ACADEMIC APPLICATION BRIEF

VB.NET Contact Manager

A desktop record-management application built in Visual Studio as an individual academic project. The project centers on structured entry, validation, record actions, and CSV export.

PORTFOLIO SNAPSHOT

- Windows Forms interface and read-only DataGridView
- Required-field validation and unique-ID checking
- Add, update, select, and delete record workflows
- CSV export with fictional demonstration data

APPLICATION DESIGN

A structured record-management workflow

The application was designed to help a user enter and manage structured contact records. It uses a Windows Forms interface, a controlled data grid, clear field labels, and direct actions for adding, updating, deleting, and exporting information.

CONTACT MANAGER · INTERFACE RECONSTRUCTION

FULL NAME

RECORD ID

PHONE

ADDRESS

CATEGORY

RECORD ID	NAME	PHONE	CATEGORY
CT-101	Sample A	555-0101	General
CT-102	Sample B	555-0102	General
CT-104	**Jordan Lee**	**555-0104**	**General**

3 VISIBLE RECORDS

EXPORT TO CSV

Presentation note: this interface reconstruction reflects the source project’s workflow and uses fictional sample information.

CONTROLS AND VALIDATION

Protect record quality before saving

Validation workflow

01 **Capture record fields**
The interface collects name, identification, phone, address, and category information.

02 **Check required fields**
The application shows field-level feedback when a required value is missing.

03 **Check unique ID**
Before a new record is added, the application checks the existing grid for a matching identification value.

04 **Perform record action**
The user can add a new entry, select an existing row for update, or confirm a delete action.

05 **Export for reuse**
The application writes the visible records to a CSV file with field headings.

Selected source-based code sample

```
Private Function IsUniqueRecordId(recordId As String) As Boolean
    For Each row As DataGridViewRow In DgvDatos.Rows
        If row.Cells("ColumnIdentificacion").Value IsNot Nothing AndAlso
           row.Cells("ColumnIdentificacion").Value.ToString() = recordId Then
            ErrorProviderValidar.SetError(TxtIdentificacion, "Use a different record ID.")
            Return False
        End If
    Next
    Return True
End Function
```

PROJECT SCOPE

A practical desktop application exercise

PRIMARY CAPABILITY

Structured data entry, validation, record actions, and simple file output in a desktop interface.

TECHNOLOGY CONTEXT

VB.NET and Microsoft Visual Studio using Windows Forms controls, a DataGridView, error feedback, and file writing.

APPROPRIATE SCOPE

An academic desktop application that manages records in the interface and exports CSV data. It is not presented as a production system or as a database-backed application.

PROFESSIONAL RELEVANCE

This project demonstrates practical habits that matter in data and records work: required-field checks, unique identifiers, controlled record changes, clear user feedback, and portable CSV output.