

Higher Nationals

Internal verification of assessment decisions – BTEC (RQF)

INTERNAL VERIFICATION – ASSESSMENT DECISIONS			
Programme title	Higher National Diploma in Computing		
Assessor		Internal Verifier	
Unit(s)			
Assignment title			
Student's name			
List which assessment criteria the Assessor has awarded.	Pass	Merit	Distinction
INTERNAL VERIFIER CHECKLIST			
Do the assessment criteria awarded match those shown in the assignment brief?	Y/N		
Is the Pass/Merit/Distinction grade awarded justified by the assessor's comments on the student work?	Y/N		
Has the work been assessed accurately?	Y/N		
Is the feedback to the student: Give details: • Constructive? • Linked to relevant assessment criteria? • Identifying opportunities for improved performance? • Agreeing actions?	Y/N Y/N Y/N Y/N		
Does the assessment decision need amending?	Y/N		
Assessor signature			Date
Internal Verifier signature			Date
Programme Leader signature (if required)			Date

Confirm action completed			
Remedial action taken Give details:			
Assessor signature		Date	
Internal Verifier signature		Date	
Programme Leader signature (if required)		Date	

Higher Nationals - Summative Assignment Feedback Form

Student Name/ID			
Unit Title			
Assignment Number		Assessor	
Submission Date		Date Received 1st submission	
Re-submission Date		Date Received 2nd submission	

Assessor Feedback:

LO1. Define basic algorithms to carry out an operation and outline the process of programming an application.

Pass, Merit & Distinction Descripts P1 M1 D1

LO2. Explain the characteristics of procedural, object-orientated and event-driven programming, conduct an analysis of an Integrated Development Environment (IDE).

Pass, Merit & Distinction Descripts P2 M2 D2

LO3. Implement basic algorithms in code using an IDE.

Pass, Merit & Distinction Descripts P3 M3 D3

LO4. Determine the debugging process and explain the importance of a coding standard.

Pass, Merit & Distinction Descripts P4 P5 M4 D4

Grade:	Assessor Signature:	Date:
Resubmission Feedback:		
Grade:	Assessor Signature:	Date:
Internal Verifier's Comments:		
Signature & Date:		

* Please note that grade decisions are provisional. They are only confirmed once internal and external moderation has taken place and grades decisions have been agreed at the assessment board.

Assignment Feedback

Formative Feedback: Assessor to Student

Action Plan

Summative feedback

Feedback: Student to Assessor

Assessor signature		Date	
Student signature		Date	



Pearson Higher Nationals in Computing

Unit 01: Programming
Assignment 01

General Guidelines

1. A Cover page or title page – You should always attach a title page to your assignment. Use previous page as your cover sheet and make sure all the details are accurately filled.
2. Attach this brief as the first section of your assignment.
3. All the assignments should be prepared using a word processing software.
4. All the assignments should be printed on A4 sized papers. Use single side printing.
5. Allow 1" for top, bottom , right margins and 1.25" for the left margin of each page.

Word Processing Rules

1. The font size should be **12** point, and should be in the style of **Time New Roman**.
2. **Use 1.5 line spacing.** Left justify all paragraphs.
3. Ensure that all the headings are consistent in terms of the font size and font style.
4. Use **footer function in the word processor to insert Your Name, Subject, Assignment No, and Page Number on each page.** This is useful if individual sheets become detached for any reason.
5. Use word processing application spell check and grammar check function to help editing your assignment.

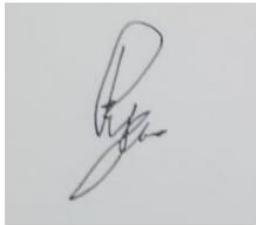
Important Points:

1. **It is strictly prohibited to use textboxes to add texts in the assignments, except for the compulsory information. eg: Figures, tables of comparison etc. Adding text boxes in the body except for the before mentioned compulsory information will result in rejection of your work.**
2. Carefully check the hand in date and the instructions given in the assignment. Late submissions will not be accepted.
3. Ensure that you give yourself enough time to complete the assignment by the due date.
4. Excuses of any nature will not be accepted for failure to hand in the work on time.
5. You must take responsibility for managing your own time effectively.
6. If you are unable to hand in your assignment on time and have valid reasons such as illness, you may apply (in writing) for an extension.
7. Failure to achieve at least PASS criteria will result in a REFERRAL grade .
8. Non-submission of work without valid reasons will lead to an automatic RE FERRAL. You will then be asked to complete an alternative assignment.
9. If you use other people's work or ideas in your assignment, reference them properly using HARVARD referencing system to avoid plagiarism. You have to provide both in-text citation and a reference list.
10. If you are proven to be guilty of plagiarism or any academic misconduct, your grade could be reduced to A REFERRAL or at worst you could be expelled from the course

Student Declaration

I hereby, declare that I know what plagiarism entails, namely to use another's work and to present it as my own without attributing the sources in the correct way. I further understand what it means to copy another's work.

1. I know that plagiarism is a punishable offence because it constitutes theft.
2. I understand the plagiarism and copying policy of the Edexcel UK.
3. I know what the consequences will be if I plagiarise or copy another's work in any of the assignments for this program.
4. I declare therefore that all work presented by me for every aspects of my program, will be my own, and where I have made use of another's work, I will attribute the source in the correct way.
5. I acknowledge that the attachment of this document signed or not, constitutes a binding agreement between myself and Edexcel UK.
6. I understand that my assignment will not be considered as submitted if this document is not attached to the attached.



ryandilthusha@gmail.com

Student's Signature:
(Provide E-mail ID)

04/02/2022

Date:
(Provide Submission Date)

Higher National Diploma in Computing

Assignment Brief

Student Name /ID Number	Ryan Wickramaratne (COL 00081762)
Unit Number and Title	Unit 01: Programming
Academic Year	2021/22
Unit Tutor	Mr. Praveen Croos
Assignment Title	Design & Implement a GUI based system using a suitable Integrated Development Environment
Issue Date	
Submission Date	
IV Name & Date	
Submission Format	
This submission will have 3 components	
1. Written Report This submission is in the form of an individual written report. This should be written in a concise, formal business style using single spacing and font size 12. You are required to make use of headings, paragraphs and subsections as appropriate, and all work must be supported with research and referenced using the Harvard referencing system. Please also provide a bibliography using the Harvard referencing system. (The recommended word count is 1,500–2,000 words for the report excluding annexures)	
2. Implemented System (Software) The student should submit a GUI based system developed using an IDE. The system should connect with a backend database and should have at least 5 different forms and suitable functionality including insert, edit and delete of main entities and transaction processing.	
3. Presentation With the submitted system student should do a presentation to demonstrate the system that was developed. Time allocated is 10 to 15 min. Student may use 5 to 10 PowerPoint slides while doing the presentation, but live demonstration of the system is required. Evaluator will also check the ability to modify and debug the system using the IDE.	

Unit Learning Outcomes:

- | | |
|--|--|
| | <p>LO1. Define basic algorithms to carry out an operation and outline the process of programming an application.</p> <p>LO2. Explain the characteristics of procedural, object-orientated and event-driven programming, conduct an analysis of a suitable Integrated Development Environment (IDE).</p> <p>LO3. Implement basic algorithms in code using an IDE.</p> <p>LO4. Determine the debugging process and explain the importance of a coding standard</p> |
|--|--|

Assignment Brief and Guidance:

	<p>Activity 1</p> <p>A. The Fibonacci numbers are the numbers in the following integer sequence. 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, 55, 89, 144,</p> <p>In mathematical terms, the sequence F_n of Fibonacci numbers is defined by the recurrence relation.</p> $F_n = F_{n-1} + F_{n-2}$ <p>B. Factorial of a non-negative integer, is multiplication of all integers smaller than or equal to n. For example, factorial of 6 is $6*5*4*3*2*1$ which is 720.</p> $n! = n * (n - 1) * 1$ <p>Define what an algorithm is and outline the characteristics of a good algorithm. Write the algorithms to display the Fibonacci series and the factorial value for a given number using Pseudo code. Determine the steps involved in the process of writing and executing a program.</p> <p>Take a sample number and dry run the above two algorithms. Show the outputs at the end of each iteration and the final output. Examine what Big-O notation is and explain its role in evaluating efficiencies of algorithms. Write the Python program code for the above two algorithms and critically evaluate their efficiencies using Big-O notation.</p> <p>Activity 2</p> <p>2.1 Explain what is meant by a Programming Paradigm and the main characteristics of Procedural, Object oriented and Event-driven paradigms and the relationships among them. Write small snippets of code as example for the above three programming paradigms using a suitable programming language(s). you also need to critically evaluate the code samples that you have given above in relation to their structure and the unique characteristics.</p>

Activity 3 and Activity 4 are based on the following Scenario.

Ayubo Drive is the transport arm of Ayubo Leisure (Pvt) Ltd, an emerging travel & tour company in Sri Lanka. It owns a fleet of vehicles ranging from cars, SUVs to vans.

The vehicles that it owns are hired or rented with or without a driver. The tariffs are based on the vehicle type. Some of the vehicle types that it operates are, small car, sedan car, SVUs, Jeep (WD), 7-seater van and Commuter van. New vehicle types are to be added in the future.

Vehicle rent and hire options are described below.

1. Rent (With or without driver) – For each type of vehicle rates are given per day, per week and per month. Rate for a driver also given per day. Depending on the rent period the total rent amount needs to be calculated. For example: if a vehicle is rented for 10 days with a driver, total amount to be calculated as follows:

$$\text{Total rent} = \text{weeklyRent} \times 1 + \text{dailyRent} \times 3 + \text{dailyDriverCost} \times 10$$

2. Hire (with driver only) – These are based on packages such as airport drop, airport pickup, 100km per day package, 200km per day package etc. Standard rates are defined for a package type of a vehicle type if that is applicable for that type of vehicle. For each package maximum km limit and maximum number of hours are also defined. Extra km rate is also defined which is applicable if they run beyond the allocated km limit for the tour. For day tours if they exceed max hour limit, a waiting charge is applicable for extra hours. Driver overnight rate and vehicle night park rate also defined which is applicable for each night when the vehicle is hired for 2 or more days.

Activity 3

Function 1: **Rent calculation.**

Return the total rent_value when vehicle_no, rented_date, return_date, with_driver parameters are sent in. with_driver parameter is set to true or false depending whether the vehicle is rented with or without driver.

Function 2: **Day tour - hire calculation.**

Calculate total hire_value when vehicle_no, package_type, start_time, end_time, start_km_reading, end_km_reading parameters are sent in. Should return base_hire_charge, waiting_charge and extra_km_charge as output parameters.

Function 3: **Long tour - hire calculation.**

Calculate total hire_value when vehicle_no, package_type, start_date, end_date, start_km_reading, end_km_reading parameters are sent in. Should return base_hire_charge, overnight_stay_charge and extra_km_charge as output parameters.

Write suitable algorithms for vehicle tariff calculation for rents and hires. Ideally 3 functions should be developed for this purpose as above. Use the visual studio IDE (using C#.net) to Implement the above algorithms and design the suitable database structure for keeping the tariffs for vehicle types and different packages which must be used for implementing the above functions.

Analyze the features of an Integrated Development Environment (IDE) and explain how those features help in application development. Evaluate the use of the Visual Studio IDE for your application development contrasted with not using an IDE.

Activity 4

4.1 Design and build a small system to calculate vehicle hire amounts and record them in a database for customer billing and management reporting for Ayubo drive. This includes the completing the database design started in 3.2 and implementing one or more GUIs for vehicle, vehicle type, and package add/edit/delete functions. It essentially requires an interface for hire calculation and recording function described above. Generating customer reports and customer invoices are not required for this course work.

4.2 Explain debugging process and the features available in Visual studio IDE for debugging your code more easily. Evaluate how you used the debugging process to develop more secure, robust application with examples.

4.3 Outline the coding standards you have used in your application development. Critically evaluate why a coding standard is necessary for the team as well as for the individual.

Grading Rubric

Grading Criteria	Achieved	Feedback
LO1 Define basic algorithms to carry out an operation and outline the process of programming an application.		
P1 Provide a definition of what an algorithm is and outline the process in building an application.		
M1 Determine the steps taken from writing code to execution.		
D1 Evaluate the implementation of an algorithm in a suitable language. Evaluate the relationship between the written algorithm and the code variant		
LO2 Explain the characteristics of procedural, object orientated and event-driven programming, conduct an analysis of a suitable Integrated Development Environment (IDE)		
P2 Give explanations of what procedural, object orientated, and event driven paradigms are; their characteristics and the relationship between them.		
M2 Compare and contrast the procedural, object orientated and event driven paradigms used in given source code of an application		
D2 Critically evaluate the source code of an application which implements the programming paradigms, in terms of the code structure and characteristics.		

LO3 Implement basic algorithms in code using an IDE.		
P3 Write a program that implements an algorithm using an IDE.		
M3 Use the IDE to manage the development process of the program.		
D3 Evaluate the use of an IDE for development of applications contrasted with not using an IDE.		
LO4 Determine the debugging process and explain the importance of a coding standard		
P4 Explain the debugging process and explain the debugging facilities available in the IDE.		
P5 Outline the coding standard you have used in your code.		
M4 Evaluate how the debugging process can be used to help develop more secure, robust applications.		
D4 Critically evaluate why a coding standard is necessary in a team as well as for the individual.		

Acknowledgement

I would like to express my special thanks of gratitude to my programming lecturer Mr. Praveen for providing invaluable guidance and giving immense amount of knowledge to work on this assignment perfectly. I specially thanks him because he helped us in doing a lot of research and I came to know about so many new things about the computer programming.

Secondly, I would like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Executive Summary

This entire assignment is based on an implementation of a Car rent and hire system design for selected company (Ayubo Drive). The purpose of this assignment is to improve programming skills.

This report includes basic programming principles, creating algorithms, understanding time complexities, creating basic programs by using Python language and C# language, understanding different programming paradigms, working with IDEs, Basic system creation using Microsoft SQL and Visual Studio C# windows forms application, debugging process and coding standards.

List of figures

FIGURE 1. 1 EXAMPLE PYTHON CODE AND OUTPUT FOR DEMONSTRATE BIG O NOTATION “CONSTANT TIME”	36
FIGURE 1. 2 EXAMPLE PYTHON CODE AND OUTPUT FOR DEMONSTRATE BIG O NOTATION “LINEAR TIME”	37
FIGURE 1. 3 EXAMPLE PYTHON CODE AND OUTPUT FOR DEMONSTRATE BIG O NOTATION “QUADRATIC TIME”	38
FIGURE 1. 4 FINDING TIME COMPLEXITY FOR SIMPLE OPERATIONS SAMPLE QUESTION.....	40
FIGURE 1. 5 FINDING TIME COMPLEXITY FOR SIMPLE OPERATIONS ANSWER FOR THE QUESTION	40
FIGURE 1. 6 FINDING OUT THE TIME COMPLEXITY FOR SINGLE LOOP SAMPLE QUESTION	41
FIGURE 1. 7 FINDING OUT THE TIME COMPLEXITY FOR SINGLE LOOP ANSWER FOR THE QUESTION.....	41
FIGURE 1. 8 FINDING OUT THE TIME COMPLEXITY FOR NESTED LOOPS SAMPLE QUESTION.....	42
FIGURE 1. 9 FINDING OUT THE TIME COMPLEXITY FOR NESTED LOOPS ANSWER FOR THE QUESTION	42
FIGURE 1. 10 FINDING OUT THE TIME COMPLEXITY FOR SEQUENTIAL STATEMENTS SAMPLE QUESTION	43
FIGURE 1. 11 FINDING OUT THE TIME COMPLEXITY FOR SEQUENTIAL STATEMENTS ANSWER FOR THE QUESTION	43
FIGURE 1. 12 FINDING OUT THE TIME COMPLEXITY FOR CONDITIONAL STATEMENTS SAMPLE QUESTION	44
FIGURE 1. 13 FINDING OUT THE TIME COMPLEXITY FOR CONDITIONAL STATEMENTS ANSWER FOR THE QUESTION	44
FIGURE 1. 14 TIME COMPLEXITIES COMPARISON GRAPH	45
FIGURE 1. 15 FIBONACCI SERIES.....	46
FIGURE 1. 16 FIBONACCI SERIES ALGORITHM BY USING PYTHON	50
FIGURE 1. 17 FIBONACCI SERIES DRY RUN TAKING SAMPLE NUMBER 3	51
FIGURE 1. 18 FIBONACCI SERIES DRY RUN TAKING SAMPLE NUMBER 2	51
FIGURE 1. 19 FIBONACCI SERIES DRY RUN TAKING SAMPLE NUMBER 1	51
FIGURE 1. 20 FIBONACCI SERIES DRY RUN TAKING SAMPLE NUMBER 0	51
FIGURE 1. 21 FIBONACCI SERIES DRY RUN TAKING SAMPLE NUMBER 10	52
FIGURE 1. 22 EVALUATION EFFICIENCY OF FIBONACCI SERIES ALGORITHM USING BIG-O NOTATION	53
FIGURE 1. 23 FACTORIAL VALUE	55
FIGURE 1. 24 FACTORIAL VALUE ALGORITHM BY USING PYTHON.....	59
FIGURE 1. 25 FACTORIAL VALUE DRY RUN TAKING SAMPLE NUMBER 3.....	60
FIGURE 1. 26 FACTORIAL VALUE DRY RUN TAKING SAMPLE NUMBER 2	60
FIGURE 1. 27 FACTORIAL VALUE DRY RUN TAKING SAMPLE NUMBER 1.....	60
FIGURE 1. 28 FACTORIAL VALUE DRY RUN TAKING SAMPLE NUMBER 0.....	60
FIGURE 1. 29 FACTORIAL VALUE DRY RUN TAKING SAMPLE NUMBER 10.....	61
FIGURE 1. 30 EVALUATION EFFICIENCY OF FACTORIAL VALUE ALGORITHM USING BIG-O NOTATION.....	62
FIGURE 2. 1 TYPES OF PROGRAMMING PARADIGMS	68
FIGURE 2. 2 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE PYTHON CODE.....	72
FIGURE 2. 3 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE PYTHON CODE OUTPUT	72
FIGURE 2. 4 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE PYTHON CODE.....	73
FIGURE 2. 5 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE PYTHON CODE OUTPUT	73
FIGURE 2. 6 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE C# CODE.....	74
FIGURE 2. 7 PROCEDURAL PROGRAMMING PARADIGM EXAMPLE C# CODE OUTPUT	74
FIGURE 2. 8 OBJECT-ORIENTED PROGRAMMING PARADIGM EXAMPLE PYTHON CODE	77
FIGURE 2. 9 OBJECT-ORIENTED PROGRAMMING PARADIGM EXAMPLE PYTHON CODE OUTPUT	77
FIGURE 2. 10 OBJECT-ORIENTED PROGRAMMING PARADIGM EXAMPLE C# CODE	78
FIGURE 2. 11 OBJECT-ORIENTED PROGRAMMING PARADIGM EXAMPLE C# CODE OUTPUT	78
FIGURE 2. 12 EVENT-DRIVEN PROGRAMMING PARADIGM EXAMPLE C# WINDOWS FORMS CODE.....	81

FIGURE 3. 1 BEST IDES TO USE	91
FIGURE 3. 2 FLOW CHART FOR VEHICLE TARIFF CALCULATION FOR RENTS	95
FIGURE 3. 3 VISUAL STUDIO IDE INTERFACE TO CALCULATE VEHICLE RENTS.....	96
FIGURE 3. 4 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE RENTS - PART 1	97
FIGURE 3. 5 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE RENTS - PART 2	97
FIGURE 3. 6 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE RENTS - PART 3	98
FIGURE 3. 7 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE RENTS - PART 4	98
FIGURE 3. 8 "DETAILS_RENTING_TABLE" TABLE TO STORE VEHICLE RENT DETAILS	99
FIGURE 3. 9 STORED TABLE VALUES IN "DETAILS_RENTING_TABLE"	99
FIGURE 3. 10 "BILL_RENT_TABLE" TABLE TO BILL VEHICLE RENT DETAILS.....	100
FIGURE 3. 11 STORED TABLE VALUES IN "BILL_RENT_TABLE"	100
FIGURE 3. 12 RENT VEHICLE FOR 1 DAY WITH DRIVER	101
FIGURE 3. 13 RENT VEHICLE FOR 8 DAYS WITHOUT DRIVER	101
FIGURE 3. 14 RENT VEHICLE FOR 31 DAYS WITH DRIVER.....	102
FIGURE 3. 15 VEHICLE RENT VALUES ADDING TO TEMPORARY BILL TABLE	102
FIGURE 3. 16 VEHICLE RENT VALUES ADDING TO RENT BILL TABLE AND PRINTING THE BILL.....	103
FIGURE 3. 17 FLOW CHART FOR VEHICLE TARIFF CALCULATION FOR ONE DAY HIRES	104
FIGURE 3. 18 VISUAL STUDIO IDE INTERFACE TO CALCULATE VEHICLE ONE DAY HIRES	105
FIGURE 3. 19 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE ONE DAY HIRES - PART 1.....	106
FIGURE 3. 20 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE ONE DAY HIRES - PART 2.....	106
FIGURE 3. 21 "DETAILS_ONEDAY_HIRE_TABLE" TABLE TO STORE VEHICLE ONE DAY HIRES DETAILS.....	107
FIGURE 3. 22 STORED TABLE VALUES IN "DETAILS_ONEDAY_HIRE_TABLE"	107
FIGURE 3. 23 "BILL_ONEDAY_HIRE_TABLE" TABLE TO BILL VEHICLE ONE DAY HIRES DETAILS	108
FIGURE 3. 24 STORED TABLE VALUES IN "BILL_ONEDAY_HIRE_TABLE"	108
FIGURE 3. 25 ONE DAY HIRE VEHICLE FOR LESS THAN TIME AND DISTANCE LIMIT	109
FIGURE 3. 26 ONE DAY HIRE VEHICLE FOR LESS THAN TIME LIMIT BUT OVER THE DISTANCE LIMIT	109
FIGURE 3. 27 ONE DAY HIRE VEHICLE FOR OVER THAN TIME LIMIT BUT LESS THE DISTANCE LIMIT	110
FIGURE 3. 28 ONE DAY HIRE VEHICLE FOR OVER THAN TIME AND DISTANCE LIMIT	110
FIGURE 3. 29 VEHICLE ONE DAY HIRE VALUES ADDING TO TEMPORARY BILL TABLE.....	111
FIGURE 3. 30 VEHICLE ONE DAY HIRE VALUES ADDING TO ONE DAY HIRE BILL TABLE AND PRINTING THE BILL	111
FIGURE 3. 31 FLOW CHART FOR VEHICLE TARIFF CALCULATION FOR LONG TOUR HIRES	112
FIGURE 3. 32 VISUAL STUDIO IDE INTERFACE TO CALCULATE VEHICLE LONG TOUR HIRES.....	113
FIGURE 3. 33 VISUAL STUDIO IDE CODING TO CALCULATE VEHICLE LONG TOUR HIRES - PART 1	114
FIGURE 3. 34 "DETAILS_LONGTour_HIRE_TABLE" TABLE TO STORE VEHICLE LONG TOUR HIRES DETAILS	115
FIGURE 3. 35 STORED TABLE VALUES IN "DETAILS_LONGTour_HIRE_TABLE"	115
FIGURE 3. 36 "BILL_LONGTour_HIRE_TABLE" TABLE TO BILL VEHICLE LONG TOUR HIRES DETAILS	116
FIGURE 3. 37 STORED TABLE VALUES IN "BILL_ONEDAY_HIRE_TABLE"	116
FIGURE 3. 38 LONG TOUR HIRE VEHICLE FOR LESS THAN DISTANCE LIMIT AND FOR 1 NIGHT	117
FIGURE 3. 39 LONG TOUR HIRE VEHICLE FOR OVER THE DISTANCE LIMIT AND FOR 2 NIGHTS.....	117
FIGURE 3. 40 VEHICLE LONG TOUR HIRE VALUES ADDING TO TEMPORARY BILL TABLE	118
FIGURE 3. 41 VEHICLE LONG TOUR HIRE VALUES ADDING TO LONG TOUR BILL TABLE AND PRINTING THE BILL.....	118

FIGURE 4. 1 CAR RENT SYSTEM ROAD MAP AS A FLOW CHART.....	120
FIGURE 4. 2 SPLASH SCREEN FORM DESIGN	121
FIGURE 4. 3 SPLASH SCREEN FORM CODE - PART 1	121
FIGURE 4. 4 LOGIN FORM DESIGN	122
FIGURE 4. 5 LOGIN FORM CODE - PART 1	123
FIGURE 4. 6 LOGIN FORM CODE - PART 2	123
FIGURE 4. 7 LOGIN FORM CODE - PART 3	124
FIGURE 4. 8 DRIVER SIGNUP FORM DESIGN	125
FIGURE 4. 9 DRIVER SIGNUP FORM CODE - PART 1	126

FIGURE 4. 10 DRIVER SIGNUP FORM CODE - PART 2	126
FIGURE 4. 11 ADMIN LOGGED DASHBOARD FORM DESIGN	127
FIGURE 4. 12 ADMIN LOGGED DASHBOARD FORM CODE - PART 1.....	128
FIGURE 4. 13 ADMIN LOGGED DASHBOARD FORM CODE - PART 2.....	128
FIGURE 4. 14 ADMIN LOGGED DASHBOARD FORM CODE - PART 3.....	129
FIGURE 4. 15 ADMIN LOGGED DASHBOARD FORM CODE - PART 4.....	129
FIGURE 4. 16 ADMIN LOGGED DASHBOARD FORM CODE - PART 5.....	130
FIGURE 4. 17 ADMIN LOGGED DASHBOARD FORM CODE - PART 6.....	130
FIGURE 4. 18 ADMIN LOGGED DASHBOARD FORM CODE - PART 7.....	131
FIGURE 4. 19 ADMIN LOGGED DASHBOARD FORM CODE - PART 8.....	131
FIGURE 4. 20 ADMIN LOGGED VEHICLES FORM DESIGN.....	132
FIGURE 4. 21 ADMIN LOGGED VEHICLES FORM CODE - PART 1	133
FIGURE 4. 22 ADMIN LOGGED VEHICLES FORM CODE - PART 2	133
FIGURE 4. 23 ADMIN LOGGED VEHICLES FORM CODE - PART 3	134
FIGURE 4. 24 ADMIN LOGGED VEHICLES FORM CODE - PART 4	134
FIGURE 4. 25 ADMIN LOGGED VEHICLES FORM CODE - PART 5	135
FIGURE 4. 26 ADMIN LOGGED RENT DETAILS FORM DESIGN	136
FIGURE 4. 27 ADMIN LOGGED RENT DETAILS FORM CODE - PART 1	137
FIGURE 4. 28 ADMIN LOGGED RENT DETAILS FORM CODE - PART 2	137
FIGURE 4. 29 ADMIN LOGGED RENT DETAILS FORM CODE - PART 3	138
FIGURE 4. 30 ADMIN LOGGED RENT DETAILS FORM CODE - PART 4	138
FIGURE 4. 31 ADMIN LOGGED RENT DETAILS FORM CODE - PART 5	139
FIGURE 4. 32 ADMIN LOGGED RENT DETAILS FORM CODE - PART 6	139
FIGURE 4. 33 ADMIN LOGGED ONE DAY HIRE DETAILS FORM DESIGN.....	140
FIGURE 4. 34 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 1	141
FIGURE 4. 35 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 2	141
FIGURE 4. 36 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 3	142
FIGURE 4. 37 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 4	142
FIGURE 4. 38 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 5	143
FIGURE 4. 39 ADMIN LOGGED ONE DAY HIRE DETAILS FORM CODE - PART 6	143
FIGURE 4. 40 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM DESIGN	144
FIGURE 4. 41 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 1	145
FIGURE 4. 42 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 2	145
FIGURE 4. 43 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 3	146
FIGURE 4. 44 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 4	146
FIGURE 4. 45 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 5	147
FIGURE 4. 46 ADMIN LOGGED LONG TOUR HIRE DETAILS FORM CODE - PART 6	147
FIGURE 4. 47 ADMIN LOGGED DRIVER DETAILS FORM DESIGN.....	148
FIGURE 4. 48 ADMIN LOGGED DRIVER DETAILS FORM CODE - PART 1	149
FIGURE 4. 49 ADMIN LOGGED DRIVER DETAILS FORM CODE - PART 2	149
FIGURE 4. 50 ADMIN LOGGED DRIVER DETAILS FORM CODE - PART 3	150
FIGURE 4. 51 ADMIN LOGGED DRIVER DETAILS FORM CODE - PART 4	150
FIGURE 4. 52 ADMIN LOGGED DRIVER DETAILS FORM CODE - PART 5	151
FIGURE 4. 53 DRIVER LOGGED RENT VEHICLE FORM DESIGN.....	152
FIGURE 4. 54 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 1	153
FIGURE 4. 55 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 2	153
FIGURE 4. 56 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 3	154
FIGURE 4. 57 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 4	154
FIGURE 4. 58 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 5	155
FIGURE 4. 59 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 6	155
FIGURE 4. 60 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 7	156

FIGURE 4. 61 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 8	156
FIGURE 4. 62 DRIVER LOGGED RENT VEHICLE FORM CODE - PART 9	157
FIGURE 4. 63 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM DESIGN	158
FIGURE 4. 64 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 1.....	159
FIGURE 4. 65 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 2.....	159
FIGURE 4. 66 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 3.....	160
FIGURE 4. 67 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 4.....	160
FIGURE 4. 68 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 5.....	161
FIGURE 4. 69 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 6.....	161
FIGURE 4. 70 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 7.....	162
FIGURE 4. 71 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 8.....	162
FIGURE 4. 72 DRIVER LOGGED ONE DAY HIRE VEHICLE FORM CODE - PART 9.....	163
FIGURE 4. 73 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM DESIGN.....	164
FIGURE 4. 74 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 1	165
FIGURE 4. 75 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 2	165
FIGURE 4. 76 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 3	166
FIGURE 4. 77 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 4	166
FIGURE 4. 78 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 5	167
FIGURE 4. 79 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 6	167
FIGURE 4. 80 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 7	168
FIGURE 4. 81 DRIVER LOGGED LONG TOUR HIRE VEHICLE FORM CODE - PART 8	168
FIGURE 4. 82 DRIVER LOGGED MY PROFILE FORM DESIGN	169
FIGURE 4. 83 DRIVER LOGGED MY PROFILE FORM CODE - PART 1.....	170
FIGURE 4. 84 DRIVER LOGGED MY PROFILE FORM CODE - PART 2	170
FIGURE 4. 85 DRIVER LOGGED MY PROFILE FORM CODE - PART 3	171
FIGURE 4. 86 DRIVER LOGGED MY PROFILE FORM CODE - PART 4	171
FIGURE 4. 87 DRIVER LOGGED ABOUT US FORM DESIGN.....	172
FIGURE 4. 88 DRIVER LOGGED ABOUT US FORM CODE - PART 1	173
FIGURE 4. 89 DRIVER LOGGED ABOUT US FORM CODE - PART 2	173
FIGURE 4. 90 VEHICLE DETAILS TABLE SQL QUERY	174
FIGURE 4. 91 VEHICLE DETAILS TABLE SQL TABLE DESIGN.....	174
FIGURE 4. 92 VEHICLE RENTING DETAILS TABLE SQL QUERY	175
FIGURE 4. 93 VEHICLE RENTING DETAILS TABLE SQL TABLE DESIGN.....	175
FIGURE 4. 94 VEHICLE ONE DAY HIRE DETAILS TABLE SQL QUERY	176
FIGURE 4. 95 VEHICLE ONE DAY HIRE DETAILS TABLE SQL TABLE DESIGN.....	176
FIGURE 4. 96 VEHICLE ONE DAY HIRE DETAILS TABLE SQL QUERY	177
FIGURE 4. 97 VEHICLE ONE DAY HIRE DETAILS TABLE SQL TABLE DESIGN.....	177
FIGURE 4. 98 DRIVER DETAILS TABLE SQL QUERY	178
FIGURE 4. 99 DRIVER DETAILS TABLE SQL TABLE DESIGN.....	178
FIGURE 4. 100 VEHICLE RENTING BILLS DETAILS TABLE SQL QUERY	179
FIGURE 4. 101 VEHICLE RENTING BILLS DETAILS TABLE SQL TABLE DESIGN	179
FIGURE 4. 102 VEHICLE ONE DAY HIRE BILLS DETAILS TABLE SQL QUERY	180
FIGURE 4. 103 VEHICLE ONE DAY HIRE BILLS DETAILS TABLE SQL TABLE DESIGN	180
FIGURE 4. 104 VEHICLE LONG TOUR HIRE BILLS DETAILS TABLE SQL QUERY	181
FIGURE 4. 105 VEHICLE LONG TOUR HIRE BILLS DETAILS TABLE SQL TABLE DESIGN	181
FIGURE 4. 106 SQL DATABASE CONNECTING VISUAL STUDIO TEST CONNECTION.....	182
FIGURE 4. 107 DEBUGGING FLAWS OF THE CODE THROUGH VISUAL STUDIO IDE	184
FIGURE 4. 108 STEPS INVOLVED IN DEBUGGING PROCESS	185
FIGURE 4. 109 C# E CODE WITH THE HAVING UNEXPECTED CONSOLE OUTPUT.....	189
FIGURE 4. 110 BREAKPOINT AT THE 1ST LINE OF THE CODE	190
FIGURE 4. 111 SMALLEST VARIABLE STORED VALUES	191

FIGURE 4. 112 MIN VARIABLE STORED VALUES	191
FIGURE 4. 113 LIST VARIABLE STORED VALUES	192
FIGURE 4. 114 WATCH 1 WINDOWS TAB OPEN BY VISUAL STUDIO IDE	192
FIGURE 4. 115 MONITOR INTERESTED VARIABLE VALUES BY WATCH TAB.....	193
FIGURE 4. 116 FIXING THE BUG	193
FIGURE 4. 117 MIN VARIABLE VALUE AFTER DEBUGGING.....	194
FIGURE 4. 118 EXPECTED OUTPUT AFTER DEBUGGING.....	194
FIGURE 4. 119 BASIC CALCULATOR EXAMPLE CODE WITH THE GIVEN PARAMETERS FOR USER INPUTS.	195
FIGURE 4. 120 EXCEPTION OCCUR WHEN RUN THE PROGRAM.....	195
FIGURE 4. 121 HANDLE EXCEPTIONS BY TRY AND CATCH BLOCKS	196
FIGURE 4. 122 USING SIMPLE CODE TO UNDERSTAND – PART OF THE AYUBO SYSTEM.....	197
FIGURE 4. 123 USING CONSISTENCY OF NAMING CONVENTION OF THE VARIABLES – PART OF THE AYUBO SYSTEM.....	198
FIGURE 4. 124 NAMING THE FUNCTION ACCORDING TO WHAT THEY PERFORM – PART OF THE AYUBO SYSTEM.....	199
FIGURE 4. 125 COMMENTING ON THE LINES OF IMPORTANT PARTS OF THE PROJECT...	200

List of Tables

TABLE 1. 1 DIFFERENCES BETWEEN ALGORITHM AND FLOWCHART. 30

TABLE 2. 1 RELATIONSHIPS AMONG PROCEDURAL, OBJECT ORIENTED AND EVENT-DRIVEN PARADIGMS..... 83

TABLE OF CONTENTS

ACTIVITY 1	26
1.1 ALGORITHMS	26
1.1.1 <i>What is an Algorithm</i>	26
1.1.2 <i>Characteristics of a good algorithm</i>	27
1.1.3 <i>Advantages and disadvantages of an Algorithm</i>	29
1.1.4 <i>Difference between Algorithm and Flowchart</i>	30
1.1.5 <i>Relationship between the written algorithm and the code variant</i>	31
1.2 THE STEPS TAKEN FROM WRITING CODE TO EXECUTION.....	32
1.2.1 <i>The Programming Process</i>	32
1.2.2 <i>The Programming Process Step 01: Defining the problem</i>	32
1.2.3 <i>The Programming Process Step 02: Planning the solution</i>	32
1.2.4 <i>The Programming Process Step 03: Coding the program</i>	33
1.2.5 <i>The Programming Process Step 04: Testing the program</i>	33
1.2.6 <i>The Programming Process Step 05: Documenting the program</i>	34
1.3 ASYMPTOTIC ANALYSIS: BIG-O NOTATION.....	35
1.3.1 <i>What is Asymptotic Analysis</i>	35
1.3.2 <i>Big O notation</i>	36
1.3.3 <i>Calculating time complexity</i>	39
1.3.4 <i>Finding out the time complexity for simple operations</i>	40
1.3.5 <i>Finding out the time complexity for Single Loop</i>	41
1.3.6 <i>Finding out the time complexity for Nested Loops</i>	42
1.3.7 <i>Finding out the time complexity for Sequential Statements</i>	43
1.3.8 <i>Finding out the time complexity for Conditional Statements</i>	44
1.3.9 <i>Comparison of time complexities</i>	45
1.4 ALGORITHM FOR FIBONACCI SERIES	46
1.4.1 <i>What is Fibonacci series</i>	46
1.4.2 <i>Algorithm to display Fibonacci series for a given number using Pseudo code</i>	47
1.4.3 <i>Dry run the Fibonacci series algorithm by using pseudocode</i>	48
1.4.4 <i>Dry run the above Fibonacci series algorithm by using Python</i>	50
1.4.5 <i>Evaluation efficiency of Fibonacci series algorithm using Big-O notation</i>	53
1.5 ALGORITHM FOR FACTORIAL VALUE.....	55
1.5.1 <i>What is Factorial Value</i>	55
1.5.2 <i>Algorithm to display Factorial Value for a given number using Pseudo code</i>	56
1.5.3 <i>Dry run the above Factorial Value algorithm by using pseudocode</i>	57
1.5.4 <i>Dry run the above Factorial Value algorithm by using Python</i>	59
1.5.5 <i>Evaluation efficiency of Factorial Value algorithm using Big-O notation</i>	62
1.6 THE PROCESS IN BUILDING AN APPLICATION	64
Step 1: <i>Defining and analyzing the requirements</i>	64
Step 2: <i>Designing the application</i>	64
Step 3: <i>Developing the application</i>	65
Step 4: <i>Testing the application</i>	65
Step 5: <i>The application's deployment</i>	65

ACTIVITY 2 67

2.1 WHAT IS PROGRAMMING PARADIGM	67
2.1.1 <i>What is Programming Paradigm</i>	67
2.1.2 <i>Imperative Paradigm</i>	68
2.1.3 <i>Declarative Paradigm</i>	69
2.2 PROCEDURAL PROGRAMMING PARADIGMS.....	70
2.2.1 <i>Main characteristics of Procedural Programming Paradigms</i>	70
2.2.2 <i>Advantages and Disadvantages of Procedural Programming</i>	71
2.2.3 <i>Procedural Programming Paradigm example snippets of code</i>	72
2.3 OBJECT-ORIENTED PROGRAMMING PARADIGMS.....	75
2.3.1 <i>Main characteristics of Object-Oriented Programming Paradigms</i>	75
2.3.2 <i>Advantages and Disadvantages of Object-Oriented Programming</i>	76
2.3.3 <i>Object-Oriented Programming Paradigm example snippets of code</i>	77
2.4 EVENT-DRIVEN PROGRAMMING PARADIGMS	79
2.4.1 <i>Main characteristics of Event-Driven Programming Paradigms</i>	79
2.4.2 <i>Advantages and Disadvantages of Object-Oriented Programming</i>	80
2.4.3 <i>Event-Driven Programming Paradigm example snippets of code</i>	81
2.5 RELATIONSHIPS AMONG PROCEDURAL, OBJECT ORIENTED AND EVENT-DRIVEN PARADIGMS.....	83
2.6 BEST PROGRAMMING PARADIGM TO USE	84
2.6.1 <i>Advantages of Object-Oriented programming (OOP) and Event-Driven Programming (EDP) over pure Procedural Programming (PP)</i>	84
2.6.2 <i>Conclusion</i>	85

ACTIVITY 3 87

3.1 INTEGRATED DEVELOPMENT ENVIRONMENT (IDE)	87
3.1.1 <i>What is an Integrated Development Environment (IDE)</i>	87
3.1.2 <i>Common characteristics of an IDE</i>	88
3.1.3 <i>Benefits of IDE</i>	89
3.1.4 <i>Types of IDE</i>	90
3.1.5 <i>Best IDEs to use</i>	91
3.1.6 <i>The use of the Visual Studio IDE</i>	92
3.2 WHAT HAPPENS WHEN DEVELOPING A SYSTEM WITHOUT AN IDE?	93
3.2.1 <i>Problems usually occurs when developing a system without an IDE</i>	93
3.2.2 <i>Ways to developing a system without an IDE</i>	93
3.2.3 <i>Difference between IDE and Text Editor</i>	94
3.2.4 <i>Difference between IDE and Code Editor</i>	94
3.3 ALGORITHMS AND DESIGN SYSTEM FOR VEHICLE TARIFF CALCULATION FOR RENTS.....	95
3.3.1 <i>Flow Chart for Vehicle tariff calculation for rents</i>	95
3.3.2 <i>Visual Studio IDE interface to calculate vehicle rents</i>	96
3.3.3 <i>Visual Studio IDE coding to calculate vehicle rents</i>	97
3.3.4 <i>Create SQL table to store Vehicle Rent details</i>	99
3.3.5 <i>Create SQL table to bill Vehicle Rent details</i>	100
3.3.6 <i>Test Cases for vehicle Rent</i>	101
3.4 ALGORITHMS AND DESIGN SYSTEM FOR VEHICLE TARIFF CALCULATION FOR ONE DAY HIRES	104
3.4.1 <i>Flow Chart for Vehicle tariff calculation for One Day Hires</i>	104
3.4.2 <i>Visual Studio IDE interface to calculate vehicle One Day Hires</i>	105
3.4.3 <i>Visual Studio IDE coding to calculate vehicle One Day Hires</i>	106
3.4.4 <i>Create SQL table to store Vehicle One Day Hires details</i>	107
3.4.5 <i>Create SQL table to bill Vehicle One Day Hires details</i>	108

3.4.6 Test Cases for vehicle One Day Hires.....	109
3.5 ALGORITHMS AND DESIGN SYSTEM FOR VEHICLE TARIFF CALCULATION FOR LONG TOUR HIRES.....	112
3.5.1 Flow Chart for Vehicle tariff calculation for Long Tour Hires	112
3.5.2 Visual Studio IDE interface to calculate vehicle Long Tour Hires	113
3.5.3 Visual Studio IDE coding to calculate vehicle Long Tour Hires	114
3.5.4 Create SQL table to store Vehicle Long Tour Hires details	115
3.5.5 Create SQL table to bill Vehicle Long Tour Hires details.....	116
3.5.6 Test Cases for vehicle Rent.....	117
 ACTIVITY 4	 120
4.1 VISUAL STUDIO FORMS DESIGNS AND SYSTEM TO CALCULATE VEHICLE RENT AND HIRE AMOUNTS.....	120
4.1.1 Car rent system road map.....	120
4.1.2 Splash Screen Form Design and Code	121
4.1.3 Login Form Design and Code.....	122
4.1.4 Driver Signup Form Design and Code	125
4.1.5 Admin logged Dashboard Form Design and Code.....	127
4.1.6 Admin logged Vehicles Form Design and Code	132
4.1.7 Admin logged Rent Details Form Design and Code.....	136
4.1.8 Admin logged One Day Hire Details Form Design and Code	140
4.1.9 Admin logged Long Tour Hire Details Form Design and Code.....	144
4.1.10 Admin logged Driver Details Form Design and Code	148
4.1.11 Driver logged Rent Vehicle Form Design and Code	152
4.1.12 Driver logged One Day Hire Vehicle Form Design and Code	158
4.1.13 Driver logged Long Tour Hire Vehicle Form Design and Code.....	164
4.1.14 Driver logged My Profile Form Design and Code	169
4.1.15 Driver logged About Us Form Design and Code	172
4.2 MICROSOFT SQL TABLE DESIGNS AND QUERIES.....	174
4.2.1 Vehicle Details Table	174
4.2.2 Vehicle Renting Details Table.....	175
4.2.3 Vehicle One Day Hire Details Table	176
4.2.4 Vehicle Long Tour Hire Details Table.....	177
4.2.5 Driver Details Table	178
4.2.6 Vehicle Renting Bills Details Table	179
4.2.7 Vehicle One Day Hire Bills Details Table	180
4.2.8 Vehicle Long Tour Hire Bills Details Table	181
4.2.9 SQL Database connecting Visual Studio test connection	182
4.3 DEBUGGING.....	183
4.3.1 What is debugging in Visual studio IDE.....	183
4.3.2 Why do we need Debugging?	183
4.3.3 Debugging process in Visual studio IDE	184
4.3.4 Steps involved in Debugging process	185
4.3.5 Strategies to follow when debugging	187
4.4 HOW I USED THE DEBUGGING PROCESS TO DEVELOP MORE SECURE, ROBUST APPLICATION WITH EXAMPLES	188
4.4.1 How I used the debugging process.....	188
4.4.2 How I handled some little bugs of the program easily.....	195
4.5 CODING STANDARDS	197
4.5.1 Importance of coding standards	197
4.5.2 Coding standards to follow and how I followed them	197
4.5.3 Benefits of Implementing Coding Standards in Software Development	201

CONCLUSION	203
REFERENCES	204

Activity 1

1.1 Algorithms

1.1.1 What is an Algorithm

A set of instructions for solving a problem is referred to as an algorithm. The best approach to comprehend an algorithm is to consider it as a recipe, which contains detailed guidelines for producing a dish. To accomplish its operations, every computer device utilizes algorithms in the form of hardware or software-based routines. In computer programming terms, an algorithm is a series of well-defined instructions for solving a certain problem. It accepts a collection of inputs and produces a desired output.

The Google Hangouts use audio and video compression algorithm to transmit live video across the Internet so quickly. The Google Maps use route finding algorithm to figure out to get one location to another. The Pixar use rendering algorithm to color a 3D model of a character based on the lighting in a virtual room. The Nasa use an optimization and scheduling algorithm to arrange the solar panels on the International Space Station. Those algorithms are more complex than our everyday algorithms like adding two numbers. But they follow the same thing to accomplish a task.

1.1.2 Characteristics of a good algorithm

Algorithms are language independent. In other words, they are simply directions that may be executed in any language and produce the same result. Not all written programming instructions are algorithms. Some instructions must have the following features in order to be considered an algorithm. These are the characteristics that a successful algorithm should possess.

1) Unambiguous and clear:

The algorithm should be straightforward and clear. Each of its phases should be obvious in every way and should only lead to one meaning. Algorithms must specify each step in the process, as well as the sequence in which the steps must be completed. Definiteness refers to the order in which operations are performed to convert input into output. The algorithm should be simple and straightforward. Each step's specifics must also be specified (including how to handle errors). Everything in it should be quantitative, not qualitative. For example, we can't expect a machine to comprehend something if we're unsure about it.

2) Well-Defined Inputs:

If an algorithm requires inputs, the inputs should be well-defined. The input is the data that will be modified to produce the output during the computation. There should be 0 or more well-defined inputs in an algorithm. Input precision demands knowing what type of data, how much of it there should be, and in what format it should be.

3) Well-Defined Outputs:

The algorithm must clearly specify and well-define what outcome will be produced. The data obtained as a result of the computation is the output. A well-defined algorithm should have one or more outputs that match the desired output. Precision in output also demands knowing what type of data, how much, and in what format the output should be.

4) Finiteness:

The algorithm must have a limited number of steps. In other words, the algorithm should not wind up in infinite repetitions or something like. The algorithm will finally come to a complete stop. Stopping could indicate that we have received the intended result. Algorithms must come to a stop after a certain number of steps. An algorithm should never be endless and should always end after a set number of steps. It's pointless to create an infinite algorithm because it will be useless to us.

5) Feasible:

The algorithm has to be simple, generic, and practical, and it must be able to be executed using the given resources. It must not include any futuristic technologies or anything else. All of the steps needed to get to the output must be achievable with the given resources for an algorithm to be effective. It should not include any extra or redundant steps that could make the algorithm useless. For example, if we are preparing a recipe and we cut vegetables that will not be utilized in the recipe, we are wasting our time.

6) Language Independence:

The proposed Algorithm must be language independent. In other words, it must be simple instructions that can be implemented in any language and produce the expected results. Step-by-step instructions should be included in an algorithm, and they should be independent of any programming code. It should be written in such a way that it may be used with any programming language.

1.1.3 Advantages and disadvantages of an Algorithm

Advantages of Algorithms :

- Algorithms are simple to comprehend.
- An algorithm is a logical representation of a solution to a problem.
- The problem is divided into smaller bits or phases in Algorithm. As a result, it is easy for the programmer to translate it into a working program.

If we know something about current algorithms, we can save time and make our programs run faster by using the correct one. Some of the uses of algorithms today world as below.

New algorithms are constantly being developed in the biological sciences for applications such as constructing the molecular structures that are at the core of disease-fighting medicines. Algorithms can be used in physics to simulate climate and weather patterns. Algorithms are used to search and analyze the large amounts of data collected by automated space telescopes on stars in the universe. In many fields, and even in everyday life, efficient algorithms are required to analyze massive data sets or to wisely choose from a wide number of possible decisions.

Disadvantages of Algorithms :

- It takes a long time to write an algorithm, hence it is time-consuming.
- Algorithms make it a lot harder to demonstrate Branching and Looping statements.

1.1.4 Difference between Algorithm and Flowchart

Algorithms and flowcharts both help programmers in the same way, yet they are slightly different. As previously stated, an algorithm is a step-by-step procedure for solving a certain problem. A flowchart is a visual depiction of an algorithm that uses geometrical diagrams and symbols to demonstrate how it works.

Flowcharts are widely used by programmers as a problem-solving technique. It employs symbols that are linked together to represent the flow of information and processing. The process of drawing a flowchart for an algorithm known as “flowcharting”.

Below table represent some differences between Algorithm and Flowchart

Algorithm	Flowchart
This is a step-by-step procedure to solve a problem.	This is a diagram made up of several shapes that displays data flow.
The program's pseudo code is called an algorithm.	A flowchart is just a graphical representation of the logic of an algorithm.
Complex to understand.	Easy to understand.
Plain texts are used in algorithms.	Symbols and Shapes are used in flowchart.
Easy to debug.	Hard to debug.
Difficult to construct.	Simple to construct.
Doesn't follow any rules.	Follow rules to be constructed.

Table 1. 1 Differences between algorithm and flowchart.

1.1.5 Relationship between the written algorithm and the code variant

Mathematicians and programmers use algorithms to show and specify a set of steps and functions that must be followed in order to solve a problem. The sequence and these phases are both fixed; the only variables are the inputs. Algorithms can be used to solve anything from simple problems to complex functions, depending on the problem.

But in other hand code, often known as coding, is a set of instructions given to a machine or computer in such a way and language that it can understand and carry out the commands. These are usually in the form of high-level languages that machines can understand and translate.

Coding is a low-level programming language that is used to specify repeated functions like as controlling a device's display, validating transactions, and so on. Algorithms, on the other hand, are high-level and complicated systems capable of bringing projects, scientific curiosity, and research to life.

Algorithms are considered to be language independent and can be written in any language. If it fulfills the goal of ensuring steps for solving a problem, it will be considered an algorithm. But coding is dependent on language. If we are coding in HTML, we will be utilizing a defined set of terminology, commands, and instructions, but if we are working in C++, everything will be completely different. We are unable to write computer programming programs in any language.

1.2 The steps taken from writing code to execution

1.2.1 The Programming Process

Developing a software requires the same processes as any other problem-solving activity. The programming process is made up of five steps as followings.

1. Defining the problem
2. Planning the solution
3. Coding the program
4. Testing the program
5. Documenting the program

1.2.2 The Programming Process Step 01: Defining the problem

To study the problem, we meet with users from the client organization, or we meet with a systems analyst who describes the project. The task of describing the problem involves deciding what we already know (input-the data) and what we want to obtain (output-the result). Eventually, we establish a written agreement that describes the type of input, processing, and output required, among other things. This is not an easy task.

