Lavoro temporaneo

1 Introduzione 3

1.1 Informazioni sul progetto 3

1.2 Abstract 3

1.3 Scopo 4

2 Analisi 5

2.1 Scopo 5

2.2 Requisiti 5

2.3 Pianificazione 9

2.4 Analisi dei mezzi 10

2.4.1 Software 10

2.4.2 Hardware 10

3 Progettazione 11

3.1 Design della struttura dell’applicativo web 11

3.2 Design dei dati e database 12

3.2.1 Progettazione logica 12

3.2.2 Progettazione fisica 13

3.3 Design delle interfacce 14

3.3.1 Progettazione pagina principale 14

3.3.2 Progettazione form di registrazione 14

3.3.3 Progettazione form di accesso 15

3.3.4 Progettazione pagina amministratore 15

3.3.5 Progettazione pagina crea fatturazione 16

3.3.6 Progettazione pagina visualizza fatturazione 16

3.3.7 Progettazione pagina datore di lavoro 17

3.3.8 Progettazione pagina datore di lavoro (creazione offerta) 17

3.3.9 Progettazione pagina Lavoratore 18

3.3.10 Progettazione pagina Lavoratore (creazione domanda) 18

4 Implementazione 19

4.1 Mail 19

4.1.1 File php.ini 19

4.1.2 File sandamil.php 19

4.2 Database 20

4.2.1 Account per l’accesso 20

4.2.2 Implementazione fisica banca dati 20

4.2.3 Implementazione grafica banca dati 21

4.3 Applicativo Web 22

4.3.1 Struttura cartelle 22

4.3.2 File index.php 22

4.3.3 File .htaccess 23

4.3.4 Cartella config 23

4.3.5 Cartella libs 23

4.3.6 Cartella Controller 26

4.3.7 Cartella Models 38

4.3.8 Cartella Views 43

5 Test 47

5.1 Protocollo di test 47

5.2 Risultati test 52

5.3 Mancanze/limitazioni conosciute 57

6 Consuntivo 58

7 Conclusioni 59

7.1 Sviluppi futuri 59

7.2 Considerazioni personali 59

8 Bibliografia 60

8.1 Sitografia 60

9 Allegati 61

# Introduzione

## Informazioni sul progetto

* Allievo:
  + Samuele Abbà
* Docente:
  + Guido Montalbetti
* Scuola:
  + SAMT Trevano
* Sezione:
  + Informatica
* Materia:
  + M306
* Inizio:
  + 16.09.2021
* Fine:
  + 23.12.2021

## Abstract

* **Situazione iniziale**
  + All’inizio di questo progetto, mi è stato chiesto di realizzare un sistema che possa gestire la domanda e l’offerta di posti di lavoro temporanei. Questo progetto nasce da un bisogno ben specifico. Fino ad ora, non esiste un prodotto o servizio che offre la possibilità di mettere in contatto due o più persone che hanno bisogno di un aiuto o che possono dare un aiuto. In breve: questo progetto è incentrato sull’elaborazione di un portale web che possa gestire comodamente le domande e le offerte per un lavoro temporaneo.
* **Attuazione**
  + Per questo progetto, saranno necessarie buone conoscenze nella programmazione web, poiché complesso nella sua struttura e nelle sue funzionalità. Seguendo i requisiti dati, per questo progetto è stato quindi necessario adoperare una banca dati per immagazzinare delle informazioni, compresa l’implementazione di tutta la sua sicurezza. Sono stati implementati una serie di sistemi che permettano l’aggiunta, la modifica e la rimozione delle varie offerte di lavoro all’interno della banca dati. Questo permettendo agli utenti presenti nel sistema di essere il più comodi, veloci e sicuri nell’utilizzo del prodotto. Vi sarà inoltre la possibilità di caricare dei documenti, quali p.es. Curriculum Vitae, Diplomi, ad ogni richiesta e una volta completata la candidatura, i clienti riceveranno tramite eMail la conferma della loro iscrizione al posto in offerta. L’offerta scadendo permette di fa calcolare al sistema i soldi che il datore di lavoro deve pagare al lavoratore e all’agenzia, inoltre ci saranno degli amministratori che potranno eseguire delle stampe, con inserimento del periodo desiderato, per gli stipendi da consegnare ai collaboratori e la fatturazione al datore di lavoro da parte dell’Agenzia.
* **Risultati**
  + Questo sito è realizzato da zero con i principali linguaggi web (HTML, CSS, PHP). Ho quindi avuto possibilità di lavorare su argomenti già consolidati e argomenti non ancora consolidati, questo mi ha permesso di ampliare il mio bagaglio delle conoscenze durante il progetto. Uno dei punti fondamentali tenuti in considerazione durante il progetto riguarda lo studio per raggiungere la facilità di utilizzo per un utente medio. Questo portale web è difatti utilizzabile e configurabile anche da un’utenza che non possiede conoscenze informatiche. In conclusione, questo progetto è ora l’unico sito funzionante che permetta di dare o prendere lavori temporanei.

## Scopo

* Vedi QdC in allegato

# Analisi

## Scopo

Spesso per poter trovare un lavoro occorre girare molti siti perché quelli presenti sono incompleti e non forniscono tutte le informazioni oppure sono troppo complicati e risultano difficili da usare. Il mio prodotto servirà a quegli utenti che cercano un sito semplice ed intuitivo con cui trovare un lavoro in pochi minuti.

Per poter accedere al mio sito sarà necessario come minimo un dispositivo con accesso ad Internet e un’e-mail e password nel caso in qui si volesse iniziare a pubblicare o accettare un lavoro, le conoscenze per utilizzare il mio servizio sono basilari per poter usare un sito Web.

In questo capitolo sono andato a suddividere i vari requisiti che ho trovato nel QdC, con i rispettivi sotto requisiti.

Come primo requisito ho messo la verifica del sito web, che funzioni correttamente e senza essere loggati, questo permette di vedere le varie offerte di lavoro.

Come secondo requisito ho messo la verifica del form di registrazione e anche la verifica del form di accesso per (amministratori, datori di lavoro e clienti).

Dopodiché come quarto requisito ho messo la verifica del sito web per i clienti nell’ proporsi per le offerte di lavoro.

Dopo come quinto requisito ho messo la verifica del sito web per i datori di lavoro nell’ aggiungere le offerte di lavoro.

Poi come sesto requisito ho messo la fatturazione per i datori di lavoro e per i collaboratori.

Successivamente i requisiti per le stampe; per gli stipendi dei collaboratori e per le fatture dei datori di lavoro.

I prossimi due requisiti concernono la gestione delle richieste dei clienti e delle offerte dei datori di lavoro registrati, è importante definire gli attributi delle richieste di lavoro e delle offerte, come anche la conferma tramite e-mail.

Come decimo requisito vi sarà inoltre la possibilità di caricare dei documenti, quali p.es. Curriculum Vitae, Diplomi, ad ogni offerta e una volta completata la candidatura, i clienti riceveranno tramite eMail la conferma della loro iscrizione al posto in offerta.

## Requisiti

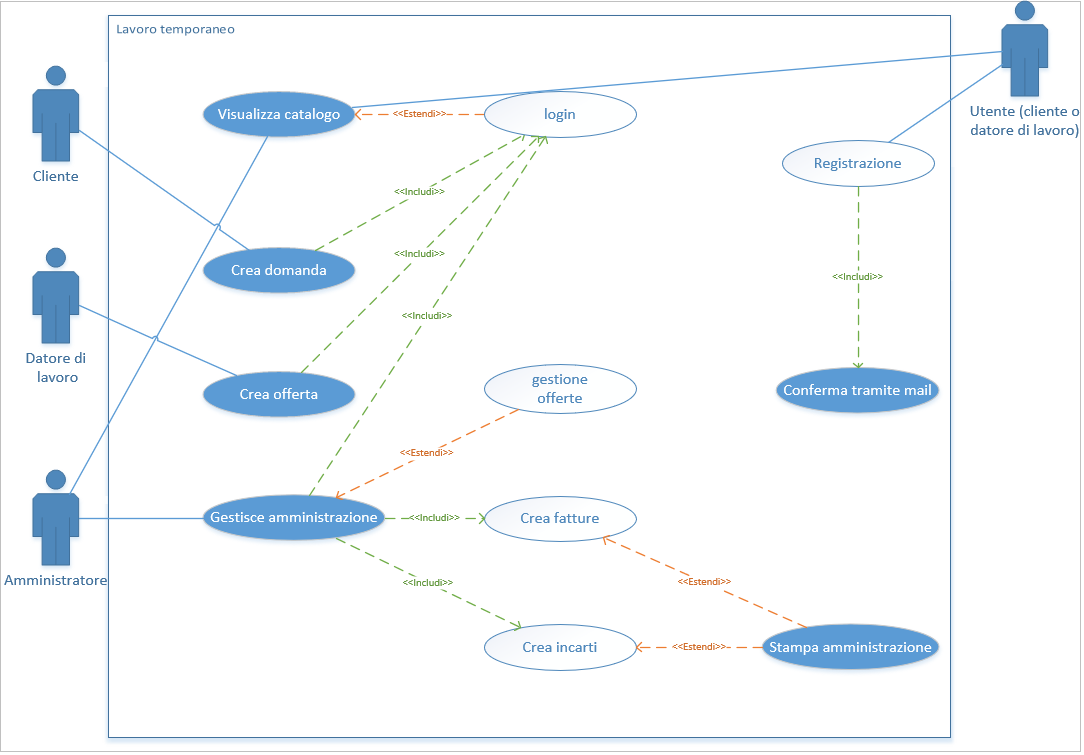
|  |  |
| --- | --- |
|  | **ID: REQ-001** |
| **Nome** | Verifica del sito web senza login |
| **Priorità** | 1 |
| **Versione** | 1.0 |
| **Note** | È necessario verificare che il sito web funzioni correttamente e senza essere loggati visualizzare i lavori disponibili. |

