

رایانش ابری

## گزارش پروژه پایانی

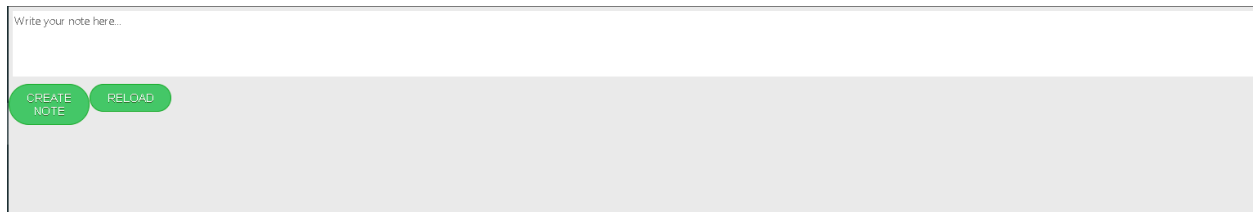
آرمینا صادقی-۹۷۳۱۰۹۳

دلارام رجائی-۹۷۳۱۰۸۴

گزارش قسمت های اصلی :

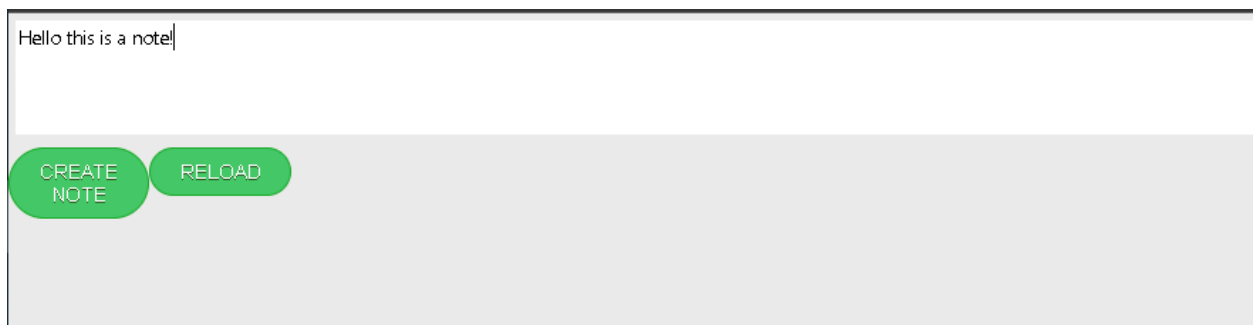
## ۱- گام اول

پروژه ی front-end ساخته شد. یک آن با زبان javascript و فرانت با ری اکت نوشته شده است. تصویر زیر نمونه ای از اجرای کد است:



## ۲- گام دوم:

۱. Build کردن ایمپج با استفاده از dockerfile ساخته شده
۲. ارسال ایمپج ساخته شده بر روی داکر هاب و نتیجه آن
۳. در صورتی که پروژه ی خود را با استفاده از ایمپج ساخته شده بر روی سیستم شخصی خود تست کردید، تصاویر مربوطه را قرار دهید.



### Note link ready

<http://localhost:3000/?note=5ecfec40-f999-11ec-aeef-1b528a9c063d>

*The note will self-destruct after reading it.*

-11ec-aefe-1b528a9c063d

Movies

localhost:3000 says

Read and destroy?

You're about to read and destroy the note with id5ecfec40-f999-11ec-aefe-1b528a9c063d

OK

Cancel

Hello this is a note!

The message will be deleted on refresh

RELOAD


## deliiled / cc\_fp

### Description





This repository does not have a description 

 Last pushed: 2 hours ago

### Tags and Scans

 VULNERABILITY SCANNING - **DISABLED**  
[Enable](#)

This repository contains 2 tag(s).

