A MINI PROJECT

REPORT

for

MINI PROJECT IN PYTHON (20CSE59)

EMPBOOK

submitted by

DIVYASHREE S

1NH18CS061

5TH SEM/A SEC

In partial fulfillment for the award of

the degree of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

NEW HORIZON COLLEGE OF ENGINEERING

Certificate

This is to certify that the mini project work titled

EMPBOOK

Submitted in partial fulfillment of the degree of Bachelor of Engineering

Submitted by

DIVYASHREE S

1NH18CS061

DURING

ODD SEMESTER 2020-2021

For

20CSE59

Signature of Reviewer Signature of HOD

SEMESTER END EXAMINATION

Name of the Examiner *Signature with date*

1._____ _____

2._____ _____

EMPBOOK

ORIGINALITY REPORT

5%	%	5%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Piyush Goel, Bala Iyer. "SQL query optimization", ACM SIGMOD Record, 1996 Publication	2%
2	Ossama Embarak. "Chapter 1 Introduction to Data Science with Python", Springer Science and Business Media LLC, 2018 Publication	1%
3	Huang, Bao Hua, Tian Jing Wang, Yan Ma, and Feng Jiang. "Encrypting Database Content with User-Defined Function", Applied Mechanics and Materials, 2014. Publication	1%
4	Jon Stephens, Chad Russell. "Chapter 3 Keys, Indexes, and Normalization", Springer Science and Business Media LLC, 2004 Publication	<1%
5	Moustafa Mohamed Abdelmohsen. "The Development and Validation of a Module on Enhancing Students' Critical Thinking, Collaboration and Writing Skills", SAR Journal -	<1%

Science and Research, 2020
Publication

6	Adam. "DALI LED Driver Control System for Lighting Operations Based on Raspberry Pi and Kernel Modules", Electronics, 2019 Publication	<1%
7	Manohar Swamynathan. "Chapter 1 Step 1 – Getting Started in Python", Springer Science and Business Media LLC, 2017 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	On		

ABSTRACT

The main aim of this project is to develop a gui based application with sqlite database which helps the administrator of the company to login using admin id and password and print the salary slip of the employees according to the basic salary, over time and loan taken. The admin can perform operations such as adding a new employee record, updating a record, searching a record, deleting a record, displaying a record and printing the salary slip of the employee. Since every operation which is performed by the administrator is in a computerized way there are very less chances of errors and the data is stored permanently in the database and also has back up and recovery hence this project EMPBOOK can be very useful to organizations to maintain their employee details without any problem.

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be impossible without the mention of the people who made it possible, whose constant guidance and encouragement crowned our efforts with success.

I have great pleasure in expressing gratitude to **Dr. Mohan Manghnani**, Chairman of New Horizon Educational Institutions for providing necessary infrastructure and creating good environment.

I take this opportunity to express my profound gratitude to **Dr. Manjunatha,** Principal NHCE, for his constant support and encouragement.

I am grateful to **Dr. Prashanth C.S.R**, Dean Academics, for his unfailing encouragement and suggestions, given to me in the course of my project work.

I would also like to thank Dr**. B. Rajalakshmi**, Professor and Head, Department of Computer Science and Engineering, for her constant support.

I express my gratitude to **Ms. S. Shanmuga Priya,** Asst professor, my project guide, for constantly monitoring the development of the project and setting up precise deadlines. Her valuable suggestions were the motivating factors in completing the work.

DIVYASHREE S (1NH18CS061)

CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

1.1 PROBLEM DEFINITION

Manual handling of employee data and information poses a variety of challenges. The use of paper work in handling some of the processes could lead to human error, papers may end up in the wrong hands and not forgetting the fact that this is time consuming. There are many more challenges when it comes to providing the salary since calculation of salary as per rules of the company, income tax calculation and various deductions to be done from the salary including statutory deductions like Income tax and provident fund deductions has to be done. Hence, the drawbacks in the current system can be overcome by developing an application which will store all the details of the employee in the database and providing automated payslips with full authority access.

1.2 COURSE OBJECTIVE

The main objective of this mini project is to develop an application that will have the following functions: -

- Basically, mini project helps us to explore and strengthen the understanding of fundamentals through practical application of theoretical concepts.
- It also helps us to boost your skills and widen your horizon of thinking.
- It helps the beginners to do larger projects in their career.
- It is helpful to design our algorithm.
- Better learning of the coding language.
- To implement the concepts and learn to implement them properly.

1.3 OUTCOMES

The operations which can be performed using this application are

- Logging in with admin id and password
- Display the details of the employee
- Perform operations such as add/edit/search and delete employee details
- Print automated payslips

CHAPTER 2

REQUIREMENTS AND DESIGN

2.1 HARDWARE REQUIREMENT

- Processor: Intel premium iv or v AMD Athlon
- Ram:512 mb or more
- Hard disk:100Gb or high
- Input device: standard keyboard and mouse
- Output device: VGA and high-resolution monitor

2.2 SOFTWARE REQUIREMENT

- Operating systems: Windows 10
- front end: python
- Database: SQlite server

2.3 DATA MODEL/ER MODEL REPRESENTATION

ER Diagram:

