1. **OBJECTIVE OF APPLICATION**

The main objective of the application was to experience and understand real life situations in industrial organizations and their related environments and accelerating the learning process of how my knowledge could be used in a realistic way. In addition to this, this project also made me understand the formal and informal relationships in an industrial organization so as to promote favorable human relations and teamwork. Besides, it provides me the exposure to practice and apply the acquired knowledge ‘hands – on’, in the working environment. After some theory and practical sessions, I was able to make the project on "STUDENT DATABASE MANAGEMENT SYSTEM" in C Language and a heavy use of data structure concepts. By this, I was able to apply the theoretical knowledge in my project.Thus by completing this project, I learned how to make a project and how it works in an organized way with limited time, accuracy and simultaneously satisfying customer needs.

**2. BASIC CONCEPTS**

**2.1 INTRODUCTION TO C**

**2.1.1 WHAT IS C?**

In [computing](http://en.wikipedia.org/wiki/Computing), C is a general-purpose [programming language](http://en.wikipedia.org/wiki/Programming_language) initially developed by [Dennis Ritchie](http://en.wikipedia.org/wiki/Dennis_Ritchie) between 1969 and 1973 at [AT&T Bell Labs](http://en.wikipedia.org/wiki/AT%26T_Bell_Labs). Like most [imperative languages](http://en.wikipedia.org/wiki/Imperative_language) in the [ALGOL](http://en.wikipedia.org/wiki/ALGOL) tradition, C has facilities for [structured programming](http://en.wikipedia.org/wiki/Structured_programming) and allows [lexical variable scope](http://en.wikipedia.org/wiki/Lexical_scope) and [recursion](http://en.wikipedia.org/wiki/Recursion_(computer_science)), while a [static type system](http://en.wikipedia.org/wiki/Static_type_system) prevents many unintended operations. Its design provides constructs that map efficiently to typical [machine instructions](http://en.wikipedia.org/wiki/Machine_instruction), and therefore it has found lasting use in applications that had formerly been coded in [assembly language](http://en.wikipedia.org/wiki/Assembly_language), most notably [system software](http://en.wikipedia.org/wiki/System_software) like the [Unix](http://en.wikipedia.org/wiki/Unix) computer [operating system](http://en.wikipedia.org/wiki/Operating_system).

C is one of the most widely used programming languages of all time and C compilers are available for the majority of available [computer architectures](http://en.wikipedia.org/wiki/Computer_architectures) and [operating systems](http://en.wikipedia.org/wiki/Operating_systems).

Many later languages have borrowed directly or indirectly from C, including [C#](http://en.wikipedia.org/wiki/C_Sharp_(programming_language)), [D](http://en.wikipedia.org/wiki/D_(programming_language)), [Go](http://en.wikipedia.org/wiki/Go_(programming_language)), [Rust](http://en.wikipedia.org/wiki/Rust_(programming_language)), [Java](http://en.wikipedia.org/wiki/Java_(programming_language)), [JavaScript](http://en.wikipedia.org/wiki/JavaScript), [Limbo](http://en.wikipedia.org/wiki/Limbo_(programming_language)), [LPC](http://en.wikipedia.org/wiki/LPC_(programming_language)),[Perl](http://en.wikipedia.org/wiki/Perl), [PHP](http://en.wikipedia.org/wiki/PHP), [Python](http://en.wikipedia.org/wiki/Python_(programming_language)), [Verilog](http://en.wikipedia.org/wiki/Verilog) (hardware description language) and Unix's [C shell](http://en.wikipedia.org/wiki/C_shell). The most pervasive influence on these languages (excluding Python) has been [syntactical](http://en.wikipedia.org/wiki/Syntax_(programming_languages)), and they tend to combine the recognizable expression and statement [syntax of C](http://en.wikipedia.org/wiki/C_syntax) with underlying type systems, data models, and semantics that can be radically different. [C++](http://en.wikipedia.org/wiki/C%2B%2B) and Objective-C started as compilers that generated C code; C++ is currently [nearly a superset of C](http://en.wikipedia.org/wiki/Compatibility_of_C_and_C%2B%2B), while [Objective-C](http://en.wikipedia.org/wiki/Objective-C) is a [strict superset](http://en.wikipedia.org/wiki/Objective-C#Syntax) of C.

C is an [imperative](http://en.wikipedia.org/wiki/Imperative_programming) ([procedural](http://en.wikipedia.org/wiki/Procedural_programming)) language. It was designed to be compiled using a relatively straightforward [compiler](http://en.wikipedia.org/wiki/Compiler), to provide low-level access to memory, to provide language constructs that map efficiently to machine instructions, and to require minimal [run-time support](http://en.wikipedia.org/wiki/Run-time_system). C was therefore useful for many applications that had formerly been coded in [assembly language](http://en.wikipedia.org/wiki/Assembly_language), such as in [system programming](http://en.wikipedia.org/wiki/System_programming).

