صفر و یک

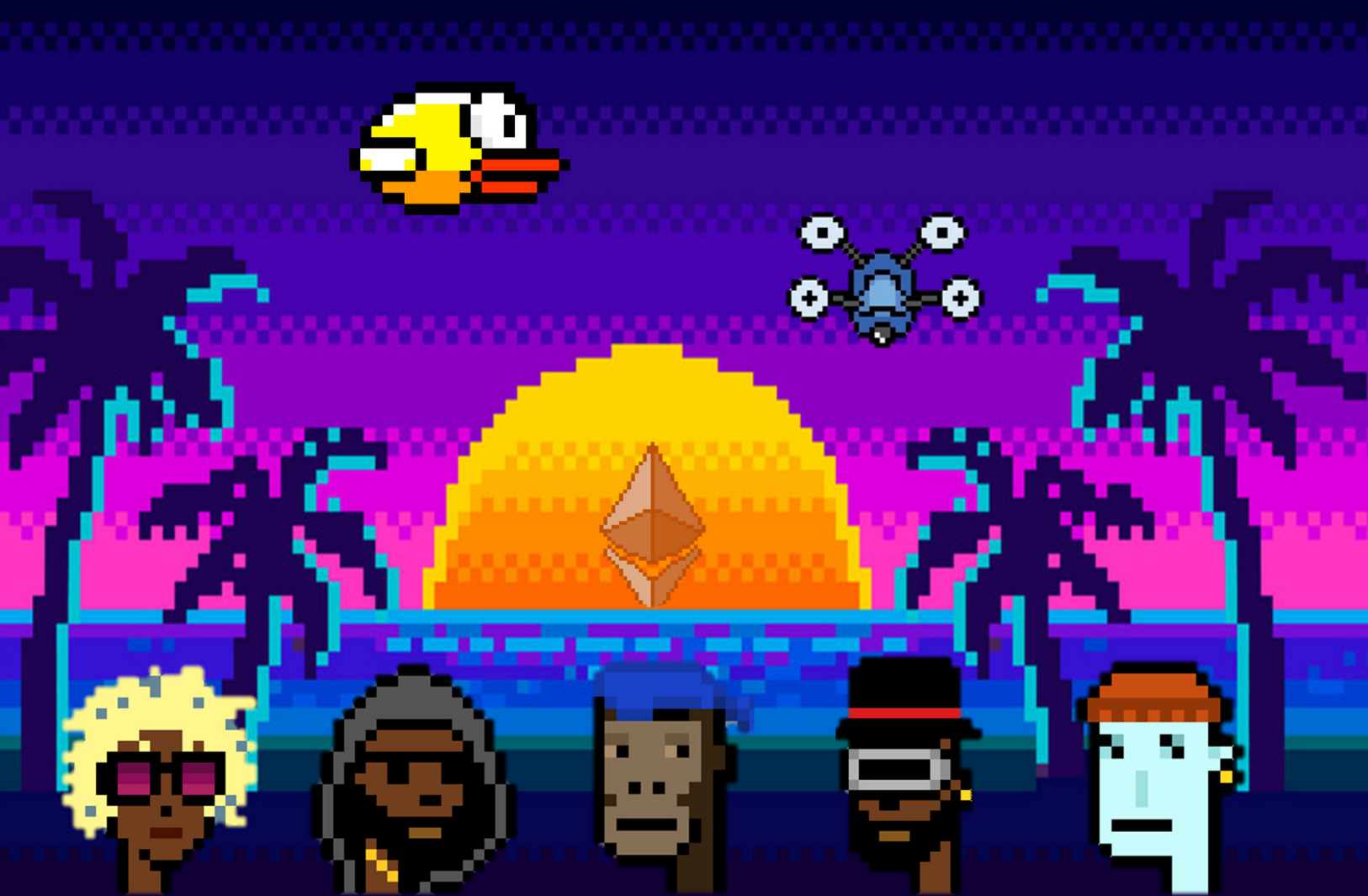
فصلنامه صفر و یک، نشریه علمی-دانشجویی انجمن علمی مهندسی کامپیوتر دانشکده مهندسی دانشکدگان فارابی دانشگاه تهران

دوره دوم، شماره 11، تابستان 1400

در این شماره بخوانید:

نسخه الکترونیکی نشریه رو از سایت و کانال انجمن علمی دریافت کنید!

- @CESA_UT
- CESA.UT.AC.IR



بسم الله الرحمن الرحیم

صفر یک

فصلنامه علمی-دانشجویی صفر و یک
دوره دوم، شماره یازدهم، ۳۱ شهریور ۱۴۰۰

شناسنامه

صادر کننده مجوز
دانشگاه تهران

صاحب امتیاز
انجمن علمی مهندسی کامپیوتر دانشکده مهندسی دانشکدگان فارابی دانشگاه تهران

مدیرمسئول
محمدعلی آصف
MAAsef@ut. ac. ir

سردبیر
مهران آدووند
Mehran. Advand@ut. ac. ir

هیئت تحریریه
دانیال فرهنگی، علیرضا زینی جلودار، صبا حیدری دوست، امیررضا پارسی زاده، محمدعلی آصف، عارف برهانی، محمد حسین شیرازی، امیرعلی خانه عنقا، عارفه عطایی نظری، مهدی ناصری، مهران آدووند، فائزه خادم، امیرحسین زین الدینی، حسین شعله رسا، علی اسدی

ویراستار
احسان شیرعلی

طراحی و صفحه آرایی
سیدمحمدصالح قطبانی

طراح جلد
محمدعلی آصف

ارتباط با نشریه و انجمن
وبسایت نشریه: cesasj. ut. ac. ir
وبسایت انجمن: cesa. ut. ac. ir
کانال اطلاع رسانی نشریه: @cesasj
انجمن در شبکه‌های اجتماعی: @Cesa_ut

سخن آغازین

خیلی خوش آمدید به شماره یازدهم از دوره دوم فصلنامه صفر و یک.

در ابتدا قدردانی میکنم از همه ی دست اندرکاران این شماره، نویسندگان، طراحان، کسانی که با نظرات و انتقاداتشون باعث بهبود مسیر صفر و یک شدند و همچنین مدیر مسئول نشریه که در این شرایط سخت، اعضای تیم نشریه را دور هم جمع کردند تا کارهای نشریه به بهترین شکل انجام شود.

در این شماره سعی کردیم که مطالب مفیدی در زمینه های مختلف علوم کامپیوتر منتشر کنیم تا شما عزیزان با طیف وسیعی از موضوعات آشنا شوید. امید است این مطالب باعث ایجاد انگیزه و سرنخی در مطالعه‌کنندگان گرامی شود و استارتی برای شروع ماجراجویی در رَسته های جذاب علوم کامپیوتر شود.

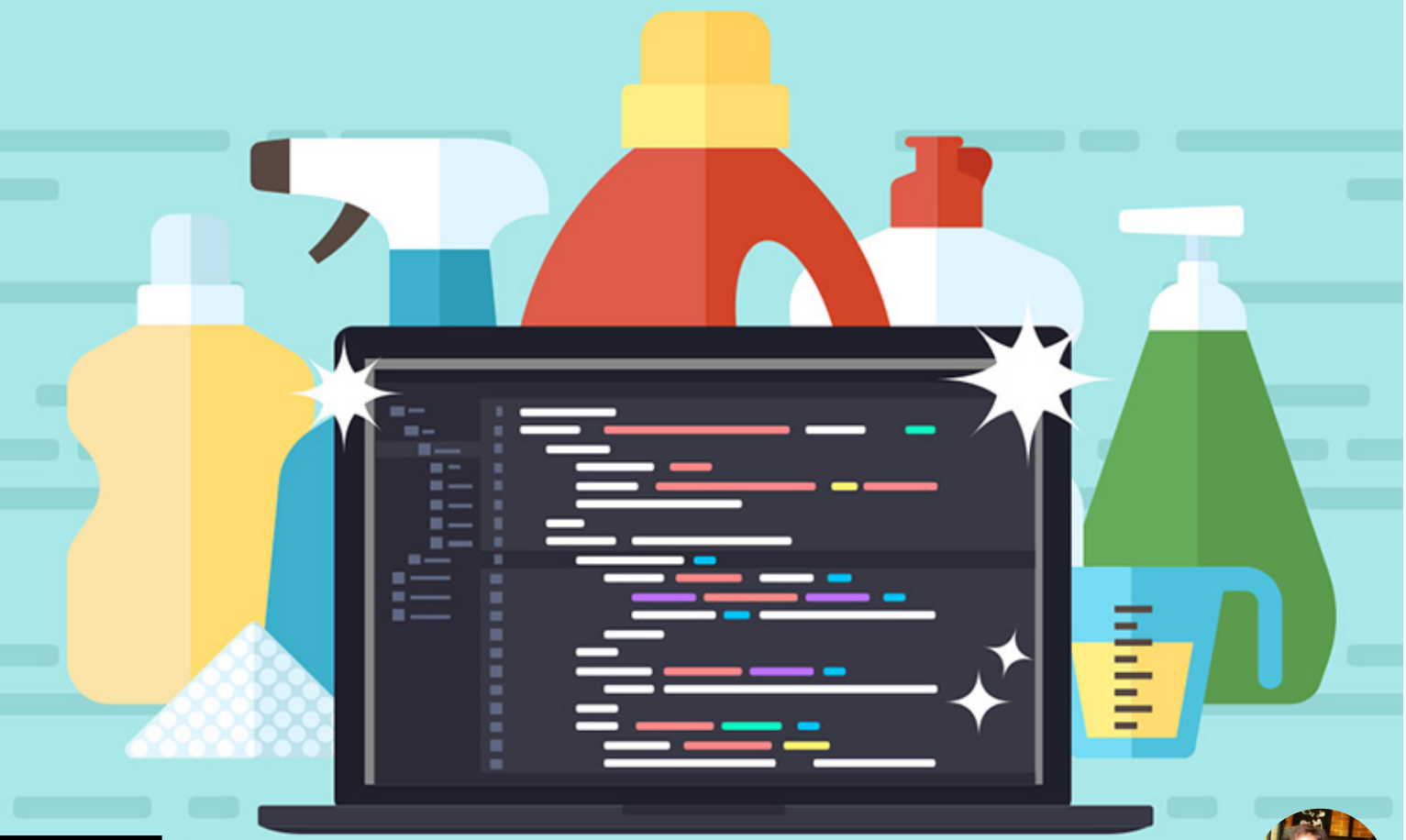
تعدادی از مقالات این شماره از فصلنامه متعلق به پژوهش دانشجویان ورودی های مختلف مهندسی کامپیوتر دانشکدگان فارابی دانشگاه تهران در دروس مختلف می باشد. این نشان دهنده گسترش هرچه بیشتر ارتباط میان نشریه ، اساتید و دروس است. این یکی از اساسی ترین اهداف تیم ما از زمان شروع کار در نشریه می باشد و امیدواریم این ارتباط در آینده نیز برقرار بماند و مطالب تخصصی تر برای علاقه مندان در نشریه به چاپ برسد.

سعی نمودیم که ایرادات و مشکلات شماره‌های قبل را در این شماره کاهش دهیم و فصلنامه باکیفیت‌تر و مناسب تری را در اختیار شما قراردهیم. امیدواریم که هم از نظر گرافیکی و صفحه آرایی و هم از نظر محتوای مورد توجه شما قرار گیرد و جرعه ای از تشنگی شما جویندگان علم و دانش را برطرف کند.

ارادتمند، مهران آدووند، سردبیر

فهرست مطالب

کدنویسی تمیز (Clean Code)



امیرعلی خانه عنقا
ورودی ۹۷

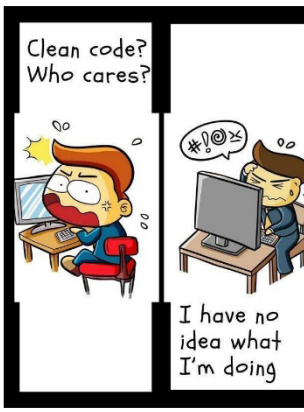
به عنوان فردی که همیشه در رسیدن به اهدافم عجله دارم ، حرف زدن در ارتباط با کد مرتب و تمیز کمی دور از انتظار است اما به تجربه برایم اثبات شد در بسیاری از کارها پیش بینی اولیه و در نظر گرفتن جزئیات کوچک ، می تواند تاثیر به سزایی در سرعت انجام کار داشته باشد.
به خوبی مقاومت های بچگانه خودم در مقابل تمیزنویسی را به یاد دارم ، تا مدت زیادی حتی با دانش بر این موضوع که در صورت نامگذاری نامناسب در خطر از دست دادن نمره بودم از گذاشتن نام هایی همچون "sdf" , "ssss" , "sss" , "ss" , "s" ، بر روی متغیرها هیچ ابایی نداشتم ، چون تصور می کردم ، حرف عموم در ارتباط با تمیز نویسی کد از روی وسواس و ترس زیادی از بهم ریختگی کد و یا حتی قضاوت شدن از سوی دیگران است .
علاوه بر مسخره بازی هایم بر سر نام ها ، ممکن بود در یک فایل حداقل ده کلاس و یا تابع مختلف نوشته باشم و تنها با کامنت کردن قسمت هایی که مورد نیاز من نبود ، اقدام به اجرای یک برنامه میکردم اما گاهی اوقات خوردن به یک سنگ سخت ابدا اتفاق بدی نیست . در شروع اولین پروژه شبه واقعی ام آنچنان در میان تک تک متد هایی که خودم نوشته بودم دست و پا میزدم که برایم قابل باور نبود ، برای اولین بار در زندگی احساس میکردم شاید علاقه ام به کدنویسی واقعی نبوده و یک برداشت کاملا سطحی از موضوع بوده است (که البته آن دوران تاریک خیلی سریع سپری شد).
کد یک برنامه نویس همچون امضای او خواهد بود و در یک محیط حرفه ای ممکن است از آن برای مقایسه افراد استفاده می شود ، با توجه به این موضوع در این مطلب سعی میکنم توجه شما را به اهمیت نگارش صحیح کد جلب کنم و سپس تلاش میکنم زمینه آشنایی شما با مفهوم "clean code" را ایجاد کنم و با چندین مثال بخش هایی از آن را به تصویر بکشم ، با این امید که مقاومت خودم در مقابل تمیز نویسی را تا حد امکان جبران کنم .
در ابتدای شروع یک پروژه معمولا بسیاری از کارها قابل انجام و راحت بنظر می آید اما هرچه یک پروژه جلو می رود، مشکلات بیشتری به وجود می آید ، ارتباط زیاد بین کلاس ها ، مدیریت درست منابع ، مدیریت درست وابستگی ها بین کلاس ها و . . . ، حال تمام این مشکلات حتی با یک دید معماری بسیار عالی در ابتدای کار باعث ایجاد مشکلاتی

در فازهای میانی و پایانی پروژه می شود و احتمال قریب به یقین نیازمند بهبود بعضی از قسمت های اولیه خواهید بود ، فرض کنید مدت زمان زیادی از نوشتن کدهای ما در فاز اول می گذرد ، و نیاز به اعمال تغییرات در آن ها داریم ، در صورتی که نامگذاری های ما مناسب نباشد و یا حتی توضیحات مناسب در کامنت ها نداده باشیم بایستی زمان بسیار زیادی مجبور به رمزگشایی کد های خودمان باشیم.

یک فرضیه دیگر را مطرح میکنم ؛ یک ایده طلایی به سراغتان آمده ، یک پروژه بینظیر که شما را به مدیرعاملی یک شرکت برپایه ی همین پروژه سوق میدهد ، شروع به نوشتن برنامه میکنید و تا مقدار خوبی رو به جلو حرکت میکنید اما نیاز به کمک دارید ممکن است آنقدر حواشی پروژه زیاد شده باشد که دیگر نمیتوانید به صورت مستقیم به عنوان یک کدنویس در پروژه مشغول باشید ، پس سعی میکنید افراد دیگری را به پروژه خود اضافه کنید ، فکر میکنید چه مقدار طول می کشد تا یک کدنویس دیگر از اسپاگتی شما سر در بیاورد؟!

در انتها باید گفت مثال های بالا برای زمانی بود که خود شما فرد محوری تیم هستید ، در صورتی که این اتفاق نیفتد و به طور مثال شما جزئی از یک پروژه بزرگتر باشید و یا در جایی مشغول به کار باشید ، کیفیت کد های نگاشته شده توسط شما ، قابلیت های شما را نشان می دهد و همچنین سطح ارشدیت شما را تعیین می کند .

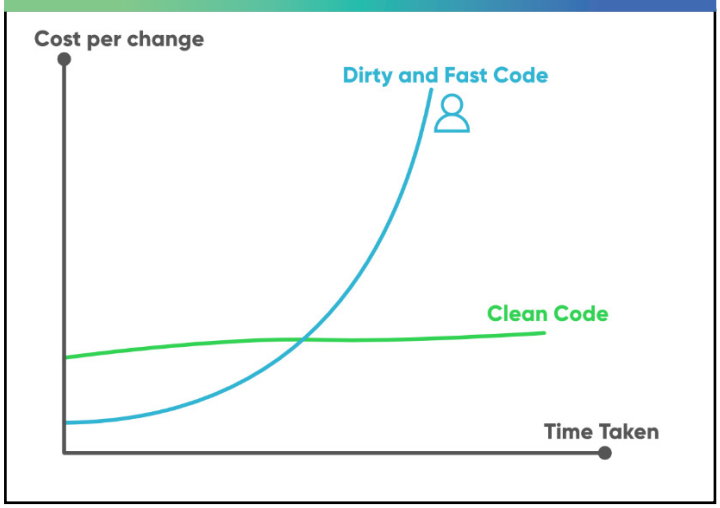
اگر تا الان مجاب نشدید که بهتر است کمی به فکر یادگیری طرز صحیح کدنویسی باشید این مطلب چیز بیشتری برای ارائه به شما ندارد ولی در غیر این صورت در ادامه چندین نکته ابتدایی ولی با درجه اهمیت بالا نگاشته شده است تا شروعی باشد بر یادگیری این مبحث ظریف اما کاربردی :



۱- معمولا نام کلاس ها و متغیر ها «اسم» است در صورتی که نام تابع ها و متد ها «فعل» هستند ، دلیل این کار به ماهیت آن ها برمیگردد ، به طور مثال هر تابع مسئولیت انجام کاری را به عهده دارد و فعلی را رقم می زند ، که نام آن نیز از همین موضوع نشأت می گیرد .

۲- در انتخاب نام ها تا حد امکان از الفاظ عامیانه استفاده نکنید ، بیشتر برنامه نویس ها حس شوخی لفظی بالایی دارند و میل استفاده از آن در کد برایشان غیرقابل اجتناب است ولی به یاد داشته باشید که کار خود را جدی بگیرید و همیشه حرفه ای بودن خود رابه نمایش بگذارید، به طور مثال اسم تابع "kill" را "bemir" نگذارید !!.

۳- با توجه به این که ادیتور های بسیار قوی برای بیشتر زبان ها وجود دارد که معمولا دارای افزونه هایی برای کمک در نگارش کد هستند ، لطفا از کوتاه نویسی و خلاصه نویسی اسم ها پرهیز کنید و هر اسم را به صورت کامل بنویسید . بسیاری از محیط های برنامه نویسی چون آردوینو از داشتن محبتی همچون افزونه codota (یک افزونه برای پیش بینی ادامه کد شما) محروم هستند و نگارش به صورت خلاصه تنها در این محیط ها قابل



توجیه است .

۴- سعی کنید اسم متد های مشابه در کلاس های مختلف را یکسان انتخاب کنید به طور مثال بجای این که در یک کلاس از واژه "delete" استفاده کنید و در کلاسی دیگر از واژه "remove" استفاده کنید ، سعی کنید یکپارچگی را در کد خود رعایت کنید و از یکی از آن ها استفاده بکنید.

۵- سعی کنید هر تابع مسئولیت یک وظیفه باشد ، همچنین تا جای امکان توابع را مختصر و کوتاه نگاه دارید.

۶- معمولا عملیات های توابع نباید بر روی برنامه اصلی تاثیری بگذارد بلکه باید از خروجی های آن استفاده کرد به این معنا که اثرات جانبی تابع باید حداقل باشد.

۷- تا جای امکان از تعدد ورودی های تابع جلوگیری کنید (به صورت نرمال هر تابع باید دو ورودی داشته باشد) بیشتر از آن معمولا باعث سخت شدن خوانش و درک تابع برای دیگران می شود.

۸- کامنت گذاری بیشتر برپایه توضیح کد و یا پاورقی های کدنویس است اما استفاده از آن به عنوان ذخیره کدهای قدیمی و یا اشتباه کاملا اشتباه است و تا جای امکان از این کار پرهیز کنید ، در ازای آن سعی کنید بجای کامنت کردن این کد ها آن هارا به شیوه درست پیاده سازی کنید و از این کامنت های بیهوده رها شوید.

۹- معمولا یک کد خوانا نیازی به کامنت های متعدد ندارد و باید بتواند منظور خود را به دیگران برساند ، این به معنای پرهیز از کامنت گذاری نیست به معنی آن است که کد را با کامنت های بیهوده که از خود کد قابل برداشت است ، پر نکنیم.

۱۰- با توجه به ابزارهای کنترل پروژه مختلف مانند گیت و همچنین وجود ابزار مختلف تست توابع و متد ها به کامنت های خود دقت کنید ، آیا بسیاری از آن ها قابلیت صرفنظر کردن ندارند؟!

امیدوارم این مثال ها مطبوع شما بوده باشد ، همچنین با توجه به این که انسانی از خطا عاری نیست درصورت وجود مشکل در نگارش و یا محتوا به بزرگی خود نادیده بگیرید. :)

پرایوسی‌کوین‌ها (Privacy Coins) و مکانیزم آنها

عارفه عطایی نظری
ورودی ۹۸

بلاکچین ها به هرکسی اجازه می دهد آدرس ها و معاملات عمومی را در شبکه خود مشاهده کند، که ردیابی سپرده ها و برداشت های افراد را نسبتاً ساده می کند. با این حال سکه های حریم خصوصی محور دارای دو جنبه متفاوت هستند. ناشناس بودن (anonymity) و عدم قابلیت ردیابی (untraceability). ناشناس بودن هویت یک تراکنش را پنهان می کند، در حالی که با نبودن قابلیت ردیابی عملاً غیرممکن است که شخصی ثالث با استفاده از ابزاری همانند تجزیه و تحلیل بلاک چین، ردپای تراکنش ها را دنبال کند.

اگرچه تجزیه و تحلیل زنجیره ای کار ساده ای نیست، اما چندین شرکت در آن تخصص دارند و برای انجام این کار پول زیادی دریافت می کنند. اغلب این یک بازی مستمر بین کوین های حریم خصوصی و نرم افزاری است که سعی در نظارت بر آنها دارد.

