

---

# **Software Requirements Specification**

**for**

## **<PHP Travel Expenses' Management>**

**Version 1.0 approved**

**Prepared by <Raffaele Esposito>**

**<Università degli Studi della Campania Luigi Vanvitelli>**

**<inserire data>**

# Table of Contents

<b>Table of Contents .....</b>	<b>ii</b>
<b>1. Introduction.....</b>	<b>1</b>
1.1 Purpose .....	1
1.2 Document Conventions.....	1
1.3 Intended Audience and Reading Suggestions .....	1
<b>2. Overall Description.....</b>	<b>1</b>
2.2 Product Functions .....	1
2.3 User Classes and Characteristics .....	2
2.4 Operating Environment.....	3
2.5 Design and Implementation Constraints .....	3
<b>3. External Interface Requirements .....</b>	<b>4</b>
3.1 User Interfaces .....	4
3.3 Software Interfaces .....	8
3.4 Communications Interfaces.....	8
<b>4. System Features .....</b>	<b>9</b>
4.1 System Feature 1 .....	9
<b>5. Other Nonfunctional Requirements.....</b>	<b>10</b>
5.1 Performance Requirements .....	10
5.3 Security Requirements .....	10
5.4 Software Quality Attributes .....	10
<b>Appendix A: Glossary.....</b>	<b>11</b>
<b>Appendix B: Analysis Models .....</b>	<b>11</b>

# **1. Introduction**

## **1.1 Purpose**

*Questo SRS fornisce le specifiche per "PHP Travel Expenses' Management ", un'applicazione PHP che permette all'utente di aggiungere, modificare, eliminare e visualizzare viaggi comprendenti diverse informazioni andando a gestire quelle che sono le spese relative ad essi.. L'utente può infatti inserire la tipologia, una eventuale descrizione e l'importo di una spesa effettuata durante uno dei suoi viaggi . Inoltre, è possibile visualizzare il resoconto delle spese sia per il singolo viaggio e sia per la totalità dei viaggi associati ad un utente. Questa è la versione 1.0 del documento.*

## **1.2 Document Conventions**

*In questo documento sono stati utilizzati diagrammi UML per l'analisi e la progettazione in accordo con lo standard UML 2.0.*

## **1.3 Intended Audience and Reading Suggestions**

*Questo documento ha come destinatari sia il personale tecnico e sia chi si occuperà dello sviluppo.*

# **2. Overall Description**

## **2.1 Product Functions**

### **REQUISITI FUNZIONALI:**

*L'applicazione deve consentire all'utente sia tramite l'interfaccia web che tramite le quattro operazioni CRUD di:*

- Registrarsi ed effettuare il login;*
- Aggiungere e rimuovere viaggi dal proprio profilo specificando i parametri relative ad essi;*
- Modificare i dati relativi ad un viaggio ;*
- Visualizzare la lista di viaggi ad esso associati;*
- Aggiungere spese in un viaggio specificando la tipologia di spesa, la descrizione e l'importo;*
- Rimuovere spese in un viaggio;*
- Visualizzare report statistici di un singolo viaggio;*
- Visualizzare report statistici della totalità dei viaggi dell'utente;*

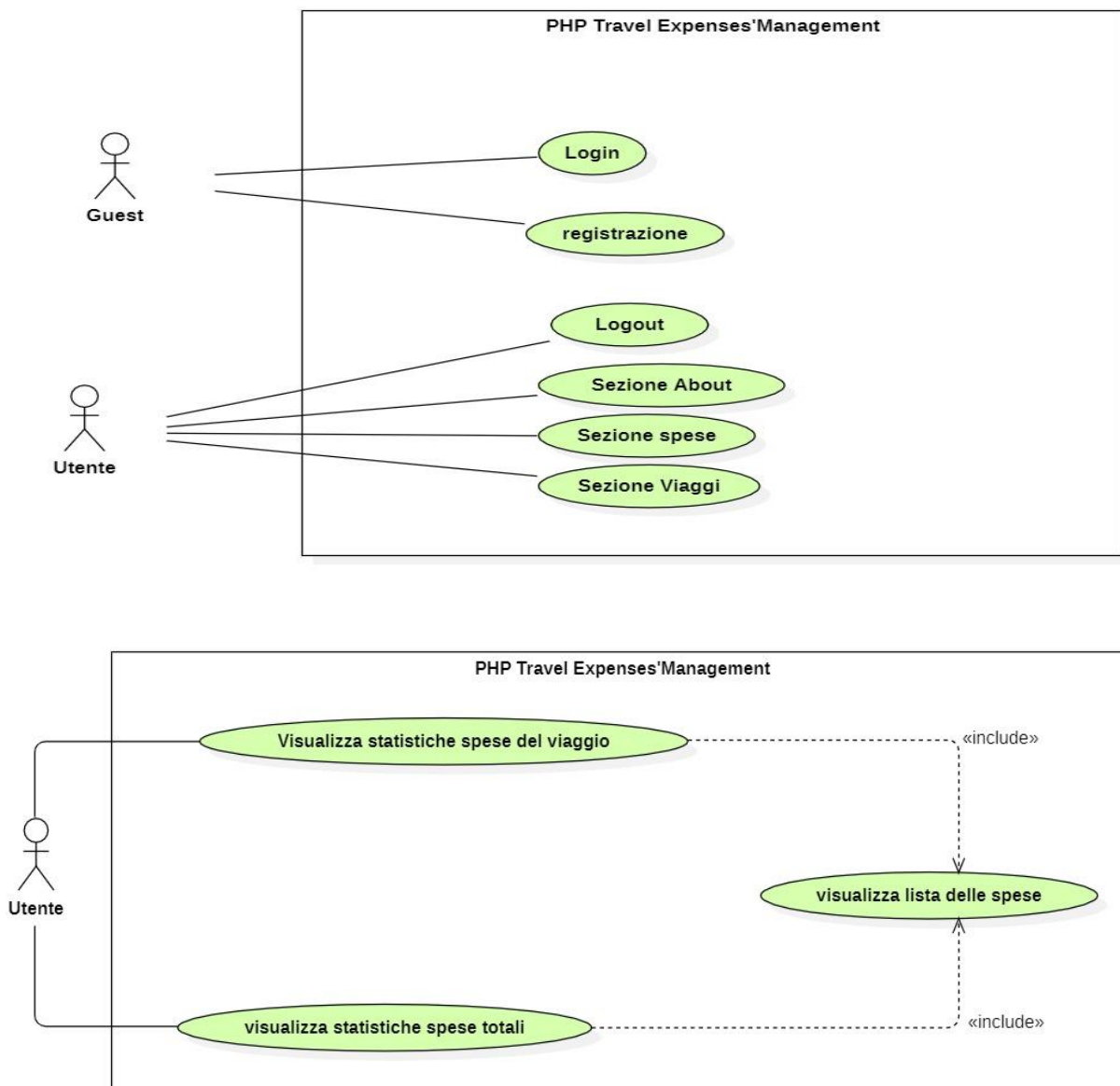
- Visualizzare proposte di viaggio;
- Effettuare il logout.

