VISVESVARAYA TECHNOLOGICAL UNIVERSITY

BELAGAVI

A Mini Project Report on

"OFFICE MANAGEMENT SOFTWARE"

Submitted in the partial fulfillment for the requirements for the conferment of degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

By

Mr. SANJAY PRABHAKAR USN: 1BY18CS141
Mr.SANTHOSH KUMAR USN: 1BY18CS143

Under the guidance of

Mrs. Ambika G.N
Assistant Professor
Department of CSE, BMSIT&M.

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

B.M.S. INSTITUTE OF TECHNOLOGY & MANAGEMENT
Yelahanka, BENGALURU-560064

2020-2021

B.M.S INSTITUTE OF TECHNOLOGY & MANAGEMENT

Yelahanka, BENGALURU-560064

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CERTIFICATE

This is to certify that the Mini Project work entitled **"PROJECT TITLE"** is a bonafide work carried out by **Mr. / Ms. NAME (1BY18CS***)** in partial fulfillment for the award of **Bachelor of Engineering Degree in Computer-Science and Engineering** of the Visvesvaraya Technological University, Belagavi during the year 2020-21. It is certified that all corrections/suggestions indicated for Internal Assessment have been incorporated in this report. The Mini project report has been approved as it satisfies the academic requirements in respect of project work for the B.E Degree.

Signature of the Guide	**Signature of the HOD**
Mrs. Ambika G.N	Dr. Anil G.N.

Name of the Examiners	**Signature with Date**
1.	
2.	

ABSTRACT

The main problem in offices nowadays is the use of work spaces efficiently, as a company grows and scales up in size it is really difficult to manage their space efficiently . This is an essential requirement for better functioning during team shufflings and to implement hot desking feature.

The main problem we are trying to solve here is managing the space in the office and increasing the productivity of employees, to do this we use the movement data collected from sensors to check where there is more activity and accordingly we can revamp the office.

ACKNOWLEDGEMENT

I am happy to present this Mini project after completing it successfully. This project would not have been possible without the guidance, assistance and suggestions of many individuals. I would like to express my deep sense of gratitude and indebtedness to each and every one who has helped me to make this project a success.

I heartily thank our **Principal, Dr. MOHAN BABU G N, BMS Institute of Technology & Management** for his constant encouragement and inspiration in taking up this Mini project.

I heartily thank our **Head of the Department, Dr. Anil G.N, Dept. of Computer Science and Engineering, BMS Institute of Technology & Management** for his constant encouragement and inspiration in taking up this Mini project.

I gracefully thank Project guide**, Mrs. Ambika G.N, Assistant Professor, Dept. of Computer Science and Engineering,** for his/her encouragement and advice throughout the course of the Project work.

Special thanks to all the staff members of Computer Science Department for their help and kind co-operation.

Lastly I thank my parents and friends for their encouragement and support given to me in order to finish this precious work.

By,

Sanjay Prabhakar
Santhosh Kumar A

CONTENTS

Chapters	**Page no**
Chapter 1: **Introduction**	06
1.1 Background	06
Chapter 2: **Literature Survey**	07
Chapter 3: **Software Requirements Specification**	10
3.1 Hardware requirements	10
3.2 Software requirements	11
Chapter 4: **Design**	**11**
4.1 Schema Diagram	11
4.2 Entity-Relationship Diagram	12
Chapter 5: **Implementation**	**13**
5.1 Implementation with Screen shot	13
Chapter 6: **Conclusion & Future Enhancement**	**53**
6.1 Conclusion	53
6.2 Future Enhancement	53
Chapter 7: **References**	54

INTRODUCTION

The main problem we are trying to solve here is managing the space in the office and increasing the productivity of employees, to do this we use the movement data collected from sensors to check where there is more activity and accordingly we can revamp the office.

We also hope to help the employees in their day to day tasks easier, according to the surveys conducted most of the MNC's still use Outlook to book rooms, which is inefficient. The functionality of having features like booking a room, selecting your seat, and other features will help the employees a lot.

The hot-desking feature , ie the employees can change their places every day thereby increasing their productivity.

In the future, I hope to expand this to the mobile platform as well and add extra functionalities like attendance, leave tracking, salary tracking, and other features. I plan on adding all these features and try to take it to the level of all in one office management software.

Qt for Python

Qt for Python offers the official Python bindings for Qt, and has two main components:

- PySide6, so that you can use Qt6 APIs in your Python applications, and
- Shiboken6, a binding generator tool, which can be used to expose C++ projects to Python, and a Python module with some utility functions.

Porting from PySide2 to PySide6 provides information on porting existing PySide2 applications.

Qt for Python Modules¶

Basic modules¶

These are the main modules that help you build a Widget-based UI.

Qt Core	Provides core non-GUI functionality, like signal and slots, properties, base classes of item models, serialization, and more.
Qt GUI	Extends QtCore with GUI functionality: Events, windows and screens, OpenGL and raster-based 2D painting, as well as images.
Qt Widgets	Provides ready to use Widgets for your application, including graphical elements for your UI.

QML and Qt Quick¶

Use these modules to interact with the QML Language <https://doc.qt.io/qt-5.qmlapplications>, from Python.

Qt QML	The base Python API to interact with the module.
Qt Quick	Provides classes to embed Qt Quick in Qt applications.

Qt QuickWidgets	Provides the QQuickWidget class to embed Qt Quick in widget-based applications.

Data visualization¶

Charts, diagrams, animations: these modules provide classes to help you include these elements in your UI.

Qt Charts	Provides a set of easy to use chart components.
Qt DataVisualization	Provides a way to visualize data in 3D as bar, scatter, or surface graphs.

Multimedia¶

Audio, video, and hardware interaction: use these modules for multimedia solutions.

Qt Multimedia	Provides low-level multimedia functionality.
Qt MultimediaWidgets	Provides the widget-based multimedia API.

WebEngine¶

If your project is based on a browser or the features around Web-based applications, use these modules to interact with them.

Qt WebEngineWidgets	Provides widgets to handle Web content.

Technical Feasability

The proposed system is developed by Pyqt5 as front –end and Sqlite3 as back-end and efficient under windows 10. It offers unmatched scalability to facilitate the management of

deeply embedded apps using a smaller footprint, even in massive warehouses that stack terabytes of data. On-demand flexibility is the star feature of Sqlite3. This open-source solution allows complete customization to eCommerce businesses with unique database server requirements.

Operational Feasability

This feasibility is carried in order to know whether the system will work with least difficulties when it is developed and installed. And also check whether it is developed and installed. And also check whether it support graphical user interface in this system. Hence it is concluded that the system is operationally feasible.

SOFTWARE REQUIREMENTS SPECIFICATION

PC GENERAL SPECIFICATION:

Hard disk : 20GB

RAM : 1GB

Processor : Pentium

ENVIRONMENT:

• PyQt5

• Sqlite3

Employee Details:

Here we store all the data of existing and all the new employees that will be joining the office.

Meeting Details:

Here we store all the data of the meetings booked by employees and the meeting rooms which they have booked and the time also

Equipment Details:

Here we store information about all the equipments in the office and details about how to use them , making employees life easier.

Feedback Details:

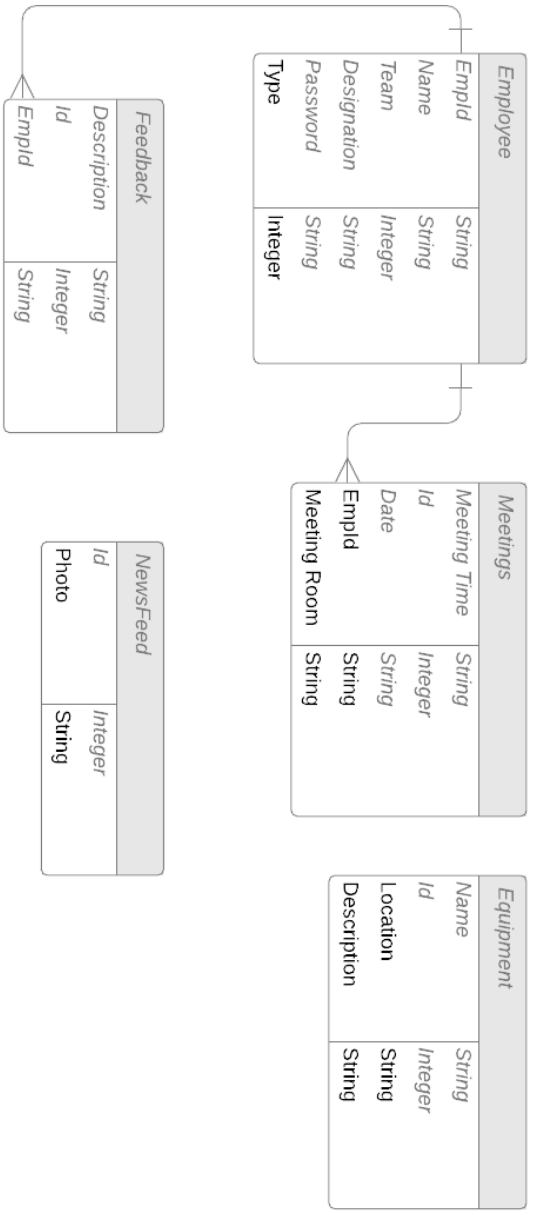
Here we store all the feedbacks and issues faced by the employees which have to be cleared or taken action on accordingly.

