

Shop online per i gruppi

Scuola d'Arti e Mestieri di Trevano (SAMT)
Documentazione

Alessandro Aloise

Indice

1	Introduzione	5
1.1	Informazioni sul progetto	5
1.2	Abstract	5
1.3	Scopo	5
2	Analisi	6
2.1	Analisi del dominio	6
2.2	Analisi dei mezzi	6
2.2.1	Software	6
2.2.2	Hardware	6
2.3	Analisi e specifica dei requisiti	7
2.4	Use Case	11
2.5	Pianificazione	12
3	Progettazione	13
3.1	Design dei dati e database	13
3.2	Design delle interfacce	14
3.2.1	Login	14
3.2.2	Registrazione	15
3.2.3	Home	16
3.2.4	Gruppi	17
3.2.5	Lista gruppi	18
3.2.6	Gestione Gruppi	19
3.2.7	Aggiungi prodotto	20
3.2.8	Modifica prodotto	21
3.2.9	Fattura	22
4	Implementazione	23
4.1	Struttura delle cartelle	23
4.2	Database	23
4.2.1	migrations	23
4.2.2	Users	23
4.3	Controller	24
4.3.1	HomeController	24
4.3.1.1	Introduzione	24
4.3.2	Implementazione	24
4.3.2.1	index	24
4.3.2.2	getGroupsWithParticipations	25
4.3.2.3	getGroupProducts	25
4.3.3	GroupController	26
4.3.3.1	Introduzione	26
4.3.3.2	Implementazione	26
4.3.3.3	managerChanges	26
4.3.3.4	createGroup	27
4.3.3.5	getGroupProducts	27
4.3.4	ParticipationController	28
4.3.4.1	Introduzione	28
4.3.4.2	Implementazione	28
4.3.4.3	list	28
4.3.4.4	requestAccessSend	29
4.3.4.5	setParticipations	29
4.3.5	ProductController	30
4.3.5.1	Implementazione	30
4.3.5.2	addRequest	30
4.3.5.3	saveOrder	31
4.3.6	InvoiceController	32

4.3.6.1	Introduzione	32
4.3.6.2	Implementazione	32
4.3.7	Implementazione	32
4.3.7.1	createPDF	32
4.3.8	EmailController	34
4.3.8.1	Introduzione	34
4.3.8.2	Implementazione	34
4.3.8.3	index	34
4.3.9	UserController	35
4.3.9.1	Introduzione	35
4.3.9.2	Implementazione	35
4.3.9.3	saveNewInfo	35
4.4	View	36
4.4.1	layouts	36
4.4.2	homeUtente:	36
4.4.3	group	36
4.4.4	emailBase:	36
4.4.5	userInfo:	36
4.5	Route	37
4.5.1	Auth::routes()	37
4.5.2	groupStart	37
4.5.3	groupCreate	37
4.5.4	groupInfo	37
4.5.5	groupManagerChanges	37
4.5.6	pageManagerProduct	37
4.5.7	visibleProduct	37
4.5.8	addProduct	38
4.5.9	addProductRequest	38
4.5.10	editProduct	38
4.5.11	saveEditProduct	38
4.5.12	listGroup	38
4.5.13	changeGroup	38
4.5.14	requestAccessSend	38
4.5.15	requestAccessPage	38
4.5.16	setParticipations	38
4.5.17	listParticipants	39
4.5.18	block	39
4.5.19	order	39
4.5.20	saveOrder	39
4.5.21	approveProducts	39
4.5.22	invoice	39
4.5.23	createPDF	39
4.5.24	user	39
4.5.25	saveNewInfo	39
5	Test	40
5.1	Protocollo di test	40
5.2	Risultati test	53
6	Consuntivo	54
7	Conclusioni	55
7.1	Sviluppi futuri	55
7.2	Considerazioni personali	55
8	Sitografia	56
9	Glossario	57

10 Elenco Immagini	58
11 Allegati	59

1 Introduzione

1.1 Informazioni sul progetto

- **Sezione:** Informatica
- **Classe:** I4AC
- **Supervisore:** William Peretti
- **Perito:** Roberto Guidi
- **Titolo:** Shop online per i gruppi
- **Data Inizio:** 02.05.2023
- **Data Fine:** 26.05.2023
- **Documentazione:** Una documentazione completa del lavoro svolto
- **Diari:** Aggiornamenti costante per ogni sessione di lavoro

1.2 Abstract

The project is to develop a dedicated online shopping site for groups, offering a collaborative, convenient and personalized shopping experience. Group members will be able to create an account, view and select products, and complete purchases collectively. The main goal is to simplify the shopping process for groups by facilitating bulk purchases and the use of exclusive discounts. In addition, invoice management will be integrated to enable group leaders to track individual expenditures. The ultimate goal of the project is to provide a reliable and comprehensive platform for the shopping needs of groups, ensuring an optimized and personalized shopping experience.

1.3 Scopo

Lo scopo del progetto riguarda lo sviluppo di un sito per lo shopping online dedicata ai gruppi. L'obiettivo è fornire agli utenti la possibilità di effettuare acquisti e gestire i loro ordini in modo collaborativo e conveniente. La piattaforma consentirà ai gruppi di creare un account, in cui i membri potranno visualizzare e selezionare prodotti, aggiungerli al carrello ed effettuare l'acquisto in modo collettivo. L'obiettivo principale è semplificare il processo di shopping per i gruppi, consentendo loro di ordinare prodotti in grandi quantità o di sfruttare vantaggi speciali come sconti e offerte promozionali riservati ai gruppi. L'obiettivo è creare un'esperienza di shopping online efficiente e personalizzata per i gruppi, che sia conveniente, user-friendly e soddisfi le loro esigenze specifiche. In sintesi, il progetto mira a creare una piattaforma di shopping online dedicata ai gruppi, offrendo loro un'esperienza di acquisto collaborativa, conveniente e personalizzata.

2 Analisi

2.1 Analisi del dominio

Fin'a ora esistevano già altri siti per lo Shopping per gruppi ma ognuno aveva delle mancanze più o meno gravi. Questo progetto vuole racchiudere in un solo sito tutta la gestione per ordini di un gruppo di persone. Per semplificare il compito al massimo per i capigruppo.

2.2 Analisi dei mezzi

2.2.1 Software

- Visual studio Code 1.67.2
- Windows 10
- Laravel 9

2.2.2 Hardware

- PC scolastico
 - Windows 10 Enterprise 21H2
 - 16 GB RAM
 - Intel(R) Xeon(R) CPU E3-1240 v5 @ 3.50GHz
 - 256 GB di disco
- Schermo Hp

2.3 Analisi e specifica dei requisiti

ID: Req-01	
Nome	Registrazione come Utente
Priorità	1
Versione	1.1
Note	Dati necessari: nome,cognome, email,password

ID: Req-02	
Nome	Creazione del gruppo
Priorità	1
Versione	1.1
Note	Dati necessari:nome,termine1 e termine2 il nome deve essere univoco

ID: Req-03	
Nome	Essere capo gruppo
Priorità	1
Versione	1.1
Note	Una volta creato il gruppo bisogna essere capo gruppo di esso

ID: Req-04	
Nome	Gestione delle richieste di accesso
Priorità	1
Versione	1.1
Note	La richiesta deve essere accettata dal capo gruppo

ID: Req-05	
Nome	Si può accedere a un gruppo creato da altri
Priorità	1
Versione	1.1
Note	Si può mandare la richiesta a un gruppo a qui non si fa ancora parte.

ID: Req-06	
Nome	Lista gruppi solo non scaduti
Priorità	1
Versione	1.1
Note	Nella lista dei gruppi ci siano solo gruppi non ancora scaduti.

ID: Req-07	
Nome	Creazione di fatture per utente
Priorità	2
Versione	1.1
Note	Formato pdf

ID: Req-08	
Nome	Deve essere online
Priorità	1
Versione	1.1
Note	

ID: Req-09	
Nome	Aggiunta di prodotti
Priorità	1
Versione	1.1
Note	Ogni capo gruppo aggiunge i prodotti per i propri gruppi

ID: Req-10	
Nome	Si devono poter modificare i prodotti una volta creati
Priorità	2
Versione	1.1
Note	

ID: Req-11	
Nome	Il repsonsabile del gurppo fa parte del gruppo
Priorità	2
Versione	1.1
Note	

ID: Req-12	
Nome	Test effettuati con molteplici dati nel database
Priorità	3
Versione	1.1
Note	

ID: Req-13	
Nome	Possibile partecipare a più gruppi
Priorità	1
Versione	1.1
Note	

ID: Req-14	
Nome	Un utente può essere sia capo gruppo che utente in altri gruppi
Priorità	1
Versione	1.1
Note	Un utente può sia essere capo gruppo che partecipante a un gruppo di qui non è capo gruppo

ID: Req-15	
Nome	Superato il termine2 l'utente non può modificare il carrello
Priorità	1
Versione	1.1
Note	Entro il tempo massimo impostato dal capo gruppo

ID: Req-16	
Nome	Scadenza termine1
Priorità	1
Versione	1.1
Note	Scaduto il termine finché il capo gruppo non aggiorna i carrelli non si possono modificare

ID: Req-17	
Nome	Effettuare ordine
Priorità	1
Versione	1.1
Note	Una volta effettuato la prima fase i carrelli si aggiornano con solo i prodotti approvati dal capo gruppo.

ID: Req-18	
Nome	Effettuare la lista dei partecipanti
Priorità	3
Versione	1.1
Note	

ID: Req-19	
Nome	Ricevere le email come capogruppo quando scade un termine
Priorità	3
Versione	1.1
Note	

2.4 Use Case

Questo è lo schema use case che ho previsto per il funzionamento del sito. Un utente può essere capo gruppo solo se crea un gruppo. Altrimenti sarà un utente normale. Che potrà gestire il proprio carrello collegato al gruppo. Un capogruppo può gestire i prodotti creandone o modificandoli, può anche nascondarli. In oltre un capo gruppo può approvare i prodotti e creare le fatture per il proprio gruppo. Un gruppo ha 1 ordine quindi una volta utilizzato il gruppo "muore".

