



دانشکده فنی و مهندسی
پایان نامه جهت دریافت درجه کارشناسی
گروه کامپیوتر

بلاکچین و رمزارز تایکی Tyche Blockchain and Cryptocurrency

تحقیق و نگارش:
آریا رادمهر

استاد راهنما:
دکتر سجاد حق‌زاد کلیدبری

۱۴۰۱ تیر

حَالِهِ
لِمَحْسُونٍ

فرم نهایی پروژه کارشناسی - گروه کامپیوتر دانشگاه زنجان

	نام و نام خانوادگی دانشجو
	شماره دانشجویی
	عنوان پروژه
	نمره نهایی به عدد
	نمره نهایی به حروف
	تاریخ دفاع
	نام و امضای استاد راهنما
	نام و امضای مدیر گروه

تقدیم نامه

این پایان نامه را تقدیم به مادر و پدر عزیزتر از جانم، این دو موهبت الهی می‌کنم که بدون حمایت و زحمات آنها، روپرتو شدن با هیچ چالشی برایم میسر و شدنی نبود.

سپاس نامه

با سپاس از استاد گرانقدرم جناب آقای دکتر سجاد حق‌زاد کلیدبری که مرا در این راه یاری نمودند.

فهرست

xii	چکیده
۱	فصل ۱. پیشگفتار
۲	۱-۱. مقدمه
۲	۱-۲. چالش‌های پیش رو
۳	۱-۳-۱. اهداف پایان نامه
۴	۱-۴-۱. کارهای مشابه
۴	۱-۵. ساختار پایان نامه
۵	فصل ۲. مفاهیم پایه
۶	۲-۱. مقدمه
۶	۲-۲. درکی درباره بلاکچین
۶	۲-۲-۱. نگاه تاریخی
۹	۲-۲-۲. تعریف استاندارد و آکادمیک
۱۰	۲-۳. ویژگی‌ها و نقاط مثبت
۱۰	۲-۳-۱. تعاملی بودن سیستم‌ها بر بستر بلاکچین
۱۱	۲-۳-۲. شفافیت
۱۲	۲-۳-۳. امنیت
۱۳	۲-۴. مفاهیم بلاک‌ها
۱۳	۲-۴-۱. داده
۱۴	۲-۴-۲. هش بلاک
۱۶	۲-۴-۳. هش بلاک قبل
۱۷	۲-۵. امضاهای امضاهای دیجیتال

۱۹	۶-۲. عدم تمرکز.....
۲۲	۷-۲. بحث بر اجماع.....
۲۳	۱-۷-۲. مسئله ژنرال‌های بیزانسی.....
۲۵	۲-۷-۲. اجماع ضمنی.....
۲۵	۲-۸. اثبات انجام کار.....
۳۱	۹-۲. استخراج.....
۳۵	۱۰-۲. درک تراکنش‌ها.....
۳۸	۱۱-۲. نگهداری‌ها.....
۳۸	۱۱-۲. فضاهای ذخیره سازی متصل به شبکه یا دستگاه.....
۴۰	۱۱-۲. فضاهای ذخیره سازی ایزوله.....
۴۱	۱۲-۲. نتیجه گیری.....
۴۲	فصل ۳. تکنولوژی‌های بکار رفته و ابزارها
۴۳	۱-۳. مقدمه.....
۴۳	۲-۳. ابزارهای مورد استفاده.....
۴۳	۱-۲-۳. جاوا اسکریپت.....
۴۴	۲-۲-۳. کتابخانه ری اکت.....
۴۴	۳-۲-۳. چارچوب جست.....
۴۴	۲-۲-۳. نرم افزار پستمن.....
۴۵	۳-۲-۳. فرآیند توسعه.....
۴۶	۴-۳. پارادایم برنامه نویسی.....
۴۶	۳-۵. نتیجه گیری.....
۴۷	فصل ۴. پیاده سازی
۴۸	۱-۴. مقدمه.....
۴۸	۲-۴. ساخت بخش اول (بلاک‌ها)
۵۰	۳-۴. ساخت بخش دوم (بلاکچین)

۴-۴. ساخت بخش سوم (اثبات انجام کار).....	۵۲
۴-۵. ساخت بخش چهارم (پیاده سازی شبکه و رابط برنامه نویسی کاربردی).....	۵۷
۴-۶. ساخت بخش پنجم (کیف پول، کلیدها و تراکنش‌ها).....	۵۹
۴-۷. ساخت بخش ششم (استخراج تراکنش‌ها).....	۶۲
۴-۸. ساخت بخش هفتم (استخراج تراکنش‌ها).....	۶۷
۴-۹. ساخت بخش هشتم (کار بر روی بخش جلویی).....	۷۳
۴-۱۰. نتیجه گیری.....	۸۱
فصل ۵. ارزیابی نتایج	۸۲
۵-۱. مقدمه.....	۸۳
۵-۲. ارزیابی‌ها از خروجی برنامه.....	۸۳
۵-۳. نتیجه گیری.....	۸۸
فصل ۶. مطالب آتی	۸۹
۶-۱. مقدمه.....	۹۰
۶-۲. ایده‌های پیش رو.....	۹۰
۶-۳. نتیجه گیری.....	۹۱
فصل ۷. منابع و مأخذ	۹۲
Abstract	۹۴

فهرست اشکال و جداول

شکل ۱-۱. شماتیک سیستم‌های متتمرکز، غیر متتمرکز و تعاملی.....	۱۱
شکل ۱-۲. شماتیک یک بلاک به صورت ساده.....	۱۳
شکل ۲-۲. زنجیره‌ای از بلاک‌ها و ارتباط آنها توسط هش بلاک قبل.....	۱۶
شکل ۳-۲. شماتیک ساده یک سکه به همراه سریال و هش.....	۱۹
شکل ۴-۲. چگونگی جایی مالکیت سکه‌ها توسط افراد.....	۲۰
شکل ۵-۲. جایی سکه‌ها و سیاهه دارندگان آنها توسط دفتر کل.....	۲۱
شکل ۶-۲. شماتیک مسئله ژنرال‌های بیزانسی.....	۲۳
شکل ۷-۲. ساختار یک بلاک به صورت بخش به بخش.....	۲۷
شکل ۸-۲. فرمول شرط پذیرش یک بلاک در بلاکچین.....	۲۷
شکل ۹-۲. ساختار کلی یک بلاک با تمامی جزئیات.....	۲۸
شکل ۱۰-۲. گره‌های یک شبکه در سناریوهای اثبات انجام کار.....	۲۹
شکل ۱۱-۲. گره‌های یک شبکه بعد از همگام سازی.....	۳۰
شکل ۱۲-۲. فرآیند خلاصه شده به صورت ریاضی از فرآیند استخراج.....	۳۲
شکل ۱۳-۲. زنجیره بلاک‌های استخراج شده.....	۳۲
شکل ۱۴-۲. شماتیک سناریوی کاهش و افزایش سرعت استخراج شبکه توسط گره‌ها.....	۳۳
شکل ۱۵-۲. فرمول تخمین میزان سختی بعدی در شبکه.....	۳۵
شکل ۱۶-۲. مثالی برای تخمین میزان سختی بعدی شبکه.....	۳۵
شکل ۱۷-۲. تاریخچه انتقالات سکه‌ها به صورت سیاهه‌ای از تراکنش‌ها.....	۳۶
شکل ۱۸-۲. تاریخچه تراکنش‌ها بر اساس ورودی و خروجی‌ها.....	۳۷
شکل ۱۹-۲. شماتیک یک تراکنش در بلاکچین به همراه ورودی و خروجی‌ها و امضای دارنده سکه.	۴۰
شکل ۲۰-۲. شماتیک فرآیند توسعه آزمایش محور نرم افزار.....	۴۵

.....	شکل ۱-۴. فرآیند تست استخراج بلاک‌ها قبل از تعادل در زمان استخراج	۵۵
.....	شکل ۲-۴. فرآیند تست استخراج بلاک‌ها پس از تعادل در زمان استخراج	۵۶
.....	شکل ۳-۴. انجام یک تراکنش در پستمن	۶۴
.....	شکل ۴-۴. انجام تراکنش بدون داشتن سکه کافی در پستمن	۶۵
.....	شکل ۴-۵. خروجی استخراج تراکنش‌ها در پستمن	۶۶
.....	شکل ۴-۶. خروجی‌های نقل و انتقالات سکه‌ها در قالب یک تراکنش در پستمن	۶۹
.....	شکل ۴-۷. محتوای استخراج تراکنش‌ها با وجود تراکنش مورد نظر در پستمن	۷۰
.....	شکل ۴-۸. ثبت بلاک با نقل و انتقالات مورد نظر در بلاکچین در پستمن	۷۱
.....	شکل ۴-۹. شماتیک خالی بودن استخراج بعد از ثبت تراکنش‌ها در بلاک در پستمن	۷۲
.....	شکل ۱۰-۴. نمایش اطلاعات کیف پول در پستمن	۷۲
.....	شکل ۱۱-۴. شماتیک اجرای گره شماره یک در ترمینال	۷۴
.....	شکل ۱۲-۴. شماتیک محیط کاربری گره شماره یک در مرورگر	۷۴
.....	شکل ۱۳-۴. شماتیک اجرای گره شماره دو در ترمینال	۷۵
.....	شکل ۱۴-۴. شماتیک محیط کاربری گره شماره یک در مرورگر	۷۵
.....	شکل ۱۵-۴. وارد کردن اطلاعات یک تراکنش از گره یک به گره دو	۷۶
.....	شکل ۱۶-۴. اعلان موفقیت آمیز بودن ثبت تراکنش	۷۶
.....	شکل ۱۷-۴. نمایش استخراج تراکنش‌ها به صورت زنده	۷۷
.....	شکل ۱۸-۴. صفحه بلاک‌های ثبت شده در بلاکچین	۷۷
.....	شکل ۱۹-۴. شماتیک بلاک در بلاکچین با جزئیات بیشتر	۷۸
.....	شکل ۲۰-۴. اطلاعات کیف پول گره شماره یک بعد از ثبت تراکنش‌ها	۷۹
.....	شکل ۲۱-۴. اطلاعات کیف پول گره شماره دو بعد از ثبت تراکنش‌ها	۷۹
.....	شکل ۲۲-۴. اطلاعات دریافت بلاکچین جدید توسط گره شماره دو توسط شبکه و جایگزینی آن	۸۰
.....	شکل ۲۳-۴. استخراج تراکنش‌ها بعد از ثبت بلاک در شبکه	۸۰
.....	شکل ۱-۵. خروجی اولیه تست‌های برنامه در ترمینال	۸۴
.....	شکل ۲-۵. نمای اولیه و اصلی از وبسایت پروژه	۸۵

.....
.....
.....
.....
.....

چکیده

در دنیای پیش رو بلاکچین‌ها یک تکنولوژی تازه و نو در عرصه علوم کامپیوتر به حساب می‌آیند. این تکنولوژی نخست در سال ۱۹۹۱ توسط عده‌ای از محققین کشف شد و بعد از آنکه ساتوشی ناکاموتو در سال ۲۰۰۹ بیت‌کوین را بوجود آورد، توجه دنیا را به خود جلب کرد. بلاکچین به زبان ساده همانند مجموعه‌ای از اطلاعات می‌باشد که بر بستر سیستمی توزیع شده نگهداری می‌شود.

شناخت بلاکچین‌ها در انواع و اقسام مختلف چه به صورت منحصر به فرد و چه به صورت کمکی و یا ابزاری در سیستم‌ها و کسب و کارها دارای اهمیت زیادی می‌باشد، این شناخت در کنار پیاده سازی یک سیستم بلاکچین به همراه یک قابلیت کاربردی بر بستر آن (ارز دیجیتال در این پایان نامه) می‌تواند به هر شخصی شناخت لازم از کارکرد یک سیستم بلاکچینی و همچنین درک لازم در مورد ارزیابی ویژگی‌ها و خصوصیات بلاکچین‌ها را اعطا نماید.

در این پایان نامه ابتدا به بررسی ساختار اصلی و سازنده بلاکچین‌ها می‌پردازیم و سپس با تفهیم مباحث تخصصی و پیشرفت‌هه در معماری رمز ارزهای بر بستر بلاکچین سعی در پیاده سازی هر آنچه که گفتیم توسط تعدادی ابزار ساده می‌نماییم.

کلمات کلیدی: بلاکچین، بلاک، رمز ارز، امضای دیجیتال، کلید عمومی و خصوصی، هش، گره، عدم تمرکز، توزیع شدگی، اجماع، اثبات انجام کار، استخراج.

فصل ۱. پیشگفتار

۱-۱. مقدمه

مسیر تبدیل تکنولوژی‌های دیجیتال بر بستر بلاکچین^۱ها به چیزی که امروزه نظاره گر بر آن هستیم، همواره مسیری مملو از تلاش‌های نافرجام زیادی بوده است [۱]. از ابتدای سال ۱۹۹۱ که بلاک چین تحت عنوان زنجیره‌ای از داده‌ها توسط محققین ابداع و معرفی شد، لیستی نامحدود از ایده‌ها، نیازها، و یا مشکلات و چالش‌ها به سمت این تکنولوژی سرازیر شد تا شاید بلاکچین نقش عنصری حیاتی و نجات بخش را برای آنها بازی کند [۲]. از ایده‌هایی بر محور مقالات آکادمیک که به تعداد زیادی به عنوان مرجع مورد استفاده قرار گرفته شده بودند تا سیستم‌های عملیاتی واقعی که جدا از روی کاغذ، توسعه پیدا کرده و تست شده بودند [۳].

برای درک بلاکچین‌ها لازم است تا نگاهی تاریخی حتی به قبل از پیدایش بلاکچین‌ها داشته باشیم که همواره این سوالات را در یک ذهن کاوشگر ایجاد می‌کند: بلاکچین چگونه کار می‌کند؟، تکنولوژی‌های موفق بر بستر بلاکچین کدام‌ها هستند و چگونه در این مسیر متلاطم راه خود را پیدا کردند؟، چگونه تکنولوژی‌های بلاکچینی با پیشرفت و تکامل تبدیل به ایده‌های تجاری شدند؟

۱-۲. چالش‌های پیش رو

نکته‌ای که حائز اهمیت است این است که همواره اکثر ایده‌های نو و تازه در تکنولوژی راه پرچالشی را جهت خو گرفتن با دنیای واقعی و پذیرفته شدن دارند. به طوری که تمایل به استفاده از روش‌ها و الگوهای قدیمی که امتحان خود را پس داده اند و پیاده سازی شده اند دارای اولویت برای کسب و کارها هستند. از آنجایی که

¹ Blockchain

تکنولوژی‌های بلاکچینی و بر بستر بلاکچین نیز از اینگونه سیستم‌های جدید و نوینیان هستند، نحوه سازگاری کسب و کارها با آنها همواره مسیری پر تلاطم و عمدتاً غیر قابل پیش‌بینی خواهد بود.

دشوارهای پیش رو در مسائل مرتبط با حوزه بلاکچین هنگامی سر از آب بر می‌آورند که بدانیم تمام ایده‌ها و بعضاً مشکلاتی که این سیستم‌ها حل می‌کنند، قدمتی ندارند.

تکنولوژی بلاکچین تنها یک دهه قدمت دارد. این تکنولوژی با ورود خود به عرصه دانش منجر به تغییر و تحولات سریعی در زمینه‌های مختلف شده است. با پیشرفت مفاهیم، مستندات، مقالات و تئوری‌های موجود در این عرصه، کسب و کارها تمایل بیشتری به انتقال سیستم‌ها و ساختارهای خود به بستر بلاکچین خواهند داشت.

چالش دیگری که با آن روبرو هستیم، نیاز به داشتن اکوسیستم و محیط‌هایی توزیع شده است تا بتوانیم بلاکچین و یا سیستم‌های بلاکچینی را با آنها پیاده سازی کنیم. در فصول بعدی بررسی خواهد شد که تکنولوژی‌های بر بستر بلاکچین، بدون مهیا بودن این اکوسیستم‌ها شامل فضای ذخیره سازی ابری غیرمت مرکز، بستر ارتباطات غیرمت مرکز و... قادر به پیاده سازی نمی‌باشند. بیشتر تکنولوژی‌های ذکر شده هنوز به میزان کافی توسعه نیافته‌اند و این امر منجر به افزایش خطر سرمایه گذاری در این عرصه شده است.

۱-۳. اهداف پایان نامه

نکته حائز اهمیت در اینجا این است که از بین محدود پروژه‌های بلاکچینی و یا رمز ارزی که در داخل کشور پیاده سازی شده‌اند، بیشتر آنها پروژه‌های کلاهبرداری و یا غیر واقعی بوده‌اند. لذا داشتن ایده و دانش لازم جهت ارزیابی یک سیستم بر بستر بلاکچین، هدف بسیار مهمی برای این پایان نامه می‌باشد. در این پایان نامه سعی بر توضیح صفر تا صد بلاکچین از قبیل فلسفه وجودی آن، اجزا و ساختارهای آن و قوانین و نکات پیاده سازی آن می‌شود.

جمع آوری این قوانین به صورت یکپارچه و یک دست یک چالش بزرگ بود؛ زیرا هر سیستم بر بستر بلاکچین بعضی قوانین اصلی و بعضی قوانین بعضاً مختص به خود را دارد، اما با مطالعه بر روی دوره بلاکچین و

رمز ارزها توسط جادی [۴] و همچنین کتاب بیتکوین و تکنولوژی رمز ارزها (یک مقدمه جامع) ^۱ [۵] که در بین بیش از ۱۲۰ دانشگاه برتر دنیا تدریس می‌شود، میسر شد.

این قوانین ما را در نهایت به ساخت یک بلاکچین و یک رمزارز بر بستر آن سوق می‌دهد. پیاده سازی یک بلاکچین و رمز ارز از صفر تا صد به منظور ارزیابی و تست آن در محیط آکادمیک و سپس در دنیای واقعی، از دیگر اهداف این پایان نامه می‌باشد.

۱-۴. کارهای مشابه

بلاکچین و رمز ارزهای زیادی در دنیای واقعی وجود دارند که کار هر یک از آنها تا حدودی مشابه کار این پایان نامه می‌باشد. از جمله این‌ها می‌توان به بیتکوین^۲ [۶]، اتریوم^۳ [۷] و تتر^۴ [۸] اشاره کرد.

۱-۵. ساختار پایان نامه

در فصل بعد، ابتدا مفاهیم پایه‌ای که برای درک بلاکچین لازم است را بررسی خواهیم کرد. سپس در فصل سه به بررسی ابزارهای مورد نیاز برای ساخت و پیاده سازی یک بلاکچین و یک رمز ارز بر بستر آن خواهیم پرداخت و سپس در فصل چهار به پیاده سازی اصلی بلاکچین و رمز ارز بر بستر آن خواهیم پرداخت.

در فصل پنج نیز سیستم پیاده سازی شده را ارزیابی کرده و مزایا و معایب آن را خواهیم دید و در فصل شش نیز جمع بندی و پیشنهادات آتی را مورد بررسی قرار خواهیم داد. فصل هفت نیز دارای منابع و مأخذ این پایان نامه می‌باشد.

¹ Bitcoin and Cryptocurrency Technologies (A Comprehensive Introduction)

² Bitcoin

³ Ethereum

⁴ Tether

فصل ۲. مفاهیم پایه

۱-۲. مقدمه

در این فصل برای شروع، نیاز به درکی درباره فلسفه وجودی بلاکچین با مروری تاریخی بر نحوه مبادلات و بوجود آمدن پول و اعتبار خواهیم داشت.

۲-۱. درکی درباره بلاکچین

۲-۱-۱. نگاه تاریخی

از ابتدایی که آدمیزاد شروع به مکتوب کردن وقایع، رویدادها، اعتبارات و یا حتی دارایی‌های خود کرد، همواره نیاز به وجود یک تشکل جامع بود تا بتواند صحت مکتوبات آدمی را تایید کند. به شرطی که آن تشکل جامع، مقبولیت عام را به همراه خود داشته باشد. با دو مثال این تشکل‌ها را در مسائل متفاوت بررسی می‌کنیم.

مثال ۱:

در زمان قدیم اگر تعدادی خانواده ساکن در کنار یک رود، زمین‌های کشاورزی داشتند که می‌بایست در انتهای فصول از آنها برداشت می‌کردند، خود می‌دانستند که سهم هر یک چه میزان از زمین‌های زیر کشت می‌باشد تا هنگام درو با یکدیگر به مشکل نخورند. مدتی بعد که تعداد افراد این جوامع رشد کرد، دیگر نیاز بود تا برای جلوگیری از وقوع اختلاف و هرج و مرج در روستا یا ده، فردی شناخته شده و تحت مقبولیت همه یا اکثریت افراد وجود داشته باشد تا با حکمیت وی در مسائل، دیگر جای شکی باقی نماند. برای مثال فردی به مانند ریش سفید یا کدخدای کدخدای را مطرح کند که اگر دو کشاورز برسر اراضی زیر کشت خود به مشکل برخورندند، آن فرد راه حلی را مطرح کند که گره گشای مشکل باشد. در واقع در آن زمان ریش سفید یا کدخدای

سندي بر اثبات درستي يك مسئله بود. برای مثال اگر کدخدا می گفت که نيمی از زمين برای کشاورز الف و نيمی دیگر برای کشاورز ب می باشد، دیگر کسی نمی توانست زیر حرف او بزند و گفته هایش را نقض کند زира کدخدا همان تشکل جامعی بود که مقبولیت عام را به همراه داشت.

با بزرگتر شدن جوامع دیگر يك کدخدا نه تنها کافي نبود بلکه اراضي تحت مالکيت انسانها به گونه ای زياد و بزرگ شد که وجود چند کدخدا نيز چاره کار نبود. اين شد که حکومت های اوليه در پی مدیریت اراضي، طومارهایي را شکل دادند که با مقيد کردن يك محدوده زمين به يك شخص، سعي در بوجود آوردن همان تشکلي کنند که قبل از آن با عنوان کدخدا وجود داشت. دیگر اگر شخصی مدعی می شد که بخشی از اراضي الف را داراست و برای حرف خود طومار با مهر رسمي آن حکومت را ارائه می داد، کسی برای مخالفت با حرف او بر نمی خواست زира آن طومار سندي بر حرف او بود و آن حکومت که حق قانونی استفاده از زمين را به شخص می داد مورد مقبولیت اکثریت افراد بود.

به مجردی که جوامع گسترش پیدا کردند و عصر تکنولوژی آمیخته با زندگی بشر شد، طومارهای ياد شده جای خود را به اسناد و مدارکی دادند که به موجب اعتبار خود، از دارایی های افراد در برابر دیگران حفاظت و حراست می کرد.

مثال ۲:

زمان قدیم را تصور کنید؛ هنگامی که افراد برای بدست آوردن کالاهای مورد نیاز خود از اجنبای را با یکدیگر تبادل می کردند. برای مثال فرض کنید فرد الف دارای چندین راس گوسفند بود اما نیاز به يك زمين برای نگهداري آنها داشت. فرد الف برای بدست آوردن يك زمين به سراغ فردی (برای مثال فرد ب) می رفت که دارای قطعه بزرگی زمين بود اما از قضا نیاز به چندین راس گوسفند برای مزرعه خود داشت. فرد الف و ب سپس با انجام توافقی با یکدیگر سعی بر برآورده کردن نیازهای خود می کردند. برای مثال فرد الف با دادن تعدادی راس گوسفند به فرد ب، قطعه ای زمين را صاحب می شد. حال فرض کنید که فرد ب به جای گوسفند نیاز به چندین راس اسب داشت. سپس فرد الف می بايست به سراغ فردی (برای مثال فرد ث) می رفت که در ازای گرفتن تعدادی راس گوسفند تعدادی راس اسب را به او می داد. حال فرد الف با داشتن تعداد اسب های کافی

می‌توانست به سراغ معامله زمین برود تا با دادن اسباب‌ها به فرد ب، همان قطعه زمین را صاحب شود. این پیچیدگی تبادل در زندگی بشری باعث اختراع ۱-پول و ۲-اعتبار شد. از این پس فرد الف قادر بود تا در ازای مبلغی از پول یا در ازای داشتن اعتباری ثابت شده در برابر فرد ب، قطعه‌ای از زمین او را صاحب شود. سپس با بزرگتر شدن جوامع بشری موسسه‌هایی تشکیل شدند که اعتبارات یا پول‌های افراد را ثبت می‌کردند تا هم امنیت آنها را تضمین کنند و هم در ازای اعتبار و دارایی‌های غیر پولی افراد (که ممکن بود شامل زمین، ملک و مابقی دارایی‌ها باشد) به آنها مقدار پولی را اختصاص دهند. این موسسات که همان نقش تشكیل جامع دارای مقبولیت عام را داشتند، بعدها به بانک‌ها تبدیل شدند.

