

Progetto PizzaDelivery

Abstract

Candidato: Jari Näser
Azienda: Scuola Arti e Mestieri di Trevano
Periodo: 3.09.2019 - 20.12.2019
Presentazione: 7.01.2020 - 17.01.2020

Situazione iniziale

È stato richiesto lo sviluppo di un software che si occupi di facilitare ed automatizzare il processo di comande online per pizzerie che offrono il servizio di consegne d'asporto.

Il prodotto dev'essere in grado di mantenere lo stato delle ordinazioni, fattorini e consegne del ristorante attraverso una banca dati. È inoltre richiesta una definizione di vari ruoli in modo da poter creare una gerarchia di permessi e di limitare l'accesso a determinate pagine all'interno dell'applicativo web.

Tutti gli impiegati possiedono un account e a dipendenza del loro ruolo possono accedere a pagine aggiuntive rispetto ai normali clienti.

Il programma dispone anche di una pagina di gestione accessibile solamente dagli amministratori. Attraverso questa pagina si può aggiungere, modificare, disabilitare ed eliminare utenti ed articoli.

Infine dispone anche di una pagina principale dalla quale è possibile comporre il proprio ordine attraverso una lista molto semplice da utilizzare.

Attuazione

Lo sviluppo di questo progetto è basato sulla struttura MVC ed è stato utilizzato il linguaggio PHP, la creazione delle varie pagine è stata principalmente effettuata con HTML e sono state utilizzate librerie esterne come Bootstrap e FontAwesome che si occupano di gestire l'aspetto grafico delle varie interfacce rendendole dinamiche su qualsiasi tipologia di dispositivo. L'applicativo è stato sviluppato principalmente per l'utilizzo da smartphone. Per quanto riguarda l'implementazione delle mappe per permettere agli impiegati vendita di sapere la posizione dei clienti e dei fattori è stato utilizzato il servizio MapBox che mette a disposizione gratuitamente mappe molto precise, interessanti graficamente e complete per quanto riguarda le funzionalità.

Risultati

Quasi tutti i requisiti iniziali sono stati soddisfatti avendo ottenuto un'applicazione robusta e funzionante.

Le varie pagine comunicano correttamente fra di esse e l'integrità dei dati rimane sempre coerente con qualsiasi tipologia di azione attraverso i vari formulari presenti nel programma, questo grazie all'implementazione di numerosi controlli.

Sono molto soddisfatto del risultato ottenuto malgrado abbia spesso riscontrato problemi i quali sono però sempre stati risolti grazie ad accurate riflessioni e alla metodologia di sviluppo.

Documentazione

Documentazione progetto PizzaDelivery

1 Introduzione.....	4
1.1 Informazioni sul progetto.....	4
1.2 Abstract.....	4
1.3 Scopo.....	4
1.4 Analisi del dominio.....	4
1.5 Analisi e specifica dei requisiti.....	5
1.6 Use case.....	7
1.7 Analisi dei mezzi.....	8
1.7.1 Software.....	9
1.7.2 Hardware.....	9
2 Progettazione.....	10
2.1 Design dell'architettura del sistema.....	10
2.2 Design dei dati e database.....	11
2.3 Design delle interfacce.....	12
2.3.1 Pagina Principale.....	12
2.3.2 Pagina Login.....	13
2.3.3 Pagina Amministratore.....	13
2.3.4 Pagina Ordinazione.....	14
2.3.5 Pagina Fattorini.....	14
2.3.6 Pagina Ordinazioni.....	15
2.3.7 Pagina Consegne.....	15
3 Implementazione.....	16
3.1 Struttura dell'applicativo.....	16
3.1.1 Routing URL.....	16
3.1.2 Gerarchia delle cartelle.....	17
3.1.3 File di configurazione.....	18
3.1.4 Classi Controller.....	19
3.1.5 Cartella libs.....	20
3.1.6 Classi Model.....	21
3.1.7 Scripts.....	22
3.1.8 Views.....	23
3.1.9 Classe Database.....	23
3.2 Database.....	24
3.3 Sicurezza e ruoli dell'applicativo.....	25
3.4 Codice JavaScript lato client.....	26
3.5 Pagina Gestione Pizzeria.....	28
3.6 Implementazione Mappe.....	30
3.7 Approccio Mobile First.....	31
3.8 Installazione debugger.....	32
3.8.1 Download.....	32
3.8.2 Configurazione php.ini.....	32
3.8.3 Configurazione phpStorm.....	33
4 Test.....	34
4.1 Protocollo di test.....	34
4.2 Risultati test.....	38
4.3 Mancanze/limitazioni conosciute.....	38
5 Installazione.....	39
5.1 Requisiti iniziali.....	39
5.2 Copiatura della cartella PizzaDelivery sulla propria macchina.....	39
5.3 Creazione DataBase e inserimento dei dati essenziali.....	39
5.4 Impostazione subfolder della cartella webroot.....	40
5.5 Modifica percorso URL del progetto.....	40
5.6 Configurazione accesso MySQL per interagire con il DataBase.....	41
5.7 Primo accesso tramite interfaccia grafica.....	41
Consuntivo.....	42
6 Conclusioni.....	43

6.1 Sviluppi futuri.....	43
6.2 Considerazioni personali.....	43
7 Bibliografia.....	44
7.1 Sitografia.....	44
8 Allegati.....	44

1 Introduzione

1.1 Informazioni sul progetto

Allievo coinvolto: Jari Näser

Classe: Informatica 4AA Presso la Scuola Arti e Mestieri di Trevano

Docenti responsabili: Ivan Raimondi

Data inizio: 3-09-2019

Data fine: 20-12-2019

1.2 Abstract

With today's world wide food demand being higher than ever and people's laziness as well the idea of this project is to unify these two requests creating a service that brings one of the world's most requested dishes to people's home.

With PizzaDelivery it's now possible for Pizza Stores to offer a really powerful service to its customers by bringing them their ordered food wherever they like in a quick and cheap way.

This software is capable to manage in a really simple and yet efficient way the serving task and all the challenges that come with it, it's possible to manage all the incoming and outgoing orders, all of your employees such as delivery men, their status and position.

1.3 Scopo

Lo scopo di questo progetto è di realizzare un sistema informatico che semplifichi la gestione delle ordinazioni online accessibile a chiunque, in questo caso per le pizzerie la consegna a domicilio.

Attraverso una piattaforma dinamica e programmata mantenendo un approccio di tipo mobile first è possibile eseguire comande di qualsiasi tipo attraverso l'applicativo Web.

Il programma offre anche svariate pagine per la gestione di tutta l'infrastruttura interna della pizzeria stessa accessibile agli impiegati come le ordinazioni, consegne, i fattorini ed una pagina amministrativa per quanto riguarda la gestione degli utenti ed articoli registrati nel sistema.

1.4 Analisi del dominio

Attualmente non esistono ancora programmi con le funzionalità richieste dal cliente.

Lo scopo è quello di offrire un servizio a basso costo e molto semplice da usare per varie pizzerie che vorrebbero entrare nel mercato delle consegne a domicilio per espandersi in modo molto rapido ed efficiente, cosa che attualmente non è ancora possibile.

1.5 Analisi e specifica dei requisiti

Il committente richiede un sistema che sia capace di gestire il servizio online di consegne a domicilio di una pizzeria attraverso un portale web.

Esso è principalmente composto dalle 5 pagine: principale, ordinazioni, consegne, fattorini e gestione pizzeria che interagiscono continuamente con il database PizzaDelivery.

Infine è possibile utilizzare il programma come cliente normale che esegue degli ordini oppure come impiegato, fattorino oppure amministratore facendo il login con il proprio account potendo accedere a funzionalità e pagine aggiuntive rispetto ai normali clienti.

ID: REQ-01	
Nome	Installazione ambiente di sviluppo
Priorità	1
Versione	1.0
Note	Si necessita di una licenza JetBrains
Sotto requisiti	
001	Si necessita di scaricare ed installare ...AMP (XAMP, MAMP o LAMP a dipendenza del sistema operativo)
002	Si necessita di scaricare ed installare l'IDE PHPStorm della JetBrains
003	Si necessita di scaricare ed installare MySqlWorkbench per la gestione semplificata del database.

ID: REQ-02	
Nome	Progettazione database
Priorità	1
Versione	1.0
Sotto requisiti	
001	Si necessita di essere in chiaro sui vari ruoli dell'applicazione

ID: REQ-03	
Nome	Creazione database
Priorità	1
Versione	1.0
Note	Dipende dal REQ-02 (Progettazione database)
Sotto requisiti	
001	È necessario un diagramma ER da seguire

ID: REQ-04

Nome	Implementazione modello MVC
Priorità	1
Versione	1.0
Sotto requisiti	
001	Bisogna impostare la root del progetto corretta nelle sue configurazioni per evitare errori nel routing

ID: REQ-05

Nome	Creazione pagina ordinazioni
Priorità	1
Versione	1.0
Note	Dipende dal REQ-04 (implementazione modello MVC)
Sotto requisiti	
001	Si necessita di una tabella nel database che memorizzi tutte le ordinazioni
002	Possibilità di ricavare dati attraverso query dal database
003	Si deve gestire il login dei vari utenti

ID: REQ-06

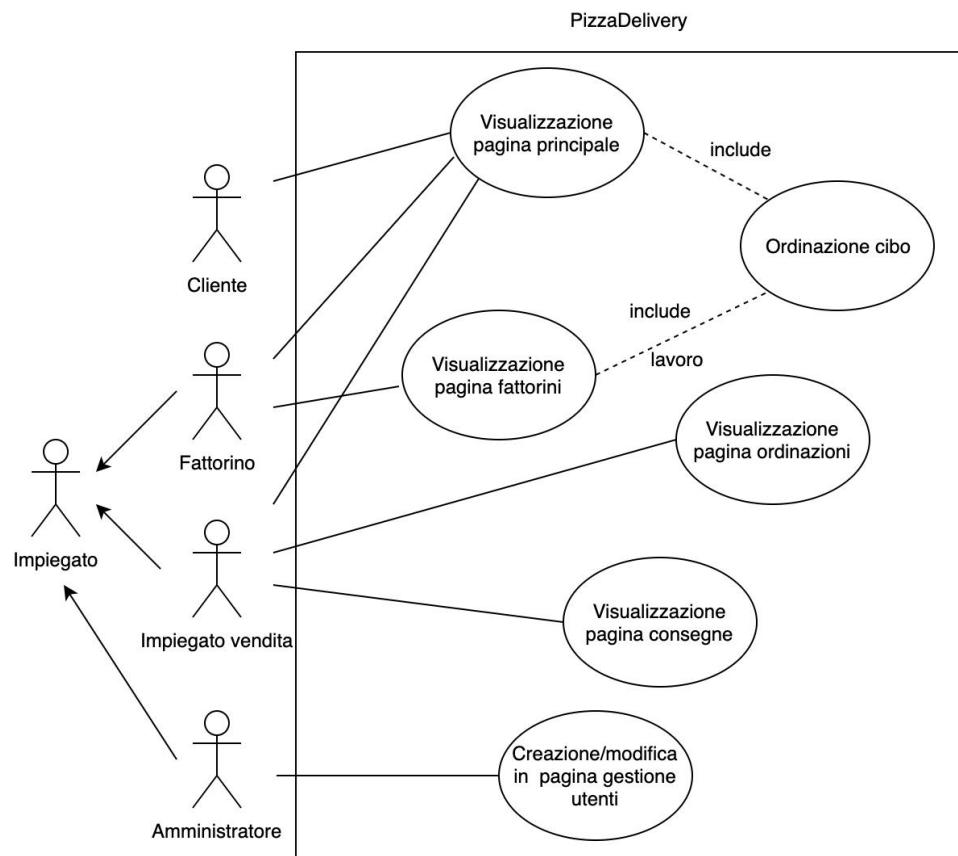
Nome	Creazione pagina consegne
Priorità	1
Versione	1.0
Note	Dipende dal REQ-04 (implementazione modello MVC)
Sotto requisiti	
001	Si necessita di una tabella nel database che memorizzi tutte le consegne
002	Possibilità di ricavare le consegne precedenti e attuali attraverso query dal database
003	Si deve gestire il login dei vari utenti

ID: REQ-07

Nome	Creazione pagina fattorini
Priorità	1
Versione	1.0
Note	Dipende dal REQ-04 (implementazione modello MVC)
Sotto requisiti	
001	Si necessita di una tabella nel database che memorizzi tutti i fattorini e le loro azioni
002	Possibilità di ricavare i dati attraverso query dal database dei fattorini e delle loro azioni giornaliere
003	Si deve gestire il login dei vari utenti

1.6 Use case

Le iterazioni tra i vari tipi di utente ed il programma avvengono nel seguente modo:

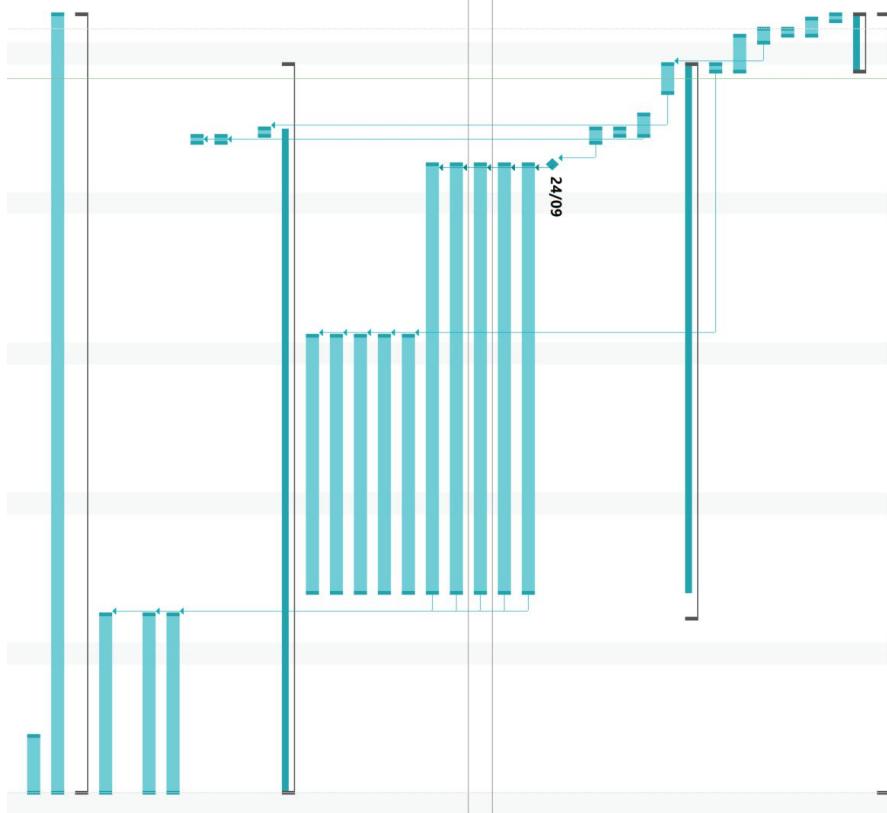


Progetto PizzaDelivery

1.7 Analisi dei mezzi

Per quanto riguarda la suddivisione dei tempi e quindi la progettazione delle tempistiche di questo progetto ho pensato di gestirle nel seguente modo per sfruttare queste 174 ore disponibili nel miglior dei modi possibili.

«Pizza Delivery»		36 days	Tue 03/09/19 Fri 20/12/19
Analisi	3 days	Tue 03/09/19 Tue 10/09/19	
Analisi dei requisiti	0,75 days	Tue 03/09/19 Tue 03/09/19	
Analisi del dominio	1 day	Tue 03/09/19 Thu 05/09/19	
Analisi DataBase	0,75 days	Thu 05/09/19 Thu 05/09/19	
Progettazione DataBase	1,5 days	Thu 05/09/19 Fri 06/09/19	
Analisi struttura progetti	1,5 days	Fri 06/09/19 Tue 10/09/19	
Schizzo delle interfacce	0,75 days	Tue 10/09/19 Tue 10/09/19	
«Implementazione»		25 days	Tue 10/09/19 Tue 26/11/19
Creazione DataBase	2,25 days	Tue 10/09/19 Fri 13/09/19	6
Implementazione modello MVC base	1,5 days	Tue 17/09/19 Thu 19/09/19	
Creazione pagine principali	0,75 days	Thu 19/09/19 Thu 19/09/19	
Creazione modulo per connessione al DataBase	1,5 days	Thu 19/09/19 Fri 20/09/19	
Connessione al DataBase	0 days	Tue 24/09/19 Tue 24/09/19	13
Sviluppo pagina principale	2,25 days	Tue 24/09/19 Fri 22/11/19	14
Sviluppo pagina ordinazioni	2,25 days	Tue 24/09/19 Fri 22/11/19	14
Sviluppo pagina fattorini	2,25 days	Tue 24/09/19 Fri 22/11/19	14
Sviluppo pagina consegne	2,25 days	Tue 24/09/19 Fri 22/11/19	14
Sviluppo pagina clienti	2,25 days	Tue 24/09/19 Fri 22/11/19	14
Creazione grafica pagina principale	12 days	Fri 18/10/19 Fri 22/11/19	8
Creazione grafica pagina ordinazioni	12 days	Fri 18/10/19 Fri 22/11/19	8
Creazione grafica pagina fattorini	12 days	Fri 18/10/19 Fri 22/11/19	8
Creazione grafica pagina consegne	12 days	Fri 18/10/19 Fri 22/11/19	8
Creazione grafica pagina clienti	12 days	Fri 18/10/19 Fri 22/11/19	8
«Test»		33,75 days	Tue 10/09/19 Fri 20/12/19
Test funzionamento corretto DataBase	0,75 days	Thu 19/09/19 Thu 19/09/19	10
Test struttura base MVC	0,75 days	Fri 20/09/19 Fri 20/09/19	11
Test routing corretto MVC	0,75 days	Fri 20/09/19 Fri 20/09/19	11
Test interface	9 days	Tue 26/11/19 Fri 20/12/19	16:15:19
Funzionamento corretto di elementi come bottoni, link e mappe eccetera	9 days	Tue 26/11/19 Fri 20/12/19	16:15:19
Test funzionamento azioni backend	9 days	Tue 26/11/19 Fri 20/12/19	16:15:19
«Documentazione»		36 days	Tue 03/09/19 Fri 20/12/19
Documentazione Word	36 days	Tue 03/09/19 Fri 20/12/19	
Sviluppo presentazione	3 days	Fri 13/12/19 Fri 20/12/19	



1.7.1 Software

Per la realizzazione di questo progetto ho utilizzato i seguenti Software:

- PHPStorm 2017.2.4: IDE principale per la scrittura di tutto il programma.
- MAMP 4.5: Programma che funge da WebServer offrendo i servizi di Apache, MySQL e PHP.
- Google Chrome 79.0.3945.79: Browser utilizzato per testare lo sviluppo dell'applicativo.
- MySQLWorkBench 8.0.17: Software usato per lo sviluppo del codice SQL per il DataBase.
- Sublime Text 3.2.1: Editore di testo per aprire vari file ed eseguire piccole modifiche.
- GitHub Desktop 2.1.3: Software per accedere alla repository di GitHub tramite interfaccia grafica.
- Draw.io 11.2.8: Software usato per quasi tutti i diagrammi e i design delle interfacce.
- Project 2016: Programma utilizzato per effettuare il diagramma di Gantt.
- Pages 8.2: Software per scrivere i diari.
- WPS Office 1.5.0: Software per scrivere i diari e la documentazione.

1.7.2 Hardware

Questo progetto è stato sviluppato unicamente sul mio portatile personale.

