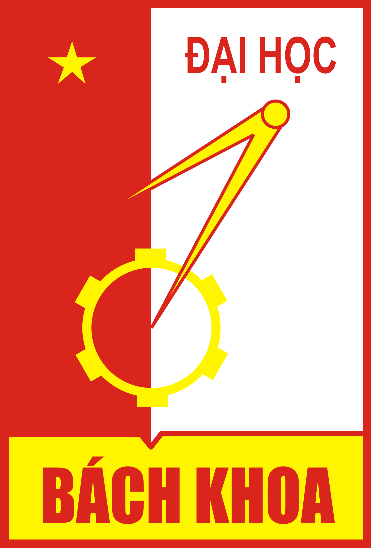
**Hanoi university of science and technology**

**School of information and communication technology**

**🙞🙜🕮🙞🙜-----**



**Graduation Research 1 Report**

**Topic: Function graphing calculator built with Reactjs**

Mentor: PhD.Nguyễn Tuấn Dũng

Student :

Đặng Quốc Trung – 20184317

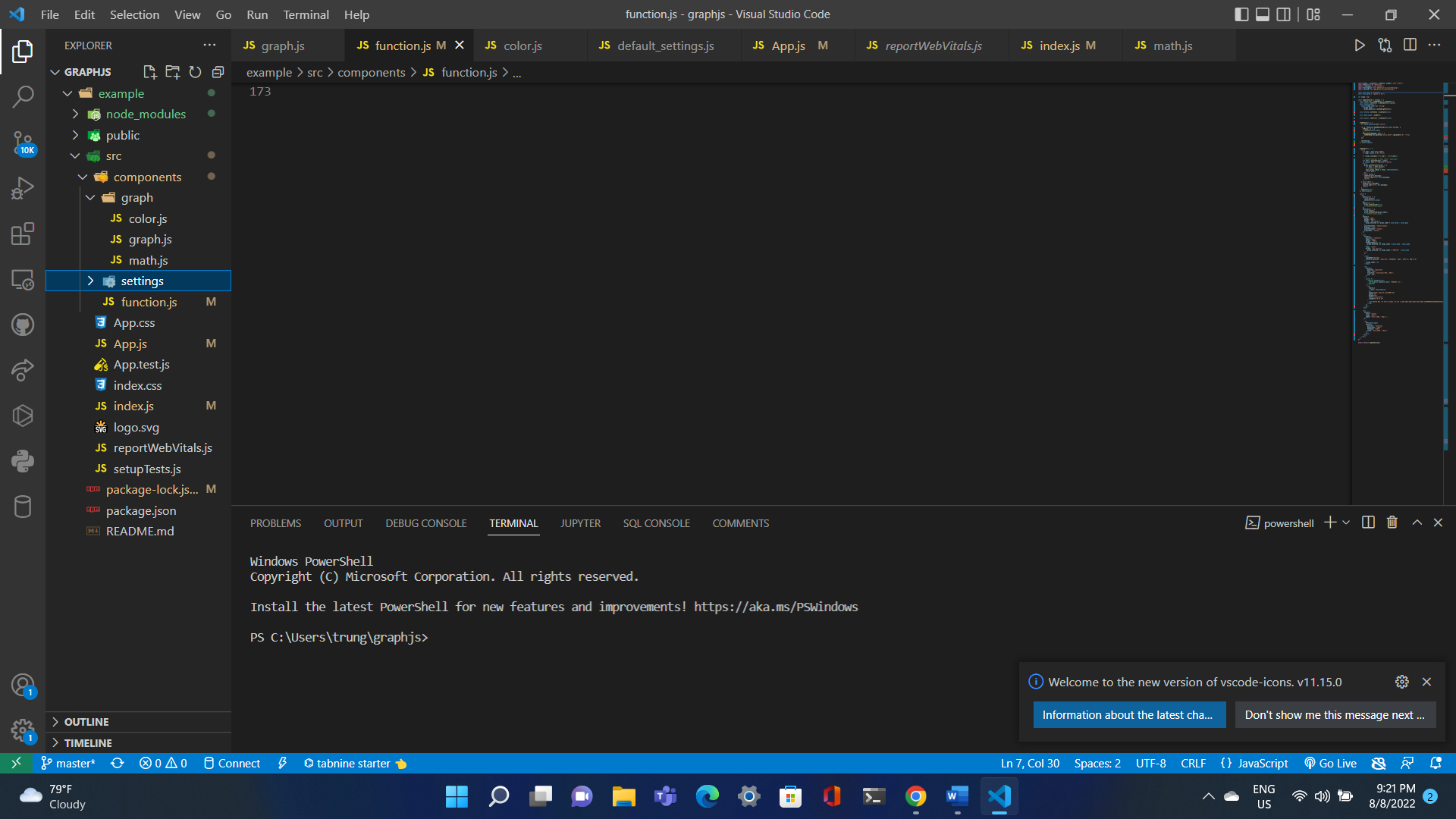
**Introduction**

My project involves creating a basic function graph in JavaScript. This is a helpful app for students who need to quickly create a basic graph with an input function.I reference the ui from desmos.com to develop this project.

It is unavoidable that mistakes will be made while implementing the topic; I look forward to receiving your feedback and suggestions to further improve this issue.

My project will use ReactJs for UI to improve user’s experience.It also allows customing components .

This is directory structure:



All the code are contained in folder src.

graph: contain code which handle all the use case related to graph and main axis

settings: handle the settings custom (in the future)

function.js:handle the input function from user.

App.js:  Contain app component- is the main component in React which acts as a container for all other components

Index.js:the main file to run code.

**Overview:**

Firstly, I will introduce important functions in graph.js in folder graph(the most important components of this app).

In graph.js, we will perform the main function of the application is to plot graphs,plot coordinates and main axis and zoom depending on the user’s actions.

Functions are used for plot main axis, coordinates ,grids are:

* drawGrid()
* drawCrosses()
* drawText()

Functions are used for plot graphs are:

* drawFunction()
* graphFunction()

Functions are used for zooming are:

* handleMouseMove()
* zoomIntoCursor()
* handleWheel()

Since I use Reactjs for this application , there are some states from useState which are used :

* ctx
* offset
* center
* size
* zoom
* mouseDown
* increment
* cursor
* canvasStyle

