
بسم الله الرحمن الرحيم

دانشکده آمار، ریاضی و علوم کامپیوتر

پروژه کارشناسی

عنوان:

فناوری‌های جدید طراحی وب سایت

استاد راهنما:

دکتر عباس نوذری دالینی

دانشجو:

مصطفی غفوری

اسفند ۹۹

چکیده

امروزه وب برای ارائه انواع برنامه ها، از خدمات در مقیاس کوچک و کوتاه مدت گرفته تا سیستم های گردش کار در مقیاس بزرگ، در بسیاری از سرورها، به یک محیط وسیع تبدیل شده است. برنامه های تحت وب در اینترنت همه جا وجود دارند و تقریباً هر نوع سازمانی اکنون باید برنامه های تحت وب خود را توسعه دهند.

هر ساله فناوری های طراحی وب سایت، دستخوش تغییر و تحولات زیادی می شوند. بسیاری از توسعه دهندگان وب از هیچ یک از تکنیک های پیشرفته و نوین طراحی وب سایت استفاده نمی کنند و یا از آن‌ها آگاهی ندارند. در نتیجه عمدتا برنامه های ایجاد شده دارای عملکرد ضعیف و کاربر ناپسند خواهند بود که نگهداری آنها دشوار نیز است. توسعه دهندگان باید سعی کنند تا برنامه‌ی آن‌ها، صحیح و بدون خطا، قابل نگهداری، قابل استفاده مجدد، قابل اعتماد و ... باشند. برای رسیدن به این اهداف، توسعه دهندگان باید از تکنیک های مهندسی نرم افزار استفاده کنند زیرا وب سایت ها به مرور زمان بیشتر و بیشتر شبیه نرم افزارهای سنتی می شوند.

فهرست مطالب

فصل ۱

تعاریف و مفاهیم اولیه

۱.۱ پیش گفتار

جایگاه و اهمیت اینترنت زمانی به طور واقعی درک شد که از طریق رابط گرافیکی در دسترس قرار گرفت. این معرفی مرورگر وب بود که در اوایل دهه ۱۹۹۰ منجر به محبوبیت گسترده شبکه جهانی وب شد.

امروزه در دنیای دیجیتال و اینترنت، سیستم های کامپیوتری به یکی از ابزارهای اساسی ارتباطات تبدیل شده است. با پیشرفت فناوری اطلاعات، یکی از نیازهای اساسی هر کسب و کاری داشتن یک وب سایت می باشد تا از طریق آن بتوانند به اعتبار تجاری خود بی افزایند و به کسب و کار خود رونق بخشند.

بر اساس آمارهای منتشر شده از موسسه معتبر Internet live stats بیش از ۱.۵ میلیارد وب سایت در سراسر جهان وجود دارند که بیش از ۲۰۰ میلیون از آن‌ها به صورت فعال مشغول کارند. و روزانه به تعداد وب سایت‌های فعال افزوده می شود.

هر ساله فناوری های نوینی در زمینه توسعه وب سایت و برنامه های تحت وب معرفی می شوند. در سال های اخیر رشد سریع کاربران اینترنت به ویژه بر روی دستگاه های موبایل، باعث شد تا استراتژی های طراحی سایت برای دسکتاپ، جای خود را به طرح هایی دهند که برای دستگاه های موبایل مناسب تر هستند. از اینرو برای اینکه کسب و کارها در رقابت تجاری خود عقب نمانند، لازم است تا از جدید ترین ترفند ها و فناوری های توسعه وب آگاه باشند.

۲.۱ تاریخچه وب

ظهور وب را می توان منشاء یکی از مهمترین تحولات قرن بیست و یک در عرصه ارتباطات دانست. ایده اولیه در مورد وب جهان گستر در سال ۱۹۸۰ (میلادی) توسط تیم برنزلی ارائه شد. او پس از حدود ۱۰ سال به کمک یکی از همکاران خود اولین صفحه وب را در آگوست سال ۱۹۹۱ ایجاد و منتشر کرد که این صفحه، تنها شامل چند خط به همراه یک لینک ایمیل بود.

۱.۲.۱ وب چیست؟

وب جهان‌گستر، یا به اختصار وب یک سامانه اطلاعاتی از پرونده‌های ابرمتنی متصل به هم است که از طریق شبکه جهانی اینترنت قابل دسترسی هستند. به کمک یک مرورگر وب می‌توان صفحات وب (که شامل متن، تصویر، ویدئو و سایر محتویات چندرسانه‌ای هستند) را مشاهده کرده و به کمک ابرپیوندها در میان آن‌ها حرکت کرد. به طور کلی به فضای آنلاین متشکل از میلیون ها، بلکه میلیاردها وب سایت اینترنتی وب گفته می شود. در حقیقت، به مجموعه وب سایت های اینترنتی مختلف، که در سراسر دنیا و به زبان های گوناگون هستند، وب گفته می شود. وب بستری از به هم پیوستن شبکه های کامپیوتری مختلف به منظور به اشتراک گذاری محتوا می باشد، اما وب سایت، یکی از واحدهای کوچک تشکیل دهنده این رسانه جهانی است.

۲.۲.۱ انواع وب

- **وب ۱.۰ :** از سال ۱۹۹۰ که وب توسط آقای برنرزلی ارائه شد، جنبشی به وجود آمد که باید همه اطلاعات از روی کاغذ به اطلاعات الکترونیکی تبدیل شود. افراد و شرکت‌ها تلاش کردند تا محتوای کاغذی خود را به محتوای دیجیتالی تبدیل کنند. جنبش سریع دیجیتالی شدن اطلاعات موجب شد که کاربران وب، امکان دسترسی به انبوهی از اطلاعات را داشته باشند. نتیجه این حرکت، میلیون‌ها صفحه حاوی اطلاعات مختلف است که امروز در دسترس همه است.
- **وب ۲.۰ :** وب ۲ را یک رویکرد نو می دانند. رویکرد جدید وب به این صورت است که اطلاعات به واحدهای کوچکتری از محتوا تقسیم می شوند. در حقیقت، وب جدید دنیای "داده" است نه دنیای "اسناد". در وب ۲ دیگر به دنبال منابع قدیمی اطلاعات نیستیم، بلکه به دنبال ابزاری هستیم تا اطلاعات را به شیوه های جدید و موثرتری جمع آوری و تلفیق کند و در اختیار ما قرار دهد. نرم افزارهای مبتنی بر وب، محیط وب را از صفحات ساده به دنیایی چند بعدی تبدیل کرده‌اند که امکان برقراری ارتباطات فردی و کارهای گروهی را فراهم کرده است. شبکه‌های اجتماعی به سرعت به وجود آمده اند و با استقبال گسترده کاربران رو به رو شده اند. وب ۲ به بستری تبدیل شده است که می‌توان انواع نرم افزار را بر پایه آن ساخت تا کاربران فارغ از نیاز به نصب آن بر روی کامپیوتر شخصی شان، بتوانند در هر مکانی به آن‌ها دسترسی داشته باشند.
- **وب ۳.۰ :** نسخه سوم وب تازه در حال متولد شدن است و مرحله ای جدید از آینده وب است. تمرکز اصلی وب ۳ روی هوشمند شدن وب است و پیش بینی می‌شود که در آینده نزدیک کامپیوتر ها، محتوای وب را می‌فهمند و آن را درک می کنند. تکنولوژی های جدید همچون وب معنایی در

دستور کار این نوع از وب قرار خواهد گرفت. نرم افزارهای تحت وب ۳، استفاده از اینترنت پر سرعت را در میان مردم جهان افزایش خواهند داد. در وب ۳، برنامه ها نسبتا کوچک و کم حجم خواهند شد، داده ها به صورت توده ای خواهند بود، برنامه بسیار سریع و قابل انعطاف خواهند بود و بر روی هر سیستمی قابل اجرا خواهند بود.

۳.۱ وب سایت و برنامه های تحت وب

۱.۳.۱ وب سایت چیست؟

وب سایت به مجموعه ای از صفحات وب متصل به هم گفته می شود که معمولا روی یک سرور قرار دارند و به عنوان مجموعه ای از اطلاعات توسط یک فرد، گروه یا سازمان، تهیه و نگهداری می شود. وب سایت در اصطلاح، به مکانی در وب گفته می شود که یک صفحه یا تعداد بیشتری از صفحات را در خود جای داده است. محتویات وب سایت ها معمولا بر روی یک کامپیوتر به نام سرور قرار دارد و مجموعه ای از سرورها، شبکه جهانی وب را تشکیل می دهند. درون هر سرور یک یا تعداد بیشتری سایت جای دارد. مانند دنیای واقعی که هر انسان برای خود یک نام منحصر به فرد دارد، در دنیای مجازی یا وب نیز چنین چیزی وجود دارد و هر سرور (یا کامپیوتر) برای آنکه از سرورهای دیگر متمایز شود و همچنین هر وب سایت برای آنکه از وب سایت های دیگر متمایز گردد، دارای یک شماره شناسایی منحصر به فرد یا IP مشخصی می باشد. وب سایت ها شامل انواع فایل ها بر روی شبکه جهانی اینترنت هستند، که ابزار مناسبی برای ارتباط با سایر کاربران در سریع ترین زمان و به صورت لحظه ای و همچنین اشتراک فایل ها و مطالب، با آن ها می باشد.

۲.۳.۱ برنامه های تحت وب

برنامه های تحت وب یا وب اپلیکیشن ها، ترکیبی از وب سایت و اپلیکیشن هستند. نرم افزاری است که با استفاده از مرورگر وب و از طریق شبکه های محلی یا اینترنت مورد استفاده قرار می گیرد. در اصل، برنامه های تحت وب برنامه هایی هستند که تنها از طریق وب قابل استفاده هستند. بیشتر به عملکرد و کارکرد برنامه های تحت وب توجه می شود و محتوای آن ها چندان اهمیت ندارد و معمولا پر محتوا نیستند. وب اپلیکیشن پس از تجزیه و تحلیل و بررسی یک مجموعه و مشخص نمودن دقیق نقش ها، عملکردها، و... برای آن مجموعه طراحی می شود و قابل ارائه در شبکه داخلی و اینترنت می باشد.

