EBIX-PORTAL

Prepared By: Mohammad Khan

0449799334

The Requirements from EBIX as followed:

**Task 1:**

Task 1 – Database I. Create a SQL Server database called Ebix

II.

Create a table called ‘Client’ and create the following fields: a. ClientId b. FirstName c. Surname d. DateOfBirth e. Address f. Suburb g. State h. Postcode i. Telephone

III.

Create a table called ‘Invoice’ and create the following fields: a. InvoiceId b. ClientId c. InvoiceDate d. InvoiceNo a. InvoiceAmount

IV.

Use your discretion to determine the best field types

V.

Create a View joining the 2 tables that displays the following fields sorted by InvoiceDate. a. InvoiceID b. InvoiceDate c. InvoiceNo d. ClientID e. ClientName

To do that I have used SQL2019 management Studio with following command:

USE [master]

GO

/\*\*\*\*\*\* Object: Database [EBIX] Script Date: 30/09/2020 3:06:58 PM \*\*\*\*\*\*/

CREATE DATABASE [EBIX]

CONTAINMENT = NONE

ON PRIMARY

( NAME = N'EBIX', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\EBIX.mdf' , SIZE = 8192KB , MAXSIZE = UNLIMITED, FILEGROWTH = 65536KB )

LOG ON

( NAME = N'EBIX\_log', FILENAME = N'C:\Program Files\Microsoft SQL Server\MSSQL15.MSSQLSERVER\MSSQL\DATA\EBIX\_log.ldf' , SIZE = 8192KB , MAXSIZE = 2048GB , FILEGROWTH = 65536KB )

WITH CATALOG\_COLLATION = DATABASE\_DEFAULT

GO

USE [EBIX]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Client] Script Date: 30/09/2020 3:08:13 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Client](

[ClientId] [int] IDENTITY(1,1) NOT NULL,

[FirstName] [varchar](50) NOT NULL,

[SurName] [varchar](50) NULL,

[DateOfBirth] [date] NOT NULL,

[Address] [varchar](200) NULL,

[Suburb] [varchar](50) NULL,

[State] [varchar](50) NULL,

[Postcode] [int] NULL,

[Telephone] [int] NULL,

CONSTRAINT [PK\_Client] PRIMARY KEY CLUSTERED

(

[ClientId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

USE [EBIX]

GO

/\*\*\*\*\*\* Object: Table [dbo].[Invoice] Script Date: 30/09/2020 3:09:51 PM \*\*\*\*\*\*/

SET ANSI\_NULLS ON

GO

SET QUOTED\_IDENTIFIER ON

GO

CREATE TABLE [dbo].[Invoice](

[InvoiceId] [int] IDENTITY(1,1) NOT NULL,

[ClientId] [int] NOT NULL,

[InvoiceDate] [date] NOT NULL,

[InvoiceNo] [varchar](50) NOT NULL,

[InvoiceAmount] [int] NOT NULL,

CONSTRAINT [PK\_Invoice] PRIMARY KEY CLUSTERED

(

[InvoiceId] ASC

)WITH (PAD\_INDEX = OFF, STATISTICS\_NORECOMPUTE = OFF, IGNORE\_DUP\_KEY = OFF, ALLOW\_ROW\_LOCKS = ON, ALLOW\_PAGE\_LOCKS = ON, OPTIMIZE\_FOR\_SEQUENTIAL\_KEY = OFF) ON [PRIMARY]

) ON [PRIMARY]

GO

ALTER TABLE [dbo].[Invoice] WITH CHECK ADD FOREIGN KEY([ClientId])

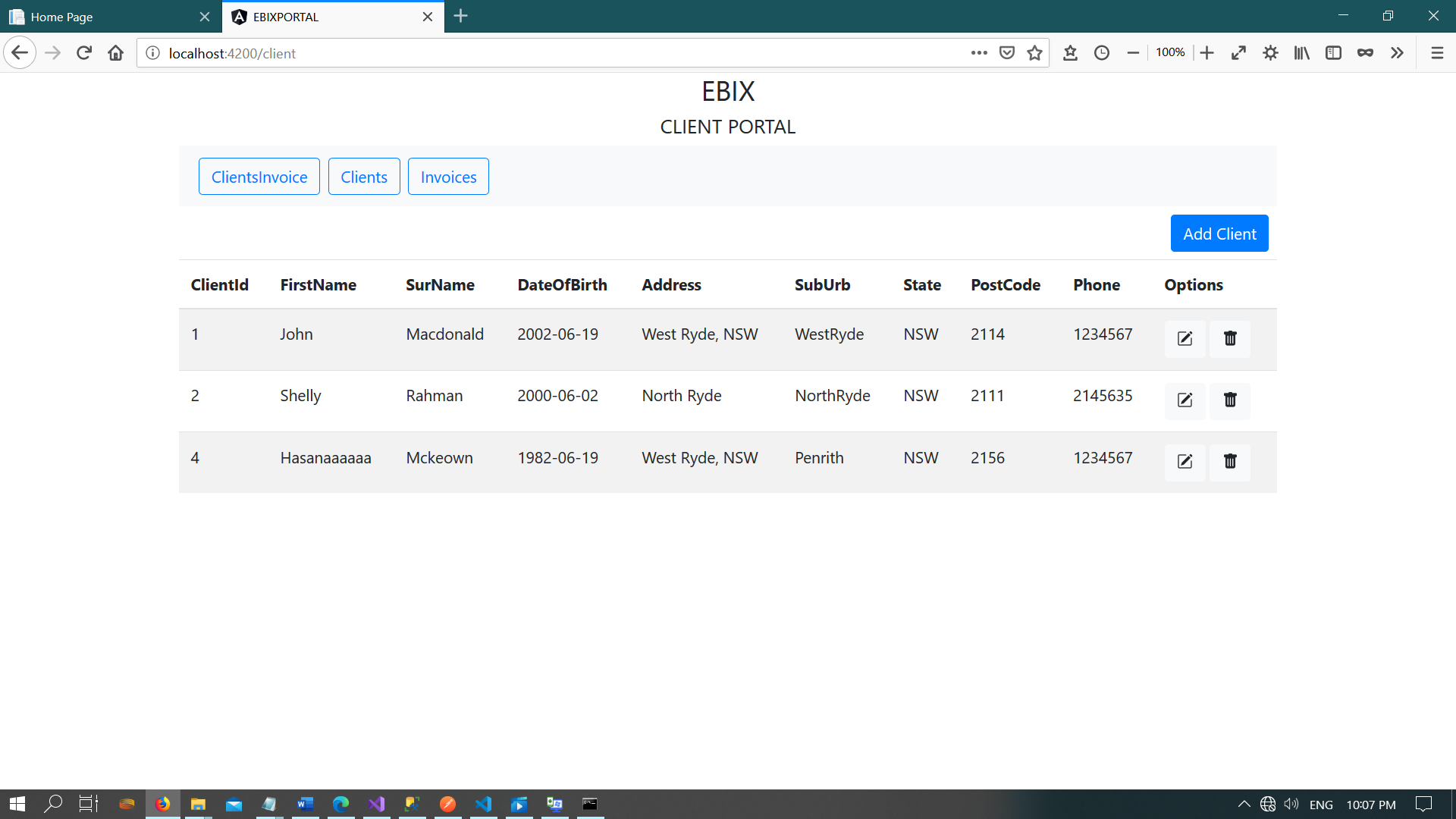
REFERENCES [dbo].[Client] ([ClientId])

