UNIVERSITÀ DEGLI STUDI DI NAPOLI FEDERICO II

CORSO DI LAUREA IN INGEGNERIA INFORMATICA

CORSO DI INGEGNERIA DEL SOFTWARE

PROF. A.R. FASOLINO - A.A. 2022 – 23

Progetto

Italian Travel
Prenotazione camere d'albergo

Studenti:

Martina	**Borgstrom**	N46005697	m.borgstrom@studenti.unina.it
Riccardo	**Brescia**	N46005663	ri.brescia@studenti.unina.it
Alessandro	**Campanella**	N46005580	ale.campanella@studenti.unina.it
Rita	**Castaldi**	N46005530	ri.castaldi@studenti.unina.it

Versione del 05/07/2023

INDICE

1. SPECIFICHE INFORMALI

Si vuole realizzare un'applicazione web per la gestione delle prenotazioni di camere d'albergo. Il sistema gestisce più catene di alberghi su tutto il territorio nazionale.

Ogni catena alberghiera, identificata da un codice e da un nome, possiede uno o più alberghi. A ciascun albergo è assegnato l'identificativo, il nome, l'indirizzo, CAP e il numero di telefono. Ogni albergo possiede diverse tipologie di camere: singole, doppie e triple. Tutte le camere sono identificate da un numero di camera. Ciascuna tipologia di camera è caratterizzata da un prezzo per notte. Inoltre, per ciascuna camera si vuole memorizzare lo stato (disponibile, prenotata, occupata).

Il sistema consente agli utenti di verificare la disponibilità di camere ed eventualmente di effettuare una prenotazione. In una prenotazione si può prenotare una sola camera. In particolare, per cercare una camera l'utente inserisce la città, la data di arrivo, la data di partenza e la tipologia di camera; il sistema cerca tutti gli alberghi (di tutte le catene) che si trovano nella città indicata e che hanno camere disponibili per il periodo indicato, e restituisce all'utente la lista di alberghi disponibili. Se decide di prenotare, il cliente fornisce gli ulteriori dati richiesti (nome, cognome, e-mail, telefono, indirizzo, e un numero di carta di credito a garanzia), che vengono memorizzati nel sistema. Per ciascuna prenotazione il sistema memorizza un codice di prenotazione, la data di arrivo, di partenza, e il numero della stanza assegnata. Al termine della prenotazione, il cliente riceve una mail di conferma, che ricapitola tutti i dati della prenotazione effettuata (incluso il prezzo complessivo).

Il sistema consente al personale della reception di effettuare il check-in e il check-out dei clienti che hanno effettuato la prenotazione, e di stampare la fattura al termine del soggiorno.

Il primo giorno di ogni mese il sistema invia per e-mail al Direttore di ogni catena d'alberghi un report con l'elenco delle notti in cui ciascuna camera è stata occupata nel mese precedente.

2. ANALISI E SPECIFICA DEI REQUISITI

2.1. Analisi nomi-verbi

Si vuole realizzare un'applicazione web per la gestione delle prenotazioni di camere d'albergo. Il sistema gestisce più catene di alberghi su tutto il territorio nazionale.

Ogni **catena alberghiera**, identificata da un **codice** e da un **nome**, possiede uno o più alberghi. A ciascun **albergo** è assegnato l'**identificativo**, il **nome**, l'**indirizzo**, **CAP** e il **numero di telefono**. Ogni albergo possiede diverse tipologie di **camere**: **singole**, **doppie** e **triple**. Tutte le camere sono identificate da un **numero di camera**. Ciascuna tipologia di camera è caratterizzata da un **prezzo per notte**. Inoltre, per ciascuna camera si vuole memorizzare lo **stato** (disponibile, prenotata, occupata).

Il sistema consente agli utenti di verificare la disponibilità di camere ed eventualmente di **effettuare una prenotazione**. In una prenotazione si può prenotare una sola camera. In particolare, per cercare una camera l'**utente** inserisce la **città**, la **data di arrivo**, la **data di partenza** e la **tipologia di camera**; **il sistema cerca tutti gli alberghi (di tutte le catene)** che si trovano nella città indicata e che hanno camere disponibili per il periodo indicato, e **restituisce all'utente la lista di alberghi disponibili**. Se decide di prenotare, il **cliente** **fornisce gli ulteriori dati richiesti** (**nome**, **cognome**, **e-mail**, **telefono**, **indirizzo**, e un **numero di carta di credito** a garanzia), che vengono memorizzati nel sistema. Per ciascuna **prenotazione** il sistema memorizza un **codice di prenotazione**, la **data di arrivo**, **di partenza**, e il **numero della stanza assegnata**. Al termine della prenotazione, il cliente **riceve una mail di conferma**, che ricapitola tutti i dati della prenotazione effettuata (incluso il prezzo complessivo).

