



دانشکده مهندسی کامپیوتر

ارائه راهکاری برای یکپارچه سازی دستگاه‌های اینترنت اشیاء با سکوی کوبرنیتز

پروژه کارشناسی مهندسی کامپیوتر گرایش مهندسی نرم افزار

سینا شعبانی کومله

استاد راهنما

دکتر محسن شریفی

تابستان ۱۴۰۲

تأییدیه‌ی هیأت داوران جلسه‌ی دفاع از پروژه

نام دانشکده: دانشکده مهندسی کامپیوتر

نام دانشجو: سینا شعبانی کومله

عنوان پروژه: ارائه راهکاری برای یکپارچه سازی دستگاه‌های اینترنت اشیا با سکوی کوبرنیتز

تاریخ دفاع: تابستان ۱۴۰۲

رشته: مهندسی کامپیوتر

گرایش: مهندسی نرم‌افزار

ردیف	سمت	نام و نام خانوادگی	مرتبه دانشگاهی	دانشگاه یا مؤسسه	امضا
۱	استاد راهنما	دکتر محسن شریفی	استاد تمام	دانشگاه علم و صنعت ایران	
۲	داور نهایی	دکتر TODO	دانشیار	دانشگاه علم و صنعت ایران	

تأییدیه‌ی صحت و اصالت نتایج

باسمه تعالی

اینجانب سینا شعبانی کومله به شماره دانشجویی 97521351 دانشجوی رشته مهندسی کامپیوتر مقطع تحصیلی کارشناسی تأیید می‌نمایم که کلیه‌ی نتایج این پروژه حاصل کار اینجانب و بدون هرگونه دخل و تصرف است و موارد نسخه‌برداری شده از آثار دیگران را با ذکر کامل مشخصات منبع ذکر کرده‌ام. در صورت اثبات خلاف مندرجات فوق، به تشخیص دانشگاه مطابق با ضوابط و مقررات حاکم (قانون حمایت از حقوق مؤلفان و مصنفان و قانون ترجمه و تکثیر کتب و نشریات و آثار صوتی، ضوابط و مقررات آموزشی، پژوهشی و انضباطی ...) با اینجانب رفتار خواهد شد و حق هرگونه اعتراض در خصوص احقاق حقوق مکتسب و تشخیص و تعیین تخلف و مجازات را از خویش سلب می‌نمایم. در ضمن، مسئولیت هرگونه پاسخگویی به اشخاص اعم از حقیقی و حقوقی و مراجع ذیصلاح (اعم از اداری و قضایی) به عهده‌ی اینجانب خواهد بود و دانشگاه هیچ‌گونه مسئولیتی در این خصوص نخواهد داشت.

نام و نام خانوادگی: سینا شعبانی کومله

تاریخ و امضا:

مجوز بهره‌برداری از پایان‌نامه

بهره‌برداری از این پایان‌نامه در چهارچوب مقررات کتابخانه و با توجه به محدودیتی که توسط استاد راهنما به شرح زیر

تعیین می‌شود، بلامانع است:

☐ بهره‌برداری از این پایان‌نامه برای همگان بلامانع است.

☐ بهره‌برداری از این پایان‌نامه با اخذ مجوز از استاد راهنما، بلامانع است.

☐ بهره‌برداری از این پایان‌نامه تا تاریخ ممنوع است.

استاد راهنما: دکتر محسن شریفی

تاریخ:

امضا:

چکیده

در حال حاضر، مدیریت و نظارت بر دستگاه‌های اینترنت اشیا¹ به یک چالش عمده تبدیل شده است. راه حل‌های موجود برای کنترل و نظارت بر این دستگاه‌ها اغلب ناسازگاری‌ها و محدودیت‌هایی دارند که موجب کاهش کارایی و پیچیدگی مدیریت در مقیاس بالا می‌شوند. به منظور حل این مسئله، این پروژه تلاش می‌کند تا با استفاده از کوبرنیتز² و پروژه کوبلت مجازی³ یک سازوکار جامع برای نظارت و مدیریت دستگاه‌های اینترنت اشیا ارائه دهد. انگیزه اصلی پروژه متمرکز کردن کنترل و نظارت بر دستگاه‌های اینترنت اشیا به صورت یکپارچه و موثر است. راه حل‌های کنونی اغلب ناسازگاری‌هایی با استانداردها و فناوری‌های مختلف دستگاه‌های اینترنت اشیا دارند و به تنهایی قادر به ارائه یک محیط یکپارچه برای مدیریت و نظارت نیستند. این پروژه شامل سه بخش اصلی، یعنی تامین‌کننده⁴، کنترل‌کننده⁵ و دستگاه‌ها⁶ این بخش‌ها با یکدیگر ارتباط برقرار میکنند تا اطلاعات مفیدی درباره دستگاه‌های اینترنت اشیا مورد کنترل ارائه دهند و این اطلاعات را در دسترس خوشه کوبرنیتز قرار دهند.

واژگان کلیدی: اینترنت اشیا، کوبرنیتز، کوبلت مجازی، نظارت یکپارچه، پایش مقیاس پذیر

Internt Of Things (IoT)¹

Kubernetes²

Virtual Kubelet³

Provider⁴

Controller⁵

Device⁶

فهرست مطالب

ح فهرست تصاویر

خ فهرست جداول

1 فصل 1: مقدمه

1-1 شرح مسأله 1

2-1 اهداف پروژه 2

3-1 ساختار گزارش 2

3 فصل 2: مفاهیم پایه

1-2 مقدمه 3

2-2 بستر ابری 3

3-2 سکوی کوبرنیتز 4

1-3-2 کانتینر 5

2-3-2 پاد 6

3-3-2 گره 6

4-3-2 کوبلت 7

5-3-2 خوشه کوبرنیتز 8

4-2 قرارداد انتقال فرا متن 8

1-4-2 توابع قرارداد انتقال فرا متن 9

2-4-2 درخواست قرارداد انتقال فرا متن 10

11	پاسخ قرارداد انتقال فرا متن	3-4-2
12	انتقال حالت نمایشی	4-4-2
13	رابط کاربردی قابل برنامه‌ریزی	5-4-2
14	معماری فن‌آوت	5-2
15	کوبلت مجازی	6-2
16	معماری کوبلت مجازی	1-6-2
17	اینترنت اشیاء	7-2
18	معماری بازخوانی	8-2
19	جمع‌بندی	9-2
20	کارهای مرتبط	فصل 3:
20	مقدمه	1-3
21	روش پیشنهادی	فصل 4:
21	مقدمه	1-4
21	معماری سامانه	2-4
22	پیاده‌سازی تامین‌کننده کوبلت مجازی	1-2-4
24	رابط برقراری ارتباط با کنترل‌کننده‌ها	2-2-4
24	شبیه‌ساز	3-2-4
25	رابط گرافیکی	4-2-4
27	نحوه کارکرد	3-4
32	ارزیابی روش پیشنهادی	فصل 5:
32	مقدمه	1-5
33	نتیجه‌گیری و کارهای آینده	فصل 6:
33	نتیجه‌گیری	1-6
33	دستاوردها	2-6

3-6 کارهای آینده 33

واژه‌نامه انگلیسی به فارسی 35

فهرست تصاویر

1-2	معماری کلی کوبرنیتز	5
2-2	نمای کلی درخواست و پاسخ قرارداد انتقال فرا متن	9
3-2	نمای کلی از انتقال حالت نمایشی	13
4-2	نمای کلی از رابط کاربردی قابل برنامه‌ریزی	14
5-2	نمای کلی از معماری فن آوت	15
6-2	کوبلت مجازی در یک خوشه کوبرنیتز	16
7-2	نمای کلی از اینترنت اشیاء	18
1-4	نمای کلی معماری	22
2-4	گره مجازی	24
3-4	رابط کاربردی قابل برنامه‌نویسی شبیه‌ساز	25
4-4	صفحه اصلی رابط گرافیکی	26
5-4	صفحه گره‌های کوبرنیتز	26
6-4	صفحه کنترل‌کننده‌های کوبرنیتز کوبرنیتز	27
7-4	دستگاه‌های اینترنت اشیاء ساخته شده در شبیه‌ساز	29
8-4	پاد ساخته شده از روی مستند در وضعیت عدم آمادگی	30
9-4	پاد ساخته شده از روی مستند در وضعیت آماده	30
10-4	تغییر وضعیت قفل مرکزی در رابط گرافیکی	31
11-4	تغییر وضعیت پاد بدلیل تغییر وضعیت قفل مرکزی	31

فهرست جداول

فصل 1

مقدمه

1-1 شرح مسأله

یکی از مسائلی که امروزه در زمینه کنترل و پایش دستگاه‌های اینترنت اشیا وجود دارد، عدم یکپارچگی و هماهنگی میان دستگاه‌های مختلف است. این دستگاه‌ها از فناوری‌ها، پروتکل‌ها و استانداردهای متنوعی برای ارتباط و عملکرد استفاده میکنند، که این تنوع باعث پیچیدگی و مشکلاتی در کنترل و پایش مرکزی آنها میشود. به عنوان مثال، در یک بستر اینترنت اشیا ممکن است دستگاه‌هایی با پروتکل‌های ارتباطی مختلف، مانند HTTP¹، MQTT² و CoAP³ داشته باشند که هر کدام نیازمند روش‌ها و فناوری‌های جداگانه برای کنترل و پایش خود هستند. همچنین، دستگاه‌های اینترنت اشیا ممکن است از نظر تکنولوژی و نوع عملکرد با هم تفاوت داشته باشند. برای مثال، یک سنسور دما و یک قفل هوشمند دارای نیازهای کنترل و پایش متفاوتی هستند. این تنوع در دستگاه‌ها باعث پیچیدگی در توسعه و اجرای یک سیستم کنترل یکپارچه می‌شود. محدودیت منابع نیز یک چالش اساسی در محیط‌های اینترنت اشیا است. این دستگاه‌ها منابع محدودی نظیر پردازشگر، حافظه و پهنای باند شبکه دارند که توان محاسباتی آنها را به شدت کاهش می‌دهد. بنابراین، ضرورت بهره‌برداری بهینه از این منابع و مدیریت آنها به منظور افزایش کارایی و بهره‌وری دستگاه‌ها مطرح میشود. همچنین، امنیت و حفاظت از اطلاعات حساس در محیط‌های اینترنت اشیا نیز از اهمیت بالایی برخوردار است، زیرا این دستگاه‌ها اطلاعاتی حساس را در محیط شبکه منتقل میکنند که به تهدیدات امنیتی از جمله نفوذ، جاسوسی و دسترسی

¹Hypertext Transfer Protocol

²Message Queuing Telemetry Transport

³Constrained Application Protocol

غیرمجاز معرض هستند.

