Lab Session 6

# Theme of this Lab Session

Today Lab is split into many parts. Each part is splitted into different questions. This lab is to prepare the front-end developer of your group to learn technical skills to develop his or her skills to enable them to write a front-end web-page to your DApp. It might be at times that the developer who is responsible of creating a front-end requires some knowledge of backend as you will see by the end of this lab. However, if that was the case, then the front-end developer needs to communicate his/her thoughts with the back-end JavaScript developer so that both could find a solution to the problem.

In this lab, if you are neither the back-end developer i.e. the individual who is concerned with JS or the front-end developer then you only need to solve part 1 of this tutorial. Otherwise, there is no need to know further information in this lab.

# Pre-requests skills for this Lab

* Having and running VS code editor with the configuration done in Lab session 1.
* Node
* JavaScript

Main Lab Questions

# Part 1: Reading, Understanding, and Modifying moderate Smart Contract using Solidity programming

Mini-video expected to be watched in order to conduct this part: 1, 2,3,

Expected time: 45 min

This question is dedicated to familiarize you with fundamentals of ReactJS. This practice is vital because it teaches developers. If you were able to answer these questions with your colleagues easily then you are ready to move to the next question. Remember although the questions here do not require you to illustrate with coding. It is a useful exercise to try to relate these concepts to the pre-recorded mini-lectures.

## Questions:

1. What is ReactJS?
2. Why a lot of dapp developers use react and why is it important to learn React?
3. What is Single-Page Application? Where is that static webpage that keeps updated found in the directory of your created React?
4. Initiating a React app on your machine:
   1. Please create a new folder in your directory calling it “Lab6”
   2. Open terminal on that folder
   3. Write the following commands   
      
   4. Open VS code editor
5. What is JSX and what is the difference from JS?
6. What is a React Component?
7. How does React work in the browser?
8. Can web browsers read JSX directly?
9. What is the virtual DOM? And how does React uses virtual DOM?
10. What is an event in React?
11. What is the use of render() in React?
12. What is a state in React?
13. What is a hook in react?
14. How do you update the state of a component?
15. What are props in React?

# Part 2: Understanding practical coding in React

In this question, we will create many components in React similar to the mini-lectures. You will be going through a set of many subquestions that will make you capable of doing your own modifications and coding.

Mini-video expected to be watched in order to conduct this part: 1, 2,3,

Expected time: 60 min

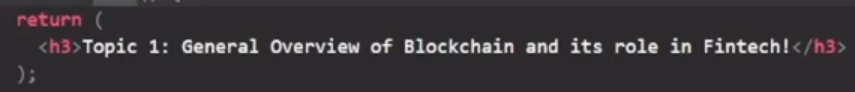
## Part 1: Prepare your React Project

1. Initialize a react application similar to ML3L6. See also part c in sub-question 4 of part 1.
2. After initialization you should find the src folder containing an App.js file. This is the main file where we need to do some modifications. In particular, you can edit the root component and import other component files here. That is why when you create new components you will need to import them into this file. At this point in time, we need to do some modification to this file. As follow:
   1. Go to “App.js” and delete all of the pre-loaded data as so:  
      Text

      Description automatically generated  
        
      You should have an “App.js” that looks like so:  
      Text

      Description automatically generated
   2. You can also delete the React pre-loaded logo as it is no longer required which is also located in the src folder and remember to remove the import in the “App.js” file.