1.2.3 The Programming Process Step 02: Planning the solution

Drawing a flowchart and writing pseudocode, or maybe both, are two typical methods for preparing a problem's solution. A flowchart is basically a visual representation of a step-by-step solution to a problem. It is made of arrows that show the program's direction and boxes and other symbols that represent activities. It's a road map for what our program will accomplish and how it will accomplish it.

Pseudocode is a nonstandard English-like language that allows us to express our solution with greater detail than plain English. This requiring less precision than a formal programming language. Pseudocode allows us to focus on the program logic without having to worry about the specific syntax of a programming language just yet. Pseudocode, on the other hand, is not computer executable.

1.2.4 The Programming Process Step 03: Coding the program

The next stage for us as programmers is to code the program or express our solution in a programming language. We'll convert the logic from the flowchart to a programming language using pseudocode or another tool. A programming language is a set of rules for guiding the computer on what operations to perform. To make any program operate, we must strictly abide by the rule's syntax of the programming language.

1.2.5 The Programming Process Step 04: Testing the program

Some experts believe that if a program is well-designed, it can be developed right the first time. Most programmers, on the other hand, have become used to the fact that their freshly built programs are likely to include a few bugs. This can be discouraging at first, because programmers are typically accurate, careful, and detail-oriented people who take pride in their work. We must eventually be ready to test the software on the computer after we have finished developing it. These 2 phases are included in this process.

- Desk Checking :- The process of manually reviewing a program's source code is known as desk checking. It involves reading through the code and manually checking each function, frequently with different input values.

- Debugging :- Debugging is the process of finding and removing errors in software code that can cause it to act incorrectly or crash. The software will be ready to use after the bug has been repaired.

1.2.6 The Programming Process Step 05: Documenting the program

A written complete description of the work - flow and specific details about the program is known as documentation. Logic tools such as flowcharts and pseudocode, data-record descriptions, program listings, and testing results are typical program documentation resources. Comments within the program are also considered to be an important part of the documentation. Many programmers keep track of their work as they code. Program documentation, in a broader sense, might be included in the documentation for a whole system.

1.3 Asymptotic Analysis: Big-O Notation

1.3.1 What is Asymptotic Analysis

The amount of time, storage, and other resources required to execute an algorithm affects its efficiency. Asymptotic Notations are used to calculate efficiency. An algorithm's performance may differ depending on the type of input. The performance will vary as the input size increases. Asymptotic analysis is the study of how the algorithm's performance changes as the order of the input size changes.

As described, Asymptotic Notations are mathematical notations that are used to express the running duration of an algorithm when the input moves toward a specific value or a limiting value. For example, when an input array is already sorted, the algorithm takes linear time, which is the best scenario. However, when the input array is in reverse order, the algorithm takes the longest (quadratic) time to sort the items, which is the worst scenario. It takes average time when the input array is not sorted or in reverse order. Hence, Asymptotic notations are used to represent these durations.

There are three types of asymptotic notations:

1. Big-O notation → The upper bound of an algorithm's execution time is represented by the Big-O notation. As a result, it determines an algorithm's worst-case complexity.
2. Omega notation → The lower bound of an algorithm's running time is represented by the omega notation. As a result, it gives the algorithm's best-case complexity.
3. Theta notation → The function from above and below is enclosed in theta notation. It is used to analyze an algorithm's average-case complexity since it represents the upper and lower bounds of the algorithm's execution time.

1.3.2 Big O notation

In computer science, the Big O notation is used to express the performance or complexity of an algorithm. Big O is a term that describes the worst-case scenario and can be used to express the amount of time an algorithm takes to execute or the amount of space it takes up in memory or on disk.

As a programmer, the best approach to fully comprehend Big O is to use code examples. So, below are some popular growth orders, along with descriptions and examples with Python algorithms.

O(1) → Constant Time

O(1) also known as " constant time". Which means the run time of the algorithm takes the same amount of time to run regardless of the size of the input. Addition, subtraction and the majority of fundamental search operations are all assumed to run at constant time.

Below figure is an algorithm of reading index of an array by Python. When considering O(1) where “1” is the number of times, we want it to execute. No matter how long the array is, the algorithm runs 1 time when execute.

Python code:

```
my_List = ["Ryan", "Dexter", "Yagi"]
print (my_List[0])
```

Output :



Figure 1. 1 Example Python code and output for demonstrate Big O notation “constant time”

O(n) → Linear Time

O(n) also known as "linear time". Which means the run time of the algorithm grows at the same pace as the input. For loops are an excellent example of a linear time operation.

Below figure is an algorithm of “for loop” by Python. When considering O(n) where “n” is the number of times, we want it to execute. According to below algorithm as we set range(5), it runs the algorithm 5 times which result printing 5 times of “Hello World”. If we set range(100), it runs the algorithm 100 times which result printing 100 times of “Hello World”.

Python code:

```
for i in range(5):
    print("Hello World")
```

Output :

		C:\Users\ryan\
		Hello World

Figure 1.2 Example Python code and output for demonstrate Big O notation “linear time”

$O(n^2) \rightarrow$ Quadratic Time

$O(n^2)$ also known as "Quadratic time". Which means that the calculation takes the squared size of the input data to perform. Nested For loops are an excellent example of a Quadratic time operation.

Below figure is an algorithm of “nested for loops” by Python. According to below algorithm as we set range(3) in outer loop, it runs 3 times. And as we set range(2) in inner loop, it runs 2 times for each iteration of the outer loop. As a result, this function takes $O(n^2)$ times to perform.

Python code:

```
for i in range(3):
    for j in range(2):
        print("Hello World")
```

Output :

```
C:\Users\rya
Hello World
Hello World
Hello World
Hello World
Hello World
Hello World
```

Figure 1. 3 Example Python code and output for demonstrate Big O notation “quadratic time”

1.3.3 Calculating time complexity

Knowing the Big O Notation, how it's computed, and what constitutes an acceptable time complexity for an algorithm will give us an edge over competitors when it comes to job seeking. One thing to be noted is that time complexity is analyzed for very large input sizes and worst-case scenario.

Big O can be calculated using the following five steps:

- Divide our algorithm or function into separate operations.
- Determine each operation's Big O value.
- Add all of the operations' Big Os together.
- Get rid of the variables.
- Find the highest order word – this is what our algorithm/function will be called the Big O.

Let's start easy and compute the Big O of a simple function with this in mind.

I'll take a polynomial example and try to calculate its time complexity.

$$T(n) = 2n^2 + 3n + 1$$

First step is to drop the lower order terms. That are $3n + 1$ in this case.

$$T(n) = 2n^2 + 3n + 1$$

Next step is to drop all the constant multipliers. Hence, I can drop the constant that is 2.

$$T(n) = 2n^2 + 3n + 1$$

So, I can get the time complexity as big O of n^2 as following.

Time complexity → $O(n^2)$

Now let's look at some rules for finding out the time complexity in algorithms.

1.3.4 Finding out the time complexity for simple operations

First let us look at how do we calculate the time complexity of a simple calculation.

```
total = num1 + num2
print (total)
```

Figure 1. 4 Finding time complexity for simple operations sample question

Whatever the inputs are, this procedure will take the same amount of time to complete.

Hence, time complexity will be constant for this operation which is a constant taken as “C”.

Therefore, time complexity will come out as C times. $\rightarrow C$

But with Big O terms, we can neglect the constant “C”. $\rightarrow C$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(1)$

<pre>total = num1 + num2 print (total)</pre>	Constant time = C	Time complexity = C $= C$
--	--------------------------	-------------------------------------

Time complexity = O(1)

Figure 1. 5 Finding time complexity for simple operations answer for the question

1.3.5 Finding out the time complexity for Single Loop

First let us look at how do we calculate the time complexity of a single loop.

```
for (i = 1; i <= n; i++)
{
    ...
    a = 1+ 2
}
```

Figure 1. 6 Finding out the time complexity for Single Loop sample question

So above represent is a “for loop” which runs from 1 to n. “n” could be any number. Then there is simple mathematical operation inside the for loop.

The operation inside the for loop takes constant time to execute. Hence, time complexity will be n times time is taken for this operation which is a constant taken as “C”.

Therefore, time complexity will come out as C times n. $\rightarrow C n$

But with Big O terms, we can neglect the constant “C”. $\rightarrow O(n)$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(n)$

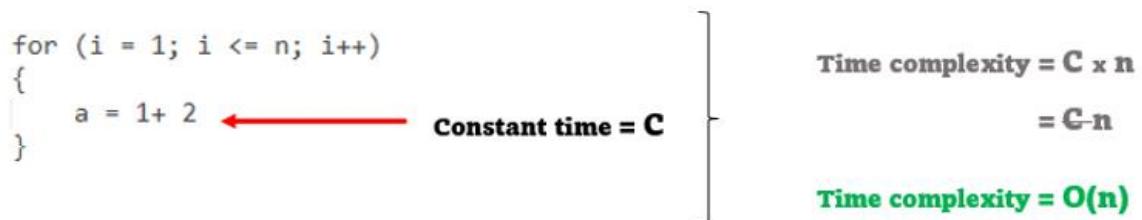


Figure 1. 7 Finding out the time complexity for Single Loop answer for the question

1.3.6 Finding out the time complexity for Nested Loops

Now let's calculate the time complexity of a nested loops.

```

for (i = 1; i <= n; i++)
{
    for (k = 1; k <= n; k++)
    {
        a = 1+ 2
    }
}

```

Figure 1. 8 Finding out the time complexity for Nested Loops sample question

As we can see, I have represented 2 loops which are nested. As I did before the operation inside of the inner for loop takes constant time to execute. Hence, time complexity will be n times time is taken for this operation which is a constant taken as “C”. Now I can simply multiply the time complexity of each loop. Hence, outer loop runs n times. For each run of the outer loop, the inner loop will also run n times.

So, the total running time will be Constant “C” time multiplied by into n into n . $\rightarrow C n^2$

But with Big O terms, we can neglect the constant “C”. $\rightarrow O(n^2)$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(n^2)$

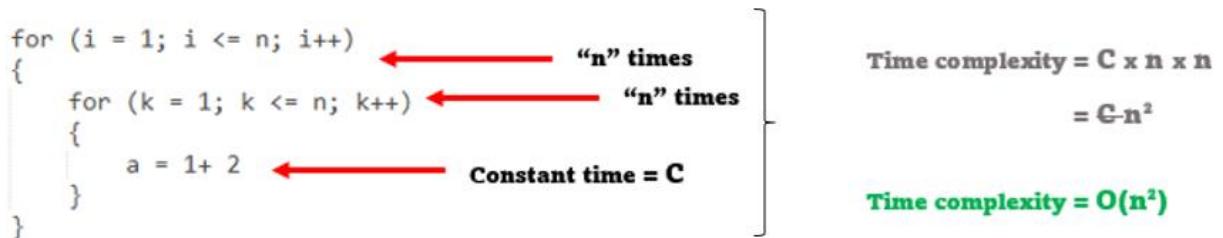


Figure 1. 9 Finding out the time complexity for Nested Loops answer for the question

1.3.7 Finding out the time complexity for Sequential Statements

Now let's calculate the time complexity of a nested loops.

```
a = 1 + 2

for (i = 1; i <= n; i++)
{
    print ("Hello")

    for (k = 1; k <= n; k++)
    {
        print ("Hello World")
    }
}
```

Figure 1. 10 Finding out the time complexity for Sequential Statements sample question

The first operation of above expression takes constant time to run. $\rightarrow C_1$.

After that we have this “for loop”. As I explained before, for loop will take C into “ n ” times to run. $\rightarrow C_2 n$

Then we have another for loop which takes C_3 into n times to run. $\rightarrow C_3 n$

So, when we have sequential statements like these, we simply add the time taken for each statement. $\rightarrow C_1 + C_2 n + C_3 n$

As I told before in Big-O terms, we can ignore constants which are C_1 , C_2 and C_3 .

$\rightarrow C_1 + C_2 n + C_3 n$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(n)$

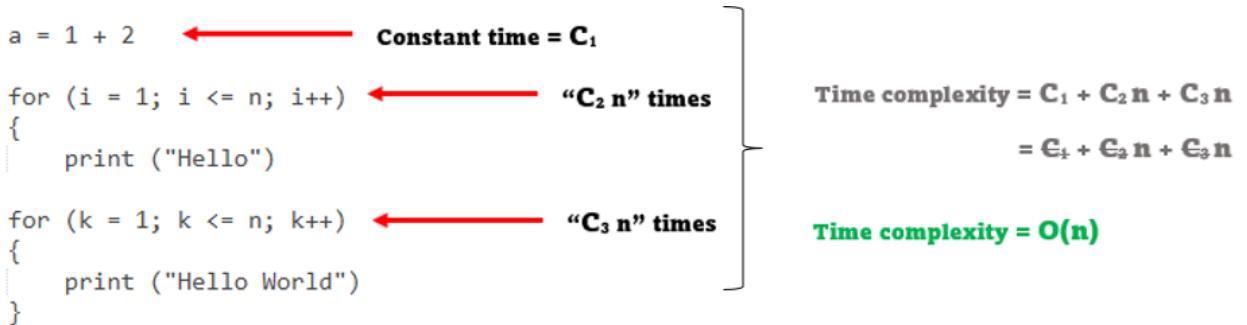


Figure 1. 11 Finding out the time complexity for Sequential Statements answer for the question

1.3.8 Finding out the time complexity for Conditional Statements

In conditional statements such as “if else” conditions, we evaluate each situation.

```

if
{
    //for loop
}

else
{
    //nested for loop
}

```

Figure 1. 12 Finding out the time complexity for Conditional Statements sample question

Let's say inside if condition we have a for loop whose time complexity is $\rightarrow O(n)$

And let's say inside else condition we have a nested for loop whose time complexity is $\rightarrow O(n^2)$

We can consider the else condition as it has a greater time complexity.

Hence the overall time complexity will come out to be $\rightarrow O(n^2)$.

```

if
{
    //for loop
}
else
{
    //nested for loop
}

```

← “ $C_1 n$ ” times
 ← “ $C_2 n^2$ ” times

Time complexity = $C_1 n + C_2 n^2$

= Since $C_1 n < C_2 n^2$

Time complexity = $O(n^2)$

Figure 1. 13 Finding out the time complexity for Conditional Statements answer for the question

1.3.9 Comparison of time complexities

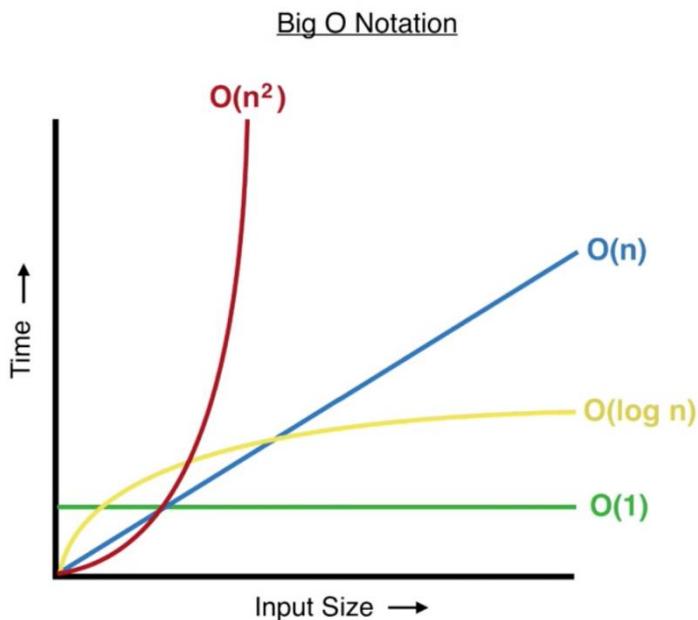


Figure 1. 14 Time Complexities Comparison Graph

According to above figure we can visually observe how the time increases as the amount of the input grows larger. And as we know time is proportional for number of operations. This will give us an idea of what an efficient runtime for more basic algorithms looks like.

$O(1)$ → Big O of 1 is the smallest and the best.

Then Respectively $O(\log n) < O(n) < O(n^2)$

$O(n!)$ → Big O of n factorial is the biggest and the worst.

1.4 Algorithm for Fibonacci series

1.4.1 What is Fibonacci series

One of the most well-known mathematical formulas is the Fibonacci sequence. The sum of the two numbers before it determines the next number in the sequence.

Hence the order of the sequence as follows : 0, 1, 1, 2, 3, 5, 8, 13, 21, 34, and so forth.

The mathematical equation describing it is → $F_n = F_{n-1} + F_{n-2}$

$n =$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	...
$x_n =$	0	1	1	2	3	5	8	13	21	34	55	89	144	233	377	...

Figure 1. 15 Fibonacci series

By adding the two numbers before it, the following number is found:

- The 2 is obtained by multiplying the two numbers preceding it. (1+1)
- The 3 is obtained by multiplying the two numbers preceding it. (1+2)
- The 5 is obtained by multiplying the two numbers preceding it. (2+3)

1.4.2 Algorithm to display Fibonacci series for a given number using Pseudo code

STEP 1 : START

STEP 2 : Read nth value from user and store it as integer in “n” variable

STEP 3 : if $n < 0$ ask positive number and go to STEP 10 else go to STEP 4

STEP 4 : Declare following variables.

*Firstnum = 0 (string variable)

*Secondnum = 1 (string variable)

STEP 5 : Declare an array to store Fibonacci numbers

My_List = []

STEP 6 : Use “for loop” for following steps until $i = n + 1$ else go to STEP 7

STEP 6.1 : Append Firstnum to My_List

STEP 6.2 : Convert Firstnum and Secondnum string variables into integer

STEP 6.3 : Next = Firstnum + Secondnum

STEP 6.4 : Firstnum = Secondnum

STEP 6.5 : Secondnum = Next

STEP 6.6 : Convert Firstnum integer variable into string

STEP 6.7 : $i = i+1$

STEP 6.8 : Repeat STEP 6 until $i = n + 1$

STEP 7 : print My_List to show Fibonacci series until to nth value

STEP 8 : Make reverse the My_List elements

STEP 9 : print 1st element of the My_List to show nth value of Fibonacci series

STEP 10 : STOP

1.4.3 Dry run the Fibonacci series algorithm by using pseudocode

The sample number used for Dry Run is number 3

STEP 1 : START

STEP 2 : 3 (Sample number which reading by the user)

STEP 3 : since $3 > 0$ go to step 4

STEP 4 : Declare following variables.

*Firstnum = 0 (string variable)

*Secondnum = 1 (string variable)

STEP 5 : Declare an array to store Fibonacci numbers

My_List = []

STEP 6 : Use “for loop” for following steps until $i = 4$ else go to STEP 7

STEP 6.1 : Append Firstnum = 0 to My_List → [0]

STEP 6.2 : Convert Firstnum = 0 and Secondnum = 1 variables into integer

STEP 6.3 : Next = Firstnum + Secondnum = $0 + 1$

STEP 6.4 : Firstnum = Secondnum = 1

STEP 6.5 : Secondnum = Next = 1

STEP 6.7 : Convert Firstnum = 1 into string

STEP 6.8 : $i = 0 + 1 = 1$

STEP 6.9 : Repeat STEP 6 until $i = 4$

STEP 6.10 : Append Firstnum = 1 to My_List → [0,1]

STEP 6.12 : Convert Firstnum = 1 and Secondnum = 1 variables into integer

STEP 6.13 : Next = $1 + 1$

STEP 6.14 : Firstnum = Secondnum = 1

STEP 6.15 : Secondnum = Next = 2

STEP 6.17 : Convert Firstnum = 1 into string

STEP 6.18 : $i = 1 + 1 = 2$

STEP 6.19 : Repeat STEP 6 until $i = 4$

- STEP 6.20 : Append Firstnum = 1 to My_List → [0,1,1]
STEP 6.21 : Convert Firstnum = 1 and Secondnum = 2 variables into integer
STEP 6.22 : Next = Firstnum + Secondnum = 1 + 2
STEP 6.23 : Firstnum = Secondnum = 2
STEP 6.24 : Secondnum = Next = 3
STEP 6.25 : Convert Firstnum = 2 into string
STEP 6.26 : i = 2 + 1 = 3
STEP 6.27 : Repeat STEP 6 until i = 4
- STEP 6.28 : Append Firstnum = 2 to My_List → [0,1,1,2]
STEP 6.29 : Convert Firstnum = 2 and Secondnum = 3 variables into integer
STEP 6.30 : Next = Firstnum + Secondnum = 2 + 3
STEP 6.31 : Firstnum = Secondnum = 3
STEP 6.32 : Secondnum = Next = 5
STEP 6.33 : Convert Firstnum = 3 into string
STEP 6.34 : i = 3 + 1 = 4
STEP 6.35 : Since i (4) = 4 Go to STEP 7

- STEP 7 : print My_List → [0,1,1,2] to show Fibonacci series until to nth value
STEP 8 : Make reverse the My_List → [2,1,1,0] elements
STEP 9 : print 1st element of My_List → “2” to show nth value of Fibonacci series
STEP 10 : STOP

Output is → [0,1,1,2] with 2

1.4.4 Dry run the above Fibonacci series algorithm by using Python

The screenshot shows a code editor window with multiple tabs at the top: Fibonacci 1.py, Fibonacci 2.py (which is the active tab), main.py, Test.py, Fibonacci 3.py, and Fibonacci 4.py. The code in the editor is a Python script for generating a Fibonacci sequence. It starts by importing the `input` function from the `builtins` module. It then prompts the user to enter a value for the nth Fibonacci number. If the input is less than 0, it prints an error message. Otherwise, it initializes the first two numbers of the sequence (0 and 1) and a list to store the sequence. It then enters a loop from index 2 to n, where it calculates the next number by summing the previous two, appends it to the list, and updates the previous two numbers. Finally, it prints the entire sequence and the nth value.

```
1 # Importing the input function from the builtins module
2 import builtins
3 
4 n = int(builtins.input("Enter a value (nth value) to display nth Fibonacci number: "))
5 
6 if n < 0:
7     print("Please enter positive integer")
8 
9 else:
10    firstnum = "0"
11    secondnum = "1"
12    my_list = []
13 
14    for i in range(n+1):
15        firstnum = str(firstnum)
16        my_list.append(firstnum)
17 
18        firstnum = int(firstnum)
19        secondnum = int(secondnum)
20        next = firstnum + secondnum
21        firstnum = secondnum
22        secondnum = next
23 
24    #Display the fibonacci sequence
25    print("Fibonacci Sequence is: ", " , ".join(my_list))
26 
27    #Display nth value
28    my_list.reverse()
29    fib = int(my_list[0])
30    print("nth value is : ", fib)
```

Figure 1. 16 Fibonacci series algorithm by using Python

Output →

The sample number used for Dry Run is number 3

Run: Fibonacci 2

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python.exe "
Enter a value (nth value) to display nth Fibonacci number: 3
Fibonacci Sequence is:  0 , 1 , 1 , 2
nth value is :  2
```

Figure 1. 17 Fibonacci series Dry Run taking sample number 3

The sample number used for Dry Run is number 2

Run: Fibonacci 2

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python.exe "
Enter a value (nth value) to display nth Fibonacci number: 2
Fibonacci Sequence is:  0 , 1 , 1
nth value is :  1
```

Figure 1. 18 Fibonacci series Dry Run taking sample number 2

The sample number used for Dry Run is number 1

Run: Fibonacci 2

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python.exe "
Enter a value (nth value) to display nth Fibonacci number: 1
Fibonacci Sequence is:  0 , 1
nth value is :  1
```

Figure 1. 19 Fibonacci series Dry Run taking sample number 1

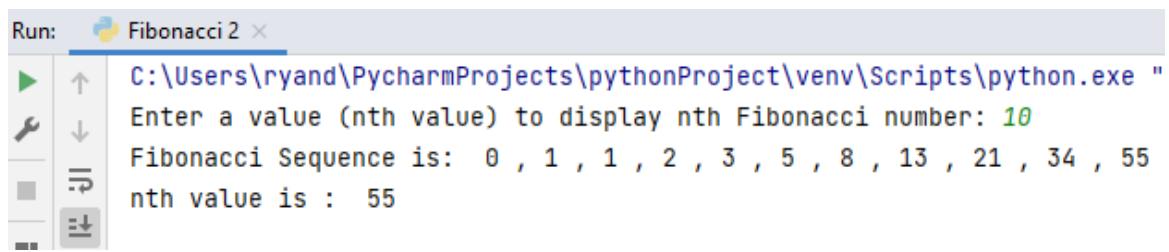
The sample number used for Dry Run is number 0

Run: Fibonacci 2

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python.exe "
Enter a value (nth value) to display nth Fibonacci number: 0
Fibonacci Sequence is:  0
nth value is :  0
```

Figure 1. 20 Fibonacci series Dry Run taking sample number 0

The sample number used for Dry Run is number 10



A screenshot of a PyCharm interface titled "Fibonacci 2". The code window shows a Python script with the following content:

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python.exe "
Enter a value (nth value) to display nth Fibonacci number: 10
Fibonacci Sequence is:  0 , 1 , 1 , 2 , 3 , 5 , 8 , 13 , 21 , 34 , 55
nth value is :  55
```

Figure 1. 21 Fibonacci series Dry Run taking sample number 10

1.4.5 Evaluation efficiency of Fibonacci series algorithm using Big-O notation

```

n = int(input("Enter a value (nth value) to display nth Fibonacci number: "))

if n < 0:
    print("Please enter positive integer") ← Constant time = C1

else:
    firstnum = "0"
    secondnum = "1"
    my_list = []

    for i in range(n+1):
        firstnum = str(firstnum)
        my_list.append(firstnum)

        firstnum = int(firstnum)
        secondnum = int(secondnum)
        next = firstnum + secondnum
        firstnum = secondnum
        secondnum = next

#Display the fibonacci sequence
print("Fibonacci Sequence is: " , " , ".join(my_list)) ← Constant time = C3

#Display nth value
my_list.reverse()
fib = int(my_list[0])
print("nth value is : " ,fib) ← Constant time = C4
    
```

$$\begin{aligned}
 \text{Time complexity} &= C_1 + C_2 n + C_3 + C_4 \\
 &= C_1 + C_2 n + C_3 + C_4
 \end{aligned}$$

Time complexity = O(n)

Figure 1. 22 Evaluation efficiency of Fibonacci series algorithm using Big-O notation

Let's Evaluation efficiency of Fibonacci series algorithm using Big-O notation. As we learned to find out the time complexity for Conditional Statements, lets evaluate each situation.

The first operation in the if condition takes constant time to run. $\rightarrow C_1$.

After in else condition we have this "for loop". As I explained in big O notation chapter, for loop will take C into " n " times to run. $\rightarrow C_2 n$

Then after Conditional Statements there is a execute operation with an in-built function which takes constant time to run. $\rightarrow C_3$.

The time complexity of accessing the value of an element at a certain index is constant. Whether the list has 5 or 500 elements, the complexity of retrieving the value at index 0 takes constant time to run. This is because, regardless of how many elements are in the array, the operations needed to access an element in memory remain constant.

Hence, despite of having in built function, the final part of the algorithm is fundamental accessing element of an array operation which takes also constant time to run. $\rightarrow C_4$.

So, when we have sequential statements like these, we simply add the time taken for each statement.

$$\rightarrow C_1 + C_2 n + C_3 + C_4$$

As I told before in Big-O terms, we can ignore constants which are C_1, C_2, C_3 and C_4 .

$$\rightarrow \epsilon_1 + \epsilon_2 n + \epsilon_3 + \epsilon_4$$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(n)$

1.5 Algorithm for Factorial Value

1.5.1 What is Factorial Value

The factorial function (symbol is !) tells us to multiply all whole integers from 1 to our selected value.

n	n!		
1	1	= 1	= 1
2	2×1	$= 2 \times 1!$	= 2
3	$3 \times 2 \times 1$	$= 3 \times 2!$	= 6
4	$4 \times 3 \times 2 \times 1$	$= 4 \times 3!$	= 24
5	$5 \times 4 \times 3 \times 2 \times 1$	$= 5 \times 4!$	= 120

Figure 1. 23 Factorial Value

The mathematical equation describing it is $\rightarrow n! = n \times (n-1)!$

1.5.2 Algorithm to display Factorial Value for a given number using Pseudo code.

STEP 1 : START

STEP 2 : Read nth value from user and store it as integer in “n” variable

STEP 3 : Declare following variables.

*Count = n (integer variable)

*Fac = 1 ((integer variable)

STEP 4 : Declare an array to store Factorial numbers

My_List = []

STEP 5 : Use “while loop” for following steps until count > 0 else go to STEP 6

STEP 5.1 : Convert Count integer variables into string

STEP 5.2 : Append Count to My_List

STEP 5.3 : Convert Count string variables into integer

STEP 5.4 : Count = Count -1

STEP 5.5 : Repeat STEP 5 until count = 1

STEP 6 : Use “while loop” for following steps until n > 0 else go to STEP7

STEP 6.1 : fac = fac * n

STEP 6.2 : n = n -1

STEP 6.3 : Repeat STEP 6 until n = 1

STEP 7 : print My_List to show Factorial Numbers until to nth value

STEP 8 : print fac to show the value of Factorial Numbers series

STEP 9 : STOP

1.5.3 Dry run the above Factorial Value algorithm by using pseudocode

STEP 1 : START

STEP 2 : 3 (Sample number which reading by the user)

STEP 3 : Declare following variables.

*Count = n = 3 (integer variable)

*Fac = 1 ((integer variable))

STEP 4 : Declare an array to store Factorial numbers

My_List = []

STEP 5 : Use “while loop” for following steps until Count (3) > 0 else go to STEP 6

STEP 5.1 : Convert Count = 3 integer variables into string

STEP 5.2 : Append Count (3) to My_List → [3]

STEP 5.3 : Convert Count = 3 string variables into integer

STEP 5.4 : Count = Count -1 → Count = 3 -1 = 2

STEP 5.5 : Repeat STEP 5 until Count (2) = 0

STEP 5.6 : Convert Count = 2 integer variables into string

STEP 5.7 : Append Count (2) to My_List → [3,2]

STEP 5.8 : Convert Count = 2 string variables into integer

STEP 5.9 : Count = Count -1 → Count = 2 -1 = 1

STEP 5.10 : Repeat STEP 5 until Count (1) = 0

STEP 5.11 : Convert Count = 1 integer variables into string

STEP 5.12 : Append Count (1) to My_List → [3,2,1]

STEP 5.13 : Convert Count = 1 string variables into integer

STEP 5.14 : Count = Count -1 → Count = 1 -1 = 0

STEP 5.15 : Since Count (0) = 0 go to STEP 6

STEP 6 : Use “while loop” for following steps until n (3) > 0 else go to STEP7

STEP 6.1 : fac = fac * n → fac = 1 * 3 = 3

STEP 6.2 : n = n -1 → n = 3 - 1 = 2

STEP 6.3 : Repeat STEP 6 until n (2) = 0

STEP 6.4 : fac = fac * n → fac = 3 * 2

STEP 6.5 : n = n -1 → n = 2 - 1 = 1

STEP 6.6 : Repeat STEP 6 until n (1) = 0

STEP 6.7 : fac = fac * n → fac = 6 * 1

STEP 6.8 : n = n -1 → n = 1 - 1 = 0

STEP 6.9 : Since n (0) = 0 go to STEP 7

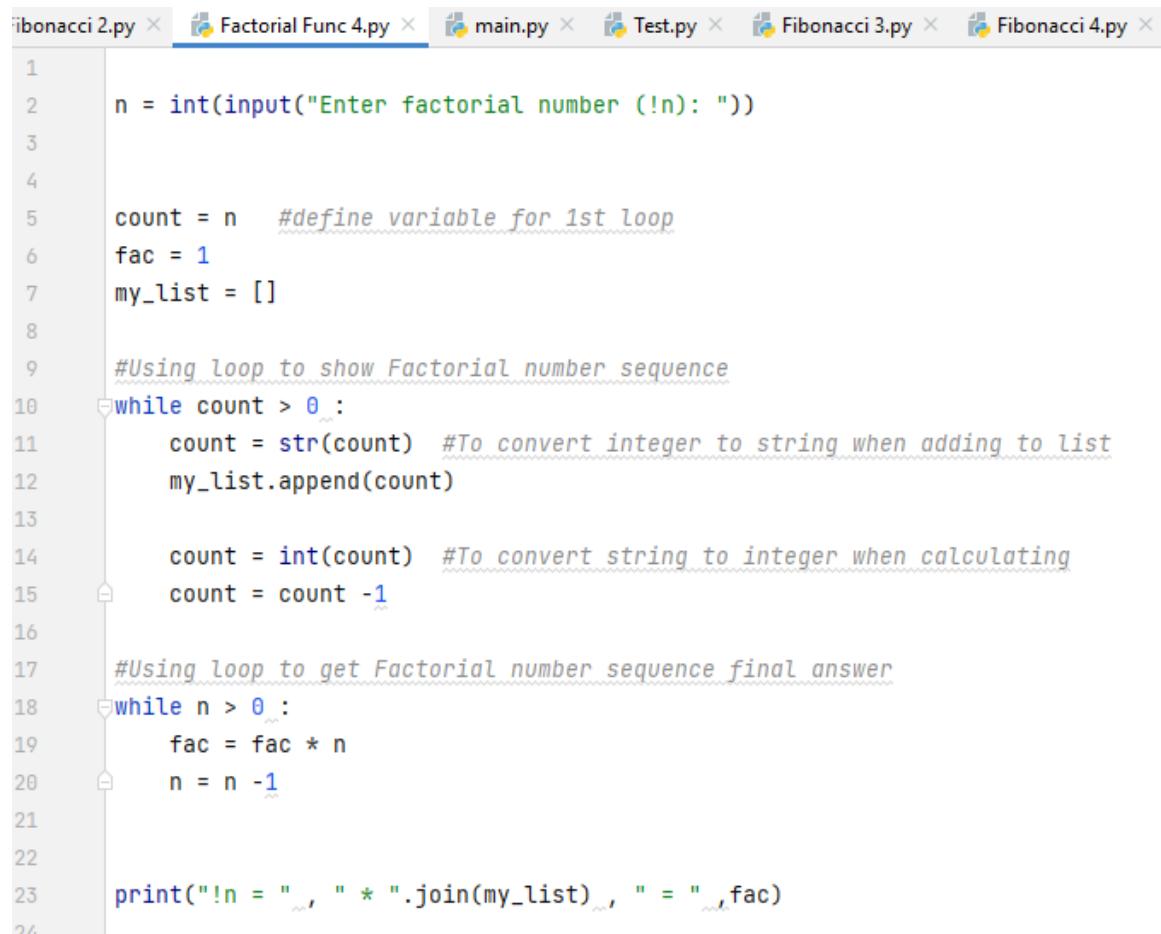
STEP 7 : print My_List → [3,2,1] to show Factorial Numbers until to nth value

STEP 8 : print fac → “6” to show the value of Factorial Numbers series

STEP 9 : STOP

Output is → [3,2,1] with 6

1.5.4 Dry run the above Factorial Value algorithm by using Python

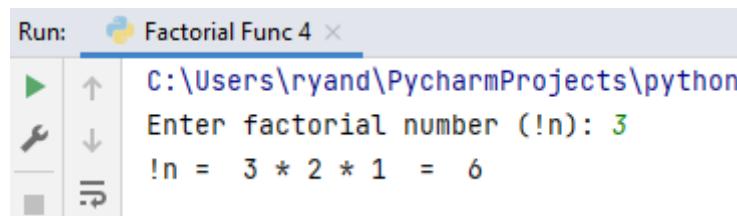


```
1 n = int(input("Enter factorial number (!n): "))
2
3
4 count = n      #define variable for 1st loop
5 fac = 1
6 my_list = []
7
8 #Using loop to show Factorial number sequence
9 while count > 0 :
10     count = str(count) #To convert integer to string when adding to list
11     my_list.append(count)
12
13     count = int(count) #To convert string to integer when calculating
14     count = count -1
15
16 #Using loop to get Factorial number sequence final answer
17 while n > 0 :
18     fac = fac * n
19     n = n -1
20
21
22 print("!n = " , " * ".join(my_list) , " = " ,fac)
23
24
```

Figure 1. 24 Factorial Value algorithm by using Python

Output →

The sample number used for Dry Run is number 3

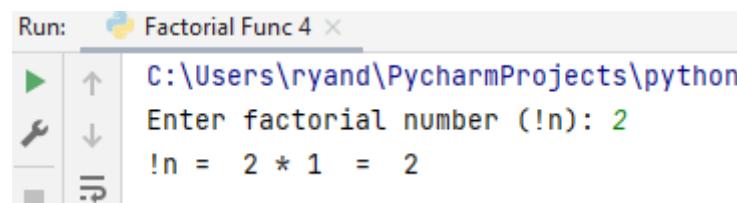


A screenshot of the PyCharm IDE interface. The title bar says "Run: Factorial Func 4". The code editor shows the following Python code and its output:

```
C:\Users\ryand\PycharmProjects\python
Enter factorial number (!n): 3
!n = 3 * 2 * 1 = 6
```

Figure 1. 25 Factorial Value Dry Run taking sample number 3

The sample number used for Dry Run is number 2

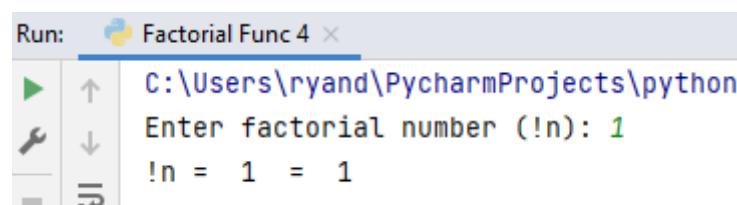


A screenshot of the PyCharm IDE interface. The title bar says "Run: Factorial Func 4". The code editor shows the following Python code and its output:

```
C:\Users\ryand\PycharmProjects\python
Enter factorial number (!n): 2
!n = 2 * 1 = 2
```

Figure 1. 26 Factorial Value Dry Run taking sample number 2

The sample number used for Dry Run is number 1

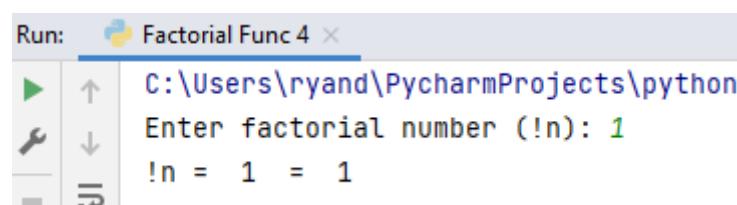


A screenshot of the PyCharm IDE interface. The title bar says "Run: Factorial Func 4". The code editor shows the following Python code and its output:

```
C:\Users\ryand\PycharmProjects\python
Enter factorial number (!n): 1
!n = 1 = 1
```

Figure 1. 27 Factorial Value Dry Run taking sample number 1

The sample number used for Dry Run is number 0



A screenshot of the PyCharm IDE interface. The title bar says "Run: Factorial Func 4". The code editor shows the following Python code and its output:

```
C:\Users\ryand\PycharmProjects\python
Enter factorial number (!n): 1
!n = 1 = 1
```

Figure 1. 28 Factorial Value Dry Run taking sample number 0

The Zero Factorial is exciting since it is widely accepted that $0! = 1$.

It may seem strange that multiplying no numbers results in 1

The sample number used for Dry Run is number 10

A screenshot of the PyCharm IDE interface. The title bar says "Run: Factorial Func 4". The code editor shows a Python script with the following content:

```
C:\Users\ryand\PycharmProjects\pythonProject\venv\Scripts\python
Enter factorial number (!n): 10
!n = 10 * 9 * 8 * 7 * 6 * 5 * 4 * 3 * 2 * 1 = 3628800
```

The code uses the `print` statement to calculate and print the factorial of 10, which is 3628800.

Figure 1. 29 Factorial Value Dry Run taking sample number 10

1.5.5 Evaluation efficiency of Factorial Value algorithm using Big-O notation

```

n = int(input("Enter factorial number (!n): "))

count = n    #define variable for 1st loop
fac = 1
my_list = []

#Using loop to show Factorial number sequence
while count > 0 :
    count = str(count)  #To convert integer to string when adding to list
    my_list.append(count)

    count = int(count)  #To convert string to integer when calculating
    count = count -1

#Using loop to get Factorial number sequence final answer
while n > 0 :
    fac = fac * n
    n = n -1

print("!n = " , " * ".join(my_list) , " = " ,fac)

```

“ $C_1 n$ ” times “ $C_2 n$ ” times Constant time = C_3

$$\begin{aligned}
 \text{Time complexity} &= C_1 n + C_2 n + C_3 \\
 &= C_1 n + C_2 n + C_3
 \end{aligned}$$

Time complexity = $O(n)$

Figure 1. 30 Evaluation efficiency of Factorial Value algorithm using Big-O notation

Let's Evaluation efficiency of Factorial Value algorithm using Big-O notation. As we learned to find out the time complexity for Sequential Statements, lets evaluate each situation.

The first operation is the “while loop”. As I explained in big O notation chapter, while loop is very similar to for loop will take C into “n” times to run. $\rightarrow C_1 n$

After the while loop there is another while loop which will take C into “n” times to run.
 $\rightarrow C_2 n$

Finally, there is a execute operation with an in-built function which takes constant time to run. $\rightarrow C_3$.

So, when we have sequential statements like these, we simply add the time taken for each statement.

$\rightarrow C_1 n + C_2 n + C_3$

As I told before in Big-O terms, we can ignore constants which are C_1, C_2, C_3 and C_4 .

$\rightarrow C_1 n + C_2 n + C_3$

And simply we can write the time complexity as big O of n as following.

Time complexity $\rightarrow O(n)$

1.6 The process in building an application

App development may be a hard and time-consuming process that, at times, involves far too many operations for a single person to efficiently execute. There are 5 steps involving in this process as following.

1. Defining and analyzing the requirements
2. Designing the application
3. Developing the application
4. Testing the application
5. The application's deployment

Step 1: Defining and analyzing the requirements

The software developers gather detailed project-related information from the customer's supporting documentation. Then, using the requirements document, project plan, use cases, and requirements traceability, software developers create and evaluate the requirements for the application.

Step 2: Designing the application

Software developers will create the application in this stage, depending on the requirements, use cases, and project scope agreed upon in the previous stage. The design will be tested against the customer's specifications after it's finished. The project plan will be verified, and a cost estimate will be made. At this point, any necessary revisions are made.

Step 3: Developing the application

Developers will create the actual code based on the finalized design papers during the development stage. After that, the application will be put to the test against the customer's specifications and test scenarios. This step involves performing code testing, quality testing, and customer acceptance tests, as well as receiving feedback from the customer. This is also where debugging takes place. The customer's acceptance of the developed application brings this stage to a close.

Step 4: Testing the application

Frontend or user interface testing and backend testing, which checks database behavior, are the two phases of application testing. In this step software developer can eliminate the bug, reduce the development cost and future maintenance cos and improve the performance of the application.

Step 5: The application's deployment

The software program will be developed according to the deployment plan in the final stage of deployment. After the customer accepts the delivered software, the developer can maintain track of the customer and continue to provide assistance.

Activity 2

2.1 What is Programming Paradigm

2.1.1 What is Programming Paradigm

Computer programs are basically built and developed to solve a certain problem. Various programming techniques have evolved during the previous few decades. Each programming paradigm offers a set of principles and rules that must be followed by the programming language. During the software compilation step, the programming language compiler follows the programming paradigm. It is essential for computer science students, as well as me, to understand the concept of programming paradigms, various types, and corresponding languages.

Each paradigm emphasizes a certain way of managing program code. A programming language is actually a strategy for solving problems. There are numerous solutions to each problem. Furthermore, each solution may take a different approach to solving the problem. Different programming languages are used to develop the computer programs.

Each programming language has a specific programming style that follows a certain programming paradigm. The programming paradigm is everything about the writing style and how the program code is arranged.

Examples :-

- C programming language belongs to the procedural programming paradigm.
- C++, Python, and Java are examples of object-oriented programming (OOP) paradigms.

Programming paradigms are divided into two categories. The paradigm type is determined by the features of the programming language as well as the way in which the program code is organized.

1. Imperative Paradigm.
2. Declarative Paradigm.

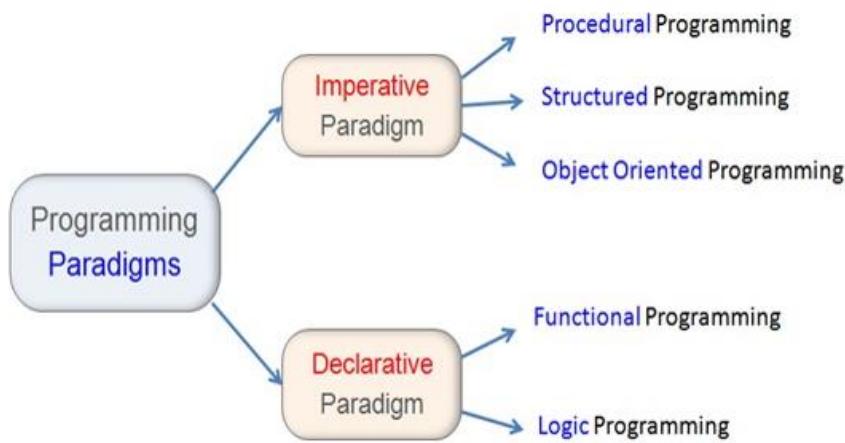


Figure 2. 1 Types of programming Paradigms

2.1.2 Imperative Paradigm.

Imperative paradigm is one of the oldest programming paradigms. It looks very similar to machine architecture. Its architecture is based on Von Neumann's. The imperative paradigm is defined as command driven. It works by changing the program's state with assignment statements. It completes tasks in a step-by-step manner by changing states. Which means in an imperative paradigm programming language, the program code leads program execution as a series of statements executed one by one. The imperative style program is made up of a series of program statements.

Each sentence instructs the computer to carry out a specific task. The programmer must elaborate on each statement in an imperative style program. Each statement describes what should be done and how it should be done. The control flow statements determine how the program statements are executed. Furthermore, the program flow can be adjusted based on the program logic. The main objective of imperative paradigm is on how to achieve the goal.

Imperative Programming Types:

1. Structured Programming.
2. Procedural Programming.
3. Object Oriented Programming.
4. Database Query Language.

2.1.3 Declarative Paradigm.

Declarative programming is a way of constructing programs in computer science that expresses computing logic without detailing its control flow. It is a programming paradigm that focuses on program logic and the final outcome. The control flow is not the most important aspect of the program under this paradigm. The basic goal of declarative programming is to achieve the desired result. While writing the program code, this paradigm is straightforward and to the point. It could make writing parallel programs easier.

The emphasis is on what must be done rather than how it should be done, emphasizing what code really does. It just declares the desired outcome rather than how it was achieved. The sole difference between imperative (how to do) and declarative (what to do) programming paradigms is that this is the only difference.

Declarative Programming Types:

1. Functional Programming.
2. Logic Programming.

2.2 Procedural Programming Paradigms

2.2.1 Main characteristics of Procedural Programming Paradigms

Procedural Programming is likely to be a new developer's first programming paradigm. A procedural paradigm program is made up of a set of procedures. Simply put, procedural programming is writing down a list of instructions that instruct the computer how to accomplish a task step by step.

Functions, methods, and subroutines are all terms used to describe procedures. In procedural programming, the program structure is made up of a series of functions. Each one carries out a unique operation. The function is made up of a series of computational steps that tell the computer how to complete a given task. Once defined, the function can be called multiple times throughout the program to repeat the same operation. The programmer has the choice of using standard library functions or creating a user-defined function library.

The C programming language is the most widely used and broadly applied programming language in the software industry. C++, Java, ColdFusion, Pascal languages are some examples for Procedural programming paradigm.

2.2.2 Advantages and Disadvantages of Procedural Programming

Advantages :-

- For general-purpose programming, procedural programming is suitable.
- A wide range of books and online course materials on tested algorithms are available, making it easy to study.
- Because the source code is portable, it can also be used to target a different CPU.
- It is not necessary to duplicate the code because it can be reused in different parts of the program.
- The memory need is also reduced using the Procedural Programming technique.
- The program's flow can be simply followed.

Disadvantages :-

- When Procedural Programming is used, the computer code is more difficult to write.
- Since procedural code is frequently not reusable, it may be necessary to rebuild it if it is required for usage in another application.
- Relationships with real-world objects are complicated.
- The operation is prioritized over the data, which may cause problems in data-sensitive situations.
- Since the data is visible to the entire program, it is not very secure.

2.2.3 Procedural Programming Paradigm example snippets of code

Example Python Code :-

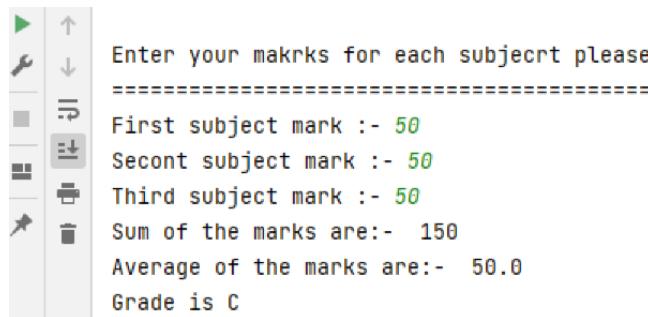
```
print ("Enter your marks for each subject please")
print ("=====")
Mark1 = int(input("First subject mark :- "))
Mark2 = int(input("Second subject mark :- "))
Mark3 = int(input("Third subject mark :- "))

SUM = Mark1 + Mark2 + Mark3
Average = SUM/3
print("Sum of the marks are:- ", SUM)
print("Average of the marks are:- ", Average)

if (Average>=75):
    print("Grade is A")
elif (Average>=60):
    print("Grade is B")
elif (Average>=40):
    print("Grade is C")
else:
    print("Grade is F")
```

Figure 2. 2 Procedural Programming Paradigm example Python code

Output :-



```
▶ ↑ ↴ ↓ ⏪ ⏩ ⏴ ⏵ ⏹ ⏸ ⏹ ⏸ Enter your marks for each subject please
=====
First subject mark :- 50
Second subject mark :- 50
Third subject mark :- 50
Sum of the marks are:- 150
Average of the marks are:- 50.0
Grade is C
```

Figure 2. 3 Procedural Programming Paradigm example Python code output

The basic grade calculation shown above. To determine the grades, this python code follows a step-by-step approach. The first two lines of code print a string. The first step is to collect the three subject marks. The second step is to add up the marks to get the total. The average is then calculated by dividing the total by the number of inputs. Finally, the program compares the average marks to the predefined evaluation grades to calculate the grade using the if condition.

The program clearly follows a step-by-step approach, as can be seen. We can see how the program flows by looking at the results. It runs line by line, step by step to produce the desired results.

Below here stated another few examples of codes for Procedural Programming Paradigm.

Example Python Code :-

```
def even_check(num_list):  
    for x in num_list:  
        if x % 2 == 0:  
            return x  
        else:  
            pass  
  
x = even_check([1,2,3,4])  
print(x)
```

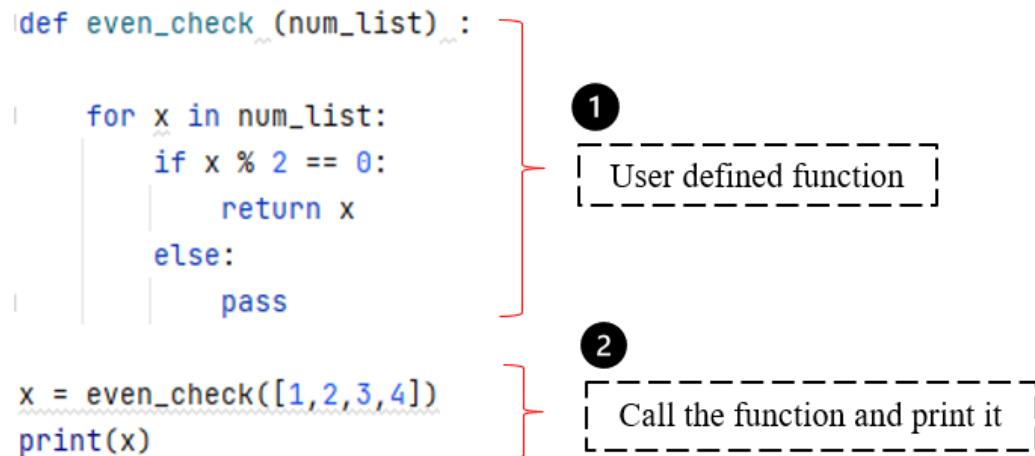
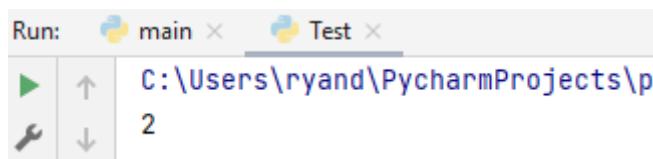


Figure 2. 4 Procedural Programming Paradigm example Python code

Output :-



```
Run:   main ×   Test ×  
▶   ↕   C:\Users\ryand\PycharmProjects\p  
🔧   ↓   2
```

Figure 2. 5 Procedural Programming Paradigm example Python code output

Example C# Code :-

```
0 references
static void Main(string[] args)
{
    GetMax(1, 10);
    Console.ReadLine();
} 2 Call the user defined method in Main Method and Read it.

