

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
“JNANA SANGAMA”, BELAGAVI - 590 018



A MINI PROJECT REPORT
on
“JOB PORTAL MANAGEMENT SYSTEM”

Submitted by

Crisel Mathias

4SF17IS015

Deeksha

4SF17IS017

In partial fulfillment of the requirements for the V semester

DBMS LABORATORY WITH MINI PROJECT

of

BACHELOR OF ENGINEERING

in

INFORMATION SCIENCE & ENGINEERING

Under the Guidance of

Mr. Ganaraj K.

Assistant Professor, Department of ISE

at



SAHYADRI

College of Engineering & Management

Adyar, Mangaluru - 575 007

2019 - 20

SAHYADRI
College of Engineering & Management
Adyar, Mangaluru - 575 007

Department of Information Science & Engineering



CERTIFICATE

This is to certify that the **Mini Project** entitled “**Job Portal Management System**” has been carried out by **Crisel Mathias (4SF17IS015)** and **Deeksha (4SF17IS017)**, the bonafide students of Sahyadri College of Engineering & Management in partial fulfillment of the requirements for the V semester **DBMS Laboratory with Mini Project (17CSL58)** of **Bachelor of Engineering in Information Science & Engineering** of Visvesvaraya Technological University, Belagavi during the year 2019 - 20. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in the report deposited in the departmental library. The mini project report has been approved as it satisfies the academic requirements in respect of mini project work.

Mr. Ganaraj K.
Assistant Professor
Dept. of ISE, SCEM

Dr. Shamanth Rai
HOD & Associate Professor
Dept. of ISE, SCEM

External Practical Examination:

Examiner's Name

Signature with Date

1.

.....

2.

.....

SAHYADRI
College of Engineering & Management
Adyar, Mangaluru - 575 007

Department of Information Science & Engineering



DECLARATION

We hereby declare that the entire work embodied in this Mini Project Report titled **“Job Portal Management System”** has been carried out by us at Sahyadri College of Engineering and Management, Mangaluru under the supervision of **Mr. Ganaraj K.** as the part of the V semester **DBMS Laboratory with Mini Project (17CSL58)** of **Bachelor of Engineering in Information Science & Engineering**. This report has not been submitted to this or any other University.

Crisel Mathias (4SF17IS015)

Deeksha (4SF17IS017)

SCEM, Mangaluru

Abstract

The importance of placement system is increasing day by day. Thousands of applicants are depending placement cell. But the applicants are facing so many problems. This project is an attempt to minimize the problems of an applicant to find a right job. This is a web-based application which will help candidates find a job with criteria like work profile, job type etc. The objective of job portal is to develop a system using which candidates and recruiters can communicate with each other. The purpose is to enable applicants to search for jobs in a convenient manner and to enable employers to find suitable candidates. Nowadays many jobs portals come up which gives information of the jobs that are available in different parts of country in different fields. Through this the job seekers can have easy search of their jobs online.

Acknowledgement

It is with great satisfaction and euphoria that we are submitting the Mini Project Report on “**Job Portal Management System**”. We have completed it as a part of the V semester **DBMS Laboratory with Mini Project (17CSL58)** of **Bachelor of Engineering in Information Science & Engineering** of Visvesvaraya Technological University, Belagavi.

We are profoundly indebted to our guide, **Mr. Ganaraj K.**, Assistant Professor, Department of Information Science & Engineering for innumerable acts of timely advice, encouragement and We sincerely express our gratitude.

We express our sincere gratitude to **Dr. Shamanth Rai**, Head & Associate Professor, Department of Information Science & Engineering for his invaluable support and guidance.

We sincerely thank **Dr. R.Srinivasa Rao Kunte**, Principal, Sahyadri College of Engineering & Management and **Dr. D. L. Prabhakara**, Director, Sahyadri Educational Institutions, who have always been a great source of inspiration.

Finally, yet importantly, We express our heartfelt thanks to our family & friends for their wishes and encouragement throughout the work.

Crisel Mathias

4SF17IS015

V Sem, B.E., ISE

SCEM, Mangaluru

Deeksha

4SF17IS017

V Sem, B.E., ISE

SCEM, Mangaluru

Table of Contents

| | |
|---|------------|
| Abstract | i |
| Acknowledgement | ii |
| Table of Contents | iii |
| List of Figures | v |
| 1 Introduction | 1 |
| 1.1 Database Management System | 2 |
| 1.2 Structured Query Language | 2 |
| 1.3 Stored Procedure | 3 |
| 1.4 Trigger | 4 |
| 1.5 Normalization | 4 |
| 2 Requirements Specification | 5 |
| 2.1 Hardware Requirements | 5 |
| 2.2 Software Requirements | 5 |
| 3 System Design | 6 |
| 3.1 ER Diagram | 6 |
| 3.2 Mapping From ER Diagram to Schema Diagram | 9 |
| 3.3 Schema Diagram | 12 |
| 4 Implementation | 14 |
| 5 Results and Discussion | 20 |
| 6 Conclusion and Future work | 28 |
| References | 29 |

List of Figures

| | | |
|------|--|----|
| 3.1 | Components of ER Diagram | 6 |
| 3.2 | ER Diagram of Job Portal Management System | 8 |
| 3.3 | Mapping of Regular Entities | 10 |
| 3.4 | Mapping of Weak Entities | 11 |
| 3.5 | 1:N Mapping | 12 |
| 3.6 | M:N Mapping | 12 |
| 3.7 | Schema Diagram of Job Portal Management System | 13 |
| 4.1 | User Table Description | 14 |
| 4.2 | JOB Table Description | 15 |
| 4.3 | Company Table Description | 16 |
| 4.4 | Skills Table Description | 16 |
| 4.5 | Skillnames Table Description | 17 |
| 4.6 | Has Table Description | 18 |
| 4.7 | Require Table Description | 18 |
| 4.8 | Searches Table Description | 19 |
| 5.1 | Homepage | 20 |
| 5.2 | Loginpage | 20 |
| 5.3 | Admin Login | 21 |
| 5.4 | Admin Options | 21 |
| 5.5 | User Display | 22 |
| 5.6 | Company Settings | 22 |
| 5.7 | Job Settings | 23 |
| 5.8 | Skill Settings | 23 |
| 5.9 | Signup Page | 24 |
| 5.10 | User Login | 24 |
| 5.11 | User Profile | 25 |
| 5.12 | Trigger Code | 25 |
| 5.13 | Stored Procedure | 26 |

| | | |
|------|-------------------------|----|
| 5.14 | Select Method | 26 |
| 5.15 | Delete Method | 27 |
| 5.16 | Insert Method | 27 |

Chapter 1

Introduction

Technology is constantly changing. Society as we know it depends on this fact. That which we take for granted today would have been the stuff of science fiction as little as fifty years ago. In fifty years' time, we will doubtless be excited, perturbed and baffled by yet more new developments. In the early years of the twenty first century, it is computers and the Internet that have captured the public imagination, and found their way into not just the working environments, but increasingly into the domestic spaces. In this modern society, if we are not capable to cope up with these changes than we are not going to stand or survive anywhere in this technical world. Today there is no place for errors, so as to make a system more effective and efficient we need such technology where error prone chances must be least.