۳.۳.۱ تفاوت وب سایت و برنامه‌ی تحت وب

وب سایت ها بر محتوا تاکید می کنند اما برنامه های تحت وب بر عملکرد و کارکرد تمرکز دارند. وب سایت، یک مطلب را برای همه به طور یکسان نشان می دهد اما یک وب اپلیکیشن بر اساس تعاملاتی که با کاربر داشته است، اطلاعات را پردازش می کند و به کاربر نمایش می دهد. یعنی مطالب نمایش داده شده برای هر کاربر بر اساس سن، علاقه، جنسیت، و... متفاوت با دیگر کاربران است. یک کاربر برای اینکه بتواند اطلاعات یک وب سایت را مشاهده یا از آن استفاده کند، بایستی حتما به اینترنت متصل

شود اما در وب اپلیکیشن این چنین نیست. تنها یک بار کافی است تا وب اپلیکیشن بارگذاری شود، سپس کاربر می تواند به صورت آفلاین هم به آن دسترسی داشته باشد.

۴.۳.۱ ویژگی های برنامه‌ی تحت وب

- **حفظ یکپارچگی بر روی دستگاه های مختلف :** ویژه ترین مزیت برنامه‌ی تحت وب را به جرأت می توان حذف محدودیت استفاده از برنامه اختصاصی سیستم عامل نام برد. با ظهور برنامه های تحت وب، کاربران برنامه های اندروید و IOS دیگر محدود به برنامه های موجود در گوگل استور و اپ استور نیستند. محدودیت اینکه یک کاربر اندروید صرفا می تواند از برنامه های مخصوص اندروید استفاده کند یا یک کاربر آیفون فقط باید برای برنامه های مورد نیاز خود به اپ استور سر بزند با وجود برنامه‌ی تحت وب دیگر وجود ندارد. با هر سیستم عاملی می توان فقط با وارد کردن آدرس وب اپلیکیشن در نوار جستجوی مرورگر به برنامه مورد نظر خود دست یافت.
- **حجم کم و سرعت استفاده بالا:** برنامه‌ی های تحت وب فضایی از حافظه‌ی دستگاه کاربران را درگیر خود نمی کنند. برنامه‌ی تحت وب یک صفحه در اینترنت است و فضایی از حافظه‌ی دستگاه را اشغال نمی کند و این یکی دیگر از مزیت های ویژه آن است.

از دیگر مزیت های برنامه‌های تحت وب میتوان به قابلیت طراحی به زبان های مختلف برنامه نویسی، سرعت، دقت و عملکرد خوب نیز اشاره نمود.

۴.۱ طراحی وب سایت و انواع آن ها

۱.۴.۱ طراحی وب سایت چیست؟

به روند برنامه ریزی و ساخت یک وب سایت، طراحی وب سایت می گویند. متن، تصاویر، فایل های صوتی و تصویری و المان های برنامه نویسی، توسط طراحان سایت برای تولید صفحه ای قابل رویت در مرورگر، فرمت دهی می شوند. طراحان سایت برای ایجاد صفحاتی که توانایی خوانده شدن توسط مرورگرها را داشته باشند، از زبان های نشانه‌ای HTML و CSS استفاده می کنند. در دنیای وب ۲، زبان های برنامه نویسی دیگر مانند جاوا اسکریپت، ASP.net PHP، Rails، on Ruby و Perl برای ساخت صفحات سایت ها استفاده می شود . در مجموع می توان گفت؛ پروسه طراحی وب سایت شامل مفهوم سازی، طرح ریزی، پیش تولید، تحقیق، تبلیغات، و همچنین مدیریت فایل های صوتی، تصویری و دیگر فایل های چند رسانه ای مورد استفاده در صفحات سایت می باشد. وب سایت می تواند شامل صفحه اصلی، که به آن خانه نیز می گویند و همچنین بخش های متفاوت دیگری باشد که هرکدام برای هدف خاصی ایجاد شده اند. صفحه اصلی وب سایت در مورد هدف اصلی کسب وکار صحبت می کند که می تواند شامل لینک هایی باشد که کاربران را به بخش های مختلف سایت هدایت می کند. بعضی از سایت‌ها با استفاده از سیستم‌های آماده و رایگان مدیریت محتوا طراحی می‌شوند. سیستم‌هایی مانند وردپرس یا جوملا که قالب‌های آماده را در اختیار طراحان قرار می‌دهند.

۲.۴.۱ انواع وب سایت ها بر اساس کاربرد و روش طراحی

- **وب سایت های استاتیک :** وب سایت های استاتیک، وب سایت هایی می باشند که محتوای آن‌ها برای تمامی کاربران یکسان و بدون تغییر می باشد. معمولا اینگونه از وب سایت ها بصورت گسترده با زبان نشانه‌ای HTML توسعه داده می شوند.
- **وب سایت های داینامیک :** در مقایسه با وب سایت های استاتیک، وب سایت های داینامیک کاربردی تر هستند. محتوای اینگونه از وب سایت ها می تواند با توجه به نوع کابران سایت متفاوت باشد. علاوه بر این، وب سایت های داینامیک به زبان های برنامه نویسی سمت سرور نیز متکی هستند.

۳.۴.۱ انواع سایت از نظر نسل

میان نسل‌ها در سایت‌های اینترنتی شکاف هم وجود دارد. سایت‌های اینترنتی از نظر نسل به سه دسته تقسیم می‌شوند:

- **سایت‌های نسل اول :** سایت‌های نسل اول، یک‌طرفه هستند. یعنی محتوا توسط مدیر یا مدیران سایت بارگذاری می‌شود و کاربران فقط می‌توانند آن را ببینند.
- **سایت‌های نسل دوم :** در این سایت‌ها کاربران و بازدیدکنندگان نیز در فرایند تولید محتوا سهیم هستند. می‌توانند مطالب را لایک کنند، کامنت بگذارند و ... در این سایت‌ها نیازی به بارگذاری مجدد صفحات نیست و تنها بخش‌هایی که تغییر کرده‌اند دوباره بارگذاری می‌شوند. به این ترتیب سرعت بارگذاری سایت بالاتر می‌رود.
- **سایت‌های نسل سوم :** سایت‌های نسل سوم، که با نام وب معنایی نیز شناخته می‌شوند، سایت‌های هوشمند هستند. این سایت‌ها مخاطب خود را می‌شناسند و با استفاده از تکنیک‌های داده‌کاوی و هوش مصنوعی، محتوایی متناسب با سلیقه و نیاز مخاطب نشان می‌دهند. این سایت‌ها بسیار پیشرفته هستند و پیچیدگی‌های زیادی دارند و به همین دلیل تعداد کمی از آن‌ها موجود است.

فصل ۲

فناوری های موجود در توسعه وب

۱.۲ مقدمه

فناوری های وب ابزارها و تکنیک های مختلفی هستند که در فرایند ارتباط بین انواع مختلف دستگاه ها از طریق اینترنت استفاده می شوند. در ادامه به تعریف فناوری های اصلی توسعه وب خواهیم پرداخت.

۲.۲ اجزاء و معماری وب

وب ترکیبی از چهار عنصر اصلی است:

- **hypertext یا ابرمتن :** فرمتی از اطلاعات که به افراد اجازه می‌دهد تا در محیط کامپیوتر با استفاده از ارتباط داخلی موجود میان دو متن از بخشی از سند به بخش دیگری از آن یا حتی سند دیگری مراجعه کنند و به اطلاعات جدیدی دسترسی پیدا کند.
- **URL :** شناسه‌های منحصر به فردی که برای مشخص کردن محل حضور اطلاعات موجود روی شبکه (فایل کامپیوتری، سند یا منابع دیگر) به کار می‌روند.
- **زبان علامتگذاری :** کاراکترها یا کدهای موجود در متن که ساختار متن وب معنایی را مشخص می‌کنند. (HTML)
- **پروتکل انتقال ابرمتن :** پروتکل انتقال ابرمتن یک پروتکل درخواست و پاسخ می‌باشد. برای مثال یک مرورگر وب می‌تواند یک کلاینت و نرم‌افزار موجود بر روی سرویس‌دهنده وب سایت، یک سرور باشد. شروع این پروتکل از طرف کلاینت است که با ارسال یک درخواست HTTP به سمت سرور گفتگو را آغاز می‌کند. سرور بر اساس درخواست ارسالی یا منبعی مانند یک فایل را در اختیار کلاینت می‌گذارد یا عملیات خاصی را انجام می‌دهد. نتیجهٔ این عملِ سرور در بستهٔ پاسخ HTTP برای کلاینت ارسال می‌شود. بسته پاسخ شامل اطلاعات وضعیت و احتمالاً محتویات منبع درخواست شده می‌باشد.

۱.۲.۲ مدل سرویس‌گیرنده_سرویس‌دهنده

در این مدل یک منبع در خواست انجام کارش را به سرویس دهنده ارائه می دهد و سرویس دهنده پس از اجرای وظیفه محوله ، نتایج حاصل را به منبع در خواست کننده عودت می دهد. در این مدل حجم اطلاعات مبادله شده شبکه ، کم است و این مدل دارای کارایی بالایی نیز می باشد.

