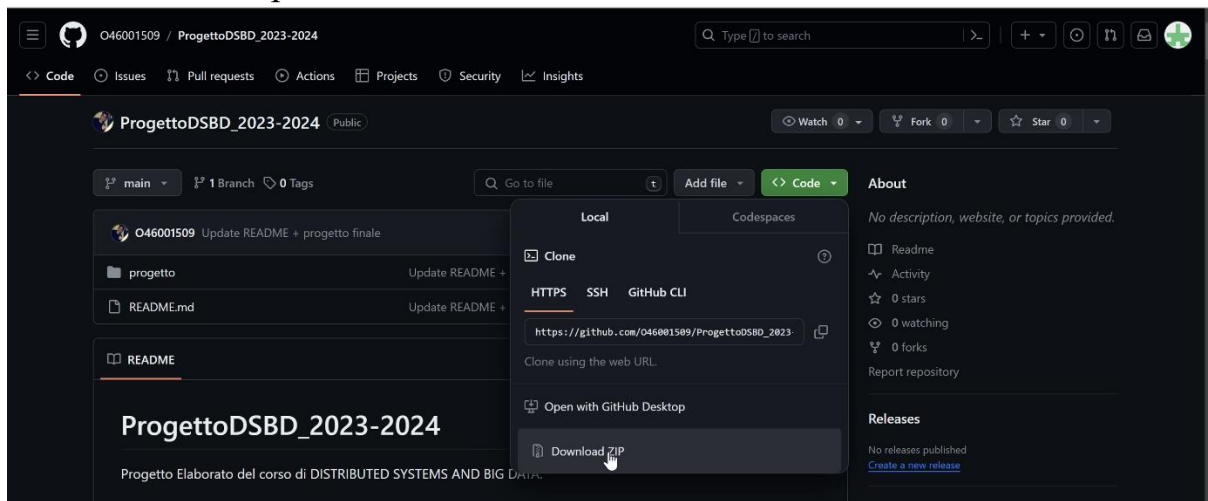
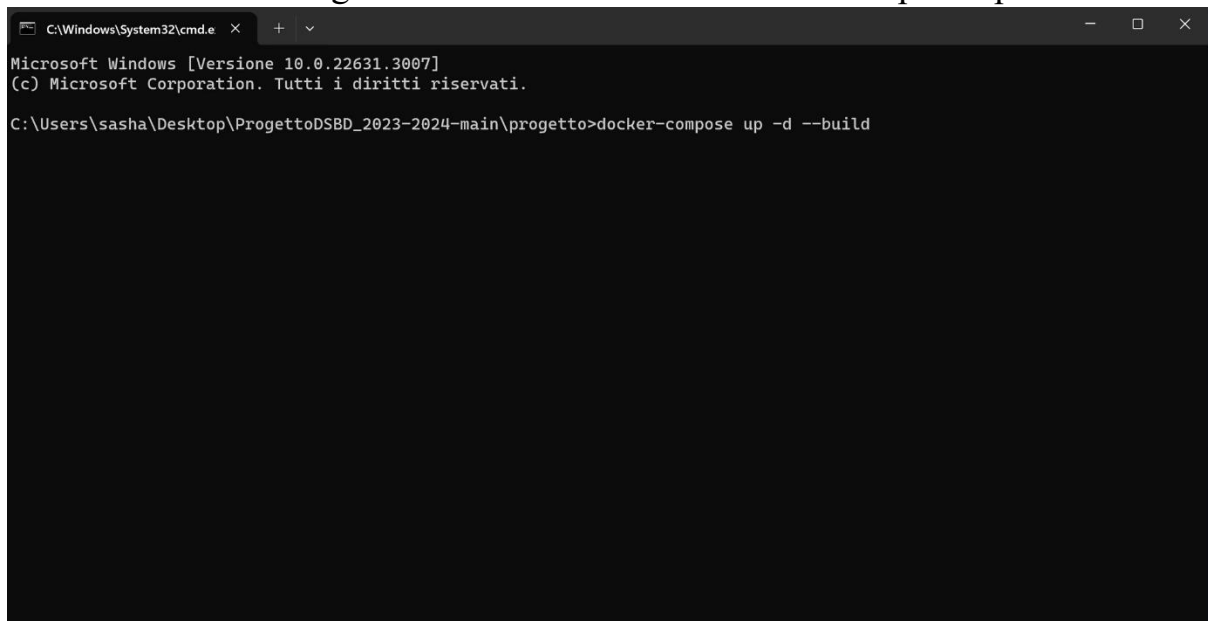


