Progetto Ingegneria del Software – Gruppo Aereo 3

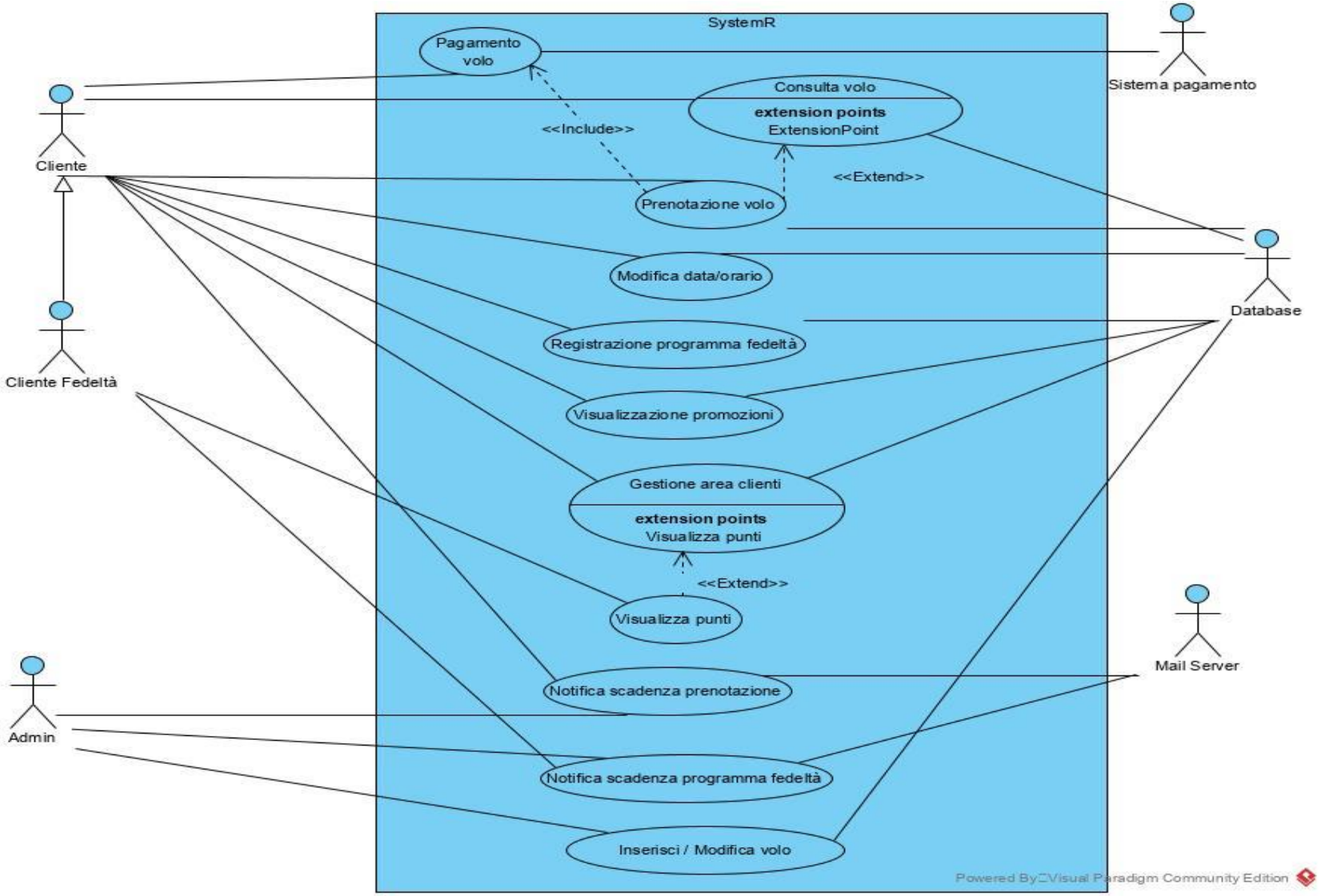
Membri :

- Alessio Fiore 821049
- Alessio Cottarelli 829659
- Clark Ezpeleta 832972
- Giovanni Gallo 829860
-

Assunzioni :

1. Pagamento della prenotazione gestito esternamente, verificata solo la correttezza dei dati.
2. I voli non vengono cancellati per estratto azienda.
3. Numero massimo di posti per prenotazione impostato a 3.
4. Controllo della necessità di inviare e-mail di notifica all'accesso dell'Admin (ad esempio nuove promozioni o scadenze).
5. Compagnia che vende solo biglietti per voli in Italia

1. Diagramma Casi D’uso :



1.1. Attori primari :

- Cliente
- Cliente Fedele
- Admin

1.2. Attori di sistema :

- Database
- Mail Server
- Sistema Pagamento

1.3. Requisiti Funzionali :

1.3.1. Pagamento volo

Il sistema deve permettere al cliente di poter effettuare il pagamento della prenotazione tramite Carta di Credito o punti fedeltà.

1.3.2. Consulta volo

Il sistema deve permettere di consultare i voli per una determinata data.

1.3.3. Prenotazione volo

Il sistema deve consentire al cliente di prenotare dei posti per un determinato volo.

1.3.4. Modifica data/orario

Il sistema deve consentire al cliente di poter modificare la data e l'orario di un volo prenotato.

1.3.5. Registrazione programma fedeltà

Il sistema deve consentire al cliente di registrarsi al programma fedeltà

1.3.6. Visualizzazione promozioni

Il sistema deve consentire al cliente di poter visualizzare l'elenco dei voli alla quale è associata una promozione.

1.3.7. Visualizza punti

Il sistema deve consentire al cliente iscritto al programma fedeltà di poter visualizzare l'estratto punti.

1.3.8. Notifica scadenza prenotazione

Il sistema deve notificare le scadenze delle prenotazioni al momento del login dell'admin.

1.3.9. Notifica scadenza programma fedeltà

Il sistema deve notificare le scadenze delle iscrizioni al programma fedeltà al momento del login dell'admin.

1.3.10. Inserisci/Modifica volo

Il sistema deve consentire all'admin di poter inserire nuovi voli e di modificare quelli già esistenti.

2. Caso d’uso dettagliato : **Prenotazione volo**

2.1. Portata :

Gestione prenotazione.

2.2. Livello :

Obiettivo utente.

2.3. Attore primario :

Cliente/Cliente fedele.

