



SAPIENZA
UNIVERSITÀ DI ROMA

BRIDGE reloaded: sistema di supporto alla gestione di homework e progetti per corsi universitari

Facoltà di INGEGNERIA DELL'INFORMAZIONE, INFORMATICA E STATISTICA

Dipartimento di INGEGNERIA INFORMATICA, AUTOMATICA E GESTIONALE "ANTONIO RUBERTI"

Corso di laurea in INGEGNERIA DELL'INFORMAZIONE [L-270 - Ordin. 2019] - Sede di LATINA (classe L-8)

Matteo Simonetti

Matricola 1854835

Relatore

Prof. Marco Temperini

Anno Accademico 2021-2022

**BRIDGE reloaded: sistema di supporto alla gestione di homework e progetti
per corsi universitari**

Tesi di laurea. Sapienza Università di Roma

© 2022 Matteo Simonetti. Tutti i diritti riservati

Questa tesi è stata composta con \LaTeX e la classe Sapthesis.

Email dell'autore: simonetti.matteo@gmail.com

*Ai miei genitori,
unica certezza della mia vita,
vi amo immensamente.*

Indice

1	Introduzione	1
1.1	Funzioni	1
1.2	Nuove funzioni	1
2	User Experience	3
2.1	Microcopy	4
2.2	In pratica	5
2.3	Colori	7
3	Tecnologie	9
3.1	Front-End	9
3.2	Back-End	10
3.3	Database	10
3.4	Altro	11
4	Specifica del sistema	13
4.1	Utenti e servizi	13
4.2	Casi d'uso:	16
4.3	Database	24
4.4	Programmazione	30
5	Manuale utente	39
5.1	Studente	39
5.2	Tutor	43
5.3	Docente	48
5.4	Admin	50
6	Sperimentazione	53
6.1	Sperimentazione effettiva	53
6.2	SUS	53
6.3	Risultati	54
7	Esperienza complessiva e conclusioni	57
7.1	Riprogettazione in itinere	57
7.2	Mindset dell'utilizzatore	58
7.3	Uso del vecchio sistema su virtual machine	58
7.4	Trasferimento dati dal vecchio sistema	59

7.5	Gestione del tempo	59
7.6	Commenti al codice	60
7.7	Repository	60
Bibliografia		61

Capitolo 1

Introduzione

BRIDGE (1.1) è un sistema di supporto alla gestione di homework e progetti per corsi universitari. È stato riprogettato e ricostruito da zero, usando tecnologie più moderne, interfacce utente più intuibili e più facili da utilizzare ed aggiungendo diverse funzionalità che consentono un uso del sistema più veloce ed intuitivo, dando origine a BRIDGE reloaded (1.2).

1.1 Funzioni

BRIDGE consente ad uno studente di registrarsi con matricola e password e di usare il profilo creato per gestire i propri homework (da qui in poi chiamati anche esercizi) e progetti. Lo studente può scegliere a quale corso iscriversi (in base a quelli che frequenta), e di conseguenza vedere gli esercizi da svolgere. Ce ne sono alcuni rappresentativi e altri non rappresentativi ed ogni esercizio si riferisce ad un argomento (chiamato anche categoria) del corso. Lo studente deve svolgere un esercizio rappresentativo per ogni argomento. Il docente del relativo corso può vedere gli esercizi svolti dallo studente e confrontarli con quelli degli altri studenti, così da poter individuare eventuali tracce di plagio. Il docente, inoltre, ha la possibilità di assegnare un voto ed un commento per l'esercizio, offrendo eventuali correzioni che lo studente può applicare per migliorare l'esercizio e riconsegnarlo una volta corretto. Quando quest'ultimo risulta senza errori, viene accettato dal docente e lo studente può procedere con gli altri esercizi del corso. Nel momento in cui è stato svolto un esercizio rappresentativo per ogni categoria, il docente può assegnare agli studenti un progetto (da qui in poi chiamato anche tesina) che racchiude tutti i maggiori contenuti del corso ed è diviso in vari stati di avanzamento. Ogni volta che lo studente si sente pronto, può parlare col docente e discutere l'operato fino a quel momento: se il docente dovesse reputarlo sufficiente, farà avanzare lo stato della tesina, e così fino ad arrivare al completamento dell'intero progetto che segnerà, per lo studente, la fine di quel corso all'interno di BRIDGE.

1.2 Nuove funzioni

In BRIDGE reloaded (da qui in avanti solo BRIDGE, in quanto non si parlerà più della vecchia versione) cambiano diverse cose:

- I termini "rappresentativo" e "non rappresentativo" sono stati sostituiti per maggior chiarezza con "obbligatorio" e "facoltativo", sebbene un esercizio obbligatorio non lo sia davvero, come spiegato precedentemente, ma basti svolgere un esercizio obbligatorio per ogni categoria.
- Il docente non deve più controllare ogni corso individualmente per vedere se sono stati consegnati nuovi esercizi: difatti, all'interno della sua home, compare una notifica in rosso se ci sono nuove consegne fra i suoi corsi, specificando anche in quale corso. Lo stesso accade per la scadenza delle tesine: nel caso in cui ad uno studente fosse assegnata una tesina con scadenza di due mesi, una volta trascorsi, il docente avrà una notifica in home se il lavoro non fosse stato portato a termine, ossia se la tesina non fosse arrivata allo stato di avanzamento finale.
- Il docente non deve più scegliere manualmente a quale studente assegnare una tesina, dato che ora il sistema autonomamente propone solo gli studenti che possono accedere alla tesina, vedendo se hanno almeno un esercizio obbligatorio (consegnato ed accettato dal docente) per ogni categoria.
- Il punto precedente porta ad un'altra novità: lo studente può ora auto-asssegnarsi una tesina senza più bisogno di passare per il docente, riducendo i tempi d'attesa. Questa funzione è opzionale e può essere attivata o meno dal docente del corso.
- Se uno studente si comporta in maniera scorretta, o semplicemente ha abbandonato il corso e non è più attivo, il docente può espellerlo dal proprio corso, così da averne una gestione più pulita e limitata ai soli studenti attivi.
- Lo studente nella versione precedente doveva dare un nome specifico al proprio file prima di consegnarlo (affinché il sistema lo riconoscesse poi nel sottosistema di antiplagio). ora ciò non è più necessario, dato che il file può essere consegnato con qualsiasi nome ed il sistema lo rinomina automaticamente in base ai dati forniti dall'utente in registrazione.
- Il docente non deve più prendere i file manualmente per correggere gli esercizi dello studente, grazie all'automatizzazione della gestione dei nomi dei file del punto precedente, il file è scaricabile dal docente con un click dalla pagina di correzione dell'esercizio.
- Ora ogni esercizio può avere più di un commento, e i commenti hanno accanto anche la versione di riferimento (se uno studente consegna ed il docente rifiuta l'esercizio con un commento, poi lo studente riconsegna ed il docente commenta, il secondo commento viene segnato come riferito alla versione due dell'esercizio).

Si può facilmente vedere che la maggior parte delle modifiche effettuate di cui si è parlato fino ad ora sono tutte pensate per rendere più comodo e snello l'utilizzo di BRIDGE, aggiungendo funzioni che ne semplificano la fruibilità per il docente e per lo studente, migliorando la user experience e riducendo il numero di attività che lo studente ed il docente devono svolgere per arrivare allo stesso risultato.

Capitolo 2

User Experience

Uno dei punti su cui è stata prestata maggiore attenzione in fase di sviluppo è stato quello della user experience. Gli utenti target di questo sistema, oltre ai docenti che però sono in minoranza, sono gli studenti, e di conseguenza la user experience deve essere:

- **Immediata:** gli studenti sanno benissimo come orientarsi in un sistema moderno e non hanno bisogno di spiegazioni verbose o che scendano troppo nei dettagli. I servizi sono tutti strutturati al fine di richiedere il minimo delle informazioni necessarie all'utente e per fornirle in maniera chiara, ma senza nessun estro particolare. Lo stile delle pagine infatti rimane costante:
 - Vengono utilizzati sempre gli stessi colori per conferire chiarezza e non confondere l'utente. Lo studente, infatti, utilizza questo sistema non per essere abbagliato da colori sgargianti e variegati, ma per portare a termine una task, ben chiara nella sua mente, nel minor tempo possibile. Sui colori si parlerà in maniera più approfondita in 2.3.
 - Il menù laterale rimane costante per permettere di arrivare in qualunque parte del programma da qualunque parte del sistema ci si trovi, senza dover tornare indietro o tornare ogni volta alla home page.
- **Priva di ridondanza:** se si vuole immediatezza, si deve evitare di sovraccaricare lo studente di informazioni. Questo non è sempre possibile in quanto spesso il quantitativo di informazioni da veicolare ad uno studente non si può riassumere in poche righe, però sicuramente si può evitare di inserire le stesse informazioni più volte in posti diversi. Infatti, se un utente nota che un microcopy¹ del sito contiene un'informazione che già conosce, probabilmente perderà fiducia nei confronti dei microcopy e tenderà ad ignorarli. Se ne parla meglio in 2.1.

¹Tradotto: microtesti. Sono parole o frasi della user interface direttamente legate alle azioni che l'utente compie

2.1 Microcopy

Un libro sui microcopy da cui si è preso spunto per dare al sistema uno stile chiaro e definito è il "Manuale pratico per ux writer, designer e architetti dell'informazione" di Kinneret Yifrah [3].

L'importanza di un microcopy adatto al contesto è spesso sottovalutata dagli sviluppatori web, in quanto è un ambito che appartiene meno all'ingegneria e più alla comunicazione. Se però non se ne tiene conto, si corrono diversi rischi. Nel libro si parla di uno studio riguardante la relazione tra esseri umani e prodotti digitali (il sistema in questo caso) con personalità incoerenti, ossia provvisto di testi che cambiano stile, che sono sbagliati rispetto alla situazione che si presenta all'utente etc. Il risultato dello studio mostrava che gli utenti si relazionavano ad un prodotto digitale contraddittorio come se si comportasse in maniera falsa e bugiarda, o comunque non meritevole della loro fiducia, proprio come ci si comporterebbe con una persona (umana) incoerente. Perdendo la fiducia, gli utenti non si lasciavano più persuadere dai messaggi, smettendo di leggerli con attenzione. Perdevano quindi informazioni necessarie all'utilizzo del sistema, rendendo tutto il processo inefficiente. Ecco il motivo per cui i microcopy sono uno dei fattori chiave per la buona riuscita di una user interface e, di conseguenza, della user experience.

Lo stile scelto per i microcopy è:

- **Leggermente formale:** trattandosi di un sistema dedicato ad un ambito universitario non si poteva scegliere uno stile completamente informale, però nemmeno del tutto formale, dato che avrebbe perso immediatezza. La tendenza all'informalità aiuta lo studente, che si trova magari al primo dei suoi anni di studio, a relazionarsi con un ambiente meno pretenzioso, che rilassa la sua esperienza.
- **Diretto e sintetico:** caratteristica fondamentale per accompagnare l'immediatezza ricercata nella user experience. I testi sono il più corti possibile, rimanendo tuttavia comprensibili. A titolo d'esempio, l'utente, dopo aver fatto login, avrà nella home uno dei testi più lunghi che dovrà leggere, ovvero la spiegazione generale del funzionamento di BRIDGE (homework, tesine, consegne ecc). Per non rendere il testo troppo prolisso, viene specificato per ogni punto la pagina di riferimento per quella funziona. Ognuna di queste pagine contiene al suo interno delle istruzioni più approfondite, questo permette allo studente di leggere il minimo delle informazioni necessarie per usare al meglio il sistema.
- **Preciso e specifico:** gli esercizi e le tesine vanno consegnate secondo un criterio specifico (per esempio, il file consegnato deve avere un'estensione specifica); di conseguenza, le richieste avvengono con testi specifici.

Una caratteristica fondamentale in un buon microcopy sta anche nel saper utilizzare una scrittura conversazionale, ossia non scrivere nulla che non **diresti ad alta voce**. Questo aiuta a rinforzare ulteriormente il rapporto che si crea fra il servizio offerto e l'utente: a renderlo più umano. Tutti questi punti di cui si è discusso, però, devono essere messi in pratica nelle varie forme con cui l'utente può interagire col sistema.

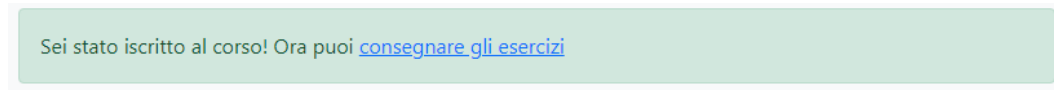
2.2 In pratica

Ci sono vari casi in cui l'utente può rendersi conto di avere davanti una macchina e non un umano, ed è in quei momenti che il sistema deve essere il più umano possibile. Di seguito, i più importanti:

2.2.1 Messaggi d'errore e di conferma

Un messaggio d'errore svolge due funzioni: spiega in modo semplice e chiaro che c'è un problema, e fornisce una soluzione costruttiva utile a riprendere il processo interrotto. Durante l'inserimento di un esercizio, per esempio, se l'utente non legge l'estensione che deve avere il file (indicata da un testo nella pagina) e carica un file con estensione sbagliata, appare un messaggio di errore che ricorda la giusta estensione da usare.

Per i messaggi di conferma bisogna in primis dare certezza. Un messaggio del tipo "operazione riuscita" ha due problemi: il primo è che non è sufficientemente specifico, dato che la specificità non deve essere presente solo negli errori ma anche nelle conferme. Il secondo problema è che nessun umano direbbe una cosa simile ad alta voce, e ciò fa perdere empatia. Inoltre, se possibile, bisogna anche informare l'utente del passo successivo. Per esempio, nella pagina di iscrizione al corso, compare un messaggio di conferma con un link alla consegna degli esercizi per il corso in cui ci si è appena iscritti:



Sei stato iscritto al corso! Ora puoi [consegnare gli esercizi](#)

2.2.2 Stati vuoti

Si definiscono stati vuoti quei momenti in cui non c'è niente da mostrare agli utenti. In genere accadono quando l'utente si è appena registrato ad un servizio e deve ancora attivarne tutte le funzioni. Nel caso di BRIDGE ci sono vari stati vuoti, per esempio quando un utente non può consegnare nessun esercizio perché ancora non è iscritto ad un corso. La gestione degli stati vuoti ottimale prevede una call to action nella pagina vuota che conduce l'utente sul percorso giusto affinché non sia più vuota. Infatti, per esempio, nella pagina di consegna di esercizi ci può essere uno stato vuoto dovuto ad un utente che ancora non si è registrato a nessun corso:

Consegna esercizi (Selezione corso)

Per quale corso vuoi consegnare gli esercizi?


Filtra:

Nome corso	Anno	Docente
Non sei iscritto a nessun corso		

Si può notare che è presente un testo con un link cliccabile che porta alla pagina di iscrizione ai corsi. Questa gestione degli stati vuoti è importante in quanto è un ulteriore aiuto per uno studente che esplora in maniera non ordinata il sistema.


2.2.3 Placeholder

Una cosa da evitare assolutamente è usare dei placeholder ridondanti. Nell'immagine qui sotto vengono presentati un caso di ridondanza (presenza di etichetta² e placeholder insieme) ed un caso corretto (solo etichetta):



First name

Last name



First name

Last name

Anche il caso in cui è presente solo il placeholder è corretto: l'importante è non avere ripetizioni.

Grazie a Bootstrap (3.1.3) questo sistema ha a disposizione dei placeholder che, una volta riempiti di testo, diventano delle etichette. Qui si vedono i campi prima dell'inserimento di un input:

²Testo presente accanto al campo di un form, in genere sopra di esso

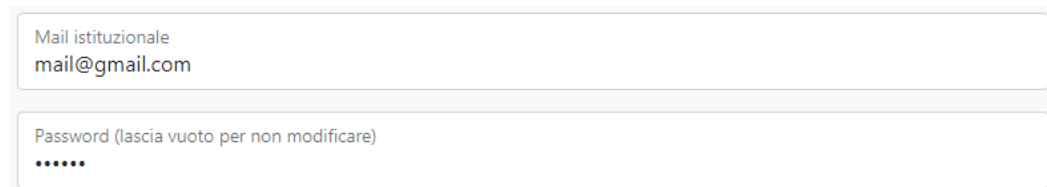


Mail istituzionale

Password (lascia vuoto per non modificare)

This image shows a login form with two input fields. The first field is labeled 'Mail istituzionale' and is empty. The second field is labeled 'Password (lascia vuoto per non modificare)' and is also empty. The form is styled with a light blue border and a white background.

e qui dopo:



Mail istituzionale
mail@gmail.com

Password (lascia vuoto per non modificare)
.....

This image shows the same login form as before, but now the first field contains the text 'mail@gmail.com'. The second field contains seven dots, representing a masked password. The form is styled with a light blue border and a white background.

Se i campi vengono di nuovo svuotati del testo, ritornano nello stato iniziale. Questa è una soluzione che è ad oggi utilizzata anche da Ebay [1] in alcune parti del suo sito, fra cui il login.

2.3 Colori

Esclusi i colori del menù e delle tabelle (che sono i colori base del sito), tutti gli altri colori presenti hanno una loro semantica e vengono usati per pulsanti, messaggi di errore e messaggi di conferma. I colori sono quelli di Bootstrap (3.1.3), ossia:

- Blu: usato per le azioni principali, si usa in genere su dei pulsanti per confermare l'azione.
- Grigio usato per le azioni secondarie, come il blu ma per cose meno importanti.
- Verde si usa come sfondo e (un po' più scuro) come colore del testo nei messaggi di conferma, i quali indicano che l'azione che l'utente stava tentando di svolgere è andata a buon fine. A volte si usa anche per i pulsanti per indicare un'azione "positiva" che va in genere in contrapposizione ad un'altra azione che si può fare e che è percepita come "negativa" (come per esempio la modifica di qualcosa contro l'eliminazione di quella stessa cosa).
- Arancione si usa per dei messaggi di warning, quando qualcosa non va esattamente come doveva andare, però comunque non c'è stato nessun errore. Viene usato sia per lo sfondo che per il colore del testo come nel punto precedente.
- Rosso si usa per dare dei messaggi di errore, oppure per dei pulsanti con azione pericolosa. Per i messaggi di errore viene usato sia per lo sfondo che per il colore del testo come nel punto precedente.

