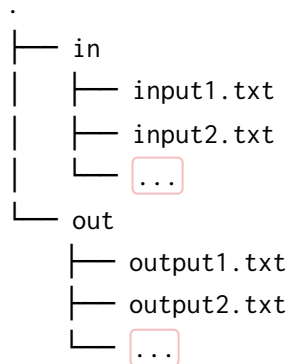


تست جنریتور

سلیب که از طراحی سوالات ورودی و خروجی کوئرا خسته شده بود، تصمیم گرفت تا با پیاده سازی اسکریپتی این فرآیند طاقت فرسا را برای خود سهل کند! ابتدا با ساختار فایل داوری ورودی خروجی آشنا شویم، ساختار فایلی که برای کوئرا مورد پذیرش است، به فرم زیر است:

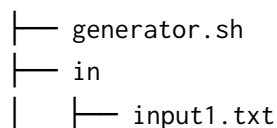


کوئرا برای هر ورودی مثل i ، محتویات فایل `in/input{i}.txt` را به کد ارسالی کاربر می دهد و خروجی را با محتویات فایل `out/output{i}.txt` مقایسه می کند و در صورت برابر بودن، نمره را به پاسخ ارسالی کاربر می دهد.

تا پیش از این سلیب به طور کامل دستی محتوای تمامی `output{i}.txt` ها را پر می کرد، حال از شما می خواهد تا اسکریپتی بنویسید که این کار را برای او انجام دهد. او می خواهد تا اسکریپت شما با دریافت تعداد ورودی، ورودی ها را از فایل مربوطه بخواند و به عنوان ورودی (`stdin`) به برنامه `main.py` دهد و خروجی برنامه را در فایل خروجی مربوط به تست کیس بنویسد. برای درک بهتر مثالی را پیش می بریم.

مثال

برای مثالی با 3 تست کیس، اسکریپت `generator.sh` شما ابتدا در ساختار زیر قرار دارد:




```

|   |— input2.txt
|   |— input3.txt
|   └─ main.py

```

سلیب می‌خواهد که وقتی اسکریپت شما را مانند زیر اجرا کند:

 bash

```
1 | ./generator.sh 3
```

- خروجی مربوط به ورودی in/input1.txt درون فایل out/output1.txt ذخیره شود.
- خروجی مربوط به ورودی in/input2.txt درون فایل out/output2.txt ذخیره شود.
- خروجی مربوط به ورودی in/input3.txt درون فایل out/output3.txt ذخیره شود.

و در نهایت به ساختار زیر برسیم:

```

.
├── generator.sh
├── in
│   ├── input1.txt
│   ├── input2.txt
│   └── input3.txt
├── main.py
└── out
    ├── output1.txt
    ├── output2.txt
    └── output3.txt

```

پروژه اولیه

برای دریافت پروژه اولیه [این لینک](#) را دانلود کنید. درون لینک ساختار فایلی زیر را مشاهده می‌کنید:

```

Test-Generator
├── generator.sh
└── in

```

```


|   |— input1.txt
|   |— input2.txt
|   |— input3.txt
|   └─ main.py

```

راه حل خود را درون فایل `generator.sh` پیاده سازی کنید.

توجه کنید

- فایل `main.py` صرفاً برای تست روی سیستم شخصی تان در اختیار شما قرار گرفته و `main.py` روی سیستم داوری خروجی های متفاوتی تولید می کند.
- برای اجرای فایل `main.py` باید از دستور زیر استفاده کنید:

 `bash`

```
1 | python3 main.py
```

بنابراین طبق دستور بالا باید نسخه ۳ پایتون را در سیستم شخصی تان نصب داشته باشید.

- در هر تست در سیستم داوری، تعدادی تست کیس در دایرکتوری `in/` قرار دارند و تضمین می شود که هیچ تست کیسی خالی نمی باشد.
- دسترسی اجرا به فایل ارسالی شما به طور خودکار در سیستم داوری داده می شود.


نحوه ارسال

برای ارسال جواب، دستورات خواسته شده را در فایلی با نام `generator.sh` بنویسید و سپس آن را انتخاب و ارسال کنید.

