

**Jose Rizal University**

**Mandaluyong City**

**Business Case**©

**For [Payroll Management System with Database ]**

**Presented to the Faculty of the**

**Department of Computer Engineering -**

**College of Computer Studies and Engineering**

**Presented by:**

**Arcangeles Lawrence**

**Lumapas Brethren**

**Detablan Omar-**

**Bequillo Christian**

**CPEC202/201G**

**In Partial Fulfillment of the Requirement for the Course of**

**Software Design (CPE C202)**

**ENGR. ROSALINA R. ESTACIO**

**March 2020**

Document Approvals

|  |  |  |  |
| --- | --- | --- | --- |
| **Role** | **Name©** | **Signature** | **Date** |
| Project Sponsor |  |  |  |
| Project Review Group |  |  |  |
| Project Manager© |  |  |  |

**Table of Contents**

1 Executive Summary 1

2 Business Problem

2.1 Environmental Analysis

2.2 Problem Analysis

3 Goals and Objectives

4 Scope

5 Software Requirements Specifications

5.1 General Design Constraints

5.2 User Characteristics

5.3 Non Functional Requirements

5.4 Functional Requirements

5.5 DATA FLOW DIAGRAM

6 USER INTERFACES AND IMPLEMENTATION

7 SOFTWARE TESTING

7.1 USER TESTS DESIGN

7.2 ALPHA TESTING

8 PROJECT MANAGEMENT AND HISTORY OF WORK

8.1 PLAN OF WORK - GANTT CHART

8.2 HISTORY OF WORK

8.3 TEAM ROLE & RESPONSIBILITY BREAKDOWN

9 REFERENCES

**10 Appendix**

*Appendix A: Transcript of client interview*

*Appendix B: Source code*

# Executive Summary

A Payroll System that enables easy and intuitive way of accessing and inputting the data in the company’s accounts and services. With its intuitive, easy and user-friendly interface, the payroll system shall gain an advantage over the current company’s MS Excel and ensure that payroll is exact and on-time.

The main objective of this project is to provide a system that will create a payroll system that calculate gross pay for each employee - hours for hourly and days worked for salary, create and file all required forms on a timely basis, and calculate payroll inclusive of potential deductions such as tax withheld, overtime, undertime, absences, employee’s benefits like SSS, PhilHealth, and also including the car loans. Payments can be in the form of cash, check, direct deposits, or debit card deposits.

The Payroll System will cover the process of preparing the pay slip of each employee, monitors loans, cash advances, charges, keeping records, and easily computes the basic pay, SSS, PAGIBIG, GSIS, gross pay, cash advances, charges, deductions, and late.

# Business Problem

The client personally asked that a payroll system is needed for their business because storing and maintaining the data of the employees manually is very difficult. The company’s payroll is also performed manually. A software system is needed to assure data security.

## 1.1 Environmental Analysis

The proposed system would let the accounting department have an easier time calculating the salary of the employees. This would reduce their work when it comes to doing the payroll and also, it is cost efficient.

The resources that are currently available for the business to adapt the proposed system are computers.

## **Problem Analysis**

According to our client, the company’s payroll is being done on a weekly basis and it is performed manually. Their main concern is that doing a manual payroll takes a lot of time.

The client wanted a system that could easily calculate the commission and salary of the employees easily without wasting too much time. The system will also be able to generate a report of the total Wage, Gross, and the Net Pay of the employees.

1.3 **Business Opportunity**

Our system is market feasible because we have communicated with one of the heads of the company and they specifically requested this type of system. Our system will benefit the organization through efficiency. The system will help the company perform payroll calculations quicker and generate accurate payslips.

## **Goals and Objectives**

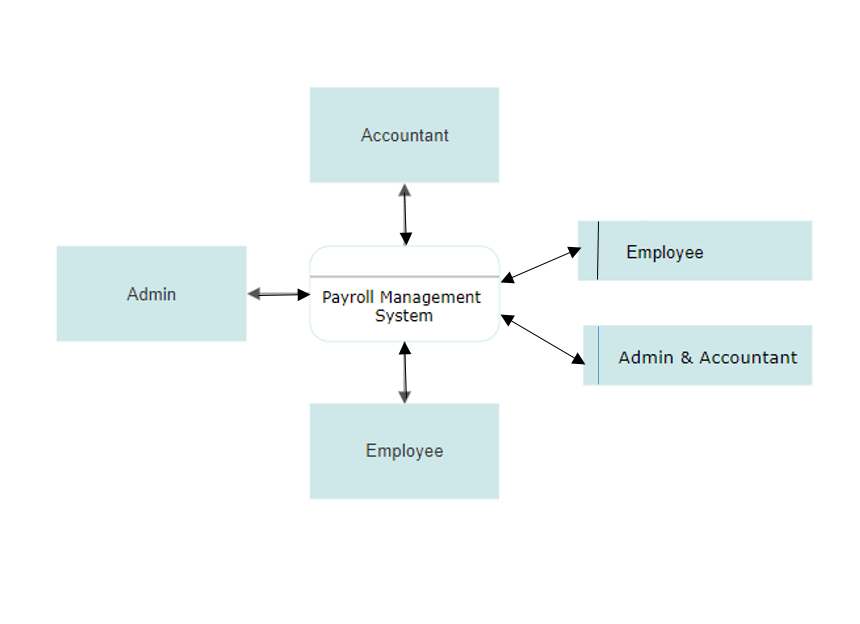
The main objective of this project is to have the users ease of access of the software and allow a way to access their account information and from the system. The Software is expected to:

1. Provide a user interface in the software to access account information more efficiently.
2. Function in a user-friendly and intuitive manner.
3. Provide employees with pay slips with a mouse click.
4. Performing a precise and quick payroll calculations according to the salary structure assigned to the employee.
5. Provide a security not comprisable.

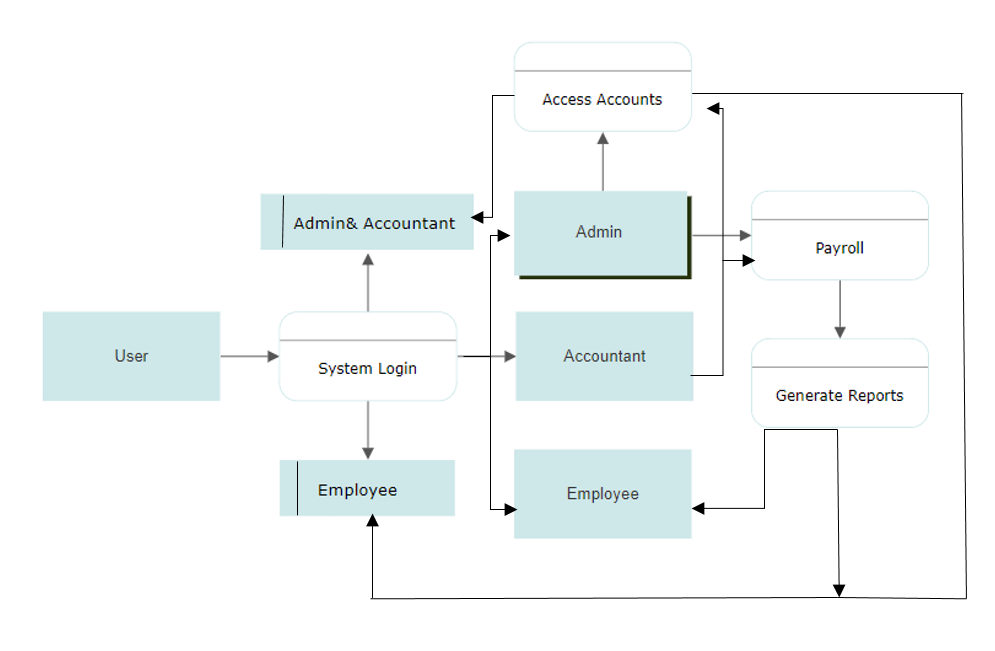
## **Scope**

The Payroll System will cover the process of preparing the pay slip of each employee, monitors loans, cash advances, charges, keeping the records and easily computes the basic pay, sss, pagibig, gsis, gross pay, cash advances, charges, deductions, and late into Python Program.

* The system will have a file management where it covers the payroll processing and reporting which includes the calculation and time-off tracking.
* The system will have a file management where it has the records of employee’s and system transaction log.
* The system will have a report consisting of a summary of sss, pagibig, gsis, and Tax refund.



**4.1 Level-0 Diagram**



**4.2 Level-1 Diagram**

1. **Software Requirements Specifications**

# 5.1 General Design Constraints

Payroll Management System Environment

The Payroll Management System will only be available as a desktop application. This desktop application will begin with the main login system and the user will be given an option whether to login as an admin, accountant or an employee. The admin login will be directed to the admin main menu, in which the admin has the option to do the payroll, add employee, update employee, delete employee, view employee records and update both the admin/accountant. The accountant login will be directed to the accountant main menu which has two options: to do the payroll or view employee records. The employee can only view his/her own account details and payroll records.

Main Login System

Payroll Management System

User Interface

Application

Admin Main Menu

Admin Login

Accountant Main Menu

Accountant Login

Employee Login

Account Details and Payroll Records

## User Characteristics

**Actors and its Roles and Goals**

|  |  |  |
| --- | --- | --- |
| **Actors** | **Roles** | **Goals** |
| **Finance & HR Head(Admin)** | **Handles the financial matters of the Body Repair & Paint Operations.** | **Manages the summary of billing statements of the company and has access to all of the records of the employees through admin login in the payroll system.** |
| **Accounting Department** | **Manages the payroll management system.** | **The main goal of the accounting department is to successfully manage the payroll system and to make sure that the system will provide the services needed. They are the people involved in adding, updating and also deleting the employee’s record in the payroll system.** |
| **Employees** | **The employees have different roles depending on the job request of the customers.** | **Their main goal is to provide services to the customers depending on their requests.** |

# User and System Requirements

**User Requirement Definition**

1. The Payroll Management System shall generate weekly reports showing the summary of the total wage, gross, and net pay of the employees of the company.
2. The system shall generate the pay slip of all the employees in the company.

**System Requirements Specifications**

1. A payroll system that is computer driven offer to manage and automate the functions of payroll system
2. The use of Computer in payroll calculation has made easier for the management to make and retain information relating to payroll.
3. The system shall generate a pay slip after the employee login their account into the computer.
4. Access of the whole system shall be restricted to authorized users/admins on a management access control list.

# Use Case Diagram



# Functional and Non-functional Requirements

5.3.1 Non-Functional Requirements

-The system will meet specified objectives like payrolls must be mathematically correct to ensure accuracy.

-The System UI will be user friendly so that the employees will all get accustomed with.

## 5.3.1.1 Operational Requirements

Usability: 95% of users will not need to read the user manual to be able to use the application.

## 5.3.1.2 Security Requirements

- Using passwords to protect the system from random users which could cause vulnerability. The client will have a pc where the employees will be able to login and access specific features depending on their role in the company.

# 5.4 Functional Requirements

## 5.4.1 Required Features

### 5.4.1 Use Case: 1

Actors: HR Head(Admin)

Basic Path:

* 1. User clicks icon for Payroll Management System.
  2. Systems prompts user to enter username and password.
  3. System prompts user to login as Admin, Accountant or Employee .
  4. User clicks Admin Login.
  5. System displays admin main menu.
  6. The Admin main menu consists of : Payroll, Add Employee, Update Employee, Delete Employee, View Employee records, Update admin and accountant, Logout.
  7. System Exit.

Actors: Accountant

Basic Path:

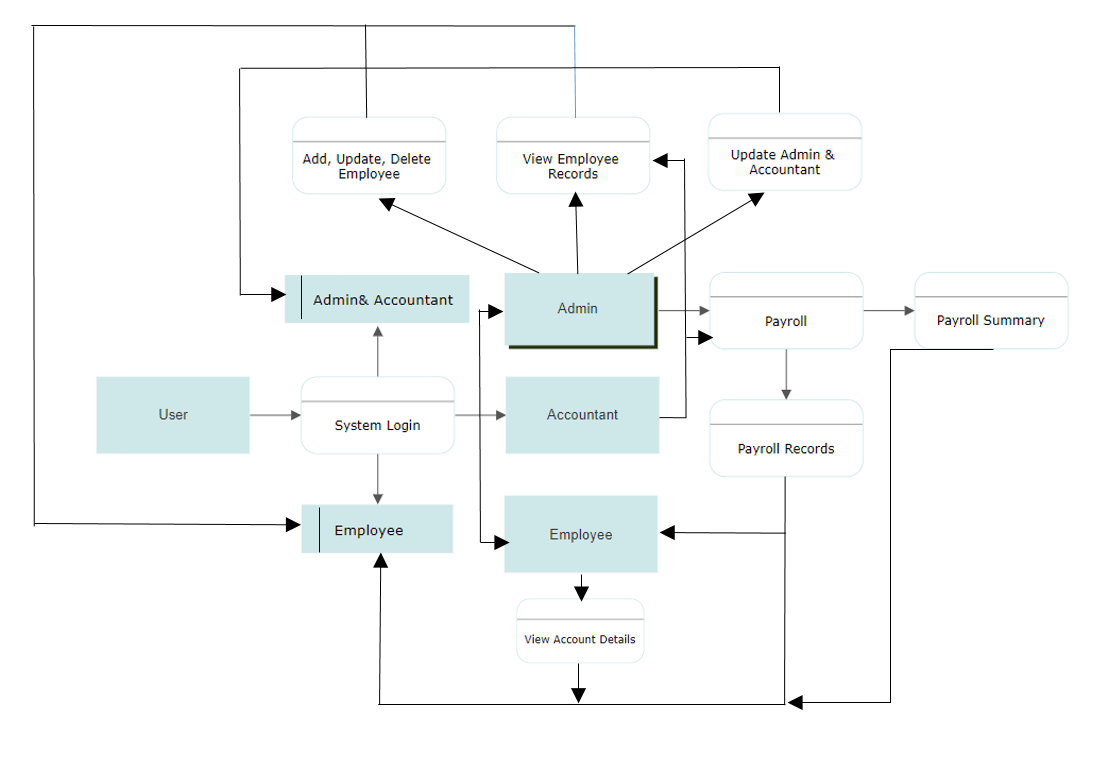
* 1. User clicks icon for Payroll Management System.
  2. Systems prompts user to enter username and password.
  3. System prompts user to login as Admin, Accountant, or Employee.
  4. User clicks Accountant Login.
  5. System displays Accountant Main Menu.
  6. The Accountant Main menu consists of: payroll, and view employee records.
  7. User logs out.
  8. System Exit.