Newsfeed Details:

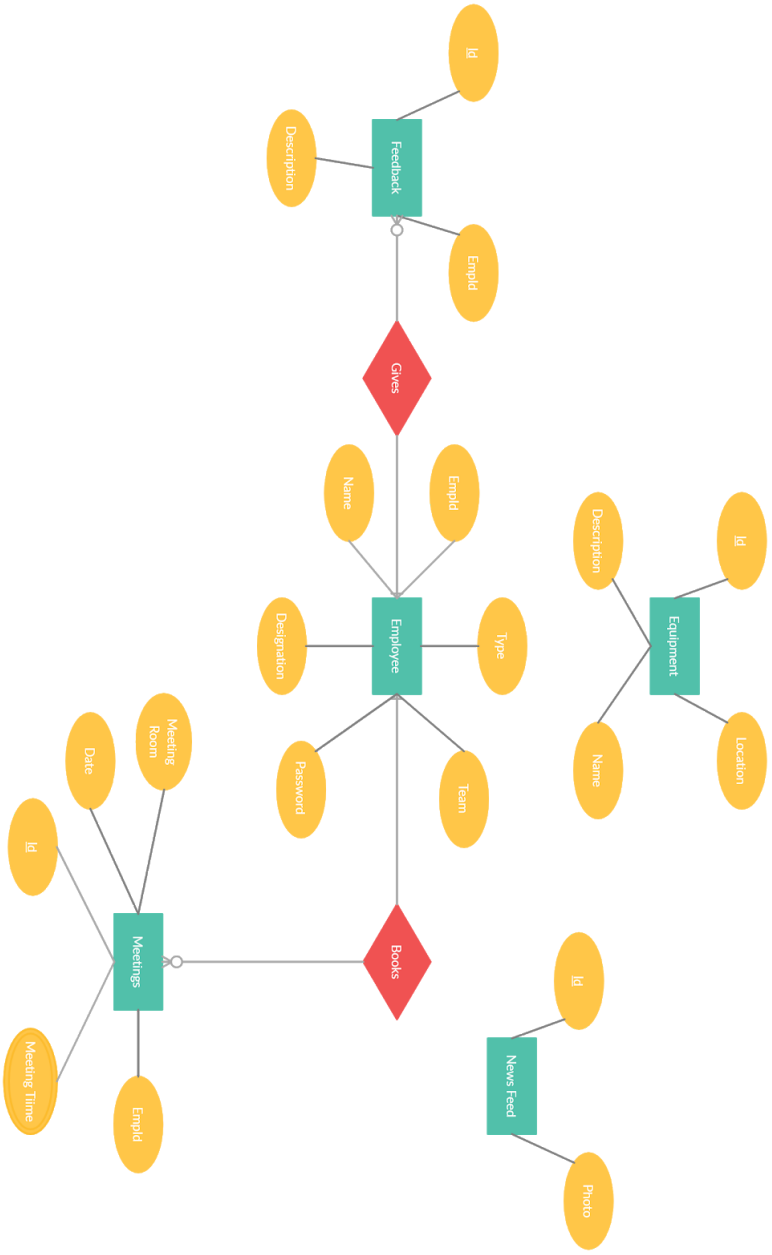
Here we store all the news and information that the admin wants to convey to its employees

DESIGN

SCHEMA DIAGRAM:



ER DIAGRAM:


Id
Description
Feedback
EmpId
Gives
Name
EmpId
Description
Id
Designation
Employee
Type
Equipment
Password
Team
Name
Location
Date
Meeting Room
Id
Meetings
Books
Id
News Feed
Meeting Time
EmpId
Photo

IMPLEMENTATION

CODE FOR LOGIN:

```
import sys


from PyQt5.QtCore import QSize, Qt

from PyQt5.QtGui import QIcon, QPixmap

from PyQt5.QtWidgets import *

import sqlite3

import data




con = sqlite3.connect('empolyee.db')

cur = con.cursor()

from MainAdm import admmain

from MainEmp import empmain
```

```
class Loginwin(QWidget):
    def __init__(self):
        super().__init__()
        self.setGeometry(750, 100, 450, 400)
        self.setWindowTitle("Login")
        self.setWindowFlag(Qt.FramelessWindowHint)
        self.layouts()


        self.show()


    def layouts(self):
        self.mainvbox = QVBoxLayout()
        self.setLayout(self.mainvbox)
        self.setStyleSheet("background:#fef2da;font-size:12pt")
        self.mainvbox.addStretch()
        self.usnlabel = QLabel("Employee Id:", self)
        self.passlabel = QLabel("Password", self)
        self.usn = QLineEdit(self)
        self.usn.setMinimumHeight(40)
        self.usn.setStyleSheet("background:#fffbf4;border-radius:10px")
        self.usn.setPlaceholderText(" Enter Employee ID")
        self.passworde = QLineEdit(self)
```

```
self.passworde.setMinimumHeight(40)
self.passworde.setStyleSheet("background:#fffbf4;border-radius:10px")
self.passworde.setPlaceholderText(" Enter Password")
self.passworde.setEchoMode(QLineEdit.Password)
self.passworde.returnPressed.connect(self.login)
self.hbox=QHBoxLayout()
self.btn = QToolButton( self)
self.btn1 = QToolButton( self)
self.hbox.setAlignment(Qt.AlignCenter)
self.hbox.addWidget(self.btn)
self.btn.setIcon(QIcon("Data/login.png"))
self.btn.setIconSize(QSize(120,40))
self.btn.resize(QSize(120, 40))
self.btn1.setIcon(QIcon("Data/close.png"))
self.btn1.setIconSize(QSize(41,35))
self.btn1.clicked.connect(self.close)
self.btn1.setStyleSheet("background:transparent;")
self.btn1.resize(QSize(41, 25))
self.imageComputer = QLabel()
self.imageComputer.setPixmap(QPixmap("Data/Logo Long@2x.png").scaled(825 /
1.25, 303 / 1.25))
self.btn.setStyleSheet("background-color:transparent ; border-style:outset")
self.hbox1=QHBoxLayout()
```

```
self.hbox2=QVBoxLayout()
self.hbox1.addWidget(self.imageComputer)
self.hbox2.addWidget(self.btn1)
self.hbox2.addStretch()
self.hbox2.addSpacing(200)

self.hbox1.setAlignment(Qt.AlignCenter)
self.hbox2.setAlignment(Qt.AlignRight)
self.mainvbox.addLayout(self.hbox2,2)
self.mainvbox.addLayout(self.hbox1,10)
self.mainvbox.addSpacing(25)
self.mainvbox.addWidget(self.usnlabel,10)
self.mainvbox.addSpacing(25)
self.mainvbox.addWidget(self.usn,10)
self.mainvbox.addSpacing(25)
self.mainvbox.addWidget(self.passlabel,10)
self.mainvbox.addSpacing(25)
self.mainvbox.addWidget(self.passworde,10)
self.mainvbox.addSpacing(25)
self.mainvbox.addLayout(self.hbox,58)
self.mainvbox.addStretch()

self.btn.clicked.connect(self.login)
```

```
def login(self):
    query = "SELECT Empid,Password,Type from employee"
    employees = cur.execute(query).fetchall()
    print(employees)
    data.id= self.usn.text()
    passw = self.passworde.text()
    flag = 0
    for employee in employees:
        if (data.id, passw) == (employee[0], employee[1]):
            data.type = employee[2]
            flag=1
    if flag==0:
        mbox = QMessageBox.information(self, "Information", "Wrong employee id or
  password")
    else:
        if data.type == 2:
            self.emplmain = empmain()
            self.close()
        if data.type == 1:
            self.admimain = admmain()
            self.close()
```

```
App = QApplication(sys.argv)
window = Loginwin()
sys.exit(App.exec())
```

