Indice

Modello Concettuale 2

Modello Relazionale 3

DDL
- Creazione Tabelle 4
- Chiavi Esterne 5
- Query
  - Richieste 6
  - Altre Query
    - *Autenticazione* 7
    - *Prenotazioni* 8
    - *Alberghi* 9
    - *Stanze* 10-11

API
- Autenticazione 12
- Prenotazioni 13
- Alberghi 14
- Stanze 15

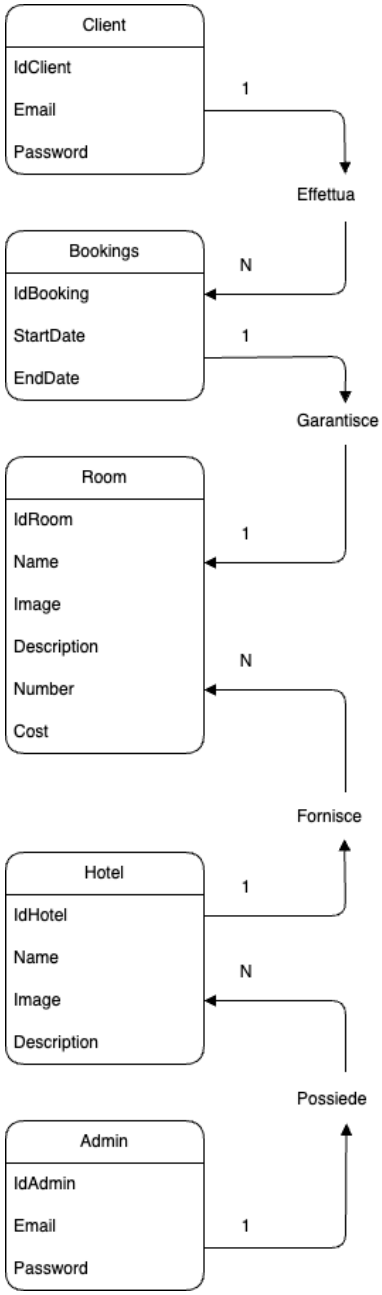
Links
- Sito Web Server 16
- Sito Web Client 16
- Mockup APP Mobile 16

Considerazioni 17

Casi d'Uso 18

Credenziali 19

Modello Concettuale



Modello Relazionale

Clients (
    IdClient(PK),
    Email (NOT NULL),
    Password (NOT NULL)
)

Admins (
    IdAdmin(PK),
    Email (NOT NULL),
    Password (NOT NULL)
)

Bookings (
    IdBooking(PK),
    IdClient (FK),
    IdRoom (FK),
    StartDate (NOT NULL),
    EndDate (NOT NULL)
)

Hotels (
    IdHotel(PK),
    IdAdmin (FK),
    IdRoom (FK),
    Name (NOT NULL),
    Image (NOT NULL),
    Description (NOT NULL)
)

Rooms (
    IdRoom(PK),
    IdHotel (FK),
    Name (NOT NULL),
    Image (NOT NULL),
    Description (NOT NULL),
    Number (NOT NULL)
    Cost (NOT NULL)
)

DDL Creazione Tabelle

```
#Creazione Tabella Clienti
CREATE TABLE Clients(
    Id INTEGER PRIMARY KEY AUTO_INCREMENT,
    Email VARCHAR(255) NOT NULL,
    Password VARCHAR(255) NOT NULL
);

#Creazione Tabella Admins
CREATE TABLE Admins(
    Id INTEGER PRIMARY KEY AUTO_INCREMENT,
    Email VARCHAR(255) NOT NULL,
    Password VARCHAR(255) NOT NULL
);

#Creazione Tabella Alberghi
CREATE TABLE Hotels(
    Id INTEGER PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(24) NOT NULL,
    Image VARCHAR(255) NOT NULL,
    Description VARCHAR(255) NOT NULL
);

#Creazione Tabella Stanze
CREATE TABLE Rooms(
    Id INTEGER PRIMARY KEY AUTO_INCREMENT,
    Name VARCHAR(24) NOT NULL,
    Image VARCHAR(255) NOT NULL,
    Description VARCHAR(255) NOT NULL,
    Number INTEGER NOT NULL,
    Cost INTEGER NOT NULL
);

 #Creazione Tabella Prenotazioni
CREATE TABLE Bookings(
    Id INTEGER PRIMARY KEY AUTO_INCREMENT,
    StartDate DATE NOT NULL,
    EndDate DATE NOT NULL
);
```

