

Diario di lavoro

Luogo	Mendrisio
Data	11.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto svolto tutta la fase di Analisi del progetto. Durante la prima mezz'ora della giornata ho letto il quaderno dei compiti e ho scritto delle domande che successivamente sono state poste al mio formatore. Le domande e risposte sono le seguenti:

La pagina di contatti che persone deve contenere?

Direttore, presidente, vicedirettore, responsabile amministrativo, segretariato, formatori (più di 1)
Potrebbe anche essere gestibile dall'utente, una semplice pagina in cui posso inserire i miei contenuti e formattarli in maniera basica (ci sono diverse librerie). Non deve essere per forza in DB
NO IMMAGINI

Il pdf di un corso cosa deve contenere?

ti allego un esempio di descrittivo corso

La durata di un corso è espressa in minuti?

va bene

Il materiale necessario per un corso va bene gestirlo con un singolo campo di testo?

sì

Come devo specificare se un corso dura più di un giorno?

vanno specificati i giorni in cui si svolge

Quali sono le tipologie di un corso?

devono essere gestibili da db, al momento puoi mettere: riduzione revoca, formazione continua OAut, corsi per maestri conducenti

Un allievo iscritto deve inserire nuovamente i propri dati per iscriversi ad un corso?

dipende, se ha deciso di registrarsi quando si è iscritto ad un altro corso basta che si logghi, al momento dell'iscrizione dovrà comunque riconfermare i propri dati personali nel caso vi siano dei cambiamenti

Le email da inviare sono quando un utente si iscrive ad un corso e quando l'amministratore conferma l'iscrizione?

sì iscrizione e iscrizione definitiva (quando ha pagato)

Ti allego cosa dovrebbe esserci indicativamente scritto nella mail (potrebbe anche essere una mail con allegato un pdf del genere) (v. formulario d'iscrizione.pdf)

Una volta che l'amministratore conferma l'iscrizione può rimuovere l'iscrizione l'utente?

potrebbe succedere

La modifica delle informazioni dell'utente è fatta dall'utente stesso o dall'amministratore?

un utente modifica i propri dati se è il caso

Cosa significa che l'amministratore può gestire gli utenti che non sono iscritti al sito?
un utente non sa usare internet, telefona all'amministratore che lo iscrive

Cosa significa che l'amministratore può gestire gli utenti che non sono iscritti al sito?
un utente non sa usare internet, telefona all'amministratore che lo iscrive

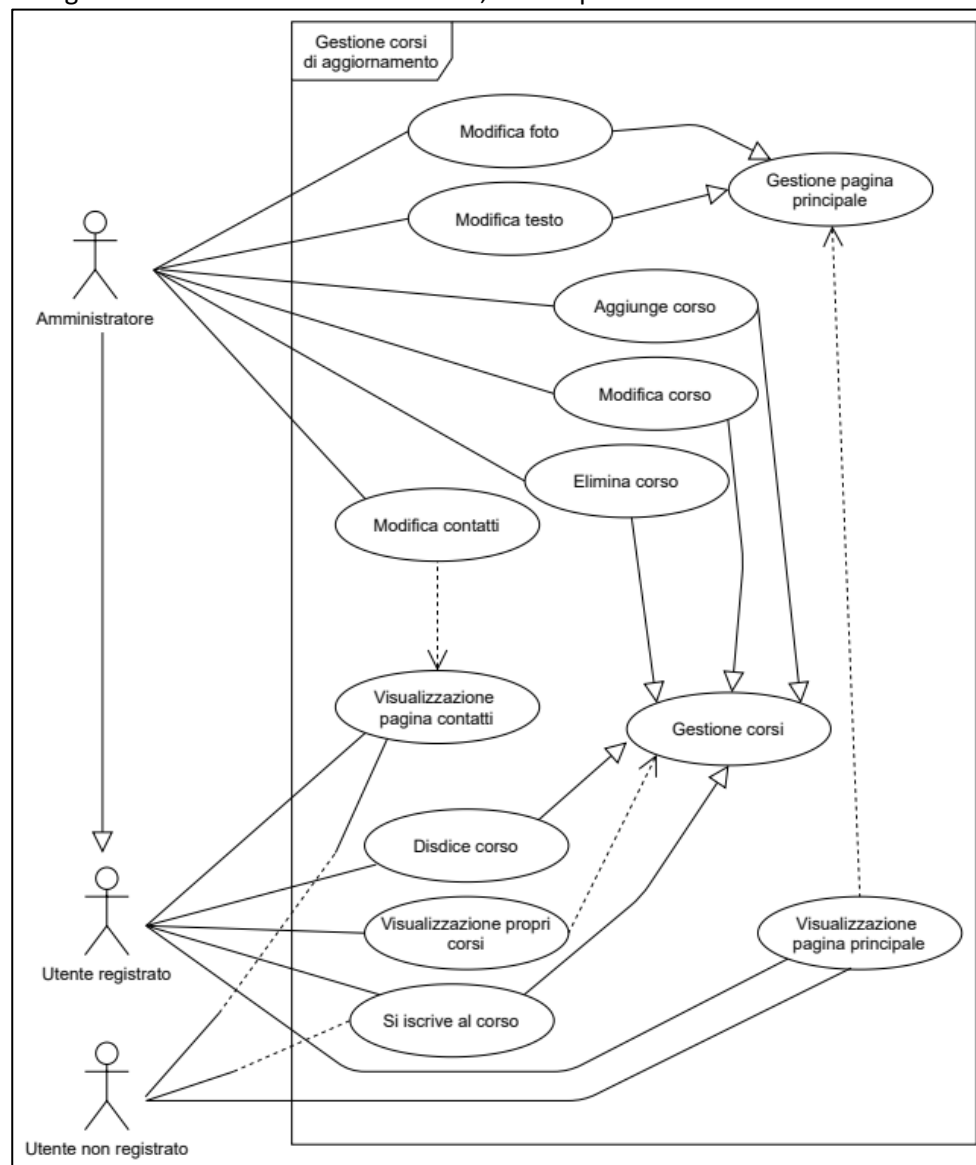
Esiste una gestione degli utenti?

No

Gli amministratori come si registrano?
per gli amministratori va bene Hard coded

Una volta scritte le domande ho incominciato a creare la fase di analisi della documentazione, in pratica riempito i capitoli 1.1 Informazioni sul progetto, 1.2 Abstract, 1.3 Scopo, 2.1 Analisi del dominio e 2.2 Analisi dei requisiti.

In seguito ho creato lo schema use case, esso si presenta così:



A questo schema ho associato la seguente descrizione:

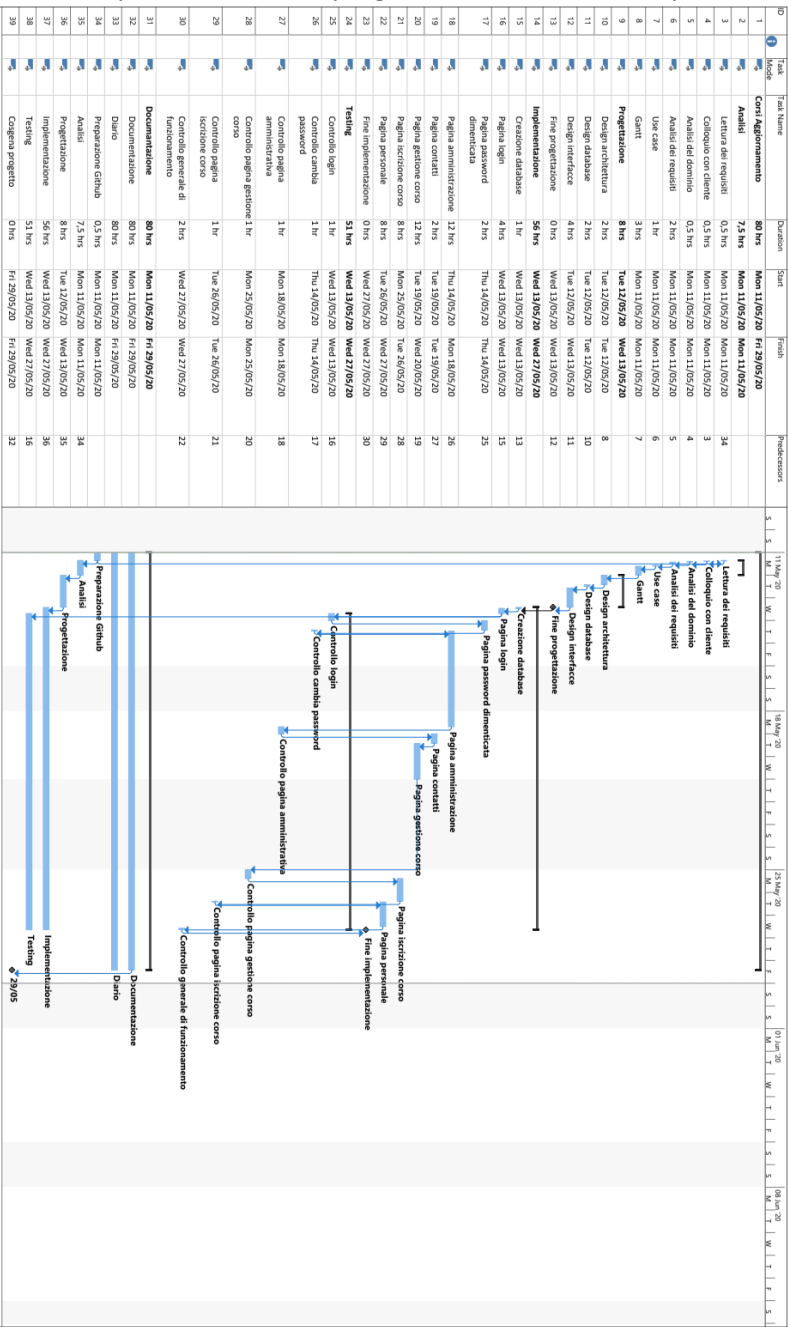
Lo schema inizia con tre tipologie di scenari: Amministratore (cioè colui che gestisce tutta l'applicazione), Utente registrato e Utente non registrato.

L'utente non registrato ha la possibilità di accedere alla pagina principale, dove gli vengono mostrate delle foto e una breve descrizione del sito. In seguito può accedere alla pagina di visualizzazione dei contatti dove può vedere i vari dipendenti dell'azienda. Infine può visualizzare e iscriversi ad dei corsi.

Per diventare un utente registrato bisogna aver effettuato almeno un'iscrizione ad un corso. Questo utente implementa tutte le funzioni di un utente non registrato e in più ha l'accesso ad una pagina dove può visualizzare tutti i corsi eseguiti e/o da fare. Inoltre ha la possibilità di disdire un corso da fare.

L'ultimo scenario è l'amministratore, esso ha la possibilità di accedere ad una pagina riservata dove può gestire la pagina principale, quindi modificare le foto e modificare il testo di descrizione del sito. Inoltre ha la possibilità di modificare la pagina di contatti attraverso un text editor. Infine ha la possibilità di aggiungere, modificare o eliminare un corso e di gestire le corrispettive iscrizioni.

Infine ho generato il Gantt di pianificazione del progetto. Questo schema di presenta così:



Infine ho descritto una ad una le varie attività del Gantt, partendo dall'analisi fino all'ultima fase.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione, in quanto oggi avevo pianificato che in 8 ore avrei concluso tutta la fase di analisi.

