**Login App**

**[Problem Statement]**

Develop a pseudo Front End Application which would let the users login and display a message..

**[Minimum Requirement]** (entry level)

— Design (Try to replicate the wireframe as closely as possible. Ideally pixel perfect)

— Use of Web API to

* Authenticate an User <https://reqres.in/api/login>
* Pass two parameters in the (POST) request body i.e email and password.

"email": "[eve.holt@reqres.in](mailto:eve.holt@reqres.in)",

"password": "cityslicka"

After Successful User Login, In Api response you will be able to get user details including auth token. you will use this token for calling a secure API to get the list of users.

https://reqres.in/api/users

headers: {

'Authorization': 'Bearer <token>',

‘Content-type’:’application/json’

}

Print the list of users in console

* After login, Should give user friendly error messages (Invalid username /password) or Successfully logged in as Toast/Popup/Inline

**— Scaffolding** is available here, Fork the repository and start coding! All the best!

Prefer to use standard fetch rather than axios or other libraries as they sometimes have issues with stackblitz.

Angular - <https://stackblitz.com/edit/wissen-angular-coding-challenge-2022>

React - [https://stackblitz.com/edit/wissen-react-coding-challenge-2022](https://stackblitz.com/edit/wissen-react-coding-challenge-2022%C2%A0)

React Native - <https://stackblitz.com/edit/react-native-web-starter?file=index.js>

**[Plus Point]** (intermediate)

— Usage of libraries (redux, rx, bootstrap etc is allowed)

— Usage of local storage/session storage to auto login on a valid session and logout after 5 mins of non activity

— Unit test cases (jest/jasmine/framework of your choice)

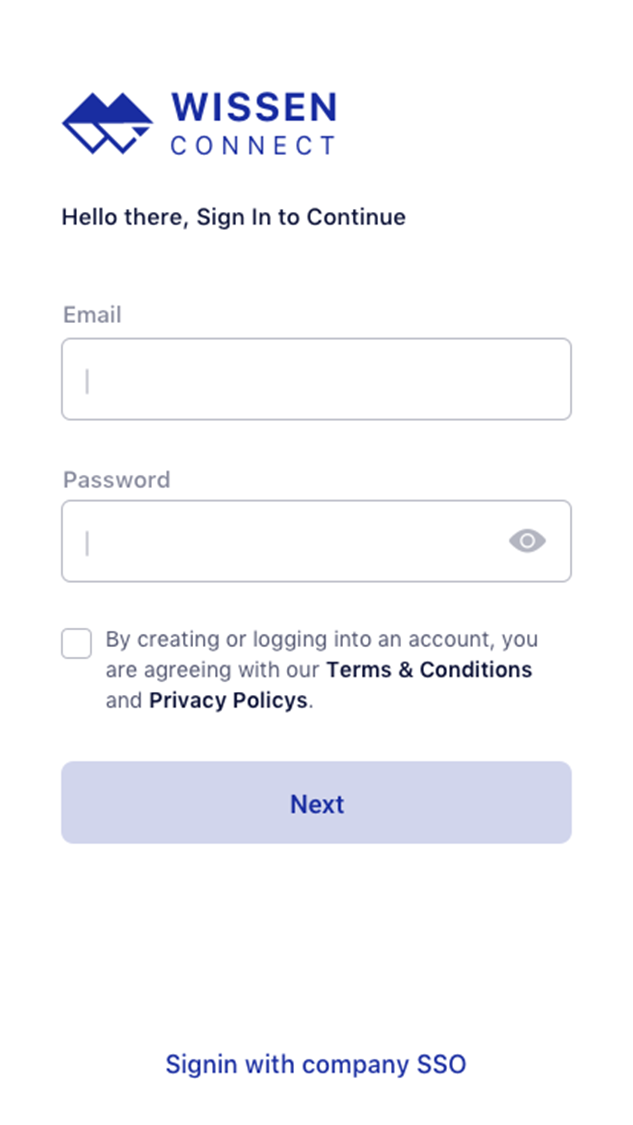
— Show table with list of users that you received instead of printing in console

