React JS

العرض من React أو Angular أو Vue
إن إحنا نبني single page application (SPA)

1. functional component & jsx
2. event handling
3. state & props
4. lifecycle
5. styling
6. API
7. Form & Validation
8. Routing
9. Redux

دول عبارة عن الخلاصة، دول أهم concepts في الـ React واللي هتستخدمها طووول.

الممسوحة ضوئيا بـ CamScanner

main concepts
part ①

Session ①

1- introduction about React
2- Single page application
3- why React?
4- what is JSX?
5- building components in React.
6- component lifecycle methods
7- props and state
8- Handling Events

الممسوحة ضوئيا بـ CamScanner

① Introduction to React JS

Q) ايه هو الـ React ؟ الـ React هو عبارة عن library js بنستخدمه عشان نبني بيئة UI ،
الـ React بيركز على بناء الـ o/p اللي الـ end-user هيشوفه .

Q) ايه هو الفرق بين الـ framework وبين الـ library ؟

عشان اجاوب على السؤال ده محتاج اجاوب على سؤال تاني وهو: مين بيندة مين؟ who calls whome?

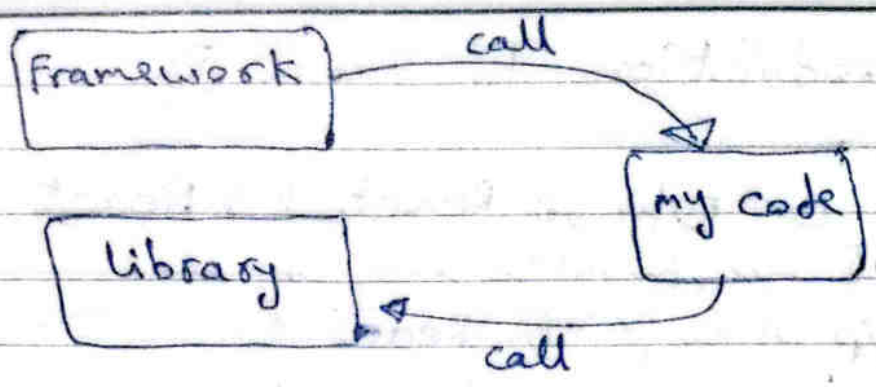
① في حالة الـ framework احنا بنكتب code ، والـ code اللي بنكتبه بيتحلل execute عن طريق الـ framework ، يعني الـ framework هو اللي بينادي على الـ code بتاعنا .

لكن في حالة الـ library ، الـ code بتاعي هو اللي بينادي على code اللي موجود في الـ library .

② في حالة الـ framework احنا بنلتزم بـ structure معين نمشي عليه وبنلتزم بـ built-in packages معينة نستخدمها .

لكن في حالة الـ library كل حاجة محتاجها بنعمل install بإيدينا ، وبنبني الـ structure اللي احنا عايزينه ، ممكن نعتبر الـ library هو حاجة فاضية احنا بنملاها .

③ الـ library ميقدرش يكون موقع كامل لوحده ، بيحتاج libraries تانية مساعدة ، على عكس الـ framework بيوفر كل حاجة .



* ال React هو library خفيف، حتى انه ليه CDN تقدر تستدعيه في كود ال html زيه اي ال jquery او ال bootstrap او ال fontawesome.

② ازاي نستخدم / ننزل ال setup ال React

عندنا طريقتين: ① يا إما ننزله locally بالجهاز بتاعنا.
② يا إما نستخدم ال CDN.

(* setup locally)
```
>> npx create-react-app hampozo ↵
>> cd hampozo ↵
>> npm start
```

③ ايه الفرق بين ال npm وال npx

① ال npm بيعمل install ال package بشكل global يعني على مستوى الجهاز كله، بالتالي احنا هنبقى ملزمين نفس نتعامل بنفس ال version في اي app جديد هنعمله.

② ال npx هو عبارة عن tool بتنزل مع ال npm وهو عبارة عن node package executer بيعمل execute ال package من غير ما يعمله install.

الممسوحة ضوئيا بـ CamScanner

(2) single page application

ايه هو ال SPA ؟ هو صفحة من غير refresh .

* ال SPA هو application عبارة عن صفحة واحدة اسمها index وبنعمل load فيها ال components اللي عايزين نعرضها لل user ، وال content بتاع صفحة ال index بيتغير dynamically على حسب ال component اللي ال user بيطلبه من غير ما يحصل reload لل page .

* ال SPA ك concept هو إن احنا نبعت كل صفحات الموقع من ال server لل client-side مرة واحدة من اول request بتبعت ، بالتالي لما نتنقل بين الصفحات مش هيحصل refresh وده أفضل ك UX .

ايه هو ال component ؟

ال component ك architecture هو إن احنا نقسم الموقع بتاعنا لأجزاء ، وكل جزء ليه ال html وال css والكود الخاص بيه ، ميزة ال component إنه :

- reusable اقدر اعيد استخدامه أكثر من مرة ، وبكتبه مرة واحدة .
- maintainable اقدر اعمل عليه والتعديل يبقى بسهولة .
- scalable اقدر اشتغل فيه زي ما أنا عايز .
- سهل تطبيق ال SPA عليه .

write once, use anytime & anywhere .

* اهم ميزة في ال component هو فكرة ال reusability .

③ why React?

١- ال React قايم على فكرة ال virtual DOM
وده بيخلي ال performance بتاع ال application .
ال React هي Just Javascript .♡♡
٢- ال React بيستخدم ال component based arch. .
٣- ال React بيسمح بإنشاء SPA .
٤- ال React بيسهلنا نبني reusable components .
٥- ال React ليه community ضخمة جداً .
٦- ال React معمول من شركة Facebook يعني مدعوم
وموثوق من شركة ضخمة زي ال Facebook .

٧- ال React هو declarative يعني قول لل react انت
عايز تعمل ايه وال react هيعمله بأبسط طريقة .

ايه هو ال virtual DOM ؟

ما هو إلا js object ، وهو عبارة عن نسخة lightweight
من ال real DOM .
ال virtual DOM كل إنه ياخد نسخة من ال DOM
اللي بتكون عليه ، وبمجرد ما تعمل اي update من ال
virtual DOM ، ال React هيبدأ يقارن بين ال
virtual DOM وال Real DOM ويعمل match للحاجات
اللي حصل تغيير فقط ، يعني مش هيعمل re render
ال DOM كله ، لكن هيعمل re render للحاجة اللي حصل
تغيير من ال DOM فقط ، وده هيحسن سرعة الموقع
جداً .

الممسوحة ضوئيا بـ CamScanner

(4) what is JSX? javascript xml

* الـ JSX ما هو إلا javascript شبه الـ html،
الغرض من الـ JSX هو تسهيل عملية إنشاء الـ html code
بالـ JS بحيث يكون كتابته شبه الـ html العادي،

* الـ JSX خلى الـ React يبقى declarative.
* الـ JSX من الآخر هو كل، يعني لازم محولة من شكل الـ
"html" لـ native js code عن طريق transpiler ينزل مع
الـ React.

without JSX: React.createElement('a', {href: '#'}, 'about')
with JSX: <a href="#">about</a>

أية هو الـ ReactDOM.render()؟

دة عبارة عن method جاهز تجيبه الـ React بنستخدمه
عشان نـ inject الـ JSX جوا الـ DOM.

(ex) ReactDOM.render(<h1>This is JSX</h1>,
document.getElementById("root"));

→ ReactDOM.render(JSX , DOM)

عايز تعرض أية؟ — عايز تعرضه فين؟

الممسوحة ضوئيا بـ CamScanner

ملحوظة: واحنا بنستخدم ال jsx ، لازم نتأكد إن احنا بنـ return فقط single parent element (.

يعني مينفعش نـ return أكتر من سطر html ، لازم نجعلهم في parent element واحد فقط.

نقدر نـ return ال elements جوا
أو جوا <></> وده بتسمية react fragment

ايه الفرق بين ال وال react fragment؟

- ال div بيرجع كـ dom element يعني بيشغل حيز في ال dom.

- ال fragment بيرجع ال element اللي جواه فقط يعني مش هيشغل حيز في ال dom.

(ex) ReactDOM.render(

```
(<h1></h1>
 <p></p>
 <a></a>)
```

✗

```
(<div>
  <h1></h1>
  <p></p>
  <a></a>
 </div>)
```

✓

```
(<>
  <h1></h1>
  <p></p>
  <a></a>
 </>)
```

✓

);

هنا مينفعش نكتب أكتر من سطر jsx من غير ما نكتبهم ما بين ( )

الممسوحة ضوئيا بـ CamScanner

⑤ build component :-

ازاي نعمل component في الـ react ؟

أولاً الـ jsx نقدر نخزنه في variable او نعمله return من خلال function ، لكن نحطه بين ( ) .

```
const jsx = (<h1></h1>);
function jsx () { return (<h1></h1>) }
```

* في الـ react نقدر نبني reusable component من خلال طريقتين :

① function component : وهو ان احنا نستخدم الـ concept بتاع الـ function js function بحيث إن الـ function بنقدر نعمله reuse أكثر من مرة ، عشان نحكي الـ function يبقى component :
١- نحكي اسم الـ function يبقى بـ pascal case .
٢- الـ function يرجع الـ return الـ JSX اللي عايزينه .
٣- الـ function نعمله call كـ html tag .

* الـ function component هو js function بيرجع الـ return الـ JSX .

② class component :
هو js class بيعمل inherit من class اسمه Component ويعمل implement لـ method اسمه render() بيرجع الـ return الـ jsx .

لازم اسم الـ function أو الـ class يكتب بـ PascalCase .

الممسوحة ضوئيا بـ CamScanner

إيه اللي بيميز ال class component عن ال function ؟

فيه ميزتين : ① بيدعم ال state يعني هو statefull
② ليه lifecycle و lifecycle hooks .

ايه هو ال React component ؟

هو عبارة عن function بيرجع react elements (UI) .

عشان نقدر نكتب JSX هنحتاج نعمل import
لحاجة اسمها React من " react " وإلا ال JSX
مش هيشتغل import React from 'react';
قبل الإصدار 17 كان لازم نعمل كده .

ال JSX هو syntax مش مفهوم ال browser بالتالي
لازم في مرحلة الproduction يتحول لـ native js code
ال browser يفهمه .
ال JSX هو بالـ development tools يعني بنستخدمه في
مرحلة ال development فقط .

ال create-react-app دة package فيه كل ال tools
وال configuration اللي هنحتاجها عشان نعمل ونشغل
react application .

ميزة ال function componen عن ال class component
إنه مش بيستخدم ال this keyword .

الممسوحة ضوئيا بـ CamScanner

⑥ component lifecycle :-

كل react component بيمر بـ ٣ مراحل :
مرحلة الـ mounting ثم الـ updating ثم الـ unmounting
وكل مرحلة بيجي فيل methods بيتعملهم invoke بشكل تلقائي .

① mounting

بيحصل في المرحلة دية calling لـ 3 methods بالترتيب :
1-) constructor()
2-) render()
3-) componentDidMount()

دية المرحلة اللي بيتعمل فيل initialization للـ component وحوا الـ dom .

② updating

0-) shouldComponentUpdate() ?
1-) render()
2-) componentDidUpdate()

دية المرحلة اللي بيحصل فيل تعديل في الـ component زي ان احنا نعمل :
new props
setState()
forceUpdate()
يعني بيحصل تغيير في الـ component فبالتالي لازم يحصل re-render .

shouldComponentUpdate() ?
هل لازم يحصل update ولا لا ؟
بيـ return يا true يا false
عبارة عن boolean function

③ unmounting

بيحصل calling لـ method واحدة اسمها .
componentWillUnmount()

دية المرحلة اللي هيتشال فيل الـ component من الـ dom .

الممسوحة ضوئيا بـ CamScanner

(7) props & state :-

ايه هو ال props ؟

* ال props عبارة عن parameter تبع ال function component
او عبارة عن property تبع ال class component
مهمتها إنه يستقبل object من ال data اللي بيتم ارسالها
للـ components من خلال ال parent component اللي جنبها

* ال data بتتبعت على شكل html attribute في ال html tag
بتاع ال component.
* ال props هي data جاية من برة ال component، بالتالي
من المنطقي انها تكون immutable او readonly يعني
مش قادر تغير ال value بتاعتها.

ايه هو ال state ؟

* ال state هو property خاص بال class Component.
* ال state هي data جاية من ال component نفسه،
يعني بتتعمل في ال component، بالتالي هي mutable
يعني قادر تغير ال value بتاعتها جوا ال component
اللي جنبها.
* بنستخدم method اسمه setState() عشان نـ update
ال object بتاع ال state.

- في النهاية ال props وال state ماهم إلا object في
- ال props بنعاملها معاملة ال html attribute.

الممسوحة ضوئيا بـ CamScanner

{stateless component هو function component يعني ايه ال

stateless component يعني component مفيهاش state object.

{state او props ليه نستخدم

① بنستخدم ال props عشان ننقل data من component ل component جواه، يعني عشان نعمل connection بين ال components.

② بنستخدم ال state عشان نخزن data تخص ال component ونقدر نتعامل change اثناء ال rendering على ال state object، يحصلة تغيير، هيحصل re-render ال component، وهيحصل execute ال componentDidUpdate() method.

③ الاثنين هما عبارة عن object زي.

④ ال props هو object بنبعته كـ parameter لل function component.

* افضل طريقة عشان ناخد ال value القديمة من ال state، ونعدل عليها وننشئ update القيمة الجديدة للـ state، هو إن احنا نستخدم callback function جوا ال setState()، وال function دي هيستقبل ال state القديمة كـ parameter ويرجع القيمة الجديدة.

```
this.setState((oldstateValue) => {
        return ({
                })
    });
```


الممسوحة ضوئيا بـ CamScanner

ملاحظة

لو هتحتاج القيمة القديمة لل state عشان تحدد بيها القيمة الجديدة لل state ، فإنك ملزم تبعتلي callback function كـ parameter لل state setter method بدلاً من إنك تستخدم ال state في طول.

ال callback function دي هتستقبل القيمة القديمة لل state كـ parameter ليها ، وال function من خلالها تقدر تحدد القيمة الجديدة لل state ، وال callback fun() هي return القيمة الجديدة ويحطها في ال state setter method .

ليه ابعت callback function كـ parameter لل state setter method ؟

لو هتحتاج القيمة القديمة لل state عشان احدد بيها القيمة الجديدة لل state :

ال props هو object جاي من برا ال component ، ال state هو object متحول في ال component نفسه :

(ملحوظة) في ال JSX لو هنعمل empty tag ، لازم يتقفل بـ (/) وإلا هيعتبرها nested tag .

(ex) <hr> ✗
<hr /> ✓

الممسوحة ضوئيا بـ CamScanner

(8) Handling Events :-

* عن طريق الـ event نقدر نخلي الـ user يـ interact مع الـ application بتاعنا.
* لما user يعمل interact يعني فية event معين حصلة fire.

لو من الـ react ، الـ function اللي بيـ fire لما الـ event يحصل بتسمية (Handler).

اية هو الـ handler ؟ هو الـ function اللي هيشتغل لما الـ event يحصل.

اية صيغة الـ React events ؟

react.Event = {eventHandler}

- الـ react Events بتكتب باستخدام الـ camelCase.
- الـ Handler بتتعمل call من غير الـ call operator وإلا هيحصلة execute بشكل تلقائي من غير ما الـ event يحصل.

- الـ Handler تمكن نعمل call operator عشان نبعتلة parameter لكن لازم يكون جوا fat arrow function.

event = {() => handler()}

- الـ Handler ممكن يكون function مبعوت كـ props من خارج الـ component.
- لو الـ Handler بيحتوي على this Keyword ، قيمة الـ this هتبقى undefined لما الـ event يحصل ؟ ليه ؟؟!! دة لان احنا ساوينا الـ handler بالـ event يعني بعتنا regular function عادية وبتتعمل call من الـ global object.

(ex)
```
test(){
    console.log(this)
}

<button onClick={this.test}> click</button>
```

ليه قيمة الـ this بـ {undefined} و ايه الحل؟

دة لأن الـ method اللي بتستدعي الـ this لما بساويناه بالـ onClick
بيعتبر regular function عادية، واللي هيعمل call للـ method
هيكون الـ global object، وإضافي حالة الـ strictmode
فبالتالي قيمة الـ this هتبقى undefined.

إيه الحل؟ فيه ٣ حلول للمشكلة دي:

- أول حل إن احنا نـ bind الـ this مع الـ function
يتحطلي execute.
  * onClick={this.test.bind(this)}

- ثاني حل إن احنا نستخدم الـ fat arrow function يعني بدل
ما نـ execute الـ handler مباشرة، هنـ execute الـ arrow.
  * onClick={() => this.test()}

- ثالث حل إن احنا نـ bind من الـ constructor من البداية
يعني هنـ bind الـ this بالـ method من الـ constructor، ودة
الـ best practice.

```
constructor(){
    this.test = this.test.bind(this)
}
```


الممسوحة ضوئيا بـ CamScanner

ايه هو الـ {} curly bracks JSX ؟

الـ {} هو عبارة عن syntax بيسمحلنا نكتب js logic .
او نطلب variable او نكتب js expression جوا الـ (JSX) .

الـ {} هو special syntax في الـ JSX بنستخدمه عشان
نـ insert js expression جوا الـ JSX ، الـ js expression
ممكن يكون variable او function او object او اي code
هيرجع بـ value .

- عن طريق الـ {} نقدر نترجم dynamic data .
- عن طريق الـ {} نقدر ندخل dynamic content في الـ JSX .

- الـ JSX هو tool بيسهل انشاء الـ html بالـ js بحيث يكون شبه
الـ html العادي لكن في النهاية هو code js لكن بشكل انضف .

ازاي نبعت data من الـ parent component للـ child ؟

عن طريق ان احنا ندخل function في الـ parent بتتعمل
execute في الـ child وبيستلم الـ data من الـ child في
parameter .

يعني هنبعت الـ data من الـ child كـ parameter لـ function
في الـ parent . ازاي ؟؟! عن طريق الـ props والـ event .

① هنستخدم الـ props عشان نبعت الـ function من الـ parent
للـ child .

② هنستخدم الـ event handler في الـ child عشان يـ execute
الـ function اللي هيستلمها من الـ props .

الممسوحة ضوئيا بـ CamScanner

- ال JSX هو javascript xml ؟
ليه xml مش html ؟

- لان ال html عبارة عن tags ثابتة معروفة مقدرش اعمل tag من دماغي.
لكن في ال xml تقدر تعمل tag باي اسم انا عايزه.

(ex) <hampoze />

- ايه الميزة من استخدام ال JSX ؟
ليه JSX ؟

① ال JSX بيمكنك / بيخليك تقدر تكتب كود html جوا كود javascript.
② ال JSX بيخليك تألف tag من دماغك.

- ال tag اللي بنعمله بيه ال component بنقول عليه ال selector tag

- ال attribute اللي بنحطه في ال JSX بنقول عليه props.
بيتعامل كـ parameter في حالة ال function component
وكـ property في حالة ال class component.

ال prop بنكتبه في ال selector بتاع ال component
وبنستلمه في ال component كـ parameter او property.

الممسوحة ضوئيا بـ CamScanner

Q) ليه اسم ال component لازم يبدأ بحرف capital؟

A: عشان ال react يميز ال component ومحدش
تتعارض مع html tag ليه نفس الاسم.

ex) <Input /> → component
<input /> → html tag

ال this Keyword جوا ال component بتعود على أو
بتشاور على ال component اللي هي فيه.

ال state بنستعمله ب this جوا ال constructor، لكن
برة ال constructor من غير this عادي.

ex) state = { } ✓
constructor(){
this.state = { } ✓
}

ليه فكرة ال super() { super() فكرة ال super() إنه بيعمل inherit
كل حاجة موجودة في ال parent component.

ليه فكرة ال render() ؟ بيعمل return ال html اللي عايز اعرضه.

ليه هو ال fragment ؟ ما هو إلا parent وهي مش بتعمله
render في ال dom.

ex) <></>
أو <Fragment></Fragment>

الممسوحة ضوئيا بـ CamScanner

main concepts
part ②

Session ②

1- images
2- revision
3- deal with style
4- List and Keys
5- Conditional Rendering.
6- Forms & form Validation

الممسوحة ضوئيا بـ CamScanner

ازاى تستخدم image فى ال react ؟ (1)

فيه طريقتين للتعامل مع الصور أو ال static files :

أول طريقة :- نحط ال static files جوا ال public folder
ونروح ال component نعمل <img />، ونجرب ما
نحط (/) فى ال src، وكده احنا جوا ال public.

(ex)

```
public
  |- images
      |- 1.png
```

```
<img src="/images/1.png" />
```

ثانى طريقة :- ال best practice إن احنا نحط الصور جوا
folder نسميه assets جوا ال src folder
لكن طريقة الاستدعاء هتبقى مختلفة.
طالما الصور جوا ال src folder، هنتعامل
معاملة ال import.

(ex)

```
src
  |- assets
      |- 1.png
```

```
import blZ from "./assets/1.png"
```

هياخد الصورة ويخزنها فى
المتغير ده،

```
<img src={blZ} />
```

عيب الطريقة دى ان كل صورة محتاجها لازم أعمل import.

الممسوحة ضوئيا بـ CamScanner

Q) امتى نستخدم الـ class based component وامتى نستخدم الـ function component ؟

لو الـ component اللى هنعمله هيبقى فيه logic هنحتاج فيه state و update و lifecycle فلازم هنستخدم الـ class based component.

لكن لو الـ component مجرد إنه هيرجع html فهنستخدم الـ function based component وخلاص.

ملحوظة: الإصدار الجديد للـ React من 2018 ظهر concept جديد اسمه Hooks ودة خلى كل مميزات الـ class component تبقى موجودة من الـ function component زى الـ state والـ lifecycle مع ميزة إن الـ function مبيستخدمش الـ this.

يعنى حاليًا بعد 2018 اصبح يفضل نستخدم الـ function component دايمًا.
طب ليه بنتعلم الـ class components ؟!!
لأن لسه فيه شركات بتستخدمه وفيه مشاريع قديمة محولتش بيه، فلازم نتعرف.