|  |  |  |
| --- | --- | --- |
|  | | **ID: REQ-002** |
| **Nome** | | Verifica del sito web con registrazione e accesso |
| **Priorità** | | 1 |
| **Versione** | | 1.0 |
| **Note** | | È necessario verificare che il sito web funzioni correttamente effettuando le due opzioni di accesso e di registrazione. |
| **Sotto Requisiti** | | |
| **0021** | Verifica del funzionamento di: form di registrazione, conferma tramite email. | |
| **0022** | Verifica del funzionamento di: form di accesso, con reindirizzamento alla propria pagina. | |

|  |  |  |
| --- | --- | --- |
|  | | **ID: REQ-003** |
| **Nome** | | Verifica risultato del login |
| **Priorità** | | 1 |
| **Versione** | | 1.0 |
| **Note** | | È necessario verificare il corretto risultato del login |
| **Sotto Requisiti** | | |
| **0031** | Si possano creare richieste. (se si è clienti) | |
| **0032** | Si possano creare offerte. (se si è datori di lavoro) | |
| **0033** | Si possano gestire le offerte. (se si è amministratori) | |

|  |  |  |
| --- | --- | --- |
|  | | **ID: REQ-004** |
| **Nome** | | Verifica gestione delle spese |
| **Priorità** | | 1 |
| **Versione** | | 1.0 |
| **Note** | | È necessario verificare che le fatture funzionino correttamente. |
| **Sotto Requisiti** | | |
| **0041** | Verifica fine del lavoro. | |
| **0042** | Calcolo spese del datore di lavoro. | |
| **0043** | Calcolo stipendio dei collaboratori. | |

Use case

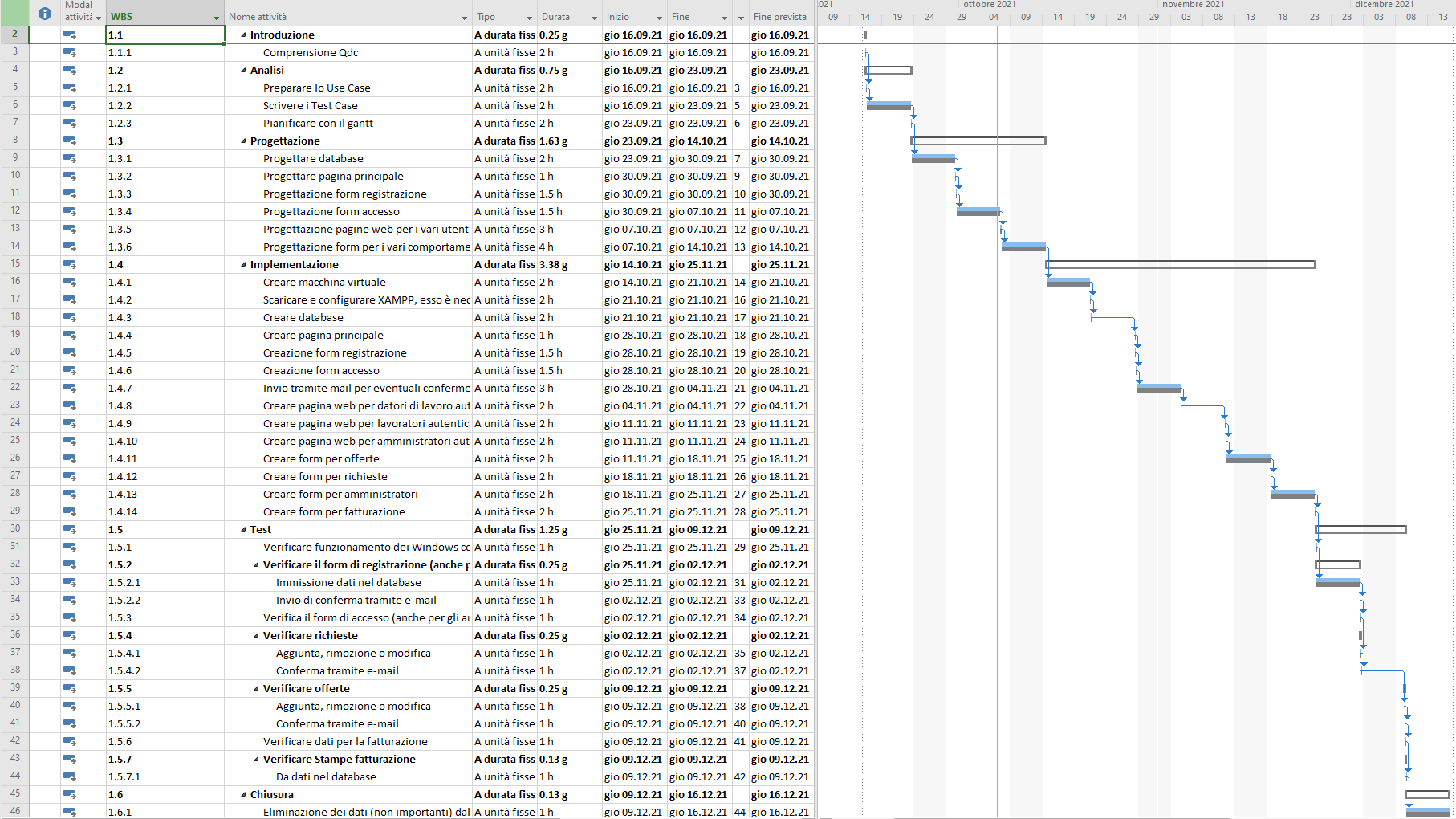


All’interno dell’applicativo ho inserito 3 tipi diversi di utente, più specificatamente:

* Amministratore: Gli utenti appartenenti a questo gruppo sono presenti nel database e i compiti che avranno saranno quelli di preparare le fatture, gli incarti e gestire il sistema. Per fare ciò gli amministratori dovranno loggarsi.
* Datore di lavoro e cliente: Gli utenti appartenenti a questo gruppo sono presenti nel database e essi potranno creare, modificare e cancellare le offerte oppure, se clienti fare una richiesta ad un lavoro e poi successivamente modificarla o cancellarla.
* In ogni caso se non si è loggati si può comunque guardare il catalogo e vedere tutti i lavori offerti dal sito.

## Pianificazione

Di seguito c’è la pianificazione preventiva, nel Gantt preventivo è stimato il tempo necessario per completare le task e le rispettive sotto task del progetto.



## Analisi dei mezzi

Per questo progetto mi sarà fornito 1 PC della scuola, sul quale ci sarà un sistema operativo Windows10Pro.

### Software

I software utilizzati per questo progetto sono XAMPP (Apache, MySQL, Php e Perl) per la creazione del server web e la sua gestione, bisogna anche installare le VC16 per il corretto funzionamento di Apache.

Per questo progetto utilizzerò inoltre i software Firefox/Google e notepad/notepad++ e MySQL Workbench.

### Hardware

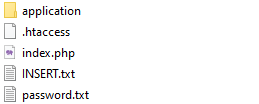
L’hardware coinvolto nel progetto è un PC fornito dalla scuola, esso ha ottime prestazioni ma come limitazioni ha il fatto che è collegato in una rete interna quindi il sito potrà venir utilizzato unicamente nella rete scolastica. Per questo progetto utilizzerò inoltre una macchina virtuale, un SSD e una serie di chiavette.

# Progettazione

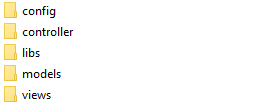
## Design della struttura dell’applicativo web

La struttura dell’applicativo WEB si basa su MVC, un framework base ma che basterà per il progetto.

La struttura delle cartelle è descritta di seguito e all’interno sono presenti i seguenti elementi:



E all’interno della cartella application sono contenute le cartelle seguenti



* **Application**: Cartella contenente tutte le cartelle e i file necessari per il funzionamento del servizio.
* **Config**: Dentro questa cartella è presente un file di configurazione contenente il codice per mostrare

tutti gli errori sulla pagina web e delle costanti che verranno utilizzate durante lo sviluppo, come l’URL, essa serve per raggiungere l’applicativo WEB (in locale).

* **Controller**: Questa cartella, come si può facilmente intuire, conterrà tutti i controller. Essi sono i file

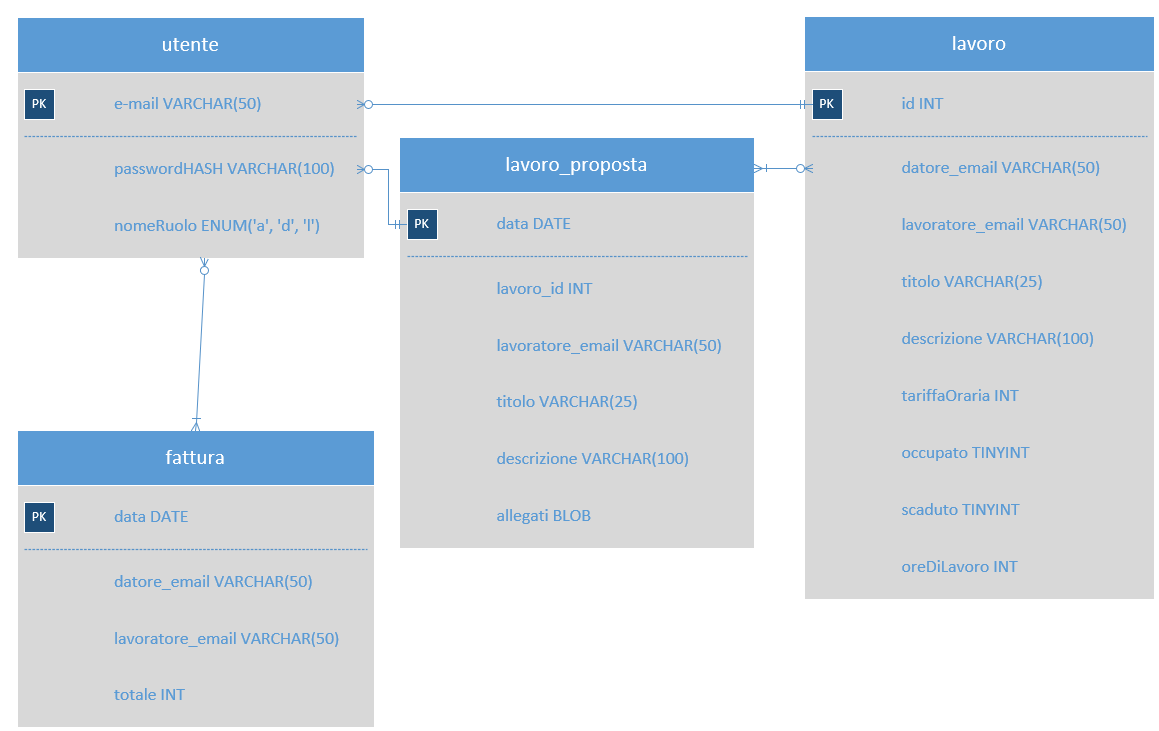
che si occupano di prendere i dati dalle classi e darli alle views. Questi file sono pensati per avere una comunicazione specifica tra i dati effettivi e quello che vede l’utente, avendo così un codice

estremamente più ordinato.