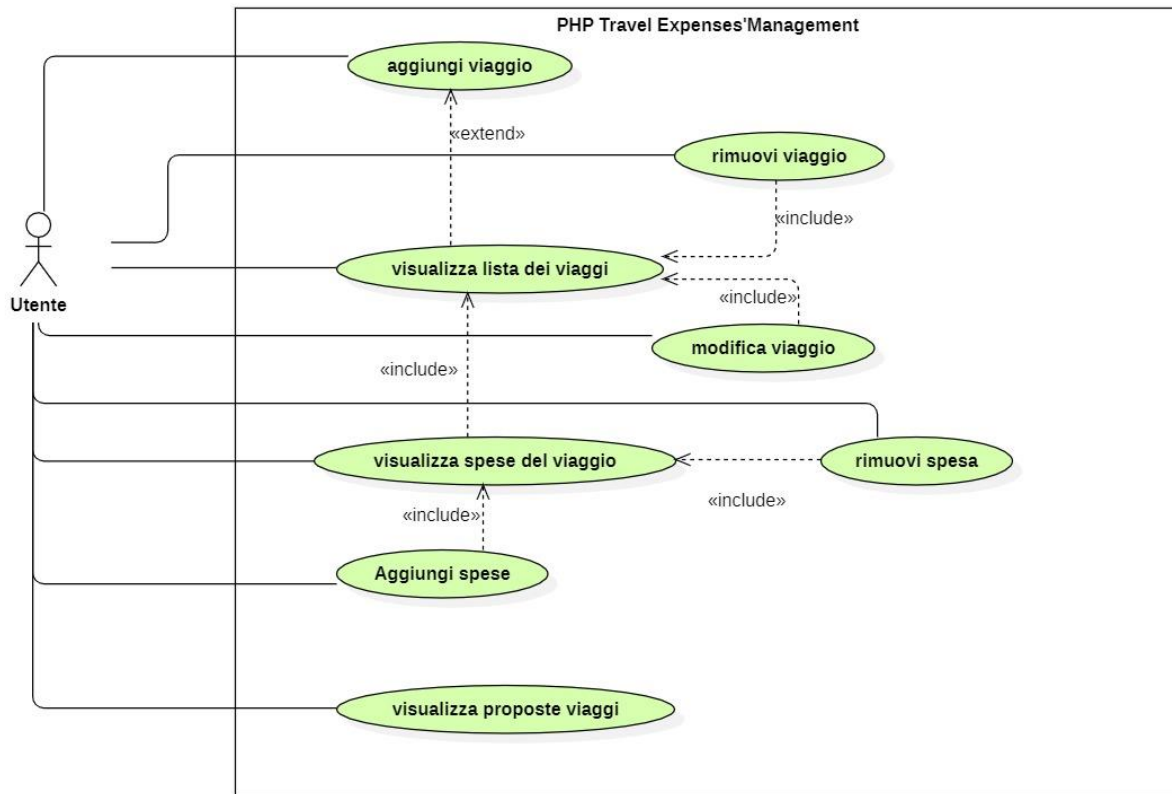
## REQUISITI NON FUNZIONALI

- Il sistema conserverà le password sotto forma di hash;
- Il sistema deve essere facile da utilizzare ed intuitivo per l'utente finale;
- L'interfaccia web sarà moderna e responsiva;
- L'utente non potrà accedere a determinare pagine se non autenticato in precedenza.