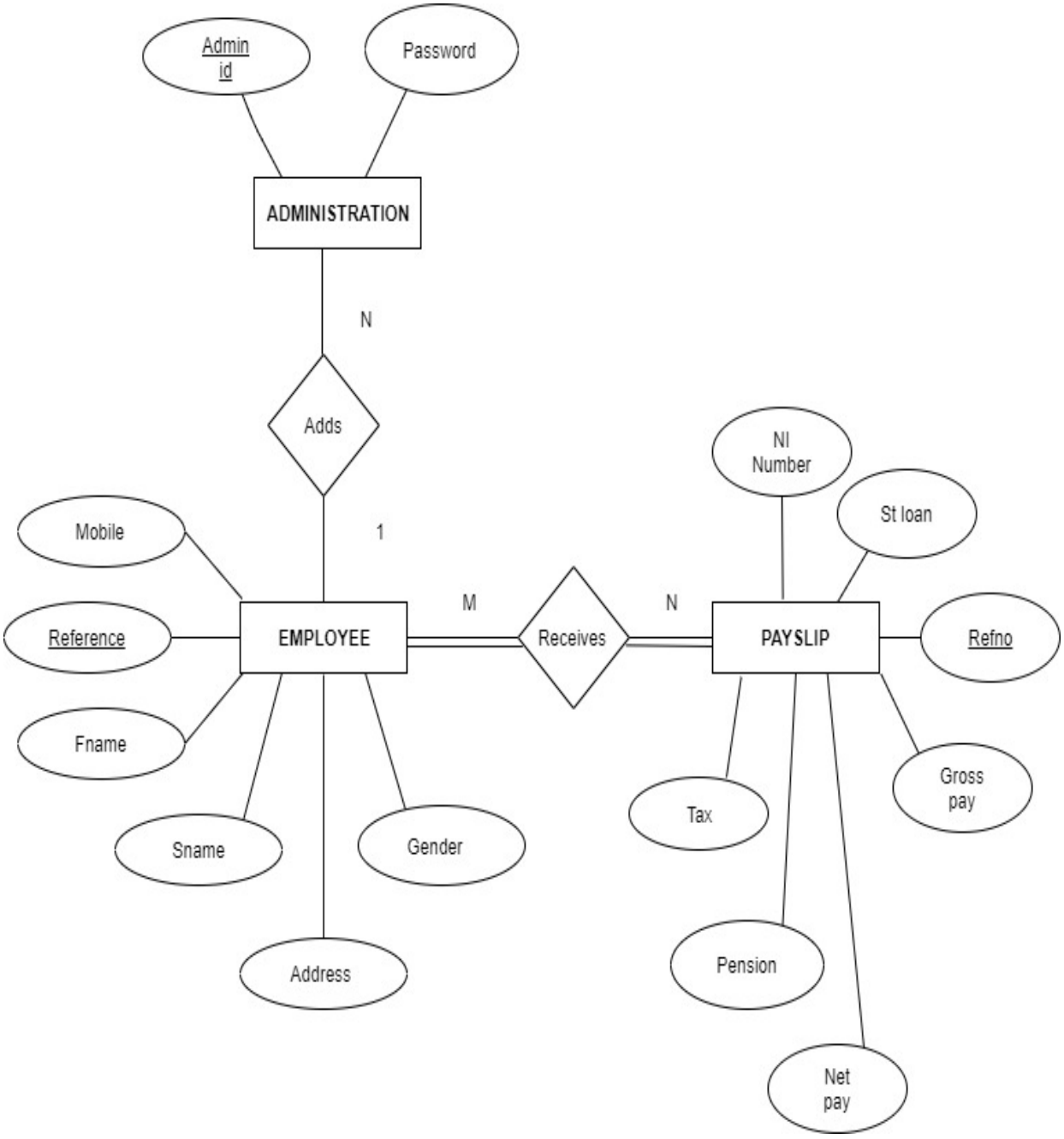

Fig 2.31 ER Diagram

Mapping:

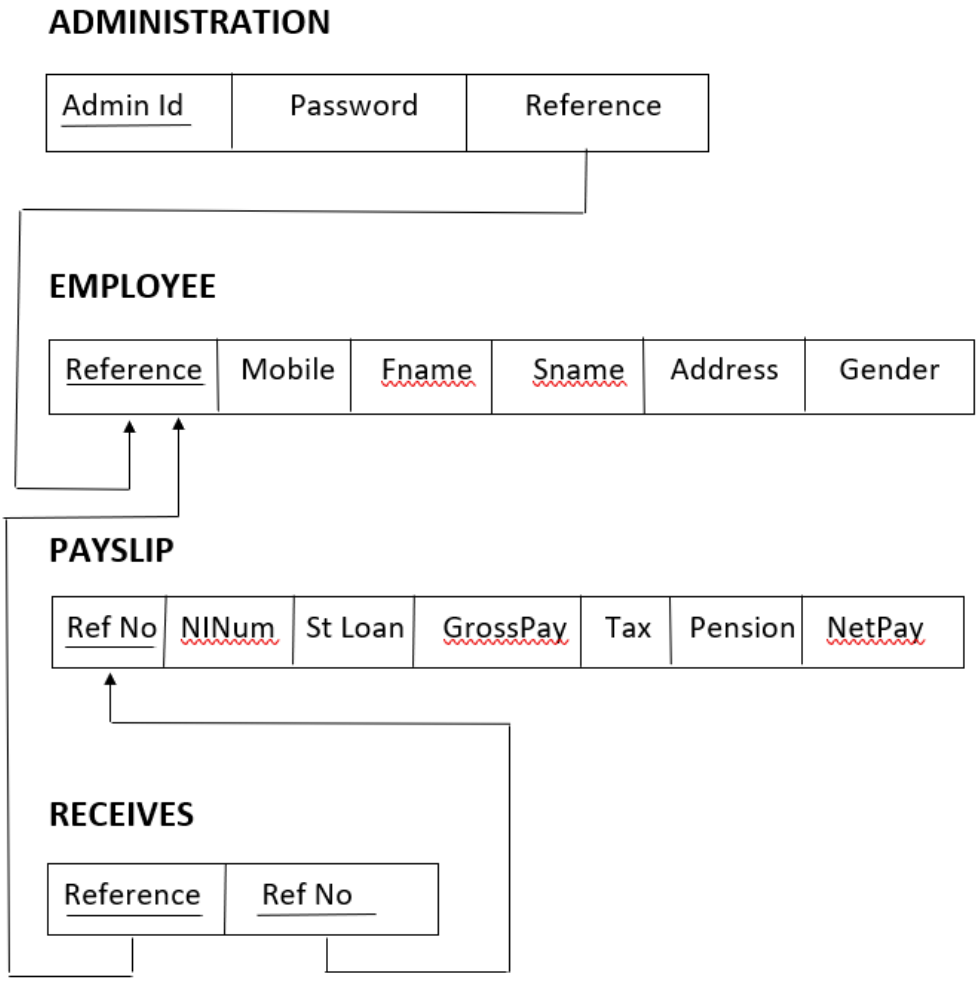

Fig 2.32 Mapping

CHAPTER 3

DATA MODELS / ER MODELS

3.1 ENTITY AND ATTRIBUTES

An entity can be a real-world object, either animate or inanimate, that can be easily identifiable. In this project we have three strong entities.

They are Administration, employee and pay slip and each of the entity has its own attributes.

- Administration
  Administration is one of the strong entities which has the attributes Admin Id and password. The admin can login using the admin id and password to view all the details of the employees.

