

**MEHR CHAND MAHAJAN DAV COLLEGE FOR WOMEN**  
**SECTOR-36A, CHANDIGARH**  
**DEPARTMENT OF COMPUTER SCIENCE AND APPLICATIONS**  
**PROJECT REPORT**  
**ON**  
**UBA-HUB**  
**(Unnat Bharat Abhiyan HUB)**  
**SESSION: APRIL 2024-25**



**SUBJECT: MAJOR PROJECT AND SEMINAR**

**SUBJECT CODE: BCA-16-605**

**SUBMITTED TO:**

**DR. NAVDEEP KAUR**

**PROFESSOR RITU PARAN**

**SUBMITTED BY:**

**AMANPREET KAUR GARCHA**

**(6700, 22049865)**

**SHEETAL**

**(6684, 22049937)**

## Acknowledgement

I take this opportunity to express my sincere gratitude to all those who have supported me directly or indirectly in the successful completion of my project titled “UBA-HUB.” This project has been a valuable part of my academic learning, allowing me to understand the real-world application of technology in business management.

We are deeply grateful to the Principal of Mehr Chand Mahajan DAV College for Women, Sector-36A, Chandigarh, for providing us with a conducive academic environment and the necessary infrastructure to carry out our project effectively.

Our heartfelt thanks also go to Dr. Indu Arora, the Head of the Department of Computer Science and Applications for their encouragement and support, and for fostering a culture of innovation and excellence in learning.

We would like to sincerely thank Dr. Navdeep Kaur and Professor Ritu Paran, our mentors, for their unwavering support, invaluable guidance, and continuous encouragement throughout the course of this research project. Their insights and motivation were instrumental in shaping the direction and outcome of our work.

We are also thankful to the Unnat Bharat Abhiyan (UBA) initiative, whose vision of community engagement and rural development inspired the core theme of our project.

Lastly, we would like to thank each other as teammates for the cooperation, shared learning, and strong teamwork that made this journey both enriching and enjoyable.

## Introduction / Overview

The **UBA HUB (Unnat Bharat Abhiyan Hub)** project is a desktop-based application designed and developed as a part of our Major Project for the Bachelor of Computer Applications (BCA) program. The objective of this project is to assist colleges participating in the Unnat Bharat Abhiyan (UBA) in managing and documenting the various rural development activities they undertake in collaboration with village communities.

Unnat Bharat Abhiyan is a flagship initiative of the Ministry of Education, Government of India, aimed at bringing transformational changes in rural development processes by leveraging knowledge institutions to help build inclusive India. As part of this initiative, higher educational institutions are assigned a set of villages to work with and are expected to conduct activities focused on education, health, sanitation, skill development, environmental awareness, and more.

Our project, **UBA HUB**, is developed using **VB.NET and MS Access** and acts as a centralized system that simplifies the management of all village activities. The system provides a structured way for multiple stakeholders—including village coordinators, organizing teams, and administrators—to input, update, and monitor data related to UBA activities.

Key features of the system include:

- Recording detailed information of each village activity such as coordinator details, village name, activity type, date, description, media evidence (images/videos), and final reporting descriptions.
- Role-based access through distinct forms such as the **Village\_Coordinator\_Team** form and **Organising\_Team** form.
- A centralized **Activities** form that allows users to search, filter, update, and delete records, along with the ability to generate well-formatted **PDF reports using Crystal Reports**.
- A user-friendly dashboard (**UBA HUB**) that allows smooth navigation between modules.
- Visual documentation and proper database structure ensuring data consistency and easy expansion in the future.

The primary motivation behind this system was to reduce manual documentation and streamline the collection and processing of outreach data in a consistent digital format. It was also designed keeping scalability and usability in mind so that it could be adopted across similar institutions working under the UBA scheme.

Through the development of this project, we were able to put into practice our understanding of software engineering principles, user interface design, database management, and practical coding skills. This project not only fulfilled academic requirements but also allowed us to contribute meaningfully to a real-world social cause.





## System Study Details

Contact Person: Dr. Navdeep Kaur

Address: Mehr Chand Mahajan DAV College for Women, Sector-36A, Chandigarh

Phone: 9988097754

## Sample Brochures/Forms Collected:

			
<b>UNNAT BHARAT ABHIYAN</b>			
Mehr Chand Mahajan DAV College for Women, Chandigarh			
<b>PROGRESS REPORT</b>			
Village			
UBA Coordinator's Name: Dr Gunjan			
Email: <a href="mailto:ubacellatmcm@gmail.com">ubacellatmcm@gmail.com</a>			
Phone Number: 9915703134			
Sr. No.	ADOPTED VILLAGES	TALUKA	DISTRICT
1	Attawa		
2	Badheri		
3	Maloya		
4	Kajheri		
5	Buterla		
Name of Activity:			
Date of Activity:			
Need of the Activity:			
Description in 200 words (along with the Pictures):			



**Photographs of the Activity:**

**Note: Detailed Village-wise reports also uploaded on UBA Portal.**

**Action plan for next month:**

Sr. No.	Activity to be conducted(along with reason)
1	
2	
3	

## **Modules to be covered**

1. UBA HUB Dashboard - Navigation form for accessing all other modules.
2. Village Coordinator Team Form - Data entry form for village activities, coordinator details, media uploads, etc.
3. Input Dialog Box - Facilitates record selection based on Village, Activity and Date.
4. Organising Team Form - Interface for team leads to update 'Need' and 'Final Description' fields only.
5. Activities Form - Filter, update, delete and generate PDF reports using Crystal Reports.