## 2.2 User Classes and Characteristics

L'applicazione ha una sola tipologia di utente, il quale, previa autenticazione, può accedere alle funzionalità precedentemente descritte dell'applicazione tramite l'interfaccia web .





## 2.3 Operating Environment

L'applicazione utilizza PHP come linguaggio server side, HTML, CSS e Javascript come linguaggi client side. Essendo un Progetto web locale, può essere eseguito su un qualsiasi OS che supporti XAMPP che include Apache e MySQL.

I back-end del sistema è stato sviluppato tramite PHP, e si occupa di gestire e di fornire le pagine web richieste dai client.

Il front-ente del sistema è stato sviluppato interamente in HTML e JavaScript e stilizzato in CSS .

Il codice HTML rappresenta la struttura della pagina web richiesta e viene fornito dal server tramite PHP, mentre in JavaScript è presente la logica per l'interazione con la pagina web, per la validazione dei dati e per inviare le eventuali richieste HTTP.

## 2.4 Design and Implementation Constraints

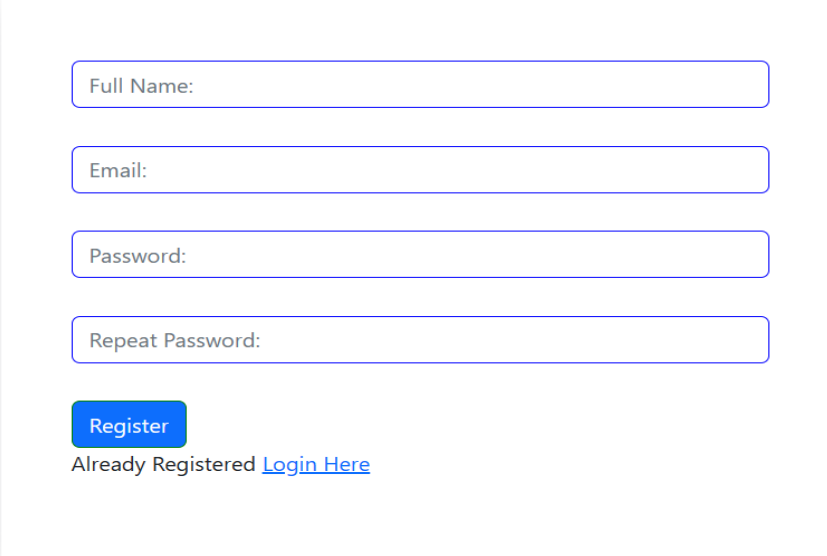
Il sistema opera tramite un Database SQL del tipo MariaDB, il server è hostato in locale tramite l'applicazione XAMPP e phpMyAdmin che si occupa della virtualizzazione del server e di gestire le richieste.

## 3. External Interface Requirements

### 3.1 User Interfaces

*L'applicazione espone un sito web in locale accessibile tramite un browser. L'interfaccia è moderna e responsive, utilizzabile sia da desktop che da mobile. Le varie sezioni del sito sono raggiungibili a partire dalla pagina del login del sito.*

- **INTERFACCIA LOGIN E REGISTRAZIONE per l'Utente o Ospite del servizio:**



A registration form with four text input fields and a submit button. The fields are labeled 'Full Name:', 'Email:', 'Password:', and 'Repeat Password:'. Below the fields is a blue 'Register' button. Under the button, the text 'Already Registered' is followed by a blue link 'Login Here'.

Full Name:

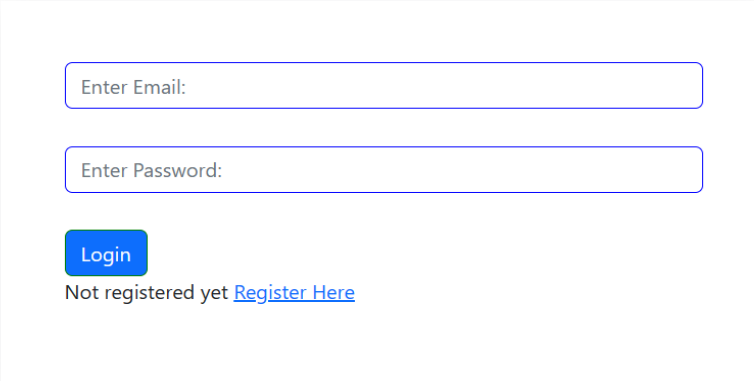
Email:

Password:

Repeat Password:

Register

Already Registered [Login Here](#)



A login form with two text input fields and a submit button. The fields are labeled 'Enter Email:' and 'Enter Password:'. Below the fields is a blue 'Login' button. Under the button, the text 'Not registered yet' is followed by a blue link 'Register Here'.

Enter Email:

Enter Password:

Login

Not registered yet [Register Here](#)

- **Home Page**

Per l'home page è stato scelto un design accattivante che mostri le funzionalità a cui l'utente può accedere mediante sia la barra di navigazione e sia a fondo pagina mediante i quick links.



- **Sezione viaggi: Visualizza, Aggiungi ed Elimina:**

La sezione viaggi mostra delle box dalle quali è possibile ,mediante click , raggiungere l'azione che si vuole effettuare.



