دانشگاه صنعتی امیرکبیر

(پلی‌تکنیک تهران)

دانشکده مهندسی کامپیوتر

پایان‌نامه کارشناسی

طراحی و پیاده سازی سامانه زنجیره تأمین مبتنی بر زنجیره بلوکی با بررسی صحت فرادادهها

نگارش

سید سپهرمیرنصرالهی پارسا

استاد راهنما

دکتر حمیدرضا زرندی

شهریور ۱۴۰۴

دانشگاه صنعتی امیرکبیر
(پلی‌تکنیک تهران)

به نام خدا

تعهدنامه اصالت اثر

تاریخ: شهریور ۱۴۰۴

اینجانب **سید سپهرمیرنصرالهی پارسا** متعهد می‌شوم که مطالب مندرج در این پایان‌نامه حاصل کار پژوهشی اینجانب تحت نظارت و راهنمایی اساتید دانشگاه صنعتی امیرکبیر بوده و به دستاوردهای دیگران که در این پژوهش از آنها استفاده شده است مطابق مقررات و روال متعارف ارجاع و در فهرست منابع و مآخذ ذکر گردیده است. این پایان‌نامه قبلاً برای احراز هیچ مدرک هم‌سطح یا بالاتر ارائه نگردیده است.

در صورت اثبات تخلف در هر زمان، مدرک تحصیلی صادر شده توسط دانشگاه از درجه اعتبار ساقط بوده و دانشگاه حق پیگیری قانونی خواهد داشت.

کلیه نتایج و حقوق حاصل از این پایان‌نامه متعلق به دانشگاه صنعتی امیرکبیر می‌باشد. هرگونه استفاده از نتایج علمی و عملی، واگذاری اطلاعات به دیگران یا چاپ و تکثیر، نسخه‌برداری، ترجمه و اقتباس از این پایان نامه بدون موافقت کتبی دانشگاه صنعتی امیرکبیر ممنوع است. نقل مطالب با ذکر مآخذ بلامانع است.

سید سپهرمیرنصرالهی پارسا

امضا

سپاس گزاری

با سپاس فراوان ازجناب آقای دکتر زرندی به عنوان‌استاد مشاور این پایان نامه‌و جناب آقای دکترجوادی به عنوان‌استاد داور، که با راهنمایی هاو نظرات ارزشمند خود نقش بسزایی در پیشبرداین پژوهش داشتند.

سید سپهر میرنصرالهی پارسا
شهریور ۱۴۰۴

چکیده

امروزه، زنجیره‌های تأمین با چالش‌های متعددی از جمله عدم شفافیت، جعل کالا و ردیابی دشوار محصولات روبرو هستند. این مسائل، به ویژه در صنایع حساس، اعتماد مصرف‌کنندگان را کاهش داده و خسارات اقتصادی قابل توجهی را به تولیدکنندگان معتبر وارد می‌سازد. این پروژه، با هدف غلبه بر این چالش‌ها، به طراحی و پیاده‌سازی یک سامانه زنجیره تأمین غیرمتمرکز مبتنی بر فناوری زنجیره بلوکی می‌پردازد. سامانه پیشنهادی با ایجاد یک دفتر کل توزیع‌شده، شفاف، و تغییرناپذیر، ردیابی کامل چرخه حیات محصول از تولید تا مصرف‌کننده نهایی را ممکن می‌سازد.

هسته اصلی این سامانه، یک قرارداد هوشمند است که با زبان برنامه‌نویسی $Solidity$ بر روی یک شبکه سازگار با ماشین مجازی اتریوم (EVM) پیاده‌سازی شده است. در این قرارداد از استاندارد توکن $ERC1155$ استفاده شده که امکان مدیریت بهینه و همزمان محصولات مثلی و غیرمثلی را با هزینه تراکنش کمتر فراهم می‌کند. یکی از ویژگی‌های کلیدی این پروژه، تضمین صحت فراداده‌ها از طریق تولید هش $Keccak256$ برای هر محصول است. این مکانیزم به تمام ذی‌نفعان زنجیره اجازه می‌دهد تا اصالت و اطلاعات محصول را در هر مرحله اعتبارسنجی کنند. این سیستم شامل نقش‌های دسترسی متفاوتی مانند تولیدکننده، توزیع‌کننده، خرده‌فروش و گمرک است که هر یک مجوزهای خاص خود را برای ثبت، انتقال یا ابطال محصول دارند. علاوه بر این، یک قابلیت نوآورانه برای محاسبه خودکار مالیات در هر مرحله از انتقال مالکیت در قرارداد هوشمند تعبیه شده است.

برای تضمین امنیت و کارایی قرارداد هوشمند، مجموعه‌ای از آزمون‌های جامع واحد و یکپارچه‌سازی با استفاده از فریم‌ورک $Foundry$ توسعه داده شده است. رابط کاربری سامانه به صورت یک برنامه وب مدرن با بهره‌گیری از کتابخانه $React$ و ابزار ساخت $Vite$ طراحی گردیده و اتصال به کیف پول‌های دیجیتال کاربران از طریق هوک‌های $Wagmi$ مدیریت می‌شود. این امر به مصرف‌کنندگان اجازه می‌دهد تا با اسکن یک کد QR، به سادگی تاریخچه کامل و تأییدیه اصالت محصول را مشاهده نمایند.

در نهایت، این پروژه یک راهکار عملی و جامع ارائه می‌دهد که با افزایش شفافیت، قابلیت ردیابی و تضمین اصالت کالا، می‌تواند به طور مؤثری با تقلب مبارزه کرده و اعتماد را به اکوسیستم زنجیره تأمین بازگرداند.

واژه‌های کلیدی:

زنجیره بلوکی، زنجیره تأمین، قرارداد هوشمند، $ERC1155$، $Solidity$، اصالت کالا، $Foundry$، $React$.

فهرست مطالب

عنوان صفحه

فهرست جداول

جدول صفحه

فصل اول

مقدمه

۱-۱ بیان مسئله و اهمیت موضوع

زنجیره تأمین، شبکه‌ای پیچیده و حیاتی از سازمان‌ها، افراد، فعالیت‌ها، اطلاعات و منابع است که در حرکت یک محصول یا خدمت از تأمین‌کننده به مصرف‌کننده نهایی نقش دارد. این زنجیره نه تنها جریان فیزیکی کالاها، بلکه جریان اطلاعات و مالی را نیز در بر می‌گیرد. کارایی و سلامت زنجیره تأمین به عنوان یکی از ارکان اساسی اقتصاد مدرن، نقشی مستقیم در رشد اقتصادی، ثبات بازار و رفاه اجتماعی یک کشور ایفا می‌کند. یک زنجیره تأمین کارآمد، هزینه‌ها را کاهش می‌دهد، دسترسی مصرف‌کنندگان به کالاها را تسهیل می‌کند و مزیت رقابتی برای تولیدکنندگان داخلی در بازارهای جهانی ایجاد می‌نماید.

با وجود این اهمیت استراتژیک، صنعت زنجیره تأمین در ایران و بسیاری از نقاط جهان با چالش‌ها و مشکلات ساختاری عمیقی مواجه است که کارایی و اعتبار آن را به شدت زیر سؤال برده است. این مشکلات صرفاً به ناکارآمدی‌های لجستیکی محدود نمی شود، بلکه یک بحران جدی در شفافیت، اعتماد و امنیت را شامل می‌گردد که تمام بازیگران این اکوسیستم، از تولیدکننده تا مصرف‌کننده، را تحت تأثیر قرار می‌دهد.

۱-۱-۱ بحران اعتماد و شفافیت در زنجیره‌های تأمین سنتی

یکی از بزرگ‌ترین چالش‌های موجود، در فرآیندهای زنجیره تأمین است. این عدم شفافیت، بستری مناسب برای بروز مشکلات متعددی فراهم آورده است که در ادامه به تفصیل بررسی می‌شوند:

گسترش پدیده جعل و تقلب در محصولات

جعل محصولات یکی از مخرب‌ترین پیامدهای یک زنجیره تأمین غیرشفاف است. این معضل دیگر به کالاهای لوکس محدود نیست و دامنه آن به حوزه‌های حیاتی مانند صنایع تولیدی و غذایی نیز کشیده شده است. کالاهای تقلبی نه تنها با ارائه کیفیت نازل به اعتبار برندهای معتبر آسیب می‌زنند و موجب خسارات اقتصادی هنگفت می‌شوند، بلکه در موارد حساس مانند دارو و قطعات صنعتی، می‌توانند سلامت و ایمنی مصرف‌کنندگان را به طور جدی به خطر اندازند. در یک سیستم سنتی، هنگامی که یک محصول از کارخانه خارج می‌شود، ردیابی دقیق آن در هر مرحله از توزیع، انبارداری و فروش تقریبا غیرممکن است. این گسست اطلاعاتی، به عوامل سودجو اجازه می‌دهد تا کالاهای تقلبی را به راحتی وارد چرخه توزیع کرده و به دست مصرف‌کننده برسانند.

نوسانات کیفیت و عدم امکان ریشه‌یابی

فقدان یک سیستم ردیابی یکپارچه، کنترل و تضمین کیفیت محصول در طول زنجیره را به امری دشوار تبدیل کرده است. یک محصول ممکن است در مرحله تولید از کیفیت بالایی برخوردار باشد، اما به دلیل شرایط نگهداری نامناسب در انبار، حمل‌ونقل غیراصولی یا تأخیر در توزیع، کیفیت خود را از دست بدهد. در سیستم‌های سنتی، زمانی که یک مصرف‌کننده با محصولی بی‌کیفیت مواجه می‌شود، ریشه‌یابی دقیق

اینکه کدام حلقه از زنجیره مسئول این افت کیفیت بوده، بسیار پیچیده و گاهی ناممکن است. این امر، پاسخگو نگه داشتن عاملان را دشوار کرده و از بهبود مستمر فرآیندها جلوگیری می‌کند.

عدم شفافیت در مسیر واردات و توزیع

در زنجیره‌های تأمین بین‌المللی، کالاها از مراحل متعددی مانند گمرک، شرکت‌های حمل‌ونقل مختلف و انبارهای متعدد عبور می‌کنند. هر یک از این مراحل می‌تواند نقطه بالقوه‌ای برای بروز فساد، تأخیرهای بی‌دلیل و ورود کالاهای قاچاق باشد. کمبود شفافیت در این مسیر، نظارت دقیق بر اصالت و سلامت کالا را برای نهادهای نظارتی و همچنین واردکنندگان دشوار می‌سازد و به اقتصاد غیررسمی دامن می‌زند.

۱-۱-۲ آسیب‌پذیری‌های معماری در سیستم‌های مدیریت متمرکز

ریشه بسیاری از مشکلات ذکر شده، در معماری فنی سیستم‌های مدیریتی نهفته است که در حال حاضر بر زنجیره‌های تأمین حاکم هستند. این سیستم‌ها غالباً بر پایه پایگاه‌های داده **متمرکز** طراحی شده‌اند که هر سازمان یا شرکت، داده‌های خود را در سیلوهای اطلاعاتی مجزا نگهداری می‌کند. این معماری دارای نقاط ضعف بنیادینی است:

- **آسیب‌پذیری در برابر دستکاری:** در یک سیستم متمرکز، یک نهاد واحد (صاحب سرور) کنترل کاملی بر روی اطلاعات دارد. این موضوع، داده‌ها را هم در برابر حملات سایبری خارجی و هم در برابر دستکاری‌های داخلی توسط افراد دارای مجوز، به شدت آسیب‌پذیر می‌کند. یک تغییر کوچک و غیرقابل ردیابی در داده‌های مربوط به تاریخ تولید یا مبدأ کالا، می‌تواند کل زنجیره را با اطلاعات نادرست تغذیه کند.
- **عدم وجود یک منبع حقیقت واحد** ($SingleSourceofTruth$)**:** هر یک از شرکت‌کنندگان در زنجیره تأمین (تولیدکننده، شرکت حمل‌ونقل، توزیع‌کننده، خرده‌فروش) پایگاه داده و سیستم مدیریتی خود را دارد. این جزیره‌ای بودن اطلاعات باعث می‌شود که هماهنگ‌سازی داده‌ها بین این سیستم‌ها به صورت دستی، با تأخیر و با احتمال بالای خطا انجام شود. این نبود یکپارچگی، منجر به ناکارآمدی‌های عملیاتی و عدم امکان مشاهده یک تصویر کامل و دقیق از وضعیت لحظه‌ای یک محصول می‌شود.
- **پیچیدگی و هزینه بالا:** نگهداری و تأمین امنیت زیرساخت‌های متمرکز، به خصوص برای شرکت‌های کوچک و متوسط، هزینه‌بر و پیچیده است. این در حالی است که تعامل و یکپارچه‌سازی این سیستم‌های ناهمگون نیز خود به پروژه‌های نرم‌افزاری گران‌قیمت و زمان‌بر نیاز دارد.

۱-۱-۳ پیامدهای اقتصادی و اجتماعی

مجموعه این چالش‌ها، پیامدهای گسترده‌ای برای اقتصاد و جامعه به همراه دارد. مهم‌ترین پیامد، است. زمانی که مصرف‌کنندگان نتوانند به اصالت و کیفیت کالایی که خریداری می‌کنند اطمینان داشته باشند، تمایل آن‌ها برای خرید محصولات داخلی و حمایت از برندهای معتبر کاهش می‌یابد. این امر مستقیماً به تولید ملی و اعتبار برندها لطمه می‌زند.

از منظر اقتصادی، ناکارآمدی‌های موجود در زنجیره تأمین منجر به افزایش هزینه‌های عملیاتی، اتلاف منابع و کاهش قدرت رقابت‌پذیری کسب‌وکارها در سطح ملی و بین‌المللی می‌شود. در نهایت، این مسائل نشان می‌دهند که مشکلات موجود در زنجیره تأمین، سطحی و قابل حل با راهکارهای مقطعی نیستند، بلکه ریشه در یک ضعف ساختاری عمیق در معماری اعتماد و جریان اطلاعات دارند. بنابراین، برای عبور از این بحران، نیاز به یک تغییر پارادایم اساسی و بهره‌گیری از فناوری‌های نوینی است که بتوانند شفافیت، امنیت و تغییرناپذیری را به اکوسیستم بازگردانند. اهمیت این موضوع، ضرورت تحقیق و توسعه راهکارهای جایگزین، مانند آنچه در این پروژه ارائه خواهد شد را دوچندان می‌کند.

۱-۲ فناوری زنجیره بلوکی به عنوان راهکار

در پاسخ به چالش‌های عمیق و ساختاری حاکم بر زنجیره‌های تأمین سنتی، که در بخش پیشین به تفصیل بررسی شد، نیاز به یک تغییر پارادایم اساسی احساس می‌شود. راهکارهای مقطعی و بهبودهای جزئی در سیستم‌های متمرکز، قادر به حل ریشه‌ای بحران اعتماد و شفافیت نیستند. در این میان، فناوری زنجیره بلوکی (□□□□□□□□□□) به عنوان یک رویکرد نوین و بنیادین، ظرفیت‌های بی‌نظیری برای بازمهندسی فرآیندهای زنجیره تأمین ارائه می‌دهد. این فناوری صرفاً یک ابزار جدید نیست، بلکه یک معماری کاملاً متفاوت برای ثبت، اشتراک‌گذاری و مدیریت اطلاعات است که می‌تواند شفافیت، امنیت و کارایی را به طور همزمان به اکوسیستم تزریق کند. در ادامه، به بررسی ابعاد مختلف این فناوری، از تاریخچه و مبانی فنی آن گرفته تا کاربرد مستقیم آن در قالب قراردادهای هوشمند، می‌پردازیم تا درک عمیق‌تری از چرایی انتخاب آن به عنوان راهکار اصلی این پروژه حاصل شود.

۱-۲-۱ نگاهی تاریخی به فناوری زنجیره بلوکی

برای درک اهمیت و جایگاه امروزی زنجیره بلوکی، باید به سیر تکاملی اینترنت و نیازهایی که در هر دوره به وجود آمد، نگاهی بیندازیم. این تاریخچه به ما نشان می‌دهد که زنجیره بلوکی، پاسخی طبیعی به محدودیت‌های نسل‌های پیشین وب بوده است.

از وب ۰.۱ تا بحران تمرکزگرایی در وب ۰.۲

دوران اولیه اینترنت، معروف به وب ۰.۱ (تقریبا از ۱۹۹۱ تا ۲۰۰۴)، به «وب فقط خواندنی» شهرت داشت. در این دوره، محتوا عمدتاً ایستا بود و توسط تعداد محدودی از سازمان‌ها و افراد تولید و بر روی وب‌سایت‌ها منتشر می‌شد. کاربران عمدتاً مصرف‌کنندگان غیرفعال اطلاعات بودند و تعامل چندانی وجود نداشت.

با ظهور وب ۰.۲، پارادایم به کلی تغییر کرد و «وب تعاملی و اجتماعی» متولد شد[۱]. پلتفرم‌هایی مانند فیسبوک، اینستاگرام و یوتیوب به کاربران عادی این قدرت را دادند که به سادگی و بدون نیاز به دانش فنی، خود به تولیدکنندگان محتوا تبدیل شوند. این تحول، منجر به انفجار تولید محتوا و ایجاد شبکه‌های اجتماعی گسترده شد. با این حال، این آزادی و سهولت، هزینه‌ای پنهان به همراه داشت: **تمرکزگرایی شدید قدرت و داده**. معماری وب ۰.۲ بر پایه سرورهای متمرکز شرکت‌های بزرگ بنا شده است. این شرکت‌ها با ارائه خدمات رایگان، کاربران را جذب کرده و در ازای آن، به بزرگ‌ترین دارایی آن‌ها، یعنی داده‌های شخصی‌شان، دسترسی پیدا کردند. مدل کسب‌وکار این غول‌های فناوری، عمدتاً بر دو پایه استوار شد: تبلیغات هدفمند یا فروش مستقیم اطلاعات کاربران به اشخاص ثالث[۲]. این ساختار متمرکز، مشکلات بنیادینی را به وجود آورد:

- **مالکیت داده:** کاربران، مالک واقعی داده‌های خود نبودند و کنترلی بر نحوه استفاده از آن نداشتند.
- **سانسور و کنترل:** شرکت‌های متمرکز می‌توانستند به صورت سلیقه‌ای محتوا را حذف کرده یا دسترسی کاربران را مسدود کنند.
- **نقطه شکست واحد** ($Single Point of Failure$): تمرکز داده‌ها بر روی سرورهای یک شرکت، آن‌ها را به هدفی جذاب برای حملات سایبری تبدیل کرد و از کار افتادن این سرورها به معنای قطع شدن سرویس برای میلیون‌ها کاربر بود.

ظهور بیت‌کوین و مفهوم عدم تمرکز

در چنین فضایی، نیاز به سیستمی که بتواند اعتماد و تعامل را بدون نیاز به یک واسطه متمرکز فراهم کند، به شدت احساس می‌شد. در سال ۲۰۰۸، فرد یا گروهی ناشناس با نام مستعار ساتوشی ناکاموتو، با انتشار وایت‌پیپر بیت‌کوین، راهکاری انقلابی ارائه داد. بیت‌کوین یک سیستم پول نقد الکترونیکی همتا به همتا بود که به کاربران اجازه می‌داد بدون نیاز به بانک یا هر مؤسسه مالی دیگری، به یکدیگر پول انتقال دهند. هسته اصلی این نوآوری، فناوری **زنجیره بلوکی** بود؛ یک پایگاه داده خاص که داده‌ها را در بلوک‌هایی ذخیره می‌کند که به صورت رمزنگاری شده به یکدیگر متصل هستند[۳]. این ساختار زنجیره‌ای، داده‌ها را به ترتیب زمانی مرتب کرده و مهم‌تر از آن، **تغییرناپذیر** می‌ساخت[۳]. هر تراکنش ثبت‌شده در زنجیره بلوکی بیت‌کوین، برای همیشه در آن باقی می‌ماند و برای همه قابل مشاهده بود، که این شفافیت، امنیت بالایی را در برابر تقلب و کلاهبرداری ایجاد می‌کرد[۳].

عصر اتریوم و تولد وب ۰.۳

بیت‌کوین ثابت کرد که می‌توان اعتماد را به صورت غیرمتمرکز ایجاد کرد، اما کاربرد آن عمدتاً به تراکنش‌های مالی محدود بود. جهش بزرگ بعدی با ظهور **اتریوم** رخ داد. اتریوم با گسترش ایده زنجیره بلوکی، این امکان را فراهم آورد که نه تنها اعداد (مانند مبالغ تراکنش)، بلکه **کد اجرایی** نیز بر روی زنجیره بلوکی ذخیره و اجرا شود[۴]. این نوآوری، منجر به پیدایش **قراردادهای هوشمند** و **برنامه‌های غیرمتمرکز** ($DApps$) شد[۴، ۵].

این تحول، زمینه را برای شکل‌گیری وب ۰.۳ فراهم کرد. وب ۰.۳ که به آن «وب غیرمتمرکز» نیز گفته می‌شود، چشم‌اندازی از اینترنت است که در آن کاربران کنترل داده‌ها و هویت دیجیتال خود را پس می‌گیرند. ویژگی‌های اصلی وب ۰.۳ که مستقیماً از فناوری زنجیره بلوکی نشأت می‌گیرند، عبارتند از:

- **غیرمتمرکز بودن** ($Decentralized$): کنترل در دست کاربران و جامعه است، نه شرکت‌های بزرگ.
- **بی‌نیاز به اعتماد** ($Trustless$): تعاملات بر اساس قوانین شفاف و تغییرناپذیر کد انجام می‌شود، نه اعتماد به یک واسطه.
- **بی‌نیاز به مجوز** ($Permissionless$): هر کسی می‌تواند بدون نیاز به کسب اجازه از یک نهاد مرکزی، در شبکه مشارکت کرده و سرویس ایجاد کند.
- **دارای پرداخت‌های درون‌ساختی** ($NativePayments$): تراکنش‌های مالی جزئی جدایی‌ناپذیر از پروتکل است و نیازی به سیستم‌های پرداخت خارجی نیست.

این سیر تکاملی نشان می‌دهد که زنجیره بلوکی، صرفاً یک فناوری برای رمزارزها نیست، بلکه زیرساختی برای نسل بعدی اینترنت و برنامه‌های کاربردی است که می‌توانند صنایع مختلف، از جمله زنجیره تأمین را متحول سازند.

۱-۲-۲ مبانی رمزنگاری در زنجیره بلوکی

امنیت، یکپارچگی و تغییرناپذیری زنجیره بلوکی، بر ستون‌های مستحکم علم رمزنگاری ($Cryptography$) استوار است. بدون رمزنگاری، اعتماد به یک سیستم غیرمتمرکز که توسط افراد ناشناس اداره می‌شود، غیرممکن بود. دو مفهوم کلیدی رمزنگاری که در قلب زنجیره بلوکی قرار دارند، توابع درهم‌سازی و رمزنگاری کلید عمومی هستند.

توابع درهم‌سازی و یکپارچگی داده‌ها ($HashFunctions$)

تابع درهم‌سازی یا هش، یک الگوریتم ریاضی است که هر ورودی با هر اندازه‌ای را دریافت کرده و یک خروجی با اندازه ثابت تولید می‌کند. این خروجی که به آن «هش» یا «چکیده» گفته می‌شود، مانند

اثر انگشت دیجیتال برای داده ورودی عمل می‌کند. توابع هش مورد استفاده در زنجیره بلوکی، مانند $Keccak256$ که در این پروژه نیز به کار گرفته شده است[؟، ؟]، دارای سه ویژگی اساسی هستند:

۱. **قطعیت** ($Deterministic$): یک ورودی مشخص، همواره هش یکسانی تولید می‌کند.

۲. **مقاومت در برابر پیش‌تصویر** ($Pre-imageResistance$): محاسبه ورودی از روی هش خروجی، از نظر محاسباتی غیرممکن است.

۳. **اثر بهمنی** ($AvalancheEffect$): کوچک‌ترین تغییری در داده ورودی، منجر به تولید یک هش خروجی کاملاً متفاوت می‌شود.

این ویژگی‌ها کاربردهای حیاتی در زنجیره بلوکی دارند. اولاً، برای اطمینان از **یکپارچگی داده‌ها** به کار می‌روند. در پروژه حاضر، با محاسبه هش اطلاعات هر محصول و ثبت آن بر روی زنجیره، تضمین می‌شود که این اطلاعات پس از ثبت، به هیچ عنوان دستکاری نشده‌اند. هرگونه تلاشی برای تغییر جزئیات محصول، منجر به تولید یک هش متفاوت شده و به راحتی قابل تشخیص خواهد بود. ثانیاً، برای **ایجاد زنجیره** به کار می‌روند. هر بلوک در زنجیره، علاوه بر داده‌های خود، هش بلوک قبلی را نیز در خود ذخیره می‌کند. این وابستگی زنجیره‌ای باعث می‌شود که تغییر اطلاعات یک بلوک، نیازمند محاسبه مجدد هش تمام بلوک‌های بعدی باشد که این امر از نظر محاسباتی، دستکاری تاریخچه را غیرممکن می‌سازد و به کل سیستم، خاصیت تغییرناپذیری می‌بخشد.

رمزنگاری کلید عمومی و هویت دیجیتال ($PublicKeyCryptography$)

یکی دیگر از ارکان رمزنگاری در زنجیره بلوکی، سیستم رمزنگاری نامتقارن یا کلید عمومی است. در این سیستم، هر کاربر دارای یک جفت کلید است: یک **کلید خصوصی** و یک **کلید عمومی**.

- **کلید خصوصی** ($PrivateKey$): این کلید باید به صورت کاملاً محرمانه توسط کاربر نگهداری شود. کاربرد اصلی آن، امضای دیجیتال تراکنش‌هاست. وقتی کاربر یک تراکنش (مانند انتقال مالکیت یک کالا) را با کلید خصوصی خود امضا می‌کند، در واقع در حال اثبات مالکیت خود بر آن دارایی و تأیید صحت آن تراکنش است.
- **کلید عمومی** ($PublicKey$): این کلید از روی کلید خصوصی تولید می‌شود و می‌توان آن را به صورت عمومی با دیگران به اشتراک گذاشت. از کلید عمومی، آدرس کاربر در شبکه استخراج می‌شود که برای دریافت دارایی‌ها به کار می‌رود. دیگران می‌توانند با استفاده از کلید عمومی یک کاربر، امضای دیجیتال او را اعتبارسنجی کرده و مطمئن شوند که تراکنش واقعاً توسط مالک کلید خصوصی مربوطه ارسال شده است.

این سازوکار، یک سیستم هویت و احراز هویت دیجیتال قدرتمند و غیرمتمرکز ایجاد می‌کند. کاربران برای تعامل با شبکه، نیازی به ثبت‌نام در یک مرجع مرکزی و ارائه اطلاعات هویتی خود ندارند[؟].

کلیدهای آن‌ها، هویت دیجیتالشان است. این ویژگی، ضمن حفظ حریم خصوصی، امکان تعامل امن و قابل اعتماد بین طرفین ناشناس را فراهم می‌آورد که برای یک زنجیره تأمین جهانی امری ضروری است.

۱-۲-۳ قراردادهای هوشمند: انقلابی در توافقات دیجیتال

اگر زنجیره بلوکی را یک سیستم عامل غیرمتمرکز در نظر بگیریم، قراردادهای هوشمند ($SmartContracts$) برنامه‌هایی هستند که بر روی این سیستم عامل اجرا می‌شوند. این مفهوم که با ظهور اتریوم به بلوغ رسید، زنجیره بلوکی را از یک سیستم صرفاً تراکنشی به یک پلتفرم محاسباتی جهانی تبدیل کرد.

تعریف و ماهیت قرارداد هوشمند

یک قرارداد هوشمند، یک برنامه کامپیوتری یا پروتکل تراکنش است که به صورت خودکار، اقدامات و توافقات مشخصی را اجرا، کنترل یا مستند می‌کند. به زبان ساده‌تر، یک قرارداد هوشمند، «کدی است که عملیات خاصی را اجرا می‌نماید و می‌تواند با سایر قراردادهای هوشمند تعامل داشته باشد»[؟]. این کد، پس از نوشته شدن، بر روی زنجیره بلوکی مستقر ($Deploy$) می‌شود و از آن پس، به صورت مستقل و خودکار بر اساس منطق برنامه‌ریزی شده خود عمل می‌کند.

جایگزینی واسطه‌های شخص ثالث

قدرت واقعی قراردادهای هوشمند در توانایی آن‌ها برای **حذف واسطه‌های شخص ثالث** نهفته است[؟]. در دنیای سنتی، اجرای توافقات نیازمند اعتماد به واسطه‌هایی مانند بانک‌ها، دفاتر اسناد رسمی، وکلا یا پلتفرم‌های آنلاین است. این واسطه‌ها وظیفه تضمین اجرای صحیح قرارداد و حل اختلافات را بر عهده دارند و در ازای آن، کارمزد دریافت می‌کنند و فرآیند را کند و پیچیده می‌سازند. قراردادهای هوشمند این نقش را به کد منتقل می‌کنند. «همان گونه که بیت‌کوین نیاز به نگهداری پول شما توسط بانک را از بین می‌برد، اتریوم نیز با استفاده از قراردادهای هوشمند، نیازی به شخصی برای نظارت بر تراکنش و یا معامله ندارد»[؟]. قوانین توافق (مانند شرایط انتقال مالکیت یک کالا در زنجیره تأمین) یک بار در کد قرارداد نوشته می‌شود و از آن پس، شبکه غیرمتمرکز زنجیره بلوکی، اجرای بی‌طرفانه و دقیق آن قوانین را تضمین می‌کند.

ویژگی‌های کلیدی قراردادهای هوشمند

ویژگی‌های قراردادهای هوشمند مستقیماً از ماهیت زنجیره بلوکی که بر روی آن اجرا می‌شوند، به ارث برده شده است:

- **تغییرناپذیری و قطعیت** ($ImmutabilityDeterminism$): پس از استقرار یک قرارداد هوشمند بر روی زنجیره بلوکی، کد آن دیگر به هیچ عنوان قابل تغییر نیست[؟]. این ویژگی تضمین می‌کند

که قوانین بازی در حین اجرا تغییر نخواهد کرد و همه شرکت‌کنندگان می‌توانند با اطمینان کامل به آن تکیه کنند.

- **شفافیت و قابلیت حسابرسی** ($Transparency\&Auditability$): کد قرارداد هوشمند و تمام تراکنش‌هایی که با آن انجام می‌شود، به صورت عمومی بر روی زنجیره بلوکی ثبت شده و برای همگان قابل مشاهده است[؟، ؟]. این شفافیت، امکان حسابرسی کامل فرآیندها را فراهم کرده و از اقدامات پنهانی جلوگیری می‌کند.
- **عدم تمرکز و پایداری** ($Decentralization\&Robustness$): قرارداد هوشمند بر روی یک سرور مرکزی اجرا نمی‌شود، بلکه بر روی هزاران گره (□□□□) در سراسر شبکه توزیع شده است. این ساختار غیرمتمرکز باعث می‌شود که قرارداد در برابر سانسور و حملات مقاوم باشد. «حذف یک گره، اجرای هیچ یک از قراردادهای هوشمند را مختل نمی‌کند»[؟] و سیستم دارای پایداری و در دسترس بودن بسیار بالایی است.

۱-۲-۴ زنجیره بلوکی به عنوان راهکار نوین در زنجیره تأمین

با در نظر گرفتن مباحث مطرح شده، اکنون می‌توانیم تصویر کامل‌تری از چرایی انتخاب زنجیره بلوکی به عنوان راهکار اصلی این پروژه ترسیم کنیم. فناوری زنجیره بلوکی، با ترکیب تاریخچه‌ای تکاملی در جهت عدم تمرکز، مبانی مستحکم رمزنگاری و قابلیت‌های برنامه‌پذیری از طریق قراردادهای هوشمند، مجموعه‌ای از ابزارهای قدرتمند را برای مقابله با چالش‌های زنجیره تأمین فراهم می‌آورد.

ترکیب این مفاهیم، یک راهکار یکپارچه ارائه می‌دهد:

۱. **اصالت تضمین‌شده:** با استفاده از توابع هش رمزنگاری، برای هر محصول یک هویت دیجیتال منحصربه‌فرد و تغییرناپذیر ایجاد می‌شود. این هویت، جعل محصول را تقریبا غیرممکن می‌سازد.

۲. **مالکیت امن:** با استفاده از رمزنگاری کلید عمومی، مالکیت هر کالا به صورت امن به آدرس دیجیتال مالک آن گره می‌خورد و انتقال آن تنها با امضای دیجیتال مالک (کلید خصوصی) امکان‌پذیر است.

۳. **فرآیندهای خودکار و شفاف:** با استفاده از قراردادهای هوشمند، قوانین مربوط به انتقال مالکیت، تأیید مراحل و حتی محاسبه مالیات، به صورت کد تعریف شده و به طور خودکار و بدون نیاز به واسطه اجرا می‌شوند. تمام این فرآیندها بر روی یک دفتر کل شفاف و قابل حسابرسی ثبت می‌گردد.

در نتیجه، زنجیره بلوکی بستری را فراهم می‌کند که در آن، اعتماد دیگر به یک نهاد مرکزی وابسته نیست، بلکه در خود معماری سیستم و قوانین ریاضی و رمزنگاری آن نهفته است. این همان تغییری است که می‌تواند بر مشکلات ساختاری زنجیره‌های تأمین سنتی غلبه کرده و عصری جدید از شفافیت، کارایی و اطمینان را برای همه ذی‌نفعان به ارمغان آورد.

۱-۳ اهداف و دستاوردهای پروژه

همانطور که در بخش‌های پیشین تشریح شد، زنجیره‌های تأمین سنتی با بحران‌های عمیقی در حوزه‌های شفافیت، اعتماد و کارایی مواجه هستند. این چالش‌ها که ریشه در معماری متمرکز و گسستگی اطلاعات دارند، نیازمند راهکاری بنیادین هستند که بتواند ساختار تعاملات در این اکوسیستم را بازتعریف کند. پروژه حاضر با درک این نیاز، هدف اصلی خود را «طراحی و پیاده‌سازی یک سامانه جامع زنجیره تأمین مبتنی بر فناوری زنجیره بلوکی» تعریف کرده است[؟]. این هدف کلان، در پی آن است تا با بهره‌گیری از ویژگی‌های منحصربه‌فرد زنجیره بلوکی، راهکاری عملی برای مقابله با تقلب، افزایش قابلیت ردیابی و بازگرداندن اعتماد به اکوسیستم ارائه دهد.

برای نیل به این هدف جامع، مجموعه‌ای از اهداف جزئی، فنی و کاربردی تعریف شده‌اند که هر یک به مثابه یک ستون، شاکله اصلی این سامانه را تشکیل می‌دهند. این اهداف نه تنها مسیر پیاده‌سازی پروژه را مشخص می‌کنند، بلکه در نهایت، دستاوردهای ملموس و قابل سنجش آن را نیز نمایندگی خواهند کرد. در ادامه این بخش، هر یک از این اهداف کلیدی به تفصیل مورد بررسی و تحلیل قرار می‌گیرند تا اهمیت، ضرورت و نحوه تحقق هر یک از آن‌ها به روشنی مشخص گردد.

