Prova Esame di Laboratorio

La prova consiste dovrà essere svolta a partire dal progetto Flask di base contenuto nel file **.zip di Moodle**. La consegna richiederà di creare un blog dove è possibile inserire dei post su tre categorie di argomenti: Tecnologia, Lifestyle e Viaggi.

Per risolvere i seguenti esercizi si richiede che il codice lato backend venga inserito in app.py, e le nuove pagine HTML sotto la cartella *templates.*

Il progetto contiene già il template di base da estendere nei nuovi template e un file *style.css.*

Lato backend sono già implementate alcune funzionalità che possono essere utilizzate negli esercizi. È inoltre già presente la route */react* per renderizzare il template index\_react.html, su cui sviluppare la SPA con React.

Il codice aggiunto deve essere commentato con *ESERCIZIO #<numero esercizio>*

A partire da questo progetto si chiede:

1. Immagine che contiene testo, schermata, Carattere, linea

   Descrizione generata automaticamente(3pt) Impostare la route */index* per renderizzare il template “index.html”. All’interno del template, inserire un’altra section in mezzo alle due già presenti, la quale dovrà contenere una scritta in grande “Cosa troverai nel nostro Blog” e sotto dovrà visualizzare dinamicamente una card per ogni categoria di post. Si prenda l’immagine sotto come esempio.
2. (3pt) Creare una route /*blog* per renderizzare un nuovo template all’interno del quale vengono elencati i post esistenti, in una lista di riquadri centrata orizzontalmente. Per ogni post, devono essere messi in evidenza il titolo in grande, l’autore, la data e la categoria. In caso di nessun post esistente, inserire la scritta “*Nessun post trovato”*.
3. (5pt) Implementare la route esistente /*new\_post*. Questa route accetta i metodi “GET” e “POST”:

* Il metodo GET dovrà renderizzare un template con un form per inserire i dati di un nuovo post: titolo, autore, categoria e contenuto.
* Il metodo POST dovrà prendere i dati da form e creare un nuovo post e salvandolo all’interno del file blog.csv. Tra i dati del post, dovranno essere aggiunti la data di creazione del post e un identificatore generato automaticamente dal formato “P001”, “P002”, “P003”, “P004”, ecc.

1. (2t) Creare una API con route */api/posts* per recuperare tutti i post presenti nel file blog.csv.
2. (5pt) Creare una API con route */api/comments/<post\_id>* per aggiungere un commento al post con identificare “post\_id”. Deve restituire anche tutti i commenti relativi a quel post. Tutti i commenti devono essere inseriti in un file *comments.csv* e per ogni commento devono essere memorizzati i seguenti dati:

* Identificatore del commento generato automaticamente (in formato “C001”, “C002”, “C003”).
* Identificatore del post a cui fa riferimento il commento.
* Autore del commento.
* E-mail dell’autore.
* Testo del commento.
* Data del commento (generata automaticamente).

Testare la API da riga di comando con la seguente istruzione:

*curl -X POST http://localhost:5000/comments/P003 -H "Content-Type: application/json" -d "{\"author\": \"Mika\", \"email\": \"\", \"text\": \"Bello!\"}"*

1. (3pt) Crea una API con route */api/post/<post\_id>* per ottenere il post identificato da “post\_id” e tutti i commenti relativi a questo post.
2. (5pt) Lato React, creare un route */react/posts* associata a una componente React *BlogList* che visualizzi tutti i post presenti in maniera analoga al punto 2. Ogni post dovrà essere cliccabile e indirizzare verso la route */react/posts/:id*, definita nel prossimo punto.
3. (7 pt) Creare una componente React *PostComponent* corrispondente alla route */react/posts/:id* per mostrare il contenuto del post e i suoi commenti.

In fondo ai commenti, creare un form per permettere di inserire un nuovo commento relativo al post visualizzato.

**Per consegnare la prova di laboratorio**

* Creare un’unica cartella compressa (ZIP, o RAR), denominata Cognome\_Nome\_Matricola.estensione (sostituire Cognome, Nome e Matricola con i propri dati e sostituire estensione con l'estensione utilizzata, es. zip o rar)
* La cartella compressa deve contenere le cartelle iniziali (*data, static, templates*), e i file richiesti nel testo dell'esame posizionati nelle cartelle apposite.
* Caricare su MOODLE il file zip