Actors: Employee

Basic Path:

* + 1. User clicks icon for Payroll Management System.
    2. Systems prompts user to enter username and password.
    3. System prompts user to login as Admin, Accountant, or Employee.
    4. User clicks Employee Login.
    5. System displays Employee Window.
    6. System displays employee’s account details and payroll records.
    7. User logs out.
    8. System Exit.
  1. **Data Flow Diagram**

**Level-2 Diagram**



# USER INTERFACE AND IMPLEMENTATION

The following are the developed user interfaces of the project and brief discussion of its implementation.

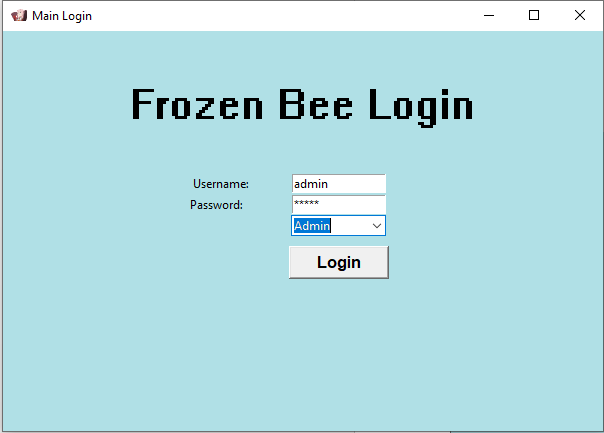
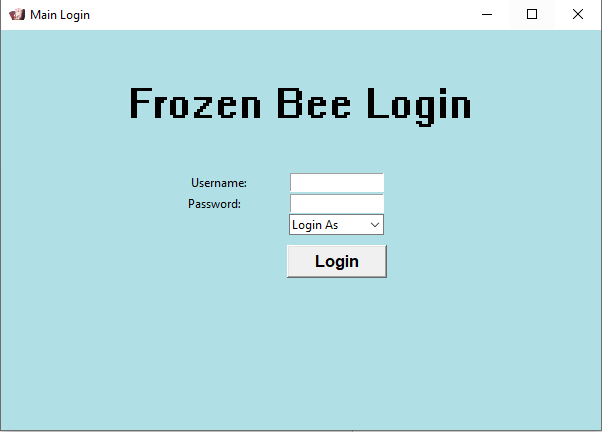


Figure 1. Main Login GUI Figure 2. Admin GUI



Figure 3. Admin Main Menu GUI

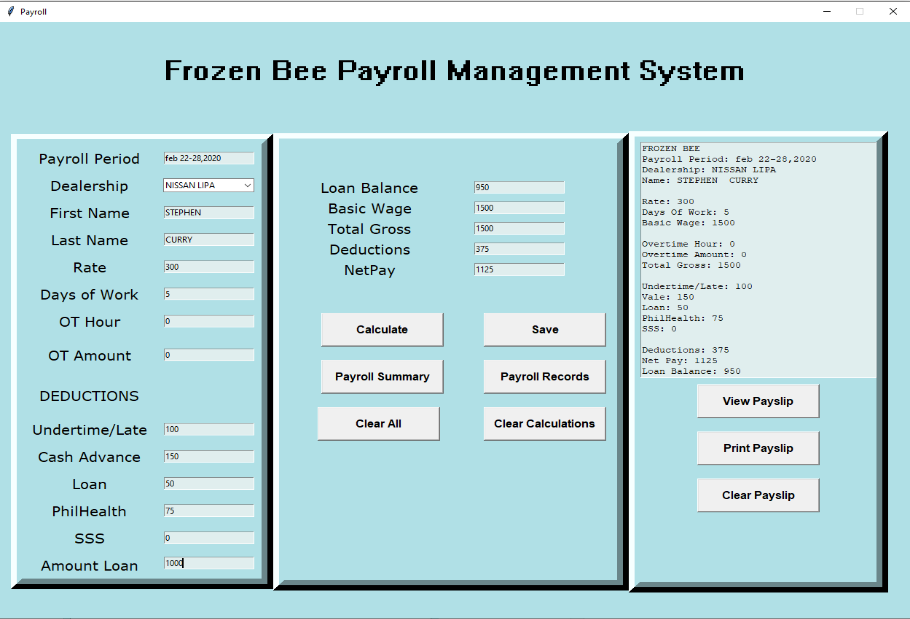


Figure 4. Payroll System GUI

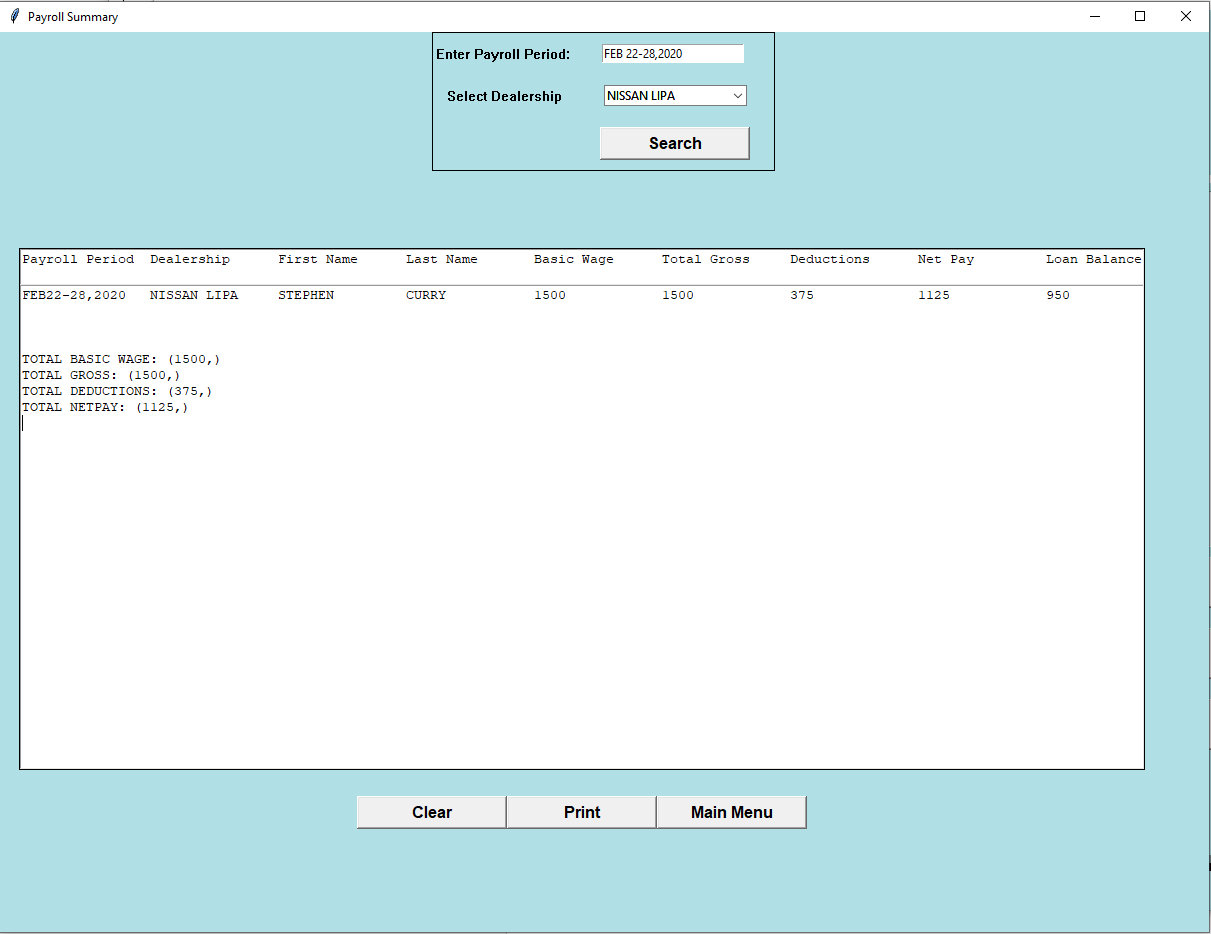


Figure 5. Payroll Summary

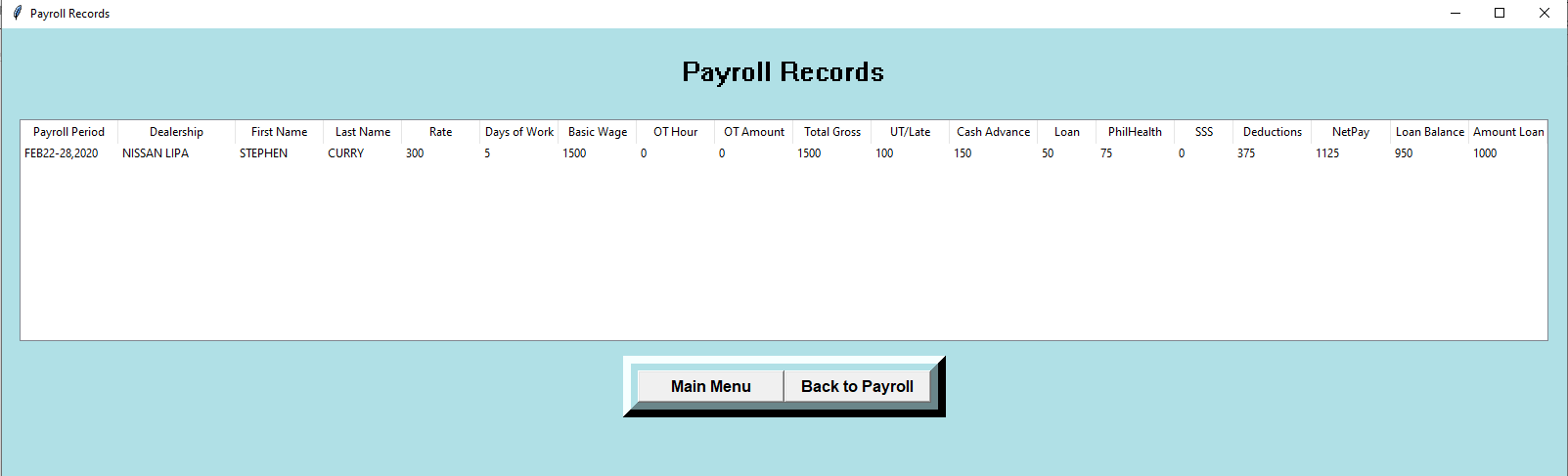


Figure 6. Payroll Record

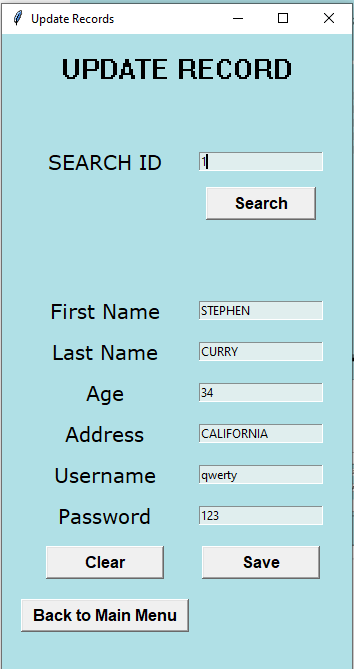
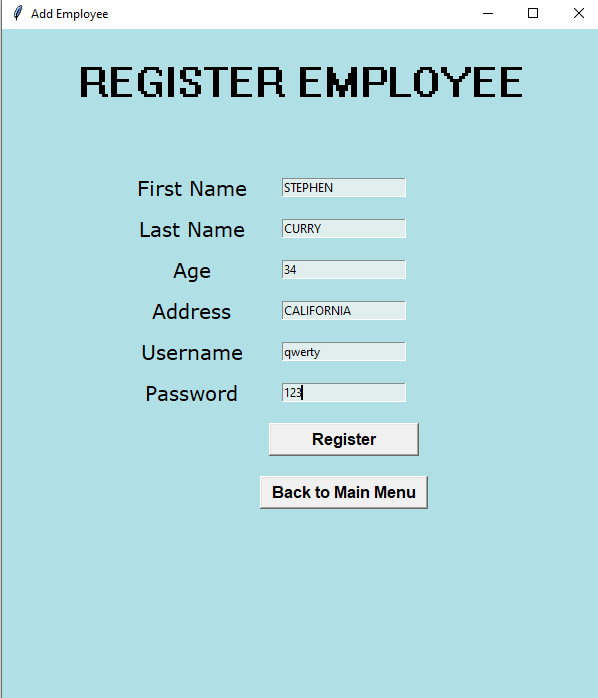