### Database Design

The database used is Microsoft Access (.accdb format). The primary table is `UBA\_Drives\_Data` with the following fields:

Field Name	Data Type	Description
ID	Number	Primary Key
Coordinator	Short Text	Name of the coordinator
Village	Short Text	Village name
Activity	Short Text	Activity type
Date_of_act	Date/Time	Date of the activity
Details	Long Text	Detailed activity information
Email_ID	Short Text	Coordinator's email
Need	Long Text	Need for the activity
Raw_Description	Long Text	Initial raw description
Final_Description	Long Text	Finalized report description
Image1	Long Text	First Best image upload
Image2	Long Text	Second Best image upload
Image3	Long Text	Third Best image upload
Video	Long Text	Video file upload



## Screen Design of All Forms

### UBA HUB (Main Dashboard):



Description:

The **Main Page** of the UBA HUB application, named `formMain`, serves as the **central navigation dashboard** that connects all other core modules of the system. Designed with a clean and minimal interface, this form provides quick access to three key sections of the project:

#### Buttons and Their Functionalities:

1. **Village Coordinator Button (`btnVillage_HOME`)**
  - **Function:** Opens the `Village_Coordinator_Team` form.
  - **Purpose:** Allows coordinators to enter detailed information about village activities including name, village, type of activity, date, description, and multimedia (images/videos).

## 2. Organising Team Button (btnOrg\_Home)

- **Function:** Opens the InputDialogBox, which further leads to the Organising\_Team form.
- **Purpose:** Enables organizing team members to select a specific record (based on village, activity, and date) and update the **Need** and **Final Description** fields of that activity.

## 3. Activities Button (btnActivity\_HOME)

- **Function:** Opens the Activities form.
- **Purpose:** This is where users can **view, filter, edit, delete, and generate reports** for existing UBA drive entries.

### Village Coordinator Team Form:

Village Coordinator Team

उन्नत भारत अभियान  
शिक्षा मंत्रालय, भारत सरकार का एक प्रमुख कार्यक्रम  
**UNNAT BHARAT ABHIYAN (UBA)**  
a flagship program of Ministry of Education (MoE), GOI  
शिक्षित भारत-स्वस्थ भारत- स्वच्छ भारत- स्वावलंबी भारत- संपन्न भारत

Coordinator

Village

Activity

Date

Email

Description

Best Image 1

Best Image 2

Best Image 3

Video Attachment

Details

Description:

Mehr Chand Mahajan College for Women, Section-36A, Chandigarh

## Village\_Coordinator\_Team – Data Entry Form for Village Coordinators

The Village\_Coordinator\_Team form plays a crucial role in the **UBA HUB** system. It is designed specifically for **village coordinators** to input detailed information regarding outreach or developmental activities conducted under the **Unnat Bharat Abhiyan** initiative.

This form is highly interactive and allows both **textual** and **multimedia input**, including images and videos related to each field activity.

### Key Features and Components:

#### Text Inputs:

- **Coordinator Name:** Dropdown selection (cmbxCoordinator\_VC)
- **Village:** Dropdown selection (cmbxVillage\_VC)
- **Activity:** Text field (txtActivity\_VC)
- **Raw Description:** Text field describing the basic idea or intent of the activity (txtDesc\_VC)
- **Details:** Additional elaboration on implementation or observations (txtDetails\_VC)
- **Email:** Email address of the coordinator (txtEmail\_VC)
- **Date of Activity:** Date selector (DateTimePicker\_VC)

#### Media Upload Capabilities:

- **Image 1, 2, and 3 Upload:**
  - Buttons: btnUploadImage1\_VC, btnUploadImage2\_VC, btnUploadImage3\_VC
  - Displayed in: picbxImage1\_VC, picbxImage2\_VC, picbxImage3\_VC
  - Purpose: Upload photographs taken during fieldwork
- **Video Upload:**
  - Button: btnUploadVideo\_VC
  - Label: lblNewVideo\_VC shows success message after upload
  - Purpose: Attach supporting field video

#### Reset Button (btnReset\_VC)

- Clears all fields including text inputs, image previews, and resets media file paths
- Useful for starting a new fresh entry

### Submit Button (btnSubmit\_vc)

- Validates all mandatory fields before submission
- On success, inserts the complete activity record into the MS Access database (UBA\_Dtbse.accdb) under the UBA\_Drives\_Data table
- Shows a success message upon completion or appropriate error message on failure
- Closes the form post-submission

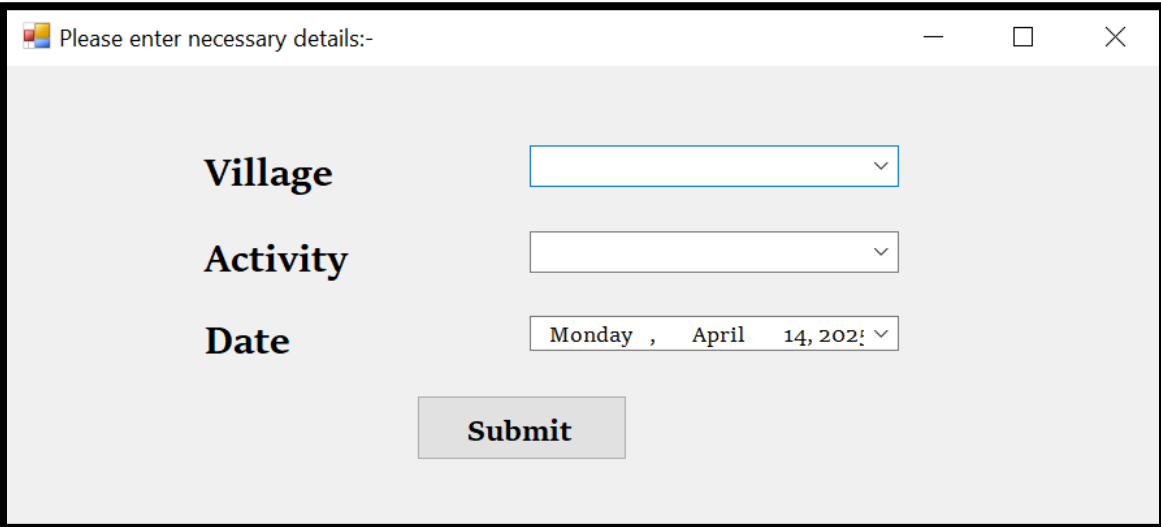
### Validation Logic:

- Ensures no field is left blank or unselected
- Escapes single quotes to prevent database errors

### Database Fields Affected:

- Coordinator, Village, Activity, Date\_of\_act, Raw\_Description, Details, Image1, Image2, Image3, Video, Email\_ID

### Search Form:

A screenshot of a Windows-style dialog box titled "Please enter necessary details:-". The dialog box has a light gray background and a black border. It contains three labels on the left: "Village", "Activity", and "Date". To the right of each label is a dropdown menu. The "Date" dropdown is currently showing "Monday , April 14, 2025". Below these fields is a "Submit" button.