- **Visualizzazione dei viaggi :**

Da questa schermata è possibile visualizzare i viaggi associate ad un utente con i relativi dati. E' possibile poi da tale schermata, sia aggiungere un nuovo viaggio, ritornare alla schermata home e, sulla destra, modificare i dati del form, eliminare il viaggio e visualizzarne le spese relative.

ID	Mezzo di Trasporto	Tempo di Viaggio	Destinazione	Durata	Alloggio	Budget	Bagaglio
52	aereo	2	roma	4	hotel	200.00	zaino
53	treno	2	catania	4	hotel	200.00	zaino

- **Visualizzazione delle spese relative ad un viaggio:**

Da questa schermata è possibile visualizzare le spese associate ad un utente di un viaggio con i relativi dati. E' possibile poi da tale schermata, sia aggiungere una nuova spesa scegliendo la tipologia di spesa che si vuole aggiungere fornendo un importo ed eventualmente una descrizione. Da qui si può poi ritornare alla schermata home ed anche eliminare delle spese. Infine si possono visualizzare i report delle spese.

ID	Tipo	Descrizione	Importo	ID Viaggio	ID Utente
22	Transport		21.00	52	2
23	Transport		8.00	52	2
25	Accommodation		7.00	52	2
26	Transport		4.00	52	2
27	Transport		21.50	52	2
28	Attractions		841.23	52	2
66	Transport	ciao	4.35	52	2
68	Shopping	Hard rock café	355.21	52	2
71	Food	eno	1.20	52	2

Tipo: Transport ▼ Descrizione:  Importo:  Aggiungi spesa

Statistiche spese del viaggio

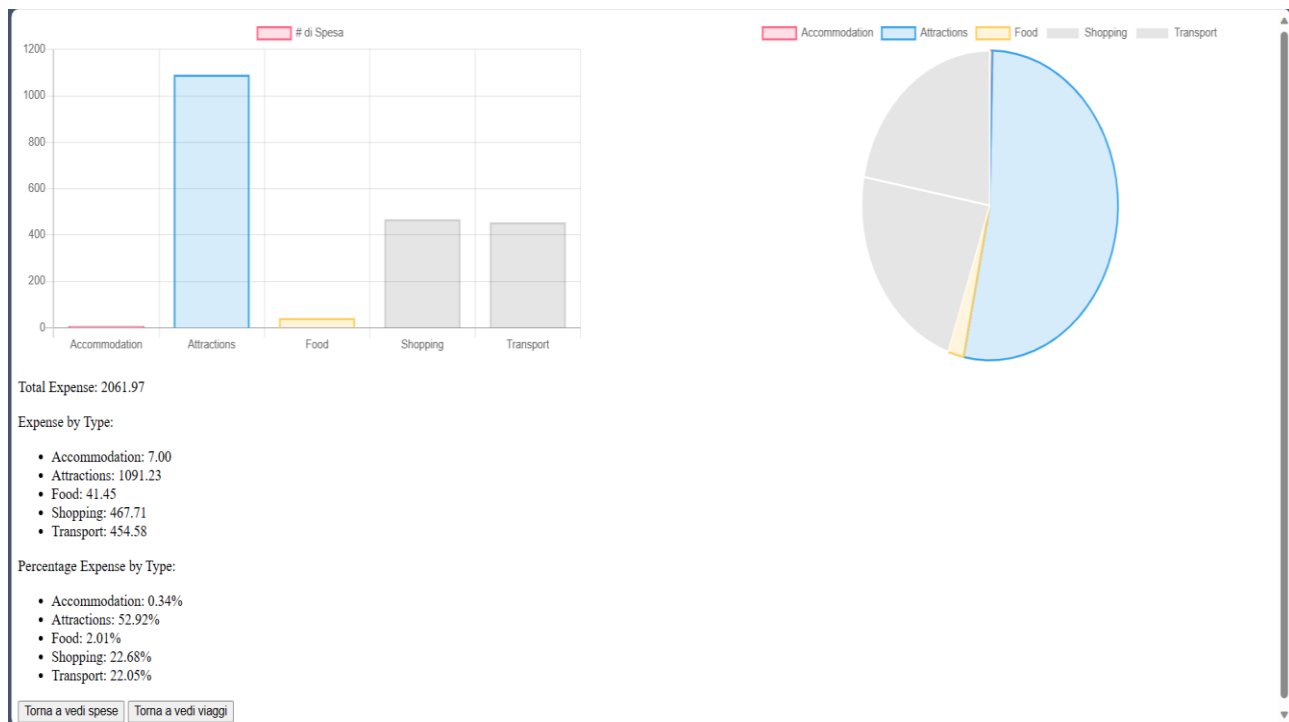
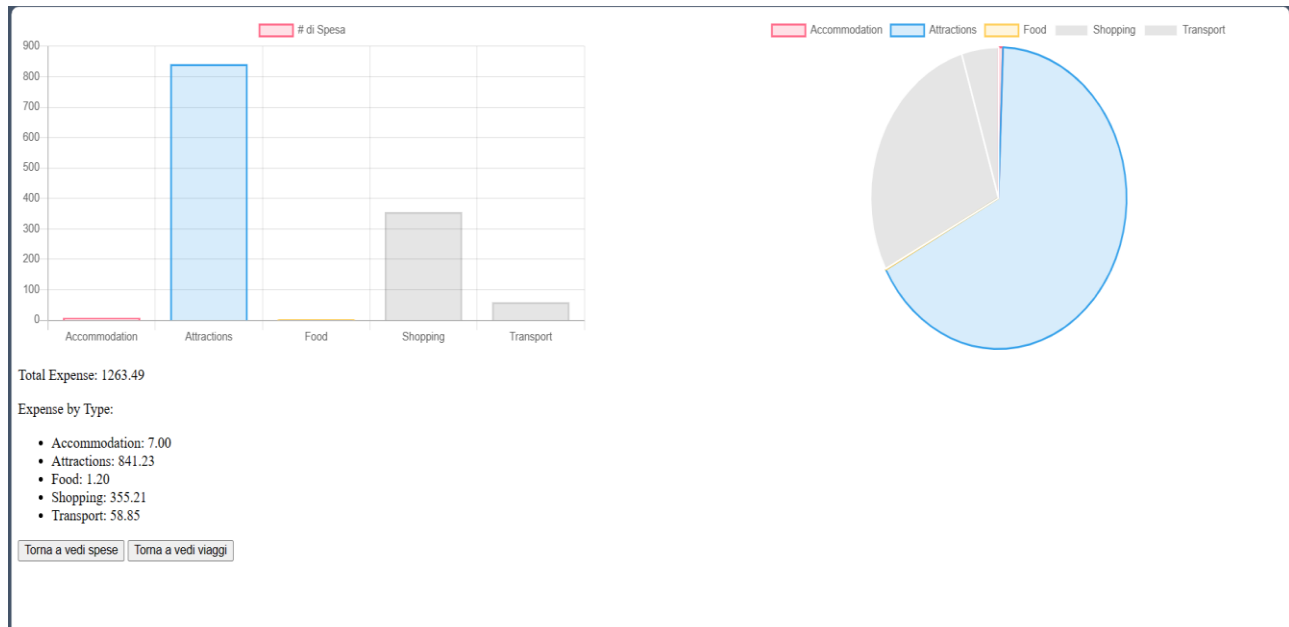
Statistiche spese totali