In the scenario of the assignment, we are required to develop a database application on Job Portal Management System. In this time of recession where everyone, is either experienced or fresher, is in search for a job. This job portal can prove to be very helpful since it allows users of different profile to search job on the basis of their qualification. Every user can access through user id and apply for multiple jobs at a time. Now-a-days the job market is so extensive that a variety of industries and companies are searching for right candidates and the prospective candidates are searching for right companies for growth opportunities. This purpose is served by most of the job portals on line. This is another job portal with an open environment for the job seekers and recruiters to meet on the same dais and know about each other so that the right candidate is placed in a right company.

1.1 Database Management System

With the widespread use of computer technology and network technology, the development of database technology has become an important part of advanced information technology. The core of the property management system is how to use and operate database, so the database design is critical. This system uses the MySQL WorkBench is a visual database design tool that integrates SQL development, database design, creation and maintenance into a single integrated development environment for the MySQL database system. It is a product that always has been a leading position in the field of database. The MySQL database system is the world popular relational database management system which easy to use, strong function and suitable for all kinds of large, medium and small, microcomputer environment. It can realize data sharing and the facilities don't need to have the powerful data storage and processing capabilities so that to reduce the hardware cost of property management. Many enterprises have their own database, and store a large number of key data in it, which shows the importance of the database.

Every table in the database is broken up into smaller entities called fields. The fields in the Participants table consist of ParticipantID, ParticipantName, PhoneNumber and Gender. A field is a column in a table that is designed to maintain specific information about every record in the table. A record, also called a row, is each individual entry that exists in a table. A record is a horizontal entity in a table. A column is a vertical entity in a table that contains all information associated with a specific field in a table. In addition to tables, a database can also contain other objects including views, stored procedures, indexes and constraints, along with a transaction log.

1.2 Structured Query Language

SQL is a domain specific language used in programming and designed for managing data held in a database management system. SQL consists of a data definition language, data manipulation language and data control language. The scope of SQL includes data insert, query update and delete, schema creation and modification and data access control. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications which may run either on the same computer or on another computer across a network.

The main mode of retrieving data from a SQL Server database is querying for it. The query declaratively specifies what is to be retrieved. It is processed by the query processor, which figures out the sequence of steps that will be necessary to retrieve the requested data. The sequence of actions necessary to execute a query is called a query plan. There might be multiple ways to process the same query.

SQL Server also allows stored procedures to be defined. Stored procedures are parameterized T-SQL queries, that are stored in the server. Stored procedures can accept values sent by the client as input parameters, and send back results as output parameters. They can call defined functions, and other stored procedures, including the same stored procedure. Unlike other queries, stored procedures have an associated name, which is used at runtime to resolve into the actual queries. Also because the code need not be sent from the client every time it reduces network traffic and improves performance.

1.3 Stored Procedure

A stored procedure (also termed proc, storp, sproc, StoPro, StoredProc, StoreProc, sp, or SP) is a subroutine available to applications that access a relational database management system (RDBMS). Such procedures are stored in the database data dictionary.

Uses for stored procedures include data-validation (integrated into the database) or access-control mechanisms. Furthermore, stored procedures can consolidate and centralize logic that was originally implemented in applications. To save time and memory, extensive or complex processing that requires execution of several SQL statements can be saved into stored procedures, and all applications call the procedures. One can use nested stored procedures by executing one stored procedure from within another.

Stored procedures may return result sets, i.e., the results of a SELECT statement. Such result sets can be processed using cursors, by other stored procedures, by associating a result-set locator, or by applications. Stored procedures may also contain declared variables for processing data and cursors that allow it to loop through multiple rows in a table. Stored-procedure flow-control statements typically include IF, WHILE, LOOP, REPEAT, and CASE statements, and more. Stored procedures can receive variables, return results or modify variables and return them, depending on how and where the variable is declared.

1.4 Trigger

A database trigger is procedural code that is automatically executed in response to certain events on a particular table or view in a database. The trigger is mostly used for maintaining the integrity of the information on the database. For example, when a new record (representing a new worker) is added to the employees table, new records should also be created in the tables of the taxes, vacations and salaries. Triggers can also be used to log historical data, for example to keep track of employees' previous salaries.

1.5 Normalization

Database Normalization is the process of restructuring a relational database in accordance with a series of so-called normal forms in order to reduce data redundancy and improve data integrity. Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like Insertion, Update and Deletion Anomalies. It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization is used for mainly two purposes,

- Eliminating redundant (useless) data.
- Ensuring data dependencies make sense i.e data is logically stored.

Normalization Rule

Normalization rules are divided into the following normal forms:

- First Normal Form
- Second Normal Form
- Third Normal Form
- BCNF
- Fourth Normal Form

Chapter 2

Requirements Specification

2.1 Hardware Requirements

- Processor : Any processor above 500 MHz
- RAM : 4GB
- Hard Disk : 700GB
- Input Device : Standard keyboard and Mouse
- Output Device : Monitor

2.2 Software Requirements

- Database : MySQL WorkBench 8.0
- Programming Language : Java
- IDE :NetBeans 8.0.2

Chapter 3

System Design

3.1 ER Diagram

An entity-relationship diagram (ERD) is a data modelling technique that graphically illustrates an information system's entities and the relationships between those entities. An ERD is a conceptual and representation model of data used to represent the entity framework infrastructure. An ERD shows the relationships of entity sets stored in a database. An entity in this context is an object, components of data. These entities can have attributes that define its properties. By defining the entities and showing the relationship between them, an ER diagram illustrates the logical structure of database.

