using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using static ProxyPattern.Proxy;

namespace ProxyPattern

{

class Proxy

{

public interface ICache

{

string GetValue(string key);

void SetValue(string key, string value);

}

public class RemoteDatabase : ICache

{

private Dictionary<string, string> Configuration;

public RemoteDatabase()

{

Configuration = new Dictionary<string, string>();

SetConfiguration();

}

public string GetValue(string key)

{

if (Configuration.ContainsKey(key))

{

Console.WriteLine($"RemoteDatabase<GetValue>: [FETCH] Key: {key}, Value: {Configuration[key]}");

return Configuration[key];

}

else

return string.Empty;

}

public void SetValue(string key, string value)

{

if (Configuration.ContainsKey(key))

{

Console.WriteLine($"RemoteDatabase<SetValue> [UPDATED]: Key: {key}, Value: {Configuration[key]}");

Configuration[key] = value;

}

else

{

Configuration.Add(key, value);

Console.WriteLine($"RemoteDatabase<SetValue> [ADDED]: Key: {key}, Value: {Configuration[key]}");

}

}

private void SetConfiguration()

{

Configuration.Add("SessionId", Guid.NewGuid().ToString());

Configuration.Add("FirstName", "John");

}

}

public class CacheProxy : ICache

{

private Dictionary<string, string> CachedConfiguration;

private RemoteDatabase databaseCache;

public CacheProxy()

{

CachedConfiguration = new Dictionary<string, string>();

databaseCache = new RemoteDatabase();

}

public string GetValue(string key)

{

if (CachedConfiguration.ContainsKey(key))

{

Console.WriteLine($"CacheProxy<GetValue>: [PROXY-FETCH] Key: {key}, Value: {CachedConfiguration[key]}");

return CachedConfiguration[key];

}

else

{

var value = databaseCache.GetValue(key);

if (!string.IsNullOrEmpty(value))

{

CachedConfiguration.Add(key, value);

Console.WriteLine($"CacheProxy<GetValue>: [PROXY-UPDATED] Key: {key}, Value: {CachedConfiguration[key]}");

return value;

}

else

Console.WriteLine($"Invalid key: {key}");

}

return string.Empty;

}

public void SetValue(string key, string value)

{

databaseCache.SetValue(key, value);

CachedConfiguration[key] = value;

Console.WriteLine($"CacheProxy<SetValue>: [PROXY-UPDATED] Key: {key}, Value: {CachedConfiguration[key]}");

}

}

}

class Client

{

static void Main(string[] args)

{

ICache cache = new CacheProxy();

Console.WriteLine($"Client: {cache.GetValue("SessionId")}");

Console.WriteLine();

Console.WriteLine($"Client: {cache.GetValue("SessionId")}");

Console.WriteLine();

cache.SetValue("UserName", "John");

Console.WriteLine($"Client: {cache.GetValue("UserName")}");

Console.WriteLine();

cache.SetValue("UserName", "Doe");

Console.WriteLine($"Client: {cache.GetValue("UserName")}");

Console.WriteLine();

}

}

}