CODE FOR MAIN ADMIN:

```
import sqlite3
import sys
import Admin
from PyQt5.QtCore import QSize, Qt, QUrl
from PyQt5.QtGui import QImage, QPalette, QBrush, QPixmap, QIcon
from PyQt5.QtWidgets import *
import Feedback
import data
import Meetings
import Equipment
import seat
con = sqlite3.connect('empolyee.db')
cur = con.cursor()
con1 = sqlite3.connect('newsfeed.db')
cur1 = con1.cursor()
from PyQt5.QtGui import QDesktopServices
import date
```

```
from PyQt5.QtCore import QTimer


class admmain(QWidget):
    def __init__(self):
        super().__init__()
        self.id=id
        self.i=0
        self.setGeometry(100, 100, 2000,1000)
        self.setWindowTitle("Admin Main")
        self.setWindowFlag(Qt.FramelessWindowHint)
        self.timer = QTimer()
        self.timer.setInterval(3000)
        self.timer.start()
        self.timer1 = QTimer()
        self.timer1.setInterval(10000)
        self.timer1.start()
        self.timer.timeout.connect(self.change)
        self.timer1.timeout.connect(self.changes)
        self.imageComputer = QLabel(self)

      self.setStyleSheet("QWidget{background:#fef2da;font-size:12pt;}QLineEdit{backgro
      und:#fffbf4;border-radius:10px}QComboBox{background:white;border-radius:10px}
      QTableWidget{background:white;border-radius:10px}QListWidget{background:whit
      e;border-radius:10px}")
        self.imageComputer.setPixmap(QPixmap("Data/Plan.jpg").scaled(1644, 959))
```

```
self.imageComputer.move(18,62-8)
self.imageComputer1 = QLabel(self)
self.imageComputer1.setPixmap(QPixmap("Data/Help Click.png").scaled(235, 236))
self.imageComputer1.move(1676, 665)
self.btn = QToolButton(self)
self.btn.setIcon(QIcon())
self.btn.setIconSize(QSize(50, 50))
self.btn.resize(QSize(50, 50))
self.btn.setStyleSheet("background-color:transparent ; border-style:outset")
self.btn.clicked.connect(self.gotolink)
self.imageComputer2 = QLabel(self)
self.seatbtn=QToolButton(self)
self.seatbtn.setIcon(QIcon("Data/myseat.png"))
self.seatbtn.setIconSize(QSize(225,75))
self.seatbtn.resize(QSize(240,90))
self.seatbtn.setStyleSheet("QToolButton{background-color:transparent ;
border-style:outset; border-radius:10px} QToolButton:hover{background:#fac34b;}")
self.seatbtn.move(1680-8,74-30)
self.seatbtn.clicked.connect(self.seats)
self.equipbtn = QToolButton(self)
self.equipbtn.setIcon(QIcon("Data/equipment.png"))
self.equipbtn.setIconSize(QSize(225,75))
self.equipbtn.resize(QSize(240, 90))
```

```
 self.equipbtn.setStyleSheet("QToolButton{background-color:transparent ;
border-style:outset; border-radius:10px} QToolButton:hover{background:#fac34b;}")

 self.equipbtn.move(1680-8,293-30)

 self.equipbtn.clicked.connect(self.equipment)

 self.feedbtn = QToolButton(self)

 self.feedbtn.setIcon(QIcon("Data/feedback.png"))

 self.feedbtn.setIconSize(QSize(225,75))

 self.feedbtn.resize(QSize(240, 90))

 self.feedbtn.setStyleSheet("QToolButton{background-color:transparent ;
border-style:outset; border-radius:10px} QToolButton:hover{background:#fac34b;}")

 self.feedbtn.move(1680-8, 401-30)

 self.feedbtn.clicked.connect(self.feedbacks)

 self.accntbtn = QToolButton(self)

 self.accntbtn.setIcon(QIcon("Data/Admin@2x.png"))

 self.accntbtn.setIconSize(QSize(225,75))

 self.accntbtn.resize(QSize(240, 90))

 self.accntbtn.setStyleSheet("QToolButton{background-color:transparent ;
border-style:outset; border-radius:10px} QToolButton:hover{background:#fac34b;}")

 self.accntbtn.move(1680-8, 512-30)

 self.accntbtn.clicked.connect(self.admin)

 self.bookbtn = QToolButton(self)

 self.bookbtn.setIcon(QIcon("Data/meetings.png"))

 self.bookbtn.setIconSize(QSize(225,75))

 self.bookbtn.resize(QSize(240, 90))
```

```
 self.bookbtn.setStyleSheet("QToolButton{background-color:transparent ;
border-style:outset; border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.bookbtn.move(1680-8, 183-30)
 self.bookbtn.clicked.connect(self.book)
 self.helpbtn = QToolButton(self)
 self.helpbtn.setIcon(QIcon("Data/help.png"))
 self.helpbtn.setIconSize(QSize(225, 75))
 self.helpbtn.resize(QSize(225, 75))
 self.helpbtn.setStyleSheet("background-color:transparent ; border-style:outset")
 self.helpbtn.move(1751, 946-30)
 self.helpbtn.clicked.connect(self.help)
 self.helpbtn1 = QToolButton(self)
 self.helpbtn1.setIcon(QIcon("Data/Home@2x.png"))
 self.helpbtn1.setIconSize(QSize(75, 75))
 self.helpbtn1.resize(QSize(75, 75))
 self.helpbtn1.setStyleSheet("background-color:transparent ; border-style:outset")
 self.helpbtn1.move(1751 - 40, 946 - 30)
 self.helpbtn1.clicked.connect(self.home)
 self.closebtn = QToolButton(self)
 self.closebtn.setIcon(QIcon("Data/close.png"))
 self.closebtn.setIconSize(QSize(41,25))
 self.closebtn.resize(QSize(41,25))
 self.closebtn.setStyleSheet("background-color:transparent ; border-style:outset")
```

```
    self.closebtn.move(1877, 0)
    self.closebtn.clicked.connect(self.close)
    self.minimbtn = QToolButton(self)
    self.minimbtn.setIcon(QIcon("Data/minimise.png"))
    self.minimbtn.setIconSize(QSize(41, 25))
    self.minimbtn.resize(QSize(41, 25))
    self.minimbtn.setStyleSheet("background-color:transparent ; border-style:outset")
    self.minimbtn.move(1834, 0)
    self.minimbtn.clicked.connect(self.showMinimized)
    self.get()


    self.j=0
    self.showMaximized()
    self.display()
def home(self):
    pass
def help(self):
    print("HI")
    if self.j==0:
        self.imageComputer2.setPixmap(QPixmap("Data/mainad.png").scaled(3840 / 2,
  2080 / 2))
        self.imageComputer2.resize(2000,1030)
        self.j=1
```

```
    else:
        self.imageComputer2.setPixmap(QPixmap("Data/Main T@2x
  (1).png").scaled(3840 / 2, 2080 / 2))
        self.imageComputer2.resize(0, 0)
        self.j=0
    print("ok")


def ok(self):
    self.t.stop()
    print("now")
    self.imageComputer2.setPixmap(QPixmap())
def changes(self):
    self.timer1.stop()
    self.imageComputer1.setPixmap(QPixmap())


def gotolink(self):
    if self.i==2 or self.i==3:
        QDesktopServices.openUrl(QUrl("https://bmsit.ac.in/"))
def book(self):
    self.book=Meetings.Meetings()
    self.close()
def feedbacks(self):
    self.feed=Feedback.Feedbacks()
```

```
    self.close()
def admin(self):
    self.admin=Admin.admin()
    self.close()
def equipment(self):
    self.eq=Equipment.Equipment()
    self.close()
def seats(self):
    self.seatts=date.date()
    self.close()
def display(self):
    query = "SELECT Location from employee WHERE Empid=?;"
    employees = cur.execute(query,(data.id,)).fetchone()
    print(employees)
    x=employees[0]
    if x!=None:
        x=x.split(",")
        self.posx=(int(x[0]))*0.9264+18
        self.posy=(int(x[1]))*0.9265+62-8
        print(x[0])
        print(x[1])
        self.btn.setIcon(QIcon("Data/seats.png"))
        self.btn.move(self.posx,self.posy)
```

```
        data.occ=True
def change(self):
    print("timeout")
    self.i=self.i+1


    print(len(self.employees))
    if self.i==len(self.employees):
        self.i=0
    if self.i==0:
        print("done")
        print(self.i)
        self.btn.setIcon(QIcon("Data/seats.png"))
        self.imageComputer.setPixmap(QPixmap("Data/Plan.jpg").scaled(1644, 959))
        print("done")
        self.display()
        if data.occ==True:
            self.btn.move(self.posx, self.posy)
            self.btn.resize(QSize(50,50))
        else:
            self.btn.setIcon(QIcon())


    else:
        print(self.i)
```

```
        print(self.employees[self.i])
        self.imageComputer.setPixmap(QPixmap(self.employees[self.i][0]).scaled(1644,
    959))
        self.btn.setIcon(QIcon("Data/More@2x.png"))
        self.btn.setIconSize(QSize(100,40))
        self.btn.resize(QSize(100, 40))
        self.btn.move(803,864)
    def get(self):
        print(1)
        query = "SELECT photo FROM feed"
        self.employees = cur1.execute(query).fetchall()
        print("hi")
        print(len(self.employees))
```