Programma di massima per la prossima giornata di lavoro

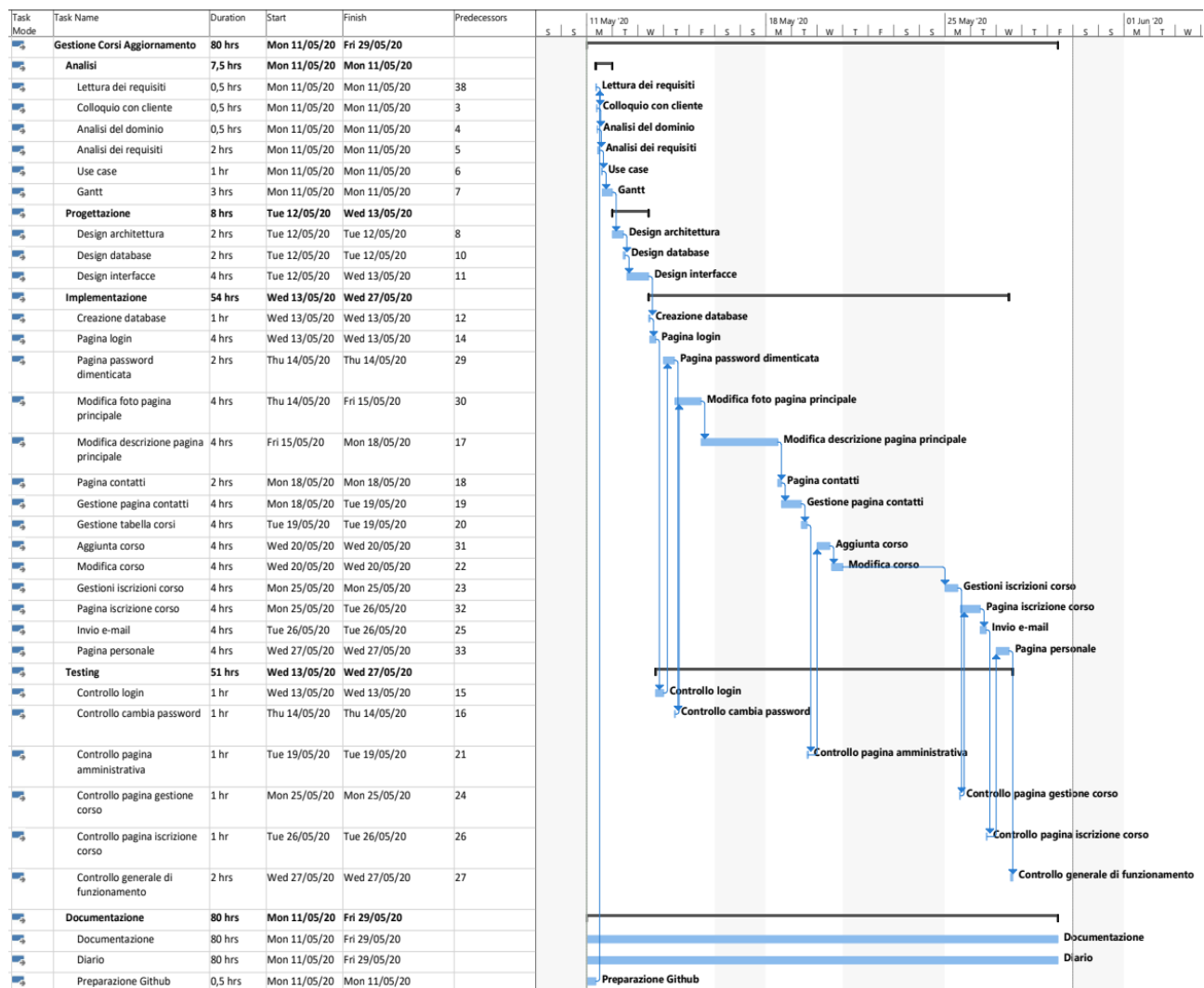
La prossima giornata di lavoro cercherò di fare e finire i capitoli 3.1 Design dell'architettura del sistema, 3.2 Design dei dati e database e iniziare il capitolo 3.3 Design delle interfacce.

Diario di lavoro

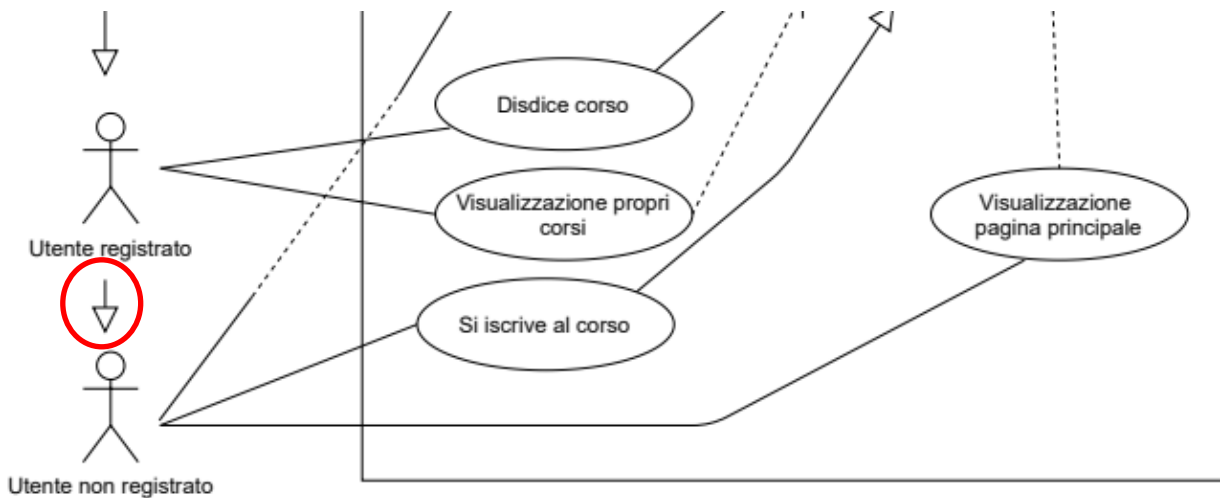
Luogo	Mendrisio
Data	12.05.2020

Lavori svolti

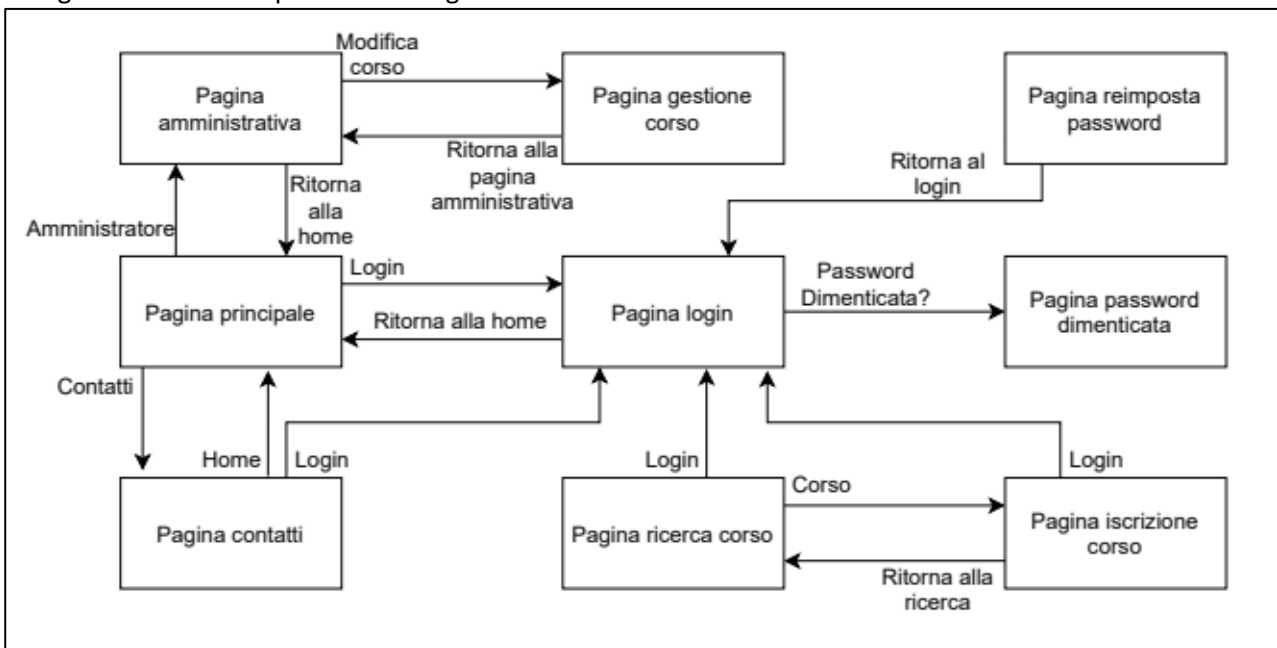
Innanzitutto ho modificato il gantt, sotto avviso del mio perito, in base a come richiesto dal criterio di valutazione A3 che dice esplicitamente che i task singoli non devono durare più di 4 ore. Inoltre ho rimosso le milestones e gran parte dei task della fase documentazione essendo ridondante.



Prima di iniziare con la progettazione ho sistemato una piccola ripetizione nello schema use case. Infatti l'utente registrato importa le funzioni di un utente non registrato e ne aggiunge di nuovi, perciò ho aggiunto una freccia extends e ho rimosso delle righe ridondanti:



In seguito ho fatto il capitolo 3.1 Design dell'architettura del sistema.



A questo schema ho assegnato la seguente descrizione:

La pagina di partenza della mia applicazione, cioè la pagina iniziale a cui si collega l'utente finale per utilizzare il mio sito, è la pagina principale. In questa pagina l'utente può farsi un'idea di quello che dispone tutto il sito, infatti può visualizzare delle immagini e leggere una breve descrizione di quello che offre l'azienda. In questa pagina, come nella maggior parte delle altre, sarà possibile effettuare un accesso cliccando il bottone "Login" in una navbar in alto. Nella pagina di login si può naturalmente fare il login inserendo le credenziali di accesso (e-mail e password) oppure si può modificare la propria password cliccando il bottone "Hai dimenticato la password?". Quando viene richiesta la procedura di modifica password tramite la pagina di password dimenticata viene inviata una e-mail, all'utente diretto interessato, la quale contiene un link di recupero. Cliccando questo link l'utente verrà trasportato sulla pagina reimposta password dove esso potrà inserire una nuova password.

