શ્રી સ્વામિનારાયણ ગુરુકુલ રાજકોટ સંસ્થાન
શાસ્ત્રી સ્વામી શ્રી ધર્મજીવનદાસજી
સાયન્સ & IT ગુરુકુલ કોલેજ
ગુરુકુલ કેમ્પસ, કોલેજ રોડ, જૂનાગઢ

Patel Drinking Water

Project Partners:

MR. VIRAJ GONDALIYA, B.Sc.I.T-5TH

MR. KASHYAP DABHI, B.SC.I.T-5TH

MR. VINEX SURELIYA, B.Sc.I.T-5TH

MR. DIPAK SOLANKI, B.Sc.I.T-5TH

:: submitted to ::
BKNM University, Junagadh

:: GUIDED BY::

Mr. Ripal Pandya

Mr. Gaurav Raval

Mr. Milind Anandpara

શ્રી સ્વામિનારાયણ ગુરુકુલ રાજકોટ સંસ્થાન

શાસ્ત્રી સ્વામી શ્રી ધર્મજીવનદાસજી

સાયન્સ & IT ગુરુકુલ કોલેજ

ગુરુકુલ કેમ્પસ, કોલેજ રોડ, જૂનાગઢ

(Affiliated to **Bhakta Kavi Narsinh Mehta University, Junagadh**)

Project Completion Certificate

This certificate is awarded to

Solanki Dipak Rameshbhai	**B.Sc.I.T-2023**
Gondaliya Viraj Rajeshbhai	**B.Sc.I.T-2023**

in completion of project work

C#	**07-07-2023**
SQL server	**07-10-2023**

Mr. Ripal V. Pandya

Mr. Gaurav J. Raval

Project Guide

Director

www.sssdiit.junagadhgurukul.org

શ્રી સ્વામિનારાયણ ગુરુકુલ રાજકોટ સંસ્થાન

શાસ્ત્રી સ્વામી શ્રી ધર્મજીવનદાસજી

સાયન્સ & IT ગુરુકુલ કોલેજ

ગુરુકુલ કેમ્પસ, કોલેજ રોડ, જૂનાગઢ

(Affiliated to **Bhakta Kavi Narsinh Mehta University, Junagadh**)

Project Completion Certificate

This certificate is awarded to

Sureliya Vinex Dineshbhai	**B.Sc.I.T-2023**
Dabhi Kashyap Nileshbhai	**B.Sc.I.T-2023**

in completion of project work

C#	**07-07-2023**
SQL server	**07-10-2023**

Mr. Ripal V. Pandya

Mr. Gaurav J. Raval

Project Guide

Director

www.sssdiit.junagadhgurukul.org

A

PROJECT REPORT ON

Patel Drinking Water

Submitted in Fulfillment of Requirements

For Completion of Semester - 5 in

Bachelor of Science in Information Technology

Year 2023

To

SHASHTRI SWAMI SHREE DHARMAJIVANDASJI INSTITUTE OF INFORMATION TECHNOLOGY

JUNAGADH

Guided By:

Mr. Gaurav Raval

Mr. Ripal Pandya

Mr. Milind Anandpara

Prepared By:

Mr. Viraj R. Gondaliya

Mr. Kashyap N. Dabhi

Mr. Vinex D. Sureliya

Mr. Dipak R. Solanki

PREFACE

Welcome to the world of "Patel Drinking Water," a cutting-edge software solution developed to streamline and enhance the management of drinking water distribution systems. This project represents the culmination of our dedication to providing efficient, sustainable, and reliable access to clean drinking water for communities everywhere.

Background:-

Access to safe and clean drinking water is a fundamental human right, and yet, many communities around the world continue to face challenges in ensuring its availability. "Patel Drinking Water" emerged from the collective vision of our team to contribute to the global effort of providing safe drinking water to all.

The Purpose:-

This C# software solution is designed to simplify the complex tasks associated with managing water distribution networks, from monitoring water quality to optimizing distribution routes. Whether you are a utility company, a municipal authority, or an organization dedicated to water resource management, "Patel Drinking Water" aims to empower you with the tools you need to make informed decisions, reduce waste, and improve service delivery.

Key Features:-

- Real-time Monitoring: Gain insights into water quality, pressure, and flow rates.
- Route Optimization: Efficiently plan and manage water distribution routes.
- Data Analytics: Utilize data-driven insights for informed decision-making.

- Sustainability: Promote responsible water resource management practices.
- User-Friendly Interface: Intuitive design for easy navigation and usability.

Getting Started:-

This documentation will guide you through the installation, setup, and utilization of "Patel Drinking Water." Whether you are a software developer, a water utility professional, or a concerned citizen, we encourage you to explore the software and contribute to our shared mission of ensuring access to safe drinking water for all.

Acknowledgments:-

We would like to express our gratitude to all the individuals, organizations, and communities who have inspired and supported the development of "Patel Drinking Water." Your dedication to the cause of clean water access has been our driving force, and we are committed to continuing our efforts in this noble endeavour.

Thank you for choosing "Patel Drinking Water."

Mr. Viraj Gondaliya
Mr. Kashyap Dabhi
Mr. Vinex Sureliya
Mr. Dipak Solanki

ACKNOWLEDGEMENT

We are very thankful to all whose have helped in preparing this project. We are feeling a great happiness to present this website project. First of all we would like to thank "BKNM University" who give me an opportunity to give a chance to prepare a project.

Before we get in to thick of the things we would to add a few heartfelt words for the people who were part of this project numerous ways, people who give unending support right from the stage project ideas was conceived. In particular we would like to thank Mr. Ripal Pandya, Mr. Gaurav Raval, Mr. Milind Anandpara (Project Guide), who has always inspired us and has directed us towards the successful completion of our project. They have been the guided through the project and their encouragement has left me indebted to them.

We are very thankful to the Director Shadhu RushikeshdashjiSwami and the Asst. Director Mr. Rajesh Bharad of Shastri Swami Shree Dharmajivandasji Institute of Information Technology – Junagadh.

We are also thankful to our classmate and few other people who helped us directly or indirectly in solving problem and in making our web development project more efficient and attractive.

Thank you…

Date:

Place: Junagadh

Mr. Viraj Gondaliya

Mr. Kashyap Dabhi

Mr. Vinex Sureliya

Mr. Dipak Solanki

I N D E X

PROJECT PROFILE

Project Title	Patel Drinking water
Project Description	It's enterprise software that can able to manage customer as well as employee, facilitated by billing and custom monthly selling reports and daily selling reports.
Front End	C#
Back End	SQL server
Other Tools	Cristal Report
Guide	Ripal V. Pandya Gaurav J. Raval Milind V. Anadpara
Submitted To	S.S.S.D.I.I.T College

USE OF SYSTEM DEVELOPMENT LIFE CYCLE MODEL

A software life cycle model (also termed process model) is a pictorial and diagrammatic representation of the software life cycle. A life cycle model represents all the methods required to make a software product transit through its life cycle stages. It also captures the structure in which these methods are to be undertaken.

In other words, a life cycle model maps the various activities performed on a software product from its inception to retirement. Different life cycle models may plan the necessary development activities to phases in different ways. Thus, no element which life cycle model is followed, the essential activities are contained in all life cycle models though the action may be carried out in distinct orders in different life cycle models. During any life cycle stage, more than one activity may also be carried out.

Need of SDLC

The development team must determine a suitable life cycle model for a particular plan and then observe to it.

Without using an exact life cycle model, the development of a software product would not be in a systematic and disciplined manner. When a team is developing a software product, there must be a clear understanding among team representative about when and what to do. Otherwise, it would point to chaos and project failure. This problem can be defined by using an example. Suppose a software development issue is divided into various parts and the parts are assigned to the team members. From then on, suppose the team representative is allowed the

freedom to develop the roles assigned to them in whatever way they like. It is possible that one representative might start writing the code for his part, another might choose to prepare the test documents first, and some other engineer might begin with the design phase of the roles assigned to him. This would be one of the perfect methods for project failure.

A software life cycle model describes entry and exit criteria for each phase. A phase can begin only if its stage-entry criteria have been fulfilled. So without a software life cycle model, the entry and exit criteria for a stage cannot be recognized. Without software life cycle models, it becomes tough for software project managers to monitor the progress of the project.

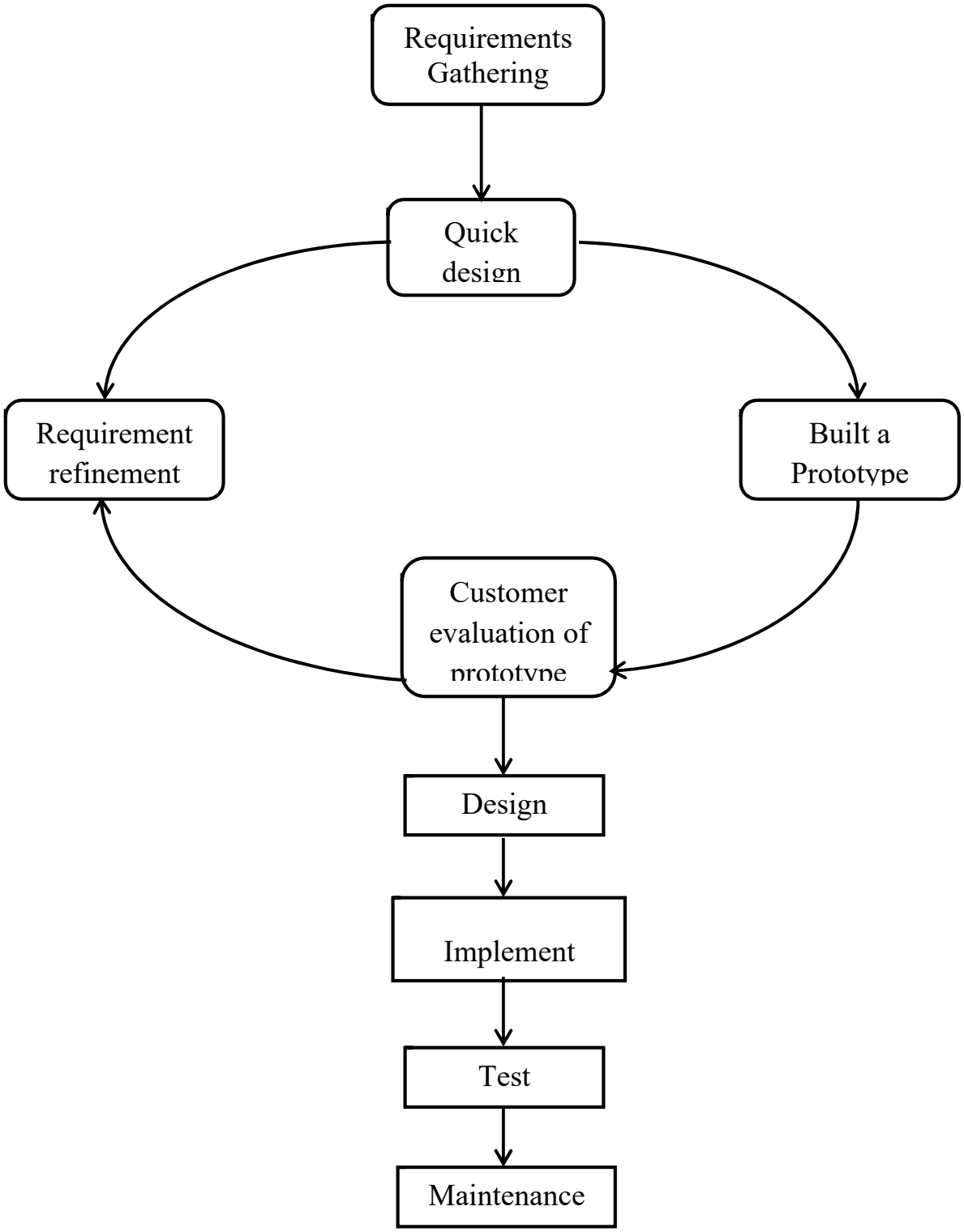
Following are the different Life Cycle Model example.

