**Mục Lục**

[1. Tạo project Net 5.0 và Publish 2](#_Toc97457494)

[2. Cài đặt TypeScript 4](#_Toc97457495)

[3. Tạo Project TypeScript và biên dịch từ .ts sang .js để sử dụng vào web 6](#_Toc97457496)

[4. Chỉnh sửa project 12](#_Toc97457497)

[4.1. index.html là trang chủ 12](#_Toc97457498)

[4.2. init\_variables.ts (tsc = init\_variables.js) file chứa các biến 12](#_Toc97457499)

[4.3. map\_variables.ts (tsc = map\_variables.js) file chứa các biến map, view, layers và basemap… 12](#_Toc97457500)

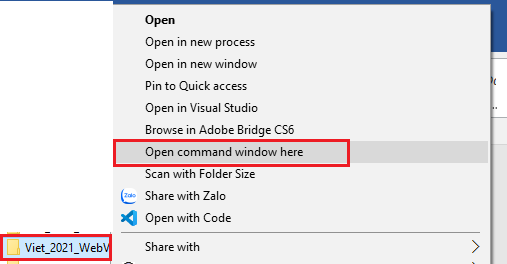
[4.4. map\_layer.ts và toggle\_layer.ts(tsc = map\_layer.js và toggle\_layer.js) file chứa hàm đọc các layer hiển thị hoặc ẩn đi 13](#_Toc97457501)

[4.5. main.ts (tsc = main.js) file add các lớp dữ liệu vào bản đồ cũng như các nút lệnh vào view 14](#_Toc97457502)

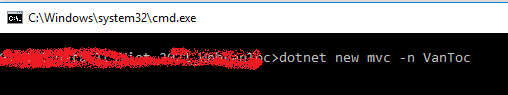
# 1. Tạo project Net 5.0 và Publish

- Bước 1 cài đặt dotnet-sdk-5.0.301-win-x64

- Bước 2 vào thư mục chứa project gọi lệnh cmd



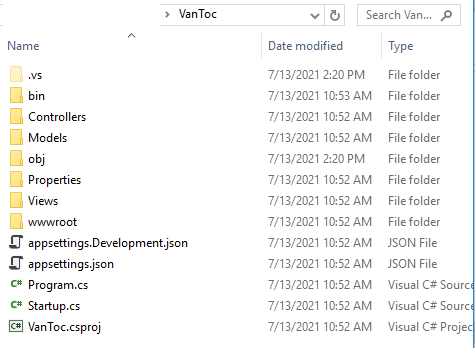
- Bước 3 cmd hiển thị lên gõ lệnh: dotnet new mvc -n VanToc



+ Có thể hỏi lệnh bằng cách dotnet --help

+ Để hỏi chi tiết lệnh tiếp gõ tiếp ví dụ dotnet new --help

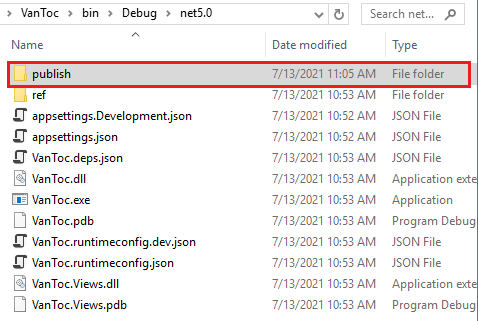
+ Sau khi tạo mới project với tên VanToc thì trong thư mục sẽ sinh ra 1 project