2-1 اهداف پروژه

هدف اصلی پروژه امکان‌سنجی، طراحی، پیاده‌سازی و ارزیابی سیستمی برای کنترل و پایش دستگاه‌های اینترنت اشیا بر سکوی کوبنیتز است.

3-1 ساختار گزارش

در این پروژه هدف ارائه روشی نو برای حل مسئله کنترل و پایش دستگاه‌های اینترنت اشیا بر سکوی کوبنیتز است. در ابتدا به معرفی مفاهیم پایه استفاده شده در این پروژه و سپس به معرفی روش‌ها و کارهای مرتبط پرداخته خواهد شد. پس از آن به معرفی روش و ارزیابی آن پرداخته شده است. در انتها نتیجه‌گیری و کارهای آینده معرفی می‌شوند.

فصل 2

مفاهیم پایه

1-2 مقدمه

در این بخش به معرفی مفاهیم پایه درباره سکوی کوبرنیتز، پروژه کوبلت مجازی، اینترنت اشیاء و برخی معماری‌هایی که در این پروژه استفاده شده‌اند به مانند استخر کارگران¹ پرداخته شده است.

2-2 بستر ابری

بستر ابری² به محیطی اشاره دارد که منابع محاسباتی، شبکه و ذخیره سازی را برای ارائه خدمات به صورت ابری و توسط یک ارائه دهنده ابری فراهم می کند. در این محیط، خدمات و برنامه‌ها بر روی خدمات دهنده‌های فیزیکی مجازی سازی شده قرار می گیرند و کاربران می توانند به آنها از طریق اینترنت وصل شوند و از آنها استفاده کنند. با استفاده از محیط ابری، امکاناتی مانند انعطاف پذیری بالا، قابلیت مقیاس پذیری، اشتراک گذاری منابع و مدیریت آسانتر برای خدمات فراهم می شود. همچنین یکی از مزیت‌های بستر ابری ساده سازی ساخت، مدیریت و انتشار یک خدمت می باشد.

Worker Pool¹
Cloud Environment²

3-2 سکوی کوبرنیتز

کوبرنیتز³ یک سامانه مدیریت کانتینرها⁴ است که توسط گوگل توسعه داده شده است و در حال حاضر تحت نظارت و پشتیبانی مؤسسه CNCF⁵ قرار دارد. این ابزار به توسعه‌دهندگان و مدیران سامانه امکان می‌دهد برنامه‌ها و خدمات را در بسترهای ابری⁶ مدیریت کنند و کانتینرها را به طور موثر و مقیاس‌پذیر در محیط‌های توزیع شده⁷ مدیریت کنند.

از طریق کوبرنیتز، می‌توان کانتینرها را بر روی یک خدمت دهنده مجازی⁸ یا فیزیکی⁹ اجرا کرده و مدیریت آنها را ساده‌تر و مؤثرتر نمود. این سامانه با بهره‌گیری از روش‌هایی مانند اتوماسیون¹⁰، توازن بار¹¹ و تشخیص خودکار اشکال¹²، مدیریت و کنترل بهبود یافته‌ای در محیط‌های مبتنی بر کانتینر فراهم می‌کند. این ابزار برای حل مشکلات زیر موثر است:

1. مقیاس‌پذیری: کوبرنیتز می‌تواند تعداد کانتینرها و پیش‌نمونه‌های برنامه را بر اساس نیازهای ترافیک و خدمت تنظیم کند. با استفاده از مدیریت منابع مبتنی بر درخواست، میزان منابع مورد استفاده توسط برنامه را به طور خودکار تنظیم می‌کند و مقیاس‌پذیری عمودی و افقی را به راحتی فراهم می‌کند.

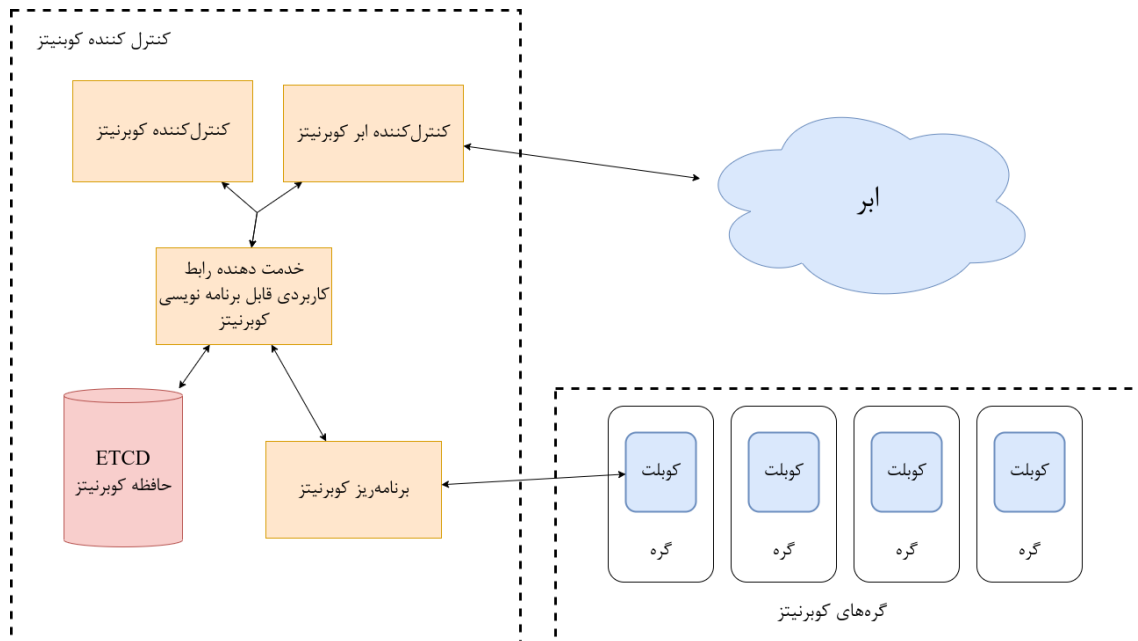
2. توازن بار: با استفاده از کوبرنیتز، می‌توان بار کار را به طور متوازن بین نودها و خدمت دهنده‌های مختلف تقسیم کرد. این باعث بهبود عملکرد و عدم وقوع اختلال در سامانه می‌شود. همچنین، در صورتی که یک نود یا خدمت دهنده دچار مشکل شود، کوبرنیتز به طور خودکار کار را به سایر نودها منتقل می‌کند.

3. مدیریت پیچیدگی: کوبرنیتز امکاناتی را برای مدیریت پیچیدگی سامانه‌های کانتینری فراهم می‌کند. این ابزار اجرا، مدیریت، نظارت و زندگی دوباره سازی کانتینرها را ساده می‌کند. همچنین امکاناتی برای مدیریت تنظیمات، آپدیت‌ها، و تغییرات در حال اجرا نیز در اختیار کاربران قرار می‌دهد.

4. قابلیت انتقال: کوبرنیتز امکان انتقال برنامه‌ها و خدمات بین بسترهای مختلف را فراهم می‌کند. با استفاده از این امکان، می‌توان برنامه‌ها را بین محیط‌های توسعه، آزمون و تولید به راحتی منتقل کرد. کوبرنیتز با استفاده از

Kubernetes³Containers⁴Cloud Native Computing Foundation⁵Cloud Environment⁶Distributed Environment⁷Virtual Machine⁸Physical Server⁹automation¹⁰Load Balancing¹¹Automatic Diagnostics¹²

این قابلیت‌ها و خصوصیات، به توسعه‌دهندگان و مدیران سامانه امکان می‌دهد برنامه‌ها را به صورت مؤثر، قابلیت مقیاس‌پذیری و قابل اطمینان در محیط‌های کانتینری مدیریت کنند و عملکرد سامانه را بهبود دهند.



شکل 2-1: معماری کلی کوبرنیتز

1-3-2 کانتینر

کانتینرها یک فناوری پیشرفته در زمینه مدیریت و اجرای نرم‌افزار هستند. یک کانتینر، یک واحد نرم‌افزاری است که تمام نیازمندی‌های لازم برای اجرای یک نرم‌افزار را شامل می‌شود. در واقع، کانتینرها مجموعه‌ای از عملیات سامانه‌ای، کدها و تنظیماتی هستند که با یکدیگر در یک بستر محصور می‌شوند. از ویژگی‌های برجسته کانتینرها، می‌توان به استقلال و حمل‌پذیری آن‌ها اشاره کرد. به عبارتی دیگر، یک کانتینر می‌تواند بدون تغییر و با حفظ کارایی خود، بین بسترها و سامانه‌های عامل منتقل شود. این ویژگی باعث شده است که کانتینرها در صنعت فناوری اطلاعات بسیار محبوب شوند. برای مدیریت کانتینرها، ابزارهای مختلفی وجود دارند. یکی از محبوب‌ترین ابزارها برای مدیریت کانتینرها، داکر¹³ است. داکر یک بستر توسعه نرم‌افزار مبتنی بر کانتینر است که به توسعه‌دهندگان امکان می‌دهد تا برنامه‌های خود را در یک

¹³Docker

کانتینر قرار داده و آن را در هر سامانه‌ای اجرا کنند. با استفاده از کانتینرها، عملیات توسعه، آزمون و استقرار نرم‌افزارها سریع‌تر و ساده‌تر می‌شود. با توجه به این که هر کانتینر دارای محیط مستقلی است، احتمال بروز تداخل بین برنامه‌ها به حداقل می‌رسد و تغییرات در یک کانتینر بر روی سایر کانتینرها تأثیری نمی‌گذارد. همچنین، سبک بودن کانتینرها امکان مقیاس‌پذیری بالایی را فراهم می‌آورد.