۱-۳-۱ هدف اول: ایجاد یک سیستم جامع برای ردیابی شفاف محصولات

تشریح هدف و اهمیت آن

اولین و پایه‌ای‌ترین هدف این پروژه، ایجاد یک سیستم یکپارچه برای ردیابی سرتاسری و شفاف محصولات ($End-to-EndTraceability$) است. در سیستم‌های کنونی، چرخه حیات یک محصول از مجموعه‌ای از مراحل گسسته و جزیره‌ای تشکیل شده است که هر کدام توسط یک نهاد مجزا مدیریت می‌شود. این گسستگی اطلاعاتی باعث ایجاد «نقاط کور» در زنجیره می‌شود که ردیابی دقیق مسیر حرکت، تاریخچه مالکیت و شرایط نگهداری محصول را ناممکن می‌سازد.

هدف این است که یک «شناسنامه دیجیتال» برای هر محصول ایجاد شود که از لحظه تولید تا رسیدن به دست مصرف‌کننده نهایی، به صورت پویا و تغییرناپذیر تکمیل گردد. این شناسنامه بر روی یک دفتر کل توزیع‌شده ثبت می‌شود که تمام ذی‌نفعان مجاز (تولیدکننده، توزیع‌کننده، نهادهای نظارتی و مصرف‌کننده) می‌توانند به آن دسترسی داشته باشند[؟].

اهمیت این هدف در سه جنبه اصلی نهفته است:

۱. **مقابله با تقلب و جعل:** با داشتن یک تاریخچه کامل و غیرقابل دستکاری، امکان ورود کالای تقلبی به زنجیره اصلی به شدت کاهش می‌یابد. هرگونه عدم تطابق در تاریخچه محصول، به سرعت قابل شناسایی خواهد بود.

۲. **افزایش اعتماد مصرف‌کننده:** مصرف‌کنندگان می‌توانند با اطمینان کامل از اصالت و پیشینه محصولی که خریداری می‌کنند، مطلع شوند. این شفافیت، وفاداری به برند را تقویت کرده و قدرت

انتخاب آگاهانه را به مصرف‌کننده می‌دهد.

۳. **مدیریت بحران و فراخوان کارآمد:** در صورت بروز مشکل کیفی یا ایمنی برای یک محصول خاص، می‌توان با مراجعه به تاریخچه دقیق آن، به سرعت منشأ مشکل را شناسایی و محصولات معیوب را از بازار جمع‌آوری (□□□□□□) کرد. این امر از توزیع گسترده‌تر محصولات مشکل‌دار جلوگیری کرده و خسارات را به حداقل می‌رساند.

نحوه تحقق و پیاده‌سازی فنی

برای دستیابی به این هدف، از یک مدل داده ساختاریافته در قرارداد هوشمند استفاده می‌شود. فرآیند ردیابی در سه مرحله اصلی پیاده‌سازی می‌شود:

- **ثبت محصول** ($Minting$): در ابتدای چرخه حیات، تولیدکننده یا واردکننده محصول جدید را در سیستم ثبت می‌کند. در این مرحله، یک توکن دیجیتال منحصربه‌فرد که نمایانگر آن کالای فیزیکی است، بر روی زنجیره بلوکی «ضرب» یا ساخته می‌شود. تمام اطلاعات اولیه محصول، مانند شماره سریال، تاریخ تولید و مشخصات فنی، به این توکن الصاق می‌گردد. این عمل از طریق فراخوانی یک تابع مشخص در قرارداد هوشمند (مانند $registerProduct$) توسط بازیگر دارای مجوز (مثلاً نقش $MANUFACTURER_ROLE$) انجام می‌شود.
- **ثبت تاریخچه مالکیت:** هر بار که محصول در زنجیره تأمین دست به دست می‌شود (مثلاً از تولیدکننده به توزیع‌کننده)، یک تراکنش انتقال مالکیت بر روی زنجیره بلوکی ثبت می‌گردد. این تراکنش که از طریق توابعی مانند $transferWithTax$ در قرارداد هوشمند مدیریت می‌شود، به صورت خودکار اطلاعات مالک جدید، زمان انتقال و سایر جزئیات مربوطه را به تاریخچه محصول اضافه می‌کند. این فرآیند، یک زنجیره مالکیت ($ChainofCustody$) شفاف و قابل حسابرسی ایجاد می‌کند که در تابع $getOwnershipHistory$ قابل بازیابی است.
- : □□..[؟].

۱-۴ هدف دوم: مدیریت بهینه دارایی‌ها با استفاده از استاندارد $ERC1155$

تشریح هدف و اهمیت آن

زنجیره‌های تأمین با طیف گسترده‌ای از محصولات سروکار دارند. برخی از محصولات، مانند یک خودرو با شماره شاسی مشخص، کاملاً منحصربه‌فرد و غیرمثلی ($Non-Fungible$) هستند. در مقابل، محصولات دیگری مانند یک بچ از هزاران پیچ یکسان، کاملاً مثلی و قابل تعویض ($Fungible$) هستند. مدیریت این

دو نوع دارایی در سیستم‌های سنتی و حتی در استانداردهای اولیه زنجیره بلوکی، نیازمند زیرساخت‌ها و قراردادهای مجزا بود. این امر منجر به افزایش پیچیدگی، هزینه‌های بالا و کاهش کارایی می‌شد.

هدف این بخش از پروژه، بهره‌گیری از یک استاندارد توکن پیشرفته به نام $ERC1155$ است تا بتوان هر دو نوع دارایی مثلی و غیرمثلی را در قالب یک قرارداد هوشمند واحد، به صورت بهینه مدیریت کرد[؟، ؟]. این استاندارد که به عنوان یک استاندارد «چند-توکنی» شناخته می‌شود، به طور خاص برای کاربردهایی مانند بازی‌های کامپیوتری و زنجیره تأمین که با انواع مختلفی از آیتم‌ها سروکار دارند، طراحی شده است.

اهمیت استفاده از $ERC1155$ در موارد زیر خلاصه می‌شود:

- **افزایش کارایی و کاهش هزینه:** به جای استقرار چندین قرارداد هوشمند مجزا (مثلاً یک قرارداد $ERC-721$ برای کالاهای منحصربه‌فرد و یک قرارداد $ERC-20$ برای کالاهای مثلی)، تمام منطق مدیریت توکن‌ها در یک قرارداد واحد متمرکز می‌شود. این امر به شدت هزینه‌های استقرار و نگهداری ($GasFee$) را کاهش داده و مدیریت سیستم را ساده‌تر می‌کند[؟].
- **انعطاف‌پذیری بالا:** این سامانه قادر خواهد بود تا هر نوع محصولی را، از یک قطعه هنری با اصالت مشخص گرفته تا یک پالت از کالاهای مصرفی، به راحتی مدیریت کند. این انعطاف‌پذیری، کاربردپذیری سیستم را برای طیف وسیعی از صنایع ممکن می‌سازد.
- **تراکنش‌های دسته‌ای** ($BatchOperations$)**:** یکی از قابلیت‌های کلیدی $ERC1155$، امکان انتقال چندین نوع توکن مختلف در یک تراکنش واحد است. برای مثال، یک توزیع‌کننده می‌تواند در یک تراکنش، ۱۰۰ عدد کالای □ و ۵۰ عدد کالای □ را از تولیدکننده دریافت کند. این قابلیت، تعداد کل تراکنش‌های مورد نیاز در شبکه را کاهش داده و به بهینه‌سازی فرآیندهای لجستیکی پیچیده کمک شایانی می‌کند.

نحوه تحقق و پیاده‌سازی فنی

قرارداد هوشمند اصلی این پروژه ($SupplyChainERC1155.sol$) با ارث‌بری از پیاده‌سازی استاندارد $ERC1155$ (که معمولاً توسط کتابخانه‌های معتبری مانند $OpenZeppelin$ ارائه می‌شود[؟]) ساخته شده است. هر نوع محصول جدید در سیستم با یک id منحصربه‌فرد تعریف می‌شود. اگر محصول غیرمثلی باشد، تنها یک توکن با آن id ساخته می‌شود. اگر محصول مثلی باشد، می‌توان هر تعداد توکن با همان id ایجاد کرد. توابع اصلی این استاندارد مانند $mint_m$، $burn_b$ و $safeTransferFrom$ برای مدیریت ایجاد، ابطال و انتقال این توکن‌ها به کار گرفته می‌شوند. این ساختار فنی، پایه و اساس مدیریت دارایی در کل سامانه را تشکیل می‌دهد.

۱-۴-۱ هدف سوم: تضمین صحت و یکپارچگی فراداده‌ها با $Keccak256$

تشریح هدف و اهمیت آن

ردیابی مالکیت یک کالا تنها نیمی از راه حل است. بخش دیگر و حیاتی‌تر، تضمین این است که اطلاعات و مشخصات آن کالا (فراداده یا $Metadata$) در طول زمان دستکاری نشده و معتبر باقی مانده است. فراداده شامل جزئیاتی مانند تاریخ تولید، شماره سریال، مبدأ جغرافیایی، مواد تشکیل‌دهنده و گواهی‌های کیفیت است. در سیستم‌های سنتی، این اطلاعات معمولاً در پایگاه‌های داده‌ای ذخیره می‌شوند که به راحتی قابل تغییر هستند.

هدف این بخش، پیاده‌سازی یک مکانیزم رمزنگاری قدرتمند برای تضمین صحت و یکپارچگی فراداده‌هاست. در این پروژه، از الگوریتم درهم‌سازی $Keccak256$ برای ایجاد یک «اثر انگشت دیجیتال» منحصربه‌فرد از فراداده هر محصول استفاده می‌شود[؟، ؟]. این اثر انگشت (هش) بر روی زنجیره بلوکی ذخیره می‌شود که تغییرناپذیر است.

اهمیت این رویکرد در دو نکته کلیدی است:

۱. **ایجاد پیوند تغییرناپذیر بین کالا و اطلاعات آن:** با ثبت هش فراداده بر روی زنجیره، هرگونه تلاش برای دستکاری اطلاعات اصلی (حتی تغییر یک کاراکتر) منجر به تولید یک هش کاملاً متفاوت خواهد شد. این عدم تطابق به راحتی قابل تشخیص بوده و تلاش برای تقلب را آشکار می‌سازد.

۲. **بهینه‌سازی هزینه ذخیره‌سازی:** ذخیره‌سازی حجم زیادی از اطلاعات (مانند تصاویر یا اسناد فنی) به صورت مستقیم بر روی زنجیره بلوکی بسیار گران است. این روش به ما اجازه می‌دهد تا فراداده اصلی را در یک سیستم ذخیره‌سازی خارج از زنجیره ($Off-chain$) مانند □□□□ یا سرورهای معمولی نگهداری کرده و تنها هش سبک و امن آن را بر روی زنجیره ($On-chain$) ثبت کنیم. این معماری، ضمن حفظ امنیت کامل، هزینه‌ها را به شدت بهینه می‌کند.

نحوه تحقق و پیاده‌سازی فنی

فرآیند تضمین صحت فراداده‌ها در قرارداد هوشمند به شرح زیر پیاده‌سازی می‌شود:

- **تولید هش در زمان ثبت:** هنگامی که یک محصول جدید از طریق تابع $registerProduct$ ثبت می‌شود، قرارداد هوشمند به صورت داخلی تابع دیگری مانند $_{g}enerateMetadataHash$ را فراخوانی می‌کند. این تابع، مقادیر کلیدی فراداده (مانند نام، دسته بندی، شماره سریال و...) را دریافت کرده، آن‌ها را به یک فرمت استاندارد تبدیل می‌کند و سپس الگوریتم $keccak256$ را بر روی آن اعمال می‌کند.
- **ذخیره‌سازی هش:** هش تولید شده در ساختار داده مربوط به آن محصول (مثلاً $Productstruct$) در کنار سایر اطلاعات آن بر روی زنجیره بلوکی ذخیره می‌شود.

- **اعتبارسنجی عمومی:** یک تابع عمومی مانند $verifyProductMetadata$ در قرارداد هوشمند در دسترس قرار می‌گیرد. هر کاربری (مثلاً یک مصرف‌کننده یا بازرس) می‌تواند با ارائه فراداده‌ای که در اختیار دارد، این تابع را فراخوانی کند. قرارداد هوشمند در لحظه، هش فراداده ارسالی را محاسبه کرده و آن را با هش ذخیره‌شده بر روی زنجیره مقایسه می‌کند. نتیجه این مقایسه (که یک مقدار صحیح/غلط است) به کاربر بازگردانده شده و بدین ترتیب، اصالت اطلاعات تأیید یا رد می‌شود[؟].

۱-۵ هدف چهارم: خودکارسازی فرآیندهای تجاری و مالی

تشریح هدف و اهمیت آن

زنجیره‌های تأمین سنتی مملو از فرآیندهای دستی، کاغذبازی‌های اداری، تأخیر در پرداخت‌ها و رویه‌های پیچیده مالیاتی هستند. این فرآیندها نه تنها کند و پرهزینه هستند، بلکه به دلیل نیاز به دخالت انسانی، مستعد خطا و فساد نیز می‌باشند. بخش قابل توجهی از این ناکارآمدی‌ها ناشی از نیاز به واسطه‌های متعدد برای تأیید مراحل، پردازش پرداخت‌ها و تضمین اجرای تعهدات است.

هدف این بخش از پروژه، استفاده از قابلیت‌های قراردادهای هوشمند برای **خودکارسازی منطق تجاری و مالی** زنجیره تأمین است[؟]. با کدنویسی قوانین کسب‌وکار به صورت مستقیم در یک قرارداد هوشمند، می‌توان اجرای آن‌ها را به صورت خودکار، قطعی و بدون نیاز به دخالت یا نظارت انسانی تضمین کرد.

اهمیت این هدف عبارت است از:

- **افزایش سرعت و کارایی:** خودکارسازی فرآیندها، تأخیرهای ناشی از هماهنگی‌های انسانی و پردازش‌های دستی را از بین برده و سرعت کل زنجیره را به طور چشمگیری افزایش می‌دهد.
- **کاهش هزینه‌های عملیاتی:** حذف یا کاهش نیاز به واسطه‌هایی که برای اموری مانند خدمات امانی ($Escrow$) یا پردازش اسناد به کار گرفته می‌شوند، منجر به صرفه‌جویی قابل توجهی در هزینه‌ها می‌شود[؟].
- **شفافیت و سازگاری در اجرا:** وقتی قوانین در قالب کد نوشته می‌شوند، به صورت یکسان و بدون تبعیض برای همه تراکنش‌ها اجرا می‌گردند. این امر از اجرای سلیقه‌ای قوانین جلوگیری کرده و شفافیت را در کل فرآیند حاکم می‌کند.

نحوه تحقق و پیاده‌سازی فنی

دو نمونه برجسته از خودکارسازی در این پروژه پیاده‌سازی شده است:

۱. **انتقال مالکیت خودکار:** تابع $transferWithTax$ در قرارداد هوشمند، فرآیند انتقال توکن از فرستنده به گیرنده را مدیریت می‌کند. این تابع به صورت اتمی عمل می‌کند؛ یعنی انتقال تنها در صورتی انجام می‌شود که تمام شروط لازم (مانند وجود توکن در کیف پول فرستنده) برقرار باشد. این فرآیند جایگزین رویه‌های سنتی مبتنی بر بارنامه و اسناد کاغذی می‌شود.

۲. **محاسبه خودکار مالیات:** یکی از قابلیت‌های نوآورانه این پروژه، تعبیه منطق محاسبه مالیات به صورت مستقیم در قرارداد هوشمند است[؟، ؟]. در قرارداد، تابعی مانند $alculateTax_c$ تعریف شده است که می‌تواند بر اساس پارامترهایی مانند نوع کالا یا ارزش تراکنش، مبلغ مالیات متعلقه را محاسبه کند. این تابع می‌تواند به صورت خودکار در حین فرآیند انتقال مالکیت فراخوانی شود. مبلغ مالیات محاسبه‌شده می‌تواند به یک آدرس از پیش تعیین‌شده (مثلاً کیف پول سازمان امور مالیاتی) ارسال گردد. این مکانیزم، فرآیند محاسبه و جمع‌آوری مالیات را شفاف، دقیق و آنی می‌سازد و بار محاسباتی را از دوش کسب‌وکارها برمی‌دارد.

در مجموع، این چهار هدف کلیدی، یک نقشه راه جامع برای ساختن یک زنجیره تأمین مدرن، شفاف و قابل اعتماد را ترسیم می‌کنند. هر یک از این اهداف، ضمن حل یکی از مشکلات اساسی سیستم‌های سنتی، در ترکیب با یکدیگر، یک راهکار هم‌افزا و قدرتمند را شکل می‌دهند که پتانسیل تحول‌آفرینی در این صنعت حیاتی را داراست.

۱-۶ چالش‌های اصلی پروژه

با وجود پتانسیل عظیم فناوری زنجیره بلوکی برای ایجاد تحول در صنایع مختلف و به‌ویژه در زنجیره تأمین، پیاده‌سازی و استقرار یک سامانه عملیاتی مبتنی بر این فناوری با چالش‌های متعدد و پیچیده‌ای همراه است. این چالش‌ها صرفاً فنی نیستند و ابعاد امنیتی، اقتصادی، قانونی و اجتماعی را نیز در بر می‌گیرند. موفقیت این پروژه در گرو شناسایی دقیق این موانع و ارائه راهکارهای مناسب برای غلبه بر آن‌هاست. در واقع، هر یک از این چالش‌ها، خود یک حوزه پژوهشی و مهندسی مستقل به شمار می‌آید که نیازمند بررسی عمیق و راهکارهای نوآورانه است. در این فصل، به تفصیل به تحلیل چهار چالش اصلی پیش روی این پروژه می‌پردازیم: چالش‌های فنی مرتبط با مقیاس‌پذیری و هزینه، چالش‌های امنیتی در یک محیط غیرمتمرکز، چالش‌های پذیرش و تجربه کاربری، و در نهایت، چالش‌های قانونی و نظارتی.

۱-۶-۱ چالش‌های فنی: مقیاس‌پذیری و هزینه

یکی از برجسته‌ترین و بحث‌برانگیزترین چالش‌ها در دنیای زنجیره بلوکی، مسئله مقیاس‌پذیری ($Scalability$) و هزینه‌های مرتبط با آن است. در حالی که سیستم‌های متمرکز سنتی می‌توانند ده‌ها هزار تراکنش را در ثانیه پردازش کنند، شبکه‌های زنجیره بلوکی عمومی مانند اتریوم، به دلیل ماهیت غیرمتمرکز و

سازوکارهای اجماع خود، دارای توان پردازشی بسیار محدودتری هستند. این محدودیت، به ویژه در کاربردهایی با حجم تراکنش بالا مانند زنجیره تأمین، به یک گلوگاه اساسی تبدیل می‌شود.

۱-۶-۲ مقدمه‌ای بر معماری ماشین مجازی اتریوم (EVM) و هزینه تراکنش (Gas)

برای درک عمیق چالش هزینه، ابتدا باید با مفهوم «گس» (Gas) در شبکه‌های سازگار با ماشین مجازی اتریوم (EVM) آشنا شویم. هر عملیات محاسباتی که بر روی EVM انجام می‌شود، از یک جمع ساده گرفته تا ذخیره‌سازی داده در قرارداد هوشمند، نیازمند مصرف منابع محاسباتی از سوی گره‌های ($Nodes$) شبکه است. این منابع رایگان نیستند[؟]. مفهوم گس برای اندازه‌گیری میزان این تلاش محاسباتی به کار می‌رود[؟]. هر عملیات یک هزینه گس ثابت دارد (مثلاً ADD هزینه ۳ گس و $SSTORE$ برای ذخیره‌سازی داده هزینه ۲۰،۰۰۰ گس دارد).

هزینه نهایی یک تراکنش از فرمول زیر به دست می‌آید:

$$TransactionFee = TotalGasUsed \times GasPrice$$

در اینجا، $TotalGasUsed$ مجموع گس مصرفی تمام عملیات‌های یک تراکنش است و $GasPrice$ قیمتی است که کاربر مایل است برای هر واحد گس بپردازد. این قیمت بر اساس عرضه و تقاضای شبکه تعیین می‌شود و در زمان‌های شلوغی شبکه، به شدت افزایش می‌یابد. پرداخت این هزینه با استفاده از ارز دیجیتال اصلی شبکه، یعنی اتر ،(□□□□□) انجام می‌شود[؟]. این سازوکار، ضمن جلوگیری از اجرای کدهای مخرب و حلقه‌های بی‌نهایت، یک مدل اقتصادی برای پاداش‌دهی به اعتبارسنج‌های شبکه فراهم می‌کند. اما همین مدل، چالش هزینه را برای کاربردهای تجاری به وجود می‌آورد.

۱-۶-۳ چالش‌های خاص استاندارد $ERC1155$ در مقیاس بزرگ

پروپوزال این پروژه به درستی به این چالش اشاره می‌کند که «استفاده از قرارداد هوشمند مبتنی بر استاندارد $ERC1155$، به ویژه در مقیاس بزرگ، می‌تواند مشکلاتی از جمله مقیاس‌پذیری و هزینه‌های تراکنش ایجاد کند»[؟]. در یک زنجیره تأمین واقعی، به‌ویژه برای کالاهای تندمصرف ($FMCG$)، ممکن است روزانه هزاران یا حتی میلیون‌ها محصول تولید ($mint$)، منتقل و مصرف ($burn$) شوند. هر یک از این اقدامات، یک تراکنش مجزا بر روی زنجیره بلوکی است که هزینه گس به همراه دارد.

فرض کنید هزینه میانگین یک تراکنش انتقال ساده در شبکه اتریوم چند دلار باشد. اگر یک شرکت بخواهد روزانه وضعیت ۱۰۰۰ محصول را به‌روزرسانی کند، هزینه عملیاتی آن به سرعت به هزاران دلار در روز می‌رسد. این هزینه برای بسیاری از کسب‌وکارها، به ویژه در مقایسه با هزینه‌های ناچیز نگهداری یک پایگاه داده متمرکز، غیرقابل قبول است. بنابراین، هرچند $ERC1155$ از نظر فنی برای مدیریت انواع

توکن‌ها کارآمد است، اما هزینه اقتصادی استفاده از آن در یک شبکه عمومی پرازدحام، یک مانع جدی برای پذیرش در مقیاس صنعتی محسوب می‌شود.

۱-۶-۴ چالش ذخیره‌سازی داده‌ها بر روی زنجیره

یکی دیگر از ابعاد چالش هزینه، مربوط به ذخیره‌سازی داده است. ذخیره‌سازی داده به صورت مستقیم بر روی زنجیره بلوکی ($On-chainStorage$) یکی از گران‌ترین عملیات‌ها در EVM است. هر کیلوبایت داده می‌تواند صدها یا هزاران دلار هزینه در بر داشته باشد. برای یک زنجیره تأمین که نیازمند ذخیره اطلاعات جامعی از هر محصول (مانند تصاویر، اسناد فنی، گواهی‌نامه‌ها و...) است، ذخیره‌سازی مستقیم این فراداده‌ها بر روی زنجیره، از نظر اقتصادی کاملاً غیرممکن است.

این چالش، تیم پروژه را به سمت یک معماری هوشمندانه سوق داده است که در بخش اهداف نیز به آن اشاره شد: معماری ترکیبی $On-chain$ و $Off-chain$. در این مدل، تنها اطلاعات حیاتی و حداقلی که برای تضمین امنیت و یکپارچگی لازم است، بر روی زنجیره ذخیره می‌شود. این اطلاعات شامل هش رمزنگاری شده فراداده است. خود فراداده حجیم، در یک سیستم ذخیره‌سازی خارج از زنجیره ($Off-chain$) مانند $IPFS$ ($InterPlanetaryFileSystem$) یا سرورهای وب سنتی نگهداری می‌شود. این رویکرد، ضمن حفظ امنیت کامل از طریق هش، هزینه‌های ذخیره‌سازی را هزاران برابر کاهش می‌دهد و سیستم را از نظر اقتصادی عملیاتی می‌سازد.

۱-۶-۵ راهکارهای بالقوه برای غلبه بر چالش فنی

اگرچه این پروژه بر روی یک شبکه تستی و محلی اجرا می‌شود، اما برای استقرار نهایی در دنیای واقعی، باید راهکارهایی برای چالش مقیاس‌پذیری و هزینه اندیشیده شود. برخی از مهم‌ترین راهکارها که در اکوسیستم زنجیره بلوکی در حال توسعه هستند عبارتند از:

- **شبکه‌های لایه ۲** ($Layer2Solutions$): فناوری‌هایی مانند $Rollups(OptimisticZK-Rollups$) تراکنش‌ها را در خارج از زنجیره اصلی پردازش کرده و تنها یک خلاصه فشرده از آن‌ها را به زنجیره اصلی ارسال می‌کنند. این کار هزینه هر تراکنش را به شدت کاهش داده و توان پردازشی را به چندین هزار تراکنش در ثانیه افزایش می‌دهد.
- **زنجیره‌های جانبی** ($Sidechains$): زنجیره‌های مستقلی که با زنجیره اصلی سازگار هستند و می‌توان دارایی‌ها را بین آن‌ها منتقل کرد. این زنجیره‌ها معمولاً دارای هزینه تراکنش بسیار پایین‌تری هستند.
- **انتخاب شبکه‌های** $EVM-Compatible$ **با هزینه پایین:** به جای استقرار بر روی شبکه اصلی اتریوم، می‌توان پروژه را بر روی شبکه‌های دیگری که با EVM سازگار هستند اما هزینه تراکنش کمتری دارند (مانند $Polygon$، $Avalanche$، $BNBSmartChain$) مستقر کرد.

انتخاب راهکار مناسب، خود نیازمند تحلیل دقیق نیازمندی‌های پروژه و بررسی مزایا و معایب هر گزینه است که می‌تواند موضوعی برای تحقیقات آینده باشد.

۱-۷ چالش‌های امنیتی در سیستم‌های غیرمتمرکز

امنیت در سیستم‌های زنجیره بلوکی یک پارادایم کاملاً متفاوت از امنیت در سیستم‌های متمرکز است. در اینجا، دیگر خبری از حفاظت از یک سرور مرکزی با استفاده از فایروال‌ها و کنترل دسترسی‌های فیزیکی نیست. امنیت به خود پروتکل، کد قرارداد هوشمند و مسئولیت‌پذیری کاربران منتقل می‌شود. پروپوزال پروژه به درستی تأکید می‌کند که «امنیت اطلاعات در سیستم زنجیره بلوکی باید در بالاترین سطح خود قرار گیرد تا از هرگونه دستکاری داده‌ها جلوگیری شود»[؟].

۱-۷-۱ امنیت قرارداد هوشمند: کد، قانون است

قراردادهای هوشمند، قلب تپنده برنامه‌های غیرمتمرکز هستند و در عین حال، بزرگ‌ترین سطح حمله ($Surface\ Attack$) را تشکیل می‌دهند. یک آسیب‌پذیری کوچک در کد یک قرارداد هوشمند می‌تواند منجر به سرقت میلیون‌ها دلار دارایی یا از کار افتادن کامل یک سیستم شود. چالش اصلی در اینجا، ویژگی **تغییرناپذیری** ($Immutability$) کد است. پس از استقرار یک قرارداد هوشمند، کد آن دیگر قابل تغییر یا اصلاح نیست[؟]. این ویژگی که برای ایجاد اعتماد ضروری است، به این معناست که اگر یک باگ یا حفره امنیتی در کد وجود داشته باشد، نمی‌توان آن را به سادگی «پچ» کرد. این ماهیت، امنیت قرارداد هوشمند را به امری بسیار حیاتی و پرمخاطره تبدیل می‌کند.

برخی از آسیب‌پذیری‌های رایج در قراردادهای هوشمند عبارتند از:

- **حملات بازگشتی** ($Reentrancy$): حمله‌ای که در آن یک قرارداد مهاجم، قبل از تکمیل یک تراکنش، به صورت مکرر یک تابع را در قرارداد قربانی فراخوانی کرده و موجودی آن را خالی می‌کند.
- **سرریز/زیرریز عدد صحیح** ($IntegerOverflow/Underflow$): به دلیل محدودیت در اندازه متغیرهای عددی، انجام محاسباتی که منجر به عبور از حداکثر یا حداقل مقدار ممکن شود، می‌تواند نتایج غیرمنتظره و خطرناکی به همراه داشته باشد.
- **منطق اشتباه در کنترل دسترسی:** عدم پیاده‌سازی صحیح مجوزها و نقش‌ها، که می‌تواند به یک کاربر غیرمجاز اجازه دهد تا اقداماتی مدیریتی مانند تغییر مالکیت یا از بین بردن دارایی‌ها را انجام دهد.

برای مقابله با این چالش‌ها، پروژه حاضر از رویکردهای استاندارد صنعتی بهره می‌برد. اول، استفاده از کتابخانه‌های معتبر و حسابرسی‌شده مانند $OpenZeppelin$[؟] برای پیاده‌سازی استانداردهایی مانند

$ERC1155$ و مکانیزم‌های کنترل دسترسی. این کتابخانه‌ها توسط متخصصان امنیت بررسی شده و ریسک وجود آسیب‌پذیری‌های رایج را به حداقل می‌رسانند. دوم، پیاده‌سازی یک مجموعه آزمون جامع با استفاده از فریم‌ورک قدرتمند $Foundry$. این آزمون‌ها، تمام توابع و سناریوهای ممکن، از جمله حالت‌های حدی و تلاش برای حملات، را شبیه‌سازی کرده و از صحت عملکرد و امنیت کد اطمینان حاصل می‌کنند.

۱-۸ امنیت فراداده و مکانیزم تأیید هش

همانطور که قبلاً ذکر شد، معماری این سیستم بر پایه ذخیره هش فراداده بر روی زنجیره و خود فراداده در یک مکان خارج از زنجیره (مانند $IPFS$) استوار است. این معماری، خود یک چالش امنیتی جدید ایجاد می‌کند: چگونه از صحت و تطابق داده $off-chain$ با هش $on-chain$ اطمینان حاصل کنیم؟

۱-۸-۱ بردارهای حمله به فراداده

یک مهاجم نمی‌تواند هش ثبت‌شده بر روی زنجیره بلوکی را تغییر دهد، اما می‌تواند تلاش کند تا به یکی از روش‌های زیر، سیستم را فریب دهد:

- **حمله مرد میانی** ($Man-in-the-Middle$): یک مهاجم می‌تواند در ارتباط بین کاربر و سرور ذخیره‌سازی $off-chain$ قرار گرفته و فراداده جعلی را به کاربر نمایش دهد، در حالی که کاربر تصور می‌کند در حال مشاهده اطلاعات اصلی است.
- **دستکاری سرور** $Off-chain$: اگر فراداده بر روی یک سرور متمرکز سنتی ذخیره شده باشد، مهاجم می‌تواند با هک کردن آن سرور، اطلاعات را تغییر دهد.
- **عدم دسترسی به داده** ($DataUnavailability$): ممکن است سرور $off-chain$ از دسترس خارج شود و کاربران دیگر نتوانند به فراداده اصلی دسترسی پیدا کنند، که این امر عملاً اعتبارسنجی را غیرممکن می‌سازد.

مکانیزم دفاعی پروژه

سیستم طراحی‌شده در این پروژه، یک مکانیزم دفاعی قوی برای مقابله با این حملات دارد که مبتنی بر **اعتبارسنجی سمت کاربر** ($Client-SideValidation$) است. فرآیند به شرح زیر است:

۱. کاربر (مثلاً مصرف‌کننده‌ای که کد QR را اسکن می‌کند) درخواستی برای مشاهده اطلاعات محصول ارسال می‌کند.

۲. برنامه کاربردی ($Front-end$) دو درخواست موازی ارسال می‌کند: یکی به سیستم ذخیره‌سازی $off-chain$ (مثلاً $IPFS$) برای دریافت فایل کامل فراداده، و دیگری به زنجیره بلوکی برای خواندن هش معتبر و ثبت‌شده آن محصول از قرارداد هوشمند.

۳. پس از دریافت فایل فراداده، برنامه کاربردی در سمت کاربر، تابع هش $Keccak256$ را بر روی آن اجرا کرده و هش آن را به صورت محلی محاسبه می‌کند.

۴. در نهایت، برنامه، هش محاسبه‌شده محلی را با هش دریافت‌شده از زنجیره بلوکی مقایسه می‌کند.

اگر این دو هش کاملاً یکسان باشند، یک علامت تأیید سبز به کاربر نمایش داده می‌شود که نشان‌دهنده اصالت و یکپارچگی کامل اطلاعات است. اگر حتی یک بیت تفاوت بین دو هش وجود داشته باشد، به کاربر یک هشدار جدی نمایش داده می‌شود که اطلاعات محصول مورد دستکاری قرار گرفته است. این فرآیند، اعتماد را از سرور $off-chain$ سلب کرده و آن را به محاسبات ریاضی و داده‌های تغییرناپذیر زنجیره بلوکی منتقل می‌کند.

نقش $IPFS$ در افزایش امنیت

استفاده از $IPFS$ (که در توضیحات تکمیلی شما به آن اشاره شد) یک لایه امنیتی دیگر به این معماری می‌افزاید. $IPFS$ یک سیستم فایل توزیع‌شده و مبتنی بر محتوا ($Content-Addressed$) است. این بدان معناست که آدرس یک فایل در $IPFS$، خودِ هشِ محتوای آن فایل است. بنابراین، اگر محتوای فایل تغییر کند، هش آن و در نتیجه آدرس آن نیز تغییر خواهد کرد. با ذخیره کردن این آدرس مبتنی بر محتوا (CID) بر روی زنجیره بلوکی، یک پیوند رمزنگاری قوی بین رفرنس آن‌چین و داده آف‌چین ایجاد می‌شود که دستکاری آن را بیش از پیش دشوار می‌سازد.

۱-۸-۲ امنیت کلید خصوصی کاربر

نهایتاً، ضعیف‌ترین حلقه در زنجیره امنیت هر سیستم مبتنی بر زنجیره بلوکی، خود کاربر است. تمام دارایی‌ها و مجوزهای یک کاربر به کلید خصوصی او گره خورده است. اگر کلید خصوصی یک کاربر به سرقت برود یا فاش شود، مهاجم کنترل کاملی بر تمام دارایی‌ها و نقش‌های آن کاربر در سیستم خواهد داشت. این چالشی است که راه حل آن کمتر فنی و بیشتر آموزشی است. کاربران باید در مورد اهمیت نگهداری امن کلیدهای خصوصی خود و استفاده از کیف پول‌های سخت‌افزاری برای دارایی‌های با ارزش، به خوبی آموزش ببینند.

۱-۹ چالش‌های پذیرش و تجربه کاربری (UX)

یک سیستم هرچقدر هم که از نظر فنی قدرتمند و امن باشد، اگر استفاده از آن برای کاربران نهایی دشوار و پیچیده باشد، هرگز به پذیرش گسترده نخواهد رسید. پروپوزال پروژه به درستی به این موضوع اشاره می‌کند که «پذیرش چنین سیستم نوآورانه‌ای در کشور نیازمند آموزش و آگاهی‌رسانی به کاربران است»[؟]. این چالش، به ویژه در صنعتی مانند زنجیره تأمین که با طیف وسیعی از کاربران با سطوح مختلف دانش فنی سروکار دارد، بسیار پررنگ‌تر است.

۱-۹-۱ فاصله دانش و موانع ذهنی

مفاهیمی مانند «زنجیره بلوکی»، «کیف پول دیجیتال»، «کلید خصوصی»، «امضای تراکنش» و «هزینه گس»، برای اکثر افراد خارج از دنیای فناوری، مفاهیمی بیگانه و ترسناک هستند. انتظار از یک مدیر انبار، یک راننده کامیون یا یک فروشنده خرده‌پا برای درک و کار با این مفاهیم، یک مانع بزرگ برای پیاده‌سازی موفق سیستم است. هدف اصلی در طراحی تجربه کاربری، **انتزاع** ($Abstraction$) این پیچیدگی‌ها و ارائه یک رابط کاربری ساده، آشنا و بصری است که به کاربران اجازه دهد بدون نیاز به درک جزئیات فنی زیرساخت، وظایف خود را به راحتی انجام دهند.