1 reference
static void GetMax (int num1 , int num2)
{
    int result;
    if (num1>num2)
    {
        result = num1;
        Console.WriteLine(result);
    }
    else
    {
        result = num2;
        Console.WriteLine(result);
    }
}
```

Figure 2. 6 Procedural Programming Paradigm example C# code

Output :-

```
C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe
10
```

Figure 2. 7 Procedural Programming Paradigm example C# code output

2.3 Object-Oriented Programming Paradigms

2.3.1 Main characteristics of Object-Oriented Programming Paradigms

Object-oriented programming (OOP) is a style of structured programming in which program components are represented as objects. The program is composed of a set of classes and objects that are used to communicate. All program components are represented as objects in OOP programming. Objects are the smallest and most basic entities, and all computations are done on them.

Data and associated methods are bound together as a single unit by an object. As a result, the programmer can define the access specifier to manage data access permissions. The focus is on data rather than procedure. The data in the program is protected by OOP programming from unintentional operations performed by other methods. As a result, object-oriented programming provides strong security features.

For enterprise-level software projects, OOP-compliant languages such as C++, Java, and Python are widely used.

2.3.2 Advantages and Disadvantages of Object-Oriented Programming

Advantages :-

- OOP is easy to maintain because of its flexibility and encapsulation.
- OOP simulates the real world, making it simpler to understand.
- Since objects are complete in and of themselves, they can be reused in other programs.

Disadvantages :-

- Object-Oriented programs are typically slower and consume a lot of memory.
- Over-generalization
- It is possible that programs designed using this paradigm will take longer to develop.

2.3.3 Object-Oriented Programming Paradigm example snippets of code

Example Python Code :-

```
class computer :

    def config (self) :
        print("i5, 16GB ram, 1TB HDD")

    def price(self):
        print("105 000/=")
```

1

User defined Class
with
Methods/Functions

com1 = computer()
pr = computer()

2

Define object with relevant Class

com1.config()
pr.price()

Object calling with a function

This is how we
call a function
in a Class.

Figure 2. 8 Object-Oriented Programming Paradigm example Python code

Output :-

```
▶ | ↑ i5, 16GB ram, 1TB HDD
  | ↓ 105 000/=
```

Figure 2. 9 Object-Oriented Programming Paradigm example Python code output

When looking at the above python code, I have defined 1 Class called Computer. In this class I have defined two methods named "config" and "price" then I've introduced a specific task for each method also. Later at the end I've defined 2 objects under this class then called out the methods by the object's name.

The programmer can capture data and keep it safe and secure from outside interfaces using the OOP programming paradigm. This is known as Encapsulation. The sharable class is a big benefit because it allows us to reuse the same code without duplicating it. If the programmer has to add additional data or functionalities, he or she can do so easily.

Below here stated another few examples of codes for Object-Oriented Programming Paradigm.

Example C# Code :-

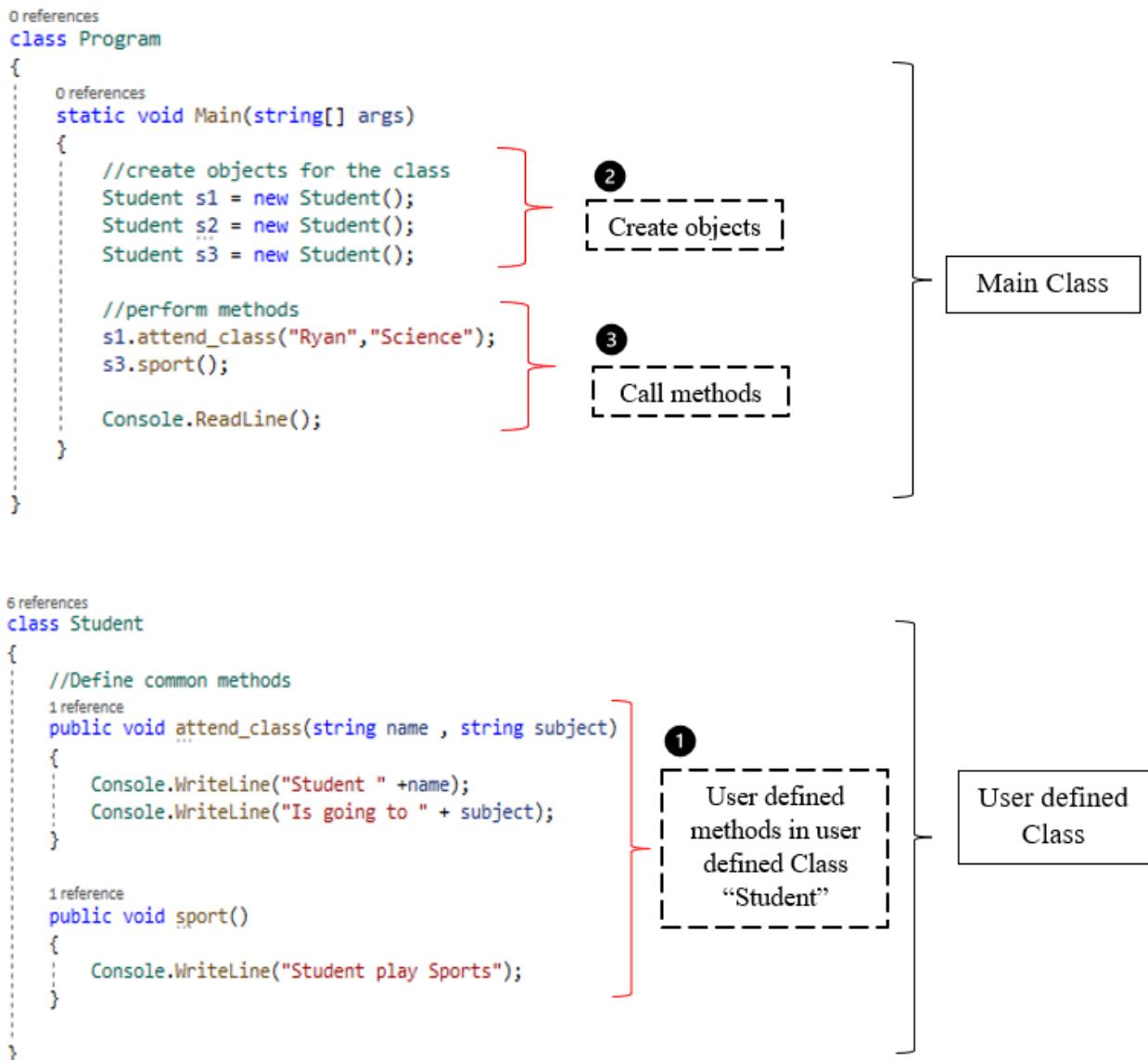


Figure 2. 10 Object-Oriented Programming Paradigm example C# code

Output :-

```

C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe
Student Ryan
Is going to Science
Student play Sports

```

Figure 2. 11 Object-Oriented Programming Paradigm example C# code output

2.4 Event-Driven Programming Paradigms

2.4.1 Main characteristics of Event-Driven Programming Paradigms

The concept behind event-driven programming is that the program is built to react. Event-driven programming is a programming paradigm in which events control the flow of program execution. It responds to specified types of user input, such as a command button click, a drop-down list selection, a text box entry, a mouse click, a key press, a message from the operating system or another program, or other user events. An event-driven program is one that identifies events as they happen and reacts with an appropriate event-handling procedure.

Event-driven programs can be created in any programming language, while some languages such as Visual Basic are built specifically for event-driven programming. They also have an integrated development environment (IDE) that partially automates code development and includes a large number of built-in objects and controls, each of which can react to a number of different events. Event-driven programming is supported by almost all object-oriented and graphic programming languages. Examples of such languages are Visual Basic, Visual C++, and Java.

Many visual programming environments will even include code templates for event-handlers, requiring the programmer to simply input the code that defines the program's response to the event. Each event handler on a form is generally tied to a specific object or control. Any additional subroutines, methods, or function procedures are usually kept in their own code module and can be called from other parts of the program when needed.

2.4.2 Advantages and Disadvantages of Object-Oriented Programming

Advantages :-

- If a programmer needs to modify something, they can quickly adjust event-driven programming because of its flexibility.
- Event-driven allows the user to choose from a variety of tools on the toolbar to create exactly what they need, such as buttons, radio buttons, and other controls.
- By allowing users to directly modify the object for which they want the code, hence event-driven programming can make programming easier for some.
- More interactive programs are accessible with event-driven programming. Event-driven programming is used in almost all recent GUI apps.

Disadvantages :-

- Event-driven programming is usually more difficult and time-consuming for basic programs.
- Errors can be harder to notice in more complex systems than in simpler, procedural programs.
- Complex graphical user interfaces may take longer to load and operate than simpler programs, especially if RAM is insufficient.

2.4.3 Event-Driven Programming Paradigm example snippets of code

Example C# Windows Forms Code :-

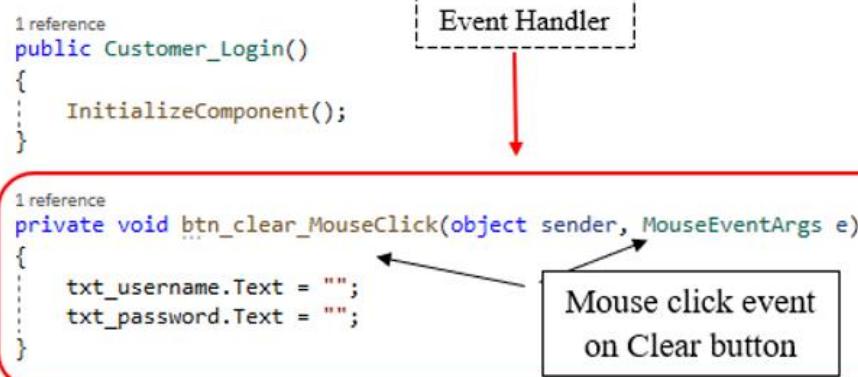


Figure 2. 12 Event-Driven Programming Paradigm example C# Windows Forms code

According to above example Mouse event must be triggered in the program for them to occur. Which means the user must interact with the targeted item in the program. For above code example “Clear Button” clicking by a mouse triggers the program.

When a specific event is triggered, event handlers are a sort of function or method that performs a specific action. For example, an event handler may be a button that displays a message when the user clicks it and closes the message when the user clicks the button again. For above example the event handler is clearing Username and Password textboxes fields when triggered.

If I further explained, when an event occurs, we usually need to pass along some information about it. This is done by using EventArgs. In above example I have created a custom EventArgs class for “Clear Button” which including clearing textboxes fields as event details. The above given event sensitive for mouse click as it refers. When I click mouse button on Clear Button the event triggers then perform specific tasks which I’ve given in the event handler.

2.5 Relationships among Procedural, Object oriented and Event-driven paradigms

Procedural Programming Paradigm	Object Oriented Programming Paradigm	Event-Driven Programming Paradigm
Helps to write commands using a character user interface.	It helps to write commands in modules.	Creates applications using a graphical user interface.
Commands are written in a linear format and are also executed in a linear format.	Objects and functions are established to interact with one another in order to complete specified tasks.	On events, actions are defined. These events can be triggered by mouse clicks and movements, as well as keyboard strokes.
It focuses on the execution of steps in a specific order.	Concentrates on objects or data and makes it easier to secure them from illegal access.	Concentrate on choosing on a user interface.
Basic, Fortran, and COBOL are the most popular languages that follow this paradigm.	Smalltalk, C++, and JAVA are the most common languages that follow this paradigm.	Visual Basic and C# are the most common languages that follow this approach.

Table 2. 1 Relationships among Procedural, Object oriented and Event-driven paradigms

2.6 Best programming paradigm to use

2.6.1 Advantages of Object-Oriented programming (OOP) and Event-Driven Programming (EDP) over pure Procedural Programming (PP)

Over pure Procedural Programming, Object-Oriented Programming has a number of advantages. The first benefit is that OOP and EDP make software development faster. Objects and classes used in OOP allow code to be reused as needed, as contrast to Procedural Programming, which needs rewriting for each usage.

The second benefit is that OOP makes debugging easier. Unlike Procedural Programming, which requires the execution of fully completed code, OOP allows the execution of individual classes. This makes error detection and troubleshooting much easy. OOP also leads to increased software development productivity.

To execute a program, EDP allows the capture of single instances of events such as mouse clicks or key presses. The user is separated from the internal procedures required to complete a task and is only responsible for triggering the program's events. Procedural Programming, on the other hand, uses only built-in instructions and cannot receive external instructions after they have been executed. Each instruction necessary to run the software must be specified by the user.

2.6.2 Conclusion

Different programming languages from various paradigms can be used to create computer programs. As previously said, each approach to the solution will have certain advantages and disadvantages. To put it another way, the programming paradigm refers to the various approaches that are utilized to discover a software solution to a problem. Each solution to the problem has its own set of benefits and drawbacks.

Programming paradigms have been constantly evolving due to the diversification of computer use and the necessity for larger and more complex applications. Object-Oriented programming and Event-Driven Programming are the most popular programming paradigms in today's world. These paradigms isolate the user from the program's internal workings, allowing them to accept various and complicated instructions as needed. These applications also allow for code reuse, which means that other programs can utilize the same codes to do identical functions. Because of these benefits, procedural programming has been pushed out in favor of OOP and EDP. Hence, Object-Oriented Programming and Event-Driven Programming are the most user-friendly paradigms to use.

Activity 3

3.1 Integrated Development Environment (IDE)

3.1.1 What is an Integrated Development Environment (IDE)

Creating software applications is a challenging task. Developers must create a variety of interconnected components, including code, user interfaces, project structures, environment configurations and more.

An integrated development environment, or IDE, is a software application that brings together all of the tools required for a software development project in one place. Users can utilize IDEs to write code, organize text groupings, and manage programming redundancies on a more basic level. IDEs, on the other hand, combine the capabilities of different programming processes into a single application.

An IDE, at a minimum level, includes an editor, compiler, and debugger, as well as code completion and general code management. Advanced capabilities like data visualization, tracing, and cross-referencing are available in some IDEs. Some IDEs specialize in a single programming language, such as Python or Java, although many others support many languages. In terms of text editing, IDEs usually include or allow the use of frameworks and element libraries to extend the capability of base-level programming.

Most IDEs have built-in debuggers that activate when the project is built. Many IDEs include visual debuggers as a feature. Users are shown which areas of the code have difficulties if any bugs or errors are found. Complex programming benefited from IDEs because they provide greater coding assistance, code completion, debugging, visual representation of code, and deep program analysis to improve the overall development experience.

3.1.2 Common characteristics of an IDE

Having a Text Editor :

Almost every IDE will have a text editor for writing and manipulating source code.

Although some programs feature visual components that allow users to drag and drop front-end components. The majority have a simple interface that highlights language-specific syntax.

Having a Debugger :

Debugging tools help users in finding and fixing bugs in source code. To evaluate functionality and performance, they frequently recreate real-world scenarios. Before an application is deployed, programmers and software engineers may usually test the various code parts and discover issues.

Having a compiler :

Compilers are pieces of software that convert programming languages into machine-readable formats, such as binary code. To guarantee that the machine code is accurate, it is examined. After that, the compiler parses and optimizes the code to improve performance.

Having code completers :

Code completion tools help programmers by finding and adding common code components intelligently. These features lower the possibility of errors and bugs while saving developers time when developing code.

Having Plugins and Integrations :

Since an IDE serves as development portal, integrating all of your other development tools will boost development workflows and productivity. Poor integrations can generate plenty of problems and difficulties.

3.1.3 Benefits of IDE

- Most developers utilize three key tools which are source code editors, debuggers, and compilers. These all 3 key tools centralized on the IDE platform. This allows users to develop, test, and process code all in the same place.
- With IDE it's also easier to navigate the source code when these tools are all in one place. Many come with extra features like code testing, organization, and restructuring.
- In addition, Developers' capabilities and development speed are greatly enhanced by IDE features like as autocomplete, build, and deployment features.
- The capable of completing code improves the programming process.
- Checks for mistakes automatically to ensure high-quality code.
- Developers can use refactoring to create comprehensive and error-free name changes.
- Ensure that the development process runs smoothly.
- Enhance developer productivity and happiness.

3.1.4 Types of IDE

1) IDE for a specific language

There are IDEs intended exclusively for developers who work in a single language. Jikes and Jcreator for Java, CodeLite and C-Free for C/C++, and Idle for Python are just a few examples.

2) Multi-language IDE

As a new developer, we should look into learning how to use a multi-language IDE. Visual Studio, for example, is a multi-language IDE that is well-known for its extensive capabilities and ongoing support for extensions and upgrades. Introducing support for a new programming language is as simple as adding an extension.

3) IDE for mobile development

Many new tools are becoming available as the mobile app development market grows. To create successful and resourceful apps, mobile app developers need a platform specialized to this type of development. Android Studio and Xcode, for example, are mobile development IDEs for the Android and iOS platforms.

4) Web/Cloud-Based IDE

When compared to local development environments, cloud-based IDEs provide numerous advantages. A SaaS IDE can do long-running jobs without using a local workstation's computational resources. Cloud IDEs are typically platform-independent, allowing connectivity to a variety of cloud providers.

3.1.5 Best IDEs to use

The ability to customize an environment with plugins and integrations is one of the most important reasons that IDEs might be more advantageous than other tools. Plugins allow us to change workflows and add new features. Color themes and schedules are examples of simple plugins, while continuous deployment and database development extensions are examples of more complex plugins. Some of the best multifunctional IDE solutions for C, C++, PHP, Java, JavaScript, Python, and other languages are listed below.

- Visual Studio
- Pycharm
- Xcode
- Eclipse
- PhpStorm
- WebStorm
- Netbeans
- AWS Cloud 9



Figure 3. 1 Best IDEs to use

3.1.6 The use of the Visual Studio IDE

Visual studio is the best IDE for any kind of coding or development. It is one of the greatest IDEs for programming because it has an extension and updates feature that allows us to utilize java, JavaScript, Python, and other languages in a single IDE since its interface is very user-friendly, allowing users to use it quickly and get the most out of our code.

C++ and C# are the most commonly used languages. We can however do program Android, Windows Phone, Mac, and Windows desktop computers. And also, different templates are supported by Visual Studio. Overall, this Visual Studio IDE is fantastic for developers, whether they're working on embedded systems, websites, games, or graphics. If we can't locate what you're looking for, there's a comprehensive search engine that looks through all of Visual Studio's features and plugins.

3.2 What happens when developing a system without an IDE?

3.2.1 Problems usually occurs when developing a system without an IDE

Following problems usually occurs when developer doesn't use an IDE for coding

- Developer won't have a compiler that can detect syntax errors.
- Developer can't execute, change, and revise the code till it works.
- Developer is unable to perform test cases on the code.
- Developer won't having a debugger to step through the code.

3.2.2 Ways to developing a system without an IDE

Before IDE come to play a major role in programmer's life, developers used text editors and code editors back then.

A text editor is simply a text editing computer software. It is one of the most essential tools for programmers because it allows them to type and edit text, most commonly programming language files. Text editors have nothing to do with programming. In reality, they're made to function with whichever framework or language we choose. The most popular text editors that come packaged with Microsoft Windows are WordPad and Notepad. When we examine these editors, we notice that they all perform the same basic text editing functions. Simply put, they take some input, change it, and then produce some output.

One of the most important tools for programmers is a code editor, which is created specifically to modify the source code of computer programs. A code editor is essentially a text editor that also serves as a tool for writing code. It allows us to color the code and gives access to more complex tools to make coding easier. It's similar to a text editor, but with extra capabilities and features built in. A basic code editor might be a small program or a component of an IDE or a web browser.

3.2.3 Difference between IDE and Text Editor

A text editor is a computer program and tool that allows you to edit plain text. An IDE, on the other hand, is a full-fledged software environment that brings together all of the essential developer tools needed to create and test software.

In a text editor, the text comes into focus. But IDEs are bringing together several parts of a computer program into a single graphical user interface (GUI). IDEs enable a group of developers to work on different modules of the same project in a logical order.

Text editors are typically simpler than integrated development environments (IDEs), but IDEs are unquestionably more feature-rich, with features like code intelligence, project management, debugging, compilation, and more.

3.2.4 Difference between IDE and Code Editor

Code editors are text editors that have sophisticated built-in features and specific functionalities that make code editing easier and faster. An integrated development environment (IDE) is a collection of software development tools that make coding easier. It simplifies the software development process by combining the various features of a computer program into a single GUI.

An integrated development environment (IDE) combines a text editor, a code editor, a debugger, a compiler, and other tools into a single utility belt. A code editor can be a standalone application or a component of an IDE or web browser. It only allows us to color our code and gives us more powerful tools to assist us code more efficiently.

3.3 Algorithms and design system for vehicle tariff calculation for Rents

3.3.1 Flow Chart for Vehicle tariff calculation for rents

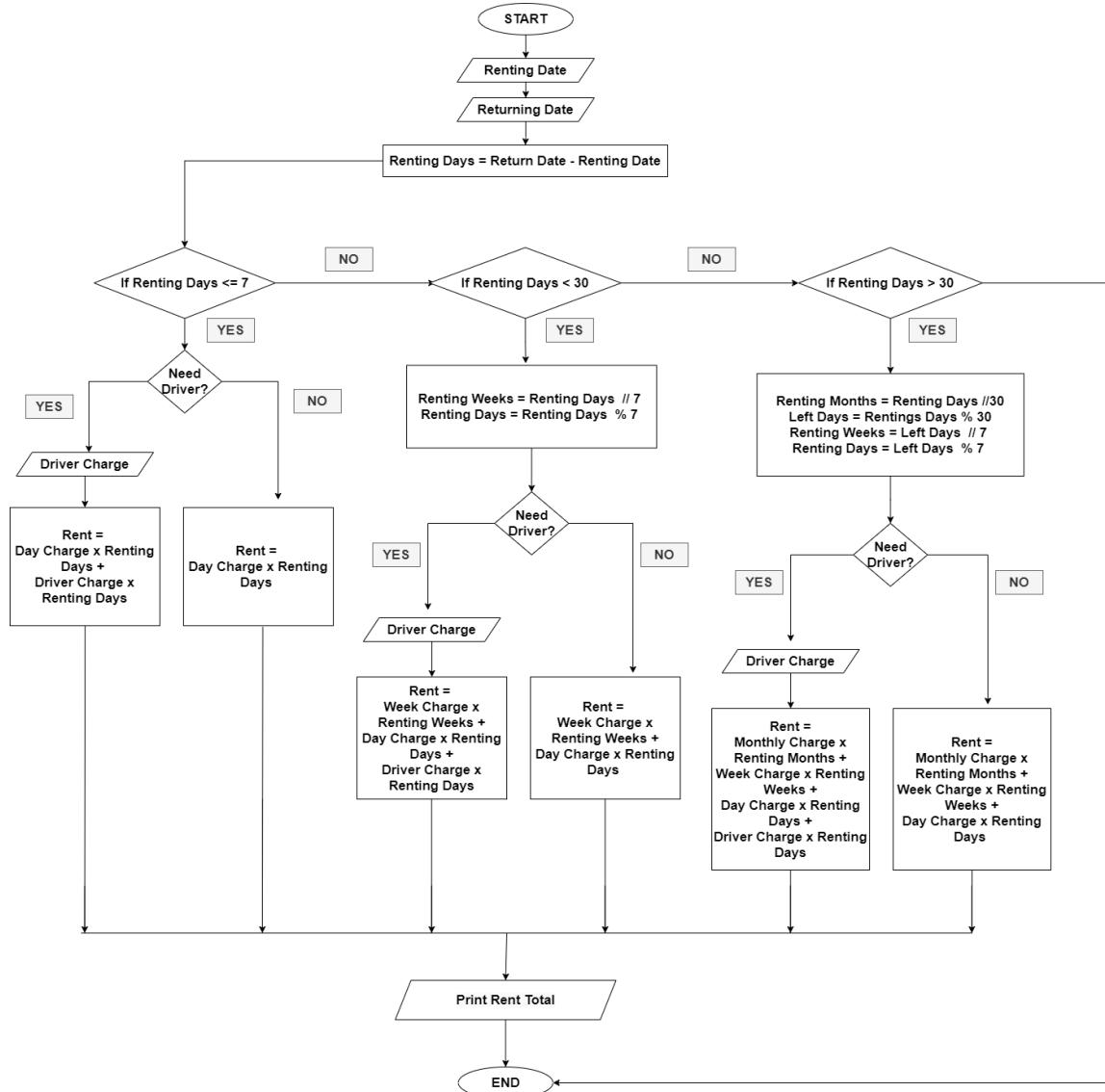


Figure 3. 2 Flow Chart for Vehicle tariff calculation for rents

3.3.2 Visual Studio IDE interface to calculate vehicle rents

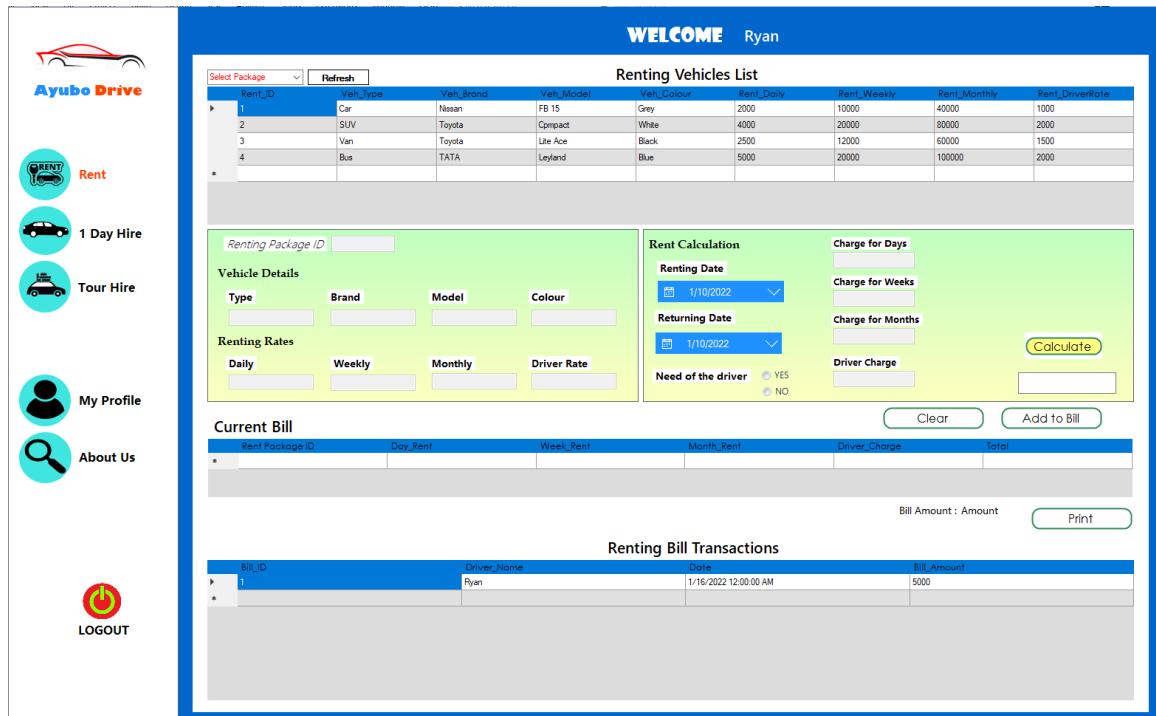


Figure 3. 3 Visual Studio IDE interface to calculate vehicle rents

3.3.3 Visual Studio IDE coding to calculate vehicle rents

```

149 //***** CALCULATION PART *****
150 //CALCULATION PART
151
152 reference
153 private void btn_Calculate_Click(object sender, EventArgs e)
154 {
155     Boolean driverYES_checked = rbtn_yes.Checked;
156     Boolean driverNO_checked = rbtn_no.Checked;
157
158     DateTime date1 = dtp_start.Value.Date;
159     DateTime date2 = dtp_end.Value.Date;
160
161     TimeSpan date_dif = date2 - date1;
162
163     int days = date_dif.Days;      //Convert date differences into integer
164
165
166
167     int cal = 0;
168     if (driverYES_checked)
169     {
170         if ((Convert.ToInt32(days) <= 7))
171         {
172             int Days_rent = Convert.ToInt32(txt_daily_rent.Text) * Convert.ToInt32(days);
173             int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
174
175             cal = driver_charge + Days_rent;
176
177             //display data
178             txt_calculate.Text = Convert.ToString(cal);
179             txt_charge_Days.Text = Convert.ToString(Days_rent);
180             txt_charge_Weeks.Text = Convert.ToString("0");
181             txt_charge_Months.Text = Convert.ToString("0");
182             txt_charge_Driver.Text = Convert.ToString(driver_charge);
183         }
184         else if ((Convert.ToInt32(days) < 30))
185         {
186             int Weeks = (int)Math.Floor(Convert.ToInt32(days) / 7.0);    //to get weeks --> ex: 17 //7 =2
187             int leftDays = Convert.ToInt32(days) % 7;                      //to get days after weeks --> ex : 17 % 7 = 3
188
189             int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
190             int Days_rent = leftDays * Convert.ToInt32(txt_daily_rent.Text);
191             int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
192
193             cal = driver_charge + Weeks_rent + Days_rent;
194
195             //display data
196             txt_calculate.Text = Convert.ToString(cal);
197             txt_charge_Days.Text = Convert.ToString(Days_rent);
198             txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
199             txt_charge_Months.Text = Convert.ToString("0");
200             txt_charge_Driver.Text = Convert.ToString(driver_charge);
201         }
202     }
203 }

```

Figure 3. 4 Visual Studio IDE coding to calculate vehicle rents - part 1

```

282
283
284     else if ((Convert.ToInt32(days) >= 30))
285     {
286         int Months = (int)Math.Floor(Convert.ToInt32(days) / 30.0); //to get months --> ex: 59//30 = 1
287         int leftDaysMonth = Convert.ToInt32(days) % 30;               //to get days after months --> ex 59%30 = 29
288
289         int Weeks = (int)Math.Floor(leftDaysMonth / 7.0);           //to get number of weeks --> ex 29//7 = 4
290         int leftDays = leftDaysMonth % 7;                            //to get days after weeks --> ex 29%7 = 1
291
292         int Months_rent = Months * Convert.ToInt32(txt_monthly_rent.Text);
293         int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
294         int Days_rent = leftDays * Convert.ToInt32(txt_daily_rent.Text);
295         int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
296
297         cal = driver_charge + Months_rent + Weeks_rent + Days_rent;
298
299         //display data
300         txt_calculate.Text = Convert.ToString(cal);
301         txt_charge_Days.Text = Convert.ToString(Days_rent);
302         txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
303         txt_charge_Months.Text = Convert.ToString(Months_rent);
304         txt_charge_Driver.Text = Convert.ToString(driver_charge);
305     }
306
307 }

```

Figure 3. 5 Visual Studio IDE coding to calculate vehicle rents - part 2

```
Program.cs  Rent_Form.cs  Rent_Form.cs [Design]  Rent_Car.Rent_Form  day_rent
Rent Car
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
}
}
else if (driverNO_checked)
{
    if (Convert.ToInt32(days) <= 7)
    {
        int Days_rent = Convert.ToInt32(txt_daily_rent.Text) * Convert.ToInt32(days);

        cal = Days_rent;

        //display data
        txt_calculate.Text = Convert.ToString(cal);
        txt_charge_Days.Text = Convert.ToString(Days_rent);
        txt_charge_Weeks.Text = Convert.ToString("0");
        txt_charge_Months.Text = Convert.ToString("0");
        txt_charge_Driver.Text = Convert.ToString("0");
    }
    else if (Convert.ToInt32(days) < 30)
    {
        int Weeks = (int)Math.Floor(Convert.ToInt32(days) / 7.0); //to get weeks --> ex: 17 //7 =2
        int leftDays = Convert.ToInt32(days) % 7; //to get days after weeks --> ex : 17 % 7 = 3

        int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
        int Days_rent = leftDays * Convert.ToInt32(txt_daily_rent.Text);

        cal = Weeks_rent + Days_rent;

        //display data
        txt_calculate.Text = Convert.ToString(cal);
        txt_charge_Days.Text = Convert.ToString(Days_rent);
        txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
        txt_charge_Months.Text = Convert.ToString("0");
        txt_charge_Driver.Text = Convert.ToString("0");
    }
}
```

Figure 3.6 Visual Studio IDE coding to calculate vehicle rents - part 3

```
Program.cs # Rent_Form.cs # Rent_Form.cs [Design] Rent_Car.Rent_Form day_rent
Rent Car
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
}
else if (Convert.ToInt32(days) >= 30)
{
    int Months = (int)Math.Floor(Convert.ToInt32(days) / 30.0); //to get months --> ex: 59/30 = 1
    int leftdaysMonth = Convert.ToInt32(days) % 30; //to get days after months --> ex 59%30 = 29

    int Weeks = (int)Math.Floor(leftDaysMonth / 7.0); //to get number of weeks --> ex 29//7 = 4
    int leftdays = leftDaysMonth % 7; //to get days after weeks --> ex 29%7 = 1

    int Months_rent = Months * Convert.ToInt32(txt_monthly_rent.Text);
    int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
    int Days_rent = leftdays * Convert.ToInt32(txt_daily_rent.Text);

    cal = Months_rent + Weeks_rent + Days_rent;

    //display data
    txt_calculate.Text = Convert.ToString(cal);
    txt_charge_Days.Text = Convert.ToString(Days_rent);
    txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
    txt_charge_Months.Text = Convert.ToString(Months_rent);
    txt_charge_Driver.Text = Convert.ToString("0");
}

}

```

Figure 3.7 Visual Studio IDE coding to calculate vehicle rents - part 4

3.3.4 Create SQL table to store Vehicle Rent details

	Column Name	Data Type	Allow Null
1	Rent_ID	int	<input type="checkbox"/>
	Veh_Type	varchar(50)	<input type="checkbox"/>
	Veh_Brand	varchar(50)	<input type="checkbox"/>
	Veh_Model	varchar(50)	<input type="checkbox"/>
	Veh_Colour	varchar(50)	<input type="checkbox"/>
	Rent_Daily	int	<input type="checkbox"/>
	Rent_Weekly	int	<input type="checkbox"/>
	Rent_Monthly	int	<input type="checkbox"/>
	Rent_DriverRate	int	<input type="checkbox"/>
			<input type="checkbox"/>

Figure 3. 8 “Details_Renting_Table” Table to store Vehicle Rent details

	Rent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	2	SUV	Toyota	Cmpact	White	4000	20000	80000	2000
3	3	Van	Toyota	Lite Ace	Black	2500	12000	60000	1500
4	4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Figure 3. 9 Stored table values in “Details_Renting_Table”

3.3.5 Create SQL table to bill Vehicle Rent details

Column Name	Data Type	Allow Nulls
Bill_ID	int	<input type="checkbox"/>
Driver_Name	varchar(50)	<input type="checkbox"/>
Date	varchar(50)	<input type="checkbox"/>
Bill_Amount	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 3. 10 “Bill_Rent_Table” Table to bill Vehicle Rent details

	Bill_ID	Driver_Name	Date	Bill_Amount
1	1	Ryan	1/16/2022 12:00:00 AM	5000
2	2	Ryan	2/3/2022 12:00:00 AM	146000
3	3	Ryan	2/3/2022 12:00:00 AM	354000

Figure 3. 11 Stored table values in “Bill_Rent_Table”

3.3.6 Test Cases for vehicle Rent

The screenshot shows the Ayubo Drive application interface. On the left sidebar, there are icons for Ayubo Drive, Rent, 1 Day Hire, Tour Hire, My Profile, About Us, and Logout. The main content area has a blue header "WELCOME Ryan". Below it is a table titled "Renting Vehicles List" with columns: Rent_ID, Veh_Type, Veh_Brand, Veh_Model, Veh_Colour, Rent_Daily, Rent_Weekly, Rent_Monthly, and Rent_DriverRate. The table contains four rows of vehicle data. To the right of the table is a "Rent Calculation" section with fields for Renting Date (1/10/2022), Returning Date (1/11/2022), Charge for Days (4000), Charge for Weeks (0), Charge for Months (0), Driver Charge (2000), and a "Calculate" button. The result "6000" is displayed in red. Below this is a "Current Bill" table with columns: Rent_Package_ID, Day_Rent, Week_Rent, Month_Rent, Driver_Charge, and Total. A "Print" button is also present. At the bottom is a "Renting Bill Transactions" table with columns: Bill_ID, Driver_Name, Date, and Bill_Amount.

Figure 3. 12 Rent vehicle for 1 day with driver

This screenshot is similar to Figure 3.12 but for an 8-day rental without a driver. The "Rent Calculation" section shows Charge for Days (4000), Charge for Weeks (20000), and Charge for Months (0). The result "24000" is displayed in red. The rest of the interface, including the sidebar and tables, remains the same.

Figure 3. 13 Rent vehicle for 8 days without driver

WELCOME Ryan

Rent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	SUV	Toyota	Compact	White	4000	20000	80000	2000
3	Van	Toyota	Lite Ace	Black	2500	12000	60000	1500
4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Renting Vehicles List

Renting Package ID: 2

Type	Brand	Model	Colour
SUV	Toyota	Compact	White

Renting Rates

Daily	Weekly	Monthly	Driver Rate
4000	20000	80000	2000

Rent Calculation

Renting Date: 1/10/2022

Returning Date: 2/10/2022

Charge for Days: 4000

Charge for Weeks: 0

Charge for Months: 80000

Driver Charge: 62000

Need of the driver: YES

Calculate: 146000

Current Bill

Rent Package ID	Day_Rent	Week_Rent	Month_Rent	Driver_Charge	Total
1					

Renting Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
1	Ryan	1/16/2022 12:00:00 AM	5000

Figure 3. 14 Rent vehicle for 31 days with driver

WELCOME Ryan

Rent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	SUV	Toyota	Compact	White	4000	20000	80000	2000
3	Van	Toyota	Lite Ace	Black	2500	12000	60000	1500
4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Renting Vehicles List

Renting Package ID: 4

Type	Brand	Model	Colour
Bus	TATA	Leyland	Blue

Renting Rates

Daily	Weekly	Monthly	Driver Rate
5000	20000	100000	2000

Rent Calculation

Renting Date: 1/10/2022

Returning Date: 1/20/2022

Charge for Days: 15000

Charge for Weeks: 20000

Charge for Months: 0

Driver Charge: 0

Need of the driver: NO

Calculate: 35000

Current Bill

Rent Package ID	Day_Rent	Week_Rent	Month_Rent	Driver_Charge	Total
1	2000	10000	40000	1000	20000
2	5000	20000	100000	2000	35000

Renting Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
1	Ryan	1/16/2022 12:00:00 AM	5000
2	Ryan	2/3/2022 12:00:00 AM	146000
3	Ryan	2/3/2022 12:00:00 AM	354000

Figure 3. 15 Vehicle rent values adding to temporary bill table

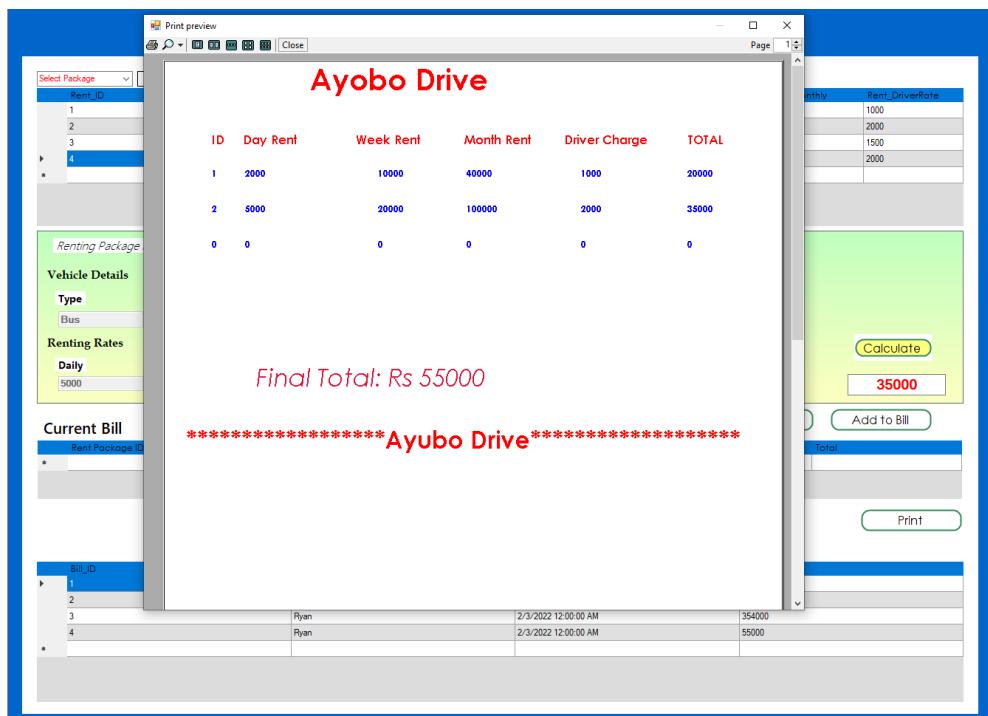


Figure 3. 16 Vehicle rent values adding to rent bill table and printing the bill

3.4 Algorithms and design system for vehicle tariff calculation for One Day Hires

3.4.1 Flow Chart for Vehicle tariff calculation for One Day Hires

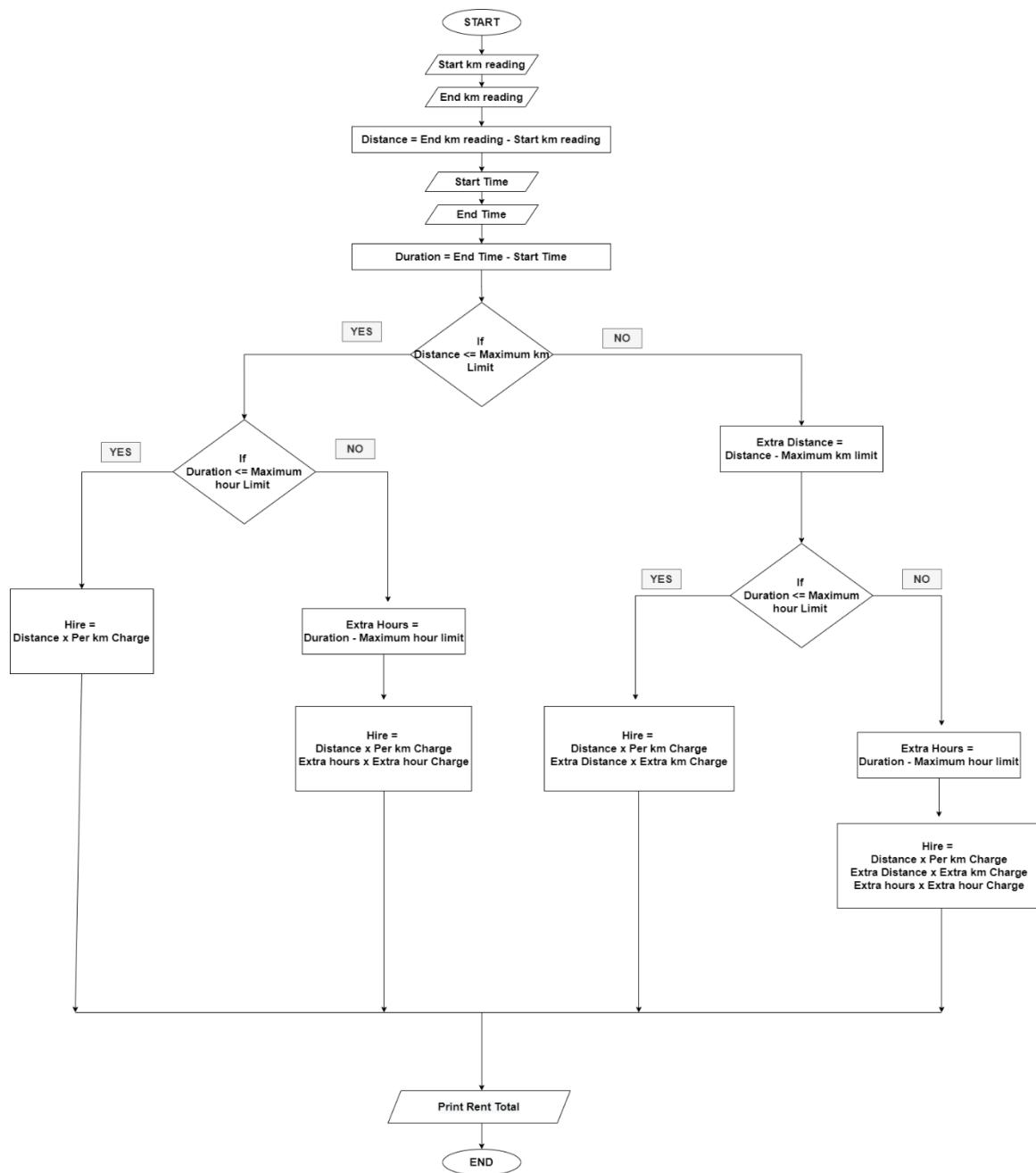


Figure 3. 17 Flow Chart for Vehicle tariff calculation for One Day Hires

3.4.2 Visual Studio IDE interface to calculate vehicle One Day Hires

The screenshot shows a Windows application window titled "WELCOME Ryan". The main area is divided into several sections:

- One Day Hiring Details:** A table showing vehicle packages. The columns include OneD_ID, Veh_Type, Veh_Brand, Veh_Model, Veh_Colour, OneD_Procedures, OneD_Per_km_charge, OneD_Max_km_limit, OneD_Extra_km_charge, OneD_Max_hours, and OneD_Extra_hours.
- Hire Package ID:** A dropdown menu currently set to "Select Package".
- Vehicle Details:** Fields for Type, Brand, Model, and Colour.
- One Day Hiring Rates:** Fields for Package, Per km Charge, Maximum km Limit, Extra km Charge, Maximum hour Limit, and Extra hour Charge.
- Rent Calculation:** Input fields for Reading at Start (km), Reading at End (km), Start Time (HH MM), and End Time (HH MM). Output fields for Distance (km), Duration (Hours), Total km Charge, Total Extra km Charge, Total Extra hour Charge, and Total Hire Charge. A "Calculate" button is present.
- Current Bill:** A table showing bill details: Bill_ID, Per_km_charge, Max_km_limit, Extra_km_charge, Max_hour_limit, Extra_hour_charge, and Total. A "Print" button is available.
- One Day Hiring Bill Transactions:** A table showing transaction details: Bill_ID, Driver_Name, Date, and Bill_Amount.

On the left side of the application window, there is a vertical sidebar with the following icons and labels:

- Ayubo Drive
- Rent
- 1 Day Hire
- Tour Hire
- My Profile
- About Us
- LOGOUT

Figure 3. 18 Visual Studio IDE interface to calculate vehicle One Day Hires

3.4.3 Visual Studio IDE coding to calculate vehicle One Day Hires

```

159 //*****CALCULATION PART *****
160
161 //CALCULATION PART
162
163 //Inference
164 private void btn_Calculate_Click(object sender, EventArgs e)
165 {
166     //Distance Calculation
167     int distance = Convert.ToInt32(txt_reading_end.Text) - Convert.ToInt32(txt_reading_start.Text);
168
169     //Duration Calculation
170     int duration_gap_in_Min = (Convert.ToInt32(txt_EndTime_H.Text) * 60 + Convert.ToInt32(txt_EndTime_M.Text)) - (Convert.ToInt32(txt_StartTime_H.Text)* 60 + Convert.ToInt32(txt_StartTime_M.Text));
171     int duration_in_Hours = (int)Math.Floor((duration_gap_in_Min) / 60.0);
172
173     //display data
174     txt_distance.Text = Convert.ToString(distance);
175     txt_duration.Text = Convert.ToString(duration_in_Hours);
176
177     if ((Convert.ToInt32(txt_distance.Text) <= Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) <= Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance and duration are less than limit
178     {
179         //FOR DISTANCE
180         int total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * Convert.ToInt32(txt_distance.Text);
181
182         //display data
183         txt_total_km_charge.Text = Convert.ToString(total_km_charge);
184
185         //FOR DURATION
186
187         //display data
188         txt_total_Extra_hour_charge.Text = Convert.ToString("0");
189
190         //display other data
191         txt_total_Extra_km_charge.Text = Convert.ToString("0");
192         txt_total_hire_charge.Text = txt_total_km_charge.Text;
193
194
195     }
196     else if ((Convert.ToInt32(txt_distance.Text) > Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) <= Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance over the limit but duration less than limit
197     {
198         //FOR DISTANCE
199         int total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * Convert.ToInt32(txt_Max_km_limit.Text);
200         int Total_Extra_km_charge = (Convert.ToInt32(txt_distance.Text) - Convert.ToInt32(txt_Max_km_limit.Text)) * Convert.ToInt32(txt_Extra_km_charge.Text);           //Taking extra distance * Extra per km price
201
202         int total_dis_charge = total_km_charge + Total_Extra_km_charge;
203
204         //display data
205         txt_total_km_charge.Text = Convert.ToString(total_km_charge);
206         txt_total_Extra_km_charge.Text = Convert.ToString(Total_Extra_km_charge);
207
208         //FOR DURATION
209
210         //display data
211         txt_total_Extra_hour_charge.Text = Convert.ToString("0");
212
213     }
214     else if ((Convert.ToInt32(txt_distance.Text) < Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) > Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance and duration are over the limit
215     {
216         //display data
217         txt_total_km_charge.Text = Convert.ToString("0");
218         txt_total_hire_charge.Text = Convert.ToString("0");
219
220         //FOR DISTANCE
221         int Total_km_charge = Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_Extra_km_charge.Text);
222         txt_total_hire_charge.Text = Convert.ToString(TOTAL);
223
224
225     }
226     else if ((Convert.ToInt32(txt_distance.Text) > Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) > Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance and duration are over the limit
227     {
228         //display data
229         txt_total_km_charge.Text = Convert.ToString("0");
230         txt_total_hire_charge.Text = Convert.ToString("0");
231
232         //FOR DISTANCE
233         int total_km_charge = total_km_charge + Total_Extra_km_charge;
234
235         //display data
236         txt_total_hire_charge.Text = Convert.ToString(total_km_charge);
237         txt_total_Extra_km_charge.Text = Convert.ToString(Total_Extra_km_charge);
238
239         //FOR DURATION
240         int Extra_hour_charge = (Convert.ToInt32(txt_duration.Text) - Convert.ToInt32(txt_Max_hour_limit.Text)) * Convert.ToInt32(txt_Extra_hour_charge.Text);
241
242         //display data
243         txt_total_Extra_hour_charge.Text = Convert.ToString(Extra_hour_charge);
244
245     }
246     else if ((Convert.ToInt32(txt_distance.Text) < Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) > Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance less than limit but duration over the limit
247     {
248         //display data
249         txt_total_km_charge.Text = Convert.ToString("0");
250         txt_total_hire_charge.Text = Convert.ToString("0");
251
252         //FOR DURATION
253         int Extra_hour_charge = (Convert.ToInt32(txt_duration.Text) - Convert.ToInt32(txt_Max_hour_limit.Text)) * Convert.ToInt32(txt_Extra_hour_charge.Text);
254
255         //display data
256         txt_total_Extra_hour_charge.Text = Convert.ToString(Extra_hour_charge);
257
258         //display other data
259         int TOTAL = Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_Extra_km_charge.Text) + Convert.ToInt32(txt_total_Extra_hour_charge.Text);
260         txt_total_hire_charge.Text = Convert.ToString(TOTAL);
261
262         txt_total_Extra_km_charge.Text = Convert.ToString("0");
263
264     }
265
266 }
267

```

Figure 3. 19 Visual Studio IDE coding to calculate vehicle One Day Hires - part 1

