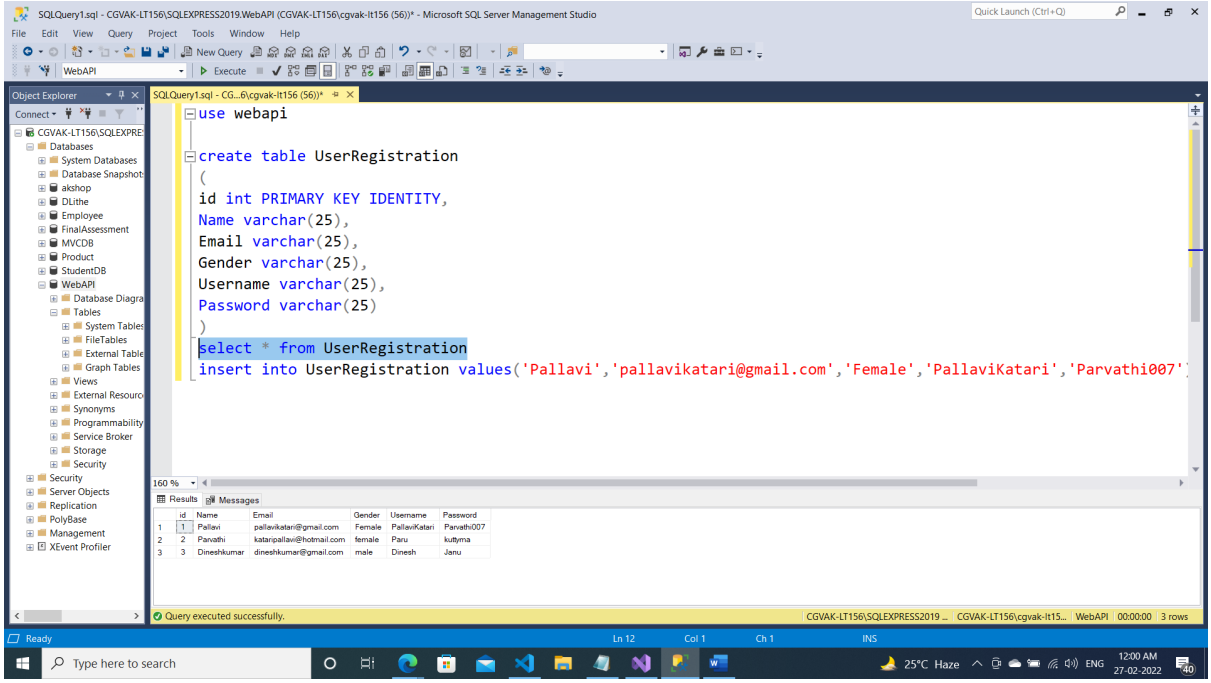
CONSUME WEB API FROM ANGULAR

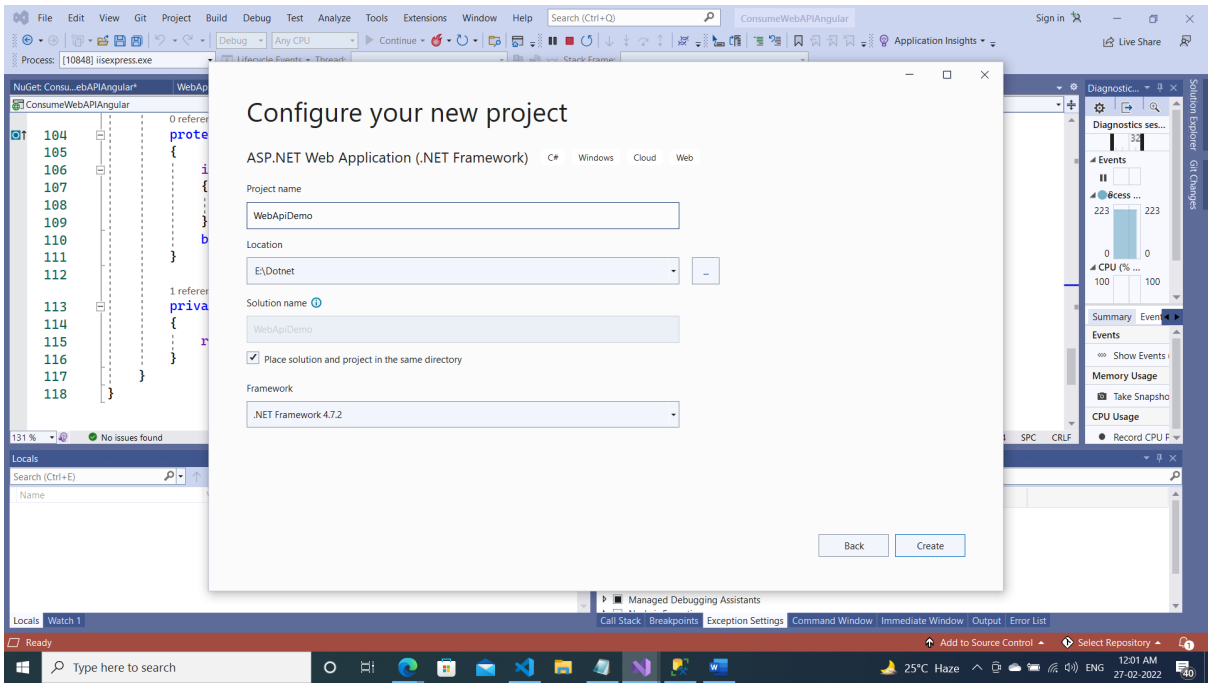
Step 1:

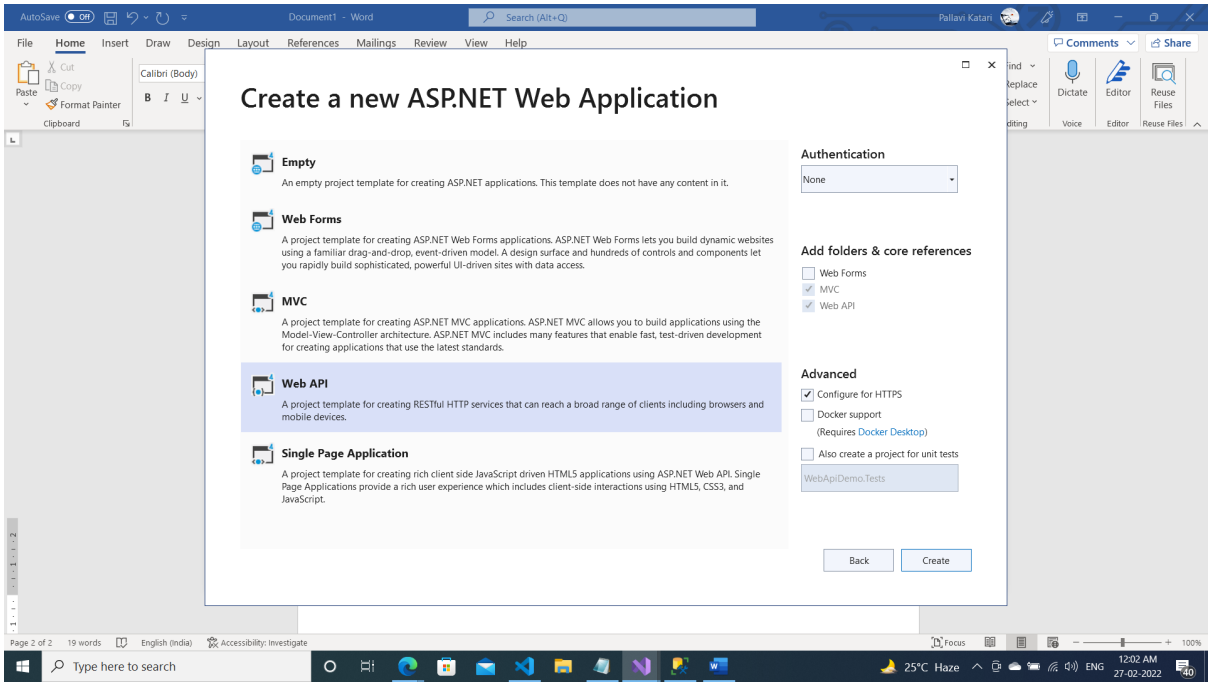
Create a Database and table:



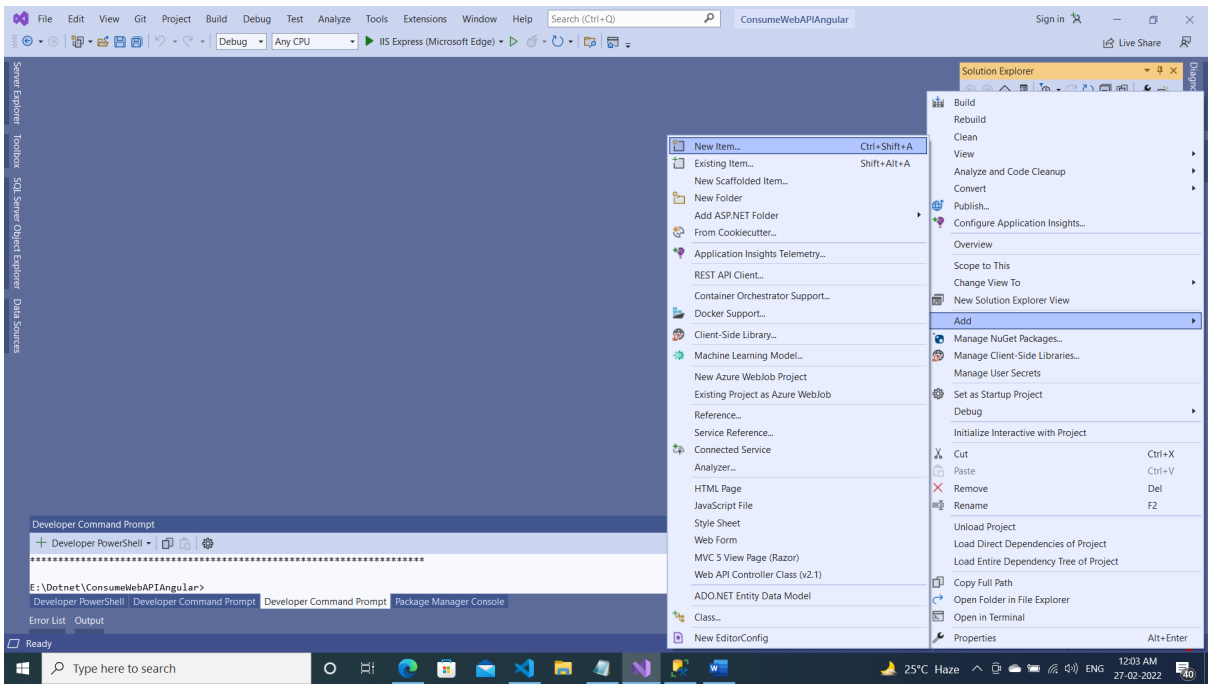
Step 2:

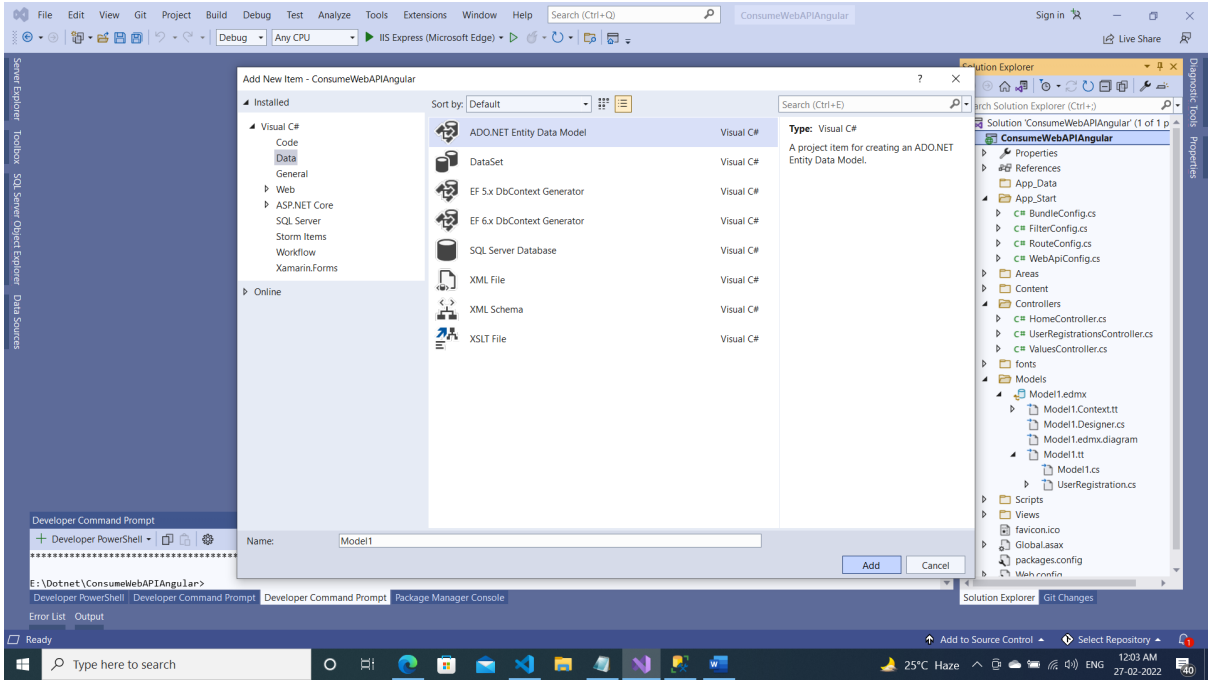
Create a ASP.NET WEB APPLICATION:



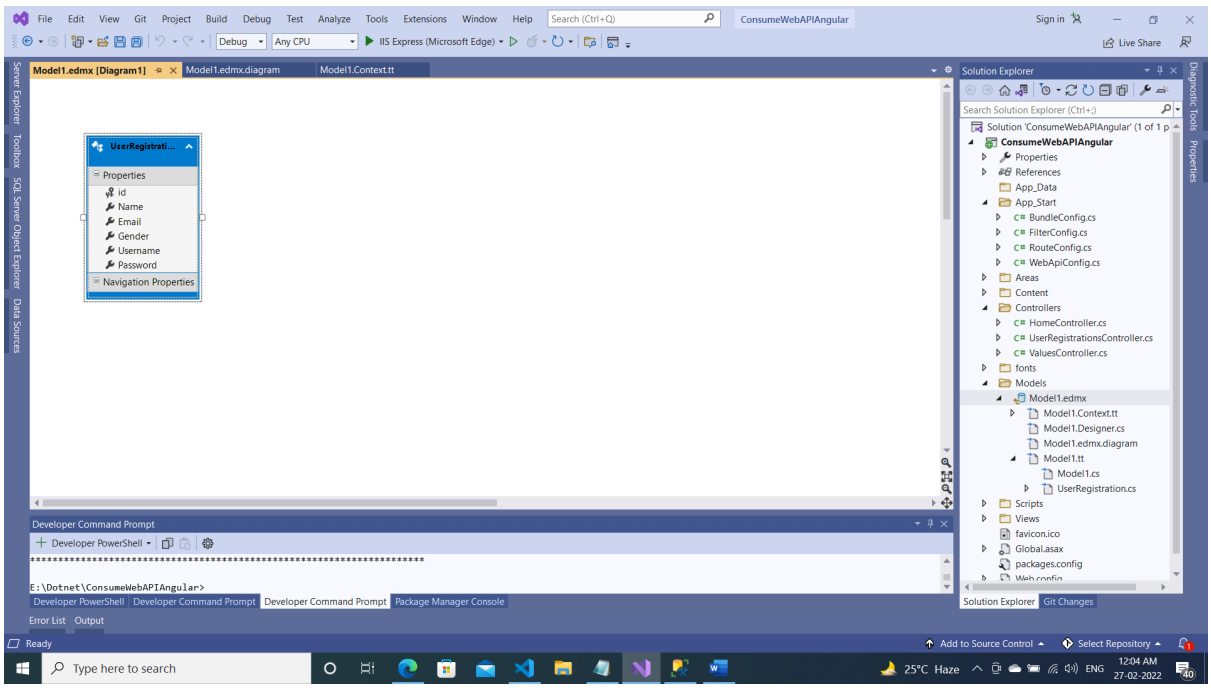

Create a new ASP.NET Web Application
Empty
An empty project template for creating ASP.NET applications. This template does not have any content in it.
Web Forms
A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.
MVC
A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.
Web API
A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.
Single Page Application
A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.
Authentication
None
Add folders & core references
Web Forms
MVC
Web API
Advanced
Configure for HTTPS
Docker support
(Requires Docker Desktop)
Also create a project for unit tests
WebApiDemo.Tests
Back
Create

Step 3:

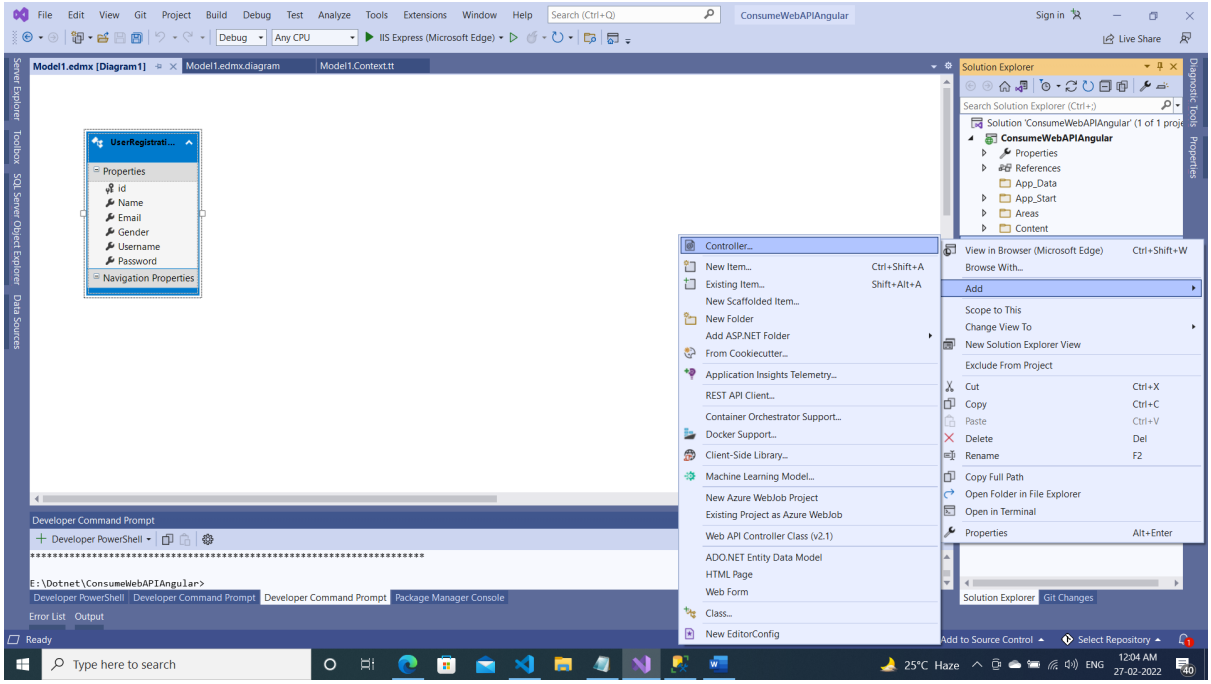

ConsumeWebAPIAngular
Solution Explorer
New Item... Ctrl+Shift+A
Existing Item... Shift+Alt+A
New Scaffolded Item...
New Folder
Add ASP.NET Folder
From Cookiecutter...
Application Insights Telemetry...
REST API Client...
Container Orchestrator Support...
Docker Support...
Client-Side Library...
Machine Learning Model...
New Azure WebJob Project
Existing Project as Azure WebJob
Reference...
Service Reference...
Connected Service
Analyzer...
HTML Page
JavaScript File
Style Sheet
Web Form
MVC 5 View Page (Razor)
Web API Controller Class (v2.1)
ADO.NET Entity Data Model
Class...
New EditorConfig
Build
Rebuild
Clean
View
Analyze and Code Cleanup
Convert
Publish...
Configure Application Insights...
Overview
Scope to This
Change View To
New Solution Explorer View
Add
Manage NuGet Packages...
Manage Client-Side Libraries...
Manage User Secrets
Set as Startup Project
Debug
Initialize Interactive with Project
Cut Ctrl+X
Paste Ctrl+V
Remove Del
Rename F2
Unload Project
Load Direct Dependencies of Project
Load Entire Dependency Tree of Project
Copy Full Path
Open Folder in File Explorer
Open in Terminal
Properties Alt+Enter
Developer Command Prompt
E:\Dotnet\ConsumeWebAPIAngular>

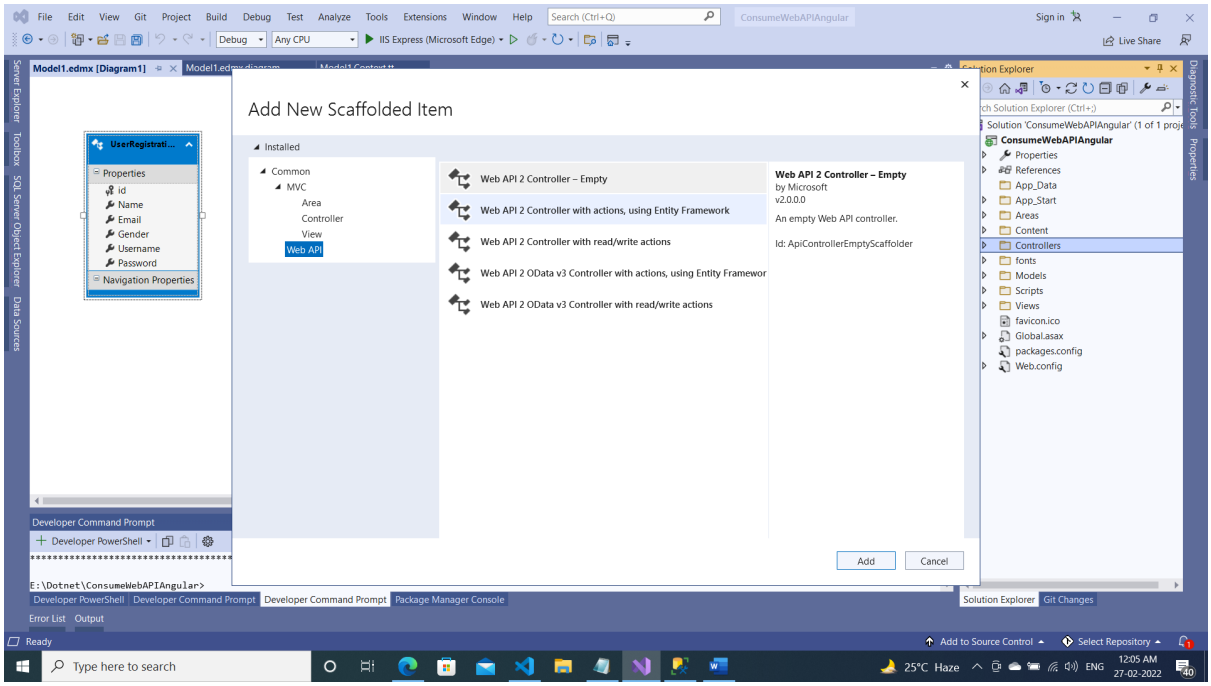

Add New Item - ConsumeWebAPIAngular
Installed
Visual C#
Code
Data
General
Web
ASP.NET Core
SQL Server
Storm Items
Workflow
Xamarin.Forms
Online
Sort by: Default
ADO.NET Entity Data Model
DataSet
EF 5.x DbContext Generator
EF 6.x DbContext Generator
SQL Server Database
XML File
XML Schema
XSLT File
Type: Visual C#
A project item for creating an ADO.NET Entity Data Model.
Name: Model1
Add
Cancel

