**CERTIFICATE**

This is to certify that YOUR NAME, of class XII-A, has prepared the Project File as per the prescribed syllabus of Computer Science (083) under the supervision of TEACHER NAME.

Academic Year: 2023-24

Internal Examiner External Examiner

**ACKNOWLEDGEMENT**

I would like to express my special thanks of gratitude to my teacher TEACHER NAME who gave me a golden opportunity to do this project of Computer Science, which also helped me in doing a lot of research and I came to know new things about it. Without their help, guidance and support it would be impossible to complete this project. Secondly, I would also like to thank my parents and friends who helped me a lot in finishing this project within a limited time. I am making this project not only for marks but also to increase my knowledge.

Once again thanks to all who helped me in doing this project.

**INDEX**

|  |  |
| --- | --- |
| Content | Pg No. |
| * Project Title | 4 |
| * Project Introduction * Needs * Benefits | 5 |
| * Project Objective * Features | 6 |
| * Technology Used | 7 |
| * Hardware & Software Requirements | 9 |
| * Project Code | 10 |
| * Table Structure | 22 |
| * Project Output | 23 |
| * Future Scope | 33 |

TITLE OF THE PROJECT:

**EMPLOYEE MANAGEMENT**

BY:

**YOUR NAME**

**PROJECT INTRODUCTION**

**Need:**

1. It is an overhead to keep the records related to employees on the papers and the chances of mistakes are high.

2. It is a burden to take out the register and view the records. It is a time-consuming process.

3. Most importantly, we should minimize paperwork so that there is no need to cut down more and more trees.

**Benefits:**

1. Keeps records related to the employees.

2. Easy to use and available at an affordable cost.

**PROJECT OBJECTIVES**

All the details regarding employees will be stored in one place safely. Chances of data redundancy are close to none.

**Key features:**

- Add Employee

- Remove Employee

- Update Employee Data

- View All Employee Data

- View Employee Data with some conditions

**Purposes:**

i) Quick and easy way to access employee data.

ii) One can search for employee details.

iii) Employer can assign a unique identification number to the employees.

iv) Employer can delete employee's record.

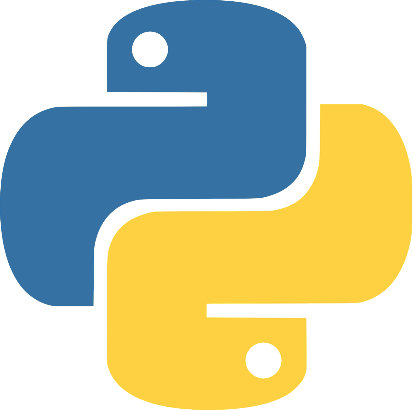
v) Employer can update employee's record.

**TECHNOLOGY USED**

**Python:**

Python is an interpreted, object-oriented, high level programming language with dynamic semantics. Its high-level

built in data structures, combined with dynamic typing and dynamic binding. It is used for web development (server-side), software development, machine learning & system scripting.



Application used-

Python IDLE 3.12.0 64-bit

**MySQL:**

SQL stands for Structured Query Language

It is a domain-specific language used in programming and designed for managing data held in a relational database (RDBMS), or for stream processing in a relational data stream management system (RDSMS). It provides many different types of commands used for different purposes. These commands can be divided into:

(i). Data Definition Language (DDL)

(ii). Data Manipulation Language (DML)

(iii). Transaction Control Language (TCL)

Application used-

MySQL 8.2.0

**MS Office:**

Microsoft Word is a widely used word processor developed by Microsoft. It allows users to create, edit, and format text documents, and comes with features for inserting tables, images, and other elements. It’s commonly used for creating professional documents like letters, reports, and resumes. The files are typically saved with a .docx extension.

Application Used-

Microsoft Office 365

**HARDWARE & SOFTWARE REQUIREMENTS**

**Hardware:**

Processor: Intel or AMD Multi-core Processor, 1.5 GHz or higher

Memory: 2 GB (minimum)

Architecture: x86 or x64

GPU: Integrated Intel or Radeon Graphics

Storage: 5 GB free disk space (HDD or SSD)

Mouse & Keyboard: Integrated Mouse & Keyboard

**Software:**

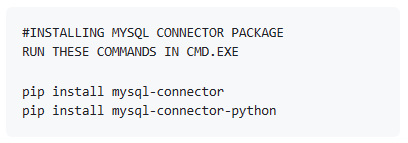
OS: Windows 7, MacOS X 10.11, or any compatible Linux distribution

Apps: MySQL 8.0 & Python IDLE 3.9

DirectX: DX9 (minimum)

**PROJECT CODE**

Install MySQL Connector package first.



