A Project Report on

Hostel Management System

**DIPLOMA ENGINEERING**

in

**COMPUTER SCIENCE AND TECHNOLOGY**

of

West Bengal State Council of Technical Education



Under The Supervision of

Lecturer Nilanjana Bose

Submitted By

|  |  |  |  |
| --- | --- | --- | --- |
| Pratik Patra  REG No. D151651363 | Sanjeev Sharma  REG No. D151651367 | | Subhasis Ghosh  REG No. D151651369 |
| Kajal Kumar Halder  REG No. D151651361 | | Jeet Dhara  REG No. D151651341 | |

DEPT. OF COMPUTER SCIENCE & TECHNOLOGY

SWAMI VIVEKANANDA INSTITUTE OF SCIENCE & TECHNOLOGY

SESSION: 2015-2018

CERTIFICATE



SWAMI VIVEKANANDA INSTITUTE

OF

SCIENCE & TECHNOLOGY

This is to certify that the project work, entitled “Hostel Management System” submitted by Pratik Patra, Sanjeev Sharma, Subhasis Ghosh, Kajal Kumar Halder, Jeet Dhara, students of 6th Semester, 3rd Year of C.S.T. has been prepared according to the regulation of Diploma in Computer Science and Technology of West Bengal State Council Technical Education. This report has been submitted in partial fulfillment of requirements for awarding Diploma in Computer Science & Technology of West Bengal State Council of Technical Education, West Bengal.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Surya Sekhar Chowdhury Nilanjana Bose

(Signature of HOD) (Signature of Supervisor)

Dept. of Computer Science & Tech Dept. of Computer Science & Tech

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(Signature of External Examiner)

ACKNOWLEDGEMENT

Here we gladly present this project report on “HOSTEL MANAGEMENT SYSTEM” as part of the 6th semester Diploma in Computer Science and Engineering. The project work in this report is an outcome of continual work and intellectual support from various sources. It is almost impossible to express adequately the debts owed to many persons who have been instrumental in imparting this work a successful status. It is however a matter of great pleasure to express my gratitude and appreciation to all those people who had helped in completion of this project.

We would like to take the opportunity to thank our Head of Department (C.S.T.) Mr. Surya Shekar Chowdhury for giving us an opportunity to work on this project, which not only has increased our awareness but also has taught the importance of teamwork.

We express our sincere thanks to our project mentor Ms. Nilanjana Bose for her invaluable guidance and frequent suggestions during the course of the project. Her suggestions helped us to maintain a good quality of work.

Our special cordial thanks to Computer Science Department, Swami Vivekananda Institute of Science & Technology for their earnest efforts and guidance throughout the project work. Last but not the least, we would like to thank friends for the support and encouragement they have given us during the course of our work.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Pratik Patra Sanjeev Sharma Subhasis Ghosh

REG No. D151651363 REG No. D151651367 REG No. D151651369

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Kajal Kumar Halder Jeet Dhara

REG No. D151651361 REG No. D151651341

|  |  |
| --- | --- |
| Table of contents | No. |
| Abstract | 5 |
| Chapter 1, Introduction | 6 |
| 1.1 Project Overview | 7 |
| 1.2 Objectives of our project | 7 |
| 1.3 Project Category | 7 |
| 1.4 Tools and platforms to be used | 7 |
| 1.5 Platform requirement | 7 |
| 1.6 Project organization (the team) | 7 |
| Chapter 2, Problem analysis, feasibility study | 8 |
| 2.1 Existing system | 9 |
| 2.2 Proposed system | 9 |
| 2.3 Feasibility study | 9 |
| 2.3.1 Technical feasibility | 10 |
| 2.3.2 Economical feasibility | 10 |
| 2.3.3 Operational feasibility | 10 |
| 2.3.4 Time Feasibility | 10 |
| 2.4 Project management approach | 11 |
| Chapter 3, Software Requirement Specification (SRS) | 12 |
| 3.1 Introduction | 13 |
| 3.2 User requirements definition | 13 |
| 3.3 System requirement specification | 13 |
| Chapter 4, System design and detailed design | 15 |
| 4.1 Database design | 16 |
| 4.2 System design | 18 |
| Chapter 5, How to use this software | 31 |
| Chapter 6, Testing, coding, logics | 38 |
| 6.1 Introduction of Testing | 39 |
| 6.1.1 White box testing | 39 |
| 6.1.2 Black box testing | 39 |
| 6.1.3 Unit testing | 39 |
| 6.1.4 System testing | 39 |
| 6.1.5 Performance testing | 40 |
| 7. Coding | 40 |
| Chapter 7, Bibliography | 58 |

ABSTRACT

The basic idea behind developing this Hostel management System Project is to make easy accommodation and easy calculation process for the Hostel Management. Through this Hostel management System Project they can easily allocate the student to their appropriate hostel and rooms. As various other services are provided by the mess department and take additional charges. So candidates who will take this service will only have to pay that exact amount. Our software will be compatible to the existing system with the system which is more users friendly and more GUI oriented. We can improve the efficiency of the system, thus overcome the drawbacks of the existing hostel management system. Less human error, Strength and strain of manual labor can be reduced, High security, Data redundancy can be avoided to some extent, Data consistency, Easy to handle, Easy data updating, Easy record keeping, Backup data can be easily generated.

Chapter – 1, Introduction

In this chapter we are going to discuss about project overview, objectives, project category, tools and platforms to be used and platform requirements.

* 1. **Project Overview:**

The online hostel management system is desktop based software to provide college students accommodation to the university hostel more efficiently. This project keeps details of the students. It is headed by Warden. He will be the administrator. For accommodate a large number of students into hostel. This project is intended to minimize human works and make hostel allocation is an easier job for hostel authorities.

* 1. **Objectives of our project:**

The objective is to make an automated system to carry out the various operation of a Hostel. The system will provide the ease of use to the staff of the hostel by performing all the work on a computer system rather than following a paper pen approach. This approach helps improving the reliability of the data maintained and provides a fast and efficient interface for the users of the software.

* 1. **Project Category:**

Desktop Application.

* 1. **Tools and platforms to be used:**

Tools: Visual Studio 2010.

Platform: Windows.

* 1. **Platform requirement:**

Windows 7 and up.

1GB of RAM.

50MB of hard disc space.

Monitor: resolution at least 1024x768.

* 1. **Project organization (the team):**

|  |  |  |
| --- | --- | --- |
| **No.** | **Job title** | **Description** |
| 1. | Project leader | To manage all the processes of the project, analyze the requirements of project. |
| 2. | SW designers | To design the models that helps the programmer in implementation phase. |
| 3. | Programmer | To implement the models. |
| 4. | Tester | To test the project at various phase. |
| 5. | Writers (Documentation) | To document the project from start to finish. |

Chapter – 2

In this chapter we are going to discuss about problem analysis, feasibility studies and which software engineering paradigm was applied in our project

**2.1 Existing system:**

The old system of keeping records in hostel was paper file system. On which records are kept in a paper file by hand. There were many problems such as security, slowness, calculation errors, storage place, backup of records, redundancy etc.

Disadvantage of existing system:

* More human power
* More strength and strain of manual labor needed
* Repetition of same procedure.
* Low security.
* Data redundancy.
* Difficulty to handle.
* Difficulty to update data.
* Record keeping is difficult.

**2.2 Proposed system:**

The proposed system is having many advantages over the existing system. It requires less overhead and very efficient. The proposed system deals with the mess calculation and allotment process efficiently.

Advantage of our proposed system:

* Less human error.
* Manual labor can be reduced.
* High security.
* Data redundancy can be avoided to some extent.
* Data consistency.
* Easy to handle.
* Easy data updating.
* Easy record keeping.

**2.3 Feasibility study:**

A feasibility study is a type of analysis used in measuring the ability and likelihood to successfully complete a project including all relevant factors. It must account for factors that affect it such as technological, economical, perational and scheduling factors. Project analyst uses feasibility studies to determine potential positive and negative outcomes of a project before investing a considerable amount of time and money into it.

**2.3.1 Technical feasibility:**

We are using .NET as front end technology, C# as scripting language and Microsoft Access as backend technology, application can be implemented using Visual Studio. Required resources such as icons are available on the internet, and we have already worked on Visual Basic in the last semester thus this project Hostel Management System is technically feasible, also Windows is a widely used operating system so there’s scope to extend our project further in future.

**2.3.2 Economical feasibility:**

Economic feasibility analysis includes a broad range of tests that include long term cooperative income strategies, cost of resources needed for development, cost benefit analysis. In existing system they had to maintain many registers/books is a costly affair. This can be reduced by keeping data in the digital format that is reliable and cheaper. Since the development cost for system satisfies the organization therefore the software is economically feasible.

**2.3.3 Operational feasibility:**

This project has been developed in such a way that it becomes very easy even for a person with little computer knowledge to operate it. This software is very user friendly and does not require any technical person to operate .Thus the project is even operationally feasible.

**2.3.4 Time Feasibility:**

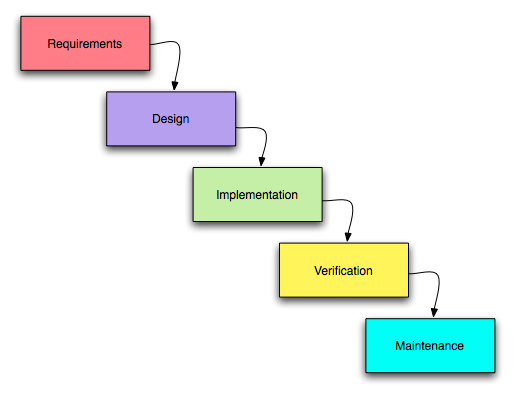
**2.4 Project management approach:**

**Software Process Model:** To solve an actual problems in an industry, software developer or a team of developers must integrate with a development strategy that include the process, methods and tools layer and generic phases. This strategy is often referred to a process model or a software developing paradigm.

Our project follows the “Waterfall” model.

The steps of waterfall model are:

* Requirement Definition
* System and Software Design
* Implementation
* Integration and System Testing
* Operation and Maintenance



Chapter – 3

In this chapter we are going to discuss about Software Requirement Specification (SRS), cost estimation and project planning.

**3.1 Introduction:**

The purpose of this SRS document is to write the functional and non-functional user or system requirements that represent the characteristics of Hostel Managements System.