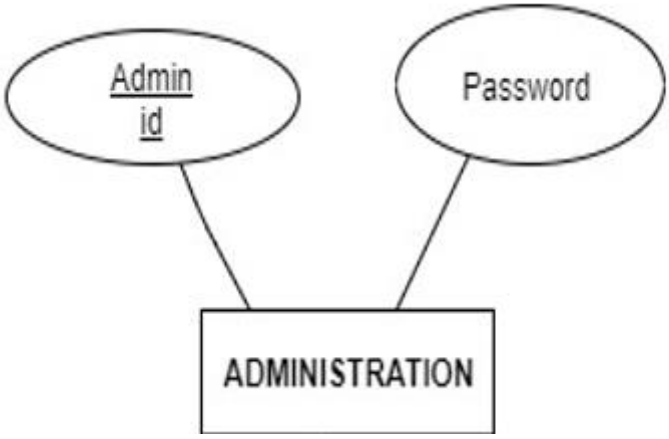

Fig 3.11 Administration entity

- Employee
  The employee entity has the attributes Reference, mobile, Fname, Sname, Address and gender. The attributes are saved under the employee information.

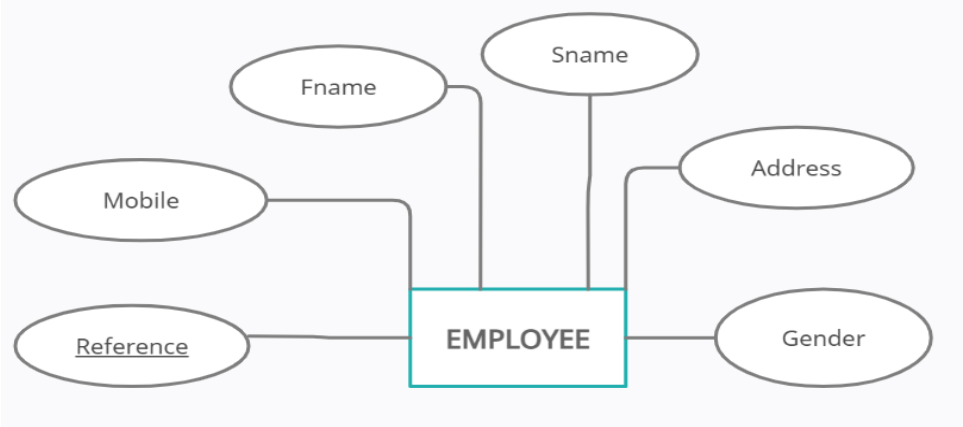

Fig 3.12 Employee entity

- Payslip
  Payslip is a strong entity with the attributes Ref no, NI Number, student loan, tax, pension, gross pay and net pay. All the above-mentioned attributes are calculated and displayed in the payslip.

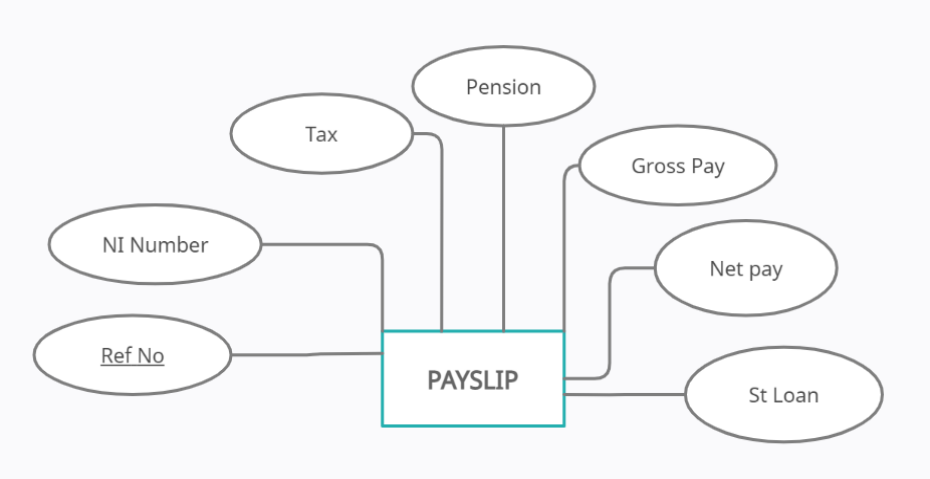

Fig 3.13 Payslip entity

3.2 KEYS AND CONSTRAINTS

Key is an attribute or collection of attributes that uniquely identifies an entity among entity set.

We have 5 types of key constraints

- NOT NULL: ensures that the specified column doesn't contain a NULL value.
- UNIQUE: provides a unique/distinct value to specified columns.
- DEFAULT: provides a default value to a column if none is specified.
- CHECK: checks for the predefined conditions before inserting the data inside the table.
- PRIMARY KEY: it uniquely identifies a row in a table.
- FOREIGN KEY: ensures referential integrity of the relationship

In this project we have primary keys which uniquely identifies each record.

In the first entity administration the attribute admin id is unique and no two Id's can have the same value hence it is the primary key of the administration entity.

In the second entity employee the attribute reference id is unique since no two employees can have the same reference id hence reference is the primary key for the entity employee and the employees are searched using their reference Id.

In the third entity Payslip, the attribute ref no is the primary key. Ref no is printed on top of each payslip and no two payslips will have same ref no.

3.3 RELATIONSHIP AND PARTICIPATION

The association among entities is called a relationship.

Cardinality defines the number of entities in one entity set, which can be associated with the number of entities of other set via relationship set.

Participation constraints define the least number of relationship instances in which an entity must compulsorily participate.

There are 2 types of participation constraints

1. Total participation
2. Partial participation

Total participation: -

It specifies that each entity in the entity set must compulsorily participate in at least one relationship instance in that relationship set.

That is why, it is also called as mandatory participation.

Total participation is represented using a double line between the entity set and relationship set.

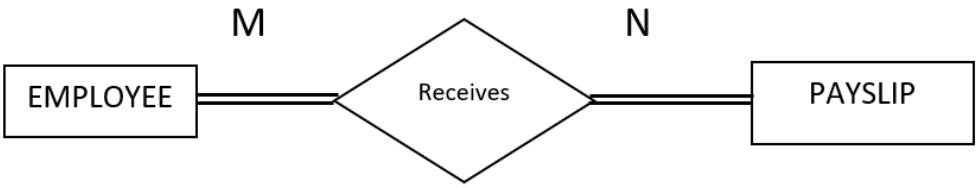

Fig 3.31 Total participation

Partial Participation: -

It specifies that each entity in the entity set may or may not participate in the relationship instance in that relationship set.

That is why, it is also called as optional participation.

Partial participation is represented using a single line between the entity set and relationship set.