2.4. Parte interessata e interessi :

L’agenzia vuole aumentare il quantitativo di Clienti.

2.5. Precondizione :

Il volo selezionato deve esistere.
Il cliente non deve possedere una prenotazione per quel volo.

2.6. Garanzia di successo :

Il cliente avrà una prenotazione per il volo scelto.
Inviata una mail di notifica di prenotazione effettuata al Cliente.

2.7. Scenario principale di successo :

L’utente cerca il volo rispetto ad una determinata data e ad un numero di posti

L’utente sceglie il volo in base agli orari disponibili

2.7.1. Se il cliente è fedele :

Effettua il login inserendo la mail e la password e gli verrà associata la prenotazione. Successivamente inserisce i dati di tutti i passeggeri.

2.7.2. Se il cliente non è fedele :

Inserisce il proprio nome, cognome, mail, data di nascita e il nome e cognome di tutti gli altri passeggeri.

Il cliente seleziona i posti relativi al volo e sceglie se pagarlo subito o se pagarlo successivamente rispettando le scadenze.

Il sistema assegna una prenotazione al cliente e gli inoltra una mail di notifica di avvenuta prenotazione.

2.8. Estensioni :

L’utente inserisce una data non valida nella ricerca di un volo, ad esempio una data passata.

⇨ *Il sistema mostra un messaggio di errore.*

L’utente ricerca un volo non programmato per la data selezionata.

⇨ *Il sistema mostra un messaggio di errore.*

L’utente effettua una prenotazione per un volo per la quale ha già effettuato una prenotazione.

⇨ *Il sistema mostra un messaggio di errore e non fa procedere con la scelta dei posti.*

L’utente inserisce i dati non validi relativi al metodo di pagamento, se decide di effettuare subito il pagamento.

⇨ *Il sistema mostra un messaggio di errore.*

2.9. Requisiti speciali :

L’utente deve possedere una connessione internet funzionante.

2.10. Elenco varianti tecnologiche e dei dati :

Utilizzo Database remoto su AWS.

2.11. Frequenza di ripetizione :

Ogni volta che l’utente vuole effettuare una prenotazione.

3. Caso d’uso dettagliato : **Modifica prenotazione**

3.1. Portata :
Gestione Prenotazione

3.2. Livello :
Obiettivo utente

3.3. Attore primario :
Cliente/Cliente fedele

3.4. Parte interessata e interessi :
L’agenzia vuole permettere ai clienti di modificare le proprie prenotazioni pagando un sovrapprezzo

3.5. Precondizione :
Il cliente deve aver effettuato almeno una prenotazione

3.6. Garanzia di successo :
Il cliente ha una nuova prenotazione associata al nuovo volo scelto.

3.7. Scenario principale di successo :

L’utente visualizza la propria prenotazione inserendo la propria mail e l’ID della vecchia prenotazione.
L’utente ricerca il volo rispetto ad una nuova data.
L’utente sceglie il volo in base agli orari disponibili.
L’utente sceglie i posti sul nuovo volo, senza la possibilità di modificarne il numero.
L’utente procede con il pagamento del sovrapprezzo.

3.8. Estensioni :

L’utente inserisce una mail errata o un ID prenotazione non esistente.

⇨ *Il sistema non permette l’accesso alla schermata di modifica prenotazione.*

L’utente ricerca un volo che non è disponibile nella nuova data selezionata.

⇨ *Il sistema mostra un messaggio di errore.*

L’utente inserisce i dati non validi relativi al metodo di pagamento del sovrapprezzo.

⇨ *Il sistema mostra un messaggio di errore.*

3.9. Requisiti speciali :

L’utente deve possedere una connessione internet funzionante.

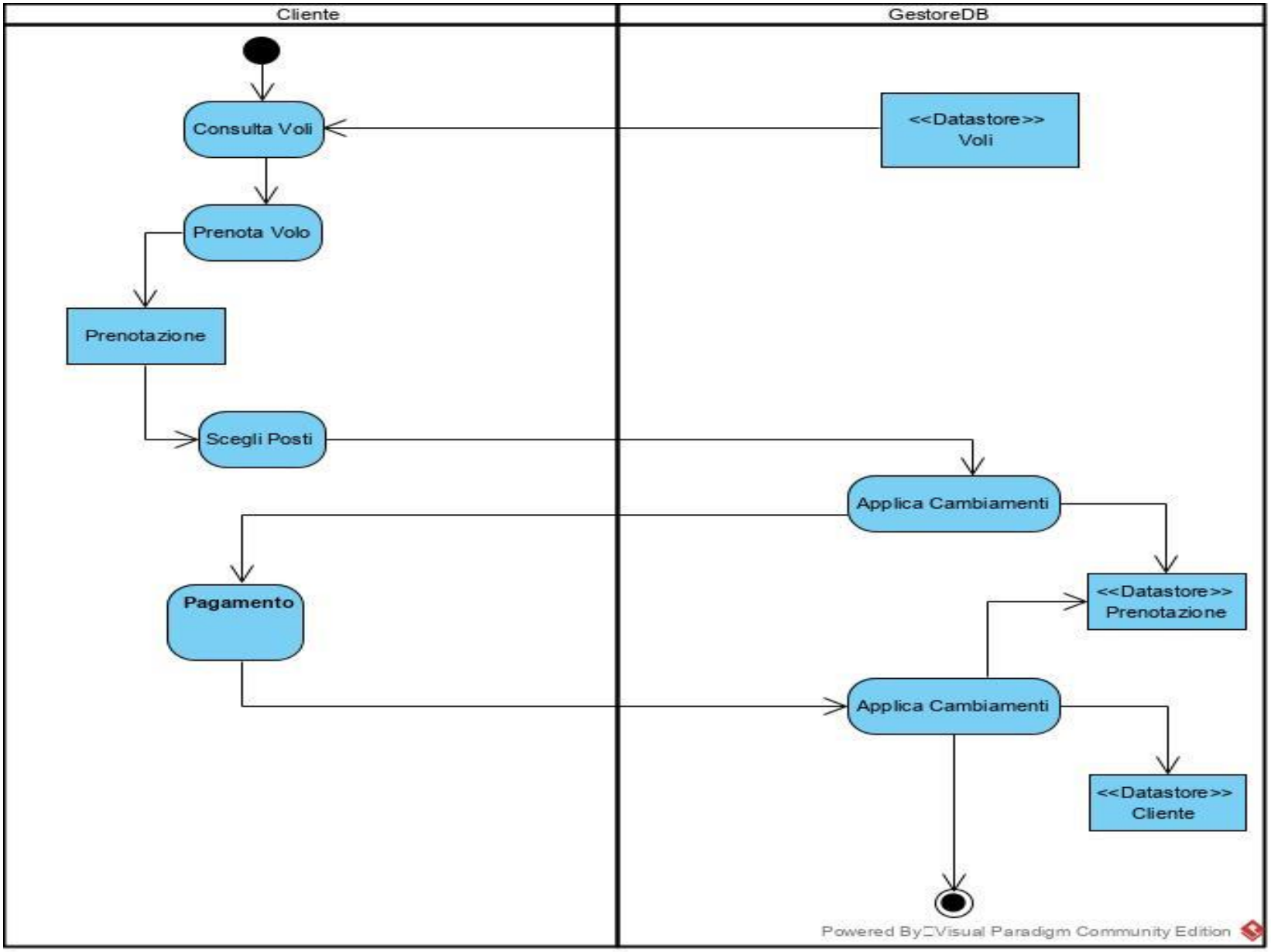
3.10. Elenco varianti tecnologiche e dei dati :

