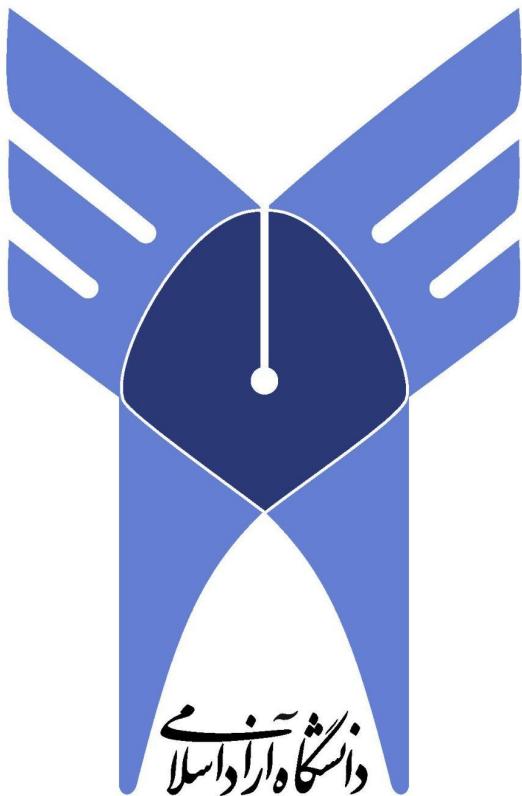


به نام خدا



پروژه فرانت اند فروشگاه اینترنتی با React

نام دانشجو: زهرا حاجی علی

شماره دانشجویی: ۹۸۰۱۱۶۹۸۲

درس های مربوطه: کامپیویلر و طراحی نرم افزار

نام استاد: جناب آقای دکتر کارگر

تکنولوژی هایی که به طور خلاصه در این پروژه استفاده شده اند عبارتند از :

Analytics



[Google Analytics](#)

JavaScript frameworks



[React](#)



[styled-components](#)

5.3.5

Development



[styled-components](#)

Miscellaneous



[webpack](#) 50% sure



[HTTP/3](#)



[PWA](#)



[Module Federation](#)

50% sure

JavaScript libraries



[Swiper](#)

UI frameworks



[Tailwind CSS](#)

مقدمه:

پروژه ای که پیاده سازی و اجرا شده است یک فروشگاه اینترنتی است که این پروژه به صورت وب اپلیکیشن طراحی و توسعه داده شده است و مدت زمان پیاده سازی آن از طراحی اولیه تا اجرا حدوداً دو ماه به طول انجامید. از ویژگی های این پروژه می توان استفاده از آخرین تکنولوژی های حال حاضر فرانت گفت که پرفورمنس بالایی دارد و اینکه این اپلیکیشن به صورت pwa طراحی و توسعه داده شده است و می توان به صورت موبایل اپلیکیشن هم در گوشی های اندروید و هم در گوشی های آیفون استفاده و بهره برداری کرد. تمام اطلاعات موجود در وبسایت کاملاً ساختگی است و به صورت ماک بوده، صرفاً جهت رونمایی از پروژه استفاده شده است.

در این گزارش ابتدا با تفاوت وبسایت معمولی با وب اپلیکیشن ها آشنا می شویم و با ابزار هایی که در پروژه استفاده شده، مفاهیم اولیه و مدل کد نویسی در اجرای پروژه آشنا می شویم و در انتها تصاویر نهایی پروژه قرار داده شده است.

تفاوت وب اپلیکیشن ها با وبسایت های معمولی در چیست ؟

همانطور که میدانید برنامه نویسی وب از گذشته تا کنونی تغییرات زیادی داشته و وبسایت از امکانات پیشرفته تری بهره مند شده است که این پیشرفت ها توسعه وبسایت ها به صورت وب اپلیکیشن ها هستند که هم سایت های ایرانی و هم سایت های خارجی همگی به این سمت حرکت کردند و وبسایت های خودشان را توسعه دادند. حال مهم ترین سوال این است که تفاوت وب اپلیکیشن ها با وبسایت های معمولی در چیه و چرا سایت های بزرگ مثل اپنستاگرام ، فیسبوک ، اسنپ ، دیجی کالا و ... همگی به این سمت حرکت کردند و اپلیکیشن های خودشون رو توسعه دادند ؟

در اپلیکیشن های وب سایت وقتی لود میشه یکبار لود میشه و وقتی ما به صفحه دیگر بخوایم بریم دیگر لودینگ نداریم و علاوه بر اینکه تجربه کاربری بسیار خوبی به کاربران میدیم باعث میشود سریعت وبسایت ما بیشتر بشه و همچنین وقتی یکبار لود میشود باعث می شود در خواست کمتری به سمت سرور برود و منابع کمتری مصرف بشود.

وب اپلیکیشن ها به دو صورت اجرا و پیاده سازی میشوند که عبارتند از :

- SPA
- MPA

وب اپلیکیشن ها **SPA** مخفف Single Page Application ها هستند که تمام وبسایت با یکی از فریم ورک های فرانت اند مثل react, vue, angular اجرا و پیاده سازی می شوند

وب اپلیکیشن ها **MPA** مخفف Multi Page Application ها هستند که یعنی بخشی از این وبسایت ها رو با فریم ورک طراحی می کنند و بخش دیگر را به صورت وبسایت معمولی با html, css, javascript می نویسند

مزایای توسعه اپلیکیشن های تک صفحه ای یا SPA در چیست ؟

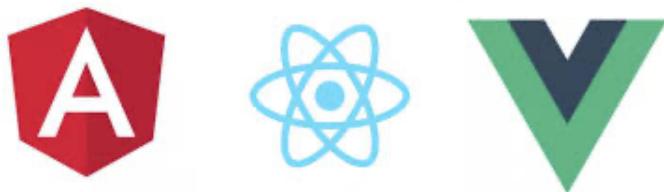
سرعت بالا از مهم ترین مزیت های اپلیکیشن های تک صفحه ای به شمار می رود. در حقیقت بیشتر ریسورس های مورد نیاز یک اپلیکیشن تک صفحه ای (HTML + CSS + Scripts) در شروع کار لود شده و در حین کار به بارگذاری مجدد (ریلودینگ) نیاز نخواهد بود. تنها چیزی که در این میان تغییر می کند، تبادل دیتا بین اپلیکیشن و سرور خواهد بود.

به طور کلی، این اپلیکیشن به کوئری های کاربران با سرعت بالایی پاسخ می دهد و نیازی به تعامل مداوم بین کاربر-سرور (Client-Server) نخواهد داشت. ساخت چنین اپلیکیشن هایی از دیدگاه یک برنامه نویس بسیار بهینه شده است. به گونه ای که برای رندر کردن صفحات بر روی سرور نیازی به کدنویسی نیست و حتی برای شروع فرآیند توسعه به سرور احتیاجی نخواهد بود. در حقیقت می توان کار را از یک فایل

شروع کرد، علاوه بر این برنامه نویس برای طراحی وب اپلیکیشن های بومی موبایل می تواند دوباره از همان کدهای سمت سرور (Backend Server-Side) و API های کاربردی استفاده کند.

برای توسعه وب اپلیکیشن ها از چه ابزار های می توانیم استفاده کنیم ؟

برای توسعه وب اپلیکیشن ها ابزار های مختلفی وجود دارد که ۳ تا از محبوب ترین های آن عبارتند از React ، Angular ، Vue که اغلب شرکت بزرگ در توسعه آنها نقش داشتند مثل فیسبوک ، گوگل و ... به طور خلاصه با هر کدام از آنها آشنا میشیم و مزايا و معایب آن ها را بررسی می کنیم.



چیست React ؟

React که توسط فیسبوک معرفی شده است یک کتابخانه متن باز JavaScript است. که برای ساخت رابط کاربری تعاملی و stateful و همچنین رابطهای کاربری قابل استفاده مجدد معرفی شده است . برای رندر کردن رابطهای پیچیده با ویژگی های زیاد خوب عمل می کند. همچنین با کمک دام مجازی (virtual DOM) می توان وب اپلیکیشن های بسیار پایدار ساخت.

چیست Angular ؟

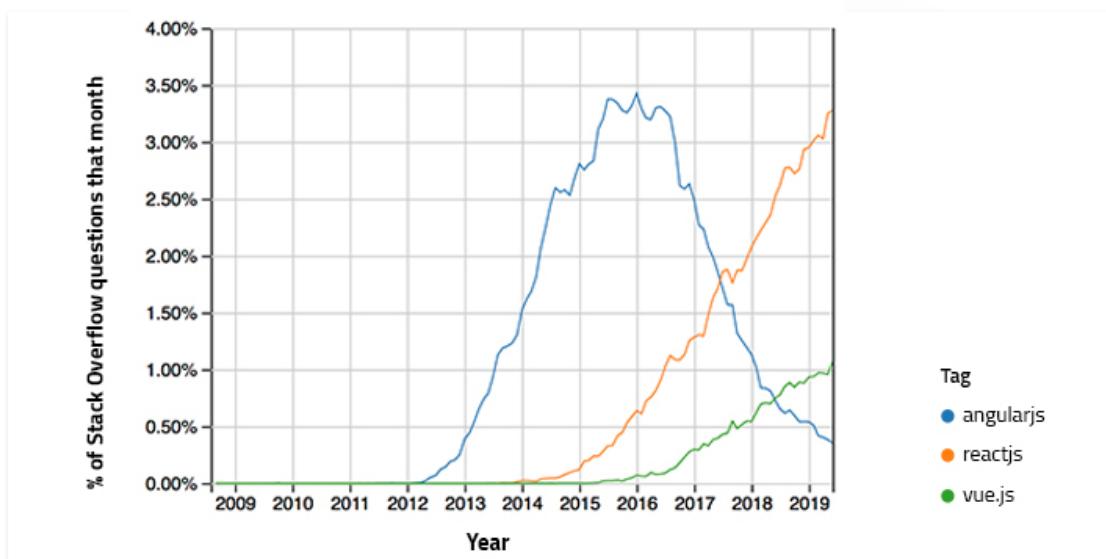
Angular یک فریمورک متن باز است که توسط گوگل ارائه شده است که دارای معماری Model-View-Controller (MVC) است و توسعه ، نگهداری و آزمایش را برای توسعه دهندگان آسانتر می کند. برای ساخت برنامه های وب پیچیده و تعاملی فوق العاده است ، اما برای برنامه های تک صفحه ای محبوبیت زیادی دارد.

DATE	STABLE RELEASE	COMPATIBILITY
May 2019	8.0.0	^7.0.0
October / November 2019	9.0.0	^8.0.0

Vue چیست ؟

Vue یک فریمورک وب پیش‌روندۀ JavaScript است که برای ایجاد رابط‌های کاربری معرفی شده است. از ابتدا برای این طراحی شده است که از همه نظر قابل قبول باشد. ابزارهای مختلف زیادی برای ایجاد رابط کاربری دارد. و قابلیت‌های زیادی برای پشتیبانی از برنامه‌های تک صفحه‌ای دارد زمانی که از ابزارهای مدرن و کتابخانه‌های پشتیبان به همراه آن استفاده کنیم.