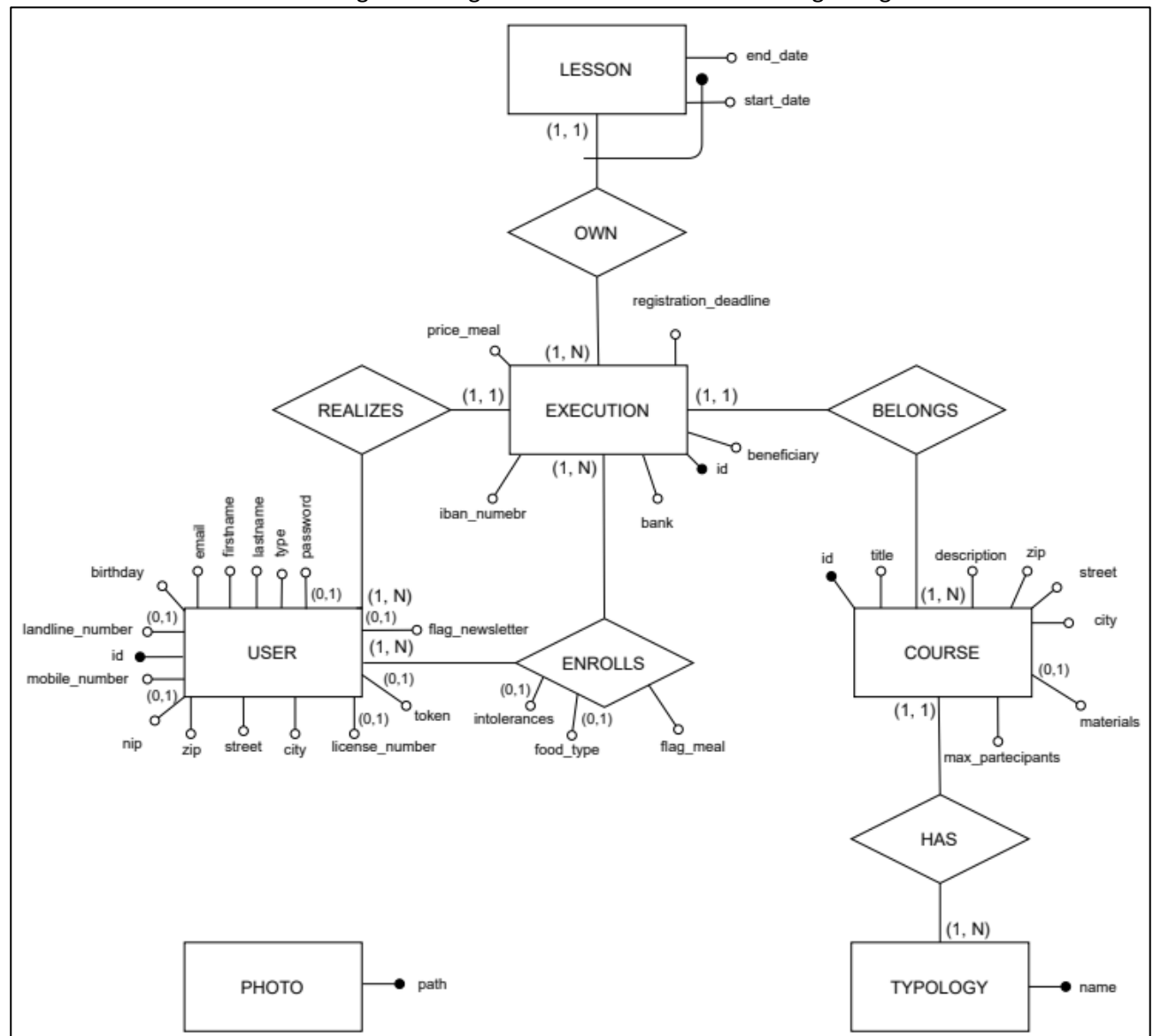
Ritornando alla pagina principale si può accedere alla pagina di ricerca corsi. In questa pagina sarà presente un filtro che permette di catalogare i corsi in base alla loro tipologia.

Cliccando su un singolo corso, l'utente potrà visualizzare tutte le informazioni di un determinato corso con

la possibilità di stampare un pdf riassuntivo. Inoltre avrà la possibilità di iscriversi al corso inserendo i propri dati oppure, se l'utente ha effettuato l'accesso, utilizzare i propri dati salvati da una precedente iscrizione. Una volta iscritto l'utente riceverà una breve email informativa di iscrizione al corso.

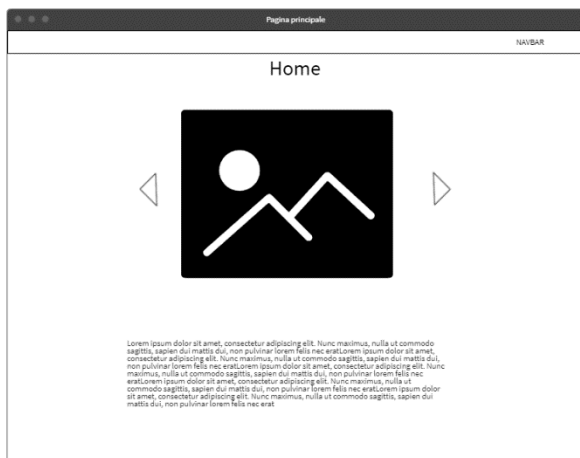
Se l'utente che ha effettuato il login è un admin, esso potrà accedere alla pagina amministrativa dove potrà gestire la pagina principale e la pagina di contatti. Inoltre avrà la possibilità di accedere alle pagine per la gestione dei singoli corsi.

Una volta fatto ciò ho creato il seguente diagramma e-r e ho documentato ogni singola entità:



Infine ho incominciato a creare i mockup delle interfacce:

Pagina principale



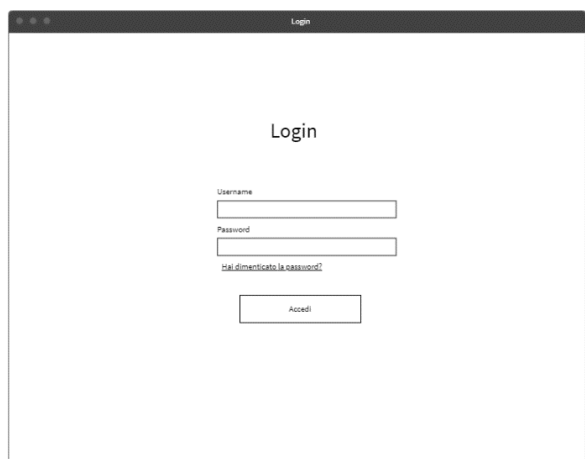
Questa è la pagina iniziale a cui l'utente finale si collega. In questa pagina si ha la possibilità di visualizzare delle immagini e una breve descrizione di cosa contiene il sito. Al di sopra della pagina si può trovare una navbar. Attraverso questa navbar si potrà accedere alla pagina di ricerca corsi e alla pagina di contatti.

Pagina contatti



Questa è la pagina per vedere i contatti dei dipendenti dell'azienda.

Pagina di login



Questa è la pagina di login in cui l'utente ha la possibilità di effettuare l'accesso inserendo le proprie credenziali. Inoltre in caso che l'utente avesse dimenticato la password si può cliccare il link "Hai dimenticato la password?" e si avvia una procedura di ripristino password.

Pagina richiesta nuova password

Recupera password

Email:

Questa pagina permette di richiedere una procedura per il ripristino della password. L'utente finale raggiunge questa pagina cliccando il link "Hai dimenticato la password?" dalla pagina di Login. Inserendo la propria mail e cliccando il bottone "Invia" si potrà avviare la procedura.

Pagina reimposta password

Cambia password

Password

Re-Password

Una volta inviata la richiesta di recupero password, arriverà una e-mail con un link che riporterà a questa pagina. In questa pagina si dovrà inserire due volte la nuova password di accesso. Una volta riempiti i due campi basterà cliccare il bottone "Cambia" e il cambiamento verrà applicato.

1.1.1 Pagina amministrativa

Pagina amministrativa

Navbar

Gestione pagina principale

	Elimina
	Elimina
	Elimina
	Elimina

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat. Nunc maximus, nulla ut commodo sagittis, sapien dui mattis dui, non pulvinar lorem fella nec erat.

Questa è la pagina di amministrazione del sito. Questa non può essere visualizzata se non dagli amministratori dell'applicazione.

Nella prima parte della pagina è possibile effettuare la modifica della pagina principale, scorrendo il sito si potrà accedere alla gestione dei singoli corsi.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi ho avuto difficoltà a progettare lo schema e-r in quanto non avevo ancora capito il funzionamento della creazione dei corsi. Dopo una chiamata con il mio formatore è risultato tutto più chiaro e sono riuscito a concludere la progettazione dell'ER.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione in quanto oggi avrei dovuto concludere il capitolo 3.1 Design architettura e il capitolo 3.2 Design database. Inoltre dovrei aver iniziato il capitolo 3.3 design delle interfacce.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro concluderò il capitolo 3.3 Design delle interfacce e inizierò la fase di implementazione.

Diario di lavoro

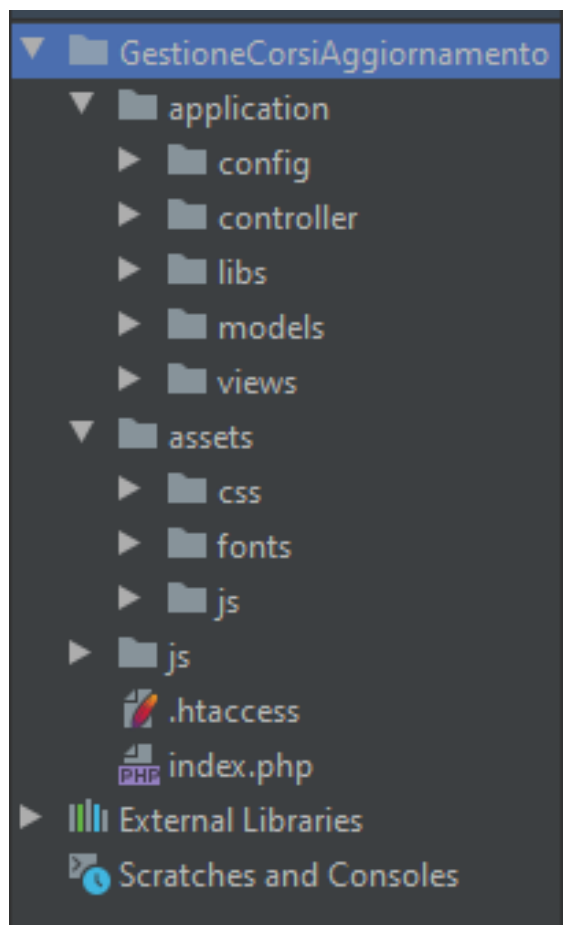
Luogo	Mendrisio
Data	13.05.2020

Lavori svolti

Durante le prime due ore della giornata di oggi ho concluso il capitolo 3.3 Design delle interfacce, in pratica ho fatto e finito altre 4 interfacce che vanno ad aggiungersi alle 6 di ieri. Le interfacce sono:

- Pagina ricerca corsi
- Pagina di iscrizione ad un corso
- Pagina gestione corsi
- Pagina personale

Una volta finita la progettazione ho scritto il codice SQL del database e ho incominciato a creare la cartella che conterrà tutti il sito web. Per la creazione del sito utilizzo un template MVC che ci è stato fornito dal docente Massimo Sartori.



Creata la cartella ho incominciato a sviluppare le interfacce di login. Per la creazione di questo sito utilizzo questo template: <https://github.com/puikinsh/srtdash-admin-dashboard>. Le interfacce che ho creato oggi sono queste:

The image displays five distinct user interface cards for a login and password management system, arranged in two rows. Each card has a purple header and a white body with teal accents.

