

ISTITUTO TECNICO INDUSTRIALE STATALE "G. VALLAURI"

Via S. Michele, 68 – 12045 Fossano

Indirizzo **INFORMATICA**

Classe **V INF B**

Chuck Norris Facts

Si vuole realizzare una applicazione per la gestione dei Chuck Norris Facts

All'avvio l'applicazione client invia una richiesta GET **/api/categories** a cui il server risponde con un vettore contenente l'elenco completo delle categorie da costruire dinamicamente sulla base del contenuto del file facts.json- Chi non dovesse riuscire può usare l'apposita variabile interna categories.

- Ricevute le categorie, il client le visualizza all'interno del listbox di sinistra. Quindi invia una richiesta GET **/api/facts** passando come parametro il nome della prima categoria ricevuta, a cui il server risponde con un vettore contenente tutti i **facts** relativi alla categoria richiesta.
- Il client visualizza tutti i **facts** ricevuti, in ordine di score decrescente, e facendo uso della seguente struttura:
`<input type="checkbox" value="factID"> factValue
`

The screenshot shows a web application interface for Chuck Norris Facts. It consists of three main panels:

- Left Panel (choose a category):** A dropdown menu with 'career' selected. Below it is a list of categories: career, money, explicit, history, celebrity, dev, fashion, food, movie, music, political, religion, science, sport, animal, travel.
- Middle Panel (rate your favorite facts):** A list of facts with checkboxes for rating. The first fact is checked. Below the list is an 'invia' button.
- Right Panel (add a fact in the selected category):** A text input field for adding a new fact, with an 'add' button below it.

In corrispondenza della selezione di una nuova voce nel ListBox di sinistra il client invia una nuova richiesta al server ed aggiorna la finestra centrale contenente l'elenco dei **facts**.

All'interno della finestra centrale l'utente può selezionare i facts preferiti.

In corrispondenza del pulsante INVIA il client invia una richiesta POST **/api/rate** passando come parametro un vettore contenente gli **id** delle voci selezionate, ad esempio `{ids : [id1, id3, id4]}`

Il server in corrispondenza della richiesta :

- incrementa di 1 gli scores di tutte le voci ricevute
- salva l'intera tabella dati su file
- risponde con un semplice messaggio di "OK"

La sezione di destra consente di inserire un nuovo **fact** che verrà inviato al server tramite una richiesta POST **/api/add** a cui verranno passati come parametri:

- categoria = categoria impostata nel listbox di sinistra
- value = contenuto del nuovo fact

In corrispondenza della richiesta il server imposta i rimanenti campi nel modo seguente:

- data di creazione e data di aggiornamento = data corrente
- url e icon_url = valori fissi definiti all'inizio del file server.js
- score = 0
- **id** generato casualmente tramite una sequenza casuale di 22 caratteri base64.

Controllare che l'id non sia già esistente ed in caso contrario rigenerarlo.

Al termine il server aggiunge il fact nella tabella, salva la tabella su file e risponde con un semplice messaggio di "OK", in corrispondenza del quale il client richiede al server l'elenco aggiornato di tutti i **facts** relativi alla categoria corrente.