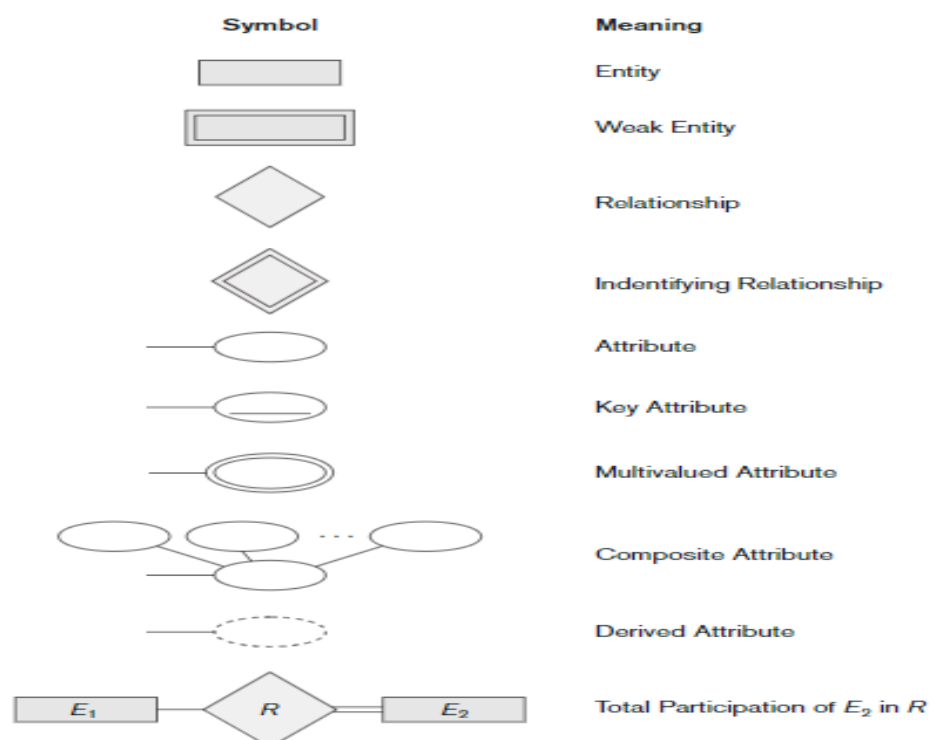


Figure 3.1: Components of ER Diagram

As shown in the above diagram, an ER diagram has three main components:

- Entity
- Attribute
- Relationship

Entity: Any real world object can be represented as an entity which data can be stored in a database. An entity is an object or component of data. An entity is represented as rectangle enclosing its name in an ER diagram.

For example: In the following ER diagram we have two entities COMPANY and JOB and these two entities have one to many relationship. We will read more about relationships later, for now focus on entities.

- **Strong Entity:** A strong entity has a primary key attribute which uniquely identifies each entity. Symbol of strong entity is same as an entity.
- **Weak Entity:** A weak entity does not have a primary key attribute and depends on other entity via a foreign key attribute.

Attribute: An attribute describes the property of an entity. An attribute is represented as Oval in an ER diagram. There are four types of attributes:

- **Key Attribute:** A column or a group of columns in a table which helps us to uniquely identify every row in that table is called key attribute.
- **Composite Attribute:** A composite attributes are the one which can be subdivided into smaller components which further forms attributes.
- **Single Valued Attribute:** If an attribute of a particular entity represents single value for each instance, then it is called a single valued attribute.
- **Multi Valued Attribute:** An attribute which can hold more than one value, it is then termed as multi valued attribute.
- **Derived Attribute:** A derived attribute calculate its value from another attribute.

Relationship A relationship is represented by diamond shape in ER diagram, it shows the relationship among entities. There are two types of relationship based on degree of relationship.

- **Unary Relationship:** If only single entity is involved in a relationship then it is a unary relationship.
- **Binary Relationship:** When two entities are associated to form a relation, then it is known as a binary relationship. There are four types of binary relationships:
 - **One to One:** The primary key of the participating relation R or S is added as primary key to second entity types by looking at the participating constraints.
 - **One to Many:** Foreign key approach is used to add one sided primary key to the n sided entity at foreign key.
 - **Many to One:** Foreign key approach is used to add n sided primary key to the one sided entity at foreign key.
 - **Many to Many:** Here we use the cross reference approach where the relationship is converted to a new relation within attributes on primary keys of both participating relation.

