

# Università degli Studi di Torino

## Tecnologie Web (6 cfu) – MFN0634

### Consegna 06 – PHP e MySQL

#### Testo del compito

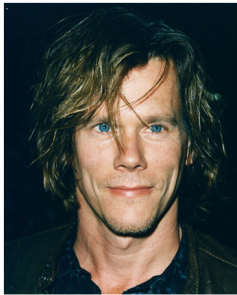
Lo scopo di questo esercizio è di fare pratica con i **database relazionali** e soprattutto su come **connettersi ad essi da PHP**. Inoltre, con questo esercizio, avrete modo di abbandonare lo stile di programmazione server-side in cui il PHP genera HTML e si alterna ad esso nelle pagine. In questo esercizio, invece, chiediamo che le parti server-side e client-side siano rigidamente separate e mediate da **Ajax**.

MyMdb

My Movie Database

### The One Degree of Kevin Bacon

Type in an actor's name to see if he/she was ever in a movie with Kevin Bacon!



All movies  
 first name  last name

Movies with Kevin Bacon  
 first name  last name

W3C HTML5

W3C CSS3

MyMdb

My Movie Database

### Results for Kevin Spacey

Films with Kevin Spacey and Kevin Bacon

#	Title	Year
1	2000 Blockbuster Entertainment Awards	2000
2	AFI's 100 Years... 100 Stars	1999

All movies  
 first name  last name

Movies with Kevin Bacon  
 first name  last name

W3C HTML5

W3C CSS3

#### Il contesto

La teoria dei Sei Gradi di Separazione è stata introdotta dallo psicologo sociale Stanley Milgram negli anni 60, ed è basata sull'idea che, presa una coppia di individui (per i sociologi: "attori") qualsiasi in una rete sociale, allora ci sarà un cammino minimo medio di lunghezza 6 tra di loro. Questa teoria è stata (inconsapevolmente o meno) applicata per un periodo di tempo ad un gioco diffusi negli Stati Uniti che tendeva a dimostrare che Kevin Bacon fosse l'attore più "importante" nella rete sociale delle collaborazioni cinematografiche e che egli fosse distante da ogni altro attore di quella rete con cammini di lunghezza non superiore a 6. Questa "diceria" ha dei fondamenti scientifici e matematici

seri (ad esempio usando il cosiddetto modello "small-world") e parte del principio dei 6 gradi di separazione è effettivamente validabile su qualsiasi rete sociale, ma è anche vero che Kevin Bacon non sia più centrale di altri attori nella rete sociale: è molto probabile che esista un cammino molto breve per qualsiasi coppia di attori. Il sito proposto questa settimana si ispira a questa idea un po' bizzarra ed un po' vera che ruota attorno a Kevin Bacon.

## Il sito

L'obiettivo di questo esercizio di laboratorio è quello di sviluppare un sito, chiamato **myMDB**, usando tecnologie **HTML/CSS/PHP/JS/JQuery**, che replica in parte le funzionalità del popolare IMDB. Questo sito serve a mostrare i film in cui Kevin Bacon ha recitato un ruolo in collaborazione di un altro attore scelto dall'utente. Se preferite, potreste anche scegliere un altro attore come ipotetico centro della rete delle collaborazioni cinematografiche, l'importante è che questo attore sia presente nel database fornito come materiale dell'esercizio e che abbia un certo numero di connessioni con altri attori, ovvero abbia recitato in molti film). Un ulteriore scopo dell'esercitazione è quello di impraticarvi con lo stile di programmazione "unobstrusive", per cui dovrete cercare di mantenere il più possibile separati i contenuti (**HTML**) dallo stile (**CSS**) e dal comportamento (**PHP/JS**). In particolare, sul comportamento ci aspettiamo che la comunicazione dinamica da server a client avvenga tramite la mediazione di **Ajax**.

