Progettazione e sviluppo di una piattaforma Web per la generazione visuale di query per tecnologie SIEM

Facoltà di Ingegneria dell'informazione, informatica e statistica presso UNITELMA
Corso di Laurea in Informatica

Alessio Giovannini

Responsabile: Angelo Monti

Co-responsabile: Alessio Dalla Piazza

SAPIENZA
UNIVERSITÀ DI ROMA

UnitelmaSapienza
Università degli Studi di Roma

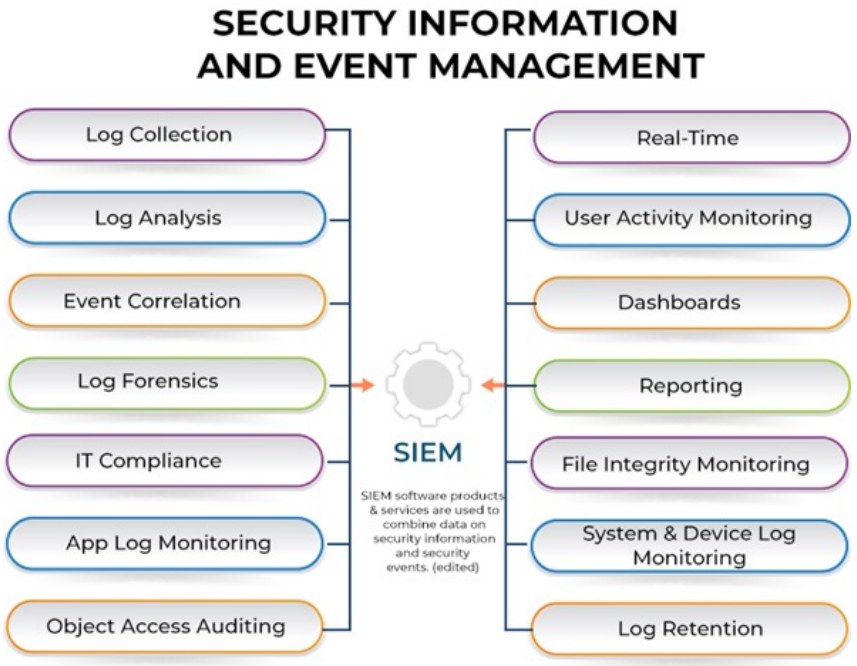
Scenario di riferimento

Scenario di riferimento: Le tecnologie SIEM e il loro ruolo nella sicurezza difensiva

Un **Security Information and Event Management (SIEM)** è una soluzione software di supporto per la rilevazione, analisi e risposta alle minacce alla sicurezza. Un SIEM è composto dalla combinazione di due soluzioni:

- Il **Security Information Management (SIM):**
  - Raccolta;
  - Monitoraggio;
  - Analisi.
- Il **Security Event Management (SEM):**
  - Registrazione eventi;
  - Valutazione eventi.

Le tecnologie SIEM vengono spesso adottate all'interno dei SOC, in singolo o combinati con altre tecnologie affini come EDR, IDS e molte altre.



Scenario di riferimento: Security Operation Center (SOC)

Un **Security Operation Center (SOC)** rappresenta un'unità che si occupa di tenere in sicurezza la rete dell'azienda presso cui opera, monitorando e rispondendo agli incidenti di Cybersecurity.

All'interno di un SOC sussistono diversi ruoli con diversi gradi di responsabilità:

- **Triage Specialist** (1° Livello);
- **Incident Responder** (2° Livello);
- **Threat Hunter** (3° Livello);
- **SOC Manager**.

Scenario di riferimento: necessità della piattaforma

- Lo scopo di rendere facile ed accessibile uno strumento come **sigma** attraverso un unico portale web:
  - Generando graficamente delle regole e generare query SIEM specifiche;
  - Testing di regole già pronte;
  - Aiuto automatico nella generazione delle regole.
- Esonerare l'utente finale da problematiche relative a:
  - Preparazione dell'ambiente di lavoro;
  - Gestione delle dipendenze imposte dal software originale;
  - Apprendimento nell'uso dello strumento originale disponibile solo da riga di comando (CLI).

Analisi dei requisiti

Analisi dei requisiti: requisiti della piattaforma

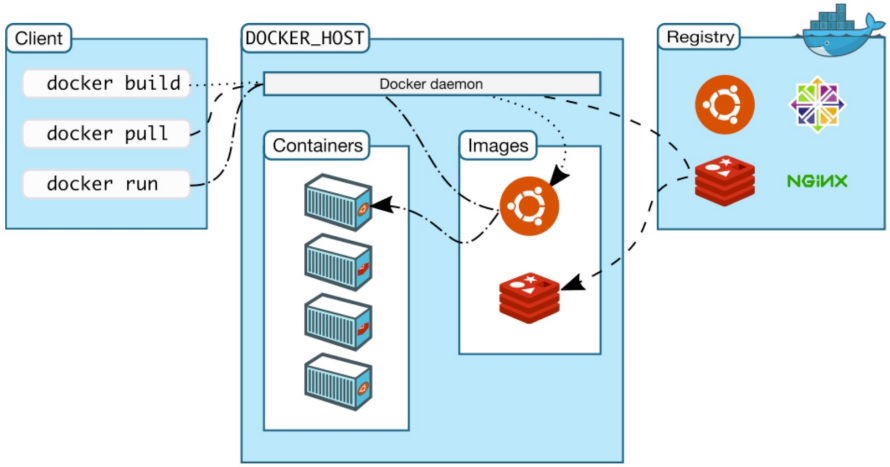
- Creazione della regola;
- Caricamento di una regola;
- Salvataggio della regola;
- Controllo della validità sintattica della regola;
- Generazione della query a partire dalla regola.