* **Libs**: In questa cartella ci saranno tutte le librerie che verranno utilizzate e saranno necessarie ai Controller.
* **Models**: All’interno ci saranno tutte le classi che si occuperanno di gestire i dati, esse si interfacceranno con il database ed estrapoleranno i dati dal suo interno, oltre alle funzioni più complesse.
* **Views**: Qui ci sono i file dove l’utente s’interfaccia, ad esempio la pagina principale.

## Design dei dati e database

### Progettazione logica



Di seguito una descrizione di tutte le tabelle del database.

* Utente: questa tabella contiene tutti gli utenti del database, ogni utente avrà una e-mail, una passwordHash e un nomeRuolo.
* Lavoro\_Proposta: questa tabella contiene tutte le proposte che verranno fatte dai lavoratori alle offerte di lavoro dei datori di lavoro esse sono caratterizzate da (data, lavoro\_id, lavoratore\_email, titolo, descrizione e allegati).
* Lavoro: questa tabella contiene tutti i lavori offerti dai datori di lavoro essi sono caratterizzate da (id, datore\_email, lavoratore\_email, titolo, descrizione, oreDiLavoro, tariffaOraria, se occupato e se scaduto).
* Fattura: questa tabella contiene tutte le fatture, che si fanno dai datori di lavoro ai lavoratori e ai collaboratori, questo grazie ai campi data, datore\_email, lavoratore\_email e totale.

### Progettazione fisica

CREATE DATABASE Lavoro\_temporaneo;

USE DATABASE Lavoro\_temporaneo;

tabella 1

CREATE TABLE utente(

email VARCHAR (50) PRIMARY KEY,

passwordHash VARCHAR (255) NOT NULL,

nomeRuolo ENUM('amministratore', 'datore', 'lavoratore') NOT NULL

);

tabella 2

CREATE TABLE fattura(

data DATETIME PRIMARY KEY,

datore\_email VARCHAR (50) NOT NULL,

lavoratore\_email VARCHAR (50) NOT NULL,

totale INT,

FOREIGN KEY (datore\_email)

REFERENCES utente(email)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email)

REFERENCES utente(email)

ON DELETE CASCADE

ON UPDATE CASCADE

);

tabella3

CREATE TABLE lavoro(

id INT AUTO\_INCREMENT PRIMARY KEY,

datore\_email VARCHAR (50) NOT NULL,

lavoratore\_email VARCHAR (50),

titolo VARCHAR (25) NOT NULL,

descrizione TEXT NOT NULL,

tariffaOraria INT NOT NULL,

occupato TINYINT NOT NULL,

scaduto TINYINT NOT NULL,

oreDiLavoro INT NOT NULL,

FOREIGN KEY (datore\_email)

REFERENCES utente(email)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email)

REFERENCES utente(email)

ON DELETE CASCADE

ON UPDATE CASCADE

);

tabella 4

CREATE TABLE lavoro\_proposta(

data DATETIME PRIMARY KEY,

lavoro\_id INT NOT NULL,

lavoratore\_email VARCHAR (50) NOT NULL,

titolo VARCHAR (25) NOT NULL,

descrizione TEXT NOT NULL,

allegati BLOB NOT NULL,

FOREIGN KEY (lavoro\_id)

REFERENCES lavoro(id)

ON DELETE CASCADE

ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email)

REFERENCES utente(email)

ON DELETE CASCADE

ON UPDATE CASCADE

);

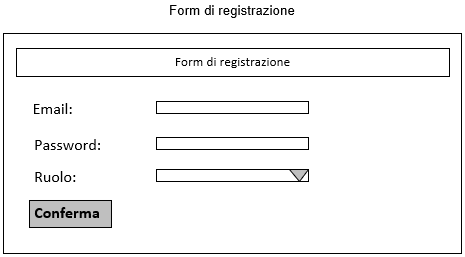
## Design delle interfacce

### Progettazione pagina principale



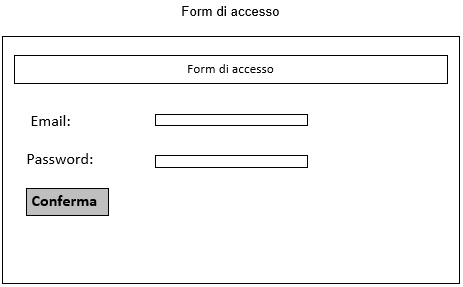
Come primissima interfaccia riporto quella che si visualizza non appena si apre il sito web senza essersi loggati, dove si potrà effettuare l’accesso, la registrazione o visualizzare semplicemente tutte le offerte di lavoro.

### Progettazione form di registrazione



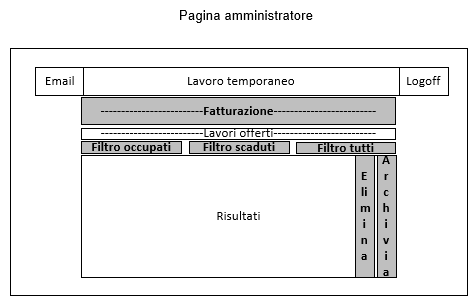
Come seconda interfaccia ho riportato quella della registrazione dove tramite di essa è possibile creare un account (e-mail e password) e definire un ruolo (datore di lavoro o lavoratore).

### Progettazione form di accesso



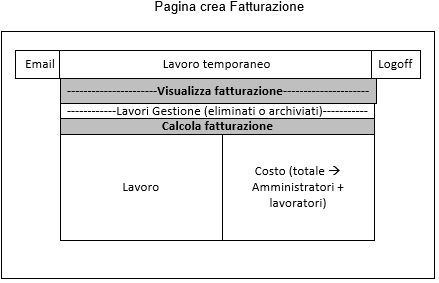
Come terza interfaccia ho riportato quella dell’accesso dove tramite di essa è possibile accedere al proprio account (e-mail e password).

### Progettazione pagina amministratore



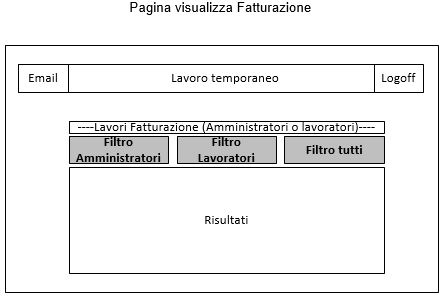
Come quarta interfaccia ho riportato quella degli amministratori una volta effettuato l’accesso, tramite di essa sarà possibile visualizzare tutte le offerte di lavoro e filtrarle (occupato, scaduto, tutti anche eliminati), è anche possibile eliminarle e quindi archiviarle e infine da questa interfaccia si può accedere all’interfaccia della fatturazione.

### Progettazione pagina crea fatturazione



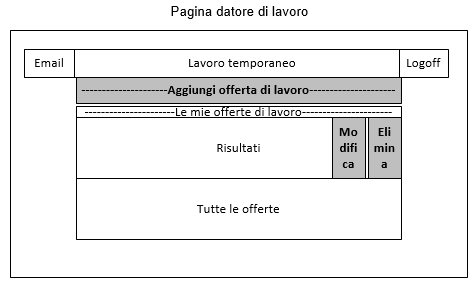
Come quinta interfaccia ho riportato quella per creare le fatture, da essa si possono vedere i lavori archiviati e di conseguenza si può calcolare il costo per gli amministratori e per i lavoratori. Inoltre sarà possibile collegarsi alla pagina che permetterà di vedere tutte le varie fatture per i datori di lavoro e per i collaboratori.

### Progettazione pagina visualizza fatturazione



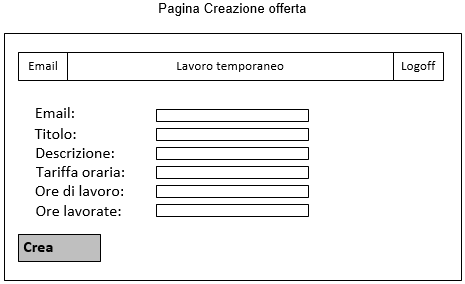
Come sesta interfaccia ho riportato quella per visualizzare le fatture, da essa si possono vedere tutte le fatture e grazie ad alcuni filtri è possibile visualizzare le paghe destinate agli amministratori e ai lavoratori date dai datori di lavoro.

### Progettazione pagina datore di lavoro



Come settima interfaccia ho riportato quella che visualizza il datore di lavoro una volta loggato nel sito, esso può vedere le sue offerte, queste ultime possono essere modificate o eliminate, oltre alle proprie offerte di lavoro si vedono anche quelle già presenti nel sito, infine cliccando sul bottone aggiungi offerta di lavoro si viene reindirizzati alla pagina per la creazione delle offerte e li è possibile crearne delle nuove.