Source Code:

import mysql.connector as sqltor

print('Enter MySQL Credentials below:')

hst = input('Enter hostname: ')

usr = input('Enter username: ')

pwd = input('Enter password: ')

mycon = sqltor.connect(host=hst, user=usr, passwd=pwd)

cursor = mycon.cursor()

if mycon.is\_connected():

print('Connected to MSQL Successfully')

else:

print('Not Connected, Please rerun the program')

def createdb():

gendetails = """

CREATE TABLE IF NOT EXISTS GeneralDetails

(EID INT PRIMARY KEY,

FName VARCHAR(20),

LName VARCHAR(20),

Email VARCHAR(100),

MobNo BIGINT,

Gender VARCHAR(2),

Department VARCHAR(50),

Title VARCHAR(50))

"""

perdetails = """

CREATE TABLE IF NOT EXISTS PersonalDetails

(EID INT PRIMARY KEY,

Address varchar(200),

DOB DATE,

Adhaar VARCHAR(12),

PAN VARCHAR(10))

"""

empdetails = """

CREATE TABLE IF NOT EXISTS EmploymentDetails

(EID INT PRIMARY KEY,

DOJ DATE,

HoursWorked INT,

HourlyRate INT)

"""

notes = """

CREATE TABLE IF NOT EXISTS Notes

(EID INT,

NID INT PRIMARY KEY,

Notes TEXT)

"""

emergencyinfo = """

CREATE TABLE IF NOT EXISTS EmergencyContactInfo

(EID INT PRIMARY KEY,

EmergencyContactName VARCHAR(50),

EmergencyContactNumber BIGINT)

"""

cursor.execute('CREATE DATABASE IF NOT EXISTS EMS')

cursor.execute('USE EMS')

cursor.execute(gendetails)

cursor.execute(perdetails)

cursor.execute(empdetails)

cursor.execute(notes)

cursor.execute(emergencyinfo)

print('Created Database and Tables successfully')

print('Run (3) to insert test records if you want')

def deldb():

cursor.execute('DROP DATABASE IF EXISTS EMS')

print('Removed sucessfully')

def regendb():

deldb()

createdb()

def testrec():

cursor.execute('USE EMS')

grec1 = "INSERT INTO GeneralDetails VALUES(001, 'Ramesh', 'Patel', 'rameshpatel1@gmail.com', 9845634724, 'M', 'Marketing', 'Marketing Manager')"

grec2 = "INSERT INTO GeneralDetails VALUES(002, 'Ujwal', 'Ahuja', 'ujwalahuja4@gmail.com', 9845734724, 'M', 'Accounting', 'Accounting Manager')"

grec3 = "INSERT INTO GeneralDetails VALUES(003, 'Yuvi', 'Garg', 'gargyuvi1@gmail.com', 9823634724, 'M', 'Accounting', 'Intern')"

prec1 = "INSERT INTO PersonalDetails VALUES(001, '285 Suchieta Niwas, 11 Shahid Bhagat Singh R, Fort, Mumbai', '1998-12-30', '662223509284', 'ALWPG5809L')"

prec2 = "INSERT INTO PersonalDetails VALUES(002, '2a, Shakespeare Sarani, Middleton Row, Kolkata', '1994-03-17', '226035064188', 'OKYZ3261NH')"

prec3 = "INSERT INTO PersonalDetails VALUES(003, 'J-1/52/f, Beriwala Bagh, Back Side Almariah Factor, Hari Nagar, Delhi', '1999-05-21', '817870326834', 'MOZKF016NB')"

erec1 = "INSERT INTO EmploymentDetails VALUES(001, '2017-02-21', 5636, 475)"

erec2 = "INSERT INTO EmploymentDetails VALUES(002, '2017-03-16', 5484, 450)"

erec3 = "INSERT INTO EmploymentDetails VALUES(003, '2021-10-23', 2037, 120)"