CODE FOR MAIN EMPLOYEE:

```
import sqlite3
import sys
import Equipment
from PyQt5.QtCore import QSize, Qt, QTimer, QUrl
from PyQt5.QtGui import QImage, QPalette, QBrush, QPixmap, QIcon,
    QDesktopServices
from PyQt5.QtWidgets import *
import Feedback
import Meetings
import Account
```

```
import seat
import data
import date
con = sqlite3.connect('empolyee.db')
cur = con.cursor()
con1 = sqlite3.connect('newsfeed.db')
cur1 = con1.cursor()


class empmain(QWidget):
    def __init__(self):
        super().__init__()
        self.setGeometry(100, 100, 2000,1000)
        self.setWindowTitle("Employee Main")


        self.setWindowFlag(Qt.FramelessWindowHint)
        self.timer = QTimer()
        self.timer.setInterval(3000)
        self.timer.start()
        self.timer.timeout.connect(self.change)
        self.imageComputer = QLabel(self)

    self.setStyleSheet("QWidget{background:#fef2da;font-size:12pt;}QLineEdit{backgro
    und:#fffbf4;border-radius:10px}QComboBox{background:white;border-radius:10px}
    QTableWidget{background:white;border-radius:10px}QListWidget{background:whit
    e;border-radius:10px}")
```

```
size = self.size()
self.imageComputer.setPixmap(QPixmap("Data/Plan.jpg").scaled(1644, 959))
self.imageComputer.move(18,62-8)




self.btn = QToolButton(self)
self.btn.setIcon(QIcon())
self.btn.setIconSize(QSize(50, 50))
self.btn.resize(QSize(50, 50))
self.btn.setStyleSheet("background-color:transparent ; border-style:outset")
self.btn.clicked.connect(self.gotolink)
self.display()
self.timer1 = QTimer()
self.timer1.setInterval(10000)
self.timer1.start()
self.timer1.timeout.connect(self.changes)
self.imageComputer1 = QLabel(self)
self.imageComputer1.setPixmap(QPixmap("Data/Help Click.png").scaled(235, 236))
self.imageComputer1.move(1676, 665)
self.imageComputer1.setStyleSheet("background:transparent")
self.imageComputer2 = QLabel(self)
self.closebtn = QToolButton(self)
```

```
self.closebtn.setIcon(QIcon("Data/close.png"))
self.closebtn.setIconSize(QSize(41, 25))
self.closebtn.resize(QSize(41, 25))
self.closebtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.closebtn.move(1877, 0)
self.closebtn.clicked.connect(self.close)
self.minimbtn = QToolButton(self)
self.minimbtn.setIcon(QIcon("Data/minimise.png"))
self.minimbtn.setIconSize(QSize(41, 25))
self.minimbtn.resize(QSize(41, 25))
self.minimbtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.minimbtn.move(1834, 0)
self.minimbtn.clicked.connect(self.showMinimized)
self.seatbtn = QToolButton(self)
self.seatbtn.setIcon(QIcon("Data/myseat.png"))
self.seatbtn.setIconSize(QSize(225, 75))
self.seatbtn.resize(QSize(240, 90))
self.seatbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
self.seatbtn.move(1680 - 8, 74 - 30)
self.seatbtn.clicked.connect(self.seats)
self.equipbtn = QToolButton(self)
```

```
 self.equipbtn.setIcon(QIcon("Data/equipment.png"))
 self.equipbtn.setIconSize(QSize(225, 75))
 self.equipbtn.resize(QSize(240, 90))
 self.equipbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.equipbtn.move(1680 - 8, 293 - 30)
 self.equipbtn.clicked.connect(self.equipment)
 self.feedbtn = QToolButton(self)
 self.feedbtn.setIcon(QIcon("Data/feedback.png"))
 self.feedbtn.setIconSize(QSize(225, 75))
 self.feedbtn.resize(QSize(240, 90))
 self.feedbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.feedbtn.move(1680 - 8, 401 - 30)
 self.feedbtn.clicked.connect(self.feedbacks)
 self.accntbtn = QToolButton(self)
 self.accntbtn.setIcon(QIcon("Data/account.png"))
 self.accntbtn.setIconSize(QSize(225, 75))
 self.accntbtn.resize(QSize(240, 90))
 self.accntbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
```

```
 self.accntbtn.move(1680 - 8, 512 - 30)
 self.accntbtn.clicked.connect(self.accnt)
 self.bookbtn = QToolButton(self)
 self.bookbtn.setIcon(QIcon("Data/meetings.png"))
 self.bookbtn.setIconSize(QSize(225, 75))
 self.bookbtn.resize(QSize(240, 90))
 self.bookbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")

 self.bookbtn.move(1680 - 8, 183 - 30)
 self.bookbtn.clicked.connect(self.book)
 self.helpbtn = QToolButton(self)
 self.helpbtn.setIcon(QIcon("Data/help.png"))
 self.helpbtn.setIconSize(QSize(225, 75))
 self.helpbtn.resize(QSize(225, 75))
 self.helpbtn.setStyleSheet("background-color:transparent ; border-style:outset")
 self.helpbtn.move(1751, 946 - 30)
 self.helpbtn.clicked.connect(self.help)
 self.helpbtn1 = QToolButton(self)
 self.helpbtn1.setIcon(QIcon("Data/Home@2x.png"))
 self.helpbtn1.setIconSize(QSize(75, 75))
 self.helpbtn1.resize(QSize(75, 75))
```

```
        self.helpbtn1.setStyleSheet("background-color:transparent ; border-style:outset")
        self.helpbtn1.move(1751 - 40, 946 - 30)
        self.helpbtn1.clicked.connect(self.home)
        self.showMaximized()
        self.i=0
        self.j=0
        self.get()
    def home(self):
        pass
    def changes(self):
        self.timer1.stop()
        self.imageComputer1.setPixmap(QPixmap())
    def gotolink(self):
        if self.i==2 or self.i==3:
            QDesktopServices.openUrl(QUrl("https://bmsit.ac.in/"))
    def book(self):
        self.book=Meetings.Meetings()
        self.close()
    def equipment(self):
        self.eq=Equipment.Equipment()
        self.close()
    def feedbacks(self):
        self.feed=Feedback.Feedbacks()
```

```
    self.close()
def accnt(self):
    self.accnt=Account.Accnt()
    self.close()
def seats(self):
    self.seatts = date.date()
    self.close()
def display(self):
    query = "SELECT Location from employee WHERE Empid=?;"
    employees = cur.execute(query,(data.id,)).fetchone()
    print(employees)
    x=employees[0]
    if x != None:
        x = x.split(",")
        self.posx = (int(x[0])) * 0.9264 + 18
        self.posy = (int(x[1])) * 0.9265 + 62-8
        print(x[0])
        print(x[1])
        self.btn.setIcon(QIcon("Data/seats.png"))
        self.btn.move(self.posx, self.posy)
        data.occ=True
def change(self):
    print("timeout")
```

```
 self.i=self.i+1
 if self.i==len(self.employees):
    self.i=0
 if self.i==0:


    print("in if")


    self.imageComputer.setPixmap(QPixmap(self.employees[self.i][0]).scaled(1644,
959))
    self.btn.setIcon(QIcon("Data/seats.png"))
    self.btn.setIconSize(QSize(50, 50))
    self.btn.resize(QSize(50, 50))


    if data.occ==True:
        self.btn.move(self.posx, self.posy)
    else:
        self.btn.setIcon(QIcon())
 else:
    print(self.i)
    print(self.employees[self.i])
    self.imageComputer.setPixmap(QPixmap(self.employees[self.i][0]).scaled(1644,
959))
    self.btn.setIcon(QIcon("Data/More@2x.png"))
    self.btn.setIconSize(QSize(100,40))
```

```
        self.btn.resize(QSize(100, 40))
        self.btn.clicked.connect(self.gotolink)
        self.btn.move(803,864)
def get(self):
    print(1)
    query = "SELECT photo FROM feed"
    self.employees = cur1.execute(query).fetchall()
    print("hi")
    print(len(self.employees))
def help(self):
    print("HI")
    if self.j==0:
        self.imageComputer2.setPixmap(QPixmap("Data/main.png").scaled(3840 / 2, 2080
  / 2))


        self.imageComputer2.resize(2000,1030)
        self.imageComputer2.move(0,-2)
        self.j=1
    else:
        self.imageComputer2.setPixmap(QPixmap("Data/Main T@2x
  (1).png").scaled(3840 / 2, 2080 / 2))
        self.imageComputer2.resize(0, 0)
```

```
            self.j=0
        print("ok")
```