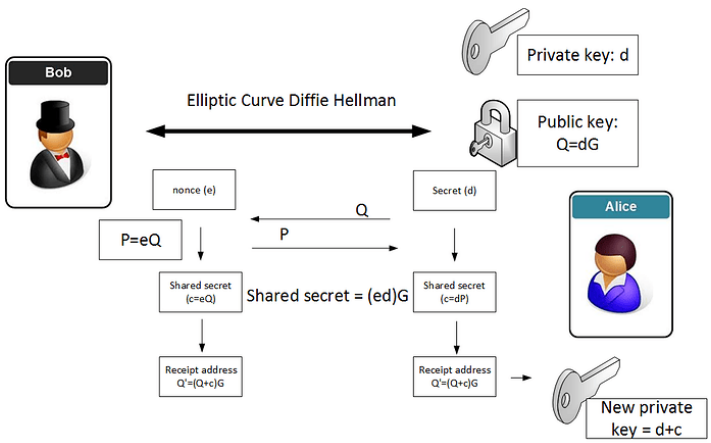
استراتژی های پرایوسی کوین ها (Privacy Coin)

پرایوسی کوین ها در تاکتیک هایی که برای محافظت از هویت کاربر استفاده می کنند متفاوت هستند ولی اغلب آنها ترکیبی از استراتژی های مختلف را پیاده سازی می کنند که رایج‌ترین آنها شامل چنین مواردی است: آدرس‌های مخفی، امضاهای حلقوی، کوین‌جوین و SNARK ها.

۱. آدرس های مخفی (Stealth Addresses)

بدون شک آدرس های مخفی ساده ترین راه برای بهبود حریم خصوصی تراکنش ها هستند. در ابتدایی ترین آنها، این فرآیند شامل ایجاد آدرس جدید در هر بار دریافت ارز رمزنگاری شده یا کوین می باشد. با این کار اطمینان حاصل می شود که طرفهای خارجی نمی توانند هیچگونه پرداختی و معامله ای در آینده را به آدرس کیف پول شما ربط دهند و آن را ردیابی کنند. با این حال برخی از ارزهای رمزنگاری شده دارای تاکتیک های پیچیده تری هستند.

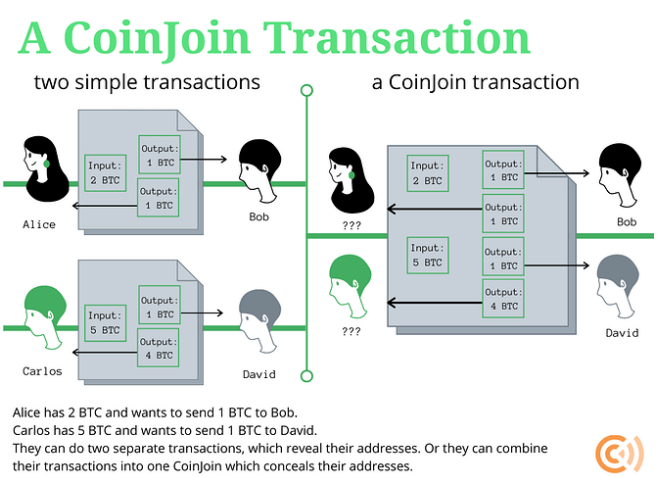
به عنوان مثال مونرو (XMR) یکی از پرایوسی کوین های برتر، نسخه‌ای از آدرس‌های مخفی را به کار می‌گیرد که پروتکل آدرس مخفی دو کلیده (Dual-key Stealth Address Protocol) یا به اختصار DKSAP خوانده می شود. این پروتکل به کاربرها یک کلید خصوصی مشاهده (private view key)، یک کلید خصوصی خرج کردن (private spend key) و یک آدرس گیرنده



(recipient address) اختصاص می‌دهد.
کاربر، ارسال تراکنش‌ها را با استفاده از کلید خصوصیِ خرج کردن انجام می‌دهد و برای مشاهده تراکنش‌های دریافتی از کلید خصوصیِ مشاهده استفاده می کند. با وجود اینکه آدرس دریافت به‌صورت عمومی منتشر می شود، دارایی‌های دریافتی به آدرس‌های جداگانه و متفاوتی روی بلاک چین ارسال می شوند و به این ترتیب فقط فرستنده و گیرنده‌های هر تراکنش هستند که آدرس دقیق مقصد را می‌دانند.

۲. کوین جوین (coinjoin)

کوین جوین به مخلوط کنِ کوین معروف است و تراکنش افراد مختلف را با یکدیگر تَرکیب و در قالب یک تراکنش واحد ادغام می کند و سپس آنها را با استفاده از آدرس‌های جدید میان کاربران مربوطه توزیع می کند. هر گیرنده سکه ها را در یک آدرس تازه (که هرگز استفاده نشده) دریافت می کند تا قابلیت ردیابی بیشتر کاهش یابد.
اگرچه CoinJoin برای اولین بار برای بهبود بیت کوین

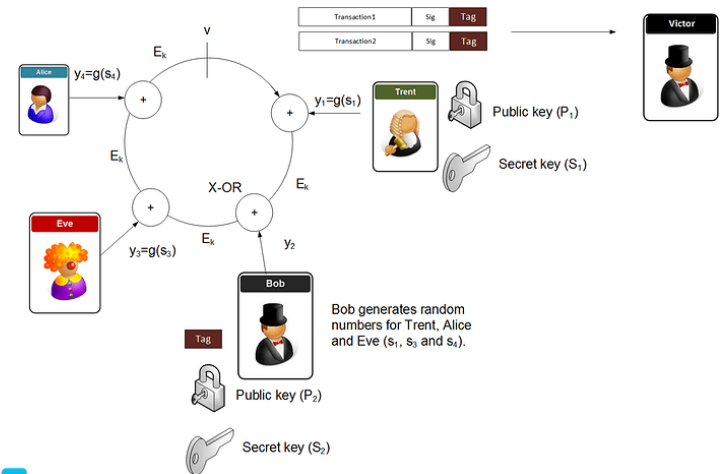


پیشنهاد شد، اما به عنوان بخشی از ویژگی ارسال خصوصی Dash ، مورد توجه قرار گرفت.

امضاهای حلقوی (Ring Signatures)

در یک تراکنش بلاک چین، یک امضای دیجیتالی ایجاد می کنید تا تأیید شود که خود شما فرستنده هستید و از آنجایی که شما تنها فردی هستید که این امضا را دارد، برای کسی سخت نیست که با ردیابی آن به شما برسد.
امضای حلقه فرستنده را با امضا کنندگان دیگر در یک حلقه ترکیب می کند تا شما را پنهان کند. هرچه تعداد کاربران اضافی در حلقه بیشتر باشد، یافتن فرستنده اصلی سخت تر است.
همچنین از بیرون حلقه راهی برای تشخیص معامله واقعی وجود ندارد از این رو سایر اعضا به عنوان ردگم کنی برای مخفی نگه داشتن فرستنده عمل می کنند. بالاترین تعداد ثبت شده از اعضای حلقه تا کنون ۱۰۰ نفر است.
مونرو نیز از امضای حلقه استفاده می کند. شبکه به طور خودکار گروهی از امضاکنندگان معامله را انتخاب می کند که با استفاده از خروجی تراکنش واقعی و همچنین ترکیبی از خروجی های معامله قبلی یک تراکنش ایجاد می کنند.

در گذشته مونرو باید به بخش های خاص تقسیم میشد. به عنوان مثال انتقال ۱۲. ۵ XMR به سه قسمت ۱۰ XMR و ۲ XMR و ۰. ۵ XMR تجزیه می گردید. سپس اعضای حلقه در صورتی که ارزش معاملاتی برابر این مقادیر می داشتند، انتخاب می شدند. تجزیه مبلغ معامله به این ترتیب این اطمینان را به وجود می آورد که فرستندگان به راحتی قادر به یافتن اعضای حلقه هستند زیرا امضای حلقه فقط می تواند معاملات با ارزش یکسان را گروه بندی کند.
با این حال این سیستم به این معنی است که کل مبلغ معامله بخشی از اطلاعات عمومی ای هست که هر کسی میتواند آن را ببیند. برای رفع این مشکل حلقه معاملات محرمانه (RingCTs) که پایه و اساس امضای حلقه را ایجاد می کند توسط مونرو در ژانویه ۲۰۱۷ ایجاد شد. RingCTs یک نوع ویژه از امضای حلقه است که علاوه بر مخفی کردن هویت شما، خروجی تراکنش را رمزگذاری می کند و مقدار آن را پنهان می کند.
با Ring CTs خروجی هایی با مقادیر مخفی تولید می شود و کیف پول دیجیتال آزاد است که اعضای حلقه را با هر ارزش معامله انتخاب کند و دیگر مبالغ معامله شده به اطلاع دیگران نمیرسد. با این وجود هنوز لازم است که شبکه بتواند اعتبار تراکنش را تایید کند. به همین دلیل مجموع ورودی معامله باید با مجموع خروجی آن برابر باشد.
فرستنده نیز موظف است که به مقدار خروجی متعهد شود که شامل افشای اطلاعات کافی برای ماینر ها در شبکه برای تایید اعتبار بدون اعلام عمومی مبلغ می شود. به این تعهد Pedersen می گویند و مفهوم آن اثبات صحت یک عبارت بدون آشکارسازی چیزی فراتر از حقیقت آن می باشد که Zk-SNARK نامیده می



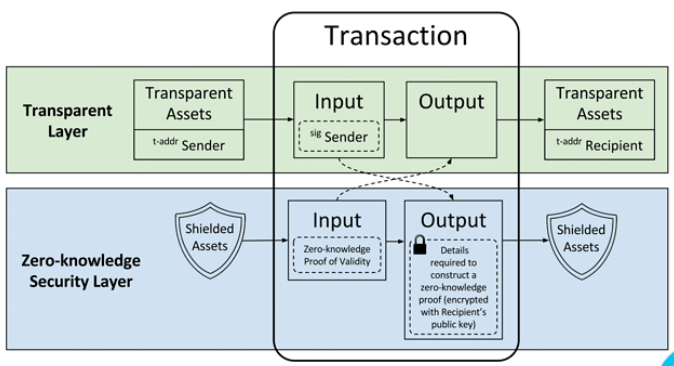
شود.

۴. zk-SNARKs

روش Zk-SNARKs مخفف عبارت Zero-Knowledge Succinct Non-Interactive Argument of Knowledge است و با استفاده از آن دارندگان ارزهای دیجیتال می‌توانند بدون نیاز به ارائه اطلاعات شناسایی حیاتی مانند هویت طرفین، تراکنش و موجودی حساب، اعتبار یک تراکنش را تأیید کنند.

به عبارت ساده zero-knowledge proof به این معنی است که بین دو طرف یک تراکنش، هر یک از طرفین می‌تواند بدون نیاز به افشا کردن خود اطلاعات، به دیگری اطمینان دهد که مجموعه خاصی از اطلاعات دارد. این موضوع کاملا با سیستم‌های اثبات دیگر که نیازمند دانستن تمام اطلاعات توسط یک سمت دارند، متفاوت است. یک مثال برای این مورد، پسورد ذخیره شده در یک سرور به صورت یک متن ساده است.

در این مورد اگر شما به یک سرور لاگین کنید، باید پسورد را داشته باشید و سرور نیز باید بررسی کند که آیا پسورد وارد شده توسط شما صحیح است یا نه. با استفاده از zero-knowledge proof، شخصی که به سرور لاگین می کند، باید توسط یک اثبات ریاضی نشان دهد که او رمز عبور درست را دارد، بدون آن



که آن را افشا کند.

نتیجه گیری:

رمزنگاری در واقع مجموعه‌ای از اقدامات و روش‌ها است که برای ایمن ساختن ارتباطات میان افراد طراحی شده است. در نتیجه پرایوسی کوین ها (Privacy Coin) یا همان ارزهای دیجیتال حریم خصوصی محور، علی‌رغم اینکه ماهیت غیرقابل‌ردیابی‌شان بسیار مناقشه‌برانگیز است و می‌توانند مورد استفاده مجرمین مالی قرار بگیرند، اما بخشی اساسی از یک اکوسیستم رمزنگاری شده به شمار می‌آیند.

منابع:

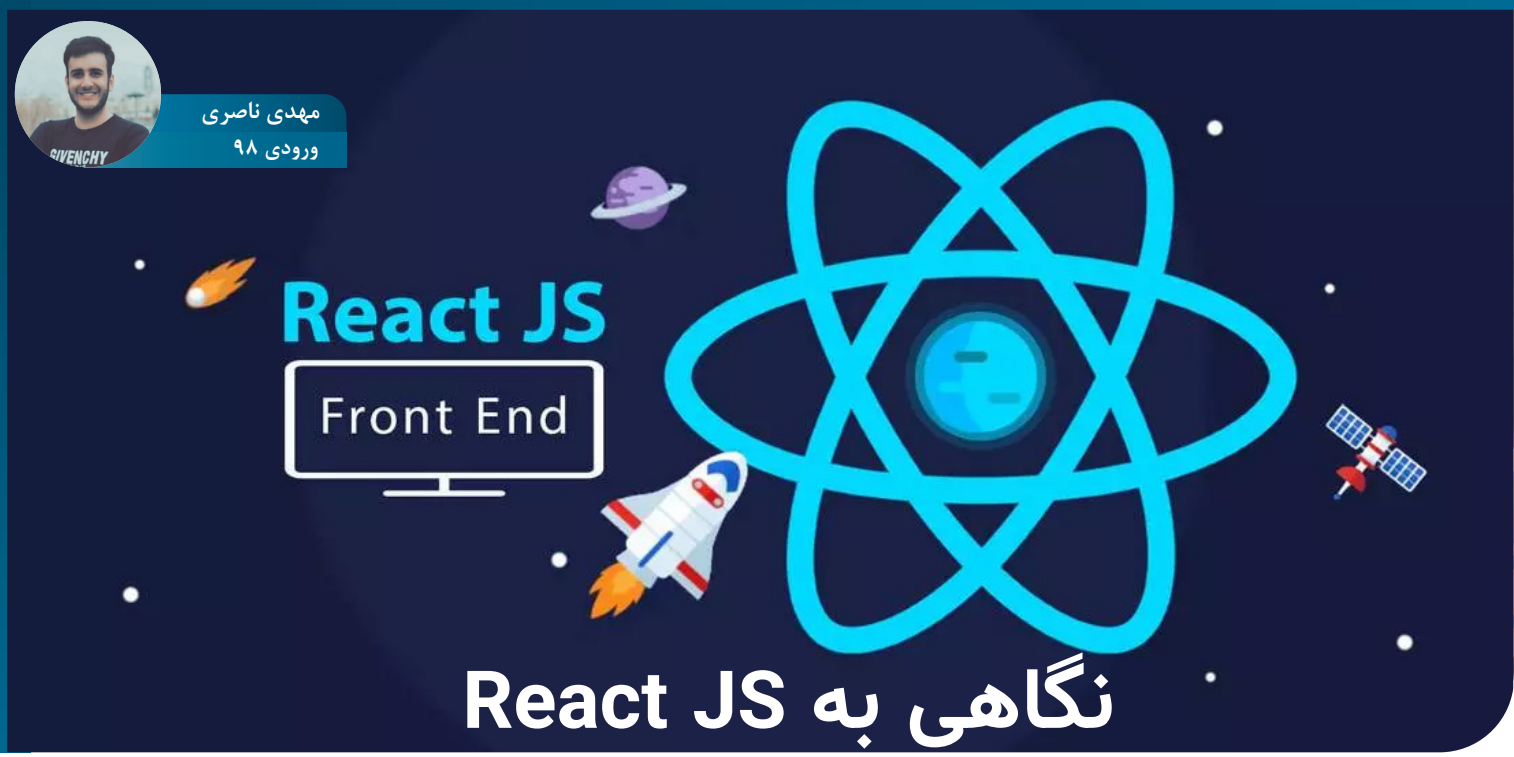
https://ledgerops.com/blog/what-are-privacy-coins-and-how-do-they-work۲۰۱۹-۱۶-۰۵-/

https://coinmarketcap.com/alexandria/article/what-are-privacy-coins

https://www.getmonero.org/resources/moneropedia/stealthaddress.html

https://beincrypto.com/learn/how-do-privacy-coins-work/

https://www.getmonero.org/resources/moneropedia/ringsignatures.html



نگاهی به React JS

مهدی ناصری
ورودی ۹۸

REACT چیست؟

React یک کتابخانه برای زبان جاوااسکریپت است که توسط فیسبوک در سال ۲۰۱۳ به صورت متن باز منتشر شده است. از این کتابخانه برای ساخت رابط های کاربری پویا برای وب و موبایل اپلیکیشن ها استفاده می شود.

مهم ترین ویژگی های React چیست؟

• Virtual DOM: احتمالا اگر با جاوا اسکریپت آشنایی داشته باشید اسم DOM را شنیده اید. DOM یک ساختار به شکل درخت است که وقتی صفحه سایت ما در مرورگر بارگذاری می شود مرورگر آن صفحه را به شکل این درخت مدل سازی می کند تا راحت تر بتواند صفحه را مدیریت کند. حال اگر فرض کنیم ما یک المان لیستی داریم که شامل چند آیتم می باشد اگر ما روی یکی یا دوتا از این آیتم ها تغییراتی ایجاد کنیم، تمام آیتم های لیست نیز مجددا در DOM بروزرسانی می شوند حتی آنهایی که تغییراتی در آنها صورت نگرفته است. این محدودیت در وبسایت های بزرگ سبب می شود تا یک تغییرکوچک در یک المانی در سایت سبب کندی سایت شود زیرا مرورگر باید تمام المان های فرزند آن المان که تغییر کرده را نیز مجددا در DOM بروزرسانی کند. کتابخانه React با استفاده از Virtual DOM این محدودیت را به خوبی رفع کرده است. در React هر DOM یک Virtual DOM متناظر با خود را دارد که به نوعی کپی ایی از آن DOM است. حال اگر ما همان المان لیست را داشته باشیم و روی چند تا از آیتم ها تغییراتی ایجاد کنیم یک Virtual DOM جدید ساخته می شود، حال React این Virtual DOM جدید را با قبلی مقایسه می کند و می فهمد که کدام آیتم های لیست در Virtual DOM تغییر کرده است، سپس React در DOM اصلی تنها آیتم هایی را بروزرسانی می کند که تغییر کرده اند نه تمام آیتم های آن لیست را. این کار سبب می شود تا اپلیکشن ما بسیار تندتر به تغییرات واکنش نشان دهد.

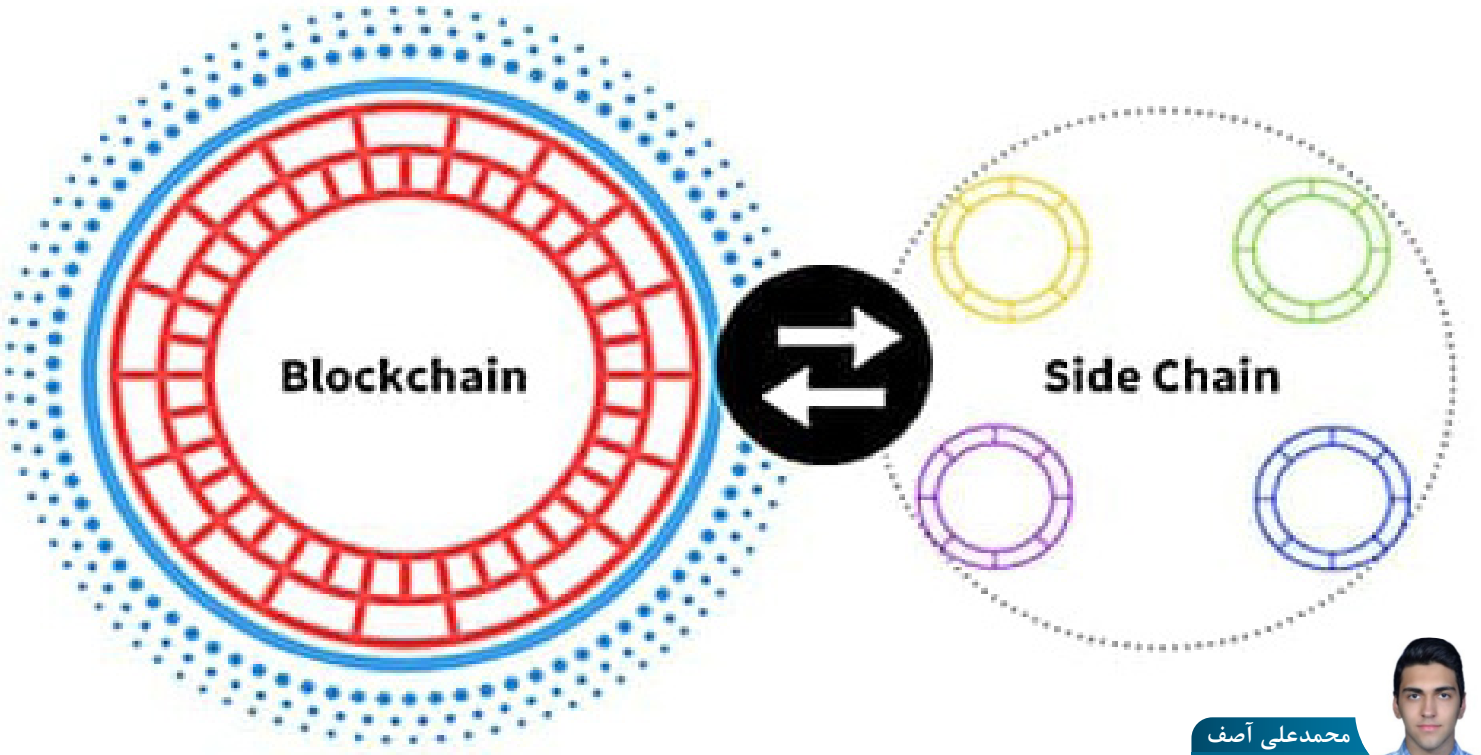
• **Component Based**: کتابخانه React به صورت Component based کار می کند. یعنی به این صورت که یک رابط کابری بزرگ را به بخش های کوچک تری به اسمcomponent تقسیم می کنیم و هر کدام از این بخش ها کدهای مخصوص به خود را دارند و در آخر در جاهای مناسب این کامپوننت ها را با هم ادغام می کنیم و آن رابط کاربری بزرگ را تشکیل می دهیم (همان از جز به کل!). برای مثال در اینستاگرام می توان گفت که بخش استوری ها یک کامپوننت می باشد، بخش پست ها یک کامپوننت می باشد، پنجره دایرکت ها یک کامپوننت می باشد که هر کدام از این کامپوننت ها در یک فضا در تعامل هستند اما هرکدام ساختار و متد های مخصوص به خود را دارند.

• **JSX**: JSX مخفف JavaScript XML است که یک اکستنشن(extension) برای جاوا اسکریپت است که به وسیله آن می توانیم المان های html را در کامپوننت های React همان گونه که در html بودند بنویسیم و سپس React آن را به المان های html تبدیل می کند و دیگر ما را از استفاده متد createElement() برای ساخت المان های html در React راحت می کند. دقت شود که JSX یک عبارت(expression) در جاوا اسکریپت است و می تواند مقدار یک متغیر باشد و یا به عنوان خروجی یک تابع در نظر گرفته شود.

نتیجه گیری:

این ویژگی های بالا تنها بخش کوچکی از ویژگی های مهم React است. کافی است وارد این حوزه شویم و ببینیم که با استفاده از امکانات React می توانیم رابط های کاربری قدرتمند و پویایی برای وب و موبایل اپلیکیشن ها بسازیم، و پی ببریم که چرا روز به روزتقاضاها برای استفاده از React در پروژه ها بیش از پیش می شود. پس اگر در زمینه ساخت وب و موبایل اپلیکشن ها علاقه داری شک نکن که React یکی از گزینه های جذاب و قدرتمند برای کار در این حوزه است.

