WebAPI Lab: How to Build an API with Angular in Windows

A step-by-step guide to install Angular and create a simple web service

# Introduction

In this lab, you will learn how to install Angular on a Windows PC and create an API that can perform CRUD (create, read, update, delete) operations on a data source. Angular is a popular framework for building web applications that use TypeScript, a superset of JavaScript. An API, or application programming interface, is a set of rules and protocols that allow different software components to communicate and exchange data. By creating an API, you can expose your data and functionality to other applications, such as web browsers, mobile apps, or desktop software.

# Prerequisites

Before you start this lab, you will need the following:

* A Windows PC with an internet connection
* A text editor, such as Visual Studio Code, Notepad++, or Sublime Text
* A command-line tool, such as Windows PowerShell, Command Prompt, or Git Bash
* Node.js, a runtime environment that allows you to run JavaScript code outside of a browser. You can download it from undefined
* npm, a package manager that comes with Node.js and allows you to install and manage dependencies for your projects. You can check if you have npm installed by running npm -v in your command-line tool
* Angular CLI, a command-line interface that helps you create and manage Angular projects. You can install it by running

npm install -g @angular/cli

ng analytics disable –global

ng version

A screenshot of a computer

Description automatically generated

# Steps

Follow these steps to install Angular and create an API:

1. Create a new Angular project by running ng new webapi-lab in your command-line tool. This will create a folder called webapi-lab with the necessary files and folders for an Angular project. You can choose the default options for the prompts, or customize them according to your preferences
2. Navigate to the webapi-lab folder by running cd webapi-lab in your command-line tool
3. Create the scaffolding for your Angular App that will call your API

ng **new** myapi

A screenshot of a computer program

Description automatically generated

1. Start VCS:

**code** .

It will want to install the Angular VSC Service, click Install

A screenshot of a computer error

Description automatically generated

1. Run the template code: CTRL + F5, or Run | Start Without Debugging

A screenshot of a computer

Description automatically generated

1. Make Changes to call our DotNet API, now we will open up our Chat Window and prompt CoPilot

A black and white square and rectangle shaped speech bubbles

Description automatically generated

Modify **the** Angular scaffolding to add API Calls **for** **a** Sales Report. Display **the** API response **as** **a** table **for** **the** sales report. Call localhost:5023/Reports/getSales to get **the** JSON data. Make another API Call to display **a** report **of** Invoices **by** Country. Display this **as** **a** Table. Call localhost:5023/Reports/getInvoices to get **the** JSON response from **the** API. Show all necessary files to change **with** **the** code to implement **the** reports. Be sure to generate CSS **for** both reports. Get rid **of** **the** default app **and** display **the** Invoice **and** Sales reports **on** **the** **homepage**.

GitHub Copilot: To implement the reports, you'll need to modify several files in your Angular project. Here's a step-by-step guide:

1. \*\*Create a service to handle API calls\*\*

In `src/app/services/report.service.ts`:

**import** { HttpClient } from '@angular/common/http';  
**import** { Injectable } from '@angular/core';  
**import** { Observable } from 'rxjs';  
  
@Injectable({  
 providedIn  
  
:  
  
 'root'  
})  
export **class** **ReportService** {  
 **private** BASE\_URL = 'http://localhost:5023/Reports';  
  
 **constructor**(**private** http: HttpClient) { }  
  
 getSalesReport(): Observable<any> {  
 **return** **this**.http.**get**(`${**this**.BASE\_URL}/getSales`);  
 }  
  
 getInvoicesReport(): Observable<any> {  
 **return** **this**.http.**get**(`${**this**.BASE\_URL}/getInvoices`);  
 }  
}

2. \*\*Create a component to display the sales report\*\*

In `src/app/components/sales-report/sales-report.component.ts`:

**import** { Component, OnInit } **from** '@angular/core';  
**import** { ReportService } **from** '../../services/report.service';  
  
@Component({  
 selector: 'app-sales-report',  
 templateUrl: './sales-report.component.html',  
 styleUrls: ['./sales-report.component.css']  
})  
**export** **class** SalesReportComponent **implements** OnInit {  
 salesReport: any;  
  