مدیریت کارمندان

سلیب که به تازگی شرکت خود را تاسیس کرده اکنون با چالش جدیدی مواجه شده، مدیریت کارمندان! در حال حاضر او نیاز دارد تا بتواند هرگاه که اراده کرد به هر کارمند خود به عنوان تشویق پاداش بدهد یا اطلاعات تمامی کارمندان ساکن شهر خاصی را ببیند.

سلیب اطلاعات تمامی کارمندان خود را در یک فایل با نام `employee.csv` ذخیره کرده که محتوای آن به فرمت زیر است:


 `employee.csv`

```
1 | 10012, Tehran, Seyed Ali Babaei, 09121212121, 4000, Narmak-Kooche-Aval
2 | 20221, Tehran, Mostafa Karimi, 09131313131, 3900, Kerman-Kooche-Aval
3 | 40521, Semnan, Amin Anvari, 09123456789, 3800, Pirooz-Kooche-Aval
4 | 12140, ALborz, Nima Heydari Nasab, 09383838383, 4100, Fardis-Kooche-Aval
```

در ستون اول آیدی کارمند، در ستون دوم شهر کارمند، در ستون سوم نام کارمند، در ستون چهارم شماره تلفن کارمند، در ستون پنجم حقوق کارمند و در ستون ششم آدرس کارمند مشاهده می شود.

او که چیزی از ترمینال لینوکس و اسکریپت نویسی نمی داند از شما کمک خواسته تا برای او دو کار زیر را انجام دهید:

- دستوری با نام `bonus` تعریف کنید تا با دریافت آیدی کارمند، به میزان ۵ درصد حقوق او به او پاداش دهد. یعنی با اجرای دستور زیر:

 `bash`

```
1 | ./manager.sh bonus 40521
```

خروجی زیر مشاهده شود:

 `bash`

```
1 | Amin Anvari will get $190 bonus
```

- دستوری با نام city تعریف کنید تا با دریافت شهر مورد نظر سلیب، نام و شماره تلفن تمام کارمندان ساکن آن شهر او را خروجی بدهد. یعنی با اجرای دستور زیر:

```
bash
```

```
1 | ./manager.sh city ALborz
```

خروجی زیر مشاهده شود:

```
bash
```

```
1 | Customer Name: Nima Heydari Nasab
2 | Mobile No: 09383838383
```

- در صورتی که پارامتر اول ورودی هیچکدام از دو حالت بالا نبود، اسکریپت باید عبارت command not found را چاپ کند.

مثال

برای مثال اگر اسکریپت شما را مانند زیر اجرا کنیم:

```
bash
```

```
1 | ./manager.sh bonus 10012
```

باید خروجی زیر را دهد:

```
bash
```

```
1 | Seyed Ali Babaei will get $200 bonus
```

یا اگر اسکریپت شما را مانند زیر اجرا کنیم:

```
bash
```

```
1 | ./manager.sh city Tehran
```

باید خروجی زیر را دهد:

```
bash
```

```
1 | Customer Name: Seyed Ali Babaei
2 | Mobile No: 09121212121
3 | Customer Name: Mostafa Karimi
4 | Mobile No: 09131313131
```

پروژه اولیه

برای دریافت پروژه اولیه این لینک را دانلود کنید. درون لینک ساختار فایلی زیر را مشاهده می کنید:

Employee-Management

```
├─ employee.csv
└─ manager.sh
```

راه حل خود را درون فایل `manager.sh` پیاده سازی کنید.

توجه کنید

- به تمامی فاصله های خالی بین حروف خروجی دقت کنید.
- اطلاعات مورد نیاز شما در سیستم داوری درون فایل `employee.csv` در کنار اسکریپت شما قرار دارد.
- دسترسی اجرا به فایل ارسالی شما به طور خودکار در سیستم داوری داده می شود.
- تضمین می شود ۵ درصد حقوق کارمندان همواره عددی صحیح است.

نحوه ارسال

برای ارسال جواب دستورات خواسته شده را درون فایلی با نام `manager.sh` وارد کنید و آن را انتخاب و سپس ارسال کنید.

ردیس لیمیتر

سلیب که به تازگی با داکر آشنا شده قصد دارد تا یک `docker-compose` ابتدایی را طراحی کند. در ادامه به بیان جزئیات کانترینر مورد نظر او می‌پردازیم، به سلیب کمک کنید محدودیت‌های زیر را در منابع یک `container` داکر `Redis` اضافه کند:

- او می‌خواهد تا `Redis` روی پورت 6379 اجرا شده باشد.
- هرگاه به هردلیلی سرویس `Redis` از کار افتاد، بلافاصله مجدداً اجرا شود.
- دارای `Memory Limit`ی برابر با 30M باشد.
- دارای `Memory Reservation`ی برابر با 30M باشد.