منابع: www. medium. com, www. codecademy. com



محمدعلی آصف
ورودی ۹۸

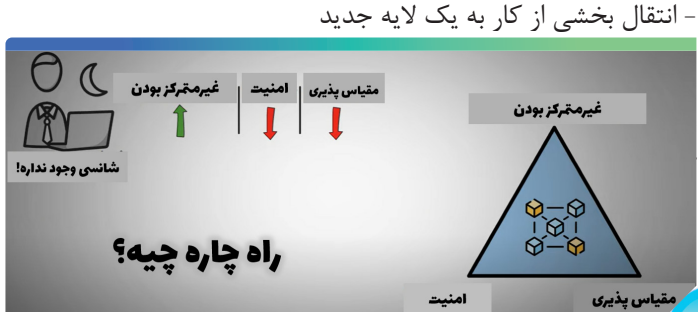
سایدچین (Sidechain) یا زنجیره جانبی به زبان ساده

سایدچین یکی از روش های پرکاربرد و مورد توجه برای بهبود مقیاس پذیری و توسعه پذیری (Scalability) بیشتر شبکه های بلاکچین هستند. در این مقاله به روش های کلی مقیاس پذیری و سپس جایگاه، چیستی، تاریخچه، کاربردها، ویژگی ها و پروژه های سایدچین ها می پردازم.

راه حل های لایه دوم چه مشکلی را حل می کنند؟

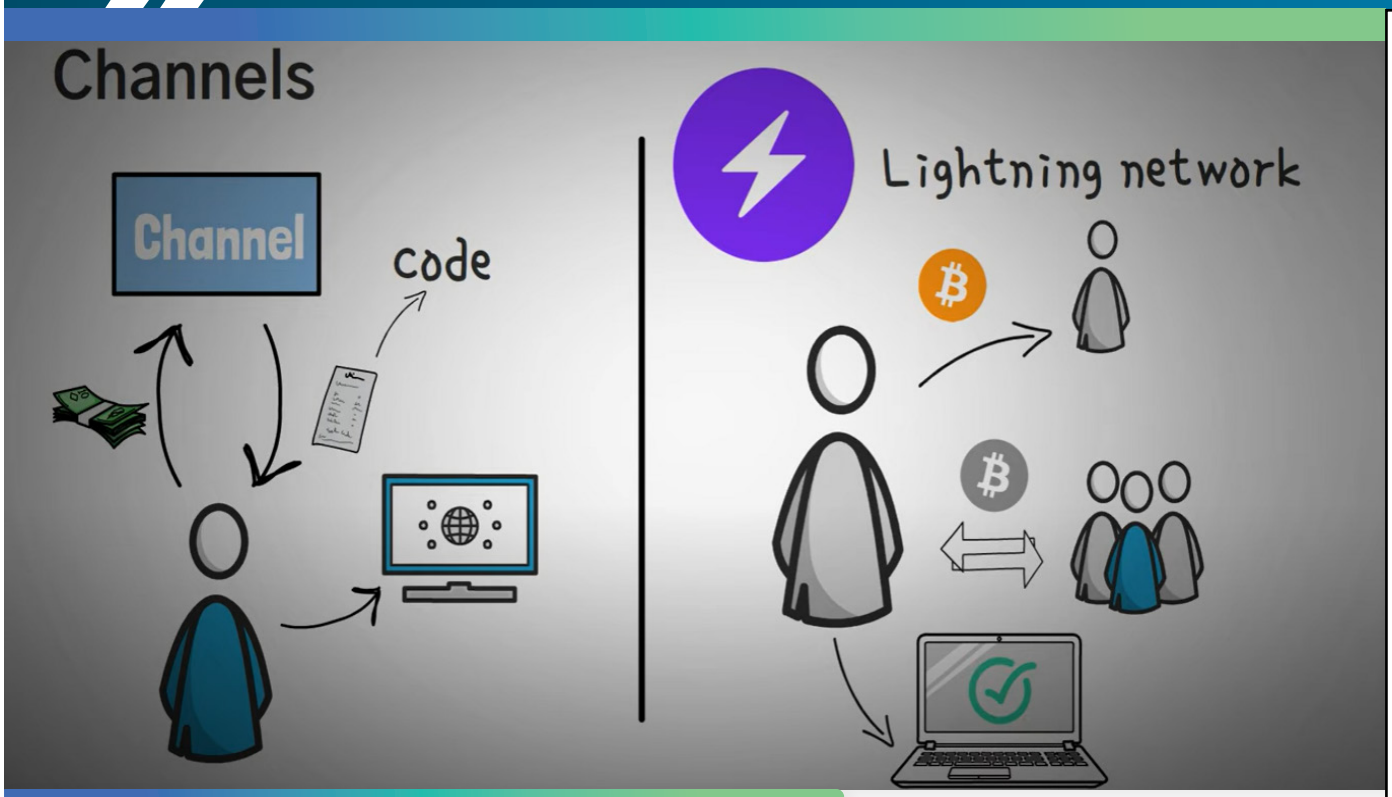
در شبکه های بلاکچین چون بیتکوین و اتریوم به طور معمول بین ۷ تا ۱۵ تراکنش در ثانیه انجام می شود، در حالی که شبکه ویزا (Visa) حدود ۱۰۰،۰۰۰ تراکنش در ثانیه انجام میدهد. بلاکچین برای رقابت با روش های متمرکز مثل ویزا و جهان روا شدن نیاز به انجام تراکنش های بیشتری در هر ثانیه دارد. به طور کلی دو روش برای مقیاس پذیری وجود دارد:

- تغییر در لایه اصلی (Base layer)
- انتقال بخشی از کار به یک لایه جدید



خصوصی را دنبال می کند. دلیل نام گذاری این روش ها به لایه دوم (Second Layer) این است که معمولا تاثیر مستقیمی بر کد بلاکچین اصلی ندارند اما باعث مقیاس پذیری بصورت بیرونی می شوند. در ادامه به اختصار به هریک اشاره میکنم:

- Rollups: به دو مدل zkrollups که قابلیت استفاده از قراردادهای هوشمند را ندارند اما سریع وکارا تر هستند و optimistic rollups که قابلیت اجرای قراردادهوشمند دارند اما کند تر ناموثر تر هستند تقسیم می شوند. به صورت خیلی خلاصه در این روش چند تراکنش در یک تراکنش جمع می شود و به شبکه اصلی ارسال می شود.
- Sidechain: سایدچین ها بصورت موازی در کنار زنجیره اصلی کار می کنند و از منابع خود و اطلاعات زنجیره اصلی برای هندل کار استفاده می کنند و در نهایت دیتا را به زنجیره اصلی ارسال می کنند. چین اصلی یا والد نیازی به سایدچین ندارد اما سایدچین نیاز به چین والد (Parent Chain) دارد. از معروف ترین سایدچین های اتریوم میتوان Polygan را نام برد.
- Channels: چنل ها راهی برای قفل کردن دارایی اصلی و استفاده از نسخه مجازی دارایی اصلی در شبکه ای با سرعت بسیار بیشتر میباشند. در کارت های بانکی نیز دلار واقعی در لحظه جابجا نمی شود، بلکه صرفا عدد و کدی که به معنای داشتن داریی است جا به جا می شود و از آنجایی که همه قبول دارند که یک دلار مجازی برابر یک دلار واقعی است، مشکلی پیش نمی آید. Lightning Network بیتکوین مثال مطرح این روش است. این روش متخص انجام تراکنش است نه قرارداد



هوشمند.

سایدچین ها:

تعریف سایت هکرمون:

سایدچین یک بخش جدا از بلاکچین است که با یک ارتباط دوطرفه به یکدیگر متصل می شوند. در این ارتباط دوطرفه قابلیت تعویض دارایی با نرخ از پیش تعیین شده وجود دارد که بین سایدچین و بلاک چین انجام می شود. همچنین این احتمال نیز وجود دارد که این مبادلات به صورت برعکس انجام شود.

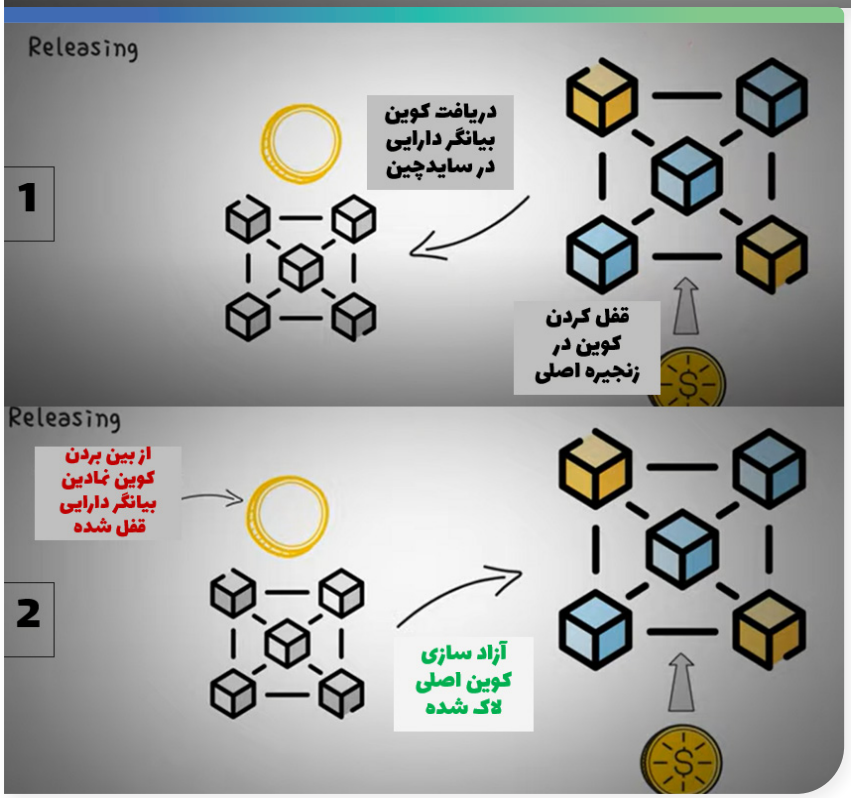
تاریخچه:

این ایده اولین بار در سال ۲۰۱۴ شکل گرفت یعنی زمانی که چندین چهره برجسته در دنیای بلاکچین و اولین ایده پردازان آن یک مقاله منتشر کردند و در آن سایدچین های تثبیت شده (Pegged Sidechains) را معرفی کردند. چندی از این نویسندگان از افراد اصلی بلاک استریم هستند که در زمینه ایده های خلاقانه در سایدچین و سایر توسعه های بیت کوین پیش قدم بوده اند.

ساختار و کارکرد:

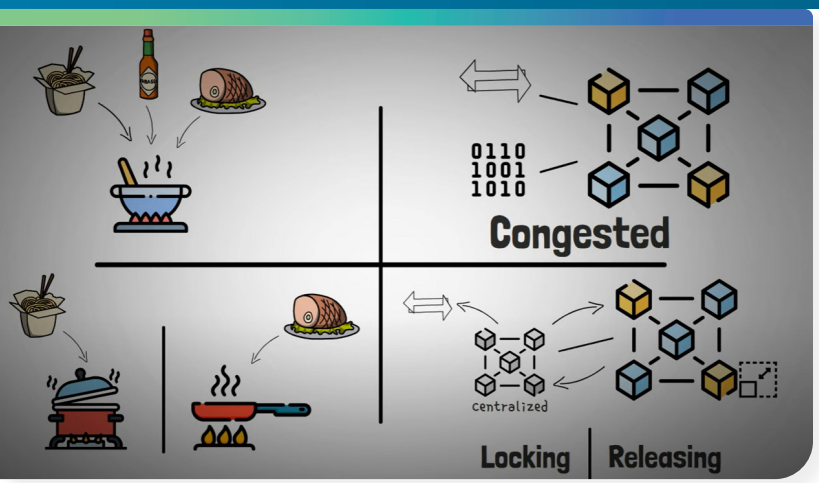
زنجیره جانبی بلاکچینی با استفاده از یک پل ۲ طرفه به نام "peg" به بلاکچین اصلی آن متصل می شود. دلیل نام گذاری «دوطرفه» این است که peg ای از چین اصلی به سایدچین وجود دارد که عملیات آن Locking up و از سایدچین به برون وجوددارد که عملیات آن Releasing نام دارد.

Locking up: بعد از انتقال کوین از مِین چین به سایدچین باید کوین ها قفل شوند، در غیر این صورت توکن های ازادی در هر دو چین وجود دارد. عملیات Lock up شدن در واقع به معنای



انتقال کوین ها به ولت یا قرارداد هوشمندی مشخص است و توسط کد کنترل می شوند نه یک انسان و بازپس گیری آنها نیاز به عملیات خاصی دارد.

Releasing: بعد از قفل کردن کوین در مِین چین، پروتکل آنها را قبول می کند و کوین هایی در سایدچین میدهد که بیانگر آن دارایی که در مین چین قفل شده اند میباشند. برای بازپس گیری کوین لاک شده در مین چین، کوین هایی که بیانگر کوین های لاک شده اند را باید از بین ببریم تا کوین های اصلی آزاد



شوند.

درواقع پلی که ارتباط میان بلاکچین اصلی و زنجیره های جانبی را برقرار می کند قابلیت تعویض سرمایه ها را با نرخ از پیش تعیین شده میان بلاکچین اصلی و زنجیره جانبی فراهم می نماید. کاربر در زنجیره اصلی اول باید کوین های خود را به یک آدرس ارسال کند، جایی که کوین ها قفل می شوند تا کاربر قادر به صرف آنها در جای دیگر نباشد. بعد از انجام تراکنش، تایید از طریق زنجیره ها و بعد از یک دوره انتظار که Confirmation Period نام دارد، معادل تعداد کوین ها در ساید چین آزاد می شود و به کاربران این امکان را می دهد به آنها دسترسی داشته باشد. البته برای جلوگیری از Double spend یک بازه زمانی دیگر بنام Contest Period کاربر اجازه استفاده از توکن ریلیز شده را ندارد.

Federation: به گروهی گفته می شود که به عنوان یک نقطه میانی بین یک زنجیره اصلی و سایدچین آن عمل می کند. البته همه سایدچین ها به فدریشن نیاز ندارند ولی اکثرا به دلیل کاربرد خوب استفاده می کنند. فدریشن میتواند بصورت کد باشد و سازندگان سایدچین اعضای این گروه را انتخاب کنند. آنها تعیین می کنند چه زمانی کوین هایی که کاربر استفاده کرده قفل شده و یا آزاد شوند و دقت می کنند تا در روند لاک و ریلیز اشتباهی رخ ندهد. وجود فدریشن یکی از دلایلی است که باعث متمرکز شدن شبکه می شود.

امنیت:

به طور کلی سایدچین ها مسئول امنیت خودشان هستند. اگر قدرت پردازشی کافی برای تامین امنیت یک زنجیره جانبی وجود نداشته باشد، می توان آن را هک کرد، اما از آنجا که هر زنجیره مستقل می باشد، به خطر بیافتد بر زنجیره اصلی تاثیر نمی گذارد. سایدچین ها میتوانند تایید کننده/ماینر و حتی الگوریتم اثبات اختصاصی خود نظر اثبات کار (Poof of Work)، اثبات سهام (Proof of Stake)، اثبات زمان و فضای ذخیرسازی (Proof of Space and Time) و اثبات اعتبار (Proof of Authority) را میتوانند داشته باشند. اکثر سایدچین ها مقداری متمرکز تر (Centrlized) نسبت به زنجیره اصلی میباشند اما در عوض از سرعت بیشتری برخوردارند.

برخی ویژگی های سایدچین ها

- بهبود مقیاس پذیری
- انعطاف پذیری
- مرج ماینینگ: مسئله جالب دیگری که بوسیله سایدچین ها ایجاد می شود Merge Mining است. بدین صورت که ماینر میتواند همزمان چند رمزارز چند بلاکچین را باهم استخراج کند.

برخی پلتفرم ها و پروژه های مبتنی بر سایدچین:

- روتاستاک Rootstock(RSK) : یک سایدچین ایجاد شده برای بیتکوین است که برای ایجاد توانایی تولید و اجرای قراردادهای هوشمند پیچیده بیت کوین طراحی شده است. فدریشن این ساید چین ۲۵ صرافی بزرگ هستند که پل نقل و انتقال بیتکوین و بیتکوین روت استاک را نظارت می کنند.
- POA : شبکه POA یک سایدچین باز اتریوم همراه با توافق اثبات اعتبار است که توسط تاییدکنندگان اعتبار مستقل محقق می شود. ما یک شبکه عمومی برای قراردادهای هوشمندی می‌سازیم که سرعت، امنیت و کارایی هزینه را ترکیب می کند. توکن چیلیز (Chiliz) از پروژه های معروف استفاده کننده از Po می باشد.
- Lisk: یک پلتفرم سایدچین است که به توسعه دهندگان امکان پیاده سازی DApp ها در javascript را می دهد. Lisk پلتفرمی است.
- Matic: پالیگان سایدچینی روی بلاکچین اتریوم است که تعداد تراکنش برابر با آن دارد اما فی بسیار پایین تری دارد. و هر بلاک آن بجای ۱۰ ثانیه در ۲ ثانیه ایجاد می شود.

جمع بندی:

سایدچین مفهومی مهم و بسیار کاربردی در دنیای بلاکچین است و پروژه های بسیاری با استفاده از این روش پدید آمده اند و به بهبود کارایی زنجیره های اصلی پرداخته اند. تفاوت عمده آن با روش های دیگر لایه ۲ مثل چنل ها و رُل آپ ها این است که راه حلی دائمی و غیرقابل تغییر است اما روش های دیگر با سرعت و راحتی بیشتری تغییر می کنند. در این مقاله سعی داشتم در ابتدا به موضوع روش های مقیاس پذیری بپردازم و سپس نگاه دقیق تر به سایدچین ها با طبقه بندی، خلاصه کردن و ساده سازی توضیحات منابع مختلف داشته باشم. امیدوارم براتون مفید بوده باشه.

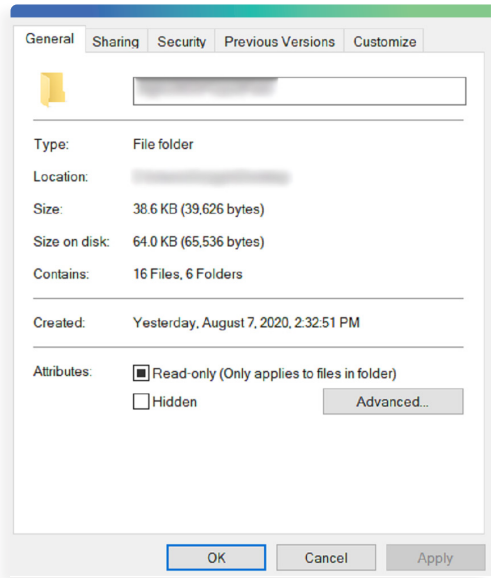
منابع:

https://hackernoon. com/what-are-sidechains-1c45ea2daf3
https://blockonomi. com/sidechains/
https://docs. ethhub. io/ethereum-roadmap/layer-2scaling/sidechains/
https://www. youtube. com/watch?v=cFRj-2jzm8E
https://www. youtube. com/watch?v=9pJjtEeq-N4
https://hackernoon. com/-13sidechain-projects-every-blockchain-developer-should-know-about-804b65364107

فضاهای ذخیره‌سازی اطلاعات

دانیال فرهنگی
ورودی ۹۸

در یک سری مقاله قصد داریم در رابطه با فضاهای ذخیره سازی اطلاعات بیشتری کسب کنیم. در قسمت اول از این سری قصد داریم بدانیم چرا فضای ذخیره سازی برای ما اهمیت دارد؟
با گذشت زمان و پیشرفت در زمینه های مختلف تکنولوژی نیاز بشر به اطلاعات و در نتیجه ذخیره سازی آن بیشتر و بیشتر شده و از این جهت نحوه ذخیره سازی آن ها و نوع نگهداری آنها نیز اهمیت می یابد. در بعد نرم افزاری اگر بخواهیم بحث کنیم می توانیم مباحث پایگاه های داده را بررسی کنیم اما در بعد سخت افزاری مهم ترین بخش فضاهای ذخیره سازی هستند که قصد داریم در این سری بیشتر درمورد آنها صحبت کنیم.
واحد های ذخیره سازی در بعد سخت افزاری و نرم افزاری:
اگر از ویندوز استفاده میکنید حتما تا به حال این صفحه را دیده اید.



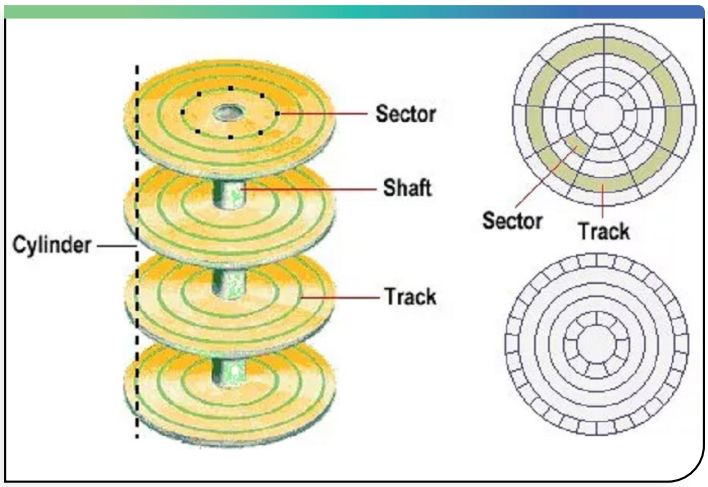
این صفحه ای است که وقتی از فایل یا فولدر در ویندوز منوی Properties را انتخاب میکنید با این صفحه مواجه خواهید شد.
نکته ی قابل توجه دراین شکل تفاوت قسمت size و قسمت size on disk می باشد. چرا این دو قسمت دو عدد متفاوت را نشان میدهند؟ تفاوت این دو ناشی از چیست ؟ برای پاسخ به این سوال باید اطلاعات بیشتری راجع به ذخیره سازی کسب کنیم.

سکتور چیست؟

سکتور در واقع سایز هر یک از تقسیمات موجود در سخت افزار ذخیره سازی است. به عبارت دیگر همانطور که در بعد نرم افزار کوچکترین واحد ذخیره سازی بیت است در بعد سخت افزار کوچکترین واحد ذخیره سازی سکتور می باشد. در واقع ذخیره سازی به این صورت انجام میگیرد که اطلاعات در سکتور یا سکتور ها نوشته می شوند و اطلاعات این سکتور های ابتدایی و انتهایی در متادیتای فایل نوشته می شود. اگر اطلاعات به گونه ای باشند که یک سکتور در آخر به صورت کامل اطلاعات درون خود نداشته باشد، اطلاعات جدید برای ذخیره سازی در سکتور بعد نوشته می شوند و در این صورت است که اتلاف فضای ذخیره سازی داریم. در این رابطه بعدا بیشتر صحبت خواهیم کرد.

به طور معمول یک سکتور از هارد دیسک و یا فلش درایو ها میتوانند ۵۱۲بایت از اطلاعات را در خود نگه دارند. این استاندارد در سال ۱۹۵۶ پایه گذاری شد. در سال ۱۹۷۰ با به میان آمدن دستگاه های ذخیره سازی بزرگتر سکتور هایی با سایز ۱۰۲۴ بایت و ۲۰۴۸ بایت نیز معرفی شد ند .
در شکل زیر میتوان یک سکتور و قسمت های دیگر فضا های ذخیره سازی را مشاهده کرد.

