Guideline on how to establish a connection between back-end and SQL Server

-Nguyen Hoang Tuan Anh-

Step 1:

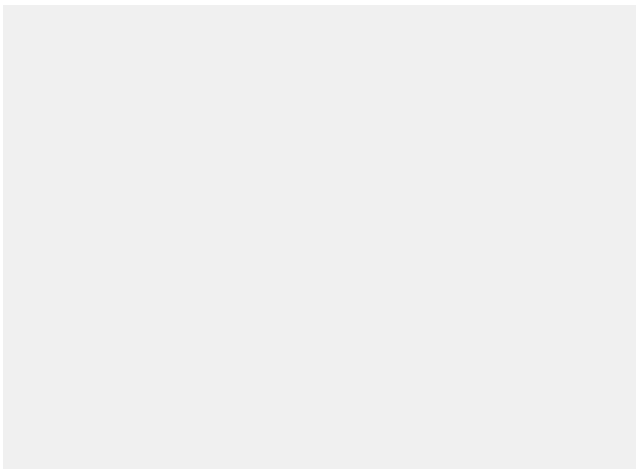
_Make sure you have already installed these things from IT departments: (guideline on how to install here : https:/example.com/install)

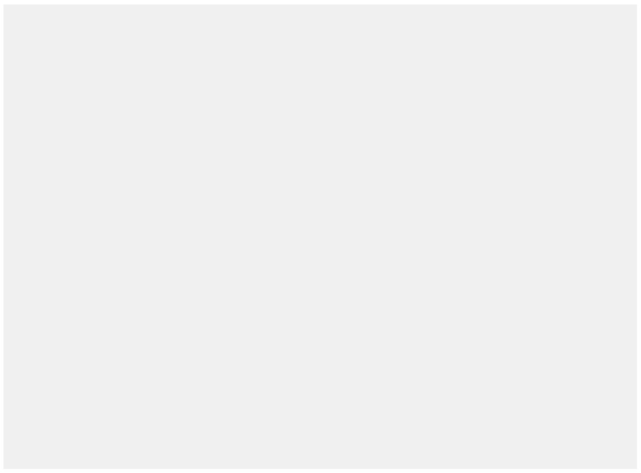
+ SQL Server 2022

+ SSMS 19.x

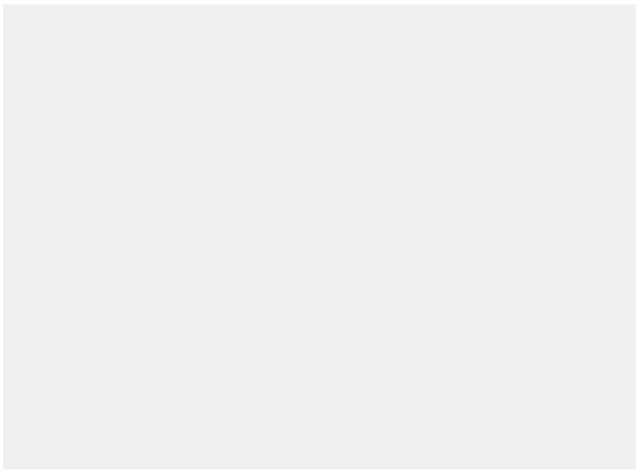
Step 2 :

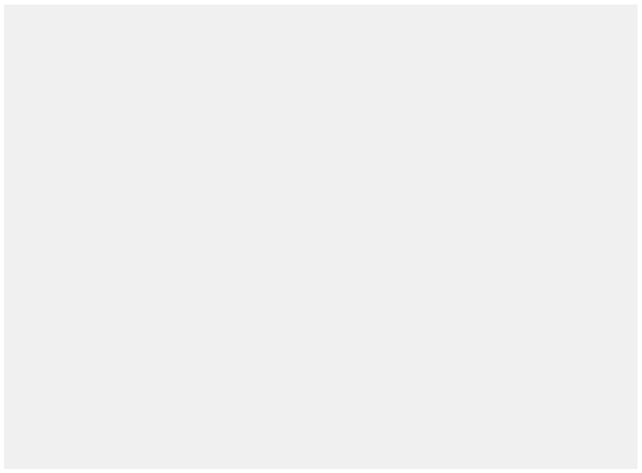
_ Enable your administrator rights (follow these steps):





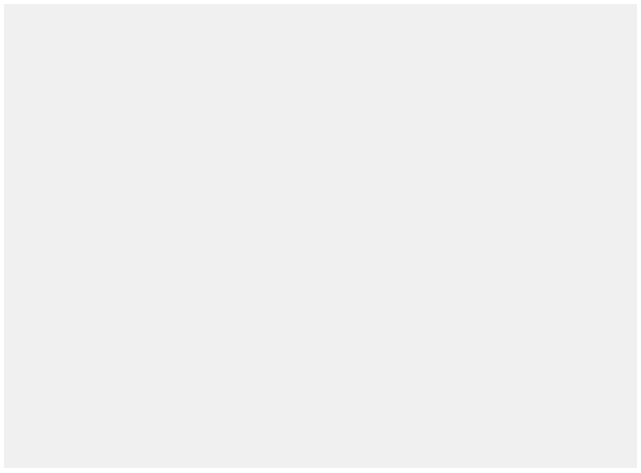
+Wait for a few days , after you see the confirmation email, Open your IT Workplace to enable administrators rights (check the pictures below for descriptions)