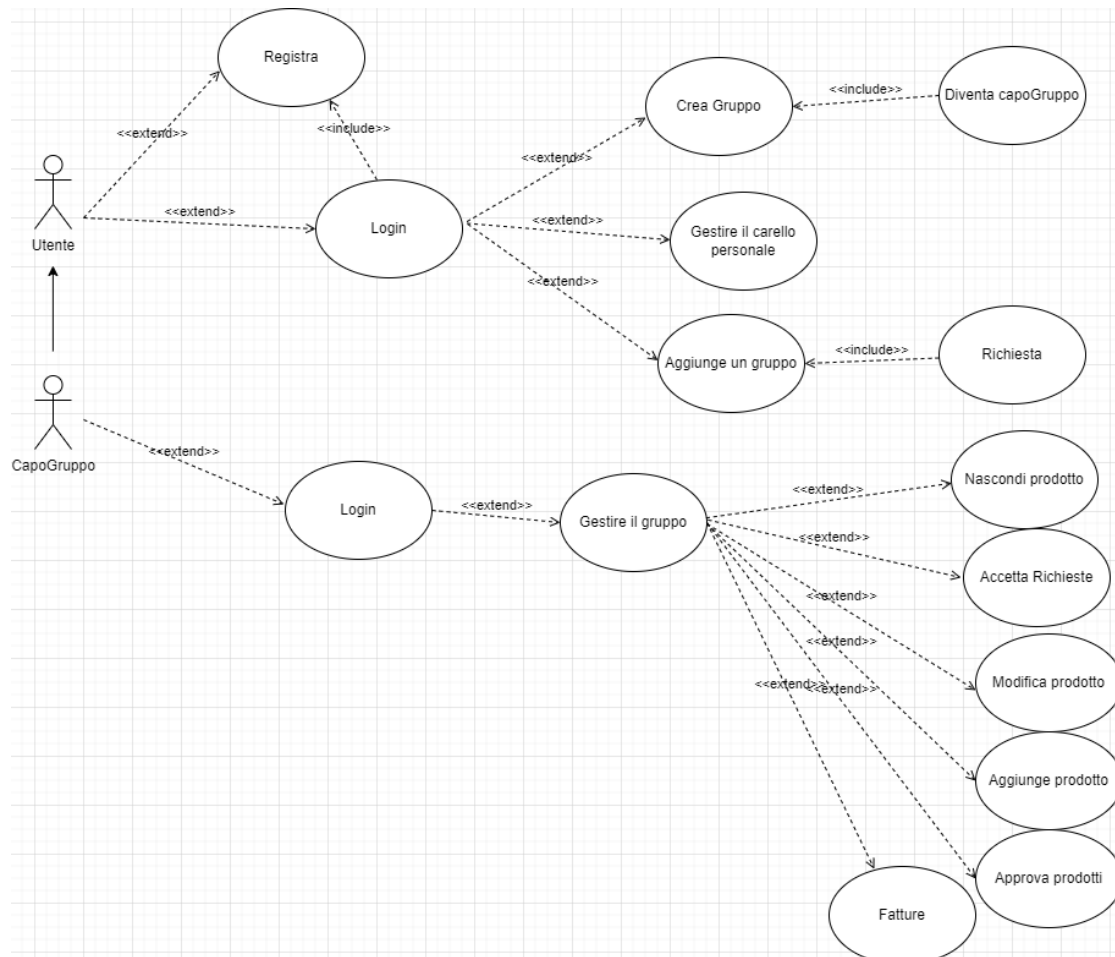


Figura 1: Schema Use Case

2.5 Pianificazione

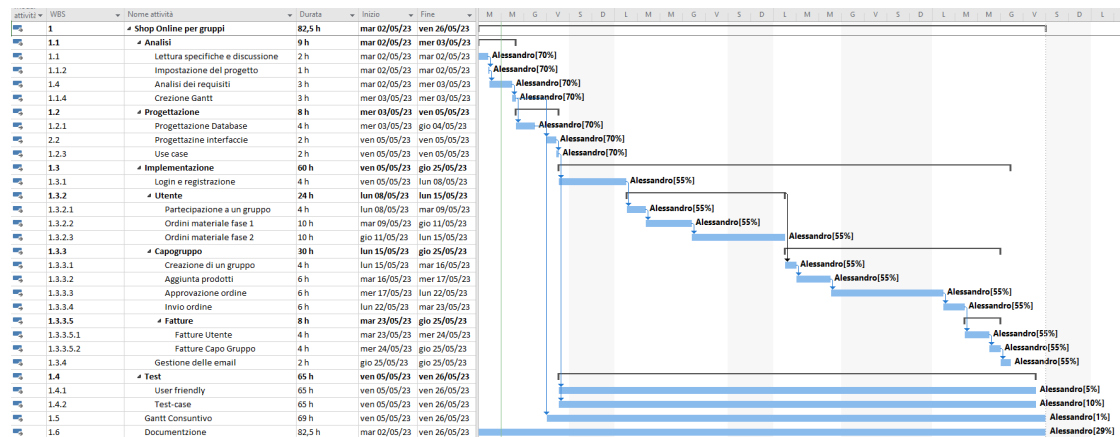


Figura 2: Gantt preventivo

Ho scelto il metodo Waterfall Gantt per il mio LPI in quanto apprezzo un approccio lineare e strutturato. Utilizzando questo metodo, posso pianificare con precisione le fasi del progetto e monitorare le mie attività. Inoltre, poiché si tratta di un progetto personale, non ho bisogno di coordinarmi con altre persone, semplificando così la gestione complessiva. In oltre essendo un progetto di solo 80 ore non avrebbe senso fare per esempio una progettazione agile.

Analizzando il diagramma di Gantt per la gestione delle ore, è evidente che la maggior parte del tempo è dedicata all'implementazione, in particolare al lato utente e alla gestione "finale" di entrambe le fasi di acquisto. Inoltre, ho dedicato una considerevole quantità di tempo alle fatture, poiché non ho mai avuto l'occasione di crearle in precedenza. Un'altra porzione significativa di tempo è stata assegnata ai test, che ho deciso di effettuare in parallelo alla fase di programmazione.

Per avere una visione più dettagliata della pianificazione, ho allegato la pianificazione completa in formato A3 alla fine della documentazione.

3 Progettazione

3.1 Design dei dati e database

Il database è stato progettato per gestire in modo efficiente un sistema complesso che coinvolge utenti, gruppi, prodotti e carrelli degli utenti. Questo viene realizzato attraverso diverse tabelle che rappresentano le diverse entità e le loro relazioni. La tabella **Users** contiene dettagli degli utenti registrati nel sistema. La tabella **Groups** rappresenta i vari gruppi presenti nel sistema. La tabella **Products** fornisce informazioni sui prodotti disponibili a quel gruppo. La tabella **Participations** gestisce le associazioni tra gli utenti e i gruppi ai quali partecipano. La tabella **Carts** contiene dettagli sugli articoli presenti nei carrelli degli utenti. Per garantire l'integrità dei dati, sono stati definiti vincoli di chiave primaria e vincoli di chiave esterna nelle tabelle. Ciò significa che gli attributi che rappresentano le relazioni tra le tabelle fanno riferimento agli id delle entità correlate.

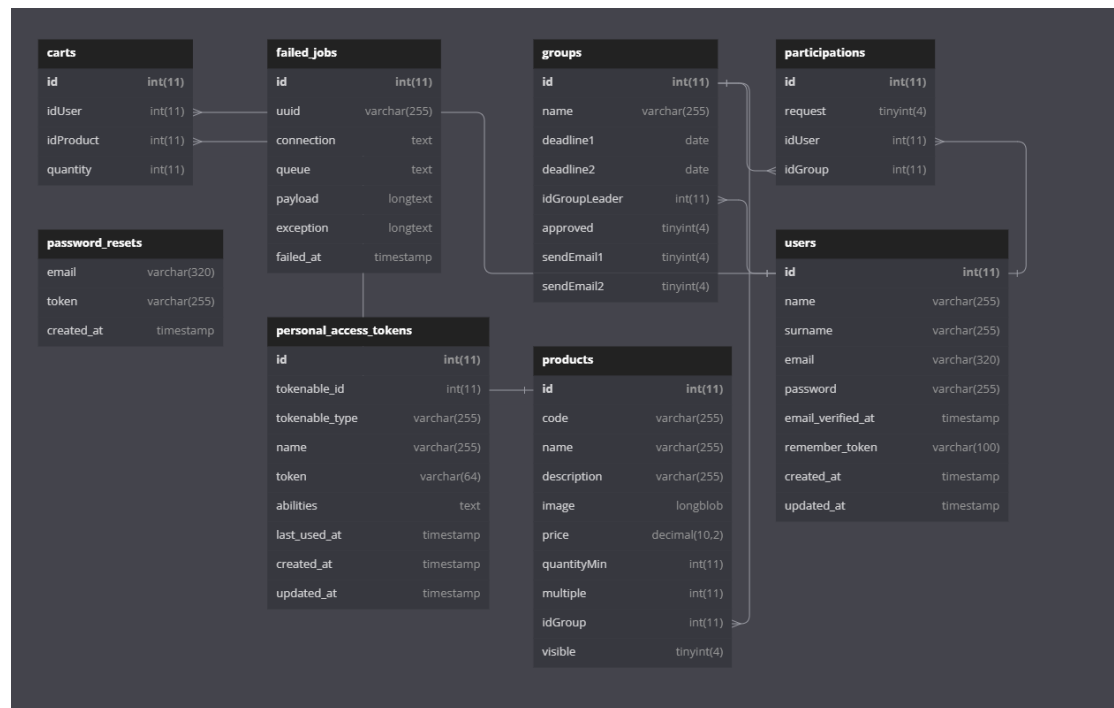
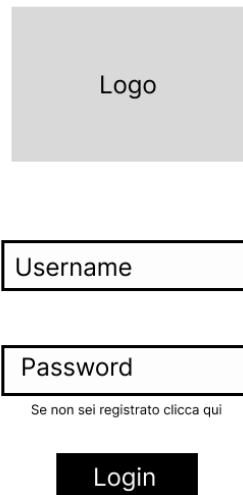


Figura 3: Design del Database

3.2 Design delle interfacce

3.2.1 Login

La prima interfaccia che ho pensato è stata la prima che gli utenti vedono quando vanno sul sito, che è quella del login. La schermata di login ha il logo del progetto al centro della pagina e i campi di input per nome utente e la password. La schermata è stata progettata in modo chiaro e semplice, con colori semplici e facilmente riconoscibili:



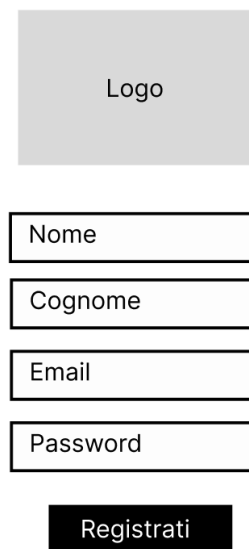
The image shows a login form design. At the top is a gray rectangular box labeled "Logo". Below it are two white rectangular input fields with black borders, labeled "Username" and "Password". Under the "Password" field is a small text link that says "Se non sei registrato clicca qui". At the bottom is a black rectangular button with the word "Login" in white text.

Figura 4: Design del Login

Inoltre, nella schermata di login è presente un messaggio sotto il campo della password che consente agli utenti non registrati di effettuare la registrazione e, successivamente, verranno reindirizzati alla pagina successiva.

3.2.2 Registrazione

La seconda schermata che ho progettato è stata proprio la schermata di registrazione con cura per offrire un'esperienza intuitiva agli utenti che desiderano creare un nuovo account. Il logo del progetto è posizionato al centro della pagina, mentre i campi di input per il nome, cognome, password e email sono disposti in modo chiaro e semplice. I colori utilizzati sono semplici e facilmente riconoscibili, contribuendo a creare un design visivamente piacevole:



Il diagramma illustra l'interfaccia di registrazione. Al centro c'è un rettangolo grigio con la scritta "Logo". Sotto di esso, quattro campi di input rettangolari sono allineati verticalmente: "Nome", "Cognome", "Email" e "Password". In fondo, un pulsante rettangolare nero con la scritta "Registrati" in bianco completa la struttura.

