Technical Report of the Police Incident Management System

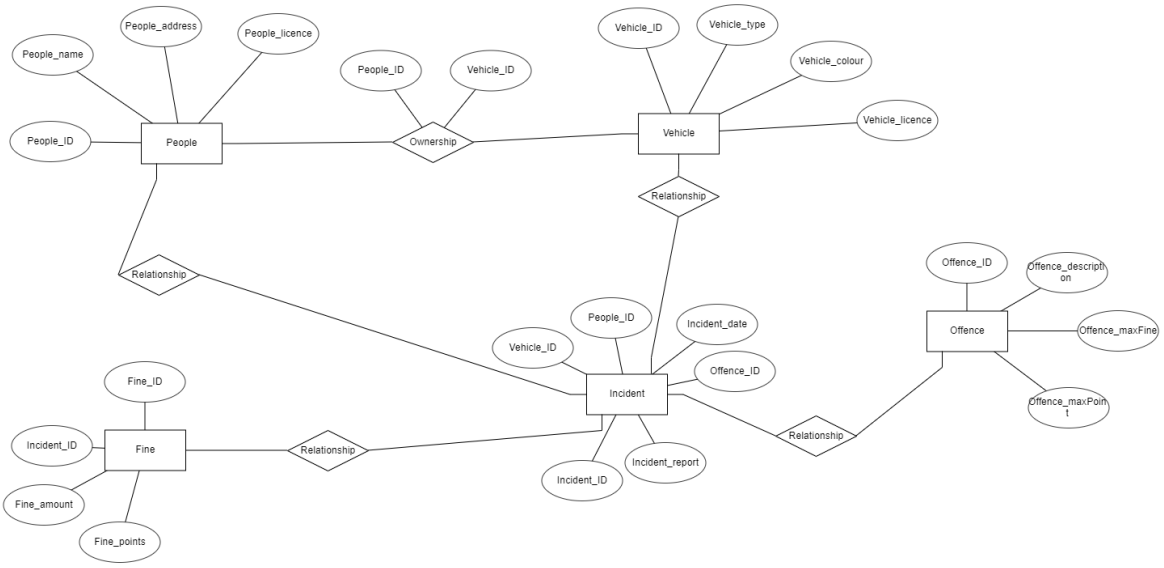
- Installation

  When unzipping the “psxsc15-20316799_InstallationFiles.zip”, users should be able to see the MySql script. Use Mysql workbench to run the script in order to input the data of the system.

  The second step is to put the php flies on the server. Users will see a folder “project”, which includes all the php files needed in the system. In the folder, I classify them into 2 files. The first type is the view files(files that include html tags), and the other type is pure php files, which are contained in the project/include folder. Put the whole project folder on the folder of the server users use. The exact position varies depending on the system each user uses. For example, users should put the files in the “htdocs” folder when using XAMPP.

  After putting the files on the server, users should modify the database connection information according to the servers users use. There is a database.php file in the project/include file. Users will see there are 4 variables $dbHOST, $dbUSER, $dbPASS, $dbNAME. Users should modify these variables according to their own systems.

- Database Design



According to the brief ERdiagram, users can see that there are 5 main entities in this database, which are people, vehicle, fine, incident, and offense. Because the relationship between vehicles and people is a many-to-many relationship, thus there is an ownership table mapping the relationship of people and vehicle. The incident entity records the people and vehicles involved, incident report, and offense_ID so that users can figure out the information of the incident information (need to join other tables in order to see the detailed information). The offense table records different

types of offenses and the specification of each offense. The fine table records the final fine information of each incident.

There is another table not shown in the ERdiagram called “users”. This table is used to record the user accounts that are allowed to login to the system. There are 2 types of users, which are normal users (police officers) and an administrator(Daniels, copper99) of the system.

