How to extract CSV from PDF and fill database in SQL server in VB.NET and ByteScout PDF Extractor SDK

How to code in VB.NET to extract CSV from PDF and fill database in SQL server with this step-by-step tutorial

Extract CSV from PDF and fill database in SQL server is easy to implement in VB.NET if you use these source codes below. ByteScout PDF Extractor SDK is the SDK that helps developers to extract data from unstructured documents, pdf, images, scanned and electronic forms. Includes AI functions like automatic table detection, automatic table extraction and restructuring, text recognition and text restoration from pdf and scanned documents. Includes PDF to CSV, PDF to XML, PDF to JSON, PDF to searchable PDF functions as well as methods for low level data extraction. It can be used to extract CSV from PDF and fill database in SQL server using VB.NET.

You will save a lot of time on writing and testing code as you may just take the VB.NET code from ByteScout PDF Extractor SDK for extract CSV from PDF and fill database in SQL server below and use it in your application. In your VB.NET project or application you may simply copy & paste the code and then run your app! Use of ByteScout PDF Extractor SDK in VB.NET is also explained in the documentation included along with the product.

Free trial version of ByteScout PDF Extractor SDK is available on our website. Documentation and source code samples are included.

VB.NET - Program.vb

```
Imports System.Data.SqlClient
Imports Bytescout.PDFExtractor

Module Program

    Sub Main()

        Try

            ' Step-1: Get Datatable
            Dim oDataTable = GetDataTableFromDocument("sample.pdf")

            ' PLEASE NOTE Please Replace With your connection String, You need to
have "PersonData" table into your database.
            ' You can find that table from Scripts.sql file
            Dim connectionString As String = "Data Source=DESKTOP-
92VMCQG\SQLEXPRESS;Initial Catalog=SampleDatabase;Persist Security Info=True;User
ID=sa;Password=Hiren@009"

            ' Step-2: Insert into database
```

```
            InsertIntoSqlServerDatabase(oDataTable, connectionString)

            ' Step-3: Fetch from database and display results
            DisplayDatabaseResults(connectionString)

        Catch ex As Exception
            Console.WriteLine(ex.Message)
        End Try

        Console.WriteLine("Press enter key to exit...")
        Console.ReadLine()
    End Sub

    '''

    ''' Inserts into Sql Server database
    '''

    Private Sub InsertIntoSqlServerDatabase(ByVal oDataTable As DataTable, ByVal
connectionString As String)

        Using con As SqlConnection = New SqlConnection(connectionString)

            ' Open connection
            con.Open()

            ' Sql query to insert data
            Dim cmdInsert As String = "Insert into PersonData (id, first_name,
last_name, email, gender, ip_address) values (@id, @first_name, @last_name, @email,
@gender, @ip_address)"

            For Each itmRow As DataRow In oDataTable.Rows

                ' Prepare sql command
                Dim cmd As SqlCommand = New SqlCommand(cmdInsert, con)
                cmd.CommandType = CommandType.Text

                cmd.Parameters.Add(New SqlParameter("@id",
Convert.ToString(itmRow("id"))))
                cmd.Parameters.Add(New SqlParameter("@first_name",
Convert.ToString(itmRow("first_name"))))
                cmd.Parameters.Add(New SqlParameter("@last_name",
Convert.ToString(itmRow("last_name"))))
                cmd.Parameters.Add(New SqlParameter("@email",
Convert.ToString(itmRow("email"))))
                cmd.Parameters.Add(New SqlParameter("@gender",
Convert.ToString(itmRow("gender"))))
                cmd.Parameters.Add(New SqlParameter("@ip_address",
Convert.ToString(itmRow("ip_address"))))

                ' Execute sql command
                cmd.ExecuteNonQuery()
            Next

            ' Close connection
            con.Close()

        End Using

    End Sub
```

```
    ' Displays inserted database results
    Private Sub DisplayDatabaseResults(ByVal connectionString As String)

        ' Person data holder
        Dim personDataTable As DataTable = New DataTable()

        Using con As SqlConnection = New SqlConnection(connectionString)

            ' Sql query to fetch data
            Dim cmdInsert As String = "SELECT id, first_name, last_name, email,