- Waterfall model
- Iterative waterfall model
- Prototyping model
- Evolutionary model
- Spiral model
- R.A.D. model (Rapid Application Development)

PROTOTYPE MODEL

The prototype model requires that before carrying out the development of actual software, a working prototype of the system should be built. A prototype is a toy implementation of the system. A prototype usually turns out to be a very crude version of the actual system, possible exhibiting limited functional capabilities, low reliability, and inefficient performance as compared to actual software. In many instances, the client only has a general view of what is expected from the software product. In such a scenario where there is an absence of detailed information regarding the input to the system, the processing needs, and the output requirement, the prototyping model may be employed.

PROTOTYPE MODEL DIAGRAM



Feasibility Study

What is feasibility study?

A feasibility study is a critical early phase in the project planning process. It involves a systematic examination of a proposed software project to determine whether it is technically, economically, and operationally viable. The primary goal of a feasibility study is to assess whether the project is worth pursuing and whether it can be completed successfully. Feasibility studies help organizations make informed decisions about whether to precede with a project, allocate resources, and set realistic expectations.

Types of feasibility study:-

- Technical Feasibility
- Operational Feasibility
- Economic Feasibility

1) **Technical Feasibility**:

Technical feasibility is an essential aspect of a feasibility study in various fields, including software engineering. It assesses whether a proposed project or initiative can be implemented from a technical standpoint. It evaluates the availability of the necessary technology, resources, and expertise to successfully complete the project. Here are some common types or aspects of technical feasibility:

➢ Hardware Requirement:-

Tools	Required
Processer	Intel i5 or higher
Hard disk	SSD 256 GB or higher
RAM	8 GB or higher
Other i/o device	Monitor/Printer

- **Software Requirement:-**

Software
Visual studio 2022
C#
MySQL

2) Operational Feasibility:-

Operational feasibility is a critical aspect of a feasibility study in project management and software engineering. It assesses whether a proposed project or system can be effectively integrated into an organization's existing operations and whether it can be operated and maintained efficiently once it is implemented.

- Admin can add & delete vehicle.
- Admin can select & display vehicle.
- Admin can add, edit & delete customer.
- Admin can view daily selling & monthly report.
- Admin can add customer payment.
- Admin can print customer bill.

3) Economic Feasibility:-

Economic feasibility, also known as cost-benefit analysis, is a crucial component of a feasibility study in project management and software engineering. It focuses on assessing whether a proposed project or system is financially viable and economically justifiable.

Our software total cost is ₹18000.

REQUIREMENT GATHERING

Requirements gathering are the process of identifying and defining what a software system needs to do to meet the needs of its users. It involves talking to stakeholders, analyzing user needs, and defining clear specifications that developers can use to build the system.

Here are the requirements of our client:-

Questionnaire:

Q.1 why you update current software?

- Some of the features are not working properly in current software.

Q.2 Do you need a login page for accessing software?

- Yes

Q.3 Do you need a payment system?

- Yes

Q.4 Do you need a show PDF of bill?

- Yes

Q.5 which kind of upgrades in your software?

- Manually change the data in whole software.

Q.6 which fields you want to change manually?

- Delete vehicle number
- Customer bills
- Add customer
- Payment system
- Daily sell

Q.7 which kind of change in monthly report?

- New format has adopted and some other column (paid amount) has removed.

REQUIREMENT ANALYSIS

Requirement analysis is significant and essential activity after elicitation. We analyze, refine, and scrutinize the gathered requirements to make consistent and unambiguous requirements. This activity reviews all requirements and may provide a graphical view of the entire system. After the completion of the analysis, it is expected that the understandability of the project may improve significantly. Here, we may also use the interaction with the customer to clarify points of confusion and to understand which requirements are more important than others.

Types of fact finding

- Interviews
- Questionnaires
- Record reviews
- Observation

From the about option, we have used the **questionnaire** and **personal interview** methods of the fact finding techniques. I have adopted Questionnaires because I can properly understand their needed of desktop app. By using Personal Interview I have understood the smallest need of their application and some idea of layout and designing

- Admin can add, delete, select & display vehicle.
- Admin can add, edit & delete customer.
- Admin can view daily selling & monthly report.
- Admin can add customer payment.
- Admin can print customer bill.

PROJECT ABSTRACTS

User Group:

- Admin

Admin:-

Admin have all rights of modify all data on whole software like, Insert-update-delete the user and vehicle record, View monthly report, Daily Selling, printing bill etc...

PROPOSED SYSTEM

The software has been developed with visual studio 2022 and framework-4.0 Collaborate with internal MySQL server is on the local machine software.

- Visual studio 2022 (window)
- C# (A programming language)
- MySQL management studio 19

Using of all is acceptable factors for the real implementation of software. Because in three are much number of advantage and simplicity and also security in the point of view for this software development task. It's enterprise software that can able to manage customer as well as employee, facilitated by billing and custom monthly selling reports and daily selling reports.

ADVANTAGES & LIMITATIONS OF PROPOSED SYSTEM

Advantages:-

- **Ease of Learning and Use:**

  C# was designed with developer productivity in mind. Its syntax is similar to other C-based languages like C++ and Java, making it relatively easy for developers with experience in those languages to pick up. Additionally, its extensive standard library (the .NET Framework) simplifies common programming tasks.

- **Strong Typing and Type Safety:**

  C# is a statically-typed language, which means that type checking is performed at compile-time. This helps catch many errors early in the development process, reducing the likelihood of runtime errors.

- **Object-Oriented:**

  C# is a fully object-oriented language, allowing developers to model real-world entities using classes and objects. This makes code more organized, maintainable, and extensible.

- **Platform Independence:**

  C# is not limited to a single platform. With .NET Core (now known as .NET 5 and later as .NET), you can write cross-platform applications that run on Windows, macOS, and Linux. This makes it

ideal for developing applications that need to work on different operating systems.

- **Rich Standard Library:**

C# benefits from the extensive .NET Framework (or .NET Core/.NET 5+ for cross-platform development) class libraries, which provide a wide range of built-in functionalities for various tasks like data access, networking, GUI development (Windows Forms, WPF), and more. This minimizes the need for developers to reinvent the wheel.

- **Performance:**

C# applications can be highly optimized, and the language itself is known for its good runtime performance. The Just-In-Time (JIT) compiler and other runtime optimizations contribute to this performance.

Limitation:-

- **Platform Dependency:**

  While C# has become more cross-platform with .NET Core and .NET 5+, it is still primarily associated with the Windows platform. Some Windows-specific features may not be available or may require workarounds when developing cross-platform applications.

- **Learning Curve for Beginners:**

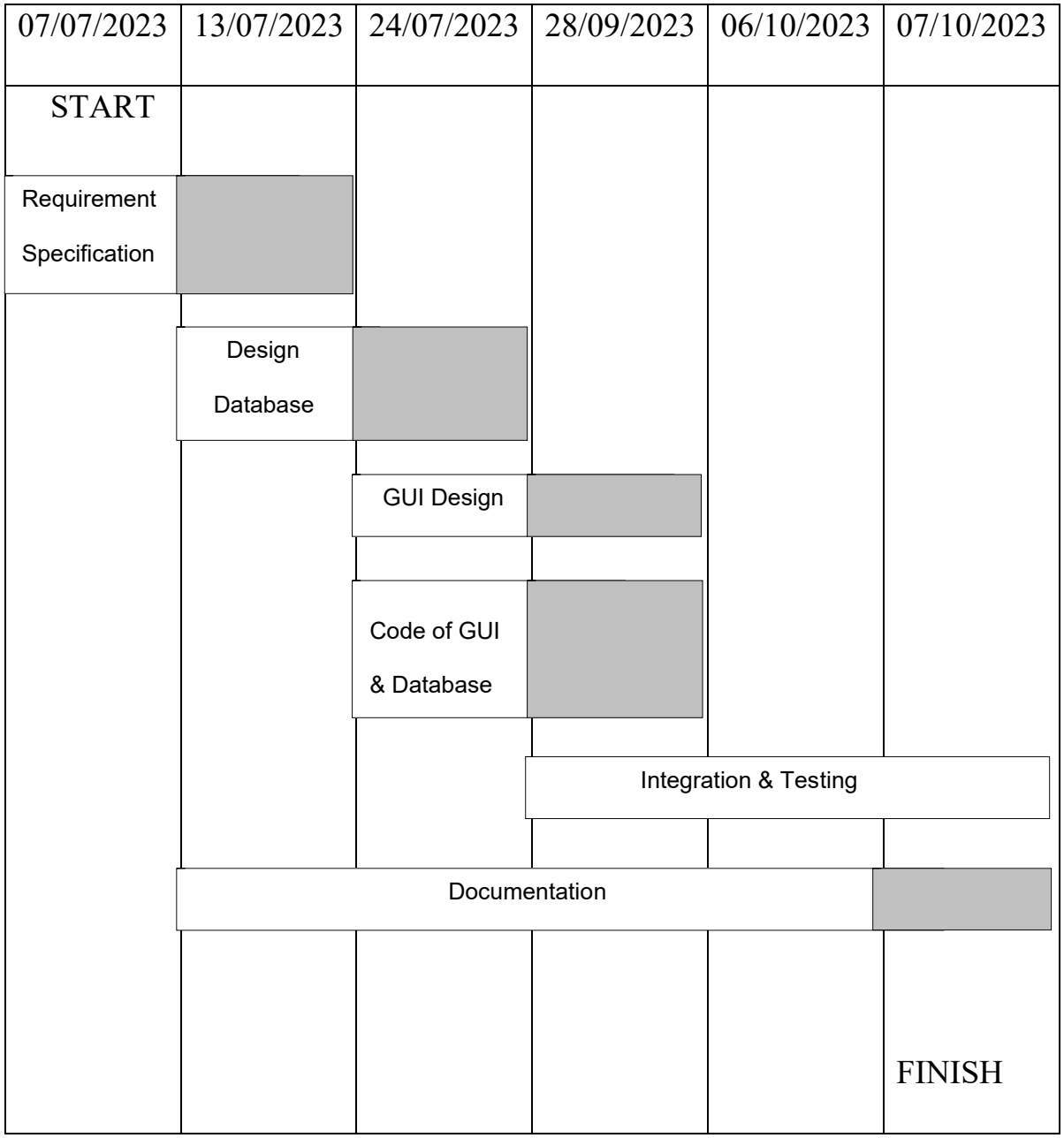
  While C# is generally considered easy to learn for experienced programmers, beginners may find it slightly more challenging due to the intricacies of object-oriented programming and the use of .NET Framework or .NET Core.