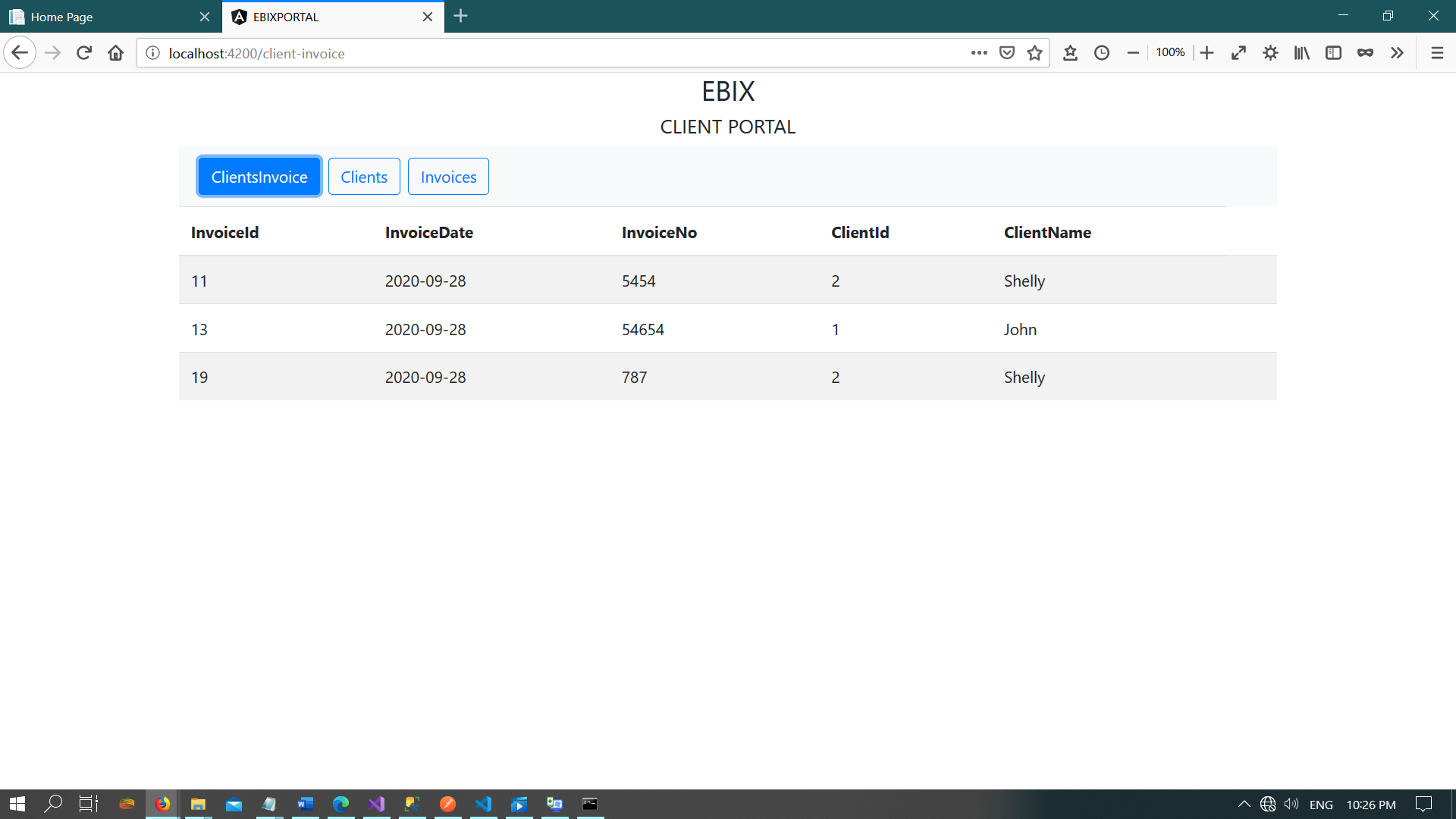
GO

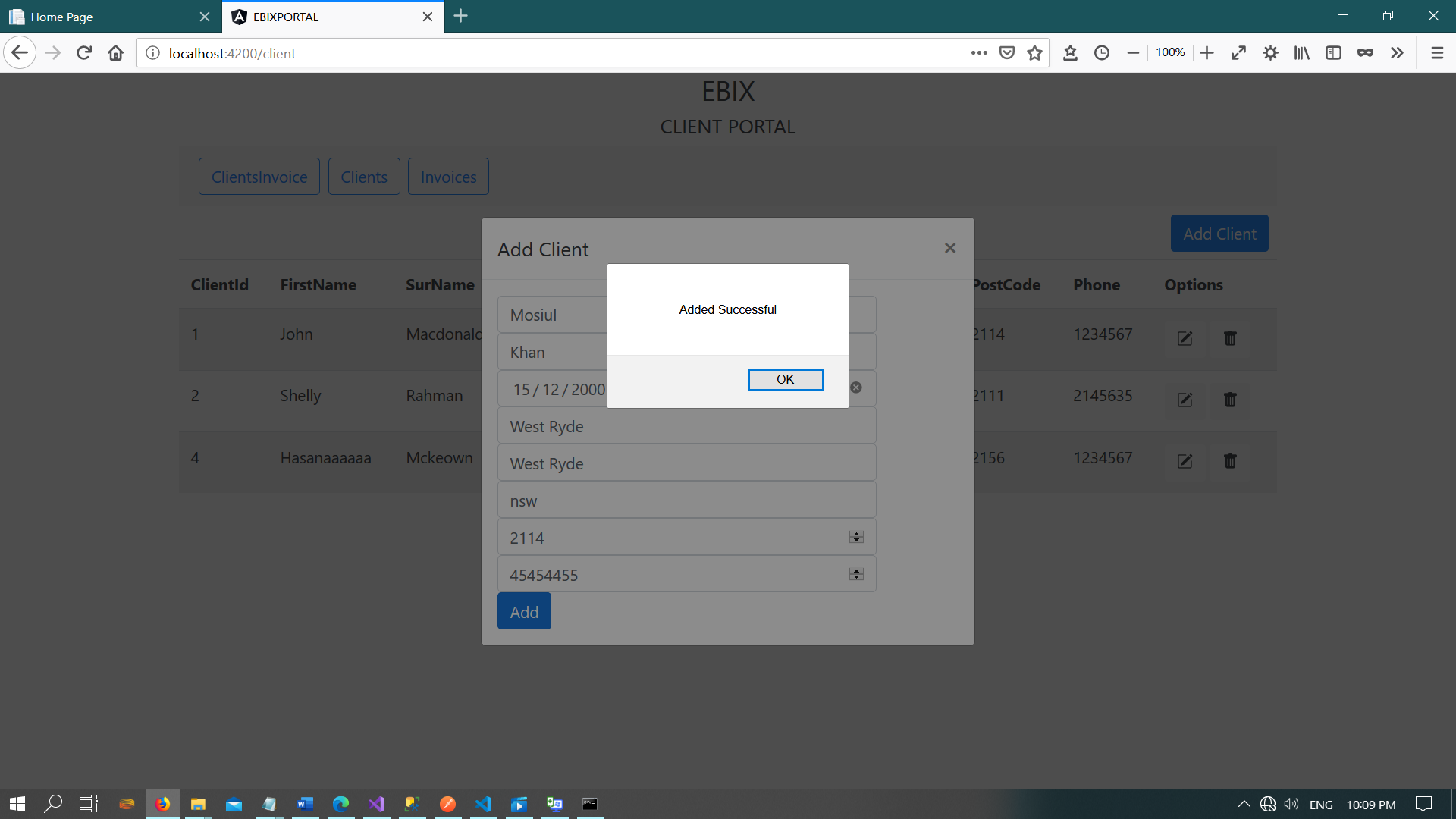
Task 2 –

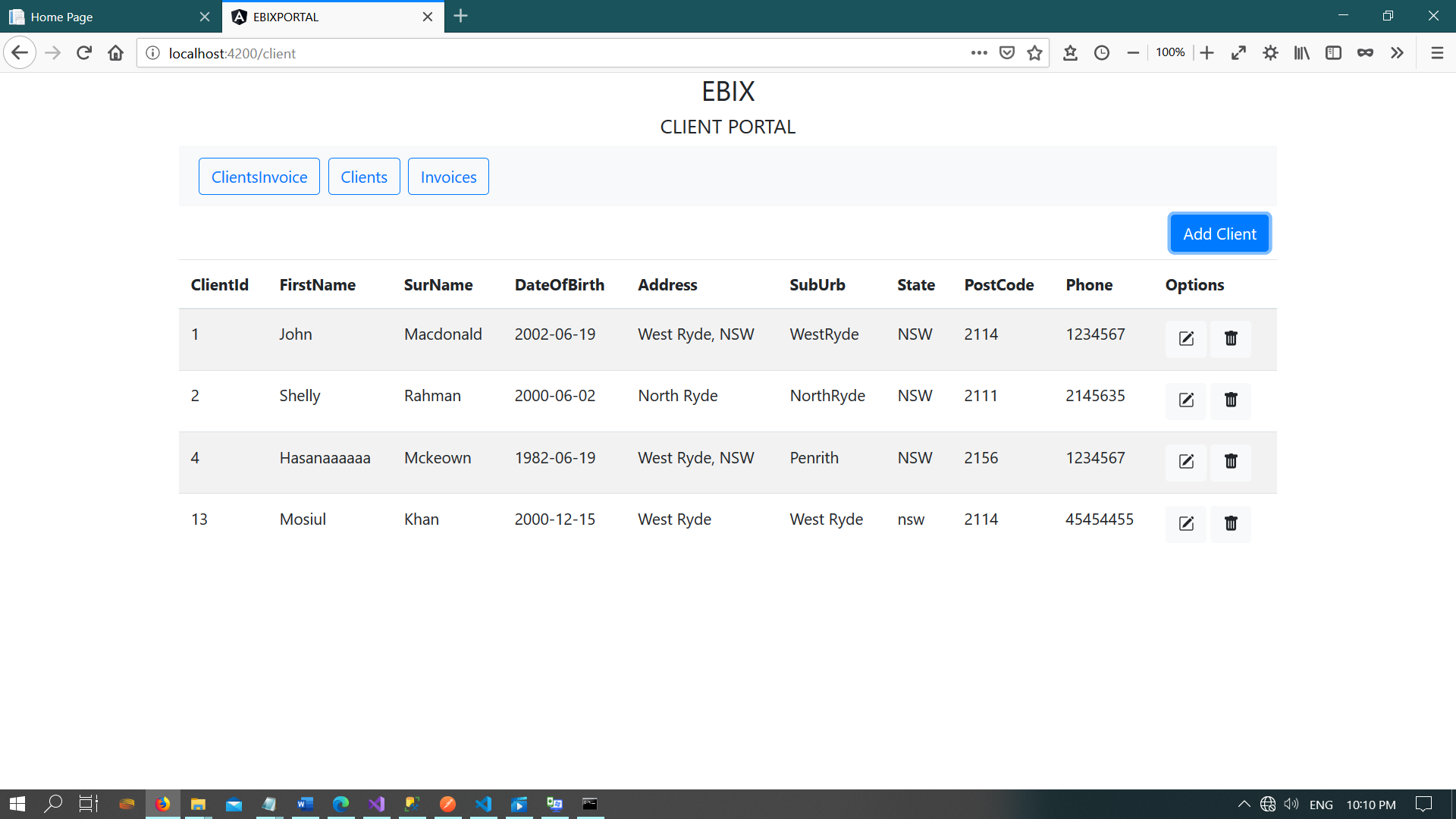
Web Page

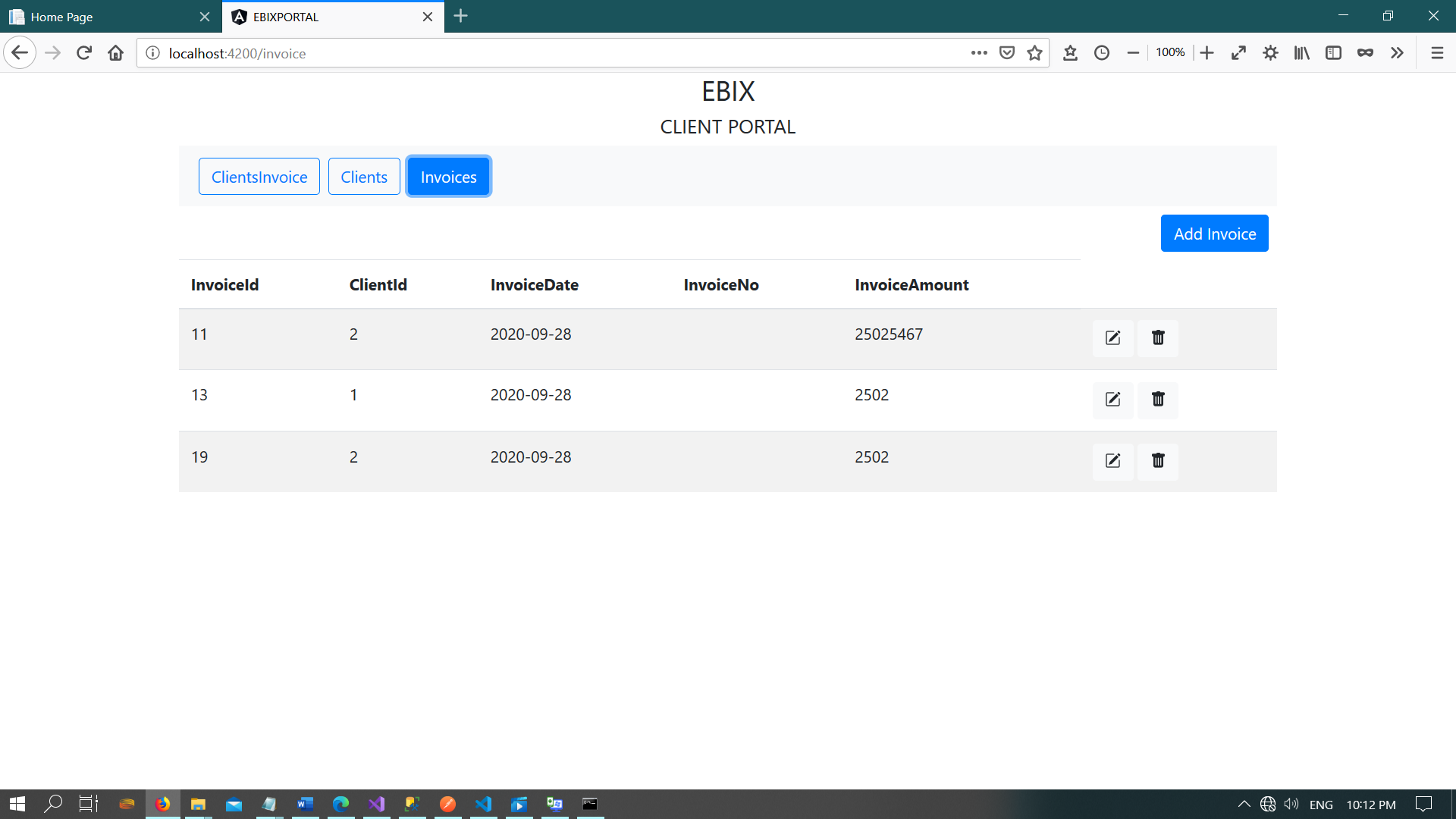
1. Create a website using any .Net technology you are comfortable with.
2. Include Bootstrap.
3. Create a page containing a table that lists the output of the view created in Task 1 step V
4. All fields should be formatted using a relevant Bootstrap style.
5. The page and table should have appropriate headings and be visually appealing
6. Dates should be displayed in the following format – 01-Jan-2020
7. Bonus marks for using client side frameworks like Angular, React etc.