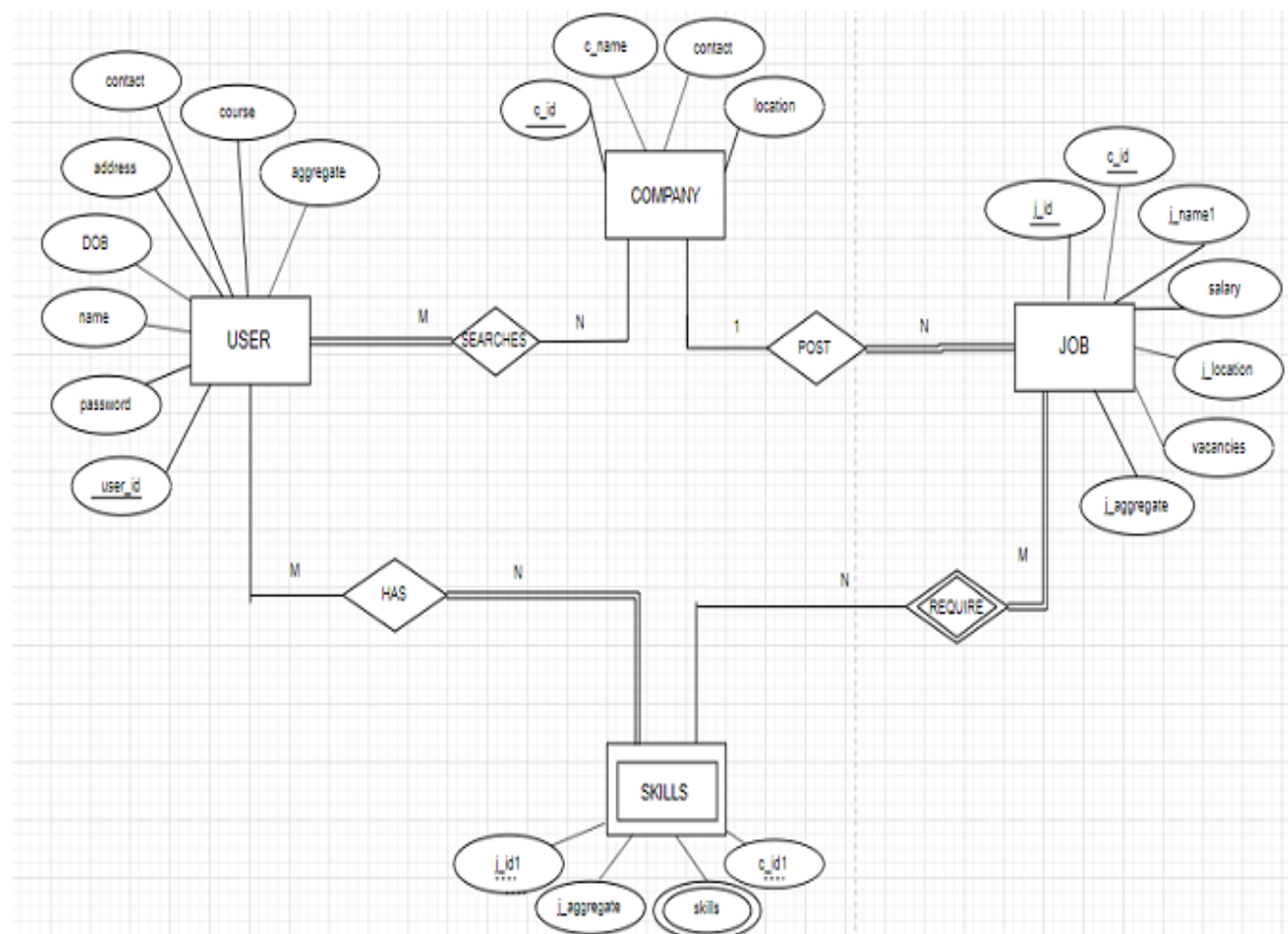


Figure 3.2: ER Diagram of Job Portal Management System

3.2 Mapping From ER Diagram to Schema Diagram

1.Mapping of Regular Entities:- This step involves mapping all the regular entity types to tabular format by identifying their primary keys.

2.Mapping of Weak Entity:- When mapping weak entity types along with other attributes the partial key and primary key of parent entity together will form their primary key of the new relation.

3.Mapping of 1:1 Relation:- In this step foreign keys are assigned using foreign key approach. The primary key of the participating relation R or S is added as primary key to second entity types by looking at the participating constraints.

4.Mapping of 1:N Relation:- Foreign key approach is used to add one sided primary key to the n sided entity as foreign key.

5.Mapping of M:N Relation:- Here we use the cross reference approach where the relationship is converted to a new relation with attributes on primary keys of both participating relation.

6.Mapping of Multivalued Relation:- For multivalued attributes a separate relation has to be created along with primary key of parent relation.

7.Mapping of N-ary Relation:- For mapping N array relationship we create a new relation with a relationship name in its attribute and primary keys of all participating entity types.

In our database we are having following mappings:

Step – 1 : Mapping of Regular Entities

From the ER diagram we identify all the strong entities E and create a relation R that includes all its simple attributes and primary keys. The following are the identified as the strong entities from our ER schema :

- 1.USER (Userid , Password, Name , DOB , Address , Contact , Course , Aggregate)
- 2.JOB(jid, cid , jname1 , Salary , jlocation , Vacancies , jaggregate)
- 3.COMPANY(cid, cname , Contact , Location)
- 4.SKILLS(jid1, cid1, jaggregate)

USER

| | | | | | | | |
|----------------|----------|------|-----|---------|---------|--------|-----------|
| <u>user_id</u> | Password | Name | DOB | Address | Contact | Course | Aggregate |
|----------------|----------|------|-----|---------|---------|--------|-----------|

COMPANY

| | | | |
|-------------|--------|---------|----------|
| <u>c_id</u> | c_name | contact | Location |
|-------------|--------|---------|----------|

JOB

| | | | | | | |
|-------------|------|--------|--------|------------|----------|-------------|
| <u>j_id</u> | c_id | j_name | salary | j_location | vacation | j_aggregate |
|-------------|------|--------|--------|------------|----------|-------------|

SKILLS

| | | | |
|-------------|-------------|-------------|-----------|
| <u>c_id</u> | <u>j_id</u> | Skill_names | Aggregate |
|-------------|-------------|-------------|-----------|

Figure 3.3: Mapping of Regular Entities

Step – 2 : Mapping of Weak Entity Types.

SKILLS

| | | | |
|-------------|-------------|-----------|--------|
| <u>j_id</u> | <u>c_id</u> | Aggregate | skills |
|-------------|-------------|-----------|--------|

Figure 3.4: Mapping of Weak Entities

Step – 3 : Mapping of Binary 1:1 Relation Types.

We do not have 1:1 Relation Type in our ER diagram.

Step – 4 : Mapping of Binary 1:N Relation Types.

Foreign key approach is used to add one sided primary key i.e CID from COMPANY to the n sided entity as foreign key in relation JOB.

JOB

| | | | | | | |
|-------------|--------|--------|------------|-----------|-------------|------|
| <u>j_id</u> | j_name | salary | j_location | vacancies | j_aggregate | c_id |
|-------------|--------|--------|------------|-----------|-------------|------|

Figure 3.5: 1:N Mapping

Step – 5 : Mapping of binary M:N Relation Types.

Here we use primary key i.e USERID from USER and primary key i.e CID and JID from SKILLS to create a new relation HAS.

Here we use primary key i.e JID from JOB and primary key i.e CID from SKILLS to create a new relation REQUIRE.

Here we use primary key i.e USERID from USER and primary key i.e CID from COMPANY to create a new relation SEARCHES.

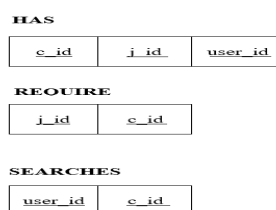


Figure 3.6: M:N Mapping

3.3 Schema Diagram

A schema is a pictorial representation between the database tables in the database created. The database schema of a database system is its structure described in a formal language supported by the database management system.