۲.۲.۲ تعریف مفاهیم back-end, front-end

- **فرانت_اند :** زمانی که در مورد فرانت_اند وب صحبت می کنیم، منظور آن بخشی است که کاربر می تواند با آن تعامل برقرار کند و قابل دیدن است. فرانت_اند کدهای غیر قابل فهم برای کاربران را در قالب ظاهری گرافیکی و بصری به آن‌ها نمایش می‌دهد تا استفاده از بخش‌های مختلف سایت برایشان ساده‌تر شود. در این بخش فرم‌های ورودی اطلاعات، صداها، تصاویر، ویدیوها و به صورت کلی هر چیز دیگری که برای کاربر قابل درک باشد، قرار می‌گیرد. فرانت_اند معمولا از دو بخش تشکیل می شود: طراحی سایت و توسعه فرانت_اند وب.
در بخش طراحی وب، طراحان با نرم افزارهای گرافیکی مانند فتوشاپ ظاهر سایت را طراحی می‌کنند. اما بخش توسعه رابط کاربری مربوط به پیاده‌سازی ظاهر سایت در قالب کدهای HTML، CSS، و JavaScript است.
- **بک_اند :** بک_اند دقیقا در مقابل مفهوم فرانت_اند قرار دارد. بک_اند، به بخشی از یک وب سایت یا نرم افزار می‌گویند که برای کاربران قابل مشاهده نیست. به عبارتی دیگر هسته و مغز یک سایت است که وظیفه کنترل منطق آن را بر عهده دارد. به عنوان تکنولوژی های مربوط به این بخش، می توان به زبان های برنامه نویسی متعددی مانند: پایتون، جاوا، جاوا اسکریپت و ... اشاره کرد.

۳.۲ فناوری های وب

۱.۳.۲ HTML

زبان نشانه‌ای (HyperText Markup Language) که معمولاً به آن اچ تی ام ال می گویند ، جایی است که وب شروع به کار کرده است. HTML یک زبان فرانت_اند می باشد که برای تعریف و توصیف معنایی محتوای یک صفحه وب استفاده می شود. به توسعه دهندگان وب کمک می کند تا چگونگی ساختار صفحه وب را تعیین کنند. بخش های مختلف توسط اجزایی به نام تگ از هم جدا شده ، که هر کدام دارای کاربرد و خواص مربوط خود هستند . این تگ ها به مرورگر اعلام می کنند که هر بخش از صفحه چه نوع عنصری است و باید به چه صورت نمایش داده شود . در یک صفحه HTML می توان انواع عناصر از قبیل متن ، تیتر ، عکس ، جدول و ... را قرار داد ، که برای هر عنصر باید از تگ مربوط به آن استفاده کرد . صفحات HTML فقط از کد ها که به صورت متن هستند تشکیل شده اند. بدین معنا که برای تصویر کد مربوط به تمایش تصویر و جدول و ... کدهای اچ تی ام ال مربوط به هر یک را باید نوشت و مرورگر با رسیدن به این کد ها و تگ ها ، المنت های مرتبط با آن را نمایش می

دهد. هر یک از کدهای html ، معنا و مفهوم خاصی دارند و تأثیر مشخصی بر محتوا می‌گذارند. مثلاً برچسب‌هایی برای تغییر شکل ظاهری متن، نظیر درشت و ضخیم کردن یک کلمه یا برقراری پیوند به صفحات دیگر در اچ‌تی‌ام‌ال تعریف شده‌اند.

```
<!DOCTYPE html>
<html>
  <body>
    <h1>Hello, World!</h1>
  </body>
</html>
```

در قطعه کد بالا، تمامی عناصر توسط علامت های <،> احاطه شده اند. عبارت docktype این مفهوم را به مرورگر می رساند که محتوای فایل در قالب HTML نوشته شده است. تمامی عناصر قابل رویت در صفحه می بایست مابین عنصر body قرار گیرند.

اچ تی‌ام ال زبان برنامه‌نویسی نیست، بلکه زبانی برای نشانه‌گذاری ابرمتن است و اساساً برای ساخت‌مند کردن اطلاعات و چینش اجزای منطقی یک نوشتار، نظیر عناوین ، تصاویر ، فهرست‌ها ، بندها و جداول به کار می‌رود. از سوی دیگر، اچ تی ام ال را نباید به عنوان زبانی برای صفحه‌آرایی صفحات وب به کار برد؛ این وظیفه اکنون بر دوش فناوری‌های دیگری همچون سی اس اس است که در ادامه به آن خواهیم پرداخت.

۲.۳.۲ CSS

سی اس اس (Cascading Style Sheets) به عبارت ساده برای آراستن صفحات وب در طراحی سایت می باشد. با استفاده از CSS میتوان تگ های html را به راحتی به استایل های مختلف در آورد. رنگ بندی و تعیین پس زمینه ، تعیین مکان و عرض و طول و ... همگی کارهایی است که با CSS می توان انجام داد.

۳.۳.۲ طراحی سایت واکنش گرا

طراحی واکنش گرا و یا responsive ، روشی است که صفحات وب نسبت به دستگاه کاربر چون گوشی تلفن هوشمند، لب تاپ ، انواع مانیتور کامپیوترها واکنش نشان داده و خود را بر مبنای اندازه صفحه و اسکرین نمایش می دهند. امروزه کاربران از بیشمار نمایشگر در ابعاد بزرگ و کوچک برای دستیابی به صفحات اینترنت استفاده میکنند. از آنجا که امروزه فناوری به یک بخش ضروری از زندگی روزمره تبدیل شده است در نتیجه می بایست طیف گسترده ای از دستگاه هایی که کاربران به واسطه آنها قادر به دسترسی به اینترنت هستند را در نظر گرفت. زیرا انعطاف پذیری، کلید موفقیت در کسب و کار، به خصوص کسب و کار آنلاین است.

Javascript ۴.۳.۲

یکی از زبان های برنامه نویسی اسکریپتی است ، که اولین بار توسط شرکت Netscape در سال ۱۹۹۵ ارائه شد و امروزه متداولترین زبان اسکریپت نویسی صفحات وب است.

جاوا اسکریپت یک زبان شی گرا، ترجمه‌ای، سبک و تابعی است. این زبان هم سمت کاربر و هم سمت سرور، برای ایجاد تعامل با صفحات وب به کار می‌رود. جایی که HTML و CSS ساختار اولیه و ظاهر صفحات وب را تعیین می‌کنند، جاوا اسکریپت نحوه عملکرد صفحات وب را کنترل می‌کند. جاوا اسکریپت عمدتا برای برنامه‌های مبتنی بر وب و مرورگرهای وب مورد استفاده قرار می‌گیرد. اما جاوا اسکریپت نیز فراتر از وب در نرم افزار، سرورها و کنترل‌های سخت افزاری استفاده می‌شود. در ادامه برخی از مواردی که جاوا اسکریپت در آنها پرکاربرد هستند را نام می‌بریم:

- **اضافه کردن رفتار تعاملی به صفحات وب :** جاوا اسکریپت به کاربران اجازه می‌دهد تا با صفحات وب ارتباط برقرار کنند. برخی از این قابلیت‌ها: پخش فایل صوتی و تصویری در یک صفحه وب، نمایش یا پنهان کردن اطلاعات، نمایش انیمیشن‌ها و ... می باشند.
- **ایجاد برنامه‌های وب و موبایل :** فریم‌ورک‌های جاوا اسکریپت مجموعه‌ای از کتابخانه‌های جاوا اسکریپت هستند که قابلیت نوشتن کد از قبل را به توسعه دهندگان برای استفاده از ویژگی‌های برنامه‌نویسی روزمره ارائه می‌دهند: به معنای واقعی کلمه یک فریم‌ورک برای ساخت وب سایت‌ها یا برنامه‌های کاربردی وب. فریم‌ورک‌های محبوب جاوا اسکریپت ،Angular React و Vue می‌باشند. بسیاری از شرکت‌ها از NodeJs که یک محیط اجرا جاوا اسکریپت برروی موتور V۸ گوگل کروم می‌باشد استفاده می‌کنند. چند نمونه از مثال‌های مشهور آن: Netflix و Uber می‌باشد.
- **ساخت وب سرورها و توسعه برنامه‌های کاربردی سرور :** فراتر از وبسایت‌ها و برنامه‌های کاربردی، توسعه دهندگان همچنین می‌توانند از جاوا اسکریپت برای ساخت وب سرورهای ساده و توسعه زیرساخت‌های بک‌ـاند توسط NodeJs استفاده کنند.

زبان های اسکریپتی ، جزء زبان های برنامه نویسی سبک هستند. این زبان ها در هنگام اجرا فازی به نام کامپایل را طی نکرده و دستورات آن ها به صورت خط به خط اجرا می شوند.

یکی از مهم ترین ویژگی های زبان جاوا اسکریپت، قابلیت ناهمگام بودن (asynchronous) آن است. از آنجا که جاوا اسکریپت یک زبان تک نخی است، در هر زمان تنها یک کار می‌تواند اجرا شود که روی نخ اصلی اجرا می‌شود و هر چیز دیگری تا زمان تکمیل شدن آن عملیات متوقف خواهد بود (همگام). برای جبران تک نخ بودن و قابلیت اجرای یک دستور در هر زمان، جاوا اسکریپت قابلیت ناهمگام بودن برای اجرای برخی از دستورات زمان گیر را به خود اضافه کرد.

دو نوع عمده از سبک کد ناهمگام وجود دارد که در جاوا اسکریپت با آن مواجه می‌شویم. یکی call-back ها و دیگری کد به سبک promise می باشد. callback های ناهمگام توابعی هستند که در زمان فراخوانی یک تابع به صورت پارامتر استفاده می‌شوند و شروع به اجرا در پس زمینه می‌کنند. زمانی که کد پس زمینه اجرای خود را تمام کند، تابع callback را فراخوانی می‌کند تا بداند که کار انجام یافته است و یا اطلاع دهد که اتفاق خاصی رخ داده است. سبک جدیدی از کد ناهمگام، promise ها هستند. promise شیئی است که تکمیل یا شکست عملیات ناهمگام را نمایش می‌دهد. این پارامتر یک

حالت واسط را نمایش می‌دهد. عملیات ناهمگام مانند promise در یک «صف رویداد» قرار می‌گیرد که پس از پایان پردازش نخ اصلی اجرا می‌شود، به طوری که کد جاوا اسکریپت بعدی را مسدود نمی‌کند. عملیات صف‌بندی شده به محض این که امکان‌پذیر باشد، اجرا می‌شوند و نتایج آن‌ها در محیط جاوا اسکریپت بازگشت می‌یابد.

