POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Progetto TIW versione RIA

Progetto di TIW
Ingegneria Informatica

Studente: **Yanfeng Li**

Codice Persona: 10650552
Prof: Prof. Fraternali Piero, Prof. Federico Milani
Anno Accademico: 2021-2022

Contents

1 | Introduzione

1.1. Specifiche

Esercizio 2: Gestione Preventivi

Un'applicazione web consente la gestione di richieste di preventivi per prodotti personalizzati. L'applicazione supporta registrazione e login di clienti e impiegati mediante una pagina pubblica con opportune form. La registrazione controlla l'unicità dello username. Un preventivo è associato a un prodotto, al cliente che l'ha richiesto e all'impiegato che l'ha gestito. Il preventivo comprende una o più opzioni per il prodotto a cui è associato, che devono essere tra quelle disponibili per il prodotto. Un prodotto ha un codice, un'immagine e un nome. Un'opzione ha un codice, un tipo ("normale", "in offerta") e un nome. Un preventivo ha un prezzo, definito dall'impiegato. Quando l'utente (cliente o impiegato) accede all'applicazione, appare una LOGIN PAGE, mediante la quale l'utente si autentica con username e password. Quando un cliente fa login, accede a una pagina HOME PAGE CLIENTE che contiene una form per creare un preventivo e l'elenco dei preventivi creati dal cliente. Selezionando uno dei preventivi il cliente ne visualizza i dettagli. Mediante la form di creazione di un preventivo l'utente per prima cosa sceglie il prodotto; scelto il prodotto, la form mostra le opzioni di quel prodotto. L'utente sceglie le opzioni (almeno una) e conferma l'invio del preventivo mediante il bottone INVIA PREVENTIVO. Quando un impiegato effettua il login, accede a una pagina HOME PAGE IMPIEGATO che contiene l'elenco dei preventivi gestiti da lui in precedenza e quello dei preventivi non ancora associati a nessun impiegato. Quando l'impiegato seleziona un elemento dall'elenco dei preventivi non ancora associati a nessuno, compare una pagina PREZZA PREVENTIVO che mostra i dati del cliente (username) e del preventivo e una form per inserire il prezzo del preventivo. Quando l'impiegato inserisce il prezzo e invia i dati con il bottone INVIA PREZZO, compare di nuovo la pagina HOME PAGE IMPIEGATO con gli elenchi dei preventivi aggiornati. Il prezzo definito dall'impiegato risulta visibile al cliente quando questi accede all'elenco dei propri preventivi e visualizza i dettagli del preventivo. La pagina PREZZA PREVENTIVO contiene anche un collegamento per tornare alla HOME PAGE IMPIEGATO. L'applicazione consente il logout dell'utente

Si realizzi un'applicazione client server web che modifica le specifiche precedenti come segue: La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. Dopo il login, l'intera applicazione è realizzata con un'unica pagina per ciascuno dei ruoli: una pagina singola per il ruolo di cliente e una pagina singola per il ruolo di impiegato. Ogni interazione dell'utente è gestita senza ricaricare completamente la pagina, ma produce l'invocazione asincrona del server e l'eventuale modifica del contenuto da aggiornare a seguito dell'evento. Nella pagina del cliente, la scelta del prodotto comporta la successiva visualizzazione delle opzioni senza produrre un'ulteriore chiamata al server. L'invio del preventivo da parte del cliente deve produrre la verifica dei dati anche a lato client (almeno un'opzione scelta). Nella pagina dell'impiegato, il controllo del prezzo (non nullo e maggiore di zero) deve essere fatto anche a lato client. Eventuali errori a lato server devono essere segnalati mediante un messaggio di allerta all'interno della pagina del cliente o dell'impiegato.

2 | Database

2.1. Analisi Testuale

1. Entities
2. Attributes
3. Relationships

Un'applicazione web consente la gestione di richieste di preventivi per prodotti personalizzati. L'applicazione supporta registrazione e login di clienti e impiegati mediante una pagina pubblica con opportune form. La registrazione controlla l'unicità dello username. Un preventivo è associato a un prodotto, al cliente che l'ha richiesto e all'impiegato che l'ha gestito. Il preventivo comprende una o più opzioni per il prodotto a cui è associato, che devono essere tra quelle disponibili per il prodotto. Un prodotto ha un codice, un'immagine e un nome. Un'opzione ha un codicecodice, un tipo ("normale", "in offerta") e un nome. Un preventivo ha un prezzo, definito dall'impiegato. Quando l'utente (cliente o impiegato) accede all'applicazione, appare una LOGIN PAGE, mediante la quale l'utente si autentica con username e password. Quando un cliente fa login, accede a una pagina HOME PAGE CLIENTE che contiene una form per creare un preventivo e l'elenco dei preventivi creati dal cliente. Selezionando uno dei preventivi il cliente ne visualizza i dettagli. Mediante la form di creazione di un preventivo l'utente per prima cosa sceglie il prodotto; scelto il prodotto, la form mostra le opzioni di quel prodotto. L'utente sceglie le opzioni (almeno una) e conferma l'invio del preventivo mediante il bottone INVIA PREVENTIVO. Quando un impiegato effettua il login, accede a una pagina HOME PAGE IMPIEGATO che contiene l'elenco dei preventivi gestiti da lui in precedenza e quello dei preventivi non ancora associati a nessun impiegato. Quando l'impiegato seleziona un elemento dall'elenco dei preventivi non ancora associati a nessuno, compare una pagina PREZZA PREVENTIVO che mostra i dati del cliente (username) e del preventivo e una form per inserire il prezzo del preventivo. Quando l'impiegato inserisce il prezzo e invia i dati con il bottone INVIA PREZZO, compare di nuovo la pagina HOME PAGE IMPIEGATO con gli elenchi dei preventivi aggiornati. Il prezzo definito dall'impiegato risulta