Torna a home Torna a vedi viaggi



- **Visualizzazione dei report delle spese**

Da tali schermate è possibile visualizzare le statistiche di spesa relative ad un viaggio o a tutti i viaggi associate all'utente. Sono presenti sia i dati in formato testuale che in formato grafico.



### 3.2 Software Interfaces

Il Sistema opera tramite:

- PHP 8.2.4 per le operazioni lato server;
- 10.4.28-MariaDB- mariadb.org binary distribution per la gestione del database ;
- phpMyAdmin per la gestione di richieste al web-server;
- HTML 5 per la struttura delle pagine web;
- JavaScript per l'interazione con l'utente lato client;
- HTTP per i protocolli di comunicazione;
- CSS per lo stile del sito web.

### 3.3 Communications Interfaces

Il Sistema elabora ogni richiesta tramite protocollo HTTP seguendo gli standard delle operazioni CRUD e risponde allo stesso modo formattando i date delle risposte in oggetti JSON.

Tutte le operazioni di lettura di informazioni del Sistema richiedono richieste HTTP GET mentre ogni azione di modifica o inserimenti di dati tramite richieste HTTP POST.

Esse sono state utilizzate per permettere all'utente di aggiungere (CREATE), visualizzare (READ), modificare (UPDATE) ed eliminare (DELETE) i suoi viaggi e le spese relative ad esse. Di seguito si riporta l'utilizzo di tali funzioni:

<b>Percorso</b>	<b>metodo</b>	<b>Descrizione</b>
project/registration.php	CREATE	Inserimento dei dati di un nuovo utente
project/login.php	READ	Verifica se l'indirizzo email dell'utente esiste nel database.
project/login.php	UPDATE	memorizza l'ID dell'utente nella sessione dopo che l'utente si è autenticato correttamente.
project/logout.php	DELETE	Distrukge la sessione dell'utente quando l'utente si disconnette.
../viaggi/add-new.php	CREATE	Creazione di un viaggio con inserimento dei relativi dati
../viaggi/edit.php	UPDATE	Modifica dati di un viaggio con modifica dei relative dati
../viaggi/index.php	READ	Visualizzazione dei viaggi associate all'utente
../viaggi/delete.php	DELETE	Eliminazione di un viaggio associato all'utente
../spese/aggiungi_spese.php	CREATE	Creazione di una spesa associata ad un viaggio dell'utente
../spese/vedi_spese.php	READ	Visualizzazione delle spese associate ad un viaggio dell'utente
../spese/del_spese.php	DELETE	Eliminazione di una spesa associata ad un viaggio

## 4. System Features

Di seguito è riportata una descrizione più dettagliata dei casi d'uso con l'utilizzo degli scenari.

### 4.1 Gestione Account

#### 4.1.1 Description and Priority

*L'utente può registrarsi, effettuare il login ed il logout.*

#### 4.1.2 Stimulus/Response Sequences

L'utente può eseguire queste operazioni accedendo a varie schermate del sito.

#### 4.1.3 Functional Requirements

- *Registrati: L'utente si registra al sito indicando username e password alfanumerici con password più lunga di 8 caratteri ;*
- *Login: L'utente accede al sito con le credenziali scelte in fase di registrazione ;*
- *Logout: L'utente effettua il logout dal sito.*

### 4.2 Gestione Viaggi

#### 4.2.1 Description and Priority

L'utente può aggiungere, rimuovere, visualizzare e gestire i propri viaggi.

