

# Grafana/Loki

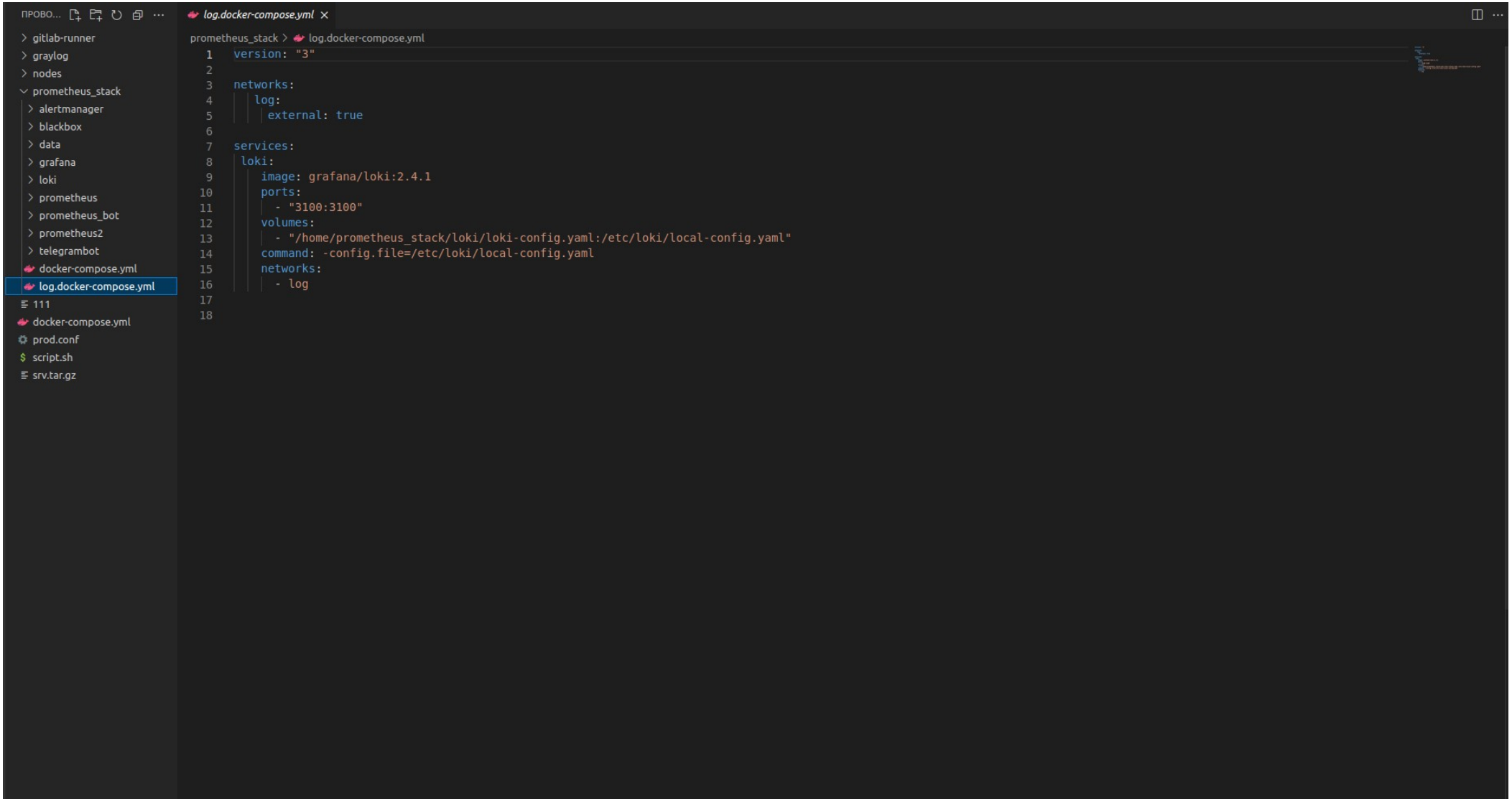
Дано:

- На сервере мониторинга запущенна Grafana

Задание:

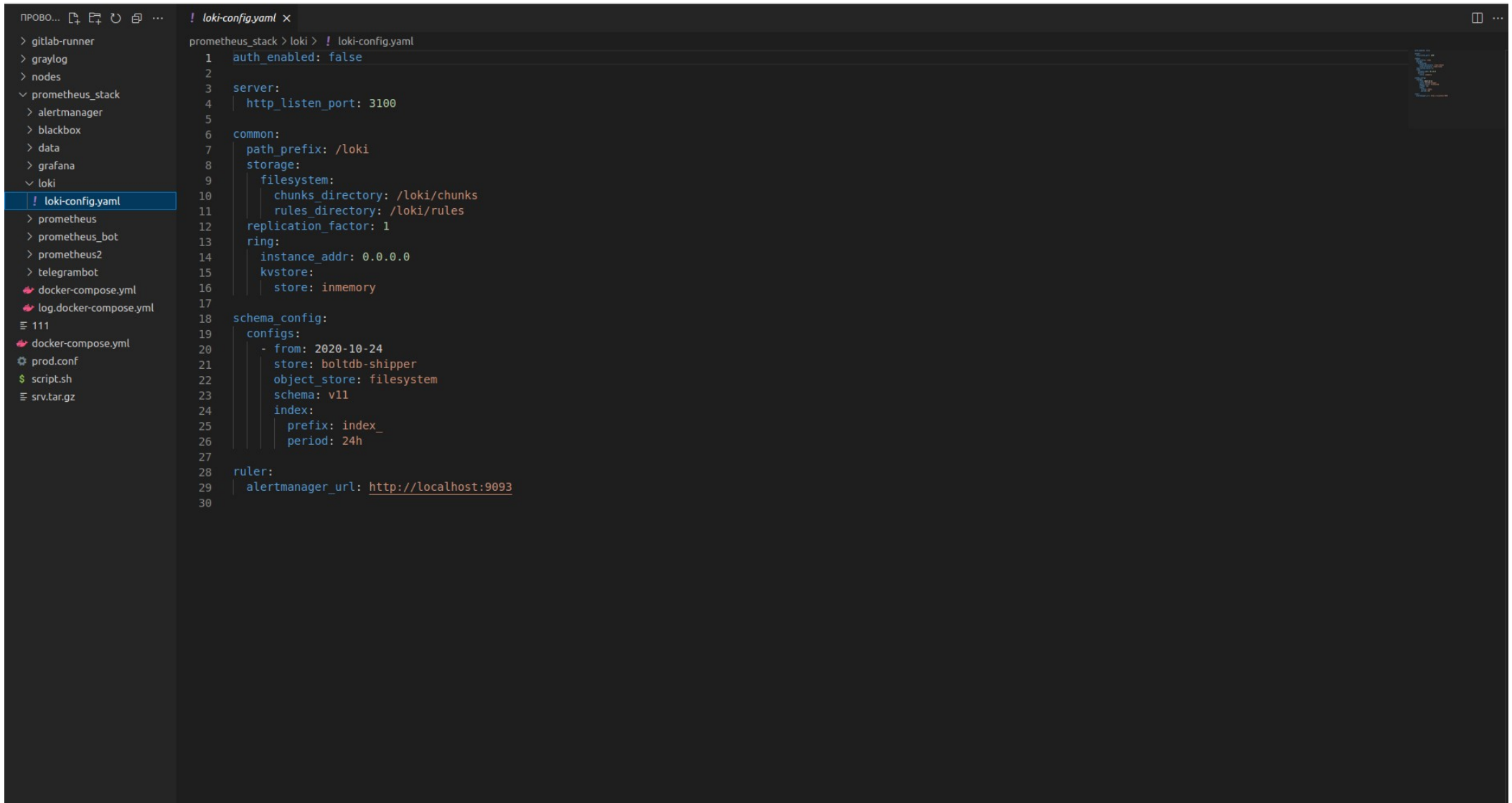
- Вывести логи с помощью Loki в Grafana с удаленного сервера.