۱-۹-۲ طراحی تجربه کاربری برای انتزاع پیچیدگی

بر اساس توضیحات تکمیلی شما، پروژه حاضر با طراحی یک تجربه کاربری هدفمند، تلاش کرده است تا این چالش را مرتفع سازد. این طراحی بر اساس نقش‌های مختلف کاربران، شخصی‌سازی شده است:

تجربه کاربری مدیر سیستم / تولیدکننده

برای کاربری که مسئول ثبت محصولات جدید در سیستم است (مثلاً یک مدیر تولید)، یک **داشبورد مدیریتی وب** طراحی می‌شود. این داشبورد، تمام پیچیدگی‌های فنی را در پس‌زمینه پنهان می‌کند:

- **فرم ثبت محصول ساده:** کاربر با یک فرم وب ساده مواجه می‌شود که در آن فیلدهای آشنایی مانند «نام محصول»، «شماره سریال»، «دسته‌بندی»، «تاریخ تولید» و امکان بارگذاری تصویر و اسناد را مشاهده می‌کند.
- **فرآیند خودکار در پس‌زمینه:** پس از اینکه کاربر اطلاعات را وارد کرده و بر روی دکمه «ایجاد محصول» کلیک می‌کند، برنامه کاربردی ($Front-end$) به صورت خودکار زنجیره‌ای از عملیات پیچیده را انجام می‌دهد:

  ۱. ابتدا فراداده وارد شده را در یک فرمت استاندارد (مانند $JSON$) بسته‌بندی می‌کند.

۲. سپس این فایل فراداده را در سیستم ذخیره‌سازی $off-chain$ (مانند $IPFS$) بارگذاری می‌کند.
۳. پس از بارگذاری، آدرس منحصربه‌فرد فایل در $IPFS$ (یعنی CID آن) را دریافت می‌کند.
۴. هش $Keccak256$ فراداده را مطابق منطق قرارداد هوشمند محاسبه می‌کند.
۵. یک تراکنش برای فراخوانی تابع $registerProduct$ در قرارداد هوشمند آماده می‌کند. این تراکنش شامل پارامترهایی مانند هش فراداده و آدرس $IPFS$ آن است.
۶. در نهایت، از طریق یک کیف پول متصل به مرورگر (مانند $MetaMask$)، از کاربر می‌خواهد تا تراکنش را با یک کلیک ساده، «امضا» یا تأیید کند.

در تمام این فرآیند، کاربر تنها یک فرم را پر کرده و یک دکمه را فشرده است. او نیازی به دانستن اینکه $IPFS$ یا $Keccak256$ چیست، ندارد. این انتزاع، پذیرش سیستم توسط کاربران سازمانی را به شدت تسهیل می‌کند.

تجربه کاربری مالک توکن (توزیع‌کننده / خرده‌فروش)

یکی از نقاط قوت کلیدی این پروژه، پایبندی به استاندارد جهانی $ERC1155$ است. این پایبندی یک مزیت بزرگ در تجربه کاربری ایجاد می‌کند: محصول ثبت‌شده به عنوان یک توکن استاندارد، به صورت خودکار در تمام کیف پول‌های دیجیتالی که از این استاندارد پشتیبانی می‌کنند (مانند $MetaMask$، $TrustWallet$ و...) قابل مشاهده و مدیریت است. این یعنی یک توزیع‌کننده یا خرده‌فروش، محصول دیجیتال را دقیقاً مانند هر توکن یا NFT دیگری در کیف پول خود مشاهده می‌کند. او می‌تواند موجودی خود را ببیند، آن را به آدرس دیگری منتقل کند و تاریخچه تراکنش‌های آن را مشاهده نماید، همگی با استفاده از رابط کاربری آشنا و استاندارد کیف پول خود. این «قابلیت همکاری» ($Interoperability$) با اکوسیستم موجود، نیاز به ساخت یک کیف پول اختصاصی را از بین برده و به کاربران اجازه می‌دهد تا از ابزارهایی که از قبل با آن آشنا هستند، استفاده کنند.

تجربه کاربری مصرف‌کننده نهایی

ساده‌ترین و در عین حال مهم‌ترین تجربه کاربری، متعلق به مصرف‌کننده نهایی است. این کاربر نباید با هیچ‌گونه پیچیدگی فنی درگیر شود. فرآیند برای او باید به سادگی یک کلیک باشد:

۱. مصرف‌کننده با دوربین تلفن همراه خود، کد QR روی محصول را اسکن می‌کند.
۲. تلفن به صورت خودکار یک صفحه وب را باز می‌کند.
۳. این صفحه وب، که با طراحی بصری و جذاب ساخته شده، اطلاعات کلیدی محصول را نمایش می‌دهد: نام، تصویر، تاریخ تولید و مهم‌تر از همه، یک **تأییدیه اصالت** واضح (مثلاً یک تیک سبز بزرگ) به همراه تاریخچه کامل مالکیت محصول در یک خط زمانی ساده و قابل فهم.

در پس‌زمینه این فرآیند ساده، برنامه وب در حال انجام همان فرآیند پیچیده اعتبارسنجی هش است، اما کاربر نهایی هیچ‌کدام از این‌ها را نمی‌بیند. او تنها نتیجه نهایی را دریافت می‌کند: «این کالا اصیل است». این سادگی، هدف نهایی پروژه یعنی توانمندسازی مصرف‌کننده و ایجاد اعتماد را محقق می‌سازد.

۱-۹-۳ اهمیت آموزش و پشتیبانی

با وجود تمام تلاش‌ها برای ساده‌سازی تجربه کاربری، ماهیت نوآورانه این فناوری ایجاب می‌کند که فرآیندهای آموزش و پشتیبانی به عنوان بخشی جدایی‌ناپذیر از استقرار سیستم در نظر گرفته شوند[؟]. برگزاری کارگاه‌های آموزشی برای کاربران سازمانی، تهیه راهنماهای ویدیویی و متنی، و ایجاد یک کانال پشتیبانی برای پاسخگویی به سؤالات کاربران، نقشی حیاتی در کاهش مقاومت در برابر تغییر و تضمین استفاده صحیح و مؤثر از سامانه خواهد داشت.

۱-۱۰ چالش‌های قانونی و نظارتی

آخرین و شاید پیچیده‌ترین چالش، مربوط به انطباق سیستم با محیط قانونی و نظارتی کشور است. فناوری زنجیره بلوکی و دارایی‌های دیجیتال، مفاهیمی نسبتاً جدید هستند و چارچوب‌های قانونی برای آن‌ها در بسیاری از کشورها، از جمله ایران، هنوز در حال تکامل و بعضاً مبهم است. پروپوزال به درستی اشاره می‌کند که «تطابق سیستم با قوانین داخلی کشور... چالشی دیگر است چرا که بسیاری از ابزارها و فناوری‌ها در حوزه زنجیره بلوکی با قوانین فعلی ایران سازگاری ندارد»[؟].

۱-۱۰-۱ ابهام در ماهیت حقوقی توکن‌ها

اولین سؤال قانونی این است که توکن دیجیتالی که نماینده یک کالای فیزیکی است، از نظر حقوقی چه ماهیتی دارد؟ آیا یک «دارایی دیجیتال» صرف است؟ آیا می‌تواند به عنوان یک «سند بهادار» ($Security$) تلقی شود؟ پاسخ به این سؤال، تأثیر مستقیمی بر قوانین حاکم بر صدور، انتقال و مالیات‌ستانی از آن خواهد داشت. فقدان یک تعریف قانونی روشن، می‌تواند ریسک حقوقی برای کسب‌وکارهایی که از این سیستم استفاده می‌کنند، به همراه داشته باشد.

۱-۱۱ قوانین مربوط به ارزهای دیجیتال و پرداخت

اگرچه در این سیستم، پرداخت هزینه کالاها می‌تواند خارج از زنجیره انجام شود، اما خود تراکنش‌های زنجیره بلوکی نیازمند پرداخت هزینه گس با استفاده از ارز دیجیتال (مانند اتر) است. قوانین مربوط به نگهداری و استفاده از ارزهای دیجیتال در کشور، همچنان دارای ابهاماتی است که باید در مدل تجاری نهایی پروژه در نظر گرفته شود.

۱-۱۱-۱ حریم خصوصی و حفاظت از داده‌ها

یکی از ویژگی‌های زنجیره بلوکی عمومی، شفافیت آن است. در حالی که این شفافیت برای ردیابی و اعتبارسنجی فوق‌العاده است، می‌تواند چالش‌هایی را برای حریم خصوصی و محرمانگی اطلاعات تجاری ایجاد کند. اطلاعاتی مانند حجم معاملات بین یک تولیدکننده و توزیع‌کننده، یا مسیرهای دقیق توزیع، می‌تواند برای رقبا بسیار ارزشمند باشد. طراحی سیستمی که بتواند بین نیاز به شفافیت برای حسابرسی و نیاز به محرمانگی برای حفظ مزیت رقابتی تعادل برقرار کند، یک چالش مهم است. راهکارهایی مانند استفاده از زنجیره‌های بلوکی خصوصی ($PrivateBlockchains$) یا فناوری‌های اثبات با دانش صفر ($Zero-KnowledgeProofs$) می‌توانند در آینده برای حل این مشکل مورد بررسی قرار گیرند.

۱-۱۱-۲ مسئولیت‌پذیری در یک محیط غیرمتمرکز

در صورت بروز خطا در یک قرارداد هوشمند که منجر به خسارت مالی شود، چه کسی از نظر قانونی مسئول است؟ توسعه‌دهنده قرارداد؟ کاربرانی که با آن تعامل کرده‌اند؟ یا کل شبکه؟ قوانین سنتی که بر پایه نهادهای متمرکز بنا شده‌اند، پاسخ روشنی برای این سؤالات در یک محیط غیرمتمرکز ندارند. این ابهام، یکی دیگر از ریسک‌های حقوقی است که کسب‌وکارها در هنگام پذیرش این فناوری با آن روبرو هستند.

رویکرد پروژه در جهت انطباق‌پذیری

پروژه حاضر، با درک این چالش‌ها، یک گام هوشمندانه در جهت افزایش انطباق‌پذیری برداشته است. تعبیه قابلیت **محاسبه خودکار مالیات** در قرارداد هوشمند[؟، ؟]، نشان‌دهنده یک رویکرد پیشگیرانه برای همسوسازی سیستم با الزامات مالیاتی کشور است. این قابلیت، به نهادهای نظارتی نشان می‌دهد که این فناوری نه تنها برای فرار از قوانین طراحی نشده، بلکه می‌تواند ابزاری بسیار کارآمد برای افزایش شفافیت مالیاتی و تسهیل فرآیندهای نظارتی باشد. این رویکرد می‌تواند به عنوان یک نقطه قوت در گفتگو با نهادهای قانون‌گذار و جلب اعتماد آن‌ها مورد استفاده قرار گیرد.

فصل دوم

مرور پژوهش‌های پیشین و سامانه‌های مشابه

هدف اصلی این فصل، قرار دادن پژوهش حاضر در بستر علمی و صنعتی موجود است. برای درک عمیق نوآوری‌ها و وجه تمایز این پروژه، ضروری است که ابتدا راهکارهای پیشین و وضعیت فعلی فناوری در حوزه مدیریت زنجیره تأمین را به دقت مورد بررسی و نقد قرار دهیم. این فصل به دو بخش اصلی تقسیم می‌شود. در بخش اول، که در ادامه به تفصیل به آن پرداخته می‌شود، به تحلیل عمیق سامانه‌های مدیریت زنجیره تأمین سنتی و همچنین نسل اول راهکارهای دیجیتال غیربلاکچینی می‌پردازیم. این تحلیل نشان خواهد داد که چرا این راهکارها، با وجود تمام پیشرفت‌ها، در حل مشکلات بنیادین مربوط به «اعتماد» و «شفافیت» ناکام مانده‌اند. در بخش دوم، به صورت متمرکز به بررسی و تحلیل پروژه‌هایی خواهیم پرداخت که از فناوری زنجیره بلوکی در حوزه زنجیره تأمین بهره برده‌اند تا با مقایسه آن‌ها، جایگاه و نوآوری پروژه حاضر به روشنی مشخص گردد.

۲-۱ تحلیل سامانه‌های سنتی و راهکارهای دیجیتال غیربلاکچینی

پارادایم مدیریت زنجیره تأمین ($SupplyChainManagement - SCM$) در طول دهه‌های گذشته، تحولات بسیاری را تجربه کرده است. هدف اصلی در نگاه سنتی، همواره بر «بهینه‌سازی» و «کارایی» متمرکز بوده است. شرکت‌ها تلاش کرده‌اند تا با استفاده از سیستم‌های اطلاعاتی و مدل‌های ریاضی، هزینه‌های موجودی را کاهش دهند، زمان تحویل را به حداقل برسانند و فرآیندهای لجستیکی خود را بهینه‌تر کنند [۶]. با این حال، این تمرکز بر بهینه‌سازی داخلی، اغلب به قیمت نادیده گرفتن اهمیت جریان شفاف اطلاعات بین شرکای تجاری تمام شده است. در این بخش، ابتدا معماری سیستم‌های اطلاعاتی متمرکزی که ستون فقرات زنجیره‌های تأمین امروزی را تشکیل می‌ده□□، بررسی کرده و سپس به تحلیل نسل اول فناوری‌های دیجیتال که برای رفع برخی از این کاستی‌ها به کار گرفته شدند، می‌پردازیم.

۲-۱-۱ معماری سیستم‌های اطلاعاتی متمرکز در زنجیره تأمین

زنجیره تأمین مدرن، بدون سیستم‌های اطلاعاتی پیچیده قابل تصور نیست. این سیستم‌ها وظیفه مدیریت جریان عظیم اطلاعات، از ثبت سفارش یک مشتری تا برنامه‌ریزی تولید و ارسال نهایی کالا را بر عهده دارند. با این حال، معماری غالب این سیستم‌ها، یک معماری «متمرکز» و «درون-سازمانی» است که خود ریشه بسیاری از مشکلات امروزی است.

ی برنامه‌ریزی منابع سازمانی (ERP) سیستم‌های برنامه‌ریزی منابع سازمانی یا ERP ($EnterpriseResourcePlanning$)، به عنوان سیستم عصبی مرکزی اکثر شرکت‌های بزرگ و متوسط عمل می‌کنند. پلتفرم‌هایی مانند SAP، $Oracle$ و $MicrosoftDynamics$، مجموعه‌ای یکپارچه از ماژول‌های نرم‌افزاری را برای مدیریت تمام

جنبه‌های یک کسب‌وکار، از منابع انسانی و مالی گرفته تا تولید و فروش، فراهم می‌آورند [؟]. ماژول‌های مرتبط با زنجیره تأمین در یک سیستم ERP معمولاً شامل موارد زیر است:

- **مدیریت موجودی** ($InventoryManagement$): ردیابی سطح موجودی مواد اولیه، کالاهای در حال ساخت و محصولات نهایی در انبارها.
- **پردازش سفارش** ($OrderProcessing$): مدیریت چرخه کامل یک سفارش از زمان ثبت توسط مشتری تا تحویل نهایی.
- **مدیریت تدارکات** ($Procurement$): خودکارسازی فرآیندهای مربوط به خرید مواد اولیه از تأمین‌کنندگان.
- **برنامه‌ریزی تولید** ($ProductionPlanning$): برنامه‌ریزی و زمان‌بندی فرآیندهای تولید بر اساس پیش‌بینی تقاضا و سطح موجودی.

بزرگ‌ترین مزیت یک سیستم ERP، ایجاد یک «منبع حقیقت واحد» ($SingleSourceofTruth$) **درون مرزهای یک سازمان** است. تمام بخش‌های یک شرکت به داده‌های یکسان و به‌روزی دسترسی دارند که این امر هماهنگی داخلی را به شدت افزایش می‌دهد. با این حال، همین نقطه قوت، بزرگ‌ترین نقطه ضعف آن در مقیاس یک زنجیره تأمین است. یک سیستم ERP اساساً برای دنیای داخل یک شرکت طراحی شده و به صورت پیش‌فرض، دیدی نسبت به فرآیندهای تأمین‌کنندگانِ تأمین‌کنندگان یا مشتریانِ مشتریان خود ندارد.

سیستم‌های مدیریت زنجیره تأمین (SCM) و تبادل الکترونیکی داده (EDI)

برای حل مشکل ارتباط بین ERPهای شرکت‌های مختلف، سیستم‌های تخصصی‌تری به نام سیستم‌های مدیریت زنجیره تأمین ($SCMSystems$) توسعه یافتند. این سیستم‌ها تلاش می‌کنند تا پلی بین سیستم‌های اطلاعاتی شرکای تجاری مختلف ایجاد کنند. یکی از قدیمی‌ترین و رایج‌ترین فناوری‌ها برای این منظور، تبادل الکترونیکی داده یا EDI ($ElectronicDataInterchange$) است [؟]. EDI به شرکت‌ها اجازه می‌دهد تا اسناد تجاری استاندارد (مانند سفارش‌های خرید، فاکتورها و بارنامه‌ها) را به صورت الکترونیکی و با فرمت مشخصی برای یکدیگر ارسال کنند.

با این حال، EDI نیز دارای محدودیت‌های جدی است:

- **هزینه و پیچیدگی بالا:** راه‌اندازی و نگهداری سیستم‌های EDI پرهزینه و پیچیده است و معمولاً تنها برای شرکت‌های بسیار بزرگ که با تعداد محدودی از شرکای اصلی و بلندمدت کار می‌کنند، مقرون‌به‌صرفه است.
- **عدم کار در زمان واقعی** ($Real-time$): تبادل داده در EDI معمولاً به صورت دسته‌ای ($Batch$) و در فواصل زمانی مشخص (مثلاً در پایان هر روز کاری) انجام می‌شود. این تأخیر در جریان اطلاعات، مانع از تصمیم‌گیری‌های سریع و واکنش به موقع به تغییرات بازار می‌شود.

- **ساختار غیرقابل انعطاف:** فرمت‌های EDI بسیار سختگیرانه و استاندارد شده هستند و تغییر یا افزودن اطلاعات جدید به آن‌ها دشوار است.

مشکل بنیادین: سیلوهای اطلاعاتی و اثر شلاقی

نتیجه نهایی معماری متمرکز و سیستم‌های ارتباطی ناکارآمد، پدیده‌ای است که از آن به عنوان «سیلوهای اطلاعاتی» ($InformationSilos$) یاد می‌شود. در این پدیده، هر شرکت در زنجیره تأمین (تولیدکننده، توزیع‌کننده، عمده‌فروش، خرده‌فروش) داده‌های خود را در یک پایگاه داده مجزا و ایزوله نگهداری می‌کند. جریان اطلاعات بین این سیلوها، کند، غیرقابل اعتماد و اغلب نیازمند ورود دستی داده است که خود منشأ بسیاری از خطاهاست.

یکی از مشهورترین و مخرب‌ترین پیامدهای سیلوهای اطلاعاتی، **اثر شلاقی** ($TheBullwhipEffect$) است [۷]. این پدیده توصیف می‌کند که چگونه نوسانات کوچک در تقاضای مشتری نهایی (در سطح خرده‌فروشی)، به صورت فزاینده‌ای در حین حرکت به سمت بالای زنجیره تأمین (به سمت تولیدکننده) تقویت می‌شود. برای مثال، یک افزایش ۱۰ درصدی در تقاضای مشتری، ممکن است باعث شود خرده‌فروش سفارش خود به عمده‌فروش را ۲۰ درصد افزایش دهد تا یک موجودی اطمینان برای خود ایجاد کند. عمده‌فروش نیز با مشاهده این افزایش، سفارش خود به تولیدکننده را ۴۰ درصد افزایش می‌دهد و این روند ادامه می‌یابد.

این تقویت نوسانات، ناشی از عدم قطعیت و فقدان دیدپذیری است. هر عضو زنجیره، تنها سفارش دریافتی از عضو پایین‌دستی خود را می‌بیند و دیدی نسبت به تقاضای واقعی مصرف‌کننده نهایی ندارد. این امر منجر به مشکلات زیر می‌شود:

- **موجودی مازاد و هزینه‌های نگهداری بالا:** تولیدکنندگان بر اساس سیگنال‌های تقاضای اغراق‌آمیز، بیش از حد تولید می‌کنند که منجر به انباشت موجودی در انبارها می‌شود.
- **کمبود موجودی:** در جهت معکوس، یک کاهش تقاضای موقتی نیز می‌تواند به صورت اغراق‌آمیز به بالا منتقل شده و باعث شود تولیدکننده تولید خود را بیش از حد کاهش دهد که منجر به کمبود کالا در زمان افزایش مجدد تقاضا می‌شود.
- **استفاده ناکارآمد از ظرفیت تولید و حمل‌ونقل:** نوسانات شدید در سفارش‌ها، برنامه‌ریزی پایدار برای تولید و لجستیک را غیرممکن می‌سازد.

اثر شلاقی، نمونه بارزی از این است که چگونه معماری اطلاعاتی یک زنجیره تأمین، تأثیر مستقیمی بر عملکرد مالی و عملیاتی آن دارد. این مشکل، یک مشکل محاسباتی یا لجستیکی صرف نیست، بلکه یک «مشکل اطلاعاتی» است که ریشه در عدم شفافیت و عدم اشتراک‌گذاری داده‌ها در زمان واقعی دارد.

۲-۱-۲ نسل اول دیجیتالی‌سازی: فناوری‌های ردیابی و شناسایی

در پاسخ به مشکلات دیدپذیری، نسل اول فناوری‌های دیجیتال با هدف بهبود فرآیندهای «شناسایی» و «ردیابی» کالاهای فیزیکی پدید آمدند. این فناوری‌ها تلاش کردند تا پلی بین دنیای فیزیکی محصولات و دنیای دیجیتال اطلاعات ایجاد کنند. با این حال، همانطور که خواهیم دید، این راهکارها نیز در نهایت به همان دیوارهای بلند سیلوهای اطلاعاتی برخورد کردند.

بارکدها و کدهای QR

بارکدها، به عنوان یک فناوری بسیار ارزان و فراگیر، انقلابی در مدیریت فروش و موجودی در سطح خرده‌فروشی ایجاد کردند. آن‌ها امکان شناسایی سریع و خودکار یک محصول را در پایانه فروش فراهم آوردند. کدهای QR ($QuickResponse$) نیز به عنوان نسل بعدی بارکدهای دوبعدی، قابلیت ذخیره‌سازی اطلاعات بیشتر (مانند یک آدرس وب) را فراهم کرده و استفاده از آن‌ها با دوربین تلفن‌های هوشمند را ممکن ساختند.

با این حال، این فناوری‌ها دارای محدودیت‌های ذاتی هستند:

- **ماهیت ایستا و محدودیت داده:** یک بارکد یا کد QR معمولی، تنها یک شناسه ثابت را در خود جای داده است و اطلاعات آن به صورت پویا به‌روز نمی‌شود.
- **آسیب‌پذیری در برابر جعل:** کپی کردن و چاپ مجدد یک بارکد یا کد QR بسیار ساده است. این امر آن‌ها را به ابزاری غیرقابل اعتماد برای کاربردهای ضدجعل تبدیل می‌کند.
- **نیاز به اسکن دستی:** هر آیتم باید به صورت جداگانه و با قرار گرفتن در خط دید اسکنر، خوانده شود که این امر در مقیاس‌های بزرگ (مانند ورودی یک انبار) ناکارآمد است.

تفاوت کلیدی کد QR در پروژه حاضر با یک کد QR معمولی در این است که کد QR ما به یک «شناسه ثابت» اشاره نمی‌کند، بلکه به یک «لینک پویا» به یک پایگاه داده امن و تغییرناپذیر (یعنی زنجیره بلوکی) اشاره دارد. این بک‌اند ($Backend$) امن است که به آن معنا و اعتبار می‌بخشد، نه خود کد QR.

۲-۱-۳ شناسایی با فرکانس رادیویی ($RFID$)

فناوری $RFID$ گام بزرگی رو به جلو برای غلبه بر محدودیت‌های بارکد بود. یک سیستم $RFID$ از دو جزء اصلی تشکیل شده است: یک **تگ** (Tag) که به محصول متصل می‌شود و حاوی یک شناسه منحصربه‌فرد است، و یک **ریدر** ($Reader$) که با ارسال امواج رادیویی، می‌تواند اطلاعات تگ‌ها را از راه دور و بدون نیاز به خط دید مستقیم بخواند [۸].

مزایای $RFID$ قابل توجه بود:

- **اسکن دسته‌ای و سریع:** یک ریدر $RFID$ می‌تواند صدها تگ را در چند ثانیه شناسایی کند، که این امر فرآیندهایی مانند شمارش موجودی یا ثبت ورود و خروج کالا از انبار را به شدت تسریع می‌کند.
- **عدم نیاز به خط دید:** تگ‌ها نیازی به دیده شدن توسط ریدر ندارند و می‌توانند در داخل بسته‌بندی یا کارتن قرار داشته باشند.
- **قابلیت ذخیره داده بیشتر:** برخی از تگ‌های $RFID$ قابلیت نوشتن و بازنویسی داده را نیز دارند.

شرکت‌های بزرگی مانند والمارت ($Walmart$) در اوایل دهه ۲۰۰۰، سرمایه‌گذاری عظیمی بر روی این فناوری انجام دادند و تأمین‌کنندگان خود را ملزم به استفاده از تگ‌های $RFID$ بر روی پالت‌ها و کارتن‌ها کردند. هدف، افزایش دیدپذیری در زنجیره تأمین و کاهش هزینه‌های ناشی از خطای انسانی بود. با وجود موفقیت‌های اولیه، پروژه‌های $RFID$ نیز با چالش‌هایی روبرو شدند، از جمله هزینه بالای تگ‌ها (در مقایسه با بارکد) و مشکلات مربوط به تداخل امواج رادیویی.

اما مهم‌ترین محدودیت $RFID$، که اغلب نادیده گرفته می‌شود، این است که این فناوری نیز تنها «ورودی» داده را بهبود می‌بخشد. داده‌های جمع‌آوری شده توسط ریدرهای $RFID$، در نهایت به همان پایگاه‌های داده متمرکز و ایزوله شرکت مربوطه ارسال می‌شدند. به عبارت دیگر، $RFID$ مشکل «جمع‌آوری سریع داده» را حل کرد، اما مشکل «اشتراک‌گذاری امن و قابل اعتماد داده» بین شرکای مختلف را دست‌نخورده باقی گذاشت. داده‌های $RFID$ نیز مانند هر داده دیگری در یک سرور متمرکز، قابل حذف یا دستکاری بودند.

اینترنت اشیاء (IoT) و سنسورهای هوشمند

اینترنت اشیاء یا IoT، تکامل طبیعی فناوری $RFID$ است. در اینجا، به جای یک تگ غیرفعال، با «سنسورهای هوشمند» سروکار داریم که می‌توانند به صورت فعال، داده‌های محیطی را جمع‌آوری کرده و از طریق اینترنت ارسال کنند [۹]. این سنسورها می‌توانند پارامترهای مختلفی را اندازه‌گیری کنند:

- **سنسورهای دما و رطوبت:** برای نظارت بر زنجیره سرد ($ColdChain$) در حمل‌ونقل مواد غذایی، داروها و مواد شیمیایی حساس.
- **سنسورهای موقعیت‌یاب (GPS):** برای ردیابی دقیق و لحظه‌ای مکان محموله‌ها.
- **شتاب‌سنج‌ها:** برای تشخیص ضربه یا سقوط که می‌تواند به کالاهای حساس آسیب برساند.
- **سنسورهای باز شدن درب کانتینر:** برای افزایش امنیت و جلوگیری از سرقت.

ترکیب این سنسورها، یک جریان داده بسیار غنی و در زمان واقعی ($Real-time$) از وضعیت و شرایط یک محصول در طول زنجیره تأمین فراهم می‌کند. این سطح از دیدپذیری، در مدیریت کیفیت و امنیت، بی‌سابقه است. اما بار دیگر، همان مشکل بنیادین پدیدار می‌شود: **این داده‌ها به کجا می‌روند؟**

داده‌های ارزشمند جمع‌آوری شده توسط سنسورهای IoT، معمولاً به یک پلتفرم ابری ($CloudPlatform$) متمرکز که توسط ارائه‌دهنده سرویس IoT یا خود شرکت کنترل می‌شود، ارسال می‌گردد. این ساختار، تمام مشکلات یک سیستم متمرکز را به ارث می‌برد:

- **مالکیت و کنترل داده:** داده‌ها در انحصار یک شرکت باقی می‌مانند. شرکت حمل‌ونقل ممکن است به دلایل مختلف، از به اشتراک گذاشتن داده‌های کامل سنسور دما با صاحب کالا یا شرکت بیمه خودداری کند.
- **قابلیت دستکاری:** هیچ تضمین رمزنگاری‌شده‌ای وجود ندارد که داده‌های ثبت‌شده در پلتفرم ابری، پس از ثبت تغییر نکرده باشند.
- **عدم وجود یک تاریخچه یکپارچه:** صاحب کالا ممکن است به داده‌های سنسور شرکت حمل‌ونقل دسترسی داشته باشد، اما وقتی کالا به شرکت حمل‌ونقل منتقل می‌شود، این دیدپذیری را از دست بدهد.

۲-۱-۴ جمع‌بندی: چرا راهکارهای سنتی و دیجیتال اولیه کافی نیستند؟

تحلیل ارائه شده در این بخش نشان می‌دهد که با وجود دهه‌ها تلاش برای بهینه‌سازی و دیجیتالی‌سازی، زنجیره‌های تأمین همچنان با یک مشکل اساسی و حل‌نشده دست و پنجه نرم می‌کنند. این مشکل، یک مشکل فنی یا لجستیکی صرف نیست، بلکه یک **مشکل اعتماد** است. جدول زیر، خلاصه‌ای از محدودیت‌های راهکارهای بررسی شده را در برابر معیارهای کلیدی یک زنجیره تأمین ایده‌آل نشان می‌دهد.

جدول ۲-۱: مقایسه محدودیت‌های راهکارهای مختلف

راهکار	شفافیت سرتاسری	امنیت/تغییرناپذیری	ایجاد اعتماد	عدم تمرکز
SCM / ERP سنتی	بسیار ضعیف	ضعیف	ضعیف	خیر
بارکد / QR کد	ضعیف	بسیار ضعیف	بسیار ضعیف	خیر
$RFID$	متوسط (درون‌سازمانی)	ضعیف	ضعیف	خیر
IoT / سنسورها	متوسط (بسته به پلتفرم)	ضعیف	ضعیف	خیر

همانطور که در جدول ۲-۱ مشاهده می‌شود، هیچ‌یک از این راهکارها قادر به ارائه ترکیبی از شفافیت، امنیت و عدم تمرکز به صورت همزمان نیستند. مشکل اصلی این است که تمام این فناوری‌ها، در نهایت داده‌های خود را به یک «مخزن متمرکز و قابل اعتماد» فرضی ارسال می‌کنند، در حالی که در یک زنجیره تأمین که از ده‌ها شرکت مستقل تشکیل شده، چنین مخزن واحد و مورد اعتمادی **وجود خارجی ندارد**.

هر شرکت به داده‌های سیستم خود اعتماد دارد، اما دلیلی ندارد که به داده‌های ارسال‌شده از سوی شرکای تجاری خود (که ممکن است در فرمت‌های مختلف و با تأخیر ارسال شوند) اعتماد کامل داشته باشد. این عدم اعتماد متقابل، منجر به ایجاد فرآیندهای پرهزینه «تطبیق» ($Reconciliation$) می‌شود. شرکت‌ها تیم‌هایی را استخدام می‌کنند تا فاکتورها، بارنامه‌ها و رسیدهای خود را با اسناد ارسال‌شده از سوی شرکایشان مقایسه و مغایرت‌ها را برطرف کنند. این فرآیندها، منشأ اصلی ناکارآمدی، اتلاف وقت و اختلافات تجاری هستند.

در نهایت، این تحلیل ما را به یک نتیجه‌گیری اساسی می‌رساند: زنجیره تأمین مدرن، بیش از یک سیستم نرم‌افزاری جدید یا یک سنسور هوشمندتر، به یک «**لایه اعتماد**» ($Layer\ Trust$) مشترک، بی‌طرف و غیرمتمرکز نیاز دارد. زیرساختی که تمام شرکت‌کنندگان بتوانند داده‌های خود را با اطمینان بر روی آن ثبت کرده و به صحت داده‌های ثبت‌شده توسط دیگران نیز اعتماد کامل داشته باشند، زیرا این زیرساخت توسط هیچ نهاد واحدی کنترل نمی‌شود و قوانین آن توسط ریاضیات و رمزنگاری تضمین شده است.

این نیاز بنیادین به یک لایه اعتماد غیرمتمرکز، دقیقاً همان مسئله‌ای است که فناوری زنجیره بلوکی برای حل آن پدید آمده است. زنجیره بلوکی، با ارائه یک دفتر کل توزیع‌شده، شفاف و تغییرناپذیر، این پتانسیل را دارد که آن لایه اعتماد گمشده را فراهم کرده و پارادایم حاکم بر مدیریت زنجیره تأمین را به کلی دگرگون سازد. مبانی و جزئیات این راهکار نوین، موضوع اصلی بخش بعدی این فصل خواهد بود.

۲-۲ بررسی پروژه‌های زنجیره تأمین مبتنی بر زنجیره بلوکی

در بخش پیشین، به تفصیل نشان داده شد که چرا سامانه‌های سنتی و نسل اول راهکارهای دیجیتال، در حل چالش‌های بنیادین اعتماد و شفافیت در زنجیره تأمین ناکام بوده‌اند. مشخص شد که مشکل اصلی، نه کمبود داده، بلکه فقدان یک «لایه اعتماد» مشترک و غیرمتمرکز برای اعتبارسنجی و اشتراک‌گذاری امن داده‌ها بین شرکای تجاری ناهمگون است. این تحلیل، زمینه را برای ورود پارادایم نوین زنجیره بلوکی به این حوزه فراهم می‌کند. فناوری زنجیره بلوکی، با ارائه یک دفتر کل توزیع‌شده، شفاف و تغییرناپذیر، دقیقاً همان لایه اعتماد گمشده را ارائه می‌دهد.

در این بخش، به صورت عمیق به بررسی و تحلیل پروژه‌ها، پلتفرم‌ها و استانداردهایی می‌پردازیم که تلاش کرده‌اند از این پتانسیل عظیم برای متحول ساختن زنجیره تأمین بهره ببرند. این بررسی یک مسیر تکاملی را دنبال می‌کند: از نسل اول راهکارها که بر پلتفرم‌های خصوصی و افزایش شفافیت متمرکز بودند، تا نسل دوم که با بهره‌گیری از شبکه‌های عمومی و مفهوم «توکنیزه‌سازی»، قابلیت‌های جدیدی را به این عرصه افزودند. هدف نهایی این بررسی، شناسایی دقیق نقاط قوت و ضعف رویکردهای مختلف و در نهایت، مشخص کردن جایگاه نوآورانه و منحصربه‌فرد پروژه حاضر در این چشم‌انداز گسترده است.