I seguenti file li potete trovare su moodle:

- [top.html](#), contenente qualche riga HTML che dovrebbe essere inclusa all'inizio di ogni altra pagina
- [top.html](#), come [top.html](#) contiene qualche riga HTML che dovrebbe essere inclusa all'inizio di ogni body delle varie pagine.
- [bottom.html](#), contenente qualche riga HTML che dovrebbe essere inclusa alla fine di ogni altra pagina. In questo file trovate anche i fieldset che includono i vari tag input che servono a chiedere all'utente i dati sull'attore da cercare.
- [index.php](#), la pagina principale che serve ad accogliere l'utente

Se, quindi, visualizzate il risultato della home [index.php](#), sul vostro browser dovrete vedere qualcosa di simile alla figura in alto a sinistra, ovvero una pagina che contiene due form che possono essere usati dall'utente per digitare il nome di un attore (diverso da Kevin Bacon). L'utente può cercare un film dove questo secondo attore è apparso, oppure ogni film dove sia l'attore cercato che Kevin Bacon siano apparsi.

Questi che seguono sono i file di cui sicuramente avrete bisogno:

- [common.php](#), contiene codice di servizio comune che può essere usato da tutte le varie pagine del sito
- [bacon.css](#), lo stile CSS da usare nelle varie pagine
- [bacon.js](#), il codice JQuery/JS che vi serve per visualizzare i risultati

Inoltre, dovrete gestire i risultati delle ricerche. Per fare questo potreste creare degli script che producano codice HTML contenenti direttamente la lista dei risultati. Questi script (ad esempio un "search-all.php" ed un "search-kevin.php") potrebbero essere richiamati direttamente grazie ad un attributo "action" da inserire in due tag "form" che includerebbero i fieldset dentro il file "bottom.html". Questi script potrebbero produrre i risultati della ricerca di tutti i film, dato un determinato attore ("search-all.php"), oppure tutti i risultati della ricerca di tutti i film dove appaiono sia il dato attore che Kevin Bacon ("search-kevin.php"). **Attenzione:** questo potreste farlo, ma non è quello che vi chiediamo di fare in questo compito. Per questo motivo non troverete il tag "form" dove vi sono i fieldset, proprio per indurvi a non usare 'action' per

richiamare lo script server-side. Infatti, vi chiediamo di usare ajax per fare comunicare dei dati JSON (o XML) tra client e server.

## La pagina principale, [index.php](#)

La pagina principale, [index.php](#), consente all'utente di cercare gli attori. Questo file è già messo a disposizione su moodle; potete modificarlo come volete, o potete lasciarlo così com'è. I moduli presenti nella pagina contengono due campi testo che permettono all'utente di cercare l'attore tramite la coppia nome/cognome.

- `firstname` per il nome dell'attore
- `lastname` per il cognome dell'attore

## Script di ricerca dei film, [getMovieList.php](#)

Questo script serve ad implementare la funzione di "search". Esegue delle query nel database imdb (già fornito su moodle precedentemente) per estrarre i film di un dato attore. Eseguite le query nel database usando le librerie PDO di PHP così come abbiamo visto a lezione. Connettetevi al database usando le vostre credenziali.

Questo script deve essere implementato come fosse un Web Service: può essere chiamato con comando GET o POST (scegliete uno dei due metodi, ma non entrambi - per semplificare il debug potreste usare GET, almeno in un primo momento).

Potete ad esempio richiamare questo script indicando nome e cognome dell'attore da cercare ed un parametro opzionale per indicare quale delle due funzioni di ricerca volete richiamare.