PERT CHART AND GANTT CHART

Gantt chart:-

The first Gantt chart was devised in the mid-1890s by Karol Adamiecki, a Polish engineer who ran a steelworks in southern Poland and had become interested in management ideas and techniques.

Gantt charts are useful for planning and scheduling projects. They help you assess how long a project should take, determine the resources needed, and plan the order in which you'll complete tasks. They're also helpful for managing the dependencies between tasks.



Advantages:-

- It is a tool for representing the project schedule.
- It is easy to create.
- It is a visual tool.

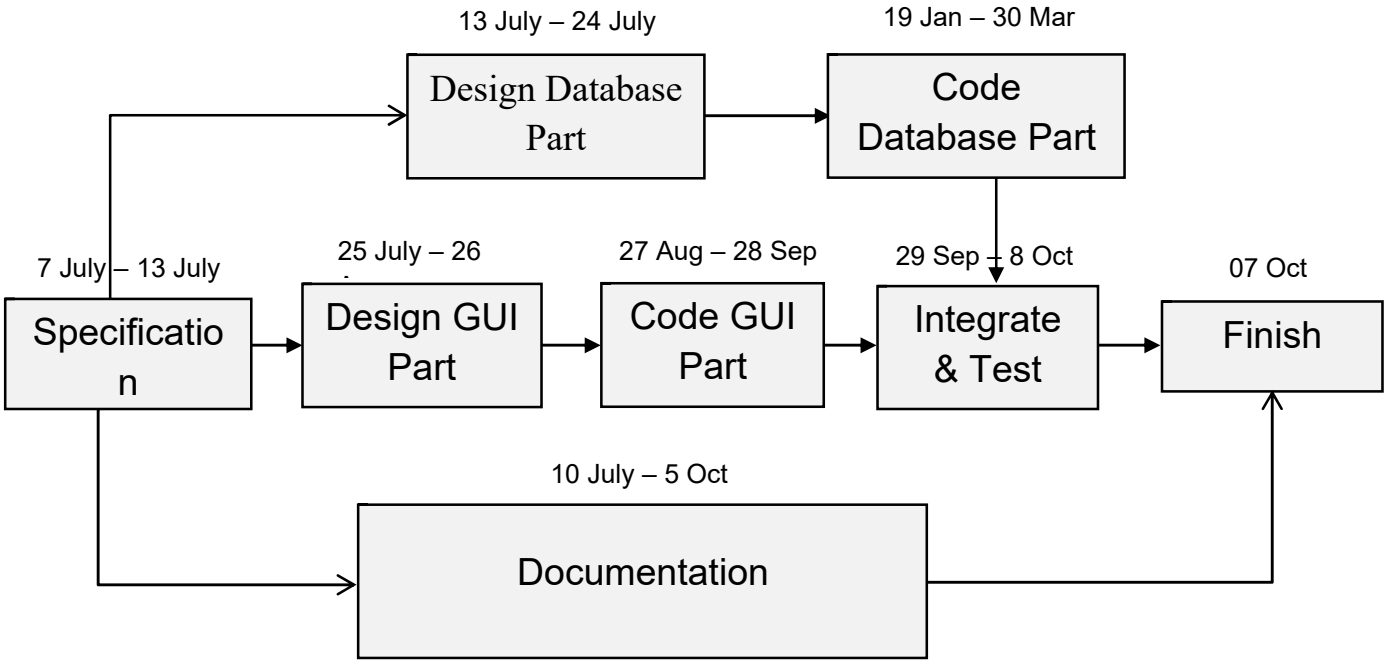
- It is easy for stakeholder's and project teams to understand the work schedule
- It enhances team productivity
- Different activity groups can be represented on the same schedule in one simple chart
- It improves communication and coordination among stakeholders in order to complete tasks as per the work schedule
- It supports procurement processes by showing delivery dates
- It helps to control the project status by comparing the planned dates shown on the chart
- It helps project and department managers to easily coordinate their teamwork
- It is a communication tool that improves stakeholder's decision-making.

Disadvantages:-

- Without software, it is hard to prepare and update the work schedule in a short period of time
- It is not easy to see all the activities in a complex project
- Inserting activities and establishing activity relationships may be time-consuming in large and complex projects
- Without software, it is not possible to assign resources to tasks
- Without software, it is not possible to make resource leveling
- It is difficult to see everything on one sheet of paper
- It is difficult to realign the activities from one WBS (Work Breakdown Structure) level to another

Pert chart:-

A program evaluation review technique (PERT) chart is a graphical representation of a project's timeline that displays all of the individual tasks necessary to complete the project.



\

Advantages:-

- A PERT chart allows a manager to evaluate the time and resources necessary to complete a project. It also allows the manager to track required assets during any stage of production in the course of the project.
- PERT analysis incorporates data and information supplied by a number of departments. This combining of information encourages department responsibility and identifies all responsible parties across the organization.
- Finally, PERT charts are useful input for what-if analyses. Understanding the possibilities concerning the flow of project resources and milestones allows management to achieve the most efficient and useful project path.

Disadvantages:-

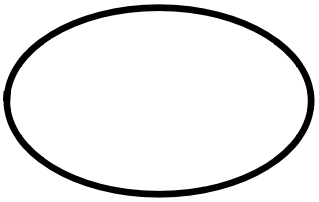
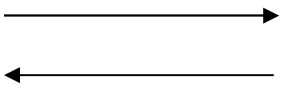
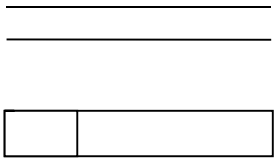
- The information that goes into a PERT chart can be highly subjective. They may include unreliable data or unreasonable estimates for cost or time.
- PERT charts are deadline-focused and might not fully communicate the financial positioning of a project.
- Creating a PERT chart is labor-intensive and maintaining and updating the information requires additional time and resources. Continual review of the information provided, as well as the prospective positioning of the project, is required for a PERT chart to be of value.

DATA FLOW DIAGRAM

What is Data flow diagram:-

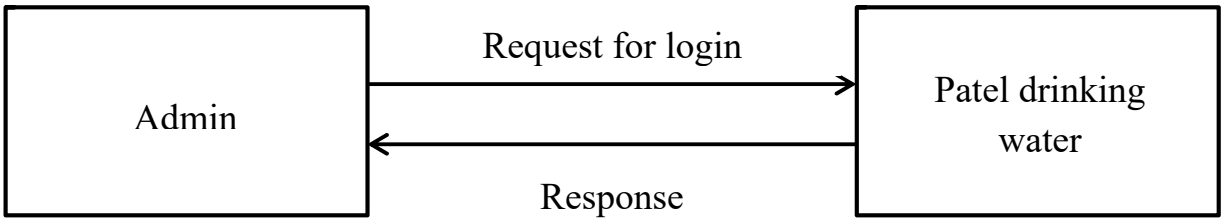
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It can be manual, automated, or a combination of both.

It shows how data enters and leaves the system, what changes the information, and where data is stored.

