به نام خدا

پروژه‌ی پایانی درس رایانش ابری

طراحی تمرین

آرمین ذوالفقاری، محمد توکلی، آریان بوکانی، دانیال حمدی

توسعه‌ی کد

امیرحسین نجفی‌زاده

استاد درس

دکتر جوادی

مهلت نهایی ارسال پاسخ

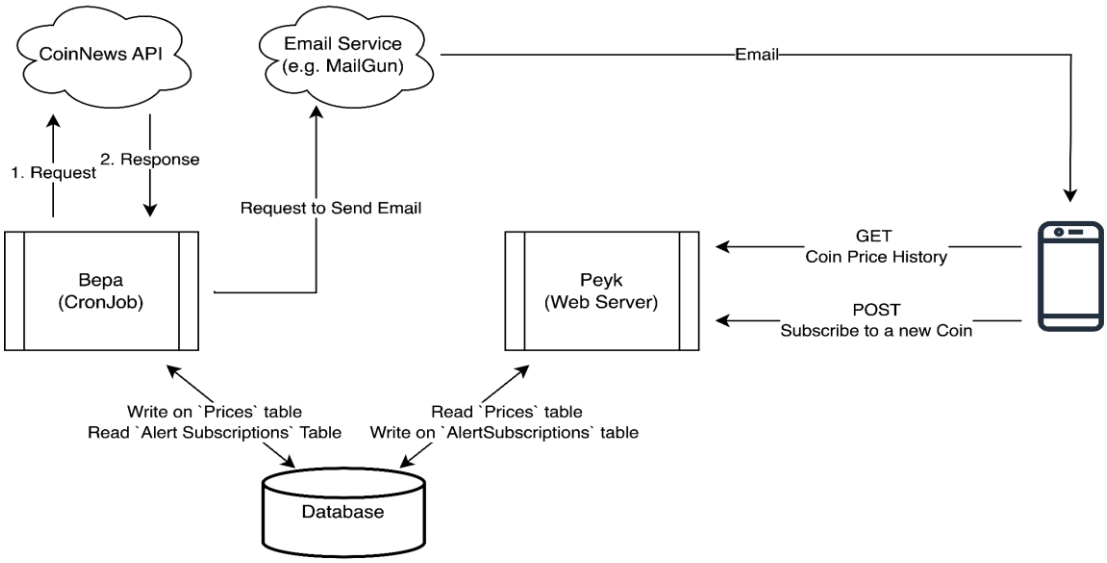
۷ تیر ماه

مقدمه

در این پروژه قصد داریم در قالب یک برنامه‌ی کاربردی از برخی مفاهیمی که در طول درس آموختیم، استفاده کنیم. در تمرین اول درس با روش بهره‌گیری از خدمات ابری آشنا شدیم. سپس در تمرین دوم آموختیم چگونه با تکنولوژی‌های داکر و کوبرنتیز، برنامه‌های خود را در قالب کانتینرها[۱] مستقر[۲] کنیم. در طول درس همچنین با روش‌های پردازش کلان‌داده، مانند آپاچی اسپارک[۳] و آپاچی هدوپ[۴] آشنا شدیم. وظیفه‌ی برنامه‌ی کاربردی‌ای که در این پروژه توسعه می‌دهیم، دیده‌بانی[۵] و هشدار[۶] قیمت رمزارزهاست.

گام اول: توسعه‌ی برنامه

برنامه‌ی مورد نظر از یک پایگاه داده و دو سرویس با نام‌های «بپا» و «پیک» تشکیل می‌شود. سرویس بپا، وظیفه‌ی دریافت تازه‌ترین قیمت رمزارزها و نوشتن این مقادیر در پایگاه داده را دارد. این سرویس، به صورت دوره‌ای در بازه‌های زمانی مشخص اجرا می‌شود. سرویس دوم، مقادیر پایگاه داده را خوانده و در قالب مناسب به کاربران ارسال می‌کند. همچنین، در جهت تعامل بهتر با برنامه، سمت ظاهر را نیز باید توسعه دهید و در انتخاب زبان و چارچوب برای توسعه سمت ظاهر[۷] آزاد هستید.
در ادامه هر بخش از برنامه را به طور کامل توضیح می‌دهیم.



نمودار کلی سرویس‌های برنامه

---

[۱] Container
[۲] Deploy
[۳] Apache Spark
[۴] Apache Hadoop
[۵] Monitoring
[۶] Alerting
[۷] Front End

پایگاه داده

پایگاه داده‌ی این برنامه از دو جدول تشکیل می‌شود.