Figura 5: Design della Registrazione

3.2.3 Home

Successivamente ho progettato la Home dell'utente, In alto a sinistra si vede il gruppo a qui si sta partecipando e i prodotti. Cliccando sui Gruppi si avrà la possibilità di creare un gruppo, accedere al Gruppo o gestire un gruppo in caso che si è capogruppo. In fine il classico tasto di logout. Nella parte inferiore sono presenti tutti i prodotti con le varie informazioni.

Nome Gruppo Scadenza		Gruppi			Logout
Nome prodotto	Descrizione	Min	Multiplo	Prezzo	Quantità
Abc	testo di prova	2	1	10.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	12	6	1.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	5	5	30.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	50	1	20.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	14	2	50.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	100	1	5.00 CHF	<input type="text" value="0"/>
Abc	testo di prova	2	2	9.50 CHF	<input type="text" value="0"/>
Abc	testo di prova	9	3	8.25 CHF	<input type="text" value="0"/>
Abc	testo di prova	2	1	10.0 CHF	<input type="text" value="0"/>

Figura 6: Design Home prima versione

							Logout
Gruppo Ordina Gestisci Crea gruppo Lista gruppi	Nome prodotto	Descrizione	Min	Multiplo	Prezzo	Quantità	
	Abc	testo di prova	2	1	10.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	12	6	1.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	5	5	30.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	50	1	20.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	14	2	50.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	100	1	5.00 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	2	2	9.50 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	9	3	8.25 CHF	<input type="text" value="0"/>	
	Abc	testo di prova	2	1	10.0 CHF	<input type="text" value="0"/>	

Figura 7: Design Home versione due

Dopo aver disegnato la prima interfaccia il mandante mi ha chiesto delle modifiche e da qui nasce la versione due dove appare il menù laterale. In questa interfaccia è cambiato poco. Parlando un attimo del menù laterale ha una prima voce che si chiama Gruppo dove cliccando si aprono tutti i gruppi a qui una persona partecipa e da lì si potrà anche gestire un gruppo in qui si è capo gruppo. Data questa aggiunta sparisce da in alto la voce gruppi.

3.2.4 Gruppi

Successivamente ho progettato la schermata dei gruppi. In tale schermata si ha sia la possibilità di creare un gruppo e di conseguenza diventare capogruppo. Che accedere sul lato destro tramite una lista a un gruppo già esistente. La barra superiore come si può vedere rimarrà uguale per tutto il sito.



Home Gruppi LogOut

Crea Gruppo

NomeGruppo

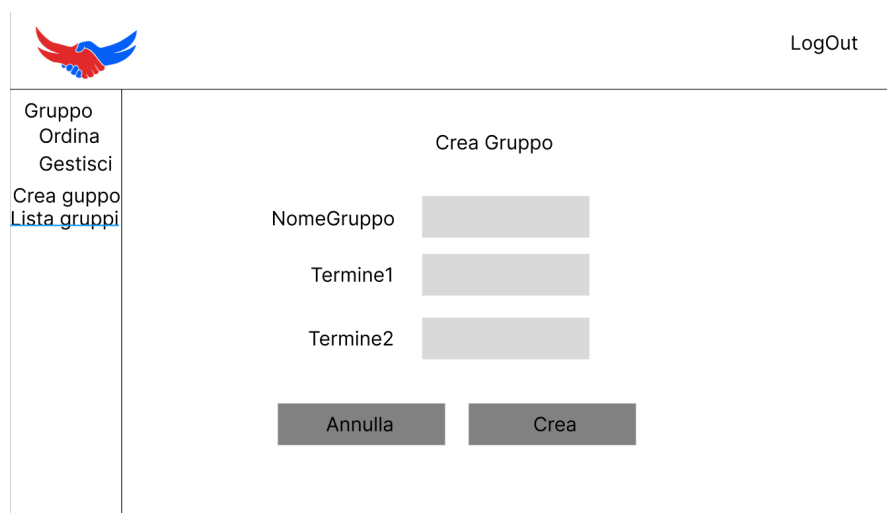
Termine1


Termine2

Lista gruppi

Figura 8: Design della schermata dei gruppi

Una volta mandata la richiesta a un gruppo codesto apparirà nella lista dei gruppi in alto a sinistra e nella lista di tutti i gruppi cambierà lo stato di quel gruppo. L'utente capirà che la sua richiesta è in attesa perché al posto del tempo che manca per il termine dell'ordine si vedrà la scritta in attesa. In caso che la richiesta venga rifiutata il gruppo sparirà dalla lista in alto a sinistra e si potrà mandare nuovamente la richiesta. Non possono esistere due gruppi con lo stesso nome. Nella seconda versione la schermata è stata sfoltita un po' e quindi la creazione e



 LogOut

Gruppo
Ordina
Gestisci
Crea gruppo
[Lista gruppi](#)

Crea Gruppo

NomeGruppo

Termine1

Termine2

Figura 9: Design creazione gruppo seconda versione

l'accesso al gruppo sono state divise in due pagine. Sono rimaste identiche ma solo divise qui sopra possiamo vedere la creazione con il menù laterale che era stato chiesto.

3.2.5 Lista gruppi

Questa interfaccia è stata disegnata durante il restyling delle interfacce. In questa interfaccia saranno presenti tutti i gruppi ancora disponibili quindi che non hanno già superato la prima scadenza.



Figura 10: Design della schermata della lista dei gruppi esistenti.

Questa interfaccia sarà anche la stessa che l'utente vedrà al suo primo login o finché non parteciperà a un gruppo o ne creerà uno.

3.2.6 Gestione Gruppi

L'ultima pagina di gestione che sono andato a progettare è quella dedicata alla gestione del gruppo stesso. In alto come su tutto il sito abbiamo la barra degli strumenti che ci segue ma in questo caso al posto di avere la voce gestisci abbiamo la voce Home in quanto siamo già nella schermata in questione. Tale schermata ha l'utilità di gestire tutti gli elementi che riguardano un gruppo. Sia le due scadenze, che le richieste di accesso al gruppo. Infine la parte più importante si trova sul lato destro dove troviamo la lista di tutti i prodotti presenti e ordinabili con anche la loro relativa quantità ordinata finora.

Nome Gruppo		Gruppi	LogOut
Fatture	Aggiungi Prodotto	Lista prodotti	
Termine 1			
Termine 2			
Gruppo	Partecipanti 10 Elimina Gruppo		
Richieste			

Figura 11: Design della schermata della gestione del gruppo.

I prodotti vanno caricati a mano per ogni gruppo, due gruppi possono avere lo stesso prodotto ma deve essere caricato per ogni gruppo. Il tasto fatture sarà disabilitato fino al termine della seconda scadenza. Una volta superato il secondo termine non si potranno più accettare utenti o modificare i prodotti. Si potrà solo visionare le fatture.

		Gestione Gruppo		LogOut
 Gruppo Ordina Gestisci Crea gruppo Lista gruppi				Elimina Gruppo
	Nome			
	Termine 1			
	Termine 2			
	Partecipanti	10		
	Richieste	2	ico	
		Annulla	Salva	

Figura 12: Design della schermata della gestione del gruppo seconda versione.

3.2.7 Aggiungi prodotto

Per quanto riguarda l'aggiunta di prodotti, ho dedicato particolare attenzione alla creazione di una schermata che fosse snella, pulita e intuitiva per gli utenti. Ho organizzato i campi necessari in modo da essere facilmente accessibili e comprensibili. La disposizione degli elementi è stata studiata per massimizzare l'efficienza dell'inserimento delle informazioni. I campi sono stati impilati in modo da seguire un flusso logico, consentendo agli utenti di procedere senza intoppi durante l'aggiunta dei prodotti.

Nome Gruppo

Gruppi

Logout

Nome prodotto

Descrizione

Prezzo

Quantità Minima


Multiplo

Immagine

Aggiungi

Figura 13: Design della schermata per l'aggiunta dei prodotti.

L'obiettivo principale era creare un'esperienza utente fluida e senza complicazioni nella gestione dei prodotti. Con questa schermata snella e pulita, gli utenti possono facilmente inserire tutte le informazioni necessarie per aggiungere nuovi prodotti al gruppo, risparmiando tempo e sforzo nel processo di gestione complessiva. Come si può notare è rimasta prettamente uguale è solo



Logout

Gruppo
Ordina
Gestisci
Crea gruppo
[Lista gruppi](#)

Nome prodotto

Descrizione

Prezzo

Quantità Minima

Multiplo

Immagine

Aggiungi

Figura 14: Design della schermata per l'aggiunta dei prodotti versione due.

apparso il menù laterale e sparita la voce soprastante per la gestione dei gruppi come già detto in precedenza.

3.2.8 Modifica prodotto

Per quanto riguarda questa schermata è identica a quella di aggiunta dei prodotti solo che sono cambiati i tasti.

Nome Gruppo		Gruppi		LogOut
Nome prodotto	<input type="text"/>	Descrizione	<input type="text"/>	
Prezzo	<input type="text"/>			
Quantità Minima	<input type="text"/>			
Multiplo	<input type="text"/>	Immagine	<input type="text"/>	
		<input type="button" value="Modifica"/> <input type="button" value="Elimina"/>		

Figura 15: Design della schermata per la modifica dei prodotti.


				LogOut
Gruppo Ordina Gestisci Crea gruppo Lista gruppi	Nome prodotto	<input type="text"/>	Descrizione	<input type="text"/>
	Prezzo	<input type="text"/>		
	Quantità Minima	<input type="text"/>		
	Multiplo	<input type="text"/>	Immagine	<input type="text"/>
			<input type="button" value="Modifica"/> <input type="button" value="Elimina"/>	

Figura 16: Design della schermata per la modifica dei prodotti versione due.

Come si può notare anche questa come la schermata per l'aggiunta di prodotti è rimasta uguale ma è solo apparso il menù laterale.

3.2.9 Fattura

Infine ho progettato la schermata delle fatture, molto semplice e pulita. Una lista di tutti gli utenti che hanno fatto un acquisto vedendo nome e spesa con la possibilità di scaricare la fattura per ogni utente. In fine nella parte inferiore troviamo il resoconto di quanto il gruppo ha speso



Home	Gruppi	LogOut
Nome	Spesa	Scarica
Alessandro	10.00 CHF	
Totale Nome Gruppo		10.00 CHF 

Figura 17: Design della schermata per le fatture.

per tutto l'ordine e la possibilità di scaricare una fattura generale. Anche per questa interfaccia




 Gruppo Ordina Gestisci Crea gruppo Lista gruppi	LogOut			
	Nome	Cognome	Importo	Scarica
	Alessandro	Aloise	10.00 CHF	
	Totale Nome Gruppo			10.00 CHF 

Figura 18: Design della schermata per le fatture versione due.