Figure 7. Add Employee

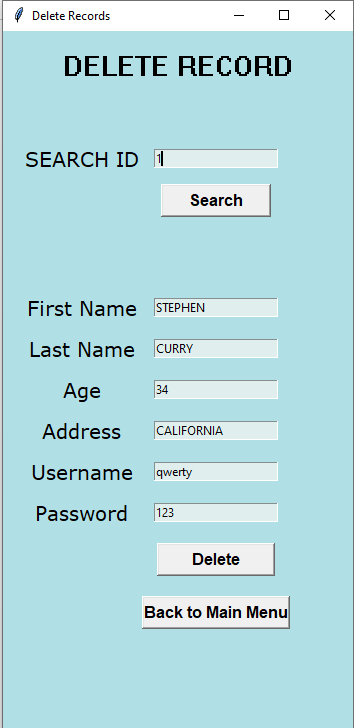


Figure 8. Update Employee

Figure 9. Delete Employee

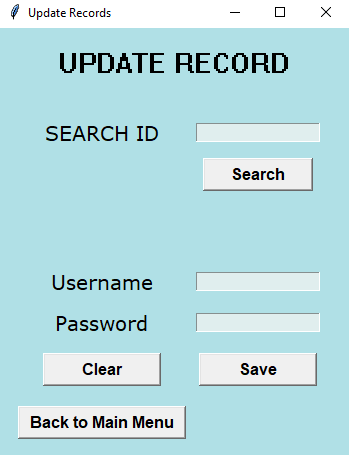
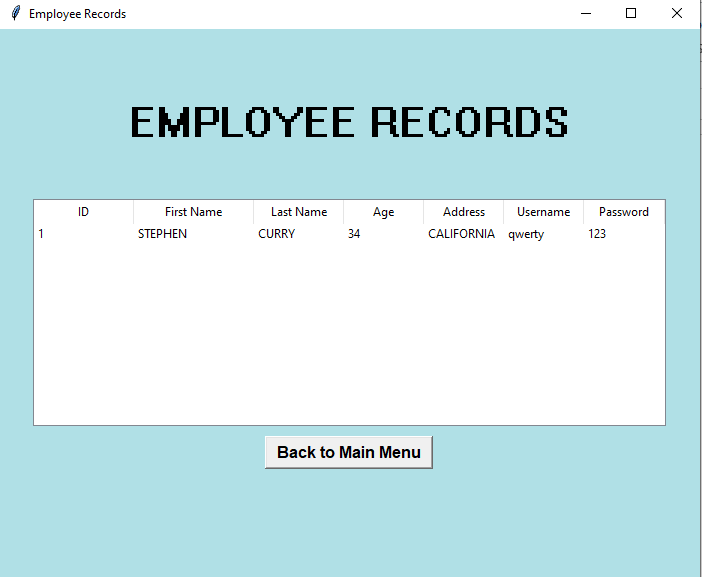


Figure 10. Employee Records

Figure 11. Update Admin and Accountant

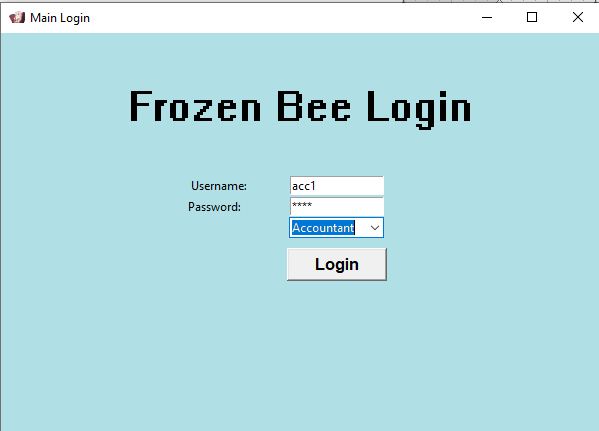


Figure 12. Accountant Login

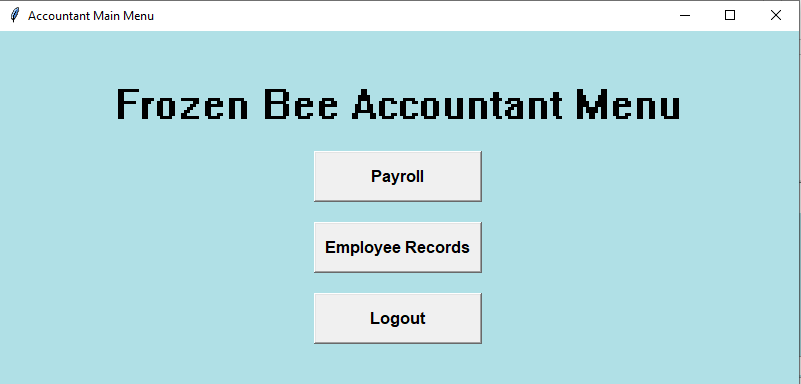
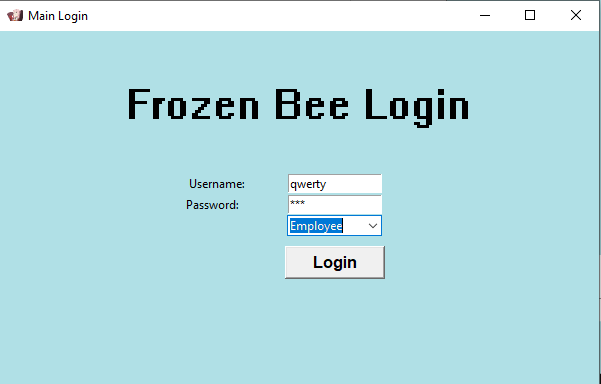


Figure 13. Accountant Main Menu Figure 14. Employee Login

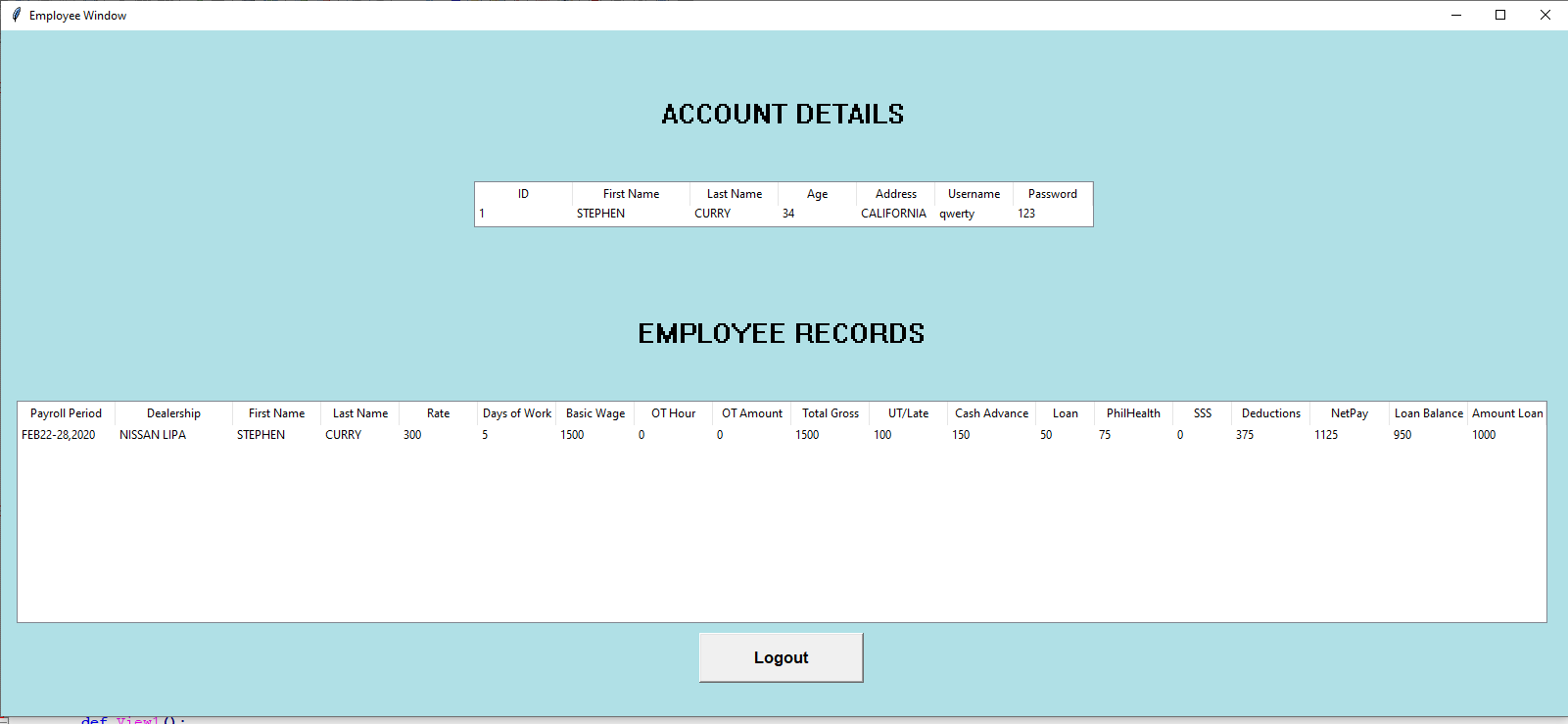


Figure 15. Employee Window

1. **SOFTWARE TESTING**
   1. **USER DESIGN OF TESTS**
   2. Employee

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Function Test** | **Test Procedure** | **Pass/Fail Criteria** | **Expected Results** |
| TC-001 | Employee login | Employee inputs his/her username & password | System recognizes that username is associated with the password  System  recognizes if the username & password are registered in the system | Employee successfully login and another interface are open  Employee cannot login because of incorrect username or password |
| TC-002 | View Account Details | Employee can view account details | System will grant permission to view account details | Employee successfully view account details.  Employee cannot view other account details. |
| TC-003 | View Employee Record | Employee can view employee record | System will grant access to employee record | Employee successfully view employee record.  Employee cannot view another employee record. |

* 1. Accountant Bequillo and Omar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Function Tested** | **Test Procedure** | **Pass/Fail Criteria** | **Expected Results** |
| TC-010 | Accountant login | Accountant inputs his/her username & password | System recognizes that username is associated with the password  System recognizes if the username & password are registered in the system | Accountant  successfully login and another interface are open  Accountant cannot login because of incorrect username or password |
| TC-020 | Accountant Payroll | Accountant inputs the exact data needed to the payroll system | System recognizes the data inputted and will compute.  System will compute precisely and print the pay slip according to the given data |  |

* 1. Admin Bequillo and Omar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test Case ID** | **Function Test** | **Test Procedure** | **Pass/Fail Criteria** | **Expected Results** |
| TC-001 | Admin login | Employee inputs his/her username & password | System recognizes that username is associated with the password  System  recognizes if the username & password are registered in the system | Admin  successfully login and another interface are open  Admin  cannot login because of incorrect username or password |
| TC-002 | Admin Payroll | Admin can access and view the payroll records and summary of it | System recognizes  And will grant permission to access the records  System will not grant access if not an admin |  |
| TC-003 | Update Employee | Admin can Update and add the employee records | System will grant access to update the records of the employee | Admin successfully view the updated record of its employee  Others cannot view/access the records |
| TC-004 | Delete Employee | Admin can Delete and Update its record | System will grant access to update and delete non-existing employee in its record |  |

* 1. **Functional & Alpha Testing**

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case ID | User | Test Case Scenario | Result vs. Expected |
| TC-001 Login | Admin | Input username & password | √ |
|  |  | Input incorrect username & correct password | √ |
|  |  | Input correct username & incorrect PW | √ |
|  | Accountant | Input username & password | √ |
|  |  | Input incorrect username & correct password | √ |
|  |  | Input correct username & incorrect PW | √ |
|  | Employee | Input username & password | √ |
|  |  | Input incorrect username & correct password | √ |
|  |  | Input correct username & incorrect PW | √ |
| TC-002 Payroll | Admin& Accountant | Input data in the payroll system | √ |
|  |  | Payroll performs computations properly. | √ |
|  |  | The clear button clears the inputted data properly. | √ |
|  |  | View payslip | √ |
| TC-003 Payroll Summary | Admin& Accountant | Input the payroll period to be searched. | √ |
|  |  | Choose the dealership. | √ |
|  |  | System displays the payroll summary of the employees under the searched payroll period and chosen dealership. | √ |
| TC-004 Payroll Records | Admin & Accountant | System displays all of the payroll records | √ |
|  |  |  |  |
| TC-005 Add Employee | Admin | Input employee details | √ |
|  |  | User clicks the register button to add the employee in the database. | √ |
| TC-006 Update Employee | Admin | Input employee ID to be updated. | √ |
|  |  | Change the employee details. | √ |
|  |  | User clicks save to update employee details. | √ |
| TC-007 Delete Employee | Admin | Input employee ID to be deleted. | √ |
|  |  | System displays the employee details from the searched ID. | √ |
|  |  | User clicks delete button to delete the employee. | √ |
| TC-008 View Employee Records | Admin & Accountant | User views the employee records. | √ |
|  |  |  |  |
| TC-009 Update Admin/Accountant | Admin | Input admin/accountant ID to be updated. | √ |
|  |  | System displays the admin/accountant details. | √ |
|  |  | User changes the username/password. | √ |
|  |  | User clicks the save button to update admin/accountant. | √ |
| TC-010 | Employee | View account details and employee payroll record. | √ |
|  |  |  |  |
| TC-011 Logout | Admin, Accountant & Employee | User clicks the logout button and gets redirected to the main login window. | √ |

1. **PROJECT MANAGEMENT AND HISTORY OF WORK**
   * 1. **PROJECT GANTT CHART**





* + 1. **HISTORY OF WORK**

Nov 19, 2019 – Nov 24, 2019

As our professor announced and discussed what is the project in our Software Design Subject. Our team discussed about our project proposal title all about a payroll management system. Because one of the team members has an acquaintance whom has a business and their problem is that they don’t have an easier way to deal with the pay slips.

