CPSC 1520 Assignment 3: Dictionary Word Search Tool

Introduction

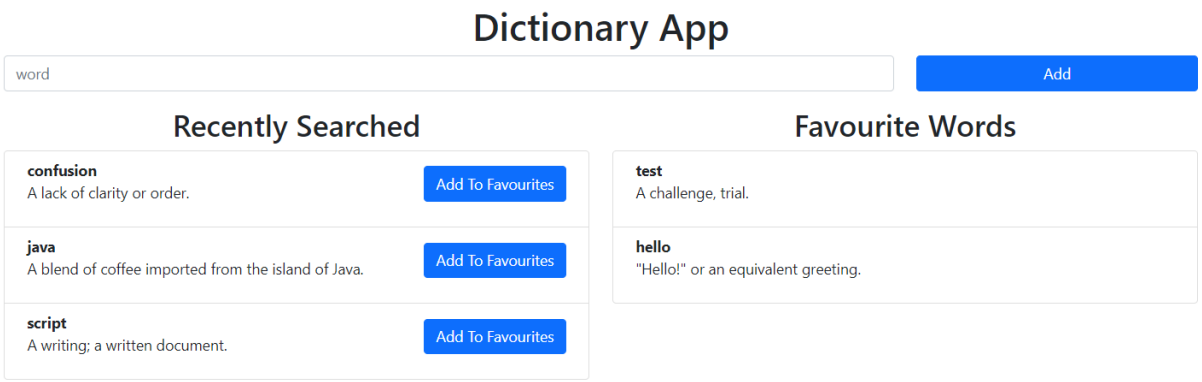
This assignment will test your knowledge of the last few topics of this course by creating a dictionary word search tool. Please refer back to all of the examples and all of the slides to complete this assignment, but it mainly focuses on the topics of “Fetch Fundamentals”, “DOM APIs and Timers”, and “NPM, Tools and ES Modules”.

Note you’ll also being using an external rest API https://dictionaryapi.dev/ to search words and populate them on the page. You might find it handy to use a rest api client such as postman to test your requests.

Overview of functionality

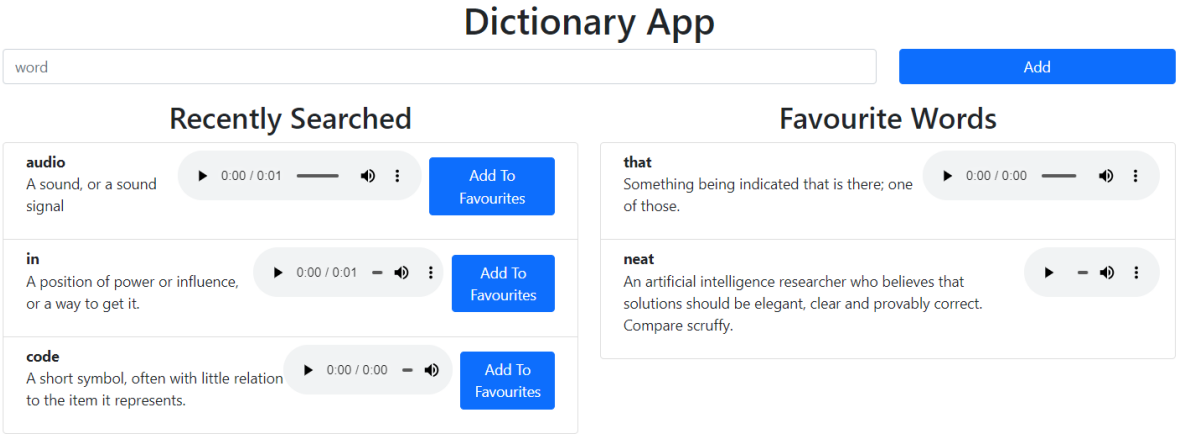
Here is the sample functionality in the image below.

- When you click “Add” your application will search for the word using the free dictionary rest api and add the html (structure provided in the JavaScript) to the “recently searched” list.
- When you click the “Add to Favourites” button the word will move from recently searched to the “Favourite words” list.



Sample Functionality Example (with Bonus)

The application has all of the same functionality as above except you also have the audio controls added as a part of the application. The HTML structure for the audio is not provided and you need to figure it out on your own (see sample screenshot on next page).



