UNIVERSIDAD
MARIANO GALVEZ

INGENIERIA EN SISTEMAS

PROYECTO 1 PROGRAMACION

Estudiante:
RONALD ANTONIO AROCHE SANTOS

Docente
ING. RULDIN EFRAIN AYALA RAMOS

Carnet
0905-24-17495

1. Introducción

El presente documento detalla el proceso de elaboración de la aplicación de escritorio "PROYECTO1", diseñada para facilitar la investigación y generación de documentos profesionales (informes en Word y presentaciones en PowerPoint) utilizando inteligencia artificial. El objetivo principal es automatizar tareas repetitivas y proporcionar al usuario una herramienta eficiente para procesar información y crear contenido estructurado.

2. Tecnologías Utilizadas

El proyecto se desarrolló utilizando las siguientes tecnologías y herramientas:

- Lenguaje de Programación: C#
- Plataforma de Desarrollo: .NET Framework (o .NET Core, dependiendo de tu configuración exacta).
- Interfaz de Usuario: Windows Forms.
- Gestión de Base de Datos: SQL Server.
- Generación de Documentos: Open XML SDK (biblioteca de Microsoft para trabajar con formatos Open XML).
- Inteligencia Artificial (API): Cohere API.
- Gestión de Dependencias JSON: Newtonsoft.Json (Json.NET).

3. Arquitectura del Proyecto

- El proyecto sigue una arquitectura modular, dividiéndose en las siguientes funcionalidades principales:
- Interfaz de Usuario (UI): Desarrollada con Windows Forms, permite al usuario ingresar un "prompt" (tema de investigación), visualizar y editar el resultado generado por la IA, y activar las funciones de guardado y generación de documentos.
- Comunicación con la API de IA: Se implementa un método asíncrono para enviar solicitudes a la API de Cohere y recibir las respuestas. Esto asegura que la aplicación se mantenga responsiva mientras se espera la respuesta de la IA.
- Persistencia de Datos: Los prompts y los resultados de la investigación se almacenan en una base de datos SQL Server, permitiendo un registro y acceso a investigaciones previas.
- Generación de Documentos: Se utilizan métodos dedicados para crear archivos Word (.docx) y PowerPoint (.pptx) a partir del texto generado por la IA. El Open XML SDK permite manipular la estructura interna de estos documentos.

- Gestión de Archivos: Se incluye una función para crear una carpeta dedicada en el escritorio del usuario, donde se guardan los documentos generados.

4. Proceso de Elaboración

- Diseño de la Interfaz de Usuario:
- Se creó un formulario principal (Form1) con controles básicos: un TextBox para el prompt (txtPrompt), un TextBox multi-línea para mostrar y editar los resultados de la IA (txtResultados), botones para "Investigar" y "Generar Informe", una etiqueta de estado (lblEstado) y una barra de progreso (progressBar).
- Se configuraron los eventos Click para los botones y el evento Load para el formulario.

Integración con la API de Cohere:

- Se añadió la biblioteca HttpClient para realizar peticiones HTTP.
- Se construyó el método ConsultarAIAsync para:
- Configurar los encabezados de autenticación (Authorization con el Bearer token y Cohere-Version).
- Crear el cuerpo de la petición JSON, especificando el modelo (command), el prompt del usuario, max_tokens y temperature.
- Serializar el objeto a JSON usando Newtonsoft.Json.JsonConvert.SerializeObject.
- Enviar la petición POST a https://api.cohere.ai/v1/generate.
- Deserializar la respuesta JSON usando Newtonsoft.Json.JsonConvert.DeserializeObject para extraer el texto generado.
- Implementar manejo básico de errores para respuestas fallidas de la API.
- Conexión y Guardado en SQL Server:

Se definió una cadena de conexión para SQL Server. (¡Importante! En un entorno de producción, esta cadena de conexión debe ser almacenada de forma segura y no directamente en el código).

Se implementó el método GuardarInvestigacion utilizando SqlConnection y SqlCommand para insertar los datos (Prompt y Resultado) en la tabla Investigaciones. Se usaron parámetros para prevenir inyección SQL.

Generación de Documentos con Open XML SDK:Documentos Word (GenerarWord):

- Se utilizó WordprocessingDocument.Create para crear un nuevo archivo .docx.
- El contenido se dividió por saltos de línea (Environment.NewLine) para asegurar que cada línea del texto generado por la IA se representara como un párrafo separado en el documento Word.
- Presentaciones PowerPoint (GenerarPowerPoint):
- Este fue el componente más complejo debido a la estructura de PowerPoint. Se utilizó PresentationDocument.Create para crear un nuevo archivo .pptx.
- Se crearon las partes esenciales de una presentación (PresentationPart, SlideMasterPart, SlideLayoutPart, SlidePart) y se les dio una estructura mínima válida para poder ser abiertas por PowerPoint.
- Se añadió una diapositiva con un título ("Informe de Investigación") y un cuerpo de texto que contenía el resultado de la IA, también manejando saltos de línea para la correcta visualización.

Gestión de Archivos y Flujo de la Aplicación:

- El método CrearCarpetaYGuardar se encargó de crear una carpeta InvestigacionAI en el escritorio del usuario y llamar a los métodos de generación de documentos.
- Los eventos Click de los botones orquestaron la secuencia de operaciones: btnInvestigar_Click llama a la IA, y btnGenerarInforme_Click guarda en la DB y genera los archivos. Se incluyeron actualizaciones de UI (estado y barra de progreso) para retroalimentación al usuario.

5. APIs y Librerías Externas Utilizadas

Cohere API:

- Propósito: Generación de texto basada en inteligencia artificial (Large Language Model). Permite obtener respuestas coherentes y relevantes a partir de un "prompt" dado.
- Endpoint Principal: https://api.cohere.ai/v1/generate
- Autenticación: Requiere un Bearer Token enviado en el encabezado Authorization.
- Modelo Utilizado: command (un modelo de propósito general de Cohere).

Open XML SDK (DocumentFormat.OpenXml):

- Propósito: Biblioteca de Microsoft que permite interactuar programáticamente con documentos en formato Open XML (Word, Excel, PowerPoint).
- Uso en el Proyecto: Creación y modificación de archivos .docx y .pptx, incluyendo la adición de texto y la estructuración básica de los documentos.

Newtonsoft.Json (Json.NET):

- Propósito: Popular librería para la serialización y deserialización de objetos .NET a y desde JSON.
- Uso en el Proyecto: Convertir el objeto de solicitud a la API de Cohere a formato JSON y parsear la respuesta JSON de la API.

ESQUEMA DE BASE DE DATOS (Script SQL)

```
-- Crea la base de datos si no existe
IF NOT EXISTS (SELECT name FROM sys.databases WHERE name = N'PROYECTO1')
BEGIN
    CREATE DATABASE PROYECTO1;
END
GO

USE PROYECTO1;
GO

-- Crea la tabla Investigaciones
IF NOT EXISTS (SELECT * FROM sysobjects WHERE name='Investigaciones' and xtype='U')
BEGIN
    CREATE TABLE Investigaciones (
        Id INT IDENTITY(1,1) PRIMARY KEY,
        Prompt NVARCHAR(MAX) NOT NULL,
        Resultado NVARCHAR(MAX) NOT NULL,
        FechaCreacion DATETIME DEFAULT GETDATE()
    );
END
GO
```