پیوند بین سخت افزار و نرم افزار:



همانطور که گفته شد سکتور کوچکترین واحد ذخیره سازی در بعد سخت افزاری و بیت کوچکترین واحد ذخیره سازی در بعد نرم افزاری است. اما این دو چه ارتباطی به هم دارند؟ به عبارت دیگر هر سکتور چند بیت را در خود ذخیره می کند؟ یا هر بیت در چند سکتور ذخیره می شود؟
این موضوع در واقع در هنگام فرمت کردن دیسک ها مشخص می شود . در هنگام فرمت کردن دیسک ها قسمتی وجود دارد به نام allocation unit size یا به اختصار AUS که رابطه ی بین سکتور و بیت ها را مشخص می کند. برای این منظور پروتکل های بین المللی وجود دارد که ابتدایی ترین آنها به این صورت بود که هر سکتور حاوی ۵۱۲ بایت از اطلاعات باشد. این استاندارد در قسمت های دیگری نیز به کار رفته مثلا اگر اسم MBR را شنیده باشید حتما میدانید که MBR نیز ۵۱۲ بایت یا به طور دقیق تر یک سکتور از اول هارد را برای ذخیره سازی اطلاعات نگه می دارد :) . پس در نتیجه می توان گفت که این AUS است که رابطه ی بین سکتور و بیت را مشخص می کند.
حال با توجه به صحبت های انجام شده میتوان دلیل تفاوت بین size و size on disk را درشکل بالا متوجه شد. دلیل این است که با توجه به نوع فرمت دیسک و اندازه AUS، سایز خود اطلاعات ما همانی است که در قسمت size نوشته شده اما چون اطلاعات به اندازه ای نبوده اند که سکتور آخر به صورت کامل پر شود پس قسمتی از سکتور آخر خالی می ماند ولی اطلاعات جدید دیگر در این قسمت نوشته نخواهد شد و از سکتور جدید شروع خواهد شد.
با ما در قسمت های بعدی از این سری همراه باشید.

منابع:

https://tosinso. com/ZE۵
https://chipiran. com

مهران آدووند
ورودی ۹۸

بررسی رفتار رانندگی با استفاده از Big Data در بیمه هوشمند

افزایش تلفات ناشی از تصادفات سبب افزایش دریافت خسارت توسط بیمه شوندگان از شرکت های بیمه می شود، بنابراین شرکت های بیمه به دنبال راه های دقیق و قابل اعتماد هستند تا بتوانند رانندگان پر خطر را شناسایی کنند و حق بیمه بیشتر از آنها دریافت کنند و از طرف دیگر خودروهایی که در طول سال مدت زمان بیشتری را در جاده ها سپری می کنند باید پول بیشتری نسبت به سایر خودرو ها به بیمه پرداخت کنند چراکه در معرض خطر بیشتری قرار می گیرند.

برای بررسی رفتار رانندگان و تخصیص حق بیمه عادلانه میان رانندگان پر خطر و رانندگان با احتیاط ، اطلاعاتی که توسط سنسور های خودرو و دستگاه های ردیاب جمع آوری شده است مورد تجزیه و تحلیل قرار می گیرد . در بین این دو منبع بیشتر سعی می شود از اطلاعات ردیاب ها استفاده شوند چون همه ی خودرو ها مجهز به سنسور نیستند. ردیاب ها از هنگام شروع حرکت اتومبیل اطلاعات مختلف نظیر موقعیت جغرافیایی ، سرعت و شتاب اتومبیل را در هر لحظه به سرور ارسال می کنند ولی این حجم از اطلاعات را هیچ دستگاهی نمی توانند ذخیره سازی و تحلیل کند که در اینجا علم **Big data** توانایی انجام این کار را فراهم می کند و با استفاده از الگوریتم های مختلف از پارامترهای فرستاده شده توسط ردیاب نظیر شدت شتاب ، شدت سرعت ، شدت ترمز و شدت پیچیدن، پرخطر و یا محتاط بودن رانندگان را مشخص می کنند بر اساس آن حق بیمه هر راننده محاسبه می شود.

رشد چشمگیر تولید، ارسال، ذخیره سازی و تجزیه و تحلیل اطلاعات توسط علم Big data باعث شد صنایع مختلف نگاهی ویژه تر به این علم داشته باشند و سرمایه گذاری کلانی در

این حوزه انجام دهند مانند شرکت های بیمه در سراسر جهان که با سرمایه گذاری در این زمینه، هوشمند سازی بیمه ها را کلید زدند.

علم Big data اطلاعات زیاد و خام را به واقعیت نزدیکتر می کند و توسط الگوریتم های پیچیده آنها را تحلیل و به شکل نمودار های مختلف آنها را گزارش می کند و ویژگی بصری اطلاعات را تقویت می کند این امر باعث شد تا شرکت های بیمه پیشگام در استفاده از آن باشند به طوری که طبق گزارش IBM، حدود ۷۵ درصد از شرکت های بیمه در سراسر دنیا سالیانه حجم زیادی از اطلاعات بیمه شوندگان را توسط Big data تجزیه و تحلیل می کنند و این عدد در مقایسه با صنایع دیگر بسیار بیشتر و قابل توجه می باشد.

دلیل اصلی استفاده شرکت های بیمه از علم Big data، تجزیه و تحلیل اطلاعات گذشته برای پیش بینی حوادث آینده، محاسبه خسارات احتمالی ناشی از آنها و حق بیمه ای که باید در قبال آن پرداخت شود، است. در زمینه بیمه خودرو بررسی اطلاعات به منظور کشف نحوی رانندگی هر نفر است به طوری با توجه به سرعت، شدت شتاب، شدت ترمز، شدت چرخش خودرو و رعایت قوانین راهنمایی و رانندگی، مشخص می شود که یک راننده رفتار پرخطر دارد و یا با رعایت اصول رانندگی می کند و در نتیجه این امر موجب پرداخت بیشتر حق بیمه توسط راننده پرخطر می شود و همچنین هر بیمه شونده می تواند تحلیل کامل و درستی از وضعیت رانندگی خود در هر زمان را داشته باشد.

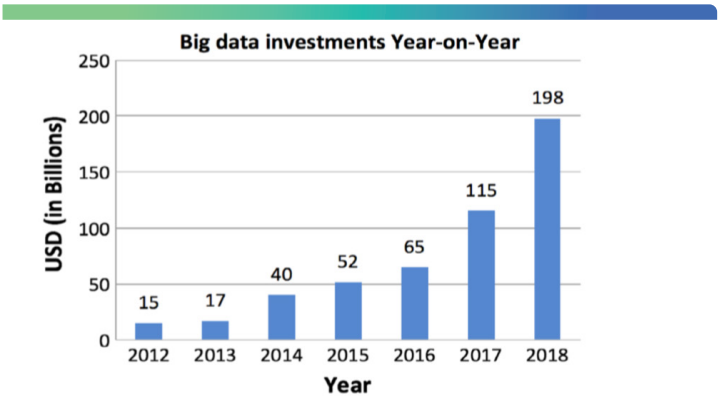
در حال حاضر تحلیل اطلاعات در شرکت های بیمه ی خودرویی به سه شکل زیر انجام می شود:

• پرداخت حق بیمه باتوجه به مسافت طی شده سالیانه (PAYD):

در این روش حق بیمه با توجه به مسافتی که خودرو در سال طی می کند محاسبه می شود بدون توجه به نحوی رانندگی فرد.

• پرداخت حق بیمه باتوجه به نحوی رانندگی (PHYD):

در این نوع، نحوی رانندگی فرد در هر سفر بر اساس پارامتر های ذکرشده محاسبه می شود و امتیازی به آن راننده داده می شود و مجموع این امتیاز ها نحوی رانندگی فرد را مشخص می کند و بر اساس آن حق بیمه متفاوت از افراد گرفته می شود.



میزان سرمایه گذاری در علم ابر داده در آمریکا

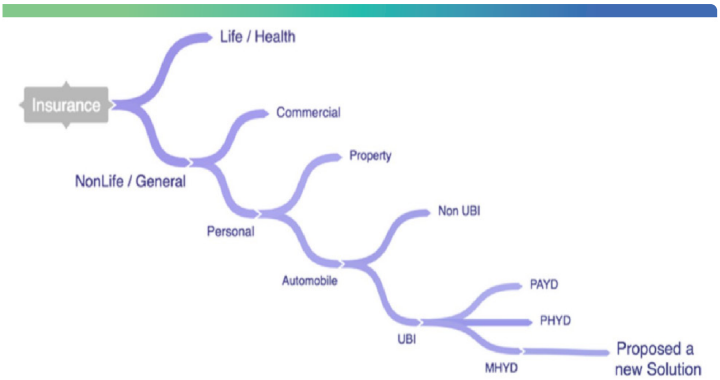


Fig. 5 Classification of UBI

• پرداخت حق بیمه بر اساس مدیریت چگونگی رانندگی (MHYD):

در این نوع علاوه بر تعیین رفتار راننده ، با استفاده از سنسور های خودرو و تلفن همراه هوشمند اطلاعاتی جهتی بررسی میزان حواس پرتی و خواب آلودگی راننده جمع آوری و به وسلیه big data تحلیل می شود و اگرراننده رفتار خطرناکی در حین رانندگی انجام دهد به صورت آنلاین به او هشدار داده می شود تا نحوی رانندگی خود را اصلاح و به شیوه ی امن تری رانندگی کند.

در میان این سه روش استفاده از بیمه هوشند بر اساس Big data، دو مورد اول در کشور های مختلف پیاده سازی شده است ولی در حال حاضر سرمایه گذاری و تلاش برای پیاده سازی مورد سوم در شرکت های بیمه است چون صحت اطلاعات تحلیل شده برای شرکت های بیمه بسیار مهم و حیاتی می باشد.

منبع:

• Subramanian Arumugam and R. Bhargavi :A survey on driving behavior analysis in usage based insurance using big data

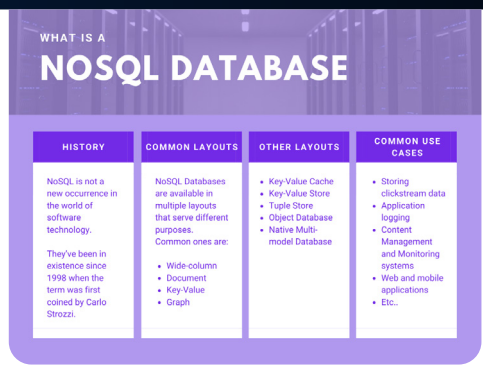
عصر جدیدی از پایگاه داده‌ها برای تجزیه و تحلیل BigData - طبقه‌بندی، ویژگی‌ها و مقایسه

فائزه خادم
ورودی ۹۸

در برنامه نویسی سنتی، پایگاه‌های داده معمولا از نوع SQL هستند که یک پایگاه داده رابطه ای یا Relational است. مدل رابطه‌ای پایگاه‌های داده در دهه ۱۹۷۰ میلادی معرفی شد و روشی منطبق بر ریاضیات برای سازمان‌دهی، نگهداری و استفاده از داده‌ها ارائه کرد.

حاصل دهه‌ها تحقیق و توسعه، سیستم‌های پایگاه داده‌ای که مدل رابطه‌ای را پیاده‌سازی می‌کردند به شدت کارآمد و پایدار بودند. پایگاه‌های داده رابطه‌ای علی‌رغم ماهیت صریحشان در شکل‌دهی و مدیریت داده‌ها می‌توانند بسیار انعطاف‌پذیر باشند و با مقدار تلاش اندکی نتایج زیادی را ارائه کنند. اما این نوع از پایگاه‌های داده یک مشکل بزرگ دارند. این مشکل زمانی خود را نشان داد که غول‌های نرم افزاری دنیا مثل گوگل، آمازون و فیسبوک احتیاج به تحلیلِ داده‌های با حجم و تعداد بالا یا همان Big Data پیدا کردند.

پایگاه‌های داده رابطه ای به دلیل نوع ساختار خود، برای تحلیل داده‌های بزرگ



غیر بهینه، ناکارا و همینطور کند بودند. البته در بعضی موارد هم استفاده از ساختار جدولی که در پایگاه‌های داده رابطه ای استفاده می شود تقریبا ناممکن بود. به همین دلیل ذخیره سازی حجم زیادی از داده‌های بی ساختار (Non-structured Data) سرعت و کارایی این پایگاه‌های داده را به شدت کاهش می‌داد. تا اینکه پایگاه‌های داده NoSQL پا به عرصه گذاشتند.

این سیستم‌های پایگاه داده از طریقِ کنار گذاشتن سبک سازماندهی کاملاً منظم داده‌ها در مدل رابطه‌ای، شروع به ارائه روشی برای کار بسیار آزادانه‌تر با اطلاعات کرده‌اند و از این رو انعطاف‌پذیری و سهولت زیادی را فراهم ساخته‌اند.

پایگاه های داده NoSQL با نام «نه تنها SQL» غیر جدولی هستند و داده ها را متفاوت از جداول رابطه ای ذخیره می کنند. پایگاه داده های NoSQL بر اساس مدل داده هایشان انواع مختلفی دارند که در ادامه هر کدام به اختصار توضیح داده شده اند.

هم‌اکنون با توسعه فناوری‌های مختلف (از جمله اینترنت اشیاء) قابلیت دریافت و تولید حجم عظیمی از داده‌ها، امکان ذخیره‌سازی و تحلیل آنها چالشی بزرگ به شمار می‌آید. داده‌هایی مانند داده‌های هواشناسی، فعالیت‌های آنلاین کاربران، اشیاء متصل شده به اینترنت و یا تحلیل‌های اقتصادی در قالب پایگاه‌های داده‌ای سنتی کارایی چندانی نخواهند داشت.

روش NoSQL برای سازمان‌دهی داده‌ها در واقع به معنی رها شدن از قیود است و از

این رو ابزار نگهداری، کوئری زدن و استفاده از اطلاعات را آزادانه‌تر می‌سازد.

پایگاه های داده NoSQL در اواخر دهه ۲۰۰۰ با کاهش چشمگیر هزینه ذخیره سازی پدیدار شدند و در پی آن ، مقدار دیتا اپلیکیشن های مورد نیاز برای ذخیره و پرس و جو افزایش یافت.

در این زمان رایانش ابری نیز از محبوبیت بیشتری برخوردار شد و توسعه دهندگان از ابرهای عمومی برای میزبانی برنامه ها و داده های خود استفاده کردند. آنها خواهان توانایی توزیع داده ها در چندین سرور و منطقه برای انعطاف پذیر ساختن برنامه هایشان و قرار دادن اطلاعات جغرافیایی هوشمند بودند. برخی از پایگاه های داده NoSQL مانند MongoDB این قابلیت ها را ارائه می دهند.

شرکتهای بزرگ اینترنت از جمله گوگل (BigTable)، آمازون (Dynamo)، لینکدین (Project Voldemort)، فیس بوک (Cassandra و HBase) سورس فورژ (MongoDB)، اوبونتو (CouchDB) طراحی و راه اندازی گروه پایگاه‌های داده NoSQL را بر عهده دارند. بخش قابل توجهی از این پروژه‌ها متن باز و آزاد است.

انواع پایگاه‌های داده NoSQL

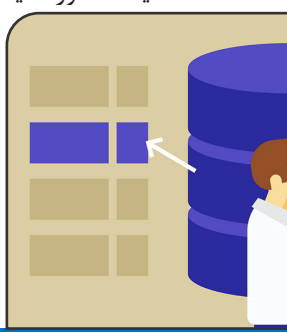
• پایگاه‌های داده کلید-مقدار یا: Key-Value Database

پایگاه داده های کلید-مقدار نوع ساده ای از پایگاه داده است که هر مورد حاوی کلیدها و مقادیر است. در این نوع از پایگاه داده اطلاعات در قالب جفت‌های کلید-مقدار یا Key-Value ذخیره می شود. و به دلیل ساده بودن در کارکرد، پرکاربردترین نوع پایگاه‌های داده NoSQL هستند.

پایگاه داده های کلید-مقدار برای مواردی که شما به ذخیره حجم زیادی از داده ها نیاز دارید اما برای بازیابی آنها نیازی به کوئری های پیچیده نیست بسیار مناسب است. Redis و DynanoDB از Key-Value Database های معروف هستند.

• پایگاه‌های داده ستونی یا Wide-Column Database:

شاید تصور کنید پایگاه‌های داده ستونی همان پایگاه‌های داده رابطه ای هستند. اما فقط در ظاهر این گونه پایگاه‌های داده شبیه به نوع رابطه ای است. در پایگاه‌های داده رابطه ای لازم است که تعداد و نوع ویژگی‌های هر موجودیت و مقادیر داخل آن مشخص و ثابت باشد. این در حالی است که در پایگاه‌های داده ستونی، هر ستون در رکوردهای مختلف می‌تواند شامل داده هایی با ساختار و نوع متفاوت باشد.



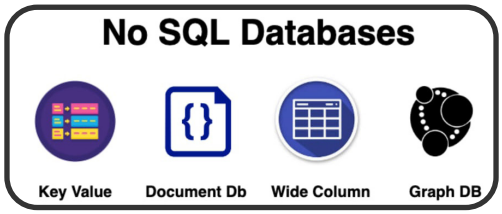
Wide-Column Database انعطاف پذیری زیادی نسبت به پایگاه داده رابطه ای ایجاد می کنند زیرا لازم نیست هر ردیف دارای ستون های یکسان باشد و برای زمانی که شما نیاز به ذخیره مقدار زیادی داده دارید بسیار مناسب هستند و از پایگاه‌های داده ستونی معمولاً برای ذخیره داده های اینترنت اشیا(Internet of Things data) و داده های نمایه کاربر استفاده می شود.

Cassandra و HBase دو مورد از محبوب ترین ها در این نوع هستند.

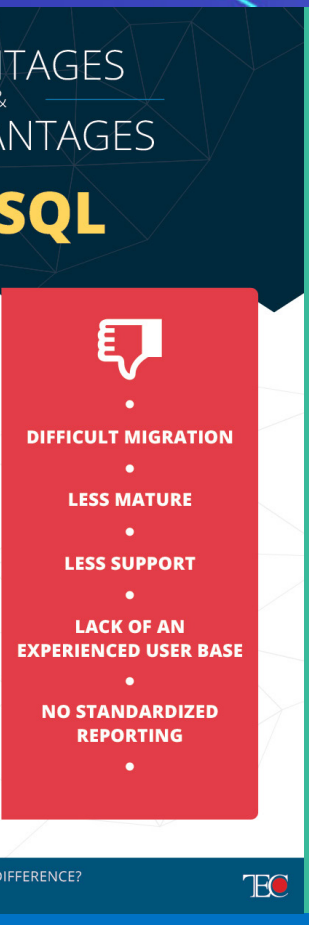
• پایگاه‌های داده سندی یا : Document Database

در این گونه پایگاه‌های داده برای ذخیره سازی داده‌ها از اسناد JSON(JavaScript Object Notation) یا XML استفاده می‌کنیم. هر سند شامل جفت فیلدها و مقادیر است. این مقادیر می توانند انواع مختلفی مانند Strings, numbers, booleans, arrays, objects و . . . داشته باشد.

پایگاه‌های داده سندی معمولا برای ذخیره سازی و استفاده از داده‌های پراکنده و بی ساختار استفاده می شوند MongoDB به طور مداوم به عنوان محبوب ترین پایگاه داده NoSQL جهان با توجه به موتورهای DB رتبه بندی می شود و نمونه ای از document database است.



یکی دیگر از نمونه پایگاه داده‌های اسنادی، CouchDB می‌باشد که کتابخانه‌های مورد نیاز برای همه زبانهای برنامه نویسی مهم را ارائه داده و مورد توجه بسیاری از توسعه دهندگان قرار گرفته است. ضمن اینکه



شرکت Cloudant سرویس میزبانی این نوع پایگاه داده‌ای را در ابر ارائه می‌دهد و استفاده از این پایگاه داده به صورت سرویس روز به روز ساده‌تر و کم هزینه‌تر شده است.

• پایگاه‌های داده گرافی یا Graph Database:

در این نوع از پایگاه‌های داده برای ذخیره سازی موجودیت‌ها و روابط بین آن‌ها از گراف استفاده می‌کنیم و داده ها در نود ها و یالها ذخیره می شوند. نودها معمولاً اطلاعات مربوط به افراد، مکان ها و . . . را ذخیره می کنند در حالی که یال ها اطلاعات مربوط به روابط بین نود ها را ذخیره می کنند. پایگاه‌های داده گرافی برای مواردی که در آن‌ها به ایجاد ارتباط‌های متعدد بین جداول و پیمایش روابط احتیاج داریم بسیار مناسب هستند. Neo٤j و JanusGraph نمونه هایی از پایگاه داده گرافی هستند.

