**Introduction about Project**

“Payroll Management” is a distributed application, developed to evaluate the performance of employees working in any organization. It maintains the information about a company, personal details of their employees, also the project details assigned to particular developer. The application is actually a suite of applications developed using Java.

It is simple to understand and can be used by anyone who is not even familiar with simple employees system. It is user friendly and just asks the user to follow step by step operations by giving him few options. It is fast and can perform many operations of a company.

This software package has been developed using the powerful coding tools of Java at Front End and Mysql at Back End. Because of the Mysql features, the software is very user friendly. The package contains different modules like Contacts, Search for property and other useful Links. This version of the software has multi-user approach. For further enhancement or development of the package, user’s feedback will be considered.

This project basically deals with five modules and their further sub modules. First module is the employee module into which we can enter employee details such as his name, address, phone number, his basic salary and many more. After that we can view the details further by using the employee id, and we can modify the details also. Similarly in department we have the details of all the HOD’s of the departments. Next comes the salary module in this we can view the salary issued to the employee. And can issue them to the employee we want to and can fix it to them.

We can put the grades also in the grade module. Basically we create the grades in this with specified details. We can view the details accordingly. We can just enter the grade name and can view the details encapsulated in the grade. Other than this we can view the whole thing all together by getting into the view grade section.

We can view the report also in this. It can be viewed in this easily that to which the monthly salary has been allotted and for which month. We can also look out those employees to whom the salary has not been issued and the further details also such as for which month it has not been issued. This is the basic overview of the whole project

**Present state of the art**

Presently salary calculation is done manually, it take so much of time to compose salary of all employees. It also takes very long time to make salary slip ready. Due to manual process some time it takes very long time, in turn it delays the salary distribution. This is a big problem to manage when salary is not generated in time. The other main problem is errors, even with double cross check here or there some errors will happen, this again create large problem. To solve all this the organization require very good software to take care of all these.

**Need of Computerization of System**

The client uses MS Excel and maintains their records however it is not possible for them to share the data from multiple system in multi user environment, there is lot of duplicate work, and chance of mistake. When the records are changed they need to update each and every excel file. There is no option to find and print previous saved records. There is no security anybody can access any report and sensitive data also reports of summary. This Payroll Management System is used to overcome the entire problem which they are facing currently, and making complete atomization of manual system to computerized system.

**OBJECTIVE:**

The main objective of our project is to prepare a record of all employees working in a firm.

* Personal Record of all Employees.
* HODs of all Departments.
* Salary Calculation of all Employees.
* Grade Assigning to all Employees.
* Record of all Employees.

## **MODULES OF PAYROLL MANAGEMENT SYSTEM:**

### **Administration of payroll management system:**

Administration of the industry can be a difficult process for many company directors. The administrators are responsible to leverage the company’s property and business to return stake holders as quickly and as fully as possible.

This is the backbone of any industry all the departments are managed as well as all the decision making is done by the administration. It will hire the employee for any type of work done required in the company.

The administration has privileges to get information about all the employees working in their department. So basically, it makes sure that company is going in right direction and employees part is to complete the project whatever provided by the administration.

### **An employee of payroll management system:**

Even after all the digitization of the industry, there will always be a requirement of an eligible employee for the correct management of events. These are the individual who works part-time or full-time by the contract of employment with the administration of the company, which could be written or oral.

### **Department of payroll management system:**

These are the different fragments of the company which are divided as the department. Each of the department has assigned the id and name to avoid any kind of confusion.

Every department has an individual head which could be in the administration. Every department works individually and provides the performance accordingly. Each department will have the manager who will manage the department.

Following are the departments a company could have:

Production

Research and Development

Purchasing

Human Resource Management

Account and Finance These departments are managed by the administration.

Each department will have the number of employees required.

### **PRE EXISTING SYSTEM OF PAYROLL MANAGEMENT SYSTEM:**

The pre-existing system for this purpose is offline so doesn’t require any device or internet connection. Previously all the information was written on the papers and the employees are hired to manage the paperwork of the company.

All this work will be depended on the individual’s working speed. All the employees have to synchronize the paper work manually by communicating each other.

This paper work takes a lot of time and requires more space to store safely and all the additional resources required to ensure the safety. In this system, there is no surety or reliability as the humans are likely to make an error.

### **DRAWBACKS OF PAYROLL MANAGEMENT SYSTEM:**

**Required a lot of paperwork.**

Everything is done on the paper and these are highly prone to damages and requires a good amount of security and space to store.

Required Buying of goods more frequent as compared to the online system.

Likely to have an error.

Lack of storage space for the documents.

If these files are less in number than it is not a that big problem but if the number increases then searching for a particular document can give a personal nightmare as these are more prone to damages and even it gets the searching job done the time required to do the job is not ideal at all.

Require more physical work and man power.

Editing is another level of problem in those paper documents the only option is to make new documents if any correction needs to be made that is not a good at all for the environmental credentials.

No backup of the information.

Where all the drawbacks of the pre-existing system have been overcome in the proposed system. It definitely improves the speed by automating most of the task required to be done manually previously.

The automated systems are more reliable and efficient now a day for these types of work.Improvements in the proposed system:

Does not require paper work

Only Human error while entering the information

Not required much space

Automatically search and sort the information

Require less physical work and man power

Editing is a lot easier

Have backup option While the proposed system offers a lot of improvement over the pre-existing but it will be at one time cost there will be a requirement of storage devices and access devices like a computer but it still has enough advantages to cover those by reducing the man power.

It will also help in generating the report of the employee and evaluating the performance through the automated system. Although this provides the safety and security from any kind of disaster and also the authorization authentication scheme provides more flexibility for the end user of the system.

#### SYSTEM ANALYSIS

## **Existing System:**

The present calculation of the salary is done manually and it takes a lot of time to compose the employee’s salary. The salary slip also takes a long time to create and as this is a tiresome and cumbersome process it takes a lot of time.

It also delays the salary distribution system. Sometimes the salary is not generated within the stipulated time period thus creating a lot of hassle.

Error is the other major problem of the manual system and even with repeated cross check some of the other errors surely persist which can cause a lot of problems.

To solve this problem the organization need an ideal software that can take care of  all these.

## **Proposed System:**

The new proposed Payroll Management system will surely eradicate all these issues that they are facing now and this software is created in such a manner that it automatically generate the salary of the employees every month without any worries.

This software also keeps a track of the employees attendance, payslip and payroll summary.

**Feasibility Study**

FEASIBILITY STUDY Feasibility study is a cross life cycle activity and should be continuously performed throughout a system project. Employee Payroll Management is feasible due to following reasons :

• EPM is feasible because it will meet the user requirements.

• This Application can be implemented using the current technology and with in specified budget and schedule.

• This application is compatible with other existing hardware and software.

* Operational Feasibility
* Technical Feasibility
* Economical Feasibility
* Motivational Feasibility
* Scheduled Feasibility

Feasibility study :

Every project is feasible for given unlimited resources and infinitive time. Feasibility study is anevaluation of the proposed system regarding its workability, impact on the organization, ability to meet the user needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study before it is approved for development .Feasibility and risk analysis and related in many ways. If a project risk is great and feasibility of producing software is reduced. During the feasibility analysis in this project has been discussed below in the abovementioned topics.

**Operational Feasibility:**

Feasibility of the working of the system after the installation in the organization as mentioned in the feasibility analysis.

**Technical Feasibility:**

Technical feasibility is frequently the most difficult area to ensure this stage. It is essential that the process of analysis and definition to be conducted parallel toan assessment of the technical feasibility. The consideration that is normally associated with technical feasibility includes the resources availability of the Organization where the project is to be developed and implemented. By taking these facts into consideration before developing the resource availability at Retail Outlet of Hindustan Petroleum was observed. As very limited resources are required for this project hence this project is considered feasible for development.

**Economic Feasibility:**

An evaluation of development cost is weighted against the ultimate income or benefits derived from the developed system. There was no need of extra hardware and software for development of this project. Hence this project has economically justified for development in this organization.

**Motivational Feasibility:**

An evaluation of the probability that the organization is sufficient motivation to support the development and implementation of the application with necessary user participation, resources, training etc. The interest and support shown by the organization during the system study do not seem that the new system developed to have efficient support from the organization.

**Schedule Feasibility:**

An evaluation of the time needed for the development of this project. The time schedule required for the development of this project is very important, since more development time effects machine time, costs and delays in the development of the other systems. So the project should be complete within affixed schedule

time as far as the organization is concerned.

**Software Requirement Specification**

**Hardware configuration**

* Processor                    core i3 or above
* RAM                          4 MB Ram
* Monitor                      15 inch color
* Hard disk                    min. 20 GB

**Software configuration**

* Operating system       Windows 7 Professional or above
* Language                   C++ - Language

**system design**

**Software Requirement Specification**

**C++ Language Introduction**

**C++** is a general-purpose programming language that was developed as an enhancement of the C language to include object-oriented paradigm. It is an imperative and a **compiled** language.

C++ is a middle-level language rendering it the advantage of programming low-level (drivers, kernels) and even higher-level applications (games, GUI, desktop apps etc.). The basic syntax and code structure of both C and C++ are the same.

Some of the ***features & key-points*** to note about the programming language are as follows:

* **Simple**: It is a simple language in the sense that programs can be broken down into logical units and parts, has a rich library support and a variety of data-types.
* **Machine Independent but Platform Dependent**: A C++ executable is not platform-independent (compiled programs on Linux won’t run on Windows), however they are machine independent.
* **Mid-level language**: It is a mid-level language as we can do both systems-programming (drivers, kernels, networking etc.) and build large-scale user applications (Media Players, Photoshop, Game Engines etc.)
* **Rich library support**: Has a rich library support (Both standard ~ built-in data structures, algorithms etc.) as well 3rd party libraries (e.g. Boost libraries) for fast and rapid development.
* **Speed of execution**: C++ programs excel in execution speed. Since, it is a compiled language, and also hugely procedural. Newer languages have extra in-built default features such as garbage-collection, dynamic typing etc. which slow the execution of the program overall. Since there is no additional processing overhead like this in C++, it is blazing fast.
* **Pointer and direct Memory-Access**: C++ provides pointer support which aids users to directly manipulate storage address. This helps in doing low-level programming (where one might need to have explicit control on the storage of variables).
* **Object-Oriented**: One of the strongest points of the language which sets it apart from C. Object-Oriented support helps C++ to make maintainable and extensible programs. i.e. Large-scale applications can be built. Procedural code becomes difficult to maintain as code-size grows.
* **Compiled Language**: C++ is a compiled language, contributing to its speed.