Description:

### Search Form

The **Search Form**, formally named `InputDialogBox`, plays an essential role in navigating and filtering data within the UBA HUB application. It allows the user—typically from the Organising Team—to **select a specific record** for editing based on three key fields: **Village**, **Activity**, and **Date**. This ensures that users are always directed

to the correct and unique entry in the database, thus **enhancing accuracy and efficiency** in data handling.

### Objective and Functionality

The main objective of the Search Form is to act as a **user-friendly filter** that collects search criteria and redirects the user to a more detailed view of the selected activity record. Once the values are selected, they are passed as parameters to the `Organising_Team` form, which displays the full record and allows limited updates.

### Form Components

- **ComboBox: cmbxVillage\_INPUT**  
Allows the user to select the name of the village. (Manually populated or through pre-defined values)
- **ComboBox: cmbxAct\_INPUT**  
Populated dynamically from the database with unique activities using a `SELECT DISTINCT` query. It includes autocomplete features for improved user experience.
- **DateTimePicker: DateTimePicker\_INPUT**  
Enables the user to select the specific date of the activity conducted.
- **Button: btnSubmit\_INPUT**  
On clicking this button, the selected village, activity, and date are stored in shared public variables. The `Organising_Team` form is then opened with these values for fetching the correct record.

### Backend Logic

On form load, the system connects to the Access database and retrieves a list of all distinct activity names from the `UBA_Drives_Data` table. This dynamic loading ensures that the activities listed are always up to date and relevant to the existing data.

## Organising Team Form:

Organising Team



UNNAT BHARAT ABHIYAAN  
Mehr Chand Mahajan DAV College for Wom, Chandigarh

PROGRESS REPORT

(Village Name)

Coordinator	(Coordinator Name)
Activity	(Name of activity)
Date	(Date of activity)
Email	(Email Id)
Raw Description	<input type="text"/>
Need of activity	<input type="text"/>
Final Description	<input type="text"/>

Reset

Submit

Description:

### Organising Team Form – *Organising\_Team*

The **Organising Team Form** is designed for internal members of the UBA HUB project who are responsible for **reviewing and finalizing field reports**. This form displays full activity records filtered by **Village, Activity, and Date**, which are selected via the Search Form. It allows authorized users to **view all details** and **edit specific fields** such as *Need* and *Final Description*.

### Objective and Purpose

The goal of this form is to provide a **structured interface** where users can review and finalize activity reports submitted by coordinators. It allows editing of fields necessary for generating **progress reports**, while maintaining the integrity of the original "Raw Description."

### Form Components and Features

- **Labels:**
  - Display readonly details like:
    - Village, Activity, Coordinator, Date, and Email.
- **Textbox: txtDescDisplay\_ORG**
  - Displays the *Raw Description* submitted earlier.
  - It is **editable for user reference only**, but any changes made **are not saved** to the database.
- **Textbox: txtNeed\_ORG and txtFinalDescription\_ORG**
  - These are **editable** fields used for updating the final report content.
- **Buttons:**
  - **Submit:** Saves only *Need* and *Final Description* back to the database.
  - **Reset:** Clears the editable fields to allow re-entry.

## Activities Form:

The screenshot shows a web application window titled "Activities". The header features a banner with images of rural development and the text "UNNAT BHARAT ABHIYAN" along with a quote from Sri Narendra Modi. Below the banner, there are three dropdown menus for filtering: "Village" (set to "ALL"), "Activity" (set to "ALL"), and "Year" (set to "ALL"). There are also "Reset" and "Show" buttons. Below the filters is a table with the following headers: "Village", "Activity", "Date", and "Description". The table body is currently empty. At the bottom of the form, there are four buttons: "View Whole Details", "Update", "Delete", and "Upload Report".

Village	Activity	Date	Description
---------	----------	------	-------------

Description:

### Activities Form – Module Description

The **Activities Form** is a core component of the UBA HUB project that enables users to view, filter, update, and manage activity records stored in the MS Access database (UBA\_Dtbse.accdb). It provides a user-friendly interface for handling Unnat Bharat Abhiyan (UBA) data entries based on **Village**, **Activity**, and **Year**.



### **Key Features:**

- **Dynamic Filtering**  
On form load, the dropdowns for Village, Activity, and Year are populated dynamically using distinct values from the database. Users can filter records using any combination of these fields or choose **ALL** to see unfiltered data.
- **Grid Display**  
Upon clicking the **Show** button, a DataGridView (`Grid_ACT`) displays relevant records with columns: Village, Activity, Date, Year, and Final Description. The "ID" column is hidden for clarity, and the "Year" column is read-only to prevent accidental changes.
- **Reset Functionality**  
The **Reset** button clears all filters and the data grid, allowing users to start a fresh search.
- **Record Viewing**  
With the **View Details** button, users can view the full contents of the selected record in a MessageBox, ensuring quick access to complete information.
- **Record Updating**  
The form supports inline editing of fields directly within the grid. When the **Update** button is clicked, all rows are looped through, and changes to Village, Activity, Date, and Description fields are updated in the database via parameterized SQL queries.
- **Record Deletion**  
The **Delete** button enables users to remove selected records after confirmation. This ensures accidental deletions are avoided.
- **Crystal Report Generation**  
A key feature of the Activities form is the **Upload Report** button, which generates a formatted **Crystal Report** (`UBA-Hub_Report.rpt`) based on the selected row. The report layout adheres to a predefined template and opens in a Crystal Report Viewer for preview or printing. This module also includes support for exporting the report to PDF format (commented-out lines provided for future extension).

