

Bachelor of Computer Applications (BCA) Program

Project Report

BCA Sem IV

AY 2023-24

*Topic Title: Smart Home Management System*

*By*

|  |  |
| --- | --- |
| Seat No | Name of Student |
| 872 | PUROHIT VINAYAK RAMESHKUMAR |

Project Guide by :

Prof.Nidhi Desai



C E R T I F I C A T E

This is to certify that **Mr**./Ms. **Purohit Vinayak Rameshkumar** examination number **2022037790** has satisfactorily completed **his**/her project work entitled **Smart Home Management System** as partial fulfilment of requirements for BCA Sem IV, during the academic year 2023-24.

Date: 07/03/2024 Prof.Nidhi Desai

Place: Surat SDJ International College, Surat

*I N D E X*

|  |  |  |
| --- | --- | --- |
| **Sr No** | **Description** | **Page No.** |
| 1 | Introduction |  |
|  | 1.1 Project Summary | 4 |
|  | 1.2 Project Technical Profile | 5 |
| 2 | Scope & Planning |  |
|  | 2.1 Requirement Analysis | 7 |
|  | 2.2 Technology Details | 8 |
| 3 | Designing |  |
|  | 3.1 Data Flow Diagram | 9 |
|  | 3.2 Flow Chart | 10 |
|  | 3.3 Usercase Diagram | 11 |
|  | 3.4 Data Dictionary | 13 |
|  | 3.5 User Interface & Coding | 16 |



# Introduction

* 1. **Project Summary**

A smart home management system is a centralized platform that allows users to control and automate various devices and appliances within their home. These systems typically utilize Internet of Things (IoT) technology to connect devices such as thermostats, lights, security cameras, and kitchen appliances to a central hub or smartphone app. Users can remotely monitor and control these devices, set schedules, receive notifications, and even integrate them with other smart home devices for seamless automation. Smart home management systems offer convenience, energy efficiency, and enhanced security for homeowners.

**Voice Control Integration**: Many smart home systems integrate with popular voice assistants like Amazon Alexa, Google Assistant, or Apple Siri, allowing users to control devices using voice commands.

**Automation and Scheduling:** Smart home systems allow users to create automation rules and schedules to streamline routine tasks and improve efficiency. For example, users can set lights to turn on automatically when motion is detected or adjust the thermostat based on occupancy patterns

**Smart Devices**: These are the physical components of the smart home system that can be controlled remotely or automated. Examples include:

**Smart Thermostats**: Allow users to remotely control and schedule heating and cooling systems, optimizing energy usage and comfort.

**Smart Lighting**: Enable users to remotely control and automate lighting fixtures, adjust brightness, and set schedules to save energy.

Smart Security Cameras and Sensors: Provide remote monitoring of the home, motion detection, and alerts for unusual activity.

A smart home integrates connected devices and technology to offer convenience, energy efficiency, and enhanced security for homeowners. Key features include remote control and automation of devices such as thermostats, lights, and security cameras through a central hub or mobile app. Users can set schedules, receive notifications, and even control their home using voice commands via integration with popular voice assistants. Smart home systems prioritize security and privacy, employing encryption and authentication methods to safeguard data and prevent unauthorized access. Interoperability with different brands and devices allows for flexibility and customization, while integration with third-party services expands functionality and automation possibilities.



# Project Technical Profile

**Software Requirements:-**

|  |  |
| --- | --- |
| **Software Configuration** | |
| **Operating System** | Windows 11 |
| **Front End** | Microsoft Visual Studio 2022 |
| **Back End** | Microsoft Office Access |
| **Other Tools** | VB.Net, SQL Queries, System Speech Recognition, OpenWeatherMap API |

**Hardware Requirements:-**

|  |  |
| --- | --- |
| **Hardware Configuration** | |
| **Processor** | Intel(R) Core(TM) i7-1065G7 CPU @ 1.30GHz 1.50 GHz |
| **RAM** | 16.00 GB |
| **Hard Disk** | 512 GB |
| **SystemType** | 64-bit Operating System, x64-based processor |



# SCOPE:-

* **Device Control**: Allow users to remotely control various smart devices such as lights, Television, Wi-FI Router, cameras, and appliances via a centralized interface like a smartphone app or voice commands.
* **Energy Management**: Provide features for monitoring and optimizing energy consumption by analyzing usage patterns, recommending energy-saving practices, and integrating with smart meters or renewable energy sources.
* **Future Expansion and Scalability:** Design the smart home management system with scalability in mind to accommodate future additions of new devices, features, and integrations as technology evolves and user needs change.
* **User Interface**: Offer user-friendly interfaces across multiple devices (smartphones, tablets, computers, smart speakers) for easy setup, configuration, and monitoring of the smart home system.

# OBJECTIVE:-

* Movie Ticket Booking System is **to manage the details of Available Movies Available Seats, Date and Time**. It manages all the information about Seats, Movies, All the Screens and Total Earnings.
* The Movie Ticket Booking system **helps register complete Booked Seats information**.
* It Stores the booked seats of particular Movie at a Particular date and time and stores the information about who booked that tickets.
* The main **objective** of the **Visual Basic** Project on **Movie Booking System** is to manage the details of Booked seats and Information of User and ADMIN section where you can change the screen and clear all the booking and set a new movie amount and other details



# Requirement Analysis

# Functional Requirements:

# User Authentication :

# Implement a secure login system for users to access the smart home management system. Support authentication methods such as username/password, biometric authentication, or integration with external identity providers (e.g., Google, Facebook,Twitter).

# Device Management:

# Allow users to add, remove, and manage smart devices within their home network. Support various types of devices including lights, thermostats, cameras, door locks, sensors, and appliances.