Despite its low-level capabilities, the language was designed to encourage [cross-platform](http://en.wikipedia.org/wiki/Cross-platform) programming. A standards-compliant and [portably](http://en.wikipedia.org/wiki/Porting) written C program can be compiled for a very wide variety of computer platforms and operating systems with few changes to its source code. The language has become available on a very wide range of platforms, from embedded [microcontrollers](http://en.wikipedia.org/wiki/Microcontroller) to [supercomputers](http://en.wikipedia.org/wiki/Supercomputer).

**2.1.2 HISTORY**

The origin of C is closely tied to the development of the [UNIX](http://en.wikipedia.org/wiki/Unix) operating system, originally implemented in assembly language on a [PDP-7](http://en.wikipedia.org/wiki/PDP-7) by Ritchie and Thompson, incorporating several ideas from colleagues. Eventually they decided to port the operating system to a [PDP-11](http://en.wikipedia.org/wiki/PDP-11). The original PDP-11 version of UNIX was developed in [assembly language](http://en.wikipedia.org/wiki/Assembly_language). The developers were considering rewriting the system using the [B language](http://en.wikipedia.org/wiki/B_language). However B's inability to take advantage of some of the PDP-11's features, notably [byte](http://en.wikipedia.org/wiki/Byte) addressability, led to the development of C.

The initial development of C occurred at [AT&T Bell Labs](http://en.wikipedia.org/wiki/AT%26T_Bell_Labs) between 1969 and 1973; according to Ritchie, the most creative period occurred in 1972. At that year a great part of UNIX was rewritten in C. By 1973, with the addition of struct types, the C language had become powerful enough that most of the [UNIX](http://en.wikipedia.org/wiki/Unix) [kernel](http://en.wikipedia.org/wiki/Kernel_(computing)) was now in C. The new language was named "C" because its features were derived from "[B](http://en.wikipedia.org/wiki/B_(programming_language))", which according to [Ken Thompson](http://en.wikipedia.org/wiki/Ken_Thompson_(computer_programmer)) was a stripped-down version of the [BCPL](http://en.wikipedia.org/wiki/BCPL) programming language.

UNIX was one of the first operating system kernels implemented in a language other than [assembly](http://en.wikipedia.org/wiki/Assembly_language). (Earlier instances include the [Multics](http://en.wikipedia.org/wiki/Multics) system (written in [PL/I](http://en.wikipedia.org/wiki/PL/I)), and MCP ([Master Control Program](http://en.wikipedia.org/wiki/MCP_(Burroughs_Large_Systems))) for the [Burroughs B5000](http://en.wikipedia.org/wiki/Burroughs_large_systems) written in [ALGOL](http://en.wikipedia.org/wiki/ALGOL) in 1961.) Circa 1977, further changes to the language were made by Ritchie and [Stephen C. Johnson](http://en.wikipedia.org/wiki/Stephen_C._Johnson) to facilitate portability of the UNIX operating system. Johnson's [Portable C Compiler](http://en.wikipedia.org/wiki/Portable_C_Compiler) served as the basis for several implementations of C on new platforms.

In 1978, [Brian Kernighan](http://en.wikipedia.org/wiki/Brian_Kernighan) and [Dennis Ritchie](http://en.wikipedia.org/wiki/Dennis_Ritchie) published the first edition of [The C Programming Language](http://en.wikipedia.org/wiki/The_C_Programming_Language). This book, known to C programmers as "K&R", served for many years as an informal [specification](http://en.wikipedia.org/wiki/Specification) of the language. The version of C that it describes is commonly referred to as K&R C. The second edition of the book covers the later [ANSI C](http://en.wikipedia.org/wiki/ANSI_C) standard.

**2.1.3 SYNTAX**

C has a [formal grammar](http://en.wikipedia.org/wiki/Formal_grammar) specified by the C standard.[[18]](http://en.wikipedia.org/wiki/C_(programming_language)#cite_note-h.26s5e-18) Unlike languages such as [FORTRAN 77](http://en.wikipedia.org/wiki/FORTRAN_77), C source code is [free-form](http://en.wikipedia.org/wiki/Free-form_language) which allows arbitrary use of whitespace to format code, rather than column-based or text-line-based restrictions. Comments may appear either between the delimiters /\* and \*/ or following // until the end of the line. Comments delimited by /\* and \*/ do not nest, and these sequences of characters are not interpreted as comment delimiters if they appear inside [string](http://en.wikipedia.org/wiki/String_literal) or character literals.[[19]](http://en.wikipedia.org/wiki/C_(programming_language)#cite_note-KandR1-19)

C source files contain declarations and function definitions. Function definitions, in turn, contain declarations and [statements](http://en.wikipedia.org/wiki/Statement_(programming)). Declarations either define new types using keywords such as struct, union, and enum, or assign types to and perhaps reserve storage for new variables, usually by writing the type followed by the variable name. Keywords such as char and int specify built-in types. Sections of code are enclosed in braces ({ and }, sometimes called "curly brackets") to limit the scope of declarations and to act as a single statement for control structures.