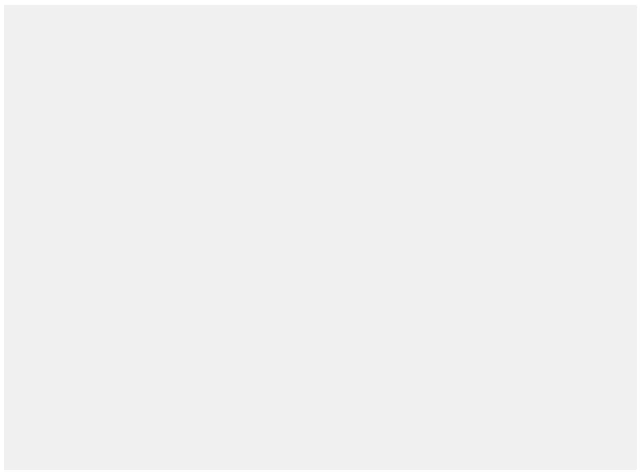


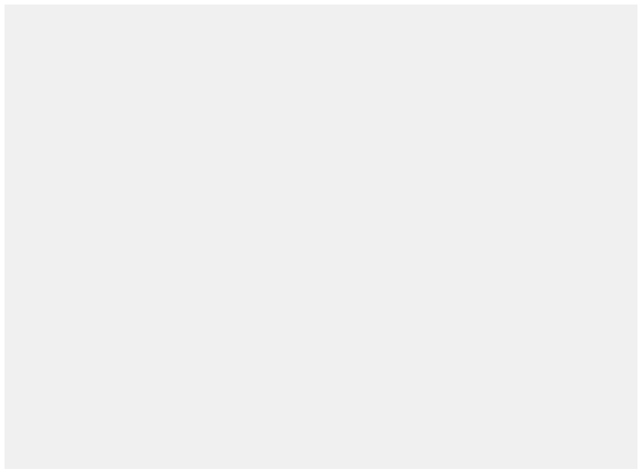
Step 3 :

_ Open **SQL Server 2022 Configuration Manager** as administrator (right-click → **run with administrator**) to open



_Go to **SQL Server Network Configuration** → right-click the Protocols for **SQLEXPRESS** → **Properties**





_ Enable the **TCP/IP**

+**Purpose**:

1. **Allow Remote Connections**
   TCP/IP is a network protocol that enables communication between devices on a network. By enabling TCP/IP, you allow SQL Server to accept incoming connections from clients or applications located on different machines. This is critical in multi-tier applications where the database server and client applications are hosted on separate systems.

2. **Support for Standard Communication Protocols**

Most SQL Server client libraries, like ADO.NET or ODBC, use TCP/IP as the default communication protocol.Enabling TCP/IP ensures compatibility with these libraries and tools.

3. **Enable Cross-Network Communication**

If your SQL Server instance needs to be accessed across subnets, offices, or even the internet (with proper security), TCP/IP is the protocol used to facilitate this communication.

4. **Default for Named Instances**

SQL Server named instances use the **SQL Server Browser** service to listen for client connections on a dynamic port.

Enabling TCP/IP ensures that named instances can communicate effectively with client applications.

5. Integration with Applications

Applications like web servers, business intelligence tools, or custom software often connect to SQL Server over TCP/IP. Without enabling TCP/IP, these applications would fail to connect if they are not on the same machine as the SQL Server instance.

_Next, go to **IPAddresses** tab, scroll down to find the **IPALL,** set the **TCP Port** to 1433

+Purpose: Setting the TCP port of IPALL to 1433 in SQL Server Configuration Manager serves a specific purpose related to how SQL Server communicates over a network. Here's why it is important:

1. Default Port for SQL Server

Port 1433 is the default port for SQL Server's TCP/IP protocol.

By setting this port, you standardize communication for the default instance of SQL Server, making it easier for client applications to connect without specifying a custom port.

2. Simplifies Client Configuration

When the TCP port is set to 1433, client applications do not need to explicitly specify the port in their connection strings.

For example:

Server=myserver;Database=mydb;UserId=myuser;Password=mypassword;

If a custom port is used, the connection string must explicitly include it:

Server=myserver,12345;Database=mydb;UserId=myuser;Password=mypassword;

3. Compatibility with SQL Server Tools

SQL Server Management Studio (SSMS) and other tools expect the default instance to listen on port 1433.

Setting this port ensures seamless connectivity with these tools without additional configuration.

4. Consistency Across Environments

Using the default port ensures consistency across different environments (e.g., development, testing, production), reducing potential configuration errors.

5. Support for Named Instances

For named instances, SQL Server dynamically assigns ports unless explicitly configured.

Setting TCP Port 1433 under IPALL can be used for a default instance or for named instances where a static port is desired.

6. Facilitates Firewall Configuration

Network administrators often configure firewalls to allow traffic on port 1433 for SQL Server.Using this port avoids the need for additional firewall changes or troubleshooting connection issues caused by blocked custom ports.

STEP 3: Test your connection

_After everything is completed , open your Command Prompt, enter the following code:

```
Sqlcmd -S <ServerName>,<port>
```

If the output is 1> (or something like that)

You have successfully set and establish a new connection to SQL Server on port 1433

(tutorial project here [: https://thriveread.com/nestjs-typeorm-mssql/)](https://thriveread.com/nestjs-typeorm-mssql/)