**Applications of C++:**   
C++ finds varied usage in applications such as:

* Operating Systems & Systems Programming. e.g. *Linux-based OS (Ubuntu etc.)*
* Browsers *(Chrome & Firefox)*
* Graphics & Game engines *(Photoshop, Blender, Unreal-Engine)*
* Database Engines *(MySQL, MongoDB, Redis etc.)*

Cloud/Distributed Systems

**C++ Compilers**

When you write any program in C++ language then to run that program you need to compile that program using a C++ Compiler which converts your program into a language understandable by a computer. This is called machine language (ie. binary format). So before proceeding, make sure you have C++ Compiler available at your computer. It comes alongwith all flavors of Unix and Linux.

If you are working over Unix or Linux then you can type *gcc -v* or *cc -v* and check the result. You can ask your system administrator or you can take help from anyone to identify an available C Compiler at your computer.

If you don't have C compiler installed at your computer then you can use below given link to download a GNU C++ Compiler and use it.

**System Design**

**Data Flow Diagram**

**DFD** is the abbreviation for **Data Flow Diagram**. The flow of data of a system or a process is represented by DFD. It also gives insight into the inputs and outputs of each entity and the process itself. DFD does not have control flow and no loops or decision rules are present. Specific operations depending on the type of data can be explained by a flowchart. Data Flow Diagram can be represented in several ways. The DFD belongs to structured-analysis modeling tools. Data Flow diagrams are very popular because they help us to visualize the major steps and data involved in software-system processes.

#### Components of DFD

The Data Flow Diagram has 4 components:

* **Process**  
  Input to output transformation in a system takes place because of process function. The symbols of a process are rectangular with rounded corners, oval, rectangle or a circle. The process is named a short sentence, in one word or a phrase to express its essence
* **Data Flow**  
  Data flow describes the information transferring between different parts of the systems. The arrow symbol is the symbol of data flow. A relatable name should be given to the flow to determine the information which is being moved. Data flow also represents material along with information that is being moved. Material shifts are modeled in systems that are not merely informative. A given flow should only transfer a single type of information. The direction of flow is represented by the arrow which can also be bi-directional.
* **Warehouse**  
  The data is stored in the warehouse for later use. Two horizontal lines represent the symbol of the store. The warehouse is simply not restricted to being a data file rather it can be anything like a folder with documents, an optical disc, a filing cabinet. The data warehouse can be viewed independent of its implementation. When the data flow from the warehouse it is considered as data reading and when data flows to the warehouse it is called data entry or data updation.

* **Terminator**  
  The Terminator is an external entity that stands outside of the system and communicates with the system. It can be, for example, organizations like banks, groups of people like customers or different departments of the same organization, which is not a part of the model system and is an external entity. Modeled systems also communicate with terminator.

#### Rules for creating DFD

* The name of the entity should be easy and understandable without any extra assistance(like comments).
* The processes should be numbered or put in ordered list to be referred easily.
* The DFD should maintain consistency across all the DFD levels.
* A single DFD can have maximum processes upto 9 and minimum 3 processes.

#### Levels of DFD

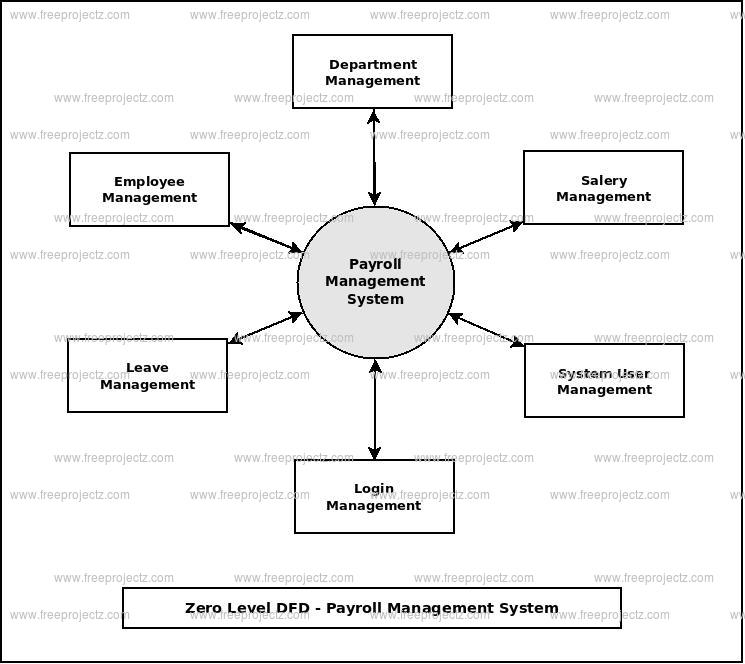
DFD uses hierarchy to maintain transparency thus multilevel DFD’s can be created. Levels of DFD are as follows:

* 0-level DFD
* 1-level DFD:
* 2-level DFD:

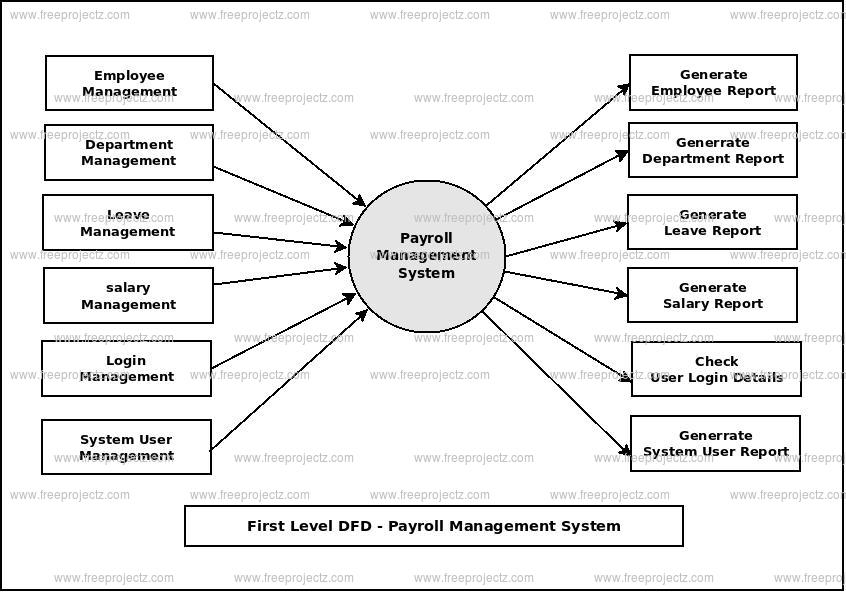
Data Flow Diagram

DFD graphically representing the functions, or processes, which capture, manipulate, store, and distribute data between a system and its environment and between components of a system.

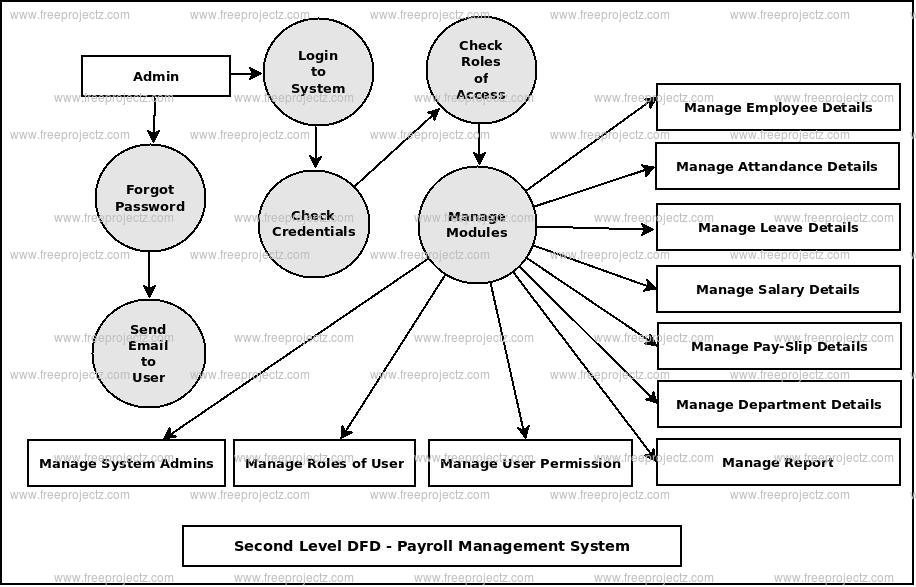
**Level – 0 DFD**



**Level – 1 DFD**



**Level – 2 DFD**



**ER Diagram**

An Entity Relationship (ER) Diagram is a type of flowchart that illustrates how “entities” such as people, objects or concepts relate to each other within a system. ER Diagrams are most often used to design or debug relational databases in the fields of software engineering, business information systems, education and research. Also known as ERDs or ER Models, they use a defined set of symbols such as rectangles, diamonds, ovals and connecting lines to depict the interconnectedness of entities, relationships and their attributes. They mirror grammatical structure, with entities as nouns and relationships as verbs

## Uses of entity relationship diagrams

* **Database design:** ER diagrams are used to model and design relational databases, in terms of logic and business rules (in a logical data model) and in terms of the specific technology to be implemented (in a physical data model.) In software engineering, an ER diagram is often an initial step in determining requirements for an information systems project. It’s also later used to model a particular database or databases. A relational database has an equivalent relational table and can potentially be expressed that way as needed.
* **Database troubleshooting:**ER diagrams are used to analyze existing databases to find and resolve problems in logic or deployment. Drawing the diagram should reveal where it’s going wrong.
* **Business information systems:**The diagrams are used to design or analyze relational databases used in business processes. Any business process that uses fielded data involving entities, actions and interplay can potentially benefit from a relational database. It can streamline processes, uncover information more easily and improve results.
* **Business process re-engineering (BPR):**ER diagrams help in analyzing databases used in business process re-engineering and in modeling a new database setup.
* **Education:**Databases are today’s method of storing relational information for educational purposes and later retrieval, so ER Diagrams can be valuable in planning those data structures.
* **Research:** Since so much research focuses on structured data, ER diagrams can play a key role in setting up useful databases to analyze the data.