As an imperative language, C uses statements to specify actions. The most common statement is an expression statement, consisting of an expression to be evaluated, followed by a semicolon; as a side effect of the evaluation, functions may be [called](http://en.wikipedia.org/wiki/Procedure_call) and variables may be [assigned](http://en.wikipedia.org/wiki/Assignment_(computer_science)) new values. To modify the normal sequential execution of statements, C provides several control-flow statements identified by reserved keywords. [Structured programming](http://en.wikipedia.org/wiki/Structured_programming) is supported by if-else conditional execution and by do-while, while and for iterative execution (looping). There is also a non-structured goto statement which branches directly to the designated [label](http://en.wikipedia.org/wiki/Label_(programming_language)) within the function. switch selects a case to be executed based on the value of an integer expression.

**C First Program**

#include <stdio.h>

int main(void)

{

printf("hello, world\n");

}

About C language, it is very important to keep in mind the following points.

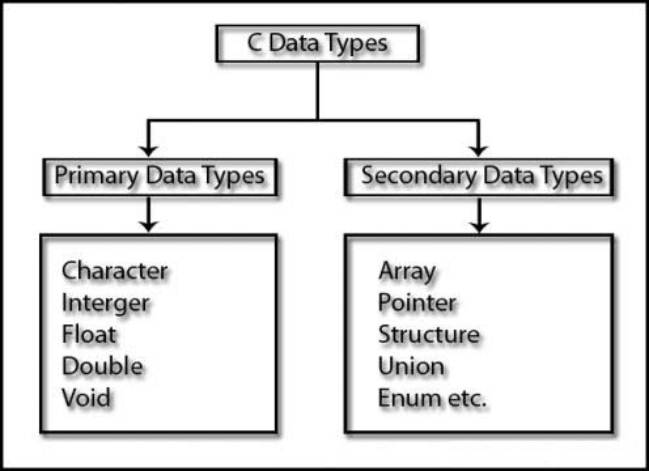
* Easy to learn
* Structured language
* It produces efficient programs.
* It can handle low-level activities.
* It can be compiled on a variety of computer platforms.

**2.1.4 DATA TYPES**

Variables are nothing but reserved memory locations to store values. This means that when you create a variable you reserve some space in memory. Based on the data type of a variable, the operating system allocates memory and decides what can be stored in the reserved memory. Therefore, by assigning different data types to variables, you can store integers, decimals, or characters in these variables.

There are two data types available in C:

1. Primitive Data Types
2. Secondary Data Types

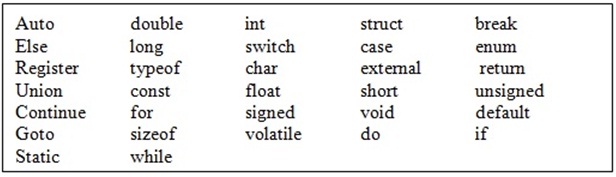


## http://www.studytonight.com/c/images/primary-data-types.jpgPrimitive Data Types - The C language provides many basic types. Most of them are formed from one of the four basic arithmetic type specifiers in C (char, int, float, double), and optional specifiers (signed, unsigned, long, short). All available basic arithmetic types are listed below:

## Secondary Data Types – They are also known as user defined data types. These include array, pointer, structure, union and enumeration.

**2.1.5 KEYWORDS**

Keywords are special words which have their own predefined meanings. The following list shows the reserved words in C. These reserved words may not be used as constant or variable or any other identifier names.



**2.1.6 FUNCTION**

A function is a group of statements that together perform a task. Every C program has at least one function, which is main(), and all the most trivial programs can define additional functions. A function is known with various names like a method or a sub-routine or a procedure, etc.

You can divide up your code into separate functions. How you divide up your code among different functions is up to you, but logically the division usually is so each function performs a specific task.

A function declaration tells the compiler about a function's name, return type, and parameters. A function definition provides the actual body of the function.

The C standard library provides numerous built-in functions that your program can call. For example, function strcat() to concatenate two strings, function memcpy() to copy one memory location to another location and many more functions.

The general form of a function definition in C programming language is as follows:

return\_type function\_name( parameter list )

{

body of the function

}

A function definition in C programming language consists of a function header and a function body. Here are all the parts of a function:

* **Return Type**: A function may return a value. The return\_type is the data type of the value the function returns. Some functions perform the desired operations without returning a value. In this case, the return\_type is the keyword void.
* **Function Name:** This is the actual name of the function. The function name and the parameter list together constitute the function signature.
* **Parameters**: A parameter is like a placeholder. When a function is invoked, you pass a value to the parameter. This value is referred to as actual parameter or argument. The parameter list refers to the type, order, and number of the parameters of a function. Parameters are optional; that is, a function may contain no parameters.
* **Function Body**: The function body contains a collection of statements that define what the function does.