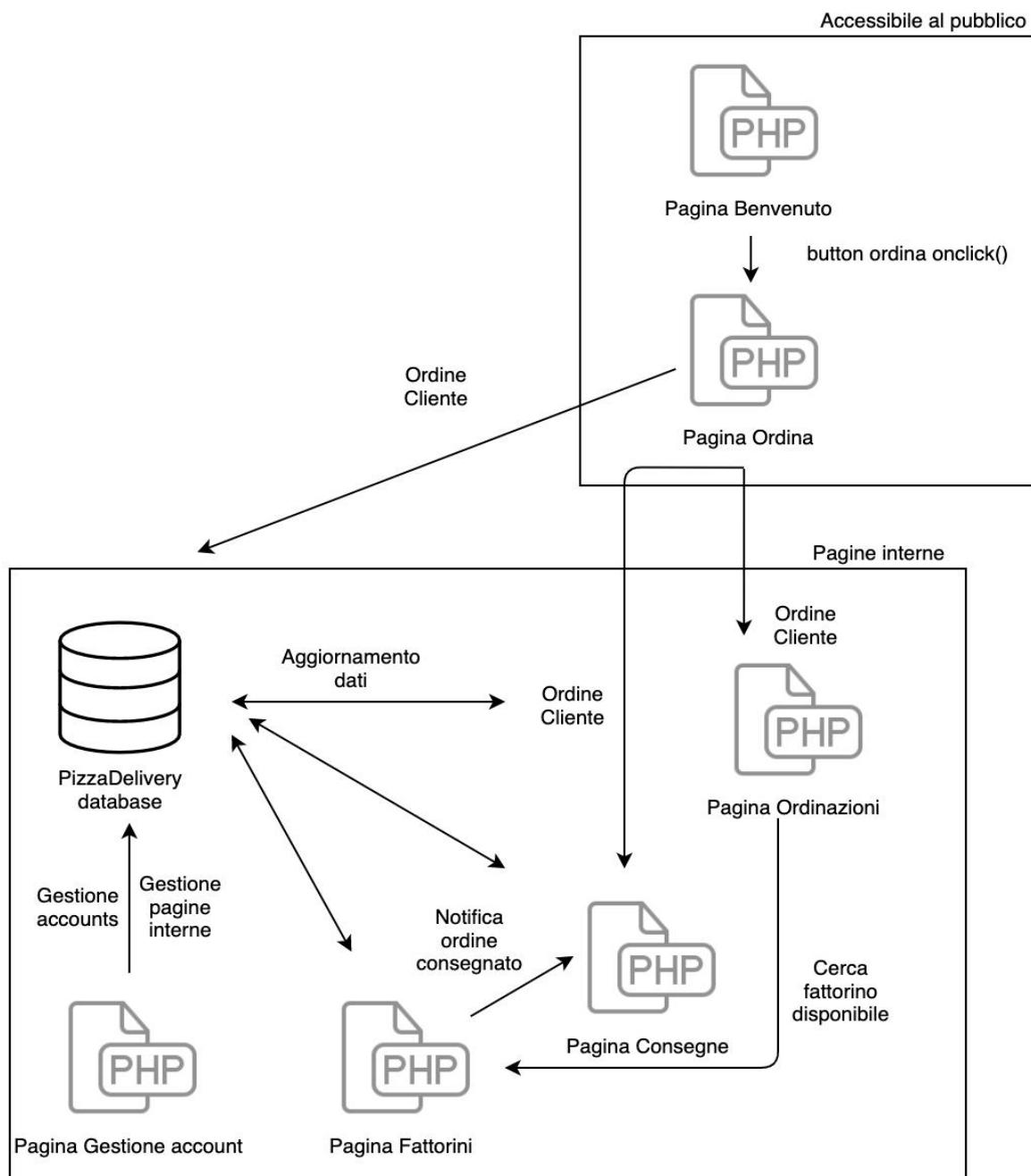
Modello: Apple Mac Book Pro mid 2015

OS: Catalina 10.15.1

2 Progettazione

2.1 Design dell'architettura del sistema

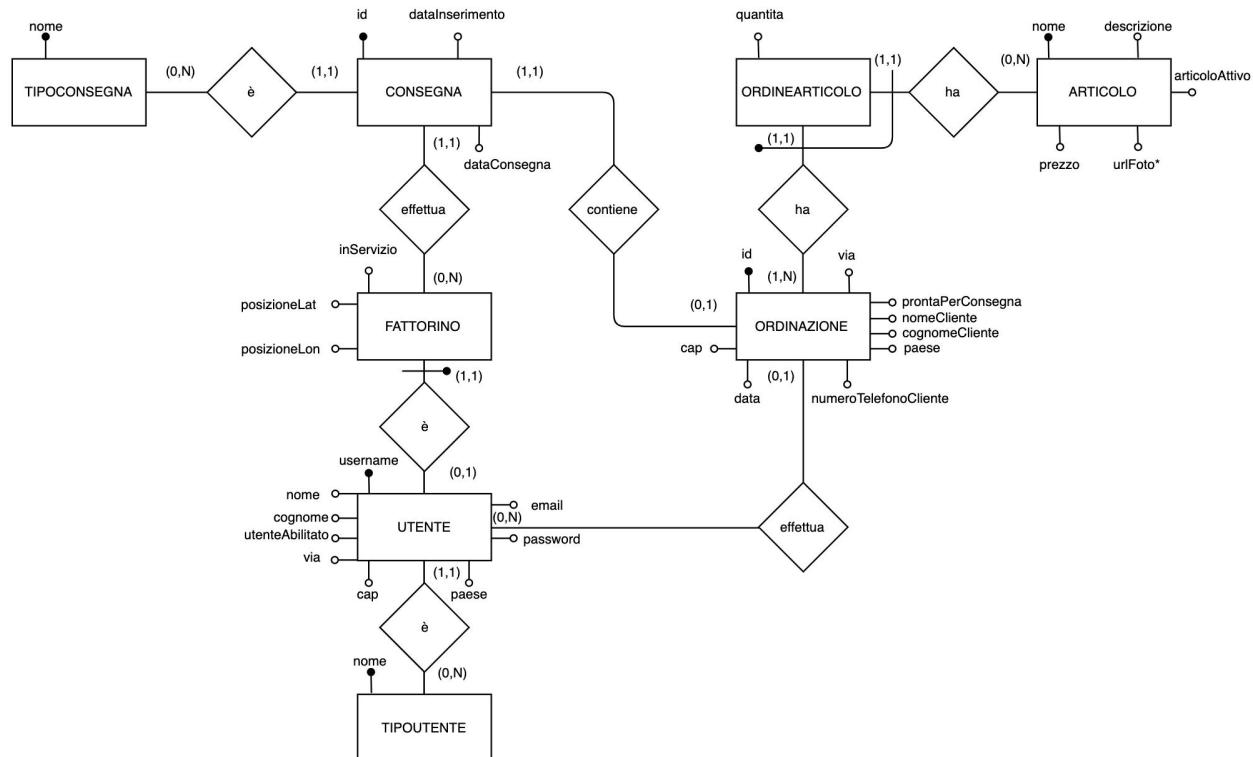
L'applicazione funziona principalmente nel seguente modo.



Progetto PizzaDelivery

2.2 Design dei dati e database

Il database che ha lo scopo di dare una base sul quale viene costruito tutto il programma è stato progettato nel seguente modo:

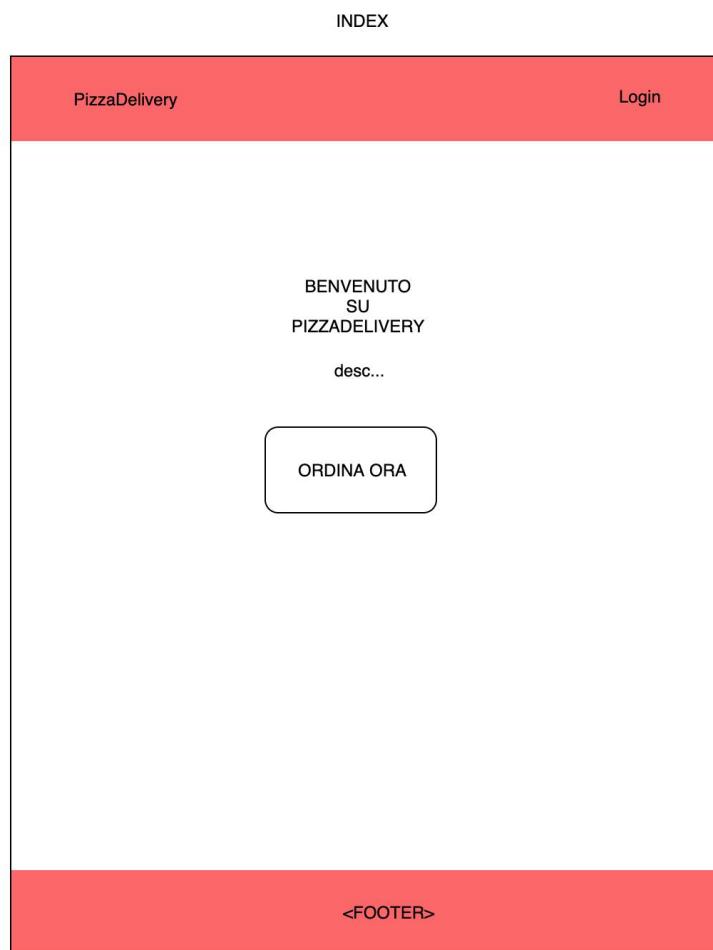


2.3 Design delle interfacce

Per le varie interfacce ho cercato di applicare lo stesso stile rendendole belle da vedere ma allo stesso tempo molto intuitive e semplici da usare.

2.3.1 Pagina Principale

È la prima pagina che appare all'apertura del sito sulla quale si può già ordinare oppure effettuare un login per gli impiegati della pizzeria.



2.3.2 Pagina Login

Pagina nel quale si può fare il login se si lavora nella pizzeria e si dispone di un account.

LOGIN

PizzaDelivery	Consegne Ordinazioni Fattorini Login
Login <input type="text" value="username"/> <input type="password" value="Password"/> <input type="button" value="Login"/> <small>Having trouble logging in? Click Here</small>	
<FOOTER>	

2.3.3 Pagina Amministratore

Pagina dalla quale l'amministratore del sistema può gestire, aggiungere, modificare e rimuovere gli account della pizzeria.

<p style="text-align: center;">ADMIN</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;">PizzaDelivery</td> <td style="width: 90%;">Login</td> </tr> <tr> <td colspan="2" style="text-align: center; padding: 10px;"> Gestione utenti <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Username</th> <th style="width: 30%;">tipologia</th> <th style="width: 40%;">Modifica</th> </tr> </thead> <tbody> <tr> <td>paolo.rossi</td> <td>fattorino</td> <td></td> </tr> <tr> <td colspan="3"><list item></td> </tr> <tr> <td colspan="3" style="height: 40px;"></td> </tr> </tbody> </table> <input type="button" value="crea utente"/> </td> </tr> <tr> <td colspan="2" style="text-align: center; padding: 10px;"><FOOTER></td> </tr> </table>	PizzaDelivery	Login	Gestione utenti <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Username</th> <th style="width: 30%;">tipologia</th> <th style="width: 40%;">Modifica</th> </tr> </thead> <tbody> <tr> <td>paolo.rossi</td> <td>fattorino</td> <td></td> </tr> <tr> <td colspan="3"><list item></td> </tr> <tr> <td colspan="3" style="height: 40px;"></td> </tr> </tbody> </table> <input type="button" value="crea utente"/>		Username	tipologia	Modifica	paolo.rossi	fattorino		<list item>						<FOOTER>		<p style="text-align: center;">ADMIN ON EDIT</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 10%;">PizzaDelivery</td> <td style="width: 90%;">Login</td> </tr> <tr> <td colspan="2" style="text-align: center; padding: 10px;"> <small>franco.rossi</small> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Nome</td> <td><input type="text"/></td> </tr> <tr> <td>cognome</td> <td><input type="text"/></td> </tr> <tr> <td>email</td> <td><input type="text"/></td> </tr> <tr> <td colspan="2" style="text-align: center;">...</td> </tr> </table> <input type="button" value="rispristina"/> <input type="button" value="elimina"/> <input type="button" value="salva"/> </td> </tr> <tr> <td colspan="2" style="text-align: center; padding: 10px;"><FOOTER></td> </tr> </table>	PizzaDelivery	Login	<small>franco.rossi</small> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Nome</td> <td><input type="text"/></td> </tr> <tr> <td>cognome</td> <td><input type="text"/></td> </tr> <tr> <td>email</td> <td><input type="text"/></td> </tr> <tr> <td colspan="2" style="text-align: center;">...</td> </tr> </table> <input type="button" value="rispristina"/> <input type="button" value="elimina"/> <input type="button" value="salva"/>		Nome	<input type="text"/>	cognome	<input type="text"/>	email	<input type="text"/>	...		<FOOTER>	
PizzaDelivery	Login																																
Gestione utenti <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr> <th style="width: 30%;">Username</th> <th style="width: 30%;">tipologia</th> <th style="width: 40%;">Modifica</th> </tr> </thead> <tbody> <tr> <td>paolo.rossi</td> <td>fattorino</td> <td></td> </tr> <tr> <td colspan="3"><list item></td> </tr> <tr> <td colspan="3" style="height: 40px;"></td> </tr> </tbody> </table> <input type="button" value="crea utente"/>		Username	tipologia	Modifica	paolo.rossi	fattorino		<list item>																									
Username	tipologia	Modifica																															
paolo.rossi	fattorino																																
<list item>																																	
<FOOTER>																																	
PizzaDelivery	Login																																
<small>franco.rossi</small> <table style="width: 100%; border-collapse: collapse;"> <tr> <td style="width: 30%;">Nome</td> <td><input type="text"/></td> </tr> <tr> <td>cognome</td> <td><input type="text"/></td> </tr> <tr> <td>email</td> <td><input type="text"/></td> </tr> <tr> <td colspan="2" style="text-align: center;">...</td> </tr> </table> <input type="button" value="rispristina"/> <input type="button" value="elimina"/> <input type="button" value="salva"/>		Nome	<input type="text"/>	cognome	<input type="text"/>	email	<input type="text"/>	...																									
Nome	<input type="text"/>																																
cognome	<input type="text"/>																																
email	<input type="text"/>																																
...																																	
<FOOTER>																																	

2.3.4 Pagina Ordinazione

Pagina sulla quale è possibile selezionare i prodotti da ordinare ed inserire i propri dati per la comanda.

ORDINA

PizzaDelivery	Login									
ORDINA <div style="display: flex; justify-content: space-between;"> <div style="width: 40%;"> <input style="width: 100%; height: 25px; margin-bottom: 5px;" type="text" value="Numero Telefono"/> <input style="width: 100%; height: 25px; margin-bottom: 5px;" type="text" value="Nome"/> <input style="width: 100%; height: 25px; margin-bottom: 5px;" type="text" value="Cognome"/> </div> <div style="width: 40%;"> <div style="border: 1px solid black; padding: 5px; width: 100%;">Google maps via picker</div> </div> </div> <div style="border: 1px solid black; padding: 10px; margin-top: 10px;"> <p style="text-align: center;">Select dei prodotti</p> </div>										
<table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">Nome</th> <th style="padding: 5px;">Pz.</th> <th style="padding: 5px;">Costo</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">Pizza margherita</td> <td style="padding: 5px; text-align: center;">2</td> <td style="padding: 5px;">21.90.-</td> </tr> <tr> <td colspan="3" style="padding: 5px; text-align: center;"><list item></td> </tr> </tbody> </table>		Nome	Pz.	Costo	Pizza margherita	2	21.90.-	<list item>		
Nome	Pz.	Costo								
Pizza margherita	2	21.90.-								
<list item>										
<input style="border: 1px solid black; border-radius: 10px; padding: 5px 20px; background-color: white; color: black; font-weight: bold; width: fit-content; margin: auto;" type="button"/>										
<FOOTER>										

2.3.5 Pagina Fattorini

Pagina dalla quale si possono visualizzare tutti i fattorini dell'azienda e le loro informazioni più importanti.

<p style="text-align: center;">FATTORINI</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="background-color: #f08080; color: white; padding: 5px;">PizzaDelivery</td> <td style="padding: 5px;">Consegne</td> <td style="padding: 5px;">Ordinazioni</td> <td style="padding: 5px;">Fattorini</td> <td style="padding: 5px;"><Clienti></td> </tr> <tr> <td colspan="5" style="padding: 10px;"> <p style="text-align: center;">FATTORINI</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID</th> <th style="padding: 5px;">Nome</th> <th style="padding: 5px;">Consegne effettuate oggi</th> <th style="padding: 5px;">Stato</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">3</td> <td style="padding: 5px;">Paolo Neri</td> <td style="padding: 5px; text-align: center;">5</td> <td style="padding: 5px; background-color: #ff0000; color: white;">In servizio</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table> </td> </tr> </table>	PizzaDelivery	Consegne	Ordinazioni	Fattorini	<Clienti>	<p style="text-align: center;">FATTORINI</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID</th> <th style="padding: 5px;">Nome</th> <th style="padding: 5px;">Consegne effettuate oggi</th> <th style="padding: 5px;">Stato</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">3</td> <td style="padding: 5px;">Paolo Neri</td> <td style="padding: 5px; text-align: center;">5</td> <td style="padding: 5px; background-color: #ff0000; color: white;">In servizio</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table>					ID	Nome	Consegne effettuate oggi	Stato	3	Paolo Neri	5	In servizio	<list item>				<list item>				<list item>				...				<p style="text-align: center;">FATTORINI ONCLICK</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <tr> <td style="background-color: #f08080; color: white; padding: 5px;">PizzaDelivery</td> <td style="padding: 5px;">Consegne</td> <td style="padding: 5px;">Ordinazioni</td> <td style="padding: 5px;">Fattorini</td> <td style="padding: 5px;"><Clienti></td> </tr> <tr> <td colspan="5" style="padding: 10px;"> <p style="text-align: center;">FATTORINO #ID</p> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Nome: Paolo</p> <p>Cognome: Neri</p> <p>Stato: Libero</p> </div> <div style="width: 45%;"> <div style="border: 1px solid black; padding: 5px; width: 100%;">Posizione attuale</div> <div style="border: 1px solid black; padding: 5px; width: 100%;">Google maps map</div> </div> </div> <p style="text-align: center;">Consegne effettuate oggi [11/03/2019]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID Consegna</th> <th style="padding: 5px;">Orario</th> <th style="padding: 5px;">Via</th> <th style="padding: 5px;">Incasso</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">345</td> <td style="padding: 5px; text-align: center;">19:02:29</td> <td style="padding: 5px;">Via Pedemonte 3</td> <td style="padding: 5px;">54.20.-</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table> </td> </tr> </table>	PizzaDelivery	Consegne	Ordinazioni	Fattorini	<Clienti>	<p style="text-align: center;">FATTORINO #ID</p> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Nome: Paolo</p> <p>Cognome: Neri</p> <p>Stato: Libero</p> </div> <div style="width: 45%;"> <div style="border: 1px solid black; padding: 5px; width: 100%;">Posizione attuale</div> <div style="border: 1px solid black; padding: 5px; width: 100%;">Google maps map</div> </div> </div> <p style="text-align: center;">Consegne effettuate oggi [11/03/2019]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID Consegna</th> <th style="padding: 5px;">Orario</th> <th style="padding: 5px;">Via</th> <th style="padding: 5px;">Incasso</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">345</td> <td style="padding: 5px; text-align: center;">19:02:29</td> <td style="padding: 5px;">Via Pedemonte 3</td> <td style="padding: 5px;">54.20.-</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table>					ID Consegna	Orario	Via	Incasso	345	19:02:29	Via Pedemonte 3	54.20.-	<list item>				<list item>				...			
PizzaDelivery	Consegne	Ordinazioni	Fattorini	<Clienti>																																																													
<p style="text-align: center;">FATTORINI</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID</th> <th style="padding: 5px;">Nome</th> <th style="padding: 5px;">Consegne effettuate oggi</th> <th style="padding: 5px;">Stato</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">3</td> <td style="padding: 5px;">Paolo Neri</td> <td style="padding: 5px; text-align: center;">5</td> <td style="padding: 5px; background-color: #ff0000; color: white;">In servizio</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table>					ID	Nome	Consegne effettuate oggi	Stato	3	Paolo Neri	5	In servizio	<list item>				<list item>				<list item>				...																																								
ID	Nome	Consegne effettuate oggi	Stato																																																														
3	Paolo Neri	5	In servizio																																																														
<list item>																																																																	
<list item>																																																																	
<list item>																																																																	
...																																																																	
PizzaDelivery	Consegne	Ordinazioni	Fattorini	<Clienti>																																																													
<p style="text-align: center;">FATTORINO #ID</p> <div style="display: flex; justify-content: space-between;"> <div style="width: 45%;"> <p>Nome: Paolo</p> <p>Cognome: Neri</p> <p>Stato: Libero</p> </div> <div style="width: 45%;"> <div style="border: 1px solid black; padding: 5px; width: 100%;">Posizione attuale</div> <div style="border: 1px solid black; padding: 5px; width: 100%;">Google maps map</div> </div> </div> <p style="text-align: center;">Consegne effettuate oggi [11/03/2019]</p> <table border="1" style="width: 100%; border-collapse: collapse;"> <thead> <tr style="background-color: #f08080;"> <th style="padding: 5px;">ID Consegna</th> <th style="padding: 5px;">Orario</th> <th style="padding: 5px;">Via</th> <th style="padding: 5px;">Incasso</th> </tr> </thead> <tbody> <tr> <td style="padding: 5px;">345</td> <td style="padding: 5px; text-align: center;">19:02:29</td> <td style="padding: 5px;">Via Pedemonte 3</td> <td style="padding: 5px;">54.20.-</td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;"><list item></td> </tr> <tr> <td colspan="4" style="padding: 5px; text-align: center;">...</td> </tr> </tbody> </table>					ID Consegna	Orario	Via	Incasso	345	19:02:29	Via Pedemonte 3	54.20.-	<list item>				<list item>				...																																												
ID Consegna	Orario	Via	Incasso																																																														
345	19:02:29	Via Pedemonte 3	54.20.-																																																														
<list item>																																																																	
<list item>																																																																	
...																																																																	
<FOOTER>		<FOOTER>																																																															

2.3.6 Pagina Ordinazioni

Pagina dalla quale è possibile visualizzare le ordinazioni, il loro stato e tutte le informazioni importanti una volta cliccato su una di esse.