# Создаем log.docker-compose.yml файл для loki на сервере мониторинга

A screenshot of a code editor with a dark theme. The left sidebar shows a file tree with a folder named 'prometheus\_stack' containing several sub-files. The 'log.docker-compose.yml' file is selected and highlighted. The main editor area shows the content of this file, which is a Docker Compose configuration for the 'loki' service. The configuration includes a version, a network, and a service definition with image, ports, volumes, command, and networks.

```
prometheus_stack > log.docker-compose.yml
1  version: "3"
2
3  networks:
4    log:
5      external: true
6
7  services:
8    loki:
9      image: grafana/loki:2.4.1
10     ports:
11       - "3100:3100"
12     volumes:
13       - "/home/prometheus_stack/loki/loki-config.yaml:/etc/loki/local-config.yaml"
14     command: -config.file=/etc/loki/local-config.yaml
15     networks:
16       - log
17
18
```

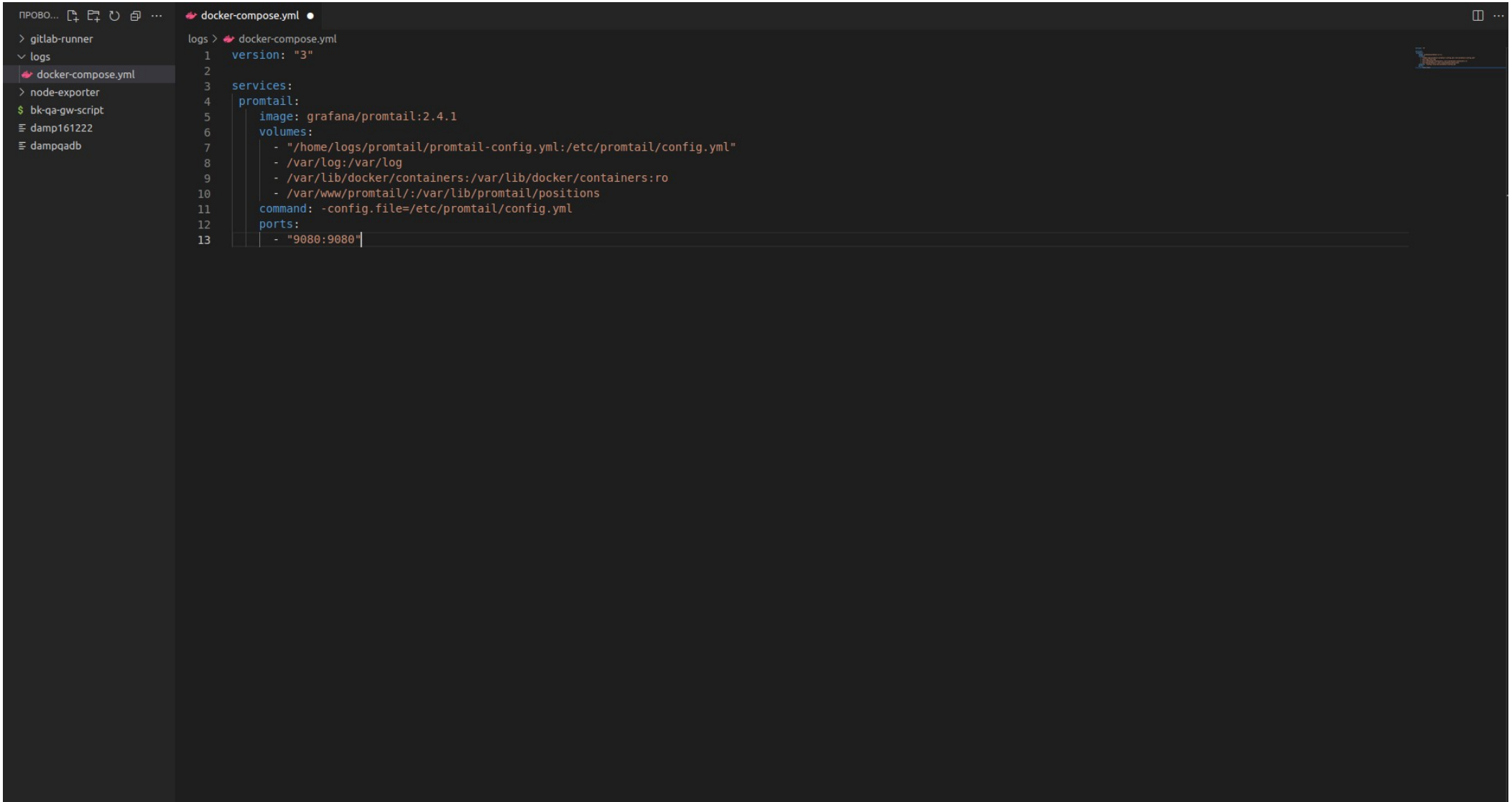
# Создаем папку и конфигурационный файл для loki и запускаем log.docker-compose.yml