Ad esempio, potreste richiamare lo script in questo modo:

[getMovieList.php?firstname=Tom&lastname=Cruise](#)

Per avere la lista dei film in cui ha recitato Tom Cruise insieme a Kevin

The screenshot shows the MyMDb website interface. At the top, there's a header with the MyMDb logo and the text "My Movie Database". Below the header, the main content area displays "Results for Tom Cruise". This is followed by a table titled "All Films" with columns for "#", "Title", and "Year". The table lists 62 entries, including movies like "Mission: Impossible III", "War of the Worlds", "2004 MTV Movie Awards", "61st Annual Golden Globe Awards, The", "76th Annual Academy Awards, The", "Collateral", "2003 ABC World Stunt Awards", "E! 101 Most Shocking Moments in Entertainment History", "Last Samurai, The", "Narc: Shooting Up", "Sex at 24 Frames Per Second", "54th Annual Primetime Emmy Awards, The", "74th Annual Academy Awards, The", "Art of Action: Martial Arts in Motion Picture, The", "Austin Powers in Goldmember", "Hitting It Hard", "Minority Report", "Prelude to a Dream", "Road to the Red Carpet", "Shirtless: Hollywood's Sexiest Men", "Space Station 3D", "Who Is Alan Smithee?", "Rank", "73rd Annual Academy Awards, The", "America: A Tribute to Heroes", "Code of Conduct", "Look Inside: The Others, A", "Stanley Kubrick: A Life in Pictures", "Vanilla Sky", "Young Hollywood Awards", "2000 Blockbuster Entertainment Awards", "2000 MTV Movie Awards", "Behind the Mission: The Making of 'M:I-2'", "Mission: Impossible II", "Mission: Impossible", "American Film Institute Salute to Dustin Hoffman, The", "Eyes Wide Shut", "Magnolia", "Warner Bros. 75th Anniversary: No Guts, No Glory", "Jerry Maguire", "Mission: Impossible", "Interview with the Vampire: The Vampire Chronicles", "Firm, The", "Far and Away", "Few Good Men, A", "Time Out: The Truth About HIV, AIDS, and You", "63rd Annual Academy Awards, The", "American Film Institute Salute to Kirk Douglas, The", "Days of Thunder", "61st Annual Academy Awards, The", "Born on the Fourth of July", "Rain Man", "Young Guns", "Color of Money, The", "Top Gun", "Legend", "All the Right Moves", "Losin' It", "Outsiders, The", "Risky Business", "Endless Love", and "Taps".

Below the table, there are two search forms. The first form is labeled "All movies" and has input fields for "first name" and "last name", followed by a "go" button. The second form is labeled "Movies with Kevin Bacon" and has similar input fields and a "go" button.

At the bottom of the page, there are logos for W3C HTML, W3C CSS, and W3C XML.

Bacon, oppure:

[getMovieList.php?firstname=Tom&lastname=Cruise&all=true](#)

Per avere la lista di tutti i film in cui ha recitato Tom Cruise.

Lo script dovrebbe produrre quindi un output JSON (suggerimento: "Content-type: application/json") - o XML, se preferite - contenente i dati dei film corrispondenti al risultato.

Sarà quindi una callback di una connessione ajax opportunamente richiamata al momento della pressione dei vari tasti di ricerca a modificare il DOM della pagina in modo da mostrare i risultati sotto il div con id `#results`. Infatti, i dati estratti dall'oggetto JSON che viene restituito al client dovrebbero venire visualizzati già ordinati per anno in senso decrescente e nel caso di film distribuiti nello stesso anno, questi dovrebbero apparire ordinati per titolo in senso crescente. I dati possono essere visualizzati in HTML tramite `<table>`, considerando tre colonne: un numero d'ordine che parte da 1; il titolo del film; l'anno di distribuzione. Le colonne devono avere delle intestazioni, alle quali applicare un qualche stile, ad esempio il grassetto. Un esempio di risultato è mostrato a dx. Chiaramente queste informazioni devono essere presenti nell'oggetto JSON che è stato inviato dal server, altrimenti il client non ha modo di conoscerle.

## Database e query

Il database ha, tra le altre, le seguenti tabelle (la tabella roles serve a mettere in relazione attori e film):

table	columns
actors	id, first_name, last_name, gender, film_count
movies	id, name, year
roles	actor_id, movie_id, role

Le funzioni di 'search' eseguono le seguenti query. Per qualche query dovreste usare un join su diverse tabelle del db.