#### 4.2.2 Stimulus/Response Sequences

L'utente può eseguire queste operazioni accedendo alla schermata home dei viaggi o alla schermata di dettaglio della visualizzazione dei viaggi .

#### 4.2.3 Functional Requirements

- *Aggiungi viaggio : L'utente inserisce un viaggio nel database specificando mezzo di trasporto, tempo di viaggio, destinazione, durata, alloggio, budget e bagaglio ;*
- *Visualizza viaggi: L'utente visualizza i viaggi associati ad un utente con i parametri prima citati . Da tale schermata può effettuare la rimozione di un viaggio, la modifica dei dati di un viaggio e la visualizzazione delle spese;*
- *Elimina viaggio: L'utente rimuove uno dei viaggi che sta visualizzando;*
- *Modifica viaggio: L'utente mediante click viene riportato al form dove può inserire I nuovi dati del viaggio.*

## 4.3 Gestione Spese

### 4.1.1 Description and Priority

L'utente può aggiungere, rimuovere, visualizzare le spese e mediante click anche le statistiche di spesa.

### 4.1.2 Stimulus/Response Sequences

L'utente può eseguire queste operazioni accedendo dalla schermata di dettaglio della visualizzazione dei viaggi e facendo click sul button delle spese.

### 4.1.3 Functional Requirements

- *Aggiungi spesa: L'utente inserisce una spesa per il viaggio nel database specificando tipologia di spesa, importo e descrizione.*
- *Rimuovi spesa: L'utente rimuove una delle spese che sta visualizzando.*
- *Visualizza statistiche di spese del viaggio: L'utente visualizza le statistiche delle spese relative ad un viaggio dell'utente.*
- *Visualizza statistiche: L'utente visualizza le statistiche delle spese relative a tutti i viaggi dell'utente.*

## 5. Other Nonfunctional Requirements

### 5.1 Performance Requirements

Il sistema deve essere reattivo e veloce, deve fare uso di query efficienti che non vadano a rallentare le elaborazioni delle richieste.

### 5.2 Security Requirements

*Il sistema non deve permettere di accedere a informazioni personali degli utenti, mantendone la privacy.*

### 5.3 Software Quality Attributes

*La struttura del software deve essere modulare e manutenibile in modo da poter effettuare facilmente modifiche e aggiornamenti.*