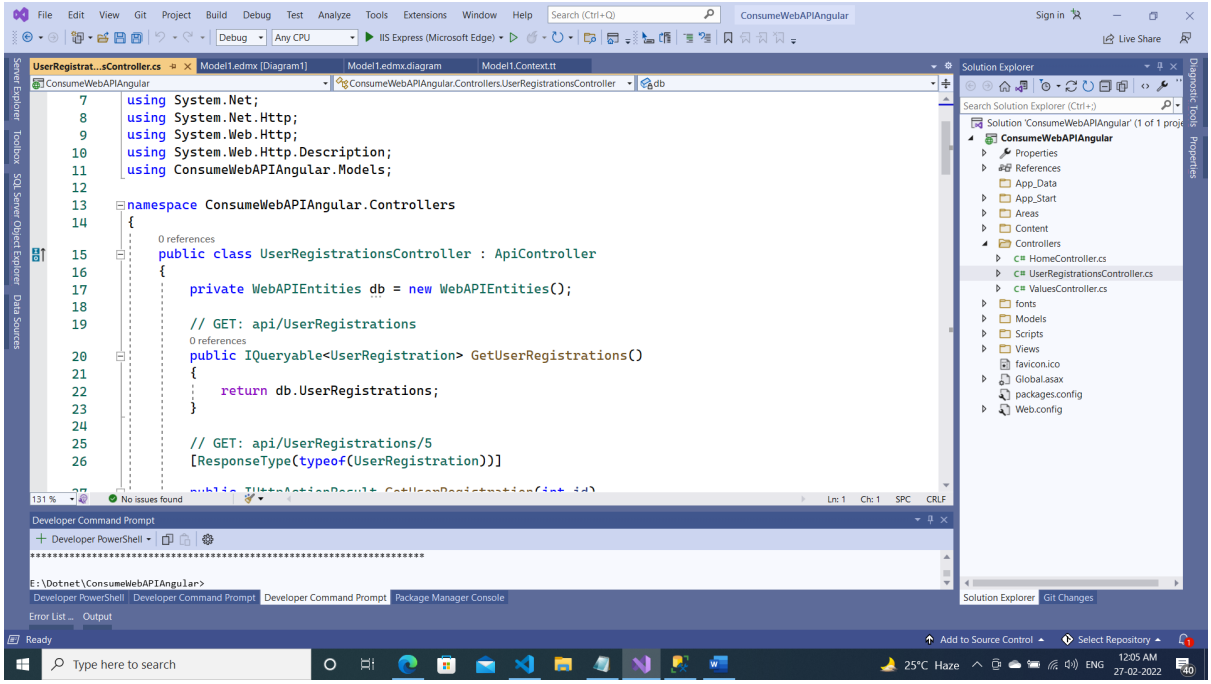
Step 4:


Model1.edmx [Diagram1]
Model1.edmx.diagram
Model1.Context.tt
UserRegistrati...
Properties
id
Name
Email
Gender
Username
Password
Navigation Properties

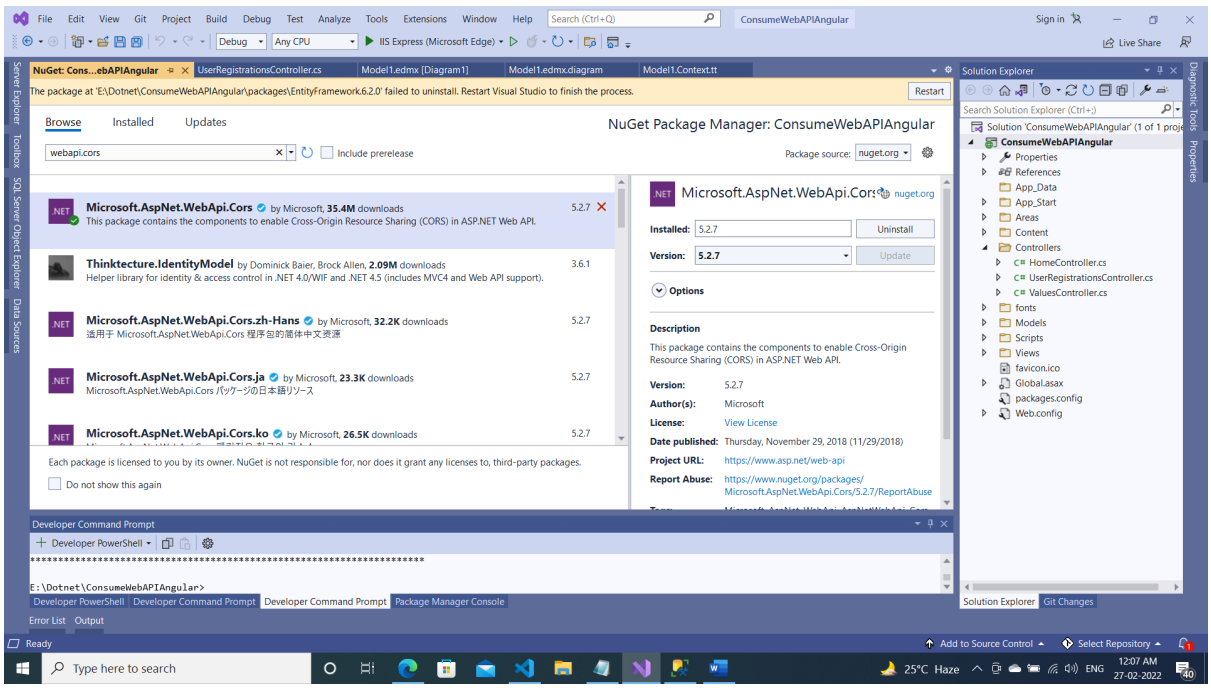
Step 5: Build your application before creating a controller