Abbiamo testato il sistema eseguendo i seguenti passi:

1. Scaricare il file zip da GitHub:



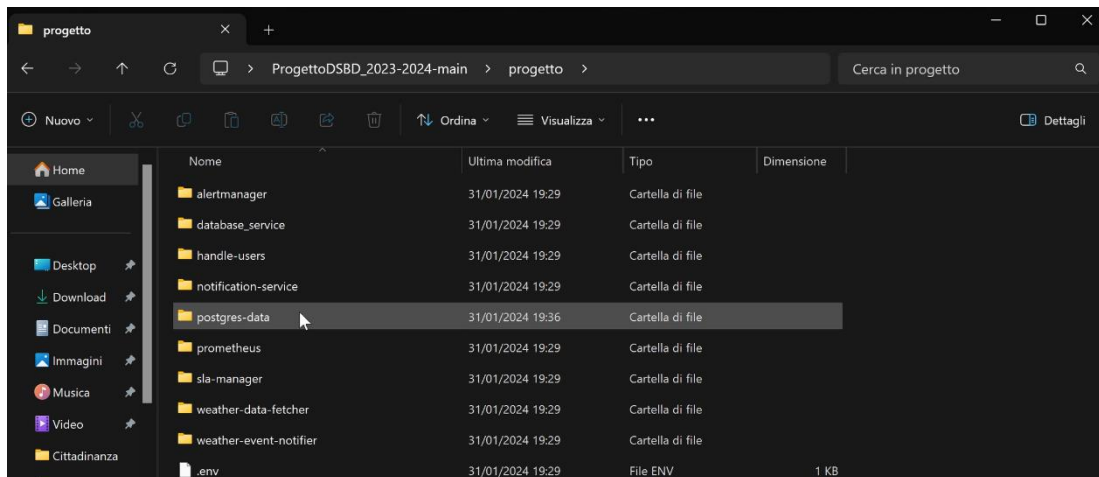
2. Tramite terminale eseguiamo il comando → docker—compose up -d --build



3. Nel caso di un errore come nell'immagine:

```
C:\Windows\System32\cmd.e x + v
=> => writing image sha256:7c9f448f6f842d4dc610fa638ca44927be4e7232c8e9874d14bdae7e228f91a 0.0s
=> => naming to docker.io/library/progetto-handle-users 0.0s
=> [weather-data-fetcher] exporting to image 0.3s
=> => exporting layers 0.3s
=> => writing image sha256:3afc6ed5c4b34f8293c4ec4e640164c9a52ce8681641c306fe4736a675f46ae0 0.0s
=> => naming to docker.io/library/progetto-weather-data-fetcher 0.0s
=> [sla-manager] exporting to image 0.9s
=> => exporting layers 0.9s
=> => writing image sha256:c80d752b81954505bce410413321d94628309fafe49491eecd50fd69328e55 0.0s
=> => naming to docker.io/library/progetto-sla-manager 0.0s
[+] Running 13/13
✓ Network progetto_weather-net Created 0.1s
✓ Network progetto_default Created 0.1s
✓ Container progetto-alertmanager-1 Started 0.3s
✓ Container progetto-cadvisor-1 Started 0.3s
✓ Container prometheus Started 0.3s
✓ Container postgres Started 0.3s
✓ Container progetto-notification-service-1 Started 0.2s
✓ Container progetto-weather-event-notifier-1 Started 0.2s
✓ Container progetto-database-service-1 Started 0.2s
✓ Container progetto-grafana-1 Started 0.2s
✓ Container progetto-weather-data-fetcher-1 Started 0.2s
✓ Container progetto-handle-users-1 Started 0.2s
✓ Container progetto-sla-manager-1 Started 0.2s
C:\Users\sasha\Desktop\ProgettoDSBD_2023-2024-main\progetto>docker-compose exec -it postgres psql -U postgres
psql: error: connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: No such file or directory
Is the server running locally and accepting connections on that socket?
```