Funzionalità essenziali per permettere la corretta generazione di una query per tecnologia SIEM specifica con relativa configurazione

Analisi dei requisiti: raffinamento e funzionalità aggiuntive

Approccio basato sui container Docker

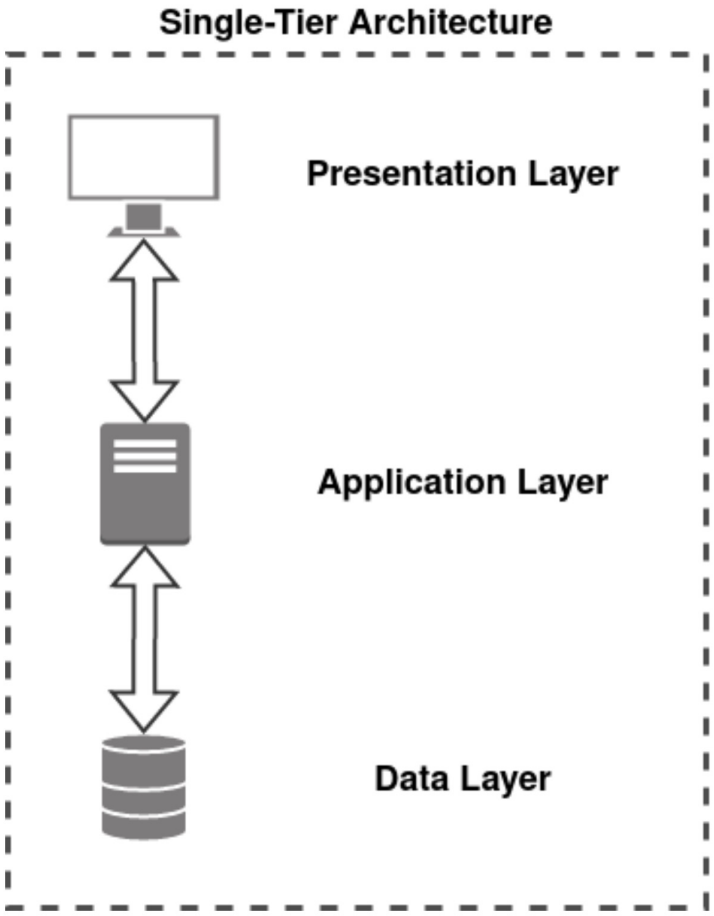
- Isolamento delle risorse e maggiore sicurezza;
- Dipendenze software soddisfatte all'interno del container;
- Portabilità dell'applicativo verso qualsiasi piattaforma che supporti Docker.



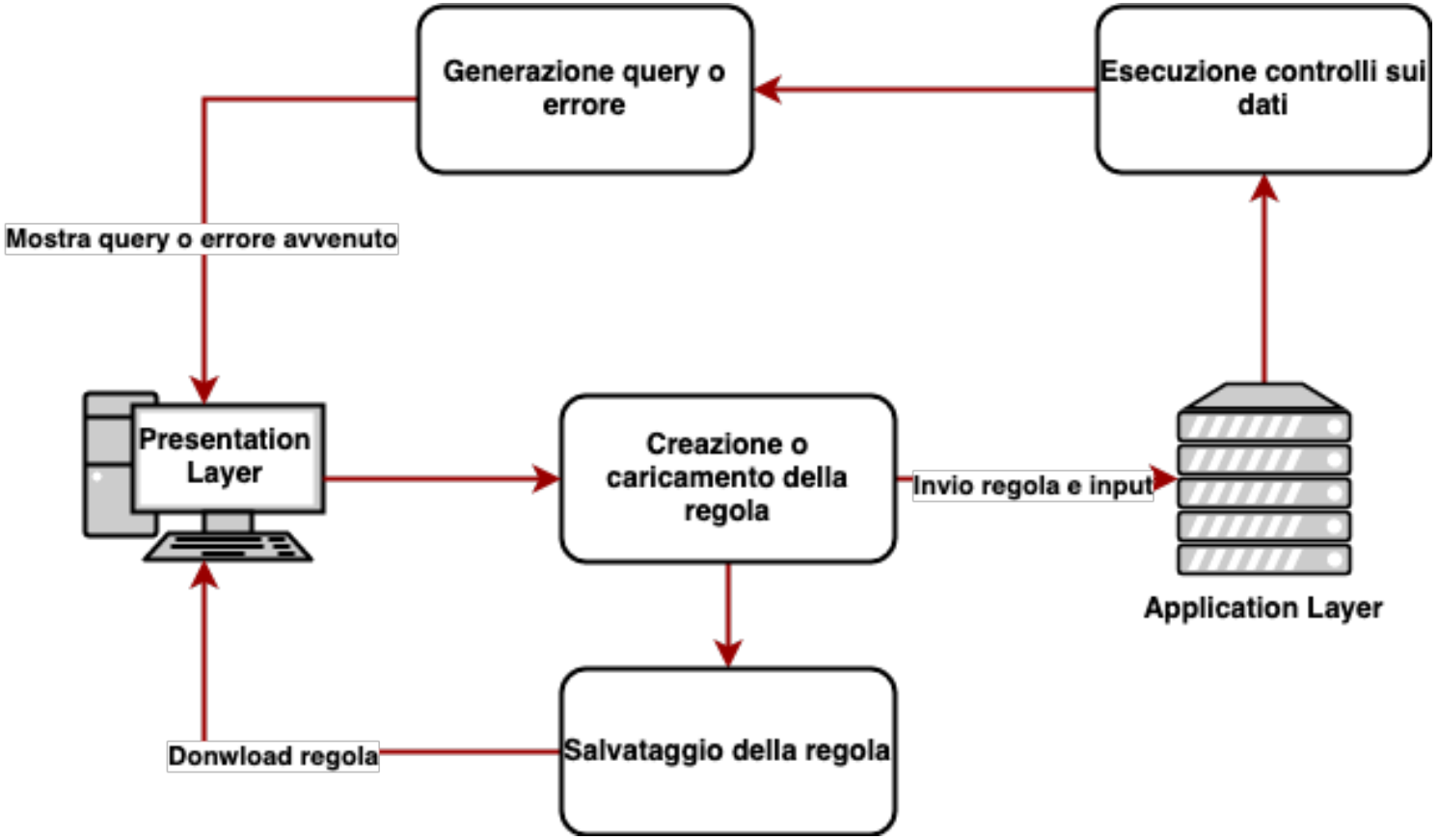
Progettazione

Progettazione: architettura

- **Presentation Layer (User Interface).** Si occupa di organizzare e mostrare i risultati forniti dall'applicativo tramite un'interfaccia che l'utente che utilizza l'applicativo può comprendere;
- **Application Layer (Backend Login)**. Si occupa della logica dell'applicativo e ha lo scopo di processare i dati tra il Presentation Layer e il Data Layer;
- **Data Layer (Database).** Si occupa della conservazione delle informazioni e si occupa di mandare quest'ultima all' Application Layer.