TAG	OS	PULLED	PUSHED
 redis		---	2 hours ago
 backend.1.0		---	4 hours ago

[See all](#)

۳- گام سوم:

۱. با استفاده از دستور `kubectl get` صحت ایجاد منابع بر روی کلاستر را نمایش دهید

Pod های ساخته شده را میتوان با آدرس Ip های آنها مشاهده کرد.

```
C:\Users\Delaram\Desktop\28_FinalProject\kubernetes>kubectl get pods -o wide
```

NAME	MINIATED NODE	READINESS GATES	READY	STATUS	RESTARTS	AGE	IP	NODE	NC
cc	one>	<none>	0/1	CrashLoopBackOff	158 (59s ago)	77d	172.17.0.2	minikube	<n
hello-minikube	one>	<none>	1/1	Running	2 (3h24m ago)	77d	172.17.0.7	minikube	<n
info-redis-deploy	one>	<none>	1/1	Running	0	116m	172.17.0.10	minikube	<n
info-server-deploy	one>	<none>	1/1	Running	2 (3h24m ago)	77d	172.17.0.3	minikube	<n
info-server-deploy	one>	<none>	1/1	Running	2 (3h24m ago)	77d	172.17.0.6	minikube	<n
info-server-deploy	one>	<none>	0/1	CreateContainerConfigError	0	140m	172.17.0.13	minikube	<n
info-server-deploy-backend	one>	<none>	0/1	CreateContainerConfigError	0	3h16m	172.17.0.8	minikube	<n
info-server-deploy-backend	one>	<none>	0/1	CreateContainerConfigError	0	3h16m	172.17.0.9	minikube	<n
info-server-deploy-backend	one>	<none>	0/1	CreateContainerConfigError	0	148m	172.17.0.12	minikube	<n
note-deploy	one>	<none>	1/1	Running	0	96s	172.17.0.11	minikube	<n
note-deploy	one>	<none>	1/1	Running	0	96s	172.17.0.15	minikube	<n
note-redis-deploy	one>	<none>	1/1	Running	0	25m	172.17.0.14	minikube	<n
sh	one>	<none>	1/1	Running	1 (3h24m ago)	65d	172.17.0.5	minikube	<n

در زیر سرویس هایی که بالا هستند را میتوان دید.

```
C:\Users\Delaram\Desktop\28_FinalProject\kubernetes>kubectl get svc
```

NAME	TYPE	CLUSTER-IP	EXTERNAL-IP	PORT(S)	AGE
hello-minikube	NodePort	10.103.132.144	<none>	8080:31050/TCP	77d
kubernetes	ClusterIP	10.96.0.1	<none>	443/TCP	77d
note-server-server	ClusterIP	10.105.166.223	<none>	6379/TCP	52m
note-service	LoadBalancer	10.110.177.59	<pending>	3001:31538/TCP	103m

```
C:\Users\Delaram\Desktop\28_FinalProject\kubernetes>
```

۲. آدرس IP پاد ها و نحوه برقراری ارتباط میان آن ها و سرویس ساخته شده

در شکل قبلی آدرس IP های هر کدام نشان داده شد. از آنجایی که دیتابیس روی یک pod جدا قرار دارد برای اینکه بتواند با سایر pod ها ارتباط بگیرد چون نوع سرویس آن از اون clusterIP است باید درون یک شبکه قرار داشته باشند، به همین دلیل باید نام label استفاده شده در سرویس مربوط به pod بک و سرور با pod دیتابیس یکی باشد.

۳. برای دیپلویمنت مربوط به دیتابیس چه تعداد پاد ایجاد کردید؟ دلیل کار خود را توضیح دهید

دو pod برای سرور چون دو تا replica میخواستیم و یک pod برای دیتابیس، که هر دو سرور به آن دسترسی دارند. این سرور ها با loadBalancer کنترل میشوند. میتوانستیم front را نیز به صورت جدا بالا بیاوریم و یک pod نیز به آن اختصاص دهیم.

برای دیتابیس باید حتما یک pod باشد چون اطلاعات باید در یک جا ذخیره شوند و که هر دو سرور به آن دسترسی داشته باشند، در صورتی که دو pod باشد ممکن است سرور در صورت درخواست زدن به یک دیتابیس به اطلاعاتی که میخواهد دسترسی پیدا نکند.