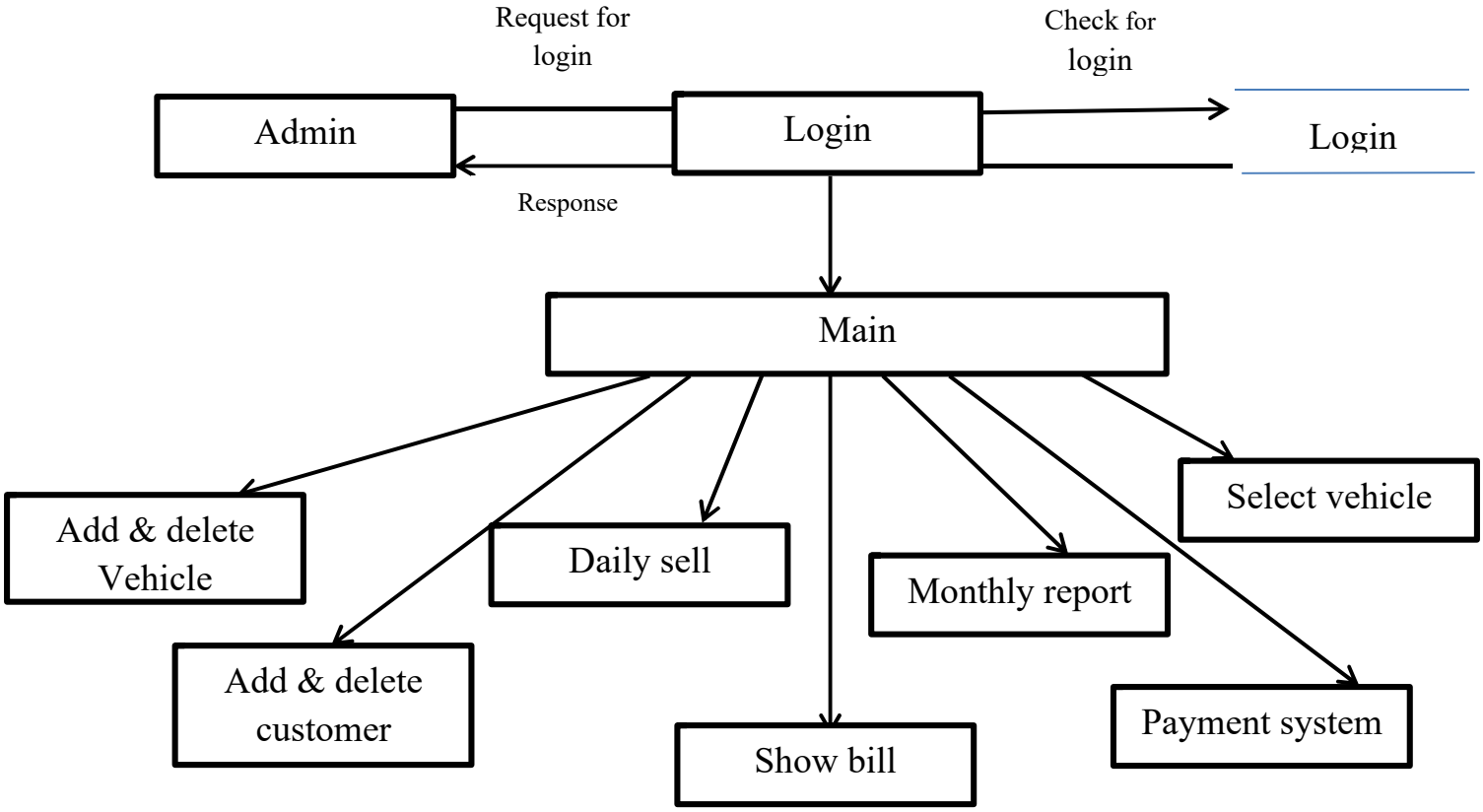
Symbol	Name	Use
	External Entity	Rectangle source and / sink destination data.
	Process / Function	Transformed, Store, or Distribute. Annotated with number and name of function.
	Data Flow	Direction of data flow single piece of data or logical collection of data.
	Data Store	Open Rectangle Parallel lines Data Structure, File, Table and Database.

The objective of a DFD is to show the scope and boundaries of a system as a whole. It may be used as a communication tool between a system analyst and any person who plays a part in the order that acts as a starting point for redesigning a system. The DFD is also called as a data flow graph or bubble chart.

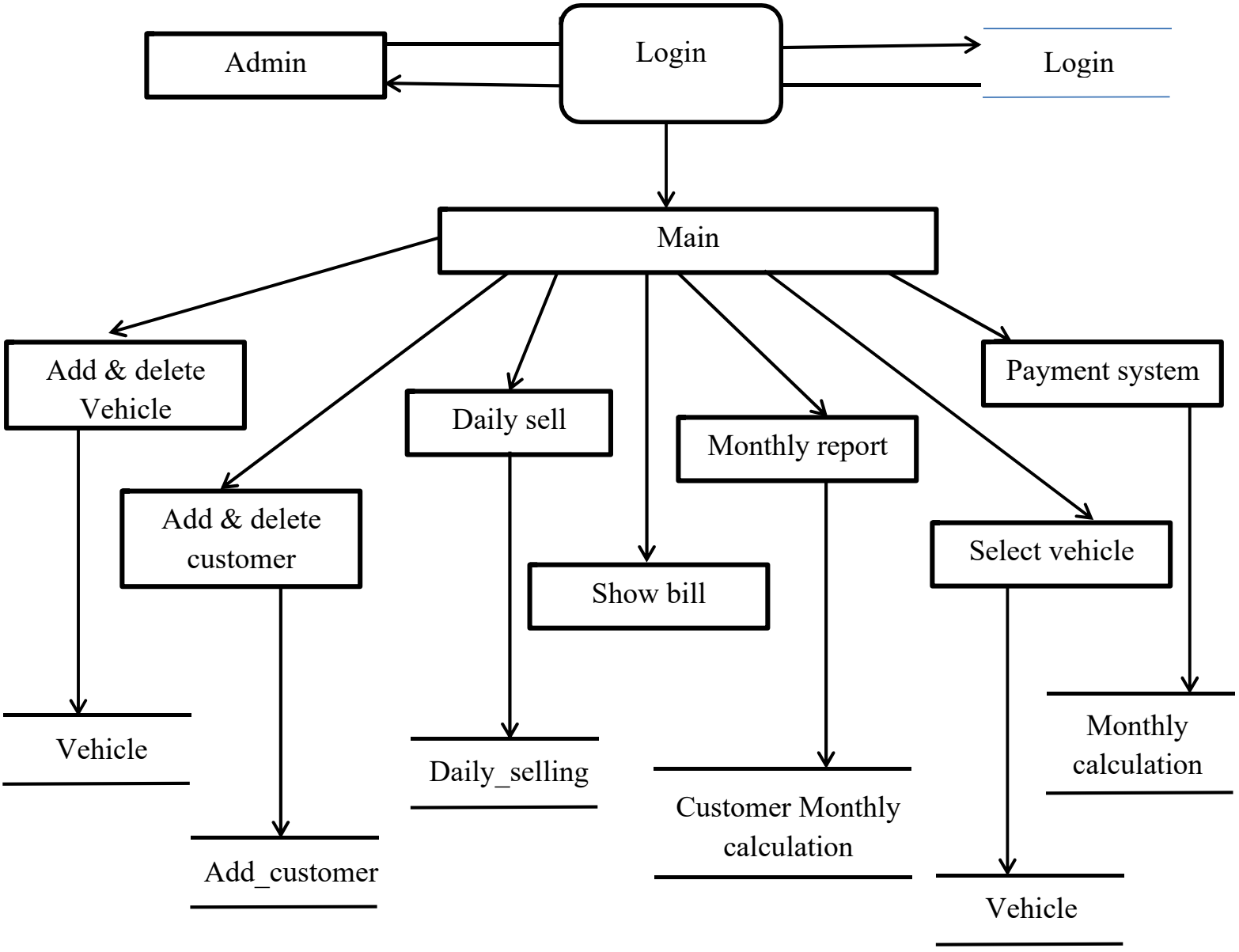
Context level diagram:-



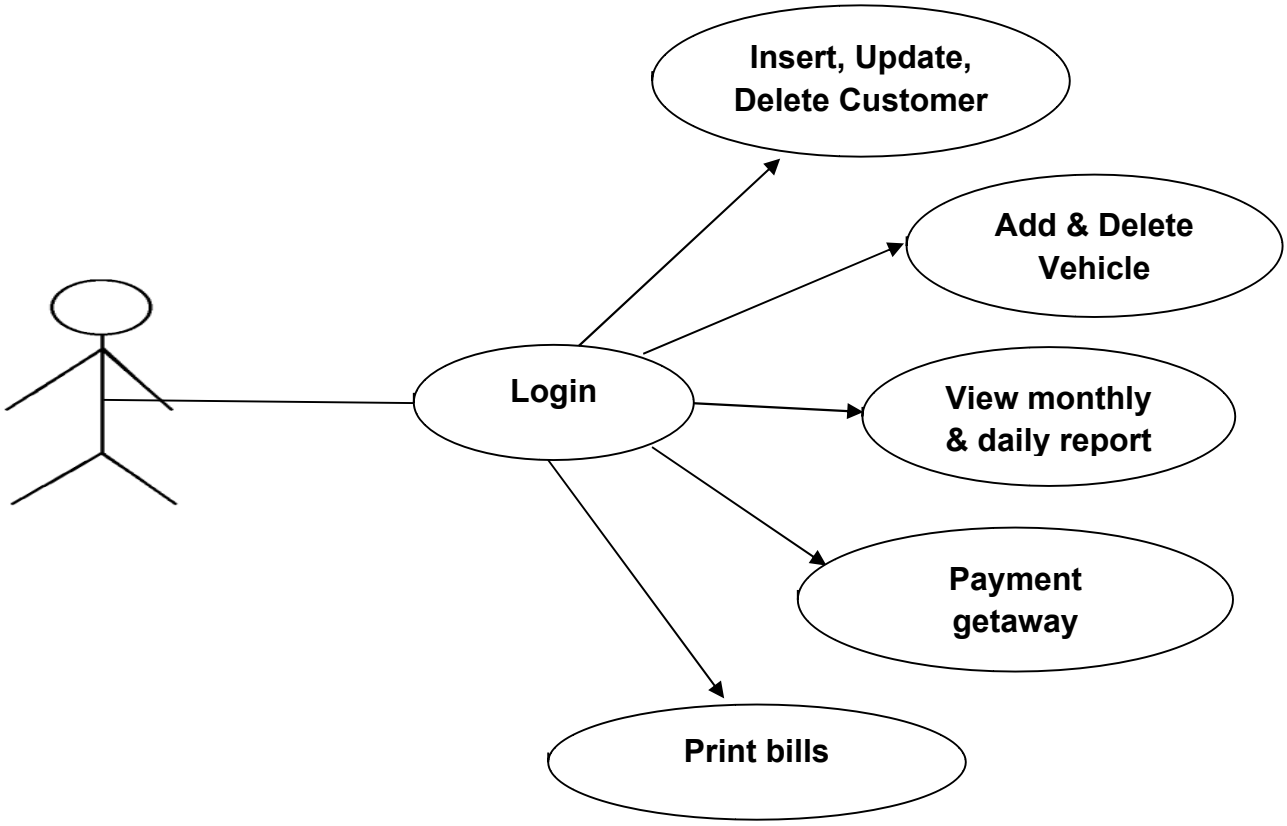
1st Level Diagram:



2nd level Diagram:



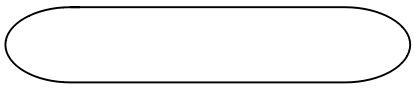
USE CASE DIAGRAM



FLOW CHART

A flowchart is a diagram that depicts a process, system or computer algorithm. They are widely used in multiple fields to document, study, and plan, improve and communicate often complex processes in clear, easy-to-understand diagrams. Flowcharts, sometimes spelled as flow charts, use rectangles, ovals, diamonds and potentially numerous other shapes to define the type of step, along with connecting arrows to define flow and sequence. They can range from simple, hand-drawn charts to comprehensive computer-drawn diagrams depicting multiple steps and routes. If we consider all the various forms of flowcharts, they are one of the most common diagrams on the planet, used by both technical and non-technical people in numerous fields. Flowcharts are sometimes called by more specialized names such as Process Flowchart, Process Map, Functional Flowchart, Business Process Mapping, Business Process Modeling and Notation (BPMN), or Process Flow Diagram (PFD). They are related to other popular diagrams, such as Data Flow Diagrams (DFDs) and Unified Modeling Language (UML) Activity Diagrams

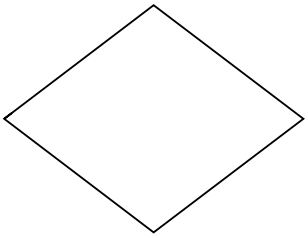
Basic Flowchart Symbols



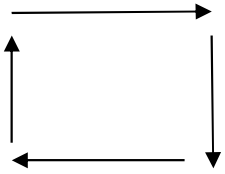
Terminal

Input / Output

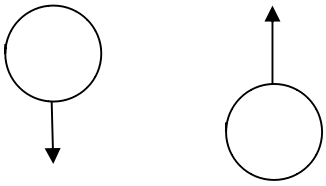
Processing



Decision

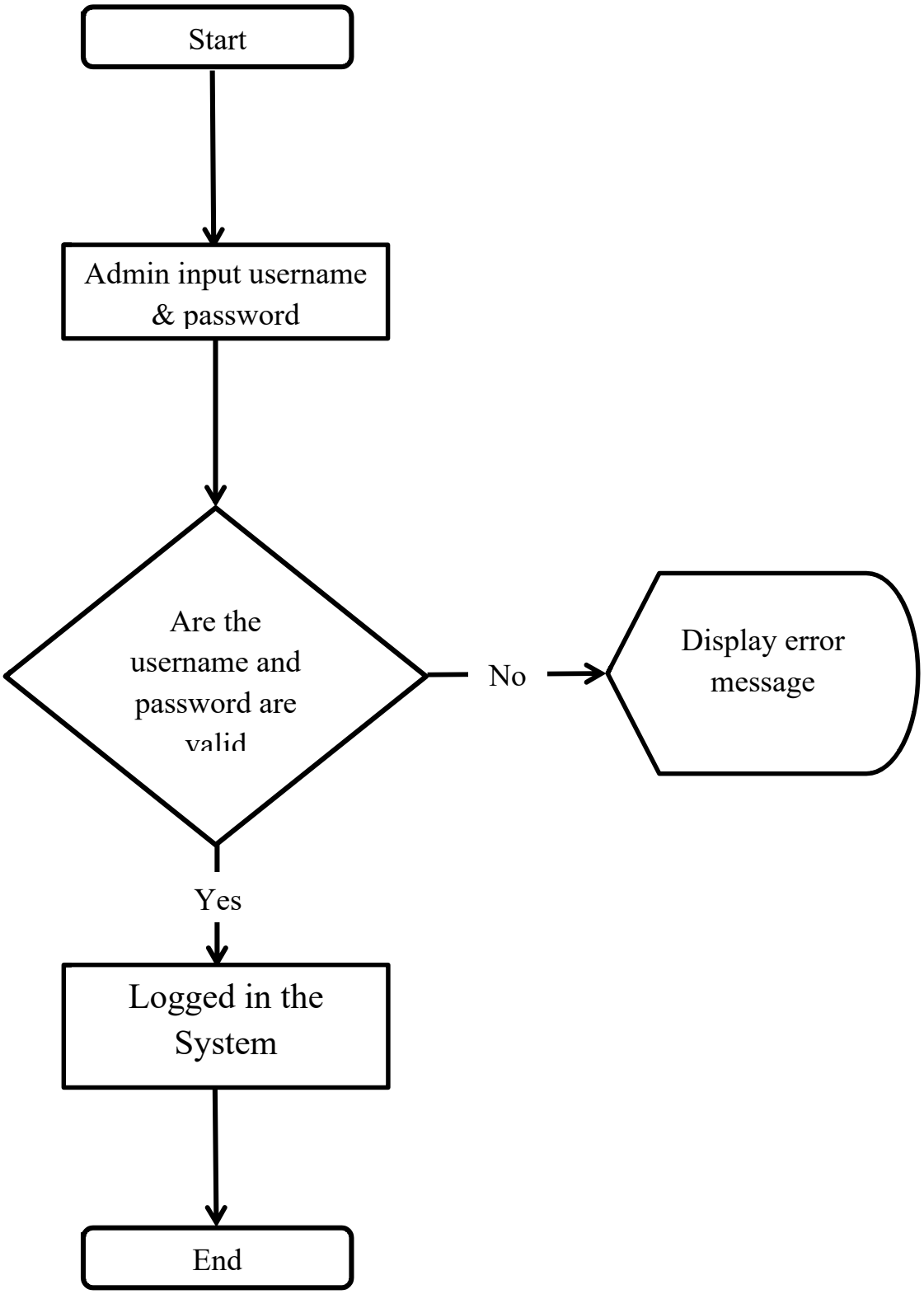


Flow Lines



Connectors

Flow chart for our program



COST ESTIMATION

For any new software project, it is necessary to know how much it will cost to develop and how much development time will it take. These estimates are needed before development is initiated, but how is this done? Several estimation procedures have been developed and are having the following attributes in common.

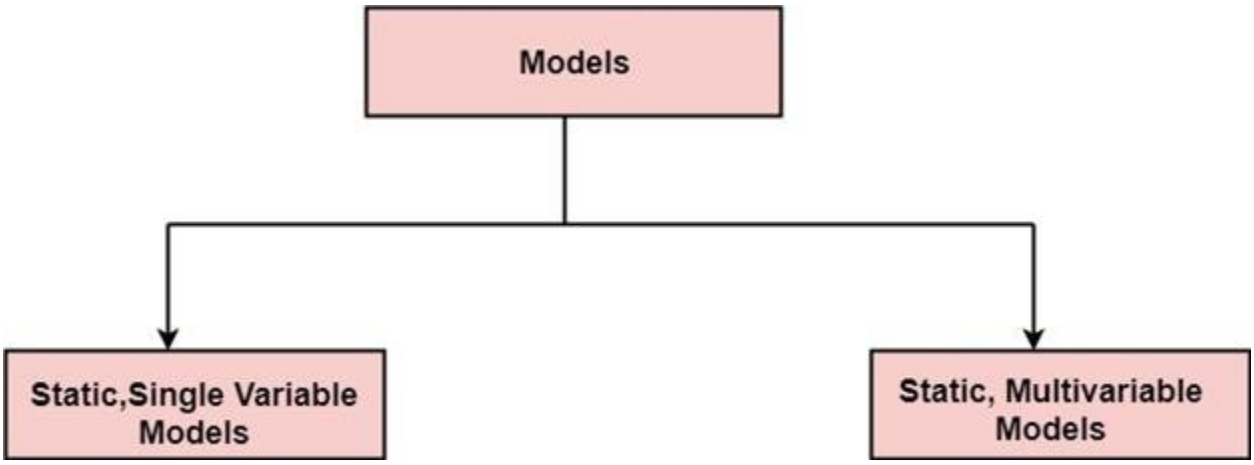
1. Project scope must be established in advanced.
2. Software metrics are used as a support from which evaluation is made.
3. The project is broken into small PCs which are estimated individually. To achieve true cost & schedule estimate, several option arise.
4. Delay estimation
5. Used symbol decomposition techniques to generate project cost and schedule estimates.
6. Acquire one or more automated estimation tools.

Uses of Cost Estimation

1. During the planning stage, one needs to choose how many engineers are required for the project and to develop a schedule.
2. In monitoring the project's progress, one needs to access whether the project is progressing according to the procedure and takes corrective action, if necessary.

Cost Estimation Models

A model may be static or dynamic. In a static model, a single variable is taken as a key element for calculating cost and time. In a dynamic model, all variable are interdependent, and there is no basic variable.



Static, Single Variable Models: When a model makes use of single variables to calculate desired values such as cost, time, efforts, etc. is said to be a single variable model. The most common equation is:

Formula:-

$$C=aL^b$$

The Software Engineering Laboratory established a model called SEL model, for estimating its software production. This model is an example of the static, single variable model.

$E=1.4L^{0.93}$

$DOC=30.4L^{0.90}$

$D=4.6L^{0.26}$

Where E= Efforts (Person per month)

DOC=Documentation (Number of pages)

D = Duration (D, in months)

L = Number of Lines per code

Static, Multivariable Models:

These models are based on method (1), they depend on several variables describing various aspects of the software development environment. In some model, several variables are needed to describe the software development process, and selected equation combined these variables to give the estimate of time & cost. These models are called multivariable models.

WALSTON and FELIX develop the models at IBM provide the following equation gives a relationship between lines of source code and effort:

$$\mathbf{E=5.2L^{0.91}}$$

In the same manner duration of development is given by

$$\mathbf{D=4.1L^{0.36}}$$

The productivity index uses 29 variables which are found to be highly correlated productivity as follows:

$$\mathbf{I = \sum_{i=1}^{29} W_i X_i}$$

Where W_i is the weight factor for the **i**th variable and X_i={-1,0,+1} the estimator gives $X_i$one of the values **-1, 0 or +1** depending on the variable decreases, has no effect or increases the productivity.

Cost estimate of our software:-

Here, we are estimating the price of the software by the module wise. So the price of the software's one module is ₹2000 because the modules are made dynamically and the market prices of the dynamic modules are ₹2000.