nrec1 = "INSERT INTO Notes VALUES(001, 001, 'Straight forward')"

nrec2 = "INSERT INTO Notes VALUES(001, 002, 'Good behaviour')"

nrec3 = "INSERT INTO Notes VALUES(002, 003, 'Lazy')"

nrec4 = "INSERT INTO Notes VALUES(003, 004, 'Newly joined')"

nrec5 = "INSERT INTO Notes VALUES(003, 005, 'Will work up the ranks')"

ecrec1 = "INSERT INTO EmergencyContactInfo VALUES(001, 'Aditya Patel', 7423101168)"

ecrec2 = "INSERT INTO EmergencyContactInfo VALUES(002, 'Sahil Ahuja', 7423101343)"

ecrec3 = "INSERT INTO EmergencyContactInfo VALUES(003, 'Farhan Garg', 7423101927)"

cursor.execute(grec1)

cursor.execute(grec2)

cursor.execute(grec3)

cursor.execute(prec1)

cursor.execute(prec2)

cursor.execute(prec3)

cursor.execute(erec1)

cursor.execute(erec2)

cursor.execute(erec3)

cursor.execute(ecrec1)

cursor.execute(ecrec2)

cursor.execute(ecrec3)

cursor.execute(nrec1)

cursor.execute(nrec2)

cursor.execute(nrec3)

cursor.execute(nrec4)

cursor.execute(nrec5)

mycon.commit()

print('Added Test Records Successfully')

def addemp():

cursor.execute('USE EMS')

ch = 'y'

while ch == 'y':

print('Enter Employee Details below:')

eid = int(input('Employee ID: '))

fname = input('First Name: ')

lname = input('Last Name: ')

email = input('Email: ')

mob = int(input('Mobile Number: '))

gender = input('Gender (M/F): ')

dept = input('Department: ')

title = input('Title: ')

address = input('Address: ')

dob = input('Date of Birth (YYYY-MM-DD): ')

adhaar = input('Aadhaar Number: ')

pan = input('PAN Number: ')

doj = input('Date of Joining (YYYY-MM-DD): ')

hours = int(input('Hours Worked: '))

rate = int(input('Hourly Rate: '))

emername = input('Emergency Contact Name: ')

emermob = int(input('Emergency Contact Number: '))

cursor.execute("INSERT INTO GeneralDetails VALUES({},'{}','{}','{}', {}, '{}', '{}', '{}')".format(eid, fname, lname, email, mob, gender, dept, title))

cursor.execute("INSERT INTO PersonalDetails VALUES({}, '{}', '{}', '{}', '{}')".format(eid, address, dob, adhaar, pan))

cursor.execute("INSERT INTO EmploymentDetails VALUES({}, '{}', {}, {})".format(eid, doj, hours, rate))

cursor.execute("INSERT INTO EmergencyContactInfo VALUES({}, '{}', {})".format(eid, emername, emermob))

mycon.commit()

print('Record added successfully')

ch = input('Want to enter more? (y/n): ')

def removeemp():

cursor.execute('USE EMS')

eid = int(input('Enter EID of employee you want to remove: '))

cursor.execute("select \* from GeneralDetails where EID={}".format(eid))

for rec in cursor:

print(rec)

ch = input('Are you sure you want to remove this employee? (y/n): ')

if ch == 'y':

cursor.execute('delete from GeneralDetails where EID={}'.format(eid))

cursor.execute('delete from PersonalDetails where EID={}'.format(eid))

cursor.execute('delete from EmploymentDetails where EID={}'.format(eid))

cursor.execute('delete from EmergencyContactInfo where EID={}'.format(eid))

mycon.commit()

print('Record removed successfully')

def updateemp():

cursor.execute('USE EMS')

eid = int(input('Enter EID of Employee to update: '))

tab = input('What detail do you want to update? (GeneralDetails, PersonalDetails, EmploymentDetails, Notes, EmergencyContactInfo): ')

cursor.execute("describe {}".format(tab))

for colname in cursor:

print(colname[0])

cursor.execute("select \* from {} where EID={}".format(tab, eid))

for rec in cursor:

print(rec)

field = input('Enter field you want to update: ')

val = input('Enter the new value: ')

cursor.execute("update {} set {}='{}' where EID={}".format(tab, field, val, eid))

mycon.commit()