```

122         txt_total_Extra_hour_charge.Text = Convert.ToString("0");
123
124         //display other data
125         txt_total_hire_charge.Text = Convert.ToString("0");
126
127         //FOR DURATION
128         int total_dis_charge = total_km_charge + Total_Extra_km_charge;
129
130         //display data
131         txt_total_hire_charge.Text = Convert.ToString(total_hire_charge);
132         txt_total_Extra_km_charge.Text = Convert.ToString(Total_Extra_km_charge);
133
134         //display other data
135         int TOTAL = Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_Extra_km_charge.Text) + Convert.ToInt32(txt_total_Extra_hour_charge.Text);
136         txt_total_hire_charge.Text = Convert.ToString(TOTAL);
137
138     }
139
140     else if ((Convert.ToInt32(txt_distance.Text) < Convert.ToInt32(txt_Max_km_limit.Text) && Convert.ToInt32(txt_duration.Text) < Convert.ToInt32(txt_Max_hour_limit.Text)))      //***distance less than limit but duration over the limit
141     {
142         //display data
143         txt_total_km_charge.Text = Convert.ToString("0");
144         txt_total_hire_charge.Text = Convert.ToString("0");
145
146         //FOR DURATION
147         int Extra_hour_charge = (Convert.ToInt32(txt_duration.Text) - Convert.ToInt32(txt_Max_hour_limit.Text)) * Convert.ToInt32(txt_Extra_hour_charge.Text);
148
149         //display data
150         txt_total_Extra_hour_charge.Text = Convert.ToString(Extra_hour_charge);
151
152         //display other data
153         int TOTAL = Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_Extra_km_charge.Text) + Convert.ToInt32(txt_total_Extra_hour_charge.Text);
154         txt_total_hire_charge.Text = Convert.ToString(TOTAL);
155
156         txt_total_Extra_km_charge.Text = Convert.ToString("0");
157
158     }
159
160 }
161

```

Figure 3. 20 Visual Studio IDE coding to calculate vehicle One Day Hires - part 2

3.4.4 Create SQL table to store Vehicle One Day Hires details

Column Name	Data Type	Allow Nulls
OneD_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
OneD_Packages	varchar(50)	<input type="checkbox"/>
OneD_Per_km_charge	int	<input type="checkbox"/>
OneD_Max_km_limit	int	<input type="checkbox"/>
OneD_Extra_km_charge	int	<input type="checkbox"/>
OneD_Max_hour_limit	int	<input type="checkbox"/>
OneD_Extra_hour_charge	int	<input type="checkbox"/>

Figure 3. 21 “Details_OneDay_Hire_Table” Table to store Vehicle One Day Hires details

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hour_limit	OneD_Extra_hour_charge
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Figure 3. 22 Stored table values in “Details_OneDay_Hire_Table”

3.4.5 Create SQL table to bill Vehicle One Day Hires details

	Column Name	Data Type	Allow Nulls
PK	Bill_ID	int	<input type="checkbox"/>
	Driver_Name	varchar(50)	<input type="checkbox"/>
	Date	varchar(50)	<input type="checkbox"/>
	Bill_Amount	int	<input type="checkbox"/>
			<input type="checkbox"/>

Figure 3. 23 “Bill_OneDay_Hire_Table” Table to bill Vehicle One Day Hires details

	Bill_ID	Driver_Name	Date	Bill_Amount
1	1	Michelle	1/16/2022 12:00:00 AM	5000
2	2	Ryan	2/3/2022 12:00:00 AM	57290
3	3	Ryan	2/3/2022 12:00:00 AM	96640
4	4	Ryan	2/3/2022 12:00:00 AM	249240

Figure 3. 24 Stored table values in “Bill_OneDay_Hire_Table”

3.4.6 Test Cases for vehicle One Day Hires

WELCOME Ryan

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hour_limit	OneD_Extra_hour_charge
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Hire Package ID : 1

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Vehicle Details

Package	Per km Charge	Maximum km Limit	Extra km Charge
Air Port Drop	100	50	120
Maximum hour Limit	Extra hour Charge		
3	150		

Rent Calculation

Reading at Start	Reading at End	Distance	Total km Charge
0	50	50	5000
km	km	km	Total Extra km Charge
Start Time	End Time	Duration	Total Extra hour Charge
08 00 HH MM	09 00 HH MM	1 Hours	
(Use 24-Hour Format)			

One Day Hiring Rates

Current Bill	Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	Total
*							

One Day Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240

Bill Amount : Amount

Figure 3. 25 One Day Hire vehicle for less than time and distance limit

WELCOME Ryan

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hour_limit	OneD_Extra_hour_charge
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Hire Package ID : 1

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Vehicle Details

Package	Per km Charge	Maximum km Limit	Extra km Charge
Air Port Drop	100	50	120
Maximum hour Limit	Extra hour Charge		
3	150		

Rent Calculation

Reading at Start	Reading at End	Distance	Total km Charge
0	100	100	5000
km	km	km	Total Extra km Charge
Start Time	End Time	Duration	Total Extra hour Charge
08 00 HH MM	09 00 HH MM	1 Hours	
(Use 24-Hour Format)			

One Day Hiring Rates

Current Bill	Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	Total
*							

One Day Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240

Bill Amount : Amount

Figure 3. 26 One Day Hire vehicle for less than time limit but over the distance limit

WELCOME Ryan

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hours	OneD_Extra_hours
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Hire Package ID : 1

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Vehicle Details

Package	Per km Charge	Maximum km Limit	Extra km Charge
Air Port Drop	100	50	120
Maximum hour Limit	Extra hour Charge		
3	150		

One Day Hiring Rates

Reading at Start	Reading at End	Distance	Total km Charge
0	40	40 km	4000
Start Time	End Time	Duration	Total Extra km Charge
08:00 HH:MM	13:00 HH:MM	5 Hours	300
Calculate			Total Hire Charge
			4300
Clear			Add to Bill

Current Bill

Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	Total
*						

Bill Amount : Amount **Print**

One Day Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	5720
3	Ryan	2/3/2022 12:00:00 AM	9640
4	Ryan	2/3/2022 12:00:00 AM	24240

Figure 3. 27 One Day Hire vehicle for over than time limit but less the distance limit

WELCOME Ryan

Rent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	SUV	Toyota	Compact	White	4000	20000	80000	2000
3	Van	Toyota	Lite Ace	Black	2500	12000	60000	1500
4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Renting Package ID : 4

Type	Brand	Model	Colour
Bus	TATA	Leyland	Blue

Vehicle Details

Daily	Weekly	Monthly	Driver Rate
5000	20000	100000	2000

Renting Rates

Renting Date	Charge for Days
1/10/2022	15000
Returning Date	Charge for Weeks
1/20/2022	20000
Need of the driver	Charge for Months
<input type="radio"/> YES	0
<input checked="" type="radio"/> NO	0
Calculate	
Driver Charge 0 35000	
Clear Add to Bill	

Current Bill

Rent Package ID	Day_Rent	Week_Rent	Month_Rent	Driver_Charge	Total
1	2000	10000	40000	1000	20000
2	5000	20000	100000	2000	35000

Bill Amount : Rs 55000 **Print**

Renting Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
1	Ryan	1/16/2022 12:00:00 AM	5000
2	Ryan	2/3/2022 12:00:00 AM	146000
3	Ryan	2/3/2022 12:00:00 AM	354000

Figure 3. 28 One Day Hire vehicle for over than time and distance limit

WELCOME Ryan

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hours	OneD_Extra_hours
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Hire Package ID 1

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Vehicle Details

Package	Per km Charge	Maximum km Limit	Extra km Charge
Air Port Drop	100	50	120
Maximum hour Limit	Extra hour Charge		
3	150		

Rent Calculation

Reading at Start	Reading at End	Distance	Total km Charge
0	100	100 km	5000
Start Time	End Time	Duration	Total Extra km Charge
08:00 AM	13:00 HH:MM	= 5 Hours	6000
HH:MM	HH:MM		300

One Day Hiring Rates

OneD_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	Total
1	100	50	120	3	150	4300
2	100	50	120	3	150	11300

Current Bill

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240

One Day Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240

Print

Figure 3. 29 Vehicle One Day Hire values adding to temporary bill table

WELCOME Ryan

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hours	OneD_Extra_hours
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3										200
4										200
5										200

Hire Package ID 1

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Vehicle Details

ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	TOTAL
1	100	50	120	3	150	4300
2	100	50	120	3	150	11300
0	0	0	0	0	0	0

One Day Hiring Rates

OneD_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	TOTAL
1	100	50	120	3	150	4300
2	100	50	120	3	150	11300
0	0	0	0	0	0	0

Current Bill

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240

Print preview

Ayubo Drive

Final Total: Rs 15600

***** Ayubo Drive *****

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240
			15600

Print

Figure 3. 30 Vehicle One Day Hire values adding to One Day Hire bill table and printing the bill

3.5 Algorithms and design system for vehicle tariff calculation for Long Tour Hires

3.5.1 Flow Chart for Vehicle tariff calculation for Long Tour Hires

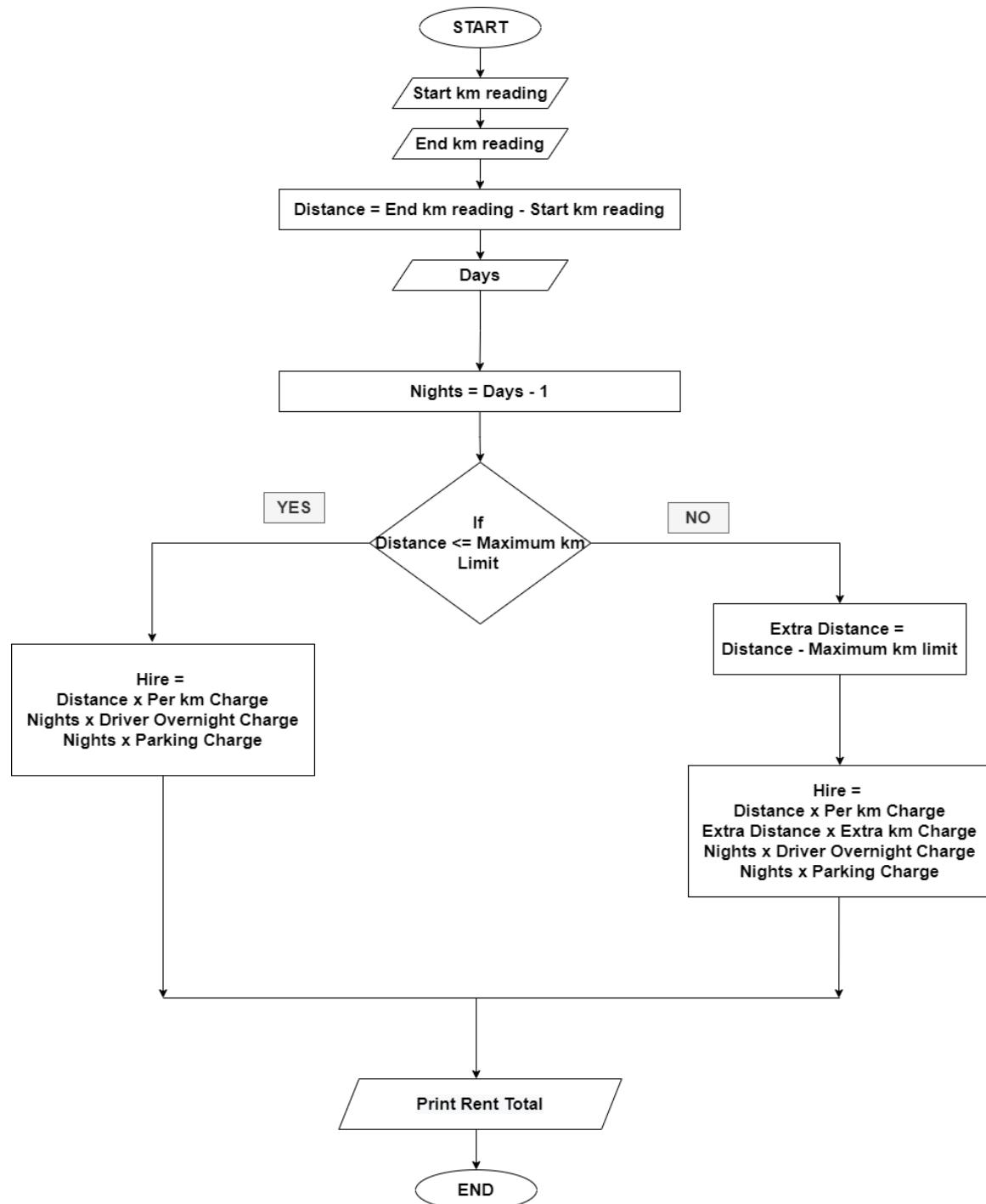


Figure 3. 31 Flow Chart for Vehicle tariff calculation for Long Tour Hires

3.5.2 Visual Studio IDE interface to calculate vehicle Long Tour Hires

The screenshot displays a Visual Studio IDE interface for managing vehicle long tour hires. On the left, a sidebar contains icons for Ayubo Drive, Rent, 1 Day Hire, Tour Hire, My Profile, About Us, and Logout. The main area is titled "WELCOME Ryan". It features several sections:

- Long Hiring Details:** A grid showing rows of data with columns for LongD_ID, Veh_Type, Veh_Brand, Veh_Model, Veh_Colour, LongD_Packages, Per_km_charge, LongD_Max_km_ltr, LongD_Extra_km_charge, LongD_Driver_over, and LongD_Vehicle_over.
- Hire Package ID:** A section for entering vehicle details (Type, Brand, Model, Colour) and one-day hiring rates (Package, Per km Charge, Maximum km Limit, Extra km Charge, Driver Charge Overnight, Vehicle Parking Charge).
- Rent Calculation:** A section for calculating distance based on reading at start and end, and for calculating total hire charge based on days, nights, and various charges (Total km Charge, Total Extra km Charge, Overnight Driver Charge, Vehicle Parking Charge).
- Current Bill:** A table showing bill details with columns for Bill_package_ID, Per_km_charge, Max_km_limit, Extra_km_charge, Overnight_Charge, Parking_Charge, and Total.
- Long Hiring Bill Transactions:** A grid showing bill transactions with columns for Bill_ID, Driver_Name, Date, and Bill_Amount.

Figure 3. 32 Visual Studio IDE interface to calculate vehicle Long Tour Hires

3.5.3 Visual Studio IDE coding to calculate vehicle Long Tour Hires

```
Programs  x Long_Hire_Form.cs  o Long_Hire_Form.cs[Design]  - Rent_Car.Long_Hire_Form  + 0_dgv_Hiring_List_CellClick(object sender, DataGridViewCellEventArgs e)
```

```
177 //***** CALCULATION PART *****
178
179 //CALCULATION PART
180
181 //Inference
182 private void btn_Calculate_Click(object sender, EventArgs e)
183 {
184     //Distance Calculation
185     int distance = Convert.ToInt32(txt_reading_end.Text) - Convert.ToInt32(txt_reading_start.Text);
186
187     //Nights
188     int nights = Convert.ToInt32(txt_days.Text) - 1;
189
190     //Display Data
191     txt_distance.Text = Convert.ToString(distance);
192     txt_nights.Text = Convert.ToString(nights);
193
194     if (Convert.ToInt32(txt_distance.Text) <= Convert.ToInt32(txt_Max_km_limit.Text))    //***Distance less than limit
195     {
196         //PER KM DISTANCE
197         int total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * Convert.ToInt32(txt_distance.Text);
198
199         //Display Data
200         txt_total_km_charge.Text = Convert.ToString(total_km_charge);
201
202         //Display Other Data
203         txt_total_extra_charge.Text = Convert.ToString("0");
204         txt_total_overnight.Text = Convert.ToString(Convert.ToInt32(txt_driver_right_charge.Text) * Convert.ToInt32(txt_nights.Text));
205         txt_total_parking_charge.Text = Convert.ToString(Convert.ToInt32(txt_parking_charge.Text) * Convert.ToInt32(txt_nights.Text));
206         txt_total_hire_charge.Text = Convert.ToString(Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_extra_km_charge.Text) + Convert.ToInt32(txt_total_overnight_charge.Text) + Convert.ToInt32(txt_total_parking_charge.Text));
207
208
209     }
210
211     else if (Convert.ToInt32(txt_distance.Text) > Convert.ToInt32(txt_Max_km_limit.Text))    //***Distance over the limit
212     {
213         //FOR DISTANCE
214         int total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * Convert.ToInt32(txt_Max_km_limit.Text);
215         int total_extra_km_charge = (Convert.ToInt32(txt_distance.Text) - Convert.ToInt32(txt_Max_km_limit.Text)) * Convert.ToInt32(txt_Extra_km_charge.Text);    //Taking extra distance * Extra per km price
216
217         int total_dis_charge = total_km_charge + total_extra_km_charge;
218
219         //Display Data
220         txt_total_km_charge.Text = Convert.ToString(total_km_charge);
221         txt_total_extra_km_charge.Text = Convert.ToString(total_extra_km_charge);
222
223         //Display Other Data
224         txt_total_overnight_charge.Text = Convert.ToString(Convert.ToInt32(txt_driver_right_charge.Text) * Convert.ToInt32(txt_nights.Text));
225         txt_total_parking_charge.Text = Convert.ToString(Convert.ToInt32(txt_parking_charge.Text) * Convert.ToInt32(txt_nights.Text));
226         txt_total_hire_charge.Text = Convert.ToString(Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_extra_km_charge.Text) + Convert.ToInt32(txt_total_overnight_charge.Text) + Convert.ToInt32(txt_total_parking_charge.Text));
227
228
229     }
230
231 }
```

Figure 3.33 Visual Studio IDE coding to calculate vehicle Long Tour Hires - part 1

3.5.4 Create SQL table to store Vehicle Long Tour Hires details

Column Name	Data Type	Allow Nulls
LongD_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
LongD_Packages	varchar(50)	<input type="checkbox"/>
LongD_Per_km_charge	int	<input type="checkbox"/>
LongD_Max_km_limit	int	<input type="checkbox"/>
LongD_Extra_km_charge	int	<input type="checkbox"/>
LongD_Driver_overnight_charge	int	<input type="checkbox"/>
LongD_Vehicle_parking_charge	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 3. 34 “Details_LongTour_Hire_Table” Table to store Vehicle Long Tour Hires details

	LongD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	LongD_Packages	LongD_Per_km_charge	LongD_Max_km_limit	LongD_Extra_km_charge	LongD_Driver_overnight_charge	LongD_Vehicle_parking_charge
1	1	Car	Nissan	FB 15	Grey	2 days trip	100	200	120	1000	200
2	2	Car	Nissan	FB 15	Grey	3 days trip	100	300	120	1000	200
3	3	Car	Nissan	FB 15	Grey	4 days trip	100	400	120	1000	200
4	4	Car	Nissan	FB 15	Grey	5 days trip	100	500	120	1000	200
5	5	Car	Nissan	FB 15	Grey	6 days trip	100	600	120	1000	200
6	6	Car	Nissan	FB 15	Grey	7 days trip	100	700	120	1000	200

Figure 3. 35 Stored table values in “Details_LongTour_Hire_Table”

3.5.5 Create SQL table to bill Vehicle Long Tour Hires details

	Column Name	Data Type	Allow Nulls
1	Bill_ID	int	<input type="checkbox"/>
	Driver_Name	varchar(50)	<input type="checkbox"/>
	Date	varchar(50)	<input type="checkbox"/>
	Bill_Amount	int	<input type="checkbox"/>
			<input type="checkbox"/>

Figure 3. 36 “Bill_LongTour_Hire_Table” Table to bill Vehicle Long Tour Hires details

	Bill_ID	Driver_Name	Date	Bill_Amount
1	1	Michelle	1/16/2022 12:00:00 AM	5000
2	2	Ryan	2/3/2022 12:00:00 AM	57290
3	3	Ryan	2/3/2022 12:00:00 AM	96640
4	4	Ryan	2/3/2022 12:00:00 AM	249240

Figure 3. 37 Stored table values in “Bill_OneDay_Hire_Table”

3.5.6 Test Cases for vehicle Rent

The screenshot shows the Ayubo Drive application interface. On the left, there is a vertical sidebar with icons for Ayubo Drive, Rent, 1 Day Hire, Tour Hire, My Profile, About Us, and Logout.

The main area is titled "WELCOME Ryan". It displays a "Long Hiring Details" table with vehicle information and a "Rent Calculation" section for a package of 2 days. The calculation shows a total km charge of 10000, total extra km charge of 0, overnight driver charge of 1000, and a total hire charge of 11200. A "Print" button is available at the bottom right of the rent calculation panel.

Below the main panel is a "Long Hiring Bill Transactions" table.

Figure 3. 38 Long Tour Hire vehicle for less than distance limit and for 1 night

This screenshot shows the same application interface as Figure 3.38, but with different input values. The "Rent Calculation" section shows a reading at start of 0 km, an end reading of 500 km, a distance of 500 km, days of 3, and nights of 2. The total km charge is 30000, total extra km charge is 24000, overnight driver charge is 2000, vehicle parking charge is 400, and the total hire charge is 56400. A "Print" button is also present.

Figure 3. 39 Long Tour Hire vehicle for over the distance limit and for 2 nights

Ayubo Drive

- Rent**
- 1 Day Hire**
- Tour Hire**
- My Profile**
- About Us**
- LOGOUT**

Long Hiring Details										
LongD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	LongD_Packages	LongD_Per_km_charge	LongD_Max_km_limit	LongD_Echo_km_charge	LongD_Driver_charge	LongD_Vehicle_charge
1	Car	Nissan	FB 15	Grey	2 days trip	100	200	120	1000	200
2	Car	Nissan	FB 15	Grey	3 days trip	100	300	120	1000	200
3	Car	Nissan	FB 15	Grey	4 days trip	100	400	120	1000	200
4	Car	Nissan	FB 15	Grey	5 days trip	100	500	120	1000	200
5	Car	Nissan	FB 15	Grey	6 days trip	100	600	120	1000	200
6	Car	Nissan	FB 15	Grey	7 days trip	100	700	120	1000	200

Hire Package ID: 2

Vehicle Details

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

One Day Hiring Rates

Package	Per km Charge	Maximum km Limit	Extra km Charge
3 days trip	100	300	120
Driver Charge Overnight	1000		200
Vehicle Parking Charge			

Rent Calculation

Reading at Start	Reading at End	Distance
0	1000	= 1000 km
Days	Nights	
10	= 9	

Current Bill

Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Overnight_Charge	Parking_Charge	Total
1	100	300	120	1000	200	124800

Bill Amount : Rs 403600

Long Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	56400
3	Ryan	2/3/2022 12:00:00 AM	132800
4	Ryan	2/3/2022 12:00:00 AM	154000
5	Ryan	2/3/2022 12:00:00 AM	278800

Print

Figure 3. 40 Vehicle Long Tour Hire values adding to temporary bill table

Ayubo Drive

- Rent**
- 1 Day Hire**
- Tour Hire**
- My Profile**
- About Us**
- LOGOUT**

Long Hiring Details										
LongD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	LongD_Packages	LongD_Per_km_charge	LongD_Max_km_limit	LongD_Echo_km_charge	LongD_Driver_charge	LongD_Vehicle_charge
1	Car	Nissan	FB 15	Grey	2 days trip	100	200	120	1000	200
2	Car	Nissan	FB 15	Grey	3 days trip	100	300	120	1000	200
3	Car	Nissan	FB 15	Grey	4 days trip	100	400	120	1000	200
4	Car	Nissan	FB 15	Grey	5 days trip	100	500	120	1000	200
5	Car	Nissan	FB 15	Grey	6 days trip	100	600	120	1000	200
6	Car	Nissan	FB 15	Grey	7 days trip	100	700	120	1000	200

Ayubo Drive

ID	Per_km_charge	Max_km_limit	Extra_km_charge	Driver_night_Charge	Parking_charge	TOTAL
1	100	300	120	1000	200	124800
2	0	0	0	0	0	0

Final Total: Rs 278800

*****Ayubo Drive*****

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	56400
3	Ryan	2/3/2022 12:00:00 AM	132800
4	Ryan	2/3/2022 12:00:00 AM	154000
5	Ryan	2/3/2022 12:00:00 AM	278800

Print

Figure 3. 41 Vehicle Long Tour Hire values adding to Long Tour bill table and printing the bill

Activity 4

4.1 Visual Studio Forms designs and system to calculate vehicle rent and hire amounts

4.1.1 Car rent system road map

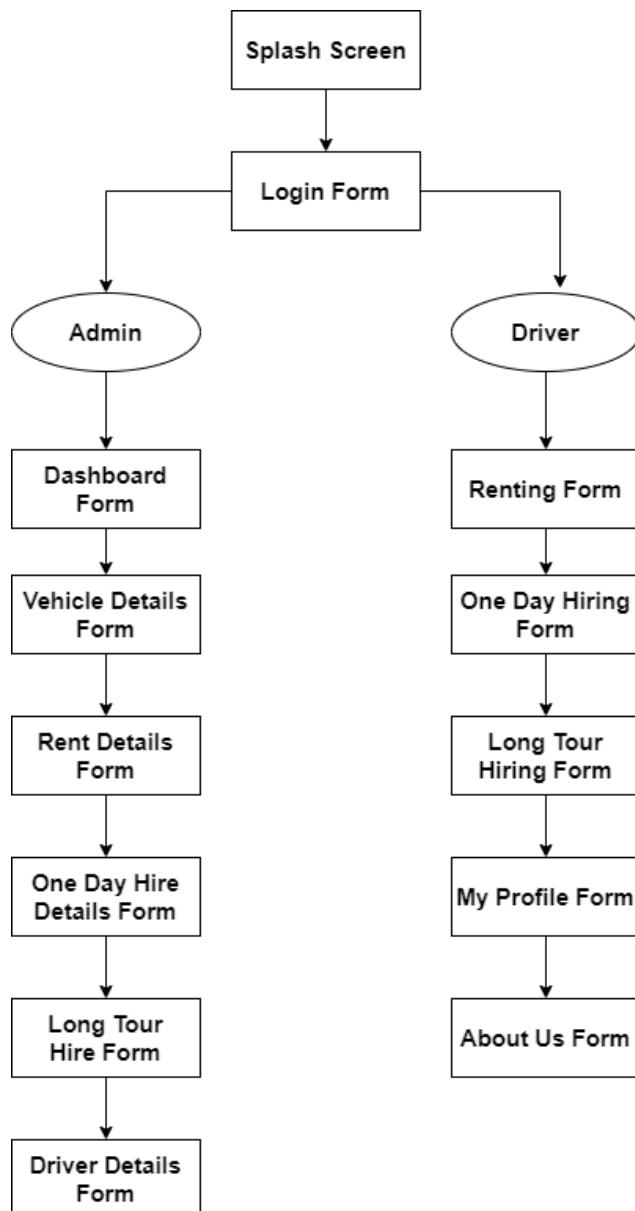


Figure 4. 1 Car rent system road map as a flow chart

4.1.2 Splash Screen Form Design and Code

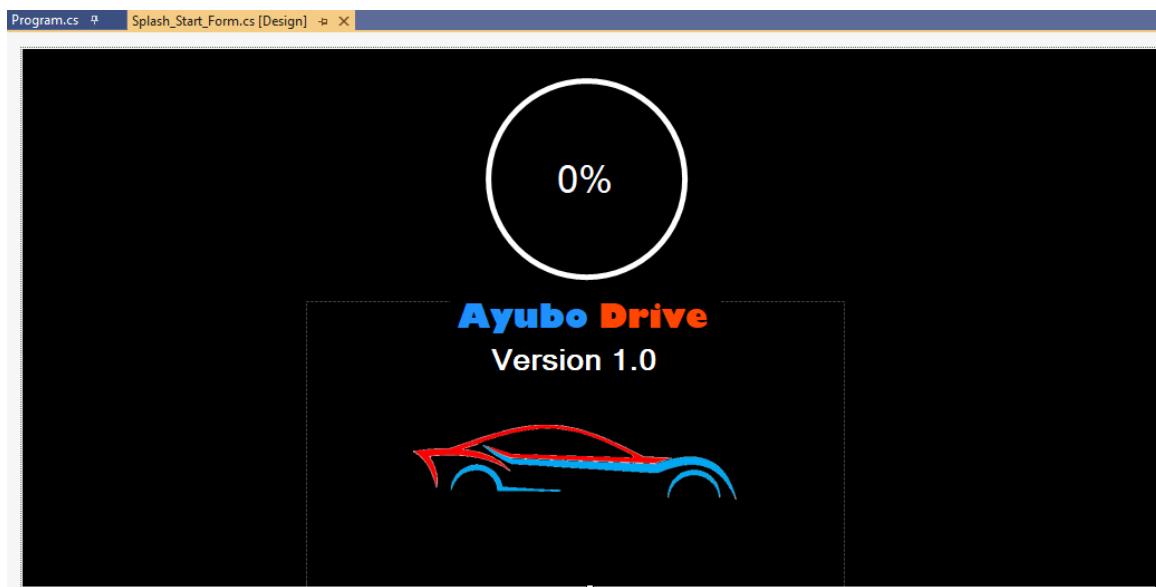


Figure 4. 2 Splash Screen Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 
11 namespace Rent_Car
12 {
13     public partial class Splash_Start_Form : Form
14     {
15         int startpoint = 0;
16 
17         public Splash_Start_Form()
18         {
19             InitializeComponent();
20         }
21 
22         private void timer1_Tick(object sender, EventArgs e)
23         {
24             bunifucircleProgressbar1.Value = startpoint;
25             startpoint += 5;
26 
27             if (bunifucircleProgressbar1.Value == 100)
28             {
29                 timer1.Stop();
30 
31                 Login_Form obj = new Login_Form();
32                 this.Hide();
33                 obj.Show();
34             }
35         }
36 
37         private void Splash_Start_Form_Load(object sender, EventArgs e)
38         {
39             timer1.Start();
40         }
41     }
42 }

```

Figure 4. 3 Splash Screen Form Code - part I

4.1.3 Login Form Design and Code



Figure 4. 4 Login Form Design

```

Program.cs    Login_Form.cs    Login_Form.cs [Design]
Rent_Car      Rent_Car.Login_Form      Login_Form()

```

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     // 38 references
15     public partial class Login_Form : Form
16     {
17         // 13 references
18         public Login_Form()
19         {
20             InitializeComponent();
21         }
22
23         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
24
25         public static string user; //Newly Created Variable --> To pass the username to bill form constructor to show in a label --> STEP 1 (Username Pass to Login Form to Billing Form)
26
27
28
29         //reference
30         private void Login_Form_Load(object sender, EventArgs e)
31         {
32             if (con.State == ConnectionState.Open)
33             {
34                 con.Close();
35             }
36             con.Open();
37         }
38
39
40

```

Figure 4. 5 Login Form Code - part 1

```

Program.cs    Login_Form.cs    Login_Form.cs [Design]
Rent_Car      Rent_Car.Login_Form      Login_Form()

```

```

38
39
40
41         //reference
42         private void btn_Login_Click(object sender, EventArgs e)
43         {
44             if (cmb_role.Text == "Select a Role" && txt_username.Text == "" && txt_password.Text == "")
45             {
46                 MessageBox.Show("Select a Role then enter username and password");
47             }
48             else if (cmb_role.Text == "Select a Role" && txt_username.Text == "" && txt_password.Text != "")
49             {
50                 MessageBox.Show("Select a Role then enter password");
51             }
52             else if (cmb_role.Text == "Select a Role" && txt_username.Text != "" && txt_password.Text == "")
53             {
54                 MessageBox.Show("Select a Role then enter password");
55             }
56             else if (cmb_role.Text == "Select a Role" && txt_username.Text != "" && txt_password.Text != "")
57             {
58                 MessageBox.Show("Select a Role");
59             }
60             else
61             {
62                 // ADMIN LOGIN CODE with defined Username and Password
63
64                 if (cmb_role.SelectedIndex > -1)
65                 {
66                     if (cmb_role.SelectedItem.ToString() == "ADMIN")
67                     {
68                         if (txt_username.Text == "Admin" && txt_password.Text == "Admin")
69                         {
70                             user = txt_username.Text; //Catching username to pass to bill form constructor to show in a label --> STEP 2 (Username Pass to Login Form to Billing Form)
71                             Dashboard obj = new Dashboard();
72                             this.Hide();
73                             obj.Show();
74                         }
75                         else
76                         {
77                             MessageBox.Show("If you are ADMIN, please enter correct Username and Password");
78                         }
79                     }
80                 }
81             }
82         }
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99

```

Figure 4. 6 Login Form Code - part 2

```

79     }
80   }
81   {
82     // DRIVER LOGIN CODE with defined Username and Password in the DATABASE
83
84     SqlCommand cmd = con.CreateCommand();
85     cmd.CommandType = CommandType.Text;
86     cmd.CommandText = "SELECT COUNT(*) FROM Details_Drivers_Table WHERE Driver_Username = '" + txt_username.Text + "' AND Driver_Password = '" + txt_password.Text + "'";
87
88     Int32 count = Convert.ToInt32(cmd.ExecuteScalar());
89
90     if (cmb_role.SelectedItem.ToString() == "DRIVER")
91     {
92       if (count > 0)
93       {
94         user = txt_username.Text; //Catching username to pass to bill form constructor to show in a label --> STEP 2 (Username Pass to Login Form to Billing Form)
95         Rent_Form obj = new Rent_Form();
96         this.Hide();
97         obj.Show();
98       }
99       else
100      {
101        MessageBox.Show("If you are DRIVER, please enter correct Username and Password");
102      }
103    }
104  }
105 }
106 }
107 }
108 }
109 }
110 }
111 }
112 }
113 }
114 }
115 }
116 }
117 }
118 }
119 }
120 }
121 }
122 }
123 }
124 }
125 }
126 }
127 }
128 }

1reference
private void lbl_clear_Click(object sender, EventArgs e)
{
  txt_username.Text = "";
  txt_password.Text = "";
}

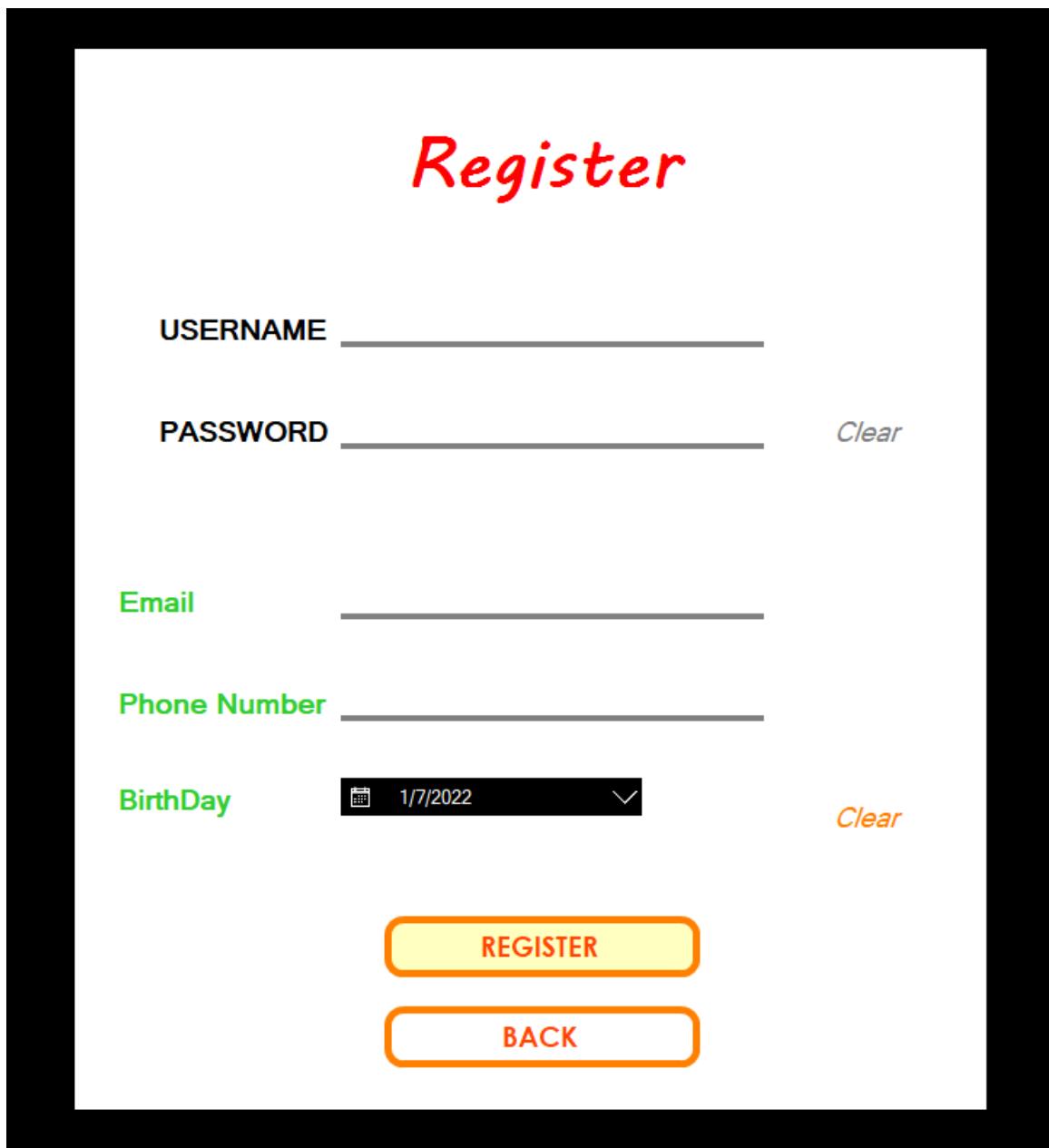
1reference
private void btn_signup_Click(object sender, EventArgs e)
{
  Driver_Signup_Form obj = new Driver_Signup_Form();
  this.Hide();
  obj.Show();
}

1reference
private void btn_Exit_Click(object sender, EventArgs e)
{
  Application.Exit();
}

```

Figure 4. 7 Login Form Code - part 3

4.1.4 Driver Signup Form Design and Code



The image shows a mobile-style driver signup form titled "Register". The form fields include "USERNAME", "PASSWORD" (with a "Clear" button), "Email", "Phone Number", and "BirthDay" (with a date picker showing "1/7/2022" and a "Clear" button). At the bottom are "REGISTER" and "BACK" buttons.

Register

USERNAME _____

PASSWORD _____ *Clear*

Email _____

Phone Number _____

BirthDay *Clear*

REGISTER

BACK

Figure 4. 8 Driver Signup Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Drawing;
5  using System.Linq;
6  using System.Text;
7  using System.Threading.Tasks;
8  using System.Windows.Forms;
9  using System.Data.SqlClient;
10 
11 namespace Rent_Car
12 {
13     public partial class Driver_Signup_Form : Form
14     {
15         InitializeComponent();
16 
17         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU\Initial Catalog=Rent_Car_Project;Integrated Security=True");
18 
19         private void Driver_Signup_Form_Load(object sender, EventArgs e)
20         {
21             if (con.State == ConnectionState.Open)
22             {
23                 con.Close();
24             }
25             con.Open();
26         }
27 
28         private void btn_register_Click(object sender, EventArgs e)
29         {
30 
31             SqlCommand cmd = con.CreateCommand();
32             cmd.CommandType = CommandType.Text;
33             cmd.CommandText = "SELECT COUNT (*) FROM Details_Drivers_Table WHERE Driver_Username = '" + txt_username.Text + "'";
34             Int32 count = Convert.ToInt32(cmd.ExecuteScalar());
35 
36             if (count > 0)
37             {
38                 MessageBox.Show("Username is already taken");
39             }
40             else
41             {
42 
43 
44 
45 
46 
47 
48 
49 

```

Figure 4. 9 Driver Signup Form Code - part I

```

49         {
50             cmd.CommandText = "INSERT INTO Details_Drivers_Table (Driver_Username,Driver_Password,Driver_Email,Driver_Phone,Driver_DOB) VALUES ('" + txt_username.Text + "','" +
51             + txt_password.Text + "','" + txt_email.Text + "','" + txt_phonenum.Text + "','" + dtp_birthday.Value.ToString("yyyy-MM-dd") + "')";
52             cmd.ExecuteNonQuery();
53             MessageBox.Show("Your details saved successfully!");
54         }
55     }
56 
57     //Inference
58     private void btn_back_Click(object sender, EventArgs e)
59     {
60         Login_Form obj = new Login_Form();
61         this.Hide();
62         obj.Show();
63     }
64 
65     //Inference
66     private void lbl_clear_Click(object sender, EventArgs e)
67     {
68         txt_username.Text = "";
69         txt_password.Text = "";
70     }
71 
72     //Inference
73     private void lbl_clear2_Click(object sender, EventArgs e)
74     {
75         txt_email.Text = "";
76         txt_phonenum.Text = "";
77         dtp_birthday.Value = DateTime.Now;
78     }
79 

```

Figure 4. 10 Driver Signup Form Code - part 2

4.1.5 Admin logged Dashboard Form Design and Code

The dashboard features a sidebar with icons for Ayubo Drive, Dashboard, Vehicles, Renting, 1 Day Hire, Tour Hire, Drivers, and Logout. The main area displays three trophies for Best Renting Driver, Best 1 Day Hire Driver, and Best Tour Driver, all awarded to 'Ryan'. A central box titled 'Financial Situation' provides earnings details for Renting, 1 Day Hire, and Tour Hire categories, broken down by seller. To the right, a 'SUMMARY' section lists vehicle counts, package counts, and driver counts across different categories.

Category	Count
Vehicles	4
Renting Packages	4
One Day Hire Packages	5
Tour Hire Packages	6
Drivers	3

Figure 4. 11 Admin logged Dashboard Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Drawing;
5  using System.Linq;
6  using System.Text;
7  using System.Threading.Tasks;
8  using System.Windows.Forms;
9  using System.Data.SqlClient;
10 
11  namespace Rent_Car
12  {
13      public partial class Dashboard : Form
14      {
15          public Dashboard()
16          {
17              InitializeComponent();
18          }
19 
20          //***** Filling Combo Boxes *****
21          fill_combo_Rent(); //Filling combobox Rent
22          fill_combo_OneD(); //Filling combobox OneD
23          fill_combo_LonD(); //Filling combobox Tours
24 
25          //***** Summary Panel Details *****
26          Get_Vehicle_Count(); //Vehicle count
27          Get_Rent_Count(); //Rent packages count
28          Get_OneD_Count(); //One Day packages count
29          Get_LonD_Count(); //Tour packages count
30          Get_Drivers_Count(); //Drivers count
31 
32          //***** Financial Panel Details *****
33          Get_Rent_Total(); //Total Earn Amount by Renting
34          Get_Rent_Amount_By_Driver(); //Driver's Earn Amount by Renting
35 
36          Get_OneD_Total(); //Total Earn Amount by One Day Hires
37          Get_OneD_Amount_By_Driver(); //Driver's Earn Amount by One Day Hiring
38 
39          Get_LonD_Total(); //Total Earn Amount by Tour Hires
40          Get_LonD_Amount_By_Driver(); //Driver's Earn Amount by Tour Hiring
41 
42          //***** Award Panel Details *****
43          Best_Rent_Driver(); //Best Rent Driver Award
44          Best_OneD_Driver(); //Best One Day Hire Driver Award
45          Best_LonD_Driver(); //Best Tour Hire Driver Award
46 
47      }
48 
49      SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
50 
51  }
52 
53  
```

Figure 4. 12 Admin logged Dashboard Form Code - part 1

```

1  //***** Quick Menu BUTTONS *****
2 
3  private void btn_Vehicles_Click(object sender, EventArgs e)
4  {
5      Vehicles_Register_Form obj = new Vehicles_Register_Form();
6      obj.Show();
7      this.Hide();
8  }
9 
10 
11  private void btn_Renting_Click(object sender, EventArgs e)
12  {
13      Give_Renting_Form obj = new Give_Renting_Form();
14      obj.Show();
15      this.Hide();
16  }
17 
18  private void btn_ID_Hire_Click(object sender, EventArgs e)
19  {
20      Give_ID_Hiring_Form obj = new Give_ID_Hiring_Form();
21      obj.Show();
22      this.Hide();
23  }
24 
25  private void btn_Tour_Hire_Click(object sender, EventArgs e)
26  {
27      Give_Long_Hiring_Form obj = new Give_Long_Hiring_Form();
28      obj.Show();
29      this.Hide();
30  }
31 
32  private void btn_Drivers_Click(object sender, EventArgs e)
33  {
34      Drivers_Register_Form obj = new Drivers_Register_Form();
35      obj.Show();
36      this.Hide();
37  }
38 
39  private void btn_Logout_Click(object sender, EventArgs e)
40  {
41      Login_Form obj = new Login_Form();
42      obj.Show();
43      this.Hide();
44  }
45 
46  //***** Summery Panel Details *****
47 
48  
```

Figure 4. 13 Admin logged Dashboard Form Code - part 2

```
Program.cs # Dashboard.cs # Dashboard.cs (Design) - Rent_Car.Dashboard - Rent_Car
```

```
102 //***** Summary Panel Details *****
103 
104     [Reference]
105     public void Get_Vehicle_Count()
106     {
107         con.Open();
108         SqlDataAdapter sda = new SqlDataAdapter("SELECT COUNT(Veh_ID) FROM Details_Vehicle_Table", con);
109         DataTable dt = new DataTable();
110         sda.Fill(dt);
111         lbl_Vehicle_Count.Text = dt.Rows[0][0].ToString();
112         con.Close();
113     }
114 
115     [Reference]
116     public void Get_Rent_Count()
117     {
118         con.Open();
119         SqlDataAdapter sda = new SqlDataAdapter("SELECT COUNT(Rent_ID) FROM Details_Renting_Table", con);
120         DataTable dt = new DataTable();
121         sda.Fill(dt);
122         lbl_Renting_Count.Text = dt.Rows[0][0].ToString();
123         con.Close();
124     }
125 
126     [Reference]
127     public void Get_Oneo_Count()
128     {
129         con.Open();
130         SqlDataAdapter sda = new SqlDataAdapter("SELECT COUNT(Oneo_ID) FROM Details_Oneo_Hire_Table", con);
131         DataTable dt = new DataTable();
132         sda.Fill(dt);
133         lbl_Oneo_Count.Text = dt.Rows[0][0].ToString();
134         con.Close();
135     }
136 
137     [Reference]
138     public void Get_LongD_Count()
139     {
140         con.Open();
141         SqlDataAdapter sda = new SqlDataAdapter("SELECT COUNT(LongD_ID) FROM Details_LongD_Hire_Table", con);
142         DataTable dt = new DataTable();
143         sda.Fill(dt);
144         lbl_Tour_Count.Text = dt.Rows[0][0].ToString();
145         con.Close();
146     }
147 
148     [Reference]
149     public void Get_Drivers_Count()
150     {
151         con.Open();
152         SqlDataAdapter sda = new SqlDataAdapter("SELECT COUNT(Driver_ID) FROM Details_Drivers_Table", con);
153         DataTable dt = new DataTable();
154         sda.Fill(dt);
155         lbl_Drivers_Count.Text = dt.Rows[0][0].ToString();
156         con.Close();
157     }
```

Figure 4. 14 Admin logged Dashboard Form Code - part 3

```
Program.cs # Dashboard.cs -x Dashboard.cs [Design] Rent_Car Rent_Car.Rent_Car Dashboard() Dashboard.cs [Design]
152 }
153
154
155
156
157 //***** Dashboard Financial Panel Details for RENT *****
158
159     1reference
160     public void Get_Rent_Total()
161     {
162         1con.open();
163         1SqlDataAdapter sda = new SqlDataAdapter("SELECT SUM(Bill_Amount) FROM Bill_Rent_Table", con);
164         1DataTable dt = new DataTable();
165         1sda.Fill(dt);
166         1lbl_Rent_Total.Text = dt.Rows[0][0].ToString();
167         1con.Close();
168     }
169
170
171     1reference
172     private void fill_combo_Rent() //ComboBox fill with sellers name
173     {
174         1con.open();
175         1SqlCommand cmd = new SqlCommand("SELECT Driver_Username FROM Details_Drivers_Table", con);
176         1SqlDataReader dr;
177         1dr = cmd.ExecuteReader();
178         1dt = new DataTable();
179         1dt.Columns.Add("cm_Renting", typeof(string));
180         1dt.Load(dr);
181         1cmb_Renting.ValueMember = "Driver_Username";
182         1cmb_Renting.DataSource = dt;
183         1con.Close();
184     }
185
186     1reference
187     public void Get_Rent_Amount_By_Driver() //TextBox link with the Combo Box
188     {
189         1con.open();
190         1SqlDataAdapter sda = new SqlDataAdapter("SELECT SUM(Bill_Amount) FROM Bill_Rent_Table WHERE Driver_Name = '" + cmb_Renting.SelectedValue.ToString() + "'", con);
191         1DataTable dt = new DataTable();
192         1sda.Fill(dt);
193         1lbl_Rents_By_Driver.Text = dt.Rows[0][0].ToString();
194         1con.Close();
195     }
196
197     1reference
198     private void cmb_Renting_SelectionChangeCommitted(object sender, EventArgs e) //According to ComboBox value change the TextBox value (ComboBox Event"SelectionChangeCommitted")
199     {
200         1Get_Rent_Amount_By_Driver();
201     }
202
203
204 //***** Dashboard Financial Panel Details for One Day Wise *****
205
```

Figure 4. 15 Admin logged Dashboard Form Code - part 4

Figure 4. 16 Admin logged Dashboard Form Code - part 5

```
Program.cs ✘ Dashboard.cs ✘ Dashboard.cs [Design] - Rent_Car.Dashboard + Get_Rent_Amount_By_Driver
rent_car
=====
245 //***** Dashboard Financial Panel Details For Your Hires *****
246
247 :reference
248 public void Get_LongD_Total()
249 {
250     : con.Open();
251     SqlDataReader sda = new SqlDataReader("SELECT SUM($Bill_Amount) FROM Bill_LongD_Hire_Table", con);
252     DataTable dt = new DataTable();
253     dt.Fill(dt);
254     lbl_Tour.Total.Text = dt.Rows[0][0].ToString();
255 }
256
257
258
259 :reference
260 private void fill_combo_LongD() //ComboBox fill with sellers name
261 {
262     : con.Open();
263     SqlCommand cmd = new SqlCommand("SELECT Driver_Username FROM Details_Drivers_Table", con);
264     SqlDataReader dr;
265     dr = cmd.ExecuteReader();
266     DataTable dt = new DataTable();
267     dt.Columns.Add("cmb_Tour", typeof(string));
268     cmb_Tour.ValueMember = "Driver_Username";
269     cmb_Tour.DataSource = dt;
270     con.Close();
271 }
272
273 :references
274 public void Get_LongD_Amount_By_Driver() //TextBox link with the Combo Box
275 {
276     : con.Open();
277     SqlDataReader sda = new SqlDataReader("SELECT SUM($Bill_Amount) FROM Bill_LongD_Hire_Table WHERE Driver_Name = '" + cmb_Tour.SelectedValue.ToString() + "'", con);
278     DataTable dt = new DataTable();
279     dt.Fill(dt);
280     lbl_Tours_by_Driver.Text = dt.Rows[0][0].ToString();
281     con.Close();
282 }
283
284 :reference
285 private void cmb_Tour_SelectionchangeCommitted(object sender, EventArgs e)
286 {
287     Get_LongD_Amount_By_Driver();
288 }
289
290
291 //***** Award Panel Details *****
292
293
294 :reference
295 private void Best_Rent_Driver()
```

Figure 4. 17 Admin logged Dashboard Form Code - part 6