Bisogna eliminare il contenuto della cartella postgres che si trova dentro il progetto:



Dopo aver eliminato il contenuto della cartella postgres e riavviato il progetto, creare il database (weather_searches) come nella seguente immagine:

```
C:\Windows\System32\cmd.e x + v
postgres | 2024-01-31 18:41:29.530 UTC [55] LOG: checkpoint starting: shutdown immediate
postgres | 2024-01-31 18:41:29.778 UTC [55] LOG: checkpoint complete: wrote 3 buffers (0.0%); 0 WAL file(s) added, 0 r
postgres | 2024-01-31 18:41:29.861 UTC [54] LOG: database system is shut down
postgres | done
postgres | server stopped
postgres | PostgreSQL init process complete; ready for start up.
postgres | 2024-01-31 18:41:30.048 UTC [1] LOG: starting PostgreSQL 16.1 (Debian 16.1-1.pgdg120+1) on x86_64-pc-linux-
postgres | 2024-01-31 18:41:30.062 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432
postgres | 2024-01-31 18:41:30.062 UTC [1] LOG: listening on IPv6 address "::", port 5432
postgres | 2024-01-31 18:41:30.106 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"
postgres | 2024-01-31 18:41:30.210 UTC [68] LOG: database system was shut down at 2024-01-31 18:41:29 UTC
postgres | 2024-01-31 18:41:30.318 UTC [1] LOG: database system is ready to accept connections
postgres | 2024-01-31 18:41:31.775 UTC [72] FATAL: database "weather_searches" does not exist
postgres | canceled
C:\Users\sasha\Desktop\ProgettoDSBD_2023-2024-main\progetto>docker-compose exec -it postgres psql -U postgres
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# \c weather_searches
connection to server on socket "/var/run/postgresql/.s.PGSQL.5432" failed: FATAL: database "weather_searches" does not
exist
Previous connection kept
postgres=# CREATE DATABASE weather_searches;
```