print('Record updated successfully')

def projinfo():

print('This Project of EMPLOYEE MANAGEMENT SYSTEM is made by:')

print('Member A')

print('Member B')

print('Member C')

print('under the supervision of Teacher Name')

def searchemp():

cursor.execute('USE EMS')

term = input('Enter Employee ID or First Name or Last Name: ')

if term.isdigit():

cursor.execute("SELECT \* FROM GeneralDetails WHERE EID=%s", (int(term),))

else:

cursor.execute("SELECT \* FROM GeneralDetails WHERE FName=%s OR LName=%s", (term, term))

records = cursor.fetchall()

if not records:

print('No records found for the given search term.')

else:

print('Employee Records Found:')

for record in records:

print(record)

def viewemp():

cursor.execute('USE EMS')

query = """

SELECT GD.EID, GD.FName, GD.LName, GD.Email, GD.MobNo, GD.Gender, GD.Department, GD.Title,

PD.Address, PD.DOB, PD.Adhaar, PD.PAN,

ED.DOJ, ED.HoursWorked, ED.HourlyRate,

N.Notes,

EC.EmergencyContactName, EC.EmergencyContactNumber

FROM GeneralDetails GD

INNER JOIN PersonalDetails PD ON GD.EID = PD.EID

INNER JOIN EmploymentDetails ED ON GD.EID = ED.EID

LEFT JOIN Notes N ON GD.EID = N.EID

LEFT JOIN EmergencyContactInfo EC ON GD.EID = EC.EID

"""

cursor.execute(query)

records = cursor.fetchall()

count = cursor.rowcount

if records:

columns = [desc[0] for desc in cursor.description]

print('Employee Records:')

print(columns)

for record in records:

print(record)

else:

print(count, 'records in database')

else:

print('No employee records found.')

def mosthard():

cursor.execute('USE EMS')

cursor.execute("SELECT EID, MAX(HoursWorked) AS MostHoursWorked FROM EmploymentDetails GROUP BY EID ORDER BY MostHoursWorked DESC")

record = cursor.fetchone()

if record:

eid, hours = record

print('Most Hardworking Employee:')

print('Employee ID:', eid)

print('Total Hours Worked:', hours)

else:

print('No records found.')

def avgsal():

cursor.execute('USE EMS')

query = """

SELECT AVG(HoursWorked \* HourlyRate / 160) AS AvgMonthlySalary

FROM EmploymentDetails

"""

cursor.execute(query)

avgmonsal = cursor.fetchone()[0]

if avgmonsal:

print('Average Monthly Salary:', round(avgmonsal), '(approx)')

else:

print('No records found or average salary is zero.')

def addnote():

cursor.execute('USE EMS')

eid = int(input('Enter Employee ID: '))

nid = int(input('Enter a Note ID: '))

note = input('Enter note for Employee ({}): '.format(eid))

cursor.execute("INSERT INTO Notes VALUES({}, {}, '{}')".format(eid, nid, note))

mycon.commit()

print('Record Added Successfully')

def viewnotes():

cursor.execute('USE EMS')

ch = int(input('(1) Individual Notes or (2) All Notes: '))

if ch == 1:

eid = int(input('Enter Employee ID: '))

cursor.execute('SELECT \* FROM Notes WHERE EID = %s', (eid,))

data = cursor.fetchall()

if data:

print('Notes for Employee ID:', eid)

for row in data:

print(f'NID: {row[1]}, Note: {row[2]}')

else:

print('No notes found for this employee.')

elif ch == 2:

cursor.execute('SELECT \* FROM Notes')

data = cursor.fetchall()

if data:

print('All Notes:')

for row in data:

print(f'EID: {row[0]}, NID: {row[1]}, Note: {row[2]}')

else:

print('No notes found.')

