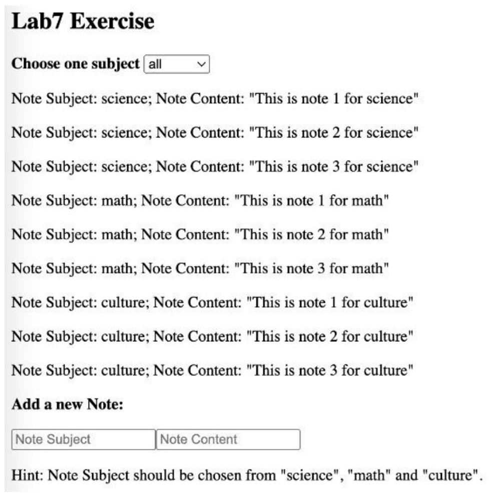
COMP3322B Modern Technologies on World Wide Web
Lab 7: React

Overview
In this lab exercise, we will use **React** to implement an interactive web page, which displays a list of subject notes. We can filter these notes by choosing a subject from a drop-down list and add a new note by filling in the input boxes and clicking the submit button. Please refer to the following screenshots:


Lab7 Exercise

Choose one subject: all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

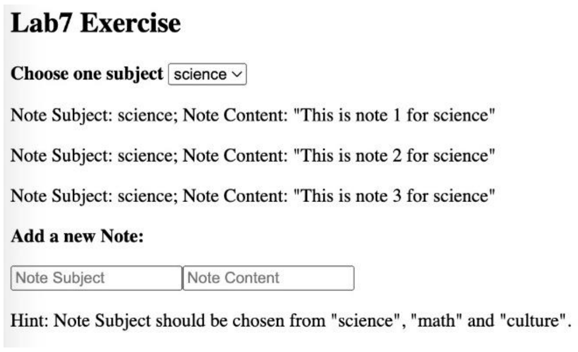
Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

Note Subject | Note Content

Hint: Note Subject should be chosen from "science", "math" and "culture".

Upon initial page load, you will see a drop-down list with the default option “all”, and a list of notes right below showing the subject (chosen from three subjects: science, math, culture) as well as the note content.


Lab7 Exercise

Choose one subject: science

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

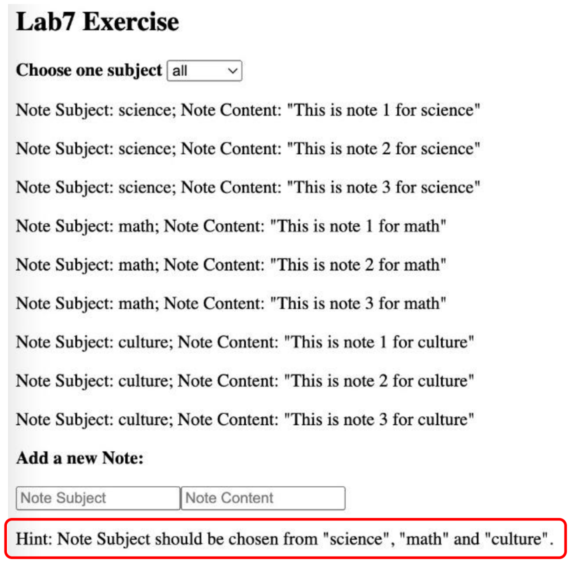
Note Subject: science; Note Content: "This is note 3 for science"

Add a new Note:

Note Subject | Note Content

Hint: Note Subject should be chosen from "science", "math" and "culture".

You can click the drop-down list to show a list of options (subjects) besides “all”. By clicking one of the options, the note list will be re-rendered to contain notes belonging to the chosen subject only. For example, when we select the subject option “science”, the notes list changes accordingly.



Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

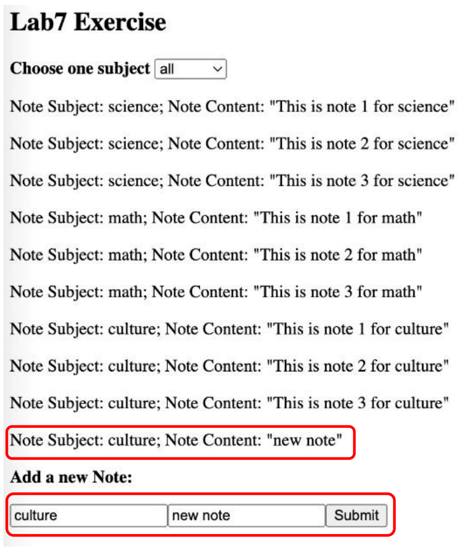
Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

Note Subject | Note Content

Hint: Note Subject should be chosen from "science", "math" and "culture".



Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Note Subject: culture; Note Content: "new note"

Add a new Note:

culture | new note | Submit

Underneath the notes list, you can add a new note by filling in the two boxes with note subject and note content, respectively. Only when your input note subject is among the three specified categories, i.e., "science", "math" or "culture" as shown in the Hint message, the "Submit" button will be rendered. For example, before we fill in the "Note Subject" box, we can only see a default Hint message; after we type "culture" in the box, the Hint message disappears and the "Submit" button is rendered on the right. After clicking the "Submit" button, the notes list will be updated accordingly, if the selected option in the drop-down list is "all" or the same subject as the newly added note's subject.

Lab Exercise

Part 1. Prepare the React App

Step 1. Create a new React App using "create-react-app" command

Launch a terminal. Go to your "lab7" directory and create a React app named "myreactapp" using the following commands:

```
cd YourPath/lab7
npx create-react-app myreactapp
```

Go inside the "myreactapp" directory just created.

```
cd myreactapp
```

Then launch the React App as follows:

```
npm start
```

After successfully launching the app, you should see prompts like the following in your terminal:

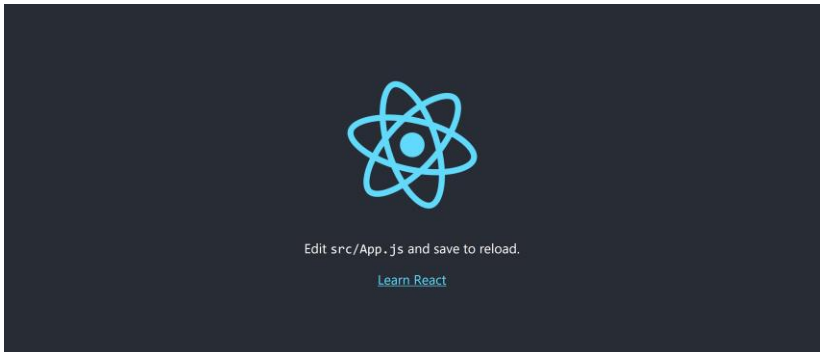
```
Compiled successfully!

You can now view myreactapp in the browser.

  Local:            http://localhost:3000
  On Your Network:  http://xxx.xxx.xx.xx:3000

Note that the development build is not optimized.
To create a production build, use npm run build.
…
```

And a web page should be loaded automatically in your browser, as follows:



Step 2. Copy the provided files to myreactapp/src folder

Download and extract the files in "lab7_materials.zip". You will see two JavaScript files "**App.js**" and "**index.js**". Replace the original "App.js" and "index.js" files in **myreactapp/src** with the provided files.

Open "**index.js**" and compare it with the default one (page 9 of lecture notes 14_React_I_COMP3322B_s2022.pdf). The differences are:

1. Import of "index.css" is removed. We will not use CSS in this lab.
2. Instead of

   ```
   import App from './App';
   ```

   we import

   ```
   import Lab7App from './App';
   ```

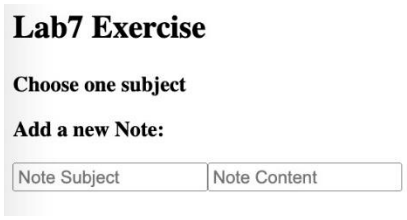
   And in the render function, we also replace "<App />" with "<Lab7App />".

"Lab7App" is the React component which renders the main page, as implemented in "**App.js**". Open "**App.js**" and observe the render() function of the class "Lab7App". You will see the "Lab7App" component returns a <div> containing four components as children:

1. A <h2> tag displaying the title "Lab7 Exercise".
2. A <DropDownList /> component rendering the drop-down list below the title.
3. A notes list containing notes corresponding to the selected subject option.
4. A <NewNote> component rendering the two input boxes, as well as the hint message and the submit button (conditionally).

You can refer to the screenshots at the beginning of the handout to understand more of the components.