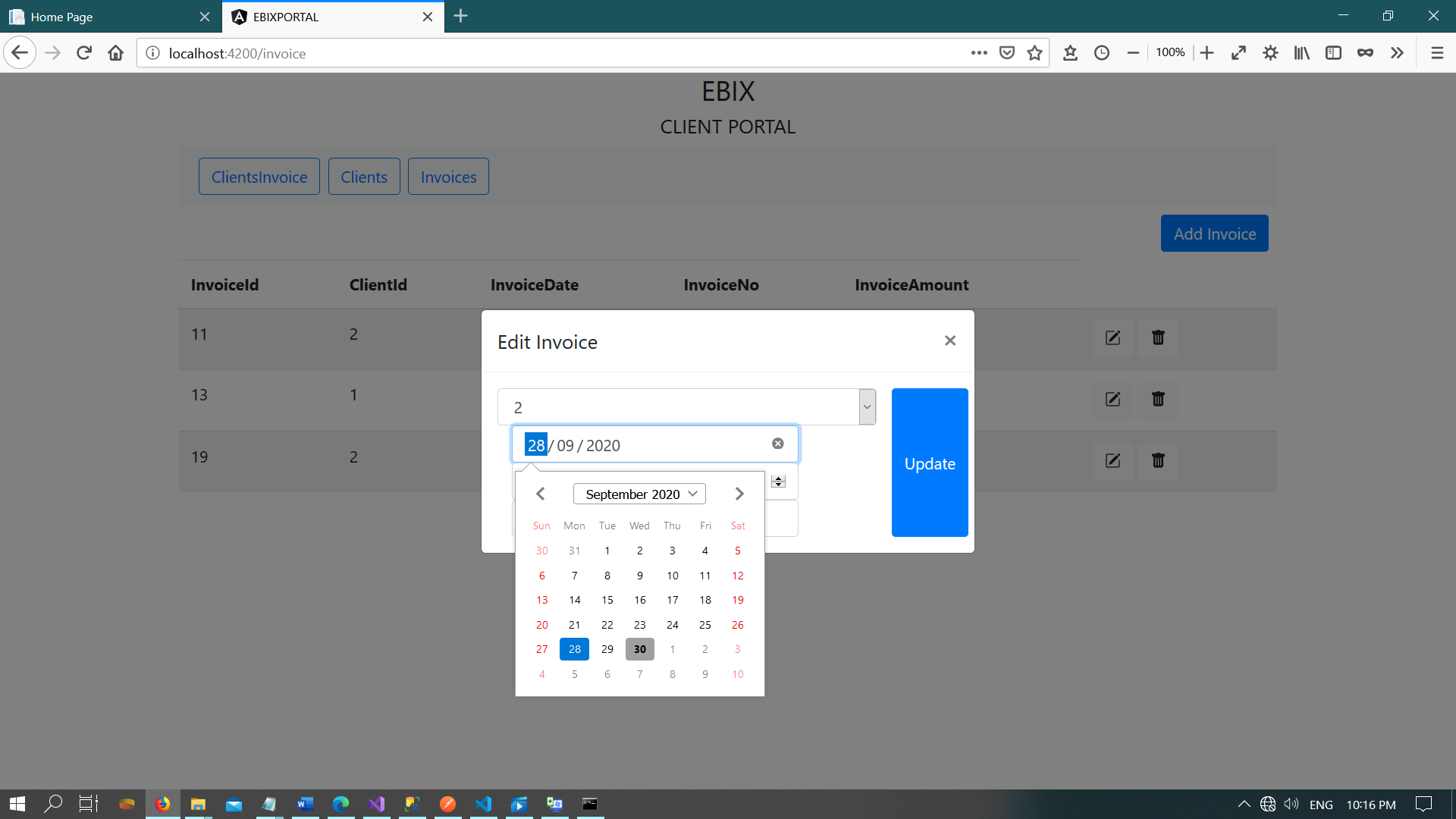












Task 3 – API

I. Create a RESTful web API that performs insert and update operations against the ‘Client’

table created in Task 1.

II. Interface your web page to your API. This page should be able to insert a new client record

and update an existing client record.

III. Add validation to at least one field, making sure no invalid data can make its way to the

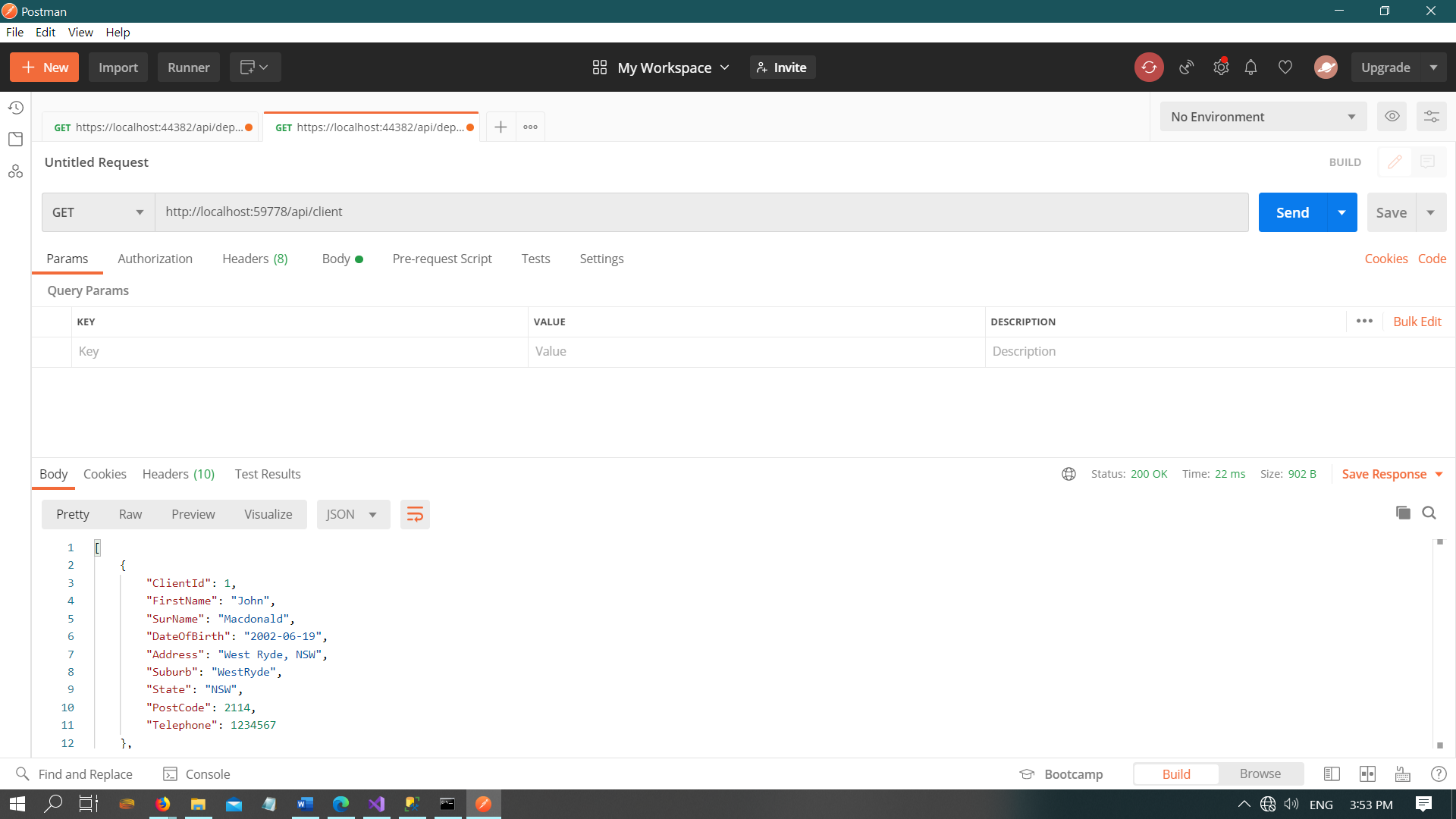
database.