- LOGIN:** Features a purple header with the text "LOGIN" and "Ciao, inserisci i tuoi dati per accedere al sito!". The body contains input fields for "Email" and "Password", a link for "Password dimenticata?", and a teal "Accedi" button.
- EMAIL INVIATA!:** Features a purple header with the text "EMAIL INVIATA!". The body contains a message: "Ti è stata inviata una email per cambiare la password del tuo account! Se non trovi la email controlla se non è finita nella cartella di spam." and a teal "Ritorna al login" button.
- PASSWORD DIMENTICATA?:** Features a purple header with the text "PASSWORD DIMENTICATA?" and "Inserisci la tua email e ti invieremo un link di recupero password!". The body contains an "Email" input field and a teal "Invia" button.
- CAMBIA PASSWORD:** Features a purple header with the text "CAMBIA PASSWORD". The body contains instructions: "Inserisci una nuova password, ripetila e poi premi Cambia. Dopodiché potrai utilizzare la nuova password per accedere. La password deve avere almeno 8 caratteri e numero/carattere speciale." It includes input fields for "Password" and "Ripeti password", and a teal "Cambia →" button.
- EMAIL INVIATA! (second instance):** Identical to the first "EMAIL INVIATA!" card, featuring a purple header, a message about password change email, and a "Ritorna al login" button.

Una volta fatte queste interfacce grafiche ho fatto una chiamata con il mio formatore e il mio perito. Durante questa chiamata ho dovuto modificare leggermente il database. Lo schema logico è diventato il seguente:

```

user(id, email, firstname, lastname, birthday, zip, city, street, mobile_number, flag_newsletter*,
landline_number*, password*, token*, type*, nip*, license_number*)
typology(name)
photo(path)
course(id, title, description, zip, street, city, max_participants, materials, name_typology (FK), meal_price,
course_price)
execution(id, id_user (FK), id_course (FK))
lesson(start_date, id_execution (FK), end_date)
enrolls(id_user (FK), id_execution(FK), intolerances*, food_type*, flag_meal)
settings(iban_number, registration_deadline, beneficiary, day_deadline)

```

Una volta modificato e-r, schema logico e descrizioni, ho incominciato a fare il back-end del login. Il metodo che si occupa di verificare che i dati inseriti siano corrispondenti ad un utente è il seguente:

```
/**
 * Metodo per effettuare il Login.
 */
public function access($email, $password)
{
    $result = $this->getUser($email);
    if (count($result) > 0) {
        if (password_verify($password, $result[0][DB_USER_PASSWORD])) {
            echo SUCCESSFUL;
            return $result;
        } else {
            echo LOGIN_DENY;
        }
    } else {
        echo LOGIN_DENY;
    }
}
```

Questo metodo si occupa di cercare l'utente nel database e solo in caso che la email inserita corrisponde a un utente verifica che la password sia corretta.

Per eseguire delle query al database utilizzo la libreria "Meekrodb" che mi permette di eseguire delle query senza perdere troppo tempo per scrivere i prepare statement.

Un esempio è il seguente:

```
$selectAccess = "SELECT * FROM user WHERE email=%s && type <> 0";
return $this->connection->query($selectAccess, $email);
```

Il metodo query prende come primo parametro la query da eseguire nel database e tutti gli altri parametri sono i valori da inserire nella query. In questo caso \$email sostituisce il valore %s che specifica una stringa.

Infine ho incominciato a creare la procedura di recupera password. Quando un utente non ricorda la propria password gli basta cliccare il link "Hai dimenticato la password?". A questo punto inserisce la sua email e gli viene inviata questa email:

Modifica la password

Da: Gestione Corsi Aggiornamento <gestionecorsiaggiornamento@gmail.com>

data: 13/05/2020 17:08

Ciao Mattia Toscanelli,
Recentemente è stata richiesta la procedura di modifica password!

[Per modificare la tua password clicca questo link!](#)

Per fare questo utilizzo questo spezzone di codice:

```
$hash = hash('sha256', random_bytes(16) . $this->email);
$updateUsers = "UPDATE user SET token=%s WHERE email=%s";
require 'application/libs/sendMail.php';
$body = "Ciao " . $result[0][DB_USER_NAME] . " " . $result[0][DB_USER_SURNAME] . ",<br>
Recentemente è stata richiesta la procedura di modifica password!<br><br>
<a href='" . URL . "resetPassword/changePassword/$hash/$this->email'> Per
modificare la tua password clicca questo link!</a>";
```

```
try {  
    $s = new SendMail();  
    $s->mailSend($this->email, "Modifica la password", $body);  
    $this->connection->query($updateUsers, $hash, $this->email);  
    return true;  
} catch (Exception $e) {  
    return false;  
}
```

In questo spezco di codice creo un valore random da 16 byte e genero la hash in sha256 e la aggiungo all'url di recupero password. Inoltre aggiungo la email dell'utente per identificare l'utente che ha richiesto la password. Una volta preparato il link invio la email grazie al metodo mailSend(), dove passo e-mail del destinatario, header e contenuto. Per inviare le email utilizzo la classe PHPMailer. Il codice che ho scritto oggi è preso in gran parte dal progetto scorso.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi ho avuto un piccolo problema nel collegarmi al database. Infatti quando cercavo di accedere al database da PHP mi ritornava il seguente errore: "The server requested authentication method unknown to the client". Per risolvere questo problema ho dovuto eliminare l'utente dal database e re-inserirlo con il seguente comando:

```
CREATE USER 'courses_management_admin'@'localhost' IDENTIFIED WITH mysql_native_password BY 'CoursesManagement&1';
```

Inoltre ho aggiunto due righe al fine my.cnf di MySQL server:

```
[mysq]  
default-authentication-plugin=mysql_native_password
```

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione in quanto oggi avrei dovuto finire la pagina di login e iniziare la pagina di recupero password.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò finirò la pagina di recupero password e inizierò la gestione della pagina principale.

Diario di lavoro

Luogo	Mendrisio
Data	14.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto inizialmente aggiunto una descrizione del pattern che utilizzo nel progetto nella documentazione e ho incominciato a spiegare un po' la pagina di login. Inoltre ho aggiunto la descrizione di due classi nel capitolo 3.4 Diagramma delle classi che avevo utilizzato nel progetto scorso.

Una volta aggiunta questa piccola parte ho concluso la parte di modifica password. Infatti ora quando l'utente clicca il link contenuto nella email accederà alla pagina in cui dovrà inserire due volte la password. Per identificare che l'utente che ha cliccato il link sia lo stesso che ha richiesto la modifica della password utilizzo questo metodo:

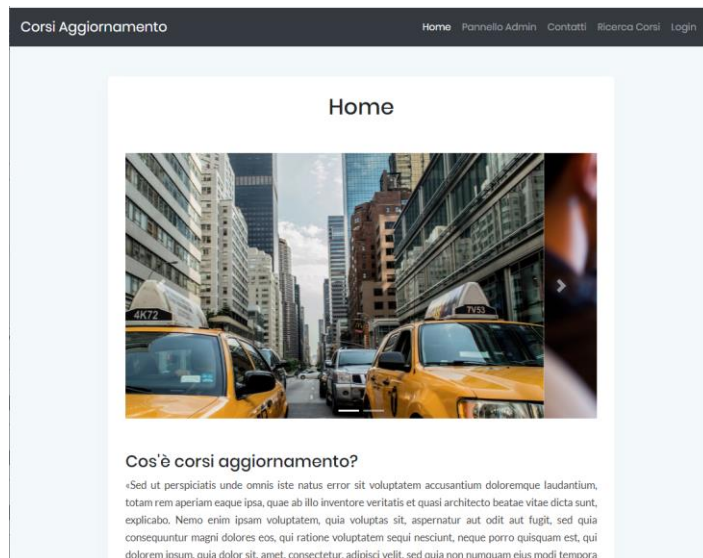
```
function resPassword($hash, $email)
{
    if($hash != null) {
        $selectUsers = "SELECT * FROM user WHERE email=%s AND token=%s";
        $result = $this->connection->query($selectUsers, $email, $hash);
        if ($result != null) {
            $updateUsers = "UPDATE user SET token=NULL WHERE email=%s";
            $this->connection->query($updateUsers, $email);
            return true;
        } else {
            return false;
        }
    } else {
        return false;
    }
}
```

Controllo che la hash in sha256 sia uguale al token salvato nel database e che la email corrisponda all'utente che vuole cambiare la password. Se tutto va a buon fine l'utente potrà cambiare la password. Per cambiare la password utilizzo questo metodo:

```
public function modifyPassword($email, $password1, $password2)
{
    if ($this->validator->checkPassword($password1, $password2)) {
        $password = password_hash($password1, PASSWORD_DEFAULT);
        $updateUsers = "UPDATE user SET password=%s WHERE email=%s";
        $this->connection->query($updateUsers, $password, $email);
        echo SUCCESSFUL;
        return true;
    }
    echo ERROR;
    return false;
}
```

Controllo che la password inserita sia uguale in entrambi i campi e che abbia almeno 8 caratteri e un numero o carattere speciale. Se tutta va a buon fine salvo la nuova hash della password.

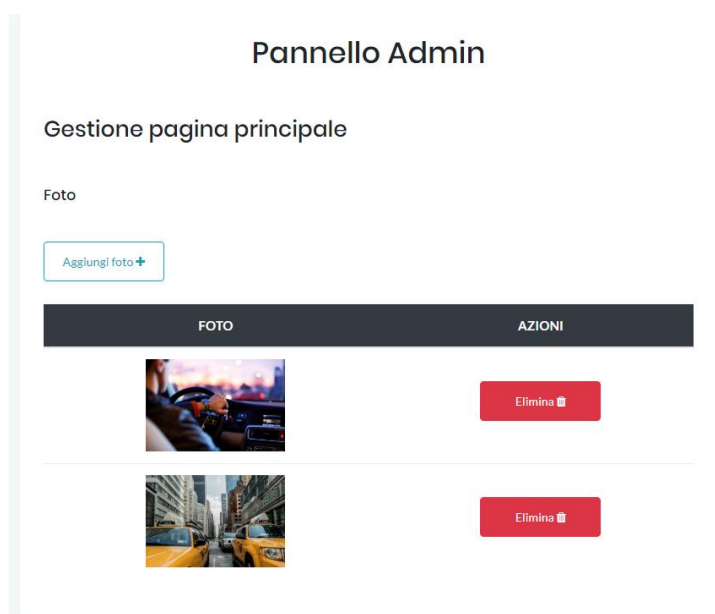
Una volta conclusa la modifica della password ho creato la pagina principale, essa si presenta così:



La pagina è formata da un carosello e da un breve descrizione. Le immagini sono all'interno della cartella img e i percorsi sono salvati in database. Per leggere le immagini utilizzo questo metodo:

```
public function getAllPathHomePicture()
{
    $selectPicture = "SELECT path FROM photo";
    $pictures = $this->connection->query($selectPicture);
    return $pictures;
}
```

Per quanto riguarda la descrizione momentaneamente è ancora statica. Infine ho incominciato a sviluppare la gestione delle immagini nella pagina di amministrazione. La tabella per gestire le immagini è questa:



Momentaneamente ho implementato solo la funzione di eliminazione. Per eliminare una foto utilizzo questo metodo:

```
public function delPhoto($path)
{
    $selectPhoto = "SELECT * FROM photo WHERE path=%s";
    $photo = $this->connection->query($selectPhoto, $path);
    if(count($photo)>0){
        $deletePhoto = "DELETE FROM photo WHERE path=%s";
        $this->connection->query($deletePhoto, $path);
        unlink($photo[0][DB_PATH_PHOTO]);
        echo SUCCESSFUL;
    } else {
        echo ERROR;
    }
}
```

Verifico che la email è presente nel database e in caso elimino il riferimento e poi elimino il file.