The screenshot shows a terminal window with a file explorer on the left and a code editor on the right. The file explorer shows a directory structure for a Prometheus stack, with 'loki' selected. The code editor shows the 'loki-config.yaml' file, which is a configuration for the loki service. The configuration includes settings for authentication, server, common, storage, replication, ring, schema config, and ruler.

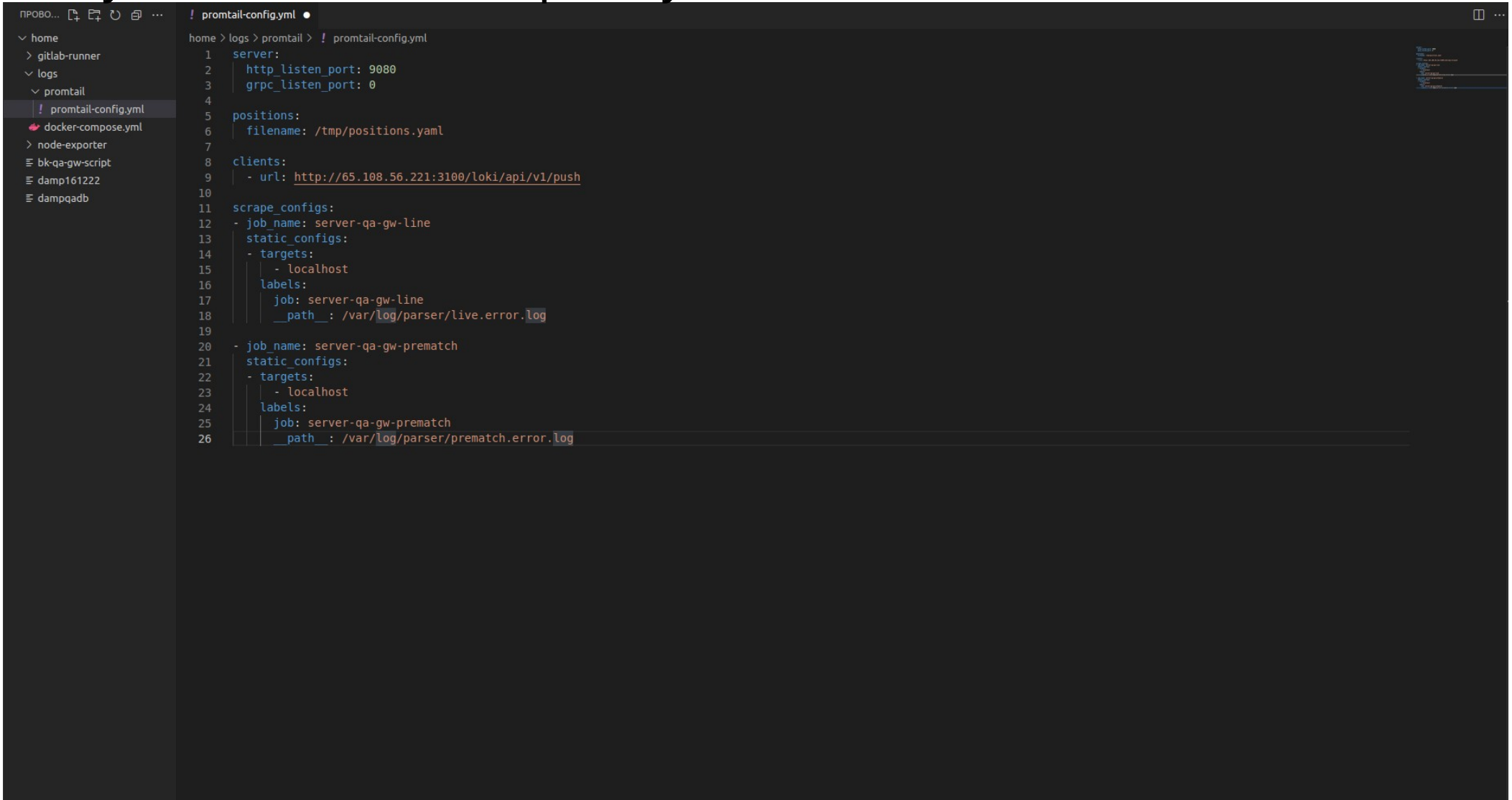
```
prometheus_stack > loki > ! loki-config.yaml
1  auth_enabled: false
2
3  server:
4    http_listen_port: 3100
5
6  common:
7    path_prefix: /loki
8    storage:
9      filesystem:
10       chunks_directory: /loki/chunks
11       rules_directory: /loki/rules
12     replication_factor: 1
13   ring:
14     instance_addr: 0.0.0.0
15     kvstore:
16       store: inmemory
17
18   schema_config:
19     configs:
20       - from: 2020-10-24
21         store: boltdb-shipper
22         object_store: filesystem
23         schema: v11
24         index:
25           prefix: index_
26           period: 24h
27
28   ruler:
29     alertmanager_url: http://localhost:9093
30
```