IV. All fields should be formatted using a relevant Bootstrap style.

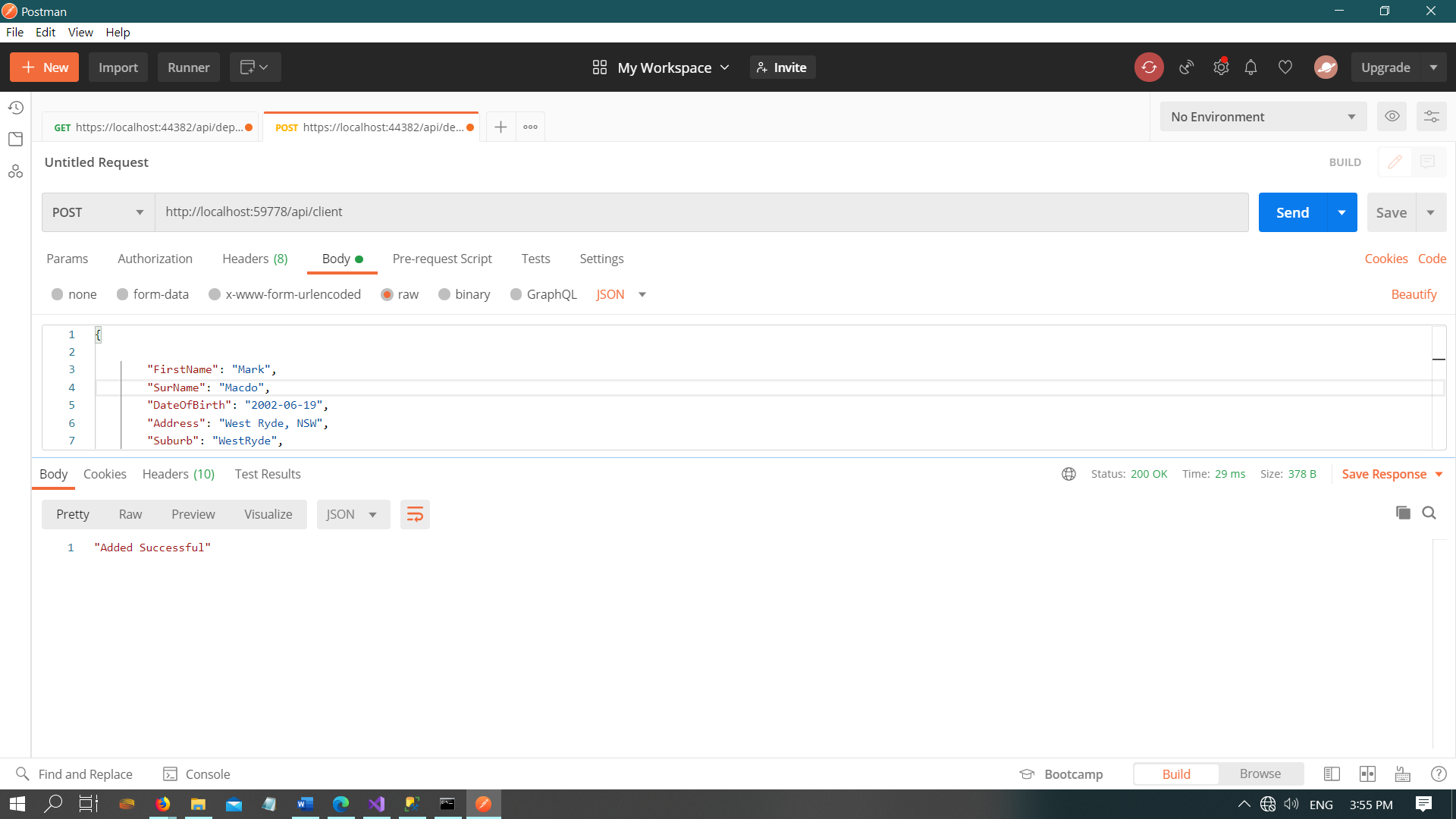
V. The page and fields should have appropriate headings and be visually appealing