### Progettazione pagina datore di lavoro (creazione offerta)



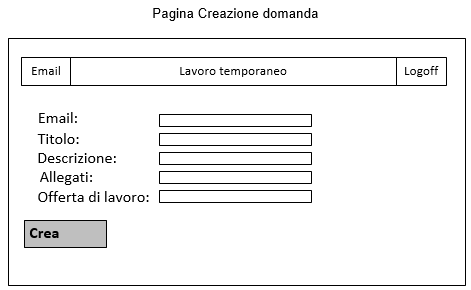
Come ottava interfaccia ho riportato quella che permette di creare un’offerta di lavoro, per farlo bisogna inserire l’e-mail del datore di lavoro, un titolo, una descrizione, una tariffa oraria, le ore di lavoro e successivamente le ore lavorate dal lavoratore. Cliccando su Crea viene creata l’offerta di lavoro.

### Progettazione pagina Lavoratore



Come nona interfaccia ho riportato quella che visualizza il lavoratore una volta loggato nel sito, esso può visualizzare le sue proposte di lavoro, queste ultime possono essere modificate o eliminate, inoltre si vede anche quelle già presenti nel sito, infine cliccando sul bottone aggiungi proposta di lavoro si viene reindirizzati alla pagina per proporsi alle offerte e li è possibile creare delle nuove domande di lavoro.

### Progettazione pagina Lavoratore (creazione domanda)



Come decima interfaccia ho riportato quella che permette di creare una domanda ad una offerta di lavoro, per farlo bisogna inserire l’e-mail del lavoratore, un titolo, una descrizione, allegare file se necessario, selezionare il lavoro interessato. Cliccando su Crea viene creata la domanda all’offerta di lavoro.

# Implementazione

## Mail

Per configurare Php per l’invio della mail tramite l’apposita funzione mail(), bisogna configurare correttamente il file php.ini e il file sandmail.ini come viene illustrato nelle sezioni sottostanti.

### File php.ini

In questo file bisogna descrivere l’account del mittente, la mail e il riferimento al file sendmail, dove in esso sono scritti tutti i dettagli per l’invio delle mail.

* Sendmail\_from = nome mittente
* Sendmail\_path = riferimento al file

### File sandamil.php

In questo file bisogna mettere quanto segue per configurare correttamente l’invio tramite mail (gmail) dall’account LavoroTemporaneoSAMT.

* smtp\_server=smtp.gmail.com
* smtp\_port=587
* smtp\_ssl=tls
* auth\_username=LavoroTemporaneoSAMT@gmail.com
* auth\_password=LavoTempSAMT1234!
* force\_sender=LavoroTemporaneoSAMT@gmail.com
* error\_logfile=error.log
* debug\_logfile=debug.log
* hostname=localhost

## Database

### Account per l’accesso

L’account di accesso al database è user\_lavoro\_temporaneo e la sua password è LavoroTemporaneo1234!. Esso appartiene al db lavoro temporaneo, su di esso ha permessi completi, sia sul db che sulle tabelle.

### Implementazione fisica banca dati

Il database è stato implementato seguendo lo schema ER creato nel capitolo di progettazione, anche se purtroppo sono stato costretto a modificare quasi tutte le tabelle e qualche campo, poiché non avevo pensato a come gestire determinati dati nel database, a conservare i dati se rimossi e a mettere una data per riuscire in seguito a fare delle verifiche. Di seguito c’è lo script per creare il nuovo db completo.

Tabella utente:

CREATE TABLE utente (

email varchar (50) NOT NULL,

passwordHash varchar (255) NOT NULL,

nomeRuolo enum('amministratore','datore','lavoratore') NOT NULL,

PRIMARY KEY (`email`)

)

Tabella lavoro:

CREATE TABLE lavoro (

id int (11) NOT NULL AUTO\_INCREMENT,

datore\_email varchar (50) NOT NULL,

lavoratore\_email varchar (50) DEFAULT NULL,

titolo varchar (25) NOT NULL,

descrizione text NOT NULL,

tariffaOraria int (11) NOT NULL,

occupato tinyint (4) NOT NULL,

scaduto tinyint (4) NOT NULL,

oreDiLavoro int (11) NOT NULL,

data datetime NOT NULL,

archiviato tinyint (4) NOT NULL,

PRIMARY KEY (id),

FOREIGN KEY (datore\_email) REFERENCES utente(email) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email) REFERENCES utente(email) ON DELETE CASCADE ON UPDATE CASCADE

)

Tabella lavoro\_proposta:

CREATE TABLE lavoro\_proposta (

data datetime NOT NULL,

lavoro\_id int (11) NOT NULL,

lavoratore\_email varchar (50) NOT NULL,

titolo varchar (25) NOT NULL,

descrizione text NOT NULL,

allegati blob NOT NULL,

archiviato tinyint (4) DEFAULT NULL,

PRIMARY KEY (data),

FOREIGN KEY (lavoro\_id) REFERENCES lavoro(id) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email) REFERENCES utente(email) ON DELETE CASCADE ON UPDATE CASCADE

)

Tabella fattura:

CREATE TABLE fattura (

data datetime NOT NULL,

datore\_email varchar (50) NOT NULL,

lavoratore\_email varchar (50) NOT NULL,

totale int (11) DEFAULT NULL,

PRIMARY KEY (data),

FOREIGN KEY (datore\_email) REFERENCES utente(email) ON DELETE CASCADE ON UPDATE CASCADE,

FOREIGN KEY (lavoratore\_email) REFERENCES utente(email) ON DELETE CASCADE ON UPDATE CASCADE

)

### Implementazione grafica banca dati

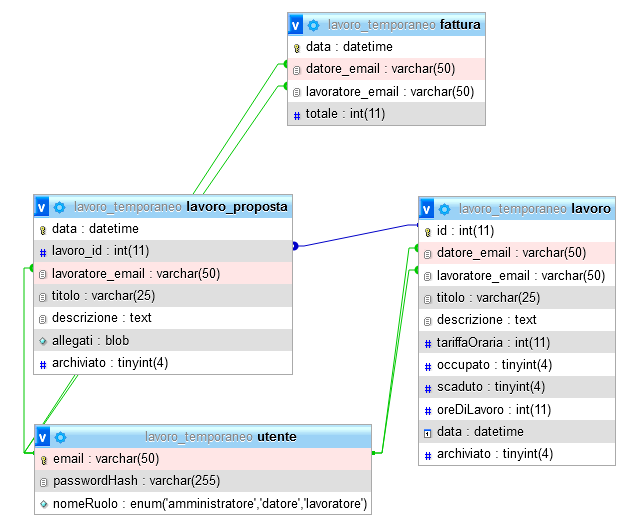
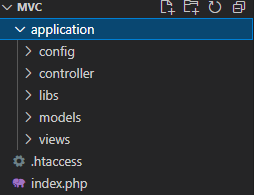


Figura - implementazione parte grafica db

## Applicativo Web

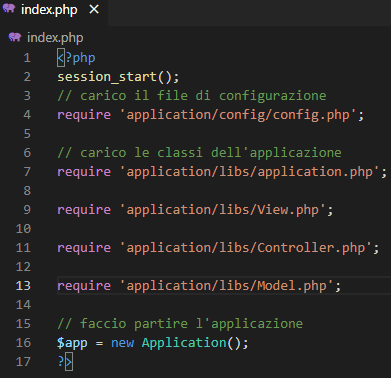
### Struttura cartelle

Per funzionare questo applicativo si basa un MVC, esso necessita di alcuni file di configurazione molto importanti e una struttura ben precisa. Deve contenere una struttura di cartelle (config, controller, libs, models e views) e avere i file .htaccess e index.php al di fuori della struttura di cartelle (x applicativo).



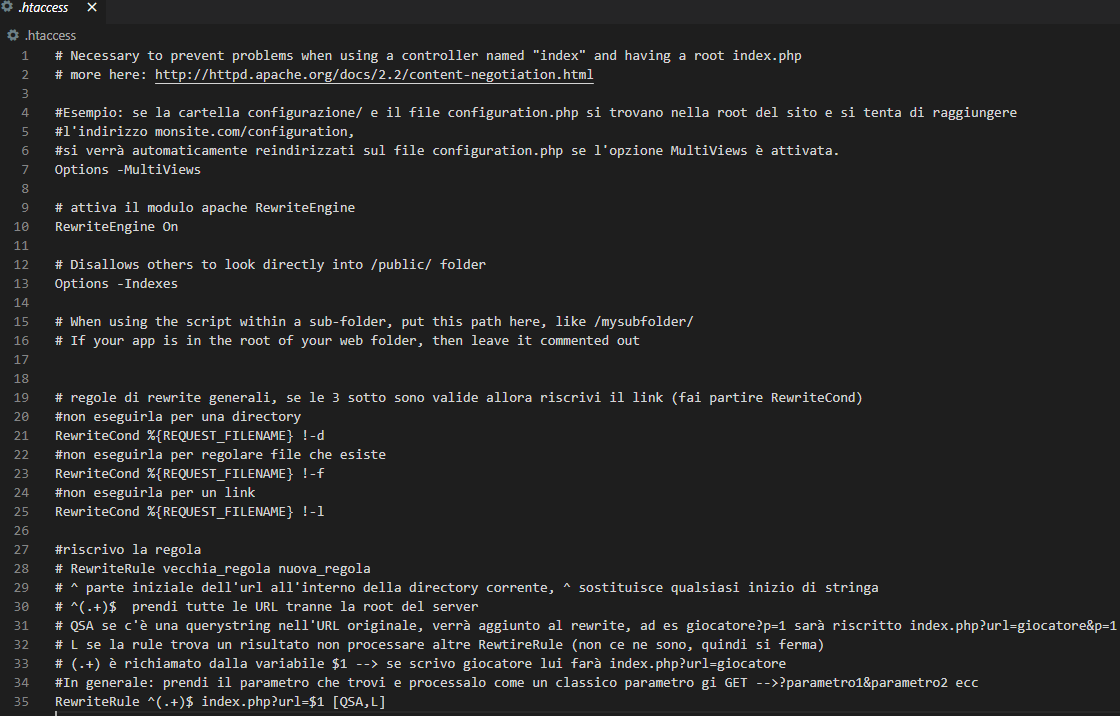
### File index.php

Il file index.php permette di creare il file base sul quale si basa poi tutto l’applicativo; infatti, esso permette di includere tutte le librerie e i file di configurazione, inizializza la sessione e inoltre crea l’oggetto applicazione.