Utilizzo Database remoto su AWS.

3.11. *Frequenza di ripetizione :*

Ogni volta che l’utente vuole modificare una prenotazione.

4. Diagramma delle Attività : Modifica Prenotazione



5. Caso d’uso dettagliato : Nuova promozione

5.1. Portata :

Gestione Promozione

5.2. Livello :

Admin

5.3. Attore primario :

Admin

5.4. Parte interessata e interessi :

L’agenzia vuole inserire una nuova promozione legata ad un volo e ad un periodo di tempo.

5.5. Precondizione :

Esistenza di almeno un volo

5.6. Garanzia di successo :

La promozione viene associata a uno o più voli che hanno partenza e destinazioni uguali a quelle scelte dall’ Admin.

5.7. Scenario principale di successo :

L’Admin effettua il login e inserisce una nuova promozione scegliendo la partenza, destinazione, periodo e la percentuale di sconto.
L’Admin sceglie anche se la promozione sarà attiva per tutti o solo per i Clienti iscritti al programma fedeltà.

5.8. Estensioni :

L’admin inserisce i dati del login errati.

⇨ *Il sistema mostra un messaggio di errore.*

L’admin inserisce una data di inizio o di fine promozione non validi, ad esempio date passate.

⇨ *Il sistema mostra un messaggio di errore.*

L’admin inserisce uno sconto non valido

⇨ *Il sistema mostra un messaggio di errore.*

5.9. Requisiti speciali :

L’utente deve possedere una connessione internet funzionante.

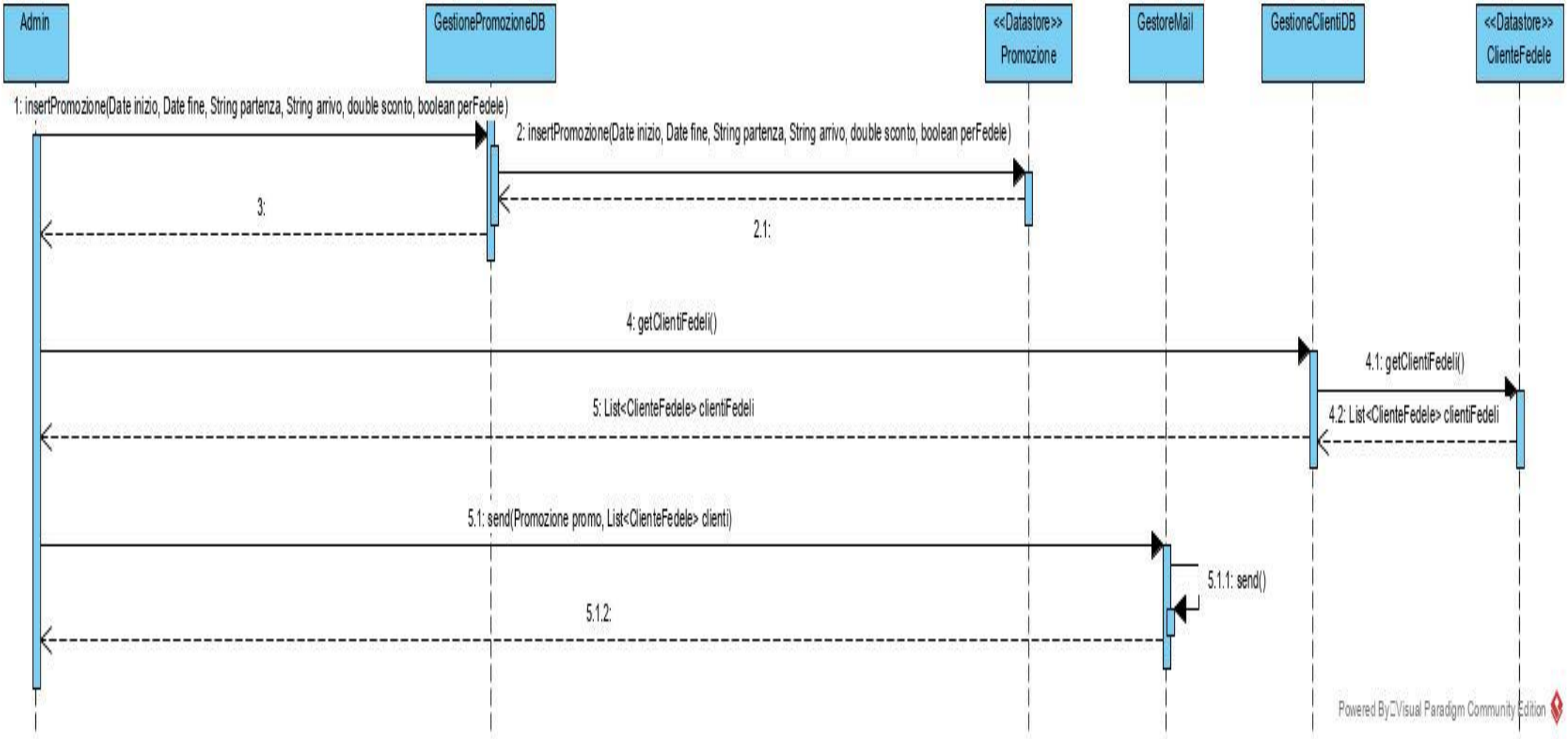
5.10. Elenco varianti tecnologiche e dei dati :

Utilizzo Database remoto su AWS.