Le immagini utilizzate sono state ricavate da questo sito che offre delle immagini copyright free:

<https://www.pexels.com/it-it/>

Nello scorso diario mi sono dimenticato di citare dove ho preso la libreria PHPMailer (<https://github.com/PHPMailer/PHPMailer>) e la libreria Meekrodb (<https://meekro.com/>).

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione in quanto oggi dovevo iniziare la gestione delle immagini della pagina principale.

Programma di massima per la prossima giornata di lavoro

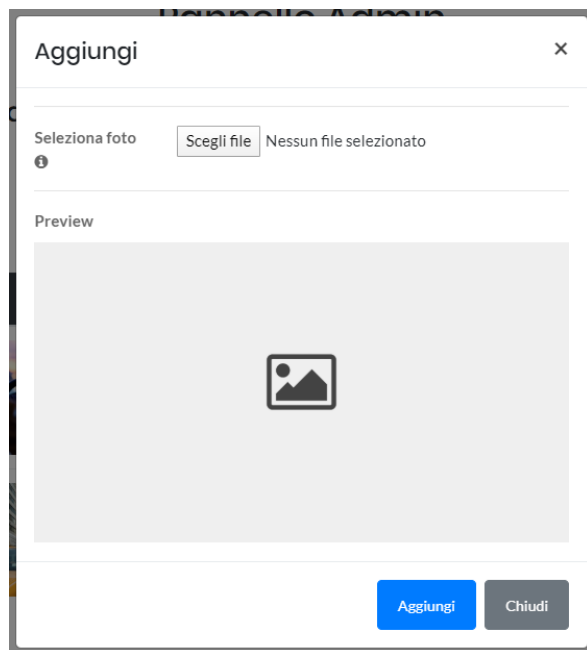
La prossima giornata di concluderò la gestione delle immagini e inizierò concluderò la gestione della descrizione della pagina principale.

Diario di lavoro

Luogo	Mendrisio
Data	15.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto ho concluso la gestione della pagina principale, quindi ho concluso la gestione delle immagini e della descrizione. Come prima cosa ho implementato la funzione per aggiungere delle immagini. Quando l'utente vuole aggiungere un'immagine viene aperto questo modale:



Aggiungendo l'immagine nell'input viene mostrata una piccola preview sotto. Per mostrare l'immagine utilizzo questo codice:

```
document.getElementById("files").onchange = function () {  
    var reader = new FileReader();  
    reader.onload = function (e) {  
        document.getElementById("image").src = e.target.result;  
    };  
    reader.readAsDataURL(this.files[0]);  
};
```

Per aggiungere un'immagine al sito utilizzo questo metodo:

```
function addPhoto($photo)  
{  
    $target_dir = "img/";  
    $target_file = $target_dir . basename($photo[PICTURE_NAME]);  
    $uploadOk = SUCCESSFUL;  
    $imageFileType = strtolower(pathinfo($target_file, PATHINFO_EXTENSION));  
    if($photo[PICTURE_TMP_NAME]!==null){  
        if (!(getimagesize($photo[PICTURE_TMP_NAME]) !== false)) {  
            $uploadOk = ERROR;  
        }  
    }  
}
```

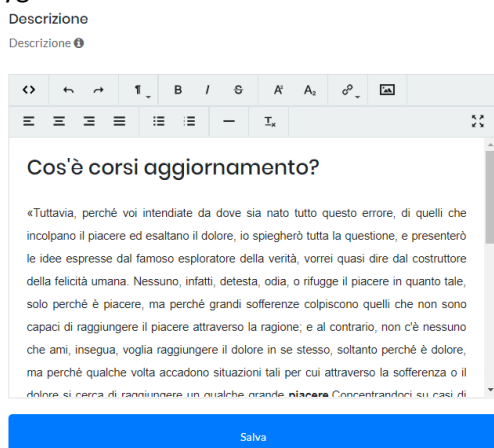
```

}else{
    $uploadOk = ERROR;
}
if ($photo[PICTURE_SIZE] > 5000000) {
    $uploadOk = ERROR;
}
if ($imageFileType != "jpg" && $imageFileType != "png" && $imageFileType != "jpeg"
    && $imageFileType != "gif") {
    $uploadOk = ERROR;
}
$hash = hash('sha256', basename($photo[PICTURE_NAME] . time()));
$target_file = $target_dir . $hash . "." . $imageFileType;
if ($uploadOk == 1) {
    if (move_uploaded_file($photo[PICTURE_TMP_NAME], $target_file)) {
        $os = $this->getOS();
        if ($os == OS_WIN) {
            exec('icacls "' . $target_file . '" /q /c /reset');
        } else if ($os == OS_LINUX || $os == OS_OSX) {
            chmod($target_file, 0777);
        } else {
            echo ERR_OS;
            exit;
        }
        $insertPhoto = "INSERT INTO photo (path) VALUES (%s)";
        $this->connection->query($insertPhoto, $target_file);
        echo SUCCESSFUL;
    } else {
        echo ERR_IMP;
    }
} else {
    echo ERROR;
}
}
}

```

Questo metodo si occupa di controllare l'immagine caricata (estensione, memoria occupata, grandezza immagine). Se tutto è corretto aggiungo l'immagine al database e nella cartella img. Per evitare che ci siano due immagini con lo stesso nome, modifico quest'ultimo con una hash in sha256 che combina nome e timestamp. Questo metodo è stato preso dal mio vecchio progetto.

Una volta finita la gestione delle immagini ho fatto la gestione della descrizione della pagina principale. Per la modifica ho utilizzato una textarea che diventa, grazie alla libreria Trumbowyg (<https://alex-d.github.io/Trumbowyg/>), un editor html. Inizialmente avevo trovato delle alternative (come Tiny e Froala), ma offrivano solo delle librerie a pagamento. Poi avevo provato bootstrap-wysihtml5 ma non sono riuscito ad implementarlo essendo una libreria ormai obsoleta (2012). Quindi la mia scelta è stata Trumbowyg:



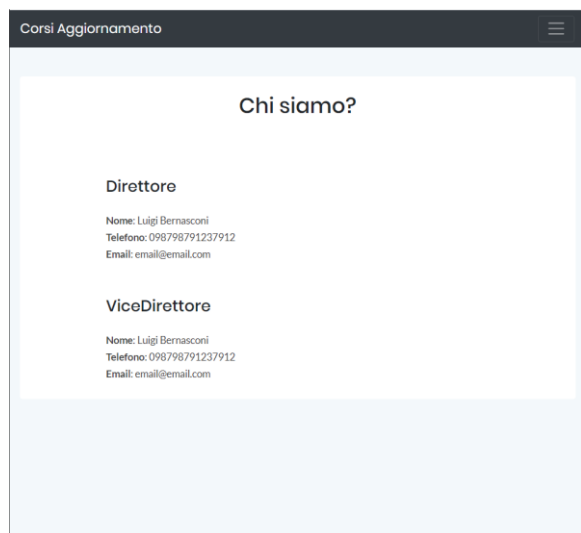
Per implementare questa libreria mi è bastato aggiungere due riferimenti (js e css) e impostare alla textarea la libreria.

```
window.onload = function() {  
    $('#description_homepage').trumbowyg({  
        lang: 'it'  
    });  
}
```

La descrizione viene salvata su un file a parte grazie a questo metodo:

```
public function saveDescription($text)  
{  
    file_put_contents($this->pathDescription, $text);  
}
```

Infine ho creato la pagina di contatti, in quanto è molto simile alla pagina principale. Momentaneamente è un po' vuota, ma voglio implementare nell'editor di modifica la creazione di tabelle.



Come già si può intuire ho ripetuto la modifica della descrizione della pagina principale per questa pagina di contatti. Quindi nel pannello admin ho aggiunto un secondo editor.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea circa 4 ore in anticipo rispetto la pianificazione. Questo perché non ho più documentato nulla dal login. Quindi queste 4 ore verranno utilizzate la prossima giornata per scrivere la documentazione fino al punto odierno.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò continuerò la documentazione, aggiungerò la creazione di tabelle nell'editor e aggiungerò la funzione per modificare le impostazioni del sito (iban, scadenza corso,...)

Diario di lavoro

Luogo	Mendrisio
Data	18.05.2020

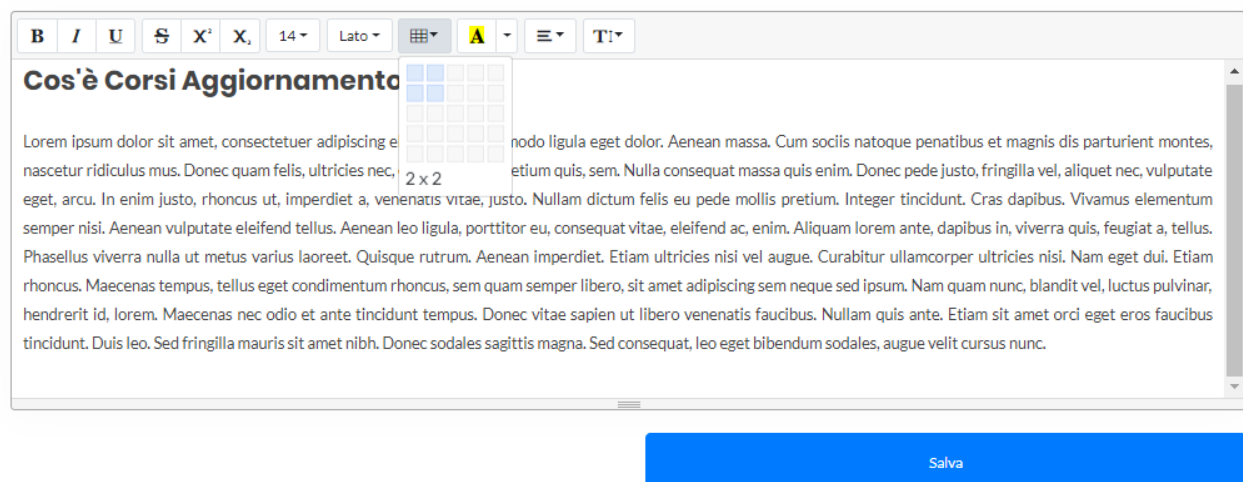
Lavori svolti

Durante la giornata di oggi ho concluso le funzionalità di gestione del sito. Più precisamente ho aggiunto la possibilità di creare delle tabelle nell'editor html per la modifica delle pagine e ho aggiunto la possibilità di modificare le impostazioni della pagina.

Per aggiungere la creazione delle tabelle ho direttamente cambiato libreria. Questa scelta è stata fatta in quanto la libreria trumbowyg, quando si andavano ad aggiungere dei plugin extra, rimuoveva tutti quelli di base. La libreria che ho scelto si chiama summernote (<https://summernote.org/>).

L'editor si presenta così:

Descrizione



Una volta fatto ciò ho aggiunto una tabella per mostrare tutti gli utenti registrati al sito. Per avere una tabella organizzata e responsive ho utilizzato la libreria datatables (<https://datatables.net/>).