è rimasto tutto uguale a parte il menù laterale come dicevo che è apparso in tutte le interfacce.

4 Implementazione

4.1 Struttura delle cartelle

Questo progetto è stato sviluppato con Laravel, prevede una struttura di cartelle organizzata, in modo gerarchico per facilitare l'implementazione delle funzionalità. In generale, per usare Laravel dovremmo lavorare con 5 cartelle principali e un file.

- **Controllers:** Contiene i controller che gestiscono le azioni scaturite dall'utente. Contiene le classi che implementano la logica dell'applicazione. Queste classi agiscono come intermediari tra il Model e la View, esponendo i dati del Model alle View e gestendo gli eventi delle View.
- **Model:** Contiene le classi che definiscono il modello dati dell'applicazione. Queste classi rappresentano i dati con cui l'applicazione lavorerà.
- **Routes:** Contiene le definizioni URL dell'applicazione e le associano alle azioni specifiche del controller permettendo di gestire le richieste in base all'URL.
- **View:** Contiene le classi che definiscono l'interfaccia grafica dell'applicazione. Queste classi si occupano di mostrare i dati dell'applicazione e di gestire le interazioni dell'utente con l'applicazione.
- **Database** Contiene le migrations.
- **.env** Contiene variabili di configurazione utilizzate dall'applicazione, come le informazioni di connessione al database, le impostazioni di caching e le credenziali per l'invio delle email.

4.2 Database

4.2.1 migrations

Le migrations per laravel sono

4.2.2 Users

Questa è un esempio di migrations che è stata creata per la creazione degli utenti. La classe Users è una classe che estende Migration. La classe che con il nome intero si chiamerebbe **2014_10_12_000000_create_users_table** dove la prima parte del nome sta a significare il giorno e l'ora in cui viene creata la classe. Mentre la seconda parte del nome è una convenzione di Laravel quindi il create seguito dal nome della tabella e in fine table. Il metodo sottostante che si chiama up, al suo interno definisce la tabella e i suoi attributi. In questo caso abbiamo id come chiave primaria e a seguire una serie di dati "standard" per un nome utente. Fino ad arrivare agli ultimi 3 parametri che dei parametri che ci mette di base Laravel quando creiamo un utente.

```
public function up()
{
    Schema::create('users', function (Blueprint $table) {
        $table->id();
        $table->string('name', 255);
        $table->string('surname', 255);
        $table->string('email', 320);
        $table->string('password', 255);
        $table->timestamp('email_verified_at')->nullable();
        $table->rememberToken();
        $table->timestamps();
    });
}
```

Un'altra cosa Laravel di default mette sempre il timestamps a tutte le tabelle. Codesto serve a...

4.3 Controller

In Laravel, i controller sono classi che gestiscono la logica dietro le richieste fatte all'applicazione. Fanno da intermediari tra le Root e i model/view, consentendo di organizzare e separare i diversi aspetti della funzionalità dell'applicazione. I controller gestiscono la convalida degli input, interagiscono con il database attraverso i model e restituiscono le risposte/view appropriate al client.

4.3.1 HomeController

4.3.1.1 Introduzione

La classe HomeController rappresenta il controller che gestisce la logica per quando si carica la home del sito web. Il controller deciderà se renderizzare l'utente alla schermata home o alla lista dei gruppi. In base a se partecipa già a un gruppo o meno. In caso che partecipasse a più gruppi viene mostrato nella home il primo in ordine alfabetico.

4.3.2 Implementazione

La parte più importante della classe è il metodo index() che recupera l'ID dell'utente autenticato e ottiene le partecipazioni associate e getGroupsWithParticipations() che si occupa di recuperare tutti i gruppi con le relative partecipazioni per l'utente specificato. Infine c'è il metodo getGroupProducts() che si occupa di recuperare i prodotti del gruppo specificato per l'utente autenticato. Grazie a questi tre metodi possiamo gestire tutta la schermata Home e la lista gruppi. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.2.1 index

Questo metodo viene invocato subito dopo il login dell'utente. Il metodo recupera l'ID dell'utente autenticato e ottiene le partecipazioni associate. Se l'utente non ha partecipazioni, recupera tutti i gruppi con le relative partecipazioni e mostra la vista 'group.list' dove sono presenti la lista di tutti i gruppi. Altrimenti, recupera l'ID del primo gruppo delle partecipazioni in ordine alfabetico, ottiene i prodotti del gruppo con le informazioni aggiuntive come la quantità dell'utente nel carrello e mostra la vista 'homeUtente' con le informazioni del gruppo e i prodotti.

```
public function index()
{
    $userID = Auth::user()->id;

    $participations = DB::table('groups')
        ->join('participations', 'groups.id', '=',
            'participations.idGroup')
        ->select( '*' )
        ->where('participations.idUser',Auth::user()->id)
        ->orderBy('groups.name', 'ASC')
        ->get();

    if ($participations->isEmpty()) {
        $groups = $this->getGroupsWithParticipations($userID);
        return view('group.list', ['groups' => $groups]);
    } else {
        $groupID = $participations[0]->idGroup;
        $products = $this->getGroupProducts(Auth::user()->id);

        return view('/homeUtente', ['groupInfo' => $participations,
            'products' => $products]);
    }
}
```


4.3.2.2 getGroupsWithParticipations

Questo metodo viene invocato quando la query che controlla se si sta partecipando a un gruppo ritorna null. Questo metodo recupera tutti i gruppi con le relative partecipazioni per l'utente specificato. Il metodo esegue una query per ottenere i gruppi a cui l'utente partecipa, inclusi quelli senza partecipazioni.

```
private function getGroupsWithParticipations($userID)
{
    return Group::leftJoin('participations',
        function ($join) use ($userID) {
            $join->on('groups.id', '=', 'participations.idGroup')
                ->where('participations.idUser', '=', $userID);
        })
        ->select('groups.*', DB::raw('IFNULL(COALESCE(
            participations.request, "dont\'set"), "dont\'set") as request'))
        ->get();
}
```

4.3.2.3 getGroupProducts

Questo metodo viene invocato quando la query che controlla se si sta partecipando a un gruppo ritorna che si sta partecipando almeno 1 gruppo. Questo metodo recupera i prodotti del gruppo specificato per l'utente autenticato. Il metodo esegue una query per ottenere i prodotti di un gruppo, inclusi informazioni aggiuntive come la quantità dell'utente nel carrello.

```
private function getGroupProducts($groupID)
{
    $products = DB::table('products')
        ->select('products.id', 'products.name', 'products.description',
            'products.price', 'products.quantityMin', 'products.multiple',
            DB::raw('IFNULL(carts.quantity, 0) AS userQuantity'),
            products.idGroup, 'products.image')
        ->join('groups', 'products.idGroup', '=', 'groups.id')
        ->join('participations', 'groups.id', '=', 'participations.idGroup')
        ->join('users', 'participations.idUser', '=', 'users.id')
        ->leftJoin('carts', function ($join) {
            $join->on('products.id', '=', 'carts.idProduct')
                ->where('carts.idUser', '=', Auth::user()->id);
        })
        ->where('users.id', Auth::user()->id)
        ->where('products.idGroup', $groupID)
        ->where('products.visible', 0)
        ->orderBy('products.name', 'ASC')
        ->get();
    return $products;
}
```

4.3.3 GroupController

4.3.3.1 Introduzione

La classe GroupController appresenta il controller che gestisce la logica per i gruppi. È responsabile di creare nuovi gruppi, gestire le modifiche al gruppo effettuate dal manager, visualizzare le informazioni del gruppo, cambiare il gruppo corrente dell'utente e altre operazioni correlate.

4.3.3.2 Implementazione

La parte più importante della classe è il metodo managerChanges() per la gestione delle modifiche al gruppo da parte del manager, createGroup() per la creazione di nuovi gruppi e getGroupProducts() per il recupero dei prodotti associati a un gruppo. Questi metodi svolgono un ruolo fondamentale nella gestione, creazione e visualizzazione delle informazioni dei gruppi nel sistema. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.3.3 managerChanges

Questo metodo gestisce le modifiche al gruppo effettuate dal CapoGruppo. Valida i dati ricevuti, ottiene le informazioni del gruppo e i prodotti e in base all'azione richiesta aggiorna il nome e le scadenze del gruppo. Infine, restituisce la vista 'group.manager' con i prodotti e le informazioni del gruppo aggiornate.

```
public function managerChanges(Request $request){
    $this->validate($request, [
        'name' => 'required|string',
        'deadline1'=>'required|date',
        'deadline2'=>'required|date',
    ]);

    $groupInfo = $this->getGroupInfo($request->groupId);
    $products = $this->getGroupProducts($groupInfo[0]->id);
    if($request->action == 1){
        $existingGroup = Group::where('name', $request->name)->first();
        if ($existingGroup && $request->oldName != $request->name) {
            // return redirect()->back()->withErrors(['name' => 'Il nome del
            gruppo gi stato utilizzato.']);
        }
        else
        {
            Group::where('id', $request->groupId)->update([
                'name' =>$request->name,
                'deadline1' =>$request->deadline1,
                'deadline2' => $request->deadline2,
            ]);
        }
    }
    return view('group.manager', ['products' => $products, 'groupInfo' =>
    $groupInfo]);
}
```

4.3.3.4 createGroup

Questo metodo crea un nuovo gruppo utilizzando i dati forniti nella richiesta e l'id dell'utente. Effettua la validazione dei dati, quindi crea un nuovo record nella tabella dei gruppi con il nome, le scadenze e l'id del leader del gruppo. Infine, restituisce l'istanza del gruppo appena creato.

```
private function createGroup(Request $request, $userID){
    $this->validate($request, [
        'nameGroup' => 'required|string',
        'deadline1' => 'required|date',
        'deadline2' => 'required|date',
    ]);

    return Group::create([
        'name' => $request->nameGroup,
        'deadline1' => $request->deadline1,
        'deadline2' => $request->deadline2,
        'idGroupLeader' => $userID,
    ]);
}
```

4.3.3.5 getGroupProducts

Questo metodo ottiene i prodotti assegnati a un gruppo specificato dall'id del gruppo. Utilizza una query per selezionare i prodotti dalla tabella dei prodotti e calcola la quantità totale ordinata per ciascun prodotto utilizzando una sottoquery. I prodotti sono ordinati per nome e infine restituiti come il risultato.

```
private function getGroupProducts($groupID){
    $products = DB::table('products')
        ->select('products.*', DB::raw('(SELECT SUM(quantity) FROM carts WHERE
idProduct = products.id) AS totalOrderedQuantity'))
        ->where('idGroup', $groupID)
        ->orderBy('name')
        ->get();
    return $products;
}
```

4.3.4 ParticipationController

4.3.4.1 Introduzione

La classe ParticipationController appresenta il controller che gestisce la logica per le partecipazioni. È responsabile di mandare richieste di partecipazione, accettarle o rifiutarle e anche responsabile di mostrare a schermo la lista dei partecipanti.