## Coding of Important Forms

### Village\_Coordinator\_Team Form:

Handles data entry for coordinators. Supports image/video upload. Saves to `UBA\_Drives\_Data`.

```
Public Class Village_Coordinator_Team

    Dim conn As System.Data.OleDb.OleDbConnection
    Dim cmd As System.Data.OleDb.OleDbCommand
    Dim reader As System.Data.OleDb.OleDbDataReader

    Dim Img1 As String = ""
    Dim Img2 As String = ""
    Dim Img3 As String = ""
    Dim Video As String = ""

    'RESET
    Private Sub btnReset_VC_Click(sender As Object, e As EventArgs) Handles
        btnReset_VC.Click
            txtEmail_VC.Text = ""
            cmbxCoordinator_VC.Text = ""
            cmbxVillage_VC.Text = ""
            txtActivity_VC.Text = ""
            txtDesc_VC.Text = ""
            txtDetails_VC.Text = ""
            Img1 = ""
            Img2 = ""
            Img3 = ""
            Video = ""
            picbxImage1_VC.Image = Nothing
            picbxImage2_VC.Image = Nothing
            picbxImage3_VC.Image = Nothing
            lblNewVideo_VC.Text = ""
        End Sub

    'SUBMIT
    Private Sub btnSubmit_VC_Click(sender As Object, e As EventArgs) Handles
        btnSubmit_VC.Click
            ' Input validation
            If cmbxCoordinator_VC.SelectedIndex = -1 OrElse
                cmbxVillage_VC.SelectedIndex = -1 OrElse
                String.IsNullOrEmpty(txtActivity_VC.Text) OrElse
                String.IsNullOrEmpty(txtDesc_VC.Text) OrElse
                String.IsNullOrEmpty(txtDetails_VC.Text) OrElse
                String.IsNullOrEmpty(txtEmail_VC.Text) Then

                MessageBox.Show("Please fill in all the fields before submitting.",
                    "Missing Data", MessageBoxButtons.OK, MessageBoxIcon.Warning)
                Exit Sub
            End If

            Try
                conn = New OleDb.OleDbConnection
                conn.ConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb"
```

```

        Dim query As String = "INSERT INTO UBA_Drives_Data (Coordinator,
Village, Activity, Date_of_act, Raw_Description, Details, Image1, Image2, Image3,
Video, Email_ID) " &
        "VALUES (" &
cmbxCoordinator_VC.SelectedItem.ToString().Replace("'", "'") & ", '" &
cmbxVillage_VC.SelectedItem.ToString().Replace("'", "'") &
        ", '" &
        txtActivity_VC.Text.Replace("'", "'") & ", #" &
        DateTimePicker_VC.Value & "#, '" &
        txtDesc_VC.Text.Replace("'", "'") & ", '" &
        txtDetails_VC.Text.Replace("'", "'") & ", '" &
        Img1 & ", '" & Img2 & ", '" & Img3 & ", '" & Video & ",
        '" &
        txtEmail_VC.Text.Replace("'", "'") & "'"")"

        cmd = New OleDb.OleDbCommand(query, conn)

        conn.Open()
        cmd.ExecuteNonQuery()
        conn.Close()

        MessageBox.Show("Data submitted successfully!", "Success",
        MessageBoxButtons.OK, MessageBoxIcon.Information)
        Me.Close()

        Catch ex As Exception
            MessageBox.Show("Error submitting data: " & ex.Message, "Submission
Failed", MessageBoxButtons.OK, MessageBoxIcon.Error)
        End Try
    End Sub

'IMAGE 1 UPLOAD BUTTON
Private Sub btnUploadImage1_VC_Click(sender As Object, e As EventArgs) Handles
btnUploadImage1_VC.Click
    Dim OpenFileDialog1_VC As New OpenFileDialog()
    If OpenFileDialog1_VC.ShowDialog() = DialogResult.OK Then
        Img1 = OpenFileDialog1_VC.FileName
        picbxImage1_VC.Image = Image.FromFile(Img1)
    End If
End Sub

'IMAGE 2 UPLOAD BUTTON
Private Sub btnUploadImage2_VC_Click(sender As Object, e As EventArgs) Handles
btnUploadImage2_VC.Click
    Dim OpenFileDialog2_VC As New OpenFileDialog()
    If OpenFileDialog2_VC.ShowDialog() = DialogResult.OK Then
        Img2 = OpenFileDialog2_VC.FileName
        picbxImage2_VC.Image = Image.FromFile(Img2)
    End If
End Sub

'IMAGE 3 UPLOAD BUTTON
Private Sub btnUploadImage3_VC_Click(sender As Object, e As EventArgs) Handles
btnUploadImage3_VC.Click
    Dim OpenFileDialog3_VC As New OpenFileDialog()

```

```

        If OpenFileDialog3_VC.ShowDialog() = DialogResult.OK Then
            Img3 = OpenFileDialog3_VC.FileName
            picbxImage3_VC.Image = Image.FromFile(Img3)
        End If
    End Sub

    'VIDEO UPLOAD BUTTON
    Private Sub btnUploadVideo_VC_Click(sender As Object, e As EventArgs) Handles
        btnUploadVideo_VC.Click
        Dim OpenFileDialog4 As New OpenFileDialog()
        If OpenFileDialog4_VC.ShowDialog() = DialogResult.OK Then
            Video = OpenFileDialog4_VC.FileName
            If Video = OpenFileDialog4_VC.FileName Then
                lblNewVideo_VC.Text = "VIDEO UPLOADED SUCCESSFULLY!!"
            End If
        End If
    End Sub
End Class