ليه نستخدم الـ props ؟ علشان نجيب data من برة الـ component.
ليه نستخدم الـ state ؟ علشان نشغل private data خاصة بالـ component ونقدر نشغل بيه الـ update cycle.

الممسوحة ضوئيا بـ CamScanner

② Revision

Q) ايه هو الـ react وليه بنستخدمه؟

هي عبارة عن js library بتبني بيها UI ، الفرق بين استخدام الـ react وبين استخدام الطرق العادية هو إن الـ react بيبني SPA وبيعتمد فكرة الـ component وهو تقسيم الموقع لأجزاء بالتالي الكود هيكون أكثر تنظيماً و reusable .

Q) ايه هو الـ virtual Dom؟

عبارة عن نسخة وهمية من الـ real Dom ميزته إنه بخلي الـ react يعمل re render للأجزاء اللي حصل update في الـ dom فقط وليس الـ dom كله ، ودة هيحسن الـ performance (لأن مش هيحصل refresh للموقع) اكاجة اللي حصل تغيير في اللي هتعمل render من جديد مش لموقع كله .

Q) ايه هو الـ SPA؟

عبارة عن صفحة html واحدة ، والـ content اللي جواها بيتغير dynamically من غير ما يحصل refresh للـ page ، بنستخدم فكرة الـ component عشان ننفذ الـ SPA .

Q) ايه هو الـ props والـ state؟

الـ props هي data بتنتقل من الـ parent component للـ child component ، والـ props بتعامل معاملة الـ html attribute .

الـ state هي data خاصة بالـ component مش قادرة توصل برة الـ component ، ولما يحصل update لقيمة الـ state ، الـ component هيعمله rerender .

الممسوحة ضوئيا بـ CamScanner

Q) ايه افضل طريقة لتقسيم مشروع الـ react؟
١) نعمل folder اسمه components هنحط فيه كل الـ component اللي هنستعملها.
٢) كل component هنعمله folder لوحده باسمه
٣) كل component هيحتوي على ملف الـ css والـ js تبعه.

(ex)
src
- components
  - Users
    - Users.css
    - Users.js

Q) ليه الـ function / class component بنستعملهم export default مش export فقط؟
← ده لأن معنيش غيرهم في الملف بيتعمله export.
default يعني معنيش ميزه / لوحده، الكود الوحيد اللي هيتعمله export.

Q) ايه فايده الـ jsx؟
بيسهلنا نكتب كود html في ملف الـ js، وبيسهلنا نكتب او نالف html tag من دماغنا.

Q) ايه ميزه الـ class component؟
(١) statefull يعني بيحتوي على state.
(٢) ليه lifecycle و lifecycle hooks.
ميزه الـ function component؟
هيستخدمش الـ this keyword.

الممسوحة ضوئيا بـ CamScanner

Q) ليه اول حرف من اسم الـ component لازم يبقى كبير؟
عشان الـ react يقدر يميزه كـ component مش كـ html tag.
وعشان ميحصلش تعارض ما بين اسم الـ component وما بين اسم tag موجود من الـ html.

Q) ايه الفرق بين الـ framework والـ library؟
١- عشان اجاوب على السؤال ده محتاج اجاوب على سؤال تاني وهو مين بيندة على مين؟
الـ framework هو اللي بيعمل call للكود بتاعي وهو اللي بيشغله execute
لكن الـ library بيتشغله call من الكود بتاعي.

٢- الـ framework بيلزمني ب structure معين امشي عليه وبيلزمني ب package جاهزة تكون موجودة حتى لو مش هستخدمها.

٣- لكن الـ library هو حاجة فاضية وانا بعملها، هو ملف واحد بيعمل شغل معين ولو احتجت حاجة تانية بعمل install.

٤- الـ library مبيقدرش يقوم موقع كامل لوحده، هيحتاج libraries تانية معاه.
لكن الـ framework فيه كل حاجة بيعمل يقدر يقوم موقع كامل لوحده.

الممسوحة ضوئيا بـ CamScanner

(3) deal with css/sass

Q) ازاى نعمل link بين ملف ال css وال component؟

عن طريق ان حنا نعمل import ملف ال css جوا ملف ال component.

(ex) components
- Test
  - Test.js → import "./Test.css"
  - Test.css

- يفضل كل component يكون ليه ملف ال css بتاعه.
- يفضل اى style نعمله يحتوى على ال selector بتاعه هو اسم ال component لو اى اسم يميز انه تبع ال component عشان ميحصلش تعارض بينه وبين style تبع component تانى.

Q) ازاى نستخدم ال sass فى ال react؟

(1) اول خطوه نعمل install لل sass لو مش معمولة install.

(2) تانى خطوه نعمل import لملف ال .scss جوا ملف ال component بتاع ال component.

(1) npm install sass --save-dev
(2) import "./Test.scss

الممسوحة ضوئيا بـ CamScanner

④ List & Keys

ايه هو الـ List ؟ الـ list عبارة عن مجموعة من الـ elements
في الـ js بنستخدم الـ array عشان نعمل list.

ايه هو الـ map() في الـ js ؟
هو عبارة عن method تبع الـ array object بنستخدمه
عشان نعمل تعديل معين على كل الـ elements في الـ array.

الـ map() method بيعمل return لـ array جديدة عبارة عن
الـ array القديمة لكن متعدلة.

ايه ذكرى الـ list في الـ React ؟

الذكرى هو إن احنا نحول الـ array لـ list of jsx elements
بالتالي نقدر نعمل render لـ data اللي في الـ array.
الفايدة إن احنا هنكرر jsx element معين وفي كل مرة هنحط
فيه data مختلفة.

ايه هو الـ Key وايه فايدته ؟

الـ Key هو عبارة عن special attribute لازم نحطه
في كل jsx element في الـ list، دايماً هنستخدم الـ Key
ليه نعمل list of jsx element في الـ React ؟؟!
لأن الـ React هيعتمد على قيمة الـ Key عشان يتابع ايه هو
الـ element اللي حصلة تغيير بالتالي هيعمله re-render من
غير مايعمل re-render لكل الـ list.

الممسوحة ضوئيا بـ CamScanner

① without Key :-

```
<ul>                          <ul>
    <li> A </li>   <-update-   <li> A </li>
    <li> B </li>   <-update-   <li> B </li>
</ul>              <-update-   <li> C </li>
                              </ul>
```

```
<ul>                          <ul>
    <li> ~~A~~ C </li>  <-update-   <li> C </li>
    <li> ~~B~~ A </li>  <-update-   <li> A </li>
</ul>               <-update-   <li> B </li>
                              </ul>
```

② using Key :-

```
<ul>                          <ul>
    ~~<li>~~ 1          <-update-   <li Key="3"> C </li>
    <li Key="1"> A </li>            <li Key="1"> A </li>
    <li Key="2"> B </li>            <li Key="2"> B </li>
</ul>                             </ul>
```

- ال Key هو special string attribute
- قيمة ال Key تتحول string.
- لازم قيمة ال Key تكون unique.
- يفضل نستخدم ال index ال loop كقيمة لل Key.
- استخدام ال Key هيزود ال performance

الممسوحة ضوئيا بـ CamScanner

(example)

```
const products = [{},{}, ...];
const listItems = products.map((product, index) =>
    {
        return ( <div key={index}>
                 ≡
                 </div> );
    });
render(){
return ( <>
        {listItems}
        </> );
}
```

امتى نستخدم الـ React list ؟

(1) لما نحوز نعرض data جوا list / array ، مثال لما نحوز نعرض الـ data الـ json الـ جاية من الـ API.

(2) React list بيعمل هنا تكرار لـ jsx معين بـ data مختلفة في كل مرة.

(3) نحول الـ array لـ list of jsx element عشان نقدر نعمل rendering في الـ dom.

← هنستخدم الـ map() عشان نعمل rendering لـ array of data في الـ React

الممسوحة ضوئيا بـ CamScanner

⑤ Conditional Rendering :-

* ال conditional rendering هو إن احنا نشكل render ل jsx element بناءاً على condition معين.

① using conditional operator:

→ condition ? true : false

(ex)→ { condition? (<> </>) : null }

② using logical && operator:

→ condition && expression/js code

(ex)→ { condition && (<> </>) }

③ using if-else statement :-

```
(ex)→ if( condition){
        return (<> </>);
    }else{
        return (<> </>)
    }
```

الممسوحة ضوئيا بـ CamScanner

⑥ Forms

* بنحتاج ال form عشان ناخد data من ال user.
يعني نخلي ال user يدخلنا data.

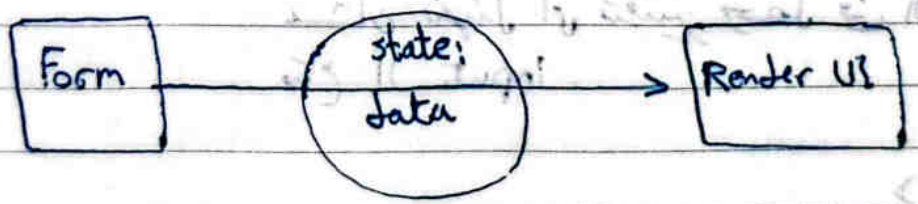
Q) ايه هو ال controlled component / uncontrolled؟
ايه الفرق بينهم؟

---

* ال react بيتعامل مع ال form بطريقتين:
حاجة اسمها controlled component ← ال react state هو المتحكم
وحاجة اسمها uncontrolled component ← ال dom هو المتحكم

* ال uncontrolled component هو ان احنا نهندل ال data بتاعة ال form من خلال ال DOM بعيداً عن ال React، يعني ال React مش متحكم في أي تغيير يحصل في ال form.

* ال controlled component هو ان احنا نهندل ال data بتاعة ال form من طريق ال React state يعني ال React هو المتحكم في أي تغيير يحصل في ال form.



① controlled يعني بنهندل او بنتحكم في ال form data من خلال ال React state.
② uncontrolled يعني بنهندل ال form data من خلال ال dom.
يعني ال dom هو اللي بيتحكم في ال form data.

Q) ازاى تتعامل form component فى الـ React؟

← اولاً هنستخدم طريقة الـ controlled component
يعنى هنخزن الـ form data فى الـ react state.

① اول خطوة: هنجهز الـ state object اللى هنخزن فيه الـ form data، بحيث كل property فى الـ state هيختص بـ field معين.

```
state = {
    username = "",
    email = ""
}
```

اى update هيحصل للـ state، هيحصل rerender للـ component

② تانى خطوة: هنجهز الـ html form مع مراعاة حاجتين:

١- كل input field يفضل يحتوى على name attribute وقيمته هتكون اسم الـ property الخاص باستقبال قيمة الـ input ده فى الـ state.

٢- كل input field هيحتوى على onChange event عشان نهندل اى تغيير يحصل فى الـ value بتاع الـ input.

```
<form>
    <input type="text" name="username" onChange={}/>
    <input type="email" name="email" onChange={}/>
</form>
```


الممسوحة ضوئيا بـ CamScanner

(٣) ثالث خطوة :- هنعمل method يهندل الـ change
اللي بيحصل في الـ value بتاع الـ input
وبناءً عليه هنعمل update لـ state .

```
handleChange = (event) => {
    this.setState({
        [event.target.name] : event.target.value
    });
}

onChange = { (e) => this.handleChange(e) }
```

(٤) رابع خطوة :- هنعمل method يهندل الـ form submit
مع مراعاة حاجتين :
١) هيفضل نستخدم الـ onSubmit event
بدلاً من الـ onClick .
٢) لازم نلغي/نعمل prevent لـ
الـ defaults behavior الخاصة بالـ form
زي انه بيعمل reload لـ page وده
ضد فكرة الـ SPA .

```
handleSubmit = (e) => {
    e.preventDefault();

}

<form onSubmit = {(e) => this.handleSubmit(e)} >
<input type = "submit" />
```


الممسوحة ضوئيا بـ CamScanner

Q) ازاى نضيف الـ validation form
ازاى نعمل validation من الـ client side
ازاى نعمل validation من الـ React .

افضل واسهل طريقة لعمل الـ validation هو
استخدام الـ Regex .

* اول خطوة :- هنضيف fields جديدة فى الـ state object
هنستقبل فيهم الـ error message .

```
state = {
    name : '',
    email : '',
    nameError : '',
    emailError : '',
}
```

---

* ثانى خطوة :- هنجهز الـ RegEx اللى هنعمل بيها test
الـ validation .

```
let validName = new RegExp('^\w\d $');
let validEmail = new RegExp('[illegible]');
```

---

* خامس خطوة :- اعرض الـ error message

```
{this.state.nameError ? (<p></p>) : null}
<p> {condition ?  : null } </p>
```


الممسوحة ضوئيا بـ CamScanner

ثالث خطوة :- هنجهز ال function اللي هتنفذ ال validation من كل مرة بتعمل فيها submit.

```
FormValidation = ()=>{
    let nameError = '';
    let emailError = '';
    let isValid = true;
    if(!validName.test(this.state.name)){
        nameError = 'name is not valid';
        isValid = false;
    }
    if(!validEmail.test(this.state.email)){
        emailError = 'email is not valid';
        isValid = false;
    }
    this.setState({nameError, emailError});
    return isValid;
}
```

رابع خطوة :- اعمل ال invoke لل validation function في ال submit handler أو المكان اللي انت عايزة فيه run فيه.

```
handleSubmit = (e)=>{
    e.preventDefault();
    let isValid = this.FormValidation();
    if(isValid){
        =
    }
}
```

Q) ازاى نعمل reset ال Form ؟

افضل حل هو إن احنا نربط ال value بتاع ال state بال value بتاع ال input field ، ودة مش هيسبب مشكلة لانه اصلاً اى تغييرات قيمة ال state هيحصل rerendering .

```
① initialstate = {
      name : '',
}

② state = this.initialstate;

③ <input value = { this.state.name } />

④ handleSubmit = (e) => {
      ≡
      this.setState(this.initialstate);
}
```

هنخلى ال value بتاعة ال input تتحدد على ال value فى ال state

١) هنعمل variable نحط فيه ال object بتاع ال state .
٢) هنساوى ال state بال variable ده .
٣) هنربط ال state بال input's value .
٤) لما يحصل submit هنعمل reset يعنى هنفضى ال form عن طريق إن احنا نفضى ال state وبالتالى هيحصل rerendering وال input fields هيفضوا .

الممسوحة ضوئيا بـ CamScanner

Q) ايه هو الفرق بين ال React element
وال React Component ؟

ال React element هو js object بيمثل node معين
من ال dom.

ال React component هو js function بيـ return
ال React elements.

الممسوحة ضوئيا بـ CamScanner

advanced concepts

Session ③

1- conditional styling in React.
2- working with API in React.
3- use axios for API calls.
4- React Routing
5- sharing data b/w components.
6- diff ways to add styling in React

الممسوحة ضوئيا بـ CamScanner

① conditional styling

ق) ازاى نضيف inline css style معين او css class معين للـ element بناءا" على condition ؟

```
1- className = { condition ? 'p-5' : null }

2- style = { condition ? {color: 'red', padding: '5px'} : {} }

3- style = {{ color: condition? 'red' : 'blue',
              padding : condition? '10 px' : '0px',
              }}

4- className = {`p-5 ${condition? '' : ''}`}
```

الممسوحة ضوئيا بـ CamScanner

② dealing with API

Q) ازاى نعمل fetch للـ data من API ؟
Q) ازاى نتعامل مع الـ server ؟

افضل مكان نعمل فيه الـ call api هو لما يحصل mounting للـ component يعنى لما الـ component يحصل فى الـ dom ، بالتالى يفضل نعمل الـ call api فى الـ componentDidMount() بحيث نضمن إن الـ component اتعمل Mounting.

اول خطوة :- جهز الـ state اللى هتستقبل فيه الـ data الـ json .
وجهز flag بتأكد بيه إن الـ data وصلت .

```
state = {
    isloading : true,
    data : null
}
```

ثانى خطوة :- استخدم الـ fetch() js للتعامل مع الـ request
واذكر إنه بيرجع promise .

```
async componentDidMount(){
    let url = "————————";
    let response = await fetch(url);
    let data = await response.json();
    this.setState({isloading: false , data});
}
```


الممسوحة ضوئيا بـ CamScanner

ثالث خطوة :- نعمل rendering condition نتأكد هل الـ data رجعت ولا لأ .

```
{ this.state.isLoading || !this.state.data ?
    (<div> — </div>) :
    (<div> — </div>)
}
```

* ممكن نستخدم .then() بدلاً من async & await .

```
componentDidMount (){
    let url = " — ";
    fetch(url)
    .then((res) => res.json())
    .then((data) => { — });
}
```

ايه هي عيوب الـ fetch() ؟

١) بيرجع الـ response على شكل binary format بالتالي لازم نحوله لـ json format .

٢) مش بيرسل الـ errors لوحده ، بيحتاج XHR object خارجي للتعامل مع الـ errors .

الممسوحة ضوئيا بـ CamScanner

③ Axios (3rd party library)

Q) ايه هو ال Axios ؟

هي عبارة عن 3rd party js library بيستخدم للتعامل مع ال request ، بديل للـ fetch().

Q) ايه ميزه ال Axios ؟

(١) بيرجع ال response data على شكل JSON على طول.
(٢) ال axios هو library مستقل بذاته.
(٣) ال axios مناسب لل browsers القديمة والجديدة.
(٤) ال axios بيوفر security
(٥) ال axios بيهندل ال errors بشكل أفضل.
(٦) ال axios بيسمحلك تـ cancle ال request.
(٧) ال axios يقدر يهندل أكثر من request في نفس الوقت.

Q) إزاي بستخدمه ؟

```
1- npm i axios --save
2- import axios from 'axios';
3- axios.get(url)                  axios.post(url, data)
   .then( res => { } )             .then( res => { } )
   .catch( err => { } )            .catch( err => { } );
   .finally( () => { } );
```

الممسوحة ضوئيا بـ CamScanner

④ React Routing

Q) ايه هو ال Routing ؟

فكرة ال Routing إن إحنا ننتقل بين ال components يعني نتحكم في ال component اللي بيتعرض بناءاً على ال url.pattern في ال browser من غير ما يحصل reload يعني من غير ما نكلم ال server-side .

Q) ايه هو ال browser history stack ؟

- هي عبارة عن حاجة بتسجل كل خطواتك على ال website اللي انت فاتحه . انت جاي منين؟ انت فاتح ايه ؟

- بيسجل ال urls اللي انت زرتها .

/about	← ثالث حاجة زرتها
/home	← ثاني حاجة زرتها
/	← اول حاجة اتفتحت

- ال history stack مش خاص بال react ، هو موجود اصلاً في ال JS .
- هنستخدم مع ال react routing عشان لما ال user يعمل back او forward ، يحصل ال routing لل component الصح .

Q) ايه هو ال object في ال JS اللي بنستعمله عشان navigation ؟

"location" object

Q) ايه هو ال object في ال JS اللي بيسهلنا نستخدم ال history stack ؟

"history" object

Q) ايه هو ال 3rd party library اللي هنستخدمه لل Routing ؟

react-router-dom

الممسوحة ضوئيا بـ CamScanner

Q) ايه هما الـ Basic Routing Components ؟

١- BrowserRouter : هيحتوى كل الـ components في الـ app.
← ده عبارة عن الـ parent component اللي هيشيل لو هيحتوى كل الـ components اللي هتستخدم routing.

← اهميته انه هيخلي الـ browser يهندل الـ routing اللي هنعمله.
← الـ component ده هننشاه جواه الـ routes بتوعنا.

٢- Routes :
← عبارة عن parent component لكل الـ Route اللي هنعملها.
→ Routes is set of Route.

٣- Route :
← ده اهم component في الـ React Router.
← الـ component ده هيـ render محتوى معين لو الـ path في الـ Browser url توافق مع الـ path بتاع الـ Route.

← كل Route ليه path و element هيـ render لما الـ path بتاع الـ Route يتنده عليه.

٤- Link :
ده عبارة عن component بديل لـ anchor tag بيسمحلنا نعمل navigation من غير reload.

الممسوحة ضوئيا بـ CamScanner

@) ازاى نستخدم الـ React Router ؟

```
1- npm i react-router-dom@6.4
2- import {BrowserRouter as Router, Routes, Route, Link}
   from 'react-router-dom';
```

aliasing

3- <BrowserRouter> ← كلمة application ، نخاوط بيه الـ

```
<Routes>
    <Route path="/" element={<h1></h1>} />
    <Route index element={} />
    <Route path="*" element={} />
</Routes>
<Link to="/" >Home</Link>
<NavLink to="/" >Home</NavLink>
</BrowserRouter>
```

index route

wild card route

@) ايه الفرق بين <Link> و <NavLink> ؟

الفرق إن الـ <NavLink> لما بيتفعل أو لما بيضغط عليه بيضيف لنفسه class اسمه "active" وده هيفيدنا لو عزنا نعمل style الـ activated لينك.

الـ element property بيقدر يـ render الـ markup بس طول ما يعني هو بيـ render أى JSX جواه مش الـ React component فقط.

الممسوحة ضوئيا بـ CamScanner

Q) امتى نستخدم ال exact prop ؟

لو فيه أكتر من Route بيحتووا على نفس ال path ، بنستخدم exact عشان نمنع أي override أو أي conflict يحصل .

Q) ازاي نعمل nested Route ؟

- nested يعني جاية جوا جاية .
- هنعمل child Route جوا ال parent Route مع مراعاة إن ال path بتاع ال child Route مبتبدأش بـ (/)

- بنستخدم ال Outlet component عشان تحدد المكان اللي هيعمل render فيه ال nested Route في ال parent Route , اللي هو فيه .

```
① <Route path="parent" element={} >
      <Route path="C1" element={} />
      <Route path="C2" element={} />
  </Route>
```

```
② import {Outlet} from 'react-router-dom';
③ <Outlet />
```

- لو ال Route بتاع nested Route جوا Route تاني ، فال nested Route دي ال component اللي فيه هتعمل render جوا ال component الخاص بال parent Route اللي شايلة .

الممسوحة ضوئيا بـ CamScanner

لو فيه Route جوا Route ، فال Route الداخلي محتواه
هيحل من ال Route الخارجي ، ومشان نحدد المنطقة اللي
هيحل فيه هنستخدم ال <Outlet />.