Il sistema consente al **personale della reception** di **effettuare il check-in** e **il check-out** dei clienti che hanno effettuato la prenotazione, e di **stampare la fattura** al termine del soggiorno.

Il primo giorno di ogni mese **il sistema invia per e-mail** al **Direttore** di ogni catena d'alberghi **un report con l'elenco delle notti** in cui ciascuna camera è stata occupata nel mese precedente.

LEGENDA:

- **Classe**
- **Attributo**
- **Attore**
- **Classe-Attore**
- **Funzionalità**

2.2. Revisione dei requisiti

1. Ogni catena alberghiera deve essere identificata da un codice e un nome
2. Ogni catena alberghiera possiede uno o più alberghi
3. A ciascun albergo è assegnato l'identificativo, il nome, l'indirizzo, CAP e il numero di telefono
4. Ogni albergo possiede diverse tipologie di camere
5. Le camere sono singole, doppie e triple
6. Tutte le camere sono identificate da un numero di camera
7. Ciascuna tipologia di camera è caratterizzata da un prezzo per notte
8. Per ciascuna camera si vuole memorizzare lo stato
9. Il sistema deve offrire agli utenti di verificare la disponibilità di camere
10. Il sistema deve offrire agli utenti la possibilità di effettuare una prenotazione
11. Ogni prenotazione deve includere al più una camera
12. Il sistema deve offrire all'utente una funzionalità per cercare una camera, inserendo città, data di arrivo, data di partenza e tipologia di camera
13. Il sistema deve garantire la possibilità di cercare tutti gli alberghi che si trovano nella città indicata e che hanno camere disponibili per il periodo indicato
14. Il sistema deve restituire all'utente la lista degli alberghi disponibili
15. Se si effettua una prenotazione, il cliente deve fornire nome, cognome, e-mail, numero di telefono, indirizzo, numero di carta di credito a garanzia
16. Il sistema deve memorizzare gli ulteriori dati richiesti al cliente, quali nome, cognome, e-mail, numero di telefono, indirizzo, numero di carta di credito a garanzia
17. Il sistema deve memorizzare per ciascuna prenotazione un codice di prenotazione, data di arrivo, partenza, numero della stanza assegnata
18. Il sistema deve permettere l'invio di una mail di conferma al cliente, elencando tutti i dati della prenotazione effettuata, incluso il prezzo complessivo
19. Per l'invio del riepilogo della prenotazione al cliente, deve essere disponibile un server di posta elettronica esterno al sistema
20. Il sistema deve offrire al personale della reception una funzionalità per effettuare il check-in dei clienti che hanno effettuato una prenotazione

21. Il sistema deve offrire al personale della reception una funzionalità per effettuare il check-out dei clienti che hanno effettuato una prenotazione
22. Il sistema deve offrire al personale della reception una funzionalità per stampare la fattura al termine del soggiorno
23. Per la stampa a fine soggiorno della fattura, deve essere presente un dispositivo di stampa esterno al sistema
24. Il sistema deve comporre una e-mail da inviare al Direttore di ogni catena d'alberghi per generare un report con l'elenco delle notti in cui ciascuna camera è stata occupata nel mese precedente
25. Il report mensile viene generato ogni primo giorno di un mese
26. Per l'invio mensile del report, deve essere disponibile un server di posta elettronica esterno al sistema

2.3. Glossario dei termini

Termine	Descrizione	Sinonimi
Catena alberghiera	Gruppi aziendali che raggruppano più alberghi sotto lo stesso marchio	-
Utente	Colui che controlla le disponibilità degli alberghi	-
Cliente	Colui che ha effettuato la procedura di prenotazione	Cliente registrato
Personale della reception	Colui che effettua il check-in e check-out dei clienti e si occupa di stampare la fattura al termine del soggiorno	-
Report	Documento che include i dati estratti sulle camere prenotate nel mese precedente per ogni catena alberghiera	-
Direttore	Colui che gestisce una catena alberghiera	-