VI. Dates should be displayed in the following format ‘01-Jan-2020’

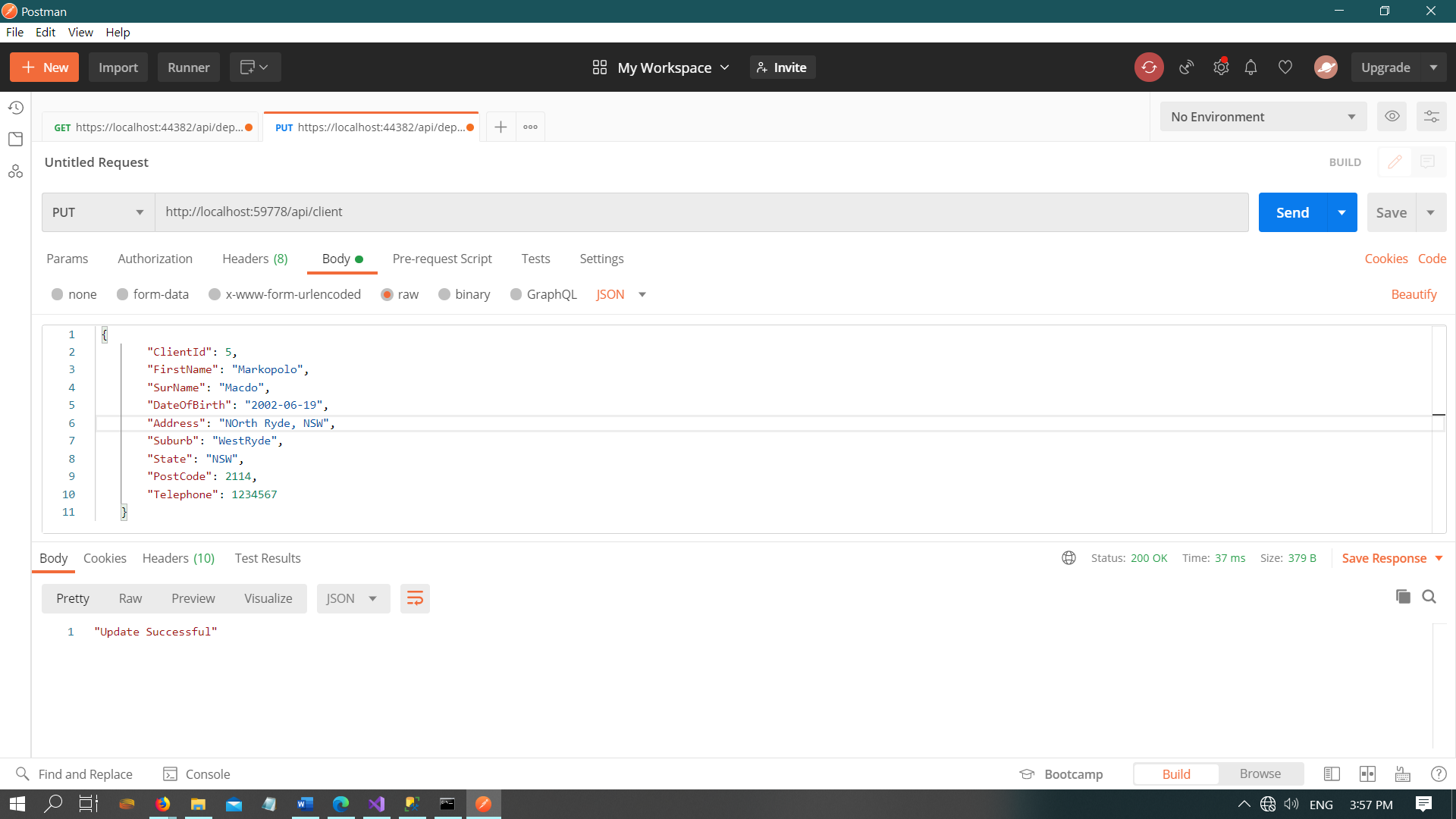
Get Client Details



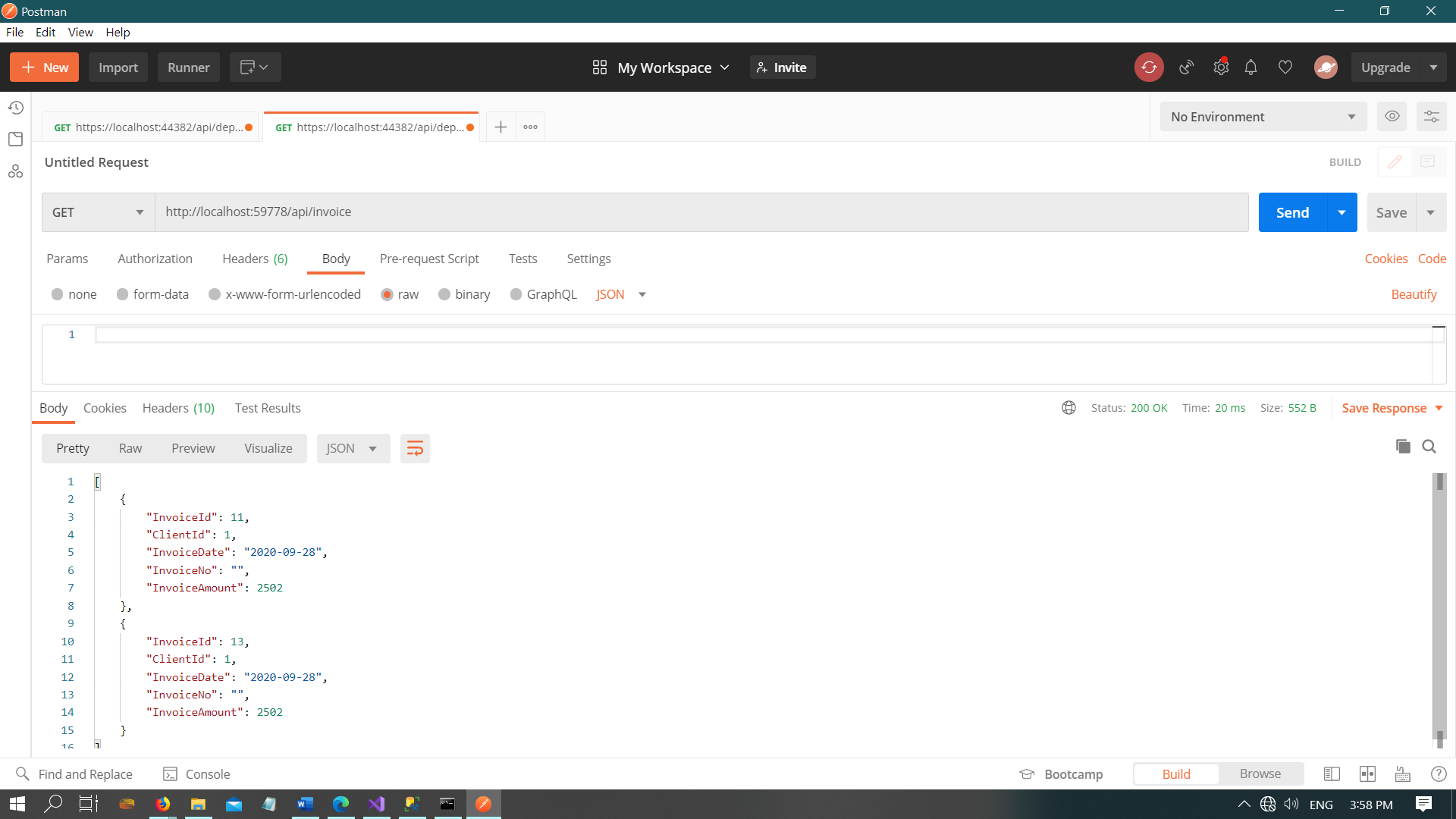
1. Test : Insert Client



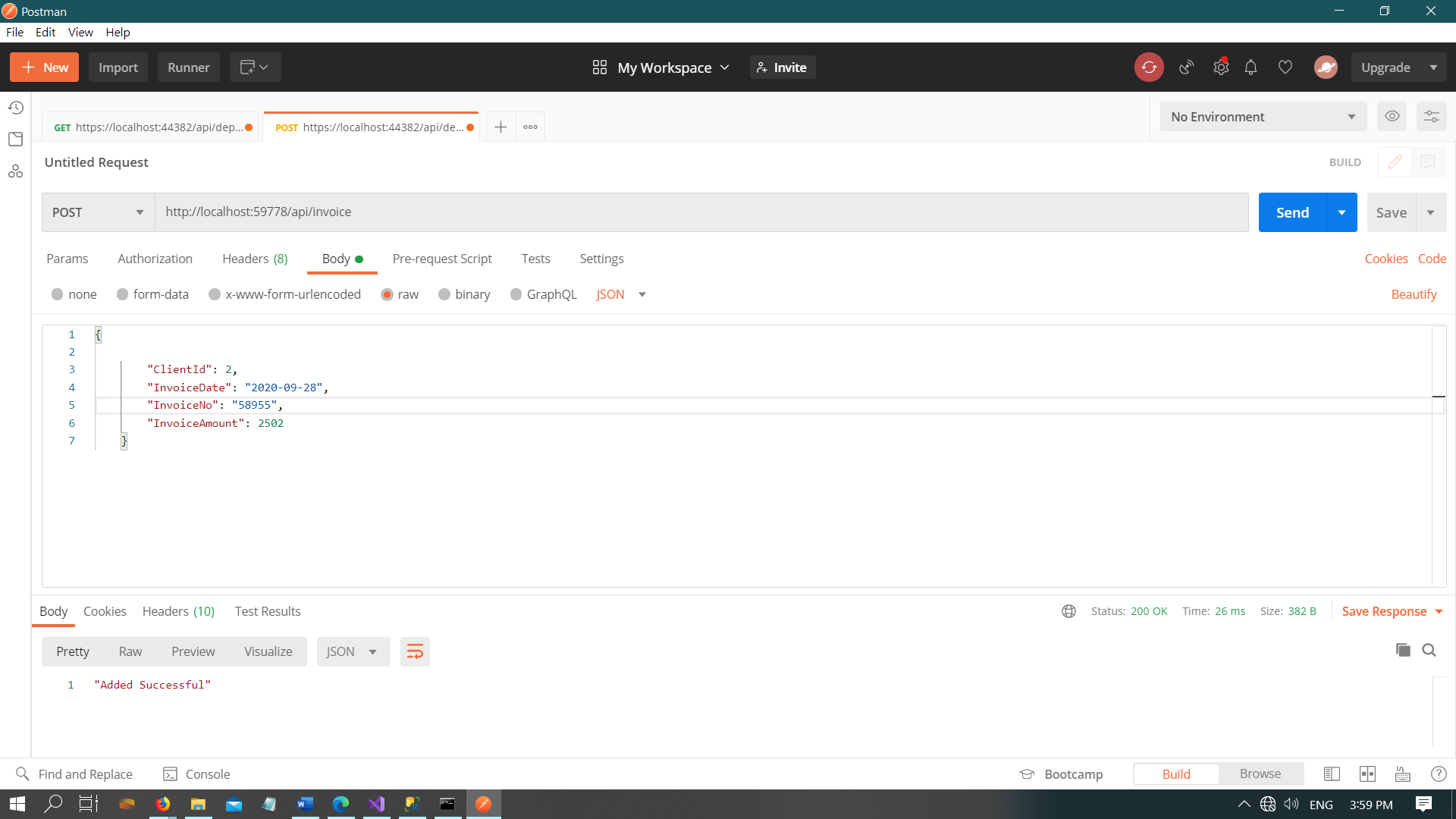
1. Update Client



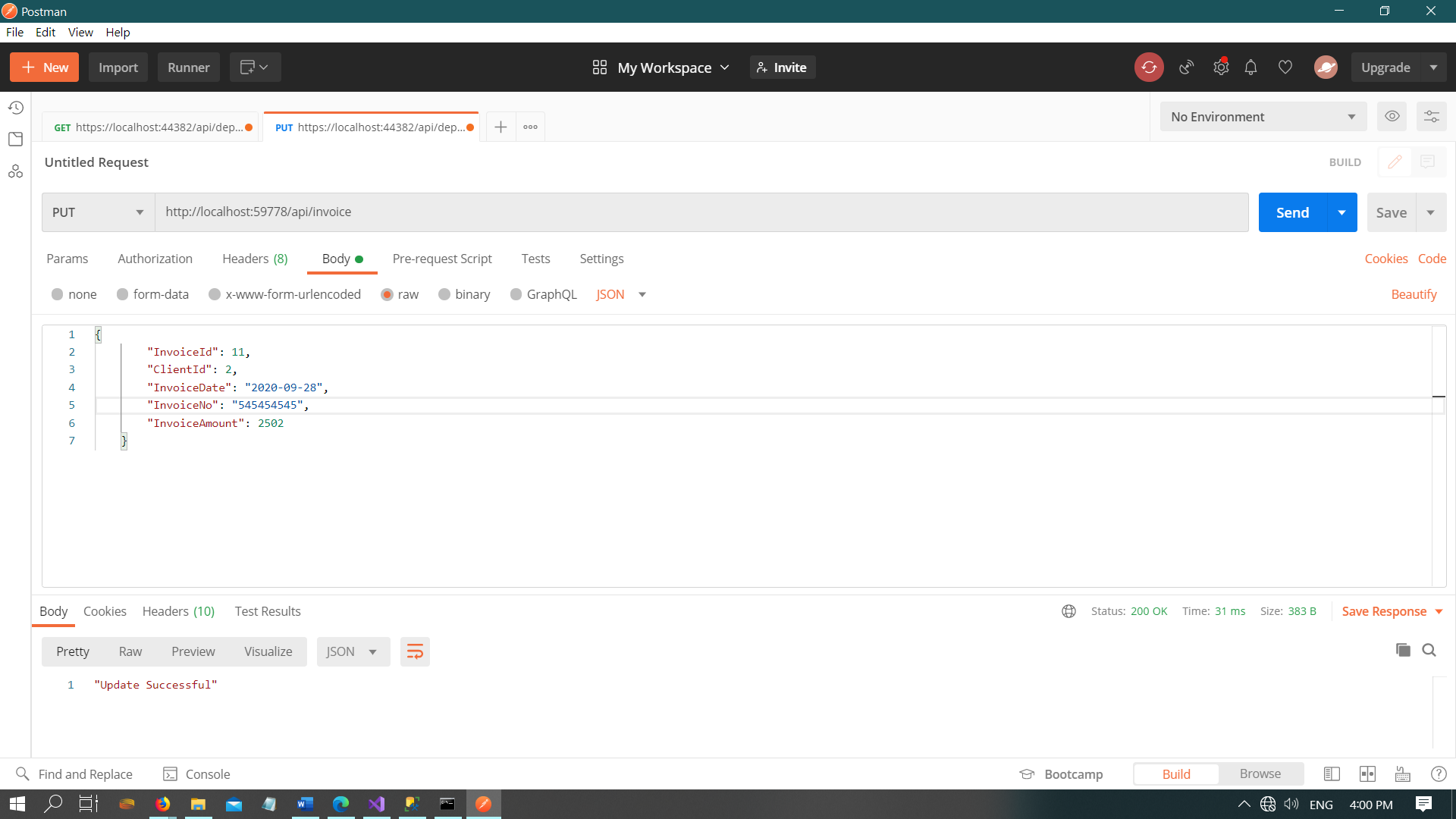
1. Add Invoice:



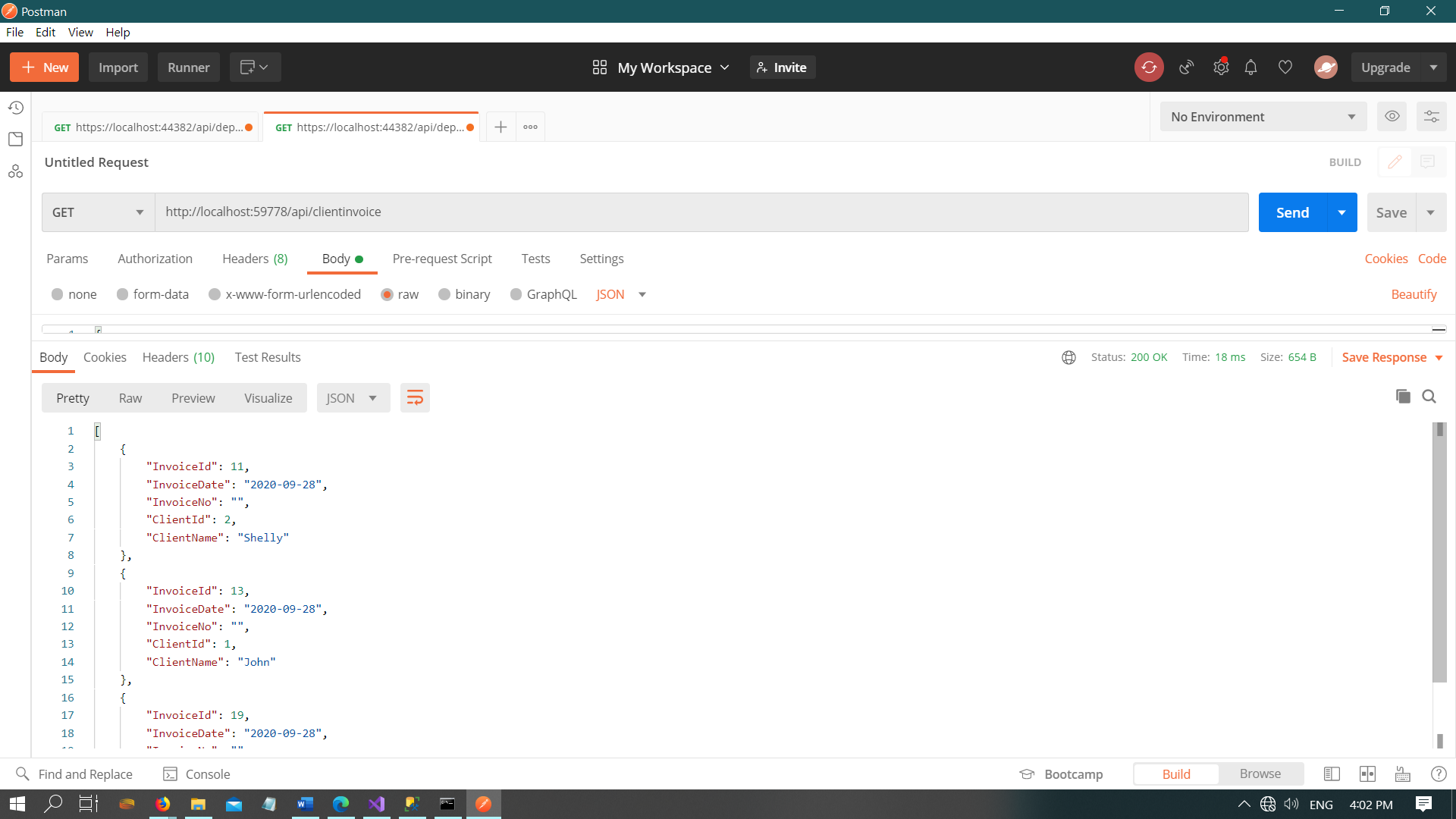
1. Insert Invoice



1. Update Invoice:



1. Client Invoice show:



using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EBIX\_WEBAPI.Models

{

public class Client

{

public int ClientId { get; set; }

public string FirstName { get; set; }

public string SurName { get; set; }

public string DateOfBirth { get; set; }

public string Address { get; set; }

public string Suburb { get; set; }

public string State { get; set; }

public int PostCode { get; set; }

public int Telephone { get; set; }

}

}

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

namespace EBIX\_WEBAPI.Models

{

public class Invoice

{

public int InvoiceId { get; set; }

public int ClientId { get; set; }

public string InvoiceDate { get; set; }

public string InvoiceNo { get; set; }

public int InvoiceAmount { get; set; }

}

}

WebConfig Files:

<?xml version="1.0" encoding="utf-8"?>

<!--

For more information on how to configure your ASP.NET application, please visit

https://go.microsoft.com/fwlink/?LinkId=301879

-->

<configuration>

<appSettings>

<add key="webpages:Version" value="3.0.0.0" />

<add key="webpages:Enabled" value="false" />

<add key="ClientValidationEnabled" value="true" />

<add key="UnobtrusiveJavaScriptEnabled" value="true" />

</appSettings>

<connectionStrings>

<add name="EBIXAppDB" connectionString="Data Source=.;Initial Catalog=EBIX;Integrated Security=True" providerName="System.Data.SqlClient"/>

</connectionStrings>

**CONTROLLERS:**

**INVOICE CONTROLLER**

using EBIX\_WEBAPI.Models;

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EBIX\_WEBAPI.Controllers

{

public class InvoiceController : ApiController

{

//Get Method will return Invoice Details from Invoice Table

//Query must use from stored procedure in SQL but for demo purpose i am using it from C# code

public HttpResponseMessage Get()

{

try

{

string query = @"

select InvoiceId,ClientId,convert(varchar(20),

invoiceDate,120) as InvoiceDate, InvoiceNo,InvoiceAmount

from dbo.Invoice

";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return Request.CreateResponse(HttpStatusCode.OK, table);

}

catch(Exception)

{

return Request.CreateResponse("Problem Occors! please check Sql server and local host server");

}

}

//This method will Insert data into invoice Table

public string Post(Invoice inv)

{

try

{

string query = @"insert into dbo.Invoice values

(

'" + inv.ClientId + @"'

,'" + inv.InvoiceDate + @"'

,'" + inv.InvoiceNo + @"'

,'" + inv.InvoiceAmount + @"'

)";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Added Successful";

}

catch(Exception)

{

return "Failed to Insert";

}

}

//Put() Method will update data into Invoice Table

public string Put(Invoice inv)

{

try

{

string query = @"update dbo.Invoice set

clientId ='" + inv.ClientId + @"'

,invoiceDate = '" + inv.InvoiceDate + @"'

,invoiceNo = '" + inv.InvoiceNo + @"'

,invoiceAmount = '" + inv.InvoiceAmount + @"'

where InvoiceId=" + inv.InvoiceId + @"";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Update Successful";

