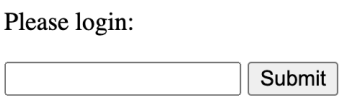
***Programmazione web – compito del 27 giugno 2022***

Un sito gestisce vari utenti.

1. Il nome del progetto deve essere il vostro cognome.
2. Vi sono due utenti normali preregistrati (pippo e pluto) ed un utente speciale preregistrato (admin). Questi tre utenti sono presenti al lancio dell'applicazione.

3. Accedendo al sito, utenti non riconosciuti passano da una semplice pagina di login, nella quale devono inserire il proprio nome (per minimizzare il codice da scrivere, non serve dare anche una password).

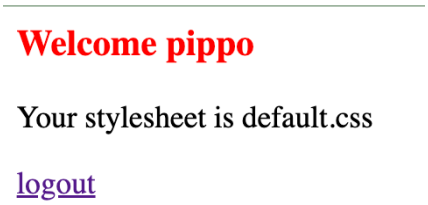



4. Se il nome dato non è valido, si torna alla pagina di login, aggiungendovi un messaggio di utente non riconosciuto.

5. Gli utenti riconosciuti entrano invece direttamente in una pagina che dipende dal tipo di utente: admin entra in una pagina di amministrazione, gli utenti normali accedono a una pagina utente (lo stesso avviene dopo un login di successo).

6. Gli utenti riconosciuti restano tali per un tempo T definito come parametro della webapp. Tale parametro è modificabile senza ricompilare il codice, e diventa attivo al prossimo riavvio della webapp. Quando un utente cessa di essere riconosciuto, in console viene stampato un messaggio "user X is now disconnected".

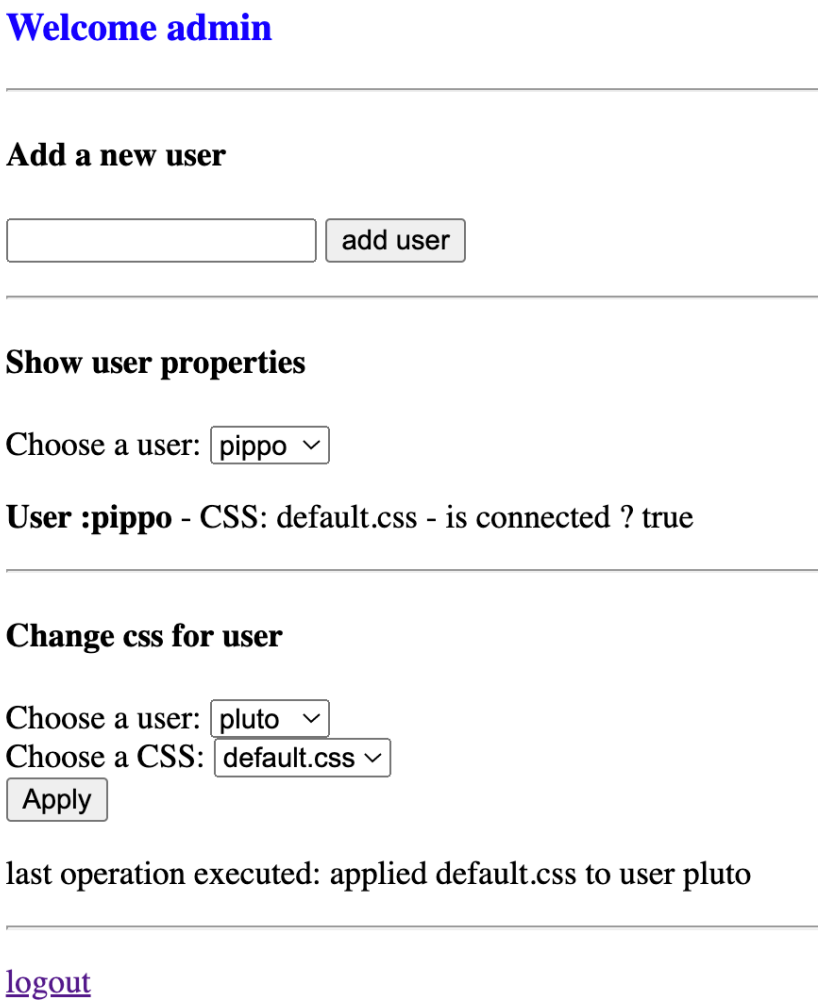
7. La pagina per l'utente normale presenta solamente:
   - il nome dell'utente
   - il nome dello stylesheet applicato
   - un link di logout che rende l'utente non più riconosciuto e riporta alla pagina di login




8. Vi sono tre css predefiniti: default.css, s1.css, s2.css (possono essere molto semplici: la cosa fondamentale è che siano riconoscibili grazie a variazioni nel rendering della pagina, come ad esempio cambiando il colore dello header).

9. La pagina di admin è mostrata sotto. Tramite di essa è possibile:
   - Aggiungere nuovi utenti (solo per l'esecuzione corrente della webapp, non vengono salvati permanentemente)

- Scegliere da una select uno specifico utente, e vederne le proprietà: username, css, stato dell'utente (connesso o disconnesso) – senza che la pagina sia ricaricata quando si sceglie un utente.
- Applicare ad un utente uno dei tre css previsti. Inizialmente gli utenti sono associati a default.css. Utenti eventualmente collegati vedranno la variazione al primo reload della loro pagina utente.
- Scollegarsi




10. Nessun dato è persistente: al riavvio della web-app tutti gli utenti esistenti ed i loro dati sono cancellati.
11. Si protegga la webapp dagli accessi non autorizzati.

**Consiglio**: leggere tutto prima di iniziare la codifica, e pianificare il lavoro da fare. Il progetto viene valutato solo se gira. Le funzionalità possono essere implementate in un ordine diverso da quello qui descritto. Quando completate una funzionalità, salvate una copia del progetto, così se poi doveste avere problemi avete sempre qualcosa di consegnabile. Vanno usate le buone prassi di programmazione. Si consiglia di usare in parallelo diversi browser (Chrome, Firefox…).

**Nota**: può essere utile di tanto in tanto ripulire la cache del browser. (Clear browsing data).

**Suggerimento:** per monitorare server side la terminazione di una sessione termina si può usare una classe che implementi HttpSessionListener. Tramite l'evento ottenuto come parametro implementando il metodo sessionDestroyed(HttpSessionEvent event) si può recuperare l'oggetto Session che rappresenta la Session che sta per essere distrutta, e da esso risalire al ServletContext e ai dati.
La classe di monitoraggio così creata va registrata nel web.xml:

```
<listener>
  <listener-class>
    fully qualified Path Name of the class implementing HttpSessionListener
  </listener-class>
</listener>
```