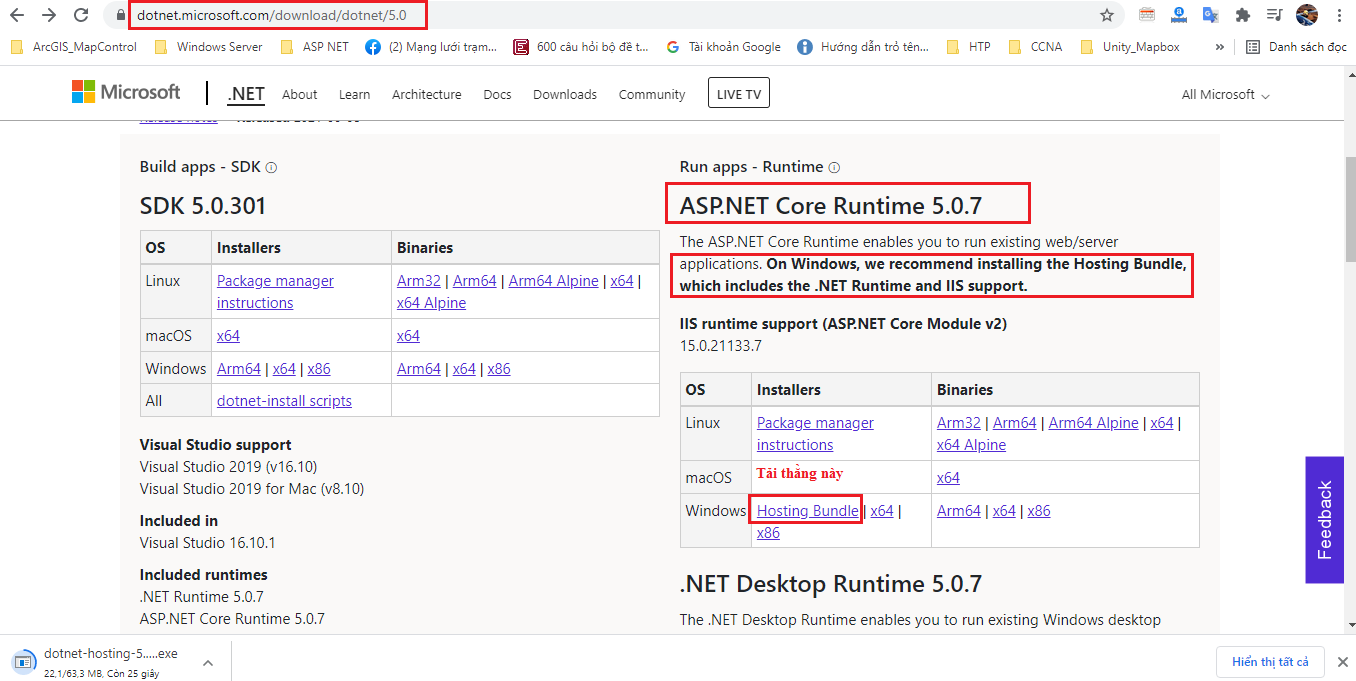
- Để publish đẩy lên server thì gõ lệnh sau: dotnet publish (còn trên local test gõ dotnet run)



+ Kết quả sẽ sinh ra thư mục publish trong project (VanToc\bin\Debug\net5.0), copy thư mục này lên server



- Trên server cần cài dotnet-hosting-5.0.7-win

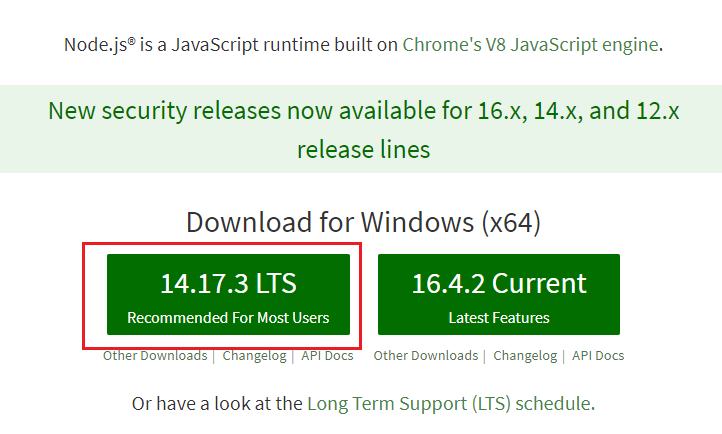


# 2. Cài đặt TypeScript

- Mục đích dùng để hỗ trợ code ngôn ngữ hướng đối tượng .ts sau đó build ra file .js (JavaScript) để sử dụng vào trong lập trình web, nếu viết trực tiếp ra .js luôn thì không cần cái này.