visibile al cliente quando questi accede all'elenco dei propri preventivi e visualizza i dettagli del preventivo. La pagina PREZZA PREVENTIVO contiene anche un collegamento per tornare alla HOME PAGE IMPIEGATO. L'applicazione consente il logout dell'utente

2.2. Database Design

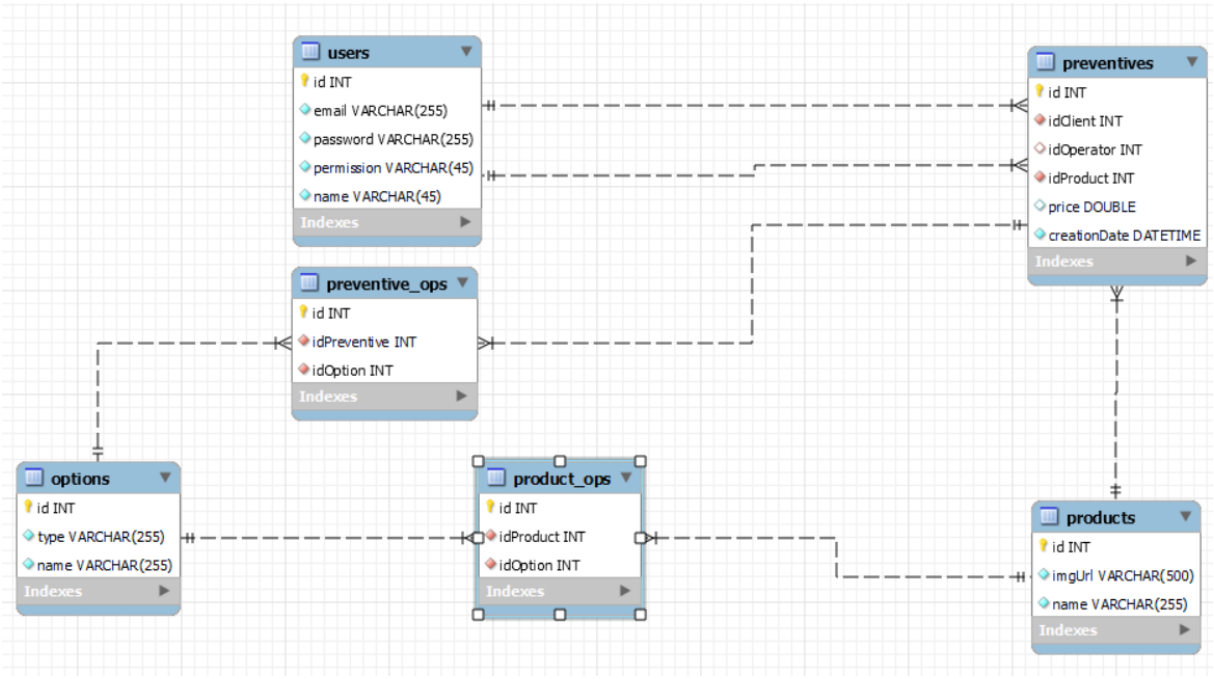

Figure 2.1: Diagramma Entita relazionale

2.3. Database Schema

2.3.1. Users

```
CREATE TABLE `users` (
  `id` int NOT NULL AUTO_INCREMENT,
  `email` varchar(255) NOT NULL,
  `password` varchar(255) NOT NULL,
  `permission` varchar(45) NOT NULL,
  `name` varchar(45) NOT NULL,
  PRIMARY KEY (`id`),
  UNIQUE KEY `email_UNIQUE` (`email`)
) ENGINE=InnoDB AUTO_INCREMENT=12 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.2: Users

2.3.2. Preventives

```
CREATE TABLE `preventives` (
  `id` int NOT NULL AUTO_INCREMENT,
  `idClient` int NOT NULL,
  `idOperator` int DEFAULT NULL,
  `idProduct` int NOT NULL,
  `price` double DEFAULT NULL,
  `creationDate` datetime NOT NULL DEFAULT CURRENT_TIMESTAMP,
  PRIMARY KEY (`id`),
  KEY `idOperator_idx` (`idOperator`),
  KEY `id_product_idx` (`idProduct`),
  KEY `idClient_idx` (`idClient`),
  CONSTRAINT `idClient` FOREIGN KEY (`idClient`) REFERENCES `users` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `idOperator` FOREIGN KEY (`idOperator`) REFERENCES `users` (`id`) ON UPDATE CASCADE,
  CONSTRAINT `idProduct` FOREIGN KEY (`idProduct`) REFERENCES `products` (`id`) ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=34 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.3: Preventives