4. Visualizziamo le seguenti tabelle (users, subscriptions, sla_definitions, sla_violations) dentro il nostro database:

```
C:\Windows\System32\cmd.exe x + v
Container progetto-handle-users-1 Started 0.1s
Container progetto-weather-data-fetcher-1 Started 0.1s

C:\Users\sasha\Desktop\ProgettoDSBD_2023-2024-main\progetto>docker-compose exec -it postgres psql -U postgres
psql (16.1 (Debian 16.1-1.pgdg120+1))
Type "help" for help.

postgres=# \c weather_searches
You are now connected to database "weather_searches" as user "postgres".
weather_searches=# SELECT * FROM users;
 id | user_name | chat_id | interval
-----+-----+-----+-----
(0 rows)

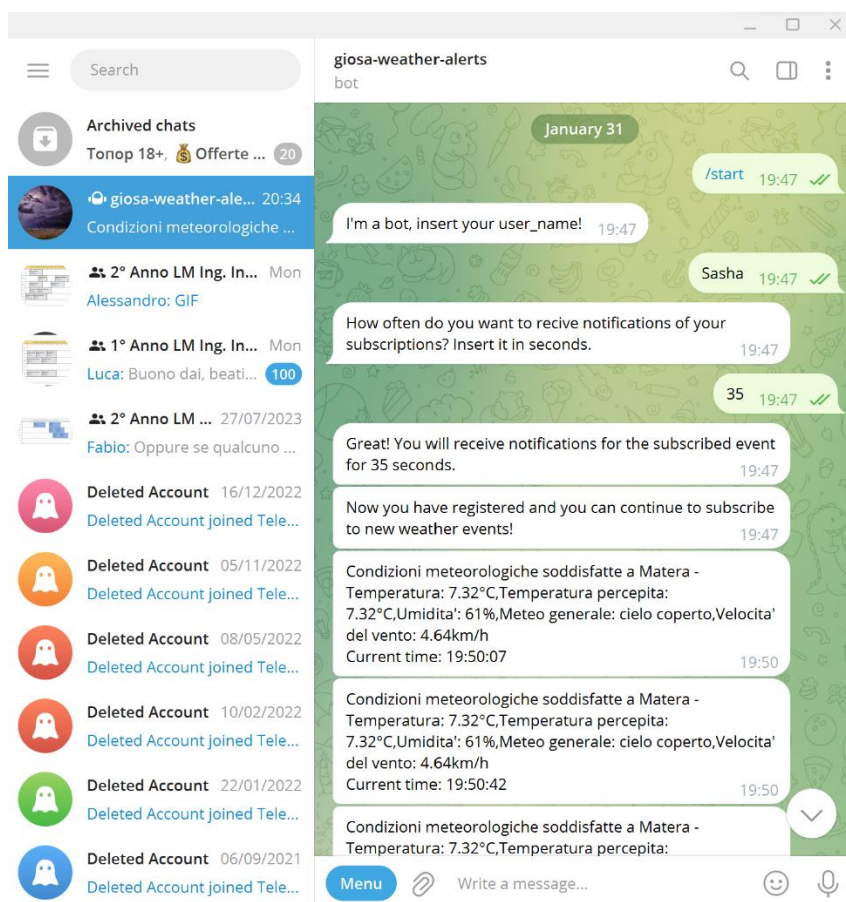
weather_searches=# SELECT * FROM subscriptions;
 id | user_name | citta | condizioni
-----+-----+-----+-----
(0 rows)

weather_searches=# SELECT * FROM sla_definitions;
 sla_id | metric_name | threshold | description
-----+-----+-----+-----
(0 rows)

weather_searches=# SELECT * FROM sla_violations;
 violation_id | sla_id | violation_time | actual_value
-----+-----+-----+-----
(0 rows)

weather_searches=# SELECT * FROM users;
```

5. Nel telegram cerchiamo il bot (giosa-weather-alerts) e cominciamo la conversazione con lui eseguendo la registrazione come mostrato nella seguente immagine:



Per il test abbiamo supposto che l'utente scelga un intervallo notifiche pari a 35 secondi (si può scegliere più grande, ex: 1 minuto, 5 minuti, ecc.) per velocizzare il test dell'applicazione (per esempio non dover aspettare troppo per vedere le violazioni).

6. Creazione della metrica `notification_interval_second`, inserimento di una nuova sottoscrizione e aggiornamento della sottoscrizione inserita:

```
$url = "http://localhost:5005/sla"
>> $headers = @{"Content-Type" = "application/json"}
>>
>> $body = @{
>>     'metric_name' = 'notification_interval_seconds'
>>     'threshold' = 10
>>     'description' = 'La differenza tra l intervallo effettivo delle notifiche e l intervallo previsto non deve superare il 10% '
>> } | ConvertTo-Json
>> Invoke-WebRequest -Uri $url -Method Post -Headers $headers -Body $body
```

La metrica inserita ha una soglia di 10. Questo significa che si verificherà una violazione nel caso in cui viene superato del 10% l'intervallo notifiche scelto dall'utente.

```
$url = "http://localhost:5001/sottoscrizioni"
>> $headers = @{"Content-Type" = "application/json"}
>>
>> $body = @{
>>     "user_name" = "Sasha"
>>     "citta" = "Matera"
>>     "condizioni" = @{
>>         "temperatura_massima" = 40
>>         "temperatura_minima" = -34
>>         "vento_max" = 99
>>         "umidita_max" = 99
>>     }
>> } | ConvertTo-Json
>> Invoke-WebRequest -Uri $url -Method Post -Headers $headers -Body $body
```