2.4. Classificazione dei requisiti

2.4.1. Requisiti funzionali

ID	Requisito	Origine
RF01	Il sistema deve offrire agli utenti di verificare la disponibilità di camere	9
RF02	Il sistema deve offrire agli utenti la possibilità di effettuare una prenotazione	10
RF03	Il sistema deve offrire all'utente una funzionalità per cercare una camera, inserendo città, data di arrivo, data di partenza e tipologia di camera	12
RF04	Il sistema deve garantire la possibilità di cercare tutti gli alberghi che si trovano nella città indicata e che hanno camere disponibili per il periodo indicato	13
RF05	Il sistema deve restituire all'utente la lista degli alberghi disponibili	14
RF06	Se si effettua una prenotazione, il cliente deve fornire nome, cognome, e-mail, numero di telefono, indirizzo, numero di carta di credito a garanzia	15
RF07	Il sistema deve permettere l'invio di una mail di conferma al cliente, elencando tutti i dati della prenotazione effettuata, incluso il prezzo complessivo	18
RF08	Il sistema deve offrire al personale della reception una funzionalità per effettuare il check-in dei clienti che hanno effettuato una prenotazione	20
RF09	Il sistema deve offrire al personale della reception una funzionalità per effettuare il check-out dei clienti che hanno effettuato una prenotazione	21

RF10	Il sistema deve offrire al personale della reception una funzionalità per stampare la fattura al termine del soggiorno	22
RF11	Il sistema deve comporre una e-mail da inviare al Direttore di ogni catena d'alberghi per generare un report con l'elenco delle notti in cui ciascuna camera è stata occupata nel mese precedente	24

2.4.2. Requisiti sui dati

ID	Requisito	Origine
RD01	Ogni catena alberghiera deve essere identificata da un codice e un nome	1
RD02	Ogni catena alberghiera possiede uno o più alberghi	2
RD03	A ciascun albergo è assegnato l'identificativo, il nome, l'indirizzo, CAP e il numero di telefono	3
RD04	Ogni albergo possiede diverse tipologie di camere	4
RD05	Le camere sono singole, doppie e triple	5
RD06	Tutte le camere sono identificate da un numero di camera	6
RD07	Ciascuna tipologia di camera è caratterizzata da un prezzo per notte	7
RD08	Per ciascuna camera si vuole memorizzare lo stato	8

2.4.3. Vincoli / Altri requisiti

ID	Requisito	Origine
V01	Ogni prenotazione deve includere al più una camera	11
V02	Il report mensile viene generato ogni primo giorno di un mese	25
RNF01	Il sistema deve memorizzare gli ulteriori dati richiesti al cliente, quali nome, cognome, e-mail, numero di telefono, indirizzo, numero di carta di credito a garanzia	16
RNF02	Il sistema deve memorizzare per ciascuna prenotazione un codice di prenotazione, data di arrivo, partenza, numero della stanza assegnata	17
RNF03	Per l'invio del riepilogo della prenotazione al cliente, deve essere disponibile un server di posta elettronica esterno al sistema	19
RNF04	Per la stampa a fine soggiorno della fattura, deve essere presente un dispositivo di stampa esterno al sistema	23
RNF05	Per l'invio mensile del report, deve essere disponibile un server di posta elettronica esterno al sistema	26

2.5. Modellazione dei casi d'uso

2.5.1. Attori e casi d'uso

ATTORI PRIMARI:

- Utente
- Cliente
- Personale Reception
- Tempo

ATTORI SECONDARI:

- Servizio E-mail
- Direttore

CASI D'USO:

- UC1 : VerificaDisponibilitàCamera
- UC2 : EffettuaPrenotazione
- UC3 : EffettuaCheckIn
- UC4 : EffettuaCheckOut
- UC5 : GeneraReportElencoNotti

CASI D'USO D'INCLUSIONE:

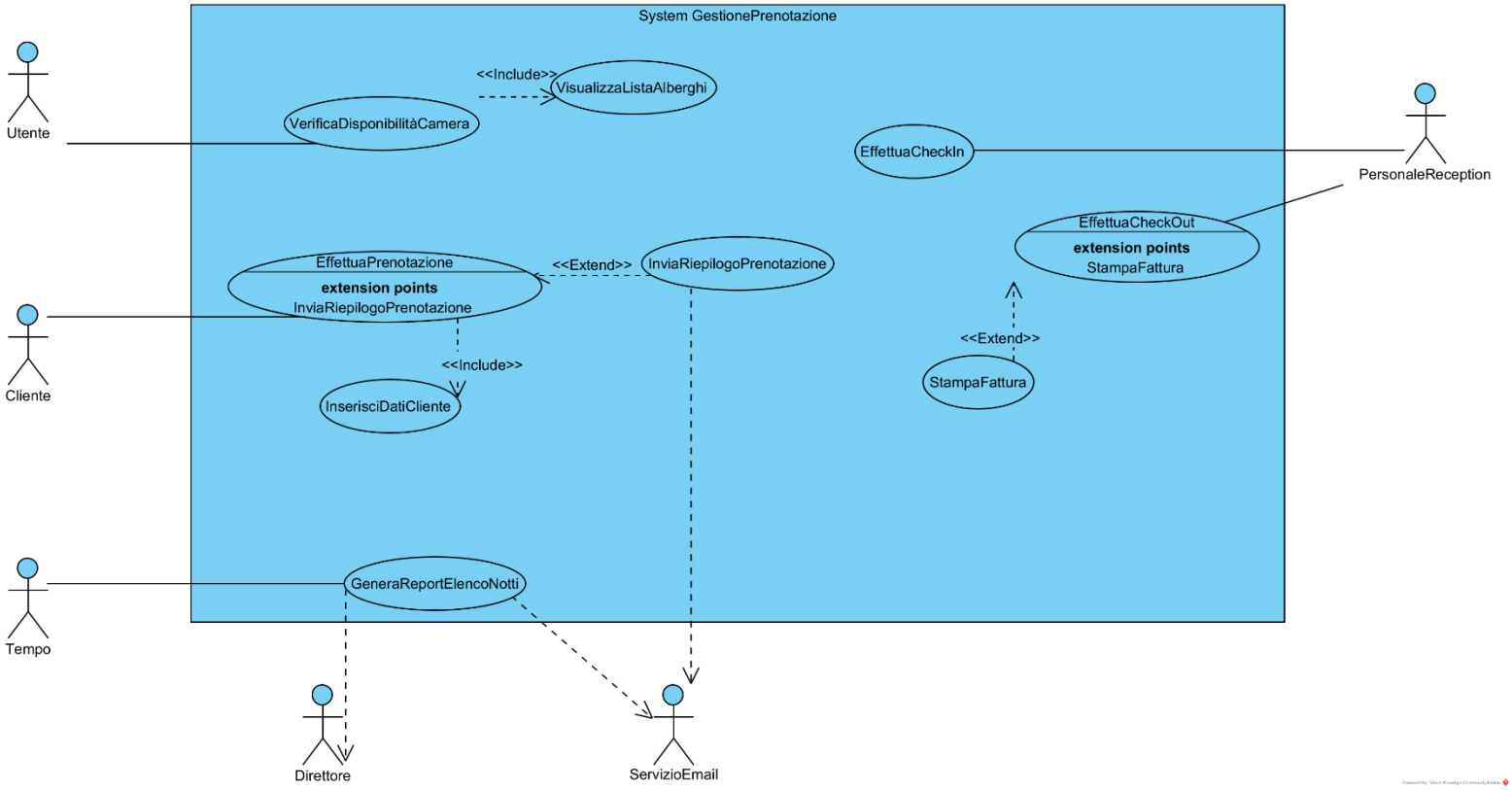
- UC6 : VisualizzaListaAlberghi (UC1)
- UC7 : InserisciDatiCliente (UC2)

CASI D'USO D'ESTENSIONE:

- UC8 : InviaRiepilogoPrenotazione (UC2)
- UC9 : StampaFattura (UC4)

