Traccia A. Pannelli informativi

Si supponga di avere accesso ad API REST pubbliche che espongano in formato JSON informazioni sulla programmazione di una sala cinematografica.

a) Descrivere la struttura di una possibile soluzione che permetta di mostrare i prossimi eventi in programma su alcuni schermi visibili al pubblico. Il sistema deve prevedere l'utilizzo di solo software open-source, e può prevedere l'utilizzo di server fisici, macchine virtuali, dispositivi embedded, o simili.

   Nella descrizione, dettagliare in particolare la scelta di eventuali database, server web, linguaggi di programmazione e framework, la loro interazione.

b) Per gestire gli schermi, una possibile soluzione, suggerita da un noto servizio d'intelligenza artificiale, prevede l'utilizzo di questo script `bash`. Si commenti il funzionamento dello script, suggerendo eventuali miglioramenti e mettendo in evidenza eventuali limiti e problematiche.

```
1  #!/bin/bash
2  URL="https://cinema.com/programma/"
3
4  firefox --kiosk "$URL" &
5  PID=$!
6
7  CURRENT_HASH=$(curl -s "$URL" | md5sum | awk '{print $1}')
8  while true; do
9  NEW_HASH=$(curl -s "$URL" | md5sum | awk '{print $1}')
10 if [ "$NEW_HASH" != "$CURRENT_HASH" ]; then
11   xdotool search --onlyvisible --class "Firefox" key 'ctrl+r'
12   CURRENT_HASH="$NEW_HASH"
13 fi
14 sleep 5
15 done
16 trap "kill $PID" EXIT
17 wait
```

c) Si supponga, nel contesto della domanda precedente, che la pagina web che rende disponibile il programma (`https://cinema.com/programma`) non risulti accessibile dal dispositivo per un periodo di circa 20 minuti. Che problematiche ci dobbiamo aspettare? Si propongano eventuali soluzioni per mitigarle.

Nota: le soluzioni proposte non devono necessariamente utilizzare ogni possibile risorsa suggerita. In generale, soluzioni più semplici e robuste saranno valutate positivamente.

Traccia B. Questionari per gli studenti

In tutte le classi della regione Toscana si vuole far compilare un questionario a fini statistici agli studenti per determinare le lingue parlate in famiglia.

A questo scopo, si ha a disposizione la possibilità d'inviare un'email con contenuto personalizzato a un docente per ogni classe. Nella raccolta dati, pur mantenendo l'anonimato, si vuole essere in grado di ricostruire a quale classe appartiene ogni questionario compilato.

Si può supporre che in ogni classe ogni studente abbia a disposizione uno smartphone, e che ogni classe sia dotata di una lavagna multimediale (LIM).

a) Descrivere la struttura di una possibile soluzione che implementi il sistema che soddisfi i requisiti. Questo deve prevedere l'utilizzo di solo software open-source, e può prevedere l'utilizzo di server fisici, macchine virtuali, dispositivi embedded o SBC, o simili.

   Nella descrizione, dettagliare in particolare la scelta di eventuali database, server web, linguaggi di programmazione e framework, la loro interazione.

b) Un noto servizio d'intelligenza artificiale ha proposto di usare come database un server SQL, con il seguente schema:

```
CREATE TABLE classes (
    class_id SERIAL PRIMARY KEY,
    class_name VARCHAR(50) NOT NULL
);
CREATE TABLE students (
    student_id SERIAL PRIMARY KEY,
    class_id INT REFERENCES classes(class_id),
    anonymous_code VARCHAR(20) UNIQUE NOT NULL,
    PRIMARY KEY (class_id, anonymous_code)
);
CREATE TABLE family_languages (
    family_language_id SERIAL PRIMARY KEY,
    student_id INT REFERENCES students(student_id),
    language VARCHAR(50) NOT NULL
);
CREATE TABLE surveys (
    survey_id SERIAL PRIMARY KEY,
    student_id INT REFERENCES students(student_id),
    submission_timestamp TIMESTAMP DEFAULT CURRENT_TIMESTAMP
);
```

   Si commenti la struttura proposta e si discutano eventuali miglioramenti o modifiche.

c) Si scrivano delle query SQL che permettano di:

   - contare il numero di studenti e di classi che hanno attualmente risposto al questionario;
   - determinare il numero medio di studenti per classe che hanno risposto.