جدول قیمت‌ها[۸] :هر سطر از این جدول، نشان‌دهنده‌ی قیمت یک ارز خاص (به دلار) در یک زمان خاص است. شمای پیشنهادی این جدول در ادامه آمده است.

Coin Name (String)	Timestamp (Time)	Price (Float)
BTC	۱۶۸۵۱۳۹۹۷۷.۱۴۳۷۶	۲۶,۷۵۵.۰۰

جدول اشتراک‌های هشدار[۹] :کاربران می‌توانند با دادن ایمیل خود از تغییرات ناگهانی قیمت یکی از رمزارزها باخبر شوند. هر سطر از جدول، نشان‌دهنده‌ی ایمیل فرد، ارز مورد نظر، و میزان تغییراتی است که باید به ازای رخداد آن، به کاربر پیام ارسال شود.

Email (String)	Coin Name (Foreign Key, String)	Difference Percentage (Int)
user@gmail.com	BTC	۱۰٪

بپا

این سرویس در هر بار اجرا، دو کار انجام می‌دهد.

- با هدف دریافت تازه‌ترین قیمت‌ها به سرویس coinnews درخواست ارسال می‌کند. پس از دریافت پاسخ[۱۰] نتایج را در جدول قیمت‌ها می‌نویسد. این API دارای سه اندپوینت برای دریافت لیست ارزهای فعال، دریافت قیمت هر ارز و تاریخچه تغییرات قیمت هر ارز می‌باشد. شما باید با استفاده از فایل‌های داده شده، این سرویس را بر روی کلاستر خود راه‌اندازی کنید و سپس از آن استفاده کنید. برای خواندن مستندات می‌تواند به این لینک مراجعه کنید. در این مستندات نحوه درخواست زدن به این سرویس آورده شده است. همچنین فایل‌های موردنیاز جهت اجرای آن بر روی کلاستر کوبرنتیز نیز قرار داده شده‌اند.

---

[۸] Prices

[۹] Alert Subscriptions

[۱۰] Response

- برای ارسال هشدار به کاربران، طی هر اجرا، درصد تغییرات هر یک از رمزارزها را نسبت به آخرین قیمت ثبت شده در جدول محاسبه می‌کند. سپس با پیمایش جدول اشتراک‌های هشدار، بررسی می‌کند که کدام یک از هشدارها فعال[۱۱] شده‌اند. برای هر یک از هشدارهای فعال شده، با ارسال یک ایمیل به کاربر، این اتفاق را به اطلاع کاربر می‌رساند.
  برای ارسال ایمیل، می‌توانید از سرویس ارسال ایمیل میل‌گان[۱۲] و یا هر سرویس دیگری استفاده کنید. در تمرین اول، با سرویس میل‌گان آشنا شده‌اید، برای توضیحات بیش‌تر در رابطه به این سرویس می‌توانید به دستور کار تمرین اول مراجعه کنید. (توجه داشته باشید بعد از ساخت اکانت در میل‌گان، احراز هویت شماره تلفن را انجام ندهید.)

پیک

این سرویس دو اندپوینت[۱۳] در اختیار کاربران می‌گذارد.

SubscribeCoin	GetPriceHistory

```python
"""Peyk Service"""

@api.route('/price')
def get_price_history():
  # extract coin name from request
  ...

  # read price history from `Price` table
  ...

  # send response: send price history to user
  ...


@api.route('/subscribe')
def subscribe_coin():
  # extract [email_addr, coin_name, price_change]
  #  from request
  ...

  # create an alert on `AlertSubscription` table
  ...

  # send response: notify user about the result
  ...
```

```python
"""Bepa Service"""

def handle_subscriptions():
  # calculate price change for each coin
  ...


  # iterate over `AlertSubscription` table
  #  & identify which subscriptions are triggered
  ...

  # send email for each triggered alert

  ...


def get_data():
  # request to coinnews api, get the data
  ...

  # write the latest prices on `Price` table
  ...


def main():
  get_data()
  handle_subscriptions()
```

---

[۱۱] Trigger

[۱۲] Mailgun

[۱۳] **جزئیات پیاده‌سازی این اندپوینت‌ها، اعم قرار گرفتن اطلاعات درخواست در سرآیند یا بدنه و فرمت اطلاعات (به صورت فرم، JSON، YAML و غیره) آزاد و به انتخاب شماست.**

هر کاربر با درخواستی حاوی ایمیل کاربر، نام رمزارز مورد نظر، و درصد تغییرات مورد نظر می‌تواند مشترک تغییرات یک رمزارز شود.	با ارسال درخواستی حاوی نام یک رمزارز، این اندپوینت تاریخچه‌ی قیمت یک ارز را برمی‌گرداند.