**2.1.7 USES**

C was initially used for system development work, in particular the programs that make-up the operating system. C was adopted as a system development language because it produces code that runs nearly as fast as code written in assembly language. Some examples of the use of C might be:

* Operating Systems
* Language Compilers
* Assemblers
* Text Editors
* Print Spoolers
* Network Drivers
* Modern Programs
* Databases
* Language Interpreters
* Utilities

**2.2 INTRODUCTION TO AVL TREE**

In [computer science](http://en.wikipedia.org/wiki/Computer_science), an AVL tree (Adelson-Velskii and Landis' tree, named after the inventors) is a [self-balancing binary search tree](http://en.wikipedia.org/wiki/Self-balancing_binary_search_tree), and it was the first such [data structure](http://en.wikipedia.org/wiki/Data_structure) to be invented. In an AVL tree, the [heights](http://en.wikipedia.org/wiki/Tree_height) of the two [child](http://en.wikipedia.org/wiki/Child_node) subtrees of any node differ by at most one; if at any time they differ by more than one, rebalancing is done to restore this property. Lookup, insertion, and deletion all take [O](http://en.wikipedia.org/wiki/Big_O_notation)(log n) time in both the average and worst cases, where n is the number of nodes in the tree prior to the operation. Insertions and deletions may require the tree to be rebalanced by one or more [tree rotations](http://en.wikipedia.org/wiki/Tree_rotation).

The AVL tree is named after its two [Soviet](http://en.wikipedia.org/wiki/Soviet_Union) inventors, [G. M. Adelson-Velskii](http://en.wikipedia.org/wiki/Georgii_Adelson-Velskii) and [E. M. Landis](http://en.wikipedia.org/wiki/Yevgeniy_Landis), who published it in their 1962 paper "An algorithm for the organization of information".

AVL trees are often compared with [red-black trees](http://en.wikipedia.org/wiki/Red-black_tree) because both support the same set of operations and take [O](http://en.wikipedia.org/wiki/Big_O_notation)(log n) time for the basic operations. For lookup-intensive applications, AVL trees are faster than red-black trees because they are more rigidly balanced

**2.3 INTRODUCTION TO DATABASE MANAGEMENT SYSTEM**

A database is a collection of related files that are usually integrated, linked or cross-referenced to one another. The advantage of a database is that data and records contained in different files can be easily organized and retrieved using specialized database management software called a database management system (DBMS) or database manager. A database is an organized collection of data. Database management systems (DBMS) are specially designed applications that interact with the user, other applications, and the database itself to capture and analyze data. A general-purpose database management system (DBMS) is a software system designed to allow the definition, creation, querying, update, and administration of databases. It is to be noted that in this project, database is not used directly. Rather, the concepts of database are implemented through file handling. The interactions catered for by most existing DBMS fall into four main groups:

* Data definition- Defining new data structures for a database, removing data structures from the database, modifying the structure of existing data.
* Update- Inserting, modifying, and deleting data.
* Retrieval- Obtaining information either for end-user queries and reports or for processing by applications.
* Administration- Registering and monitoring users, enforcing data security, monitoring performance, maintaining data integrity, dealing with concurrency control, and recovering information if the system fails.

Formally, the term "database" refers to the data itself and supporting data structures. Databases are created to operate large quantities of information by inputting, storing, retrieving, and managing that information. Databases are set up so that one set of software programs provides all users with access to all the data.  
A "database management system" (DBMS) is a suite of computer software providing the interface between users and a database or databases. Because they are so closely related, the term "database" when used casually often refers to both a DBMS and the data it manipulates.

A collection of software for organizing the information in a database that might contain routines for data input, verification, storage, retrieval, and combination. Stands for "Database Management System." In short, a DBMS is a database program. Technically speaking, it is a software system that uses a standard method of cataloging, retrieving, and running queries on data. The DBMS manages incoming data, organizes it, and provides ways for the data to be modified or extracted by users or other programs.

**3. HANDS ON EXPERIENCE**

Under the curriculum of Rajasthan Technical University, a minor project is to be submitted in 7th Semester. Using C fundamentals, File Handling, datastructures, pointers, structures and other concepts, I developed an application named “STUDENT DATABASE MANAGEMENT SYSTEM”. It is very much essential to implement applications like Student Database Management Systems.I learned how to develop a project which I implemented and made on my own. The complete description about the project is given in next sections.With the advent of the development (project) I learnt a lot of new things like how to manage time, decision making and team work.The project started with the theoretical analysis and study which included formation of flow charts and generation of psuedocodes regarding every module of the application. I then proceeded with a project beginning with a thorough study of the given task, considering its various requirements.

**4. PROJECT MODULE**

**4.1 INTRODUCTION**

Student Database Management Systems are the primary systems for operating colleges and schools. It deals with the student-level data collection that allows the Department to collect and analyze more accurate and comprehensive information. It provides capabilities for entering student records, modifying student information and managing many other student-related data needs in a college or university.

I am confident that this software package can be readily used by non-programming personal avoiding human handled chance of error. The user of this project will be the Administrator who will have access to all the student records.

This software is going to describe the way that how to manage a data/information easily. This software will provide the best way of storing and retrieval of information. In this software the technique is used which is easy to understand & implement.