*- Bước 1 cài NodeJS*

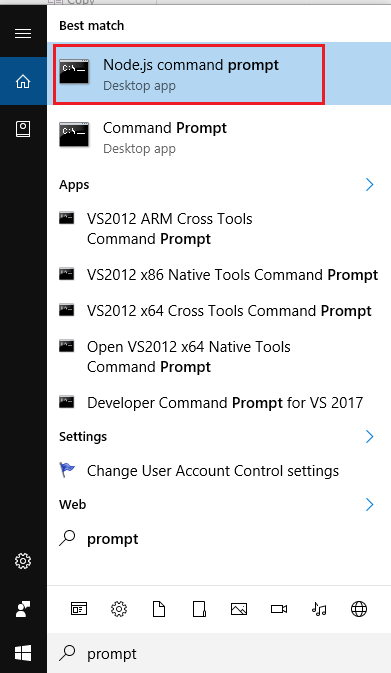
+ Truy cập vào địa chỉ [nodejs.org](http://nodejs.org/) sẽ xuất hiện một trang web với giao diện như sau:



+ Bấm tải bản 14.17.3 LTS, có thể thời điểm vào phiên bản sẽ khác lưu ý tải bản LTS

+ Tiến hành cài bản mới tải về (để mặc định)

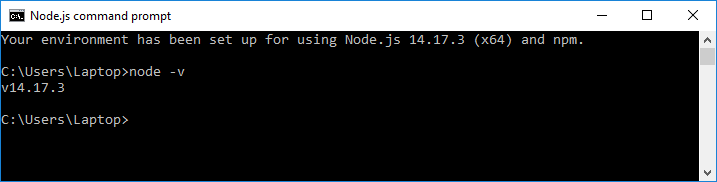
+ Để chắc chắn là bạn đã cài đặt thành công thì bạn mở chương trình Node.js Command Prompt lên bằng cách vào Start gõ search từ "prompt".



Bạn click vào biểu tượng tìm kiếm được sẽ xuất hiện một màn hình cmd màu đen, bạn nhập vào đoạn code sau:

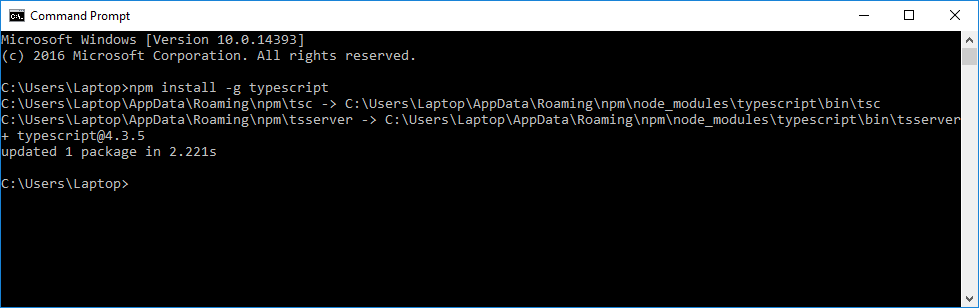
|  |  |
| --- | --- |
|  | node -v |

Dòng code này dùng để kiểm tra Version của NodeJS mà bạn đã cài đặt, vì vậy nếu có kết quả trả về tức là bạn đã cài đặt thành công.



*- Bước 2: Cài đặt TypeScript*

+ Vào Run gõ cmd và nhập lệnh: npm install -g typescript

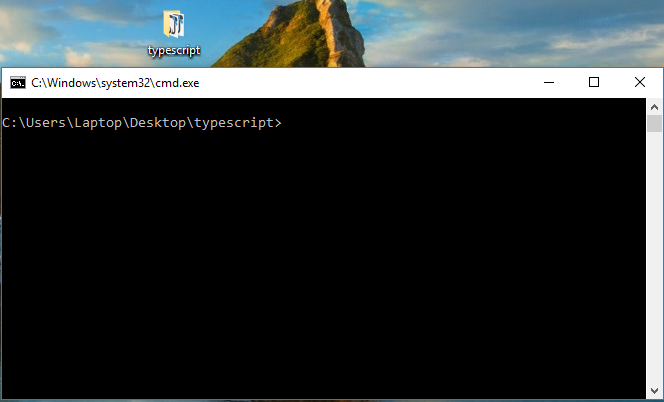


# 3. Tạo Project TypeScript và biên dịch từ .ts sang .js để sử dụng vào web

*a. Tạo Project TypeScript*

- Bước 1 tạo 1 folder là nơi chứa project, sau đó cmd đến thư mục này

+ Ví dụ đặt 1 folder “typescript” ở ngoài màn hình desktop



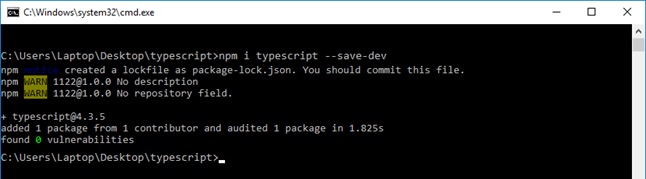
- Bước 2 gõ lệnh: *npm init -y* trên cmd đang mở ở bước 1, gõ lệnh này sẽ tạo ra file package.json trong thư mục