# На сервере с которого будем собирать логи создаем docker-compose файлы для promtail

A screenshot of a code editor with a dark theme. On the left is a sidebar with a file explorer showing a directory structure: 'logs' containing 'docker-compose.yml', 'node-exporter', 'bk-qa-gw-script', 'damp161222', and 'dampqadb'. The main editor area displays the content of 'docker-compose.yml'. The file is a valid YAML configuration for a Docker Compose service named 'promtail'. It specifies the image 'grafana/promtail:2.4.1', mounts several volumes for configuration and log storage, sets a specific command to run, and maps port 9080. Line numbers 1 through 13 are visible on the left side of the editor.

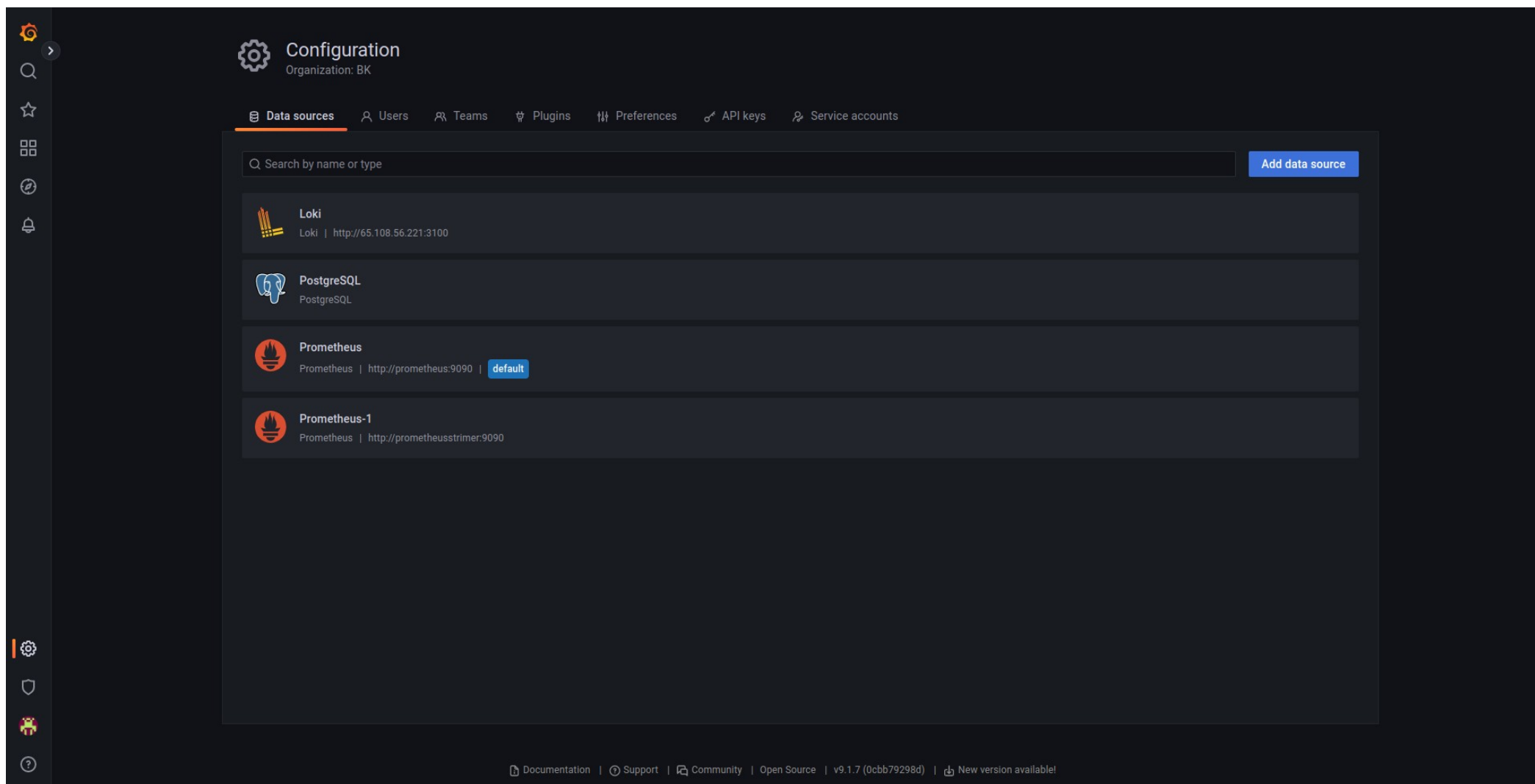
```
logs > docker-compose.yml
1  version: "3"
2
3  services:
4    promtail:
5      image: grafana/promtail:2.4.1
6      volumes:
7        - "/home/logs/promtail/promtail-config.yml:/etc/promtail/config.yml"
8        - "/var/log:/var/log"
9        - "/var/lib/docker/containers:/var/lib/docker/containers:ro"
10       - "/var/www/promtail:/var/lib/promtail/positions"
11      command: -config.file=/etc/promtail/config.yml
12      ports:
13        - "9080:9080"
```

Создаем конфигурационный файл для promtail в котором прописываем url адрес loki и джобы с указанием файла логов которые необходимо отправлять в loki.  
Запускаем docker-compose.yml.