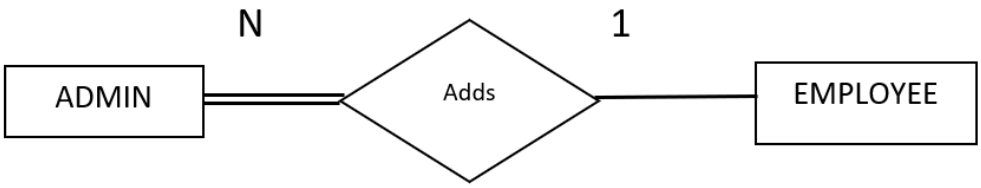

Fig 3.32 Partial participation

CHAPTER 4

DESIGNS

4.1 ALGORITHM

STEP 1: create a login page using tkinter module which has the fields username and password and create three buttons login, reset and exit

STEP 2: for login give the user name and password in the if condition if the username and password match then open the main empbook window else print a message displaying invalid details.

STEP 3: for reset function set the values to null

STEP 4: for exit function if the input is greater than 0 use destroy function

STEP 5: open a new file and add the neccessary modules required such as messagebox,random,datetime ect

STEP 6: add frames for the main empbook window which consists of feilds such as Firstname,Surname,Address,Reference,CityWeighting,Mobile,BasicSalary,OverTime,Gross Pay,NetPay,tax,pension,stdloan,ni number,deductions,gender, payday,tax period,ni code,taxablepay,pensionablepay,taxcode and otherpaymentsdue.

STEP7: create add,print,update,search,delete,reset,display and exit buttons.

STEP 8: create a function monthly salary and add all the necessary calculations needed.

STEP 9: open a new file for database add functions add,search,display and delete functions which consists of the related queires

STEP 10: import the database into the main empbook window code

STEP 11: import the main empbook window file to login system file

CHAPTER 5

IMPLEMENTATION

5.1 MODULE1 FUNCTIONALITY

Creating login page: -

```
class emplogin:
    def __init__(self, master):
        self.master = master
        self.master.title("empbook")
        self.master.geometry("1350x750+0+0")
        self.master.config(bg='cadet blue')
        self.frame = Frame(self.master, bg='cadet blue')
        self.frame.pack()
        self.Username = StringVar()
        self.Password = StringVar()
def Login_System(self):
        user = (self.Username.get())
        pas = (self.Password.get())
        if(user == str(user) and pas == str(123)):
                os.system('empbook.py')
        else:
            tkinter.messagebox.showerror("empbook","Invalid details")
            self.Username.set("")
            self.Password.set("")
    def iReset(self):
        self.Username.set("")
        self.Password.set("")


    def iExit(self):
        self.iExit = tkinter.messagebox.askyesno("empbook","Confirm if you want to exit")
```

```
    if self.iExit > 0:
        self.master.destroy()
        return
```

5.2 MODULE 2 FUNCTIONALITY

Adding employee records: -

```
def EmployeeData():
    con = sqlite3.connect( "Employee.db")
    cur = con.cursor()
    cur.execute("CREATE TABLE IF NOT EXISTS Employee (id INTEGER PRIMARY KEY, Reference text, \
    Firstname text, Surname text, Address text , Gender text, Mobile text, NINumber text, stdLoan text, Tax text, \
    Pension text, Deductions text, NetPay text, GrossPay text)")
    con.commit()
    con.close()
```

5.3 MODULE 3 FUNCTIONALITY

Viewing employee records: -

```
def viewData():
    con = sqlite3.connect("Employee.db")
    cur = con.cursor()
    cur.execute("SELECT * FROM Employee")
    rows = cur.fetchall()
    con.close()
    return rows
```

5.4 MODULE 4 FUNCTIONALITY

Deleting employee records: -

```
def deleteRec(id):
    con = sqlite3.connect("Employee.db")
    cur = con.cursor()
    cur.execute("DELETE  FROM Employee WHERE id=?", (id,))
    con.commit()
    con.close()
```

5.5 MODULE 5 FUNCTIONALITY

Searching employee records: -

```
def searchData(Reference= "", Firstname= "", Surname= "", Address= "", Gender= "", Mobile= "", NINumber= "", stdLoan= "", Tax= "", Pension= "", Deductions= "", NetPay= "", GrossPay= ""):
    con = sqlite3.connect("Employee.db")
    cur = con.cursor()
 cur.execute("SELECT * FROM Employee WHERE Reference = ? oR Firstname =? oR Surname = ? oR Address =? oR Gender =? oR Mobile =? oR NINumber =? oR stdLoan =? oR Tax =? oR Pension =? oR Deductions=? oR NetPay =? oR GrossPay=?",\
       (Reference, Firstname, Surname, Address, Gender, Mobile, NINumber, stdLoan, Tax, Pension, Deductions, NetPay, GrossPay))
    rows = cur.fetchall()
    con.close()
  return rows
```

5.6 MODULE 6 FUNCTIONALITY

Updating employee records: -

```
def dataUpdate(Reference= "", Firstname= "", Surname= "", Address= "", Gender= "", Mobile= "", NINumber= "", stdLoan= "", Tax= "", Pension= "", Deductions= "", NetPay= "", GrossPay= ""):
    con = sqlite3.connect("Employee.db")
    cur = con.cursor()
    cur.execute("UPDATE Employee SET Reference = ? , Firstname =? , Surname = ? ,dress =? , Gender =? , Mobile =? , NINumber =? , stdLoan =?  , Tax =? , Pension =? , Deductions=? , NetPay =? , GrossPay=?",\
                (Reference, Firstname, Surname, Address, Gender, Mobile, NINumber, stdLoan, Tax, Pension, Deductions, NetPay, GrossPay , id))
    con.commit()
    con.close()
```