کدام یک از فریم‌ورک‌ها محبوبیت بیشتری دارند ؟



خوب ، اگر نمودار و آنالیز گوگل را مشاهده کنیم ، می‌بینیم Angular تا سال 2016 بسیار محبوب بود ، اما از سال 2017 به بعد ، جستجوهای برای React به طور پیوسته در حال رشد هستند. همچنین محبوبیت تگ‌ها را نشان می‌دهد. بین ReactJS و AngularJS و VueJS که محبوبیت React در آن مشخص است.

تفاوت بین Angular ، Vue ، React چیست ؟

ATTRIBUTES	ANGULAR	REACT	VUE.JS
Type	JavaScript framework	Open Source JS Library	Progressive JavaScript Framework
Npm weekly downloads (2018)	444,794	5,036,078	996,293
Size	167 KB production 1.2 MB development	109.7 KB production 774.7 KB development	30.67 KB production 279 KB development
Easy to learn	Steep (Learn TypeScript)	Moderate	Easy
Coding speed	Slow	Normal	Fast
Documentation	✓	✓	✓
Performance	✓	✓	✓
Startup time	Longer due to its large codebase	Quick	Quick
Complete web apps	Can be used on standalone basis	Needs to be integrated with many other tools	Requires third party tools
Data binding	Bi-directional	Uni-directional	Bi-directional
Rendering	Client side	Server side	Server side
Model	MVC	Virtual	Virtual
Code reusability	Yes	No, only CSS	Yes, CSS & HTML
When to use	Production, esp. enterprise apps with Material UI	Production, custom UI apps	Startups, production

چند نمونه از شرکت های بزرگی که از این فریمورک ها استفاده کرده اند

ANGULAR.JS	REACT.JS	VUE.JS
The Guardian 	AirBnB 	Alibaba 
Upwork 	Instagram 	Grammarly 
Pay Pal 	UberEats 	IPL dashboard 
Sony 	Dropbox 	Gitlab 

همانطور که گفته شد ما با توسعه وب اپلیکیشن ها و ابزار های آنها و همچنین مزایا و معایب هر کدام آشنا شدیم. ما در این پژوهه فروشگاه اینترنتی از کتابخانه React استفاده کردیم. دلیل انتخاب من از ری اکت این بود که برخلاف انگیولار و ویو ری اکت یک کتابخانه بود و حجم کمتری رو در توسعه نسبت به دو فریم ورک قبلی داشت و چون توسعه داخل React به صورت component base هست توسعه رو برام آسون تر کرده بود و اینکه بیشتر ما با زبان جاوا اسکریپت کار داریم و کامیونیتی که ری اکت داره خیلی بیشتر هست و بیشتر ابهامات اون برطرف شده از همه مهم تر هم اینکه وبسایت های بزرگی مثل فیسبوک و اینستاگرام از اون استفاده می کنند و توسعه میدهند آینده خیلی روشنی برای این کتاب خانه وجود دارد.

PWA یا اپلیکیشن های پیشرونده چیست؟

PWA مخفف Progressive Web Application می باشد. این فناوری از اوایل سال 2015 توسط شرکت گوگل معرفی شد. یک PWA به اصطلاح وب سایتی است که از فناوری های مدرن و جدید وب استفاده می کند اما ظاهر و کارکرد آن شبیه یک اپلیکیشن معمولی می باشد. به بیان ساده تر، PWA یک وب سایت می باشد که با استفاده از مرورگرهایی مثل کروم و فایرفاکس وارد آن شده و سپس با یک اپلیکیشن سروکار داریم.

ویژگی های اپلیکیشن های پیش رونده وب (PWA) :

- قابل استفاده در هر دستگاه و سیستم عاملی هستند به همین دلیل به آن ها پیش رونده می گویند.
- به دلیل اینکه اپلیکیشن های پیش رونده وب در اصل یک وب سایت می باشند، از طریق موتورهای جستجو قابل یافتن هستند.
- برخلاف اپلیکیشن های بومی، نیازی به نصب و مراحل پیچیده دانلود ندارند و با استفاده از یک URL می توان به راحتی آن ها را به اشتراک گذاشت.
- در ظاهر شبیه یک اپلیکیشن بومی هستند و رابط کاربری مشابه آن ها دارند.
- بدون اتصال به اینترنت و حتی با سرعت پایین اینترنت قابل استفاده هستند.
- اپلیکیشن های بومی نیازمند آپدیت از طریق فروشگاه های نرم افزاری هستند اما PWA ها به دلیل استفاده از Service Worker همیشه به روز می باشند و به محض اینکه کاربر به اینترنت وصل باشد و محتوای جدیدی انتشار داده شود، آن محتوا بلا فاصله در اختیار کاربر قرار می گیرد.
- اپلیکیشن های پیش رونده وب، در بستر HTTPS قرار دارند در نتیجه از نظر مسائل امنیتی بسیار ایمن هستند.
- این اپلیکیشن ها واکنش گرا (Responsive) و کاملا انعطاف پذیر می باشند.

بعد از تحریم هایی که در ایران به وجود آمد و مارکت های اپلیکیشن های موبایل مثل گوگل پلی و اپ استور تحریم شد توسعه این نوع اپلیکیشن ها توسط شرکت های داخلی بسیار رواج پیدا کرد و تقریبا همه شرکت های بزرگ مثل دیجی کالا ، اسنپ ، تیپسی ، ایرانسل و ... همگی اپلیکیشن های خودشون رو بر این بستر توسعه دادند.

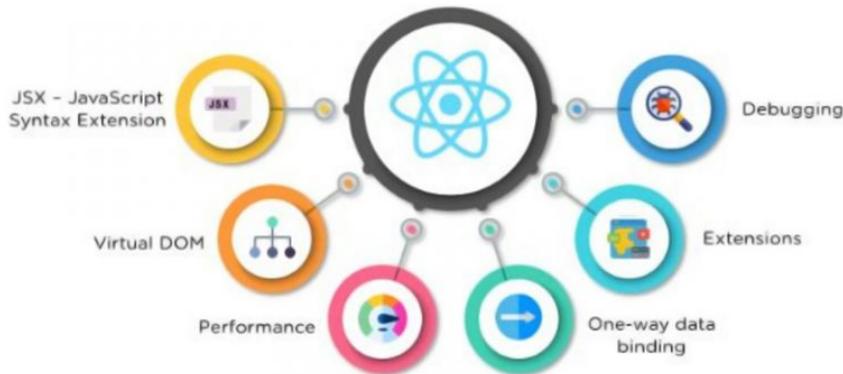
منبع اصلی استفاده شده در pwa

<https://web.dev/progressive-web-apps>

به طور خلاصه راجب وب اپلیکیشن ها و توسعه اپلیکیشن های pwa صحبت کردیم و سعی کردم با توضیح دادن درباره این ابزارها علت انتخابم از این ابزار ها برای توسعه این پروژه رو توضیح داده باشم در ادامه با جزیات بیشتر پروژه آشنا می شویم.

ویژگی های منحصر به فرد React

تا الان با هسته اصلی پیاده سازی پروژه آشنا شدیم حال نوبت آن رسید که با ویژگی های React آشنا شویم



1. استفاده از سینتکس JSX

react از سینتکسی به اسم JSX استفاده می کند. مخفف Javascript XML هست. برای این که نحوه نوشتن این سینتکس را یاد بگیرید باید با مفهوم XML آشنا باشید.

اگر بخواهیم خیلی ساده و خلاصه توضیح بدھیم، XML را همان HTML در نظر بگیرید اما با این تفاوت که در HTML تگ های از پیش تعیین شده وجود دارند که فقط مجاز به استفاده از این تگ ها هستیم و نمی توانیم از تگ هایی با اسم دلخواه استفاده کنیم. به عنوان مثال نمی توانیم از تگی به اسم <Amin> استفاده کنیم.

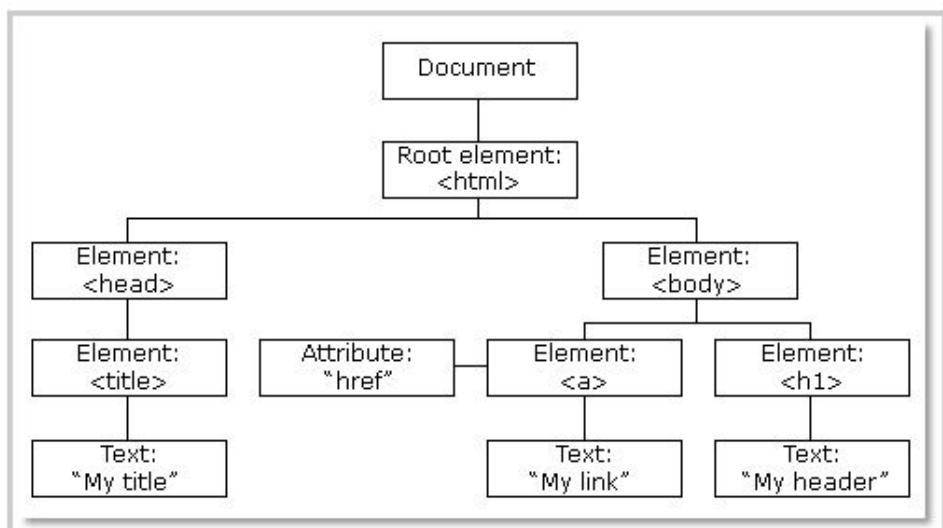
اما شما در XML می توانید تگ هایی با اسمی دلخواه را ایجاد کرده و از آنها استفاده کنید. داخل ری اکت هم شما می توانید قسمت های مختلف وب سایتتان را با اسم های دلخواه ایجاد کرده و استفاده کنید.

اکنون که دانستیم منظور از XML در تعریف ری اکت یعنی چه، به معنای jsx بپردازیم. به طور کلی jsx یعنی این که ما داخل صفحات جاوا اسکریپت کد هایی در قالب HTML و XML بنویسیم. به همین دلیل در پروژه هایی که با فریمورک ری اکت توسعه داده می شوند هیچ خبری از فایل HTML نیست و تمام کد های HTML توسط فایل های جاوا اسکریپت Render میشوند.

2. استفاده از Virtual DOM

در ابتدا بهتر است اول با مفهوم DOM آشنا شویم ، DOM مخفف کلمه Document Object Model می باشد یعنی ما با استفاده از زبان برنامه نویسی جاوا اسکریپت به محتوای HTML سایت دسترسی پیدا می کنیم و می توانیم یکسری event رو المنت های HTML اجرا کنیم. المنت هایی که در DOM وجود دارد به صورت ساختار درختی هستند و با هر تغییری که در DOM میدهیم همه فرزند های اون DOM دوباره از اول رندر می شوند.

ری اکت یک نسخه سبک از DOM اصلی را در حافظه نگه می دارد که به عنوان DOM مجازی یا (VDOM) نیز شناخته می شود. دستکاری DOM اصلی بسیار کنترل از دستکاری VDOM است. ری اکت می تواند با استفاده از VDOM در هنگام تغییر وضعیت یک شی در برنامه به جای بروزرسانی همه اشیا، فقط آن شی را در DOM اصلی تغییر دهد.