## The components and features of an ER diagram

ER Diagrams are composed of entities, relationships and attributes. They also depict cardinality, which defines relationships in terms of numbers. Here’s a glossary:

### Entity

A definable thing—such as a person, object, concept or event—that can have data stored about it. Think of entities as nouns. Examples: a customer, student, car or product. Typically shown as a rectangle.

**Entity type:**A group of definable things, such as students or athletes, whereas the entity would be the specific student or athlete. Other examples: customers, cars or products.

**Entity set:** Same as an entity type, but defined at a particular point in time, such as students enrolled in a class on the first day. Other examples: Customers who purchased last month, cars currently registered in Florida. A related term is instance, in which the specific person or car would be an instance of the entity set.

**Entity categories:** Entities are categorized as strong, weak or associative. A **strong entity** can be defined solely by its own attributes, while a **weak entity** cannot. An associative entity associates entities (or elements) within an entity set.

**Entity keys:** Refers to an attribute that uniquely defines an entity in an entity set. Entity keys can be super, candidate or primary. **Super key:**A set of attributes (one or more) that together define an entity in an entity set. **Candidate key:**A minimal super key, meaning it has the least possible number of attributes to still be a super key. An entity set may have more than one candidate key. **Primary key:**A candidate key chosen by the database designer to uniquely identify the entity set. **Foreign key:**Identifies the relationship between entities.

### Relationship

How entities act upon each other or are associated with each other. Think of relationships as verbs. For example, the named student might register for a course. The two entities would be the student and the course, and the relationship depicted is the act of enrolling, connecting the two entities in that way. Relationships are typically shown as diamonds or labels directly on the connecting lines.

**Recursive relationship:**The same entity participates more than once in the relationship.

### Attribute

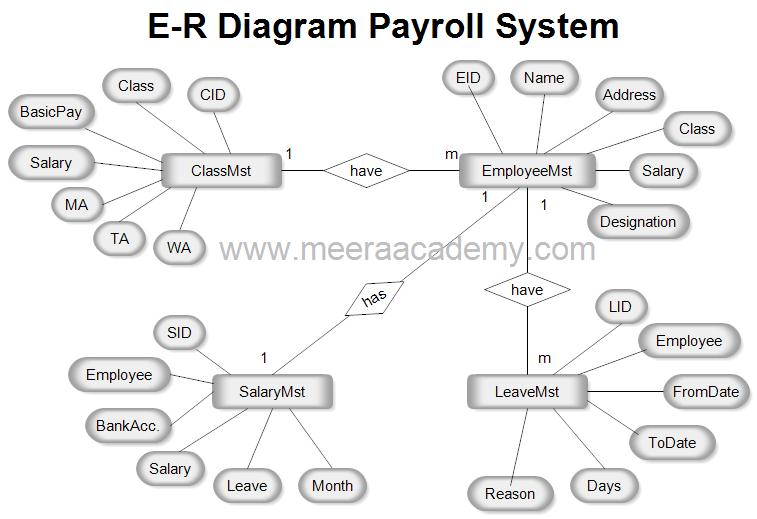
A property or characteristic of an entity. Often shown as an oval or circle.

**Descriptive attribute:**A property or characteristic of a relationship (versus of an entity.)

**Attribute categories:**Attributes are categorized as simple, composite, derived, as well as single-value or multi-value. **Simple:** Means the attribute value is atomic and can’t be further divided, such as a phone number. **Composite:**Sub-attributes spring from an attribute. **Derived:**Attributed is calculated or otherwise derived from another attribute, such as age from a birthdate.

**Multi-value:**More than one attribute value is denoted, such as multiple phone numbers for a person.

**Single-value:** Just one attribute value. The types can be combined, such as: simple single-value attributes or composite multi-value attributes.



**Database**

A database is an organized collection of structured information, or data, typically stored electronically in a computer system. A database is usually controlled by a database management system (DBMS). Together, the data and the DBMS, along with the applications that are associated with them, are referred to as a database system, often shortened to just database.

Data within the most common types of databases in operation today is typically modeled in rows and columns in a series of tables to make processing and data querying efficient. The data can then be easily accessed, managed, modified, updated, controlled, and organized. Most databases use structured query language (SQL) for writing and querying data.

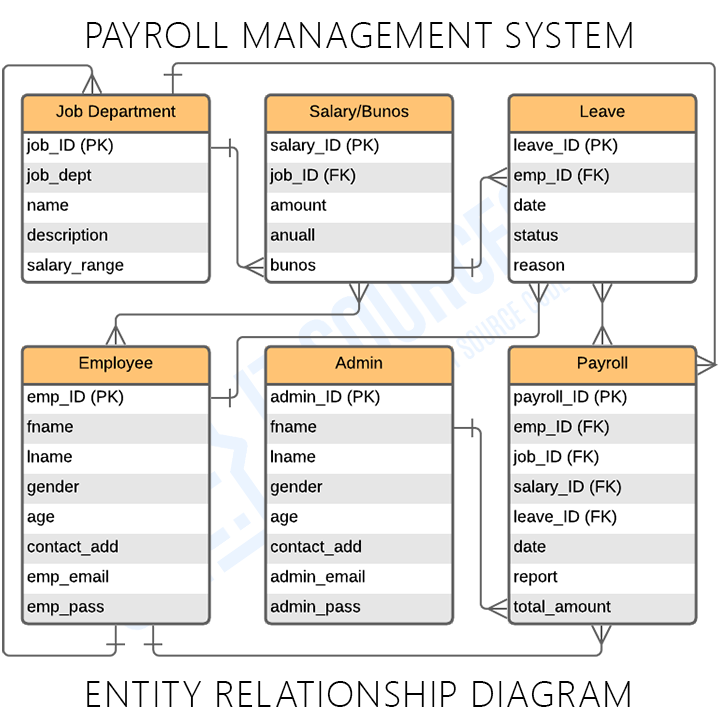
## What is Structured Query Language (SQL)?

SQL is a programming language used by nearly all relational databases to query, manipulate, and define data, and to provide access control. SQL was first developed at IBM in the 1970s with Oracle as a major contributor, which led to implementation of the SQL ANSI standard, SQL has spurred many extensions from companies such as IBM, Oracle, and Microsoft. Although SQL is still widely used today, new programming languages are beginning to appear.

**What are databases used for?**

Businesses use data stored in databases to make informed business decisions. Some of the ways organizations use databases include the following:

* **Improve business processes.** Companies collect data about [business processes](https://searchcio.techtarget.com/definition/business-process), such sales, order processing and customer service. They analyze that data to improve these processes, expand their business and grow revenue.
* **Keep track of customers.** Databases often store information about people, such as customers or users. For example, [social media](https://whatis.techtarget.com/definition/social-media) platforms use databases to store user information, such as names, email addresses and user behavior. The data is used to recommend content to users and improve the [user experience](https://searchcio.techtarget.com/definition/UX-user-experience).
* **Secure personal health information.** Healthcare providers use databases to securely store [personal health data](https://searchhealthit.techtarget.com/definition/personal-health-information) to inform and improve patient care.
* **Store personal data.** Databases can also be used to store personal information. For example, personal [cloud storage](https://searchstorage.techtarget.com/definition/cloud-storage) is available for individual users to store media, such as photos, in a managed cloud.



**Implementation & Testing**

**Testing**

An entity is shown in rectangle.

 A diamond represent the relationship among number of entities.

 The attributes shown as ovals are connected to the entities or relationship bylines.

 Diamond,oval and relationships are labeled.

**Security Testing of the Project**

Testing is vital for the success of any software. no system design is ever perfect. Testing is alsocarried in two phases. first phase is during the software engineering that is during the modulecreation. second phase is after the completion of software. this is system testing which verifiesthat the whole set of programs hanged together.

**White Box Testing:**

In this technique, the close examination of the logical parts through the software are tested bycases that exercise species sets of conditions or loops. all logical parts of the software checkedonce. errors that can be corrected using this technique are typographical errors, logicalexpressions which should be executed once may be getting executed more than once and errorresulting by using wrong controls and loops. When the box testing tests all the independent partwithin a module a logical decisions on their true and the false side are exercised , all loops andbounds within their operational bounds were exercised and internal data structure to ensuretheir validity were exercised once.

**Black Box Testing:**

This method enables the software engineer to device sets of input techniques that fully exerciseall functional requirements for a program. black box testing tests the input, the output and theexternal data. it checks whether the input data is correct and whether we are getting the desiredoutput.

**Alpha Testing:**

 Acceptance testing is also sometimes called alpha testing. Be spoke systems are developed fora single customer. The alpha testing proceeds until the system developer and the customeragree that the provided system is an acceptable implementation of the system requirements.

Beta Testing:

On the other hand, when a system isto be marked as a software product, another processcalled beta testing is often conducted. During beta testing, a system is delivered among anumber of potential users who agree to use it. The customers then report problems to the

developers. This provides the product for real use and detects errors which may not have beenanticipated by the system developers.

Unit Testing:

Each module is considered independently. it focuses on each unit of software as implementedin the source code. it is white box testing.

Integration Testing:

Integration testing aims at constructing the program structure while at the same constructingtests to uncover errors associated with interfacing the modules. modules are integrated by usingthe top down approach.