La tabella si mostra così:

Utenti registrati

Mostra 10 utenti per pagina

Cerca:

NEWSLETTER	↑↓	NOME	↑↓	COGNOME	↑↓	EMAIL	↑↓	DATA DI NASCITA	↑↓	VIA	↑↓	CITTÀ	↑↓	CAP	↑↓
	✓	Mattia		Toscanelli		gestionecorsi@yopmail.com		2001-03-25		Via rime 10		Mendrisio		6850	
	✓	Mattia		Toscanelli		gestionecorsi3@yopmail.com		2001-03-25		Via rime 10		Mendrisio		6850	
	✗	Mattia		Toscanelli		gestionecorsi2@yopmail.com		2001-03-25		Via rime 10		Mendrisio		6850	
	✗	Mattia		Toscanelli		gestionecorsi4@yopmail.com		2001-03-25		Via rime 10		Mendrisio		6850	

Pagina 1 di 1

Precedente **1** Successivo

Infine ho creato la gestione delle impostazioni della pagina principale. In questa parte di pagina si possono modificare i dati di pagamento, modificare i giorni rimanenti per iscriversi ad un corso prima del suo inizio e aggiungere/rimuovere tipologie di corsi.

The screenshot shows the 'Corsi Aggiornamento' (Course Update) admin interface. It has a top navigation bar with links: Home, Pannello Admin, Contatti, Ricarica Corsi, and Login. The main content area is divided into three sections:

- Pagamento** (Payment): Contains a 'Modifica pagamento / CF' button and a table with two columns: TIPO and VALORE. The table has three rows: IBAN (value: CH8901244907444344457), Banca (value: UBS), and Beneficiario (value: Mattia Toscanelli).
- Giorni scadenza corso** (Course expiration days): Contains a 'Modifica giorni / CF' button and a table with two columns: TIPO and VALORE. The table has one row: Giorni scadenza (value: 3).
- Tipologie** (Types): Contains an 'Aggiungi tipologia +>' button and a table with two columns: TIPOLOGIA and AZIONI. The table has three rows: Auto, Canion, and Moto. Each row has a red 'Elimina' button with a trash icon.

La gestione dei dati di pagamento e dei giorni di scadenza è molto simile. Il metodo per modificare i dati di pagamento è il seguente:

```
public function modifyPayment($iban, $bank, $beneficiary)
{
    $data = array();
    $checkIBAN = $checkBank = $checkBeneficiary = $checkAll = SUCCESSFUL;
    if(!$this->validator->checkIBAN($iban)){
        $checkIBAN = ERROR;
        $checkAll = ERROR;
    }
    if(!$this->validator->checkNameNumber($bank)){
        $checkBank = ERROR;
        $checkAll = ERROR;
    }
    if(!$this->validator->checkName($beneficiary)){
        $checkBeneficiary = ERROR;
        $checkAll = ERROR;
    }
    if($checkAll == SUCCESSFUL){
        $updateSettings = "UPDATE settings SET iban_number=%s, bank=%s,
beneficiary=%s";
        $this->connection->query($updateSettings,$iban,$bank,$beneficiary,$iban);
    }
    $data[] = array(
        SATUTS => $checkAll,
        CHECK_IBAN => $checkIBAN,
        CHECK_BANK => $checkBank,
        CHECK_BENEFICIARY => $checkBeneficiary
    );
    echo json_encode($data);
}
```

Questo metodo si occupa di verificare tutti i dati passati tramite il form e ritornare un json che contiene lo stato dell'operazione (fallito/bocciato) e i vari risultati dei controlli degli input. Grazie a questo json posso mostrare all'utente delle notifiche di successo/errore. Lo stesso procedimento è stato fatto per la gestione dei giorni di scadenza, ma naturalmente con un solo dato da controllare.

Invece per quanto riguarda la gestione delle tipologie ho implementato l'aggiunta e l'eliminazione. Per aggiungere una tipologia utilizzo questo metodo:

```
function addTypology($typology){
    if($this->validator->checkNameNumber($typology)){
        $selectTypology = "SELECT * FROM typology WHERE name=%s";
        $typologies = $this->connection->query($selectTypology,$typology);
        if(count($typologies) == 0) {
            $insertTypology = "INSERT INTO typology (name) VALUES (%s)";
            $this->connection->query($insertTypology,$typology);
            echo SUCCESSFUL;
        }else{
            echo ERR_DUP;
        }
    }else{
        echo ERROR;
    }
}
```

Esso verifica che la tipologia non sia già stata aggiunta in precedenza e che abbia un nome valido. In caso che entrambi i controlli vadano a buon fine la tipologia viene aggiunta al database e la tabella html viene aggiornata tramite ajax.

Per rimuovere una tipologia utilizzo questo metodo:

```
public function delTypology($name)
{
    $selectTypology = "SELECT * FROM typology WHERE name=%s";
    $typologies = $this->connection->query($selectTypology, $name);
    if(count($typologies)>0){
        $deleteTypology= "DELETE FROM typology WHERE name=%s";
        $this->connection->query($deleteTypology, $name);
        echo SUCCESSFUL;
    } else {
        echo ERROR;
    }
}
```

Esso verifica che la tipologia sia presente nel database e nel caso la elimina.

Infine ho documentato tutta la parte di login, recupero password, reimposto password e gestione della pagina principale.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione in quanto oggi avrei dovuto concludere la gestione delle impostazioni dell'applicazione.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò l'aggiunta dei corsi di aggiornamento e de resta tempo la modifica.

Diario di lavoro

Luogo	Mendrisio
Data	19.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto aggiunto nella documentazione la spiegazione delle caratteristiche principali della pagina di amministrazione. Fatto ciò ho iniziato e finito di implementare l'aggiunta di un corso. La pagina di aggiunta è la seguente:

The screenshot shows a web application titled 'Corsi Aggiornamento'. The navigation bar includes links for Home, Pannello Admin, Gestioni corsi, Contatti, Ricerca Corsi, and Login. The main content area is titled 'Aggiunta corso' and contains a form with the following fields:

- Titolo* (text input)
- Via* (text input)
- CAP* (text input)
- Città* (text input)
- Numero max. partecipanti* (text input)
- Tipologia* (dropdown menu with 'Auto' selected)
- Prezzo corso* (text input)
- Prezzo pasto* (text input)
- Descrizione* (text area)
- Materiale* (text area)

At the bottom of the form, there is a note: '* campi da compilare'. Below the form are two buttons: 'Aggiungi Corso' (blue) and 'Torna indietro' (white with blue border).

Per aggiungere un corso utilizzo questo metodo:

```
public function addCourse($title, $street, $zip, $city, $maxParticipants, $typology,
$coursePrice, $mealPrice, $courseDescription, $materials)
{
    $data = array();
    $checkTitle = $checkStreet = $checkZip = $checkCity = $checkMaxParticipants =
$checkTypology = $checkCoursePrice = $checkMealPrice = $checkCourseDescription =
$checkMaterials = $checkAll = SUCCESSFUL;
    if (!$this->validator->checkTitle($title)) {
        $checkTitle = ERROR;
        $checkAll = ERROR;
    }
    [...]

    if($materials != ""){
        $materials == null;
    }
    if ($checkAll == SUCCESSFUL) {
        $addCourse = "INSERT INTO course
```



```

(title,description,zip,city,street,max_partecipants,materials,meal_price,course_price,n
ame_typology) VALUES (%s,%s,%i,%s,%s,%i,%s,%d,%d,%s)";
    $this->connection->query($addCourse, $title, $courseDescription, $zip, $city,
    $street, $maxPartecipants, $materials, $mealPrice, $coursePrice, $typology);
}
$data[] = array(
    SATUTS => $checkAll,
    CHECK_TITLE => $checkTitle,
    CHECK_COURSE_DESCRIPTION => $checkCourseDescription,
    CHECK_ZIP => $checkZip,
    CHECK_CITY=> $checkCity,
    CHECK_STREET => $checkStreet,
    CHECK_MAX PARTECIPANTS => $checkMaxPartecipants,
    CHECK_MATERIALS => $checkMaterials,
    CHECK_MEAL_PRICE => $checkMealPrice,
    CHECK_COURSE_PRICE => $checkCoursePrice,
    CHECK TYPOLOGY => $checkTypology
);
echo json_encode($data);
}

```

In pratica si occupa di verificare tutti i dati del form e di ritornare un json contenente tutti i vari risultati dei controlli. In caso che i controlli fossero andati tutti a buon fine il corso viene aggiunto al database.

Una volta finita l'aggiunta ho creato una pagina dove vengono mostrati tutti i corsi:

Per la creazione della tabella ho utilizzato la libreria Datatables, già spiegata in un diario scorso. Cliccando il "+" a inizio riga possiamo accedere a dei pulsanti di gestione di quel determinato corso:

Momentaneamente ho implementato solamente la modifica del corso. Non inserisco codice in quanto è praticamente lo stesso dell'aggiunta.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono circa 8 ore in anticipo rispetto alla mia pianificazione. Queste ore saranno utilizzate per scrivere la documentazione, in quanto il mio perito mi ha consigliato da adesso di portarmi avanti solo con il progetto.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò l'eliminazione di un corso e la creazione di lezioni per un determinato corso.

Diario di lavoro

Luogo	Mendrisio
Data	20.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto l'eliminazione di un corso e l'aggiunta di uno svolgimento (compresa la tabella che mostra tutti gli svolgimenti).

Per eliminare un corso utilizzo il metodo seguente:

```
public function deleteCourse($id)
{
    $selectCourse = "SELECT * FROM course WHERE id=%i";
    $courses = $this->connection->query($selectCourse, $id);
    if (count($courses) > 0) {
        $deleteCourse = "DELETE FROM course WHERE id=%i";
        $this->connection->query($deleteCourse, $id);
        echo SUCCESSFUL;
    } else {
        echo ERROR;
    }
}
```

Una volta finite la gestione dei corsi ho iniziato la gestione dell'aggiunta di uno svolgimento, a mio parere la parte più complessa del progetto. Prima di iniziare però ho effettuato una chiamata con il mio docente responsabile dove abbiamo discusso dei seguenti argomenti:

- Nell'applicazione possono venir aggiunti in modo hardcoded i docenti, essi prenderanno nella tabella user il type=2. Questo tipo di utente non interagisce con l'applicazione ma serve solamente per specificare l'utente che insegnerà al corso.
- Per insegnare un corso l'utente deve essere docente o admin.
- L'utente che vuole insegnare non può insegnare altri corsi durante lo svolgimento di un corso (non può eseguire due corsi in contemporaneo).

Il form per aggiungere uno svolgimento è questo

The screenshot shows a web application interface for managing courses. In the background, there is a table titled 'Corsi' with columns for 'TIPOLOGIA' and 'TITOLO'. A modal window titled 'Aggiungi svolgimento' is open in the foreground. The modal contains the following fields:

- Data inizio**: A date input field with the value '24.05.2020'.
- Durata Scadenza**: A date input field with the value '21.05.2020'.
- Durata corso**: A dropdown menu with the selected value '1 Giorno'.
- Insegnate**: A dropdown menu with the selected value 'gestionecorsi@yopmail.com'.
- Inserisci ora inizio e ora fine dei seguenti giorni**: A section with two input fields:
 - Inizio**: A time input field with the value '14:00'.
 - Fine**: A time input field with the value '15:00'.