وقتی وضعیت یک شی در یک برنامه React تغییر می کند، VDOM به روز می شود. سپس حالت قبلی خود را مقایسه می کند و سپس به جای به روزرسانی همه اشیا، فقط آن اشیا موجود در DOM واقعی را به روز می کند. این باعث می شود همه چیز سریع پیش برود، به خصوص وقتی با سایر فناوری های Front-end مقایسه می شود که مجبور هستند همه اشیا را بروز کنند حتی اگر فقط یک Object در برنامه وب تغییر کند.

3. پرفورمنس بالای برنامه های React

ریکت از VDOM استفاده می کند، که باعث می شود برنامه های وب بسیار سریعتر از برنامه های توسعه یافته با فریم ورک های Front-end متناوب اجرا شوند React. یک رابط کاربری پیچیده را به اجزای جداگانه تقسیم می کند و به چندین کاربر اجازه می دهد تا همزمان روی هر مولفه کار کنند که در نتیجه آن سرعت توسعه را افزایش می دهند.

ری اکت یک لایبریری کامپوننت بیس هست. ممکن هست برایتان سوال شود که معنی و مفهوم کامپوننت چی هست؟! کامپوننت یعنی یک تکه کوچک از یک چیز بزرگ.

یعنی ری اکت وب سایت شما را تبدیل به تکه های کوچیک کرده و با قرار دادن این تکه های کوچک در کنار همدیگر، وب سایت شما ساخته می شود.

حال ممکن هست برایتان سوال شود که این Component Based بودن زبان برنامه نویسی react چه فایده ای دارد و چه قابلیت مثبتی به ما می دهد. تا اینجا مفهوم کامپوننت را اینطور گفتیم که یک تکه کوچک از وب سایت ما؛ مثل هدر، فوتر، باکس محصول، نوبار و

همان طور که می دانیم هدر و فوتر در همه صفحه های یک وب سایت یکسان هستند و تغییری نمی کنند. حال فرض کنید وب سایت ما 10 صفحه دارد و قطعاً فوتر همه صفحات باید یکی باشد. به جای این که تمام کد های فوتر را در همه 10 صفحه بنویسیم، فوتر را تبدیل می کنیم به یک کامپوننت، با اسم مثلا Footer.js.

از این به بعد در هر کدام از صفحه های وب سایتمان که به فوتر نیاز داشتیم، به جای تکرار چند خط کد های فوتر فقط اسم کامپوننت Footer را می نویسیم.

این موضوع خوانایی کدهای مان را افزایش می دهد، تست نویسی را راحت تر کرده، دیباگینگ را راحت تر می کند و ... بسیاری مزایای دیگر.

مثلا اگر شما سایتتان را با ریکت به صورت Component Based توسعه داده باشید و در قسمتی از وب سایتتان مشکل و باگی وجود داشته باشد، خیلی راحت اطلاع دارید که مشکل در کدام کامپوننت رخ داده و آن را حل می کنید و با قسمت های دیگر وب سایتتان درگیر نمی شوید و وقتتان را تلف نمی کنید .

(Extention) 4. افزونه ها

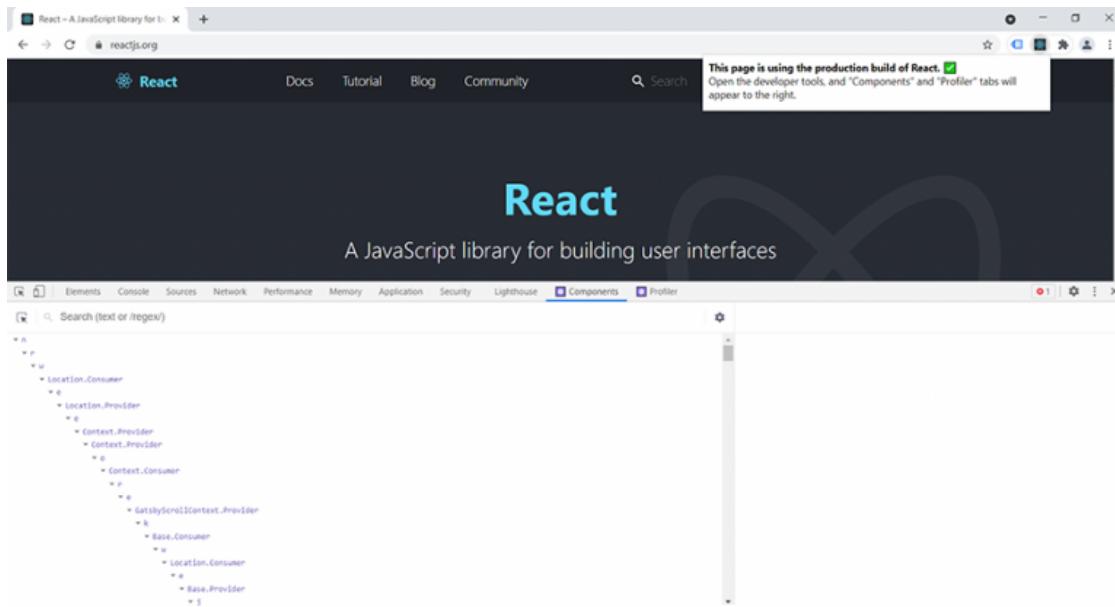
ری اکت فراتر از طراحی رابط کاربر ساده است و افزونه های بسیاری دارد که پشتیبانی کاملی از معماری برنامه را ارائه می‌دهند. به عنوان مثال امکان Server-Side Rendering را ارائه دهنده که یک برنامه تحت وب Client-Side در سمت سرور رندر می‌شود و سپس یک صفحه کامل به مرورگر کاربر ارسال کنند که برای مسائل مربوط به سئو می‌تواند مفید باشد یا استفاده از Flux و Redux و به طور گسترده در توسعه برنامه های وب و سرانجام، React Native که یک چارچوب محبوب مشتق شده از Cross-platform استفاده می‌شود.

(One-way Data Binding) 5. اتصال داده یک طرفه

اتصال داده یک طرفه ری اکت همه چیز را modular و سریع نگه می‌دارد. جریان داده یک طرفه به این معنی است که داده ها به یک جهت حرکت می‌کنند که این نشان دهنده Model to View است. اگر Model بروزرسانی شود View نیز بروزرسانی می‌شود و به دلیل استفاده ری اکت از DOM مجازی این عمل باعث خواهد شد فرآیند طراحی مجدد View بسیار کارآمد باشد.

(Debugging) 6. رفع باگ

تست و خطایابی برنامه های ری اکت به دلیل وجود یک جامعه بزرگ توسعه دهنده بسیار آسان است. همچنین فیس بوک یک افزونه کوچک مرورگر ارائه داده است که اشکال زدایی React را سریعتر و آسان تر می‌کند. به عنوان مثال، این افزونه برگه های Profiler و Components را در گزینه developer tools در مرورگر وب Chrome اضافه می‌کند که بررسی مستقیم اجزای استفاده شده در صفحه را آسان خواهد کرد.



برنامه های ReactJS به راحتی قابل تست هستند. به همین دلیل برنامه نویسان می توانند به وسیله پاس دادن پارامترها، state و خروجی ها، توابع و کامپوننت های خود را چک و ارزیابی کنند.

امروزه ری اکت جایگاه ویژه ای در توسعه برنامه های front-end پیدا کرده است که نشان دهنده استفاده مداوم آن در بین توسعه دهندگان است. با توجه به ارائه سریع برنامه های ری اکت، شرکت های بیشتری این ابزار توسعه را جایگزین روش های قدیمی کرده اند که متعاقباً این امر منجر به تقاضای بیشتر توسعه دهندگان در سراسر جهان شده است. React

فیسبوک بطور مدام ویژگی هایی را به React اضافه می کند و با گذشت زمان قدرت آن بیشتر شده است. همانطور که در نمودار روند Google مشاهده می کنید، همچنان توانسته است جایگاه اول خود را در سال 2022 حفظ کند.

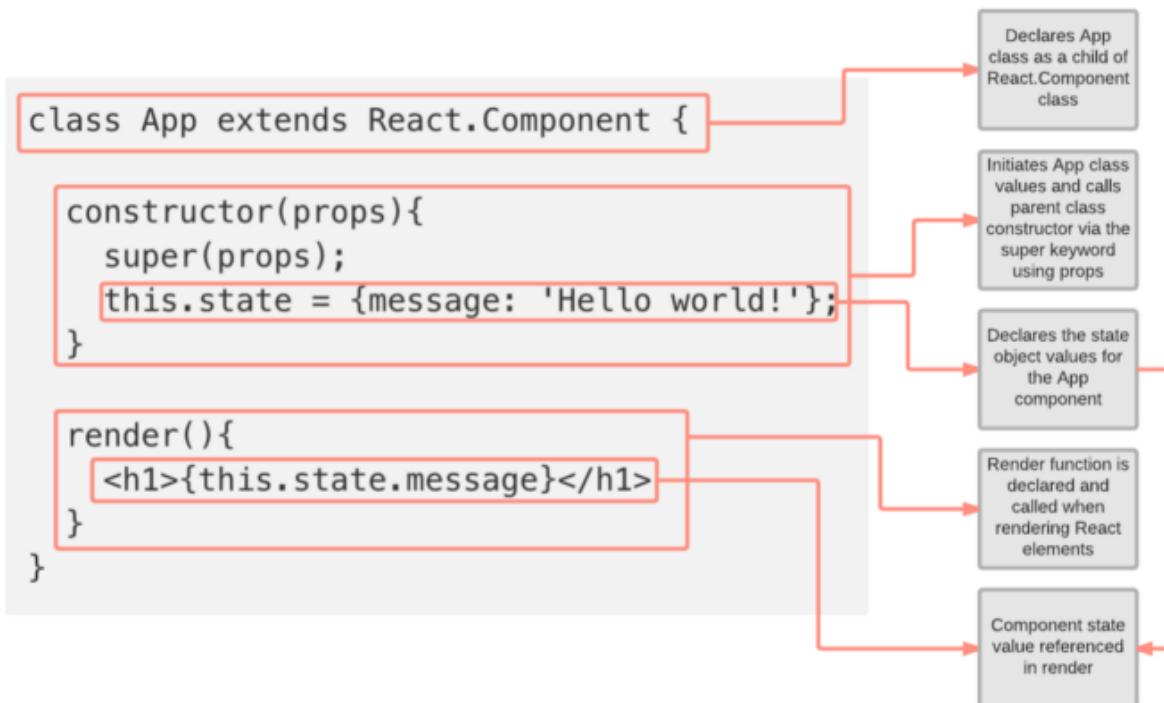
در ری اکت دو مفهوم اصلی وجود دارد که در تمام توسعه نرم افزار ما به آن ها نیاز داریم اولی State و دیگری Props می باشد که در ادامه بیشتر با آنها آشنا می شویم.