**1. query che restituisce la lista intera (all=true):** Dovreste scrivere una query che restituisca la lista di tutti i film di un dato attore che abbiamo nel database. Da questa lista, il vostro script php dovrebbe produrre un oggetto JSON (o XML) da restituire al client. Se l'attore non esiste nel database, non inviate alcuna lista, ma piuttosto un messaggio tipo: "Actor Borat Sagdiyev not found." che andrebbe mostrato nel div di errore che è presente dentro la pagina [index.-php](#). Se invece l'attore è presente nel db, potete assumere che ci sia almeno un film in cui l'attore ha recitato.

*Suggerimento:* Per scrivere la query corretta, avrai bisogno di fare un join almeno tra le tabelle actors, movies e roles. Tenete solo le righe dove le ID delle varie tabelle combaciano e dove l'attore è quello cercato.

**2. query che restituisce la lista di tutti filtrata (all=false o parametro all omesso):** Dovreste scrivere una query che restituisca la lista di tutti i film di un dato attore che abbiamo nel database dove recita anche Kevin Bacon. Anche questi film saranno inviati al client con le modalità descritte precedentemente

ed infine visualizzati in un'altra tabella HTML, con lo stesso stile della prima, In questo caso la query SQL da eseguire è leggermente più complicata e vi consigliamo di scriverla dopo l'altra. Se l'attore non esiste nel db, non mostrare una tabella, ma un messaggio tipo: "Actor Borat Sagdiyev not found.". Se invece l'attore è presente nel db, ma non ha fatto neanche un film con Kevin Bacon, allora scrivete un messaggio tipo: "Borat Sagdiyev wasn't in any films with Kevin Bacon."

*Suggerimento:* dovete eseguire un join su una coppia di attori (quello che l'utente ha scelto e Kevin Bacon), una coppia di ruoli relativi a quegli attori, ed un film che è associato a quei ruoli. Se viene fuori una join tra 5 tabelle e tre condizioni WHERE, non vi spaventate: potrebbe essere la query giusta. Non valutiamo in questa sede la vostra capacità di scrivere codice SQL efficiente. L'importante è che i risultati siano quelli attesi.

**3. entrambi gli script - trova l'ID per il dato attore:** Una cosa che rende questo programma leggermente complicato è il fatto che alcuni attori hanno lo stesso nome. I dati imdb risolvono questa ambiguità dando a questi attori dei nomi leggermente diversi, ad esempio: "Will (I) Smith" vs. "Will (II) Smith". L'utente probabilmente non capirà la differenza tra i due ed è legittimo aspettarsi che digiterà semplicemente "Will Smith", aspettandosi che il programma farà poi la cosa corretta. Ma se il codice cerca esattamente "Will Smith" nel database, non lo troverà!

Per risolvere questo problema, avete bisogno di scrivere una terza query che cerchi il **miglior risultato relativo** al nome dell'attore che è stato digitato. Questa query cerca e restituisce l'ID dell'attore il cui cognome è esattamente quello cercato dall'utente, ed il cui nome di battesimo inizia con quello che è stato digitato. Se esiste più di un attore che corrisponde a questo criterio di ricerca, **considerate soltanto l'attore che ha fatto più film**. In caso di parità, scegliete l'attore con il numero di ID più basso.

Potete capire in quanti film un attore ha recitato usando una serie di join tra tabelle, ma questo potrebbe risultare troppo inefficiente.

Per fare qualche esempio, se scrivete la query correttamente e cercate nel database più grande (imdb) l'attore "Will Smith", vi verrà restituito quello con ID 444807. Per "David Cohen" avrete quello con ID 90749. Per "Elizabeth Taylor" avrete 809516. Se usate il database più piccolo imdb\_small, la ricerca di "Chris Miller" restituirà l'ID 321300.

*Suggerimento:* Non avete bisogno di usare alcuna JOIN qui, perché tutte le informazioni sono contenute nella tabella "actors". Se all'inizio non volete scrivere questa query, potete fissare l'ID di un attore nel codice, o semplicemente scrivere una query che restituisce il primo attore che risponde alla coppia nome/cognome, che sarebbe anche la risposta corretta quando non ci sono ambiguità da risolvere. Il comportamento della pagina non è definito se l'attore che viene cercato è lo stesso Kevin Bacon.