## Part 2: Creating your first Component

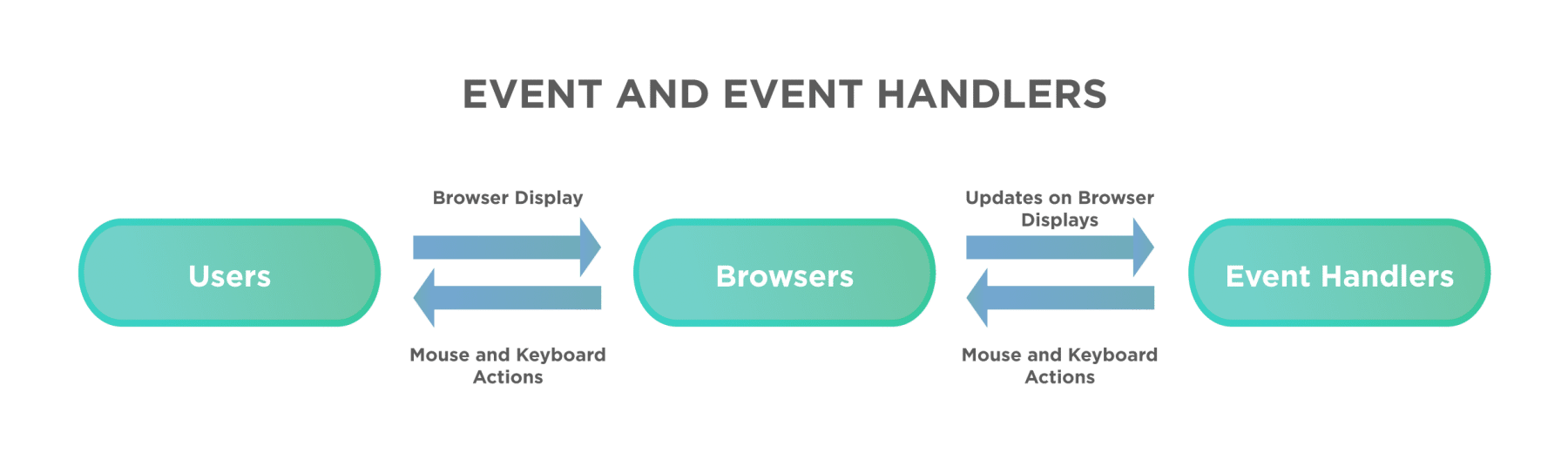
1. Create a new file in the “src” folder call it “Components” where all your React components will be located.
2. Download the “bootstrapcustom.css” available [here](https://uob-my.sharepoint.com/:u:/g/personal/zm21385_bristol_ac_uk/ESTJ8MP32sRMhG3IzNTAVogBlqqRAFhheLThcqNNOEuAEw?e=LySL9A) into this folder.
3. Create a “WeekTitle.js” page in your “Components” folder. Why “.js” as opposed to “.jsx”?
4. In this file, import react and the “bootstrapcustom.css” and start writing a new function called WeekTitle as so:  
   
5. Create a simple return of your choice to this function. An example would be  
   
6. In the same file export the function as so   
   
7. Import this function in your “App.js” as so  
   
8. Why is it important to use React Fragments?
9. Run this app and see what you get. Is it what you expect? If not, please raise your hand and one of the teaching team will come to support and provide you with feedback.
10. Introduce new styling to your output by importing the “bootstrapcustom.css” file that you downloaded. This needs to be done using className as opposed class. That is by writing the following code into your “WeekTitle.js”:  
    Text

    Description automatically generated  
    Why do we need to use className as opposed to class attribute in React?
11. Save your updated coding file. Did the webpage automatically updates its self? Why was that the case?

## Part 3: Creating your first properties feature within components

1. We can create variables in components and change these variables the same way we do with vanilla JavaScript. We can do this by changing the “WeekTitle” function as so:  
   

Realize the curly braces under which these variables are inserted in the return function. What is the task of these curly braces?

1. It is known that Data within React Components is stored as either properties or states. Moreover, Properties are the input values to the component. They are used when rendering the component and initializing the state. Following the previous sub-question, introduce “TopicNumber” and “TitleName” as properties to the function “WeekTitle”. Furthermore, create two versions of the component in your main “App.js” file with a different input value to each of these two variables.
2. In the “WeekTitle” function, we need to introduce a click handler. A Click handler is one of the ways a user can interact with the browser.   
     
   **Event handlers** are the JavaScript code that invokes a specific piece of code when a particular action happens on an HTML element. The event handler can either invoke the direct JavaScript code or a function. In particular, the event handlers are the properties of the HTML or DOM elements, which manages how the element should react to a specific event. This figure illustrates this concept:   
     
     
   Briefly, the user performs a particular mouse or keyword action on the browser, this triggers the corresponding event handler associated with that HTML element. The event handler, in turn, executes a piece of JavaScript code, which performs a particular action on the webpage, and the browser displays the results of those actions to the end-users. Please list and describe different types of event handlers provided by JavaScript?
3. Introduce a Click button into the React Component function “WeekTitle” and create an associated handler that will print “hello word”. (hint: if you want a review please look at ML4 at time 6:00 min)

## Part 4: Introducing React State variables into your components

Following up on part 3, as front-end developers, we need to change the DOM when the button is pressed. This means that we need our browser to be dynamically changing.

The issue with using properties (or what is known by React “property data”) is that after instantiating the component, properties should be considered immutable. This implies that Property values can only be set when instantiating the component or when the component is re-rendered in the DOM. In particular, React will compare the old and new property values to determine what DOM updates are required (hence making the webpage feels to the user as dynamic). Unfortunately, by simply using JavaScript coding using property data we can not change the property values in an event handler (Please watch ML4 at min 8:30 to get the details). Therefore, we need a different type of data in React.

Remember that Data within React Components is stored as either properties or state. The advantage of React “State data” is that it can be changed by the component and is usually wired into the component’s event handlers. This is a very desired feature to have and as a front-end developer, you need to know how (handle) to do such coding. The questions in this part will break it down for you to do such coding.