DDL Chiavi Esterne

```
#Chiave Esterna Alberghi per Relazione 1-N Alberghi-Admins con clausola ON DELETE CASCADE
ALTER TABLE Hotels
ADD COLUMN IdAdmin INTEGER,
ADD CONSTRAINT FK_IdAdmin
FOREIGN KEY (IdAdmin)
REFERENCES Admins(Id)
ON DELETE CASCADE;

#Chiave Esterna Stanze per Relazione 1-N Alberghi-Stanze — con clausola ON DELETE CASCADE
ALTER TABLE Rooms
ADD COLUMN IdHotel INTEGER,
ADD CONSTRAINT FK_IdHotel
FOREIGN KEY (IdHotel)
REFERENCES Hotels(Id)
ON DELETE CASCADE;

#Chiave Prenotazioni per Relazione 1-N Clienti-Prenotazioni con clausola ON DELETE CASCADE
ALTER TABLE Bookings
ADD COLUMN IdClient INTEGER,
ADD CONSTRAINT FK_IdClient
FOREIGN KEY (IdClient)
REFERENCES Clients(Id)
ON DELETE CASCADE;

#Chiave Esterna Prenotazioni per Relazione 1-1 Prenotazioni-Stanze con clausola ON DELETE CASCADE
ALTER TABLE Bookings
ADD COLUMN IdRoom INTEGER,
ADD CONSTRAINT FK_IdRoom
FOREIGN KEY (IdRoom)
REFERENCES Rooms(Id)
ON DELETE CASCADE;
```

DDL Query ~ Query Richieste

Query Alberghi

```
#Query che permette la visualizzazione delle strutture alberghiere con il numero di camere
SELECT Hotels.Id as IdHotels, COUNT(Rooms.Id) as NumRooms
FROM Hotels
JOIN Rooms
ON Hotels.Id = Rooms.IdHotel
GROUP BY Hotels.Id;
```

Query Stanze

```
#Query che permette la visualizzazione delle camere disponibili
SELECT (TotalRooms.NumRooms - BookedRooms.NumRooms) AS Availability
FROM
   (SELECT COUNT(Rooms.Id) AS NumRooms, Hotels.Id AS HotelId
    FROM Rooms
    JOIN Hotels ON Rooms.IdHotel = Hotels.Id
    GROUP BY Hotels.Id) AS TotalRooms
JOIN
   (SELECT COUNT(CASE WHEN Bookings.StartDate >= :startDate AND
Bookings.EndDate <= :endDate THEN Rooms.Id END) AS NumRooms, Hotels.Id AS
HotelId
    FROM Rooms
    JOIN Hotels ON Rooms.IdHotel = Hotels.Id
    JOIN Bookings ON Rooms.Id = Bookings.IdRoom
    GROUP BY Hotels.Id) AS BookedRooms
ON TotalRooms.HotelId = BookedRooms.HotelId;
```

DDL Query ~ Altre Query

Query Admins

```
-- query per il selezionamento di un Admin —-
SELECT * FROM Admins WHERE Email = :email AND Password = :password
-- query per inserire un Admin —-
INSERT INTO Admins (Email, Password) VALUES (:email, :password)
-- query per eliminare un Admin —-
DELETE FROM Admins WHERE Id = :userId
-- query per il verifica esista Email di un Admin —-
SELECT Email FROM Admins WHERE Email = :email
```