برای دانلود پروژه‌ی اولیه روی [این لینک](#) کلیک کنید.

نکات تکمیلی

۱. شما اجازه‌ی `build` کردن یک `Image` جدید نخواهید داشت و باید از `Image`های استاندارد استفاده کنید. به‌طور خاص، برای `Redis` می‌توانید از `Image` با آدرس زیر استفاده کنید:

```
registry.gitlab.com/qio/standard/redis:latest
```

۲. نام `container` باید `redis` باشد.

۳. سیستم دآوری کوئرا به‌صورت خودکار فایل `docker-compose.yml` را با کامند زیر اجرا می‌کند. شما نیازی به کد یا اسکریپتی برای اجرای این کار ندارید.

```
docker-compose up --no-build -d
```

۴. توجه داشته باشید که ورژن `docker-compose.yml` باید 2 باشد.

نحوه ارسال پاسخ

شما تنها مجاز به ارسال فایل `docker-compose.yml` هستید.

 `docker-compose.yml`

```
1 | version: "2"
2 |
3 | # Do not forget that the only available
4 | # redis image is accessible with the following url:
5 | # registry.gitlab.com/qio/standard/redis:latest
```

کشورداری

در تیم زیرساخت دیجی کالا، میکروسرویس‌های مختلفی در حال پدید آمدن هستند. برخی از این میکروسرویس‌ها قرار است تنها برای کاربران برخی از کشورها در دسترس باشند. سالار که وظیفه‌ی مدیریت درخواست‌های ورودی از کشورهای مختلف را برعهده دارد، لیستی از بازه‌های آی‌پی‌های کشورهای مدنظرش جمع‌آوری کرده است. او اکنون از شما می‌خواهد اسکریپتی برای او بنویسید که با استفاده از آن بتوان درخواست‌های ورودی از کشورهای مختلف را مسدود یا آزاد کرد.

جزئیات پروژه

پروژه‌ی اولیه را از [این لینک](#) دانلود کنید.

ساختار فایل‌های پروژه به‌صورت زیر است:

```
country-manager
├── ip_lists
│   ├── AG.txt
│   ├── CC.txt
│   └── IR.txt
└── country_manager.sh
```

اسکریپت `country_manager.sh` را مطابق توضیحات زیر پیاده‌سازی کنید:

- نحوه‌ی اجرای اسکریپت به‌صورت `./country_manager.sh command ip_list_filepath` است. مثال:

```
bash
```

```
1 | ./country_manager.sh block ./ip_lists/CC.txt
```

- اگر تعداد آرگومان‌های ورودی اسکریپت کمتر از ۲ تا (یکی برای `command` و یکی برای `ip_list_filepath`) بود، اسکریپت باید عبارت `not enough arguments` را چاپ کند و

متوقف شود.

- اگر مقدار `command` برابر با `block` یا `unblock` نبود، اسکریپت باید عبارت `invalid command` را چاپ کند و متوقف شود.
- اگر فایلی که مسیر آن به عنوان آرگومان دوم وارد می شود وجود نداشت، اسکریپت باید عبارت `ip list file not found` را چاپ کند و متوقف شود.
- اگر مقدار `command` برابر با `block` باشد، اسکریپت باید لیست بازه های آی پی موجود در فایل را دریافت کند و درخواست های ورودی مربوط به این آی پی ها را به ازای همه ی پروتکل ها و پورت ها `DROP` کند. اگر `rule` مربوط به `ACCEPT` کردن این درخواست ها از قبل موجود بود، اسکریپت باید آن ها را حذف کند.
- اگر مقدار `command` برابر با `unblock` باشد، اسکریپت باید لیست بازه های آی پی موجود در فایل را دریافت کند و درخواست های ورودی مربوط به این آی پی ها را به ازای همه ی پروتکل ها و پورت ها `ACCEPT` کند. اگر `rule` مربوط به `DROP` کردن این درخواست ها از قبل موجود بود، اسکریپت باید آن ها را حذف کند.