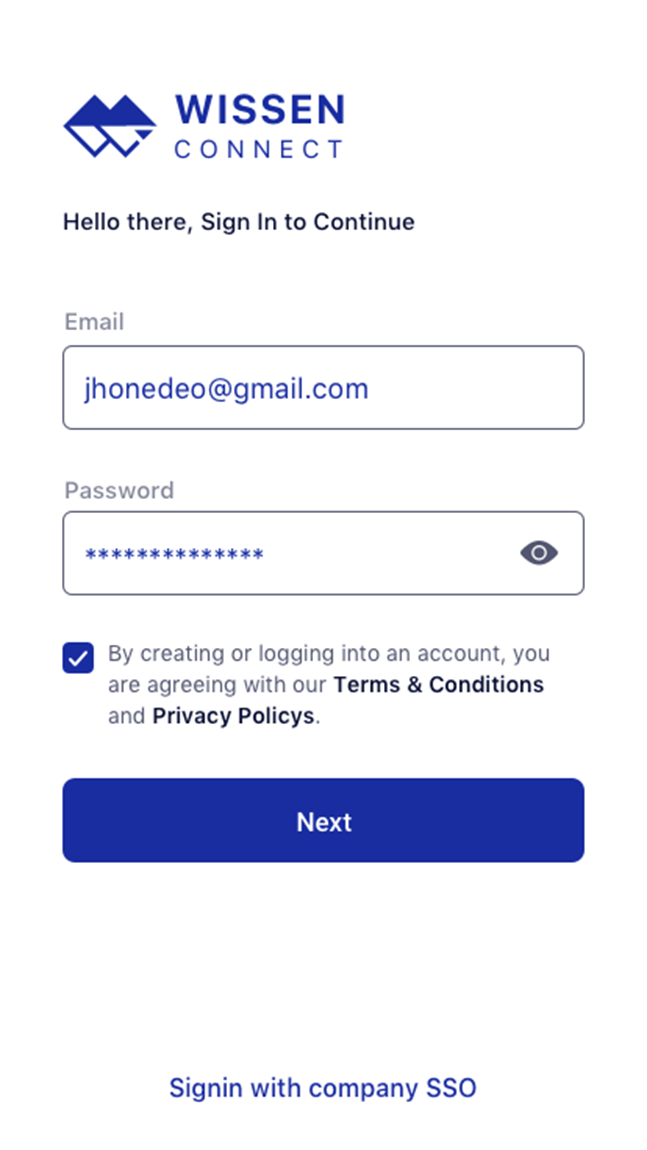
Logo is available here as image

<https://drive.google.com/uc?export=view&id=1hvRAGrdq0SqFBZApx2--IcuDf-DOmOBH>

Or svg

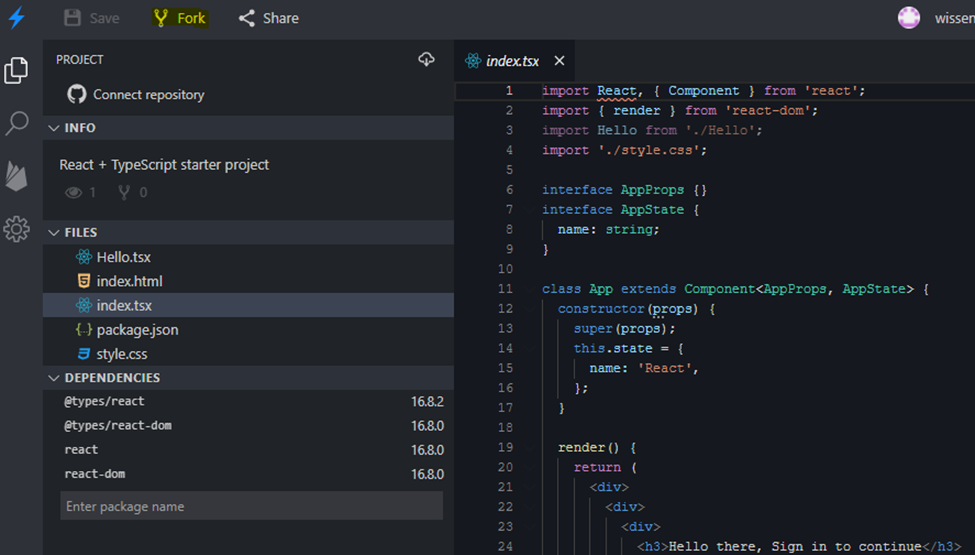
data:image/png;base64,



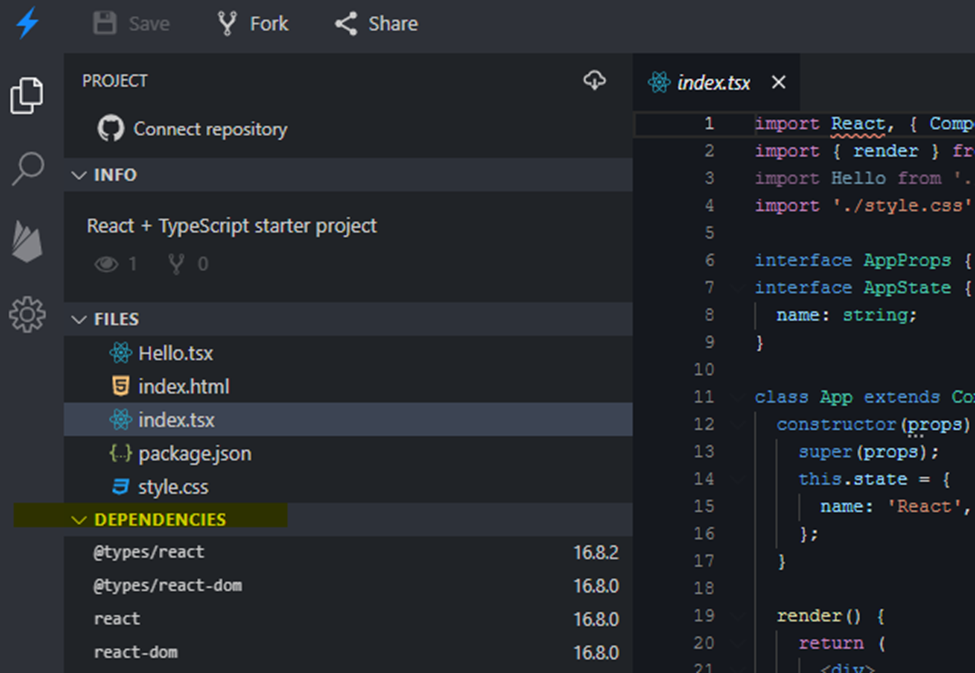


**Stackblitz**

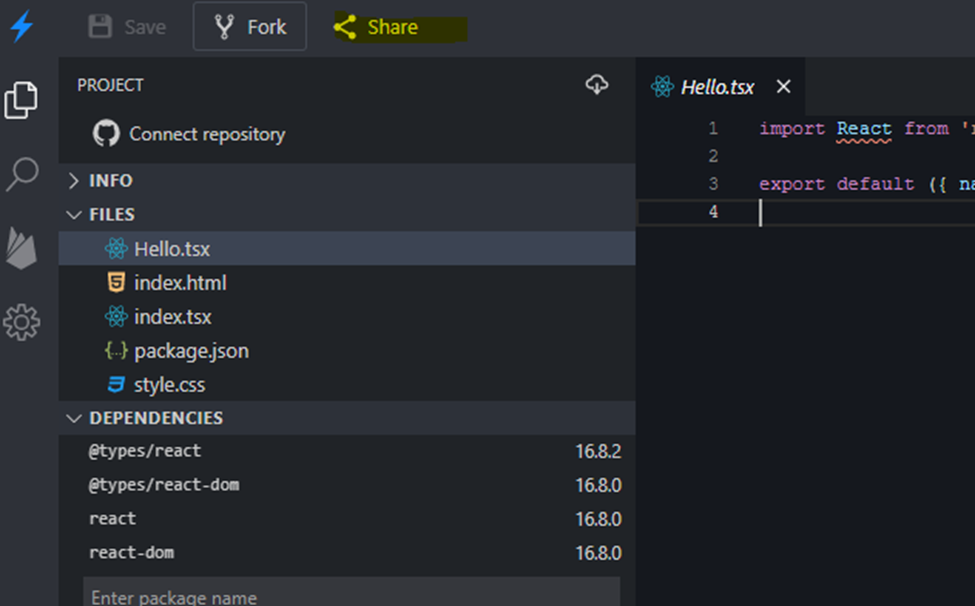
Fork the repository and proceed with solving the problem statement.



You can install optional dependencies to solve the problem statement by installing here.



Once you are done with the problem statement please share the url for review



**JS MCQ**

Copy the following questions into the separate file in the project called as js-mcq.js/js-mcq.ts and indicate your answers.

mcq (same in document and https://github.com/niteshjainy/wissen-ui-coding-challange/blob/main/src/ja-mcq.txt)

https://wissen-ui-coding-challange.vercel.app/

1. **What is the output of**

function f1() {  
 console.log(“this is”, this);  
 console.log(“arguments is”, arguments);  
 }  
 f1()

Choices are

1. Null, null
2. Undefined, undefined
3. **global, arguments**
4. f1, undefined