Nov 25, 2019 – Dec 8, 2019

As we tackle creating the initial project proposal. We come up with the idea to interview further the client as what or what is the causes and effect of the problem. We learn that

They still not have a system and still used a system used by common people, that’s why they can’t finish the job easily.

Dec 9, 2019 - Dec 16, 2019

Despite all the shortcomings. We succeeded to finish our first initial project proposal to be check and revised by our professor.

Feb 2, 2020 – Feb 19, 2020

We revised our project documentation because the initial documentation is wrong in some ways. And with the help of the documentation template which our professor sent online our work become less hard and we can work on other stuff like the start of creating the client wants a payroll system.

Feb 20, 2020 - March 12, 2020

The team started coding the program for the payroll management database system approx. 2 after because we have an idea but we are not sure so we research for it first then apply it in our program. Our first task is to finish coding for the main menu login which we finished it on time so that the team can move on to the calculations to be precise and exact formulas in the main program. After 2.2 weeks of no rest we finally done with the program.

* 1. **ROLES & RESPONSIBILITY**

Lawrence Arcangeles

Lead programmer. Responsible for overall program design including algorithm constructions. Oversees other programmers in the team.

Christian Bequillo

Assistant programmer. Responsible for any updates or bugs within the program

Construction. One of the Program Tester.

Omar Detablan

Assistant and One of the Program Tester. Responsible for the finalization of

the documentation, the ppt and moral support of the group.

Brethren Lumapas

Encoder and Assistant. Responsible for the group PowerPoint presentation including the Project Documentation.

# Appendices

*Appendix A: Transcript of client interview*

1. **What are current business activities?**

* Car body repair and paint.
* Contractor on Automotive Industries**.**
* Dealing with some dealership companies such as: **Nissan**, **Honda**, **Chevrolet**, **Kia**, and **Suzuki.**
* Repair and restore damaged cars.
* Inline in carwash and auto detailing services.

1. **Why do they need a software system for their business?**

* A software system is needed for their business because storing and maintaining the data of the customers manually is very difficult. Also, a software system is needed to assure data security.
* “We do need the system because it will serve as a big help to my accounting system. We have more than 60 employees and it’s to do a manual payroll. Our payroll is on a weekly basis.”

1. **How would the proposed system leverage their business?**

* The proposed system would let the employees have an easier time recording and storing data. The payroll management system will help the company store and manage financial record of the employees easily.

1. **What resources are currently available for the business to adapt the proposed system?**

* The resources that are currently available for the business to adapt the proposed system are computers.

*Appendix B: Source code*

import tempfile

import os

from tkinter import\*

from tkinter import ttk

from tkinter import messagebox as ms

import sqlite3

def chooselogin1():

global chooselogin

chooselogin=Tk()

chooselogin.title('Main Login')

chooselogin.geometry('600x400')

chooselogin.configure(background='powder blue')

chooselogin.iconbitmap('zero.ico')

#chooselogin.resizable(0,0)

def Database():

global conn, c

conn = sqlite3.connect("admin.db")

c = conn.cursor()

c.execute("""CREATE TABLE IF NOT EXISTS member (

mem\_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,

fname TEXT,

lname TEXT,

age TEXT,

address TEXT,

username TEXT,

password TEXT)""")

c.execute("SELECT \* FROM `member` WHERE `username` = 'admin' AND `password` = 'admin'")

if c.fetchone() is None:

c.execute("INSERT INTO `member` (username, password) VALUES('admin', 'admin')")

conn.commit()

def Database2():

global conn, c

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

c.execute("""CREATE TABLE IF NOT EXISTS member2 (

mem\_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,

fname TEXT,

lname TEXT,

age TEXT,

address TEXT,

username TEXT,

password TEXT)""")

c.execute("SELECT \* FROM member2 WHERE username = ? AND password = ?", (username.get(),password.get(),))

#=======================for accountant========================

def Database3():

global conn, c

conn = sqlite3.connect("admin.db")

c = conn.cursor()

c.execute("""CREATE TABLE IF NOT EXISTS member2 (

mem\_id INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,

fname TEXT,

lname TEXT,

age TEXT,

address TEXT,

username TEXT,

password TEXT)""")

c.execute("SELECT \* FROM `member2` WHERE `username` = 'admin' AND `password` = 'admin'")

if c.fetchall() is None:

c.execute("INSERT INTO `member2` (username, password) VALUES('acc1', 'acc1')")

c.execute("INSERT INTO `member2` (username, password) VALUES('acc2', 'acc2')")

c.execute("INSERT INTO `member2` (username, password) VALUES('acc3', 'acc3')")

conn.commit()

return

conn.commit()

#=======================ADMIN LOGIN==================================

def Login():

global Login

select=drop.get()

if select=="Admin":

Database()

if username.get() == "" or password.get() == "":

lbl\_text.config(text="Please complete the required field!", fg="red")

else:

c.execute("SELECT \* FROM `member` WHERE `username` = ? AND `password` = ?", (username.get(), password.get()))

if c.fetchone() is not None:

chooselogin.destroy()

main2()

#lbl\_text.config(text="")

else:

lbl\_text.config(text="Invalid username or password", fg="red")

username.set("")

password.set("")

c.close()

conn.close()

#=======================ACCOUNTANT LOGIN==================================

def Login2():

global Login2

select=drop.get()

if select=="Accountant":

Database3()

if username.get() == "" or password.get() == "":

lbl\_text.config(text="Please complete the required field!", fg="red")

else:

c.execute("SELECT \* FROM `member2` WHERE `username` = ? AND `password` = ?", (username.get(), password.get()))

if c.fetchone() is not None:

chooselogin.destroy()

accountantmain()

#lbl\_text.config(text="")

else:

lbl\_text.config(text="Invalid username or password", fg="red")

username.set("")

password.set("")

c.close()

conn.close()

#=====================EMPLOYEE LOGIN======================================

def Login3():

select=drop.get()

if select=="Employee":

Database2()

if c.fetchone() is not None:

chooselogin.destroy()

empmain1()

username.set("")

password.set("")

#lbl\_text.config(text="")

else:

#lbl\_text.config(text="Invalid username or password", fg="red")

username.set("")

password.set("")

c.close()

conn.close()

def logincommand():

select = drop.get()

if select=="Admin":

Login()

elif select=="Employee":

Login3()

#===========================accountant==================================

def accountantmain():

global accmain

accmain=Tk()

accmain.title('Accountant Main Menu')

accmain.geometry('800x400')

accmain.configure(background='powder blue')

#empmain.resizable(0,0)

#==========================Frames=============================

Tops=Frame(accmain, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(accmain, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#button

choice1=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Payroll",command=pr).grid(row=3, column=1,pady=10,padx=(200))

choice2=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Employee Records",command=emprecords2).grid(row=4, column=1,pady=10,padx=30)

choice3=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Logout",command=logout2).grid(row=5, column=1,pady=10,padx=(200))

#labels

adminlabel=Label(Tops, text='Frozen Bee Accountant Menu',font=('system',30),bg='powder blue')

adminlabel.grid(row=2,column=1,pady=(30,10),padx=50)

#================EMPLOYEE RECORDS in ADMIN MAIN MENU==================================

def emprecords2():

global emp2

accmain.destroy()

emp2=Tk()

emp2.title('Employee Records')

emp2.geometry('700x600')

emp2.configure(background='powder blue')

#emp.resizable(0,0)

def View():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

rows=c.execute("SELECT \* FROM member2")

rows = c.fetchall()

for row in rows:

tree.insert("",END, values = row)

#============================FRAMES===============================================

Tops=Frame(emp2, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(emp2, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#============================LABELS===============================================

addlabel=Label(Tops, text='EMPLOYEE RECORDS',font=('system',30),bg='powder blue')

addlabel.grid(row=0,column=1,pady=50)

#===========================BUTTONS=====================================

back=Button(Tops, fg="black",padx=10, font=('arial', 12, 'bold'), width=14,

text="Back to Main Menu",command=back5).grid(row=3, column=1,pady=(10,210))

#===========================TREEVIEW================================================

tree= ttk.Treeview(Tops, column=("column1", "column2", "column3", "column4", "column5", "column6", "column7"), show='headings')

tree.heading("#1", text="ID")

tree.column("#1", width = 100, anchor = "w")

tree.heading("#2", text="First Name")

tree.column("#2", width = 120, anchor = "w")

tree.heading("#3", text="Last Name")

tree.column("#3", width = 90, anchor = "w")

tree.heading("#4", text="Age")

tree.column("#4", width = 80, anchor = "w")

tree.heading("#5", text="Address")

tree.column("#5", width = 80, anchor = "w")

tree.heading("#6", text="Username")

tree.column("#6", width = 80, anchor = "w")

tree.heading("#7", text="Password")

tree.column("#7", width = 80, anchor = "w")

tree.grid(row=2,column=1)

View()

#=============EMPLOYEE MAIN MENU==============================#

def empmain1():

global empmain

empmain=Tk()

empmain.title('Employee Window')

empmain.geometry('1600x700')

empmain.configure(background='powder blue')

#empmain.resizable(0,0)

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

def View1():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

#rows=c.execute("SELECT \* FROM member2")

rows=c.execute("SELECT \*,oid FROM member2 WHERE `username` = ? AND `password` = ?", (username.get(), password.get()))

rows = c.fetchall()

for row in rows:

tree1.insert("",END, values = row)

def View2():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

#rows=c.execute("SELECT \* FROM member2")

rows=c.execute("SELECT \* FROM payroll INNER JOIN member2 ON payroll.lname=member2.lname WHERE `username` = ? AND `password` = ?", (username.get(), password.get()))

rows = c.fetchall()

for row in rows:

tree.insert("",END, values = row)

#============================FRAMES===============================================

Tops=Frame(empmain, width=1350, height=1, bd=16, bg='powder blue')

Tops.pack(side=TOP)

Tops2=Frame(empmain, width=1350, height=50, bd=16, bg='powder blue')

Tops2.pack(side=LEFT)

bot=Frame(empmain, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#============================LABELS===============================================

addlabel=Label(Tops, text='ACCOUNT DETAILS',font=('system',20),bg='powder blue')

addlabel.grid(row=0,column=1,pady=50)

addlabel2=Label(Tops2, text='EMPLOYEE RECORDS',font=('system',20),bg='powder blue')

addlabel2.grid(row=0,column=1,pady=50)

#====================================button=================================

choice1=Button(Tops2, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Logout",command=logout3).grid(row=3, column=1,pady=10,padx=(200))

#===========================TREEVIEW================================================

s = ttk.Style()

s.configure('MyStyle.Treeview', rowheight=2)

tree1= ttk.Treeview(Tops, column=("column1", "column2", "column3", "column4", "column5", "column6", "column7"), show='headings', style='MyStyle.Treeview')

tree1.heading("#1", text="ID")

tree1.column("#1", width = 100, anchor = "w")

tree1.heading("#2", text="First Name")

tree1.column("#2", width = 120, anchor = "w")

tree1.heading("#3", text="Last Name")

tree1.column("#3", width = 90, anchor = "w")

tree1.heading("#4", text="Age")

tree1.column("#4", width = 80, anchor = "w")

tree1.heading("#5", text="Address")

tree1.column("#5", width = 80, anchor = "w")

tree1.heading("#6", text="Username")

tree1.column("#6", width = 80, anchor = "w")

tree1.heading("#7", text="Password")

tree1.column("#7", width = 80, anchor = "w")

tree1.grid(row=1,column=1)

View1()

tree= ttk.Treeview(Tops2, column=("column1", "column2", "column3", "column4", "column5", "column6", "column7", "column8", "column9", "column10"