JSON ۵.۳.۲

جی سون (JavaScript Object Notation) یک فرمت فایل برای ذخیره سازی و یا انتقال فایل ها می باشد. داده ها در آن بصورت کلید و مقدار (key, value) ذخیره می شوند. یکی از مهم ترین کاربرد های جی سون انتقال داده می باشد و بصورت عمده برای ارتباط بین سرور و مرورگر ها استفاده می شود.

API ۶.۳.۲

رابط برنامه‌نویسی برنامه کاربردی (Application Programming Interface) قراردادهای استانداردی هستند که تعیین می‌کنند توسعه‌دهندگان چگونه با یک سرویس و انواع خروجی‌های آن تعامل داشته باشند. می‌توان گفت که API فصل مشترکی مابین دو نرم‌افزار و یا برنامه است.

دسته‌بندی‌های مختلفی برای ای‌پی‌آی‌ها در نظر گرفته می شود، که برخی از آن‌ها عبارتند از:

- **ای‌پی‌آی زبان‌های برنامه‌نویسی :** زبانی همچون جاوا یک هستهٔ اصلی دارد که شامل سینتکس این زبان، نحوهٔ ساخت متغیر، دیتا تایپ‌ها و ... می‌شود اما در کنار آن‌ها صدها کلاس مختلف توسط توسعه‌دهندگان این زبان عرضه شده که تحت عنوان API شناخته می‌شوند که ویژگی های تکمیلی این زبان را در دسترس توسعه دهندگان قرار می‌دهند.
- **ای‌پی‌آی تحت وب :** این نوع ای‌پی‌آی یکی از متداول‌ترین و کاربردی‌ترین انواع ای‌پی‌آی است. ای‌پی‌آی تحت وب یا وب سرویس به هر پروتکلی گفته می‌شود که از طریق شبکهٔ اینترنت و وب تعامل مابین اپلیکیشن‌های مختلف را امکان‌پذیر سازد.

وب سرویس ها ۷.۳.۲

همان طور که ذکر شد، ب سرویس به هر پروتکلی گفته می‌شود که از طریق شبکهٔ اینترنت و وب تعامل مابین برنامه های مختلف را امکان‌پذیر سازد. وب سرویس ها را می توان بطور کلی به دسته های زیر تقسیم بندی کرد:

- **PRC :** این نوع وب سرویس که نام آن مخفف واژگان Programmable Remote Client در دو نوع مدل XML-RPC و JSON-RPC عرضه شده است و همان‌طور که از نام آن‌ها مشخص است، مدل اول از فرمت اکس‌ام‌ال پشتیبانی می‌کند و مدل دوم از جی سون. این وب سرویس امروزه کاربرد چندانی ندارد.

- **SOAP :** پروتکل دسترسی آسان به اشیاء و مخفف عبارت Simple Object Access Protocol می باشد. این پروتکل برای رد و بدل کردن اطلاعات بین برنامه ها استفاده می شود. اطلاعات در SOAP به صورت پیام و از طریق پروتکل های موجود در اینترنت مانند HTTP منتقل میشود. به زبان ساده تر، SOAP یک پروتکل برای دستیابی به یک سرویس ارایه شده در وب می باشد. پروتکل SOAP یکی از عمومی ترین استاندارد هایی است که در وب سرویس ها استفاده می شود. وقتی یک برنامه شروع به ارتباط با وب سرویس می کند، پیغام های SOAP وسیله ای برای ارتباط و انتقال دیتا بین آن دو هستند. یک پیغام SOAP به وب سرویس فرستاده می شود و یک تابع را در آن به اجرا در می آورد، وب سرویس نیز از محتوای پیغام SOAP استفاده کرده و عملیات خود را آغاز می کند و در انتها نیز نتایج را با یک پیغام SOAP دیگر به برنامه اصلی می فرستد.

- **REST :** این اصطلاح که مخفف واژگان Representational State Transfer است بر خلاف موارد قبل یک پروتکل حساب نمی‌شود بلکه نوعی معماری است. از آنجا که REST تقریباً بر روی همه پروتکل‌ها استفاده می‌شود، هنگام استفاده بر روی API های وب، می‌تواند از مزایای HTTP بهره‌مند شود. یک وب سرویس بصورت REST طراحی شده است هرگاه شرایط زیر را داشته باشد:

مدل آن بصورت سرویس‌گیرنده‌ـ سرویس‌دهنده باشد. بدین معناست که سرویس‌گیرنده و سرویس‌دهنده بایستی از یکدیگر جدا باشند و اجازه توسعه جداگانه و مستقل را داشته باشند. به‌عبارت‌دیگر، باید بتوانیم روی اپلیکیشن موبایل خود، بدون آنکه اثری روی ساختار داده یا طراحی پایگاه داده در سرویس‌دهنده داشته باشد، تغییراتی را اعمال کنیم. همچنین بدون تأثیر بر سرویس‌گیرنده موبایل، بتوانیم پایگاه داده را تغییر دهیم یا تغییراتی در اپلیکیشن سرویس‌دهنده اعمال کنیم. بدین ترتیب ارتباطات از هم جدا می‌شوند و به هر برنامه کاربردی اجازه رشد و مقیاس‌پذیری مستقل داده می‌شود و سازمان شما سریع‌تر و کارآمدتر رشد می‌کند.

می بایست مستقل از حالت باشند. بدین معنی که فراخوانی‌ها مستقل از یکدیگر انجام می‌شود و هر فراخوانی شامل داده‌هایی است که برای اتمام موفق خودش به آن‌ها نیاز دارد. یک REST API نباید وابسته به داده ذخیره‌شده در سرویس‌دهنده یا نشست‌ها باشد تا تعیین کند که با یک فراخوانی چه شود بلکه باید وابسته به داده‌ای باشد که فراخوانی، خود آن را فراهم می‌کند. زمانی که فراخوانی‌ها انجام می‌شود، شناسایی اطلاعات در سرویس‌دهنده ذخیره نمی‌شود. در عوض هر فراخوانی داده خود را دارد. همچنین این مسئله با داشتن همه اطلاعات لازم برای فراخوانی به افزایش قابلیت اطمینان API ها کمک می‌کند تا اینکه وابسته به یک سری فراخوانی با سرویس‌دهنده باشد تا آبجکتی را بسازد که ممکن است منجر به شکست جزئی شود. در عوض برای کاهش نیازمندی‌های حافظه و هرچه مقیاس‌پذیرتر بودن برنامه کاربردی، یک API RESTful نیاز به ذخیره وضعیت در سرویس‌گیرنده دارد — نه در سرویس‌دهنده.

ازآنجاکه یک API مستقل از حالت می‌تواند سربار درخواست را با بار زیاد مدیریت فراخوانی‌های ورودی و خروجی افزایش دهد، یک REST API ساخته می‌شود تا منجر به ذخیره‌سازی داده قابل کش شود. بدین معنی که زمانی که داده قابل کش است، پاسخ باید نشان دهد که داده می‌تواند تا زمان مشخصی (زمان انقضا) ذخیره گردد یا در مواردی که به داده بلادرنگ نیاز داریم، دیگر نباید پاسخ در سرویس‌گیرنده کش شود. با فعال نمودن این شرط بحرانی، نه‌تنها تعداد تعاملات

با API از طریق کاهش استفاده از سرویس‌دهنده داخلی به‌شدت کاهش می‌یابد بلکه ابزار لازم برای تهیه کارآمدترین و سریع‌ترین برنامه‌های کاربردی ممکن را در اختیار کاربران API شما قرار می‌دهد. به خاطر داشته باشید که کش کردن سمت سرویس‌گیرنده انجام می‌شود. ازآنجاکه شما ممکن است بتوانید برخی داده‌ها را در معماری خود کش کنید تا بیشترین کارایی را داشته باشید، تمایل به سمت آموزش سرویس‌گیرنده است که گونه کار را ادامه دهد و داده را موقتاً ذخیره کند یا خیر.

کلید تفکیک سرویس‌گیرنده از سرویس‌دهنده داشتن یک رابط کاربری یکنواخت است که بدون وابستگی سرویس‌ها، مدل‌ها و عملیات برنامه کاربردی به لایه ،API به برنامه کاربردی اجازه تکامل مستقل می‌دهد. رابط کاربری یکنواخت به سرویس‌گیرنده اجازه می‌دهد تا مستقل از معماری بک‌اند با یک‌زبان واحد با سرویس‌دهنده صحبت کند. این رابط کاربری ابزار ارتباطی استاندارد و ثابتی بین سرویس‌دهنده و سرویس‌گیرنده فراهم می‌کند.

شرط پیروی از مدل MVC که در ادامه بررسی خواهد شد نیز وجود دارد.

۸.۳.۲ مدل MVC

سیستم MVC (Model View Controller) سیستمی است که از چندلایه تشکیل‌شده است و هر لایه عملکرد و مسئولیت خاص خود را دارد. هر لایه مسئولیت‌های مخصوص به خود را دارد. Models شامل چگونگی تشکیل داده است، Controller بر عملیات ورودی تمرکز دارد و View روی خروجی تمرکز دارد. هر لایه مستقل است اما با دیگر لایه‌ها در تعامل است. در طراحی API REST همین قوانین حاکم است با لایه‌های مختلفی از معماری که برای ساخت یک سلسه‌مراتب با هم کار می‌کنند و به ساخت یک برنامه کاربردی مقیاس‌پذیرتر و ماژولارتر کمک می‌کنند.