So we have the total 9 modules available in the project. Then in total amount of the software is ₹18,000

1 dynamic module price = ₹2000

Module 9 * 2000 = 18000/-

DATA DICTIONARY & NORMALIZATION

Database Name: Patel Drinking

Table 1: Login

Field	Type	Null	Primary Key	Extra
username	varchar(200)	No	No	
password	varchar(200)	No	No	

Table 2: Add customer

Field	Type	Null	Primary Key	Extra
Id	varchar(200)	No	No	
Name	Varchar(200)	No	No	
Price	numeric(38,0)	No	No	
number	varchar(20)	No	No	
address	varchar(150)	No	No	
vehicle	varchar(100)	No	No	

Table 3: customer monthly supply

Field	Type	Null	Primary Key	Extra
customer_id	varchar(100)	No	No	
customer_name	Varchar(100)	No	No	
Supply	numeric(18,0)	No	No	
Total	numeric(18,0)	No	No	
Month	varchar(100)	No	No	

Table 4: customer daily selling

Field	Type	Null	Primary Key	Extra
customer_id	varchar(100)	No	No	
customer_name	Varchar(100)	No	No	
Supply	numeric(18,0)	No	No	
customer_return	numeric(18,0)	No	No	
remark	varchar(100)	No	No	
d_date	date	No	No	
amount	numeric(38,0)	No	No	
Price	numeric(38,0)	No	No	
customer_vehicle	varchar(100)	No	No	

Table 5: monthly calculation

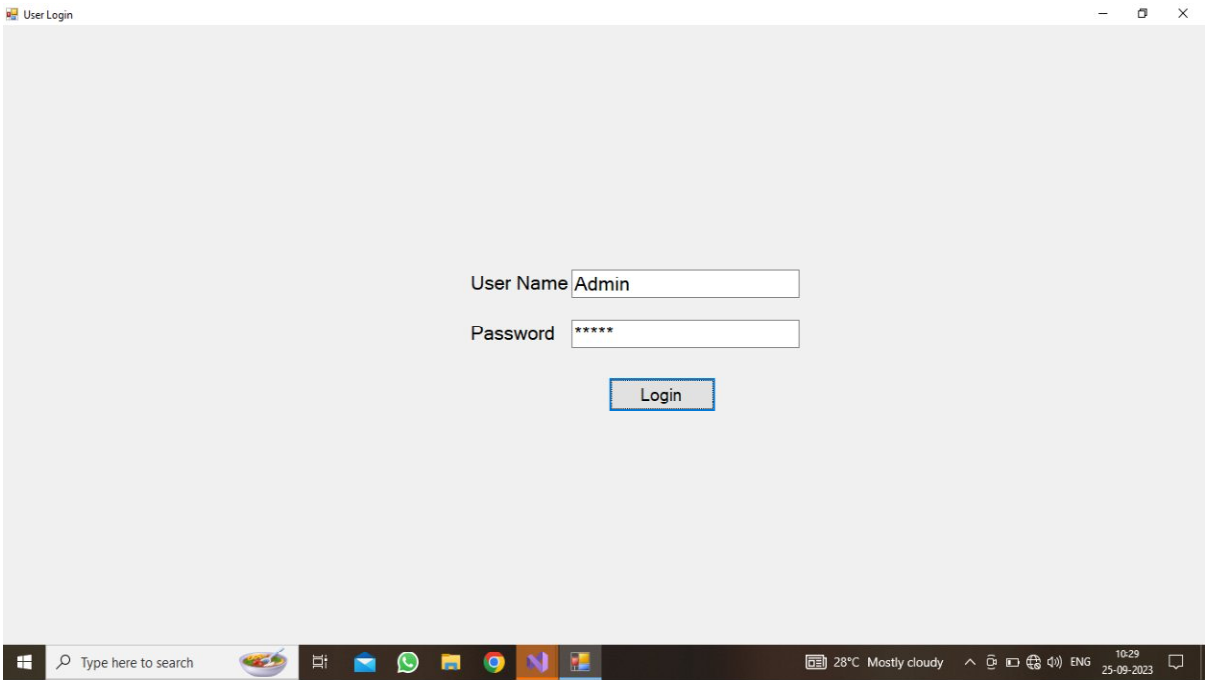
Field	Type	Null	Primary Key	Extra
customer_id	varchar(100)	No	No	
Month	varchar(200)	No	No	
Total	numeric(38,0)	No	No	
deposite	numeric(38,0)	No	No	
Due	numeric(38,0)	No	No	
date_transaction	varchar(100)	No	No	
customer_name	varchar(100)	No	No	
vehicle	varchar(100)	No	No	

Table 6: vehicle

Field	Type	Null	Primary Key	Extra
number	varchar(200)	No	No	

SCREEN LAYOUTS

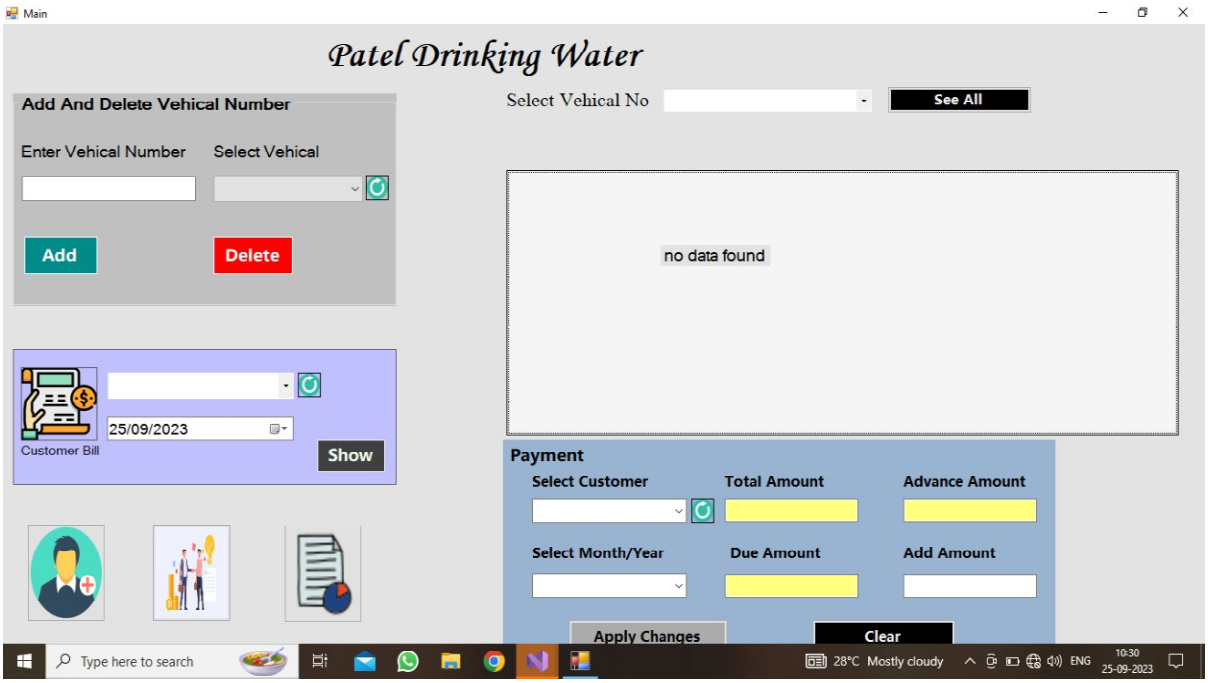
Login Page



Code:-

```
Private void btn_login_Click(object sender, EventArgs e)
    {
      if (txt_username.Text == "Admin" && txt_password.Text == "Admin")
      {
        Form2 frm2 = new Form2();
        frm2.Show();
        this.Hide();
      }
      else
      {
        MessageBox.Show("Login Failed");
        this.Close();
      }
    }
```

Main page:-



Add vehicle:-

```
if(txt_ent_veh_number.Text != "")
            {
               cmd1 = new SqlCommand("select number from vehical where number = '" +
txt_ent_veh_number.Text + " ' ", add_vhl);
               SqlDataReader chk = cmd1.ExecuteReader();
               if(chk.HasRows)
               {
                  chk.Close();
                  MessageBox.Show("Vehical Number Already Exist... ");
               }
               else
               {
                  chk.Close();
                  cmd1 = new SqlCommand("insert into vehical values('" + txt_ent_veh_number.Text + "
')", add_vhl);
                  int a = cmd1.ExecuteNonQuery();
```

```
                if(a > 0)
                {
                   txt_ent_veh_number.Text = "";
                   vehical_no();
                   MessageBox.Show("Vehical Insert Successfully....",
"",MessageBoxButtons.OK,MessageBoxIcon.Information);
                }
             }
             add_vhl.Close();
          }
          else
          {
             MessageBox.Show("Please Insert Vehical Number.." , " ", MessageBoxButtons.OK ,
MessageBoxIcon.Warning);
          }
```

Delete vehicle:-

```
private void btn_delete_Click(object sender, EventArgs e)
    {
       Connection con2 = new Connection();
       SqlConnection delet_vhl = new SqlConnection(con2.connection);
       SqlCommand cmd4;
       delet_vhl.Open();
       try
       {
          if(cmb_sel_vehical.Text != "")
          {
             cmd4 = new SqlCommand("delete from vehical where number = '" + cmb_sel_vehical.Text +
"'", delet_vhl);
             int a = cmd4.ExecuteNonQuery();
             if(a > 0)
             {
                vehical_no();
```

```
                MessageBox.Show("Vehical Number Deleted..", "Delete", MessageBoxButtons.OK,
MessageBoxIcon.Information);
            }
            else
            {
                MessageBox.Show("HELLO");
            }
            delet_vhl.Close();
        }
        else
        {
            MessageBox.Show("Please Select Vehical Number...","Vehical Not
Selected..",MessageBoxButtons.OK,MessageBoxIcon.Warning);
        }
      }
      catch(Exception ed)
      {
        MessageBox.Show("" + ed);
      }
    }
```

Show all:-

```
private void btn_see_all_Click(object sender, EventArgs e)
    {
      vehical_no();
    }
    //reset the payment panel
    private void btnClear_Click(object sender, EventArgs e)
    {
      foreach (Control cnt in pnl_payment.Controls)
      {
        if(cnt is TextBox)
        {
          cnt.Text = " ";
        }
```

```
        }
        cmbSelectDate.Items.Clear();
        txtTotalAmount.Text = " ";
        card_customer_id();
    }
```

Add customer:-

```
private void btnAddCustomer_Click(object sender, EventArgs e)
    {
        Add_Customer add = new Add_Customer();
        add.Show();
    }
```