CODE FOR EQUIPMENT:

```
import sys
from PyQt5.QtCore import QSize, Qt
from PyQt5.QtGui import QImage, QPalette, QBrush, QPixmap, QIcon
from PyQt5.QtWidgets import *
import sqlite3
import Feedback
import Meetings
import Account
import MainEmp
import MainAdm
import Admin
import data
import seat
import date
con = sqlite3.connect('equipment.db')
cur = con.cursor()



class Equipment(QWidget):
    def __init__(self):
```

```
 super().__init__()

 self.setGeometry(100, 100, 2000, 1000)

 self.setWindowTitle("Employee Main")

 self.setWindowFlag(Qt.FramelessWindowHint)

 self.photo = QLabel(self)


self.setStyleSheet("QWidget{background:#fef2da;font-size:12pt;}QLineEdit{backgro
und:#fffbf4;border-radius:10px}QComboBox{background:white;border-radius:10px}
QTableWidget{background:white;border-radius:10px}QListWidget{background:whit
e;border-radius:10px}")

 self.photo.resize(QSize(300, 300))

 self.photo.setPixmap(QPixmap("Data/Equip Default.png").scaled(300, 300))

 self.photo.move(1000+170, 0)

 self.locationlbl = QLabel("Location:", self)

 self.description = QLabel("Description:", self)

 self.destext = QTextEdit(self)

 self.destext.setStyleSheet("background:white;border-radius:10px")

 self.name = QLabel("Name:", self)

 self.nams = QLineEdit(self)

 self.location = QLineEdit(self)

 self.name.move(1000+170, 323)

 self.nams.move(1090+170, 323)

 self.locationlbl.move(1000+170, 353)

 self.location.move(1090+170, 353)

 self.location.setMinimumWidth(250)
```

```
self.nams.setMinimumWidth(250)
self.description.move(1000+170, 383)
self.destext.move(1000+170, 410)
self.destext.resize(QSize(400, 500))
self.eqplbl = QLabel("Equipment:", self)
self.eqplbl.move(100, 220+110-250)
self.srch = QLineEdit(self)
self.srch.setPlaceholderText("Enter Equipment Name to Search")
self.srch.resize(QSize(600+57+170, 50))
self.srch.move(100, 260+110-250)
self.srcbtn = QToolButton(self)
self.srcbtn.setIcon(QIcon("Data/search.png"))
self.srcbtn.setIconSize(QSize(40,40))
self.srcbtn.resize(QSize(40, 40))
self.srcbtn.setStyleSheet("background-color:transparent ;border-style:outset ")
self.srcbtn.move(700+57+170, 260+110-250)
self.srcbtn.clicked.connect(self.search)
self.equiplist = QListWidget(self)
self.equiplist.move(100, 300+110-250)
self.equiplist.resize(QSize(700+170, 500+250))
self.equiplist.itemClicked.connect(self.singleClicks)
self.seatbtn = QToolButton(self)
self.seatbtn.setIcon(QIcon("Data/myseat.png"))
```

```
 self.seatbtn.setIconSize(QSize(225, 75))
 self.seatbtn.resize(QSize(240, 90))
 self.seatbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.seatbtn.move(1680 - 8, 74 - 30)
 self.seatbtn.clicked.connect(self.seats)
 self.equipbtn = QToolButton(self)
 self.equipbtn.setIcon(QIcon("Data/equipment.png"))
 self.equipbtn.setIconSize(QSize(225, 75))
 self.equipbtn.resize(QSize(240, 90))
 self.equipbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.equipbtn.move(1680 - 8, 293 - 30)
 self.feedbtn = QToolButton(self)
 self.feedbtn.setIcon(QIcon("Data/feedback.png"))
 self.feedbtn.setIconSize(QSize(225, 75))
 self.feedbtn.resize(QSize(240, 90))
 self.feedbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.feedbtn.move(1680 - 8, 401 - 30)
 self.feedbtn.clicked.connect(self.feedbacks)
```

```
 self.accntbtn = QToolButton(self)
 if data.type == 1:
    self.accntbtn.setIcon(QIcon("Data/Admin@2x.png"))
 else:
    self.accntbtn.setIcon(QIcon("Data/account.png"))
 self.accntbtn.setIconSize(QSize(225, 75))
 self.accntbtn.resize(QSize(240, 90))
 self.accntbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.accntbtn.move(1680 - 8, 512 - 30)
 self.accntbtn.clicked.connect(self.accnt)
 self.bookbtn = QToolButton(self)
 self.bookbtn.setIcon(QIcon("Data/meetings.png"))
 self.bookbtn.setIconSize(QSize(225, 75))
 self.bookbtn.resize(QSize(240, 90))
 self.bookbtn.setStyleSheet(
    "QToolButton{background-color:transparent ; border-style:outset;
border-radius:10px} QToolButton:hover{background:#fac34b;}")
 self.bookbtn.move(1680 - 8, 183 - 30)
 self.bookbtn.clicked.connect(self.book)
 self.helpbtn = QToolButton(self)
 self.helpbtn.setIcon(QIcon("Data/help.png"))
 self.helpbtn.setIconSize(QSize(225, 75))
```

```
self.helpbtn.resize(QSize(225, 75))
self.helpbtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.helpbtn.move(1751, 946 - 30)
self.helpbtn.clicked.connect(self.help)
self.closebtn = QToolButton(self)
self.closebtn.setIcon(QIcon("Data/close.png"))
self.closebtn.setIconSize(QSize(41, 25))
self.closebtn.resize(QSize(41, 25))
self.closebtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.closebtn.move(1877, 0)
self.closebtn.clicked.connect(self.close)
self.minimbtn = QToolButton(self)
self.minimbtn.setIcon(QIcon("Data/minimise.png"))
self.minimbtn.setIconSize(QSize(41, 25))
self.minimbtn.resize(QSize(41, 25))
self.minimbtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.minimbtn.move(1834, 0)
self.minimbtn.clicked.connect(self.showMinimized)
op = QGraphicsOpacityEffect(self)
op.setOpacity(0.3)
op1 = QGraphicsOpacityEffect(self)
op1.setOpacity(0.3)
op2 = QGraphicsOpacityEffect(self)
```

```
op2.setOpacity(0.3)
op3 = QGraphicsOpacityEffect(self)
op3.setOpacity(0.3)
self.feedbtn.setGraphicsEffect(op)
self.feedbtn.setAutoFillBackground(True)
self.accntbtn.setGraphicsEffect(op1)
self.accntbtn.setAutoFillBackground(True)
self.seatbtn.setGraphicsEffect(op2)
self.seatbtn.setAutoFillBackground(True)
self.bookbtn.setGraphicsEffect(op3)
self.bookbtn.setAutoFillBackground(True)
self.getEquipment()
self.imageComputer2 = QLabel(self)
self.helpbtn = QToolButton(self)
self.helpbtn.setIcon(QIcon("Data/help.png"))
self.helpbtn.setIconSize(QSize(225, 75))
self.helpbtn.resize(QSize(225, 75))
self.helpbtn.setStyleSheet("background-color:transparent ; border-style:outset")
self.helpbtn.move(1751, 946 - 30)
self.helpbtn.clicked.connect(self.help)
self.helpbtn1 = QToolButton(self)
self.helpbtn1.setIcon(QIcon("Data/Home@2x.png"))
self.helpbtn1.setIconSize(QSize(75, 75))
```