In this lab, we will only modify "**App.js**" to achieve the web page. Before you modify "**App.js**", the initial page is as follows when you launch the app using "npm start":



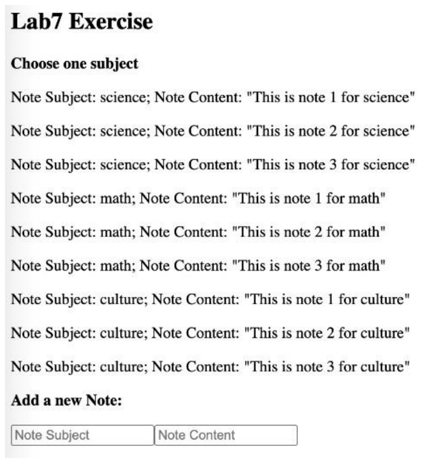
Part 2. Show notes list and filter notes using the drop-down list

Step 3. Render notes list in class Lab7App

Open "**App.js**", class Lab7App has a state with two key-value pairs: (1) *userNotes* stores the list of notes to be shown on the page, initialized using the given array *userNotesDB*; (2) *currentFilter* indicates the selected option in the drop-down list, which is needed to decide whether a newly added note should be shown among the current notes list on the page. We also add a handler function **handleNotesChange()** to change the state values upon selecting a different option in the drop-down list or adding a new note (to be implemented later).

Complete the render function of class Lab7App: within **{ }**, apply **map()** function on *userNotes* to create a number of <p> elements, each presenting information of one note in the *userNotes* array as follows: <p>Note Subject: {note.subject}; Note Content: {note.content}</p>.

After this step, you will see the notes list shown up on the page:



Lab7 Exercise

Choose one subject

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

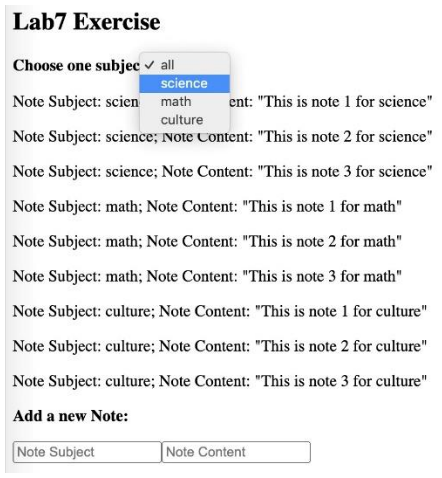
Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

Step 4. Render a drop-down list in class DropDownList

Now go to class DropDownList, which receives the handler function **handleNotesChange()** of class Lab7App as its props *handleNotesChange.* Complete its render() function as follows: remove "{/* step 4*/}", and in its place, create a drop-down list using the <select> element with four options (with values “all”, “science”, “math” and “culture”, respectively). The option “all” should be placed first, or associated with the attribute “selected” as the default option. Use “e => this.props.handleNotesChange(e.target.value)” as the handler function of onChange event on the <select> element (to track the change of selected option and update notes list accordingly).

After this step, you will see the drop-down list above the notes list, but the notes list will not change according to different selected options yet.



Lab7 Exercise

Choose one subjec

Note Subject: scien ent: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

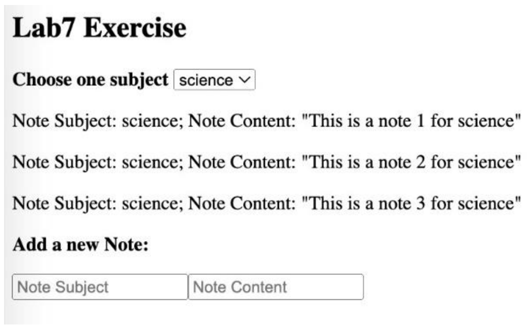
Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

Step 5. Implement filtering of notes list

Now implement the function **handleNotesChange()** in class Lab7App as follows: if value of the argument *option* is not "all" (i.e., it is one of the three subjects instead), apply **filter()** function (learn more about JavaScript Array filter method at https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/filter) on *userNotesDB* to obtain a filtered notes array where each note's subject is *option,* set the state *userNotes* to be the filtered array and set the state *currentFilter* to be *option*; otherwise, set the state *userNotes* to be *userNotesDB* and state *currentFilter* to be *option*. Note that you need to use **this.setState()** to change a state variable's value.