```

### Search Form:

Helps select Village, Activity and Date to load Organising\_Team form.

```
Imports System.Data.OleDb
```

```
Public Class InputDialogBox
```

```

    Public Shared SelectedVillage As String
    Public Shared SelectedActivity As String
    Public Shared SelectedDate As Date

```

```
'SUBMIT BUTTON
```

```

    Private Sub btnSubmit_INPUT_Click(sender As Object, e As EventArgs) Handles
        btnSubmit_INPUT.Click

```

```

        SelectedVillage = cmbxVillage_INPUT.SelectedItem.ToString()
        SelectedActivity = cmbxAct_INPUT.SelectedItem.ToString()
        SelectedDate = DateTimePicker_INPUT.Value

```

```

        Dim frm As New Organising_Team(selectedVillage, selectedActivity,
        selectedDate)
        frm.Show()
        Me.Hide()
    End Sub

```

```
'SUB FOR ACTIVITY OPTION OF COMBOBOX
```

```
Private Sub LoadActivityList()
```

```

    Dim conn As System.Data.OleDb.OleDbConnection
    Dim cmd As System.Data.OleDb.OleDbCommand
    Dim reader As System.Data.OleDb.OleDbDataReader

```

```

    Dim queryLOAD_ACT As String = "SELECT DISTINCT Activity FROM
    UBA_Drives_Data ORDER BY Activity"

```

```
    conn = New OleDb.OleDbConnection
```

```

        conn.ConnectionString = "Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb"

        cmbxAct_INPUT.Items.Clear()
        cmbxAct_INPUT.AutoCompleteCustomSource.Clear()

        cmd = New OleDb.OleDbCommand(queryLOAD_ACT, conn)

        conn.Open()
        reader = cmd.ExecuteReader()

        While reader.Read()
            Dim activityName As String = reader("Activity").ToString()
            cmbxAct_INPUT.Items.Add(activityName)
            cmbxAct_INPUT.AutoCompleteCustomSource.Add(activityName)
        End While

        conn.Close()
    End Sub

    'ON LOAD CALL THE SUB ON FORM LOAD
    Private Sub InputDialogBox_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        LoadActivityList()
    End Sub

End                                         Class

```

### Organising\_Team Form:

Only allows updating 'Need' and 'Final Description'. Warns if Raw Description is edited.

```

Imports System.Data.OleDb

Public Class Organising_Team
    Private originalRawDesc As String = ""
    Private village As String
    Private activity As String
    Private activityDate As Date

    Dim conn As OleDbConnection
    Dim cmd As OleDbCommand
    Dim reader As OleDbDataReader

    ' CONSTRUCTOR TO RECEIVE VALUES
    Public Sub New(selectedVillage As String, selectedActivity As String,
selectedDate As Date)
        InitializeComponent()
        village = selectedVillage
        activity = selectedActivity
        activityDate = selectedDate
    End Sub

    ' RESET BUTTON

```

```

Private Sub btnReset_ORG_Click(sender As Object, e As EventArgs) Handles
btnReset_ORG.Click
    txtNeed_ORG.Text = ""
    txtFinalDescription_ORG.Text = ""
End Sub

' SUBMIT BUTTON
Private Sub btnSubmit_ORG_Click(sender As Object, e As EventArgs) Handles
btnSubmit_ORG.Click
    ' Check if Raw Description was modified
    If txtDescDisplay_ORG.Text.Trim() <> originalRawDesc.Trim() Then
        MessageBox.Show("You modified 'Raw Description', but it will NOT be
saved to the database." & vbCrLf &
            "Only 'Need' and 'Final Description' will be
updated.", "Warning", MessageBoxButtons.OK, MessageBoxIcon.Warning)
    End If

    Dim need As String = txtNeed_ORG.Text.Trim()
    Dim finalDesc As String = txtFinalDescription_ORG.Text.Trim()

    conn = New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb")

    Dim querySubmit As String = "UPDATE UBA_Drives_Data SET [Need] = ?,
[Final_Description] = ? WHERE [Village] = ? AND [Activity] = ? AND [Date_of_act] =
?"
    cmd = New OleDbCommand(querySubmit, conn)

    cmd.Parameters.AddWithValue("?", need)
    cmd.Parameters.AddWithValue("?", finalDesc)
    cmd.Parameters.AddWithValue("?", village)
    cmd.Parameters.AddWithValue("?", activity)
    cmd.Parameters.AddWithValue("?", activityDate)

    conn.Open()
    Dim rowsAffected As Integer = cmd.ExecuteNonQuery()
    conn.Close()

    If rowsAffected > 0 Then
        MessageBox.Show("Report submitted successfully!" & vbCrLf &
            "Note: Only 'Need' and 'Final Description' were
saved." & vbCrLf &
            "'Raw Description' is shown for reference only and was
not updated.", "Success", MessageBoxButtons.OK, MessageBoxIcon.Information)
    Else
        MessageBox.Show("Failed to update the record.", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error)
    End If

    Me.Close()
End Sub

' ON LOAD EVENT - Load data into form
Private Sub Organising_Team_Load(sender As Object, e As EventArgs) Handles
 MyBase.Load
    conn = New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb")