 **constructor**(**private** reportService: ReportService) { }  
  
 ngOnInit(): void {  
 **this**.reportService.getSalesReport().subscribe(data => {  
 **this**.salesReport = data;  
 });  
 }  
}

In `src/app/components/sales-report/sales-report.component.html`:

<**table**>  
 <**tr**>  
 <**th**>Column1</**th**>  
 <**th**>Column2</**th**>  
 <!-- Add more columns as needed -->  
 </**tr**>  
 <**tr** \*ngFor="let item of salesReport">  
 <**td**>{{item.BillingCountry}}</**td**>  
 <**td**>{{item.TotalSales}}</**td**>  
 <!-- Add more data cells as needed -->  
 </**tr**>  
</**table**>

In `src/app/components/sales-report/sales-report.component.css`:

**table** {  
 **width**: 100%;  
 **border-collapse**: collapse;  
}  
  
**th**, **td** {  
 **text-align**: center;  
 **padding**: 8px;  
}  
  
**tr**:nth-child(even) {  
 **background-color**: #f2f2f2;  
}

3. \*\*Create a component to display the invoices report\*\*

In `src/app/components/invoices-report/invoices-report.component.ts`:

**import** { Component, OnInit } **from** '@angular/core';  
**import** { ReportService } **from** '../../services/report.service';  
  
@Component({  
 selector: 'app-invoices-report',  
 templateUrl: './invoices-report.component.html',  
 styleUrls: ['./invoices-report.component.css']  
})  
**export** **class** InvoicesReportComponent **implements** OnInit {  
 invoicesReport: any;  
  
 **constructor**(**private** reportService: ReportService) { }  
  
 ngOnInit(): void {  
 **this**.reportService.getInvoicesReport().subscribe(data => {  
 **this**.invoicesReport = data;  
 });  
 }  
}

In `src/app/components/invoices-report/invoices-report.component.html`:

<**table**>  
 <**tr**>  
 <**th**>Country</**th**>  
 <**th**>Count of Invoices</**th**>  
 </**tr**>  
 <**tr** \*ngFor="let item of invoicesReport">  
 <**td**>{{item.BillingCountry}}</**td**>  
 <**td**>{{item.InvoiceCount}}</**td**>  
 </**tr**>  
</**table**>

In `src/app/components/invoices-report/invoices-report.component.css`:

**table** {  
 **width**: 100%;  
 **border-collapse**: collapse;  
}  
  
**th**, **td** {  
 **text-align**: center;  
 **padding**: 8px;  
}  
  
**tr**:nth-child(even) {  
 **background-color**: #f2f2f2;  
}

4. \*\*Update AppModule\*\*

In `src/app/app.module.ts`, import the new components and the service, and add them to the appropriate sections:

**import** { HttpClientModule } **from** '@angular/common/http';  
**import** { NgModule } **from** '@angular/core';  
**import** { BrowserModule } **from** '@angular/platform-browser';  
  
**import** { AppComponent } **from** './app.component';  
**import** { SalesReportComponent } **from** './components/sales-report/sales-report.component';  
**import** { InvoicesReportComponent } **from** './components/invoices-report/invoices-report.component';  
**import** { ReportService } **from** './services/report.service';  
  
@NgModule({  
 declarations: [  
 AppComponent,  
 SalesReportComponent,  
 InvoicesReportComponent  
 ],  
 imports: [  
 BrowserModule,  
 HttpClientModule  
 ],  
 providers: [ReportService],  
 bootstrap: [AppComponent]  
})  
**export** **class** **AppModule** { }

5. \*\*Update AppComponent\*\*

In `src/app/app.component.ts`, update the template to display the reports:

import { Component } from '@angular/core';  
  
@Component({  
 selector: 'app-root',  
 template: `  
 <app-sales-report></app-sales-report>  
 <app-invoices-report></app-invoices-report>  
 `,  
 styleUrls: ['./app.component.css']  
})  
export **class** AppComponent { }

Please replace `Column1`, `Column2`, etc. with actual property names from your API response.