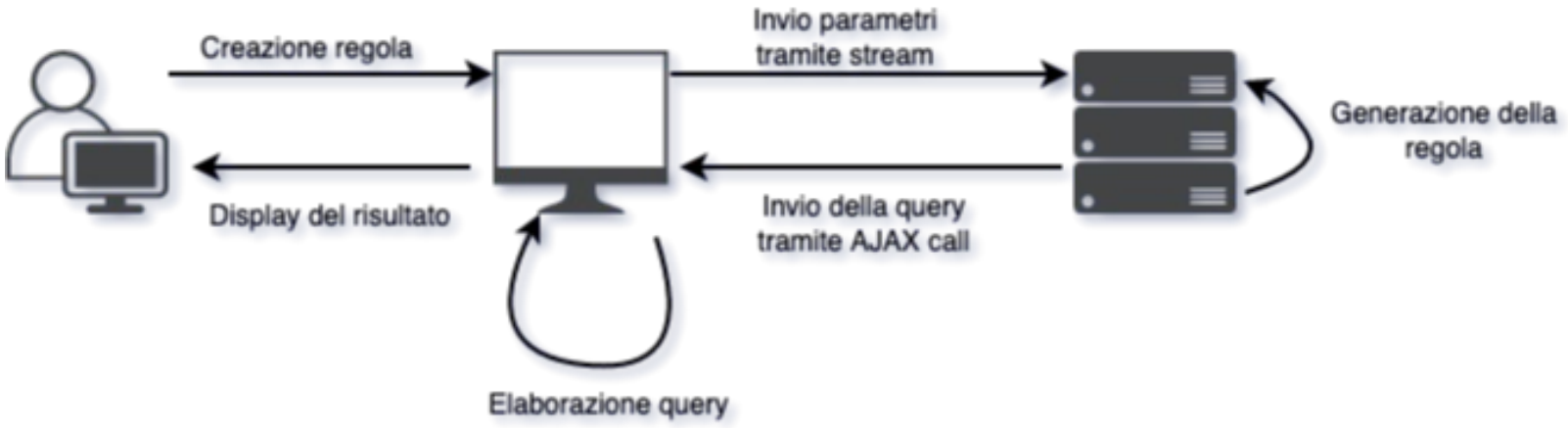


Progettazione: architettura



Progettazione: Approccio basato su stream

- Azzeramento delle operazioni I/O su disco;
- Maggiore velocità di esecuzione;
- Richieste non bloccanti.



Implementazione

Implementazione: Application Layer

- **Parte logica ed applicativa della piattaforma**: controllo sui dati forniti dall'utente, vincoli derivanti dai requisiti e implementazione delle funzionalità;
- Trasmissione dei dati mediante l'utilizzo degli **stream**;
- **Core di sigma modificato** per interfacciare l'applicativo al **server web**;

- **Flask & moduli python custom**, framework che permette la creazione e la gestione di un server web out of the box.

Implementazione: Application Layer

- Definizione delle **route** per definire i percorsi raggiungibili via richieste HTTP/S;
- Definizione del metodo di interfaccia tra il server web e il **core di sigma modificato**.

```python
from flask import Flask, render_template, url_for, request
import custom_sigmac

# The name of Flask Application
sigma_app = Flask(__name__)

#
    ------------------------------------------------------------------


# ROUTES FOR REACHABLE PAGES

# Homepage
@sigma_app.route("/", methods=["GET"])
@sigma_app.route("/homepage", methods=["GET"])
@sigma_app.route("/index", methods=["GET"])
def homepage():
    return render_template("index.html")

# Documentation page
@sigma_app.route("/doc", methods=["GET"])
@sigma_app.route("/documentation", methods=["GET"])
def doc():
    return render_template("documentation.html")

#
    ------------------------------------------------------------------


# ROUTES FOR ERROR HANDLING

# 404 error handler
@sigma_app.errorhandler(404)
def error_handler(error):
    return render_template("404.html")

#
    ------------------------------------------------------------------


# ROUTES FOR BACKEND FUNCTIONALITIES (NO REACHABLE HTML PAGES)
@sigma_app.route("/convert", methods=["POST"])
def sigmaconverter():

    srule = str(request.form.get('srule'))           # the sigma
    rule generated by front-end
    target  =  request.form.get('target')       # this field go
    with `-t` option
    config = request.form.get('config')         # this field go
    with `-c` option
```

Implementazione: Application Layer

```
43      # Check for missing parameters that are required for sigma
        basic usage
44      if(srule==None or target==None or config==None):
45          missing_params = []
46          error = "<h1>The following parameter/s missing:\n"
47          if srule==None:
48              error += "Sigma Rule\n"
49          if target==None:
50              error += "Target\n"
51          if config==None:
52              error += "Configuration\n"
53          error += "</h1>"

55          return error

57      result = custom_sigmac.CustomSigmac([str(srule)], target, [
        config+".yml"]).convert()
58      print("RESULT: ",result)
59      return result #DEBUG


62  #
        ------------------------------------------------------------------


64  # Use this for DEVELOPMENT DEPLOYMENT
65  if __name__ == "__main__":
66      sigma_app.run(debug=True) # Use this command for DEVELOPMENT
        DEPLOYMENT

68  # Use the next commands for PRODUCTION DEPLOYMENT
69  #   from waitress import serve
70  #   serve(sigma_app, host="0.0.0.0", port=8080)
```