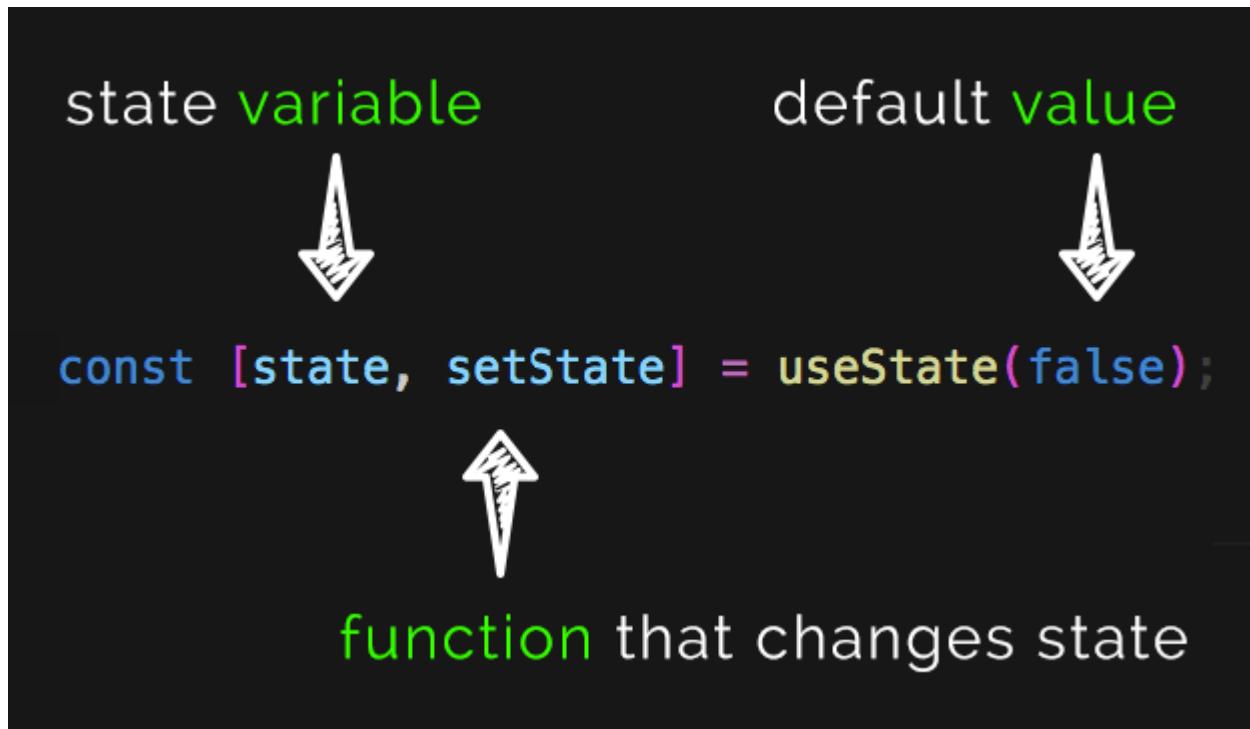
تعریف State در ری اکت

یک شی JavaScript است که داده های پویای یک کامپوننت را ذخیره می کند و رفتار آن را تعیین می کند. از آنجا که state پویاست ، این پویایی یک کامپوننت را قادر می سازد تا اطلاعات را بین رندرها رديابی کند و پویا و تعاملی باشد.

فقط در یک کامپوننت کلاس قابل استفاده است. اگر پیش بینی می کنید که یک کامپوننت به مدیریت نیاز دارد ، باید به عنوان یک کامپوننت کلاس ایجاد شود نه یک کامپوننت کاربردی . برای تغییر دادن مقادیر state می توانیم ازتابع setState استفاده کنیم.

یک component باید در طول عمر غالب باشد ، بنابراین ابتدا باید مقداری state اولیه داشته باشیم ، برای این منظور باید state را در سازنده کلاس component تعریف کنیم. برای تعریف وضعیت هر کلاس می توانیم از قالب نمونه زیر استفاده کنیم.





هرگز نباید به طور مستقیم به روز شود state

از یک شی قابل مشاهده به عنوان state استفاده می کند که مشاهده می کند چه تغییراتی در state ایجاد می شود و به کامپوننت کمک می کند تا بر طبق آن رفتار کند. به عنوان مثال ، اگر state هر یک component ای مانند صفحه زیر را به روز کنیم مجدداً نمایش داده نمی شود زیرا React State نمی تواند تغییرات ایجاد شده را تشخیص دهد.

```

;this.State.attribute= "new-value"

```

بنابراین ، متده است setstate() خود را ارائه می دهد. متده setState() یک پارامتر را می گیرد و انتظار دارد یک شی object که باید مجموعه ای از مقادیر باشد را به روز کند. پس از به روزرسانی ، روش به طور ضمنی روش render() را فراخوانی می کند تا صفحه را دوباره رنگ کند.

تنها زمانی که ما مجاز به تعریف صريح state هستیم در سازنده است که state اولیه را ارائه می دهد.

بسیار کارآمد است و بنابراین از به روزرسانی های state ناهمگام استفاده می کند ، یعنی React است چندین به روزرسانی setState() را به صورت یکجا به روز کند. بنابراین استفاده از مقدار state فعلی ممکن است همیشه نتیجه دلخواه را ایجاد نکند .

به روزرسانی های state مستقل هستند

ممکن است شی state از یک کامپوننت چندین ویژگی داشته باشد و React اجازه می دهد تا از تابع setState() برای به روزرسانی فقط زیرمجموعه ای از این ویژگی ها و مانند چندین روش () برای به روزرسانی هر مقدار صفت به طور مستقل استفاده شود.

چه تفاوتی بین state و props در ریکت وجود دارد

از Props برای انتقال داده ها بین اجزای React استفاده می شود. جریان داده های React بین کامپوننت ها یک جهته است فقط از والدین به فرزند.

کامپوننت ها داده ها را از خارج با props دریافت می کنند ، در حالی که می توانند داده های خود را با state ایجاد و مدیریت کنند Props. برای انتقال داده استفاده می شود ، در حالی که state برای مدیریت داده است. داده های تهیه شده از props فقط خواندنی است و توسط کامپوننتی که آن را از خارج دریافت می کند، قابل اصلاح نیست. داده های state را می توان توسط کامپوننت های خود اصلاح کرد ، اما خصوصی است .

از خارج قابل دسترسی نیست
props فقط از والدین به فرزند منتقل می شوند. جریان یک جهته
تغییر state باید با روش setState() اتفاق بیفتد.

بررسی دقیق و کامل چرخه‌ی حیات (Lifecycle) در ری‌اکت

در اطراف ما همه چیز از طریق یک چرخه‌ی تولد شروع می‌شود، بعد از تولد شروع به رشد (به‌روزرسانی) می‌کند و در یک نقطه نابود می‌شود مانند درختان، یک نرم‌افزار، خود ما انسان‌ها و یا یک Component در ری‌اکت، همه‌ی این موارد به وجود می‌آیند، رشد و تغییر می‌کنند و در یک نقطه نابود می‌شوند.

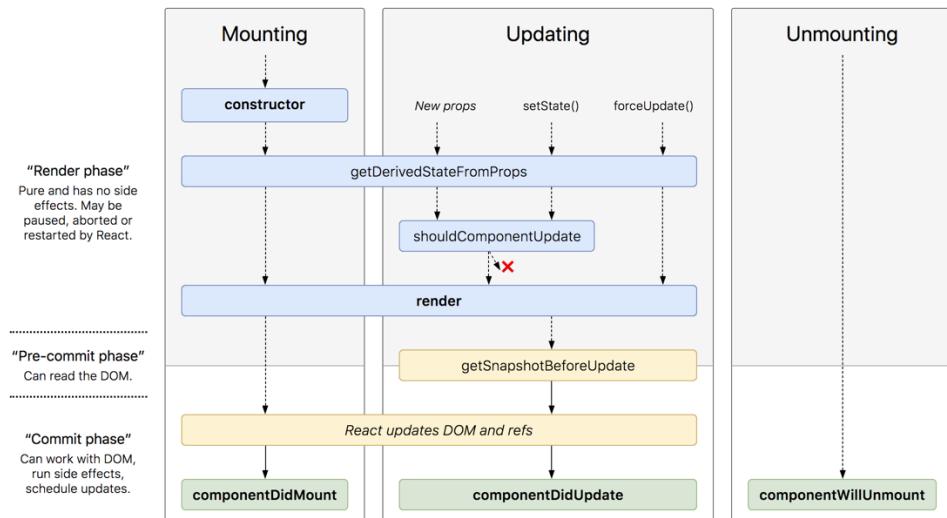
متدهای چرخه‌ی حیات (Lifecycle)، متدهایی هستند که برای دسترسی به هر یک از مراحل تولد تا نابودی یک Component مورد استفاده قرار می‌گیرند. فرض کنید می‌خواهید یک برنامه مشابه یوتیوب بسازید. که این برنامه از اینترنت برای لود کردن ویدئو و از باتری به عنوان منبع تغذیه استفاده می‌کند.

اگر کاربر در حال تماشای ویدئو در این برنامه باشد و بطور هم زمان یک برنامه دیگر را باز کند در اینجا ما باید مطمئن باشیم که از باتری و اینترنت به کارآمدترین روش ممکن استفاده می‌کنیم. یعنی وقتی کاربر وارد یک برنامه‌ی دیگر می‌شود ما باید مطمئن شویم که پخش فیلم قطع شده و در نتیجه از اینترنت و باتری استفاده نشود.

این کار را می‌توانیم با استفاده از متدهای چرخه‌ی حیات در ری‌اکت انجام دهیم.

وقتی در یک پروژه تعداد component‌ها زیاد می‌شود. این خیلی مهم است که هر component وقتی کار خود را انجام داد نابود شود و دیگر از منابع استفاده نکند.

با یادگیری چرخه‌ی حیات و تمرین درست می‌توانیم برنامه‌هایی با کیفیت بالا و کارایی مناسب بسازیم.



به طور کلی می‌توان چرخه حیات در ری‌اکت را به سه مرحله تقسیم کرد:

- نصب کردن (**Mounting**)
- بهروزرسانی (**Updating**)
- نابودی (**Unmounting**)

خوب وقت شه که تک تک این مراحل رو با هم بررسی کنیم:

مرحله اول: نصب کردن

در ری‌اکت نصب کردن به معنی بارگذاری component در DOM است.

این مرحله شامل مجموعه‌ای از متدها است که موقع مقدار دهی اولیه component و بارگذاری آن در DOM فراخوانی می‌شوند.

متدها به ترتیب زیر فراخوانی می‌شوند:

Constructor . 1

هر وقت یک component ساخته می‌شود اولین متدهی که فراخوانی می‌شود constructor است. این متدهی فقط یکبار در طول چرخه حیات فراخوانی می‌شود. ما از این متدهی برای تعریف مقادیر state استفاده می‌کنیم.

```
1 constructor() {  
2     super();  
3     this.state = {  
4         name: 'Jon Snow',  
5         age: 22  
6     }  
7 }
```



در ری‌اکت constructor تنها جایی است که می‌توان از this.state استفاده کرد، در بقیه متدها فقط می‌توان با استفاده از this.setState آن را تغییر داد.