```

290
291
292
293
294     [Reference]
295     private void Best_Rent_Driver()
296     {
297         try
298         {
299             con.Open();
300             string InnerQuery = "SELECT MAX($11_Amount) FROM Bill_Rent_Table";
301             DataTable dt1 = new DataTable();
302             SqlDataAdapter sda1 = new SqlDataAdapter(InnerQuery, con);
303             sda1.Fill(dt1);
304
305             string Query = "SELECT Driver_Name FROM Bill_Rent_Table WHERE Bill_Amount = '" + dt1.Rows[0][0].ToString() + "'";
306             SqlDataAdapter sda = new SqlDataAdapter(Query, con);
307             DataTable dt = new DataTable();
308             sda.Fill(dt);
309
310             lbl_Best_Rent.Text = dt.Rows[0][0].ToString();
311             con.Close();
312         }
313         catch (Exception ex)
314         {
315             MessageBox.Show(ex.Message);
316             con.Close();
317         }
318     }
319
320     [Reference]
321     private void Best_OneD_Driver()
322     {
323         try
324         {
325             con.Open();
326             string InnerQuery = "SELECT MAX($11_Amount) FROM Bill_OneD_Hire_Table";
327             DataTable dt1 = new DataTable();
328             SqlDataAdapter sda1 = new SqlDataAdapter(InnerQuery, con);
329             sda1.Fill(dt1);
330
331             string Query = "SELECT Driver_Name FROM Bill_OneD_Hire_Table WHERE Bill_Amount = '" + dt1.Rows[0][0].ToString() + "'";
332             SqlDataAdapter sda = new SqlDataAdapter(Query, con);
333             DataTable dt = new DataTable();
334             sda.Fill(dt);
335
336             lbl_Best_OneD.Text = dt.Rows[0][0].ToString();
337             con.Close();
338         }
339         catch (Exception ex)
340         {
341             MessageBox.Show(ex.Message);
342             con.Close();
343         }
344     }

```

Figure 4. 18 Admin logged Dashboard Form Code - part 7

```

329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345     [Reference]
346     private void Best_LongD_Driver()
347     {
348         try
349         {
350             con.Open();
351             string InnerQuery = "SELECT MAX($11_Amount) FROM Bill_LongD_Hire_Table";
352             DataTable dt1 = new DataTable();
353             SqlDataAdapter sda1 = new SqlDataAdapter(InnerQuery, con);
354             sda1.Fill(dt1);
355
356             string Query = "SELECT Driver_Name FROM Bill_LongD_Hire_Table WHERE Bill_Amount = '" + dt1.Rows[0][0].ToString() + "'";
357             SqlDataAdapter sda = new SqlDataAdapter(Query, con);
358             DataTable dt = new DataTable();
359             sda.Fill(dt);
360
361             lbl_Best_LongD.Text = dt.Rows[0][0].ToString();
362             con.Close();
363         }
364         catch (Exception ex)
365         {
366             MessageBox.Show(ex.Message);
367             con.Close();
368         }
369     }
370
371

```

Figure 4. 19 Admin logged Dashboard Form Code - part 8

4.1.6 Admin logged Vehicles Form Design and Code

Ayubo Drive

Vehicles List

Veh_ID	Veh_Type	Veh_RegNo	Veh_Brand	Veh_Model	Veh_Colour	Own_Name	Own_ID	Own_Phone
1	Car	KD 7275	Nissan	FB 15	Grey	Kelum	745736735V	077333076
2	SUV	CAR 3553	Toyota	Compact	White	Anam	705967735V	0773959457
3	Van	PG 6587	Toyota	Lite Ace	Black	Kapila	805635785V	0747356845
4	Bus	LT 7653	TATA	Leyland	Blue	Saman	727563456V	0716457452
*								

Figure 4. 20 Admin logged Vehicles Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class Vehicles_Register_Form : Form
15     {
16         //reference
17         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
18         SqlCommand cmd;
19         SqlDataAdapter adpt;
20         DataTable dt;
21
22         //***** Quick Menu BUTTONS *****
23
24         private void btn_dashboard_Click(object sender, EventArgs e)
25         {
26             Dashboard obj = new Dashboard();
27             obj.Show();
28             this.Hide();
29         }
30
31
32
33         private void btn_renting_Click(object sender, EventArgs e)
34         {
35             Give_Renting_Form obj = new Give_Renting_Form();
36             obj.Show();
37             this.Hide();
38         }
39
40
41         private void btn_10_Hire_Click(object sender, EventArgs e)
42         {
43             Give_ID_Hiring_Form obj = new Give_ID_Hiring_Form();
44             obj.Show();
45             this.Hide();
46         }
47
48
49         private void btn_Your_Hire_Click(object sender, EventArgs e)
50         {
51             Give_Long_Hiring_Form obj = new Give_Long_Hiring_Form();
52             obj.Show();
53             this.Hide();
54         }
55
56
57         private void btn_drivers_Click(object sender, EventArgs e)
58         {
59             Drivers_Register_Form obj = new Drivers_Register_Form();
60             obj.Show();
61             this.Hide();
62         }
63
64
65         private void btn_logout_Click(object sender, EventArgs e)
66         {
67             Login_Form obj = new Login_Form();
68             obj.Show();
69             this.Hide();
70         }
71
72
73         //***** OTHER METHODS *****
74
75         public void clear()
76         {
77             txt_vehID.Text = "";
78             cmb_type.Text = "";
79             txt_color.Text = "";
80             cmb_brand.Text = "";
81             txt_model.Text = "";
82             cmb_colour.Text = "";
83             txt_name.Text = "";
84             txt_own_id.Text = "";
85             txt_own_phone.Text = "";
86         }
87
88
89
90
91
92
93

```

Figure 4. 21 Admin logged Vehicles Form Code - part 1

```

49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93

```

Figure 4. 22 Admin logged Vehicles Form Code - part 2

```

Program.cs 9 Vehicles_Register_Form.cs 9 Vehicles_Register_Form.cs [Design]
Rent Car Rent_Car.Vehicles_Register_Form Vehicles_Register_Form
private void display_data_grid_view() //For the data grid view
{
    try
    {
        dt = new DataTable();
        con.Open();
        adap = new SqlDataAdapter("SELECT * FROM Details_Vehicle_Table", con);
        adap.Fill(dt);
        dgv_Vehicles_List.DataSource = dt;
        con.Close();
    }
    catch (Exception ex)
    {
        MessageBox.Show(ex.Message);
        con.Close();
    }
}
private void dgv_Vehicles_List_CellClick(object sender, DataGridViewCellEventArgs e)
{
    con.Open();
    int ID;
    ID = int.Parse(dgv_Vehicles_List.Rows[e.RowIndex].Cells[0].Value.ToString());
    SqlCommand cmd = con.CreateCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "SELECT * FROM Details_Vehicle_Table WHERE Veh_ID = '" + ID + "' ";
    SqlDataAdapter DR1 = cmd.ExecuteReader();
    if (DR1.Read())
    {
        txt_VehID.Text = DR1.GetValue(0).ToString();
        cmb_type.Text = DR1.GetValue(1).ToString();
        txt_Regno.Text = DR1.GetValue(2).ToString();
        cmb_Brand.Text = DR1.GetValue(3).ToString();
        txt_Model.Text = DR1.GetValue(4).ToString();
        cmb_Colour.Text = DR1.GetValue(5).ToString();
        txt_Own_name.Text = DR1.GetValue(6).ToString();
        txt_Own_id.Text = DR1.GetValue(7).ToString();
        txt_Own_phone.Text = DR1.GetValue(8).ToString();
    }
    DR1.Close();
    con.Close();
}
//*****SAVE*****Edit*****Delete*****

```

Figure 4. 23 Admin logged Vehicles Form Code - part 3

```

Program.cs 9 Vehicles_Register_Form.cs 9 Vehicles_Register_Form.cs [Design]
Rent Car Rent_Car.Vehicles_Register_Form Vehicles_Register_Form
//*****SAVE*****Edit*****Delete*****
private void btn_Save_Click(object sender, EventArgs e)
{
    if (cmb_type.Text == "" || txt_Regno.Text == "" || cmb_Brand.Text == "" || txt_Own_name.Text == "" || txt_Own_id.Text == "" || txt_Own_phone.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("INSERT INTO Details_Vehicle_Table(Veh_Type,Veh_Regno,Veh_Brand,Veh_Model,Veh_Colour,Own_Name,Own_ID,Own_Phone) VALUES('" + cmb_type.Text + "','" + txt_Regno.Text + "','" + cmb_Brand.Text + "','" + txt_Model.Text + "','" + cmb_Colour.Text + "','" + txt_Own_name.Text + "','" + txt_Own_id.Text + "','" + txt_Own_phone.Text + "')", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Vehicle added successfully!!!");
            display_data_grid_view(); //data grid view method
            clear(); //data clear method
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            con.Close();
        }
    }
}
private void btn_Edit_Click(object sender, EventArgs e)
{
    if (cmb_type.Text == "" || txt_Regno.Text == "" || cmb_Brand.Text == "" || cmb_Colour.Text == "" || txt_Own_name.Text == "" || txt_Own_id.Text == "" || txt_Own_phone.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("UPDATE Details_Vehicle_Table SET Veh_Type = '" + cmb_type.Text + "', Veh_Regno = '" + txt_Regno.Text + "', Veh_Brand = '" + cmb_Brand.Text + "', Veh_Model = '" + txt_Model.Text + "', Veh_Colour = '" + cmb_Colour.Text + "', Own_Name = '" + txt_Own_name.Text + "', Own_ID = '" + txt_Own_id.Text + "', Own_Phone = '" + txt_Own_phone.Text + "' WHERE Veh_ID = '" + txt_VehID.Text + "'", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Vehicle edit successfully!!!");
            display_data_grid_view(); //data grid view method
            clear(); //data clear method
        }
        catch (Exception ex)
        {

```

Figure 4. 24 Admin logged Vehicles Form Code - part 4

```
Program.cs # Vehicles_Register_Form.cs < Rent_Car Vehicles_Register_Form.cs [Design] Rent_Car Vehicles_Register_Form + ⌂ Vehicles_Register_Form

195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
}
}

catch (Exception ex)
{
    MessageBox.Show(ex.Message);
    con.Close();
}

}

private void btn_Delete_Click(object sender, EventArgs e)
{
    if (cmb_type.Text == "")
    {
        MessageBox.Show("Select VEHICLE to Delete");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("DELETE FROM Details_Vehicle_Table WHERE Veh_ID = '" + txt_vehID.Text + "' ", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Vehicle delete successfully!!!");

            display_data_grid_view(); //data grid view method
            clear(); //data clear method
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            con.Close();
        }
    }
}

private void btn_Clear_Click(object sender, EventArgs e)
{
    txt_vehID.Text = "";
    cmb_type.Text = "";
    txt_rego.Text = "";
    txt_name.Text = "";
    txt_model.Text = "";
    cmb_colour.Text = "";
    txt_own_name.Text = "";
    txt_own_email.Text = "";
    txt_own_phone.Text = "";
}

}

```

Figure 4. 25 Admin logged Vehicles Form Code - part 5

4.1.7 Admin logged Rent Details Form Design and Code

Veh_ID	Veh_Type	Veh_RegNo	Veh_Brand	Veh_Model	Veh_Colour	Own_Name	Own_ID	Own_Phone
1	Car	KD 7275	Nissan	FB 15	Grey	Kelum	745736735V	0773733076
2	SUV	CAR 3553	Toyota	Cimpact	White	Anam	7059877235V	0773989457
3	Van	PG 6587	Toyota	Lite Ace	Black	Kapila	805635785V	0747356845
4	Bus	LT 7653	TATA	Leyland	Blue	Saman	727563456V	0716457452

Vehicle Details				Save	
Type	Brand	Model	Colour	Edit	Delete
Car	Nissan	FB 15	Grey		
Renting ID				Clear	

Renting Rates			
Daily	Weekly	Monthly	Driver Rate

Rent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	SUV	Toyota	Cimpact	White	4000	20000	80000	2000
3	Van	Toyota	Lite Ace	Black	2500	12000	60000	1500
4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Figure 4. 26 Admin logged Rent Details Form Design

Program.cs 9 Give_Renting_Form.cs 0 Give_Renting_Form.cs [Design] Rent_Car - Rent_Car.Give_Renting_Form Give_Renting_Form

```

1  using System;
2  using System.Collections.Generic;
3  using System.Data;
4  using System.Drawing;
5  using System.Linq;
6  using System.Text;
7  using System.Threading.Tasks;
8  using System.Windows.Forms;
9  using System.Data.SqlClient;
10
11  namespace Rent_Car
12  {
13      public partial class Give_Renting_Form : Form
14      {
15          public Give_Renting_Form()
16          {
17              InitializeComponent();
18
19              display_data_grid_view_Vehicle_List();
20              display_data_grid_view_Rent_List();
21          }
22
23
24          SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
25          SqlCommand cmd;
26
27
28          //For data grid view
29          SqlDataAdapter adapt;
30          DataTable dt;
31
32
33
34          //***** Quick Menu BUTTONS *****
35
36          private void btn_dashboard_Click(object sender, EventArgs e)
37          {
38              Dashboard obj = new Dashboard();
39              obj.Show();
40              this.Hide();
41          }
42
43          private void btn_vehicles_Click(object sender, EventArgs e)
44          {
45              Vehicles_Register_Form obj = new Vehicles_Register_Form();
46              obj.Show();
47              this.Hide();
48          }
49

```

Figure 4. 27 Admin logged Rent Details Form Code - part 1

Program.cs 9 Give_Renting_Form.cs 0 Give_Renting_Form.cs [Design] Rent_Car - Rent_Car.Give_Renting_Form Give_Renting_Form

```

49
50          private void btn_ID_Hire_Click(object sender, EventArgs e)
51          {
52              Give_ID_Hiring_Form obj = new Give_ID_Hiring_Form();
53              obj.Show();
54              this.Hide();
55          }
56
57          private void btn_Tour_Hire_Click(object sender, EventArgs e)
58          {
59              Give_Long_Hiring_Form obj = new Give_Long_Hiring_Form();
60              obj.Show();
61              this.Hide();
62          }
63
64          private void btn_drivers_Click(object sender, EventArgs e)
65          {
66              Drivers_Register_Form obj = new Drivers_Register_Form();
67              obj.Show();
68              this.Hide();
69          }
70
71          private void btn_Logout_Click(object sender, EventArgs e)
72          {
73              Login_Form obj = new Login_Form();
74              obj.Show();
75              this.Hide();
76          }
77
78          //***** OTHER METHODS *****
79
80          public void clear()
81          {
82              txt_rentID.Text = "";
83              txt_type.Text = "";
84              txt_mileage.Text = "";
85              txt_model.Text = "";
86              txt_colour.Text = "";
87              txt_dvRent.Text = "";
88              txt_weekRent.Text = "";
89              txt_monthRent.Text = "";
90              txt_driverRate.Text = "";
91
92          }

```

Figure 4. 28 Admin logged Rent Details Form Code - part 2

```
Program.cs # Give_Renting_Form.cs × Give_Renting_Form.cs[Design] - Rent_Car.Give_Renting_Form - Give_Renting_Form()
```

```
93 //FOR 1ST DATA GRID VIEW
94 //4 references
95 public void display_data_Grid_view_Vehicle_List() //For the data grid view
96 {
97     try
98     {
99         dt = new DataTable();
100         con = new SqlConnection();
101         cmd = new SqlCommand("SELECT * FROM Details_Vehicle_Table", con);
102         adapt = new SqlDataAdapter("SELECT * FROM Details_Vehicle_Table", con);
103         adapt.Fill(dt);
104         dgv_Vehicles_List.DataSource = dt;
105         con.Close();
106     }
107     catch (Exception ex)
108     {
109         MessageBox.Show(ex.Message);
110         con.Close();
111     }
112 }
113 //Inference
114 private void dgv_Vehicles_List_CellClick(object sender, DataGridViewCellEventArgs e)
115 {
116     con.Open();
117     int ID;
118     ID = int.Parse(dgv_Vehicles_List.Rows[e.RowIndex].Cells[0].Value.ToString());
119     SqlCommand cmd = con.CreateCommand();
120     cmd.CommandType = CommandType.Text;
121     cmd.CommandText = "SELECT * FROM Details_Vehicle_Table WHERE Veh_ID = '" + ID + "' ";
122     SqlDataReader DR1 = cmd.ExecuteReader();
123     if (DR1.Read())
124     {
125         txt_type.Text = DR1.GetValue(1).ToString();
126         txt_brand.Text = DR1.GetValue(3).ToString();
127         txt_model.Text = DR1.GetValue(4).ToString();
128         txt_colour.Text = DR1.GetValue(5).ToString();
129     }
130     DR1.Close();
131     con.Close();
132 }
```

Figure 4. 29 Admin logged Rent Details Form Code - part 3

```
Program.cs # Give_Renting_Form.cs x Give_Renting_Form.cs[Design] - Rent_Car - Rent_Car.Give_Renting_Form - dgv_Vehicles_List_CellClick(object sender, DataGridViewCellEventArgs e)
137
138
139
140 //FOR 2nd DATA GRID VIEW
141 4 references
142 public void display_data_grid_view_Rent_List() //For the data grid view
143 {
144     try
145     {
146         dt = new DataTable();
147         Con.Open();
148         adapt = new SqlDataAdapter("SELECT * FROM Details_Renting_Table", con);
149         adapt.Fill(dt);
150         dgv_Renting_List.DataSource = dt;
151         Con.Close();
152     }
153     catch (Exception ex)
154     {
155         MessageBox.Show(ex.Message);
156         con.Close();
157     }
158 }
159
160 //reference
161 private void dgv_Renting_list_CellClick(object sender, DataGridViewCellEventArgs e)
162 {
163     con.open();
164     int ID;
165
166     ID = int.Parse(dgv_Renting_list.Rows[e.RowIndex].Cells[0].Value.ToString());
167
168     SqlCommand cmd = con.CreateCommand();
169     cmd.CommandType = CommandType.Text;
170     cmd.CommandText = "SELECT * FROM Details_Renting_Table WHERE Rent_ID = " + ID + " ";
171
172     SqlDataReader DR1 = cmd.ExecuteReader();
173
174     if (DR1.Read())
175     {
176         txt_RentID.Text = DR1.getvalue(0).ToString();
177         txt_RentDate.Text = DR1.getvalue(1).ToString();
178         txt_brand.Text = DR1.getvalue(2).ToString();
179         txt_model.Text = DR1.getvalue(3).ToString();
180         txt_colour.Text = DR1.getvalue(4).ToString();
181         txt_driverName.Text = DR1.getvalue(5).ToString();
182         txt_weekRent.Text = DR1.getvalue(6).ToString();
183         txt_monthRent.Text = DR1.getvalue(7).ToString();
184         cmb_driverRate.Text = DR1.getvalue(8).ToString();
185     }
186
187     DR1.Close();
188     con.Close();
189 }
```

Figure 4. 30 Admin logged Rent Details Form Code - part 4

Figure 4. 31 Admin logged Rent Details Form Code - part 5

Figure 4. 32 Admin logged Rent Details Form Code - part 6

4.1.8 Admin logged One Day Hire Details Form Design and Code

Veh_ID	Veh_Type	Veh_RegNo	Veh_Brand	Veh_Model	Veh_Colour	Own_Name	Own_ID	Own_Phone
1	Car	KD 7275	Nissan	FB 15	Grey	Kelum	745736725V	0773733076
2	SUV	CAR 3553	Toyota	Cimpact	White	Anam	705967725V	0773963457
3	Van	PG 6587	Toyota	Lite Ace	Black	Kapila	805635785V	0747356345
4	Bus	LT 7653	TATA	Leyland	Blue	Saman	727563456V	0716457452

Type	Brand	Model	Colour
Van	Toyota	Lite Ace	Black

Packages	Per km Charge	Maximum km Limit	Extra km Charge

OneD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	OneD_Package	OneD_Per_Km	OneD_Max_Km	OneD_Extra_Km	OneD_Max_Hr	OneD_Extra_Hr
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per ...	100	300	120	12	200

Figure 4. 33 Admin logged One Day Hire Details Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class Give_ID_Hiring_Form : Form
15     {
16         public Give_ID_Hiring_Form()
17         {
18             InitializeComponent();
19         }
20
21         display_data_grid_view_Vehicle_List();
22         display_data_grid_view_OHO_Hire_List();
23
24         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
25         SqlCommand cmd;
26
27         //For data grid view
28         SqlDataAdapter adapt;
29         DataTable dt;
30
31
32         //***** Quick Menu BUTTONS *****
33
34         reference
35         private void btn_dashboard_Click(object sender, EventArgs e)
36         {
37             Dashboard obj = new Dashboard();
38             obj.Show();
39             this.Hide();
40         }
41
42         reference
43         private void btn_vehicles_Click(object sender, EventArgs e)
44         {
45             Vehicles_Register_Form obj = new Vehicles_Register_Form();
46             obj.Show();
47             this.Hide();
48         }
49
50         reference
51         private void btn_renting_Click(object sender, EventArgs e)
52         {
53             Give_Renting_Form obj = new Give_Renting_Form();
54         }

```

Figure 4. 34 Admin logged One Day Hire Details Form Code - part 1

```

51         Give_Renting_Form obj = new Give_Renting_Form();
52         obj.Show();
53         this.Hide();
54     }
55
56     reference
57     private void btn_Your_Hire_Click(object sender, EventArgs e)
58     {
59         Give_Long_Hiring_Form obj = new Give_Long_Hiring_Form();
60         obj.Show();
61         this.Hide();
62     }
63
64     reference
65     private void btn_drivers_Click(object sender, EventArgs e)
66     {
67         Drivers_Register_Form obj = new Drivers_Register_Form();
68         obj.Show();
69         this.Hide();
70     }
71
72     reference
73     private void btn_Logout_Click(object sender, EventArgs e)
74     {
75         Login_Form obj = new Login_Form();
76         obj.Show();
77         this.Hide();
78     }
79
80     references
81     public void clear()
82     {
83         txt_One_Day.Text = "";
84         txt_SepetText = "";
85         txt_brand.Text = "";
86         txt_model.Text = "";
87         txt_colour.Text = "";
88         cmb_packages.Text = "";
89         txt_max_charge.Text = "";
90         txt_Max_Km_limit.Text = "";
91         txt_Extra_km_charge.Text = "";
92         txt_Max_Hour_limit.Text = "";
93         txt_Extra_hour_charge.Text = "";
94     }
95
96     //For 1st DATA GRID VIEW
97     reference
98     public void display_data_grid_view_Vehicle_List() //For the data grid view
99     {
100         try
101         {
102             dt = new DataTable();
103             com.open();
104         }
105     }

```

Figure 4. 35 Admin logged One Day Hire Details Form Code - part 2

```

Program.cs 9 × Give_ID_Hiring_Form.cs 10 × Give_ID_Hiring_Form.cs [Design]
Rent Car Rent_Car.Give_ID_Hiring_Form Give_ID_Hiring_Form
181     command = new SqlCommand("SELECT * FROM Details_Vehicle_Table", con);
182     adapt = new SqlDataAdapter("SELECT * FROM Details_Vehicle_Table", con);
183     dgv_Vehicles_List.DataSource = dt;
184     con.Close();
185   }
186   catch (Exception ex)
187   {
188     MessageBox.Show(ex.Message);
189     con.Close();
190   }
191 }
192 //Reference
193 private void dgv_Vehicles_List_CellClick(object sender, DataGridViewCellEventArgs e)
194 {
195   con.Open();
196   int ID;
197   ID = int.Parse(dgv_Vehicles_List.Rows[e.RowIndex].Cells[0].Value.ToString());
198   SqlCommand cmd = con.CreateCommand();
199   cmd.CommandType = CommandType.Text;
200   cmd.CommandText = "SELECT * FROM Details_Vehicle_Table WHERE Veh_ID = '" + ID + "' ";
201   SqlDataReader DR1 = cmd.ExecuteReader();
202   if (DR1.Read())
203   {
204     txt_Type.Text = DR1.GetValue(1).ToString();
205     txt_Brand.Text = DR1.GetValue(3).ToString();
206     txt_Model.Text = DR1.GetValue(4).ToString();
207     txt_Colour.Text = DR1.GetValue(5).ToString();
208   }
209   DR1.Close();
210   con.Close();
211 }
212 //FOR Grid View
213 public void display_data_grid_view_OneD_Hire_List() //For the data grid view
214 {
215   try
216   {
217     dt = new DataTable();
218     con.Open();
219     adapt = new SqlDataAdapter("SELECT * FROM Details_OneD_Hire_Table", con);
220     dgv_OneD_Hire_List.DataSource = dt;
221     con.Close();
222   }
223   catch (Exception ex)
224   {

```

Figure 4. 36 Admin logged One Day Hire Details Form Code - part 3

```

Program.cs 9 × Give_ID_Hiring_Form.cs 10 × Give_ID_Hiring_Form.cs [Design]
Rent Car Rent_Car.Give_ID_Hiring_Form Give_ID_Hiring_Form
154   {
155     MessageBox.Show(ex.Message);
156     con.Close();
157   }
158 }
159
160 //Reference
161 private void dgv_OneD_Hire_List_CellClick(object sender, DataGridViewCellEventArgs e)
162 {
163   con.Open();
164   int ID;
165   ID = int.Parse(dgv_OneD_Hire_List.Rows[e.RowIndex].Cells[0].Value.ToString());
166   SqlCommand cmd = con.CreateCommand();
167   cmd.CommandType = CommandType.Text;
168   cmd.CommandText = "SELECT * FROM Details_OneD_Hire_Table WHERE OneD_ID = '" + ID + "' ";
169   SqlDataReader DR1 = cmd.ExecuteReader();
170   if (DR1.Read())
171   {
172     txt_OneD_ID.Text = DR1.GetValue(1).ToString();
173     txt_Type.Text = DR1.GetValue(2).ToString();
174     txt_Brand.Text = DR1.GetValue(3).ToString();
175     txt_Model.Text = DR1.GetValue(4).ToString();
176     txt_Colour.Text = DR1.GetValue(5).ToString();
177     cmb_Packages.Text = DR1.GetValue(6).ToString();
178     txt_Per_Km_Charge.Text = DR1.GetValue(7).ToString();
179     txt_Max_Km_Limit.Text = DR1.GetValue(8).ToString();
180     txt_Extra_Km_Charge.Text = DR1.GetValue(9).ToString();
181     txt_Max_Hour_Limit.Text = DR1.GetValue(10).ToString();
182     txt_Extra_Hour_Charge.Text = DR1.GetValue(11).ToString();
183   }
184   DR1.Close();
185   con.Close();
186 }
187
188 //*****SAVE*****Edit*****Delete*****
189 private void btn_Save_Click(object sender, EventArgs e)
190 {
191   if ((txt_Type.Text == "") || (txt_Brand.Text == "") || (txt_Model.Text == "") || (txt_Colour.Text == "") || (cmb_Packages.Text == "") || (txt_Per_Km_Charge.Text == "") || (txt_Max_Km_Limit.Text == "") || (txt_Extra_Km_Charge.Text == "") || (txt_Max_Hour_Limit.Text == "") || (txt_Extra_Hour_Charge.Text == ""))
192   {
193     MessageBox.Show("Missing Information");
194   }
195   else
196   {
197     try
198     {
199       con.Open();
200     }
201     catch (Exception ex)
202     {
203       MessageBox.Show(ex.Message);
204     }
205   }
206 }

```

Figure 4. 37 Admin logged One Day Hire Details Form Code - part 4

```

Program.cs 9 Give_1D_Hiring_Form.cs -> Give_1D_Hiring_Form.cs [Design]
Rent Car Rent_Car.Give_1D_Hiring_Form
    0 btn_Edit_Click(object sender, EventArgs e)
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
}
}

private void btn_Edit_Click(object sender, EventArgs e)
{
    if (txt_type.Text == "" || txt_brand.Text == "" || txt_model.Text == "" || txt_colour.Text == "" || cmb_packages.Text == "" || txt_Per_Km_charge.Text == "" || txt_Max_Km_limit.Text == "" || txt_Extra_Km_charge.Text == "" || txt_Max_Hour_limit.Text == "" || txt_Extra_Hour_charge.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("UPDATE Details_OneD_Hire_Table SET Veh_Type = '" + txt_type.Text + "' , Veh_Brand = '" + txt_brand.Text + "' , Veh_Model = '" + txt_model.Text + "' , "
                + "Veh_Colour = '" + txt_colour.Text + "' , OneD_Packages = '" + cmb_packages.Text + "' , OneD_Per_Km_charge = '" + txt_Per_Km_charge.Text + "' , "
                + "OneD_Max_Km_limit = '" + txt_Max_Km_limit.Text + "' , OneD_Extra_Km_charge = '" + txt_Extra_Km_charge.Text + "' , OneD_Max_Hour_limit = '" + txt_Max_Hour_limit.Text + "' , "
                + "OneD_Extra_Hour_charge = '" + txt_Extra_Hour_charge.Text + "' WHERE OneD_ID = '" + txt_OneD_ID.Text + "' ", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("One Day Package edit successfully!!!!");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            con.Close();
        }
    }
}
}

private void btn_Delete_Click(object sender, EventArgs e)
{
    if (cmb_packages.Text == "")
    {
        MessageBox.Show("Select One Day Package to Delete");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("DELETE FROM Details_OneD_Hire_Table WHERE OneD_ID = '" + txt_OneD_ID.Text + "' ", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("One Day Package delete successfully!!!!");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            con.Close();
        }
    }
}
}

private void btn_Clear_Click(object sender, EventArgs e)
{
    txt_OneD_ID.Text = "";
    txt_type.Text = "";
    txt_brand.Text = "";
    txt_model.Text = "";
    txt_colour.Text = "";
    cmb_packages.Text = "";
    txt_Per_Km_charge.Text = "";
    txt_Max_Km_limit.Text = "";
    txt_Extra_Km_charge.Text = "";
    txt_Max_Hour_limit.Text = "";
    txt_Extra_Hour_charge.Text = "";
}
}
}

```

Figure 4. 38 Admin logged One Day Hire Details Form Code - part 5

```

Program.cs 9 Give_1D_Hiring_Form.cs -> Give_1D_Hiring_Form.cs [Design]
Rent Car Rent_Car.Give_1D_Hiring_Form
    0 btn_Edit_Click(object sender, EventArgs e)
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
}
}

private void btn_Delete_Click(object sender, EventArgs e)
{
    if (cmb_packages.Text == "")
    {
        MessageBox.Show("Select One Day Package to Delete");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("DELETE FROM Details_OneD_Hire_Table WHERE OneD_ID = '" + txt_OneD_ID.Text + "' ", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("One Day Package delete successfully!!!!");
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
            con.Close();
        }
    }
}
}

private void btn_Clear_Click(object sender, EventArgs e)
{
    txt_OneD_ID.Text = "";
    txt_type.Text = "";
    txt_brand.Text = "";
    txt_model.Text = "";
    txt_colour.Text = "";
    cmb_packages.Text = "";
    txt_Per_Km_charge.Text = "";
    txt_Max_Km_limit.Text = "";
    txt_Extra_Km_charge.Text = "";
    txt_Max_Hour_limit.Text = "";
    txt_Extra_Hour_charge.Text = "";
}
}
}

```

Figure 4. 39 Admin logged One Day Hire Details Form Code - part 6

4.1.9 Admin logged Long Tour Hire Details Form Design and Code

Vehicles List

Veh_ID	Veh_Type	Veh_RegNo	Veh_Brand	Veh_Model	Veh_Colour	Own_Name	Own_ID	Own_Phone
1	Car	KD 7275	Nissan	FB 15	Grey	Kelum	745736735V	0773733076
2	SUV	CAR 3553	Toyota	Cpmpact	White	Anam	705967735V	0773969457
3	Van	PG 6587	Toyota	Lite Ace	Black	Kapila	805635785V	0747356845
4	Bus	LT 7653	TATA	Leyland	Blue	Saman	727563456V	0716457452

Vehicle Details

Type	Brand	Model	Colour
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Vehicle Package ID: <input type="text"/>			
One Day Hiring Rates			
Packages	Per km Charge	Maximum km Limit	Extra km Charge
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>
Driver Charge Overnight: <input type="text"/>		Vehicle Parking Charge: <input type="text"/>	

Long Hiring Details

LongD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	LongD_Package	LongD_Per_Km	LongD_Max_Km	LongD_Extra_Km	LongD_Driver_C	LongD_Vehicle
1	Car	Nissan	FB 15	Grey	2 days trip	100	200	120	1000	200
2	Car	Nissan	FB 15	Grey	3 days trip	100	300	120	1000	200
3	Car	Nissan	FB 15	Grey	4 days trip	100	400	120	1000	200
4	Car	Nissan	FB 15	Grey	5 days trip	100	500	120	1000	200
5	Car	Nissan	FB 15	Grey	6 days trip	100	600	120	1000	200
6	Car	Nissan	FB 15	Grey	7 days trip	100	700	120	1000	200

Figure 4. 40 Admin logged Long Tour Hire Details Form Design

```

Program.cs 0 X Give_Long_Hiring_Form.cs 0 X Give_Long_Hiring_Form.cs [Design]* Rent_Car
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class Give_Long_Hiring_Form : Form
15     {
16         public Give_Long_Hiring_Form()
17         {
18             InitializeComponent();
19             display_data_grid_view_Vehicle_List();
20             display_data_grid_view_LongHire_List();
21         }
22
23         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVNU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
24
25         SqlCommand cmd;
26
27         //For data grid view
28         SqlDataAdapter adapt;
29         DataTable dt;
30
31
32         //***** Quick Menu BUTTONS *****
33
34         //Inference
35         private void btn_dashboard_Click(object sender, EventArgs e)
36         {
37             Dashboard obj = new Dashboard();
38             obj.Show();
39             this.Hide();
40         }
41
42         //Inference
43         private void btn_vehicles_Click(object sender, EventArgs e)
44         {
45             Vehicles_Register_Form obj = new Vehicles_Register_Form();
46             obj.Show();
47             this.Hide();
48         }
49
50         //Inference
51         private void btn_renting_Click(object sender, EventArgs e)
52         {
53             Give_Renting_Form obj = new Give_Renting_Form();

```

Figure 4. 41 Admin logged Long Tour Hire Details Form Code - part 1

```

Program.cs 0 X Give_Long_Hiring_Form.cs 0 X Give_Long_Hiring_Form.cs [Design]* Rent_Car
54     }
55
56     //Inference
57     private void btn_ID_Hire_Click(object sender, EventArgs e)
58     {
59         Give_ID_Hiring_Form obj = new Give_ID_Hiring_Form();
60         obj.Show();
61         this.Hide();
62     }
63
64     //Inference
65     private void btn_drivers_Click(object sender, EventArgs e)
66     {
67         Drivers_Register_Form obj = new Drivers_Register_Form();
68         obj.Show();
69         this.Hide();
70     }
71
72     //Inference
73     private void btn_Logout_Click(object sender, EventArgs e)
74     {
75         Login_Form obj = new Login_Form();
76         obj.Show();
77         this.Hide();
78     }
79
80     //***** OTHER METHODS *****
81
82     public void clear()
83     {
84         txt_LongID.Text = "";
85         txt_Type.Text = "";
86         txt_Make.Text = "";
87         txt_Model.Text = "";
88         txt_Colour.Text = "";
89         cmb_Packages.Text = "";
90         txt_Driver_charge.Text = "";
91         txt_Max_Km_limit.Text = "";
92         txt_Extra_Km_charge.Text = "";
93         txt_Driver_overnight_charge.Text = "";
94         txt_Vehicle_parking_charge.Text = "";
95
96         //FOR 1ST DATA GRID VIEW
97         //Inference
98         public void display_data_grid_view_Vehicle_List() //For the data grid view
99         {
100             try
101             {
102                 dt = new DataTable();
103                 con.open();
104             }
105         }

```

Figure 4. 42 Admin logged Long Tour Hire Details Form Code - part 2

Figure 4.43 Admin logged Long Tour Hire Details Form Code - part 3

```
Program.cs    Give_Long_Hiring_Form.cs*  [Give_Long_Hiring_Form.cs[Design]]  - Rent_Car.Give_Long_Hiring_Form  + Obj.btn_Edit_Click(object sender, EventArgs e)
154     con.Close();
155   }
156 }
157
158 //*****Save*****
159 private void dgv_LongD_Hire_List_CellClick(object sender, DataGridViewCellEventArgs e)
160 {
161   con.Open();
162   int ID;
163
164   ID = int.Parse(dgv_LongD_Hire_List.Rows[e.RowIndex].Cells[e.ColumnIndex].Value.ToString());
165
166   SqlCommand cmd = con.CreateCommand();
167   cmd.CommandType = CommandType.Text;
168   cmd.CommandText = "SELECT * FROM Details_LongD_Hire_Table WHERE LongD_ID = '" + ID + "' ";
169
170   SqlDataReader DR1 = cmd.ExecuteReader();
171
172   if (DR1.Read())
173   {
174     txt_LongD_ID.Text = DR1.GetValue(0).ToString();
175     txt_type.Text = DR1.GetValue(1).ToString();
176     txt_brand.Text = DR1.GetValue(2).ToString();
177     txt_model.Text = DR1.GetValue(3).ToString();
178     txt_colour.Text = DR1.GetValue(4).ToString();
179     cmb_packages.Text = DR1.GetValue(5).ToString();
180     txt_Per_km_charge.Text = DR1.GetValue(6).ToString();
181     txt_Max_km_limit.Text = DR1.GetValue(7).ToString();
182     txt_Extra_km_charge.Text = DR1.GetValue(8).ToString();
183     txt_Driver_overnight_charge.Text = DR1.GetValue(9).ToString();
184     txt_Vehicle_parking_charge.Text = DR1.GetValue(10).ToString();
185   }
186
187   DR1.Close();
188   con.Close();
189 }
190
191 //*****Save*****"Edit*****"Delete*****
192
193 //*****Save*****
194 private void btn_Save_Click(object sender, EventArgs e)
195 {
196   if ((txt_type.Text == "") || txt_brand.Text == "" || txt_model.Text == "" || txt_colour.Text == "" || cmb_packages.Text == "" || txt_Per_km_charge.Text == "" || txt_Max_km_limit.Text == "" || txt_Extra_km_charge.Text == "" || txt_Driver_overnight_charge.Text == "" || txt_Vehicle_parking_charge.Text == "")
197   {
198     MessageBox.Show("Missing Information");
199   }
200   else
201   {
202     try
203     {
204       con.Open();
205       cmd = new SqlCommand("INSERT INTO Details_LongD_Hire_Table(Veh_Type,Veh_Brand,Veh_Model,Veh_Colour,LongD_Packages,LongD_Per_km_charge,LongD_Max_km_limit,LongD_Extra_km_charge," +
206         "Long_Driver_overnight_charge,Long_Vehicle_parking_charge) VALUES('" + txt_type.Text + "','" + txt_brand.Text + "','" +
207         "','" + txt_model.Text + "','" + txt_colour.Text + "','" + cmb_packages.Text + "','" + txt_Per_km_charge.Text + "','" +
208         ",'" + txt_Max_km_limit.Text + "','" + txt_Extra_km_charge.Text + "','" + txt_Driver_overnight_charge.Text + "','" +
209         ",'" + txt_Vehicle_parking_charge.Text + "')");
210       cmd.ExecuteNonQuery();
211     }
212     catch { }
213   }
214 }
```

Figure 4. 44 Admin logged Long Tour Hire Details Form Code - part 4

```
Give_Long_Hiring_Form.cs.cs [Design] +> RentiCar_Give_Long_Hiring_Form +> @_btn_Edit_Click(object sender, EventArgs e)
RentCar
285     cmd = new SqlCommand("INSERT INTO Details_Loan_Hire_Table SET Veh_Type = '" + txt_type.Text + "', Veh_Brand = '" + txt_brand.Text + "', Veh_Model = '" + txt_model.Text + "', Veh_Colour = '" + txt_colour.Text + "', LongD_Per_km_charge = '" + txt_Per_km_charge.Text + "', LongD_Max_km_limit = '" + txt_Max_km_limit.Text + "' , LongD_Extr_km_charge = '" + txt_Extr_km_charge.Text + "' , LongD_Driver_overnight_charge = '" + txt_Driver_overnight_charge.Text + "' , LongD_Vehicle_parking_charge = '" + txt_Vehicle_parking_charge.Text + "' ", con);
286     cmd.ExecuteNonQuery();
287     con.Close();
288     MessageBox.Show("Long Day Package added successfully!!!!");
289
290     display_data_grid_view_Vehicle_List();
291     display_data_grid_view_Long_Hire_List(); //data grid view method
292     clear(); //data clear method
293
294 }
295
296 catch (Exception ex)
297 {
298     MessageBox.Show(ex.Message);
299     con.Close();
300 }
301
302 }
303
304 private void btn_Edit_Click(object sender, EventArgs e)
305 {
306     if (txt_type.Text == "" || txt_brand.Text == "" || txt_model.Text == "" || txt_colour.Text == "" || cmb_packages.Text == "" || txt_Per_km_charge.Text == "" || txt_Max_km_limit.Text == ""
307     || txt_Extr_km_charge.Text == "" || txt_Driver_overnight_charge.Text == "" || txt_Vehicle_parking_charge.Text == "")
308     {
309         MessageBox.Show("Missing Information");
310     }
311     else
312     {
313         try
314         {
315             con.Open();
316             cmd = new SqlCommand("UPDATE Details_Loan_Hire_Table SET Veh_Type = '" + txt_type.Text + "', Veh_Brand = '" + txt_brand.Text + "', Veh_Model = '" + txt_model.Text + "', Veh_Colour = '" + txt_colour.Text + "', LongD_Per_km_charge = '" + txt_Per_km_charge.Text + "' , LongD_Max_km_limit = '" + txt_Max_km_limit.Text + "' , LongD_Extr_km_charge = '" + txt_Extr_km_charge.Text + "' , LongD_Driver_overnight_charge = '" + txt_Driver_overnight_charge.Text + "' , LongD_Vehicle_parking_charge = '" + txt_Vehicle_parking_charge.Text + "' WHERE LongD_ID = '" + txt_LongD_ID.Text + "' ", con);
317             cmd.ExecuteNonQuery();
318             con.Close();
319             MessageBox.Show("Long Day Package edit successfully!!!!");
320
321             display_data_grid_view_Vehicle_List();
322             display_data_grid_view_Long_Hire_List(); //data grid view method
323             clear(); //data clear method
324
325         }
326         catch (Exception ex)
327         {
328             MessageBox.Show(ex.Message);
329             con.Close();
330         }
331     }
332 }
333
334 }
```

Figure 4. 45 Admin logged Long Tour Hire Details Form Code - part 5

```
Program.cs # Give_Long_Hiring_Form.cs.cs [Design] + Rent_Car.Give_Long_Hiring_Form + ④ btn_Edit_Click(object sender, EventArgs e)

Rent_Car
255
256
257
258
259     ireference
260     private void btn_Delete_Click(object sender, EventArgs e)
261     {
262         if (cMB_packages.Text == "")
263         {
264             MessageBox.Show("Select Long Day Package to Delete");
265         }
266         else
267         {
268             try
269             {
270                 con.Open();
271                 cmd = new SqlCommand("DELETE FROM Details_LongD_Hire_Table WHERE LongD_ID = '" + txt_LongD_ID.Text + "' ", con);
272                 cmd.ExecuteNonQuery();
273                 con.Close();
274                 MessageBox.Show("Long Day Package delete successfully!!!\"");
275
276                 display_data_grid_view_Vehicle_list();
277                 display_data_grid_view_LongD_Hire_List(); //data grid view method
278                 clear(); //data clear method
279             }
280             catch (Exception ex)
281             {
282                 MessageBox.Show(ex.Message);
283                 con.Close();
284             }
285         }
286     }
287
288     ireference
289     private void btn_Clear_Click(object sender, EventArgs e)
290     {
291         txt_LongD_ID.Text = "";
292         txt_Type.Text = "";
293         txt_Brand.Text = "";
294         txt_Model.Text = "";
295         txt_Colour.Text = "";
296         txt_Mileage.Text = "";
297         txt_Per_km_charge.Text = "";
298         txt_Max_km_limit.Text = "";
299         txt_Extra_km_charge.Text = "";
300         txt_Driver_overnight_charge.Text = "";
301         txt_Vehicle_parking_charge.Text = "";
302     }
303 }
```

Figure 4. 46 Admin logged Long Tour Hire Details Form Code - part 6

4.1.10 Admin logged Driver Details Form Design and Code

The screenshot displays a web-based application interface for managing driver details. On the left, there is a sidebar with icons and labels: Ayubo Drive (car icon), Dashboard (monitor icon), Vehicles (car icon), Renting (rental car icon), 1 Day Hire (car icon), Tour Hire (taxi icon), Drivers (two people icon), and LOGOUT (red circle with a white arrow). The main content area has a green header titled "Driver Details". It contains input fields for "Driver ID" (1), "Username" (Ryan), "Password" (Ryan), "Email" (ryandilithusha@gmail.com), "Phone Number" (0764170647), and "BirthDay" (2/25/1997). Below these are four buttons: "Save", "Edit", "Delete", and "Clear". Underneath the header is a table titled "Drivers List" with columns: Driver_ID, Driver_Username, Driver_Password, Driver_Email, Driver_Phone, and Driver_DOB. The table contains three rows of data:

Driver_ID	Driver_Username	Driver_Password	Driver_Email	Driver_Phone	Driver_DOB
1	Ryan	Ryan	ryandilithusha@gmail.com	0764170647	1997-02-25
2	Michelle	Michelle	michellefernando@gmail.com	0764170495	1997-06-24
3	Pathum	Pathum	pathumfernando@gmail.com	0766123468	1997-03-03

Figure 4. 47 Admin logged Driver Details Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class Drivers_Register_Form : Form
15     {
16         // references
17         public Drivers_Register_Form()
18         {
19             InitializeComponent();
20             display_data_grid_view();
21         }
22
23         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
24         SqlCommand cmd;
25
26         //For data grid view
27         SqlDataAdapter adapt;
28         DataTable dt;
29
30         //***** Quick Menu BUTTONS *****
31
32         //reference
33         private void btn_dashboard_Click(object sender, EventArgs e)
34         {
35             Dashboard obj = new Dashboard();
36             obj.Show();
37             this.Hide();
38         }
39
40         //reference
41         private void btn_vehicles_Click(object sender, EventArgs e)
42         {
43             Vehicles_Register_Form obj = new Vehicles_Register_Form();
44             obj.Show();
45             this.Hide();
46         }
47
48         //reference
49         private void btn_renting_Click(object sender, EventArgs e)
50         {
51             Give_Renting_Form obj = new Give_Renting_Form();
52             obj.Show();
53         }
54     }
55 
```

Figure 4. 48 Admin logged Driver Details Form Code - part 1

```

56         //reference
57         private void btn_ID_Hire_Click(object sender, EventArgs e)
58         {
59             Give_ID_Hiring_Form obj = new Give_ID_Hiring_Form();
60             obj.Show();
61             this.Hide();
62         }
63
64         //reference
65         private void btn_Tour_Hire_Click(object sender, EventArgs e)
66         {
67             Give_Long_Hiring_Form obj = new Give_Long_Hiring_Form();
68             obj.Show();
69             this.Hide();
70         }
71
72         //reference
73         private void btn_logout_Click(object sender, EventArgs e)
74         {
75             Login_Form obj = new Login_Form();
76             obj.Show();
77             this.Hide();
78         }
79
80         //***** OTHER METHODS *****
81
82         //reference
83         public void clear()
84         {
85             txt_driverID.Text = "";
86             txt_username.Text = "";
87             txt_password.Text = "";
88             txt_email.Text = "";
89             txt_phone.Text = "";
90             dtp_DOB.Value = DateTime.Now;
91         }
92
93         //reference
94         public void display_data_grid_view() //For the data grid view
95         {
96             try
97             {
98                 dt = new DataTable();
99                 con.Open();
100                adapt = new SqlDataAdapter("SELECT * FROM Details_Drivers_Table", con);
101                adapt.Fill(dt);
102                dgv_drivers.DataSource = dt;
103                con.Close();
104            }
105            catch (Exception ex)
106            {
107                MessageBox.Show(ex.Message);
108            }
109        }
110 
```

Figure 4. 49 Admin logged Driver Details Form Code - part 2

```

Program.cs 0 X Drivers_Register_Form.cs + X Drivers_Register_Form.cs [Design]
Rent Car Drivers_Register_Form.cs [Design] - Rent_Car.Drivers_Register_Form
Drivers_Register_Form()
Drivers_Register_Form()

198     {
199         MessageBox.Show(ex.Message);
200         con.Close();
201     }
202 }
203 }
204 }
205 }

1 reference
private void dgv_drivers_CellClick(object sender, DataGridViewCellEventArgs e)
{
    con.Open();
    int ID;
    ID = int.Parse(dgv_drivers.Rows[e.RowIndex].Cells[0].Value.ToString());
    SqlCommand cmd = con.CreateCommand();
    cmd.CommandType = CommandType.Text;
    cmd.CommandText = "SELECT * FROM Details_Drivers_Table WHERE Driver_ID = '" + ID + "' ";
    SqlDataReader DR1 = cmd.ExecuteReader();
    if (DR1.Read())
    {
        txt_driverID.Text = DR1.GetValue(0).ToString();
        txt_username.Text = DR1.GetValue(1).ToString();
        txt_password.Text = DR1.GetValue(2).ToString();
        txt_email.Text = DR1.GetValue(3).ToString();
        txt_phone.Text = DR1.GetValue(4).ToString();
        dtb_dob.Value = DateTime.Parse(DR1.GetValue(5).ToString());
    }
    DR1.Close();
    con.Close();
}
}

//*****SAVE*****Edit*****Delete*****
1 reference
private void btn_Save_Click(object sender, EventArgs e)
{
    if (txt_username.Text == "" || txt_password.Text == "" || txt_email.Text == "" || txt_phone.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            string date = Convert.ToString(dtb_dob.Value);
            date = date.Remove(date.Length - 11);
            con.Open();
            cmd = new SqlCommand("INSERT INTO Details_Drivers_Table(Driver_Username,Driver_Password,Driver_Email,Driver_Phone,Driver_DOB) VALUES('" + txt_username.Text + "','" + txt_password.Text + "','" + txt_email.Text + "','" + txt_phone.Text + "','" + dtb_dob.Value.ToString("yyyy-MM-dd") + "')", con);
            cmd.ExecuteNonQuery();
            con.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}
}

1 reference
private void btn_Edit_Click(object sender, EventArgs e)
{
    if (txt_username.Text == "" || txt_password.Text == "" || txt_email.Text == "" || txt_phone.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("UPDATE Details_Drivers_Table SET Driver_Username = '" + txt_username.Text + "', Driver_Password = '" + txt_password.Text + "', Driver_Email = '" + txt_email.Text + "', Driver_Phone = '" + txt_phone.Text + "', Driver_DOB = '" + dtb_dob.Value.ToString("yyyy-MM-dd") + "' WHERE Driver_ID = '" + txt_driverID.Text + "'", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Driver edit successfully!!!");

            display_data_grid_view(); //data grid view method
            clear(); //data clear method
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}
}

1 reference
private void btn_Delete_Click(object sender, EventArgs e)
{
    if (txt_driverID.Text == "")
    {
        MessageBox.Show("Select driver to Delete");
    }
    else
    {
}
}
}

```

Figure 4. 50 Admin logged Driver Details Form Code - part 3

```

Program.cs 0 X Drivers_Register_Form.cs + X Drivers_Register_Form.cs [Design]
Rent Car Drivers_Register_Form.cs [Design] - Rent_Car.Drivers_Register_Form
Drivers_Register_Form()
Drivers_Register_Form()

152     con.Close();
153     MessageBox.Show("Driver added successfully!!!");

154     display_data_grid_view(); //data grid view method
155     clear(); //data clear method
156 }
157 }
158 }
159 }
160 }

161 catch (Exception ex)
162 {
163     MessageBox.Show(ex.Message);
164     con.Close();
165 }
166 }

1 reference
private void btn_Edit_Click(object sender, EventArgs e)
{
    if (txt_username.Text == "" || txt_password.Text == "" || txt_email.Text == "" || txt_phone.Text == "")
    {
        MessageBox.Show("Missing Information");
    }
    else
    {
        try
        {
            con.Open();
            cmd = new SqlCommand("UPDATE Details_Drivers_Table SET Driver_Username = '" + txt_username.Text + "', Driver_Password = '" + txt_password.Text + "', Driver_Email = '" + txt_email.Text + "', Driver_Phone = '" + txt_phone.Text + "', Driver_DOB = '" + dtb_dob.Value.ToString("yyyy-MM-dd") + "' WHERE Driver_ID = '" + txt_driverID.Text + "'", con);
            cmd.ExecuteNonQuery();
            con.Close();
            MessageBox.Show("Driver edit successfully!!!");

            display_data_grid_view(); //data grid view method
            clear(); //data clear method
        }
        catch (Exception ex)
        {
            MessageBox.Show(ex.Message);
        }
    }
}
}

1 reference
private void btn_Delete_Click(object sender, EventArgs e)
{
    if (txt_driverID.Text == "")
    {
        MessageBox.Show("Select driver to Delete");
    }
    else
    {
}
}
}