Il punto di forza di usare questo schema di colori è che la semantica è ormai inconsciamente chiara ad ogni utente che ne fa uso, in quanto il sistema di colori di Bootstrap è ad oggi mediamente utilizzato da 1 sito su 4 [2].

Capitolo 3

Tecnologie

Per realizzare questo tipo di progetto si è scelto di creare un'applicazione web, così come era stato scelto anche per la versione precedente di BRIDGE. La comodità dell'applicazione web è data dall'utilizzo senza necessità di download e dalla facile raggiungibilità: trattandosi di un sistema utilizzato da molti utenti, sarà infatti facilmente raggiungibile tramite ricerca Google.

Di seguito l'elenco delle principali tecnologie utilizzate:

3.1 Front-End

Una tecnologia è definibile come "front-end" nel momento in cui essa ha delle funzioni che – traducendo letteralmente – hanno "finalità frontale", ossia hanno il compito di dare all'utente un'interfaccia sulla quale visualizzare informazioni, ed all'interno della quale poter inserire altre informazioni. Le informazioni mostrate nel front-end arrivano dal back-end, e quelle che vengono inserite nel front-end vanno a finire nel back-end (di cui si parla in breve in 3.2).

3.1.1 HTML

L'HyperText Markup Language [5] è un linguaggio di markup. In HTML ci sono dei tag, ogni tag va aperto e chiuso ed al suo interno ci possono essere altri tag oppure del contenuto effettivo. L'HTML è lo scheletro della pagina web, la struttura base. I dati che l'utente può inserire vengono gestiti quasi sempre con dei form, forniti da HTML. HTML5 in particolare supporta nuove funzioni, fra cui la grafica vettoriale che è stata usata per questo progetto.



3.1.2 CSS

CSS (sigla di Cascading Style Sheets, in italiano fogli di stile a cascata) [5], è un linguaggio usato per definire la formattazione di documenti HTML, ossia il modo in cui viene mostrato quello che è contenuto nel documento HTML. La user-experience di cui si è parlato in 2 è resa soprattutto grazie all'uso di CSS, che permette di dare colori, forme, margini e molto altro ai testi forniti da HTML.



3.1.3 Bootstrap

Bootstrap [6] è una raccolta di strumenti liberi per la creazione di siti e applicazioni per il Web. Essa contiene modelli di progettazione basati su HTML e CSS, sia per la tipografia che per le varie componenti dell'interfaccia, come moduli, pulsanti e navigazione. Bootstrap è quindi stato importante per mettere a disposizione dello sviluppatore delle soluzioni che hanno un aspetto moderno, seguendo tutti i punti descritti nel capitolo 2 riguardo i colori dei messaggi di conferma o di errore etc.



Supporta anche il responsive web design. Ciò significa che il layout delle pagine web si regola dinamicamente, tenendo conto delle caratteristiche del dispositivo utilizzato, sia esso desktop, tablet o telefono cellulare. Bootstrap fornisce inoltre delle icone (1500 all'incirca) per rendere più visivo l'utilizzo del sistema.

3.1.4 JavaScript e jQuery

JavaScript [7] è il linguaggio di programmazione lato client per eccellenza, usato per il 98% della programmazione client-side [8], jQuery [9] è una libreria che semplifica notevolmente la selezione, la manipolazione, la gestione degli eventi e l'animazione di elementi DOM in pagine HTML. jQuery è stato fondamentale per tutti gli elementi ad apparizione, per esempio i testi degli esercizi e delle tesine, che sono nascosti (in quanto molto lunghi, occuperebbero tutta la pagina) e sono visibili premendo un pulsante che li rivela con un'animazione morbida e regolare. Un altro utilizzo fondamentale è stato quello del filtro, ossia grazie a jQuery l'utente che si trova davanti ad una tabella, con dati di qualunque tipo, può inserire dalla tastiera un parametro e filtrare i dati che ha davanti, lasciando solo quelli inerenti.



3.2 Back-End

Le informazioni di cui si è parlato in (3.1) vengono quindi lette, modificate, ed in generale gestite nel back-end. Il back-end si occupa anche di inviare i dati al database (3.3).

3.2.1 PHP

In questo progetto è stato usato PHP [10] senza l'uso di nessun framework o libreria. PHP è il linguaggio di programmazione lato server più usato [11]. PHP è un preprocessore di ipertesti, ossia processa le pagine prima di servirle all'utente che le richiede via web, questo permette di avere pagine web dinamiche. PHP è un linguaggio interpretato e l'interprete è un software libero.



3.3 Database

Con base di dati (database) si indica un insieme di dati strutturati memorizzati in un computer.

3.3.1 MariaDB

Come DBMS¹ è stato utilizzato MariaDB [12]. Si tratta di un relational DBMS ed ha la stessa sintassi di MySQL [13]. Questa è stata la maggiore novità rispetto alla vecchia versione di BRIDGE che era priva di DBMS e basava il suo storage dati su XML. Un progetto di queste dimensioni però necessita di un DBMS alle spalle per garantire la totale integrità dei dati, in riferimento alle chiavi primarie (PK), alle chiavi esterne (FK) e a tutto quanto offerto dal DBMS.



Particolare attenzione è stata posta al rendere impossibile da parte di un utente malintenzionato di usufruire di SQL injection. L'SQL injection è una tecnica usata per attaccare applicazioni che gestiscono i propri dati attraverso database relazionali, che sfruttano quindi il linguaggio SQL, come MariaDB. La tecnica consiste nell'inserire delle stringhe di codice SQL nell'input fornito. Se l'input dell'utente non viene sanificato² c'è il rischio che questo codice inserito in input arrivi effettivamente al DBMS e venga eseguito, permettendo a chiunque di eseguire qualunque comando SQL. Ciò si risolve grazie all'estensione mysqli fornita da PHP che prevede i "prepared statement" che impediscono all'utente di eseguire comandi SQL. Ciò, in breve, avviene grazie al fatto che tutto quello che viene inserito dall'utente viene preso come puro argomento e non come possibile comando.

3.4 Altro

Ulteriori tecnologie che non rientrano nelle sezioni precedenti sono le seguenti:

3.4.1 Apache

Apache [14] è un server web libero, è stato scelto in quanto molto usato ed, appunto, libero. Apache realizza le funzioni di trasporto delle informazioni, e in Windows ciò è realizzato con un servizio, mentre in Linux con un demone. Ha vari moduli: quelli principali sono due, ovvero il polling e il core. Quello di polling interroga continuamente le linee logiche da cui possono pervenire messaggi di richiesta da un client. Quello di core riceve le informazioni e le smista negli altri moduli per produrre una risposta al client.



3.4.2 HTTPS

L' HyperText Transfer Protocol (protocollo di trasferimento di un ipertesto) è, appunto, il protocollo usato come principale sistema per la trasmissione d'informazioni sul web e quindi su architetture client-server. La "S" identifica la connessione criptata dal Transport Layer Security (TLS) che ne garantisce l'integrità. Tra il protocollo TCP e HTTP si interpone un livello di crittografia/autenticazione che cripta il messaggio HTTP prima della



¹ Acronimo di database management system

² Dall'inglese input sanitization, ossia la pratica di controllo e sostituzione di caratteri o stringhe che potrebbero iniettare codice malevolo nel sistema

trasmissione e lo decripta una volta arrivato a destinazione. In pratica, viene stabilita una connessione criptata tra client e server tramite lo scambio di certificati che attestano l'identità delle due controparti.

3.4.3 Bash

Essendo il sistema destinato ad essere hostato su un server con Linux, Bash era il miglior modo per interagire con esso. Bash è infatti la shell di linux, ossia l'interprete dei comandi del sistema operativo, attraverso il quale si può avere un livello di interazione alto col sistema operativo. Bash è stato usato per poter fare backup e restore del database attraverso i comandi "mysql" e "mysqldump" forniti da MariaDB.



3.4.4 SHA512

SHA512 va a sostituire MD5, l'algoritmo di hashing del vecchio BRIDGE, ormai non più ritenuto sicuro, in quanto nel 2005 sono stati trovati dei metodi per generare collisioni in un tempo di un minuto [15], cosa che quindi permetteva a chiunque avesse accesso all'hash di fingere di avere la password originale, e quindi di autenticarsi in maniera malintenzionata. SHA512 appartiene alla famiglia SHA, seconda generazione. Esso produce un digest di 512bit ed attualmente non sono noti attacchi a questa generazione, il che lo rende un algoritmo sicuramente più sicuro di MD5.



3.4.5 Graphviz

Graphviz [21] è un software di visualizzazione di grafi open source. Nel progetto infatti era necessario un software in grado di trasformare delle informazioni relative al plagio fra coppie di studenti in un'unica informazione visualizzabile globalmente, quindi un grafo. Graphviz trasforma delle informazioni testuali relative ai nodi e agli archi che uniscono i nodi nell'immagine finale del grafo. L'immagine è rappresentata attraverso un file SVG. Include varie opzioni fra cui font, colori, layout ed altro. Una delle funzionalità più importanti è la costruzione automatica dell'SVG del grafo senza sovrapposizione fra archi o nodi.



3.4.6 sim

Sim [22] cerca somiglianze lessicali in testi scritti in linguaggio umano ed in programmi in C, C++, Java, Pascal, Modula-2, Miranda, Lisp, ed 8086 assembly. È usato per trovare codice duplicato in grandi progetti software, script o documentazioni di progetti. È anche usato per trovare plagio fra gli studenti in progetti educazionali. Le somiglianze sono riportate tramite percentuale, riportando anche i pezzi di testo simile. In questo progetto è stato utilizzato sim per trovare somiglianze nel codice C dei progetti degli studenti. Il funzionamento di sim è basato su un sistema di tokenizzazione del codice per fare in modo, per esempio, che la variazione dei nomi delle variabili non influisca sul risultato.

Capitolo 4

Specifica del sistema

Prima di iniziare con la specifica, è bene fare alcune precisazioni per comprendere al meglio questo sistema. Maggiori dettagli sono forniti nelle tabelle del database nella sezione 4.3.

I "Tipi di corso" sono la generalizzazione di un corso, per esempio un corso può essere "Linguaggi per il Web 2021-2022", e il tipo di corso corrispondente sarà "Linguaggi per il Web". Il tipo di corso contiene tutti i dati generici di un corso, dati che sono comuni a tutti i corsi di quel tipo. I tipi di esercizio sono la generalizzazione degli esercizi, per esempio creare un programma che sommi due numeri interi è un tipo di esercizio; quando poi uno studente svolge quel tipo di esercizio, allora si tratta di un esercizio effettivo. Un tipo di esercizio vale per tutti i corsi appartenenti allo stesso tipo di corso. Idem per tesine e tipi di tesina.

4.1 Utenti e servizi

1. **Login:** tutti gli utenti prima di usare qualunque altro servizio devono passare per il login usando matricola (o username se non sono studenti) e password.

4.1.1 Studente

Ogni volta che si specifica che "si sceglie un corso" si intende che lo studente deve essere iscritto a quel corso.

Si ricorda che se un tipo di esercizio è obbligatorio non significa che lo studente lo deve svolgere obbligatoriamente, ma ne deve svolgere uno per ogni categoria se vuole accedere alle tesine.

2. **Registrazione:** se lo studente entra nella piattaforma per la prima volta, può registrarsi inserendo i suoi dati anagrafici, matricola, password e mail. Fatto ciò, sarà fatto il login automaticamente.
3. **Modifica mail o password:** dalla home lo studente può cambiare mail e password tramite un form.

4. **Iscrizione corso:** lo studente si può iscrivere ad uno qualunque dei corsi a cui non è ancora iscritto (se il corso è attivo, ossia non archiviato) per poter poi consegnare gli esercizi e ricevere l'assegnazione di una tesina.
5. **Dettagli corso:** si possono visualizzare i dettagli più specifici di un corso come il sito del corso, il nome dei tutor e la presenza o meno di tesine.
6. **Consegna esercizi:** per prima cosa si seleziona il corso per cui consegnare l'esercizio, poi si sceglie il tipo di esercizio da una lista, si carica il file dell'estensione richiesta e si preme il pulsante di upload.
7. **Gestione esercizi:** dopo aver selezionato il corso si possono visualizzare gli esercizi caricati, che possono essere caricati di nuovo oppure eliminati.
8. **Lista tesine:** si sceglie un corso e si visualizzano i tipi di tesina di quel corso, col rispettivo testo, e si vede se sono disponibili o già prese da altri studenti.
9. **Auto assegnazione tesine:** dalla pagina delle liste delle tesine si può accedere (se la funzione è attiva per il tipo di corso) alla sezione dove ci si può assegnare autonomamente una tesina, scegliendo tra i tipi di tesina disponibili. Va però sottolineato che in questo modo non la si potrà svolgere in gruppo.
10. **Tesina assegnata:** si sceglie un corso e si vede la tesina assegnata con allegati stato di avanzamento, scadenza, account assegnato e commenti del docente.

4.1.2 Tutor

Ogni volta che si specifica che "si sceglie un corso" si intende che il tutor deve avere assegnato il tipo di corso a cui esso appartiene.

11. **Riepilogo corso:** si sceglie un corso e si possono vedere gli studenti iscritti a quel corso. Per ogni studente si possono vedere quanti esercizi ha consegnato e quanti ne sono stati accettati.
12. **Correggi esercizi (e plagio):** si sceglie un corso e poi un esercizio consegnato da uno studente di quel corso. Si può poi scegliere o di correggerlo, assegnando un voto, un commento ed una valutazione di accettazione o di rifiuto, oppure si può confrontare la percentuale di plagio con un altro studente o più studenti.
13. **Assegna tesine:** si sceglie un corso, si mostrano poi i tipi di tesina già assegnati e quelli ancora liberi. Premendo il pulsante accanto a quelli non assegnati, si apre una pagina dalla quale si possono selezionare gli studenti a cui assegnare le tesine, e gli account da fornire ad ogni studente. Se invece si seleziona un tipo di tesina già assegnato, si possono modificare studenti ed account assegnati.
14. **Stati tesine:** si sceglie un corso, poi una tesina assegnata, poi si presenta una pagina identica al pt.9 ma con i commenti modificabili e la possibilità di far avanzare di stato la tesina.

4.1.3 Docente

Ogni volta che si specifica che "si sceglie un corso" si intende che il docente deve avere assegnato il tipo di corso a cui esso appartiene.

15. **Crea corso:** si può creare un corso che appartenga ad uno dei tipi di corso assegnati dal docente, specificando anno ed istruzioni del corso.
16. **Riepilogo corso:** (già descritta nel punto 11).
17. **Modifica corso:** si possono modificare le istruzioni di un corso creato.
18. **Archivia corso:** si può archiviare uno dei corsi creati o riattivare i corsi già archiviati.
19. **Elimina corso:** si può eliminare definitivamente un corso archiviato.
20. **Crea tipo esercizio:** si sceglie un tipo di corso e si può creare un tipo di esercizio, selezionando categoria (fra quelle del tipo di corso), testo dell'esercizio e specificando se sia obbligatorio o meno.
21. **Gestisci tipi esercizio:** si sceglie un tipo di corso, si sceglie un tipo di esercizio creato e si modificano i campi specificati nel punto precedente.
22. **Correggi esercizi (e plagio):** (già descritta nel punto 12).
23. **Crea tipo tesina:** si sceglie un tipo di corso e si può creare un tipo di tesina, selezionando se sia specifica per un solo corso, testo della tesina, scadenza e titolo.
24. **Gestisci tipi tesina:** si sceglie un tipo di corso, si sceglie un tipo di tesina creata e si modificano i campi specificati nel punto precedente.
25. **Assegna tesine:** (già descritta nel punto 13).
26. **Stati tesine:** (già descritta nel punto 14).
27. **Reset account tesine:** si sceglie un tipo di corso assegnato e si revocano tutti gli account assegnati agli studenti di quel tipo di corso (utile per riutilizzarli per il corso dell'anno successivo).
28. **Corsi frequentati da studenti e disiscrizione:** si possono vedere quali corsi frequentano gli studenti. Il docente può espellere uno studente da un corso, solamente, però, se il corso è uno dei suoi.
29. **Gestisci utenti:** il docente può gestire studenti e tutor. Selezionandone uno può cambiare tutti i dati che erano stati immessi durante la sua registrazione.

4.1.4 Admin

30. **Crea tipologia di corso:** si può creare un tipo di corso, inserendo nome del tipo di corso, studenti massimi per tesina, numero degli stati delle tesine, numero account tesine (fac.), numero account totali (fac.), possibilità di auto assegnazione tesina, docente e tutor assegnati, tipo di voti, software antiplagio, sito del corso, estensione degli esercizi e categorie degli esercizi.
31. **Gestisci tipologia di corso:** si sceglie un tipo corso esistente e si modificano le informazioni del punto precedente.
32. **Archivia corso:** (già descritta nel punto 18) con la differenza che sono mostrati i corsi di tutti i docenti.
33. **Elimina corso:** (già descritta nel punto 19) con la differenza che sono mostrati i corsi di tutti i docenti.
34. **Crea docente:** form di registrazione come pt.2 ma per docenti.
35. **Crea tutor:** form di registrazione come pt.2 ma per tutor.
36. **Corsi frequentati da studenti e disiscrizione:** (già descritta nel punto 28) con la differenza che può disiscrivere chiunque.
37. **Messaggio home studenti:** si può cambiare il messaggio che si trova nella home degli utenti.
38. **Gestisci utenti:** (già descritta nel punto 29). L'unica differenza sta nel poter gestire anche docenti ed altri admin.