else:

print('Invalid choice')

def updatenotes():

cursor.execute('USE EMS')

nid = int(input('Enter NID of note to update: '))

cursor.execute('select \* from Notes where NID={}'.format(nid))

for rec in cursor:

print(rec)

val = input('Enter the new note: ')

cursor.execute("update Notes set Notes='{}' where NID={}".format(val, nid))

mycon.commit()

print('Record updated successfully')

def emergencyprotocol():

cursor.execute('USE EMS')

eid = int(input('Enter Employee ID: '))

cursor.execute('SELECT \* FROM EmergencyContactInfo WHERE EID={}'.format(eid))

data = cursor.fetchall()

print('Emergency Contact Details:')

print('EID | EmergencyContactName | EmergencyContactNumber')

for row in data:

print(row)

def empdeptwise():

cursor.execute('USE EMS')

cursor.execute('SELECT DISTINCT Department FROM GeneralDetails')

departments = [row[0] for row in cursor.fetchall()]

print('Employee Records by Department:')

for department in departments:

print('Department:', department)

cursor.execute("SELECT \* FROM GeneralDetails WHERE Department=%s", (department,))

records = cursor.fetchall()

if records:

for record in records:

print(record)

else:

print('No records found for this department.')

print('----------------------------------------')

conti = 'y'

while conti == 'y':

print('')

print('')

print('')

print('')

print('')

print('Welcome to EMPLOYEE MANAGEMENT SYSTEM (v0.9-rev6)!')

print('')

print('')

print('Currently Available Operations (More coming soon)')

print('--------------------------------------')

print('1. Generate Database and Tables')

print('2. Regenerate Database and Tables')

print('3. Insert Test Records')

print('4. Add an Employee record')

print('5. Remove an Employee record')

print('6. Update an Employee record')

print('7. Search for an Employee record')

print('8. View all Employee records')

print('9. View Employees Department-wise')

print('10. Most Hardworking Employee')

print('11. Current Average Monthly Salary')

print('12. Add a note to an Employee')

print('13. View notes of an Employee')

print('14. Update notes of an Employee')

print('15. Emergency Contact Protocol')

print('--------------------------------------')

choice = int(input('Enter Operation Number (1-15): '))

if choice == 1:

createdb()

print('')

elif choice == 2:

regendb()

print('')

elif choice == 3:

testrec()

print('')

elif choice == 4:

addemp()

print('')

elif choice == 5:

removeemp()

print('')

elif choice == 6:

updateemp()

print('')

elif choice == 7:

searchemp()

print('')

elif choice == 8:

viewemp() **Scan for Code:**

 print('')

elif choice == 9:

empdeptwise()

print('')

elif choice == 10:

mosthard()

print('')

elif choice == 11:

avgsal()

print('')

elif choice == 12:

addnote()

print('')

elif choice == 13:

viewnotes()

print('')

elif choice == 14:

updatenotes()

print('')

elif choice == 15:

emergencyprotocol()

print('')

elif choice == 0:

projinfo()

print('')

else:

print('Invalid choice')

conti = input('Would you like to do more operations? (y/n): ')

else:

print('Bye! See you soon!')

**TABLE STRUCTURE**

**Tables and Fields:**

1) General Details

(Employee ID, First Name, Last Name, Email, Phone Number, Gender, Department, Job Title)

2) Personal Details

(Employee ID, Address, Date of Birth, Adhaar Number, PAN Number)

3) Employment Details

(Employee ID, Date of Joining, Hours Worked, Hourly Rate)

4) Notes Table

(Employee ID, Note ID, Notes)