Caso d'Uso	Attori Primari	Attori Secondari	Incude/Extend	Requisiti Corrispondenti
UC1 : VerificaDisponibilità Camera	Utente	-	Include UC6	RF01, RF03
UC2 : EffettuaPrenotazione	Cliente	Servizio E-mail	Include UC7, Estende UC8	RF02
UC3 : EffettuaCheckIn	Personale Reception	-	-	RF08
UC4 : EffettuaCheckOut	Personale Reception	-	Estende UC9	RF09
UC5 : GeneraReport ElencoNotti	Tempo	Direttore, Servizio E-mail	-	RF11
UC6 : VisualizzaListaAlberghi	Utente	-	Incluso in UC1	RF04, RF05
UC7 : InserisciDatiCliente	Cliente	Servizio E-mail	Incluso in UC2	RF06
UC8 : InviaRiepilogo Prenotazione	Tempo	Servizio E-mail	Estensione di UC2	RF07
UC9 : StampaFattura	Personale Reception	-	Estensione di UC4	RF10

2.5.2. Diagramma dei casi d'uso



2.5.3. Scenari

Caso d'Uso	UC1: VerificaDisponibilitàCamera
Attore Primario	Utente
Attore Secondario	-
Descrizione	L'utente cerca e visualizza le camere disponibili per il periodo selezionato e la città desiderata.
Pre-Condizioni	Il sistema deve risultare accessibile.
Sequenza di eventi principale	1. L'utente accede al sistema. 2. L'utente inserisce città, data di arrivo, data di partenza, tipologia camera per cercare la camera desiderata. 2.1. Se la città, la data di arrivo, la data di partenza o la tipologia camera non sono validi, il sistema riporta un messaggio di ERRORE. 3. Include (VisualizzaListaAlberghi). 4. Il sistema mostra all'utente la lista di alberghi disponibili.
Post-Condizioni	L'utente visualizza la lista di alberghi con camere disponibili.
Casi d'uso correlati	UC6 (VisualizzaListaAlberghi)
Sequenza di eventi alternativi	2.1

Caso d'Uso	UC2: EffettuaPrenotazione
Attore Primario	Cliente
Attore Secondario	Servizio E-mail
Descrizione	Il cliente prenota nell'albergo ad esso gradito inserendo ulteriori dati personali.
Pre-Condizioni	Il sistema possiede una lista di alberghi disponibili.
Sequenza di eventi principale	1. Il Cliente seleziona l'albergo e procede alla prenotazione. 2. Include (InserisciDatiCliente) 3. Il sistema cerca se il cliente è già presente nella lista di clienti registrati. 3.1. Se non è presente, crea un nuovo cliente. 3.2. Se il nome, cognome, e-mail, numero di telefono, indirizzo o numero di carta di credito non sono validi, il sistema riporta un messaggio di ERRORE. 4. Il sistema conferma l'avvenuta prenotazione del cliente. 5. Punto di estensione : InviaRiepilogoPrenotazione.
Post-Condizioni	Il cliente ha prenotato la camera d'albergo.
Casi d'uso correlati	UC7 (InserisciDatiCliente), UC8 (InviaRiepilogoPrenotazione)
Sequenza di eventi alternativi	3.1, 3.2

Caso d'Uso	UC3: EffettuaCheckIn
Attore Primario	Personale Reception
Attore Secondario	-
Descrizione	Il cliente viene accolto dal personale della reception per effettuare il Check-in.
Pre-Condizioni	Lo stato della camera deve essere necessariamente "PRENOTATA".
Sequenza di eventi principale	1. Il personale della reception accede al sistema e richiede nome, cognome ed e-mail al Cliente. 1.1. Se nome, cognome ed e-mail non sono validi, il sistema riporta un messaggio di ERRORE. 2. Il sistema controlla che il cliente abbia effettuato una prenotazione. 2.1. Se il cliente non esiste, il sistema genera un ERRORE. 3. Il sistema controlla se la data odierna corrisponde alla data di arrivo indicata sulla prenotazione. 3.1. Se le date non corrispondono, il sistema genera un messaggio di ERRORE. 4. Il personale della reception controlla il numero della camera assegnata. 5. Il sistema aggiorna lo stato della camera a "OCCUPATA".
Post-Condizioni	Il cliente ha effettuato il check-in.
Casi d'uso correlati	-
Sequenza di eventi alternativi	1.1, 2.1, 3.1