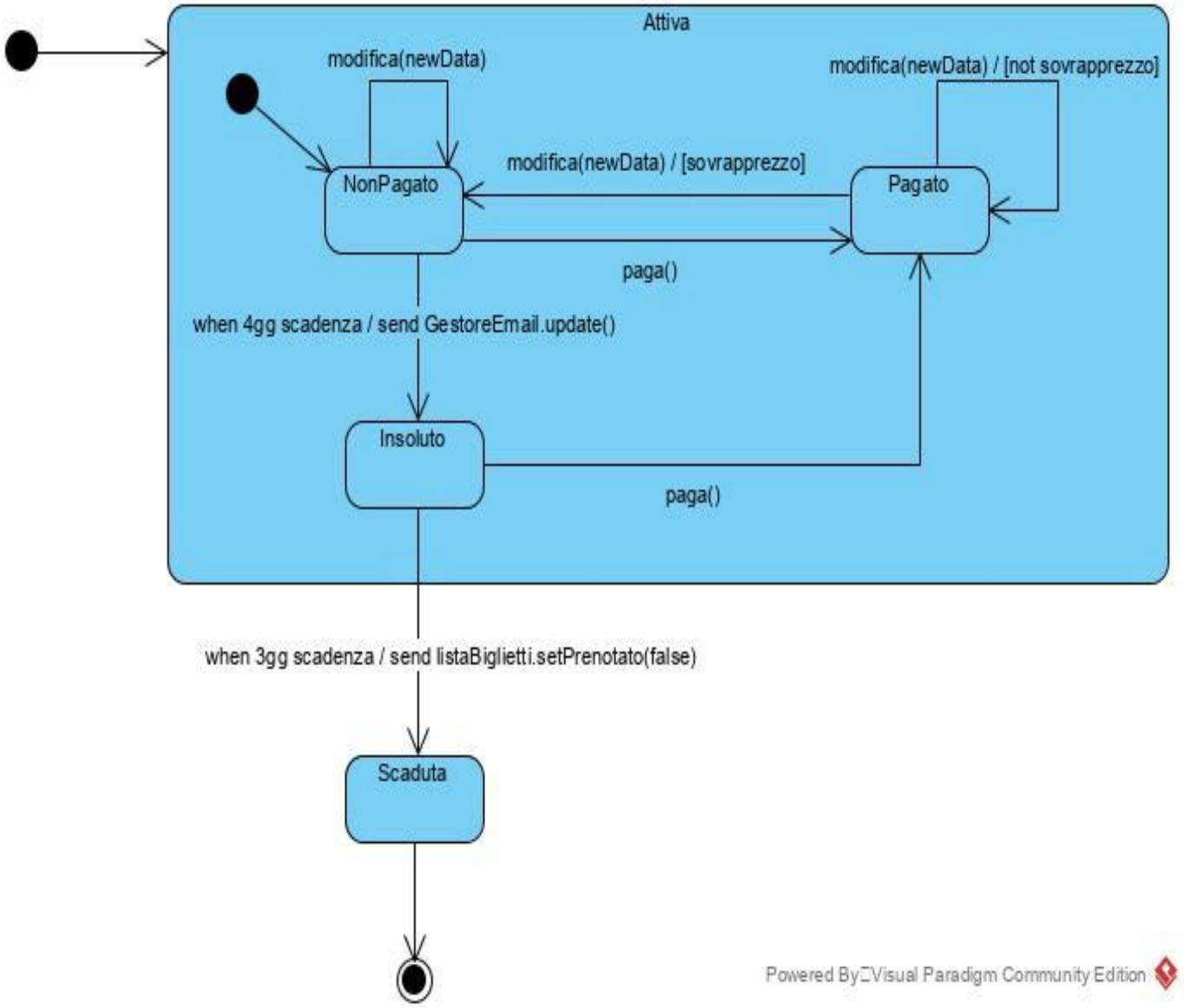
5.11. *Frequenza di ripetizione :*

Ogni volta che l’utente vuole effettuare il login come Admin.