4.2 Casi d'uso:

N.B.:

- Per utilizzare i servizi dal numero 4 in poi l'utente deve innanzitutto cliccare nel menù a sinistra sulla voce corrispondente al nome del caso d'uso. Per esempio, lo studente che volesse iscriversi ad un corso dovrà per prima cosa cliccare nel menù a sinistra su "iscrizione corso".
- Per la struttura dei messaggi di conferma e di errore, per la semantica dei colori e per la user experience in generale si rimanda al capitolo 2.
- In quasi tutte le pagine che contengono una lista di oggetti (quindi una tabella) è presente sopra ad essa un input all'interno del quale l'utente può scrivere qualsiasi cosa per filtrare fra gli elementi.
Se, per esempio, lo studente sta cercando il corso con tutor "marco rossi", gli basta scrivere nel campo di ricerca "marco" oppure "rossi" e dalla lista scompaiono tutti i corsi che non hanno "marco" oppure "rossi" fra le informazioni del corso.

- In quasi tutte le pagine che contengono una lista di oggetti (quindi una tabella) è presente un'altra funzione: si può cliccare il titolo di una colonna della lista per ordinare la lista alfabeticamente (in ordine crescente o decrescente dando rispettivamente uno o due click) secondo il campo che è stato cliccato. Se, per esempio, si vogliono mettere i corsi in ordine alfabetico per tutor, basta cliccare il titolo della colonna "tutor".

4.2.1 Funzioni comuni

1. **Login:** si può accedere al sistema fornendo matricola e password (nel caso di un utente che non sia studente si fornisce username e password). Una volta fatto ciò, se i dati esistono nel sistema, l'utente viene loggato e può sfruttare i servizi del sistema a proprio nome.

4.2.2 Studente

2. **Registrazione:** ci si può registrare al sistema inserendo i propri dati anagrafici, matricola e password, una volta fatto ciò l'utente viene reindirizzato direttamente alla sua home (con login già effettuato tramite i dati che ha fornito). Se in fase di registrazione l'utente inserisce una mail o un matricola già usate da qualcun altro, allora il sistema restituisce un errore e chiede all'utente di cambiare mail o matricola (a seconda di quale delle due sia il duplicato).
3. **Modifica mail o password:** nella home lo studente può controllare i dati inseriti in fase di registrazione all'interno di un form, nel quale può modificare solo due campi: mail, che contiene la mail attuale, e password, che risulta vuoto. Per modificare mail l'utente deve cancellare la precedente e inserire la nuova, mentre per modificare la password gli basta scrivere la nuova nel campo vuoto. Lo studente può cambiare entrambi i campi o anche uno solo. Se modificando la mail essa risulta già presente nel sistema, viene generato un errore.
4. **Iscrizione corso:** L'iscrizione ad un corso è necessaria allo studente per poter consegnare esercizi e tesine. Lo studente che si vuole iscrivere ad un corso si trova davanti ad una lista di corsi ai quali si può iscrivere. Ogni elemento della lista presenta il nome del corso, l'anno, il nome e cognome del docente ed un pulsante blu per iscriversi al corso. Una volta premuto il pulsante appare un messaggio che sarà di conferma nel caso in cui l'iscrizione sia andata a buon fine, o di errore in caso di problemi col database. Inoltre, se lo studente volesse dei dettagli su un determinato corso, gli basterà cliccare sul nome del corso e verrà rimandato ad una pagina contenente i dettagli. La struttura della pagina di dettagli è spiegata nel prossimo punto.
5. **Dettagli corso:** Consiste in una lista di tutti i corsi, utile per vedere i dettagli più specifici di ogni corso disponibile per lo studente. Oltre a poter vedere il nome del corso, l'anno ed il nome e cognome del docente (come nel punto precedente) qui si possono trovare anche i nomi dei tutor e si può vedere se

il corso contiene o meno delle tesine, ed infine si può accedere al sito web del corso fornito dal docente, all'interno del quale si possono trovare ulteriori informazioni utili allo studente per il corso.

6. **Consegna esercizi:** in primo luogo si presenta la scelta del corso per il quale si vuole consegnare l'esercizio (e quindi bisogna avere fatto l'iscrizione al corso come visto nel pt.4), con dei dettagli generici per ogni corso; una volta selezionato il corso si accede alla pagina vera e propria di consegna esercizi. In cima c'è una sintetica spiegazione sulla differenza fra esercizi obbligatorio e facoltativi e quanti bisogna farne. Segue una lista dei tipi di esercizio disponibili forniti dal docente per quel corso: si può selezionare un tipo di esercizio con il pallino sulla sinistra e si può vedere il testo premendo sul pulsante "testo" del relativo esercizio. C'è un pulsante "testo" per ogni esercizio, e quindi per ogni riga della lista. In fondo alla pagina c'è una sezione dove caricare il file ed accanto un testo che indica quali estensioni sono accettate per il file da caricare, ed infine il pulsante di consegna. Se il file caricato ha un'estensione corretta, il sistema risponde con un messaggio di conferma e l'esercizio consegnato scompare dalla lista degli esercizi consegnabili. Inoltre, scompaiono anche tutti gli esercizi di tipo "obbligatorio" della stessa categoria, poiché l'utente può consegnare un solo esercizio obbligatorio per categoria. Se lo studente consegna un esercizio obbligatorio di una categoria, ma cambia idea e vuole consegnare un altro esercizio obbligatorio di quella categoria lo può fare eliminando il primo tramite pt.7,
7. **Gestione esercizi:** si sceglie prima un corso (è necessaria l'iscrizione al corso come visto nel pt.4) e poi si accede alla gestione. La struttura della pagina è la stessa del punto precedente, con la differenza che nella lista ci sono gli esercizi consegnati e non i tipi di esercizio consegnabili. Gli esercizi sono selezionabili con il pallino solo se non sono stati accettati dal docente. In fondo alla pagina ci sono due pulsanti, uno per la riconsegna ed uno per l'eliminazione. La lista degli esercizi consegnati presenta inoltre una sezione commenti (fatti dal docente, con indicato accanto al commento la versione dell'esercizio a cui si riferisce il commento stesso) ed una sezione voto, per un eventuale voto assegnato dal docente all'esercizio svolto.

Quando lo studente seleziona la riconsegna di un esercizio, se l'esercizio era stato segnato dal docente come "da rivedere", esso ritorna allo stato "inviato"; se invece era inviato, rimane tale. Se lo studente sceglie l'eliminazione, l'esercizio scompare dalla lista e ricompare nella sezione di consegna esercizi vista nel punto precedente. Anche qui se si carica un file con estensione sbagliata compare un messaggio di errore specifico. La riconsegna dell'esercizio mantiene i commenti del docente, l'eliminazione li perde.
8. **Lista tesine:** in primis si presenta la selezione del corso (è necessaria l'iscrizione al corso come visto nel pt.4); scelto il corso, compare una pagina contenente una lista di tutti i tipi di tesina del corso: per ognuna è specificato se è disponibile (se non è presa da altri studenti) e un pulsante per scoprire il testo. Se il corso lo prevede, inoltre, in fondo alla pagina sarà presente un pulsante di "Auto assegnazione", il quale porterà al prossimo servizio. servizio non compare nel

menù, ma è raggiungibile dal punto precedente. Serve per ottenere una tesina senza doverne fare richiesta al docente, ma permette l'assegnazione solamente a se stessi: per poterla fare in gruppo bisogna comunque rivolgersi al docente. La pagina consiste in una lista di tesine disponibili, ognuna con titolo, testo ed un pallino per selezionarla. In fondo c'è un pulsante per assegnarsela. Fatto ciò, si viene portati alla pagina di "Tesina assegnata" insieme ad un messaggio di avvenuta assegnazione.

9. **Tesina assegnata:** dopo aver selezionato il corso per cui si vuole vedere la tesina (è necessaria l'iscrizione al corso come visto nel pt.4) che è stata assegnata dal docente (o auto assegnata), si presenta una pagina con le informazioni sulla tesina. Se non si ha una tesina assegnata compare solo un pulsante per tornare indietro ed un testo che spiega cosa bisogna fare per farsi assegnare una tesina. Se invece si avesse già la tesina, in cima sono indicati la scadenza della tesina (può anche essere un campo vuoto, poiché non tutte le tesine hanno una scadenza, dato che dipende dalla scelta del docente) e l'eventuale account assegnato. Sotto è presente il testo del tipo di tesina e poi c'è lo stato di avanzamento indicato con una barra. La barra inizialmente è vuota, se poi gli incontri col docente vanno bene, quest'ultimo farà avanzare la barra di un passo alla volta fino a riempirla, indicando che la tesina è completa. Sotto allo stato ci sono i commenti del docente, e per ogni commento è indicato in quale stato è stato fatto e la data ed ora del commento.

4.2.3 Tutor

Per tutte le azioni svolte dal tutor si prevede che egli abbia almeno un corso assegnato (da un admin) sul quale poter agire.

10. **Riepilogo corso:** dopo aver scelto il corso fra quelli disponibili (ossia quelli in cui il tutor è tutor del corso), compare la pagina di riepilogo del corso. Si presenta come una lunga lista che permette al tutor di vedere in ogni riga uno studente che è iscritto al corso, con relativa matricola, mail, dati anagrafici, numero di esercizi consegnati e numero di esercizi accettati. Di default le righe sono ordinate mettendo in cima chi ha più esercizi svolti e poi chi ne ha più accettati.
11. **Correggi esercizi (e plagio):** dopo aver scelto il corso fra quelli disponibili, il tutor si ritrova davanti una lista di tutti gli esercizi consegnati dagli studenti per quel corso. Oltre ai dati dello studente, ogni riga contiene lo stato dell'esercizio, la categoria, la data di consegna (o dell'ultima riconsegna) e indica se è obbligatorio o meno. Lo stato può essere "inviato", "da rivedere", oppure "accettato". Il primo stato indica che lo studente ha inviato l'esercizio ed è da correggere; il secondo ed il terzo, invece, indicano che qualcuno ha già corretto l'esercizio rispettivamente con esito negativo o positivo.
Una volta scelto un esercizio da correggere attraverso il pulsante "correggi" si accede alla correzione di quell'esercizio. Oltre ai dati che erano già visibili durante la selezione dell'esercizio, ora si può vedere anche il testo del tipo

di esercizio e si può scaricare il file che ha caricato l'utente. Inoltre, sotto a questi dati si trova una sezione dove si possono fare commenti. Ci sono tante sezioni per i commenti quanti sono i commenti lasciati fino ad ora, più una sezione vuota per aggiungere un nuovo commento. Sotto alla sezione vuota c'è un checkbox per indicare se si vuole che il nuovo commento sia riservato o meno. Infine ci sono due pulsanti, uno per indicare che l'esercizio è stato accettato e uno per indicare che è da rivedere. Premendo uno dei due pulsanti vengono salvati i commenti vecchi (se sono state effettuate modifiche ad essi) e il commento nuovo (se è stato aggiunto). Se da uno dei commenti vecchi è stato rimosso il testo, il commento viene eliminato.

Se dal menu di selezione invece di premere "correggi" si preme "plagio", si entra in una sezione dove si può confrontare il lavoro dello studente selezionato con gli altri. Ci sono una select e tre pulsanti: selezionando uno studente dalla select ed il pulsante "confronto 1 ad 1" si ottiene dal sistema una pagina con i testi degli esercizi consegnati dai due studenti e l'eventuale percentuale di plagio dell'uno rispetto all'altro. Inoltre, i pezzi di esercizio simile sono evidenziati in grassetto. Se si sceglie il pulsante "confronto con tutti" il lavoro dello studente viene confrontato con gli altri che hanno consegnato quell'esercizio e si genera una foto di un grafo con le percentuali di plagio. Il pulsante "confronto tutti con tutti" fa la stessa cosa, ma il confronto invece di essere 1 ad n è n ad n.

12. **Assegna tesine:** dopo aver scelto il corso, il tutor si ritrova davanti ad una lista contenente i tipi di tesina del corso: se una tesina è già assegnata, quella riga ha come pulsante "Modifica assegnazione", altrimenti c'è "Assegna".

Nel caso dell'assegnazione, si presenta una pagina con tanti campi quanti sono gli studenti massimi per tesina del corso, ed accanto altrettanti campi per gli account. Aprendo i primi campi si hanno come opzioni fra cui scegliere gli studenti che hanno consegnato (ed a cui sono stati accettati) tutti gli esercizi obbligatori che erano da consegnare. Accanto si trovano gli account disponibili (ossia non usati da altri utenti).

Nel caso, invece, della modifica, si trova la stessa pagina dell'assegnazione, ma con i nomi e gli account già scelti in base a chi è stato già assegnato alla tesina. Inoltre, in fondo c'è una sezione per modificare la scadenza della tesina e poi ci sono due pulsanti, uno per modificare l'assegnazione e l'altro per eliminare del tutto l'assegnazione, rimuovendo quella tesina e rendendo il tipo di tesina di nuovo disponibile ad altri utenti del corso.

13. **Stati tesine:** Dopo aver scelto il corso si accede ad una lista delle tesine assegnate: per ogni tesina è indicato il testo e gli studenti partecipanti alla tesina. Una volta selezionata una tesina si accede ad una pagina identica al pt.9, con l'unica differenza che i commenti sono modificabili e aggiungibili e che ci sono due pulsanti, uno per aggiungere il commento ed avanzare di stato, e l'altro per aggiungere il commento senza avanzare di stato.

4.2.4 Docente

Per tutte le azioni svolte dal docente si prevede che egli abbia almeno un corso assegnato (da un admin) sul quale poter agire.

14. **Crea corso:** si può creare un corso scegliendo a quale tipo di corso appartiene, l'anno del corso e le istruzioni specifiche per il corso. Le istruzioni sono quelle che compariranno allo studente quando andrà a consegnare gli esercizi. Creare il corso serve per permettere agli studenti di consegnare esercizi e tesine. Se si crea un corso con tipo di corso ed anno già esistenti appare un errore.
15. **Riepilogo corso:** (già descritta nel punto 10).
16. **Modifica corso:** si seleziona un corso fra quelli creati precedentemente e si possono modificare le istruzioni per gli studenti.
17. **Archivia corso:** si possono vedere tutti i corsi attivi (non archiviati) all'interno di una lista, accanto ad ogni corso c'è un pulsante per archivarlo. Un corso archiviato non è più raggiungibile da nessun utente del sistema, l'unico modo per renderlo di nuovo utilizzabile è usare la sezione di riattivazione del corso presente in fondo a questa stessa pagina. Questa seconda sezione presenta una lista con tutti i corsi archiviati ed accanto ad ognuno un pulsante per riattivarlo.
18. **Elimina corso:** si presenta una pagina con una lista contenente tutti i corsi archiviati, infatti un corso prima di essere eliminato deve essere archiviato (per evitare eliminazioni accidentali). Una volta eliminato, il corso ed i suoi contenuti non sono più recuperabili.
19. **Crea tipo esercizio:** si sceglie per prima cosa un tipo di corso fra quelli che il docente ha a lui assegnati. Fatto ciò ci sono tre campi da riempire: categoria, testo, obbligatorio. La categoria contiene un menù a tendina con varie opzioni fra cui scegliere, le opzioni dipendono dalle categorie definite nella creazione del tipo di corso; il testo è un campo testuale e si sceglie se rendere il tipo di esercizio obbligatorio o meno con una spunta.
20. **Gestisci tipi esercizio:** si sceglie un tipo di corso, si apre poi una pagina con tutti i tipi di esercizio di quel tipo di corso, e per ogni tipo di esercizio sono presenti il testo, la categoria e se è obbligatorio o facoltativo. Inoltre, c'è un pulsante per andare nella sezione di modifica. Aperta la pagina di modifica è possibile modificare tutti i parametri descritti nel punto precedente e poi premere il pulsante "Modifica" per confermare le modifiche. Se invece si vuole eliminare il tipo di esercizio basta premere il pulsante "Elimina" senza apportare modifiche ai campi.
21. **Correggi esercizi (e plagio):** (già descritta nel punto 11).
22. **Crea tipo tesina:** si sceglie per prima cosa un tipo di corso fra quelli che il docente ha a lui assegnati. Fatto ciò, ci sono quattro campi da riempire: titolo, corso specifico, testo, scadenza. Il corso specifico indica se la tesina è solo per

un corso invece che per tutti i corsi del tipo di corso scelto. La scadenza indica dopo quanti mesi la tesina scadrà una volta assegnata ad uno studente. Titolo e testo sono campi testuali.

23. **Gestisci tipi tesina:** si sceglie un tipo di corso, si apre poi una pagina con tutti i tipi di tesina di quel tipo di corso; per ogni tipo di tesina sono presenti titolo, corso specifico, testo e scadenza, ed inoltre c'è un pulsante per andare nella sezione di modifica di quel tipo di tesina. Aperta la pagina di modifica è possibile modificare tutti i parametri descritti nel punto precedente e poi premere il pulsante "Modifica" per confermare le modifiche. Se invece si vuole eliminare il tipo di tesina basta premere il pulsante "Elimina" senza apportare modifiche ai campi.
24. **Assegna tesine:** (già descritta nel punto 12).
25. **Stati tesine:** (già descritta nel punto 13).
26. **Reset account tesine:** si presenta una pagina con una select da cui scegliere fra i tipi di corso assegnati al docente. Una volta scelto uno e premuto il pulsante di reset vengono revocati tutti gli account assegnati agli studenti di quel tipo di corso. Se per esempio sta per iniziare un nuovo anno accademico e si vogliono revocare tutti gli account assegnati agli studenti degli anni precedenti per riutilizzarli per l'anno nuovo, lo si può fare da qui. Se nessun account era stato assegnato, non viene modificato nulla.
27. **Corsi frequentati da studenti e disiscrizione:** si presenta una lista formata per ogni riga da nome, cognome e matricola di uno studente ed il corso a cui sono iscritti. Se il corso appartiene al docente c'è anche un pulsante di disiscrizione per espellere lo studente dal corso. Una volta premuto il pulsante vengono eliminati tutti gli esercizi, tesine, commenti ed ogni cosa che abbia a che fare con quel corso e quello studente.
28. **Gestisci utenti:** Il docente può gestire solo tutor e studenti. La prima cosa che appare è una lista di tutti gli studenti e tutor registrati, con matricola, nome, cognome, permesso e pulsante per modificare l'utente. Dopo aver selezionato un utente si apre una pagina dove si possono modificare nome, cognome, username (o matricola), permesso, mail, password. Si può modificare uno qualsiasi dei campi o più di uno. Il campo password è vuoto. Se il campo viene riempito la password viene cambiata con quanto inserito, altrimenti non viene modificata. Una volta fatte tutte le modifiche si preme il pulsante "Modifica" per confermarle. Se invece l'intenzione era di espellere totalmente l'utente dal sistema, lo si può fare premendo il pulsante "Elimina".