سیستم لایه‌ای به شما اجازه می‌دهد سیستم‌های بازمانده را کپسوله کنید و عملکردهای دسترسی کمتری را به یک واسطه مشترک منتقل می‌کند درحالی‌که از مؤلفه‌های پراستفاده و مدرن‌تر در برابر آن‌ها محافظت می‌کند. به‌علاوه، سیستم‌های لایه‌ای به شما آزادی عمل می‌دهند تا سیستم‌ها را به درون یا بیرون معماری‌تان به‌عنوان تکنولوژی و سرویس‌های تکامل‌یافته منتقل کنید و تا زمانی که ماژول‌های مختلف را از آن‌ها مجزا نگهدارید، انعطاف‌پذیری و طول عمر آن‌ها را افزایش می‌دهید. مزایای امنیتی دیگری نیز برای سیستم‌های لایه‌ای وجود دارد؛ زیرا به شما اجازه می‌دهد تا حمله‌های لایه پروکسی را متوقف کنید یا در بین لایه‌ها از نفوذ آن‌ها به معماری سرویس‌دهنده اصلی جلوگیری کنید.

با بهره‌برداری از سیستم لایه‌ای با استفاده از پروکسی یا ساخت یک نقطه دسترسی، می‌توانید جنبه‌های بحرانی و آسیب‌پذیر معماری‌تان را پشت یک فایروال محافظت کنید و از تعامل مستقیم سرویس‌گیرنده‌ها جلوگیری کنید.

۹.۳.۲ پایگاه های داده

پایگاه داده مجموعه ای از داده های سازمان یافته است که بر اساس قوائدی مشخص در کنار یکدیگر نگهداری می‌شوند. در سیستم توسعه وب، معمولا بطور عمده از دو نوع پایگاه داده SQL و NOSQL استفاده می شود. در ادامه به بررسی جزییات این دو پایگاه داده و خصوصیات آن‌ها خواهیم پرداخت.

- **SQL :** این نوع از پایگاه داده ها به پایگاه داده های رابطه‌ای مشهور هستند. SQL مخفف Structured Query Language است و زبانی مخصوص جستجو و دستکاری و تغییر در کوئری‌ها یا همان داده‌ها در یک پایگاه داده رابطه‌ای می باشد. برخی از معروف ترین پایگاه داده های SQL، Oracle, MySQL, PostgreSQL می باشند. مدل رابطه‌ای پایگاه‌های داده در دهه ۱۹۷۰ میلادی معرفی شد و یک روش بسیار منطبق بر ریاضیات برای سازمان‌دهی، نگهداری و استفاده از داده‌ها ارائه کرد. این مدل، طراحی‌های قبلی عموماً ساده، یا مدل شبکه‌ای را با معرفی مفهوم رابطه در پایگاه داده متحول ساخت. از مهم ترین خصوصیات این نوع پایگاه داده‌ها می توان به موارد زیر اشاره کرد:

  داده ها بر اساس چیزی به نام رابطه، بین جدول ها پخش می شوند.

  داده ها در این نوع از پایگاه های داده در جدول ها ذخیره می شوند. این جداول ساختار کاملا مشخصی دارند. این ساختار محکم بر اساس نام فیلدها و نوع داده هایی است که باید وارد جدول شوند.

- **NOSQL :** این نوع از پایگاه داده ها، به پایگاه داده های غیررابطه‌ای مشهور هستند. مدل غیررابطه‌ای برای سازمان‌دهی داده‌ها به اصل رها شدن از قیدهایی که پیرامون مدل رابطه‌ای وجود دارد اشاره دارد و سعی می‌کند به توسعه‌دهندگان اجازه دهد از ابزارهای نگه‌داری، ساخت و به‌کارگیری محاوره‌ها به شکل آزادانه‌تری استفاده کنند. بانک‌های اطلاعاتی غیررابطه‌ای بر مبنای رویکرد غیرساخت‌یافتگی سعی می‌کنند محدودیت‌های موجود در روابط صریح را حذف کرده و انواع مختلفی از روش‌ها را برای نگه‌داری و کار با داده‌ها به روشی خاص‌منظوره پیاده‌سازی کنند. روش‌هایی که قادر هستند با عملکرد بالا به ذخیره‌سازی اسناد تمام‌متنی بپردازند.

  از مهم ترین پایگاه داده های غیر رابطه ایی می توان به MongoDB اشاره کرد.

  انتخاب پایگاه داده‌ی غیر رابطه ایی در موارد زیر می تواند مفید باشد:

  هیچ تعریف مشخصی از رابطه‌ی بین داده ها وجود ندارد.

  ذخیره‌سازی داده‌هایی با حجم بسیار و ساختاردهی نشده و مقیاس پذیری عمودی و افقی. مقیاس پذیری عمودی یعنی بالا بردن قدرت سرورها مانند افزودن تعداد CPU . مقیاس پذیری افقی یعنی سرورهای جدیدی اضافه شوند و پایگاه داده ی فعلی بین آن ها تقسیم شود.

  دریافت سریع تر داده ها و نرخ پایین ویرایش و نوشتن داده.

  ذخیره‌سازی و انجام محاسبات ابری.

۴.۲ روش های بارگذاری صفحات وب

بحث در رابطه با بارگذاری صفحات وب، در چند سال گذشته مطرح شده است. پیشتر، وبسایت‌ها و برنامه‌های وب یک محتوا HTML دارند که از سمت سرور به مرورگرها ارسال می‌شود. این محتوا به عنوان یک سند HTML به همراه استایل‌های CSS در مرورگر بارگذاری می‌شوند.

با ظهور و ایجاد فریم‌ورک‌های جاوا اسکریپت، راه و روش جدیدی برای توسعه برنامه‌های وب ایجاد شد. فریم‌ورک‌های جاوااسکریپت این امکان را فراهم کردند که سربار و پردازش سمت سرور کمتر شود.

با استفاده از فریم‌ورک‌های جاوا اسکریپت، بارگذاری محتوا توسط ایجاد یک درخواست از طرف مرورگر، برای محتوای خواسته شده، به صورت پویا، انجام می‌شود. در این سناریو، سرور تنها شامل یک فایل پایه‌ای HTML ضروری است. این تغییر باعث می‌شود که تجربه بهتری برای کاربران سایت ایجاد شود، زیرا زمان بارگذاری وب سایت خیلی کمتر می‌شود. علاوه بر این، وقتی که صفحه وب لود می‌شود، این صفحه دوباره بارگذاری نخواهد شد.

در ادامه با دو روش معمول بارگذاری صفحات وب آشنا خواهیم شد.

۱.۴.۲ بارگذاری صفحات در سمت سرور (SSR)

بارگذاری صفحات در سمت سرور (Server-side Rendering) یک راه قدیمی و مرسوم برای نمایش صفحات وب در مرورگر است. مراحل بارگذاری صفحات بصورت زیر می باشد:

در ابتدا کاربر درخواستی را معمولا از طریق مرورگر به سمت سرور ارسال می‌کند. سرور منابع را بررسی می‌کند، سپس محتوای HTML را بعد از اجرای کدهای مربوطه در سمت سرور، آماده می‌کند. این سند HTML آماده، به سمت مرورگر کاربر، برای نمایش و رندر، ارسال می‌شود. مرورگر سند HTML را دانلود می‌کند و آن را به کاربری که درخواست را ایجاد کرده، نمایش می‌دهد. سپس مرورگر فایل‌های جاوا اسکریپت را دانلود می‌کند و بعد از اجرای آن‌ها، صفحه رندر شده را، تعاملی می‌کند.

در این فرآیند، تمام بار و مسئولیت دریافت محتوای پویا، به HTML مربوط می‌شود، همچنین ارسال این HTML به مرورگر برعهده سرور است. از این رو این فرآیند، SSR نام دارد. مسئولیت رندر کامل HTML، باعث ایجاد باری بر memory و توان پردازشی سرور می‌شود. این قضیه باعث افزایش زمان بارگذاری یک صفحه در مقابل وب سایت‌های استاتیک، که محتوای پویا و متغیر برای نمایش ندارند، می‌شود.

۲.۴.۲ بارگذاری صفحات در سمت مرورگر (CSR)

بارگذاری صفحات در سمت مرورگر (Client-side Rendering) راه دیگری برای نمایش و رندر صفحه وب در مرورگر است. در CSR، بار ایجاد محتوای پویا و ایجاد HTML برای آن‌ها، برعهده مرورگر کاربر است. این روش توسط فریم‌ورک‌ها و کتابخانه‌های جاوااسکریپتی، نظیر: ReactJS، VueJS و AngularJS، پیاده‌سازی می‌شود. روند طبیعی رندر صفحات وب در این روش، مراحل زیر را دنبال می‌کند:

ر ابتدا کاربر درخواستی را معمولا از طریق مرورگر به سمت سرور ارسال می‌کند. در این مرحله، بجای سرور، می‌توان از یک شبکه تحویل محتوا یا CDN، برای ارسال HTML استاتیک، CSS و سایر فایل‌های ثابت و استاتیک به سمت کاربر استفاده کرد. مرورگر HTML و سپس فایل‌های جاوااسکریپت را در حالی که کاربر تنها یک نشانه‌ای از بارگذاری سایت می‌بیند، دانلود می‌کند. بعد از اینکه مرورگر فایل‌های جاوا اسکریپت را دانلود و اجرا کرد، درخواست‌هایی را به سمت API ایجاد می‌کند تا از طریق آن محتوای پویا را دریافت و آن را برای نمایش در مرورگر، بارگذاری کند. بعد از ارسال پاسخ از سمت سرور، محتوای نهایی در مرورگر کاربر پردازش می‌شود.

۳.۴.۲ مقایسه CSR, SSR

از آنجایی که هردو روش، در نحوه پردازش محتوا تفاوت دارند، هرکدام مزیت‌های مربوط به خودش را دارد.

مدت زمان بارگذاری صفحه، (زمانی که طول می‌کشد تا درخواست به سرور برسد و نتیجه در مرورگر، بارگذاری شود) در اولین بار در ،CSR مرورگر HTML پایه، CSS و تمام جاوااسکریپت‌هایی که نیاز است را لود می‌کند و آن‌ها را به یک HTML قابل استفاده برای مرورگر تبدیل می‌کند. این زمان معمولا بیشتر از دریافت یک HTML آماده و از پیش کامپایل‌شده به همراه اسکریپت‌های مورد نیاز خواهد بود. بنابراین SSR به زمان کمتری در این سناریو نیاز دارد. مدت زمان بارگذاری برای دومین بار و یا دفعات بعد در CSR کوتاه تر است چون تمامی اسکریپت‌ها در بار اول لود شده‌اند. این نکته را نیز باید ذکر کرد که روش CSR تاثیر منفی بر سئو (Search Engine Optimization) سایت دارد.