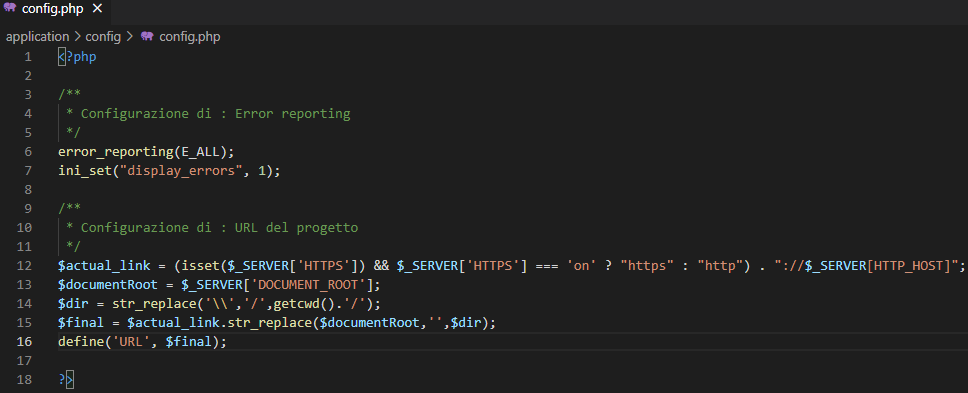
### File .htaccess

Il file .htaccess è necessario per prevenire problemi quando si utilizza un controller "index" e con un root index.php. Questo è un semplice file di testo contenenti le direttive di apache per la sua configurazione.



### Cartella config

Nella cartella config c’è il file config.php, dove in esso è contenuto il codice necessario alla configurazione degli errori e del URL. URL è una variabile super globale che viene utilizzata per tutti i riferimenti interni.



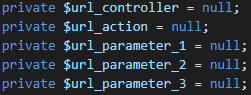
### Cartella libs

Nella cartella libs ci sono tutti i file che contengono le varie classi che necessita un sistema basato su MVC ovvero application, model, view e controller. In questa cartella sono contenuti i file seguenti.

#### File application.php

Il file application.php contiene la classe Application, essa viene inizializzata nel file index.php visto in precedenza. Questa classe è necessaria per splittare l’url e tramite questo sistema richiamare un determinato controller, un metodo e da uno a tre parametri specificati, inoltre se non esisterà quello che si cerca nella barra di ricerca si verrà direttamente reindirizzati al controller della pagina index.php. Dalla classe Application si possono descrivere i metodi e gli attributi che seguono:

* l’attributo url\_controller è utile per conservare il controller al quale ci riferiamo.
* l’attributo url\_action è utile per conservare il metodo del controller.
* I 3 attributi parametri sono utili da passare al metodo del controller.



* Il metodo costruttore richiama il metodo splitUrl, che permette di dividere l’url e assegnare alle variabili citate in precedenza il valore corrispettivo, il costruttore invece serve per chiamare fisicamente il controller, il metodo e gli attributi che sono assegnati alle variabili e in un caso di errore esso ricaricherebbe il controller index.



* Il metodo splitUrl che permette di dividere l’url correttamente le informazioni date, validarle e assegnare alle variabili citate in precedenza il valore corrispettivo.



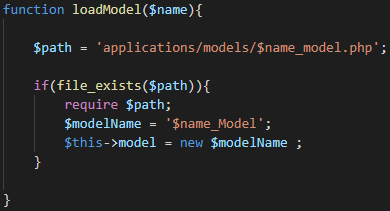
#### File Controller.php

Il file Controller.php contiene la classe Controller, essa è necessaria da estendere in ogni Controller che si va a creare. Questa classe contiene due metodi:

* Il metodo costruttore che permette di creare una nuova view.

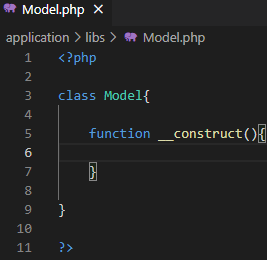


* Il metodo loadModel che permette di implementare un model, dato il nome come parametro.



#### File Model.php

Il file Model.php contiene la classe Model, essa è necessaria da estendere in ogni Model che si va a creare. Questa classe contiene un costruttore vuoto, essa è comunque esistente per il futuro in caso di aggiornamenti.



#### File View.php

Il file View.php contiene la classe View, essa non è necessaria da estendere in ogni View ma viene invece utilizzata nei controller per richiedere la View alla quale ci si riferisce. Questa classe contiene un costruttore vuoto e un metodo chiamato render utile per permettere di includere nella pagina del controller la view fisica.

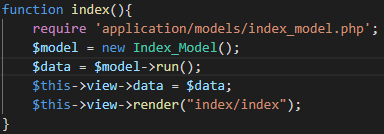


### Cartella Controller

Nella cartella Controller ci sono tutti i file che contengono le varie classi controller che servono per prendere i dati da una view e darli ad un model, per poi riprenderli dal model e ridarli alla view. In questa cartella sono contenuti i file seguenti.

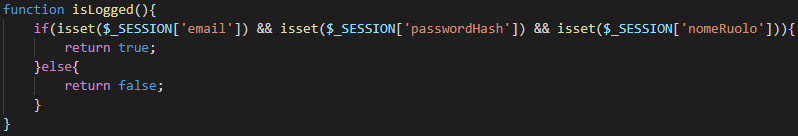
#### File Index.php

Questo file contiene la classe Index che estende la classe controller. Questa classe ha due metodi uno è il costruttore che fa riferimento al costruttore della super classe e l’altro è il metodo index che include il file contenente la classe Index\_model, successivamente viene creato un oggetto di tipo Index\_model, dopo viene richiamato il metodo run del Index\_model, poi viene passato il suo risultato ad una variabile contenuta nella view e infine viene reindirizzata la pagina index.



#### File session.php

Questo file contiene la classe session che estende la classe controller. Questa classe ha due metodi uno è il costruttore vuoto e l’altro è il metodo isLogged, questo metodo ritorna true se sono settate le variabili session e-mail, passwordHash e nomeRuolo.



#### File connection.php

Questo file contiene tutte le variabili necessarie per creare la connessione al database e un oggetto contenente la connessione fisica.

Immagine che contiene testo

Descrizione generata automaticamente

#### File registrazione.php

Questo file contiene la classe Registrazione che estende la classe controller. E i metodi index, rerender, test, validationEmail, validationPassword e test\_input.

* Il metodo index è utile per reindirizzare la pagina registrazione.php.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo rerender è utile per reindirizzare al controller Accesso.



* Il metodo test è utile per testare e validare i dati immessi per la registrazione, se sono corretti  
  viene chiamato un metodo per inserirli nel database altrimenti vengono dati degli errori.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo validationEmail è utile per validare l’e-mail passata come parametro, se è valida ritorna true altrimenti ritorna false.

Immagine che contiene testo

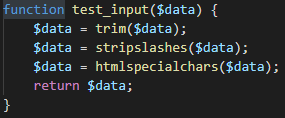
Descrizione generata automaticamente

* Il metodo validationPassword è utile per validare la password passata come parametro, se è valida ritorna true altrimenti ritorna false.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo testinput è utile per testare e validare il parametro stringa passato al metodo e ritorna la stringa validata.



#### File accesso.php

Questo file contiene la classe Accesso che estende la classe controller. E i metodi index, rerender, test, validationEmail, validationPassword e test\_input.

* Il metodo index è utile per reindirizzare la pagina accesso.php.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo rerender è utile per reindirizzare al controller Registrazione.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo test è utile per testare e validare i dati immessi per l’accesso, se sono corretti  
  viene chiamato un metodo per reindirizzare l’utente nella propria pagina altrimenti vengono dati degli errori.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo validationEmail è utile per validare l’e-mail passata come parametro, se è valida ritorna true altrimenti ritorna false.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo validationPassword è utile per validare la password passata come parametro, se è valida ritorna true altrimenti ritorna false.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo testinput è utile per testare e validare il parametro stringa passato al metodo e ritorna la stringa validata.

Immagine che contiene testo

Descrizione generata automaticamente

#### File amministratori.php

Questo file contiene la classe Amministratori che estende la classe controller. E i metodi index, modifica, elimina, esegui, esci, creaFattura, calcolaFattura e visualizzazioneConFiltri.

* Il metodo index è utile per reindirizzare la pagina amministratori.php, se il login è stato effettuato altrimenti si viene reindirizzati alla pagina di accesso.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modifica è utile per modificare un lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo elimina è utile per eliminare un lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esegui è utile per permette di eseguire un metodo in base al bottone che è stato schiacciato.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esci è utile per permette di uscire dalla sessione.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo creaFattura è utile per permette di creare una fattura.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo calcolaFattura è utile per permette di calcolare il costo della fattura.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo visualizzaLavoriConFiltro è utile per permette di avere delle visualizzazioni differenti della tabella dei lavori nella view.

Immagine che contiene testo

Descrizione generata automaticamente

#### File datoriDiLavoro.php

Questo file contiene la classe DatoriDiLavori che estende la classe controller. E i metodi index, modifica, elimina, esegui, accettaRichiesta, rifiutaRichiesta, accettaORifiutaRichiestaDiLavoro, esci, aggiungiOffertaDiLavoro, aggiungi e test\_input.