1. What is a State variable and how do we declare it in React? Furthermore, why should hooks be involved?
2. If you are using VS code editor, you might find it beneficial to download an extension known by “ES7+ React”. This will help you use typical snippets in React JS (have a look into ML4 min 10).
3. In the React functional component “WeekTitle” let the event handler change the “TopicNumber” of your document when pressing on the button.
4. Create another State variable known by “TitleName” which was considered the second property in the “WeekTitle” functional component.
5. In the functional component “WeekTitle” create a textbox. This textbox should take an input text. Then once you click on the button, the title name of the week must change as to what has been written in that textbox. To do so, you will need to realize and answer the following set of mini-questions along the way. (hint: to answer this question, please watch ML6)
   1. What is an <input> tag in HTML do?
   2. What are the list of attributes that are specific to the input tag?
   3. Which input tag attributes do we use to create our task and why?
   4. In answering sub-question c, you must have realized that there is one generic event attribute that we will need to use in order to conduct our task. This event attribute is known as “onChange”. Realize that there is a difference in the letter “c” of the generic HTML “onchange” attribute vs the “onChange” special attribute in React. What does this attribute do?
   5. When deciding to change the name upon button click. Why do you think it is important to introduce a new state variable when deciding the value that is written inside the textbox?
   6. Write the updated full code of the React functional component “WeekTitle”.

## Part 5: Metamask and React Components

Sometimes using the MetaMask wallet directly is the simplest approach to making transactions on Ethereum blockchains. Therefore, you will find that as a front-end developer you need to create a user interface that is easy for anyone who does not know the technological technicalities to be able to use your group’s Dapp without any hassle.   
  
In this question, you will be introduced to how to tackle such issues and learn more about the details of connecting a MetaMask account to your Dapp.

1. In your VS code editor (hint: check min 2 in ML7):
   1. create a new component and call it “MetaMask.js”.
   2. Inside the file,
      1. import React, useState, bootstrapcustom.css, and ethers.
      2. Create a new exported function call it “UnderstandingMetamask”
      3. Write an h3 tag element that asks the user whether he/she wants to be connected to metamask.
      4. Add two buttons to the return JSX code that says “Yes” and “No” respectively along with their associated two “onClick” SyntheticEvent wrapper methods “ClickHandlerYes” and “ClickHandlerNo”.
   3. Go to “App.js” and import this function.
2. What is the “useEffect” hook and when do we use it? Can you illustrate its use with an example? (hint: adapt the knowledge you gain from this [webpage](https://dmitripavlutin.com/react-useeffect-explanation/#:~:text=1.-,useEffect()%20is%20for%20side%2Deffects,calculations%20are%20named%20side%2Deffects.), to illustrate to your group colleagues the answer to this question)
3. Why is “useEffect” hook important for initialization, particularly when using metamask?
4. How do we check if Metamask is installed on the machine of the user? How do we code it in React?
5. Does Metamask inject web3 into the browser? What does injecting web3 into the browser mean?
6. What does “windows.ethereum” do? In ML7L6 min 9:37. Rami has written the following code.  
     
   May you explain this code?
7. Using the ethers library, how do we check if MetaMask is connected? i.e. the user is logged into Metamask.
8. Write a code inside the function “ClickHandlerYes” that displays the public account and the balance of the used Ethereum account. Please do this without the use of web3 library. In the return, JSX code use the inline if-else condition to conditionally render what is displayed on the DOM. (Hint: see this [link](https://reactjs.org/docs/conditional-rendering.html#:~:text=Inline%20If%2DElse%20with%20Conditional%20Operator) for further details on how to conditionally render elements)
9. If we want to change the “ClickHandlerNo” to stop from displaying the user account address and balance. Do we need to confer this function to an asynchronous one, given that we will need to call a react state variable?
10. Modify your code so that if the front-end user changes the Metamask account or network, the displayed account and user’s balance automatically changes based on the retrieved values. Please include this in the “useEffect”.
11. Modify the code, so that it has a textbox that input is stored in a state variable called “BoxInput”. (hint: remember that you will need to use the “onChange” within the JSX and create a new function that stores the input into the react state variable “StoreTheInputInState”)
12. How could we change the styling of the input tag to look nicer? In particular, can you add the following styling effects to the input textbox?   
    