ال <Outlet /> هو المكان اللي عايز تحل فيه ال nested Route ،
داخل ال parent Route.

Q) ازاي نعمل Route parameter ؟
Q) ازاي نقرأ parameter من ال url ؟

```
<Route path="/product/:id" />
```

اسم ال parameter.

- فايدة ال parameter ان احنا نعطي قيمة dynamic من ال route.
- ال React-Router بيوفر method اسمه useParams بيرجع
object عبارة عن ال parameters اللي مبعوت من ال Route.
شايل

```
① import {useParams} from 'react-router-dom';
② const params = useParams();
```

Q) ازاي نعمل programatically navigation ؟

من خلال useNavigate hook وهي hook بيرجع function ،
ومن خلال ال function دي نقدر ننقل ال navigate لل url اللي
احنا عايزينه.

```
① import {useNavigate} from 'react-router-dom';
② const navigate = useNavigate();
③ navigate("/home");
```

الممسوحة ضوئيا بـ CamScanner

@) ازاى نعمل search params ؟ query string ؟

- ال search params او ال query string هى عبارة عن الحاجة اللى بعد ال (؟) فى ال url .

- ال query string هى data بتكون على شكل Key=value وبتبعتها فى ال url.

- بنستخدم useSearchParams hook عشان نقرأ ونعدل ال query string اللى موجود فى ال url.

- ال useSearchParams بيرجع حاجتين :
① ال search param نقرأ.
② ال set method اللى هتعمل update بيه ال search param.

ex)
```
① import {useSearchParams} from 'react-router-dom';
② const [ sp , setsp ] = useSearchParams();
③ setsp({ key1 : "value1" }) → ?key1 = value1
④ sp.get("key1") → "value1"
```

____/____ ?name1=value1&name2=value2
(search params)

ال search param بتبعت للـ backend عشان تستقبل منه حاجة معينة .

الممسوحة ضوئيا بـ CamScanner

@) ايه فايدة الـ switch component؟

دية component تبع الـ react-router-dom وطبيعتها إنه بتأكد ان Route واحد فقط هو اللي هيتعرض. بيعني مع أول path match يحصل هيـ break.

```
<BrowserRouter>
    <Route path="/" element={<p>home</p>} />
    <Route path="/about" element={<p>about</p>} />
</BrowserRouter>
```

ايه اللي هيتعرض لو الـ path url كان "https://—/"؟
" " " " " " " "/about"—"؟
وايه السبب؟

في حالة الـ "/" هيتعرض home.
في حالة الـ "/about" هيتعرض home about ليه؟

لأن الـ React Router بيـ render كل الـ Routes اللي بتـ match مع الـ path وده by default.

الحل ان احنا نـ wrap الـ Routes جوا الـ <Switch> علشان نتأكد إن Route واحد بس اللي هيـ render مع اول match، ونستخدم الـ property (exact).

@) إيه اهمية الـ BrowserRouter؟
بيـ enable عملية الـ Routing React في الـ components اللي جواه.

Q) ايه الفرق بين <Switch> و <Routes> ؟

ال Route هو بديل ال switch ، هو نفسه ال Switch لكن مع features زيادة.
ال Route خاص بـ V6 ، Switch خاص بـ V5.

Q) ايه الفرق بين ال Switch وال Router ؟

ال Router بيحاوط ال main app component عشان يوفر فيه ال React Routing ، ال Router هو ال BrowserRouter.

ال Router هي render كل ال Routes اللي هت match مع ال url path.
ال Switch هي render أول Route يحصله match فقط.

اي حاجة خاصة بال React Routing منقدر نشغلها خارج ال BrowserRouter component.

Q) ازاي نتعامل مع ال Browser history stack ؟

من خلال hook تبع ال react-router اسمه (useHistory)
1) import {useHistory} from 'react-router-dom';
2) const history = useHistory();

من خلال ال useHistory نقدر نعمل programatically navigate.
→ history.push('/');

الممسوحة ضوئيا بـ CamScanner

(5) sharing data b/w components :

دايماً هنستخدم الـ props عشان نـ share الـ data بين الـ components.
الـ props بيبعت data من الـ parent لـ child .
الـ props ممكن يبعت :- ١) direct value
٢) variable
٣) callback function
الـ props هو readonly .
الـ callback function هي function بنبعتها كـ argument لـ function تانية .

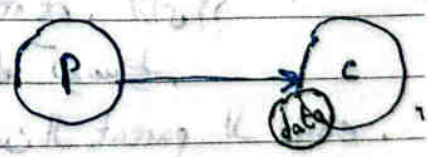
(Q) ايه هي الطرق المختلفة عشان نـ share الـ data ؟

عندنا ٣ طرق :
① نبعت data من الـ parent لـ child . P ⟶ C
② ~ ~ ~ الـ child لـ parent . P ⟵ C
③ ~ ~ ~ الـ child لـ child . C ⟷ C

---

(1) ازاي نبعت data من الـ parent لـ child ؟
← هنستخدم الـ props .

```
(ex) function Child (props){
       return <h1>{props.data}</div>
     }
     function Parent (){
       return (<>
                <Child data="hello" />
              </>)
     }
```



الممسوحة ضوئيا بـ CamScanner

② ازاى نبعت data من ال child لل parent؟

١- هنبعت callback function من ال parent لل child كـ props.
٢- ال child هيـ invoke ال callback ويباصن فيه ال data.

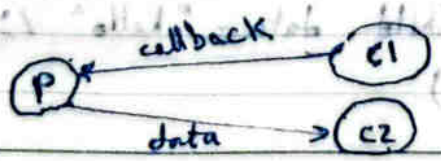
(ex)

```
function Parent (){
    const X = (data)=>{
    };
    return <Child callback={X} />
}
function Child (props){
    const data = "hello";
    return (<>
        <button onClick={()=>props.X(data)}> </button>
    </>)
}
```

① ②

③ ازاى نبعت data من child لـ child؟

عندنا ٣ طرق:
① هنستخدم ال parent كوسيط، وهنستخدم ال state عشان ال parent يحس بالتغيير ويعمل re-render. ازاى؟؟
١-) هنعمل state نستقبل فيه ال data كوسيط.
٢-) هنبعت ال setState كـ callback من ال parent لل child.
٣-) هنحط ال state من ال child اللى هيستقبل ال data.
٤-) ال child هيـ invoke ال setState من ال parent.



الممسوحة ضوئيا بـ CamScanner

```
(ex) function Parent(){
        const [x , setX] = React.useState("");
        return ( <>
            <Child1   callback={setX} />
            <Child2   data={x} />
            </> )
    }
    function Child1(props){
        return (
            <button onClick={()=>props.callback("hello")}>
            </button>).
    }
    function Child2(props){
        return <div>{props.data}</div>
    }
```

الـ ٣ طرق هما :

① هنستخدم الـ state بتاع الـ parent كوسيط .

② ~ الـ Redux .

③ ~ الـ React context .

الممسوحة ضوئيا بـ CamScanner

(6) diff ways to style React component?

Q) ايه الطرق المختلفة ممكن نعمل style؟

عندنا 5 طرق :-

① اول طريقة : هنستخدم plain CSS ، يعني هنعمل stylesheet عادي ونعمله import في الـ component .

② ثاني طريقة : هنستخدم inline style بشرط إن الـ value بتاع الـ inline style يكون js object لأن إحنا بنتعامل مع JSX مش html .

③ ثالث طريقة : هنستخدم css-in-js يعني هنعمل js object عبارة عن css properties وبنبعته لـ inline style .

④ رابع طريقة : هنستخدم css module وده عبارة عن stylesheet عادي لكن الـ classes اللي فيه بتكون local على مستوى الـ component مش global على مستوى الـ project كله .

ذكره الـ css module هو إن إحنا نعمل limit لـ scope بتاعه بحيث يكون local على مستوى الـ component اللي بيستخدمه . ازاي؟؟!
عن طريق إن احنا ندخل ملف الـ css ده في الـ css module compiler وده هيعمل إعادة تسمية لكل الـ classes بحيث يبقوا unique .

الممسوحة ضوئيا بـ CamScanner

ليه ممكن يحصل leak لل css ؟
لان ال css بيجي globally على كل ال app تلقائياً .
الحل إن احنا نعدل أو نخلي أسامي ال classes تبقى unique .

مين المسؤول إنه يعمل generate ال css module ؟
ال webpack هو المسؤول انه ياخد ال css ويدخله على
ال css module compiler .
لازم ال extention يكون (.module.css) عشان
ال webpack يفهم ان ده css module ويدخله على ال css modules .

ايه ناتج ال css module compiler ؟
بيعمل generate حاجتين :
١) ال css باسامي ال classes الجديدة .
٢) object فيه map الاسامي القديمة بالجديدة .

ازاي نستخدم Module css من ال React ؟

١) لازم نسمي ال css بـ (.module.css) عشان ال webpack يميزه .
٢) لازم نـ import ال css module من variable عشان نـ access ال object فيه اللي هي map الاسامي القديمة بالجديدة .

(1) blz.module.css
(2) import hambozo from './blz.module.css' ;

الممسوحة ضوئيا بـ CamScanner

advanced concept (Session 4)

1- class based component Vs functional component.
2- React pure component.
3- React memo
4- Higher Order Components (HOC)
5- Render props
6- Refs - Forward Refs
7- portals
8- Error Boundary
9- state lifting up
10- props drilling
11- context

الممسوحة ضوئيا بـ CamScanner

Q) وضح الفرق الجوهري ما بين ال class component (1)
والـ functional component ؟!!

(1) من حيث ال syntax :
ال functional component هي مجرد plain function عادية بتاخد
React props كـ input وترجع jsx كـ output .

ال class component بيـ inherit من class اسمه Component
وبيعمل override على method اسمه render() بيرجع jsx .

(2) من حيث ال props concept :-
ال functional component بيستقبل ال props كـ argument
بشكل صريح ، يعني لازم نجهز parameter نستقبله فيه .

ال class component بيستقبل ال props في property
جاهز منه اسمه props .

(3) من حيث ال lifecycle وال state :
ال function component بيستخدم React Hooks وهول عبارة عن
special method بنقدر نتعامل بيهم state و lifecycle .
ال function component هي اصلاً stateless يعني مفيهاش
state بالتالي بيستخدم حاجة خارجية زي React Hooks عشان نتعامل .

ال functional اسهل وابسط ومباشرة أكثر من ال class
وحاليا" أي حاجة ال class component بتعملها نقدر نعملها في
ال function component ما عدا ال error boundaries .

الممسوحة ضوئيا بـ CamScanner

② pure component

Q) ايه هو ال pure component ؟

هي component مستحيل re-render إلا لما يحصل تغيير بقيم مختلفة من ال props أو ال state، حتى لو ال parent اللي ساباها عمل rerender هي مش هتعمل rerender مع نفس ال component العادية.

Q) ايه الفرق بينها وبين ال component العادية؟

الفرق في ال implementation بتاع ال shouldComponentUpdate() في حالة ال component العادية دايماً هيرجع بـ true، لكن في حالة ال pure component بيعمل shallow comparision ما بين ال props أو ال state لو حصل فيهم تغيير.
القديم بالجديد القديم بالجديد

Q) ايه هو ال shallow comparision ؟

هي strict equality ما بين 2 variables.
في حالة ال primitive لو ليهم نفس ال value وال type هترجع true.
في حالة ال object لو ليهم نفس ال reference هترجع true.

Q) ايه الغرض من ال pure component ؟

الهدف ان احنا نمنع أي waste re-render، عن طريق مش هيحصل أي rerender إلا لما يحصل تغيير حقيقي في قيم ال props أو ال state.

الممسوحة ضوئيا بـ CamScanner

pure functional component.

③ memo

إختصار لـ memorize.

Q) ايه هو الـ memo()؟

هي method بتاخد component كـ argument وبترجع component جديده لكن مبتعملش re render إلا لو حصل تغيير في الـ props.

Q) ازاي بنستخدمه؟

بنحط الـ component اللي عايزين نعمله memorize كـ argument لـ memo() method.

(ex)

```
import {memo} from 'react'
function x(){
}
export default memo(x);
```

```
let x = memo(function x(){
});
export default x;
```

Q) ايه الفرق بين memo و pure component ؟!

- الـ pure component بتستخدم مع الـ class based component وبتعمل shallow comparision بين الـ props والـ state.

- الـ memo هي تبع الـ functional component وبتعمل shallow comparision بالـ props فقط.

- الـ memo هو HOC.

الممسوحة ضوئيا بـ CamScanner

الممسوحة ضوئيا بـ CamScanner

١- هنعمل file.js نعمل فيه الـ HOC ويفضل الاسم يبدأ بكلمة (with). (ex) withHamboZo

٢- هنعمل الـ HOC وهو function بياخد component كـ argument ويرجع component جديد، الـ component الجديد هيرجع الـ original component ومعبوته فيه الـ functionality / الـ logic، كـ props عشان يقدر يوصلهم.

(ex) withHamboZo.js

```
import React, {Component} from 'react';
const withHamboZo = (OriginalComp) => {
    return class newComp extends Component {
        state = {}
        m1() {}
        m2() {}
        render() {
            return <OriginalComp
                p1 = {this.state}
                p2 = {this.m1} />
                {...this.props}
        }
    }
}
```

common functionality دية الـ

عشان يبعتهم كـ props اقدر اوصلهم.

٣- عشان نستخدمه هنعمله import والـ component اللي عايزين نضيف الـ functionality دية هنحطه كـ argument.

1) import withHamboZo from './withHamboZo';
2) export default withHamboZo(C1);

(ملحوظة) لو هتبعت الـ props لـ component
والـ component دي انت حاطها في HOC
والـ props هتتبعت للـ HOC مش
للـ component.

ازاي اعمل ؟

في الـ HOC وانت بتبعت الـ functionality
كـ props للـ original component اللي راجع
من الـ new component أوعى تنسى تبعت
باقي الـ props ازاي؟؟! باستخدام الـ
spread operator.

```
<OriginalComp
    p1={ }
    p2={ }
    {...this.props}
/>
```

(ملحوظة) انت ممكن تخلي الـ HOC يستقبل اي
parameter انت عايزه، بالتالي تقدر تخلي
الـ functionality تبقى dynamic حسب
الـ component.
الـ HOC في الآخر هي function زي عادية جداً.

```
const withBLT = (Original, x, y) => {
    =
}
```

هتستخدمهم هنا

⑤ Render props = HOC

Q) ايه هو الـ render props ؟ هو pattern ..

- هو نفس فكرة الـ HOC اى احنا نـ share الـ functionality ما بين الـ component ، لكن الـ implementation مختلف .

- الـ render props هو عبارة عن technique من خلاله نقدر نشارك الكود ما بين الـ components عن طريق الـ prop الـ value بتاعته عبارة عن function .

Q) ايه الغرض من الـ render props ؟

الـ reusability ، إن احنا نـ share الـ functionality بين الـ component بحيث نمنع تكرار الكود .

نفس فكرة الـ HOC لكن بـ implementation مختلف .

Q) ايه الفرق بين الـ HOC والـ render props ؟

الاثنين بيحققوا نفس الغرض لكن بـ implementation مختلف .

الـ HOC هو function بياخد component كـ argument ويرجع component جديد .

الـ render props هو prop ، الـ value بتاعته function .

الممسوحة ضوئيا بـ CamScanner

الخلاصة إنك بتعمل component بتحط فيه الـ common logic
وبتبعت الـ data commonlogic بتاعة الـ component دي
كـ props الـ component بتاعتك من خلال
render prop كوسيط

Q) ازاي نعمل implement الـ render props !!؟

اول خطوة هنعمل component عادي جدا" وهنحط فيه
الـ common logic اللي عايزين نشاركها.

تاني خطوة هنبعت الـ common logic كـ argument
لـ prop اسمها render ، وهي دي الـ render prop
هتستقبل الـ common logic وترجع component معين.

```
(ex) import React ,{Component} from 'react'
class B17 extends Component {
    state ={ x:'Ali'}
    m1(){ }
    render(){
        return (<>
            { this.props.render( this.state.x , this.m1 )}
        </>)
    }
}
```

ثالث خطوة ، هنستخدم الـ component ده ، وهنبعتله
render prop اللي هيكون عبارة عن function بيستقبل
argument معينة وترجع component ، هتستقبل الـ argument
دي كـ props .

```
<B17 render={( x, m1 ) => <C1 x={x} m1={m1} />}/>
```

رابع خطوة ، هنعمل الـ props اللي اتبعتت من الـ
component اللي بتشارك فيها الـ logic .

الممسوحة ضوئيا بـ CamScanner

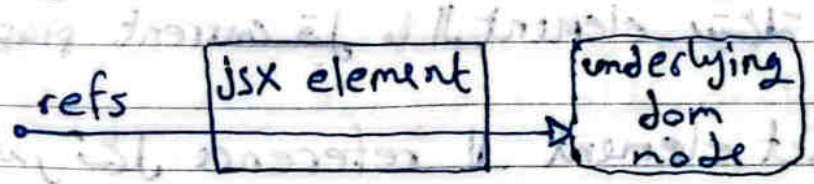
⑥ Refs

Q) ايه هو الـ Refs ؟!!

الـ refs هو technique بيسهلنا ن access الـ dom الاساسي بتاع الـ react element.

Q) ليه هنحتاج الـ Refs ؟

عشان نقدر نمسك الـ dom الحقيقي وراء الـ JSX.



Q) ازاي نستخدم الـ Refs ؟

① اول خطوة هنعمل create للـ reference اللي هيمسك الـ underlying dom node.

① → let inputRef = React.createRef();

(كده انا بعمل container للـ dom element.
الـ dom element هيرجع كـ js object.)

② ثاني خطوة هنباهي أو هنخزن الـ dom node بتاع element معين في الـ reference اللي إحنا مجهزينه من خلال attribute باسم (ref)

② → <Input ref={inputRef} />

③ ثالث خطوة هو إن إحنا نـ access الـ dom node
اللي جوا الـ reference object وده من خلال
property اسمه current.

③ → let node = inputRef.current.value;

الخلاصة من الـ Refs هو إن إحنا بنقدر نمسك
reference بيه الـ real dom node بتاع react element معين.

(ملحوظة)

① الـ ref هيبقى undefined لو حاولنا نـ access
الـ current property قبل ما الـ element يتعمله render.

② احنا نقدر نمثل الـ reference الـ react element
سواء كان <html> أو <class component>

```
1) let componentRef = React.createRef();
2) <C1 ref={this.componentRef};
3) this.componentRef.current.____;
```

④ نقدر نمثل الـ reference الـ child component من
الـ parent component باستخدام الـ Refs بالتالي نقدر
نـ access أي حاجة في الـ child.

④ الـ refs يقدر يمسك class component فقط
وليس functional component.

⑤ الـ refs هو طريقة بنمسك بيها الـ dom node
بتاع react element معين.

الممسوحة ضوئيا بـ CamScanner

Q) ليه نستخدم الـ Refs؟

لو عايز اتحكم في element بيتعمله render عندي.
لو عايز امسك الـ dom بتاع element معين.

(مثال) لو عندي input وعايزه يحصله focus بعد ٥ ثواني.

Q) ليه ممكن نعمل reference لل class component
وليس الـ function component؟

لان الـ function component مفيش منه instance
او object بيطلع منه، عكس الـ class.

احنا ممكن نعمل reference لـ html element
او لـ class component.

Q) ازاي نعمل reference لـ function component؟

هنستخدم الـ React.forwardRef() method وده بياخد
callback function بيستقبل 2 parameters، الاول هو الـ
props والثاني هو الـ ref اللي جاي من الـ parent.

(ex)

```
const fC1 = forwardRef( (props, ref) => <h1 ref={ref}>
</h1>)
```

فكرة الـ forwarding Refs ان احنا نعمل reference
لـ element جوا child component.

P → C → < >

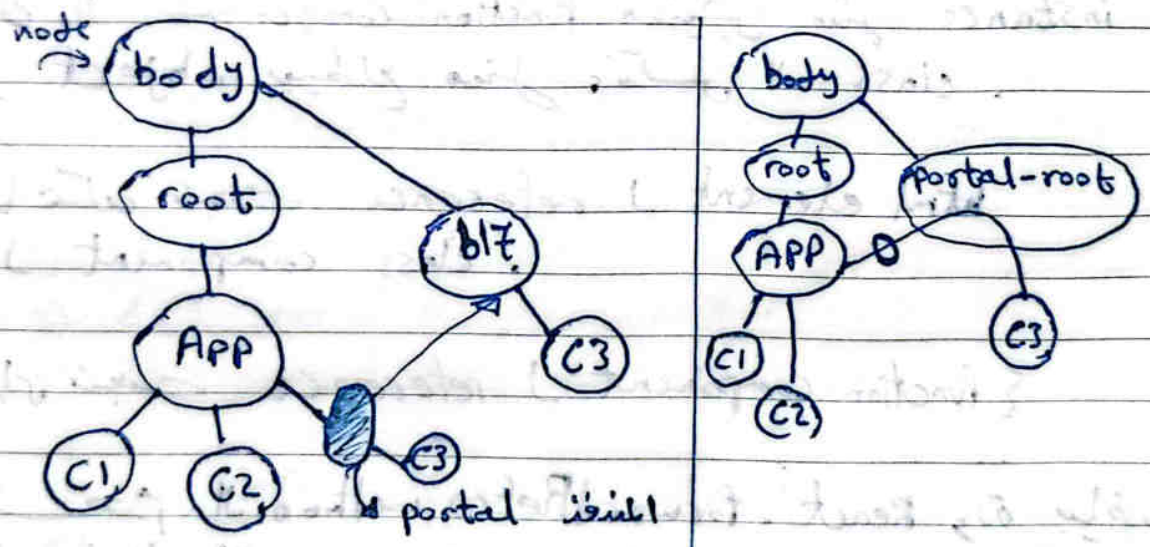
الممسوحة ضوئيا بـ CamScanner

⑦ React Portals

Q) ايه هو ال React Portal ؟

هي عبارة عن طريقة بتسمحلنا ننقل ال renders لـ child component في dom node موجود خارج ال dom hierarchy بتاع ال parent component.