2-3-2 پاد

پادها¹⁴ در کوبرنیتز واحد اصلی اجرا و مدیریت برنامه‌ها و خدمات هستند. یک پاد شامل یک یا چند کانتینر مرتبط است که به صورت مشترک منابع شبکه و ذخیره‌سازی را به اشتراک می‌گذارند. همچنین، هر پاد دارای یک آدرس یکتا درون کلاستر است. پادها به صورت لایه‌ای مجازی شبیه‌سازی می‌شوند و انتزاعی از یک ماشین مجازی یا سامانه عامل فیزیکی هستند. این انتزاع به برنامه‌ها امکان می‌دهد تا بدون احتیاج به اطلاعات جزئیات بستری که برای اون اجرا می‌شوند، در محیط کنترلی کوبرنیتز اجرا شوند. بنابراین، پادها برای توسعه‌دهندگان و مدیران سامانه، یک واسطه سطح بالا و یک فضای کاری است.

2-3-3 گره

در بستر کوبرنیتز، گره‌ها¹⁵ از اجزای کلیدی هستند که برنامه‌ها و خدمات در آن‌ها اجرا می‌شوند. یک گره معمولاً یک خدمت‌دهنده فیزیکی یا ماشین مجازی است که بر روی آن کانتینرها اجرا می‌شوند. هر گره شامل عناصر زیر است:

1. پلتفرم سخت‌افزاری: این شامل خدمت‌دهنده‌ها، سامانه‌های فیزیکی، یا ماشین‌های مجازی است که منابع سخت‌افزاری مانند پردازنده، حافظه، و دیسک را فراهم می‌کنند. گره‌ها بسته به نیازهای برنامه‌ها و خدمات، می‌توانند از طریق شبکه به یکدیگر متصل شوند.

2. کوبلت¹⁶: مسئول مدیریت و اجرای کانتینرها در گره است. آن بر روی هر گره نصب شده و با کنترل‌کننده‌های کوبرنیتز برای دریافت توصیف کانتینرها و مدیریت آن‌ها در ارتباط است.

¹⁴ پاد
¹⁵ Node
¹⁶ Kublet

3. پروکسی¹⁷: یک کنترل کننده شبکه است که مسئول مدیریت ترافیک شبکه بین کانتینرها در گره است. این عملکرد به ارتباط و مسیریابی درخواستها بین کانتینرها و اجزای دیگر کوبرنیتز مرتبط است.

4. حافظه مشترک¹⁸: گرهها از یک حافظه مشترک برای ذخیره و به اشتراک گذاری اطلاعاتی مانند پیکربندیها و وضعیت گرهها استفاده می کنند. این حافظه مشترک معمولاً از طریق ابزارهای ذخیره سازی مانند ETCD پیاده سازی می شود.

با استفاده از گرهها، کوبرنیتز قادر است برنامهها و خدمات را بر روی یک سری از خدمت دهندهها یا ماشینهای مجازی توزیع کند و به طور همزمان و مقیاس پذیر اجرا کند. این باعث افزایش انعطاف پذیری، بهره وری و پایداری در محیطهای ابری و مجازی می شود.

2-3-4 کوبلت

کوبلت یکی از اجزای اصلی سامانه مدیریت کانتینرها کوبرنیتز است. کوبلت مسئول اجرا و مدیریت کانتینرها در یک گره می باشد. کوبلت در هر گره از خوشه¹⁹ کوبرنیتز نصب شده و وظیفه ای اساسی را بر عهده دارد که شامل موارد زیر است:

1. مدیریت کانتینرها: کوبلت مسئول ساخت و اجرای کانتینرها بر اساس توصیفهایی که از طرف کنترل کنندههای کوبرنیتز به آن ارسال می شود، می باشد. این توصیفها شامل اطلاعاتی مانند نرم افزار مورد نظر، تنظیمات شبکه و منابع مصرفی کانتینر می شوند.

2. پایش²⁰ منابع: کوبلت مسئول نظارت بر منابع مصرفی کانتینرها است و اطلاعات مربوط به استفاده از پردازنده، حافظه، شبکه و دیگر منابع سامانه را جمع آوری کرده و گزارش می دهد. این اطلاعات به کنترل کنندههای کوبرنیتز ارسال می شوند تا بتوانند به طور هوشمند منابع را تخصیص دهند و بهینه سازی منابع را انجام دهند.

3. بروزرسانی و نگهداری کانتینرها: کوبلت مسئول بروزرسانی و نگهداری کانتینرها است. اگر نسخه جدیدی از نرم افزار موجود باشد، کوبلت قادر است آن را دریافت و کانتینرها را بروزرسانی کند. همچنین، در صورت خطا در اجرای کانتینر یا توقف آن، کوبلت تلاش می کند کانتینر را به طور خودکار مجدداً راه اندازی کند.

¹⁷Kube-proxy
¹⁸Shared Memory
¹⁹Cluster
²⁰Monitoring

4. ارتباط با سایر اجزا: کوبلت وظیفه برقراری ارتباط با اجزای دیگر کوبرنیتز را نیز دارد. به عنوان مثال، با کنترل کننده²¹ برای دریافت دستورات مدیریتی، با برنامه ریز²² برای دریافت جدول بندی پیشنهادی و با کنترل کننده شبکه²³ برای تنظیمات شبکه در ارتباط است.

به طور خلاصه، کوبلت یکی از اجزای کلیدی کوبرنیتز است که وظیفه مدیریت و اجرای کانتینرها را در گره های سامانه بر عهده دارد. این کامپوننت از طریق ارتباط با سایر اجزا و دریافت توصیف های مربوطه، به ایجاد و مدیریت یک محیط توزیع شده و مقیاس پذیر برای اجرای برنامه ها و خدمات در کوبرنیتز کمک می کند.

2-3-5 خوشه کوبرنیتز

یک خوشه کوبرنیتز²⁴ یک بستر توزیع شده است که شامل مجموعه ای از گره ها است که برای مدیریت و اجرای برنامه ها و ارائه خدمات از طریق کوبرنیتز استفاده می شود. خوشه کوبرنیتز شامل اجزا و خدماتی متعددی است که با همکاری میان گره ها، برنامه ها را مدیریت می کنند.

2-4 قرارداد انتقال فرا متن

قرارداد انتقال فرا متن²⁵ یک پروتکل ارتباطی است که در اینترنت استفاده می شود و برای انتقال اطلاعات بین خدمت دهنده²⁶ و خدمت گیرنده²⁷ استفاده می شود. به طور کلی، قرارداد انتقال فرا متن به عنوان روشی برای انتقال اطلاعات و محتوا در وب مورد استفاده قرار می گیرد. در یک ارتباط، HTTP خدمت گیرنده درخواستی به خدمت دهنده می فرستد و سپس خدمت دهنده با پاسخ مناسب به درخواست خدمت گیرنده پاسخ می دهد. این درخواست و پاسخ در قالب پیام های متنی انجام می شود، که ممکن است شامل سرآیندها²⁸ و محتوای پیام²⁹ باشند. سرآیندها شامل اطلاعاتی مانند نوع محتوا، تاریخ، طول پیام و سایر جزئیات مربوط به ارتباط است. قرارداد انتقال فرا متن برای انتقال انواع مختلف منابع و اطلاعات در وب استفاده می شود. مثلاً می توان از قرارداد انتقال فرا متن برای دریافت صفحات وب، تصاویر، ویدیوها و

²¹kube-controller-manager

²²kube-scheduler

²³kube-proxy

²⁴Kubernetes Cluster

²⁵Hypertext Transfer Protocol (HTTP)

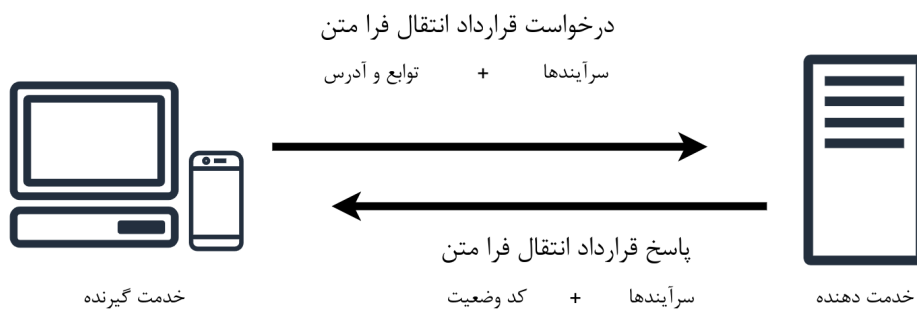
²⁶Server

²⁷Client

²⁸Header

²⁹Body

سایر منابع در خدمت دهنده استفاده کرد. همچنین، قرارداد انتقال فرا متن از مدل درخواست-پاسخ پیروی می‌کند، به این معنی که خدمت گیرنده درخواستی ارسال می‌کند و خدمت دهنده با یک پاسخ مناسب به آن پاسخ می‌دهد. قرارداد انتقال فرا متن اساسی است در عملکرد وب و تعامل بین سرویس دهنده و خدمت گیرنده. این پروتکل به صورت گسترده در نرم افزارها و سرویس های وب مورد استفاده قرار می‌گیرد و امکان انتقال اطلاعات و ارتباط بین کامپیوترها و دستگاه ها را فراهم می‌کند.



شکل 2-2: نمای کلی درخواست و پاسخ قرارداد انتقال فرا متن

1-4-2 توابع قرارداد انتقال فرا متن

قرارداد انتقال فرا متن توابع³⁰ مختلفی را برای تعیین نوع عملیاتی که باید انجام شود تعریف کرده است که عبارت است از:

1. دریافت³¹: این تابع برای دریافت (خواندن) اطلاعات از یک منبع مشخص درخواست می‌شود. مثلاً با استفاده از تابع دریافت می‌توانید صفحات وب، تصاویر یا سایر منابع را از خدمت دهنده دریافت کنید. درخواست دریافت، بدون تغییر و اثری روی منبع مورد درخواست، انجام می‌شود.
2. ساخت³²: این تابع برای ارسال داده‌ها به خدمت دهنده استفاده می‌شود. از تابع ساخت برای ایجاد یا ارسال داده جدید به خدمت دهنده استفاده می‌شود، مانند ارسال فرم‌ها در وب، ارسال نظرات یا انجام عملیات‌های پردازشی.