Ricordiamo che gli script possono essere richiamati tramite AJAX e restituiscono la lista in formato JSON o XML (a vostra scelta). Usate JQuery e JS per creare le tabelle contenenti i risultati.

### **Estensione Opzionale (uso di sessioni per login/logout)**

Se vi rimane tempo, potete provare a modificare il vostro sito in modo da prevedere un login/logout.

Questo significa che, ispirandovi anche al caso di studio visto in classe (User Login Session), dovreste:

- modificare il database inserendo una tabella `user` che includa almeno i campi `user.id`, `user.username`, `user.password`, dove andrete a mettere i dati per almeno due/tre utenti fittizi.
- Aggiungere un controllo che, al caricamento della pagina, grazie ad un'operazione ajax verifichi se una sessione è già attiva ed un utente ha già fatto login. Nel caso positivo, `index.php` la pagina mostrerà i campi per cercare l'attore nelle modalità descritte sopra. Nel caso negativo, sarà mostrato un form che chiede all'utente di effettuare il login fornendo username/password. Suggerimento: possibilmente aiutatevi con una pagina `login.php` che contiene questo ultimo form e che viene "richiamata" solo all'occorrenza, sostituendosi ad `index.php`.
- Modificare la gestione dell'evento di caricamento della pagina `index.php` ed aggiungere dei controlli sulla sessione in tutti gli script (come `getMovieList.php`) perché non si possa accedere direttamente ad essi se l'utente non ha fatto prima login. Nel caso in cui un utente accedesse direttamente ad esse senza che una sessione sia partita in modo corretto, allora il browser sarà re-direzionato verso `login.php` mostrando il form che chiede in input i dati di login.
- Modificare la pagina `banner.html` perché mostri, possibilmente in alto a destra, un link alla funzione di logout, che consenta la chiusura corretta della sessione e faccia fare una redirection alla pagina `login.php`. Tale link deve essere mostrato solo a sessione attiva a seguito di un login andato a buon fine e nascosto viceversa.
- Potreste avere bisogno di creare un file `login.js` per effettuare dei controlli ed istanziare delle connessioni ajax dalla pagina `login.php`.

### **Vincoli sulla presentazione (su tutte le pagine)**

Le varie pagine PHP dovranno adottare dei criteri di presentazione precisi, che sono specificati qui. Oltre a questi, qualsiasi altro aspetto legato alla presentazione potete sceglierlo voi, a patto che non entri in conflitto con quanto richiesto esplicitamente. L'intenzione è quella di consentirvi una certa flessibilità in modo da provare la vostra creatività in vista anche del progetto finale, in cui dovrete compiere molte scelte in autonomia; nello stesso tempo vale la pena tornare ad esercitarsi un po' con CSS, per non trascurarlo. Per favore, includete le immagini nel vostro sito tramite dei riferimenti assoluti e non relativi.

- Tutte le pagine devono iniziare con il contenuto presente in `top.html`, `bottom.html` e terminare con quello presente in `bottom.html`, inclusi i riferimenti al "favicon", al logo MyMDb, ai validatori W3C ed i due form che consentono la ricerca dei film. È importante non modificare i nomi dei parametri usati nei fieldset, come ad esempio `first_name`.
- La sezione principale del contenuto della pagina deve essere all'interno di un'area centrata più stretta rispetto alla pagina complessiva. Questa sezione principale dovrebbe avere un colore di background diverso di quello del body dietro di essa, per creare maggiore contrasto. L'esempio in

figura usa un width del 90% e un background bianco, mentre il body ha un background con colore #dad9d4.

