Run the UPS Application:

Install the below required software:

- **MySQL Community Server version 8.0.21.**
- **Python version 2.7 (Prefered) or version 3.0. (Steps for running the functions in each case have been specified above)**
- **MySQL Workbench version 8.0.21 (Optional)**
- **For connecting python with MySql we need to install mysql python connector which can be installed using the below :**

```
pip install mysql-connector-python
```

Key Steps while Installation:

You need to specify a **root password** using strong password encryption while installing MySql Community Server. This root password must be used to connect to the database created using SQL Scripts and the Python scripts would use the same credentials to connect to the database. The username in each case would be **root.**

Load the Project Demo Data:

There are two ways to load demo data into the database,

1. SQL:
   a) Create tables into the database using commands in design_tables.sql file.
   b) Fill in the data into the tables using commands in fill_tables.sql

2. Python: In order to load the Demo Data, execute the below command:
   python demo_data.py
   In order to run using Python you need to edit the lines 862-866 as shown below:

```
if __name__ == '__main__':
db = mysql.connector.connect(host='localhost',
                    database='TEST_PROJECT',
                    user='root',
                    password='Rajson251710@')
```

The database name can be kept accordingly keeping in mind the same database has been created in SQL. The username and password must be kept the same that is kept during installation of MySQL Community Server.

Starting the program:

Execute *"python main_ui.py"* and then give the inputs as per the requirements(Detail on each option is provided in the **Menu Section mentioned in Report**)

Prompt Options on running menu_ui.py:

At first, it will ask for what action program is needed and select from below options

```
1. Ups_Admin
2. Employee
```

```
3. Student
4. Sample_query
5. Reporting_query
```

Opted for 1> ups_admin

Then, below are the actions can be performed by selecting corresponding input number

```
1. ExitLot
2. IssueCitation
3. PayCitation
4. AssignPermit
5. addLot
6. AssignZoneToLot
7. AssignTypetoSpace
8. CheckVValidParking
9. CheckNVValidParking
```

Opted for 2> Employee

Then, below are the actions can be performed by selecting corresponding input number

```
1. ChangeVehicleList
2. GetVisitorPermit
```

Opted for 3> Student

Then, below are the actions can be performed by selecting corresponding input number

```
1. GetVisitorPermit
```

Opted for 4> Sample_query

Then, below are the actions can be performed by selecting corresponding input number

```
1. Show the list of zones for each lot as tuple pairs (lot, zone).
2. Get permit information for a given employee with UnivID: 1006020
3. Get vehicle information for a particular UnivID: 1006003
4. Find an available space# for Visitor for an electric vehicle in a specific parking
lot: Justice Lot
5. Find any cars that are currently in violation
6. How many employees have permits for parking zone D
```

Opted for 5> Reporting_query

Then, below are the actions can be performed by selecting corresponding input number

1. For each lot, generate the total number of citations given in all zones in the lot for a three month period (07/01/2020 - 09/30/2020)
2. For Justice Lot , generate the number of visitor permits in a date range: 08/12/2020 -08/20/2020 , grouped by permit type e.g. regular, electric, handicapped.
3. For each visitor’s parking zone, show the total amount of revenue generated (including pending citation fines) for each day in August 2020