³⁰Methods متن فرا انتقال قرارداد

³¹GET

³²POST

درخواست ساخت می‌تواند تغییراتی روی منبع مورد درخواست ایجاد کند.

3. بروزرسانی³³: این تابع برای به‌روزرسانی (بازنویسی) یک منبع مشخص استفاده می‌شود. با استفاده از تابع بروزرسانی، می‌توانید اطلاعات موجود در خدمت دهنده را با داده‌های جدید جایگزین کنید. درخواست بروزرسانی می‌تواند تغییراتی روی منبع مورد درخواست اعمال کند یا در صورت عدم وجود، یک منبع جدید ایجاد کند.

4. حذف³⁴: این تابع برای حذف یک منبع مشخص استفاده می‌شود. با استفاده از تابع حذف می‌توانید یک منبع را از خدمت دهنده حذف کنید. درخواست حذف، منبع مورد نظر را از خدمت دهنده حذف می‌کند.

5. بروزرسانی مقطعی³⁵: این تابع برای اعمال تغییرات جزئی روی یک منبع مشخص استفاده می‌شود. با استفاده از تابع بروزرسانی مقطعی، می‌توانید تغییرات کوچکی را روی یک منبع اعمال کنید بدون ایجاد تغییرات بزرگتری در داده‌های موجود.

این توابع به عنوان پایه‌های قرارداد انتقال فرا متن عمل می‌کنند و به خدمت گیرنده و خدمت دهنده امکان ارتباط و انجام عملیات‌های مختلف را می‌دهند. درخواست‌ها و پاسخ‌های قرارداد انتقال فرا متن بر اساس این توابع تعریف می‌شوند و برای تعامل موثر بین سرویس دهنده و خدمت گیرنده استفاده می‌شوند.

2-4-2 درخواست قرارداد انتقال فرا متن

در قرارداد انتقال فرا متن، وقتی که خدمت گیرنده (مانند مرورگر وب یا برنامه‌ای که درخواست ارسال می‌کند) برای دستیابی به منبع مورد نظر خود، درخواستی³⁶ ایجاد می‌کند، یک درخواست قرارداد انتقال فرا متن ایجاد می‌شود. درخواست‌های قرارداد انتقال فرا متن شامل اطلاعاتی درباره نوع و منبع درخواست شده هستند. در زیر توضیحی از مهمترین اجزای یک درخواست قرارداد انتقال فرا متن را می‌یابید:

1. توابع قرارداد انتقال فرا متن: تابع مشخص می‌کند خدمت گیرنده درخواست انجام چه عملیاتی را بر روی منبع مورد نظر دارد.

³³ PUT

³⁴ DELETE

³⁵ PATCH

³⁶ HTTP Request

2. درس³⁷: مشخص می‌کند که منبع مورد نظر کجا قرار دارد و آدرس دقیق آن چیست. آدرس شامل پروتکل، نام دامنه خدمت دهنده³⁸ و مسیر³⁹ منبع مورد نظر است.

3. سرآیندها: درخواست قرارداد انتقال فرا متن می‌تواند شامل سرآیندها باشد که اطلاعات اضافی درباره درخواست و مشخصات خدمت گیرنده را ارائه می‌دهند. به عنوان مثال، سرآیندها می‌توانند شامل اطلاعات مربوط به نوع محتوا، زبان مورد نظر، تاریخ و غیره باشند.

4. بدنه درخواست⁴⁰: بدنه درخواست، در صورتی که درخواست ارسالی دارای داده‌های ارسالی است، این داده‌ها را در خود نگه می‌دارد. بدنه درخواست به صورت متنی یا در قالب داده‌های باینری می‌تواند باشد و معمولاً در درخواست‌هایی مانند ساخت و بروزرسانی استفاده می‌شود.

هنگامی که خدمت گیرنده درخواست قرارداد انتقال فرا متن را به خدمت دهنده ارسال می‌کند، خدمت دهنده با استفاده از اطلاعات درخواست، منبع مورد نظر را پیدا کرده و به مناسبت درخواست، پاسخ مناسب را به خدمت گیرنده ارسال می‌کند. درخواست‌های قرارداد انتقال فرا متن بسته به نوع و عملیات مورد نظر، تعیین می‌کنند که چه عملیاتی باید در سمت خدمت دهنده انجام شود و اطلاعات مربوطه را برگردانند.

2-4-3 پاسخ قرارداد انتقال فرا متن

هنگامی که یک درخواست از سوی خدمت گیرنده به خدمت دهنده ارسال می‌شود، خدمت دهنده با یک پاسخ⁴¹ مناسب به آن درخواست پاسخ می‌دهد. پاسخ‌های قرارداد انتقال فرا متن حاوی اطلاعات مربوط به نتیجه درخواست و وضعیت آن است.

1. کد وضعیت⁴²: هر پاسخ قرارداد انتقال فرا متن دارای یک کد وضعیت است که نشان می‌دهد درخواست با موفقیت انجام شده است یا با مشکل مواجه شده است. برخی از کدهای وضعیت رایج عبارتند از:

(آ) کد ۲۰۰: درخواست با موفقیت انجام شده است و پاسخ داده مورد نظر در دسترس است.

Uniform Resource Locator (URL)³⁷

Domain Name³⁸

Path³⁹

Request Body⁴⁰

HTTP Response⁴¹

Status Code⁴²

(ب) کد ۴۰۴: منبع مورد نظر پیدا نشد.

(ج) کد ۵۰۰: مشکلی در سمت خدمت دهنده رخ داده است که موجب عدم توانایی در ارسال پاسخ مورد نظر شده است.

2. سرآیندها: پاسخ قرارداد انتقال فرا متن شامل سرآیندها است که اطلاعات اضافی در مورد پاسخ و منبع مورد نظر را ارائه می‌دهند. به عنوان مثال، سرآیندها می‌توانند شامل اطلاعات مربوط به نوع محتوا، طول پاسخ، تاریخ ارسال، و غیره باشند.

3. محتوای پاسخ⁴³: پاسخ قرارداد انتقال فرا متن ممکن است حاوی محتوای مورد نظر باشد که توسط خدمت دهنده ارسال می‌شود. محتوای پاسخ می‌تواند اطلاعاتی مانند متن، تصویر، ویدیو یا سایر منابع مورد نیاز را شامل شود.

پاسخ‌های قرارداد انتقال فرا متن ارسال شده توسط خدمت دهنده معمولاً به درخواست‌های ارسالی از سوی مشتری پاسخ می‌دهند و اطلاعات مورد نیاز را در اختیار مشتری قرار می‌دهند. این پاسخ‌ها با کدهای وضعیت، سرآیندها و محتوای پاسخ کامل شده و تعیین می‌کنند که درخواست با موفقیت انجام شده است یا خطا رخ داده است.

4-4-2 انتقال حالت نمایشی

انتقال حالت نمایشی⁴⁴ یک معماری نرم‌افزاری است که برای طراحی سامنه‌های توزیع شده و مبتنی بر وب استفاده می‌شود. انتقال حالت نمایشی بر اساس مجموعه‌ای از اصول و محدودیت‌ها ساختاردهی شده است که در تبادل اطلاعات بین خدمت دهنده و خدمت گیرنده نقش مهمی دارد. یکی از اصول اساسی انتقال حالت نمایشی، تعریف یکپارچگی⁴⁵ برای سامنه است. این به این معنی است که سامنه باید یک مجموعه‌ی مشخص از روش‌های استاندارد را برای تعاملات در نظر بگیرد. معمولاً در سامنه‌های انتقال حالت نمایشی، از متدهای قرارداد انتقال فرا متن مانند دریافت، ساخت، بروزرسانی و حذف برای تعیین عملیات مورد نیاز استفاده می‌شود. همچنین، منابع⁴⁶ در سامنه انتقال حالت نمایشی به صورت یکتا شناسایی می‌شوند و آدرس‌های مشخصی برای آنها تعیین می‌شود. با توجه به این اصول و محدودیت‌ها، سامنه‌های انتقال حالت نمایشی قابلیت‌هایی مانند قابلیت توزیع، قابلیت مقیاس‌پذیری، انعطاف‌پذیری و قابلیت استفاده مجدد را فراهم

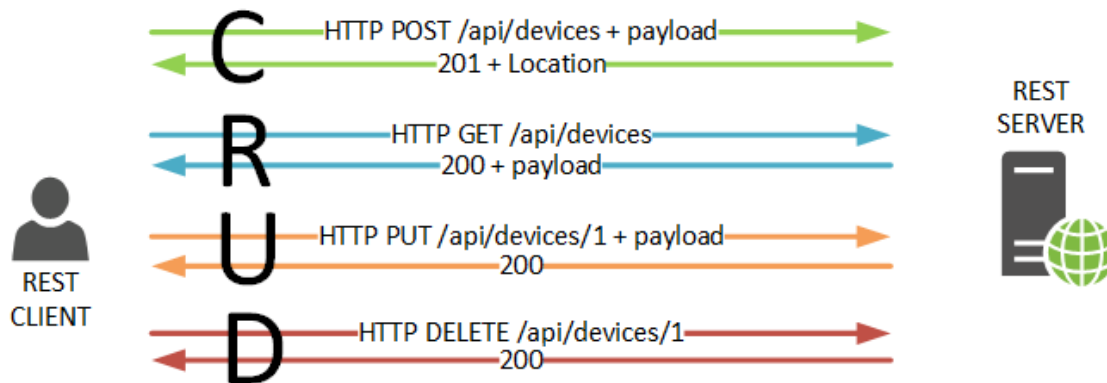
⁴³Response Body

⁴⁴Representational State Transfer (REST)

⁴⁵Uniform Interface

⁴⁶Resources

می‌کنند. از طریق روش‌های استاندارد و اصول انتقال حالت نمایشی، سامانه‌ها قادر به تعامل با یکدیگر بدون توجه به جزئیات داخلی و پیچیدگی‌های نحوه پیاده‌سازی هستند.