5.7 MODULE 7 FUNCTIONALITY

Printing of receipt and calculations like monthly salary, tax ect: -

```
def PayRef():
        Payday.set(time.strftime("%d/%m/%Y"))
        Refpay = random.randint(16462, 7084009)
        Refpaid = ("Ref" +str(Refpay))
        Reference.set(Refpaid)

        NIpay = random.randint(40005, 408400)
        NIpaid = ("NI" +str(NIpay))
        NINumber.set(NIpay)

        iDate = datetime.datetime.now()
        TaxPeriod.set(iDate.month)
```

```
        NCode = random.randint(1556, 13976)
        CodeNI = ("NIC" + str(NCode))
        NICode.set(CodeNI)

        iTaxCode = random.randint(7556, 15976)
        PaymentTaxCode = ("TCode" + str(iTaxCode))
        TaxCode.set(PaymentTaxCode)
    def MonthlySalary():
        PayRef()

        BS= float(BasicSalary.get())
        CW= float(CityWeighting.get())
        OT= float(OverTime.get())
        OPD= float(OtherPaymentDue.get())

        MTax = ( (BS+CW+OT+OPD) *0.3)
        TTax = "Rs",str('%.2f'%(MTax))
        Tax.set(TTax)

        M_Pension = ((BS + CW + OT + OPD) * 0.02)
        MM_Pension = "Rs",str('%.2f'%(M_Pension))
        Pension.set(MM_Pension)

        M_stdLoan = ((BS +CW+OT+OPD)* 0.012)
        MM_stdLoan = "Rs",str('%.2f'%(M_stdLoan))
        stdLoan.set(MM_stdLoan)

        M_NIPayment = ((BS +CW+OT+OPD)* 0.011)
        MM_NIPayment  = "Rs",str('%.2f'%(M_NIPayment))
```

```
        NIPayment.set(MM_NIPayment)

        Deduct = (MTax + M_Pension + M_stdLoan + M_NIPayment)
        Deduct_Payment = "Rs",str('%.2f'%(Deduct))
        Deductions.set(Deduct_Payment)
        Gross_Pay = "Rs",str('%.2f'%( BS + CW + OT + OPD))
        GrossPay.set(Gross_Pay)

        NetPayAfter = (BS +CW+OT+OPD) - Deduct
        NetAfter = "Rs",str('%.2f'%(NetPayAfter))
        NetPay.set(NetAfter)
        TaxablePay.set(TTax)
        PensionablePay.set(MM_Pension)

        self.txtReceipt.insert(END,'\t\t Monthly Pay Slip' +"\n\n")
        self.txtReceipt.insert(END,'Reference: \t\t'+Reference.get() +"\n")
        self.txtReceipt.insert(END,'Reference: \t\t'+Payday.get() +"\n")
        self.txtReceipt.insert(END,'Employer Name:\t\t'+ Firstname.get() +"\n")
        self.txtReceipt.insert(END,'Employer Name:\t\t'+ Surname.get() +"\n\n")
        self.txtReceipt.insert(END,'Tax:\t\t'+ Tax.get() +"\n")
        self.txtReceipt.insert(END,'Pension:\t\t'+ Pension.get() +"\n")
        self.txtReceipt.insert(END,'Student Loan:\t\t'+ stdLoan.get() +"\n")
        self.txtReceipt.insert(END,'NI Number:\t\t'+ NINumber.get() +"\n")
        self.txtReceipt.insert(END,'NI Payment:\t\t'+ NIPayment.get() +"\n")
        self.txtReceipt.insert(END,'Deductions:\t\t'+ Deductions.get() +"\n")
        self.txtReceipt.insert(END,'City Weighting:\t\t'+ str('Rs %.2f'
%(CityWeighting.get())) +"\n")
        self.txtReceipt.insert(END,'\nTax Paid:\t\t'+ str('Rs %.2f' %(BasicSalary.get()))
+"\n")
        self.txtReceipt.insert(END,'OverTime:\t\t'+ "Rs" +OverTime.get() +"\n")
```

```
self.txtReceipt.insert(END,'NetPay:\t\t'+ "Rs" +NetPay.get() +"\n")

self.txtReceipt.insert(END,'GrossPay:\t\t'+ GrossPay.get() +"\n")
```

CHAPTER 6

RESULTS

- Entering the admin id and password in the login page:
  The admin enters the admin id and password in the login page if the id and the password is correct then it logs in otherwise it pops a message box which says invalid details

