Tecnologie Informatiche per il Web
Prof. Piero Fraternali
I° Prova in Itinere – 19 Aprile 2015

Si consideri l'applicazione Web ***LOFI*** (Local Deals Finder). La applicazione si compone di un modulo server-side e di un modulo client-side. ***LOFI*** è geo-enabled, e invia notifiche relative a promozioni offerte da ristoranti e negozi nelle vicinanze dell'utente. ***LOFI*** sfrutta metodi di crowdsourcing per aggiornare continuamente il proprio database con nuove promozioni. L'interfaccia utente di LOFI è mostrata in Figura. I client (Web browser) interrogano il server ogni 5 secondi per recuperare, e mostrare all'utente, una lista aggiornata delle promozioni correntemente disponibili. Il client filtra le promozioni ricevute, mostrando all'utente solo quelle che rispettano alcune specifiche condizioni.

OVERVIEW DETAILS RESERVE

La applicazione Web si compone delle seguenti tre pagine:

- Overview: la pagina mostra la lista di tutte le promozioni offerte da negozi e ristoranti *che sono correntemente aperti al pubblico*. La lista è automaticamente recuperata dal server ogni 5 secondi. Ogni offerta ha un *timestamp* di validità, utilizzato per nascondere le offerte correntemente scadute. Il client filtra **localmente** le offerte di ristorante e negozi che si trovano a distanza **X** dalla posizione corrente dell'utente, e che hanno prezzo massimo **Y**. Ogni utente di LOFI può impostare **X** e **Y** individualmente.
- Details: quando un utente è interessato in una particolare offerta, un click sul titolo della offerta attiva la navigazione verso la pagina Details. La pagina contiene informazioni relative alla offerta selezionata, come l'indirizzo del negozio/ristorante, i suoi orari di apertura, e la descrizione del prodotto in offerta. La pagina Details contiene un link "*Make the deal*" che consente all'utente di accedere alla pagina di prenotazione del prodotto/servizio (pagina Reserve).
- Reserve: in questa pagina, l'utente fornisce informazioni necessarie per l'acquisto del prodotto, per esempio: il nome/cognome dell'utente, il suo indirizzo, e una stima del tempo richiesto per il recupero del prodotto in offerta. Una volta inviate al server, le informazioni sono inoltrate al rispettivo negozio/ ristorante.

Esercizio 1 (15 pt.)

Realizzare, utilizzando HTML, Javascript e CSS, il componente client-side della pagina Overview. La pagina deve permettere di specificare filtri per il prezzo massimo **Y** e la distanza massima **X**, e di filtrare le offerte disponibili in funzione dei valori **X** e **Y**.

La pagina deve essere progettata per una visualizzazione su dispositivo mobile[1] (e.g. 375 x 667). Si assuma la presenza sul client di una struttura dati (e.g. un oggetto Javascript pre-definito) contenente le informazioni relative a tutte le offerte correntemente nella base di dati. All'apertura della pagina, del codice Javascript analizza la struttura dati, e visualizza solo le offerte che rispettano i valori di **X** e **Y** impostati di default per la pagina. Variazioni dei valori **X** e **Y** da parte dell'utente provocano l'aggiornamento dell'elenco di offerte correntemente mostrate all'utente.

Si progetti:

a) Il layout della pagina in HTML e CSS (5pt)
b) La struttura dati contente i dati delle offerte (2pt)
c) La funzione responsabile della visualizzazione delle offerte che rispettano i filtri **X** e **Y** (5pt)
d) La funzione che varia le offerte visualizzate a seguito del cambiamento dei valori **X** e **Y** (3pt)

Esercizio 2 (17 pt.)

Realizzare, utilizzando una delle architetture Java server-side mostrate durante il corso (Java Servlet – o JSP), il componente server-side della applicazione *LOFI*.

Si progetti:

a) (3pt) Lo *schema* della base di dati a supporto del sito Web, utilizzando la notazione ER; *si consiglia* anche la creazione della relativa base di dati, al fine di semplificare lo sviluppo della gestione dati.
b) (7pt) La (o le) Java Servlet/JSP necessarie per fruire delle 3 pagine della applicazione LOFI.
c) (5pt) La funzionalità che consentono l'utilizzo della applicazione da parte di più utenti contemporaneamente, gestendo in modo persistente le preferenze relative ai parametri X e Y dell'esercizio precedente (per semplicità, si assuma che nei punti 2a e 2b l'utente sia noto a priori)
d) (2pt - opzionale) Le funzionalità che consente l'aggiornamento periodico della lista di offerte disponibili nella applicazione. La modifica può richiedere la modifica del codice client-side precedentemente sviluppato.

NOTA: il design della base di date è obbligatorio, ma la realizzazione del database non lo è. Tra le tecnologie RDBMS disponibili, si suggerisce l'uso di PostgreSQL o di MySQL Server.

[1] http://viewportsizes.com

Istruzioni di Consegna

Per la consegna è necessario riportare tutto il codice sorgente in un unico documento (pdf o doc), completo di eventuali vostre annotazioni utili a completare le specifiche o a consentire una migliore comprensione del vostro elaborato al docente.

Inserite in un **unico** archivio zip (**cognome-nome-matricola.zip**) il file sopra descritto **e** la cartella contenente tutti i file della applicazione, in modo che essa sia direttamente eseguibile in un application server (e.g. Tomcat). Allegare anche il dump della base di dati realizzata a supporto della applicazione.

Caricate l'archivio ZIP nella directory di consegna "**Prima Prova In Itinere A.A. 2015-2016"**. Avete la possibilità di aggiornare l'archivio quante volte vorrete: il docente scaricherà e correggerà I compiti solo dopo la data di consegna.

La data di consegna è fissata per Lunedi 2 Maggio 2016 alle ore 23.30 (**improrogabile**).