```

```

        Dim queryLoad As String = "SELECT * FROM UBA_Drives_Data WHERE Village=?
AND Activity=? AND [Date_of_act]=?"
        cmd = New OleDbCommand(queryLoad, conn)

        cmd.Parameters.AddWithValue("?", village)
        cmd.Parameters.AddWithValue("?", activity)
        cmd.Parameters.AddWithValue("?", activityDate)

        conn.Open()
        reader = cmd.ExecuteReader()

        If reader IsNot Nothing AndAlso reader.Read() Then
            lblVillageDisplay_ORG.Text = reader("Village").ToString()
            lblCoordinatorDisplay_ORG.Text = reader("Coordinator").ToString()
            lblActivityDisplay_ORG.Text = reader("Activity").ToString()
            lblDateDisplay_ORG.Text = reader("Date_of_act").ToString()
            lblEmailDisplay_ORG.Text = reader("Email_ID").ToString()

            txtDescDisplay_ORG.Text = reader("Raw_Description").ToString()
            originalRawDesc = txtDescDisplay_ORG.Text ' Store original value for
later comparison

            txtNeed_ORG.Text = reader("Need").ToString()
            txtFinalDescription_ORG.Text = reader("Final_Description").ToString()
        Else
            MessageBox.Show("No data found for the selected input.", "Not Found",
MessageBoxButtons.OK, MessageBoxIcon.Warning)
            Me.Close()
        End If

        conn.Close()

        ' Allow user to view/edit Raw Description (but it won't be saved)
        txtDescDisplay_ORG.ReadOnly = False
    End Sub
End Class

```

### Activities Form:

Allows filtering, updating, deleting records. Supports Crystal Report PDF generation.

```
Imports System.Data.OleDb
```

```
Imports CrystalDecisions.CrystalReports.Engine
```

```
Imports CrystalDecisions.Shared
```

```
Public Class Activities
```

```
Dim conn As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb")
```

```
Dim dt As New DataTable()
```

```
Dim da As New OleDbDataAdapter()
```

```
Dim reader As OleDbDataReader
```

```
' ON LOAD
```

```
Private Sub Activities_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
```

```
    FillVillageComboBox()
```

```
    FillActivityComboBox()
```

```
    FillYearComboBox()
```

```
End Sub
```

```
' COMBOBOX FILLERS
```

```
Private Sub FillVillageComboBox()
```

```
    cmbxVillage_ACT.Items.Clear()
```

```
    cmbxVillage_ACT.Items.Add("ALL")
```

```
    Dim cmd As New OleDbCommand("SELECT DISTINCT Village FROM UBA_Drives_Data
ORDER BY Village", conn)
```

```
    conn.Open()
```

```
    reader = cmd.ExecuteReader()
```

```
    While reader.Read()
```

```
        cmbxVillage_ACT.Items.Add(reader("Village").ToString())
```

```
    End While
```

```
    conn.Close()
```

```
    cmbxVillage_ACT.SelectedItem = "ALL"
```

```
End Sub
```

```
Private Sub FillActivityComboBox()
```



```

        cmbxActivity_ACT.Items.Clear()

        cmbxActivity_ACT.Items.Add("ALL")

        Dim cmd As New OleDbCommand("SELECT DISTINCT Activity FROM UBA_Drives_Data
ORDER BY Activity", conn)

        conn.Open()

        reader = cmd.ExecuteReader()

        While reader.Read()

            cmbxActivity_ACT.Items.Add(reader("Activity").ToString())

        End While

        conn.Close()

        cmbxActivity_ACT.SelectedItem = "ALL"

End Sub

```

```

Private Sub FillYearComboBox()

    cmbxYear_ACT.Items.Clear()

    cmbxYear_ACT.Items.Add("ALL")

    Dim cmd As New OleDbCommand("SELECT DISTINCT Year(Date_of_act) FROM
UBA_Drives_Data ORDER BY Year(Date_of_act) DESC", conn)

    conn.Open()

    reader = cmd.ExecuteReader()

    While reader.Read()

        cmbxYear_ACT.Items.Add(reader(0).ToString())

    End While

    conn.Close()

    cmbxYear_ACT.SelectedItem = "ALL"

End Sub

```

```

' RESET BUTTON

```

```

Private Sub btnReset_ACT_Click(sender As Object, e As EventArgs) Handles
btnReset_ACT.Click

```

```

cmbxVillage_ACT.SelectedItem = "ALL"

cmbxActivity_ACT.SelectedItem = "ALL"

cmbxYear_ACT.SelectedItem = "ALL"

Grid_ACT.DataSource = Nothing

Grid_ACT.Columns.Clear()

dt.Clear()

End Sub

```

```

' SHOW BUTTON

```

```

Private Sub btnShow_ACT_Click(sender As Object, e As EventArgs) Handles
btnShow_ACT.Click

```

```

    Dim villageFilter As String = cmbxVillage_ACT.Text.Trim()

```

```

    Dim activityFilter As String = cmbxActivity_ACT.Text.Trim()

```

```

    Dim yearFilter As String = cmbxYear_ACT.Text.Trim()

```

```

    Dim query As String = "SELECT ID, Village, Activity, Date_of_act AS
[Date], Year(Date_of_act) AS [Year], Final_Description AS [Description] FROM
UBA_Drives_Data WHERE 1=1"

```

```

    If villageFilter <> "ALL" Then

```

```

        query &= " AND Village = ?"

```

```

    End If

```

```

    If activityFilter <> "ALL" Then

```

```

        query &= " AND Activity = ?"

```

```

    End If

```

```

    If yearFilter <> "ALL" Then

```

```

        query &= " AND Year(Date_of_act) = ?"