Anche qui per fare in modo di non dover aspettare il verificarsi delle condizioni della sottoscrizione inserita, abbiamo inserito: un range abbastanza grande per la temperatura, e un valore grande per il vento e per l'umidità. In un contesto reale l'utente potrebbe inserire le condizioni che vuole.

```
$url = "http://localhost:5001/sottoscrizioni"
>> $headers = @{"Content-Type" = "application/json"}
>>
>> $body = @{
>>     "user_name" = "Sasha"
>>     "citta" = "Crotone"
>>     "condizioni" = @{
>>         "temperatura_massima" = 20
>>         "temperatura_minima" = 9
>>         "vento_max" = 9
>>         "umidita_max" = 20
>>     }
>> } | ConvertTo-Json
>> Invoke-WebRequest -Uri $url -Method Put -Headers $headers -Body $body
```

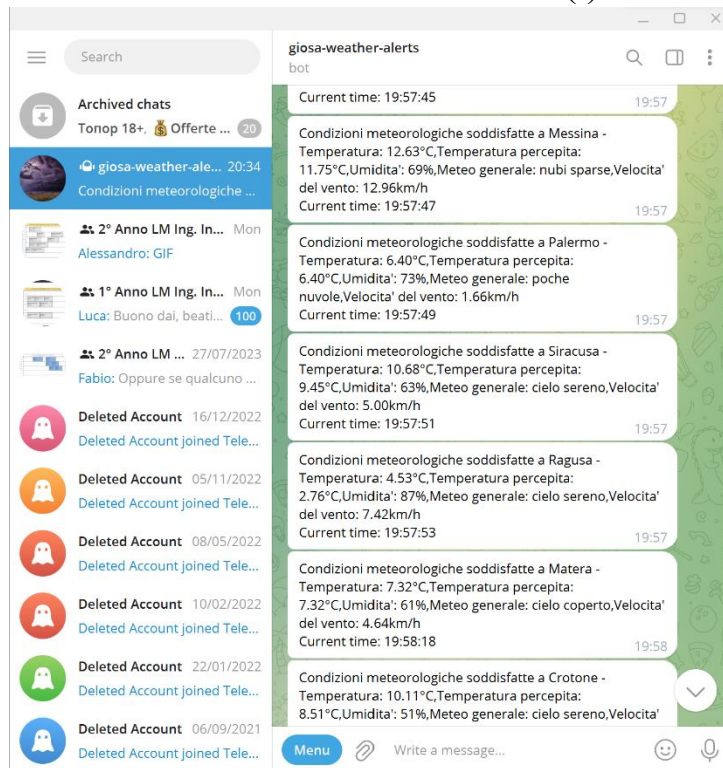
7. Visualizzare logs dei servizi: (docker-compose logs database-service -f, docker-compose logs notification-service -f, docker-compose logs sla-manager -f):

```
C:\Windows\System32\cmd.e x + v - □ x
progetto-database-service-1 | 2024-01-31 20:02:03 - INFO - 172.25.0.7 -
- [31/Jan/2024 19:02:03] "GET /recupera_utente?user_name=Sasha&chat_id=
659173906 HTTP/1.1" 200 -
progetto-database-service-1 | 2024-01-31 20:02:03 - INFO - 172.25.0.7 -
- [31/Jan/2024 19:02:03] "GET /recupera_utente?user_name=Sasha&chat_id=
659173906 HTTP/1.1" 200 -
progetto-database-service-1 | 2024-01-31 20:02:03 - INFO - 172.25.0.7 -
- [31/Jan/2024 19:02:03] "GET /recupera_utente?user_name=Sasha&chat_id=
659173906 HTTP/1.1" 200 -
progetto-database-service-1 | 2024-01-31 20:02:03 - INFO - Inserimento
sottoscrizione - User name: Sasha, chat_id: Kiev, conditions: {'vento_ma
x': 99, 'temperatura_minima': -34, 'umidita_max': 99, 'temperatura_massi
ma': 40}
progetto-database-service-1 | 2024-01-31 20:02:03 - INFO - Inserimento
sottoscrizione - User name: Sasha, chat_id: Kiev, conditions: {'vento_ma
x': 99, 'temperatura_minima': -34, 'umidita_max': 99, 'temperatura_massi
```