ANSHSWER: Ans is c. global, arguments

this inside the function refers to the global object (in a browser environment, it would be the Window object), and arguments refers to the arguments object associated with the function.

function addNumbers(a, b) {

console.log(arguments);

}

addNumbers(3, 5, 7);

The arguments object is a special object available within the scope of every function in JavaScript, and

it contains all the passed arguments when the function is invoked.

//an object that looks like an array but is not an array. It will contain the values [3, 5, 7].

arguments object is not a real array, so it doesn't have array methods like forEach or map, but it can be iterated through using a loop.

The arguments object allows you to access all the passed values, even if the function is defined with a specific number of parameters.

better recommended alternate:

function addNumbers(...numbers) {

console.log(numbers);

}

addNumbers(3, 5, 7);

//numbers is an array containing the values [3, 5, 7]

Arrow functions in JavaScript do not have their own arguments object. Unlike regular functions, arrow functions do not bind their own this, arguments, super, or new.target. Instead, they inherit these values from the enclosing lexical context.

If you need to access the arguments passed to an arrow function, you would typically use the rest parameter syntax (...).

const myArrowFunction = (...args) => {

console.log(args);

};

myArrowFunction(1, 2, 3);

**2) in the above function f1, I do the following**

let f11 = f1.bind({a:1}, 'abc');  
 f11('def');

What is the output now ?

Choices are

1. Null, null
2. **Undefined, undefined**
3. {a:1}, arguments = [“abc”, “def”]
4. {a: 1}, arguments = [“def”]

ANSWER: Ans is g.{a:1}, arguments = [“abc”, “def”]

**3) Write a simple function that will split an array into chunks with a specified size.**

Example - array = [1,2,3,4,5]

chunkSize = 1 output = [1] [2] [3] [4] [5]

chunkSize = 2 [1, 2] [3, 4] [5]

chunkSize = 3 [1, 2, 3] [4, 5]

chunkSize = 4 [1, 2, 3, 4] [5]

chunkSize = 5 [1, 2, 3, 4, 5]

chunkSize = 6 [1, 2, 3, 4, 5]

Function signature is

function (array, chunkSize) {

//TODO: implement this.

}

**DONE IN VS CODE**