Table of Contents

[I. Requirement. 2](#_Toc131206149)

[II. Prepare the environment. 2](#_Toc131206150)

[III. Implementation steps. 2](#_Toc131206151)

[1. Build the database for the program. 2](#_Toc131206152)

[2. Create a Windows Form App C# project. 3](#_Toc131206153)

[3. Design the "ViewProduct" form. 4](#_Toc131206154)

[4. Connect to the database and display data from the database. 6](#_Toc131206155)

[4.1. Configure connection to the database. 6](#_Toc131206156)

[4.2. Display data from the database. 7](#_Toc131206157)

[IV. Program result. 7](#_Toc131206158)

# I. Requirement.

In this lab, you need to complete the following tasks:

* Create a new Window Application Form project
* Connect to the database
* Display data from the database

# II. Prepare the environment.

* Visual Studio.
* Microsoft SQL Server Management Studio.

# III. Implementation steps.

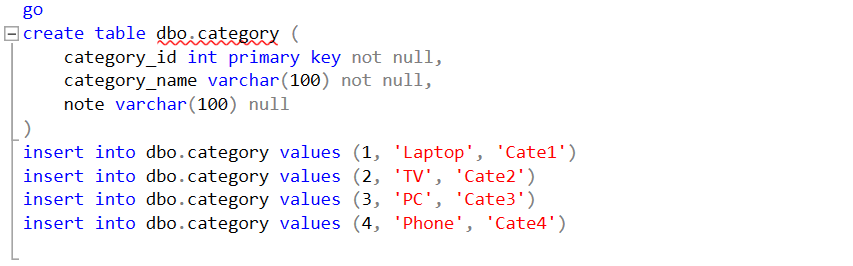
## 1. Build the database for the program.

At this step, we need to create a database called “product\_management”, then create the tables such as dbo.category, dbo.product and add the corresponding data to the table.

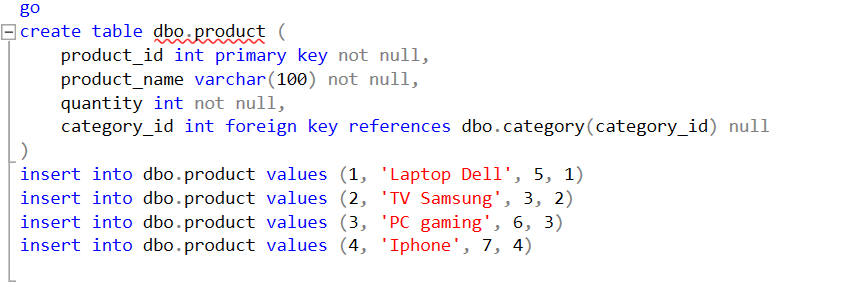
The queries are as follows:







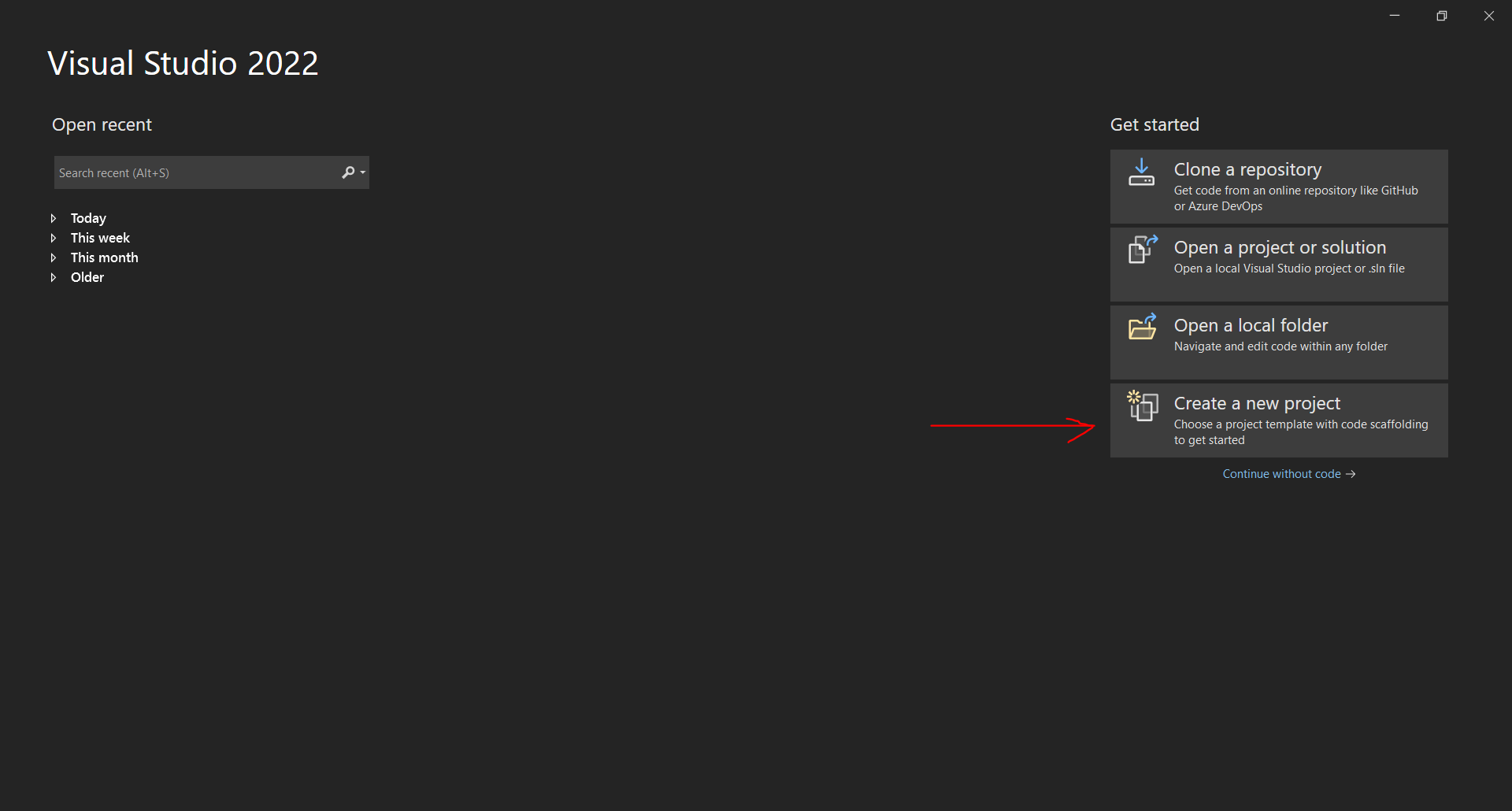
The data added to the dbo.category table will be displayed as the category's names for the user to choose when running the program.



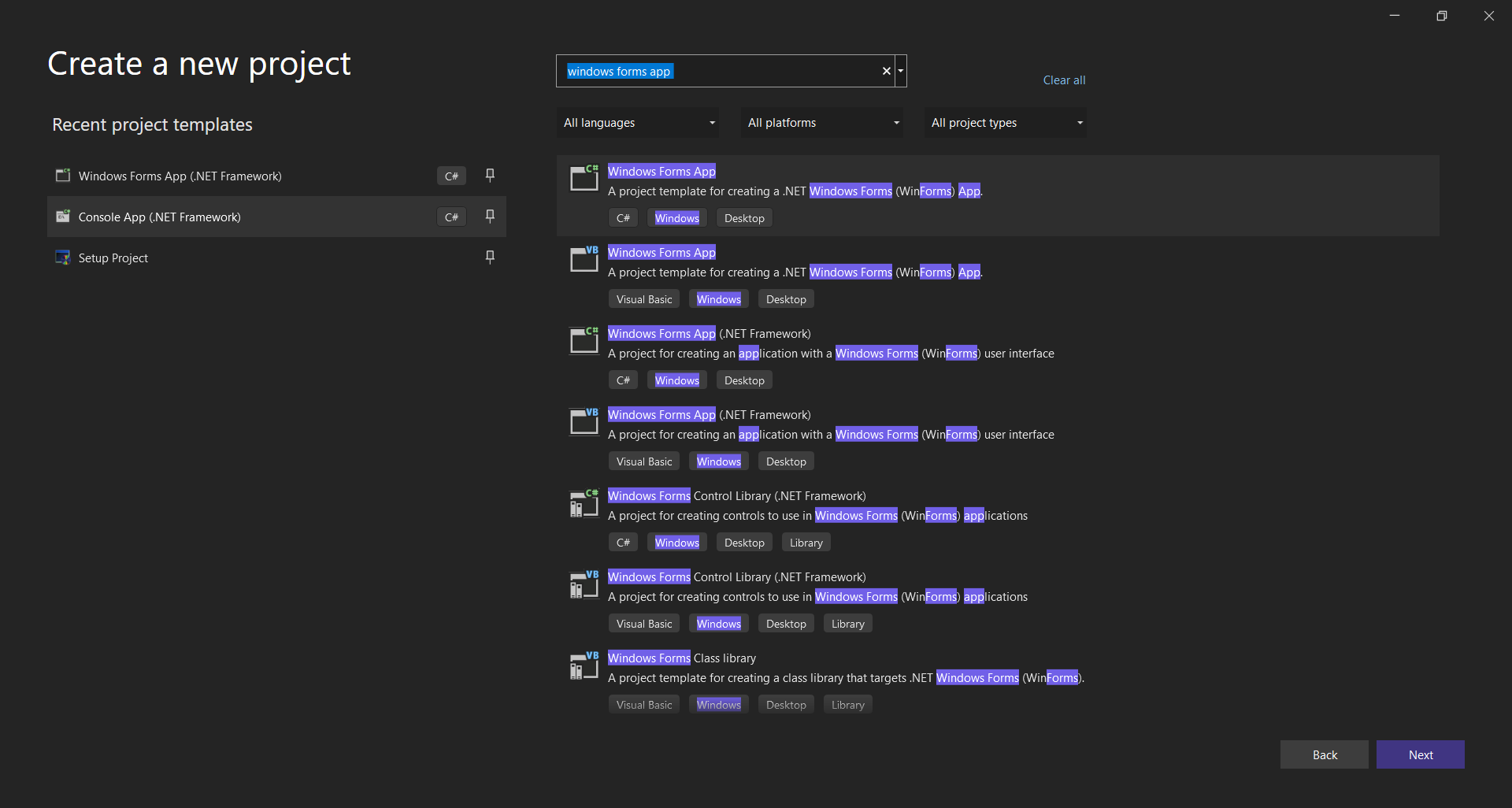
The data added to the dbo.product table will be displayed after the running without any new additions. This is also one of the proofs of a successful connection to the database.