ORDINAZIONI																											
PizzaDelivery Consegne Ordinazioni Fattorini <Clienti>																											
ORDINAZIONI																											
<table border="1"> <thead> <tr> <th>ID</th> <th>Cliente</th> <th>Posizione</th> <th>Pz. Ordine</th> </tr> </thead> <tbody> <tr> <td>64</td> <td>Franco Rossi</td> <td>Via Scuola 11</td> <td>3</td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4">...</td> </tr> </tbody> </table>				ID	Cliente	Posizione	Pz. Ordine	64	Franco Rossi	Via Scuola 11	3	<list item>				<list item>				<list item>				...			
ID	Cliente	Posizione	Pz. Ordine																								
64	Franco Rossi	Via Scuola 11	3																								
<list item>																											
<list item>																											
<list item>																											
...																											
<FOOTER>																											

ORDINAZIONI ONCLICK											
PizzaDelivery Consegne Ordinazioni Fattorini <Clienti>											
ORDINAZIONE #ID											
Cliente Nome: Franco Cognome: Rossi E-mail: Franco@rossi.ch											
 Via scuole 11											
Prodotti ordinati <table border="1"> <thead> <tr> <th>Nome</th> <th>Prezzo</th> </tr> </thead> <tbody> <tr> <td>1x Pizza margherita</td> <td>12.-</td> </tr> <tr> <td colspan="2"><list item></td> </tr> <tr> <td colspan="2">...</td> </tr> </tbody> </table>				Nome	Prezzo	1x Pizza margherita	12.-	<list item>		...	
Nome	Prezzo										
1x Pizza margherita	12.-										
<list item>											
...											
<FOOTER>											

2.3.7 Pagina Consegne

Pagina dalla quale si possono visualizzare tutte le consegne delle ultime 48 ore.

CONSEGNE																											
PizzaDelivery Consegne Ordinazioni Fattorini <Clienti>																											
CONSEGNE																											
<table border="1"> <thead> <tr> <th>ID Ordine</th> <th>Fattorino</th> <th>Data consegna</th> <th>Stato</th> </tr> </thead> <tbody> <tr> <td>64</td> <td>Paolo Neri</td> <td>12:34:12 24/10/19</td> <td>Consegnato</td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4"><list item></td> </tr> <tr> <td colspan="4">...</td> </tr> </tbody> </table>				ID Ordine	Fattorino	Data consegna	Stato	64	Paolo Neri	12:34:12 24/10/19	Consegnato	<list item>				<list item>				<list item>				...			
ID Ordine	Fattorino	Data consegna	Stato																								
64	Paolo Neri	12:34:12 24/10/19	Consegnato																								
<list item>																											
<list item>																											
<list item>																											
...																											
<FOOTER>																											

3 Implementazione

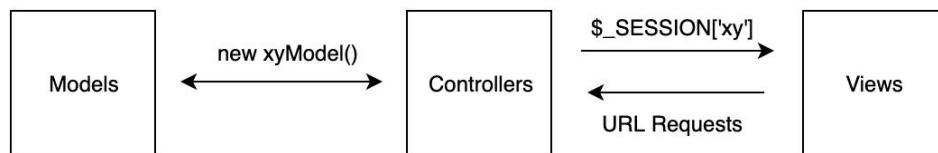
3.1 Struttura dell'applicativo

Tutta la struttura dell'applicativo è stato basato su una base del modello MVC di PHP fornita dal docente Massimo Sartori.

Lo scopo di questo modello Model-View-Controller è quello di semplificare la gestione di un progetto relativamente grande e soprattutto di offrire una visione più globale e pulita su tutti i file che lo compongono e del loro scopo.

La comunicazione tra queste tre principali tipologie di classi e file avviene nel seguente modo:

- I Model comunicano esclusivamente con i controller
- Il controller è la parte principale che collega i model e le views, per richiamare i metodi dai model crea un'istanza di quelli necessari e li richiama attraverso quest'oggetto. Con le views invece mette i dati all'interno della variabile `$_SESSION['xy']` con un nome 'xy' specifico.
- Le views infine ricevono attraverso le sessioni i dati e li stampano, per ricontattare i controllers usano chiamate URL che con il sistema di routing del modello MVC vengono reindirizzare ai metodi della classe model desiderata.



3.1.1 Routing URL

Una delle cose fondamentali dell'MVC è la classe `application.php` che gestisce il routing di tutte le richieste HTTP sotto forma di URL richiamando i metodi dei controller con eventuali parametri corretti.

Il risultato delle richieste viene effettuato nel seguente modo (esempio):

Richiesta URL:	Controller	Metodo	Parametri
pizzadelivery.com/prodotti/getProdotti	prodotti	getProdotti()	-
pizzadelivery.com/utenti/getUtente/paolo.neri	utenti	getUtente()	"paolo.neri"

3.1.2 Gerarchia delle cartelle

Lo scopo di questa suddivisione è di portare ordine al progetto, avendo più di 60 file e 20 immagini è ora possibile accedere in modo rapido all'elemento desiderato:

📁 PizzaDelivery		
└ application		
▶ config	MVC:	Cartella Padre di tutta la struttura MVC
▶ controller	Config:	Contiene il file di configurazione della root della struttura
▶ database	Controller:	Contiene tutte le classi che fanno da controller per l'applicazione
▶ img	Database:	Contiene tutti i file di creazione del database e la classe che collega la struttura MVC con il DB.
▶ libs		
▶ models	Models:	Contiene tutte le classi che fanno da model per l'applicazione
▶ scripts	Scripts:	Contiene tutti i file javascript che le varie views utilizzano per eseguire svariate operazioni sul lato client
└ views	Views:	Contiene tutte le views: <ul style="list-style-type: none">● Header● Body● Footer
└ _templates		
▶ headers		
▶ footer.php		
▶ error		
▶ pages		
▶ css		
 .htaccess		
 index.php		

3.1.3 File di configurazione

I due file principali che si occupano della configurazione di MVC sono config.php che si trova nella cartella application/config e il .htaccess presente nella root del progetto.

3.1.3.1 .htaccess

Si occupa di gestire ed impostare il funzionamento di tutto il traffico di richieste e non, inoltre va ad interfacciarsi con il servizio Apache del WebServer attivando moduli come il RewriteBase che da la possibilità di definire una root per il progetto oppure di riscrivere una regola come il RewriteRule.

Importante: questo file è solamente compatibile con WebServer di tipo Apache, se lo si vuole utilizzare ad esempio con Nginx bisognerà cambiare il formato del contenuto.

```
#Esempio: se la cartella configuration/ e il file configuration.php si trovano nella root del sito e si tenta di
#raggiungere l'indirizzo monsite.com/configuration,
#si verrà automaticamente reindirizzati sul file configuration.php se l'opzione MultiViews è attivata.
Options -MultiViews

# attiva il modulo apache RewriteEngine
RewriteEngine On

# Disallows others to look directly into /public/ folder
Options -Indexes

# Project root
RewriteBase /php/Progetti/PizzaDelivery/src/MVC/

# regole di rewrite generali, se le 3 sotto sono valide allora riscrivi il link (fai partire RewriteCond)
#non eseguirla per una directory
RewriteCond %{REQUEST_FILENAME} !-d
#non eseguirla per regolare file che esiste
RewriteCond %{REQUEST_FILENAME} !-f
#non eseguirla per un link
RewriteCond %{REQUEST_FILENAME} !-l

#riscrivo la regola#In generale: prendi il parametro che trovi e processalo come un classico parametro di
#GET -->?parametro1&parametro2 eccetera
RewriteRule ^(.+)$ index.php?url=$1 [QSA,L]
```

3.1.3.2 config.php

Ha la funzione di specificare l'url del progetto nella costante URL, essa verrà poi usata in tutte le classi ogni volta che bisogna richiamare un metodo di un qualche controller attraverso richiesta HTTP.

```
 */**
 * Configurazione di : URL del progetto
 */
define('URL', 'http://localhost:8888/php/Progetti/PizzaDelivery/src/MVC/');
```

3.1.4 Classi Controller

Lo scopo principale delle classi controller è quello di unire i model con le view fornendo svariati metodi richiamabili attraverso richiesta URL.

Nei loro metodi vengono preparati i dati con un'istanza di una classe Model mettendoli successivamente all'interno della variabile globale `$_SESSION` di php.

Infine richiamano le view corrette attraverso il `require_once` che si occuperà di caricarle e mostrarle a schermo per interagire con l'utente.

Ad esempio la classe `fattorini` esporta il metodo `home()` che viene caricato come pagina di default al click o ricerca dei fattorini, esso mette in sessione tutti i fattorini ricevuto da `fattoriniModel` e richiama le views necessarie:

```
public function home(){
    $_SESSION['fattorini'] = $this->pdModel->getFattorini();

    // Carico Views
    $this->header->getRightHeader();
    require 'application/views/pages/fattorini/fattorini.php';
    require 'application/views/_templates/footer.php';
}
```

Un esempio della struttura di una di queste classi può essere la classe controller `GestionePizzeria.php` che mette a disposizione tutti i suoi metodi che sono richiamabili tramite chiamata HTTP, il diagramma della classe è il seguente:



Se ad esempio volessimo disabilitare un utente (`franco.neri`) basterebbe fare la seguente chiamata con la tipologia di utente corretta:

`http://<serverWebRoot>/PizzaDelivery/gestionePizzeria/disabilita/franco.neri`

3.1.5 Cartella libs

Nella cartella libs sono presenti tutte le librerie scaricate ed usate in questo progetto:

Bootstrap 4.3.1-dist

Si occupa di gestire tutta la struttura html e css delle varie interfacce oltre al loro design e rende anche tutta l'applicazione utilizzabile su tutti le tipologie di dispositivo essendo una libreria creata seguendo l'approccio mobile-first.

jQuery 3.4.1

Libreria basata su JavaScript che semplifica la scrittura del codice lato client con JavaScript offrendo anche metodi suoi per accorciare la lunghezza delle istruzioni.

Popper: 1.3.3

Libreria JavaScript utilizzata da certe operazioni di Bootstrap.

Fontawesome free-5.11.2-web

Libreria di icone che servono a rendere più belle graficamente le pagine html.

3.1.6 Classi Model

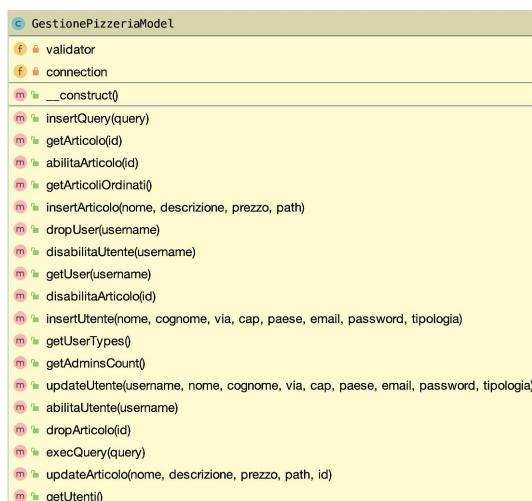
Le classi presenti nella cartella model sono quelle che lavorano direttamente con le fonti di dati come ad esempio un database, file o socket.

Elaborano e preparano i dati in svariati metodi che poi possono essere richiamati da classi esterne (nel nostro caso i controller) attraverso un'istanza del proprio oggetto.

In questo esempio la classe ordinazioniModel.php prepara il metodo getArticoli() che si occupa di ritornare un'array contenente tutti gli articoli presenti nella tabella *articolo* del database.

```
public function getArticoli(){
    return $this->execQuery("SELECT * FROM articolo");
}
```

Spesso la struttura delle classi Model è molto simile, ad esempio la GestionePizzeriaModel.php è strutturata nel seguente modo:



A differenza del controller GestionePizzeria.php il model lavora direttamente con i dati che si trovano sul database, se ad esempio eseguo il metodo *disabilitaArticolo(3)* viene impostato a 0 (false) il valore del campo “articoloAttivo” dell’articolo con id = 3.

3.1.7 Scripts

Generalmente tutti gli script utilizzati dalle varie pagine (file .js) sono salvati in questa cartella. Essi si occupano di eseguire determinate azioni sul lato client dell'applicazione (codice accessibile a qualsiasi utente ispezionando la pagina) a differenza di PHP che lavora lato server.

La funzione validate si occupa di controllare il testo immesso in un input che poi viene passato a questo metodo per poter dare un feedback visivo all'utente.

```
function validate(string, maxLen, regex){  
    try{  
        if(regex === null){  
            return (string.length > 0);  
        }  
  
        regex = new RegExp(regex);  
        string = string.trim();  
  
        if(string.length > 0 && string.length <= maxLen){  
            if(regex.test(string)){  
                return true;  
            }  
        }  
  
        return false;  
    }catch(error){  
        console.log("Error: " + error);  
        return false;  
    }  
}
```

(Metodo semplificato per la dimostrazione, interagisce anche con altri)

A dipendenza del suo esito l'utente vede se il suo input è accettato oppure no.

Informazioni personali

Jari

Informazioni personali

Jari3

Oppure

3.1.8 Views

Nella cartella views sono presenti tutte le pagine del progetto che uniscono l'iterazione dell'utente con quella dell'applicazione.

Mostrano a schermo tutti i dati ricevuti ed elaborati dalle classi controller con un certo senso creando un'interfaccia grafica semplice ed intuitiva da utilizzare accessibile a qualsiasi tipologia di utente.

Un esempio è la pagina di benvenuto che si occupa di accogliere l'utente al suo primo accesso alla pagina.

3.1.9 Classe Database

Lo scopo della classe database.php è quello di stabilire una connessione con il database 'PizzaDelivery' sul quale sono salvati tutti i dati per il funzionamento dell'applicativo e di ritornarla alle classi Model che la utilizzano nel seguente modo:

```
public static function getConnection(){

    if(self::$_connection == null){

        try {
            self::$_connection = new PDO(
                dsn: "mysql:host=" . self::HOST . ";port=" . self::PORT . ";dbname=" . self::DATABASE,
                username: self::USERNAME,
                password: self::PASSWORD
            );
            // set the PDO error mode to exception
            self::$_connection->setAttribute( attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION );
        }catch(PDOException $e){
            header( string: "Location: " . URL . "errorController/requireConnectionError");
        }

    }

    if(isset(self::$_connection)){
        return self::$_connection;
    }else{
        header( string: "Location: " . URL . "errorController/requireConnectionError");
    }
}
```

Oltre a ciò offre anche tutti i metodi che servono ad interagire con il DB, il suo funzionamento è relativamente semplice, basta creare una query ed eseguirla:

```
public function execQuery(string $query){
    try{
        $query = $this->validator->validateString($query);
        $tmp = $this->connection->prepare($query);
        $tmp->execute();
        return $tmp->fetchAll( fetch_style: PDO::FETCH_ASSOC );
    }catch(PDOException $e){
        $_SESSION['queryError'] = $e->getMessage();
        header( string: "Location: " . URL . "errorController/requireQueryError");
    }
}
```

3.2 Database

La fonte principale sulla quale si trovano tutti i dati utilizzati dall'applicazione è il database chiamato PizzaDelivery.

Attraverso la classe database.php spiegata nel capitolo 3.1.9 e l'utente SQL PD_Admin il programma si interfaccia con la banca dati continuando a scambiare, inserire o modificare dati a dipendenza delle operazioni eseguite tramite l'interfaccia grafica costituita dalle varie views.

Per il funzionamento corretto del software il database deve contenere alcuni dati di default/importanti come almeno un'utente amministratore già presente nel database con il quale eseguire il primo login potendo così personalizzare l'applicativo.

Inoltre si necessitano anche dei informazioni come le tre tipologie di utente avanzate (fattorino, impiegato vendita e amministratore) e le tipologie di consegna (da effettuare, in corso e terminata).

Tutti questi dati per il setup iniziale di default si possono trovare nel file **DefaultDataTabelle.sql**.

Inoltre sono anche stati creati dei trigger per semplificare e ridurre i comandi lato PHP, ad esempio quando viene creato un nuovo utente di tipo fattorino viene automaticamente creato un nuovo record nella tabella fattorino con i suoi dati nel seguente modo:

```
DELIMITER //
CREATE TRIGGER addFattorino AFTER INSERT ON Utente FOR EACH ROW
BEGIN
    IF NEW.tipoUtente LIKE 'fattorino' THEN
        INSERT INTO fattorino VALUES (NEW.username, 8.971826, 46.029792, false);
    END IF;
END
//DELIMITER ;
```

Quando invece viene eliminato un utente di tipo fattorino bisogna anche toglierlo nei fattorini per non avere dati incoerenti, per quest'azione non si necessita un trigger essendo sviabile aggiungendo la clausola "ON DELETE CASCADE" sulla chiave esterna che punta al record dell'utente:

```
FOREIGN KEY (username) REFERENCES Utente(username) ON DELETE CASCADE
```

3.3 Sicurezza e ruoli dell'applicativo

Essendo un programma che viene utilizzato commercialmente e che quindi tratta con soldi e dati sensibili di persone deve garantire un livello di sicurezza molto alto.

Questo viene raggiunto implementando ed utilizzando un sistema di login il quale gestisce quattro livelli di sicurezza tra cui si trovano le seguenti tipologie di account.

Utente normale	Persona qualsiasi che effettua un'ordinazione sul sito.					
Fattorino	Fattorino che esegue le consegne a domicilio.					
Impiegato vendita	Impiegato che si occupa di assegnare le varie ordinazioni e monitorare il loro stato.					
Amministratore	Ha il potere completo sull'applicazione potendo anche creare, modificare ed eliminare persone oppure prodotti.					

Queste quattro tipologie di utente possono accedere alle rispettive pagine:

	Home	Ordina	Ordinazioni	Consegne	Fattorini	Gestione Pizzeria
Utente normale	X	X				
Fattorino	X	X		X	X	
Impiegato vendita	X	X	X	X	X	
Amministratore	X	X	X	X	X	X

Attraverso un pannello di login è possibile effettuare l'accesso con un account ottenendo più funzionalità di quelle dell'utente normale.

Se ad esempio accedo con un utente di tipo fattorino l'applicazione si adatterà automaticamente cambiando il contenuto della barra di navigazione mostrando solamente le pagine accessibili nel seguente modo:

PizzaDelivery
Home Ordina Login

(Prima del login)

↓

PizzaDelivery • fattorino
Home Ordina Consegne Fattorini Logout

(Dopo il login)

Inoltre l'applicazione blocca anche l'accesso alle pagine nelle quali non si hanno i permessi sufficienti, se come Fattorino provo ad accedere alla pagina Gestione Pizzeria dell'amministratore il programma mi ritorna il seguente errore:

ERRORE: Permessi insufficienti.

3.4 Codice JavaScript lato client

Per rendere l'applicativo in parte più sicuro ma soprattutto più invitante per l'utente ho implementato vari controlli lato client che danno un feedback visivo all'utente come ad esempio nei form della pagina ConfermaOrdine dove bisogna immettere i propri dati personali.

In ogni form sono stati apportati dei criteri di validazione leggermente differenti a dipendenza del loro scopo, ad esempio il campo che richiede il CAP accetta solamente numeri mentre il campo Nr. (numero della via di casa) accetta sia numeri che lettere.

Nr.	Cap
-----	-----

Quando si immette del testo il bordo dei vari campi cambia a dipendenza del contenuto immesso, se è accettato diventa verde altrimenti rosso.

12A?	12
------	----

Inoltre non è nemmeno possibile confermare il proprio ordine se non sono stati completati tutti i campi obbligatori nel modo corretto, la pagina mostra il seguente errore.

Errore: Immetti correttamente tutti i dati.

La validazione dei vari campi funziona nel seguente modo, la struttura è quasi sempre la stessa e cambia solamente l'espressione regolare che definisce i caratteri accettati.

Inizialmente per ogni input c'è un evento collegato che ad ogni carattere nuovo immesso controlla la validità del contenuto per dare un feedback immediato, questo accade nel seguente modo:

```
// nome field
var nameSelector = $('input[name=nome]');
nameSelector.keyup(function(event){
    if(validate(nameSelector.val(), LUNGHEZZA_MAXIMA_NOME, regex: /^[A-Za-zöäüÖÜäèìòùÀÈÌÒÙÉé\.\-\-]+$/)){
       isOk(nameSelector);
        status[0] = true;
    }else{
        isNotOk(nameSelector);
        status[0] = false;
    }
    checkIfOk();
});
```

(Esempio con l'input del campo "nome")

Una volta controllato il testo con il metodo validate a dipendenza del suo valore di ritorno il bordo del campo viene colorato in rosso oppure verde.

La struttura del metodo validate invece è la seguente, riceve tre parametri tra cui il valore dell'input stesso, la lunghezza massima della stringa definita da una costante ed infine l'espressione regolare contenente i caratteri accettabili.

```
function validate(string, maxLen, regex){  
    try{  
        if(regex === null){  
            return (string.length > 0);  
        }  
  
        regex = new RegExp(regex);  
        string = string.trim();  
  
        if(string.length > 0 && string.length <= maxLen){  
            if(regex.test(string)){  
                return true;  
            }  
        }  
  
        return false;  
    }catch(error){  
        console.log("Error: " + error);  
        return false;  
    }  
}
```

Inizialmente controlla se la regex non è nulla, altrimenti ritorna false. Successivamente crea l'oggetto RegExp con la stringa regex ricevuta come parametro e se la lunghezza della stringa è all'interno dei limiti accettati la testa guardando se rispetta i criteri impostati dall'espressione regolare ritornando true oppure false a dipendenza del suo esito.