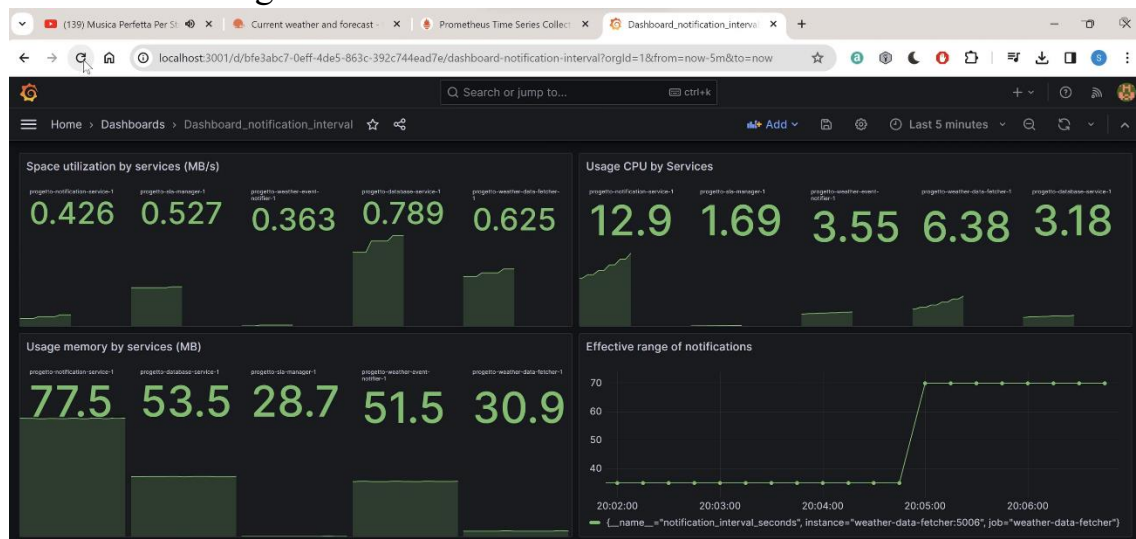
```
C:\Windows\System32\cmd.e x + v - □ x
progetto-notification-service-1 | 2024-01-31 19:57:47 - INFO - 172.25.0
.11 - - [31/Jan/2024 18:57:47] "POST /notifiche HTTP/1.1" 200 -
progetto-notification-service-1 | 2024-01-31 19:57:48 - INFO - HTTP Req
uest: POST https://api.telegram.org/bot6891484766:AAFPTKTsq0RiynexY1bgc
1Q0B73jFpEn-A/sendMessage "HTTP/1.1 200 OK"
progetto-notification-service-1 | 2024-01-31 19:57:48 - INFO - 172.25.0
.11 - - [31/Jan/2024 18:57:48] "POST /notifiche HTTP/1.1" 200 -
progetto-notification-service-1 | 2024-01-31 19:57:50 - INFO - HTTP Req
uest: POST https://api.telegram.org/bot6891484766:AAFPTKTsq0RiynexY1bgc
1Q0B73jFpEn-A/sendMessage "HTTP/1.1 200 OK"
progetto-notification-service-1 | 2024-01-31 19:57:50 - INFO - 172.25.0
.11 - - [31/Jan/2024 18:57:50] "POST /notifiche HTTP/1.1" 200 -
progetto-notification-service-1 | 2024-01-31 19:57:52 - INFO - HTTP Req
uest: POST https://api.telegram.org/bot6891484766:AAFPTKTsq0RiynexY1bgc
1Q0B73jFpEn-A/sendMessage "HTTP/1.1 200 OK"
progetto-notification-service-1 | 2024-01-31 19:57:52 - INFO - 172.25.0
```

```
C:\Windows\System32\cmd.e x + v - □ x
progetto-sla-manager-1 | 2024-01-31 19:49:01 - INFO - 172.25.0.5 - - [3
1/Jan/2024 18:49:01] "GET /metrics HTTP/1.1" 200 -
progetto-sla-manager-1 | 2024-01-31 19:49:06 - INFO - 172.25.0.1 - - [3
1/Jan/2024 18:49:06] "POST /sla HTTP/1.1" 201 -
progetto-sla-manager-1 | 2024-01-31 19:49:17 - INFO - Running job "eval
uate_sla (trigger: interval[0:00:30], next run at: 2024-01-31 18:49:47 U
TC)" (scheduled at 2024-01-31 18:49:17.439150+00:00)
progetto-sla-manager-1 | 2024-01-31 19:49:17 - INFO - Lista metriche-->
[{'description': 'La differenza tra l intervallo effettivo delle notific
he e l intervallo previsto non deve superare il 10% ', 'metric_name': 'n
otification_interval_seconds', 'sla_id': 1, 'threshold': '10'}]
progetto-sla-manager-1 | 2024-01-31 19:49:17 - INFO - Metric-->{'descri
ption': 'La differenza tra l intervallo effettivo delle notifiche e l in
tervallo previsto non deve superare il 10% ', 'metric_name': 'notificati
on_interval_seconds', 'sla_id': 1, 'threshold': '10'}
progetto-sla-manager-1 | 2024-01-31 19:49:17 - INFO - Intervallo effett
```