# Remote Control and Automation:

# Enable users to remotely control devices through a user-friendly interface.

# Implement automation features to create custom routines and schedules for device control based on triggers like time, user presence, or sensor inputs.

# Energy Monitoring and Management:

# Provide tools for monitoring energy consumption of connected devices.

# Offer recommendations and optimization suggestions to improve energy efficiency.

# Notification System:

# Implement a notification system to alert users of important events such as security breaches, device malfunctions, or scheduled maintenance.

# Non-Functional Requirements:

# Performance:

# Ensure fast response times and minimal latency for device control and system interactions. Optimize resource usage to support a large number of concurrent users and connected devices.

# Usability:

# Design an intuitive and user-friendly interface accessible to users of all technical levels. Provide comprehensive documentation and help resources to assist users in setting up and using the system effectively.

# Technology Details



**Technology Used:**

1. **Development Environment:**

* **Visual Studio**: Utilize Visual Studio as the integrated development environment (IDE) for coding, debugging, and deploying the VB.NET application. .NET Framework: Develop the application using the .NET Framework, which provides a rich set of libraries and tools for building Windows-based applications.

1. **Backend Development:**

* **VB.NET:** Use VB.NET as the primary programming language for developing the backend logic of the Smart Home Management System.
* **ADO.NET**: Utilize ADO.NET for building query for accessing MSACCESS database to Insert, Update, Delete query in Database

1. **Frontend Development:**

* **Windows Forms**: Design the user interface (UI) of the Smart Home Management System using Windows Forms, which provides a rich set of controls for creating desktop applications.
* **WPF (Windows Presentation Foundation):** Alternatively, consider using WPF for building the UI if more advanced graphics and multimedia capabilities are required.

1. **Database :**

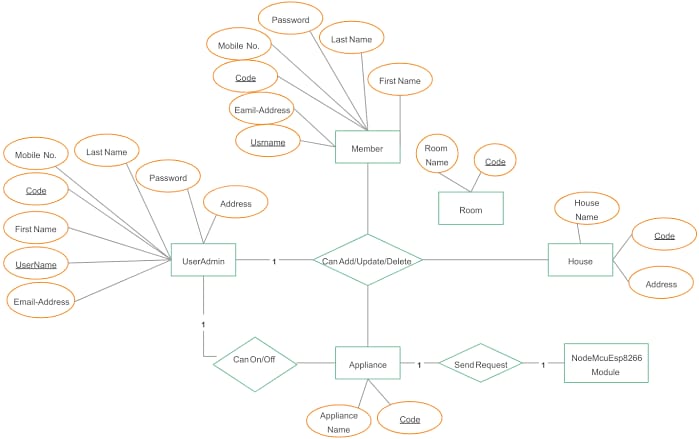
* Use Microsoft MS ACCESS as the relational database management system (RDBMS) for storing and managing data related to users, devices, configurations, and system logs.



# Designing

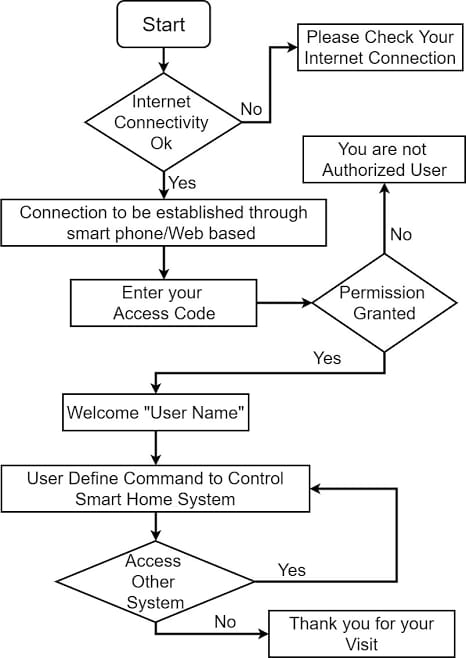
* 1. Data Flow Diagram

Context Level :





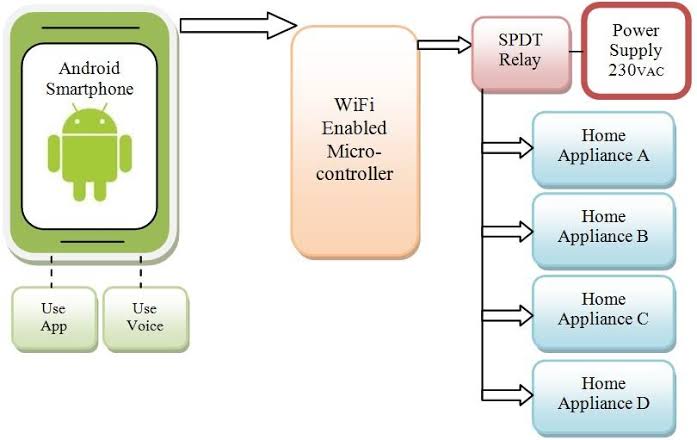
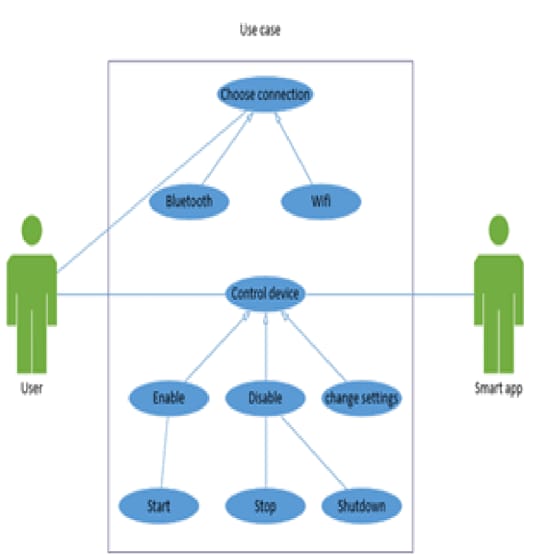
**Data Flow Diagram**