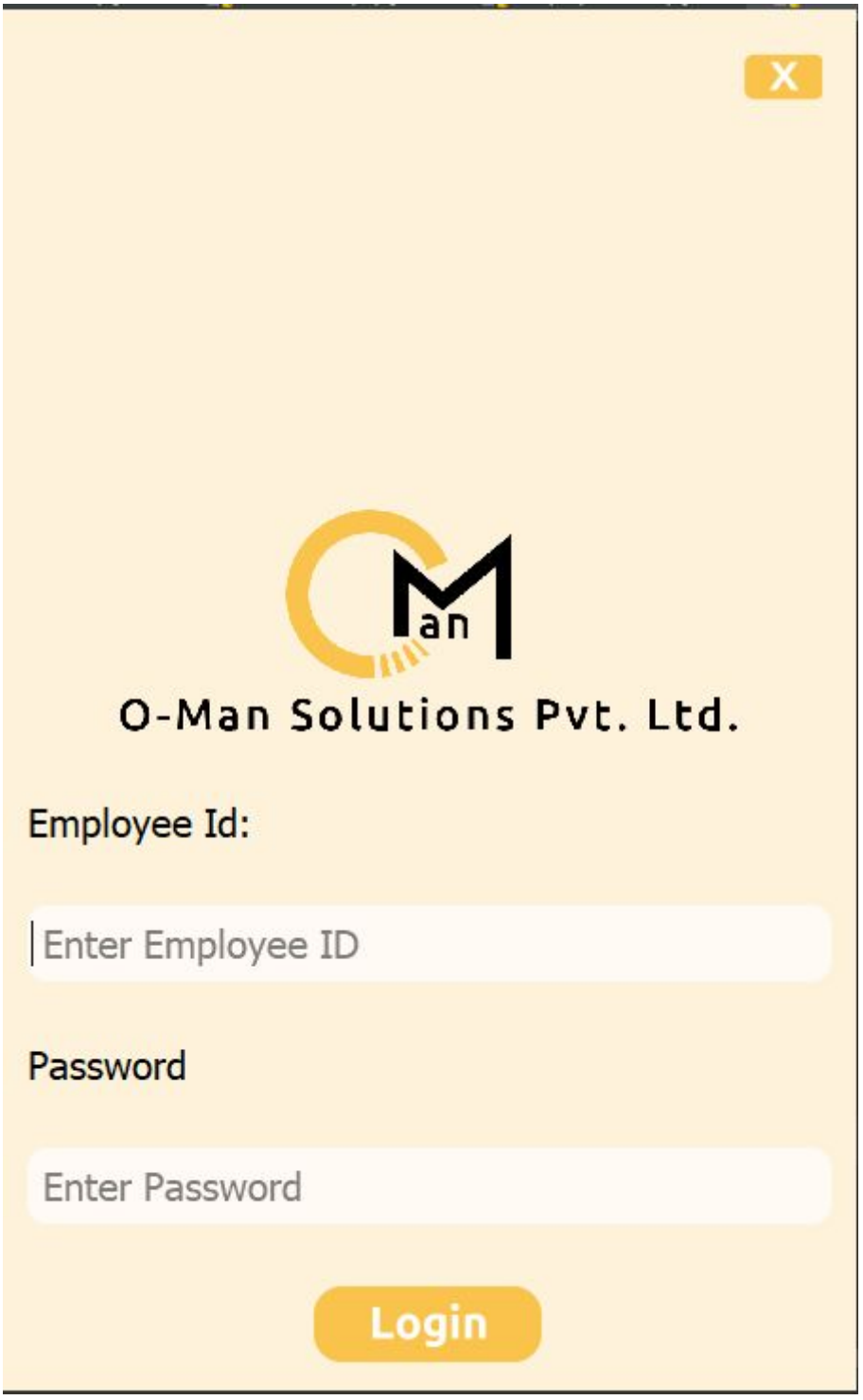
```
    self.helpbtn1.resize(QSize(75, 75))
    self.helpbtn1.setStyleSheet("background-color:transparent ; border-style:outset")
    self.helpbtn1.move(1751 - 40, 946 - 30)
    self.helpbtn1.clicked.connect(self.home)
    self.i = 0
    self.showMaximized()
def home(self):
    if data.type==1:
        self.adm=MainAdm.admmain()
    else:
        self.emp=MainEmp.empmain()
    self.close()
def help(self):
    print("HI")
    if self.i==0:
        self.imageComputer2.setPixmap(QPixmap("Data/Equip Help.png").scaled(756,
  1029))
        self.imageComputer2.resize(756,1029)
        self.imageComputer2.move(498,33)
        self.imageComputer2.setStyleSheet("background:transparent")
        self.i=1
        print("in")
    else:
```

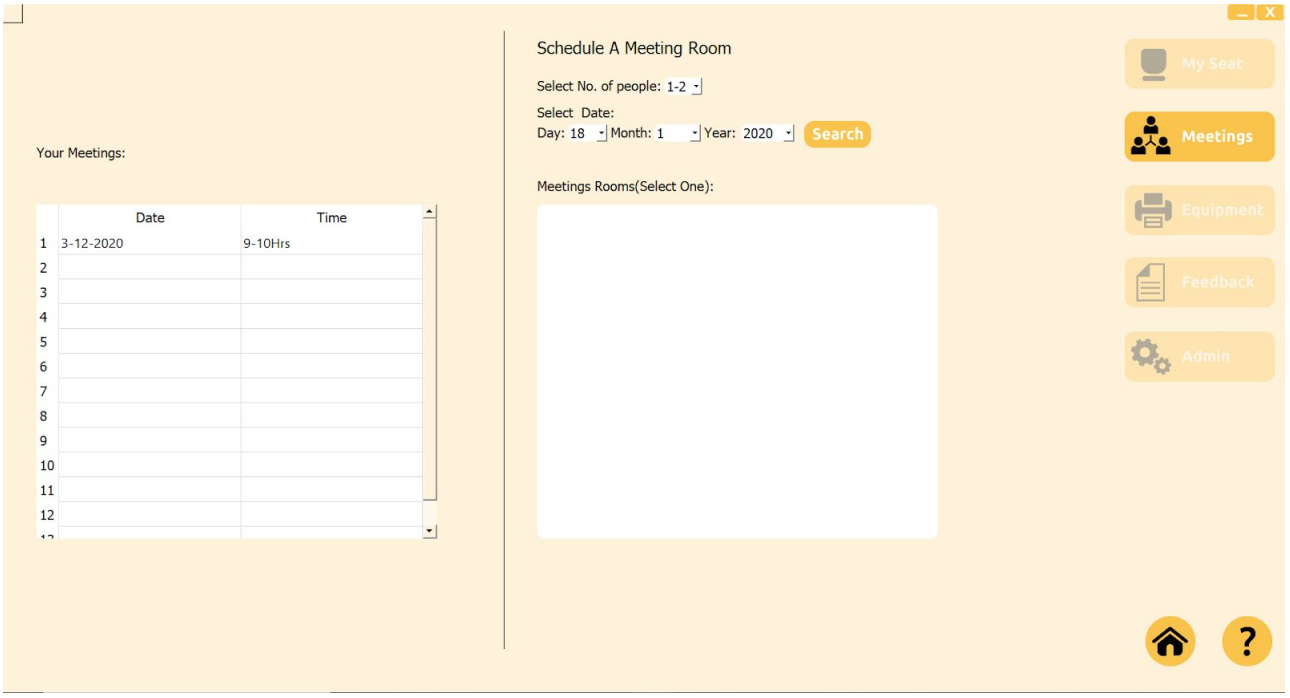
```
        self.imageComputer2.setPixmap(QPixmap("Data/Equip
  Help@2x.png").scaled(3840 / 2, 2080 / 2))
        self.imageComputer2.resize(0, 0)
        self.i=0
def singleClicks(self):
    value = self.equiplist.currentItem().text()
    id = self.getid(value)
    query = ("SELECT * FROM equipment WHERE id=?")
    person = cur.execute(query, (id,)).fetchone()
    self.nams.setText(person[0])
    self.location.setText(person[3])
    self.destext.setText(person[4])
    self.photo.setPixmap(QPixmap(person[2]).scaled(300, 300))

def getid(self, person):
    query = "SELECT id,name,location,photo,description FROM equipment"
    employees = cur.execute(query).fetchall()
    for employee in employees:
        if person == employee[1]:
            return employee[0]

def getEquipment(self):
```