```

```

    End If

```

```
Using conn As New OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb")
```

```
Using cmd As New OleDbCommand(query, conn)
```

```
If villageFilter <> "ALL" Then
```

```
    cmd.Parameters.AddWithValue("?", villageFilter)
```

```
End If
```

```
If activityFilter <> "ALL" Then
```

```
    cmd.Parameters.AddWithValue("?", activityFilter)
```

```
End If
```

```
If yearFilter <> "ALL" Then
```

```
    cmd.Parameters.AddWithValue("?", yearFilter)
```

```
End If
```

```
Dim dt As New DataTable()
```

```
Dim adapter As New OleDbDataAdapter(cmd)
```

```
adapter.Fill(dt)
```

```
Grid_ACT.Columns.Clear()
```

```
Grid_ACT.DataSource = dt
```

```
Grid_ACT.AutoSizeColumnsMode =
DataGridViewAutoSizeColumnsMode.AllCells
```

```
If Grid_ACT.Columns.Contains("Date") Then
```

```
    Grid_ACT.Columns("Date").DefaultCellStyle.Format = "dd-MMM-
yyyy"
```

```
End If
```

```
If Grid_ACT.Columns.Contains("ID") Then
```

```
    Grid_ACT.Columns("ID").Visible = False
```

```
End If
```

```

        End Using

    End Using

End Sub

' VIEW DETAILS

Private Sub btnViewDetails_ACT_Click(sender As Object, e As EventArgs) Handles
btnViewDetails_ACT.Click

    If Grid_ACT.SelectedRows.Count = 0 Then

        MessageBox.Show("Please select a row to view details.", "No Row
Selected", MessageBoxButtons.OK, MessageBoxIcon.Information)

        Exit Sub

    End If

    Dim selectedRow As DataGridViewRow = Grid_ACT.SelectedRows(0)

    Dim details As String = ""

    For Each cell As DataGridViewCell In selectedRow.Cells

        If cell.Visible AndAlso cell.Value IsNot Nothing Then

            details &= Grid_ACT.Columns(cell.ColumnIndex).HeaderText & ": " &
cell.Value.ToString() & Environment.NewLine

        End If

    Next

    MessageBox.Show(details, "Full Record Details", MessageBoxButtons.OK,
MessageBoxIcon.Information)

End Sub

' UPDATE BUTTON

Private Sub btnUpdate_ACT_Click(sender As Object, e As EventArgs) Handles
btnUpdate_ACT.Click

    Dim updatedCount As Integer = 0

    Dim notUpdatedList As New List(Of String)

```

```

For Each row As DataGridViewRow In Grid_ACT.Rows

    If row.IsNewRow Then Continue For

    Dim newVillage = row.Cells("Village").Value.ToString().Trim()

    Dim newActivity = row.Cells("Activity").Value.ToString().Trim()

    Dim newDate = Convert.ToDateTime(row.Cells("Date").Value).Date

    Dim newDesc = row.Cells("Description").Value.ToString().Trim()

    Dim recordID = Convert.ToInt32(row.Cells("ID").Value)

    Dim query As String = "UPDATE UBA_Drives_Data SET Village = ?,
Activity = ?, Date_of_act = ?, Final_Description = ? WHERE ID = ?"

    Using con As New OleDbConnection(conn.ConnectionString)

        con.Open()

        Using cmd As New OleDbCommand(query, con)

            cmd.Parameters.AddWithValue("?", newVillage)

            cmd.Parameters.AddWithValue("?", newActivity)

            cmd.Parameters.AddWithValue("?", newDate)

            cmd.Parameters.AddWithValue("?", newDesc)

            cmd.Parameters.AddWithValue("?", recordID)

            Dim rowsAffected = cmd.ExecuteNonQuery()

            If rowsAffected > 0 Then

                updatedCount += 1

            Else

                notUpdatedList.Add($"{newVillage},           {newActivity},
{newDate.ToShortDateString()}")

            End If

        End Using

    End Using

```

```

        End Using

    Next

    If notUpdatedList.Count > 0 Then

        MessageBox.Show("Changes were not updated:" & vbCrLf &
String.Join(vbCrLf, notUpdatedList), "Warning", MessageBoxButtons.OK,
MessageBoxIcon.Warning)

    Else

        MessageBox.Show("Changes are updated successfully.", "Update",
MessageBoxButtons.OK, MessageBoxIcon.Information)

    End If

End Sub

' DELETE BUTTON

Private Sub btnDelete_ACT_Click(sender As Object, e As EventArgs) Handles
btnDelete_ACT.Click

    If Grid_ACT.SelectedRows.Count = 0 Then

        MessageBox.Show("Please select a row to delete.", "No Row Selected",
MessageBoxButtons.OK, MessageBoxIcon.Warning)

    Exit Sub

    End If

    Dim result As DialogResult = MessageBox.Show("Are you sure you want to
delete the selected row?", "Confirm Delete", MessageBoxButtons.YesNo,
MessageBoxIcon.Question)

    If result = DialogResult.No Then Exit Sub

    Dim row As DataGridViewRow = Grid_ACT.SelectedRows(0)

    Dim recordID = Convert.ToInt32(row.Cells("ID").Value)

    Dim deleteQuery = "DELETE FROM UBA_Drives_Data WHERE ID = ?"

    Using cmd As New OleDbCommand(deleteQuery, conn)

```

Mehr Chand Mahajan College for Women, Section-36A, Chandigarh

```

cmd.Parameters.AddWithValue("?", recordID)

conn.Open()

Dim rows = cmd.ExecuteNonQuery()

conn.Close()

If rows > 0 Then

    Grid_ACT.Rows.Remove(row)

    MessageBox.Show("Selected row deleted successfully.", "Deleted",
    MessageBoxButtons.OK, MessageBoxIcon.Information)

Else

    MessageBox.Show("No matching record found in database.", "Delete
    Failed", MessageBoxButtons.OK, MessageBoxIcon.Warning)

End If

End Using

End Sub

```

```

' MAKE YEAR COLUMN READ-ONLY

```

```

Private Sub Grid_ACT_DataBindingComplete(sender As Object, e As
DataGridViewBindingCompleteEventArgs) Handles Grid_ACT.DataBindingComplete

    If Grid_ACT.Columns.Contains("Year") Then

        Grid_ACT.Columns("Year").ReadOnly = True

    End If

End Sub