- Database function queries

1. The first function is the login function. In this function, the php code should check the information typed in by users in the database. The query "select * from users where username= '$username' && password='$password'" is used to search the user information in the database. If the users exist and the password is correct, then the code should tell whether the user is a normal user or an administrator, and lead them to different pages. If no data exists in the database about this user, there will be an error notice and will ask the user to type in login information again.
   Files: index.php, login.php, login-inc.php

2. There are 2 queries used in this function. The first query "SELECT * FROM People WHERE People_name LIKE '%$searchname%'" is used to search people’s names in the database. The search is case-insensitive, and the result will include all the data that contains the search pattern typed in by the user.
   The second query "SELECT * FROM People WHERE People_licence LIKE '$searchlicence'" works in a similar way except it searches the people’s licence instead of people’s name.
   If no data exists in the database, then the system will give an error notice using javascript and ask the user to search again.
   Files: people_lookup.php, people_name_lookup-inc.php, people_licence_lookup-inc.php

3. The third function is to look up the vehicle information. The query used to implement this function is slightly complex because it involves the owner information, and thus needs to join the ownership table and the owner table in order to list these data. The query "SELECT * FROM Ownership
   inner join Vehicle on Ownership.Vehicle_ID=Vehicle.Vehicle_ID
   inner join People on Ownership.People_ID=People.People_ID
   where Vehicle_licence LIKE '$vlicence'" first join the vehicle and ownership table using vehicle_ID, and then join the people and ownership table using people_ID. Therefore, in the final joined table, users can see the relationship of vehicles and people.
   Files: vehicle_lookup.php, vehicle_lookup-inc.php

4. The fourth function is to add new vehicle data. Before allowing to add new data into the database, the php code will use"SELECT * FROM People WHERE People_name ='$owner'" to check whether the owner data is present. If no personal data exists yet, then use the query "INSERT INTO People (People_name, People_address, People_licence) VALUES ('$name', '$address', '$licence')" to insert the personal data first. If the personal data is present, then the php code will check whether the vehicle

data is already in the database. If this vehicle data is not yet in the database, then use the search query "SELECT * from Vehicle where Vehicle_licence='$vlicence'" and "SELECT * FROM People where People_name LIKE '$owner'" to search the vehicle id and people id. At the end, insert the 2 IDs searched above into the last query "INSERT INTO Ownership (People_ID, Vehicle_ID) VALUES ('$pid', '$vid')".
Files: add_vehicle.php, add_vehicle-inc.php, add_people.php, add_people-inc.php

5. The fifth function is to add, retrieve and edit the incident content. First of all, like the function above, before inserting the incident, this function needs to check the existence of people and vehicle data and deal with the absence situation. Then use the query "INSERT INTO Incident (Vehicle_ID, People_ID, Incident_Date, Incident_Report, Offence_ID) VALUES ('$vid', '$pid', '$date', '$report', '$oid');" to insert the data into the table.
   The second step is to retrieve the existent incident report. Use the query "SELECT * from Incident where Incident_ID='$incidentid'" to check and retrieve the report. If users want to edit the report, then use the "UPDATE Incident SET Incident_Report = '$editreport' WHERE Incident.Incident_ID = $incidentid"; to update the report content in the incident table.
   Files: add_people2.php, add_people3.php, add_vehicle2.php, retrieve_report.php, retrieve_report_display.php, add_people2-inc.php, add_people3-inc.php, add_vehicle-inc.php, add_incident.php, add_incident-inc.php.
6. The last function is to allow the administrator of the system to add new user (police officer) accounts in the system and add new data into the fine table. The first step is checking whether the password and confirmation password typed in by the user match with each other. Then check whether the user is already in the database. Then implement the query "insert into users (username,password) values ('$username',$password)".
   In order to add fine data into the fine table, the php code needs to check whether the incident is in the database, and then check whether this incident already has fine data. If not, use the query "INSERT INTO Fines (Fine_Amount, Fine_Points, Incident_ID) VALUES ('$fineamount', '$finepoints', '$incidentid');" to insert data into fine table.
   Files: add_fines.php, add_fines-inc.php, update_account.php, update_account-inc.php