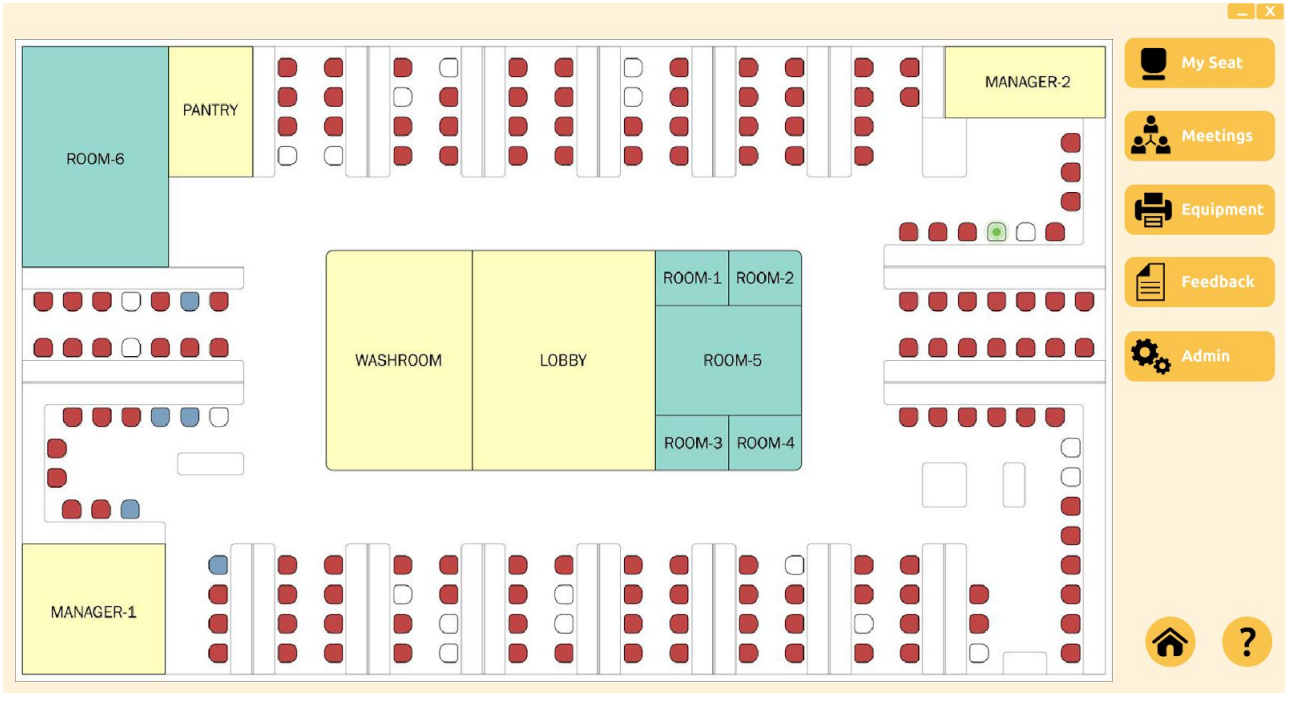
```
    query = "SELECT id,name,location,photo,description FROM equipment"
    employees = cur.execute(query).fetchall()
    for employee in employees:
        self.equiplist.addItem(employee[1])


def search(self):
    value = self.srch.text()
    count = self.equiplist.count()
    print(count)
    if value == "":
        QMessageBox.information(self, "Warning!!!", "Please type something to search")
    else:
        self.srch.setText("")

        query = (
            "SELECT id,name FROM equipment WHERE name LIKE ?")
        results = cur.execute(query, ('%' + value + '%',)).fetchall()
        print(results)

        if results == []:
            QMessageBox.information(self, "Warning", "There is no such Equipment")

        else:
```

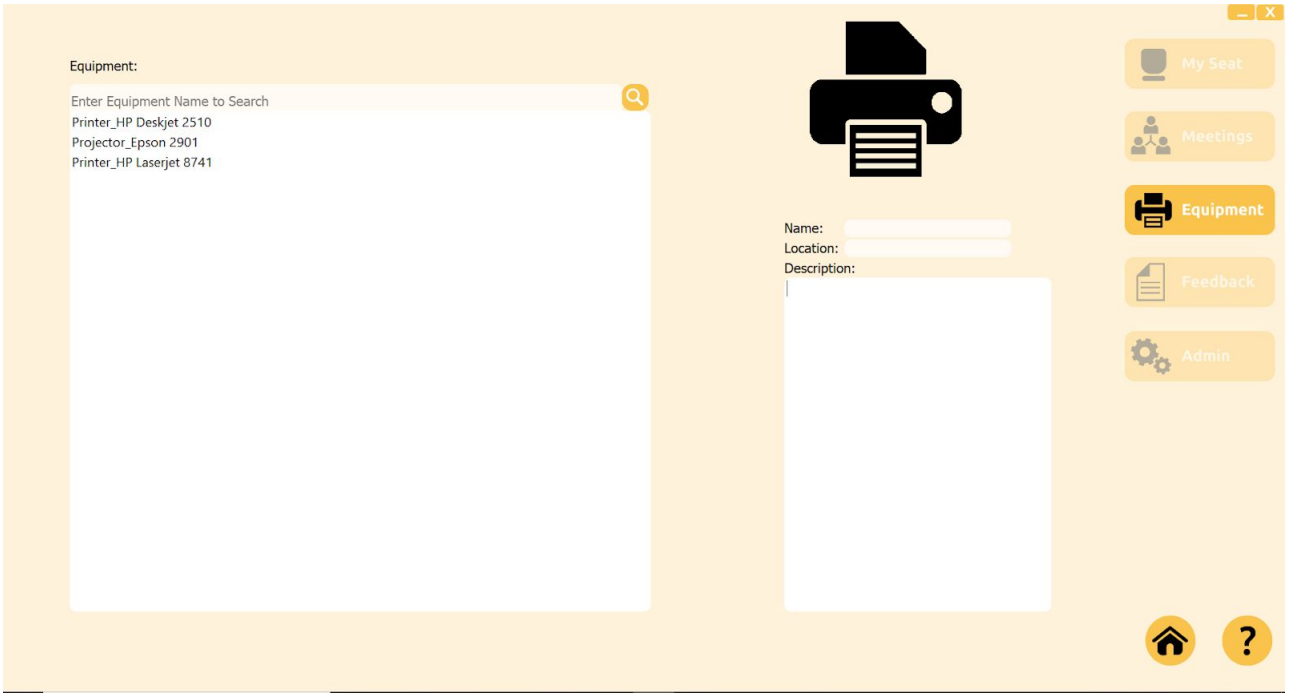
```
            self.equiplist.clear()
            for employee in results:
                self.equiplist.addItem(employee[1])


def book(self):
    self.book=Meetings.Meetings()
    self.close()
def feedbacks(self):
    self.feed=Feedback.Feedbacks()
    self.close()
def accnt(self):
    if data.type==1:
        self.accnt=Admin.admin()
        self.close()
    else:
        self.accnt=Account.Accnt()
        self.close()



def seats(self):
    self.seatts = date.date()
    self.close()
```