حال در مثال شماره ۱، تصور که فردی با داشتن روابطی بهتر با کدخدا روزتا، زمین و دارایی فرد دیگری را صاحب می‌شد و یا تصور کنید کسی سندی را نزد اهالی روزتا می‌آورد که نشان می‌داد تمام زمین‌های آن روزتا را او صاحب شده‌است. در این صورت مشکل از چه بود؟ آیا مشکل از کدخدا و یا آن حکومت بود؟ آیا آنها به اشتباه اعتباری را به آن فرد داده بودند؟ آیا در پس این جریانات فسادی در کار بود؟

یا مثال شماره ۲ را بررسی کنید؛ فردی به یکباره صاحب ثروتی گزار می‌شد، یا یک فرد که برای بازپس گیری مقداری از دارایی‌های خود نزد بانک می‌رفت و بانک به او اعلام می‌کرد که او هیچ مقدار پول و یا اعتباری نزد بانک ندارد.

مشکل از کجا پیش می‌آمد؟ آیا خطایی توسط بانک پیش آمده بود؟ آیا فردی بواسطه دوستی خود با رئیس بانک اموال فرد دیگری را تصاحب کرده بود؟

اگر با دقت به این دو مثال بنگریم، متوجه خواهیم شد که آن تشكیل‌های جامعی که از آن یاد کردیم (برای مثال ۱ سیستم مدیریت اراضی و برای مثال ۲ سیستم بانکی)، دارای یک نقطه ضعف بسیار مهم می‌باشد و آن، مرکز بودن تشكیل‌ها است.

تشکل‌های مثال ۱ و ۲ تشكیل‌های جامعی بودند و همچنین مقبولیت عام را داشتند اما توسط یک نفر و یا تعداد محدودی از انسان‌ها مدیریت می‌شدند که شخص ثالث بودند.

این چالش نیز بعدها باعث بوجود آمدن سیستم بلاکچین شد. این فناوری از ابتدای پیدایش خود سعی بر آن داشت تا سیستم‌های متتمرکز را تبدیل به سیستم‌ها و تشکل‌های غیر متتمرکز و توزیع شده کند. همین ویژگی خاص غیر متتمرکز سازی و توزیع شدگی سیستم‌ها است که بلاک چین را بسیار محبوب کرد، زیرا غیر متتمرکز و توزیع شدگی سیستم‌ها با دیدگاه بلاکچین خود ویژگی‌های مهم دیگری را بوجود می‌آورند که به تفضیل به آنها می‌پردازیم.

۲-۲-۲. تعریف استاندارد و آکادمیک

به طور خلاصه بلاکچین سازوکاری برای مرتب سازی و تایید تراکنش‌ها در ساختمان دادهای به نام دفتر کل^۱ می‌باشد. می‌توان بلاکچین را نوعی ساختمان داده با ویژگی‌های منحصر به فرد مانند توزیع شده بودن، امنیت و ... در نظر گرفت. به طور کلی یافتن تعریف دقیق و جامع از بلاکچین کاری سخت و به سبب تازگی و متنوع بودن کاربردهای آن بعضاً می‌تواند ناممکن باشد اما با مراجعه به منابع مختلف از جمله تعاریف ارایه شده توسط بزرگان و تاثیرگذاران این عرصه می‌توان به تعریف کامل و قابل قبولی رسید. در همین راستا کمیته‌ای فنی در سال ۲۰۱۶ در سازمان ایزو^۲ تشکیل شد که تعاریفی در این زمینه ارایه دهد. این کمیته بلاکچین را اینگونه توصیف می‌کند: «یک دفتر کل توزیع شده و غیرقابل ویرایش که توانایی ثبت تراکنش‌های تولیدشده توسط طرفین معامله و تعامل کنندگان کسب و کار را دارد [۹].» در قسمتی دیگر از تعریف آمده که: «بلاکچین یک بستر دیجیتال است که توانایی ذخیره و تایید صحت تراکنش‌ها را به شیوه‌های امن و شفاف دارد؛ به طوریکه نیازی به موجودیت‌های میانی حقیقی مانند ناظرین و واسطه‌ها نباشد.» همچنین مطرح می‌شود که «بلاکچین یک دفتر کل مشترک و غیرقابل ویرایش برای ذخیره سازی تاریخچه تراکنش‌ها است [۱۰].»

¹ Ledger

² ISO

بلاکچین یک معماری توزیع شده کامپیوتری و نرم افزاری است که یک کامپیوتر در صورت شرکت در شبکه آن، یک گره^۱ نامیده می‌شود. هر گره اطلاعات کامل و دقیقی از تمامی تراکنش‌های انجام شده دارد و تمامی اطلاعات در خصوص تراکنش‌ها مابین گره‌ها مشترک است. تراکنش‌ها به طور پیوسته در قالب گروههایی با نام بلاک^۲ دسته بندی شده و در هر گره ذخیره می‌شوند. تنها یک بلاک در واحد زمان اجازه اضافه شدن به بلاکچین را دارد و حاوی الگویی ریاضیاتی و محاسبه شده می‌باشد که صحت بلاک قبلی را تایید می‌کند. به این ترتیب بلاک‌ها به طور سلسله وار به یکدیگر متصل می‌شوند و موجودیتی واحد را تشکیل می‌دهند [۱۱]. تعریف بالا را می‌توان به عنوان توصیفی کلی و جامع از بلاکچین در نظر گرفت به طوریکه اکثریت پیاده سازی‌ها و توزیع‌های بلاکچین را دربر می‌گیرد.

۲-۳. ویژگی‌ها و نقاط مثبت

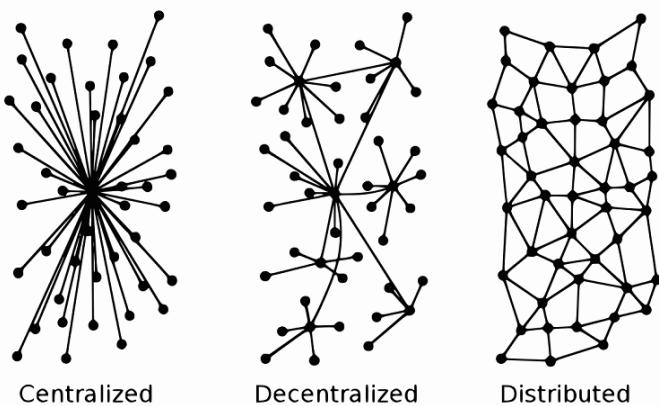
۲-۳-۱. تعاملی بودن سیستم‌ها بر بستر بلاکچین

یک تفاوت عمده سیستم‌های مبتنی بر بلاکچین توانایی ساخت محیطی کاملاً توزیع شده^۳، در مقایسه با سیستم‌هایی که تکنیک‌های توزیع سازی را در اموری با ماهیت مرکزی اعمال می‌کنند می‌باشد.

¹ Node

² Block

³ Fully Distributed



شکل ۱-۱. شماتیک سیستم‌های مرکزی، غیر مرکز و تعاملی

اجزای سیستم‌های تعاملی می‌توانند با بهره گرفتن از بستر تعاملی با یکدیگر همکاری کنند. با حذف موجودیت یا موجودیت‌های مرکزی ویژگی منحصر به فردی از بلاکچین متولد می‌شود و آن مهم این است که اجزای این سیستم می‌توانند بدون هماهنگی و کسب اجازه از موجودیتی بالادستی اقدام به برقراری قرارداد هوشمند^۱ و تعامل جدیدی کنند و این توانایی یکی از تعاریف تعاملی بودن سیستم‌های بلاکچین است. قراردادهای هوشمند یکی از بازویان اصلی تعامل اجزای سیستم‌های مبتنی بر بلاکچین می‌باشد که در ادامه تعاریف و ویژگیهای آن آورده شده‌است: قابلیت تعاملی بودن بلاکچین ویژگیهای دیگری از جمله قابلیت تحمل و مدیریت بحران بالا را ایجاد می‌کند به طوری که با قطع ارتباط یک گره سیستم دچار بحران نمی‌شود و تنها بخش کوچک و قابل اغماضی از سیستم از مدار خارج می‌شود. در ضمن اجزای یک سیستم تعاملی می‌توانند در مقابله با سوء‌رفتار و تهاجم خارجی با یکدیگر همکاری کرده و مشکل را برطرف سازند و این خود موجب افزایش امنیت این سیستم‌ها خواهد شد.

۲-۳-۲. شفافیت²

¹ Smart Contract

² Transparency

ماهیت توزیع شده بودن سیستمهای مبتنی بر بلاکچین امکان دیگری با عنوان شفافیت را به سیستم اضافه می‌کند که از حیث اهمیت در کسب کارها و صنایع جایگاه انکارناپذیری دارد. توزیع شده بودن و تعاملی بودن موجب ایجاد سابقه برای اعضا می‌شود و این خود به مرور موجب ایجاد اعتبار بین هر دو عضو دلخواه سیستم می‌شود. این ویژگی از آنجا ناشی شد که هر تغییری در سیستم برای تمامی اعضا (چه اعضای دخیل در آن تغییر و چه باقی اعضا) قابل مشاهده است و نیز ویرایش اطلاعات در بلاکچین ناممکن است. در اهمیت موضوع شفافیت این نکته قابل ذکر است که در قشر خاصی از کسب وکارها هرچه میزان شفافیت بالاتر باشد تقاضا برای همکاری و یا سرمایه گزاری افزایش می‌یابد و این موجب افزایش بهره وری و کارایی کلی سیستم می‌شود.

۳-۳-۲. امنیت

در بالا به لزوم وجود شفافیت در برخی زمینه‌ها اشاره شد ولی این نکته ناگفته ماند که مسئله امنیت و محramانگی بخش جدانشدنی از کسب وکارها می‌باشد و لازم است تمهدی جهت ایجاد توازن^۱ و مصالحه بین این دو مهم انجام شود. تکنولوژیهای مبتنی بر بلاکچین در این راستا و جهت کنترل شفافیت تعریف جدیدی به بلاکچین اضافه کرده اند : بلاکچین‌های خصوصی^۲ و بلاکچین‌های عمومی^۳.

در تعریف بلاکچین‌های عمومی ذکر شده که هر موجودیت که متمایل باشد می‌تواند اطلاعات درون شبکه بلاکچین را مشاهده کرده و آنها را به اختیار خود تغییر دهد. در حالیکه بنا بر تعریفی که برای بلاکچین‌های خصوصی ارایه شده در این نوع از سیستمهای مشاهده، تغییر و مشارکت در فرایند سیستم باید اجازه‌ای از سمت مالکان و تصمیم گیرندگان سیستم به موجودیت‌ها داده شود. به این ترتیب اقدامی جهت کنترل مسئله شفافیت و مخفی نگه داشتن اطلاعات کسب و کار از دسترسی موجودیت‌های رقیب و خارجی انجام شده است [۱۲].

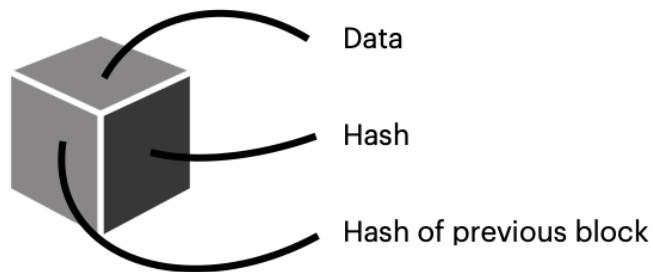
¹ Trade-off

² Private Blockchains

³ Public Blockchains

۴-۲. مفاهیم بلاک‌ها

حال که با تعریف بلاکچین آشنا شدیم می‌توانیم به صورت دقیق‌تر به اجزای تشکیل دهنده بلاکچین نگاه کنیم. به صورت ساده هر بلاک از ۳ بخش اصلی ۱-داده^۱ ۲-هش بلاک^۲ و ۳-هش بلاک قبل^۳ تشکیل شده‌است.



شکل ۱-۲. شماتیک یک بلاک به صورت ساده

۱-۴-۲. داده

داده‌ها در بلاکچین‌ها همواره با توجه به نوع و ساختار آن بلاکچین متفاوت هستند. برای مثال؛ بلاکچین‌های رمز ارزها اطلاعات تراکنش‌ها را در خود به عنوان داده ذخیره می‌کنند. این اطلاعات شامل اطلاعات فرستنده^۴ رمز ارز، گیرنده^۵ رمز ارز و مقدار سکه مبادله شده می‌باشد. این داده‌ها در نهایت توسط فرآیند هش ، هش بلاک را بوجود می‌آورند.

¹ Data

² Block Hash

³ Previous Block Hash

⁴ Sender

⁵ Recipient

۲-۴-۲. هش بلاک

فرآیند هش از همان ابتدا جوابی برای سوالات امنیتی بلاکچین‌ها بود. برای مثال فرآیند تغییر ناپذیری بلاک‌ها و دستکاری نکردن اطلاعات بلاک‌ها در بلاکچین توسط هش محقق می‌شود. هش برای بلاک‌ها به مانند اثر انگشت برای آدمهای است. این جزء همواره هویت بلاک و کل محتویات آن را تعیین می‌کند و با توجه به الگوریتمی که دارد همواره منحصر به فرد می‌باشد. و اما خود هش چیست و چگونه تولید می‌شود.

هش به طور کلی یک تابع ریاضی است که یک ورودی می‌گیرد و در ازای آن ورودی خاص یک خروجی خاص را برمی‌گرداند. یعنی در تابع f به صورت $y = f(x)$ ، همواره در ازای ورودی x به تابع هش f ، y بازگردانده می‌شود.

به عنوان مثال، فرض کنید تابع هش^۱ f به صورتی است که به ازای دریافت هر عدد، مقادیر آن عدد را با هم دیگر جمع زده و یکان حاصل جمع را بازمی‌گرداند.

$$f(142) = 1+4+2 = \underline{7} \rightarrow \text{hash}(142) \text{ by hash function } f = 7$$

$$f(237) = 2+3+7 = \underline{12} \rightarrow \text{hash}(237) \text{ by hash function } f = 2$$

در مثال فوق f یک تابع هش به صورت ساده می‌باشد، اما در دنیای کامپیوترها و بلاکچین‌ها توابع هش می‌باشند که چند ویژگی خاص را دارا باشند.

۱- توابع هش باید ورودی با هر اندازه و طولی را دریافت کنند.

۲- توابع هش باید یک خروجی با اندازه و طول ثابت داشته باشند.

۳- خروجی توابع هش باید توسط کامپیوترها در زمان نسبتاً سریع و قابل قبولی (برای مثال $O(n)$) حساب شوند.

¹ Hash Function

حال بنظر می‌رسد که تابع هش f که در بالا ساختیم تمام ویژگی‌های نام برده شده را داراست اما، نکات و نواعصی در تابع هش f جامانده که با ذکر مثال هویدا می‌شوند؛ برای مثال می‌خواهیم عدد ۶۰۱ را به تابع f بدهیم تا هش آن محاسبه شود. با توجه به ساختار تابع هش f ، جواب عدد ۷ می‌باشد. حال دوباره این سوال را از خود می‌پرسیم که مگر هش در بلاک‌ها همواره منحصر به فرد نبودند؟ پس تابع هش f به قدر کافی پاسخگوی نیازهای ما نمی‌باشد.

نکته دیگری که حائز اهمیت است که عدد ۷ به طرز غیر قابل چشم پوشی جواب بسیاری از اعدادی است که ما حتی از آنها مطلع نیستیم. این باعث می‌شود تا متوجه نقص دیگر تابع هش f بشویم.
آخرین نکته مهم نیز در تابع هش f این است که ما بر احتی می‌توانیم با داشتن عدد ۷ به عنوان هش، به سرعت اعدادی را تست کنیم تا در نهایت مثالی را به عنوان ورودی به تابع هش f در نظر بگیریم که با این جواب مطابقت دارد.

از سه نکته بالا به این نتیجه دست می‌یابیم که توابع هش به خودی خود پاسخگوی امنیت داده‌های ما در بلاک‌ها نمی‌باشند. در اینجا می‌توانیم به سراغ هش‌های کریپتوگرافیک برویم که دقیقاً سه نقص بالا را برطرف کرده و ویژگی زیر را دارند.

- ۱- هش‌های کریپتوگرافیک^۱ در برابر برخورد مقاوم^۲ هستند. این بدان معناست که ورودی‌های متفاوت دارای خروجی‌های یکسان نیستند و یا این مقدار از برخورد بقدری کم و نادر است که قابل چشم پوشی است.
- ۲- هش‌های کریپتوگرافیک یک طرفه^۳ می‌باشند. بدان معنا که هیچ وقت با دانستن هش یک ورودی، نمی‌توان مقدار خود ورودی را حدس زد یا آن را بدست آورد.

¹ Cryptographic Hash

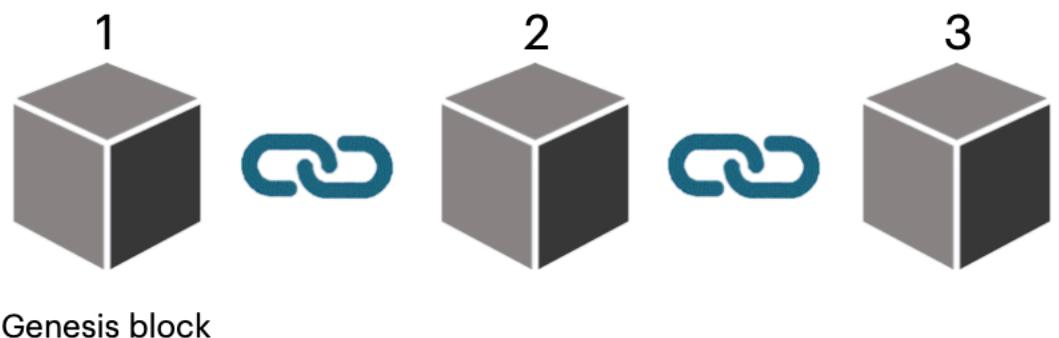
² Collision Resistance

³ One-Way

۳- هش‌های کریپتوگرافیک هش‌هایی هستند که امکان تست کردن مقادیر ورودی با سرعت بالا^۱ بر روی آنها وجود ندارد و این قابلیت راحتی پازل^۲ هش‌های کریپتوگرافیک می‌باشد تا عملاً کامپیوترها با تست کردن سریع تعداد بسیار زیادی از ورودی‌ها به خروجی هش مورد نظر برسند.

۳-۴-۲. هش بلاک قبل

عنصر سومی که در بلاک ذخیره می‌شود هش بلاک قبل است. که با مثال زیر آن را شرح می‌دهیم.



شکل ۲-۲. زنجیره‌ای از بلاک‌ها و ارتباط آنها توسط هش بلاک قبل

در شکل ۳ یک زنجیره از سه بلاک وجود دارد. همانطور که در تصویر مشاهده می‌کنیم هر بلاک شامل یک هش و هش بلاک قبلی می‌باشد. از این رو بلاک شماره ۳ به بلاک شماره ۲، و بلاک شماره ۲ به بلاک شماره ۱ مربوط است.

¹ Brute Force

² Puzzle Friendliness

حال بلاک شماره ۱ کمی با دیگر بلاک‌ها متفاوت است چون بلاکی قبل از آن وجود ندارد و به اصطلاح به آن بلاک اولیه^۱ گفته می‌شود.

حال فرض کنیم که بلاک دوم دستکاری بشود. با ایجاد تغییرات در بلاک شماره ۲، هش آن بلاک نیز متقابلاً تغییر می‌کند و هیچکدام از بلاک‌های بعدی معتبر نیستند، زیرا دیگر هشی که از بلاک قبل خود به دست دارند معتبر نیست. از این رو اگر بلاکی کوچکترین تغییری بکند، تمام بلاک‌های بعد از آن نیز نامعتبر می‌شوند. این سیستم است که باعث شکل گیری یک زنجیره در بلاک‌ها می‌شود.

۲-۵. امضاهای امضاها و دیجیتال^۲

استفاده از هش تنها ایده نبوغ آمیز در ایجاد و بهبود امنیت و اعتبار بخشیدن به بلاک‌ها نبود. استفاده از سیستم امضای دیجیتال ایده دیگری که به سراغ بلاکچین آمد تا در کنار فرآیند هش راهی برای تشخیص صحت و درستی داده‌های بلاک‌ها باشد.

فرض کنید که در بخش نظارت بر کیفیت یک شرکت تولید حافظه رم هستید. شما حافظه رم‌های تولید شده را می‌گیرید و در صورت کارکرد صحیح هر کدام از آنها، بارکدی را در جعبه آن می‌زنید که آن بارکد نمایانگر تایید کیفیت آن حافظه رم توسط شرکت شما می‌باشد. درج بارکد به صورتی است که تنها شما قادر به تولید آن هستید اما هر کسی می‌تواند با اسکن بارکد از صحت اصل بودن و کارکرد درست آن مطلع شود. در حقیقت این سیستمی است که باعث می‌شود کالاهای شما از دیگر کالاهای نامعتبر مجزا شود.

¹ Genesis

² Digital Signature

این سیستم در دنیای دیجیتال به گونه‌ای توسعه یافت که با یک جفت کلید خصوصی^۱ و عمومی^۲ می‌توان مدارک، اسناد یا داده‌ها را امضا کرد. کلید خصوصی و کلید عمومی به این صورت زیر عمل می‌کنند.

۱- کلید خصوصی همانطور که از اسمش پیداست تنها برای شماست و کسی نباید اطلاعات آن را داشته باشد.

۲- کلید عمومی نیز بر خلاف کلید خصوصی برای عموم قابل دسترسی است.

فرض کنید که شما یک پیام دارید که می‌خواهید آن را به فرد دیگری انتقال دهید به طوری که فرد از ارسال آن پیام توسط شما اطمینان یابد، برای این کار فرآیند زیر انجام می‌شود.

۱- شما پیام خود را توسط کلید خصوصی تبدیل به یک امضا می‌کنید.

$\text{Sig} = (\text{Secret Key}, \text{Msg})$

۲- سپس امضا خود را به همراه کلید عمومی و پیام در دسترس آن فرد قرار می‌دهید.

حال چنانچه آن فرد امضا، کلید عمومی و پیام شما را در کنار هم قرار داد و مقدار بازگشتی معتبر شد، می‌توان ثابت کرد که آن پیام از طرف شخص خود شما به او مخابره شده است.

IsValid if and only if -> (Public key, msg, sig)

در حقیقت می‌توان گفت که آن فرد اطمینان می‌یابد که آن پیام از طرف کسی آمده است که کلید خصوصی را به همراه خود دارد.

کلیدهای خصوصی و عمومی به ترتیب به نوعی تداعی کننده قفل کردن و باز کردن یک داده هستند. شما با کلید خصوصی داده‌ای را قفل می‌کنید و چون دیگران با کلید عمومی شما می‌توانند آن را باز کنند مطمئن می‌شوند که شما آن را قفل کرده بودید.

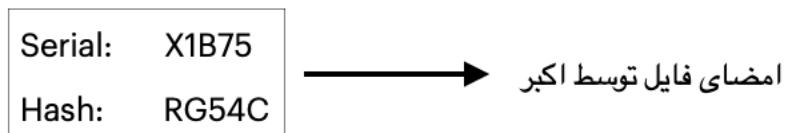
¹ Private Key

² Public Key

علاوه بر تایید اطلاعات بلاک‌ها از جفت کلید خصوصی و عمومی در دیگر اجزای وابسته به بلاکچین مثل ساخت کیف پول‌ها برای رمز ارزهای مبتنی بر بلاکچین استفاده می‌شود که در آینده به آن می‌پردازیم.