- Ogni pagina deve contenere un'intestazione descrittiva di livello 1 (h1) che spiega il contenuto della pagina (Ad esempio: "Results for Kevin Spacey" oppure "The One Degree of Kevin Bacon".)
- Le aree dei banner in alto ed in basso contengono il logo MyMdb e le immagini W3C che dovrebbero avere un colore comune e/o un'immagine di background per creare maggiore contrasto tra loro ed il resto della pagina. L'esempio mostrato nelle figure in alto usa come immagine di background il file *banner-background.png* e testo bianco.
- Il sito dovrebbe avere sia una colorazione che uno schema di font consistenti. Il vostro CSS dovrebbe essere strutturato in modo che sia facile cambiare il colore o i font senza che sia necessario modificare i colori ed i font in ogni singola pagina. Nelle figure di esempio sono stati usati *Verdana / sans-serif* per il testo, nero-su-bianco per l'area principale e centrale e #dad9d4 per la sfumatura di grigio presente nel background.
- Tutto il contenuto dovrebbe poter essere ridimensionato in modo ragionevole, definire il padding ed il margine in modo tale che il contenuto di una parte della pagina non debba sovrapporsi ad altro contenuto presente. Il contenuto dovrebbe essere anche allineato in modo ragionevole per consentire una lettura semplice. Nell'esempio nelle figure in alto, gli elementi sono stati centrati; i form hanno width 24em; molti elementi hanno 1em di padding o margine per le separazioni rispetto agli elementi circostanti.
- Ogni risultato della query dovrebbe essere mostrato in tabelle con 'caption' adeguate che descrivano il contenuto e con intestazioni che descrivono ogni colonna. Le righe di una tabella dovrebbero alternare il colore dello sfondo, creando l'effetto noto come "zebra striping." I bordi dovrebbero essere 'collassati'. Nell'esempio in figura, ogni riga pari ha un background con colore #dad9d4, a partire dalla seconda.

## Consigli per lo sviluppo, deploy e test

Sebbene su moodle troviate sia il database grande (imdb) che quello più piccolo (imdb\_small), vi consigliamo di usare il secondo per sviluppare e di caricare quello più grande solo alla fine, quando siete sicuri che tutto funzioni bene, cambiando opportunamente i parametri di connessione PDO. Questo dovrebbe rendere la prova delle query più veloce. Potete consegnare il compito anche se il db usato è solo imdb\_small, nel caso in cui il caricamento del db completo vi desse problemi.

Catturate le eccezioni generabili dal vostro oggetto PDO. Durante il debugging, stampate (print) le query che avete costruito per essere sicuri di eseguire le istruzioni SQL corrette. Attenzione agli errori frequenti (apici, variabili, sintassi SQL...). Usate la funzione **quote**.

Infine, caricate tutti i file su una cartella **lab06** sotto il vostro progetto **tweb** su gitlab2. Non dimenticate di caricare anche i file [top.html](#), [banner.html](#), [bottom.html](#), [index.php](#) e [bacon.css](#) anche se non li avete modificati. Potrete poi controllare il vostro sito semplicemente usando la URL:

**<http://localhost:8080/tweb/lab06/>**

Valutate anche di usare la configurazione di XDebug per il vostro IDE. In ogni caso, ricordate che gli errori lato server vengono SEMPRE memorizzati nei file di logs (cercate dove il vostro sistema li ha installati...).

## Implementazione e valutazione



Vi ricordiamo che consegnare gli esercizi di laboratorio serve solo ed unicamente a fornirvi un servizio di correzione, per cercare il più possibile di farvi capire cosa sbagliate ed in modo che possiate migliorare. Non assegneremo delle penalità a chi non li consegna, dato che il voto finale sarà dato in base alla correttezza e alla completezza del progetto e della relazione che ne descrive le specifiche (maggiori dettagli verso la fine del corso). Non controlleremo in questa fase se copiate oppure no: se copiate danneggiate solo voi stessi perché non approfitterete del servizio fornito. Un compito perfetto ci fa piacere perché ci fa perdere meno tempo, un compito molto incompleto e pieno di errori eviteremo di correggerlo per evitare di perdere troppo tempo. In ogni caso, anche in questi casi estremi, non ci faremo influenzare in fase di valutazione finale.