6. Diagramma di Sequenza : Nuova Promozione



7. Diagramma a Stati : Gestione prenotazione



8. Modello di Dominio

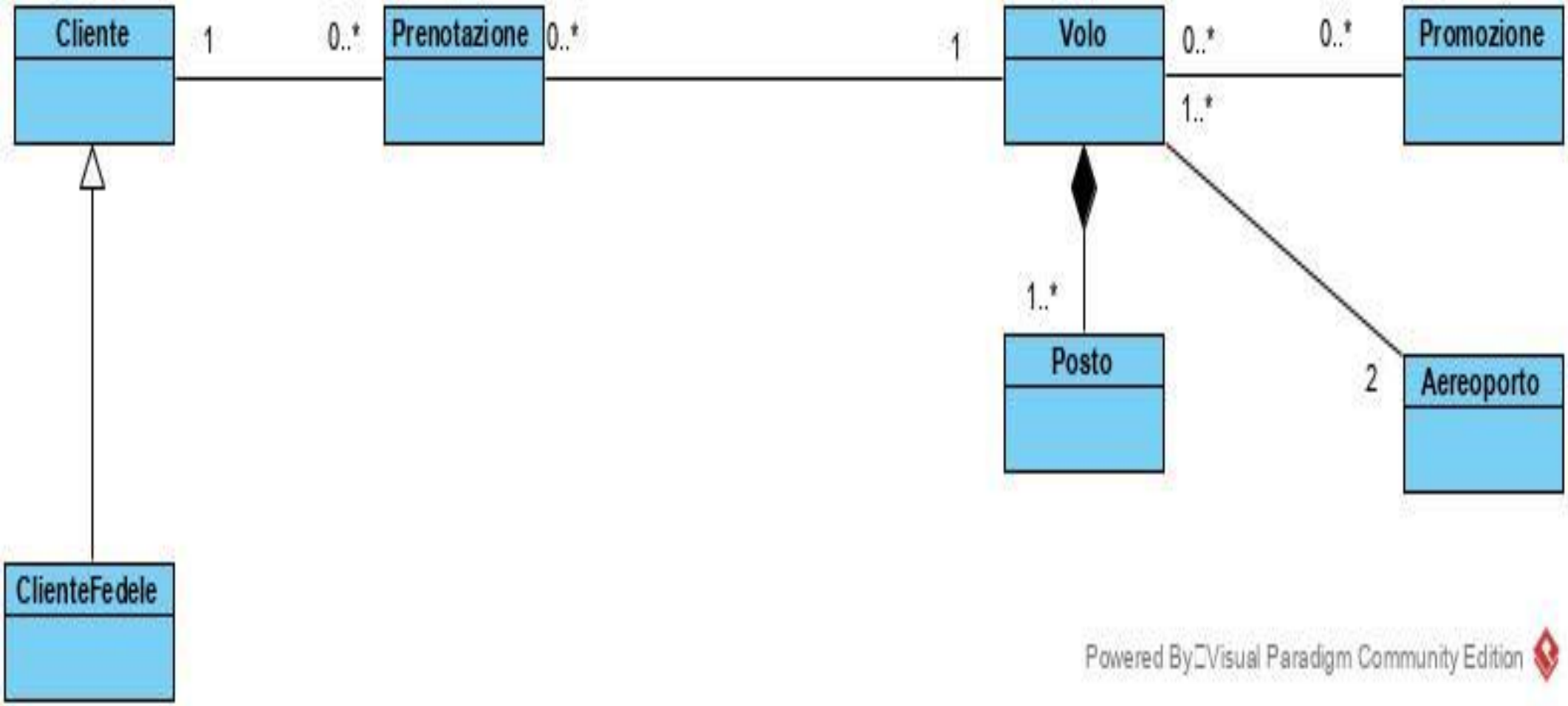
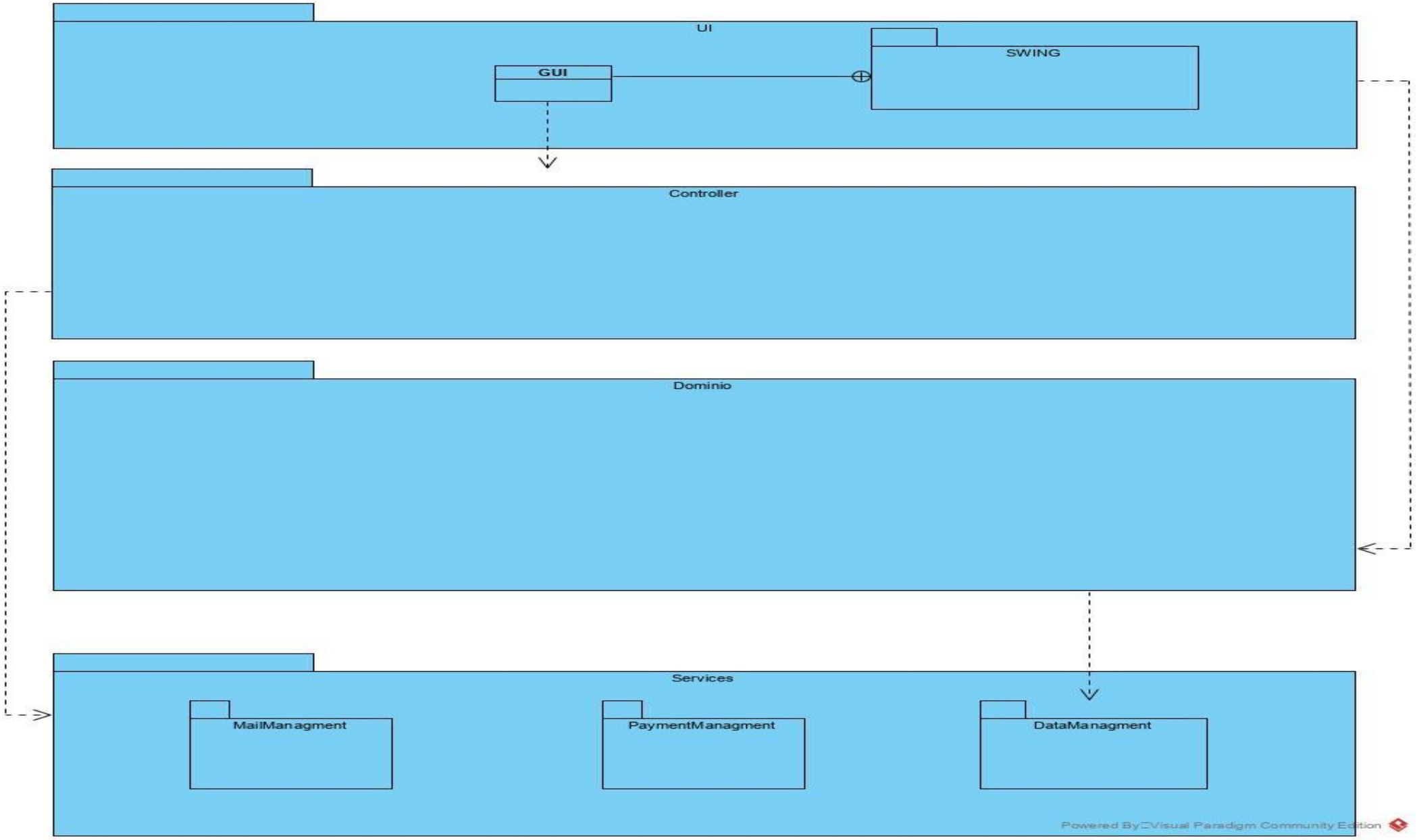
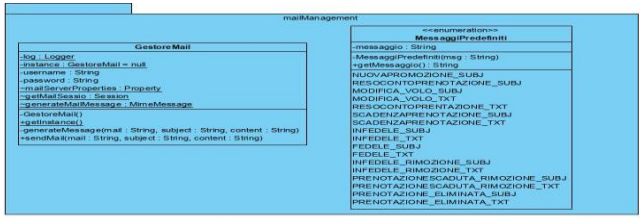
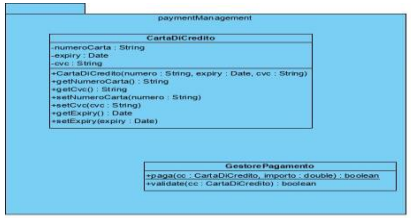