, "column11", "column12", "column13", "column14", "column15", "column16", "column17", "column18", "column19"), show='headings')

tree.heading("#1", text="Payroll Period")

tree.column("#1", width = 100, anchor = "w")

tree.heading("#2", text="Dealership")

tree.column("#2", width = 120, anchor = "w")

tree.heading("#3", text="First Name")

tree.column("#3", width = 90, anchor = "w")

tree.heading("#4", text="Last Name")

tree.column("#4", width = 80, anchor = "w")

tree.heading("#5", text="Rate")

tree.column("#5", width = 80, anchor = "w")

tree.heading("#6", text="Days of Work")

tree.column("#6", width = 80, anchor = "w")

tree.heading("#7", text="Basic Wage")

tree.column("#7", width = 80, anchor = "w")

tree.heading("#8", text="OT Hour")

tree.column("#8", width = 80, anchor = "w")

tree.heading("#9", text="OT Amount")

tree.column("#9", width = 80, anchor = "w")

tree.heading("#10", text="Total Gross")

tree.column("#10", width = 80, anchor = "w")

tree.heading("#11", text="UT/Late")

tree.column("#11", width = 80, anchor = "w")

tree.heading("#12", text="Cash Advance")

tree.column("#12", width = 90, anchor = "w")

tree.heading("#13", text="Loan")

tree.column("#13", width = 60, anchor = "w")

tree.heading("#14", text="PhilHealth")

tree.column("#14", width = 80, anchor = "w")

tree.heading("#15", text="SSS")

tree.column("#15", width = 60, anchor = "w")

tree.heading("#16", text="Deductions")

tree.column("#16", width = 80, anchor = "w")

tree.heading("#17", text="NetPay")

tree.column("#17", width = 80, anchor = "w")

tree.heading("#18", text="Loan Balance")

tree.column("#18", width = 80, anchor = "w")

tree.heading("#19", text="Amount Loan")

tree.column("#19", width = 80, anchor = "w")

tree.grid(row=2,column=1)

View2()

#=============ADMIN MAIN MENU==============================#

def main2():

global main

main=Tk()

main.title('Admin Main Menu')

main.geometry('600x750')

main.configure(background='powder blue')

main.resizable(0,0)

#==========================Frames=============================

Tops=Frame(main, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(main, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#button

choice1=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Payroll",command=pr).grid(row=1, column=1,pady=10,padx=(200))

choice2=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Employee Records",command=emprecords).grid(row=5, column=1,pady=10,padx=30)

choice3=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Add Employee",command=addemp1).grid(row=2, column=1,pady=10,padx=30)

choice4=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Update Employee",command=updaterecord).grid(row=3, column=1,pady=10,padx=30)

choice5=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Delete Employee",command=delrec).grid(row=4, column=1,pady=10)

choice6=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Update Admin",command=updaterecord2).grid(row=6, column=1,pady=10)

choice7=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Update Accountant",command=updaterecord3).grid(row=7, column=1,pady=10)

choice8=Button(bot, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Logout",command=logout).grid(row=8, column=1,pady=(10,210),padx=30)

#labels

adminlabel=Label(Tops, text='Frozen Bee Admin Menu',font=('system',30),bg='powder blue')

adminlabel.grid(row=2,column=1,pady=(30,10),padx=50)

def logout():

main.destroy()

chooselogin1()

def logout2():

accmain.destroy()

chooselogin1()

def logout3():

empmain.destroy()

chooselogin1()

#==============add employee===================================#

def addemp1():

global addemp

main.destroy()

addemp=Tk()

addemp.title('Add Employee')

addemp.geometry('600x700')

addemp.configure(background='powder blue')

#addemp.resizable(0,0)

def register():

#database connection

Database2()

#insert into tables

c.execute("INSERT INTO member2 (fname,lname,age,address,username,password )VALUES( :fname, :lname,:age, :address, :username, :password)",

{

#'mem\_id':identry.get(),

'fname':fnameentry.get().upper(),

'lname':lnameentry.get().upper(),

'age':ageentry.get(),

'address':addressentry.get(),

'username':usernameentry.get(),

'password':passwordentry.get()})

#commit changes

conn.commit()

#close connection

conn.close()

#identry.delete(0,END)

fnameentry.delete(0,END)

lnameentry.delete(0,END)

ageentry.delete(0,END)

addressentry.delete(0,END)

usernameentry.delete(0,END)

passwordentry.delete(0,END)

addemp.destroy()

main2()

#=========================VARIABLES===================================

username = StringVar()

password = StringVar()

fname= StringVar()

lname= StringVar()

age= StringVar()

address= StringVar()

#===========================FRAMES==================================

Tops=Frame(addemp, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

lframe=Frame(addemp, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=16,bg='powder blue')

inlframel.pack(side=LEFT)

#========================LABELS===================================

addlabel=Label(Tops, text='REGISTER EMPLOYEE',font=('system',30),bg='powder blue')

addlabel.grid(row=1,column=1,pady=10)

#idlabel=Label(inlframel, text='ID',font=('verdana',15),bg='powder blue')

#idlabel.grid(row=0,column=1,pady=5,padx=(100,10))

firstnamelabel=Label(inlframel, text='First Name',font=('verdana',15),bg='powder blue')

firstnamelabel.grid(row=1,column=1,pady=5,padx=(100,10))

lastnamelabel=Label(inlframel, text='Last Name',font=('verdana',15),bg='powder blue')

lastnamelabel.grid(row=2,column=1,pady=5,padx=(100,10))

agelabel=Label(inlframel, text='Age',font=('verdana',15),bg='powder blue')

agelabel.grid(row=3,column=1,pady=5,padx=(100,10))

addresslabel=Label(inlframel, text='Address',font=('verdana',15),bg='powder blue')

addresslabel.grid(row=4,column=1,pady=5,padx=(100,10))

usernamelabel=Label(inlframel, text='Username',font=('verdana',15),bg='powder blue')

usernamelabel.grid(row=5,column=1,pady=5,padx=(100,10))

passwordlabel=Label(inlframel, text='Password',font=('verdana',15),bg='powder blue')

passwordlabel.grid(row=6,column=1,pady=5,padx=(100,10))

#============================ENTRY BOXES==============================

#identry = Entry(inlframel, width=20,bg='azure2')

#identry .grid(row=0,column=2,pady=10,padx=10)

fnameentry = Entry(inlframel, width=20,bg='azure2')

fnameentry .grid(row=1,column=2,pady=10,padx=10)

lnameentry = Entry(inlframel, width=20,bg='azure2')

lnameentry.grid(row=2,column=2,pady=10,padx=10)

ageentry = Entry(inlframel, width=20,bg='azure2')

ageentry.grid(row=3,column=2,pady=10,padx=10)

addressentry = Entry(inlframel, width=20,bg='azure2')

addressentry.grid(row=4,column=2,pady=10,padx=10)

usernameentry = Entry(inlframel, width=20,bg='azure2')

usernameentry.grid(row=5,column=2,pady=5,padx=10)

passwordentry = Entry(inlframel, width=20,bg='azure2')

passwordentry.grid(row=6,column=2,pady=10,padx=10)

#===========================BUTTONS=====================================

mainmenubutton=Button(inlframel, fg="black",padx=10, font=('arial', 12, 'bold'), width=14,

text="Back to Main Menu",command=back2).grid(row=8, column=2,pady=(10,210))

registerbutton=Button(inlframel, fg="black", font=('arial', 12, 'bold'), width=14,

text="Register",command=register).grid(row=7, column=2,pady=10)

#================EMPLOYEE RECORDS in ADMIN MAIN MENU==================================

def emprecords():

global emp

main.destroy()

emp=Tk()

emp.title('Employee Records')

emp.geometry('700x600')

emp.configure(background='powder blue')

#emp.resizable(0,0)

def View():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

rows=c.execute("SELECT \* FROM member2")

rows = c.fetchall()

for row in rows:

tree.insert("",END, values = row)

#============================FRAMES===============================================

Tops=Frame(emp, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(emp, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#============================LABELS===============================================

addlabel=Label(Tops, text='EMPLOYEE RECORDS',font=('system',30),bg='powder blue')

addlabel.grid(row=0,column=1,pady=50)

#===========================BUTTONS=====================================

back=Button(Tops, fg="black",padx=10, font=('arial', 12, 'bold'), width=14,

text="Back to Main Menu",command=back1).grid(row=3, column=1,pady=(10,210))

#===========================TREEVIEW================================================

tree= ttk.Treeview(Tops, column=("column1", "column2", "column3", "column4", "column5", "column6", "column7"), show='headings')

tree.heading("#1", text="ID")

tree.column("#1", width = 100, anchor = "w")

tree.heading("#2", text="First Name")

tree.column("#2", width = 120, anchor = "w")

tree.heading("#3", text="Last Name")

tree.column("#3", width = 90, anchor = "w")

tree.heading("#4", text="Age")

tree.column("#4", width = 80, anchor = "w")

tree.heading("#5", text="Address")

tree.column("#5", width = 80, anchor = "w")

tree.heading("#6", text="Username")

tree.column("#6", width = 80, anchor = "w")

tree.heading("#7", text="Password")

tree.column("#7", width = 80, anchor = "w")

tree.grid(row=2,column=1)

View()

def back1():

emp.destroy()

main2()

def back2():

addemp.destroy()

main2()

def back3():

update.destroy()

main2()

def back4():

del1.destroy()

main2()

def back5():

emp2.destroy()

accountantmain()

def back6():

update22.destroy()

main2()

def back7():

update33.destroy()

main2()

#========================UPDATE RECORDS=========================

def updaterecord():

global update

main.destroy()

update=Tk()

update.title('Update Records')

update.geometry('350x700')

update.configure(background='powder blue')

#update.resizable(0,0)

#=================UPDATE RECORD FUNCTION=========================

def update1():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

recordid=identry.get()

c.execute("SELECT \* FROM member2 WHERE mem\_id = ? ", (identry.get(),))

records = c.fetchall()

for record in records:

fnameentry.insert(0,record[1])

lnameentry.insert(0,record[2])

ageentry.insert(0,record[3])

addressentry.insert(0,record[4])

usernameentry.insert(0,record[5])

passwordentry.insert(0,record[6])

conn.commit()

def clearvp():

fnameentry.delete(0,END)

lnameentry.delete(0,END)

ageentry.delete(0,END)

addressentry.delete(0,END)

usernameentry.delete(0,END)

passwordentry.delete(0,END)

def update2():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

recordid=identry.get()

c.execute("""UPDATE member2 SET

fname=:fname2,

lname=:lname2,

age=:age2,

address=:address2,

username=:username2,

password=:password2

WHERE mem\_id= :mem\_id""",

{'fname2':fnameentry.get(),

'lname2':lnameentry.get(),

'age2':ageentry.get(),

'address2':addressentry.get(),

'username2':usernameentry.get(),

'password2':passwordentry.get(),

'mem\_id':recordid

})

conn.commit()

conn.close()

ms.showinfo(" ","Record Updated")

update.destroy()

main2()

#=============================FRAMES=============================

Tops=Frame(update, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(update, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

lframe=Frame(update, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=3, bg='powder blue')

inlframel.pack(side=TOP)

#==============================LABELS=============================

addlabel=Label(Tops, text='UPDATE RECORD',font=('system',20),bg='powder blue')

addlabel.grid(row=1,column=1)

idlabel=Label(inlframel, text='SEARCH ID',font=('verdana',15),bg='powder blue')

idlabel.grid(row=2,column=1)

firstnamelabel=Label(inlframel, text='First Name',font=('verdana',15),bg='powder blue')

firstnamelabel.grid(row=4,column=1,pady=5)

lastnamelabel=Label(inlframel, text='Last Name',font=('verdana',15),bg='powder blue')

lastnamelabel.grid(row=5,column=1,pady=5)

agelabel=Label(inlframel, text='Age',font=('verdana',15),bg='powder blue')

agelabel.grid(row=6,column=1,pady=5)

addresslabel=Label(inlframel, text='Address',font=('verdana',15),bg='powder blue')

addresslabel.grid(row=7,column=1,pady=5)

usernamelabel=Label(inlframel, text='Username',font=('verdana',15),bg='powder blue')

usernamelabel.grid(row=8,column=1,pady=5)

passwordlabel=Label(inlframel, text='Password',font=('verdana',15),bg='powder blue')

passwordlabel.grid(row=9,column=1,pady=5)

#=============================ENTRYBOXES========================

global identry

global fnameentry

global lnameentry

global ageentry

global addressentry

global usernameentry

global passwordentry

identry = Entry(inlframel, width=20,bg='azure2')

identry.grid(row=2,column=2)

fnameentry = Entry(inlframel, width=20,bg='azure2')

fnameentry .grid(row=4,column=2,pady=10,padx=10)

lnameentry = Entry(inlframel, width=20,bg='azure2')

lnameentry.grid(row=5,column=2,pady=10,padx=10)

ageentry = Entry(inlframel, width=20,bg='azure2')

ageentry.grid(row=6,column=2,pady=10,padx=10)

addressentry = Entry(inlframel, width=20,bg='azure2')

addressentry.grid(row=7,column=2,pady=10,padx=10)

usernameentry = Entry(inlframel, width=20,bg='azure2')

usernameentry.grid(row=8,column=2,pady=5,padx=10)

passwordentry = Entry(inlframel, width=20,bg='azure2')

passwordentry.grid(row=9,column=2,pady=10,padx=10)

#===========================BUTTONS=============================

searchid=Button(inlframel, fg="black", font=('arial', 12, 'bold'), width=10,

text="Search",command=update1).grid(row=3, column=2,pady=(10,70))

save=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Save",command=update2).grid(row=10, column=2,pady=(10,10))

clear=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Clear",command=clearvp).grid(row=10, column=1,pady=(10,10))

back=Button(inlframel, fg="black",padx=15, font=('arial', 12, 'bold'), width=13,

text="Back to Main Menu",command=back3).grid(row=11, column=1,pady=(10,10))

def updaterecord2():

global update22

main.destroy()

update22=Tk()

update22.title('Update Records')

update22.geometry('350x430')

update22.configure(background='powder blue')