Nota: le soluzioni proposte non devono necessariamente utilizzare ogni possibile risorsa suggerita. In generale, soluzioni più semplici e robuste saranno valutate positivamente. Nella scrittura del codice SQL, la scelta di un particolare dialetto è irrilevante: si può utilizzare eventualmente un qualunque dialetto SQL con cui si abbia maggiore familiarità.

Traccia C. Newsletter per gli studenti

Si consideri il problema di gestire una mailing list a cui devono essere iscritti tutti gli studenti dei corsi di Laurea in Matematica. L'ateneo pisano mette a disposizione delle API che permettono di avere la lista corrente degli studenti iscritti.

a) Si discuta come configuare o sviluppare un server per la gestione di questa mailing-list e come mantenere sincronizzata la lista degli iscritti tramite le API dell'ateneo. Il server si deve occupare di gestire le iscrizioni e spedire la posta elettronica.

b) Un noto servizio di intelligenza artificiale propone di utilizzare Mailman 3 come server per le mailing list, e il seguente codice Python 3 per effettuare periodicamente la sincronizzazione:

```python
import requests

# Configurazioni
MAILMAN_API_URL = "http://mailman.example.com/3.1"
MAILMAN_USERNAME = "admin_username"
MAILMAN_PASSWORD = "admin_password"
LIST_NAME = "math_students"

# Funzione per ottenere la lista degli iscritti da Mailman 3
def get_mailman_subscribers():
    url = f"{MAILMAN_API_URL}/lists/{LIST_NAME}/roster/"
    response = requests.get(url, auth=(MAILMAN_USERNAME, MAILMAN_PASSWORD))
    if response.status_code == 200:
        return set(member['subscriber']['address'] for member in response.json()['members'])
    else:
        print(f"Errore nell'ottenere la lista degli iscritti: {response.status_code}")
        return set()

# Funzione per ottenere la lista degli studenti da Ateneo API
def get_ateneo_students():
    response = requests.get("https://ateneo-api.example.com/students",
        auth=("ateneo_username", "ateneo_password")
    )
    if response.status_code == 200:
        return set(response.json())
    else:
        print(f"Errore nell'ottenere la lista degli studenti: {response.status_code}")
        return set()

# Funzione per sincronizzare gli iscritti tra Mailman 3 e Ateneo
def sync_subscribers():
    # To be implemented
    pass

if __name__ == "__main__":
    sync_subscribers()
```

Si completi il codice sopra proponendo una possibile implementazione di `sync_subscribers`. A tal scopo, si supponga che Mailman fornisca le seguenti API per iscrivere o rimuovere membri delle mailing list:

- `POST /3.1/lists/math_student/members/address@example.com`
  Questo endpoint iscrive `address@example.com` alla mailing list `math_student`. Nel caso l'indirizzo sia già presente viene inserito due volte (e dunque riceverebbe due copie dei messaggi, situazione che si vuole evitare).

- `DELETE /3.1/lists/math_student/members/address@example.com`
  Questo endpoint rimuove `address@example.com` dalla mailing list `math_student`.

c) Si discuta come rendere robusto lo script a eventuali errori di rete durante le richieste HTTP, ad esempio causate da momentanea indisponibilità dell'accesso alla rete Internet.

Nota: le soluzioni proposte non devono necessariamente utilizzare ogni possibile risorsa suggerita. In generale, soluzioni più semplici e robuste saranno valutate positivamente.