Validation Testing:

Validation testing was performed to ensure that all the functional and performance requirementsare met.

System Testing:

It is executing programs to check logical changes made in it with intention of finding errors. asystem is tested for online response, volume of transaction, recovery from failure etc. Systemtesting is done to ensure that the system satisfies all the user requirements.

**Implementation Methodology:**

Model View Controller or MVC as it is popularly called, is a software design pattern fordeveloping web applications. A Model View Controller pattern is made up of thefollowing three parts:

Model

The lowest level of the pattern which is responsible for maintaining data.

View

This is responsible for displaying all or a portion of the data to the user.

Controller

Software Code that controls the interactions between the Model andView.MVC is popular as it isolates the application logic from the user interface layer andsupports separation of concerns. Here the Controller receives all requests for theapplication and then works with the Model to prepare any data needed by the View. TheView then uses the data prepared by the Controller to generate a final presentableresponse. The MVC abstraction can be graphically represented as follows.

## **Coding**

#include<stdio.h>

#include<iostream>

#include<stdlib.h>

#include<string.h>

#include<conio.h>

#include<windows.h>

#include<time.h>

#include<iomanip>

#include<ctype.h>

#define max 50

using namespace std;

int num=0;

void gotoXY(int,int);

void Cdelay(int);

void border(int, int,int, int);

void borderNoDelay(int, int,int, int);

void loginFrame(int, int, int, int);

void intro();

void login();

void menu();

void insert();

void edit();

void editmenu();

void editname(int);

void editcode(int);

void editdes(int);

void editexp(int);

void editage(int);

void editsalary(int);

void list();

void deletes();

void search();

void setWindowSize(int,int);

void saverecords();

void getrecords();

bool isFilePresent();

void displayPayslip();

class employee

{

public:

char name[20];

int code;

char designation[20];

int exp;

int age;

int salary;

char AnyLoan;

int HRA;

int PF;

int tax;

int MealAllowance;

int TransportAllowance;

int MedicalAllowance;

int LoanBalance;

int LoanDebit;

int grosspay;

int workingHours;

int DA;

};

employee emp[max],tempemp[max];

void getrecords()

{

FILE \*fp;

fp = fopen("Records.txt","r");

int c=0;

if(fp!=NULL)

{

while(feof(fp)==0)

{

fscanf(fp,"%s\t%d\t%s\t%d\t%d\t%d\t%c\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\n",&emp[c].name,&emp[c].code,&emp[c].designation,&emp[c].exp,&emp[c].age,&emp[c].salary,&emp[c].AnyLoan,&emp[c].HRA,&emp[c].PF,&emp[c].tax,&emp[c].MealAllowance,&emp[c].TransportAllowance,&emp[c].MedicalAllowance,&emp[c].LoanBalance,&emp[c].LoanDebit,&emp[c].grosspay,&emp[c].workingHours,&emp[c].DA);

c++;

}

num=c;

}

fclose(fp);

}

void saverecords()

{

if(num==0)

{

system("del Records.txt");

}

else

{

FILE \*fp;

fp = fopen("Records.txt","w");

for(int i=0;i<num;i++)

{

fprintf(fp,"%s\t%d\t%s\t%d\t%d\t%d\t%c\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\t%d\n",emp[i].name,emp[i].code,emp[i].designation,emp[i].exp,emp[i].age,emp[i].salary,emp[i].AnyLoan,emp[i].HRA,emp[i].PF,emp[i].tax,emp[i].MealAllowance,emp[i].TransportAllowance,emp[i].MedicalAllowance,emp[i].LoanBalance,emp[i].LoanDebit,emp[i].grosspay,emp[i].workingHours,emp[i].DA);

}

fclose(fp);

}

}

void Cdelay(int msec)

{

long goal = msec + (clock());

while (goal > (clock()));

}

bool isFilePresent()

{

FILE \*fp;

fp = fopen("Records.txt","r");

if(fp==NULL)

return false;

else

return true;

}

void loginFrame1(int xLenS = 18, int yLenS = 8, int xLenE = 55, int yLenE = 15)

{

system("cls");

gotoXY(xLenS,yLenS);printf("%c",201);

gotoXY(xLenS,yLenE);printf("%c",200);

for(int i=xLenS+1;i<=xLenE-1;i++) //Top and Bottom Border line

{

//Cdelay(0);

gotoXY(i,yLenS);

printf("%c",205);

//puts(style);

gotoXY(i,yLenE);

//puts(style);

printf("%c",205);

}

gotoXY(xLenE,yLenS);printf("%c",187);

gotoXY(xLenE,yLenE);printf("%c",188);

for(int i=yLenS+1;i<=yLenE-1;i++) //Left and Right Border Line

{

//Cdelay(20);

gotoXY(xLenS, i);

printf("%c",186);

//puts(style);

gotoXY(xLenE, i);

printf("%c",186);

//puts(style);

}

printf("\n\n");

}

void login()

{

char UserName[30],Password[30],ch;int i=0;

gotoXY(20,10);

printf("Enter UserName : ");

cin>>UserName;

gotoXY(20,12);

cout<<"Enter Password : ";

while(1)

{

ch = getch();

if(ch==13)

break;

if(ch==32||ch==9)

continue;

else

{

cout<<"\*";

Password[i]=ch;

i++;

}

}

Password[i] = '\0';

if(strcmp(UserName,"admin")==0 && strcmp(Password,"password")==0)

{

system("cls");

loginFrame1();

gotoXY(27,10);

cout<<"Login Success!!!";

gotoXY(21,12);

cout<<"Will be redirected in 3 Seconds...";

gotoXY(56,12);

Cdelay(1000);

gotoXY(44,12);

cout<<"\b \b2";

gotoXY(56,12);

Cdelay(1000);

gotoXY(44,12);

cout<<"\b \b1";

gotoXY(56,12);

Cdelay(1000);

}

else

{

system("cls");

loginFrame1();

gotoXY(27,10);

printf("Access Denied!!!\a");

gotoXY(21,12);

cout<<"Will be redirected in 3 Seconds...";

gotoXY(56,12);

Cdelay(1000);

gotoXY(44,12);

cout<<"\b \b2";

gotoXY(56,12);

Cdelay(1000);

gotoXY(44,12);

cout<<"\b \b1";

gotoXY(56,12);

Cdelay(1000);

system("cls");

loginFrame1();

login();

}

}

void setWindowSize(int width=670,int height=445)

{

//HWND console = GetConsoleWindow();

RECT r;

// GetWindowRect(console, &r);

// MoveWindow(console, r.left, r.top, width, height, TRUE);

}

void gotoXY(int X, int Y)

{

COORD coordinates;

coordinates.X = X;

coordinates.Y = Y;

SetConsoleCursorPosition(GetStdHandle(STD\_OUTPUT\_HANDLE), coordinates);

}

void borderNoDelay(int xLenS = 2, int yLenS = 2,int xLenE = 76, int yLenE = 24 )

{

system("cls");

gotoXY(xLenS,yLenS);printf("%c",201);

gotoXY(xLenS,yLenE);printf("%c",200);

for(int i=xLenS+1;i<=xLenE-1;i++) //Top and Bottom Border line

{

gotoXY(i,yLenS);

printf("%c",205);

gotoXY(i,yLenE);

printf("%c",205);

}

gotoXY(xLenE,yLenS);printf("%c",187);

gotoXY(xLenE,yLenE);printf("%c",188);

for(int i=yLenS+1;i<=yLenE-1;i++) //Left and Right Border Line

{

gotoXY(xLenS, i);

printf("%c",186);

gotoXY(xLenE, i);

printf("%c",186);

}

printf("\n\n");

}

void border(int xLenS = 2, int yLenS = 2,int xLenE = 76, int yLenE = 22 )

{

system("cls");

gotoXY(xLenS,yLenS);printf("%c",201);

gotoXY(xLenS,yLenE);printf("%c",200);

for(int i=xLenS+1;i<=xLenE-1;i++) //Top and Bottom Border line

{

Cdelay(15);

gotoXY(i,yLenS);

printf("%c",205);

//puts(style);

gotoXY(i,yLenE);

//puts(style);

printf("%c",205);

}

gotoXY(xLenE,yLenS);printf("%c",187);

gotoXY(xLenE,yLenE);printf("%c",188);

for(int i=yLenS+1;i<=yLenE-1;i++) //Left and Right Border Line

{

Cdelay(15);

gotoXY(xLenS, i);

printf("%c",186);

//puts(style);

gotoXY(xLenE, i);

printf("%c",186);

//puts(style);

}

printf("\n\n");

}

void loginFrame(int xLenS = 18, int yLenS = 8, int xLenE = 55, int yLenE = 15)

{

border(xLenS,yLenS,xLenE,yLenE);

}

void insert()

{

borderNoDelay();

int i=num;

int sal,PF,TAX,HRA,MealA,MedicalA,TransportA,LoanBal,LoanDeb,h,DA;

char loan;

num+=1;

gotoXY(28,4);

printf("Insert New Record");

gotoXY(10,6);

cout<<"Name : ";

//cin.getline(emp[i].name,sizeof(emp[i].name));

cin>>emp[i].name;

gotoXY(10,8);

cout<<"Code : ";

cin>>emp[i].code;

gotoXY(10,10);

cout<<"Designation : ";

cin>>emp[i].designation;

gotoXY(10,12);

cout<<"Years of Experience : ";

cin>>emp[i].exp;

gotoXY(10,14);

cout<<"Age : ";

cin>>emp[i].age;

gotoXY(10,16);

cout<<"Enter Working Hours : ";

cin>>h;

sal = h\*300;

emp[i].workingHours = h;

do

{

gotoXY(10,18);

cout<<"Any Loan (Y/N) ?: \b \b";

loan=getche();

if(loan=='Y'|| loan == 'y' || loan =='n' || loan == 'N')

break;

}while(1);

if(loan=='y'|| loan=='Y')