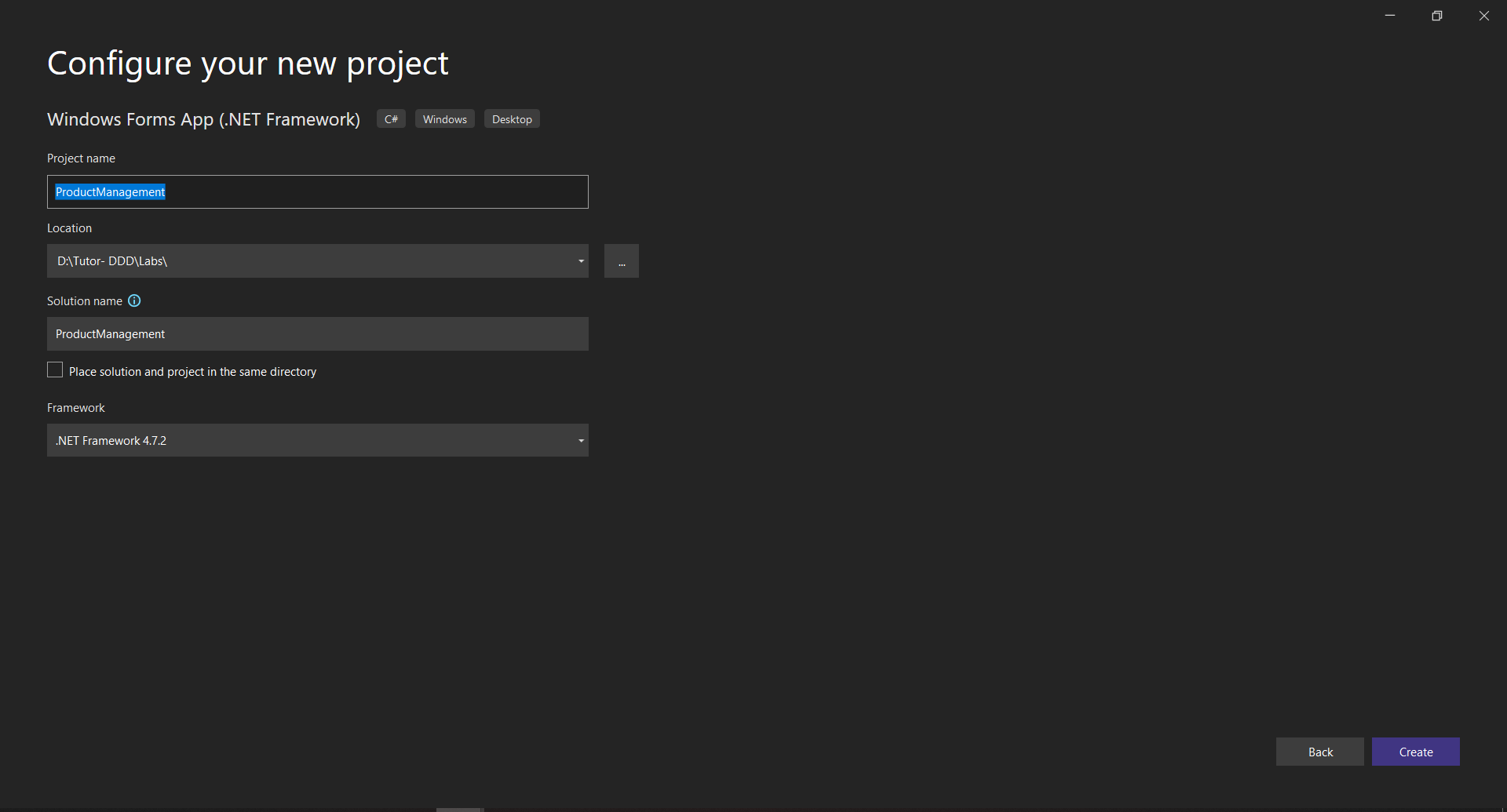
## 2. Create a Windows Form App C# project.

