Michelle Aubin
Ryan Kang
Daniel Han

Mini Project 1 Design Document

General Overview

Our system, *Service Alberta*, provides many services to users and keeps enterprise data in a database. Our system uses a simple command line interface. The user is greeted by a welcome message and is given the choice to either login or exit. If they do not enter a valid choice, the system continues to prompt them. If they choose to log in, the system moves to the login screen and prompts the user to enter their user id and password. If the user is not in the system or has entered the wrong credentials, the system will display an "Invalid Credentials" message and continue to ask for their user ID and password. If the user wishes to exit the application while they are at the login screen, they can enter "exit".

After a successful login, the user enters a menu of operations with actions specific to their role where they can choose an action. They can enter "exit" to exit the application or "logout" to logout and return to the welcome screen. If the user enters anything other than these options, they are prompted to enter a valid option. After selecting an operation, they can enter "quit" at any time to quit the operation and return to the menu of operations screen. The menu of operations calls the function associated with the operation. Upon completion of an operation, the user returns to the menu of operations screen.

Detailed Design

Michelle Aubin (Functionalities: agent 1-3 and agent menu)

***The agent can always return to the main menu by typing "quit" (This is stated in the second statement when any of the functions are called).**

Function "agent_menu": Menu of operations for agents

This function displays the operations that an agent can perform and prompts the user to choose an operation. The user can also choose to exit or logout. Depending on the choice, this function calls the function responsible for the chosen operation, returns to the home screen function, or it exits the program.

Function "a1": Register a birth

The function prompts the user for the first name, last name, gender, birth date, the birthplace of the newborn, as well as the first and last names of the parents. It checks that each input is valid before moving on to the next prompt. If any of the parents is not in the database, the function prompts the user for the parent's birth date, birth place, address and phone (these values may be NULL). The function inserts the newborn into the database as an entry in the "persons" table, and does the same for any of the

parents that were not already in the database. It also generates a unique registration number for the birth record, sets the registration date to the current date, sets the registration place to the city of the user, then inserts the birth record into the database as an entry in the “births” table. Then, the function prints a success message and returns to the agent_menu function.

Function “a2”: Register a marriage

The function prompts the user for the first and last names of each partner. If any of the partners is not found in the database, the user is prompted for the partner’s birth date, birth place, address and phone (these values may be NULL). The function checks that each input is valid before moving on to the next prompt. It inserts into the database any of the partners that were not already in the database as entries in the “persons” table. It also generates a unique registration number for the marriage record, sets the registration date to the current date, sets the registration place to the city of the user, then inserts the marriage record into the database as an entry in the “marriages” table. Then, the function prints a success message and returns to the agent_menu function.

Function “a3”: Renew a vehicle registration

The function prompts the user for a registration number. If that input is invalid or the reg number is not in the system, the function displays an appropriate error message and continues to prompt the user until it receives a valid registration number. Then, it selects the registration record from the database and updates the expiry date to one year from today’s date if the registration has expired or expires today. If the registration has not expired yet, it updates the expiry date to one year after the current expiry date. Then, the function prints a success message and returns to the agent_menu function.

<u>Daniel Han (Functionalities: 4-6)</u>

***The agent can always return to the main menu by typing “quit” (This is stated in the second statement when any of the functions are called).**

Function “a4”: Processing a Bill of Sale

This function provides the agent a method to keep track of sales between two registered “persons” in the DB. The agent is prompted to enter the VIN, full name of the seller and buyer, and the plate of the car. The VIN must be valid (in the DB) and will be continually prompted. If any of the other variable inputs are invalid, the system will display an error message. The previous owner will be kept in the DB with an updated expiry date (current date). The new owner of the car (buyer) will be inserted into the DB

with unique regno, expiry date (1 year from today) and appropriate information.

Function “a5”: Processing a Payment for a Fine

The function provides a system of payment tracking accessible directly to the user. The user is prompted to enter a ticket number. This ticket number must be in the DB and be numeric. The system retrieves and displays the fine amount and the remaining amount. The user is prompted to enter an amount to pay. A user is only allowed to pay an amount for a ticket once per day regardless of if the ticket is completely paid off or not. If the ticket has no remaining dues, the user is automatically returned to the main menu.

Function “a6”: Retrieving a Driver’s Abstract

The function prompts the agent with a person’s full name and returns a table including the total number of tickets, demerit points and demerit notices a driver has received in the past 2 years and their lifetime. If a name that is not registered in the DB is entered, the full driver’s abstract is displayed with “0”s and “None”. The agent is also provided with an option to display individual tickets of the individual with a limit of 5 tickets per page.

<u>Ryan Kang (Functionalities: Officers)</u>

***The agent can always return to the main menu by typing “quit” (This is stated in the second statement when any of the functions are called).**

Function “o1”: Issue a Ticket

The user will enter a valid vehicle registration number. This will produce the make, model, year, and color of the respective car. The user can then create a new ticket for that car by entering a violation date, violation text, and fine amount. The violation date is set to today’s date if not entered. A unique ticket number is assigned by finding the first available integer in ticket number. The fine amount can only be an integer value greater than 0. This information is then inserted into the database.

Function “o2”:

The user can identify a vehicle by one or more attributes: make, model, year, color, and plate. If no attribute is entered the user is prompted again. Year must be an integer greater than 0 with 4 digits. A string is created with these attributes where blank attributes are not concatenated and non-blank are concatenated. This string is then queried in the WHERE clause to produce all matching vehicles. Vehicles which are not registered are given prompts in their registration attributes. If there are 4 or more matching vehicles the program prints all of them and asks the user to choose one. The

program then prints the latest information of each care by ordering by registration date descending and returning the most recent one.

Testing Strategy: Performed by all members

Necessary testing that applies to each function: Case insensitivity, Whether inputs are valid (if they adhere to the constraints for each value), testing with values that are in the database and values that are not, catching inputs that don't match the type required and making sure that each function works as intended (basic functionality).

Group Work Break-down

The breakdown of the work was divided into three sections: functions "a1" to "a3", functions "a4" to "a6", and functions "o1" and "o2". The individuals responsible for finishing these function sets were Michelle, Daniel and Ryan respectively. Michelle took on the extra role of creating the "agent menu" function as well as additional sub-functions for organization and efficiency. Ryan took on the additional role of creating the home screen and login functions and SQLinjection prevention. All group members contributed to debugging and testing for the entire program. Version control was established through GitHub. Communication was done in-person as well as through the usage of group chats.

Group Meetings (Total): ~8 hours
Michelle:

- Individual work: ~5-6 hours
- Progress made: Completed implementation and debugging for functions a1, a2, a3 and agent menu.

Daniel:

- Individual work: ~5-6 hours
- Progress made: Completed implementation and debugging for functions "a4"-"a6".

Ryan:

- Time spent: ~5-6 hours
- Progress made: Completed implementation and debugging for function "o1", "o2", home screen, login screen, and SQLinjection prevention.