This is a simplified guide on how to deploy the database for testing. It is not representative of a production environment. This guide does not cover:

- Setting up a key for our custom assembly
- Adding our dependencies as trusted assemblies (the TRUSTWORTHY setting is used to bypass these requirements)
- Setting up the Powershell script and associated event trigger that automatically blocks login attempts from port scanners.
- Setting up the EC2 security group traffic rules properly (we're just allowing all traffic to make our lives easier)

Steps (quick and dirty database setup):

1. Set up a new SQL Server Express install on an Amazon EC2 instance.
    - Set up the security group to allow all inbound traffic (normally we'd only allow all traffic on port 1433 and allow access to RDP ports from select locations)
    - Also install SQL Server Management Studio (SSMS) which we need to configure some things. Install it on the server for now.
2. Open up SSMS to tweak some settings:
    - Enable mixed mode authentication:
        - After connecting to the database, right click the server in the Object Explorer (the topmost item in the tree), and click Properties. A "Server Properties" window will open.
        - Under Security, click "SQL Server and Windows Authentication mode".
    - Enable database containment:
        - In the same Server Properties window, go to the Advanced page, and set "Enable Contained Databases" to True.
        - Make sure to apply these.
3. Enable CLR integration with the following commands (hit "New Query" in SSMS to open a query box, and hit Execute to run it):

    ```
    EXEC sp_configure 'show advanced options', 1;
    RECONFIGURE;
    GO
    EXEC sp_configure 'clr enabled', 1;
    RECONFIGURE;
    GO
    ```

4. Right click "Databases" in the object explorer and select "New Database". Put in an appropriate name, and under Options set "Containment type" to "Partial". Create the database. The guide will assume the name is "TestDatabase".
5. In a query window, enter this command and run it:

    ```
    ALTER DATABASE TestDatabase SET TRUSTWORTHY ON;
    ```

(Normally we'd set up a key pair for our custom assembly and add the checksums of our dependencies to a list of trusted assemblies, but that's a bit of an involved process and for testing things out the Trustworthy option is faster. Trustworthy would not be used in a production environment.)

6. We need to add some dependency assemblies. Open a query window. Make sure "TestDatabase" is selected next to the Execute button. Use this query:

   ```
   CREATE ASSEMBLY [System.Runtime.Serialization] FROM
   'C:\Windows\Microsoft.NET\Framework64\v4.0.30319\System.Runtime
   .Serialization.dll' WITH PERMISSION_SET = UNSAFE
   GO
   CREATE ASSEMBLY [System.ServiceModel.Internals] FROM
   'C:\Windows\Microsoft.NET\Framework64\v4.0.30319\System.Service
   Model.Internals.dll' WITH PERMISSION_SET = UNSAFE
   GO
   CREATE ASSEMBLY [SMDiagnostics] FROM
   'C:\Windows\Microsoft.NET\Framework64\v4.0.30319\SMDiagnostics.
   dll' WITH PERMISSION_SET = UNSAFE
   GO
   ```

7. We will be installing the third party library Newtonsoft.Json. Download from this link:
   - https://github.com/JamesNK/Newtonsoft.Json/releases/download/12.0.3/Json120r3.zip
   - From this zipfile, browse to Bin/net45 and copy Newtonsoft.Json.dll to somewhere on the server. The guide will assume C:\Newtonsoft.Json.dll
   - Run this query like before:

     ```
     CREATE ASSEMBLY [Newtonsoft.Json] FROM
     'C:\Newtonsoft.Json.dll' WITH PERMISSION_SET = UNSAFE
     GO
     ```
8. The database should now be ready to deploy from the pipeline. Make a copy of the pipeline and reconfigure the values to match the new database's address and name and login details.
9. You will also need to change the server address in the source code.