In this step, we will create a project named “ProductManagement”.



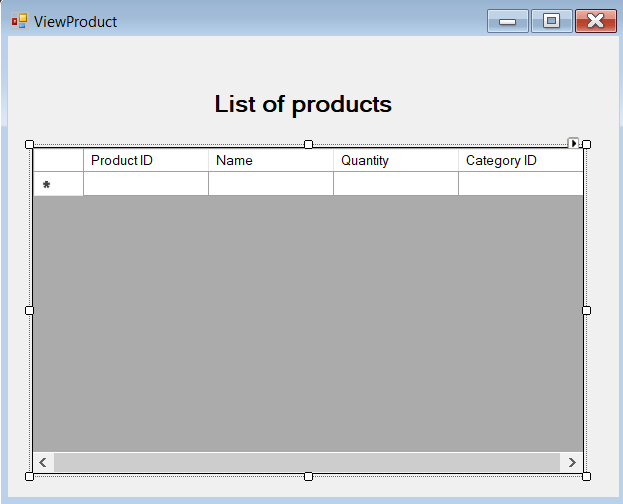
After launching Visual Studio, select "Create a new project". Next, choose "Windows Forms App" with C# language template to create a new one.





## 3. Design the "ViewProduct" form.

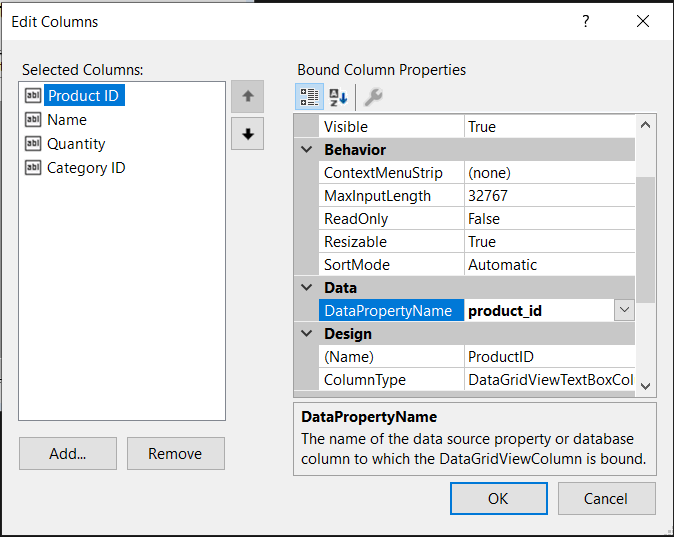
Design the form to view a list of products layout as shown below:



This form includes Label and DataGridView.

In DataGridView, create a table to store product information by adding columns corresponding to the product's attributes.

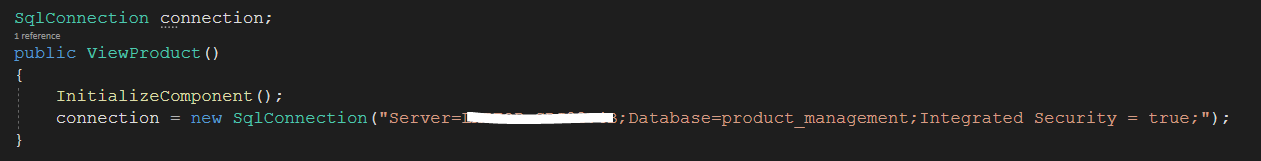
Note: The DataPropertyName of each column must match the corresponding property name declared in the dbo.product table in the database.