* Il metodo index è utile per reindirizzare la pagina datoriDiLavoro.php, se il login è stato effettuato altrimenti si viene reindirizzati alla pagina di accesso.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modifica è utile per modificare un lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo elimina è utile per eliminare un lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esegui è utile per permette di eseguire un metodo in base al bottone che è stato schiacciato.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo accettaRichiesta è utile, infatti permette di accettare una richiesta ad un lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo rifiutaRichiesta è utile, infatti permette di rifiutare una richiesta ad un lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo accettaORifiutaRichiestaDiLavoro è utile per permette di eseguire un metodo in base al bottone che è stato schiacciato.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esci è utile per permette di uscire dalla sessione.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo aggiungiOffertaDiLavoro è utile per venire reindirizzati alla pagina per l’aggiunta delle offerte di lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo aggiungi è utile per aggiungere un offerte di lavoro, dopo una validazione degli input.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo testinput è utile per testare e validare il parametro stringa passato al metodo e ritorna la stringa validata.

Immagine che contiene testo

Descrizione generata automaticamente

#### File lavoratori.php

Questo file contiene la classe Lavoratori che estende la classe controller. E i metodi index, modifica, elimina, esegui, esci, aggiungiRichiestaDiLavoro, aggiungi e test\_input.

* Il metodo index è utile per reindirizzare la pagina lavoratori.php, se il login è stato effettuato altrimenti si viene reindirizzati alla pagina di accesso.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modifica è utile per modificare una offerta di lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo elimina è utile per eliminare una offerta di lavoro, questo metodo riceve un parametro numerico e non restituisce niente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esegui è utile per permette di eseguire un metodo in base al bottone che è stato schiacciato.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo esci è utile per permette di uscire dalla sessione.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo aggiungiRichiestaDiLavoro è utile per venire reindirizzati alla pagina per l’aggiunta delle richieste per un lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo aggiungi è utile per aggiungere una richiesta per un lavoro, dopo una validazione degli input.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo testinput è utile per testare e validare il parametro stringa passato al metodo e ritorna la stringa validata.

Immagine che contiene testo

Descrizione generata automaticamente

### Cartella Models

Nella cartella Models ci sono tutti i file che contengono le varie classi models che vengono richiamate dai controller e servono per prendere i dati dal db e ridarli al controller. In questa cartella sono contenuti i file seguenti.

#### File index\_model.php

Questo file contiene la classe Index\_Model che estende la classe Model. E il metodo run.

* Il metodo run è utile per ottenere tutti i dati non archiviati all’interno della tabella lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

#### File registrazione\_Model.php

Questo file contiene la classe Registrazione\_Model che estende la classe Model. E i metodi insertData e extractData.

* Il metodo insertData è utile per inserire nella tabella utente un nuovo utente se non esistente. Con i parametri forniti.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo extractData è utile per estrarre dalla tabella utente tutti gli utenti che soddisfano i criteri dei parametri forniti.

Immagine che contiene testo, schermo, screenshot

Descrizione generata automaticamente

#### File accesso\_Model.php

Questo file contiene la classe Accesso\_Model che estende la classe Model. E i metodi openSession e extractData.

* Il metodo openSession è utile per aprire una nuova sessione se l’utente è esistente nella tabella utente. Con i parametri forniti.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo extractData è utile per estrarre dalla tabella utente tutti gli utenti che soddisfano i criteri dei parametri forniti.

Immagine che contiene testo

Descrizione generata automaticamente

#### File amministratori\_Model.php

Questo file contiene la classe Amministratori\_Model che estende la classe Model. E i metodi getLavoriOccupati, getLavoriScaduti, getLavori, modificaLavoro, eliminaLavoro, getFatture, calcolaFattura e creaFattura.

* Il metodo getLavoriOccupati è utile per avere tutti i lavori occupati che non sono ancora stati archiviati.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo getLavoriScaduti è utile per avere tutti i lavori scaduti che non sono ancora stati archiviati.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo getLavori è utile per avere tutti i lavori.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modificaLavoro è utile per modificare un lavoro con i parametri passati al metodo.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo eliminaLavoro è utile per eliminare un lavoro con i parametri passati al metodo.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo getFatture è utile per ottenere tutte le fatture.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo calcolaFattura è utile per ottenere tutti i dati relativi al calcolo del costo di un lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo creaFattura è utile per inserire una nuova fattura nella tabella fattura.

Immagine che contiene testo

Descrizione generata automaticamente

#### File datoriDiLavoro\_Model.php

Questo file contiene la classe DatoriDiLavoro\_Model che estende la classe Model. E i metodi getLavoriDatoriDiLavoro, getOfferteDiLavoro, modificaOffertaDiLavoro, eliminaOffertaDiLavoro, accettaRichiestaDiLavoro, rifiutaRichiestaDiLavoro, lavoroNonAncoraEsistente e aggiungiLavoro.

* Il metodo getLavoriDatoriDiLavoro è utile per avere tutti i lavori non scaduti di un determinato utente.

Immagine che contiene testo

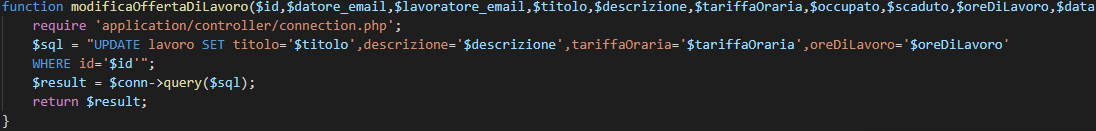
Descrizione generata automaticamente

* Il metodo getOfferteDiLavoro è utile per avere tutte le proposte ad un lavoro di un determinato utente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modificaOffertaDiLavoro è utile per modificare un’offerta di lavoro grazie ai parametri forniti dal metodo.



* Il metodo eliminaOffertaDiLavoro è utile per eliminare un’offerta di lavoro grazie ai parametri forniti dal metodo.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo accettaRichiestaDiLavoro è utile per accettare un’offerta di lavoro grazie ai parametri forniti dal metodo.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo rifiutaRichiestaDiLavoro è utile per rifiutare un’offerta di lavoro grazie ai parametri forniti dal metodo.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo lavoroNonAncoraEsistente è utile per selezionare tutti i lavori che soddisfano i parametri.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo aggiungiLavoro è utile per aggiungere un lavoro.

Immagine che contiene testo, screenshot, monitor, schermo

Descrizione generata automaticamente

#### File lavoratori\_Model.php

Questo file contiene la classe Lavoratori\_Model che estende la classe Model. E i metodi getRischiesteLavori, getLavoriNonOccupati, modificaRichiestaDiLavoro, getTitolo, lavoroNonAncoraEsistente, aggiungiLavoro, getLavoro e inviaEmailDiConferma.

* Il metodo getRischiesteLavori è utile per avere tutte le richieste ai lavori di un determinato utente.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo getLavoriNonOccupati è utile per avere tutti i lavori non occupati e non archiviati.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo modificaRichiestaDiLavoro è utile per permette di modificare una richiesta ad un lavoro grazie ai parametri passati.

Immagine che contiene testo, interni, monitor, screenshot

Descrizione generata automaticamente

* Il metodo getTitolo è utile per permette di ottenere tutti i lavori che hanno un determinato id specificato dal parametro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo lavoroNonAncoraEsistente è utile per permette di ottenere tutte le richieste ai lavori che soddisfano i parametri.

Immagine che contiene testo, monitor, schermo, screenshot

Descrizione generata automaticamente

* Il metodo aggiungiLavoro è utile per permette di aggiungere una richiesta ad un lavoro tramite ai parametri.

Immagine che contiene testo, screenshot, schermo, vicino

Descrizione generata automaticamente

* Il metodo getLavoro è utile per permette di ottenere un lavoro tramite un id passato come parametro.

Immagine che contiene testo

Descrizione generata automaticamente

* Il metodo inviaEmailDiConferma è utile per permette di inviare una e-mail di conferma all'utente quando accetta un lavoro.

Immagine che contiene testo

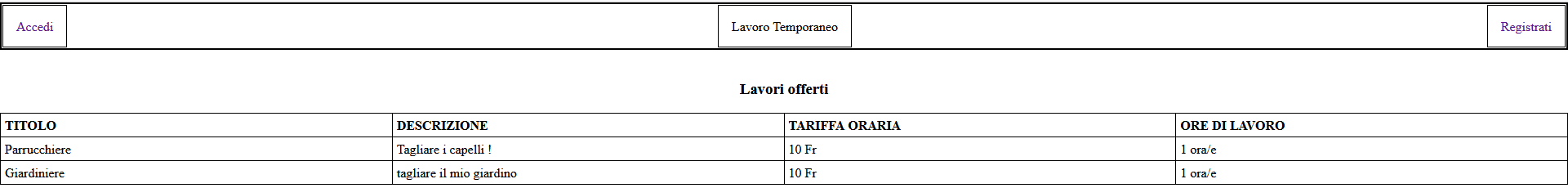
Descrizione generata automaticamente

### Cartella Views

Le Views sono i file contenente per lo più codice html che contengono quello che vedono gli utenti. Generalmente si ha una View per ogni pagina che l’utente visualizza. Di seguito sono descritte le interfacce.

#### Index

Come può far intuire il nome questa view rappresenta la pagina index appena si accede al sito senza ancora aver effettuato il login. Nell’immagine sottostante è riportata la resa grafica di tutto il codice all’interno del file.



#### Registrazione

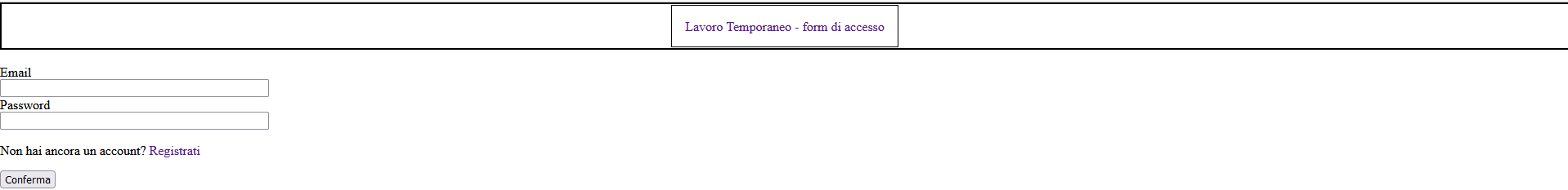
La view registrazione rappresenta la pagina contenente il form dove è possibile creare un nuovo utente tramite la sua e-mail, la sua password e il ruolo che vuole rappresentare ovvero datore di lavoro o lavoratore. Una volta effettuata la registrazione, verrà inviata una mail all’utente dal lavoro temporaneo.