2.3.3. Products

```
CREATE TABLE `products` (
  `id` int NOT NULL AUTO_INCREMENT,
  `imgUrl` varchar(500) NOT NULL,
  `name` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=14 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.4: Products

2.3.4. Options

```
CREATE TABLE `options` (
  `id` int NOT NULL AUTO_INCREMENT,
  `type` varchar(255) NOT NULL,
  `name` varchar(255) NOT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.5: Options

2.3.5. Preventive_ops

```
CREATE TABLE `preventive_ops` (
  `id` int NOT NULL AUTO_INCREMENT,
  `idPreventive` int NOT NULL,
  `idOption` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `idPreventive_idx` (`idPreventive`),
  KEY `idOp_idx` (`idOption`),
  CONSTRAINT `idOp` FOREIGN KEY (`idOption`) REFERENCES `options` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `idPreventive` FOREIGN KEY (`idPreventive`) REFERENCES `preventives` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=51 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.6: Preventive_ops

2.3.6. Product_ops

```
CREATE TABLE `product_ops` (
  `id` int NOT NULL AUTO_INCREMENT,
  `idProduct` int NOT NULL,
  `idOption` int NOT NULL,
  PRIMARY KEY (`id`),
  KEY `id_product_idx` (`idProduct`),
  KEY `idOption_idx` (`idOption`),
  CONSTRAINT `id_product` FOREIGN KEY (`idProduct`) REFERENCES `products` (`id`) ON DELETE CASCADE ON UPDATE CASCADE,
  CONSTRAINT `idOption` FOREIGN KEY (`idOption`) REFERENCES `options` (`id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoDB AUTO_INCREMENT=5 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci;
```

Figure 2.7: Product_ops

3 | Applicazione

3.1. Analisi Requisiti Applicazione

1. Pages (view)
2. view components
3. events
4. actions

Si realizzi un'applicazione client server web che modifica le specifiche precedenti come segue: L'applicazione supporta registrazione e login mediante una pagina pubblicauna pagina pubblica con opportune form. La registrazione controlla la validità sintattica dell'indirizzo di email e l'uguaglianza tra i campi "password" e "ripeti password", anche a lato client. Dopo il login, l'intera appli- cazione è realizzata con un'unica pagina per ciascuno dei ruoli: una pagina singola per il ruolo di cliente e una pagina singola per il ruolo di impiegato. Ogni interazione dell'utente è gestita senza ricaricare completamente