

**CIS020-1 – Introduction to Software Development**

**CIS093-1 – Mathematics and Concepts for Computational Thinking**

**Assignment 2 – Group / Individual Project – Case Study**

**(Taxi Booking System)**

**Student Name: Nitesh Hamal - University ID: 2126474**

Table of Contents

List of Group Members (if Group Work)………………………………………………………5

Introduction / Overview………………………………………………………………………… 5

Task Description………………………………………………………………………………... 5

Project Plan / Schedule….

Tasks:

Requirements Analysis

Overview of Functional, Technical (Non-Functional) and Usability Requirements

Design:

* UML Diagrams
* Use Case Diagrams
  + Use Case Specifications/Description
  + Use Case Scenario
* Activity Diagrams
* Class Diagram
* Database Design
* Logical Database Design
* Entity Relationship Model
  + ERM Diagram
  + List of Entities
* Physical Database Design (including Data Dictionary)
  + Skeleton Tables
  + Data Dictionary
* User Interface Design

Implementation

Testing

List of Group Members – Roles and Responsibilities (if Group Work)

Discussion / Reflection / Critical Analysis

Conclusion

References

Appendix

* Complete code with comments and class / filenames.

You will need to complete the Table of Contents and Page Nos manually

PLEASE NOTE THAT THIS IS A PROFESSIONAL REPORT

ALL SCREENSHOTS, DIAGRAMS, etc SHOULD BE GIVEN A FIGURE NUMBER, A CAPTION AND AN EXPLANATION

CIS020-1 – Introduction to Software Development - 2021-2022

CIS093-1 – Mathematics and Concepts for Computational Thinking – 2021-2022  
Assignment 2 – Group or Individual Project – Case Study (Taxi Booking System)

University ID and Name

University ID 2126474 Full name Nitesh Hamal

Introduction/Overview

The university assigned student to create a taxi booking system. It is critical to undertake a requirement analysis before beginning implementation. The taxi booking system has three users: customer, driver, and administrator. Prior to development, a Class Diagram, Use Case Diagram, and Activity Diagram are created to visualize the project's operating mechanism. An Entity-Relationship Model is used to construct a Relational Database Management System (RDMS) for the database (ERM). The taxi booking system is built using the PyCharm IDE. The Python Libraries provide capabilities for constructing Python programs, and the Object-Oriented Programming idea is employed in development. This program is built with Tkinter, a Python GUI toolkit. MySQL is used in the database to store data from the taxi booking system. Completing a taxi booking system is a difficult process. A difficulty is unavoidable. Solving such challenges and creating a fantastic application is possible with the assistance of teachers, friends, and the internet.

Task Description

The task is about a taxi company's taxi booking system. A taxi firm would want to provide an online service for the convenience of individuals traveling about, as well as for drivers to monitor their forthcoming journeys.

To access the taxi booking system, the customer must first register with the system. A person must enter information such as their name, address, phone number, email address, and so on to register. They must also enter a password to check in later using their email address and the password they entered. The customer may request the booking and examine their booking history after signing in. The administrator must also log in to access the booking system. The admin can validate the customer's desired booking and assign a driver to the booking. The administrator registers the driver by entering the driver's information. The administrator will then provide the password to the driver. The driver must first log in. After logging in, the driver may view their planned trips and reset their password. Each trip must be assigned to a unique driver, and trips assigned to the same driver must not overlap. The bill will be provided by the administrator only after the journey has been finished. The journey will be finished when the driver drops the customer off at their location.

Project Plan/Schedule

|  |  |  |
| --- | --- | --- |
| Week No. | Tasks | Priority |
| 7 | Case Study  System Analysis  Use Case-V1  UI/UX Design-V1  Activity Diagram-V1 | MUST |
| 8 | Database Connectivity  ER-Diagram-V1  CRUD Version-1  OOP Implementation-V1  Class Diagram-V1 | MUST |
| 9 | Use Case-V2  UI/UX Design-V2  Activity Diagram-V2  Class Diagram-V2  ER-Diagram-V2  OOP Implementation-V2  CRUD Version-2 | MUST |
| 10 | **Final Version**  Use Case  UI/UX Design  Activity Diagram  Class Diagram  ER-Diagram  OOP Implementation  CRUD Implementation  Final Report-V1  Final App-V1 | MUST |
| 11 | **Demo Demonstration and Presentation**  Final Report- Final Version  Final App- Final Version | MUST |
|  |  |  |
|  |  |  |
|  | Submit Group Report, Project Code and Video Recording (if required) |  |
|  | Project Presentation (if required) |  |

Requirements Analysis

Overview of Functional, Technical (Non-Functional Requirements) and Usability Requirements