4.3.4.2 Implementazione

La parte più importante della classe è il metodo list() mostra l'elenco dei gruppi con le relative partecipazioni. Il metodo requestAccessSend() invia una richiesta di accesso per un gruppo. Il metodo setParticipations() accetta o rifiuta le richieste di accesso. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.4.3 list

Questo metodo mostra l'elenco dei gruppi con le relative partecipazioni. Il metodo recupera l'ID dell'utente autenticato e ottiene i gruppi con le partecipazioni associate. Mostra la vista 'group.list' con l'elenco dei gruppi.

```
public function list(){
    $userID = Auth::user()->id;

    $groups = $this->getGroupsWithParticipations($userID);

    return view('group.list', ['groups' => $groups]);
}
```

4.3.4.4 requestAccessSend

Questo metodo invia una richiesta di accesso per un gruppo. Il metodo valida l'ID del gruppo fornito nella richiesta. Verifica se l'utente ha già inviato una richiesta per lo stesso gruppo. Se non ha ancora inviato una richiesta, crea una nuova partecipazione con lo stato di richiesta in sospeso. Infine, ottiene nuovamente i gruppi con le partecipazioni associate e mostra la vista 'group.list'.

```
public function requestAccessSend(Request $request){
    $request->validate([
        'groupId' => 'required'
    ]);

    $userID = Auth::user()->id;

    $controlRequest = Participation::where('idUser', $userID)
        ->where('idGroup', $request->groupId)
        ->get();

    if ($controlRequest->isEmpty()) {
        Participation::create([
            'request' => 0,
            'idUser' => $userID,
            'idGroup' => $request->input('groupId'),
        ]);
    }

    $groups = $this->getGroupsWithParticipations($userID);

    return view('group.list', ['groups' => $groups]);
}
```

4.3.4.5 setParticipations

Questo metodo gestisce le richieste di accesso ai gruppi. Se l'azione richiesta è 1, aggiorna lo stato della partecipazione a "accettata". Altrimenti, elimina la partecipazione. Infine, ottiene le richieste di accesso in sospeso per il gruppo specificato e mostra la vista corrispondente.

```
public function setParticipations(Request $request){
    if($request->action == 1){
        DB::table('participations')
            ->where('idUser', $request->idUser)
            ->where('idGroup', $request->idGroup)
            ->update(['request' => 1]);
    }else{
        DB::table('participations')
            ->where('idUser', $request->idUser)
            ->where('idGroup', $request->idGroup)
            ->delete();
    }

    $accessRequests = Participation::where('idGroup', $request->idGroup)
        ->where('request', 0)
        ->get();

    return view('group.request', ['accessRequests' => $accessRequests]);
}
```

4.3.5 ProductController

La classe ProductController rappresenta il controller che gestisce la logica per l'aggiunta, modifica e visualizzazione dei prodotti all'interno di un gruppo. Gestisce anche la visibilità dei prodotti e la visualizzazione delle pagine di gestione e ordinazione dei prodotti del gruppo.

4.3.5.1 Implementazione

La parte più importante della classe è il metodo `addRequest()` che si occupa di aggiungere un nuovo prodotto al gruppo quando il metodo viene invocato. Un altro metodo molto importante è il metodo `saveOrder()` che si occupa di salvare gli ordini degli utenti. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.5.2 addRequest

Questo metodo riceve una richiesta con i dati del prodotto da aggiungere. Se l'azione richiesta è 1, crea un nuovo oggetto "Product" nel database con i dati forniti. Successivamente, ottiene i prodotti del gruppo e le informazioni del gruppo stesso e restituisce la vista "group.manager" con i prodotti e le informazioni aggiornate.

```
public function addRequest(Request $request){
    $groupId = $request->groupId;
    $currency= str_replace("'", "",substr($request->currency, 0, -3));
    if( $request->action == 1){
        Product::create([
            'code' => $request->codice,
            'name' => $request->nome,
            'description' => $request->description,
            'price' => $currency,
            'quantityMin' => $request->minimumQuantity,
            'multiple' => $request->quantityMultiple,
            'idGroup' => $groupId,
            'visible' => 0,
        ]);
    }

    $products = $this->getGroupProducts($groupId);
    $groupInfo = $this->getGroupInfo($groupId);

    return view('group.manager', ['products' => $products, 'groupInfo' =>
    $groupInfo]);
}
```

4.3.5.3 saveOrder

Questo metodo verifica la validità della quantità del prodotto selezionato, quindi crea o aggiorna un oggetto "Cart" nel database corrispondente all'ordine dell'utente. Infine, restituisce la vista "homeUtente" con i dati aggiornati dei prodotti e le informazioni del gruppo, eventualmente mostrando un messaggio di errore se l'utente ha inserito una quantità sbagliata.

```
public function saveOrder(Request $request){
    $error= "";
    $idUser = Auth::user()->id;
    $idProduct = $request->product_id;
    $quantity = $request->quantity;

    $controlProduct = DB::table('carts')
        ->where('idUser', $idUser)
        ->where('idProduct', $idProduct)
        ->get();

    $controlMultiple = DB::table('products')
        ->where('id', $idProduct)
        ->get();

    $groupId = $request->group_id;
    $groupInfo = $this->getGroupInfo($groupId);
    $products = $this->getGroupProducts($groupId);

    if ($quantity == 0 || ($quantity >= $controlMultiple[0]->quantityMin && (
    $quantity % $controlMultiple[0]->multiple == 0) && $controlMultiple[0]->
    visible == 0)) {
        if ($controlProduct->isEmpty()) {
            Cart::create([
                'idUser' => $idUser,
                'idProduct' => $idProduct,
                'quantity' => $quantity,
            ]);
        }else{
            DB::table('carts')
                ->where('idUser', $idUser)
                ->where('idProduct', $idProduct)
                ->update(['quantity' => $quantity]);
        }
    }else{
        $error = "Campo non salvato multiplo non valido";
    }

    return view('homeUtente', ['products' => $products , 'groupInfo' =>
    $groupInfo, 'error'=>$error ]);
}
```

4.3.6 InvoiceController

4.3.6.1 Introduzione

La classe InvoiceController rappresenta il controller che gestisce la logica per quanto riguarda le creazione delle fatture ma anche il caricamento delle pagine. Il controller si occuperà con due semplici metodi di o caricare la view delle fatture oppure creare una fattura da scaricare in pdf.

4.3.6.2 Implementazione

4.3.7 Implementazione

La parte più importante della classe è il metodo createPDF() che si occupa di creare il pdf quando il metodo viene invocato. Questo metodo si appoggia a un template che ho creato per le fatture. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.7.1 createPDF

Questo metodo viene invocato quando si clicca il tasto per scaricare le fatture per ogni utente. Il metodo recupera l'ID dell'utente autenticato e va a recuperare tutti i prodotti che ha inserito nel carrello per quel gruppo. Una volta fatto questo va anche a prendere le informazioni del capogruppo per poterle mettere nella fattura. In fine va a richiamare la pagina invoiceBase che è il template della fattura che ho deciso.

```
public function createPDF(Request $request)
{
    $cartProducts = DB::table('users')
        ->join('carts', 'users.id', '=', 'carts.idUser')
        ->join('products', 'carts.idProduct', '=', 'products.id')
        ->join('groups', 'products.idGroup', '=', 'groups.id')
        ->select('products.name', 'carts.quantity', 'products.price',
            'products.code',
            DB::raw('products.price * carts.quantity AS totalPrice'),
            'groups.name AS group_name')
        ->where('users.id', $request->idUser)
        ->where('carts.quantity', '>', 0)
        ->groupBy('products.name', 'carts.quantity', 'products.price',
            'groups.name', 'products.code')
        ->get();

    $user = DB::table('users')
        ->select('name', 'surname', 'email')
        ->where('id', $request->idUser)
        ->get();

    $totalValue = DB::table('users')
        ->join('carts', 'carts.idUser', '=', 'users.id')
        ->join('products', 'products.id', '=', 'carts.idProduct')
        ->join('groups', 'groups.id', '=', 'products.idGroup')
        ->where('users.id', $request->idUser)
        ->where('groups.name', $cartProducts[0]->group_name)
        ->sum(DB::raw('products.price * carts.quantity'));
```



```
$idcapoGruppo = DB::table('groups')
    ->select('idGroupLeader')
    ->where('name', $cartProducts[0]->group_name)
    ->get();

$capoGruppo = DB::table('users')
    ->select('name', 'surname', 'email')
    ->where('id', $idcapoGruppo[0]->idGroupLeader)
    ->get();

$dmpdf = new Dmpdf( );
$dmpdf->setPaper('A4', 'portrait');
$dmpdf->load_html( view('group.invioceBase',
    ["cartProducts"=>$cartProducts, 'user'=>$user,
    'capoGruppo'=>$capoGruppo, 'totalValue'=>$totalValue]) );
$dmpdf->render( );
$dmpdf->stream('Fattura_' . $cartProducts[0]->group_name . "_" .
    $user[0]->name . "_" . $user[0]->surname . '.pdf');

}
```

4.3.8 EmailController

4.3.8.1 Introduzione

La classe EmailController rappresenta il controller che gestisce la logica per quanto riguarda le email.

4.3.8.2 Implementazione

La parte più importante della classe è il metodo index() che si di mandare l'email ai capigruppo quando i termini 1 e 2 vengono superati. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.8.3 index

Questo metodo viene invocato ogni volta che un utente fa login. Da lì parte a controllare se dall'ultima volta che qualcuno ha fatto un login ci sono email da mandare inerenti al termine 1 e al termine 2.

```
public function index(){
    $currentDate = date("Y-m-d");
    $emailData = [
        [
            'deadlineColumn' => 'deadline1',
            'sendEmailColumn' => 'sendEmail1',
            'emailSubject' => 'Termine 1 scaduto'
        ],
        [
            'deadlineColumn' => 'deadline2',
            'sendEmailColumn' => 'sendEmail2',
            'emailSubject' => 'Termine 2 scaduto'
        ]
    ];

    foreach ($emailData as $data) {
        $emails = DB::table('users')
            ->join('groups', 'groups.idGroupLeader', '=', 'users.id')
            ->where($data['deadlineColumn'], '<', $currentDate)
            ->where($data['sendEmailColumn'], '=', null)
            ->select('users.email', 'groups.name', $data['sendEmailColumn'])
            ->get();

        foreach ($emails as $email) {
            $testMailData = [
                'title' => 'Gruppo ' . $email->name,
                'body' => $data['emailSubject']
            ];

            Mail::to($email->email)->send(new SendMail($testMailData));

            Group::where('name', $email->name)->update([
                $data['sendEmailColumn'] => 1
            ]);
        }
    }
}
```

4.3.9 UserController

4.3.9.1 Introduzione

La classe UserController rappresenta il controller che gestisce la logica per quanto riguarda l'aggiornamento dei dati dell'utente.

4.3.9.2 Implementazione

La parte più importante della classe è il metodo saveNewInfo() che si occupa di aggiornare le informazioni dell'utente. Vediamo dunque il cuore di questa classe nel dettaglio.