**4.2 SCOPE**

This project is about student information; it provides different student information for different students. The system focus on enhance the measurement and interaction with the database in the most quick, easy, and efficient way.

During the past several decade personnel function has been transformed from a relatively obscure record keeping staff to central and top level management function. There are many factors that have influenced this transformation like technological advances, professionalism, and general recognition of human beings as most important resources.

* A computer based management system is designed to handle all the primary information required to calculate monthly statements.
* Separate database is maintained to handle all the details required for the correct statement calculation and generation.
* This project intends to introduce more user friendliness in the various activities such as record updation, maintenance, and searching.
* The searching of record has been made quite simple as all the details of the customer can be obtained by simply keying in the identification of that customer.
* Similarly, record maintenance and updation can also be accomplished by using the identification of the customer with all the details being automatically generated. These details are also being promptly automatically updated in the master file thus keeping the record absolutely up-to-date.
* The entire information has maintained in the database or Files and whoever wants to retrieve can’t retrieve, only authorization user can retrieve the necessary information which can be easily be accessible from the file.

**4.3 REQUIREMENT GATHERING**

**Software Requirement: -** The following are the software requirements:-

Front End : .exe file

Back End : Binary Files

Platform : Windows 98/2000/XP/7/8/Vista

Linux, Macintosh

**Hardware Requirement: -** The following are the hardware requirements:-

Processor : Intel Pentium 4 class processor or above

RAM : 100 MB

Hard Disk : 50 MB

Monitor : 15” CRT or LCD

Keyboard : Normal or Multimedia

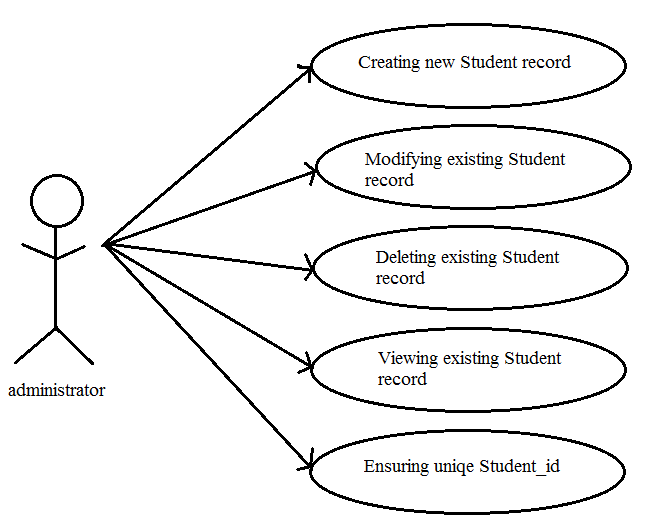
Mouse : Compatible Mouse

**4.4 OBJECTIVE**

The following are the objectives to develop the Student Database management System:-

* An understanding of the needs for and uses of database management systems in business.
* An understanding of the context, phases and techniques for designing and building database information systems in business.
* An understanding of the components of a computerized database information system (application).
* An ability to correctly use the techniques, components and tools of a typical database management system to build a comprehensive database information system (application).
* An ability to design a correct, new database information system for a business functional area and implement the design in any of the commonly used databases.
* An introductory understanding of some advanced topics in, e.g., relational databases and design, user authentication, database administration (security modification, backup and restore, tuning) and file indexing.

**4.5. USE-CASE DIAGRAM**

**4.6 SYSTEM EVOLUTION**

Today in colleges student details are entered manually. The student details in separate records are tedious task. Referring to all these records and updating is needed. There is a chance for more manual errors.

**Problems in existing system:**

* It was less user-friendly.
* It have a lots of manual work (Manual system does not mean that we are working with pen and paper, it also include working on spread sheets and other simple software's)
* It requires more no of employees need to work.
* It was time consuming process.
* The present system was very less secure.

**Solution to these problems:**

The development of the new system contains the following activities, which try to automate the entire process keeping in view of the database integration approach.

* User friendliness is provided in the application with various controls.
* The system makes the overall project management much easier and flexible.
* All the work is done in primary memory and whenever the system is closed or on demand of the user ,the changes are permanently stored in the disk.
* Similarly when existing database (file) is opened ,the primary key of all records are automatically arranged in the form of tree.
* Apart from the actual database , an index file is maintained .In primary memory , it is implemented in the form of **AVL search tree** and **linked list** to ensure efficient insertion, deletion and searching .
* There is no risk of data mismanagement at any level while the project development is under process.
* User Authentication is ensured maintaining **user name** and **password.**
* The possibility of invalid data entry is reduced to a large extend by adding some restrictions during the data entry.