منابع:

https://Vlearn. com/programming/what-is-nosql

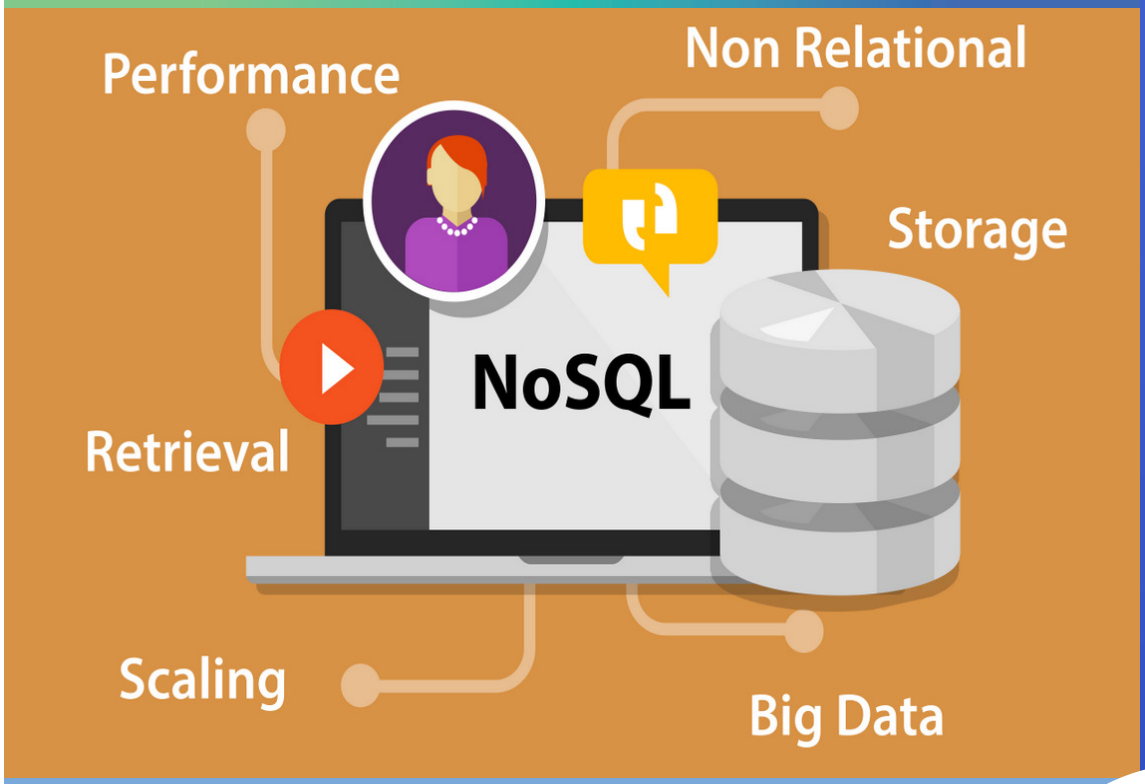
https://www. semanticscholar. org/

International Journal of Database Theory and Application Vol. ٦, No. ٢٠١٣ .٤

https://www. mongodb. com/

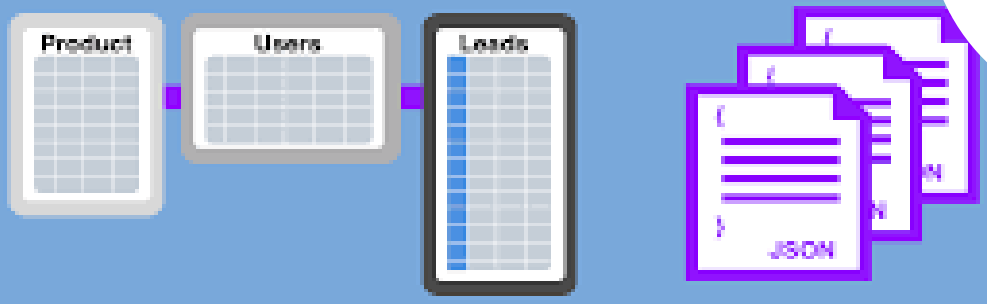
https://blog. faradars. org/

https://db-engines. com/en/search

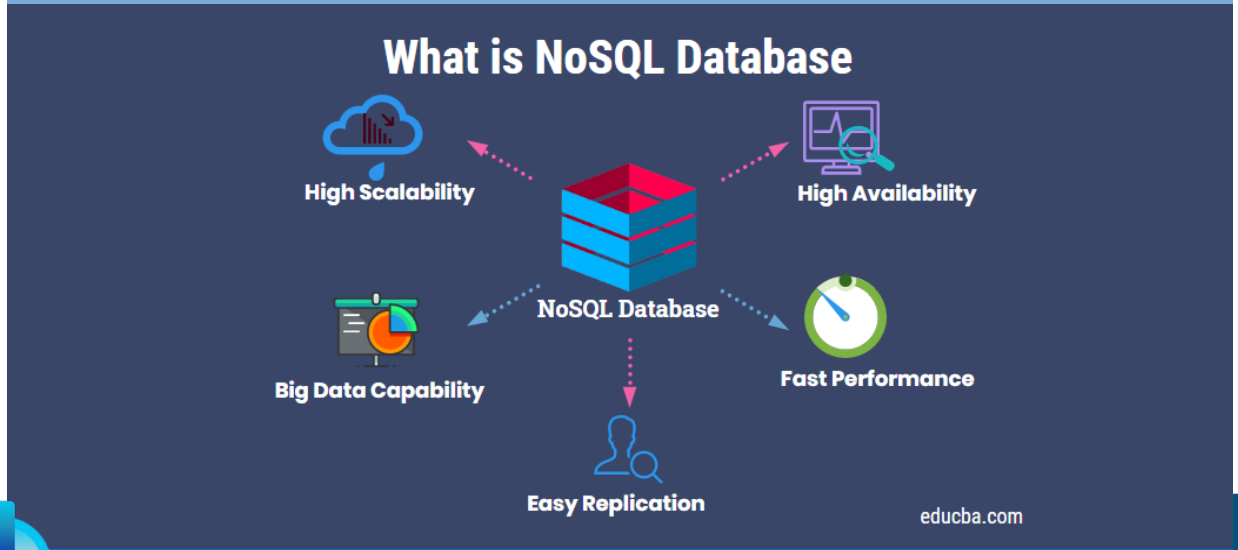


nos

ADVAN
&
DISADVA
No



NoSQL



- EASIER TO MANAGE
- OPEN SOURCE
- QUICK DEVELOPMENT
- EASY TO SCALE
- QUICK SETUP
- EXTREME FLEXIBILITY

SQL VS. NOSQL DATABASES: WHAT'S THE D
technologyevaluation.com

علی اسدی
ورودی ۹۸

مروری بر انواع دیتا و دیتابیس

دیتا و داده های بزرگ:

به مجموعه ای از اعداد(Quantities) و کاراکتر ها(Characters) و نماد ها، (Symbols) دیتا یا داده ها (data) گفته می شود. مقدار بزرگی از داده ها که برای ذخیره، پردازش و تحلیل آنها به روش ها و به کمک ابزارهای سنتی بسیار بزرگ هستند big data گفته می شود. اندازه ی این نوع دیتا به صورت نمایی افزایش می یابد و به همین دلیل، یک مشکل بزرگ در تحلیل آنها به روش تحلیل دیتای معمولی به وجود می آید، اما بر خلاف وجود مشکلات فراوان در تحلیل و پردازش داده های بزرگ این نوع داده ها مورد علاقه ی شرکت های بزرگ و کسب و کار های فراوان می باشند. علت هم این است که به آنها در تصمیمات تجاری و جذب بیشتر مشتریان کمک می کند. پس big data بسیار ارزشمند هستند و این لزوم ذخیره ی این نوع دیتا است.

به طور مثال شبکه ی بزرگ Netflix بیش از ۱۰۰ میلیون مشتری در سراسر جهان دارد، این کمپانی داده های رفتاری کاربر (User behavior data) که شامل سرچ ها و فیلم های تماشا شده توسط کاربر و . . . می شود را جمع آوری و ذخیره می کند، سپس با آنالیز این داده ها محتوایی را به مشتری نشان می دهد که مورد علاقه ی اوست، از طرفی با داده های آماری بدست آمده سعی می کند که محتوایی تولید کند، که اکثرا کاربران یک منطقه به تماشای آن علاقه داشته باشند و به این ترتیدیتب تعداد مشتریان نه تنها ثابت می ماند بلکه افزایش می یابد.

data به سه دسته ی زیر تقسیم می شود:

Structured data

به داده هایی که با یک فرمت مشخص و به ترتیب داخل دیتابیس ذخیره شده اند گفته می شود.

Unstructured data

بر خلاف Structured data که به فرم ستون-سطر در دیتابیس ها ذخیره می شوند این نوع از داده ها هیچ فرمت ذخیره ی شفافی ندارند.

Semi-Structured data

این نوع دیتا خط بین ۲ نوع دیتای قبلی -که شفاف نیست- می باشد. به فرمت Structured نیستند اما یکسری خصوصیات سامان دهی دارند که کار پردازش آنها را بسیار ساده می کند برای مثال فایل های JSON.

انواع دیتابیس ها:

برای کار با داده های بزرگ Framework های مختلفی وجود دارند که کار با big data را بسیار ساده تر می کنند. برای مثال یک فریم ورک معروف Hadoop می باشد که یک محیط را برای کار روی داده های بزرگ در اختیار ما می گذارد.

برای ذخیره سازی دیتا های معمولی در حال حاظر، در بین دیتابیس های SQL دیتابیس های رابطه ای یا Relational database ها هستند که DBMS ها یا RDBMS های مختلفی هم برای کار با این نوع دیتا بیس وجود دارد مانند MySQL، Microsoft SQL server و. . . . اما SQL دیتابیس ها نقاط ضعفی دارند، که باعث می شود برای ذخیره سازی big data مناسب نباشند و به همین دلیل نوع جدید از دیتا بیس ها مطرح شد به نام NoSQL دیتابیس ها که DBMS ها و ابزارهای مختلفی برای کار با این نوع دیتا بیس ها وجود دارد.

مقایسه ی SQL و NoSQL دیتابیس ها:

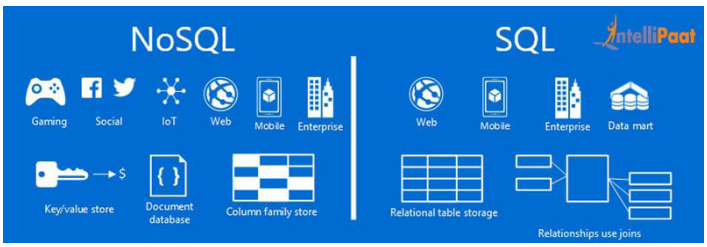
SQL دیتا بیس ها اطلاعات را با یک فرمت مشخص در جدول های خودشان ذخیره می کنند و برای ذخیره سازی دیتا باید، آن را به فرمت دلخواه پایگاه داده تبدیل کنیم و به همین دلیل دیتای ذخیره شده شفافیت دارد. بین دیتا ها روابطی وجود دارد که به وسیله ی مفاهیم جبر رابطه ای و تبدیل آن به زبان SQL که توسط نرم افزار DBMS با پایگاه داده ها ارتباط برقرار می کند می توانیم داده ها را از جدول های مختلفی که در آن قرار دارند استخراج کنیم و تحویل کاربر دهیم. وجود این نوع روابط و فرمت های مشخص، باعث شده که دیتابیس های رابطه ای در مواجه با داده های بزرگ نتوانند عمل Load را هندل کنند و از طرفی تنها به صورت عمودی (Vertically) قابل مقیاس بندی یا Scale هستند که در مواجه با داده های بزرگ ما به مقیاس بندی افقی نیز نیاز داریم زیرا، اندازه ی داده ها به صورت نمایی افزایش می یابد و یک سیستم، پاسخگوی تعداد بالای کاربران نیست. این نوع دیتابیس ها به صورت افقی قابل مقیاس بندی نیستند زیرا ایجاد ارتباط بین جدول ها که در دیتاسنتر ها یا سیستم های مختلف ذخیره شده اند پیچیده و حتی

غیر ممکن می باشد. در حقیقت در SQL دیتابیس ها باید یک سیستم را قوی و قوی تر کنیم تا پاسخگوی درخواست های بالای کاربران باشد اما در NoSQL دیتابیس ها می توان تعداد سیستم ها را افزایش داد و هر بخش از دیتا را در یک سیستم ذخیره کرد (distributed data) و درخواست میان این سیستم ها پخش شده و هر بخش مسئول اطلاعات همان بخش است. از طرفی ساختار NoSQL دیتابیس ها به صورت کلید-مقدار (key-value) می باشد و به کمک تابع هشینگ(Hashing function) و کلید می توان مشخص کرد که داده ی مورد نظر در کدام سیستم قرار گرفته است. البته باید اشاره کرد در صورت وجود درخواست های نوشتن یا Write فراوان به دلیل اینکه دیتا در چندین مکان باید تغییر یابد یکی از نقاط ضعف این نوع دیتابیس ها می باشد. Big data بهتر است که در دیتابیس های NoSQL ذخیره شوند که انواع مختلفی دارند. در این نوع دیتا بیس ها هر آیتم به خودی خود وجود دارد و داده ها فرمت مشخصی ندارند.

NoSQL دیتابیس ها برخلاف SQL ها بدون شما یا به عبارتی Schema-less هستند، یعنی آیتم های ذخیره شده در داخل دیتابیس نیاز به دارا بودن ساختار یکسان نیستند و این نکته ی مهمی است که باعث شده دیتاهای بزرگ بتوانند به راحتی در داخل NoSQL ها ذخیره شوند زیرا از نوع Unstructured هستند و فرمت شفافی ندارند.

برای ذخیره سازی و مدیریت big data نرم افزار های مختلفی وجود دارد:

DBMS های معروف در این زمینه MongoDB, Cassandra, Neo۴j, HBase



هستند. همچنین Hadoop, Talented, Zookeeper نیز از ابزار ها و محیط های کاری برای ذخیره سازی و کار با داده های بزرگ هستند.

حال که انواع دیتابیس ها را شناختیم و متوجه ی این نکته ی مهم شدیم که که برای ذخیره سازی آنها باید از NoSQL دیتابیس ها استفاده کنیم، انواع این نوع دیتابیس را به همراه سیستم مدیریت معروف هر کدام را بررسی خواهیم کرد. البته یک نکته ی مهم در این میان این است که در پروژه های بزرگ، از ترکیب دیتابیس های SQL و NoSQL استفاده می کنند.

انواع NoSQL دیتابیس ها:

این نوع دیتابیس ها همگی فرمت کلید و مقدار را دارند و به اشکال مختلفی پیاده سازی شده اند.

معروف ترین نوع این دیتابیس ها Document-base ها هستند مانند MongoDB و یا CouchDB در حقیقت همان فرمت کلید و مقدار را دارند اما در اینجا مقدار ها یکسری فایل مشخص هستند مانندJSON, Binary, JSON, XML, ... که به هرکدام یک کلید نسبت داده شده و داده های ما در این فایل ها ذخیره می شوند. این روش بسیار بهینه و پرسرعت است.

نوع دوم هم key-value معمولی است مانند Oracle. در این نوع مقدار می تواند هر نوع شی ای باشد مانند یک فایل یک ویدیو، یک عکس، یک فایل تکست یا یک عدد و. . . .

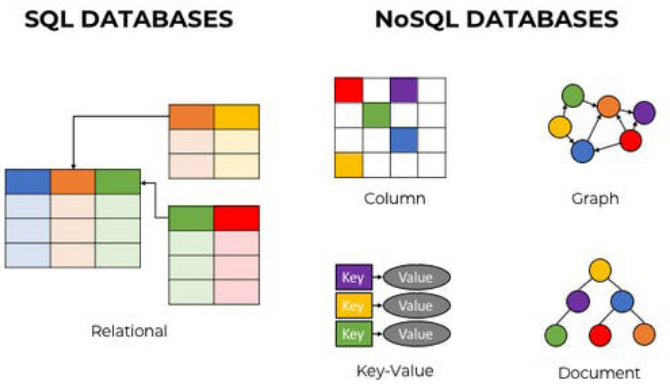
نوع سوم Graph-store ها هستند مانند Neo۴j که در این زمینه بسیار معروف است. در این دیتابیس را بر اساس مفهوم گراف ها پیاده سازی کرده اند. در حقیقت، داده ها در نود های گراف ذخیره می شوند و ارتباط بین این داده ها با یال های گراف نشان داده خواهند شد. زمانی که ارتباط میان موارد ذخیره شده داخل دیتابیس برای ما مهم باشد از این نوع اشتفاده خواهیم کرد.

نوع چهارم Wide-column store ها هستند مانند Cassandra. فیسبوک این روش را ابداع کرده و برای ذخیره سازی داده های خود از آن استفاده می کند. مانند دیتابیس های رابطه ای این نوع دیتا بیس یک جدول بزرگ دارد که تفاوت آن در این است که نام ستون ها نیازی نیست که برای همه ی سطر ها ثابت باشد.

MongoDB:

نرمال سازی هایی که در مدل های رابطه ای وجود داشت در MongoDB وجود ندارد. تمام قدرت و ایده ی این دیتابیس وجود نداشتن شِما در آن است و مجموعه ای(Collection) از داکیومنت ها است و به دلیلِ عدم وجود جدول های مدل رابطه ای بسیار انعطاف پذیر است، در واقع داده ها را در یک مکان و در یک فایل بسیار بزرگ ذخیره می کند، البته فایل های متعددی نیز می تواند وجود داشته باشد. داخل یک فایل برای هر داده یکسری فیلد ها وجود دارد و اگر در جایی از پروژه متوجه شدیم به یک فیلد جدید نیاز داریم به راحتی می توان آن را اضافه کرد چه برای داده هایی که از آن به بعد وارد می شوند، چه برای تمام دیتابیس و حتی می توان یک فیلد را حذف نمود، به همین دلیل این دیتابیس بسیار سریع است و کارایی بالایی دارد.

MongoDB برای پروژه هایی خواندن و نوشتن های بزرگ از داخل



دیتابیس دارند بسیار مناسب است، برای مثال اگر برنامه ای داشته باشیم که تک تک حرکات کاربر را دنبال می کند و دیتای آن را داخل دیتابیس ذخیره می کند. یا برای برنامه هایی که در هر ثانیه هزاران عمل نوشتن داخل دیتابیس را انجام میدهد بسیار مناسب می باشد.

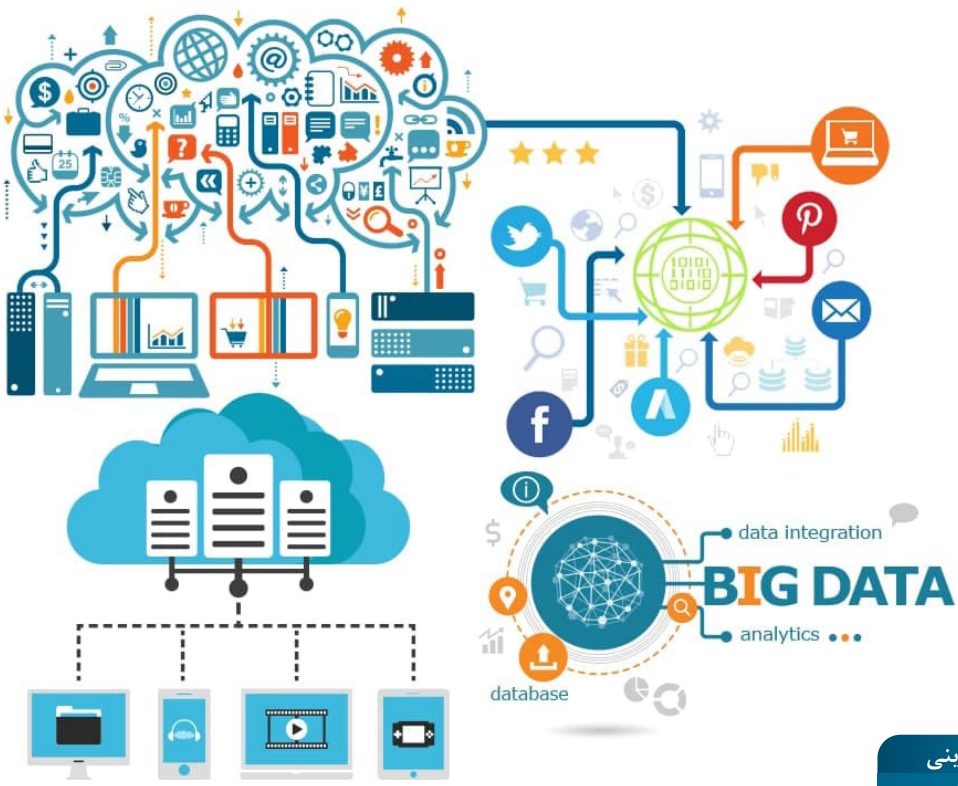
Replication set

MongoDB یک سرور اصلی دارد که اطلاعات آن روی سرور های ثانویه تکرار می شود تا درخواست ها روی سریع ترین و خلوت ترین سرور پردازش شوند و اگر سرور اصلی از دسترس خارج شود برنامه ی ما همچنان قابل استفاده باشد، البته سرور اصی در سریع ترین زمان ممکن باید دوباره آنلاین شود.

Sharding

برای کار با داده های بزرگ سرور های اصلی مختلف را به همراه سرورهای ثانویه برای هرکدام در نظر می گیرند، به دلیل اینکه حجم بالای این داده ها یک سیستم کافی نیست.

اپ سرور های مختلفی که در حال ران شدن هستند به کمک Mongos و ران کردن آن، با سرور MongoDB ارتباط برقرار می کنند و از طرف دیگر ۳ Config server داریم که مشخص می کنند که در کدام یک از سرور های اصلی دیتای مورد نظر وجود دارند و در نتیجه به دیتابیس مورد نظر دسترسی پیدا می کند.



علیرضا زینی
ورودی ۹۸

سیستم مدیریت کلان‌داده و شرکت‌های بزرگ نرم‌افزاری

دوران معاصر دوران عصر اطلاعات می باشد، عصری که در آن رشد روز افزون اطلاعات بیش از هر زمان به چشم می خورد. در این عصر سامانه های اطلاعاتی به طور پیوسته در حال دریافت و ذخیره سازی طیف وسیعی از داده های جمع آوری شده در پایگاه داده و تبدیل آن به اطلاعات مفید برای تصمیم گیری و مدیریت در امور مختلف هستند. در این تحقیق به موضوع مدیریت و پردازش کلان داده ها و در انتها به بررسی موردی شرکت های بزرگ نرم افزاری، نحوه ذخیره سازی داده های آنها میپردازیم.

سیستم مدیریت کلان داده های پایگاه داده (Big Data DBMS):

کلان داده ها یا Big Data به معنای دارایی‌های اطلاعاتی [یک مجموعه یا سازمان] است که:

- حجم بالایی دارند.
- با سرعت زیاد تولید می شوند یا تنوع گسترده ای دارند.

که نیازمند شیوه‌های پردازش نوآورانه با هزینه‌ی مناسب هستند تا بتوان از آن برای اتوماسیون فرایندها، تصمیم گیری و بهبود شهود و بینش [در سازمان] بهره گرفت.

کلان داده ها به تنهایی مانند یک دی وی دی خامی هستند، که ارزشی والایی نخواهند داشت، تحلیل و مدیریت این داده ها باعث ارزشمندی آنها خواهند شد. بنابراین، مدیریت و تحلیل کلان داده ها مهم ترین هدف از جمع آوری داده ها است. طبیعی است که هرچه تحلیل داده ها دقیق تر باشد، منجر به تصمیم بهتر، ریسک و هزینه کمتر خواهد شد.

سیستم مدیریت پایگاه داده(DBMS) پل ارتباطی میان پایگاه داده، کاربر و اپلیکیشن است که وظیفه دارد ارتباط و تعامل میان این اجزا را آسان سازد، نرم افزار DBMS درخواست دیتا از یک اپلیکیشن را قبول کرده و سپس به سیستم عامل دستور ساخت دیتایی خاص را ارسال می کند.

روند تاریخی طراحی سیستم مدیریت پایگاه داده:

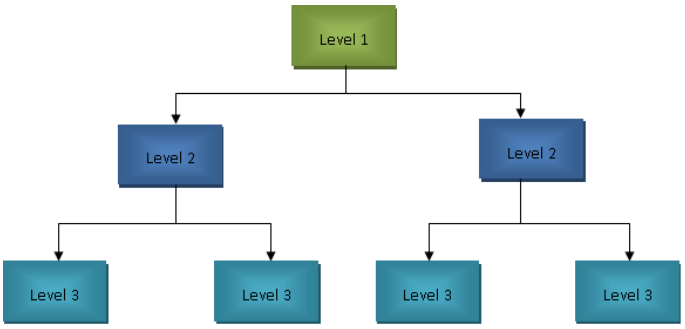
- سال ۱۹۶۰، چارلز بکمن اولین سیستم DBMS را طراحی کرد.
- سال ۱۹۷۰، سیستم مدیریت اطلاعات (IMS) کمپانی IBM توسط ادگار کاد معرفی شد.
- سال ۱۹۷۶، پیتر چن مدل موجودیت-رابطه (Entity-relationship) یا همان ER مدل را معرفی کرد.
- سال ۱۹۸۰، مدل رابطه ای به عنوان یکی از کامپوننت های اصلی دیتابیس پذیرفته شد.
- سال ۱۹۸۵، سیستم DBMS شی گرایی توسعه داده شد.
- سال ۱۹۹۰، سیستم DBMS شی گرایی با رابطه ای ادغام شد.
- سال ۱۹۹۱، مایکروسافت از نرم افزار خود رونمایی کرد و این محصول جایگزین تمام محصولات DBMS دیگر شد.
- سال ۱۹۹۵، از اولین اپلیکیشن پایگاه داده اینترنت رونمایی شد.
- سال ۱۹۹۷، زبان نشانه گذاری گسترش پذیر یا همان XML به روی پردازش دیتابیس اعمال شد. بسیاری از کمپانی های نرم افزاری XML را با محصولات DBMS خود ادغام کردند.

از نظر ساختار و قواعد ذخیره اطلاعات انواع مختلفی از سیستم های مدیریت پایگاه داده توسعه داده شده است، هر یک از این نوع ها متناسب با هدف و نیازی که وجود دارد استفاده می شود، از جمله آنها:

سلسله مراتبی (Hierarchical DBMS): در این پایگاه داده مدل داده ها درساختاری شبیه به درخت می باشد به طوری که داده ها به صورت سلسله مراتبی از بالا به پایین یا از پایین به بالا ذخیره می شود. در این مدل داده ها به صورت شاخه و برگ های درخت نمایش داده می شوند. در این مدل دیتای هر برگ فقط میتواند یک شاخه داشته باشد.