4.2.5 Admin

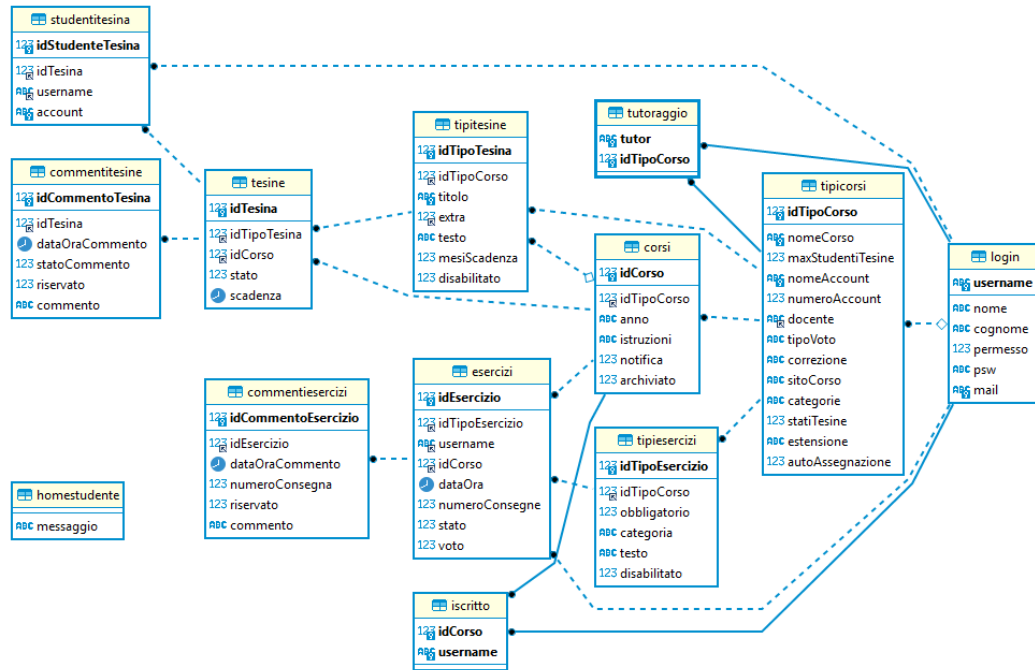
29. **Crea tipologia di corso:** si può creare un tipo di corso, inserendo nome del tipo di corso, studenti massimi per tesina, numero degli stati delle tesine, numero account tesine (fac.), numero account totali (fac.), possibilità di auto assegnazione tesina, docente e tutor assegnati, tipo di voti, software antiplagio

(sim o nessuno), sito del corso, estensione degli esercizi e categorie degli esercizi. I seguenti campi sono inizialmente nascosti: numero degli stati delle tesine, numero account tesine e numero account totali, compaiono solo se si inserisce un numero maggiore di zero in studenti massimi per tesina. Dal campo tutor si possono scegliere quali tutor sono assegnati al corso, non selezionando nulla non ci sono tutor, se invece se ne vuole più di uno basta premere ctrl e selezionare tutti i tutor desiderati. Le categorie degli esercizi invece definiscono le categorie (e quindi il numero di esercizi obbligatori minimi da consegnare per accedere alle tesine) dei tipi di esercizio che verranno poi creati dal docente. Se la possibilità di auto assegnazione viene messa a "no" (di default è "si") allora lo studente non può accedere alla pagina del pt.8 per quel tipo di corso.

30. **Gestisci tipologia di corso:** si sceglie un tipo di corso esistente e si modificano le informazioni del punto precedente. C'è però da fare attenzione, poiché modificando le categorie del tipo di corso potrebbero crearsi delle inconsistenze coi dati: è bene modificare questo campo solo se non sono stati ancora consegnati esercizi dagli studenti.
31. **Archivia corso:** (già descritta nel punto 17) con la differenza che vede i corsi di tutti i docenti.
32. **Elimina corso:** (già descritta nel punto 18) con la differenza che vede i corsi di tutti i docenti.
33. **Crea docente:** si presenta un form che chiede nome, cognome, mail, username, password e conferma password. Una volta inseriti i dati viene creato un utente nel sistema con permesso da docente. Se si inserisce uno username già utilizzato o una mail già utilizzata compare sullo schermo un errore relativo a quale campo risulta duplicato.
34. **Crea tutor:** come pt.33 ma per il tutor
35. **Corsi frequentati da studenti e disiscrizione:** (già descritta nel punto 27) con la differenza che può disiscrivere chiunque da qualunque corso.
36. **Messaggio home studenti:** si può cambiare il messaggio con le istruzioni base che si trova nella home degli utenti. Il messaggio va scritto in linguaggio HTML. Se non si hanno conoscenze a riguardo è sufficiente scrivere un testo normale, se invece si hanno conoscenze si possono usare testi in strong o italic per migliorare la leggibilità del testo.
37. **Gestisci utenti:** (già descritta nel punto 28) L'unica differenza sta nel poter gestire anche docenti ed altri admin, l'admin che usa il servizio non può però modificare se stesso. Bisogna fare attenzione all'eliminazione di docenti che hanno corsi assegnati: l'eliminazione, infatti, è consentita ma lascia dei tipi di corso privi di docente, fino a che non ne sarà assegnato uno nuovo.

4.3 Database

Qui sotto è presente uno schema logico del progetto del database con le varie tabelle ed i vincoli di primary e foreign key (rispettivamente indicati col grassetto e con le linee che uniscono 2 tabelle).



In particolare, qui vengono mostrati i campi di ogni tabella (primary key sarà indicata come PK e foreign key come FK), campi simili sono raggruppati in unico punto:

4.3.1 login

Questa è la tabella che viene consultata dal sistema quando serve un qualsiasi dato riguardante gli utenti:

- nome, cognome e mail contengono i dati anagrafici e la mail di tutti gli utenti del sistema (tutte stringhe);
- username contiene lo username scelto dall'utente, per lo studente corrisponde alla matricola (STRING, PK);
- permesso è un numero che va da uno a quattro ed indica rispettivamente se l'utente è studente, tutor, docente o admin (INT da 1 a 4, DEFAULT=1);
- password, salvata come digest di SHA512 (STRING di 128 char).

4.3.2 tipiCorsi

I "Tipi di corso" sono la generalizzazione di un corso, per esempio un corso può essere Linguaggi per il Web 2021-2022, il tipo di corso corrispondente è Linguaggi

per il Web. Il tipo di corso contiene quindi tutti i dati generici di un corso, dati che sono comuni a tutti i corsi di quel tipo.

- `idTipoCorso` è l'identificativo del tipo di corso (INT, AUTOINCREMENT, PK);
- `nomeCorso` e `sitoCorso` contengono il nome del corso ed il sito (STRING);
- `maxStudentiTesine` contiene il numero massimo di studenti che possono fare una tesina, infatti le tesine possono essere fatte da 1 o più studenti, in base alle preferenze del docente. Se il docente decide di fare dei corsi senza tesine, e vuole quindi sfruttare solo la funzione degli esercizi del sistema, `maxStudentiTesine` viene messo a 0, indicando la disattivazione delle tesine per questo tipo di corso (INT, DEFAULT=0);
- `nomeAccount`: per alcuni corsi il docente può fornire degli account affinché gli studenti possano usarli per caricare i propri esercizi/tesine online. Questi account possono avere un prefisso comune, questo campo è dove viene salvato il prefisso. Verrà fatto un esempio fra poco per chiarire meglio il concetto (STRING, DEFAULT=NULL);
- `numeroAccount`: riprendendo quanto scritto nel punto precedente, il docente ha un numero di account limitato, qui va il numero. Per esempio: se un docente possiede degli account con username `lweb1`, `lweb2`... `lweb40`, si avrà `nomeAccount=lweb` e `numeroAccount=40`. Le password degli account vengono date via mail dal docente e non vengono memorizzate nel sistema. Se non ci sono tesine questo campo ed il precedente sono NULL (INT, DEFAULT=NULL);
- `docente` contiene l'username del docente che tiene il tipo di corso (STRING, FK);
- `tipoVoto` contiene un valore fra "binario", "decimale", "trentesimi", che indica che tipo di voto può essere dato agli esercizi. Binario indica che non c'è voto e quindi l'esercizio può essere solo accettato o rifiutato (STRING, DEFAULT=binario);
- `correzione` indica se il tipo di corso ha un sistema antiplagio per gli esercizi. Il sistema propone sim [22] come sistema antiplagio per gli esercizi in .c oppure nessun sistema antiplagio (STRING, DEFAULT=NULL);
- `categorie` contiene i nomi delle categorie dei tipi di esercizio proposti dal tipo di corso, ogni categoria è separata dall'altra col carattere "|" (STRING);
- `statiTesine` indica quanti stati di avanzamento hanno le tesine, se presenti (INT, DEFAULT=0);
- `estensione` contiene l'estensione (o le estensioni, separate da "|") che il sistema accetta per gli esercizi consegnati dagli studenti (STRING);
- `autoAssegnazione` può essere 0 o 1 ed indica se il sistema permette o meno che lo studente si auto-assegna una tesina (INT, DEFAULT=1).

4.3.3 tutoraggio

Questa tabella contiene le informazioni riguardanti i tutor assegnati ai tipi di corso.

- tutor contiene l'username del tutor che tiene il tipo di corso (STRING, FK);
- idTipoCorso contiene l'id del tipoCorso (INT, FK).

I due campi insieme formano la primary key.

4.3.4 corsi

- idCorso è l'identificativo del corso (INT, AUTOINCREMENT, PK);
- idTipoCorso indica a quale tipo corso appartiene questo corso (INT, FK);
- anno indica l'anno accademico in cui si svolge il corso (STRING);
- istruzioni contiene le istruzioni che verranno presentate allo studente in fase di consegna esercizi (STRING);
- notifica può essere 0 o 1, se è 1 vuol dire che qualche studente ha consegnato un esercizio per quel corso e viene mostrata una notifica al docente del corso nella sua home page (INT, DEFAULT=0);
- archiviato indica se il corso è stato archiviato, può essere 0 o 1 (INT, DEFAULT=0).

4.3.5 iscritto

Contiene le iscrizioni ai corsi, rappresentate da una coppia idCorso ed username.

- idCorso è l'identificativo del corso (INT, FK);
- username è l'identificativo dello studente (STRING, FK).

I due campi insieme formano la primary key.

4.3.6 tipiEsercizi

I tipi di esercizio sono la generalizzazione degli esercizi, per esempio creare un programma che sommi due numeri interi è un tipo di esercizio, quando poi uno studente svolge quel tipo di esercizio allora si tratta di un esercizio effettivo. Un tipo di esercizio vale per tutti i corsi del tipo di corso a cui appartiene.

- idTipoEsercizio è l'identificativo del tipo di esercizio (INT, PK, AUTOINCREMENT);
- idTipoCorso identificativo del tipo di corso a cui il tipo di esercizio fa riferimento (INT, FK);
- obbligatorio 0 o 1, se è 1 è obbligatorio (INT, DEFAULT=0);

- categoria dell'esercizio che deve essere una delle categorie contenute nel campo categorie del tipo corso di riferimento (STRING);
- testo che contiene la consegna del tipo di esercizio (STRING);
- disabilitato 0 o 1, se è 1 il tipo di esercizio non è più visibile ai nuovi studenti ma solo a chi l'ha già sottomesso (INT, DEFAULT=0).

4.3.7 esercizi

Qui vanno gli esercizi svolti dagli studenti, i due elementi importanti della tabella sono il tipo di esercizio svolto e l'username di chi l'ha svolto:

- idEsercizio è l'identificativo dell'esercizio (INT, PK, AUTOINCREMENT);
- idTipoEsercizio identificativo del tipo di esercizio che è stato svolto dallo studente (INT, FK);
- username dello studente (STRING, FK);
- idCorso corso per il quale lo studente ha consegnato (visto che i tipi di esercizio si ripetono per tutte le annate dei vari corsi, bisogna specificare qui di quale anno si sta parlando) (INT, FK);
- dataOra data ed ora dell'ultima consegna dell'esercizio (DATETIME);
- numeroConsegne serve a capire quante volte lo studente ha effettuato la riconsegna di quell'esercizio, se è=1 vuol dire che lo studente ha solo consegnato, se è=2 allora ha consegnato e poi riconsegnato 1 volta etc... (INT, DEFAULT=1);
- stato indica lo stato dell'esercizio, va da 1 a 3 che significano rispettivamente "inviato", "da rivedere", "accettato" (INT, DEFAULT=1);
- voto indica il voto. il sistema usato per il voto (decimale o trentesimi) dipende dal tipo di corso (INT, DEFAULT=NULL).

4.3.8 tipiTesine

Per le tesine (e i tipi di tesina) vale lo stesso discorso degli esercizi

- idTipoTesina è l'identificativo del tipo di tesina (INT, PK, AUTOINCREMENT);
- idTipoCorso è l'identificativo del tipo di corso a cui il tipo di esercizio fa riferimento (INT, FK);
- titolo è il titolo del tipo di tesina (STRING);
- extra è un parametro che indica se il tipo di tesina è specifico di un singolo corso (invece che di un tipo di corso), se è NULL vuol dire che non è specifica, se è diverso da NULL allora il valore è l'id del corso per cui è stata creata (INT, FK, DEFAULT=NULL);

- mesiScadenza indica dopo quanti mesi dall'assegnazione la tesina risulta scaduta, se è 0 allora non ha scadenza (INT, DEFAULT=0);
- disabilitato 0 o 1, se è 1 il tipo di tesina non è più visibile ai nuovi studenti ma solo a chi ha già quel tipo di tesina assegnato (INT, DEFAULT=0).

4.3.9 commentiEsercizi

- idCommentoEsercizio è l'identificativo del commento (INT, PK, AUTOINCREMENT);
- idEsercizio è l'identificativo dell'esercizio a cui si riferisce il commento (INT, FK);
- dataOraCommento si riferisce alla data e ora di quando l'esercizio è stato consegnato l'ultima volta (DATETIME);
- numeroConsegna indica a quale consegna è stato fatto il commento, quindi è uguale al campo numeroConsegne (della tabella esercizi) dell'esercizio alla creazione del commento (INT);
- riservato può essere 0 o 1 ed indica se il commento è riservato al docente o meno (INT, DEFAULT=0);
- commento contiene il testo del commento (STRING).

4.3.10 tesine

- idTesina è l'identificativo della tesina (INT, PK, AUTOINCREMENT);
- idTipoTesina è l'identificativo del tipo di tesina svolta dall'utente (INT, FK);
- idCorso è il corso per il quale lo studente ha consegnato (visto che i tipi di tesina si ripetono per tutte le annate dei vari corsi, bisogna specificare qui di quale anno si sta parlando) (INT, FK);
- stato indica lo stato di avanzamento di questa tesina, parte da 1 ed arriva fino al numero di statiTesine di tipiCorsi + 1, infatti questo numero indica in che stato si trova la tesina, trovarsi nello stato 4 significa aver completato lo stato 3, quindi si eccede di 1 rispetto al numero di stati massimi definiti nel tipoCorso (INT, DEFAULT=1);
- scadenza indica la data scadenza della tesina (DATE).

4.3.11 commentiTesine

- idCommentoTesina è l'identificativo del commento (INT, PK, AUTOINCREMENT);
- idTesina è l'identificativo della tesina a cui si riferisce il commento (INT, FK);

- dataOraCommento è la data e ora in cui è stato inserito il commento (DATE-TIME);
- statoCommento è lo stato della tesina a cui si riferisce il commento (INT);
- riservato può essere 0 o 1 ed indica se il commento è riservato al docente o meno (INT, DEFAULT=0);
- commento contiene il testo del commento (STRING).

4.3.12 studentiTesina

Questa tabella contiene gli studenti che stanno volgendo una tesina (indicata da idTesina), questa tabella esiste in quanto non c'è solo uno studente a svolgere ogni tesina, ma ce ne può essere più di uno.

- idStudenteTesina è un identificativo (INT, PK, AUTOINCREMENT);
- idTesina è l'identificativo della tesina a cui si riferisce (INT, FK);
- username è la matricola dello studente (STRING, FK);
- account indica l'account assegnato allo studente (STRING).

4.3.13 homeStudente

- messaggio contiene una sola row col messaggio che viene mostrato nella home dello studente, in modo da renderlo modificabile dall'admin (STRING, PK).

Inoltre il db ha un trigger:

```
create trigger delete_iscritto
before delete on iscritto
for each row
BEGIN
DELETE FROM esercizi WHERE idCorso=OLD.idCorso AND username=OLD.username;
DELETE FROM tesine WHERE idCorso=OLD.idCorso AND idTesina IN
(SELECT idTesina FROM studentitesina WHERE idTesina IN
(SELECT idTesina FROM studentiTesina where username=OLD.username)
GROUP BY idTesina HAVING count(idTesina)=1);
DELETE studentitesina FROM studentitesina
INNER JOIN tesine on tesine.idTesina=studentitesina.idTesina
WHERE idCorso=OLD.idCorso AND username=OLD.username;
END;
```

Questo trigger agisce prima di un'eliminazione sulla tabella iscritto. La sua funzione è eliminare tutti gli esercizi dello studente e tutte quelle tesine in cui lo studente è l'unico partecipante. Se invece ci sono più partecipanti nella tesina, viene eliminata solo la partecipazione dello studente eliminato.