Caso d'Uso	UC4: EffettuaCheckOut
Attore Primario	Personale Reception
Attore Secondario	-
Descrizione	Il cliente viene accolto dal personale della reception per effettuare il Check-out.
Pre-Condizioni	Lo stato della camera deve essere necessariamente "OCCUPATA".
Sequenza di eventi principale	1. Il personale della reception accede al sistema e richiede nome, cognome ed e-mail al Cliente. 1.1. Se nome, cognome ed e-mail non sono validi, il sistema riporta un messaggio di ERRORE. 2. Il sistema controlla che il cliente abbia effettuato una prenotazione. 2.1. Se il cliente non esiste, il sistema genera un ERRORE. 3. Il sistema controlla se la data odierna corrisponde alla data di partenza indicata sulla prenotazione. 3.1. Se le date non corrispondono, il sistema genera un messaggio di ERRORE. 4. Il personale della reception controlla il numero della camera assegnata e il sistema aggiorna lo stato della camera a "DISPONIBILE". 5. Punto di estensione: StampaFattura.
Post-Condizioni	Il cliente ha effettuato il check-out.
Casi d'uso correlati	UC9 (StampaFattura)
Sequenza di eventi alternativi	1.1, 2.1, 3.1

Caso d'Uso	UC5: GeneraReportElencoNotti
Attore Primario	Tempo
Attori Secondari	Direttore, Servizio E-mail
Descrizione	Il primo giorno di ogni mese il sistema invia per e-mail al Direttore della catena alberghiera un report con l'elenco delle notti in cui ciascuna camera è stata occupata nel mese precedente
Pre-Condizioni	È il primo giorno di un mese.
Sequenza di eventi principale	1. Il sistema richiede al Direttore di immettere l'indirizzo e-mail a cui inviare il report. 1.1. Se l'e-mail non è valida, il sistema riporta un messaggio di ERRORE. 2. Il sistema verifica che l'e-mail inserita corrisponda ad un Direttore. 2.1. Se l'e-mail non corrisponde all'e-mail di un Direttore, il sistema riporta un messaggio di ERRORE. 3. Il sistema elabora le informazioni e crea l'e-mail contenente il report: se non risultano prenotazioni di camere d'albergo nel mese precedente a quello attuale, il sistema genera un messaggio che esplicita l'assenza di prenotazioni. 4. Il sistema invia l'e-mail con il report/messaggio d'errore allegato all'indirizzo del Direttore.
Post-Condizioni	Viene generato un report che elenca le notti prenotate per ogni camera dell'albergo e il report viene inviato per e-mail al Direttore
Casi d'uso correlati	-
Sequenza di eventi alternativi	1.1, 2.1

Caso d'Uso	UC6: VisualizzaListaAlberghi
Attore Primario	Utente
Attore Secondario	-
Descrizione	Il sistema ricerca una lista di alberghi che rispetti le richieste dell'utente.
Pre-Condizioni	L'utente inserisce dati validi (città, data di arrivo, data di partenza, tipologia camera).
Sequenza di eventi principale	1. Il sistema cerca tutti gli alberghi, di tutte le catene, che si trovano nella città indicata dall'utente e con camere disponibili per il periodo indicato. 1.1. Se la lista è vuota, il sistema restituisce un messaggio d'ERRORE.
Post-Condizioni	-
Casi d'uso correlati	-
Sequenza di eventi alternativi	1.1

Caso d'Uso	UC7: InserisciDatiCliente
Attore Primario	Cliente
Attore Secondario	-
Descrizione	Il Cliente inserisce i suoi dati personali.
Pre-Condizioni	Il cliente ha deciso di prenotare una camera d'albergo.
Sequenza di eventi principale	1. Il cliente fornisce nome, cognome, e-mail, numero di telefono, indirizzo, numero di carta di credito a garanzia. 1.1. Il cliente dovrà inserire i dati fintantoché questi non siano considerati validi dal sistema.
Post-Condizioni	-
Casi d'uso correlati	-
Sequenza di eventi alternativi	1.1

Caso d'Uso	UC8: InviaRiepilogoPrenotazione
Attore Primario	Tempo
Attore Secondario	Servizio E-mail
Descrizione	Il sistema invia un'e-mail con il riepilogo dei dati della prenotazione effettuata dal cliente.
Pre-Condizioni	Il sistema ha confermato la prenotazione.
Sequenza di eventi principale	1. Il sistema invia all'e-mail del cliente un riepilogo dei dati della prenotazione, incluso il prezzo complessivo del pernottamento.
Post-Condizioni	Il cliente ha ricevuto l'e-mail di riepilogo.
Casi d'uso correlati	-
Sequenza di eventi alternativi	-