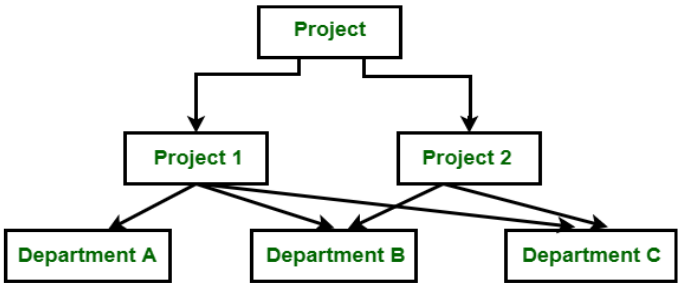
مدل شبکه ای (Network DBMS): در این مدل پایگاه داده، داده ها به صورت شبکه ای ذخیره می شوند. برخلاف مدل سلسله مراتبی در این مدل دیتای هر برگ می تواند چند شاخه داشته باشد.

مدل رابطه ای(Relational DBMS): این نوع سیستم مدیریت پایگاه داده، توسط ادگار کاد(Edgar Codd) در موسسه تحقیقاتی IBM ابداع شد. این مدل داده ها بر مبنای رابطه ریاضیات ذخیره می شوند، به دلیل سهولت یکی از رایج ترین سیستم های پایگاه



داده می باشد.

مدل شی گرایی(Object DBMS): در این مدل داده ها به صورت شی ها، ذخیره می شوند. یکی از ویژگی های ذخیره شی گرایی این خواهد بود که داده ها با نوع های مختلف را می پذیرد.



مزیت های سیستم مدیریت پایگاه داده:

- سیستم DBMS تکنیک های مختلفی برای بازیابی و ذخیره دیتا در اختیار شما قرار می دهد.
- سیستم DBMS به عنوان یک گرداننده کارآمد و با استفاده از دیتاهای یکسان، نیاز چندین اپلیکیشن را برطرف می کند.
- روش های مدیریت یکسان برای دیتا را فراهم می کند.
- برنامه نویسان اپلیکیشن هرگز در معرض جزییات نمایش و ذخیره سازی دیتا قرار نخواهند گرفت.
- سیستم DBMS از توابع گوناگون و قدرتمند در راستای بازیابی و ذخیره کارآمد داده ها استفاده می کند.
- این سیستم امنیت و یکپارچگی داده ها را در اختیار شما قرار می دهد.
- سیستم مدیریت پایگاه داده در قبال دسترسی غیر مجاز به دیتا، محدودیت های سختگیرانه ای اعمال می کند.
- روش دسترسی همزمان به دیتا در این سیستم به گونه ای است که فقط یک کاربر اجازه دسترسی به دیتای مشابه را خواهد داشت.
- کاهش زمان توسعه اپلیکیشن را برای توسعه دهندگان به ارمغان می آورد.

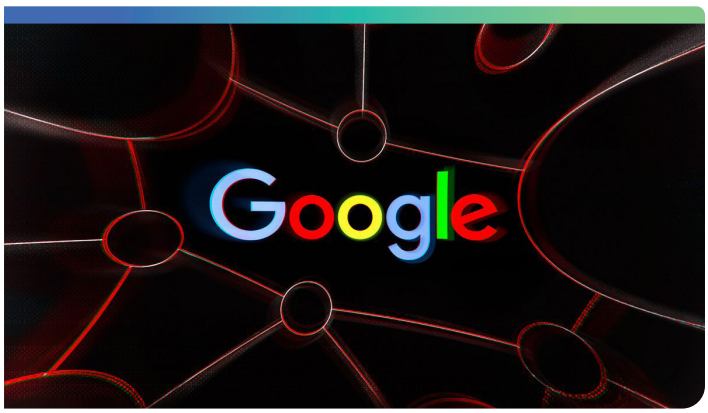
معایب سیستم های مدیریت پایگاه داده

علی رغم اینکه سیستم مدیریت پایگاه داده، مزیت های بسیاری در اختیارمان قرار می دهد اما معایبی نیز دارد که مهم ترین آنها عبارتند از:

- هزینه بالای خرید سخت افزار و نرم افزار سیستم های مدیریت پایگاه داده
- آسیب های جدی در صورت نوسانات از جمله نوسانات برق
- از دست رفتن برخی داده ها به دلیل استفاده تمام کاربران از یک برنامه واحد

بررسی موردی شرکت های بزرگ (Big Companies Case Studies):

شرکت های بزرگ نرم افزاری همواره در رقابت با هم بودند، با ارائه خدمات بهتره در سمت پیشگیری از هم تلاش می کنند، هر یک از این شرکت ها برای دادهای خود یا برای ذخیره سازی داده ها روش هایی را پیاده سازی کردند، از جمله این شرکت ها میتوان به شرکت های گوگل، آمازون، ماکروسافت، علی بابا و. . . اشاره کرد. که به بررسی برخی از این شرکت ها خواهیم پرداخت.



گوگل:

کلمه گوگل، برگرفته شده از واژه گوگول ، به معنی «یک عدد یک و صد صفر جلوی آن» است.

شرکت گوگل در سال ۱۹۹۸ به عنوان یک شرکت خصوصی، توسط لری پیج، سرگئی برین تاسیس شد. این شرکت در حوزه فناوری اطلاعات، شامل جستجوی اینترنتی، رایانش ابری و تبلیغات، نرم‌افزار و سخت‌افزار فعالیت می کند. محدوده فعالیت این شرکت کل جهان می باشد، به طوری که به‌طور تخمینی دارای بیش از یک میلیون سرور در سراسر جهان دارد، دفتر مرکزی آنها نیز در کالیفرنیا می باشد.

یکی از مشهور ترین و پرکاربرد ترین مجموعه این شرکت، موتور جستجوی آن می باشد، موتور جستجوی گوگل برای ذخیره اطلاعات شامل بخش های مختلفی است، به عنوان نمونه BigTable یک سیستم طراحی شده گوگل برای ذخیره سازی توزیع شده

است، برای ذخیره حجم عظیم از اطلاعات ساخت یافته روی سیستم های توزیع شده استفاده می شود. برخلاف پایگاه های داده رابطه ای، BigTabl یک سیستم توزیع شده چندبعدی پایدار و از نوع Stored Map است. و برای ذخیره داده های حجیم روی چند سرور استفاده می شود.

آمازون:

آمازون، شرکت تجارت الکترونیک آمریکایی است، که در سال ۱۹۹۴ توسط جف بیزوس، در شهر سیاتل تأسیس شد. دفتر مرکزی آن در سیاتل واقع شده است.

اطلاعات این شرکت با سرویس S۳ ذخیره می شود.

آمازون S۳ یا سرویس ذخیره‌سازی ساده آمازون (Simple Storage Service) از ذخیره‌سازی شیءگرا استفاده می کند که داده‌ها را به عنوان یک شی مستقل ذخیره می کند. هر شی می‌تواند متادیتا و شناسه مرتبط خود را ذخیره کند.

علی بابا:

علی بابا مانند آمازون، یک شرکت تجارت الکترونیک چینی است که در سال ۱۹۹۹ توسط جک ما تاسیس شد. در زمینه های نرم افزاری و سخت افزاری (هوش مصنوعی و . . .) فعالیت هایی داشته است اما به عنوان یک سایت خرید و فروش آنلاین معروف است، این شرکت دفترهای مرکزی متعددی در کشور های مختلف دارد و داده های آن نیز به صورت ابری ذخیره سازی می شود.

مایکروسافت:

کلمه مایکروسافت برگرفته از میکرو و سافت (Microcomputer software) می باشد. این شرکت یکی از بزرگترین شرکت های نرم افزاری جهان است، که درسال ۱۹۷۵ توسط بیل گیتس و پل آلن تاسیس ودفتر مرکزی آن در شهر رموند واقع شده است. این شرکت در تلاشهای خود خدماتی را برای ذخیره سازی داده ها راه اندازی و ارائه داد، به عنوان نمونه مایکروسافت آژور یکی از فضا های ابری است که توسط این شرکت تولید شد، یک نمونه فضای ابری وان درایور است. وان درایو؛یک سیستم میزبانی فایل است، که به کاربران امکان آپلودکردن و همگام سازی فایل ها را در یک فضای ابری می دهد. یکی از مهم ترین فواید این فضای ابری این است که، شرکت ها نیاز به زیرساخت های سخت افزاری (با هزینه های بالا) ندارند.

(در کنار ذخیره سازی داده ها این شرکت سیستم های قدرتمندی در جهت مدیریت داده ها طراحی کرده است.)

منابع:

Using web services for remote data access and distributed applications

https://Vlearn. com/computer-science/what-is-dbms

http://www. pandia. com/articles/gartner

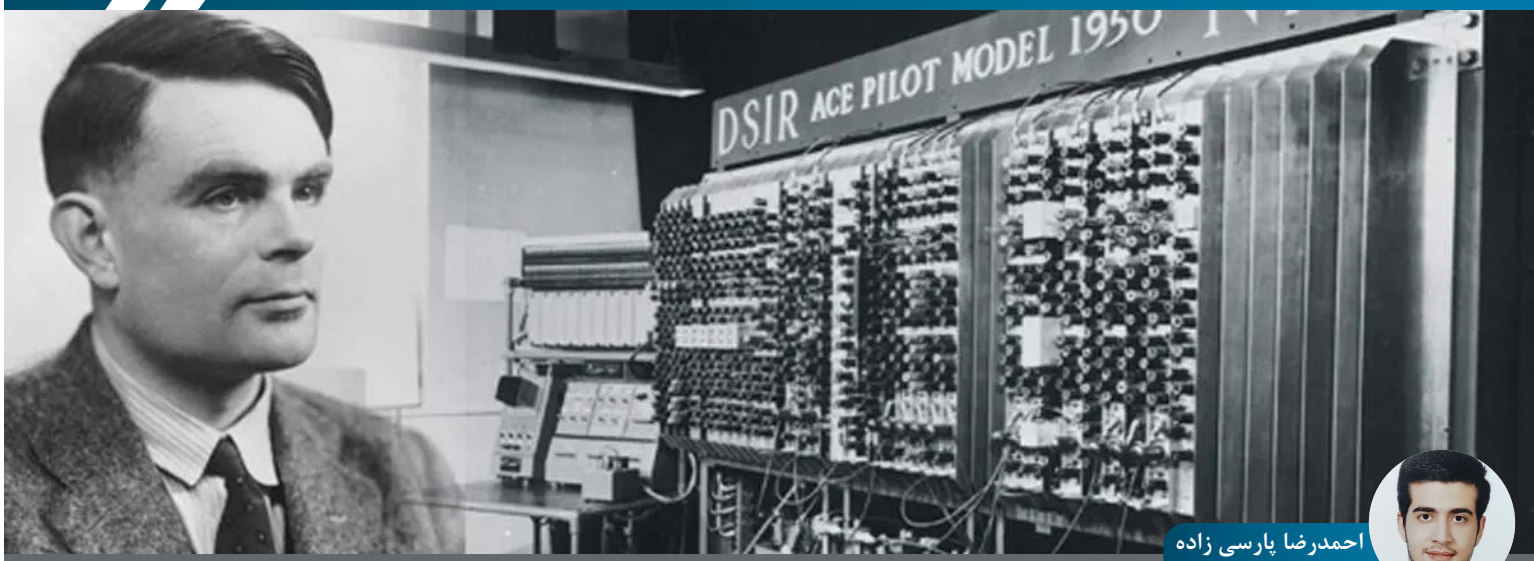
https://danup. ir/

https://dba. tosinso. com/

خداداد هلیلی_جلیل مظلوم_بهرنگ هادیان (برسی کاربردهای نظامی فناوری کلان داده و نقش آن در مدیریت صحنه نبرد)

سون لرن

ویکی پدیا



احمدرضا پارسی زاده
ورودی ۹۸

بیوگرافی آلن تورینگ

آلن تورینگ، نامی شناخته شده در دنیای کامپیوتر است. تورینگ را به خاطر فعالیت های تحقیقاتی در خلال جنگ جهانی دوم، پدرهوش مصنوعی نامیده اند.

آلن متیسون تورینگ (Alan Mathison Turing) دانشمند بریتانیایی متولد قرن بیستم است. تورینگ را می توان با القابی مانند دانشمند علوم کامپیوتر، ریاضی دان، استاد منطق و فلسفه، تحلیل گر رمز و زیست شناس ریاضی شناخت. تحقیقات این دانشمند بزرگ را می توان پایه و اساس علوم کامپیوتر مدرن دانست. نتایج تحقیقات او، مبنا و مفهموم هایی از الگوریتم و پردازش را به کمک ماشین مخصوص یعنی ماشین تورینگ، توصیف کرد. ماشین تورینگ (Turing Machine) را می توان از اولین نسل های کامپیوتردانست.

آلن تورینگ در خلال جنگ جهانی دوم برای انگلستان و جبهه ی متفقین فعالیت می کرد. او در یک مرکز آموزشی رمز نگاری تعلیم دیده و در پروژه ای برای رمز گشایی ارتباط نیروهای دریایی نازی ها نقش مدیر را بر عهده داشت. هدف اصلی تیم تحت رهبری او رمز گشایی پیام های ماشین معروف آلمانی ها به نام انیگما بود.

روش هایی که تورینگ و تیمش در این پروژه ابداع کردند، سرعت رمز گشایی پیام های این ماشین را افزایش داده و موجب موفقیت های زیادی برای ارتش متفقین در جنگ جهانی دوم شد. این پیروزی ها، جان میلیون ها نفر (تقریبا ۱۴ میلیون نفر) را در این جنگ خانمان سوز نجات داد.

آلن تورینگ پس از جنگ نیز به مطالعات و فعالیت های خود پیرام. ن علوم کامپیوترادامه داد. او در این زمان به علوم زیستی ریاضیاتی علاقه مند شده و مقالاتی را در زمینه ی ریخت زایی و واکنش های ساعتی شیمیایی منتشر کرد. آلن تورینگ در سال ۱۹۵۲ به خاطر گرایش های خاص جنسی محکوم شناخته شده و مجازات تزریق دی اتیل استیل بسترول برای او در نظر گرفته شد. او در سال ۱۹۵۴ و ۱۶ روز پیش از جشن تولد ۴۲ سالگی از دنیا رفت. بزرگترین و معتبر ترین جایزه ی دنیای کامپیوتر، از سال ۱۹۶۶ به نام این دانشمند فرهیخته ، با عنوان Turing Award به فعالان تأثیر گذار عرصه ی کامپیوتر اعطا می گردد.

تولد و تحصیل

آلن تورینگ متولد ۲۳ ژوئن ۱۹۱۲ در منطقه ی ویدا میل لندن است.

پدر او ژولیوس متیسون تورینگ ، یکی از کارگذاران شرکت Indian Civil Service در چاتراپور هند بود. او به خاطر کار در این شرکت مجبور به سکونت در منطقه ی هند بریتانیا بود و همسرش اتل سارا را نیز به آن منطقه برده بود. والدین آلن تمایل داشتند فرزندانشان را در بریتانیا پرورش بدهند، به همین دلیل به میدا ویل در لندن نقل مکان کردند.

قرار داد کاری پدر آلن با شرکت، والدین او را مجبور می کرد که مدام در حال سفر های کاری بین هند و انگلستان باشند. این سفر های کاری باعث شدند که آلن و برادر بزرگترش جان، نزد یک زوج ارتشی بازنشسته پرورش یابند. آلن از سال های کودکی نبوغ و استعداد خود را نشان داده بود. پدر و مادر او در سال ۱۹۲۷ خانه ای در منطقه ی گیلفورد خریداری کردند تا او در زمان تعطیلات مدرسه در سکونت داشته باشد.

معرفی مشاهیر

آلن در سن ۶ سالگی به مدرسه ی شبانه روزی سنت مایکل رفت. مدیر و معلمان این مدرسه، به سرعت متوجه هوش سرشار او شدند. او در سال های ۱۹۲۲ تا ۱۹۲۶ در مدرسه غیر انتفاعی Hazelhurst به تحصیل پرداخت. درسال ۱۹۲۶ و در ۱۳ سالگی ، او به مدرسه ی شربورن رفت. مدیر این مدرسه که فعالیت های خارج از برنامه و علاقه ی شدید آلن به تحقیقات پیشرفته ی ریاضی را می دید، به والدینش هشدار داد که این نوع تحصیل مناسب او نیست. به هر حال آلن با وجود مخالفت ها و عدم درک از طرف معلمان، به فعالیت های خود ادامه می داد و حتی در سن ۱۶ سالگی نظریات انیشتین را بررسی می کرد.

یکی از دوستان نزدیک آلن در سال های نوجوانی، کریستوفر مورکام بود. این دو نفر رابطه ی دوستانه ی قوی ای داشتند. مورکام در سال ۱۹۳۰ و به دلیل به سل گاوی از دنیا رفت. بسیاری از کارشناسان بر این باورند که اعتقادات و رفتار های خاص آلن در سال های جوانی، نشأت گرفته از مرگ این دوست نزدیکش بوده است.

آلن تورینگ در سال ۱۹۳۱، برای تحصیلات دانشگاهی وارد Kings college دانشگاه کمبریج شد و در سال ۱۹۳۴ با درجه ی عالی در رشته ی ریاضیات فارغ التحصیل شد. او در سال ۱۹۳۵ و در سن ۲۲ سالگی ، به خاطر پایان نامه ی حرفه ای کاندید عضویت درهیأت تحقیقاتی دانشکده شد. او در پایان نامه ی خود «حد مرکزی» را اثبات کرده بود. مطالعات پیشرفته ی تورینگ در علوم ریاضی، عموما به حل سوالات بزرگ این علم از طریق پردازش های کامپیوتری مربوط بود. نظریه های آلن در سال های بعدی، پیرامون ساخت ماشینی با قابلیت حل کردن مسائل مختلف ریاضی تألیف شدند. او تحصیلات تکمیلی را در دانشگاه پرینستون و حول مطالعات علوم رمز نگاری و رمز گشایی گذراند و در سال ۱۹۳۸ موفق به دریافت مدرک دکتری شد. جان وُن نویمان، ریاضی دان مشهور آمریکایی با مطالعات نظریه های تورینگ از او خواست که در تحقیقات با هم همکاری داشته باشند؛ اما آلن تصمیم گرفته بود که به انگلستان باز گردد.

تورینگ پس از رد پیشنهاد نویمان به کمبریج بازگشت و در تحقیقات معروف فیلسوف اتریشی ، لودویگ ویتگنشتاین با او همکاری کرد. تحقیقات دانشمند اتریشی پیرامون پایه های علوم ریاضی بود. آلن و لودویگ در جریان تحقیقات، بحث ها و

مجادله های زیادی با هم داشتند؛ چرا که آلن به فرمالیسم معتقد بود و در مقابل ، لودویگ اعتقاد داشت که ریاضیات ، هیچ حقیقت مطلقی را کشف یا اختراع نمی کند.

ورود به حوزه ی رمز نگاری:

فعالیت های اصلی تورینگ در زمینه ی رمز نگاری به دوران جنگ جهانی دوم مربوط است. در طول این جنگ، تورینگ در بلچلی پارک (Bletchley park) به تحقیق و ساخت دستگاه های رمزنگاری مشغول بود.

از سپتامبر سال ۱۹۳۸، آلن تورینگ در سازمان رمز نگاری بریتانیا به نام GC&CS به فعالیت پاره وقت مشغول بود. این سازمان امروزه با نام ستاد ارتباطات دولت بریتانیا شناخته می شود. تمرکز اصلی فعالیت های تورینگ در این سازمان ، رمز گشایی دستگاه انیگما بود. دستگاه انیگما متعلق به دولت آلمان بود که

دانشمندی آلمانی به نام آرتور شربیوس آن را اختراع کرده بود.

دستگاه انیگما در دهه ی ۱۹۲۰برای محافظت از ارتباطات تجاری، نظامی و دیپلماتیک استفاده می شد. آلن تورینگ رمز نگاری این دستگاه را به همراه همکار ارشدش دیلی ناکس انجام می داد.

در گردهمایی معروف ورشودر سال ۱۹۳۹، سازمان رمزنگاری لهستان به کشور های فرانسه و انگلستان اعلام کرد که روشی برای رمز گشایی دستگاه انیگما دارد. توریتک و ناکس پس از شنیدن این خبر، تلاش کردند تا روشی پایدارتر برای این رمز گشایی کشف کنند.

روش لهستانی ها روی فرایندی تشخیصی استوار بود. که امکان تغییر آن هر لحظه توسط آلمان ها وجود داشت. روشی که تورینگ ابداع کرد، رمزگشایی براساس Crib نام داشت و فرایندی مفید تر و عمومی تر بود. او با استفاده از همین روش ، مشخصات

عملکردی دستگاه رمزگشایی الکتریکی-مکانیکی Bombe را با بهبود دستگاه لهستانی Bomba تدوین کرد.

تورینگ ۴ سپتامبر سال ۱۹۳۹ و پس از آن که انگلستان به صورت رسمی در برابر آلمان اعلام جنگ کرد، به بلچلی پارک اعزام شد. بلچلی پارک در آن زمان ایستگاه جنگی GC&QC بود. آلن با استفاده از تکنیک های آماری توانست روشی برای آزمایش نتایج احتمالی در فرایند رمز گشایی ابداع کند. او روندی خلاقانه را ابداع کرد و دو مقاله در ارتباط با آنها تألیف کرد . نام مقالات او «استفاده از احتمالات در رمز گشایی» و «مقاله ای در باب آمار

تکرارهای ریاضی» بود. این مقالات آن چنان برای سازمان ارتباطات انگلستان با ارزش بودند که تا آوریل سال ۲۰۱۲ به آرشیو ملی انگلستان تحویل داده نشدند. یکی از ریاضیدانان این سازمان که با هویتی مبهم و نام ریچارد اخیرا در این باره مصاحبه کرده گفته است:

تأخیر ۷۰ ساله در تحویل این مقاله ها، نشان دهنده ی اهمیت آن هاست.

مقاله های تورینگ، جزئیات آنالیزهای ریاضی را برای سعی و خطا و پیدا کردن محتمل ترین تنظیمات ماشین های رمز نگاری شرح داده اند. سازمان ارتباطات انگلستان تا کنون حداکثر استفاده را از این مقاله ها کرده و اکنون با انتشار عمومی آن ها مشکلی ندارد.

نابغه ی ریاضیات و احتمالات قرن بیستم، در زمان فعالیت در بلچلی به رفتار های خاص شهرت داشت. همکارانش لقب Prof را برایش انتخاب کردند و تا سال ها از رفتارهای عجیب او یاد می کردند.

تورینگ در این زمان یک دونده ی دوی استقامت حرفه ای بود. امسافت ۶۴ کیلومتری تا لندن را برای حضور در جلسات مهم می دوید. این آمادگی بدنی باعث شد که این دانشمند ریاضیات برای حضور در المپیک۱۹۴۸ انگلستان کاندید شود. او تنها ۱۱ دقیقه دیرتر از قهرمان آن سال های انگلستان یعنی توماس ریچارد به خط پایان رسید و از تیم المپیک جا ماند.