4.3.9.3 saveNewInfo

```
public function saveNewInfo(Request $request){
    $user =DB::table('users')
    ->select('*')
    ->where('id',Auth::user()->id)
    ->get();
    if($request->action == 1)
    {
        $existingEmail = User::where('id', $user[0]->id)->first();
        if ($existingEmail && $user[0]->email != $request->email) {
            return redirect()->back()->withErrors(['email' => 'Email gi
            utilizzata.']);
        }else{
            User::where('id', $user[0]->id)->update([
                'name' =>$request->name,
                'surname' =>$request->surname,
                'email' => $request->email,
            ]);
        }
    }
    return redirect('/');
}
```

4.4 View

Le view in Laravel sono i file che definiscono come appare il sito web. Sono come dei template delle pagine, dove si combinano HTML e codice PHP per mostrare il contenuto dinamicamente. I controller utilizzano le view per generare e visualizzare le pagine che gli utenti vedono nel browser.

4.4.1 layouts

Questa cartella contiene i layouts di default che verranno utilizzati all'interno del sito.

- **app:** template che viene utilizzato all'interno del sito una volta effettuato il login.
- **basic:** template che viene usato nella schermata di login e quella di registrazione.

4.4.2 homeUtente:

View che viene caricata non appena si fa il login nel sito.

4.4.3 group

Ho creato una cartella dove al suo interno sono contenute le seguenti view.

- **addProduct:** View che permette all'utente di aggiungere un nuovo prodotto.
- **change:** View che permette all'utente di cambiare gruppo.
- **changeInfo:** View che permette di cambiare le informazioni del gruppo come nome o date di termine.
- **create:** View dove creare il gruppo dove si sarà capogruppo.
- **editProduct:** View dove si può modificare il prodotto selezionato.
- **invoiceBase:** Template per le fatture.
- **invoice:** View dove si appaiono tutte le fatture degli utenti di quel gruppo.
- **list:** Lista dei gruppi esistenti e non ancora scaduti, con rispettivo stato della richiesta.
- **listParticipants:** View dove si può vedere chi sta partecipando al gruppo.
- **manager:** View per la gestione del gruppo, ci si può accedere solo se capigruppo.
- **request:** View dove si possono accettare le richieste di partecipazione al gruppo.

4.4.4 emailBase:

Template per le email da mandare.

4.4.5 userInfo:

View per le informazioni utente.

4.5 Route

Le route definiscono gli URL che gli utenti possono visitare per accedere a specifiche pagine o funzionalità. Le route associano gli URL alle azioni o ai metodi dei controller che gestiscono le richieste degli utenti e restituiscono le risposte corrispondenti.

4.5.1 Auth::routes()

Questa route è responsabile per la gestione dell'autenticazione dell'utente. Registra le route per il login, la registrazione, il logout e altre funzionalità correlate.

- `Route::get('/', [App::class, 'index'])→name('home')`
- `Route::get('/home', [App::class, 'index'])→name('home')`

Queste due route gestiscono la pagina iniziale dell'applicazione. Quando l'utente accede alla root del sito o visita `/home`, viene chiamato il metodo `index` del controller `HomeController` per visualizzare la pagina principale.

4.5.2 groupStart

Questa route gestisce la creazione di un nuovo gruppo. Quando l'utente accede a `/groupStart`, viene chiamato il metodo `start` del controller `GroupController` per avviare il processo di creazione del gruppo. È richiesta l'autenticazione per accedere a questa route.

4.5.3 groupCreate

Questa route gestisce la creazione effettiva di un nuovo gruppo. Quando l'utente invia i dati del nuovo gruppo tramite il form, viene chiamato il metodo `create` del controller `GroupController` per creare il gruppo. È richiesta l'autenticazione per accedere a questa route.

4.5.4 groupInfo

Questa route gestisce la visualizzazione delle informazioni del gruppo. Quando l'utente richiede le informazioni del gruppo, viene chiamato il metodo `info` del controller `GroupController` per recuperare e visualizzare le informazioni. È richiesta l'autenticazione per accedere a questa route.

4.5.5 groupManagerChanges

Questa route gestisce le modifiche al responsabile del gruppo. Quando l'utente invia il form con le modifiche al responsabile del gruppo, viene chiamato il metodo `managerChanges` del controller `GroupController` per effettuare le modifiche. È richiesta l'autenticazione per accedere a questa route.

4.5.6 pageManagerProduct

Questa route gestisce la visualizzazione della pagina di gestione dei prodotti. Quando l'utente richiede la pagina di gestione dei prodotti, viene chiamato il metodo `pageManagerProduct` del controller `ProductController` per visualizzare la pagina. È richiesta l'autenticazione per accedere a questa route.

4.5.7 visibleProduct

Questa route gestisce la modifica della visibilità di un prodotto. Quando l'utente modifica la visibilità di un prodotto tramite il tasto, viene chiamato il metodo `visibleProduct` del controller `ProductController` per salvare la modifica della visibilità del prodotto. È richiesta l'autenticazione per accedere a questa route.

4.5.8 addProduct

Questa route gestisce l'aggiunta di un nuovo prodotto. Quando l'utente clicca sul tasto per aggiungere un prodotto, viene chiamato il metodo "add" del controller "ProductController" per aggiungere il prodotto. È richiesta l'autenticazione per accedere a questa route.

4.5.9 addProductRequest

Questa route gestisce la richiesta di aggiunta di un prodotto. Quando l'utente invia la richiesta di aggiunta di un prodotto tramite il form, viene chiamato il metodo "addRequest" del controller "ProductController" per gestire la richiesta. È richiesta l'autenticazione per accedere a questa route.

4.5.10 editProduct

Questa route gestisce la modifica di un prodotto esistente. Quando l'utente seleziona un prodotto per modificarlo, viene chiamato il metodo "editProduct" del controller "ProductController" per visualizzare il modulo di modifica del prodotto. È richiesta l'autenticazione per accedere a questa route.

4.5.11 saveEditProduct

Questa route gestisce il salvataggio delle modifiche apportate a un prodotto. Quando l'utente invia il form con le modifiche del prodotto, viene chiamato il metodo "saveEditProduct" del controller "ProductController" per salvare le modifiche. È richiesta l'autenticazione per accedere a questa route.

4.5.12 listGroup

Questa route gestisce la visualizzazione della lista dei gruppi. Quando l'utente richiede la lista dei gruppi, viene chiamato il metodo "list" del controller "ParticipationController" per recuperare e visualizzare la lista dei gruppi. È richiesta l'autenticazione per accedere a questa route.

4.5.13 changeGroup

Questa route gestisce il cambio del gruppo attivo. Quando l'utente seleziona un gruppo diverso come gruppo attivo, viene chiamato il metodo "change" del controller "GroupController" per cambiare il gruppo attivo. È richiesta l'autenticazione per accedere a questa route.

4.5.14 requestAccessSend

Questa route gestisce l'invio di una richiesta di accesso a un gruppo. Quando l'utente invia la richiesta di accesso tramite il tasto, viene chiamato il metodo "requestAccessSend" del controller "ParticipationController" per inviare la richiesta. È richiesta l'autenticazione per accedere a questa route.

4.5.15 requestAccessPage

Questa route gestisce la visualizzazione della pagina di richiesta di accesso a un gruppo specifico. Quando l'utente accede a "/requestAccessPage/group", dove "group" è l'identificatore del gruppo, viene chiamato il metodo "requestAccessPage" del controller "ParticipationController" per visualizzare la pagina di richiesta di accesso al gruppo specifico. È richiesta l'autenticazione per accedere a questa route.

4.5.16 setParticipations

Questa route gestisce la configurazione delle partecipazioni ai gruppi. Quando l'utente invia la richiesta con le partecipazioni ai gruppi, viene chiamato il metodo "setParticipations" del controller "ParticipationController" per salvare le partecipazioni. È richiesta l'autenticazione per accedere a questa route.

4.5.17 listParticipants

Questa route gestisce la visualizzazione della lista dei partecipanti di un gruppo specifico. Quando l'utente richiede la lista dei partecipanti del gruppo tramite il form, viene chiamato il metodo "listParticipants" del controller "ParticipationController" per recuperare e visualizzare la lista dei partecipanti. È richiesta l'autenticazione per accedere a questa route.

4.5.18 block

Questa route gestisce il blocco di un partecipante. Quando l'utente blocca un partecipante tramite il tasto, viene chiamato il metodo "block" del controller "ParticipationController" per bloccare il partecipante. È richiesta l'autenticazione per accedere a questa route.

4.5.19 order

Questa route gestisce l'ordine di un prodotto. Quando l'utente invia l'ordine di un prodotto tramite il tasto, viene chiamato il metodo "order" del controller "ProductController" per gestire l'ordine. È richiesta l'autenticazione per accedere a questa route.

4.5.20 saveOrder

Questa route gestisce il salvataggio dell'ordine di un prodotto. Quando l'utente invia il tasto con l'ordine del prodotto, viene chiamato il metodo "saveOrder" del controller "ProductController" per salvare l'ordine. È richiesta l'autenticazione per accedere a questa route.

4.5.21 approveProducts

Questa route gestisce l'approvazione dei prodotti. Quando l'utente approva i prodotti tramite il tasto, viene chiamato il metodo "approveProduct" del controller "GroupController" per gestire l'approvazione dei prodotti. È richiesta l'autenticazione per accedere a questa route.

4.5.22 invoice

Questa route gestisce la visualizzazione di una fattura. Quando l'utente richiede la visualizzazione di una fattura tramite il tasto, viene chiamato il metodo "invoiceShow" del controller "InvoiceController" per visualizzare la fattura. È richiesta l'autenticazione per accedere a questa route.

4.5.23 createPDF

Questa route gestisce la creazione di un file PDF per la fattura. Quando l'utente crea il file PDF per la fattura tramite il tasto, viene chiamato il metodo "createPDF" del controller "InvoiceController" per generare il file PDF. È richiesta l'autenticazione per accedere a questa route.

4.5.24 user

Questa route gestisce la visualizzazione delle informazioni dell'utente. Quando l'utente richiede le proprie informazioni tramite il tasto, viene chiamato il metodo "getInfo" del controller "UserController" per recuperare e visualizzare le informazioni dell'utente. È richiesta l'autenticazione per accedere a questa route.

4.5.25 saveNewInfo

Questa route gestisce il salvataggio delle nuove informazioni dell'utente. Quando l'utente invia il form con le nuove informazioni, viene chiamato il metodo "saveNewInfo" del controller "UserController" per salvare le modifiche. È richiesta l'autenticazione per accedere a questa route.

5 Test

5.1 Protocollo di test

Test-01	
Nome	Controllo che il sito sia online
Riferimento	Req-08
Descrizione	Questo test verifica se si riesce a raggiungere il sito.
Prerequisiti	<ul style="list-style-type: none">• Avere una connessione a internet
Procedura	<ul style="list-style-type: none">• Scrivere il seguente link : https://progetto1.shop
Risultati attesi	Che il sito carichi la schermata di login.”