كلمة portal يعني مخرج أو منفذ ، يعني إحنا بننقل منفذ يخلي ال component يـ render في dom node معين.



Q) ازاي ننقل portal ؟

1)

 : اولاً بحدد المكان اللي هتترندر فيه

ثانياً أعمل create لل portal.

2) return ReactDOM.createPortal(
<h1>hello</h1>,
document.getElementById("blz").
)

الممسوحة ضوئيا بـ CamScanner

ReactDOM.createPortal(jsX , dom?)

هعرضه فين؟ هعرض ايه؟

يفضل اسم الـ id بتاع الـ portal container ينتهي بـ (-root)

(ex)

<root />
<App />
<C1>
<C2>
<C3>

break out

portal المنفذ

<portal-root
<C3>

الـ portal بيسمحلي أعرض الـ component في dom node خارج الـ dom tree اللي هو فيه.

Q) ايه الفرق بين .render() Vs .createPortal()؟

الاثنين هيعملوا نفس الحاجة لكن:
في حالة الـ render مش هيحصل event propagation للـ parent.
في حالة الـ portal هيحصل event propagation عادي، وهيحصل event bubbling في الـ dom tree root حتى لو هو اتعرض في dom node برا الـ tree بتاعه.
يعني الـ portal هيتصرف كـ react child عادي هو بس هيتعرض في node تاني.

الممسوحة ضوئيا بـ CamScanner

Q) ليه نستخدم الـ portal ؟

لما نكون نعمل component يظهر خارج الـ parent container اللي هو فيه زي مثلاً الـ (modal) أو الـ (pop up window)،

الـ portal هيعرض الـ element أو الـ component في node برا الـ dom tree اللي هو فيه، ودة هيخلي الـ style ميتأثرش بالـ style بتاع الـ parent.

يعني بالبلدي أنا بعرض الـ component في مكان خاص بيه لوحده بعيداً عن الـ parent اللي هو فيه مع مراعاة إنه لسه مرتبط بالـ parent بتاعه.

لو استخدمت الـ render() كده أنا هفصله تماماً عن الـ dom tree بتاع الـ parent بتاعه.

الممسوحة ضوئيا بـ CamScanner

⑧ Error Boundary

Q) ايه هو الـ Error Boundary ؟

هو أى class component بيعمل implement لواحد من الـ lifecycle methods الخاصة بالـ error handling أو لكلهم وهما:

① getDerivedStateFromError()
② componentDidCatch()

لو فيه component عامل implement لواحد من الاثنين دول أو الاثنين فهو كده بقى error boundary.

Q) ايه الذكرى من الـ Error boundary ؟
Q) ايه الغرض منه ؟ ليه ظهر ؟!!

الغرض منه انه لو حصل اى error اثناء الـ runtime / اثناء الـ rendering فهو هيمسك الـ error دة ويعرضه فى fallback UI بدل ما يحصل crash أو unmount لكل الـ component tree.

الطبيعى انه لو حصل أى error فى الـ component اثناء الـ runtime / rendering فالـ react هيعمل crash أو unmount لكل الـ component عشان يعرض الـ error ودة مش لطيف اثناء عملية الـ development، وهنا ظهر الـ error boundary كـ development tool بيساعدك فى مرحلة الـ development.

← في حالة استخدام الـ error boundary لو حصل error في component معين، فال component ده فقط هيتعرض في fallback UI وباقي الـ components مش هتتأثر.

Summary

١- الـ error boundary هو react component بيمسك الـ errors في الـ child component اللي بتحصل في الـ جواه،
لو حصل error هيعمل log، ويعرض UI معين.
ولو محصلش error هيعرض الـ child component اللي جواه.

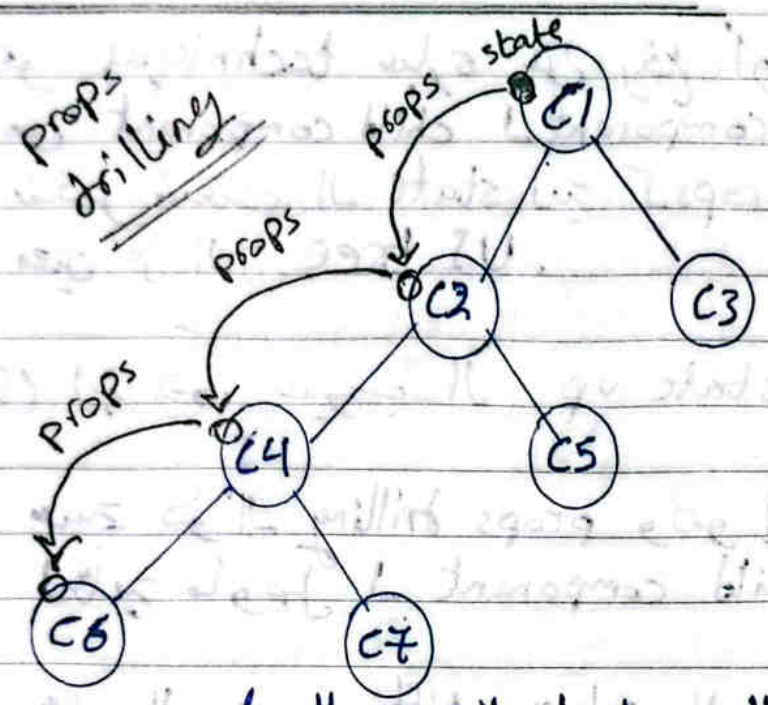
٢- الـ error boundary تقدر تحاوط بيه كل الـ app
او تحاوط بيه كل component لوحده او مجموعة component
هو في الآخر عبارة عن component مع مراعاة انه مش هيشوف غير الـ errors بتاعة الـ child بتوعه.

```
<ErrorBoundary>        if error then:
  <C1>                   1-) log that error
  <C2>                   2-) display UI <...>
</ErrorBoundary>       else:
                         display <C1>
                                 <C2>
```

٣- لو invoke بيه lifecycle method ده ← getDerivedS_f_E_C() وبيعمل throw لـ error من child component
وبيعمل return لـ update state.

الممسوحة ضوئيا بـ CamScanner

⑨ lifting up the state



U.I tree
(or)
component tree

١- الشكل دة عبارة عن التسلسل الهرمي لل components بتاعة ال application. tree of components.

٢- كل component بيكون ليه state خاص بيه لوحده وبيكون عبارة عن ال data الخاصة بال component وبنسميه ال local state.

٣- عشان نبعت state من component لـ component تانية هنستخدم ال props بالتالي لازم ال state يكون موجود في parent component عشان نقدر نبعته بال props.

٤- لما بنعمل نقل ال props عشان نوصل لـ child component معين دة بنسميه props drilling، كلمة drilling معناها حفر، يعني إحنا بنحفر في ال UI tree بال props.

الممسوحة ضوئيا بـ CamScanner

Q) ايه هو ال lifting state up ؟

هو technique عبارة عن رفع أو نقل ال state من child component لـ parent component معين، بحيث نقدر نمرر ال state دي كـ props لـ child component معين في ال UI tree.

Q) ايه هي عيوب ال lifting state up ؟

عيبه هو ال props drilling وهو إن افضل اعمل prop لغاية ما وصل لـ child component معين في ال UI tree.

عملية ال lifting up مرتبطة بال props drilling، طول ما هتعمل lifting state up، هتعمل props drilling.

Q) ازاي نتجنب ال props drilling ؟!!

- اولاً الغرض من ال lifting state up هو ان احنا نخلي ال state يبقى global بالتالي نقدر نوصله من اي component تاني.

- ثانياً العملية اللي احنا بنعملها دي اسمها state management وهو إن إحنا بنـ manage ال state بتاعتنا.

- الحل عشان نـ share ال state من غير ما نعمل prop drilling هو إن إحنا نستخدم Context أو Redux library وده بنعمل بيهم "state management".

الممسوحة ضوئيا بـ CamScanner

(١١) Context API

Q) ايه هو الـ context API ؟

هي طريقة بتسهلنا نشارك data معينة في أي component في الـ component tree من غير ما نستخدم الـ props، بالتالي بتجنب مشكلة الـ props drilling.

Q) ايه عيب الـ props drilling ؟

هي عملية بسيطة لكن مكلفة ليه؟

١- هنضطر نبعت الـ state كـ prop لـ components مش محتاجاها وده عشان توصل لـ component معين.

٢- كل prop هنبعته، هياخد حيز من الـ memory.

٣- أي state update يحصل، كل الـ child component اللي متأثرة هيحصل re-render.

٤- صعب نتتبع عملية الـ update.

Q) ليه الـ context API أسرع من الـ Redux ؟

لأن الـ context هو جزء من الـ React نفسه، لكن الـ Redux هو 3rd party library منفصل عن الـ React.

الممسوحة ضوئيا بـ CamScanner

Q) ازاى نستخدم الـ Context API ؟

اولاً: هنعمل create للـ context.
ثانياً: الـ context بيحتوى على معلومتين مهمين وهما:
1) معلومة توفير الـ data (الـ provider)
2) معلومة استخدام الـ data (الـ consumer)

ثالثاً: هنعمل export للـ context عشان نقدر نستخدمه:

(ex)

```
1) const blZ = React.createContext();
2) const blZProvider = blZ.Provider;
   const blZConsumer = blZ.Consumer;
3) export {blZProvider, blZConsumer};
```

(ملحوظة:)

① الـ context هو object بيحتوى على 2 properties وهما الـ Provider والـ Consumer، وهول عبارة عن React components.

② الـ Provider هو component هنبعت فيه الـ data اللى هنشاركها من prop اسمه "value"، وهنعمل wrap بيه الـ components اللى هنشارك فيها الـ data.

③ الـ Consumer هو component بنعمل wrap بيه function، والـ function دى بتستقبل الـ data من الـ provider وبترجع الـ JSX.

الممسوحة ضوئيا بـ CamScanner

① الـ Provider هنوفر بيئة الـ data .
الـ Consumer هنستخدم بيئة الـ data دي .

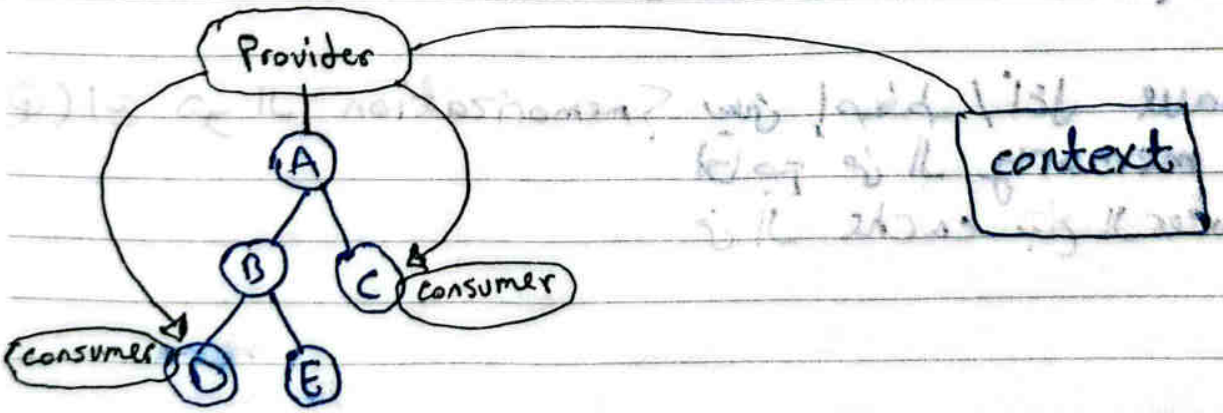
(ex) ① <blZProvider value={this.state.name}>
<C1 />
<C2 />
</blZProvider>

معلومة توفير الـ data
الـ state
اللي هنشاركه

② class C1 extends Component {
render() {
return (
<blZConsumer>
{ (lolo) => {
return (<>
</>)
} }
</blZConsumer>
} })

معلومة استخدام الـ data .

عشان نقدر نكتب كل ده في الـ JSX



- الغرض من الـ context إن إحنا نحل مشكلة الـ props drilling .
- الـ context بيعمل manage لـ global state .

الممسوحة ضوئيا بـ CamScanner

Q) ايه اهمية الـ props؟ ان احنا نشارك الـ data بين الـ components

Q) ايه اهمية الـ state؟ ان احنا نـ manage الـ data الخاصة بالـ component.

Q) ايه اهمية الـ context؟ ان احنا نعمل global state

Q) ايه اهمية الـ HOC pattern؟ الـ reusability

Q) ايه اهمية الـ ref؟ نمسك الـ dom الحقيقي بتاع JSX معين.

Q) ايه اهمية الـ memo؟ ان احنا نمنع أي waste rerender يحصل من الـ function component، يعني مش هيـ rerender إلا لما يحصل تغيير كامل من الـ state أو الـ props.
ليه؟؟
عشان نحسن الـ performance

Q) ايه هو الـ memorization؟ يعني إحفظ / اعمل cache للحاجة من الـ memory أو من الـ cache بتاع الـ browser.

الممسوحة ضوئيا بـ CamScanner

React Hooks

Session (5)

write short and clean code.

1- intro ...
2- useState()
3- useEffect()
4- useContext()
5- useReducer()
6- useCallback()
7- useMemo()
8- useRef()
9- useLayoutEffect()
10- create custom hook

الممسوحة ضوئيا بـ CamScanner

① introduction (write more clean code) ♡

Q) ايه هو الـ Hooks ؟

* الـ hooks هي عبارة عن functions بتسمحلنا نستخدم الـ React features من خلال الـ function component بدلاً من الـ class based component.

* features زي : ١- الـ state ٢- الـ life cycle
٣- الـ context ٤- الـ Refs ...

Q) ايه عيوب الـ class based component ؟

١- اول مشكلة إنه بيتطلب شوية boiler plate code يعني شوية أكواد لازم أكتبها عشان أعمل حاجة معينة، زي مثلاً إني لازم أعمل extend للـ Component وأعمل constructor وأعمل super() وأعمل override على الـ render() method وفيه أكواد أنا وارثها لكن مش محتاجها.

٢- ثاني مشكلة هو إنها بتكون hard to read & reuse.

٣- ثالث مشكلة هي الـ this keyword، المشكلة إنها بتغير الـ context بتاعها بالتالي لازم أعمل bind.

Q) ايه مميزات الـ function component ؟

١- مفيش الـ this keyword.
٢- أبسط بكتير من الـ class من حيث الـ reusability
٣- هي مجرد js function بترجع UI.

ملاحظات

١- كل الـ hooks بتبدأ بكلمة " use ".
٢- الـ hook ما هو إلا function.
٣- الـ hooks بيستخدم مع الـ function component فقط.
٤- الـ hooks ظهر عشان نكتب more short & clean code.
٥- الـ hooks ظهر عشان نقدر نستغنى عن الـ class component.
٦- الـ hooks خلى الـ function component يبقى statefull.

function component ≠ class component
function component + Hooks = class component

٧- الـ hooks بتخلينا نـ hook into بالـ react features
من غير ما نستخدم الـ class components.

٨- الـ hook خلى الـ function تـ manage الـ state بتاعها.

٩- فيه كمان لازم نتعلم واحنا بنستخدم الـ hook وهى :-
① الـ hook لازم تكون متعرفة فى الـ main scope بتاع
الـ function يعنى ميندرش تكون فى block/nested scope.
② الـ React hook بنستخدمها جوا React function compo فقط.
ميندرش نستخدمها جوا class compo أو function عادية ،
لازم تكون جوا React function component.

الممسوحة ضوئيا بـ CamScanner

② useState() hook

← هي function بتاخد initial value وبترجع array.
← الـ array اللي راجع عبارة عن الـ value والـ setter method.
← الـ hook دي بتسمحلنا نعمل state في الـ function component.

(ex) `const [ blZ , setBlZ ] = React.useState('hi');`
state variable ↵ setter function initial state

Q) ايه الفرق بين الـ state في الـ function والـ class؟

← الـ state في الـ class عبارة عن object من key-value.
← الـ state في الـ function عبارة عن variable منفردة بذاتها وكل variable ليها الـ setter الخاص بيها، ودة اندفع وابسط بكتيير من الـ class.

Q) ايه الـ حاجات اللي لازم نتبعها وإحنا بنستخدم الـ hooks؟

١- لازم نعمل call للـ hook عند الـ main scope / top level.
٢- لازم نستخدمه في الـ React function فقط.

Q) ايه اهمية الـ setState؟
إن احنا نعمل update للـ state ويعمل rerender.

Q) ازاي نعمل set للـ state بالـ previous value؟
هنثبت callback هيستقبل الـ old value كـ parameter.

`setBlZ( prevBlZ => prevBlZ + 1 )`

Q) ازاي نستخدم الـ useState مع الـ object أو الـ array؟

الـ setState في الـ hook مش بيعمل merge للـ state
بشكل تلقائي زي الـ class، بالتالي لازم نعمل merge
للـ state بإيدينا manually، وإلا هيحصل clear
للـ value القديمة لما نعمل update.

ازاي نعمل الـ manual merge؟ هنستخدم الـ spread operator.

(ex) const [x, setX] = useState({a:"", b:""});
setX({...x, a:"hello"}) → {a:"hello", b:""}
setX({...x, b:5}) → {a:"", b:5}
setX({b:5}) → {b:5} الـ a اتمسحت

(خلاصة) لما تتعامل مع الـ object أو الـ array:
إتأكد إنك عامل copy للـ object أو الـ array
قبل ما تعمل update أو override وإلا الـ data
القديمة هتتمسح تلقائي، ودة لأن الـ setter
في الـ hook مبيعملش merge للـ state
فبالتالي لازم تعمله بنفسك، ازاي؟؟!!
باستخدام الـ spread operator.

(ex) setX({...x, ...})
setY([...y, ...])
copy — update أو override

③ useEffect()

@) ايه هو الـ useEffect() ؟

هي hook بنستخدمها نتعامل side effect في الـ functional component ، وبتكون بديل للـ lifecycle methods في الـ class component .

@) يعني ايه side effect ؟!

الـ side effect هي اي حاجة بتأثر على شيء معين خارج الـ scope بتاع الـ function .

مثال) ١- نتعامل api request في الـ function .
٢- نتعامل console.log()
٣- نتعامل cache أو localstore

@) ايه الفرق بين الـ function : pure / impure ؟

الـ impure function هي function فيها side effect يعني بتعدل أو بتأثر على حاجة خارج الـ scope بتاعها .

الـ pure function هي function مفيهاش side effect يعني تأثيرها كله جوا الـ scope بتاعها .

في الـ React الـ side effect هو اي حاجة بتحصل خارج الـ scope بتاع الـ React component زي مثلا نتعامل api request أو نتعامل dom manipulation .

Q) ازاى تستخدم ال useEffect() ؟

ال useEffect() بيستقبل two parameters
اول parameter هيكون callback function وده إجبارى .
تانى parameter هيكون dependency array وده إختيارى .

Q) ايه هى ال cases المختلفة لل useEffect() ؟

case 1 : مش هنبعت dependency ، فى الحالة دى ال useEffect
هيه execute بعد كل ما يحصل ال render لل component .

(ex)

```
useEffect( () => { }); = componentDidMount() + componentDidUpdate()
```

case 2 : هنبعت array فاضى ، فى الحالة دى ال useEffect
هيه execute بعد أول render فقط .

(ex)

```
useEffect( () => { }, [] ); = componentDidMount()
```

case 3 : هنبعت array فيه props أو state ، فى الحالة دى
هيه execute بعد أول render يحصل ، وكل ما يحصل
تغيير فى ال dependency value .

(ex)

```
useEffect( () => { }, [props, state] );
```

(props, state) ← prop أو state أو variable

case 4 : هتشغل return function فى ال callback function
وال function دى هتـ run بعد ما يحصل unmount
لل component ، بتسميلها cleanup function .

الممسوحة ضوئيا بـ CamScanner

```
<ex> useEffect ( () => {

    return () => { } // cleanup function
  }, [state] );  = componentWillUnMount()
```

Q) ايه لازمة الـ cleanup ليه بنحتاجه؟

عشان نوقف بيه even listeners معين بحيث مايحصلش "شغال بعد ما نعمل unmount الـ component.
او نعمل clear الـ localStorage، أو نوقف نفس data معينة
او نوقف عملية الـ execution لـ code معين.

ليه؟ عشان مايحصلش leak الـ memory، وحصول crash الـ application.

Q) ايه ميزة الـ useEffect عن الـ lifecycle methods؟!

* الـ useEffect هو flexible أكثر من الـ lifecycle method
وده لأن الـ lifecycle methods هي built-in methods بتعمل
طيل override، يعني methods معينة فقط أقدر استخدمها.

* لكن في حالة الـ useEffect() هي function أقدر أعملها call
كذا مرة بـ argument مختلفة، يعني أقدر أعمل:
أكثر من componentDidMount()
" " componentDidUpdate()
وكل واحد منهم ليه logic معين بيعمله، و dependency
معين بيـ execute بناءاً عليه.

ملحوظات

① الكود اللي جوا الـ callback بتاع الـ useEffect
بنقول عليه/هو ده الـ effect بتاع الـ useEffect

(ex) useEffect(() => { });
الـ effect اللي هيحصل.

② ملخص الـ cases كلها.

① useEffect(() => {} , []);
② useEffect(() => {} , [props or state or var]);
③ useEffect(() => {});
④ useEffect(() => {
return (() => {}) → cleanup
});

نضف وراك
وإنته ماشي

③ الـ useEffect دايماً هتـ run بعد الـ render مش قبله.

④ الـ cleanup function هيـ run بعد الـ unMounting
أو قبل ما يحصل الـ re-execute الـ useEffect.

1) first render → useEffect → initiate cleanup1
2) dependency updated → re-render → run cleanup1
→ init cleanup2 ← run useEffect
3) unmount → run cleanup2

الممسوحة ضوئيا بـ CamScanner

(4) useContext()

هي hook بتستقبل الـ context كـ argument
وبترجع الـ context value.

ليه ظهر الـ useContext()؟

عشان يسهل عملية الـ context consumption.
ظهر كبديل انضف للـ Consumer component.