```

Figure 4. 51 Admin logged Driver Details Form Code - part 4

```
Program.cs  Drivers_Register_Form.cs  Drivers_Register_Form.cs [Design]
Rent Car  Rent_Car.Drivers_Register_Form  Drivers_Register_Form

282     }
283     else
284     {
285         try
286         {
287             con.open();
288             cmd = new SqlCommand("DELETE FROM Details_Drivers_Table WHERE Driver_ID = '" + txt_driverID.Text + "' ", con);
289             cmd.ExecuteNonQuery();
290             con.Close();
291             MessageBox.Show("Driver delete successfully!!!");
292             display_data_grid_view(); //data grid view method
293             clear(); //data clear method
294         }
295         catch (Exception ex)
296         {
297             MessageBox.Show(ex.Message);
298             con.Close();
299         }
300     }
301 }
302
303 1 reference
304 private void btn_Clear_Click(object sender, EventArgs e)
305 {
306     txt_driverID.Text = "";
307     txt_username.Text = "";
308     txt_password.Text = "";
309     txt_email.Text = "";
310     txt_phone.Text = "";
311     dtp_DOB.Value = DateTime.Now;
312 }
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336 }
```

Figure 4. 52 Admin logged Driver Details Form Code - part 5

4.1.11 Driver logged Rent Vehicle Form Design and Code

- Ayubo Drive
- Rent
- 1 Day Hire
- Tour Hire
- My Profile
- About Us

WELCOME Ryan

Renting Vehicles List

Kent_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	Rent_Daily	Rent_Weekly	Rent_Monthly	Rent_DriverRate
1	Car	Nissan	FB 15	Grey	2000	10000	40000	1000
2	SUV	Toyota	Compact	White	4000	20000	80000	2000
3	Van	Toyota	Lte Ace	Black	2500	12000	60000	1500
4	Bus	TATA	Leyland	Blue	5000	20000	100000	2000

Renting Package ID

Type	Brand	Model	Colour
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Renting Rates

Daily	Weekly	Monthly	Driver Rate
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

Rent Calculation

Renting Date <input type="date" value="1/10/2022"/>	Charge for Days <input type="text"/>
Returning Date <input type="date" value="1/10/2022"/>	Charge for Weeks <input type="text"/>
	Charge for Months <input type="text"/>
	Driver Charge <input type="text"/>

Need of the driver YES NO

Calculate

Current Bill

Bent Package ID	Day_Rent	Week_Rent	Month_Rent	Driver_Charge	Total
<input type="text"/>					

Bill Amount : Amount **Print**

Renting Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
1	Ryan	1/16/2022 12:00:00 AM	5000
2	Ryan	2/3/2022 12:00:00 AM	146000
3	Ryan	2/3/2022 12:00:00 AM	354000
4	Ryan	2/3/2022 12:00:00 AM	55000

Figure 4. 53 Driver logged Rent Vehicle Form Design

Ryan Wickramaratne (COL 00081762)

Unit_1:PRO - Programming

152

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Text;
7  using System.Threading.Tasks;
8  using System.Windows.Forms;
9  using System.Data.SqlClient;
10 
11 namespace Rent_Car
12 {
13     public partial class Rent_Form : Form
14     {
15         public Rent_Form()
16         {
17             InitializeComponent();
18             display_data_grid_view();
19             display_data_grid_view_BILL();
20             lbl_driver.Text = login_Form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
21         }
22 
23         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
24 
25         SqlDataAdapter adapt;
26         DataTable dt;
27 
28         //For data grid view
29         //***** Quick Menu BUTTONS *****
30         private void btn_ID_Hire_Click(object sender, EventArgs e)
31         {
32             _ID_Hire_Form obj = new _ID_Hire_Form();
33             obj.Show();
34             this.Hide();
35         }
36 
37         private void btn_Tour_Hire_Click(object sender, EventArgs e)
38         {
39             Long_Hire_Form obj = new Long_Hire_Form();
40             obj.Show();
41             this.Hide();
42         }
43 
44         private void btn_MyProfile_Click(object sender, EventArgs e)
45         {
46             ...
47         }
48 
49         private void btn_About_Us_Click(object sender, EventArgs e)
50         {
51             ...
52         }
53 
54         private void btn_Logout_Click(object sender, EventArgs e)
55         {
56             ...
57         }
58 
59         //***** OTHER METHODS *****
60         public void clear()
61         {
62             txt_type.Text = "";
63             txt_brand.Text = "";
64             txt_model.Text = "";
65             txt_color.Text = "";
66             txt_daily_rent.Text = "";
67             txt_weekly_rent.Text = "";
68             txt_monthly_rent.Text = "";
69             cmb_refresh_category.Text = "";
70 
71             dtp_start.Value = DateTime.Now;
72             dtp_endValue = DateTime.Now;
73             txt_calculate.Text = "";
74             rbth_yes.Checked = false;
75             rbth_no.Checked = false;
76         }
77 
78         public void display_data_grid_view() //For the data grid view
79         {
80             try
81             {
82                 dt = new DataTable();
83                 con.open();
84                 adapt = new SqlDataAdapter("SELECT * FROM Details_Renting_Table", con);
85                 adapt.Fill(dt);
86                 dgw_Renting_List.datasource = dt;
87             }
88         }
89 
90         public void display_data_grid_view_BILL()
91         {
92             ...
93         }
94 
95         public void display_data_grid_view()
96         {
97             ...
98         }
99 
100        public void display_data_grid_view()
101        {
102            ...
103        }
104    }
105 }

```

Figure 4. 54 Driver logged Rent Vehicle Form Code - part 1

```

51         ...
52         ...
53         ...
54         ...
55         ...
56         ...
57         ...
58         ...
59         ...
60         ...
61         ...
62         ...
63         ...
64         ...
65         ...
66         ...
67         ...
68         ...
69         ...
70         ...
71         ...
72         ...
73         ...
74         ...
75         ...
76         ...
77         ...
78         ...
79         ...
80         ...
81         ...
82         ...
83         ...
84         ...
85         ...
86         ...
87         ...
88         ...
89         ...
90         ...
91         ...
92         ...
93         ...
94         ...
95         ...
96         ...
97         ...
98         ...
99         ...
100        ...
101        ...
102        ...
103        ...
104        ...
105        ...
106        ...
107        ...
108        ...
109        ...
110        ...
111        ...
112        ...
113        ...
114        ...
115        ...
116        ...
117        ...
118        ...
119        ...
120        ...
121        ...
122        ...
123        ...
124        ...
125        ...
126        ...
127        ...
128        ...
129        ...
130        ...
131        ...
132        ...
133        ...
134        ...
135        ...
136        ...
137        ...
138        ...
139        ...
140        ...
141        ...
142        ...
143        ...
144        ...
145        ...
146        ...
147        ...
148        ...
149        ...
150        ...
151        ...
152        ...
153        ...
154        ...
155        ...
156        ...
157        ...
158        ...
159        ...
160        ...
161        ...
162        ...
163        ...
164        ...
165        ...
166        ...
167        ...
168        ...
169        ...
170        ...
171        ...
172        ...
173        ...
174        ...
175        ...
176        ...
177        ...
178        ...
179        ...
180        ...
181        ...
182        ...
183        ...
184        ...
185        ...
186        ...
187        ...
188        ...
189        ...
190        ...
191        ...
192        ...
193        ...
194        ...
195        ...
196        ...
197        ...
198        ...
199        ...
200        ...
201        ...
202        ...
203        ...
204        ...
205        ...
206        ...
207        ...
208        ...
209        ...
210        ...
211        ...
212        ...
213        ...
214        ...
215        ...
216        ...
217        ...
218        ...
219        ...
220        ...
221        ...
222        ...
223        ...
224        ...
225        ...
226        ...
227        ...
228        ...
229        ...
230        ...
231        ...
232        ...
233        ...
234        ...
235        ...
236        ...
237        ...
238        ...
239        ...
240        ...
241        ...
242        ...
243        ...
244        ...
245        ...
246        ...
247        ...
248        ...
249        ...
250        ...
251        ...
252        ...
253        ...
254        ...
255        ...
256        ...
257        ...
258        ...
259        ...
260        ...
261        ...
262        ...
263        ...
264        ...
265        ...
266        ...
267        ...
268        ...
269        ...
270        ...
271        ...
272        ...
273        ...
274        ...
275        ...
276        ...
277        ...
278        ...
279        ...
280        ...
281        ...
282        ...
283        ...
284        ...
285        ...
286        ...
287        ...
288        ...
289        ...
290        ...
291        ...
292        ...
293        ...
294        ...
295        ...
296        ...
297        ...
298        ...
299        ...
300        ...
301        ...
302        ...
303        ...
304        ...
305        ...
306        ...
307        ...
308        ...
309        ...
310        ...
311        ...
312        ...
313        ...
314        ...
315        ...
316        ...
317        ...
318        ...
319        ...
320        ...
321        ...
322        ...
323        ...
324        ...
325        ...
326        ...
327        ...
328        ...
329        ...
330        ...
331        ...
332        ...
333        ...
334        ...
335        ...
336        ...
337        ...
338        ...
339        ...
340        ...
341        ...
342        ...
343        ...
344        ...
345        ...
346        ...
347        ...
348        ...
349        ...
350        ...
351        ...
352        ...
353        ...
354        ...
355        ...
356        ...
357        ...
358        ...
359        ...
360        ...
361        ...
362        ...
363        ...
364        ...
365        ...
366        ...
367        ...
368        ...
369        ...
370        ...
371        ...
372        ...
373        ...
374        ...
375        ...
376        ...
377        ...
378        ...
379        ...
380        ...
381        ...
382        ...
383        ...
384        ...
385        ...
386        ...
387        ...
388        ...
389        ...
390        ...
391        ...
392        ...
393        ...
394        ...
395        ...
396        ...
397        ...
398        ...
399        ...
400        ...
401        ...
402        ...
403        ...
404        ...
405        ...
406        ...
407        ...
408        ...
409        ...
410        ...
411        ...
412        ...
413        ...
414        ...
415        ...
416        ...
417        ...
418        ...
419        ...
420        ...
421        ...
422        ...
423        ...
424        ...
425        ...
426        ...
427        ...
428        ...
429        ...
430        ...
431        ...
432        ...
433        ...
434        ...
435        ...
436        ...
437        ...
438        ...
439        ...
440        ...
441        ...
442        ...
443        ...
444        ...
445        ...
446        ...
447        ...
448        ...
449        ...
450        ...
451        ...
452        ...
453        ...
454        ...
455        ...
456        ...
457        ...
458        ...
459        ...
460        ...
461        ...
462        ...
463        ...
464        ...
465        ...
466        ...
467        ...
468        ...
469        ...
470        ...
471        ...
472        ...
473        ...
474        ...
475        ...
476        ...
477        ...
478        ...
479        ...
480        ...
481        ...
482        ...
483        ...
484        ...
485        ...
486        ...
487        ...
488        ...
489        ...
490        ...
491        ...
492        ...
493        ...
494        ...
495        ...
496        ...
497        ...
498        ...
499        ...
500        ...
501        ...
502        ...
503        ...
504        ...
505        ...
506        ...
507        ...
508        ...
509        ...
510        ...
511        ...
512        ...
513        ...
514        ...
515        ...
516        ...
517        ...
518        ...
519        ...
520        ...
521        ...
522        ...
523        ...
524        ...
525        ...
526        ...
527        ...
528        ...
529        ...
530        ...
531        ...
532        ...
533        ...
534        ...
535        ...
536        ...
537        ...
538        ...
539        ...
540        ...
541        ...
542        ...
543        ...
544        ...
545        ...
546        ...
547        ...
548        ...
549        ...
550        ...
551        ...
552        ...
553        ...
554        ...
555        ...
556        ...
557        ...
558        ...
559        ...
560        ...
561        ...
562        ...
563        ...
564        ...
565        ...
566        ...
567        ...
568        ...
569        ...
570        ...
571        ...
572        ...
573        ...
574        ...
575        ...
576        ...
577        ...
578        ...
579        ...
580        ...
581        ...
582        ...
583        ...
584        ...
585        ...
586        ...
587        ...
588        ...
589        ...
590        ...
591        ...
592        ...
593        ...
594        ...
595        ...
596        ...
597        ...
598        ...
599        ...
600        ...
601        ...
602        ...
603        ...
604        ...
605        ...
606        ...
607        ...
608        ...
609        ...
610        ...
611        ...
612        ...
613        ...
614        ...
615        ...
616        ...
617        ...
618        ...
619        ...
620        ...
621        ...
622        ...
623        ...
624        ...
625        ...
626        ...
627        ...
628        ...
629        ...
630        ...
631        ...
632        ...
633        ...
634        ...
635        ...
636        ...
637        ...
638        ...
639        ...
640        ...
641        ...
642        ...
643        ...
644        ...
645        ...
646        ...
647        ...
648        ...
649        ...
650        ...
651        ...
652        ...
653        ...
654        ...
655        ...
656        ...
657        ...
658        ...
659        ...
660        ...
661        ...
662        ...
663        ...
664        ...
665        ...
666        ...
667        ...
668        ...
669        ...
670        ...
671        ...
672        ...
673        ...
674        ...
675        ...
676        ...
677        ...
678        ...
679        ...
680        ...
681        ...
682        ...
683        ...
684        ...
685        ...
686        ...
687        ...
688        ...
689        ...
690        ...
691        ...
692        ...
693        ...
694        ...
695        ...
696        ...
697        ...
698        ...
699        ...
700        ...
701        ...
702        ...
703        ...
704        ...
705        ...
706        ...
707        ...
708        ...
709        ...
710        ...
711        ...
712        ...
713        ...
714        ...
715        ...
716        ...
717        ...
718        ...
719        ...
720        ...
721        ...
722        ...
723        ...
724        ...
725        ...
726        ...
727        ...
728        ...
729        ...
730        ...
731        ...
732        ...
733        ...
734        ...
735        ...
736        ...
737        ...
738        ...
739        ...
740        ...
741        ...
742        ...
743        ...
744        ...
745        ...
746        ...
747        ...
748        ...
749        ...
750        ...
751        ...
752        ...
753        ...
754        ...
755        ...
756        ...
757        ...
758        ...
759        ...
760        ...
761        ...
762        ...
763        ...
764        ...
765        ...
766        ...
767        ...
768        ...
769        ...
770        ...
771        ...
772        ...
773        ...
774        ...
775        ...
776        ...
777        ...
778        ...
779        ...
780        ...
781        ...
782        ...
783        ...
784        ...
785        ...
786        ...
787        ...
788        ...
789        ...
790        ...
791        ...
792        ...
793        ...
794        ...
795        ...
796        ...
797        ...
798        ...
799        ...
800        ...
801        ...
802        ...
803        ...
804        ...
805        ...
806        ...
807        ...
808        ...
809        ...
810        ...
811        ...
812        ...
813        ...
814        ...
815        ...
816        ...
817        ...
818        ...
819        ...
820        ...
821        ...
822        ...
823        ...
824        ...
825        ...
826        ...
827        ...
828        ...
829        ...
830        ...
831        ...
832        ...
833        ...
834        ...
835        ...
836        ...
837        ...
838        ...
839        ...
840        ...
841        ...
842        ...
843        ...
844        ...
845        ...
846        ...
847        ...
848        ...
849        ...
850        ...
851        ...
852        ...
853        ...
854        ...
855        ...
856        ...
857        ...
858        ...
859        ...
860        ...
861        ...
862        ...
863        ...
864        ...
865        ...
866        ...
867        ...
868        ...
869        ...
870        ...
871        ...
872        ...
873        ...
874        ...
875        ...
876        ...
877        ...
878        ...
879        ...
880        ...
881        ...
882        ...
883        ...
884        ...
885        ...
886        ...
887        ...
888        ...
889        ...
890        ...
891        ...
892        ...
893        ...
894        ...
895        ...
896        ...
897        ...
898        ...
899        ...
900        ...
901        ...
902        ...
903        ...
904        ...
905        ...
906        ...
907        ...
908        ...
909        ...
910        ...
911        ...
912        ...
913        ...
914        ...
915        ...
916        ...
917        ...
918        ...
919        ...
920        ...
921        ...
922        ...
923        ...
924        ...
925        ...
926        ...
927        ...
928        ...
929        ...
930        ...
931        ...
932        ...
933        ...
934        ...
935        ...
936        ...
937        ...
938        ...
939        ...
940        ...
941        ...
942        ...
943        ...
944        ...
945        ...
946        ...
947        ...
948        ...
949        ...
950        ...
951        ...
952        ...
953        ...
954        ...
955        ...
956        ...
957        ...
958        ...
959        ...
960        ...
961        ...
962        ...
963        ...
964        ...
965        ...
966        ...
967        ...
968        ...
969        ...
970        ...
971        ...
972        ...
973        ...
974        ...
975        ...
976        ...
977        ...
978        ...
979        ...
980        ...
981        ...
982        ...
983        ...
984        ...
985        ...
986        ...
987        ...
988        ...
989        ...
990        ...
991        ...
992        ...
993        ...
994        ...
995        ...
996        ...
997        ...
998        ...
999        ...
1000        ...
1001        ...
1002        ...
1003        ...
1004        ...
1005        ...
1006        ...
1007        ...
1008        ...
1009        ...
1010        ...
1011        ...
1012        ...
1013        ...
1014        ...
1015        ...
1016        ...
1017        ...
1018        ...
1019        ...
1020        ...
1021        ...
1022        ...
1023        ...
1024        ...
1025        ...
1026        ...
1027        ...
1028        ...
1029        ...
1030        ...
1031        ...
1032        ...
1033        ...
1034        ...
1035        ...
1036        ...
1037        ...
1038        ...
1039        ...
1040        ...
1041        ...
1042        ...
1043        ...
1044        ...
1045        ...
1046        ...
1047        ...
1048        ...
1049        ...
1050        ...
1051        ...
1052        ...
1053        ...
1054        ...
1055        ...
1056        ...
1057        ...
1058        ...
1059        ...
1060        ...
1061        ...
1062        ...
1063        ...
1064        ...
1065        ...
1066        ...
1067        ...
1068        ...
1069        ...
1070        ...
1071        ...
1072        ...
1073        ...
1074        ...
1075        ...
1076        ...
1077        ...
1078        ...
1079        ...
1080        ...
1081        ...
1082        ...
1083        ...
1084        ...
1085        ...
1086        ...
1087        ...
1088        ...
1089        ...
1090        ...
1091        ...
1092        ...
1093        ...
1094        ...
1095        ...
1096        ...
1097        ...
1098        ...
1099        ...
1100        ...
1101        ...
1102        ...
1103        ...
1104        ...
1105        ...
1106        ...
1107        ...
1108        ...
1109        ...
1110        ...
1111        ...
1112        ...
1113        ...
1114        ...
1115        ...
1116        ...
1117        ...
1118        ...
1119        ...
1120        ...
1121        ...
1122        ...
1123        ...
1124        ...
1125        ...
1126        ...
1127        ...
1128        ...
1129        ...
1130        ...
1131        ...
1132        ...
1133        ...
1134        ...
1135        ...
1136        ...
1137        ...
1138        ...
1139        ...
1140        ...
1141        ...
1142        ...
1143        ...
1144        ...
1145        ...
1146        ...
1147        ...
1148        ...
1149        ...
1150        ...
1151        ...
1152        ...
1153        ...
1154        ...
1155        ...
1156        ...
1157        ...
1158        ...
1159        ...
1160        ...
1161        ...
1162        ...
1163        ...
1164        ...
1165        ...
1166        ...
1167        ...
1168        ...
1169        ...
1170        ...
1171        ...
1172        ...
1173        ...
1174        ...
1175        ...
1176        ...
1177        ...
1178        ...
1179        ...
1180        ...
1181        ...
1182        ...
1183        ...
1184        ...
1185        ...
1186        ...
1187        ...
1188        ...
1189        ...
1190        ...
1191        ...
1192        ...
1193        ...
1194        ...
1195        ...
1196        ...
1197        ...
1198        ...
1199        ...
1200        ...
1201        ...
1202        ...
1203        ...
1204        ...
1205        ...
1206        ...
1207        ...
1208        ...
1209        ...
1210        ...
1211        ...
1212        ...
1213        ...
1214        ...
1215        ...
1216        ...
1217        ...
1218        ...
1219        ...
1220        ...
1221        ...
1222        ...
1223        ...
1224        ...
1225        ...
1226        ...
1227        ...
1228        ...
1229        ...
1230        ...
1231        ...
1232        ...
1233        ...
1234        ...
1235        ...
1236        ...
1237        ...
1238        ...
1239        ...
1240        ...
1241        ...
1242        ...
1243        ...
1244        ...
1245        ...
1246        ...
1247        ...
1248        ...
1249        ...
1250        ...
1251        ...
1252        ...
1253        ...
1254        ...
1255        ...
1256        ...
1257        ...
1258        ...
1259        ...
1260        ...
1261        ...
1262        ...
1263        ...
1264        ...
1265        ...
1266        ...
1267        ...
1268        ...
1269        ...
1270        ...
1271        ...
1272        ...
1273        ...
1274        ...
1275        ...
1276        ...
1277        ...
1278        ...
1279        ...
1280        ...
1281        ...
1282        ...
1283        ...
1284        ...
1285        ...
1286        ...
1287        ...
1288        ...
1289        ...
1290        ...
1291        ...
1292        ...
1293        ...
1294        ...
1295        ...
1296        ...
1297        ...
1298        ...
1299        ...
1300        ...
1301        ...
1302        ...
1303        ...
1304        ...
1305        ...
1306        ...
1307        ...
1308        ...
1309        ...
1310        ...
1311        ...
1312        ...
1313        ...
1314        ...
1315        ...
1316        ...
1317        ...
1318        ...
1319        ...
1320        ...
1321        ...
1322        ...
1323        ...
1324        ...
1325        ...
1326        ...
1327        ...
1328        ...
1329        ...
1330        ...
1331        ...
1332        ...
1333        ...
1334        ...
1335        ...
1336        ...
1337        ...
1338        ...
1339        ...
1340        ...
1341        ...
1342        ...
1343        ...
1344        ...
1345        ...
1346        ...
1347        ...
1348        ...
1349        ...
1350        ...
1351        ...
1352        ...
1353        ...
1354        ...
1355        ...
1356        ...
1357        ...
1358        ...
1359        ...
1360        ...
1361        ...
1362        ...
1363        ...
1364        ...
1365        ...
1366        ...
1367        ...
1368        ...
1369        ...
1370        ...
1371        ...
1372        ...
1373        ...
1374        ...
1375        ...
1376        ...
1377        ...
1378        ...
1379        ...
1380        ...
1381        ...
1382        ...
1383        ...
1384        ...
1385        ...
1386        ...
1387        ...
1388        ...
1389        ...
1390        ...
1391        ...
1392        ...
1393        ...
1394        ...
1395        ...
1396        ...
1397        ...
1398        ...
1399        ...
1400        ...
1401        ...
1402        ...
1403        ...
1404        ...
1405        ...
1406        ...
1407        ...
1408        ...
1409        ...
1410        ...
1411        ...
1412        ...
1413        ...
1414        ...
1415        ...
1416        ...
1417        ...
1418        ...
1419        ...
1420        ...
1421        ...
1422        ...
1423        ...
1424        ...
1425        ...
1426        ...
1427        ...
1428        ...
1429        ...
1430        ...
1431        ...
1432        ...
1433        ...
1434        ...
1435        ...
1436        ...
1437        ...
1438        ...
1439        ...
1440        ...
1441        ...
1442        ...
1443        ...
1444        ...
1445        ...
1446        ...
1447        ...
1448        ...
1449        ...
1450        ...
1451        ...
1452        ...
1453        ...
1454        ...
1455        ...
1456        ...
1457        ...
1458        ...
1459        ...
1460        ...
1461        ...
1462        ...
1463        ...
1464        ...
1465        ...
1466        ...
1467        ...
1468        ...
1469        ...
1470        ...
1471        ...
1472        ...
1473        ...
1474        ...
1475        ...
1476        ...
1477        ...
1478        ...
1479        ...
1480        ...
1481        ...
1482        ...
1483        ...
1484        ...
1485        ...
1486        ...
1487        ...
1488        ...
1489        ...
1490        ...
1491        ...
1492        ...
1493        ...
1494        ...
1495        ...
1496        ...
1497        ...
1498        ...
1499        ...
1500        ...
1501        ...
1502        ...
1503        ...
1504        ...
1505        ...
1506        ...
1507        ...
1508        ...
1509        ...
1510        ...
1511        ...
1512        ...
1513        ...
1514        ...
1515        ...
1516        ...
1517        ...
1518        ...
1519        ...
1520        ...
1521        ...
1522        ...
1523        ...
1524        ...
1525        ...
1526        ...
1527        ...
1528        ...
1529        ...
1530        ...
1531        ...
1532        ...
1533        ...
1534        ...
1535        ...
1536        ...
1537        ...
1538        ...
1539        ...
1540        ...
1541        ...
1542        ...
1543        ...
1544        ...
1545        ...
1546        ...
1547        ...
1548        ...
1549        ...
1550        ...
1551        ...
1552        ...
1553        ...
1554        ...
1555        ...
1556        ...
1557        ...
1558        ...
1559        ...
1560        ...
1561        ...
1562        ...
1563        ...
1564        ...
1565        ...
1566        ...
1567        ...
1568        ...
1569        ...
1570        ...
1571        ...
1572        ...
1573        ...
1574        ...
1575        ...
1576        ...
1577        ...
1578        ...
1579        ...
1580        ...
1581        ...
1582        ...
1583        ...
1584        ...
1585        ...
1586        ...
1587        ...
1588        ...
1589        ...
1590        ...
1591        ...
1592        ...
1593        ...
1594        ...
1595        ...
1596        ...
1597        ...
1598        ...
1599        ...
1600        ...
1601        ...
1602        ...
1603        ...
1604        ...
1605        ...
1606        ...
1607        ...
1608        ...
1609        ...
1610        ...
1611        ...
1612        ...
1613        ...
1614        ...
1615        ...
1616        ...
1617        ...
1618        ...
1619        ...
1620        ...
1621        ...
1622        ...
1623        ...
1624        ...
1625        ...
1626        ...
1627        ...
1628        ...
1629        ...
1630        ...
1631        ...
1632        ...
1633        ...
1634        ...
1635        ...
1636        ...
1637        ...
1638        ...
1639        ...
1640        ...
1641        ...
1642        ...
1643        ...
1644        ...
1645        ...
1646        ...
1647        ...
1648        ...
1649        ...
1650        ...
1651        ...
1652        ...
1653        ...
1654        ...
1655        ...
1656        ...
1657        ...
1658        ...
1659        ...
1660        ...
1661        ...
1662        ...
1663        ...
1664        ...
1665        ...
1666        ...
1667        ...
1668        ...
1669        ...
1670        ...
1671        ...
1672        ...
1673        ...
1674        ...
1675        ...
1676        ...
1677        ...
1678        ...
1679        ...
1680        ...
1681        ...
1682        ...
1683        ...
1684        ...
1685        ...
1686        ...
1687        ...
1688        ...
1689        ...
1690        ...
1691        ...
1692        ...
1693        ...
1694        ...
1695        ...
1696        ...
1697        ...
1698        ...
1699        ...
1700        ...
1701        ...
1702        ...
1703        ...
1704        ...
1705        ...
1706        ...
1707        ...
1708        ...
1709        ...
1710        ...
1711        ...
1712        ...
1713        ...
1714        ...
1715        ...
1716        ...
1717        ...
1718        ...
1719        ...
1720        ...
1721        ...
1722        ...
1723        ...
1724        ...
1725        ...
1726        ...
1727        ...
1728        ...
1729        ...
1730        ...
1731        ...
1732        ...
1733        ...
1734        ...
1735        ...
1736        ...
1737        ...
1738        ...
1739        ...
1740        ...
1741        ...
1742        ...
1743        ...
1744        ...
1745        ...
1746        ...
1747        ...
1748        ...
1749        ...
1750        ...
1751        ...
1752        ...
1753        ...
1754        ...
1755        ...
1756        ...
1757        ...
1758        ...
1759        ...
1760        ...
1761        ...
1762        ...
1763        ...
1764        ...
1765        ...
1766        ...
1767        ...
1768        ...
1769        ...
1770        ...
1771        ...
1772        ...
1773        ...
1774        ...
1775        ...
1776        ...
1777        ...
1778        ...
1779        ...
1780        ...
1781        ...
1782        ...
1783        ...
1784        ...
1785        ...
1786        ...
1787        ...
1788        ...
1789        ...
1790        ...
1791        ...
1792        ...
1793        ...
1794        ...
1795        ...
1796        ...
1797        ...
1798        ...
1799        ...
1800        ...
1801        ...
1802        ...
1803        ...
1804        ...
1805        ...
1806        ...
1807        ...
1808        ...
1809        ...
1810        ...
1811        ...
1812        ...
1813        ...
1814        ...
1815        ...
1816        ...
1817        ...
1818        ...
1819        ...
1820        ...
1821        ...
1822        ...
1823        ...
1824        ...
1825        ...
1826        ...
1827        ...
1828        ...
1829        ...
1830        ...
1831        ...
1832        ...
1833        ...
1834        ...
1835        ...
1836        ...
1837        ...
1838        ...
1839        ...
1840        ...
1841        ...
1842        ...
1843        ...
1844        ...
1845        ...
1846        ...
1847        ...
1848        ...
1849        ...
1850        ...
1851        ...
1852        ...
1853        ...
1854        ...
1855        ...
1856        ...
1857        ...
1858        ...
1859        ...
1860        ...
1861        ...
1862        ...
1863        ...
1864        ...
1865        ...
1866        ...
1867        ...
1868        ...
1869        ...
1870        ...
1871        ...
1872        ...
1873        ...
1874        ...
1875        ...
1876        ...
1877        ...
1878        ...
1879        ...
1880        ...
1881        ...
1882        ...
1883        ...
1884        ...
1885        ...
1886        ...
1887        ...
1888        ...
1889        ...
1890        ...
1891        ...
1892        ...
1893        ...
1894        ...
1895        ...
1896        ...
1897        ...
1898        ...
1899        ...
1900        ...
1901        ...
1902        ...
1903        ...
1904        ...

```

Program.cs 9 X Rent_Form.cs 9 X Rent_Form.cs [Design]

```

102     dgv_Renting_List.DataSource = dt;
103     con.Close();
104   }
105   catch (Exception ex)
106   {
107     MessageBox.Show(ex.Message);
108   }
109 }
110 }

111 int day_rent;
112 int week_rent;
113 int month_rent;
114 int driver_charge;
115
116 //reference
117 private void dgv_Renting_List_CellClick(object sender, DataGridViewCellEventArgs e)
118 {
119   {
120     con.open();
121     int ID;
122
123     ID = int.Parse(dgv_Renting_List.Rows[e.RowIndex].Cells[0].Value.ToString());
124
125     SqlCommand cmd = con.CreateCommand();
126     cmd.CommandType = CommandType.Text;
127     cmd.CommandText = "SELECT * FROM Details_Renting_Table WHERE Rent_ID = '" + ID + "' ";
128
129     SqlDataReader DR1 = cmd.ExecuteReader();
130
131   if (DR1.Read())
132   {
133     txt_rentID.Text = DR1.GetValue(0).ToString();
134     txt_type.Text = DR1.GetValue(1).ToString();
135     txt_brand.Text = DR1.GetValue(2).ToString();
136     txt_color.Text = DR1.GetValue(3).ToString();
137     txt_colour.Text = DR1.GetValue(4).ToString();
138     day_rent = Convert.ToInt32(txt_daily_rent.Text = DR1.GetValue(5).ToString()); //should Covert to int, becc we calculate them when adding the bill
139     week_rent = Convert.ToInt32(txt_weekly_rent.Text = DR1.GetValue(6).ToString()); //should Covert to int, becc we calculate them when adding the bill
140     month_rent = Convert.ToInt32(txt_monthly_rent.Text = DR1.GetValue(7).ToString()); //should Covert to int, becc we calculate them when adding the bill
141     driver_charge = Convert.ToInt32(txt_driver_rate.Text = DR1.GetValue(8).ToString()); //should Covert to int, becc we calculate them when adding the bill
142
143     DR1.Close();
144     con.Close();
145   }
146
147
148 //***** CALCULATION PART *****
149
150 //CALCULATION PART
151
152 //reference
153 private void btn_Calculate_Click(object sender, EventArgs e)
154 {

```

Figure 4. 56 Driver logged Rent Vehicle Form Code - part 3

Program.cs 9 X Rent_Form.cs 9 X Rent_Form.cs [Design]

```

155     Boolean driverRE5_checked = rbtn_re5.Checked;
156     Boolean driverNO_Checked = rbtn_no.Checked;
157
158     DateTime date1 = dtp_start.Value.Date;
159     DateTime date2 = dtp_end.Value.Date;
160
161     TimeSpan date_dif = date2 - date1;
162
163     int days = date_dif.Days; //Convert date differences into integer
164
165
166     int cal = 0;
167     if (driverRE5_checked)
168     {
169       if (Convert.ToInt32(days) <= 7)
170       {
171         int Days_rent = Convert.ToInt32(txt_daily_rent.Text) * Convert.ToInt32(days);
172         int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
173
174         cal = driver_charge + Days_rent;
175
176         //display data
177         txt_calculate.Text = Convert.ToString(cal);
178         txt_charge_Days.Text = Convert.ToString(Days_rent);
179         txt_charge_Charge.Text = Convert.ToString(driver_charge);
180         txt_charge_Months.Text = Convert.ToString("0");
181         txt_charge_Driver.Text = Convert.ToString(driver_charge);
182
183       }
184       else if (Convert.ToInt32(days) > 30)
185       {
186         int weeks = (int)Math.Floor(Convert.ToInt32(days) / 7.0); //to get weeks --> ex: 17 //7 = 2
187         int leftdays = Convert.ToInt32(days) % 7; //to get days after weeks --> ex : 17 % 7 = 3
188
189         int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
190         int Days_rent = leftdays * Convert.ToInt32(txt_daily_rent.Text);
191         int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
192
193         cal = driver_charge + Weeks_rent + Days_rent;
194
195         //display data
196         txt_calculate.Text = Convert.ToString(cal);
197         txt_charge_Days.Text = Convert.ToString(Days_rent);
198         txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
199         txt_charge_Months.Text = Convert.ToString("0");
200         txt_charge_Driver.Text = Convert.ToString(driver_charge);
201
202       }
203     }
204     else if (Convert.ToInt32(days) >= 30)
205     {
206       int Months = (int)Math.Floor(Convert.ToInt32(days) / 30.0); //to get months --> ex: 59//30 = 1
207       int leftDaysMonth = Convert.ToInt32(days) % 30; //to get days after months --> ex 59%30 = 29
208
209       int weeks = (int)Math.Floor(leftDaysMonth / 7.0); //to get number of weeks --> ex 29//7 = 4
210
211       int Weeks_rent = Weeks * Convert.ToInt32(txt_weekly_rent.Text);
212       int Days_rent = leftDaysMonth * Convert.ToInt32(txt_daily_rent.Text);
213       int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);
214
215       cal = driver_charge + Weeks_rent + Days_rent;
216
217       //display data
218       txt_calculate.Text = Convert.ToString(cal);
219       txt_charge_Days.Text = Convert.ToString(Days_rent);
220       txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
221       txt_charge_Months.Text = Convert.ToString("0");
222       txt_charge_Driver.Text = Convert.ToString(driver_charge);
223
224     }
225   }
226 }
227
228
229 
```

Figure 4. 57 Driver logged Rent Vehicle Form Code - part 4

```
Program.cs # Rent_Form.cs x Rent_Form.cs [Design] - Rent_Car.Rent_Form - Rent_Form()
```

```
int Weeks = (int)Math.Floor(leftDaysMonth / 7.0); //to get number of weeks --> ex 29/7 = 4
int leftdays = leftDaysMonth % 7; //to get days after weeks --> ex 29%7 = 1

int Months_rent = months * Convert.ToInt32(txt_monthly_rent.Text);
int Weeks_rent = weeks * Convert.ToInt32(txt_weekly_rent.Text);
int Days_rent = leftdays * Convert.ToInt32(txt_daily_rent.Text);
int driver_charge = Convert.ToInt32(txt_driver_rate.Text) * Convert.ToInt32(days);

cal = driver_charge + Months_rent + Weeks_rent + Days_rent;

//display data
txt_calculate.Text = Convert.ToString(cal);
txt_charge_Days.Text = Convert.ToString(Days_rent);
txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
txt_charge_Months.Text = Convert.ToString(Months_rent);
txt_charge_Driver.Text = Convert.ToString(driver_charge);

}

else if (driverNO_checked)
{
if (Convert.ToInt32(days) <= 7)
{
int Days_rent = Convert.ToInt32(txt_daily_rent.Text) * Convert.ToInt32(days);

cal = Days_rent;

//display data
txt_calculate.Text = Convert.ToString(cal);
txt_charge_Days.Text = Convert.ToString(Days_rent);
txt_charge_Weeks.Text = Convert.ToString("0");
txt_charge_Months.Text = Convert.ToString("0");
txt_charge_Driver.Text = Convert.ToString("0");

}
else if (Convert.ToInt32(days) < 30)
{
int Weeks = (int)Math.Floor(Convert.ToInt32(days) / 7.0); //to get weeks --> ex: 17 //7 = 2
int leftdays = Convert.ToInt32(days) % 7; //to get days after weeks --> ex : 17 % 7 = 3

int Days_rent = weeks * Convert.ToInt32(txt_weekly_rent.Text);
int Days_rent = leftdays * Convert.ToInt32(txt_daily_rent.Text);

cal = Weeks_rent + Days_rent;

//display data
txt_calculate.Text = Convert.ToString(cal);
txt_charge_Days.Text = Convert.ToString(Days_rent);
txt_charge_Weeks.Text = Convert.ToString(Weeks_rent);
txt_charge_Months.Text = Convert.ToString("0");
txt_charge_Driver.Text = Convert.ToString("0");
}
```

Figure 4. 58 Driver logged Rent Vehicle Form Code - part 5

Figure 4. 59 Driver logged Rent Vehicle Form Code - part 6

Program.cs 9 Rent_Form.cs -> Rent_Form.cs [Design] Rent_Car.Rent_Form

```

316 newRow.Cells[5].Value = txt_calculate.Text;
317 dgv_Current_Bill.Rows.Add(newRow);
318 n++; //ID Column value increment 1 by 1
319
320 finalTotal = finalTotal + Convert.ToInt32(txt_calculate.Text);
321
322 lbl_Amount.Text = "Rs " + finalTotal;
323
324 }
325
326 }
327
328
329 //*****Bill Transaction Records*****Bill Transaction Records*****Bill Transaction Records*****Bill Transaction Records*****
330
331 //Inference
332 private void insert_bill() //Create a new method to insert records into Bill table
333 {
334     try
335     {
336         con.Open();
337         cmd = new SqlCommand("INSERT INTO Bill_Rent_Table(Driver_Name,Date,Bill_Amount) VALUES('"+lbl_driver.Text+"','"+DateTime.Today.Date+"','"+
338             "+finalTotal+"')");
339         cmd.ExecuteNonQuery();
340         cmd.Close();
341         MessageBox.Show("Bill added successfully!!!");
342
343         display_data_grid_view_bill(); //data grid view method
344
345     }
346     catch (Exception ex)
347     {
348         MessageBox.Show(ex.Message);
349         con.Close();
350     }
351 }
352
353
354
355 //References
356 public void display_data_grid_view_bill() //For the Bill Table data grid view
357 {
358     try
359     {
360         lbl_driver.Text = Login_Form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
361
362         dt = new DataTable();
363         con.Open();
364         adap = SqlDataAdapter("SELECT * FROM Bill_Rent_Table WHERE Driver_Name = '" + lbl_driver.Text + "'", con);
365         adap.Fill(dt);
366         dgv_Bill_Transaction.DataSource = dt;
367         con.Close();
368     }
369 }

```

Figure 4. 60 Driver logged Rent Vehicle Form Code - part 7

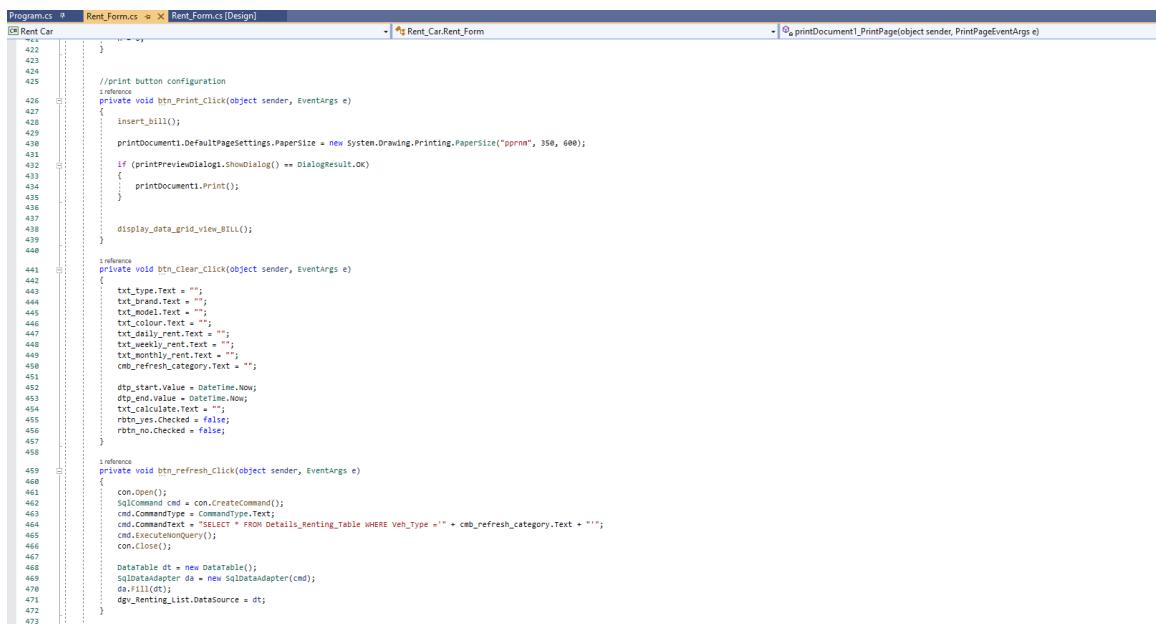
Program.cs 9 Rent_Form.cs -> Rent_Form.cs [Design] Rent_Car.Rent_Form

```

369 //This function will be passed to printing button. --> After printing a bill, this transaction record will be passed to Bill table.
370
371 catch (Exception ex)
372 {
373     MessageBox.Show(ex.Message);
374     con.Close();
375 }
376
377
378
379
380 //*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****
381
382 //Print Document Tool double click and configuration
383 int Rent_Package_ID;
384 int Day_Rent;
385 int Week_Rent;
386 int Month_Rent;
387 int Driver_Charge;
388 int Total;
389 int pos = 60;
390
391 //Inference
392 private void printdocument1_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
393 {
394     e.Graphics.DrawString("Ayubo Drive", new Font("Century Gothic", 12, FontStyle.Bold), Brushes.Red, new Point(80));
395     e.Graphics.DrawString("ID Day Rent Week Rent Month Rent Driver Charge", new Font("Century Gothic", 8, FontStyle.Bold), Brushes.Red, new Point(26, 40));
396
397     foreach (DataGridViewRow row in dgv_Current_Bill.Rows)
398     {
399         Rent_Package_ID = Convert.ToInt32(row.Cells["Column1"].Value);
400         Day_Rent = Convert.ToInt32(row.Cells["Column2"].Value);
401         Week_Rent = Convert.ToInt32(row.Cells["Column3"].Value);
402         Month_Rent = Convert.ToInt32(row.Cells["Column4"].Value);
403         Driver_Charge = Convert.ToInt32(row.Cells["Column5"].Value);
404         Total = Convert.ToInt32(row.Cells["Column6"].Value);
405
406         e.Graphics.DrawString(" " + Rent_Package_ID, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(6, pos));
407         e.Graphics.DrawString(" " + Day_Rent, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(108, pos));
408         e.Graphics.DrawString(" " + Week_Rent, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(138, pos));
409         e.Graphics.DrawString(" " + Month_Rent, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(178, pos));
410         e.Graphics.DrawString(" " + Driver_Charge, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(238, pos));
411         e.Graphics.DrawString(" " + Total, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(298, pos));
412         pos = pos + 20;
413
414
415         e.Graphics.DrawString("Total Total: " + lbl_Amount.Text, new Font("Century Gothic", 18, FontStyle.Italic), Brushes.Crimson, new Point(58, pos + 50));
416         e.Graphics.DrawString("*****Ayubo Drive*****", new Font("Century Gothic", 18, FontStyle.Bold), Brushes.Red, new Point(10, pos + 85));
417         dgv_Current_Bill.Rows.Clear();
418         dgv_Current_Bill.Refresh();
419         pos = 60;
420         lbl_Amount.Text = "0";
421         n = 0;
422     }

```

Figure 4. 61 Driver logged Rent Vehicle Form Code - part 8



The screenshot shows the Visual Studio IDE with the 'Rent_Form.cs' file open. The code is part 9 of a larger program, containing methods for printing, clearing, and refreshing vehicle rental data. It includes references to UI controls like txt_type, txt_brand, txt_model, txt_calculate, and checkboxes for daily, weekly, and monthly rent. It also handles database connections and command execution to refresh the DataGridView (dgv_Renting_List).

