Case Study: MyTrip – Trip Planner

Group 5

Team member Emp ID	Team member Name	Team Member Email Id
46289283	**Himanshi Garg**	**himanshi.a.garg@capgemini.com**
46283116	Monalisha Sahu	monalisha.sahu@capgemini.com
46283115	Lakshmi Kumari	lakshmi.b.kumari@capgemini.com
46282717	Shilpa Gantayat	shilpa.gantayat@capgemini.com
46289284	abantika saha roy	abantika.saha-roy@capgemini.com
46283117	shivangi kumari	shivangi.a.kumari@capgemini.com

PROBLEM STATEMENT

1.2 OBJECTIVE

To create MyTrip system for the users where they can book transport and hotels for the selected holiday destination. The application is to be develope as executable file compiled on Linux. Information regarding Destinations, Hotels, ModeOfTransport, Tickets, Users, HotelBooking, Trips should be maintained by MyTrip is comma separated txt files.

1.3 ABSTRACT OF THE PROJECT

1. User should be able to choose a holiday destination.
2. Once selected show them different modes of transports and give them option to choose a particular way to reach.
3. Transport should contain at least two modes to reach the destination (E.g.: Train-Cab, Flight-Bus, etc.)
4. Simple reservation for transport booking. Should generate tickets.
1. Hotel Booking with some filters and sort logic.
2. User should have an option to skip hotel booking.
3. Show them total cost for the trip and checkout gateway to confirm their trip.
4. User should be able to check his planned trip whenever he wants.
5. User should be able to check his transport details when he enters ticket number.
6. Handle data and errors properly. Show appropriate messages to user.
7. Display good input, output messages and reports in proper format.
8. Security features should be implemented wherever possible. For example user passwords can be stored in encrytped format.

1.4 FUNCTIONAL COMPONENTS OF THE PROJECT

Following is a list of functionalities of the system. Wherever, the description of functionality is not adequate; you can make appropriate assumptions and proceed.

1. When MyTrip starts it displays Following Screen -

----------Login Screen----------

1. Register new user
2. Login
0. Quit

System should maintain comma separated users.txt where each line stores username, password and all other information about registered user. Be default this file contents one line – admin,admin,..... – denoting admin user.

Whenever username and password is entered, system authenticates it with entry in "users.txt" file.

- If match is found and if admin user then "Admin Screen" is displayed.
- If match is found and if not admin user then "User Screen" is displayed.
- It match is not found then message "Invalid User or password" is displayed and system exits.

2. When admin user login My Trip starts it displays "Admin Screen"

----------Admin Screen----------

1. Manage Destinations
2. Manage Hotels
3. Manage Transport Modes
4. Check Hotel Bookings
5. Check Tickets
0. Quit

Enter your option : <option>

option = 1 (Manage Destinations)

This further displays sub-menu -

----------Destination Screen----------

1. Add New Destination
2. Modify Destination
3. Delete Destination
0. Quit

Write all information regarding destinations in "destinations.txt" file where each line corresponds to one destination. Information about destinations to be stored as comma separated fields. MyTrip should ask for at least two modes of transport to

reach destination.

option = 2 (Manage Hotels)

This further displays sub-menu -

----------Hotel Screen----------

1. Add New Hotel
2. Modify Hotel
3. Delete Hotel
0. Quit

Write all information regarding hotels in “hotels.txt” file where each line corresponds to one hotel. Information about hotels to be stored as comma separated fields. MyTrip should ask for fare of stay in hotel for one day and store in “hotels.txt”.

Option = 3 (Manage Transport Modes)

This further displays sub-menu -

----------Transport Modes Screen----------

1. Add New Transport Mode
2. Modify Transport Mode
3. Delete Transport Modes
0. Quit

Write all information regarding destinations in “transportModes.txt” file where each line corresponds to one transport mode. Information about transport modes to be stored as comma separated fields. These transport modes to be displayed while adding destination.

Option = 4 (Check Hotel Bookings)

This further displays sub-menu -

---------- Hotel Bookings Screen----------

1. Check Bookings at a hotel
2. Display Bookings in Date Range
-- add more filters on hotel bookings here --
0. Quit

When option = 1 – MyTrip should ask hotel name and display its bookings for today.

When option = 2 – MyTrip should ask <from date> and <to date> and display its bookings within given date range.

You can add more filters on bookings data. Have at least 10 filters in above menu.

Option = 4 (Check Tickets)

This further displays sub-menu -

---------- Tickets Screen----------

1. Check Tickets of users
2. Display Tickets booked on particular date
-- add more filters on tickets here --
0. Quit

When option = 1 – MyTrip should ask user name and display its his/her tickets.
When option = 2 – MyTrip should ask date and display its tickets generated on that date

You can add more filters on tickets data. Have at least 10 filters in above menu.

3. When normal user login MyTrip displays "User Screen"

----------User Screen----------

1. Plan Trip
2. Book Hotels
3. Book Tickets
4. Check Status
0. Quit

Enter your option : <option>

option = 1 (Plan Trip)
This option helps user to plan trip using destination, hotel, transport data.
Destinations are displayed to user, he/she selects one destination. MyTrip asks what dates user want to travel, number of persons travelling, names and ages of persons..
All information about trip is stored in comma separated file "<username>_trip.txt".

option = 2 (Book Hotels)
Trip information is displayed to user from "< username>_trip.txt" and "hotels.txt" file. User can choose hotels.
When user confirms hotel bookings this information is written to "bookings.txt" file.

option = 3 (Book Tickets)
Trip information is displayed to user from "< username>_trip.txt" file and "transportModes.txt" file.User can choose transport.
When user confirms transport tickets are created for every person in the trip. Tickets are stored in "tickets.txt" file.

option = 4 (Check Status)
Various status of the trip should be maintained in "< username>_trip.txt" file and displayed to user.

Assumptions:

Technical Requirements -

1) C programming language
2) Use file input/output operations to read and write data.
3) Use multiple Linked Lists to read data from files at the beginning and write updated data to files before applicaton ends.
4) Use dynamic memory allocation.

Non Functional Requirements

1) Multi-file multi-directory solution is expected. Modular and maintainable code (comments) and all coding standards should be followed.
2) makefile to build application. Two-step compilation process - .o and then executable should be generated.
3) Use valgrind tool on application executable to detect memory leak. Final valgrind report to be submitted in "reports" directory.
4) Level 0 DFD (context diagram), Level 1 DFD, Flow diagram and pseudocode for 2 complex functions logic.
5) SRS in pdf format, RTM, Plan, Presentation. MOMs
6) HLD_LLD Document (optional)
7) Design review, Code review, Inspection Logs of design and code reviews
8) Unit test cases and Integration test cases in UT_IT document. Both types of test cases i.e. sunny and rainy should be present in this document

Set Up Checklist for Project

Software Requirement:

Vi Editor, ctags, splint, valgrind, gcc, make, git account

Minimum System / Hardware Requirements:

Laptop with access to internet and Linux OS