8. Visualizzazione del meteo desiderato(i):

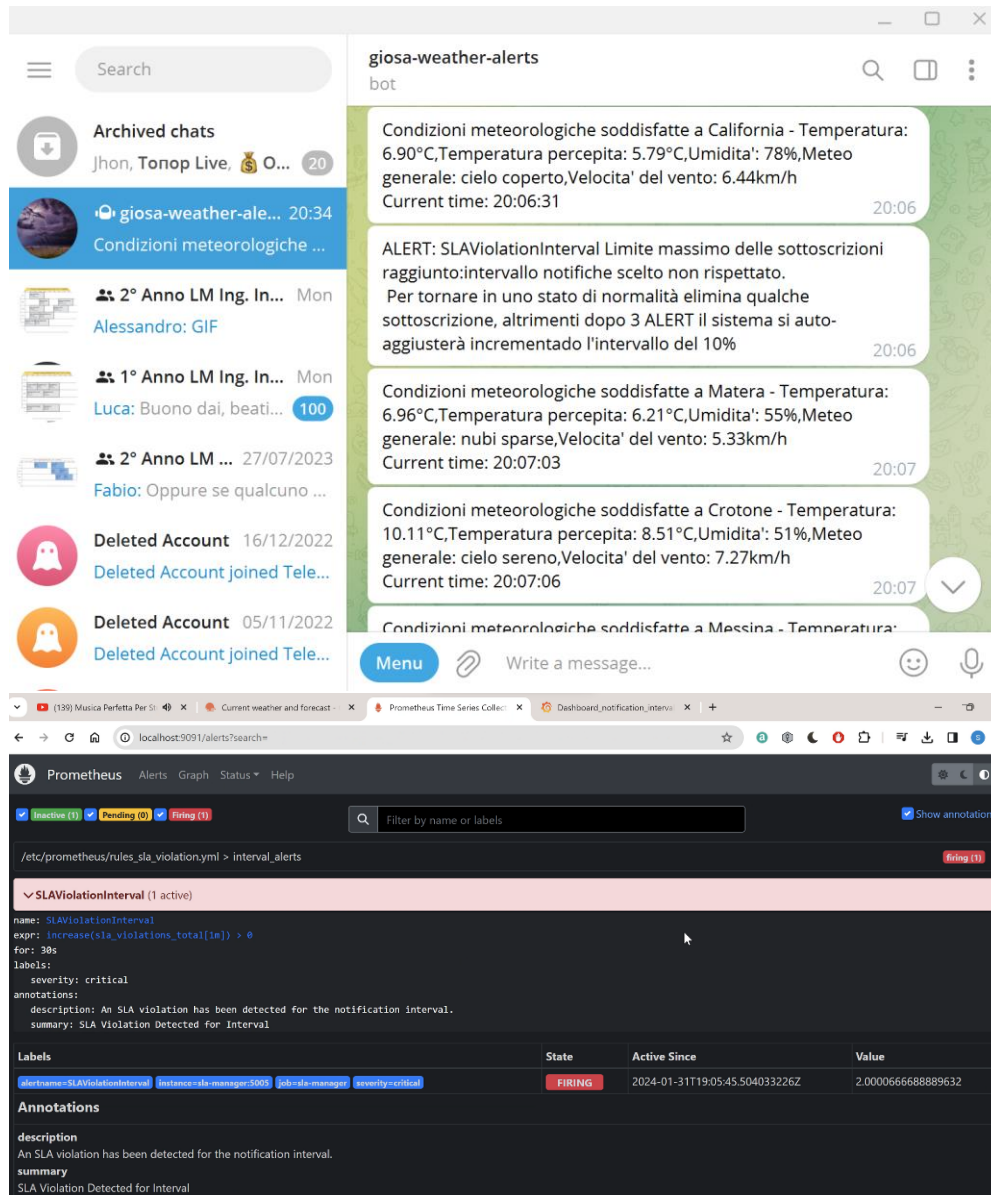


9. Visualizzazione grafici Grafana:



In basso a destra, si vedono, nel grafico del range effettivo di notifiche, dei valori che sono maggiori dell'intervallo scelto dall'utente, secondo la soglia scelta nella metrica (10%). Quindi: $\text{intervallo_effettivo} > 10\% \text{ di intervallo notifiche}$ Ex: $70 > 39$, dove $39 = 35 + (10\% \text{ di } 35)$

10. Nelle seguenti foto si vede l'Alert riferito alla metrica `notification_interval_seconds`, che viene inviato dal `notification-service` all'utente:



11. Come potrebbe eliminare le sottoscrizioni l'utente:

```
InputFields      : {}
Links            : {}
ParsedHtml       : mshtml.HTMLDocumentClass
RawContentLength : 59

Invoke-WebRequest -Uri "http://localhost:5001/sottoscrizioni" -Method Delete -Headers @{"Content-Type":"application/json"} -Body '{"user_name": "Sasha", "citta": "Matera"}'
```

12. Nella seguente immagine si vede il tempo aggiornato dell'arrivo delle notifiche da telegram all'utente dopo l'esecuzione di un'operazione correttiva (se l'utente non ha ancora eliminato alcune sottoscrizioni dopo che gli sono arrivati 3 Alerts), che nel nostro caso mirra ad aumentare l'intervallo notifiche dell'utente per fare rientrare tutte le sottoscrizioni scelte:

```
C:\Windows\System32\cmd.exe x + v
(0 rows)

weather_searches=# SELECT * FROM users;
 id | user_name | chat_id | interval