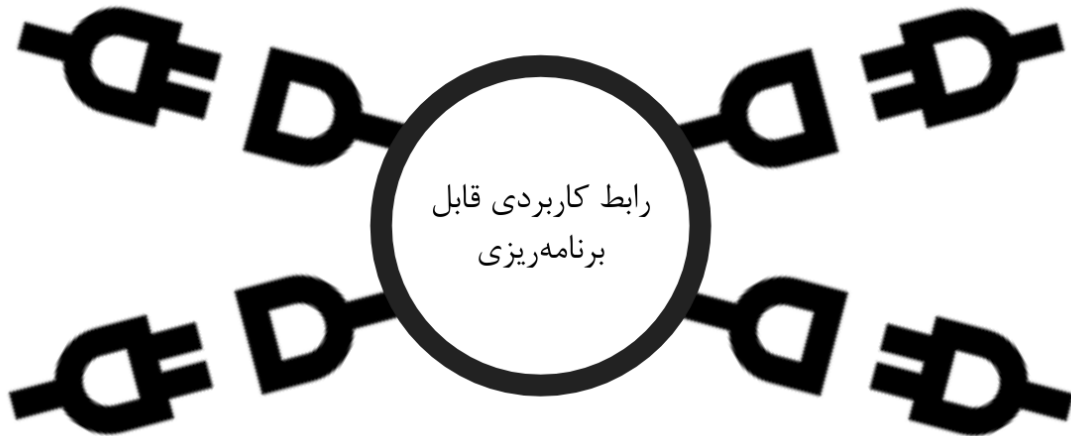
شکل 2-3: نمای کلی از انتقال حالت نمایشی

5-4-2 رابط کاربردی قابل برنامه‌ریزی

رابط کاربردی قابل برنامه‌ریزی⁴⁷ به قوانین، پروتکل‌ها و دستوراتی گفته می‌شود که برای ارتباط و تعامل بین نرم‌افزارها، خدمات و برنامه‌های مختلف به کار می‌رود. به طور کلی، رابط کاربردی قابل برنامه‌ریزی نقش یک میانجی بین سامانه‌ها را بازی می‌کند و به برنامه‌نویسان اجازه می‌دهد با استفاده از آن، به منابع و امکانات موجود در سامانه دیگر دسترسی پیدا کنند. رابط کاربردی قابل برنامه‌ریزی‌ها می‌توانند در دو شکل مختلف عمل کنند: به صورت وب خدمت یا به صورت کتابخانه برنامه‌نویسی. در حالت وب خدمت، رابط کاربردی قابل برنامه‌ریزی بر روی یک خدمت دهنده میزبان شده است و از طریق پروتکل‌های اینترنتی مانند قرارداد انتقال فرا متن قابل دسترسی است. در حالت کتابخانه برنامه‌نویسی، دستورات و توابع مشخصی به برنامه اضافه می‌شوند که برنامه‌نویسان می‌توانند از آنها به عنوان قسمتی از برنامه خود استفاده کنند. استفاده از رابط کاربردی قابل برنامه‌ریزی‌ها به برنامه‌نویسان امکان می‌دهد که بخشی از دستورات را استفاده کنند و از خدمات و قابلیت‌های ارائه شده توسط یک سامانه دیگر بهره ببرند. این راهکار می‌تواند زمان و هزینه توسعه برنامه را کاهش

⁴⁷Application Programming Interface

داده و امکان ادغام بین نرم‌افزارها را فراهم کند. به طور خلاصه، رابط کاربردی قابل برنامه‌ریزی، مانند یک پل ارتباطی است که برنامه‌نویسان می‌توانند از آن استفاده کنند تا بین نرم‌افزارها و خدمات اطلاعات را به اشتراک بگذارند و تعامل کنند.

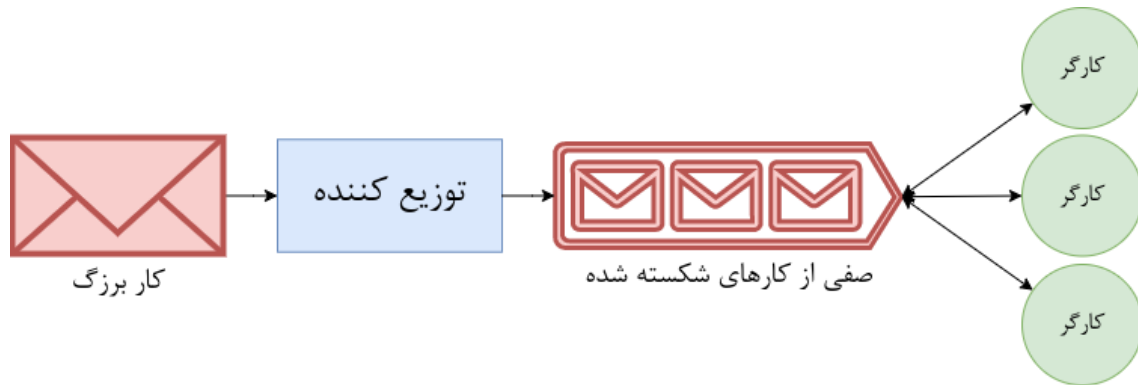


شکل 2-4: نمای کلی از رابط کاربردی قابل برنامه‌ریزی

5-2 معماری فن‌آوت

معماری فن‌آوت⁴⁸: الگوی طراحی است که به طور معمول در سامانه‌های توزیع‌شده برای مدیریت همزمان یا موازی درخواست‌ها استفاده می‌شود. این معماری به توزیع درخواست‌های ورودی به چند واحد پردازش، که به عنوان کارگرها شناخته می‌شوند، برای انجام عملیات مورد نیاز می‌پردازد. معماری فن‌آوت با بهره‌گیری از پردازش موازی و توازن بار، امکان مقیاس‌پذیری و بهبود عملکرد سیستم را فراهم می‌کند. در معماری فن‌آوت، هنگامی که یک درخواست دریافت می‌شود، آن را به چندین کارگر مستقل تکرار یا ارسال می‌کنند تا هر کدام از آن‌ها قادر به پردازش درخواست به صورت

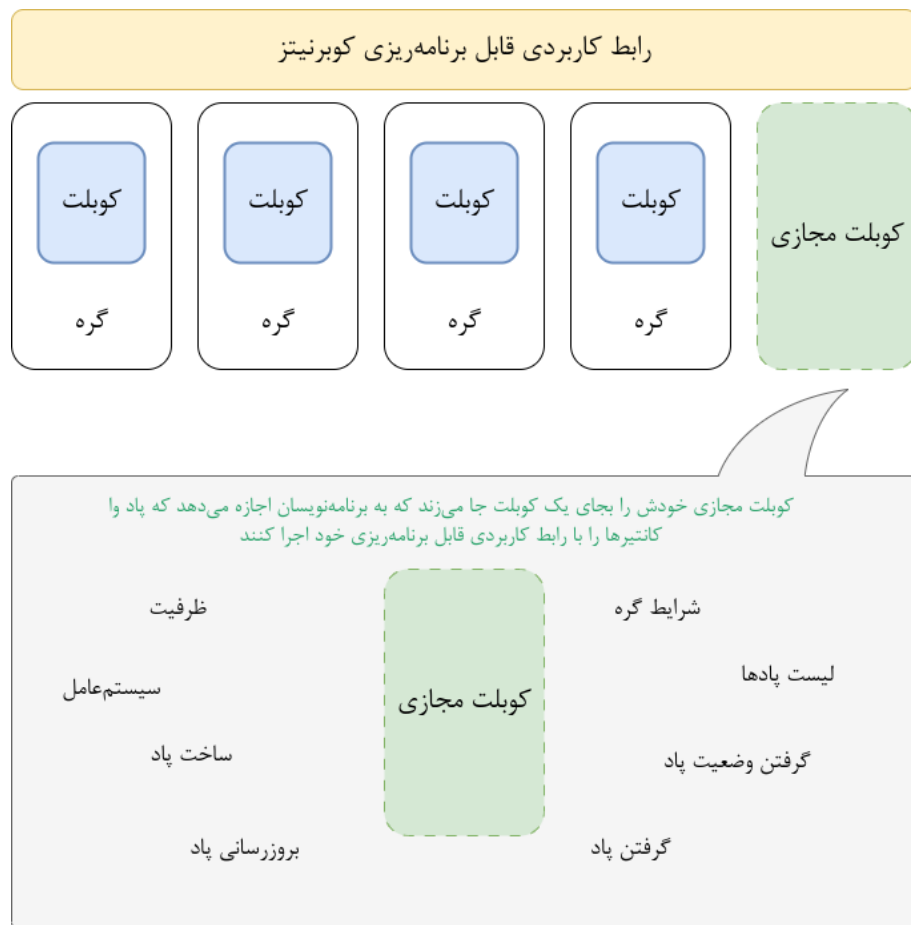
مستقل باشند. این رویکرد به چندین کارگر اجازه می‌دهد تا به صورت همزمان روی بخش‌های مختلف درخواست کار کنند و بهبود ظرفیت تولید و کاهش زمان پاسخ را به همراه داشته باشد.



شکل 2-5: نمای کلی از معماری فن‌آوت

6-2 کوبلت مجازی

کوبلت مجازی⁴⁹ یک پروژه متن باز است که با گسترش رابط کاربردی قابل برنامه‌ریزی کوبرنیتز، امکان ادغام⁵⁰ خدمات و سامانه‌های خارجی را در قالب گره‌های کوبرنیتز فراهم می‌کند. این پروژه این امکان را می‌دهد که یک گره "مجازی" بسازیم که توسط کوبرنیتز قابل مدیریت باشد و امکان ادغام سامانه‌های متنوع را در داخل خوشه کوبرنیتز به صورت یک پارچه فراهم کند.