**Brief explanation about the code:**

First of all , in graph.js we will set state for ctx to canvas.current.getContext(‘2d’) for providing the 2d rendering context for the drawing surface of a canvas element.Then we draw the grid with drawGrid() function.Next,we plot the x-y axis with drawCrosses().moveTo and lineTo will be used to draw axis depend on the state center and the size of the canvas for graphComponent. After that,we draw the coordinates with drawText() in two axis x-y.The coordinates is rounded to the first decimal.

Next step , we want to plot graph with function input. The graphFunction(\_,x\_cords,inc) will draw the graph depending on the 3 parameters.

\_ is the array that indicates the functions which the user inputs.\_[0] is the function with parameter x return f(x) value.\_[1] is the color of that graph. We will use loop from x\_cords.start(the start of the grid) to x\_cords.end(the end of the grid) with the increment(inc) = absolute value of toCartesian(x\_cords.end)-toCartesian(x\_cords.end) divide 10000. The smaller the increment , the smoother the graph will be. With every coordinate x, we convert it to Pixel to use canvas draw the line from it to the corresponding y value.

Finally , I want my app can be zoomed by user’s event input.User can hold and move mouse to move all the content in graphComponent.The handleMouseMove() will resolve that use case.When user hold and move mouse, it will update the state offset by setOffset(offset is the pixel value of the displacement of the main axis).User can zoom the content with the position of the cursor by zoomIntoCursor().When we zoom, the offset will be changed so we need update the offset state by setOffset.The handleWheel will handle the event onWheel when we want to zoom with the wheel. An unit\_distance will be smaller when we zoom in and bigger when we zoom out.

In color.js , we will generate random color for multiple graphs .

In function.js (InputFunctionComponent) ,we will evaluate the expression of user’s input function.

Then we will call GraphDraw and InputFunction components in App.js and it will render all the content in this two components.The index.js will call App and run the application.

**Exprience**

Following the project, I learned a variety of frontend techniques and understood the fundamentals of javascript .E.g:Hook in Reacjs (useState,useEffect,useRef)

The project still has many limitations; hopefully, the mentor will be able to comment and make suggestions to improve the exercise.