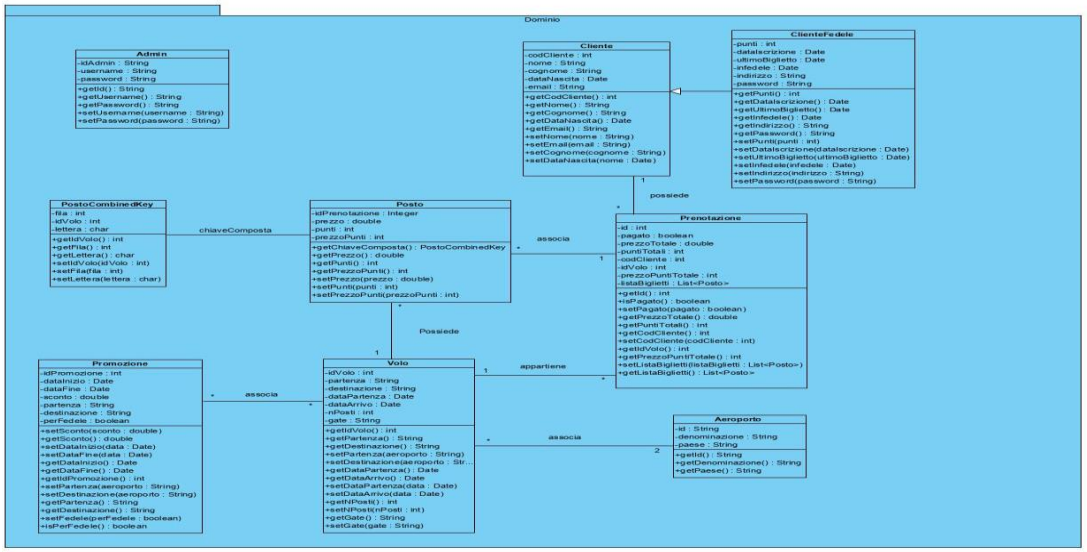
Diagramma di Architettura Software

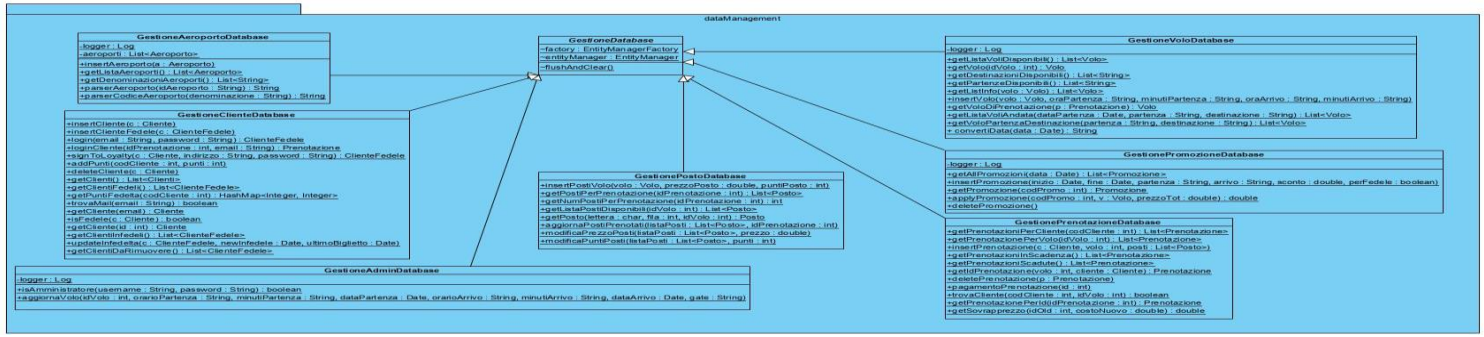


9. Diagramma delle Classi

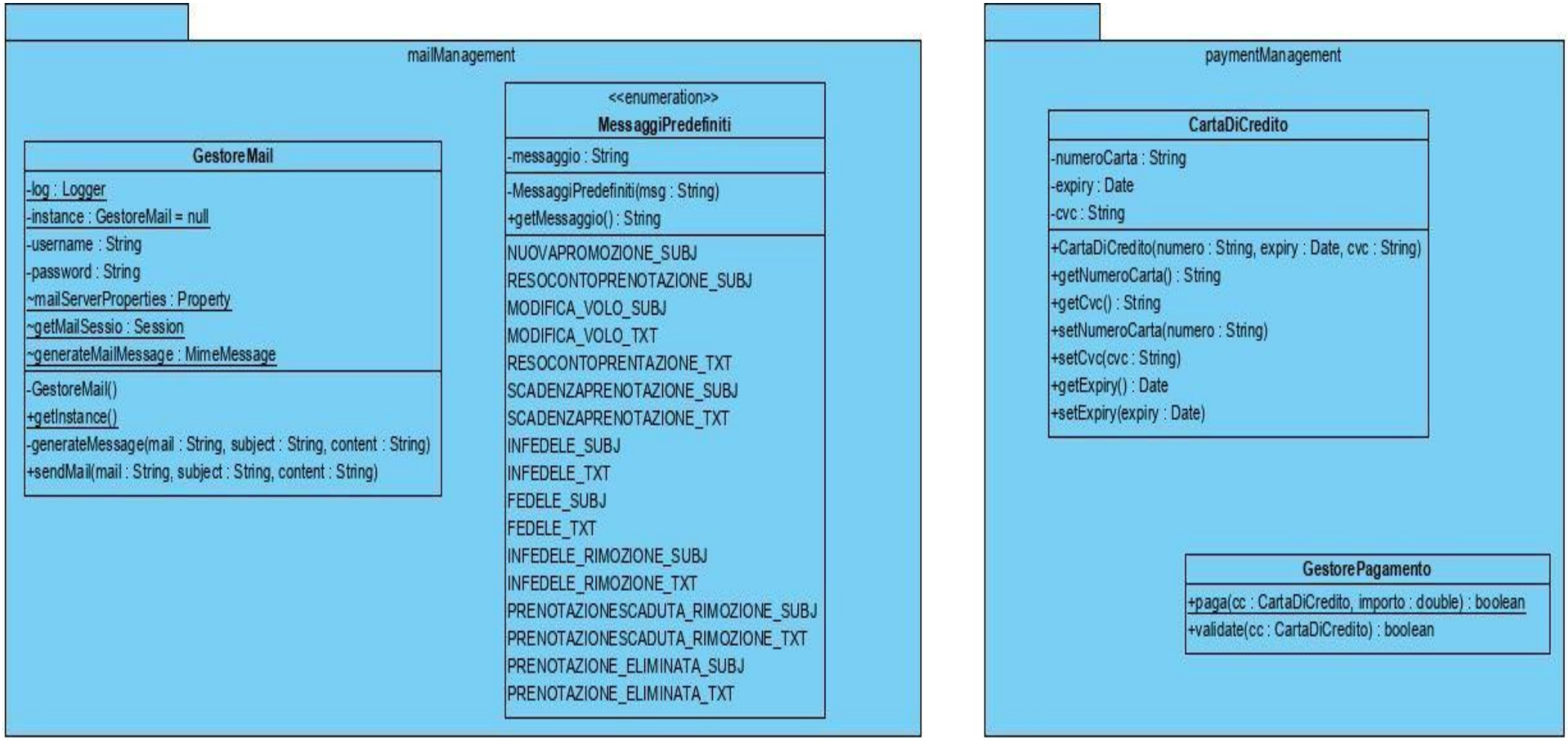






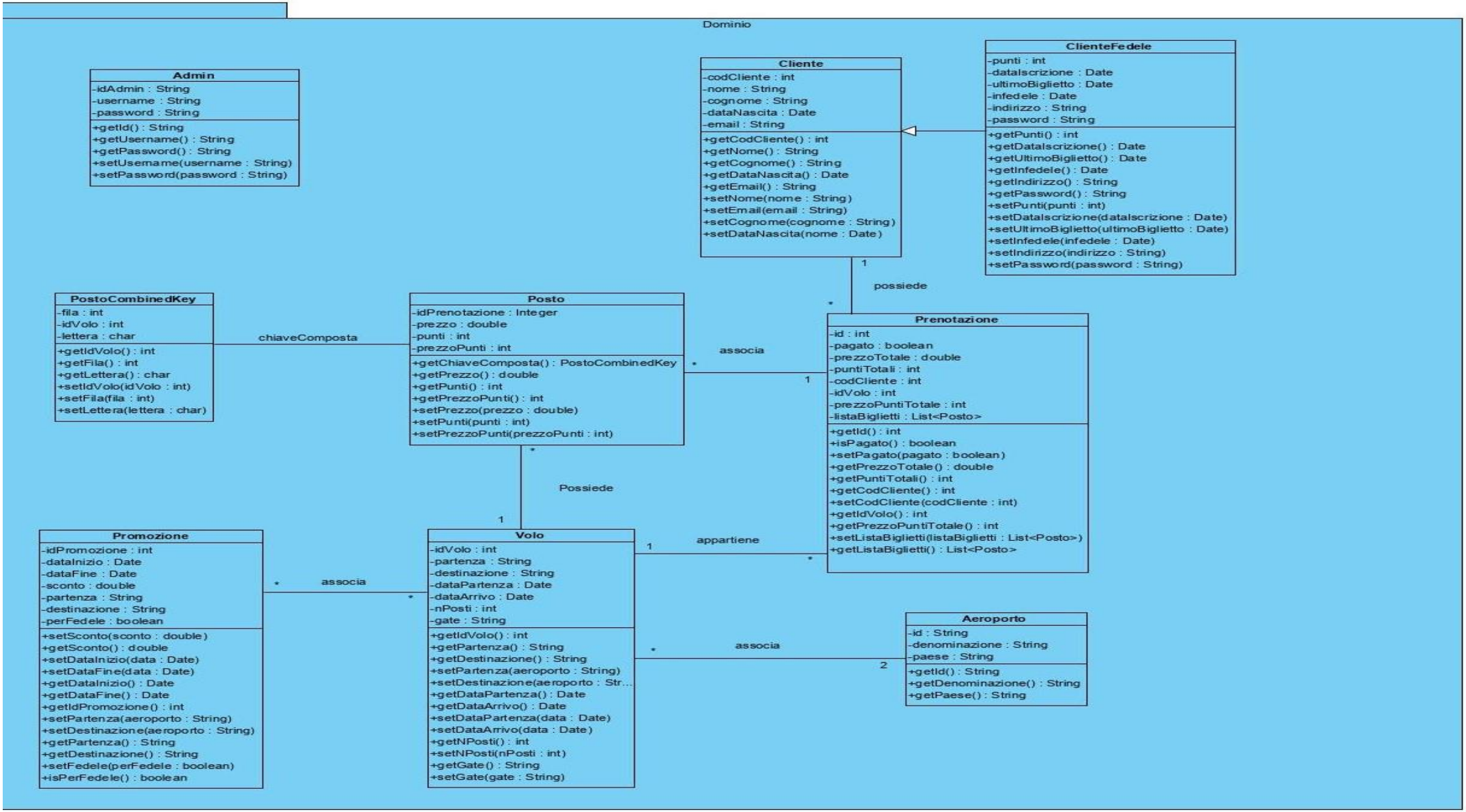


9.1. Mail Management e Payment Management

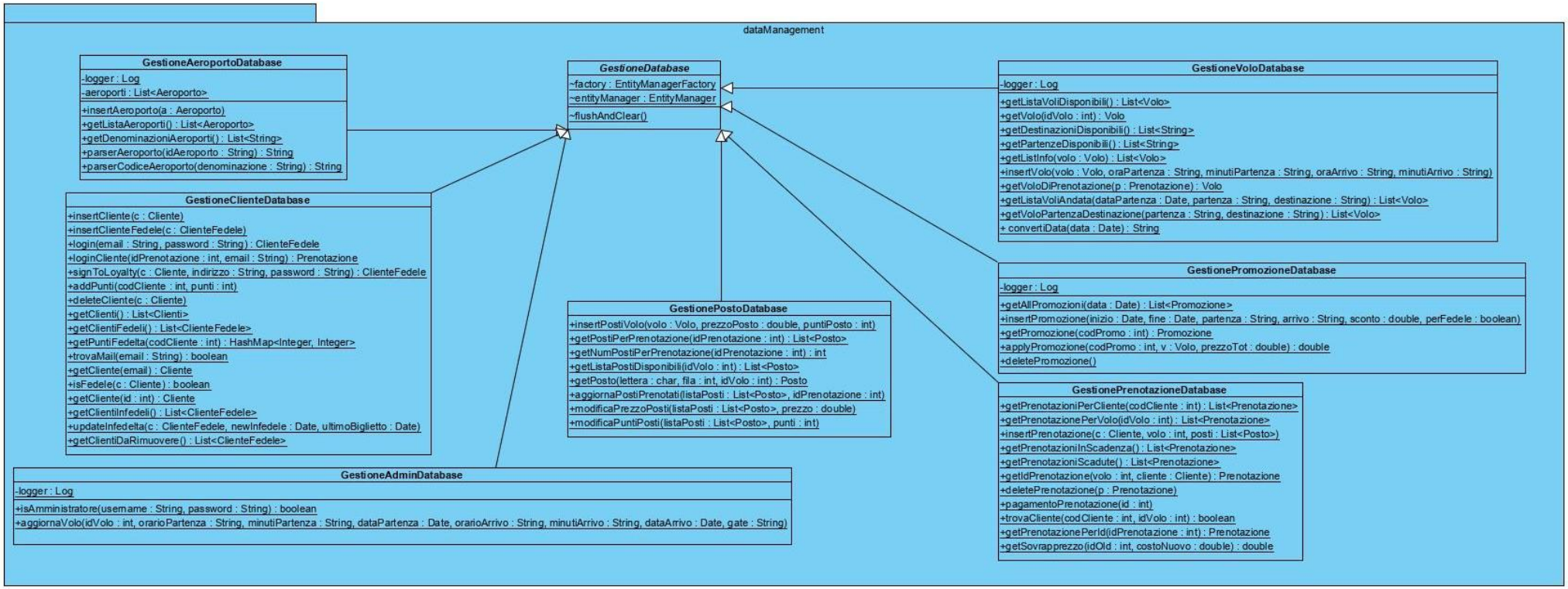


9.2. Controller

9.3. Dominio


Dominio
Admin
-idAdmin : String
-username : String
-password : String
+getId() : String
+getUsername() : String
+getPassword() : String
+setUsername(username : String)
+setPassword(password : String)
Cliente
-codCliente : int
-nome : String
-cognome : String
-dataNascita : Date
-email : String
+getCodCliente() : int
+getNome() : String
+getCognome() : String
+getDataNascita() : Date
+getEmail() : String
+setNome(nome : String)
+setEmail(email : String)
+setCognome(cognome : String)
+setDataNascita(nome : Date)
ClienteFedele
-punti : int
-dataIscrizione : Date
-ultimoBiglietto : Date
-infedele : Date
-indirizzo : String
-password : String
+getPunti() : int
+getDataIscrizione() : Date
+getUltimoBiglietto() : Date
+getInfedele() : Date
+getIndirizzo() : String
+getPassword() : String
+setPunti(punti : int)
+setDataIscrizione(dataIscrizione : Date)
+setUltimoBiglietto(ultimoBiglietto : Date)
+setInfedele(infedele : Date)
+setIndirizzo(indirizzo : String)
+setPassword(password : String)
1
possiede
*
PostoCombinedKey
-fila : int