شکل 2-6: کوبلت مجازی در یک خوشه کوبرنیتز

1-6-2 معماری کوبلت مجازی

کوبلت مجازی از چندین بخش کلیدی تشکیل شده است:

1. تامین‌کننده: تامین‌کننده مسئول پیاده‌سازی رابط کوبلت مجازی است و به عنوان پل ارتباطی بین سامانه یا خدمت خارجی و کوبرنیتز عمل می‌کند. این بخش، فراخوانی‌های رابط کاربردی قابل برنامه‌ریزی کوبرنیتز را به عملیات مناسب در سامانه خارجی ترجمه می‌کند.
2. گره: گره مجازی نمایانگر یک سامانه یا خدمت خارجی در خوشه کوبرنیتز است. عملکرد آن شبیه یک گره کوبرنیتز عادی است، اما به جای اجرا در زیرساخت فیزیکی، از طریق تامین‌کننده با سامانه خارجی ارتباط برقرار می‌کند.

3. یاد: یاد در کورنیتز، شامل یک یا چند کانتینر است. در کوبلت مجازی، پادها نماینده کارها⁵¹ هستند که در گره‌های مجازی ایجاد شده توسط سامانه خارجی اجرا می‌شوند.

4. برنامه‌ریز⁵²: برنامه‌ریز کورنیتز مسئول تخصیص پادها به گره‌های موجود بر اساس نیازهای منابع و محدودیت‌ها است. در کوبلت مجازی، برنامه‌ریز مسئول زمانبندی پادها به گره‌های مجازی ایجاد شده توسط تامین‌کننده است. این بخش تضمین می‌کند که منابع بصورت بهینه استفاده شده و کارها به طور مناسب در سامانه خارجی توزیع شوند.

7-2 اینترنت اشیاء

اینترنت اشیاء⁵³ به مجموعه‌ای از دستگاه‌ها، سنسورها، دستگاه‌های هوشمند و شبکه‌های مرتبط که قادر به تبادل اطلاعات با یکدیگر از طریق اتصال به اینترنت هستند، اشاره دارد. این اشیاء می‌توانند شامل تلفن همراه و ساعت هوشمند تا لوازم خانگی هوشمند، خودروهای متصل و تجهیزات صنعتی باشند. اینترنت اشیاء با اتصال اشیاء و جمع‌آوری اطلاعات، امکان برقراری ارتباط و کنترل بیشتری را بین دنیای فیزیکی و دنیای دیجیتال فراهم می‌کند. مزیت اصلی اینترنت اشیاء در جمع‌آوری و تبادل داده‌ها است. سنسورها و دستگاه‌ها در اینترنت اشیاء می‌توانند اطلاعات مربوط به بستر، شرایط، موقعیت جغرافیایی، وضعیت و داده‌های دیگر را جمع‌آوری کرده و به خدمت‌دهنده‌ها یا سامانه‌های مرکزی ارسال کنند. این اطلاعات در خدمت‌دهنده‌ها تحلیل می‌شوند و می‌توانند به عنوان منبعی برای ارائه داده‌های مفید، تجزیه و تحلیل ترافیک، پیش‌بینی و اتخاذ تصمیم‌های هوشمند استفاده شوند. از جمله کاربردهای اینترنت اشیاء می‌توان به موارد زیر اشاره کرد:

1. خانه هوشمند: اتصال لوازم خانگی مانند تلویزیون، سامانه‌های روشنایی، دستگاه‌های گرمایشی و سرمایشی، سامانه‌های امنیتی و سایر دستگاه‌ها به اینترنت به کاربران امکان می‌دهد تا این دستگاه‌ها را از راه دور کنترل و مدیریت کنند.

2. صنعت هوشمند: در صنعت، اینترنت اشیاء می‌تواند در جمع‌آوری داده‌ها از تجهیزات و سنسورها به منظور نظارت بر فرآیندها، پیشگیری از خرابی‌ها، بهینه‌سازی استفاده از منابع و افزایش بهره‌وری مورد استفاده قرار گیرد.

Workload⁵¹Scheduler⁵²Internet of Things (IoT)⁵³

امکانات شهری مانند مدیریت ترافیک، پارکینگ هوشمند، مدیریت پسماند و رصد آلودگی هوا را فراهم کرد.

بر سلامتی فرد، جمع‌آوری داده‌های پزشکی، پیشگیری از بیماری‌ها و ارائه مراقبت بهتر را فراهم می‌کند.

به طور کلی، اینترنت اشیاء با اتصال اشیاء به اینترنت و جمع‌آوری داده‌ها، امکانات جدیدی را در دسترس قرار می‌دهد و قابلیت‌های جدیدی را برای کنترل، مدیریت و بهبود عملکرد اشیاء فراهم می‌کند.



شکل 2-7: نمای کلی از اینترنت اشیاء

8-2 معماری بازخوانی

نرم‌افزارهای مبتنی بر بازخوانی⁵⁴، الگوها و روش‌هایی در برنامه‌نویسی هستند که در آن بخشی از کد به عنوان ورودی به یک تابع دیگر داده می‌شود، و آن تابع در زمان لازم، این ورودی را به صورت بازخوانی⁵⁵ فراخوانی می‌کند. با این روش، کاربر می‌تواند اجرای برنامه را کنترل کند و در زمان‌های خاص، تابع مورد نظر خود را صدا کند. یکی از استفاده‌های متداول

Callback-Based Software⁵⁴
 Callback⁵⁵

این الگو در پاسخ به وقوع رویدادها است. در این حالت، یک تابع بازخوانی به عنوان ورودی به یک رویداد مرتبط ارسال می‌شود. وقتی که رویداد رخ دهد، نرم‌افزار بازخوانی را فراخوانی می‌کند و عملیات مورد نظر را اجرا می‌کند. این روش به برنامه‌نویس امکان می‌دهد تا همزمان با اجرای دیگر بخش‌های برنامه، به رویدادها واکنش نشان دهد. در نرم‌افزارهای مبتنی بر بازخوانی، تعامل بین بخش‌های مختلف برنامه به صورت غیر همزمان انجام می‌شود. با این روش، برنامه‌نویس می‌تواند به طور کنترل شده تغییرات را پیگیری کند و اقدامات مناسبی را در زمان لازم انجام دهد. این الگو به برنامه‌نویسان امکان می‌دهد تا برنامه‌های پیچیده‌تر و قابل اطمینان‌تری را طراحی کنند، زیرا تعاملات غیر همزمان میان بخش‌ها را کنترل می‌کند و بهبود قابلیت اطمینان سیستم را فراهم می‌کند. بنابراین، نرم‌افزارهای مبتنی بر بازخوانی، با استفاده از تابع بازخوانی به عنوان مکانیزم اصلی برای تعامل بین بخش‌های مختلف برنامه، سبب افزایش انعطاف‌پذیری، کنترل و قابلیت اطمینان در طراحی و پیاده‌سازی نرم‌افزارها می‌شوند.

9-2 جمع‌بندی

فصل 3

کارهای مرتبط

1-3 مقدمه

فصل 4

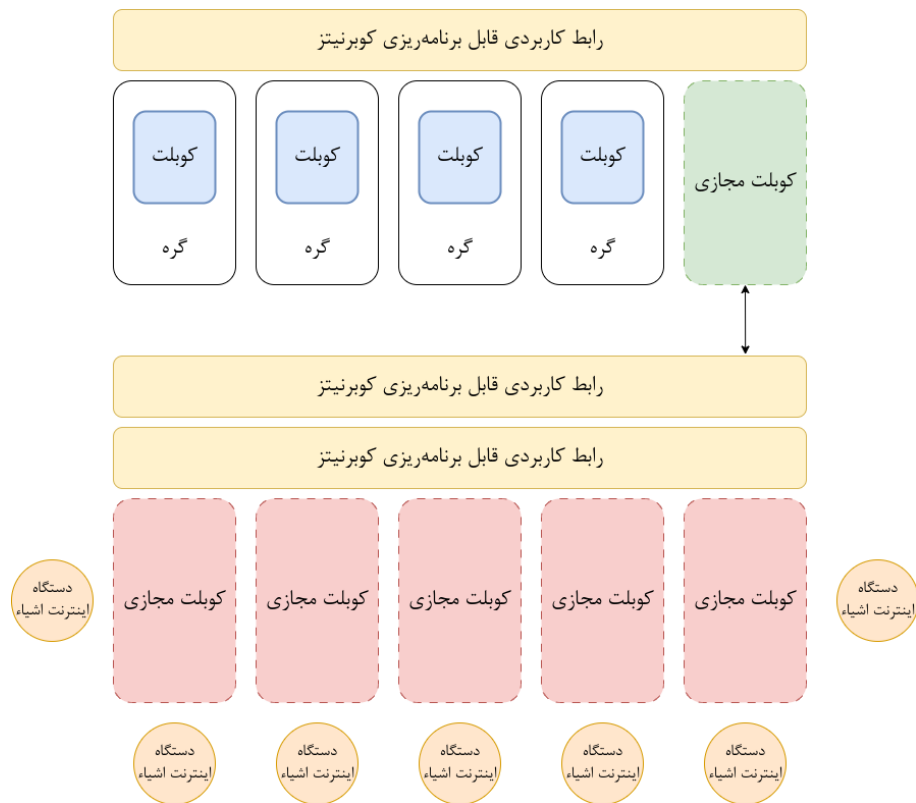
روش پیشنهادی

1-4 مقدمه

TODO

2-4 معماری سامانه

اجزای اصلی سامانه متشکل از پیاده سازی تامین کننده کوبلت مجازی، رابط بین تامین کننده و کنترل کننده ها، کنترل کننده ها و دستگاه ها می باشد که در ادامه به تعریف هر کدام پرداخته



شکل 4-1: نمای کلی معماری

1-2-4 پیاده‌سازی تامین‌کننده کوبلت مجازی

کوبلت مجازی با در اختیار گذاشتن رابط‌هایی برای برنامه‌نویس، این امکان را می‌دهد که بتوان گره‌های کوبرنیتز با پشته‌های سفارشی‌سازی شده داشته باشیم. برای مثال رابط زیر چرخه وجودی یک پاد را نشان می‌دهد. حال با پیاده‌سازی این رابط، ما این امکان را داریم که از ساخته‌شدن، بروزرسانی‌شدن، حذف‌شدن و حتی تغییر وضعیت‌های پادهای مورد نظر خود با خبر شویم.