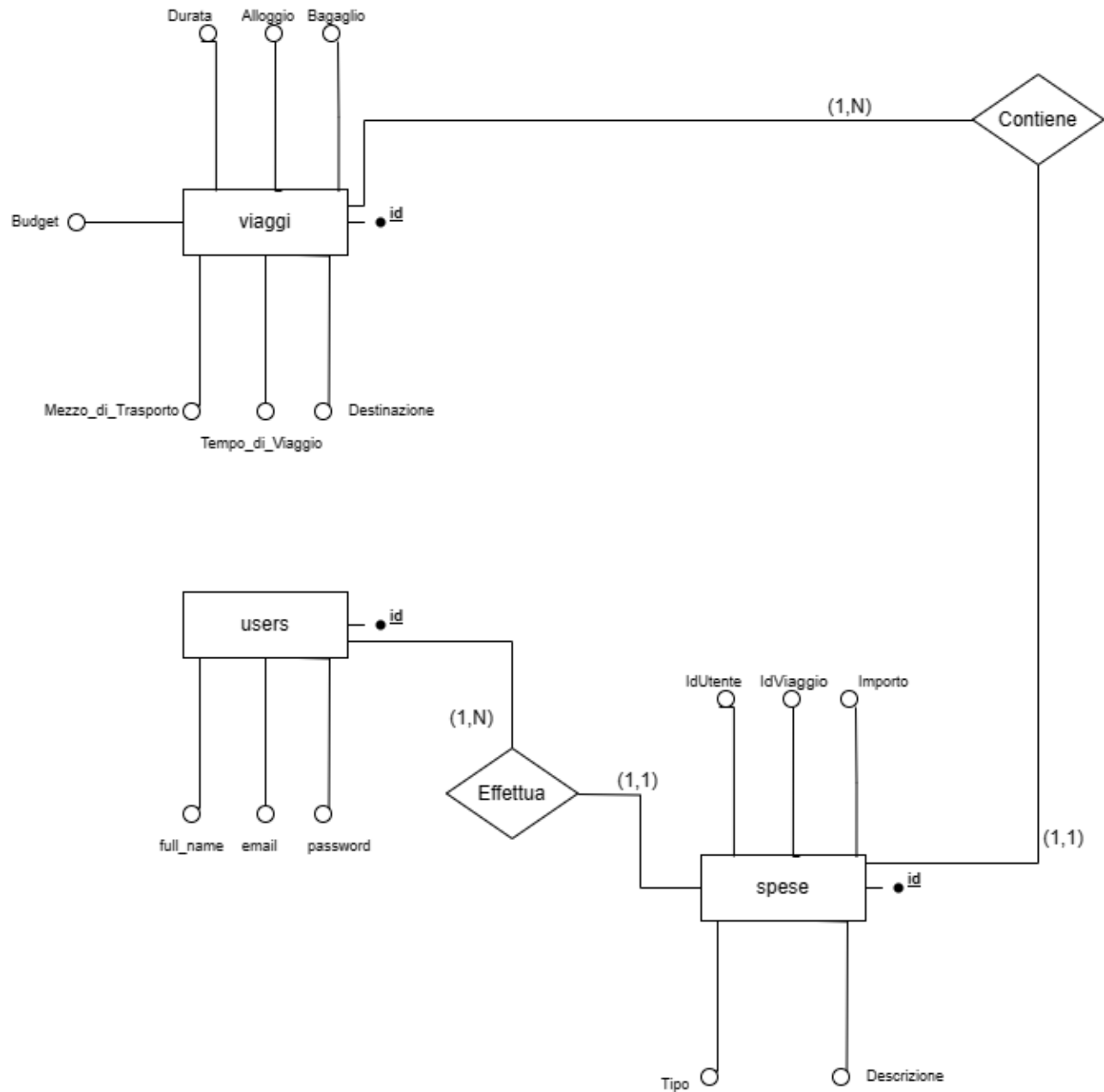
## Appendix A: Glossary

Nome	Descrizione
Utente	Persona che utilizza il sito. Può aggiungere, modificare, rimuovere e visualizzare viaggi e spese.
Viaggio	Entità associata ad un utente che può essere descritto da diversi campi. Si possono associare diverse spese ad esso.
Spese	Entità con una relazione stretta con utente e viaggio che può essere descritta andando a specificare tipo di spesa, descrizione e importo.
Statistiche	Entità che vengono rilevate in corrispondenza al carattere scelto di volta in volta per studiare il fenomeno in esame.

