CMPUT 291 MINI PROJECT 1 DESIGN DOCUMENT

Fall 2015

Professor: Davood Rafiei

LEC Section: A1

Group Member: Alan Zhang

CCID: 1295834

LAB Section: D06

Group Member: Imran Ali

CCID: 1447593

LAB Section: D02

The program functions are separated at the start of the interface function after obtaining user email and password. Instead of python classes, it was better to utilize functions because we found that passing down values was more efficient in this instance, where we don’t have to call the cursor and connection every time we made a new function. Four functions are made instead of 5 (interface, search, book, and existing bookings) because we noticed that the logout was better made at the interface (def interface()) so that every time the user wanted to log out, they are given the choices at the main menu. A simple breakdown of our design can be seen:

|  |
| --- |
|  |

Interface

-contains the

logout

|  |
| --- |
| Search |

|  |
| --- |
|  |

Bookings

-can delete bookings

This project is developed with Python and is made to run in terminal. The module cx\_Oracle is imported at the start of the program to link the program and database in SQL. At initial boot-up the program, it prompts the user to enter UNIX username, and Oracle password in order to log into SQLPLUS. It tests the connection with a try, except and else statement. If successful in connecting to Oracle, the except statement is ignored and the program is ran in the interface function. It asks the user to input whether the individual is registered or not. If not registered, it will give the option to make a new user email and password in table users. Error handling for these inputs include the email string not being greater than a length of twenty and password not greater than a length of four as specified by the constraints for user table. Most inputted information in the program are handled the same way in regards to table value constraints. The program calls the time module to obtain a system login date in the format DD/MM/YYYY. This is later changed at logout through an update statement, where the Oracle execution cursor and connection are then closed. If not logout, the user is asked whether they want to search for flights, or list existing bookings. To make a booking the user must first search for flights and if it exists, then they are prompted to enter a name and then a country later down the search function. To delete the booking the user must first list their existing bookings and delete from that option in main menu. If the individual chooses to search for flights, we first try to drop the views available\_flights, and good\_connections if it exists, because they are made later on in the function. The user is asked a source airport and with martial matching, the query is ran through SQLPLUS whether the user inputted a name that is like an airport or a name that is like a city in which the airport resides (found in the table airports). The same is done in the destination search. Once inputted, the user is asked to input a date, and this is executed where a final flight table is sent to standard output with the available\_flights view and good\_connections view. The user is prompted to input whether they would like to book a flight from the table provided. If not they are directed back to the main menu in interface, given the option to log out. If user decides to book, they are prompted to input a name in the passenger table and if the pass-by-value email is in the passenger table. A random 3-digit ticket number is generated by the built in random module in python when booking is ready. This is inserted into the bookings table. Information regarding the booking is printed to standard output whether it was successful or not. Bookings can then be listed by the existing bookings function and can be deleted from there by simple delete SQL queries.

For testing purposes, we realized that it is possible for the user to accidentally miss click the enter bar without inputting a string for each input option, therefore we made sure , the individual is placed back at the original loop with “if input == ‘’ “, rather than crashing. Every time a string is inputted, it is lowered and/or uppered case as required by the table value constraints. The length is also checked whether the input is not greater than the specified table value constraints. If errors occur because of queries or oracle related issues, they are mostly handled by cx\_Oracle.DatabaseError as an exception (this was specified and found in the sample CreateToffees.py found in the labs). This will print the Oracle error code and Oracle message as specified by the system as standard error. Some major Oracle related errors that are found include: ORA-00904: invalid identifier (realized that in order to modify the dates in a view, we need to use to\_char(variable, ‘HH24:MI’) at the beginning of create view), ORA-01036: illegal variable name/number (by not adding a colon(s) and single quotations to our queries), ORA-01400 (by not changing a variable name when running a for loop to print information), and not typing curs.commit() in order to insert values into table. For source and destination airport search, partial matching is used by executing an SQL statement and seeing if it is somewhat similar or matches the name or city to obtain the acode in airports.

Our group work breakdown was that Alan Zhang would handle the interface, logout, partial start to the booking function, error handling, and the design document, whereas Imran Ali would handle the search function, booking function, error handling and interface. Both individuals believe the search and booking functions of the program (question 1 and 2 of the assignment) would require the most work therefore a majority of the time (80%, roughly 25 hours) is spent on building these two functions, at csc second floor computer or through Skype at home together. The other functions (interface, logout, listing bookings, and deleting bookings) were finished in approximately 12 hours. The work distribution is distributed (45% for Alan, and 55% for Imran) because the most credit is given to building the booking search function that links to booking. Both have spent roughly the same amount of time building these two functions, but Imran Ali deserves more credit for getting it to finally work.