نکات

- عملیات مسدودسازی و آزادسازی درخواست ها باید با استفاده از `iptables` انجام شود.
- داوری این سؤال کمی بیشتر از سایر سؤالات طول می کشد.
- بازه های آی پی طبق نگارش `CIDR` هستند.
- تضمین می شود که هیچ `rule` ای خارج از اسکریپت اضافه نمی شود.

آنچه باید آپلود کنید

پس از پیاده سازی موارد خواسته شده، فایل `country_manager.sh` را آپلود کنید.

سرویس گیت

روزی روزگاری، محمد از دست گیت لب به کفر آمد و تصمیم گرفت تا سرویسی برای شرکت در همین زمینه راه اندازی کند ولی به دلیل خستگی زیاد از شما خواسته تا به او در راستای این هدف کمک کنید.

در این تمرین، ما از شما می‌خواهیم یک سرویس gitea را تنظیم و راه اندازی کنید. برای دانلود پروژه اولیه روی [این لینک](#) کلیک کنید.

جزئیات پروژه

- سرویس gitea شما باید روی پورت 3000 اجرا شده باشد.
- دیتای سرویس gitea نباید به هیچ عنوان از بین برود (یعنی اگر به هر دلیلی سرویس ها متوقف شدند، نباید هیچ دیتایی از دست رفته باشد).
- اگر سیستم به هر دلیلی با مشکلی مواجه شد، باید سرویس gitea بلافاصله مجددا اجرا شود.
- پورت ssh سرویس برای محمد اهمیتی ندارد، برای اطمینان می‌توانید از پورت 222 استفاده کنید.
- باید id کاربر و id گروهی که به دستورات gitea دسترسی دارند برابر با 1000 باشد.
- نام کانتینری که سرویس gitea را اجرا می‌کند باید برابر با gitea باشد.

نکات تکمیلی

۱. شما تمامی تغییرات خود را درون فایل docker-compose.yml اعمال کنید. شما تنها مجاز به ارسال این فایل خواهید بود و با دستور زیر فایل ارسالی شما درون سیستم داوری اجرا خواهد شد:

```
docker-compose up --no-build -d
```

پس از اجرای دستور بالا باید تمامی سرویس‌های مورد نظر شما تنظیم و راه اندازی و آماده تست کردن باشند.

۲. شما تنها مجاز به استفاده از ایمیج‌های موجود در این لینک هستید و می‌توانید هر تعداد سرویس که نیاز دارید تنظیم و راه‌اندازی کنید تا به خواسته مسئله برسید. توجه داشته باشید که شما مجاز به استفاده از ایمیج‌های دیگر نیستید و فقط می‌توانید از مجموعه ایمیج‌های موجود در همین ایمیج‌ریستری استفاده کنید.

۳. شما اجازه‌ی *build* کردن یک *Image* جدید نخواهید داشت و باید از *Image*های استاندارد کوئرا استفاده کنید. به‌طور خاص، برای *gitea* می‌توانید از *Image* با آدرس زیر استفاده کنید:

```
registry.gitlab.com/qio/standard/gitea:1.16.8
```

۴. در صورت نیاز به دیتابیس، از ایمیج‌های زیر نیز می‌توانید استفاده کنید:

```
registry.gitlab.com/qio/standard/mysql:8.0
```

```
registry.gitlab.com/qio/standard/postgres:14.1
```

```
registry.gitlab.com/qio/standard/postgres:14.1-alpine
```

ارسال فایل‌های اضافی

توجه کنید که هر فایل و محتوای دیگری به جز `docker-compose.yml` که در فایل ارسالی شما قرار داشته باشد حذف خواهد شد.

در صورتی که نیاز دارید که فایلی علاوه بر `docker-compose.yml` به سیستم دآوری کوئرا ارسال کرده و در راه‌حل خود از آن‌ها استفاده کنید، می‌توانید از پوشه‌ی `data` استفاده کنید. برای این کار کافیهست که پوشه‌ای به نام `data` در کنار `docker-compose.yml` ایجاد کرده و برای سیستم دآوری کوئرا ارسال نمایید.

```
services:
  sample_container:
    volumes:
      - ./data/extrafile:/extrafile
```

فراموش نکنید که در این حالت، فایل `extrafile` که `volume` شده است را درون پوشه‌ی `data` قرار داده و آن را برای سیستم داوری ارسال نمایید. برای مثال، پوشه‌ی ارسالی شما برای داوری کوئرا به شکل زیر می‌تواند باشد:

```
.
├── data
│   └── extrafile
└── docker-compose.yml
```

توجه: سیستم داوری کوئرا به صورت خودکار فایل `docker-compose.yml` را با کامند `up` اجرا می‌کند. شما نیازی به کد یا اسکریپتی برای اجرای این کار ندارید.