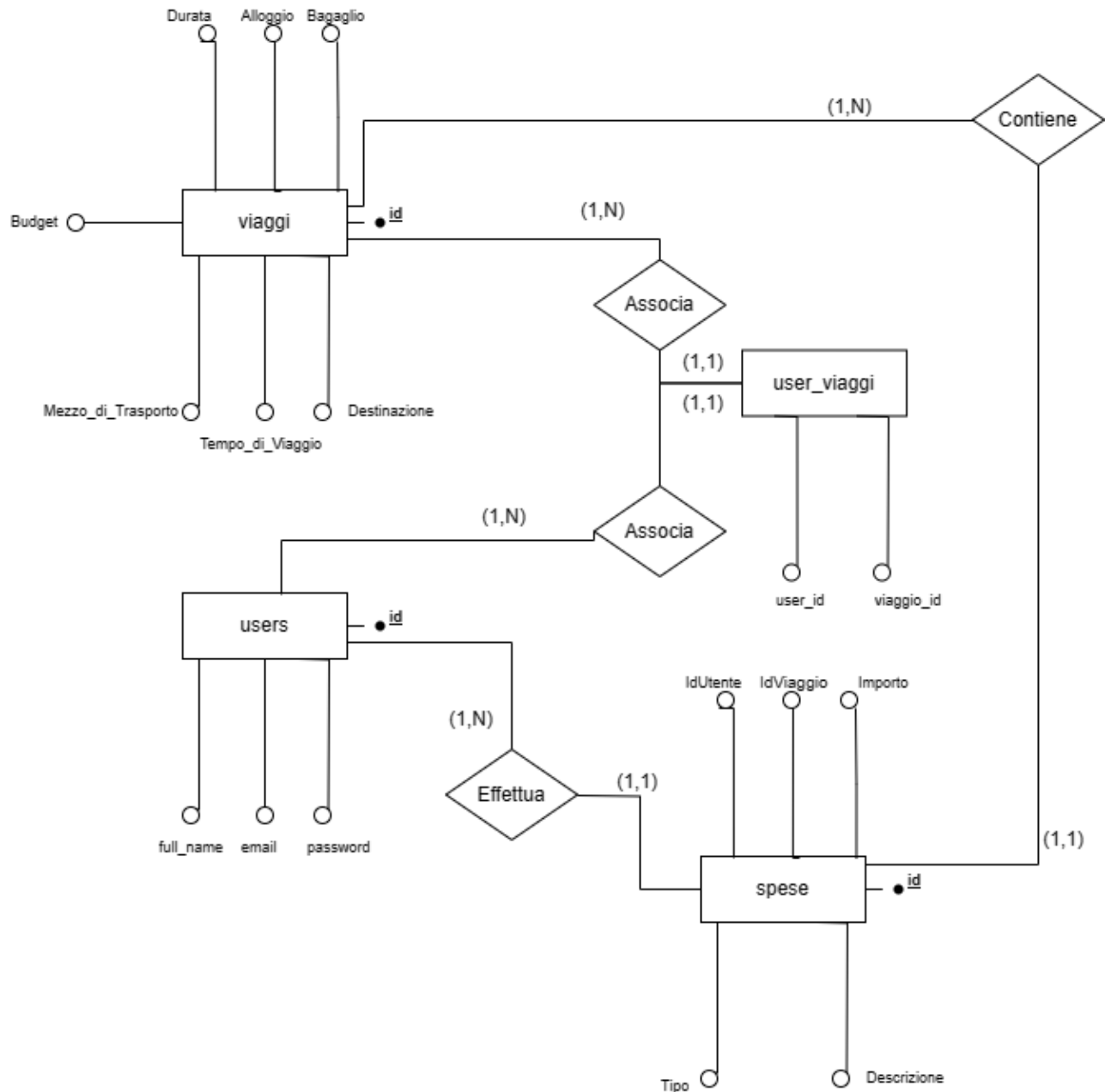
## Appendix B: Analysis Models

*L'analisi dei requisiti ha portato alle seguenti considerazioni. Ogni Utente dovrà essere identificato da un ID , da un nome complete ed una e-mail e potrà accedere al sistema con una password. Ogni viaggio dovrà essere identificato da un ID e sarà costituito da una serie di attributi opzionali (possono essere lasciati come null) quali il mezzo di trasporto , tempo di viaggio, la destinazione, il budget a disposizione, l'alloggio, il bagaglio e la durata. Ad ogni viaggio dell'utente può essere associate una o più spese effettuate dall'utente che dovrà essere identificata da un ID ed associata mediante id utente ed id viaggio . Gli attributi delle varie spese sono il tipo di spesa, la descrizione e l'importo della stessa.*

*Queste considerazioni sono state rappresentate nel seguente schema E-R.*



Lo schema E-R è stato ristrutturato aggiungendo un'entità separata che permette di associare e di tenere traccia delle associazioni tra gli id degli utenti e degli id dei viaggi. Tale entità prenderà il nome di *users\_viaggi* e sarà relazionata alle entità *viaggi* e *users*. Di seguito il diagramma E-R ristrutturato.



Di seguito il dizionario dei dati.:

<b>Entità</b>	<b>Descrizione</b>	<b>Attributi</b>	<b>Identificatore</b>
<i>Users</i>	Utilizzatore del sito	<i>Id,full_name,email,password</i>	<i>Id</i>
<i>Viaggi</i>	Entità che contiene spese ed info su un viaggio dell'utente	<i>Mezzo_di_Trasporto,Tempo_di_Viaggio, Destinazione,Durata, Alloggio,Bagaglio,Budget</i>	<i>Id</i>

<i>user_viaggi</i>	<i>Entità che associa id del viaggio con id dell'utente</i>	<i>user_id, viaggio_id</i>	
<i>Spese</i>	<i>Entità che può essere presente in un viaggio</i>	<i>Id, tipo, descrizione, importo, IdViaggio, IdUtente</i>	<i>id</i>

<b>Relazione</b>	<b>Descrizione</b>	<b>Componenti</b>
Associa	Informazione della connessione tra utente e viaggio	viaggi, user_viaggi
Associa	Informazione della connessione tra utente e viaggio	users, user_viaggi
Contiene	Informazione sulle spese presenti in un viaggio	Viaggi, spese
Effettua	Informazioni sulle spese dell'utente	users, spese

Dallo schema E-R si è passato allo schema logico. Le relazioni sono diventati vincoli di integrità referenziale.





Dallo schema logico si è poi passati alla scrittura di uno script SQL per la creazione della tabella.

```
CREATE TABLE `users` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `full_name` varchar(128) NOT NULL,  
  `email` varchar(255) NOT NULL,  
  `password` varchar(255) NOT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

```
CREATE TABLE `viaggi` (  
  `id` int(11) NOT NULL AUTO_INCREMENT,  
  `Mezzo_di Trasporto` varchar(255) DEFAULT NULL,  
  `Tempo_di Viaggio` int(11) DEFAULT NULL,  
  `Destinazione` varchar(255) DEFAULT NULL,  
  `Durata` int(11) DEFAULT NULL,  
  `Alloggio` varchar(255) DEFAULT NULL,  
  `Budget` decimal(10,2) DEFAULT NULL,  
  `Bagaglio` varchar(255) DEFAULT NULL,  
  PRIMARY KEY (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci ;
```

```
CREATE TABLE `user_viaggi` (  
  `user_id` int(11) DEFAULT NULL,  
  `viaggio_id` int(11) DEFAULT NULL,  
  KEY `user_id` (`user_id`),  
  KEY `viaggio_id` (`viaggio_id`),  
  CONSTRAINT `user_viaggi_ibfk_1` FOREIGN KEY (`user_id`) REFERENCES `users` (`id`),  
  CONSTRAINT `user_viaggi_ibfk_2` FOREIGN KEY (`viaggio_id`) REFERENCES `viaggi`  
  (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```

```
CREATE TABLE `spese` (  
  `Id` int(11) NOT NULL AUTO_INCREMENT,  
  `Tipo` varchar(50) NOT NULL,  
  `Descrizione` text DEFAULT NULL,  
  `Importo` decimal(10,2) NOT NULL,  
  `IdViaggio` int(11) DEFAULT NULL,  
  `IdUtente` int(11) DEFAULT NULL,  
  PRIMARY KEY (`Id`),  
  KEY `IdViaggio` (`IdViaggio`),  
  KEY `IdUtente` (`IdUtente`),  
  CONSTRAINT `spese_ibfk_1` FOREIGN KEY (`IdViaggio`) REFERENCES `viaggi` (`id`)  
ON DELETE CASCADE,  
  CONSTRAINT `spese_ibfk_2` FOREIGN KEY (`IdUtente`) REFERENCES `users` (`id`)  
) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_general_ci;
```