4.4 Programmazione

Un modello usato per tutti quei servizi che hanno bisogno di un id prima di funzionare, come per esempio per modificare un tipo di tesina serve prima l'id del tipo tesina da modificare, è il seguente:

La pagina controlla `if(isset($_GET["nomeIdNecessario"]))` se ciò è false rimanda ad una pagina di selezione, se per esempio cerco l'id di un esercizio avrò `if(isset($_GET["idEsercizio"]))` `go to` `selezioneEsercizio.php?to=__FILE__`, dove `__FILE__` indica il nome del file da cui sto lanciando lo script. Quindi in pratica la pagina su cui mi trovo e che ha bisogno di un parametro, chiama la pagina utile ad ottenerlo, dando in GET il parametro `to =` al nome della pagina su cui lo script chiamato deve poi tornare. La pagina di selezione è sempre costituita come una lista dei corsi, esercizi, tesine etc. che sono disponibili all'utente. Ogni categoria di utente ha le sue pagine di selezione. Per esempio, lo studente che vuole modificare un esercizio consegnato deve poter selezionare solo gli esercizi di cui lui è autore (e quindi dove risulti nel campo username della tabella esercizi il suo username). Invece magari il docente deve vedere solo i corsi di cui è docente (quindi risulti campo docente nella tabella tipiCorsi = al suo username) etc... non andrò nel dettaglio di ogni script di selezione in quanto molto simili. Nel caso in cui sono necessarie selezioni multiple, come per esempio quando si corregge un esercizio in cui prima si sceglie il corso e poi l'esercizio da correggere di quel corso, lì la pagina di correzione ridireziona a `selezioneCorso.php?to=selezioneEsercizio.php` e poi `selezioneEsercizio`, essendo utilizzata solo da `correggiEsercizio.php`, è hard-coded per portare sempre a `correggiEsercizio.php` (in quanto unico script che usa questo doppio percorso). Questo accade per quasi tutti i servizi del sistema, quindi non verrà ripetuto ogni volta.

Ogni pagina (eccetto login e registrazione) controlla il permesso dell'utente tramite la variabile `$_SESSION["permesso"]`, se la variabile non è settata oppure se un utente tenta di accedere ad una pagina di un utente di diverso permesso, come se per esempio uno studente (`permesso=1`) tenta di accedere alla pagina di correzione esercizi (`permesso richiesto=3`), allora verrà rimandato alla pagina di login che provvederà a fare il redirect verso la giusta pagina home.

1. **Login:** Il login setta le variabili di sessione nome, cognome, permesso (da 1 a 4, con stessa semantica che hanno quei numeri nel db, si veda 4.3), email, username e valid (true se sessione valida). Fatto ciò l'utente sarà portato ad una pagina in base al suo permesso. Se si visita la pagina di login dopo essersi loggati, si verrà comunque rimandati alla propria home, sempre in base al permesso. Per controllare se loggare un utente si fa una classica query al db selezionando utenti con nome utente e password che siano uguali al nome utente fornito e all'hash SHA512 della psw. Se c'è una corrispondenza allora setta le variabili di sessione altrimenti no.

4.4.1 Studente

2. **Registrazione:** I campi del form vengono inseriti in db con una query (la password viene sempre hashata), l'insert però avviene solo se password e conferma password combaciano e se non ci sono utenti con mail o username uguali a quelli da inserire nel db (si fa una query per ognuno dei 2 campi, inserendo nella WHERE clause email e username inseriti in registrazione). Se le due query restituiscono due risultati vuoti, si registra e si settano anche variabili di sessione per dargli già il login fatto.
3. **Modifica mail o password:** una semplice query di update della tabella login con WHERE clause basata sulla matricola e dati presi dai campi nel form. I campi del form vengono riempiti da dati presi dal db con una query che abbia come WHERE la matricola.
4. **Iscrizione corso:** all'interno di un form c'è la tabella che contiene i corsi, presi dal db dalla table corsi, che hanno archiviato=0 (ossia che non sono archiviati), però da questi corsi vengono tolti con un NOT IN quei corsi da cui l'utente è già iscritto. Qui si vede il NOT IN nel codice:

```
WHERE archiviato=0 and c.idCorso NOT IN (  
    SELECT idCorso  
    FROM iscritto  
    WHERE username=?  
)
```

il "?" è per il prepared statement, al suo posto viene messa la matricola dello studente. Una volta premuto uno dei pulsanti di iscrizione, viene aggiunta una entry al db alla tabella iscritto con idCorso preso dal value del submit eseguito, infatti ogni riga della tabella HTML ha un submit con name identico e value=idCorso.

5. **Dettagli corso:** è un semplice dump della tabella corsi del db, joinata a tipiCorsi sul campo idTipoCorso.

Da questo punto in poi verrà fatto sempre un controllo a priori dell'effettiva iscrizione dell'utente al corso tramite query, se non è registrato viene mandato al login. La query consiste nel prendere l'idCorso e l'username dell'utente e nel vedere se c'è una corrispondenza nella table iscritto del db.

6. **Consegna esercizi e testi ad apparizione:** dopo aver scelto il corso si seleziona un tipo di esercizio, ogni tipo di esercizio ha un testo a comparsa. I testi a comparsa sono inizialmente nascosti. Viene infatti incluso il file js/showHideTesto.js, scritto in jQuery, che nasconde tutte le div con classe "show_hide". Una volta premuto un pulsante con classe "control" (come lo sono tutti i pulsanti per mostrati i testi) tramite jQuery parte un trigger che prende l'id del pulsante premuto. Ogni pulsante di classe control ha id=all'id del tipo di esercizio a cui si riferisce. Ogni div nascosta ha un id che è "testo\$id", con \$id=all'id del tipo di esercizio, sfruttando ciò lo script fa .fadeIn("slow")

dell'elemento con `id="testo$id"`, ossia del testo relativo al pulsante premuto. Dopo aver fatto il `fadeIn`, ossia lo `show` con transizione, viene salvato l'id del tipo di esercizio tramite la classe di JavaScript `localStorage`. Questo serve perché prima di fare l'effettivo `fadeIn` lo script controlla anche quale id è salvato in `localStorage`, se c'è salvato lo stesso che sto trattando adesso, ciò significa che sto premendo il pulsante relativo ad un testo già aperto, in quel caso allora lo script lo nasconde.

Dopo aver spiegato come funzionano i testi ad apparizione, bisogna spiegare la consegna di esercizi. La lista include tutti i tipi di esercizi del tipo di corso meno quelli per cui esiste un esercizio già consegnato dall'utente e tolti anche quelli con `obbligatorio=1` tali per cui l'utente ha già consegnato un esercizio della stessa categoria.

Dopo aver selezionato il file da caricare e quale tipo di esercizio consegnare si ha la gestione in POST nella stessa pagina. Da POST vengono presi l'id del tipo di esercizio (salvato nel value dell'input radio), tramite `move_uploaded_file` si sposta il file dello studente nella cartella con nome `"nomeCorso.anno"` (nomeCorso ed anno sono presi tramite query dal tipo corso relativo al corso), il file viene però prima rinominato in `categoria.cognome.nome.matricola.idTipoEsercizio.estensione` (i dati mancanti sono presi dal db a partire dalla matricola). Fatto ciò viene messo il campo `notifica=1` per il corso con `idCorso` corrispondente. Se ciò va a buon fine c'è l'insert nel db nella tabella esercizi con data ed ora dell'inserimento.

7. **Gestione esercizi:** dopo aver scelto il corso vengono mostrati gli esercizi che sono quelli contenuti nella tabella esercizi con `idCorso` preso in GET e username preso da session. Si seleziona un esercizio e si può fare riconsegna o eliminazione, entrambe gestite in post tramite `idEsercizio` presente come value nel radio input. La riconsegna funziona come la consegna mostrata sopra, ma con delle differenze: innanzitutto viene riportato lo stato dell'esercizio ad 1, se l'esercizio era stato rifiutato (ossia aveva `stato=2` prima della riconsegna) viene incrementato il campo `numeroConsegne`. Se invece si era scelta l'eliminazione allora viene tolto dalla tabella esercizi e si fa `unlink` del file.
8. **Lista tesine:** si sceglie il corso e poi è un semplice dump della tabella tesine con `idCorso` di questo corso.
9. **Auto assegnazione tesine:** se il tipoCorso ha `autoAssegnazione=0` la pagina fa redirect al login. Se `autoAssegnazione=1` allora viene fatto redirect alla pagina di autoassegnazione `richiediTesina.php` dove ci sono dei controlli, prima viene controllato dalla tabella esercizi del db il numero di esercizi obbligatori e con `stato=3` dell'utente, se questo numero è = al count del numero di parole in categorie del tipoCorso, ossia uguale al numero di categorie, allora si passa al controllo successivo. Viene controllato che nella tabella tesine non ci siano entry con username dello studente e `idCorso` del corso (passato in GET da `listaTesine.php`). Se uno dei controlli fallisce, viene fatto redirect alla pagina `listaTesine.php` con parametro GET dell'idcorso & `warning=1`. Questa pagina se ha in GET `warning=1` stampa in cima un messaggio di warning per l'utente.

Se supera i controlli può rimanere nella pagina dove c'è una tabella presa con la seguente query:

```
SELECT *
FROM tipitesine
WHERE idTipoCorso=? AND disabilitato=0 AND idTipoTesina NOT IN(
SELECT t.idTipoTesina
FROM tipitesine tt
INNER JOIN tesine t on t.idTipoTesina=tt.idTipoTesina
WHERE idCorso=?
)
ORDER BY titolo
```

La query prende i tipi di tesina tale per cui idTipoTesina non esiste una entry in tesine (avente quell'idTipoTesina) tale per cui idCorso sia uguale all'idCorso in GET, quindi prende le tesine non ancora assegnate durante questo corso. Dopo aver fatto submit del form avviene la gestione in POST: viene per prima cosa preso il valore di mesiScadenza dalla tabella tipiCorsi del db, questi mesi vengono aggiunti alla data odierna per creare la scadenza, se mesiScadenza è 0 allora scadenza viene messo NULL. Una volta fatto ciò avviene l'insert nella tabella tesine con idTipoTesina preso in post dal value dell'input radio, idCorso da GET e scadenza come visto prima. C'è poi un secondo insert in studentiTesina: idTesina viene preso dalla tabella tesina tramite idTipoTesine ed idCorso e studente è l'username della sessione.

10. **Tesina assegnata:** Le informazioni sulla tesina sono prese dalla table tesina tramite l'idCorso e l'username della sessione, i commenti sono presi dalla tabella commentiTesine, lo stato è disegnato come una progress bar di Bootstrap divisa in n parti, con n = numero di stati massimi, e riempiti per m parti, con m = stato della tesina-1 (perchè lo stato indica lo stato in cui ci si trova, non indica che è completato, stato=2 per esempio significa che si è completato il primo e si è nel secondo).

4.4.2 Tutor

11. **Riepilogo corso:** si tratta di una query piuttosto lunga, in cui prima dalla tabella esercizi vengono presi tutti gli utenti di quel corso e viene fatto group by idEsercizio per fare count degli esercizi consegnati, poi viene fatto left join sullo username con una query identica ma con WHERE stato=3 per avere il numero di esercizi accettati, infine vengono aggiunti tutti gli utenti iscritti al corso ma che non hanno consegnato nulla, ciò viene fatto prendendo tutti dalla tabella iscritto con idCorso del corso e che non sono (NOT IN) nella query precedentemente fatta.
12. **Correggi esercizi (e plagio):** dopo aver scelto il corso si trova una tabella con tutti gli esercizi presi dalla tabella esercizi con l'idCorso giusto, ogni esercizio ha 2 pulsanti, correzione e plagio. Se però il plagio non è attivo nel

tipoCorso, ossia il campo correzione è NULL, allora il pulsante di plagio non viene mostrato. Per la correzione vengono stampate tante textareas quante sono i commenti nella table commentiEsercizi con quell'idEsercizio + una per un eventuale commento nuovo, poi in post si fa per ogni textarea vecchia query di update del testo dei commenti tramite l'id del commento che è salvato nella name della textarea, il value della textarea è il testo ovviamente. Se il testo dell'ultima textarea non è vuoto si fa insert nei commenti con data e ora presa dalla tabella esercizi tramite l'idEsercizio. Lo stato dell'esercizio viene poi updatato in base a quale dei due pulsanti si è premuto, 3 se accettato e 2 se rifiutato.

Le textareas dei commenti sono aggiornate di dimensione da uno script di jQuery, ossia js/textarea.js che controlla al caricamento della pagina, per ogni textarea, che la height sia uguale alla scrollheight, se non lo è viene messa height=scrollHeight, così la textareas non ha bisogno di essere scrollata, in quanto alta tanto quanto il testo contenuto. Inoltre ogni volta che si inserisce un carattere in una textarea viene eseguita la stessa cosa.

Invece per il plagio, prima si sceglie che tipo di confronto fare, se 1 ad 1, 1 ad n oppure n ad n. La select contiene tutti gli studenti che hanno consegnato un esercizio dello stesso tipo in quel corso. In base a quale pulsante viene premuto viene generato un comando diverso passato in GET alla pagina successiva (c'è una pagina per ognuna di queste scelte). Il comando viene eseguito dalla pagina su cui si è arrivati tramite

```
$risultato=shell_exec($_GET["comando"]);
```

il comando chiama, fornendo vari parametri e i path degli esercizi da confrontare, un .exe esterno chiamato sim_c.exe [22] il quale analizza la somiglianza degli esercizi. Il risultato è contenuto nella variabile \$risultato, nel caso del confronto 1 ad 1 contiene le percentuali sotto forma di stringa su cui fare parse, negli altri confronti contiene il codice HTML di un SVG che è il grafico che sarà stampato sulla pagina, contenente il grafo con le percentuali di plagiarismo fra i vari studenti, ottenuto tramite Graphviz [21].

13. **Assegna tesine:** dopo aver scelto un corso ed un tipo di tesina, se il tipo di tesina non è ancora assegnato si accede alla pagina di assegnazione delle tesine. Questa ha un numero di select pari a maxStudentTesina della tabella tipiCorsi. Ogni select contiene tutti gli studenti che hanno consegnato tanti esercizi (con obbligatorio=1 ed accettati, ossia con stato=3) quanto il numero di categorie del tipo di corso, ossia gli studenti a cui è stato accettato un esercizio obbligatorio per categoria. Inoltre accanto ad ogni select c'è un'altra select per l'account da assegnare che contiene come options gli account che non sono usati da altri studenti, quindi:

```
$AccountTotali = [];  
$i = 0;  
while ($i < $numeroAccount) {  
    $num = $i + 1;
```



```

    $AccountTotali[$i] = "$nomeAccount$num"; //array tipo lweb1,lweb2 ecc...
    $i++;
}

```

nomeAccount e numeroAccount sono presi dal db dalla table tipiCorsi, a questi però vengono tolti gli account presenti nella tabella studentiTesina (ossia quelli già assegnati a qualcuno):

```

$AccountPresi = [];
$i = 0;
$query = "SELECT account
FROM studentitesina s";
$result = $link->query($query);
while ($row = $result->fetch_assoc()) {
    extract($row, EXTR_OVERWRITE);
    $AccountPresi[$i] = "$account";
    $i++;
}
//faccio array_diff ossia differenza fra array
$accountDisponibili = array_diff($AccountTotali, $AccountPresi);

```

Una volta premuto il pulsante di assegnazione vengono fatti, in POST, gli inserimenti nelle tabelle tesine e studentiTesina. Nello specifico, viene per prima cosa preso il valore di mesiScadenza dalla tabella tipiCorsi del db, questi mesi vengono aggiunti alla data odierna per creare la scadenza, se mesiScadenza è 0 allora scadenza viene messo NULL. Una volta fatto ciò avviene l'insert nella tabella tesine con idTipoTesina preso in post dal value dell'input radio, idCorso da GET e scadenza come visto prima. Ci sono poi altri insert in studentiTesina, in particolare un insert per ognuno degli n studente selezionato: idTesina viene preso dalla tabella tesina tramite idTipoTesine ed idCorso e studente è la matricola dell'i-esimo studente selezionato.

Se il tipo di tesina era già assegnato, si accede ad una pagina che è identica a quella di assegnazione, però con i select già riempiti con i dati di chi è già assegnato, ciò è fatto prendendo l'id del corso e l'id del tipo tesina, da questi si prende l'idTesina dalla tabella tesine, e dall'idTesina si recuperano gli studenti assegnati alla tesina e i relativi account da studentiTesina. Inoltre si può modificare anche la data di scadenza, presa dalla tabella tesine. Facendo submit in POST vengono eliminate tutte le entry di studentiTesina con idTesina corrispondente, e subito dopo vengono reinseriti gli studenti presenti nel form in studentiTesina, che potrebbero essere più o meno di prima. Viene anche fatto l'update della scadenza in tesine.

14. **Stati tesine:** I commenti sono inseriti con la stessa tecnica spiegata in 12. L'unica differenza è che se viene premuto il pulsante che fa avanzare di stato, oltre ad essere salvati i commenti, viene avanzato lo stato della tesina di 1. Il pulsante di avanzamento è disponibile finché stato della tesina <= statiTestine della tabella tipiCorsi. Lo stato indica lo stato in cui ci si trova, quindi per

esempio se gli stati sono al massimo 3, per indicare di aver completato il secondo stato lo stato sarà=3, e quindi devo poter ancora avanzare di uno stato prima di completare, quindi è giusto il \leq nel confronto.