User Case Diagram





# Database Design & 3.5 Data Dictionary

**Table name** : user\_Table

**Description** : This Table gives Detail about Admin

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| ID | AutoNumber | Primary Key | Unique Identification |
| Username | Short Text | Not Null | Username |
| Password | Short Text | Not Null | Password |

**Table name** : EnergyMonitoring

**Description** : This Table gives Detail about Energy Usage of Devices in Rupees

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| EntryID | AutoNumber | Primary Key | Unique Identification |
| DeviceID | Number | Not Null | Seat Numbers |
| EnergyConsumed | Number | Not Null | EnergyConsumed in Rupees |



**Table name**: DeviceStatus

**Description**: This Table gives Detail about DeviceStatus

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| DeviceStateID | Auto Number | Primary Key | Unique Identification |
| DeviceID | Short Text | Not Null | Date of the  Show |
| State | Short Text | Not Null | State of Devices |
| Timestamp | Date/Time | Not Null | Timestamp of Device |

**Table Name** : Devices\_Table

**Description** : This Table gives Detail about All the Devices.

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| DeviceID | Auto Number | Primary Key | Unique  Identification |
| DeviceName | Short Text | Not null | Device name |
| DeviceType | Short Text | Not null | Device Type |
| Manufacturer | Short Text | Not null | Device Manufacturer |

**Table name**: DeviceAnalytics

**Description**: This Table gives Detail About Device History

|  |  |  |  |
| --- | --- | --- | --- |
| **FIELD NAME** | **FIELD TYPE** | **CONSTRAINT** | **DESCRIPTION** |
| AnalyticsID | Auto Number | Primary Key | Unique  identification |
| DeviceID | Short text | Not Null | DeviceID |
| StartTime | Date/Time | Not Null | Starting time of Devices |
| EndTime | Date/Time | Not Null | Ending time of Devices |
| RunningTime | Date/Time | Not null | Running time of Devices  EndTime – RunningTime |
| EnergyConsumed | Number | Not Null | Keep Track of EnergyConsumed in Watt |
| EnergyCost | Short text | Not null | Keep Track of EnergyCost in Rupees |



# 3.6 User Interface & Coding

Imports System.Data.OleDb

Public Class loginpage

Dim cn As New OleDbConnection

Dim cmd As OleDbCommand

Private Sub btnLogin\_Click(sender As Object, e As EventArgs) Handles btnLogin.Click

cn.ConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Study Materials\Jounral Projects\SmartHomeHub - Copy.accdb"

Dim query As String = ("select Username, Password from User\_Table")

cmd = New OleDbCommand(query, cn)

Try

cn.Open()

Dim reader As OleDbDataReader

reader = cmd.ExecuteReader()

If txtUsername.Text() = "" Or txtPassword.Text() = "" Then

MsgBox("Please Enter All Details")

End If

While reader.Read()

If txtUsername.Text() = reader("UserName") Then

If txtPassword.Text() = reader("Password") Then

MsgBox("Logined....")

Else

MsgBox("!! Incorrect Username or Password !!")

End If

End If

End While

Catch ex As Exception

MsgBox("Error :" & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub Guna2CircleButton2\_Click(sender As Object, e As EventArgs) Handles Guna2CircleButton2.Click

Me.WindowState = FormWindowState.Minimized

End Sub

Private Sub Guna2CircleButton1\_Click\_1(sender As Object, e As EventArgs) Handles Guna2CircleButton1.Click

Me.Close()

End Sub

Private Sub GunaCirclePictureBox4\_Click(sender As Object, e As EventArgs) Handles GunaCirclePictureBox4.Click

If txtPassword.PasswordChar = "\*" Then

txtPassword.PasswordChar = ""

GunaCirclePictureBox4.Image =

Else

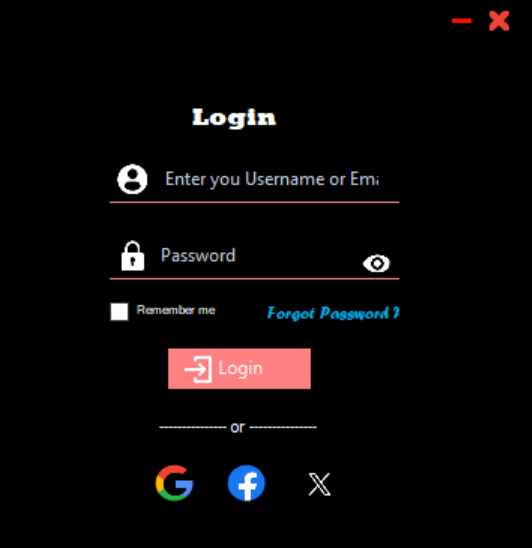
txtPassword.PasswordChar = "\*"