۲-۲-۱ نسل اول راهکارها: تمرکز بر شفافیت و پلتفرم‌های خصوصی

در سال‌های اولیه معرفی زنجیره بلوکی به دنیای کسب‌وکار (تقریباً بین سال‌های ۲۰۱۴ تا ۲۰۱۸)، هیجان زیادی پیرامون این فناوری وجود داشت. بسیاری آن را به عنوان یک «گلوله نقره‌ای» (*bullet silver*) برای حل تمام مشکلات زنجیره تأمین می‌دیدند. با این حال، استفاده از شبکه‌های زنجیره بلوکی عمومی و بدون نیاز به مجوز (*Permissionless*) مانند بیت‌کوین یا اتریوم برای کاربردهای سازمانی، با موانع جدی روبرو بود:

- **مقیاس‌پذیری و هزینه:** این شبکه‌ها دارای توان پردازشی پایین و هزینه تراکنش (گس) بالا و غیرقابل پیش‌بینی بودند.
- **حریم خصوصی:** تمام داده‌های ثبت‌شده بر روی یک زنجیره بلوکی عمومی، برای همه افراد در سراسر جهان قابل مشاهده است. این سطح از شفافیت برای اطلاعات حساس تجاری (مانند قیمت‌گذاری، حجم معاملات و هویت شرکا) غیرقابل قبول بود.
- **حاکمیت و کنترل:** در یک شبکه عمومی، هیچ نهاد مرکزی برای مدیریت شبکه، حل اختلافات یا کنترل دسترسی شرکت‌کنندگان وجود ندارد. این عدم کنترل برای محیط‌های تجاری که نیازمند قوانین و مقررات مشخص هستند، یک نقطه ضعف بزرگ محسوب می‌شد.

این چالش‌ها منجر به ظهور دسته‌ای جدید از پلتفرم‌های زنجیره بلوکی شد که به طور خاص برای نیازهای سازمانی طراحی شده بودند: زنجیره‌های بلوکی خصوصی یا کنسرسیومی (*Consortium/Private Blockchains*). در این مدل، به جای یک شبکه باز، یک شبکه بسته و نیازمند مجوز (*Permissioned*) ایجاد می‌شود که تنها اعضای تأییدشده (مانند چند شرکت در یک زنجیره تأمین) می‌توانند در آن مشارکت کنند. این رویکرد، ضمن حفظ برخی از مزایای زنجیره بلوکی (مانند تغییرناپذیری و دفتر کل مشترک)، مشکلات مربوط به حریم خصوصی و حاکمیت را برطرف می‌کرد. برجسته‌ترین و تأثیرگذارترین پلتفرم در این نسل از راهکارها، بدون شک *Fabric Hyperledger* است.

معرفی و تحلیل عمیق پلتفرم *Fabric Hyperledger*

Hyperledger یک پروژه چتر (*project umbrella*) متن‌باز است که در سال ۲۰۱۵ توسط بنیاد لینوکس (*Foundation Linux*) با هدف ترویج و توسعه فناوری‌های زنجیره بلوکی برای کاربردهای سازمانی آغاز شد. این پروژه شامل چندین فریم‌ورک و ابزار مختلف است که مشهورترین آن‌ها، *Hyperledger Fabric* است. *Fabric* که توسعه آن توسط شرکت *IBM* رهبری می‌شد، با یک معماری کاملاً ماژولار و متفاوت از اتریوم طراحی شد تا به نیازهای خاص کسب‌وکارها پاسخ دهد.

معماری منحصربه‌فرد *Fabric Hyperledger*: برخلاف معماری یکپارچه اتریوم، *Fabric* از یک رویکرد ماژولار بهره می‌برد که در آن وظایف مختلف شبکه (مانند اجرای تراکنش، اجماع و به‌روزرسانی

دفتر کل) بین مؤلفه‌های مختلف تقسیم شده است [۱۰]. این معماری به انعطاف‌پذیری و مقیاس‌پذیری بیشتر کمک می‌کند. مؤلفه‌های کلیدی آن عبارتند از:

- **همتاها** ($Peers$): گره‌هایی در شبکه هستند که میزبان دفتر کل ($Ledger$) و قراردادهای هوشمند (که در $Fabric$ به آن $Chaincode$ گفته می‌شود) هستند. همتاها تراکنش‌ها را اجرا و اعتبارسنجی می‌کنند.

- **سرویس ترتیب‌دهی** ($Service\ Ordering$): این مؤلفه مسئول ایجاد اجماع بر روی ترتیب تراکنش‌ها و بسته‌بندی آن‌ها در بلوک‌های جدید است. $Fabric$ از الگوریتم‌های اجماع مختلفی مانند $Solo$ (برای توسعه) و $Raft$ یا $Kafka$ (برای تولید) پشتیبانی می‌کند که برخلاف اثبات کار ($Proof-of-Work$) در اتریوم، نیازی به ماینینگ پرمصرف ندارند.

- **کانال‌ها** ($Channels$): این یکی از نوآورانه‌ترین ویژگی‌های $Fabric$ است. کانال یک مکانیزم ارتباطی خصوصی بین زیرمجموعه‌ای از اعضای شبکه است. هر کانال، دفتر کل مخصوص به خود را دارد و تراکنش‌های انجام‌شده در یک کانال، تنها برای اعضای همان کانال قابل مشاهده است. این ویژگی، راهکاری قدرتمند برای حل مشکل حریم خصوصی داده‌ها ارائه می‌دهد. برای مثال، در یک زنجیره تأمین، تولیدکننده و یک توزیع‌کننده خاص می‌توانند یک کانال خصوصی برای ثبت معاملات و قیمت‌گذاری‌های محرمانه خود داشته باشند، در حالی که سایر اعضای شبکه از آن بی‌اطلاع هستند.

- **قرارداد هوشمند** ($Chaincode$): منطق کسب‌وکار در $Fabric$ در قالب $Chaincode$ نوشته می‌شود. برخلاف اتریوم که تنها از زبان $Solidity$ پشتیبانی می‌کند، $Chaincode$ را می‌توان با زبان‌های برنامه‌نویسی عمومی مانند Go، $Node.js$ و $Java$ نوشت که این امر توسعه را برای برنامه‌نویسان وب آسان‌تر می‌کند.

- **سیاست‌های تأیید** ($Policies\ Endorsement$): برای هر $Chaincode$ می‌توان یک سیاست تأیید تعریف کرد که مشخص می‌کند یک تراکنش برای معتبر بودن، باید توسط کدام یک از همتاهای شبکه تأیید (امضا) شود. برای مثال، می‌توان سیاستی تعریف کرد که طبق آن، یک تراکنش انتقال مالکیت باید هم توسط فروشنده و هم توسط خریدار تأیید شود.

مزایا و معایب $Fabric\ Hyperledger$ در زنجیره تأمین:

این معماری منحصربه‌فرد، مزایای قابل توجهی برای کاربردهای زنجیره تأمین به همراه داشت:

- **محرمانگی داده‌ها:** قابلیت ایجاد کانال‌های خصوصی، بزرگ‌ترین مزیت $Fabric$ بود که به شرکت‌ها اجازه می‌داد تا داده‌های حساس خود را تنها با شرکای مورد نظر به اشتراک بگذارند.

- **توان پردازشی بالا:** به دلیل استفاده از الگوریتم‌های اجماع سبک‌تر و عدم نیاز به مشارکت تمام گره‌ها در تمام تراکنش‌ها، $Fabric$ می‌تواند به توان پردازشی بسیار بالاتری (هزاران تراکنش در ثانیه) نسبت به شبکه‌های عمومی دست یابد.
- **عدم وجود هزینه گس:** در $Fabric$، هزینه تراکنش به صورت مستقیم (مانند گس در اتریوم) وجود ندارد. هزینه‌ها بیشتر مربوط به زیرساخت‌های محاسباتی برای اجرای گره‌ها و مدیریت شبکه است.
- **شبکه نیازمند مجوز:** امکان کنترل دقیق اینکه چه کسی می‌تواند به شبکه بپیوندد و چه مجوزهایی داشته باشد، برای محیط‌های تجاری که نیازمند حاکمیت مشخص هستند، یک مزیت کلیدی بود.

با این حال، این رویکرد با چالش‌ها و معایبی نیز همراه بود:

- **پیچیدگی در راه‌اندازی و مدیریت:** راه‌اندازی یک شبکه $Fabric$ با چندین سازمان، کانال و سیاست‌های مختلف، بسیار پیچیده‌تر از استقرار یک قرارداد هوشمند بر روی شبکه اتریوم است.
- **فقدان قابلیت همکاری با اکوسیستم عمومی:**.. : □□□□□□□)□□□□.(

مطالعات موردی برجسته با $Fabric\ Hyperledger$

برای درک بهتر تأثیر عملی این پلتفرم، دو مورد از بزرگ‌ترین و مشهورترین پروژه‌های زنجیره تأمین که بر پایه $Fabric$ ساخته شده‌اند را بررسی می‌کنیم.

مطالعه موردی اول: $Trust\ Food\ IBM$ صنعت مواد غذایی یکی از اولین و مستعدترین حوزه‌ها برای پذیرش فناوری زنجیره بلوکی بود. شیوع بیماری‌های ناشی از مواد غذایی آلوده و نیاز به فراخوان سریع محصولات از بازار، هزینه‌های هنگفتی را به شرکت‌ها تحمیل کرده و جان مصرف‌کنندگان را به خطر می‌انداخت. مشکل اصلی این بود که ردیابی منشأ یک محصول آلوده در یک زنجیره تأمین پیچیده، ممکن بود روزها یا حتی هفته‌ها طول بکشد.
IBM با همکاری غول‌های خرده‌فروشی مانند $Walmart$، پروژه $Trust\ Food$ را بر پایه $Hyperledger$ $Fabric$ راه‌اندازی کرد [۱۱]. هدف این پلتفرم، ایجاد یک دفتر کل مشترک و تغییرناپذیر برای ثبت تمام رویدادهای مربوط به یک محصول غذایی، از مزرعه تا قفسه فروشگاه، بود.

- **نحوه عملکرد:** هر شرکت‌کننده در زنجیره (کشاورز، فرآوری‌کننده، شرکت حمل‌ونقل، خرده‌فروش) یک گره در شبکه $Fabric$ اجرا می‌کند. اطلاعات مربوط به هر بچ از محصول (مانند تاریخ برداشت، گواهی‌های ارگانیک، اطلاعات حمل‌ونقل و...) به عنوان یک دارایی ($Asset$) در دفتر کل ثبت می‌شود. هر مرحله از انتقال، به عنوان یک تراکنش جدید ثبت شده و یک تاریخچه کامل و قابل ردیابی ایجاد می‌کند.

- **دستاورد اصلی:** بزرگ‌ترین دستاورد $Trust\ Food$، کاهش چشمگیر زمان ردیابی بود. در یکی از پایلوت‌های اولیه با $Walmart$، زمان لازم برای ردیابی منشأ یک بسته انبه از ۶ روز و ۱۸ ساعت به تنها ۲.۲ ثانیه کاهش یافت. این سرعت، امکان واکنش سریع در مواقع بحرانی و جلوگیری از توزیع گسترده محصولات آلوده را فراهم می‌کند.

- **چالش‌ها و درس‌آموخته‌ها:** با وجود موفقیت فنی، $Trust\ Food$ با چالش پذیرش نیز روبرو شد. متقاعد کردن هزاران کشاورز و تأمین‌کننده کوچک برای پیوستن به پلتفرم و ثبت دقیق داده‌ها، یک چالش بزرگ بود. همچنین، مدل کسب‌وکار مبتنی بر حق عضویت، برای بازیگران کوچک‌تر جذابیت کمتری داشت. این پروژه نشان داد که موفقیت یک راهکار زنجیره بلوکی، تنها به فناوری آن بستگی ندارد، بلکه به شدت به مدل کسب‌وکار، حاکمیت شبکه و ایجاد انگیزه برای تمام شرکت‌کنندگان وابسته است.

مطالعه موردی دوم: $TradeLens$ صنعت حمل‌ونقل کانتینری بین‌المللی، یکی از پیچیده‌ترین زنجیره‌های تأمین در جهان است. یک محموله ساده ممکن است در طول سفر خود توسط ۳۰ نهاد مختلف (از جمله گمرک، مقامات بندری، شرکت‌های حمل‌ونقل زمینی و دریایی) و با استفاده از بیش از ۲۰۰ تعامل و تبادل سند مختلف، جابجا شود. این فرآیند که عمدتاً مبتنی بر کاغذبازی و سیستم‌های ارتباطی قدیمی است، مملو از ناکارآمدی، تأخیر و ریسک خطا است.

برای حل این مشکل، دو غول این صنعت، شرکت کشتیرانی $Maersk$ و شرکت فناوری IBM، با یکدیگر همکاری کرده و پلتفرم $TradeLens$ را بر پایه $Fabric\ Hyperledger$ ایجاد کردند [؟]. هدف $TradeLens$، دیجیتالی کردن و ایجاد یک منبع حقیقت واحد برای تمام اسناد و رویدادهای مربوط به یک محموله کانتینری بود.

- **نحوه عملکرد:** پلتفرم به تمام طرف‌های درگیر اجازه می‌دهد تا به صورت آنی و امن، به اسناد حمل‌ونقل، اطلاعات گمرکی و وضعیت لحظه‌ای کانتینرها دسترسی داشته باشند. این دیدپذیری سرتاسری، هماهنگی بین نهادهای مختلف را به شدت بهبود بخشیده و نیاز به ارسال فیزیکی یا فکس اسناد را از بین می‌برد.

- **چالش اصلی: اثر شبکه‌ای** ($Effect\ Network$): بزرگ‌ترین چالش $TradeLens$، متقاعد کردن رقبای $Maersk$ (یعنی سایر خطوط کشتیرانی بزرگ) برای پیوستن به یک پلتفرم بود که توسط رقیب اصلی آن‌ها رهبری می‌شد. بسیاری از شرکت‌ها نگران بودند که $Maersk$ به داده‌های حساس آن‌ها دسترسی پیدا کند. اگرچه معماری $Fabric$ با استفاده از کانال‌ها می‌توانست این نگرانی را از نظر فنی برطرف کند، اما چالش اصلی، یک چالش «اعتماد تجاری» بود، نه یک مشکل فنی.

- **درس‌آموخته‌ها:** $TradeLens$ نشان داد که در یک کنسرسیوم، حاکمیت باید کاملاً بی‌طرف و غیرمتمرکز باشد. در نهایت، این پلتفرم با پیوستن سایر غول‌های کشتیرانی مانند $CGM\ CMA$

و MSC توانست بر این چالش غلبه کند، اما این فرآیند سال‌ها طول کشید. این مورد تأکید می‌کند که موفقیت یک شبکه کنسرسیومی، نیازمند یک مدل حاکمیتی قوی و مورد اعتماد همه اعضاست.

در مجموع، نسل اول راهکارها با استفاده از پلتفرم‌های خصوصی مانند $Fabric$، توانستند با موفقیت مشکل حریم خصوصی را حل کرده و کاربردهای عملی و تأثیرگذار زنجیره بلوکی را در مقیاس سازمانی به نمایش بگذارند. با این حال، پیچیدگی و ماهیت ایزوله این شبکه‌ها، زمینه را برای ظهور نسل دومی از راهکارها فراهم کرد که تلاش می‌کردند از قدرت و قابلیت همکاری اکوسیستم‌های عمومی بهره ببرند.

۲-۲-۲ نسل دوم راهکارها: استفاده از شبکه‌های عمومی و توکنیزه‌سازی

با بلوغ اکوسیستم زنجیره بلوکی، محدودیت‌های اولیه شبکه‌های عمومی تا حد زیادی برطرف یا کمرنگ شد. ظهور راهکارهای مقیاس‌پذیری لایه ۲ و زنجیره‌های جانبی سازگار با EVM، هزینه و سرعت تراکنش‌ها را به سطحی رساند که برای کاربردهای تجاری قابل قبول بود. این تحول، همراه با درک عمیق‌تر از مزایای شبکه‌های عمومی، منجر به یک تغییر پارادایم به سمت استفاده از این شبکه‌ها برای کاربردهای زنجیره تأمین شد.

مزایای کلیدی شبکه‌های عمومی عبارتند از:

- **عدم تمرکز واقعی:** امنیت شبکه توسط هزاران اعتبارسنج ناشناس در سراسر جهان تأمین می‌شود که این امر، ریسک تبانی یا کنترل توسط یک نهاد واحد را تقریباً به صفر می‌رساند.
- **قابلیت همکاری** ($Interoperability$): دارایی‌های ایجاد شده بر روی یک شبکه عمومی (مانند توکن‌های نماینده محصولات)، می‌توانند به راحتی با هزاران برنامه و پروتکل دیگر در همان اکوسیستم تعامل داشته باشند. برای مثال، می‌توان یک دارایی زنجیره تأمین را در یک پروتکل مالی غیرمتمرکز ($DeFi$) به عنوان وثیقه برای دریافت وام استفاده کرد.
- **دسترسی بدون نیاز به مجوز:** هر کسی می‌تواند بدون نیاز به کسب اجازه، یک قرارداد هوشمند را بر روی شبکه مستقر کرده و یک برنامه کاربردی ایجاد کند. این امر نوآوری را به شدت تسریع می‌کند.

مفهوم محوری که این نسل از راهکارها را به پیش می‌راند، «توکنیزه‌سازی دارایی‌ها» ($Tokenization\ Asset$) است.

توکنیزه‌سازی دارایی‌ها در زنجیره تأمین

توکنیزه‌سازی، فرآیند ایجاد یک نماینده دیجیتال (یک توکن) برای یک دارایی واقعی یا دیجیتال بر روی یک شبکه زنجیره بلوکی است. این توکن، مالکیت آن دارایی را نمایندگی می‌کند و می‌تواند بر اساس قوانین تعریف‌شده در یک قرارداد هوشمند، منتقل، معامله یا مدیریت شود.

توکنیزه‌سازی، دارایی‌های سنتی و غیرنقدشونده را به دارایی‌هایی «برنامه‌پذیر» تبدیل می‌کند. وقتی یک کالای فیزیکی در زنجیره تأمین به یک توکن دیجیتال تبدیل می‌شود، مزایای زیر حاصل می‌گردد:

- **مالکیت شفاف و قابل تأیید:** مالکیت توکن به صورت شفاف بر روی زنجیره بلوکی ثبت شده و هر کسی می‌تواند با اطمینان، مالک فعلی آن را شناسایی کند.
- **انتقال آنی و همتا به همتا:** انتقال مالکیت، به سادگی انتقال توکن از یک کیف پول دیجیتال به دیگری است. این فرآیند در چند ثانیه و بدون نیاز به هیچ واسطه‌ای انجام می‌شود.
- **قابلیت تقسیم‌پذیری** ($Fractionalization$)**:** می‌توان مالکیت یک دارایی گران‌قیمت (مانند یک محموله بزرگ) را به چندین توکن کوچک‌تر تقسیم کرد و به چندین نفر فروخت.

- . : برای پیاده‌سازی توکنیزه‌سازی، مجموعه‌ای از استانداردهای فنی توسعه یافته‌اند که اطمینان می‌دهند توکن‌های ایجاد شده توسط برنامه‌های مختلف، با یکدیگر سازگار و قابل تعامل هستند. در اکوسیستم اتریوم، این استانداردها به نام ERC ($Comment\ for\ Request\ Ethereum$) شناخته می‌شوند.

تحلیل عمیق استانداردهای توکن ERC در زنجیره تأمین

انتخاب استاندارد توکن مناسب، یکی از مهم‌ترین تصمیمات معماری در طراحی یک سیستم زنجیره تأمین مبتنی بر زنجیره بلوکی است. هر استاندارد، برای نوع خاصی از دارایی طراحی شده و دارای مزایا و محدودیت‌های خود است.

$ERC-20$**: استاندارد توکن‌های مثلی** ($Tokens\ Fungible$) $ERC-20$ اولین و مشهورترین استاندارد توکن در اتریوم است که در سال ۲۰۱۵ معرفی شد. این استاندارد، یک رابط کاربری مشترک برای توکن‌هایی تعریف می‌کند که «مثلی» هستند؛ یعنی هر واحد از آن‌ها با هر واحد دیگری از همان توکن، قابل تعویض و دارای ارزش یکسان است. بهترین مثال برای یک دارایی مثلی، پول است: یک اسکناس ۱۰ دلاری با هر اسکناس ۱۰ دلاری دیگری ارزش یکسانی دارد.
توابع کلیدی استاندارد $ERC-20$ عبارتند از:

-

$$totalSupply()$$

: تعداد کل توکن‌های موجود را برمی‌گرداند.

-

$$balanceOf(addressaccount)$$

: موجودی توکن یک آدرس خاص را نشان می‌دهد.

- $transfer(addressrecipient, uint256amount)$

: تعداد مشخصی توکن را به یک آدرس دیگر منتقل می‌کند.

- $approve(addressspender, uint256amount)$

: به یک آدرس دیگر (معمولاً یک قرارداد هوشمند) اجازه می‌دهد تا از طرف شما، تا سقف مشخصی توکن خرج کند.

- $transferFrom(addresssender, addressrecipient, uint256amount)$

: توسط آدرس $spender$ برای انتقال توکن از $sender$ به $recipient$ استفاده می‌شود.

کاربرد در زنجیره تأمین: $ERC-20$ برای نمایندگی کالاهای انبوه و قابل تعویض بسیار مناسب است. برای مثال، یک شرکت کشاورزی می‌تواند موجودی گندم خود را در قالب توکن‌های $ERC-20$ (مثلاً هر توکن نماینده یک کیلوگرم گندم) توکنیزه کند. این توکن‌ها می‌توانند به راحتی بین تولیدکنندگان، توزیع‌کنندگان و کارخانه‌ها منتقل و معامله شوند. **محدودیت اصلی:** این استاندارد به هیچ عنوان قادر به نمایندگی دارایی‌های منحصربه‌فرد نیست. تمام توکن‌های یک قرارداد $ERC-20$ یکسان هستند و راهی برای تمایز قائل شدن بین آن‌ها وجود ندارد. این امر استفاده از آن را برای ردیابی آیتم‌های خاص و غیرمثلی غیرممکن می‌سازد.

$ERC-721$: **استاندارد توکن‌های غیرمثلی** ($NFTs-Tokens\ Non-Fungible$) برای حل محدودیت $ERC-20$، استاندارد $ERC-721$ در سال ۲۰۱۸ و با الهام از پروژه محبوب $CryptoKitties$ معرفی شد. این استاندارد برای نمایندگی دارایی‌هایی طراحی شده که هر کدام «منحصربه‌فرد» و «غیرقابل تعویض» هستند. هر توکن در یک قرارداد $ERC-721$ دارای یک شناسه یکتا ($tokenId$) است که آن را از تمام توکن‌های دیگر متمایز می‌کند.
ویژگی‌های کلیدی $ERC-721$ عبارتند از:

- هر توکن یک شناسه منحصربه‌فرد و یک مالک مشخص دارد.
- تابعی مانند

$ownerOf(uint256tokenId)$

وجود دارد که مالک یک توکن خاص را برمی‌گرداند.

- انتقال مالکیت به صورت یک به یک انجام می‌شود؛ یعنی یک توکن خاص از یک مالک به مالک دیگر منتقل می‌گردد.

کاربرد در زنجیره تأمین: $ERC-721$ راهکاری ایده‌آل برای ردیابی کالاهای با ارزش و منحصربه‌فرد است. هر NFT می‌تواند به عنوان شناسنامه دیجیتال یک آیتم خاص عمل کند. برخی از کاربردهای آن عبارتند از:

- **کالاهای لوکس:** ردیابی یک ساعت سوئیسی یا یک کیف دستی برند با شماره سریال مشخص.
- : □□□. : .

- . : **محدودیت اصلی:** در حالی که $ERC-721$ برای آیتم‌های منحصربه‌فرد عالی است، برای مدیریت کالاهای مثلی یا نیمه‌مثلی بسیار ناکارآمد است. فرض کنید یک شرکت بخواهد ۱۰۰۰ عدد از یک قطعه یدکی یکسان را منتقل کند. با استفاده از $ERC-721$، باید ۱۰۰۰ توکن مجزا (با $tokenId$های متفاوت) ساخته شود و انتقال آن‌ها نیازمند ۱۰۰۰ تراکنش جداگانه خواهد بود. این فرآیند از نظر هزینه گس و سرعت، بسیار ناکارآمد و غیراقتصادی است.

$ERC-1155$: **استاندارد چند-توکنی** ($The Multi-Token Standard$) با توجه به محدودیت‌های دو استاندارد قبلی، مشخص شد که بسیاری از کاربردها (به ویژه بازی‌های کامپیوتری و زنجیره تأمین) نیازمند یک راهکار ترکیبی هستند که بتواند هر دو نوع دارایی مثلی و غیرمثلی را به صورت همزمان و کارآمد مدیریت کند. این نیاز منجر به توسعه استاندارد $ERC-1155$ توسط تیم پروژه $Enjin$ در سال ۲۰۱۸ شد. $ERC-1155$ یک استاندارد «چند-توکنی» است که به یک قرارداد هوشمند واحد اجازه می‌دهد تا تعداد نامحدودی از انواع توکن‌های مختلف (اعم از مثلی و غیرمثلی) را مدیریت کند.

نوآوری کلیدی $ERC-1155$: ایده اصلی در این استاندارد، تفکیک «نوع توکن» از «تعداد» آن است. در حالی که در $ERC-721$ هر توکن یک موجودیت مستقل بود، در $ERC-1155$ ما با «کلاس‌های توکن» سروکار داریم که هر کدام با یک شناسه (id) مشخص می‌شوند. سپس برای هر آدرس، موجودی آن از هر کلاس توکن به صورت یک عدد ($amount$) ذخیره می‌شود.

- برای نمایندگی یک **توکن غیرمثلی** (NFT)، یک کلاس توکن جدید با یک id منحصربه‌فرد ایجاد کرده و تنها یک واحد ($amount = 1$) از آن را به یک مالک اختصاص می‌دهیم.
- برای نمایندگی یک **توکن مثلی** ($Fungible$)، یک کلاس توکن با یک id مشخص ایجاد کرده و می‌توانیم هر تعداد از آن را بین مالکان مختلف توزیع کنیم.

این معماری، قدرت و انعطاف‌پذیری بی‌نظیری را فراهم می‌کند. توابع اصلی این استاندارد نیز این ماهیت دوگانه را بازتاب می‌دهند:

- $$balanceOf(addressaccount, uint256id)$$

: موجودی یک آدرس خاص از یک کلاس توکن مشخص را برمی‌گرداند.

- $safeTransferFrom(address from, address to, uint256 id, uint256 amount, bytes data)$

: تعداد مشخصی ($amount$) از یک کلاس توکن (id) را منتقل می‌کند.

- $balanceOfBatch(address[] accounts, uint256[] ids)$

: موجودی چندین آدرس از چندین کلاس توکن را در یک فراخوانی واحد برمی‌گرداند.

- $safeBatchTransferFrom(address from, address to, uint256[] ids, uint256[] amounts, bytes data)$

: این تابع، قابلیت کلیدی و انقلابی این استاندارد است. این تابع اجازه می‌دهد تا چندین نوع توکن مختلف با مقادیر متفاوت، همگی در **یک تراکنش واحد** منتقل شوند.

چرا $ERC-1155$ برای زنجیره تأمین ایده‌آل است؟ این استاندارد، پاسخی مستقیم به نیازهای پیچیده زنجیره تأمین مدرن است. پروژه حاضر با انتخاب هوشمندانه این استاندارد[؟، ؟]، از مزایای زیر بهره‌مند می‌شود:

۱. **کارایی بی‌نظیر:** فرض کنید یک کارخانه خودروسازی، یک خودروی جدید را به یک نمایندگی ارسال می‌کند. این محموله شامل خود خودرو (یک آیتم غیرمثلی)، ۴ حلقه لاستیک (یک دسته از آیتم‌های مثلی) و ۱۰ لیتر روغن موتور (یک دسته دیگر از آیتم‌های مثلی) است. با استفاده از استانداردهای قدیمی، این فرآیند نیازمند چندین تراکنش مجزا بود. اما با $ERC-1155$، تمام این دارایی‌ها را می‌توان با فراخوانی تابع

$safeBatchTransferFrom$

در یک تراکنش واحد و بهینه منتقل کرد. این امر به شدت هزینه گس را کاهش داده و توان عملیاتی سیستم را بالا می‌برد.

۲. **انعطاف‌پذیری کامل:** سیستم طراحی‌شده در این پروژه، محدود به یک نوع کالا نیست. این سیستم می‌تواند به صورت همزمان یک قطعه ماشین‌آلات سنگین و منحصربه‌فرد را به عنوان یک NFT و هزاران پیچ و مهره استاندارد را به عنوان توکن‌های مثلی، همگی در یک قرارداد واحد مدیریت کند.

۳. .. : انتخاب $ERC-1155$ نشان‌دهنده بلوغ معماری پروژه و درک عمیق از نیازهای عملیاتی یک زنجیره تأمین واقعی است.

مطالعات موردی با استفاده از شبکه‌های عمومی

با جذاب‌تر شدن شبکه‌های عمومی، پروژه‌های متعددی تلاش کرده‌اند تا از این پلتفرم‌ها برای کاربردهای زنجیره تأمین استفاده کنند.

- **پلتفرم $VeChain$ (VET):** $VeChain$ یک زنجیره بلوکی عمومی است که از ابتدا به طور خاص با هدف کاربردهای سازمانی و زنجیره تأمین طراحی شده است. این پلتفرم از یک مدل دو-توکنی استفاده می‌کند (VET برای ارزش و $VTHO$ برای پرداخت هزینه تراکنش‌ها) تا هزینه گس را برای شرکت‌ها قابل پیش‌بینی‌تر کند. $VeChain$ با شرکت‌های بزرگی در صنایع مختلف از جمله کالاهای لوکس ($LVMH$) و ایمنی مواد غذایی همکاری کرده و با ترکیب تگ‌های $RFID/NFC$ با زنجیره بلوکی، راهکارهای ردیابی جامعی ارائه داده است.
- **پروژه‌های اصالت‌سنجی کالاهای لوکس:** شرکت‌هایی مانند $Arianee$ از NFTها بر روی شبکه اتریوم برای ایجاد یک «پاسپورت دیجیتال» برای کالاهای لوکس استفاده می‌کنند. هر محصول دارای یک NFT منحصربه‌فرد است که تاریخچه مالکیت آن را ثبت کرده و اصالت آن را تضمین می‌کند. این NFT می‌تواند به همراه کالای فیزیکی به مالک بعدی منتقل شود.

این پروژه‌ها نشان‌دهنده روند رو به رشد استفاده از شبکه‌های عمومی و توکنیزه‌سازی برای حل مشکلات زنجیره تأمین هستند.

۲-۲-۳ تحلیل شکاف پژوهشی و جایگاه نوآورانه پروژه حاضر

پس از بررسی دقیق نسل‌های مختلف راهکارهای زنجیره بلوکی، از پلتفرم‌های خصوصی مانند $Hyperledger$ $Fabric$ گرفته تا راهکارهای مبتنی بر استانداردهای مختلف توکن در شبکه‌های عمومی، اکنون می‌توانیم جایگاه پروژه حاضر را در این چشم‌انداز مشخص کنیم. جدول زیر یک مقایسه کیفی بین رویکردهای اصلی ارائه می‌دهد:

جدول ۲-۲: مقایسه کیفی رویکردهای مختلف زنجیره بلوکی برای زنجیره تأمین

معیار	$Fabric\ Hyperledger$	$onPublicNet\ ERC-721$	$onPublicNet\ ERC-1155$ **(پروژه حاضر)**
وصی داده	عالی	ضعیف	ضعیف (قابل بهبود با لایه ۲)
ن پردازشی	بالا	پایین	متوسط (قابل بهبود با لایه ۲)
ت همکاری	بسیار ضعیف	عالی	عالی
ری دارایی	متوسط	ضعیف	عالی
ه تراکنش	پایین (هزینه زیرساخت)	بالا	متوسط (بسیار بهینه در بچ)
عدم تمرکز	متوسط (کنسرسیوم)	کامل	کامل

تحلیل جدول ۲-۲ و بررسی کارهای پیشین، یک **شکاف پژوهشی و عملیاتی** را آشکار می‌سازد: نیاز به یک راهکار که بتواند **انعطاف‌پذیری و قابلیت همکاری شبکه‌های عمومی** را با **کارایی عملیاتی و پاسخگویی به نیازهای پیچیده زنجیره تأمین** ترکیب کند.

- پلتفرم‌های خصوصی مانند $Fabric$، با قربانی کردن عدم تمرکز و قابلیت همکاری، به حریم خصوصی و توان پردازشی دست یافتند، اما در یک اکوسیستم ایزوله باقی ماندند.
- راهکارهای مبتنی بر $ERC-721$ بر روی شبکه‌های عمومی، قابلیت همکاری را فراهم کردند، اما برای مدیریت زنجیره‌های تأمین با محصولات متنوع، ناکارآمد و گران بودند.

نوآوری پروژه حاضر در پر کردن این شکاف است. این پروژه با اتخاذ یک رویکرد چندلایه و هوشمندانه، تلاش می‌کند تا بهترین ویژگی‌های هر دو جهان را با یکدیگر ترکیب کند:

۱. **انتخاب استراتژیک** $ERC-1155$: همانطور که به تفصیل شرح داده شد، این استاندارد به تنهایی مشکل مدیریت دارایی‌های متنوع را به کارآمدترین شکل ممکن حل می‌کند و پایه و اساس یک زنجیره تأمین انعطاف‌پذیر را فراهم می‌آورد.

۲. **معماری هوشمند برای مدیریت فراداده:** این پروژه به جای نادیده گرفتن مشکل هزینه ذخیره‌سازی، با ارائه یک راهکار مبتنی بر هش $Keccak256$ و ذخیره‌سازی $off-chain$، یک معماری اقتصادی و در عین حال امن را برای مدیریت فراداده‌ها پیاده‌سازی می‌کند. این مکانیزم، یکپارچگی داده‌ها را بدون تحمیل هزینه‌های گزاف زنجیره بلوکی تضمین می‌نماید.

۳. **نگاه آینده‌نگر به انطباق‌پذیری:** با تعبیه قابلیت‌هایی مانند محاسبه خودکار مالیات، این پروژه از یک راهکار صرفاً فنی فراتر رفته و به چالش‌های دنیای واقعی کسب‌وکار، یعنی انطباق با قوانین نظارتی و مالی، پاسخ می‌دهد. این ویژگی، پذیرش عملیاتی سیستم توسط شرکت‌ها را تسهیل می‌کند.

بنابراین، پروژه حاضر نه تنها یک پیاده‌سازی دیگر از زنجیره بلوکی در زنجیره تأمین نیست، بلکه یک «سنتز نوآورانه» از بهترین فناوری‌ها و معماری‌های موجود است. این پروژه با یادگیری از محدودیت‌های نسل‌های پیشین، یک راهکار جامع، کارآمد و عملیاتی ارائه می‌دهد که یک گام مهم رو به جلو در تکامل سامانه‌های زنجیره تأمین غیرمتمرکز محسوب می‌شود.