Static getDerivedStateFromProps . 2

متد getDerivedStateFromProps درست قبل از render فراخوانی می‌شود. این متد می‌تواند یک Object برای بهروزرسانی state و یا مقدار null را برای جلوگیری از بهروزرسانی بازگردد.

ما وقتی از این متد استفاده می‌کنیم که بخواهیم state خود را به وسیله‌ی props بهروزرسانی کنیم.

این یک متد static است پس به کلید واژه‌ی this دسترسی ندارد. این متد می‌تواند به state و props دسترسی داشته باشد. از این رو وقتی یک state وابسته به props تعریف می‌کنیم با استفاده از این متد می‌توانیم آن را بهروزرسانی کنیم.

با پاس دادن آرگومان‌های nextProps و prevState به getDerivedStateFromProps جدید موجود دسترسی پیدا کردیم و سپس با استفاده از دستور if چک می‌کنیم که اگر مقدار state جدید با state موجود متفاوت بود آن را بهروزرسانی کن و اگر با هم مساوی بودند نیاز به بهروزرسانی نیست و مقدار null را بازگردداند می‌شود.

Render . 3

این یک متد ضروری در component های ریاکت است، در این متد المنت‌ها آماده‌ی نصب در DOM می‌شوند.

```
1 render() {  
2     return (  
3         <div>  
4             <h1>Hello World!</h1>  
5         </div>  
6     )  
7 }
```

همانطور که در مثال بالا مشاهده می‌کنید، متد render یک JSX را برای نمایش در UI باز می‌گرداند. یک متد render می‌تواند وقتی چیزی برای نمایش ندارد مقدار null را بازگردداند.

ComponentDidMount . 4

بعد از این که ما برای اولین بار component render خود را کردیم این متد فراخوانی می‌شود.

در این متد ما می‌توانیم به DOM دسترسی داشته باشیم، اگر هم بخواهیم یک کتابخانه مانند D3 یا Highcharts به پروژه‌ی خود اضافه کنیم از این متد استفاده می‌کنیم.

```
1 componentDidMount() {  
2   fetch('url')  
3     .then(response => response.json() );  
4     .then(data => this.setState({ info: data }) );  
5 }
```

یکی دیگر از مهم‌ترین کاربردهای این متد استفاده از آن برای درخواست‌های Ajax و API است.

```
1 componentDidMount() {  
2   fetch('url')  
3     .then(response => response.json() );  
4     .then(data => this.setState({ info: data }) );  
5 }
```

مرحله دوم: بهروزرسانی

static getDerivedStateFromProps

همان گونه که در تصویر بالا نمودار چرخهی حیات را مشاهده می‌کنید. می‌توانید ببینید که متدهای `getDerivedStateFromProps` در مرحله اول و دوم مشرک است. البته کارایی این متدهای مرحله بیشتر است.

اگر احتیاج دارید که `state` خود را با تغییرات `props` بهروزرسانی کنید. می‌توانید با بازگرداندن یک `Object` در این متدهای این کار را انجام دهید.

* البته ناگفته نماند که معمولاً بروزرسانی `state` توسط `props` توصیه نمی‌شود. و باید به عنوان آخرین راه حل از آن استفاده کرد.

shouldComponentUpdate

زمانی که یک `props` جدید دارید یا از `(component) setState()` استفاده می‌کنید، شما به طور خودکار توسط ری‌اکت دوباره رندر می‌شود. تا شما بتوانید تغییرات را در آن مشاهده کنید.

اما با استفاده از این متدهای می‌توانید به `component` خود اجازه ندهید که دوباره رندر شود.

به طور کلی این متدهای مفید باشد که شما نمی‌خواهید ری‌اکت تغییرات جدید `state` و `props` را رندر کند.

شما نمی‌توانید در این متدهای `setState()` برای بهروزرسانی `state` استفاده کنید.

```
1 | shouldComponentUpdate(nextProps, nextState) {  
2 |     return nextProps.title !== nextState.title || this.state.input !=  
3 | }
```

همان طور که در مثال بالا مشاهده می‌کنید این متدهای مقادیر `nextProps` و `nextState` را به عنوان آرگومان می‌پذیرد و باید یک `Boolean` را در جواب سوال "آیا باید دوباره رندر کنم؟" بازگرداند که به طور پیش فرض مقدار `true` را باز می‌گرداند.

render

در این بخش component شما دوباره رندر می‌شود.

getSnapshotBeforeUpdate

متد getSnapshotBeforeUpdate یکی دیگر از متدهای چرخهٔ حیات ریاکت است که به تازگی معرفی شده و برای امنیت بیشتر جایگیزین متد componentWillUpdate شد.

```
1  getSnapshotBeforeUpdate(prevProps, prevState) {  
2      // ...  
3  }
```

این متد درست قبل از بهروزرسانی DOM فراخوانی می‌شود. هر مقداری که بازمی‌گرداند به عنوان آرگومان به متد componentDidUpdate پاس داده می‌شود.

یادتون باشه که این متد به ندرت استفاده می‌شود یا اصلاً استفاده نمی‌شود. مگر این که بخواهید به یکی از های DOM دسترسی داشته باشید و اون رو به عنوان آرگومان به componentDidUpdate() پاس بدهید.

componentDidUpdate

حالا که به این بخش رسیدیم یعنی تغییرات ما در DOM اعمال شد. بیشترین مورد استفاده از این متد برای بهروزرسانی DOM با تغییرات props و state است.

در این متد ما به سه چیز دسترسی داریم، props موجود، State موجود و مقداری که متد getSnapshotBeforeUpdate

```
1  componentDidUpdate(prevProps, prevState, snapshot) {  
2      if (this.props.userName !== prevProps.userName) {  
3          this.fetchData(this.props.userName);  
4      }  
5  }
```

در مثال بالا ما مقدار props جدید را با props موجود مقایسه کردیم و اگر مقدار آن متفاوت بود آن را به `fetchData` پاس می‌دهیم.

مرحله 3: نابودی

این مرحله فقط شامل یک متده است. همان طور که از نامش پیداست این متده قبل از نابود شدن component اجرا می‌شود. اگر نیاز به هرگونه عملیات پاکسازی برای component خود دارید می‌توانید از این متده استفاده کنید.

شما در این متده اجازه استفاده از `setState` برای تغییر state را ندارید.

به عنوان مثال برای این متده، وقتی که می‌خواهید یک ثانیه شمار بسازید و از `setInterval` استفاده می‌کنید باید راهی وجود داشته باشد که وقتی دیگر از این component در پروژه‌ی خود استفاده نمی‌کنیم بتوانیم جلوی اجرا شدن آن را با `clearInterval` بگیریم.

```
1 this.state = {  
2     stopWatch: 0  
3 }  
4  
5 stopWatchHandler = () => {  
6     this.setState(prevState => {  
7         return {  
8             stopWatch: prevState.stopWatch + 1  
9         }  
10    });  
11 }  
12  
13 componentDidMount() {  
14     setInterval(this.stopWatchHandler, 1000);  
15 }  
16  
17 componentWillUnmount() {  
18     clearInterval(this.stopWatchHandler);  
19 }
```

برای ساخت ثانیه شمار در مثال بالا ابتدا یک متغیر state با نام stopWatch و مقدار اولیه 0 در کامپوننت خود تعریف کرده‌ایم. سپس در متد stopWatchHandler یک واحد به مقدار stopWatch اضافه می‌کنیم. مرحله‌ی بعد در متد componentDidMount با استفاده از setInterval متد stopWatchHandler را هر یک ثانیه اجرا می‌کنیم و مقدار stopWatch را یک واحد افزایش می‌دهیم.

حالا فرض کنید که ثانیه شمار بخشی از پروژه شما است. که پس از استفاده دیگر به آن احتیاج ندارید اگر تابع clearInterval را با استفاده از clearInterval غیرفعال نکنید این تابع هر یک ثانیه اجرا می‌شود و یک واحد به stopWatch اضافه می‌کند. این یعنی حتی وقتی که دیگر نیازی به ثانیه شمار در پروژه‌ی خود ندارید، ثانیه شمار همچنان در حال اجرا در پروژه‌ی شما است و از منابع استفاده می‌کند.

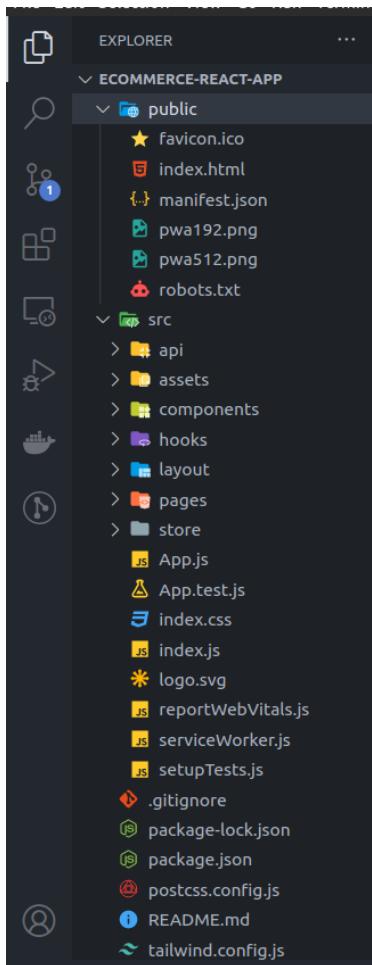
برای جلوگیری از این کار می‌توانید از متد componentWillUnmount استفاده کنید. این متد آخرین مرحله از چرخه‌ی حیات در ری‌اکت می‌باشد که در آن می‌توانید clearInterval را اجرا کرده و از اجرای ثانیه شمار در موقعی که به آن احتیاج ندارید جلوگیری کنید.

ساختار فolder بندی پروژه فروشگاه اینترنتی

علی‌رغم اینکه در پروژه‌های ریکت ساختار استاندارد و مشخصی برای فایل‌ها و فولدرها وجود ندارد اما بهتر است طبق یک ساختار مناسب و بهینه کدنویسی پروژه را شروع کنیم. زیرا اگر در آینده بخواهیم ساختار و مسیر فایل‌ها و فولدرهای پروژه ریکتی را تغییر دهیم، نیاز است کدهای نوشته شده را مجدداً بازبینی کرده و اصلاحات لازم را انجام دهیم.

پس پیشنهاد می‌شود قبل از شروع کدنویسی پروژه، به یک ساختار بهینه برای منابع اپلیکیشن ریکت خود برسیم تا در زمان و انرژی صرفه جویی کرده باشیم.

اگر file structure در پروژه ریکت بهینه باشد، فرقی ندارد که توسط ReactJS یک اپلیکیشن تحت وب تولید می‌کنیم یا توسط React-Native یک اپلیکیشن موبایلی. در یک تیم دو نفره کار می‌کنیم یا در یک تیم 10 نفره، با نسخه‌های قدیمی ریکت کد می‌زنیم یا با نسخه‌های جدید.