At the bottom of the modal, there are two buttons: 'Aggiungi' (Add) and 'Chiudi' (Close).

Nel form va specificato la data di inizio, la durata in giorni, il docente che insegnerà lo svolgimento e i vari orari delle lezioni. La creazione di tutte le iterazioni di questo form mi ha preso tutta la giornata. Per calcolare la data di inizio minima utilizzo questo codice:

```
$.when(
  getDeadline()
).done(function (result) {
  var min = new Date();
  min.setDate(min.getDate() + Number(result) + 1);
  var dd = min.getDate();
  var mm = min.getMonth()+1;
  var yyyy = min.getFullYear();
  if(dd<10){
    dd='0'+dd;
  }
  if(mm<10){
    mm='0'+mm;
  }
  min = yyyy + '-' + mm + '-' + dd;
  var day1 = dd + '.' + mm + '.' + yyyy;
  var deadline = new Date(min);
  deadline.setDate(deadline.getDate() - Number(result));
  document.getElementById("deadline_date").innerText = getDateFormat(deadline);
  document.getElementById("in_date").value = min;
  document.getElementById("day1").innerText = day1;
  document.getElementById("in_date").setAttribute("min", min);
  document.getElementsByTagName("modal_input_add_exe").value = id;
});
```

In pratica leggo i giorni per calcolare la scadenza dell'iscrizione dal database e poi imposto in base a questo la data minima di inizio corso.

La select degli insegnati viene riempita grazie a questo metodo:

```
public function getTeachers(){
  $SelectUsers = "SELECT * FROM user WHERE type>1";
  $users = $this->connection->query($SelectUsers);
  return $users;
}
```

La parte più difficile da capire è quella di aggiungere/rimuovere input time in base alla select della durata del corso:

```
function changeDay(x) {
  var count = Number(x.value);
  if (last_count < count) {
    var date = document.getElementById("in_date").value;
    var newData = new Date(date);
    for (var i = last_count + 1; i <= count; i++) {
      $('#exe_body').append('<tr id="raw_exe" + i + ">' +
        '<td>' +
        '<strong class="dL">' + getDateFormat(newData.setDate(newData.getDate()
+ 1)) + '</strong><br>' +
        'Inizio: <input id="start_time" + i + "' type="time" class="form-
control mb-1" min="06:00" max="22:00" value="14:00">' +
        'Fine: <input id="end_time" + i + "' type="time" class="form-control
mb-1" min="06:00" max="22:00" value="15:00" required>' +
        '</td>' +
```

```

        '</tr>');
    }
    last_count = count;
} else {
    for (var i = last_count; i > count; i--) {
        $('#raw_exe'+i+'').remove();
    }
    last_count = count;
}
}
}

```

Una volta preparato il form ho fatto l'aggiunta la validazione e l'aggiunta dei dati al database grazie a questo metodo:

```

public function
addExecution($start_day_lesson,$duration_day,$teacher,$times,$id_course)
{
    $data = array();
    $checkStartDayLesson = $checkDurationDay = $checkTeacher = $checkTimes =
    $checkOverlap = $checkAll = SUCCESSFUL;
    $end = "";
    $day = explode("-", $start_day_lesson);
    if (!$checkdate($day[1], $day[2], $day[0])) {
        $checkStartDayLesson = ERROR;
        $checkAll = ERROR;
    } else {
        $end = date('Y-m-d', strtotime($start_day_lesson. ' + ' . ($duration_day-1). '
days'));
        if ($this->validator->checkOverlap($start_day_lesson, $end, $teacher)) {
            $checkOverlap = ERROR;
            $checkAll = ERROR;
        }
    }
    if (!$this->validator->checkDuration($duration_day)) {
        $checkDurationDay = ERROR;
        $checkAll = ERROR;
    }
    if (!$this->validator->checkTeacher($teacher)) {
        $checkTeacher = ERROR;
        $checkAll = ERROR;
    }
    if (!$this->validator->checkTimes($times)) {
        $checkTimes = ERROR;
        $checkAll = ERROR;
    }

    if ($checkAll == SUCCESSFUL) {
        $selectCourse = "SELECT id FROM course WHERE id=%i";
        $course = $this->connection->query($selectCourse, $id_course);
        if (count($course) > 0) {
            $addExecution = "INSERT INTO execution (id_user, id_course, start, end)
VALUES (%i, %i, %s, %s)";
            $this->connection->query($addExecution, $teacher, $id_course,
$start_day_lesson, $end);
            $result = $this->connection->query("SELECT LAST_INSERT_ID() as 'id'");
            $idE = $result[0]["id"];
            $date = $start_day_lesson;
            $times = explode(" ", $times);
            for ($i = 0; $i < count($times); $i+=2){

```

```

        $addLesson = "INSERT INTO lesson (start,end,id_execution) VALUES
(%s,%s,%i)";
        $this->connection->query($addLesson, $date." ".$times[$i],$date."
".$times[$i+1], $idE);
        $date = date('Y-m-d', strtotime($date. ' + 1 days'));
    }
    }else{
        Util::fail();
    }
}
}
$data[] = array(
    SATUTS => $checkAll,
    CHECK_START_DAY => $checkStartDayLesson,
    CHECK_OVERLAP => $checkOverlap,
    CHECK_DURATION => $checkDurationDay,
    CHECK_TEACHER => $checkTeacher,
    CHECK_TIMES => $checkTimes
);
echo json_encode($data);
}

```

In questo metodo controllo la data di inizio, la durata del corso, verifico che l'utente che insegnerà il corso sia un docente o admin, verifico tutti gli intervalli di orari e infine controllo se l'utente che insegnerà il corso non sia già occupato con un altro corso, grazie a questo metodo:

```

public function checkOverlap($start,$end,$idU){
    $dStart = strtotime($start);
    $dEnd = strtotime($end);
    $selectDays = "SELECT start, end FROM execution WHERE id_user=%i";
    $days = $this->connection->query($selectDays,$idU);
    $check = 0;
    foreach ($days as $row) {
        $xstart = strtotime($row[DB_EXECUTION_START]);
        $xend = strtotime($row[DB_EXECUTION_END]);
        if (!((($dStart > $xend) || ($dEnd < $xstart))) {
            $check++;
        }
    }
    return $check != 0;
}

```

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò l'eliminazione di uno svolgimento e l'iscrizione ad un corso da parte dell'utente finale.

Diario di lavoro

Luogo	Mendrisio
Data	25.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto la pagina per iscriversi ad uno svolgimento di un corso e la pagina di gestione delle iscrizioni di un corso.

Nella prima parte della pagina per iscriversi ad un corso possiamo trovare le informazioni del corso e un form di login:

Corso anti-sbandamento

Informazioni
Tipologia: Auto
Descrizione:
Corso obbligatorio patente B.
Materiale necessario:
Patente B

Costi
Prezzo corso: 300.00-
Prezzo pasto (opzionale): 10.00-
(Dati di pagamento vengono inviati per e-mail dopo aver effettuato l'iscrizione)

Dove
Città: Lugano
CAP: 6900
Via: Zurigo 12

Scorica Corso

Login
Esegui il login per compilare il form con i dati utilizzati in una vecchia iscrizione.

Email

Password

[Password dimenticata?](#)

Accedi

Per ricavare le informazioni del corso eseguo questo metodo:

```
public function getAllDataCourse($id)
{
    $selectCourse = "SELECT * FROM course WHERE id=%i";
    $course = $this->connection->query($selectCourse, $id);
    return $course;
}
```

Per quanto riguarda il login utilizzo lo stesso metodo che ho implementato nella pagina di login. Eseguendo il login l'utente avrà il form di iscrizione pre-compilato.

Scorrendo per la pagina si possono vedere le date disponibili di un determinato corso. Selezionata la data mostro a destra gli orari, i posti disponibili, la data di chiusura di un'iscrizione e l'insegnate del corso.

Date

Seleziona la data

28.05.2020 - 30.05.2020

Info Data

Orari

28.05.2020 Ora inizio: 14:00 Ora fine: 15:00

29.05.2020 Ora inizio: 14:00 Ora fine: 15:00

30.05.2020 Ora inizio: 14:00 Ora fine: 15:00

Posti disponibili

1 su 4

Chiusura iscrizione

25.05.2020

Insegnate

Nome: Mattia

Cognome: Toscanelli

Email: gestionecorsi2@yopmail.com

Infine scorrendo nell'ultima parte della pagina è possibile compilare un form di iscrizione al corso.

Compila l'iscrizione

Salva Account ☐ SI ☒ NO

Nome*

Cognome*

Data di nascita*

Via*

Città*

CAP*

Numero mobile*

Numero fisso*

Numero licenza*

Numero nip*

Email*

Pranzo* ☐ SI ☒ NO

Tipologia cibo*

Intolleranze*

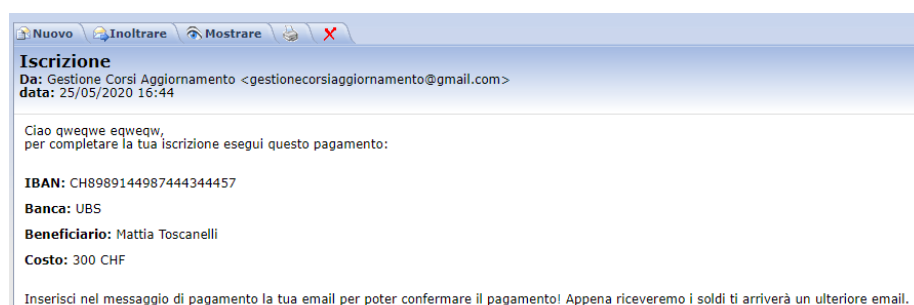
* campi da compilare

☐ Non sono un robot



In fondo all'immagine è presente il captcha v2 di Google. Per implementarlo ho seguito la guida di Google: <https://developers.google.com/recaptcha/docs/display>

Quando ci si iscrive viene inviata questa email, dove vengono mostrati i dati di pagamento:



Una volta conclusa la pagina di iscrizione ho aggiunto la pagina di gestione di uno svolgimento:

Informazioni

Orari
 05.06.2020 Ora inizio: 14:00 Ora fine: 15:00
 06.06.2020 Ora inizio: 14:00 Ora fine: 15:00
 07.06.2020 Ora inizio: 14:00 Ora fine: 15:00

Chiusura iscrizione
 02.06.2020

Posti disponibili
 1 su 4

Iscritti

Questa tabella mostra tutti gli iscritti al corso. È possibile eliminare un'iscrizione (se fatto prima della data di chiusura iscrizione) o confermare/cancellare il pagamento.

Mostra corsi per pagina Cerca:

EMAIL	NOME	COGNOME	PASTO	TIPOLOGIA	INTOLLERANZE	PAGATO
gestioneaggiornamento5@yopmail.com	fsdof	sdfsdf	✓	Nessuna Preferenza	-	<input type="checkbox"/>
luigi.bernasconi@yopmail.com	Luigi	Bernasconi	×	-	-	<input type="checkbox"/>
mattia.toscanelli0325@gmail.com	Mattia	Toscanelli	×	-	-	<input checked="" type="checkbox"/>

Pagina 1 di 1 Precedente **1** Successivo

Aggiungi iscrizione

Nome* Cognome*

Data di nascita* Via*

Nella prima parte dell'immagine vengono nuovamente mostrati gli orari, la data di chiusura iscrizione e i posti disponibili del corso. A seguire è presente una tabella che mostra gli iscritti al corso, con la possibilità di confermare il pagamento di un utente o eliminare l'iscrizione. Infine è presente il form per aggiungere un'iscrizione per un utente che ha chiamato per telefono.