Functional Requirements

TBS = Taxi Booking System

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | A customer must be able to register on the TBS | MUST |
| 2 | A customer must be able to log in to the TBS | MUST |
| 3 | A customer must be able to log out of the TBS | COULD |
| 4 | An administrator must be able log in to the TBS | SHOULD |
| 5 | An administrator must be able to log out of the TBS | COULD |
| 6 | An administrator must be able to confirm a booking | SHOULD |
| 7 | An administrator must be able to allocate a driver to a booking | SHOULD |
| 8 | An administrator must be able to view all bookings | SHOULD |
| 9 | An administrator must be able to cancel a booking | SHOULD |
| 10 | A customer must be able to view his/her bookings | MUST |
| 11 | A customer must be able to make a booking | MUST |
| 12 | A customer must be able to cancel a booking | MUST |
| 13 | A taxi driver must be able to log in to the TBS | MUST |
| 14 | An administrator must be able to log out of the TBS | SHOULD |
| 15 | A taxi driver must be able to view his bookings | MUST |

Non-functional Requirements

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | The TBS should process input and return results within 10 seconds |  |
| 2 | The TBS should run on a range of devices from PCs to mobile phones |  |
| 3 | The TBS design should be sufficiently scalable and flexible to allow for further future enhancements |  |
| 4 | The TBS users should not experience critical system failures. 99.99% ‘uptime’ should be achieved. |  |

Usability Requirements

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | The TBS should incorporate a user-centric design |  |
| 2 | The design should demonstrate evidence of a good understanding of interface design issues – for example, a consistent design for each form, layout of content, use of colour schemes and images, navigational methods, usability when viewed at various screen resolutions and various monitor dimensions. |  |
| 3 | All data entry forms should be short and easy to complete and there should be entry validation. |  |
| 4 | The TBS should have clear and intuitive navigation |  |
| 5 | The TBS should comply with WW3 Web Accessibility Standards (WCAG)  Text easy to read and language and language style should be appropriate with absence of grammar / spelling errors |  |
|  | There should be a clear layout which remains consistent throughout the application. Style, layout and content should be appropriate for the purpose of the application. |  |

\*MOSCOW Notation:

M = MUST

S = SHOULD

C = COULD

W = WON’T

Design

* UML Diagrams
* Uses Case Diagrams(s)

Diagram

Description automatically generated

Fig 1: Taxi Booking System - Use Case Diagram (Sea Level)

* + - Use Case Specifications/Description
    - Use Case Scenario
    - Use Case Diagrams at more than one level (Sea Level and Fish Level)

EXAMPLE OF USE CASE SPECIFICATIONS (NOT FOR THIS SCENARIO)

Use Case Specifications / Description

Allocate Accommodation

A specific room is allocated to each guest.

Change Booking

An unguaranteed booking may become guaranteed on provision of credit card details. Other changes (for example to the composition of the party or to the dates booked) may be made on the customer’s request.

Calculate Bill Total

The total bill for a room comprises the cost of the room plus any items charged to it (for example, from the minibar, hotel bar or restaurant and any additional services provided) plus the appropriate rate of VAT.

Cancel Booking

A customer may cancel a booking any time before checking in. The booked accommodation is de-allocated.

Charge Item to Room

A guest may use items from the minibar in the room, make external telephone calls, make purchases from the hotel bar or restaurant or order additional services such as laundry. These items are charged to the guest’s room. An external service ordered by the guest may also be charged to their room (although some services, such as taxis, will be paid for directly by the guest).

Check In Guest

When a party (one or more guests) arrives at the hotel each guest is checked in and the booking becomes registered. Details of all the guests are confirmed, or if they have not been given at the time of booking they are recorded. The method of payment (by which guest(s) or whether to be charged to a corporate account) is confirmed. Accommodation is allocated if not already done at the time of booking and special requests are recorded.

Check Out Guest

Before the guest leaves the hotel they need to check out and the booking becomes completed. The receptionist provides them with the total room bill, which an individual customer must pay. If the bill is to be charged to a corporate customer the guest is asked to check and sign the bill, it is then added to the appropriate corporate account.

Make a Booking

When a customer contacts the receptionist with a booking request the receptionist checks the room occupancy for the required period and if the customer's needs can be met, makes a booking.

A guaranteed booking is only made for corporate customers or for individual customers who supply credit card details.

Accommodation may be allocated at this stage; alternatively where it is a group booking (for example, for 6 twin-bedded rooms) the allocation of each guest in the party to a specific room may be done on checking in. Either way, occupancy records are updated to avoid double booking.

Order External Service

Services not offered by the hotel itself can be ordered, once their availability has been checked. The guest name, room number and details of the service required are taken when the service is ordered.

Provide Guest with Current Bill

On request a guest may be provided with the current total of their bill.

* Activity Diagram(s)
* Class Diagram(s)

Table

Description automatically generated with medium confidence

* Classes (omitting methods / operations)
  + Customer (customertid, title, firstname, lastname, email, telno, password, address1, town, county, postcode, paymentmethod)
  + Booking (bookingid, datebooked, startaddress, destinationaddress, date, time, status, paid)
  + Taxidriver (driverid, title, firstname, lastname, email, password, regno)
  + Administrator (adminid, firstname, lastname, email, password)
* Database Design
* Logical Database Design
  + Entity Relationship Model (ERM)
    - ERM Diagram

Diagram

Description automatically generated

Diagram

Description automatically generated

* + - List of Entities

Customer (**customertid**, title, firstname, lastname, email, telno, password, address1, town, county, postcode, paymentmethod)