Infine i metodi che si occupano di colorare il bordo degli input funzionano molto semplicemente nel seguente modo ricevendo come parametro l'elemento da colorare:

```
function isOk(selector){  
    selector.css('border-color', '#abdd92');  
}
```

Tutti questi controlli lato client sono generalmente effettuati dai vari file javascript presenti nella cartella scripts.

3.5 Pagina Gestione Pizzeria

La pagina che si occupa di gestire tutta la pizzeria è la GestionePizzeria accessibile solamente agli amministratori del sistema.

Tramite i vari form che contiene è possibile aggiungere, modificare, disabilitare ed eliminare utenti oppure articoli offerti dalla pizzeria.

Per ognuna di queste due sezioni esiste una tabella che mostra i record già presenti nel database, inoltre è anche possibile eseguire una ricerca lato client senza dover effettuare alcune query ma lavorando direttamente sui dati ricevuti e presenti nella tabella grazie alla seguente funzione della libreria jQuery:

```
//Tabella Utente
$("#cercaUtente").on("keyup", function() {
    var value = $(this).val().toLowerCase();
    $("#utentiTable tr").filter(function() {
        $(this).toggle($(this).text().toLowerCase().indexOf(value) > -1)
    });
});
```

Questo trucchetto l'ho trovato su W3Schools (https://www.w3schools.com/jquery/jquery_filters.asp) e funziona nel seguente modo.

Ogni volta che l'utente aggiunge o rimuove un carattere nella barra di ricerca viene richiamata la funzione mostrata nell'immagine soprastante, successivamente prende il contenuto della ricerca e la filtra per tutte le righe della tabella togliendo quelle che non contengono un'uguaglianza con il testo immesso.

Un esempio pratico è il seguente:

Gestisci Utenti			
Cerca Utente		+ Crea Utente	
Username	Tipologia	Abilitato	Modifica
franco.gialli	impiegato vendita	✓	
franco.neri	fattorino	✗	
gianni.grandi	fattorino	✓	
jari.naeser	amministratore	✓	

(Nessuna ricerca)

Gestisci Utenti			
Cerca Utente		+ Crea Utente	
Username	Tipologia	Abilitato	Modifica
jari.naeser	amministratore	✓	

(Con ricerca "ja")

Successivamente è possibile aggiungere utenti o articoli attraverso i bottoni “crea Utente”/“crea Articolo”. Cliccando sull'icona della voce modifica in una delle righe si possono modificare i dati del record selezionato oppure disattivarlo.

Progetto PizzaDelivery

Se ad esempio modifico l'utente franco.gialli arrivo alla seguente interfaccia:

Utente franco.gialli

Nome *	Franco
Cognome *	Gialli
Via *	Via Scuole 12
CAP *	6900
Paese *	Svizzera
E-Mail *	franco.gialli@pizzadelivery.ch
Password *	Password non mostrata.
Stato *	<input type="button" value="Disabilita"/>
Tipologia *	impiegato vendita
<input type="button" value="Esci"/> <input type="button" value="Elimina"/> <input type="button" value="Aggiorna"/>	

Se si modificano i valori dell'utente è possibile salvarli attraverso il bottone aggiorna.

Inoltre esiste anche l'opzione "disabilita" che se attivata disabilita l'account rendendolo inutilizzabile ed inaccessibile, questa feature può tornare utile se un impiegato va in ferie per svariato tempo oppure ha un infortunio assentandolo dal lavoro per un grande periodo di tempo.

Disabilitando il suo account basterà un click da parte dell'amministratore quando torna e si evita anche la perdita di dati collegati al suo account come le consegne effettuate nel caso di un fattorino.

Infine è anche stato implementato un ultimo filtro di sicurezza che riguarda l'eliminazione degli utenti amministratori, infatti il sistema deve sempre averne almeno uno.

Se si prova ad eliminare oppure disabilitare l'ultimo account amministratore il programma ritorna il seguente messaggio di errore:

ERRORE

Stai cercando di eliminare o disattivare l'unico admin del sistema.

Se si vuole proseguire con questa azione eliminarlo attraverso l'SQL.

3.6 Implementazione Mappe

Per quanto riguarda le mappe implementate nelle pagine Ordinazione e Fattorino ho utilizzato il servizio gratuito di MapBox il quale rende molto semplice l'implementazione delle mappe con lo stile di Google Maps nelle proprie pagine attraverso JavaScript.

La documentazione è molto intuitiva e permette di aggiungere molti elementi aggiuntivi come puntatori, popup eccetera.

Attraverso una chiave privata ricevuta alla creazione di un account sul loro sito è possibile fare delle richieste HTTP specificando vari parametri come la posizione della mappa che si vuole ricevere oppure i valori di latitudine e longitudine, successivamente se la richiesta è strutturata correttamente il sito ritorna un file in formato JSON contenente tutti i dati necessari per la costruzione della mappa.

Per mostrare la mappa sulle proprie pagine bisogna seguire la loro documentazione utilizzando i vari metodi e funzioni necessari javascript, uno di questi è la funzione onload:

```
map.on('load', function () {
  map.addImage('pulsing-dot', pulsingDot, { pixelRatio: 2 });

  map.addLayer({
    "id": "places",
    "type": "symbol",
    "source": {
      "type": "geojson",
      "data": {
        "type": "FeatureCollection",
        "features": [
          {
            "type": "Feature",
            "properties": {
              "description": "<strong>Fattorino: <?php echo $userFattore; ?>",
              "icon": "theatre"
            },
            "geometry": {
              "type": "Point",
              "coordinates": [<?php echo $fattorino['posizioneLat']; ?>, <?php echo $fattorino['posizioneLon']; ?>]
            }
          }
        ]
      },
      "layout": {
        "icon-image": "pulsing-dot",
        "icon-allow-overlap": true
      }
    });
});
```

Si occupa di impostare i valori più importanti per la mappa come la posizione ed il puntatore, una volta implementato e personalizzato correttamente tutti i metodi e funzioni si ottiene la propria mappa.

Nel mio caso la posizione attuale di ogni fattorino viene mostrata su una mappa basata su MapBox:



3.7 Approccio Mobile First

Essendo uno dei temi più importanti nella realizzazione di applicativi basati su pagine web ho ritenuto necessario creare un programma che sia utilizzabile ed apprezzabile graficamente su qualsiasi tipologia di dispositivo.

Per poter far ciò ho utilizzato un approccio Mobile first usando la libreria Bootstrap che si occupa di gestire automaticamente gran parte dell'adattamento delle dimensioni a dipendenza della grandezza dello schermo sul quale viene aperta l'applicazione.

Ecco un paragone tra un dispositivo mobile ed un computer desktop:

The image shows a side-by-side comparison of the PizzaDelivery website's mobile and desktop versions. Both versions feature a red header bar with the 'PizzaDelivery' logo and a menu icon. The mobile version has a large 'Benvenuto da PizzaDelivery' title and a 'Hai fame? Ecco la soluzione per te, ordina e ricevi in breve tempo la tua pizza preferita.' message, with a prominent red 'Ordina ora' button. The desktop version has a similar layout but includes a top navigation bar with 'Home', 'Ordina', and 'Login' links, and a smaller message below the main title.

(Versione Mobile)

© 2019 Copyright: Jari Näser

(Versione Desktop)

© 2019 Copyright: Jari Näser

3.8 Installazione debugger

Il debugger che ho scelto di utilizzare in questo progetto è xDebug.

3.8.1 Download

Usando MAMP non ho dovuto effettuare alcun tipo di download essendo che la sua libreria esiste già nella cartella della versione di php che si utilizza attualmente.

Path sul mio computer: /Applications/MAMP/bin/php/php7.2.1/lib/php/extensions/no-debug-non-zts-20170718/xdebug.so

3.8.2 Configurazione php.ini

Per poter implementare il debugger bisogna aprire il file di configurazione php.ini della versione di php utilizzata attualmente dal WebServer.

Per semplificare questo passo è possibile farlo anche dalla IDE di phpStorm sotto “Preferences > Languages & Frameworks > PHP”.

Una volta in questo tab bisogna cliccare i tre puntini che stanno di fianco a CLI Interpreter:



E poi successivamente cliccare “Open in Editor”

Configuration file: /Applications/MAMP/bin/php/php7.2.1/conf/php.ini [Open in Editor](#)

Una volta aperto il file bisogna aggiungere/modificare i seguenti elementi:

Togliere Zend debugger e Zend tools (rimuovere o commentare le sue seguenti righe)

```
zend_extension=<path_to_zend_debugger>
zend_extension=<path_to_zend_optimizer>
```

Spostarsi in fondo al file alla voce [xdebug] e settare la path di xdebug nel zend_extension

```
zend_extension="/Applications/MAMP/bin/php/php7.2.1/lib/php/extensions/no-debug-non-zts-20170718/xdebug.so"
```

Aggiungere le ulteriori righe di configurazione (non copiare anche le descrizioni delle righe in *italico*)

xdebug.remote_autostart = 1	Inizia una sessione di debug esterna
xdebug.profiler_append = 0	Aggiunge il profiler al nuovo profilo
xdebug.profiler_enable = 0	Disabilita il profiler
xdebug.profiler_enable_trigger = 0	Disabilita il trigger del profiler
xdebug.profiler_output_dir = "/tmp"	Imposta la cartella di output del profiler
xdebug.remote_enable = 1	Definisce se contattare un client di debug (phpstorm)
xdebug.remote_handler = "dbgp"	Definisce il protocollo di debug
xdebug.remote_host = "localhost"	Definisce l'host remoto a cui collegarsi
xdebug.remote_log = "/tmp/xdebug.log"	Definisce il file in cui vengono messi i log
xdebug.remote_port = 10000	Definisce la porta di ascolto remota
xdebug.trace_output_dir = "/tmp"	Definisce la cartella in cui mette l'output
xdebug.remote_cookie_expire_time = 36000;	Definisce il tempo di esistenza di un cookie, in questo caso 36000 secondi equivalgono a 10 ore

Progetto PizzaDelivery

3.8.3 Configurazione phpStorm

Per poter utilizzare xDebug anche sull'IDE che ho utilizzato per lo sviluppo di questo progetto, phpStorm, basta eseguire pochi passi molto semplici.

Inizialmente bisogna aprire le preferenze e andare nel tab "PHP" sotto "Languages & Frameworks > PHP".

Successivamente si devono cliccare i tre puntini che stanno alla destra della versione del CLI interpreter

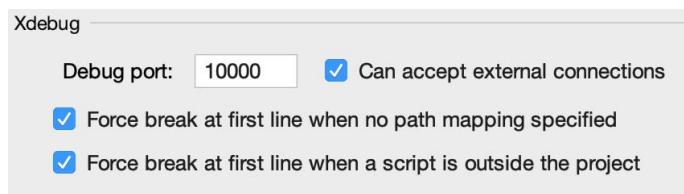


Una volta cliccati dobbiamo inserire la path del file xdebug.so (.dll su windows) inserita precedentemente nel php.ini in "zend_extension" nel textbox "Debugger extension:"

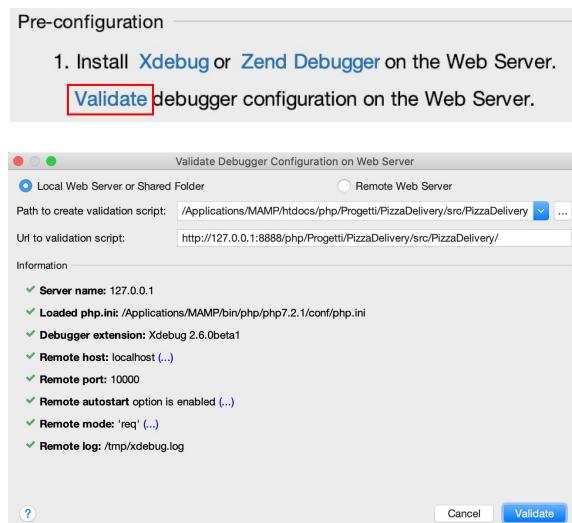


Una volta fatto questo premere "Apply" e "OK" per uscire da questa schermata.

Come passo successivo dobbiamo andare nella voce "debug" del tab PHP nel quale dobbiamo aggiornare la porta di connessione di xdebug inserendo quella specificata precedentemente con "xdebug.remote_port = 10000" nel file php.ini



Infine possiamo cliccare "Validate debugger" che esegue dei test sulle impostazioni appena modificate e se tutto è andato a buon fine appare la seguente finestra



Infine per eseguire un test di debug basta mettere un breakpoint in un file e cliccare l'icona del telefono e quella del "bug", successivamente appena l'esecuzione del codice arriverà al punto del breakpoint l'IDE entrerà in modalità debug mostrando tutti i dati.



4 Test

4.1 Protocollo di test

Test Case:	TC-01	Nome:	Test del database
Riferimento:	REQ-03		
Descrizione:	Test sul funzionamento corretto della banca dati utilizzata dal progetto		
Prerequisiti:	Disporre di un database funzionante e costruito rispettando il diagramma ER		
Procedura:	<ol style="list-style-type: none"> Collegarsi con il database PizzaDelivery attraverso l'utente PD_Admin Eseguire una insert nella tabella TipoConsegna: <code>INSERT INTO TipoConsegna VALUES ("in corso");</code> 		
Risultati attesi:	La tabella TipoConsegna contiene il nuovo record con il valore "in corso"		

Test Case:	TC-02	Nome:	Test struttura MVC
Riferimento:	REQ-04		
Descrizione:	Test del funzionamento corretto della struttura MVC, viene soprattutto testato il routing corretto all'interno dell'applicazione		
Prerequisiti:	Disporre della struttura MVC configurata correttamente e funzionante		
Procedura:	<ol style="list-style-type: none"> Effettuare una richiesta HTTP tramite browser che richiama il controller home: <code>http://<server>/<htdocs>/PizzaDelivery/home/index</code> 		
Risultati attesi:	Il metodo index del controller home ci ritorna la pagina index di default		

Test Case:	TC-03	Nome:	Test connessione tra modello MVC e database
Riferimento:	-		
Descrizione:	Test del funzionamento e comunicazione corretta tra la struttura MVC e il database		
Prerequisiti:	Disporre di un database funzionante contenente dei dati nella tabella TipoUtente Disporre del file database.php presente di default nella struttura MVC Disporre di una struttura MVC funzionante		
Procedura:	<ol style="list-style-type: none"> Creazione di una classe model di test nella cartella models Creazione di un'oggetto di tipo Database utilizzando la classe Database Eseguire il metodo getConnection dall'oggetto Database Utilizzare il metodo prepare() della connessione inserendo la seguente query: <code>SELECT * FROM TipoUtente;</code> Richiamare il metodo execute() della connessione Eseguire il fetch() della connessione ricavando i dati ritornati dal database 		
Risultati attesi:	Il database ritorna le varie tipologie di utente presenti nella tabella TipoUtente.		

Progetto PizzaDelivery

Test Case:	TC-04	Nome:	Test funzionamento pagina Ordina
Riferimento:	-		
Descrizione:	Test del funzionamento corretto della pagina Ordina e delle sue sotto-pagine.		
Prerequisiti:	Disporre della pagina Ordina funzionante e completa		
Procedura:	<ol style="list-style-type: none"> 1. Accedere alla pagina Ordina 1. Creazione del proprio ordine simulando quello di un'utente qualsiasi 2. Proseguire alla pagina conferma ordine tramite il bottone "Proseguì all'ordinazione" 3. Immettere correttamente i propri dati personali 4. Definire la quantità desiderata per ogni articolo 5. Cliccare il bottone "Ordina" 		
Risultati attesi:	Deve apparire la pagina che comunica "Il tuo ordine verrà elaborato il più presto possibile"		

Test Case:	TC-05	Nome:	Test funzionamento pagina Login
Riferimento:	-		
Descrizione:	Test del funzionamento corretto della pagina di login, deve rispettare tutti i criteri di sicurezza facendo effettuare un login in modo sicuro.		
Prerequisiti:	Disporre di almeno un account con il quale poter effettuare il login e dei livelli di sicurezza come diverse tipologie di account		
Procedura:	<ol style="list-style-type: none"> 2. Accedere alla pagina Login 3. Immettere il nome utente dell'account scelto 4. Immettere la password dell'account scelto 		
Risultati attesi:	Login effettuato correttamente, a dipendenza della tipologia dell'utente testato si possono accedere svariate pagine in più rispetto all'utente qualsiasi.		

Test Case:	TC-06	Nome:	Test funzionamento pagina Ordinazioni
Riferimento:	REQ-05		
Descrizione:	Test del funzionamento corretto della pagina Ordinazioni		
Prerequisiti:	Aver effettuato il login con un account di tipologia impiegato vendita oppure amministratore		
Procedura:	<ol style="list-style-type: none"> 1. Accedere alla pagina Ordinazioni 2. Assegnare un'ordinazione ad un fattorino tramite il bottone "Assegna a fattorino" 		
Risultati attesi:	L'ordinazione assegnata ad un fattorino deve sparire dalle ordinazioni, si è spostata automaticamente nelle consegne.		

Progetto PizzaDelivery

Test Case:	TC-07	Nome:	Test funzionamento pagina Consegne
Riferimento:	REQ-06		
Descrizione:	Test del funzionamento corretto della pagina Consegne		
Prerequisiti:	Aver effettuato il login con un account di tipologia superiore all'utente qualsiasi (quindi fattorino, impiegato vendita oppure amministratore)		
Procedura:	<ol style="list-style-type: none"> 1. Accedere alla pagina Consegne 2. Modificare il filtro che mostra le consegne a dipendenza della loro data d'inserimento "Fino a" 3. Modificare lo stato di una consegna 		
Risultati attesi:	Modificando il filtro "Fino a" si vedono diverse quantità di record, inoltre modificando lo stato di una consegna si vede anche visualmente il cambiamento.		

Test Case:	TC-08	Nome:	Test funzionamento pagina Fattorini
Riferimento:	REQ-07		
Descrizione:	Test del funzionamento corretto della pagina Fattorini		
Prerequisiti:	Aver effettuato il login con un account di tipologia superiore all'utente qualsiasi (quindi fattorino, impiegato vendita oppure amministratore)		
Procedura:	<ol style="list-style-type: none"> 4. Accedere alla pagina Fattorini 1. Cliccare su uno dei fattorini 		
Risultati attesi:	Caricamento della pagina fattorino cliccato con tutti i suoi dati		

Test Case:	TC-09	Nome:	Test funzionamento pagina GestionePizzeria
Riferimento:	-		
Descrizione:	Test del funzionamento corretto della pagina GestionePizzeria		
Prerequisiti:	Aver effettuato il login con un account di tipologia amministratore.		
Procedura:	<ol style="list-style-type: none"> 1. Accedere alla pagina GestionePizzeria I Prossimi test effettuarli sia per gli utenti che per gli articoli: 2. Eseguire una ricerca 3. Aggiungere un nuovo elemento 4. Modificare l'elemento creato 5. Disabilitare l'elemento creato 6. Eliminare l'elemento creato 		
Risultati attesi:	<p>Alla ricerca il sistema mostra solamente gli elementi nelle due tabelle che corrispondono al valore cercato, alla creazione di un nuovo elemento esso viene creato, se viene modificato e si salvano le modifiche l'elemento applica i nuovi valori modificati anche nel database.</p> <p>Quando viene cliccato il bottone "disabilita" l'account viene disabilitato ed infine alla sua eliminazione viene cancellato dal database.</p>		

Test Case:	TC-10	Nome:	Test dei criteri di sicurezza
Riferimento:	-		
Descrizione:	Test del corretto funzionamento dei criteri di sicurezza		
Prerequisiti:	Disporre di un programma completamente funzionante		
Procedura:	<ol style="list-style-type: none"> 1. Accedere con un utente qualsiasi alla pagina Gestione Pizzeria 2. Eseguire il login con un account disabilitato 3. Eseguire delle SQL injection immettendo valori pericolosi nei vari form 4. Eliminare oppure disabilitare l'ultimo utente amministratore 5. Richiamare il metodo disabilitaUtente senza aver effettuato il login disabilitando un'utente a scelta: <a href="http://<server>/<htdocs>/PizzaDelivery/gestionePizzeria/disabilitaUtente/<nome.utente>">http://<server>/<htdocs>/PizzaDelivery/gestionePizzeria/disabilitaUtente/<nome.utente> 		
Risultati attesi:	<p>Il programma ritorna la pagina di errore “permessi insufficienti” se si tenta di accedere una pagina privata come GestionePizzeria senza avere i permessi di amministratore.</p> <p>Il login con un account disabilitato non è possibile e anche le SQL injection non vengono eseguite grazie ai svariati controlli prima dell'esecuzione delle query.</p> <p>Non è possibile disabilitare oppure eliminare l'ultimo amministratore tramite interfaccia grafica e il metodo disabilitaUtente non può venir richiamato senza aver effettuato il login.</p>		

Test Case:	TC-11	Nome:	Funzionamento corretto algoritmo path finding
Riferimento:	-		
Descrizione:	Test del corretto funzionamento dell'algoritmo di path finding che trova la strada più corta da percorre per un fattorino quando esegue più consegne in una sola uscita		
Prerequisiti:	Disporre di un programma completamente funzionante		
Procedura:	<p>Accedere alla pagina consegne</p> <p>Visualizzare il percorso trovato dall'algoritmo in una consegna</p>		
Risultati attesi:	Il programma ritorna il percorso più breve che il fattorino dovrà effettuare unendo più consegne.		