```
421 } //>
422 
423 
424 
425 //print button configuration
426 private void btn_Print_Click(object sender, EventArgs e)
427 {
428     insert_bill();
429 
430     printDocument1.DefaultPageSettings.PaperSize = new System.Drawing.Printing.PaperSize("pprnm", 350, 600);
431 
432     if (printPreviewDialog1.ShowDialog() == DialogResult.OK)
433     {
434         printDocument1.Print();
435     }
436 
437 }
438 
439 display_data_grid_view_BILL();
440 
441 private void btn_Clear_Click(object sender, EventArgs e)
442 {
443     txt_type.Text = "";
444     txt_brand.Text = "";
445     txt_model.Text = "";
446     txt_calculate.Text = "";
447     txt_daily_rent.Text = "";
448     txt_weekly_rent.Text = "";
449     txt_monthly_rent.Text = "";
450     cmb_refresh_category.Text = "";
451 
452     dtp_start.Value = DateTime.Now;
453     dtp_end.Value = DateTime.Now;
454     txt_calculate.Text = "";
455     rbt_res.Checked = false;
456     rbt_no.Checked = false;
457 }
458 
459 private void btn_refresh_Click(object sender, EventArgs e)
460 {
461     con.open();
462     SqlCommand cmd = con.CreateCommand();
463     cmd.CommandType = CommandType.Text;
464     cmd.CommandText = "SELECT * FROM Details_Renting_Table WHERE Veh_Type ='" + cmb_refresh_category.Text + "'";
465     cmd.ExecuteNonQuery();
466     con.Close();
467 
468     DataTable dt = new DataTable();
469     SqlDataAdapter da = new SqlDataAdapter(cmd);
470     da.Fill(dt);
471     dgv_Renting_List.DataSource = dt;
472 }
473 }
```

Figure 4. 62 Driver logged Rent Vehicle Form Code - part 9

4.1.12 Driver logged One Day Hire Vehicle Form Design and Code

- Ayubo Drive**
- Rent
- 1 Day Hire
- Tour Hire
- My Profile
- About Us

WELCOME Ryan

One Day Hiring Details

OneD_ID	Veh_Type	Veh_Bond	Veh_Model	Veh_Colour	OneD_Packages	OneD_Per_km_charge	OneD_Max_km_limit	OneD_Extra_km_charge	OneD_Max_hour_limit	OneD_Extra_hour_charge
1	Car	Nissan	FB 15	Grey	Air Port Drop	100	50	120	3	150
2	Car	Nissan	FB 15	Grey	Air Port Pickup	100	50	120	3	150
3	Car	Nissan	FB 15	Grey	100km Per Day	100	100	120	4	200
4	Car	Nissan	FB 15	Grey	200km Per Day	100	200	120	8	200
5	Car	Nissan	FB 15	Grey	Over 200km Per Day	100	300	120	12	200

Hire Package ID 1

Vehicle Details

Type	Brand	Model	Colour
Car	Nissan	FB 15	Grey

Rent Calculation

Reading at Start	Reading at End	=	Distance	Total km Charge
km	km		km	Total Extra km Charge
Start Time	End Time	=	Duration	Total Extra hour Charge
HH MM (Use 24-Hour Format)	HH MM		Hours	Total Hire Charge

Current Bill

Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Max_hour_limit	Extra_hour_charge	Total
*						

Bill Amount : Amount

One Day Hiring Bill Transactions

Bill_ID	Driver_Name	Date	Bill_Amount
2	Ryan	2/3/2022 12:00:00 AM	57290
3	Ryan	2/3/2022 12:00:00 AM	96640
4	Ryan	2/3/2022 12:00:00 AM	249240
5	Ryan	2/3/2022 12:00:00 AM	15600

Figure 4. 63 Driver logged One Day Hire Vehicle Form Design

Ryan Wickramaratne (COL 00081762)

Unit_1:PRO - Programming

158

```
Program.cs 10_Hire_Form.cs 10_Hire_Form.cs [Design] Rent_Car
1 using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class _10_Hire_Form : Form
15     {
16         public _10_Hire_Form()
17         {
18             InitializeComponent();
19
20             display_data_grid_view();
21             display_data_grid_view_bill();
22
23             lbl_driver.Text = Login_Form.user; //Login user name display --> STEP 3 (Username Pass From Login Form To Billing Form)
24         }
25
26         SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRMU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
27         SqlCommand cmd;
28
29
30         //For data grid view
31         SqlDataAdapter adapt;
32         DataTable dt;
33
34
35         //***** Quick Menu BUTTONS *****
36
37         private void btn_rent_Click(object sender, EventArgs e)
38         {
39             Rent_Form obj = new Rent_Form();
40             obj.Show();
41             this.Hide();
42         }
43
44         private void btn_Tour_Hire_Click(object sender, EventArgs e)
45         {
46             Long_Hire_Form obj = new Long_Hire_Form();
47             obj.Show();
48             this.Hide();
49         }
50
51         private void btn_MyProfile_Click(object sender, EventArgs e)
```

Figure 4.64 Driver logged One Day Hire Vehicle Form Code - part 1

```
Program.cs # ID_Hire_Form.cs # ID_Hire_Form.cs [Design] # Rent_Car # _ID_Hire_Form()
Rent_Car
51
52    private void btn_MyProfile_Click(object sender, EventArgs e)
53    {
54        My_Profile_Form obj = new My_Profile_Form();
55        obj.Show();
56        this.Hide();
57    }
58
59    //references
60    private void btn_About_Us_Click(object sender, EventArgs e)
61    {
62        About_Us_Form obj = new About_Us_Form();
63        obj.Show();
64        this.Hide();
65    }
66
67    //references
68    private void btn_Logout_Click(object sender, EventArgs e)
69    {
70        Login_Form obj = new Login_Form();
71        obj.Show();
72        this.Hide();
73    }
74
75    //***** OTHER METHODS *****
76
77    //references
78    public void clear()
79    {
80        txt_Type.Text = "";
81        txt_Brand.Text = "";
82        txt_Rate.Text = "";
83        txt_Colour.Text = "";
84        txt_Per_Km_charge.Text = "";
85        txt_Max_Km_limit.Text = "";
86        txt_Extra_km_charge.Text = "";
87        txt_Mileage_limit.Text = "";
88        txt_Extra_charge.Text = "";
89        txt_Extra_km_charge.Text = "";
90        txt_reading_start.Text = "";
91        txt_reading_end.Text = "";
92        txt_distance.Text = "";
93        txt_total_hire_charge.Text = "";
94        txt_StartTime_H.Text = "";
95        txt_EndTime_H.Text = "";
96        txt_Endtime_M.Text = "";
97        txt_duration.Text = "";
98
99        txt_total_km_charge.Text = "";
100       txt_total_Extra_km_charge.Text = "";
101       txt_total_Extra_hour_charge.Text = "";
102   }
103
104   //references
```

Figure 4.65 Driver logged One Day Hire Vehicle Form Code - part 2

Program.cs 9 ID_Hire_Form.cs □ X ID_Hire_Form.cs [Design] Rent Car Rent_Car_ID_Hire_Form @ _ID_Hire_Form()

```

103     public void display_data_grid_view() //For the data grid view
104     {
105         try
106         {
107             dt = new DataTable();
108             con.open();
109             adapt = new SqlDataAdapter("SELECT * FROM Details_OneD_Hire_Table", con);
110             adapt.Fill(dt);
111             dgv_Hiring_List.DataSource = dt;
112             con.Close();
113         }
114         catch (Exception ex)
115         {
116             MessageBox.Show(ex.Message);
117             con.Close();
118         }
119     }
120
121     int Per_Km_charge;
122     int Max_Km_Limit;
123     int Max_Hour_Limit;
124     int Extra_Hour_Charge;
125
126     private void dgv_Hiring_List_CellClick(object sender, DataGridViewCellEventArgs e)
127     {
128         con.open();
129         int ID;
130
131         ID = int.Parse(dgv_Hiring_List.Rows[e.RowIndex].Cells[0].Value.ToString());
132
133         SqlCommand cmd = con.CreateCommand();
134         cmd.CommandType = CommandType.Text;
135         cmd.CommandText = "SELECT * FROM Details_OneD_Hire_Table WHERE OneD_ID = '" + ID + "' ";
136
137         SqlDataReader DR1 = cmd.ExecuteReader();
138
139         if (DR1.Read())
140         {
141             txt_HireID.Text = DR1.GetValue(0).ToString();
142             txt_Type.Text = DR1.GetValue(1).ToString();
143             txt_Make.Text = DR1.GetValue(2).ToString();
144             txt_Model.Text = DR1.GetValue(3).ToString();
145             txt_Colour.Text = DR1.GetValue(4).ToString();
146             txt_Package.Text = DR1.GetValue(5).ToString();
147             Per_Km_charge = Convert.ToInt32(txt_Per_Km_charge.Text = DR1.GetValue(6).ToString()); //should convert to int, becz we calculate them when adding the bill
148             Max_Km_Limit = Convert.ToInt32(txt_Max_Km_Limit.Text = DR1.GetValue(7).ToString()); //should convert to int, becz we calculate them when adding the bill
149             Extra_Hour_charge = Convert.ToInt32(txt_Extra_Hour_Charge.Text = DR1.GetValue(8).ToString()); //should convert to int, becz we calculate them when adding the bill
150             Max_Hour_Limit = Convert.ToInt32(txt_Max_Hour_Limit.Text = DR1.GetValue(9).ToString()); //should convert to int, becz we calculate them when adding the bill
151             Extra_Hour_charge = Convert.ToInt32(txt_Extra_Hour_Charge.Text = DR1.GetValue(10).ToString()); //should convert to int, becz we calculate them when adding the bill
152         }
153
154         DR1.Close();
155         con.Close();
156     }

```

Figure 4. 66 Driver logged One Day Hire Vehicle Form Code - part 3

Program.cs 9 ID_Hire_Form.cs □ X ID_Hire_Form.cs [Design] Rent_Car_ID_Hire_Form @ _ID_Hire_Form()

```

156     con.Close();
157
158
159 //***** CALCULATION PART *****
160
161 //CALCULATION PART
162
163     private void btn_Calculate_Click(object sender, EventArgs e)
164     {
165         //distance calculation
166         int distance = Convert.ToInt32(txt_reading_end.Text) - Convert.ToInt32(txt_reading_start.Text);
167
168         //duration calculation
169         int duration_gap_in_min = ((Convert.ToInt32(txt_EndTime_H.Text) * 60 + Convert.ToInt32(txt_EndTime_M.Text)) - (Convert.ToInt32(txt_StartTime_H.Text) * 60 + Convert.ToInt32(txt_Starttime_M.Text)));
170         int duration_in_hours = (int)math.Floor(duration_gap_in_min / 60);
171
172         //display data
173         txt_distance.Text = Convert.ToString(distance);
174         txt_duration.Text = Convert.ToString(duration_in_hours);
175
176         if (Convert.ToInt32(txt_distance.Text) <= Convert.ToInt32(txt_Max_Km_Limit.Text) && Convert.ToInt32(txt_duration.Text) <= Convert.ToInt32(txt_Max_Hour_Limit.Text)) //****distance and duration are less than limit
177         {
178             //FOR DISTANCE
179             int Total_Km_charge = Convert.ToInt32(txt_Per_Km_charge.Text) * Convert.ToInt32(txt_distance.Text);
180
181             //display data
182             txt_total_km_charge.Text = Convert.ToString(Total_Km_charge);
183
184             //FOR DURATION
185
186             //display data
187             txt_total_extra_hour_charge.Text = Convert.ToString("0");
188
189             //display other data
190             txt_total_extra_km_charge.Text = Convert.ToString("0");
191             txt_total_hire_charge.Text = txt_total_km_charge.Text;
192
193         }
194
195     else if (Convert.ToInt32(txt_distance.Text) > Convert.ToInt32(txt_Max_Km_Limit.Text) && Convert.ToInt32(txt_duration.Text) <= Convert.ToInt32(txt_Max_Hour_Limit.Text)) //****distance over the limit but duration less than limit
196     {
197             //FOR DISTANCE
198             int Total_Km_charge = Convert.ToInt32(txt_Per_Km_charge.Text) * Convert.ToInt32(txt_Max_Km_Limit.Text);
199             int Total_Extra_Km_charge = (Convert.ToInt32(txt_distance.Text) - Convert.ToInt32(txt_Max_Km_Limit.Text)) * Convert.ToInt32(txt_Extra_Km_charge.Text); //Taking extra distance * Extra per km price
200
201             int total_Dis_charge = Total_Km_charge + Total_Extra_Km_charge;
202
203             //display data
204             txt_total_km_charge.Text = Convert.ToString(total_Dis_charge);
205             txt_total_hire_charge.Text = Convert.ToString(total_Dis_charge);
206
207             //FOR DURATION
208
209         }
210
211     }

```

Figure 4. 67 Driver logged One Day Hire Vehicle Form Code - part 4

```

Program.cs 9  ID_Hire_Form.cs  w X ID_Hire_Form.cs [Design]
Rent_Car
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263

```

Figure 4. 68 Driver logged One Day Hire Vehicle Form Code - part 5

```

Program.cs 9  ID_Hire_Form.cs  w X ID_Hire_Form.cs [Design]
Rent_Car
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314

```

Figure 4. 69 Driver logged One Day Hire Vehicle Form Code - part 6

```

Program.cs  ID_Hire_Form.cs  ID_Hire_Form.cs [Design]
Rent Car
315     cmd.ExecuteNonQuery();
316     con.Close();
317     MessageBox.Show("Bill added successfully!!!");
318
319     display_data_grid_view_BILL(); //data grid view method
320
321 }
322
323 }
324
325 }
326 catch (Exception ex)
327 {
328     MessageBox.Show(ex.Message);
329     con.Close();
330 }
331
332 //Invoices
333 public void display_data_grid_view_BILL() //For the Bill Table data grid view
334 {
335     try
336     {
337         lbl_Driver.Text = Login_Form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
338
339         dt = new DataTable();
340         con.Open();
341         adapt = new SqlDataAdapter("SELECT * FROM BILL_ONED_HIRE_TABLE WHERE Driver_Name = '" + lbl_Driver.Text + "'", con);
342         adapt.Fill(dt);
343         dgv_Bill_Transaction.DataSource = dt;
344         con.Close();
345
346         //This function will be passed to printing button. --> After printing a bill, this transaction record will be passed to Bill table.
347     }
348     catch (Exception ex)
349     {
350         MessageBox.Show(ex.Message);
351         con.Close();
352     }
353
354
355 //*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****
356 //Print Document Tool double click and configuration
357 int Hire_Package_ID;
358 int Per_km_value;
359 int Extra_km_value;
360 int Max_hour_limitation;
361 int Extra_hour_value;
362 int Max_hour_limitation;
363 int Extra_hour_value;
364 int Total;
365 int pos = 60;
366
367 //reference
368 private void printDocument1_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
369 {
370     e.Graphics.DrawString("Ayubo Drive", new Font("Century Gothic", 12, FontStyle.Bold), Brushes.Red, new Point(80));
371 }

```

Figure 4. 70 Driver logged One Day Hire Vehicle Form Code - part 7

```

Program.cs  ID_Hire_Form.cs  ID_Hire_Form.cs [Design]
Rent Car
368     e.Graphics.DrawString("Ayubo Drive", new Font("Century Gothic", 12, FontStyle.Bold), Brushes.Red, new Point(80));
369     e.Graphics.DrawString("ID Per_km_charge Max_km_limit Extra_km_charge Max_hour_limitation Extra_hour_charge TOTAL ", new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Red, new Point(100));
370
371 foreach (DataGridViewRow row in dgv_Current_Bill.Rows)
372 {
373     Hire_Package_ID = Convert.ToInt32(row.Cells["Column1"].Value);
374     Per_km_value = Convert.ToInt32(row.Cells["Column2"].Value);
375     Max_km_limitation = Convert.ToInt32(row.Cells["Column3"].Value);
376     Extra_km_value = Convert.ToInt32(row.Cells["Column4"].Value);
377     Max_hour_limitation = Convert.ToInt32(row.Cells["Column5"].Value);
378     Extra_hour_value = Convert.ToInt32(row.Cells["Column6"].Value);
379     Total = Convert.ToInt32(row.Cells["Column7"].Value);
380
381     e.Graphics.DrawString(" " + Hire_Package_ID, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(20, pos));
382     e.Graphics.DrawString(" " + Per_km_value, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(40, pos));
383     e.Graphics.DrawString(" " + Max_km_limitation, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(120, pos));
384     e.Graphics.DrawString(" " + Extra_km_value, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(170, pos));
385     e.Graphics.DrawString(" " + Max_hour_limitation, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(235, pos));
386     e.Graphics.DrawString(" " + Extra_hour_value, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(295, pos));
387     e.Graphics.DrawString(" " + Total, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(340, pos));
388
389     pos = pos + 20;
390
391
392     e.Graphics.DrawString("Final Total: " + lbl_Amount.Text, new Font("Century Gothic", 10, FontStyle.Italic), Brushes.Crimson, new Point(80, pos + 50));
393     e.Graphics.DrawString("*****Ayubo Drive*****", new Font("Century Gothic", 10, FontStyle.Bold), Brushes.Red, new Point(10, pos + 85));
394     dgv_Current_Bill.Rows.Clear();
395     dgv_Current_Bill.Refresh();
396     pos = 100;
397     lbl_Amount.Text = "0";
398     n = 0;
399 }
400
401 //print button configuration
402 //reference
403 private void btn_Print_Click(object sender, EventArgs e)
404 {
405     insert_bill();
406     printDocument1.DefaultPageSettings.PaperSize = new System.Drawing.Printing.PaperSize("ppnm", 400, 600);
407
408     if (printPreviewDialog1.ShowDialog() == DialogResult.OK)
409     {
410         printDocument1.Print();
411     }
412
413
414     display_data_grid_view_BILL();
415
416
417 //reference
418 private void btn_Clear_Click(object sender, EventArgs e)
419 {
420     txt_type.Text = "";
421     txt_brand.Text = "";
422 }

```

Figure 4. 71 Driver logged One Day Hire Vehicle Form Code - part 8

```

Program.cs 9 | ID_Hire_Form.cs 9 | ID_Hire_Form.cs [Design]
Rent Car | Rent_Car_ID_Hire_Form | btn_AddToBill_Click(object sender, EventArgs e)
417     private void btn_Clear_Click(object sender, EventArgs e)
418     {
419         txt_type.Text = "";
420         txt_brand.Text = "";
421         txt_model.Text = "";
422         txt_color.Text = "";
423         txt_Per_Km_charge.Text = "";
424         txt_Max_Km_limit.Text = "";
425         txt_Extra_Km_charge.Text = "";
426         txt_Extra_Hr_charge.Text = "";
427         txt_Extra_In_charge.Text = "";
428         cmb_refresh_category.Text = "";
429
430         txt_reading_start.Text = "";
431         txt_reading_end.Text = "";
432         txt_distance.Text = "";
433         txt_StartTime_H.Text = "";
434         txt_StartTime_M.Text = "";
435         txt_EndTime_H.Text = "";
436         txt_EndTime_M.Text = "";
437         txt_duration.Text = "";
438
439         txt_total_km_charge.Text = "";
440         txt_total_Extra_km_charge.Text = "";
441         txt_total_Extra_hour_charge.Text = "";
442         txt_total_hire_charge.Text = "";
443     }
444
445     private void btn_refresh_Click(object sender, EventArgs e)
446     {
447         con.Open();
448         SqlCommand cmd = con.CreateCommand();
449         cmd.CommandType = CommandType.Text;
450         cmd.CommandText = "SELECT * FROM Details_One_Hire_Table WHERE Oneo_Packages ='" + cmb_refresh_category.Text + "'";
451         cmd.ExecuteNonQuery();
452         con.Close();
453
454         DataTable dt = new DataTable();
455         SqlDataAdapter da = new SqlDataAdapter(cmd);
456         da.Fill(dt);
457         dgv_Hiring_List.DataSource = dt;
458     }
459
460
461
462
463

```

Figure 4. 72 Driver logged One Day Hire Vehicle Form Code - part 9

4.1.13 Driver logged Long Tour Hire Vehicle Form Design and Code

[Rent](#)
[1 Day Hire](#)
[Tour Hire](#)
[My Profile](#)
[About Us](#)

[LOGOUT](#)

WELCOME Ryan

LongD_ID	Veh_Type	Veh_Brand	Veh_Model	Veh_Colour	LongD_Packages	LongD_Per_Km_Chg	LongD_Max_Km_Fr	LongD_Extra_Km_C	LongD_Driver_Chg	LongD_Vehicle_Pg
1	Car	Nissan	FB 15	Grey	2 days trip	100	200	120	1000	200
2	Car	Nissan	FB 15	Grey	3 days trip	100	300	120	1000	200
3	Car	Nissan	FB 15	Grey	4 days trip	100	400	120	1000	200
4	Car	Nissan	FB 15	Grey	5 days trip	100	500	120	1000	200
5	Car	Nissan	FB 15	Grey	6 days trip	100	600	120	1000	200
6	Car	Nissan	FB 15	Grey	7 days trip	100	700	120	1000	200

Hire Package ID : 3		Vehicle Details			Rent Calculation			Total km Charge	
Type	Brand	Model	Colour	Reading at Start	Reading at End	Distance	km	km	Total Extra km Charge
Car	Nissan	FB 15	Grey			=			
One Day Hiring Rates				Days	Nights				Oversight Driver Charge
Package	Per km Charge	Maximum km Limit	Extra km Charge			=			Vehicle Parking Charge
4 days trip	100	400	120	Driver Charge Overnight	Vehicle Parking Charge				Total Hire Charge
				1000	200				

Hire_package_ID	Per_km_charge	Max_km_limit	Extra_km_charge	Overnight_Charge	Parking_Charge	Total
*						

Bill Amount : Amount [Print](#)

Long Hiring Bill Transactions				
Bil_ID	Driver_Name	Date	Bill_Amount	
2	Ryan	2/3/2022 12:00:00 AM	56400	
3	Ryan	2/3/2022 12:00:00 AM	132800	
4	Ryan	2/3/2022 12:00:00 AM	154000	
5	Ryan	2/3/2022 12:00:00 AM	278800	

Figure 4. 73 Driver logged Long Tour Hire Vehicle Form Design

Figure 4.74 Driver logged Long Tour Hire Vehicle Form Code - part 1

```
Program.cs  x Long_Hire_Form.cs  x Long_Hire_Form.cs [Design]  Rent_Car  Rent_Car.Long_Hire_Form  Long_Hire_Form()
51     private void btn_MyProfile_Click(object sender, EventArgs e)
52     {
53         My_Profile_Form obj = new My_Profile_Form();
54         obj.Show();
55         this.Hide();
56     }
57
58     //reference
59     private void btn_About_Us_Click(object sender, EventArgs e)
60     {
61         About_Us_Form obj = new About_Us_Form();
62         obj.Show();
63         this.Hide();
64     }
65
66     //reference
67     private void btn_Logout_Click(object sender, EventArgs e)
68     {
69         Login_Form obj = new Login_Form();
70         obj.Show();
71         this.Hide();
72     }
73
74     //***** OTHER METHODS *****
75
76     //reference
77     public void clear()
78     {
79         txt_type.Text = "";
80         txt_model.Text = "";
81         txt_colour.Text = "";
82         txt_Per_km_charge.Text = "";
83         txt_Max_km_limit.Text = "";
84         txt_extra_km_charge.Text = "";
85         txt_Max_km_limit.Text = "";
86         txt_Extra_km_charge.Text = "";
87         cmd_refresh_category.Text = "";
88
89         txt_reading_start.Text = "";
90         txt_reading_end.Text = "";
91         txt_distance.Text = "";
92         txt_fuel.Text = "";
93         txt_rights.Text = "";
94
95         txt_total_kg_charge.Text = "";
96         txt_Total_Extra_kg_charge.Text = "";
97         txt_total_overnight_charge.Text = "";
98         txt_drive_night_charge.Text = "";
99         txt_total_hire_charge.Text = "";
100    }
101
102    //reference
103    public void display_data_grid_view() //For the data grid view
```

Figure 4.75 Driver logged Long Tour Hire Vehicle Form Code - part 2

Program.cs 9 Long_Hire_Form.cs 0 X Long_Hire_Form.cs [Design]

```

162     public void display_data_grid_view() //For the data grid view
163     {
164         try
165         {
166             dt = new DataTable();
167             con.Open();
168             adpt = new SqlDataAdapter("SELECT * FROM Details_LongHire_Table", con);
169             adpt.Fill(dt);
170             dgv_Hiring_List.DataSource = dt;
171             con.Close();
172         }
173         catch (Exception ex)
174         {
175             MessageBox.Show(ex.Message);
176             con.Close();
177         }
178     }
179
180     int Per_km_charge;
181     int Max_km_limit;
182     int Extra_km_charge;
183     int Driver_Overnight;
184     int Parking_charge;
185     int ID;
186
187     private void dgv_Hiring_List_CellClick(object sender, DataGridViewCellEventArgs e)
188     {
189         con.Open();
190         ID = int.Parse(dgv_Hiring_List.Rows[e.RowIndex].Cells[0].Value.ToString());
191
192         SqlCommand cmd = con.CreateCommand();
193         cmd.CommandType = CommandType.Text;
194         cmd.CommandText = "SELECT * FROM Details_LongHire_Table WHERE LongHire_ID = '" + ID + "' ";
195
196         SqlDataReader DR1 = cmd.ExecuteReader();
197
198         if (DR1.Read())
199         {
200             txt_hireID.Text = DR1.GetValue(0).ToString();
201             txt_type.Text = DR1.GetValue(1).ToString();
202             txt_brand.Text = DR1.GetValue(2).ToString();
203             txt_color.Text = DR1.GetValue(3).ToString();
204             txt_colour.Text = DR1.GetValue(4).ToString();
205             txt_package.Text = DR1.GetValue(5).ToString();
206             Per_km_charge = Convert.ToInt32(txt_Per_km_charge.Text = DR1.GetValue(6).ToString()); //should Covert to int, becc we calculate them when adding the bill
207             Max_km_limit = Convert.ToInt32(txt_Max_km_limit.Text = DR1.GetValue(7).ToString()); //should Covert to int, becc we calculate them when adding the bill
208             Extra_km_charge = Convert.ToInt32(txt_Extra_km_charge.Text = DR1.GetValue(8).ToString()); //should Covert to int, becc we calculate them when adding the bill
209             Driver_Overnight = Convert.ToInt32(txt_driver_night_charge.Text = DR1.GetValue(9).ToString()); //should Covert to int, becc we calculate them when adding the bill
210             Parking_Charge = Convert.ToInt32(txt_parking_charge.Text = DR1.GetValue(10).ToString()); //should Covert to int, becc we calculate them when adding the bill
211
212         }
213         DR1.Close();
214         con.Close();
215     }
216 
```

Figure 4. 76 Driver logged Long Tour Hire Vehicle Form Code - part 3

Program.cs 9 Long_Hire_Form.cs 0 X Long_Hire_Form.cs [Design]

```

156
157
158     //***** CALCULATION PART *****
159
160     //Nights
161     int nights;
162
163     private void btn_Calculate_Click(object sender, EventArgs e)
164     {
165         //Distance Calculation
166         int distance = Convert.ToInt32(txt_reading_end.Text) - Convert.ToInt32(txt_reading_start.Text);
167
168         //Nights
169         int nights = Convert.ToInt32(txt_days.Text) - 1;
170
171         //Display data
172         txt_distance.Text = Convert.ToString(distance);
173         txt_nights.Text = Convert.ToString(nights);
174
175         if (Convert.ToInt32(txt_distance.Text) <= Convert.ToInt32(txt_Max_km_limit.Text)) //***Distance less than limit
176         {
177             //FOR DISTANCE
178             int Total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * convert.ToInt32(txt_distance.Text);
179
180             //display data
181             txt_total_km_charge.Text = Convert.ToString(Total_km_charge);
182
183             //display other data
184             txt_total_extra_km_charge.Text = Convert.ToString("0");
185             txt_total_parking_charge.Text = Convert.ToString(Convert.ToInt32(txt_driven_night_charge.Text) * Convert.ToInt32(txt_nights.Text));
186             txt_total_parking_charge.Text = Convert.ToString(Convert.ToInt32(txt_parking_charge.Text) * Convert.ToInt32(txt_nights.Text));
187             txt_total_km_charge.Text = Convert.ToString(Convert.ToInt32(txt_total_km_charge.Text) + convert.ToInt32(txt_total_extra_km_charge.Text)+ convert.ToInt32(txt_total_overnight_charge.Text));
188             txt_total_dis_charge = Total_km_charge + Total_km_charge;
189
190         }
191
192         else if (Convert.ToInt32(txt_distance.Text) > Convert.ToInt32(txt_Max_km_limit.Text)) //***Distance over the limit
193         {
194             //FOR DISTANCE
195             int Total_km_charge = Convert.ToInt32(txt_Per_km_charge.Text) * Convert.ToInt32(txt_Max_km_limit.Text);
196             int total_extra_km_charge = (Convert.ToInt32(txt_distance.Text) - Convert.ToInt32(txt_Max_km_limit.Text)) * Convert.ToInt32(txt_Extra_km_charge.Text); //Taking extra distance * Extra per km price
197
198             int total_dis_charge = Total_km_charge + Total_km_charge;
199
200             //display data
201             txt_total_km_charge.Text = Convert.ToString(Total_km_charge);
202             txt_total_extra_km_charge.Text = Convert.ToString(Total_km_charge);
203
204             //display other data
205             txt_total_overnight_charge.Text = Convert.ToString(Convert.ToInt32(txt_driven_night_charge.Text) * Convert.ToInt32(txt_nights.Text));
206             txt_total_parking_charge.Text = Convert.ToString(Convert.ToInt32(txt_parking_charge.Text) * Convert.ToInt32(txt_nights.Text));
207             txt_total_km_charge.Text = Convert.ToString(Convert.ToInt32(txt_total_km_charge.Text) + Convert.ToInt32(txt_total_extra_km_charge.Text));
208             txt_total_dis_charge = Total_km_charge + Total_km_charge;
209         }
210     }
211 
```

Figure 4. 77 Driver logged Long Tour Hire Vehicle Form Code - part 4

```

Program.cs 0 Long_Hire_Form.cs 0 X Long_Hire_Form.cs [Design]
Rent Car
209     * Convert.ToInt32(txt_total_overnight_charge.Text) + Convert.ToInt32(txt_total_parking_charge.Text));
210 
211 }
212 
213 
214 
215 //*****Add to Bill Button*****Add to Bill Button*****Add to Bill Button*****Add to Bill Button*****Add to Bill Button*****
216 
217 
218     int finalTotal = 0;
219     n = 0;
220 
221     private void btn_AddToBill_Click(object sender, EventArgs e)
222     {
223         if (txt_type.Text == "" & txt_brand.Text == "" & txt_model.Text == "" & txt_colour.Text == "" & txt_package.Text == "" & txt_Per_km_charge.Text == "" & txt_Max_km_limit.Text == "")
224         {
225             MessageBox.Show("Enter Package Details Please");
226         }
227         else
228         {
229             DataGridviewRow newrow = new DataGridViewRow();
230             newrow.CreateCells(dgv_Current_Bill);
231             newrow.Cells[0].Value = n + 1;
232             newrow.Cells[1].Value = txt_Per_km_charge.Text;
233             newrow.Cells[2].Value = txt_Max_km_limit.Text;
234             newrow.Cells[3].Value = txt_extra_km_charge.Text;
235             newrow.Cells[4].Value = txt_driver_night_charge.Text;
236             newrow.Cells[5].Value = txt_parking_charge.Text;
237             newrow.Cells[6].Value = txt_total_hire_charge.Text;
238             newrow.Cells[7].Value = txt_total_hire_charge.Text;
239             dgv_Current_Bill.Rows.Add(newrow);
240             n++;
241             //ID Column value increment 1 by 1
242             finalTotal = finalTotal + Convert.ToInt32(txt_total_hire_charge.Text);
243         }
244         lbl_Amount.Text = "Rs " + finalTotal;
245     }
246 
247 }
248 
249 
250 //*****Bill Transaction Records*****Bill Transaction Records*****Bill Transaction Records*****Bill Transaction Records*****
251 
252 //Inference
253     private void insert_bill() //Create a new method to insert records into Bill table
254     {
255         try
256         {
257             con.Open();
258             cmd = new SqlCommand("INSERT INTO Bill_Longo_Hire_Table(Driver_Name,Date,Bill_Amount) VALUES('"+lbl_driver.Text+"','"+DateTime.Today.Date+"','"+
259             "+finalTotal+"')", con);
260             cmd.ExecuteNonQuery();
261         }
262     }

```

Figure 4. 78 Driver logged Long Tour Hire Vehicle Form Code - part 5

```

Program.cs 0 Long_Hire_Form.cs 0 X Long_Hire_Form.cs [Design]
Rent Car
261     cmd.ExecuteNonQuery();
262     con.Close();
263     MessageBox.Show("Bill added successfully!!!");
264 
265     display_data_grid_view_BILL(); //data grid view method
266 
267 }
268 
269 }
270 catch (Exception ex)
271 {
272     MessageBox.Show(ex.Message);
273     con.Close();
274 }
275 
276 }
277 
278 //Inference
279 public void display_data_grid_view_BILL() //For the Bill Table data grid view
280 {
281     try
282     {
283         lbl_driver.Text = Login_Form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
284 
285         dt = new DataTable();
286         con.Open();
287         adapt = new SqlDataAdapter("SELECT * FROM Bill_Longo_Hire_Table WHERE Driver_Name = '" + lbl_driver.Text + "'", con);
288         adapt.Fill(dt);
289         dgv_Bill_Transaction.DataSource = dt;
290         con.Close();
291 
292         //This function will be passed to printing button. --> After printing a bill, this transaction record will be passed to Bill table.
293 
294     catch (Exception ex)
295     {
296         MessageBox.Show(ex.Message);
297         con.Close();
298     }
299 
300 }
301 
302 //*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****For Bill Printing*****
303 
304 //Print Document Tool double click and configuration
305     int hire_Package_ID;
306     int Per_km_value;
307     int Max_km_limited;
308     int Extra_km_value;
309     int Night_overnight_Charge;
310     int Parking_charge;
311     int Total;
312     int pos = 60;
313 
314     private void printDocument1_PrintPage(object sender, System.Drawing.Printing.PrintPageEventArgs e)
315     {
316         e.Graphics.DrawString("Ayubo Drive", new Font("Century Gothic", 12, FontStyle.Bold), Brushes.Red, new Point(80));

```

Figure 4. 79 Driver logged Long Tour Hire Vehicle Form Code - part 6

```
Program.cs    Long_Hire_Form.cs x Long_Hire_Form.cs [Design] - Rent_Car.Long_Hire_Form + display_data_grd_view_BILL()
Rent Car

314     e.Graphics.DrawString("Ayubo Drive", new Font("Century Gothic", 12, FontStyle.Bold), Brushes.Red, new Point(80));
315     e.Graphics.DrawString("ID Per Km_charge Max Km_limit Extra Km_charge Driver_night_Charge Parking_charge TOTAL ", new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Red,
316     new Point(26, 40));
317
318     foreach (DataRow dr in dgv_Current_Bill.Rows)
319     {
320         Hire_Package_ID = Convert.ToInt32(dr["cells"]["Column1"].Value);
321         Per_Km_value = Convert.ToInt32(dr["cells"]["Column2"].Value);
322         Max_Km_limitation = Convert.ToInt32(dr["cells"]["Column3"].Value);
323         Extra_Km_value = Convert.ToInt32(dr["cells"]["Column4"].Value);
324         Driver_Night_Charge = Convert.ToInt32(dr["cells"]["Column5"].Value);
325         Parking_charge = Convert.ToInt32(dr["cells"]["Column6"].Value);
326         Total = Convert.ToInt32(dr["cells"]["Column7"].Value);
327
328         e.Graphics.DrawString(" " + Hire_Package_ID, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(26, pos));
329         e.Graphics.DrawString(" " + Per_Km_value, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(45, pos));
330         e.Graphics.DrawString(" " + Max_Km_limitation, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(120, pos));
331         e.Graphics.DrawString(" " + Extra_Km_value, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(176, pos));
332         e.Graphics.DrawString(" " + Driver_Night_Charge, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(235, pos));
333         e.Graphics.DrawString(" " + Parking_charge, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(295, pos));
334         e.Graphics.DrawString(" " + Total, new Font("Century Gothic", 4, FontStyle.Bold), Brushes.Blue, new Point(345, pos));
335         pos = pos + 20;
336     }
337
338
339     e.Graphics.DrawString("Final Total: " + lbl_Amount.Text, new Font("Century Gothic", 10, FontStyle.Italic), Brushes.Crimson, new Point(50, pos + 50));
340     e.Graphics.DrawString("*****Ayubo Drive*****", new Font("Century Gothic", 10, FontStyle.Bold), Brushes.Ted, new Point(10, pos + 85));
341     dgv_Current_Bill.Rows.Clear();
342     dgv_Current_Bill.Refresh();
343     pos = 100;
344     lbl_Amount.Text = "0";
345     n = 8;
346 }
347
348 //reference
349 private void btn_Print_Click(object sender, EventArgs e)
350 {
351     insert_bill();
352
353     printDocument1.DefaultPageSettings.PaperSize = new System.Drawing.Printing.PaperSize("ppnm", 400, 600);
354
355     if (printPreviewDialog1.ShowDialog() == DialogResult.OK)
356     {
357         printDocument1.Print();
358     }
359
360
361     display_data_grid_view_BILL();
362 }
363
364 //reference
365 private void btn_Clear_Click(object sender, EventArgs e)
366 {
367     txt_type.Text = "";
368     txt_start.Text = "";
369 }
```

Figure 4.80 Driver logged Long Tour Hire Vehicle Form Code - part 7

Figure 4.81 Driver logged Long Tour Hire Vehicle Form Code - part 8

4.1.14 Driver logged My Profile Form Design and Code

The screenshot shows the Ayubo Drive application interface. At the top, there's a blue header bar with the 'WELCOME' message and the user's name 'Ryan'. On the left side, there's a sidebar with icons for 'Rent', '1 Day Hire', 'Tour Hire', 'My Profile' (which is highlighted in red), and 'About Us'. Below the sidebar is a 'LOGOUT' button. The main content area has a light green background. It displays a 'Driver Details' section with fields for 'Driver ID' (containing '1'), 'Username' ('Ryan'), 'Password' ('Ryan'), 'Email' ('ryandilthusha@gmail.com'), 'Phone Number' ('0764170647'), and 'BirthDay' ('2/25/1997'). There are 'Edit' and 'Clear' buttons below these fields. Below this is a 'Drivers List' section with a table:

Driver_ID	Driver_Username	Driver_Password	Driver_Email	Driver_Phone	Driver_DOB
1	Ryan	Ryan	ryandilthusha@gmail.com	0764170647	1997-02-25
*					

Figure 4. 82 Driver logged My Profile Form Design

```
Program.cs # My_Profile_Form.cs => My_Profile_Form.cs [Design] | Rent_CarMy_Profile_Form - @ My_Profile_Form()
```

```
using System;
2 using System.Collections.Generic;
3 using System.ComponentModel;
4 using System.Data;
5 using System.Drawing;
6 using System.Linq;
7 using System.Text;
8 using System.Threading.Tasks;
9 using System.Windows.Forms;
10 using System.Data.SqlClient;
11
12 namespace Rent_Car
13 {
14     public partial class My_Profile_Form : Form
15     {
16         public My_Profile_Form()
17         {
18             InitializeComponent();
19
20             display_data_grid_view();
21
22             lbl_driver.Text = login_form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
23
24             SqlConnection con = new SqlConnection(@"Data Source=DESKTOP-44KSVRU;Initial Catalog=Rent_Car_Project;Integrated Security=True");
25             SqlCommand cmd;
26
27
28             //For data grid view
29             SqlDataAdapter adapt;
30             DataTable dt;
31
32
33
34             /****** Quick Menu BUTTONS *****/
35
36             reference
37             private void btn_rent_Click(object sender, EventArgs e)
38             {
39                 Rent_Form obj = new Rent_Form();
40                 obj.Show();
41                 this.Hide();
42             }
43
44             reference
45             private void btn_ID_Hire_Click(object sender, EventArgs e)
46             {
47                 _ID_Hire_Form Obj = new _ID_Hire_Form();
48                 Obj.Show();
49                 this.Hide();
50             }
51
52             reference
53             private void btn_Tour_Hire_Click(object sender, EventArgs e)
```

Figure 4. 83 Driver logged My Profile Form Code - part 1

```
Program.cs 0 My_Profile_Form.cs => My_Profile_Form.cs [Design] - Rent_Car.Rent_Car.My_Profile_Form - My_Profile_Form()
```

```
51     {
52         Long_Hire_Form obj = new Long_Hire_Form();
53         obj.Show();
54         this.Hide();
55     }
56 
57     //Logout
58     private void btn_About_Us_Click(object sender, EventArgs e)
59     {
60         About_Us_Form obj = new About_Us_Form();
61         obj.Show();
62         this.Hide();
63     }
64 
65     //Logout
66     private void btn_Logout_Click(object sender, EventArgs e)
67     {
68         Login_Form obj = new Login_Form();
69         obj.Show();
70         this.Hide();
71     }
72 
73     //***** OTHER METHODS *****
74 
75     //reference
76     public void clear()
77     {
78         txt_driverID.Text = "";
79         txt_username.Text = "";
80         txt_password.Text = "";
81         txt_email.Text = "";
82         txt_phone.Text = "";
83         dtb_DOB.Value = DateTime.Now;
84     }
85 
86     //reference
87     public void display_data_grid_view() //For the data grid view
88     {
89         try
90         {
91             lbl_driver.Text = Login_Form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
92             dt = new DataTable();
93             con.Open();
94             adapt = new SqlDataAdapter("SELECT * FROM Details_Drivers_Table WHERE Driver_Username ='" + lbl_driver.Text + "'", con);
95             adapt.Fill(dt);
96             dgw_driver.DataSource = dt;
97             dgw_driver.DataBind();
98         }
99         catch (Exception ex)
100         {
101             MessageBox.Show(ex.Message);
102             con.Close();
103         }
104     }
105 }
```

Figure 4. 84 Driver logged My Profile Form Code - part 2

```

Program.cs 9 My_Profile_Form.cs 0 × My_Profile_Form.cs [Design]
Rent_Car
183
184     private void dgv_drivers_CellClick(object sender, DataGridViewCellEventArgs e)
185     {
186         con.open();
187         int ID;
188
189         ID = int.Parse(dgv_drivers.Rows[e.RowIndex].Cells[0].Value.ToString());
190
191         SqlCommand cmd = con.CreateCommand();
192         cmd.CommandType = CommandType.Text;
193         cmd.CommandText = "SELECT * FROM details_Drivers_Table WHERE Driver_ID = '" + ID + "' ";
194
195         SqlDataReader DR1 = cmd.ExecuteReader();
196
197         if (DR1.read())
198         {
199             txt_driverID.Text = DR1.GetValue(0).ToString();
200             txt_username.Text = DR1.GetValue(1).ToString();
201             txt_password.Text = DR1.GetValue(2).ToString();
202             txt_email.Text = DR1.GetValue(3).ToString();
203             txt_phone.Text = DR1.GetValue(4).ToString();
204             dtp_DOB.Value = DateTime.Parse(DR1.GetValue(5).ToString());
205
206         }
207         DR1.Close();
208         con.Close();
209     }
210
211
212 //*****SAVE*****Edit*****Delete*****
213
214     private void btn_Edit_Click(object sender, EventArgs e)
215     {
216         if (txt_username.Text == "" || txt_password.Text == "" || txt_email.Text == "" || txt_phone.Text == "")
217         {
218             MessageBox.Show("Missing Information");
219         }
220         else
221         {
222             try
223             {
224                 string date = Convert.ToString(dtp_DOB.Value);
225                 date = date.Remove(date.Length - 12);
226
227                 con.Open();
228                 cmd = new SqlCommand("UPDATE details_Drivers_Table SET Driver_Username = '" + txt_username.Text + "', Driver_Password = '" + txt_password.Text + "', Driver_Email = '" + txt_email.Text + "', Driver_Phone = '" + txt_phone.Text + "', Driver_DOB = '" + date + "' WHERE Driver_ID = '" + txt_driverID.Text + "'", con);
229                 cmd.ExecuteNonQuery();
230                 con.Close();
231                 MessageBox.Show("Your're details edited successfully!!!");
232
233                 display_data_grid_view(); //data grid view method
234                 clear(); //data clear method
235             }
236             catch (Exception ex)
237             {
238                 MessageBox.Show(ex.Message);
239                 Con.Close();
240             }
241         }
242     }
243
244     private void btn_Clear_Click(object sender, EventArgs e)
245     {
246         txt_driverID.Text = "";
247         txt_username.Text = "";
248         txt_password.Text = "";
249         txt_email.Text = "";
250         txt_phone.Text = "";
251         dtp_DOB.Value = DateTime.Now;
252     }
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
627
628
629
629
630
631
632
633
633
634
635
635
636
636
637
637
638
638
639
639
640
640
641
641
642
642
643
643
644
644
645
645
646
646
647
647
648
648
649
649
650
650
651
651
652
652
653
653
654
654
655
655
656
656
657
657
658
658
659
659
660
660
661
661
662
662
663
663
664
664
665
665
666
666
667
667
668
668
669
669
670
670
671
671
672
672
673
673
674
674
675
675
676
676
677
677
678
678
679
679
680
680
681
681
682
682
683
683
684
684
685
685
686
686
687
687
688
688
689
689
690
690
691
691
692
692
693
693
694
694
695
695
696
696
697
697
698
698
699
699
700
700
701
701
702
702
703
703
704
704
705
705
706
706
707
707
708
708
709
709
710
710
711
711
712
712
713
713
714
714
715
715
716
716
717
717
718
718
719
719
720
720
721
721
722
722
723
723
724
724
725
725
726
726
727
727
728
728
729
729
730
730
731
731
732
732
733
733
734
734
735
735
736
736
737
737
738
738
739
739
740
740
741
741
742
742
743
743
744
744
745
745
746
746
747
747
748
748
749
749
750
750
751
751
752
752
753
753
754
754
755
755
756
756
757
757
758
758
759
759
760
760
761
761
762
762
763
763
764
764
765
765
766
766
767
767
768
768
769
769
770
770
771
771
772
772
773
773
774
774
775
775
776
776
777
777
778
778
779
779
780
780
781
781
782
782
783
783
784
784
785
785
786
786
787
787
788
788
789
789
790
790
791
791
792
792
793
793
794
794
795
795
796
796
797
797
798
798
799
799
800
800
801
801
802
802
803
803
804
804
805
805
806
806
807
807
808
808
809
809
810
810
811
811
812
812
813
813
814
814
815
815
816
816
817
817
818
818
819
819
820
820
821
821
822
822
823
823
824
824
825
825
826
826
827
827
828
828
829
829
830
830
831
831
832
832
833
833
834
834
835
835
836
836
837
837
838
838
839
839
840
840
841
841
842
842
843
843
844
844
845
845
846
846
847
847
848
848
849
849
850
850
851
851
852
852
853
853
854
854
855
855
856
856
857
857
858
858
859
859
860
860
861
861
862
862
863
863
864
864
865
865
866
866
867
867
868
868
869
869
870
870
871
871
872
872
873
873
874
874
875
875
876
876
877
877
878
878
879
879
880
880
881
881
882
882
883
883
884
884
885
885
886
886
887
887
888
888
889
889
890
890
891
891
892
892
893
893
894
894
895
895
896
896
897
897
898
898
899
899
900
900
901
901
902
902
903
903
904
904
905
905
906
906
907
907
908
908
909
909
910
910
911
911
912
912
913
913
914
914
915
915
916
916
917
917
918
918
919
919
920
920
921
921
922
922
923
923
924
924
925
925
926
926
927
927
928
928
929
929
930
930
931
931
932
932
933
933
934
934
935
935
936
936
937
937
938
938
939
939
940
940
941
941
942
942
943
943
944
944
945
945
946
946
947
947
948
948
949
949
950
950
951
951
952
952
953
953
954
954
955
955
956
956
957
957
958
958
959
959
960
960
961
961
962
962
963
963
964
964
965
965
966
966
967
967
968
968
969
969
970
970
971
971
972
972
973
973
974
974
975
975
976
976
977
977
978
978
979
979
980
980
981
981
982
982
983
983
984
984
985
985
986
986
987
987
988
988
989
989
990
990
991
991
992
992
993
993
994
994
995
995
996
996
997
997
998
998
999
999
1000
1000
1001
1001
1002
1002
1003
1003
1004
1004
1005
1005
1006
1006
1007
1007
1008
1008
1009
1009
1010
1010
1011
1011
1012
1012
1013
1013
1014
1014
1015
1015
1016
1016
1017
1017
1018
1018
1019
1019
1020
1020
1021
1021
1022
1022
1023
1023
1024
1024
1025
1025
1026
1026
1027
1027
1028
1028
1029
1029
1030
1030
1031
1031
1032
1032
1033
1033
1034
1034
1035
1035
1036
1036
1037
1037
1038
1038
1039
1039
1040
1040
1041
1041
1042
1042
1043
1043
1044
1044
1045
1045
1046
1046
1047
1047
1048
1048
1049
1049
1050
1050
1051
1051
1052
1052
1053
1053
1054
1054
1055
1055
1056
1056
1057
1057
1058
1058
1059
1059
1060
1060
1061
1061
1062
1062
1063
1063
1064
1064
1065
1065
1066
1066
1067
1067
1068
1068
1069
1069
1070
1070
1071
1071
1072
1072
1073
1073
1074
1074
1075
1075
1076
1076
1077
1077
1078
1078
1079
1079
1080
1080
1081
1081
1082
1082
1083
1083
1084
1084
1085
1085
1086
1086
1087
1087
1088
1088
1089
1089
1090
1090
1091
1091
1092
1092
1093
1093
1094
1094
1095
1095
1096
1096
1097
1097
1098
1098
1099
1099
1100
1100
1101
1101
1102
1102
1103
1103
1104
1104
1105
1105
1106
1106
1107
1107
1108
1108
1109
1109
1110
1110
1111
1111
1112
1112
1113
1113
1114
1114
1115
1115
1116
1116
1117
1117
1118
1118
1119
1119
1120
1120
1121
1121
1122
1122
1123
1123
1124
1124
1125
1125
1126
1126
1127
1127
1128
1128
1129
1129
1130
1130
1131
1131
1132
1132
1133
1133
1134
1134
1135
1135
1136
1136
1137
1137
1138
1138
1139
1139
1140
1140
1141
1141
1142
1142
1143
1143
1144
1144
1145
1145
1146
1146
1147
1147
1148
1148
1149
1149
1150
1150
1151
1151
1152
1152
1153
1153
1154
1154
1155
1155
1156
1156
1157
1157
1158
1158
1159
1159
1160
1160
1161
1161
1162
1162
1163
1163
1164
1164
1165
1165
1166
1166
1167
1167
1168
1168
1169
1169
1170
1170
1171
1171
1172
1172
1173
1173
1174
1174
1175
1175
1176
1176
1177
1177
1178
1178
1179
1179
1180
1180
1181
1181
1182
1182
1183
1183
1184
1184
1185
1185
1186
1186
1187
1187
1188
1188
1189
1189
1190
1190
1191
1191
1192
1192
1193
1193
1194
1194
1195
1195
1196
1196
1197
1197
1198
1198
1199
1199
1200
1200
1201
1201
1202
1202
1203
1203
1204
1204
1205
1205
1206
1206
1207
1207
1208
1208
1209
1209
1210
1210
1211
1211
1212
1212
1213
1213
1214
1214
1215
1215
1216
1216
1217
1217
1218
1218
1219
1219
1220
1220
1221
1221
1222
1222
1223
1223
1224
1224
1225
1225
1226
1226
1227
1227
1228
1228
1229
1229
1230
1230
1231
1231
1232
1232
1233
1233
1234
1234
1235
1235
1236
1236
1237
1237
1238
1238
1239
1239
1240
1240
1241
1241
1242
1242
1243
1243
1244
1244
1245
1245
1246
1246
1247
1247
1248
1248
1249
1249
1250
1250
1251
1251
1252
1252
1253
1253
1254
1254
1255
1255
1256
1256
1257
1257
1258
1258
1259
1259
1260
1260
1261
1261
1262
1262
1263
1263
1264
1264
1265
1265
1266
1266
1267
1267
1268
1268
1269
1269
1270
1270
1271
1271
1272
1272
1273
1273
1274
1274
1275
1275
1276
1276
1277
1277
1278
1278
1279
1279
1280
1280
1281
1281
1282
1282
1283
1283
1284
1284
1285
1285
1286
1286
1287
1287
1288
1288
1289
1289
1290
1290
1291
1291
1292
1292
1293
1293
1294
1294
1295
1295
1296
1296
1297
1297
1298
1298
1299
1299
1300
1300
1301
1301
1302
1302
1303
1303
1304
1304
1305
1305
1306
1306
1307
1307
1308
1308
1309
1309
1310
1310
1311
1311
1312
1312
1313
1313
1314
1314
1315
1315
1316
1316
1317
1317
1318
1318
1319
1319
1320
1320
1321
1321
1322
1322
1323
1323
1324
1324
1325
1325
1326
1326
1327
1327
1328
1328
1329
1329
1330
1330
1331
1331
1332
1332
1333
1333
1334
1334
1335
1335
1336
1336
1337
1337
1338
1338
1339
1339
1340
1340
1341
1341
1342
1342
1343
1343
1344
1344
1345
1345
1346
1346
1347
1347
1348
1348
1349
1349
1350
1350
1351
1351
1352
1352
1353
1353
1354
1354
1355
1355
1356
1356
1357
1357
1358
1358
1359
1359
1360
1360
1361
1361
1362
1362
1363
1363
1364
1364
1365
1365
1366
1366
1367
1367
1368
1368
1369
1369
1370
1370
1371
1371
1372
1372
1373
1373
1374
1374
1375
1375
1376
1376
1377
1377
1378
1378
1379
1379
1380
1380
1381
1381
1382
1382
1383
1383
1384
1384
1385
1385
1386
1386
1387
1387
1388
1388
1389
1389
1390
1390
1391
1391
1392
1392
1393
1393
1394
1394
1395
1395
1396
1396
1397
1397
1398
1398
1399
1399
1400
1400
1401
1401
1402
1402
1403
1403
1404
1404
1405
1405
1406
1406
1407
1407
1408
1408
1409
1409
1410
1410
1411
1411
1412
1412
1413
1413
1414
1414
1415
1415
1416
1416
1417
1417
1418
1418
1419
1419
1420
1420
1421
1421
1422
1422
1423
1423
1424
1424
1425
1425
1426
1426
1427
1427
1428
1428
1429
1429
1430
1430
1431
1431
1432
1432
1433
1433
1434
1434
1435
1435
1436
1436
1437
1437
1438
1438
1439
1439
1440
1440
1441
1441
1442
1442
1443
1443
1444
1444
1445
1445
1446
1446
1447
1447
1448
1448
1449
1449
1450
1450
1451
1451
1452
1452
1453
1453
1454
1454
1455
1455
1456
1456
1457
1457
1458
1458
1459
1459
1460
1460
1461
1461
1462
1462
1463
1463
1464
1464
1465
1465
1466
1466
1467
1467
1468
1468
1469
1469
1470
1470
1471
1471
1472
1472
1473
1473
1474
1474
1475
1475
1476
1476
1477
1477
1478
1478
1479
1479
1480
1480
1481
1481
1482
1482
1483
1483
1484
1484
1485
1485
1486
1486
1487
1487
1488
1488
1489
1489
1490
1490
1491
1491
1492
1492
1493
1493
1494
1494
1495
1495
1496
1496
1497
1497
1498
1498
1499
1499
1500
1500
1501
1501
1502
1502
1503
1503
1504
1504
1505
1505
1506
1506
1507
1507
1508
1508
1509
1509
1510
1510
1511
1511
1512
1512
1513
1513
1514
1514
1515
1515
1516
1516
1517
1517
1518
1518
1519
1519
1520
1520
1521
1521
1522
1522
1523
1523
1524
1524
1525
1525
1526
1526
1527
1527
1528
1528
1529
1529
1530
1530
1531
1531
1532
1532
1533
1533
1534
1534
1535
1535
1536
1536
1537
1537
1538
1538
1539
1539
1540
1540
1541
1541
1542
1542
1543
1543
1544
1544
1545
1545
1546
1546
1547
1547
1548
1548
1549
1549
1550
1550
1551
1551
1552
1552
1553
1553
1554
1554
1555
1555
1556
1556
1557
1557
1558
1558
1559
1559
1560
1560
1561
1561
1562
1562
1563
1563
1564
1564
1565
1565
1566
1566
1567
1567
1568
1568
1569
1569
1570
1570
1571
1571
1572
1572
1573
1573
1574
1574
1575
1575
1576
1576
1577
1577
1578
1578
1579
1579
1580
1580
1581
1581
1582
1582
1583
```

4.1.15 Driver logged About Us Form Design and Code

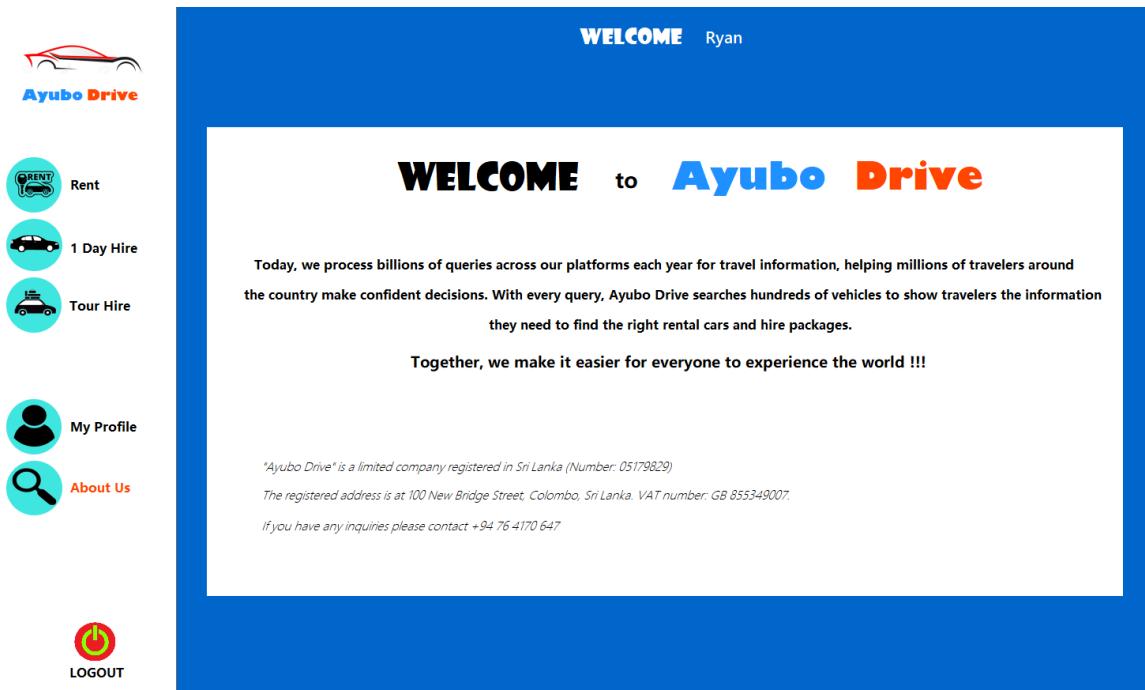


Figure 4. 87 Driver logged About Us Form Design