#update.resizable(0,0)

#=================UPDATE RECORD FUNCTION=========================

def update1():

conn = sqlite3.connect("admin.db")

c = conn.cursor()

recordid=identry.get()

c.execute("SELECT \* FROM member WHERE mem\_id = ? ", (identry.get(),))

records = c.fetchall()

for record in records:

usernameentry.insert(0,record[5])

passwordentry.insert(0,record[6])

conn.commit()

def clearvp():

usernameentry.delete(0,END)

passwordentry.delete(0,END)

def update2():

conn = sqlite3.connect("admin.db")

c = conn.cursor()

recordid=identry.get()

c.execute("""UPDATE member SET

username=:username2,

password=:password2

WHERE mem\_id= :mem\_id""",

{

'username2':usernameentry.get(),

'password2':passwordentry.get(),

'mem\_id':recordid

})

conn.commit()

conn.close()

ms.showinfo(" ","Record Updated")

update22.destroy()

main2()

#=============================FRAMES=============================

Tops=Frame(update22, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

lframe=Frame(update22, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=3, bg='powder blue')

inlframel.pack(side=TOP)

#==============================LABELS=============================

addlabel=Label(Tops, text='UPDATE RECORD',font=('system',20),bg='powder blue')

addlabel.grid(row=1,column=1)

idlabel=Label(inlframel, text='SEARCH ID',font=('verdana',15),bg='powder blue')

idlabel.grid(row=2,column=1)

usernamelabel=Label(inlframel, text='Username',font=('verdana',15),bg='powder blue')

usernamelabel.grid(row=8,column=1,pady=5)

passwordlabel=Label(inlframel, text='Password',font=('verdana',15),bg='powder blue')

passwordlabel.grid(row=9,column=1,pady=5)

#=============================ENTRYBOXES========================

global identry

global usernameentry

global passwordentry

identry = Entry(inlframel, width=20,bg='azure2')

identry.grid(row=2,column=2)

usernameentry = Entry(inlframel, width=20,bg='azure2')

usernameentry.grid(row=8,column=2,pady=5,padx=10)

passwordentry = Entry(inlframel, width=20,bg='azure2')

passwordentry.grid(row=9,column=2,pady=10,padx=10)

#===========================BUTTONS=============================

searchid=Button(inlframel, fg="black", font=('arial', 12, 'bold'), width=10,

text="Search",command=update1).grid(row=3, column=2,pady=(10,70))

save=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Save",command=update2).grid(row=10, column=2,pady=(10,10))

clear=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Clear",command=clearvp).grid(row=10, column=1,pady=(10,10))

back=Button(inlframel, fg="black",padx=15, font=('arial', 12, 'bold'), width=13,

text="Back to Main Menu",command=back6).grid(row=11, column=1,pady=(10,10))

def updaterecord3():

global update33

main.destroy()

update33=Tk()

update33.title('Update Records')

update33.geometry('350x430')

update33.configure(background='powder blue')

#update.resizable(0,0)

#=================UPDATE RECORD FUNCTION=========================

def update1():

conn = sqlite3.connect("admin.db")

c = conn.cursor()

recordid=identry.get()

c.execute("SELECT \* FROM member2 WHERE mem\_id = ? ", (identry.get(),))

records = c.fetchall()

for record in records:

usernameentry.insert(0,record[5])

passwordentry.insert(0,record[6])

conn.commit()

def clearvp():

usernameentry.delete(0,END)

passwordentry.delete(0,END)

def update2():

conn = sqlite3.connect("admin.db")

c = conn.cursor()

recordid=identry.get()

c.execute("""UPDATE member2 SET

username=:username2,

password=:password2

WHERE mem\_id= :mem\_id""",

{

'username2':usernameentry.get(),

'password2':passwordentry.get(),

'mem\_id':recordid

})

conn.commit()

conn.close()

ms.showinfo(" ","Record Updated")

update33.destroy()

main2()

#=============================FRAMES=============================

Tops=Frame(update33, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

lframe=Frame(update33, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=3, bg='powder blue')

inlframel.pack(side=TOP)

#==============================LABELS=============================

addlabel=Label(Tops, text='UPDATE RECORD',font=('system',20),bg='powder blue')

addlabel.grid(row=1,column=1)

idlabel=Label(inlframel, text='SEARCH ID',font=('verdana',15),bg='powder blue')

idlabel.grid(row=2,column=1)

usernamelabel=Label(inlframel, text='Username',font=('verdana',15),bg='powder blue')

usernamelabel.grid(row=8,column=1,pady=5)

passwordlabel=Label(inlframel, text='Password',font=('verdana',15),bg='powder blue')

passwordlabel.grid(row=9,column=1,pady=5)

#=============================ENTRYBOXES========================

global identry

global usernameentry

global passwordentry

identry = Entry(inlframel, width=20,bg='azure2')

identry.grid(row=2,column=2)

usernameentry = Entry(inlframel, width=20,bg='azure2')

usernameentry.grid(row=8,column=2,pady=5,padx=10)

passwordentry = Entry(inlframel, width=20,bg='azure2')

passwordentry.grid(row=9,column=2,pady=10,padx=10)

#===========================BUTTONS=============================

searchid=Button(inlframel, fg="black", font=('arial', 12, 'bold'), width=10,

text="Search",command=update1).grid(row=3, column=2,pady=(10,70))

save=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Save",command=update2).grid(row=10, column=2,pady=(10,10))

clear=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Clear",command=clearvp).grid(row=10, column=1,pady=(10,10))

back=Button(inlframel, fg="black",padx=15, font=('arial', 12, 'bold'), width=13,

text="Back to Main Menu",command=back7).grid(row=11, column=1,pady=(10,10))

#===========================DELETE RECORD=============================

def delrec():

global del1

main.destroy()

del1=Tk()

del1.title('Delete Records')

del1.geometry('350x700')

del1.configure(background='powder blue')

#update.resizable(0,0)

#============================DELETE FUNCTIONS===================

def dels():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

recordid=identry.get()

c.execute("SELECT \* FROM member2 WHERE mem\_id = ? ", (identry.get(),))

records = c.fetchall()

for record in records:

fnameentry.insert(0,record[1])

lnameentry.insert(0,record[2])

ageentry.insert(0,record[3])

addressentry.insert(0,record[4])

usernameentry.insert(0,record[5])

passwordentry.insert(0,record[6])

conn.commit()

def del2():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

c.execute("DELETE from member2 WHERE mem\_id = ? ", (identry.get(),))

identry.delete(0,END)

fnameentry.delete(0,END)

lnameentry.delete(0,END)

ageentry.delete(0,END)

addressentry.delete(0,END)

usernameentry.delete(0,END)

passwordentry.delete(0,END)

ms.showinfo(" ","Data Deleted")

del1.destroy()

main2()

conn.commit()

#=============================FRAMES=============================

Tops=Frame(del1, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(del1, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

lframe=Frame(del1, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=3, bg='powder blue')

inlframel.pack(side=TOP)

#==============================LABELS=============================

addlabel=Label(Tops, text='DELETE RECORD',font=('system',20),bg='powder blue')

addlabel.grid(row=1,column=1)

idlabel=Label(inlframel, text='SEARCH ID',font=('verdana',15),bg='powder blue')

idlabel.grid(row=2,column=1)

firstnamelabel=Label(inlframel, text='First Name',font=('verdana',15),bg='powder blue')

firstnamelabel.grid(row=4,column=1,pady=5)

lastnamelabel=Label(inlframel, text='Last Name',font=('verdana',15),bg='powder blue')

lastnamelabel.grid(row=5,column=1,pady=5)

agelabel=Label(inlframel, text='Age',font=('verdana',15),bg='powder blue')

agelabel.grid(row=6,column=1,pady=5)

addresslabel=Label(inlframel, text='Address',font=('verdana',15),bg='powder blue')

addresslabel.grid(row=7,column=1,pady=5)

usernamelabel=Label(inlframel, text='Username',font=('verdana',15),bg='powder blue')

usernamelabel.grid(row=8,column=1,pady=5)

passwordlabel=Label(inlframel, text='Password',font=('verdana',15),bg='powder blue')

passwordlabel.grid(row=9,column=1,pady=5)

#=============================ENTRYBOXES========================

global identry

global fnameentry

global lnameentry

global ageentry

global addressentry

global usernameentry

global passwordentry

identry = Entry(inlframel, width=20,bg='azure2')

identry.grid(row=2,column=2)

fnameentry = Entry(inlframel, width=20,bg='azure2')

fnameentry .grid(row=4,column=2,pady=10,padx=10)

lnameentry = Entry(inlframel, width=20,bg='azure2')

lnameentry.grid(row=5,column=2,pady=10,padx=10)

ageentry = Entry(inlframel, width=20,bg='azure2')

ageentry.grid(row=6,column=2,pady=10,padx=10)

addressentry = Entry(inlframel, width=20,bg='azure2')

addressentry.grid(row=7,column=2,pady=10,padx=10)

usernameentry = Entry(inlframel, width=20,bg='azure2')

usernameentry.grid(row=8,column=2,pady=5,padx=10)

passwordentry = Entry(inlframel, width=20,bg='azure2')

passwordentry.grid(row=9,column=2,pady=10,padx=10)

#===========================BUTTONS=============================

searchid=Button(inlframel, fg="black", font=('arial', 12, 'bold'), width=10,

text="Search",command=dels).grid(row=3, column=2,pady=(10,70))

save=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=10,

text="Delete",command=del2).grid(row=10, column=2,pady=(10,10))

back=Button(inlframel, fg="black",padx=5, font=('arial', 12, 'bold'), width=13,

text="Back to Main Menu",command=back4).grid(row=11, column=2,pady=(10,10))

#Variables-------------------------------------------------------------

username = StringVar()

password = StringVar()

fname= StringVar()

lname= StringVar()

age= StringVar()

address= StringVar()

#Frames----------------

Tops=Frame(chooselogin, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

bot=Frame(chooselogin, width=1350, height=50, bd=16,bg='powder blue')

bot.pack(side=BOTTOM)

#====================login function==================

def loginall():

select=drop.get()

if select=="Admin":

return Login()

elif select=="Accountant":

return Login2()

elif select=="Employee":

return Login3()

#button==========================================================

choice3=Button(bot, fg="black", font=('arial', 12, 'bold'), width=9,

text="Login",command =loginall).grid(row=4, column=2,pady=10,padx=30)

drop = ttk.Combobox(bot,value=["Admin","Accountant", "Employee" ],width=12)

drop.set("Login As")

drop.grid(row=3,column=2,padx=(10,10))

#labels==========================================================

adminlabel=Label(Tops, text='Frozen Bee Login',font=('system',30),bg='powder blue')

adminlabel.grid(row=2,column=1,pady=(30,10),padx=50)

usernamelabel=Label(bot,text="Username: ",bg='powder blue')

usernamelabel.grid(row=1,column=1,padx=0)

password1=Label(bot,text="Password: ",bg='powder blue')

password1.grid(row=2,column=1,padx=(0,10))

#Entry

USERNAME=Entry(bot,text=username,width=15)

USERNAME.grid(row=1,column=2)

PASSWORD=Entry(bot,width=15,text=password,show ='\*')

PASSWORD.grid(row=2,column=2)

lbl\_text = Label(bot, bg='powder blue')

lbl\_text.grid(row=6, columnspan=3,pady=(10,210))

#=======================QUIT FUNCTIONS==================================

def quit1():

main.destroy()

def quit2():

addemp1.destroy()

chooselogin.mainloop()

#============================PAYROLL PART NA ITO=================================================

#============================PAYROLL PART NA ITO=================================================

#============================PAYROLL PART NA ITO=================================================

#============================PAYROLL PART NA ITO=================================================

#============================PAYROLL PART NA ITO=================================================

#============================PAYROLL PART NA ITO=================================================

def pr():

root=Tk()

root.title('Payroll')

root.geometry('1250x900')

root.configure(background='powder blue')

root.resizable(0,0)

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

#creating tables

c.execute("""CREATE TABLE IF NOT EXISTS payroll(

payrollperiod text,

dealership text,

fname text,

lname text,

rate integer,

daysofwork integer,

totalwage integer,

othour integer,

otamount integer,

totalamount integer,

ut integer,

vale integer,

loan integer,

ph integer,

sss integer,

netamount integer,

netpay integer,

loanbalance integer,

amountloan integer)""")

#functions

#submitbtn function

def submit():

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

#insert into tables

c.execute("INSERT INTO payroll VALUES(:payrollperiod,:dealership,:fname, :lname, :rate,:daysofwork, :totalwage, :othour, :otamount, :totalamount,:ut, :vale, :loan, :ph, :sss, :netamount, :netpay, :loanbalance, :amountloan)",

{

'payrollperiod':payrollperiod.get().upper().replace(" ",""),

'dealership':drop.get().upper(),

'fname':fname.get().upper(),

'lname':lname.get().upper(),

'rate':rate.get(),

'daysofwork':daysofwork.get(),

'totalwage':totalwage.get(),

'othour':othour.get(),

'otamount':otamount.get(),

'totalamount':totalamount.get(),

'ut':ut.get(),

'vale':vale.get(),

'loan':loan.get(),

'ph':ph.get(),

'sss':sss.get(),

'netamount':netamount.get(),

'netpay':netpay.get(),

'loanbalance':loanbalance.get(),

'amountloan':amountloan.get()})