{

gotoXY(10,20);

cout<<"Enter Loan Balance : ";

cin>>LoanBal;

}

else

{

LoanBal=0;

}

gotoXY(14,22);

cout<<"Recorded Succesfully...!!!";

TAX = 0.04 \* sal;

DA = 1.20 \* sal;

PF = 0.12 \* sal;

HRA = sal \* 0.27;

MealA = 300;

MedicalA = 300;

TransportA = 300;

LoanDeb = sal \* 0.09;

if(LoanDeb > LoanBal)

{

LoanBal = 0;

LoanDeb = LoanBal;

}

emp[i].salary = sal;

emp[i].DA = DA;

emp[i].tax=TAX;

emp[i].PF = PF;

emp[i].HRA = HRA;

emp[i].MealAllowance = MealA;

emp[i].MedicalAllowance = MedicalA;

emp[i].TransportAllowance = TransportA;

emp[i].LoanBalance = LoanBal-LoanDeb;

emp[i].AnyLoan = loan;

emp[i].LoanDebit = LoanDeb;

emp[i].grosspay = (sal+MealA+MedicalA+TransportA+HRA+DA)-(PF+TAX+LoanDeb) ;

getch();

}

void intro()

{

gotoXY(27,4); printf("PAYROLL MANAGEMENT SYSTEM");

gotoXY(24,20);printf("Press Any key to continue...");

getch();

}

void list()

{

//system("cls");

borderNoDelay();

gotoXY(20,4);

printf(" \*\*\*\*\*\*\*\* List of the Employees \*\*\*\*\*\*\*\*");

gotoXY(6,6);

cout<<"Name\tCode\tDesignation\tYears(EXP)\tAge\tSalary "<<endl;

gotoXY(6,7);

cout<<"------------------------------------------------------------------"<<endl;

for(int i=0,j=8;i<=num-1;i++,j++)

{

gotoXY(6,j);

cout<<emp[i].name;

gotoXY(19,j);

cout<<emp[i].code;

gotoXY(26,j);

cout<<emp[i].designation;

gotoXY(47,j);

cout<<emp[i].exp;

gotoXY(58,j);

cout<<emp[i].age;

gotoXY(66,j);

cout<<emp[i].grosspay;

}

getch();

}

void loading()

{

system("cls");

gotoXY(55,20);

printf("Loading...");

gotoXY(50,22);

for(int i = 0; i<20; i++)

{

Cdelay(200);

printf("%c",254);

}

}

void menu()

{

//system("cls");

borderNoDelay();

//gotoXY(0,0);

//cout<<"Number of Records Avaliable : "<<num;

gotoXY(16,4);

printf("\*\*\*\*\* Payroll Management System 1.0 \*\*\*\*\* ");

gotoXY(12,6);

cout<<"Press i ----> Insert New Record.";

gotoXY(12,8);

cout<<"Press e ----> Edit a Record.";

gotoXY(12,10);

cout<<"Press d ----> Delete a Record.";

gotoXY(12,12);

cout<<"Press s ----> Search a Record.";

gotoXY(12,14);

cout<<"Press l ----> List The Employee Table.";

gotoXY(12,16);

cout<<"Press p ----> Print Employee PaySlip.";

gotoXY(12,18);

cout<<"Press q ----> Quit Program.";

gotoXY(16,22);

cout<<"Select Your Option ====> ";

}

void deletes()

{

for(int i=0;i<num;i++)

{

tempemp[i]=emp[i];

}

//system("cls");

borderNoDelay();

int code;

int check=-1;

gotoXY(28,4);

printf("Delete a Record");

gotoXY(10,6);

cout<<"Enter the JobCode To Delete That Record :";

cin>>code;

int i,j;

for(i=0;i<=num-1;i++)

{

if(emp[i].code==code)

{

check=i;

}

}

if(check!=-1)

{

for(i=0,j=0;i<=num-1;i++,j++)

{

if(i==check)

{

i++;

}

emp[j]=tempemp[i];

}

num--;

}

}

void search()

{

//system("cls");

borderNoDelay();

int jobcode;

bool found = false;

gotoXY(10,4);

cout<<"You can Search only through the Jobcode of an Employee";

gotoXY(10,6);

cout<<"Enter Code Of the Employee : ";

cin>>jobcode;

for(int i=0;i<=num-1;i++)

{

if(emp[i].code==jobcode)

{

gotoXY(6,8);

cout<<"Name\tCode\tDesignation\tYears(EXP)\tAge\tSalary "<<endl;

gotoXY(6,9);

cout<<"------------------------------------------------------------------"<<endl;

gotoXY(6,11);

cout<<emp[i].name;

gotoXY(19,11);

cout<<emp[i].code;

gotoXY(26,11);

cout<<emp[i].designation;

gotoXY(47,11);

cout<<emp[i].exp;

gotoXY(58,11);

cout<<emp[i].age;

gotoXY(66,11);

cout<<emp[i].grosspay;

found = true;

}

//else

//

}

if(!found)

{

gotoXY(26,11);

cout<<"No records Found...!!!\a";

}

getch();

}

void editmenu()

{

//system("cls");

borderNoDelay();

gotoXY(28,4);

printf("Edit An Entry");

gotoXY(10,6);

cout<<"What Do You Want To edit";

gotoXY(12,8);

cout<<"n ---------> Name ";

gotoXY(12,9);

cout<<"c ---------> Code ";

gotoXY(12,10);

cout<<"d ---------> Designation";

gotoXY(12,11);

cout<<"e ---------> Experience ";

gotoXY(12,12);

cout<<"a ---------> Age";

gotoXY(12,13);

cout<<"s ---------> Salary";

gotoXY(12,14);

cout<<"q ---------> QUIT";

gotoXY(10,16);

cout<<"Enter Choice ---->>> ";

}

void editname(int i)

{

gotoXY(10,18);

cout<<"Enter New Name-----> ";

cin>>emp[i].name;

}

void editcode(int i)

{

gotoXY(10,18);

cout<<"Enter New Job Code-----> ";

cin>>emp[i].code;

}

void editdes(int i)

{

gotoXY(10,18);

cout<<"enter new designation-----> ";

cin>>emp[i].designation;

}

void editexp(int i)

{

gotoXY(10,18);

cout<<"Enter new Years of Experience";

cin>>emp[i].exp;

}

void editage(int i)

{

gotoXY(10,18);

cout<<"Enter new Age ";

cin>>emp[i].age;

}

void editsalary(int i)

{

int sal,PF,TAX,HRA,MealA,MedicalA,TransportA,LoanBal=emp[i].LoanBalance,LoanDeb,DA;

char loan;

gotoXY(10,18);

cout<<"Enter new Salary ";

cin>>sal;

DA = 1.20 \* sal;

TAX = 0.04 \* sal;

PF = 0.12 \* sal;

HRA = 4000;

MealA = 300;

MedicalA = 300;

TransportA = 300;

LoanDeb = sal \* 0.09;

if(LoanDeb > LoanBal)

{

LoanBal = 0;

LoanDeb = LoanBal;

}

emp[i].salary = sal;

emp[i].tax=TAX;

emp[i].PF = PF;

emp[i].HRA = HRA;

emp[i].MealAllowance = MealA;

emp[i].MedicalAllowance = MedicalA;

emp[i].TransportAllowance = TransportA;

emp[i].LoanBalance = LoanBal;

emp[i].AnyLoan = loan;

emp[i].LoanDebit = LoanDeb;

emp[i].grosspay = (sal+MealA+MedicalA+TransportA+HRA+DA)-(PF+TAX+LoanDeb) ;

}

void edit()

{

//system("cls");

borderNoDelay();

int jobcode;

gotoXY(28,4);

printf("Edit a Record");

int i;

char option;

gotoXY(10,6);

cout<<"Enter the jobcode To Edit : ";

cin>>jobcode;

editmenu();

for(i=0;i<=num-1;i++)

{

if(emp[i].code==jobcode)

{

while((option=cin.get())!='q')

{

switch(option)

{

case 'n':

editname(i);

break;

case 'c':

editcode(i);

break;

case 'd':

editdes(i);

break;

case 'e':

editexp(i);

break;

case 'a':

editage(i);

break;

case 's':

editsalary(i);

break;

}

editmenu();

}

}

}

}

void displayPayslip()

{

system("cls");

borderNoDelay();

gotoXY(10,4);

int code,i;

cout<<"Enter Employee Job Code :";

cin>>code;

for(i=0;i<=num-1;i++)

{

if(emp[i].code==code)

{

gotoXY(12,6);

cout<<"Name : "<<emp[i].name;

gotoXY(45,6);

cout<<"Working Hours : "<<emp[i].workingHours<<" Hrs";

gotoXY(6,8);

cout<<"Earnings :-";

gotoXY(8,10);

cout<<"Basic Pay : "<<emp[i].salary<<endl;

gotoXY(8,12);

cout<<"HRA(27% of Basic): "<<emp[i].HRA<<endl;

gotoXY(8,14);

cout<<"DA (120% of Basic):"<<emp[i].DA;

gotoXY(8,16);

cout<<"Meal Allowance : "<<emp[i].MealAllowance<<endl;

gotoXY(8,18);

cout<<"Medical Alowance : "<<emp[i].MedicalAllowance<<endl;

gotoXY(8,20);

cout<<"Transport Allowance : "<<emp[i].TransportAllowance<<endl;

gotoXY(40,8);

cout<<"Deductions :- "<<endl<<endl;

gotoXY(42,10);

cout<<"PF : "<<emp[i].PF<<endl;

gotoXY(42,12);

cout<<"Tax : "<<emp[i].tax<<endl;

gotoXY(42,14);

int l = emp[i].AnyLoan;

char l2 = toupper(l);

cout<<"Loan Status : "<<l2<<endl;

gotoXY(42,16);

cout<<"Loan Debit This Month : "<<emp[i].LoanDebit<<endl;

gotoXY(42,18);

cout<<"Loan Balance : "<<emp[i].LoanBalance<<endl;

gotoXY(32,22);

cout<<"Total Gross Pay : "<<emp[i].grosspay;