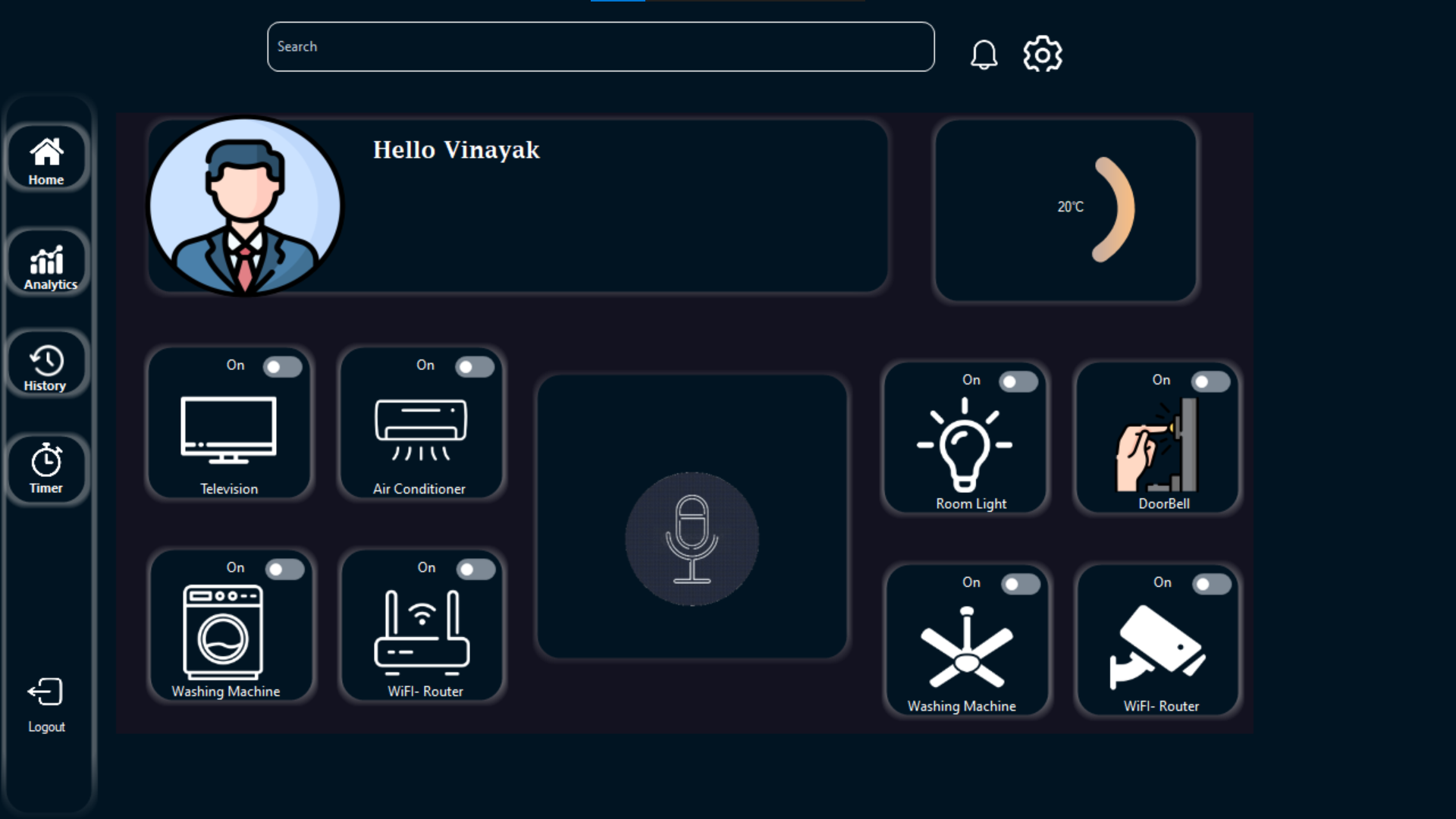
GunaCirclePictureBox4.Image = My.Resources.eye

End If

End Sub

End Class





**Main.vb**

Imports System.Net

Imports Newtonsoft.Json.Linq

Imports Guna.UI2.WinForms

Imports LiveCharts

Imports LiveCharts.Wpf

Imports System.Data.OleDb

Imports System.Timers

Imports System.Windows.Controls.Primitives

Public Class Main

Dim cn As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Study Materials\Jounral Projects\SmartHomeHub - Copy.accdb")

Dim cmd As New OleDbCommand

Dim toggleSwitches As New List(Of Guna2ToggleSwitch)()

Private Sub Main\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

AddToggleSwitchesFromContainer(Me)

SyncToggleStatus()

getusernm()

For Each toggleSwitch As Guna2ToggleSwitch In toggleSwitches

AddHandler toggleSwitch.CheckedChanged, AddressOf ToggleSwitch\_CheckedChanged

Next

End Sub

Private Sub getusernm()

Dim query As String = "SELECT FirstName FROM User\_Table"

Dim result As String = ""

' Open connection

cn.Open()

cmd = New OleDbCommand(query, cn)

Try

' Execute the query

Dim reader As OleDbDataReader = cmd.ExecuteReader()

' Loop through the results and concatenate the first names

While reader.Read()

result &= reader("FirstName").ToString() & Environment.NewLine

End While

' Close the reader

reader.Close()

Catch ex As Exception

' Handle exceptions here

Finally

' Close the connection

cn.Close()

End Try

showusernm.Text = "Hello " & result

End Sub

Private Sub Guna2PictureBox1\_Click(sender As Object, e As EventArgs) Handles Guna2PictureBox1.Click

With frmvoicecontrol

.TopLevel = False

voicepanel.Controls.Add(frmvoicecontrol)

.BringToFront()

.Show()

End With

End Sub

Private Sub SyncToggleStatus()

Try

cn.Open()

For i As Integer = 0 To toggleSwitches.Count - 1

getStatus(toggleSwitches(i), i + 1)

Next

Catch ex As Exception

MsgBox("Error syncing toggle status: " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Public Sub getStatus(ByRef toggleButton As Guna2ToggleSwitch, id As Integer)