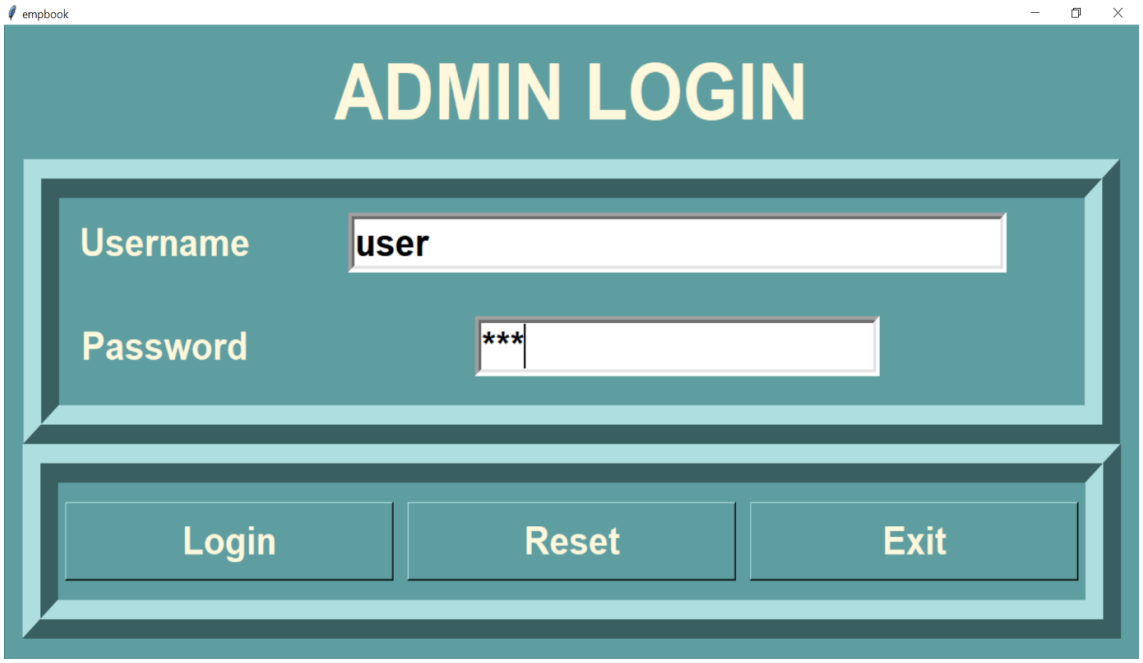

Fig 6.1 login page

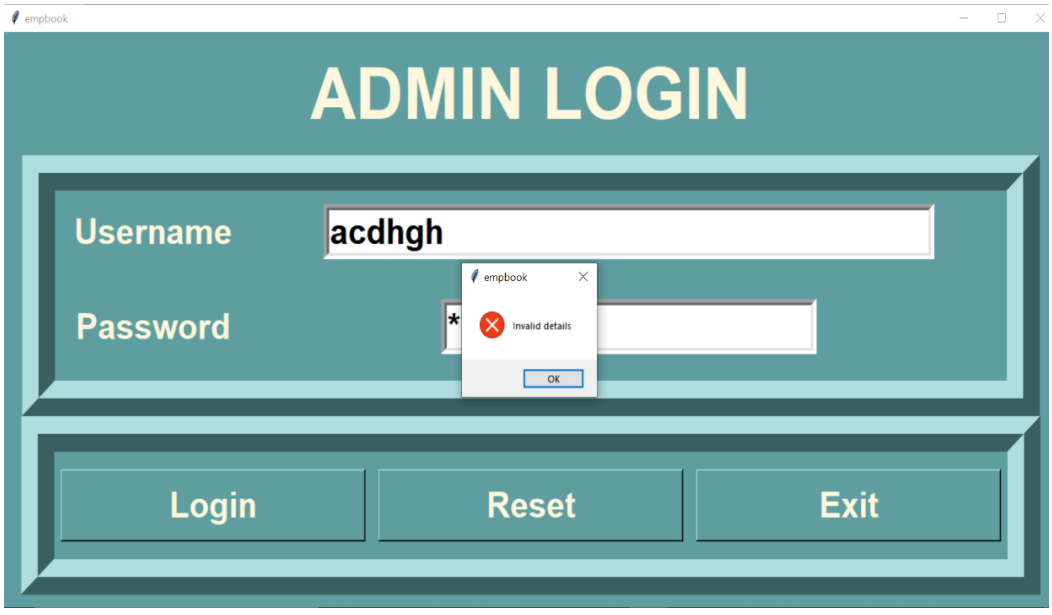

Fig 6.2 invalid details

- Adding new employee record:

  When the admin wants to add a new employee to the database, he can do it by clicking on the add new button he can store the details such as first name, surname, address, gender and mobile number of the employee in the database and the reference number will automatically be generated

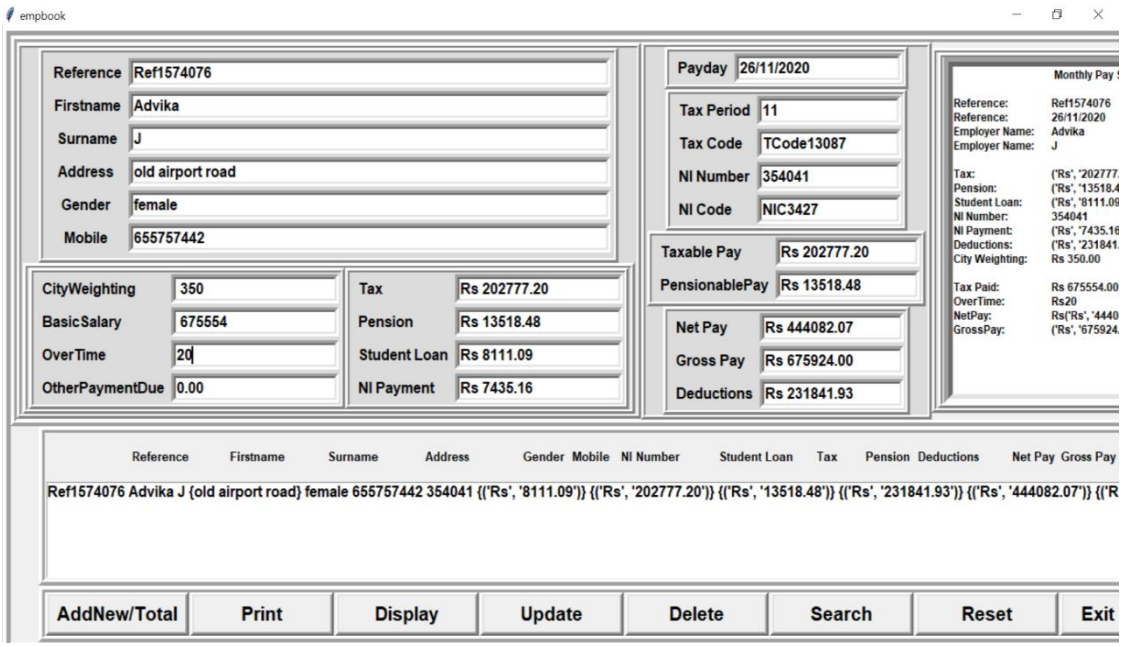

Fig 6.3 Adding new record

- Printing the pay slip:

If the admin wants to give the salary slip/ pay slip for the employees then he has to first enter the employee id and open that particular employee's record and enter details such as basic salary, number of over time worked days and city weighting and the salary slip will be generated according to the details entered and if the employee has taken any loan that amount will be deducted and the pay slip will be generated for printing

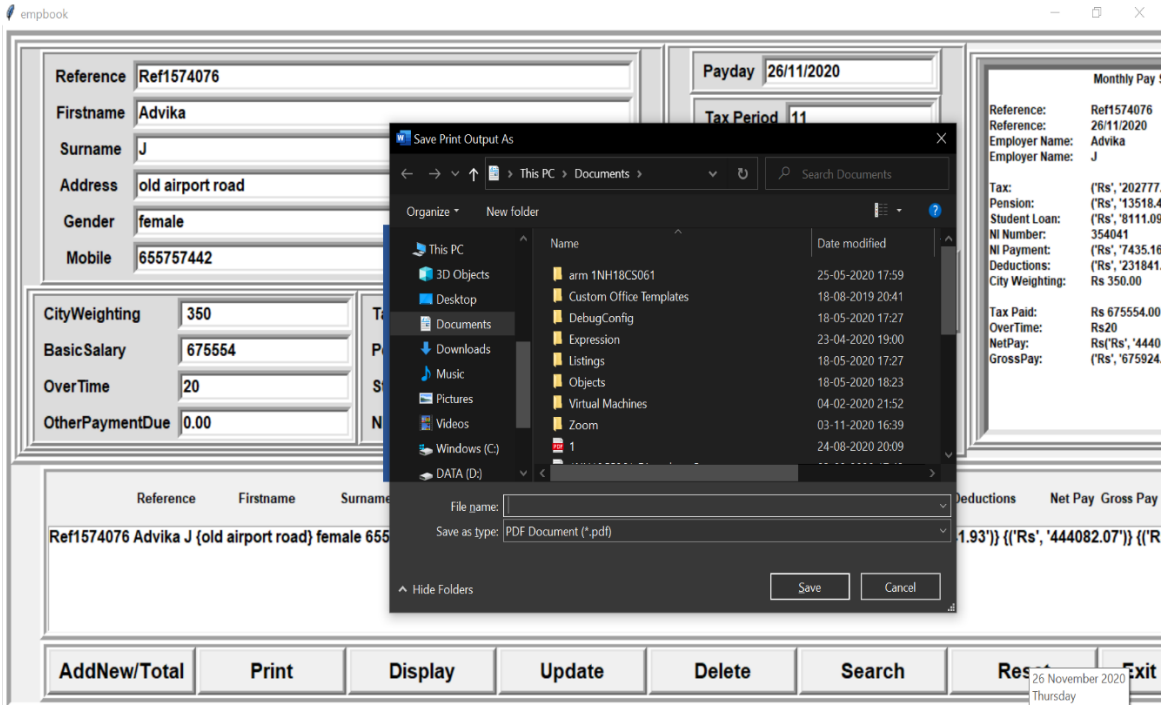

Fig 6.4 printing pay slip

- Displaying all employee records:

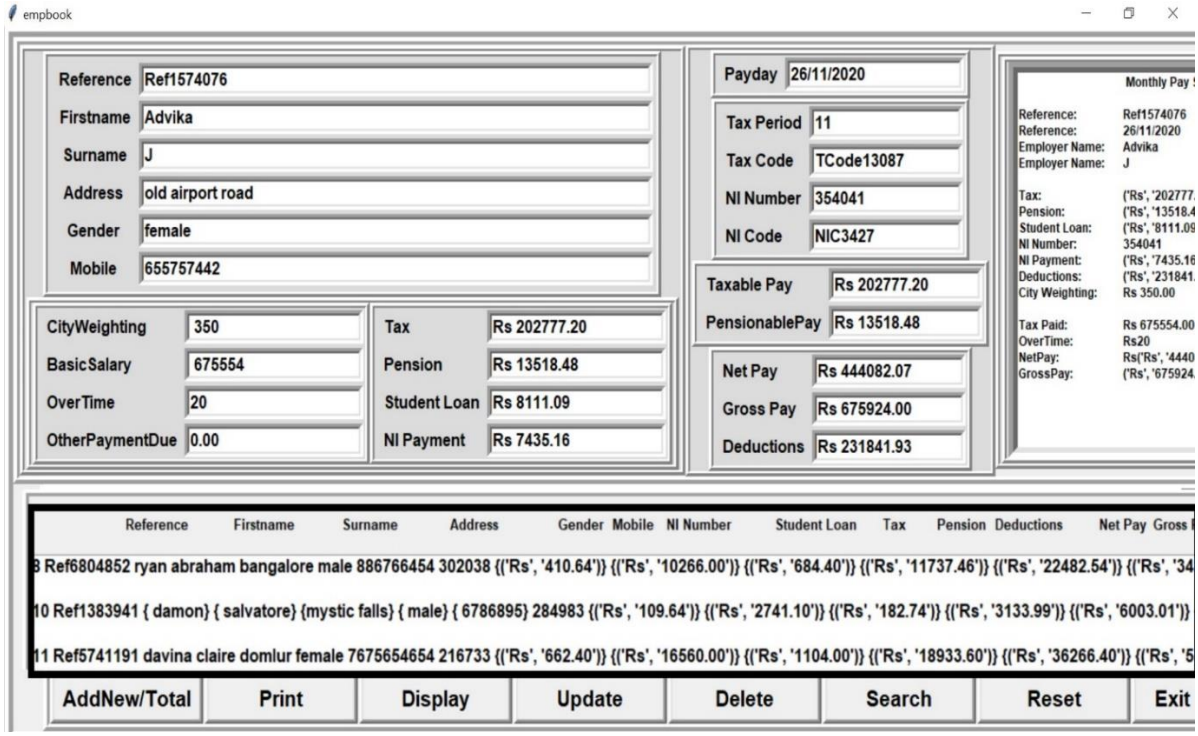

Fig 6.5 Displaying records

- Updating employee record:

We can change any details and click on the update button for updating

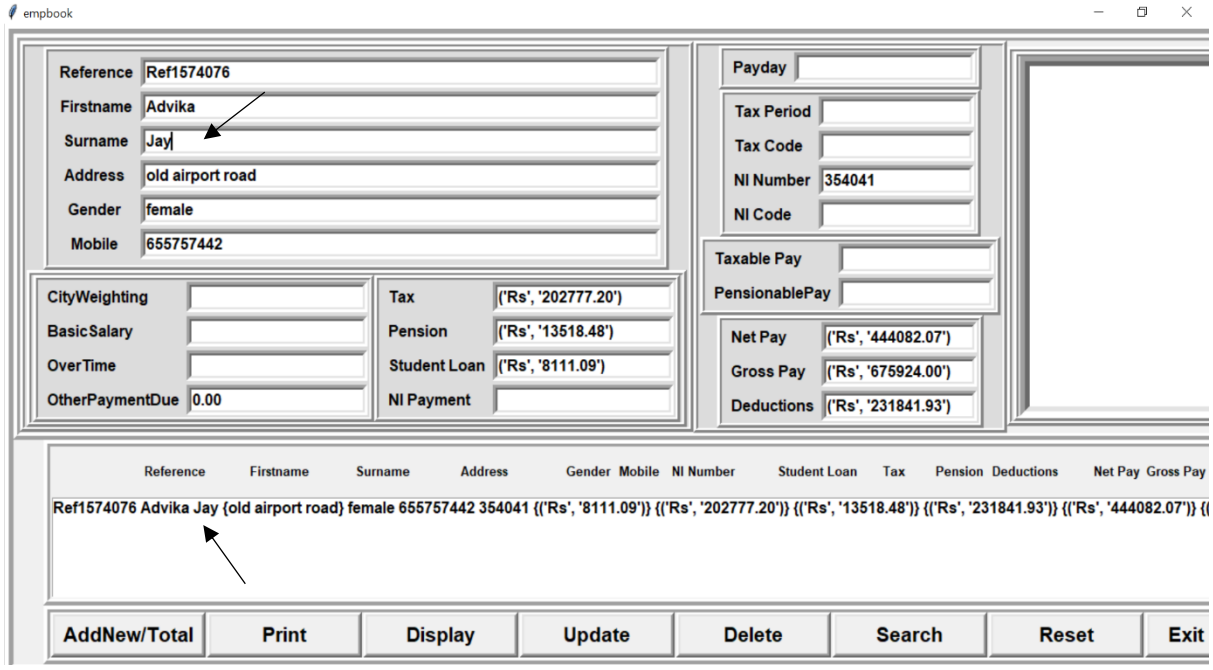

Fig 6.6 Updating a record

- Deleting employee record:
  As we can see in the below image all the employee records except damon and Davina are deleted