Query Clienti

```
-- query per il selezionamento di un Cliente —-
SELECT * FROM Clients WHERE Email = :email AND Password = :password
-- query per inserire un Cliente —-
INSERT INTO Clients (Email, Password) VALUES (:email, :password)
-- query per eliminare un Cliente —-
DELETE FROM Clients WHERE Id = :userId
-- query per il verifica esista Email di un Cliente —-
SELECT Email FROM Clients WHERE Email = :e
```

DDL Query ~ Altre Query

Query Prenotazioni

```
-- query per aggiunta di una Prenotazione —-
INSERT INTO Bookings (IdClient, IdRoom, StartDate, EndDate)
VALUES (:idClient, :idRoom, :startDate, :endDate)
-- query per selezionare Prenotazioni di Cliente —-
SELECT
b.Id AS BookingId,
b.StartDate AS BookingStartDate,
b.EndDate AS BookingEndDate,
r.Number AS BookingRoomNumber,
r.Name AS RoomName,
r.Cost * DATEDIFF(b.EndDate, b.StartDate) AS TotalCost,
r.IdHotel AS RoomHotelId,
h.Name AS HotelName
FROM Bookings b
JOIN Rooms r ON b.IdRoom = r.Id
JOIN Hotels h ON r.IdHotel = h.Id
WHERE b.IdClient = :idClient
-- query per selezionare Prenotazioni di Admin —-
SELECT
b.Id AS BookingId,
b.StartDate AS BookingStartDate,
b.EndDate AS BookingEndDate,
r.Number AS BookingRoomNumber,
r.Name AS RoomName,
r.Cost * DATEDIFF(b.EndDate, b.StartDate) AS TotalCost,
r.IdHotel AS RoomHotelId,
h.Name AS HotelName,
c.Email AS ClientEmail
FROM Bookings b
JOIN Rooms r ON b.IdRoom = r.Id
JOIN Hotels h ON r.IdHotel = h.Id
JOIN Clients c ON b.IdClient = c.Id
JOIN Admins a ON a.Id = :adminId
```

DDL Query ~ Altre Query

Query Alberghi

```
-- query per eliminare un Albergo —-
DELETE FROM Hotels WHERE Id = :idHotel;
-- query per selezionare un Albergo —-
SELECT b.name FROM Hotels b WHERE b.Id = :id;
-- query per aggiornare un Albergo —-
UPDATE Hotels
SET Name = :name, Image = :image, Description = :description
WHERE Id = :idHotel
-- query per aggiungere un Albergo —-
INSERT INTO Hotels (IdAdmin, Name, Image, Description)
VALUES (:idAdmin, :name, :image, :description)
-- query per verificare esistenza di un Albergo —-
SELECT COUNT(*) AS numHotels FROM Hotels WHERE Id = :idHotel
-- query per selezionare gli Alberghi di un Admin —-
SELECT * FROM Hotels WHERE IdAdmin = :idAdmin
-- query per selezionare tutti gli Alberghi —-
SELECT h.*,
(SELECT COUNT(*) FROM Rooms r WHERE r.IdHotel = h.Id) AS TotalRooms,
(SELECT COUNT(*)
FROM Rooms r
LEFT JOIN Bookings b
ON r.Id = b.IdRoom
WHERE h.Id = r.IdHotel
AND (b.Id IS NULL OR :startDate > b.EndDate OR :endDate < b.StartDate)
) AS Availability
FROM Hotels h;
-- query per selezionare un Albergo di Id —-
SELECT h.*,
COUNT(r.Id) AS TotalRooms,
SUM(CASE
WHEN b.IdRoom IS NULL THEN 1
WHEN :startDate > b.EndDate OR :endDate < b.StartDate THEN 1
ELSE 0
END) AS Availability
FROM Hotels h
LEFT JOIN Rooms r ON h.Id = r.IdHotel
LEFT JOIN Bookings b ON r.Id = b.IdRoom
WHERE h.Id = :id
GROUP BY h.Id
```