۲-۶. عدم تمرکز^۱

برای اینکه به درک چالش‌های اجماع و بحث بر روی عدم تمرکز در بلاکچین پی ببریم، می‌خواهیم تا با مثالی آن را شرح دهیم. اکبر را تصور کنید که تصمیم می‌گیرد سکه‌ای با نام خودش در دنیای دیجیتال بسازد. از قضا او تنها کسی است که می‌تواند این کار را انجام دهد. سکه او در اصل یک فایل دیجیتالی می‌باشد که شماره سریالی خاص^۲ را برای آن فایل، به عنوان سریال سکه خودش در نظر می‌گیرد. سپس آن فایلی که دارای شماره سریال می‌باشد را هش کرده و با استفاده از کلید خصوصی خودش آن فایل را امضا می‌کند.



سکه‌ی اول

شکل ۲-۳. شماتیک ساده یک سکه به همراه سریال و هش

این به دیگران اطمینان می‌دهد که اولاً شماره سریال آن سکه معتبر است (زیرا فایل سریال آن سکه هش شده) و دوماً اکبر شخصاً این سکه را ساخته (زیرا با کلید خصوصی خودش آن را امضا کرده) است.

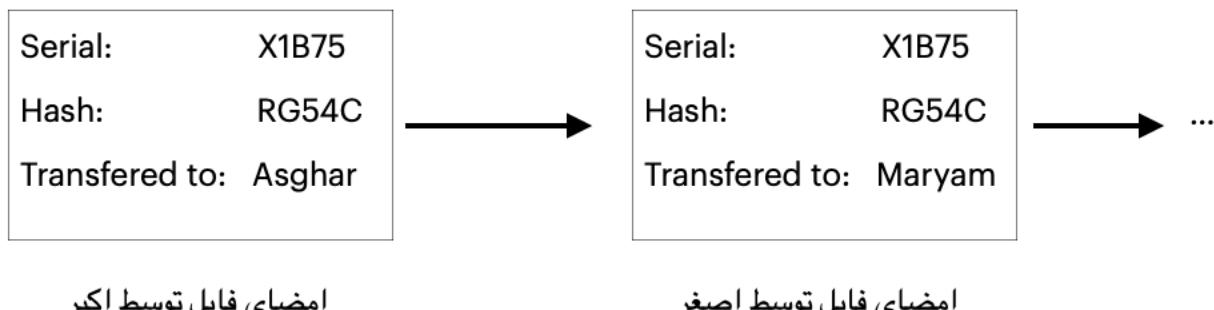
¹ Decentralization

² Unique

اکنون فرض کنید که اکبر سکه خود را به اصغر می‌دهد. اصغر همواره می‌تواند ثابت کند که آن سکه اصل بوده و توسط اکبر ساخته و امضا شده‌است و چنانچه کسی شک به اصالت سکه دارد می‌تواند با کلید عمومی اکبر از صحت اطلاعات محتوای سکه (که در اصل همان فایل دیجیتالی است) اطمینان حاصل نماید.

بعد از مدتی اصغر تصمیم می‌گیرد تا سکه‌ای که از اکبر گرفته بوده‌است را به مریم بدهد. اما مریم این سکه را قبول نخواهد کرد. او مطمئناً از اصغر می‌پرسد: از کجا معلوم است که تو این سکه را به من دادی؟ این سکه را که اکبر صادر کرده‌است! این سکه اصلاً متعلق به چه کسی است؟

در اینجا به این نکته دست می‌بابیم که همواره هش کردن و امضای دیجیتال، پاسخگوی مشکلات ما در سیستم ایمن بر بستر بلاکچین نخواهد بود. ابزاری که می‌توانیم توسط آن مشکل را حل کنیم، ابزاریست که رد و بدل کردن سکه‌ها را ذخیره کند. برای اینکار اکبر در ابتدا اگر می‌خواهد که سکه خود را به اصغر بدهد، علاوه بر شماره سریال، بر روی آن می‌نویسد: «به اصغر انتقال داده شد» و سپس آن را امضا می‌کند. حال اگر اصغر این سکه را داشته باشد ادعا می‌کند که سکه اصل بوده و همچنین توسط گفته اکبر به او منتقل شده است. اکنون اصغر می‌تواند سکه خود را به مریم بدهد و مریم سکه را به شرطی قبول خواهد کرد که اصغر بر روی آن بنویسد: «به مریم انتقال داده شد» و آنرا با کلید خصوصی خود امضا کند.



شکا ۴-۲. حگونگ حایه حایه مالکت سکه‌ها تو سط اف اد

اما مشکل بزرگی این وسط امنیت سکه‌های تولید شده توسط اکبر را تهدید می‌کند. یکی از چالشی‌ترین و خطرناک‌ترین مسائل در دنیای ارزهای دیجیتال، دوبار خرج کردن^۱ می‌باشد. این مسئله به شکل ساده بیانگر آن

1 Double Spending

است که واحد از یک ارز نباید بیش از یکبار توسط کسی خرج شود. ممکن است که اصغر به شکل همزمان پول را به مریم و شخص دیگری انتقال بدهد.

می‌خواهیم برای حل این مشکل مثالی را طرح کنیم؛ تصور کنید جمشید نیز فردی است که عیناً به مانند اکبر سکه‌های معتبری را تولید می‌کند. اما راه حل او برای مسئله این است که علاوه بر کاری که اکبر در انتقال سکه انجام می‌داد، او لیستی از نقل و انتقالات را بوجود می‌آورد که بر روی آن رد و بدل شدن سکه‌ها از فردی به فرد دیگر ثبت می‌شود. در اینجا چنانچه فردی بخواهد به شکلی سعی در دوبار خرج کردن سکه متعلق به خود کند، دیگر افراد می‌توانند به آن لیست مراجعه کرده و از یک و فقط یکبار خرج شدن آن سکه اطمینان حاصل کنند.

Serial:	X1B75
Hash:	RG54C
Transferred to: Sadegh	

امضای فایل توسط جمیشد

Serial:	X1B75
Hash:	RG54C
Transferred to: Sogol	

امضای فایل توسط صادق

Serial:	X1B75
Hash:	RG54C
Transferred to: Keyvan	

امضای فایل توسط سوگل

دفتر کل نقل و انتقالات جمیشد

در مرحله اول

در مرحله دوم

در مرحله سوم

Serial:	X1B75
Is now for:	Sadegh

Serial:	X1B75
Is now for:	Sadegh
Is now for:	Sogol

Serial:	X1B75
Is now for:	Sadegh
Is now for:	Sogol
Is now for:	Keyvan

شکل ۵-۲. جایه جایی سکه‌ها و سیاهه دارندگان آنها توسط دفتر کل

بایایید تا نگاهی کلی بر روی سیستم ساخت سکه جمشید داشته باشیم. این سیستم هرچند که مشکل دوبار خرج کردن سکه‌ها را حل کرده، ولی عملاً بمانند یک بانک مرکزی عمل می‌کند. بنظر می‌آید که راه حل برطرف کردن مشکل دوبار خرج کردن در متمرکز کردن سیستم است؛ اما همانطور که در ابتدای بحث نیز گفته شد، دیدگاه بلاکچین بر خلاف این رویکرد و عدم تمرکز در سیستم است تا همواره یک بانک، یک مرکز، یک انبار ذخیره داده و ... وجود نداشته باشد.

مسئله‌ای که در اینجا مطرح می‌شود این است که در صورت پخش کردن این سیستم در دست چند نفر، مشکل به اجماع^۱ در داده‌ها بوجود می‌آید. زیرا اگر تعدادی افراد دیگر نیز بخواهند سکه‌ای را که جمشید می‌ساخته را بسازند، آن موقع در نقل و انتقالات آن دچار مشکل می‌شوند؛ چون هر کدام از آنها یک دفتر کل مخصوص به خود را نگه خواهند داشت و دیگر نمی‌توانند بر سر معتبر بودن یک سکه یا نقل و انتقالات آن به اجماع برسند.

۷-۲. بحث بر اجماع

هنگامی که قرار است سیستمی با ساختار غیر متمرکز پیکر بندی شود، همواره پنج اصل مهم درباره آن مطرح می‌باشد که عبارتند از:

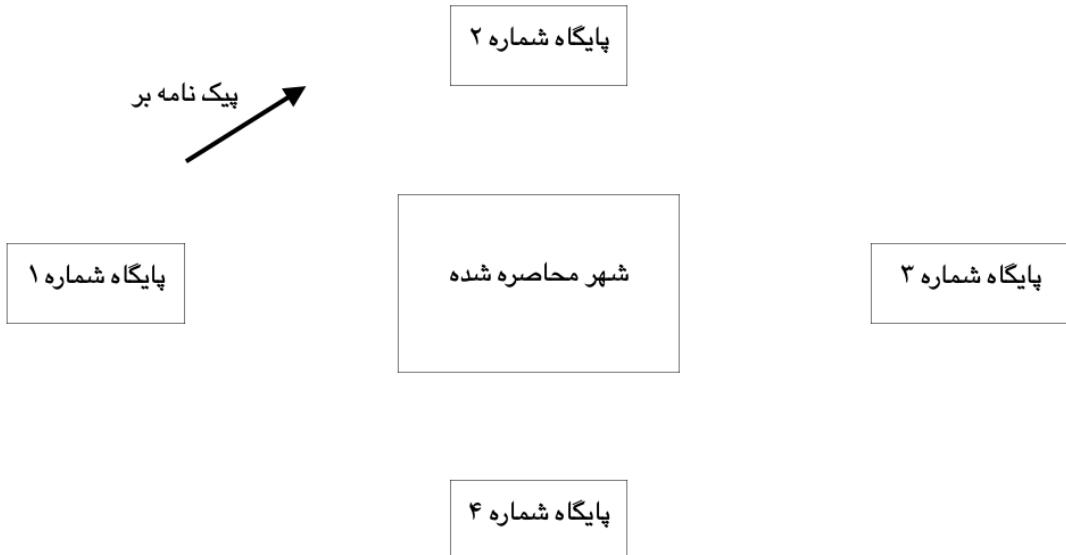
- ۱- چه افرادی مجاز هستند که دفتر کل داده‌ها را نگه داری کنند؟
- ۲- چه افرادی می‌توانند صحت اطلاعات (برای مثال تراکنش‌ها یا حتی اطلاعات دفتر کل داده‌ها) را چک کنند؟
- ۳- چه افرادی می‌توانند داده جدید (برای مثال در اینجا سکه جدید) را تولید کنند؟
- ۴- چه افرادی قوانین و خط مشی سیستم را تعیین خواهند کرد؟
- ۵- چگونه داده‌های تولیدی (بمانند سکه‌ها) ارزش گذاری خواهند شد؟

¹ Consensus

توجه کنید که همواره ۳ اصل اول از اصول بالا، از مباحث تکنیکال سیستم‌های غیر متمرکز هستند. در حالی که اصول ۴ و ۵ از بیشتر رویکرد مدیریتی و اقتصادی دارند. این سوالات همواره با روشی درست از اجماع و توافق نظر بین افراد شرکت کننده در سیستم پاسخ داده خواهد شد. (گفتنی است که در دنیای رمز ارزها ابتدا بیتکوین بود که جواب تمامی سوالات بالا با روشی نوین طرح کرد و باعث شگفتی و توجه همگان شد). رویکرد درست اجماع در سیستم‌های غیر متمرکز ممکن است بسیار دشوار و حتی غیر ممکن باشد. برای درک بهتر اجماع در این سیستم‌ها می‌خواهیم یکی از مسائل کلاسیک مرتبط با این مسئله را مطرح کنیم.

۱-۷-۲. مسئله ژنرال‌های بیزانسی^۱

در این مسئله، گروهی از ژنرال‌های بیزانسی شهری را محاصره کرده اند. ژنرال‌ها به علت کمبود نیرو، امکان حمله مستقل به شهر را ندارند. در نتیجه ژنرال‌ها به توافقی برای حمله یا عقب نشینی احتیاج دارند. آنها برای انتقال پیام و نظر خود تنها می‌توانند از هر پایگاه یک پیک به پایگاه‌های دیگر بفرستند.



شکل ۶-۲. شماتیک مسئله ژنرال‌های بیزانسی

¹ Byzantine Generals

در مسئله ژنرال‌های بیزانسی، به جز چالش زندانی شدن پیک‌ها، چند چالش دیگر هم وجود دارد. چالش نخست با تاخیر رسیدن، گم شدن یا نابودی پیام است. چالش دوم این است که به علت خطرات، ژنرال‌ها امکان تغییر رای ندارند. چالش سوم این است که شاید ژنرال یا ژنرال‌هایی خیانتکار باشند. جالب اینجا است که در صورت افزایش تعداد ژنرال‌های خیانتکار، آن‌ها می‌توانند حمله را مختل کنند.

آن چنان که احتمالاً تاکنون حدس زده اید این حالت دقیقاً مدل ارتباط گره‌ها در شبکه بلاکچین است. (توجه کنید که در مسئله ما هر فردی که در سیستم غیر مت مرکز مشارکت دارد یک گره تلقی می‌شود) هر گره نقش ژنرالی است که باید نظرش را درباره هر حمله یا تراکنش اعلام کند.

راه حل‌ها و حتی فرمول‌های بسیار زیادی برای اجماع در چنین مسائلی بیان شد بطوری که حتی گفته می‌شود در چنین سیستم‌هایی چنانچه یک سوم افراد مشارکت کننده بر خلاف انتظار عمل کنند، اجماع با شکست مواجه خواهد شد. اما دو راه ساده برای اجماع در دنیای دیجیتال مطرح شد.

۱- هر کسی در قبال مشارکت خود پاداش می‌گیرد.

برای مثال در سیستم غیر مت مرکزی که از آن یاد کردیم، هر فرد در صورت امضای یک بلاک مقداری سکه را به عنوان پاداش دریافت می‌کند. این باعث می‌شود تا افراد در سیستم بر خلاف عرف عمل نکنند بلکه صاحب پاداش شوند.

۲- ساخت و افزودن بلاک‌ها به سیستم باید به نحوی به صورت شناسی و رندوم^۱ صورت بگیرد.

در اینجا نکته حائز اهمیت این است که تصمیم درمورد ساخت بلاک به صورت شناسی در افراد ممکن است باعث وقوع حمله سیبیل^۲ شود. این حمله به صورتیست که امکان دارد شخصی چندین و چند کپی و گره از خود در سیستم ایجاد نماید تا امکان پیروزی وی در این سیستم رندوم بیشتر شود.

¹ Random

² Sybil Attack

۲-۷-۲. اجماع ضمنی^۱

برای اجماع به صورت ضمنی همواره پنج گام اساسی نیاز است.

- ۱- تراکنش‌های جدید به تمام گره‌ها پخش^۲ می‌شوند.
- ۲- هر گره تراکنش‌های جدید را در یک بلاک ذخیره می‌کند. (به این روند ذخیره سازی تراکنش‌ها در بلاک ممپول^۳ گفته می‌شود)
- ۳- در هر راند^۴ (زمان معین شده) یک گره به صورت رندوم بلاک ساخته شده خود را برای تمامی گره‌ها پخش می‌کند.
- ۴- گره‌های دیگر آن بلاک را به عنوان بلاک جدید می‌پذیرند، تنها به شرطی که تمامی تراکنش‌های داخل بلاک معتبر باشند. (سکه‌ها خرج نشده باشند، دارای اعتبار باشند و امضا شده باشند)
- ۵- گره‌ها پذیرش بلاک را با گنجاندن هش آن در بلاک بعدی که ایجاد می‌کنند، بیان می‌کنند.

با این شیوه می‌توانیم ادعا کنیم که مشکل‌ها و چالش‌هایی که پیش از این با آنها رو به رو بودیم را حل کرده‌ایم.

۲-۸. اثبات انجام کار^۵

¹ Implicit Consensus

² Broadcast

³ Mempool

⁴ Round

⁵ Proof of Work

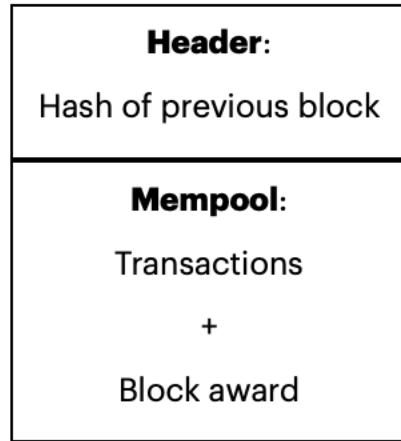
به یاد داشتیم که با استفاده از الگوریتم اجماع ضمنی توانستیم از پس حملات مختلفی که با آنها رویرو بودیم برباییم، اما کماکان یک حمله مهم یعنی حمله سبیل به راحتی ممکن است در این سیستم رخ بدهد. برای جلوگیری از این حمله ایده نبوغ آمیزی در میان آمد که اثبات انجام کار بود.

اثبات انجام کار بدین معناست که هر گره باید برای آنکه بتواند بلاکی را به بلاکچین اضافه کند، عملی را از قبیل به اشتراک گذاری منابع سخت افزاری و یا حل معادلات مربوط به هش انجام دهد تا ثابت کند در ساخت بلاک جدید بی تاثیر نبوده است. اگر این قبیل کارها را در سیستم انجام دهید، آن وقت می توانید در سیستم ساخت و افزودن بلاک به صورت رندوم حضور داشته باشید. از دیگر مزیت های اثبات انجام کار این است که هر فرد به میزان کاری که انجام می دهد و به میزان منبع و انرژی که صرف می کند شانس خواهد داشت. پس همواره یک گره باید برای اینکه بتواند در سیستم شناسی برای ساخت بلاک و افزودن آن به شبکه داشته باشد و پاداش خود را دریافت کند، عمل به نسبت دشواری را انجام دهد.

فرآیند اثبات انجام کاری که باید مورد استفاده در سیستم قرار بگیرد می بایست از نظر کریپتوگرافی غیر قابل نفوذ باشد. پس استفاده از هش های کریپتوگرافیک بهترین گزینه برای این مسئله می باشند. بیایید دوباره یک بلاک را با دقت بررسی کنیم.

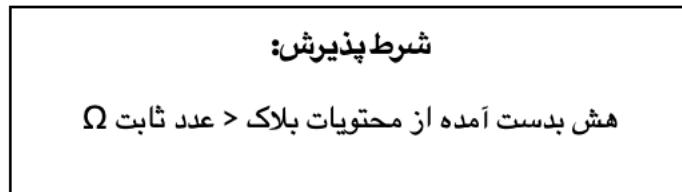
با توجه به قوانین اجماع ضمنی که بالاتر گفته شد، یک بلاک (برای نمونه بلاک مرتبط با رمز ارز) شامل تراکنش های افراد و مبادلات سکه از فردی به فرد دیگر است که ممپول نامیده شد. همچنین علاوه بر تراکنش های افراد در بلاک تراکنشی وجود دارد که در آن چند سکه (که قبل از آن با عنوان پاداش ساخت بلاک یاد کردیم) به صاحب و سازنده بلاک منتقل خواهد شد.

بلاک زیر تا به اینجا نمایانگر خصوصیات گفته شده می باشد.



شکل ۲. ساختار یک بلاک به صورت بخش به بخش

سیستم اثبات انجام کار اینچنین است که از گره‌ها توقع دارد تا تمامی اطلاعات یک بلاک را که در بالا گفته شد را هش کرده به نحوی که مقدار آن هش از یک عدد معین (که آن را امگا - Ω می‌نامیم) همواره کمتر باشد. در این صورت آن بلاک قابل قبول در سیستم است و می‌تواند به عنوان بلاکی جدید به بلاکچین اضافه شود.

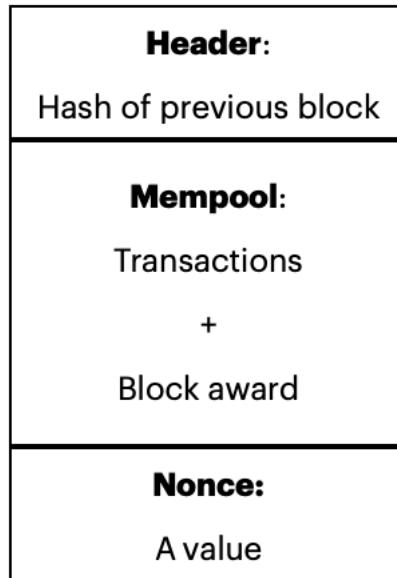


شکل ۲. فرمول شرط پذیرش یک بلاک در بلاکچین

اما می‌دانیم که اطلاعات داخل بلاک همواره ثابت می‌باشد و برای آن تنها یک هش محاسبه خواهد شد که منحصر به فرد می‌باشد و ممکن است که میزان هشی که از بلاک بدست می‌آید حتی به عدد تعیین شده نزدیک هم نباشد.

برای حل این مشکل می‌بایست بلاک‌ها علاوه بر ویژگی‌های هدر^۱ و همچنین ممپول، یک متغیر با عنوان نانس^۲ در خود داشته باشند.

متغیر نانس بدین گونه عمل می‌کند که به ازای هر مقدار از آن هش بلاک محاسبه می‌شود تا بررسی شود که آیا هش بلاک با مقدار نانس مشخص شده، از میزان عدد تعیین شده کمتر شده است یا خیر.



شکل ۹-۲. ساختار کلی یک بلاک با تمامی جزئیات

برای مثال در ابتدا متغیر نانس مقدار ۱ را در خود جای می‌دهد. سپس هش بلاک که اینبار علاوه بر هدر و ممپول، شامل نانس هم می‌باشد محاسبه خواهد شد، اگر شرط پذیرش بلاک برقرار نشد مقدار نانس برابر ۲ می‌شود تا بار دیگر هش محاسبه شود. این روند آنقدر ادامه پیدا می‌کند تا بالاخره مقدار نانسی پیدا شود که بتواند هش تولیدی بلاک را در محدوده شرط پذیرش بلاک قرار دهد.

این نکته حائز اهمیت است که سیستم اثبات انجام کار بر مبنای راحتی پازل است، یعنی با این که پیدا کردن یک بلاک با شرایط گفته شده کار سخت و زمانبری می‌باشد، اما به محض پیدا شدن بلاک^۳ دارای شرط پذیرش،

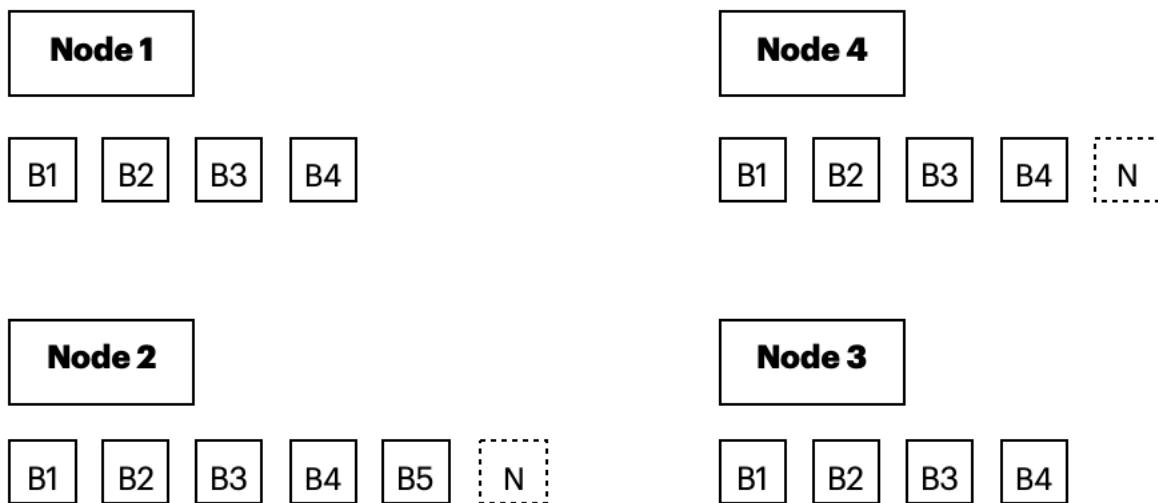
¹ Header

² Nonce

همه گره‌های دیگر می‌توانند به یکباره هش آن بلاک را محاسبه کنند و بررسی کنند که آیا بلاک معتبر است و در شرط پذیرش قرار می‌گیرد یا خیر، تا در نهایت همگی آن بلاک را در بلاکچین خود قرار دهند. این باعث می‌شود تا کسی بدون سختی فرآیند اثبات انجام کار بلاک نامعتبری نسازد.