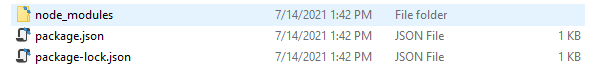
* - Bước 3 gõ lệnh: Cài thư viện (package) typescript bằng lệnh: npm i typescript --save-dev

ở đây mình dùng option --save-dev vì package typescript chỉ cần thiết cho môi trường dev. Sau khi build project thành các file .js thì package typescript không cần thiết

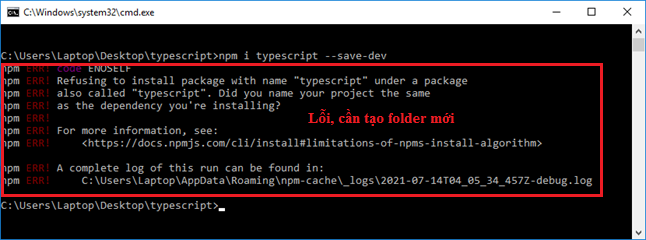
+ Nếu thành công thì hiển thị như này



+ Và có folder node\_modules được tạo ra chứa các thư viện của typescript



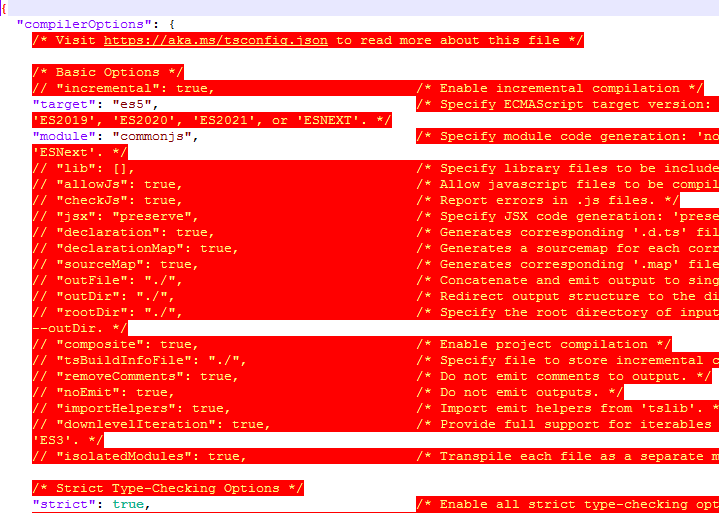
+ Nếu thư mục đã tạo 1 lần rồi thì sẽ báo lỗi như dưới, thì cần phải tạo ở folder khác



* - Bước 4 gõ lệnh: Sau khi cài xong package typescript thì ta có thể khởi tạo project TypeScript bằng lệnh: **npx tsc --init**

Lệnh này sẽ tạo ra file tsconfig.json, chứa cấu hình của project TypeScript như: file js được build ra chỗ nào? những file nào sẽ được build…

+ Cấu hình mặc định file tsconfig.json được tạo ra:



+ Có thể sửa thành như sau:

{

"compilerOptions": {

"target": "es6",

"module": "commonjs",

"strict": true,

"outDir": "dist",

"sourceMap": true

}

}

Ý nghĩa của các field trong file tsconfig.json thì các bạn có thể đọc ở comment trong file sau khi được sinh ra ở trên. Ở đây mình giải thích lại một số field mình dùng:

* "target": "es6" các file .js được generate từ file .ts sẽ được format theo chuẩn es6
* "outDir": "dist" các file .js được sinh ra ở folder dist

+ Hoặc sửa thành như sau: (Mẫu giống project web bản đồ quân đội)