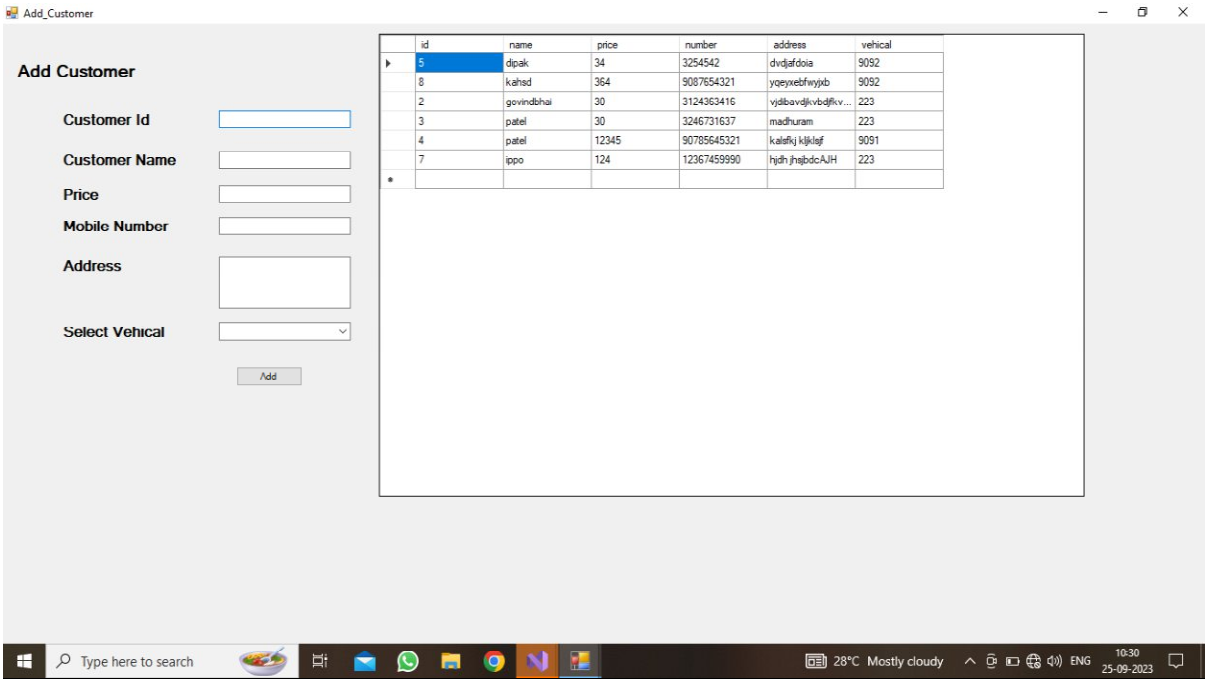
Daily selling:-

```
private void btn_daily_selling_Click(object sender, EventArgs e)
    {
        if(ds == null)
        {
            ds= new Daily_selling();
            ds.Show();
        }
        else
        {
            ds.WindowState = FormWindowState.Normal;
            ds.StartPosition = FormStartPosition.CenterScreen;
            ds.BringToFront();
        }
    }
```

Monthly Report:-

```
private void btn_monthly_report_Click(object sender, EventArgs e)
    {
      if(rpt_m == null)
      {
        rpt_m = new Monthly_Report();
        rpt_m.Show();
      }
      else
      {
        rpt_m.BringToFront();
        rpt_m.WindowState = FormWindowState.Maximized;
        rpt_m.Show();
      }
    }
```

Add customer:-

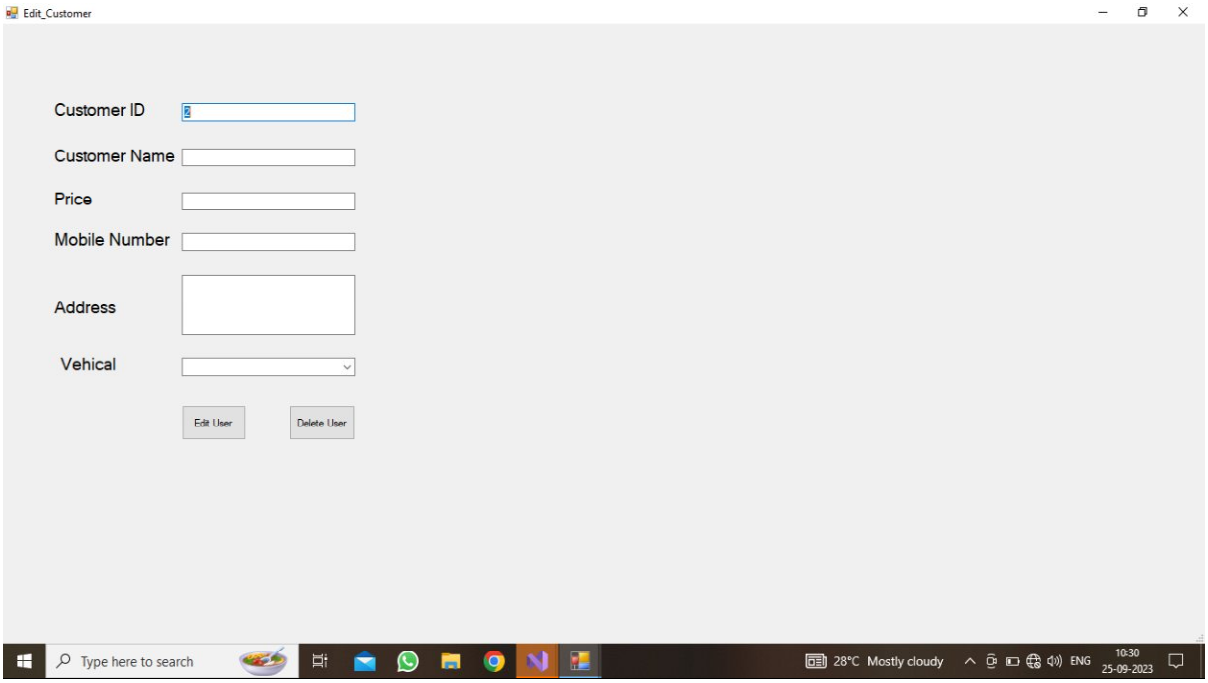


Code:-

```
private void btnAdd_Click(object sender, EventArgs e)
    {
      if (txtCustomerId.Text != null && cmbSelectVehical!= null)
      {
        cmbSelectVehical.Items.Clear();
        Connection con_string = new Connection();
        con = new SqlConnection(con_string.connection);
        try
        {
          con.Open();
          cmd = new SqlCommand("insert into Add_customer values('" +
txtCustomerId.Text + "','" + txtCustomerName.Text + "','" +
```

```
Double.Parse(txtPrice.Text) + "'," + txtMobileNumber.Text + "'," +
txtAddress.Text + "'," + cmbSelectVehical.Text + "')", con);
                int ans = cmd.ExecuteNonQuery();
                if (ans > 0)
                {
                   MessageBox.Show("data inserted");
                }
                getdata();
                if(grid_Customer.Rows.Count > 0)
                {
                   lblLoading.Invoke(new Action(() => lblLoading.Visible = false));
                }
                con.Close();
            }
            catch(Exception ee)
            {
                MessageBox.Show("there is some error" + ee);
            }
        }
        else
        {
            MessageBox.Show("please Fill the From ");
        }
    }
```

Edit customer:-



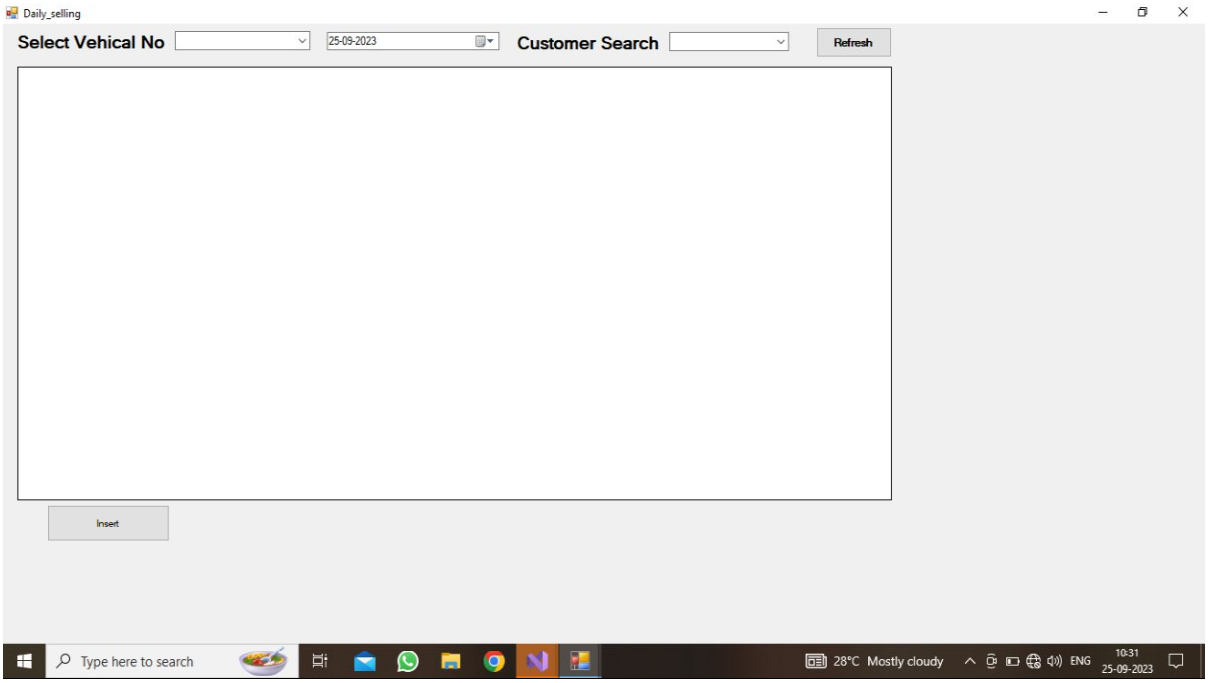
Code:-

```
private void btnSavePdf_Click(object sender, EventArgs e)
    {
       try
        {
            string file = cus_no + "-" + customer_name + "-" +
DateTime.Now.ToString("dd-MM-yyyy");
            string filename = @"D:\User Login\Monthlybill" + file + ".pdf";
bill_report.ExportToDisk(CrystalDecisions.Shared.ExportFormatType.PortableDo
cFormat, filename);
            PrinterSettings getprinterName = new PrinterSettings();

            bill_report.PrintOptions.PrinterName = getprinterName.PrinterName;
```

```
            bill_report.PrintToPrinter(1, true, 1, 1);
            bill_report.PrintOptions.PaperSize =
CrystalDecisions.Shared.PaperSize.PaperA4;
            MessageBox.Show("DONE");
            Connection con1 = new Connection();
            SqlConnection conn = new SqlConnection(con1.connection.ToString());
            conn.Open();
            Int64 id = Int64.Parse(str_bill_no);
            id++;
            if(id == 10000)
            {
               id = 1;
            }
            SqlCommand cmd_update_bill = new SqlCommand("update
bill_number set bill_id=" + id + "", conn);
            cmd_update_bill.ExecuteNonQuery();
            conn.Close();

            Customer_Bill_Load(null, null);
         }
         catch(Exception ee)
         {
            if(ee is System.Runtime.InteropServices.COMException)
            {
               btnSavePdf.PerformClick();
```

```
            }
            else
            {
                MessageBox.Show(" " + ee);
            }
        }
    }
```