13. Modify the code so that when you press the “Yes” button, it does the following:
    1. Check if the entered text box is empty. If it was empty it aware the front-end user that the textbox is empty.
    2. Check if the entered textbox is a mnemonic phrase. If it was not a mnemonic phrase, it alerts the user that the entered text is not a mnemonic phrase and it empties the textbox. (hint: see what “isValidMnemonic” do and which library does it exist).
    3. If the textbox is a mnemonic phrase, it connects to the first account on that mnemonic and displays the public account along with the balance on the connected network of the current metamask provider. You are not allowed to use infura account for this application!
    4. Describe what you have done to a knowledgeable programmer.
14. At many instances, as a front-end programmer, you want to ensure that regardless of the network selected on metamask, you want to fix the network to Rinkeby. That is, even if the front-end user of your dapp chooses the mainnet, you want them to be restricrted to the Rinkeby testnet. How could you do such a code in React?

## Part 5: Dealing with contracts and React components.

In previous parts, we have only designed a front-end interface to connect to metamask and use basic retrieving data from the blockchain such as reading balance. Although this sounds simple, its practice was not as straightforward. It requires some subtle lessons to be learned along the way.   
  
In this part, we need to build a simple storage contract front-end interface to get connected to the local testnet. This among other things will require you to use truffle. There are many things that you need to know in order to design and get the know-how knowledge to connect your front end development with sending transactions on the blockchain. The questions below contain essential information that you are expected to know when you build your own Dapp.

1. Getting Started and Setting up
   1. Create a new component call it “interactingSmartContract.js”
   2. Write the following code initialization  
      
   3. Import this function in ‘App.js’ and display it using return JSX in that JS file.
2. To deploy contract, we need to ensure that we are connected to the metamask extension. Using the knowledge you gained, modify the above code to ensure that you are connected to the Metamask’s account and network.
3. To deploy a contract, we need to get the contract abi and bytecoe. The one advised in general is to do it using truffle. Follow the following steps to be able to do it in truffle. (Hint: Please watch video ML9 at10 min)
   1. Create a new folder (not in the React app directory)
   2. In the terminal write “truffle init” which should initialize the truffle.
   3. In the subfolder created “contract”, delecte the “Migration.sol” file
   4. In that subfolder create a new file call it “Storage.sol”
   5. Paste the following code into it:  
      
   6. Go to the “migration” subfolder and change the “migration” inside of artifacts.require to “SimpleStorage” (the contract name!)
   7. Fire up ganache (you might need to exist the ganache-cli open in another terminal!).
   8. In the fired up ganache terminal, write “truffle migrate”
   9. This creates a new subfolder called “build” inside that folder there is another subfolder “contract”. There you will need to copy the file created under the name “SimpleStorage.json”
   10. The code is available here (double click to open):  
       
   11. Either copy the file and put it in your React app component.
4. Connect to local blockchain (Ganache)
   1. Open a new terminal in your VS code editor and code “ganache-cli -m YOUR\_MNEMONIC\_PHRASE”. In my case it is as follows:   
      
   2. Change Metamask network to Localhost 8545
   3. Modify the code, to ensure so that when you press on “Deploy contract”, the code prints to you the public address and the user balance. Check that the balance is equal to 100! If it is, you are ready to go to the next question!
5. Modify the code so that you can send test ether from your first account to another account within the local ganache lists of accounts! When writing the code, please take into account the following issues:
   1. Please use “ethers” library only!
   2. Make sure that you reset the account by going to Metamask extension, then to “Settings”, then to “Advanced”, then press on “Reset Account”
   3. Make sure that your “Advanced gas Controls” in your Metamask extension is set to “ON”, not “off”
   4. Make sure that if the MetaMask extension suggests you a better way of estimating gas to accept it.
   5. Input a comparable or slightly higher gas limit than you did for the transaction that failed
   6. Raise the max priority fee at least 10% higher (in gwei) than the gas fee of the failed transaction
   7. Set your max fee at least 30% higher than that of the failed transaction.
6. In question 5 of this part, do front-end developers need to sign the transaction? Justify your answer with illustrative codes/figures.
7. In lab 4, we have worked on deploying smart contracts using the web3 library (so is the mini-lecture ML9 of lecture 6). Now, you are required to remind/research how to deploy a contract using ethers.js library only! Modify the code for the deploy button so that it does deploy the simple storage contract.
8. Based on your research and understanding of both ReactJS and Ethereum, state why is it important to store the deployed contract as a react state variable as opposed to a normal JS variable. Illustrate this using a modified version of the written code. (Please make a copy of the component so that it does not get damaged!)
9. Modify the code of “DeployContract” so that the deployed contract is stored as a React state variable.
10. Modify the code so that the “handlerClickSets” changes the existing value stored in the Simple Storage contract to an unsigned integer that is written in the textbox.
11. One of the things that front-end developers forget is to update the react state variable after interacting with the smart contract. Therefore, in your simple example, you might have coded to set the stored variable but you might have forgotten to update the react state variable where your contract is stored. If so, please modify the “handlerClickSets” so that it updates your contract state variable to include the recently engaged activity.