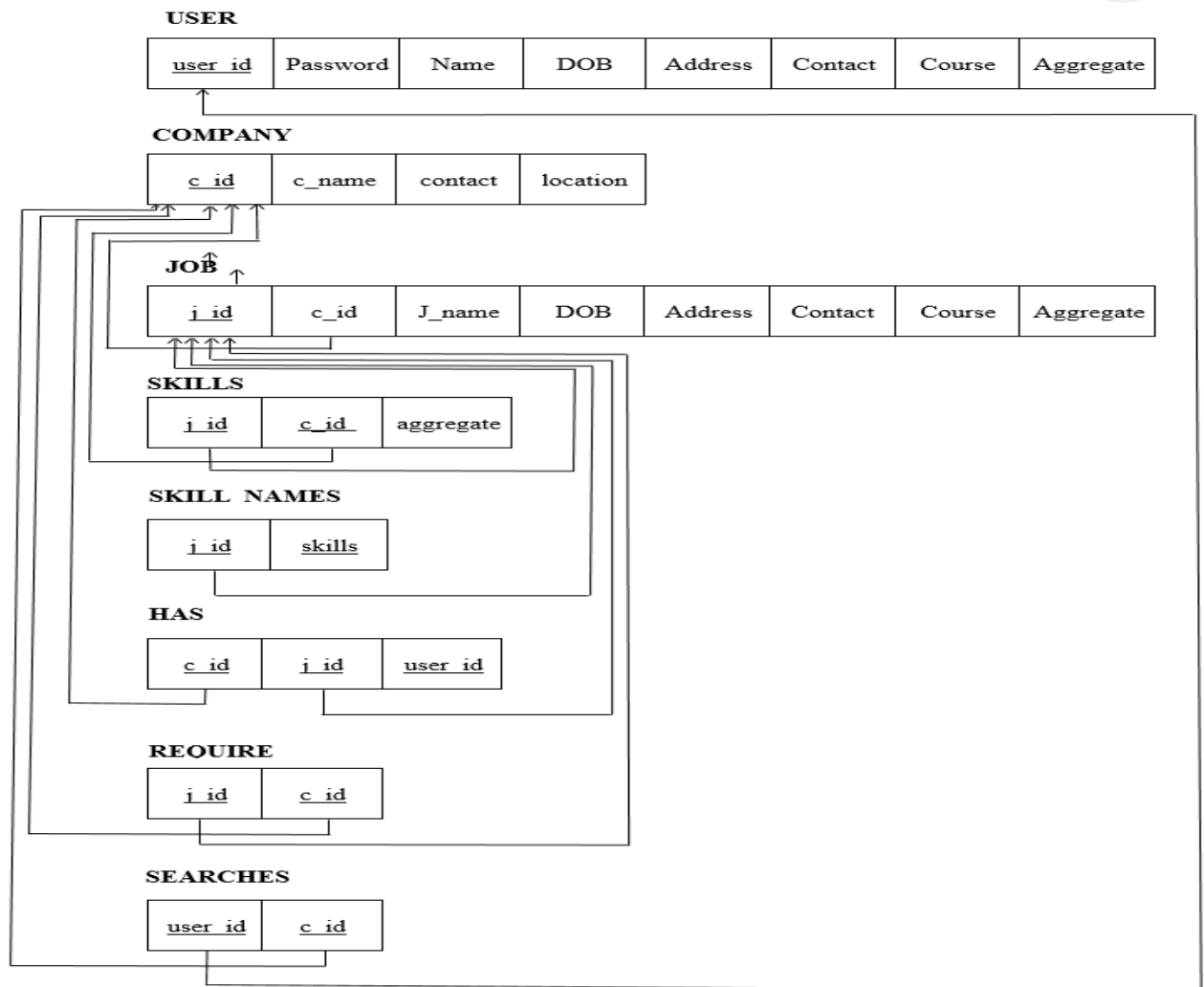


Figure 3.7: Schema Diagram of Job Portal Management System

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagram. It's the database designers who design the schema to help programmers understand the database and make it useful.

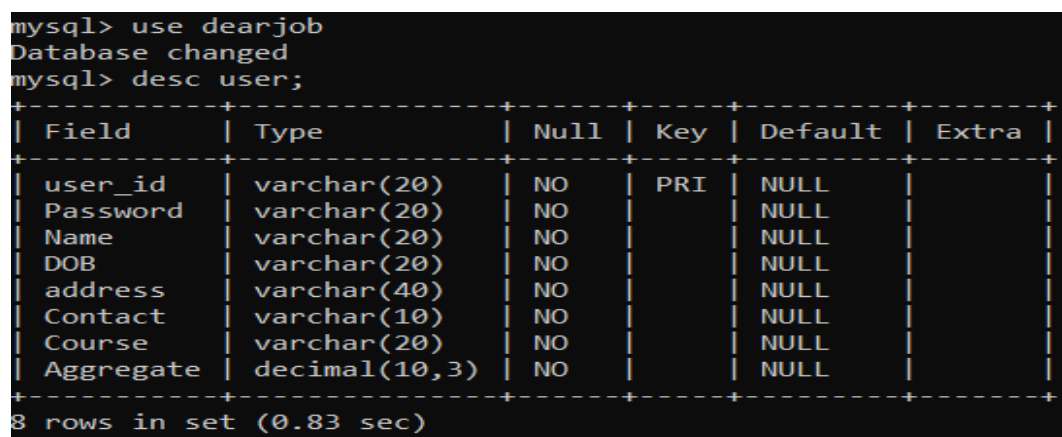
Chapter 4

Implementation

Table Structure

USER

```
CREATE TABLE USER
(USERID VARCHAR(20),
PASSWORD VARCHAR(20),
NAME VARCHAR(20),
DOB VARCHAR(20),
ADDRESS VARCHAR(40),
CONTACT VARCHAR(10),
COURSE VARCHAR(20),
AGGREGATE DECIMAL(10,3),
PRIMARY KEY(USERID));
```



```
mysql> use dearjob
Database changed
mysql> desc user;
```

| Field | Type | Null | Key | Default | Extra |
|-----------|---------------|------|-----|---------|-------|
| user_id | varchar(20) | NO | PRI | NULL | |
| Password | varchar(20) | NO | | NULL | |
| Name | varchar(20) | NO | | NULL | |
| DOB | varchar(20) | NO | | NULL | |
| address | varchar(40) | NO | | NULL | |
| Contact | varchar(10) | NO | | NULL | |
| Course | varchar(20) | NO | | NULL | |
| Aggregate | decimal(10,3) | NO | | NULL | |

8 rows in set (0.83 sec)

Figure 4.1: User Table Description

JOB

```

CREATE TABLE JOB
(JID VARCHAR(10),
CID VARCHAR(10),
JNAME1 VARCHAR(20),
SALARY VARCHAR(20),
JLOCATION VARCHAR(40),
JAGGREGATE VARCHAR(10),
PRIMARY KEY(JID));
FOREIGN KEY(CID) REFERENCES COMPANY(CID) ON DELETE CASCADE);