4.4.3 Docente

15. **Crea corso:** semplice insert in POST nella tabella corsi con dati forniti dal form.
16. **Riepilogo corso:** (già descritta nel punto 11).
17. **Modifica corso:** update nella tabella corsi delle istruzioni con idCorso fornito in GET.
18. **Archivia corso:** la pagina mostra tutti i corsi assegnati al docente, se si preme il pulsante di archiviazione o di attivazione si passa alla gestione in POST. Siccome ci sono 2 tabella nella stessa pagina con 2 azioni diverse da eseguire, si cerca nell'array POST e si vede se il value="Archivia" oppure "Attiva", se ci si trova nel primo caso si pone archiviato=1 per il corso con idCorso = al name del submit, altrimenti stessa cosa ma archiviato=0.
19. **Elimina corso:** viene fatto delete da db con idCorso preso dal name del submit premuto. Inoltre viene eliminata la directory del corso con una funziona che fa scan ricorsivo delle sottocartelle, eliminando ogni file che trova ed infine la cartella madre:

```
function deleteDirectory($dir) {
    if (!file_exists($dir)) return true;
    if (!is_dir($dir)) return unlink($dir);
    foreach (scandir($dir) as $item) {
        if ($item == '.' || $item == '..') continue;
        if (!deleteDirectory($dir . DIRECTORY_SEPARATOR . $item)) return false;
    }
    return rmdir($dir);
}
```

20. **Crea tipo esercizio:** si seleziona il tipo di corso e poi viene fatto insert in db nella tabella tipiEsercizi con i dati presi in POST dal form ed idTipoCorso da GET.
21. **Gestisci tipi esercizio:** si seleziona il tipo di corso, poi il tipo di esercizio e poi viene fatto update della tabella tipiEsercizi con i dati presi in POST dal form ed idTipoEsercizio da GET.
22. **Correggi esercizi (e plagio):** (già descritta nel punto 12).
23. **Crea tipo tesina:** si seleziona il tipo di corso e poi viene fatto insert in db nella tabella tipiTesine con i dati presi in POST dal form ed idTipoCorso da GET.

- 24. **Gestisci tipi tesina:** si seleziona il tipo di corso, poi il tipo di tesina e poi viene fatto update della tabella tipiTesine con i dati presi in POST dal form ed idTipoTesina da GET.
- 25. **Assegna tesine:** (già descritta nel punto 13).
- 26. **Stati tesine:** (già descritta nel punto 14).
- 27. **Reset account tesine:** in POST si prende nomeAccount dalla select, infatti ogni option ha come value nomeAccount, preso dal tipo di corso dal db. A questo punto si fa una query di update che setta NULL nomeAccount, nella tabella studentiTesine, per le entry che hanno il campo account formato da quel nomeAccount ed un numero. Ciò è fatto usando l'operatore LIKE di SQL:

```
$query = "UPDATE studentiTesina
        SET account=null
        WHERE account LIKE '$nomeAccount%'";
```

- 28. **Corsi frequentati da studenti e disiscrizione:** Quando viene selezionata la disiscrizione di uno studente da un corso, in POST viene preso il name del pulsante che contiene username|idCorso e viene fatto delete da iscritto con idCorso e username. Inoltre il db ha un trigger descritto in 4.3 che elimina esercizi e tesine dello studente prima di disiscriverlo dal corso. Il pulsante di disiscrizione è presente solo se il corso è di un tipoCorso che ha il campo docente = all'username della sessione.
- 29. **Gestisci utenti:** vengono mostrati tutti gli utenti del sistema con permesso < 3, poi è un semplice update della table login dato l'username, però come option nella select del permesso ci sono solo quelle con permesso < 3, ossia tutor e studente (per evitare privilege escalation).

4.4.4 Admin

- 30. **Crea tipologia di corso:** viene fatto insert dei dati inseriti nella table tipiCorsi. Una particolarità sta nel fatto che 3 dei campi da riempire sono nascosti, a meno che non venga inserito un numero>0 in studenti massimi per tesina. Ciò è gestito da uno script (js/showHideCreaTipoCorso.js) che controlla se viene premuto un tasto in un campo che ha id=control (studenti massimi per tesina ha id=control), a quel punto controlla il value, se la lunghezza>0 e valore >0 allora mostra i campi con classe=show_hide (ossia i 3 campi nascosti) con animazione .fadeIn("slow").
- 31. **Gestisci tipologia di corso:** si seleziona il tipo di corso e poi viene fatto update della tabella tipiCorsi con i dati presi in POST dal form ed idTipoCorso da GET.
- 32. **Archivia corso:** (già descritta nel punto 18) la differenza sta nel fatto che nella query per mostrare i corsi non viene considerato il WHERE docente=\$username, così da mostrare tutti i corsi.

33. **Elimina corso:** (già descritta nel punto 19) la differenza sta nel fatto che nella query per mostrare i corsi non viene considerato il WHERE docente=\$username così da mostrare tutti i corsi.
34. **Crea docente:** come 2 ma viene anche inserito il permesso=3 nella query.
35. **Crea tutor:** come 2 ma viene anche inserito il permesso=2 nella query.
36. **Corsi frequentati da studenti e disiscrizione:** (già descritta nel punto 28) ma tutte le righe hanno il pulsante di disiscrizione.
37. **Messaggio home studenti:** update del campo messaggio dell'unica entry della tabella homeStudente.
38. **Gestisci utenti:** (già descritta nel punto 29) non c'è però il vincolo del permesso.

Capitolo 5

Manuale utente

Per prima cosa per avere un'idea di cosa fa il sistema si rimanda al capitolo di introduzione (1)

5.1 Studente

5.1.1 Primo utilizzo

Se non si è mai usato il sistema la prima cosa da fare è la registrazione, per registrarsi bisogna premere il pulsante "Registrati" che compare nella prima pagina di BRIDGE, una volta fatto ciò bisogna inserire tutti i dati richiesti e si viene registrati. Viene anche fatto il login automaticamente, quindi nella sezione successiva si deve saltare la parte relativa al login.

5.1.2 Utilizzo

Per prima cosa fare il login con matricola e password forniti durante la registrazione, nel caso in cui si fosse dimenticata la password contattare l'amministratore del sistema (o un docente) per il recupero della password.

Se si vuole cambiare la password o la mail associata al proprio account si può fare nella home, infatti sotto al piccolo manuale utente è presente una sezione dove si possono modificare mail o inserire una nuova password. Una volta scelto cosa modificare basta premere "modifica" e le modifiche vengono salvate.

Per **consegnare gli esercizi** di un corso bisogna innanzitutto essere iscritti al corso, se non si è iscritti bisogna selezionare dal menù la voce "iscrizione corso", si accede ad una pagina con tutti i corsi a cui ci si può iscrivere.

Se prima di iscriversi si è interessati a vedere dei dettagli sui corsi si possono vedere dalla voce del menù "Dettagli corsi"

Se invece si è già registrati si va direttamente in "Consegna esercizi", si sceglie il corso per il quale consegnare l'esercizio e ci si ritrova davanti ad una pagina simile:

Consegna esercizi Linguaggi per il web 2020-2021

Ci sono degli esercizi obbligatori ed altri facoltativi, il tuo compito è svolgere un esercizio obbligatorio per ognuna delle **3** categorie. Gli esercizi facoltativi sono invece a tua discrezione. Farli non è pericoloso, potrebbe invece essere utile! Ci possono essere più esercizi "obbligatori" per un medesimo argomento: in tal caso scegline uno

Seleziona	Categoria	Obbligatorio	Testo
<input type="radio"/>	xhtml	×	Testo
<input type="radio"/>	php	×	Testo
<input type="radio"/>	xml	×	Testo
<input type="radio"/>	xml	×	Testo

Seleziona uno degli esercizi con il pallino alla sua sinistra, carica il file corrispondente e premi "Consegna"

Se devi consegnare una cartella, comprimila e poi fai l'upload

Se una volta inserito l'esercizio vorrai fare dei cambiamenti, ti basterà selezionare dal menù [esercizi consegnati](#) e fare la riconsegna o l'eliminazione

Scegli file Nessun file selezionato

Per questo corso devi consegnare file che abbiano estensione **.rar / .zip / .tar.gz / .tgz / .7z / .xz**

Consegna

Per ogni esercizio si può vedere il testo tramite i pulsanti evidenziati in azzurro sulla destra, una volta che si è svolto l'esercizio bisogna consegnarlo. Un esercizio è obbligatorio se nella colonna obbligatorio compare una spunta verde (qui ci sono tutte X rosse, sono tutti facoltativi). Per consegnare basta selezionare l'esercizio scelto con il pallino (i pallini sono quelli evidenziati in rosso a sinistra), caricare il file premendo "scegli file" e premere consegna. I formati supportati per il file sono riportati sotto la sezione dove si carica il file.

Una volta che l'esercizio è stato caricato viene visualizzato un messaggio di conferma. Inoltre l'esercizio non è più presente nella pagina, così come non sono più presenti tutti gli esercizi obbligatori della stessa categoria dell'esercizio consegnato. Questo accade perchè per ogni corso bisogna consegnare un solo esercizio obbligatorio per categoria, invece per gli esercizi facoltativi c'è piena libertà, si possono consegnare anche tutti, ciò non influirà sull'avanzamento nel corso ma solo sulla preparazione personale.

Inoltre sopra la tabella c'è scritto il numero di categorie presenti nel corso, e quindi il numero di esercizi obbligatori da consegnare per accedere alle tesine.

Se si vuole apportare una **modifica agli esercizi consegnati** o se si vuole vedere se il docente li ha corretti si può selezionare dal menù "Gestione esercizi", si sceglie

quindi il corso e ci si ritrova davanti ad una pagina simile:

Seleziona	Categoria	Obbligatorio	Stato	Voto	Commenti (e numero consegna)	Testo
<input type="radio"/>	php	✓	Da rivedere		1° consegna: manca il readme, veda il sito del corso per capire come strutturarla	Testo
	xhtml	✓	Accettato		1° consegna: vedi issue github 2° consegna: ora va bene	Testo
	xml	✓	Accettato		1° consegna: commento in aggiungiVoto su github ----- forse qui ci sono ancora un po' di problemi con la gestione dei caratteri accentati	Testo

1) Seleziona uno degli esercizi con il pallino alla sua sinistra

2) Se vuoi riconsegnare l'esercizio scegli il file e clicca "Riconsegna", se invece vuoi solo eliminarlo clicca solo "Elimina" (un esercizio accettato non può essere riconsegnato o eliminato)

Se devi riconsegnare una cartella, inseriscila in un file .rar

Se elimini un esercizio perderai tutti i commenti associati, a meno che tu non voglia cambiare esercizio riconsegnarlo invece di eliminarlo

Scegli file

Nessun file selezionato

Per questo corso devi consegnare file che abbiano estensione **.rar / .zip / .tar.gz / .tgz / .7z / .xz**

Riconsegna

Elimina

Qui sono presenti gli esercizi consegnati per quel corso, si può sempre vedere il testo dell'esercizio nel caso in cui si volesse riconsegnare l'esercizio. Per riconsegnare l'esercizio basta selezionare il pallino dell'esercizio che si vuole riconsegnare, selezionare il nuovo file da consegnare e premere "Riconsegna". Se invece si è deciso di consegnare un altro esercizio di quella categoria, è necessario prima eliminare quello consegnato selezionandolo e premendo "Elimina".

Qui sono presenti anche altre informazioni: l'esercizio ha uno stato, che può essere "Inviato" se è stato solo inviato ma non ancora corretto, "Da rivedere" se il docente lo ha visto e non lo ha ancora ritenuto sufficiente, "Accettato" se l'esercizio è ok. Un esercizio accettato non può essere riconsegnato. Ci sono poi i commenti, per ogni esercizio ci possono essere 1 o più commenti scritti dal docente, per ogni commento è anche indicato a quale consegna si riferisce.

Per accedere alle **tesine** è necessario aver consegnato 1 esercizio obbligatorio per

ogni categoria ed è necessario che il docente li abbia accettati tutti, se non sono stati accettati bisogna riconsegnarli nella sezione di gestione esercizi precedentemente descritta. Si può accedere alla lista delle tesine del corso anche se non si ha ancora completato la parte degli esercizi, andando dal menù in "Lista tesine" e scegliendo il corso, fatto ciò si accede ad una tabella con vari informazioni sulle tesine: il titolo, la disponibilità ed il testo a comparsa (come visto negli esercizi). In fondo a questa pagina, se il corso lo permette, si trova un pulsante blu con scritto "Auto assegnazione", l'auto assegnazione permette di lavorare su una tesina da soli. Il pulsante porterà ad una pagina per auto assegnarsi la tesina, in questa nuova pagina compariranno tutte le tesine disponibili e basterà selezionarne una premendo il pallino, allo stesso modo in cui si selezionavano gli esercizi, e premendo poi "Assegna". Se invece il corso non presenta auto assegnazione, oppure si vuole fare la tesina con uno o più compagni, bisogna fare la richiesta al docente via mail ed attendere l'assegnazione.

Quando si ha una **tesina assegnata** la pagina "Tesina assegnata", dopo aver scelto il corso di cui vedere la tesina, risulta simile alla seguente:

HELPDESK

Qui trovi tutti i dati relativi alla tesina che ti è stata assegnata, se vuoi avanzare al prossimo stato parla col professore!

Scadenza	Account assegnato
Mancano ancora 2 mesi e 15 giorni	lweb21

Testo:

```
Gestione help desk
-----
testo
```

Questa barra indica lo stato di avanzamento della tesina:

1° Stato

2° Stato

3° Stato

In questa pagina è riportata per prima cosa il titolo, poi la scadenza della tesina (cioè

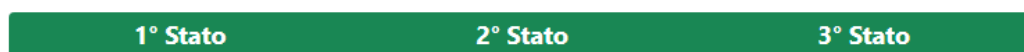
entro quanti mesi e giorni dovrà essere completata la tesina), l'account assegnato, il testo della tesina e lo stato di avanzamento. Lo stato di avanzamento indica a che punto ci si trova con la tesina, maggiore lo stato di avanzamento e più vicini al completamento ci si trova. L'account assegnato è l'username di un account di un servizio esterno a BRIDGE, per ulteriori dettagli bisogna fare riferimento al docente. Anche per la password dell'account assegnato bisogna parlare col docente. Lo stato di avanzamento è indicato da una barra di progresso che qui è completamente grigia in quanto la tesina è appena cominciata, per avanzare di stato bisogna parlare col docente, è lui infatti ad avanzare di stato la tesina, una tesina col primo stato completato appare così:

Questa barra indica lo stato di avanzamento della tesina:



Ogni tesina può avere un numero di stati diversi, ma quando la tesina è terminata appare comunque interamente riempita di verde:

Questa barra indica lo stato di avanzamento della tesina:



5.2 Tutor

5.2.1 Primo utilizzo

Se non si è mai usato il sistema prima d'ora bisogna chiedere ad un admin di farsi registrare come tutor dando un username ed una password da usare poi per il login. Inoltre, dopo che l'account è stato registrato, per essere utilizzabile è necessario che venga inserito come tutor di un corso dall'admin.

5.2.2 Utilizzo

Per prima cosa fare il login con username e password forniti dall'admin, una volta fatto ciò si avrà accesso alle funzione del tutor.

Il tutor può fare alcuni dei compiti del docente, può correggere gli esercizi (e controllare il plagio fra gli studenti), assegnare tesine ad 1 o più studenti e modificare lo stato di avanzamento di una tesina, lasciando anche dei commenti volendo.

Se si vuole si possono vedere alcuni dati su ogni corso assegnato andando in "Riepilogo corso" e scegliendo il corso. Nella pagina che si apre si possono vedere tutti gli studenti iscritti a quel corso, il numero di esercizi svolti e quelli accettati per ogni studente.

Per la correzione degli esercizi bisogna andare dal menù in "Correggi esercizi (e plagio)" e scegliere il corso. Una volta scelto si ha davanti una tabella con tutti gli esercizi consegnati dallo studente, in cima si trovano quelli non ancora corretti, ordinati

dal più vecchio al più recente. Lo stato di correzione di ogni esercizio è riportato sulla colonna "Stato". Per ogni esercizio ci sono 2 pulsanti: "Correggi" (o "Correggi ancora" se l'esercizio è già stato corretto) e "plagio", il secondo pulsante compare solo se il corso ha il controllo del plagio attivo. Si osserva ora il comportamento del sistema premendo i 2 pulsanti.