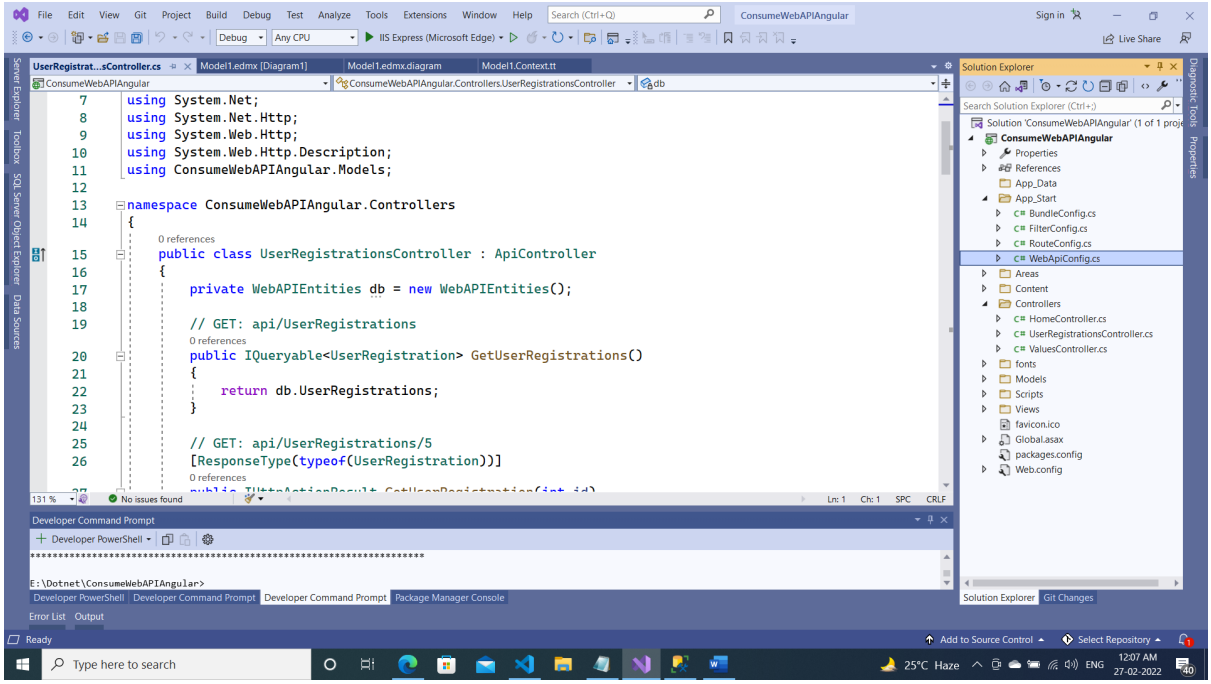




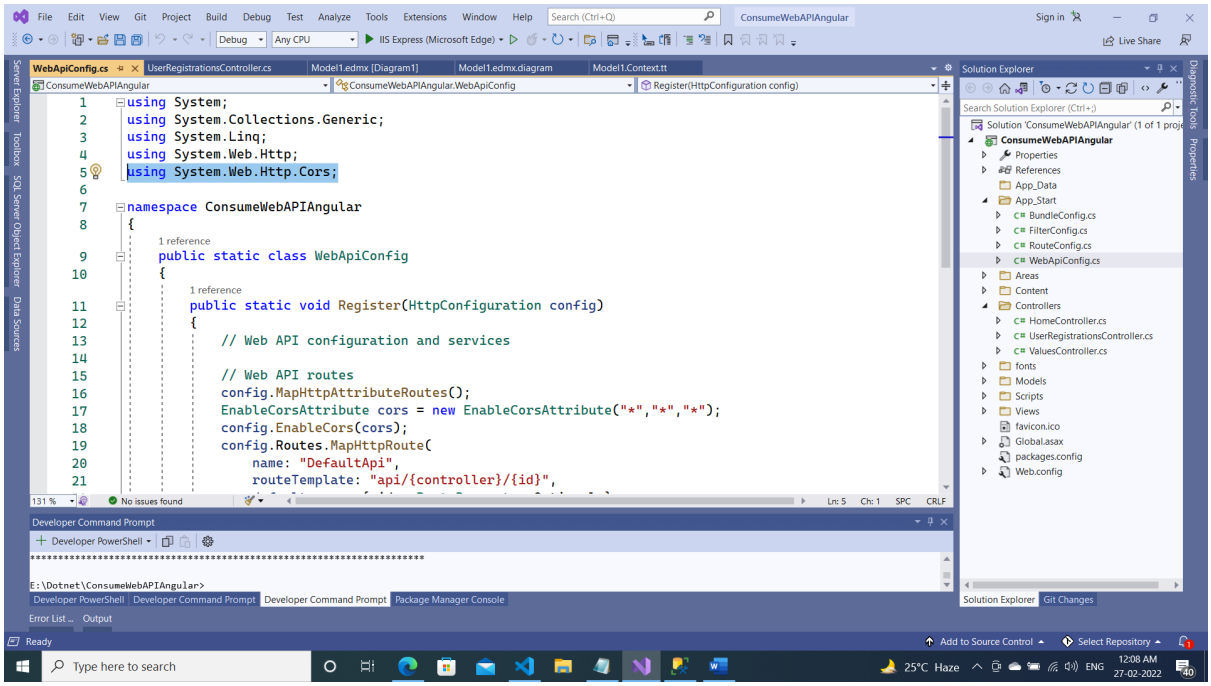
Step 6: Install



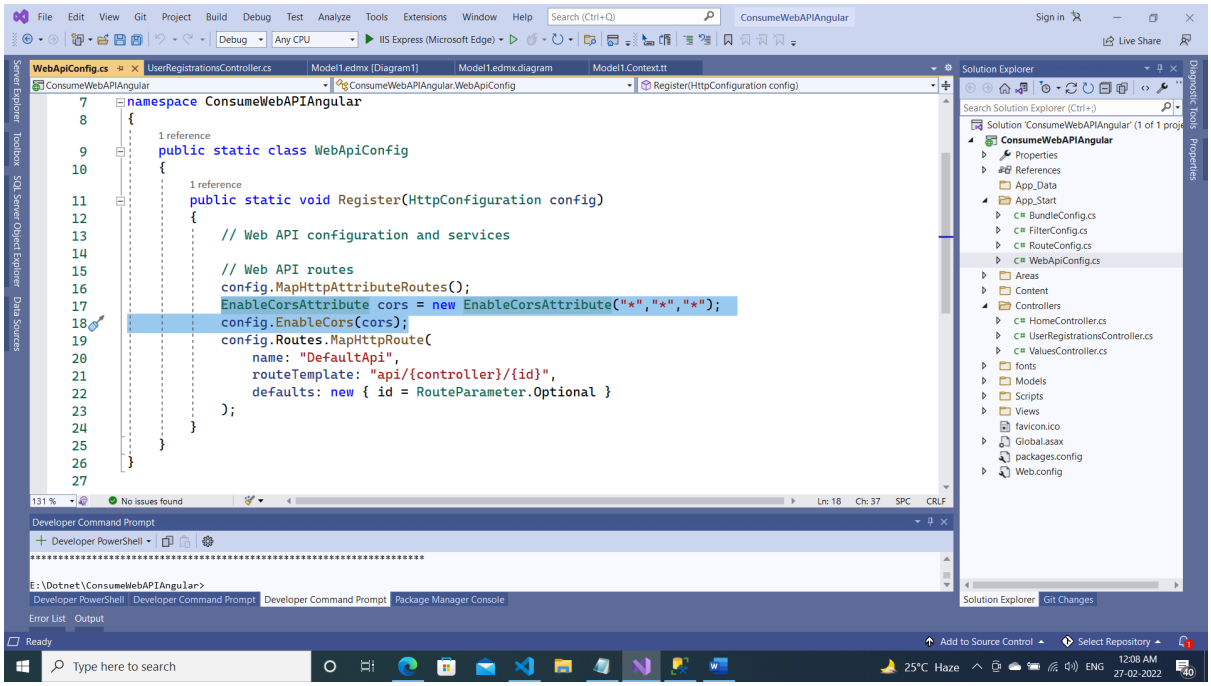
Step 7:



Include the namespace

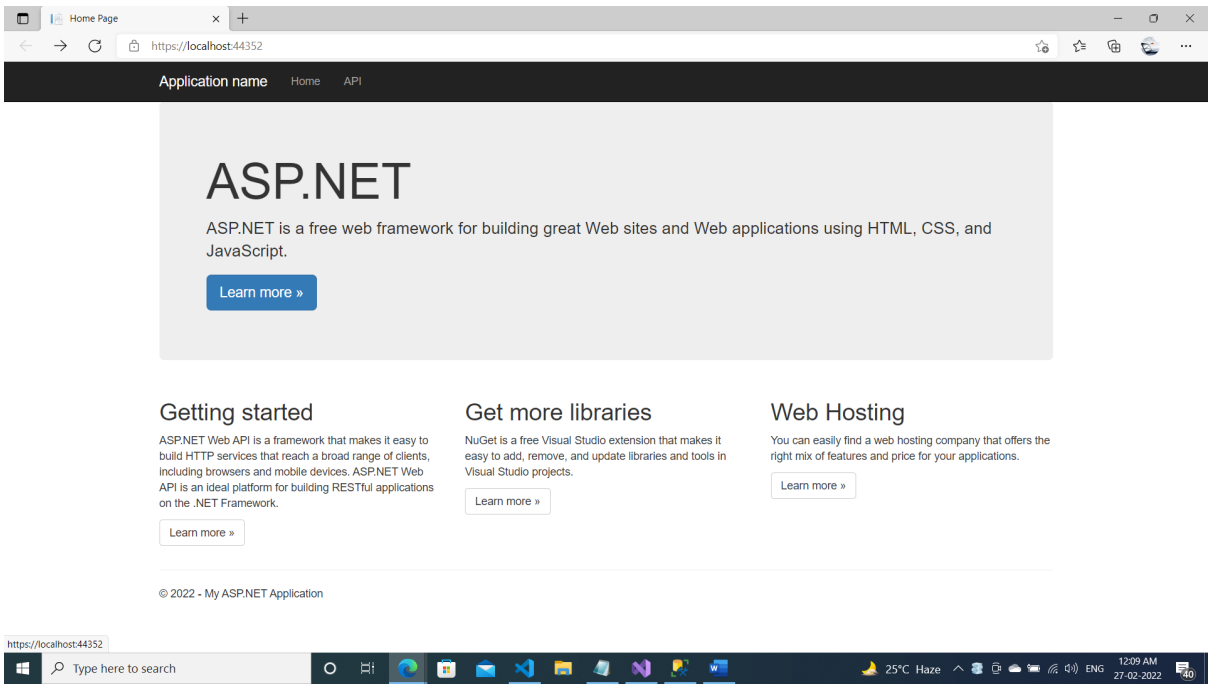


Add the highlighted lines

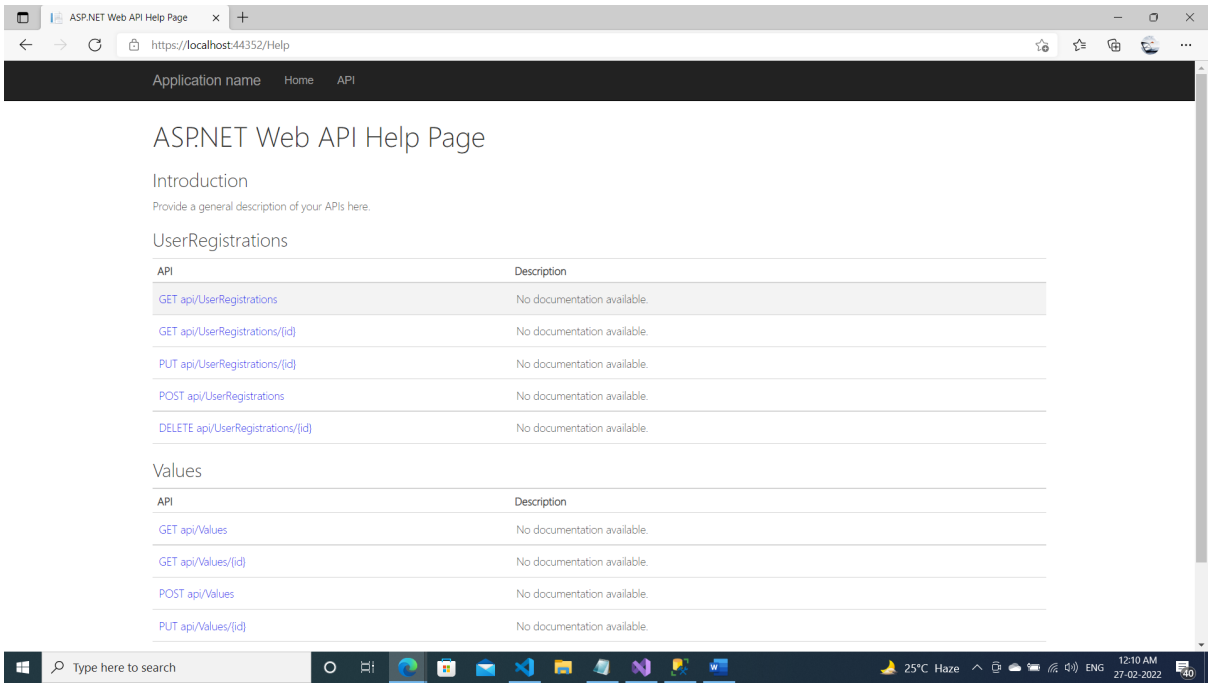


Step 8:

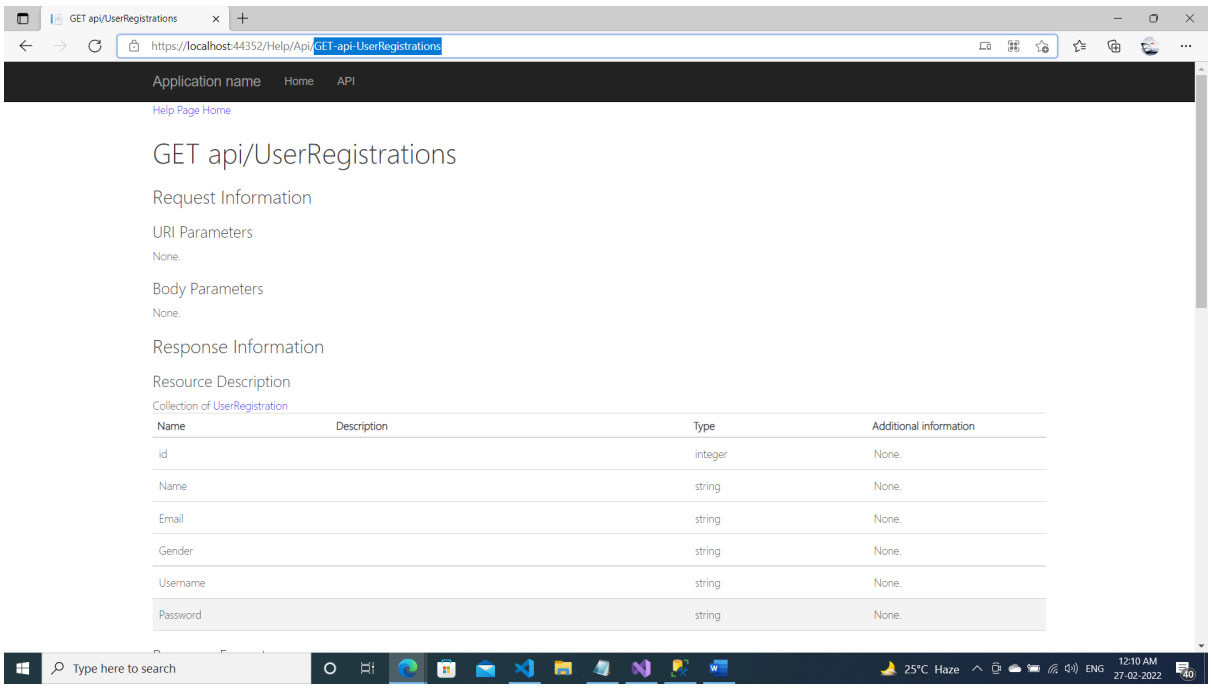
Execute your application



Click API Link



Click GET-api-UserRegistrations



Let the application keep running

Step 9:

Creating Angular App

- ➔ Create a class -> ng g class user
- ➔ Create a service->ng g s userservice
- ➔ Create a component->ng g c UserRegistration

Step 10:

User.ts

```
export class User
{
    //create a class based on the columns created in SQL
    id:number;
    Name:string;
    Gender:string;
    Email:string;
    Username:string;
    Password:string;
}
```

Step 11:

User-service.service.ts

```
//HttpClient API service is used to make communication between front-end web
apps with backend services.
//This communication is done over HTTP protocol.
import { HttpClient, HttpHeaders } from '@angular/common/http';
import { Injectable } from '@angular/core';
//Observable module
import { Observable } from 'rxjs';
//user.ts class
import { User } from './user';

@Injectable({
  providedIn: 'root'
})
export class UserServiceService {
//asp.net web api (the api should be running while consuming from Angular)
  url = 'https://localhost:44352/Api/UserRegistrations';
  //Will invoke UserRegistrationsController->GetUserRegistrations()
  //GET->Read records
      constructor(private http: HttpClient) { }
      getAllUsers(): Observable<User[]> {
        return this.http.get<User[]>(this.url );
```

```
      }
      getEmployeeById(employeeId: string): Observable<User> {
        return this.http.get<User>(this.url + '/GetEmployeeDetailsById/' +
employeeId);
      }
      // Will invoke UserRegistrationsController->PostUserRegistration
      createUser(user: User): Observable<User> {
        const httpOptions = { headers: new HttpHeaders({ 'Content-Type':
'application/json'}) };
        return this.http.post<User>(this.url ,
        user, httpOptions);
      }
      updateEmployee(employee: User): Observable<User> {
        const httpOptions = { headers: new HttpHeaders({ 'Content-Type':
'application/json'}) };
        return this.http.put<User>(this.url + '/UpdateEmployeeDetails/',
        employee, httpOptions);
      }
      deleteEmployeeById(employeeid: string): Observable<number> {
        const httpOptions = { headers: new HttpHeaders({ 'Content-Type':
'application/json'}) };
        return this.http.delete<number>(this.url +
'/DeleteEmployeeDetails?id=' +employeeid,
     httpOptions);
      }

}
```

Step 12:

App.module.ts

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
//HttpClient for API
import { HttpClientModule } from '@angular/common/http';
import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
//import the below modules
import { UserRegistrationComponent } from './user-registration/user-
registration.component';
import { FormsModule } from '@angular/forms';
import { ReactiveFormsModule } from '@angular/forms';
import { LoginComponent } from './login/login.component';

@NgModule({
  declarations: [
```

```
    AppComponent,
    UserRegistrationComponent,
    LoginComponent
  ],
  imports: [
    BrowserModule,
    AppRoutingModule,
    HttpClientModule,
    FormsModule,
    ReactiveFormsModule
  ],
  providers: [],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

Step 13:

User-registration.component.ts

```
import { Component, OnInit } from '@angular/core';
//Observable module to Observables provide support for passing messages
between
//parts of your application. They are used frequently in Angular and are
//the recommended technique for event handling,
//asynchronous programming, and handling multiple values.
import { Observable } from 'rxjs';
//import user.ts
import { User } from '../user';
//import user-service.service.ts
import { UserServiceService } from '../user-service.service';
//import for Form designing in Angular
import { FormGroup,FormControl } from '@angular/forms';
@Component({
  selector: 'app-user-registration',
  templateUrl: './user-registration.component.html',
  styleUrls: ['./user-registration.component.css']
})
export class UserRegistrationComponent implements OnInit
{

  Gender = ['male', 'female'];
  allUsers :Observable<User[]>;
  userForm :FormGroup;
  dataSaved = false;

  userIdUpdate = null;
  massage = null;
  //inheriting UserServiceService from user-service.service.ts
```

```
  constructor(private userservice:UserServiceService)
  {

  }
  //create a function onSubmit
  onSubmit()
  {
    console.log(this.userForm)
    this.dataSaved = false;
    const user = this.userForm.value;
    //Invoking the CreateUser function
    this.CreateUser(user);
    this.userForm.reset();
  }
  //create a function onSubmit CreateUser
  CreateUser(user: User) {
    if (this.userIdUpdate == null) {
      //subscribe() is a method on the Observable type. The Observable type is
a
      //utility that asynchronously or synchronously streams data
      //to a variety of components or services that have subscribed to the
observable.
      this.userservice.createUser(user).subscribe(
        () => {
          this.dataSaved = true;
          this.massage = 'Record saved Successfully';
          this.loadAllUsers();
          this.userIdUpdate = null;
          this.userForm.reset();
        }
      );
    } else {
      user.id = this.userIdUpdate;
      this.userservice.updateEmployee(user).subscribe(() => {
        this.dataSaved = true;
        this.massage = 'Record Updated Successfully';
        this.loadAllUsers();
        this.userIdUpdate = null;
        this.userForm.reset();
      });
    }
  }
  ngOnInit(): void {
    this.loadAllUsers();
    console.log(this.allUsers)
    this.userForm=new FormGroup({
      'Name':new FormControl(null),
      'Email':new FormControl(null),
```

```
      'Gender':new FormControl(null),
      'Username':new FormControl(null),
      'Password':new FormControl(null)
    })
  }
  loadAllUsers()
  {
    this.allUsers=this.userservice.getAllUsers();
  }
}
```

Step 14:

User-registration.component.html

```
<!--Registration Form-->
<div class="container">
    <!--Consuming Web API and displaying in the table (READ OPERATION -> GET)-
->
    <table class="table table-responsive" >
        <thead>
            <tr>
                <th>UserId</th>
                <th>Name</th>
                <th>Email</th>
                <th>Gender</th>
                <th>UserName </th>
            </tr>
        </thead>
        <tbody>
            <tr *ngFor="let u of allUsers |async;">
                <td> {{u.id}}</td>
                <td> {{u.Name}}</td>
                <td> {{u.Email}}</td>
                <td> {{u.Gender}}</td>
                <td> {{u.Username}}</td>
            </tr>
        </tbody>
    </table>
    <!--Creating a user using Web API (Create Operation -> POST)-->
    <div class="container">
        <div class="row">
            <div class="form_bg">
                <!--accessing onSubmit() from user-registration.component.ts--
>
                <form [formGroup]="userForm" (ngSubmit)="onSubmit()">
                     <h2 class="text-center">Registration page</h2>
```