Immagine che contiene testo

Descrizione generata automaticamente

#### Accesso

La view accesso rappresenta la pagina contenente il form dove è possibile accedere con un utente immettendo la sua e-mail, la sua password e il suo ruolo. Una volta effettuato l’accesso, si verrà reindirizzati alla pagina relativa al proprio ruolo, le possibilità sono tre: amministratori, datori di lavoro e lavoratori.



#### Pagina Amministratori

La view pagina amministratori rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out. Inoltre, questa prima pagina permette all’utente loggato come amministratore di vedere tutti i lavori offerti in quel momento, questa funzione è supportata da tre filtri il primo permette di mostrare tutti i lavori non ancora archiviati che sono occupati, il secondo permette di mostrare tutti i lavori non ancora archiviati che sono scaduti, mentre l’ultimo permette di vedere tutti i lavori anche quelli archiviati. I lavori non ancora archiviati si possono modificare tramite l’apposito bottone modifica (numero lavoro) e si possono creare le fatture con il primo bottone sotto l’intestazione, esso reindirizza alla pagina per aggiungere la fattura.

Immagine che contiene testo

Descrizione generata automaticamente

#### Pagina DatoriDiLavori

La view pagina datoriDiLavoro rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out. Inoltre, questa prima pagina permette all’utente loggato come datore di lavoro di vedere tutti i lavori offerti in quel momento, questa si può eliminare o modificare. Mentre sotto questa sezione c’è la sezione delle richieste ai lavori offerti, essi possono essere accettati o rifiutati. Si possono creare nuove offerte di lavoro con il primo bottone sotto l’intestazione, esso reindirizza alla pagina per aggiungere le offerte di lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

#### Pagina Lavoratori

La view pagina lavoratori rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out. Inoltre, questa prima pagina permette all’utente loggato come lavoratore di vedere tutte le richieste fatte ai lavori in quel momento, questa si può eliminare o modificare. Mentre sotto questa sezione c’è la sezione dei lavori offerti, ad essi si può fare una richiesta, facendo così si possono creare nuove richieste ai lavori offerte e per farlo si viene reindirizzati alla pagina per aggiungere le richieste per un lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

#### Pagina aggiungiFattura

La view pagina aggiungi fattura rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out. Inoltre, questa prima pagina permette all’utente loggato come amministratore di vedere tutti i lavori che sono finiti e quindi sono disponibili per essere fatturati, tramite il bottone calcola fattura si può calcolare la fattura del lavoro corrispondente. Mentre sotto questa sezione c’è la sezione delle fatture qui si possono visualizzare tutte le fatture che ha fatto l’applicativo. Infine, con il bottone Home sotto l’intestazione si può tornare nella pagina principale degli amministratori.

Immagine che contiene testo

Descrizione generata automaticamente

#### Pagina aggiungiOffertaDiLavoro

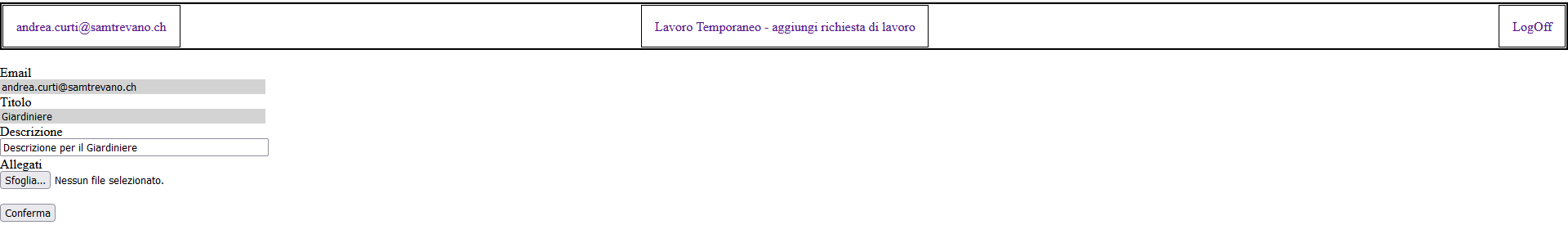
La view pagina aggiungi offerta di lavoro rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out e un link che permette di tornare nella pagina principale dei datori di lavoro. In questa pagina è contenuto il form per la creazione di un nuovo lavoro, per farlo sono necessari un titolo, una descrizione, una tariffa oraria e le ore di lavoro, cliccando sul bottone verrà inserito un nuovo lavoro nel db e si verrà reindirizzati nella home per i datori di lavoro.

Immagine che contiene testo

Descrizione generata automaticamente

#### Pagina aggiungiRichiestaDiLavoro

La view pagina aggiungi richiesta ad un lavoro rappresenta la pagina contenente una intestazione con la propria mail e la possibilità di effettuare il log out e un link che permette di tornare nella pagina principale dei lavoratori. In questa pagina è contenuto il form per la creazione di un’offerta ad un lavoro, per farlo sono necessari una descrizione e un allegato cliccando sul bottone verrà inserito una nuova richiesta ad un lavoro nel db e si verrà reindirizzati nella home per i datori di lavoro.



# Test

## Protocollo di test

|  |  |
| --- | --- |
| **Prerequisiti base:** | 1. Essere al punto 1.5 del Gantt 2. Avere XAMPP 3. Avere il database 4. Avere le pagine web |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-001  REQ-001 | **Nome:** | Verifica del sito web senza login |
| **Descrizione:** | È necessario verificare che il sito web funzioni correttamente e senza essere loggati visualizzare i lavori disponibili. | | |
| **Prerequisiti:** | 1. Prerequisiti base | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che si possono vedere tutte le offerte di lavoro. | | |
| **Risultati attesi:** | 1. Vedere tutte le offerte di lavoro senza essere loggati. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-0020 | **Nome:** | Verifica del sito web con login |
| **Descrizione:** | È necessario verificare che il sito web funzioni correttamente essendo loggati. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale e le pagine per l’amministratore, il datore di lavoro ed il cliente | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che si possa effettuare il login nel ruolo di amministratore, datore di lavoro o cliente. | | |
| **Risultati attesi:** | 1. Vedere in base al ruolo la rispettiva pagina web. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-0021 | **Nome:** | Verifica del funzionamento di: form di registrazione |
| **Descrizione:** | È necessario verificare che il sito web permetta ad un nuovo possibile utente di registrarsi. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di registrazione | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone registrati si apra una pagina con il form di registrazione 5. Controllare che la registrazione sia avvenuta correttamente con e-mail | | |
| **Risultati attesi:** | 1. Possibilità di fare la registrazione e conferma tramite e-mail. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-002  REQ-0022 | **Nome:** | Verifica del funzionamento di: form di accesso |
| **Descrizione:** | È necessario verificare che il sito web permetta ad un utente di accedere. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. | | |
| **Risultati attesi:** | 1. Riuscire ad entrare nella pagina corretta. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-003 | **Nome:** | Verifica risultato del login |
| **Descrizione:** | È necessario verificare che dopo il login l’utente possa accedere alla sua corretta sezione. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta | | |
| **Risultati attesi:** | 1. Riuscire ad entrare nella pagina corretta. 2. Vedi per REQ-0031, REQ-0032, REQ-0033. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-0031 | **Nome:** | Si possano creare richieste. (se si è lavoratori) |
| **Descrizione:** | È necessario verificare che dopo il login l’utente con il ruolo di cliente possa accettare le offerte. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta 7. Possibilità di accettare le offerte di lavoro | | |
| **Risultati attesi:** | 1. Riuscire a fare una richiesta per un lavoro. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-0032 | **Nome:** | Si possano creare offerte. (se si è datori di lavoro) |
| **Descrizione:** | È necessario verificare che dopo il login l’utente con il ruolo di datore di lavoro possa creare le offerte. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta 7. Possibilità di creare le offerte di lavoro | | |
| **Risultati attesi:** | 1. Riuscire a creare le offerte di lavoro. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-003  REQ-0033 | **Nome:** | Si possano gestire le offerte. (se si è amministratori) |
| **Descrizione:** | È necessario verificare che dopo il login l’utente con il ruolo di amministratore possa gestire le offerte di lavoro. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta 7. Possibilità di gestire le offerte di lavoro | | |
| **Risultati attesi:** | 1. Riuscire a gestire le offerte di lavoro. | | |
| **glossario** | 1. Gestire: possibilità di modificare. | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-0041 | **Nome:** | Verifica fine del lavoro. |
| **Descrizione:** | È necessario verificare che dopo il login l’utente con il ruolo di amministratore possa gestire le offerte di lavoro. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta 7. Possibilità di gestire le offerte di lavoro 8. Possibilità di verificare che un lavoro sia finito | | |
| **Risultati attesi:** | 1. Vedere i lavori scaduti occupati oppure tutti i lavori | | |

|  |  |  |  |
| --- | --- | --- | --- |
| **Test Case:**  **Riferimento**: | TC-004  REQ-0042 | **Nome:** | Calcolo spese del datore di lavoro. |
| **Descrizione:** | È necessario verificare che dopo il login l’utente con il ruolo di amministratore possa gestire le spese per il datore di lavoro. | | |
| **Prerequisiti:** | 1. Prerequisiti base 2. Avere la pagina web principale 3. Aver creato il form di accesso 4. Aver creato la pagina per il cliente 5. Aver creato la pagina per il datore di lavoro 6. Aver creato la pagina per gli amministratori | | |
| **Procedura:** | 1. Accendere tutti i servizi XAMPP (…) 2. Recarsi sulla pagina web principale 3. Controllare di non essere loggati 4. Verificare che cliccando sul bottone accedi si apra una pagina con il form di accesso 5. Controllare che l’accesso sia avvenuto correttamente. 6. Avere accesso alla pagina corretta 7. Possibilità di calcolare le spese per il datore di lavoro | | |
| **Risultati attesi:** | 1. Riuscire a calcolare le spese di lavoro per i datori e i collaboratori | | |