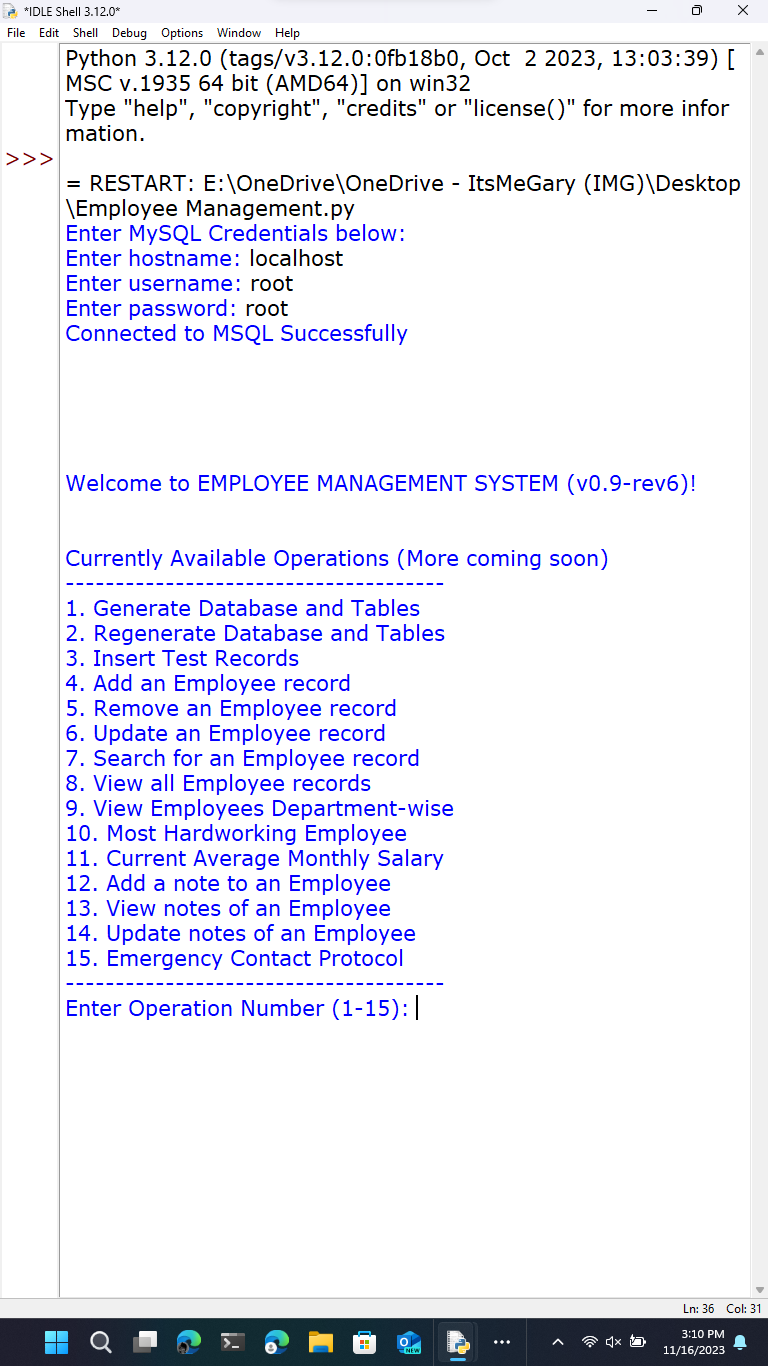
5) Emergency Contact Information Table

(Employee ID, Emergency Contact Name, Emergency Contact Number)

\*Database and Tables are made automatically by operation 1\*

**PROJECT OUTPUT**

Initializing:

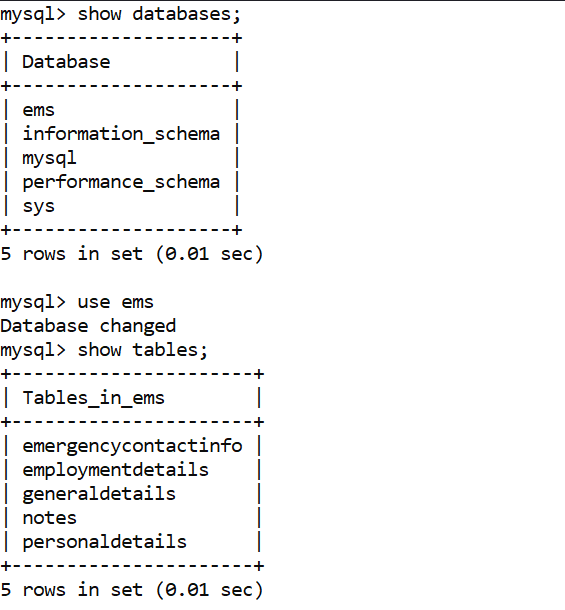


Operation 1:

A close-up of a number

Description automatically generated(Front-end Output)

(Back-end Output)



Operation 2:

(Front-end Output)

A white background with blue text

Description automatically generated

\*Back-end output is same as (1)