The scope and limitations of this system is:

* Hostel management system is designed for hostel (like schools and universities)
* Admin checks the bed availability of a certain hostel and if available then he first registers the student on software’s database and allots room to student under student’s choice of an AC or NON-AC room.
* This project can only store records and unable to print it, we plan to add printing facilities in future.

**3.2 User requirements definition:**

* Less human error.
* Manual labor can be reduced.
* High security.
* Data redundancy can be avoided to some extent.
* Data consistency.
* Easy to handle.
* Easy data updating.
* Easy record keeping.

**3.3 System requirement specification:**

This section gives a functional requirement that is applicable to the HMS.

These are sub modules in this phase:

* Administrator module (main page)
* User Registration module
* Hostel module
* Room module
* Students Registration module
* Students Module
* Services module
* Fee Payment module
* Inventory module
* Records modules

The functionality of each module is as follows:

* **Administrator module (Main page)**

Administrator or hostel authority can:

1. Register hostel details.
2. Update room availability of hostels.
3. Register student details.
4. Add services and their charges availed by students.
5. Control the status of fee payment.
6. See due dates.
7. Add products to inventory with their respective charges.
8. See various records.

* **User Registration Module**

1. To register users who can access the software.

* **Hostel Module**

Here hostel authority can:

1. Add his different hostels.

* **Room Module**

Here hostel authority can:

1. Add room details that are available under a certain hostel.

* **Students Module**

Here hostel authority can:

1. Add/update/edit student details.

* **Student’s Registration Charges Module**

1. Here hostel authority can add registration fees such as caution money details.

* **Services Module**

Here hostel authority can:

1. Upload details of services that have been availed by the students.

* **Fee Payments Module**

Here hostel authority can:

1. See the total due of a student and make a payment.

* **Due Dates Module**

Here hostel authority can:

1. See the fees due by the students on the current date.

* **Inventory Module**

Here hostel authority can:

1. Upload details of purchases that has been purchased by the hostel for it’s own services.

Chapter – 4

In this chapter we are going to discuss about System design and detailed design.

**4.1 Database design:**

|  |  |
| --- | --- |
| DueDate |  |
|  | ID |
|  | HostelerID |
|  | DueDate |
| FeePayment |  |
|  | FeePaymentID |
|  | HostelerID |
|  | ServiceCharge |
|  | FeeMonth |
|  | PaymentDate |
|  | TotalPaid |
|  | Fine |
|  | DuePayment |
| Hostel |  |
|  | HostelName |
|  | Hostel\_Address |
|  | Hostel\_Phone |
|  | ManagedBy |
|  | Hostel\_ContactNo |
| Hostelers |  |
|  | HostelerID |
|  | HostelerName |
|  | DOB |
|  | Gender |
|  | RoomNo |
|  | HostelName |
|  | DateOfJoining |
|  | Purpose |
|  | FatherName |
|  | MobNo1 |
|  | Phone1 |
|  | MotherName |
|  | MobNo2 |
|  | City |
|  | Address |
|  | Email |
|  | ContactNo |
|  | InsOfcDetails |
|  | Phone2 |
|  | Agreement |
|  | GurdianName |
|  | GurdianAddress |
|  | MobNo3 |
|  | Phone3 |
|  | Photo |
|  | DocsPic |
|  | Completion |
| PurchasedInventory |  |
|  | ID |
|  | ProductName |
|  | Category |
|  | Transaction |
|  | PartyName |
|  | PurchaseDate |
|  | Quantity |
|  | Unit |
|  | Price |
|  | TotalPrice |
| RegCharges |  |
|  | ReceiptNumber |
|  | CautionMoney |
|  | RentalCharges |
|  | PaymentDate |
| Registration |  |
|  | Username |
|  | Password |
|  | Name |
|  | ContactNo |
| Room |  |
|  | HostelName |
|  | RoomNo |
|  | RoomType |
|  | NoOfBeds |
|  | BedsAvailable |
| Services |  |
|  | ID |
|  | HostelerID |
|  | ServiceName |
|  | ServiceCharges |
| Users |  |
|  | Username |
|  | Password |

**4.2 System design:**

OR

Password

Username

Admin

DFD LEVEL 0

Login:

Access denied

Main menu

Access granted

Input information

Admin

Username & password

Error messege

User Registration:

Input information

Saves user details

User details

Admin

Hostel:

Saves hostel details.

Hostel details

Admin

Input information

Room:

Room

details

Admin

Saves room details.

Input information

Students:

Bed not available, error

Bed not available

Saves student details, agreement months, allots room

Bed available

Input information

Student details

Admin

Registration:

Saves student registration details

Import Student details from records

Admin

Services:

Admin

Import Student details from records

Saves student’s service charges

Fee Payment:

Import Student details, service charge from records

Admin

Already paid

Saves payment details, dues if any.

Not paid

Inventory:

Input information

Saves Inventory details.

Inventory details

Admin

ER Diagram:

Hostel

Manage

Users

Has

Login

Room

DueDate

Student

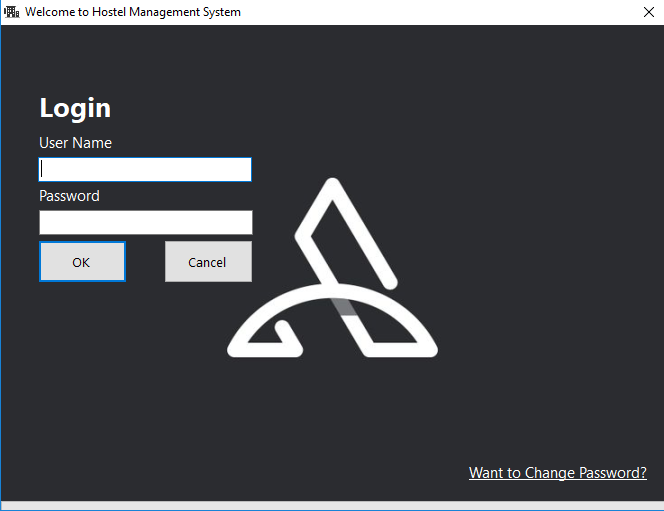
Has

Services

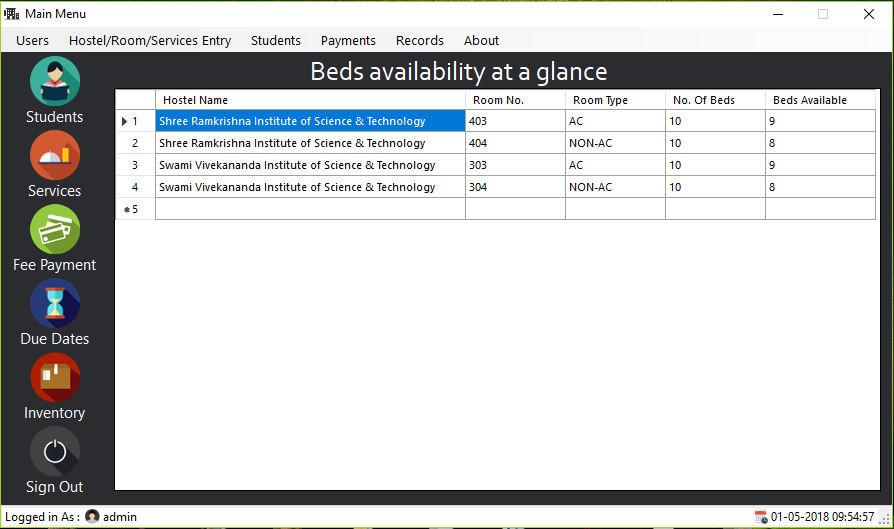
Inventory

RegCharges

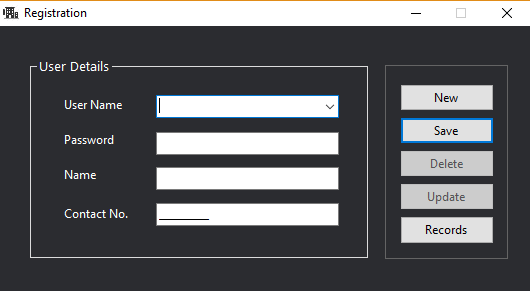
Login Page:



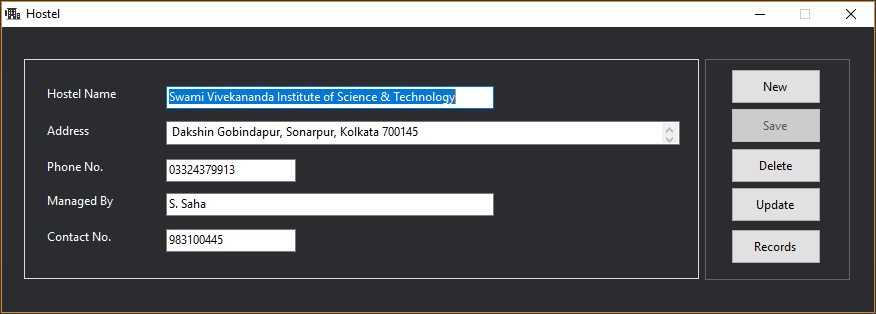
Main page:



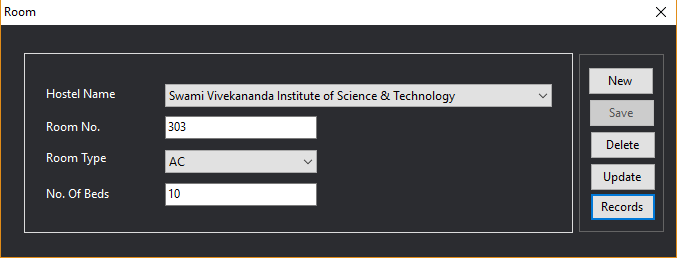
User Registration:



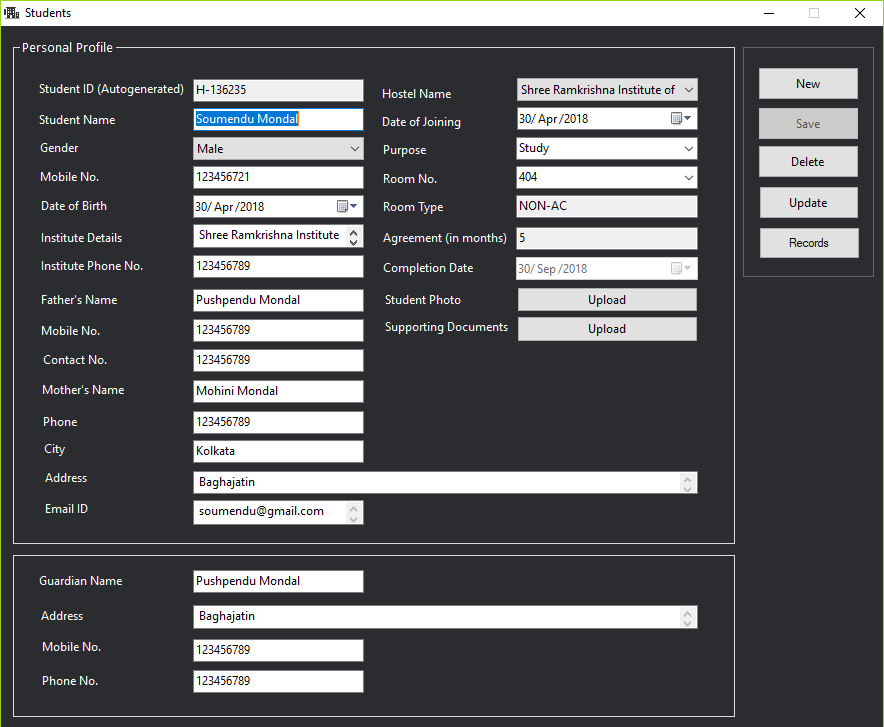
Hostel Registration:



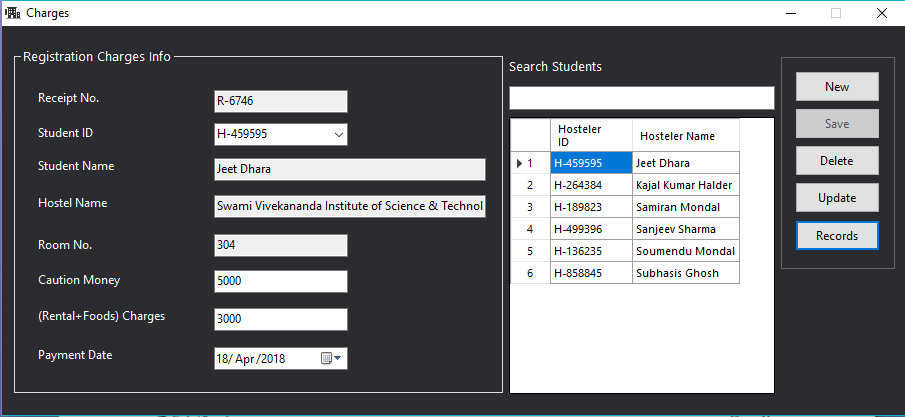
Room Registration:



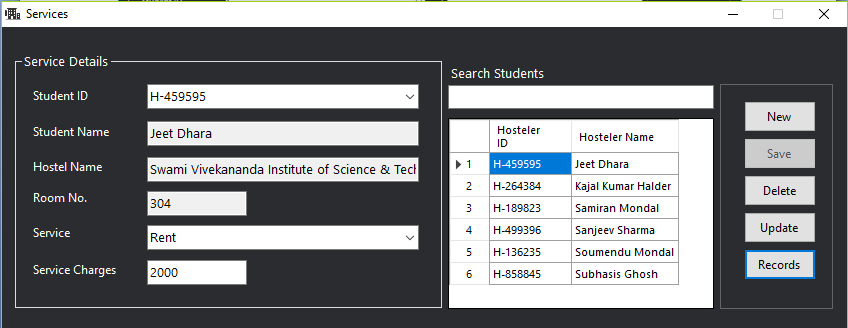
Student’s module:



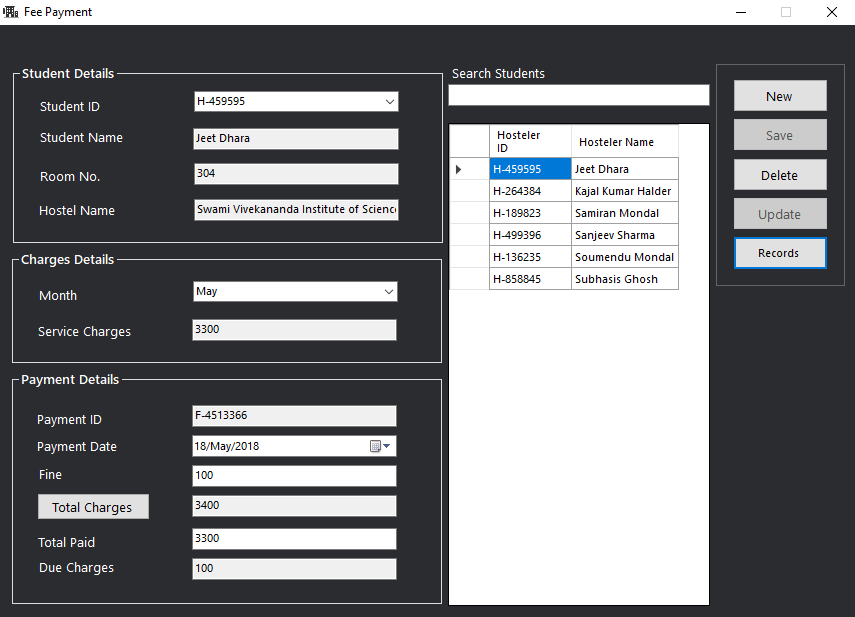
Student’s registration charges page:



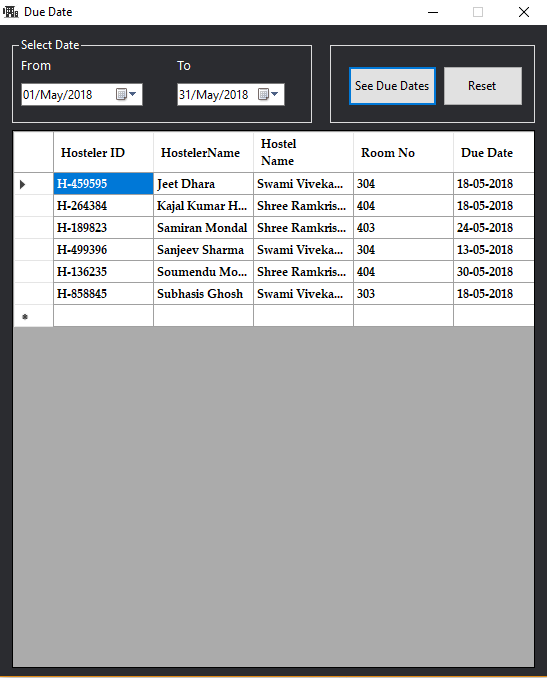
Services module:



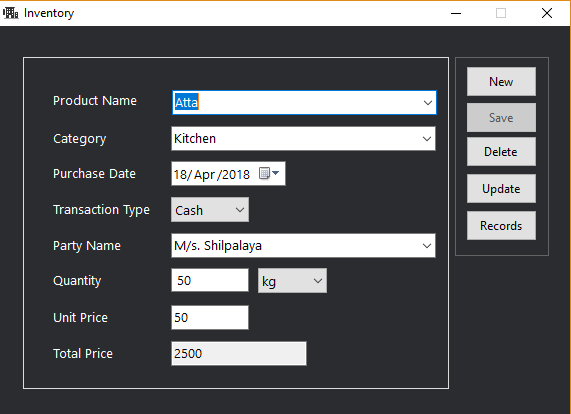
Fee Payment module:



Due Dates module:

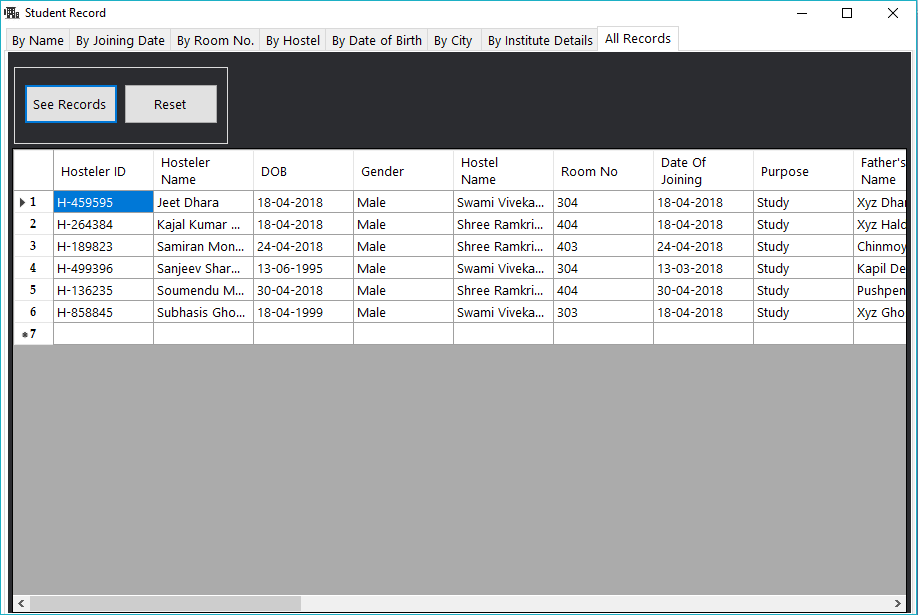


Inventory module:

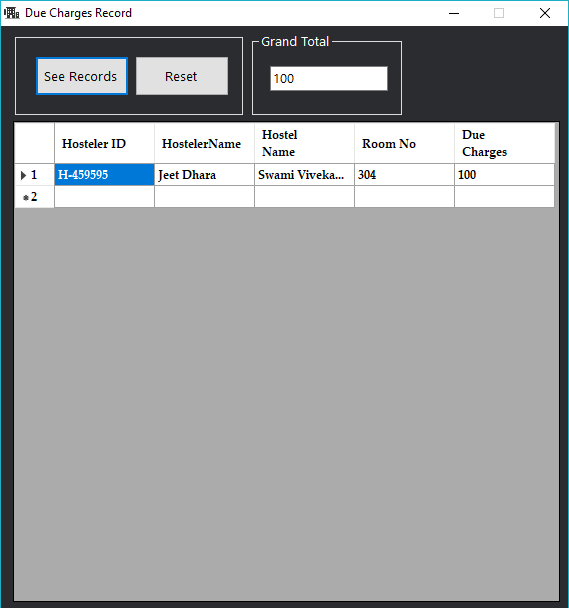


**Records modules**

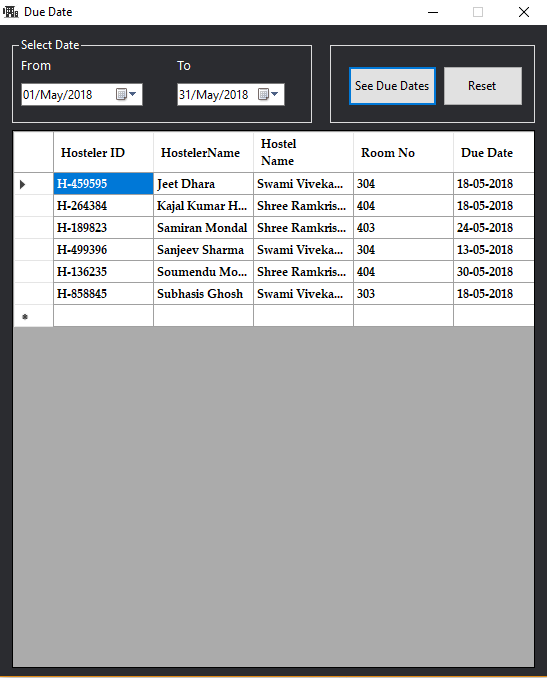
Student’s records:



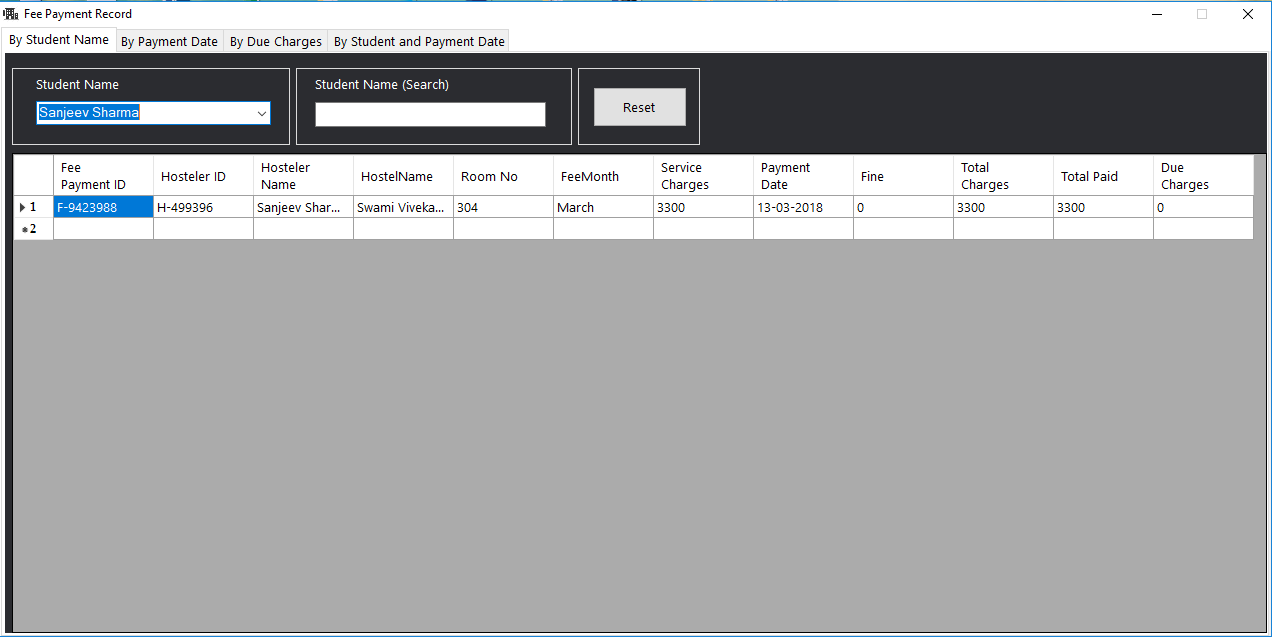
Due Charges record:



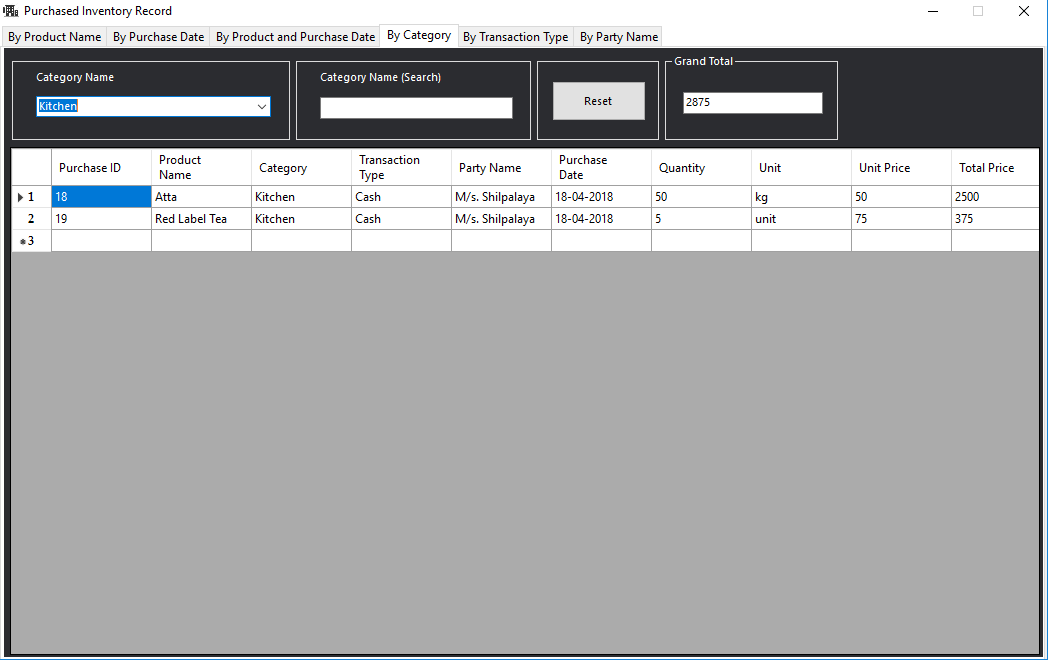
Due date:



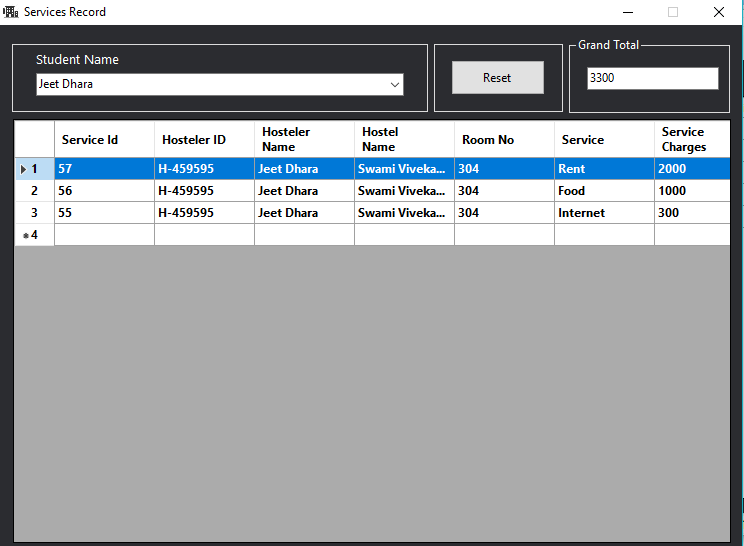
Fee payment record:



Purchased inventory record:



Services record:



Chapter – 5

In this chapter we are going to discuss how to use this software.

Case 1: Login

1. Open the program.
2. Enter your “User Name” and “Password”.
3. Click “OK”.
4. You’ll be logged in to Main Page where you can see current bed availability of hostel rooms.
5. If you want to change password then click “Want to Change Password?”.

Case 2: Add Hostel details.

1. On main page, from the upper menu bar select “Hostel/Room/Services Entry”.
2. Select “Hostel”.
3. A new window “Hostel will be opened”.
4. Now enter Hostel details.
5. Click “Save”
6. Hostel details will be saved.

Case 3: Update Hostel details.

1. From same page, click on “Records”.
2. A new window “Hostel record” will be opened.
3. Choose the Hostel.
4. You’ll be back to “Hostel” page, where you can see the existing details.
5. Edit the details you want and click “Update”.
6. Hostel details will be updated.

Case 4: Delete Hostel details.

1. From same page, click on “Records”.
2. A new window “Hostel record” will be opened.
3. Choose the Hostel.
4. You’ll be back to “Hostel” page, where you can see the existing details.
5. Click “Delete” from right hand side bar.
6. Hostel details will be deleted.

Case 5: Add room details of a Hostel.

1. On main page, from the upper menu bar select “Hostel/Room/Services Entry”.
2. Select “Room”.
3. A new window “Room” will be opened.
4. Select the Hostel where you want to add rooms.
5. Now enter room details (Room No, type, No. of Beds).
6. Click “Save”.
7. Room details will be saved.

Case 6: Edit room details of a Hostel.

1. From same page, click on “Records”.
2. A new window “Record” will be opened.
3. Choose the Hostel.
4. Select Room No.
5. You’ll be back to “Room” page, where you can see the existing details.
6. Edit the details you want and click “Update”.
7. Room details will be updated.

Case 7: Delete Hostel details.

1. From same page, click on “Records”.
2. A new window “Record” will be opened.
3. Select Room No.
4. You’ll be back to “Room” page, where you can see the existing details.
5. Click “Delete” from right hand side bar.
6. Room details will be deleted.

Case 8: Student details upload.

1. From upper menu tabs Click on “Students” and select “Profile entry” or from main menu left side bar select “Students”
2. Now fill up student details along with “Hostel Name”, “Date of Joining”, “Purpose”, “Room No.”, “Room Type” and “Agreement”.
3. Upload student photo and supporting documents such as proof of identity and proof of address.
4. Now click on “Save”.
5. Student’s details will be saved.

Case 9: Update Student details:

1. From same page click on “Records Students”, it’ll open the “Students Record” page.
2. From “Students Record” page click on the Student’s details you want to update.
3. Now you’ll be back to Students page with all the student’s details. Update the portion you want and select “Update”.
4. Student’s details will be updated.

Case 10: Enter the registration charges

(Caution money, Rental+Food charges as discussed with hostel authority).