}

}

getch();

}

int main()

{

setWindowSize();

border();

intro();

loading();

loginFrame();

login();

menu();

getrecords();

char option;

if(emp[0].code==0 && isFilePresent())

num--;

while(1)

{

option=getch();

switch(option)

{

case 'l':

list();

break;

case 'i':

insert();

break;

case 'd':

deletes();

break;

case 'e':

edit();

break;

case 's':

search();

break;

case 'p':

displayPayslip();

break;

case 'q':

saverecords();

exit(0);

}

menu();

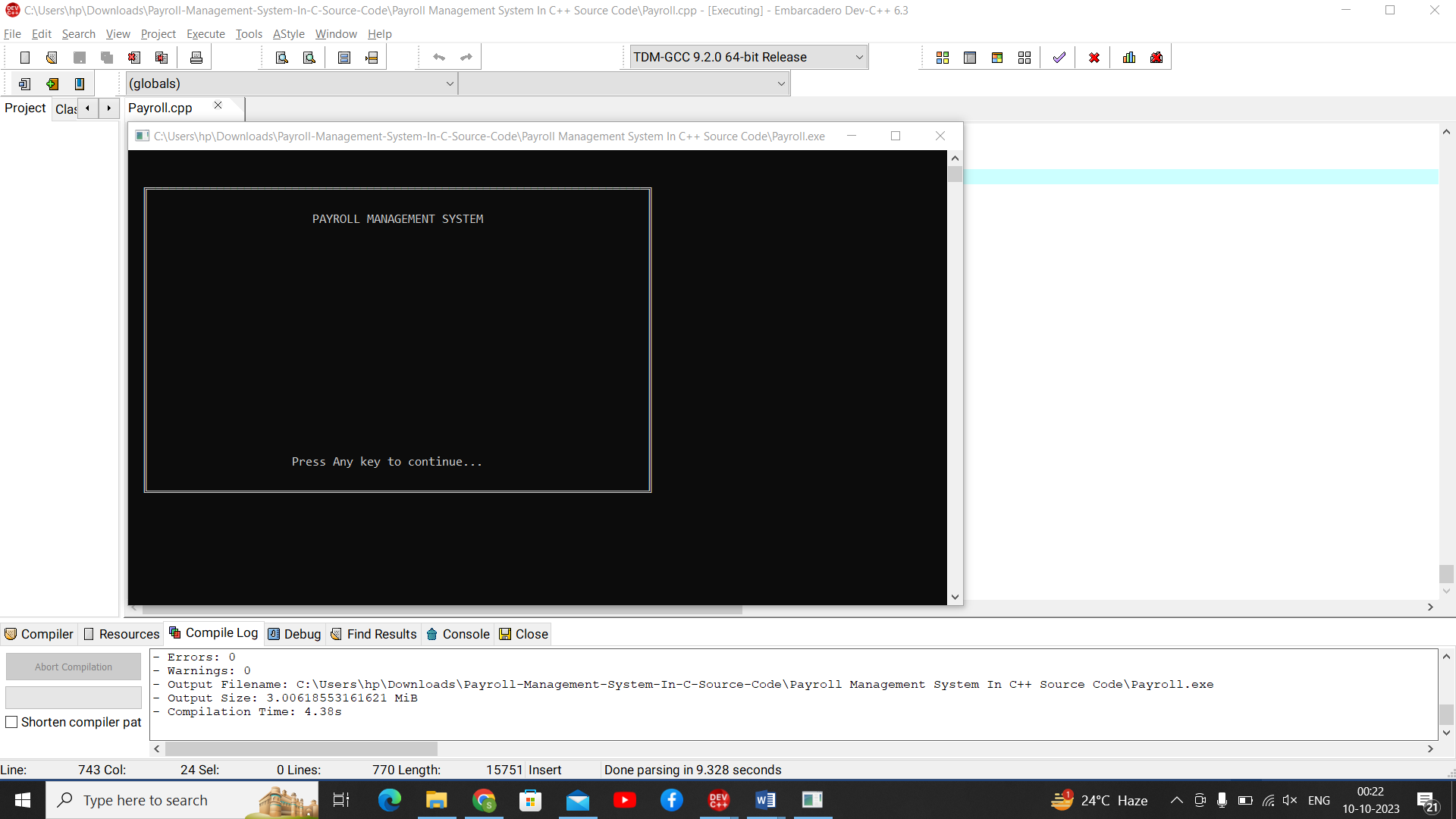
}

return 0;