-idVolo : int
-lettera : char
+getIdVolo() : int
+getFila() : int
+getLettera() : char
+setIdVolo(idVolo : int)
+setFila(fila : int)
+setLettera(lettera : char)
chiaveComposta
Posto
-idPrenotazione : Integer
-prezzo : double
-punti : int
-prezzoPunti : int
+getChiaveComposta() : PostoCombinedKey
+getPrezzo() : double
+getPunti() : int
+getPrezzoPunti() : int
+setPrezzo(prezzo : double)
+setPunti(punti : int)
+setPrezzoPunti(prezzoPunti : int)
*
associa
1
Prenotazione
-id : int
-pagato : boolean
-prezzoTotale : double
-puntiTotali : int
-codCliente : int
-idVolo : int
-prezzoPuntiTotale : int
-listaBiglietti : List<Posto>
+getId() : int
+isPagato() : boolean
+setPagato(pagato : boolean)
+getPrezzoTotale() : double
+getPuntiTotali() : int
+getCodCliente() : int
+setCodCliente(codCliente : int)
+getIdVolo() : int
+getPrezzoPuntiTotale() : int
+setListaBiglietti(listaBiglietti : List<Posto>)
+getListaBiglietti() : List<Posto>
*
Possiede
1
Promozione
-idPromozione : int
-dataInizio : Date
-dataFine : Date
-sconto : double
-partenza : String
-destinazione : String
-perFedele : boolean
+setSconto(sconto : double)
+getSconto() : double
+setDataInizio(data : Date)
+setDataFine(data : Date)
+getDataInizio() : Date
+getDataFine() : Date
+getIdPromozione() : int
+setPartenza(aeroporto : String)
+setDestinazione(aeroporto : String)
+getPartenza() : String
+getDestinazione() : String