توجه: استفاده کردن از پوشه‌ی `data` ، کاملاً به خواست شما بوده و ممکن است این سؤال بدون نیاز به استفاده از پوشه‌ی `data` حل شود.

نحوه ارسال پاسخ

شما تنها مجاز به تغییر در فایل `docker-compose.yml` هستید و در صورت نیاز در کنار آن می‌توانید پوشه‌ای با نام `data` نیز قرار داده و ارسال نمایید. تمامی فایل‌ها یا پوشه‌های دیگر حذف خواهند شد!

سامانه ثبت پیشنهادات و انتقادات

این سؤال تنها با زبان‌های *Go* ، *Python* ، *PHP* و *Node.js* (*JS*) قابل حل است.

دیجی‌کالا قصد دارد برای بخش تحویل محصولات خود یک سامانه‌ی ساده‌ی ثبت پیشنهادات و انتقادات راه‌اندازی کند. از شما می‌خواهیم یک *API* برای این سامانه طراحی کنید.

جزئیات پروژه

پروژه‌ی اولیه را از این لینک دانلود کنید.

در این سؤال، یک *REST API* شامل *endpoint* های زیر باید پیاده‌سازی شود:

| عنوان | آدرس |
|----------------------------------|-------------------|
| بودن سرویس <i>up</i> بررسی | GET / |
| ثبت‌نام | POST /signup |
| ورود به حساب کاربری | POST /login |
| درج پیشنهاد یا انتقاد | POST /suggestions |
| دریافت لیست پیشنهادات و انتقادات | GET /suggestions |

در این *API* هر کاربر باید یک توکن داشته باشد. این توکن برای هر کاربر ثابت است.

endpoint های موردنیاز

در همه‌ی *endpoint* ها، پاسخ باید به‌صورت *JSON* باشد.

اطلاعات ورودی به‌صورت `application/x-www-form-urlencoded` به *endpoint* ها ارسال می‌شوند.

بررسی *up* بودن سرویس

پاسخ این *endpoint* باید به صورت زیر باشد:

- کد وضعیت: 200
- بدنه: {"ok":true}

ثبت نام

دو پارامتر `username` و `password` باید به این *endpoint* ارسال شوند. در صورتی که حداقل یکی از این پارامترها ارسال نشده باشد یا برابر با رشته‌ی خالی باشد، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 400
- بدنه: {"ok":false,"error":"no username or password provided"}

اگر کاربری با نام کاربری وارد شده از قبل موجود باشد، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 400
- بدنه: {"ok":false,"error":"user already exists"}

در غیر این صورت، کاربر باید ساخته شود، یک توکن یکتا برایش تولید شود و پاسخ به صورت زیر باشد:

- کد وضعیت: 201
- بدنه: {"ok":true,"token":"USER_TOKEN"}

ورود به حساب کاربری

دو پارامتر `username` و `password` باید به این *endpoint* ارسال شوند. در صورتی که حداقل یکی از این پارامترها ارسال نشده باشد یا برابر با رشته‌ی خالی باشد، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 400
- بدنه: {"ok":false,"error":"no username or password provided"}

اگر نام کاربری یا رمز عبور نادرست باشد، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 400

• بدنه: `{"ok":false,"error":"invalid username or password"}`

در غیر این صورت، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 200

• بدنه: `{"ok":true,"token":"USER_TOKEN"}`

درج پیشنهاد یا انتقاد

این *endpoint* نیازمند *authentication* است. در ریکوئست ارسالی مقدار هدر *Authorization* باید برابر با توکن کاربر باشد (بدون **Bearer** یا موارد مشابه).