#commit changes

conn.commit()

#close connection

conn.close()

#clear text boxes

payrollperiod.delete(0,END)

drop.current(0)

fname.delete(0,END)

lname.delete(0,END)

rate.delete(0,END)

daysofwork.delete(0,END)

totalwage.delete(0,END)

othour.delete(0,END)

otamount.delete(0,END)

totalamount.delete(0,END)

ut.delete(0,END)

vale.delete(0,END)

loan.delete(0,END)

ph.delete(0,END)

sss.delete(0,END)

netamount.delete(0,END)

netpay.delete(0,END)

loanbalance.delete(0,END)

amountloan.delete(0,END)

#calculate

def compute():

rate1=rate.get()

daysofwork1=daysofwork.get()

w= totalwage.insert(0,int(rate1)\*int(daysofwork1))

#totalwage.config(state="disabled")

othour1=othour.get()

otamount1=otamount.get()

twage=totalwage.get()

totalamount.insert(0,int(othour1)\*int(otamount1)+int(twage))

#totalamount.config(state="disabled")

vale1=vale.get()

loan1=loan.get()

ut1=ut.get()

ph1=ph.get()

sss1=sss.get()

netamount.insert(0,int(vale1)+int(loan1)+int(ph1)+int(sss1)+int(ut1))

#netamount.config(state="disabled")

tamount=totalamount.get()

netamt=netamount.get()

netpay.insert(0,int(tamount)-int(netamt))

#netpay.config(state="disabled")

amount=amountloan.get()

loan1=loan.get()

loanbalance.insert(0,int(amount)-int(loan1))

def viewpay():

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

vpayrollperiod =payrollperiod.get()

vdealership=drop.get()

vfname=fname.get()

vlname=lname.get()

vrate=rate.get()

vdaysofwork=daysofwork.get()

vtotalwage=totalwage.get()

vothour=othour.get()

votamount=otamount.get()

vtotalamount=totalamount.get()

vut=ut.get()

vvale=vale.get()

vloan=loan.get()

vph=ph.get()

vsss=sss.get()

vnetamount=netamount.get()

vnetpay=netpay.get()

vloanbalance=loanbalance.get()

global vp

vp=viewpayment.insert(END,'FROZEN BEE '+'\n'+'Payroll Period: '+vpayrollperiod+'\n'+'Dealership: '+vdealership+'\n'+'Name: '+vfname+" "+

vlname+'\n\n'+

'Rate: '+vrate+'\n'+

'Days Of Work: '+vdaysofwork+'\n'+

'Basic Wage: '+vtotalwage+'\n\n'+

'Overtime Hour: '+vothour+'\n'+

'Overtime Amount: '+votamount+'\n'+

'Total Gross: '+vtotalamount+'\n\n'+

'Undertime/Late: '+vut+'\n'+

'Cash Advance: '+vvale+'\n'+

'Loan: '+vloan+'\n'+

'PhilHealth: '+vph+'\n'+

'SSS: '+ vsss+'\n\n'+

'Deductions: '+vnetamount+'\n'+

'Net Pay: '+vnetpay+'\n'+

'Loan Balance: '+vloanbalance+'\n\n')

#commit changes

conn.commit()

#close connection

conn.close()

def printslip():

q=viewpayment.get("1.0","end-1c")

filename=tempfile.mktemp(".txt")

open(filename,"w").write(q)

os.startfile(filename,"print")

def clearvp():

viewpayment.delete("1.0",END)

payrollperiod.delete(0,END)

drop.current(0)

fname.delete(0,END)

lname.delete(0,END)

rate.delete(0,END)

daysofwork.delete(0,END)

totalwage.delete(0,END)

othour.delete(0,END)

otamount.delete(0,END)

totalamount.delete(0,END)

ut.delete(0,END)

vale.delete(0,END)

loan.delete(0,END)

ph.delete(0,END)

sss.delete(0,END)

netamount.delete(0,END)

netpay.delete(0,END)

loanbalance.delete(0,END)

amountloan.delete(0,END)

def clearvp2():

viewpayment.delete("1.0",END)

def clearvp3():

totalwage.delete(0,END)

totalamount.delete(0,END)

netamount.delete(0,END)

netpay.delete(0,END)

loanbalance.delete(0,END)

#new window

def reports():

rep=Tk()

rep.title('Payroll')

rep.geometry('400x550')

rep.configure(background='powder blue')

rep.resizable(0,0)

#functions

def query2():

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

#query database

c.execute("SELECT \* FROM payroll")

records = c.fetchall()

print(records)

for index, x in enumerate(records):

num=0

for y in x:

querylabel=Label(inlframer, text=y,bg='powderblue')

querylabel.grid(row=index, column=num,padx=10,pady=10)

num+=1

def searchemp():

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

searched= searchbox.get()

#c.execute("SELECT \*,oid from payroll WHERE oid= "+searchbox.get())

#records = c.execute("SELECT \* FROM payroll WHERE lname = (?)", searchbox.get())

c.execute("SELECT \* from payroll where lname = ?", (searchbox.get(),))

records = c.fetchall()

print(records)

'''

for index, x in enumerate(records):

num=0

for y in x:

querylabel=Label(inlframel, text=y,bg='powderblue')

querylabel.grid(row=2, column=num,padx=10,pady=10)

num+=1

'''

printrecords=''

for index, x in enumerate(records):

index=3

num=1

for y in x:

#querylabel=Label(inlframer, text=y,bg='powderblue')

#querylabel.grid(row=index, column=num,padx=(10,1000),pady=4)

viewemp.insert(END,str(y)+'\n')

index+=1

searchbox.delete(0,END)

def clearvp():

viewemp.delete("1.0",END)

viewemp=Text(rep,width=25,height=20,bg='azure2')

viewemp.grid(row=4,column=1,pady=10)

#labels

emplabel=Label(rep, text='Employee Reports',font=('system',20),bg='powder blue')

emplabel.grid(row=1,column=1)

payrollperiodlabel=Label(rep, text='Payroll Period\n Dealership\n First Name\n Last Name\n Rate\n Days Of Work\n Basic Wage\n OT Hour\nOT Amount\n Total Gross\n UT/Late \n Vale\n Loan\n PhilHealth\n SSS\n Deductions\n Net Pay\n Loan Balance\n Amount Loan\n',font=('verdana',10),bg='powder blue')

payrollperiodlabel.grid(row=4,column=0)

#entry

searchbox = Entry(rep,width=10,bg='azure2')

searchbox.grid(row=2,column=1,pady=10)

#buttons

#Query Button

#querybtn=Button(inlframel, text='Show All Records', command=query2, padx=13)

#querybtn.grid(row=1, column=1)

#searchbtn

searchbtn=Button(rep, fg="black", font=('arial', 12, 'bold'), width=20,

text="Search",command=searchemp).grid(row=3, column=1,pady=10)

#CLEAR

clearbtn=Button(rep, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Clear",command=clearvp).grid(row=5, column=1)

#commit changes

#close connection

#end of window2------------------------------------------------

#-----------------------------------------------------------

#new window

def empmenu():

emp=Tk()

emp.title('Employee Reports')

emp.geometry('400x500')

emp.configure(background='powder blue')

emp.resizable(0,0)

#Frames----------------

Tops=Frame(emp, width=1350, height=50, bd=16, relief="raise",bg='powder blue')

Tops.pack(side=TOP)

#label

searchlabel = Label(Tops, text='Employee Records',font=('system',20),bg='powder blue')

searchlabel.grid(row=0,column=1,pady=10)

#search button

searchbtn=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Search Employee ",command=reports).grid(row=2, column=1,pady=10,padx=(100,100))

#show all records button

showrecordbtn=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Show All Records",command=showall).grid(row=3, column=1,pady=10,padx=(100,100))

#update button

updatebtn=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Update Record").grid(row=4, column=1,pady=10,padx=(100,100))

#delete button

deleterbtn=Button(Tops, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Delete Record").grid(row=5, column=1,pady=(10,350),padx=(100,100))

#window for show all records:

def showall():

sa=Tk()

sa.title('Employee Records')

sa.geometry('1300x1400')

sa.configure(background='powder blue')

sa.resizable(0,0)

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

#label

searchlabel = Label(sa, text='Employee Records',font=('system',20),bg='powder blue')

searchlabel.grid(row=0,column=1,pady=10)

#label

vlabel=Text(sa,width=160,height=1,bg='azure2')

vlabel.grid(row=2,column=1,padx=10)

vlabel.insert(END,"Payroll Period\t\t\tDealership\t\tName\t\t\tBasic Wage\t\tTotal Gross\t\tDeductions\t\tNet Pay\t\tLoan Balance")

vlabel.config(state="disabled")

viewall=Text(sa,width=160,height=250,bg='azure2')

viewall.grid(row=3,column=1,padx=10)

#query database

c.execute("SELECT \*,oid FROM payroll")

records = c.fetchall()

print(records)

printrecords=''

for record in records:

printrecords= str(record[0])+'\t\t\t'+str(record[1])+'\t\t'+str(record[2])+'\t'+str(record[3])+'\t\t'+str(record[6])+'\t\t'+str(record[9])+'\t\t'+str(record[15])+'\t\t'+str(record[16])+'\t\t'+str(record[17])+'\t\t'+'\n'

#querylabel=Label(Tops, text=printrecords,bg='powderblue')

#querylabel.grid(row=3, column=1,padx=10,pady=10)

viewall.insert(END,str(printrecords)+'\t'+'\n')

def paysum1():

global ps1

root.destroy()

ps1=Tk()

ps1.title('Payroll Summary')

ps1.geometry('1210x900')

ps1.configure(background='powder blue')

#ps1.resizable(0,0)

#database connection

conn=sqlite3.connect('payroll.db')

#cursor

c = conn.cursor()

#Frames----------------

Tops=Frame(ps1, width=1350, height=50, bd=1 ,bg='powder blue')

Tops.pack(side=TOP)

lframe=Frame(ps1, width=500, height=500 ,bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500,bd=1, height=500, bg='powder blue')

inlframel.pack(side=TOP,padx=20)

inlframel2=Frame(lframe, width=500, height=500, bd=16, bg='powder blue')

inlframel2.pack(side=BOTTOM)

#label

searchlabel = Label(Tops, text='Enter Payroll Period: ',font=('system',10),bg='powder blue')

searchlabel.grid(row=0,column=1,pady=10)

deallabel = Label(Tops, text='Select Dealership',font=('system',10),bg='powder blue')

deallabel.grid(row=1,column=1,pady=10)

#entry

searchbox=Entry(Tops,width=23)

searchbox.grid(row=0, column=2,padx=(25,30))

viewall=Text(inlframel,width=140,height=30,bg='white')

viewall.grid(row=2,column=1)

viewall1=Text(inlframel,width=140,height=2,bg='white')

viewall1.grid(row=1,column=1)

viewall1.insert(END,"Payroll Period\t\tDealership\t\tFirst Name\t\tLast Name\t\tBasic Wage\t\tTotal Gross\t\tDeductions\t\tNet Pay\t\tLoan Balance")

#viewall.insert(END,"Payroll Period\t\tDealership\t\tFirst Name\t\tLast Name\t\tBasic Wage\t\tTotal Gross\t\tDeductions\t\tNet Pay\t\tLoan Balance")

#viewall.configure(state="disabled")

drop = ttk.Combobox(Tops,value=["NISSAN LIPA","NISSAN DASMA", "NISSAN SAN PABLO","NISSAN ALABANG","NISSAN CARMONA" ])

drop.set("SELECT DEALERSHIP")

drop.grid(row=1,column=2)

def search():

c.execute("SELECT payrollperiod,dealership,fname,lname,totalwage,totalamount,netamount,netpay,loanbalance from payroll where dealership = ? and payrollperiod=?", (drop.get(),searchbox.get().upper().replace(" ",""),))

records = c.fetchall()

print(records)

records1=c.execute("SELECT sum(totalwage) from payroll where dealership = ? and payrollperiod=?", (drop.get(),searchbox.get().upper().replace(" ",""),))

records1=c.fetchone()

records2=c.execute("SELECT SUM(totalamount) from payroll where dealership = ? and payrollperiod=?", (drop.get(),searchbox.get().upper().replace(" ",""),))

records2=c.fetchone()

records3=c.execute("SELECT SUM(netamount) from payroll where dealership = ? and payrollperiod=?", (drop.get(),searchbox.get().upper().replace(" ",""),))

records3=c.fetchone()

records4=c.execute("SELECT SUM(netpay) from payroll where dealership = ? and payrollperiod=?", (drop.get(),searchbox.get().upper().replace(" ",""),))

records4=c.fetchone()

searched=searchbox.get()

if searched=="":

#viewall.insert(END,"PLEASE INPUT PAYROLL PERIOD")

messagebox.showwarning("ERROR","Please input payroll period")

viewall.configure(state="disabled")

else:

viewall.configure(state="normal")

viewall.delete("1.0",END)

for index, x in enumerate(records):

index=3

num=1

for y in x:

viewall.insert(END,str(y)+'\t\t')