DDL Query ~ Altre Query

Query Stanze

```
-- query per aggiungere una Stanza —-
INSERT INTO Rooms (IdHotel, Name, Image, Description, Cost, Number)
SELECT :idHotel, :name, :image, :description, :cost, IFNULL(MAX(Number) + 1, 1) AS
NewNumber
FROM Rooms
WHERE IdHotel = :idHotel AND Name = :name
-- query per aggiornare una Stanza —-
UPDATE Rooms
SET Name = :name, Image = :image, Description = :description, Cost = :cost
WHERE IdHotel = :idHotel
AND Name = :name;
-- query per selezionare le Stanze Disponibili per Prenotazione —-
SELECT
DATEDIFF(:endDate, :startDate) * MAX(r.Cost) AS Cost,
SUM(
CASE
WHEN b.IdRoom IS NULL THEN 1
WHEN :startDate > b.EndDate OR :endDate < b.StartDate THEN 1
ELSE 0
END
) AS Availability
FROM Rooms AS r
LEFT JOIN Bookings AS b ON r.Id = b.IdRoom
WHERE r.IdHotel = :hotelId AND r.Name LIKE CONCAT('%', :roomName, '%')
GROUP BY r.Name
-- query per selezionare la prima Stanza Disponibili per Nome Stanza —-
SELECT r.Id AS RoomId
FROM Rooms AS r
LEFT JOIN Bookings AS b ON r.Id = b.IdRoom
WHERE
r.IdHotel = :hotelId
AND r.Name LIKE CONCAT('%', :roomName, '%')
AND (
b.IdRoom IS NULL
OR :startDate > b.EndDate
OR :endDate < b.StartDate
)
GROUP BY r.Id, r.
```

DDL Query ~ Altre Query

Query Stanze

```
-- query per selezionare le Stanze di un Albergo —-
SELECT r.Name, MAX(r.Image) AS Image, MAX(r.Description) AS Description, MAX(r.Cost)
AS Cost, COUNT(r.Id) AS TotalRooms, SUM(CASE
WHEN b.IdRoom IS NULL THEN 1
WHEN :startDate > b.EndDate OR :endDate < b.StartDate THEN 1
ELSE 0
END) AS Availability
FROM Rooms AS r
LEFT JOIN Bookings AS b ON r.Id = b.IdRoom
WHERE r.IdHotel = :hotelId
GROUP BY r.Name
ORDER BY Cost ASC;
-- query per selezionare le Stanze di un Albergo raggruppate per Nome Stanza —-
SELECT r.Name, MAX(r.Image) AS Image, MAX(r.Description) AS Description,
DATEDIFF(:endDate, :startDate) * MAX(r.Cost) AS Cost, COUNT(r.Id) AS TotalRooms,
SUM(
CASE
WHEN b.IdRoom IS NULL THEN 1
WHEN :startDate > b.EndDate OR :endDate < b.StartDate THEN 1
ELSE 0
END
) AS Availability
FROM Rooms AS r
LEFT JOIN Bookings AS b ON r.Id = b.IdRoom
WHERE r.IdHotel = :hotelId AND r.Name LIKE CONCAT('%', :roomName, '%')
GROUP BY r.Name
-- query per contare le Stanze di un Albergo per Nome Stanza —-
SELECT COUNT(*) AS totalRooms
FROM Rooms
WHERE IdHotel = :idHotel
AND Name = :name;
-- query per eliminare una Stanza di un Albergo da Nome Stanza e Id Albergo —-
DELETE FROM Rooms
WHERE IdHotel = :idHotel
AND Name = :name
LIMIT 1;
```

API Autenticazione

Descrizione: Autenticazione Utente per effettuare Accesso Admin o Cliente
Endpoint: type=auth&method=login&authState={authState}
Metodo HTTP: POST
Parametri:

- **AuthState** (obbligatorio): Stato di login dell'utente: 'Admin' o 'Cliente'
- **Email** (obbligatorio): Email dell'utente
- **Password** (obbligatorio): Password dell'utente

Descrizione: Autenticazione Utente per effettuare Registrazione Admin o Cliente
Endpoint: type=auth&method=register&authState={authState}
Metodo HTTP: POST
Parametri:

- **AuthState** (obbligatorio): Stato di login dell'utente: 'Admin' o 'Cliente'
- **Email** (obbligatorio): Email dell'utente
- **Password** (obbligatorio): Password dell'utente