4.2 Risultati test

Test case	Esito Test	Note
TC-01	PASSATO	
TC-02	PASSATO	
TC-03	PASSATO	
TC-04	PASSATO	
TC-05	PASSATO	
TC-06	PASSATO	
TC-07	PASSATO	
TC-08	PASSATO	
TC-09	PASSATO	
TC-10	PASSATO	
TC-11	NON PASSATO	Non implementato

4.3 Mancanze/limitazioni conosciute

Le uniche mancanze del programma rispetto alle specifiche sono la rilevazione di trovare il percorso più breve possibile quando un fattorino deve eseguire più consegne in una sola uscita, non è stato implementato per la mancanza di tempo e la complessità del problema.

Inoltre quando si vuole modificare lo stato di una consegna non è già selezionato di default il valore attuale, sarebbe da implementare per portare più facilità nell'uso.

5 Installazione

Per poter utilizzare questa applicazione bisogna seguire un paio di passi molto semplici ma fondamentali per il corretto funzionamento dell'applicazione.

5.1 Requisiti iniziali

- Bisogna avere una macchina che faccia da WebServer (Ad esempio con software ...AMP) con:
 - Almeno 200MB di spazio disponibile sul disco
 - PHP installato, versione minima dell'interprete: 7.2.1
 - Servizio MySQL attivo (versione consigliata, da 5.6.38 in su)

5.2 Copiatura della cartella PizzaDelivery sulla propria macchina

Per poter usare l'applicazione bisogna copiare tutta la cartella "PizzaDelivery" ed il suo contenuto presente nella cartella "src" della repository di GitHub nella propria cartella "htdocs" (cartella di default dove apache cerca i file da servire) sul proprio WebServer.

ATTENZIONE: Assicurarsi di copiare tutti i file, anche quelli nascosti come il `.htaccess` altrimenti l'applicazione non funziona.

5.3 Creazione DataBase e inserimento dei dati essenziali

Per la creazione del database basta eseguire il file `PizzaDeliveryDB.sql` attraverso console oppure programmi come MySQL WorkBench avendo il servizio MySQL attivo.

Questo file che si trova in: `PizzaDelivery/application/database/PizzaDeliveryDB.sql` e contiene tutta la struttura del database oltre all'utente che permette di accedere ad MVC al database.

Se il database non è sulla stessa macchina del WebServer bisogna andare a modificare l'indirizzo di accesso dell'utente dal quale esegue le query, quindi l'ip pubblico della macchina WebServer:

Modificare da:

```
DROP USER 'PD_Admin'@'localhost';

CREATE USER 'PD_Admin'@'localhost' IDENTIFIED BY 'P0r74C431CaTo3Enwicncw';
GRANT ALL PRIVILEGES ON PizzaDelivery.* TO 'PD_Admin'@'localhost';
FLUSH PRIVILEGES;
```

A:

```
DROP USER 'PD_Admin'@'10.20.4.135';

CREATE USER 'PD_Admin'@'10.20.4.135' IDENTIFIED BY 'P0r74C431CaTo3Enwicncw';
GRANT ALL PRIVILEGES ON PizzaDelivery.* TO 'PD_Admin'@'10.20.4.135';
FLUSH PRIVILEGES;
```

Altrimenti lasciare "localhost" se tutto si trova sulla stessa macchina.

Per modificare la password invece basta cambiare il valore del campo:

```
IDENTIFIED BY 'NuovaPassword';
```

Infine eseguiamo anche il file DefaultDataTabelle.sql che contiene i dati default del database come le tipologie di utente, utente amministratore eccetera.

Il file è presente nel percorso: *PizzaDelivery/application/database/DefaultDataTabelle.sql*

Se si vogliono modificare dati è consigliato farlo dall'interfaccia "Gestione pizzeria" una volta eseguito il login con l'account admin (admin.pizzadelivery nel nostro caso) e dopo l'esecuzione dell'sql.

5.4 Impostazione subfolder della cartella webroot

Per poter permettere al sistema MVC di funzionare correttamente deve sapere in quale percorso (subfolder) si trova della webroot (cartella htdocs).

Per impostare correttamente questo percorso basta aprire con un qualsiasi editore di testo il file .htaccess presente nella cartella PizzaDelivery ed andare a modificare la seguente riga dove si trova questo percorso di default:

```
# When using the script within a sub-folder, put this path here, like /mysubfolder/
# If your app is in the root of your web folder, then leave it commented out
RewriteBase /php/Progetti/PizzaDelivery/src/PizzaDelivery/
```

Bisogna immettere il percorso del file .htaccess rispettivamente alla cartella htdocs del proprio WebServer:

```
RewriteBase /Il/Tuo/Percorso/dalla/cartella/htdocs
```

Se si mette direttamente la cartella PizzaDelivery nella htdocs il RewriteBase risulterà così:

```
RewriteBase /PizzaDelivery/
```

5.5 Modifica percorso URL del progetto

Per poter effettuare tutte le richieste correttamente all'interno della struttura MVC bisogna impostare la costante "URL" che si trova nel file di configurazione nel seguente percorso:

PizzaDelivery/application/config/config.php

Una volta aperto il file con un qualsiasi editor di testo possiamo andare nella riga dove viene dichiarato l'URL e modificare il suo valore.

```
 /**
 * Configurazione di : URL del progetto
 */
define('URL', 'http://localhost:8888/php/Progetti/PizzaDelivery/src/PizzaDelivery/');

Esempio di configurazione.
```

Se si vuole rendere accessibile l'applicazione da ovunque digitare (consigliato per questo progetto):

```
define('URL', 'http://<ip pubblico>:<porta apache>/<percorso dalla cartella htdocs, uguale a quello immesso precedentemente>/');
```

Se invece si vuole solamente provarla senza dar la possibilità ad altri utenti di raggiungerla digitare:

```
define('URL', 'http://localhost:<porta>/<percorso dalla cartella htdocs, uguale a quello immesso precedentemente>/');
```

Un risultato finale potrebbe essere il seguente se mettiamo la cartella PizzaDelivery direttamente nella htdocs:

```
define('URL', 'http://10.20.4.145:8080/PizzaDelivery/');
```

5.6 Configurazione accesso MySQL per interagire con il DataBase

Come ultima operazione essenziale per il funzionamento dell'applicazione dobbiamo andare nella classe database.php che è quella che connetterà tutta la parte PHP del programma con il Database e modificare i valori delle costanti.

Il file si trova nel seguente percorso: *PizzaDelivery/application/database/database.php*

Le costanti da modificare sono:

- **HOST**: IP pubblico della macchina sulla quale è presente il DB, oppure “localhost” se è la stessa del WebServer.
- **USERNAME ***: Username dell'utente con il quale verranno effettuate le query sul DataBase.
- **PASSWORD ***: Password dell'utente
- **DATABASE ***: Nome del DataBase
- **PORT**: Porta sulla quale ascolta il servizio MySQL

* Campi già impostati di default che rispettano la struttura del programma, sconsigliato modificarli.

5.7 Primo accesso tramite interfaccia grafica

Una volta essendosi assicurati di aver effettuato tutti i passi correttamente del capitolo 5 possiamo accedere al nostro programma tramite browser, per far ciò basta andare nel percorso nel quale si ha piazzato la cartella PizzaDelivery.

Una volta raggiunto l'index con <http://<serverWebRoot>/PizzaDelivery> possiamo andare sulla voce “login” nella barra di navigazione ed entrare con il seguente utente amministratore:

- Username: admin.pizzadelivery
- Password: test

ATTENZIONE: Una volta eseguito il login è vivamente consigliato andare nella pagina Gestione Pizzeria e premere il bottone “Crea Utente”, successivamente creare un nuovo utente di tipologia amministratore ed eliminare admin.pizzadelivery oppure cambiarli la password.

Con l'account amministratore è possibile allestire il programma come meglio si preferisce, aggiungendo nuovi utenti ed articoli.

Progetto PizzaDelivery

Consuntivo

Come si può vedere dal Gantt consuntivo le tempistiche sono principalmente state rispettate.
 L'unica parte che ha subito delle modifiche è quella dell'implementazione essendo che lo sviluppo di certe pagine richiedeva più tempo che quello di altre.

Inoltre sono anche stati anticipati e prolungati le fasi di test, dopo ogni modifica che effettuavo eseguivo una prova del suo funzionamento oltre ad aver effettuato nuovamente tutti i test alla fine dell'implementazione.



6 Conclusioni

Questa tipologia di applicazione è già presente sul mercato e viene anche adoperata da molte pizzerie. Purtroppo ogni volta che un ristorante vuole offrire un servizio come la consegna di pizze a domicilio devono contattare programmati oppure aziende che sviluppano questo software esclusivamente per loro dovendo pagare enormi somme che in molti casi non sono possibili finanziarie.

Attraverso il prodotto PizzaDelivery diventa possibile gestire in modo semplice la propria pizzeria per quanto riguarda il lato delle consegne e ordinazioni, inoltre rimane anche molto intuitivo il processo di ordinazione per una tipologia qualsiasi di utente.

Infine il vantaggio che offre questo software è quello di poter essere acquistato da svariate pizzerie dando poi l'opzione di personalizzarlo ma soprattutto ad un costo molto più accessibile rendendolo molto competitivo sul mercato.

6.1 Sviluppi futuri

Per quanto riguarda gli sviluppi futuri ci sarebbero varie cose che si potrebbero aggiungere oppure migliorare.

Inizialmente sarebbe da implementare l'algoritmo di pathfinding per i percorsi dei vari fattorini guadagnando tempo e denaro (meno sprechi di benzina). Inoltre restando sempre sui fattorini si potrebbero aggiungere dei sensori di posizione su ogni veicolo oppure sul fattorino stesso per poter aggiornare in tempo reale la mappa del sito fornendo dati attuali e precisi.

Un'ulteriore aggiunta potrebbe essere l'implementazione del pagamento online evitando così lo scambio di denaro tra cliente e fattorino centralizzando il tutto su pagamenti online.

Si potrebbe anche aggiungere l'opzione di far creare un account ai clienti evitando così di dover sempre immettere i propri dati personali e quelli della carta di credito.

Infine per evitare la ricezione di comande eseguite da script l'aggiunta di una qualche tipologia di recaptcha potrebbe risolvere la problematica.

6.2 Considerazioni personali

Personalmente ho trovato molto interessante accettare questa sfida essendo che era il mio primo progetto di queste dimensioni, di conseguenza era qualcosa di nuovo anche per me.

Durante il percorso di progettazione ed implementazione ho imparato moltissime cose nuove potendo consolidare varie nozioni apprese già precedentemente nei vari moduli seguiti negli scorsi 3 anni oltre ad arricchire le mie conoscenze.

Ho anche capito cosa vuol dire dover progettare e successivamente implementare un progetto abbastanza grande se uno di dimensioni relativamente piccole come questo mi ha già fatto pensare a tantissime cose, casi d'uso e messo vari dubbi che senza una buona riflessione avrei magari svolto in un modo sbagliato. Sono molto contento di aver scelto questo progetto e anche dei risultati ottenuti.

7 Bibliografia

7.1 Sitografia

- <https://www.php.net/manual/en/>, Manuale ufficiale di PHP, 24-09-2019 - 22-11-2019.
- <https://stackoverflow.com/>, StackOverFlow, 24-09-2019 - 22-11-2019.
- <https://www.w3schools.com/php/>, W3Schools PHP, 24-09-2019 - 22-11-2019.
- <https://www.w3schools.com/js/>, W3Schools Java Script, 24-09-2019 - 22-11-2019.
- <https://www.w3schools.com/bootstrap/>, W3Schools Bootstrap, 24-09-2019 - 22-11-2019.
- <https://getbootstrap.com/docs/4.3/getting-started/introduction/>, Manuale ufficiale di Bootstrap, 24-09-2019 - 22-11-2019.
- <https://api.jquery.com/>, Manuale ufficiale di jQuery, 24-09-2019 - 22-11-2019.
- <https://xdebug.org/docs/remote>, Manuale ufficiale di xDebug, 15-11-2019 - 22-11-2019.
- <https://www.draw.io/>, Sito per creare tutti i diagrammi, 24-09-2019 - 22-11-2019.
- <https://docs.mapbox.com/mapbox-gl-js/api/>, Documentazione delle mappe utilizzate nel progetto, 24-09-2019 - 22-11-2019.
- https://www.ilovepdf.com/it/unire_pdf, Sito per unire svariati PDF utilizzato per i diari, 13-12-2019

8 Allegati

- Abstract
- Documentazione del programma (con istruzione di installazione ambiente + installazione xDebug)
- Quaderno Dei Compiti
- Diari di lavoro
- Codice sorgente (sulla piattaforma GitHub: <https://github.com/JariNaeser/PizzaDelivery>)

Diari

Diario di lavoro

Luogo	Scuola Arti e Mestieri Trevano
Data	03.09.19

Lavori svolti

Oggi ho conosciuto il mio nuovo progetto che si tratta di un sistema per la gestione di una ditta di pizze d'asporto.

Il mio compito è quello di creare una piattaforma sulla quale si possano visualizzare principalmente gli ordini e varie informazioni sui propri fattorini.

Inoltre ho creato la repository su github con una struttura di cartelle che in parte ho già cominciato a popolare.

Problemi riscontrati e soluzioni adottate

Non ho incontrato nessun problema

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto alla pianificazione

Programma di massima per la prossima giornata di lavoro

Creazione di un diagramma Gantt

Schema database

Analisi dei requisiti

Diario di lavoro

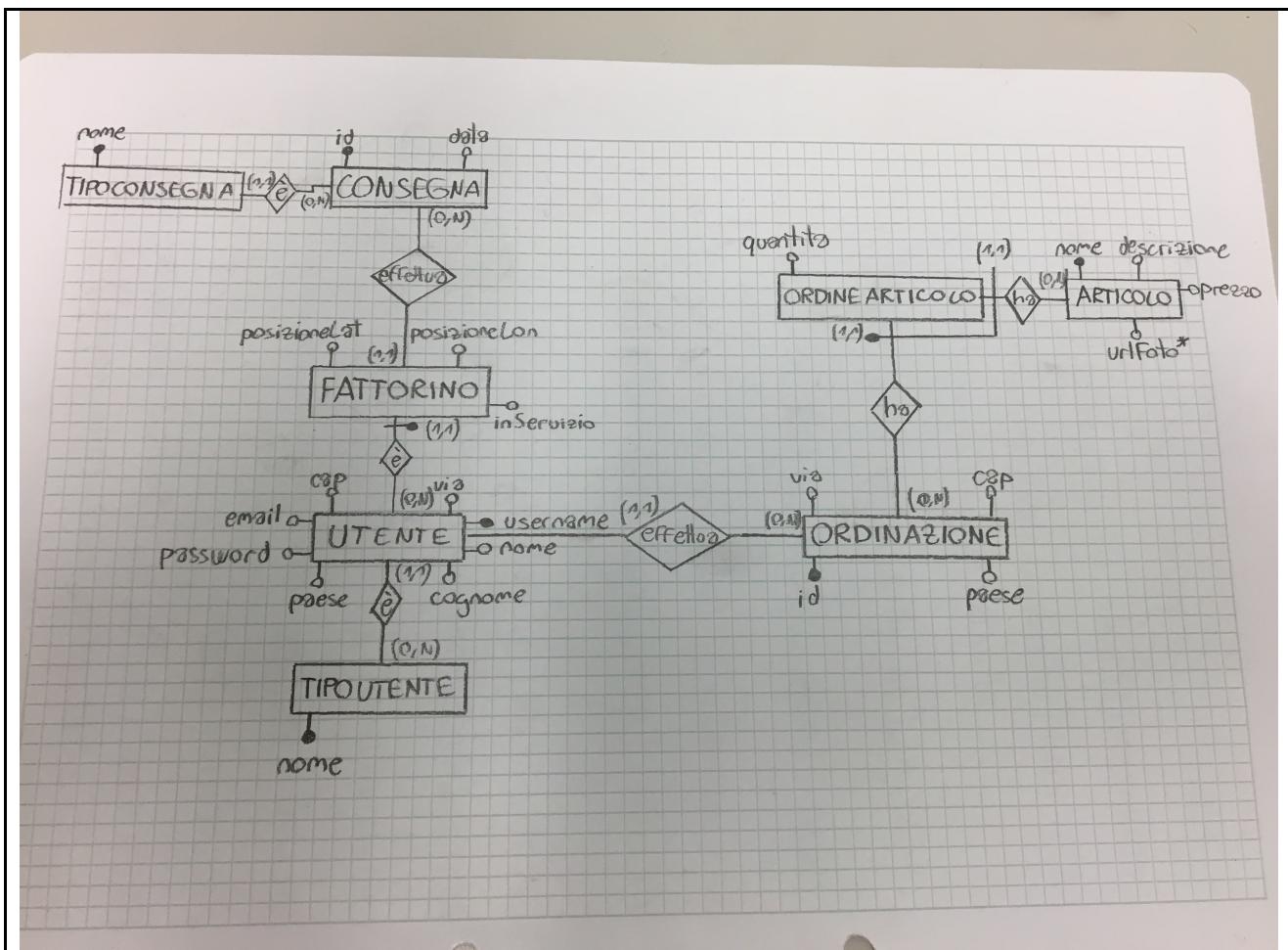
Luogo	Scuola Arti e Mestieri Trevano
Data	05.09.19

Lavori svolti

Oggi ho iniziato con lo sviluppo del diagramma di Gantt con il quale sono arrivato ad un buon punto anche se devo ancora approfondire l'implementazione ed i test.

Pizza Delivery			
Analisi	3 days	Tue 03/09/19	Fri 06/09/19
Analisi dei requisiti	0,75 days	Tue 03/09/19	Tue 03/09/19
Analisi del dominio	1 day	Tue 03/09/19	Thu 05/09/19
Analisi DataBase	0,75 days	Thu 05/09/19	Thu 05/09/19
Progettazione DataBase	1,5 days	Thu 05/09/19	Fri 06/09/19
Analisi struttura progetti	1,5 days	Fri 06/09/19	Tue 10/09/19
Schizzo delle interfacce	0,75 days	Tue 10/09/19	Tue 10/09/19
Implementazione	25 days	Tue 10/09/19	Tue 26/11/19
Creazione DataBase	2,25 days	Tue 10/09/19	Fri 13/09/19
Implementazione modello MVC base	1,5 days	Tue 17/09/19	Thu 19/09/19
Creazione pagine principali	0,75 days	Thu 19/09/19	Thu 19/09/19
Creazione modulo per connessione al DataBase	1,5 days	Thu 19/09/19	Fri 20/09/19
Connessione al DataBase	0 days	Tue 24/09/19	Tue 24/09/19
Sviluppo pagine principali	20,25 days	Tue 24/09/19	Fri 22/11/19
Creazione grafica pagina	12 days	Fri 18/10/19	Fri 22/11/19
Test	33,75 days	Tue 10/09/19	Fri 20/12/19
Test funzionamento corretto DataBase	0,75 days	Thu 19/09/19	Thu 19/09/19
Test struttura base MVC	0,75 days	Fri 20/09/19	Fri 20/09/19
Test routing corretto MVC	0,75 days	Fri 20/09/19	Fri 20/09/19
Test interfacce	9 days	Tue 26/11/19	Fri 20/12/19
Funzionamento corretto di elementi come buttoni, link e mappe eccetera	9 days	Tue 26/11/19	Fri 20/12/19
Test funzionamento azioni backend	9 days	Tue 26/11/19	Fri 20/12/19
Documentazione	36 days	Tue 03/09/19	Fri 20/12/19
Documentazione Word	36 days	Tue 03/09/19	Fri 20/12/19
Sviluppo presentazione	3 days	Fri 13/12/19	Fri 20/12/19

Inoltre mi sono anche dato da fare con la progettazione del database nella quale ho iniziato creando uno schema logico estrapolando tutte le informazioni importanti dalle consegne del QdC e successivamente l'ho tradotto in uno schema ER su carta; anche questo ancora da ottimizzare.
Vedi immagine nella prossima pagina.



Problemi riscontrati e soluzioni adottate

Ho incontrato alcuni problemini nel strutturare il DB ma dopo averci ragionato sopra sono riuscito a risolverli (ad esempio che tabelle fare, quali attributi mettere dove e come semplificare).

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto alla pianificazione

Programma di massima per la prossima giornata di lavoro

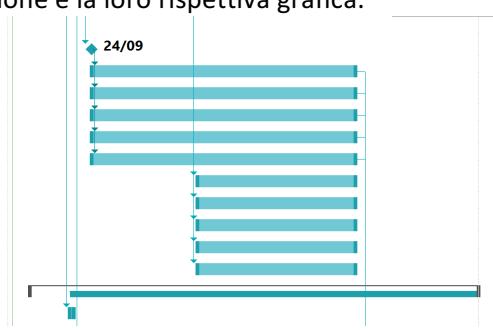
Finire il diagramma di Gantt

Portarmi avanti se non finire con la progettazione del DB.