#index+=1

viewall.insert(END,"\n\n\n"+"TOTAL BASIC WAGE: "+str(records1)+'\n')

viewall.insert(END,"TOTAL GROSS: "+str(records2)+'\n')

viewall.insert(END,"TOTAL DEDUCTIONS: "+str(records3)+'\n')

viewall.insert(END,"TOTAL NETPAY: "+str(records4)+'\n')

#print

def clearvp():

viewall.delete("1.0",END)

#search button

searchbtn=Button(Tops, fg="black", font=('arial', 12, 'bold'), width=14,

text="Search",command=search).grid(row=3, column=2,pady=10)

#CLEAR

clearbtn=Button(inlframel2, fg="black", font=('arial', 12, 'bold'), width=14,

text="Clear",command=clearvp).grid(row=1, column=1,pady=10)

#print

mainbtn=Button(inlframel2, fg="black", font=('arial', 12, 'bold'), width=14,

text="Back to Payroll",command=backk1).grid(row=1, column=3,pady=10)

def backk1():

ps1.destroy()

pr()

def backk2():

showall2.destroy()

pr()

def showall22():

global showall2

root.destroy()

showall2=Tk()

showall2.title("Payroll Records")

showall2.geometry("1600x550")

showall2.configure(background='powder blue')

#showall2.resizable(0,0)

def View():

conn = sqlite3.connect("payroll.db")

c = conn.cursor()

rows=c.execute("SELECT \* FROM payroll")

rows = c.fetchall()

for row in rows:

tree.insert("",END, values = row)

frame1 = Frame(showall2, width=850, height=350,bd=15,bg='powder blue')

frame1.pack(side=TOP)

frame2 = Frame(showall2, width=850, height=350,bd=15,relief="raise",bg='powder blue')

frame2.pack(side=TOP)

#print

mainbtn=Button(frame2, fg="black", font=('arial', 12, 'bold'), width=14,

text="Main Menu").grid(row=1, column=1)

btopr=Button(frame2, fg="black", font=('arial', 12, 'bold'), width=14,

text="Back to Payroll",command=backk2).grid(row=1, column=2)

#logout=Button(frame2, fg="black", font=('arial', 12, 'bold'), width=14,

#text="Logout").grid(row=1, column=3)

#label

searchlabel = Label(frame1, text='Payroll Records',font=('system',20),bg='powder blue')

searchlabel.pack(pady=(10,30))

s = ttk.Style()

s.configure('MyStyle.Treeview', rowheight=500)

tree= ttk.Treeview(frame1, column=("column1", "column2", "column3", "column4", "column5", "column6", "column7", "column8", "column9", "column10"

, "column11", "column12", "column13", "column14", "column15", "column16", "column17", "column18", "column19"), show='headings',style='MyStyle.Treeview')

tree.heading("#1", text="Payroll Period")

tree.column("#1", width = 100, anchor = "w")

tree.heading("#2", text="Dealership")

tree.column("#2", width = 120, anchor = "w")

tree.heading("#3", text="First Name")

tree.column("#3", width = 90, anchor = "w")

tree.heading("#4", text="Last Name")

tree.column("#4", width = 80, anchor = "w")

tree.heading("#5", text="Rate")

tree.column("#5", width = 80, anchor = "w")

tree.heading("#6", text="Days of Work")

tree.column("#6", width = 80, anchor = "w")

tree.heading("#7", text="Basic Wage")

tree.column("#7", width = 80, anchor = "w")

tree.heading("#8", text="OT Hour")

tree.column("#8", width = 80, anchor = "w")

tree.heading("#9", text="OT Amount")

tree.column("#9", width = 80, anchor = "w")

tree.heading("#10", text="Total Gross")

tree.column("#10", width = 80, anchor = "w")

tree.heading("#11", text="UT/Late")

tree.column("#11", width = 80, anchor = "w")

tree.heading("#12", text="Cash Advance")

tree.column("#12", width = 90, anchor = "w")

tree.heading("#13", text="Loan")

tree.column("#13", width = 60, anchor = "w")

tree.heading("#14", text="PhilHealth")

tree.column("#14", width = 80, anchor = "w")

tree.heading("#15", text="SSS")

tree.column("#15", width = 60, anchor = "w")

tree.heading("#16", text="Deductions")

tree.column("#16", width = 80, anchor = "w")

tree.heading("#17", text="NetPay")

tree.column("#17", width = 80, anchor = "w")

tree.heading("#18", text="Loan Balance")

tree.column("#18", width = 80, anchor = "w")

tree.heading("#19", text="Amount Loan")

tree.column("#19", width = 80, anchor = "w")

tree.pack()

View()

#Frames----------------

Tops=Frame(root, width=1350, height=50, bd=16, bg='powder blue')

Tops.pack(side=TOP)

lframe=Frame(root, width=500, height=500, bd=16, bg='powder blue')

lframe.pack(side=LEFT)

inlframel=Frame(lframe, width=500, height=500, bd=16, relief="raise",bg='powder blue')

inlframel.pack(side=LEFT)

inlframer=Frame(lframe, width=500, height=500, bd=16, relief="raise",bg='powder blue')

inlframer.pack(side=LEFT)

inlframer2=Frame(lframe, width=100, height=500, bd=16, relief="raise",bg='powder blue')

inlframer2.pack(side=LEFT)

#Labels----------------------

payrolllabel=Label(Tops, text='Frozen Bee Payroll Management System',font=('system',30),bg='powder blue')

payrolllabel.grid(row=0,column=1,pady=(30,10))

payrollperiodlabel=Label(inlframel, text='Payroll Period',font=('verdana',15),bg='powder blue')

payrollperiodlabel.grid(row=1,column=1,pady=5)

dealershiplabel=Label(inlframel, text='Dealership',font=('verdana',15),bg='powder blue')

dealershiplabel.grid(row=2,column=1,pady=5)

fnamelabel=Label(inlframel, text='First Name',font=('verdana',15),bg='powder blue')

fnamelabel.grid(row=3,column=1,pady=5)

lnamelabel=Label(inlframel, text='Last Name',font=('verdana',15),bg='powder blue')

lnamelabel.grid(row=4,column=1,pady=5,padx=10)

ratelabel=Label(inlframel, text='Rate',font=('verdana',15),bg='powder blue')

ratelabel.grid(row=5,column=1,pady=5,padx=10)

daysofworklabel=Label(inlframel, text='Days of Work',font=('verdana',15),bg='powder blue')

daysofworklabel.grid(row=6,column=1,pady=5,padx=10)

othourlabel=Label(inlframel, text='OT Hour',font=('verdana',15),bg='powder blue')

othourlabel.grid(row=7,column=1,pady=5,padx=10)

otamountlabel=Label(inlframel, text='OT Amount',font=('verdana',15),bg='powder blue')

otamountlabel.grid(row=8,column=1,pady=15,padx=10)

deductionslabel=Label(inlframel, text='DEDUCTIONS',font=('verdana',15),bg='powder blue')

deductionslabel.grid(row=9,column=1,pady=15,padx=10)

utlabel=Label(inlframel, text='Undertime/Late',font=('verdana',15),bg='powder blue')

utlabel.grid(row=10,column=1,pady=5,padx=10)

valelabel=Label(inlframel, text='Cash Advance',font=('verdana',15),bg='powder blue')

valelabel.grid(row=11,column=1,pady=5,padx=10)

loanlabel=Label(inlframel, text='Loan',font=('verdana',15),bg='powder blue')

loanlabel.grid(row=12,column=1,pady=5,padx=10)

phlabel=Label(inlframel, text='PhilHealth',font=('verdana',15),bg='powder blue')

phlabel.grid(row=13,column=1,pady=5,padx=10)

ssslabel=Label(inlframel, text='SSS',font=('verdana',15),bg='powder blue')

ssslabel.grid(row=14,column=1,pady=5,padx=10)

loanbalancelabel=Label(inlframer, text='Loan Balance',font=('verdana',15),bg='powder blue')

loanbalancelabel.grid(row=1,column=1,pady=(50,0),padx=10)

amountloanlabel=Label(inlframel, text='Amount Loan',font=('verdana',15),bg='powder blue')

amountloanlabel.grid(row=15,column=1,pady=5,padx=10)

totalwagelabel=Label(inlframer, text='Basic Wage',font=('verdana',15),bg='powder blue')

totalwagelabel.grid(row=2,column=1,padx=10)

totalamountlabel=Label(inlframer, text='Total Gross',font=('verdana',15),bg='powder blue')

totalamountlabel.grid(row=3,column=1,padx=10)

netamountlabel=Label(inlframer, text='Deductions',font=('verdana',15),bg='powder blue')

netamountlabel.grid(row=4,column=1,padx=10)

netpaylabel=Label(inlframer, text='NetPay',font=('verdana',15),bg='powder blue')

netpaylabel.grid(row=5,column=1,padx=10)

#Entry

payrollperiod = Entry(inlframel, width=20,bg='azure2')

payrollperiod.grid(row=1,column=2,pady=10,padx=10)

#dealership = Entry(inlframel, width=20,bg='azure2')

#dealership.grid(row=2,column=2,pady=10,padx=10)

fname = Entry(inlframel, width=20,bg='azure2')

fname.grid(row=3,column=2,pady=10,padx=10)

lname = Entry(inlframel, width=20,bg='azure2')

lname.grid(row=4,column=2,pady=10,padx=10)

rate = Entry(inlframel, width=20,bg='azure2')

rate.grid(row=5,column=2,pady=10,padx=10)

daysofwork = Entry(inlframel, width=20,bg='azure2')

daysofwork.grid(row=6,column=2,pady=10,padx=10)

totalwage = Entry(inlframer, width=20,bg='azure2')

totalwage.grid(row=2,column=2,pady=5,padx=(5,50))

othour = Entry(inlframel, width=20,bg='azure2')

othour.grid(row=7,column=2,pady=10,padx=10)

otamount = Entry(inlframel, width=20,bg='azure2')

otamount.grid(row=8,column=2,pady=10,padx=10)

totalamount = Entry(inlframer, width=20,bg='azure2')

totalamount.grid(row=3,column=2,pady=5,padx=(5,50))

ut = Entry(inlframel, width=20,bg='azure2')

ut.grid(row=10,column=2,pady=10,padx=10)

vale = Entry(inlframel, width=20,bg='azure2')

vale.grid(row=11,column=2,pady=10,padx=10)

loan = Entry(inlframel, width=20,bg='azure2')

loan.grid(row=12,column=2,pady=10,padx=10)

ph =Entry(inlframel, width=20,bg='azure2')

ph.grid(row=13,column=2,pady=10,padx=10)

sss = Entry(inlframel, width=20,bg='azure2')

sss.grid(row=14,column=2,pady=10,padx=10)

netamount = Entry(inlframer, width=20,bg='azure2')

netamount.grid(row=4,column=2,pady=5,padx=(5,50))

netpay = Entry(inlframer, width=20,bg='azure2')

netpay.grid(row=5,column=2,pady=5,padx=(5,50))

loanbalance = Entry(inlframer, width=20,bg='azure2')

loanbalance.grid(row=1,column=2,pady=(50,0),padx=(5,50))

amountloan = Entry(inlframel, width=20,bg='azure2')

amountloan.grid(row=15,column=2,pady=(5,10),padx=10)

viewpayment=Text(inlframer2,width=40,height=22,bg='azure2')

viewpayment.grid(row=0,column=1)

#buttons

#compute

computebtn=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Calculate",command=compute).grid(row=6, column=1,pady=(50,10),padx=(50,15))

#submit button

submitbtn=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Save",command=submit).grid(row=6, column=2,pady=(50,10),padx=(40,15))

#viewreports button

viewreports=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Payroll Summary",command=paysum1).grid(row=7, column=1,pady=(10,10),padx=(50,15))

#show all

allreports=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Payroll Records",command=showall22).grid(row=7, column=2,pady=(10,10),padx=(40,15))

#back to main menu button

back=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Clear Calculations",command=clearvp3).grid(row=8, column=2,pady=(10,210),padx=(40,15))

#logout button

#logout=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

#text="Logout").grid(row=8, column=1,pady=(10,10),padx=(15,50))

#Query Button

#querybtn=Button(inlframer, text='Show Records', command=query, padx=13)

#querybtn.grid(row=7, column=1)

#total payslip button

viewpaymentbtn=Button(inlframer2, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="View Payslip",command=viewpay).grid(row=8, column=1,pady=10,padx=(50,50))

#print payslip button

printpaymentbtn=Button(inlframer2, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Print Payslip",command=printslip).grid(row=9, column=1,pady=10,padx=(50,50))

#total clear payslip button

clearbtn2=Button(inlframer2, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Clear Payslip",command=clearvp2).grid(row=10, column=1,pady=(10,105),padx=(50,50))

#total clear payslip button

clearbtn=Button(inlframer, padx=10, pady=10, fg="black", font=('arial', 12, 'bold'), width=14,

text="Clear All",command=clearvp).grid(row=8, column=1,pady=(10,210),padx=(40,15))

drop = ttk.Combobox(inlframel,value=["SELECT DEALERSHIP","NISSAN LIPA","NISSAN DASMA", "NISSAN SAN PABLO","NISSAN ALABANG","NISSAN CARMONA" ],width=17)

drop.current(0)

drop.grid(row=2,column=2)

#commit changes

conn.commit()

#close connection

conn.close()

root.mainloop()

chooselogin1()