

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

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

**(Taxi Booking System)**

**Student Name: Hancie Phago - University ID: 2126466**

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 –Individual Project – Case Study (Taxi Booking System)

Student Name and University ID

Include University IDs and Full Names

University ID: 2126466 Full name: Hancie Phago

IF YOU ARE WORKING AS AN INDIVIDUAL YOU NEED TO MAKE THIS VERY CLEAR

# Introduction/Overview

Every day, technology advances to a higher level. Modern technology allows people to quickly find a solution to a complex problem. Thus, programming languages have been crucial in improving computer technology in the current period. A growing number of programming languages are being developed day by day to fulfil peoples' demands in daily life. So, for this assignment, Python programming language is used. Python is a general-purpose, high-level programming language that can be used to create applications for various platforms and supports a variety of programming paradigms.

For this assignment, we were assigned to create an Online Taxi booking system by the university. The Taxi Booking system is a desktop online-based application created using Python programming. The Taxi booking system provides many features which help to manage the taxi booking requested by the customer. Basically, it plays three key roles. They are customers, company administration and taxi drivers.

This assignment includes everything needed to meet the university's requirements. The report and application section focused mostly on producing attractive reports and useful applications. The database design idea of the Rational Database Management System (RDMS) is used for this Taxi Booking System assignment. The local server is powered by Xampp, and the database for the taxi booking system is built using MySQL. In order to demonstrate the functional application for this Taxi Booking System assignment, a visually appealing interface is created using the Python Tkinter GUI toolkit. The Object-Oriented Programming Language idea is the foundation of this application in its entirety.

Many problems were faced but due to consistent attempts, this assignment has turned to this platform. Exploring Python programming knowledge and abilities has been made easier with the help of this assignment. The exploration of application knowledge as well as the creation of a thorough report were both helped by this assignment directly or indirectly.

Task Description

Taxi Booking System is an assignment which is based on python programming language. The assignment scenario was provided by the university, and this assignment was created using that scenario. Taxi booking system is an online desktop application where customer can request a ride and book a trip using customer dashboard. Before requesting a ride, customers must register with the company. After successfully registering, the customer can log in to the customer dashboard. Customers can request a ride by providing the pickup and drop-off addresses, as well as the time and date. The day prior to the current date cannot be requested by the customer. Before admin confirms the booking, the customer can also make changes in the booking. If a customer wants to cancel a booking, they may do so by cancelling the booking before admin confirm the booking.

Customers can view their booking history via their dashboard, among other advantages. The customer can view the costs incurred when booking the ride. Customers can also see their account details and, if necessary, they can change the details. If a customer decides they no longer want to use this program, they can delete their account. The driver can be assigned to the customer for the trip by the admin. The driver status won't be available after being assigned until the driver has completed the trip.

The taxi driver can access the system as well using his or her account. Taxi drivers can indicate when they are available and when they are not, depending on how busy they are. When a customer requests a ride, the taxi driver proceeds to pick them up while checking their information. After the ride is finished, the taxi driver indicates the ride is complete.

The company admin can also view the overall number of clients, revenue, rides, drivers, etc. The admin of the company can also modify the customer and driver information. Additionally, the administrator can create a bill for the consumer and store the bill voucher in the database for later use.

This is basically a description of the task as on the Assignment brief. You can paraphrase what is written in the Assignment 2 brief. LOOK AT THE ASSIGNMENT BRIEF AND SEE WHAT YOU CAN INCLUDE.

Include the scenario, and Use Case Diagram from the Brief.

Include any assumptions you make about the system.

Project Plan/Schedule

|  |  |  |
| --- | --- | --- |
| Week No. | Tasks | Priority |
| 7 | System Analysis and case study of assignment. Made use case diagram with sea level, fish level and clam level. Finished making UI/UX design for prototype. Finished making activity diagram as per clam level use case instructed by the university. | Use case diagram and UI/UX design |
| 8 | Successfully made database connectivity with a graphical user interface (GUI). ER-Diagram creation for the database and report is complete. I've completed developing some CRUD for a GUI using object-oriented programming (OOP). I've also completed the class diagram I was supposed to create for this assignment. | CRUD operation with OOP implementation. Class diagram. |
| 9 | Completed CRUD operation for customer and taxi driver in objected-oriented method. Completed making GUI for additional features for this assignment. Made activity diagram for some of the role. | CRUD operation for customer and taxi driver. |
| 10 | Added some additional features for the assignment like password change, account view system, report of growth and loss of company in graph. | Focused on additional features. |
| 11 | Completed final report for the assignment by filling all the requirement mentioned in the assignment scenario. And made final report and application presentation in the college. | Activity diagram, class diagram, data dictionary, implementation, testing |

Requirements Analysis

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

Functional Requirements

|  |  |  |
| --- | --- | --- |
| Req. No | Requirement | Priority\* |
| 1 | A customer must be able to register on the Taxi Booking System before login. | MUST |
| 2 | A customer must be able to log in to the Taxi Booking System after successful register in the registration form. | MUST |
| 3 | A customer must be able to log out of the Taxi Booking System after customer request the car for travelling | COULD |
| 4 | An administrator must be able log in to the Taxi Booking System by using predefined username and email stored in the system database. | SHOULD |
| 5 | An administrator must be able to log out of the Taxi Booking System after performing certain work. | COULD |
| 6 | An administrator must be able to confirm a booking requested by the customer. | MUST |
| 7 | An administrator must be able to allocate a driver to a booking after customer request the taxi by allocating the available driver to the customer. | SHOULD |
| 8 | An administrator must be able to view all bookings made by the customer in the company. | SHOULD |
| 9 | An administrator must be able to cancel a booking if driver is not available in the company. | SHOULD |
| 10 | A customer must be able to view his/her bookings with CRUD functions. | MUST |
| 11 | A customer must be able to make a booking by send the data like pickup address, drop-off address with current time. | MUST |
| 12 | A customer must be able to cancel a booking if customer make booking mistakenly. | MUST |
| 13 | A taxi driver must be able to log in to the Taxi Booking system with the details like driver name and taxi plate number. | MUST |
| 14 | An administrator must be able to log out of the Taxi Booking system after using it. | SHOULD |
| 15 | A taxi driver must be able to view his bookings and can make himself available and not available for certain period of time if he is busy. | 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)**

A class diagram is a graphic depiction of a class of objects in a model system that are grouped by class types. Class diagrams are used to model the system's constituent parts and show the connections between them. The class structure of a system is illustrated by each class type being represented as a rectangle with three compartments for the class name, attributes, and operations.

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.