**Azure Pipeline Integration with Database Key Management**

**Description**

* Automates the retrieval of keys from a SQL Server database.
* Uses retrieved keys to authenticate with Azure APIs.
* Triggers Azure pipelines based on found keys.
* Supports both single and multiple keys using a loop.

**Prerequisites**

* **Visual Studio**: Required for development and running the project.
* **SQL Server**:
  + Database containing keys.
  + Permissions to read from the database.
* **Azure DevOps**:
  + Azure DevOps organization.
  + Configured Azure pipelines.
  + API access and credentials for pipeline triggers.
* **Git**: Required for cloning, committing, and pushing the project.

**Installation**

1. **Clone the repository**:

bash

Copy code

git clone https://github.com/yourusername/yourproject.git

cd yourproject

1. **Open the project in Visual Studio**.
2. **Dependencies**:
   * Install NuGet packages:
     + System.Data.SqlClient for database access.
     + Azure DevOps SDK or API client libraries.
3. **Setup Configuration Files**:
   * Create a config.json configuration file.

**Example config.json**:

json

Copy code

{

"SqlConnectionString": "Your SQL Server Connection String",

"AzureDevOpsOrganization": "Your Azure DevOps Organization Name",

"AzureDevOpsPipelineId": "Your Azure DevOps Pipeline ID",

"AzureDevOpsPersonalAccessToken": "Your Azure DevOps PAT"

}

**How to Use**

1. **Fetch keys from the database**:

csharp

Copy code

using (SqlConnection conn = new SqlConnection(Configuration["SqlConnectionString"]))

{

conn.Open();

SqlCommand cmd = new SqlCommand("SELECT KeyColumn FROM YourTable WHERE IsActive = 1", conn);

SqlDataReader reader = cmd.ExecuteReader();

while (reader.Read())

{

string key = reader["KeyColumn"].ToString();

CallAzureApiAndTriggerPipeline(key); // Method to execute Azure API

}

}

1. **Call Azure API**:

csharp

Copy code

private void CallAzureApiAndTriggerPipeline(string key)

{

// Azure API call logic to authenticate and trigger pipeline

string token = Configuration["AzureDevOpsPersonalAccessToken"];

string pipelineId = Configuration["AzureDevOpsPipelineId"];

HttpClient client = new HttpClient();

client.DefaultRequestHeaders.Authorization = new AuthenticationHeaderValue("Bearer", token);

var response = client.PostAsync($"https://dev.azure.com/{Configuration["AzureDevOpsOrganization"]}/\_apis/pipelines/{pipelineId}/runs?api-version=6.0", new StringContent("{}")).Result;

if (response.IsSuccessStatusCode)

{

Console.WriteLine("Pipeline triggered successfully.");

}

else

{

Console.WriteLine("Failed to trigger pipeline.");

}

}

1. **Loop through multiple keys if necessary**:

csharp

Copy code

while (reader.Read())

{

string key = reader["KeyColumn"].ToString();

CallAzureApiAndTriggerPipeline(key);

}

**Usage Notes**

* Ensure the database connection string and Azure DevOps token are stored securely.
* Verify Azure DevOps organization and pipeline IDs are correctly configured.

**Contributing**

* Fork the repository.
* Make your changes.
* Submit a pull request.