Dim query As String = "SELECT State FROM DeviceStates WHERE DeviceID = @ID"

cmd = New OleDbCommand(query, cn)

cmd.Parameters.AddWithValue("@ID", id)

Try

Dim state As String = Convert.ToString(cmd.ExecuteScalar())

If state = "On" Then

toggleButton.Checked = True

End If

Catch ex As Exception

MsgBox("Error getting device status: " & ex.Message)

End Try

End Sub

Private Sub AddToggleSwitchesFromContainer(container As Control)

For Each control As Control In container.Controls

If TypeOf control Is Guna2ToggleSwitch Then

toggleSwitches.Add(DirectCast(control, Guna2ToggleSwitch))

End If

If control.HasChildren Then

AddToggleSwitchesFromContainer(control)

End If

Next

End Sub

Private Sub ToggleSwitch\_CheckedChanged(sender As Object, e As EventArgs)

Dim toggleSwitch As Guna2ToggleSwitch = DirectCast(sender, Guna2ToggleSwitch)

Dim id As Integer = toggleSwitches.IndexOf(toggleSwitch) + 1

Dim status As String = If(toggleSwitch.Checked, "On", "Off")

UpdateStatus(status, id)

If status = "On" Then

NewUsage(id)

Else

UpdateUsage(id)

End If

End Sub

Public Sub UpdateStatus(status As String, id As Integer)

Dim query As String = "UPDATE DeviceStates SET State = @Status WHERE DeviceID = @ID"

cmd = New OleDbCommand(query, cn)

cmd.Parameters.AddWithValue("@Status", status)

cmd.Parameters.AddWithValue("@ID", id)

Try

cn.Open()

cmd.ExecuteNonQuery()

query = "Update DeviceStates SET Timestamp = @timestamp WHERE DeviceID = @ID"

cmd = New OleDbCommand(query, cn)

cmd.Parameters.AddWithValue("@timestamp", DateTime.Now())

MsgBox("Device status updated to " & status & ".")

MsgBox("Device status updated to " & DateTime.Now().ToString() & ".")

Catch ex As Exception

MsgBox("Error updating device status: " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Public Sub NewUsage(id As Integer)

Dim query As String = "INSERT INTO DeviceAnalytics(DeviceID, StartTime) VALUES (@deviceID, '" & DateTime.Now() & "')"

Using cn As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Study Materials\Jounral Projects\SmartHomeHub - Copy.accdb"),

cmd As New OleDbCommand(query, cn)

cmd.Parameters.AddWithValue("@deviceID", id)

Try

cn.Open()

cmd.ExecuteNonQuery()

Catch ex As Exception

MsgBox("Error inserting device analytics: " & ex.Message) ' Display error message

End Try

End Using

End Sub

Public Sub UpdateUsage(id As Integer)

Dim query As String = "UPDATE DeviceAnalytics SET EndTime = '" & DateTime.Now() & "' WHERE DeviceID = @ID "

cmd = New OleDbCommand(query, cn)

Try

cmd.Parameters.AddWithValue("@ID", id)

cn.Open()

cmd.ExecuteNonQuery()

Catch ex As Exception

MsgBox("Error updating device analytics: " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub Guna2ShadowPanel7\_Paint(sender As Object, e As PaintEventArgs) Handles Guna2ShadowPanel7.Paint

'UpdateTemperature()

lblshowtemp.Text = "20" & "°C"

End Sub

Private Sub UpdateTemperature()

Dim apiKey As String = "cadae7dff2a967e3f61837b7d7d910be"

Dim city As String = "surat"

' API URL

Dim apiUrl As String = $"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={apiKey}&units=metric"

' Fetch temperature value from API

Dim temperature As Double = GetTemperatureFromAPI(apiUrl)

' Update label with temperature value

UpdateLabel(temperature)

End Sub

Private Function GetTemperatureFromAPI(ByVal apiUrl As String) As Double

Try

Using webClient As New WebClient()

Dim jsonData As String = webClient.DownloadString(apiUrl)

Dim jsonObject As JObject = JObject.Parse(jsonData)

Dim temperature As Double = jsonObject.SelectToken("main.temp").ToObject(Of Double)()

Return temperature

End Using

Catch ex As Exception

MessageBox.Show("Error fetching temperature data: " & ex.Message, "Error", MessageBoxButtons.OK, MessageBoxIcon.Error)

Return Double.NaN

End Try

End Function

Private Sub UpdateLabel(ByVal temperature As Double)

If lblshowtemp.InvokeRequired Then

lblshowtemp.Invoke(New Action(Of Double)(AddressOf UpdateLabel), temperature)

Else

lblshowtemp.Text = "Temperature: 20" & "°C"

End If

End Sub

End Class

**History.vb**

Imports System.Data.OleDb

Public Class History

Dim cn As New OleDbConnection

Dim cmd As OleDbCommand

Dim ds As New DataSet

Dim ad As OleDbDataAdapter

Dim dr As OleDbDataAdapter

Private Sub History\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

cn.ConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Study Materials\Jounral Projects\SmartHomeHub - Copy.accdb"

loadData()

ShowDevices()

End Sub

Private Sub loadData()

cmd = New OleDbCommand("select \* from DeviceAnalytics", cn)

Try

cn.Open()

ad = New OleDbDataAdapter(cmd)

ds = New DataSet

ad.Fill(ds)

DataGridView1.DataSource = ds.Tables(0)

Catch ex As Exception

MsgBox("Error : " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub ShowDevices()

cmd = New OleDbCommand("select \* from Devices\_Table", cn)