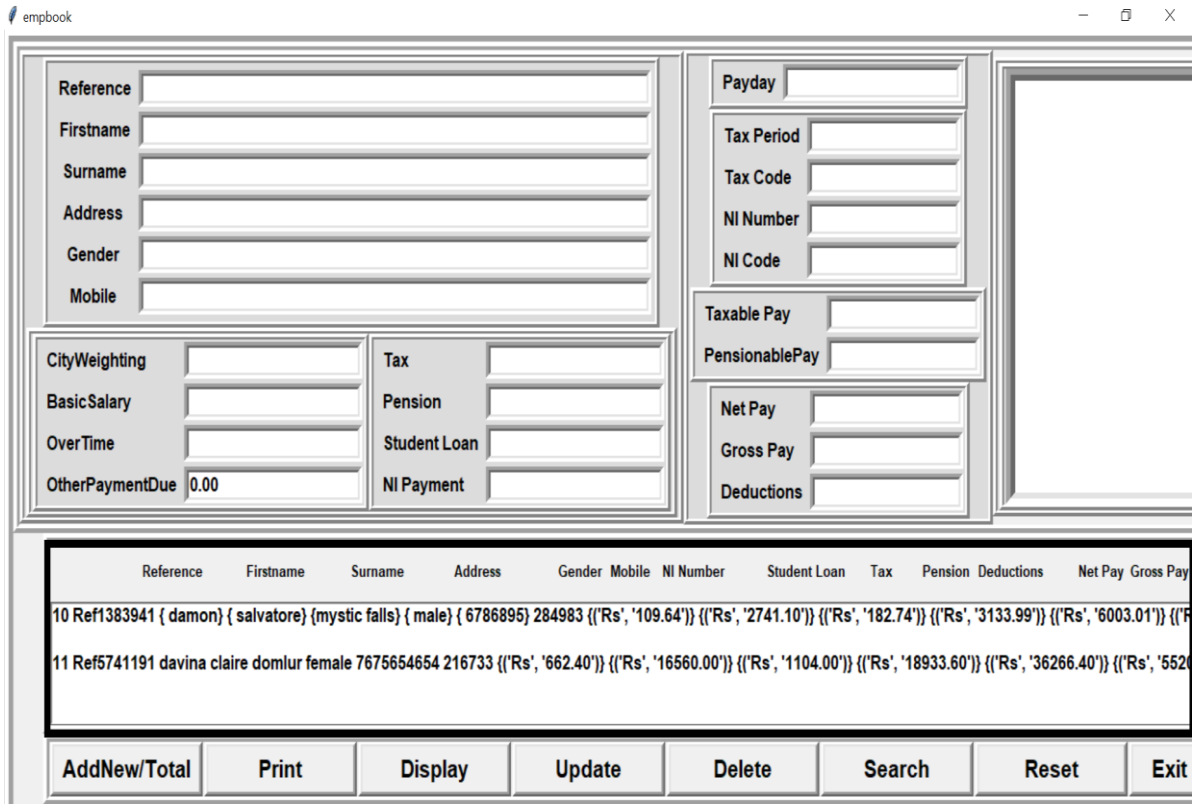

Fig 6.7 Deleting a record

- Searching an employee record:

  If a particular employee's record needs to be searched then we can enter the employee reference id and click on the search button

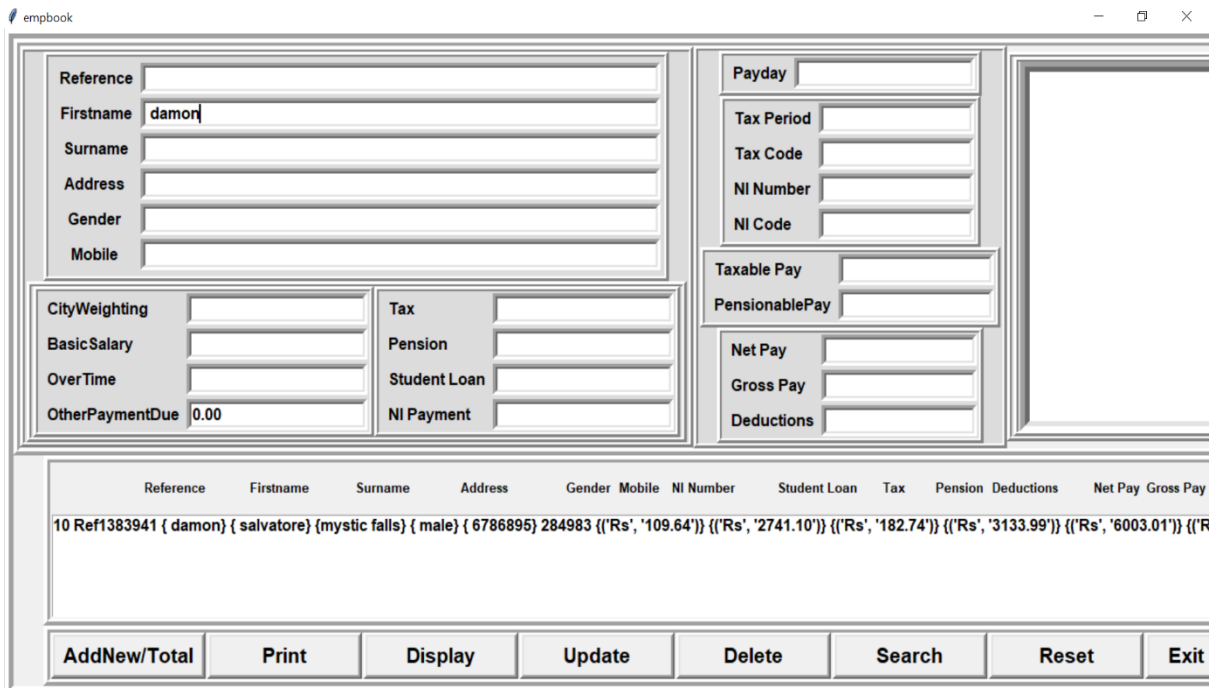

Fig 6.8 Searching a record

CHAPTER 7

CONCLUSION

The project 'EMPBOOK' is for computerizing the working of an organization. The software takes care of all the requirements such as storing the employee details, calculating the salary according to each individual and printing the pay slips. The software is run by the administration where the admin can easily access the employee details and perform operations such as adding new employee, updating the details of the current employee, deleting an employee details, searching an employee detail and printing the pay slips. The employee details are stored in a safe and secure manner in the database.

This project is a GUI based project developed using python language as front end and sqlite database as a backend with the help of tkinter module.

REFERENCES

- https://www.tutorialspoint.com/sqlite/sqlite_python.htm
- https://pynative.com/python-sqlite/
- https://www.geeksforgeeks.org/python-tkinter-tutorial/
- http://zetcode.com/python/sqlite/