1. On main page, from the upper menu bar select “Students”.
2. Select “Registration Charges”.
3. A new window called “Charges” will be opened.
4. Here select the student’s name from the right hand side table, or search.
5. Enter the details, payment date.
6. Click “Save”.
7. Details will be saved.

Case 11: Update Registration Charge’s details of an existing student.

1. On same page, click on “Existing”.
2. A new window called “Registration Charges Info” will be opened.
3. Search or select the student’s name.
4. You’ll be back to previous “Charges” page.
5. Edit the details you want.
6. Click on “Update” from right hand side bar.
7. New charge details will be updated.

Case 12: Delete Registration Charge’s details of an existing student.

1. On same page, click on “Existing”.
2. A new window called “Registration Charges Info” will be opened.
3. Search or select the student’s name.
4. You’ll be back to previous “Charges” page.
5. Click on “delete” from right hand side bar.
6. Charge details will be deleted.

Case 13: Enter Service Charge details.

1. On main page, from the upper menu bar select “Hostel/Room/Services Entry”.
2. Select “Services Entry” from dropdown menu.
3. Select the student name from right hand side list or search.
4. Enter details of service such as rent, food, internet and charges of them.
5. Click “Save” from the right hand side options.
6. Service Charge details will be saved.

Case 14: Edit Service Charge details for an existing student.

1. From same page, right hand side options choose “Existing”.
2. A new window called “Services Record” will be opened.
3. Select Student’s name and click on the service that you want to edit.
4. You’ll be back to “Services” page where you can see the details.
5. Edit the charge or service details.
6. Click “update” from right hand side options.
7. Charge details will be updated.

Case 15: Delete Service Charge details for an existing student.

1. From same page, right hand side options choose “Existing”.
2. A new window called “Services Record” will be opened.
3. Select Student’s name and click on the service that you want to delete.
4. You’ll be back to “Services” page where you can see the details.
5. Click “Delete” from right hand side options.
6. Charge details will be deleted.

Case 16: Fee payment.

1. On main page, from the upper menu bar select “Payments” and click on “Fee Payments” from the dropdown menu or select “Fee Payment” from main menu left hand side options.
2. A new window called “Services” will be opened.
3. Select student name from right hand side list or search for the name.
4. Select “Month” of the fee that will be paid.
5. Select “Payment Date”.
6. Add late fine (if any).
7. Click on “Total charges” to calculate the total charge.
8. Now enter the amount that student has paid.
9. Click on “Save” from right hand side menu options.
10. Data will be saved.

Case 17: Update a paid fee details for an existing student who already paid fee.

1. On same page click on “Records Payments”.
2. A new window called “Fee Payment Record” will pop up.
3. Select Student’s name.
4. Now you’ll be back to previous page.
5. Update/edit the details you want.
6. Click on “Update” from right hand side options.

Case 18: Delete a paid fee details for an existing student who already paid fee.

1. On same page click on “Records Payments”.
2. A new window called “Fee Payment Record” will pop up.
3. Select Student’s name.
4. Now you’ll be back to previous page.
5. Click on “Delete” from right hand side options.
6. Fee Payment record will be deleted.

Case 19: Inventory

1. On main page, from left side menus, click on “Inventory”.
2. A new window called “Inventory” will be opened.
3. Here enter the product name.
4. Create a category of the product such as Bathroom, Kitchen or Laundry or select existing categories.
5. Select purchase date.
6. Select Transaction type: Cash or Credit.
7. Enter Party name: Store name where you have bought the product.
8. Enter Quantity in gm/kg/litre/unit.
9. Enter unit price.
10. Total price will be automatically calculated,
11. Click “Save” from right hand side options.
12. Inventory details will be saved.

Case 20: Edit Inventory details from existing records.

1. On same page, right hand side menus, click on “Records”.
2. A new window called “Purchased Inventory Record” will be opened.
3. Select Product name.
4. You’ll back on previous “Inventory” page.
5. Edit the details you want.
6. Click on “Update” from right hand side menus.
7. Inventory details will be updated.

Case 21: Delete Inventory details from existing records.

1. On same page, right hand side menus, click on “Records”.
2. A new window called “Purchased Inventory Record” will be opened.
3. Select Product name.
4. You’ll back on previous “Inventory” page.
5. Click on “Delete” from right hand side menus.
6. Inventory details will be deleted.

Case 26: See Student’s records.

1. On main page. Click “Records” from top menus.
2. Select “Students”
3. Here you can check student’s records.

Case 22: See Check Out records. (When a student vacated a room)

1. On main page. Click “Records” from top menus.
2. Select “Check Out”
3. Select a date.
4. Click on “See Records”.
5. Now you can see details of students who and when has vacated a room.

Case 23: See Services record that has been availed by a student.

1. On main page. Click “Records” from top menus.
2. Select “Services”
3. Select a student.
4. Now the student’s services details will be shown.

Case 24: See Fee Payment records.

1. On main page. Click “Records” from top menus.
2. Select “Fee Payment”
3. Select a student.
4. Now the student’s fee payment will be shown.

Case 25: See Due Charges.

1. On main page. Click “Records” from top menus.
2. Select “Due Charges”.
3. Click on “See Records”.
4. Now the student with a payment due will be shown.

Case 26: See Due Dates.

1. On main page Click “Records” from top menus.
2. Select “Due Dates”
3. Select a date range.
4. Click “See Due Dates”.
5. Now Student’s due date details will be shown.

Case 27: See Inventory records.

1. On main page Click “Records” from top menus.
2. Select “Inventory”
3. Select a product name, or you can choose to see By Purchase date and other categories listed in upper menu tabs.
4. Now Student’s due date details will be shown.

Case 28: User Registration.

1. On main page Click “Users” from top menus.
2. Select “Registration”
3. Choose a username and password, enter Name and Contact No.
4. Click “Save”.
5. You can also Delete an user, Update details from same menu.

Chapter – 6

In this chapter we are going to discuss testing, coding, logics.

**6.1 Introduction of Testing**

Software testing is a process of executing a program or application with the intent of finding the software bugs. It can also be stated as the process of validating and verifying that a software program or application or product: Meets the business and technical requirements that guided its design and development

**6.1.1 White box testing:**

White Box Testing is applied in Hostel Management System’s System Design.

* It is also called as STRUCTURAL TESTING or GLASS BOX TESTING.
* Testers use the knowledge of internal logic of the system.
* Mostly verification techniques are used.
* It does not ensure that the user requirement had been met.
* Cost is very high since expert testers are required.
* Example: feasibility review, designer review.

**6.1.2 Black box testing:**

Black Box Testing applied on Hostel Management System’s Requirements.

* It is also called as FUNCTIONAL TESTING.
* These tests are conducted at interface.
* Testers do not have information about the internal functionality of the system.
* Mostly validation techniques are used.
* It stimulates the actual system usage.
* They have potential of not detecting the logical errors.
* Example: unit testing, integration testing, system testing, and acceptance testing.

**6.1.3 Unit testing:**

Testing of individual software components or modules.Typically done by the programmer and not by testers as it requires detailed knowledge of the internal program design and code. May requires developing test driver modules or test harnesses. In which they check the every unit or part of HMS.

**6.1.4 System testing:**

Entire system is tested as per the requirements. Black-box type testing that is based on overall requirements specifications, covers all combined parts of a system. In which Testing check the Whole HMS System beginning to the end of HMS SRS.

**6.1.5 Performance testing**

Performance testing evaluates the run time performance of the software especially real time software.

**7. Coding**

* **Login module**
* Logic:

1. If Username and Password field is blank while pressing “Ok” button then software will give an error message box saying please enter username or password.

If Len(Trim(UserName.Text)) = 0 Then

MessageBox.Show("Please enter username", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

UserName.Focus()

Exit Sub

End If

If Len(Trim(Password.Text)) = 0 Then

MessageBox.Show("Please enter password", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Password.Focus()

Exit Sub

1. Then it’ll establish connection with database.
2. If the data in username and password textbox matches with database then it’ll hide itself and show the Main Page.
3. If the data in username and password textbox doesn’t match with database the a message box will show up saying Login failed.

Dim myConnection As OleDbConnection

myConnection = New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=|DataDirectory|\HMS\_DB.accdb;Persist Security Info=False;")

Dim myCommand As OleDbCommand

myCommand = New OleDbCommand("SELECT Username,[password] FROM Users WHERE username = @username AND password = @UserPassword", myConnection)

Dim uName As New OleDbParameter("@username", SqlDbType.VarChar)

Dim uPassword As New OleDbParameter("@UserPassword", SqlDbType.VarChar)

uName.Value = UserName.Text

uPassword.Value = Password.Text

myCommand.Parameters.Add(uName)

myCommand.Parameters.Add(uPassword)

myCommand.Connection.Open()

Dim myReader As OleDbDataReader = myCommand.ExecuteReader(CommandBehavior.CloseConnection)

Dim Login As Object = 0

If myReader.HasRows Then

myReader.Read()

Login = myReader(Login)

End If

If Login = Nothing Then

MsgBox("Login failed, please check Username & Password", MsgBoxStyle.Critical, "Login Denied")

UserName.Clear()

Password.Clear()

UserName.Focus()

Else

ProgressBar1.Visible = True

ProgressBar1.Maximum = 5000

ProgressBar1.Minimum = 0

ProgressBar1.Value = 4

ProgressBar1.Step = 1

For i = 0 To 5000

ProgressBar1.PerformStep()

Next

Me.Hide()

MainPage.lblUser.Text = UserName.Text

MainPage.Show()

End If

myCommand.Dispose()

myConnection.Close()

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

* **User registration module**
* Logic (Save):

1. On clicking “Save button” it checks if details already exists or not.
2. If exists then it’ll show “Username already exists”

con = New OleDbConnection(cs)

con.Open()

Dim ct As String = "select username from registration where username=@find"

cmd = New OleDbCommand(ct)

cmd.Connection = con

cmd.Parameters.Add(New OleDbParameter("@find", System.Data.OleDB.OleDBType.VarChar, 30, "username"))

cmd.Parameters("@find").Value = UserName.Text

rdr = cmd.ExecuteReader()

If rdr.Read Then

MessageBox.Show("User Name Already Exists", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

UserName.Text = ""

If Not rdr Is Nothing Then

rdr.Close()

End If

1. If data doesn’t exist then it’ll save them in database and user will be greeted with “Succesfully registered” messagebox.