با اجرای دستور `npx create-react-app app-name` می توانید یک پروژه ریکت جی اس راه اندازی کنید و نیازی به هیچ کار دیگری ندارید!

```
1. my-app
2.   └── build
3.   └── node_modules
4.   └── public
5.     └── favicon.ico
6.     └── index.html
7.     └── manifest.json
8.   └── src
9.     └── App.css
10.    └── App.js
11.    └── App.test.js
12.    └── index.css
13.    └── index.js
14.    └── logo.svg
15.    └── serviceWorker.js
16.   └── .gitignore
17.   └── package.json
18.   └── README.md
```

پوشه `build` شامل فایل های نهایی و به اصطلاح `production` پروژه شما می باشد. این فolder تا وقتی که دستور `yarn build` یا `npm build` را اجرا نکنید وجود ندارد. پس از بیلد کردن پروژه، این فolder بطور اتوماتیک ساخته می شود. فایل ها و فolderهایی که در پوشه `build` قرار دارند آماده پابلیش بر روی هاست هستند.

پوشه `node_modules` شامل تمام پکیج های نصب شده توسط دستور `yarn add` یا `npm install` می باشد.

پوشه `public` محل قرارگیری فایل های استاتیک پروژه است. فایل هایی که در این فolder هستند با همین نام در پوشه `build` یعنی نسخه `production` قرار خواهند داشت. معمولاً فایل های داخل `public` در مرورگر کش می شوند و نیازی نیست هر بار توسط مرورگر کاربر دانلود شوند. فایل هایی که مانند `index.html` و `manifest.json` و `index.css` نام آنها مهم است و نباید تغییر کند باید در فolder `src` `robots.txt` درج شوند.

پوشه `src` در این پوشه فایل های داینامیک قرار می گیرند. اگر فایلی توسط کدهای جاوا اسکریپت `import` می شود یا توسط کدهای `js` محتوا ایش تغییر می کند در این پوشه قرار دهید. فایل های داینامیک معمولاً نباید کش شوند. به این منظور می توان از ابزارهایی مانند `webpack` استفاده کرد. `webpack` پک می تواند یک `cache` را به انتهای نام هر فایل اضافه کند تا از کش خوانده نشوند. مانند `hash-code` `banner-hash-code.jpg` بجای `hash-code` یک رشته رندوم قرار خواهد گرفت.

یکی از مواردی که معمولاً در ساختار پروژه های ریکت رعایت می شود، تعریف کامپوننت های اپلیکیشن در فولدری به نام components می باشد. البته برخی از توسعه دهنگان ری اکت از دو فولدر برای کامپوننت های برنامه استفاده می کنند: یک فولدر برای کامپوننت های stateless و فولدری برای stateful

اما پیشنهاد ما استفاده از یک فولدر کلی بنام components می باشد که تمام اجزا یا کامپوننت های برنامه داخل آن تعریف شود.

منابع assets پروژه مانند تصاویر، فونت ها یا فایل های sass می توانند در فولدر مجزای خودشان تعریف شوند. مثلا همانطور که در ساختار فوق مشاهده می کنید، تصاویر در فولدر images درج خواهند شد. و به همین ترتیب فایل های sass می توانند در فولدر styles تعریف شوند. یعنی اپلیکیشن فقط یک فولدر برای تصاویر دارد و هر زمان که نیاز به تغییر عکس یا درج یک عکس جدید باشد، فقط کافیست به این فولدر مراجعه شود. یا اگر وب اپلیکیشن شما چند زبانه است می توانید فایل های مربوط به ترجمه زبان ها را در یک فولدر بنام locales تعریف کنید.

مانند src/utils یا src/helpers این پوشه ای است که شامل توابع helper می باشد که بصورت گلوبال در پروژه مورد استفاده قرار می گیرند.

در این صورت، بخشی از اپلیکیشن شما که کد آن در جاهای مختلف برنامه مورد استفاده قرار می گیرد نیاز به کپی پیست نخواهد داشت و از یک فولدر مثلا بنام utility تمامی کدها خوانده می شود.

نام این پوشه دلخواه است و بنا به نیازتان می توانید اسمی مناسب انتخاب کنید. اما چیزی که مهم است ایجاد یک فایل بنام index.js در این فولدر، مثلا utility است که شامل دستور export تمام فایل های js موجود در فولدر باشد.

در نهایت پیشنهاد می کنیم مازول های مستقل و مجزای پروژه خود را در یک فولدر مثلا با نام src/shared یا src/packages تعریف کنید تا بتوانید در سایر پروژه های خود یا سایر تیم ها از آن استفاده کنید.

بر خلاف فولدرهای components و utils در این فولدر، مثلا shared packages یا index.js نیازی به فایل shared نیست. زیرا معمولاً نمی خواهیم از تمام مازول های export شده استفاده کنیم. بلکه بر اساس نیاز می خواهیم یک کامپوننت را ایمپورت کنیم.

دایرکتوری components در پروژه ریکت:

ساختار پوشش components شبیه سایر پوشش های پروژه ریکت می باشد. همانطور که می دانید یک اپلیکیشن react از چندین کامپوننت تشکیل شده است که بهتر است در مسیر `src/components/component-name` تعریف شوند.

بطور معمول یک کامپوننت بیش از یک فایل دارد. بعنوان مثال هر کامپوننت معمولاً شامل فایل های CSS یا SASS می باشد. یک کامپوننت امکان دارد دارای کامپوننت فرزند یا Child باشد Component.

به همین دلیل است که فolder components در ریکت در اغلب موارد بیش از یک فایل component.js دارد و احتمالاً فایل ها و پوشش های مختلفی در هر فolder کامپوننت وجود دارد.

استفاده از هوک ها در ری اکت:

در این حالت احتمالاً یک کامپوننت root در مسیر `src/components/component-name` وجود دارد. توسط هوک hook در ری اکت می توانید local state و global state را مدیریت کنید. معمولاً به-HOC: High-Order-Component نیازی ندارید.

به منظور رفع مشکل یکسان بودن نام کامپوننت ها و عدم خوانایی نام کامپوننت ها در ادیتور، نام هر فایل کامپوننت را بصورت component-name.js تعریف می کنیم و در یک فایل index.js این کامپوننت را import کنیم.

برای راحتی تست پروژه پیشنهاد می کنیم هوک های مربوط به هر کامپوننت را در فایلی بنام useComponentName تعریف کنید. ساختار درختی پروژه در حالتی که از Hook استفاده می کنیم بصورت زیر است:

1.	my-app
2.	└ src
3.	└ components
4.	└ component-name
5.	└ hooks
6.	└ index.js
7.	└ use-component-name.js
8.	└ component-name.css
9.	└ component-name.scss
10.	└ component-name-styles.js
11.	└ component-name.js
12.	└ index.js
13.	└ index.js

درست مانند index.js فولدر هوک هم می تواند شامل یک فایل src/utils و src/components باشد که تمام هوک های کامپوننت در آن export شده اند.

جمع بندی ساختار پروژه ری اکت در صورت استفاده از Hook

- یک فایل ورودی یا مدخل entry.js: هوک ها در آن export شده اند.
- یک هوک واحد است که تمام هوک ها در آن فراخوانی شده اند که در کامپوننت مربوطه استفاده شده اند.
- یک فایل استایل component-name.css: هوک است که در کامپوننت مربوطه import شده است.
- یک SASS component-name.scss: هوک است که در کامپوننت مربوطه import شده است.
- مدخل ورودی یا entry point کامپوننت است که در آن export شده است و هر جای برنامه که بخواهیم از این کامپوننت استفاده کنیم، این فایل را import می کنیم.

```
1. | export { default as ComponentName } from './component-name';
```

عدم استفاده از هوک های ریکت:

در فرآیند توسعه اپلیکیشن، باید بررسی کنیم که آیا یک کامپوننت state خود را دارد یا آن بصورت گلوبال در global store مدیریت می شود یا یک کامپوننت والد props و state ها را به این کامپوننت ارسال می کند؟

ممکن است یک کامپوننت به هریک از روش های مذکور کار کند. بدلیل خوانایی فایل ها و لزوم جداسازی بخش های مستقل از یکدیگر، هر یک از موارد مذکور باید در کامپوننت مجزای خودشان تعریف شوند و فایل مستقل داشته باشند اما ما فقط component-name/index.js را داریم. بنابراین به چه شکل باید تعریف شود؟

راه حل ریفکتور یا refactor کردن می باشد. state ها در یک کامپوننت ممکن است از طریق یک global state مدیریت شود. مانند ریداکس. همچنین ممکن است state های یک کامپوننت از طریق props از کامپوننت والد ارسال شوند. یا در حالت سوم ممکن است state ها به شکل لوکال و محلی در کامپوننت کنترل شوند.

در نهایت هم ممکن است یک کامپوننت اصلا state نداشته باشد. در حالتی که کامپوننت از یک global state استفاده می کند مجبور هستیم کل فایل را از index.js متنقل کنیم و component-name-view.js را به global state متصل کنیم.

مسلماً جابجایی فایل‌ها در پروژه دشوار است و احتمال دارد دچار خطا در برنامه شویم.

کامپوننت stateless را نیز در فایل component-name-view.js تعریف کنید.

کامپوننت container را در فایل component-name-container.js تعریف کنید.

گلوبال در component-name-redux.js state تعریف شود.

و در نهایت هر کدام از موارد فوق را که در کامپوننت نیاز داشتید می‌توانید export کنید. اگر کامپوننت شما قادر state باشد فقط view.js را اکسپورت می‌کنید. اگر به state نیاز پیدا کردید container.js را خواهید داشت. فقط کافیست در دستور export کلمه view را با container view کلمه export جایگزین کنید. در صورتی که به یک global نیاز داشتید فقط کافیست redux.js را در فایل index.js اکسپورت کنید. در این حالت فقط کافیست در دستور export کامپوننت، کلمه container.js را با redux.js جایگزین کنید. به همین راحتی!

ساختار سلسله مراتبی فوق خلاصه مواردیست که تا اینجا ذکر کردیم:

- آن component-name-container.js: state management شامل business logic کامپوننت و
- می‌باشد.
- در component-name-redux.js: mapDispatchToProps و mapStateToProps شامل
- ریداکس می‌باشد. اما ممکن است شما بجای Redux از یک state manager دیگر مثلاً mobX استفاده کنید. در این حالت نام فایل را می‌توانید component-name-mobx.js در نظر بگیرید. با این روش نام گذاری این امکان را خواهید داشت که برای از چندین global state در اپلیکیشن خود بهره ببرید. و هر زمان که لازم باشد از mobx یا redux استفاده کنید.
- component-name-view.js: stateless-component معادل کامپوننت بدون state یا state می‌باشد. در اغلب موارد این کامپوننت باید بصورت تابعی یا functional تعریف شود.
- index.js: entry point کامپوننت شما می‌باشد. محتوای آن چیزی نیست به جز فایل‌های موردنیاز. که در بالا بطور مفصل توضیح دادیم.

دایرکتوری component-name می‌تواند شامل زیرفولدرهای مختص خود باشد. مانند utils منظورمان فولدرهایی است که فقط مختص این کامپوننت هستند و در سایر بخش‌ها و کامپوننت‌های برنامه کاربردی ندارند. در اینصورت تست واحد یا unit test برنامه سریع‌تر و راحت‌تر خواهد شد.