Infine ho aggiunto nel pannello admin la gestione dell'età minima per partecipare ad un corso:

Età minima iscrizione

Grazie a questa tabella può specificare l'età minima per iscriversi ad un corso.

[Modifica giorni](#)

TIPO	VALORE
Età minima	14

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro farò l'eliminazione di uno svolgimento, la pagina personale, la gestione dei permessi e la newsletter.

Luogo	Mendrisio
Data	26.05.2020

Lavori svolti

Durante la giornata di oggi ho quasi concluso il codice del mio progetto. Quasi perché ho un piccolo bug per quanto riguarda l'invio delle email, ogni tanto funziona e ogni tanto no, non mi spiego ancora il perché. Il resto invece, cioè l'eliminazione di uno svolgimento, la pagina personale e la gestione dei permessi sono stati fatti. Inoltre ho implementato che se viene impostato il prezzo del pasto ad un corso a "0" viene rimossa l'opzione per partecipare a quest'ultimo.

Questa è la pagina del profilo:

Profilo

Corsi da fare:

Questa tabella mostra tutti i corsi ancora da fare. In questa tabella ti puoi disiscrivere ad corso se fatto prima della loro scadenza.

Mostra
corsi per pagina

Cerca:

TITOLO	PAGATO	DATA CHIUSURA ISCRIZIONE	INIZIO	FINE	PASTO	TIPOLOGIA
<div>+</div> Corso anti-sbandamento	✓	02.06.2020	05.06.2020	07.06.2020	✗	-
<div>+</div> Modulo 1	✗	27.05.2020	30.05.2020	31.05.2020	✗	-

Pagina 1 di 1

Precedente

1

Successivo

Corsi fatti:

Questa tabella mostra tutti i corsi fatti.



Non hai concluso nessun corso

In questa pagina si possono vedere i corsi ancora da fare, rimuovere l'iscrizione di un corso prima della sua scadenza e visualizzare i corsi fatti (in questo caso non ce ne sono).

L'interfaccia grafica per l'invio delle email per la newsletter è la seguente:

Newsletter

Carica file Nessun file selezionato

B **I** **U** **S** **X'** **X_o** 14 **Lato**  **A**  **T'**

Invia

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi se non quello descritto all'inizio del diario.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto alla mia pianificazione.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro cercherò di sistemare questo piccolo problema descritto all'inizio del diario e cercherò di concludere la parte di implementazione della documentazione.

Diario di lavoro

Luogo	Mendrisio
Data	27.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto sistemato il problema dell'invio email. Infatti PHP di default permette di caricare file con peso massimo di 2MB. Dopo aver effettuato la chiamata con il perito e il formatore ho capito che questa impostazione può essere cambiata dal file php.in:

```
823
824 ; Maximum allowed size for uploaded files.
825 ; http://php.net/upload-max-filesize
826 upload_max_filesize = 2M
827
```

Adesso ho impostato un filesize massimo di 25MB. Inoltre ho aggiunto delle informazioni della email quando un utente si iscrive a un corso. Infatti la email che riceverà avrà un aspetto simile:

Ciao Mattia Toscanelli,
per completare la tua iscrizione esegui questo pagamento:

IBAN: CH8989144987444344457

Banca: UBS

Beneficiario: Mattia Toscanelli

Costo: 300 CHF

Inserisci nel messaggio di pagamento la tua email per poter confermare il pagamento! Appena riceveremo i soldi ti arriverà un'ulteriore email.

Informazioni corso

Titolo: Corso anti-sbandamento

Città: Lugano

Via: Zurigo 12

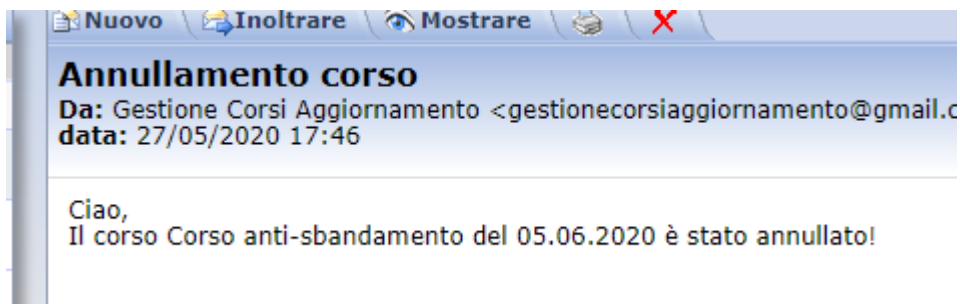
Tipologia: Auto

Infine ho aggiunto che quando un utente rimuove l'iscrizione ad un corso riceve questa email:

data: 27/05/2020 17:22

Ciao Mattia Toscanelli,
Sei stato rimosso dal corso Corso anti-sbandamento!

Oppure se lo svolgimento o il corso viene eliminato questo:



Oltre ad avere aggiunto/sistemato il codice ho anche continuato la documentazione, sono circa a 3/4 della parte implementazione.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione.

Programma di massima per la prossima giornata di lavoro

La prossima giornata di lavoro concluderò la parte di implementazione e aggiungerò i test-case.

Diario di lavoro

Luogo	Mendrisio
Data	28.05.2020

Lavori svolti

Durante la giornata di oggi ho sistemato dei piccoli errori di lessico nei commenti e nell'email inviate dell'applicazione. In seguito ho continuato la documentazione, più precisamente ho finito di documentare la parte di implementazione (cioè ho documentato la pagina gestione corsi/svolgimenti/iscrizioni, la pagina di ricerca, la pagina di iscrizione e la pagina personale). In seguito ho iniziato a fare i test case.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione, ora mi manca da finire i test case, fare il consuntivo (che bene o male è rimasto praticamente invariato rispetto la pianificazione), le conclusioni, l'abstract in italiano e qualche spiegazione nel readme di github per installare l'applicativo.

Programma di massima per la prossima giornata di lavoro

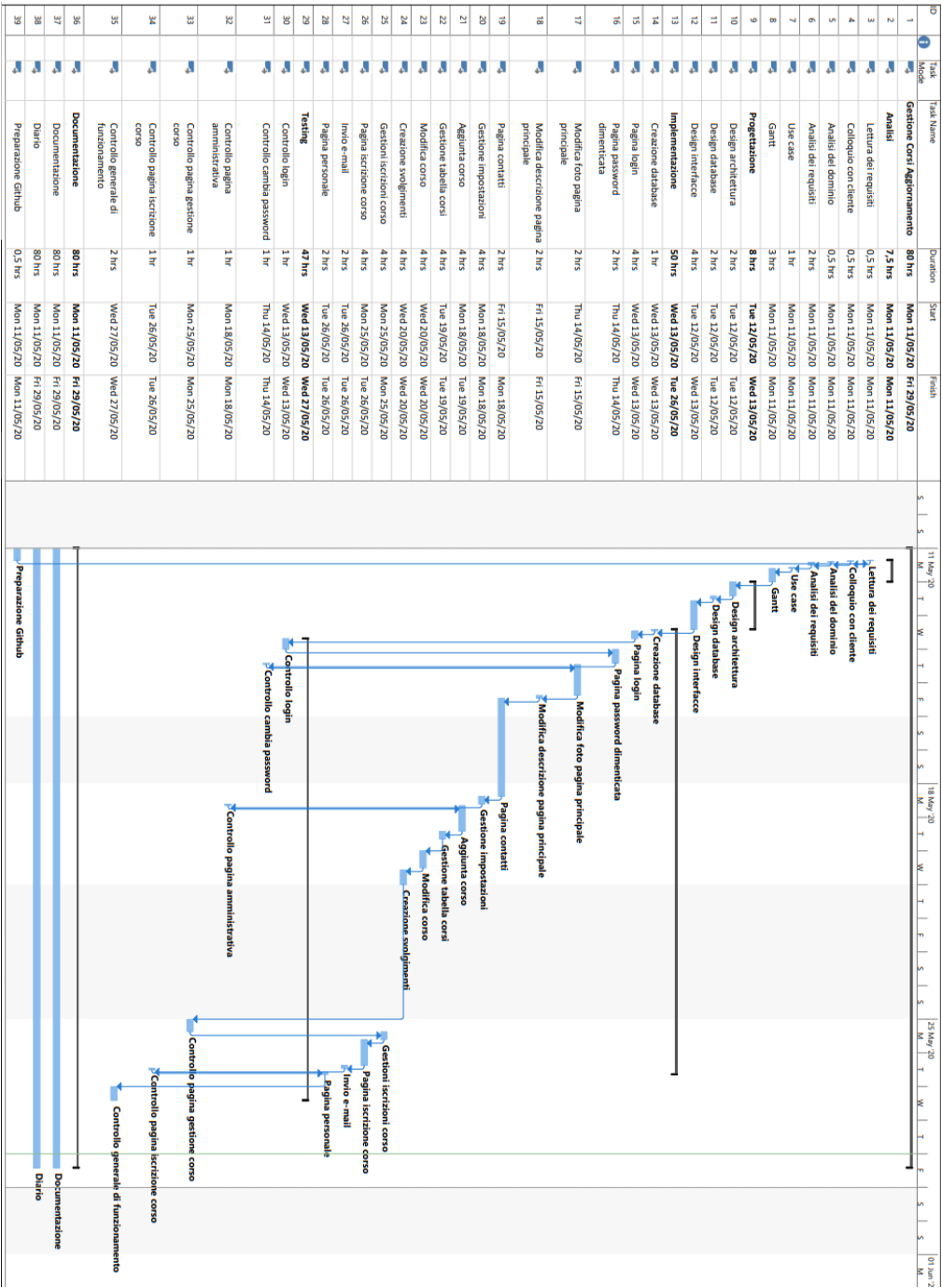
La prossima giornata di lavoro consegnerò il progetto e svolgerò le operazioni descritte nel punto della situazione.

Diario di lavoro

Luogo	Mendrisio
Data	29.05.2020

Lavori svolti

Durante la giornata di oggi ho fatto ho finito le ultime cose e ho consegnato il progetto. Andando in ordine ho concluso il capitolo testcase con i relativi risultati. In seguito ho creato il gantt consuntivo:



Il gantt consuntivo è stato modificato leggermente rispetto a quello preventivo in alcune attività dell'implementazione. Una volta finito il gantt ho fatto scritto le conclusioni del progetto.

Finita la documentazione ho creato la pagina di abstract in italiano, scritto un breve readme su come si installa l'applicazione e ho aggiornato lo script di creazione del database, così da avere una base di prova di utilizzo dell'applicazione.

Problemi riscontrati e soluzioni adottate

Durante la lezione di oggi non ho avuto particolari problemi.

Punto della situazione rispetto alla pianificazione

Sono in linea rispetto la mia pianificazione.

Programma di massima per la prossima giornata di lavoro

Essendo oggi l'ultimo giorno non ci sarà nessuna prossima giornata di lavoro.