Premendo il pulsante "**Correggi**" si raggiunge questa pagina:

Correggi esercizio (Linguaggi per il web 2020-2021)

Se vuoi rimuovere un commento togli tutto il suo testo e sarà rimosso

Questo esercizio è stato consegnato 1 volta

Matricola	Nome	Cognome	File	Testo
1854835	Matteo	Simonetti	File	<button>Testo</button>

Commento versione 1, 16/04/2021 00:00

vedi issue, ma non serve risottomettere qui

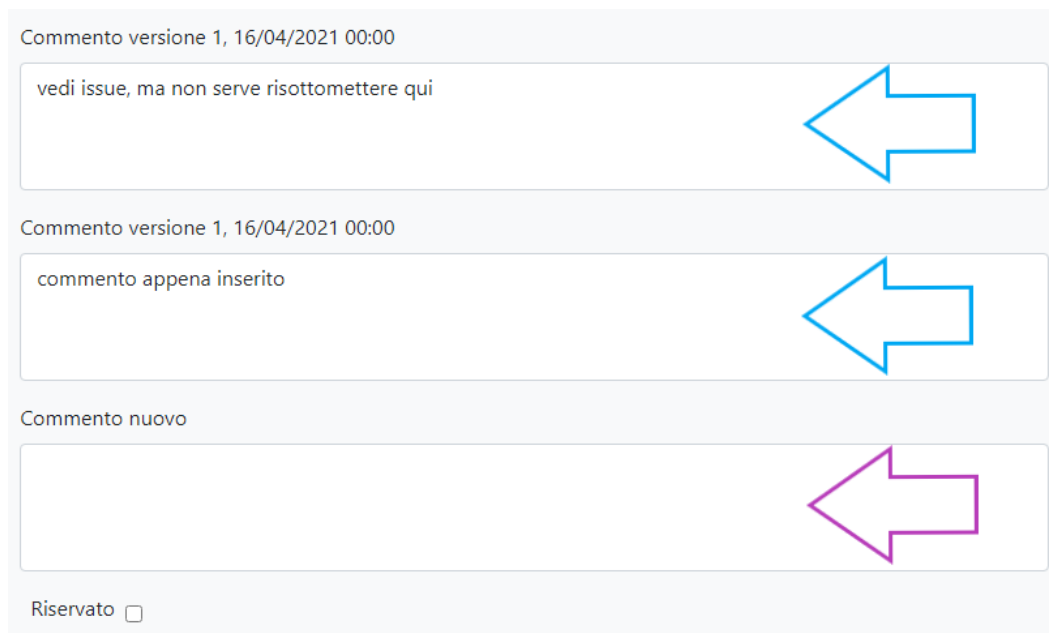
Commento nuovo

Riservato ☐

Accetta

Da rivedere

Qui ci sono i dettagli dell'esercizio, cliccando sul link blu chiamato "File" si può scaricare il file sottomesso dallo studente, cliccando su "Testo" si avrà accesso al testo dell'esercizio. Sopra ai dati dell'utente compare scritto anche quante volte l'utente ha riconsegnato quell'esercizio. Una volta controllato l'esercizio si può lasciare un commento, con la freccia azzurra è stato indicato un commento lasciato in precedenza. Se fosse la prima volta che l'esercizio viene corretto, la sezione indicata dalla freccia azzurra non sarebbe presente, ci sarebbe solo quella indicata dalla freccia viola. La freccia viola indica appunto il nuovo commento, ossia una sezione di testo dentro la quale si può scrivere per inserire un commento nuovo che rimarrà collegato all'esercizio e sarà visibile allo studente. Se venisse inserito un commento e si rientrasse nella pagina di correzione, questo è quello che comparirebbe nella zona dei commenti:



Commento versione 1, 16/04/2021 00:00

vedi issue, ma non serve risottomettere qui

Commento versione 1, 16/04/2021 00:00

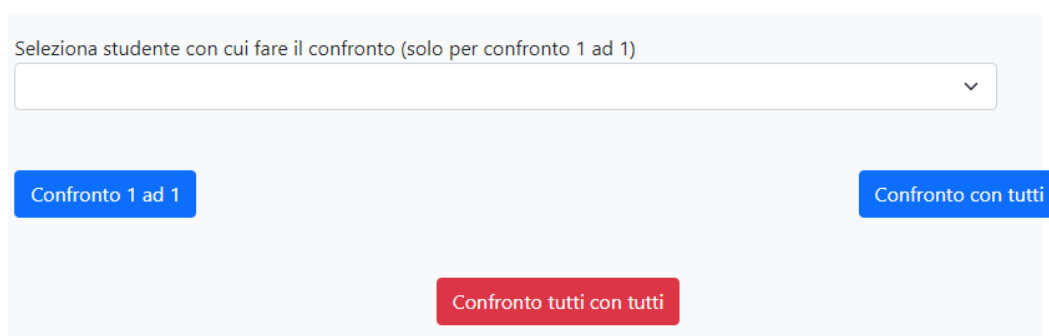
commento appena inserito

Commento nuovo

Riservato ☐

Si può notare che ora c'è un secondo commento "vecchio", la data ed ora del commento è la stessa del precedente in quanto si riferisce alla data ed ora in cui è stato consegnato l'esercizio, non alla data ed ora in cui è stato lasciato il commento. Per inserire un commento privato basta mettere una spunta su "Riservato". Si possono inoltre modificare i vecchi commenti (freccie azzurre) modificando il testo. Quindi si può aggiungere un nuovo commento, modificare i vecchi, oppure entrambe le cose. Per confermare queste modifiche all'esercizio è necessario premere uno dei 2 pulsanti in fondo (si vedono nella prima foto), il primo dà all'esercizio un esito positivo, il secondo no. Si può anche dare solo l'esito senza inserire commenti, premendo semplicemente il pulsante desiderato.

Se invece si fosse premuto il tasto "**plagio**":



Seleziona studente con cui fare il confronto (solo per confronto 1 ad 1)

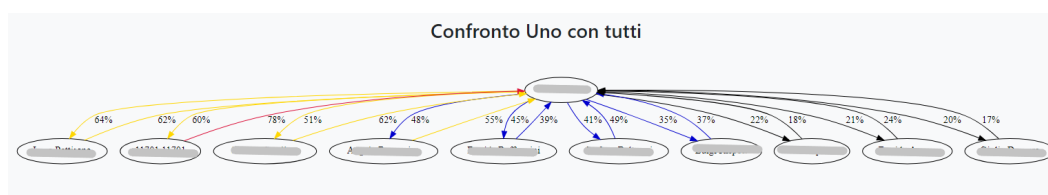
Confronto 1 ad 1

Confronto con tutti

Confronto tutti con tutti

Le opzioni sono 3: si può fare un confronto con un solo studente selezionando lo studente dal menù a tendina e premendo il primo pulsante, ciò porta ad una pagina con i due file consegnati messi a confronto con accanto due percentuali di plagio, ognuna indica quanto uno ha copiato dall'altro. Magari studente1 ha scritto 10 righe copiate da studente2, mentre studente2 ha scritto 15 righe, in tal caso risulterebbe un plagio del 100% di studente1 verso studente2 ma non il contrario. Il secondo

pulsante confronta lo studente selezionato con tutti gli altri che hanno fatto lo stesso esercizio, ciò genera un grafo orientato. Sopra agli archi del grafo sono presenti le % di plagio, i nodi rappresentano gli studenti. Il terzo pulsante confronta tutti gli studenti con tutti gli studenti, a prescindere da chi si era selezionato inizialmente, il pulsante è in rosso in quanto un confronto simile necessita di molte risorse e potrebbe causare grossi rallentamenti al sistema. Qui sotto si vede il risultato di un confronto uno a tutti (secondo pulsante) è il seguente:



I nomi dentro gli ovali della foto sono nascosti. Le frecce hanno un colore basato sulla percentuale di plagio, nero sotto il 30%, blu fra il 30% ed il 50%, arancione fra il 50% ed il 70% e rosso oltre il 70%.

Se si vuole **assegnare una tesina** a degli studenti, oppure modificare un'assegnazione già fatta, lo si può fare tramite "Assegna tesine", dopo aver scelto il corso si potrà scegliere da una tabella con tutte le tesine assegnabili e anche quelle già assegnate:

Filtra:

Titolo	Proposta da studente?	Assegnata a	Testo	Seleziona
FIFA Club World Cup	Si	-	Testo	Assegna
ByteCourier2	Si	Angelica Della Vecchia 1746294 Simone Orelli 1749732	Testo	Modifica assegnazione

Qui ci sono 2 tesine, la prima non assegnata e la seconda assegnata. Sopra alla lista di tesine è presente un campo per filtrare le tesine inserendo un qualunque parametro della tabella. Se si vuole fare l'assegnazione di una tesina si preme il pulsante blu e si accede ad una pagina simile:

Assegna tesine (Linguaggi per il web 2020-2021)

Seleziona gli studenti a cui vuoi assegnare la tesina (minimo 1, massimo 2)

<input type="text"/>	<input type="text"/>
<input type="text"/>	<input type="text"/>

Il numero delle righe dipende dal numero massimo di studenti per tesina del corso. Ogni riga ha 2 campi, il più grande permette di selezionare lo studente per la tesina, il più piccolo permette di assegnargli un account (facoltativo). Si può inserire un solo studente o si possono inserire anche uno per ogni riga (arrivando quindi al massimo consentito per il corso).

Se invece si vuole fare la **modifica di un'assegnazione**, si preme il pulsante giallo e si accede ad una pagina simile a quella di assegnazione, con alcune differenze. La prima differenza è che alcuni campi sono già riempiti dalle assegnazioni effettuate, e la seconda è che si può eliminare l'assegnazione col pulsante "Elimina" per rendere la tesina utilizzabile per altri studenti. L'ultima differenza è che nella modifica di assegnazione si può cambiare la data di scadenza della tesina per dare magari più tempo agli studenti di terminare il lavoro.

Se si vogliono modificare gli **stati di avanzamento** di una tesina, o fare dei commenti sulla tesina, si va nella voce del menù corrispondente, si sceglie il corso, e si seleziona la tesina su cui fare la modifica. A questo punto si possono vedere tutti i dettagli della tesina, si possono lasciare commenti e si può avanzare di stato.

Titolo	Scadenza
ForumLW	Nessuna

Testo:

vedi scambio di email. Vale l'ultima versione approvata dal docente

Questa barra indica lo stato di avanzamento della tesina:

1° Stato	2° Stato	3° Stato

Commento nuovo

Riservato ☐

Avanza al prossimo stato

Rimani allo stato attuale

Il sistema di commenti è lo stesso della correzione degli esercizi, solo che qui i pulsanti non sono "Accetta" o "Da rivedere" ma sono "Avanza al prossimo stato" e "Rimani allo stato attuale". Ovviamente se si vuole avanzare di stato si usa il primo pulsante, altrimenti si usa il secondo per confermare i commenti che sono stati scritti, ma senza avanzare di stato.

5.3 Docente

Tutte le funzioni del tutor sono anche funzioni del docente, quindi quelle funzioni non sono riportate in questo manuale, si consiglia però di leggere prima questo manuale e poi quello del tutor.

5.3.1 Primo utilizzo

Prima di iniziare con la specifica è bene fare alcune precisazioni per comprendere al meglio questo sistema:

- 1) I "Tipi di corso" sono la generalizzazione di un corso, per esempio un corso può essere Linguaggi per il Web 2021-2022, il tipo di corso corrispondente è Linguaggi per il Web.
- 2) I tipi di esercizio sono la generalizzazione degli esercizi, per esempio creare un

programma che sommi due numeri interi è un tipo di esercizio, quando poi uno studente svolge quel tipo di esercizio allora si tratta di un esercizio effettivo. Un tipo di esercizio vale per tutti i corsi del tipo di corso a cui appartiene. Idem per tesine e tipi di tesina.

Se non si è mai usato il sistema prima d'ora bisogna chiedere ad un admin di farsi registrare come docente, dando un username ed una password con cui poi fare il login. Inoltre bisogna farsi creare una tipologia di corso sempre dall'admin. Le informazioni da dare all'admin sono le seguenti: nome del tipo di corso, eventuali tutor, tipologia di voti per gli esercizi (si possono avere voti in decimale, in trentesimi o non avere voti ma solo accettare o rifiutare gli esercizi), si può richiedere il software antiplagio sim [22] (per esercizi in linguaggio C), sito del corso, categorie degli esercizi ed estensione dei file che devono essere consegnati dagli studenti (una o più estensioni). Se si vogliono utilizzare anche le tesine bisogna fornire anche il numero massimo di studenti assegnabili per ogni tesina, il numero di stati delle tesine. Inoltre, se si vuole fornire agli studenti degli account su cui caricare i propri lavori, bisogna dare all'admin anche il prefisso degli account ed il numero totale di account disponibili. Infine bisogna anche comunicare se si vuole che gli studenti abbiano la possibilità di autoassegnarsi la tesina tramite il sistema, senza doverla richiedere al docente. L'autoassegnazione permette allo studente di prendere una tesina, solo dopo che gli è stato accettato un esercizio obbligatorio per ogni categoria dal docente.

5.3.2 Utilizzo

Il compito principale del docente è di creare tipi di esercizio e tipi di tesina per i propri corsi e tenere traccia dello stato delle tesine e correggere gli esercizi degli studenti.

Se non si è ancora creato un corso, la prima cosa che va fatta è proprio **creare un corso**, per fare ciò si clicca dal menù "Crea corso", si inserisce il tipo di corso, l'anno di svolgimento del corso (es. 2021-2022) e le istruzioni specifiche per il corso. Le istruzioni appariranno allo studente nella sua pagina di consegna degli esercizi. Se si vogliono modificare le istruzioni di un corso si può andare in "Modifica corso" e dopo aver scelto il corso desiderato si può fare la modifica.

Per **archiviare un corso** o per riattivare un corso archiviato si va in "Archivia corso", ci sono 2 liste, la prima permette di archiviare i propri corsi, la seconda di riattivare i corsi archiviati, una volta che un corso è archiviato esso risulta irraggiungibile per chiunque, fino a che non viene riattivato.

Se si vuole **eliminare un corso** in maniera definitiva bisogna per prima cosa archivarlo, una volta fatto ciò si procede dalla pagina "Elimina corso", e si preme il pulsante per l'eliminazione.

Se si vuole **creare un tipo di esercizio**, si usa la voce corrispondente del menù e si sceglie il tipo di corso per cui creare l'esercizio, questo esercizio compare per tutti i corsi di quel tipo. La prima cosa da fare è selezionare la categoria dell'esercizio dal menù a tendina, poi inserire il testo ed infine se l'esercizio è "obbligatorio" fare un clic per dare la spunta. Per modificarlo invece bisogna andare in "Gestisci tipi esercizio" e sostituire quel che si desidera modificare, i parametri da inserire sono gli stessi della creazione del tipo di esercizio.

Se si vuole **creare un tipo di tesina** si può andare nella pagina corrispondente,

selezionare il tipo di corso e poi inserire titolo, testo, e mesi di scadenza, inoltre si può inserire volendo un corso specifico per il quale la tesina sia valida, in tal caso la tesina verrebbe visualizzata solo per quel corso e non per tutti i corsi del tipo di corso selezionato. I mesi di scadenza indicano dopo quanti mesi dall'assegnazione la tesina risulterà scaduta. Per **modificare un tipo di tesina**, e quindi modificare i parametri dati in creazione bisogna andare "Gestisci tipi tesina" e modificare quel che si desidera. I parametri sono gli stessi della creazione.

Siccome per le tesine possono essere assegnati degli account, quando inizia un nuovo anno, e quindi un nuovo corso, risulta utile **azzerare le assegnazioni degli account** per poterli assegnare ai nuovi studenti, per fare ciò c'è la pagina "Reset account tesine" all'interno della quale si seleziona un tipo di corso dal menù a tendina e si preme reset per liberare i relativi account.

Se si vogliono vedere i **corsi frequentati dagli studenti** si clicca la relativa pagina dal menù. Qui si ha una tabella in cui ogni riga contiene nome e cognome di uno studente ed un corso a cui è iscritto, se lo si vuole disiscrivere da quel corso basta premere il pulsante di disiscrizione su quella riga. Ovviamente si possono disiscrivere dai corsi solo studenti iscritti a corsi assegnati al docente

Infine si possono **gestire gli utenti**, selezionando la voce dal menù, si ha una lista di tutti gli studenti presenti nel sistema. Selezionandone uno si possono cambiare i dettagli anagrafici, il permesso (si può trasformare uno studente in un tutor e viceversa), la mail e la password, questo risulta utile soprattutto per permettere ad uno studente che si è dimenticato la password di accedere di nuovo al sistema.

5.4 Admin

5.4.1 Primo utilizzo

Prima di iniziare con la specifica è bene fare una precisazione per comprendere al meglio questo sistema:

1) I "Tipi di corso" sono la generalizzazione di un corso, per esempio un corso può essere Linguaggi per il Web 2021-2022, il tipo di corso corrispondente è Linguaggi per il Web.

5.4.2 Utilizzo

Se un docente volesse iniziare un suo tipo di corso la prima cosa da fare è **creare un docente** nel sistema, lo si fa scegliendo la voce "Crea docente" dal menù. Si inserisce l'anagrafica ed username e password scelti dal docente. Inoltre se fossero necessari dei tutor bisogna crearli dalla voce del menù "Crea tutor" facendo lo stesso procedimento usato per creare il docente. Fatto ciò si **crea il tipo di corso** dalla corrispondente voce del menù. Per crearlo bisogna farsi dare dal docente tutte le informazioni necessarie che vengono richieste nella creazione. Alcuni campi sono nascosti, infatti se il numero di studenti massimi per la tesina viene riempito (e quindi ci sono tesine) appariranno altri campi da riempire secondo quanto richiesto dal docente. Si possono scegliere un numero di tutor a piacere: nessuno, uno o più di uno; per selezionarne più di uno basta premere il tasto ctrl e selezionare i

tutor desiderati. Per quanto riguarda le categorie: bisogna inserire (separate da uno spazio) le categorie dei tipi di esercizio che il docente andrà poi ad inserire. Per quanto riguarda le estensioni: bisogna inserire (separate da uno spazio) le estensione accettate per i file degli esercizi consegnati dagli studenti. Se dopo la creazione ci fosse la necessità di **modificare il tipo corso**, ossia di modificare qualcuno dei parametri inseriti in creazione, si può andare in "Gestisci tipologia di corso", selezionare la tipologia ed applicare le modifiche necessarie. L'unico campo che è meglio non modificare è quello delle categorie, poichè se sono stati consegnati esercizi dagli studenti di una categoria che si va magari a togliere, ci sarebbero delle incoerenze nel sistema.

Per **archiviare un corso** o per riattivare un corso archiviato si va in "Archivia corso", ci sono 2 liste, la prima permette di archiviare i corsi, la seconda di riattivare i corsi archiviati, una volta che un corso è archiviato esso risulta irraggiungibile per chiunque, fino a che non viene riattivato.

Se si vuole **eliminare un corso** in maniera definitiva bisogna per prima cosa archivarlo, una volta fatto ciò si procede dalla pagina "Elimina corso", e si preme il pulsante per l'eliminazione.

Se si vogliono vedere i **corsi frequentati dagli studenti** si clicca la relativa pagina dal menù. Qui si ha una tabella in cui ogni riga contiene nome e cognome di uno studente ed un corso a cui è iscritto, se lo si vuole disiscrivere da quel corso basta premere il pulsante di disiscrizione su quella riga. Infine si possono **gestire gli utenti**, selezionando la voce dal menù, si ha una lista di tutti gli studenti presenti nel sistema. Selezionandone uno si possono cambiare i dettagli anagrafici, il permesso, la mail e la password, questo risulta utile soprattutto per permettere ad uno studente che si è dimenticato la password di accedere di nuovo al sistema. Infine, se si vuole modificare il **messaggio nella home** dello studente lo si può fare dalla relativa voce del menù. Questo messaggio dovrebbe contenere delle istruzioni base per permettere allo studente di orientarsi, come un piccolo manuale utente. Se si vuole che un testo sia in grassetto si fa nel seguente modo: testo.

Capitolo 6

Sperimentazione

Il sistema è stato caricato ed installato su un server per permettere a 10 studenti di testarlo. Gli studenti, 6 ragazzi e 4 ragazze, sono tutti laureandi o laureati in Ingegneria dell'informazione presso la sede di Latina della Sapienza e sono tutti residenti nel Lazio. L'età media si aggira intorno ai 23 anni. Tutti hanno utilizzato la precedente versione di BRIDGE, sapevano quindi già quale fosse lo scopo del sistema che hanno testato.