استفاده از ریکت روتر React Router

یکی دیگر از دایرکتوری هایی که احتمالا در اغلب اپلیکیشن های ریکتی وجود دارد `src/routes` است. بعنوان مثال اگر یک صفحه در مسیر `amahdavi.dev/portfolio` واقع باشد، کامپوننت مسیریابی یا `routing` در مسیر `src/routes/portfolio/index.js` قرار خواهد داشت.

قابلیت تست اپلیکیشن ری اکت:

در نهایت برای سادگی اجرای تست واحد unit test یا بهتر است فایل های مرتبط در کنار هم قرار داشته باشند. در اینصورت، جابجایی بین فایل های کامپوننت و فایل های تست ساده خواهد بود. همچنین import آنکردن فایل های تست و هماهنگی بین فایل های تست و فایل های کامپوننت نیز ساده تر خواهد بود.

صفحاتی که در این پروژه پیاده سازی و اجرا شدند عبارتند از :

- **صفحه اصلی فروشگاه اینترنتی**

در این صفحه امکاناتی همچون اسلایدر توضیح محصولات ، اطلاعات کلی فروشگاه ، دسته بندی کلی محصولات (موبایل و لوازم جانبی موبایل) ، اسلایدر برند ها ، وبلاگ

- **صفحه ورود**

بدلیل نبود api صفحه به صورت استاتیک طراحی شده است.

- **صفحه ثبت نام**

بدلیل نبود api صفحه به صورت استاتیک طراحی شده است.

- **صفحه سبد خرید**

در این صفحه اگر محصولی اضافه نشده باشد می نویسد سبد خرید خالی است اما اگر محصولی رو اضافه کنیم اطلاعاتی همچون تعداد محصولات ، مشخصات آدرس خریدار ، قیمت کل و مالیات داخل آن نوشته شده است

- **صفحه دسته بندی محصولات**

در این صفحه محصولات هر دسته بندی قابل مشاهده است و همه محصولات سایت رو می توانیم به طور کامل مشاهده کنیم در این صفحه من از filter و sort محصول استفاده کرده ام

- **صفحه جزئیات محصول**

در این صفحه عکس محصول ، قیمت محصول ، برند محصول ، کد محصول و اطلاعات ارسال محصول نوشته شده و محصول را داخل این صفحه هم می توانیم به سبد خرید اضافه کنیم.

- **صفحه وبلاگ**

در این صفحه مقالات وبسایت نوشته می شود

در این پروژه از پارادایم برنامه نویسی functional programming استفاده شده است و تمام کدها به صورت component base می‌باشد. یعنی هر بخش سایت به صورت جدا جدا نوشته شده است مانند تیکه‌ای از پازل و از اون می‌توانیم داخل پروژه‌های دیگر نیز استفاده کنیم. برای مثال جستجو، آیتم محصول، برندها و ... همگی به صورت کامپوننت مجزا طراحی شده است

در ری اکت چیست؟ Layout

اکثر وب‌اپلیکیشن‌ها لی‌آوت یکسانی در تمامی صفحات خود ندارند. برخی صفحه‌ها هدر و فوتر دارند، برخی صفحه‌ها ناوبری سمت چپ دارند، برخی صفحه‌ها هم این فوتر یا ناوبری چپ را ندارند. یک روش برای پیاده‌سازی این وضعیت پنهان کردن یا نمایش این بخش‌ها بر مبنای نوع صفحه است، اما این روش بهینه‌ای محسوب نمی‌شود.

راه حل کارآمد برای وضعیتی که توصیف کردیم، تعریف کردن لی‌آوت یا قالب‌های مختلف و هدایت به مسیرهای صحیح به وسیله React Router است.

همه ما با اصل DRY که اعلام می‌کنند باید از کد تکراری در پروژه‌ها استفاده کنیم، آشنا هستیم. زمانی که لی‌آوت‌ها را برای مسیرها تعریف می‌کنیم، در واقع کارکردهای مشترک زیادی از قبیل هدر، فوتر، ناوبری و غیره را از صفحه جدا می‌کنیم. به این ترتیب اگر بخواهیم چیزی را در هدر تغییر دهیم، این تغییر را باید تنها در یک بخش انجام دهیم.

هر اپلیکیشن از لی‌آوت‌های یکسانی در همه صفحه‌های خود استفاده نمی‌کند. در اغلب موارد باید لی‌آوت‌های متفاوتی برای صفحه‌های گوناگون اپلیکیشن پیاده‌سازی کنیم. ما می‌توانیم به کمک کتابخانه react-router، لی‌آوت‌های مختلفی در اپلیکیشن ری‌اکت پیاده‌سازی کنیم. ابتدا باید مسیر context را تعریف کنیم و همچنین لی‌آوت را برای هر مسیر context تعریف کرده باشیم. زمانی که مسیرهای context را تعریف کردیم، همه مسیرهای فرعی باید در فایل لی‌آوت تعریف شوند به این ترتیب معماری اپلیکیشن ری‌اکت به مقدار زیادی تمیزتر می‌شود.

تعریف کردن لی‌آوت‌ها در اپلیکیشن به روش مطرح شده در این مقاله موجب می‌شود که اپلیکیشن‌های پیچیده ساده‌تر شوند. همچنین موجب می‌شود که اصل (DRY) عدم استفاده از کدهای تکراری) در اپلیکیشن رعایت شود. زمانی که این لی‌آوت‌ها پیاده‌سازی شدند، ایجاد تغییر در اپلیکیشن از قبیل قرار دادن مسیر در لی‌آوت‌های مختلف یا تغییر ظاهر اپلیکیشن و مواردی از این دست ساده‌تر خواهد بود.

۳ نوع لیوت در این پروژه استفاده شده است که یکی مربوط به صفحه دسکتاب که با لپ تاپ میریم هست و دو تای دیگر مربوط به اپلیکیشن موبایل می باشد:

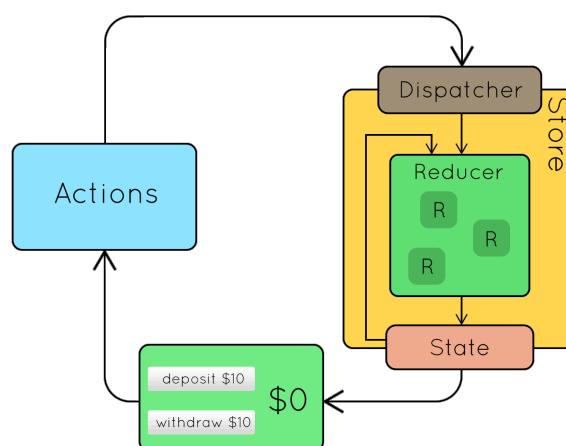
- Baselayout
- Mobilelayout
- MobileDetailLayout

ریداکس چیست ؟

کتابخانه Redux یک کتابخانه جاوا اسکریپت متن باز همراه با یک API ساده و محدود برای مدیریت وضعیت برنامه ها است که معمولا همراه با کتابخانه های React یا Angular برای ایجاد رابط کاربری استفاده می شود.

قوانين سه گانه Redux

- تمامی استیت برنامه در یک آبجکت جاوا اسکریپتی نگه داری می شن که بهش میگیم **store**
- استیت ها (همون store) فقط قابل خواندن است، و برای تغییرش باید از اکشن **action** استفاده کنیم.
- تغییرات توسط که تابع pure داده می شه به اسم **reducer**. در واقع استیت قبلی و اکشن را می گیره و برنامه را به استیت جدیدی می بره.



کاربر با کلیک (هر رویداد event دیگر) یک اکشن را dispatch می‌کند.

تابع reducer استیت قبلی و اکشن رو می‌گیره، و یه استیت جدید بر می‌گردونه.

با روی store subscribe، استیت جدید رو می‌گیریم و رابط کاربری رو باهاش به روز می‌کنیم.

یکی دیگر از فایل‌های مهم هم package.json می‌باشد که اطلاعات کاملی راجب پکیج‌های نصب شده در پروژه و ورژن‌های آن وجود دارد.

تکنولوژی‌های که در این پروژه استفاده شده عبارتند از :

- React •
- Redux •
- Styled-components •
- Twin.macro •
- Tailwind •
- Swiper •
- React-icons •

معرفی فریمورک Tailwind CSS

یکی از تکنولوژی‌های مهم دنیای وب است که برای زیباسازی صفحات وب استفاده می‌شود. شما با استفاده از قواعدی که CSS در اختیارتان قرار می‌دهد می‌توانید ظاهر صفحات HTML را به هر شکلی که بخواهید تغییر دهید.

اما نوشتن قواعد CSS از ابتدای کار ممکن است برای وبسایت‌ها و اپلیکیشن‌های بزرگ کار درستی به نظر نرسد؛ به همین دلیل بهتر است که سراغ فریمورک‌های موجود CSS برویم. این فریمورک‌ها همراه با یکسری قواعد از پیش‌تعریف شده CSS عرضه می‌شوند. بنابراین تنها کاری که باید بکنیم این است که قواعد مربوطه را روی صفحات وب اعمال نماییم. برخی از محبوب‌ترین فریمورک‌های CSS عبارت هستند از Bootstrap، Foundation ... و Materialize، Pure، Bulma.

تیلویند فریمورک جدید CSS است که نسبت به فریمورک‌های گفته شده تفاوت عظیمی دارد. Tailwind CSS فاقد هر گونه قالب پیش‌فرض است و در آن هیچ کامپوننت داخلی برای رابط کاربری وجود ندارد. را می‌تواند یک فریمورک CSS برای ساخت رابطهای کاربری سفارشی دانست. این بدان معناست که اگر دنبال فریمورکی هستید که یکسری ویجت و کامپوننت از پیش طراحی شده را در خود داشته باشد، Tailwind انتخاب مناسبی برای شما نخواهد بود. اما اگر قصد ساخت یک رابط کاربری پیچیده و البته سفارشی را دارید Tailwind CSS می‌تواند به خوبی به شما کمک بکند. از آنجایی که در این ابزار خبری از کامپوننت‌های آماده به شکل کامپوننت‌های بوت‌استرپ و یا فاوندیشن وجود ندارد بنابراین دو وبسایت طراحی شده با استفاده از این فریمورک الزاماً شبیه همدیگر نخواهند بود.

ما در واقع به جای اینکه از css in js استفاده کنیم از css استفاده کردیم یعنی فایل‌های css به صورت جاوا اسکریپت نوشته می‌شود که برای اینکار از ابزار styled-components استفاده کرده‌ام.

از swiper هم برای توسعه اسلایدر‌ها در صفحه اصلی و برندها استفاده کرده‌ام.