```

```
mysql> desc job;
```

| Field | Type | Null | Key | Default | Extra |
|-------------|-------------|------|-----|---------|-------|
| j_id | varchar(10) | NO | PRI | NULL | |
| c_id | varchar(10) | NO | MUL | NULL | |
| j_name1 | varchar(20) | NO | | NULL | |
| salary | varchar(20) | NO | | NULL | |
| j_location | varchar(40) | NO | | NULL | |
| vacancies | int(11) | NO | | NULL | |
| j_aggregate | varchar(10) | YES | | NULL | |

```
7 rows in set (0.00 sec)
```

Figure 4.2: JOB Table Description

COMPANY

```

CREATE TABLE COMPANY
(CID VARCHAR(10),
CNAME VARCHAR(20),
CONTACT VARCHAR(10),
LOCATION VARCHAR(40),
PRIMARY KEY(CID),

```

```
mysql> desc company;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| c_id       | varchar(10)   | NO   | PRI | NULL    |       |
| c_name     | varchar(20)   | NO   |     | NULL    |       |
| lontact    | varchar(10)   | NO   |     | NULL    |       |
| location   | varchar(40)   | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
4 rows in set (0.02 sec)
```

Figure 4.3: Company Table Description

SKILLS

```
CREATE TABLE SKILLS
```

```
(JID1 VARCHAR(10),
```

```
CID1 VARCHAR(45),
```

```
JAGGREGATE VARCHAR(45),
```

```
FOREIGN KEY(JID1) REFERENCES JOB(JID) ON DELETE CASCADE);
```

```
FOREIGN KEY(CID1) REFERENCES COMPANY(CID) ON DELETE CASCADE);
```

```
mysql> desc skills;
+-----+-----+-----+-----+-----+-----+
| Field      | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| j_id1      | varchar(10)   | NO   | PRI | NULL    |       |
| c_id1      | varchar(45)   | NO   | MUL | NULL    |       |
| j_aggregate | varchar(45)   | NO   |     | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (0.00 sec)
```

Figure 4.4: Skills Table Description

SKILLNAMES

```
CREATE TABLE SKILLNAMES
```

```
(JID2 VARCHAR(10),
```

```
SKILLS VARCHAR(20),
```

```
PRIMARY KEY(COMMID),
```

```
FOREIGN KEY(JID2) REFERENCES JOB(JID) ON DELETE CASCADE);
```

```
mysql> desc skill_names;
```

| Field | Type | Null | Key | Default | Extra |
|--------|-------------|------|-----|---------|-------|
| j_id2 | varchar(10) | NO | PRI | NULL | |
| skills | varchar(20) | NO | | NULL | |

```
2 rows in set (0.00 sec)
```

Figure 4.5: Skillnames Table Description

HAS

```
CREATE TABLE HAS
```

```
(CID VARCHAR(10),
```

```
JID VARCHAR(20),
```

```
USERID VARCHAR(10),
```

```
PRIMARY KEY(CID),
```

```
PRIMARY KEY(JID),
```

```
PRIMARY KEY(USERID),
```

```
FOREIGN KEY(CID) REFERENCES COMPANY(CID) ON DELETE CASCADE);
```

```
FOREIGN KEY(JID) REFERENCES JOB(JID) ON DELETE CASCADE);
```

```
mysql> desc has;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| c_id  | varchar(20)   | NO   | PRI | NULL    |       |
| j_id  | varchar(20)   | NO   | PRI | NULL    |       |
| user_id | varchar(20)  | NO   | PRI | NULL    |       |
+-----+-----+-----+-----+-----+-----+
3 rows in set (1.33 sec)
```

Figure 4.6: Has Table Description

REQUIRE

```
CREATE TABLE REQUIRE
(JID VARCHAR(10),
CID VARCHAR(20),
PRIMARY KEY(JID),
PRIMARY KEY(CID),
FOREIGN KEY(JID) REFERENCES JOB(JID) ON DELETE CASCADE);
FOREIGN KEY(JID) REFERENCES JOB(JID) ON DELETE CASCADE);
```

```
mysql> desc requires;
+-----+-----+-----+-----+-----+-----+
| Field | Type          | Null | Key | Default | Extra |
+-----+-----+-----+-----+-----+-----+
| j_id  | varchar(20)   | NO   | PRI | NULL    |       |
| c_id  | varchar(20)   | NO   | PRI | NULL    |       |
+-----+-----+-----+-----+-----+-----+
2 rows in set (0.03 sec)
```

Figure 4.7: Require Table Description

HAS

```
CREATE TABLE SEARCHES
(USERID VARCHAR(10),
```

```
CID VARCHAR(20),  
PRIMARY KEY(USERID),  
PRIMARY KEY(CID),  
FOREIGN KEY(USERID) REFERENCES USER(USERID) ON DELETE CASCADE);  
FOREIGN KEY(CID) REFERENCES COMPANY(CID) ON DELETE CASCADE);
```

```
mysql> desc searches;  
+-----+-----+-----+-----+-----+-----+  
| Field      | Type          | Null | Key | Default | Extra |  
+-----+-----+-----+-----+-----+-----+  
| user_id    | varchar(20)   | NO   | PRI | NULL    |       |  
| c_idb      | varchar(20)   | NO   | PRI | NULL    |       |  
+-----+-----+-----+-----+-----+-----+  
2 rows in set (0.04 sec)
```