اختراع دستگاهBombe

آلن تورینگ چند هفته پس از ورود به بلچلی پارک، دستگاه الکترومکانیکی Bombe را اختراع کرد. این دستگاه به جستجو میان تنظیمات احتمالی برای رمز گشایی دستگاه انیگما می پرداخت. تیم تحت رهبری تورینگ از اختراع خود شگفت زده بودند؛زیرا سرعت رمز گشایی آن بسیار بیش تر از دستگاه لهستانی بود. تنها کمبود آن ها در دوران جنگ کمبود نیروی انسانی و دستگاه رمز گشایی بود. تیم رمز گشایی بلچلی به خطر رنج از کمبود امکانات در اقدامی جسورانه به خود وینستون چرچیل نامه زدند. در این نامه که امضای تورینگ و نام او در صدر لیست امضاکنندگان بود به خاطر کمبود امکانات برای رمز گشایی گلایه کرده بودند. چرچیل نیز به این نامه واکنشی سریع انجام داد و فوری به مسئولان امر کرد که تمامی نیاز های این تیم رمزگشایی را تأمین کنند و در ان مدت پس از اقدام چیچیل حدود ۲۰۰ دستگاه رمز گشایی تولید شد.

یکی دیگر از دستگاه هایی که تورینگ در ساخت آن نقش داشت، Delilah بود. این دستگاه برای رمز گشایی پیام های صوتی استفاده می شد. او این دستگاه را به کمک دونالد بیلی ساخت. دستگاه دلایلا به خوبی کار می کرد اما ساخت آن بیش از حد به طول انجامید و نیروهای متفقین نتوانستند از این دستگاه در جنگ استفاده کنند.

از فعالیت های نرم افزاری تورینگ، در آن سال ها می توان به توسعه نرم افزار شطرنج برای کامپیوتر۱ Ferranti Mark اشاره کرد. البته این نرم افزار و الگوریتم به عنوان اولین نرم افزار بازی شطرنج در کامپیوترشناخته شد.

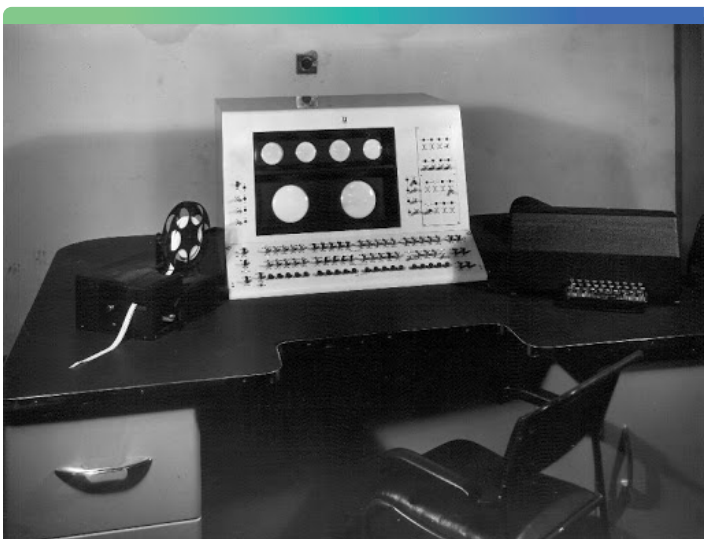
این نابغه ی ریاضی در سال ۱۹۴۸ روشی برای حل معادلات ماتریسی تحت عنوانLU decomposition ارائه کرد.

زندگی شخصی و مرگ:

آلن تورینگ یک نامزدی کوتاه مدت با همکارش در پروژه یHut۸ یعنی Juan clark داشت. نامزدی آن ها به دلایلی به ازدواج منجر نشد.

این نابغه ی ریاضی به خاطر همین دلایل و تمایلات در ۲۷ فوریه ی سال ۱۹۵۲ محاکمه شد. او از میان زندان و درمان هورمونی، دومی را انتخاب کرد. اتهامات و محاکمه های وی، حفاظت اطلاعاتی و امنیتی و همکاری او با دولت انگلستان را لغو کرد. البته شغل های دانشگاهی وی تا زمان مرگش پابرجا بود. او پس از اتفاقات سال۱۹۵۲ از ورود به خاک آمریکا منع شد اما مجاز به تردد در کشور های مختلف اروپایی بود. او هیچگاه در زندگی به جاسوسی متهم نشده بود.

در ۸ ژوئن سال ۱۹۵۴، خدمتکار جسد او را در اتاق خوابش پیدا کرد. او روز قبل از دنیا رفته بود. در پی تحقیقات روی جسد او مرگ با سم سیانید و به صورت خودکشی را تأیید کردند. هرچند تحقیقات تکمیلی سال های بعد احتمال استنشاق این اسید از طریق دستگاه پوشش دهی طلا به صورت اتفاقی را نیز تأیید کرد چون نمونه ای از آن در آزمایشگاهش و در کنار سیبی که نیم خورده بود یافت شد. .



منابع:

https://www. zoomit. ir/biography/-۲۴۴۵۳۵alan-turing-biography/

https://www. pinterest. com/pin/۷۶۹۹۷۴۸۶۷۵۲۱۳۲۰۷۲۵/

https://sokanacademy. com

https://www. biography. com/scientist/alan-turing

https://www. britannica. com/biography/Alan-Turing

https://mathshistory. st-andrews. ac. uk/Biographies/Turing/

FRONT-END

صبا حیدری دوست
ورودی ۹۸

۶ دوره‌ی برتر و رایگان یادگیری Frontend در سال ۲۰۲۱

در این مقاله قصد داریم تعدادی از بهترین دوره ها را برای توسعه وب معرفی کنیم که برای استفاده از آنها احتیاج به پرداخت هیچ هزینه ای ندارید. شما می توانید Frontend را از این دوره ها به راحتی در هر جا که هستید، یاد بگیرید. می توانید هر وقت که خواستید یادگیری خود را متوقف کنید، کمی تمرین کنید و دوباره یادگیری را از سر بگیرید.

۱. **Web Development By Doing: HTML / CSS From Scratch**

این دوره شما را با دو پایه توسعه وب آشنا می کند: HTML و CSS. علاوه بر همه چیز، اینکه نقش اساسی آنها چیست و چگونه در کنار هم یک وب سایت را تشکیل می دهند را خواهید آموخت.

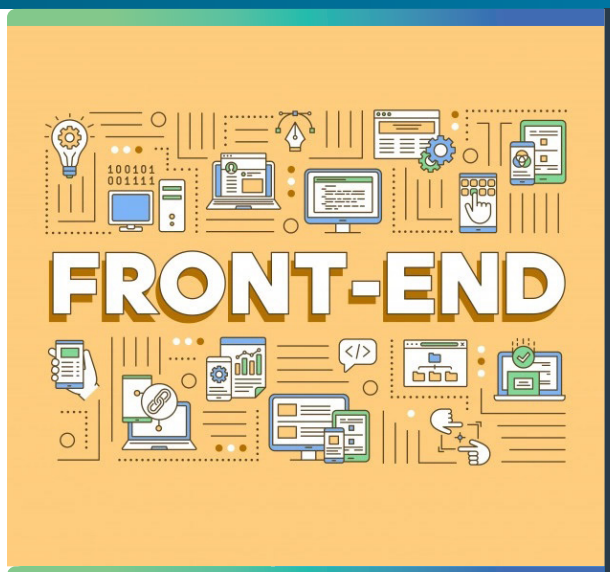
در بخش HTML، یاد می گیرید تمرین های ساده مثل ساخت یک صفحه وب ابتدایی، اضافه کردن تصاویر، تعویض رنگ ها، ساخت لیست ها را انجام دهید. همچنین در بخش CSS، می بینید که چگونه می توان صفحه وب را زیباتر کرد.

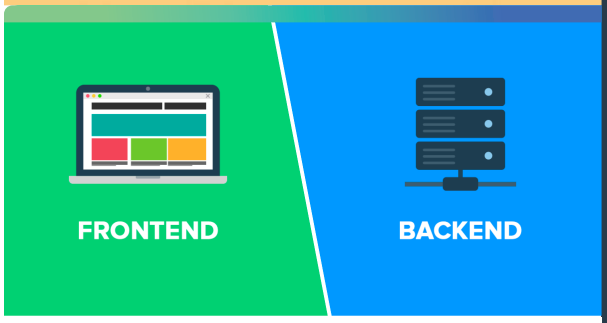
البته امروزه برای زیبایی صفحه از Bootstrap استفاده می شود؛ ولی برای استفاده از آن نیاز به دانستن مفاهیم پایه‌ای CSS دارید که در این دوره یاد داده می شود.

۲. **Foundations of Front-end Web Development**

از بهترین دوره های Frontend در Udemy می توان این دوره را نام برد که توسعه وب را از scratch آغاز کرده و علاوه بر آموزش تکنولوژی های پایه، کتابخانه ها و فریمورک ها، نحوه استفاده موثر آنها را با استفاده از ساخت پروژه‌های واقعی یاد می دهد.

برای اینکه توسعه دهنده Frontend خوبی باشید، نباید فقط به دانستن HTML، CSS و JavaScript اکتفا کنید بلکه فریمورک ها و کتابخانه های برجسته ای مانند React، Angular و BootStrap و... را هم باید یاد بگیرید. این دوره تمامی این ها را به‌طور تعاملی و جامع یاد خواهد داد.





۳. Learn HTML & CSS: How To Start Your Web Development Career:

آموزش های HTML و CSS و توسعه وب این دوره بسیار عالی است، مخصوصا برای توسعه دهنده های تازه وارد و فارغ‌التحصیلان علوم کامپیوتر که می خواهند در سال ۲۰۲۱ توسعه وب را یاد بگیرند.
این دوره به ۴ بخش اصلی تقسیم می شود:

۱) آموزش HTML
۲) آموزش CSS
۳) چالش نهایی کدزنی
۴) پروژه

همانطور که می بینید این دوره خیلی خوب ساختاربندی شده است. دو بخش اول، HTML۵ و CSS ۳ را از اول تا آخر آموزش می دهد و دو بخش نهایی آنچه از دو قسمت اول یاد گرفتید را می سنجد.

۴. Beginner Web :CSS۳ & Master the Basics of HTML۵ Development:

این دوره یکی از به روزترین دوره های توسعه وب است که در Udemy به صورت رایگان می توانید از آن استفاده کنید. این دوره روی آخرین نسخه HTML و CSS تمرکز دارد که برای کسی که تازه توسعه وب را شروع کرده، بسیار مهم است.
اگر تازه شروع به یاد گرفتن توسعه وب کرده اید، لزومی به یادگیری نسخه های قدیمی نیست. یادگرفتن آخرین ورژن بهترین کارایی را برای شما خواهد داشت. همچنین واضح است که وقتی برای این مهارت به دنبال کار می گردید، شرکت ها کسی را که از آخرین تکنولوژی های توسعه وب و آپدیت ها آگاه است، ترجیح می‌دهند.

۵. Web Design for Web Developers: Build Beautiful Websites:

این دوره یکی از بهترین منابع یادگیری طراحی وب در ۱ ساعت می باشد که در آن بیشتر از ۲۵ قاعده ساده و دستورالعمل و تعداد زیادی منبع شگفت انگیز موجود است!
چیزهایی که در این دوره یاد می گیرید از این قرار است:

- چگونه متن را حرفه ای طراحی کنیم؟
- چگونه از قدرت رنگ ها استفاده کنیم؟
- برای ساخت وب سایت، چگونه از تصاویر، فونت ها و آیکون های متحرک‌کننده استفاده کنیم؟
- چگونه انگیزه یاد گرفتن بیشتر طراحی وب را در خود افزایش دهیم؟
- چگونه با استفاده از ۸ تکنیک ساده، کاری کنیم وب سایت بهتر convert شود؟

۶. Web Design for Everybody: Basics of Web Development & Coding Specialization:

این یکی از منابع رایگان یادگیری مفاهیم پایه ای توسعه وب و کد زدن در Coursera می باشد. در این دوره ها، HTML۵، CSS۳، نحوه نوشتن درست و با قاعده آنها و ساخت تجربه وب تعاملی با JavaScript آموزش داده می شود.
این منبع، ۵ دوره را شامل می شود:

۱. Introduction to HTML۵
۲. Introduction to CSS۳
۳. Interactivity with JavaScript
۴. Advanced Styling with Responsive Design
۵. Web Design for Everybody Capstone

در نهایت به عنوان بخشی از این تخصص Coursera، یک پورتفولیو وب با کیفیت بسیار بالا برای اثبات توانایی و دانش خود در توسعه و طراحی وب خواهید ساخت.

لینک دوره‌ها به ترتیب:

۱. https://www. udemy. com/course/web-development-learn-by-doing-html-۵ css-۳from-scratch-introductory/
۲. https://www. udemy. com/course/foundations-of-front-end-development/
۳. https://www. udemy. com/course/how-i-landed-a-web-development-job-earned۵-k-freelancing/
٤. https://www. udemy. com/course/master-the-basics-of-html-۵css-۳ beginner-web-development/
۵. https://www. udemy. com/course/web-design-secrets/
٦. https://www. coursera. org/specializations/web-design

پروتکل SSH چیست؟

امیرحسین زین‌الدینی
ورودی ۹۷

پوسته امن (Secure Shell) یا به اختصار SSH یک پروتکل اینترنتی است که امکان تبادل اطلاعات با استفاده از یک کانال امن را بین دستگاه های متصل در شبکه ایجاد می کند. و در سال ۱۹۹۵ توسط تاتو یلونن، محقق دانشگاه فنی هلسینکی فنلاند به وجود آمد.

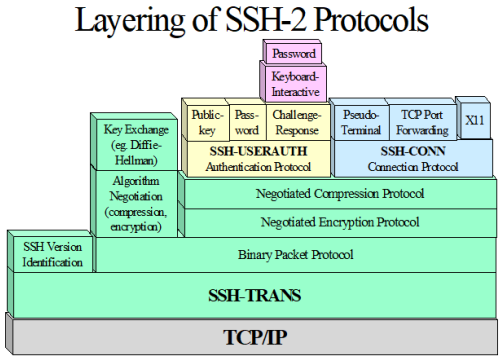
SSH احراز هویت و رمزگذاری را برای برنامه های مهم تجاری برای کارکرد امن آنها از طریق اینترنت فراهم می کند و دو نسخه اصلی این پروتکل به نام های SSH۱ و SSH۲ شناخته می شود. که در ابتدا بر روی سیستم های مبتنی بر یونیکس و لینوکس برای دسترسی به حساب‌های پوسته استفاده شد، SSH جایگزینی برای telnet و سایر پوسته‌های ارتباط از راه دور غیر امن همچون rcp ، rlogin ، rsh ایجاد شد.
بر روی پروتکل TCP / IP تعریف شده و روی پورت ۲۲ اجرا می شود.
به عنوان یک تعریف بسیار ساده می‌توان SSH را این گونه بیان کرد که: SSH یک پروتکل قدرتمند و پراستفاده نرم افزاری است که برای دستیابی به امنیت شبکه طراحی شده‌است.
سه رکن امنیت در ارتباط از راه دور که در این پروتکل مورد توجه قرارگرفته که عبارت اند از محرمانگی (confidentiality) ، درستی (integrity) و تصدیق هویت (authentication). اکثر فناوری های پیشین فاقد عناصر محرمانگی و درستی بودند. به عنوان مثال FTP نام کاربری و رمزهای عبور را در متن (به شکل رمزنگاری نشده) منتقل می کنند. علاوه بر این آنها در برابر حملاتی مانند

اتصال در پروتکل SSH-1

IP spoofing, DoS, MITM و استراق سمع آسیب پذیر بودند.
پوسته امن هر سه نیاز ذکر شده را با استفاده از موارد زیر مرتفع می کند.
رمزگذاری داده ها برای ایجاد محرمانگی
احراز هویت مبتنی بر میزبان (Host-based) و یا مشتری محور (client-based)
برسی صحت داده ها با استفاده از MAC و hash
نسخه های مختلف SSH
دو نسخه کاملاً متفاوت از پروتکل SSH وجود دارد: SSH-۱ و SSH-۲
SSH-۲ به منظور رفع چندین اشکال در SSH-۱ معرفی شد و همچنین دارای چندین ویژگی جدید بود.
هر دو نسخه دارای اشکالات فنی هستند و در برابر حملات شناخته شده آسیب پذیر هستند (در ادامه بحث خواهد شد)
OpenSSH یک نسخه رایگان از SSH با کد منبع باز است. این از هر دو نسخه پشتیبانی می کند و عمدتا در محیط UNIX استفاده می شود.
۱- SSH پروتکلی یکپارچه است ، شامل چندین تابع در قالب یک پروتکل واحد. در حالی که SSH-۲ به ماژول های مختلفی تقسیم شده و از ۳ پروتکل شکل گرفته است:

- پروتکل لایه انتقال SSH (SSH-TRANS): اتصال اولیه ، کنترل بسته ها ، تأیید اعتبار سرور و سرویس های رمزگذاری و برسی درستی اطلاعات را به عهده دارد
- پروتکل احراز هویت SSH (SSH-AUTH): با استفاده از سه روش احراز هویت: کلید عمومی ، میزبان محور و رمز ورود ، هویت کلاینت را در ابتدای جلسه SSH-۲ تأیید می کند.
- پروتکل اتصال SSH (SSH-TRANS): به تعدادی از سرویس های مختلف اجازه می دهد تا داده ها را از طریق کانال ایمن ارائه شده توسط SSH-TRANS مبادله کنند.

پروتکل SSH-2 (ادامه)



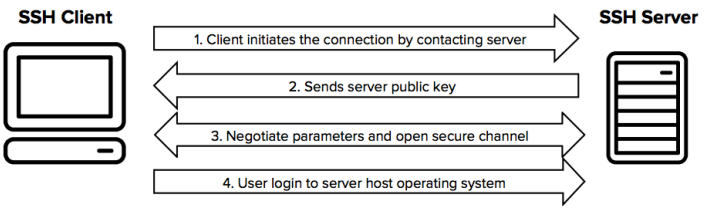
شمای گرافیکی از لایه بندی های پروتکل SSH-2.

SSH چگونه کار می کند؟

به طور خلاصه ، هر بار که داده‌ای از طرف کامپیوتر به شبکه فرستاده می شود، به صورت خودکار توسط SSH رمزنگاری می شود. هنگامی که داده به مقصد خود می‌رسد به صورت خودکار رمزگشایی می شود.
نتیجه‌ای که خواهد داشت کدگذاری نامرئی خواهد بود.
بدین صورت کاربران نهایی درگیر پروسه کدگذاری و کدگشایی نخواهند شد و از ارتباط امن خود می‌توانند به خوبی استفاده کنند. امنیت سیستم کدگذاری SSH با استفاده از الگوریتم‌های پیچیده و مدرن تضمین می شود. تا آنجا که امروزه در سیستم‌های حیاتی و بسیار حساس از این سیستم استفاده می شود. به صورت
معمول محصولاتی که از SSH استفاده می کنند از دو بخش کلاینت / سرور تشکیل می شوند. کلاینت با استفاده از تنظیمات سرور مربوط به آن وصل می شوند و سرور وظیفه تأیید هویت و قبول یا رد ارتباط را به عهده دارد.
همان گونه که گفتیم این پروتکل در مدل سرویس گیرنده سرویس دهنده کار می کند ، به این معنی که اتصال توسط سرویس گیرنده SSH که به سرور SSH متصل می شود برقرار می شود. سرویس گیرنده SSH فرایند تنظیم اتصال را هدایت می کند و از رمزنگاری کلید عمومی برای تأیید هویت سرور SSH استفاده می کند.
پس از مرحله راه اندازی ، پروتکل SSH از رمزنگاری متقارن قوی و الگوریتم های هش برای اطمینان از حریم خصوصی و درستی داده هایی که بین کلاینت و سرور رد و بدل می شود، استفاده می کند.

آیا پروتکل SSH کاملا ایمن است؟

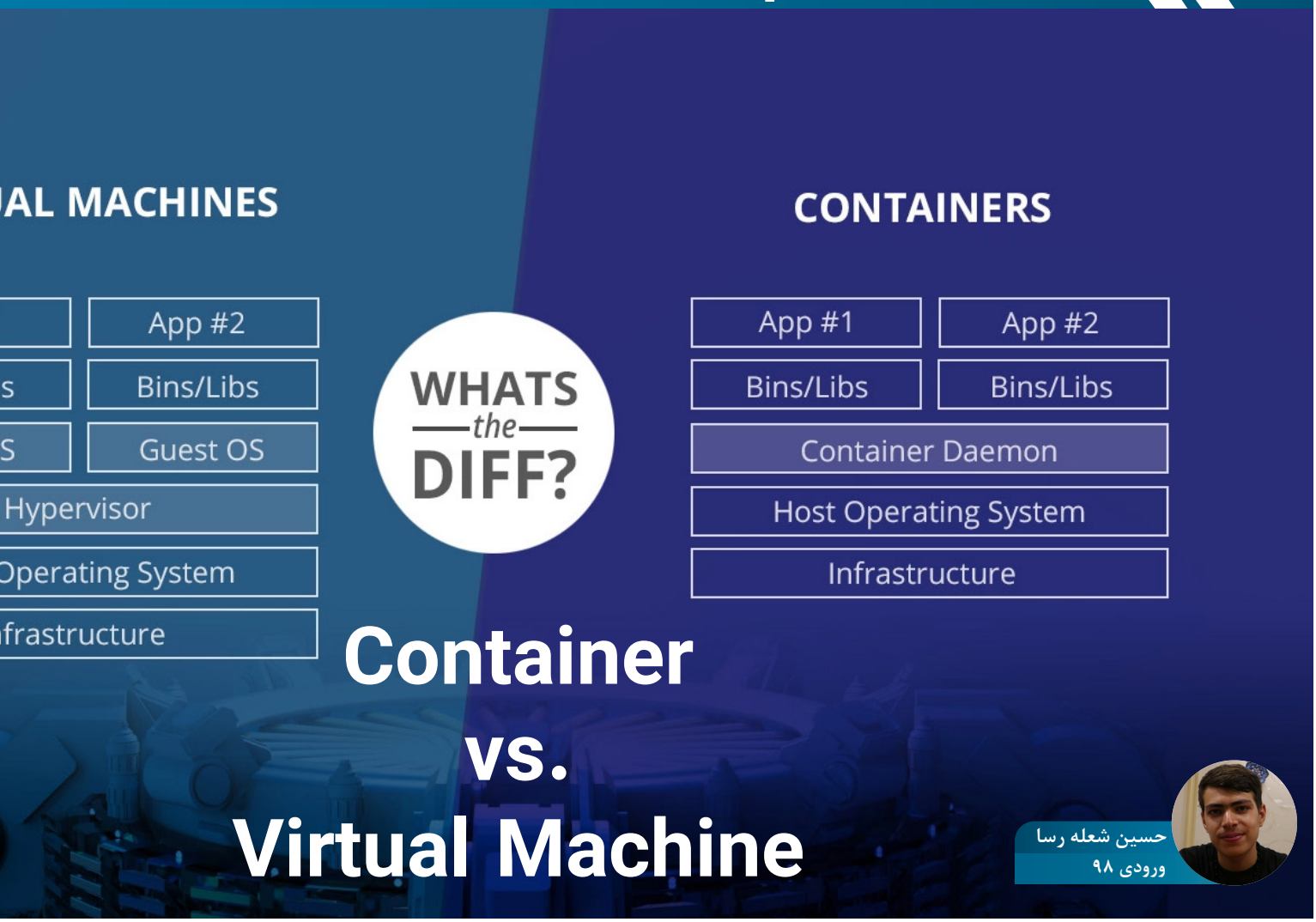
SSH توانایی مقابله با بسیاری از حملات را دارد همچون: Eavesdropping ، Name service ، IP spoofing ، Connection hijacking ، Insertion attack ، MITM ولی در برابر حملات زیر کارایی خود را از دست می دهد:



کرک پسورد، حملات IP و TCP مانند: SYN flooding، TCP RST، bogus ICMP ، TCP desynchronization and hijacking

منبع:
جزوات استاد هوشمند

momtazserver. com
maralhost. com
itbaz. net



حسین شعله رسا
ورودی ۹۸

ماشین مجازی یا همان Virtual Machine تفاوت زیادی با یک کامپیوتر فیزیکی مانند لپ تاپ, گوشی هوشمند یا حتی سرور ندارد. همانند کامپیوتر های دیگر حاوی cpu, ممووری و دیسک برای ذخیره است و حتی در صورت لزوم میتواند به اینترنت هم متصل بشود. تنها تفاوت این است که کامپیوترهای ما فیزیکی و محسوس هستند در حالیکه ماشینهای مجازی, مجازی هستند.