Test-02	
Nome	Possibilità di accedere al sito con un email non registrata
Riferimento	Req-01
Descrizione	Questo test verifica che si riesca ad accedere al sito web con un email non registrata.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante
Procedura	<ul style="list-style-type: none">• Aprire il sito .• Provare a inserire la propria email e la password• Cliccare il tasto accedi
Risultati attesi	Che il sito faccia non faccia il login e ritorni una schermata di errore.Dicendo ”Queste credenziali non corrispondono ai nostri registri.”

Test-03	
Nome	Possibilità di registrarsi al sito con un email non ancora registrata
Riferimento	Req-01
Descrizione	Questo test verifica se ci si riesce a registrare al sito web.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Non avere ancora mai usato email per la registrazione
Procedura	<ul style="list-style-type: none">• Aprire il sito• Cliccare in alto a destra Registrati• Compilare il form con informazioni valide e la propria email personale
Risultati attesi	Che il sito faccia login senza problemi e che venga caricata la schermata della lista dei gruppi

Test-04	
Nome	Controllo degli errori in caso di email già registrata
Riferimento	Req-01
Descrizione	Questo test verifica se ci si riesce a registrare al sito web con un email già registrata.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• essersi già registrati al sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Cliccare in alto a destra Registrati• Compilare il form con informazioni valide e la propria email che si é usato in precedenza
Risultati attesi	Che il sito non faccia la registrazione e sollevi un errore a schermo. Che dice email già in uso.

Test-05	
Nome	Creazione di un gruppo
Riferimento	Req-02
Descrizione	Questo test verifica se si riesce a creare un gruppo con un nome non ancora esistente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• essersi già registrati al sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce crea gruppo• Compilare il form con un nome che si sa che non esiste• Cliccare crea gruppo.
Risultati attesi	Che il sito ci rimandi alla pagina della lista dei gruppi e che sia presente il nostro gruppo.

Test-06	
Nome	Creazione di un gruppo già esistente
Riferimento	Req-02
Descrizione	Questo test verifica se si riesce a creare un gruppo con un nome già esistente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce crea gruppo• Compilare il form con un nome che si sa che esiste già.• Cliccare crea gruppo.
Risultati attesi	Che il sito ci mostri a schermo il seguente errore. Il nome del gruppo è già stato utilizzato.

Test-07	
Nome	Creazione di un gruppo mettendo date sbagliate per il termine1
Riferimento	Req-02
Descrizione	Questo test verifica se si riesce a creare un gruppo inserendo delle date antecedenti al giorno stesso per il termine1 .
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce crea gruppo• Compilare il form una data nel termine 1 antecedente a oggi.
Risultati attesi	Che il sito non ci faccia selezionare la data antecedente a oggi

Test-08	
Nome	Creazione di un gruppo mettendo date sbagliate per il termine2
Riferimento	Req-02
Descrizione	Questo test verifica se si riesce a creare un gruppo inserendo delle date antecedenti al termine1 per il termine2 .
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce crea gruppo• Compilare il nome e il termine1• Compilare il termine2 una data precedente al termine1
Risultati attesi	Che il sito non ci faccia selezionare la data antecedente al termine1

Test-09	
Nome	Creazione di un gruppo mettendo date sbagliate per il termine2
Riferimento	Req-02
Descrizione	Questo test verifica se si riesce a creare un gruppo inserendo delle date antecedenti al termine1 per il termine2 .
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce crea gruppo• Compilare il termine2 una data precedente al termine1• Compilare il nome e il termine1
Risultati attesi	Che il sito dia un errore dicendo che la data deve essere successiva al termine1

Test-10	
Nome	Essere capogruppo dopo la creazione di un gruppo
Riferimento	Req-03
Descrizione	Questo test verifica se dopo la creazione di un gruppo si é davvero capi gruppo.
Prerequisiti	<ul style="list-style-type: none"> • Avere il sito online • Avere il database funzionate • Aver fatto login nel sito • Aver creato un gruppo con l'utente con qui si fa login
Procedura	<ul style="list-style-type: none"> • Aprire il sito • Fare login • Selezionare la voce Gruppi Partecipanti • Selezionare la voce gestisci gruppo
Risultati attesi	Che il sito carichi la schermata di capogruppo del gruppo cliccato

Test-11	
Nome	Non essere capogruppo di un gruppo non creato da noi
Riferimento	Req-03
Descrizione	Questo test verifica se si é capi gruppo solo dei gruppi creati da noi.
Prerequisiti	<ul style="list-style-type: none"> • Avere il sito online • Avere il database funzionate • Aver fatto login nel sito • Partecipare a un gruppo creato da qualcun'altro
Procedura	<ul style="list-style-type: none"> • Aprire il sito • Fare login • Selezionare la voce Gruppi Partecipanti
Risultati attesi	Che il sito mostri solo la voce ordina per i gruppi non creati da noi

Test-12	
Nome	Che si possa mandare la richiesta a un gruppo dove non si fa parte
Riferimento	Req-03
Descrizione	Questo test verifica se si può iscriversi a un gruppo a qui ancora non si partecipa.
Prerequisiti	<ul style="list-style-type: none"> • Avere il sito online • Avere il database funzionate • Aver fatto login nel sito • Che siano presenti gruppi a qui non si é partecipante o si ha gia mandato la richiesta
Procedura	<ul style="list-style-type: none"> • Aprire il sito • Fare login • Selezionare la voce Lista Gruppi • Mandare la richiesta a un gruppo della lista in qui il tasto é blu e c'é scritto richiesta di accesso.
Risultati attesi	Che il tasto una volta premuto cambi di stato e diventi giallo con scritto in attesa.

Test-13	
Nome	Che si possa mandare la richiesta a un gruppo dove non si fa parte
Riferimento	Req-05
Descrizione	Questo test verifica se si può mandare la richiesta a un gruppo a qui ancora non si partecipa.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Che siano presenti gruppi a qui non si è partecipante o si ha già mandato la richiesta
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Lista Gruppi• Mandare la richiesta a un gruppo della lista in cui il tasto è blu e c'è scritto richiesta di accesso.
Risultati attesi	Che il tasto una volta premuto cambi di stato e diventi giallo con scritto in attesa.

Test-14	
Nome	Che si non possa mandare molteplici richieste a un gruppo dove non si fa parte
Riferimento	Req-05
Descrizione	Questo test verifica se si può mandare molteplici richieste a un gruppo a qui ancora non si partecipa.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Aver già mandato una richiesta non ancora accettata
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Lista Gruppi
Risultati attesi	Che dove ho già mandato la richiesta di partecipazione ci sia scritto in attesa.

Test-15	
Nome	Che si riesca a accettare la richiesta di partecipazione di un utente
Riferimento	Req-04
Descrizione	Questo test verifica se si riesce a accettare una richiesta di un utente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di almeno 1 gruppo• Avere una richiesta in attesa
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce gruppi partecipanti• Selezionare il gruppo dove abbiamo fatto arrivare la richiesta cliccando su gestisci gruppo• Cliccare in alto a destra gestisci gruppo.• Cliccare sull'icona sulla riga dove c'è scritto richieste seguito da un numero.• Cliccare accetta sulla richiesta scelta.
Risultati attesi	Che cliccando partecipanti si veda la persona appena accettata.

Test-16	
Nome	Che si riesca a rifiutare la richiesta di partecipazione di un utente
Riferimento	Req-04
Descrizione	Questo test verifica se si riesce a rifiutare una richiesta di un utente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di almeno 1 gruppo• Avere una richiesta in attesa
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce gruppi partecipanti• Selezionare il gruppo dove abbiamo fatto arrivare la richiesta cliccando su gestisci gruppo• Cliccare in alto a destra gestisci gruppo.• Cliccare sull'icona sulla riga dove c'è scritto richieste seguito da un numero.• Cliccare rifiuta sulla richiesta scelta.
Risultati attesi	Che cliccando partecipanti non si veda la persona appena rifiuta.

Test-17	
Nome	Che nella lista dei gruppi non siano presenti gruppi scaduti prima di oggi
Riferimento	Req-06
Descrizione	Questo test verifica se nella lista dei gruppi non si vedono i gruppi scaduti.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Che sia stato creato un gruppo già terminato non dall'utente loggato
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Lista Gruppi
Risultati attesi	Che non sia presente il gruppo in questione.

Test-18	
Nome	Che si riesca a aggiungere prodotti nel gruppo se si é capogruppo
Riferimento	Req-09
Descrizione	Questo test verifica se si riesce a aggiungere un prodotto all'interno di un gruppo dove si é capogruppo.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di un gruppo
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare il gruppo in qui si sa che si é capi gruppo• Cliccare il pulsante aggiungi prodotto• Compilare tutti i campi e aggiungere il prodotto
Risultati attesi	Che il sito aggiunga il prodotto e lo mostri nella lista dei prodotti del gruppo.

Test-19	
Nome	Tentare di mettere un prezzo non valido
Riferimento	Req-09
Descrizione	Questo test verifica se si riesce a inserire un prezzo non valido.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di un gruppo
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare il gruppo in cui si sa che si è capi gruppo• Cliccare il pulsante aggiungi prodotto• Compilare il campo del prezzo con delle lettere
Risultati attesi	Che il sito non prenda le lettere e lasci in bianco il campo

Test-20	
Nome	Modifica di un prodotto esistente
Riferimento	Req-10
Descrizione	Questo test verifica se si riesce a modificare un prodotto esistente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di un gruppo• Avere un prodotto nel gruppo
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare il gruppo in cui si sa che si è capi gruppo• Cliccare il pulsante modifica del prodotto• Compilare modificare un campo a propria scelta• Cliccare su salva
Risultati attesi	Che il prodotto venga modificato e mostrato a schermo

Test-21	
Nome	Annullamento delle modifica di un prodotto esistente
Riferimento	Req-10
Descrizione	Questo test verifica se si riesce a modificare un prodotto esistente.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di un gruppo• Avere un prodotto nel gruppo
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare il gruppo in cui si sa che si è capi gruppo• Cliccare il pulsante modifica del prodotto• Compilare modificare un campo a propria scelta• Cliccare su annulla
Risultati attesi	Che il prodotto non venga modificato e a schermo ci siano le vecchie info del prodotto

Test-22	
Nome	Che si possa essere partecipante in più gruppi
Riferimento	Req-13
Descrizione	Questo test verifica se si può essere partecipanti di più gruppi
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• che siano state accettate almeno 2 richieste di partecipazione
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti
Risultati attesi	Che nella lista io possa ordinare da tutti i gruppi presenti non scaduti.