Figure 4.8: Searches Table Description

Chapter 5

Results and Discussion



Figure 5.1: Homepage

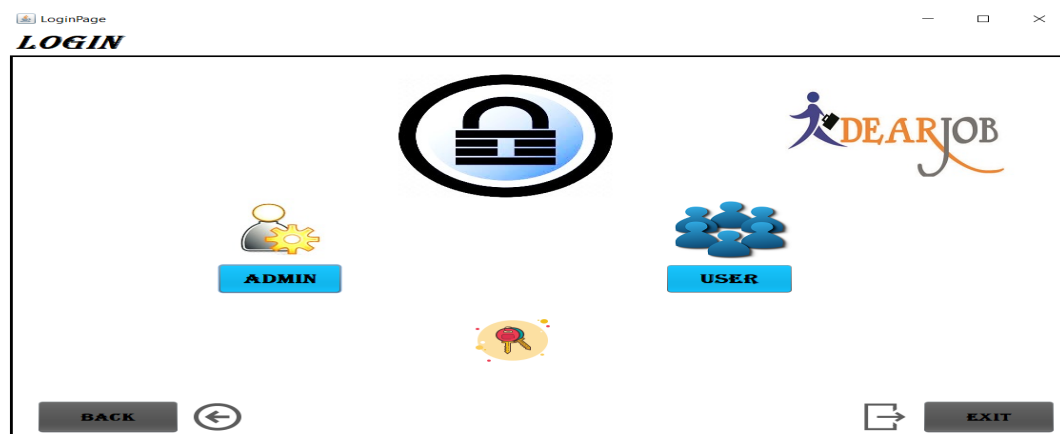


Figure 5.2: Loginpage

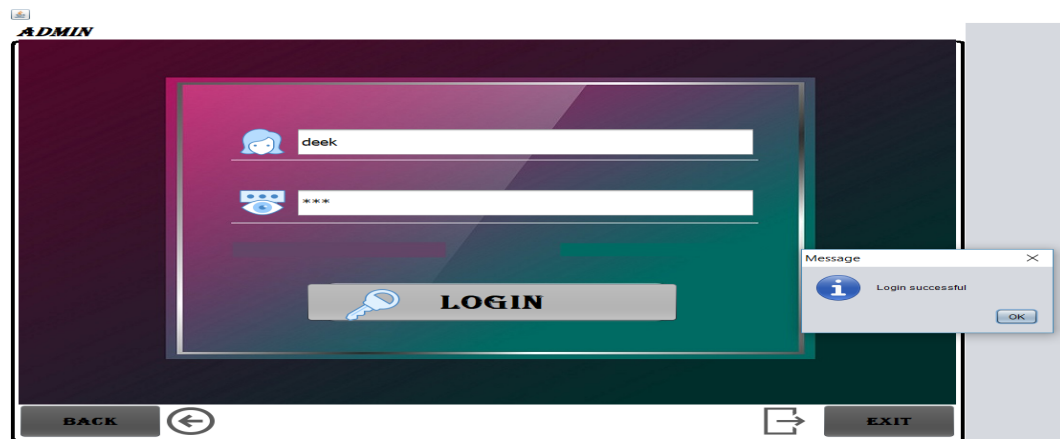


Figure 5.3: Admin Login



Figure 5.4: Admin Options

USER

| user_id | password | name | DOB | address | contact | course | aggregate |
|---------------|----------|---------|------------|-----------|------------|--------|-----------|
| crisel@gma... | 121212 | Crisel | 12-02-1999 | Bc road | 9483094847 | ISE | 8.080 |
| deek@gmai... | 123 | Deek | 27-05-2000 | Mangalore | 7894561230 | ISE | 9.000 |
| id | pass | jado | 4-6-2000 | bangalore | 4567891230 | ise | 5.000 |
| shdjh | 12dh | vjjgasd | 27jun | asfdgja | 123654 | is | 9.000 |

Figure 5.5: User Display

COMPANY

| C_ID | C_NAME | CONTACT | LOCATION |
|------|---------|------------|-----------|
| c01 | infosys | 9764318520 | mangalore |
| c02 | ibm | 4569871230 | blah |
| c03 | wipro | 9876541590 | usa |
| c04 | Adobe | 7847584783 | Goa |
| c05 | Canara | 9898989898 | Bantwal |
| c06 | Amazon | 6789876321 | Hyderabad |
| c07 | Google | 7865439870 | Usa |

Figure 5.6: Company Settings

The screenshot shows a window titled "JOB" with a yellow background. It contains a table with the following data:

| J_ID | C_ID | J_NAME | SALARY | J_LOCATI... | VACANCIES | aggregate |
|------|------|---------------------|--------|-------------|-----------|-----------|
| J01 | c03 | Manager | 60000 | Bangalore | 3 | 5 |
| J02 | c04 | Chef | 67000 | Delhi | 7 | 5 |
| J03 | c04 | Business Analyst | 70000 | California | 5 | 8 |
| J04 | c02 | designer | 50000 | kudla | 5 | 8 |
| J05 | c01 | tester | 10000 | london | 7 | 4 |
| J06 | c01 | Software developers | 105600 | London | 3 | 9 |
| J07 | c05 | Tester | 67888 | Bangalore | 9 | 7 |
| J08 | c03 | Python Developer | 304500 | New York | 10 | 7 |
| J09 | c01 | Marketing Manager | 70000 | Delhi | 8 | 7 |
| J10 | c04 | Tester | 5600 | Managlore | 6 | 5 |

Below the table are several buttons: "INSERT" (cyan), "DELETE" (black), "DISPLAY" (cyan), "BACK" (black), and "EXIT" (black). There is also a text input field labeled "enter j_id" and a circular arrow icon.

Figure 5.7: Job Settings

The screenshot shows a window titled "JOB" with a red background. It contains a table with the following data:

| j_id | skills |
|------|-----------------|
| J01 | Self management |
| J04 | python |
| J05 | java |
| J06 | Teamwork |
| J07 | Leadership |
| J10 | Technical |

Below the table are several buttons: "INSERT" (cyan), "DELETE" (black), "DISPLAY" (cyan), "BACK" (black), and "EXIT" (black). There is also a text input field labeled "insert j_id" and a circular arrow icon.

Figure 5.8: Skill Settings

SIGNUP

Mail ID :

New Password :

Name :

Birth date :

Address :

Contact :

Course :

Aggregate :

CREATE

Message: registration succes

OK

Figure 5.9: Signup Page

USER

Username :

Password :

LOGIN

Message: Login successfully

OK

BACK EXIT

Figure 5.10: User Login

PROFILE

Name:

DOB:

Address:

Contact:

Course:

Aggregate:

VIEW PROFILE

UPDATE

| c_id | j_id | c_name | contact | locati... | Jobna... | Job_L... | salary | vacan... |
|------|------|---------|------------|-----------|--------------|------------|--------|----------|
| c03 | j01 | wipro | 9876541... | usa | Manager | Bangalore | 60000 | 3 |
| c04 | j02 | Adobe | 7847584... | Goa | Chef | Delhi | 67000 | 7 |
| c04 | j03 | Adobe | 7847584... | Goa | Business... | California | 70000 | 5 |
| c02 | j04 | Ibm | 4559871... | India | designer | India | 55000 | 5 |
| c01 | j05 | Infosys | 9754318... | mangalore | tester | london | 10000 | 7 |
| c01 | j06 | Infosys | 9754318... | mangalore | Software... | London | 105500 | 3 |
| c05 | j07 | Canara | 9898989... | Banaral | Tester | Bangalore | 67888 | 9 |
| c03 | j08 | wipro | 9876541... | usa | Python D... | New York | 304500 | 10 |
| c01 | j09 | Infosys | 9754318... | mangalore | Marketing... | Delhi | 70000 | 8 |
| c04 | j10 | Adobe | 7847584... | Goa | Tester | Mangalore | 5500 | 6 |
| c11 | j12 | TCS | 9731919... | Bangalore | Software... | Bangalore | 55000 | 15 |