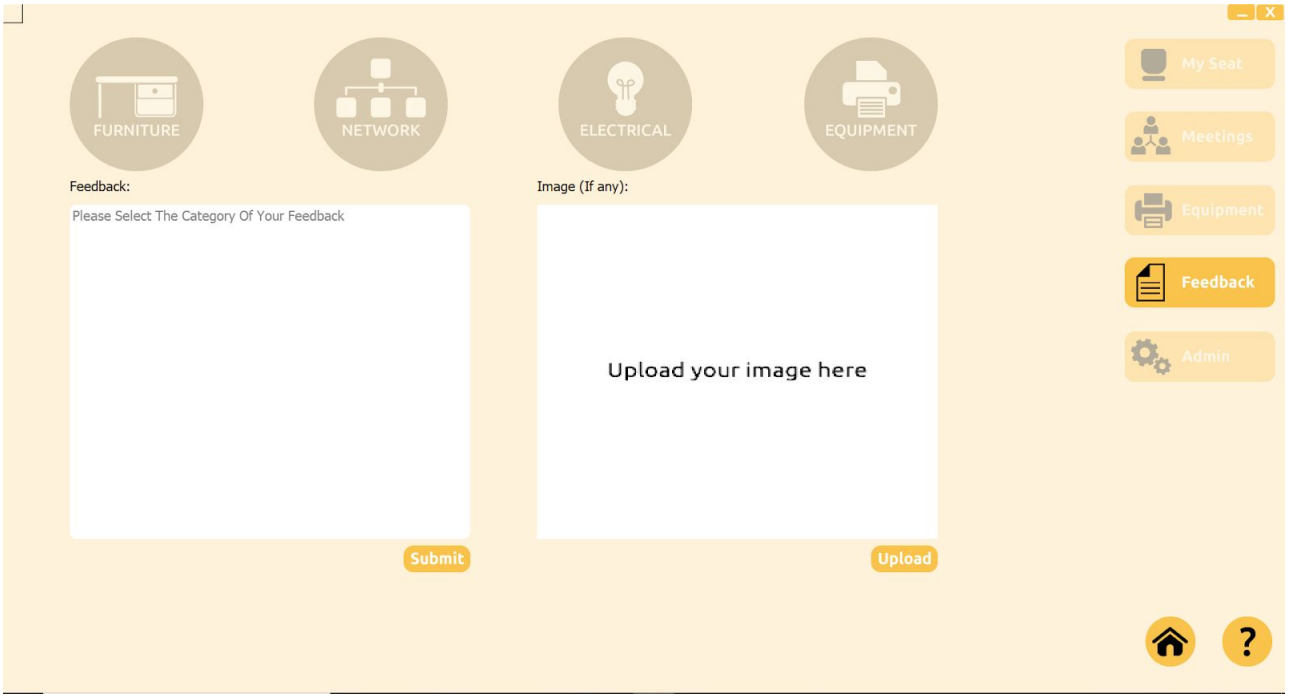
OUTPUT SCREENSHOTS

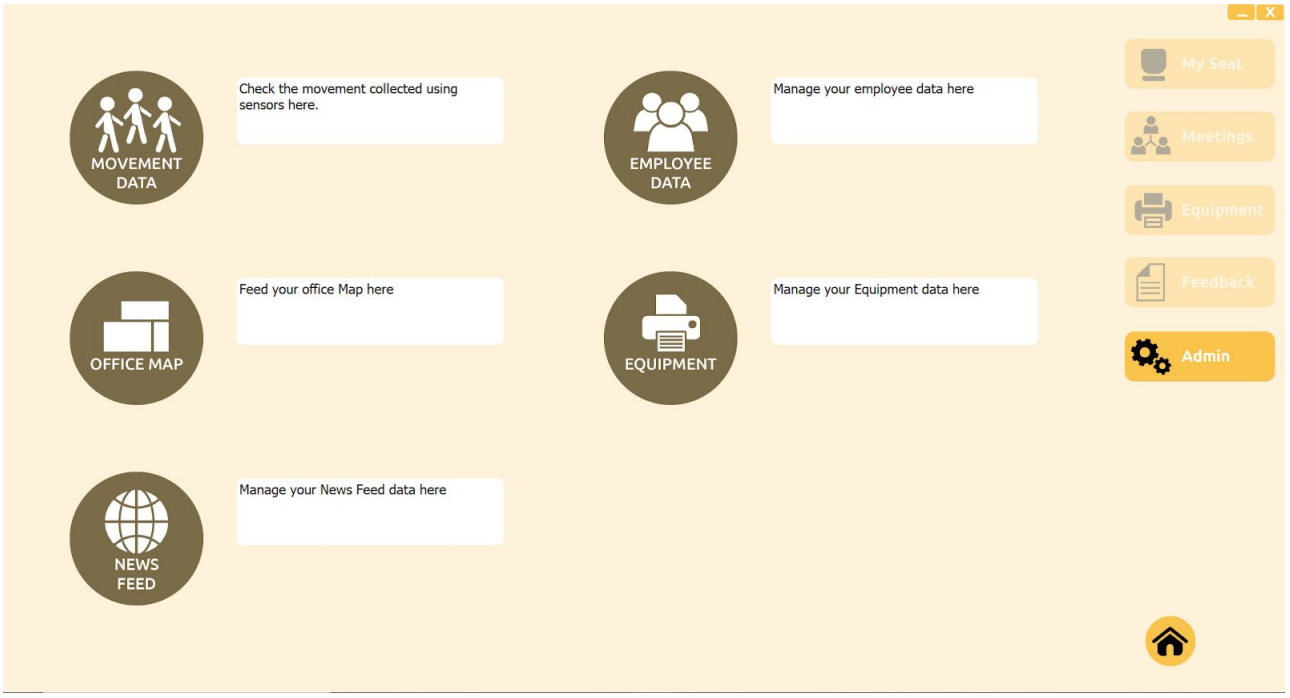

X
O-Man Solutions Pvt. Ltd.
Employee Id:
Enter Employee ID
Password
Enter Password
Login


Your Meetings:
Date
Time
1 3-12-2020 9-10Hrs
Schedule A Meeting Room
Select No. of people: 1-2
Select Date:
Day: 18 Month: 1 Year: 2020
Search
Meetings Rooms(Select One):
My Seat
Meetings
Equipment
Feedback
Admin


ROOM-6
PANTRY
MANAGER-2
WASHROOM
LOBBY
ROOM-1
ROOM-2
ROOM-5
ROOM-3
ROOM-4
MANAGER-1
My Seat
Meetings
Equipment
Feedback
Admin


Equipment:
Enter Equipment Name to Search
Printer_HP Deskjet 2510
Projector_Epson 2901
Printer_HP Laserjet 8741
Name:
Location:
Description:
My Seat
Meetings
Equipment
Feedback
Admin


FURNITURE
NETWORK
ELECTRICAL
EQUIPMENT
Feedback:
Please Select The Category Of Your Feedback
Submit
Image (If any):
Upload your image here
Upload
My Seat
Meetings
Equipment
Feedback
Admin


MOVEMENT DATA
Check the movement collected using sensors here.
EMPLOYEE DATA
Manage your employee data here
OFFICE MAP
Feed your office Map here
EQUIPMENT
Manage your Equipment data here
NEWS FEED
Manage your News Feed data here
My Seat
Meetings
Equipment
Feedback
Admin

CONCLUSION AND FUTURE ENHANCEMENT

According to a few surveys conducted, most of the MNC's still use outlook for booking things like meeting rooms manually , which is a bit outdated. With O-Man they can implement all office management functionalities in a single platform

This project makes the work of an office administrator easy and also helps streamline the working of a big office.

It also improves the efficiency of working of employees in an organization in a positive way.

Hereby we the students of BTECH 5th semester conclude that the project was completely and solely developed by the team and team members. All the information can be retrieved by the team members. We also conclude that this project has helped us gain more knowledge about the topic that we indulged ourselves into 'PyQt5 and Sqlite3'. We as a team would be glad to enhance and promote this project if given chance and help ourselves and the society in the near future.

Expansion to mobile platform and addition of more functionalities like attendance , leaves , salary can be added to improve it further more.

Each office has different needs, hence the functionalities can be easily tailored according to the needs of the office

REFERENCES

1.https://doc.qt.io/qtforpython/