Test-23	
Nome	Aggiunta nel carrello di un prodotto
Riferimento	Req-15
Descrizione	Questo test verifica se si riesce a aggiungere dei prodotti al carrello.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Partecipare a un gruppo con almeno 1 prodotto
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare ordine sul gruppo in cui si sa che c'è un prodotto.• Modificare la quantità del prodotto rispettando i criteri del prodotto• Cliccare salva
Risultati attesi	Che appaia a schermo la scritta verde

Test-24	
Nome	test errore aggiunta nel carrello
Riferimento	Req-15
Descrizione	Questo test verifica se il sito solleva l'errore in caso che si sbaglia a inserire la quantità.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Partecipare a un gruppo con almeno 1 prodotto
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare ordine sul gruppo in cui si sa che c'è un prodotto.• Modificare la quantità del prodotto inserendo un valore inferiore alla quantità minima• Cliccare salva
Risultati attesi	Che appaia a schermo la scritta rossa

Test-25	
Nome	Fatture per utente
Riferimento	Req-07
Descrizione	Questo test verifica se si riesce a scaricare la fattura per l'utente dopo che ha messo qualcosa nel carrello.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Aver effettuato il test-23• Essere capogruppo del gruppo in questione
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare la voce gestisci gruppo• Cliccare il pulsante fattura• Cliccare il tasto scarica a fianco del nome utente
Risultati attesi	Che la fattura venga scaricata e sia in pdf

Test-26	
Nome	Vedere la lista degli utenti
Riferimento	Req-18
Descrizione	Questo test verifica se si riesce a vedere la lista dei partecipanti di un gruppo.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di almeno un gruppo
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare la voce gestisci gruppo• Cliccare il pulsante fattura• Cliccare sull'icona della persona sulla riga dei partecipanti
Risultati attesi	Che appaia una lista con tutti i partecipanti al gruppo

Test-27	
Nome	Scadenza termine2 non si può più modificare la quantità
Riferimento	Req-15
Descrizione	Questo test verifica se una volta scaduto il termine2 non si possa più modificare la quantità come partecipante del gruppo.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere partecipante di almeno un gruppo scaduto con prodotti
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare la voce ordina
Risultati attesi	Che appaia a lato di tutti i prodotti la scritta attesa e non più salva

Test-28	
Nome	Scadenza termine1 non si può più modificare la quantità
Riferimento	Req-16
Descrizione	Questo test verifica se una volta scaduto il termine1 non si possa più modificare la quantità come partecipante del gruppo. Finché non vengono sbloccati i carrelli dal capogruppo.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere partecipante di almeno un gruppo scaduto con prodotti
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Selezionare la voce Gruppi Partecipanti• Selezionare la voce ordina
Risultati attesi	Che appaia a lato di tutti i prodotti la scritta attesa e non più salva

Test-29	
Nome	Ricevere le email quando scade un termine
Riferimento	Req-19
Descrizione	Questo test verifica se una volta scaduto un termine arriva un email al capogruppo avvertendolo.
Prerequisiti	<ul style="list-style-type: none">• Avere il sito online• Avere il database funzionante• Aver fatto login nel sito• Essere capogruppo di un gruppo con un termine scaduto
Procedura	<ul style="list-style-type: none">• Aprire il sito• Fare login• Controllare le email
Risultati attesi	Che sia presente un email dal sito dove ti avverete che un termine é scaduto dicendo anche che gruppo.

5.2 Risultati test

ID	Risultato	Data di test	Note
Test-01	Passato	25.05.2023	
Test-02	Passato	25.05.2023	
Test-03	Passato	25.05.2023	
Test-04	Passato	25.05.2023	
Test-05	Passato	25.05.2023	
Test-06	Passato	25.05.2023	
Test-07	Passato	25.05.2023	
Test-08	Passato	25.05.2023	
Test-09	Passato	25.05.2023	
Test-10	Passato	25.05.2023	
Test-11	Passato	25.05.2023	
Test-12	Passato	25.05.2023	
Test-13	Passato	25.05.2023	
Test-14	Passato	25.05.2023	
Test-15	Passato	25.05.2023	
Test-16	Passato	25.05.2023	
Test-17	Passato	25.05.2023	
Test-18	Passato	25.05.2023	
Test-19	Passato	25.05.2023	
Test-20	Passato	25.05.2023	
Test-21	Passato	25.05.2023	
Test-22	Passato	25.05.2023	
Test-23	Passato	25.05.2023	
Test-24	Passato	25.05.2023	
Test-25	Passato	25.05.2023	
Test-26	Passato	25.05.2023	
Test-27	Passato	25.05.2023	
Test-28	Passato	25.05.2023	
Test-29	Passato	25.05.2023	

6 Consuntivo

Quello che si trova qui sotto é il GANTT consuntivo del progetto:

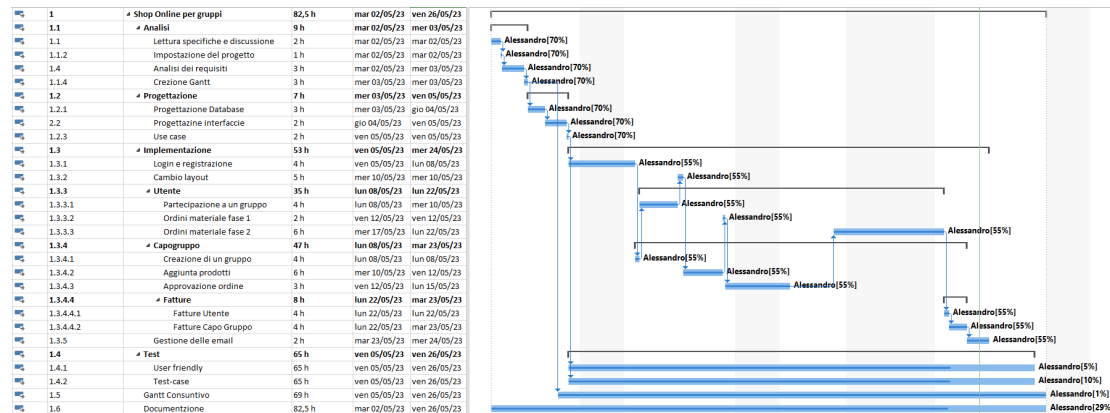


Figura 19: Gantt consuntivo

Come possiamo notare il GANTT consuntivo è cambiato leggermente rispetto al preventivo. L'attività che cambiato l'ordine delle attività è stato che ho dovuto rifare fare un cambio di layout e ciò ha portato via tempo dato che ho dovuto modificare codice già scritto e interfacce già progettate. Le due attività che alla fine mi hanno portato via più tempo sono state l'approvazione degli ordini e la gestione nella fase due dei prodotti. Nonostante in realtà la fase due delle ordinazioni a livello pratico non ha così tanta differenza. Ma la gestione dei prodotti che possono essere nascosti e la gestione delle regole per delle ordinazioni, aggiunte a qualche problema incontrato hanno portato a essere una parte più grande. Inoltre è cambiata anche leggermente la struttura dell'implementazione dato che ho deciso di spostare prima alcune attività e altre dopo.

Per avere una visione più dettagliata della pianificazione, ho allegato la pianificazione completa in formato A3 alla fine della documentazione.

7 Conclusioni

7.1 Sviluppi futuri

Per gli sviluppi futuri si potrebbe aggiungere i log dato che attualmente non sono presenti. Mandare più email magari mandare automaticamente le fatture via email e implementare una creazione di Qr per pagare le fatture. Un altro sviluppo futuro potrebbe quello di farlo responsive così da poterlo usare anche sui telefoni. Inoltre come sviluppo futuri si potrebbe migliorare la gestione degli errori quando si prova a cambiare il nome del gruppo o la data con informazioni sbagliate.

7.2 Considerazioni personali

Il progetto su cui ho lavorato mi ha permesso di ampliare le mie conoscenze ma anche conoscere molte nuove cose. Soprattutto perché non avevo mai sviluppato in Laravel un progetto così grande. È stato molto interessante anche dover capire come creare dei PDF dato che non l'avevo mai fatto finora. Se devo essere sincero quando ho ricevuto il progetto ero un po' spaventato perché avevamo poco tempo e come detto non avevo mai usato Laravel; Ma in fondo non mi sono trovato così male per fortuna avevo già fatto dei piccoli progetti web per prepararmi. Sicuramente per colpa della mia inesperienza ci sono delle cose che si potevano gestire meglio. Ma mi ritengo comunque soddisfatto di ciò che sono riuscito a progettare e sviluppare.

8 Sitografia

Sitografia

- [1] dbdiagram. *Tool per diagramma db*. 25.05.2023. URL: <https://dbdiagram.io/>.
- [2] dompdf. *Documentazione dompdf*. 22.05.2023. URL: <https://github.com/dompdf/dompdf>.
- [3] laravel. *Documentazione laravel*. 25.05.2023. URL: <https://laravel.com/>.
- [4] overleaf. *Progetto latex*. 26.05.2023. URL: <https://www.overleaf.com>.
- [5] stackoverflow. *Consultazione errori*. 22.05.2023. URL: <https://stackoverflow.com/>.
- [6] w3schools. *Consultazione per Html e Css*. 24.05.2023. URL: <https://www.w3schools.com/>.

9 Glossario

Terminologia	Descrizione
Blade	Blade è il motore di template predefinito di Laravel. Offre una sintassi semplice e potente per la generazione di HTML. I template Blade consentono di separare la logica di presentazione dal resto dell'applicazione.
Chiave primaria	Una colonna o un gruppo di colonne che identificano univocamente ogni riga in una tabella
Colonna	Una singola unità di dati all'interno di una tabella, come un nome o un valore numerico
Laravel	Un framework PHP open-source per lo sviluppo di applicazioni web.
Middleware	Il middleware in Laravel è uno strato intermedio tra le richieste HTTP e le risposte. È possibile utilizzare il middleware per filtrare, manipolare o verificare le richieste prima che vengano gestite dai controller.
PHP	PHP è un linguaggio di scripting ampiamente utilizzato per lo sviluppo di applicazioni web. Laravel è scritto in PHP e richiede una conoscenza di base di PHP per utilizzarlo efficacemente.
QdC	Quaderno dei Compiti: il documento consegnato dal docente responsabile che descrive le specifiche del progetto.
Query	Una richiesta per ottenere dati da un database
Routing	Il routing definisce come le richieste HTTP vengono indirizzate alle azioni appropriate all'interno dell'applicazione. In Laravel, è possibile definire i percorsi delle URL e associarli a metodi nei controller per gestire le richieste.
Tabella	Una struttura di dati all'interno di un database relazionale che organizza i dati in righe e colonne

10 Elenco Immagini

Elenco delle figure

1	Schema Use Case	11
2	Gantt preventivo	12
3	Design del Database	13
4	Design del Login	14
5	Design della Registrazione	15
6	Design Home prima versione	16
7	Design Home versione due	16
8	Design della schermata dei gruppi	17
9	Design creazione gruppo seconda versione	17
10	Design della schermata della lista dei gruppi esistenti.	18
11	Design della schermata della gestione del gruppo.	19
12	Design della schermata della gestione del gruppo seconda versione.	19
13	Design della schermata per l'aggiunta dei prodotti.	20
14	Design della schermata per l'aggiunta dei prodotti versione due.	20
15	Design della schermata per la modifica dei prodotti.	21
16	Design della schermata per la modifica dei prodotti versione due.	21
17	Design della schermata per le fatture.	22
18	Design della schermata per le fatture versione due.	22
19	Gantt consuntivo	54

11 Allegati

- Gantt preventivo
- Gantt consuntivo
- Schema Database
- Qdc
- Diari