در صفحه‌های بعدی تصاویر پروژه رو متوانید ببینید:

 حساب کاربری

جستجو کنید

پروژه فروشگاه اینترنتی


 بدنست به آرامش بیشتری نیازداره...
به ماساژوره خودت هدیه بده

 پشتیبانی ۲۴ ساعته
 همکاران ما پاسخگو شما
هستند

 بازگشت وجه
 کالا هنگام نارضایتی مرجوع می
گردد

 امنیت در پرداخت
 با خیال راحت از ما خرید کنید

 ارسال رایگان
 سفارشات خود را رایگان دریافت
کنید

گوشی موبایل

همه محصولات <



لوازم جانبی گوشی

همه محصولات <



برند های ما



ANKER

TSCO

HUAWEI

MI



مقالات و بلاگ



ویرگی های شخصیتی و ههارت های کارآفرینان را بشناسید



برای میلیونر شدن از کارآفرینان برتر الگوی گیرید



از تاریخچه باندکاری کامپیوچی چه می دانید؟

درباره فروشگاه

این وبسایت برای پروژه دانشگاه آزاد واحد تهران مرکز توسعه علی مهدوی امیری و برای استاد تناوش ظریحی و پیاده ساری شده و هیچگوئم از کالاها و نوشته و اغصی بیستند صرفا برای تعاملی در این پروژه و به صورت ماقبل قرار داده شده اند.


 اطلاعات دانشجو
 نام و نام خانوادگی: علی مهدوی امیری
 شماره دانشجویی: 9683100002



حساب کاربری



جستجو کنید

پروژه فروشگاه اینترنتی

همراه با گارانتی ویژه فروشگاه

- ضمانت پارگشت کالا
- پشتیبانی در ساعت کاری
- امکان تحویل اکسپرس
- ضمانت اصل بودن کالا
- ارسال با پست در شهرستان ها

گوشی 11

125,000 تومان

برند: شیائومی

کد محصول: shop-11

موجود در انبار

ارسال فوری در تهران



افزودن به سبد خرید



اطلاعات دانشجو

نام و نام خانوادگی : علی مهدوی امیری

شماره دانشجویی : 9683100002

درباره فروشگاه

این وبسایت برای پروژه دانشگاه آزاد واحد تهران مرکز توسعه علی مهدوی امیری و برای استاد تناوش طراحی و پیاده سازی شده و همچنین از کالاهای و نوشه واقعی نیستند مراقب برای نمایش در این پروژه و به صورت ماق قرار داده شده اند.



همراه همیشگی ما شوید:

تمام حقوق سایت محفوظ است



حساب کاربری

جستجو کنید

پروژه فروشگاه اینترنتی

مجموع قیمت: 125000
مالیات: 0
قیمت نهایی: 125000

اقدام به پرداخت

سبد خرید

xiaomi 11 گوشی 125,000 توoman

مشخصات خریدار

نام و نام خانوادگی

آدرس خود را وارد کنید



اطلاعات دانشجو

نام و نام خانوادگی: علی مهدوی امیری
شماره دانشجویی: 9683100002

درباره فروشگاه

این وبسایت برای پروژه دانشگاه آزاد واحد تهران مرکز توسعه علی مهدوی امیری و برای استاد تناوش طراحی و پیاده سازی شده و همچنددم از کالاهای و نوشتہ واقعی نیستند صرفا برای نمایش در این پروژه و به صورت مات قرار داده شده اند.



همراه همیشگی ما شوید:

تمام حقوق سایت محفوظ است



حساب کاربری

جستجو کنید

پروژه فروشگاه اینترنتی

بیشترین قیمت
کمترین قیمت
مرتب سازی بر اساس

دسته بندی ها

همه محصولات

گوشی موبایل

لوازم جانبی گوشی



گوشی موبایل
Xiaomi 11 125,000 تومان



گوشی موبایل
Poco 120,000 تومان



گوشی موبایل
Samsung A73 110,000 تومان



گوشی موبایل
Huawei Mate 50 100,000 تومان



گوشی موبایل
Poco X4 90,000 تومان



گوشی آیفون ۱۳ پرو
5,400,000 تومان



گوشی آیفون ۱۳
4,000,000 تومان



گوشی شیائومی Redmi Note 11 1,250,000 تومان



گوشی موبایل
Samsung A23 692,000 تومان



گوشی موبایل
Samsung A03 200,000 تومان



دانشگاه تهران



دانشکده فنی



دانشکده مهندسی کامپیوتر

اطلاعات دانشجو
نام و نام خانوادگی : علی مهدوی امیری
شماره دانشجویی : 9683100002

درباره فروشگاه

این وبسایت برای پروژه دانشگاه آزاد واحد تهران مرکز
توسط علی مهدوی امیری و برای استاد تناوش طراحی و
پیاده سازی شده و هیچگذوه از کالاهای و نوشته واقعی
نیستند صرفا برای تعامل در این پروژه و به صورت مات
قرارداده شده اند.



همراه همیشگی ما شوید:

تمام حقوق سایت محفوظ است



پروژه فروشگاه اینترنتی

نام کاربری

نام کاربری را وارد کنید

رمزعبور



رمزعبور را وارد کنید

ورود

اکانت ندارید؟ [ثبت نام](#)



پروژه فروشگاه اینترنتی

نام کاربری

نام کاربری را وارد کنید

ایمیل

ایمیل خود را وارد کنید

رمزعبور



رمزعبور را وارد کنید

تکرار رمز عبور



رمزعبور را دوباره وارد کنید

ثبت نام

با کلیک بر روی دکمه ثبت نام تمام قوانین سایت را پذیرفته اید.

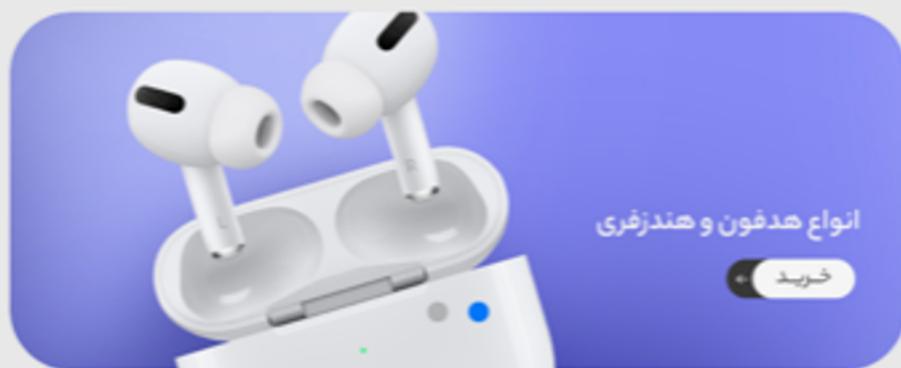
Irancell 21:39 63%

پروژه فروشگاه اینترنتی - استاد تناوش

حساب کاربری

فروشگاه اینترنتی

جستجو کنید... 



امنیت در پرداخت 

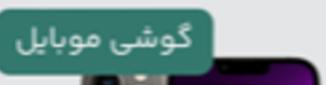
ارسال رایگان 

پشتیبانی 24 ساعته 

بازگشت وجه 

همه محصولات <

گوشی موبایل 



بلاگ



دسته بندی ها



سبد خرید



خانه

No Service VPN

21:40

63%

پروژه فروشگاه اینترنتی - استاد تناوش

حساب کاربری

فروشگاه اینترنتی •

جستجو کنید...



بلاگ



دسته بندی ها

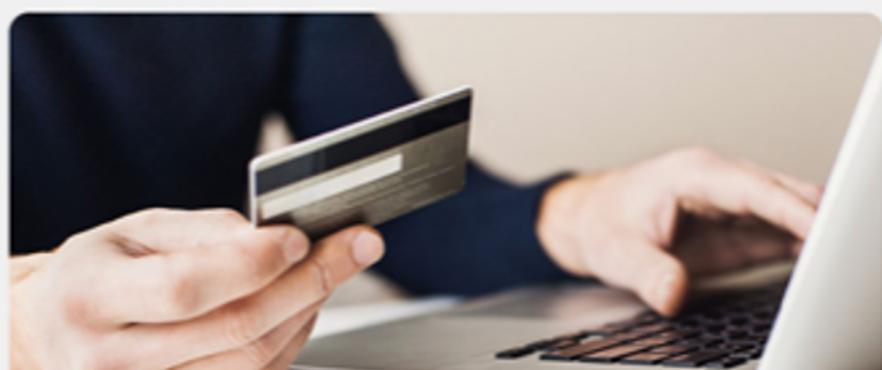


سبد خرید



خانه

بلاگ



۱۰ تیر

از تاریخچه بانکداری الکترونیک چه می‌دانید؟



۱۲ تیر

برای میلیونتر شدن از کارآفرینان برتر الگو بگیرید



بلاگ



دسته‌بندی‌ها



سبد خرید



خانه

No Service  

21:40

 63%

پروژه فروشگاه اینترنتی - استاد تناوش

حساب کاربری

فروشگاه اینترنتی

جستجو کنید... 

دسته بندی ها

همه محصولات گوشی موبایل لوازم جانبی گوشی

مرتب سازی بر اساس

بیشترین قیمت کمترین قیمت

گوشی موبایل

گوشی موبایل



بلاگ



دسته بندی ها



سبد خرید



خانه

No Service VPN

21:40

63%

پروژه فروشگاه اینترنتی - استاد تناوش

حساب کاربری

فروشگاه اینترنتی •

جستجو کنید...



1401 تیر 10

از تاریخچه بانکداری الکترونیک چه می‌دانید؟

بانکداری الکترونیک یا e-banking اصطلاحی
است که به معاملاتی که بین شرکت‌ها،



بلاگ



دسته‌بندی‌ها



سبد خرید



خانه

سبد خرید

گوشی آیفون ۱۳ پرو

-

1

+

5,400,000 تومان



مجموع قیمت: 5,400,000 تومان

مالیات: 0

قیمت نهایی: 5,400,000 تومان

مشخصات خریدار

نام و نام خانوادگی

شهر

استان



بلاگ



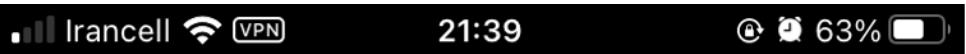
دسته بندی ها



سبد خرید



خانه



دسته بندی ها



لوازم جانبی موبایل



گوشی موبایل



بلاگ



دسته بندی ها



سبد خرید



خانه

برای دیپلو سایت از سرور لینوکس استفاده شد و چون ما از تکنولوژی PWA در اجرای پروژه استفاده کردیم و بسایت باید از SSL استفاده میشد که هم در سایت نصب و پیاده سازی شد. برای مشاهده سورس کد هم می توانید از ریپازیتوری گیت هابم کد ها را برداشته و مشاهده کنید.

<https://github.com/zahrahajiali/ecommerce-project-university>

برای راه اندازی به حالت development ابتدا باید ریپازیتوری که در بالا داده شد را گرفته و سپس آخرین نسخه node.js را داخل سیستمان نصب کنید سپس وارد دایرکتوری پروژه می شوید و با دستور npm install تمام پکیج های استفاده شده را که داخل فایل package.json هست را روی پروژه نصب کنید. سپس با وارد کردن دستور npm start در ترمینال وبسایت به حالت development در مرورگر بالا می آید. ترجیحا برای استفاده بهتر از پروژه بهتر است از مرورگر کروم آخرین نسخه آن استفاده کنید.

باتشکر

زهرا حاجی علی - ۹۸۰۱۱۶۹۸۲