{

"compilerOptions": {

"outDir": "js",

"watch": true,

"removeComments": true,

"module": "amd",

"noImplicitAny": true,

"esModuleInterop": true,

"sourceMap": true,

"jsx": "react",

"jsxFactory": "tsx",

"lib": ["es6", "dom", "es2017"],

"target": "es6",

"experimentalDecorators": true,

"preserveConstEnums": true,

"suppressImplicitAnyIndexErrors": true,

"importHelpers": true,

"moduleResolution": "node",

// "noUnusedLocals": true

},

"include": [

"./app/\*"

],

"exclude": [

"css",

"fonts",

"js",

"img",

"node\_modules"

]

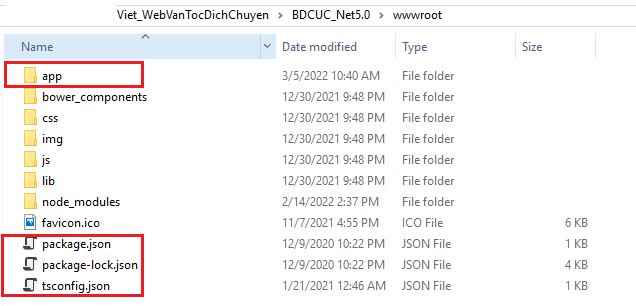
}

"outDir": "js" là các file được sinh ra có thể sửa đường dẫn sâu hơn dạng wwwroot/js

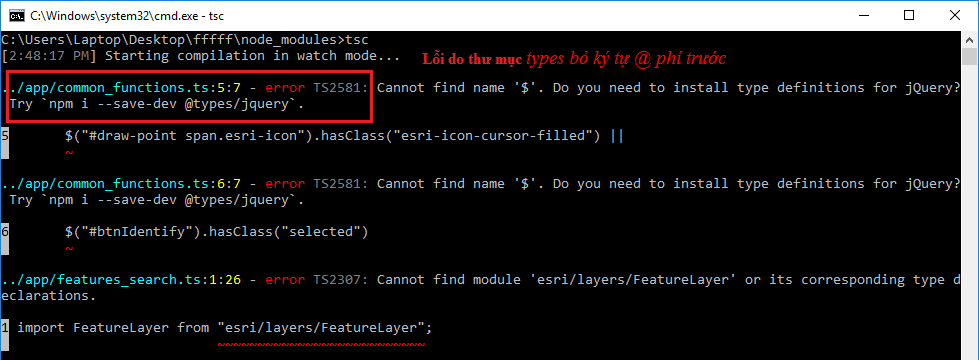
"./app/\*": các file đầu vào .ts ở thư mục app

+ Lưu ý khi tạo 1 project mới mà có mẫu project cũ rồi thì có thể copy các file cần thiết như: package.json, package-lock.json, tsconfig.json và folder node\_modules và các folder khác cần thiết từ project cũ sang project mới.

//Trong projcet NetCore gồm các folder: app, file: package.json, package-lock.json, tsconfig.json



*Lưu ý thư mục node\_modules/@types phải có @phía trước và trong này là cấu hình liên kết các thư viện của esri, nếu không sẽ báo lỗi khi build file .ts*

**

*b. Biên dịch từ .ts sang .js để sử dụng vào web*

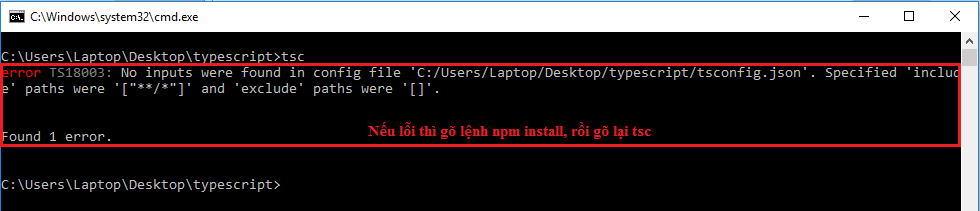
- Bước 1 cmd đến thư mục mà chứa file tsconfig.json, ở project Netcore là thư mục wwwroot

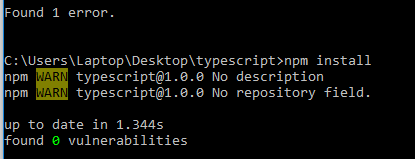
- Bước 2 biên dịch:

+ Biên dịch 1 file thì sử dụng lệnh: tsc tenFile.ts

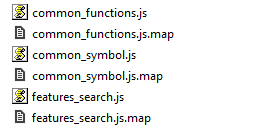
tenFile.ts là file typescript viết, và file .js được sinh ra ở cùng với thư mục này luôn, cái này thì ít dùng (có thể tham khảo <https://freetuts.net/cai-dat-typescript-745.html>)

+ Biên dịch toàn bộ file .ts trong thư mục đích ở trong project. Trong cmd gõ lệnh **tsc**, nếu gõ lệnh này lỗi thì gõ lệnh npm install trước đó

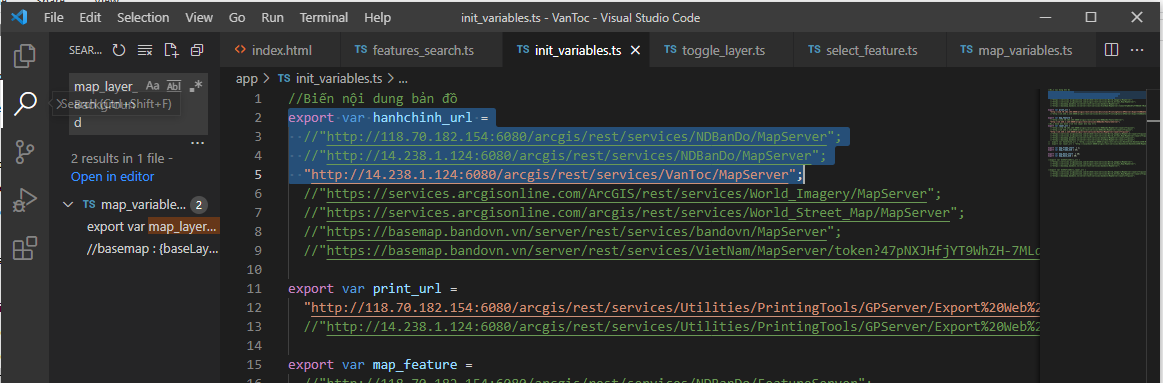




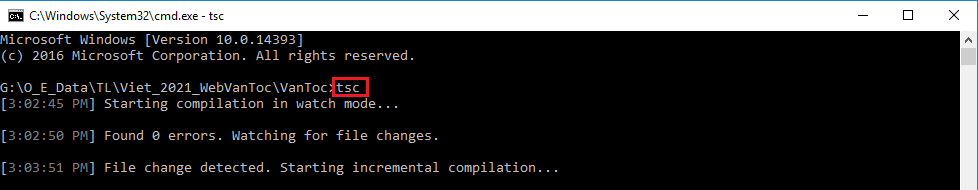
Sau đó gõ lại lệnh tsc, các file .js sẽ được tạo ra ở folder đích outDir



# 4. Chỉnh sửa project



Code trên Visual Studio Code sau đó build file .ts bằng cmd với dòng lệnh tsc



## 4.1. index.html là trang chủ

## 4.2. init\_variables.ts (tsc = init\_variables.js) file chứa các biến (trong wwwroot/app)

export var hanhchinh\_url =

  //"http://118.70.182.154:6080/arcgis/rest/services/NDBanDo/MapServer";

  //"http://14.238.1.124:6080/arcgis/rest/services/NDBanDo/MapServer";

  "http://14.238.1.124:6080/arcgis/rest/services/VanToc/MapServer";

…

## 4.3. map\_variables.ts (tsc = map\_variables.js) file chứa các biến map, view, layers và basemap…(trong wwwroot/app)

import WMSLayer from "esri/layers/WMSLayer";//Thêm để đọc layer wms

import Basemap from "esri/Basemap";

……

//Thêm để đọc layer wms

//require(["esri/Map", "esri/views/SceneView", "esri/layers/WMSLayer"], (Map, SceneView, WMSLayer) => {

  //const layer = new WMSLayer({

    export var layer = new WMSLayer({

    //url: "https://ows.terrestris.de/osm/service",

    url: "http://ows.mapvn.vn:8080/wms?",

    sublayers: [

      {

        name: "bandodatyle"

      }

    ]

  });

export const map = new EsriMap({

  //basemap: "streets-vector"