رابط کنترل‌کننده چرخه وجودی پاد : Listing 4-1

```

type PodLifecycleHandler interface {
    CreatePod(ctx context.Context, pod *corev1.Pod) error

    UpdatePod(ctx context.Context, pod *corev1.Pod) error

    DeletePod(ctx context.Context, pod *corev1.Pod) error

    GetPod(ctx context.Context, namespace, name string)
        (*corev1.Pod, error)

    GetPodStatus(ctx context.Context, namespace, name string)
        (*corev1.PodStatus, error)

    GetPods(context.Context) ([]*corev1.Pod, error)
}

```

پیاده‌سازی این رابط امکان این را می‌دهد که بتوانیم یک پاد بر روی کوبرنیتز اعمال کرده و سپس بوسیله کوبرنیتز فراخوانی شده تا پاد مورد نظر را بسازیم. در این پروژه یک پاد نقش یک دستگاه اینترنت اشیاء را دارد. همچنین برای ارسال وضعیت گره مجازی ساخته شده، نیاز به پیاده‌سازی رابط دیگری داریم.

رابط کنترل‌کننده وضعیت گره : Listing 4-2

```

type NodeProvider interface {
    Ping(context.Context) error

    NotifyNodeStatus(ctx context.Context, cb func(*corev1.Node))
}

```

پیاده‌سازی این رابط باعث می‌شود هنگامی که کد مربوطه در حال اجرا می‌باشد، گره مورد در نظر در خوشه کوبرنیتز بصورت آماده ظاهر شود و اگر این کد متوقف شود، بصورت غیر آماده ظاهر شود. بعد از ثبت این دو رابط و انجام چند مرحله دیگر، تامین‌کننده ما آماده استفاده می‌شود و بصورت یک گره در کوبرنیتز ظاهر خواهد شد.

```
→ examples git:(docs) x kubectl get nodes
NAME        STATUS    ROLES    AGE   VERSION
minikube    Ready    control-plane  59d   v1.26.3
vkube       Ready    agent      5d17h test-v0.0.1
→ examples git:(docs) x
```

شکل 2-4: گره مجازی

2-2-4 رابط برقراری ارتباط با کنترل کننده‌ها

بعد از اتصال به کوبرنیتز و دریافت درخواست‌ها و بروزرسانی‌ها از سوی این سکو، باید با دستگاه‌های اینترنت اشیاء ارتباط برقرار کرده و وضعیتشان را در اختیار کوبرنیتز قرار دهیم. منطق ارتباط با کنترل کننده‌های دستگاه‌های اینترنت اشیاء به این صورت است که بصورت مداوم درخواست‌های خاصی به سمت کنترل کننده‌ها می‌فرستد تا از وضعیت خود کنترل کننده‌ها و همچنین دستگاه‌های اینترنت اشیاء تحت کنترلشان با خبر شود و در صورت نیاز کوبرنیتز را بروزرسانی کند. این درخواست‌ها چیزی نیست جز درخواست‌های قرارداد انتقال فرا متن. همچنین برای اینکه بتوان تعداد زیادی دستگاه اینترنت اشیاء را با یک کنترل کننده، کنترل کرد؛ از روش تکرار مقطعی استفاده شده است. این روش به ما کمک می‌کند تا وضعیت دستگاه‌ها را بصورت مقطعی (نه یکجا) دریافت کرده که بتوان در صورت امکان از همزمانی، برای تسریع کار، استفاده کرد.

این رابط برای اینکه داده‌های مربوط به وضعیت دستگاه‌ها و کنترل کننده‌ها را در اختیار تامین کننده قرار دهد، از معماری بازخوانی¹ استفاده می‌کند.

3-2-4 شبیه‌ساز

در این پروژه یک شبیه‌ساز هم پیاده‌سازی شده که نقش کنترل کننده دستگاه‌های اینترنت اشیاء و همچنین خود این دستگاه‌ها را ایفا می‌کند. این شبیه‌ساز یک خدمت‌دهنده قرارداد انتقال فرامتن می‌باشد که امکانات زیر را هم برای کنترل کننده و هم برای دستگاه‌های اینترنت اشیاء فراهم می‌کند:

1. ساخت

2. ساخت جمعی (برای ارزیابی ساده)

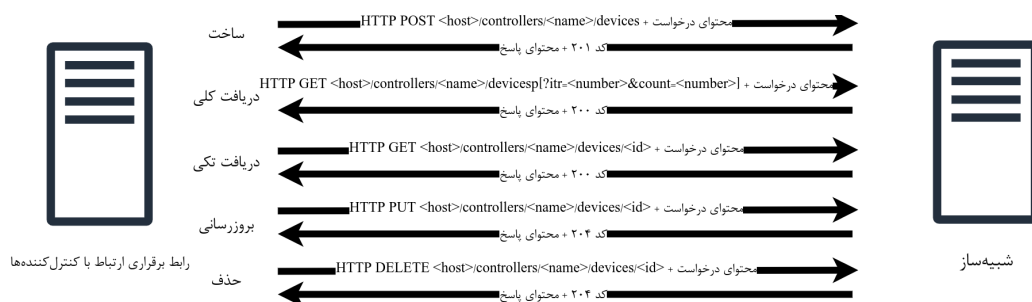
¹ Callback

3. بروزرسانی

4. حذف

5. دریافت تکی، همه و مقطعی

دستگاه‌های اینترنت اشیا شبیه‌سازی شده قفل‌های هوشمند یک ساختمان می‌باشند که امکان باز کردن و بستن قفل را دارند.



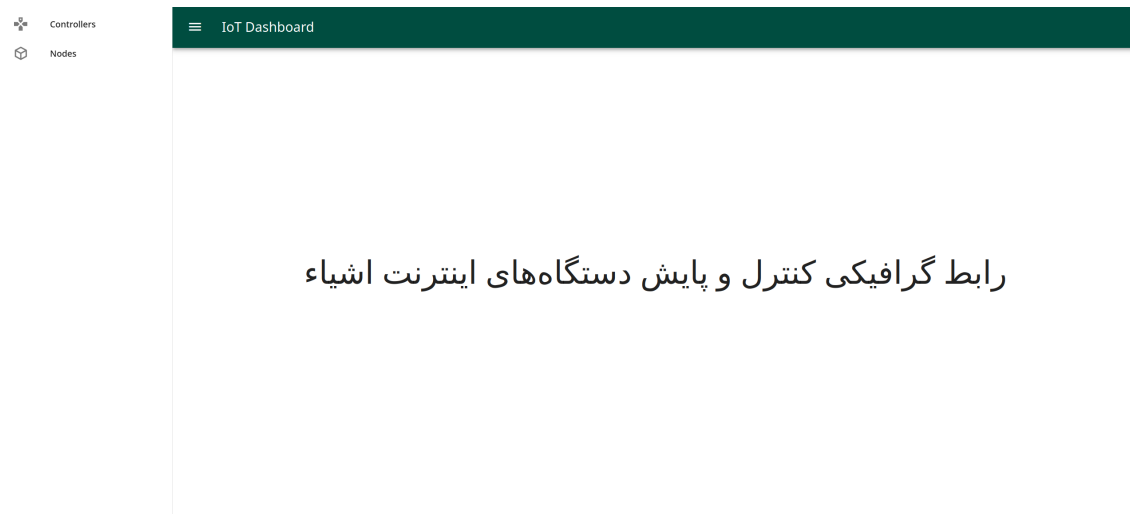
شکل 4-3: رابط کاربردی قابل برنامه‌نویسی شبیه‌ساز

4-2-4 رابط گرافیکی

با توجه به اینکه هدف این پروژه امکان‌سنجی و پیاده‌سازی روشی برای پایش دستگاه‌های اینترنت اشیا بوسیله بستر کوبرنیتز است، اما رابط گرافیکی نیز طراحی شد برای نمایش دادن هرچه بهتر اجزای پروژه. این رابط از دو بخش تشکیل شده است.

1. بخشی که با کنترل‌کننده دستگاه‌های اینترنت اشیا ارتباط دارد و کمک به تسهیل ساخت و نمایش دستگاه‌های اینترنت اشیا می‌کند.

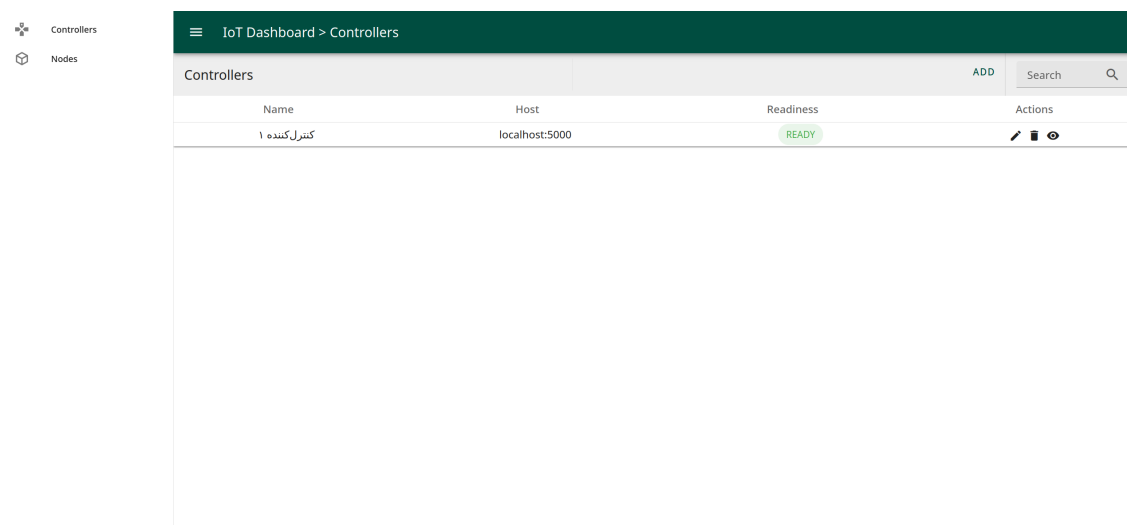
2. بخش دیگر که با تامین‌کننده در ارتباط است و بصورت مداوم وضعیت گره‌ها و پادهای مجازی را بروزرسانی می‌کند.