پارامتر *text* (متن پیشنهاد یا انتقاد) باید به این *endpoint* ارسال شود. در صورتی که این پارامتر ارسال نشده باشد یا برابر با رشته‌ی خالی باشد، پاسخ باید به صورت زیر باشد:

- کد وضعیت: 400

• بدنه: `{"ok":false,"error":"no text provided"}`

در غیر این صورت، پیشنهاد یا انتقاد باید ثبت شود و پاسخ به صورت زیر باشد:

- کد وضعیت: 201

• بدنه: `{"ok":true}`

دریافت لیست پیشنهادات و انتقادات

این *endpoint* باید لیست پیشنهادات و انتقادات ثبت شده در سامانه را در قالب یک لیست برگرداند. این لیست شامل پیشنهادات و انتقادات ثبت شده توسط همه‌ی کاربران است.

- کد وضعیت: 200

• مثالی از پاسخ:

```
1 | [
  {
```

```

2      "user": "username1",
3      "text": "sample suggestion 1"
4  },
5  {
6      "user": "username2",
7      "text": "sample suggestion 2"
8  }
9  ]
10

```

نکات تکمیلی

▼ نصب نیازمندی‌ها و اجرا

برای حل این سؤال می‌توانید از هر زبان و هر تکنولوژی‌ای که می‌خواهید استفاده کنید. به‌صورتی که در یک پوشه به نام `api` کد برنامه را نوشته و در فایل به نام `runner.sh` که توسط `sh` اجرا می‌شود، باید برنامه‌ی خود را اجرا کنید. توجه کنید که حتماً باید `Dockerfile` مربوط به پروژه‌ی خود را برای ما ارسال کنید.

در پروژه‌ی اولیه، ۴ داکرفایل برای `php`، `python`، `golang` و `node` قرار دادیم که می‌توانید از آن‌ها مستقیماً استفاده کنید. در صورتی که از یکی از این زبان‌ها برای حل سؤال استفاده می‌کنید، کافیست که `Dockerfile` مربوط به آن را در پوشه‌ی `api` کپی کنید و طبق توضیحات داده شده، سؤال را حل کنید. برای نصب نیازمندی‌های پایتون از `requirements.txt`، برای پی‌اچ‌پی از `composer.json`، برای گولنگ از `go.mod` و برای نودجی‌اس از `package.json` استفاده کنید.

در صورتی که زبان مورد استفاده‌ی شما، چیزی به جز این ۴ مورد است، باید خودتان داکرفایلی در پوشه‌ی `api` به‌شکلی بنویسید که بتواند نیازمندی‌های پروژه‌ی شما را نصب کرده و برنامه‌ی شما را مانند داکرفایل‌های موجود اجرا کند.

- نیازی به *persistent* بودن داده‌ها نیست!
- سیستم داوری `docker-compose.yml` زیر را خارج از فولدر `api` پاسخ شما قرار می‌دهد و با دستور `docker-compose up --build` آن را اجرا می‌کند.



docker-compose.yml

```

1 | version: "3"
2 |
3 | services:
4 |   api:
5 |     build: "./api"
6 |     container_name: "api"
7 |     ports:
8 |       - "80:80"

```

- شما مجاز به تغییر یا ارسال docker-compose.yml دلخواه نیستید.
- سرویس شما باید روی پورت 80 آدرس localhost قابل دسترسی باشد.
- توصیه می‌کنیم در runner.sh خود API تان را روی 0.0.0.0:80 اجرا کنید.

▼ تغییر Dockerfile

امکان تغییر فایل Dockerfile وجود ندارد، اما در اسکریپت runner.sh می‌توانید هر دستوری را اجرا کنید.

نحوه ارسال پاسخ

شما می‌توانید تمامی محتوای موجود در پوشه‌ی api را تغییر دهید و هر فایلی که می‌خواهید اضافه یا کم کنید.

```

1 | api
2 | └─ api.py # or main.go somefile.js anyfile.php name.any ...
3 | └─ Dockerfile
4 | └─ requirements.txt # or go.mod package.json composer.json
5 | └─ runner.sh

```

توجه کنید که نام فایل کد شما برای سیستم داوری اهمیتی ندارد و این خود شما هستید که در runner.sh از نام آن برای اجرای پروژه استفاده می‌کنید.

در نهایت این پوشه را *zip* کرده و ارسال کنید. توجه کنید که پس از *extract* کردن فایل *zip* شما، باید پوشه‌ی *api* را ببینیم که درون آن *Dockerfile* وجود دارد.