VIEW

SEARCH VIA JOBNAM:

SEARCH VIA SKILLS:

APPLY

Company name:

Contact:

Company location:

Job name:

Job location:

Salary:

Vacancies:

Figure 5.11: User Profile

```

1 CREATE DEFINER='root'@'localhost' TRIGGER `user_BEFORE_INSERT` BEFORE INSERT ON `user` FOR EACH ROW BEGIN
2   IF length(new.Contact)!=10 THEN
3     SIGNAL SQLSTATE '02000' SET MESSAGE_TEXT='PHONE NUMBER SHOUDL HAVE 10 DIGITS!';
4   END IF;
5
6 END

```

Figure 5.12: Trigger Code

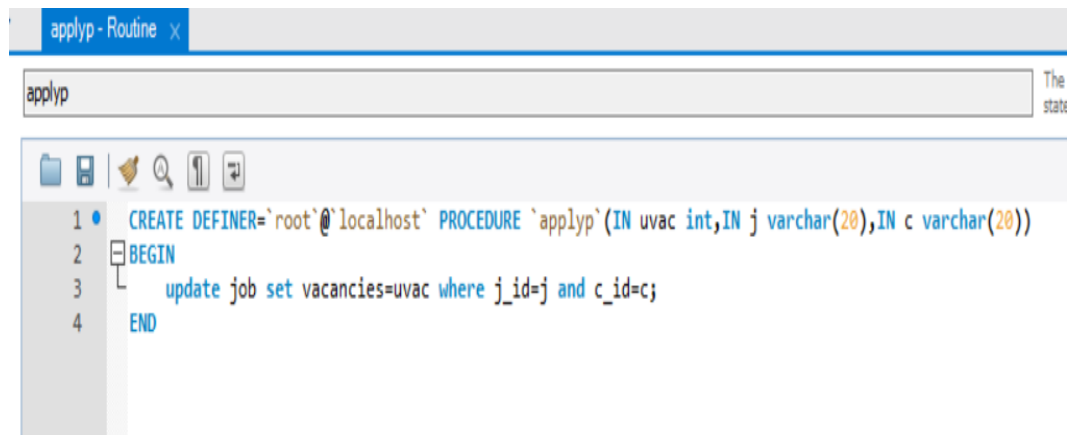


Figure 5.13: Stored Procedure

```

try
{
    Class.forName("com.mysql.cj.jdbc.Driver");
    Connection conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/dearjob","root","97319196dD");

    String sql="select * from dearjob.company inner join dearjob.job on dearjob.company.c_id=dearjob.job.c_id && dearjob.job.j_aggregate<=?";
    PreparedStatement psmt=conn.prepareStatement(sql);
    psmt.setString(1,aggregate.getText());
    ResultSet rs=psmt.executeQuery();
    DefaultTableModel tm=(DefaultTableModel)cstable.getModel();
    tm.setRowCount(0);
    while(rs.next())
    {

```

Figure 5.14: Select Method

```
private void cidActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try{  
        Class.forName("com.mysql.cj.jdbc.Driver");  
        Connection conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/dearjob","root","97319196dD");  
        String sql="delete from dearjob.company where c_id=?";  
        PreparedStatement psmt=conn.prepareStatement(sql);  
  
        psmt.setString(1,cid.getText());  
  
        psmt.executeUpdate();  
    }  
}
```

Figure 5.15: Delete Method

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try{  
        Class.forName("com.mysql.cj.jdbc.Driver");  
        Connection conn=DriverManager.getConnection("jdbc:mysql://localhost:3306/dearjob","root","97319196dD");  
        String sql="insert into dearjob.company values(?,?,?,?)";  
        PreparedStatement psmt=conn.prepareStatement(sql);  
  
        psmt.setString(1,cid.getText());  
        psmt.setString(2,cname.getText());  
        psmt.setString(3,contact.getText());  
        psmt.setString(4,location.getText());  
        psmt.executeUpdate();  
    }  
}
```

Figure 5.16: Insert Method

Chapter 6

Conclusion and Future work

Job Search Portals stands as a revolutionizing element in the sphere of recruitment. They act as a communication bridge between applicants and recruiters facilitating their requirements. This application helps organizations to have a greater exposure to the candidate pool and also job seekers facilitating wide search of jobs matching their interests. The web based application provides flexibility to the jobseekers to view and apply for the suitable jobs. This application provides an enhanced user experience for both employer and jobseeker. It provides user friendly interface which facilitates in reaching wide range of audience.

The project has achieved all the requirements that were initially set in the requirements gathering phase. This project taught us some best practices in the technology stack. Starting from requirements elicitation to design, construction, implementation, we have gained a very good experience working with various technologies at every phase. Development of this project boosted our confidence in web application development and work with modern technology.

This project fulfils the primary requirements of the job seekers and employers. It can be extended in several ways – We can provide recommendations and email updates for new job postings based on the job seeker’s search history and profile. Since, the job seekers might be interested in building a strong Resume, we can provide tips and information for the same. We can also provide templates for building the Resumes which might interest most applicants. The web application is developed fulfilling the functionalities of job seeker, it can be extended to support functionalities of Employer as well. And we can also make the automation to send the resume of the candidate to the recruiters when they click on “apply” and make it easy for the recruiters to conduct the further recruitment process for the selected candidates.

References

- [1] Database systems Models, Languages, Design and Application Programming, Ramez Elmasri and Shamkant B. Navathe, 6th Edition, Pearson.
- [2] Database management systems, Ramakrishnan, and Gehrke, 3rd Edition, 2014, McGraw Hill.
- [3] Silberschatz Korth and Sudharshan: Database System Concepts, 6th Edition, McGraw Hill, 2013.
- [4] Coronel, Morris, and Rob, Database Principles Fundamentals of Design, Implementation and Management, Cengage Learning 2012.