Implementazione: Application Layer

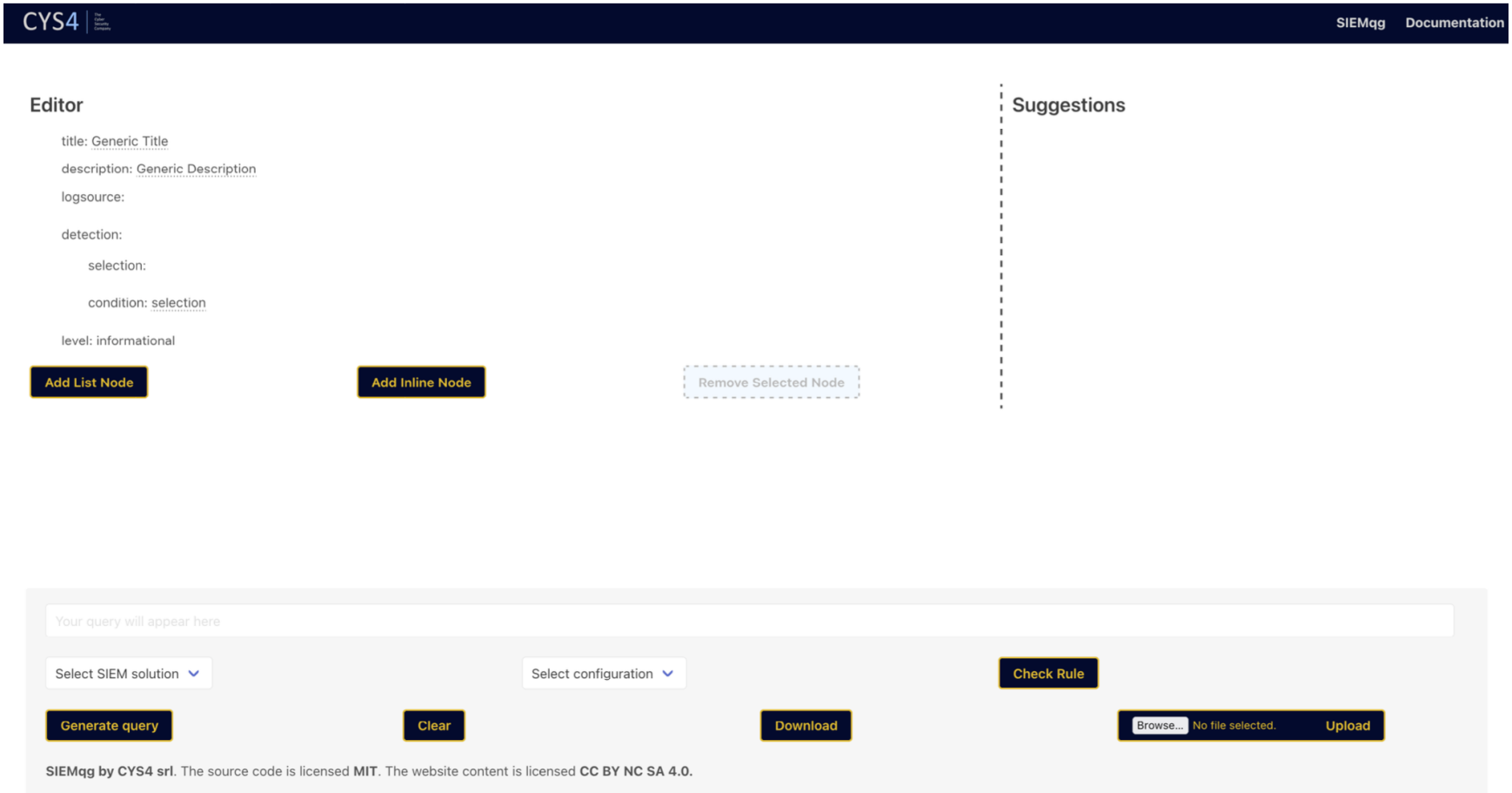
- Implementazione del modulo **io.StringIO** di python per la creazione e gestione degli stream.

```
1...
2 f = io.StringIO(str(sigmafile))
3 parser = SigmaCollectionParser(f, sigmaconfigs, rulefilter, "
      sigmafile")
4 backend.setYmlFileName(str(sigmafile))
5 results = parser.generate(backend)
6...
```