Required Tasks

- Install the bootstrap and parcel packages, and setup the required scripts using npm (as in class).
    - Your project should be using parcel and not Live Server when ran (it won't work anyways if you try it that way).
    - Your scripts need to be correct and should be the same as other projects you have done in class with your instructor.
    - Bootstrap should be included in your main.js file so your project looks like the images.
    - NOTE: In your solution there should be no node_modules folder or ".parcel-cache" folder. Marks will be taken off if they are included.
- Create a function (that takes a single word as a parameter) in the "api/dictionary.js" file that will use fetch to call the rest endpoint of the free dictionary api, return all of the data from that endpoint and export that function from that file.
    - Your function should be a promise and should resolve and return the data so that the function itself is a promise; basically, return the fetch call.
    - This needs to be in the "api/dictionary.js" file or you'll be penalized on this.
- Create a function named "createWordItem" (and export this function) in the file "dom/word-item.js" that will use the DOM API to create element, update the class list of that element, update the text content of the element, and set any attributes required by the html structure (provided in that file).
    - This needs to be in the file "dom/word-item.js" file or you'll be heavily penalized on this.
    - This function will take one parameter wordData which will be one item from the data of the rest endpoint.
        - Get the "word" from the wordData.
        - Get first "definition" in the first "meaning" from the wordData.
- In the main.js file import the two function created above.
- In the main.js file, create an event listener that will listen to the submission of the form and will take the first array value of the array returned by the function from "api/dictionary.js", pass it in to the "createWordItem" function.
- In the main.js file, create an event listener on the "searched words" list that will listen to clicks, check if it's the button on the word item (the best way to do this is check if the element contains

a certain class). If it is then it will add that element to the “Favourite Words” list and also remove the button itself.

- Your code must adhere to the Google style guide (Nathan).
- **BONUS** in the “createWordItem” function you will create an audio element control that will be able to play the first valid audio url from the “phonetics” part of the wordData (refer to the rest api).
    - if it doesn’t work you don’t get any marks, you’ll have to do research on this one and you can’t ask the instructor. You need to search the answer on your own for the bonus!

Marking key

Tasks	Grade	Marks	Total
NPM and Packages. • Packages are installed correctly and are in the correct dependency section (dev dependency vs dependency) • Scripts are setup correctly and the source attribute in the package.json is setup correctly. • Node_modules folder and parcel cache are included (if you included them it’s negative marks on your assignment)		**3** **3** **-5**	
Fetch • The fetch request is takes in a word, and returns the data from the fetch request as a promise. • The function is in the correct file and exported correctly • The function is imported correctly in the index.js file		**5** **1** **1**	
Dom API and Manipulation. • Create word item function is in the correct file and exported correctly. • The function is imported correctly in the index.js file • The create word item function uses ONLY DOM API to create the html structure (if innerHTML is used this section is zero.) • The create word item returns the element created. • The form is intercepted and prevented from submitting in the index.js. • The event handler of the form calls the fetch api function, creates the word item and attached it to the correct area on the page. • On the recently search words an event handler is added and using the dom api it appends it to the new area and removes the button correctly.		**1** **1** **5** **1** **1** **3** **3**	
Bonus • Audio element is correctly formed and added in the correct order. • Audio can be played from every word, and is correctly extracted from the word data.		**3**	
Code Formatting and Style		**-3**	

Marking Rubric

Marks	5 Marks Criteria
5	**Task was completed with the highest of proficiency adhering to best practices and followed subject matter guidelines all tasks were completed to a professional standard.**
4	**Task was completed well some minor mistakes. Well above average work shows good understanding of the task and high degree of competence**
3	**Satisfactory work some features missing or incorrectly implemented. Show a moderate level of understanding in the task with room for improvement.**
2	**Below average work. Task was poorly complete. Show understanding of the task and the requirements to implement but implementation was poorly executed.**
1	**Some of the task was completed. Showed a lack of understanding in the subject matter and very poorly executed**
0	**Not completed.**

Marks	**3 Marks Criteria**
3	**Proficient shows a high degree of competence in completing task.**
2	**Capable above average degree of competence in completing task**
1	**Satisfactory shows a satisfactory degree of competence in completing task.**
0	**Shows a limited degree of competence in completing task.**

Marks	**1 Marks Criteria**
1	**Task Completed satisfactorily**
0	**Task was not executed.**