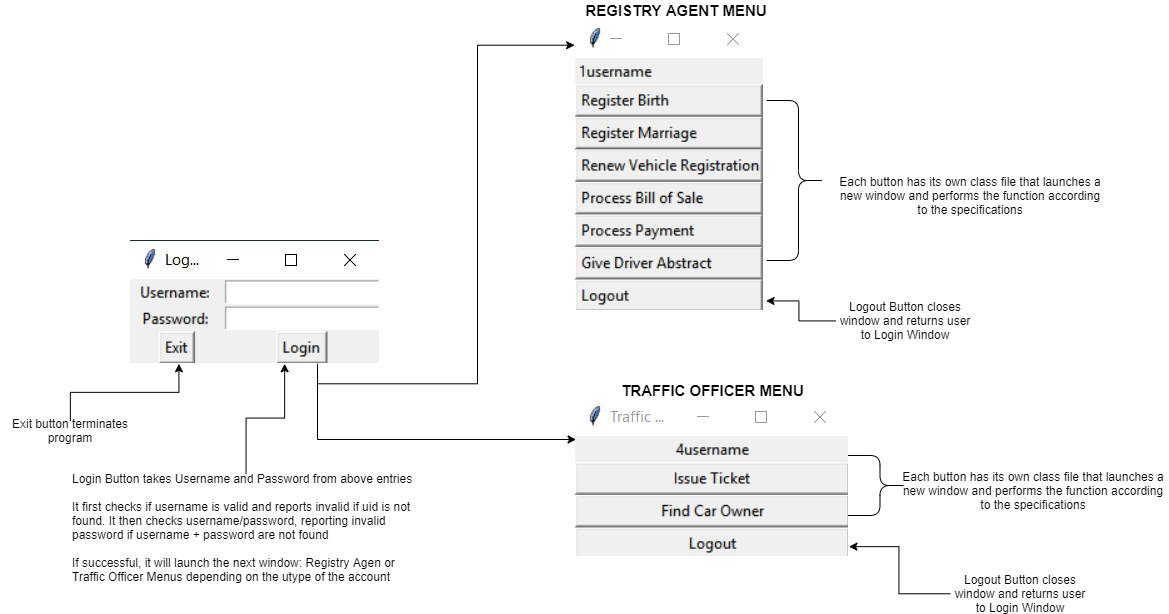
**DESIGN DOCUMENT**

Created by: Jian Xian, Marshall Chang, Tina Do

**GENERAL OVERVIEW OF OUR SYSTEM:**

****Our program is written in python3 using the built-in modules tkinter for GUI and sqlite3. It is launched by running Launcher.py with the database path as the first parameter.

**DESIGN OF SOFTWARE:**

Our software is divided into 3 types of classes:

1. Launcher:

Main class which can be called in command line to start the program. It contains the command line argument: “database path” which passes the database of the system to the program. Launches the Login Window Class.

2. Independent Window Classes:

Each function of either a registry agent or traffic officer has its own Tkinter classes which are independent of each other. That is, a window does not require another to run. This makes separate development and testing easier. Windows may destroy themselves and launch others as necessary, while passing through needed information such as the name of the user, and the database to be modified. Each window app contains GUI and methods to handle GUI actions as well as a “Cancel” or “exit” button that returns the user back to the menu according to their uid.

3. SQL Controller:

A class that imports sqlite3 module and has methods for all SQL queries used in the program. Passed between windows.

**OUR TESTING STRATEGY:**

To test the program, we will each test our own work for functionalities (Performs specifications properly) as well as quality assurance (ensure that the program does not break from simple user mistakes). Then we will also perform the same test process on another group member’s work.

Functionality Tests will check all valid inputs for each specification, and ensure they produce a correct or reasonable output.  
  
Quality Assurance Tests (QA) will check for possible user errors such as typos, blank entries, incorrect types, and ensure these produce a reasonable error output and do not crash the program.

**GROUP WORK BREAK-DOWN AND DETAILS**

**Group Meetings:**

**Meeting1 10/16 (1h)**

-Discuss possible strategies for program design

**Meeting2: 10/25 (2h)**

-Confirm program design

-Divide remaining work

-Specify planned progress/deadlines

**Meeting3: 10/30 (1h)**

-Review testing strategy

-Divide testing work

**Software Development- Includes Design, Development, and self-QA time of functions**

**Marshal:**

-Overall Design and Structure (2h)

-Login, Launcher (0.5h)

-SQLController Outline (0.5h)

-Register a Birth (1h)

-Process a payment (0.5h)

**jianxian:**

-Register a Marriage (2h)

-Renew a Vehicle Registration (0.5h)

-Get a Driver Abstract (4h)

**Tina:**

-Create Test Database (1.5h)

-Process a Bill of Sale (1.5h)

-Issue a Ticket (2-3h)

-Find a Car Owner (3h)

**QUALITY ASSURANCE TESTING:**

**Tina updating code per Marshall’s Quality Assurance Testing: ~1h**

* corrected unique constraint failures and handling of crashes, ensured user input is in correct date format, adjusted case sensitivity of find car owners, made sure log out button re-directed user to the traffic officer window

**Tina testing Marshall: ~1h**

* Tested “Register Birth” function: ~30min
* Tested “Process Payment” function: ~30 min
* checked for correct date formats, blank entry inputs, updating new data to database correctly, function performs as per the specification and clarifications, updating new data to database correctly, window displays correct title

**Tina testing jianxian: ~1.5h**

* Tested “Register Marriage” function: ~30min
* Tested “Renew Vehicle Registration” function: ~30 min
* Tested “Driver Abstract” function~30min
* checked for valid date inputs, review of SQL queries to perform as intended, ensured SQL queries returned correct result and displayed accordingly, found program crashes