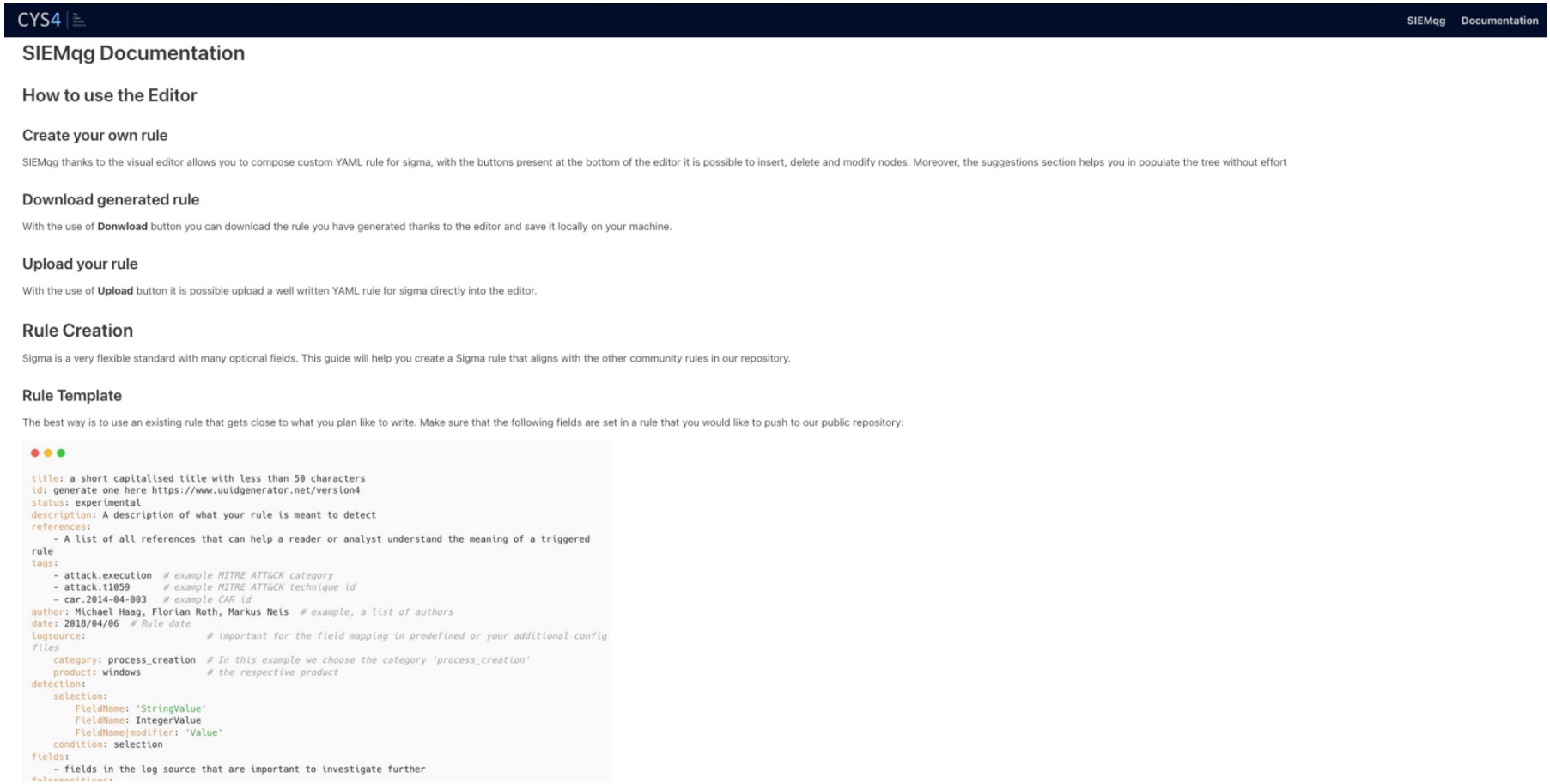
Implementazione: Presentation Layer

- **Interfaccia utente**, pagine web che consentono la comunicazione tra l'utente finale e le funzionalità dei precedenti moduli;
- **Javascript**, linguaggio di scripting orientato agli oggetti e agli eventi che viene utilizzato nella programmazione web lato client per creare controlli sulle pagine o effetti dinamici;
- **BULMA**, framework CSS per design responsive e javascript-free;
- **SASS**, estensione di CSS che aggiunge variabili, funzioni e una migliore struttura.