Else

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "insert into registration(username,[password],name,contactno) VALUES (@d1,@d2,@d3,@d4)"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.Parameters.Add(New OleDbParameter("@d1", System.Data.OleDB.OleDBType.VarChar, 30, "username"))

cmd.Parameters.Add(New OleDbParameter("@d2", System.Data.OleDb.OleDbType.VarChar, 30, "password"))

cmd.Parameters.Add(New OleDbParameter("@d3", System.Data.OleDB.OleDBType.VarChar, 30, "name"))

cmd.Parameters.Add(New OleDbParameter("@d4", System.Data.OleDB.OleDBType.VarChar, 15, "contactno"))

cmd.Parameters("@d1").Value = Trim(UserName.Text)

cmd.Parameters("@d2").Value = Trim(Password.Text)

cmd.Parameters("@d3").Value = Trim(txtName.Text)

cmd.Parameters("@d4").Value = Trim(ContactNo.Text)

cmd.ExecuteReader()

If con.State = ConnectionState.Open Then

con.Close()

End If

con = New OleDbConnection(cs)

con.Open()

Dim cz As String = "insert into users(UserName,[password]) VALUES (@INSERT1,@INSERT2)"

cmd = New OleDbCommand(cz)

cmd.Connection = con

cmd.Parameters.Add(New OleDbParameter("@INSERT1", System.Data.OleDb.OleDbType.VarChar, 30, "Username"))

cmd.Parameters.Add(New OleDbParameter("@INSERT2", System.Data.OleDb.OleDbType.VarChar, 30, "password"))

cmd.Parameters("@INSERT1").Value = UserName.Text

cmd.Parameters("@INSERT2").Value = Password.Text

cmd.ExecuteReader()

MessageBox.Show("Successfully registered", "User", MessageBoxButtons.OK, MessageBoxIcon.Information)

Save.Enabled = False

fillcombo()

con.Close()

End If

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

* Logic (update):

1. After filling up the details from record and putting them in respective text field it is updating “Registration” field in database uplon clicking “Update”.

If UserName.Text = "" Then

MessageBox.Show("Please select user name", "Entry", MessageBoxButtons.OK, MessageBoxIcon.Warning)

Exit Sub

End If

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update registration set [password]='" & Password.Text & "',name='" & txtName.Text & "',contactno='" & ContactNo.Text & "' where username='" & UserName.Text & "'"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

If con.State = ConnectionState.Open Then

con.Close()

End If

con = New OleDbConnection(cs)

con.Open()

Dim cz As String = "update users set [password]='" & Password.Text & "' where username='" & UserName.Text & "'"

cmd = New OleDbCommand(cz)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully Updated", "User details", MessageBoxButtons.OK, MessageBoxIcon.Information)

Update\_record.Enabled = False

fillcombo()

con.Close()

* Logic (Delete):

1. After filling up the details from record and putting them in respective text field it is deleting rows from “Registration” field in database uplon clicking “Delete”.

Try

Dim RowsAffected As Integer = 0

con = New OleDbConnection(cs)

con.Open()

Dim cq As String = "delete from registration where username=@DELETE1;"

cmd = New OleDbCommand(cq)

cmd.Connection = con

cmd.Parameters.Add(New OleDbParameter("@DELETE1", System.Data.OleDb.OleDbType.VarChar, 30, "username"))

cmd.Parameters("@DELETE1").Value = Trim(UserName.Text)

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* **Main Page**
* Logic:

1. Here we have designed a top menu strip, left hand side menu strip to access various modules in our project.

Private Sub LoginDetailsToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles LoginDetailsToolStripMenuItem.Click

LoginDetails.Show()

End Sub

Private Sub RegistrationToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles RegistrationToolStripMenuItem.Click

Me.Hide()

Registration.Show()

End Sub

1. We have a datagridview that’s getting data from “Room” in database and showing Room Availability where room available is greater than zero (>0).

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Try

con = New OleDbConnection(cs)

con.Open()

Dim sql As String = "SELECT \* from Room where bedsavailable > 0 order by hostelname,roomno,roomtype"

cmd = New OleDbCommand(sql, con)

rdr = cmd.ExecuteReader(CommandBehavior.CloseConnection)

DataGridView1.Rows.Clear()

While (rdr.Read() = True)

DataGridView1.Rows.Add(rdr(0), rdr(1), rdr(2), rdr(3), rdr(4))

End While

con.Close()

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

1. When we are clicking on objects in this module, the main page hides itself and calls the intended module.

Private Sub FeePaymentToolStripMenuItem2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles FeePaymentToolStripMenuItem2.Click

Me.Hide()

FeePaymentRecord.Show()

End Sub

Private Sub ServicesToolStripMenuItem\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ServicesToolStripMenuItem.Click

Me.Hide()

Services.Show()

End Sub

* **Student’s Module**
* Logic (Save):

1. User has to fill up the student’s details.
2. While saving details if a textbox is blank a messgebox will appear respective to the textbox saying “Please Enter Details”.

If Len(Trim(txtHostelerName.Text)) = 0 Then

MessageBox.Show("Please enter hosteler name", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

txtHostelerName.Focus()

Exit Sub

End If

If Len(Trim(cmbHostelName.Text)) = 0 Then

MessageBox.Show("Please select hostel name", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

cmbHostelName.Focus()

Exit Sub

End If

1. It checks “Room” from database if bed is available or not.
2. If bed not available then a messgebox will appear saying bed not available.

con = New OleDbConnection(cs)

con.Open()

cmd = con.CreateCommand()

cmd.CommandText = "SELECT BedsAvailable FROM Room WHERE RoomNo= '" & cmbRoomNo.Text & "' and HostelName='" & cmbHostelName.Text & "' and BedsAvailable <= 0"

rdr = cmd.ExecuteReader()

If rdr.Read() Then

MessageBox.Show("Bed not Available in selected room no.", "", MessageBoxButtons.OK, MessageBoxIcon.Information)

cmbRoomNo.Focus()

1. If all fields are entered and bed is available then an Unique ID is generated and all fields are inserted in “Hostelers” database.

txtHostelerID.Text = "H-" & GetUniqueKey(6)

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "insert into Hostelers(HostelerID,HostelerName,DOB,RoomNo,DateOfJoining,Purpose,HostelName,FatherName,MobNo1,Phone1,MotherName,MobNo2,Address,Email,ContactNo,InstOfcDetails,Phone2,Agreement,GuardianName,GuardianAddress,MobNo3,Phone3,City,Photo,DocsPic,CompletionDate,Gender) VALUES('" & txtHostelerID.Text & "','" & txtHostelerName.Text & "',#" & dtpDOB.Text & "#,'" & cmbRoomNo.Text & "',#" & dtpDateOfJoining.Text & "#,'" & cmbPurpose.Text & "','" & cmbHostelName.Text & "','" & txtFatherName.Text & "','" & txtMobNo1.Text & "','" & txtPhone.Text & "','" & txtMotherName.Text & "','" & txtMobNo2.Text & "','" & txtAddress.Text & "','" & txtEmail.Text & "','" & txtContactNo.Text & "','" & txtInOfcDeatils.Text & "','" & txtInstPhoneNo.Text & "','" & txtAgreement.Text & "','" & txtGuardianName.Text & "','" & txtGuardianAddress.Text & "','" & txtGuardianContactNo.Text & "','" & txtGuardianPhoneNo.Text & "','" & txtCity.Text & "',@image,@docspic,#" & dtpCompletionDate.Text & "#,'" & cmbGender.Text & "')"

cmd = New OleDbCommand(cb)

cmd.Connection = con

Dim ms, ms1 As New MemoryStream()

Dim bmpImage As New Bitmap(Picture.Image)

Dim bmpImage1 As New Bitmap(PictureBox2.Image)

bmpImage.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg)

bmpImage1.Save(ms1, System.Drawing.Imaging.ImageFormat.Jpeg)

Dim data As Byte() = ms.GetBuffer()

Dim data1 As Byte() = ms1.GetBuffer()

Dim p As New OleDbParameter("@d1", OleDbType.VarBinary)

Dim p1 As New OleDbParameter("@d2", OleDbType.VarBinary)

p.Value = data

p1.Value = data1

cmd.Parameters.Add(p)

cmd.Parameters.Add(p1)

cmd.ExecuteNonQuery()

con.Close()

1. Updates beds available from “Room” in database to “BedsAvailable – 1”

con.Open()

Dim ct As String = "update room set BedsAvailable = BedsAvailable - 1 where HostelName= '" & cmbHostelName.Text & "' and RoomNo='" & cmbRoomNo.Text & "'"

cmd = New OleDbCommand(ct)

cmd.Connection = con

cmd.ExecuteNonQuery()

Autocomplete()

con.Close()

1. Creates and saves due dates to “DueDate” in database based on agreement values and date of joining.

For i = 1 To Val(txtAgreement.Text)

con = New OleDbConnection(cs)

con.Open()

Dim ct2 As String = "insert into DueDate(HostelerID,DueDate) values('" & txtHostelerID.Text & "',#" & dtpDateOfJoining.Value.Date.AddMonths(i) & "#)"

cmd = New OleDbCommand(ct2)

cmd.Connection = con

cmd.ExecuteNonQuery()

con.Close()

* Logic (Update): Only student details are updateable

1. User has to click on “Records” to open the Student’s record module where user can import student details. (Same for all modules)

Me.Reset()

StudentsRecord1.fillHostelerName()

StudentsRecord1.fillRoomNo()

StudentsRecord1.fillCity()

StudentsRecord1.fillIns()

StudentsRecord1.cmbBranch.Text = ""

StudentsRecord1.DataGridView4.DataSource = Nothing

StudentsRecord1.cmbRoomNo.Text = ""

StudentsRecord1.DataGridView3.DataSource = Nothing

StudentsRecord1.DateFrom.Text = Today

StudentsRecord1.DateTo.Text = Today

StudentsRecord1.DataGridView2.DataSource = Nothing

StudentsRecord1.cmbHostelerName.Text = ""

StudentsRecord1.txtHostelerName.Text = ""

StudentsRecord1.DataGridView1.DataSource = Nothing

StudentsRecord1.DataGridView6.DataSource = Nothing

StudentsRecord1.DateTimePicker1.Text = Today

StudentsRecord1.DataGridView7.DataSource = Nothing

StudentsRecord1.cmbCity.Text = ""

StudentsRecord1.ComboBox1.Text = ""

StudentsRecord1.DataGridView5.DataSource = Nothing

StudentsRecord1.DataGridView8.DataSource = Nothing

StudentsRecord1.DateTimePicker3.Text = Today

StudentsRecord1.Show()

End Sub

1. While saving details if a textbox is blank a messgebox will appear respective to the textbox saying “Please Enter Details”.

