Temă - Laborator05

Implementați o clasă pentru salvarea logurilor într-o bază de date SQL.

SqlServerLogger

```
namespace Logger {
  public class SqlServerLogger: ILogger {
    private SqlConnection con;
    private DataSet dataSet;
    private SqlDataAdapter dataAdapter;
    private static string connectionString;
    private static string query;
    private static string dbName;

    private static void GetParameters() {
      var section = ConfigurationManager.GetSection("sql-server") as
NameValueCollection;
      var source = section["Source"];
      var catalog = section["Catalog"];
      var persist = section["Persist"];
      var id = section["ID"];
      var password = section["Password"];
      dbName = section["DBName"];
      query = $ "select * from {dbName}";
      connectionString = $@"Data Source='{source}';Initial Catalog=
{catalog};Persist Security Info={persist};User ID={id};Password={password}";
    }

    public SqlServerLogger() {
      GetParameters();
      con = new SqlConnection(connectionString);
      dataSet = new DataSet();
      try {
        dataAdapter = new SqlDataAdapter(query, con);
        dataAdapter.Fill(dataSet, dbName);
      }
      catch(Exception) {
        CreateDB();
        dataAdapter = new SqlDataAdapter(query, con);
        dataAdapter.Fill(dataSet, dbName);
      }
    }

    private void AddLogToDB(LogModel log) {
      try {
        DataRow newRow = dataSet.Tables[dbName].NewRow();
        newRow["Date"] = log.Date;
        newRow["LogType"] = log.Type;
        newRow["Message"] = log.Message;
        newRow["ApplicationName"] = log.ApplicationName;
        dataSet.Tables[dbName].Rows.Add(newRow);
      }
      catch(Exception) {
        Console.WriteLine($ "Undefined Database: {dbName}");
      }
    }

    private void SaveChangesToDB() {
      SqlCommandBuilder commandBuilder = new SqlCommandBuilder(dataAdapter);
      DataSet dsChange = dataSet.GetChanges();
      if (dsChange != null) {
        dataAdapter.Update(dsChange, dbName);
        dataSet.AcceptChanges();
      }
    }

    private void CreateDB() {
      try {
        SqlCommand cmd = new SqlCommand(@"create table tLogger(Id int
Identity(1, 1), Date datetime2 not null, LogType varchar(10) not null, Message
varchar(256), ApplicationName varchar(50) not null); ", con);
        con.Open();
        cmd.ExecuteNonQuery();
        Console.WriteLine("Table Created Successfully...");
        con.Close();
      }
      catch(Exception e) {
        Console.WriteLine("Exception occured while creating table:" + e.Message
+ "\t" + e.GetType());
      }
    }

    public void Log(List < LogModel > logs) {
      if (logs != null) {
        foreach(var log in logs) {
          log.ApplicationName =
ConfigurationManager.AppSettings["ApplicationName"];
          log.Date = DateTime.Now;
          AddLogToDB(log);
        }
      }
      SaveChangesToDB();
    }
  }
}
```

App.config

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
    <configSections>
      <section name="sql-server"
type="System.Configuration.NameValueSectionHandler" />
    </configSections>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
    </startup>
    <appSettings>
      <add key="LogFile" value="Log.txt" />
      <add key="ApplicationName" value="LogApplication" />
    </appSettings>
    <sql-server>
      <add key="Source" value="localhost, 1433" />
      <add key="Catalog" value="master" />
      <add key="Persist" value="True" />
      <add key="ID" value="SA" />
      <add key="Password" value="DBPass.123" />
      <add key="DBName" value="tLogger" />
    </sql-server>
</configuration>
```