۲-۳ تحلیل شکاف پژوهشی و ارائه نوآوری پروژه

در بخش‌های پیشین این فصل، یک تحلیل جامع از سیر تکاملی سیستم‌های مدیریت زنجیره تأمین ارائه گردید. این تحلیل از سیستم‌های برنامه‌ریزی منابع سازمانی (ERP) متمرکز آغاز شد، به بررسی نسل اول فناوری‌های دیجیتالی مانند $RFID$ و IoT پرداخت و در نهایت، به ارزیابی دو نسل اصلی از راهکارهای مبتنی بر زنجیره بلوکی منتهی شد: نسل اول مبتنی بر پلتفرم‌های خصوصی و نیازمند مجوز مانند $Fabric\ Hyperledger$، و نسل دوم مبتنی بر شبکه‌های عمومی و استانداردهای توکنیزه‌سازی مانند $ERC-721$. هر یک از این پارادایم‌ها، گامی مهم در جهت حل مشکلات پیچیده زنجیره تأمین بوده‌اند، اما در عین حال، هر کدام با محدودیت‌ها و چالش‌های خاص خود روبرو شده و شکاف‌های مهمی را در ادبیات تحقیق و در پیاده‌سازی‌های عملی باقی گذاشته‌اند.

هدف اصلی این بخش، انجام یک ارزیابی انتقادی و عمیق بر روی این شکاف‌هاست. ما با سنتز یافته‌های بخش‌های قبل، به صورت نظام‌مند نشان خواهیم داد که راهکارهای پیشین در پاسخگویی به نیازهای چندوجهی یک زنجیره تأمین مدرن، دچار کاستی بوده‌اند. این تحلیل شکاف، بستری را فراهم می‌آورد تا بتوانیم جایگاه نوآورانه و منحصربه‌فرد پروژه حاضر را به روشنی مشخص کنیم. در نهایت، استدلال خواهد شد که این پروژه، با ارائه یک معماری سنتز شده و هوشمندانه، نه تنها به شکاف‌های شناسایی‌شده پاسخ می‌دهد، بلکه نماینده یک «نسل سوم» از راهکارهای زنجیره تأمین غیرمتمرکز است که یک گام به پیاده‌سازی عملیاتی و پذیرش گسترده نزدیک‌تر شده است.

۲-۳-۱ شناسایی شکاف‌های کلیدی در ادبیات تحقیق

پس از مرور گسترده راهکارهای موجود، می‌توان سه شکاف اصلی و بنیادین را شناسایی کرد که اکثر پروژه‌های پیشین به صورت جامع به آن‌ها نپرداخته‌اند. این شکاف‌ها در سه حوزه کلیدی قرار دارند: مدیریت دارایی‌های ناهمگون، یکپارچگی داده‌های خارج از زنجیره، و انطباق‌پذیری با محیط‌های واقعی تجاری و نظارتی.

شکاف اول: چالش مدیریت دارایی‌های ناهمگون

یک زنجیره تأمین واقعی، اکوسیستمی بسیار متنوع از دارایی‌هاست. این دارایی‌ها از نظر ماهیت، ارزش و نحوه مدیریت، تفاوت‌های بنیادینی با یکدیگر دارند. می‌توان آن‌ها را در یک طیف، از کاملاً مثلی تا

کاملاً غیرمثلی، دسته‌بندی کرد:

- **دارایی‌های کاملاً مثلی** ($Fungible$): این‌ها مواد اولیه یا کالاهای انبوهی هستند که هر واحد از آن‌ها با واحد دیگر قابل تعویض است. به عنوان مثال، یک کیلوگرم گندم از یک بچ مشخص، با کیلوگرم دیگری از همان بچ تفاوتی ندارد. مدیریت این دارایی‌ها مبتنی بر «تعداد» و «مقدار» است.
- **دارایی‌های کاملاً غیرمثلی** ($Non-Fungible$): این‌ها آیتم‌های منحصربه‌فردی هستند که هر کدام هویت و تاریخچه مختص به خود را دارند. یک خودرو با شماره شاسی مشخص، یک الماس با گواهی اصالت، یا یک قطعه هنری، نمونه‌هایی از این دارایی‌ها هستند. مدیریت این‌ها مبتنی بر «هویت یکتا» است.
- **دارایی‌های نیمه‌مثلی** ($Semi-Fungible$): این دسته که اغلب نادیده گرفته می‌شود، دارایی‌هایی هستند که در یک دوره زمانی مثلی بوده و در دوره‌ای دیگر به غیرمثلی تبدیل می‌شوند. برای مثال، یک بلیط کنسرت برای یک جایگاه مشخص، قبل از شروع رویداد با بلیط دیگری از همان جایگاه قابل تعویض است (مثلی)، اما پس از استفاده و تبدیل شدن به یک یادگاری، منحصربه‌فرد و غیرمثلی می‌شود.

اکثر راهکارهای زنجیره بلوکی پیشین، در ارائه یک مدل یکپارچه برای مدیریت این طیف گسترده از دارایی‌ها دچار مشکل بوده‌اند.

محدودیت‌های راهکارهای تک-استانداردی: راهکارهای نسل دوم که بر روی شبکه‌های عمومی ساخته شده‌اند، معمولاً خود را به یکی از دو استاندارد اصلی محدود کرده‌اند:

۱. **رویکرد مبتنی بر** $ERC-20$: این پروژه‌ها بر روی مدیریت مواد اولیه و کالاهای انبوه تمرکز کرده‌اند. در حالی که این رویکرد برای زنجیره‌های تأمین کالاهای اساسی (مانند محصولات کشاورزی) کارآمد است، اما به کلی از ردیابی آیتم‌های منحصربه‌فرد و محصولات نهایی که نیازمند شناسنامه دیجیتال یکتا هستند، عاجز است.

۲. **رویکرد مبتنی بر** $ERC-721$: این پروژه‌ها که محبوبیت بیشتری داشته‌اند، بر روی تضمین اصالت کالاهای لوکس، داروها و قطعات صنعتی متمرکز شده‌اند. هر محصول به یک NFT منحصربه‌فرد تبدیل می‌شود که تاریخچه آن را ثبت می‌کند. مشکل این رویکرد، ناکارآمدی شدید آن در مدیریت اجزای تشکیل‌دهنده یا مواد اولیه آن محصول است. برای مثال، در زنجیره تأمین یک خودرو، ردیابی خود خودرو با یک NFT منطقی است، اما ردیابی هزاران پیچ و مهره یا لیترها روغن موتور که در تولید آن به کار رفته، با استفاده از NFTهای مجزا، از نظر هزینه و سرعت، یک فاجعه عملیاتی خواهد بود. این امر منجر به ایجاد یک دید ناقص از زنجیره تأمین می‌شود که در آن، تنها محصول نهایی قابل ردیابی است و نه مواد اولیه آن.

محدودیت‌های پلتفرم‌های خصوصی: پلتفرم‌های سازمانی مانند $Fabric\ Hyperledger$ مدل دارایی انعطاف‌پذیرتری را ارائه می‌دهند که در آن می‌توان هر نوع ساختار داده‌ای را به عنوان یک دارایی تعریف کرد. با این حال، این پلتفرم‌ها فاقد یک «استاندارد» مورد توافق جهانی برای تمایز بین دارایی‌های مثلی و غیرمثلی هستند. این امر منجر به ایجاد راهکارهای جزیره‌ای و سفارشی می‌شود که قابلیت همکاری با یکدیگر یا با اکوسیستم گسترده‌تر دارایی‌های دیجیتال را ندارند. یک دارایی تعریف‌شده در یک شبکه $Fabric$، نمی‌تواند به راحتی در یک بازار NFT عمومی لیست شود یا به عنوان وثیقه در یک پروتکل $DeFi$ استفاده گردد.

بیان دقیق شکاف اول: با توجه به این تحلیل، شکاف اول را می‌توان اینگونه تعریف کرد: **فقدان یک راهکار جامع و استاندارد در ادبیات تحقیق که بتواند به صورت بومی، کارآمد و یکپارچه، کل طیف دارایی‌های ناهمگون یک زنجیره تأمین (از مواد اولیه مثلی تا محصولات نهایی غیرمثلی) را در قالب یک پروتکل واحد و قابل همکاری با اکوسیستم گسترده‌تر مدیریت کند.**

شکاف دوم: مسئله یکپارچگی داده‌های خارج از زنجیره

همانطور که در بخش چالش‌ها ذکر شد، ذخیره‌سازی داده‌های حجیم بر روی زنجیره بلوکی از نظر اقتصادی غیرعملی است. این یک واقعیت فنی است که تمام پروژه‌های جدی زنجیره تأمین باید با آن روبرو شوند. در نتیجه، یک معماری ترکیبی که در آن، داده‌های اصلی (فراداده) در خارج از زنجیره ($Off-chain$) و تنها یک اثبات یا ارجاع به آن در داخل زنجیره ($On-chain$) ذخیره می‌شود، امری اجتناب‌ناپذیر است. با این حال، نحوه پیاده‌سازی این معماری، خود یک چالش بزرگ و یک شکاف مهم در پژوهش‌های پیشین است.

بسیاری از پروژه‌های اولیه، این چالش را به سادگی نادیده گرفته یا راهکارهای ضعیفی برای آن ارائه داده‌اند:

- **نادیده گرفتن مشکل:** برخی پروژه‌های آکادمیک، صرفاً بر روی منطق $On-chain$ تمرکز کرده و فرض می‌کنند که فراداده به نوعی در دسترس و معتبر است، بدون اینکه معماری مشخصی برای آن ارائه دهند.
- **استفاده از سرورهای متمرکز:** بسیاری از راهکارهای تجاری، برای ذخیره‌سازی فراداده از سرورهای وب سنتی ($Web2$) و پایگاه‌های داده متمرکز استفاده می‌کنند. در این مدل، یک URL به سرور مربوطه در قرارداد هوشمند ذخیره می‌شود. این رویکرد، کل فلسفه زنجیره بلوکی را زیر سؤال می‌برد. زیرا با این کار، ما مجدداً یک «نقطه شکست واحد» و یک «مرجع قابل اعتماد» مرکزی را به سیستم وارد کرده‌ایم. اگر آن سرور هک شود و داده‌ها تغییر کنند، یا اگر شرکت مالک سرور ورشکست شود و سرور از دسترس خارج گردد، ارجاع ثبت‌شده بر روی زنجیره بلوکی بی‌معنی و بی‌ارزش خواهد شد. این راهکار، مشکل اعتماد را حل نمی‌کند، بلکه صرفاً آن را به مکانی دیگر منتقل می‌نماید.

مفهوم گسترده‌تر «مشکل اوراکل»: این چالش، نمونه‌ای از یک مسئله بزرگ‌تر در دنیای زنجیره بلوکی است که به آن «مشکل اوراکل» ($Problem\ Oracle\ The$) گفته می‌شود. قراردادهای هوشمند، محیط‌های اجرایی بسته‌ای هستند که به صورت بومی، به داده‌های دنیای خارج از خود دسترسی ندارند. اوراکل‌ها، سرویس‌هایی هستند که به عنوان پل عمل کرده و داده‌های دنیای واقعی را به صورت قابل اعتماد به داخل زنجیره بلوکی وارد می‌کنند. در مسئله ما، سیستم ذخیره‌سازی $Off-chain$ نقش یک نوع اوراکل را برای فراداده ایفا می‌کند. اگر این اوراکل متمرکز و غیرقابل اعتماد باشد، کل امنیت و اعتبار سیستم به خطر می‌افتد.

بیان دقیق شکاف دوم: بنابراین، شکاف دوم را می‌توان اینگونه تعریف کرد: **فقدان یک معماری استاندارد و غیرمتمرکز برای مدیریت چرخه حیات فراداده‌های** $Off-chain$ **که بتواند سه ویژگی کلیدی را به صورت همزمان تضمین کند: یکپارچگی** ($Integrity$)**، در دسترس بودن** ($Availability$) **و تغییرناپذیری** ($Immutability$) **داده‌ها، بدون تکیه بر یک مرجع مرکزی قابل اعتماد.**

شکاف سوم: فقدان انطباق‌پذیری با محیط‌های نظارتی و تجاری بزرگ‌ترین مانع بر سر راه پذیرش گسترده فناوری زنجیره بلوکی در سطح سازمانی، صرفاً فنی نیست. بسیاری از پروژه‌های زنجیره بلوکی در یک «خلأ تجاری و قانونی» توسعه می‌یابند. آن‌ها بر روی جنبه‌های الگوریتمی و رمزنگاری تمرکز می‌کنند و واقعیت‌های پیچیده دنیای کسب‌وکار و الزامات قانونی را نادیده می‌گیرند. یک شرکت نمی‌تواند سیستمی را به کار گیرد که با قوانین مالیاتی، گمرکی و تجاری که ملزم به رعایت آن‌هاست، در تضاد باشد.

اکثر پروژه‌های زنجیره تأمین پیشین، در این حوزه سکوت کرده‌اند. آن‌ها نشان می‌دهند که چگونه می‌توان یک کالا را ردیابی کرد، اما به سؤالات حیاتی زیر پاسخ نمی‌دهند:

- چگونه مالیات بر ارزش افزوده (VAT) در هر مرحله از انتقال مالکیت محاسبه و پرداخت می‌شود؟
- چگونه اسناد مورد نیاز گمرک به صورت دیجیتال و قابل تأیید تولید و ارائه می‌گردد؟
- چگونه می‌توان بین شفافیت مورد نیاز برای حسابرسی و محرمانگی لازم برای حفظ مزیت رقابتی، تعادل برقرار کرد؟
- در صورت بروز اختلاف تجاری، وضعیت حقوقی تراکنش‌های ثبت‌شده بر روی زنجیره بلوکی چیست؟

این بی‌توجهی به الزامات دنیای واقعی، باعث شده است که بسیاری از این پروژه‌ها در حد یک طرح آزمایشی ($Pilot$) باقی بمانند و به مرحله تولید انبوه نرسند. زیرا ادغام آن‌ها با فرآیندهای مالی و قانونی موجود شرکت‌ها، بسیار دشوار و پرهزینه است.

بیان دقیق شکاف سوم: بر این اساس، شکاف سوم به این صورت تعریف می‌شود: **وجود یک گسست عمیق بین قابلیت‌های فنی پلتفرم‌های زنجیره بلوکی و الزامات عملیاتی، تجاری و نظارتی که شرکت‌ها در زنجیره تأمین با آن روبرو هستند، و فقدان راهکارهایی که به صورت بومی، قابلیت‌های انطباق‌پذیری** ($Compliance$) **را در پروتکل خود تعبیه کرده باشند.**

۲-۳-۲ ارائه راهکار نوآورانه پروژه: یک معماری سنتز شده

پروژه حاضر، با شناسایی دقیق این سه شکاف کلیدی، یک راهکار جامع و چندلایه ارائه می‌دهد که هدف آن، نه تنها پیاده‌سازی یک قابلیت فنی جدید، بلکه ارائه یک «سنتز نوآورانه» از بهترین رویکردها برای پر کردن این شکاف‌هاست. معماری این پروژه، پاسخی مستقیم به هر یک از چالش‌های مطرح‌شده است.

نوآوری اول: مدیریت یکپارچه دارایی‌ها با استاندارد $ERC-1155$

این پروژه به صورت مستقیم به «شکاف مدیریت دارایی‌های ناهمگون» پاسخ می‌دهد. با انتخاب استراتژیک استاندارد $ERC-1155$، این سیستم از ابتدا با این فرض طراحی شده است که یک زنجیره تأمین واقعی، با ترکیبی از دارایی‌های مثلی و غیرمثلی سروکار دارد. این انتخاب، یک تصمیم فنی صرف نیست، بلکه یک تصمیم معماری بنیادین با پیامدهای عملی گسترده است.

فراتر از یک استاندارد فنی: یک مدل عملیاتی انعطاف‌پذیر قدرت واقعی $ERC-1155$ در توانایی آن برای مدل‌سازی فرآیندهای لجستیکی پیچیده در دنیای واقعی نهفته است. در ادامه با چند مثال، این قابلیت تشریح می‌شود:

- **صنعت داروسازی:** یک شرکت داروسازی را در نظر بگیرید. این شرکت می‌تواند یک بچ کامل از یک داروی خاص را که شامل هزاران ویال یکسان است، به عنوان یک دسته از توکن‌های مثلی با شناسه مثلاً

$$ID = 101$$

و تعداد

$$Amount = 10000$$

توکنیزه کند. سپس، هر یک از این ویال‌ها را در حین بسته‌بندی نهایی، به یک توکن غیرمثلی منحصربه‌فرد با شماره سریال مشخص تبدیل نماید. استاندارد $ERC-1155$ این قابلیت تبدیل بین حالت مثلی و غیرمثلی را نیز تسهیل می‌کند. این فرآیند، امکان ردیابی هم در سطح بچ (برای کنترل کیفیت کلی) و هم در سطح آیتم (برای جلوگیری از فروش داروی تقلبی) را فراهم می‌آورد.

- **صنعت الکترونیک:** یک شرکت تولیدکننده لپ‌تاپ را تصور کنید. این شرکت می‌تواند هر

لپ‌تاپ تولید شده را با شماره سریال منحصربه‌فرد خود، به عنوان یک NFT (مثلاً

$$ID = 202$$

،

$$Amount = 1$$

) در سیستم ثبت کند. همزمان، می‌تواند قطعات یدکی استاندارد مانند باتری یا شارژر را به عنوان توکن‌های مثلی (مثلاً

$$ID = 203$$

،

$$Amount = 5000$$

) مدیریت نماید. زمانی که یک مشتری لپ‌تاپ را به همراه یک شارژر اضافی خریداری می‌کند، تابع

$$safeBatchTransferFrom$$

به فروشنده اجازه می‌دهد تا هر دو آیتم (یک NFT و یک توکن مثلی) را در یک تراکنش واحد و بهینه به مشتری منتقل کند.

این سطح از انعطاف‌پذیری و کارایی، که مستقیماً از قابلیت‌های استاندارد $ERC-1155$ نشأت می‌گیرد، پاسخی قدرتمند به شکاف اول است و سیستم را برای کاربرد در طیف وسیعی از صنایع آماده می‌سازد.

نوآوری دوم: تضمین صحت فراداده با معماری ترکیبی $On-Chain/Off-Chain$

این پروژه برای پاسخ به «شکاف یکپارچگی داده‌های خارج از زنجیره»، یک معماری دقیق و امن ارائه می‌دهد که بر پایه دو فناوری کلیدی استوار است: سیستم فایل بین‌سیاره‌ای ($IPFS$) و هش رمزنگاری $Keccak256$.

چرخه حیات کامل فراداده در معماری پیشنهادی: برای درک کامل این نوآوری، باید چرخه کامل ثبت و اعتبارسنجی فراداده را دنبال کنیم:

۱. **مرحله اول: ایجاد و بسته‌بندی فراداده:** هنگامی که یک تولیدکننده قصد ثبت محصول جدیدی را دارد، اطلاعات کامل آن را در داشبورد مدیریتی وارد می‌کند. برنامه کاربردی، این اطلاعات را در یک فایل با ساختار استاندارد (مانند $JSON$) بسته‌بندی می‌کند. این فایل شامل تمام جزئیات محصول است.

۲. **مرحله دوم: بارگذاری در $IPFS$ و دریافت شناسه محتوا (CID):** برنامه کاربردی، این فایل $JSON$ را در شبکه $IPFS$ بارگذاری می‌کند. $IPFS$ یک شبکه ذخیره‌سازی همتا به همتا و غیرمتمرکز است. برخلاف سرورهای وب سنتی که در آن، محتوا بر اساس «مکان» آدرس‌دهی می‌شود ($addressing\ Location-based$ - مانند یک URL)، در $IPFS$ محتوا بر اساس «هش» خود آدرس‌دهی می‌شود ($addressing\ Content-based$). پس از بارگذاری، $IPFS$ یک شناسه منحصربه‌فرد به نام CID ($Identifier\ Content$) به فایل اختصاص می‌دهد که در واقع هش رمزنگاری شده محتوای آن فایل است. این ویژگی دو مزیت بزرگ دارد:

- **تغییرناپذیری:** اگر حتی یک بیت از محتوای فایل تغییر کند، CID آن نیز کاملاً تغییر خواهد کرد.
- **عدم تمرکز و در دسترس بودن:** فایل در چندین گره در شبکه $IPFS$ توزیع می‌شود که این امر، ریسک از دسترس خارج شدن به دلیل خرابی یک سرور واحد را از بین می‌برد.

۳. **مرحله سوم: محاسبه هش تأیید و ثبت بر روی زنجیره:** برنامه کاربردی، به صورت موازی، محتوای فایل $JSON$ را با استفاده از الگوریتم $Keccak256$ (که الگوریتم هش استاندارد در اتریوم است) هش می‌کند. سپس، در حین فراخوانی تابع $registerProduct$، هر دو مقدار، یعنی CID دریافت شده از $IPFS$ و هش $Keccak256$ محاسبه‌شده، به عنوان پارامتر به قرارداد هوشمند ارسال و بر روی زنجیره بلوکی ذخیره می‌شوند.

۴. **مرحله چهارم: فرآیند اعتبارسنجی غیرمتمرکز:** زمانی که یک مصرف‌کننده کد QR را اسکن می‌کند، برنامه کاربردی او فرآیند اعتبارسنجی زیر را به صورت خودکار انجام می‌دهد:

(آ) ابتدا CID و هش $Keccak256$ معتبر را از قرارداد هوشمند بر روی زنجیره بلوکی می‌خواند.

(ب) سپس با استفاده از CID، فایل فراداده اصلی را از شبکه غیرمتمرکز $IPFS$ بازیابی می‌کند.

(ج) به صورت محلی، هش $Keccak256$ محتوای فایل بازیابی‌شده را مجدداً محاسبه می‌کند.

(د) در نهایت، هش محاسبه‌شده محلی را با هش معتبر خوانده‌شده از زنجیره بلوکی مقایسه می‌کند.

تنها در صورتی که این دو هش کاملاً یکسان باشند، اصالت اطلاعات تأیید می‌شود. این معماری چندلایه، یک راهکار بسیار قوی، غیرمتمرکز و اقتصادی برای حل شکاف دوم ارائه می‌دهد.

نوآوری سوم: پل زدن به دنیای واقعی با محاسبه خودکار مالیات

مهم‌ترین و شاید جسورانه‌ترین نوآوری این پروژه، پاسخگویی مستقیم به «شکاف انطباق‌پذیری با محیط‌های نظارتی» است. این پروژه، به جای نادیده گرفتن الزامات دنیای واقعی، تلاش می‌کند تا از ویژگی‌های منحصربه‌فرد زنجیره بلوکی برای ایجاد راهکارهای نوین در حوزه فناوری‌های نظارتی ($RegTech$) بهره

ببرد. در این راستا، یک راهکار قابل اجرا برای **محاسبه و پرداخت خودکار مالیات** به عنوان بخشی از پروتکل ارائه می‌شود.

چرا زنجیره بلوکی بستر ایده‌آلی برای مالیات هوشمند است؟ زنجیره بلوکی، به دلیل سه ویژگی کلیدی خود، یک زیرساخت بی‌نظیر برای مدرن‌سازی سیستم‌های مالیاتی فراهم می‌کند:

۱. **شفافیت و قابلیت حسابرسی آنی:** هر تراکنشی که منجر به انتقال ارزش یا مالکیت می‌شود (و بالقوه مشمول مالیات است)، به صورت شفاف و تغییرناپذیر بر روی یک دفتر کل عمومی ثبت می‌گردد. این امر به نهادهای نظارتی اجازه می‌دهد تا به جای حسابرسی‌های دوره‌ای و مبتنی بر اسناد کاغذی، به یک حسابرسی آنی و مستمر دسترسی داشته باشند.

۲. **داده‌های قابل اعتماد و قطعی:** زمان، مبلغ و طرفین هر تراکنش به صورت رمزنگاری‌شده تأیید و ثبت می‌شوند. این قطعیت، اختلافات مربوط به زمان و مبلغ معاملات را که بخش بزرگی از فرآیندهای حسابرسی سنتی را تشکیل می‌دهد، از بین می‌برد و فرصت‌های فرار مالیاتی را به شدت کاهش می‌دهد.

۳. **قابلیت برنامه‌پذیری و خودکارسازی:** با استفاده از قراردادهای هوشمند، می‌توان قوانین مالیاتی را به صورت مستقیم در قالب کد پیاده‌سازی کرد. این کد می‌تواند به صورت خودکار و بدون دخالت انسان، در زمان وقوع هر تراکنش اجرا شود.

معماری پیشنهادی برای ماژول مالیات هوشمند راهکار قابل اجرای ارائه شده در این پروژه، مبتنی بر گسترش منطق تابع $transferWithTax$ است. این تابع، علاوه بر انتقال مالکیت توکن، زنجیره‌ای از اقدامات مرتبط با مالیات را نیز به صورت اتمی انجام خواهد داد:

۱. **فراخوانی منطق محاسبه مالیات:** در حین اجرای تابع انتقال، یک تابع داخلی به نام $alculateTax_c$ فراخوانی می‌شود.

۲. **پیاده‌سازی قوانین مالیاتی:** منطق این تابع می‌تواند به صورت‌های مختلفی پیاده‌سازی شود:

- **مدل ساده (نرخ ثابت):** ساده‌ترین مدل، اعمال یک نرخ مالیات ثابت (مثلاً درصد مشخصی به عنوان مالیات بر ارزش افزوده) بر ارزش اسمی معامله است.
- **مدل پویا (مبتنی بر اوراکل):** در یک مدل پیشرفته‌تر، قرارداد هوشمند می‌تواند از طریق یک «اوراکل»، اطلاعاتی مانند قیمت روز کالا یا نرخ‌های مالیاتی متغیر را از منابع خارجی دریافت کرده و محاسبات خود را بر اساس آن انجام دهد.
- **مدل چندنرخی (مبتنی بر دسته‌بندی):** قوانین مالیاتی می‌توانند بر اساس دسته‌بندی محصول (که در فراداده آن مشخص شده) متفاوت باشند. قرارداد هوشمند می‌تواند این دسته‌بندی را خوانده و نرخ مناسب را اعمال کند.

۳. **انتقال خودکار مبلغ مالیات:** پس از محاسبه مبلغ مالیات، قرارداد هوشمند به صورت خودکار آن مبلغ را از حساب فروشنده کسر کرده و مستقیماً به یک آدرس کیف پول از پیش تعیین‌شده که متعلق به سازمان امور مالیاتی است، واریز می‌کند.

۴. **ثبت رویداد مالیاتی:** یک رویداد ($Event$) مشخص برای ثبت جزئیات تراکنش مالیاتی (مبلغ، مبنای محاسبه، آدرس پرداخت) بر روی زنجیره بلوکی ثبت می‌شود تا برای حسابرسی‌های بعدی به راحتی قابل استناد باشد.

تمام این مراحل در یک تراکنش واحد و به صورت **اتمی** انجام می‌شود؛ یعنی یا تمام مراحل با موفقیت اجرا می‌شوند، یا کل تراکنش ناموفق خواهد بود. این ویژگی، تضمین می‌کند که هیچ معامله‌ای بدون پرداخت مالیات متعلقه انجام نخواهد شد. این رویکرد نوآورانه، نه تنها یک قابلیت فنی، بلکه یک «پل استراتژیک» بین دنیای غیرمتمرکز زنجیره بلوکی و دنیای ساختاریافته نظارتی است و به شکاف سوم به صورت مستقیم پاسخ می‌دهد.

۲-۳-۳ جمع‌بندی: جایگاه پروژه به عنوان یک راهکار نسل سوم

با توجه به تحلیل جامع ارائه شده، می‌توان ادعا کرد که پروژه حاضر، نماینده یک «نسل سوم» از راهکارهای زنجیره تأمین مبتنی بر زنجیره بلوکی است. این نسل، با یادگیری از تجربیات و محدودیت‌های دو نسل پیشین، به یک رویکرد سنتز شده و جامع‌تر دست یافته است:

- **نسل اول (مبتنی بر پلتفرم خصوصی):** تمرکز اصلی بر «حریم خصوصی» و «توان پردازشی» بود، اما به قیمت از دست دادن «قابلیت همکاری» و «عدم تمرکز واقعی».
- **نسل دوم (مبتنی بر توکنیزه‌سازی اولیه):** تمرکز بر «قابلیت همکاری» و «مالکیت دیجیتال» بود، اما با چالش‌های جدی در «کارایی» (به دلیل استفاده از استانداردهای تک‌منظوره) و «یکپارچگی داده» روبرو بود.
- **نسل سوم (رویکرد سنتز شده پروژه حاضر):** این نسل با ترکیب هوشمندانه فناوری‌ها، تلاش می‌کند تا به صورت همزمان به چندین هدف کلیدی دست یابد:

  ۱. **انعطاف‌پذیری دارایی:** با استفاده از استاندارد قدرتمند $ERC-1155$.

  ۲. **یکپارچگی داده:** با استفاده از معماری امن و اقتصادی $Keccak256 + IPFS$.

  ۳. **انطباق‌پذیری تجاری:** با ارائه راهکارهای قابل اجرا برای نیازمندی‌های دنیای واقعی مانند مالیات.

  ۴. **قابلیت همکاری:** با پایبندی به استانداردهای شبکه عمومی اتریوم.

بنابراین، این پروژه صرفاً یک پیاده‌سازی دیگر از یک ایده موجود نیست، بلکه یک گام رو به جلو در جهت بلوغ و عملیاتی‌سازی فناوری زنجیره بلوکی برای یکی از مهم‌ترین و پیچیده‌ترین صنایع جهان است. این پایان‌نامه، یک نقشه راه دقیق و یک نمونه اولیه قوی برای ساختن نسل آینده زنجیره‌های تأمین ارائه می‌دهد؛ زنجیره‌هایی که نه تنها کارآمدتر، بلکه به صورت قابل اثباتی، شفاف‌تر، امن‌تر و عادلانه‌تر خواهند بود.

فصل سوم

معماری و روش پیاده‌سازی سامانه

□□□□□□□ □□□□□□□□□

پس از بررسی مبانی نظری و تحلیل شکاف‌های موجود در پژوهش‌های پیشین در فصل دوم، این فصل به صورت کاملاً عملی و فنی به تشریح «معماری و روش پیاده‌سازی سامانه پیشنهادی» می‌پردازد. هدف این فصل، ارائه یک نقشه راه دقیق و شفاف از تمامی اجزای تشکیل‌دهنده سیستم، از قرارداد هوشمند در لایه زنجیره بلوکی گرفته تا واسطهای کاربری در لایه کاربری است. در این بخش، نه تنها «چه چیزی» ساخته شده، بلکه «چرا» و «چگونه»ی آن نیز با استناد به انتخاب‌های فنی و نمایش قطعه کدهای کلیدی، به تفصیل مورد بحث و بررسی قرار خواهد گرفت.

این فصل به مثابه قلب فنی پایان‌نامه عمل می‌کند و به چهار بخش اصلی تقسیم می‌شود: ابتدا، به معرفی و توجیه پشته فناوری ($Stack\ Technology$) انتخاب شده برای پروژه می‌پردازیم. سپس، معماری کلان و چندلایه سیستم را تشریح می‌کنیم. در ادامه، به صورت عمیق وارد جزئیات پیاده‌سازی هر یک از لایه‌های اصلی سیستم، یعنی لایه زنجیره بلوکی ($Backend$)، لایه ذخیره‌سازی خارج از زنجیره ($Off-chain$) و لایه کاربری ($Frontend$) خواهیم شد.

۳-۱ مقدمه و انتخاب فناوری‌ها

انتخاب مجموعه مناسبی از فناوری‌ها، یکی از حیاتی‌ترین مراحل در موفقیت هر پروژه نرم‌افزاری، به ویژه در حوزه‌های نوظهوری مانند زنجیره بلوکی است. پشته فناوری این پروژه با در نظر گرفتن اهداف کلیدی مانند امنیت، عدم تمرکز، کارایی، تجربه کاربری مدرن و قابلیت توسعه در آینده، به دقت انتخاب شده است. هر یک از ابزارهای به کار رفته، نقشی کلیدی در تحقق یکی از اهداف پروژه ایفا می‌کند.

۳-۱-۱ توجیه انتخاب فناوری‌های لایه زنجیره بلوکی

لایه زنجیره بلوکی، به عنوان هسته امنیتی و منطقی سیستم، نیازمند فناوری‌هایی است که بالاترین سطح از بلوغ، امنیت و پشتیبانی جامعه توسعه‌دهندگان را داشته باشند.

- **زبان برنامه‌نویسی $Solidity$ و ماشین مجازی اتریوم (EVM):** $Solidity$ به عنوان زبان برنامه‌نویسی پیشرو برای نوشتن قراردادهای هوشمند و EVM به عنوان پلتفرم اجرایی آن، به دلیل بلوغ، مستندات گسترده، جامعه توسعه‌دهندگان فعال و اکوسیستم وسیعی از ابزارها و کتابخانه‌ها، به عنوان استاندارد صنعتی شناخته می‌شوند. انتخاب این پلتفرم، قابلیت همکاری با هزاران برنامه غیرمتمرکز دیگر را نیز تضمین می‌کند.
- **کتابخانه‌های $OpenZeppelin$:** امنیت در قراردادهای هوشمند از اهمیت فوق‌العاده‌ای برخوردار است. به جای اختراع مجدد چرخ، این پروژه از قراردادهای پایه ارائه شده توسط $OpenZeppelin$ بهره می‌برد[؟]. این قراردادها توسط متخصصان امنیت به صورت دقیق حسابرسی ($audited$) شده و پیاده‌سازی‌های استانداردی برای توکن‌هایی مانند $ERC-1155$ و مکانیزم‌هایی مانند

کنترل دسترسی ($AccessControl$) و توقف اضطراری ($Pausable$) ارائه می‌دهند که ریسک بروز آسیب‌پذیری‌های رایج را به حداقل می‌رساند.

- **فریم‌ورک توسعه و آزمون** $Foundry$: برای توسعه، کامپایل، استقرار و آزمون قرارداد هوشمند، از فریم‌ورک مدرن $Foundry$ استفاده شده است. برخلاف ابزارهای قدیمی‌تر مانند $Truffle$ که نیازمند نوشتن آزمون‌ها به زبان $JavaScript$ هستند، $Foundry$ به توسعه‌دهندگان اجازه می‌دهد تا آزمون‌های خود را مستقیماً به زبان $Solidity$ بنویسند (همانطور که در فایل $SupplyChainERC1155.t.sol$ مشهود است). این ویژگی، فرآیند آزمون را سریع‌تر، کارآمدتر و برای توسعه‌دهندگان $Solidity$ طبیعی‌تر می‌سازد. ابزارهای همراه آن مانند $Anvil$ (یک نود محلی برای توسعه) و $Forge$ (موتور آزمون)، چرخه توسعه را به شدت تسریع می‌کنند.

۳-۱-۲ توجیه انتخاب فناوری‌های لایه ذخیره‌سازی و کاربری

برای لایه‌هایی که مستقیماً با کاربر در تعامل هستند، انتخاب فناوری‌هایی که تجربه کاربری مدرن، سریع و امنی را فراهم کنند، در اولویت قرار داشته است.

- **ذخیره‌سازی** $Off-chain$ **با** $IPFS$ **و** $Pinata$: همانطور که در فصل قبل تشریح شد، برای حل چالش هزینه ذخیره‌سازی، از معماری ترکیبی استفاده می‌شود. $IPFS$ ($InterPlanetary\ File\ System$) به دلیل ماهیت غیرمتمرکز و آدرس‌دهی مبتنی بر محتوا، به عنوان راهکار ایده‌آل برای ذخیره‌سازی فراداده انتخاب شد. برای تضمین در دسترس بودن دائمی فایل‌ها، از یک سرویس پینینگ به نام $Pinata$ استفاده شده است (که در فایل $ipfs.ts$ پیکربندی شده[؟]) که نیاز به اجرای یک گره $IPFS$ توسط خود کاربر را مرتفع می‌سازد.