فصل ۳

فناوری های نوین طراحی وب سایت

۱.۳ مقدمه

فناوری اطلاعات به سرعت در حال پیشرفت است و پیروی از آخرین روندها بیش از هر زمان دیگری مهم است. بسیاری از گرایش های فناوری توسط وب هدایت می شوند، به همین دلیل پیشرفت در فناوری بسیار وابسته به روند توسعه وب است. برخی از روندهای توسعه وب سال ها پیش آغاز شده است اما هنوز باید مورد توجه قرار گیرند. نادیده گرفتن فناوری های نوین منجر به عقب افتادن سازمان ها در رقابت تجاری اشان خواهد شد. در این فصل به برخی از مفاهیم و فناوری های نوین توسعه وب در چند سال اخیر خواهیم پرداخت.

۲.۳ برنامه های تک صفحه‌ای تحت وب (SPA)

یک برنامه با رویکرد تک صفحه ای یا Single Page Application که به اختصار به آن SPA در برنامه نویسی هم می گویند، یک برنامه تحت وب است که فقط و فقط برای یک بار از سمت سرور بارگذاری می شود و برای هر تعامل دیگر کاربر را به سمت سرور نمی فرستد بلکه تنها محتوای مورد نیاز را با استفاده از API جاوا اسکریپت تغییر می دهد. این روش به کاربر اجازه می دهد بدون بارگذاری صفحات اضافه به هدف خود برسد و در نتیجه سرعت بسیار بالاتری را تجربه خواهد کرد.

نحوه عملکرد این نوع از برنامه ها را در فصل قبل ذکر کردیم. در ادامه با یکی از مطرح ترین فریم ورک‌های SPA جاوا اسکریپت بیشتر آشنا میشویم.

React ۱.۲.۳

این کتابخانه در سال ۲۰۱۱ توسط یکی از مهندسین نرم‌افزار شرکت فیسبوک معرفی شد و توسط دیگر جامعه های توسعه دهنده، مدیریت و نگهداری می گردد. ری‌اکت در واقعا یک کتابخانه ی متن باز است و کاملا رایگان در اختیار کاربران قرار گرفته است.

در js React ، برای قالب بندی به جای استفاده از جاوا اسکریپت معمولی از JSX استفاده می شود. JSX یک جاوا اسکریپت ساده است که از HTML پیروی می کند و از این دستورات تگ HTML برای ارائه زیر کامپوننت ها در React استفاده می کند. همچنین کتابخانه js React این اجازه را میدهد تا بتوانیم کامپوننت هایی با قابلیت استفاده مجدد نیز طراحی و ایجاد کنیم.

۳.۳ برنامه های پیش رونده تحت وب (PWA)

برنامه های پیش رونده تحت وب و یا Progressive Web Apps نسل جدید برنامه‌های تحت وب هستند که در سال ۲۰۱۵ توسط گوگل معرفی شد.

امروزه با توجه گستردگی استفاده از تلفن های همراه و پیشی گرفتن آن از دسکتاپ موجب شده تا تمرکز روی اپلیکیشن های موبایل بیشتر شود. از منظر رابط کاربری امروزه تقریبا هیچ تفاوتی بین برنامه‌های بومی و برنامه‌های تحت وب وجود ندارد و هر دوی آن‌ها امکانات مشابهی را با اختلافات اندکی در اختیار کاربر قرار می‌دهند. در حال حاضر، اغلب توسعه‌دهندگان برنامه‌های خود را هم در نسخه‌های موبایل و هم در بستر وب توسعه می‌دهند و با این کار امکان دسترسی به سرویس‌های خود را تا حد امکان افزایش می‌دهند. هر پلتفرم موبایل از زبان برنامه‌نویسی مختلفی استفاده می‌کند. در طرف مقابل، هیچ استانداردسازی مشخصی برای اپلیکیشن‌های وب وجود ندارد و توسعه‌دهندگان محدود به استفاده از چارچوب‌ها یا ابزارهای توسعه‌ی خاصی نیستند.

وب اپلیکیشن‌های پیش رونده از جدیدترین فناوری‌ها در ترکیب اپلیکیشن‌های موبایل و وب‌سایت‌ها بهره می‌گیرند. یک وب اپلیکیشن پیش رونده در واقع وب‌سایتی است که از فناوری‌های مدرن وب استفاده می‌کند؛ اما ظاهر و کارکرد آن همانند یک اپلیکیشن معمولی است. پیشرفت‌های اخیر در مرورگرها، سرویس ورکرها، کش‌ها و رابط‌های برنامه‌نویسی نرم‌افزار ،(API) توسعه‌دهندگان وب را قادر کرده تا وب اپلیکیشن‌هایی با قابلیت افزودن به صفحه‌ی خانگی سیستم‌عامل با امکان ارسال اعلان از سمت سرور و حتی عملکرد آفلاین توسعه دهند.

وب اپلیکیشن‌های پیش رونده در مقایسه با اپلیکیشن‌های بومی موجود در فروشگاه‌های نرم‌افزاری از مزیت اکوسیستم گسترده‌تر وب و پلاگین‌ها و آسودگی نسبی توسعه و حفظ وب‌سایت‌ها برخوردار هستند. ساخت یک وب‌سایت با صرف زمان کمتری قابل انجام است و نیازی به حفظ قابلیت پس‌سازگاریِ رابط‌های برنامه‌نویسی وجود ندارد؛ زیرا برخلاف چندپارگی نسخه‌های اپلیکیشن‌های بومی، تمام کاربرانِ نسخه‌ای یکسان از کد وب‌سایت را اجرا می‌کنند.

۱.۳.۳ ویژگی های برنامه‌های پیش رونده وب

بدون اتصال به اینترنت و حتی با سرعت پایین اینترنت قابل استفاده هستند.

برنامه های بومی نیازمند آپدیت از طریق فروشگاه های نرم افزاری هستند اما PWA ها به محض اینکه کاربر به اینترنت وصل باشد و محتوای جدیدی انتشار داده شود، آن محتوا بلافاصله در اختیار کاربر قرار می گیرد.

برنامه های پیش رونده وب، در بستر HTTPS قرار دارند در نتیجه از نظر مسائل امنیتی بسیار ایمن هستند.

این برنامه ها واکنش گرا (responsive) و کاملا انعطاف پذیر می باشند.

قابل استفاده در هر دستگاه و سیستم عاملی هستند به همین دلیل به آن ها پیش رونده می گویند.

به دلیل اینکه برنامه های پیش رونده وب در اصل یک وب سایت می باشند، از طریق موتورهای جستجو قابل یافتن هستند.

در ظاهر شبیه یک اپلیکیشن بومی هستند و رابط کاربری مشابه آن ها دارند.

۴.۳ وب‌اسمبلی (WASM)

وب‌اسمبلی استانداردی باز است که یک قالب جدید دستورالعمل‌های باینری را معرفی می‌کند. این فناوری نوید این را می‌دهد که برنامه‌ها با کارآیی بومیِ خود در بستر وِب اجرا شوند. به عبارت ساده‌تر می‌توان گفت، این فناوری امکان این را می‌دهد که کدهای نوشته شده با زبان‌های سطح بالاتر مانند C و C++ به ماژول Wasm کامپایل شوند که مستقیماً در مرورگرهای مدرن قابل اجرا هستند. وب اسمبلی معرفی شد تا محدودیت سرعت و پردازش برنامه های بزرگ و سنگین توسط جاوا اسکریپت در مرورگرها را از بین ببرد. در حالیکه این یک استاندارد نسبتا جدید است که در حال حاضر توجه بیشتری را به خود جلب کرده است، در حال حاضر در تمام مرورگرهای اصلی پشتیبانی می شود.

این فناوری تنوع در استفاده از زبان‌های برنامه‌نویسی را برای ساخت برنامه‌های تحت وب افزایش می‌دهد.

۵.۳ رابط کاربری صوتی (VUI)

رابط کاربری صوتی (Voice User Interface) تعامل انسان با رایانه‌ها را از طریق گفتار ممکن می‌سازد، با استفاده از تشخیص گفتار برای درک فرمان‌های گفتاری و پاسخ به سؤالات، و معمولاً با تبدیل متن به گفتار برای تولید پاسخ، این امر را ممکن می‌سازد. دستگاه فرمان صوتی (VCD) وسیله‌ای است که با رابط کاربری صوتی کنترل می‌شود.

رابط های کاربر صوتی در صنعت وب، در مرحله طراحی هستند، اما در نرم افزار رایج شده اند. مزیت اصلی این روند طراحی وب این است که نیازی به استفاده از دست افراد نیست.

۶.۳ برنامه ها و معماری بدون سرور

برنامه و معماری بدون سرور (Serverless Application and Architecture) مربوط به اجرای برنامه ها و خدمات بدون ساختن سرور است. این برنامه ها به جای اینکه توسط برنامه نویس توسعه یابند ، با اجاره سرور از فروشندگان خارجی کار می کنند. فروشندگان مشهور مختلفی مانند AWS ، Azure و

غیره وجود دارند که این امکانات سرور را برای توسعه وب سایت با مقیاس پذیری و امنیت بهتر ارائه می دهند. این فناوری در ساخت وب برای جلوگیری از عوارضی مانند بار بیش از حد سیستم، هزینه های بالای توسعه ، سرعت توسعه برنامه ها و موارد دیگر به وجود آمده است.