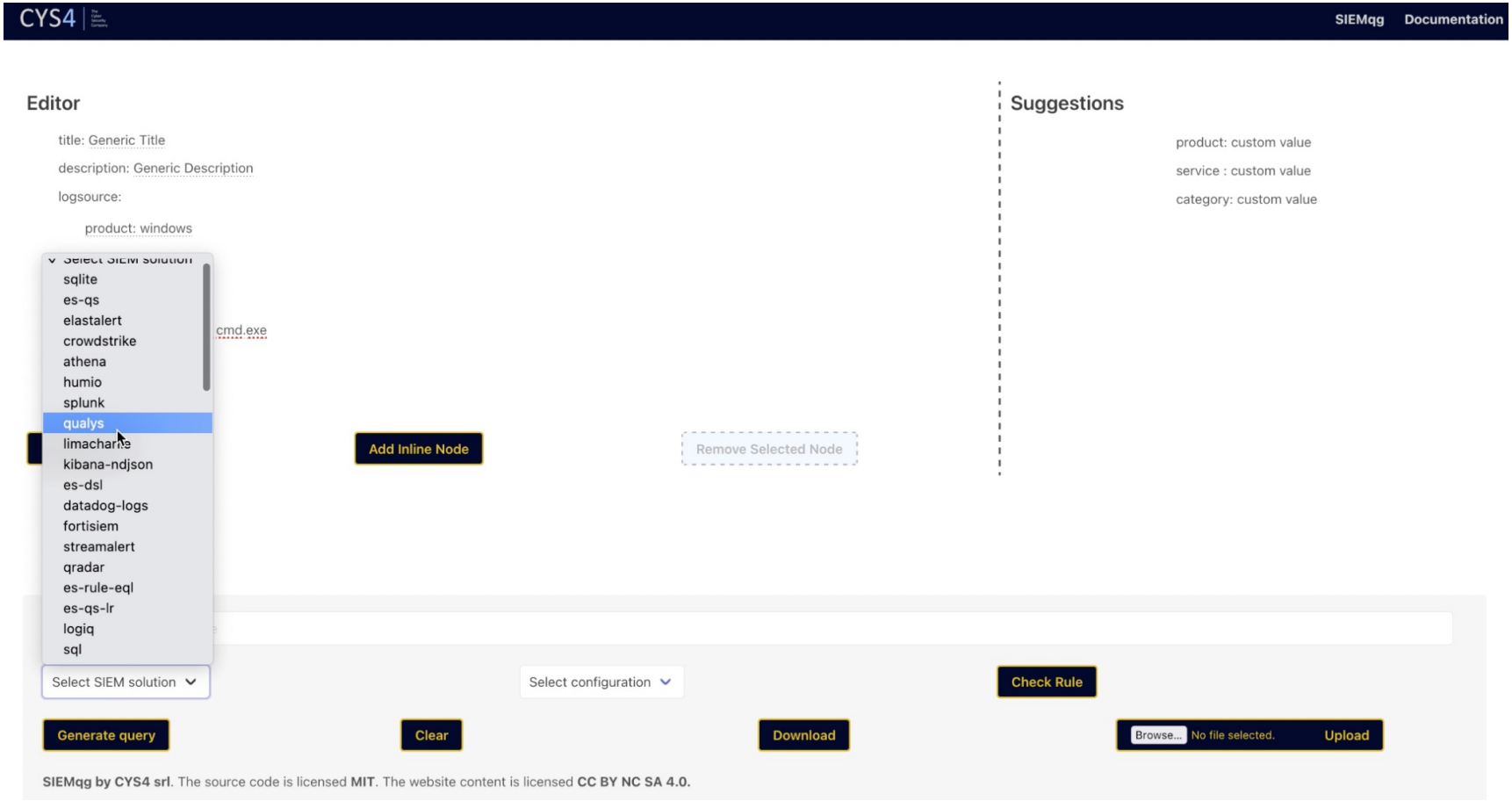
Implementazione: Presentation Layer



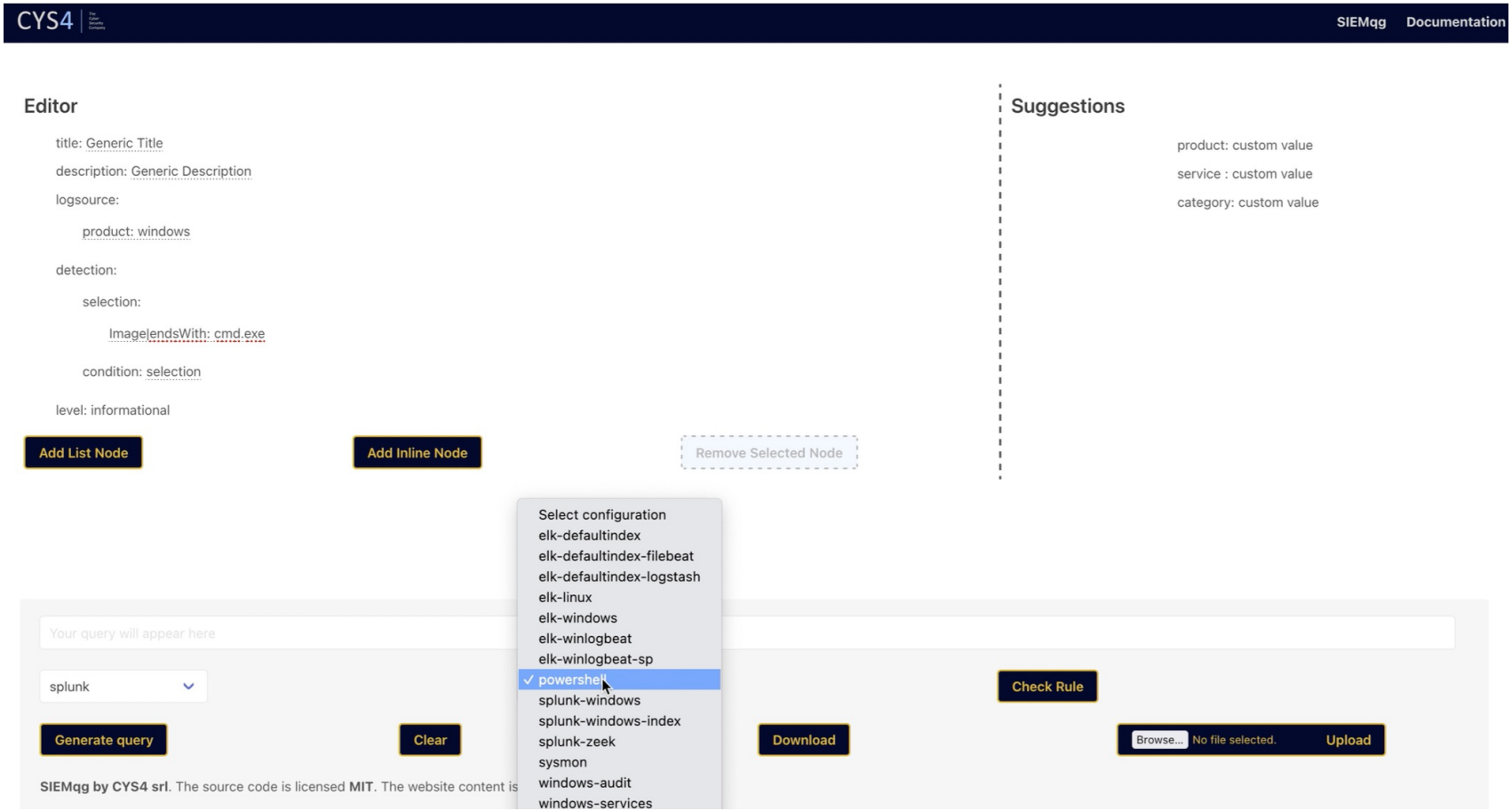
Implementazione: Presentation Layer



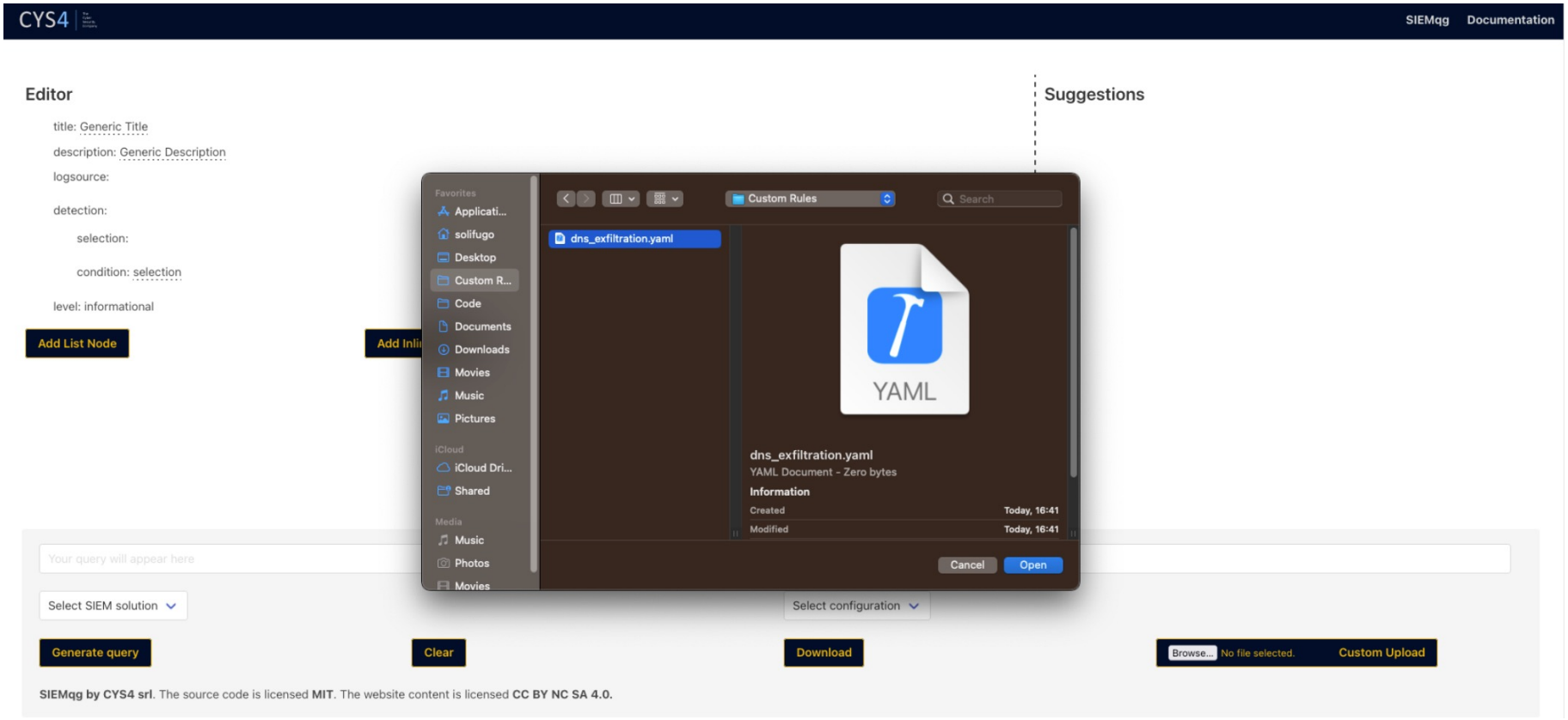
Implementazione: Presentation Layer



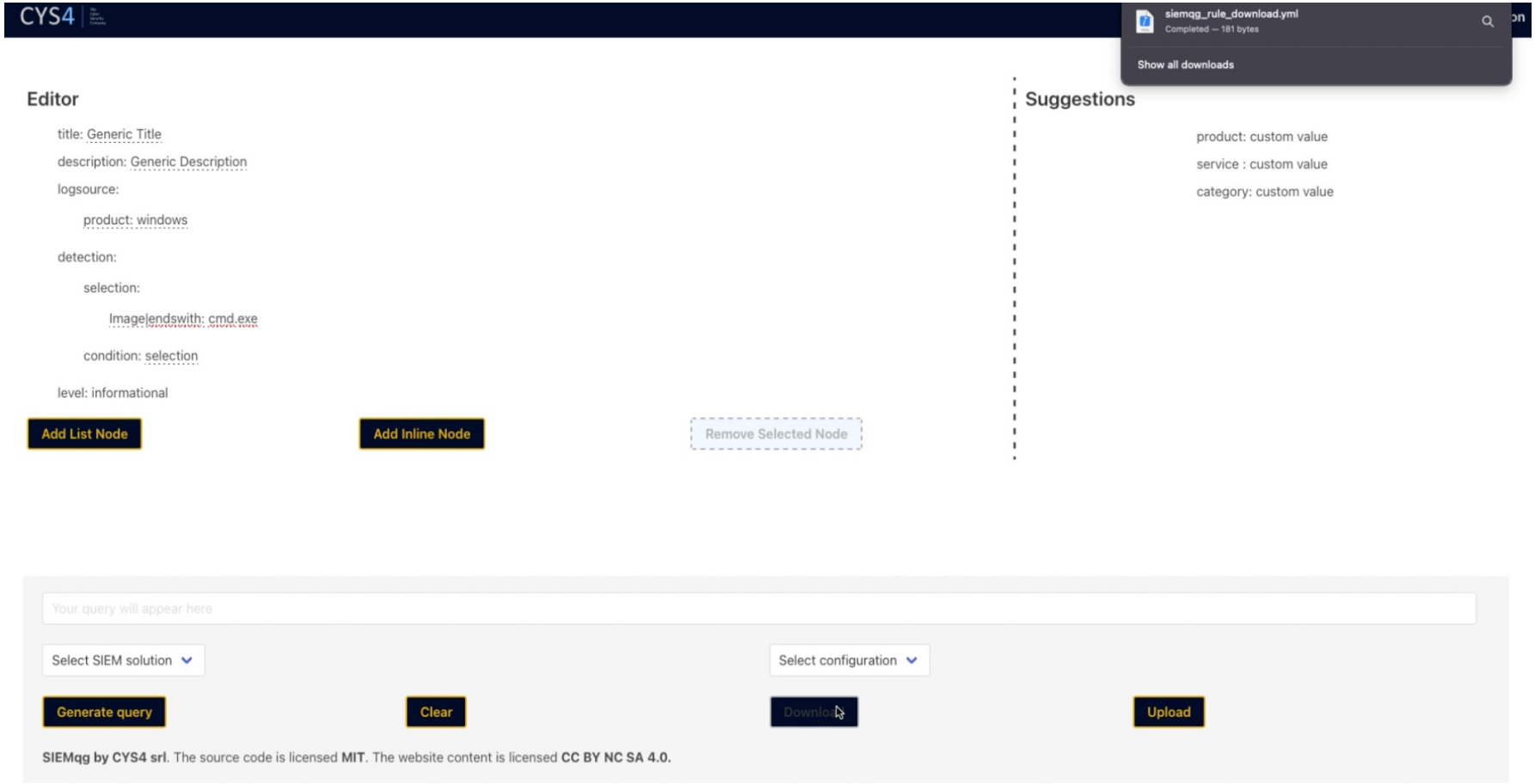
Implementazione: Presentation Layer



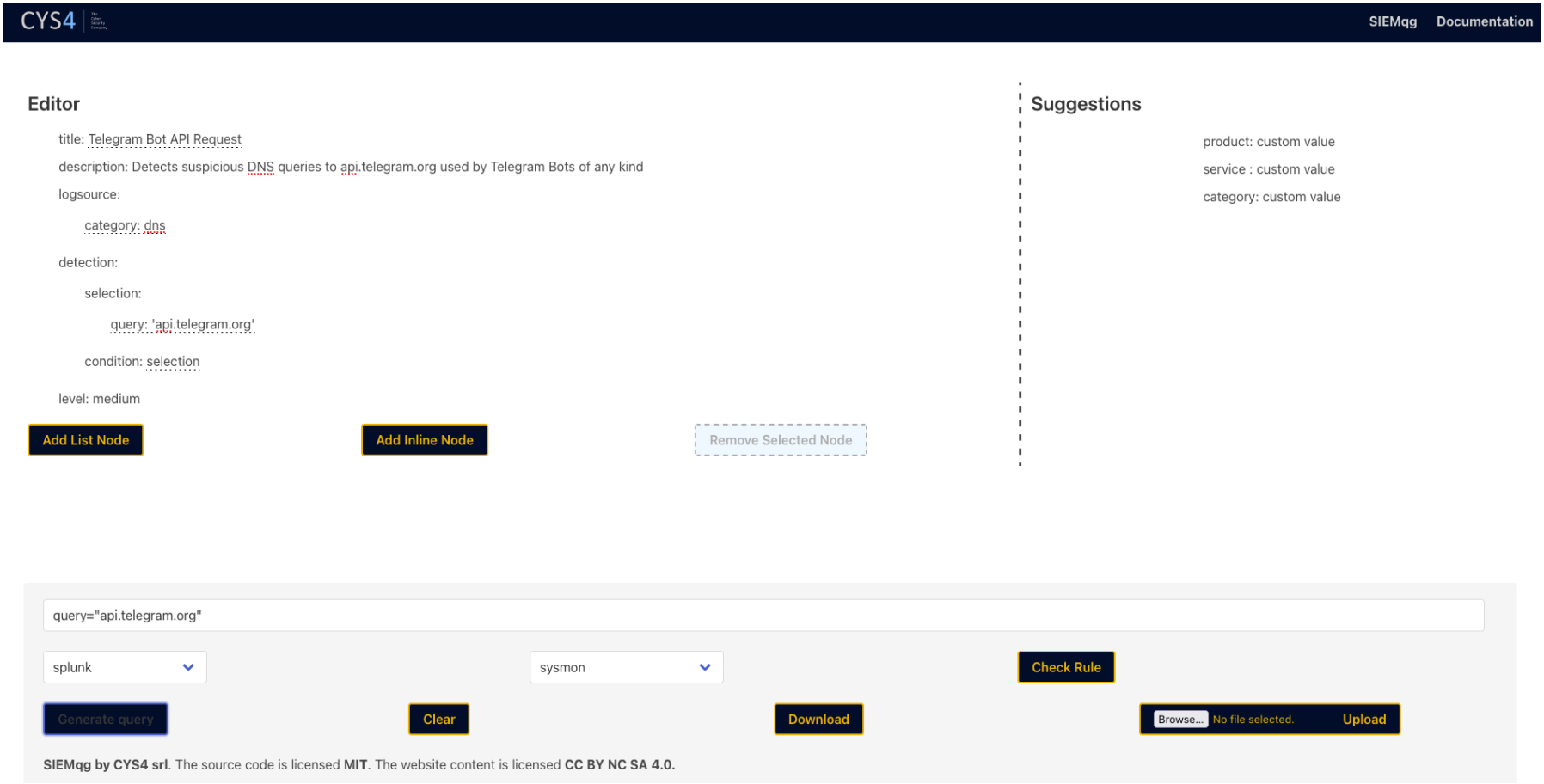
Implementazione: Presentation Layer



Implementazione: Presentation Layer



Implementazione: Presentation Layer



Implementazione: Presentation Layer

```
1  //Send the proper header information along with the request
2  http.setRequestHeader('Content-type', 'application/x-www-form-
       urlencoded');

4  http.onreadystatechange = function() {//Call a function when the
       state changes.