مجازی سازی فرایند ساخت یک ورژن مجازی از محیط کامپیوتری که داریم می باشد که cpu, ممووری, و حافظه را از کامپیوتر یا سرور میزبان قرض میگیرد. به عبارتی ماشین مجازی یک فایل است که به آن image گفته می شود و مانند کامپیوتری مستقل عمل می کند. این کامپیوتر مجازی مستقل, میتواند به عنوان یک محیط محاسباتی جداگانه وجود داشته باشد که روی آن سیستم عامل های مختلف اجرا می شوند. ماشین مجازی جدا از بقیه بخش های سیستم عمل می کند; به این معنی که نرم افزاری که داخل یک ماشین مجازی در حال اجراست ارتباطی با ماشین های مجازی دیگر یا حتی کامپیوتر یا سرور میزبان ندارد.

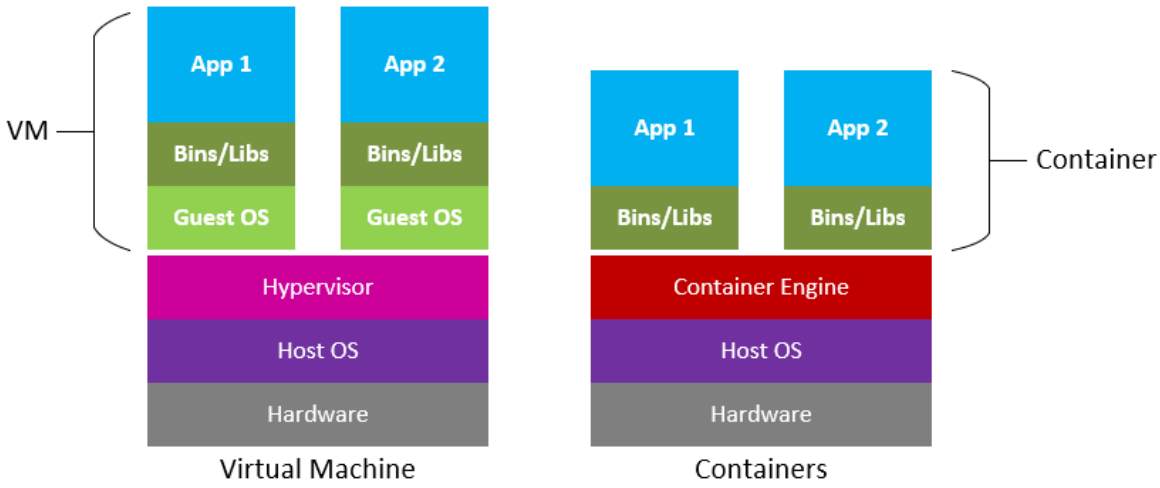
ماشین مجازی کاربردهای زیادی دارد و ممکن است برحسب نیازهای مختلفی مورد استفاده قرار بگیرد; از جمله آن میتوان به استفاده از سیستم عامل های دیگر به عنوان میزبان, تست اپلیکیشن ها و نرم افزارها در محیطی امن و ایزوله, بک-آپ گرفتن از سیستم عامل فعلی, دسترسی به دیتاهایی که ویروسی شده اند یا استفاده از اپلیکیشن ها و نرم افزارهای قدیمی تر با نصب ورژن قدیمی سیستم عاملی دیگر و... اشاره کرد .

از بزگترین مزایایی که ماشین مجازی دارند استقلال آنها از بقیه ماشین های مجازی و سیستم میزبان است. نرم افزاری به نام hypervisor یا virtual machine manager اجازه میدهد در آن واحد چند سیستم عامل روی یک سیستم اجرا شوند و تداخلی با هم نداشته باشند که این منجر به استفاده بهینه از فضای فیزیکی و زمان می شود. این استقلال به آنها قابلیت قابل حمل بودن را نیز اضافه می کند.

در این بین مفهوم دیگری با عنوان container مطرح است که گاها با ماشین مجازی اشتباه گرفته می شود.

همانند ماشین های مجازی, container ها نیز اپلیکیشن ها و نرم افزارهای مختلف را به صورت ایزوله و محافظت شده از بقیه ی منابع اجرا می کنند با این تفاوت که ماشین مجازی بخشی از سخت افزار را برای راه اندازی محیطی کاملا جداگانه در اختیار میگیرد ولی کانتینر فقط همان نرم افزار خاص و وابستگی هایش را از بقیه ی سیستم جدا می کند و همچنین کانتینرهای مختلف سیستم عامل میزبان را باهم به صورت اشتراکی استفاده می کنند درحالیکه ماشین های مجازی هر کدام مجاز بودند سیستم عاملی مجزا داشته باشند.

مزیت کلیدی کانتینر ها نسبت به ماشین های مجازی این است که سربار خیلی کمتری به نسبت ماشین های مجازی دارند از این جهت که فقط حاوی فایل های باینری و کتابخانه ها و وابستگی های مخصوص به همان نرم افزاری که اجرایش می کنند هستند, کرنل یک سیستم عامل میزبان را به صورت اشتراکی استفاده می کنند که این مزیت, آنها را نسبت به ماشین های مجازی کوچکتر می کند و درنتیجه ی آن سریع تر بوت می شوند, استفاده ی بهینه ای از منابع دارند و همچنین



قابلیت انتقال نرم افزار ها و اپلیکیشن ها را بیشتر می کنند.

برعکس آن ماشین های مجازی بزرگتر هستند و نسبت به کانینرها کندتر بوت می شوند, آنها کاملا از یکدیگر جدا هستند و هرکدام کرنل سیستم عامل مخصوص خود را دارند. ماشین های مجازی برای اجرای چند اپلیکیشن با یکدیگر, ایجاد محیطی ایزوله بین اپلیکیشن های مختلف و اجرای برنامه های قدیمی که روی سیستم عامل های با ورژن قدیمی تر اجرا می شوند بهترین انتخاب هستند.

داکر مفهوم کانتینر را ارایه میدهد و همانطور که گفته شد استفاده از ماشین مجازی برای اجرای یک برنامه سربار بسیار زیادی دارد و اصلا مقرون به صرفه نیست, با این حال داکر در سیستم عامل های macOS و windows به صورت یک ماشین مجازی کوچک ارایه می شود, البته اگر دقیق تر به آن نگاه کنیم, ما یک ماشین مجازی کوچک ایجاد میکنیم که توسط آن تمام کانتینر ها را اجرا میکنیم و هرچه تعداد کانتینر ها زیاد می شوند سربار استفاده از ماشین مجازی نیز کمتر می شود.

منابع: [virgool. io](virgool.io) [medium. com](medium.com)

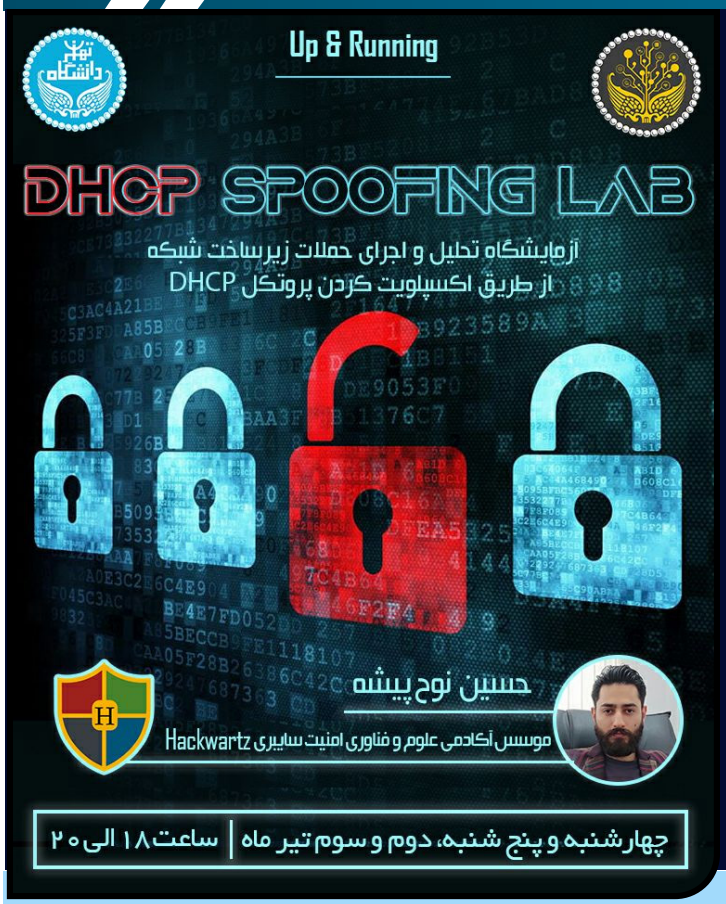
فعالیت‌های انجمن علمی

محمدحسین شیرازی
ورودی ۹۷

نوینوس با همکاری انجمن علمی مهندسی کامپیوتر پردیس فارابی دانشگاه تهران برگزار می کند.
- وبینار تخصصی آشنایی با بازی سازی

نتورکینگ ایونت دانشجویان علوم و مهندسی کامپیوتر ایران
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس

همایش رایگان و تخصصی آمادگی در کنکور ارشد ۱۴۰۱
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر



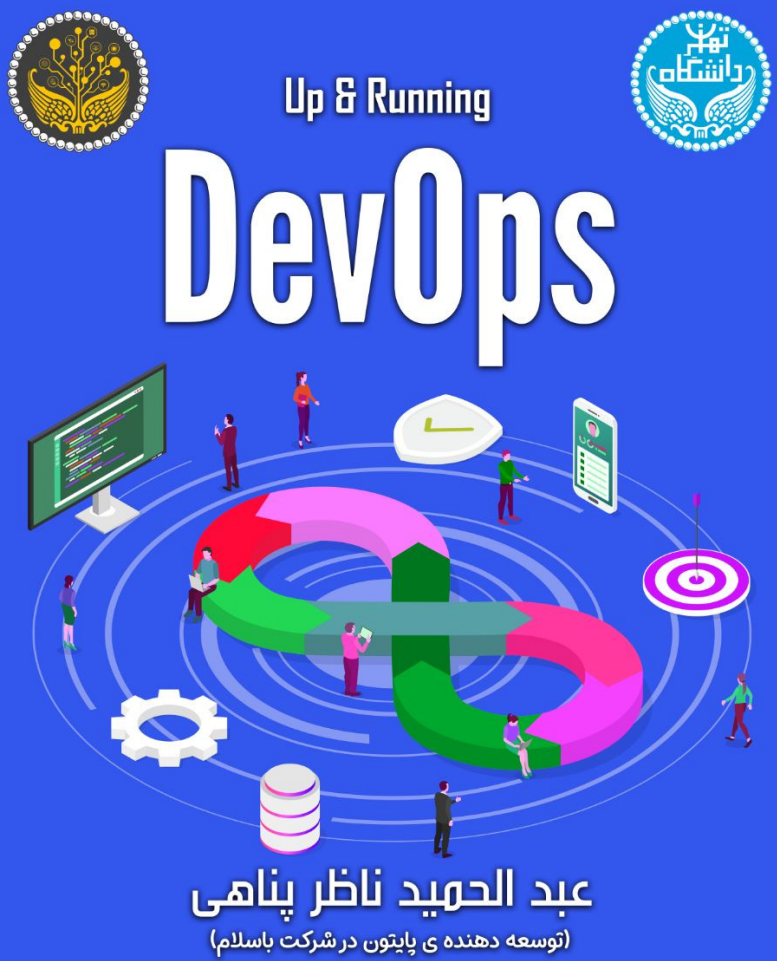
کارگاه دو روزه با موضوع شبکه و امنیت
ارائه دهنده : حسین نوح پیشه، موسس آکادمی علوم و فناوری امنیت سایبری Hackwartz

برگزاری سلسله جلسات مجازی انتقال تجربه از شرکت های دانش بنیان به دانشجویان
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر

لایو اینستاگرامی انجمن های علمی کامپیوتر با آکادمی امنیت سایبری راوین

با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر

فعالیت‌های انجمن علمی



چهارمین جلسه از سری جلسات Up&running، با موضوع « DevOps» توسط انجمن علمی مهندسی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران برگزار می شود.

ارائه دهنده : عبدالحمید ناظرپناهی؛ توسعه دهنده پایتون در شرکت باسلام و دانشجوی مهندسی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران

لایو اختصاصی با پارک علم و فناوری تربیت مدرس
موضوع: بحث و گفتگو پیرامون #سومین_دوره_طرح_توانمندسازی_صدف
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر

وبینار ناگفته های نقشه راه امنیت سایبری
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر

مرکز مشاوره دانشگاه صنعتی قم
انجمن علمی مهندسی کامپیوتر دانشگاه صنعتی قم
با همکاری انجمن علمی مهندسی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران برگزار می کنند:

کارگاه امتحاناتت را قورت بده

سلسله جلسات مجازی انتقال تجربه از شرکت های دانش بنیان به دانشجویان
با همکاری انجمن های علمی دانشجویی کامپیوتر دانشکده مهندسی پردیس فارابی دانشگاه تهران، خواجه نصیرالدین طوسی تهران، ارشاد دماوند تهران، توس مشهد، گیلان، شهید باهنر کرمان، بین المللی امام خمینی(ره) قزوین و خلیج فارس بوشهر

برگزاری «سومین وبینار تخصصی بینایی ماشین و پردازش تصویر ایران « با همکاری انجمن بینایی ماشین و پردازش تصویر ایران و دانشکده مهندسی پردیس فارابی دانشگاه تهران؛ با حضور دکتر حسین عزیزپور

برگزاری «چهارمین وبینار تخصصی بینایی ماشین و پردازش تصویر ایران « با همکاری انجمن بینایی ماشین و پردازش تصویر ایران و دانشکده مهندسی پردیس فارابی دانشگاه تهران؛ با حضور دکتر مهدی رضایی

برگزاری «دومین وبینار تخصصی بینایی ماشین و پردازش تصویر ایران» با همکاری انجمن بینایی ماشین و پردازش تصویر ایران و دانشکده مهندسی پردیس فارابی دانشگاه تهران؛ با حضور پروفسور شهره کسائی استاد تمام دانشکده مهندسی کامپیوتر دانشگاه صنعتی شریف

Feng. ut. ac. ir

http://ismvip. ir

آزمایشگاه پژوهشی فضای سایبر :

سلسله نشست های علمی با موضوع Corona+X نشست اول:

عنوان: «طبیعه نوع جدید حکومت بر جهان» با سخنرانی دکتر امیر اشرفی رئیس دپارتمان آناتومیکال پاتولوژی بیمارستان Nepean سیدنی استرالیا

نشست دوم: عنوان : «بررسی اسناد آژانس‌های ایالات متحده آمریکا در پدیده کرونا» با سخنرانی جناب آقای محمد دلخوش رئیس اندیشکده هجرت

نشست سوم: عنوان : «فعل‌ها و ترک فعل‌های منجر به واردات واکسن‌های ممنوعه» با سخنرانی دکتر جواد جاویدنیا معاون قضایی دادگستری کل استان تهران در پیگیری امور فضای مجازی

عنوان: «کرونا و تغییرات حاصل از آن در روند زندگی ایرانیان و سایر مردم جهان (نگاه تطبیقی)» با سخنرانی مهندس عاطفه نصیری کارشناس ارشد مهندسی منابع طبیعی و پژوهشگر در اسناد و منابع بین المللی

نشست پنجم: عنوان : «واکسن‌های ژنی، راه پیشگیری یا سلاح‌های بیولوژیک: بررسی واکسن‌های نسل جدید و خطرات آن» با سخنرانی دکتر امیر اشرفی رئیس دپارتمان آناتومیکال پاتولوژی بیمارستان Nepean سیدنی استرالیا

برگزاری « کارگاه آموزشی الگوریتم بهینه سازی کشتل » توسط دانشکده مهندسی پردیس فارابی دانشگاه تهران؛ با حضور دکتر مصطفی حاجی آقایی کشتلی استاد دانشگاه صنعتی مونتری مکزیک

با دومین سری مسابقات نشریه در خدمت شما هستیم! در شماره قبلی نشریه ۱۰ سوال مطرح شد که جناب آقای سید محمد صالح قطبانی با پاسخ صحیح به اکثر سوالات تونستن نفر اول بشن و جایزه نفیسی از طرف نشریه دریافت خواهند کرد! در ادامه به سراغ سوالات این شماره می رویم!

سوالات هوش و الگوریتم

دانسته هات رو به چالش بکش و پاسخش رو برامون ارسال کن!

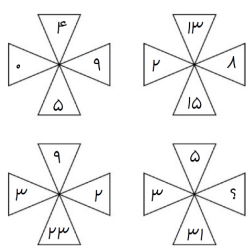
عارف برهانی
ورودی ۹۸

سوال ۱: بین اعداد زیر، از چپ به راست ارتباط خاصی برقرار است. به جای علامت سوال، چه عددی باید قرار گیرد؟

؟، ۶۱۳۰ ، ۵۷۳۴ ، ۵۱۴۰ ، ۴۳۴۸

۱) ۶۴۲۷
۲) ۶۳۲۸
۳) ۶۵۲۰
۴) ۶۵۲۶

سوال ۲: بین اعداد داخل هر شکل، ارتباط خاصی برقرار است، به جای علامت سوال، کدام عدد باید قرار داده شود؟



۱) ۸
۲) ۹
۳) ۷
۴) ۶

سوال ۳: ثلث مزرعه ای را یک پدر و پسر با هم در دو ساعت درو می کنند. پدر از پسر جدا شده و پسر به تنهایی نصف قسمت درو نشده را در ۵ ساعت درو می کند و سپس جای خود را به پدرش می دهد. پدر باقی مانده زمین را به تنهایی در چه مدت زمانی درو خواهد کرد؟

۱) ۱ ساعت و ۱۵ دقیقه
۲) ۱ ساعت و ۲۰ دقیقه
۳) ۳ ساعت و ۲۰ دقیقه
۴) ۳ ساعت و ۴۵ دقیقه

سوال ۴: ۱۴ عدد متمایز داریم که اگر آن ها را از کوچک به بزرگ مرتب کنیم، اختلاف هر دو عدد متوالی برابر ۶ می شود. میانگین این اعداد برابر ۷۵ می باشد. کوچک ترین عدد کدام است؟

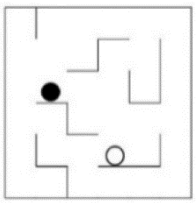
۱) ۴۲
۲) ۴۸
۳) ۳۶
۴) نمی توان تعیین کرد.

سوال ۵: شخصی برای خرید ۲ کیلو سیب و ۵ کیلو پرتقال، ۲۴۰۰۰ تومان می پردازد. اگر وی اشتباها ۵ کیلو سیب و ۲ کیلو پرتقال خریداری کند، هزینه او ۶۰۰۰ تومان کاهش می یابد. قیمت هر کیلو پرتقال چند هزار تومان است؟

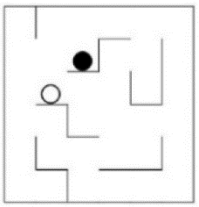
۱) ۵ ۱. ۲) ۴ ۳) ۲ ۴) ۳

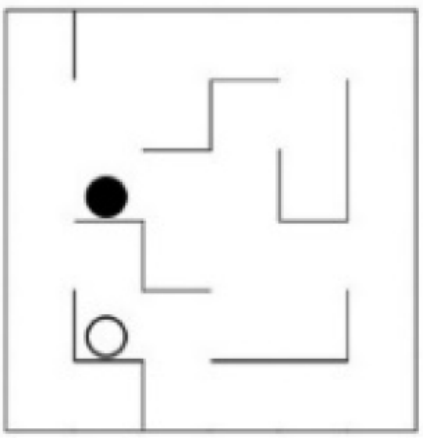
سوال ۶: در سوال زیر، پازلی حاوی دو گوی سفید و مشکی ارائه شده که در آن، پس از هر چرخش ۹۰ درجه ای، گوی ها فقط به دلیل دزنی که دارند، سقوط کرده و بدون لغزش در محل سقوط، ثابت می مانند. بعد از این که این پازل، سه دور کامل (هر دو چهار چرخش) در جهت حرکت عقربه های ساعت بچرخد، کدام طرح محل قرار گرفتن گوی ها را پس از دوران، به درستی نشان می دهد؟

الف)

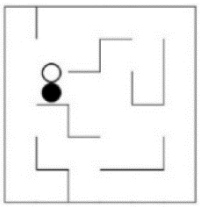


ب)

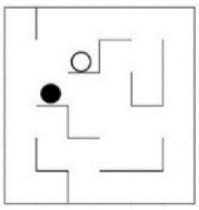




ج)



د)



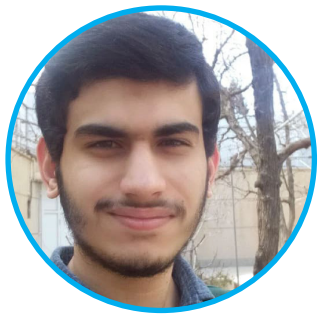
نحوه ارسال پاسخ‌ها:

پس از تامل برروی سوالات و رسیدن به یکی از گزینه‌ها، کافیه وارد کانال اطلاع‌رسانی انجمن بشید و پاسخ‌هاتون رو به آیدی دبیر انجمن (@CESA_PR) ارسال فرمایید!

اسامی برندگان در شماره بعدی اعلام خواهد شد!

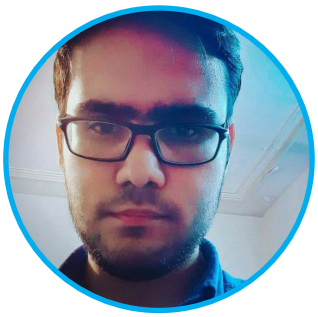
همکاران این شماره

محمدعلی آصف
مدیرمسئول، طراحی و گرافیک، نویسنده



سیدمحمدصالح قطبانی
طراحی و صفحه‌آرایی، کتاب الکترونیک

مهران آدووند
سردبیر، نویسنده



محمدحسین شیرازی
نویسنده

دانیال فرهنگی
نویسنده

عارفه عطایی نظری
نویسنده

فائزه خادم
نویسنده

علیرضا زینی جلودار
نویسنده

مهدی ناصری مجد
نویسنده

احمدرضا پارسی‌زاده
نویسنده

صبا حیدری دوست
نویسنده

امیرعلی خانه عنقا
نویسنده

امیرحسین زین الدینی
نویسنده

عارف برهانی
نویسنده

حسین شعله رسا
نویسنده

علی اسدی
نویسنده

همکاری در نشریه و انجمن