6.1 Sperimentazione effettiva

Sono state create 15 attività da eseguire, modellate a partire dai servizi disponibili per lo studente. Alcune delle attività erano: la registrazione al sistema, la consegna di un determinato esercizio, la visualizzazione dei commenti a quell'esercizio, la riconsegna etc.

A questo punto per ogni studente è stato definito un username ed un insieme di attività da eseguire. Le attività sono state assegnate in modo tale da creare la maggior diversificazione possibile, facendo in modo che ognuno si ritrovasse con un insieme diverso di cose da fare, così da avere un uso del sistema completo. Ogni studente è stato contattato e gli è stata fornita la sua lista di attività da svolgere ed il sito su cui andare per usare il sistema. Ogni studente ha svolto le sue attività, aspettando se necessario le correzioni degli esercizi da parte del docente, ed una volta terminate ha riempito un questionario SUS per fornire la sua opinione sull'usabilità del sistema. Il questionario è stato inserito su Google Forms, in modo tale da poter raccogliere i risultati in un file .csv e poterli gestire facilmente.

6.2 SUS

Il System Usability Scale (SUS) [16] è un test, distribuito liberamente e molto utilizzato, in quanto riconosciuto da numerose fonti, che consiste in 10 affermazioni alle quali si deve dare un voto da 1 a 5, dove 1 indica che si è molto in disaccordo, 2 in disaccordo, 3 neutrale, 4 d'accordo e 5 molto d'accordo. Il questionario SUS risulta fra i migliori in termini di rapidità, essendo composto da sole 10 domande. Il questionario in origine veniva condotto verbalmente, con l'ampio uso che ne è stato fatto è passato poi in forma scritta, ci sono stati però dei problemi, per chi

non parla inglese, a comprendere a pieno la traduzione fornita nella propria lingua [17]. Una delle fonti più riconosciute in merito, ossia l'International journal of human-computer interaction [18], ha svolto una ricerca all'interno della quale si fornivano delle traduzioni in lingue estere e se ne misurava l'attendibilità, dalla ricerca sono uscite varie traduzioni attendibili di SUS, nessuna di queste però è in italiano. In genere per la nostra lingua viene utilizzata quindi la traduzione fornita da CognitiveLab [19], che è quella usata anche per questa tesi.

Il test produce come risultato un valore fra 0 e 100. In generale un punteggio superiore al 68 viene considerato sopra la media. Per calcolare il risultato la procedura è la seguente:

1. Se i voti sono "Molto in disaccordo", [...], "Molto d'accordo" bisogna convertirli da testo a numeri, 1 per il parere peggiore e 5 per il parere migliore.
2. Si calcola il punteggio con la seguente formula:

$$\text{PUNTEGGIO} = 2.5(20 + \text{SUM}(\text{SUS01}, \text{SUS03}, \text{SUS05}, \text{SUS07}, \text{SUS09}) - \text{SUM}(\text{SUS02}, \text{SUS04}, \text{SUS06}, \text{SUS08}, \text{SUS10}))$$

Sove SUS01 indica il voto dato alla prima affermazione, lo stesso per 02, 03 etc...

3. Si fa la media fra i risultati ottenuti dai singoli questionari per ottenere il punteggio globale di usabilità.

Il punteggio può poi essere utilizzato per confrontare l'usabilità di un sistema con un altro, ed è quindi un modo oggettivo per descrivere la qualità di un sistema, a prescindere dal tipo di sistema. Questa "monodimensionalità" può essere un vantaggio o uno svantaggio. Può essere uno svantaggio poichè comparare l'usabilità di sistemi completamente diversi è un po' come "confrontare le mele con le pere". Il vantaggio sta invece, come già detto, nei costi praticamente nulli e nella grande rapidità.

6.3 Risultati

Tutti gli studenti hanno compilato il questionario, non è stato necessario convertire i voti in quanto il questionario era già in una scala da 1 a 5. Sono state fatte tutte le operazioni matematiche per ottenere il punteggio finale per ogni studente e poi la media dei punteggi e questo sistema ha ottenuto 87.75, il punteggio risulta quindi ampiamente sopra la media. Sicuramente influisce nel risultato la somiglianza del sistema con quello precedentemente utilizzato, questa familiarità quindi permette al punteggio di salire. Essendo però il punteggio ben al di sopra della media, si può affermare che il sistema ha una buona usabilità, si può anche dare un "voto" al sistema ricorrendo alla seguente tabella [20]:

SUS score range	Grade	Percentile range
84.1–100	A+	96–100
80.8–84.0	A	90–95
78.9–80.7	A–	85–89
77.2–78.8	B+	80–84
74.1–77.1	B	70–79
72.6–74.0	B–	65–69
71.1–72.5	C+	60–64
65.0–71.0	C	41–59
62.7–64.9	C–	35–40
51.7–62.6	D	15–34
0.0–51.6	F	0–14

Il sistema ottiene quindi come voto A+, voto che viene fornito solo al miglior 4% dei sistemi. Come già detto c'è da tenere conto dell'influenza data dell'uso pregresso di BRIDGE. Si sottolinea inoltre che durante la sperimentazione nessuno studente ha riportato difficoltà al docente o al tesista riguardanti l'impossibilità di proseguire con le attività. L'esito finale è quindi molto positivo.

Capitolo 7

Esperienza complessiva e conclusioni

Nel realizzare il primo progetto di dimensioni abbastanza grandi, non possono non verificarsi problemi e necessariamente si deve imparare qualcosa. Sono state realizzate in precedenza altre applicazioni web più piccole per uso personale o per uso professionale, ma il livello di difficoltà non era eguagliabile. Affrontare un progetto così grande, da solo, è stata una sfida non indifferente.

Questo progetto è risultato necessario al fine di svecchiare BRIDGE, che si basava ancora su PHP 5 e che aveva un tipo di interfaccia e di esperienza utente ormai datate. È stato quindi definito, e successivamente implementato, un progetto che abbiamo riportato nel capitolo 4. Il sistema è stato anche sottoposto ad una sperimentazione che ha visto partecipi studenti che hanno utilizzato la precedente versione di BRIDGE, e che quindi hanno familiarità con essa. La sperimentazione ha fornito risultati positivi descritti nel capitolo 6. In questo capitolo conclusivo si descrive in generale l'esperienza derivata dallo sviluppo di questa Tesi. In primo luogo, la riprogettazione che è stata necessaria durante lo sviluppo del programma, dato che strada facendo emergevano dettagli ignorati; in secondo luogo si parla della capacità di sviluppare il software con la mentalità del fruitore, e non del creatore. Si affronta poi il problema del trasferimento dei dati da un sistema con storage basato su XML ad uno basato su SQL, e di come sia stato necessario utilizzare il vecchio sistema in generale per costruire il nuovo. Gli ultimi argomenti riguardano la gestione del tempo e l'utilità dei commenti al codice.

7.1 Riprogettazione in itinere

Prima di iniziare a lavorare al progetto sono stati scritti in maniera informale tutti i casi d'uso, per avere un quadro generale di cosa si sarebbe dovuto costruire; fatto ciò, sono stati definiti gli "oggetti" con cui si sarebbe dovuto lavorare (come i corsi, esercizi ecc) e da lì è stato progettato il database. La progettazione, però, essendo stata sviluppata da un solo individuo, è passibile di piccoli errori. Per esempio, è capitato in alcune situazioni di non considerare dei casi limite, delle eccezioni, degli stati vuoti e di trovarsi di conseguenza, durante la realizzazione, con degli errori nella logica del codice, che sono stati corretti dunque durante la fase di riprogettazione.

di quel che era stato progettato inizialmente. Per fare ciò è stato anche necessario quindi andare ad applicare queste modifiche a tutto ciò che era stato costruito fino a quel momento. Un esempio pratico è dato dall'archiviazione dei corsi: l'idea iniziale era quella di archiviare i corsi nel seguente modo (in poche parole): prendere tutto ciò che riguarda quel corso, quindi esercizi tesine tipi di esercizio etc, e spostarli in delle tabelle tipo `eserciziArchivio`, `tesineArchivio` etc., durante la realizzazione, però, è stata pensata una soluzione molto più semplice, ossia avere un campo "archiviato" per ogni corso: ciò ha portato a dover modificare il database e a dover rivedere ogni pagina costruita finora, facendo, per esempio, in modo che quando si selezionano i corsi vengano mostrati solo quelli con `archiviato=0`. Una cosa quindi che è stata imparata è che dedicare maggior tempo alla progettazione, in ogni suo piccolo dettaglio, è assolutamente utile per risparmiare tempo in fase di realizzazione.

7.2 Mindset dell'utilizzatore

Una grande sfida è stata quella di uscire dal mindset del programmatore ed entrare nel mindset del fruitore dell'applicazione. Ciò è stato fondamentale per evitare di creare un'applicazione con ridotta usabilità. È stato ovviamente un lavoro prettamente di front-end, un occhio particolare è stato infatti dato alla user experience. Inizialmente, per esempio, i testi degli esercizi erano mostrati all'interno della tabella, non essendo nascosti occupavano una quantità molto importante di schermo, e quindi una pagina che mostrava 5-6 esercizi riempiva 3-4 volte in altezza lo schermo di un portatile. Questa cosa non è accettabile per un'interfaccia chiara, però quella era l'idea iniziale, data dal non aver considerato la dimensione che i testi avrebbero poi raggiunto. Ora i testi sono stati nascosti ed si possono vedere premendo il relativo pulsante, così da non occupare spazio inutilmente.

In questa situazione è stato necessario uscire dalla mentalità secondo la quale basta che un sistema funzioni per essere ottimale, cosa che può facilmente accadere ad uno sviluppatore non ancora esperto. È stato necessario entrare nella mentalità del ragazzo di primo anno di università che ha i suoi dubbi e che, trovandosi davanti ad una pagina piena per gran parte di testi di esercizi, si sarebbe sentito sopraffatto e demotivato. Lo stesso stile di pensiero è stato messo in atto per gli stati degli esercizi, che inizialmente erano stati pensati come: inviato, accettato, rifiutato. Sono infatti diventati: inviato, accettato, da rivedere. Il colore iniziale del rifiuto era rosso, nella versione finale è arancione. Queste 2 modifiche sono importanti in quanto, pur esprimendo lo stesso concetto, lo rendono più incoraggiante e meno d'impatto.

7.3 Uso del vecchio sistema su virtual machine

Risulta abbastanza ovvio che per realizzare molte parti del sistema è stato utile vedere come era strutturato il vecchio sistema, in quanto probabilmente i dubbi venuti durante lo sviluppo di una versione si sarebbero ripresentati pure nell'altra. Il problema qui risultava nel fatto che il vecchio sistema usava PHP 5 [10], qui invece si è usato PHP 8. In un ambiente con PHP 8 il vecchio sistema dava problemi e quindi non si potevano utilizzare entrambi i sistemi nello stesso ambiente. La soluzione che è stata usata è stata quella di creare una virtual machine (Oracle VM VirtualBox

[4]) dove installare PHP 5 ed usare la vecchia webapp. Inoltre la virtual machine era su linux, così da poter fare testing degli script di backup e restore creati per il sistema e anche per testare il sistema antiplagio tramite `shell_exec()`

7.4 Trasferimento dati dal vecchio sistema

Un altro punto impegnativo durante il lavoro è stato il dover trasferire tutti i dati dal vecchio sistema al nuovo. Il vecchio sistema memorizzava i dati su XML, la struttura dati era completamente diversa, mentre il nuovo sistema si basa su SQL. Il primo problema si è presentato nel trovare il modo di trasferire i dati. La prima ipotesi ingenua è stata di trasferire i dati a mano, svanita non appena notata la quantità di esercizi presenti nel vecchio sistema (375). La seconda ipotesi è stata ovviamente di scrivere uno script che interagisse col file system del computer e selezionasse dai file XML le informazioni necessarie sotto forma di stringhe, modificasse le stringhe secondo il formato con il quale il nuovo sistema salvava quei dati, ed infine facesse le query necessarie ad inserire questi dati manipolati all'interno del db. L'idea era giusta ma la realtà era leggermente diversa: infatti, sebbene lo script fosse corretto, non funzionava. Il problema si è rintracciato nelle incoerenze contenute nei dati del primo sistema. C'erano infatti matricole e mail presenti due o più volte nel vecchio sistema il quale, evidentemente, non implementava un controllo durante la registrazione.

Risolti questi conflitti sullo storage degli esercizi svolti, è stato eseguito il trasferimento dei file degli esercizi. Per fare ciò non bastava spostare i file in quanto la nomenclatura dei file doveva essere specifica: il nuovo sistema prevede infatti un nome per il file del tipo: `categoria.cognome.nome.matricola.idTipoEsercizio.estensione`; il vecchio sistema invece conteneva molte meno info nel nome, quindi anche qui è stato fatto uno script che doveva prendere i dati vecchi, manipolarli e, tramite le info nel db, creare i nomi finali. I problemi anche qui si sono presentati nel momento in cui per alcuni utenti mancavano degli esercizi, per altri ce n'erano troppi e così via.

7.5 Gestione del tempo

Questo probabilmente è uno dei punti su cui a priori c'è stato un errore maggiore. Infatti, la realizzazione di questo sistema ha richiesto due mesi, dedicandoci quasi ogni giorno diverse ore. Qui si è andata a scontrare la velocità stimata di scrittura del codice contro quella reale: spesso ci sono state giornate in cui, dopo aver scritto 50-100 righe di codice, sorgeva un problema, e quel problema richiedeva tutto il resto della giornata per essere risolto. Un esempio di questo si trova nella sezione precedente, infatti diversi sono stati gli imprevisti nel trasferimento dei vecchi dati. Come già detto, i progetti finora seguiti erano solo progetti personali o progetti relativi a degli esami, in cui tutto è talmente piccolo da essere sempre sotto l'occhio di chi progetta e costruisce. Qui per la prima volta, rileggendo del codice, la sensazione è stata di vedere un qualcosa scritto da qualcun altro. Questo ovviamente ha come conseguenza che gli imprevisti tendano ad aumentare, non avendo più il quadro generale al 100% basta una distrazione e sorgono errori. Importantissimo in questo aspetto sono stati infatti i commenti:

7.6 Commenti al codice

Uno dei consigli sempre ripetuti da chi ha più esperienza è quello di commentare il codice, e non di commentarlo tanto per farlo, ma di commentarlo come se il prossimo giorno dovesse leggerlo qualcuno che non sa nulla del tuo stile di scrittura del codice o di che tipo di idea ti è venuta in mente per risolvere un problema particolarmente complesso. Per fortuna, conscio della grandezza del progetto, il consiglio è stato seguito alla lettera ed è stato molto utile: è stato sempre fatto testing del progetto un po' alla volta, ossia ogni volta che una nuova funzione veniva creata veniva poi subito testata. Non sempre però si riescono a trovare tutti i problemi in una sola fase di testing. Proprio in questi casi i commenti sono stati fondamentali. Difatti, quando in una seconda fase di testing uscivano nuovi bug, la correzione faceva largo affidamento sui commenti per capire come funzionasse una determinata pagina. Sono stati utili soprattutto i commenti per le query complesse al database, le quali, se non commentate, richiedono diversi minuti per essere comprese.

Un altro aspetto importante dei commenti è stato per il riutilizzo di alcune parti del codice: quando è stato necessario implementare in una pagina qualcosa di già implementato in una precedente, è bastato cercare il commento invece che il codice.

7.7 Repository

In conclusione si lascia il link alla repository del progetto, privato dei dati degli studenti importati dal vecchio sistema: <https://github.com/anonimo345423/BRIDGEreloaded>

Bibliografia

- [1] eBay. 2022. <https://ebay.com/>
- [2] W3Techs. "Usage statistics and market share of Bootstrap for websites". 2022. <https://w3techs.com/technologies/details/js-bootstrap>
- [3] Kinneret Yifrah. "UX Microcopy. Manuale pratico per UX writer, designer e architetti dell'informazione". Flacowski. 2021.
- [4] VirtualBox. 2022. <https://www.virtualbox.org/>
- [5] W3C. 2022. <https://www.w3.org>
- [6] Bootstrap. 2022. <https://getbootstrap.com/>
- [7] JavaScript. 2022. <https://www.javascript.com/>
- [8] W3Techs. "Usage statistics of JavaScript as client-side programming language on websites". 2022. <https://w3techs.com/technologies/details/cp-javascript>
- [9] jQuery. 2022. <https://jquery.com/>
- [10] PHP. 2022. <https://www.php.net/>
- [11] W3Techs.com. "Usage statistics of PHP for websites". 2021 https://w3techs.com/technologies/overview/programming_language
- [12] MariaDB. 2022. <https://mariadb.org/>
- [13] MySQL. 2022. <https://www.mysql.com/>
- [14] Apache. 2022. <https://httpd.apache.org/>
- [15] Vlastimil Klima. IACR. "Tunnels in Hash Functions: MD5 Collisions Within a Minute". 2006. <https://eprint.iacr.org/2006/105>
- [16] Wikipedia. "SUS (System Usability Scale)". 2022. https://en.wikipedia.org/wiki/System_usability_scale
- [17] Kraig Finstad. "The System Usability Scale and Non-Native English Speakers". Journal of usability studies. 2006.

- [18] Researchgate. "Multi-Language Toolkit for the System Usability Scale". International Journal of Human-Computer Interaction. 2020. https://www.researchgate.net/publication/343766416_Multi-Language_Toolkit_for_the_System_Usability_Scale
- [19] Questionario SUS (System Usability Scale) <https://www.cognitivelab.it/wp-content/uploads/2011/07/SUS-scheda.pdf>
- [20] Chuniversiteit. "How to use the System Usability Scale (SUS) in 2021". 2021 <https://chuniversiteit.nl/papers/sus-past-present-future>
- [21] Graphviz. 2022. <https://graphviz.org/>
- [22] Dick grune. "sim". 2021. https://dickgrune.com/Programs/similarity_tester/