Descrizione: Eliminazione Admin o Cliente
Endpoint: type=auth&method=delete&authState={authState}
Metodo HTTP: POST
Parametri:

- **AuthState** (obbligatorio): Stato di login dell'utente: 'Admin' o 'Cliente'
- **IdUtente** (obbligatorio): Id Utente da Eliminare

API Prenotazioni

Descrizione: Recupera le Prenotazioni effettuate da un Cliente
Endpoint: type=book&method=getClientBookings&clientId={clientId}
Metodo HTTP: GET
Parametri:
- **ClientId** (obbligatorio): L'ID del Cliente da cui recuperare le Prenotazioni

Esempio: Link API getClientBookings

Descrizione: Recupera Prenotazioni gestite da un Admin
Endpoint: type=book&method=getClientBookings&adminId={adminId}
Metodo HTTP: GET
Parametri:
- **AdminId** (obbligatorio): L'ID dell'Admin da cui recuperare le Prenotazioni

Esempio: Link API getAdminBookings

Descrizione: Aggiunta di una nuova Prenotazione
Endpoint: type=book&method=newBooking
Metodo HTTP: POST
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo delle stanze da recuperare
- **StartDate** (obbligatorio): La data di inizio nel formato 'YYYY-MM-DD'
- **EndDate** (obbligatorio): La data di fine nel formato 'YYYY-MM-DD'
- **RoomName** (obbligatorio): Il Nome della Stanza da recuperare

API Alberghi

Descrizione: Recupero Alberghi di Admin
Endpoint: type=hotel&method=getHotelsByIdAdmin&idAdmin={idAdmin}
Metodo HTTP: GET
Parametri:
- **IdAdmin** (obbligatorio): L'ID dell'Admin da cui recuperare gli Alberghi

Esempio: Link API getHotelsByIdAdmin

Descrizione: Recupero di tutti gli Alberghi
Endpoint: type=hotel&method=getAllHotels&today={today}&tomorrow={tomorrow}
Metodo HTTP: GET
Parametri:
- **Today** (opzionale): La data di inizio nel formato 'YYYY-MM-DD'
- **Tomorrow** (opzionale): La data di fine nel formato 'YYYY-MM-DD'

Esempio: Link API getAllHotels

Descrizione: Recupero di un solo Albergo
Endpoint: type=hotel&method=getHotelById&idHotel={idHotel}&today={today}
&tomorrow={tomorrow}
Metodo HTTP: GET
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo da recuperare
- **Today** (opzionale): La data di inizio nel formato 'YYYY-MM-DD'
- **Tomorrow** (opzionale): La data di fine nel formato 'YYYY-MM-DD'

Esempio: Link API getHotelById

Descrizione: Eliminazione di un Albergo e di tutti i dati ad esso associati
Endpoint: type=hotel&method=deleteHotelData
Metodo HTTP: POST
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo da eliminare

Descrizione: Aggiunta o Aggiornamento di un Albergo e delle sue Stanze
Endpoint: index.php?type=hotel&method=updateHotelData
Metodo HTTP: POST
Parametri:
- **Hotel** (obbligatorio): Albergo da aggiungere nel formato:
{"hotel": { "idHotel": "123", "name": "Nome dell'hotel", "image": "URL dell'immagine", "description": "Descrizione dell'hotel", "adminId": "456" }}
- **Rooms** (opzionale): Stanze dell'Albergo da aggiungere in formato:
{"room": [{ "name": "Nome stanza", "image": "URL immagine", "description": "Descrizione", "cost": "Costo", "totalRooms": "Totale Stanze" }, { "name": "Nome stanza", "image": "URL immagine", "description": "Descrizione", "cost": "Costo", "totalRooms": "Totale Stanze" }]}