سوالی که در اینجا مطرح می‌شود این است که ممکن است به طور شناسی در یک زمان مشخص، یک یا چند گره دیگر نیز بلاکی را کشف کنند که دارای شرط پذیرش است. در این صورت کدام بلاک به عنوان بلاک معتبر بر روی بلاکچین تمامی گره‌ها قرار خواهد گرفت؟

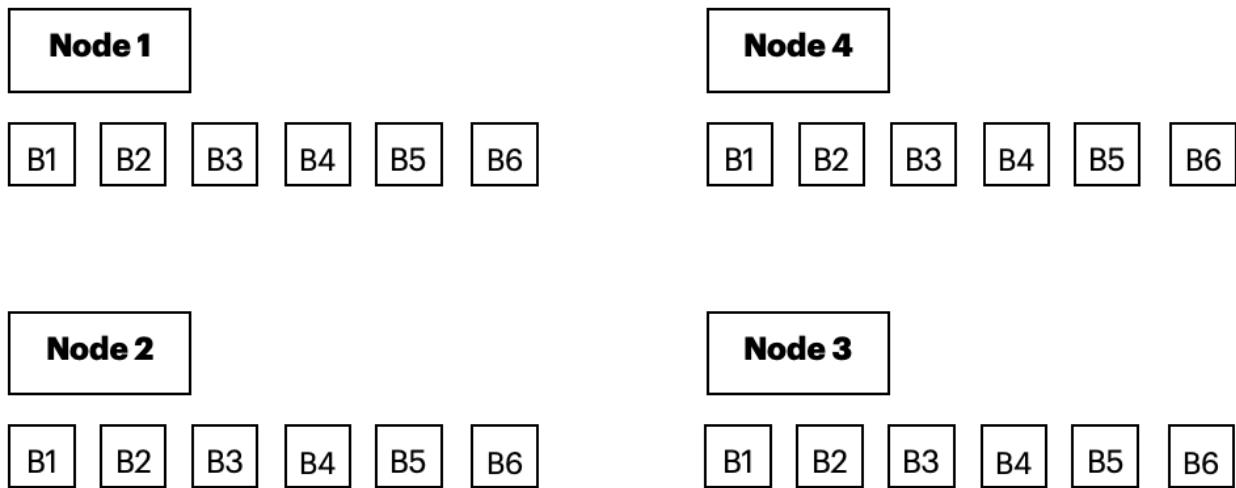
در جواب این سوال یک قانون اساسی در پذیرش بلاک‌ها مطرح می‌شود بدین شکل که: همواره طولانی‌ترین زنجیر از بلاک‌ها، معتبرترین بلاکچین می‌باشد. می‌خواهیم تا با یک مثال آن را بررسی کنیم. در سناریوی پایین ۴ گره وجود دارند که مشغول فرآیند اثبات انجام کار هستند.



شکل ۲-۱۰. گره‌های یک شبکه در سناریوهای اثبات انجام کار

در شکل مشاهده می‌کنیم که از قضا گره‌های ۲ و ۴ به صورت همزمان هر کدام به ترتیب بلاک جدیدی را که شامل اصل پذیرش می‌باشد را بدست آورده‌اند. حال بر طبق اصل گفته شده در بالا بلاکی که گره ۲ آن را بدست آورده معتبر است، زیرا بلاکچینی که گره ۲ توسط خود دارد طولانی‌تر است. این بدان معناست که گره

۲ فرآیند اثبات انجام کار بیشتری را انجام داده است و یک بلاک بیشتر از گره ۴ دارد، پس بلاکچین او معتبر خواهد شد و بقیه گره‌ها موظفند از آن پس همگی یک نمونه از بلاکچین گره ۲ را برای خود نگه دارند و بر روی کشف بلاک‌های بعدی آن تلاش کنند.



شکل ۱۱-۲. گره‌های یک شبکه بعد از همگام سازی

فرآیند چک کردن بلاکچین گره‌ها با یکدیگر در مدت زمان‌های معینی می‌تواند انجام بپذیرد. از دیگر نکات مهم در فرآیند اثبات انجام کار این است که عدد معین امگا (Ω)، در اصل سختی شبکه را تعیین می‌کند، بدین معنا که با تغییر این عدد پیدا کردن بلاک جدید با شرط پذیرش ممکن است سریع‌تر یا کندر به وقوع بپیوندد.

برای مثال فرض کنید امگا در دو حالت برابر اعداد زیر باشد.

$$\Omega_1 = 0 \dots 43753546, \Omega_2 = 0 \dots 00000003546$$

واضح است که پیدا کردن بلاک با شرط پذیرش Ω_2 به مراتب سخت‌تر از پیدا کردن بلاک با حالت امگا Ω_1 می‌باشد. زیرا بدست آوردن هش بلاکی که مقدارش از Ω_2 کمتر باشد به مراتب سخت‌تر از است و مقدار Ω_2

بسیار کمتر از Ω_1 است. پس چنانچه Ω_2 برای شبکه در نظر گرفته شود سختی شبکه بیشتر از زمانی است که Ω_1 برای شبکه تعیین شود.

۱-۹. استخراج^۱

برای درک درست ماین شدن بلاک‌های جدید بهتر است تا دوره‌ای بر روی روند ساخت بلاک‌ها به صورت دقیق‌تر از قبل داشته باشیم.

بلاک‌ها به صورت پیش فرض می‌بایست:

- ۱- تراکنش‌ها (انتقالات سکه‌ها از یک حساب به یک حساب دیگر) توسط افراد حاضر را در خود ذخیره کنند، مادامی که نسبت به امضای درست و دوبار خرج نشدن سکه‌ها صحت حاصل می‌کنند.
- ۲- یک تراکنش شامل تعداد معینی سکه به خود، به عنوان پاداش در صورت پذیرفته شدن بلاک ذخیره کنند.
- ۳- کل تراکنش‌های افراد و تراکنش در صورت پذیرفته شدن بلاک را در بخش ممپول ذخیره کنند.
- ۴- بلاک‌چین معتبر تا آن لحظه را نگه داری کنند.
- ۵- یک بلاک جدید را تحت عنوان بلاک کاندیدا^۲ سرهم کنند. این بلاک باید شامل هش آخرین بلاک پذیرفته شده ماقبل خود، ممپول یادشده، و نانس باشد.
- ۶- به دنبال مقدار نانسی باشند که هش محتويات کلی بلاک توسط آن، در شرط پذیرش بلاک مورد قبول واقع می‌شود.

¹ Mining

² Candidate Block

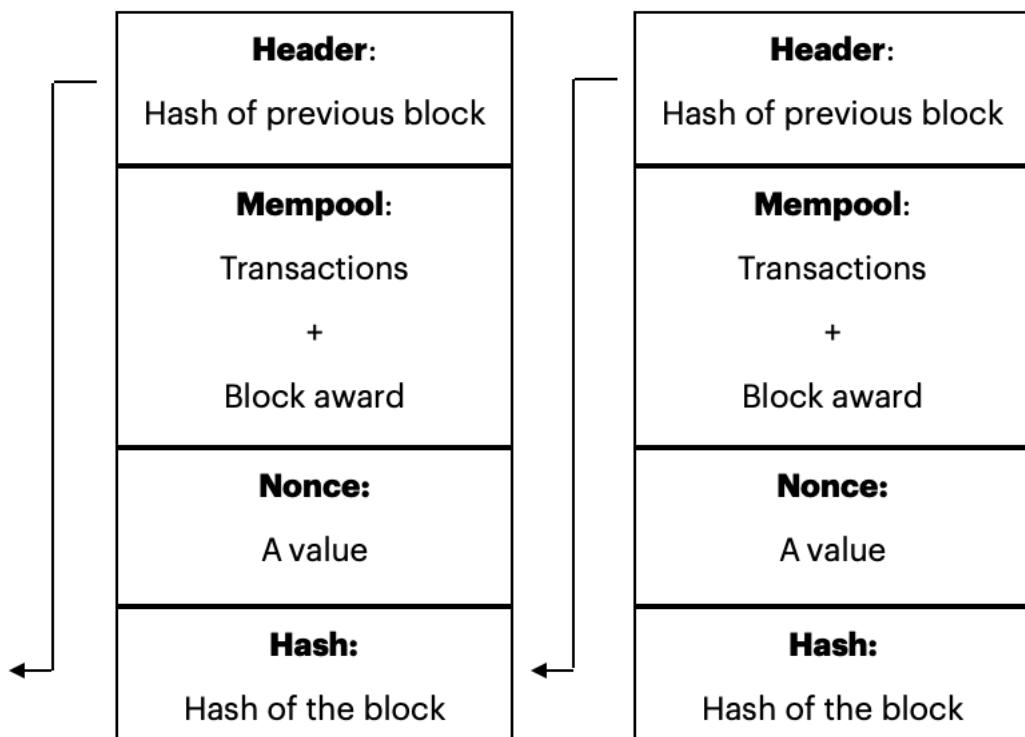
به مجموع این فرآیندها ماین کردن یا استخراج می‌گوییم. منظور از ماینینگ به طور واضح تر، پیدا کردن نанс برای بلاک سرهم شده می‌باشد تا آن بلاک را توسط شرط پذیرش مورد تایید قرار دهد. در آن صورت گفته می‌شود که یک بلاک جدید ماین (استخراج) شده است.

Mining:

Hash (Hash of previous block, tx, tx, tx,..., Block award, Nonce)

شکل ۱۲-۲. فرآیند خلاصه شده به صورت ریاضی از فرآیند استخراج

در شکل زیر آخرین بلاک استخراج شده و مورد پذیرش در بلاک چین را مشاهده می‌کنید.



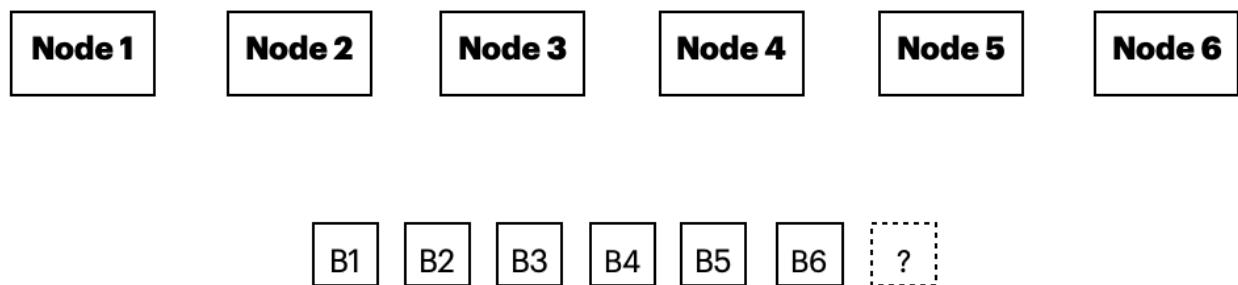
شکل ۱۳-۲. زنجیره بلاک‌های استخراج شده

این روند به طور کلی به ازای هر بار ماین کردن بلاک‌های جدید طی می‌شود.

همانطور که بخاطر داریم سختی شبکه^۱ که پیش از این با عنوان مقدار امگا (Ω) از آن یاد کردیم، تاثیر مستقیم بر روی سرعت ماین شدن بلاک‌های جدید دارد. اما نکته حائز اهمیت این است که مدت زمان ماین شدن بلاک‌های جدید بر روی بلاکچین باید روندی به نسبت ثابت و منطقی داشته باشد. به طوری که بلاک‌ها تقریباً در مدت زمان ثابتی بتوانند ماین شوند. از این رو تعیین مقدار درست سختی شبکه، امری بسیار حیاتی در بلاکچین‌ها می‌باشد.

حالت زیر را در نظر بگیرید.

بلاکچینی را فرض کنید که در شبکه آن ۶ گره به صورت مشترک مشغول به ماین کردن بلاک‌های جدید بر روی بلاکچین هستند.



شکل ۱۴-۲. شماتیک سناریوی کاهش و افزایش سرعت استخراج شبکه توسط گره‌ها

سناریوهای زیر را تصور کنید:

- در حالی که ۶ گره در شبکه مشغول ماین بلاک‌های جدید هستند، یک یا چند گره به سیستم اضافه شوند، کپی کامل بلاکچین را دریافت کرده و سپس آنها نیز مشغول ماین کردن بلاک‌ها بشوند.
- در حالی که ۶ گره در شبکه مشغول ماین بلاک‌های جدید هستند، گره ۲ و ۴ قدرتشان در محاسبه تولید هش بلاک‌ها (بواسطه افزودن سخت افزار به گره، بهبود سخت افزارهای فعلی و ...) افزایش یابد.

¹ Difficulty Level

۳- در حالی که ۶ گره در شبکه مشغول ماین بلاک‌های جدید هستند، یک یا چند گره از سیستم خارج شوند. (بواسطه نقص فنی، تمام روند کار بر روی بلاکچین و یا...).

۴- در حالی که ۶ گره در شبکه مشغول ماین بلاک‌های جدید هستند، گره ۱ و ۵ قدرتشان در محاسبه تولید هش بلاک‌ها (بواسطه نقص فنی ، قطع ارتباط به صورت لحظه‌ای و یا...) کاهش یابد.

منطقی است که در سناریوهای ۱ و ۲، سرعت ماین شدن بلاک‌ها به نسبت قبل بیشتر خواهد شد، زیرا تعداد بیشتری از گره‌ها بر روی ماین بلاک‌ها متمرکز می‌شوند و یا قدرت پردازش در ماین کردن بلاک‌ها در آنها بیشتر می‌شود. و یا در سناریوهای ۳ و ۴، سرعت ماین شدن بلاک‌ها نسبت به قبل کمتر خواهد شد، زیرا تعداد کمتری از گره‌ها و یا سخت افزارها مشغول به فرآیند ماینینگ می‌شوند. به عبارتی شانس پیدا کردن بلاک‌ها با سناریوهای گفته شده دستخوش تغییر می‌شوند.

سختی شبکه باید نسبت به این تغییرات انعطاف پذیر باشد، بدان گونه که در فرآیند ماینینگ سختی شبکه باید به گونه‌ای تنظیم شود که بلاک‌ها در بازه‌های زمانی تقریباً منظمی ماین شده و به بلاکچین افزوده شوند. برای مثال چنانچه بلاکچین مثال بالا در مدت زمان ۵ دقیقه (به صورت متوسط) یک بلاک جدید ماین شده را به بلاکچین خود می‌افزاید، در صورت رخداد سناریوهای ۱ یا ۲ در شبکه، می‌بایست سختی شبکه بالاتر برود تا بمانند قبل بلاک‌ها به صورت حدودی در هر ۵ دقیقه یکبار ماین شوند. یا در صورت وقوع سناریوهای ۳ و ۴ در شبکه، سختی شبکه نیز باید به همان نسبت پایین بیاید.

اما سوالی که مطرح می‌شود آن است که سختی یک شبکه چگونه باید محاسبه شود.
راه حل به طور مستقیم به ۲ خصوصیه بلاکچین ما برمی‌گردد.

- ۱- مدت زمان متوسطی که می‌خواهیم بلاک‌ها ماین شوند. آنرا آلفا (α) می‌نامیم.
- ۲- تعداد بلاک‌هایی که می‌خواهیم هر بار بعد از آنها میزان سختی شبکه را بروز کنیم. آنرا بتا (β) می‌نامیم.

به طور کلی می‌توانیم برای محاسبه سختی شبکه به صورت زیر عمل کنیم:

Determine Network Difficulty:

Next difficulty = (previous difficulty * α * β) / (time to mine last number of blocks)

شکل ۲-۱۵. فرمول تخمین میزان سختی بعدی در شبکه

تصور کنید بلاکچینی داریم که می‌خواهیم روند ماین شدن بلاک‌ها در آن به طور متوسط هر ۵ دقیقه یکبار باشد و همچنین مایل هستیم بعد ماین شدن هر ۱۰۰ تا بلاک، میزان سختی شبکه را بروز کنیم. در این صورت خواهیم داشت.

Next difficulty = (previous difficulty * 5 minutes * 100) / (time to mine last number of 100 blocks)

شکل ۲-۱۶. مثالی برای تخمین میزان سختی بعدی شبکه

۱۰-۲. درک تراکنش‌ها

برای فهم چگونگی انتقال پول‌ها و انجام تراکنش‌ها، نکته حائز اهمیت این است که به طور کلی در این ساختار ایده کلی جا به جایی یک پول دیجیتال است و نه پول واقعی. از این رو نوع نگاه به نحوه شکل دهی ساختاری برای مدیریت و ردیابی پول‌های دیجیتال بسیار با اهمیت است. پول‌های دیجیتال به دو صورت قابل پیاده سازی هستند.

مدل اول پیاده سازی، مدلی در فرم پیاده سازی مبتنی بر حساب^۱ است. در این مدل که اصولاً ساده‌ترین مدل پیاده سازی به شما می‌آید، تراکنش‌ها به صورت رویدادهایی ثبت می‌شوند و نحوه انتقال پول‌ها را شرح

¹ Account Based

می‌دهند به طوری که در حالت کلی یک سیاهه‌ای از تراکنش‌ها بر حسب انتقالات از شخصی به شخص دیگر یا در صورت دقیق‌تر از حسابی به حساب دیگر تشکیل می‌شود.
برای مثال حالت زیر را در نظر بگیرید.

10 coins credit to Ali	(Asserted by miners)
Transfer -> 3 coins from Ali to Sohrab	Signed(Ali)
Transfer -> 2 coins from Sohrab to Hanie	Signed(Sohrab)
Transfer -> 2 coins from Hanie to Ali	Signed(Hanie)
Transfer -> 5 coins from Ali to Sajjad	Signed(Ali)

شکل ۱۷-۲. تاریخچه انتقالات سکه‌ها به صورت سیاهه‌ای از تراکنش‌ها

این شیوه قابل قبولی از نگهداری تراکنش‌های است؛ اما یک مشکل اساسی در آن وجود دارد و آن این است که اگر در این لحظه بپرسیم که سهراب چند سکه دارد، می‌بایست از تاریخ شروع فعالیت سهراب، تراکنش‌های او را رهگیری کنیم و سپس به زمان حال برسیم تا به جواب دست پیدا کنیم. راه بهتری برای ردیابی دارایی‌های افراد در دنیای رمز ارزها وجود دارد که بجای ردیابی^۱ حساب‌ها، به ردیابی سکه‌ها می‌پردازد و سکه‌ها را ردیابی می‌کند. این مدل عموماً به عنوان شیوه‌ای بر اساس دفتر کل^۲ شناخته می‌شود.

در این مدل به صورت دقیق‌تر، تراکنش‌ها بر اساس ورودی و خروجی‌ها ذخیره می‌شوند. این حالت تشخیص تعداد سکه‌های هر فرد را در لحظه بسیار ساده‌تر می‌کند و همچنین امکان چک کردن معتبر بودن تراکنش‌ها را نیز فراهم می‌کند زیرا معماری این نوع از مدل به صورت غیر قابل تغییر^۳ می‌باشد.

¹ Track

² Ledger Based

³ Immutable

حالت زیر چگونگی پیاده سازی این شیوه را بیان می کند.

1	Inputs: \emptyset	Outputs: $20.0 \rightarrow \text{Ali}$
2	Inputs: $1[0]$	Outputs: $17.0 \rightarrow \text{Soheil}, 8.0 \rightarrow \text{Ali}$ Signed(Ali)
3	Inputs: $2[0]$	Outputs: $8.0 \rightarrow \text{Hanie}, 9.0 \rightarrow \text{Soheil}$ Signed(Soheil)
4	Inputs: $2[1]$	Outputs: $6.0 \rightarrow \text{Amir}, 2.0 \rightarrow \text{Ali}$ Signed(Ali)

شکل ۱۸-۲. تاریخچه تراکنش‌ها بر اساس ورودی و خروجی‌ها

در جدول بالا ابتدا هر تراکنش یک شماره به خود می‌گیرد. همچنین هر تراکنش شامل یک ورودی^۱ و یک خروجی^۲ می‌باشد. در ورودی اشاره به شماره تراکنش مرجع می‌شود و در خروجی تقسیم سکه‌ها قابل مشاهده می‌باشد.

همانطور که می‌بینیم، فرض کنید که ۲۰ سکه توسط ماینر^۳‌ها به علی داده شده است.

در تراکنش دوم، مشاهده می‌کنیم $1[0]$ است، یعنی علی می‌خواهد با استفاده از دارایی‌های خود در تراکنش مرجع شماره یک، تقسیم سکه انجام دهد. او در بخش خروجی ۱۷ سکه از ۲۵ سکه خود را به سهیل می‌دهد ($17.0 \rightarrow \text{Soheil}$) و مابقی را نیز برای خود نگه می‌دارد ($8.0 \rightarrow \text{Ali}$) و آنرا با کلید شخصی خود امضا می‌کند.

¹ Input

² Output

³ Miner

در تراکنش سوم، سهیل می‌خواهد با توجه به سکه‌هایی که از تراکنش مرجع شماره ۲ در اختیار دارد (تعداد ۱۷ سکه)، ۸ سکه را به‌هانیه بدهد و ۹ سکه را نیز برای خود نگه دارد.

و در تراکنش چهارم، علی با استفاده از تعداد سکه‌هایی که در تراکنش مرجع شماره ۲ در اختیار دارد، ۶ سکه را به امیر و ۲ سکه را برای خود نگه می‌دارد.

اکنون اگر بپرسیم علی چند سکه دارد، می‌گوییم ۲ سکه. اگر بپرسیم سهیل چند سکه دارد، می‌گوییم ۹ سکه و دیگر افراد را نیز به همین صورت می‌توان مورد بررسی قرار داد.

۱۱-۲. نگهداری‌ها^۱

تا به اینجای کار در کردیم که تراکنش‌ها چگونه در بلاکچین ثبت و ذخیره می‌شوند و همچنین سکه‌ها چگونه منتقل می‌شوند، اما سوالی که مطرح می‌شود این است که چگونه می‌توان سکه‌ها را ذخیره کرد؟ در گذشته بررسی کردیم که هر شخص برای امضای یک تراکنش و مابقی کارها نیاز به دو کلید خصوصی و عمومی داشت، حال باید بگوییم که عملیات‌های ذخیره سازی و انتقال و ارسال پول نیز از طریق کلید خصوصی قابل انجام می‌باشند.

رویکردهای مختلف برای مدیریت کلیدها، مبادلات متفاوتی بین در دسترس بودن، امنیت و راحتی ارائه می‌دهد.

روش‌های مختلفی برای ذخیره سازی سکه‌ها وجود دارد.

۱۱-۲-۱. فضاهای ذخیره سازی متصل به شبکه یا دستگاه^۲

¹ Preservation

² Hot Storages

این فضاهای ذخیره سازی عموماً دستگاه‌ها و دیوایس‌هایی هستند که توسط کامپیوتری پیاده سازی و مدیریت می‌شوند و به شبکه بلاکچین نیز می‌توانند به صورت مستقیم متصل شوند. کیف پول‌های دیجیتال مثال خوبی برای نحوه ذخیره سازی به این صورت هستند.

کیف پول‌های دیجیتال که مرسوم‌ترین نحوه ذخیره سازی ارزهای دیجیتال می‌باشند، می‌توانند به صورت یک برنامه روی کامپیوتر، یک اپلیکیشن موبایل^۲ و یا یک وبسایت^۳ وجود داشته باشند. آنها به این صورت عمل می‌کنند که در هنگام ثبت اطلاعات اولیه در آن و یک پسورد^۴، یک جفت کلید خصوصی و عمومی برای شما می‌سازند. کلید خصوصی توسط برنامه مورد امنیت قرار می‌گیرد و فقط و فقط دارنده کیف پول باید آن را به همراه داشته باشد. از طرف دیگر کلید عمومی برای همگان می‌تواند مورد استفاده قرار بگیرد؛ مثلاً وقتی شخصی آدرس کیف پول خود را به شخص دیگری می‌دهد، در حقیقت مقدار کلید عمومی خودش را با آن شخص به اشتراک می‌گذارد. پس در هنگام جا به جایی یک پول صاحب آن از کلید خصوصی خود استفاده می‌کند اما دریافت پول از طریق داشتن کلید عمومی امکان پذیر است.