If Len(Trim(txtHostelerName.Text)) = 0 Then

MessageBox.Show("Please enter hosteler name", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

txtHostelerName.Focus()

Exit Sub

End If

If Len(Trim(cmbHostelName.Text)) = 0 Then

MessageBox.Show("Please select hostel name", "Input Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

cmbHostelName.Focus()

Exit Sub

End If

1. If user has insterted all details then all data will be pushed into database and will show a messagebox “Successfully updated”

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update Hostelers set HostelerName='" & txtHostelerName.Text & "',DOB=#" & dtpDOB.Text & "#,FatherName='" & txtFatherName.Text & "',MobNo1='" & txtMobNo1.Text & "',Phone1='" & txtPhone.Text & "',MotherName='" & txtMotherName.Text & "',MobNo2='" & txtMobNo2.Text & "',Address='" & txtAddress.Text & "',Email='" & txtEmail.Text & "',ContactNo='" & txtContactNo.Text & "',InstOfcDetails='" & txtInOfcDeatils.Text & "',Phone2='" & txtInstPhoneNo.Text & "',GuardianName='" & txtGuardianName.Text & "',GuardianAddress='" & txtGuardianAddress.Text & "',MobNo3='" & txtGuardianContactNo.Text & "',Phone3='" & txtGuardianPhoneNo.Text & "',Photo=@d1,DocsPic=@d2,City='" & txtCity.Text & "',CompletionDate=#" & dtpCompletionDate.Text & "#,Gender='" & cmbGender.Text & "' where HostelerID='" & txtHostelerID.Text & "'"

cmd = New OleDbCommand(cb)

cmd.Connection = con

Dim ms, ms1 As New MemoryStream()

Dim bmpImage As New Bitmap(Picture.Image)

Dim bmpImage1 As New Bitmap(PictureBox2.Image)

bmpImage.Save(ms, System.Drawing.Imaging.ImageFormat.Jpeg)

bmpImage1.Save(ms1, System.Drawing.Imaging.ImageFormat.Jpeg)

Dim data As Byte() = ms.GetBuffer()

Dim data1 As Byte() = ms1.GetBuffer()

Dim p As New OleDbParameter("@d1", OleDbType.VarBinary)

Dim p1 As New OleDbParameter("@d2", OleDbType.VarBinary)

p.Value = data

p1.Value = data1

cmd.Parameters.Add(p)

cmd.Parameters.Add(p1)

cmd.ExecuteNonQuery()

Autocomplete()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

btnUpdate\_record.Enabled = False

con.Close()

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

* Logic (delete):

1. A message box will appear saying "Unable to delete..Already in use" if a student’s data exists in “FeePayment”, “Services”, “Due Date”, “RegCharges” in the database.

Try

Dim RowsAffected As Integer = 0

con = New OleDbConnection(cs)

con.Open()

Dim ct As String = "select HostelerID from FeePayment where HostelerID = '" & txtHostelerID.Text & "'"

cmd = New OleDbCommand(ct)

cmd.Connection = con

rdr = cmd.ExecuteReader()

If rdr.Read Then

MessageBox.Show("Unable to delete..Already in use", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Reset()

If Not rdr Is Nothing Then

rdr.Close()

1. If no data exists then delete the Student from “Hostelers” in the database, and users will be greeted with “Successfully deleted” messagebox.

con = New OleDbConnection(cs)

con.Open()

Dim cq As String = "delete from Hostelers where HostelerID = '" & txtHostelerID.Text & "'"

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

Autocomplete()

Reset()

* **Student’s registration module**
* Logic (Save):

1. A datagrid view that’s connected with “Hostelers” from database shows the list of Students by default.

con = New OleDbConnection(cs)

con.Open()

cmd = New OleDbCommand("SELECT HostelerID as [Hosteler ID], HostelerName as [Hosteler Name] from Hostelers where HostelerName like '" & txtHosteler.Text & "%' order by HostelerName", con)

Dim myDA As OleDbDataAdapter = New OleDbDataAdapter(cmd)

Dim myDataSet As DataSet = New DataSet()

myDA.Fill(myDataSet, "Hostelers")

DataGridView2.DataSource = myDataSet.Tables("Hostelers").DefaultView

con.Close()

1. After selecting a Student’s name it fills up the data of the student from database in appropriate text boxes.

Dim dr As DataGridViewRow = DataGridView2.SelectedRows(0)

cmbHostelerID.Text = dr.Cells(0).Value.ToString

con = New OleDbConnection(cs)

con.Open()

cmd = con.CreateCommand()

cmd.CommandText = "SELECT HostelerName,RoomNo,hostelname FROM Hostelers WHERE HostelerID= '" & cmbHostelerID.Text & "'"

rdr = cmd.ExecuteReader()

If rdr.Read() Then

txtHostelerName.Text = rdr.GetString(0)

txtRoomNo.Text = rdr.GetString(1)

txtHostelName.Text = rdr.GetString(2)

1. Upon clicking “Save” it generates Unique key and inserts data into database. User is greeted with “Successfully saved” messagebox.

txtReceiptNo.Text = "R-" & GetUniqueKey(4)

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "insert into RegCharges(HostelerID,CautionMoney,RentalCharges,PaymentDate,ReceiptNumber) VALUES ('" & cmbHostelerID.Text & "'," & txtCautionMoney.Text & "," & txtRentalCharges.Text & ",#" & dtpPaymentDate.Text & "#,'" & txtReceiptNo.Text & "')"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

con.Close()

btnSave.Enabled = False

* Logic (Delete)

1. After pulling data from records (same as Student module) it deletes the data in database.
2. Upon confirmation from user, cliclking on delete button "Do you really want to delete the record?" we are deleting the service record using Hosteler ID.

Private Sub btnDelete\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDelete.Click

Try

If MessageBox.Show("Do you really want to delete the record?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation) = Windows.Forms.DialogResult.Yes Then

delete\_records()

End If

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private Sub delete\_records()

Try

Dim RowsAffected As Integer = 0

con = New OleDbConnection(cs)

con.Open()

Dim cq As String = "delete from RegCharges where HostelerID= '" & cmbHostelerID.Text & "'"

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

Reset()

* **Services Module:**
* Logic (Save):

1. A datagrid view that’s connected with “Hostelers” from database shows the list of Students by default.

con = New OleDbConnection(cs)

con.Open()

cmd = New OleDbCommand("SELECT HostelerID as [Hosteler ID], HostelerName as [Hosteler Name] from Hostelers order by HostelerName", con)

Dim myDA As OleDbDataAdapter = New OleDbDataAdapter(cmd)

Dim myDataSet As DataSet = New DataSet()

myDA.Fill(myDataSet, "Hostelers")

DataGridView2.DataSource = myDataSet.Tables("Hostelers").DefaultView

con.Close()

1. After selecting a Student’s name it fills up the data of the student from database in appropriate text boxes.

Dim dr As DataGridViewRow = DataGridView2.SelectedRows(0)

cmbHostelerID.Text = dr.Cells(0).Value.ToString

con = New OleDbConnection(cs)

con.Open()

cmd = con.CreateCommand()

cmd.CommandText = "SELECT HostelerName,RoomNo,hostelname FROM Hostelers WHERE HostelerID= '" & cmbHostelerID.Text & "'"

rdr = cmd.ExecuteReader()

If rdr.Read() Then

txtHostelerName.Text = rdr.GetString(0)

txtRoomNo.Text = rdr.GetString(1)

txtHostelName.Text = rdr.GetString(2)

1. After user enters details in Service and Service Charges textbox and clicks save it saves them “Services” in database.

Dim cb As String = "insert into Services(HostelerID,ServiceName,ServiceCharges) VALUES ('" & cmbHostelerID.Text & "','" & cmbService.Text & "'," & txtServiceCharges.Text & ")"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Update):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database.

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update Services set HostelerID='" & cmbHostelerID.Text & "', ServiceName='" & cmbService.Text & "',ServiceCharges= " & txtServiceCharges.Text & " where ID= " & txtServiceID.Text & ""

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Delete):

1. After pulling data from records (same as Student module) it deletes the data in database.
2. Upon confirmation from user, cliclking on delete button "Do you really want to delete the record?" we are deleting the service record using service ID.

Private Sub btnDelete\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles btnDelete.Click

Try

If MessageBox.Show("Do you really want to delete the record?", "Confirmation", MessageBoxButtons.YesNo, MessageBoxIcon.Exclamation) = Windows.Forms.DialogResult.Yes Then

delete\_records()

End If

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

Private Sub delete\_records()

Try

Dim RowsAffected As Integer = 0

con = New OleDbConnection(cs)

con.Open()

Dim cq As String = "delete from Services where ID= " & txtServiceID.Text & ""

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

Autocomplete()

Reset()

* **Fee Payment module:**
* Logic (Save):

1. A datagrid view that’s connected with “Hostelers” and “services” from database shows the list of Students by default.

cmd = New OleDbCommand("SELECT distinct Hostelers.HostelerID as [Hosteler ID], HostelerName as [Hosteler Name] from Hostelers,Services where Hostelers.HostelerID=Services.HostelerID order by HostelerName", con)

Dim myDA As OleDbDataAdapter = New OleDbDataAdapter(cmd)

Dim myDataSet As DataSet = New DataSet()

myDA.Fill(myDataSet, "Hostelers")

myDA.Fill(myDataSet, "Services")

dataGridView1.DataSource = myDataSet.Tables("Hostelers").DefaultView

dataGridView1.DataSource = myDataSet.Tables("Services").DefaultView

con.Close()

1. On clicking it is filling up the value in appropriate field.

con = New OleDbConnection(cs)

con.Open()

cmd = con.CreateCommand()

cmd.CommandText = "SELECT HostelerName,RoomNo,hostelname,sum(ServiceCharges) FROM Hostelers,Services WHERE Hostelers.HostelerID= '" & cmbHostelerID.Text & "' and Hostelers.HostelerID=Services.HostelerID group by hostelername,roomno,hostelname"

rdr = cmd.ExecuteReader()

If rdr.Read() Then

txtHostelerName.Text = rdr.GetString(0)

txtRoomNo.Text = rdr.GetString(1)

txtBranch.Text = rdr.GetString(2)

txtServiceCharges.Text = rdr.GetValue(3)

1. On clicking “Total Charges” button it is summing up the value in textfield: service charge and fine.

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

txtTotalCharges.Text = Val(txtServiceCharges.Text) + Val(txtFine.Text)

1. On entering the amount paid by the student in “Total Paid” we are automatically calculating the due charges, if there’s any.