گزارش بخش های امتیازی:

#### ۱- ساختن یک کامپوننت HPA

۱. پارامتر های موجود جهت مقیاس کردن خودکار را بیان کنید.

باید معیاری که به کمک آن بار پادها و نیار به scale up/down مشخص میشود داریم. در api نسخه یک، تنها معیار بهروری cpu برای پادها وجود داشت اما در نسخه های بتای فعلی، میتوان از مهیار های متفاوتی برای این کار استفاده کرد. معیارها عبارت اند از:

Container resource, external, pods, resource. در اینجا تنها معیار resource برای مقیاس کردن استفاده میشود.

۲. شما کدام یک از این پارامتر ها را برای ایجاد HPA استفاده کردید؟ دلیل خود را شرح دهید.

۳. دستور و یا توصیف مورد استفاده برای ساخت HPA

۲- اجرای دیتا بیس با استفاده از stateful set

۱. دلیل استفاده از stateful set به جای deployment

۲. توصیف مورد استفاده برای ساخت stateful set

پاسخ سوال ۱ و ۲: در واقع statefulset یک کامپوننت در کوبرنتیز میباشد که برای اپلیکیشن های stateful استفاده میشود، اپلیکیشن های stateful اپلیکیشن هایی هستند که دیتا را در خود ذخیره میکنند تا بتوانند حالت خود را ذخیره کنند و بتوانند حالت خود را دنبال کنند در مقابل اپلیکیشن های stateless هیچ حالتی از خود ذخیره نمیکند و هر ریکوئست به صورت کاملاً isolated هندل میشود و با استفاده از اطلاعات خود درخواست فعلی هندل میشود. دیپولی کردن این اپلیکیشن ها نیز متفاوت است. در deployment به صورت رندوم تولید میشوند و در انتهای آنها random hashes وجود دارد. در statefulset نمیتوان به صورت همزمان و به هر ترتیب دلخواهی پادها را create یا delete کرد و نمیتوان پادها را به صورت رندوم آدرسدهی کرد و این امر به این دلیل است که پادها identical نیستند.

۳. نحوه ی استفاده از سرویس مستر و رپلیکا ها

در این حالت در واقع پاد مستر یک نقش را دارد و پادهای دیگر نقش slave را ایفا میکنند و مستر میتواند بخواند و بنویسد اما دیگر پادها فقط اجازه ی read را دارند. دز غیز اینصورت ناسازگاری به وجود می آید.

۳- پیاده سازی helm chart

۱. توضیح مختصر ساختار helm chart

پکیجی است که در آن تمام توصیف های لازم برای بالا آوردن یک برنامه منابع مورد نیاز آن در کوبرنتیز وجود دارد و با نصب چارت این توصیف ها به کوبرنتیز ارسال و ایجاد میشود.

۲. محتویات و توضیح مختصر پارامتر های تعریف شده در فایل values مربوط به چارت(تعریف درست پارامتر ها بسیار مهم است)

۴- محتویات و توضیح مختصر docker compose پیاده سازی شده

```

version: "3.8"

services:
  frontend:
    build:
      context: ./frontend
      dockerfile: ./Dockerfile
    ports:
      - 3000:80
    networks:
      - CC_Network

  backend:
    build:
      context: ./backend
      dockerfile: ./Dockerfile
    ports:
      - 3001:3001
    env_file:
      - ./backend/.env
    depends_on:
      - redis-server
    networks:
      - CC_Network

  redis-server:
    build:
      context: ./redis
      dockerfile: ./Dockerfile
    networks:
      - CC_Network

networks:
  CC_Network:

```

برای سه قسمت باید درست شود، فرانت، بک و دیتابیس که برای هر کدام داکر فابل مربوطه را به آن مدهیم و پورت ها را مپ میکنیم.

```

depends_on:
  - redis-server

```

این قسمت نیز به این معنی است که باید منتظر بماند تا ابتدا حتما دیتابیس بالا بیاید.