گام دوم: بسته‌بندی[۱۴] برنامه به کمک داکر

پس از اتمام توسعه‌ی دو سرویس بخش قبل، باید هر یک را با نوشتن یک داکرفایل در قالب یک ایمیج[۱۵] داکر بسازید.
در ساخت ایمیج تکنیک multistage build کمک بگیرید و در دو مرحله ایمیج خود را تولید کنید. وظیفه مرحله اول تنها build کردن پروژه شما و ساخت فایل قابل اجرا است تا نهایتا در مرحله دوم این فایل در یک کانتینر alpine اجرا شود. این تکنیک بیش‌تر برای کدهای زبان‌های کامپایلری کاربرد دارد، هر چند که برای زبان‌های مفسری نیز مزایایی دارد. فارغ از این که از چه نوع زبانی در توسعه‌ی سرویس‌ها می‌برید، از این تکنیک برای سخت ایمیج‌های خود استفاده کنید.

موارد زیر را در فایل گزارش نمایش دهید.

- ساخت ایمیج با استفاده از Dockerfile نوشته شده
- ارسال ایمیج ساخته شده بر روی داکرهاب و نتیجه آن
- در صورتی که پروژه خود را با استفاده از ایمیج ساخته شده بر روی سیستم شخصی خود تست کردید، تصاویر مربوطه را قرار دهید (این مرحله اجباری نیست ولی توصیه می‌شود.)
- محتویات داکرفایل

گام سوم: استقرار برنامه با کوبرنتیز

در این گام کانتینرهای خود را با کوبرنتیز مستقر می‌کنیم. هر یک از بخش‌های پروژه یعنی پایگاه داده، سرویس بپا و سرویس پیک برای اجرا در کوبرنتیز نیاز به برخی منابع دارند که در ادامه آورده شده است.

---

[۱۴] Package/Containerize
[۱۵] Image

موارد کلی

یک کانفیگ‌مپ[۱۶] حاوی اطلاعات پیکربندی پروژه شامل پورت سرور، آدرس پایگاه داده	یک سیکرت[۱۷] حاوی نام و رمز عبور پایگاه داده.

پایگاه داده

یک Persistent Volume و یک Persistent Volume Claim به منظور حفظ داده‌های پایگاه داده	یک دیپلویمنت[۱۸] که وظیفه‌ی اجرای آماده‌سازی و اجرای پایگاه داده را در قالب پادها دارد. نحوه ایجاد پایگاه داده به انتخاب شماست.	یک سرویس[۱۹] که ارتباط با پادهای دیپلویمنت را ممکن می‌سازد.

دیپلویمنت پایگاه داده، باید «نام و رمز عبور تعریف شده در سیکرت»، و «همچنین فضای حافظه‌ی تعریف شده در Persistent Volume Claim» را به کار ببرد.

سرویس بپا

این سرویس در قالب یک کرون‌جاب کوبرنتیز[۲۰] اجرا می‌شود. کرون‌جاب سرویس بپا، ایمیج ساخته شده در گام قبل را به صورت دوره‌ای هر ۳ دقیقه اجرا می‌کند.

سرویس پیک

یک دیپلویمنت که سرور توسعه داده شده را در قالب دو پاد اجرا می‌کند. این دیپلویمنت باید به محتوای کانفیگ‌مپ و سیکرت نوشته شده دسترسی داشته باشد تا پادهای آن از این اطلاعات برای دسترسی به پایگاه داده و... استفاده کنند.	یک سرویس برای برقراری ارتباط با پادهای سرویس پیک.

---

[۱۶] Configmap

[۱۷] Secret

[۱۸] Deployment

[۱۹] Service

[۲۰] CronJob

هدف کرون‌جاب‌ها در کوبرنتیز اجرای منظم و دوره‌ای یک کار به کمک یک کانتینر است. این کار می‌تواند بک‌آپ‌گیری دوره‌ای از پایگاه داده، تولید گزارش دوره‌ای و یا هر کار منظم دیگر باشد. توصیف کرون‌جاب در کوبرنتیز بسیار ساده بوده و در کم‌تر از ۱۵ خط ممکن است.