**4.**7 **DATA DICTIONARY**

* **security.dat**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Null** | **Key** |
| user\_id | char[20] | No | --- |
| password | char[20] | Yes | --- |

* **student.dat**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Field name** | **Data type** | **Null** | **Default** | **Key** |
| Student\_rollno | int | No | NOT NULL | PRI |
| Student\_name | char[20] | Yes | NULL | --- |
| Marks | float | Yes | NULL | --- |
| Address | char[30] | Yes | NULL | --- |

* **bintree.dat**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Null** | **Key** |
| roll\_no | int | No | --- |
| Location | int | No | --- |

* **empty.dat**

|  |  |  |  |
| --- | --- | --- | --- |
| **Field name** | **Data type** | **Null** | **Key** |
| Location | int | No | --- |

**4.8 DATA TABLES**

* **security.dat**

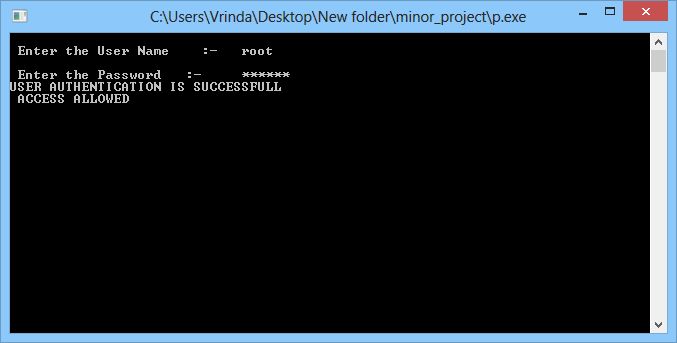
|  |  |
| --- | --- |
| **User\_name** | **password** |
| root | 123456 |

* **student.dat**

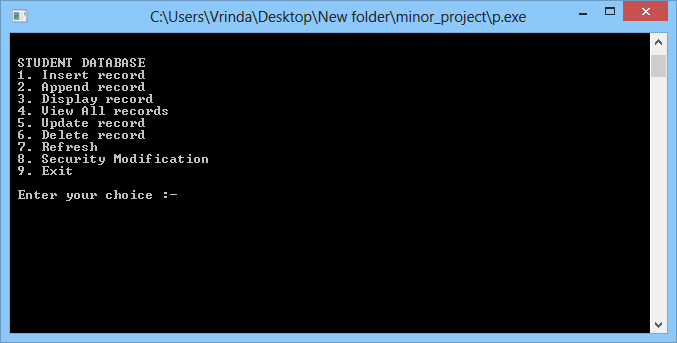
|  |  |  |  |
| --- | --- | --- | --- |
| **Student\_rollno** | **Student\_name** | **Marks** | **Address** |
| 1 | Alex | 65 | aaa |
| 2 | Bobby | 78 | bbb |
| 3 | Carren | 44 | ccc |
| 4 | Dylan | 90 | ddd |
| 5 | Eli | 32 | eee |
| 6 | Fred | 79 | fff |
| 7 | George | 51 | ggg |
| 8 | Harper | 40 | hhh |
| 9 | Ira | 99 | iii |
| 10 | Jerry | 37 | jjj |
| 11 | Kelvin | 96 | kkk |
| 12 | Lisa | 83 | lll |
| 13 | Milley | 61 | mmm |
| 14 | Nancy | 88 | nnn |
| 15 | Owsen | 20 | ooo |

**4 .9 SYSTEM DESIGN**

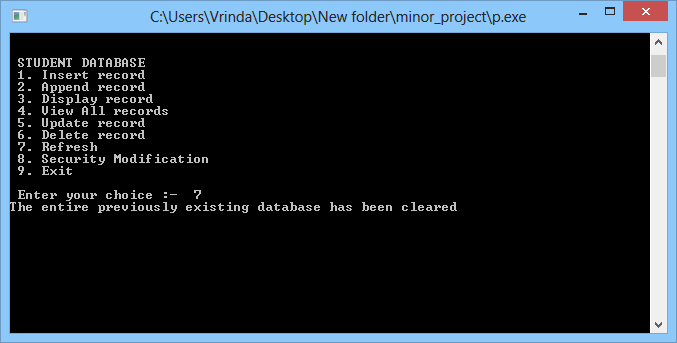
User Authentication:



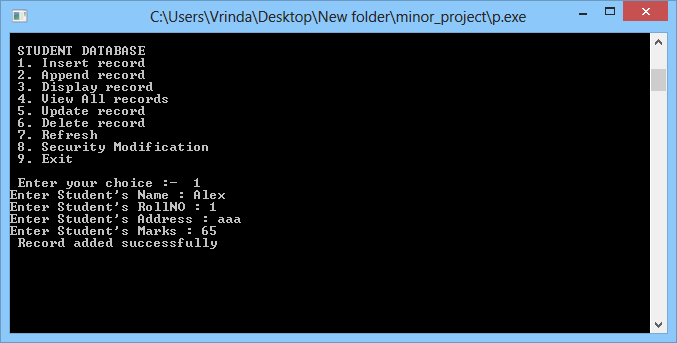
Admin Interface:



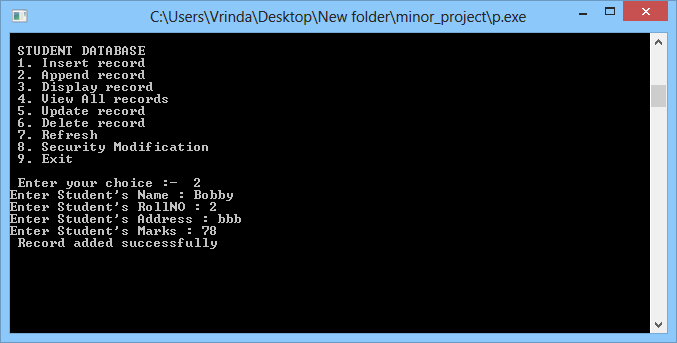
Refresh Option :



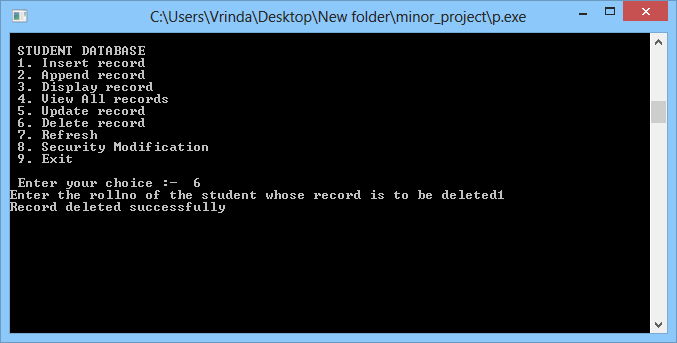
Insert Entry:



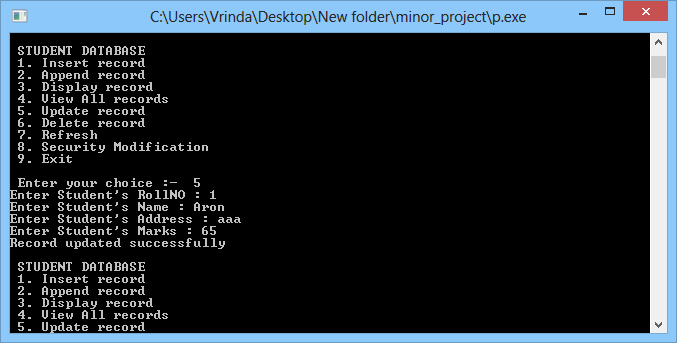
Append Entry:



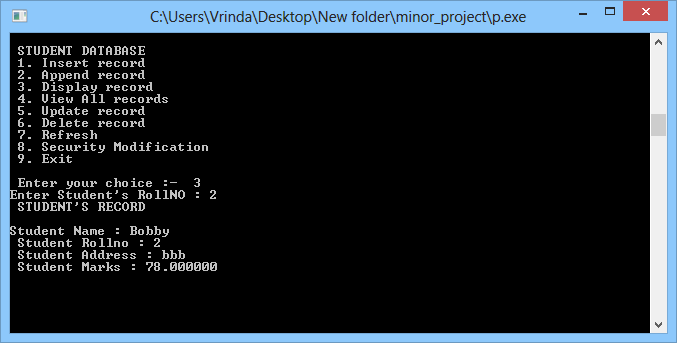
Delete Entry:



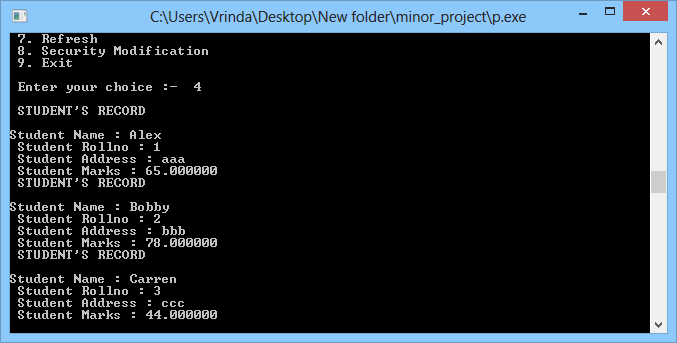
Update Entry:



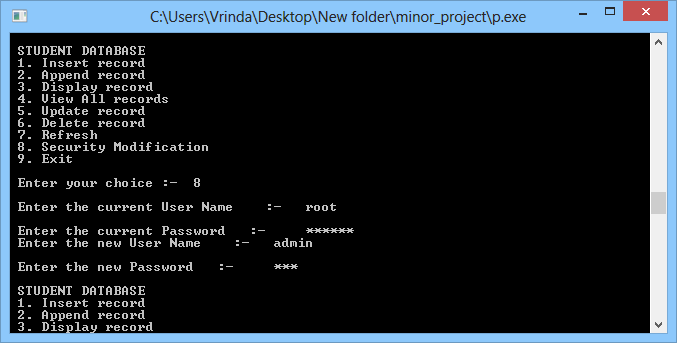
Display Entry:



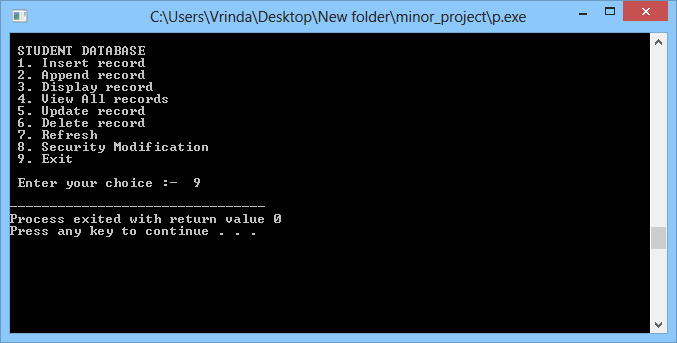
View All Entry:



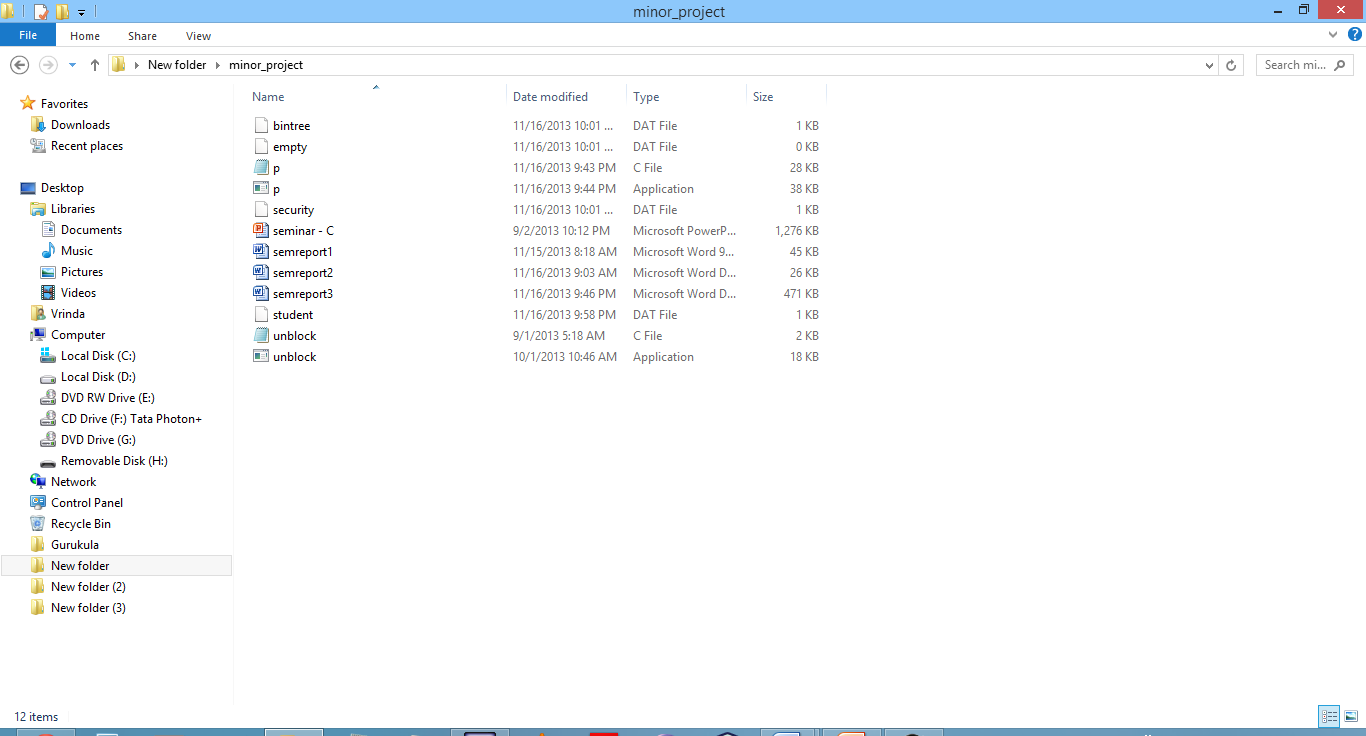
Security Modification:



Exit:



Project Files:



**5. CONCLUSION**

The college can handle full-scale computer and computer related resources. The Student Database management System deals with all the activities done by computer such as insertion, deletion and modification process. All these processes are handled by this application. The system should be user appropriate, easy to use, provide easy recovery of errors and have an overall end user high subjective satisfaction.

The project enabled us to understand all the design patterns thoroughly. Various techniques like use case analysis, diagram are helpful in prototyping software design.

## BIBLIOGRAPHY

[1] www.c\_lang.net & en.wikipedia.org

[2] "C: Basic syntax". The C Group. Retrieved 2008-02-22.

[3]"Your first C-enabled system" The Core Group. Retrieved 2008-02-25

[4] Let US C (by Yashvant Kanatekar)

[5] ANSI C (by Kernighan and Ritchie)

[6] Mastering C (by K R Venugopal)

[7] Datastructures (by Tenenbaum)

[8] Data Structures (by Seymour Lipschutz)