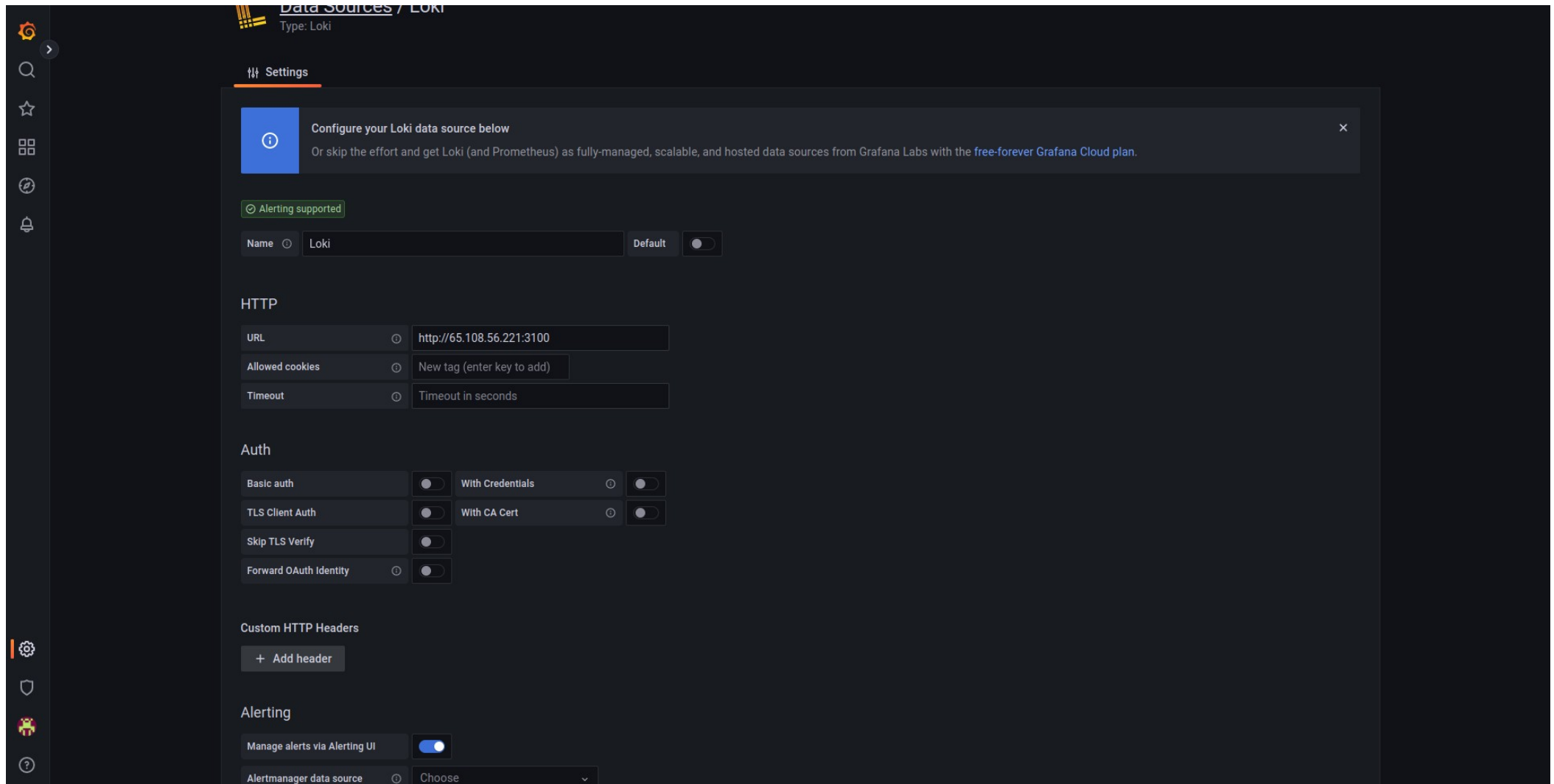


```
! promtail-config.yml
home > logs > promtail > ! promtail-config.yml
1  server:
2    http_listen_port: 9080
3    grpc_listen_port: 0
4
5  positions:
6    filename: /tmp/positions.yaml
7
8  clients:
9    - url: http://65.108.56.221:3100/loki/api/v1/push
10
11 scrape_configs:
12   - job_name: server-qa-gw-line
13     static_configs:
14       - targets:
15         - localhost
16         labels:
17           job: server-qa-gw-line
18           __path__: /var/log/parser/live.error.log
19
20   - job_name: server-qa-gw-prematch
21     static_configs:
22       - targets:
23         - localhost
24         labels:
25           job: server-qa-gw-prematch
26           __path__: /var/log/parser/prematch.error.log
```