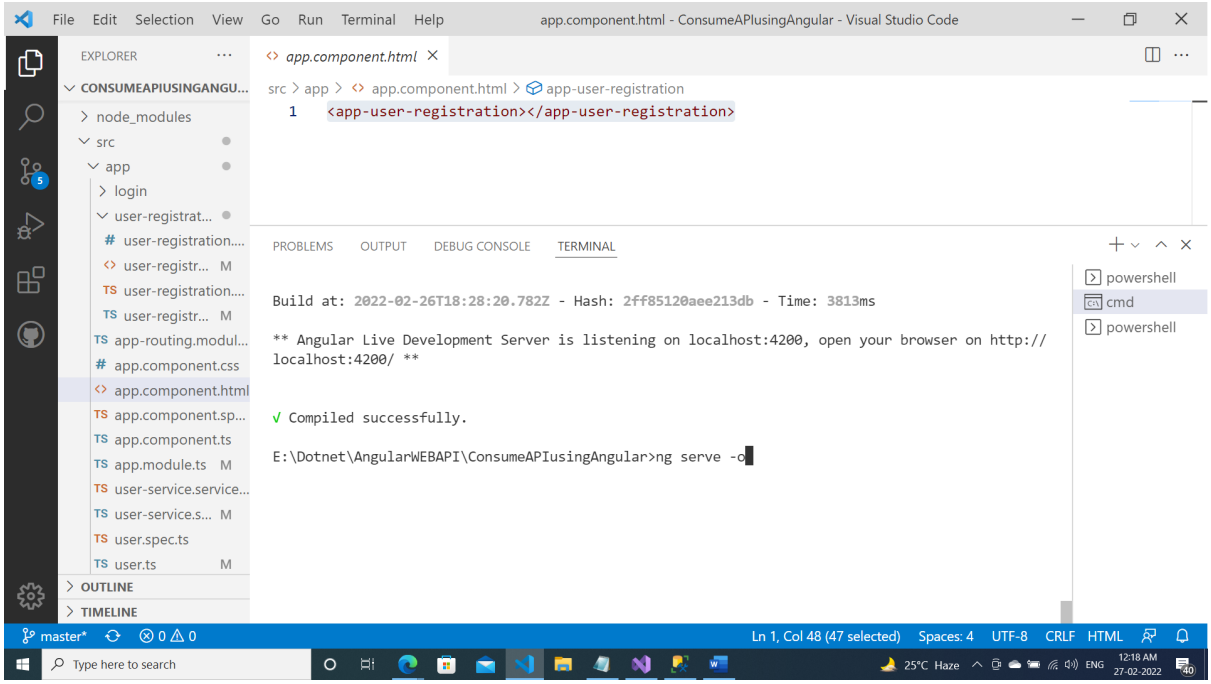
```
                    <br/>
                    <div class="form-group">
                        <label for="name">Name</label>
                        <input type="text" class="form-control"
                        formControlName="Name">
                    </div>
                    <div class="radio" *ngFor="let gender of Gender">
                    <label>
                      <input formControlName="Gender"
                        type="radio"
                        [value]="gender">{{ gender }}
                    </label>
                  </div>
                    <div class="form-group">
                        <label for="Email">Email</label>
                        <input type="text" class="form-control"
                        placeholder="sample@example.com"
formControlName="Email">
                    </div>
                    <div class="form-group">
                        <label for="username">UserName</label>
                        <input type="text" class="form-control" id="username"
                        formControlName="Username">
                       </div>
                       <div class="form-group">
                        <label for="password">Password</label>
                        <input type="text" class="form-control" id="password"
                        formControlName="Password">
                       </div>
                    <br/>
                    <div class="align-center">
                        <button type="submit"
                        class="btn btn-primary"
id="register">Register</button>
                    </div>
                </form>
            </div>
        </div>
```

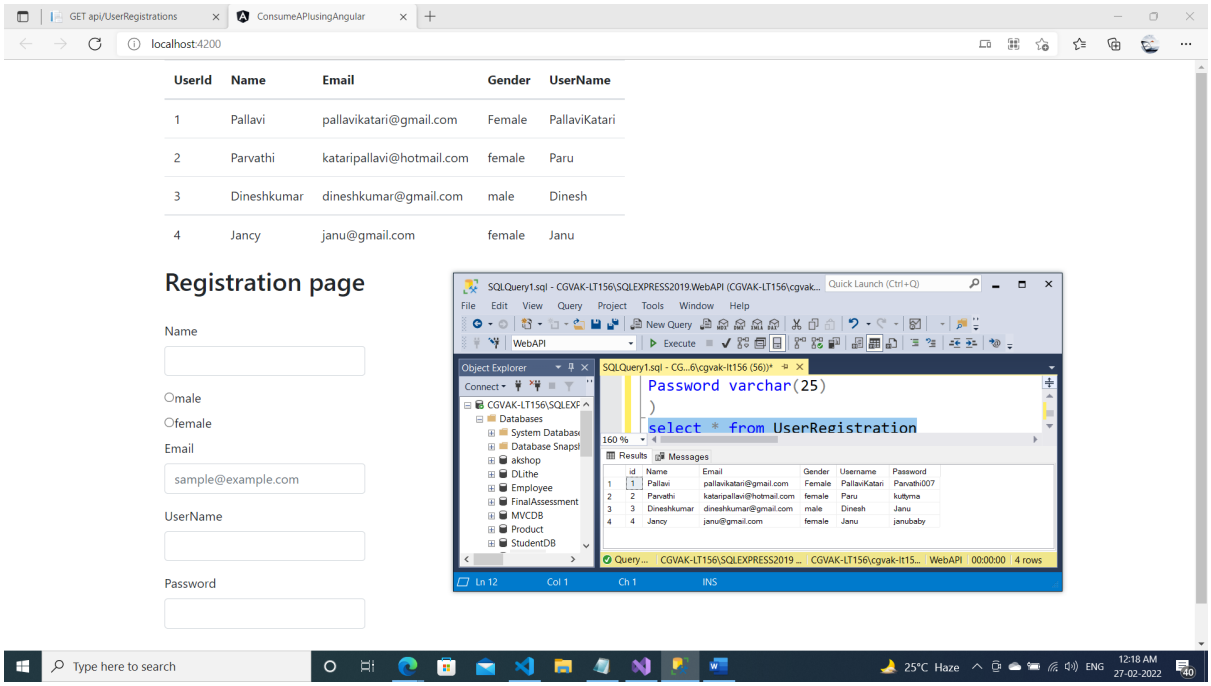
Step 15:

App.component.html

```
<app-user-registration></app-user-registration>
```

Step:16


app.component.html - ConsumeAPIusingAngular - Visual Studio Code
app.component.html
<app-user-registration></app-user-registration>
Build at: 2022-02-26T18:28:20.782Z - Hash: 2ff85120aee213db - Time: 3813ms
** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **
√ Compiled successfully.
E:\Dotnet\AngularWEBAPI\ConsumeAPIusingAngular>ng serve -o


localhost:4200
UserId Name Email Gender UserName
1 Pallavi pallavikatari@gmail.com Female PallaviKatari
2 Parvathi kataripallavi@hotmail.com female Paru
3 Dineshkumar dineshkumar@gmail.com male Dinesh
4 Jancy janu@gmail.com female Janu
Registration page
Name
male
female
Email
sample@example.com
UserName
Password
Password varchar(25)
select * from UserRegistration

GET api/UserRegistrations | ConsumeAPIusingAngular

localhost:4200

1	Pallavi	pallavikatari@gmail.com	Female	PallaviKatari
2	Parvathi	kataripallavi@hotmail.com	female	Paru
3	Dineshkumar	dineshkumar@gmail.com	male	Dinesh
4	Jancy	janu@gmail.com	female	Janu

Registration page

Name

Nancy

○male

◉female

Email

nancy@gmail.com

UserName

Nancy

Password

Nani

Register

Type here to search | 25°C Haze | ENG | 12:19 AM 27-02-2022

GET api/UserRegistrations | ConsumeAPIusingAngular

localhost:4200

UserId	Name	Email	Gender	UserName
1	Pallavi	pallavikatari@gmail.com	Female	PallaviKatari
2	Parvathi	kataripallavi@hotmail.com	female	Paru
3	Dineshkumar	dineshkumar@gmail.com	male	Dinesh
4	Jancy	janu@gmail.com	female	Janu
5	Nancy	nancy@gmail.com	female	Nancy

Registration page

Name

○male

○female

Email

sample@example.com

UserName

Password

SQLQuery1.sql - CGVAK-LT156\SQLEXPRESS2019.WebAPI (CGVAK-LT156\cgvak...

```
Password varchar(25)
)
select * from UserRegistration
```

	id	Name	Email	Gender	Username	Password
1	1	Pallavi	pallavikatari@gmail.com	Female	PallaviKatari	Parvathi007
2	2	Parvathi	kataripallavi@hotmail.com	female	Paru	kuttyma
3	3	Dineshkumar	dineshkumar@gmail.com	male	Dinesh	Janu
4	4	Jancy	janu@gmail.com	female	Janu	janubaby
5	5	Nancy	nancy@gmail.com	female	Nancy	Nani

Type here to search | 25°C Haze | ENG | 12:19 AM 27-02-2022