منابع کوبرنتیز ذکر شده در این مرحله به ترتیب وابستگی منطقی آمده‌اند. بنابراین برای اجرا، می‌توانید منابع را به ترتیب از (راست به چپ، بالا به پایین) ایجاد کنید. برای ساخت هر مورد، از دستور kubectl apply استفاده می‌کنیم.

موارد زیر را در گزارش خود بیاورید.

- با استفاده از دستور kubectl get صحت ایجاد منابع بر روی کلاستر را نمایش دهید.
- آدرس آی‌پی[۲۱] پادها و نحوه برقراری ارتباط میان آنها و سرویس ساخته شده.
- برای دیپلویمنت مربوط به پایگاه داده چه تعداد پاد ایجاد کردید؟ دلیل کار خود را توضیح دهید.
- به کمک یک پاد با ایمیج کرل[۲۲] و یا قابلیت port-forwarding کوبرنتیز، سیستم نهایی خود را بیازمایید. تصاویر صحت عملکرد درخواست‌های **subscribe_coin** و **get_price_history** و **ارسال ایمیل به کاربر** را بیاورید.

گام آخر: موارد امتیازی[۲۳]

بخش‌های امتیازی در ادامه آورده شده‌اند. برای هر بخش، به مواردی که باید در گزارش کار بیاورید، توجه کنید.

- ساخت HPA به منظور مقیاس‌سازی خودکار پادهای سرویس پیک.
- جایگزین کردن دیپلویمنت پایگاه داده با استیت‌فول‌ست‌ها. توجه داشته باشید این مورد نیازمند تغییراتی در کد پروژه شما است تا به درستی از مستر[۲۴] و اسلیو[۲۵] ساخته شده استفاده شود.
- پیاده‌سازی چارت هلم[۲۶]
- پیاده‌سازی داکر کامپوز[۲۷] جهت خودکارسازی ایجاد منابع و وابستگی‌های مورد نیاز پروژه و نهایتاً build و اجرای آن.

---

[۲۱] IP

[۲۲] cURL

[۲۳] **بخش‌های امتیازی، مباحثی هستند که برخلاف گام‌های تعریف شده، در درس و تمرین‌های قبلی به آن‌ها پرداخته نشده است، بنابراین لازم است برای انجام آن‌ها کمی در اینترنت جستجو کنید. بخش اول این گام، بسیار ساده است و توصیه می‌شود آن را انجام دهید.**

[۲۴] Master

[۲۵] Slave

[۲۶] Helm Chart

[۲۷] Docker compose

موارد زیر را در گزارش خود بیاورید.

- برای HPA
  - پارامترهای موجود جهت مقیاس کردن خودکار را بیان کنید.
  - شما کدام یک از این پارامترها را برای ایجاد HPA استفاده کردید؟ دلیل خود را شرح دهید.
- برای استیت‌فول‌ست
  - دلایل استفاده از استیت‌فول‌ست به جای دیپلویمنت.
  - نحوه‌ی استفاده از سرویس مستر و رپلیکاها
- برای چارت هلم
  - توضیح کوتاه ساختار چارت‌های هلم
  - محتویات و توضیح کوتاه پارامترهای تعریف شده در فایل values مربوط به چارت[۲۸].
- داکرکامپوز
  - توضیح مختصر فایل داکرکامپوز نوشته شده.

نکات بارگذاری و تحویل

- برای کیفیت پیاده‌سازی رابط کاربری، نمره‌ای تعلق نمی‌گیرد و تنها برای پیاده‌سازی سمت ظاهر فارغ از کیفیت و ظاهر آن، نمره منظور می‌شود.
- در تحویل آنلاین پروژه، تسلط شما روی فهم و ساخت منابع کوبرنتیز، نحوه‌ی ارتباط آن‌ها اهمیت دارد. همچنین باید بتوانید پس از تغییر فایل‌های پیکربندی[۲۹]، آن‌ها را اعمال کنید.
- ابهامات خود را در سایت و یا گروه تلگرامی درس مطرح کنید و ما در سریع‌ترین زمان ممکن به آن‌ها پاسخ خواهیم داد.
- پروژه‌ی خود را در قالب یک فایل زیپ با نام GID_FinalProject.zip (شامل گزارش کار و تمام فایل‌های پروژه اعم از کد، فایل‌های داکر و کوبرنتیز) بارگذاری کنید.

---

[۲۸] **(تعریف درست پارامترها بسیار مهم است.)**

[۲۹] Config