Diario di lavoro

Luogo	Scuola Arti e Mestieri Trevano
Data	06.09.19

Lavori svolti			
Oggi ho continuato e finito la prima versione del mio diagramma di gantt aggiungendo svariate fasi di lavoro più precise riguardanti tutte le diverse pagine dell'applicazione e la loro rispettiva grafica.			
DataBase			
Connessione al DataBase	0 days	Tue 24/09/19	Tue 24/09/19 13
Sviluppo pagina principale	20,25 days	Tue 24/09/19	Fri 22/11/19 14
Sviluppo pagina ordinazioni	20,25 days	Tue 24/09/19	Fri 22/11/19 14
Sviluppo pagina fattorini	20,25 days	Tue 24/09/19	Fri 22/11/19 14
Sviluppo pagina consegne	20,25 days	Tue 24/09/19	Fri 22/11/19 14
Sviluppo pagina clienti	20,25 days	Tue 24/09/19	Fri 22/11/19 14
Creazione grafica pagina principale	12 days	Fri 18/10/19	Fri 22/11/19 8
Creazione grafica pagina ordinazioni	12 days	Fri 18/10/19	Fri 22/11/19 8
Creazione grafica pagina fattorini	12 days	Fri 18/10/19	Fri 22/11/19 8
Creazione grafica pagina consegne	12 days	Fri 18/10/19	Fri 22/11/19 8
Creazione grafica pagina clienti	12 days	Fri 18/10/19	Fri 22/11/19 8
-Test	33,75 days	Tue 10/09/19	Fri 20/12/19
Test funzionamento corretto	0,75 days	Thu 19/09/19	Thu 19/09/19 10



Inoltre ho anche finito la progettazione del DataBase semplificandola il più possibile e mi sono informato sui software di debug, soprattutto su Xdebug.

Problemi riscontrati e soluzioni adottate
Non ero presente in classe ma ho lavorato da remoto

Punto della situazione rispetto alla pianificazione
Sono in tempo rispetto alla pianificazione

Programma di massima per la prossima giornata di lavoro
Eventualmente riguardare il diagramma dopo aver riparlato con il cliente su certe domande ancora in sospeso.
Fare uno schizzo per le varie interfacce del futuro sito.

Diario di lavoro

Progetto	PizzaDelivery
Data	10.09.19

Lavori svolti

Oggi ho concluso per la maggior parte tutto quello che riguarda l'implementazione di questo progetto, infatti ho creato i vari design delle interfacce come ad esempio il seguente:

FATTORINI
ONCLICK

The wireframe illustrates a user interface for a pizza delivery application. It features a header bar with the application name "PizzaDelivery" and links for "Consegne", "Ordinazioni", "Fattorini", and "<Clienti>". A sidebar on the left contains personal information for a delivery person: "Nome: Paolo", "Cognome: Neri", and "Stato: Libero". The main content area displays a placeholder for "Posizione attuale" with a "Google maps map" link. Below this, a section titled "Consegne effettuate oggi [11/03/2019]" lists delivery details in a table format. The table has columns for "ID Consegna", "Orario", "Via", and "Incasso". One row is populated with values: "345", "19:02:29", "Via Pedemonte 3", and "54.20.-". There are also three empty rows for additional items and a placeholder for more data.

FATTORINO #ID

Nome: Paolo

Cognome: Neri

Stato: **Libero**

Posizione attuale

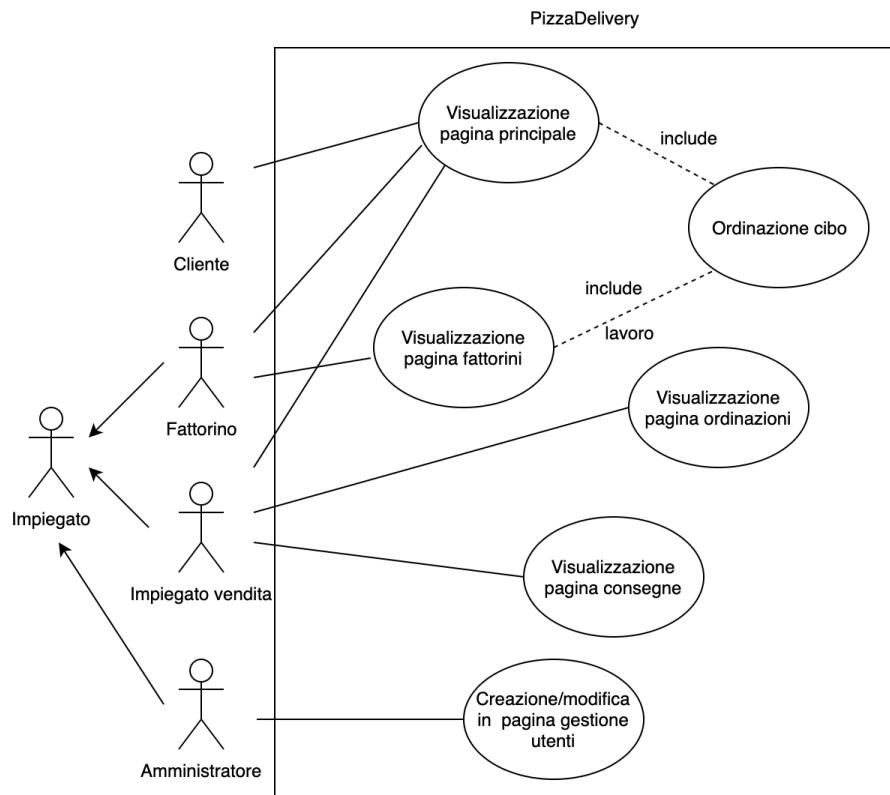
Google maps map

Consegne effettuate oggi [11/03/2019]

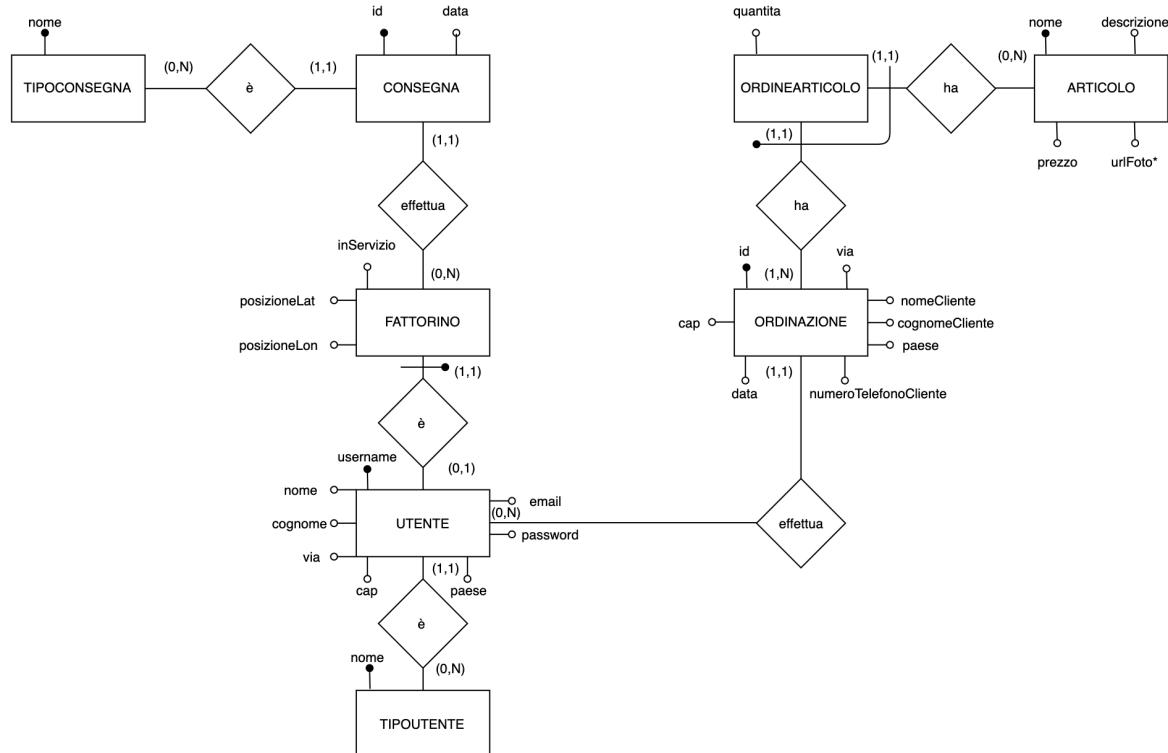
ID Consegna	Orario	Via	Incasso
345	19:02:29	Via Pedemonte 3	54.20.-
<list item>			
<list item>			
...			

<FOOTER>

Ho anche creato lo use case di questa applicazione che è il seguente:



Infine ho messo a posto il diagramma ER del Database e ho iniziato in parte con lo sviluppo del codice SQL:



Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema, i dubbi come ad esempio quali fossero gli attori principali del programma li ho discussi con il cliente.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Concludere lo sviluppo del codice del DB ed eventualmente se riesco già farci alcuni test.

Assicurarmi di aver fatto tutti i passaggi dell'implementazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	12.09.19

Lavori svolti

Oggi ho finito con l'implementazione del codice SQL per la creazione del DataBase e ho anche creato una classe di Test che popola tutte le tabelle simulando un set di dati con il quale ho eseguito i test e provato la correttezza dei vari tipi di dato immessi nei campi.

```
/* CREAZIONE UTENTE */
INSERT INTO Utente VALUES (
    "jari.naeser",
    "jari",
    "naeser",
    "Via Mer Zarei 12",
    6965,
    "Svizzera",
    "jari.naeser@samtrevano.ch",
    "test",
    "amministratore"
);
```

Oltre a ciò ho anche corretto un piccolo errore di cardinalità che ho trovato nel diagramma ER e ho continuato con lo sviluppo della parte di analisi della documentazione.

Problemi riscontrati e soluzioni adottate

Ho riscontrato un piccolo errore per via di una svista, ogni volta che inserivo un numero di telefono di tipo intero nel database questo mi dava un warning per un eccesso di dimensione; Dopo aver riflettuto per un attimo mi sono accorto di aver usato un INT al posto di un BIGINT risolvendo così il problema.

Punto della situazione rispetto alla pianificazione

Sono leggermente in avanti rispetto alla pianificazione, questo piccolo vantaggio però lo sfrutto per continuare il lavoro sulla documentazione.

Programma di massima per la prossima giornata di lavoro

Continuare con la parte della progettazione sulla documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	13.09.19

Lavori svolti

Oggi ho continuato con lo sviluppo della documentazione nella parte di analisi, mi mancano ancora pochi capitoli e poi è in grandi linee fatta per quanto riguarda la progettazione.

Problemi riscontrati e soluzioni adottate

Non ho riscontrato nessun problema.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Finire la parte di progettazione nella documentazione

Iniziare con lo sviluppo MVC

Diario di lavoro

Progetto	PizzaDelivery
Data	17.09.19

Lavori svolti

Oggi ho iniziato ad implementare il modello base dell'MVC per l'applicazione, ho anche creato la classe che si connette al DataBase la quale funziona.

Header

Utenti

```
Array
(
    [0] => Array
        (
            [username] => jari.naeser
            [nome] => jari
            [cognome] => naeser
            [via] => Via Mer Zarei 12
            [cap] => 6965
            [paese] => Svizzera
            [email] => jari.naeser@samtrevano.ch
            [password] => test
            [tipoUtente] => amministratore
        )

    [1] => Array
        (
            [username] => paolo.naeser
            [nome] => paolo
            [cognome] => naeser
            [via] => Via Mer Zarei 12
            [cap] => 6965
            [paese] => Svizzera
            [email] => paolo.naeser@samtrevano.ch
            [password] => test
            [tipoUtente] => fattorino
        )
)
```

Footer

Inoltre ho anche iniziato a sviluppare la barra di navigazione ed il footer.

Problemi riscontrati e soluzioni adottate

Ho avuto vari problemi con il toggle della barra di navigazione e con l'allineamento di un gruppo di elementi, entrambi ancora da mettere a posto.

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto alla pianificazione ma avendo riscontrato questo problema posso usare le ore per metterlo a posto e trovare una buona soluzione.

Programma di massima per la prossima giornata di lavoro

Mettere a posto gli errori mensionati e aggiornare il diagramma di gantt consuntivo.

Diario di lavoro

Progetto	PizzaDelivery
Data	19.09.19

Lavori svolti

Oggi ho messo a posto il problema della navbar il quale centrava con il fatto che all'interno della struttura MVC chiudevo sempre ogni file (header, contenuto e footer) con il tag </body></html> che successivamente riapriavo; Questo non andava d'accordo con il toggle della navbar non facendolo funzionare.

Inoltre ho anche messo a posto il footer e iniziato a creare le pagine di benvenuto e ordina aggiustando anche il routing (controller home).

Problemi riscontrati e soluzioni adottate

Il problema della navbar l'ho risolto con l'aiuto del docente Massimo Sartori, per il resto non ho incontrato grandi problemi.

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto alla pianificazione.

Programma di massima per la prossima giornata di lavoro

Aggiornare il gantt consuntivo.

Continuare con lo sviluppo delle pagine benvenuto e ordina.

Diario di lavoro

Progetto	PizzaDelivery
Data	20.09.19

Lavori svolti

Oggi ho principalmente continuato a lavorare sulla pagina di ordinazione dalla quale l'utente può selezionare i vari tipi di pizza da poi successivamente ordinarli.

Ho anche creato delle pagine per la gestione degli errori.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Aggiornare il gantt consuntivo.

Mettere messaggio che se database ad esempio non contiene prodotti non esca una tabella vuota.

Diario di lavoro

Progetto	PizzaDelivery
Data	24.09.19

Lavori svolti

Oggi ho continuato con lo sviluppo della pagina di ordinazione della quale tra non troppo dovrebbe essere presente la prima versione funzionante.

Inoltre mi sono anche occupato della pagina di login e ho creato tutti i diversi header che serviranno a visualizzare i diversi tab (pagine) a dipendenza della propria tipologia di utente che si ha.

Ovviamente avendo fatto questo ho implementato i diversi ruoli.

Ecco un esempio di come viene visto l'header dall'admin:



E di come viene visto da un fattorino:



Problemi riscontrati e soluzioni adottate

Ho riscontrato un paio di problemi di design, quindi con bootstrap che non sono riuscito ad allineare certi elementi come volevo io ma di questo me ne occuperò poi più tardi quando si tratterà di sviluppare per bene la grafica delle varie pagine.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare se non finire la pagina ordina.

Diario di lavoro

Progetto	PizzaDelivery
Data	26.09.19

Lavori svolti

Oggi ho continuato con lo sviluppo della pagina di ordinazione la quale ho deciso di suddividere in due parti distinte aggiungendo la seconda pagina "confermaOrdine.php" nella quale si inseriranno tutti i propri dati personali e le varie quantità delle pizze selezionate che si vuole ordinare.

Inoltre ho anche modificato la struttura della tabella "articolo" sul DB nella quale ho cambiato la chiave primaria dal nome del prodotto ad un id visto che dava errori quando cercavo di usare pizze che contenevano uno spazio nel nome.

Infine ho iniziato a ripulire un po' tutto il codice della parte dell'ordinazione.

Problemi riscontrati e soluzioni adottate

Riscontrato il problema della chiave primaria contenente spazi come mensionato sopra, inoltre ho anche avuto un po' di errori di struttura i quali però ho messo a posto riguardando il codice.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Finire pagina ordina.

Spostare metodo execQuery da controller a model.

Diario di lavoro

Progetto	PizzaDelivery
Data	27.09.19

Lavori svolti

Oggi ho finito con lo sviluppo della prima versione della pagina ordina nella quale però devo ancora implementare una piccola funzione nel bottone “Proseguì con l’ordinazione” che non fa andare avanti l’utente se non ha selezionato nessun articolo.

Inoltre ho anche lavorato su confermaOrdine.php dove sono circa a un terzo dello sviluppo.

Infine ho anche adattato e modificato la gestione degli errori che venivano sollevati quando c’era un problema con la connessione del DB oppure l’esecuzione delle query spostando anche completamente il metodo execQuery dal controller al model.

Problemi riscontrati e soluzioni adottate

Metodo \$.post di ajax (jQuery) non funziona correttamente, devo ancora trovare il motivo.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementare funzione del bottone nella pagina ordina.

Continuare con confermaOrdine.php

Diario di lavoro

Progetto	PizzaDelivery
Data	01.10.19

Lavori svolti

Oggi ho finito lo sviluppo della prima versione delle pagine di ordinazione e conferma dell'ordine.
Ho anche sviluppato tutto quello che riguarda la parte user-friendly aggiungendo messaggi di errore se necessari e aiuti visivi per l'utente.

PizzaDelivery≡

Informazioni personali

Jari	Näser	
Svizzera	0796267382	
SELECT * FROM utenti;	6963	12

Prodotti selezionati

Immagine	Articolo	Descrizione	Quantità	Prezzo
	Pizza margherita	Molto Buona e grande	<input type="text" value="1"/>	12
	Calzone	Molto Buona	<input type="text" value="1"/>	15

Costo Totale: 27.-

© 2019 Copyright: Jari Näser

Oltre a ciò ho anche gestito vari errori riguardanti il database e la sua iterazione con l'applicazione creando messaggi di errore anche per lui:

PizzaDelivery

C'è stato un problema
nell'esecuzione di una query.

[Torna alla home](#)

© 2019 Copyright: Jari Näser

Infine ho anche modificato leggermente la struttura del DB facendo che mette automaticamente l'orario di un'ordinazione appena questo record finisce nella tabella ordinazione.

Problemi riscontrati e soluzioni adottate

Problema incontrato con la query che immette l'ordinazione di una persona nella tabella 'ordinazione', non riuscito a risolvere per via della mancanza di tempo; lo farò la prossima lezione.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare a sviluppare le query dopo che l'utente ha effettuato l'utente per popolare nel modo corretto le varie tabelle del DB.

Aggiungere correttamente il file CSS a tutti i header.

Diario di lavoro

Progetto	PizzaDelivery
Data	03.10.19

Lavori svolti

Oggi ho finito la parte back-end dell'inserimento delle query all'ordinazione dei vari articoli ove scrivo nelle tabelle "Ordinazione" e "OrdinazioneArticolo".

Inoltre ho anche creato la pagina di overview di tutte le ordinazioni che la ditta ha e sto lavorando sulla pagina "Ordine" che si apre al click di un'elemento della tabella degli ordini e contiene tutti i dati relativi a quell'ordinazione.

Infine ho messo a posto la problematica dell'import del file css che veniva sempre importato come file di testo, questo comportamento era dovuto al posizionamento sbagliato nella struttura delle cartelle del file.

Per poter effettuare il click su una riga di una tabella ho preso spunto dalla seguente pagina sul quale viene spiegato come fare: <https://stackoverflow.com/questions/17147821/how-to-make-a-whole-row-in-a-table-clickable-as-a-link>

Problemi riscontrati e soluzioni adottate

Ho un problema con l'import del file css quando passo dei valori come parametro al controller.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Mettere a posto il problema dell'import del file CSS.

Continuare con lo sviluppo della pagina "Ordinazione".

Diario di lavoro

Progetto	PizzaDelivery
Data	04.10.19

Lavori svolti

Ho risolto il problema dell'import del file CSS mettendo tutta la path assoluta invece di una relative nei vari header dove veniva importato.

Inoltre ho finito lo sviluppo della prima versione delle pagine "Ordinazioni" e "Ordinazione" le quali si occupano di mostrare agli impiegati dell'azienda gli ordini in entrata dei vari client con tutte le rispettive informazioni; Essi verranno poi mandate alla pagina di consegne dove verranno elaborate ed assegnati ad un fattorino.

Infine ho incominciato con lo sviluppo della pagina di controllo dell'amministratore dalla quale si potranno principalmente getire gli utenti e articoli della pizzeria.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della pagina di gestione dell'admin.

Diario di lavoro

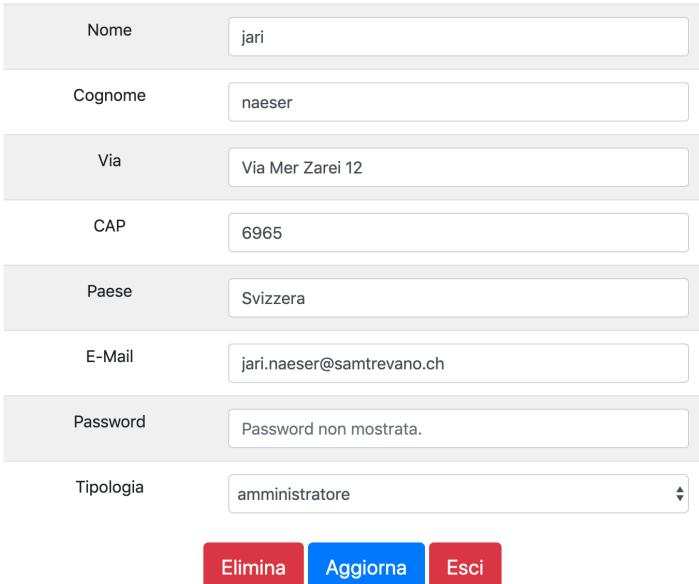
Progetto	PizzaDelivery
Data	08.10.19

Lavori svolti

Oggi ho sviluppato la prima versione funzionante della gestione degli utenti (per l'admin) nella quale ho anche già implementato le funzionalità di aggiunta e modifica utente; Mi manca ancora la parte per i prodotti.