API Stanze

Descrizione: Recupera tutte le Stanze di un Albergo
Endpoint: type=room&method=getHotelRooms&idHotel={idHotel}&today={today}
&tomorrow={tomorrow}
Metodo HTTP: GET
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo delle stanze da recuperare
- **Today** (opzionale): La data di inizio nel formato 'YYYY-MM-DD'
- **Tomorrow** (opzionale): La data di fine nel formato 'YYYY-MM-DD'

Esempio: Link API getHotelRooms

Descrizione: Recupera le Stanze di un Albergo raggruppate per nome
Endpoint: type=room&method=getHotelRooms&idHotel={idHotel}
&roomName={roomName}&startDate={startDate}&endDate={endDate}
Metodo HTTP: GET
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo delle stanze da recuperare
- **StartDate** (obbligatorio): La data di inizio nel formato 'YYYY-MM-DD'
- **EndDate** (obbligatorio): La data di fine nel formato 'YYYY-MM-DD'

Esempio: Link API getHotelRoom

Descrizione: Recupera le Stanze disponibili di un Albergo per Nome entro la data fornita
Endpoint: index.php?type=room&method=getBookingRoom
&idHotel={idHotel}&roomName={roomName}&startDate={startDate}
&endDate={endDate}
Metodo HTTP: GET
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo delle stanze da recuperare
- **StartDate** (obbligatorio): La data di inizio nel formato 'YYYY-MM-DD'
- **EndDate** (obbligatorio): La data di fine nel formato 'YYYY-MM-DD'
- **RoomName** (obbligatorio): Il Nome della Stanza da recuperare

Esempio: Link API getBookingRoom

Descrizione: Per la prima Stanza disponibili dal Nome della Stanza, la Data e l'Albergo
Endpoint: index.php?type=room&method=getFirstFreeRoom
&idHotel={idHotel}&roomName={roomName}&startDate={startDate}
&endDate={endDate}
Metodo HTTP: GET
Parametri:
- **IdHotel** (obbligatorio): L'ID dell'Albergo delle stanze da recuperare
- **StartDate** (obbligatorio): La data di inizio nel formato 'YYYY-MM-DD'
- **EndDate** (obbligatorio): La data di fine nel formato 'YYYY-MM-DD'
- **RoomName** (obbligatorio): Il Nome della Stanza da recuperare

Esempio: Link API getFirstFreeRoom

Links

Mockup APP Mobile	Link a Figma per Mockup App Mobile
Pagina Web Client	Link al Sito di Gestione Hotels
Pagina Web Server	Link al Server di Gestione Hotels

Considerazioni

Il Backend lato Server ed il Frontend lato Client del Sito Web sono stati progettati in modo che risultino totalmente indipendenti tra loro.
Entrambi sono caricati su host differenti ed il Server PHP permette la comunicazione tramite Endpoint con API specifiche per gestire le richieste HTTP del Client con diversi livelli di autorizzazione garantiti dalla gestione della sessione per tipo di Utente (Cliente o Admin) .
Ciò garantisce l'interoperabilità tra il Server ed i Client che possono essere sviluppati con diverse Tecnologie e su diverse Piattaforme.
Un esempio ne è la possibile realizzazione del Mockup proposto e pensato per essere un Applicazione Client sviluppata in React Native che sfrutterebbe del Server gli stessi Endpoint del Sito Web.

Casi d'Uso

Permessi utente	Schermate	Link Screenshots
Non Autenticato	Accesso o Registrazione	PC Mobile
	Lista degli Alberghi	PC Mobile
	Lista Stanze di un Albergo	PC Mobile
Autenticato come Cliente	Disconnessione o Elimina	PC Mobile
	Lista degli Alberghi	PC Mobile
	Stanze di un Albergo	PC Mobile
	Checkout	PC Mobile
	Prenotazioni del Cliente	PC Mobile
Autenticato come Admin	Disconnessione o Elimina	PC Mobile
	Alberghi e Aggiungi nuovo	PC Mobile
	Personalizza Dati Albergo	PC Mobile
	Prenotazioni dei Clienti di Admin	PC Mobile

Credenziali

Credenziali di Accesso al Database:

Nome Utente:	id22013064_admin
Password	Uva12345!
Link di Accesso:	phpMyAdmin