ايه عيب الـ consumer component؟

في حالة لو فيه أكثر من nested context جوا بعض،
استخدام الـ <Consumer> هيعمل context hell.

ازاي نستخدمه؟

```
(1) const Blz = React.createContext()
(2) <Blz.Provider value= >...</>
(3) const v1 = useContext(Blz);

بدلاً من

<Blz.Consumer>
  {(v1) => {
    }
  }
</Blz.Consumer>
```

⑤ useReducer()

Q) ايه هو الـ useReducer ؟

① هي hook بتشتغل بيل state management ، وهي بديل الـ useState .

② الـ useReducer بيستقبل 2 parameters ،
الاول هو: reducer function ،
التاني هو: initialState value .

③ الـ reducer function بيستقبل 2parameter :
الاول: الـ currentState .
التاني: الـ action اللي هيتم على الـ currentstate .
الـ reducer fun~ هيرجع state جديدة

④ الـ useReducer هيرجع حاجتين / قيمتين :
الاول: الـ state الجديدة اللي راجعة من الـ reducer .
التاني: dispatch method اللي هيحدد الـ action اللي هيتم واللي هيتبعت للـ reducer .

---

useState related to → state
useEffect ~ ~ → side-effects
useContext ~ ~ → Context API
useReducer ~ ~ → reducer function

الممسوحة ضوئيا بـ CamScanner

@) ازاى نستخدم الـ useReducer() ؟

أولاً: اعمل call الـ useReducer() function وابعت فيها الـ reducer function والـ initialState.

(ex)

① const [state, dispatch] = useReducer(reducer, {x:0});

ثانياً: جهز الـ reducer function وابعت فيها الـ state والـ action كـ parameter.

(ex)

```
② const reducer = (state, action) => {
    switch(action){
        case 'doA':
            return state+1
        default:
            return state-1
    }
}
```

ثالثاً: استخدم الـ dispatch عشان تعمل call/invoke للـ reducer، وابعت من الـ dispatch الـ action اللى هيتم على الـ state.

(ex)

```
③ <button onClick={()=> dispatch('doA')}>
  </button>
```


الممسوحة ضوئيا بـ CamScanner

Q) ازاى نستخدم الـ Reducer مع الـ context API؟

الـ useReducer والـ useState بيعملوا manage
لـ local state.
مستحيل نعمل manage لـ global state للزوم
نستخدم الـ context.

ازاى ؟؟!! فى خطوتين:

① هنعمل الـ useReducer عادى جداً.
② هنبعت state والـ dispatch اللى راجعين من الـ useReducer
هنبعتهم كـ value لـ context Provider، وهنعملهم
access باستخدام الـ useContext.

(ex)

```
① const [x, dispatch] = useReducer(reducer, { });
② <Provider
      value={{ xState: x , xDispatch: dispatch }}>
      ≡
  </Provider>

③ xContext = useContext(——);
  xContext.xState;
  xContext.xDispatch(——);
```

الممسوحة ضوئيا بـ CamScanner

Q) ايه الفرق بين ال useState وال useReducer ؟

- الاثنين بنقل بينهم local state management.
- ايه الفرق ؟؟!
- امتى نستخدم دة وامتى دة ؟؟

	useReducer()	useState()	Scenario
١	يفضل نستخدمه مع ال complex state زي object , array	يفضل نستخدمه مع ال data العادية زي number, boolean, string	من حيث نوع ال state
٢	لو فيه أكتر من update مختلف زي اجمع، اطرح، ...	لو هنعمل مع الأكتر 2 updates مختلفين	من حيث عدد ال update المختلفة functions
٣	لو هنعمل complex logic	لو فيه / لو هنعمل business logic عادي وضعيف.	من حيث ال bussines logic بتاع ال update
٤			
٥	يفضل مع ال global state	يفضل مع ال local state	من حيث ال local / global
	لو هنعمل states مرتبطين ببعض.	لو هنعمل states مش معتمدين على بعض	من حيث ال dependency
	for complex state and complex logic	for simple state and simple logic	

الممسوحة ضوئيا بـ CamScanner

⑥ useCallback() ⑦ useMemo()

Q) ايه هو الـ useCallback()؟

① هي عبارة عن hook بيرجع memorized version من الـ callback function . اللي مبعوتالة بالتالي متى هيعمل re-evaluate إلا لما يحصل تغيير في الـ dependencies.

② الـ useCallback بيستقبل حاجتين :
① callback function اللي هيتعمل memorize.
② الـ dependencies اللي بناءً عليها الـ function هتتغير.

③ الـ useCallback هو hook بيعمل memorization أو caching للـ function عشان يمنع ان الـ function يحصلها re-define من أول وجديد وبالتالي يحسن الـ performance.

Q) امتى نستخدم الـ useCallback؟

لما نبعت callback function كـ props لـ child component وعايزين نمنع أي waste rerender لـ child component تيجي.

Q) ازاي نستخدمه؟

```
const cbFn = useCallback(
() => { ... },
[,] );
```


الممسوحة ضوئيا بـ CamScanner

Q) ايه هو React.memo ؟

هي عبارة عن HOC بيرجع memorized version من ال function component حيث مش هيعمل rerender إلا لو حصل تغيير في ال state أو ال props بتوعه. ازاي؟؟

بيعمل shallow comparision ما بين ال state القديمة والجديدة وبين ال props القديمة والجديدة وبيعمل check لو حصل تغيير.

Q) ايه هو ال useMemo() ؟

هي عبارة عن hook بيعمل memorization ال value الراجع من function معينة، يعني بيعمل cach للناتج بتاع ال function من ال previous render وتستخدمه في ال render اللي بعدين بدل ما تحسبه من أول وجديد.

Q) ايه الفرق بين ال useMemo() وال useCallback ؟

الاثنين بيستقبلوا callback function و dependency list (لكن)

① ال useCallback بيعمل caching ال function نفسها.
② ال useMemo بيعمل caching ال value اللي راجع من ال function.
③ ال React memo بيعمل caching لل function component.

بنستخدمهم عشان نحسن ال performance، عن طريق إنهم بيمنعوا أي waste rerender/reevaluation

⑧ useRef()

* هو hook بيسهلنا نعمل reference لـ dom node بتاع react element معين.

* هو hook بيسهلنا نمسك الـ dom node بتاع element معين.

Q) ازاي نستخدمه؟

① const h1Ref = useRef(); — هنربط الـ container
② <h1 ref={h1Ref}> — </h1> — اربطه بالـ element
③ h1Ref.current.___ — اعمل access

Q) امتى / ليه نستخدم الـ useRef()؟

هنستخدمه في حالتين:

① لما نحتاج نمسك الـ dom node بتاع أي element.

② لما نحتاج نعمل variable يحصلله update ويتغير من الـ runtime من غير ما يحصل rerendering للـ component، بدلاً من الـ state اللي هيعمل rerender لما يتغير.

الممسوحة ضوئيا بـ CamScanner

⑨ useLayoutEffect()

الـ useLayoutEffect هي نفس الـ useEffect لكن الاختلاف بينهم هو وقت الـ execution بتاعهم، يعني امتى هيشغلهم fire.

① الـ useEffect بيـ run بشكل asynchronously، وبيـ run بعد ما الـ render يخلص ويظهر في الـ screen.

② الـ useLayoutEffect بيـ run بشكل syncronously وبيـ run بعد الـ render وقبل ما يظهر حاجة في الشاشة.

الخطوات:

① هيحصل render لسبب ما
② الـ react هيـ render الـ component
③ الـ useLayoutEffect هيـ run ويخلص
④ الـ component هيظهر في الشاشة

① هيحصل render لسبب ما
② الـ react هيـ render الـ component
③ الـ component هيظهر في الشاشة
④ الـ useEffect هيـ run

عملية الـ rendering → useLayoutEffect → الـ dom هيتعرض في الشاشة → useEffect →

⑨ ليه هنحتاج الـ useLayoutEffect؟!!

- هنحتاجه لما نعوز ننفذ logic معين قبل ما الـ component يظهر للـ user في الـ screen.
(هو أي فيه حاجة render خلاص لكن لسه مظهرش للشاشة)!!
- اعمل كذا قبل ما يظهر في الشاشة ♡

(10) custom Hooks

Q) ايه هو ال custom hook ؟ ليه هنستخدمه ؟

اولاً : ال hooks ما هي إلا js function بنعمل invoke وبتنفذ logic معين أو بتـ return حاجة معينة .

ثانياً : احنا ممكن نعمل ال hook الخاص بينا ، ال hook ما هو إلا js function .

- ال custom hook ما هي إلا js function أسمها بيبدأ بكلمة " use " ، والغرض من ال custom hook ان احنا نعمل share لل logic معين ما بين ال function components .

- هي بديل لل HOC وال render props .
- ممكن نستخدم/نعمل call لأي built-in hook جوا ال custom hook اللي عاملينه .

- ال custom hook بنحط فيه ال common logic اللي بيتكرر بين ال components .

- ال custom hook كـ reusability انضف وابسط بكتير من ال HOC و ال render props .

- ال custom hook ممكن يستقبل أي parameter عادي وممكن يرجع state أو variable أو method أو function معين أو array من كل دول .

الممسوحة ضوئيا بـ CamScanner

⑦ إزاي نعمل custom Hook ؟!

- الـ hooks هي reusable functions .
- لما نعوز نعمل common logic نكرر نستخدمه في أكتر من component ، في الحالة دي هنعمل extract لـ logic دة جوا custom hook .

① هنعمل function js يبدأ بكلمة " use " .

```
export function useBlZ( ){ }
```

② هنحط جواه الـ reusable code اللي عايزين نشاركه .

③ هنحط parameter لما نعوز نبعت dynamic data .

④ لو فيه state أو variable أو function أو أي logic عايزين نوصله ، هنبعتهم في array / هنرجعهم في array .

```
(ex). export function useBlZ( x=3, y){
        const [t, sett] = useState(x);
        const m1 = ()=>{ }
        const m2 = ()=>{ }
        return [t , sett, m1, m2]
      }
```

→ الـ common reusable logic

⑧ هل لازم نبدأ بكلمة " use " ؟!

لا، مش لازم ، لكن يفضل نبدأ بكلمة " use " عشان الـ React يميزه كـ hook ويعمل check لـ rules بتاعة الـ hooks عليه .

الممسوحة ضوئيا بـ CamScanner

ايه الغرض من ال custom hook؟ code reusability.
اي اجزاء reusable logic

ملحوظة مهمة جداً جداً:

① هل ال React هيعمل create لل state في كل مرة يحصل re-render؟!

لأ، ال react بيعمل create لل state مرة واحدة فقط في أول render يحصل لل component، بعد كده هيستخدمه عادي من غير ما يعمل re-create.

الممسوحة ضوئيا بـ CamScanner

Redux (Session 6)

1- introduction
2- core concepts of Redux
3- Actions, Reducers, store as implementation
4- connect / link React with Redux
5- Redux Thunk Middleware
6- Multiple Reducers
7- combine Reducers

الممسوحة ضوئيا بـ CamScanner

① introduction

@) ايه هو الـ Redux ؟

الـ redux هي library في عبارة عن centralized container
أو data store بتخزن فيه الـ state بتاعة الـ application في
والغرض من الـ Redux ان إحنا نعمل state management
الـ global state بشكل اسهل.

@) ايه هو الـ state ؟

الـ state هي عبارة عن data نقدر نعمل عليها listen
ونتابع اي تغيير يحصل عليها، وبناءاً على التغيير ده فيه
حاجة معينة هتنفذ (زي إنه يحصل rerendering، أو يحصل invoke
لـ function معينة ...)

@) ايه الغرض من الـ Redux ؟

الغرض منه إن إحنا نعمل global state نقدر نوصلها من
اي مكان في الـ application بتاعي وبرضه الـ manage الـ state ده
بسهولة.

الخلاصة: الـ redux عبارة عن state container أو مكان
بتخزن فيه الـ state بتاع الـ application بتاعي في
مكان واحد، واي component يقدر يوصلها بسهولة.

الممسوحة ضوئيا بـ CamScanner

Q) ايه هو الـ React-Redux ؟

Q) ايه هو الفرق بينه وبين الـ React ؟ والـ Redux ؟

الـ React هو js library بنعمل بيه UI بشكل سهل.
الـ Redux هي js library بنعمل بيه state management لأي js application.
الـ React-Redux هي js library بتسهل عملية ربط الـ Redux بالـ React application.

React ⟷ React-Redux ⟷ Redux

Q) ايه هو الـ Redux-toolkit ؟!

هي عبارة عن طريقة جديدة لكتابة الـ Redux بشكل ابسط.
هي عبارة عن مجموعة tools بتسهل كتابة الـ Redux.
هي آخر إصدار حاليا من الـ Redux.

Q) ايه الفرق بين الـ Redux والـ Redux-toolkit ؟

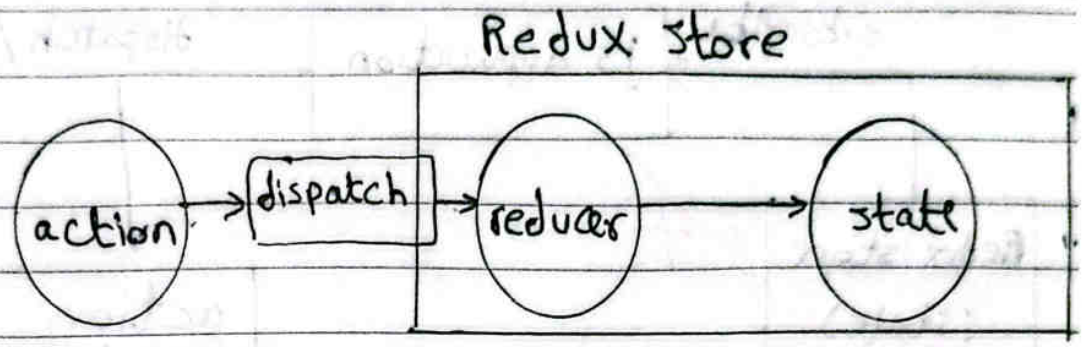
الاتنين Redux، لكن الـ Redux-toolkit هي طريقة جديدة لكتابة الـ Redux بسهولة.

Q) ايه الفرق بين الـ Redux والـ context API ؟

الـ Context هي feature تبع الـ React.
الـ Redux هي js library تتبع الـ js عموما.

الممسوحة ضوئيا بـ CamScanner

(2) core concepts :-



* ال store هو عبارة عن js object بيحتوي على ال reducer وال state بتاعة ال application .

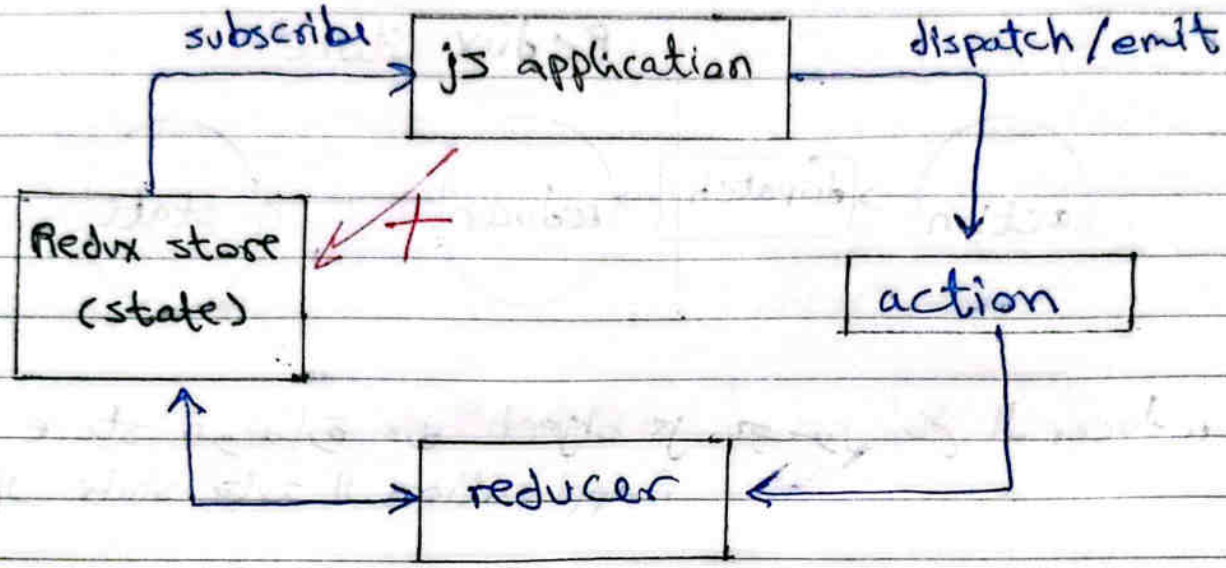
* ال state هي data بنقدر نعمل عليها listen / subscribe ونتابع اي تعديل يحصل فيها .
ال state هي ال global data بتاعة ال application .

* ال reducer هي function بتعمل update على ال state بناءاً على action معين ، وبترجع state جديدة .

* ال action هو عبارة عن js object بيحتوي على property اسمه " type " ، وهو عبارة عن أمر أو instruction بيعرف ال reducer إزاي يعدل على ال state .

* ال dispatch هو method بيخلي ال action يكلم ال reducer هو وسيط ما بين ال reducer وال action .
وظيفته إنه يعمل emit / إرسال ل action معين ، بعد كده ال reducer هيستقبل ال action اللي إتعمل emit / dispatch .

Q) ازاى ال Redux بيشتغل؟



(١) اولاً عندنا application js بيعمل subscribe أو listen على state متخزن فى Redux store.

(٢) ثانياً ال app js ميقدرش يعدل على ال state اللى متخزن فى ال redux بشكل مباشر، عشان ال app يعدل على state لازم يعمل emit أو dispatch لـ action معين.

(٣) لما ال action يجى dispatch، ال reducer هيمسك ال action ده وبناءاً على ال action ده هينفذ update معين على ال current/previous state.

(٤) ال reducer هيعمل return لل state الجديدة وتتخزن فى ال store مكان ال state القديمة، بعد كده التغيير هيسمع فى ال app اللى عامل subscribe لل store.

الممسوحة ضوئيا بـ CamScanner

الخلاصة

١- الـ store هو الـ redux ، وهو object بيمثل الـ state بتاعة الـ application ، وبيمثل الـ reducer اللي هتعدل بيه على الـ state .

٢- الـ state هي الـ global data اللي هتعمل عليها subscribe .

٣- الـ reducer هي الـ function اللي هتعدل على الـ state بناءاً على action معين .

٤- الـ action هو object هيقول للـ reducer ازاي أو ايه التعديل اللي يعمله على الـ state .

٥- الـ dispatch هو method هيبعت الـ action للـ reducer .

الـ reducer باختصار هي function بترجع data بناءاً على instruction معين

Q) ايه هي أجزاء الـ Redux ؟

الـ Redux بيتكون من ٣ أجزاء :

1. Redux store object
2. Reducer function
3. action object

Q) ايه هو الـ payload في الـ action object ؟

هي عبارة عن additional data بنبعتها في الـ action .

الممسوحة ضوئيا بـ CamScanner

③ implementation of Action, Reducer, store

① Actions :

Q) ايه هو ال action و ال action creator؟

- ال action هو js object بيحتوي على property اسمه "type"
- ال action creator هو function بيرجع action object .

(ex)

```
function orderPizza(){ ——→ action creator
    return {
        type: 'Pizza_Order',  → action
        quantity: 3
        payload
    }
}
```

② Reducer

Q) ازاي نعمل ال Reducer؟

ال reducer عبارة عن pure function بتستقبل ال state وال action كـ arguments وبترجع ال next state بناءً عليهم.

(previousState, action) => newState

(ex)

```
const reducer = (state = initialState, action) => {
    switch(action.type){
        case '...' : return { ... }
        default: return state
    }
}
```


الممسوحة ضوئيا بـ CamScanner

③ Store

Q) ازاى شغل الـ Redux store ؟

* من خلال method تبع الـ redux اسمه createStore()
ولازم يستقبل reducer function كـ callback.

(ex)
```
import {createStore} from 'redux';
const myStore = createStore(reducer);
```

Q) ايه هى مهام الـ Redux store ؟ إيه اللى بيعمله ؟!!

١- بيمسك الـ state بتاع الـ application.
٢- بيوفر built-in method بيعمل access لـ state، اسمه getState().
٣- بيوفر dispatch() method اللى هيستقبل الـ action كـ argument وهيبعته للـ reducer.
٤- بيوفر subscribe() method والـ method دى:
① هيستقبل callback function كـ argument.
② هيعمل fire للـ callback بتاعه من كل مرة يحصل update للـ state.
③ هيرجع unsubscribe() method عشان نوقف بيه.

(ex)
```
console.log(myStore.getState());
const unsubscribe = myStore.subscribe(()=>{ });
myStore.dispatch(orderPizza());
myStore.dispatch({type: ' '});
unsubscribe();
```

الممسوحة ضوئيا بـ CamScanner

④ use Redux in React

اولاً، الـ redux هو الـ store ، يعني لما تقول store ، آلتك بتقول redux.

ثانياً، انشاء الـ redux بيتكون من ٣ خطوات.

① جهز الـ actions .
② جهز الـ reducer .
③ جهز الـ store .

ثالثاً، الـ redux ملهوش علاقة بالـ react ، هو معمول لأي js application عموماً ، وعشان نربط الـ redux اللي عملناه بالـ react app هنستخدم react-redux library .

⑤ ازاي نربط الـ redux store بالـ react app ؟

① من خلال component تبع الـ react-redux library اسمه <Provider> ، وظيفته إنه يعمل inject للـ store في الـ child component اللي جواه .

```
(ex) import {Provider} from 'react-redux';
      =
     <Provider store={myStore}>
       <App />
     </Provider>
```

هنبعتله الـ redux store من الـ props.

هنحاول بقية الـ components اللي عايزين تشارك في الـ redux .

الممسوحة ضوئيا بـ CamScanner

① ازاي نستخدم الـ Redux في الـ component؟
② ازاي نـ access الـ Redux؟!