-----+-----+-----+-----
  1 | Sasha    | 770285017 |      35
(1 row)

weather_searches=# SELECT * FROM sla_definitions;
weather_searches=# SELECT * FROM subscriptions;
weather_searches=# SELECT * FROM subscriptions;
weather_searches=# SELECT * FROM subscriptions;
weather_searches=# SELECT * FROM users;
 id | user_name | chat_id | interval
-----+-----+-----+-----
  1 | Sasha    | 770285017 |      39
(1 row)

weather_searches=# SELECT * FROM users;
 id | user_name | chat_id | interval
-----+-----+-----+-----
  1 | Sasha    | 770285017 |      43
(1 row)
```

13. Nelle seguenti immagini si vede l'inserimento di una nuova metrica `usage_memory_by_service` e l'eliminazione

```
>> $headers = @{"Content-Type" = "application/json"}
>>
>> $body = @{
>>     'metric_name' = 'usage_memory_by_service'
>>     'threshold' = 65
>>     'description' = 'Numero massimo di richieste di dati meteorologici ammesse per evitare il sovraccarico del servizio'
>> } | ConvertTo-Json
>>
>> Invoke-WebRequest -Uri $url -Method Post -Headers $headers -Body $body
```

```
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS
InputFields : {}
Links : {}
ParsedHtml : mshtml.HTMLDocumentClass
RawContentLength : 59

Invoke-WebRequest -Uri "http://localhost:5005/sla" -Method Delete -Headers @{"Content-Type"="application/json"} -Body '{"sla_id" : 2}'
```


14. Nelle seguenti immagini si vede l'Alert riguardo (usage_memory_by_service)