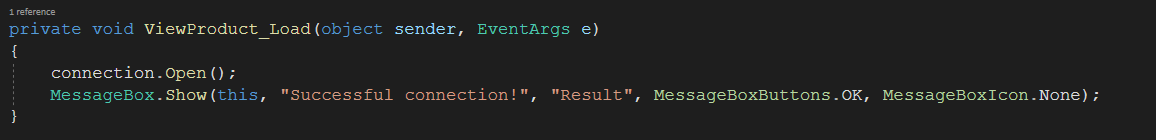
## 4. Connect to the database and display data from the database.

### 4.1. Configure connection to the database.

Declare an SQLConnection variable at the beginning of the program.



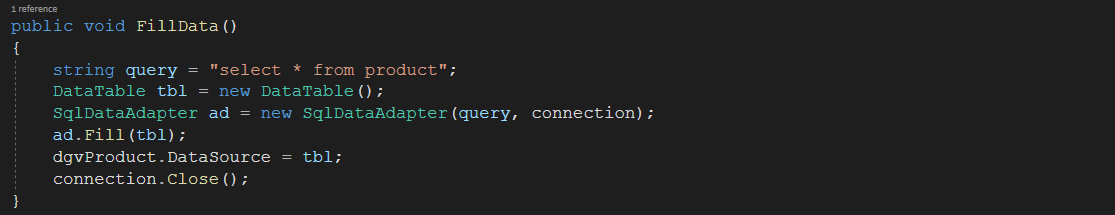
The server is the server name that connects to the database engine in SQL Server, and the database is the database built for the program.



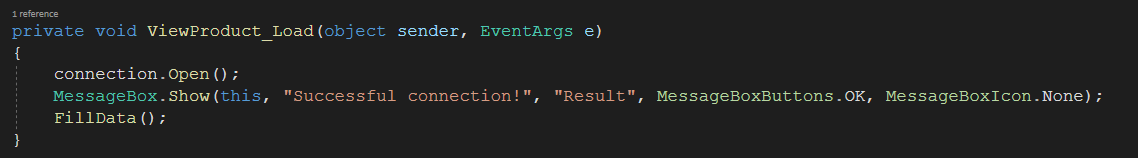
Implement the form load event and check the connection to the database.

### 4.2. Display data from the database.

Build the FillData() function to get all product data out to the DataGridView.

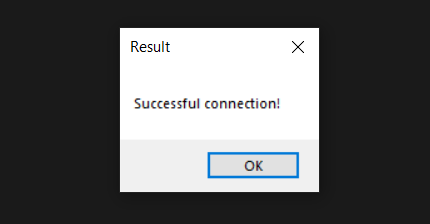


In the form load event, call the FillData() function.

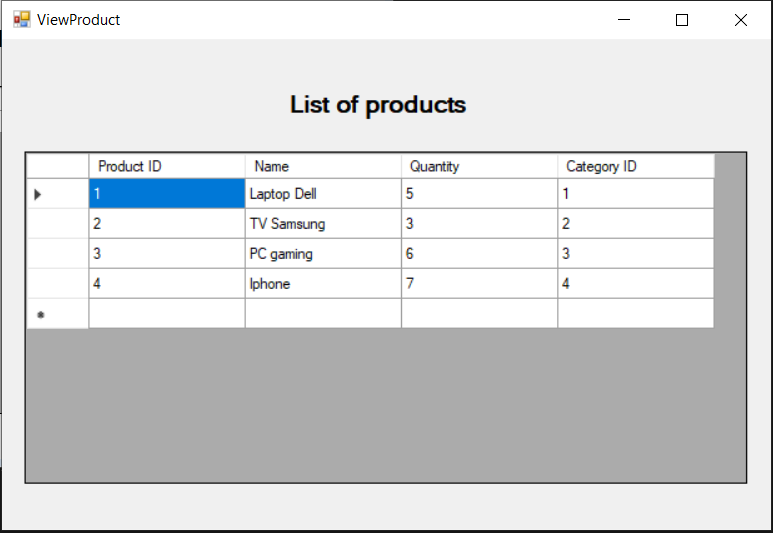


# IV. Program result.

When running the program, a message dialog will be displayed.



When the user selects the "OK" button, the "ViewProduct" form will be displayed.



Here, the user can see the list of products available in the database.