gender, ip_address FROM PersonData;"

            ' Prepare sql command
            Dim cmd As SqlCommand = New SqlCommand(cmdInsert, con)
            cmd.CommandType = CommandType.Text

            ' Prepare DataAdapter
            Dim dataAdapter As SqlDataAdapter = New SqlDataAdapter(cmd)

            ' Fill person dataTable
            dataAdapter.Fill(personDataTable)

        End Using

        ' Display all person data if any
        If personDataTable IsNot Nothing AndAlso personDataTable.Rows.Count > 0 Then

            ' Print all columns
            For Each column As DataColumn In personDataTable.Columns
                Console.Write("{0} | ", column.ColumnName)
            Next
            Console.WriteLine()

            ' Print all data
            For Each dataRow As DataRow In personDataTable.Rows

                For Each column As DataColumn In personDataTable.Columns
                    Console.Write("{0} | ", dataRow(column.ColumnName))
                Next

                Console.WriteLine()
            Next
        Else
            Console.WriteLine("No data retrieved..")
        End If
    End Sub

    '''

    ''' Get DataTable from Document
    '''

    Private Function GetDataTableFromDocument(ByVal fileName As String) As DataTable
        Dim oDataTable As DataTable = Nothing

        ' Initialise table detector
        Using tableDetector As TableDetector = New TableDetector("demo", "demo")

            Using CSVExtractor As CSVExtractor = New CSVExtractor("demo", "demo")
```

```
                ' Set table detection mode to "bordered tables" - best for tables
with closed solid borders.
                tableDetector.ColumnDetectionMode =
ColumnDetectionMode.BorderedTables

                ' We should define what kind of tables we should detect.
                ' So we set min required number of columns to 2 ...
                tableDetector.DetectionMinNumberOfColumns = 2
                ' ... and we set min required number of rows to 2
                tableDetector.DetectionMinNumberOfRows = 2

                ' Load PDF document
                tableDetector.LoadDocumentFromFile(fileName)
                CSVExtractor.LoadDocumentFromFile(fileName)

                ' Get page count
                Dim pageCount As Integer = tableDetector.GetPageCount()

                If tableDetector.FindTable(0) Then
                    ' Set extraction area for CSV extractor to rectangle received
from the table detector
                    CSVExtractor.SetExtractionArea(tableDetector.FoundTableLocation)

                    ' Generate CSV data
                    Dim allCsvData = CSVExtractor.GetCSV()

                    ' Generate Datatable
                    oDataTable = GetDataTableFromCSV(allCsvData)
                End If
            End Using
        End Using

        Return oDataTable
    End Function

    '''

    ''' Get Datatable from CSV
    '''

    Private Function GetDataTableFromCSV(ByVal allCsvData As String) As DataTable

        Dim oRetDataTable = New DataTable()
        oRetDataTable.Columns.Add("id")
        oRetDataTable.Columns.Add("first_name")
        oRetDataTable.Columns.Add("last_name")
        oRetDataTable.Columns.Add("email")
        oRetDataTable.Columns.Add("gender")
        oRetDataTable.Columns.Add("ip_address")

        Dim rows = allCsvData.Split(vbLf)

        For iRow As Integer = 1 To rows.Length - 1

            ' Get all column data
            Dim columns = rows(iRow).Split(","c)

            If columns.Length >= 5 Then
                ' Prepare new row
```

```
                Dim oRow = oRetDataTable.NewRow()
                oRow("id") = columns(0)
                oRow("first_name") = columns(1)
                oRow("last_name") = columns(2)
                oRow("email") = columns(3)
                oRow("gender") = columns(4)
                oRow("ip_address") = columns(5)

                ' Add row back to datatable
                oRetDataTable.Rows.Add(oRow)
            End If
        Next

        ' Return DataTable
        Return oRetDataTable
    End Function

End Module
```

FOR MORE INFORMATION AND FREE TRIAL:

[Download Free Trial SDK (on-premise version)]

[Read more about ByteScout PDF Extractor SDK]

[Explore documentation]

[Visit www.ByteScout.com]

or

[Get Your Free API Key for www.PDF.co Web API]