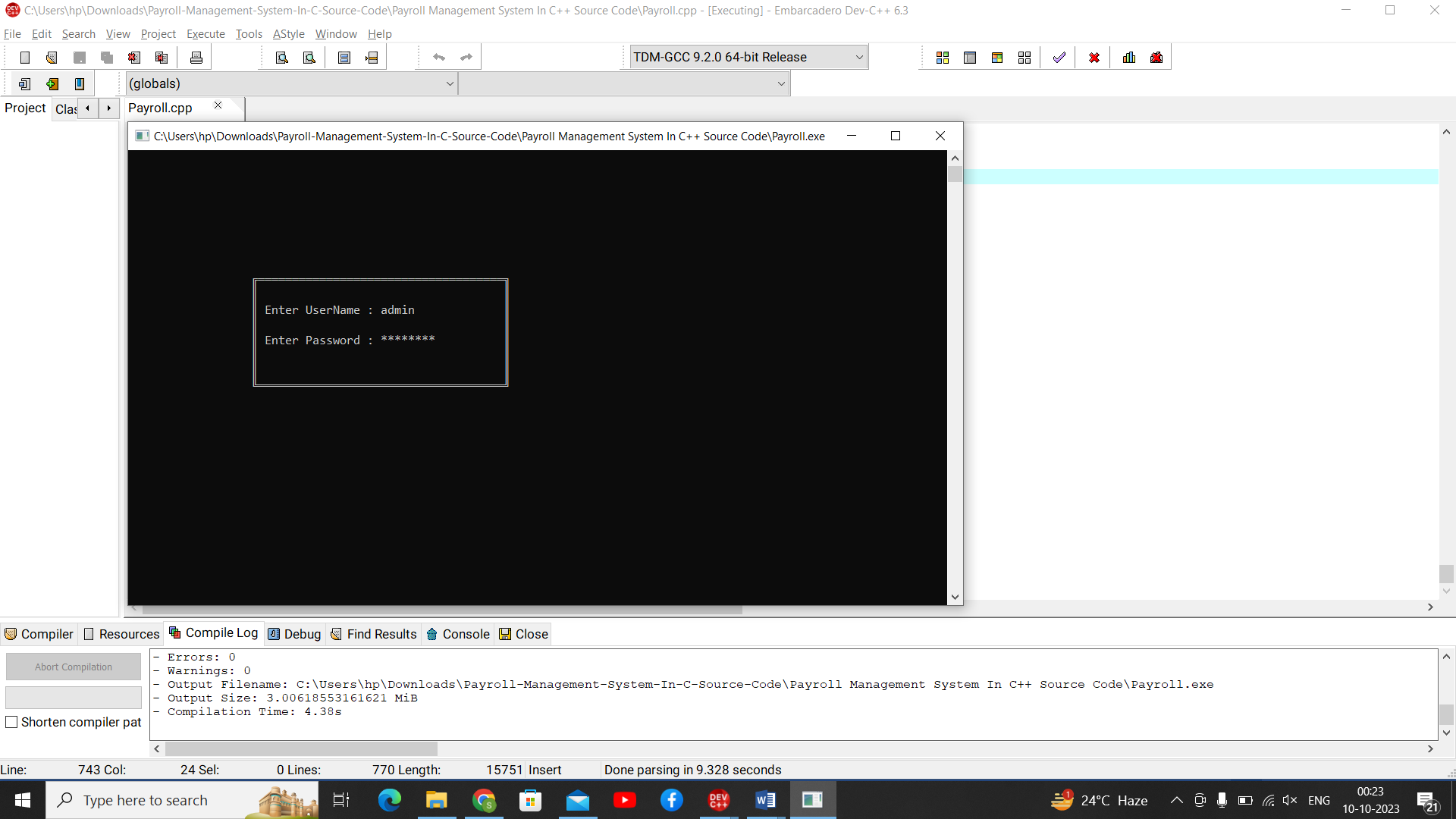
}

**Screenshots**

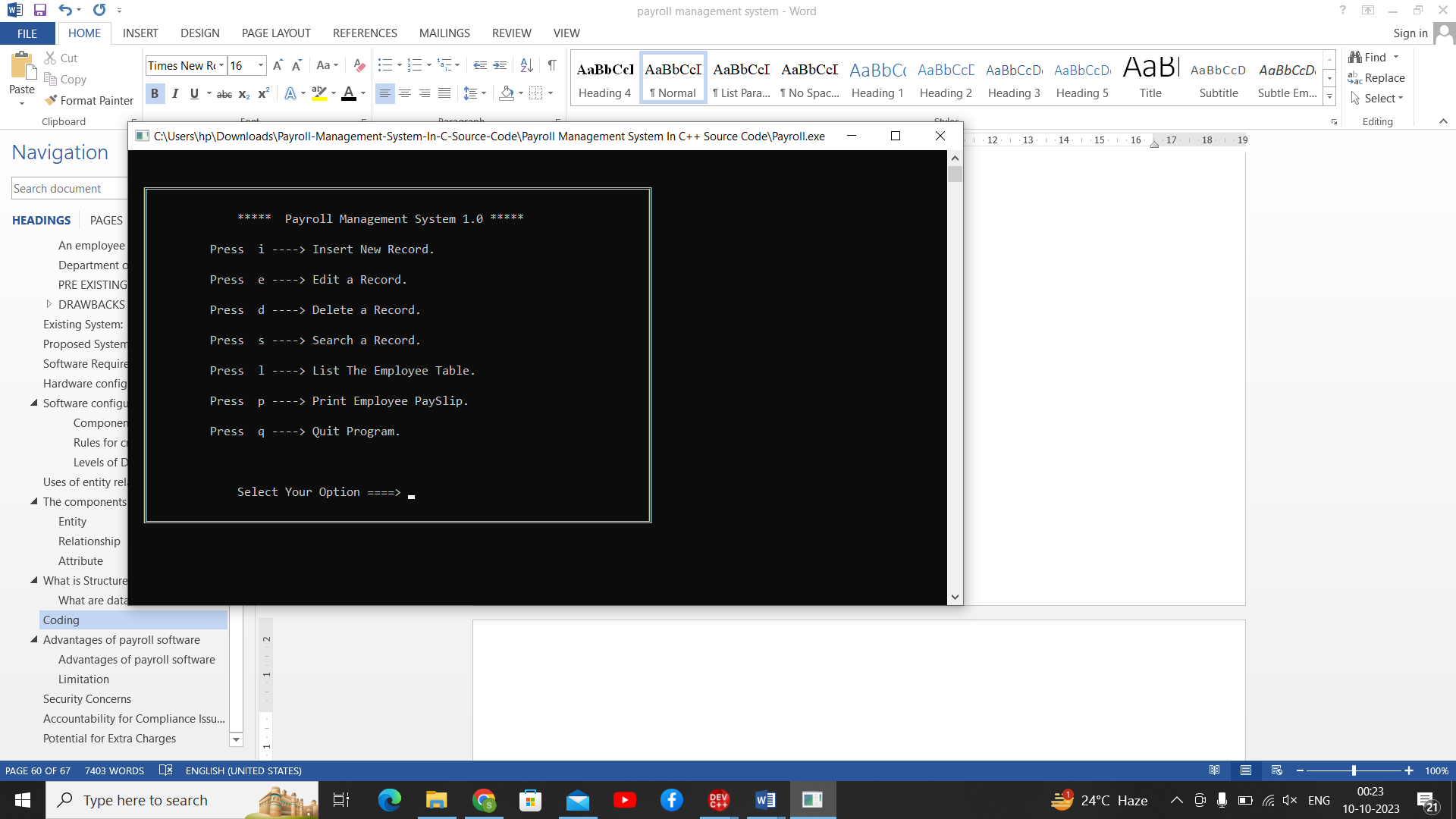
**Welcome page**



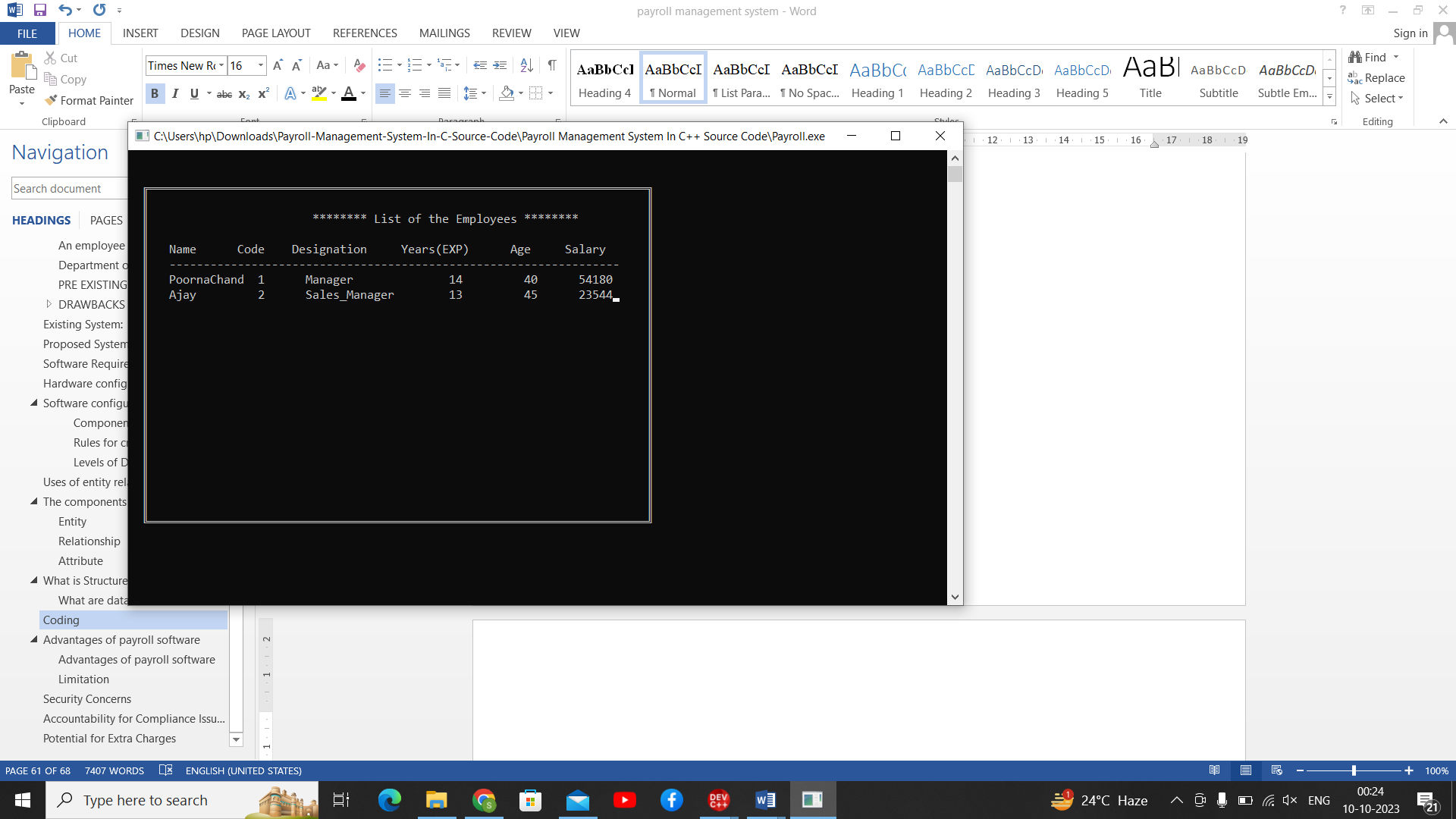
**Admin Validation**



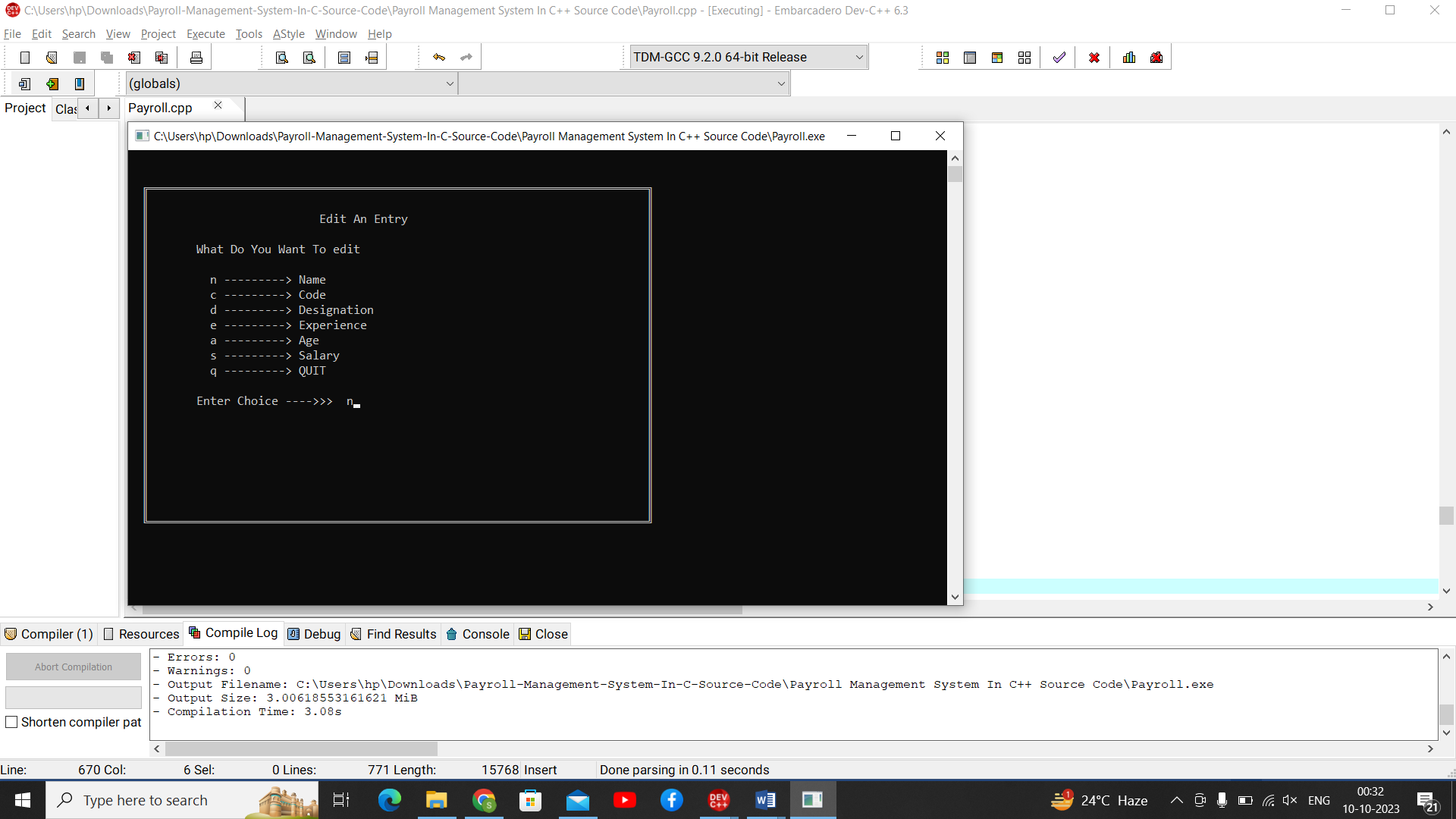
**Home Page for Admin**



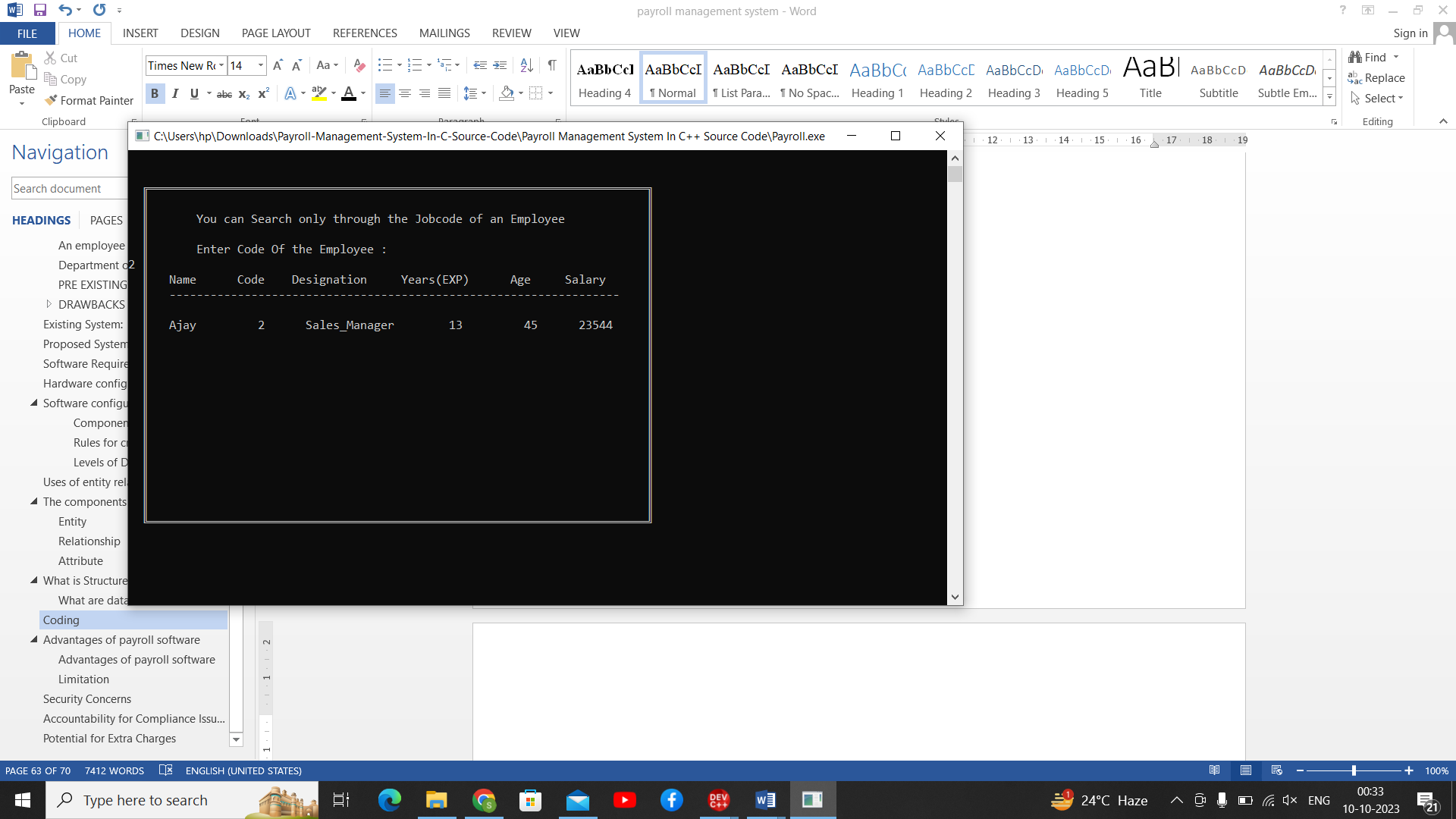
**List Of Employee**



**Edit Employee**



**Search Enployee**



## **Advantages of payroll software**

**Payroll** is a business-critical operation for every organisation. You must pay your staff accurately and on time to avoid low morale, poor performance and possibly even reputational and legal difficulties. A good computerised payroll system can help you carry out your pay run with greater speed, efficiency and confidence.

### **Advantages of payroll software**

Many businesses choose to use payroll software over manual processing, as it can help them to:

* work out payroll calculations and deductions quicker
* generate accurate payslips
* calculate bonuses, expenses, holiday pay, etc with minimum effort
* send returns to HMRC and print P45, P60 and other forms for employees
* automate certain tasks, such as year-end reporting
* reduce the burden of compliance
* remove the need to understand complex tax legislation
* store data such as payslips and annual reports in a secure, easily accessible system

Payroll software can also provide additional data and analysis to make payroll information more useful to your business. See [**do you need payroll software**](https://www.nibusinessinfo.co.uk/content/do-you-need-payroll-software).

**Linking payroll software with time recording**  
You can link payroll to timesheet systems that record employee attendance or time worked. This allows you to automatically transfer information about hours worked into the payroll system and make payroll calculations much simpler.

**Using payroll software for reporting**  
By using basic payroll data, together with data on attendance and hours worked, payroll systems can provide a wealth of **reports**. This allows in-depth analysis of staff costs for the business as a whole, across departments and even individual jobs and contracts.

**Storing personnel records**  
Most organisations will also keep other data about employees, such as records of annual leave. By getting payroll systems that record these additional types of information you can avoid the need for a separate software package.

**Using payroll system to plan future costs**  
As payroll packages can provide forecasts, you can use these to plan staff costs and budgets by entering hypothetical numbers to see the exact total cost of an employee.

Depending on your business needs, you may find other [**payroll software features**](https://www.nibusinessinfo.co.uk/content/payroll-software-features) more useful

### **Limitation**

Every company has slightly different methods for [managing payroll](https://www.hrpayrollsystems.net/how-to-manage-payroll/). Many companies are now utilizing  online payroll services to assist with their [payroll](https://www.hrpayrollsystems.net/payroll/), but some drawbacks arise when using these services. If you’re considering a switch from a manual [payroll system](https://www.hrpayrollsystems.net/payroll-systems/) to an online one, or have recently started your business and are weighing your options, make sure to factor in the following potential disadvantages.

## **Security Concerns**

Since online payroll services rely on a third party to process and store payroll information, sensitive information about employees and the hours that they work doesn’t remain solely within the walls of one’s workplace. In an ideal world, all online payroll services would remain very secure and be guaranteed against leaks, but this may not always be the case. Before working with a vendor, be sure to ask about security measures.

Most online payroll services perform many functions automatically, with just a few areas that rely on authorization. While this mitigates the potential for human errors, there may still be imperfections caused by things like an employee forgetting to clock in for several hours. When these errors occur, it’s generally necessary to call customer service – which may be inconsistent and inefficient depending on the vendor.