شکل 4-4: صفحه اصلی رابط گرافیکی

IoT Dashboard > Nodes						
Nodes					Search	
Name	Cpu	Memory	Allocatable Pods	Max Pods	Readiness	Actions
minikube	16	15594756Ki	110	110	READY	
vkube	16	4Gi	10k	10k	READY	

شکل 4-5: صفحه گره‌های کوبرنیتز



IoT Dashboard > Controllers			
Controllers			ADD
Name	Host	Readiness	Actions
کنترل کننده ۱	localhost:5000	READY	  

شکل 4-6: صفحه کنترل کننده‌های کوبرنیتز کوبرنیتز

3-4 نحوه کارکرد

ابتدا باید یک مستند یاد که در ذیل آمده مهیا کرده و در کوبرنیتز اعمال کنیم. بعد از اعمال این مستند، کوبرنیتز تامین کننده را از ایجاد این مستند با خبر می کند. حال تامین کننده با بازفراخوانی رابط کنترل کننده ها منجر به شروع دریافت وضعیت این دستگاه اینترنت اشیا بصورت مداوم از کنترل کننده تعریف شده در مستند می شود. بنابراین رابط کنترل کننده ها بصورت مداوم با رابط کاربردی قابل برنامه نویسی شبیه ساز ارتباط گرفته و وضعیت دستگاه ها را بروزرسانی می کند.

مستند ساخت پاد در کوبرنیتز : Listing 4-3

```
apiVersion: v1
kind: Pod
metadata:
  name: lock-main
  annotations:
    controllerName: "controller1"
    controllerAddress: "localhost:5000"
spec:
  containers:
    # this is so that kubernetes validation will pass
    - image: doesntmatter/smart_lock
      name: lock1
  dnsPolicy: ClusterFirst
  nodeSelector:
    kubernetes.io/role: agent
    kubernetes.io/os: linux
    type: virtual-kubelet
  tolerations:
    # this will target Virtual Kubelets nodes only
    - key: itzloop.dev/virtual-kubelet
      operator: Exists
```

قبل از اعمال مستند بالا، از طریق رابط گرافیکی در شبیه‌ساز چهار دستگاه ساخته که در شکل زیر مشاهده می‌کنید.

Controllers

Nodes

IoT Dashboard > Controllers > Controller1

Devices

ADD

Search

Name	Readiness	Actions
lock-main	READY	<div></div> <div></div>
lock-storage	READY	<div></div> <div></div>
lock-room1	READY	<div></div> <div></div>
lock-room2	READY	<div></div> <div></div>

Items per page:

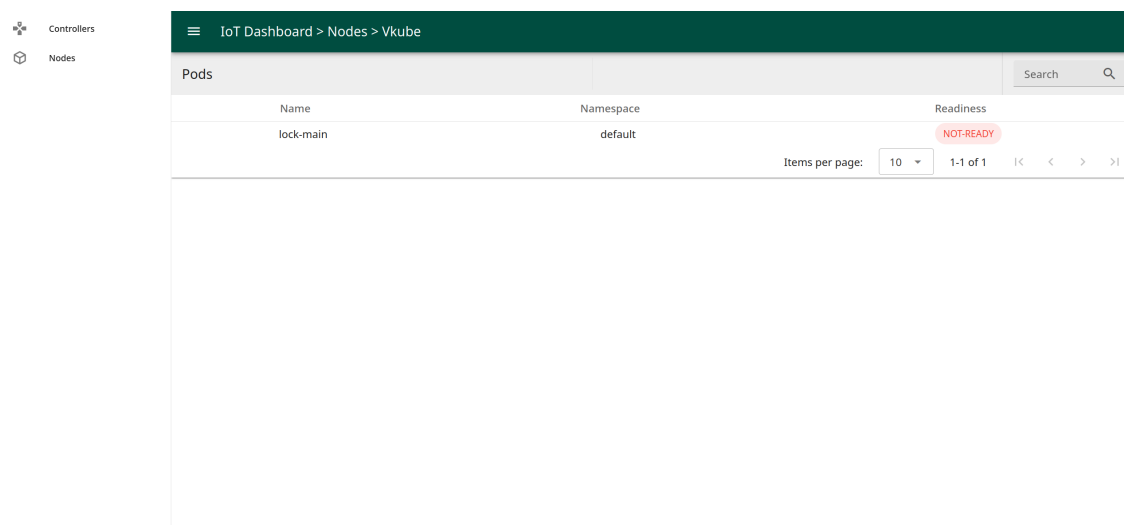
10

1-4 of 4

شکل 4-7: دستگاه‌های اینترنت اشیاء ساخته شده در شبیه‌ساز

در ادامه مستند پاد را اعمال می‌کنیم. همانطور که در مستند آماده است فقط یکی از این دستگاه‌ها یعنی قفل مرکزی² را از طریق کوبرنیتز رصد می‌کنیم. بعد از اعمال این مستند پادهای کوبرنیتز را مشاهده کرده که ابته در وضعیت عدم آمادگی³ می‌باشند.

lock-main²
Not-Ready³

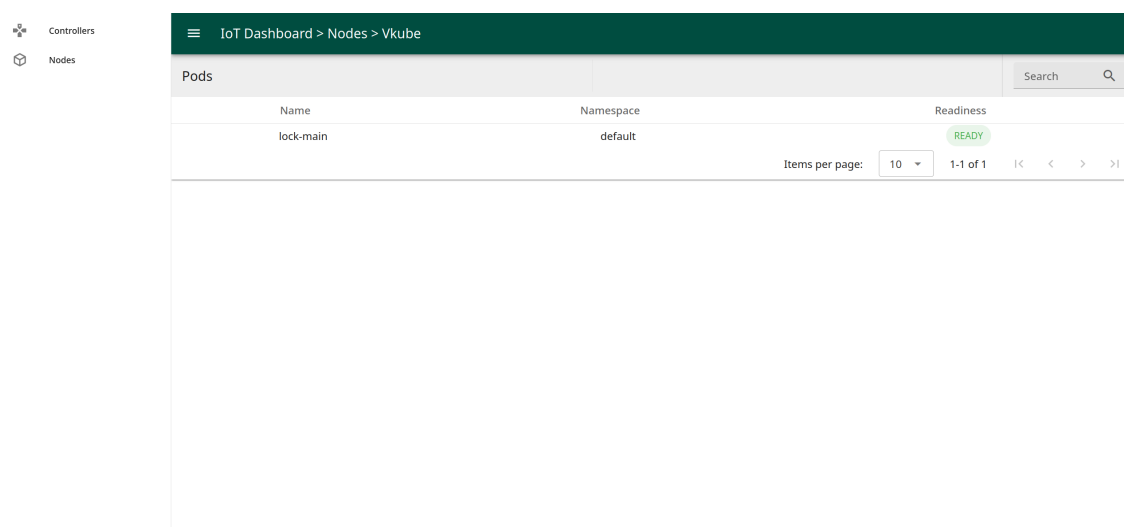


The screenshot shows the 'IoT Dashboard > Nodes > Vkube' interface. On the left, there are tabs for 'Controllers' and 'Nodes'. The main area displays a table of Pods. The table has columns for 'Name', 'Namespace', and 'Readiness'. A single pod named 'lock-main' in the 'default' namespace is shown with a 'NOT-READY' status, indicated by a red label. Below the table, there is a pagination control showing 'Items per page: 10' and '1-1 of 1'.

Name	Namespace	Readiness
lock-main	default	NOT-READY

شکل 4-8: پاد ساخته شده از روی مستند در وضعیت عدم آمادگی

پس از گذر مدتی (مدت زمانی که طول می کشد رابط ارتباط با کنترل کننده ها وضعیت قفل مرکزی را از شبیه ساز دریافت کند) خواهیم دید که پاد مورد نظر در کوبرنیتز به وضعیت آماده⁴ در می آید.



The screenshot shows the same 'IoT Dashboard > Nodes > Vkube' interface. The pod 'lock-main' in the 'default' namespace now has a 'READY' status, indicated by a green label. The pagination control remains the same, showing 'Items per page: 10' and '1-1 of 1'.

Name	Namespace	Readiness
lock-main	default	READY

شکل 4-9: پاد ساخته شده از روی مستند در وضعیت آماده

حال اگر با استفاده از رابط گرافیکی، وضعیت قفل مرکزی در شبیه‌ساز را به وضعیت عدم آمادگی تغییر دهیم خواهیم دید که بعد از مدتی وضعیت پاد نیز تغییر می‌کند.

IoT Dashboard > Controllers > Controller1		
Devices		ADD Search
Name	Readiness ↑	Actions
lock-main	NOT-READY	[edit] [delete]
lock-storage	READY	[edit] [delete]
lock-room1	READY	[edit] [delete]
lock-room2	READY	[edit] [delete]

Items per page: 10 1-4 of 4 |< < > >|

شکل 4-10: تغییر وضعیت قفل مرکزی در رابط گرافیکی

IoT Dashboard > Nodes > Vcube		
Pods		Search
Name	Namespace	Readiness
lock-main	default	NOT-READY

Items per page: 10 1-1 of 1 |< < > >|

شکل 4-11: تغییر وضعیت پاد بدلیل تغییر وضعیت قفل مرکزی

فصل 5

ارزیابی روش پیشنهادی

1-5 مقدمه

فصل 6

نتیجه‌گیری و کارهای آینده

1-6 نتیجه‌گیری

2-6 دستاوردها

3-6 کارهای آینده

واژه‌نامه فارسی به انگلیسی

واژه‌نامه انگلیسی به فارسی

گره	Node
خدمت گیرنده	Client
خدمت دهنده	Server
قرارداد	Protocol
رابط	Interface
رابط کاربردی قابل برنامه‌نویسی	Application Programming Interface
قرارداد انتقال فرا متن	Hypertext Transfer Protocol
پشتوانه	Backend
بازخوانی	Callback

Abstract

TODO

Keywords : Internet of Things, Kubernetes, Virtual Kubelet, Centralized Monitoring, Scalable Monitoring



Iran University of Science and Technology
Computer Engineering Department

A K8-Based Mechanism for Remote Monitoring and Control of IoT Devices

Bachelor of Science Thesis in Computer Engineering – Software Engineering

By:

Sina Shabani Kumeleh

Supervisor:

Dr. Mohsen Sharifi

Summer 2023