- **کتابخانه** $React$ **و ابزار ساخت** $Vite$: برای توسعه لایه کاربری، از کتابخانه محبوب $React$ استفاده شده است که به دلیل معماری مبتنی بر کامپوننت، مدیریت حالت قدرتمند و اکوسیستم وسیع، امکان ساخت رابط‌های کاربری پیچیده و در عین حال قابل نگهداری را فراهم می‌کند. ابزار ساخت $Vite$ نیز به دلیل سرعت بسیار بالا در فرآیندهای توسعه و ساخت نهایی پروژه، جایگزین ابزارهای قدیمی‌تر مانند $Create-React-App$ شده است.

- **کتابخانه** $Wagmi$ **برای تعامل با زنجیره بلوکی:** $Wagmi$ مجموعه‌ای از هوک‌های ($Hooks$) $React$ برای تعامل با اتریوم است. این کتابخانه، فرآیندهای پیچیده‌ای مانند اتصال به کیف پول، خواندن داده از قراردادهای هوشمند، ارسال تراکنش و مدیریت وضعیت شبکه را به شدت ساده‌سازی می‌کند. استفاده از $Wagmi$ (که در $main.tsx$[؟] و $wagmi.ts$[؟] پیکربندی شده) به توسعه‌دهنده اجازه می‌دهد تا به جای درگیر شدن با جزئیات سطح پایین پروتکل RPC، بر روی منطق اصلی برنامه تمرکز کند.

- **کتابخانه $TailwindCSS$ برای طراحی واسط کاربری:** برای استایل‌دهی، از رویکرد $utility-first$ کتابخانه $TailwindCSS$ استفاده شده است (فایل $index.css$[؟]). این رویکرد، به جای نوشتن فایل‌های CSS جداگانه، امکان استایل‌دهی سریع و مستقیم در خود کامپوننت‌ها را فراهم کرده و منجر به ایجاد یک سیستم طراحی منسجم و قابل نگهداری می‌شود.

این پشته فناوری مدرن و یکپارچه، زیربنای لازم برای ساخت یک سامانه قوی، امن و کاربرپسند را فراهم می‌آورد.

۳-۲ معماری کلان سامانه

سامانه پیشنهادی بر اساس یک معماری چندلایه طراحی شده است که در آن، هر لایه مسئولیت مشخصی را بر عهده دارد. این تفکیک مسئولیت‌ها، به توسعه، نگهداری و مقیاس‌پذیری سیستم در آینده کمک می‌کند. همانطور که در نمودار بلوکی پروپوزال نیز نشان داده شده[؟]، می‌توان سه لایه اصلی را برای این سیستم متصور شد. در ادامه، این معماری با جزئیات بیشتری تشریح شده و جریان داده در یک سناریوی کلیدی (ثبت محصول جدید) ردیابی می‌شود.

۳-۲-۱ معماری سه لایه سیستم

۱. **لایه زنجیره بلوکی** ($Layer\ Blockchain$): این لایه، هسته غیرمتمرکز و قابل اعتماد سیستم است که به آن «لایه اعتماد» ($Layer\ Trust$) نیز گفته می‌شود. این لایه مسئولیت‌های زیر را بر عهده دارد:

- تعریف و مدیریت هویت دیجیتال محصولات از طریق توکن‌های $ERC-1155$.
- اجرای منطق کسب‌وکار به صورت تغییرناپذیر از طریق قرارداد هوشمند.
- ثبت تاریخچه کامل و قابل حسابرسی تمام تراکنش‌های مالکیت.
- نگهداری ارجاع‌های امن (هش‌ها) به داده‌های خارج از زنجیره.
- مدیریت کنترل دسترسی و مجوزهای بازیگران مختلف شبکه.

این لایه، منبع حقیقت واحد و غیرقابل انکار سیستم است.

۲. **لایه ذخیره‌سازی خارج از زنجیره** ($Layer\ Storage\ Off-chain$): این لایه برای نگهداری داده‌های حجیم و غنی که ذخیره‌سازی آن‌ها بر روی زنجیره بلوکی اقتصادی نیست، به کار می‌رود. مسئولیت اصلی این لایه، ذخیره‌سازی فایل‌های فراداده محصولات (در فرمت $JSON$) و فایل‌های چندرسانه‌ای مرتبط (مانند تصاویر و اسناد) است. در این پروژه، این لایه با استفاده از شبکه غیرمتمرکز $IPFS$ پیاده‌سازی شده تا همراستا با فلسفه عدم تمرکز کل سیستم باشد.

۳. **لایه کاربری** ($Layer\ User$)**:** این لایه که به آن $Frontend$ یا لایه ارائه نیز گفته می‌شود، نقطه تعامل کاربران با سیستم است. این لایه مسئولیت‌های زیر را بر عهده دارد:

- ارائه واسط‌های کاربری گرافیکی (GUI) ساده و کاربرپسند برای نقش‌های مختلف (داشبورد ادمین، داشبورد مشتری).
- جمع‌آوری داده‌ها از کاربران (مانند اطلاعات محصول جدید).
- تعامل با لایه ذخیره‌سازی $Off-chain$ برای آپلود و بازیابی فراداده.
- تعامل با کیف پول دیجیتال کاربر (مانند $MetaMask$) برای امضای تراکنش‌ها.
- ساخت و ارسال تراکنش‌ها به لایه زنجیره بلوکی.
- خواندن داده‌ها از زنجیره بلوکی و نمایش آن‌ها به صورت قابل فهم برای کاربر.

جریان داده در سناریوی ثبت محصول جدید

برای درک بهتر تعامل بین این سه لایه، فرآیند ثبت یک محصول جدید را به صورت گام به گام دنبال می‌کنیم:

۱. **شروع در لایه کاربری:** یک کاربر با نقش «تولیدکننده» ($MANUFACTURER_ROLE$) وارد داشبورد ادمین شده و فرم «ایجاد محصول جدید» را با اطلاعاتی مانند نام، شماره سریال، و فایل‌های تصویری پر می‌کند (صفحه $CreateProduct.tsx$).

۲. **تعامل با لایه ذخیره‌سازی** $Off-chain$: پس از فشردن دکمه ثبت، برنامه کاربردی ابتدا با لایه $Off-chain$ تعامل می‌کند. منطق موجود در $ipfs.ts$، فایل‌های تصویری و فراداده محصول را در شبکه $IPFS$ بارگذاری کرده و یک شناسه محتوای منحصربه‌فرد (CID) برای فایل فراداده دریافت می‌کند.

۳. **آماده‌سازی تراکنش در لایه کاربری:** برنامه کاربردی سپس هش $Keccak256$ فراداده را به صورت محلی محاسبه می‌کند. سپس یک تراکنش برای فراخوانی تابع $registerProduct$ در قرارداد هوشمند آماده می‌کند. این تراکنش شامل پارامترهایی مانند نام، شماره سریال، CID دریافت شده از $IPFS$ (به عنوان $metadataUrl$) و هش $Keccak256$ محاسبه‌شده است.

۴. **امضای تراکنش:** لایه کاربری، از طریق کتابخانه $Wagmi$، از کیف پول کاربر می‌خواهد تا این تراکنش را امضا کند. این امضا با استفاده از کلید خصوصی کاربر انجام شده و اثبات می‌کند که درخواست واقعاً از طرف او ارسال شده است.

۵. **ارسال به لایه زنجیره بلوکی:** پس از امضا، تراکنش به یک گره در شبکه زنجیره بلوکی ارسال می‌شود.

۶. **اجرا در لایه زنجیره بلوکی:** قرارداد هوشمند، تابع $registerProduct$ را اجرا می‌کند. این تابع، پس از بررسی مجوز کاربر، یک توکن $ERC-1155$ جدید می‌سازد، اطلاعات و هش‌ها را در متغیرهای حالت خود ذخیره می‌کند و یک رویداد $ProductRegistered$ را منتشر می‌نماید.

۷. **بازخورد به لایه کاربری:** لایه کاربری منتظر تأیید تراکنش در شبکه می‌ماند. پس از تأیید، یک پیام موفقیت به کاربر نمایش داده شده (با استفاده از $react-hot-toast$[؟]) و او به داشبورد ادمین هدایت می‌شود، جایی که محصول جدید ثبت‌شده اکنون قابل مشاهده است.

این جریان کار نشان می‌دهد که چگونه این سه لایه به صورت هماهنگ با یکدیگر کار می‌کنند تا یک فرآیند پیچیده را به یک تجربه کاربری ساده و امن تبدیل نمایند.

۳-۳ پیاده‌سازی لایه زنجیره بلوکی ($Backend$)

این بخش به تشریح عمیق و خط به خط قرارداد هوشمند $SupplyChainERC1155.sol$ می‌پردازد که هسته اصلی منطق و امنیت کل سامانه را تشکیل می‌دهد.

۳-۳-۱ ساختار کلی و وراثت قرارداد

قرارداد هوشمند این پروژه، با ارث‌بری از چندین قرارداد استاندارد و حسابرسی‌شده از کتابخانه $OpenZeppelin$، بر پایه‌ای محکم و امن بنا شده است. این رویکرد، ضمن کاهش حجم کدهای نوشته شده، از بهترین شیوه‌های ($practices\ best$) امنیتی بهره می‌برد.

```
contract SupplyChainERC1155 is ERC1155, AccessControl, Pausable,
    ERC1155Supply {
    // ...
}
```

برنامه ۳-۱: تعریف و وراثت قرارداد هوشمند

- $ERC1155$: این قرارداد پایه، تمام منطق اصلی استاندارد چند-توکنی $ERC-1155$ را پیاده‌سازی می‌کند، از جمله توابع $balanceOf$، $mint$، $burn$ و $safeTransferFrom$[؟].
- $AccessControl$: این قرارداد یک مکانیزم قدرتمند و انعطاف‌پذیر برای مدیریت کنترل دسترسی مبتنی بر نقش ($RBAC$ - $Control\ Access\ Role-Based$) فراهم می‌کند. این ماژول به ما اجازه می‌دهد تا نقش‌های مختلفی تعریف کرده و دسترسی به توابع حساس را تنها به نقش‌های مجاز محدود کنیم[؟].

- $Pausable$: این ماژول یک قابلیت ایمنی حیاتی را اضافه می‌کند: امکان توقف اضطراری تمام فعالیت‌های اصلی قرارداد (مانند انتقالات) توسط یک مدیر. این ویژگی می‌تواند در صورت کشف یک آسیب‌پذیری، از بروز خسارات بیشتر جلوگیری کند[؟].
- $ERC1155Supply$: این یک افزونه برای $ERC-1155$ است که تعداد کل توکن‌های موجود از هر نوع ($totalSupply$) را ردیابی می‌کند. این قابلیت برای حسابرسی و نظارت بر کل سیستم مفید است[؟].

۳-۳-۲ نقش‌ها و کنترل دسترسی ($Control\ Access$)

یکی از مهم‌ترین جنبه‌های یک سیستم زنجیره تأمین، تعریف دقیق نقش‌ها و مسئولیت‌های هر یک از بازیگران است. قرارداد هوشمند این پروژه با استفاده از ماژول $AccessControl$، چهار نقش اصلی را تعریف و مدیریت می‌کند:

```
bytes32 public constant MANUFACTURER_ROLE = keccak256("MANUFACTURER_ROLE
    ");
bytes32 public constant DISTRIBUTOR_ROLE = keccak256("DISTRIBUTOR_ROLE");
bytes32 public constant RETAILER_ROLE = keccak256("RETAILER_ROLE");
bytes32 public constant CUSTOMS_ROLE = keccak256("CUSTOMS_ROLE");
```

برنامه ۳-۲: تعریف نقش‌ها در قرارداد هوشمند

- $MANUFACTURERROLE$: این نقش مجوز ثبت (ساخت) محصولات جدید را دارد. تنها آدرس‌هایی این نقش به آن‌ها اعطا شده، می‌توانند تابع $registerProduct$ را فراخوانی کنند.
- $DISTRIBUTORROLE$ و $RETAILER_ROLE$: اگرچه در نسخه فعلی قرارداد، توابع خاصی برای این نقش‌ها تعریف نشده، اما وجود آن‌ها زیرساخت لازم برای افزودن منطق‌های تجاری آینده (مانند ثبت مراحل توزیع خاص) را فراهم می‌کند.
- $CUSTOMSROLE$: این نقش، مجوز ابطال یا از بین بردن یک محصول (مثلاً به دلیل شناسایی به عنوان کالای تقلبی یا تاریخ مصرف گذشته) را دارد. این نقش، کنترل تابع حساس $destroyProduct$ را در اختیار دارد.
- $DEFAULTADMIN_ROLE$: این نقش که در سازنده ($constructor$) به آدرس مستقرکننده قرارداد اعطا می‌شود[؟]، بالاترین سطح دسترسی را دارد و می‌تواند نقش‌های دیگر را به سایر آدرس‌ها اعطا یا از آن‌ها سلب کند (از طریق توابعی مانند $grantRole$ و $revokeRole$).

استفاده از اصلاح‌گر $onlyRole$ در توابع حساس، این سیاست‌های دسترسی را به صورت قاطع اعمال می‌کند. برای مثال، تعریف تابع $registerProduct$ تضمین می‌کند که هیچ بازیگر دیگری جز یک تولیدکننده تأییدشده، قادر به افزودن محصول به سیستم نخواهد بود:

```
function registerProduct(
    // ...
) external onlyRole(MANUFACTURER_ROLE) whenNotPaused returns (uint256) {
    // ...
}
```

برنامهٔ ۳-۳: استفاده از $onlyRole$ برای حفاظت از تابع

۳-۳-۳ ساختارهای داده اصلی ($Structures\ Data\ Core$)

قرارداد هوشمند از چندین ساختار داده و نگاشت ($mapping$) برای ذخیره‌سازی وضعیت سیستم به صورت کارآمد و ساختاریافته استفاده می‌کند.

۳-۳-۴ ساختار داده محصول ($Struct\ Product$)

این ساختار، شناسنامه دیجیتال هر محصول را تعریف می‌کند و تمام اطلاعات کلیدی آن را در خود جای داده است:

```
struct Product {
    uint256 id;
    string name;
    string category;
    string serialNumber;
    uint256 productionDate;
    string geographicalOrigin;
    bytes32 metadataHash; // Keccak256 hash of metadata content
    string metadataUrl;  // IPFS URL for full metadata
    address manufacturer;
    bool exists;
}
```

برنامهٔ ۳-۴: تعریف $ProductStruct$

هر یک از فیلدهای این ساختار با دقت انتخاب شده است. id شناسه توکن $ERC-1155$ است. $metadataHash$ و $metadataUrl$ هسته اصلی مکانیزم اعتبارسنجی را تشکیل می‌دهند[؟، ؟]. فیلد $exists$ به ما اجازه می‌دهد تا یک محصول را بدون حذف کامل اطلاعات آن از تاریخچه، به عنوان «باطل شده» علامت‌گذاری کنیم که برای اهداف حسابرسی بسیار مهم است. این ساختارها در یک نگاشت اصلی ذخیره می‌شوند:

```
mapping(uint256 => Product) public products;
```

برنامه□ ۳-۵: نگاشت اصلی محصولات

۳-۳-۵ ساختار داده تاریخچه مالکیت ($Struct\ OwnershipRecord$)

برای ردیابی کامل زنجیره مالکیت، از این ساختار استفاده می‌شود:

```
struct OwnershipRecord {
    address owner;
    uint256 timestamp;
    string transferReason; // "manufactured", "sold", etc.
}
```

برنامه□ ۳-۶: تعریف □□□□□□□□□□□□□□□□ □□□□□□

برای هر محصول، آرایه‌ای از این رکوردها نگهداری می‌شود که تاریخچه کامل انتقالات آن را از ابتدا تا کنون ثبت می‌کند[؟].

۳-۳-۶ مدیریت چرخه حیات محصول

قرارداد هوشمند، توابع اصلی برای مدیریت چرخه کامل حیات یک محصول را فراهم می‌کند.

ثبت محصول (تابع $registerProduct$)

این تابع، نقطه ورود محصولات به اکوسیستم زنجیره بلوکی است.

```
function registerProduct(
    address to,
    string memory name,
    string memory category,
    string memory serialNumber,
```

```
    string memory geographicalOrigin,
    string memory metadataUrl,
    bytes32 metadataHash,
    uint256 amount
) external onlyRole(MANUFACTURER_ROLE) whenNotPaused returns (uint256) {
    uint256 tokenId = nextTokenId;
    nextTokenId++;

    metadataRegistry[tokenId] = metadataUrl;
    urlToTokenId[metadataUrl] = tokenId;

    products[tokenId] = Product({
        id: tokenId,
        name: name,
        category: category,
        serialNumber: serialNumber,
        productionDate: block.timestamp,
        geographicalOrigin: geographicalOrigin,
        metadataHash: metadataHash,
        metadataUrl: metadataUrl,
        manufacturer: msg.sender,
        exists: true
    });

    _mint(to, tokenId, amount, "");

    ownershipHistory[tokenId].push(OwnershipRecord({
        owner: to,
        timestamp: block.timestamp,
        transferReason: "manufactured"
    }));

```

```
    emit ProductRegistered(tokenId, msg.sender, metadataHash, metadataUrl
    );

    return tokenId;
}
```

برنامه ۳-۷: کد کامل تابع $registerProduct$

تحلیل گام به گام این تابع:

۱. **بررسی مجوزها:** اصلاح‌گرهای $onlyRole$ و $whenNotPaused$ ابتدا بررسی می‌کنند که آیا فرستنده تراکنش نقش تولیدکننده را دارد و آیا قرارداد در حالت فعال است.

۲. **تخصیص شناسه یکتا:** به جای استفاده از هش‌های پیچیده، قرارداد از یک شمارنده ساده و کارآمد به نام $nextTokenId$ برای تخصیص یک شناسه عددی منحصربه‌فرد و قابل پیش‌بینی به هر محصول جدید استفاده می‌کند[؟، ؟].

۳. **ثبت فراداده:** آدرس $IPFS$ فراداده و هش آن در نگاشت‌های مربوطه ($metadataRegistry$ و $products$) ذخیره می‌شوند[؟، ؟].

۴. **ساخت توکن:** تابع $_mint$ از استاندارد $ERC-1155$ فراخوانی شده و توکن‌های جدید را به آدرس گیرنده (to) با تعداد ($amount$) مشخص شده، ایجاد می‌کند[؟].

۵. **ثبت در تاریخچه:** اولین رکورد در تاریخچه مالکیت محصول، با دلیل «ساخته شده» ($manufactured$) ثبت می‌شود.

۶. **انتشار رویداد:** رویداد $ProductRegistered$ منتشر می‌شود تا برنامه‌های کاربردی خارج از زنجیره (مانند $Frontend$) از ثبت محصول جدید مطلع شوند.

ابطال محصول (تابع $destroyProduct$)

این تابع برای حذف منطقی یک محصول از چرخه فعال زنجیره تأمین به کار می‌رود.

```
function destroyProduct(
    uint256 tokenId,
    string memory reason
) external onlyRole(CUSTOMS_ROLE) {
    OwnershipRecord[] memory history = ownershipHistory[tokenId];
    address currentOwner = history[history.length - 1].owner;
```

```
    _burn(currentOwner, tokenId, 1);

    ownershipHistory[tokenId].push(OwnershipRecord({
        owner: address(0),
        timestamp: block.timestamp,
        transferReason: reason
    }));

    products[tokenId].exists = false;
}
```

برنامهٔ ۳-۸: کد تابع $destroyProduct$

این تابع ابتدا مالک فعلی توکن را از تاریخچه استخراج می‌کند (زیرا یک توکن $ERC-1155$ می‌تواند چندین مالک داشته باشد، اما در منطق این پروژه، هر محصول غیرمثلی یک مالک دارد). سپس تابع $burn$ را برای سوزاندن و از بین بردن توکن فراخوانی می‌کند. در نهایت، یک رکورد جدید در تاریخچه با مالک $address(0)$ (یک آدرس استاندارد برای نمایش توکن سوخته شده) ثبت کرده و فیلد $exists$ محصول را به $false$ تغییر می‌دهد[؟].

۳-۳-۷ مدیریت مالکیت و تاریخچه

یکی از پیچیده‌ترین و در عین حال نوآورانه‌ترین بخش‌های این قرارداد، نحوه ردیابی و بازیابی کارآمد محصولات تحت مالکیت هر کاربر است.

سازوکار ردیابی مالکیت در تابع $update$

قراردادهای $ERC-1155$ دارای یک تابع داخلی و محوری به نام $update$ هستند که تمام منطق انتقال، ساخت و سوزاندن توکن‌ها از آن عبور می‌کند. این پروژه، با بازنویسی ($override$) این تابع، یک قلاب ($hook$) هوشمندانه برای ردیابی مالکیت ایجاد کرده است.

```
function _update(
    address from,
    address to,
    uint256[] memory ids,
    uint256[] memory amounts
```

```
)۶ internal override(ERC1155, ERC1155Supply) {
۷   // ...
۸   for (uint256 i = 0; i < ids.length; i++) {
۹       uint256 tokenId = ids[i];
۱۰
۱۱      if (from != address(0)) {
۱۲          if (balanceOf(from, tokenId) == amounts[i]) {
۱۳              _removeFromOwnedProducts(from, tokenId);
۱۴          }
۱۵      }
۱۶
۱۷      if (to != address(0)) {
۱۸          if (balanceOf(to, tokenId) == 0) {
۱۹              _addToOwnedProducts(to, tokenId);
۲۰          }
۲۱      }
۲۲  }
۲۳
۲۴  super._update(from, to, ids, amounts);
۲۵
۲۶  // ...
}۲۷
```

برنامه ۳-۹: بازنویسی تابع $update$ برای ردیابی مالکیت

قبل از اجرای منطق اصلی انتقال در $super._update$، این تابع بررسی می‌کند که آیا این انتقال، موجودی فرستنده را صفر می‌کند یا موجودی گیرنده را از صفر بیشتر می‌کند. در این صورت، توابع کمکی $_removeFromOwnedProducts$ و $_addToOwnedProducts$ را برای به‌روزرسانی لیست مالکیت کاربران فراخوانی می‌کند.

بهینه‌سازی بازیابی محصولات با الگوریتم $Swap-and-Pop$

نگهداری یک لیست پویا از محصولات هر کاربر در یک آرایه، چالش حذف یک عنصر از وسط آرایه را به همراه دارد که عملیاتی پرهزینه در $Solidity$ است. این قرارداد برای حل این مشکل از یک الگوریتم

بهینه‌سازی شده به نام «تعویض و حذف» ($Swap - and - Pop$) استفاده می‌کند.

در این الگوریتم، برای حذف یک عنصر از وسط آرایه، به جای جابجا کردن تمام عناصر بعدی، آخرین عنصر آرایه به جای عنصر حذفی منتقل شده و سپس آخرین خانه آرایه حذف می‌شود[؟]. این عملیات، هزینه حذف را به یک مقدار ثابت ($O(1)$) کاهش داده و کارایی سیستم را به شدت افزایش می‌دهد.

۳-۳-۸ توابع خواندنی و بازیابی داده‌ها

برای اینکه لایه کاربری بتواند داده‌ها را به صورت بهینه از قرارداد بخواند، چندین تابع $view$ طراحی شده است:

- $getOwnedProductsCount(addressowner)$

).().□□□□□□□□□□□(:$getOwnedProductsBatch(addressowner, uint256 startIndex, uint256 count)$.:

- $getProductsBatch(uint256 startId, uint256 endId)$

:.. این توابع دسته‌ای، از ارسال درخواست‌های متعدد به شبکه جلوگیری کرده و عملکرد لایه کاربری را به طور قابل توجهی بهبود می‌بخشند.

۳-۴ پیاده‌سازی لایه ذخیره‌سازی خارج از زنجیره ($Off-chain$)

این بخش به تشریح کامل منطق پیاده‌سازی شده در فایل $gpt/src - front/lib/ipfs.ts$ می‌پردازد که مسئولیت مدیریت تمام تعاملات با لایه ذخیره‌سازی غیرمتمرکز را بر عهده دارد.

۳-۴-۱ انتخاب $IPFS$ و سرویس پینینگ $Pinata$

همانطور که پیش‌تر ذکر شد، انتخاب $IPFS$ به دلیل ماهیت غیرمتمرکز و آدرس‌دهی مبتنی بر محتوا، یک انتخاب استراتژیک برای همسویی با اهداف پروژه بوده است. با این حال، داده‌ها در شبکه $IPFS$ تنها تا زمانی در دسترس هستند که حداقل یک گره در شبکه، آن داده را «پین» کرده و نگهداری کند. اجرای یک گره $IPFS$ به صورت ۲۴/۷ برای هر کاربر، عملی نیست. برای حل این مشکل، از یک سرویس پینینگ به نام $Pinata$ استفاده شده است. $Pinata$ یک پلتفرم ابری است که در ازای دریافت هزینه، تضمین می‌کند که فایل‌های آپلود شده توسط کاربر، برای همیشه در شبکه $IPFS$ پین شده و در دسترس باقی بمانند. تمام توابع این بخش از طریق API این سرویس عمل می‌کنند[؟].

۳-۴-۲ فرآیند آپلود فایل و فراداده

کتابخانه $ipfs.ts$ دو تابع اصلی برای آپلود انواع مختلف داده به $Pinata$ فراهم می‌کند: $uploadFileToIPFS$ برای فایل‌های باینری (مانند تصاویر و اسناد) و $uploadJSONToIPFS$ برای فایل‌های فراداده با فرمت $JSON$. □□□□□□□□□

```
export async function uploadFileToIPFS(file: File): Promise<string> {
// ... check for API keys ...

const formData = new FormData();
formData.append('file', file);

const response = await fetch(IPFS_UPLOAD_ENDPOINT, {
  method: 'POST',
  headers: {
    'pinata_api_key': PINATA_API_KEY,
    'pinata_secret_api_key': PINATA_SECRET_KEY,
  },
  body: formData,
});

// ... error handling ...

const result = await response.json();
return result.IpfsHash;
}
```

این تابع یک فایل را دریافت کرده، آن را در یک شیء $FormData$ قرار می‌دهد و با استفاده از متد $POST$ به نقطه پایانی ($Endpoint$) API پیناتا ارسال می‌کند. هدرهای مربوط به کلیدهای API نیز برای احراز هویت در درخواست گنجانده شده‌اند[؟]. در صورت موفقیت، تابع هش $IPFS$ (یا همان CID) فایل آپلود شده را برمی‌گرداند.

۳-۴-۳ ساخت و اعتبارسنجی فراداده

منطق اصلی این ماژول در تابع $uploadProductMetadata$ قرار دارد که یک فرآیند چند مرحله‌ای را ارکسترا می‌کند:

۱. **ساخت شیء فراداده:** ابتدا تابع $createProductMetadata$ فراخوانی می‌شود. این تابع، داده‌های خام دریافت شده از فرم کاربر را به یک ساختار $JSON$ استاندارد و غنی تبدیل می‌کند. این ساختار شامل «ویژگی‌ها» ($attributes$) است که با استاندارد فراداده NFT در پلتفرم‌هایی مانند $OpenSea$ سازگار است[؟]. همچنین، تمام فایل‌های تصویری و اسناد به صورت موازی در $IPFS$ آپلود شده و لینک آن‌ها در ساختار $JSON$ قرار می‌گیرد[؟، ؟].

۲. **آپلود فراداده نهایی:** شیء $JSON$ نهایی، خود با استفاده از تابع $uploadJSONToIPFS$ در $IPFS$ بارگذاری شده و CID اصلی آن به دست می‌آید.

۳. **محاسبه هش** $On-chain$: سپس، تابع $calculateKeccak256Hash$ فراخوانی می‌شود. این تابع، برای اطمینان از تطابق کامل با نحوه محاسبه هش در $Solidity$، از کتابخانه

$$ethers$$

برای اعمال الگوریتم $Keccak256$ بر روی نسخه رشته‌ای شده فراداده استفاده می‌کند[؟].

خروجی نهایی این تابع، یک شیء است که شامل آدرس کامل $IPFS$ ($metadataUrl$) و هش $Keccak256$ ($metadataHash$) است. این دو مقدار، دقیقاً همان ورودی‌هایی هستند که برای تابع $registerProduct$ در قرارداد هوشمند ارسال می‌شوند.

در نهایت، تابع $verifyMetadataIntegrity$ منطق اعتبارسنجی سمت کاربر را پیاده‌سازی می‌کند[؟]. این تابع، یک آدرس URL و یک هش مورد انتظار را دریافت کرده، محتوا را از URL دانلود می‌کند، هش آن را مجدداً محاسبه کرده و با هش مورد انتظار مقایسه می‌نماید تا یکپارچگی داده را تأیید کند. این تابع، آینه سمت کاربرِ منطق امنیتی است که در قرارداد هوشمند طراحی شده است.

۳-۵ پیاده‌سازی لایه کاربری ($Frontend$)

لایه کاربری، نقطه نهایی تماس کاربر با سیستم و ویترین تمام قابلیت‌های پیچیده لایه‌های زیرین است. این لایه با استفاده از پشته فناوری مدرن $React$، $Vite$، $Wagmi$ و $TailwindCSS$ پیاده‌سازی شده تا یک تجربه کاربری سریع، واکنش‌گرا و بصری را ارائه دهد.

۳-۵-۱ پروژه‌بندی و تنظیمات اولیه

ساختار پروژه در پوشه $src - front$ به صورت ماژولار و بر اساس مسئولیت‌ها سازماندهی شده است:

- */components*: شامل کامپوننت‌های قابل استفاده مجدد مانند دکمه‌ها، کارت‌ها و هدر و فوتر ($Layout.tsx$).
- */pages*: شامل کامپوننت‌های اصلی که هر کدام یک صفحه کامل از برنامه را نمایندگی می‌کنند (مانند $AdminDashboard.tsx$ و $ClientDashboard.tsx$).
- */lib*: شامل منطق‌های کمکی و پیکربندی‌های اصلی، از جمله تنظیمات اتصال به زنجیره بلوکی ($wagmi.ts$)، تعامل با $IPFS$ ($ipfs.ts$) و تعریف ABI قرارداد ($contract.ts$).

نقطه ورود اصلی برنامه، فایل $main.tsx$ است که برنامه $React$ را رندر کرده و آن را با فراهم‌کننده‌های ($Providers$) لازم احاطه می‌کند[؟]. $WagmiProvider$ وضعیت اتصال به کیف پول و شبکه را در کل برنامه مدیریت می‌کند و $QueryClientProvider$ برای کش کردن و مدیریت بهینه درخواست‌های داده به کار می‌رود.

۳-۵-۲ مدیریت اتصال به کیف پول و شبکه

برنامه قبل از هر چیز، وضعیت اتصال کیف پول کاربر را بررسی می‌کند. در فایل $App.tsx$، هوک $useAccount$ از کتابخانه $Wagmi$ برای این منظور استفاده شده است[؟]. اگر کاربر متصل نباشد، تنها کامپوننت $ConnectWallet$ نمایش داده می‌شود که یک رابط کاربری ساده برای انتخاب و اتصال به کیف پول‌های مختلف (مانند $MetaMask$ یا از طریق $WalletConnect$) فراهم می‌کند[؟، ؟].

پس از اتصال، کامپوننت $Layout.tsx$ به عنوان پوسته اصلی برنامه عمل می‌کند. این کامپوننت با استفاده از هوک‌های $useChainId$ و $useSwitchChain$، به صورت فعال شبکه متصل شده کاربر را شناسایی کرده و در صورتی که شبکه پشتیبانی نشود، یک هشدار به کاربر نمایش می‌دهد و امکان تغییر شبکه را فراهم می‌آورد[؟، ؟]. این مدیریت فعال شبکه، از بروز خطاهای ناشی از تعامل با یک شبکه اشتباه جلوگیری می‌کند.

۳-۵-۳ کامپوننت‌ها و صفحات اصلی

در ادامه، به تحلیل پیاده‌سازی چند صفحه کلیدی در برنامه می‌پردازیم.

داشبورد ادمین ($AdminDashboard.tsx$)

این صفحه برای کاربرانی با نقش مدیریتی (مانند تولیدکننده) طراحی شده و نمای کلی از تمام محصولات ثبت‌شده در سیستم را ارائه می‌دهد.

- **بازیابی داده‌ها:** این صفحه از تابع $getProductsBatch$ در قرارداد هوشمند برای بازیابی محصولات به صورت صفحه‌بندی شده استفاده می‌کند[؟]. برای نمایش آخرین محصولات ابتدا، این تابع با شناسه‌هایی از $nextTokenId - 1$ به سمت عقب فراخوانی می‌شود. این رویکرد، ضمن نمایش اطلاعات مرتبط‌تر به ادمین، از بازیابی یکباره حجم عظیمی از داده که می‌تواند منجر به کندی برنامه شود، جلوگیری می‌کند.
- **واسط کاربری:** داده‌ها در یک جدول جامع با قابلیت جستجو و فیلتر بر اساس دسته‌بندی و وضعیت نمایش داده می‌شوند[؟]. هر ردیف، شامل اقدامات مدیریتی مانند «مشاهده جزئیات» یا «ابطال محصول» است.
- **نقطه ورود برای ایجاد محصول:** این صفحه شامل یک دکمه برجسته برای هدایت کاربر به صفحه «ایجاد محصول جدید» است[؟].

داشبورد مشتری ($ClientDashboard.tsx$)

این صفحه، نمایی شخصی‌سازی‌شده برای مصرف‌کنندگان یا مالکان محصولات است.

- **بازیابی داده‌های اختصاصی:** برخلاف داشبورد ادمین، این صفحه از توابع بهینه‌سازی‌شده $getOwnedProductsCount$ و $getOwnedProductsBatch$ برای بازیابی محصولاتی که **تنها متعلق به آدرس متصل شده کاربر** هستند، استفاده می‌کند[؟، ؟]. این امر، هم از نظر حفظ حریم خصوصی و هم از نظر کارایی، بسیار بهینه‌تر است.
- **واسط کاربری:** محصولات در قالب کارت‌های بصری نمایش داده می‌شوند که اطلاعات کلیدی هر محصول را به صورت خلاصه نشان می‌دهد. هر کارت، شامل دکمه‌هایی برای «مشاهده جزئیات کامل» و «انتقال مالکیت» است[؟، ؟].

این تفکیک بین داشبوردها، نشان‌دهنده طراحی دقیقی است که تجربه کاربری را برای هر نقش، متناسب با نیازهای آن، بهینه کرده است.

صفحه ثبت محصول ($CreateProduct.tsx$)

این صفحه، پیچیده‌ترین فرم برنامه و نقطه اوج تعامل بین تمام لایه‌های سیستم است.

- **مدیریت حالت و اعتبارسنجی:** حالت فرم با استفاده از هوک $useState$ از $React$ مدیریت می‌شود[؟]. قبل از ارسال، تابع $validateForm$ تمام فیلدهای ضروری را بررسی کرده و از صحت ورودی‌ها اطمینان حاصل می‌کند[؟].
- **ارکستراسیون فرآیند در $handleSubmit$:** تابع $handleSubmit$ که پس از فشردن دکمه نهایی فراخوانی می‌شود، به عنوان یک ارکستراتور عمل می‌کند[؟]. این تابع ابتدا $uploadProductMetadata$

را برای انجام عملیات $IPFS$ فراخوانی کرده و منتظر دریافت $metadataUrl$ و $metadataHash$ می‌ماند. سپس، با استفاده از هوک $useWriteContract$ از $Wagmi$، تراکنش نهایی را برای فراخوانی $registerProduct$ در قرارداد هوشمند آماده و ارسال می‌کند.