5    if(http.readyState == 4   http.status == 200) {
6      document.getElementById("qresult").value = http.responseText;
7    }
8    else if(http.readyState == 4   http.status == 500){
9      alert("The configuration selected is not currently implemented
        correctly in sigma. Please use different configuration...");
10   }
11 }
12 http.send(params);
13 })
```

Implementazione: Dockerfile

```
1  FROM python:3.8.15-bullseye

3  COPY . /sigma_app
4  RUN git clone https://github.com/SigmaHQ/sigma.git
5  RUN mv /sigma/tools/sigma /usr/local/lib/python3.8/site-packages/
6  RUN rm -rf sigma
7  RUN pip install coverage==5.0 yamllint==1.21 elasticsearch==7.6
       elasticsearch-async==6.2 pytest==5.4 colorama setuptools stix2
       attackcti pipfile requests==2.25 urllib3==1.26 progressbar2==3
       .47 pymisp==2.4.123 PyYAML==5.1 ruamel.yaml flask waitress

9  EXPOSE 5954

11 CMD ["python", "/sigma_app/src/sigma_server.py"]
```

Sviluppi futuri

Sviluppi futuri

- Integrazione con YARA;
- Autocompletamento delle regole in fase di creazione
- Autocorrezione semantica delle regole create
- Supporto alle API di differenti tecnologie SIEM
- Valutazione di tecnologie server web alternative

Progettazione e sviluppo di una piattaforma Web per la generazione visuale di query per tecnologie SIEM

Facoltà di Ingegneria dell'informazione, informatica e statistica presso UNITELMA
Corso di Laurea in Informatica

Alessio Giovannini

Responsabile: Angelo Monti

Co-responsabile: Alessio Dalla Piazza

SAPIENZA
UNIVERSITÀ DI ROMA

UnitelmaSapienza
Università degli Studi di Roma