۷.۳ معماری میکروسرویس

امروزه با نیاز رو به رشدی که به برنامه های بزرگ به وجود آمده است، مفهومی تحت عنوان میکروسرویس هم در صنعت توسعهٔ نرم‌افزار رواج یافته است چرا که برنامه نویسان برای توسعه سیستم‌های بزرگ تجاری چاره‌ای جز به‌کارگیری از معماری میکروسرویس ندارند.

برای درک ماهیت میکروسرویس‌ها، ابتدا باید ببینیم که معماری Monolithic چگونه کار می‌کند. به طور کلی، در این نوع معماری ما سه لایه داریم تحت عناوین Models, Views, Controllers که در فصل قبل در مورد این نوع معماری صحبت شد.

این نوع معماری که تحت عنوان معماری MVC هم شناخته می‌شود دارای یکسری نواقصی است. به عبارت دیگر، تمامی لایه‌ها (مدل، ویو و کنترلر) زیر پرچم یک پلتفرم واحد مدیریت می‌شوند و ارتباط بسیار تنگاتنگی با یکدیگر دارند و مثلاً به سادگی نمی‌توان Model یک اپلیکیشنی که تحت معماری MVC نوشته شده را برداشته و در پروژهٔ دیگری استفاده کرد. در معماری Monolithic یک هستهٔ مرکزی داریم که کاربران و حتی دیگر سرویس‌ها از طُرق مختلف می‌توانند با آن ارتباط برقرار سازند و می‌بینیم گرچه خودِ این هستهٔ مرکزی ماژولار (بخش‌بندی) است، اما همگی تحت یک پلتفرم واحد کنار یکدیگر قرار گرفته‌اند و امکان مجزاسازی این ماژول‌ها وجود ندارد و یا اگر هم چنین امکانی وجود داشته باشد، بسیار دشوار خواهد بود.

اگرچه که در معماری MVC کدها ماژولار هستند و نسبت به گذشته که تمامی فایل‌ها در یک پوشه گذاشته می‌شدند و اصلاً مفهومی تحت عنوان ماژول در کار نبود شرایط به مراتب بهتر است، اما همان‌طور که گفته شد، هر ماژول برای کارکرد صحیح و اصولی خود نیاز به سایر ماژول‌ها دارد. به طور کلی، مشکلات مرتبط با معماری Monolithic را می‌توان به دسته‌های زیر تقسیم‌بندی کرد:

از آنجا که یک سورس‌کد اصلی بیشتر وجود ندارد، تک‌تک اعضای تیم، از دولوپر فرانت‌اند گرفته تا برنامه‌نویس‌های سمت سرور و غیره، باید روی یک سورس‌کد کار کنند و به طور مثال اگر کسی بخواهد صرفاً روی بخش مدیریت کاربران کار کند، باید کل پروژه را دریافت کرده، یک هاست لوکال پیکره‌بندی کرده و شروع به کار روی پروژه کند.

یک تغییر کوچک در یکی از ماژول‌ها، ممکن است عملکرد دیگر ماژول‌ها را تحت‌تأثیر خود قرار دهد.

درست است که در این نوع معماری مفهومی داریم تحت عنوان MVC، اما در طول زمان این سه لایه آن‌قدر با یکدیگر عجین خواهند شد که به سختی می‌توان مرز مشخصی مابین آن‌ها ایجاد کرد.

کامپوننت‌ها را به سادگی نمی‌توان با یک معماری جدیدتر و بهینه‌تر جایگزین کرد چرا که کل معماری نرم‌افزار باید دستخوش تغییر گردد.

تنوع تکنولوژی من‌جمله زبان‌های برنامه‌نویسی، دیتابیس‌های مختلف، لایبرری‌ها و فریمورک‌های مختلف وجود نخواهد داشت و اگر هم این‌گونه باشد، ارتباط برقرار کردن مابین آن‌ها بسیار دشوار خواهد بود.

اینجا است که پای میکروسرویس‌ها به میان می‌آید و مزایایش منجر بدین گشته تا کمپانی‌های بزرگی همچون آمازون یا نتفلیکس به استفاده از میکروسرویس‌ها ترغیب شوند.

۱.۷.۳ معماری میکروسرویس چیست؟

میکروسرویس روشی به منظور تقسیم‌بندی کردن یک اپلیکیشن (نرم‌افزار) به بخش‌ها یا سرویس‌های کوچک، سبُک، مستقل از یکدیگر و قابل‌مدیریت است. به عبارت دیگر، میکروسرویس یک معماری توسعهٔ نرم‌افزار به اصطلاح توزیع شده (distributed) است.

این نوع سرویس‌ها صرفاً به منظور کنترل کردن یک وظیفه خاص طراحی می‌شوند. به طور مثال، یک سرویس صرفاً وظیفهٔ مدیریت کاربران را دارا است و سرویس دیگر فقط و فقط برای بخش جستجوی سایت کاربرد دارد و با توجه به اینکه میکروسرویس‌ها مجزا و مستقل از یکدیگر هستند، به راحتی قادر خواهیم بود تا آن‌ها را با زبان‌های برنامه‌نویسی مختلفی نوشته و برای ذخیره‌سازی داده‌های مرتبط با آن‌ها نیز از سیستم‌های مدیریت دیتابیس مختلفی استفاده کنیم (به عبارت دیگر، جاهایی که نیاز به ذخیره‌سازی سنتی داده‌ها داریم می‌توانیم از MySQL استفاده کنیم و جاهایی دیگر هم به خاطر ساختار غیرقابل پیش‌بینی دیتای خود می‌توانیم به استفاده از دیتابیس‌های به اصطلاح NoSQL بپردازیم.)

۲.۷.۳ مزایای استفاده از میکروسرویس‌ها

امروزه ماژولار بودن به یک مزیت رقابتی در هر صنعتی مبدل شده‌اند؛ از مبلمان IKEA گرفته تا گوشی‌های موبایل ماژولار و حتی اسباب‌بازی‌های LEGO و این در حالی است که ایدهٔ پشت میکروسرویس‌ها این است تا این امکان به دولوپرها داده شود تا اپلیکیشن‌های خود را بر مبنای اجزا یا سرویس‌هایی که مستقل از یکدیگر هستند و به سادگی قابل‌تغییر، حذف و به‌روزرسانی می‌باشند توسعه دهند بدون اینکه کل اپلیکیشن تحت‌الشعاع قرار گیرد. روی هم رفته، مهم‌ترین مزایای استفاده از معماری میکروسرویس عبارتند از:

بر خلاف معماری مونولیتیک، در یک اپلیکیشنی که در آن از معماری میکروسرویس استفاده شده باشد سرویس‌ها هرگز بر اساس معماری MVC تقسیم‌بندی نمی‌شوند بلکه بر اساس کاری که انجام می‌دهند به بخش‌های مختلف تقسیم می‌شوند. به عبارت دیگر، یک سرویس همچون آپلود فایل شامل بخش‌هایی همچون رابط کاربری، مدل‌های مرتبط با دیتابیس، کنترلر، سیستم لاگینگ و ... خواهد بود.

یکی دیگر از مزیت‌های میکروسرویس‌ها این است که ما مجبور به استفاده از صرفاً یک زبان برنامه‌نویسی یا فناوری در کل پروژه نمی‌شویم. در واقع، با توجه به اینکه امروزه برخی زبان‌های برنامه‌نویسی برای حوزه‌های خاصی تخصصی‌تر هستند و استفاده از زبانی که اختصاصاً برای کار خاصی طراحی شده پرفورمنس اپلیکیشن ما را بالاتر می‌برد، با استفاده از میکروسرویس‌ها قادر خواهیم بود تا بسته به نوع سرویس مد نظر خود از چندین زبان برنامه‌نویسی و فناوری مختلف استفاده کرده و پرفورمنس را به حد اعلای خود برسانیم.

علاوه بر موارد فوق، میکروسرویس‌ها اصطلاحاً Scalable (قابل‌توسعه) هستند. ماهیت مستقل ماژول‌های مختلف یک میکروسرویس این امکان را برای‌مان فراهم می‌آورند تا با استفاده از زبانی خاص، دیتابیسی خاص و همچنین سروری خاص به توسعهٔ اپلکیکشن مد نظر خود بپردازیم و در صورت نیاز صرفاً منابع همان پلتفرم را ارتقاء دهیم.

۳.۷.۳ معایب استفاده از میکروسرویس‌ها

این نوع معماری توسعهٔ اپلیکیشن نقاط ضعف خاص خود را هم دارا است که برخی از مهم‌ترین آن‌ها عبارتند از:

از آنجا که هر سرویس مسئول انجام تَسک خاصی است، در اپلیکیشن‌ها بسیار بزرگ تعداد سرویس‌های بی‌شماری خواهیم داشت و از همین روی برقراری ارتباط مابین این سرویس‌ها و از همه مهم‌تر مانیتور کردن آن‌ها کاری بس دشوار خواهد بود.

با توجه به اینکه سرویس‌ها برای برطرف کردن نیازهای خود دیگر سرویس‌ها را فراخوانی می‌کنند، رصد کردن آن‌ها و بالتبع فرایند دیباگینگ بسیار دشوار خواهد شد.

هر سرویس لاگ‌گیری اختصاصی خود را دارا است و از همین روی هیچ سیستم مانیتورینگ مرکزی برای بررسی لاگ‌ها وجود ندارد و در چنین شرایطی نیاز به یک سیستم مدیریت لاگ مرکزی وجود خواهد داشت.

با توجه به اینکه ارتباط سرویس‌ها با یکدیگر از طریق API است، تعداد در خواست ها نسبت به یک معماری مونولیتیک به مراتب بیشتر خواهد بود.

دیپلوی کردن اپلیکیشن‌هایی که با استفاده از معماری میکروسرویس طراحی شده‌اند به صورت دستی مشکل است و در چنین شرایطی نیاز به ابزارهای اتوماسیون پیشرفته خواهد بود.