Try

cn.Open()

ad = New OleDbDataAdapter(cmd)

ds = New DataSet

ad.Fill(ds)

DataGridView2.DataSource = ds.Tables(0)

Catch ex As Exception

MsgBox("Error : " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub btnSearch\_Click(sender As Object, e As EventArgs) Handles btnSearch.Click

Try

cn.Open()

' Retrieve DeviceName from Devices\_Table

Dim deviceNameQuery As String = "SELECT DeviceName FROM Devices\_Table WHERE DeviceID = @DeviceID"

Using cmdDeviceName As New OleDbCommand(deviceNameQuery, cn)

cmdDeviceName.Parameters.AddWithValue("@DeviceID", CInt(txtdeviceID.Text))

Dim deviceNameReader As OleDbDataReader = cmdDeviceName.ExecuteReader()

If deviceNameReader.Read() Then

Label4.Text = deviceNameReader("DeviceName").ToString()

Else

Label4.Text = "Device not found"

End If

deviceNameReader.Close()

End Using

' Retrieve DeviceAnalytics data

Dim analyticsQuery As String = "SELECT \* FROM DeviceAnalytics WHERE DeviceID = @DeviceID"

Using cmdAnalytics As New OleDbCommand(analyticsQuery, cn)

cmdAnalytics.Parameters.AddWithValue("@DeviceID", CInt(txtdeviceID.Text))

ad = New OleDbDataAdapter(cmdAnalytics)

ds = New DataSet

ad.Fill(ds)

DataGridView1.DataSource = ds.Tables(0)

End Using

Catch ex As Exception

MsgBox("Error : " & ex.Message)

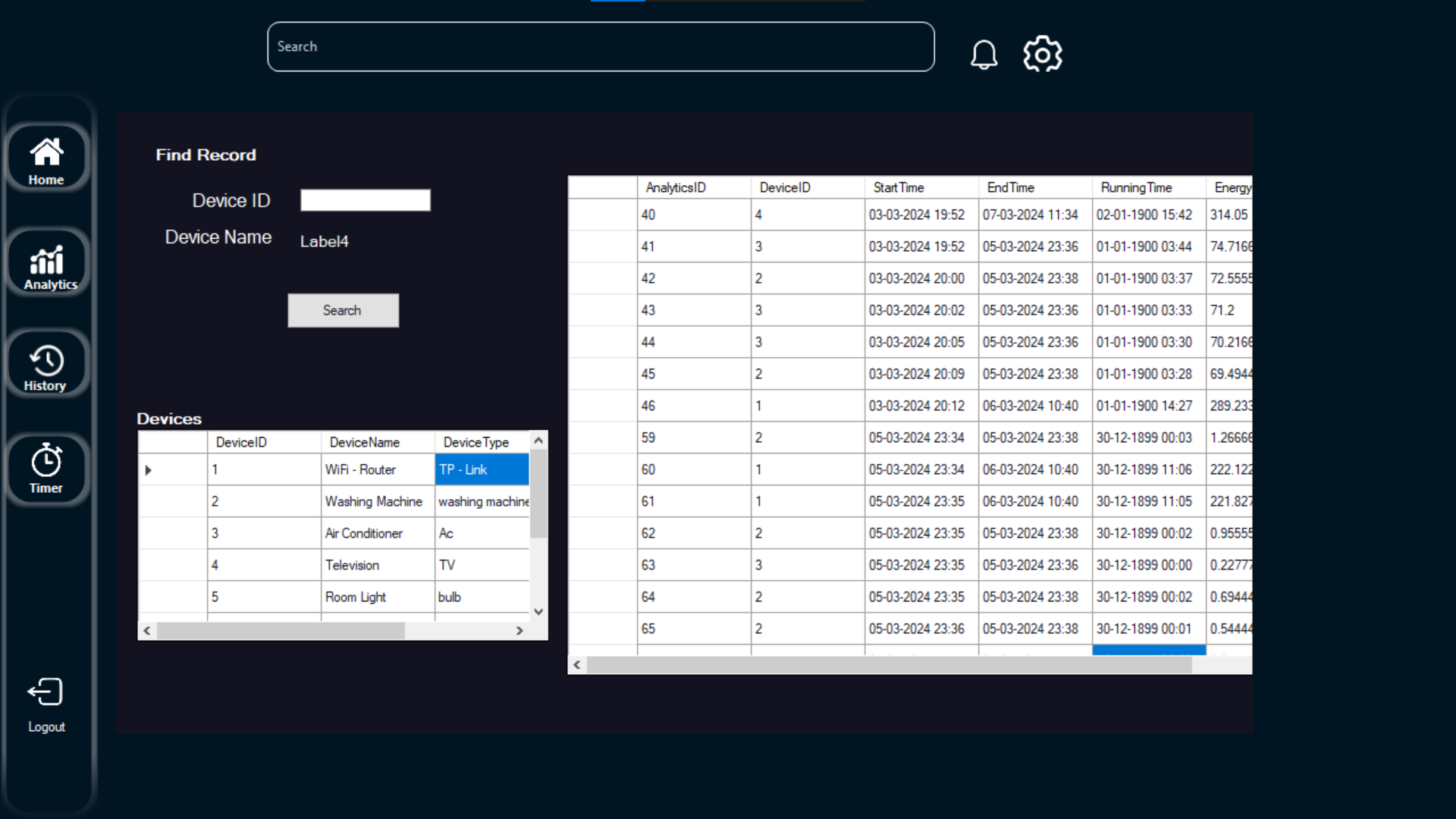
Finally

cn.Close()