```

1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10 
11 namespace Rent_Car
12 {
13     public partial class About_Us_Form : Form
14     {
15         public About_Us_Form()
16         {
17             InitializeComponent();
18         }
19         lbl_driver.Text = Login_form.user; //Login user name display --> STEP 3 (Username Pass from Login Form to Billing Form)
20     }
21 
22     //***** Quick Menu BUTTONS *****
23 
24     private void btn_rent_Click(object sender, EventArgs e)
25     {
26         Rent_form obj = new Rent_form();
27         obj.Show();
28         this.Hide();
29     }
30 
31     private void btn_ID_Hire_Click(object sender, EventArgs e)
32     {
33         _ID_Hire_form obj = new _ID_Hire_form();
34         obj.Show();
35         this.Hide();
36     }
37 
38     private void btn_Tour_Hire_Click(object sender, EventArgs e)
39     {
40         Long_Wire_form obj = new Long_Wire_form();
41         obj.Show();
42         this.Hide();
43     }
44 
45     private void btn_MyProfile_Click(object sender, EventArgs e)
46     {
47         My_Profile_form obj = new My_Profile_form();
48         obj.Show();
49         this.Hide();
50     }
51 
52     private void btn_Logout_Click(object sender, EventArgs e)
53     {
54         Login_form obj = new Login_form();
55         obj.Show();
56         this.Hide();
57     }
58 
59 
60 
61 
62 
63 
64 
65

```

Figure 4. 88 Driver logged About Us Form Code - part 1

```

39     private void btn_Logout_Click(object sender, EventArgs e)
40     {
41         Login_form obj = new Login_form();
42         obj.Show();
43         this.Hide();
44     }
45 
46     private void btn_MyProfile_Click(object sender, EventArgs e)
47     {
48         My_Profile_form obj = new My_Profile_form();
49         obj.Show();
50         this.Hide();
51     }
52 
53     private void btn_Logout_Click(object sender, EventArgs e)
54     {
55         Login_form obj = new Login_form();
56         obj.Show();
57         this.Hide();
58     }
59 
60 
61 
62 
63 
64 
65

```

Figure 4. 89 Driver logged About Us Form Code - part 2

4.2 Microsoft SQL table designs and queries

4.2.1 Vehicle Details Table

```
|CREATE DATABASE Rent_Car_Project;

|CREATE TABLE [dbo].[Details_Vehicle_Table] (
    [Veh_ID]      INT          IDENTITY (1, 1) NOT NULL,
    [Veh_Type]    VARCHAR (50) NOT NULL,
    [Veh_RegNo]   VARCHAR (50) NOT NULL,
    [Veh_Brand]   VARCHAR (50) NOT NULL,
    [Veh_Model]   VARCHAR (50) NOT NULL,
    [Veh_Colour]  VARCHAR (50) NOT NULL,
    [Own_Name]   VARCHAR (50) NOT NULL,
    [Own_ID]     VARCHAR (50) NOT NULL,
    [Own_Phone]  VARCHAR (50) NOT NULL,
    PRIMARY KEY CLUSTERED ([Veh_ID] ASC)
);
```

Figure 4. 90 Vehicle Details Table SQL query

Column Name	Data Type	Allow Nulls
Veh_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_RegNo	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
Own_Name	varchar(50)	<input type="checkbox"/>
Own_ID	varchar(50)	<input type="checkbox"/>
Own_Phone	varchar(50)	<input type="checkbox"/>

Figure 4. 91 Vehicle Details Table SQL table design

4.2.2 Vehicle Renting Details Table

```
CREATE TABLE [dbo].[Details_Renting_Table] (
    [Rent_ID]           INT          IDENTITY (1, 1) NOT NULL,
    [Veh_Type]          VARCHAR (50) NOT NULL,
    [Veh_Brand]          VARCHAR (50) NOT NULL,
    [Veh_Model]          VARCHAR (50) NOT NULL,
    [Veh_Colour]         VARCHAR (50) NOT NULL,
    [Rent_Daily]         INT          NOT NULL,
    [Rent_Weekly]        INT          NOT NULL,
    [Rent_Monthly]       INT          NOT NULL,
    [Rent_DriverRate]    INT          NOT NULL,
    PRIMARY KEY CLUSTERED ([Rent_ID] ASC)
);
```

Figure 4. 92 Vehicle Renting Details Table SQL query

DESKTOP-44KSVRU.R...ails_Renting_Table ➔ X Object Explorer Details		
Column Name	Data Type	Allow Nulls
Rent_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
Rent_Daily	int	<input type="checkbox"/>
Rent_Weekly	int	<input type="checkbox"/>
Rent_Monthly	int	<input type="checkbox"/>
Rent_DriverRate	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 4. 93 Vehicle Renting Details Table SQL table design

4.2.3 Vehicle One Day Hire Details Table

```

CREATE TABLE [dbo].[Details_OneD_Hire_Table] (
    [OneD_ID]           INT          IDENTITY (1, 1) NOT NULL,
    [Veh_Type]          VARCHAR (50) NOT NULL,
    [Veh_Brand]          VARCHAR (50) NOT NULL,
    [Veh_Model]          VARCHAR (50) NOT NULL,
    [Veh_Colour]         VARCHAR (50) NOT NULL,
    [OneD_Packages]      VARCHAR (50) NOT NULL,
    [OneD_Per_km_charge] INT          NOT NULL,
    [OneD_Max_km_limit] INT          NOT NULL,
    [OneD_Extra_km_charge] INT          NOT NULL,
    [OneD_Max_hour_limit] INT          NOT NULL,
    [OneD_Extra_hour_charge] INT          NOT NULL,
    PRIMARY KEY CLUSTERED ([OneD_ID] ASC)
);

```

Figure 4. 94 Vehicle One Day Hire Details Table SQL query

Column Name	Data Type	Allow Nulls
OneD_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
OneD_Packages	varchar(50)	<input type="checkbox"/>
OneD_Per_km_charge	int	<input type="checkbox"/>
OneD_Max_km_limit	int	<input type="checkbox"/>
OneD_Extra_km_charge	int	<input type="checkbox"/>
OneD_Max_hour_limit	int	<input type="checkbox"/>
OneD_Extra_hour_charge	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 4. 95 Vehicle One Day Hire Details Table SQL table design

4.2.4 Vehicle Long Tour Hire Details Table

```

CREATE TABLE [dbo].[Details_LongD_Hire_Table] (
    [LongD_ID]           INT          IDENTITY (1, 1) NOT NULL,
    [Veh_Type]            VARCHAR (50) NOT NULL,
    [Veh_Brand]           VARCHAR (50) NOT NULL,
    [Veh_Model]           VARCHAR (50) NOT NULL,
    [Veh_Colour]          VARCHAR (50) NOT NULL,
    [LongD_Packages]      VARCHAR (50) NOT NULL,
    [LongD_Per_km_charge] INT          NOT NULL,
    [LongD_Max_km_limit]  INT          NOT NULL,
    [LongD_Extra_km_charge] INT          NOT NULL,
    [LongD_Driver_overnight_charge] INT          NOT NULL,
    [LongD_Vehicle_parking_charge]  INT          NOT NULL,
    PRIMARY KEY CLUSTERED ([LongD_ID] ASC)
);

```

Figure 4. 96 Vehicle One Day Hire Details Table SQL query

Column Name	Data Type	Allow Nulls
LongD_ID	int	<input type="checkbox"/>
Veh_Type	varchar(50)	<input type="checkbox"/>
Veh_Brand	varchar(50)	<input type="checkbox"/>
Veh_Model	varchar(50)	<input type="checkbox"/>
Veh_Colour	varchar(50)	<input type="checkbox"/>
LongD_Packages	varchar(50)	<input type="checkbox"/>
LongD_Per_km_charge	int	<input type="checkbox"/>
LongD_Max_km_limit	int	<input type="checkbox"/>
LongD_Extra_km_charge	int	<input type="checkbox"/>
LongD_Driver_overnight....	int	<input type="checkbox"/>
LongD_Vehicle_parking_...	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 4. 97 Vehicle One Day Hire Details Table SQL table design

4.2.5 Driver Details Table

```
CREATE TABLE [dbo].[Details_Drivers_Table] (
    [Driver_ID]           INT            IDENTITY (1, 1) NOT NULL,
    [Driver_Username]     VARCHAR (50)   NOT NULL,
    [Driver_Password]     VARCHAR (50)   NOT NULL,
    [Driver_Email]        VARCHAR (50)   NOT NULL,
    [Driver_Phone]        VARCHAR (50)   NOT NULL,
    [Driver_DOB]          VARCHAR (50)   NOT NULL,
    PRIMARY KEY CLUSTERED ([Driver_ID] ASC)
);
```

Figure 4. 98 Driver Details Table SQL query

Column Name	Data Type	Allow Nulls
Driver_ID	int	<input type="checkbox"/>
Driver_Username	varchar(50)	<input type="checkbox"/>
Driver_Password	varchar(50)	<input type="checkbox"/>
Driver_Email	varchar(50)	<input type="checkbox"/>
Driver_Phone	varchar(50)	<input type="checkbox"/>
Driver_DOB	varchar(50)	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 4. 99 Driver Details Table SQL table design

4.2.6 Vehicle Renting Bills Details Table

```
CREATE TABLE [dbo].[Bill_Rent_Table] (
    [Bill_ID]           INT            IDENTITY (1, 1) NOT NULL,
    [Driver_Name]       VARCHAR (50)   NOT NULL,
    [Date]              VARCHAR(50)    NOT NULL,
    [Bill_Amount]        INT            NOT NULL,
    PRIMARY KEY CLUSTERED ([Bill_ID] ASC)
);
```

Figure 4. 100 Vehicle Renting Bills Details Table SQL query

Column Name	Data Type	Allow Nulls
Bill_ID	int	<input type="checkbox"/>
Driver_Name	varchar(50)	<input type="checkbox"/>
Date	varchar(50)	<input type="checkbox"/>
Bill_Amount	int	<input type="checkbox"/> <input type="checkbox"/>

Figure 4. 101 Vehicle Renting Bills Details Table SQL table design

4.2.7 Vehicle One Day Hire Bills Details Table

```
CREATE TABLE [dbo].[Bill_OneD_Hire_Table] (
    [Bill_ID]           INT            IDENTITY (1, 1) NOT NULL,
    [Driver_Name]       VARCHAR (50)   NOT NULL,
    [Date]              VARCHAR(50)    NOT NULL,
    [Bill_Amount]        INT            NOT NULL,
    PRIMARY KEY CLUSTERED ([Bill_ID] ASC)
);
```

Figure 4. 102 Vehicle One Day Hire Bills Details Table SQL query

Column Name	Data Type	Allow Null
Bill_ID	int	<input type="checkbox"/>
Driver_Name	varchar(50)	<input type="checkbox"/>
Date	varchar(50)	<input type="checkbox"/>
Bill_Amount	int	<input type="checkbox"/>
		<input type="checkbox"/>

Figure 4. 103 Vehicle One Day Hire Bills Details Table SQL table design

4.2.8 Vehicle Long Tour Hire Bills Details Table

```
CREATE TABLE [dbo].[Bill_LongD_Hire_Table] (
    [Bill_ID]           INT            IDENTITY (1, 1) NOT NULL,
    [Driver_Name]       VARCHAR (50)   NOT NULL,
    [Date]              VARCHAR(50)    NOT NULL,
    [Bill_Amount]        INT            NOT NULL,
    PRIMARY KEY CLUSTERED ([Bill_ID] ASC)
);
```

Figure 4. 104 Vehicle Long Tour Hire Bills Details Table SQL query

Column Name	Data Type	Allow Nulls
Bill_ID	int	<input type="checkbox"/>
Driver_Name	varchar(50)	<input type="checkbox"/>
Date	varchar(50)	<input type="checkbox"/>
Bill_Amount	int	<input type="checkbox"/> <input type="checkbox"/>

Figure 4. 105 Vehicle Long Tour Hire Bills Details Table SQL table design

4.2.9 SQL Database connecting Visual Studio test connection

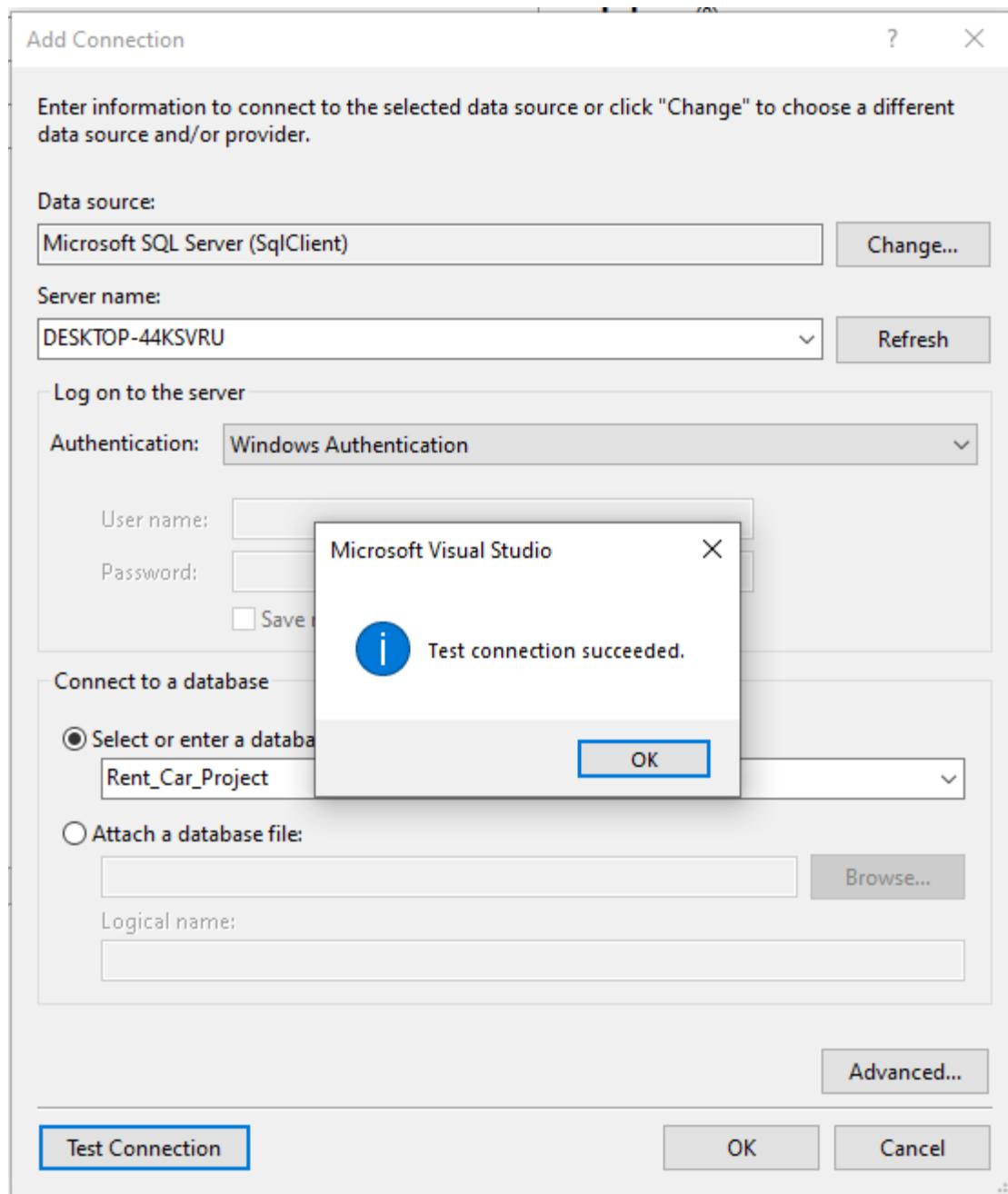


Figure 4. 106 SQL Database connecting Visual Studio test connection

4.3 Debugging

4.3.1 What is debugging in Visual studio IDE

Despite our best efforts, the code we write as software engineers does not always perform as predicted. It can also do something entirely unexpected! When this happens, the next step is to figure out why, and while we may be tempted to just stare at our code for hours. But by using a debugging tool or debugger is helping to resolve those problems far easier and more efficient.

Unfortunately, a debugger will not suddenly expose all of the issues or "bugs" in our code. Debugging is the process of running the code in a debugging tool, such as Visual Studio, step by step to determine where programmers committed a programming error. So, programmers can then see what modifications they need to make to the code, and debugging tools frequently allow them to make temporary changes while the program is still running. Effectively using a debugger is a skill that takes time and experience to master, but it is a must-have for any software engineer.

4.3.2 Why do we need Debugging?

An exception, often known as an error, is an unexpected event that occurs while running code. A debugging tool can lead programmers to the specific spot in the code where the error occurred and help them in coming up with solutions. If code displays some text but the text is incorrect, programmers know that either the data is incorrect or the code that sets the display text has a bug. They can analyze each and every modification to variables by stepping through the code in a debugger to see when and how wrong values are assigned.

When we start an app normally, we only see bugs after the code has completed. A program may also suddenly end without informing us of the reason. When we run an app in a debugger, also known as debugging mode, the debugger actively monitors everything that happens while the program is running. It also enables us to pause the app at any time to inspect its current state, and then move through our code line by line to inspect every detail as it occurs.

4.3.3 Debugging process in Visual studio IDE

We can enter debugging mode in Visual Studio by using following methods.

- F5 key in the keyboard
- Debug tab → Start Debugging menu command
- Start Debugging button in quick buttons bar

If an exception occurs, the Exception Helper in Visual Studio takes us to the exact location where the error happened and provides more details.

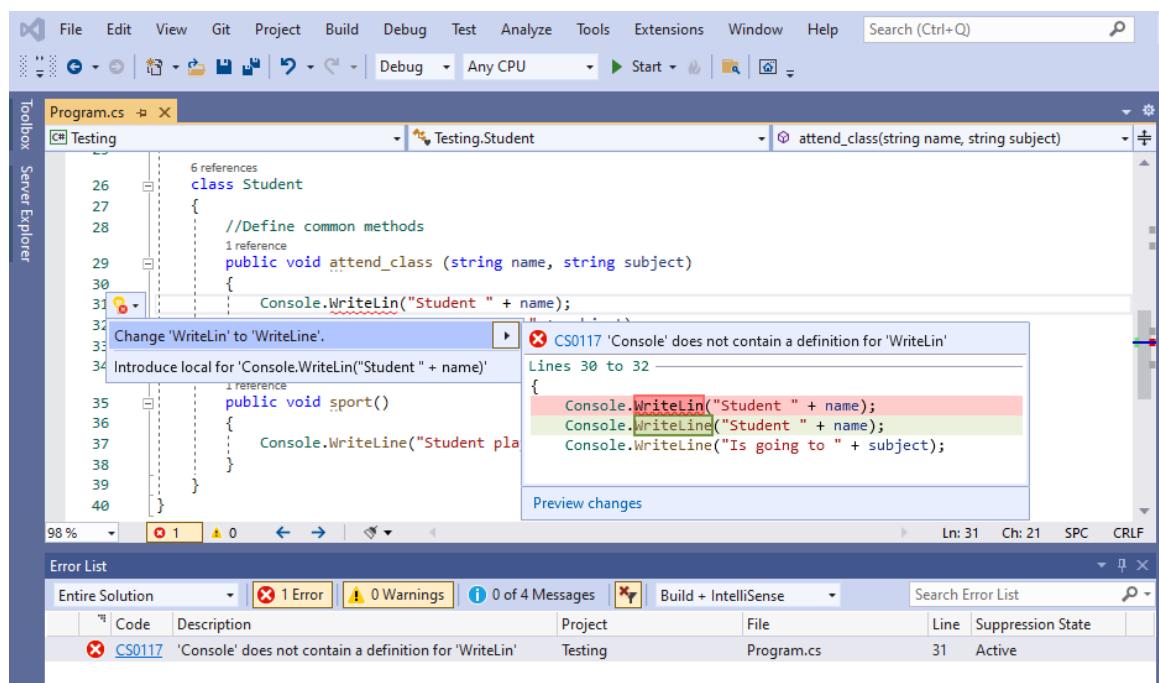


Figure 4. 107 Debugging flaws of the code through Visual Studio IDE

If we didn't get an exception, we should be able to figure out where the flaw is in our code. This is where we use the debugger's breakpoints to give us a chance to look over our code more closely. Breakpoints are the most fundamental and important aspect of a reliable debugging system. A breakpoint is a place in the running code where Visual Studio should pause it so we may examine the values of variables, memory behavior, or the order in which code runs. By clicking in the left margin next to a line of code in Visual Studio, we can immediately set a breakpoint. Alternatively, position mouse cursor on a line and hit F9.

4.3.4 Steps involved in Debugging process



Figure 4. 108 Steps involved in Debugging process

Step 01 : Identify the error →

A faulty error identification can lead to a waste development time. It is common for users to report production mistakes that are difficult to interpret, and the information we receive can occasionally be misleading. It's crucial to figure out what's wrong.

Step 02 : Find the error location →

We need to walk through the code to pinpoint the exact area where the issue is found after accurately detecting the error. At this step, we should concentrate on locating the issue rather than analyzing and understanding it.

Step 03 : Analyze the error →

In the third step, we must evaluate the code using a bottom-up technique starting from the error location. This will support us in understanding the error. Analyzing a bug has two basic goals: finding more faults around the error and ensuring that there are no risks of introducing collateral damage into the solution.

Step 04 : Prove the Analysis→

We need to identify a few more mistakes that may emerge on the program once we've finished examining the original bug. With the use of a test framework, this step involves building automated tests for these areas.

Step 05 : Cover Lateral Damage→

We must build or gather all unit tests for the code where we will make modifications at this step. These unit tests should now all pass if we run them.

Step 06 : Fix and Validate→

The final step is to correct all of the mistakes and run all of the test scripts to ensure that they all succeed.

4.3.5 Strategies to follow when debugging

- To fully comprehend the system, it is necessary to examine it in depth. It supports the debugger in constructing various models of the systems to be debugged.
- Backward analysis of the problem tracks the program backward from the place of the error indication to find the defective code portion. To determine the cause of flaws, we must thoroughly investigate the defect area.
- Forward analysis of a program consists of using breakpoints or print statements at various places throughout the program to track it forward. It's critical to focus on the area where incorrect results are obtained.
- To check for similar problems, we must recall previous experience. The debugger's skill is critical to the success of this strategy.

4.4 How I used the debugging process to develop more secure, robust application with examples

4.4.1 How I used the debugging process

I've built a small program to show the smallest 3 numbers of a list. As in below figure code 1st line defines the list of integers starting with number 1 and ending with number 6. Then in the 2nd line user defined method Get Smallest Number method is being called to catch 3 smallest numbers of the list. Then in the 3rd line is created to iterate these 3 numbers and display 1 by 1 in the console.

But and you can see in the output in the console, this is not the result we've been expected. This is totally opposed of the expected result. This is not the list of smallest numbers. Hence, we have to go deep of to the code first.

```

0 references
class Program
{
    0 references
    public static void Main(string[] args)
    {
        var numbers = new List<int> { 1, 2, 3, 4, 5, 6 };
        var smallest = GetSmallNumbers(numbers, 3);

        foreach (var number in smallest)
            Console.WriteLine(number);
        Console.ReadLine();
    }
}

1 reference
public static List<int> GetSmallNumbers(List<int> list, int count)
{
    var smallests = new List<int>();

    while (smallests.Count < count)
    {
        var min = GetSmallestNum(list);
        smallests.Add(min);
        list.Remove(min);
    }

    return smallests;
}

1 reference
public static int GetSmallestNum(List<int> list)
{
    var min = list[0];
    for (var i = 1; i < list.Count; i++)
    {
        if (list[i] > min)
            min = list[i];
    }
    return min;
}
}

```

C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe

6
5
4

Figure 4. 109 C# e code with the having unexpected Console Output.

When we debug the program usually, we do is put 1 or more breakpoints in the program then run the application in debug mode. When we run the application in debug mode, the execution stops at the breakpoints. And there we can inspect the values of different variables to make sure they have right expected value. If not, we know there is a problem with the code that need to be change.

In the figure below there is a breakpoint at the beginning of the code. Breakpoint is represented with a red circle at the left margin. By pressing F9 key, we can insert a breakpoint. And now we can run the application in debug mode with the help of the key F5. Eventually the program execution is stopped at the breakpoint line.

The screenshot shows a code editor window with the following code:

```

9 0 references
10 class Program
11 {
12     0 references
13     public static void Main(string[] args)
14     {
15         var numbers = new List<int> { 1, 2, 3, 4, 5, 6 };
16         var smallest = GetSmallNumbers(numbers, 3);
17
18         foreach (var number in smallest)
19             Console.WriteLine(number);
20
21     }

```

A red circle (breakpoint) is visible at the start of line 13. A yellow arrow is positioned above line 13, indicating the current line of execution.

Figure 4. 110 Breakpoint at the 1st line of the code

At this point I can look values of different variables, or I can continue the execution. There are few ways to continue the execution. One of common way is by “step over” with the help of the key F10. This step over is doing is executing 1 line at a time. The yellow arrow at the left margin is representing step over line.

When I move mouse curser on variable of executed lines by step over, we can see the variables stored values at that time. As in below figure the values stored in “smallest” variable is 6,5,4 which is totally different of expected values.

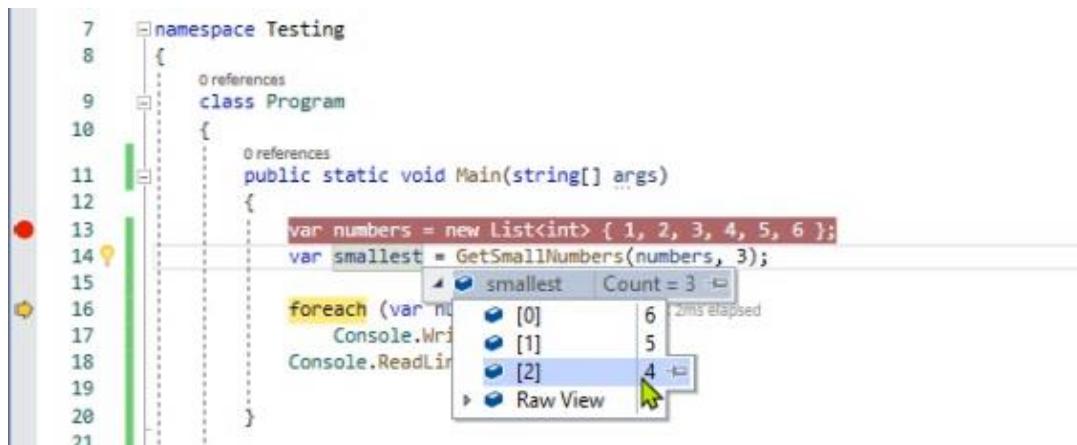


Figure 4. 111 Smallest variable stored values

Then I want to know what is happening in the Get Small Number method. For that I'm going to use debugging tool “step into” with the help of the key F11. It will step into the method Get Small Number. The reason I added a breakpoint here because I don't have to use F10 over and over again to come here. Next time when I run the application in debug mode I can jump here straightway.

When we look into min variable the stored value is 6 which is not expected value. When we look into list, we can see the entire list we've given which is ok here. So, we can determine the problem is with the Get Smallest Num method. So we can step into that method by F11 key and investigate it.



Figure 4. 112 min variable stored values

The screenshot shows a Visual Studio code editor with C# code. A tooltip is displayed over the line of code `var min = GetSmallestNum(list);`. The tooltip shows a table with the following data:

	list	Count = 6
[0]	1	
[1]	2	
[2]	3	
[3]	4	
[4]	5	
[5]	6	
Raw View		

Figure 4. 113 List variable stored values

Then this is a very useful tool for debugger. It can be reach by Debug Tab → Windows → Watch → Watch1.

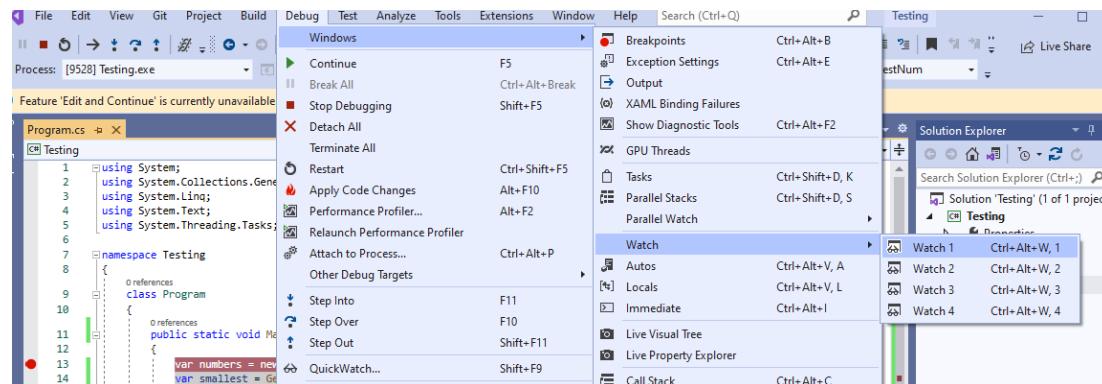


Figure 4. 114 Watch 1 windows tab open by Visual Studio IDE

The use of this is we can put our interested variables into this windows tab and can monitor the values of them. Hence, we don't have to hover our mouse over every variable we want to look up again and again.

```

1 reference
public static List<int> GetSmallNumbers(List<int> list, int count)
{
    var smallests = new List<int>();

    while (smallests.Count < count)
    {
        var min = GetSmallestNum(list);
        smallests.Add(min);
        list.Remove(min);
    }

    return smallests;
}

1 reference
public static int GetSmallestNum(List<int> list)
{
    var min = list[0];
    for (var i = 1; i < list.Count; i++)
    {
        if (list[i] > min) s im elapsed
            min = list[i];
    }
    return min;
}

```

Watch 1

Name	Value	Type
min	1	int
list[0]	1	int
list[i]	2	int

Figure 4. 115 Monitor interested variable values by watch tab.

According to above figure the value of min is 2. But at the execution at the line 41 the value of the list[i] in watch window is 2. The current value of the “i” is 2. We are comparing this with min variable. We are saying here if the list of “i” is greater than min, set min in the list of “i”. This proves that we are looking for the larger number here. But the minimum. So, this is the part where the bug has. After fixing this as below figure we can “step out” from the method by shortcut key Shift + F11.

```

1 reference
public static int GetSmallestNum(List<int> list)
{
    var min = list[0];
    for (var i = 1; i < list.Count; i++)
    {
        if (list[i] > min)
            min = list[i];
    }
    return min;
}

1 reference
public static int GetSmallestNum(List<int> list)
{
    var min = list[0];
    for (var i = 1; i < list.Count; i++)
    {
        if (list[i] < min)
            min = list[i];
    }
    return min;
}

```

Figure 4. 116 Fixing the bug

Now when we look into the min variable value, it is as we expected as in the below figure.

The screenshot shows a code editor with a debugger sidebar on the left. The code is a C# program that finds the three smallest numbers from a list. A red dot on the left indicates a breakpoint at line 13. A tooltip over the variable 'min' at line 28 shows its value as '1'. The code includes a reference to 'GetSmallestNum'.

```
11 public static void Main(string[] args)
12 {
13     var numbers = new List<int> { 1, 2, 3, 4, 5, 6 };
14     var smallest = GetSmallNumbers(numbers, 3);
15
16     foreach (var number in smallest)
17         Console.WriteLine(number);
18     Console.ReadLine();
19 }
20
21 //reference
22 public static List<int> GetSmallNumbers(List<int> list, int count)
23 {
24     var smallests = new List<int>();
25
26     while (smallests.Count < count)
27     {
28         var min = GetSmallestNum(list);
29         smallests.Add(min);
30         list.Remove(min);
31     }
32
33     return smallests;
34 }
```

Figure 4. 117 min variable value after debugging

So now we can stop the debugger and we can run the application with normal mode without debugging to see if we get the right results in the console. To stop the debugging, we can use the key F5.

The screenshot shows a terminal window titled 'C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe'. The output shows the numbers 1, 2, and 3, which are the expected results of the program.

```
1
2
3
```

Figure 4. 118 Expected output after debugging

4.4.2 How I handled some little bugs of the program easily

Visual Studio C# throws an exception when something goes wrong. The exception basically informs us that something went wrong in the code that C# couldn't manage. When this happens, the software usually crashes. When I spend too much time on the code, I don't always want an exception to crash the program. I handle exceptions so that the application's usual flow can be maintained even when runtime issues occur.

```
0 references
static void Main(string[] args)
{
    Console.WriteLine("Enter a number: ");
    int num1 = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("Enter a number: ");
    int num2 = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine(num1 / num2);
    Console.ReadLine();
}

C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe
Enter a number:
5
Enter a number:
0
```

Figure 4. 119 Basic calculator example code with the given parameters for user inputs.

```
0 references
static void Main(string[] args)
{
    Console.WriteLine("Enter a number: ");
    int num1 = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine("Enter a number: ");
    int num2 = Convert.ToInt32(Console.ReadLine());

    Console.WriteLine(num1 / num2); ✘
    Console.ReadLine();
}
```

Exception Unhandled

System.DivideByZeroException: 'Attempted to divide by zero.'

[View Details](#) | [Copy Details](#) | [Start Live Share session...](#)

[Exception Settings](#)

Figure 4. 120 Exception occur when run the program

As in above figure this error occurs because simple mathematics, we can't divide an integer with zero. It's impossible for us to and to computer to do this math.

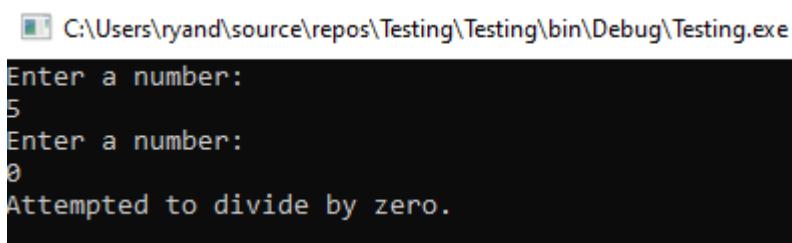
We don't want the program to crash while dealing with exceptions. This procedure detects exceptions and instructs them on how to manage them without crashing. To do this, we can utilize the try and catch blocks. The "try" and "catch" blocks work in the same way. We can put any code in the try block that we believe will break the application. And if that code crashes the program, it enters the catch block and executes all of the code contained inside. As in below figure we can throw exception in the console without crashing the program.

```
0 references
static void Main(string[] args)
{
    try
    {
        Console.WriteLine("Enter a number: ");
        int num1 = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine("Enter a number: ");
        int num2 = Convert.ToInt32(Console.ReadLine());

        Console.WriteLine(num1 / num2);
    }
    catch (Exception e)
    {
        Console.WriteLine(e.Message);
    }

    Console.ReadLine();
}
```



```
C:\Users\ryand\source\repos\Testing\Testing\bin\Debug\Testing.exe
Enter a number:
5
Enter a number:
0
Attempted to divide by zero.
```

Figure 4. 121 Handle exceptions by try and catch blocks

4.5 Coding Standards

4.5.1 Importance of coding standards

Software developers must follow to coding standards for a variety of reasons. The maintenance of software accounts for 40% to 80% of the entire cost. The original writer of software usually never completely supports it. Coding standards increase software readability by making it easier for developers to understand new code.

4.5.2 Coding standards to follow and how I followed them

- 1) The code should be simple to understand.

As a result, the developer should try to separate code blocks into paragraphs to define different areas of the code. Indentation should be used to identify the start and end of control structures, as well as a clear indication of where the code is between them.

```
private void btn_1D_Hire_Click(object sender, EventArgs e)
{
    _1D_Hire_Form obj = new _1D_Hire_Form(); // Create an object for Hire Form
    obj.Show(); // Show the object (Hire Form)
    this.Hide(); // Current Form (Rent Form) Hide
}

private void btn_Tour_Hire_Click(object sender, EventArgs e)
{
    Long_Hire_Form obj = new Long_Hire_Form();
    obj.Show();
    this.Hide();
}

private void btn_MyProfile_Click(object sender, EventArgs e)
{
    My_Profile_Form obj = new My_Profile_Form();
    obj.Show();
    this.Hide();
}
```

Figure 4. 122 Using simple code to understand – Part of the Ayubo System

2) The naming convention for variables in the code should be consistent throughout.

Furthermore, the data in the code should be specified.

There are two popular naming conventions being used.

- I. Camel Case – The first letter of each word is capital except the first word
- II. Underscore – Using underscore between words.

In the following figure red circled parts indicate where I've used Underscore naming convention for variables. And green arrow parts indicate where I've used Camel Case naming convention for Methods.

```

Program.cs    Rent_Form.cs    Rent_Form.cs [Design]    Long_Hire_Form.cs [Design]    1D_Hire_Form.cs [Design]
Rent Car      Rent_Car.Rent_Form
111
112
113
114
115
116
117     int day_rent;
118     int week_rent;
119     int month_rent;
120     int driver_charge;
121
122     SqlCommand cmd = con.CreateCommand();
123     cmd.CommandType = CommandType.Text;
124     cmd.CommandText = "SELECT * FROM Details_Renting_Table WHERE Rent_ID = '" + ID + "' ";
125
126     SqlDataReader DR1 = cmd.ExecuteReader();
127
128     if (DR1.Read())
129     {
130         txt_rentID.Text = DR1.GetValue(0).ToString();
131         txt_type.Text = DR1.GetValue(1).ToString();
132         txt_brand.Text = DR1.GetValue(2).ToString();
133         txt_model.Text = DR1.GetValue(3).ToString();
134         txt_color.Text = DR1.GetValue(4).ToString();
135         day_rent = Convert.ToInt32(txt_daily_rent.Text = DR1.GetValue(5).ToString()); //should Covert to int, becz we calculate them when adding the bill
136         week_rent = Convert.ToInt32(txt_weekly_rent.Text = DR1.GetValue(6).ToString()); //should Covert to int, becz we calculate them when adding the bill
137         month_rent = Convert.ToInt32(txt_monthly_rent.Text = DR1.GetValue(7).ToString()); //should Covert to int, becz we calculate them when adding the bill
138         driver_charge = Convert.ToInt32(txt_driver_rate.Text = DR1.GetValue(8).ToString()); //should Covert to int, becz we calculate them when adding the bill
139
140     }
141     DR1.Close();
142     con.Close();
143
144
145
146

```

Figure 4. 123 Using consistency of naming convention of the variables – Part of the Ayubo System

3) Should give the functions/methods names based on what they do.

```
Program.cs Rent_Form.cs [Design] Long_Hire_Form.cs [Design] 1D_Hire_Form.cs [Design]
Rent Car Rent_Car.Rent_Form

70
71
72 //***** OTHER METHODS *****
73
74 public void clear() //Clear Method
75 {
76     txt_type.Text = "";
77     txt_brand.Text = "";
78     txt_model.Text = "";
79     txt_colour.Text = "";
80     txt_daily_rent.Text = "";
81     txt_weekly_rent.Text = "";
82     txt_monthly_rent.Text = "";
83     cmb_refresh_category.Text = "";
84
85     dtp_start.Value = DateTime.Now;
86     dtp_end.Value = DateTime.Now;
87     txt_calculate.Text = "";
88     rbtn_yes.Checked = false;
89     rbtn_no.Checked = false;
90
91 }
92
93
94 public void display_data_grid_view() //For the data grid view //Data Grid View Method
95 {
96     try
97     {
98         dt = new DataTable();
99         con.Open();
100        adpt = new SqlDataAdapter("SELECT * FROM Details_Renting_Table", con);
101        adpt.Fill(dt);
102        dgv_Renting_List.DataSource = dt;
103        con.Close();
104    }
105    catch (Exception ex)
106    {
107        MessageBox.Show(ex.Message);
108        con.Close();
109    }
110 }
```

Figure 4. 124 Naming the function according to what they perform – Part of the Ayubo System

- 4) Use a specific way for making comments on the project. As a result, the developer may be capable of understanding it even after a period of time has passed.

```

10  using System.Data.SqlClient;
11
12  namespace Rent_Car
13  {
14      public partial class Dashboard : Form
15      {
16          public Dashboard()
17          {
18              InitializeComponent();
19
20
21
22          //***** Filling Combo Boxes
23          fill_combo_Rent(); //Filling combobox Rent
24          fill_combo_OneD(); //Filling combobox OneD
25          fill_combo_LongD(); //Filling combobox Tours
26
27
28          //***** Summery Panel Details
29          Get_Vehicle_Count(); //Vehicle count
30          Get_Rent_Count(); //Rent packages count
31          Get_OneD_Count(); //One Day packages count
32          Get_LongD_Count(); //Tour packages count
33          Get_Drivers_Count(); //Drivers count
34
35
36          //***** Financial Panel Details
37          Get_Rent_Total(); //Total Earn Amount by Renting
38          Get_Rent_Amount_By_Driver(); //Drivers's Earn Amount by Renting
39
40          Get_OneD_Total(); //Total Earn Amount by One Day Hires
41          Get_OneD_Amount_By_Driver(); //Drivers's Earn Amount by One Day Hiring
42
43          Get_LongD_Total(); //Total Earn Amount by Tour Hires
44          Get_LongD_Amount_By_Driver(); //Drivers's Earn Amount by Tour Hiring
45
46
47          //***** Award Panel Details
48          Best_Rent_Driver(); //Best Rent Driver Award
49          Best_OneD_Driver(); //Best One Day Hire Driver Award
50          Best_LongD_Driver(); //Best Tour Hire Driver Award
51      }
}

```

Figure 4. 125 Commenting on the lines of important parts of the project

4.5.3 Benefits of Implementing Coding Standards in Software Development

- 1) The probability of a project's failure is minimized.

IT projects frequently fail as a result of issues encountered while developing software. Many problems and the probability of project failure are reduced when code quality standards are followed.

- 2) Improve efficiency

It is frequently observed that software developers spend a lot of time on correcting code quality issues that could have been avoided. Coding standards and best practices would assist the team in detecting problems early on, if not totally preventing them. This will improve productivity throughout the software development process.

- 3) Cost efficiency

Developers can reuse code that is written in a clear manner whenever they need it. This can significantly minimize the cost of development as well as the time spent on it.

- 4) Easy to maintain

If coding standards are followed, the code is consistent and easy to maintain. This is due to the fact that anyone may comprehend it and change it at any time.

- 5) Making the code minimal complexity

Whenever a code is complex, it is more likely to be vulnerable to bugs. Coding standards help in the construction of less complex software programs, which reduces errors.

6) Rectification of bugs

If the source code is written consistently, it becomes much easier to find and fix flaws in the product.

Conclusion

This entire assignment is based on an implementation of a Car rent and hire system design for selected company (Ayubo Drive). The purpose of this assignment is to improve programing skills.

This report includes basic programming principles, creating algorithms, understanding time complexities, creating basic programs by using Python language and C# language, understanding different programming paradigms, working with IDEs, Basic system creation using Microsoft SQL and Visual Studio C# windows forms application, debugging process and coding standards.

References

Investopedia. 2022. Algorithm Definition.

[ONLINE] Available at: <https://www.investopedia.com/terms/a/algorithm.asp>.

[Accessed 03 February 2022].

Techopedia. 2022. What is an Algorithm? - Definition from Techopedia.

[ONLINE] Available at: <https://www.techopedia.com/definition/3739/algorithm>.

[Accessed 03 February 2022].

Code with C. 2022. Fibonacci Series Algorithm and Flowchart | Code with C.

[ONLINE] Available at: <https://www.codewithc.com/fibonacci-series-algorithm-flowchart/>.

[Accessed 03 February 2022].

GeeksforGeeks. 2022. Difference Between Algorithm and Flowchart - GeeksforGeeks.

[ONLINE] Available at: <https://www.geeksforgeeks.org/difference-between-algorithm-and-flowchart/>.

[Accessed 03 February 2022].

GeeksforGeeks. 2022. Introduction to Algorithms - GeeksforGeeks.

[ONLINE] Available at: <https://www.geeksforgeeks.org/introduction-to-algorithms/>.

[Accessed 03 February 2022].

Investopedia. 2022. Algorithm Definition.

[ONLINE] Available at: <https://www.investopedia.com/terms/a/algorithm.asp>.

[Accessed 03 February 2022].

What is an Algorithm?. 2022. What is an Algorithm?.

[ONLINE] Available at: <https://www.programiz.com/dsa/algorithm>.

[Accessed 03 February 2022].

Khan Academy. 2022. What is an algorithm and why should you care? (video) | Khan Academy.

[ONLINE] Available at: <https://www.khanacademy.org/computing/computer-science/algorithms/intro-to-algorithms/v/what-are-algorithms>.

[Accessed 03 February 2022].

Shlok Bhatt. 2022. Characteristics of an Algorithm. There are some characteristics which... | by Shlok Bhatt | Medium.

[ONLINE] Available at: <https://medium.com/@bhattshlok12/characteristics-of-an-algorithm-49cf4d7bcd9>.

[Accessed 03 February 2022].

atechdaily.com. 2022. No page title.

[ONLINE] Available at: <https://atechdaily.com/posts/algorithm-for-factorial-number>.

[Accessed 03 February 2022].

livescience.com. 2022. What is the Fibonacci sequence? | Live Science.

[ONLINE] Available at: <https://www.livescience.com/37470-fibonacci-sequence.html>.

[Accessed 03 February 2022].

Fibonacci Sequence. 2022. Fibonacci Sequence.

[ONLINE] Available at: <https://www.mathsisfun.com/numbers/fibonacci-sequence.html>.

[Accessed 03 February 2022].

Factorial Function !. 2022. Factorial Function !.

[ONLINE] Available at: <https://www.mathsisfun.com/numbers/factorial.html>.

[Accessed 03 February 2022].

Big-O Notation, Omega Notation and Big-O Notation (Asymptotic Analysis). 2022. Big-O Notation, Omega Notation and Big-O Notation (Asymptotic Analysis).

[ONLINE] Available at: <https://www.programiz.com/dsa/asymptotic-notations>.

[Accessed 03 February 2022].

Big-O Notation: A simple explanation with examples . 2022. Big-O Notation: A simple explanation with examples .

[ONLINE] Available at: <https://www.linkedin.com/pulse/big-o-notation-simple-explanation-examples-pamela-lovett>.

[Accessed 03 February 2022].

Developer Insider. 2022. Big-O Notation Explained with Examples.

[ONLINE] Available at: <https://developerinsider.co/big-o-notation-explained-with-examples/>.

[Accessed 03 February 2022].

Career Karma. 2022. Big O Notation - Time Complexity.

[ONLINE] Available at: <https://careerkarma.com/blog/big-o-notation-time/>.

[Accessed 03 February 2022].

Maxwell Harvey Croy. 2022. How To Calculate Time Complexity With Big O Notation | by Maxwell Harvey Croy | DataSeries | Medium.

[ONLINE] Available at: <https://medium.com/dataseries/how-to-calculate-time-complexity-with-big-o-notation-9afe33aa4c46>.

[Accessed 03 February 2022].

freeCodeCamp.org. 2022. What exactly is a programming paradigm?.

[ONLINE] Available at: <https://www.freecodecamp.org/news/what-exactly-is-a-programming-paradigm/>.

[Accessed 03 February 2022].

Learn Computer Science. 2022. What Is Programming Paradigm ? | Paradigm Types, Features Explained..

[ONLINE] Available at: <https://www.learncomputerscienceonline.com/programming-paradigm/>.

[Accessed 03 February 2022].

IONOS Digitalguide. 2022. Imperative programming: Advantages & disadvantages of the paradigm - IONOS.

[ONLINE] Available at: <https://www.ionos.com/digitalguide/websites/web-development/imperative-programming/>.

[Accessed 03 February 2022].

GeeksforGeeks. 2022. Introduction of Programming Paradigms - GeeksforGeeks.

[ONLINE] Available at: <https://www.geeksforgeeks.org/introduction-of-programming-paradigms/>.

[Accessed 03 February 2022].

Hackr.io. 2022. What is Procedural Programming? [Definition] - Key Features.

[ONLINE] Available at: <https://hackr.io/blog/procedural-programming>.

[Accessed 03 February 2022].

Object oriented vs procedural vs event driven programming - ppt download. 2022. Object oriented vs procedural vs event driven programming - ppt download.

[ONLINE] Available at: <https://slideplayer.com/slide/16031203/>.

[Accessed 03 February 2022].

cwells. 2022. Event-driven Programming .

[ONLINE] Available at: <https://www.technologyuk.net/computing/software-development/software-design/event-driven-programming.shtml>.

[Accessed 03 February 2022].

Techopedia. 2022. What is an Event-Driven Program? - Definition from Techopedia.

[ONLINE] Available at: <https://www.techopedia.com/definition/7083/event-driven-program>.

[Accessed 03 February 2022].

Bench Partner. 2022. Advantages and Disadvantages of Event-Driven Programming - Bench Partner.

[ONLINE] Available at: <https://benchpartner.com/advantages-and-disadvantages-of-event-driven-programming>.

[Accessed 03 February 2022].

Characteristics of Object Oriented programming language - oops. 2022. Characteristics of Object Oriented programming language - oops.

[ONLINE] Available at: <https://www.careerride.com/oops-characteristics.aspx>.

[Accessed 03 February 2022].

Event-Driven Programming Features. 2022. Event-Driven Programming Features.

[ONLINE] Available at: <https://www.ukessays.com/essays/computer-science/eventdriven-programming-features-6167.php>.

[Accessed 03 February 2022].

Edureka. 2022. What is Debugging? Different Stages of Debugging | Edureka.

[ONLINE] Available at: <https://www.edureka.co/blog/what-is-debugging/#:~:text=Debugging%20has%20many%20benefits%20such,structures%20and%20allows%20easy%20interpretation..>

[Accessed 03 February 2022].

Mikejo5000. 2022. Debugging code for absolute beginners - Visual Studio (Windows) | Microsoft Docs.

[ONLINE] Available at: <https://docs.microsoft.com/en-us/visualstudio/debugger/debugging-absolute-beginners?view=vs-2022&tabs=csharp>.

[Accessed 03 February 2022].

Multidots. 2022. Importance of Coding Standard and Code Quality in Software Development.

[ONLINE] Available at: <https://www.multidots.com/importance-of-code-quality-and-coding-standard-in-software-development/#:~:text=Coding%20standards%20help%20in%20the,reduce%20the%20errors.&text=If%20programming%20standards%20in%20software,at%20any%20point%20in%20time..>

[Accessed 03 February 2022].

Difference Between IDE and Code Editor | Difference Between. 2022. Difference Between IDE and Code Editor | Difference Between.

[ONLINE] Available at: <http://www.differencebetween.net/technology/difference-between-ide-and-code-editor/>.

[Accessed 03 February 2022].

Difference Between IDE and Text Editor | Difference Between. 2022. Difference Between IDE and Text Editor | Difference Between.

[ONLINE] Available at: <http://www.differencebetween.net/technology/difference-between-ide-and-text-editor/>.

[Accessed 03 February 2022].

Computer Programming. 2022. Computer Programming.

[ONLINE] Available at:

<https://homepage.cs.uri.edu/faculty/wolfe/book/Readings/Reading13.htm>.

[Accessed 03 February 2022].