- **بازخورد به کاربر:** در طول این فرآیند چند مرحله‌ای، وضعیت به صورت مداوم با استفاده از نوتیفیکیشن‌های $toast$ به کاربر اطلاع داده می‌شود (مثلاً «در حال آپلود در $IPFS$...»، «در انتظار تأیید تراکنش...»)[؟]. این بازخورد آنی، تجربه کاربری را به شدت بهبود بخشیده و از سردرگمی کاربر جلوگیری می‌کند.

صفحه جزئیات محصول ($ProductDetail.tsx$) این صفحه، شناسنامه دیجیتال کامل یک محصول را نمایش می‌دهد.

- **بازیابی داده‌های جامع:** این صفحه با استفاده از شناسه توکن ($tokenId$) دریافت شده از URL، تمام اطلاعات مربوط به محصول را از نگاشت $products$ در قرارداد هوشمند می‌خواند. همچنین، با استفاده از $metadataUrl$، فراداده کامل را از $IPFS$ بازیابی کرده و نمایش می‌دهد.
- **قابلیت اعتبارسنجی:** این صفحه شامل بخش «تأیید فراداده» است که به کاربر اجازه می‌دهد با فشردن یک دکمه، فرآیند $verifyMetadataIntegrity$ را فعال کرده و به صورت آنی، از یکپارچگی اطلاعات محصول اطمینان حاصل کند[؟].
- **نمایش تاریخچه و QR کد:** تاریخچه کامل مالکیت و یک کد QR قابل اسکن که حاوی اطلاعات کلیدی محصول برای اشتراک‌گذاری آسان است نیز در این صفحه نمایش داده می‌شود[؟].

در مجموع، لایه کاربری این پروژه، نمونه‌ای کامل از یک برنامه غیرمتمرکز ($dApp$) مدرن است که با انتزاع پیچیدگی‌های زنجیره بلوکی، یک تجربه کاربری روان، امن و قابل فهم را برای تمام کاربران، صرف نظر از دانش فنی آن‌ها، فراهم می‌آورد.

۳-۶ محیط توسعه و راهبرد آزمون

با توجه به ماهیت تغییرناپذیر و حساس قراردادهای هوشمند که مستقیماً با دارایی‌های دیجیتال سروکار دارند، اتخاذ یک راهبرد آزمون جامع و دقیق، امری حیاتی و غیرقابل چشم‌پوشی است. یک آسیب‌پذیری کوچک در کد می‌تواند منجر به خسارات جبران‌ناپذیر شود. از این رو، این پروژه یک رویکرد چندلایه برای تضمین کیفیت و امنیت کد، هم در لایه $Backend$ و هم در لایه $Frontend$، به کار گرفته است.

۳-۶-۱ پشته توسعه و آزمون $Backend$ (فریم‌ورک $Foundry$)

همانطور که پیش‌تر ذکر شد، برای توسعه قرارداد هوشمند از فریم‌ورک مدرن $Foundry$ استفاده شده است. این انتخاب، تأثیر مستقیمی بر راهبرد آزمون پروژه داشته است.

معرفی اجزای $Foundry$

$Foundry$ یک جعبه ابزار سریع، قابل حمل و ماژولار برای توسعه برنامه‌های مبتنی بر اتریوم است که به زبان $Rust$ نوشته شده و شامل سه ابزار اصلی است:

- $Forge$: موتور اصلی کامپایل، آزمون و استقرار قراردادهای هوشمند است. بزرگ‌ترین مزیت آن، امکان نوشتن آزمون‌ها به زبان $Solidity$ است.
- $Anvil$: یک گره زنجیره بلوکی محلی برای توسعه و آزمون است که به صورت آنی و با قابلیت‌های پیشرفته‌ای مانند فورک کردن شبکه‌های عمومی، اجرا می‌شود.
- $Cast$: یک ابزار خط فرمان برای انجام فراخوانی‌های RPC و تعامل مستقیم با قراردادهای هوشمند مستقر شده است.

تحلیل فایل آزمون $SupplyChainERC1155.t.sol$

فایل آزمون ارائه شده[؟]، یک نمونه کامل از راهبرد آزمون به کار رفته برای تضمین صحت عملکرد قرارداد هوشمند است.

ساختار آزمون و تابع $setUp$: هر مجموعه آزمون در $Foundry$، یک قرارداد است که از قرارداد $Test$ از کتابخانه $forge-std$ ارث‌بری می‌کند. تابع $setUp$ یک تابع ویژه است که قبل از اجرای هر تابع آزمون، یک بار اجرا می‌شود. در این پروژه، از این تابع برای استقرار یک نسخه تازه از قرارداد $SupplyChainERC1155$ و اعطای نقش‌های اولیه به آدرس‌های آزمایشی استفاده شده است. این کار تضمین می‌کند که هر آزمون در یک محیط ایزوله و تمیز اجرا می‌شود.

```
function setUp() public {
    vm.prank(admin);
    supplyChain = new SupplyChainERC1155();

    vm.startPrank(admin);
    supplyChain.grantManufacturerRole(manufacturer1);
    supplyChain.grantDistributorRole(distributor);
    // ...
```

```
    vm.stopPrank();
}
```

برنامهٔ ۳-۱۰: تابع $setUp$ برای آماده‌سازی محیط آزمون[؟، ؟]

استفاده از ابزارهای شبیه‌سازی ($Cheatcodes$): $Foundry$ مجموعه‌ای قدرتمند از توابع ویژه به نام $Cheatcodes$ را از طریق یک متغیر سراسری به نام vm در اختیار آزمون‌ها قرار می‌دهد. این ابزارها امکان شبیه‌سازی دقیق شرایط مختلف شبکه را فراهم می‌کنند. در فایل آزمون این پروژه، از این ابزارها به صورت گسترده استفاده شده است:

- $vm.prank(address)$ و $vm.startPrank(address)$: این دستورات به آزمون اجازه می‌دهند تا هویت خود را جعل کرده و تراکنش بعدی (یا تراکنش‌های بعدی) را از طرف یک آدرس مشخص ارسال کند. این برای آزمودن منطق کنترل دسترسی حیاتی است. برای مثال، در آزمون $testRegisterProductOnlyManufacturer$[؟]، با استفاده از $vm.prank(distributor)$، تلاش برای ثبت محصول از طرف یک آدرس فاقد نقش تولیدکننده شبیه‌سازی می‌شود.
- $vm.expectRevert()$: این دستور به آزمون اعلام می‌کند که انتظار دارد تراکنش بعدی با یک خطای مشخص ناموفق شود. این برای آزمودن اینکه آیا اصلاح‌گرهای حفاظتی مانند $onlyRole$ به درستی کار می‌کنند، ضروری است[؟].

پوشش جامع آزمون‌ها: فایل آزمون $SupplyChainERC1155.t.sol$ سناریوهای مختلفی را برای پوشش کامل منطق قرارداد، شبیه‌سازی می‌کند:

- **آزمون مسیر شاد** ($Path\ Happy$): تابع $testRegisterProduct$[؟] و $testTransferWithTax$[؟]، عملکرد صحیح توابع اصلی را در شرایط عادی بررسی می‌کنند. در این آزمون‌ها، پس از اجرای تابع، با استفاده از دستورات $assertEq$ و $assertTrue$، بررسی می‌شود که آیا وضعیت قرارداد (مانند موجودی توکن‌ها و تاریخچه مالکیت) به درستی به‌روز شده است.
- **آزمون کنترل دسترسی:** توابعی مانند $testRegisterProductOnlyManufacturer$[؟] و $testDestroyProductOnlyCustoms$[؟] تضمین می‌کنند که تنها کاربران دارای نقش صحیح می‌توانند توابع حساس را فراخوانی کنند.
- **آزمون سناریوی سرتاسری:** تابع $testCompleteSupplyChainFlow$[؟] یک سناریوی کامل را از ثبت محصول توسط تولیدکننده، انتقال به توزیع‌کننده، سپس به خرده‌فروش و در نهایت به مصرف‌کننده شبیه‌سازی می‌کند. این آزمون یکپارچه‌سازی، تضمین می‌کند که تمام اجزای قرارداد به درستی با یکدیگر کار می‌کنند.

- **آزمون اعتبارسنجی فراداده:** تابع $testVerifyProductMetadata$[؟] (مربوط به نسخه اولیه قرارداد)، هم با داده‌های صحیح و هم با داده‌های نادرست، تابع اعتبارسنجی را فراخوانی کرده و از صحت پاسخ آن اطمینان حاصل می‌کند.
- **آزمون حالت‌های حدی** ($Cases\ Edge$): توابعی مانند $testGetNonExistentProduct$[؟] و $testUnauthorizedTransfer$[؟]، رفتار سیستم را در شرایط غیرمنتظره یا تلاش برای اقدامات غیرمجاز، مورد سنجش قرار می‌دهند.

۳-۶-۲ راهبرد آزمون لایه کاربری ($Frontend$)

اگرچه فایل‌های آزمون مشخصی برای لایه کاربری ارائه نشده است، اما یک راهبرد آزمون استاندارد برای تضمین کیفیت برنامه $React$ شامل سه لایه آزمون است:

آزمون واحد ($UnitTesting$)

در این لایه، هر کامپوننت یا تابع کمکی به صورت ایزوله آزموده می‌شود. با استفاده از فریم‌ورک‌هایی مانند

$$Jest$$

و کتابخانه $Library\ Testing\ React$، می‌توان اطمینان حاصل کرد که هر کامپوننت به درستی و مطابق با ورودی‌های ($props$) مختلف، رندر می‌شود. برای مثال، می‌توان یک آزمون واحد برای کامپوننت $ProductCard$ نوشت تا بررسی کند که آیا اطلاعات محصول را به درستی نمایش می‌دهد و آیا دکمه «انتقال» تنها در صورتی که محصول $exists$ باشد، نمایش داده می‌شود.

آزمون یکپارچه‌سازی ($IntegrationTesting$)

این آزمون‌ها، تعامل بین چندین کامپوننت را بررسی می‌کنند. برای مثال، می‌توان یک آزمون نوشت که تعامل بین فرم جستجو در $ClientDashboard$ و لیست محصولات نمایش داده شده را شبیه‌سازی کند تا اطمینان حاصل شود که فیلتر کردن به درستی کار می‌کند. این آزمون‌ها نیز معمولاً با استفاده از $Library\ Testing\ React$ نوشته می‌شوند اما سناریوهای پیچیده‌تری را پوشش می‌ده□□□.

آزمون سرتاسری ($E2E - End - to - End$)

این لایه از آزمون، بالاترین سطح اطمینان را فراهم می‌کند. با استفاده از ابزارهایی مانند $Cypress$ یا $Playwright$، می‌توان یک مرورگر واقعی را به صورت خودکار کنترل کرده و سناریوهای کامل کاربر را شبیه‌سازی کرد. یک سناریوی $E2E$ برای این پروژه می‌تواند شامل مراحل زیر باشد:

۱. باز کردن برنامه و اتصال به یک کیف پول آزمایشی.

۲. ناوبری به صفحه «ایجاد محصول».

۳. پر کردن فرم با اطلاعات آزمایشی و آپلود یک فایل.

۴. تأیید تراکنش در پنجره پاپ‌آپ کیف پول.

۵. انتظار برای تأیید تراکنش و بررسی اینکه آیا محصول جدید در داشبورد ادمین نمایش داده می‌شود.

این راهبرد آزمون چندلایه، از پایین‌ترین سطح (منطق قرارداد هوشمند) تا بالاترین سطح (تعامل کاربر در مرورگر)، کیفیت، صحت عملکرد و امنیت کل سامانه را تضمین می‌کند

فصل چهارم

ارزیابی و تحلیل نتایج

□□□□□□□□ □□□□□□ □□□□□□□□□□ □□□□□

پس از تشریح دقیق معماری و فرآیند پیاده‌سازی سامانه در فصل سوم، این فصل به ارزیابی جامع و تحلیل نتایج عملکرد آن اختصاص دارد. هدف از این فصل، سنجش میزان موفقیت پروژه در دستیابی به اهداف تعریف شده و بررسی عملکرد سیستم در برابر معیارهای کلیدی است. ارزیابی یک سامانه غیرمتمرکز، فرآیندی چندوجهی است که فراتر از آزمون‌های عملکردی صرف رفته و جنبه‌های امنیتی، اقتصادی و تجربه کاربری را نیز در بر می‌گیرد. این فصل به دو بخش اصلی تقسیم می‌شود: در بخش اول، چارچوب ارزیابی، معیارها و محیط آزمون به تفصیل تشریح می‌شوند. در بخش دوم، نتایج به دست آمده از اجرای این آزمون‌ها ارائه و تحلیل خواهند شد.

۴-۱ معیارها و محیط ارزیابی

این بخش به عنوان سنگ بنای فرآیند ارزیابی، به تعریف دقیق معیارها، روش‌ها و محیطی می‌پردازد که برای سنجش کیفیت و عملکرد سامانه به کار گرفته خواهد شد. ارائه یک چارچوب ارزیابی شفاف و دقیق، برای اطمینان از تکرارپذیری ($Reproducibility$) و اعتبار نتایج، امری ضروری است.

۴-۱-۱ مقدمه: چارچوب ارزیابی یک سامانه غیرمتمرکز

ارزیابی یک برنامه غیرمتمرکز ($dApp$) مانند سامانه زنجیره تأمین حاضر، تفاوت‌های بنیادینی با ارزیابی نرم‌افزارهای متمرکز سنتی دارد. در یک سیستم سنتی، معیارها عمدتاً بر کارایی سرور، زمان پاسخ پایگاه داده و قابلیت‌های رابط کاربری متمرکز هستند. اما در یک سیستم غیرمتمرکز، ابعاد جدیدی از ارزیابی پدیدار می‌شود که مستقیماً از ماهیت فناوری زنجیره بلوکی نشأت می‌گیرد. اعتماد در این سیستم‌ها به جای یک نهاد مرکزی، به کد، پروتکل و اصول رمزنگاری تفویض شده است. بنابراین، ارزیابی باید بتواند میزان موفقیت این تفویض اعتماد را بسنجد.

برای این منظور، یک چارچوب ارزیابی چندبعدی تعریف شده است که پروژه را از چهار منظر کلیدی مورد سنجش قرار می‌دهد:

۱. **صحت عملکرد و کارایی** ($Performance\ and\ Correctness$): آیا سیستم همانطور که طراحی شده، به درستی و با کارایی قابل قبول کار می‌کند؟ این بعد به بررسی صحت منطق قرارداد هوشمند و عملکرد فنی آن می‌پردازد.

۲. **امنیت و استحکام** ($Robustness\ and\ Security$): آیا سیستم در برابر حملات شناخته‌شده و شرایط غیرمنتظره مقاوم است؟ این بعد، امنیت کد و معماری را در برابر تهدیدات داخلی و خارجی می‌سنجد.

۳. **کاربرپذیری و تجربه کاربری** ($Experience\ User\ and\ Usability$): آیا تعامل با سیستم برای کاربران نهایی ساده، قابل فهم و کارآمد است؟ این بعد بر طراحی انسان-محور و میزان پذیرش

سیستم توسط کاربران تمرکز دارد.

۴. **تحلیل اقتصادی و عملیاتی** (*Analysis Operational and Economic*): آیا پیاده‌سازی این سیستم در دنیای واقعی از نظر اقتصادی مقرون‌به‌صرفه و از نظر عملیاتی امکان‌پذیر است؟ این بعد به بررسی هزینه‌ها و مزایای عملیاتی سیستم می‌پردازد.

در ادامه، معیارها و روش‌شناسی ارزیابی برای هر یک از این چهار بعد به تفصیل تشریح خواهد شد.

۴-۱-۲ بعد اول: ارزیابی صحت عملکرد و کارایی

این بعد، فنی‌ترین بخش ارزیابی را تشکیل می‌دهد و هدف آن، اطمینان از صحت منطق پیاده‌سازی شده در قرارداد هوشمند و سنجش عملکرد آن تحت بارهای کاری شبیه‌سازی شده است. این ارزیابی مستقیماً بر اساس راهبرد آزمون تعریف شده در پروپوزال پروژه استوار است که بر «نوشتن یک مجموعه آزمون واحد و اجرای آن با کمک ابزار $foundry$» تأکید دارد[؟، ؟].

معیارهای صحت عملکرد (*Metrics Correctness*)

صحت عملکرد به این سؤال پاسخ می‌دهد: «آیا سیستم کاری را که باید، به درستی انجام می‌دهد؟». برای سنجش این موضوع، از معیارهای کمی و کیفی زیر استفاده خواهد شد:

- **پوشش آزمون‌های واحد و یکپارچه‌سازی:** این یک معیار کمی است که نشان می‌دهد چه درصدی از خطوط کد و شاخه‌های منطقی (*branches*) در قرارداد هوشمند توسط مجموعه آزمون‌ها اجرا و بررسی شده‌اند. هدف در این پروژه، دستیابی به پوشش آزمون نزدیک به ۱۰۰٪ برای تمام منطق‌های حیاتی کسب‌وکار است. ابزار $Foundry$ قابلیت گزارش‌گیری دقیق از پوشش آزمون را فراهم می‌کند.
- **میزان موفقیت آزمون‌ها:** معیار اصلی صحت، نرخ موفقیت ۱۰۰٪ برای کل مجموعه آزمون‌های تعریف شده در فایل $SupplyChainERC1155.t.sol$ است. هرگونه شکست در آزمون‌ها، نشان‌دهنده وجود یک باگ در منطق قرارداد است.
- **صحت اجرای سناریوهای سرتاسری:** موفقیت در اجرای سناریوهای پیچیده‌ای که تعامل چندین نقش و چندین تابع را شبیه‌سازی می‌کنند، به عنوان یک معیار کلیدی برای صحت یکپارچگی سیستم در نظر گرفته می‌شود. آزمون $testCompleteSupplyChainFlow$[؟] به طور خاص برای سنجش این معیار طراحی شده است.
- **مدیریت صحیح خطاها:** یک سیستم صحیح، نه تنها باید در مسیر شاد (*path happy*) به درستی عمل کند، بلکه باید در مواجهه با ورودی‌های نامعتبر یا اقدامات غیرمجاز، به صورت قابل

پیش‌بینی و امن، خطا برگردانده و از تغییر وضعیت ناخواسته جلوگیری کند. معیار سنجش این قابلیت، موفقیت آزمون‌هایی است که از

$$vm.expectRevert$$

برای بررسی بازگشت خطاهای مورد انتظار استفاده می‌کنند[؟].

معیارهای کارایی ($PerformanceMetrics$)

کارایی به این سؤال پاسخ می‌دهد: «آیا سیستم وظایف خود را با مصرف بهینه منابع انجام می‌دهد؟». در دنیای زنجیره بلوکی، «منابع» عمدتاً به معنای «هزینه گس» و «زمان» است.

- **هزینه گس** ($GasCost$): این مهم‌ترین معیار کارایی برای یک قرارداد هوشمند است. برای هر یک از توابع کلیدی که وضعیت زنجیره را تغییر می‌دهند، هزینه گس مصرفی به صورت دقیق اندازه‌گیری و ثبت خواهد شد. توابع مورد ارزیابی عبارتند از:
  - $registerProduct()$: هزینه ساخت یک یا چند توکن محصول جدید.
  - $destroyProduct()$: هزینه ابطال یک محصول.
  - توابع انتقال (که در $update$ مدیریت می‌شوند): هزینه انتقال مالکیت.

  تحلیل این معیار به ما نشان می‌دهد که سیستم از نظر اقتصادی چقدر برای پیاده‌سازی در یک شبکه عمومی مقرون‌به‌صرفه است. کاهش هزینه گس یکی از اهداف اصلی در بهینه‌سازی قراردادهای هوشمند است و الگوریتم‌هایی مانند $Swap-and-Pop$ که در این پروژه به کار رفته، مستقیماً در جهت بهبود این معیار طراحی شده‌اند.

- **توان پردازشی تراکنش** ($TransactionThroughput$): این معیار به تعداد تراکنش‌هایی که سیستم می‌تواند در یک بازه زمانی مشخص (مثلاً یک ثانیه) پردازش کند، اشاره دارد. لازم به ذکر است که این معیار، بیشتر به مشخصات شبکه زنجیره بلوکی زیربنایی (مانند اندازه بلوک و زمان بلوک) بستگی دارد تا خود قرارداد هوشمند. با این حال، با اندازه‌گیری هزینه گس هر تراکنش، می‌توان تخمینی از تعداد تراکنش‌هایی که در یک بلوک با سقف گس مشخص جای می‌گیرند، به دست آورد و بدین ترتیب، یک تخمین نظری از توان پردازشی ارائه داد.

- **کارایی توابع خواندنی:** توابع $view$ که وضعیت را تغییر نمی‌دهند، هزینه گس ندارند، اما کارایی آن‌ها از منظر زمان پاسخ برای لایه کاربری بسیار مهم است. در این ارزیابی، زمان اجرای توابع خواندنی پیچیده مانند $getOwnedProductsBatch$ و $getProductsBatch$ در یک گره محلی اندازه‌گیری خواهد شد تا از عدم وجود حلقه‌های پرهزینه یا منطق‌های کند در بازیابی داده‌ها اطمینان حاصل شود.

محیط آزمون فنی ($Technical\ Testing\ Environment$)

برای اطمینان از صحت و تکرارپذیری نتایج، تمام آزمون‌های فنی در یک محیط کاملاً مشخص و کنترل‌شده اجرا خواهند شد.

- **پیکربندی نرم‌افزاری:**
  - **فریم‌ورک آزمون:** $Foundry$ (نسخه مشخص خواهد شد).
  - **کامپایلر** $Solidity$: نسخه $0.8.20$ مطابق با تعریف قرارداد[؟].
  - **کتابخانه‌ها:** $OpenZeppelinContracts$ (نسخه مشخص خواهد شد).
- **پیکربندی شبکه محلی:**
  - **گره محلی:** از $Anvil$، گره آزمایشی همراه $Foundry$، استفاده خواهد شد.
  - **پیکربندی** $Anvil$: تمام آزمون‌ها با پیکربندی پیش‌فرض $Anvil$ اجرا می‌شوند که شامل حساب‌های آزمایشی با موجودی اتر کافی، زمان بلوک آنی (برای سرعت بخشیدن به آزمون‌ها) و سقف گس بالا برای هر بلوک است.
- **اسکریپت‌های استقرار:** برای آزمون‌های یکپارچه‌سازی و سرتاسری، از اسکریپت‌های استقرار نوشته شده با $Foundry$ (مانند $DeploySupplyChain.s.sol$[؟]) برای ایجاد یک وضعیت اولیه مشخص و قابل تکرار در شبکه آزمایشی استفاده خواهد شد.

۴-۱-۳ بعد دوم: ارزیابی امنیت و استحکام

امنیت، حیاتی‌ترین جنبه یک قرارداد هوشمند است. این بخش از ارزیابی، با هدف شناسایی و سنجش مقاومت سیستم در برابر آسیب‌پذیری‌های شناخته‌شده و بردارهای حمله بالقوه طراحی شده است.

مقدمه: امنیت به عنوان یک فرآیند

امنیت یک ویژگی صفر و یک نیست، بلکه یک فرآیند مستمر است که از مرحله طراحی معماری آغاز شده، در حین پیاده‌سازی با رعایت بهترین شیوه‌ها ادامه یافته و در نهایت، از طریق آزمون‌های دقیق و حسابرسی‌های مستقل، تأیید می‌شود. چارچوب ارزیابی امنیت این پروژه، تمام این مراحل را در بر می‌گیرد.

معیارهای امنیت قرارداد هوشمند

- **مقاومت در برابر آسیب‌پذیری‌های رایج:** معیار اصلی، عدم وجود هرگونه آسیب‌پذیری شناخته‌شده در کد قرارداد هوشمند است. لیستی از این آسیب‌پذیری‌ها که مورد بررسی قرار خواهند گرفت، عبارتند از:

- حملات بازگشتی ($Reentrancy$)
- سرریز/زیرریز عدد صحیح ($Integer\ Overflow/Underflow$)
- کنترل دسترسی نادرست ($Control\ Improper\ Access$)
- آسیب‌پذیری‌های مربوط به ترتیب تراکنش‌ها ($Front-running$)

- **صحت پیاده‌سازی کنترل دسترسی:** این معیار به صورت کمی سنجیده می‌شود: آیا تمام توابع محافظت‌شده با $onlyRole$، به ازای تمام نقش‌های غیرمجاز، تراکنش را بازگشت ($revert$) هند؟ آزمون‌هایی مانند $testRegisterProductOnlyManufacturer$[؟] و $testDestroyProductOnlyCustoms$[؟] برای سنجش این معیار طراحی شده‌اند.

- **امنیت در شرایط اضطراری:** عملکرد صحیح مکانیزم توقف اضطراری ($Pausable$) به عنوان یک معیار امنیتی کلیدی در نظر گرفته می‌شود. آزمون $testPauseUnpause$[؟] بررسی می‌کند که آیا پس از فعال‌سازی حالت توقف، تمام توابع حساس از کار می‌افتند و پس از غیرفعال‌سازی، به حالت عادی بازمی‌گردند.

معیارهای امنیت معماری ترکیبی ($On-Chain/Off-Chain$)

امنیت این سیستم تنها به قرارداد هوشمند محدود نمی‌شود و باید یکپارچگی کل معماری، به ویژه ارتباط بین داده‌های $On-chain$ و $Off-chain$ را نیز در بر گیرد.

- **یکپارچگی فراداده:** معیار اصلی، نرخ موفقیت ۱۰۰٪ تابع $verifyMetadataIntegrity$[؟] در لایه کاربری است. این تابع باید در دو سناریو آزموده شود: (۱) با استفاده از فراداده معتبر بازیابی شده از $IPFS$ که باید نتیجه «صحیح» برگرداند، و (۲) با استفاده از یک نسخه دستکاری شده از فراداده که باید نتیجه «غلط» برگرداند.

- **در دسترس بودن فراداده:** این معیار، پایداری لینک‌های $IPFS$ ذخیره شده در قرارداد را می‌سنجد. روش ارزیابی، تلاش برای بازیابی تمام $metadataUrl$های ثبت شده در طول آزمون‌ها و سنجش نرخ موفقیت در دسترسی به محتوای آن‌ها از طریق یک گیت‌وی عمومی $IPFS$ است.

روش‌شناسی ارزیابی امنیتی

برای سنجش معیارهای فوق، از یک رویکرد چندلایه استفاده خواهد شد:

۱. **تحلیل استاتیک کد** ($Static\ Code\ Analysis$): قبل از اجرای هر آزمونی، کد منبع $Solidity$ با استفاده از ابزارهای تحلیل استاتیک مانند $Slither$ مورد بررسی قرار خواهد گرفت. این ابزارها کد را برای یافتن الگوهای آسیب‌پذیری شناخته‌شده، خطاهای منطقی رایج و عدم رعایت بهترین شیوه‌های امنیتی اسکن می‌کنند. خروجی این ابزارها، یک گزارش اولیه از نقاط ضعف بالقوه فراهم می‌کند.

۲. **آزمون‌های امنیتی مبتنی بر سناریو:** مجموعه آزمون‌های $Foundry$ ($t.sol$. (□□□□ گسترش داده خواهد شد تا شامل سناریوهایی باشد که به طور خاص برای آزمودن امنیت طراحی شده‌اند. برای مثال، یک قرارداد مهاجم شبیه‌سازی خواهد شد که تلاش می‌کند یک حمله بازگشتی را به توابع انتقال اجرا کند. انتظار می‌رود که به دلیل استفاده از استانداردهای امن $OpenZeppelin$ (که دارای مکانیزم‌های ضد-بازگشتی هستند)، این آزمون‌ها با موفقیت نشان دهند که قرارداد آسیب‌پذیر نیست.

۳. **حسابرسی دستی کد** ($Manual\ Code\ Audit$)**:** کد قرارداد به صورت دستی و خط به خط مورد بازبینی قرار خواهد گرفت. این بازبینی بر روی منطق کسب‌وکار، مدیریت وضعیت‌ها، و به ویژه تعاملات بین توابع مختلف تمرکز خواهد کرد تا آسیب‌پذیری‌هایی که ممکن است از دید ابزارهای خودکار پنهان بمانند، شناسایی شوند.

۴. **تأکید بر اصول طراحی امن:** در گزارش نهایی ارزیابی، نشان داده خواهد شد که چگونه اصول طراحی امن از ابتدا در پروژه رعایت شده‌اند. مواردی مانند استفاده از کتابخانه‌های حسابرسی‌شده، پیروی از الگوی $Checks - Effects - Interactions$، و استفاده صحیح از کنترل دسترسی، همگی به عنوان بخشی از راهبرد امنیتی پروژه مستند خواهند شد.

۴-۱-۴ بعد سوم: ارزیابی کاربرپذیری و تجربه کاربری

یک سامانه زنجیره تأمین، در نهایت توسط انسان‌ها با سطوح مختلف دانش فنی مورد استفاده قرار می‌گیرد. بنابراین، ارزیابی موفقیت آن بدون در نظر گرفتن جنبه‌های انسانی و تجربه کاربری (UX) ناقص خواهد بود. هدف این بخش، ارائه یک چارچوب برای ارزیابی میزان سادگی، کارایی و رضایت‌بخش بودن تعامل کاربران با لایه کاربری سامانه است.

مقدمه: اهمیت طراحی انسان-محور در $dApps$

تاریخچه برنامه‌های غیرمتمرکز نشان داده است که یکی از بزرگ‌ترین موانع بر سر راه پذیرش گسترده آن‌ها، تجربه کاربری ضعیف و پیچیده بوده است. مفاهیمی مانند مدیریت کلید خصوصی، امضای تراکنش و پرداخت هزینه گس، برای کاربران عادی موانع بزرگی ایجاد می‌کنند. یک $dApp$ موفق، برنامه‌ای است که می‌تواند این پیچیدگی‌ها را در پس‌زمینه انتزاع کرده و یک تجربه کاربری آشنا و روان را ارائه دهد. ارزیابی این پروژه باید نشان دهد که تا چه حد در این امر موفق بوده است.

معیارهای ارزیابی UX

برای ارزیابی تجربه کاربری، از ترکیبی از معیارهای کمی و کیفی استفاده خواهد شد:

- **کارایی** ($Efficiency$): این معیار به میزان تلاش (زمان و تعداد کلیک‌ها) که یک کاربر برای انجام یک وظیفه اصلی نیاز دارد، اشاره می‌کند. برای مثال: «چند ثانیه طول می‌کشد تا یک کاربر تولیدکننده، یک محصول جدید را با موفقیت ثبت کند؟»
- **میزان خطا** ($Error\ Rate$): تعداد خطاهایی که کاربران در حین انجام یک سناریوی مشخص مرتکب می‌شوند. یک رابط کاربری خوب، باید کاربر را راهنمایی کرده و از بروز خطاهای رایج جلوگیری کند.
- **یادگیری‌پذیری** ($Learnability$): این معیار نشان می‌دهد که یک کاربر جدید با چه سرعتی می‌تواند یاد بگیرد که چگونه وظایف اصلی را در سیستم انجام دهد، بدون اینکه نیاز به آموزش رسمی گسترده داشته باشد.
- **رضایت کاربر** ($User\ Satisfaction$): این یک معیار کیفی است که احساسات و نظرات کاربران را در مورد تجربه کلی استفاده از سیستم می‌سنجد.

روش‌شناسی ارزیابی UX (پروتکل آزمون کاربر)

برای سنجش معیارهای فوق، یک پروتکل آزمون کاربردپذیری شبیه‌سازی خواهد شد. این فرآیند شامل مراحل زیر است:

۱. **تعریف پرسوناها** ($Personas$): دو پرسونای اصلی برای کاربران سیستم تعریف می‌شود:

- **پرسونای مدیر/تولیدکننده:** فردی که با فرآیندهای تولید و مدیریت موجودی آشناست اما دانش فنی محدودی در زمینه زنجیره بلوکی دارد. وظیفه اصلی او، ثبت محصولات جدید و مدیریت آن‌ها در داشبورد ادمین است.
- **پرسونای مصرف‌کننده/مالک:** فردی که یک محصول را خریداری کرده و می‌خواهد از اصالت و تاریخچه آن اطمینان حاصل کند. او با مفاهیم فنی زنجیره بلوکی آشنا نیست و تنها به دنبال یک تجربه ساده و قابل اعتماد است.

۲. **تدوین سناریوهای آزمون** ($Test\ Scenarios$): بر اساس قابلیت‌های پیاده‌سازی شده در لایه کاربری، سناریوهای مشخصی برای هر پرسونا تدوین می‌شود. این سناریوها، وظایف واقعی را که کاربر در سیستم انجام خواهد داد، شبیه‌سازی می‌کنند.

- **سناریوی ۱ (برای پرسونای مدیر):** «شما مدیر تولید یک شرکت الکترونیکی هستید. لطفاً با استفاده از اطلاعات زیر و تصویر محصول، یک بچ جدید شامل ۱۰۰ عدد "گوشی هوشمند مدل "□ را در سیستم ثبت کرده و مالکیت آن را به کیف پول توزیع‌کننده به آدرس [...] منتقل نمایید.»

- **سناریوی ۲ (برای پرسونای مصرف‌کننده):** «شما به تازگی یک "گوشی هوشمند مدل "□ خریداری کرده‌اید. لطفاً با استفاده از کد QR ارائه شده، ابتدا از اصالت و یکپارچگی اطلاعات آن اطمینان حاصل کرده و سپس تاریخچه کامل مالکیت آن از زمان تولید را مشاهده نمایید.»

۳. **اجرای آزمون و جمع‌آوری داده‌ها:** آزمون با شرکت‌کنندگانی که نماینده پرسوناها هستند، اجرا خواهد شد. در طول آزمون، از روش‌های زیر برای جمع‌آوری داده استفاده می‌شود:

- **پروتکل تفکر با صدای بلند** ($Think-aloud\ Protocol$): از شرکت‌کنندگان خواسته می‌شود تا در حین انجام سناریوها، افکار، ابهامات و تصمیمات خود را با صدای بلند بیان کنند.
- **مشاهده و زمان‌سنجی:** یک مشاهده‌گر، زمان انجام هر وظیفه و تعداد خطاهای کاربر را ثبت می‌کند.
- **پرسشنامه‌های پس از آزمون:** پس از اتمام سناریوها، از شرکت‌کنندگان خواسته می‌شود تا پرسشنامه‌های استانداردی مانند مقیاس کاربردپذیری سیستم ($System\ Usability\ Scale - SUS$) را پر کنند تا یک نمره کمی برای رضایت کلی آن‌ها به دست آید.

نتایج حاصل از این ارزیابی، بینش‌های ارزشمندی در مورد نقاط قوت و ضعف طراحی رابط کاربری فراهم کرده و پیشنهاداتی مشخص برای بهبود تجربه کاربری در نسخه‌های آینده ارائه خواهد داد.

۴-۱-۵ بعد چهارم: تحلیل اقتصادی و عملیاتی

در نهایت، موفقیت یک پروژه فناورانه در دنیای واقعی، نه تنها به برتری فنی، بلکه به توجیه اقتصادی و امکان‌پذیری عملیاتی آن نیز بستگی دارد. این بخش از ارزیابی، با هدف تحلیل هزینه‌ها و مزایای سامانه و بررسی امکان‌سنجی استقرار آن در یک محیط تجاری واقعی انجام می‌شود.