## **Accountability for Compliance Issues**

[Compliance with laws](https://www.hrpayrollsystems.net/hris/payroll-compliance/) is ultimately the responsibility of a company. For example, if an online payroll service is used and compliance errors are found when reports are reviewed or taxes are filed, the company and not the vendor is responsible for those errors and any penalties incurred because of them. If the errors are due to a glitch in the system or a failure to communicate, it can be frustrating and potentially costly.

## **Potential for Extra Charges**

If you’re not converting all your HR processes to an online platform, you may find yourself paying for some unnecessary services. Some payroll vendors package their offerings, potentially giving you access to services currently handled by other third parties or that you manage in house using other systems. There may also be extra charges for things like report generation or support services.

If you are considering online payroll services, be sure to read the fine print and get answers to all your questions, especially regarding fees. The best vendors will walk you through every aspect of their pricing scheme and make sure that you understand and agree with everything. If you find that there are more features than needed, you may be better suited to partner with a vendor that allows you to customize your services.

[Online payroll services can be helpful](https://www.hrpayrollsystems.net/advantages-using-online-payroll-services/), but there are disadvantages that you should be aware of before committing to a specific vendor or system. If you decide to incorporate online payroll services into your company, [working with experts](https://www.hrpayrollsystems.net/vendor-match/) to select the right system for your company may help you avoid some of the potential disadvantages.

**Conclusion**

With the theoretical inclination of our syllabus it becomes very essential to take the utmost advantage of any opportunity of gaining practical experience that comes along. The construction of this Minor Project **“**PAYROLL MANAGEMENT SYSTEM” was one of these opportunities. It gave us the requisite practical knowledge to supplement the already taught theoretical concepts thus making us more competent as a computer engineer.

The project from a personal point of view also helped us in understanding the following aspects of project development:

* The planning that goes into implementing a project.
* The importance of proper planning and an organized methodology.
* The key element of team spirit and co-ordination in a successful project

The project also provided us the opportunity of interacting with our teachers and to gain from their vast experience.

**Future Scope for Modification**

Although we tried to be as thorough as possible in the implementation of our project but there were a large number of features we could not implement due to want of time or for other reasons.

. **References:**

1. **C++ Documentation** (cppreference.com): This is an excellent online resource with comprehensive documentation of C++ features, including code examples and explanations. It's often considered the go-to reference for C++.

Website: [cppreference.com](https://en.cppreference.com/)

1. **C++ Standard**: The official C++ Standard provides the formal specification of the C++ language and libraries. The latest version is C++20 (as of my last update), but newer standards may have been released since then.

Website: [ISO C++ Standards Committee](https://isocpp.org/std/the-standard)

1. **Books**:
   * "The C++ Programming Language" by Bjarne Stroustrup: Written by the creator of C++, this book is considered a classic and provides in-depth insights into C++.
   * "Effective C++" by Scott Meyers: This book offers practical advice and best practices for C++ programming.
   * "Accelerated C++" by Andrew Koenig and Barbara E. Moo: A good introductory book for learning C++ programming.