طبق این تقاسیر می‌توان ادعا کرد که نگهداری سکه‌ها در حقیقت به نگهداری و مدیریت کلیدهای خصوصی خلاصه می‌شود.

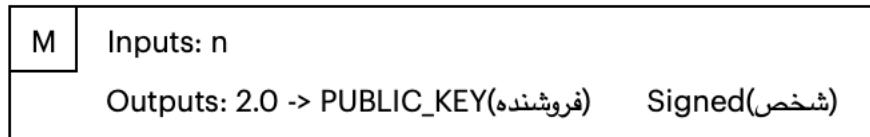
اکنون فرض کنید می‌خواهیم در ازای خرید یک کالا مقداری سکه به فروشنده آن بدھیم، در ابتدا آدرس کیف پول فروشنده که همانطور که گفته شد کلید عمومی کیف پول اوست را دریافت می‌کنیم و در کیف پول دیجیتال خود میزان سکه‌ای را برای این مبادله در نظر می‌گیریم. حال کاری که کیف پول شما انجام می‌دهد این است که با داشتن کلید خصوصی شما، یک درخواست انتقال سکه با کلید عمومی از طرف شما به کلید عمومی فروشنده ایجاد می‌کند.

¹ Device

² Mobile Application

³ Website

⁴ Password



شکل ۲-۱۹. شماتیک یک تراکنش در بلاکچین به همراه ورودی و خروجی‌ها و امضای دارنده سکه

سپس این درخواست را به گره‌های شبکه می‌فرستد، آنها این درخواست را در ممپول خود قرار داده و سپس عملیات استخراج بلاک جدید را آغاز می‌کنند. حال بعد از مدت زمانی، چنانچه بلاک جدیدی توسط گره‌ها استخراج شود که یکی از تراکنش‌های داخل آن تراکنش شما باشد، آن میزان سکه از شما به فروشنده منتقل می‌شود.

کیف پول‌های دیجیتال دارای مزایای دیگری نیز هستند، آنها می‌توانند لیست تراکنش‌های شما شامل ارسال و یا دریافت سکه‌ها، آدرس کیف پول‌های افرادی که قبلاً با آنها تبادل سکه داشته‌ایم و یا قابلیت اسکن^۱ کلید خصوصی افراد از روی متن بر روی یک کاغذ و یا به صورت کد کیو-آر^۲ را مدیریت کنند. اما در نهایت همه چیز به کلید خصوصی در کیف پول‌ها بازمی‌گردد.

۲-۱۱-۲. فضاهای ذخیره سازی ایزوله^۳

این فضاهای ذخیره سازی عموماً ذخیره جفت کلیدها به دور از هر دستگاه کامپیوترا متعلق به شبکه و یا بر روی کامپیوتر می‌باشند. راه و روش‌های مختلفی برای ذخیره داده‌ها به این صورت وجود دارد. می‌توان کلیدها را

¹ Scan

² QR Code

³ Cold Storages

به صورت یک فایل متنی^۱ در داخل یک حافظه خارجی^۲ ذخیره کرد، در حافظه سپرد و یا حتی بر روی کاغذ نوشت و در جایی امن قرار داد. به طور کلی انتخاب روش ذخیره سازی به شخص بازمی‌گردد.

حال باید بدانیم که شکل واقعی کلید خصوصی یا عمومی کلیدها به چه صورت است.

در دنیای دیجیتال و کامپیوتر همه چیز در انتهای تعدادی صفر و یک است و کلیدها هم از این قاعده مستثنی نیستند. آنها به صورت کلی یک رشته بسیار طولانی از صفر و یکها (در مبنای^۳ ۲) هستند که برای راحتی کار عموماً در مبناهای بالاتر مثل مبنای ۵۸ قرار می‌گیرند و خواندن می‌شوند تا هم امکان ساخت تعداد بیشماری کلید و در نتیجه کیف پول فراهم شود و هم از طرفی تعداد کاراکترهای رشته ساخته شده بیش از حد طولانی نشود.

برای مثال شکل یک کلید عمومی می‌تواند به صورت زیر باشد:

Public key:

63FaC9201494f0bd17B9892B9fae4d52fe3BD377

Private key:

8da4ef21b864d2cc526dbdb2a120bd2874c36c9d0a1fb7f8c63d7f7a8b41de8f

۱۲-۲. نتیجه گیری

تا به اینجای کار دریافتیم که یک بلاکچین چگونه ایده پردازی می‌شود، قوانین کلی در بلاکچین‌ها چیست و شبکه‌ها و گره‌ها در سیستم بلاکچین چگونه به هم مرتبط و متصل می‌شوند. همچنین دریافتیم سکه‌ها چگونه توسط اشخاص رمزگذاری، ارسال و یا دریافت می‌شوند. حال وقت آن است که این سیستم را دقیقاً با همین معماری و ساختار طراحی و پیاده سازی کنیم.

¹ Text

² External

فصل ۳. تکنولوژی‌های بکار رفته و ابزارها

۱-۳. مقدمه

در این فصل ابتدا به بررسی و معرفی ابزارهای لازم برای پیاده سازی بلاکچین می‌پردازیم. این ابزارها شامل زبان برنامه نویسی، کتابخانه‌های توسعه نرم افزار، نرم افزارهای ارزیابی قسمت عقب^۱ نرم افزار، چارچوب‌های تست نرم افزار می‌باشد. دیگر بخش‌های این فصل شامل فرآیند توسعه پروژه^۲ و همچنین پارادایم برنامه نویسی^۳ پروژه می‌باشد.

۲-۳. ابزارهای مورد استفاده

۱-۲-۳. جاوا اسکریپت^۴

جاوا اسکریپت [۱۳] نوعی زبان برنامه نویسی سطح بالا^۵ است که دارای ویژگی‌هایی از جمله کامپایل درجاء^۶، چند الگویی^۷، چند پارادایمی و سرعت بالا است. این زبان در کنار HTML [۱۴] و CSS [۱۵]، یکی از

^۱ Back-End

^۲ Project Development

^۳ Programming Paradigm

^۴ JavaScript

^۵ High-Level

^۶ In-Place Compile

^۷ Multiple Patterns

فناوری‌های هسته ای دنیای وب است. بدلیل استفاده کتابخانه‌های معتبر این زبان در دیگر بخش‌های پروژه از جمله قسمت‌های رابط کاربری^۱ از این زبان به عنوان زبان اصلی پروژه استفاده شده است.

۲-۲-۳. کتابخانه ری اکت^۲

ری اکت [۱۶] یک کتابخانه قسمت جلویی^۳ برای زبان جاوا اسکریپت می‌باشد. این کتابخانه رایگان و متن باز^۴ برای ساخت رابطه‌های کاربری است. ری اکت هم اکنون توسط شرکت متا^۵ و جامعه‌ای از توسعه دهندگان^۶ و شرکت‌ها نگهداری می‌شود. از این کتابخانه برای نمایش خروجی‌ها و رابطه‌های کاربری پروژه استفاده شده است.

۳-۲-۳. چارچوب جست^۷

جست [۱۷] یک چارچوب آزمایشی جاوا اسکریپت است که توسط شرکت متا نگهداری می‌شود و توسط سازندگان با تمرکز بر سادگی و پشتیبانی از برنامه‌های کاربردی دنیای وب طراحی و ساخته شده است. از این چارچوب برای تست سلامت کدها و پیشبرد صحیح هر بخش پروژه استفاده شده است.

۴-۲-۳. نرم افزار پستمن^۸

¹ User Interface

² React

³ Front-End

⁴ Open Source

⁵ Meta

⁶ Developers

⁷ Jest Framework

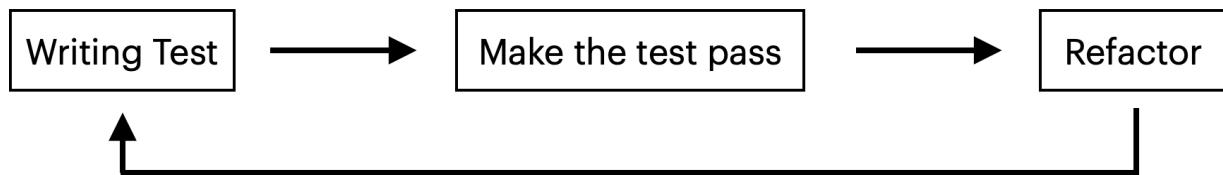
⁸ Postman

پستمن [۱۸] یک پلتفرم^۱ رابط برنامه نویسی کاربردی^۲ برای توسعه دهنده‌گان جهت طراحی، ساخت، آزمایش و تکرار رابطه‌های خود است. از آوریل ۲۰۲۲ پستمن گزارش می‌دهد که بیش از ۲۰ میلیون کاربر ثبت شده و ۷۵۰۰۰ رابط برنامه نویسی کاربردی باز دارد که به گفته او بزرگترین هاب عمومی^۳ جهان را تشکیل می‌دهد.

از این برنامه جهت ارزیابی رابطه‌های برنامه نویسی کاربردی پروژه استفاده خواهیم کرد.

۳-۳. فرآیند توسعه

فرآیند توسعه این پروژه با رویکرد توسعه آزمایش محور^۴ نرم افزار است. در این نوع فرآیند توسعه نرم افزار، ابتدا تست‌هایی برای کدهای اصلی برنامه (که در ابتداء بدون هیچ محتوایی هستند) نوشته می‌شوند. سپس به تدریج سعی می‌شود تا کدها بصورتی تکمیل شوند که در انتهای تمامی تست‌ها را با موفقیت به پایان برسانند.



شکل ۱-۳. شماتیک فرآیند توسعه آزمایش محور نرم افزار

از ویژگی‌های مثبت این نوع رویکرد توسعه می‌توان به موارد زیر اشاره کرد.

¹ Platform

² Application Programming Interface (API)

³ Public Hub

⁴ Test Driven Development

- ۱- طراحی بهتر برنامه و کیفیت بالاتر کدها.
- ۲- ساخت مستندات پروژه به صورت دقیق تر.
- ۳- کاهش زمان مورد نیاز برای توسعه پروژه.
- ۴- انعطاف پذیری کد و نگهداری آسان تر.
- ۵- قابلیت اعتماد و اطمینان بالاتر به کد و نرم افزار نهایی.
- ۶- صرفه جویی در هزینه‌های پروژه در طولانی مدت.

۴-۴. پارادایم برنامه نویسی

پارادایم برنامه نویسی در این پروژه، شی‌گرایی^۱ می‌باشد. در این پارادایم با تمرکز به مدل نرم افزاری کدهای با ویژگی مشترک به شی‌ها سعی بر توسعه برنامه و ساخت پروژه می‌شود.

۴-۵. نتیجه گیری

حال با شناخت ابزارهای لازم برای پیاده سازی پروژه، وقت آن رسیده است تا پروژه را به بخش‌های کوچک شکسته تا بتوانیم در عین راحتی توسعه، نحوه پیشبرد آن را نیز درک کنیم.

¹ Object Oriented Programming

فصل ۴. پیاده سازی

۱-۴. مقدمه

در این فصل، با توجه به دانسته‌ها، قوانین و تئوری‌های موجود در دنیای بلاکچین، ابزارها و یک رایانه ساده فارغ از سیستم عامل و یا سخت افزار خاص سعی در ساخت بلاکچین به صورت بخش به بخش می‌کنیم تا در کنار درک الگوی کار و طراحی بلاکچین، فرآیند نهایی پیاده سازی را تسهیل دهیم.

۲-۴. ساخت بخش اول (بلاک‌ها)

فایل `Block.test.js`

در بخش اول بعد از نصب تکنولوژی‌های لازم، شروع به ساخت فایل `Block.test.js` می‌کنیم. این فایل در اصل یک فایل تست بر محوریت چارچوب `Jest` برای فایل `Block.js` می‌باشد. در `Block.js` ویژگی‌های یک بلاک را توصیف کرده و آنرا پیاده سازی می‌کنیم.

Describe -> Block

it -> has a timestamp, lastHash, hash, and data property

Describe -> Genesis

it -> returns a Block instance

it -> returns the genesis data of genesis block

Describe -> mineBlock

it -> returns a Block instance

it -> sets the `lastHash` to be the `hash` of the last block

```
it -> sets the `data`  
it -> sets a `timestamp`  
it -> creates a SHA-256 `hash` based on the proper inputs  
it -> sets a `hash` that matches the difficulty criteria
```

فایل Block.js

فایل Block.js برای ساخت یک بلاک می‌باشد. رویکرد این بلاک همانطور که در قبل نیز گفته شد، ساخت اسکلت کلی بلاک‌ها در بلاکچین است، به طوری که دارای موارد زیر باشد. در فایل Block.js یک کلاس با نام Block ایجاد می‌کنیم و متدهای constructor آنرا که مقادیر Elements در غالب یک شی (object) می‌گیرد، پیاده سازی می‌کنیم.

```
Class Block {  
    constructor ({ })  
    static genesis ()  
    static mineBlock ()  
}
```

فایل config.js

همانطور که قبلاً گفته شد، در ابتدا برای شروع کار با یک بلاک، نیاز به بلاک اولیه (genesis) داریم. در فایل config.js شروع به پیاده سازی یک بلاک اولیه می‌کنیم. از این بلاک در آینده برای ساخت اولین بلاک از بلاکچین استفاده می‌شود.

```
Const GENESIS_DATA = { Timestamp , lastHash , data , hash }
```

۴-۳. ساخت بخش دوم (بلاکچین)

فایل blockchain.test.js

برای ساخت زنجیره‌ای که بلاک‌ها را به هم متصل کند، ابتدا فایل blockchain.test.js را می‌سازیم و سپس در آن تست‌هایی که باید کد اصلی blockchain.js با موفقیت بگذراند را به ازای رخداد هر سناریو بررسی و پیاده سازی می‌کنیم.

همانطور که در قبل اشاره کردیم، بلاکچین زنجیره‌ای از بلاک‌ها اول آن، بلاک ابتدایی است و همچنین قابلیت اضافه کردن بلاک‌ها به آن وجود دارد.

همچنین بلاکچین می‌بایست در ازای دریافت هر زنجیره (در صورت تعویض زنجیره قبلی با زنجیره طولانی تر)، آن را بررسی کرده و از معتبر بودن آن اطمینان حاصل نماید. این اعتبار سنگی توسطتابع تست isvalidChain بررسی می‌شود و در آن سناریوهای زیر بررسی می‌شوند.

۱- اگر زنجیره مورد نظر با بلاک اولیه (genesis) شروع نشود.

۲- اگر زنجیره مورد نظر با بلاک اولیه شروع شود و همچنین دارای تعدادی بلاک باشد و

۱-۲- هش قبلی بلاک تغییر کرده باشد.

۲-۲- زنجیره دارای بلاک نامعتبر باشد.

۳-۲- زنجیره دارای هیچ مشکل یا نقصی نباشد.

۳- اگر شرایط تعویض زنجیره برقرار باشد، که در آن

۱-۳- زنجیره جدید طولانی‌تر نباشد.

۲-۳- زنجیره جدید طولانی‌تر باشد.

۱-۲-۳- زنجیره معتبر نباشد.

۲-۲-۳- تمامی جزئیات زنجیره معتبر باشد.

Describe -> Blockchain

it -> contains a `chain` array instance

it -> starts with the genesis block as the first block

it -> adds a new block to the chain

Describe -> isValidChain()

Describe -> chain does not starts with genesis block

it -> returns false

Describe -> starts with genesis block and has multiple blocks

Describe -> and a lastHash reference has changed

it -> returns false

Describe -> the chain contains a block with an invalid field

it -> returns false

Describe -> the chain does not contain any invalid blocks

it -> returns true

Describe -> replaceChain()

Describe -> the new chain is not longer

it -> does not replace the chain

it -> logs an error

Describe -> when the new chain is longer

Describe -> and the chain is not valid

it -> does not replace the chain

it -> logs an error

Describe -> and the chain is valid

it -> replace the chain

it -> logs about the chain replacement

blockchain.js فایل

فایل blockchain.js به عنوان فایل اصلی وظیفه پیاده سازی و تشکیل ساختار یک زنجیره از بلاکها را دارد که روند کار آن در فایل blockchain.test.js مشخص شده است.

```
Class Blockchain {  
    constructor ()  
    addBlock ()  
    replaceChain ()  
    static isValidChain ()  
}
```

۴-۴. ساخت بخش سوم (اثبات انجام کار)

Block.test.js فایل

در این بخش سعی بر این شد تا با افزودن ویژگی های دیگر بلاکها، فرآیند اثبات انجام کار در آنها تکمیل و عملی شود. این فرآیند همانطور که در گذشته به آن اشاره شد نیاز به داشتن `difficulty` و `nonce` برای هر بلاک دارد. در فایل block.test.js مقادیر `difficulty` و `nonce` را تعیین می کنیم تا سپس برای هر سناریو از رخدادها، تست ها را طراحی کنیم.

تست هایی که توسط این بخش به فایل block.test.js افزوده شده اند عبارتست از

Describe -> mineBlock ()

it -> sets a `hash` that matches the difficulty criteria

it -> adjusts the difficulty

Describe -> adjustDifficulty ()

it -> raises the difficulty for a quickly mined block

it -> lowers the difficulty for a slowly mined block

it -> has a lower limit of 1

نکته‌ای که در این بخش وجود دارد این است که هنگامی که بلاک‌ها ماین می‌شوند، بسته به سرعت ماین شدن‌شان، سختی ماین کردن بلاک بعدی را تعیین می‌کنند، برای مثال اگر معیار ماین بلاک‌ها در بلاکچین حدود ۱۰ ثانیه بود و بلاکی در مدت زمان ۷ ثانیه ماین شد، سختی شبکه می‌بایست بالا برود و بالعکس؛ تا همواره در حدود همان معیار اولیه بماند.
از طرفی نیز میزان سختی شبکه نباید به عدد منفی تبدیل شود (کمتر از ۱).

Block.js فایل

سپس در فایل `block.js` با توجه به تست‌های تعبیه شده، کد را توسعه می‌دهیم تا بتواند تمامی تست‌ها را با موفقیت پشت سر بگذارد. در این فایل برای متدهای `mineBlock` و `adjustDifficulty` می‌نویسیم تا همواره فرآیند ماین کردن بلاک‌ها را انجام دهد. این متدهای همانطور که در قبیل گفته شد، تنها بلاکی را به عنوان بلاک ماین شده می‌پذیرد که شرط پذیرش بلاک در شبکه را دارا باشد. همچنین در این فایل می‌بایست متدهای `difficulty` و `block` را تعریف کنیم. برای این کار با چک کردن تفاوت مدت زمان معین شبکه برای ماین بلاک و مدت زمان واقعی ماین کردن بلاک، به بیشتر یا کمتر کردن میزان `difficulty` در بلاک می‌پردازیم.

Static adjustDifficulty ({ originalBlock, timestamp })

فایل Blockchain.test.js

در فایل Blockchain.test.js، در بخش مرتبط با چک کردن اعتبار زنجیره بلاک‌ها (isValidChain)، یک تست برای چک کردن تغییر غیر معمول سختی ماین بلاک‌ها می‌نویسیم تا در صورت رخداد این سناریو آن زنجیره را مورد قبول قرار ندهد.

Describe -> and the chain contains a block with a jumped difficulty

it -> returns false

فایل Blockchain.js

در فایل Blockchain.js نیز برای تست‌های نوشته شده، کدها را توسعه می‌دهیم. در این بخش در متدها isValiChain، مقدار قدر مطلق اختلاف سختی آخرین بلاک و سختی بلاک ماین شده را می‌سنجدیم تا همواره برابر با یک باشد. و در غیر این صورت این زنجیره را نمی‌پذیریم.

فایل average-work.js

در این فایل، برای تست ماین شدن بلاک‌ها بدون داده (داشتن شماره بلاک به عنوان داده آن)، به مانیتور کردن سه المان مهم (زمان ماین شدن بلاک، سختی هر بلاک و همچنین میزان متوسط ماین شدن) بلاک‌ها می‌پردازیم تا از روند کار تا به اینجا اطمینان حاصل نماییم.

برای تست فعلی شروع به ماین ۱۰ هزار بلاک می‌کنیم، هدف این است که بلاک‌ها در حدود زمان متوسط یک ثانیه (۱۰۰۰ میلی ثانیه) ماین شوند.

خروجی این فایل به شکل زیر مبیاشد.

```
First block: Block {
  timestamp: 1645202068164,
  lastHash: '_hash_',
  hash: '0932af1d8a367a37cf2e18addc943949ac061085d829f793f146bee8eb0ba78e',
  data: 'initial',
  nonce: 5,
  difficulty: 2
}
Time to mine block 2: 18ms      Difficulty: 3    Average time: 18ms
Time to mine block 3: 0ms       Difficulty: 4    Average time: 9ms
Time to mine block 4: 1ms       Difficulty: 5    Average time: 6.33333333333333ms
Time to mine block 5: 0ms       Difficulty: 6    Average time: 4.75ms
Time to mine block 6: 0ms       Difficulty: 7    Average time: 3.8ms
Time to mine block 7: 24ms      Difficulty: 8    Average time: 7.16666666666667ms
Time to mine block 8: 6ms       Difficulty: 9    Average time: 7ms
Time to mine block 9: 4ms       Difficulty: 10   Average time: 6.625ms
Time to mine block 10: 46ms     Difficulty: 11   Average time: 11ms
Time to mine block 11: 15ms     Difficulty: 12   Average time: 11.4ms
Time to mine block 12: 13ms     Difficulty: 13   Average time: 11.5454545454545ms
Time to mine block 13: 105ms    Difficulty: 14   Average time: 19.3333333333332ms
Time to mine block 14: 179ms    Difficulty: 15   Average time: 31.615384615384617ms
Time to mine block 15: 291ms    Difficulty: 16   Average time: 50.142857142857146ms
Time to mine block 16: 970ms    Difficulty: 17   Average time: 111.46666666666667ms
Time to mine block 17: 649ms    Difficulty: 18   Average time: 145.0625ms
Time to mine block 18: 1972ms   Difficulty: 17   Average time: 252.52941176470588ms
Time to mine block 19: 1206ms   Difficulty: 16   Average time: 305.5ms
Time to mine block 20: 43ms     Difficulty: 17   Average time: 291.6842105263158ms
```

شکل ۴-۱. فرآیند تست استخراج بلاک‌ها قبل از تعادل در زمان استخراج

در ابتدا بلاک‌ها به سرعت یافت می‌شوند، اما به تدریج میزان سختی شبکه هم بالا می‌رود تا به سرعت متعادلی برسند.

سپس بعد از گذشت مدت زمان کوتاهی مشاهده می‌کنیم که مدت زمان متوسط ماین بلاک‌ها (Average time) بسیار نزدیک به ۱۰۰۰ میلی ثانیه می‌باشند.

