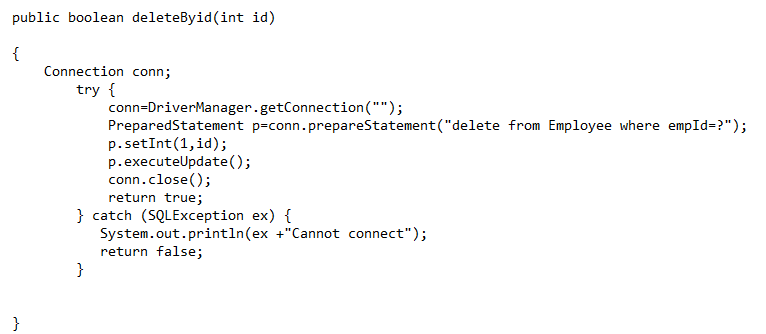


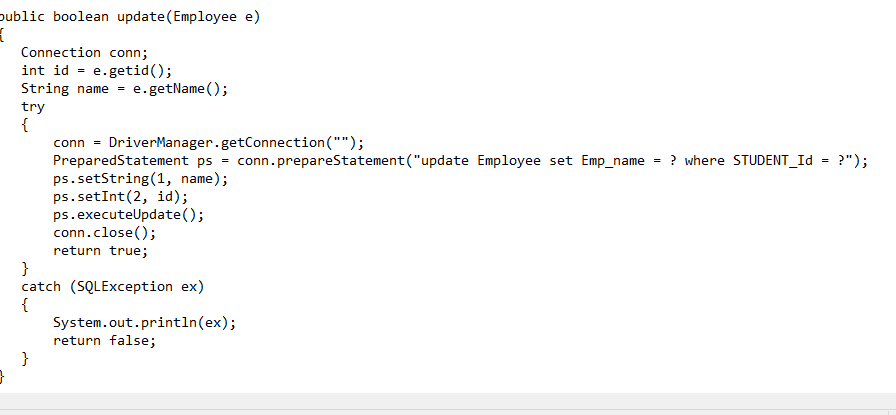
* Setters are used to assign a value to another variable

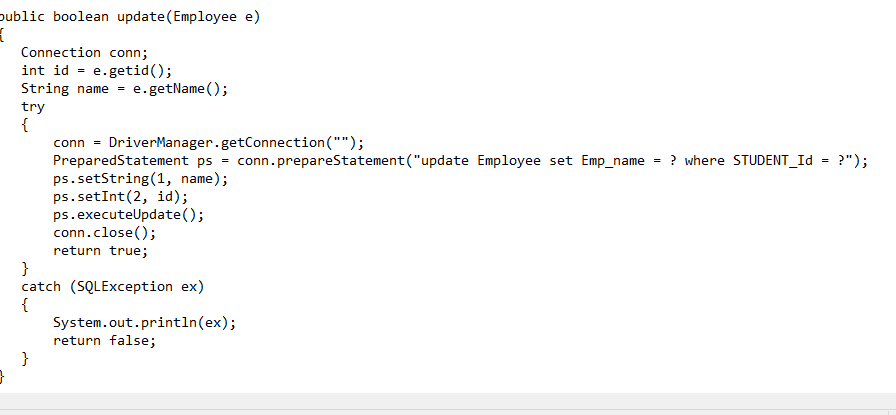


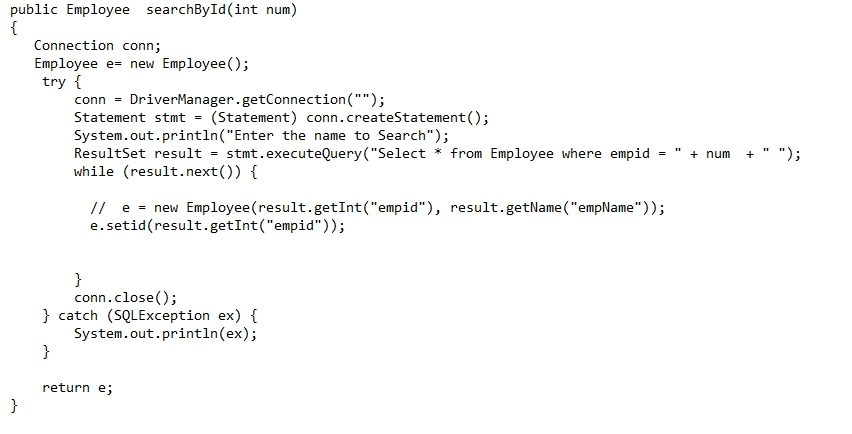
* Getters are used to return value

In this system it consists with employee class and employee manager class. when employee class is performed declaration part, manager class deals with complex coding parts which are connected with database.





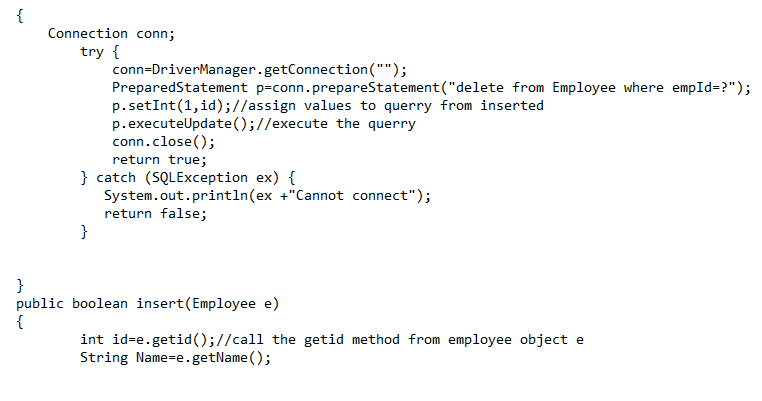




As mentioned in above, other classes also have manager class to manage system by using objects and methods which we have declared.