  //basemap: "satellite"

  //basemap: "hybrid"

  //basemap : {baseLayers: [layer]}

  //basemap : {baseLayers: [map\_layer\_Background.getMapLayer()]}

  //basemap: basemap

});

export const view = new MapView({

  map: map,

  container: "viewDiv",

  center: [105, 15.5],

  zoom: 6,

});

export const basemapToggle = new BasemapToggle({

  view: view,  // The view that provides access to the map's "streets-vector" basemap

  //nextBasemap: "satellite"  // Allows for toggling to the "hybrid" basemap

  nextBasemap: basemap//{baseLayers: [layer]}//"streets-vector"  // Allows for toggling to the "hybrid" basemap

  //nextBasemap: "satellite",

});

/\*Tạm thời bỏ add chuyển đổi nền Background

view.ui.add(basemapToggle, {

  position: "bottom-left"

  //position: "top-right"

});\*/

//Thêm ảnh thể hiện tỷ lệ mũi tên bên phải phía dưới

export const imgScale = new Image();

imgScale.src="img/Scale.png";

view.ui.add(imgScale, {

  position: "bottom-right"

  //position: "top-right"

});

## 4.4. map\_layer.ts và toggle\_layer.ts(tsc = map\_layer.js và toggle\_layer.js) file chứa hàm đọc các layer hiển thị hoặc ẩn đi (trong wwwroot/app)

/// map\_layer.ts

import MapImageLayer from "esri/layers/MapImageLayer";

import \* as init from "./init\_variables";

export class MapLayer {

  private hanhChinhMap: MapImageLayer;

  getMapLayer(): MapImageLayer {

    return this.hanhChinhMap;

  }

  constructor(url: string) {

    this.hanhChinhMap = new MapImageLayer({

      url: url,

    });

…..

/// toggle\_layer

import { view } from "./map\_variables";

import { map\_layer } from "./map\_variables";

import \* as init from "./init\_variables";

interface HttpResponse<T> extends Response {

  parsedBody?: T;

}

var toggle\_layer = false;

$(document).on("change", ".layer-toggle", function () {

  //do something

  var id = $(this).data("id");

  //console.log(id);

  var sublayer = map\_layer.findSublayerById(parseInt(id));

  sublayer.visible = !sublayer.visible;

  //console.log(sublayer);

});

…

## 4.5. main.ts (tsc = main.js) file add các lớp dữ liệu vào bản đồ cũng như các nút lệnh vào view (trong wwwroot/app)

import { map } from "./map\_variables";

import { map\_layer } from "./map\_variables";

//import { map\_layer1 } from "./map\_variables";

import { layer } from "./map\_variables";

import { basemap } from "./map\_variables";

import { view } from "./map\_variables";

import Home from "esri/widgets/Home";

// import $ from "jquery";

//map.add(map\_layer1.getMapLayer()); // Thêm nhóm layer của service khác

//map.add(layer); // Thêm nhóm layer của service khác

map.add(map\_layer.getMapLayer()); // adds the layer to the map

// view.ui.move("zoom", "bottom-left");

// view.ui.add("logo", "top-left");