```

```

' CRYSTAL REPORT BUTTON UOPLoad REPORT

```

```

Private Sub btnUploadReport_Click(sender As Object, e As EventArgs) Handles
    btnUpload_ACT.Click

    Try

        ' Ensure a row is selected

        If Grid_ACT.SelectedRows.Count = 0 Then

            MessageBox.Show("Please select a record first.", "No Row
            Selected", MessageBoxButtons.OK, MessageBoxIcon.Warning)

        End If

    End Try

```

```

        Return

    End If

    ' Get selected record data

    Dim village =
Grid_ACT.SelectedRows(0).Cells("Village").Value.ToString()

    Dim activity =
Grid_ACT.SelectedRows(0).Cells("Activity").Value.ToString()

    Dim activityDate =
Convert.ToDateTime(Grid_ACT.SelectedRows(0).Cells("Date").Value)

    ' Query to fetch full data for selected record

    Dim query = "SELECT * FROM UBA_Drives_Data WHERE Village = ? AND
Activity = ? AND Date_of_act = ?"

    Dim dt As New DataTable()

    Using conn As New
OleDbConnection("Provider=Microsoft.ACE.OLEDB.12.0;Data
Source=D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_Dtbse.accdb")

        Using cmd As New OleDbCommand(query, conn)

            cmd.Parameters.AddWithValue("?", village)

            cmd.Parameters.AddWithValue("?", activity)

            cmd.Parameters.AddWithValue("?", activityDate)

            Using adapter As New OleDbDataAdapter(cmd)

                adapter.Fill(dt)

            End Using

        End Using

    End Using

    If dt.Rows.Count = 0 Then

```



```
        MessageBox.Show("No data found for this record.", "No Data",  
        MessageBoxButtons.OK, MessageBoxIcon.Information)
```

```
        Return
```

```
    End If
```

```
    ' Load and configure the Crystal Report
```

```
        Dim rptPath As String =  
        "D:\UBA_HUB\UBA_HUB_PROJECT_FINALLLLLLLLLLLLLLLLL\UBA_HUB_PROJECT\UBA_HUB_PROJECT\  
        UBA-Hub_Report.rpt"
```

```
    If Not IO.File.Exists(rptPath) Then
```

```
        MessageBox.Show("Report file not found at: " & rptPath, "Missing  
        Report", MessageBoxButtons.OK, MessageBoxIcon.Error)
```

```
        Return
```

```
    End If
```

```
    Dim rpt As New ReportDocument()
```

```
    rpt.Load(rptPath)
```

```
    rpt.SetDataSource(dt)
```

```
    ' === Force consistent layout ===
```

```
    rpt.PrintOptions.PaperSize = PaperSize.PaperA4
```

```
    rpt.PrintOptions.PaperOrientation = PaperOrientation.Portrait
```

```
    rpt.PrintOptions.PrinterName = "" ' Avoid using a physical printer
```

```
    ' === Optional PDF Export ===
```

```
        'Dim exportPath = "C:\UBA_Report_" &  
        DateTime.Now.ToString("yyyyMMdd_HH:mm:ss") & ".pdf"
```

```
        'Dim diskOpts As New DiskFileDestinationOptions With {
```

```
        '    .DiskFileName = exportPath
```

```
        '}
```

```

        'With rpt.ExportOptions
        '    .ExportDestinationType = ExportDestinationType.DiskFile
        '    .ExportFormatType = ExportFormatType.PortableDocFormat
        '    .DestinationOptions = diskOpts
        '    .FormatOptions = New PdfRtfWordFormatOptions()
        'End With

        'rpt.Export()

        'MessageBox.Show("Report exported to: " & exportPath, "Export
        Success", MessageBoxButtons.OK, MessageBoxIcon.Information)

        ' Show in viewer

        Dim viewer As New CrystalReportViewerForm(rpt)

        viewer.Show()

        Catch ex As Exception

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

        End Try

    End Sub

End Class

```

## Conclusion and Challenges

### Conclusion:

The UBA HUB project stands as a practical and impactful software solution developed to assist in the documentation and management of activities conducted under the **Unnat Bharat Abhiyan (UBA)** initiative. It centralizes critical data across multiple villages, allowing users to input, filter, update, and report on various development-related activities through an easy-to-use **Windows Forms Application** developed in **VB.NET** with an **MS Access** backend.

Throughout the project, the implementation of features like dropdown-based filtering, auto-refreshing DataGridViews, dynamic report generation with **Crystal Reports**, and well-structured forms like **Activities**, **Organising Team**, and **Village Coordinator Team** contributed to a highly usable and efficient system. The modular design encourages future scalability and maintenance.

Not only has the project achieved its technical goals, but it also emphasized the role of technology in community service and rural development. It has enhanced our programming, UI design, and data management skills, and most importantly, bridged the gap between academic knowledge and its real-world application.

### Challenges:

During the development of the UBA HUB project, several challenges were encountered:

1. **Crystal Reports Integration**  
Integrating Crystal Reports with VB.NET and passing dynamic parameters required careful configuration and handling of data connections. Ensuring that the report layout matched actual progress report expectations took multiple iterations.
2. **Data Filtering Logic**  
Developing the logic for dynamic filtering based on multiple criteria (Village, Activity, Year) without compromising performance or query correctness took time to implement and debug.
3. **Maintaining a Smooth Interface**  
Maintaining a consistent and responsive user interface while handling large amounts of data in `DataGridView` controls required attention to formatting, layout, and control visibility.
4. **Error Handling**  
Since users may attempt to perform unintended actions (like updating read-only fields or exporting without selecting a row), extensive error-checking and validation messages had to be implemented to prevent crashes or data loss.

## **Bibliography**

- <https://unnatbharatabhiyan.gov.in/>
- <https://mcmdavcwchd.edu.in/unnat-bharat/>
- <https://www.google.com>
- <https://www.youtube.com>
- Class notes
- Book: Application Development Using VB .Net by Indu Arora (Kalyani).