امروزه اکثر اپلیکیشن‌ها نیاز دارند تا در آنِ واحد چندین رکورد را در دیتابیس حذف یا به‌روزرسانی کنند. در چنین مواقعی با توجه به اینکه در معماری مونولیتیک صرفاً یک دیتابیس وجود دارد، اینکار به سادگی صورت خواهد گرفت اما در میکروسرویس‌ها چنین حذف یا به‌روزرسانی‌هایی چالشی خواهند شد چرا که ممکن است رکوردی در دیتابیس یکی از سرویس‌ها در یک سرور خاص به همراه رکورد دیگری در سرویس دیگری روی سرور دیگری بخواهند با یکدیگر سینک شوند.

با توجه به اینکه ممکن است از چندین زبان برنامه‌نویسی و تکنولوژی مختلف در چنین اپلیکیشن‌هایی استفاده شود، هزینهٔ نگهداری چنین سیستم‌ها گاهی اوقات زیاد می‌شود به طوری که مثلاً نیاز به استخدام برنامه نویس زبان‌های مختلف خواهیم داشت.

ورژن‌بندی میکروسرویس‌ها باید به صورت مجزا از یکدیگر صورت گیرد و اینجا است که نیاز داریم تا مشخص کنیم به طور مثال کدام ورژن سرویس A با کدام ورژن سرویس Z باید دیپلوی شود.

مستندات‌سازی چنین اپلیکیشن‌هایی مشکل‌تر است چرا که با توجه به ماهیت مستقل هر ماژول، سرویس‌ها باید به صورت کامل مستندسازی شوند.

۸.۳ آشنایی با پلتفرم Docker

۱.۸.۳ Docker چیست ؟

داکر یک پلتفرم متن باز برای توسعه، توزیع و اجرای برنامه ها در یک محیط ایزوله‌شده (کانتینر) است. داکر به شما این امکان می دهد که برنامه های خود را از زیرساخت هایش جدا کنید تا بتوانید سریع تر نرم افزار خود را تولید کنید.

داکر ایمیج (docker image) یک قالب فقط خواندنی است که دارای دستورالعمل های برای ایجاد داکر کانتینر (docker container) است.

معمولا، یک ایمیج بر اساس ایمیج دیگری، با برخی دستور های اضافی ساخته می شود. به عنوان مثال، ممکن است ایمیجی بسازید که بر اساس ایمیج ubuntu باشد. شما ممکن است ایمیج های خود را ایجاد کنید و یا اینکه فقط از ایمیج های ایجاد شده توسط دیگران و منتشر شده در یک رجیستری استفاده کنید.

برای ساختن ایمیج، ابتدا باید یک Dockerfile ایجاد کنید، سپس برای تعریف مراحل مورد نیاز برای ایجاد ایمیج و اجرای آن را معین می کنید. هر دستورالعمل در یک Dockerfile یک لایه در ایمیج ایجاد می کند. وقتی Dockerfile را تغییر می دهید و تصویر را دوباره می سازید ، فقط لایه هایی که تغییر کرده اند دوباره ساخته می شوند. این بخشی از مزایای استفاده از داکر به جای سایر فناوری های مجازی سازی است که ایمیج ها را بسیار سبک، کوچک و سریع می کند.

کانتینر یک نمونه قابل اجرا از یک ایمیج می باشد. با استفاده از API Docker یا CLI می توانید کانتینر ایجاد کنید و یا آن را متوقف، جابجا یا حذف کنید.

به طور پیش فرض، یک کانتینر از سایر کانتینر های دیگر و دستگاه میزبان ایزوله شده است. به راحتی می توان کنترل کرد که شبکه، فضای ذخیره سازی یا سایر زیر سیستم های کانتینر از کانتینرهای دیگر یا از دستگاه میزبان ایزوله هستند.

یک کانتینر با ایمیج و همچنین گزینه های پیکربندی که هنگام ایجاد یا راه اندازی به آن ارائه می دهید، تعریف می شود. وقتی کانتینر حذف شود، هر تغییری در حالت آن که درحافظه مداوم ذخیره نشده باشد، از بین خواهد رفت.

۲.۸.۳ مزایای استفاده کردن از داکر

- **ایزوله سازی :** هر نرم‌افزار در محیطی مجزا از سایر نرم‌افزارها اجرا می‌شود و یک نرم‌افزار با سایر نرم‌افزارها تداخل ایجاد نمی‌کند.
- **اجرای توزیع شده بین چند سرور و چند نمونه روی هر سرور :** به کمک ساختار غیر وابسته کانتینر به راحتی می‌توان چندین نمونه از نرم‌افزار را روی چندین سرور اجرا کرد.
- **مدیریت منابع :** به کمک داکر می‌توان مشخص کرد که هر برنامه چه میزان از منابع را در اختیار داشته باشد.

۹.۳ ادغام مستمر و تحویل مستمر CI/CD

ادغام مستمر (Continuous Integration) و تحویل مستمر (Continuous Delivery) دو فرآیند علمی و روش شناخته شده در طراحی و توسعه نرم‌افزار هستند که توسعه‌دهندگان برای اتمام سریع و با کیفیت پروژه‌های نرم‌افزاری از آن استفاده می‌کنند. CI/CD به گروه‌های توسعه نرم‌افزار امکان می‌دهد تا تغییرات اعمال شده در کدها را سریع‌تر و با اطمینان بیشتر پیگیری کنند تا نیازهای مشتریان در کوتاه‌ترین زمان ممکن به سرانجام برسد.

ادغام مستمر یک فلسفه کدنویسی و مجموعه‌ای از شیوه‌های عملی طراحی شده برای هدایت گروه‌های توسعه است که به آن‌ها اجازه می‌دهد به‌طور مداوم تغییرات کوچک اعمال شده در کدهای خود را به مخازن کنترل نسخه (version control) ارسال و صحت آن را بررسی کنند. با توجه به این‌که امروزه گروه‌های توسعه برای تولید اکثر اپلیکیشن‌ها به پلتفرم‌ها و ابزارهای مختلفی نیاز دارند، این گروه‌ها باید بتوانند تغییرات اعمال شده را یکپارچه‌سازی و اعتبارسنجی کنند.

هدف از ادغام مستمر ایجاد روشی سازگار و خودکار برای ساخت، بسته‌بندی و آزمایش نرم‌افزارهای کاربردی است. ثبات در فرآیند ادغام به گروه‌های توسعه اجازه می‌دهد بهتر از گذشته کدها را ویرایش و تغییرات را اعمال کنند. رویکردی که تعامل بیشتر اعضای گروه را به همراه داشته و باعث خلق نرم‌افزارهای با کیفیت‌تری می‌شود.

در ادغام مستمر هر تغییری در کد، یک زنجیره ساخت و آزمایش را برای پروژه‌ای که تغییرات در آن اعمال شده ایجاد می‌کند و بازخورد این تغییر کد را به برنامه‌نویسانی که این تغییرات را اعمال کرده‌اند اعلام می‌کند. مهم‌ترین مزیت ادغام و تحویل مستمر (از آزمایش و استقرار تا خروجی از برنامه) انجام تمامی فرآیندها در مدت زمان کوتاه است.

تحویل مستمر که از ادغام مستمر دنباله‌روی می‌کند، تحویل اپلیکیشن‌ها به محیط‌های زیربنایی منتخب را خودکار می‌کند. امروزه اغلب گروه‌های توسعه علاوه بر تولید در گروه‌های دیگری همچون تیم‌های آزمایش (تیم‌هایی که مسئولیت آزمایش نرم‌افزارها و شناسایی باگ‌ها را عهده‌دار هستند.) کار می‌کنند. تحویل مستمر این اطمینان خاطر را می‌دهد که یک روش خودکارسازی وجود خواهد داشت که در قالب اعلانی، تغییرات اعمال شده در کدها را به این محیط‌های گوناگون ارسال می‌کند.

واژه‌نامه فارسی به انگلیسی

ابر لینک ..... hyper link
ابر متن ..... hyper text
استاتیک ـ ثابت ..... static
انتقال دادن ..... transfer
ایمیج ـ تصویر ..... image
برنامه ..... application
برنامه نویسی ..... programming
بهبود ..... optimization
بارگذاری کردن ..... render
برنامه تحت وب ..... web application
بدون سرور ..... serverless
پیش رونده ..... progressive
تک صفحه‌ای ..... single page
توزیع شده ..... distributed
تحویل ..... delivery
حالت ..... state
داده ..... data
داینامیک ـ پویا ..... dynamic
رابطه ..... relation
رابط ..... interface
رسانه ..... media
سند ..... document
سخت افزار ..... hardware
سرور ..... server

سرویس دهنده‌ـ سرویس گیرنده ..... client-server
صدا ..... voice
طراحی وب ..... web design
ظاهر ـ دید ..... view
غیر همگام ..... asynchronous
فریم ورک ـ چارچوب نرم افزاری ..... framework
قابل توسعه ..... scalable
کاربر ..... user
کانتینر ـ ظرف ..... container
کنترل کننده ..... controller
کتاب خانه ..... library
کنترل نسخه ..... version control
ماشین جستجو ..... search engine
مستمر ..... continues
مشتری ..... client
مدل ..... model
معماری ..... architecture
مرورگر ..... browser
میکرو ..... micro
نرم افزار ..... software
نمایشی ..... representational
همگام ..... synchronous
واکنش گرا ..... responsive

مراجع

[۱] بهزاد مرادی. میکرو سرویس. https://sokanacademy.com/blog/۵۰۳۲/، ۱۳۹۶. Accessed: ۱۳۹۶_۰۸_۰۳.

[۲] محسن آقاجانی. ci-cd. https://www.shabakeh-mag.com/workshop/۱۷۰۱۳، ۱۳۹۹. Accessed: ۱۳۹۹_۰۳_۰۸.

[۳] docker introduction. https://docs.docker.com، ۲۰۲۱.

[۴] web-application. https://www.ezweb.ir، ۱۳۹۹. Accessed: ۱۳۹۶_۱۲_۰۸.

[۵] World-wide-web. https://en.wikipedia.org، ۲۰۲۱.