// create and add Home widget to the view

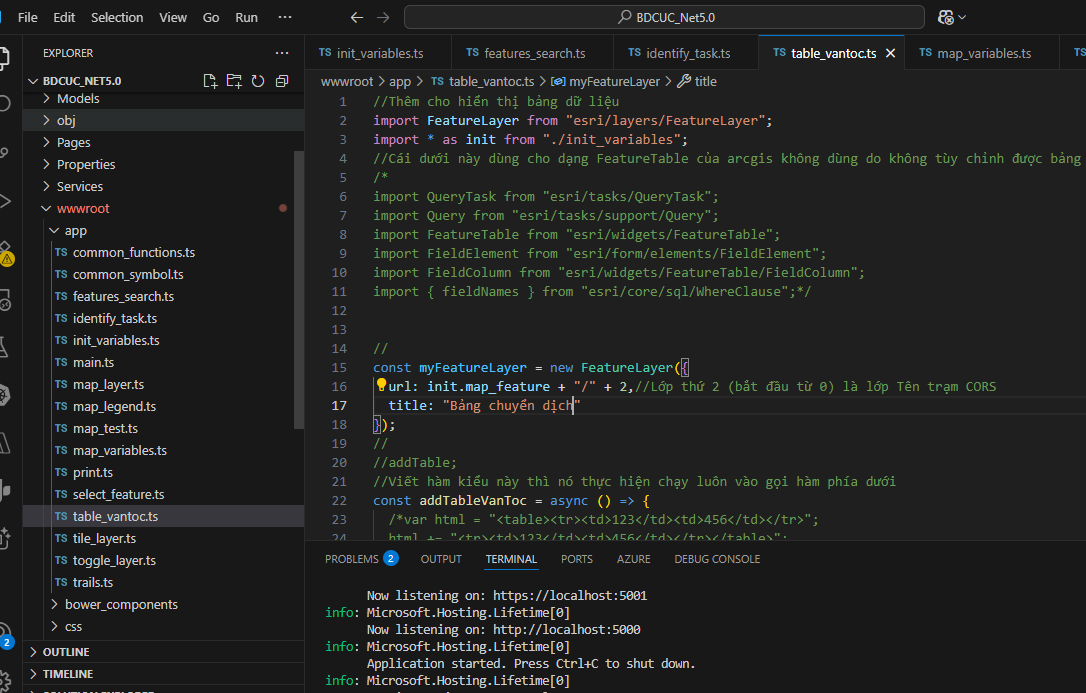
const home = new Home({

  view: view,

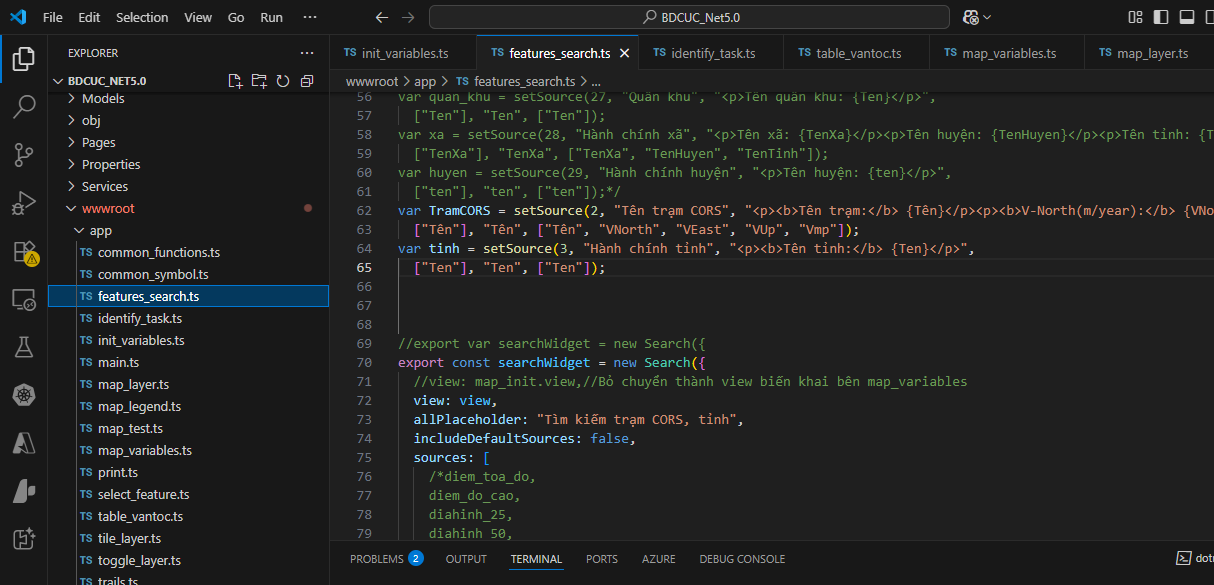
});

view.ui.add(home, "top-left");

## 4.6. table\_vantoc.ts (tsc = table\_vantoc.js) file hiển thị bảng giá trị của lớp đối tượng (trong wwwroot/app)



***4.7. features\_search.ts (tsc = features\_search.js) file hiển thị kết quả tìm kiếm ở thanh tim kiếm (đang tìm 2 lớp) (trong wwwroot/app)***



***4.8. identify\_task.ts (tsc = identify\_task.js) file hiển thị identify, sử dụng nút i để hỏi thông tin đối tượng bấm chuột trái vào (trong wwwroot/app)***