la pagina, ma produce l'invocazione asincrona del server e l'eventuale modifica del contenuto da aggiornare a seguito dell'evento. Nella pag- ina del cliente, la scelta del prodotto comporta la successiva visualizzazione delle opzioni senza produrre un'ulteriore chiamata al server. L'invio del preventivo da parte del cliente deve produrre la verifica dei dati anche a lato client (almeno un'opzione scelta). Nella pagina dell'impiegato, il controllo del prezzo (non nullo e maggiore di zero) deve essere fatto anche a lato client. Eventuali errori a lato server devono essere segnalati mediante un messaggio di allerta all'interno della pagina del cliente o dell'impiegato.

4 | Componenti di Server_side

4.1. Model Objects (Beans)

- User
- Preventive
- Product
- Option

4.2. Data Access Objects (DAO)

- UserDAO
  - findUserByCredentials(email, password) : User
  - signupUser(name, email, password, permission) : void
  - getUserByEmail(email) : User
- ClientDAO
  - getPreventivesByUserId() : List<Preventive>
  - sendPreventive(productId, options) : void
- AdminDAO
  - getAdminPreventives() : List<Preventive>
  - getAllAvailablePreventive() : List<Preventive>
  - sendPreventive(preventiveId, price) : void
- PreventiveDAO
  - getPreventiveById() : Preventive
  - getOptionsById() : List<Option>

- ProductDao
  - getAllProduct() : List<Product>
  - getProductById(productId) : Product
  - getProductOptions(productId) : List<Option>

4.3. Filters

- CheckAdmin
- CheckClient
- CheckNotLoggedUser
- CheckLoggedUser

4.4. Controllers (servlets) [access rights]

- Login [Not Logged User]
- Logout [ALL]
- Signup [Not Logged User]
- SendPreventivo [User, Client]
- SendPrezzo [User, Admin]
- GetClientPreventivoList [User, Client]
- GetProductList [User]
- GetAdminAvailableList [User, Admin]
- GetAdminPreventivoList [User, Admin]
- GetPreventivoPrezza [User, Admin]
- GetPreventivoDetails [User]

4.5. Views (Templates)

- index.html

- home$_c$*lient.html*home$_a$*dmin.html*

5 | Componenti di Client_Side

5.1. Index

- Login form
  - Gestisce invio dei dati e eventuali errori
- Signup form
  - Gestisce invio dei dati e eventuali errori

5.2. Home_client

- PageOrchestrator
  - start() : Inizializza le componenti di pagina
  - refresh() : Aggiorna i contenuti delle componenti
  - hide() : Nasconde le componenti