L'output HTML per tutte le pagine dovrebbe superare il test di validazione W3C. Ovviamente il test non coinvolge il codice PHP, ma solo l'HTML da esso generato). Non usate tabelle HTML per definire il layout, ma solo per contenere dati ed informazioni (<table> serve per organizzare il contenuto, non per organizzare la presentazione e lo stile). Dato che stiamo usando form HTML, scegliete i controlli corretti ed i loro attributi di conseguenza. Scegliete GET e POST per spedire i dati al server come richiesto nel testo dell'esercizio. Il vostro codice PHP non deve generare **warning** o **errori** (possibilmente neanche **notice**). Minimizzate l'uso di variabili globali (a volte sono necessarie, ma abituatevi ad usare funzioni parametrizzate), usate correttamente l'indentazione e la spaziatura, evitate linee più lunghe di cento caratteri che rendono il codice difficile da leggere. Usate il materiale prodotto durante le settimane precedenti e quello allegato ai vari capitoli del libro studiati finora (il materiale è reperibile su moodle).

Alcune sezioni HTML sono in comune nelle varie pagine, ad esempio quelle presenti nei file [top.html](#) e [bottom.html](#). Includete questi file in modo appropriato nel resto delle vostre pagine usando la funzione PHP include (o require).

La ridondanza rende il codice difficile da leggere e complicato da riutilizzare. Usate funzioni, parametri, return, file esterni da includere, cicli, variabili, etc. per evitare ridondanza. Se avete del codice PHP (ad esempio una funzione che usate in diversi punti del sito) considerate la possibilità di inserirlo in un file [common.php](#) che poi potrà essere incluso nelle altre pagine.

Non filtrate i vostri dati con PHP: usate le giuste query SQL piuttosto. Se fate così, potrete trattare i vostri dati molto più semplicemente ed efficientemente. Ad esempio, è decisamente un esempio di pessimo stile di programmazione recuperare tutti i dati di una tabella dal db e poi eseguire complesse operazioni sull'array ottenuto per cercare le informazioni che corrispondono alla richiesta dell'utente. SQL è un linguaggio potente: usatelo.

Per fare tutto alla perfezione, riducete la quantità di codice PHP nel mezzo del vostro codice HTML. Il codice ridondante può essere inserito in una funzione, da dichiarare dentro il già citato file [common.php](#). Ricordatevi di limitare il più possibile le istruzioni PHP print ed echo: usate piuttosto le espressioni blocco <?= ... ?>, così come abbiamo visto a lezione.

Un altro aspetto importante è legato all'uso dei **commenti** PHP. Ad esempio, all'inizio di ogni file, di ogni funzione e di ogni blocco PHP, ricordatevi di mettere dei commenti descrittivi.

Cercate di strutturare il vostro codice in un modo simile agli esempi visti in classe, facendo particolare riferimento ai diversi casi di studio che abbiamo



analizzato al termine di ogni capitolo. Usate spazi bianchi ed intestazioni in modo appropriato. Non mettete più di un elemento blocco per ogni linea della vostra pagina HTML.

Ricordate infine che il compito dovrà essere consegnato esclusivamente su gitlab2 e che su moodle dovete indicare solo la URL del vostro repo, come ad esempio:

**`https://gitlab2.educ.di.unito.it/ruffo/tweb.git`**

**Se lo avete già fatto non c'è bisogno di rifarlo:** noi faremo semplicemente un aggiornamento del progetto che avremo nel frattempo già clonato. Ricordate che le **scadenze** vengono date per consentirvi di seguire al meglio la correzione generale che verrà pubblicata in seguito. Noi faremo solo un controllo, alla scadenza, se sono state rispettate o meno. In ogni caso, questo non pregiudica la possibilità per le studentesse e gli studenti di ottenere il massimo dei voti all'esame. Inoltre, ricordate di dare l'autorizzazione all'accesso ai Proff. Botta e Ruffo.