① عشان نمسك الـ state من الـ Redux هنستخدم hook اسمه useSelector() تبع الـ react-redux.

```
import {useSelector} from 'react-redux';
const state = useSelector((state)=> state);
```

الـ useSelector() بيستقبل callback، والـ callback بيستقبل الـ state من الـ Redux ويرجع value.
الـ value اللي راجع من الـ callback هو اللي هيرجع من الـ useSelector.

② عشان نشغل الـ dispatch لـ action هنستخدم hook اسمه useDispatch().

```
const dispatch = useDispatch();
dispatch({type: ''})
```

الخلاصة

- الـ reducer بيعمل update للـ state بناءاً على action معين.
- الـ dispatch بيبعت الـ action للـ reducer وبيـ run الـ reducer.
- الـ dispatch هو بديل الـ setter method.

① الـ useSelector بيمسك الـ state من الـ store اللي جاي من الـ <Provider>.
② الـ useDispatch بيمسك الـ dispatch بتاع الـ store اللي جاي من الـ <Provider>، بيرجع reference للـ dispatch بتاع الـ store.

الممسوحة ضوئيا بـ CamScanner

كلمة middleware
معناها (وسيط)
وسيط بين حاجتين

⑤ Redux Thunk Middleware

Q) ايه هو الـ Middleware ؟!

اولاً الـ action creator هي مجرد pure function يعني مبتستخدمش باي asynchronous task او اي side effect ، بالتالي منقدرش نبعت api call من الـ function دي .

اكل ايه ؟؟ نستخدم Middleware .

الـ middleware هي function بتعترض طريق الـ action قبل ما يوصل للـ reducer ، ميزتها انها impure function يعني أقدر اعمل فيها side effect ، بالتالي نقدر نعمل async call مع الـ action قبل ما يوصل ،

action → middleware → reducer

action: no async
middleware: api call, async code

Q) ليه نستخدم Middleware ؟

عشان نتحكم الـ action creator يعمل async task قبل ما الـ action يوصل للـ reducer ، بالتالي ممكن نعمل update على الـ action او نـ cancel الـ action قبل ما يوصل للـ reducer .

الممسوحة ضوئيا بـ CamScanner

Q) امتى ال middleware بيشتغل؟

بيشتغل بعد ما يحصل dispatch لل action وقبل ما يوصل لل reducer.

Q) ازاى نشغل middleware؟؟

* هنستخدم package تبع ال Redux echosystem اسمه Redux-thunk وده عبارة عن middleware من خلاله نقدر نشغل async action creators.

```
① npm i redux-thunk
② import {applyMiddleware} from 'redux';
③ import thunkMIZ from 'redux-thunk';
④ const store = createStore(reducer, applyMiddleware(thunkMIZ));
⑤ const action1 = () => {
                    return function (dispatch){ . . }
                }

(or) const action2 = async (dispatch) => {
                    return dispatch({type: ''});
                }

⑥ action1(dispatch)
```

هما ٣ خطوات:

① نزل ال redux-thunk.
② اعمل apply ل ال store.
③ اعمل ال async action creator.

Q) ايه ميزة الـ redux-thunk Middleware؟

بيسمحلنا نعمل asyncc tasks في الـ action creator
عن طريق إنه بيسمح للـ action creator إنه
يرجع function بدلاً من action object وميزة
الـ function اللي راجعة إنها مش لازم تكون pure
يعني ممكن تعمل side effects (asyncc) ، والـ function
دي ممكن تعمل dispatch لـ action .

```
ex  const a1 = ( ) => {
        return function (dispatch) { }
    }
```

ميزة الـ thunk Middleware :

① بيسمح للـ action creator يرجع function بدلاً من
action object .
② الـ function اللي راجع ممكن يكون impure يعني
ينوع ينفذ asyncc tasks .

③ الـ function اللي راجع ممكن تعمل جوا dispatch
لـ action معين بشرط إن إحنا نبعته الـ dispatch
كـ argument .

الـ redux-thunk هو middleware بيسمح للـ
action creator إنه يرجع function بالتالي بتقدر
تعمل asyncc call جوا ، وتقدر تعمل dispatching جوا .

الممسوحة ضوئيا بـ CamScanner

⑥ Multiple Reducers

@) ايه هو ال multiple reducers ؟

ال multiple reducers هو إن إحنا نعمل أكتر من reducer function ، وكل reducer منهم بيعدل على جزء معين من ال global state .

ايه المشكلة ؟

المشكلة هنا أن ال redux store مبيستقبلش إلا reducer واحد فقط .

ايه الحل ؟

الحل إن إحنا نعمل combination ل ال reducers دول ، يعني نجمعهم في reducer واحد .

@) ليه نعمل multiple reducer ؟

لغرض تنظيم مش أكتر ؟

ازاي ؟؟

عن طريق إن إحنا نعمل لكل data مستقلة من ال state نعمل reducer خاص بيها .

data مستقلة يعني مبتتأثرش ومش بتأثر على ال data اللي حواليها .

ال reduce هي function بتعدل على ال state بناءا" على action معين .

الممسوحة ضوئيا بـ CamScanner

(7) How to combine multiple reducers :-

- ال redux بيوفر method اسمه "combineReducers()"
وده method بيقبل combine ال reducers لـ reducer واحد
عشان ال store.

ازاي نستخدمه ؟؟

① اعمل import من ال redux.

```
① import {combineReducers} from 'redux';
```

② قبل ال redux store هنعمل combination ال reducers في
reducer واحد هيكون عبارة عن ال rootReducer ،
. ال combineReducers() هيستقبل object وكل key فيه
هيكون عبارة عن reducer معين .

```
② const rootReducer = combineReducers(
        { r1 : reducer1 ,
          r2 : reducer2
        });
```

③ ابعت ال rootReducer في ال store .

```
③ const store = createStore(rootReducer);
```

(ملحوظة) - كل reducer هيدير manage الجزء بتاعه في ال global state ،
- لما نعمل dispatch لـ action ، كل reducers هتستقبله
الفرق إن واحد بس هيمثل ال action ده والباقي
هيعملوله ignore .

الممسوحة ضوئيا بـ CamScanner

احدث طريقة للتعامل مع الـ Routing

① Routing

١- نزل الـ library ← npm i react-router-dom

٢- في الـ App.jsx اعمل import للآتي:

```
import {createBrowserRouter, RouterProvider}
from "react-router-dom";
```

٣- اعمل router باستخدام ()createBrowserRouter ، وهو بيستقبل array of objects ، كل object عبارة عن route/مسار معين، وكل route بيتكون من حاجتين اساسيين:

① path:'' ← المسار / الكلمة اللي بعد الـ domainname.

② element:'' ← الـ component اللي هيـ render لما المسار يتكتب.

```
let routes = createBrowserRouter([
    {path:"/home", element: <Home />},
    {path:"*", element: <NotFound />},
]);
```

٤- عشان نعرض الـ route اللي هيشتغل، هنستخدم RouterProvider.

```
return ( <>
        <RouterProvider router={routes} />
    </>
);
```


الممسوحة ضوئيا بـ CamScanner

(2) nested Routes

١- دية عبارة عن subroutes تبع route ثاني .
هو routes جوا route ثاني .

٢- عشان نعمل subroute جوا route معين ، هنستخدم property اسمه children:[] ودة بياخد array of routes .

```
let routes = createBrowserRouter([
  { path: "/",
    element: <Parent />,
    children: [{index: true, element: <Home/>},
               {path: "about", element: },
               {path: "*", element: }],
  },
]);
```

في ال subRoute مش بنحط / قبل ال path لإنه كده هيمسح كل ال path وراه .

٣- عشان نعرض ال subroute هنستخدم component <Outlet> في ال parent route بتاعته .

```
return ( <>
        <Navbar />
        <Outlet></Outlet>
        </> );
```


الممسوحة ضوئيا بـ CamScanner

③ protected route

هو عبارة عن route مش هيـ render إلا لما
الـ user يكمل login .

إزاى نعمله؟
① هنعمل component خاص بالـ protection
② هنحط فيه الـ components اللى عايزين نحميها كـ child .
③ لو الـ user كامل login هيرجعه .
④ لو مش كامل login ، هيوديه لصفحة الـ login .

```
export default function ProtectedRoutes(props){
    let token = localStorage.getItem("token");
    if(!token){
        return <Navigate to="/login" />
    }else{
        return props.children
    }
}

let routes = createBrowserRouter([
    { path: '/',
      element: <ProtectedRoutes><Home />></ProtectedRoutes>,
    }, ...
]);
```


الممسوحة ضوئيا بـ CamScanner

أفضل طريقة للتعامل مع الـ API بال axios

١- منعمل ملف JS مخصوص للـ Api ← Api.jsx

٢- كل api منعمله function مخصوص ليه ،
الـ function هيستقبل argument لأي حاجة عايز ابعتها
مع الـ api ، وهيرجع الـ data اللي جاية إن وجدت ،

٣- منعمل call للـ function بتاعة الـ api في الـ useEffect
لأن دي أفضل مكان نحط / نعمل فيه side Effect
، والـ data اللي راجعة منخزنها في state ، علشان
نحس بالتغيير

Api.jsx

```
import axios from 'axios';

export let getTrending = async (p1, p2) => {
    let {data} = await axios.get(`API`);
    return data
}

export let axiosPost = async (formdata, endpoint) => {
    let {data} = await axios.post(`ht—/${endpoint}`,
                                  formdata);
    return data;
}
```


الممسوحة ضوئيا بـ CamScanner

```
import axios from "axios";

const req1 = {
    method: 'GET',
    url: '_____',
    params: {'x': 'y', ...},
    headers: {
        'x': 'y',
        'x': 'y'
    }
};

export let result1 = axios.request(req1)
.then(function ({data}){
        return data
  }).catch(function (error){
        return error
  });
```


الممسوحة ضوئيا بـ CamScanner

React JS notes

قبل ما ندرس React أو Angular لازم
نلزم نكون عارفين ٣٠ حاجات :

HTML → it is what data to display/render
to the user on the Browser

CSS → it is about the decoration

JS → used to make the web page dynamic,
dynamic may be for styling or data
→ dynamic style
→ dynamic data
- dynamic يعني حاجة متغيرة مش ثابتة
- بنستخدم الـ DOM عشان نعدل/نغير
الـ web page يبقى dynamically .

أي Javascript library سواء Angular أو React أو Vue
كلهم اتبنوا على الـ DOM manipulation .
كلهم معمولين بالـ DOM.
كل القصة دايرة حوالين الـ DOM .

الـ response اللي بيرجع من الـ server ولما
بيكون html ، الـ browser هيحول الـ html code
اللي جاي لـ dom tree ، الـ node في الـ dom tree
ممكن يكون (element node أو attribute node أو text node)

Name Tel. Email

ديسمبر ٢٠١٦

السبت	٣	١٠	١٧	٢٤	٣١
الأحد	٤	١١	١٨	٢٥	
الاثنين	٥	١٢	١٩	٢٦	
الثلاثاء	٦	١٣	٢٠	٢٧	
الأربعاء	٧	١٤	٢١	٢٨	
الخميس	١	٨	١٥	٢٢	٢٩
الجمعة	٢	٩	١٦	٢٣	٣٠

HTML + CSS = static page
HTML + JS = dynamic page

الممسوحة ضوئيا بـ CamScanner

١ الـ html اللي جاي من الـ server الـ browser محوله
لـ dom tree ، بين parents و childs متعرفين
في nodes .

١٠ كل node من الـ dom ممكن يكون ،
① html element
١١ ② html attribute
③ text

١٢
بعد ما الـ browser حول الـ html لـ dom tree ،
هيعمل render لكل node في الـ dom tree .

٢ وظيفة الـ js هي التعديل على الـ dom tree دية
بيعمل ده dom manipulation .

٣
الـ js بيعدل عالـ html من خلال الـ dom

٤

الـ dom هي عملية تمثيل الـ html على هيئة
٥ tree of js objects .

القصة كلها فايدة حوالين الـ dom manipulation .

٧

Q) ليه الشركات بتشتغل بالـ React أو Angular ؟
٨

← لان الـ React والـ Angular بيبنوا single page application
وده خلى الـ performance عالي جدا جدا .

← السبب هو فكرة الـ SPA .

الممسوحة ضوئيا بـ CamScanner

طب بيتبعت request ؟

١-) ال browser بيستقبل ال html.
٢-) ثم بيحولها لـ dom tree.
٣-) ثم بعد كده بيـ render ال dom، بيعمل render لكل node في ال dom.

ال ٣ خطوات دول بيعملوا مع كل request.

ايه ميزة ال SPA؟ مبيعملش reloading ال html كلها يرجع مع اول request بيتبعت
بالتالي عدد ال request هيقل
فبالتالي ال performance هيعلى جدا" جدا" جدا"

ايه عكس ال SPA؟ Multipage Application

ايه مشكلة ال MPA؟ كل page ليها ملف html منفصل
بالتالي كل page محتاج لاجيب
حاجات request مخصوص.

كل ما يزيد عدد ال request
كل ما قل ال performance.

الممسوحة ضوئيا بـ CamScanner

- الـ SPA بتبعتلك request واحد بس.
ميزة الـ SPA هو الـ performance.

- الشركات بتستخدم الـ React أو الـ Angular عشان يعملوا SPA.

- الحوار كله قايم على الـ performance.

٢ الحوار اللي بيروح للـ server عشان نرجع بالـ html
ده بياخد وقت.

١

- الـ SPA هو application عبارة عن page واحدة فقط
ليه؟ عشان اجيب من الـ server جبت request
واحد فقط.

٣

- ازاي نجيب data من الـ database من غير ما:
نعمل reloading للـ page؟

هنستخدم الـ asynchronous communication.

الـ asynchronous قايم على فكرة الـ parallel tasks.
يعني 2 tasks شغالين في نفس الوقت parallely.

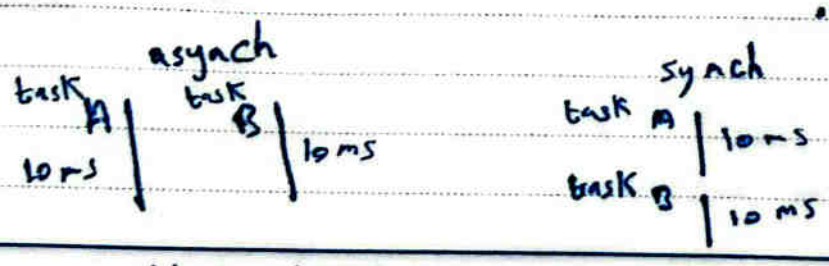
٧

الـ synchronous بيشتغل من series task.

٨

- هنستخدم الـ ajax call اللي قايم على فكرة
الـ asynchronous.

٩



time = 10 ms

time = 20 ms

الممسوحة ضوئيا بـ CamScanner

ال dom هو tree of nodes وال tree هو نوع من
أنواع ال data structure نوع من أنواع طرق
تخزين ال data ،

ال html هو مجموعة data بتحول لـ tree

ال API هو وسيط ما بين 2 programs

برنامج ال كز عشان يتعامل مع البرنامج الخاص بال dom .
هيحتاج API ← DOM API .

ال API من الآخر هو function .

React ecosystem هي البيئة / ال tools اللي لازم
تتوفر عشان تشغل وتشغل react app .

ecosystem معناه الحرفي هو النظام البيئي ،

ال webpack بشكل ذكي بيـ bundle ال modules
اللي بتعتمد على بعض لـ bundle واحد أو أكثر .

الممسوحة ضوئيا بـ CamScanner

lab 1

session [1] intro & installation

ال React ما هي إلا library لبناء user interface ...

Q) ايه هو ال echosystem ؟ ببساطة هي tool مساعدة.
Q) ايه هي ميزة ال React ؟ بتبني SPA.
Q) ايه هما ال React ecosystem ؟

← ال React بتعمل ما هي إلا UI library.
← ال React echosystem هي ال tools اللي بتساعدنا في بناء React application.

← أمثلة على ال React ecosystem :-

1) React
2) BABEL
3) webpack
4) React Router
5) styling
6) Redux

← ال ecosystem دول بيتكامل بينهم React-app.

---

① BABEL ما هو إلا js compiler بنستخدمه عشان نحول ال jsx لـ native js code لأن ال Browser مبيفهمش غير js native.

---

② webpack ما هو إلا module bundler ، وظيفته إنه يجمع أو يجمع كل ال modules كل ال imports وال exports وحطهم في file واحد فقط مبيحتويش على أي import أو أي export syntax.

---

③ وظيفة ال React Router إنه يعرض component معين بناءاً على ال current url path.

الممسوحة ضوئيا بـ CamScanner

Q) ايه فكرة الـ function composition ؟
دي design pattern فكرتها :-
ان احنا نعمل individual functions ونركبهم على بعض
يعني نعملهم composing بحيث نحصل على value معين.

← الـ function composition هو ان احنا نركب functions
على بعض بحيث نطلع منهم بـ value.

← الـ React component هو function بيرجع React elements.

← الـ function composition في الـ React هو ان احنا
نركب الـ functions على بعض بحيث نطلع منهم UI.

$f1() + f2() + f3() = UI$

← الـ React component بنعمله invoke على شكل html tag.

<C1> <C2></C2> <C3></C3> </C1>	function composition in React	component composition

← الـ module عبارة عن شوية code مجموعين grouping
مع بعض في file بحيث يكونوا reusable.
← الـ package هو مجموعة modules معتمدين على بعض.

Name Tel. Email

ديسمبر ٢٠١٦

السبت	٣	١٠	١٧	٢٤	٣١
الأحد	٤	١١	١٨	٢٥	
الاثنين	٥	١٢	١٩	٢٦	
الثلاثاء	٦	١٣	٢٠	٢٧	
الأربعاء	٧	١٤	٢١	٢٨	
الخميس	١	٨	١٥	٢٢	٢٩
الجمعة	٢	٩	١٦	٢٣	٣٠

الممسوحة ضوئيا بـ CamScanner

ال element هو dom node .
ال element هو اصغر وحدة من ال React .
ال React element هو js object بيمثل dom node .
ال React component هو function بيرجع React element .

@) ايه هو ال npm ؟

ال npm هو package manager خاص بال node ،
بيتكون من جزئين : ① Registry مكان نرفع عليه ال package .
② cli مكان نتعامل بيه إلخ .

node_module ده folder بيحتوي على ال source code
بتاع كل ال packages اللي اتعملها install بال npm .

package.json ده file فيه meta-data عن كل
ال packages اللي اتعملها install .

أي package متعملها install ، ال source code بتاعه
هيتخزن في ال node_module ، والاسم وال version وال dependencies
بتوعه هيتكتبوا في ال package.json .

ال dependencies هي ال packages اللي ال app هيحتاجها
عشان يـ run .

```
npm init
npm install
npm run [name of script]
npm install
npm install — --save-dev
npm install — @5.1.3
```

الممسوحة ضوئيا بـ CamScanner

Q) ال webpack ؟؟!

* ال webpack هي build tool وظيفتها انها تـ bundle او تجمع كل ملفات ال js في ملف واحد فقط.

* ال default بتاع ال webpack إنه يـ bundle ال js أو ال json لكن عشان نـ bundle ملفات frontend تانية غير ال js بنحتاج حاجة اسمها loaders.

* فيه loader خاص بال CSS
  " " " html
  " " " images ... etc.

* ازاي نشغل/نستخدم ال webpack؟!!
  ازاي نشغل custome webpack؟!!

① اولاً لازم نعمل install للـ webpack package والـ webpack.
② ثانياً ننزل ال loaders اللي هنحتاجها لو هنعمل bundle لحاجة تانية غير ال files بتاع ال js.
③ ثالثاً نجهز ال webpack configuration file.
- ال webpack هيحتاج يعرف ٣ حاجات:

① ايه/فين مكان ال entry point؟
② ايه ال files اللي هنعمل ال bundle؟
③ فين المكان اللي هنحط فيه ال bundle o/p file؟

Name Tel. Email

ال webpack بيحول أو بيجمع الملفات اللي معتمدة على بعض لملف واحد أو أكثر، وبنسمي الملفات دي bundle.

ديسمبر ٢٠١٦
السبت ٣ ١٠ ١٧ ٢٤ ٣١
الأحد ٤ ١١ ١٨ ٢٥
الاثنين ٥ ١٢ ١٩ ٢٦
الثلاثاء ٦ ١٣ ٢٠ ٢٧
الأربعاء ٧ ١٤ ٢١ ٢٨
الخميس ١ ٨ ١٥ ٢٢ ٢٩
الجمعة ٢ ٩ ١٦ ٢٣ ٣٠

الممسوحة ضوئيا بـ CamScanner

Q) ايه الفرق بين React web app Vs web application ؟

(1) web application :
→ it is collection of web pages (html files)
→ It is M.P.A

(2) React web application :
→ it is only one web page (index.html)
+ many of React components

Q) ايه الفرق بين الـ page web العادي Vs الـ React component ؟

← الـ page web العادي بيكون ملف html .
← الـ react component بيكون ملف jsx / js ، الـ html
مكتبة في ملف js .
← الـ jsx مش pure js .
← الـ jsx هو combination مابين الـ js والـ html .

Q) ازاي نـ setup الـ react project ؟

عندنا طريقتين : طريقة سهلة وطريقة صعبة .

• اول طريقة إن إحنا نستخدم tool اسمها CReate React APP بتخلينا نعمل react application من خلال command line واحد بس .

• ثاني طريقة إن احنا نعمل الـ react app بنفسنا واحدة واحدة باستخدام الـ webpack و Babel .

الممسوحة ضوئيا بـ CamScanner

Q) ازاى نشغل react app من غير create-react-app ؟
~ ~ ~

6- create first React component :

index.js

```
import React from 'react'
import ReactDOM from 'react-dom'
import './index.css'
class App extends React.Component {
    render(){
        return ( <div> hello </div> )
    }
}
```

7- render the React component into DOM .

```
ReactDOM.render(
    <App />,
    document.getElementById('app'));
```

React Element

where to render it ?

Q) ليه الـ ReactDOM هي package منفصلة عن React
مش جزء منه ؟!!
لأن الـ react فيه solution ان يقدر يـ render
في environment مختلف زي الـ dom environment
أو ios environment أو android environment .
بالتالي الـ react هي مش dom فقط .