Booking (**bookingid**, customerid\*, driverid\*, datebooked, startaddress, destinationaddress, date, time, status, paid)

Taxidriver (**driverid**, title, firstname, lastname, email, password, regno)

Administrator (**adminid**, firstname, lastname, email, password)

Primary Key = Bold and Underlined

Foreign Key = \*

* Physical Database Design
  + Skeleton Tables (with Primary Keys and Foreign Keys)

Customer (**customertid**, title, firstname, lastname, email, telno, password, address1, town, county, postcode, paymentmethod)

Booking (**bookingid**, customerid\*, driverid\*, datebooked, startaddress, destinationaddress, date, time, status, paid)

Taxidriver (**driverid**, title, firstname, lastname, email, password, regno)

Administrator (**adminid**, firstname, lastname, email, password)

Primary Key = Bold and Underlined

Foreign Key = \*

* + Data Dictionary

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Customer | | | | | | | |
| Description: Customer details | | | | | | | |
| Field Name | **Datatype** | **Length** | **Index** | **Null** | **Default** | **Validation rule** | **Description** |
| customerid *(Primary)* | int (10)  unsigned | 10 | PK | No |  |  | Autoincremented Uniquely identifies every customer |
| title | varchar (10) | 10 |  | No |  |  | Title of customer |
| firstname | varchar (20) | 20 |  | No |  |  | First name of customer |
| lastname | varchar (30) | 30 |  | No |  |  | Last name of customer |
| email | varchar (50) | 50 |  | No |  | Must be email format containing an @ and a ‘.’  Regex expression used | Email of customer |
| telno | varchar (20) | 20 |  | No |  |  | Telephone number of customer |
| password | varchar (20) | 20 |  | No |  |  | TBS password |
| address1 | varchar (30) | 30 |  | No |  |  | First line of address |
| town | varchar (30) | 30 |  | No |  |  | Customer town |
| county | varchar (30) | 30 |  | No |  |  | Customer county |
| postcode | varchar (15) | 15 |  | No |  |  | Customer postcode |
| paymentmethod | varchar (20) | 20 |  | No |  |  | Customer payment method |

Indexes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Keyname** | **Type** | **Unique** | **Column** | **Null** |
| PRIMARY | BTREE | Yes | customerid | No |

* User Interface Design

Sketches of your GUI or text-based interfaces

You should use Balsamiq Wireframes (see under Announcements) or an online prototyping tool (for example, <https://marvelapp.com/>)

* Algorithm Design (could be omitted)

For CSI093-1 only

Examples later!

Implementation

You should discuss YOUR application. How was the program created?

You should describe the development stages.

Explain how you manged the implementation. How did you ensure all group members actively contributed to writing code? How did you distribute the workload? Was it successful?

What Integrated Development Environment (IDE) did you use? Justify your choice of IDE, Python Framework (if used), packages / libraries.

What problems did you encounter during the coding, and explain how you overcame these?

Include snippets of code here – that either caused problems or show original, novel approaches

Include a couple of screen shots of your application here

What rDBMS did you use? Possibly, SQLite, MySQL. Justify your choice of database system.

Did you use WAMP / XAMPP / MAMP or an equivalents stack on your own PC?

Detailed discussion of your experience coding is required here.

Did you incorporate any novel approaches to coding and meeting the Assignment requirements?

Testing

Include evidence of detailed and robust testing.

There are various ways to document this…

Test Plan / Test Log (tabular format) with date, input/expected output/actual output and screen shot of result.

OR

Documentation of each test – one after the other

You need to include:

* Test No
* Test Date
* Purpose of test
* Input data or action
* Expected result
* Actual result
* Action if the above are not the same
* Screenshot of output

Discussion / Reflection / Critical Analysis

* What went well and what went wrong?
* What would you have done differently?
* How would you improve your application in the future?

This part is very important. You must discuss the project, group work, and time management skills.

Basically, you are evaluating what you have done. Did you achieve all the required specification of the Assignment brief? If not, why not? Discuss the problems you encountered and how you overcame them.

What stopped you achieving what you had hoped to achieve and to the standard you know you are capable of achieving? Remote learning, lack of access to BREO, COVID, lack of suitable devices remotely, Internet connection issues etc.

Did you give yourselves enough time to accomplish this work? Was it harder than you expected? What have you learned from tackling this Assignment?

If working as a group, did your group work well together? What did you learn about the dynamics of group work?

Did this work enhance your understanding of the Python programming language? Did it motivate you to learn more Python, and other programming languages?

If you had to undertake this or something similar again would you tackle it differently?

What improvements or enhancements would you recommend for a future iteration of this project?

Conclusion

What were you asked to do? Did you achieve it?

References

Use the Harvard Referencing System. Any reference must be cited in text -otherwise it should go in a Bibliography section.

Appendix

Make sure that all content has a Fig. No., caption and explanation.

You MUST include:

Complete project code.

Include file/class names.

Make sure you code is fully commented and well presented with correct indentation and colour coding.

You could include:

Details of you database/table structure and content (from PHPMyAdmin / SQLite Studio)

Screen shots of your GUI or text-based interfaces

Anything else that you might want to include which does not fit well into the main report body.