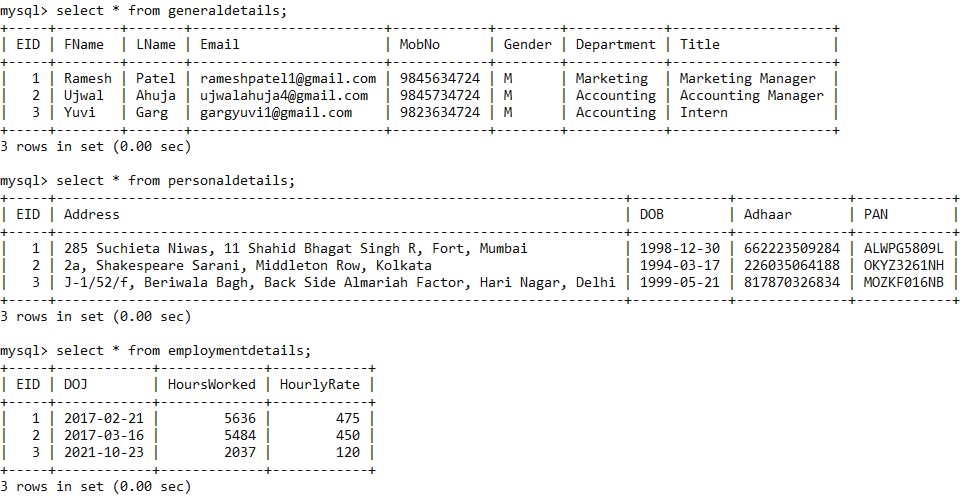
Operation 3:

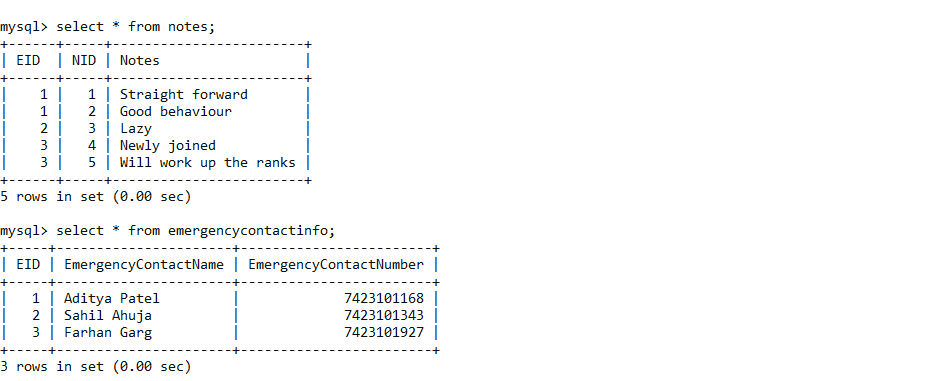
(Front-end Output)

A close up of text

Description automatically generated

(Back-end Output)





Operation 4:

(Front-end Output)

A screenshot of a computer

Description automatically generated

(Back-end Output)

A screenshot of a computer

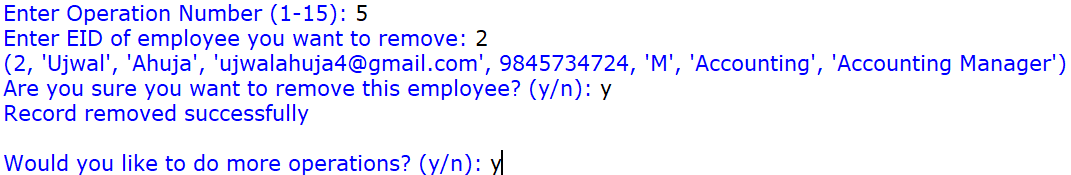
Description automatically generated

A screenshot of a computer code

Description automatically generated

Operation 5:

(Front-end Output)



(Back-end Output)

A close-up of numbers

Description automatically generated

A close-up of a computer screen

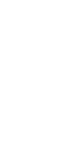
Description automatically generated

Operation 6:

(Front-end Output)