Esempio interfaccia modifica:

Utente jari.naeser



Inoltre ho anche modificato e pulito tutta la struttura MVC aggiungendo un controller per ogni pagina e mettendo a posto tutti i link e i redirect all'interno dell'applicazione che mi ha preso parecchio tempo.

Infine ho anche creato una cartella per ogni pagina con le quali ho poi ordinato le varie view.

Problemi riscontrati e soluzioni adottate
-

Punto della situazione rispetto alla pianificazione
Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della pagina di gestione dell'admin.

Vedere se è necessario aggiungere il campo numero di telefono negli utenti registrati nel DB.

Scoprire come fare per le API di Google Maps.

Diario di lavoro

Progetto	PizzaDelivery
Data	10.10.19

Lavori svolti

Oggi ho finito la prima versione della pagina di gestione per l'amministratore dalla quale si possono visualizzare, aggiungere, modificare ed eliminare prodotti ed utenti della pizzeria.

Gestisci Utenti

Username	Tipologia	Modifica
gennaro.verdi	impiegato vendita	
jari.naeser	amministratore	
matteo.forni	impiegato vendita	
paolo.naeser	fattorino	
paolo.neri	fattorino	
test.tester	impiegato vendita	

Gestisci Articoli

Immagine	Nome	Prezzo	Modifica
	Calzone	15	
	Focaccia	6	
	Pizza ai funghi	14	
	Pizza alla marinara	10	
	Pizza margherita	12	
	Pizza prosciutto	14	

Inoltre ho anche modificato il valore di default del campo urlFoto nella tabella Articolo la quale adesso utilizza una vera immagine di default se questa non viene immessa, fatto con la seguente query:

```
ALTER TABLE Articolo ALTER urlFoto SET DEFAULT "application/img/defaultPizza.png";
```

Infine ho creato la classe Validator che si occupa di controllare tutti i dati e stringhe in input per evitare SQLInjection e attacchi di simile tipologia.

Problemi riscontrati e soluzioni adottate

Guardando per caso su w3schools ho trovato un modo che mi ha fatto risparmiare molto tempo nella ricerca di utenti ed articoli attraverso un form (lavoro fatto da jQuery):

https://www.w3schools.com/bootstrap4/bootstrap_filters.asp

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Iniziare con lo sviluppo della pagina di gestione dei fattorini.

Diario di lavoro

Progetto	PizzaDelivery
Data	11.10.19

Lavori svolti

Oggi le prime due ore della lezione sono state occupate dalla presentazione di Valsangiacomo che parlava dei criteri di valutazione del progetto di conclusione finale.

Nelle ultime due invece ho messo un po' a posto il codice facendo pulizia e rimuovendo parti inutili oltre a spostare anche certi metodi dalle classi controller al model (quelle che lavoravano soprattutto con i dati).

Infine ho creato in parte la pagina di overview “fattorini” dalla quale è possibile visualizzare ogni singolo fattorino presente nel sistema e le sue rispettive informazioni.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Aggiunta del trigger che alla creazione di un utente di tipologia fattorino lo aggiunga direttamente nella tabella fattorino con i rispettivi valori di default.

Continuare con lo sviluppo delle pagine fattorini e fattorino.

Diario di lavoro

Progetto	PizzaDelivery
Data	15.10.19

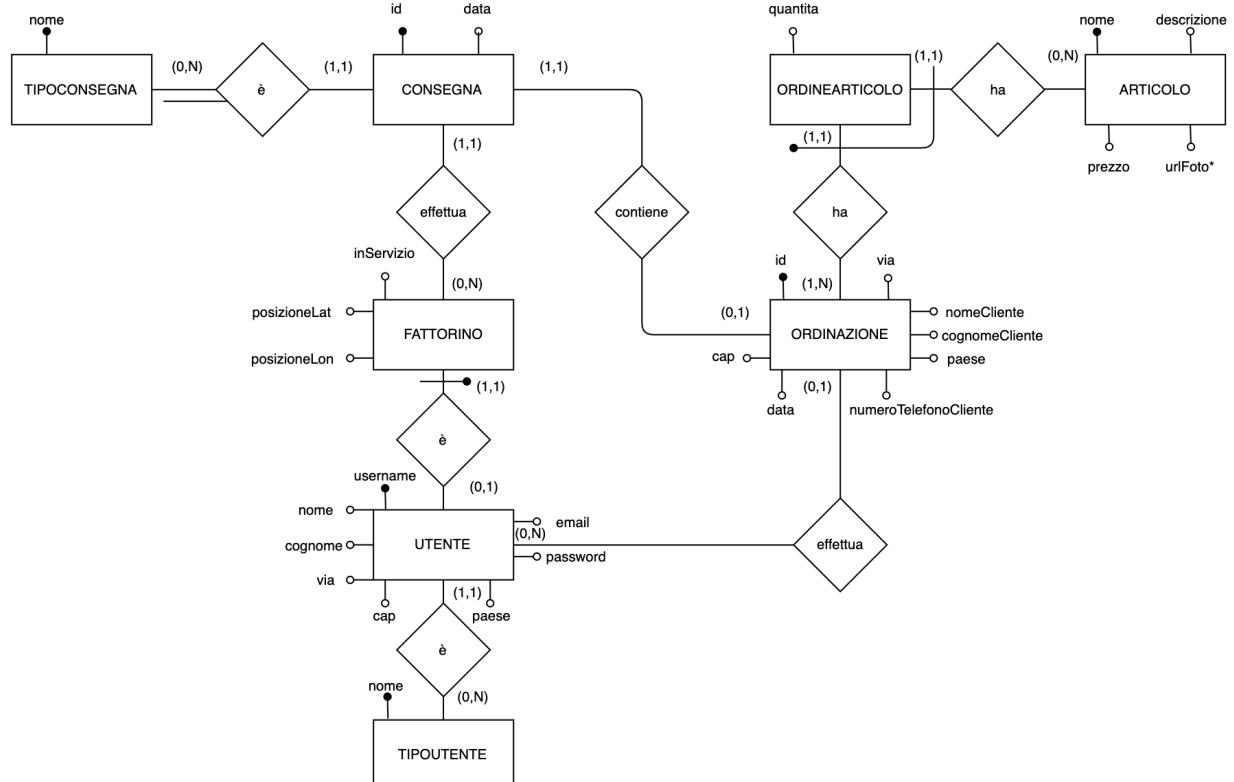
Lavori svolti

Oggi mi sono principalmente occupato di lavorare con il DataBase creando un trigger che aggiunge un nuovo record nella tabella fattorino alla creazione di un utente di tipologia fattorino:

```
DELIMITER //
CREATE TRIGGER addFattorino AFTER INSERT ON Utente FOR EACH ROW
BEGIN
    INSERT INTO fattorino VALUES (NEW.username, 0, 0, false);
END
//DELIMITER ;
```

Inoltre ho anche continuato lo sviluppo delle pagine fattorini e fattorino nelle quali mi manca ancora una query complessa per estrarre il prezzo totale delle varie consegne effettuate.

Come ultima cosa mi sono accorto della mancanza di un attributo nella tabella consegna, cioè l'identificativo dell'ordinazione che è stata assegnata da consegnare.



Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Creazione query complicata per il prezzo totale della consegna.
Finire se possibile le pagine fattorini e fattorino.

Diario di lavoro

Progetto	PizzaDelivery
Data	17.10.19

Lavori svolti

Oggi ho concluso lo sviluppo della prima versione delle pagine Fattorini e Fattorino. Quando si entra sulla pagina di un fattorino l'interfaccia è la seguente:

Nome: paolo

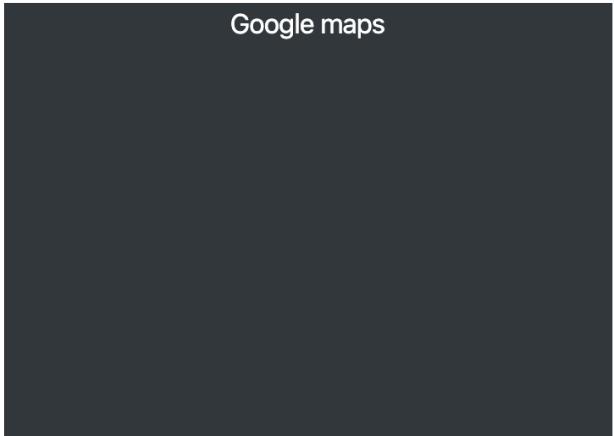
Cognome: naeser

E-mail: paolo.naeser@samtrevano.ch

Via: Via Mer Zarei 12

Stato: In Servizio

Google maps



Consegne [17.10.2019]

ID Consegna	Orario	Via	Tipologia	Incasso
7	2019-10-17 13:35:56	Via Italia 134C	Terminata	54
12	2019-10-17 14:20:50	via focaccia 23	In Corso	-

[Torna ai fattorini](#)

Inoltre ho anche sviluppato la famosa query “complicata” della scorsa lezione il quale codice SQL è il seguente:

```
SELECT SUM((SELECT prezzo FROM Articolo WHERE id = articolo) * quantita) AS 'SommaCosti'  
FROM OrdineArticolo o WHERE ordinazione = (  
    SELECT ordinazione FROM Consegna c, Fattorino f  
    WHERE c.fattorino LIKE f.username AND tipoConsegna LIKE 'terminata' AND id = " . $consegne[$i]['id'] . "  
);
```

Infine ho anche iniziato con lo sviluppo della pagina Gestione.

Problemi riscontrati e soluzioni adottate

Ho riscontrato il problema di non poter inserire caratteri come il < e > nelle query essendo che prima di eseguirle le filtravo ancora con il htmlspecialchars eccetera che mi trasformavano gli operatori in < e > rompendo così la query.

Ho sviato questo problema creando un metodo privato nel model il quale non filtra la query ricevuta (più insicuro ma non pericoloso visto che il metodo è solamente accessibile dal model).

Punto della situazione rispetto alla pianificazione

Sono in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Mettere a posto il contenuto della select che viene sempre impostato a “Tutte” al caricamento della pagina consegne.

Far in modo che l’ora venga solamente salvata quando una consegna è terminata, modificare DEFAULT sul db e creare metodo setter setTerminato per farla impostare automaticamente dall’esterno.

Diario di lavoro

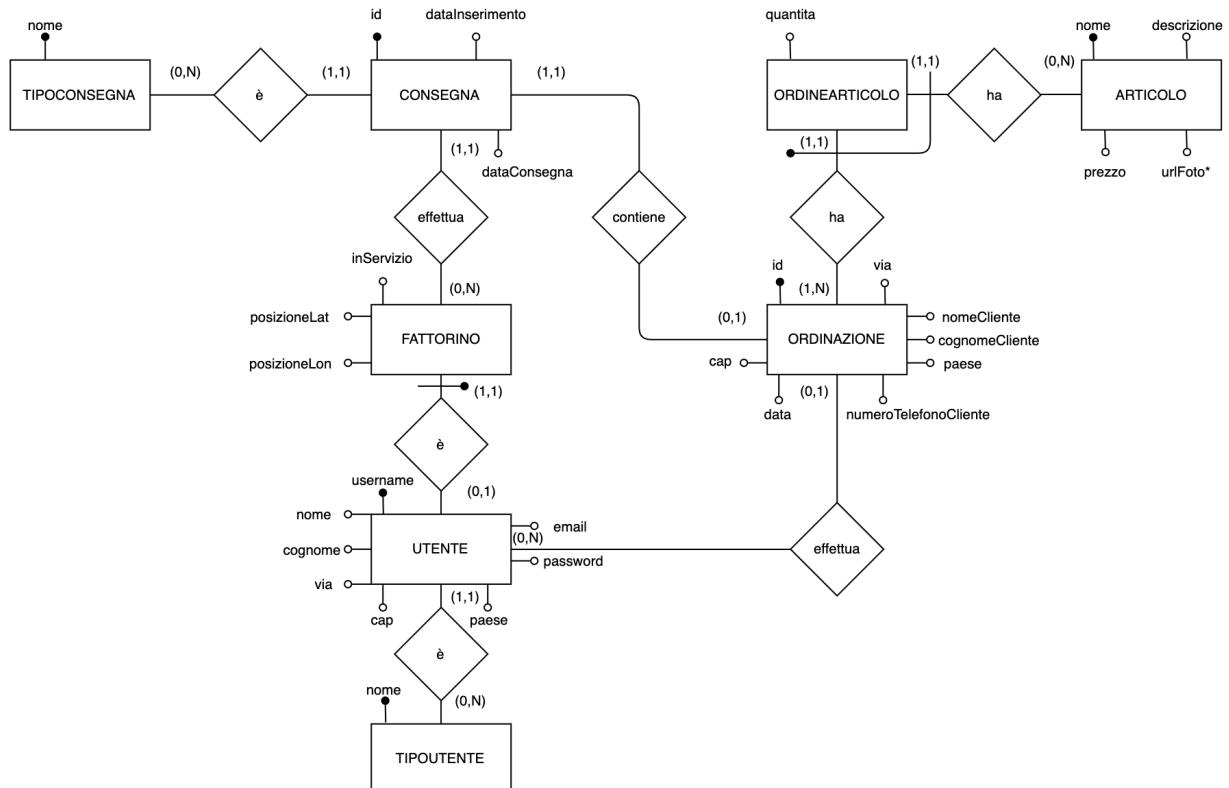
Progetto	PizzaDelivery
Data	18.10.19

Lavori svolti

Oggi ho concluso lo sviluppo della pagina Consegne e ho messo a posto l'errore del dropdown che non cambiava il valore selezionato correttamente alla ricarica della pagina e ho anche messo a posto la questione degli orari aggiungendone un secondo che si occupa di memorizzare l'orario di quando una consegna è effettivamente stata consegnata da un fattorino al cliente e viene quindi marcata come "Terminata".

Avendo fatto ciò ho anche creato i metodi helper setConsegnaInCorso e setConsegnaTerminata che verranno richiamati una volta che cambia lo stato della tipologia di consegna.

Nuovo schema del DB:



Infine ho anche messo un po' a posto tutte le tabelle che stampo con php rendendole più leggibili dal lato codice.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

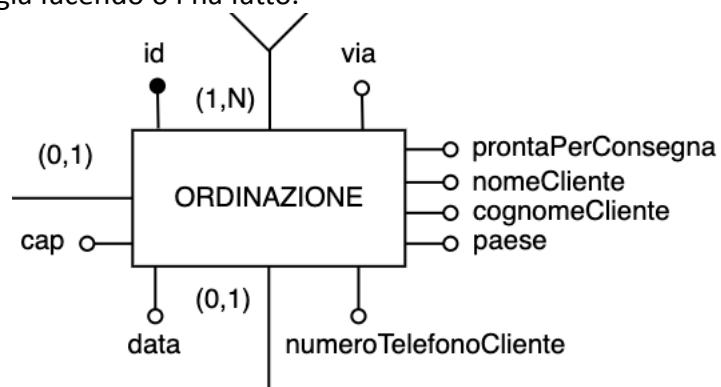
Discutere se una volta selezionato un fattorino per un'ordinazione essa viene tolta dalle ordinazioni e messa nelle consegne (implementare l'assegnazione con modal (ricava fattorini liberi se possibile)).
Mettere a posto errore che quando uso il metodo setConsegnaTerminata mi da "general error".

Diario di lavoro

Progetto	PizzaDelivery
Data	22.10.19

Lavori svolti

Oggi ho finito con la prima versione della pagina delle ordinazioni e quindi ho concluso in parte questo primo capitolo dell'implementazione avendo una beta del software funzionante, nelle prossime lezioni si tratterà di approfondire ed implementare i dettagli richiesti e necessari. Ho anche modificato il DB aggiungendo il campo booleano `prontaPerConsegna` alla tabella `Consegna` che definisce il suo stato, quindi se è già pronta e aspetta che il fattorino la consegni oppure esso lo stia già facendo o l'ha fatto.



Inoltre ho anche implementato un modale nella pagina delle ordinazioni con il quale si può assegnare un'ordinazione ad un fattorino ed infine ho anche messo a posto e raffinato i metodi con i quali è possibile cambiare lo stato delle consegne.

Assegna Ordinazione [51] ad un Fattorino

Selezione	Nome	Stato
<input type="radio"/>	franco.fattorino	Libero
<input type="radio"/>	luca.grandi	Libero
<input type="radio"/>	paolino.forni	Libero
<input type="radio"/>	pao.lo.naeser	In Servizio

Esci **Assegna**

Problemi riscontrati e soluzioni adottate

Al click del bottone “assegna a fattorino” nella pagina “ordinazioni” scattava l’evento di click della riga <tr> (genitore) nella quale si trovava, per sviare questo problema e far scattare l’evento di click del bottone invece di quello del <tr> ho implementato il seguente codice che ho trovato su stackoverflow:
<https://stackoverflow.com/questions/13589022/can-i-exclude-a-button-click-inside-a-tr-click-event/13589117>

```
$('.assegnaAFattorino').click(function(e){  
    e.preventDefault();  
    e.stopPropagation();  
    console.log("Hello");  
});
```

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Rileggere tutto il QDC per assicurarmi di aver messo tutte le funzionalità nella versione Beta richieste.
Eliminare tutto il DB e ricostruirlo simulando una ditta che acquista il prodotto e si mette ad usarlo.
Vedere come gestire lo stato di un fattorino, quando riceve una consegna diventa occupato?
Iniziare ad informarmi su come implementare le mappe open source.

Diario di lavoro

Progetto	PizzaDelivery
Data	24.10.19

Lavori svolti

Oggi ho inizialmente riletto il QDC confermando che ho implementato le richieste e ho anche ricreato il DB con i suoi dati per essere sicuro di usare l'ultima versione dell'SQL contenente tutte le nuove modifiche (del DB).

Dopo aver fatto ciò ho anche messo a posto un paio di piccolezze riguardanti il css ed il funzionamento delle pagine di errore e ho creato un trigger che si occupa di aggiornare la tabella fattorino quando cambia la tipologia di un'utente.

Successivamente ho implementato i due metodi setFattorinoLibero(username) e setFattorinoOccupato(username) che cambiano lo stato del fattorino passato attraverso il suo username ai metodi e quindi all'assegnazione di un'ordinazione esso risulterà occupato.

Ho anche aggiunto la cifratura alle password con l'algoritmo sha256 per portare maggiore sicurezza al sistema e non mostrare in chiaro le password dei vari utenti nel database.

Infine ho iniziato con lo sviluppo dello spinner di caricamento che scompare una volta che la pagina è completamente caricata (devo ancora decidere se utilizzarlo o meno).

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Finire il sistema dello spinner al caricamento delle varie pagine.

Trovare una soluzione per le mappe.

Popup all'eliminazione di un prodotto e/o utente con "Sei sicuro?".

Diario di lavoro

Progetto	PizzaDelivery
Data	25.10.19

Lavori svolti

Oggi ho iniziato continuando lo sviluppo dello spinner di caricamento con il quale ho avuto dei problemini (con la parte javascript) e non essendo ancora sicuro se effettivamente lo userò o meno l'ho lasciato da parte per il momento.

Successivamente ho implementato i modali che richiedono la conferma all'eliminazione di un'utente/prodotto dalla dashboard di gestione della pizzeria.

Infine ho anche creato i controlli in tutte le pagine che siano impostate correttamente tutte le sessioni e in caso contrario ogni pagina stampa un messaggio di errore del formato "sessione xy non trovata."

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Trovare una soluzione per le API di google maps.

Diario di lavoro

Progetto	PizzaDelivery
Data	05.11.19

Lavori svolti

Oggi non ero presente a lezione ma ho lavorato da casa e ho iniziato ad implementare le varie mappe con le loro relative informazioni nelle pagine fattorino ed ordinazione.
Questo è stato possibile attraverso il servizio di Mapbox che sotto le 50'000 richieste offre le sue mappe gratuitamente.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono leggermente in anticipo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Finire l'implementazione delle mappe.

Diario di lavoro

Progetto	PizzaDelivery
Data	07.11.19

Lavori svolti

Oggi ho finito l'implementazione delle mappe nelle pagine fattorino ed ordinazione con le quali si possono visualizzare le posizioni attuali di un fattorino e di un cliente (posto in cui consegnare un'ordinazione).

Oltre a ciò ho continuato con lo sviluppo della documentazione lavorando soprattutto sul capitolo dell'implementazione e altri piccoli dettagli.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Mettere a posto i metodi del Model, fare tutti i metodi get.

Controllare che sia completa l'installazione del debugger e creare una piccola documentazione guida.

Diario di lavoro

Progetto	PizzaDelivery
Data	08.11.19

Lavori svolti

Oggi ho dedicato tutta la lezione a riscrivere la classe PizzaDeliveryModel rendendola molto più leggibile, semplice per l'uso e sicura visto che ho implementato il metodo bindParam() ad ogni parametro che inserisco in una query.
Questo lavoro rende anche più pulito e corto il lavoro nelle altre classi che richiamano metodi di essa di ogni tipo.