Daily selling:-




```
private void cmbSelectVehicalNo_SelectedIndexChanged(object sender,
EventArgs e)

    {

       cmbCustomerSearch.Text = " ";

       DataTable tbl = new DataTable();

       tbl.Columns.Add("Customer Id");

       tbl.Columns.Add("Customer Name");

       tbl.Columns.Add("Supply");

       tbl.Columns.Add("Return");

       tbl.Columns.Add("Remark");

       tbl.Columns.Add("Supply??", typeof(bool));

       tbl.Columns.Add("Date");
```

```
        try
        {
            Connection con_string = new Connection();
            SqlConnection veh_con = new SqlConnection(con_string.connection);
            veh_con.Open();
            cmd = new SqlCommand("select id,name from Add_customer where
vehical='" + cmbSelectVehicalNo.Text + "' order by convert(int,id)", veh_con);
            SqlDataReader rdr_client = cmd.ExecuteReader();
            while (rdr_client.Read())
            {
                row = tbl.NewRow();
                row[0] = rdr_client["id"].ToString();
                row[1] = rdr_client["name"].ToString();
                row[6] = dateTimePicker1.Text.ToString();
                tbl.Rows.Add(row);
            }
            rdr_client.Close();
            veh_con.Close();
        }
        catch (Exception ex)
        {
            MessageBox.Show("" + ex);
```

```
        }
        grid_DailySelling.DataSource = tbl;
        grid_DailySelling.Refresh();
        grid_DailySelling.Columns[0].ReadOnly = true;
        grid_DailySelling.Columns[1].ReadOnly = true;
        grid_DailySelling.Columns[6].ReadOnly = true;
    }
```

SPECIAL UTILITIES

- Admin can create PDF.
- Admin can print bills.

TESTING

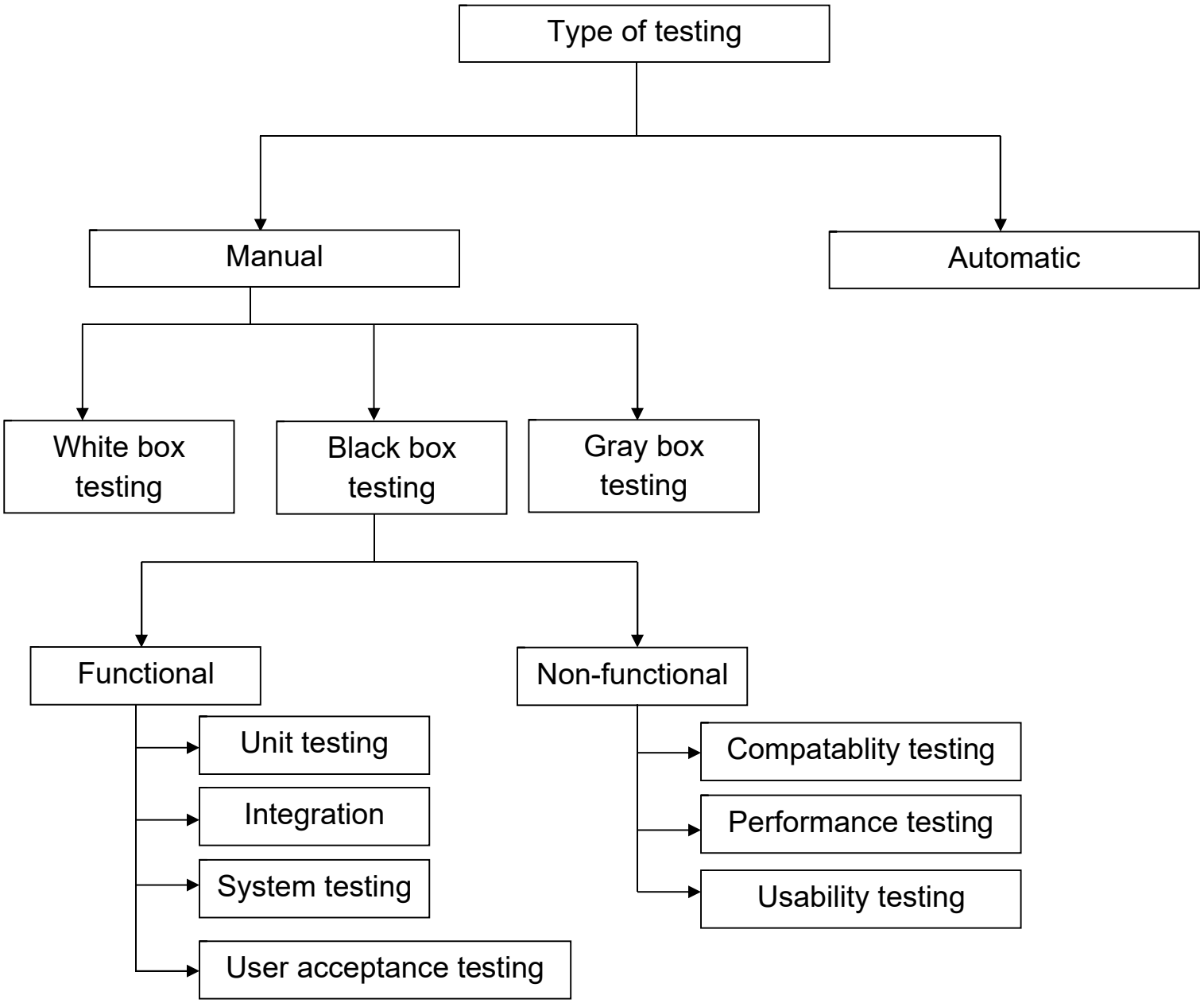
What is Software Testing

Software testing is a process of identifying the correctness of software by considering its all attributes (Reliability, Scalability, Portability, Re-usability, Usability) and evaluating the execution of software components to find the software bugs or errors or defects.

Software testing provides an independent view and objective of the software and gives surety of fitness of the software. It involves testing of all components under the required services to confirm that whether it is satisfying the specified requirements or not. The process is also providing the client with information about the quality of the software.

Types of Software testing

We have various types of testing available in the market, which are used to test the application or the software. The types of testing are below:-


Type of testing
Manual
Automatic
White box testing
Black box testing
Gray box testing
Functional
Non-functional
Unit testing
Integration
System testing
User acceptance testing
Compatablity testing
Performance testing
Usability testing

1. Manual testing

The process of checking the functionality of an application as per the customer needs without taking any help of automation tools is known as manual testing. While performing the manual testing on any application, we do not need any specific knowledge of any testing tool, rather than have a proper understanding of the product so we can easily prepare the test document.

Manual testing can be further divided into three types of testing, which are as follows:

- **White box testing**
- **Black box testing**
- **Gray box testing**

➢ White-box testing

The white box testing is done by Developer, where they check every line of a code before giving it to the Test Engineer. Since the code is visible for the Developer during the testing, that's why it is also known as White box testing.

➢ Black box testing

The black box testing is done by the Test Engineer, where they can check the functionality of an application or the software according

to the customer /client's needs. In this, the code is not visible while performing the testing; that's why it is known as black-box testing.

- Gray Box testing

Gray box testing is a combination of white box and Black box testing. It can be performed by a person who knew both coding and testing. And if the single person performs white box, as well as black-box testing for the application, is known as Gray box testing.

2. Automation testing

Automation testing is a process of converting any manual test cases into the test scripts with the help of automation tools, or any programming language is known as automation testing. With the help of automation testing, we can enhance the speed of our test execution because here, we do not require any human efforts. We need to write a test script and execute those scripts.

Test case

- If you can enter number in name then it can't allow.
- If you can enter alphabet in mobile no then it can't allow.

IMPLEMENTATION

In software engineering, implementation refers to the process of translating a software design or specification into a working and executable program or system. It is one of the crucial phases in the software development life cycle (SDLC) and typically follows the design phase. Implementation involves writing, coding, and testing the actual software components based on the design and requirements.

Here are some key aspects of implementation in software engineering:

1) Coding:
   - During implementation, software developers write the source code for the software. This involves translating the high-level design and algorithms into a programming language that the computer can understand and execute.
2) Testing:
   - Testing is an integral part of implementation. Developers perform unit testing to ensure that individual components or modules of the software work correctly. They also conduct integration testing to verify that these modules can work together seamlessly.
3) Debugging:
   - Debugging is the process of identifying and fixing errors or bugs in the code. It's a critical step in ensuring the software functions as intended and doesn't produce unexpected or incorrect results.
4) Documentation:

- Developers create documentation during implementation to describe the code, its purpose, usage, and any dependencies. Proper documentation is essential for future maintenance and collaboration among team members.

5) Version Control:
   - Version control systems, such as Git, are often used during implementation to track changes to the code base, collaborate with team members, and ensure that different versions of the software are well-managed.
6) Optimization:
   - Developers may also optimize the code for performance, memory usage, and other factors to ensure the software meets its performance requirements.
7) Adherence to Coding Standards:
   - Many development teams have coding standards or style guides that developers must follow to ensure consistency and maintainability of the code base.
8) Security Considerations:
   - Implementers should be mindful of security best practices to minimize vulnerabilities and protect the software from potential threats.
9) Reusability:
   - In some cases, code modules or components may be designed for reuse in future projects or within the same project to improve efficiency and maintainability.
10) Peer Review:

- Code reviews are often conducted by other team members to assess the quality of the implementation, identify potential issues, and ensure compliance with coding standards.

BIBLIOGRAPHY

Website:-

- www.geeksforgeeks.org
- https://javatpoint.com/
- https://chat.openai.com

Books:-

- Microsoft visual C# step by step 8th edition