A close-up of a computer screen

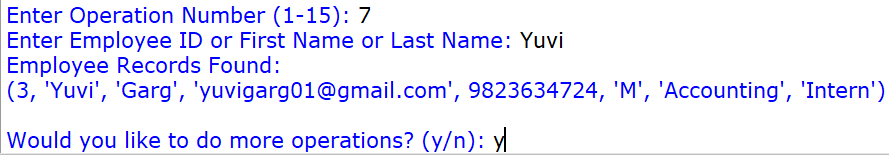
Description automatically generated



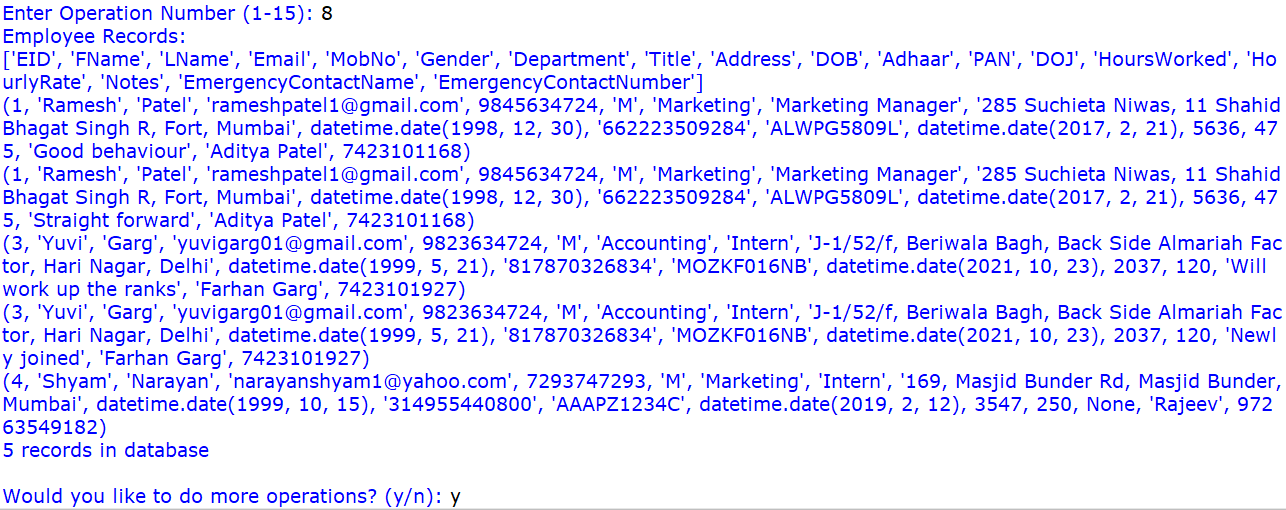
A close-up of numbers

Description automatically generated(Back-end Output)

Operation 7:



Operation 8:



Operation 9:

A close-up of a computer screen

Description automatically generated

A close-up of a number

Description automatically generatedOperation 10:

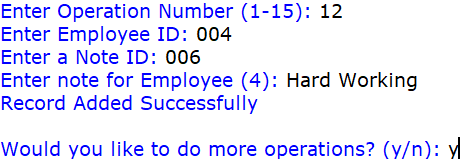
Operation 11:

A close-up of numbers

Description automatically generated

Operation 12:

(Front-end Output)



(Back-end Output)

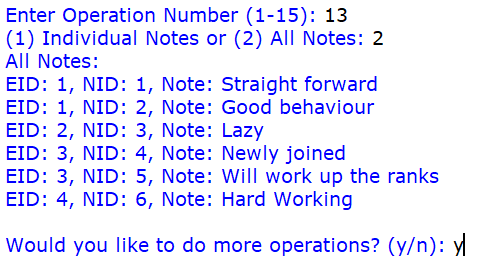
A white paper with black text

Description automatically generated

Operation 13:

A screenshot of a computer

Description automatically generated



Operation 14:

A screenshot of a computer

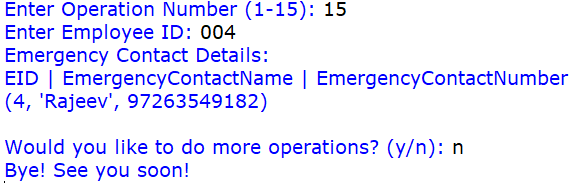
Description automatically generated(Front-end Output)

(Back-end Output)

A white text with black text

Description automatically generated

Operation 15:





**FUTURE SCOPE**

In any project, though present satisfaction is important but also it is equally important to see and visualize the future scope. The project which is developed now may need to undergo some changes in future in order to match up the technology prevailing that time, thus change due to development in technology are advisable.

Some of the future scopes include:

1. Exporting MySQL Data to a .csv file or an excel spreadsheet.

2. Website or Mobile app for managing on the go.

3. A dashboard for employers to view analytics.

4. Access to training materials for the employees and to track their personal development.

5. Store employee related documents like resumes, certifications, and contracts.