Problemi riscontrati e soluzioni adottate

Problema con l'update di un record non possibile visto che c'era un'errore in un trigger che veniva scattato alla sua modifica, dopo svariate ricerche ho finalmente trovato il problema visto che non mi ricordavo di aver usato questo trigger.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Riscrivere i metodi restanti del model.
Suddividere il model in vari model seguendo in grandi linee le tabelle del DB.
Controllare che sia completa l'installazione del debugger e creare una piccola documentazione guida.

Diario di lavoro

Progetto	PizzaDelivery
Data	12.11.19

Lavori svolti

Oggi ho concluso la suddivisione delle classi model e la loro riscrittura, successivamente ho speso una mezz'oretta a ritestare ogni singola cosa di modo da evitare futuri problemi dovuti a questa migrazione.

Successivamente ho implementato un trigger che mi facilita l'eliminazione di un'utente eliminando anche il suo record nella tabella fattorino se esso è di questa tipologia.

```
/* Trigger che toglie un'utente dai fattorini alla sua eliminazione se  
esso è di tipo fattorino */  
DELIMITER //  
CREATE TRIGGER removeFattorino BEFORE DELETE ON Utente FOR EACH ROW  
BEGIN  
    IF OLD.tipoUtente LIKE 'fattorino' THEN  
        DELETE FROM fattorino WHERE username = OLD.username;  
    END IF;  
END  
//DELIMITER ;
```

Infine ho creato un documento che serve a valutare i vari aspetti dell'applicazione e dei suoi funzionamenti per la demo che farò con il docente Ivan Raimondi il prossimo martedì.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Controllare funzionamento debugger

Evitare l'eliminazione dell'ultimo admin

Cosa fare se un'utente ha lo stesso nome e cognome di un'altro

Controlli input creazione/modifica dalla pagina dell'admin

Capire quali caratteri non sono supportati dal db.

Continuare con la documentazione

Diario di lavoro

Progetto	PizzaDelivery
Data	14.11.19

Lavori svolti

Oggi ho implementato la funzionalità che non si può eliminare l'ultimo admin della ditta attraverso l'interfaccia grafica, se questa operazione la si vuole forzare bisogna farlo attraverso codice SQL (in una scena reale avrebbe pochissimo senso gestire l'applicativo senza amministratore).

Inoltre ho anche fatto in modo che se si vuole registrare un'impiegato che ha lo stesso nome di uno già esistente il sistema crea il suo username attaccandoci un numero ad esempio:

Primo account: marco.verdi

Secondo account: marco.verdi2

Successivamente ho messo a posto il fatto che non funzionavano correttamente i rindirizzamenti degli URL quando si usavano utenti con caratteri “speciali” come la ä nell’username (vedi soluzione in problemi riscontrati).

Infine ho anche messo a posto un’errore nell’aggiornamento dei dati di un’utente in quanto la problematica aveva a che fare con gli username che contenevano numeri.

Problemi riscontrati e soluzioni adottate

Il punto dove si generava l’errore degli url era una parte nel file application che gestisce tutto ciò. Infatti c’era il metodo filter_var che toglieva i caratteri speciali dalle stringhe come ad esempio jari.näser diventava jari.nser portando così incongruenze in tutto il programma.

```
//ATTENZIONE: Ho aggiunto urlencode e urldecode per far sì  
che metta le percentuali prima dei caratteri e  
// poi li ritolga, con questo sistema anche i caratteri  
come la ä vengono passati invece che tagliati  
// fuori dal metodo filter_var  
  
$url = urlencode($url);  
  
//rimuove tutti i caratteri illegali dall'URL  
$url = filter_var($url, FILTER_SANITIZE_URL);  
  
$url = urldecode($url);
```

Ho risolto questo problema anche con l’aiuto del docente Massimo Sartori in quanto abbiamo implementato i metodi urlencode() e urldecode() prima e dopo il filter_var().

Questi metodi rimpiazzano i caratteri speciali con un codice ed una percentuale, da jari.näser a jari.n%C3%A4ser; Così il metodo filter_var lo accetta e non lo rimuove dalla stringa.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Controllare funzionamento debugger

Testare urlencode in application (con accenti eccetera)

Mettere controlli degli input nella pagina di gestione (modifica e creazione utente).

Continuare con la documentazione

Diario di lavoro

Progetto	PizzaDelivery
Data	15.11.19

Lavori svolti

Oggi ho implementato tutti i metodi che controllano da lato client l'input del testo immesso dell'amministratore nei vari campi alla creazione e modifica di un'utente o articolo.

Successivamente ho continuato con lo sviluppo della documentazione lavorando soprattutto sul capitolo dell'implementazione.

Infine ho provato ad installare xDebug senza successo visto che non riesco a vedero nella pagina di `phpinfo()`; e di conseguenza non funziona neanche.

Ho provato a capire il perchè anche se i percorsi e la sintassi sono corretti ma per la mancanza di tempo non sono riuscito a finire.

Problemi riscontrati e soluzioni adottate

L'unico problema riscontrato era quello del debugger mensionato nei lavori svolti.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Capire l'errore e mettere a posto il debugger.

Effettuare la demo del programma.

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	19.11.19

Lavori svolti

Oggi ho importato tutte le librerie come Bootstrap, jQuery, FontAwesome e popper.js scaricandole invece di utilizzarle attraverso i loro CDN, in questo modo il sito mantiene gli aspetti grafici e funzionali anche se non è connesso alla rete.

Successivamente ho avuto la Demo con il docente Ivan Raimondi nel quale ho presentato quanto svolto fino ad ora, e discutendone abbiamo stabilito cosa c'è da migliorare e cosa c'è ancora da sviluppare.

Infine ho iniziato ad implementare la visualizzazione della tipologia dell'utente una volta eseguito il login che però è ancora da concludere visto che quando si apre il sito con mobile la barra di navigazione viene schiacciata.

PizzaDelivery • impiegato vendi...

Problemi riscontrati e soluzioni adottate

L'unico problema riscontrato è quello della barra di navigazione menzionato sopra.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementazione dettagli da aggiungere/modificare visti nella demo.

Completamento installazione debugger.

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	21.11.19

Lavori svolti

Oggi ho messo a posto un piccolo errore che c'era se non si riusciva ad eseguire una connessione con il database (richiamavo un controller sbagliato). Successivamente ho implementato la possibilità di modificare lo stato di una consegna implementando un modale che gestisce questa cosa. Inoltre ho anche continuato a sviluppare la presentazione.

Problemi riscontrati e soluzioni adottate

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementazione dettagli da aggiungere/modificare visti nella demo.
Completamento installazione debugger.
Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	22.11.19

Lavori svolti

Oggi ho continuato con lo sviluppo della documentazione per le prime due ore della lezione. Sono stato assente nella terza e quarta ora, le quali recupererò martedì 26.11 dalle 8.20 alle 9.50.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementazione dettagli da aggiungere/modificare visti nella demo.
Completamento installazione debugger.
Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	26.11.19

Lavori svolti

Oggi ho messo a posto l'installazione di xDebug assieme al compagno Bryan Beffa mettendo le configurazioni corrette nel php.ini.

Successivamente ho implementato l'opzione della cancellazione di un'ordinazione se essa è sbagliata oppure un cliente chiama per annullare il proprio ordine (se non è ancora in esecuzione).

Ho anche messo a posto la questione della visualizzazione della tipologia di utente all'interno della navbar che andava ad invadere lo spazio del bottone rendendo il testo più piccolo a dipendenza della larghezza dello schermo.

Infine ho aggiunto il controllo che evita di fare effettuare un'ordine se il numero di articoli da ordinare corrisponde a 0 e se ce ne sono di più di 1 toglie gli articoli a 0; Facendo così si hanno delle comande coerenti.

Esempio:

Ordinazione composta da:

2x Pizza

0x Focaccia

Dopo il controllo:

2x Pizza

Problemi riscontrati e soluzioni adottate

Per mettere a posto il problema del debugger ho rimpiazzato le righe esistenti riguardanti xDebug con le seguenti:

```
[xdebug]  
  
;xDebug Configuration starts  
zend_extension="/Applications/MAMP/bin/php/php7.2.1/lib/php/extensions/no-debug-non-zts-20170718/xdebug.so"  
  
xdebug.remote_autostart = 1  
xdebug.profiler_append = 0  
xdebug.profiler_enable = 0  
xdebug.profiler_enable_trigger = 0  
xdebug.profiler_output_dir = "/tmp"  
xdebug.remote_enable = 1  
xdebug.remote_handler = "dbgp"  
xdebug.remote_host = "localhost"  
xdebug.remote_log = "/tmp/xdebug.log"  
xdebug.remote_port = 10000  
xdebug.trace_output_dir = "/tmp"  
;36000 = 10h  
xdebug.remote_cookie_expire_time = 36000;  
;xDebug Configuration ends
```

Adesso funziona correttamente e posso anche usarlo nell'IDE phpStorm.

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Implementazione dettagli da aggiungere/modificare visti nella demo.

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	28.11.19

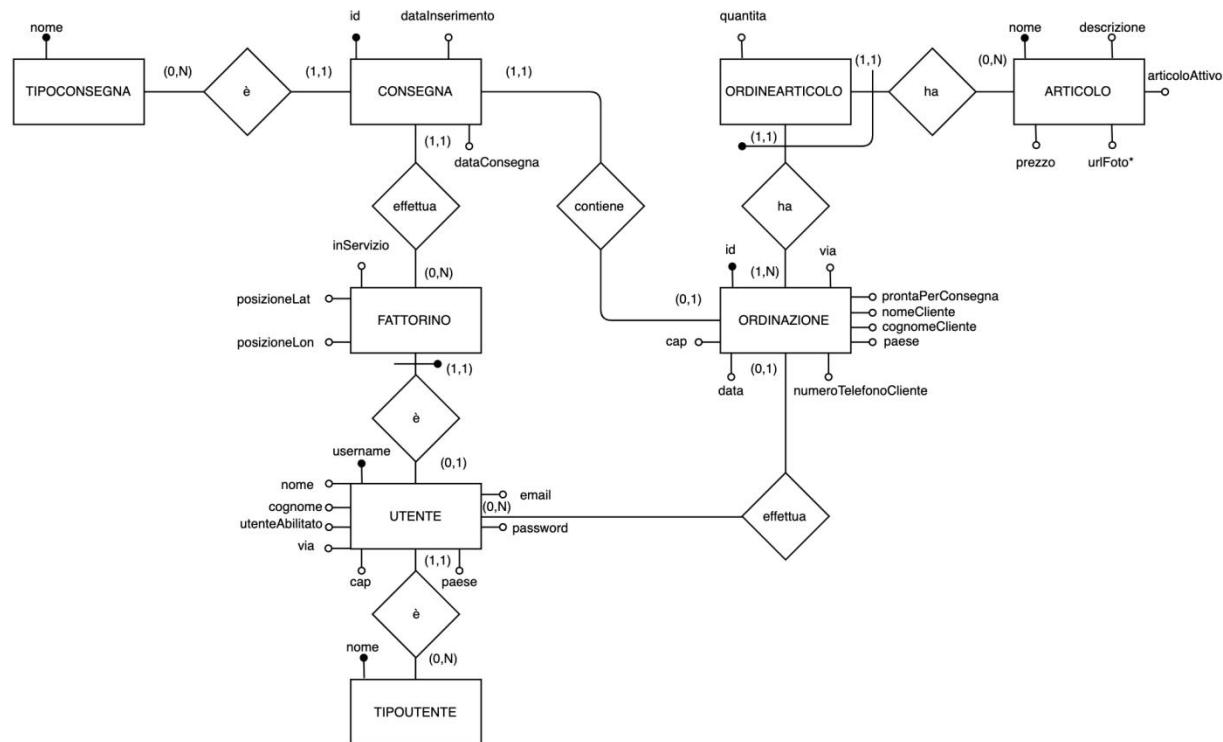
Lavori svolti

Oggi ho implementato il fatto di poter togliere ed aggiungere degli articoli dal listino senza doverli eliminare, basta attivarli e disattivarli dalla pagina Gestione Pizzeria, in “ordina” verranno solamente mostrati i prodotti attivati.

La stessa cosa anche con gli utenti, infatti ora un’account può essere disattivato invece di doverlo eliminare sviando così il problema delle foreign key nelle altre tabelle.

Inoltre si può solamente effettuare il login con account attivati.

Infine ho aggiornato lo schema del database aggiungendo i due nuovi campi utenteAbilitato e articoloAttivo.



Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Fare in modo che vengano solamente mostrati gli utenti attivi nelle varie pagine (login incluso)
Modificare apertura delle pagine di default (Esempio: al login con un fattorino va direttamente sulla pagina fattorini).
Eventualmente implementare una lista dei paesi facendo diventare select.
Riprovarle i vari e la loro validazione input.
Trovare soluzione per hosting del progetto (con supporto per DB).
Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	29.11.19

Lavori svolti

Oggi ho apportato le ultime “grandi” modifiche all’applicativo mostrando solamente gli utenti attivi nelle varie select e bloccando il login con gli account disabilitati.

Ho anche implementato il fatto che a dipendenza del tipo di utente con il quale si effettua il login si accede direttamente alla pagina desiderata (esempio: l’admin accede subito alla pagina “gestione pizzeria”).

Inoltre ho anche ritestato tutte le regex mettendo a posto gli errori che ho trovato.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Trovare soluzione per hosting del progetto (con supporto per DB).

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	03.12.19

Lavori svolti

Oggi ho continuato con lo sviluppo della documentazione e ho modificato i nomi delle cartelle che contengono l'MVC.

Ho anche creato una cartella database all'interno della application nella quale ho messo la classe che si connette al db e i due file sql che generano e riempiono il db.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	05.12.19

Lavori svolti

Oggi ho lavorato per 1 ora di orologio sul progetto visto che è stata prevista una gita all'università di friborgo.

Ho continuato con la documentazione e ho messo a posto delle piccolezze nel codice come il colore di button eccetera per rendere il programma più user-friendly.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con la documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	06.12.19

Lavori svolti

Oggi ho continuato con lo sviluppo della documentazione per tutta la durata del pomeriggio.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	10.12.19

Lavori svolti

All'inizio della lezione ho riscritto i controlli della tipologia di utente alla richiesta delle varie pagine assicurandomi che se si prova ad accedere ad una pagina attraverso link url senza essersi loggati questa operazione non sarà possibile.

Successivamente ho continuato con l'implementazione della documentazione.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della documentazione.

Completare il modello Abstract.

Diario di lavoro

Progetto	PizzaDelivery
Data	12.12.19

Lavori svolti

Oggi ho continuato lo sviluppo della documentazione per tutta la durata della lezione.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della documentazione.

Completare il modello Abstract.

Diario di lavoro

Progetto	PizzaDelivery
Data	13.12.19

Lavori svolti

Oggi ho continuato lo sviluppo della documentazione per tutta la durata della lezione.

Inoltre ho modificato il seguente diario “diari 2019_09_12_I4_Naeser_PizzaDelivery” dopo il consenso del docente Fabrizio Valsangiacomo modificando esclusivamente il colore di sfondo dell’immagine da nero a bianco.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della documentazione.

Completare il modello Abstract.

Diario di lavoro

Progetto	PizzaDelivery
Data	17.12.19

Lavori svolti

Oggi ho continuato con lo sviluppo della documentazione e ho terminato l'abstract.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Continuare con lo sviluppo della documentazione.

Diario di lavoro

Progetto	PizzaDelivery
Data	19.12.19

Lavori svolti

Oggi ho concluso lo sviluppo della documentazione, ho stampato e rilegato tutta la parte cartacea richiesta.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

Sono in tempo rispetto la pianificazione.

Programma di massima per la prossima giornata di lavoro

Consegnare il lavoro

Diario di lavoro

Progetto	PizzaDelivery
Data	20.12.19

Lavori svolti

Oggi ho consegnato tutto il progetto e ho mandato i documenti richiesti al perito.

Problemi riscontrati e soluzioni adottate

-

Punto della situazione rispetto alla pianificazione

-

Programma di massima per la prossima giornata di lavoro

-

Quaderno Dei Compiti

1 INFORMAZIONI GENERALI

Candidato	Nome: Jari	Cognome: Naeser
	jari.naeser@samtrevano.ch	079 122 81 42
Luogo di lavoro	Scuola Arti e Mestieri / CPT Trevano-Canobbio	
Orientamento	<input type="checkbox"/> 88601 Sviluppo di applicazioni <input checked="" type="checkbox"/> 88602 Informatica aziendale <input type="checkbox"/> 88603 Tecnica dei sistemi	
Superiore professionale	Nome: Ivan	Cognome: Raimondi
	ivan.raimondi@edu.ti.ch	
Perito 1	Nome:	Cognome:
Perito 2	Nome:	Cognome:
Periodo	3 settembre 2019 – 20 dicembre 2019 (presentazioni: 7-17 gennaio 2020)	
Orario di lavoro	Secondo orario scolastico 1° semestre	
Numero di ore	174	
Pianificazione 80h (in ore o %)	Analisi: 10% Implementazione: 50% Test: 10% Documentazione: 30%	

2 PROCEDURA

- Il candidato realizza il lavoro autonomamente sulla base del quaderno dei compiti ricevuto il 1 ° giorno.
- Il quaderno dei compiti è approvato dai periti. È anche presentato, commentato e discusso con il candidato. Con la sua firma, il candidato accetta il lavoro proposto.
- Il candidato ha conoscenza della scheda di valutazione prima di iniziare il lavoro.
- Il candidato è responsabile dei suoi dati.
- In caso di problemi gravi, il candidato o il superiore professionale avvertono immediatamente il perito.
- Il candidato ha la possibilità di chiedere aiuto, ma deve menzionarlo nella documentazione.
- Alla fine del tempo a disposizione per la realizzazione del LPI, il candidato deve inviare via e-mail il progetto al superiore professionale e al perito 1. In parallelo, una copia cartacea della documentazione dovrà essere fornita in duplice copia (superiore professionale e perito). Quest'ultima deve essere in tutto identica alla versione elettronica.

3 TITOLO

Applicativo web per la gestione online delle consegne di pizze a domicilio

4 HARDWARE E SOFTWARE DISPONIBILE

1 PC

Ambiente di sviluppo per siti in PHP

Software ...AMP (web server, database e linguaggio di programmazione PHP)

Eventuali framework o template per lo sviluppo del sito

5 PREREQUISITI

nessuno

6 DESCRIZIONE DEL PROGETTO

Realizzare un applicativo web per la gestione delle consegne di pizze a domicilio. Il programma serve a mantenere lo stato delle ordinazioni e delle consegne, tra cui: le ordinazioni (cliente, indirizzo, articoli ordinati), fattorini (stato "in servizio o no", posizione attuale, attività eseguite giornaliere), consegne (da effettuare, in corso, terminate). Il lavoro viene svolto a fasi e comprende:

1) preparazione di un ambiente di lavoro specifico con web server, PHP, editor, debugger (obiettivo minimo per la sufficienza)

2) realizzazione di una versione base del programma in PHP (obiettivi minimi per la sufficienza):

- definizione dei ruoli e delle funzioni del programma

- procedure di utilizzo

- casi di test

- schema banca dati

- interfaccia minima per gestione fattorini, clienti, ordinazioni e consegne. Assegnazione cliente/comanda/fattorino/consegna, gestione programma attuale fattorino

- struttura del programma basata su programmazione ad oggetti: classi e astrazione

3) struttura del programma

Obiettivi supplementari per ottenere una valutazione eccellente:

- visualizzazione mappa posizione cliente per consegna

- il programma deve espletare le funzioni previste e funzionare correttamente

- funzioni aggiuntive: calcolo itinerario per consegne multiple

- l'interfaccia user friendly (facile da utilizzare, graficamente apprezzabile, visibile su uno smartphone)

4) dettagli applicativo:

- attori dell'applicativo: amministratore, impiegato vendita, fattorini

- elementi: ditta, ordinazioni, clienti, consegne

7 RISULTATI FINALI

Il candidato è responsabile della consegna al superiore professionale e al perito:

- Una pianificazione iniziale (entro il primo giorno)
- Una documentazione del progetto comprendente:
 - prerequisiti tecnici
 - installazione e configurazione del sito

- descrizione della banca dati
- descrizione dei programmi/pagine
- schema funzioni per ruolo (use case)
- test e risultati
- Un diario di lavoro
- La pianificazione finale

8 PUNTI TECNICI SPECIFICI VALUTATI

La griglia di valutazione definisce i criteri generali secondo cui il lavoro del candidato sarà valutato (documentazione, diario, rispetto dei standard, qualità, ...).

Inoltre, il lavoro sarà valutato sui seguenti 7 punti specifici (punti da A14 a A20):

1. 190 - *Design della GUI (Elaborazione di una maschera/schermo/sito/Internet)*
2. 192 - *Criteri di sicurezza IT*
3. 193 - *Design del GUI*
4. 194 - *Attendibilità dei dati inseriti dall'utilizzatore*
5. 196 - *Illustrazione dello stato effettivo/obiettivo del Workflow*
6. 232 - *Programmazione web professionale*
7. 237 - *Analisi di sicurezza (Applicazione Web)*

9 FIRMA

Candidato

Canobbio, 03.09.2019

Superiore professionale

Canobbio, 03.09.2019

Perito 1

(luogo e data)

Perito 2

(luogo e data)

Prodotto

(<https://github.com/JariNaeser/PizzaDelivery>)