## Risultati test

|  |  |  |  |
| --- | --- | --- | --- |
| Test Case | Funzionamento | Commento | Data |
| TC-001 | OK | Quando accedo al sito web senza effettuare il login vedo tutti i lavori proposti dal sito web. | 09.12.2021 |
| TC-002  REQ-0020 | OK | Quando effettuo il login al sito web, a dipendenza del ruolo al quale appartengo vedrò la pagina corrispettiva.  Nel caso dell’amministratore vedrò:    Nel caso del datore vedrò:    Nel caso del lavoratore vedrò: | 09.12.2021 |
| TC-002  REQ-0021 | OK | Quando si registra una nuova persona, le viene inviata una mail che conferma di aver creato il suo profilo per il sito del lavoro temporaneo. | 09.12.2021 |
| TC-002  REQ-0022 | OK | Quando accede una persona, le viene la rispettiva pagina per il sito del lavoro temporaneo, vedi risultato test TC-002 REQ-0020. | 09.12.2021 |
| TC-003  REQ-0031 | OK | Quando si accede con il ruolo lavoratore, si può creare una richiesta per un determinato lavoro. | 09.12.2021 |
| TC-003  REQ-0032 | OK | Quando si accede con il ruolo datore di lavoro, si può creare un’offerta di lavoro. | 09.12.2021 |
| TC-003  REQ-0033 | OK | Quando si accede con il ruolo di amministratore, si possono gestire tutti i lavori (modificarli) se non già archiviati, in quel caso sarà possibile visualizzarli ma non modificarli. | 09.12.2021 |
| TC-004  REQ-0041 | OK | Quando si accede con il ruolo di amministratore, si possono vedere tutti i lavori scaduti occupati oppure tutti i lavori:  Nel caso dei lavori occupati vedrò:    Nel caso dei lavori scaduti vedrò:    Nel caso di tutti i lavori vedrò: | 09.12.2021 |
| TC-004  REQ-0042 | OK | Quando si clicca su crea fattura, potremo vedere tutti i lavori che bisogna fatturare. Cliccando successivamente sul bottone calcola fatturazione, verrà creata la fattura per il datore e per l’amministratore. | 09.12.2021 |

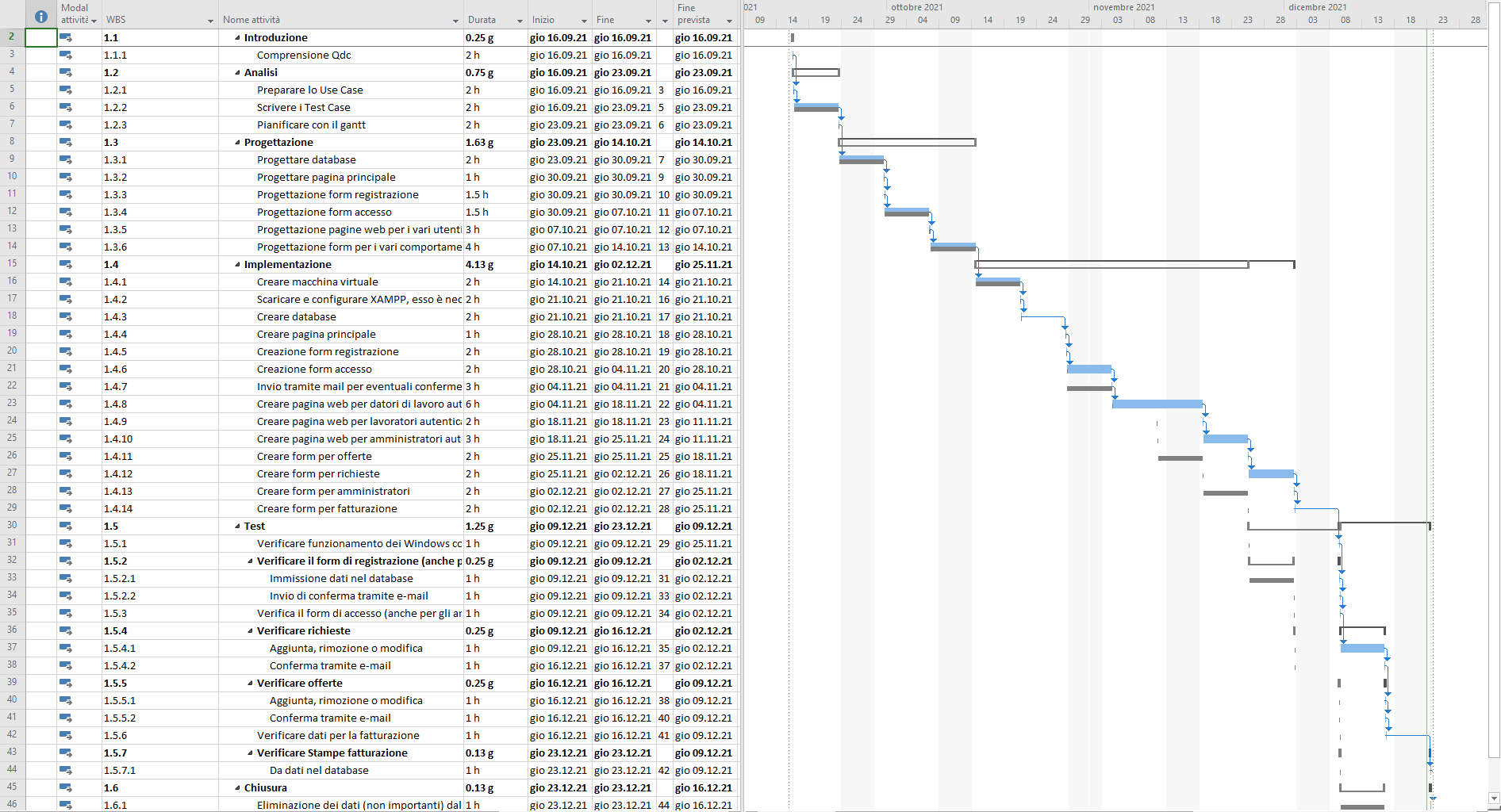
## Mancanze/limitazioni conosciute

A livello estetico manca un po' di parte grafica che renderebbe il sito più bello, semplice e intuitivo.

A livello funzionale manca un sistema di recupero password e un sistema per cambiare la password. Successivamente si potrebbe mettere questo servizio online e quindi mancherebbe uno spazio fisico dove salvare l’applicativo e farlo raggiungere da chiunque tramite l’ausilio di un browser con accesso ad Internet e con un determinato dominio. Infine, manca una vera e propria fatturazione con delle vere transazioni bancarie.

# Consuntivo

Rispetto alla pianificazione preventiva, nel Gantt consuntivo sono cambiati diversi tempi di diverse attività, questo era dovuto dal fatto che sono stati calcolati male alcuni tempi e non sono state prese in considerazione ore di margine per eventuali problemi riscontati durante l’implementazione. Nonostante ciò, sono comunque riuscito a stare nei tempi e a finire il progetto anche se alcune accortezze le ho dovute fare a casa per poter finire.



# Conclusioni

## Sviluppi futuri

Come detto nel punto 5.3, uno sviluppo futuro potrebbe essere quello di implementare una grafica più bella e user friendly, inoltre sarebbe auspicabile implementare un sistema di recupero password e un sistema per cambiare la password.

## Considerazioni personali

In conclusione, posso dire che il progetto mi ha insegnato moltissimo, sia a livello personale che a livello lavorativo. Grazie ad esso sono riuscito a lavorare in modo estremamente ordinato, sia lato di codice che lato dei diari. Infatti, durante questo progetto a mio parere ho tenuto molta cura nei diari e in quello che andavo a svolgere durante ogni giornata di lavoro e ho migliorato di molto le mie competenze in php e nella gestione di un progetto. Il risultato finale del progetto mi ha abbastanza soddisfatto, anche perché inizialmente non ero minimamente in grado di farlo e alla fine ho comunque raggiunto l’obbiettivo. Grazie a questo ho aumentato la mia autostima e la mia voglia di mettermi in gioco in qualcosa che non conosco bene. Come ultimo ma non meno importante a livello di tempistiche posso dire che sono parzialmente soddisfatto del progetto siccome non sono riuscito a gestire al meglio tutto il tempo, a mia disposizione infatti mi sono ritrovato a concludere alcune cosette a casa per riuscire a finire tutto. Per il prossimo progetto dovrò calcolare del tempo in più per il progetto, fare parallelamente la documentazione e sicuramente aumentando le mie conoscenze impiegherò sempre meno tempo nel finire progetti che richiedono queste competenze.

# Bibliografia

## Sitografia

<https://www.php.net/manual/en/index.php> , *Sito di Php*, ho consultato questo sito durante tutto il progetto

<https://www.w3schools.com/php/default.asp> , *Sito W3schools per php*, ho consultato questo sito durante tutto il progetto

<https://www.youtube.com/watch?v=IS74TMFIWBs> , *Video How to send emails from localhost using PHP and Sendmail | Send email with PHP,* 04.11.2021

<https://stackoverflow.com/questions/2350937/php-fatal-error-using-this-when-not-in-object-context> , *PHP Fatal Error,* 11.11.2021

<https://stackoverflow.com/questions/25174183/warning-mysqli-connect-hy000-1045-access-denied-for-user-usernameloca> , *Warning,* 21.10.2021

<https://stackoverflow.com/questions/21659691/error-1452-cannot-add-or-update-a-child-row-a-foreign-key-constraint-fails> , *Error*, 21.10.2021

# Allegati

Elenco degli allegati, esempio:

* Mandato e/o QdC
* Prodotto
* Diari di lavoro
* Database
* Gantt preventivo
* Gantt consultivo