8- هنحتاج ننزل webpack عشان نـ bundle الـ modules
اللي بنستخدمها import
محتاج ننزل Babel عشان نحول الـ x كود لـ normal كود
الـ browser يقدر يفهمه .

الممسوحة ضوئيا بـ CamScanner

```
>> npm i --save-dev @babel/core @babel/preset-env
@babel/preset-react babel-loader css-loader
html-webpack-plugin style-loader webpack
webpack-cli webpack-dev-server ↵
```

دي كل الـ packages اللي هنحتاجها عشان نشغل
الـ webpack و babel .

9- create webpack.config.js to perform the configuration of webpack.

webpack.config.js

```
const path = require('path');
module.exports = {
    entry: './app/index.js',
    output: {
        path: path.resolve(__dirname, 'dist'),
        filename: 'index_bundle.js'
    },
    { module: {
        rules: [
            {test: /\.(js)$/, use: 'babel-loader'},
            {test: /\.(css$/, use: ['css-loader',
                'style-loader']}]
    }}
},
mode: 'development'
}
```


الممسوحة ضوئيا بـ CamScanner

الخلاصة عندنا طريقتين لـ setup بيئة react :

1. setup react with create react app tool
2. setup react with webpack and babel

ايه هي فكرة الـ props ؟

الـ props للـ component هي الـ argument للـ function .

فكرة الـ props هو ان اصنا نبعت data للـ component زي ما بنبعت argument للـ function .

الـ props هي data بنبعتها للـ component على شكل html attribute وبنستقبلها كـ argument في حالة الـ function component .

الـ props في النهاية هي object .

```
<Child name = 'Ali'
       isAuth = {true}
       logout = {() => alert()}
       head = {<h1>hello</h1>} />
```

الممسوحة ضوئيا بـ CamScanner

session [2]

Q) ايه هو ال Basic flow of React app ؟

index.html
index.js
index.css
App.js

Q) ايه هي ال data اللي نقدر نخزنها في ال state ؟

1) simple data type
2) object data type
3) array of simple data type
4) array of object
5) collection of all of the above

* ملحوظة: ال react state بيتعدل update من خلال ال react setter method فقط.

ميزة ال state انه لما بيحصله update بيعمل re render لل component وبيدخل ال updating lifecycle

الممسوحة ضوئيا بـ CamScanner

rendering list:

the idea is to create list of item element that you could show to the UI.

هنستخدم الـ map().

.map() used to create new array based on the previous array.

ايه اهمية الـ key؟ زيادة الـ performance.

لو حصل أي تعديل على list زي إن فيه element انضاف أو element اتحذف أو element اتعدل، فالحاجة اللي حصلها update هي اللي هتعمل re render فقط وليس الـ list كلها.

```
list = [ , , , ];
List.map( (item) => );
```

ايه اهمية الـ this؟

this keyword allow us to reuse function with diff context.

ايه هو الـ context؟

هو الـ object، وهو القيمة بتاعة الـ this.
هو الـ object اللي هيـ call الـ method.

الممسوحة ضوئيا بـ CamScanner

{ايه هو الفرق الاساسي بين الـ bind() والـ call()}

الـ call() بيـ invoke الـ function بـ context جديدة.
الـ bind() بيـ return new function بـ context جديدة.

```
var f = function (x, y, z) {
    console.log (this.name);
}
var obj = { name: "Ali" };
① f.call (obj, 'A', 'B', 'C');
var arr = ['A', 'B', 'C'];
② f.apply (obj, arr)
```

context argument

```
③ var f2 = f.bind (obj)
```

الـ call() : ١-) بيـ invoke الـ function على طول.
٢-) بيستقبل الـ context بتاعت الـ function.
٣-) بيستقبل الـ argument واحدة واحدة.

الـ apply() زي الـ call() لكن بيستقبل الـ argument كـ array وهو بيفككها.

الـ bind() زي الـ call() لكن بيرجع function جديدة ومش بيـ invoke على طول زي الـ call().

Name Tel. Email

فايدة الـ bind() إن احنا نغير الـ context بتاع الـ function.

الممسوحة ضوئيا بـ CamScanner

ايه فايده الـ this Keyword ؟

- بتشاور على object اللى بيحتويها.
- من خلال الـ this نقدر نـ reuse الـ function، contexts مختلفة.

والـ context هو الـ object اللى الـ this هيشاور عليه.
هو الـ object اللى هي الـ call الـ function.

ازاي نغير الـ context بتاع الـ function ؟

عندنا 3 methods بيغيروا الـ context وهما:

1. call( )
2. apply( )
3. bind( )

ايه مشكلة الـ this Keyword ؟

إنها بتتغير الـ context بتاع الـ function.

ايه الأهمية الـ this ؟ ليه اتعلمها ؟

بتسهلنا نـ reuse الـ function بـ context مختلف، الـ context هو قيمة الـ this، هو الـ object اللى هي الـ call الـ function.

الممسوحة ضوئيا بـ CamScanner

① React Hooks Basics

Q) ايه ميزه ال react في ال SUI؟

ال react بيعمل state و lifecycle لل component.

ال state هو اول اهنا بناخد نسخة من ال data ونقدر نعمل عليها lifecycle، ونقدر نغير في ال runtime.

ال react بيقدر يعمل listen على data معينة ولو حصل تغيير هنعمل rerender ونعرضه، ال data دي هي ال state.

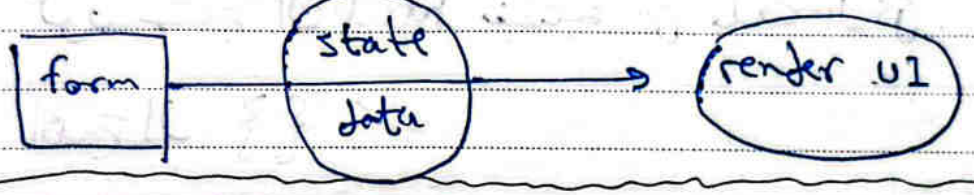
مميزات ال react:

① ال react هي بتستخدم كل pure.
② ال react هي declerative programing.
③ ال react بيستخدم virtual dom
④ ال react بيستخدم ال state.
⑤ ال react بيعمل SPA
⑥ ال react بيبني على ال component.

الممسوحة ضوئيا بـ CamScanner

Q) ايه ميزة الـ component؟ الـ reusability
Q) ايه ميزة الـ SPA؟ الـ performance
Q) ايه هو الـ state؟ هي data أقدر أعمل عليها listen
وأتابع أي تغيير عليها.

Q) ايه هو الـ state؟ هي data الـ system يقدر يعمل عليها listen ويتابع أي تغيير يحصل عليها
وبناءاً على التغيير دة فيه حاجة هتنفذ.



Q) ايه الفرق بين الـ declarative والـ imperative؟

* الـ imperative هو إني أعمل كل حاجة بنفسي، كل صغيرة وكبيرة أنا اللي هتحلها.

* الـ declarative هو إني أقول للـ react أنا عايز أعمل أية، وهو هيهندلها (مثال) الـ lifecycle method والـ state والـ JSX.

الخلاصة: الـ React هو declarative يعني بيوفرلي workflow جاهز استخدمه على طول من غير ما أعمل كل حاجة من الأول.

Q) ايه هو الـ react-dom؟

دي الـ library اللي بتخلي الـ React يتعامل مع الـ virtual dom، هو اللي هيحول الـ virtual dom في الـ runtime.

الممسوحة ضوئيا بـ CamScanner

Q) ايه هو الـ react-script؟

ده هيسهلي أقل run للـ command بتاعة الـ react زي مثلاً test، built، start.

Q) ايه هو الـ JSX؟

← هي tool بتسهلنا نكتب كود html في الـ JS
← هو كود شبه الـ html بنكتبه في ملف الـ JS.

Q) ايه اهمية الـ { }؟

بتسهلنا نكتب كود JS في الـ JSX code.

Q) ازاي الـ JSX بيتعامل مع الـ variable لو كان شكله array؟

في الـ JSX بيعمل foreach على array تلقائياً.
ثم يعرض الـ JSX لو لقى جواه array هيكتبه لوحده.

Q) ليه مش منطقي نفك الـ array بـ foreach في الـ JSX؟؟!

لأن الـ JSX بيفك الـ array بـ foreach تلقائياً لوحده.

Name Tel. Email

(ملحوظة) الـ JSX بيستخدم foreach لوحده تلقائياً عشان يفك الـ array.

الممسوحة ضوئيا بـ CamScanner

ملحوظة: ال JS هيتعامل مع ال HTML كـ object وهيتعامل مع أي JSX كـ object ..

- أي حاجة في ال JSX هي object .

Q) ايه الفرق بين ال state وال variables العادية ؟

ال state هي data لها lifecycle ، يعني نقدر نتتبعها ، ال state هي asynchronous .

ال state ليها lifecycle يعني لما نورك صفحة نقدر نتابعها ، يعني مثلاً نقدر نشوف ال previousvalue بتاعة ال state وال currentvalue وال nextvalue ، ولو حصل تغيير نقدر نحسب بيه وننفذ حاجة معينة زي أن إحنا نعمل re-rendering

---

Q) ازاي ال React بيتعامل مع ال form ؟

عندنا طريقتين :

① controlled component ← يعني ال state هو اللي هيتحكم في ال data بتاعة ال form .

② uncontrolled component ← يعني ال dom هو اللي هيتحكم في ال data بتاعة ال form ، هيستخدم في الحالة دي ال Refs .

الممسوحة ضوئيا بـ CamScanner

controlled يعني ال state هو اللي هيتحكم في ال form.
uncontrolled " ال dom " " " " " " ،

Q) ايه اهمية ال Ref ؟ نمسك ال dom بتاع react element معين.

Q) ايه تاني استخدام لل Ref ؟ نعمل variable يتغير في ال runtime من غير ما يحصل re-render.

Q) هل ال state بيتعمل re-create مع كل rerender ؟

لأ خالص ، ال state بيتعمل create مرة واحدة فقط من أول render ،

Q) امتى ال useEffect بيشتغل ؟

بعد عملية ال render !

Q) امتى ال cleanup function بتشتغل ؟

بعد عملية ال render وقبل ما يحصل re-execute لل useEffect.

← بعد ال render وقبل ال useEffect.

الممسوحة ضوئيا بـ CamScanner

Routing React-Router DOM V6

- فكرة الـ routing هو إن إحنا نتنقل بين الـ components بناءاً على الـ pattern من الـ Browser url .

- الشرط مهم إن إحنا نـ navigate بين الـ components من غير ما يحصل reload للـ page .

Q) ازاى نعمل الـ Routing ؟؟؟

① لازم نحاوط الـ components اللى عايزين نـ route ما بينها بـ <BrowserRouter> component .

الـ <BrowserRouter> هيسهلنا نعمل routing / أو نستخدم الـ routing جواه ، لكن خارجه مش هنعرف نعمل routing .

② لازم نحاوط الـ Route اللى هنتعامل بـ <Routes> علشان يرجع الـ route صح أو match يحصل .

③ الـ route هيرجع الـ component لو الـ path عمله match .

Q) ايه أهمية الـ BrowserRouter ؟

١- هيسهلنا نستخدم الـ routing جواه .
٢- هيحس بالتغيير اللى هيحصل من الـ browser url وبالتالى يعمل rerender ويـ check ايه الـ route اللى عمل match ويعرضه .

الممسوحة ضوئيا بـ CamScanner

ملحوظة: مش هنقدر نستخدم ال react-router-dom
إلا جوا ال <BrowserRouter> ، يعني ال
features بتاعة ال react-router-dom مش
هنقدر نعمل access / مش هنقدر نستخدمها
إلا جواه.

```
<BrowserRouter>
  [
    <Link />
    <Routes>
      <Route />
    </Routes>
</BrowserRoute>
```

~~<Link />~~ X X X

لازم يبقى جوا ال BrowserRouter

الممسوحة ضوئيا بـ CamScanner

Redux Toolkit

٨

أية ميزة الـ global state ؟

بتسهلك تعمل عليل global access يعني تقدر توصلها من اي مكان بشكل direct على طول.

١١

• global يعني تقدر توصله direct من أي مكان.

١٢

إيه الفرق بين الـ local state والـ global state ؟

١

الـ local state مقدرش اعمل share بشكل direct لازم هعمل props.

الـ global state اقدر اوصله من أي مكان بشكل direct.

أية ذكرة الـ Redux ؟

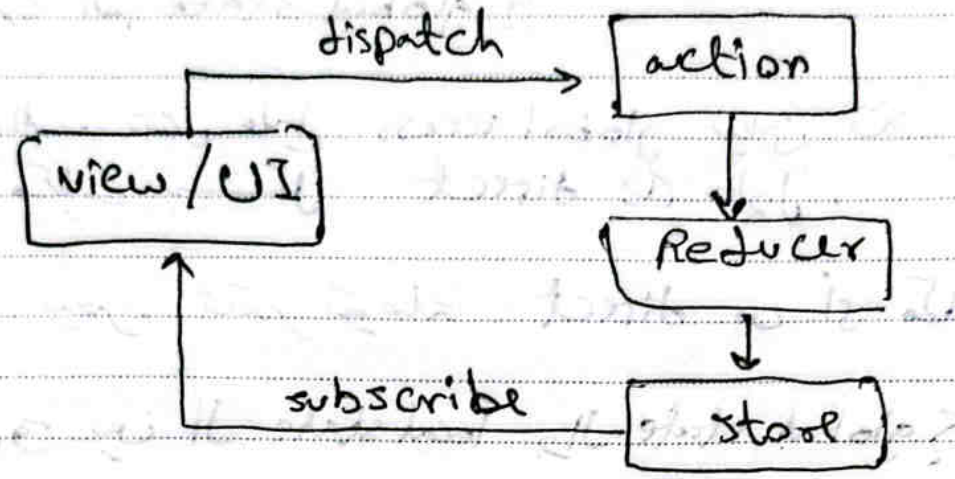
بتعمل state management لـ global state.

أية مكونات الـ Redux ؟

① الـ store اللي هتخزن فيه الـ state.
② الـ reducer اللي هيعدل على الـ state بناءً على action معين.
③ الـ action اللي هيقول للـ ~~state~~ reducer يعدل على الـ state ازاي.
④ الـ dispatch اللي هيبعت الـ action للـ reducer ويشغله.

الممسوحة ضوئيا بـ CamScanner

① use Redux with JS



{ازاي شغل ال Redux}

① جهز ال actions

② جهز ال reducer

③ جهز ال store

- ال store هو object محتوى على ال state وال built-in methods بتوع ال Redux.

- ال action اللي هيقول للـ reducer يعمل ايه.
- ال reducer هو ال function اللي هيحدل مال state.
- ال dispatch هو ال method اللي هيبعت ال action لل reducer وبيحدثه.

الممسوحة ضوئيا بـ CamScanner

ازاى اصيب data من ال store ؟

ـ getState()

ايه ميزة ال store ؟

١١ نقدر نعمل علية subscribe ونتابع اى تغيير يحصل .

١٢

ايه الفرق بين ال getState وال subscribe ؟

١

← ال getState بيشتغل مرة واحدة .
← ال subscribe بيشتغل كل ما يحصل تغيير فى ال state .

ايه هو ال subscribe ؟

هى built-in method بيستقبل callback وبيشتغل كل ما يحصل تغيير فى ال store .

٥

كلمة subscribe يعنى تابع / يعنى حس بالتغيير .

٦

ايه هو ال Redux store ؟

٧

هى object بتحتوى على ال state وال reducer و ٨ built-in methods زى ال redux
[... , subscribe() , getState() , dispatch()] ٩

الممسوحة ضوئيا بـ CamScanner

ايه فايدة ال context api و ال Redux ؟

بتخلي global state متقدر موصلة direct
من اي مكان من غير ما نستخدم props .

ايه هو ال redux-thunk ؟

هي عبارة عن middleware بيسمحلنا
نشغل async tasks في ال action creator
وبعدين dispatch الـ action creator ،

اوامتى ؟؟ ال thunk هيخلي ال action creator
يرجع function مش object ،
وال function اللي راجعة ديه :
١-) ممكن تكون impure (async) .
٢-) ممكن تعمل dispatch .

ايه هو ال middleware ؟

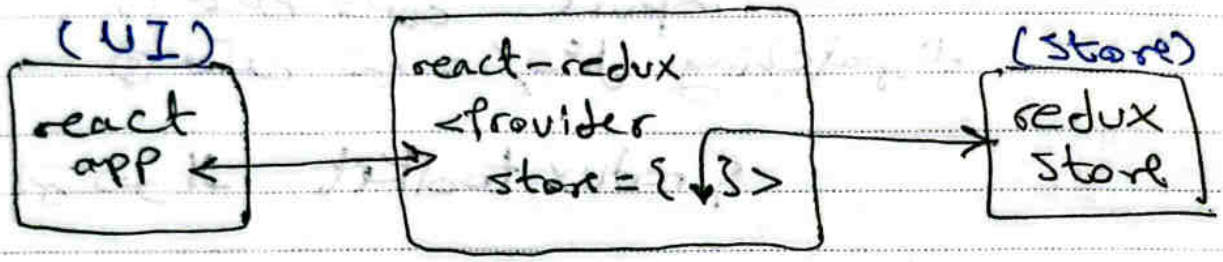
هو function بيعترض طريق ال dispatched action
قبل ما يوصل لل reducer ، ميزته إنه بيسمحلنا
نشغل change على ال action قبل ما يوصل ، أو
حتى نـ cancel ال action ، أو نشغل async task
قبل ما ال action يوصل لل reducer .

Name Tel. Email

الممسوحة ضوئيا بـ CamScanner

ايه هو الـ Redux؟ state management.

ايه هو الـ react-redux؟
هي library بتساعدنا نربط الـ redux بالـ react
هي UI binding library.
ايه هو الـ redux Toolkit؟
هي طريقة جديدة لكتابة الـ redux بشكل أسهل.



ازاي نوصل الـ redux اللي مبثوث من الـ Provider؟
من خلال hooks تبع الـ react-redux.
① const state = useSelector(state => state);
② const dispatch = useDispatch();

ايه هو الـ multiple reducers؟
هو إني إصنا نعمل أكثر من reducer مختلف
للـ global state، وكل reducer منهم بيهتم
بـ data معينة من الـ global state.

ليه نعمله؟ للتنظيم مش أكثر.

ايه أهمية الـ react-redux؟
هي ui binding library، بتربط الـ UI بالـ store:
① بتخلي الـ component يعمل subscribe لـ store.
② بتعمل check لو فيه update جه.
③ لو حصل change، هيحصل re-render للـ component.

الممسوحة ضوئيا بـ CamScanner

{ايه هو ال redux-thunk}

هي عبارة عن middleware بيعمل وسيط بيعترض طريق ال action قبل ما يوصل لل reducer، بنستخدمه عشان نخلي ال action creator يعمل ال async task، إزاي؟؟ عن طريق إنه بيسمح لل action creator يرجع function بدلاً من action وال function دي:

① ممكن تكون impure.
② ممكن تتأخر/تتعطل dispatching.

{ايه هو ال redux toolkit}

- package بتحسن كتابة ال redux.
- طريقة جديدة لكتابة ال redux.

{هل لازم نستخدم ال redux toolkit}

- لا، مش لازم.
- هي مجرد طريقة جديدة لكتابة ال redux لو عجبك استخدمها.
- لو مش عجباك استخدم الطريقة القديمة عادي.

الممسوحة ضوئيا بـ CamScanner

ازاى الـ redux toolkit بتسهل كتابة الـ redux؟

عن طريق إنها بتقسم الـ state لمجموعة slices وكل slice عبارة عن شريحة بتمثل جزء من الـ state. والـ actions والـ reducers في مكان واحد، وكل slice بيختص بجزء معين من الـ state.

ايه الجديد؟

قبل ما كنا بنكتب الـ actions لوحدهم والـ reducers لوحدهم، والـ state لوحدة.

الـ redux toolkit بتجمعهم في مكان واحد اسمه slice.

مثال: نعمل slice للـ user data
" للـ student data
" للـ school data
" للـ counter

ايه هو الـ state slice؟

هو شريحة من الـ actions والـ reducers في مكان واحد.

ايه مكونات الـ Redux والـ Redux toolkit.

old Redux	Redux toolkit
① redux store	① redux store
② actions	② slices → reducer, initstate, actions
③ reducers	
④ state	

الممسوحة ضوئيا بـ CamScanner

إيه هي مكونات ال slice؟
هي شريحة شاملة جواها:
① ال actions
② ال reducers
③ ال initial state

هي عبارة عن object شامل ال actions وال reducers وال initial state في مكان واحد.