+setFedele(perFedele : boolean)
+isPerFedele() : boolean
*
associa
*
Volo
-idVolo : int
-partenza : String
-destinazione : String
-dataPartenza : Date
-dataArrivo : Date
-nPosti : int
-gate : String
+getIdVolo() : int
+getPartenza() : String
+getDestinazione() : String
+setPartenza(aeroporto : String)
+setDestinazione(aeroporto : Str...
+getDataPartenza() : Date
+getDataArrivo() : Date
+setDataPartenza(data : Date)
+setDataArrivo(data : Date)
+getNPosti() : int
+setNPosti(nPosti : int)
+getGate() : String
+setGate(gate : String)
1
appartiene
*
*
associa
2
Aeroporto
-id : String
-denominazione : String
-paese : String
+getId() : String
+getDenominazione() : String
+getPaese() : String

9.4. Data Management



10. Design Pattern e Architectural Pattern

Facade Controller:

Utilizzato per chiamare i metodi della gestione database che servono alle view per effettuare le operazioni richieste.

Singleton:

Utilizzato all'interno della classe "Gestore Mail". Usato poichè il gestore delle mail è univoco e una volta inizializzato non è necessario istanziare un nuovo oggetto dello stesso tipo.

MVC: Model View Controller

Table Data Gateway: usato per la mappatura nel DB

Foreign Key Mapping: Utilizzato per rappresentare il riferimento tra diversi oggetti del Database

11. Design Principle

ISP: I client sono stati classificati in base al loro tipo e sono state create interfacce per ogni tipo di client.