After this step, when you select one subject from the drop-down list, the notes list will change accordingly:


Lab7 Exercise

Choose one subject science

Note Subject: science; Note Content: "This is a note 1 for science"

Note Subject: science; Note Content: "This is a note 2 for science"

Note Subject: science; Note Content: "This is a note 3 for science"

Add a new Note:

Part 3. Implement Add of a new Note

Step 6. Handle input changes in the two input boxes

Go to class NewNote, which has two props received from parent component Lab7App: function *handleNotesChange* and *currentFilter*. We have created a state variable for each input box's value in the component, which is updated whenever the value of the input box changes, by using **handleInputChange()** function as handler of the onChange event. In this way, values of the state variables are always in sync with values of the input boxes. We have also provided the implementation of function **handleInputChange()**, where we use **this.setState{[name]: value}** to set value of a state variable according to the respective input value, where *name* refers to the name attribute of the input element and *value* refers to the input value. Learn more about the usage of [name] at https://reactjs.org/docs/forms.html ("Handling Multiple Inputs").

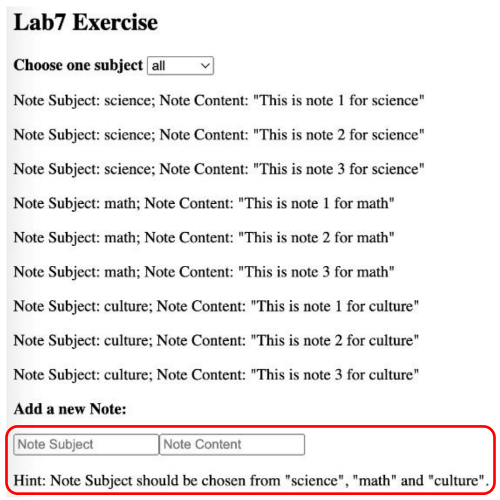
Step 7. Implement conditional rendering of Hint and Button

7.1. In the render() function of class NewNote, remove "{/* step 7.1 */}", and in its place, render a HintAndSubmit component. The HintAndSubmit component receives four props:

1. *new_subject*: equals the value of state *inputNoteSubject*;
2. *new_content*: equals the value of state *inputNoteContent*;
3. *handleNotesChange*: equals this.props.handleNotesChange (function received from parent component NewNote).
4. *currentFilter*: equals this.props.currentFilter (received from parent component NewNote).

7.2. Now go to class HintAndSubmit. Remove the given "return" in the render() function (which we added to allow your code to be compilable to render intermediate pages since every render() function needs to call "return"), and implement the render() function as follows: check if the value of prop *new_subject* is among "science", "math" and "culture": if so, render a "Submit" button element with onClick event handled by function **handleClick()** (to be implemented later); otherwise, render a hint message in a <p> element *Hint: Note Subject should be chosen from "science", "math", and "culture".*

After this step, you will see the component is rendered conditionally: when you type a valid subject name, the "Submit" button will show up; otherwise, the Hint message is displayed.


Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

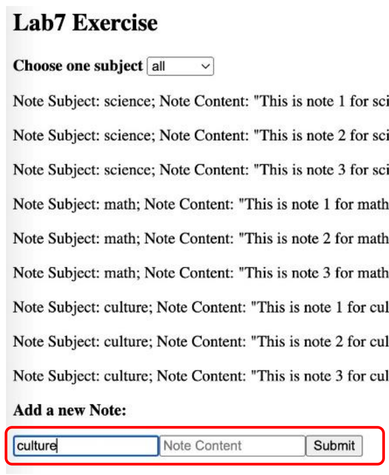
Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

Note Subject | Note Content

Hint: Note Subject should be chosen from "science", "math" and "culture".


Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

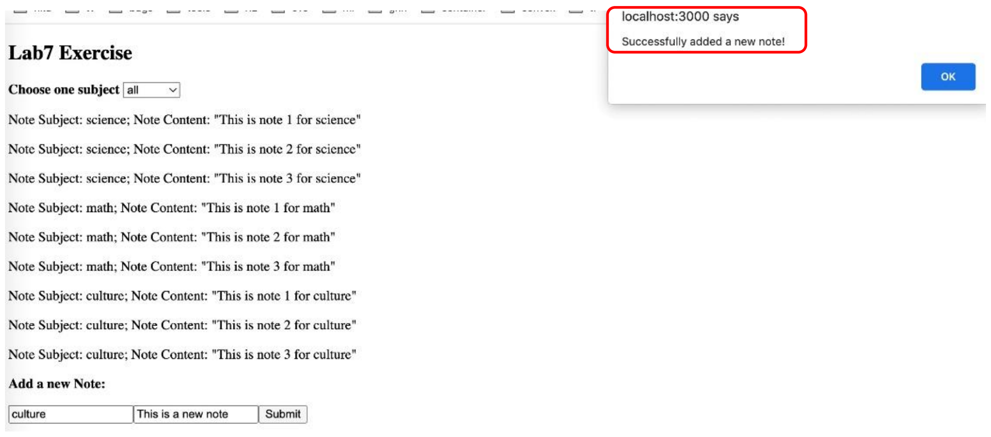
Add a new Note:

culture | Note Content | Submit

Step 8. Implement onclick event handler of the "Submit" Button

Complete the function **handleClick()** in class HintAndSubmit: Push the *new_note* object to the array *userNotesDB*, alert a message "Successfully added a new note!", and then update the notes list displayed by calling **this.prop.handleNotesChange()** with an argument equaling the value of prop variable *currentFilter*.

After this step, on the rendered web page, you can add a new note by clicking the "Submit" button. If the current selected option in the drop-down list is "all" or equals the subject of your newly added note, you will see the notes list changes immediately with the new note displayed; otherwise, you can check whether the new note is successfully added by selecting the "all" option or the subject option corresponding to that of the new note in the drop downlist.



localhost:3000 says

Successfully added a new note!

Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

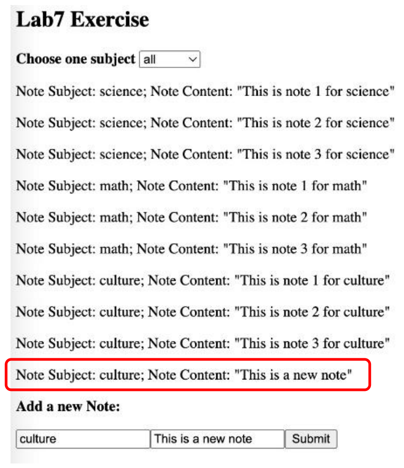
Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Add a new Note:

culture | This is a new note | Submit



Lab7 Exercise

Choose one subject all

Note Subject: science; Note Content: "This is note 1 for science"

Note Subject: science; Note Content: "This is note 2 for science"

Note Subject: science; Note Content: "This is note 3 for science"

Note Subject: math; Note Content: "This is note 1 for math"

Note Subject: math; Note Content: "This is note 2 for math"

Note Subject: math; Note Content: "This is note 3 for math"

Note Subject: culture; Note Content: "This is note 1 for culture"

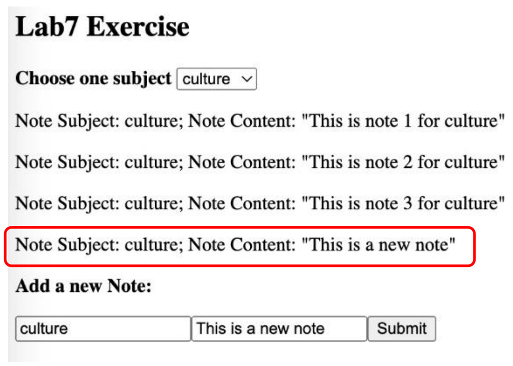
Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Note Subject: culture; Note Content: "This is a new note"

Add a new Note:

culture | This is a new note | Submit



Lab7 Exercise

Choose one subject culture

Note Subject: culture; Note Content: "This is note 1 for culture"

Note Subject: culture; Note Content: "This is note 2 for culture"

Note Subject: culture; Note Content: "This is note 3 for culture"

Note Subject: culture; Note Content: "This is a new note"

Add a new Note:

culture | This is a new note | Submit

Congratulations! Now you have finished Lab 7. You should test the page and the final results should look similar to the screenshots at the beginning of this document.

Submission:

Please finish this lab exercise before 23:59 Wednesday April 20, 2022. You should submit the **App.js** file only.

(1) Login Moodle.
(2) Find “Labs” section and click “Lab 7”.
(3) Click “Add submission”, browse your zip file and save it. Done.
(4) You will receive an automatic confirmation email, if the submission was successful.
(5) You can “Edit submission” to your already submitted file, but ONLY before the deadline.