}

catch (Exception)

{

return "Failed to Update";

}

}

//Delete Method will delete data from Invoice

public string Delete(int id)

{

try

{

string query = @"delete from dbo.Invoice

where InvoiceId=" + id + @"";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Delete Successful";

}

catch (Exception)

{

return "Failed to Delete";

}

}

//GetAllClient method will return all client stored into CLient table

[Route("api/Invoice/GetAllClient")]

[HttpGet]

public HttpResponseMessage GetAllClient()

{

try {

string query = @"

select ClientId from dbo.Client

";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return Request.CreateResponse(HttpStatusCode.OK, table);

}

catch (Exception)

{

return Request.CreateResponse("Problem Occors! please check Sql server and local host server");

}

}

}

}

**CLIENT CONTROLLER**

using EBIX\_WEBAPI.Models;

using System;

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EBIX\_WEBAPI.Controllers

{

public class ClientController : ApiController

{

public HttpResponseMessage Get()

{

string query = @"

select ClientId,FirstName,SurName,convert(varchar(20),

DateOfBirth,120) as DateOfBirth, Address,Suburb,State,PostCode,Telephone

from dbo.Client

";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return Request.CreateResponse(HttpStatusCode.OK, table);

}

public string Post(Client cln)

{

string query = @"insert into dbo.Client values

(

'" + cln.FirstName + @"'

,'" + cln.SurName + @"'

,'" + cln.DateOfBirth + @"'

,'" + cln.Address + @"'

,'" + cln.Suburb + @"'

,'" + cln.State + @"'

,'" + cln.PostCode + @"'

,'" + cln.Telephone + @"'

)";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Added Successful";

}

public string Put(Client cln)

{

try

{

string query = @"update dbo.Client set

FirstName ='" + cln.FirstName + @"'

,SurName = '" + cln.SurName + @"'

,DateOfBirth = '" + cln.DateOfBirth + @"'

,Address = '" + cln.Address + @"'

,Suburb = '" + cln.Suburb + @"'

,State = '" + cln.State + @"'

,PostCode = '" + cln.PostCode + @"'

,Telephone = '" + cln.Telephone + @"'

where ClientId=" + cln.ClientId + @"";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Update Successful";

}

catch (Exception)

{

return "Failed to Update";

}

}

public string Delete(int id)

{

try

{

string query = @"delete from dbo.Client

where ClientId=" + id + @"";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return "Delete Successful";

}

catch (Exception)

{

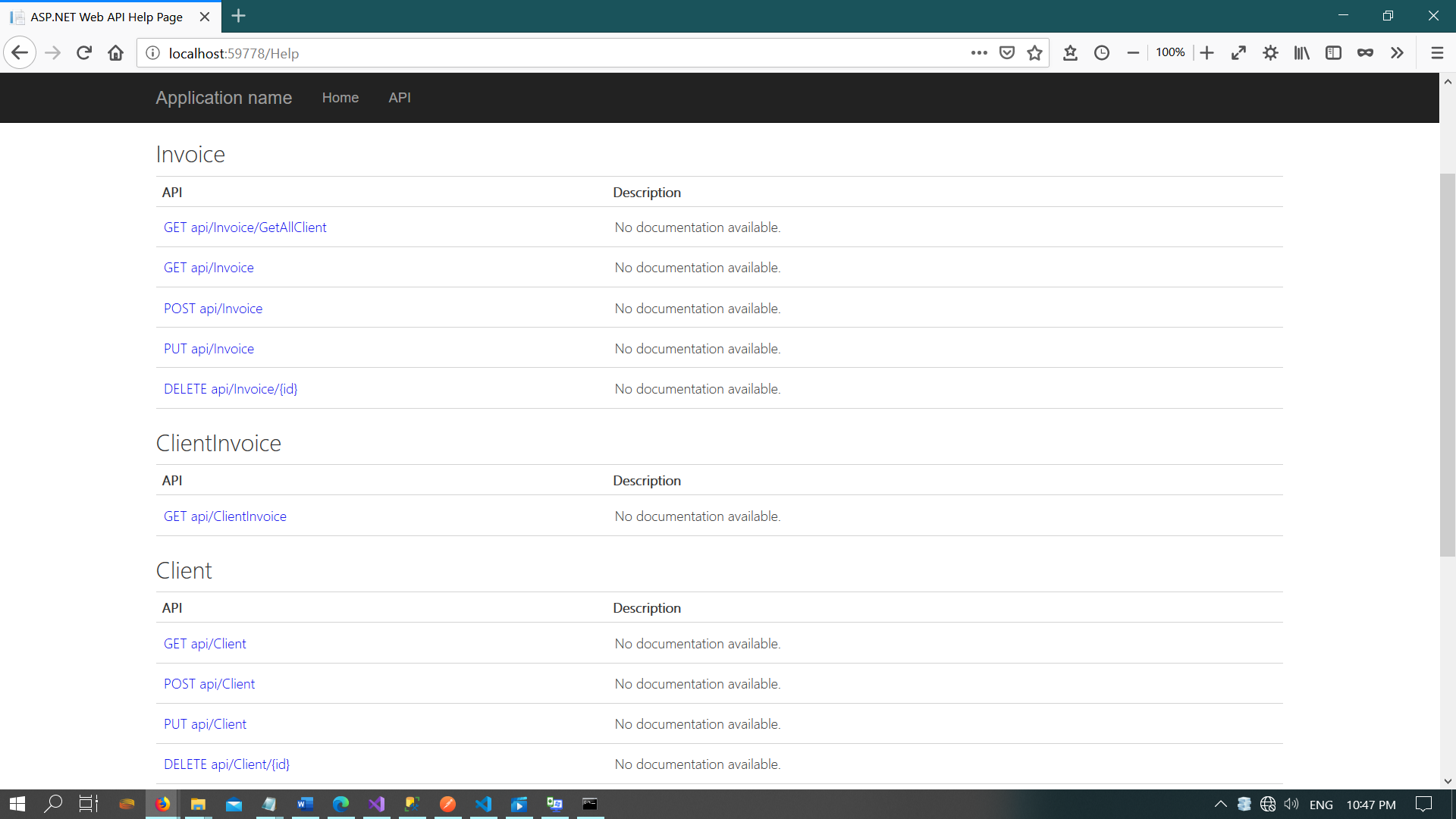
return "Failed to Delete";

}

}

}

}



**CLIENT-INVOICE CONTROLLER**

using System.Configuration;

using System.Data;

using System.Data.SqlClient;

using System.Net;

using System.Net.Http;

using System.Web.Http;

namespace EBIX\_WEBAPI.Controllers

{

public class ClientInvoiceController : ApiController

{

public HttpResponseMessage Get()

{

string query = @"

select InvoiceId,convert(varchar(20),

invoiceDate,120) as InvoiceDate, InvoiceNo,ClientId,ClientName

from dbo.ClientInvoice

";

DataTable table = new DataTable();

using (var conn = new SqlConnection(ConfigurationManager.ConnectionStrings["EBIXAppDB"].ConnectionString))

using (var cmd = new SqlCommand(query, conn))

using (var da = new SqlDataAdapter(cmd))

{

cmd.CommandType = CommandType.Text;

da.Fill(table);

}

return Request.CreateResponse(HttpStatusCode.OK, table);

}

}

}

Task 4 – Documentation Create a document to explain how you completed each task and why you chose each particular technology.

1. Create Database and table
2. Create API with C# .net Framework
3. Install postman and check each of the API output/input
4. Created a website with Angular 10 and call all api function from API
5. Use Bootstrap to make presentable.

What I did not use:

1. Function I have written in C# .Net Framework it should be written from stored procedure otherwise a chance of sqlInjection could have happened.
2. Have to put more validation in every single textbox, otherwise user can put incorrect data on it.

Technology used:

C# .Net Framework => For Creating API

POSTMAN => For checking API

SQL Server => Creating Database and tables

Visual Studio Code => To write Angular Code

CMD => to install Angular, run and test

BootStrap => for style and presentation.

**Why I use them:**

Those are latest technology .

Angular with .Net Framework work very faster then traditional approach of HTML/CSS/JavaScript.

SPECIAL INSTRUCTION: Please change the URL in the shared service.ts File

Thank you