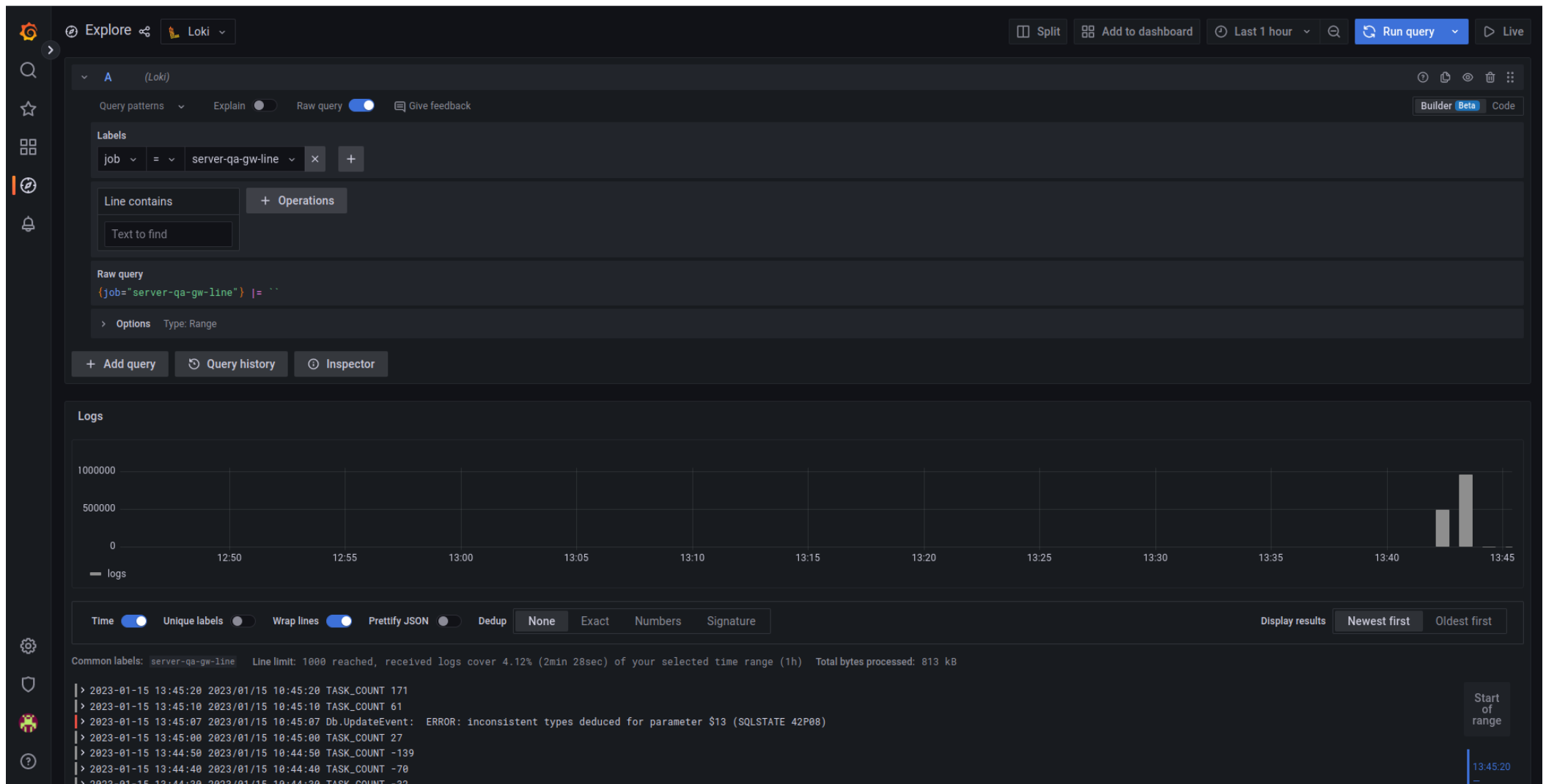
# Заходим в grafana в раздел data sources и добавляем источник loki



# Настраиваем источник данных указав url loki



В grafana переходим в раздел explore выбираем источник данных loki  
В Labels выбираем job и из списка выбираем название соответствующее названию джобы в настройках promtail нажимаем Run.  
Через ярлык Add to dashboard можем добавить список логов на нужный над дажборд.







New dashboard



Last 6 hours



### New Panel

```
> 2023/01/15 11:08:00 TASK_COUNT 1085
> 2023/01/15 11:07:50 TASK_COUNT 484
> 2023/01/15 11:07:40 TASK_COUNT 519
> 2023/01/15 11:07:30 TASK_COUNT 570
> 2023/01/15 11:07:20 TASK_COUNT 569
> 2023/01/15 11:07:12 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
> 2023/01/15 11:07:10 TASK_COUNT 568
> 2023/01/15 11:07:09 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
> 2023/01/15 11:07:09 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
> 2023/01/15 11:07:08 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
> 2023/01/15 11:07:08 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
> 2023/01/15 11:07:08 Db.UpdateEvent: ERROR: inconsistent types deduced for parameter $13 (SQLSTATE 42P08)
```