مقدمه: از اثبات مفهوم تا طرح تجاری

پروژه حاضر در مرحله فعلی، یک «اثبات مفهوم» ($Proof\ of\ Concept$) قدرتمند است که نشان می‌دهد ایده اصلی از نظر فنی قابل پیاده‌سازی است. اما برای تبدیل شدن به یک محصول تجاری، باید به سؤالات اقتصادی پاسخ دهد. این تحلیل، یک چارچوب اولیه برای این ارزیابی اقتصادی فراهم می‌کند.

معیارهای ارزیابی اقتصادی

- **تحلیل هزینه کل مالکیت** ($TCO - Total\ Cost\ of\ Ownership$): این معیار، تمام هزینه‌های مرتبط با راه‌اندازی و نگهداری سیستم را در یک دوره زمانی مشخص در بر می‌گیرد. اجزای اصلی TCO برای این سامانه عبارتند از:

- **هزینه‌های تراکنش (هزینه گس):** این هزینه که متغیر اصلی است، بر اساس نتایج تحلیل کارایی در بخش اول، و با در نظر گرفتن سناریوهای مختلف برای قیمت گس و حجم تراکنش‌ها، مدل‌سازی خواهد شد.
- **هزینه‌های ذخیره‌سازی** $Off-chain$: شامل هزینه‌های مربوط به اشتراک سرویس پینینگ $Pinata$.
- **هزینه‌های زیرساخت** $Frontend$: شامل هزینه‌های مربوط به میزبانی وب‌سایت.
- **هزینه‌های توسعه و نگهداری:** شامل هزینه‌های نیروی انسانی برای توسعه، به‌روزرسانی و پشتیبانی از سیستم.

- **تحلیل بازگشت سرمایه** ($ROI - Investment\ on\ Return$): این معیار، مزایای مالی حاصل از پیاده‌سازی سیستم را در مقایسه با هزینه‌های آن می‌سنجد. شناسایی و کمی‌سازی این مزایا، بخش چالش‌برانگیز این تحلیل است. مزایای اصلی عبارتند از:
  - **کاهش خسارات ناشی از جعل:** با تخمین درصد کالاهای تقلبی در بازار هدف و جلوگیری از فروش آن‌ها.
  - **افزایش کارایی عملیاتی:** با کاهش زمان و نیروی انسانی مورد نیاز برای فرآیندهای تطبیق، حسابرسی و ردیابی.
  - **کاهش هزینه‌های فراخوان محصول:** با امکان ردیابی سریع و دقیق محصولات معیوب.
  - **افزایش درآمد بالقوه:** ناشی از افزایش اعتماد مشتریان و تقویت ارزش برند.

روش‌شناسی تحلیل

برای انجام این تحلیل، یک رویکرد مبتنی بر مدل‌سازی و شبیه‌سازی به کار گرفته خواهد شد:

۱. **ساخت یک مدل هزینه:** یک مدل صفحه‌گسترده ($Spreadsheet$) ایجاد خواهد شد که به عنوان ورودی، پارامترهایی مانند «تعداد محصولات ثبت‌شده در روز»، «تعداد انتقالات در روز» و «قیمت متوسط گس» را دریافت کرده و به عنوان خروجی، «هزینه عملیاتی روزانه» سامانه را محاسبه می‌کند.

۲. **تحلیل مقایسه‌ای:** هزینه‌های به دست آمده از مدل، با هزینه‌های عملیاتی یک سیستم سنتی معادل (با در نظر گرفتن هزینه‌های نیروی انسانی برای حسابرسی دستی و غیره) مقایسه خواهد شد.

۳. **تحلیل حساسیت:** تأثیر تغییرات پارامترهای کلیدی (مانند قیمت گس) بر روی هزینه کل، مورد تحلیل قرار خواهد گرفت تا نقاط سربه‌سر و ریسک‌های اقتصادی پروژه شناسایی شوند.

این تحلیل اقتصادی، یک دید واقع‌بینانه از چالش‌ها و فرصت‌های تجاری پیش روی پروژه فراهم کرده و به تصمیم‌گیران برای استقرار نهایی آن کمک خواهد کرد.

فصل پنجم

جمع‌بندی و پیشنهاد برای کارهای آینده

□□□□□□□ □□□□□□□□□

این فصل به عنوان نقطه پایانی این پژوهش، دو هدف اصلی را دنبال می‌کند. در بخش اول، با نگاهی به گذشته، به جمع‌بندی و مرور جامع تمام مباحث مطرح شده در فصول پیشین می‌پردازیم. در این بخش، دستاوردهای کلیدی پروژه در پاسخ به مسئله اصلی تحقیق، یعنی بحران اعتماد و شفافیت در زنجیره‌های تأمین، به صورت نظام‌مند تحلیل و ارائه خواهد شد. این جمع‌بندی نشان می‌دهد که چگونه معماری و پیاده‌سازی ارائه شده، به اهداف اولیه پروژه دست یافته است.

در بخش دوم، با نگاهی به آینده، ابتدا محدودیت‌های پژوهش حاضر به صورت شفاف مورد بحث قرار گرفته و سپس، یک نقشه راه دقیق و بلندپروازانه برای توسعه‌های آتی این سامانه ارائه می‌گردد. این بخش، مسیرهای پژوهشی و فنی جدیدی را ترسیم می‌کند که می‌توانند این پروژه را از یک «اثبات مفهوم» قدرتمند، به یک پلتفرم کامل و آماده برای استقرار در دنیای واقعی تبدیل کنند.

۵-۱ جمع‌بندی و مرور دستاوردها

این پژوهش با هدف ارائه یک راهکار نوین برای مقابله با چالش‌های بنیادین زنجیره‌های تأمین سنتی آغاز شد. در این بخش، با مرور خلاصه پژوهش، دستاوردهای اصلی آن را در چارچوب اهداف اولیه تحلیل می‌کنیم.

۵-۱-۱ مقدمه: بازگشت به مسئله اصلی

همانطور که در فصل اول به تفصیل بیان شد، زنجیره‌های تأمین سنتی از مشکلات ساختاری عمیقی رنج می‌برند. مسائلی همچون جعل محصولات، عدم شفافیت در فرآیندها، ناکارآمدی ناشی از سیلوهای اطلاعاتی و در نهایت، «بحران اعتماد» بین شرکای تجاری و مصرف‌کنندگان، انگیزه‌های اصلی این پژوهش را تشکیل دادند[؟، ؟]. سیستم‌های متمرکز سنتی، به دلیل ماهیت قابل دستکاری و عدم قابلیت همکاری، در حل این مشکلات ناتوان بوده‌اند. این پژوهش با این فرضیه آغاز شد که فناوری زنجیره بلوکی، با ارائه یک لایه اعتماد غیرمتمرکز، می‌تواند راهگشای این چالش‌ها باشد.

۵-۱-۲ خلاصه جامع پژوهش

برای پاسخ به این مسئله، این پایان‌نامه یک مسیر منطقی را از تئوری تا عمل طی کرد:

- **در فصل اول،** صورت مسئله به دقت تعریف شد. اهداف پروژه شامل ایجاد یک سیستم ردیابی شفاف، مدیریت بهینه دارایی‌های ناهمگون، تضمین صحت فراداده‌ها و خودکارسازی فرآیندهای تجاری، به روشنی مشخص گردید. همچنین، چالش‌های فنی، امنیتی، قانونی و کاربرپذیری به عنوان موانع اصلی پیش روی پروژه شناسایی شدند.

- **در فصل دوم،** یک مرور جامع بر ادبیات تحقیق انجام شد. ابتدا، محدودیت‌های سیستم‌های سنتی و راهکارهای دیجیتال غیربلاکچینی تحلیل گردید. سپس، دو نسل از راهکارهای زنجیره بلوکی (پلتفرم‌های خصوصی و راهکارهای مبتنی بر توکنیزه‌سازی در شبکه‌های عمومی) مورد بررسی انتقادی قرار گرفتند. این تحلیل، به شناسایی سه شکاف پژوهشی کلیدی منتهی شد: چالش مدیریت دارایی‌های ناهمگون، مسئله یکپارچگی داده‌های خارج از زنجیره، و فقدان انطباق‌پذیری با محیط‌های نظارتی.

- **در فصل سوم،** معماری و روش پیاده‌سازی سامانه پیشنهادی به تفصیل تشریح شد. این فصل، جزئیات فنی هر سه لایه سیستم (لایه زنجیره بلوکی، لایه ذخیره‌سازی $Off-chain$ و لایه کاربری) را پوشش داد. انتخاب‌های کلیدی فناوری مانند $Solidity$، $ERC-1155$، $Foundry$، $IPFS$، $React$ و $Wagmi$ توجیه شده و پیاده‌سازی‌های مهم، از منطق قرارداد هوشمند گرفته تا کامپوننت‌های رابط کاربری، با استفاده از قطعه کدها به نمایش گذاشته شد.

- **در فصل چهارم،** یک چارچوب ارزیابی جامع و چندبعدی برای سنجش موفقیت پروژه ارائه گردید. این چارچوب، معیارهای دقیقی را برای ارزیابی سیستم از چهار منظر صحت عملکرد و کارایی، امنیت و استحکام، کاربرپذیری و تجربه کاربری، و تحلیل اقتصادی و عملیاتی تعریف کرد و روش‌شناسی دقیقی برای سنجش هر یک از این معیارها ارائه داد.

۵-۱-۳ تحلیل دستاوردهای کلیدی پروژه

اکنون می‌توانیم با اطمینان بیشتری ادعا کنیم که سامانه طراحی و پیاده‌سازی شده در این پژوهش، به صورت موفقی به اهداف اولیه خود دست یافته است. در ادامه، هر یک از این دستاوردها به تفصیل تحلیل می‌شوند.

دستاورد اول: تحقق ردیابی شفاف و سرتاسری

یکی از اصلی‌ترین اهداف پروژه، ایجاد یک تاریخچه کامل، شفاف و تغییرناپذیر برای هر محصول بود. این هدف از طریق ترکیب هوشمندانه چندین سازوکار در قرارداد هوشمند محقق شده است:

- **ثبت تاریخچه در ساختار داده اختصاصی:** با استفاده از ساختار داده $OwnershipRecord$ و نگاشت $ownershipHistory[?, ?]$، هر رویداد مالکیتی (از تولید تا انتقالات بعدی) به صورت یک رکورد مجزا با ذکر مالک، زمان دقیق و دلیل انتقال، ثبت می‌شود.

- **قلاب کردن در فرآیند انتقال:** با بازنویسی تابع محوری $update_u[?]$، قرارداد تضمین می‌کند که هیچ انتقال توکنی (که نمایانگر انتقال مالکیت است) بدون ثبت رکورد متناظر در تاریخچه، انجام نخواهد شد. این سازوکار، یکپارچگی کامل بین وضعیت مالکیت توکن و تاریخچه ثبت‌شده را تضمین می‌کند.

- **قابلیت حسابرسی عمومی:** از آنجا که این تاریخچه بر روی زنجیره بلوکی عمومی ثبت می‌شود، هر فردی (از جمله مصرف‌کنندگان و نهادهای نظارتی) می‌تواند به صورت مستقل و بدون نیاز به کسب اجازه، تاریخچه کامل یک محصول را مشاهده و حسابرسی کند. این شفافیت رادیکال، هسته اصلی راهکار ارائه شده برای مقابله با عدم شفافیت سیستم‌های سنتی است.

دستاورد دوم: مدیریت بهینه دارایی‌های ناهمگون

این پروژه با موفقیت به «شکاف مدیریت دارایی‌های ناهمگون» که در فصل دوم شناسایی شد، پاسخ داده است. انتخاب استراتژیک استاندارد

$$ERC-1155$$

[؟]، یک دستاورد معماری کلیدی است که سیستم را قادر می‌سازد تا پیچیدگی‌های دنیای واقعی زنجیره تأمین را مدل‌سازی کند.

- **انعطاف‌پذیری ذاتی:** همانطور که در تحلیل‌های پیشین نشان داده شد، این استاندارد به سیستم اجازه می‌دهد تا دارایی‌های کاملاً متفاوت (مثلاً یک خودروی منحصربه‌فرد به عنوان NFT و هزاران لاستیک یکسان به عنوان توکن مثلی) را در یک قرارداد واحد مدیریت کند.
- **کارایی عملیاتی:** قابلیت $Batch\ Transfer$ در این استاندارد، که امکان انتقال چندین نوع دارایی مختلف را در یک تراکنش واحد فراهم می‌کند، یک مزیت عملیاتی بسیار بزرگ برای فرآیندهای لجستیکی است. این قابلیت به صورت مستقیم در تابع $_{u}pdate$ که آرایه‌ای از شناسه‌ها و مقادیر را می‌پذیرد، پشتیبانی می‌شود[؟]. این دستاورد، نشان‌دهنده یک طراحی بهینه است که نه تنها از نظر تئوری، بلکه از نظر هزینه و کارایی عملیاتی نیز بر راهکارهای مبتنی بر استانداردهای تک‌منظوره برتری دارد.

دستاورد سوم: تضمین یکپارچگی داده‌ها از طریق معماری ترکیبی

پروژه حاضر با ارائه یک معماری ترکیبی امن و اقتصادی، به «شکاف یکپارچگی داده‌های $Off-chain$» پاسخ داده است. این دستاورد، حاصل ترکیب سه فناوری است:

۱. **ذخیره‌سازی غیرمتمرکز با $IPFS$:** با انتخاب $IPFS$ به جای سرورهای متمرکز، سیستم از ریسک‌های مربوط به نقطه شکست واحد و سانسور داده‌ها اجتناب می‌کند[؟].

۲. **پیوند رمزنگاری با $Keccak256$:** با ثبت هش $Keccak256$ فراداده بر روی زنجیره بلوکی در ساختار داده $Product$[؟]، یک پیوند تغییرناپذیر و قابل تأیید بین توکن $On-chain$ و داده‌های $Off-chain$ ایجاد می‌شود.

۳. **اعتبارسنجی سمت کاربر:** با پیاده‌سازی تابع $verifyMetadataIntegrity$ در لایه کاربری[؟]، قدرت اعتبارسنجی مستقیماً به دست کاربر نهایی سپرده می‌شود. این مکانیزم، نیاز به اعتماد به

هرگونه واسطه (حتی گیت‌وی $IPFS$) را از بین برده و یک سیستم واقعاً «بی‌نیاز به اعتماد» ($trustless$) را محقق می‌سازد.

این معماری، یک الگوی قدرتمند برای تمام برنامه‌های غیرمتمرکزی است که نیاز به مدیریت داده‌های حجیم دارند.

دستاورد چهارم: انتزاع پیچیدگی از طریق تجربه کاربری مدرن

این پروژه نشان داد که می‌توان یک برنامه غیرمتمرکز پیچیده را در قالب یک تجربه کاربری ساده و آشنا ارائه داد.

- **داشبوردهای نقش-محور:** با طراحی داشبوردهای مجزا برای ادمین و مشتری ($AdminDashboard.tsx$[؟] و $ClientDashboard.tsx$[؟])، رابط کاربری متناسب با نیازها و وظایف هر نقش، بهینه شده است.

- **انتزاع فرآیندهای پیچیده:** فرآیند چندمرحله‌ای و پیچیده ثبت محصول (شامل آپلود در $IPFS$، محاسبه هش و ارسال تراکنش) در پشت یک فرم ساده در صفحه $CreateProduct.tsx$[؟] پنهان شده است. کاربر تنها با پر کردن چند فیلد و یک کلیک، این فرآیند را به انجام می‌رساند.

- **یکپارچگی با اکوسیستم موجود:** با استفاده از کتابخانه $Wagmi$ و پشتیبانی از کیف پول‌های استاندارد مانند $MetaMask$، کاربران می‌توانند از ابزارهایی که از قبل با آن آشنا هستند، برای تعامل با سیستم استفاده کنند. این امر، موانع ورود کاربران جدید را به شدت کاهش می‌دهد.

۵-۱-۴ پاسخ نهایی به سؤالات تحقیق

این پژوهش در تلاش برای پاسخ به چندین سؤال کلیدی بود که اکنون می‌توان به صورت خلاصه به آن‌ها پاسخ داد:

- **سؤال ۱: چگونه می‌توان از فناوری زنجیره بلوکی برای ایجاد یک سیستم ردیابی شفاف و قابل اعتماد در زنجیره تأمین استفاده کرد؟ پاسخ:** با توکنیزه کردن هر محصول به عنوان یک دارایی دیجیتال ($ERC-1155$) و ثبت تمام رویدادهای مربوط به چرخه حیات آن (از تولید تا انتقالات) به عنوان تراکنش‌های تغییرناپذیر در یک قرارداد هوشمند، می‌توان یک تاریخچه کامل و قابل حسابرسی عمومی ایجاد کرد.

- **سؤال ۲: راهکار بهینه برای مدیریت دارایی‌های ناهمگون (مثلی و غیرمثلی) در یک زنجیره تأمین چیست؟ پاسخ:** استاندارد چند-توکنی $ERC-1155$ به دلیل انعطاف‌پذیری ذاتی در مدیریت همزمان هر دو نوع دارایی و قابلیت‌های کارآمدی مانند انتقال دسته‌ای، راهکار برتر و بهینه‌تر نسبت به استانداردهای تک‌منظوره مانند $ERC-20$ و $ERC-721$ است.

- **سؤال ۳: چگونه می‌توان مشکل هزینه و محدودیت ذخیره‌سازی داده در زنجیره بلوکی را بدون به خطر انداختن یکپارچگی داده‌ها حل کرد؟ پاسخ:** با استفاده از یک معماری ترکیبی که در آن، داده‌های حجیم در یک سیستم ذخیره‌سازی غیرمتمرکز $Off-chain$ (مانند $IPFS$) نگهداری شده و تنها هش رمزنگاری شده آن داده‌ها به صورت $On-chain$ ثبت می‌گردد. اعتبارسنجی نیز به صورت سمت کاربر و با مقایسه مجدد هش‌ها انجام می‌شود.

۵-۲ محدودیت‌های پروژه و پیشنهاد برای کارهای آینده

این پژوهش، با وجود دستاوردهای قابل توجه، به عنوان گامی اولیه در یک مسیر طولانی و پیچیده محسوب می‌شود. شناخت و بیان شفاف محدودیت‌های این پروژه، نه تنها برای حفظ اعتبار علمی آن ضروری است، بلکه زمینه را برای تحقیقات و توسعه‌های آتی هموار می‌سازد.

۵-۲-۱ تحلیل محدودیت‌های پژوهش

- **محدودیت‌های محیط ارزیابی:** تمام آزمون‌های عملکردی و امنیتی این پروژه در یک محیط توسعه محلی و کنترل‌شده ($Anvil$) انجام شده است. اگرچه این محیط برای اطمینان از صحت منطقی کد عالی است، اما معیارهایی مانند هزینه واقعی گس، زمان تأیید تراکنش و توان پردازشی، در یک شبکه عمومی واقعی مانند $Mainnet\ Ethereum$ یا حتی یک $Testnet$ عمومی مانند $Sepolia$، تحت تأثیر ازدحام شبکه و شرایط متغیر بازار قرار خواهند گرفت. بنابراین، نتایج عملکرد ارائه شده باید به عنوان یک معیار پایه در نظر گرفته شوند و نه یک شاخص قطعی از عملکرد در دنیای واقعی.

- **محدودیت‌های دامنه پیاده‌سازی:** برای تمرکز بر روی هسته اصلی نوآوری، برخی از جنبه‌های سیستم به صورت ساده‌سازی شده پیاده‌سازی شده‌اند. برای مثال، منطق محاسبه مالیات در قرارداد هوشمند به عنوان یک قلاب تعریف شده اما پیاده‌سازی آن به صورت یک مدل ساده است و از اوراکل‌ها برای دریافت نرخ‌های پویا استفاده نمی‌کند. همچنین، مدیریت خطاهای لایه کاربری و ارائه بازخوردهای دقیق‌تر به کاربر در شرایط خاص (مانند رد شدن تراکنش توسط کاربر) می‌تواند بسیار جامع‌تر شود.

- **محدودیت‌های امنیتی:** با وجود پیروی از بهترین شیوه‌های امنیتی و استفاده از کتابخانه‌های حسابرسی‌شده، قرارداد هوشمند این پروژه هنوز تحت یک فرآیند حسابرسی امنیتی رسمی و مستقل توسط یک شرکت شخص ثالث قرار نگرفته است. این فرآیند، یک گام ضروری و حیاتی قبل از استقرار هرگونه سیستمی است که با دارایی‌های واقعی سروکار خواهد داشت.

- **محدودیت‌های مدل کسب‌وکار و حاکمیت:** این پژوهش عمدتاً بر جنبه‌های فنی متمرکز بوده و یک مدل کسب‌وکار پایدار یا یک چارچوب حاکمیتی دقیق برای مدیریت شبکه در دنیای

واقعی (که شامل چندین شرکت رقیب است) ارائه نداده است. مسائلی مانند نحوه تأمین هزینه‌های عملیاتی شبکه و فرآیند تصمیم‌گیری برای به‌روزرسانی پروتکل، نیازمند تحقیقات بیشتری هستند.

۵-۲-۲ نقشه راه برای توسعه‌های آینده

بر اساس دستاوردها و با در نظر گرفتن محدودیت‌های ذکر شده، یک نقشه راه هیجان‌انگیز برای توسعه‌های آتی این پروژه قابل ترسیم است. هر یک از موارد زیر می‌تواند به عنوان یک پروژه تحقیقاتی یا توسعه‌ای مستقل، این سامانه را به بلوغ و پذیرش گسترده نزدیک‌تر کند.

پیشنهاد اول: ادغام با اینترنت اشیاء (IoT) برای ورود خودکار داده

مسئله: یکی از بزرگ‌ترین چالش‌ها در هر سیستم ردیابی، اطمینان از صحت داده‌های ورودی است. در مدل فعلی، داده‌ها به صورت دستی توسط تولیدکننده وارد می‌شوند که این امر، احتمال خطای انسانی یا حتی ورود داده جعلی را به همراه دارد (مشکل معروف به «آشغال ورودی، آشغال خروجی» یا $In\ Garbage$، $Out\ Garbage$).

راهکار پیشنهادی: ادغام سامانه با سنسورهای اینترنت اشیاء (IoT). سنسورهای فیزیکی (مانند سنسورهای دما، رطوبت، موقعیت‌یاب GPS و شتاب‌سنج) می‌توانند به صورت خودکار و مستمر، داده‌های مربوط به وضعیت و شرایط نگهداری محصول را جمع‌آوری کرده و به سیستم ارسال کنند. این داده‌ها می‌توانند به عنوان بخشی از تاریخچه تغییرناپذیر محصول بر روی زنجیره بلوکی ثبت شوند.

معماری پیشنهادی: یک معماری امن برای این منظور می‌تواند به شرح زیر باشد:

۱. هر دستگاه IoT دارای یک جفت کلید رمزنگاری منحصربه‌فرد است و هویت آن در یک قرارداد هوشمند مجزا به نام «رجیستری دستگاه‌ها» ثبت شده است.

۲. دستگاه IoT داده‌های جمع‌آوری شده (مثلاً دمای ۳ درجه سانتی‌گراد در زمان (را با استفاده از کلید خصوصی خود امضا کرده و به یک سرویس اوراکل ارسال می‌کند.

۳. سرویس اوراکل (مانند شبکه‌های غیرمتمرکز اوراکل نظیر $Chainlink$)، امضای دستگاه را با کلید عمومی ثبت‌شده آن در رجیستری اعتبارسنجی می‌کند.

۴. پس از تأیید، اوراکل یک تراکنش به قرارداد هوشمند $SupplyChainERC1155$ ارسال کرده و یک تابع جدید (مثلاً $addIoTData$) را فراخوانی می‌کند. این تابع، داده‌های معتبر دریافت شده از سنسور را به تاریخچه محصول مربوطه اضافه می‌کند.

این معماری، فرآیند ورود داده را خودکار، بی‌نیاز به اعتماد و مقاوم در برابر دستکاری می‌سازد و یکپارچگی داده‌ها را از دنیای فیزیکی تا دنیای دیجیتال تضمین می‌کند.

پیشنهاد دوم: پیاده‌سازی راهکارهای مقیاس‌پذیری لایه ۲ ($Layer\ 2$)

مسئله: همانطور که در فصل چالش‌ها به تفصیل بحث شد، هزینه‌های بالا و توان پردازشی محدود شبکه‌های لایه ۱ ($Layer\ 1$) مانند اتریوم، بزرگ‌ترین مانع بر سر راه پذیرش صنعتی این سامانه است.

راهکار پیشنهادی: مهاجرت کامل یا بخشی از منطق سامانه به یک شبکه لایه ۲. راهکارهای لایه ۲، پروتکل‌هایی هستند که بر روی یک زنجیره بلوکی لایه ۱ ساخته می‌شوند و هدف آن‌ها، افزایش چشمگیر مقیاس‌پذیری و کاهش هزینه‌هاست. دو دسته اصلی از این راهکارها عبارتند از:

- $Optimistic\ Rollups$: پلتفرم‌هایی مانند $Arbitrum$ و $Optimism$ که تراکنش‌ها را در خارج از زنجیره اجرا کرده و به صورت دسته‌ای به لایه ۱ ارسال می‌کنند. آن‌ها فرض را بر صحت تراکنش‌ها می‌گذارند مگر اینکه خلاف آن توسط یک «چالش» اثبات شود.
- $Zero-Knowledge\ Rollups$ ($ZK-Rollups$): پلتفرم‌هایی مانند $zkSync$ و $StarkNet$ که برای هر دسته از تراکنش‌های $Off-chain$، یک «اثبات رمزنگاری» ($cryptographic\ proof$) تولید می‌کنند. این اثبات به صورت ریاضی تضمین می‌کند که تمام تراکنش‌ها معتبر بوده‌اند، بدون اینکه نیاز به افشای خود تراکنش‌ها باشد.

مزایا و فرآیند مهاجرت: مهاجرت به یک لایه ۲ سازگار با EVM (مانند $Arbitrum$ یا $Optimism$) از نظر فنی بسیار ساده است، زیرا قرارداد هوشمند $Solidity$ فعلی می‌تواند تقریباً بدون هیچ تغییری در آنجا مستقر شود. این کار می‌تواند هزینه‌های تراکنش را ۱۰ تا ۱۰۰ برابر کاهش داده و توان پردازشی را به چندین هزار تراکنش در ثانیه برساند. این بهبود عملکرد، استقرار سامانه را برای زنجیره‌های تأمین با حجم بالا کاملاً اقتصادی و عملیاتی می‌سازد.

پیشنهاد سوم: افزایش حریم خصوصی با اثبات با دانش صفر ($Zero-Knowledge\ Proofs$)

مسئله: شفافیت کامل یک زنجیره بلوکی عمومی، اگرچه برای مصرف‌کننده یک مزیت است، اما برای شرکت‌ها می‌تواند یک ریسک رقابتی باشد. افشای اطلاعاتی مانند حجم دقیق معاملات، هویت شرکای تجاری و قیمت‌گذاری‌ها، می‌تواند به رقبا اطلاعات استراتژیک ارزشمندی بدهد.

راهکار پیشنهادی: استفاده از فناوری پیشرفته «اثبات با دانش صفر» ($Zero-Knowledge\ Proofs$ یا $ZKPs$) برای ایجاد حریم خصوصی در عین حفظ قابلیت اعتبارسنجی. $ZKPs$ یک پروتکل رمزنگاری است که به یک طرف (اثبات‌کننده) اجازه می‌دهد به طرف دیگر (تأییدکننده) ثابت کند که یک گزاره صحیح است، بدون اینکه هیچ اطلاعات اضافی به جز صحت خود گزاره را فاش کند.

کاربرد در سامانه: می‌توان یک نسخه جدید از قرارداد هوشمند را طراحی کرد که با $ZKPs$ (به ویژه $zk-SNARKs$) کار کند. در این مدل:

- یک تولیدکننده می‌تواند مالکیت یک محصول را به یک توزیع‌کننده منتقل کند.
- به جای ثبت عمومی این تراکنش، آن‌ها یک اثبات ZK تولید کرده و به قرارداد هوشمند ارسال می‌کنند.
- این اثبات، به صورت ریاضی به قرارداد ثابت می‌کند که یک انتقال معتبر (از یک مالک قانونی به یک گیرنده) انجام شده است، اما هویت فرستنده، گیرنده و شناسه محصول را فاش نمی‌کند.

قرارداد هوشمند تنها این اثبات را تأیید کرده و وضعیت مالکیت را به صورت رمزنگاری‌شده به‌روز می‌کند. این رویکرد، تعادل کاملی بین نیاز به حسابرسی و نیاز به محرمانگی تجاری برقرار می‌کند و یکی از پیشرفته‌ترین مسیرهای تحقیقاتی در حوزه زنجیره بلوکی است.

پیشنهاد چهارم: توسعه یک سازمان خودگردان غیرمتمرکز (DAO) برای حاکمیت

مسئله: در یک شبکه زنجیره تأمین واقعی که متشکل از چندین شرکت رقیب است، چه کسی پروتکل را کنترل می‌کند؟ چه کسی مسئول به‌روزرسانی قرارداد هوشمند، افزودن نقش‌های جدید یا تغییر پارامترها (مانند نرخ مالیات) است؟ اگر یک مدیر مرکزی (مانند توسعه‌دهنده اولیه سیستم) این قدرت را در دست داشته باشد، ما مجدداً به مشکل تمرکزگرایی بازگشته‌ایم.

راهکار پیشنهادی: ایجاد یک سازمان خودگردان غیرمتمرکز ($Decentralized\ Autonomous\ Organization$ یا DAO) برای مدیریت و حاکمیت پروتکل. DAO یک نهاد است که قوانین آن در قالب قراردادهای هوشمند نوشته شده و فرآیند تصمیم‌گیری آن توسط اعضای آن و از طریق مکانیزم‌های رأی‌گیری $On-chain$ انجام می‌شود.

ساختار پیشنهادی:

۱. یک توکن حاکمیتی جدید (مثلاً $SCMGOV$) ایجاد می‌شود که بین تمام شرکت‌کنندگان معتبر شبکه (تولیدکنندگان، توزیع‌کنندگان و...) توزیع می‌گردد.

۲. دارندگان این توکن می‌توانند پیشنهاداتی برای تغییر در پروتکل ارائه دهند (مثلاً «پیشنهاد افزودن نقش جدید بازرس کیفیت»).

۳. این پیشنهادات به صورت عمومی به رأی گذاشته می‌شود و وزن رأی هر عضو، متناسب با تعداد توکن‌های حاکمیتی است که در اختیار دارد.

۴. اگر یک پیشنهاد رأی کافی را کسب کند، قرارداد حاکمیت به صورت خودکار، تغییرات لازم را در قرارداد اصلی اعمال می‌کند (مثلاً با فراخوانی تابع $grantRole$ برای یک نقش جدید).

این ساختار، حاکمیت شبکه را به صورت دموکراتیک و شفاف به دست خود ذی‌نفعان می‌سپارد و یک مدل پایدار برای تکامل و رشد بلندمدت پروتکل فراهم می‌کند.

پیشنهاد پنجم: ادغام با اکوسیستم مالی غیرمتمرکز ($DeFi$)

فرصت: دارایی‌های توکنیزه شده در این سامانه (محصولات در انبار، محموله‌های در حال حمل) دارای ارزش اقتصادی واقعی هستند. این ارزش در سیستم‌های سنتی به صورت سرمایه «راکد» باقی می‌ماند.

راهکار پیشنهادی: ایجاد پل ارتباطی بین سامانه زنجیره تأمین و پروتکل‌های مالی غیرمتمرکز ($DeFi$). $DeFi$ اکوسیستمی از برنامه‌های مالی است که بر روی زنجیره بلوکی ساخته شده و خدمات سنتی مانند وام‌دهی، استقراض و بیمه را به صورت غیرمتمرکز ارائه می‌دهد.

موارد کاربرد:

- **تأمین مالی موجودی** ($Inventory\ Financing$): یک شرکت کوچک یا متوسط می‌تواند موجودی کالاهای خود در انبار (که به صورت توکن‌های $ERC-1155$ در کیف پول او وجود دارد) را به عنوان وثیقه در یک پروتکل وام‌دهی مانند $Aave$ یا $Compound$ قفل کرده و در ازای آن، یک وام به صورت استیبل‌کوین دریافت کند. این فرآیند که به صورت کاملاً خودکار و آنی توسط قراردادهای هوشمند انجام می‌شود، می‌تواند سرمایه در گردش بسیار مورد نیاز را برای کسب‌وکارها فراهم آورد.
- **بیمه غیرمتمرکز محموله‌ها:** می‌توان با استفاده از پروتکل‌های بیمه غیرمتمرکز، یک محصول را در برابر ریسک‌های حمل‌ونقل (مانند خروج از بازه دمایی مجاز که توسط سنسورهای IoT گزارش می‌شود) بیمه کرد. پرداخت خسارت نیز می‌تواند به صورت خودکار و بر اساس داده‌های غیرقابل انکار اوراکل‌ها انجام شود.

این ادغام، مرز بین دنیای لجستیک و دنیای مالی را از بین برده و مدل‌های کسب‌وکار کاملاً جدید و کارآمدی را امکان‌پذیر می‌سازد. در نهایت، این نقشه راه نشان می‌دهد که پروژه حاضر، نه تنها یک راهکار کامل برای یک مشکل مشخص، بلکه یک پلتفرم پایه‌ای است که می‌توان بر روی آن، لایه‌های جدیدی از نوآوری در حوزه‌های مختلف، از حاکمیت و حریم خصوصی گرفته تا مالی، ایجاد کرد.

منابع و مراجع

[1] O'Reilly, Tim. What is web 2.0: Design patterns and business models for the next generation of software. O'Reilly Media, 2005.

[2] Zuboff, Shoshana. The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power. PublicAffairs, 2019.

[3] Nakamoto, Satoshi. Bitcoin: A peer-to-peer electronic cash system, 2008.

[4] Buterin, Vitalik. Ethereum white paper: A next-generation smart contract and decentralized application platform, 2014.

[5] Wood, Gavin. Ethereum: A secure decentralised generalised transaction ledger. Yellow paper, Ethereum Project, 2014.

[6] Chopra, Sunil and Meindl, Peter. Supply Chain Management: Strategy, Planning, and Operation. Pearson, 7th ed. , 2019.

[7] Lee, Hau L, Padmanabhan, V, and Whang, Seungjin. The bullwhip effect in supply chains. Sloan Management Review, 38(3):93–102, 1997.

[8] Wamba, Samuel Fosso, Lefebvre, Louis A, Bendavid, Ygal, and Lefebvre, Élisabeth. Rfid-enabled supply chain management: a literature review. Production Planning & Control, 26(12):1031–1050, 2015.

[9] Gubbi, Jayavardhana, Buyya, Rajkumar, Marusic, Slaven, and Palaniswami, Marimuthu. Internet of things (iot): A vision, architectural elements, and future directions. Future Generation Computer Systems, 29(7):1645–1660, 2013.

[10] Androulaki, Elli, Barger, Artem, Borkowski, Vita, Cachin, Christian, Christidis, Konstantinos, De Caro, Angelo, Enyeart, David, Ferris, Christopher, Laventman, Gennady, Mane, Yacov, et al. Hyperledger fabric: a distributed operating system for permissioned blockchains. in Proceedings of the thirteenth EuroSys conference, pp. 1–15, 2018.

[11] Kamilaris, Andreas, Fonts, Angels, and Prenafeta-Boldú, Francesc X. The rise of blockchain technology in agriculture and food supply chains. Trends in Food Science & Technology, 91:640–652, 2019.