End Try

End Sub

End Class

****

**Frmvoicecontrol.vb**

Imports System.Speech.Recognition

Imports System.Speech.Synthesis

Public Class frmvoicecontrol

Dim WithEvents recognizer As New SpeechRecognitionEngine()

Dim synthesizer As New SpeechSynthesizer()

Private Sub frmvoicecontrol\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

recognizer.SetInputToDefaultAudioDevice()

Dim grammarBuilder As New GrammarBuilder()

grammarBuilder.Append(New Choices("turn off television", "turn off ac", "how are you", "hello", "what is the time"))

Dim grammar As New Grammar(grammarBuilder)

recognizer.LoadGrammar(grammar)

recognizer.RecognizeAsync(RecognizeMode.Multiple)

End Sub

Private Sub recognizer\_SpeechRecognized(sender As Object, e As SpeechRecognizedEventArgs) Handles recognizer.SpeechRecognized

Dim recognizedText As String = e.Result.Text

If recognizedText.Contains("what is the time") Then

Label1.Text = "Sir now time is " & DateTime.Now().ToString()

synthesizer.Speak(Label1.Text)

Me.Close()

End If

If recognizedText.Contains("turn off ac") Then

Label1.Text = "turning off AC successfully"

synthesizer.Speak(Label1.Text)

Me.Close()

End If

If recognizedText.Contains("turn off television") Then

Label1.Text = "turning off television successfully"

synthesizer.Speak(Label1.Text)

Me.Close()

End If

If recognizedText.Contains("how are you") Then

Label1.Text = "I'm Fine sir"

synthesizer.Speak(Label1.Text)

Me.Close()

End If

End Sub

Private Sub PictureBox1\_Click(sender As Object, e As EventArgs) Handles PictureBox1.Click

Me.Close()

End Sub

End Class

****

**Frmtimer.vb**

Imports System.Data.OleDb

Imports System.Timers

Imports Guna.UI2.WinForms.Suite

Public Class frmTimer

Dim cn As New OleDbConnection

Dim cmd As OleDbCommand

Dim ds As New DataSet

Dim ad As OleDbDataAdapter

Dim dr As OleDbDataAdapter

Dim timerDictionary As New Dictionary(Of Integer, Timer)()

Dim timer As New Timer()

Private Sub frmTimer\_Load(sender As Object, e As EventArgs) Handles MyBase.Load

dtpTime.Format = DateTimePickerFormat.Custom

dtpTime.CustomFormat = "hh:mm tt" ' 12-hour format with AM/PM

dtpTime.ShowUpDown = True

cn.ConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data Source=D:\Study Materials\Jounral Projects\SmartHomeHub - Copy.accdb"

checkTime()

Displayrecords()

AddHandler timer.Elapsed, AddressOf TimerElapsed

timer.Interval = 500 ' Set the interval to 1 second (1000 milliseconds)

timer.Start()

End Sub

Private Sub TimerElapsed(sender As Object, e As ElapsedEventArgs)

checkTime()

End Sub

Public Sub setTimer(id As Integer, time As DateTime)

Dim query As String = "INSERT INTO Timers(DeviceID, TimerValue) VALUES (?, ?)"

cmd = New OleDbCommand(query, cn)

cmd.Parameters.AddWithValue("@DeviceID", id)

cmd.Parameters.AddWithValue("@TimerValue", time)

Try

cn.Open()

cmd.ExecuteNonQuery()

Catch ex As Exception

MsgBox("Error : " & ex.Message())

Finally

cn.Close()

End Try

End Sub

Private Sub btnset\_Click(sender As Object, e As EventArgs) Handles btnInsert.Click

Dim query As String = "INSERT INTO Timers(DeviceID, TimerValue) VALUES (?, ?)"

cmd = New OleDbCommand(query, cn)

Try

If txtDeviceID.Text = "" Then

MsgBox("Please Enter DeviceID")

Else

Dim deviceID As Integer = CInt(txtDeviceID.Text)

Dim selectedTime As DateTime = dtpTime.Value

cmd.Parameters.AddWithValue("@DeviceID", deviceID)

cmd.Parameters.AddWithValue("@TimerValue", selectedTime)

If selectedTime > DateTime.Now() Then

cn.Open()

cmd.ExecuteNonQuery()

MsgBox("Timer Added Successfully....")

Else

MsgBox("please select correct time")

End If

End If

Catch ex As Exception

MsgBox("Error : " & ex.Message())

Finally

cn.Close()

Displayrecords()

End Try

End Sub

Private Sub Displayrecords()

Dim query As String = "SELECT \* FROM Timers"

cmd = New OleDbCommand(query, cn)

Try

cn.Open()

ad = New OleDbDataAdapter(cmd)

ds = New DataSet

ad.Fill(ds)

ShowTimers.DataSource = ds.Tables(0)

For Each row As DataRow In ds.Tables(0).Rows

Dim deviceID As Integer = CInt(row("DeviceID"))

Dim targetTime As DateTime = Convert.ToDateTime(row("TimerValue"))

Next

Catch ex As Exception

MsgBox("Error : " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub checkTime()

Dim query As String = "SELECT TimerValue, TimerID FROM Timers"

Try

cn.Open()

cmd = New OleDbCommand(query, cn)

showtime.Text = DateTime.Now().ToString()

Dim reader As OleDbDataReader = cmd.ExecuteReader()

If reader.HasRows Then

While reader.Read()

Dim timerValue As DateTime = Convert.ToDateTime(reader("TimerValue"))

Dim timerID As Integer = Convert.ToInt32(reader("TimerID"))

If timerValue <= DateTime.Now Then

deleteTimer(timerID)

MsgBox("truning off devices Success.....")