  - handleError(error): Mostra il messaggio di errore
- UserInfo
  - show() : carica i dati di user e li rende visibili e abilita i pulsanti per logout e indietro
- PreventivoDetail
  - show(): Scarica i dati di un preventivo dando id
  - update(): Carica i dati e li rende visibili
  - hide()
- PreventivoList
  - show(): Scarica una lista dei preventivi

  - update(): Costruisce una lista di componenti e carica i dati e li rende visibili ()

- ProductList
  - show(): Scarica una lista dei product
  - update(): Costruisce una lista di componenti e carica i dati e li rende visibili
  - hide()
- ErrorDisplay
  - show(errorContent)
  - hide()

5.3. Home_admin

- PageOrchestrator
  - start() : Inizializza le componenti di pagina
  - refresh() : Aggiorna i contenuti delle componenti
  - hide() : Nasconde le componenti
  - handleError(error): Mostra il messaggio di errore
- UserInfo
  - show() : carica i dati di user e li rende visibili e abilita i pulsanti per logout e indietro
- PreventivoDetail
  - show(): Scarica i dati di un preventivo dando id
  - update(): Carica i dati e li rende visibili
  - hide()
- PreventivoList
  - show(): Scarica una lista dei preventivi
  - update(): Costruisce una lista di componenti e carica i dati e li rende visibili ()

- AvailableList
  - show(): Scarica una lista dei preventivi
  - update(): Costruisce una lista di componenti e carica i dati e li rende visibili
  - hide()
- prezzaPreventivo
  - show(): Scarica i dati di un preventivo dando id
  - update(): Carica i dati e li rende visibili
  - hide()
- ErrorDisplay
  - show(errorContent)
  - hide()

6 | Design Applicativo(IFML)

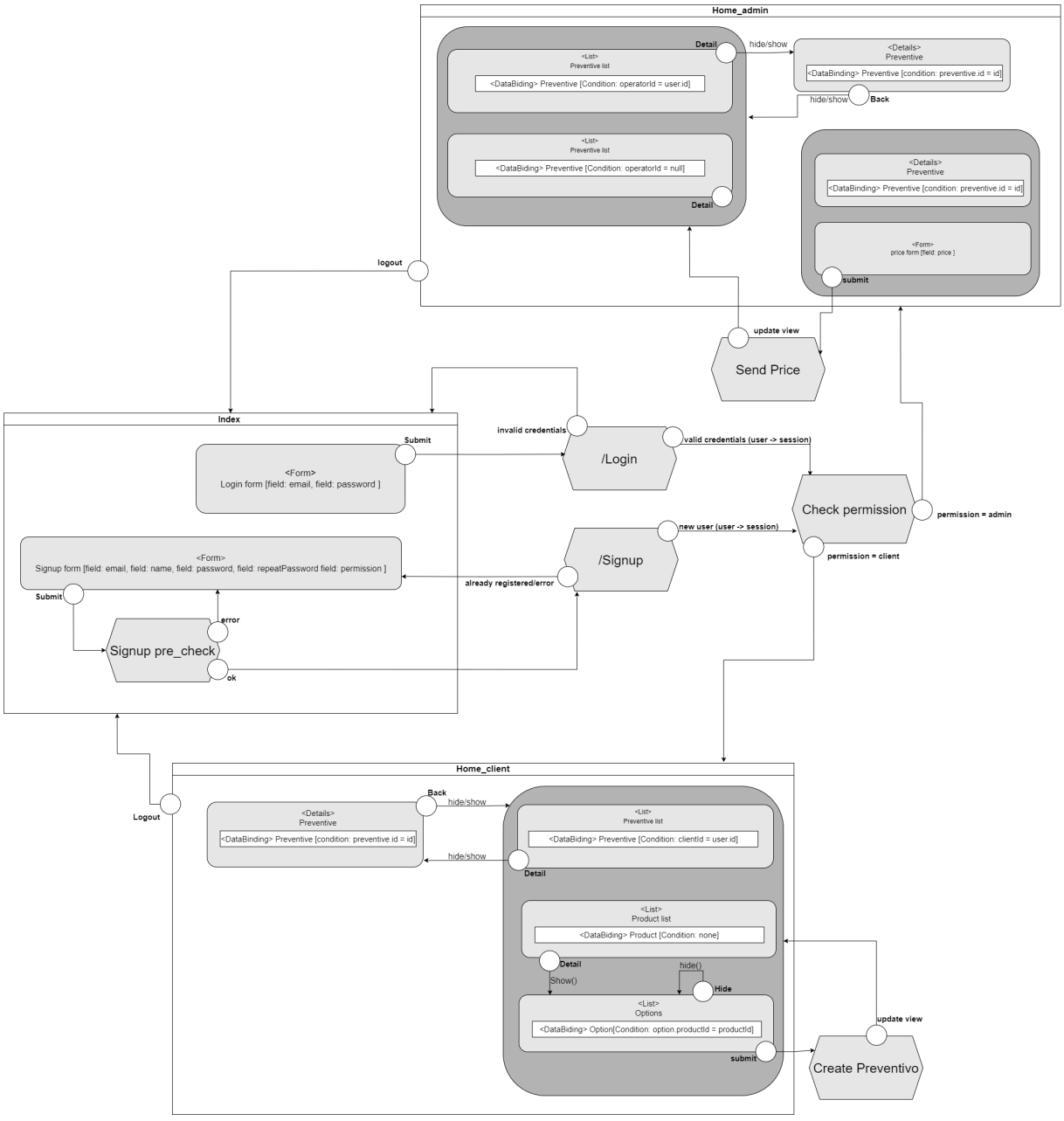

Figure 6.1: Design Applicativo(IFML)

7 | Sequence Diagrams di Interazioni

Molti interazioni sono simili a quelli versione puramente html, qui vengono riportati una parte di quelli di RIA.

7.1. CheckLoggedUser

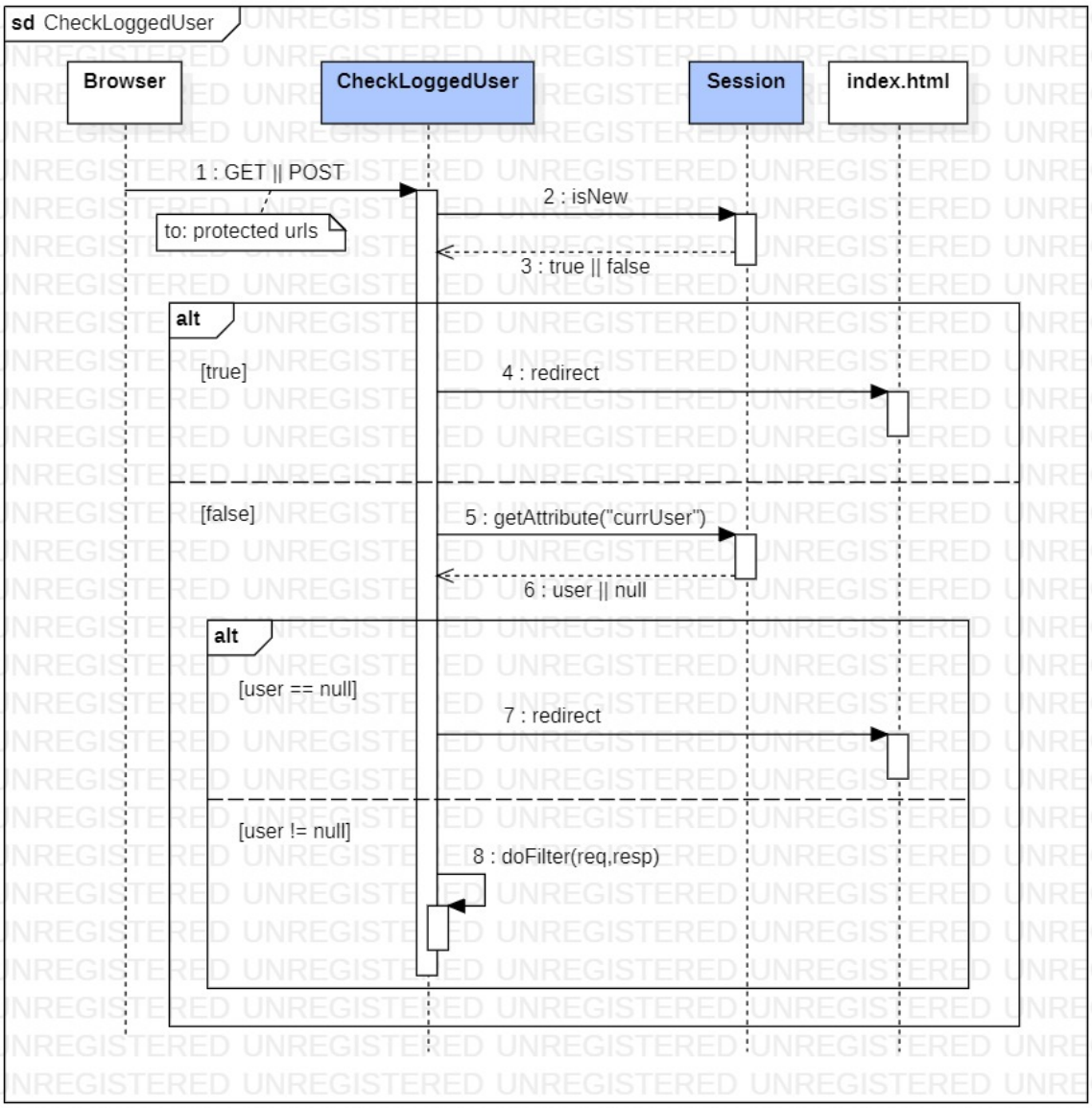

Figure 7.1: CheckLoggedUser

7.2. NoCacher

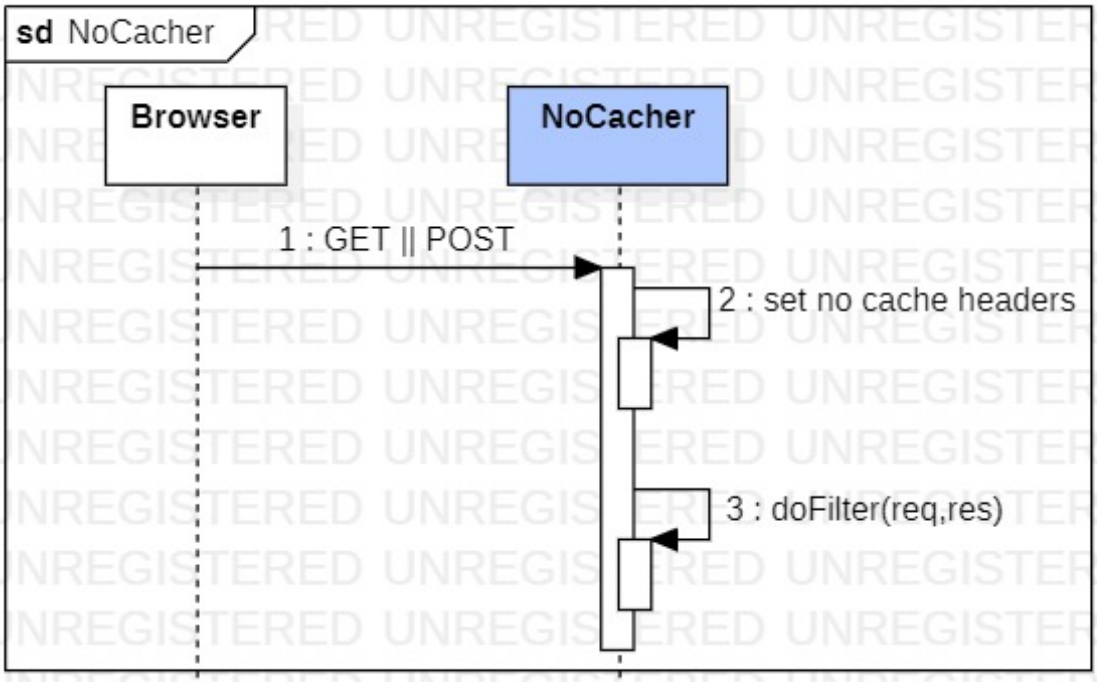

Figure 7.2: NoCacher

7.3. Login

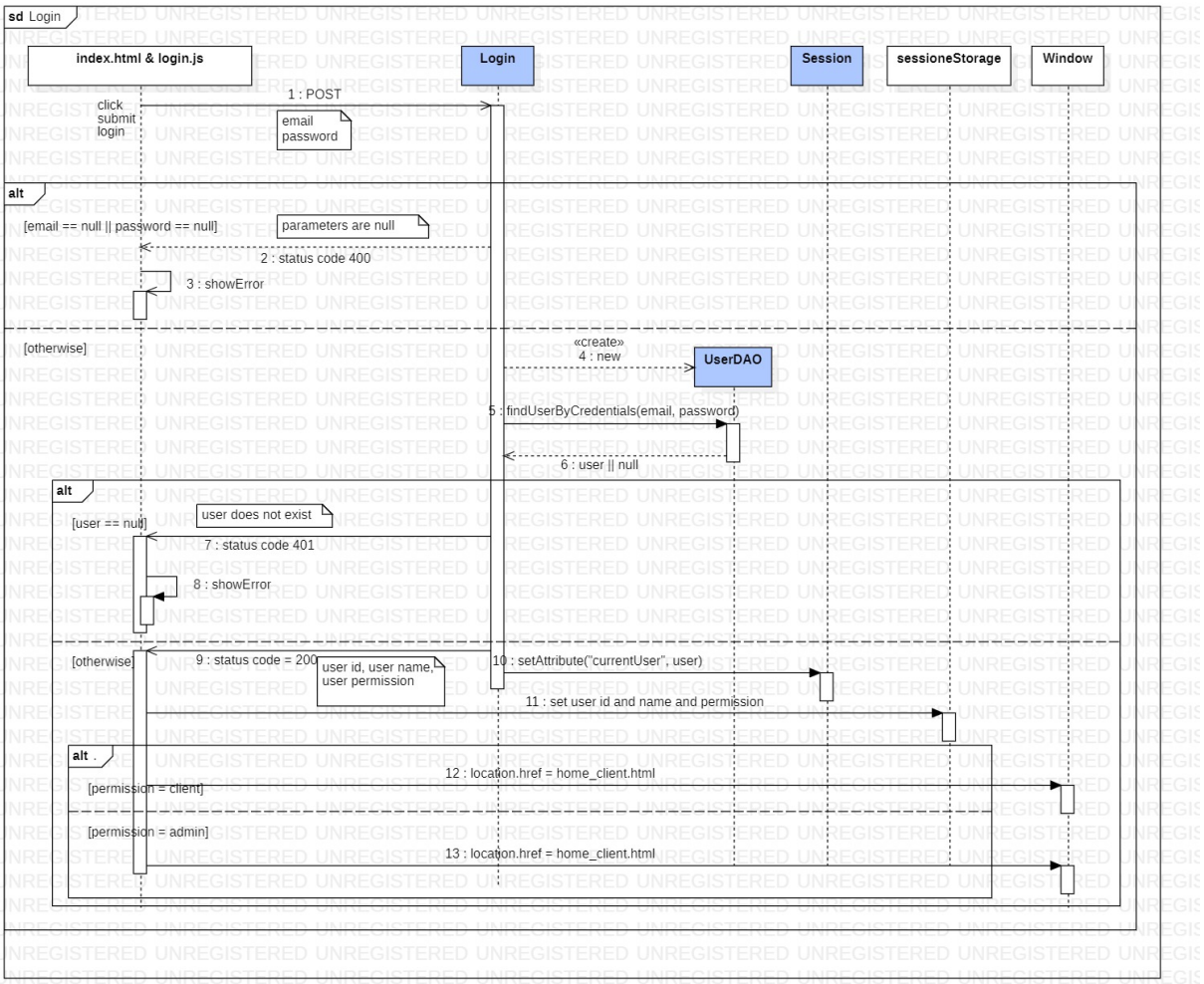

Figure 7.3: Login

7.4. Logout

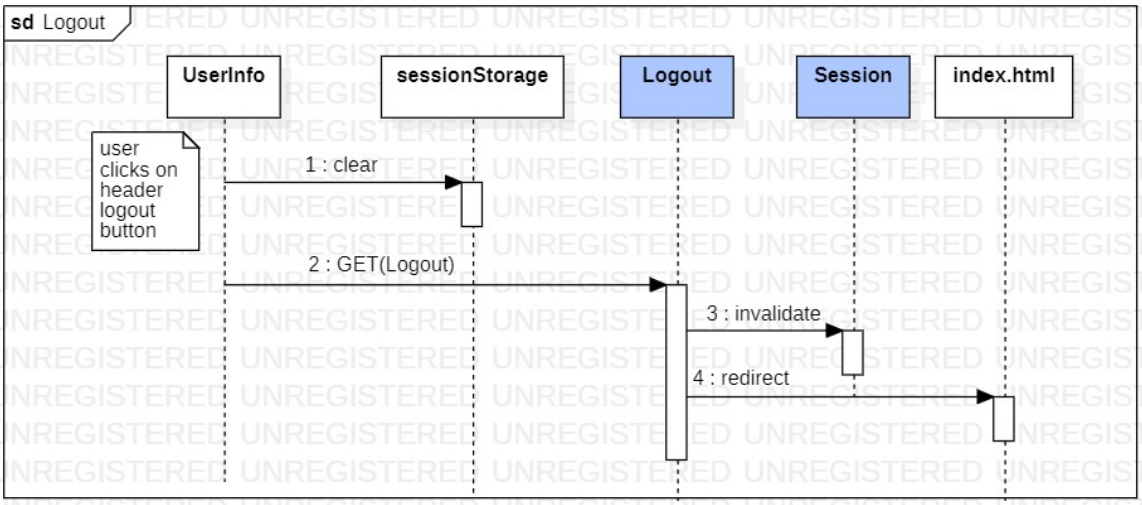

Figure 7.4: Logout

7.5. Signup

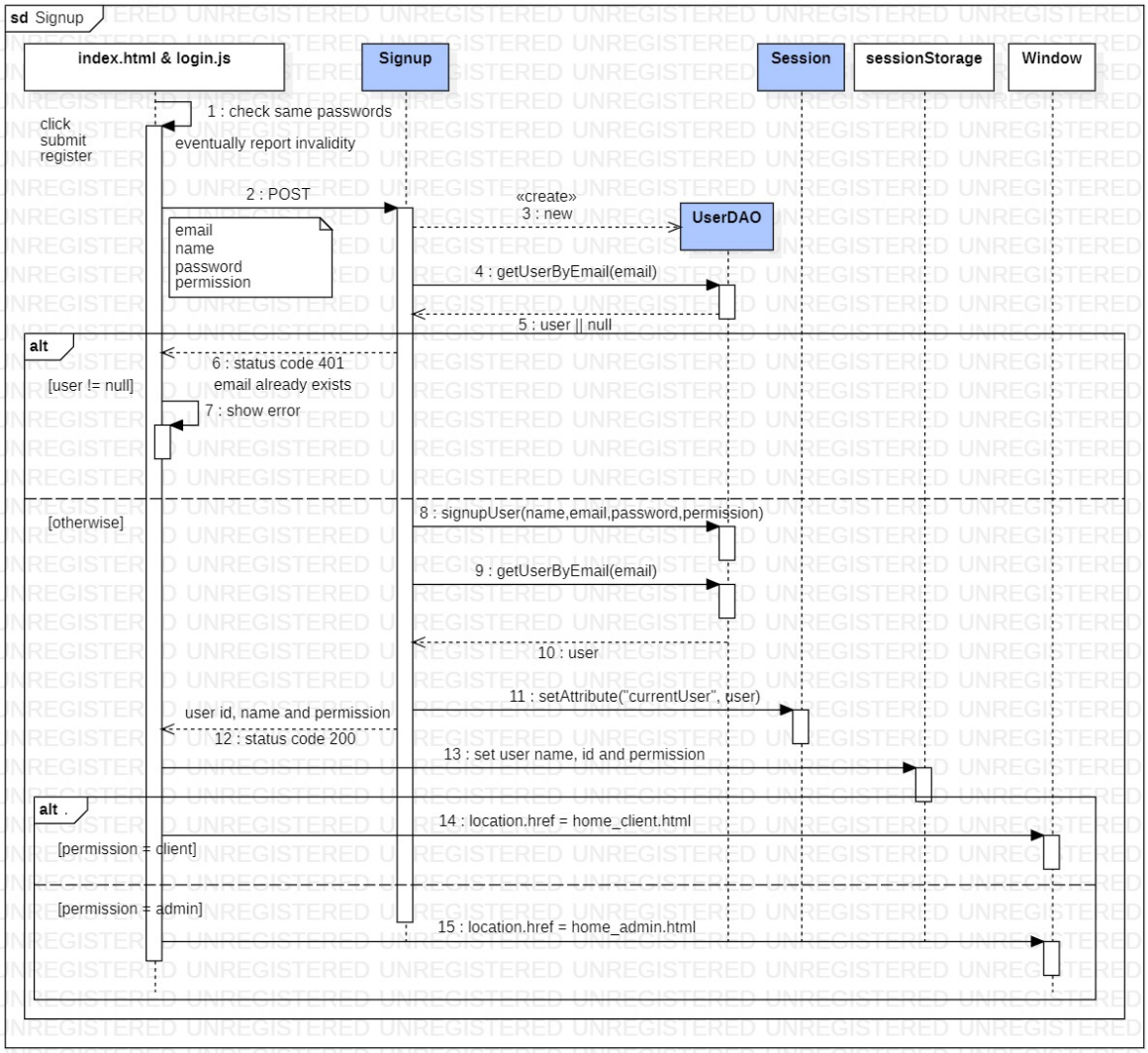

Figure 7.5: Signup

7.6. Home client page load

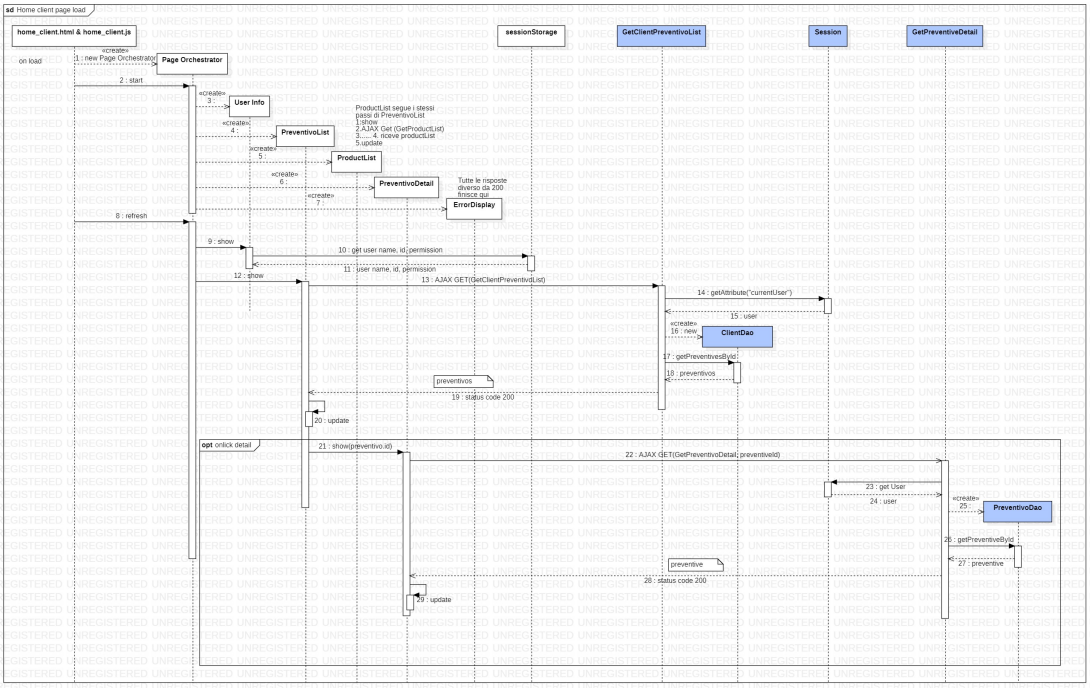

Figure 7.6: Home client page load