Time to mine block 331: 76ms	Difficulty: 18	Average time: 995.9819277108434ms
Time to mine block 332: 1588ms	Difficulty: 17	Average time: 997.7597597597597ms
Time to mine block 333: 1307ms	Difficulty: 16	Average time: 998.685628742515ms
Time to mine block 334: 885ms	Difficulty: 17	Average time: 998.3462686567165ms
Time to mine block 335: 1155ms	Difficulty: 16	Average time: 998.8125ms
Time to mine block 336: 793ms	Difficulty: 17	Average time: 998.2017804154302ms
Time to mine block 337: 1055ms	Difficulty: 16	Average time: 998.3698224852071ms
Time to mine block 338: 380ms	Difficulty: 17	Average time: 996.5457227138643ms
Time to mine block 339: 95ms	Difficulty: 18	Average time: 993.8941176470588ms
Time to mine block 340: 42ms	Difficulty: 19	Average time: 991.1026392961877ms
Time to mine block 341: 2565ms	Difficulty: 18	Average time: 995.7046783625731ms
Time to mine block 342: 158ms	Difficulty: 19	Average time: 993.2623906705539ms
Time to mine block 343: 308ms	Difficulty: 20	Average time: 991.2703488372093ms
Time to mine block 344: 2834ms	Difficulty: 19	Average time: 996.6115942028986ms
Time to mine block 345: 3117ms	Difficulty: 18	Average time: 1002.7398843930636ms
Time to mine block 346: 800ms	Difficulty: 19	Average time: 1002.1556195965418ms
Time to mine block 347: 2487ms	Difficulty: 18	Average time: 1006.4224137931035ms
Time to mine block 348: 1530ms	Difficulty: 17	Average time: 1007.9226361031518ms
Time to mine block 349: 1019ms	Difficulty: 16	Average time: 1007.9542857142857ms
Time to mine block 350: 1253ms	Difficulty: 15	Average time: 1008.6524216524217ms
Time to mine block 351: 385ms	Difficulty: 16	Average time: 1006.8806818181819ms
Time to mine block 352: 412ms	Difficulty: 17	Average time: 1005.1954674220963ms
Time to mine block 353: 379ms	Difficulty: 18	Average time: 1003.4265536723163ms
Time to mine block 354: 1134ms	Difficulty: 17	Average time: 1003.7943661971831ms
Time to mine block 355: 317ms	Difficulty: 18	Average time: 1001.8651685393259ms
Time to mine block 356: 2040ms	Difficulty: 17	Average time: 1004.7731092436975ms
Time to mine block 357: 34ms	Difficulty: 18	Average time: 1002.0614525139665ms
Time to mine block 358: 125ms	Difficulty: 19	Average time: 999.6183844011142ms
Time to mine block 359: 1733ms	Difficulty: 18	Average time: 1001.6555555555556ms
Time to mine block 360: 162ms	Difficulty: 19	Average time: 999.3296398891966ms
Time to mine block 361: 3178ms	Difficulty: 18	Average time: 1005.3480662983426ms
Time to mine block 362: 850ms	Difficulty: 19	Average time: 1004.9201101928375ms
Time to mine block 363: 122ms	Difficulty: 20	Average time: 1002.4945054945055ms
Time to mine block 364: 2883ms	Difficulty: 19	Average time: 1007.6465753424658ms
Time to mine block 365: 1355ms	Difficulty: 18	Average time: 1008.5956284153006ms
Time to mine block 366: 1612ms	Difficulty: 17	Average time: 1010.2397820163487ms
Time to mine block 367: 1412ms	Difficulty: 16	Average time: 1011.3315217391304ms
Time to mine block 368: 1480ms	Difficulty: 15	Average time: 1012.6016260162602ms
Time to mine block 369: 48ms	Difficulty: 16	Average time: 1009.9945945945946ms
Time to mine block 370: 213ms	Difficulty: 17	Average time: 1007.8463611859838ms
Time to mine block 371: 120ms	Difficulty: 18	Average time: 1005.4596774193549ms
Time to mine block 372: 582ms	Difficulty: 19	Average time: 1004.3243967828419ms
Time to mine block 373: 1423ms	Difficulty: 18	Average time: 1005.4438502673797ms
Time to mine block 374: 808ms	Difficulty: 19	Average time: 1004.9173333333333ms
Time to mine block 375: 31ms	Difficulty: 20	Average time: 1002.3271276595744ms
Time to mine block 376: 1418ms	Difficulty: 19	Average time: 1003.4297082228117ms
Time to mine block 377: 1144ms	Difficulty: 18	Average time: 1003.8015873015873ms
Time to mine block 378: 473ms	Difficulty: 19	Average time: 1002.4010554089709ms
Time to mine block 379: 963ms	Difficulty: 20	Average time: 1002.2973684210526ms
Time to mine block 380: 4679ms	Difficulty: 19	Average time: 1011.9475065616798ms
Time to mine block 381: 3695ms	Difficulty: 18	Average time: 1018.9712041884817ms

شكل ۲-۴. فرآیند تست استخراج بلاک‌ها پس از تعادل در زمان استخراج

۴-۵. ساخت بخش چهارم (پیاده سازی شبکه و رابط برنامه نویسی کاربردی)

فایل index.js

در فایل index.js که در دایرکتوری root پروژه قرار دارد و به صورت خودکار به عنوان فایل اجرا کننده پروژه صدا زده می‌شود، حاوی تمامی api‌های است که از آنها برای ارتباط با شبکه بلاکچین استفاده می‌شود. رویکرد کلی در فایل index.js این است که یک اپلیکیشن توسط فریمورک express با نام app ساخته می‌شود تا توسط آن متدهای get و post برای دریافت و ارسال اطلاعات بر روی شبکه امکان پذیر باشد. در این فایل تعداد ۴ عدد api استفاده می‌شود که عبارتند از

۱- پخش کردن بلاک‌های موجود و ثبت شده در بلاکچین

```
app.get('/api/blocks', (req, res) => { });
```

۲- ساخت و پخش کردن اطلاعات ماین یک بلاک و فرستادن آن به مابقی گره‌ها

```
app.post('/api/mine', (req, res) => { });
```

۳- ساخت و پخش کردن داده‌های تراکنش‌های ثبت شده

```
app.post('/api/transact', (req, res) => { });
```

۴- اطلاع از تراکنش‌های ثبت شده و آماده به ماین در شبکه

```
app.post('/api/transaction-pool-map', (req, res) => { });
```

و در نهایت توسط متدهای listen، تمامی گره‌ها با اجرای برنامه بر روی پورت مورد نظر شروع به فعالیت و اتصال به یکدیگر می‌کنند.

```
app.listen(PORT, () => { });
```

pubsub.js فایل

در فایل pubsub.js، شروع به ساخت کانال (channel) برای ارسال و دریافت اطلاعات می‌کنیم. رویکرد کلی در این فایل استفاده از دیتابیس redis برای ساخت کانال‌هایی بر بستر شبکه است به گونه‌ای که هر زمان گره‌ای به شبکه اضافه شود، می‌تواند با مابقی گره‌ها از طریق این کانال‌ها در ارتباط باشند. آنها وقتی اطلاعاتی را روی این کانال‌ها ارسال کنند، هر گره‌ای که عضو این کانال‌ها باشد (subscribe کرده باشد) می‌تواند آن اطلاعات را دریافت کند. مزیت استفاده از این رویکرد این است که گره‌ها بدون دانستن آدرس گره فرستنده، اطلاعات را دریافت می‌کنند و همچنین گره فرستنده بدون دانستن آدرس گره‌های گیرنده، اطلاعات را تنها بر روی کانال به اشتراک می‌گذارد.

```
Const redis = require('redis');

Const CHANNELS = {

    BLOCKCHAIN,
    TRANSACTIONS

}

Class PubSub {

    constructor( );

    handleMessage( channel, message );

    subscribeToChannels( );

    publish( channel, message );

    broadcastChain( );

    broadcastTransactions( transaction );

}

}
```

۶-۶. ساخت بخش پنجم (کیف پول، کلیدها و تراکنش‌ها)

فایل index.test.js

در فایل index.js شروع به نوشتتن تست‌هایی می‌کنیم که قرار است وضعیت یک کیف پول (wallet) را در هر سناریویی شرح دهند.

Describe -> Wallet

it -> has a balance

it -> has a public key

Describe -> signing data

it -> verifies a signature

it -> does not verify an invalid signature

Describe -> createTransaction ()

Describe -> and the amount is more than balance

it -> throws an error

Describe -> and the amount is valid

it -> creates an instance of `Transaction`

it -> matches the transaction input with the wallet

it -> outputs the amount the recipient

فایل index.js

سپس در فایل index.js شروع به نوشتتن کد می‌کنیم به گونه‌ای که تست‌های نوشته شده در فایل index.test.js را با موفقیت پشت سر بگذراند.

```
Class Wallet {  
    constructor ()  
    sing ( data )  
    createTransaction({ recipient, amount })  
}  
}
```

فایل transaction.test.js

در فایل transaction.test.js نیز سناریوهای یک تراکنش را مورد بررسی قرار می‌دهیم. این سناریوها همچنین شامل چک کردن اعتبار تراکنش‌ها و همچنین بروز رسانی تراکنش‌ها بعد ثبت یک یا چند تراکنش می‌باشد.

Describe -> Transaction

it -> has an `id`

Describe -> outputMap

it -> has an `outputMap`

it -> outputs the amount to the recipient

it -> outputs the remaining balance for the `senderWallet`

Describe -> input

it -> has an `input`

it -> has a `timestamp` in the input

it -> sets the `amount` to the `senderWallet` balance

it -> sets the `address` to the `senderWallet` public key

it -> signs the input with the `senderWallet`

Describe -> validTransaction ()

Describe -> when transaction is valid

it -> returns true

Describe -> when transaction is not valid

Describe -> and a transaction outputMap value is not valid

it -> returns false and logs an error

Describe -> and a transaction input signature is not valid

it -> returns false and logs an error

Describe -> update ()

Describe -> and the amount is invalid

it -> throws an error

Describe -> and the amount is valid

it -> outputs the amount to the next recipient

it -> subtracts the amount from the original sender output amount

it -> maintains a total that matches the input amount

it -> re-signs the transaction

Describe -> and another update for the same recipient

it -> adds to the recipient amount

it -> should subtracts the amount from the original sender output amount

transaction.js فایل

در فایل transaction.js نیز یک متاد سازنده به همراه ساختار اصلی تراکنش‌ها (ساخت خروجی، ورودی، بروز رسانی و چک کردن هر یک از تراکنش‌ها) می‌باشد.

```
Class Transaction {  
  constructor ( senderWallet, recipient, amount )  
  
  createOutputMap ({ senderWallet, recipient, amount })  
  createInput ({ senderWallet, outputMap })  
  update ({ senderWallet, recipient, amount })  
  static validTransaction ( transaction )  
}
```

۴-۷. ساخت بخش ششم (استخراج تراکنش‌ها)

فایل transaction-pool.test.js

در فایل transaction-pool.test.js سعی در ساخت تست‌هایی جهت توصیف یک استخراج از تراکنش‌ها می‌کنیم که بعد از ثبت شدن، می‌بایست معتبر باشند و در بلاکچین ذخیره و سپس از استخراج به طور کامل حذف شوند.

Describe -> TransactionPool

Describe -> setTransaction ()

it -> adds a transaction

Describe -> existingTransaction

it -> returns an existing transaction given an input address

Describe -> validTransactions ()

it -> returns valid transaction

it -> logs errors for invalid transactions

Describe -> clear ()

it -> clears the transaction pool

Describe -> clearBlockchainTransactions ()

it -> clears the pool of any existing blockchain transactions

فایل transaction-pool.js

در فایل transaction-pool.js نیز شروع به ساخت کلاس transactionPool می‌کنیم و سپس متدهای داخل کلاس آن را به نحوی طراحی می‌کنیم که تست‌های تعبیه شده در مرحله قبل را با موفقیت به پایان برساند.

```
Class TransactionPool {  
    constructor ()  
    clear ()  
    setTransaction ( transaction )  
    setMap ( transactionMap )  
    existingTransaction({ inputAddress })  
    validTransactions ()  
    clearBlockchainTransactions ({ chain })  
}
```

فایل index.js

در این فایل نیز سپس app را ساخته و توسط transaction-pool-map آنرا با متدهای سازی می‌کنیم.

```
app.get ('/api/transaction-pool-map', ( req, res ) => { });
```

در ادامه چند خروجی از قسمت api‌های پروژه با استفاده از ابزار پستمن را مشاهده می‌کنیم.

تصویر اول : توسط /api/transact یک تراکنش با نام گیرنده و مقدار سکه مبادله شده را ارسال می‌کنیم. سپس در قسمت پایین عکس، خروجی آن را به فرمت یک تراکنش کامل باز می‌گردانیم تا سپس برای ثبت در بلاکچین به استخراج تراکنش‌ها ارسال شود.

The screenshot shows the Postman interface with a red box highlighting the URL field and another red box highlighting the JSON response body. The URL is `http://localhost:3000/api/transact`. The JSON body sent is:

```
1
2   ...
3     "recipient": "SajjadHaghzad",
4     "amount": 3
```

The response body is a detailed JSON object representing a transaction:

```
1
2   "transaction": {
3     "id": "1cf785a0-964b-11ec-816e-6132f98bd719",
4     "outputMap": {
5       "undefined": 3,
6       "04c5bd9ec3b931c839fddee61606c08f27bdd412859e57e0d9d15972be195a15675ab105ab582fb4eae3dc603f82cec263b914f5d9ed90edca5d725eb7e27e056": 5
7     },
8     "input": {
9       "timestamp": 1645800989434,
10      "amount": 8,
11      "address": "04c5bd9ec3b931c839fddee61606c08f27bdd412859e57e0d9d15972be195a15675ab105ab582fb4eae3dc603f82cec263b914f5d9ed90edca5d725eb7e27e056",
12      "signature": {
13        "r": "d1b9229cc513b4a5d36e69347d6d87077c604bac327c7763bd324686f34eee8",
14        "s": "cf48e5e9ac909fd423410174147b858a9e9570b162ddfa7001d66783a296c4cc",
15        "recoveryParam": 1
16      }
17    }
18  }
19 }
```

شکل ۳-۴. انجام یک تراکنش در پستمن

تصویر دوم : چنانچه میزان سکه مبادله شده بیش از حد دارایی کیف پول شخص باشد، با پیغام خطای مواجه خواهیم شد.

The screenshot shows a POST request to `http://localhost:3000/api/transact`. The request body contains the following JSON:

```
1 {
2   "recipient": "SajjadHaghzad",
3   "amount": 100000
4 }
```

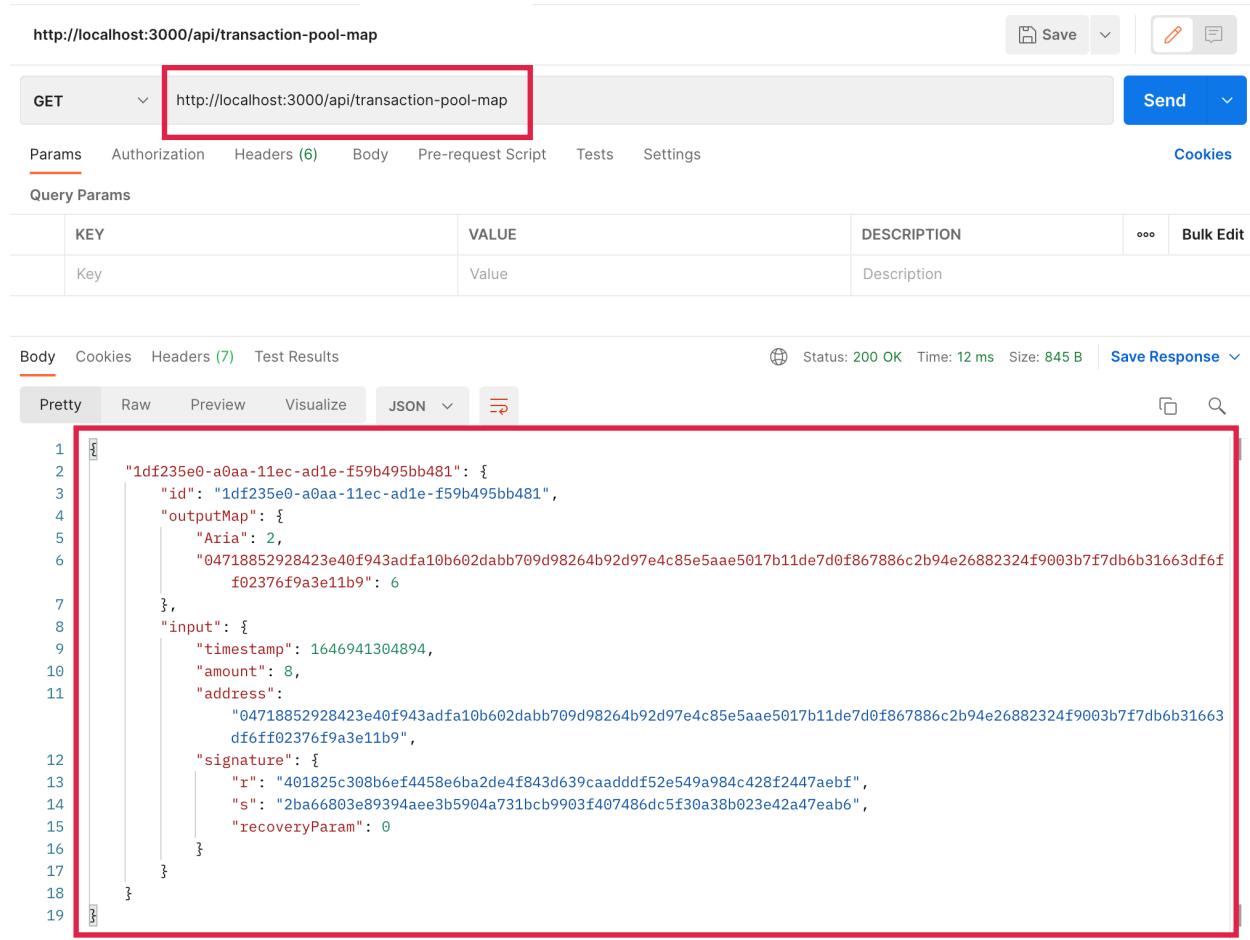
The response is a 400 Bad Request with the following JSON message:

```
1 {
2   "type": "error",
3   "message": "Amount exceeds balance"
4 }
```

شکل ۴-۴. انجام تراکنش بدون داشتن سکه کافی در پستمن

تصویر سوم : در این بخش نیز با فراخوانی `/api/transaction-pool-map` مشاهده می کنیم که یک تراکنش با `id` تراکنش درخواستی در بخش قبل به استخراج تراکنشها فرستاده شده و آماده ماین و ثبت در بلاکچین می باشد.

همانطور که در قبل نیز گفته شد، تراکنشها به طور کلی شامل یک `id`، `input` و `outputMap` هستند که اجزای داخلی آنها جزئیات تراکنش را نشان می دهند.



The screenshot shows a Postman interface with the following details:

- URL:** `http://localhost:3000/api/transaction-pool-map`
- Method:** GET
- Headers:** (6)
- Body:** (Raw, JSON, Preview, Visualize)
- Query Params:** (Key: Value, Description)
- Response Headers:** Status: 200 OK, Time: 12 ms, Size: 845 B
- Response Body (JSON):**

```

1 "1df235e0-a0aa-11ec-ad1e-f59b495bb481": {
2   "id": "1df235e0-a0aa-11ec-ad1e-f59b495bb481",
3   "outputMap": {
4     "Aria": 2,
5     "04718852928423e40f943adfa10b602dabb709d98264b92d97e4c85e5aae5017b11de7d0f867886c2b94e26882324f9003b7f7db6b31663df6f
6       f02376f9a3e11b9": 6
7   },
8   "input": {
9     "timestamp": 1646941304894,
10    "amount": 8,
11    "address":
12      "04718852928423e40f943adfa10b602dabb709d98264b92d97e4c85e5aae5017b11de7d0f867886c2b94e26882324f9003b7f7db6b31663
13        df6ff02376f9a3e11b9",
14    "signature": {
15      "r": "401825c308b6ef4458e6ba2de4f843d639caaddf52e549a984c428f2447aebf",
16      "s": "2ba66803e89394aee3b5904a731bcb9903f407486dc5f30a38b023e42a47eab6",
17      "recoveryParam": 0
18    }
19  }

```

شكل ۴-۵. خروجی استخراج تراکنشها در پستمن

۴-۸. ساخت بخش هفتم (استخراج تراکنش‌ها)

فایل transaction-miner.js

ابتدا فایل transaction-miner.js ایجاد کرده، سپس یک کلاس با نام TransactionMiner می‌کنیم که متدهای خود را دارد.

```
class TransactionMiner {  
  constructor({ blockchain, transactionPool, wallet, pubsub }) {}  
  mineTransactions() {}  
}
```

فایل transaction-pool.test.js

در فایل transaction-pool.js تست‌های لازم را می‌نویسیم، از جمله بررسی معتبر بودن یک تراکنش، خالی کردن استخراج تراکنش‌ها و همچنین خالی کردن تراکنش‌های ثبت شده در بلاکچین.

Describe -> validTransactions()

it -> returns valid transactions

it -> logs errors for invalid transactions

Describe -> clear()

it -> clears the transaction pool

Describe -> clearBlockchainTransactions()

it -> clear the pool of any existing blockchain transactions

config.js فایل

در فایل config.js نیز یک قرارداد ثبت می کنیم به گونه ای که ورودی یک تراکنش پاداشی به صورت زیر باشد. همچنین مقدار سکه پاداشی جهت استخراج یک بلاک ۱۶ تا باشد.

```
const REWARD_INPUT = {  
    address: '*authorized-reward'*  
};  
  
const MINING_REWARD = 16;
```

transaction.test.js فایل

تست های بخش پاداش استخراج بلاک ها هم در این بخش نهادینه می شود.

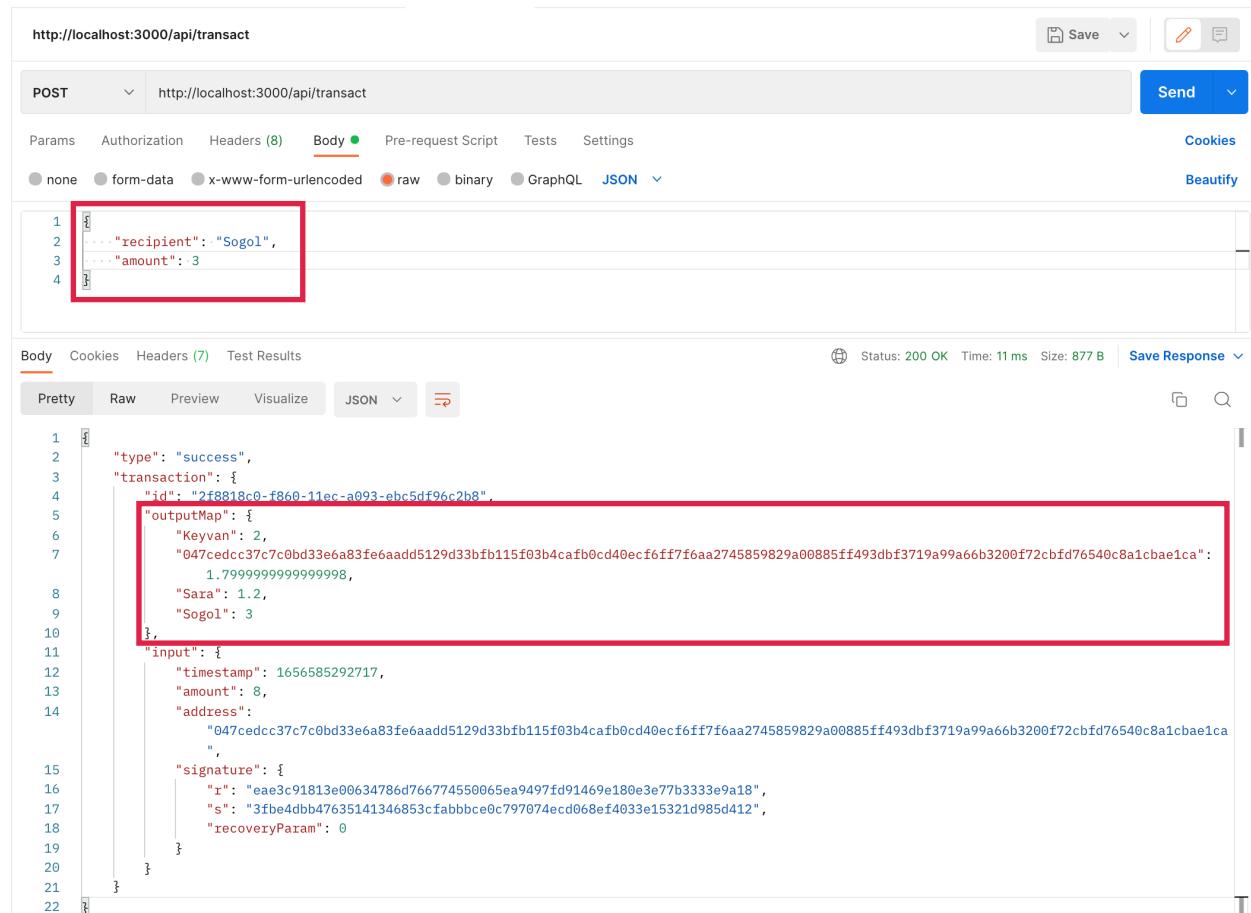
```
describe -> rewardTransaction( )  
    it -> creates a transaction with the reward input  
    it -> creates a transaction for the miner with the `MINING_REWARD`
```

transaction.js فایل

در فایل transaction.js نیز با متدهای پاداش استخراج بلاک را به کمک کیف پول گره مورد نظر می نویسیم.

```
Static rewardTransactions ( { minerWallet } )
```

حال برای ارزیابی این بخش، ابتدا بر روی گره اصلی که اجرا کرده‌ایم چند تراکنش توسط می‌سازیم.