End If

End While

End If

Catch ex As Exception

Finally

cn.Close()

End Try

End Sub

Private Sub ShowTimers\_CellContentClick(sender As Object, e As DataGridViewCellEventArgs) Handles ShowTimers.CellContentClick

Dim i As Integer

i = ShowTimers.CurrentRow.Index

txtDeviceID.Text = ShowTimers.Item(0, i).Value

dtpTime.Text = ShowTimers.Item(2, i).Value

End Sub

Private Sub btnUpdate\_Click(sender As Object, e As EventArgs) Handles btnUpdate.Click

cmd = New OleDbCommand("DELETE FROM Timers WHERE TimerID = " & CInt(txtDeviceID.Text) & "", cn)

Try

cn.Open()

cmd.ExecuteNonQuery()

MsgBox("record delated")

Displayrecords()

Catch ex As Exception

MsgBox("Error: " & ex.Message)

Finally

cn.Close()

End Try

End Sub

Private Sub deleteTimer(timerID As Integer)

Try

cn.Open()

cmd = New OleDbCommand("DELETE FROM Timers WHERE TimerID = @id", cn)

cmd.Parameters.AddWithValue("@id", timerID)

cmd.ExecuteNonQuery()

Finally

cn.Close()

End Try

End Sub

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

Try

cn.Open()

cmd = New OleDbCommand("update Timers set TimerValue = '" & dtpTime.Value & "' where TimerID = " & CInt(txtDeviceID.Text) & "", cn)

cmd.ExecuteNonQuery()

MsgBox("Record Updated SuccessFully....")

Catch ex As Exception

MsgBox("Error : " & ex.Message)

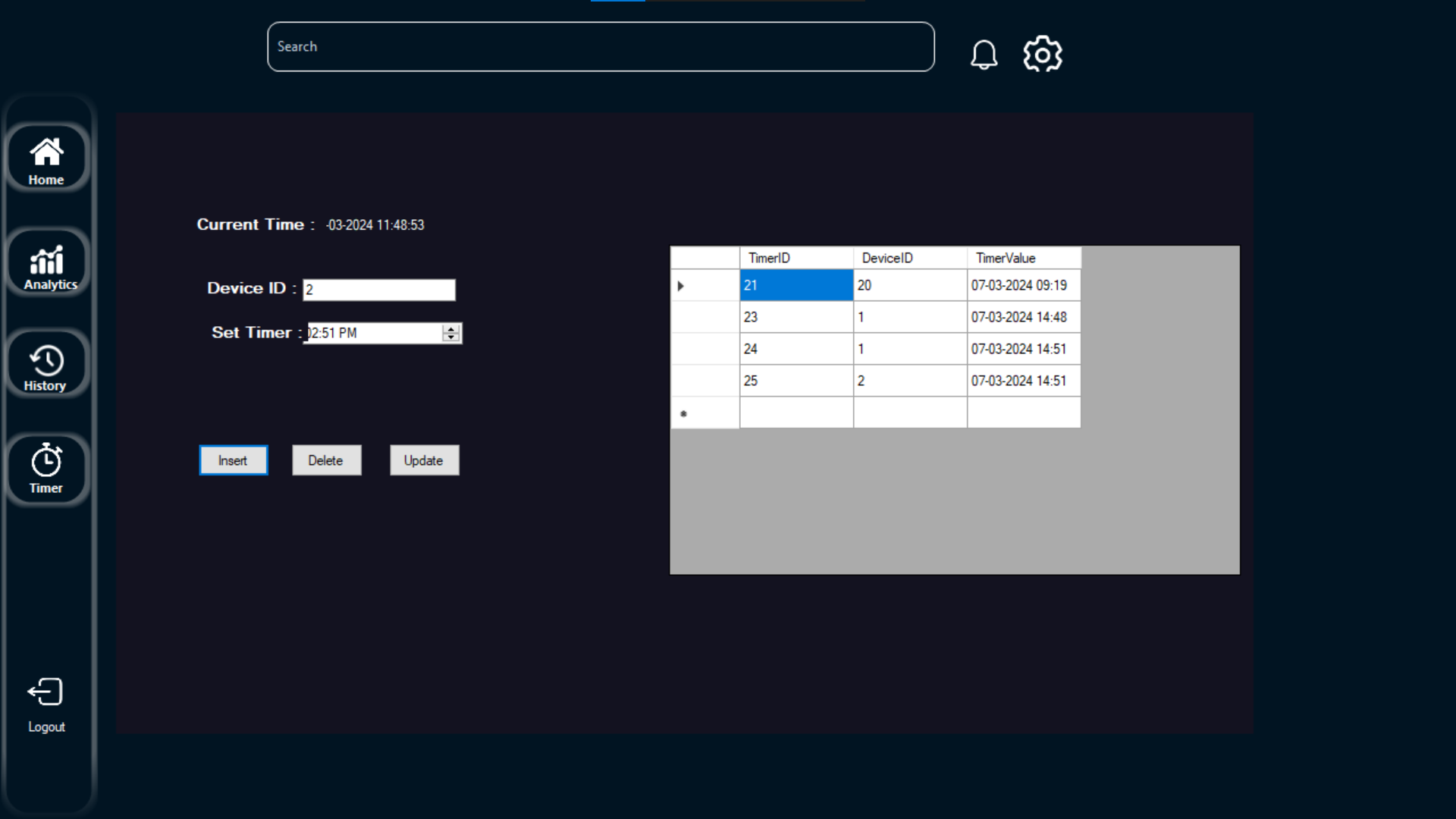
Finally

cn.Close()

End Try

End Sub

End Class

****