Comments are denoted by ‘//’ and comments makes the code much earlier to flower to clearly understand it.

Example: Employee



## Hardware requirements

Suitable computer which is having enough storage and good performance to run developed system.

## Software requirements

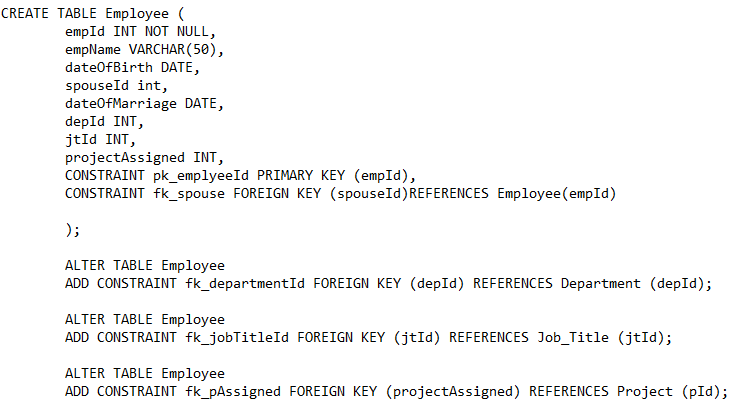
All cording parts are done by using NetBeans so user should have particular software facilities to run the system.

## Database design

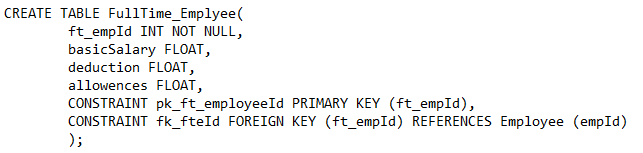
Database is created by using MS SQL

The database is defined as INC. Tables, attributes and the data types of database are mentioned below.

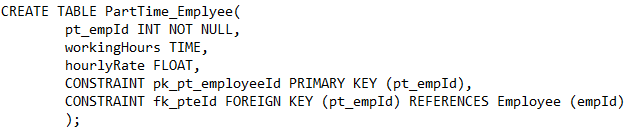
Employee table



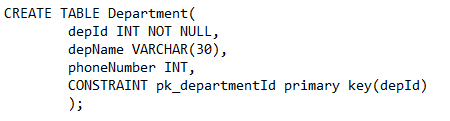
Full time employee table



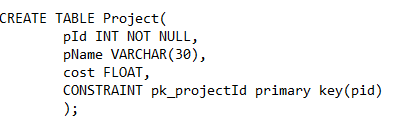
Part time employee table



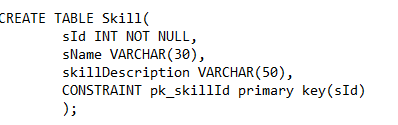
Department table



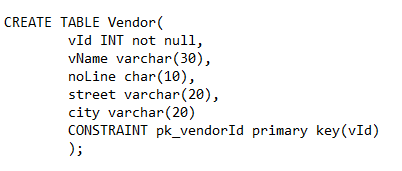
Project table



Skill table



Vendor table



## System security

For the security purpose there are three levels of access permissions

1.Access as an administrator

Administrator has access to all interfaces internal codes and database. he is the person who has granted access to add, delete and update data in all fields.

2.Access as a tester

Tester can access to the data base for testing purposes. He is not allowed to access actual data in the system or do any changes.

1. Access as other user

There are two types of other users

Employee- employee can search data from employee, projects and skills, venders and department. They allow to add and update only few data such as marriage states, working hours. Employee revoke access to delete recodes.

Vendor- vendor can access only few fields in database. he not allows to access data such as employee, skills and projects.

## Roles and responsibilities

* Administrator: Access granting, revoking to the system and manage entire system and databases.

Maintenance process

* Developers: develop all codes which are relevant to each class.
* Testers: responsible with testing process (both black box and white box testing)

# References

|  |  |
| --- | --- |
| [1] | "www.greekforgreeks.com," [Online]. |
| [2] | "guru99.com," 2019. [Online]. |
| [3] | "www.javaworld.com," [Online]. [Accessed 17 12 2019]. |
| [4] | "studytoNight," [Online].  www.javapoint.com  www.w3 schools dog com |

# **Individual contribution**

|  |  |  |  |
| --- | --- | --- | --- |
| Registration number | Name | System | Document |
| UIT/2019/FEB/0060 | Rochana Fernando | 1. Vender class 2. Department class | 1. Test plan 2. Test cases 3. User manual 4. Description of entities in diagrams |
| USE/2019/FEB/0062 | Shalan Dulanja | 1. Employee class 2. Full time employee class 3. Part time employee class | 1. Database diagram-ER diagram 2. Implement java solutions (cardinalities and OOP concepts) 3. Control structures 4. Error handling and reporting |
| UIT/2019/FEB/0064 | Harshani Cooray | 1. Project class 2. Skill class | 1. Use case diagram 2. Use case narratives 3. Testing methods 4. Test cases 5. Technical document |
| USE/2019/FEB/0066 | Tharindu Dhanushka | 1. Login 2. Database | 1. Introduction 2. Features of java   Evaluate JVM   1. Class diagram |