The screenshot shows a Postman request to `http://localhost:3000/api/transact` with a POST method. The request body is JSON:

```

1 {
2   "recipient": "Sogol",
3   "amount": 3
4 }

```

The response status is 200 OK, with a timestamp of 11 ms and a size of 877 B. The response body is:

```

1 {
2   "type": "success",
3   "transaction": {
4     "id": "2f8818c0-f860-11ec-a093-ebc5df96c2b8",
5     "outputMap": {
6       "Keyvan": 2,
7         "047cedcc37c0bd33e6a83fe6aadd5129d33fb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbae1ca": 1.7999999999999998,
8       "Sara": 1.2,
9       "Sogol": 3
10    },
11   "input": {
12     "timestamp": 1656585292717,
13     "amount": 8,
14     "address": "047cedcc37c0bd33e6a83fe6aadd5129d33fb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbae1ca",
15     "signature": {
16       "r": "eae3c91813e00634786d766774550065ea9497fd91469e180e3e77b3333e9a18",
17       "s": "3fbe4dbb47635141346853cfabbce0c797074ecd068ef4033e15321d985d412",
18       "recoveryParam": 0
19     }
20   }
21 }

```

شکل ۴-۶. خروجی‌های نقل و انتقالات سکه‌ها در قالب یک تراکنش در پستمن

سپس با بررسی استخراج تراکنش‌های در حال انتظار توسط `/api/transaction-pool-map` از قرار داشتن تراکنش مورد نظر در استخراج، اطمینان حاصل می‌کنیم.

```

1 "2f8818c0-f860-11ec-a093-ebc5df96c2b8": {
2   "id": "2f8818c0-f860-11ec-a093-ebc5df96c2b8",
3   "outputMap": {
4     "Keyvan": 2,
5     "047cedcc37c0bd33e6a83fe6aadd5129d33bfb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbae1ca": 1.7999999999999998,
6     "Sara": 1.2,
7     "Sogol": 3
8   },
9   "input": {
10     "timestamp": 1656585292717,
11     "amount": 8,
12     "address": "047cedcc37c0bd33e6a83fe6aadd5129d33bfb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbae1ca"
13     ,
14     "signature": {
15       "r": "eae3c91813e00634786d766774550065ea9497fd91469e180e3e77b3333e9a18",
16       "s": "3fbe4db47635141346853cfabbbe0c797074ecd068ef4033e15321d985d412",
17       "recoveryParam": 0
18     }
19   }
20 }
21 
```

شکل ۴-۷. محتوای استخراج تراکنش‌ها با وجود تراکنش مورد نظر در پستمن

سپس با سعی در استخراج تراکنش‌های داخل استخراج توسط `/api/mine-transactions` سعی در پیدا کردن بلاک و قرار دادن آن در بلاکچین می‌کنیم.

در نهایت مشاهده می‌کنیم که کل اطلاعات لازم از تراکنش از جمله یک خروجی (`outputMap`) در آخر بلاک وجود دارد که در آن تراکنش پاداش دار برای کیف پولی که این بلاک را استخراج کرده، نهادینه شده است.

http://localhost:3000/api/mine-transactions

```

7   "nonce": 0,
8   "difficulty": 3
9 },
10 {
11   "timestamp": 1656585340487,
12   "lastHash": "_hash_",
13   "hash": "23f83bb87937f50102bb6ca11d18aab8c187ec68e9250a06292843c15168206",
14   "data": [
15     {
16       "id": "2f8818c0-f860-11ec-a093-ebc5df96c2b8",
17       "outputMap": {
18         "Keyvan": 2,
19         "O47cedcc37c7c0bd33e6a83fe6aadd5129d33bfb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbaelca": 1.7999999999999998,
20         "Sara": 1.2,
21         "Sogol": 3
22       },
23       "input": {
24         "timestamp": 1656585292717,
25         "amount": 8,
26         "address": "O47cedcc37c7c0bd33e6a83fe6aadd5129d33bfb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbaelca",
27         "signature": {
28           "r": "ae3c91813e00634786d766774550065ea9497fd91469e180e3e77b3333e9a18",
29           "s": "3fbe4dbd47635141346853cfabbce0c797074ecd068ef4033e15321d985d412",
30           "recoveryParam": 0
31         }
32       }
33     },
34     {
35       "id": "631a7570-f860-11ec-a093-ebc5df96c2b8",
36       "outputMap": {
37         "O47cedcc37c7c0bd33e6a83fe6aadd5129d33bfb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbaelca": 16
38       },
39       "input": {
40         "address": "*authorized-reward*"
41       }
42     }
43   ],
44   "nonce": 1,
45   "difficulty": 2
46 }
47

```

شکل ۴-۸ ثبت بلاک با نقل و انتقالات مورد نظر در بلاکچین در پستمن

سپس اگر دوباره به بخش استخراج تراکنش‌ها رجوع کنیم، خواهیم دید که بعد از هر قرارگیری موفق یک بلاک در بلاکچین، استخراج خالی از تراکنش‌های گذشته می‌شود.

The screenshot shows a Postman interface with the following details:

- URL:** `http://localhost:3000/api/transaction-pool-map`
- Method:** GET
- Headers:** (6)
- Body:** (Empty)
- Response Status:** 200 OK
- Response Body:** `{}`

شکل ۴-۹. شماتیک خالی بودن استخراج بعد از ثبت تراکنش‌ها در بلاک در پستمن

و اگر توسط `/api/wallet-info` درخواستی را توسط متده `get` ارسال کنیم، آدرس کیف پول ما به همراه میزان موجودی آن، قابل مشاهده خواهد بود.

The screenshot shows a Postman interface with the following details:

- URL:** `http://localhost:3000/api/wallet-info`
- Method:** GET
- Headers:** (6)
- Body:** (Empty)
- Response Status:** 200 OK
- Response Body:**

```

1  {
2    "address": "047cedcc37c7c0bd33e6a83fe6aadd5129d33fb115f03b4caf0cd40ecf6ff7f6aa2745859829a00885ff493dbf3719a99a66b3200f72cbfd76540c8a1cbae1ca",
3    "balance": 17.8
4  }

```

شکل ۴-۱۰. نمایش اطلاعات کیف پول در پستمن

۹-۴. ساخت بخش هشتم (کار بر روی بخش جلویی)

در این بخش، سعی در ساخت بخش جلویی پروژه (محتویات قابل مشاهده توسط کاربر و ابزارهای اصلی) می‌کنیم. این قسمت توسط کتابخانه ری اکت بوجود آمده است. کامپوننت^۱های این بخش شامل موارد زیر است.

App.js - ۱

Block.js - ۲

Blocks.js - ۳

ConductTransactions.js - ۴

Transaction.js - ۵

TransactionPool.js - ۶

فایل اصلی نیز index.js است که روابط صفحه اصلی را می‌سازد و تمام مسیریابی بین صفحه‌ها را انجام می‌دهد.

همچنین فایل index.html و index.css نیز برای نمایش محتویات صفحه اصلی می‌باشند و از فایل history.js نیز برای ثبت مسیریابی بین صفحات کاربرد و جلو و یا عقب رفتن بین صفحات استفاده می‌شود. در بخش پیش رو، ابتدا یک گره را با دستور "npm run dev" ایجاد می‌کنیم.

سپس برای نمایش تعامل در بلاکچین، یک گره دیگر را نیز با دستور "npm run dev-peer" به شبکه می‌افزاییم.

سپس می‌توانیم با استفاده از مرورگر و آدرس‌های localhost:۳۰۰۰ برای گره اول و localhost:۳۴۰۳ برای گره دوم، به پنل این دو گره و اطلاعات کیف پول آنها دست پیدا کنیم.

^۱ Component

```

→ Tyche_Blockchain git:(main) ✘ npm run dev

> Tyche_Blockchain@1.0.0 dev /Users/aria/VisualStudioProjects/Tyche_Blockchain
> npm run dev-client & npm run start-redis && nodemon index.js

> Tyche_Blockchain@1.0.0 dev-client /Users/aria/VisualStudioProjects/Tyche_Blockchain
> npm run clean && parcel client/src/index.html --out-dir client/dist

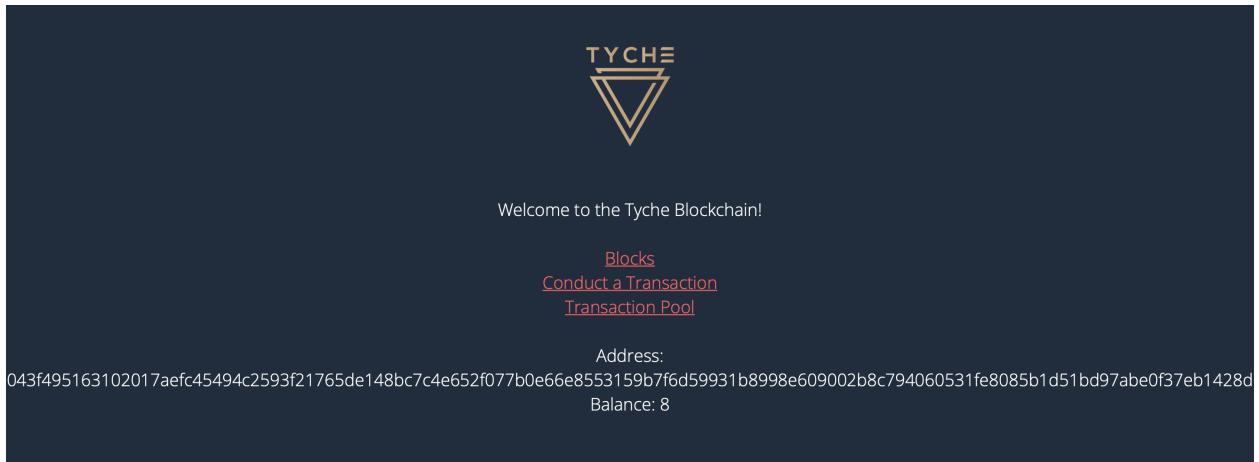
> Tyche_Blockchain@1.0.0 start-redis /Users/aria/VisualStudioProjects/Tyche_Blockchain
> redis-server --daemonize yes

> Tyche_Blockchain@1.0.0 clean /Users/aria/VisualStudioProjects/Tyche_Blockchain
> rm -rf .cache client/dist

[nodemon] 1.18.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: '*.*'
[nodemon] starting `node index.js`
Listening at http://localhost:3000
Server running at http://localhost:1234
:: Building index.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Building index.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Building scheduler.development.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Building capitalize.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Building react-is.development.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Building react-is.development.js...Error: ENOENT: no such file or directory, stat '/Users/aria/VisualStudioProjects/Tyche_Blockchain/client/dist/index.html'
:: Packaging...[nodemon] restarting due to changes...
:: Building hmr-runtime.js...[nodemon] starting `node index.js`
[nodemon] restarting due to changes...
[nodemon] starting `node index.js`
Built in 10.14s
Listening at http://localhost:3000

```

شکل ۱۱-۴. شماتیک اجرای گره شماره یک در ترمینال



شکل ۱۲-۴. شماتیک محیط کاربری گره شماره یک در مرورگر

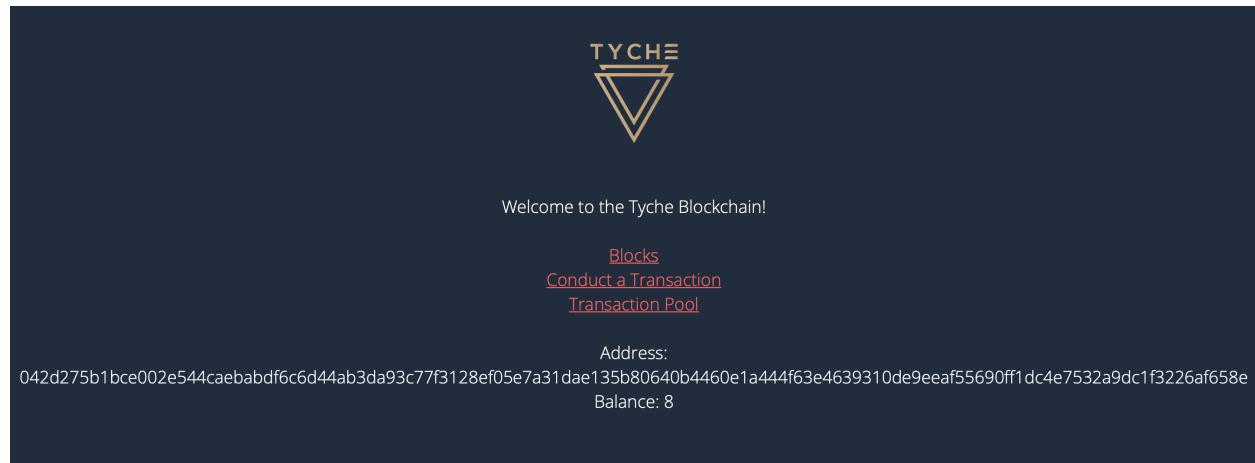
```

→ Tyche_Blockchain git:(main) ✘ npm run dev-peer
> Tyche_Blockchain@1.0.0 dev-peer /Users/aria/VisualStudioProjects/Tyche_Blockchain
> cross-env GENERATE_PEER_PORT='true' nodemon index.js

[nodemon] 1.18.4
[nodemon] to restart at any time, enter `rs`
[nodemon] watching: ***!
[nodemon] starting `node index.js`
Listening at http://localhost:3403
replace chain on a sync with [
  {
    timestamp: 1,
    lastHash: '----',
    hash: '_hash_',
    data: [],
    nonce: 0,
    difficulty: 3
  }
]
The incoming chain must be longer
replace transaction pool map on a sync with {}

```

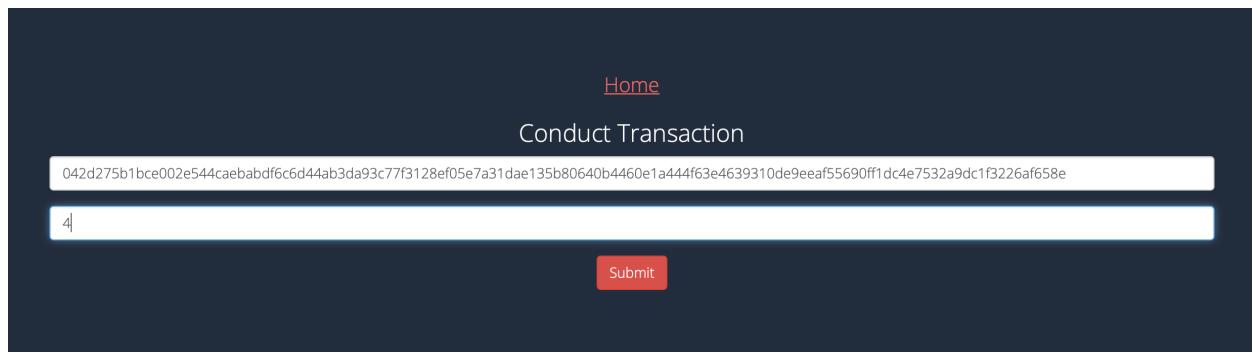
شکل ۱۳-۴. شماتیک اجرای گره شماره دو در ترمینال



شکل ۱۴-۴. شماتیک محیط کاربری گره شماره یک در مرورگر

حال می خواهیم برای تست، یک تراکنش از گره شماره یک به گره شماره دو، با مقدار ۴ سکه انجام دهیم.
برای اینکار توسط لینک Conduct a Transaction وارد صفحه ساخت تراکنشها می شویم.

سپس در نوار بالایی آدرس کیف پول گره شماره دو را در بخش recipient وارد کرده و در نوار پایینی میزان سکه مورد نظر برای جایی را وارد می کنیم.



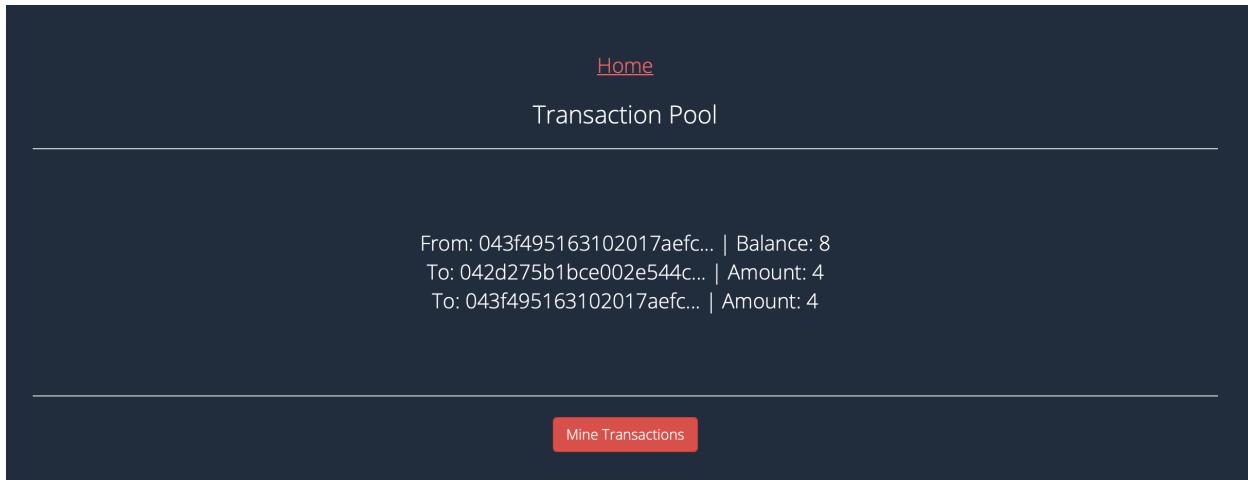
شکل ۱۵-۴. وارد کردن اطلاعات یک تراکنش از گره یک به گره دو

سپس چنانچه همه شرایط برای ارسال سکه مهیا باشد (قوانين گفته شده از قبل)، یک اعلان بر روی صفحه نمایش داده خواهد شد که ادعا بر موفقیت آمیز بودن ثبت تراکنش دارد.



شکل ۱۶-۴. اعلان موفقیت آمیز بودن ثبت تراکنش

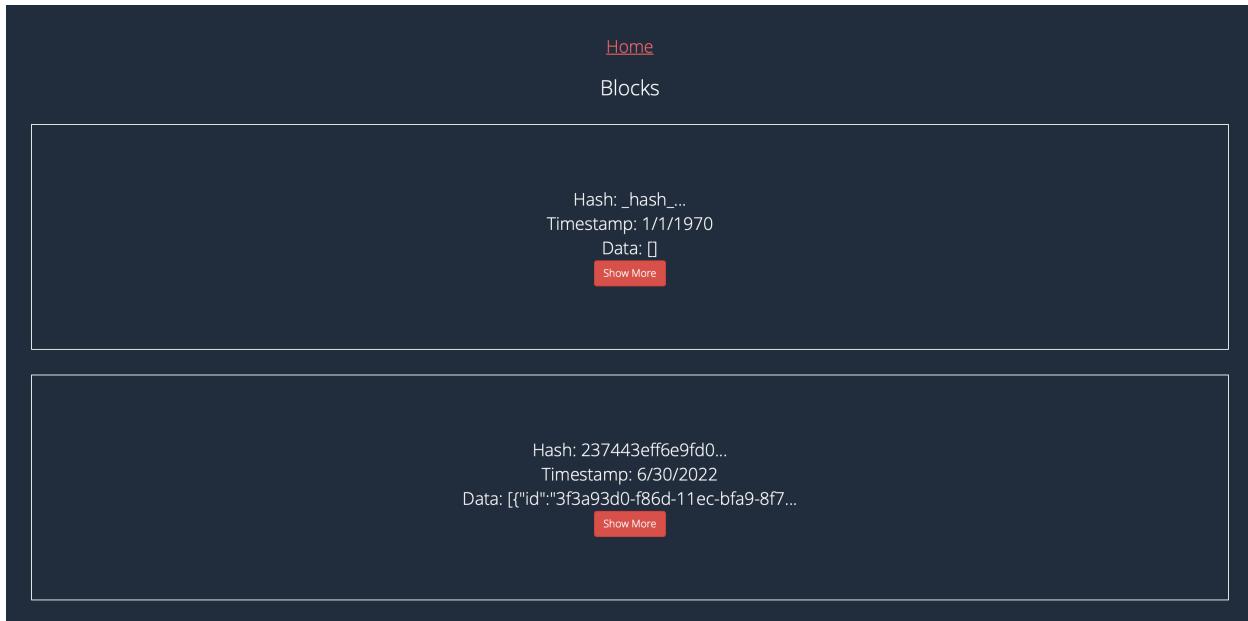
حال برنامه خود به خود به بخش استخراج تراکنشها مسیریابی خواهد شد که در آنجا تراکنشهای درخواستی برای ثبت به صورت زنده در بلاکچین وجود دارند. در این بخش، اطلاعات آدرس فرستنده، گیرنده و میزان سکه مبادله شده وجود دارد.



شکل ۱۷-۴. نمایش استخر تراکنش‌ها به صورت زنده

حال هر گره‌ای که بتواند زودتر تراکنش‌های موجود در استخر تراکنش‌ها را استخراج کند، بلاک جدیدی را در بلاکچین ثبت کرده است.

صفحه بلاک‌های ثبت شده در بلاکچین توسط لینک Blocks قابل دسترس خواهد بود.



شکل ۱۸-۴. صفحه بلاک‌های ثبت شده در بلاکچین

در این بخش می‌توان با زدن دکمه Show More، اطلاعات بیشتری را از هر بلاک ثبت شده در بلاکچین کسب کرد.

این اطلاعات شامل، هش بلاک، زمان ساخت بلاک، تراکنش‌های ثبت شده در هر بلاک و تراکنش پاداش می‌باشد.

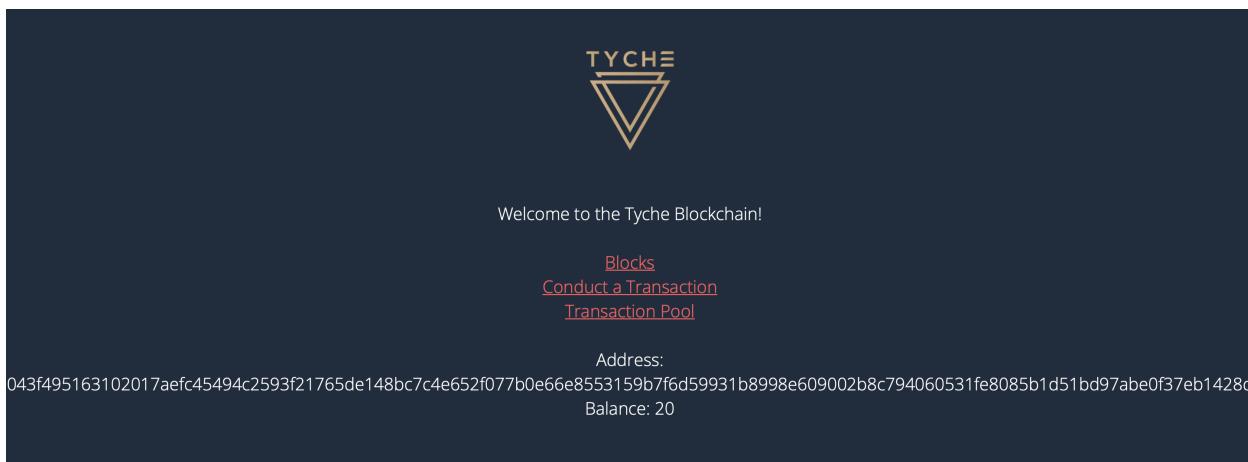
لازم به ذکر است که در این سناریو، خود گره شماره یک بلاک را استخراج کرده است و از این جهت، تراکنش پاداشی (شامل ۱۶ سکه) به آدرس آن گره تعلق گرفته است.