Private Sub txtTotalPaid\_TextChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles txtTotalPaid.TextChanged

txtDueCharges.Text = CInt((Val(txtServiceCharges.Text) + Val(txtFine.Text)) - Val(txtTotalPaid.Text))

1. On clicking the save button user will be greeted by successfully saved messagebox and data will be inserted into “FeePayment” in the database.

Dim cb As String = "insert into FeePayment(FeePaymentID,HostelerID,ServiceCharges,FeeMonth,PaymentDate,TotalPaid,DuePayment,Fine) VALUES ('" & txtFeePaymentID.Text & "','" & cmbHostelerID.Text & "'," & txtServiceCharges.Text & ",'" & cmbMonth.Text & "',#" & dtpPaymentDate.Text & "#," & txtTotalPaid.Text & "," & txtDueCharges.Text & "," & txtFine.Text & ")"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

1. If payment is already made for a month then it’ll show “Already Paid”

Dim ct As String = "select FeeMonth from FeePayment where FeeMonth = '" & cmbMonth.Text & "'"

cmd = New OleDbCommand(ct)

cmd.Connection = con

rdr = cmd.ExecuteReader()

If rdr.Read Then

MessageBox.Show("Already Paid", "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

* Logic (Update):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database.
2. Upon clicking the “Update button”

Private Sub Update\_record\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Update\_record.Click

Try

If (txtFine.Text = "") Then

txtFine.Text = 0

End If

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update FeePayment set HostelerID='" & cmbHostelerID.Text & "',ServiceCharges=" & txtServiceCharges.Text & ",FeeMonth='" & cmbMonth.Text & "',PaymentDate=#" & dtpPaymentDate.Text & "#,TotalPaid=" & txtTotalPaid.Text & ",DuePayment=" & txtDueCharges.Text & ",Fine=" & txtFine.Text & " where FeePaymentID='" & txtFeePaymentID.Text & "'"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Delete):

1. After pulling data from records (same as Student module) it deletes the row in database.
2. Upon confirmation from user, cliclking on delete button "Do you really want to delete the record?" we are deleting the service record using Feepayment ID.
3. Upon deletation user will be greeted with “Succesfully deleted” messagebox.

Try

Dim RowsAffected As Integer = 0

con = New OleDbConnection(cs)

con.Open()

Dim cq As String = "delete from FeePayment where FeePaymentID = '" & txtFeePaymentID.Text & "'"

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

Reset()

* **Inventory module**
* Logic (save):

1. On clicking save it is inserting the data in textfields into database.

Try

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "insert into PurchasedInventory(ProductName,Quantity,Unit,Price,PurchaseDate,TotalPrice,Category,TransactionType,PartyName) VALUES ('" & cmbProductName.Text & "'," & txtQuantity.Text & ",'" & cmbUnit.Text & "'," & txtUnitPrice.Text & ",#" & dtpPurchaseDate.Text & "#," & txtTotalPrice.Text & ",'" & cmbCategory.Text & "','" & cmbTransType.Text & "','" & cmbPartyName.Text & "')"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK,

* Logic (update):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database upon clicking the “Update button”

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update PurchasedInventory set ProductName='" & cmbProductName.Text & "',Quantity=" & txtQuantity.Text & ", Unit='" & cmbUnit.Text & "',Price=" & txtUnitPrice.Text & ",PurchaseDate=#" & dtpPurchaseDate.Text & "#,TotalPrice=" & txtTotalPrice.Text & ",Category='" & cmbCategory.Text & "',TransactionType='" & cmbTransType.Text & "',PartyName='" & cmbPartyName.Text & "' where ID= " & txtPurchaseID.Text & ""

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Delete):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database upon clicking the “Delete” button.

con.Open()

Dim cq As String = "delete from PurchasedInventory where ID = " & txtPurchaseID.Text & ""

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* **Room module**
* Logic (Save):

1. On clicking save it is inserting the data in textfields into database and user is greeted with “Successfully saved” messagebox.

con.Open()

Dim cb As String = "insert into PurchasedInventory(ProductName,Quantity,Unit,Price,PurchaseDate,TotalPrice,Category,TransactionType,PartyName) VALUES ('" & cmbProductName.Text & "'," & txtQuantity.Text & ",'" & cmbUnit.Text & "'," & txtUnitPrice.Text & ",#" & dtpPurchaseDate.Text & "#," & txtTotalPrice.Text & ",'" & cmbCategory.Text & "','" & cmbTransType.Text & "','" & cmbPartyName.Text & "')"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Update):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database upon clicking the “Update button” and user is greeted with “Successfully updated” messagebox.

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update PurchasedInventory set ProductName='" & cmbProductName.Text & "',Quantity=" & txtQuantity.Text & ", Unit='" & cmbUnit.Text & "',Price=" & txtUnitPrice.Text & ",PurchaseDate=#" & dtpPurchaseDate.Text & "#,TotalPrice=" & txtTotalPrice.Text & ",Category='" & cmbCategory.Text & "',TransactionType='" & cmbTransType.Text & "',PartyName='" & cmbPartyName.Text & "' where ID= " & txtPurchaseID.Text & ""

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Delete):

1. After pulling data from records (same as Student module) and putting them in respected text fields it deletes the data in database upon clicking the “Delete” button. User is greeted with “Successfully deleted” messagebox.

con.Open()

Dim cq As String = "delete from PurchasedInventory where ID = " & txtPurchaseID.Text & ""

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* **Hostel module**
* Logic (Save):

1. On clicking save it is inserting the data in textfields into database and user is greeted with “Successfully saved” messagebox.

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "insert into Hostel(Hostelname,Hostel\_address,Hostel\_Phone,ManagedBy,Hostel\_contactno) VALUES ('" & txtHostelName.Text & "','" & txtAddress.Text & "','" & txtPhoneNo.Text & "','" & txtManagedBy.Text & "','" & txtContactNo.Text & "')"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteReader()

MessageBox.Show("Successfully saved", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

btnSave.Enabled = False

con.Close()

* Logic (Update):

1. After pulling data from records (same as Student module) and putting them in respected text fields it updates the data in database upon clicking the “Update button” and user is greeted with “Successfully updated” messagebox.

con = New OleDbConnection(cs)

con.Open()

Dim cb As String = "update Hostel set HostelName='" & txtHostelName.Text & "', Hostel\_address='" & txtAddress.Text & "',Hostel\_Phone='" & txtPhoneNo.Text & "',ManagedBy='" & txtManagedBy.Text & "',Hostel\_contactno='" & txtContactNo.Text & "' where hostelname= '" & TextBox1.Text & "'"

cmd = New OleDbCommand(cb)

cmd.Connection = con

cmd.ExecuteNonQuery()

MessageBox.Show("Successfully updated", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* Logic (Delete):

1. After pulling data from records (same as Student module) and putting them in respected text fields it deletes the data in database upon clicking the “Delete” button. User is greeted with “Successfully deleted” messagebox.

con.Open()

Dim cq As String = "delete from PurchasedInventory where ID = " & txtPurchaseID.Text & ""

cmd = New OleDbCommand(cq)

cmd.Connection = con

RowsAffected = cmd.ExecuteNonQuery()

If RowsAffected > 0 Then

MessageBox.Show("Successfully deleted", "Record", MessageBoxButtons.OK, MessageBoxIcon.Information)

* **Records module (Same logic for all record modules)**

Example: Student’s records.

* Logic:

1. We have tab pages in the top with various ways to see the records. Example: By Student Name, Joining Date and others.
2. We are connecting the database.

Try

con = New OleDbConnection(cs)

con.Open()

cmd = New OleDbCommand("SELECT HostelerID as [Hosteler ID],HostelerName as [Hosteler Name],DOB,Gender,HostelName as [Hostel Name],RoomNo as [Room No],DateOfJoining as [Date Of Joining],Purpose,FatherName as [Father's Name],MobNo1 as [Mobile No],Phone1 as [Phone No],MotherName as [Mother's Name],MobNo2 as [Mobile No 2],City,Address,Email,ContactNo as [Contact No],InstOfcDetails as [Ins/Ofc Details],Phone2 as [Phone No 2],Agreement,GuardianName as [Guardian Name],GuardianAddress as [Guardian Address],MobNo3 as [Guardian Mobile No],Phone3 as [Guardian Phone No],CompletionDate as [Completion Date],Photo,DocsPic as [Docs Pic] from Hostelers where HostelerName='" & cmbHostelerName.Text & "' order by HostelerName", con)

1. We have combo box which is connected to database.
2. Combo box is filling up from database (Student name in this case).

Sub fillHostelerName()

Try

Dim CN As New OleDbConnection(cs)

CN.Open()

adp = New OleDbDataAdapter()

adp.SelectCommand = New OleDbCommand("SELECT distinct RTRIM(HostelerName) FROM Hostelers", CN)

ds = New DataSet("ds")

adp.Fill(ds)

dtable = ds.Tables(0)

cmbHostelerName.Items.Clear()

For Each drow As DataRow In dtable.Rows

cmbHostelerName.Items.Add(drow(0).ToString())

Next

Catch ex As Exception

MessageBox.Show(ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

End Try

End Sub

1. Respected tabs are connected with a datagridview.
2. When user selects a name in combobox that is filled up from database respectively such as “Student Name” it connected with “Hostelers” in database and fills up the data from there in the datagridview.

Dim myDA As OleDbDataAdapter = New OleDbDataAdapter(cmd)

Dim myDataSet As DataSet = New DataSet()

myDA.Fill(myDataSet, "Hostelers")

DataGridView1.DataSource = myDataSet.Tables("Hostelers").DefaultView

con.Close()

Chapter – 7

Bibliography

1. https://www.google.co.in/search?q=hostel+management+system

2. https://www.google.co.in/search?q=vb.+net

3. https://www.youtube.com/results?search\_query=hostel+management+system

4. https://www.youtube.com/results?search\_query=vb.net+

5. https://www.slideshare.net/

6. https://www.wikipedia.org/