---

② إزاي تعمل slice؟

من خلال ال method ، createSlice وده built-in method
تتيح ال redux-toolkit بيستخدم object يحتوي على:

① name property : لازم يكون اسم unique لل slice.
② initialState : هيحتوي على object عبارة عن ال state.
③ reducers : عبارة عن object هيحتوي على actions وكل action هو عبارة عن method بينفذ update معين على ال state اللي مبعوتة.

```
(ex) const slice1 = createSlice({
name : 'slice1',
initialState : { x = 1 } ,
reducers : { a1 : (state) => { } ,
             a2 : (state, action) => { }
} })
```


الممسوحة ضوئيا بـ CamScanner

Q) ازاى نستخدم ال actions؟

① هنعمل export لل actions بتاعة ال slice
② هنعمل export default لل reducer بتاع ال slice.

```
→ export const {a1, a2} = slice1.actions
→ export default slice1.reducer
```

③ في ال redux store هنبعت ال slice بتاعنا في ال reducer object.

```
→ const store1 = configureStore({
      reducer: {
          r1: slice1,
      }
  })
```

Q) ازاى نعمل store باستخدام ال Redux toolkit؟

هنستخدم configureStore() method

Q) ازاى ال Redux toolkit بيشتغل؟

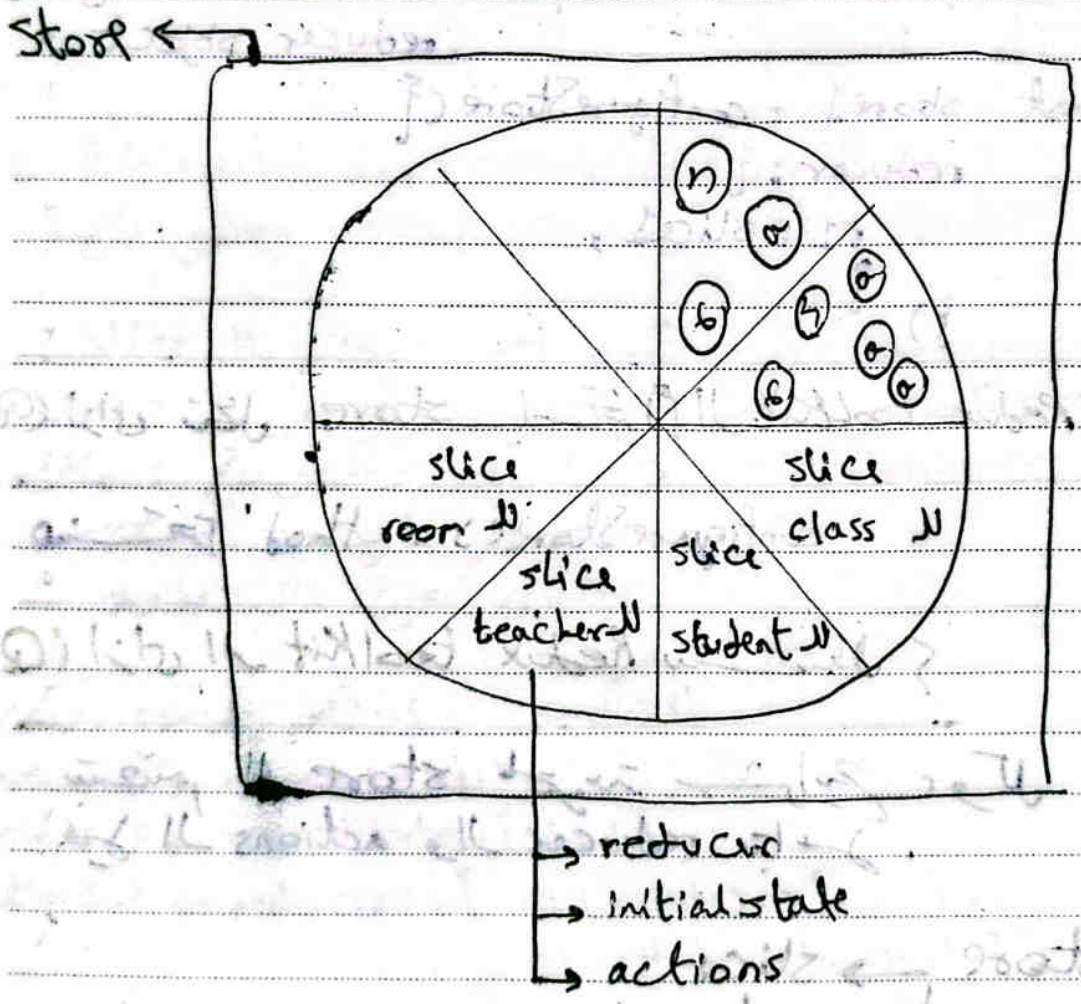
بنقسم ال store لمجموعة شرايح، وكل شريحة فيها ال actions وال reducer بتاعها.

```
store → slice
        ├→ reducer
        ├→ initialstate
        └→ actions
        ⋮
```

الممسوحة ضوئيا بـ CamScanner

Q) إيه أفضل structure نمشي عليه؟

→ "redux" folder
- → store.js file
- → slice1.js file
- → slice2.js file
- ⋮



Name Tel. Email

middleware مش بيحتاج redux toolkit الـ
ومش محتاج يعمل combine لل reducers
ومش محتاج boilerplate كتير ملهوش لازمة.

الممسوحة ضوئيا بـ CamScanner

④) ماهي انواع ال state ؟!

فيه نوعين من ال state :

① component level state وديه عبارة عن local state يعني ال scope بتاعها على مستوى ال component اللي هيا فيه ..

② application level state وديه عبارة عن global state يعني ال scope بتاعها على مستوى ال application كله.

⑤) ايه وظيفة / ايه فايدة ال Redux ؟

هي library بتعمل state management على ال global state .

⑥) ايه هو ال state ؟

هي data نقدر نعمل عليها listen ونحس بأي تغيير يحصل فيها، وبناءاً على التغيير ده، فيه حاجة هتنفذ.

⑦) ايه الفرق بين ال context API وال Redux ؟

الاثنين بيعملوا نفس الغرض، لكن:
ال context API هي feature تبع ال react نفسه.
اما ال Redux 3rd party js library يعني library مستقلة بالعلاقة بال react يعني تقدر تستخدمها مع اي js application.

الممسوحة ضوئيا بـ CamScanner

Q) ايه المشكلة اللي حلتها الـ context API؟
Q) ليه ظهر الـ context API؟

حلت مشكلة الـ props drilling

Q) ايه الـ data اللي ممكن تتحطها من الـ global state؟

مثلاً، الـ data الخاصة بالـ authentication
أو لو عملت multi language app

خطوات التعامل مع الـ Redux :-

① npm i redux react-redux redux-thunk

```
② import {createStore,
          combineReducers,
          applyMiddleware } from 'redux';

  import thunk from 'redux-thunk';

③ const initialState = {}

④ const reducer = combineReducers({})

⑤ const store = createStore(reducer,
                            initialState);

⑥ const middleware = [thunk, ...]

  const store = createStore(reducer,
                            initialState,
                            applyMiddleware[...middleware]
                            );

⑦ export default store;

⑧ import {Provider} from 'react-redux'
  import store from './store'
⑨ <Provider store={store}> <App /> </Provider>
```

الممسوحة ضوئيا بـ CamScanner

أيه أهمية الـ useDispatch()؟

من خلاله هنقدر نـ fire الـ actions اللي عايزينها.

أيه أهمية الـ useSelector()؟

هتسهلنا نـ select جزء معين من الـ state، تقدر نـ select جزء معين من الـ redux store.

من الآخر هنستخدمه عشان نجيب الـ data من الـ store.

الممسوحة ضوئيا بـ CamScanner

Q) How to use redux toolKit?

1- npm i @reduxjs/toolKit react-redux
toolKit package لـ هتنزل الـ ← هتستخدمها نربط
الـ react app بالـ redux.

2- "redux" folder

3- slice1.js

```
① → import {createSlice} from '@reduxjs/toolKit';
② → const initialState = {
        value1: ,
        value2: ,
    }
③ → const slice1 = createSlice({
        name: 'x',
        initialState,
        reducers: {},
    });
④ → export default slice1.reducer;
```

الممسوحة ضوئيا بـ CamScanner

4 store.js

```
import {configureStore} from '@ — ';
import xReducer from './slice1';
export const store = configureStore({
    reducer:{ x:xReducer,
    },
});
```

5 Use the Store in your js app!??
Connect React app with the Redux/store??

```
import {Provider} from 'react-redux';
import {store} from './redux/store';
≡
<Provider store={store}>
    <App />
</Provider>
```

6

```
import {useDispatch, useSelector} from '
≡
const value1 = useSelector((state) => state.x);
const v1 = useSelector((state) => state.x.value1);
const {v1, v2, ...} = useSelector((state) => state.x);
```

destruct هنستخدم

مرجع كل الـ data

الممسوحة ضوئيا بـ CamScanner

* هيف الـ functionality اللي انتة عايزها تتم على slice ، إزاى ؟؟!

```
{
  name: 'x',
  initialState,
  reducers: {
    addToX: (state, action) => {
      state.value1 = action.payload;
      state.value2 = value2 + action.payload.tot;
    },
    ≡
  }
}
```

* ازاى نعمل dispatch ؟؟

→ (اولاً) من الـ slice اعمل export للـ actions :

```
export const {addToX, } = slice1.actions;
export default slice1.reducer;
```

(ثانياً) اعمل import للـ action اللي انتة عايزها تتم :

```
import {addToX} from './redux/slice1';
≡
const dispatch = useDispatch();
≡
dispatch(addToX({pop: '', tot: '', ...}))
```

action.payload

الممسوحة ضوئيا بـ CamScanner

كلمة statefull يعني نقدر نختكر بيانات معينة بتسمى ال state، بالتالي نقدر نتابع أي تغيير يحصل فيها.

* ال react مبيحتويش على directives أي ال Vue وال Angular بدلا من كتير بتستخدم ال jsx وال virtual dom.

* ال Angular وال Vue بتستخدموا ال directive وهي عبارة عن طريقة بتسهلنا شغل dom manipulation على ال element ذات نفسه بشكل مباشر أو بتقول عليه logicless syntax / declarative systax.

* بشكل declaratively يعني بتعرف ال framework عايزين شغل logic اية من غير ما تشرحلها آليات تنفيذ ال logic دة، مجرد هتوصلها عايزين شغل اية وهو هيفهم.

* في ال jsx إحنا بنكتب ال html في وسط كود ال js ذات نفسه، في وسط ال logic، بالتالي مش هنحتاج directive يـ bind أو يربط ما بينهم.

الممسوحة ضوئيا بـ CamScanner

Next 35

٨
٩
١٠
١١
١٢
١
٢
٣
٤
٥
٦
٧
٨

الممسوحة ضوئيا بـ CamScanner

1- ايه الفرق بين ال SSR و ال SPA / CSR ؟

← ال SSR اختصار لـ server-side rendering ومعناها إن ال server هو المسؤول عن انشاء ملف ال html وتجهيزه وتعبئته بالبيانات المطلوبة، ثم ارساله لل client بحيث يكون جاهز للعرض على طول.

وديه هي الحالة الطبيعية اللي بتتم قبل ظهور فكرة ال SPA وظهور ال frameworks اللي بتشتغل فيها.

في حالة ال CSR

← في حالة ال SPA ال server بيبعت ملف html فاضي مبيحتويش على اي بيانات، الملف اسمه index.html، وال client هو المسؤول عن انشاء وملء ملف ال index بالبيانات المطلوبة في الاماكن المخصصة ليه.

CSR اختصار لـ client-side-rendering

2- ايه مشكلة استخدام ال js framework ؟

مشكلة في ال SEO، وال SEO هي ما مدى قابلية موقعك للظهور كجزء من نتائج البحث اللي بيظهر لل users لما يدور على موقعك في google او edge او اي متصفح تاني. استخدام ال js fw بيقلل على ال SEO.

الممسوحة ضوئيا بـ CamScanner

٧- ازاى الـ CSR بيقلل على الميكانيزم تبع الـ SEO ؟

اولاً الـ SEO عشان يحدد مدى قابلية ظهور موقعك من نتائج البحث، لازم الـ SEO يبعت request للملف اللى موجود من السيرفر ويعدها يقرأ محتواه.

وزى ما حالة الـ CSR، ملف الـ html من السيرفر بيكون محتواه فارغ، بالتالى الـ SEO مش هيقدر يظهره من نتائج البحث.

٩- ايه هو الـ NEXT ؟

الـ nextjs هو framework مبنى على reactjs، إتعمل من الأساس عشان يحل مشكلة الـ SEO ازاى؟

عن طريق انه بعد ما الـ client-side يبنى ويملأ الـ index.html بالـ content عن طريق الـ react / ملف الـ react فاضلة، الـ NEXT هياخد نسخة من المحتوى اللى اتملا، وحطها زى ما هو من الـ server-side بحيث لما الـ google تقدر تقرأ المحتوى من السيرفر.

باختصار الـ nextjs بيخلينا نعمل CSR عادى، والـ content هيتبعت للسيرفر آلينا علشان SSR.

٨- ايه ميزة الـ NEXT.JS ؟

الميزة انه بتسهلنا نبنى SPA باستخدام الـ react، ومن نفس الوقت بتعمل الـ SSR بالتالى بتخلى الموقع يبقى صديق للـ SEO.

next.js = SPA + SSR

الممسوحة ضوئيا بـ CamScanner

٥- ازاى نشغل next application ؟

٩ لـ myapp > npx create-next-app@latest

٦- ايه بنية مشروع الـ next ؟ ايه الـ structure العام للمشروع ؟

١١ ① folder اسمه "app" بيحتوى على ملفين مهمين:
١- page.js وده الـ entry point
١٢ ٢- layout.js

١ ② folder اسمه "public" وده بيحتوى الملفات اللى
يتقدر نوصلها بشكل مباشر زى الـ images.
٢

③ file اسمه "next.config.js" وده ملف كز بنحدد فيه
٣ الـ configurations اللى بنحدد بيها حلول الـ next وطريقة
عمله.
٤

٧- ازاى نشغل مشروع الـ next ؟ > npm run dev
٥

٨- ازاى الـ next بيعمل الـ SSR ؟
٦

عن طريق إن الـ next فعلياً عندها client و server ،
٧ يعنى الـ next بحد ذاته بيحتوى على client-side بنشغل
فيه الـ react ، وبيحتوى على server-side نعمل بيه الـ SSR .
٨

Name Tel. Email

يوليه ٢٠١٦

السبت	٢	٩	١٦	٢٣	٣٠
الأحد	٣	١٠	١٧	٢٤	٣١
الاثنين	٤	١١	١٨	٢٥	
الثلاثاء	٥	١٢	١٩	٢٦	
الأربعاء	٦	١٣	٢٠	٢٧	
الخميس	٧	١٤	٢١	٢٨	
الجمعة	١ ٨	١٥	٢٢	٢٩	

الممسوحة ضوئيا بـ CamScanner

٩- ازاى ننشاء page جديدة فى ال next؟

واحد من اهم واجمل المميزات اللى بتوفرها ال next هو ان الصفحات اللى بيتم انشائها فى ال next بيتم ربطها ال routing اللى من خلالها هنزور الصفحات دية بدون الحاجة لانشاء router او استخدام react-router.

ازاى دة بيتم:

① اولاً أى صفحة عايزين نعملها، لازم أولاً نعمل folder باسمها داخل ال app folder.

② ثانياً داخل ال folder بتاع الصفحة هنعمل file اسمه "page.js" ولازم يكون اسمه كده.

③ داخل ال page.js هنعمل react component عادى جداً.

```
app
 |- posts
      |- page.js
```

→ localhost:3000/posts

mapping

١٠- اى page بنعرفه داخل ال folder "app" بيتم اعتباره react-server-component بشكل تلقائى by default.

ايه هو ال react-server-component؟

هى عبارة عن ميزة جديدة فى ال react بتساعد ال component على إنه يتم بناءه اصلاً من ال server-side قبل ما يتم وضع هذا الشيء بالية من ال client-side.

المقصود هنا ال react

الممسوحة ضوئيا بـ CamScanner

١١- ازاى نتنقل ما بين الصفحات؟
ازاى ف navigate ما بين الصفحات؟

من خلال component تبع ال next اسمه Link.
import Link from "next/Link";

<Link href="/posts">
لازم نديله المسار اللى هيروحله والا هيطلع error
</Link>

١٢ ايه هو ملف ال layout.js؟ وايه اهميته؟ تخطيط الصفحات

هو عبارة عن ملف بيحدد الاشياء المشتركة اللى هتظهر فى كل الصفحات اللى موجودة فى نفس ال directory اللى ملف ال layout موجود فيه.

مثال) لو عملنا layout.js جوا ملف ال "posts"، بالتالى كل الصفحات اللى موجودة فى فولدر ال "posts" هيظهر فيها ال layout دة.

ايه محتويات ال layout.js:
① بنعرف فيه ال metadata
② بنعرف فيه component بياخد children كـ props، وال children دة بنحطه فى المكان اللى عايزين ال pages تظهر فيه فى ال layout.

Name Tel. Email

ال children prop بيمثل كل ال pages اللى موجودة فى نفس ال directory اللى فيه ال layout.js.

الممسوحة ضوئيا بـ CamScanner

١٢- ايه اهمية ال children prop في ال layout component؟

بيمثل كل ال pages اللي مع ال layout في نفس ال directory.

بالتالي لو شيلناه مفيش حاجة هتظهر في الشاشة.

١٤- ازاي نعمل nested route؟

هنعمل nested folder باسم ال nested page، وبداخله هنعمل ملف "page.js"

اي page في ال next بيتكون من: ① folder باسمه ② ملف page.js

١٥- ازاي نعمل dynamic route؟

اولا ده عبارة عن route / url قيمة متغيرة.
وكل اللي محتاجين نعمله دلوقتي إننا نحط اسم ال page، اسم ال folder ما بين [ ].

(ex)
```
posts
├ [title]
│   └ page.js
└ page.js
```

ملحوظة ①: اسم ال folder هيمثل اسم ال param اللي هنجيب منه ال value.

② بنوصل لقيمة ال param من خلال props.params.اسم

ايه اهمية الـ meta tags؟

بتحتوي على معلومات عن الصفحة اللي عايزين نعرضها، ودة بيساعد الـ SEO إنه يعرف محتوى الصفحة.

ازاي نعمل meta data في الـ next page؟

هنعمل متغير اسمه "metadata" ولازم اسمه يكون كده، وهيستقبل object بيحتوي على الـ properties بتاع meta tags.

```
(ex) const metadata = {
        title: ' ',
    }
```

لازم نعمل export للـ metadata عشان نشغلها.

مش محتاج حاجة ← const metadata = { }
export const metadata = { }

الـ metadata ممكن تتعرف على مستوى الـ page لوحدها، وممكن على مستوى الـ layout، بيبقى على كل الصفحات اللي تبع الـ layout.

الممسوحة ضوئيا بـ CamScanner

إزاى تعمل data fetching ؟
إزاى تتعامل مع الـ API ؟

← في الـ react كنا بنعمل fetch data في الـ useEffect
بحيث يكون استنى mount للـ component وجاهز يعرض
data .

لكن !
← في الـ next أي component بنعمله بيكون عبارة عن
server component ، والـ useState والـ useEffect وكل
الـ react hooks بتشتغل مع الـ client-component فقط .

← أي component في الـ next هو server-component
تلقائياً وبيكون ليه شروط خاصة بيه غير الـ
client-component .

ليه الفكرة من الـ server-component ؟

- هو عبارة عن شغل بنروح نطلب من الـ server إنه
يجيب هو لنا جاهز ، يعني آكواد الـ html وبناءها هيحصل
في الـ server .
- بالتالي فكرة الـ state وإنه يحصل rerender مش منطقي
أنه يتحصل في الـ server .
- بالتالي مفهوم الـ hooks والـ state والـ rerender ملوش
أي استخدام مع الـ server-component .

امتى نقدر نستخدم الـ hooks والـ state في الـ next ؟

لما نعرف client-component ، وده الـ component
اللي هيتم إنشاؤه في الـ client-side بالتالي مفهوم
الـ states والـ rerender نقدر نطبقه .

الممسوحة ضوئيا بـ CamScanner

١٨ ازاى تتعامل Data fetching فى ال server-component؟
ايه ميزة التعامل مع ال APIs فى ال next؟

١) اولاً ميزة ال server-component إنه بيشيل عن هم ال state management وال life cycle،

بالتالى تقدر تبعت request مباشراً من غير ما تهتم بال state وال useEffect وال rerendering ومشاكل ال side-effect.

٢) تقدر تتعامل side-effects بشكل مباشر، وده شىء محرم وممنوع فى ال react، ولكن ال next وفرته باستخدام ال server-component.

١٩ أى component / page فى ال next بيكون server-component بشكل تلقائى، وده ال default.

Q) ازاى تتعامل client-component؟

فى أول سطر فى الملف هتكتب "use client"

(ex) components
└ tobo.jsx → "use client"

الممسوحة ضوئيا بـ CamScanner

٢٠ ايه الفرق بين ال client-component & server ؟

الاثنين هما server-side-rendered (SSR) ، ولكن ١
① ال client-component بيحصل فيه rerender يعني بيعمل
مفهوم ال state وال effect والحاجات اللي ليها علاقة
بال re-rendering .

② ال server-component بيتم بناءه اساساً من ال server
بالتالي مش هنقدر نستخدم الحاجات المتعلقة بالـ rerendering ،

٢١ ايه الفرق بين ال SSR وال SSG وال ISG ؟

ال SSR هو ال server-side rendering اللي احنا
عملية من رقم (1) .

ال SSG اختصار لـ static-site generation وده بيكلم
انه بدل ما يكون عندنا server بنعمل عملية request
وبناءً على ال request ال server بيبدأ يفهم ال request
ويجهز الملفات اللي هتتبعت لـ response ،

ال SSG بيقول إنه فيه طريقة اسرع من حالة إن
البيانات متتغيرش كتير ، ازاي؟ إنه في الوقت اللي
بنعمل فيه build للمشروع عشان نجهزه للـ deploy
في الوقت بتاع ال build ، ال next هيروح يبني كل
الصفحات الموجودة ويجهز للنتيجة النهائية لكل
ال API request ، بحيث تكون النتيجة النهائية للصفحة
هي ملف html ، والملف ده مش هيتغير مع كل
API request نبعته ، من الآخر هيبقى عندنا نتيجة
نهائية جاهزة . وده اسرع بمراحل لأن إحنا مش
هنضطر نبعت API request تاني لأنه خلاص اتبعت اثناء ال build .

الممسوحة ضوئيا بـ CamScanner

الـ next بتوفر طريقة الـ SSR والـ SSG
واستخدم اللي بتقرر انها انسب لمشروعك.

ايه هو الـ ISG؟

اختصار (ISR) incremental-static-regeneration
وفكرته انه بيجمع ما بين الـ SSR والـ SSG بحيث
اول مرة بنطلب request للسيرفر عشان نجيب
المشروع / الموقع، السيرفر هيشتغل وحسب النتيجة
النهائية ويرجعلي، من المرة الثانية اللي هتعمل فيل
request خلاص النتيجة النهائية كانت محسوبة عندي
من اول مرة ومحفوظة في الـ cache.

الـ cache هو ذاكرة تخزين مؤقتة عادية limit
لمدة زمنية محدودة.

لو البيانات في الصفحة مش بتتغير بشكل دائم
ويمكن نغيرها مع الـ build time (في كل مرة)
فيفضل نعتمد على الـ SSG

لو البيانات بتتغير يفضل نعتمد على الـ SSR.

الممسوحة ضوئيا بـ CamScanner