شکل ۴-۱۹. شماتیک بلاک در بلاکچین با جزئیات بیشتر

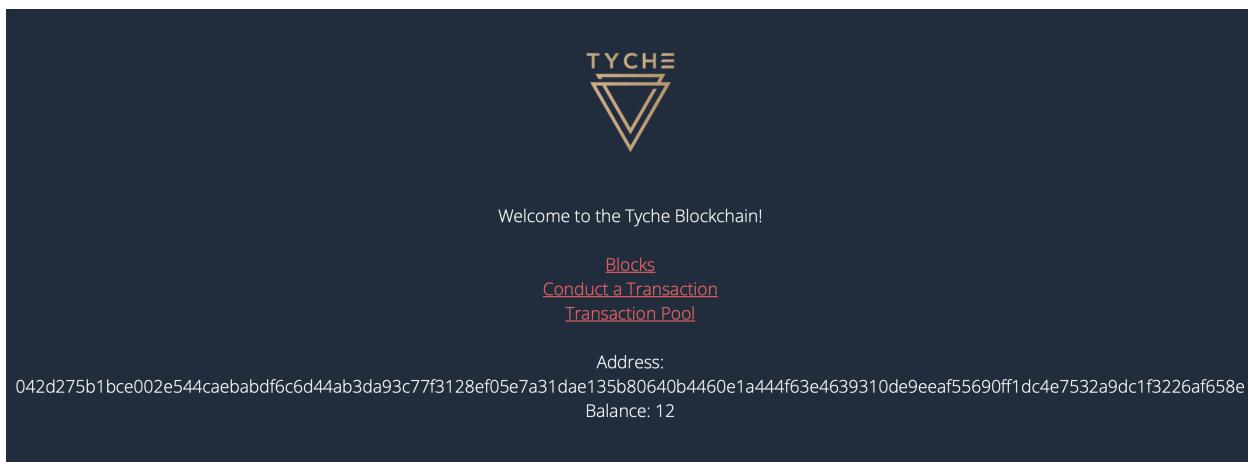
حال اگر به صفحه اصلی هر یک از گره‌ها برویم، مشاهده می‌کنیم که موجودی سکه‌های آنها با توجه به تراکنش‌های ثبت شده، تغییر کرده است.

در گره شماره یک، از ابتدا ۸ سکه وجود داشت که با ارسال ۴ سکه به گره شماره دو، ۴ سکه از خود باقی دارد و همچنین ۱۶ سکه پاداشی را نیز جهت استخراج بلاک دریافت کرده است.



شکل ۲۰-۴. اطلاعات کیف پول گره شماره یک بعد از ثبت تراکنش‌ها

همچنین گره شماره دو نیز که از ابتدا ۸ سکه از خود داشت، با دریافت ۴ سکه از گره شماره یک، تعداد سکه‌های خود را به ۱۲ تا افزایش می‌دهد.



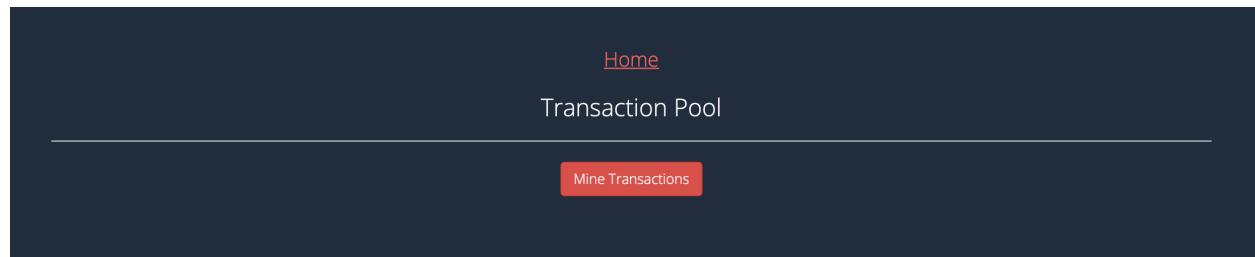
شکل ۲۱-۴. اطلاعات کیف پول گره شماره دو بعد از ثبت تراکنش‌ها

می‌توانیم سری به قسمت ثبت لاغ‌ها در پشت پرده بزنیم. مشاهده خواهیم کرد که گره یک پس از استخراج بلاک جدید، آن را برای شبکه ارسال کرده و گره شماره دو یک مسیج حاوی بلاکچین جدید دریافت کرده که بعد از صحت سنجی آن، بلاکچین جدید را با بلاکچین خود جابه جا می‌کند.

```
Message received. Channel: TRANSACTION. Message: {"id":"3f3a93d0-f86d-11ec-bfa9-8f7bcd6170d8","outputMap": {"042d275b1bce002e544caeababdf6c6d44ab3da93c77f3128ef05e7a31dae135b80640b44601ea444f63e4639310de9eef55690ff1dc4e7532a9dc1f3226af658e":4, "043f495163102017aefc45494c2593f21765de148bc7c4e652f077b0e66e8553159b7f6d59931b8998e609002b8c794060531fe8085b1d51bd97abe0f37eb1428d":4}, "input": {"timestamp":1656590863757, "amount":8, "address": "043f495163102017aefc45494c2593f21765de148bc7c4e652f077b0e66e8553159b7f6d59931b8998e609002b8c794060531fe8085b1d51bd97abe0f37eb1428d", "signature": [{"r": "9b5518d04ce69582aab51aebe42983c032ff3f44dcba8be963a9284f736f34d43", "s": "534276d418edf9bb5ba5976c9106c0540cb2ce5cd80ebab284c2d6c225fcadc", "recoveryParam":0}]}}  
Message received. Channel: BLOCKCHAIN. Message: [{"timestamp":1,"lastHash": "-----", "hash": "_hash_","data":[], "nonce":0, "difficulty":3}, {"timestamp":16565908631442, "lastHash": "_hash_","hash": "237443eff6e9fd0d25246e6dd2f5a11a12355cec29eb2af9e8b70aea45702925", "data": [{"id": "3f3a93d0-f86d-11ec-bfa9-8f7bcd6170d8", "outputMap": {"042d275b1bce002e544caeababdf6c6d44ab3da9310de9eef55690ff1dc4e7532a9dc1f3226af658e":4, "043f495163102017aefc45494c2593f21765de148bc7c4e652f077b0e66e8553159b7f6d59931b8998e609002b8c794060531fe8085b1d51bd97abe0f37eb1428d":4}, "input": {"timestamp":1656590863757, "amount":8, "address": "043f495163102017aefc45494c2593f21765de148bc7c4e652f077b0e66e8553159b7f6d59931b8998e609002b8c794060531fe8085b1d51bd97abe0f37eb1428d", "signature": [{"r": "9b5518d04ce69582aab51aebe42983c032ff3f44dcba8be963a9284f736f34d43", "s": "534276d418edf9bb5ba5976c9106c0540cb2ce5cd80ebab284c2d6c225fcadc", "recoveryParam":0}]}}, {"id": "67927d20-f86d-11ec-bfa9-8f7bcd6170d8", "outputMap": {"043f495163102017aefc45494c2593f21765de148bc7c4e652f077b0e66e8553159b7f6d59931b8998e609002b8c794060531fe8085b1d51bd97abe0f37eb1428d":16}, "input": {"address": "*authorized-reward*"}, {"nonce":7, "difficulty":2}]]  
Replacing chain with [ { timestamp: 1, lastHash: '-----', hash: '_hash_', data: [], nonce: 0, difficulty: 3 }, { timestamp: 16565908631442, lastHash: '_hash_', hash: '237443eff6e9fd0d25246e6dd2f5a11a12355cec29eb2af9e8b70aea45702925', data: [ [Object], [Object] ], nonce: 7, difficulty: 2 } ]
```

شکل ۲۲-۴. اطلاعات دریافت بلاکچین جدید توسط گره شماره دو توسط شبکه و جایگزینی آن

اگر بعد از این جایگزینی و ثبت بلاک جدید در بلاکچین، نگاهی به استخر تراکنش‌ها بزنیم، خواهیم دید که استخر حالی از تراکنش‌های ثبت شده در بلاکچین می‌باشد.



شکل ۲۳-۴. استخر تراکنش‌ها بعد از ثبت بلاک در شبکه

۱۰-۴. نتیجه گیری

حال که توانستیم یک بلاکچین به صورت غیر مرکز پیاده سازی کنیم و رمز ارزی را بربستر آن اجرا کنیم و نقل و انتقالات اولیه را در کنار استخراج بلاکها داشته باشیم، وقت ان رسیده است که نگاهی به یک سری تستها و ارزیابی‌ها در این مورد برسیم.

تست‌های چارچوب جست در فصل بعدی مورد بررسی قرار خواهند گرفت.
همچنین نگاهی به وبسایت و گیتهاب^۱ پروژه خواهیم کرد.

¹ Github

فصل ۵. ارزیابی نتایج

۱-۵. مقدمه

در این بخش ابتدا به بررسی خروجی تست‌ها در چارچوب جست می‌پردازیم. در چارچوب جست همانطور که از قبل گفته شد تمامی تست‌هایی که برای بخش عقب پروژه نوشته شده، بعد از اجرای کدهای اصلی مورد ارزیابی قرار می‌گیرند. این تست‌ها مدت زمان اجرای کدها، درستی منطق برنامه و همچنین روند برنامه به صورت کلی و با دید ماکرو مورد بررسی قرار می‌دهند.

۲-۵. ارزیابی‌ها از خروجی برنامه

با استفاده از دستور “`npm run test`” می‌توانیم به محیط چارچوب جست در پروژه خود دسترسی پیدا کنیم.

در ابتدا برنامه به صورت کلی تمامی تست‌ها را مورد بررسی قرار می‌دهد و نتایج کلی را از جمله میزان زمان مورد نیاز برای انجام تست‌ها و تعداد کلی تست‌ها به صورت جزئی در خروجی مشخص می‌کند.

```

PASS util/crypto-hash.test.js
PASS blockchain/Block.test.js
PASS blockchain/index.test.js
PASS wallet/index.test.js
PASS wallet/transaction.test.js (6.146s)
PASS wallet/transaction-pool.test.js (7.806s)

Test Suites: 6 passed, 6 total
Tests:       77 passed, 77 total
Snapshots:   0 total
Time:        8.113s
Ran all test suites.

Watch Usage
> Press f to run only failed tests.
> Press o to only run tests related to changed files.
> Press q to quit watch mode.
> Press t to filter by a test name regex pattern.
> Press p to filter by a filename regex pattern.
> Press Enter to trigger a test run.

```

شکل ۱-۵. خروجی اولیه تست‌های برنامه در ترمینال

تست‌های پروژه به طور کلی به ۶ قسمت تقسیم می‌شود.

۱- تست‌های بخش crypto-hash

۲- تست‌های بخش Block

۳- تست‌های بخش Blockchain

۴- تست‌های بخش wallet

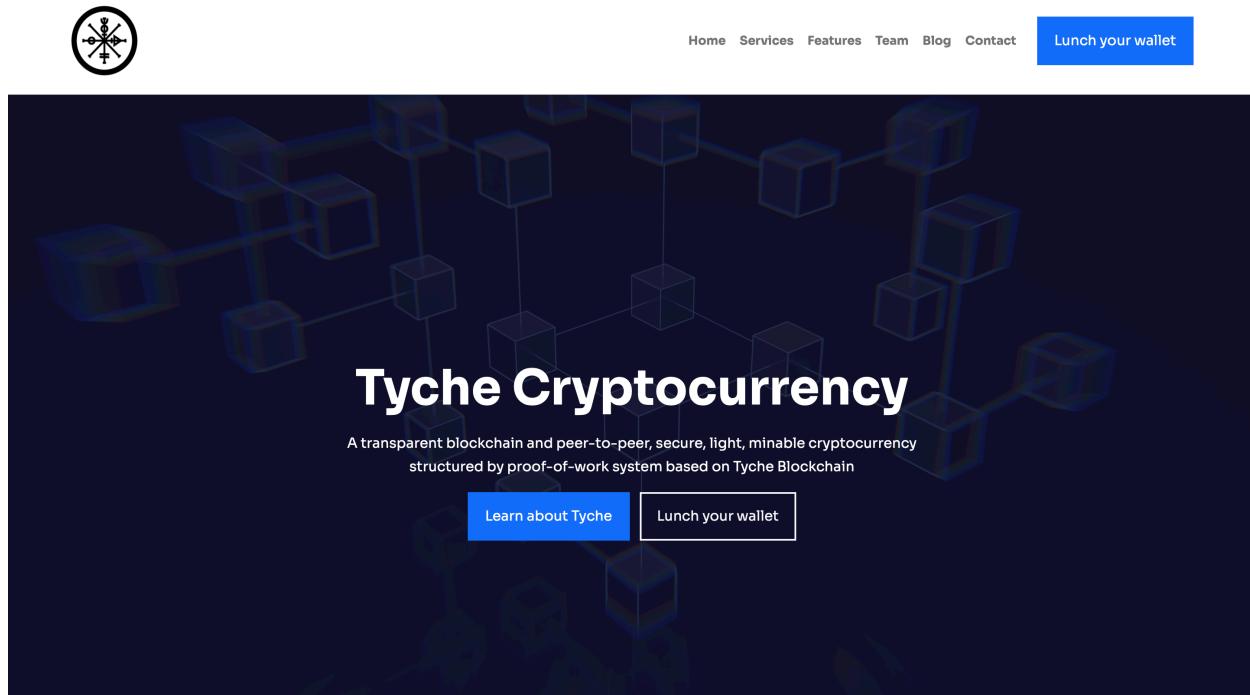
۵- تست‌های بخش transaction

۶- تست‌های بخش transaction-pool

مشاهده می‌کنیم که از این ۶ تست، ۴ تای اول در مدت زمان مناسب اجرا و پاس شده اند و ۲ تست آخر نیز در مدت زمان نسبتاً طولانی اجرا و پاس شده اند. این تاخیر زمانی به علت وجود سناریوهای و بخش‌های زیاد برای بخش کیف پول در قسمت‌های تست تراکنش‌ها و همچنین تست استخراج تراکنش‌ها می‌باشد.

اکنون نگاهی به سایت پروژه خواهیم زد.

این سایت با دامنه <http://tyche-project.org>^۱ قابل دسترس می‌باشد.



شکل ۲-۵. نمای اولیه و اصلی از وبسایت پروژه

سایت از یک صفحه اصلی تشکیل می‌شود که در آن توضیحات کلی پروژه، ویژگی‌ها و نقاط مثبت، معرفی دست اندرکاران، اطلاعات جرئی از نحوه ساختار پروژه و همچنین لینک‌های ارتباطی و یا لینک پروژه در دسترس می‌باشد.

^۱ وبسایت پروژه با دامنه خارجی (org) در زمانی که این پایان نامه ویرایش می‌شد، پیکر بندی شده است. چنانچه در سالهای آتی این پایان نامه را مطالعه می‌کنید و یا به هر دلیلی دسترسی به این سایت برای شما میسر نبود، به لینک گیتهاب پروژه مراجعه کنید.



Home Services Features Team Blog Contact

Lunch your wallet

About Tyche

As you walk through the Tyche architecture, you will discover lots of features and positive points such as following



Stability

Tyche is developed by multiple infrastructure tests in back-end, all APIs are stable with standard responses



Transparency

Since genesis block all blocks are visible in history, with sender and receiver public keys and the amount of transferred coins



Open Source

Tyche is on Github, open for soft or hard forks, also contributing to project will be our pleasure



Minable

Coins are minable from transaction-pool, by CPU by Proof-of-work strategy of mining and fetching hashes



Easy to launch

With just some clicks and commands, Tyche will lunch and can have your own node on your local machine



Light

Tyche was based on developing a coin with a light system and normal size. There will be no extra or cache data

شکل ۵-۳. نمایش مزیت‌های بلاکچین ساخته شده در وبسایت پروژه



Home Services Features Team Blog Contact

Lunch your wallet



WHY TO CHOOES US

Best solution for your situation

Not only Tyche could be a nice and convenient cryptocurrency, but also you can simply learn how a blockchain works and what a cryptocurrency does from it by reading documents and reviewing codes of the project simply on project's Github page



Learn about transparency

By understanding a blockchain structure, learning about transparency will be easier than anything



Contribute to our project

If you have any new idea, feel free to share it with us on Github on the issues page



Download source code now

Start your node now and make some transactions to your known wallets to see how it works

شکل ۴-۵. نمایش سرویس‌های اصلی در وبسایت پروژه



TEAM

Meet Our Crew Members

Tyche project is developed by Aria Radmehr under supervising vision of Dr. Sajad Haghzad Klidbary based on final project of bachelor degree in University of Zanjan



Aria Radmehr
Project Developer



Sajad Haghzad Klidbary
Head Supervisor



شکل ۵-۵. نمایش تیم اصلی توسعه در وبسایت پروژه



BLOG

Latest News From The Blog

Visit our blog for latest news about how we code, test and deploy our project to be used in the real time world

The basics of the project and back-end stuffs

It was all a nice journey to develop back-end with javascript and some modules

Test units and analyzing results and speed

Tyche tests units was coded by Jest test framework, by TDD development approach

Creating Tyche website with Bootstrap 5

Tyche website and documents was based on BS5 with simple and smooth templates

شکل ۵-۶. اطلاعات ساختار و ابزارهای بکار رفته در وبسایت پروژه

۵-۳. نتیجه گیری

با توجه به تست‌های انجام شده در این فصل و همچنین کار کردن بلاکچین به صورت همزمان با تعداد گره‌های زیاد و ساخت بخش وبسایت معرفی پروژه به صورت کامل، می‌توانیم بگوییم که از هر آنچه که باید از این پروژه انتظار داشته باشیم را برآورد کردہ‌ایم.

حال در فصل بعد به جمع بندی و دادن الگو و ایده‌هایی جهت کارهای آتی می‌پردازیم.

فصل ٦. مطالب آتى

۱-۶. مقدمه

در این پایان نامه سعی بر ساخت یک بلاکچین و یک رمز ارز بر بستر آن کردیم. پی بردیم که چگونه یک بلاکچین ایده پردازی می‌شود و قوانین حاکم بر آن به چه صورت است و چگونه پیاده سازی می‌شود. سپس به ساخت معماری و بخش عقب و جلوی پروژه کردیم و در نهایت ارزیابی‌هایی از کل پروژه داشتیم.

حال در این بخش به بررسی کارهای آتی و ایده‌های پیش رو پس از این پروژه می‌پردازیم.

۲-۶. ایده‌های پیش رو

در دنیای واقعی و کسب و کارها، برنامه‌های کاربردی که می‌توان آنها را بر بستر بلاکچین ایده پردازی کرد و گسترش داد بسیار زیاد است. از جمله این ایده‌ها می‌توان به گسترش سندهای دیجیتال و همچنین حفظ اسناد مهم و محترمانه به صورت بلاکچین‌های خصوصی و یا غیر محترمانه به صورت بلاکچین‌های عمومی استفاده کرد تا از تغییرنگردن ماهیت و هویت آنها و یا دستکاری نشدن عمدى یا سهوی بر روی آنها اطمینان حاصل کرد. دیگر ایده پیش رو، ساخت و گسترش رمز ارزهای جدید با ایده‌های نو و یا با ساختارهایی جدا از اثبات انجام کار (مثل اثبات داشتن سهم و یا اثبات داشتن ظرفیت) می‌باشد که بعضاً می‌تواند چاره مشکلات پیش رو در رمز ارزهای با محوریت اثبات انجام کار (مثل استفاده زیاد از برق و هدر دادن نیروی الکتریسیته به صورت گرما) باشد.

همچنین می‌توان به پیاده سازی توکن‌های غیر قابل تعویض^۱ بر بستر بلاکچین پرداخت. این ایده به صورت دادن مالکیت اشیای فیزیکی در دنیای دیجیتال به اشخاص حقیقی و یا حقوقیست به صورتی که حق استفاده از آن توکن را برای عموم میسر می‌دارد.

از بلاکچین‌ها نیز می‌توان در پیاده سازی سیستم‌های زیر استفاده کرد.

- ۱- مدیریت ارزهای دیجیتال
- ۲- سیستم ذخیره سوابق پزشکی
- ۳- ایجاد دفاتر ثبت اسناد رسمی
- ۴- ایجاد سیستم جمع آوری مالیات

و ...

۶-۳. نتیجه گیری

همانطور که دیدیم، برای هر چالشی در دنیای دیجیتال راه حلی وجود دارد که بلاکچین‌ها می‌توانند پاسخی برای بسیاری از آنها باشد. تنها کافیست مسئله و چالش پیش روی خود را بشناسیم و سیستم بلاکچین با محوریت حل آن مسئله را پیاده سازی کنیم.

لازم به ذکر است که همواره ایده پردازی‌های جدید در این سیستم و تکنولوژی که به خودی خود نو بنیان است، می‌تواند بسیار جذاب و یا انقلابی باشد.

امید است که با پیش روی و خوش بینی، راهی پر از موفقیت برای هر شخص در این عرصه هموار شود تا روز به روز شاهد پیشرفت دانش و خرد جمیعی در این دنیای پهناور باشیم.

¹ Non Fungible Tokens

فصل ٧. منابع و مأخذ

- [1] Narayanan, Arvind, BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES (A Comprehensive Introduction), vol.1, Princeton, Princeton University Press, pp. 3 , 2016
- [2] Narayanan, Arvind, BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES (A Comprehensive Introduction), vol.1, Princeton, Princeton University Press, pp. 4-6 , 2016
- [3] Narayanan, Arvind, BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES (A Comprehensive Introduction), vol.1, Princeton, Princeton University Press, pp. 7-11 , 2016
- [4] [Online]. Available: <https://www.youtube.com/c/JadiMirmirani>
- [5] Narayanan, Arvind, BITCOIN AND CRYPTOCURRENCY TECHNOLOGIES (A Comprehensive Introduction), vol.1, Princeton, Princeton University Press, 2016
- [6] [Online]. Available: <https://www.bitcoin.org/en/>
- [7] [Online]. Available: <https://www.ethereum.org/en/>
- [8] [Online]. Available: <https://www.tether.to/en/>
- [9] ISO/TC, Blockchain and electronic distributed ledger technologies, 2016.
- [10] IBM.[Online]. Available: <https://www.ibm.com/blockchain/what-is-blockchain>.
- [11] E. Homestead, "Ethereum Homestead Documentation," 2016. [Online]. Available: <https://ethdocs.org/en/latest/introduction/what-is-ethereum.html>.
- [12] D. Guegan, "Public blockchain versus private blockchain," 2017.
- [13] [Online]. Available: <https://www.javascript.com>
- [14] [Online]. Available: <https://www.html.com>
- [15] [Online]. Available: <https://www.w3.org/Style/CSS/Overview.en.html>
- [16] [Online]. Available: <https://www.reactjs.org>
- [17] [Online]. Available: <https://www.jestjs.io>
- [18] [Online]. Available: <https://www.postman.com>

Abstract

In the upcoming world, blockchains are considered a new technology in the field of computer science. This technology was first discovered in 1991 by some researchers and after Satoshi Nakamoto created Bitcoin in 2009, it attracted the attention of the world. In simple language, blockchain is like a collection of information that is stored on a distributed system platform.

Knowledge of blockchains in different types and cases, either individually or as an aid or a tool in systems and businesses, is very important. Its platform (digital currency in this thesis) can give any person the necessary knowledge of the functioning of a blockchain system, as well as the necessary understanding of evaluating the features and characteristics of blockchains.

In this thesis, we will first examine the main and constructive structure of blockchains, and then by explaining specialized and advanced topics in the architecture of crypto-currencies on the blockchain platform, we will try to implement everything we said using a number of simple tools.

Keywords: blockchain, block, cryptocurrency, digital signature, public and private key, hash, node, decentralization, distributed, consensus, proof of work, mining.



University of Zanjan

Faculty of Engineering

Dissertation to receive a bachelor's degree

Computer Engineering

Tyche Blockchain and Cryptocurrency

By:

Aria Radmehr

Supervisor:

Dr. Sajjad Haghzad Kelidbary

Summer 2022