Caso d'Uso	UC9: StampaFattura
Attore Primario	Personale Reception
Attore Secondario	-
Descrizione	Il personale della reception stampa la fattura relativa al pernottamento.
Pre-Condizioni	Check-out effettuato.
Sequenza di eventi principale	1. Il sistema, al termine del check-out, genera una fattura in cui è presente il prezzo complessivo da saldare.
Post-Condizioni	Il personale della reception ha stampato la fattura.
Casi d'uso correlati	-
Sequenza di eventi alternativi	-

2.6. Diagramma delle classi

Diagramma delle classi:

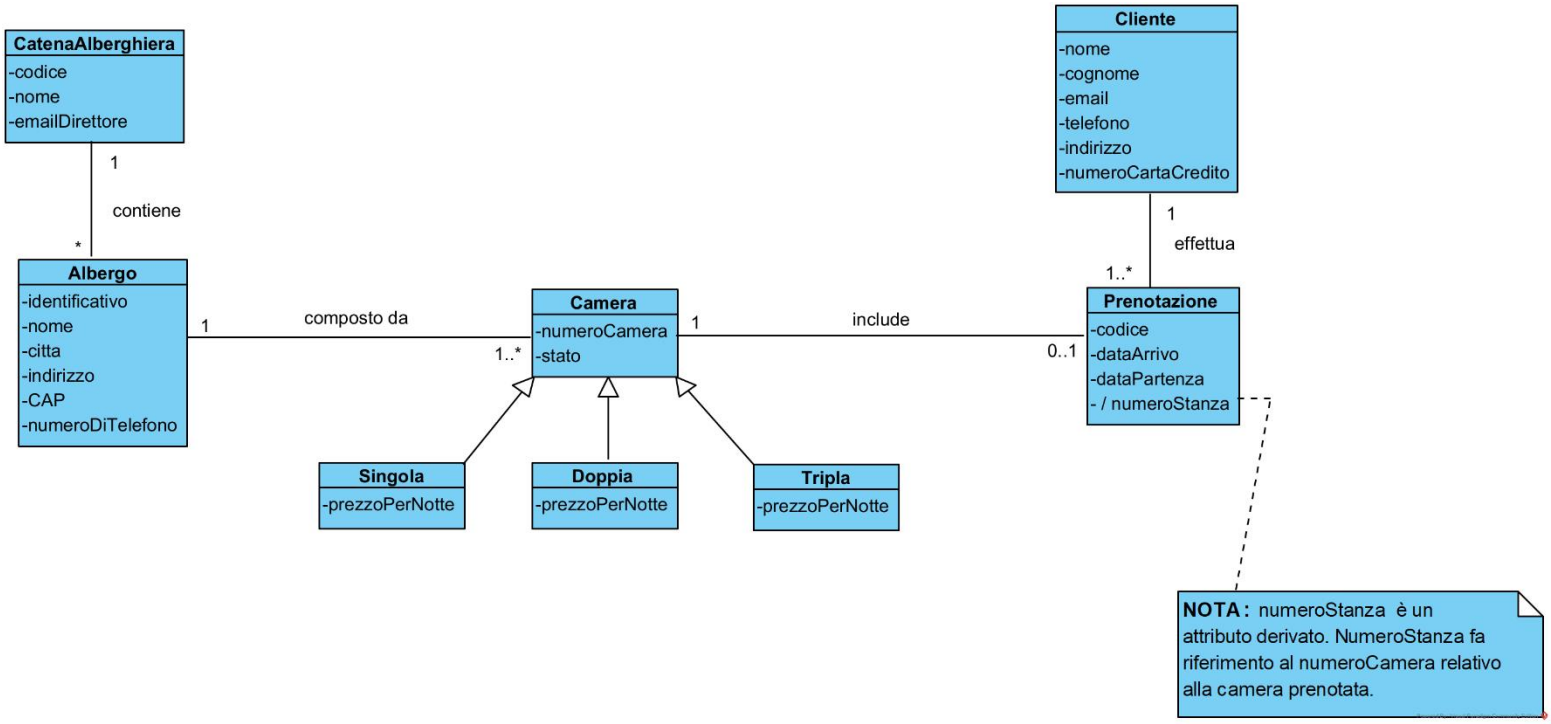
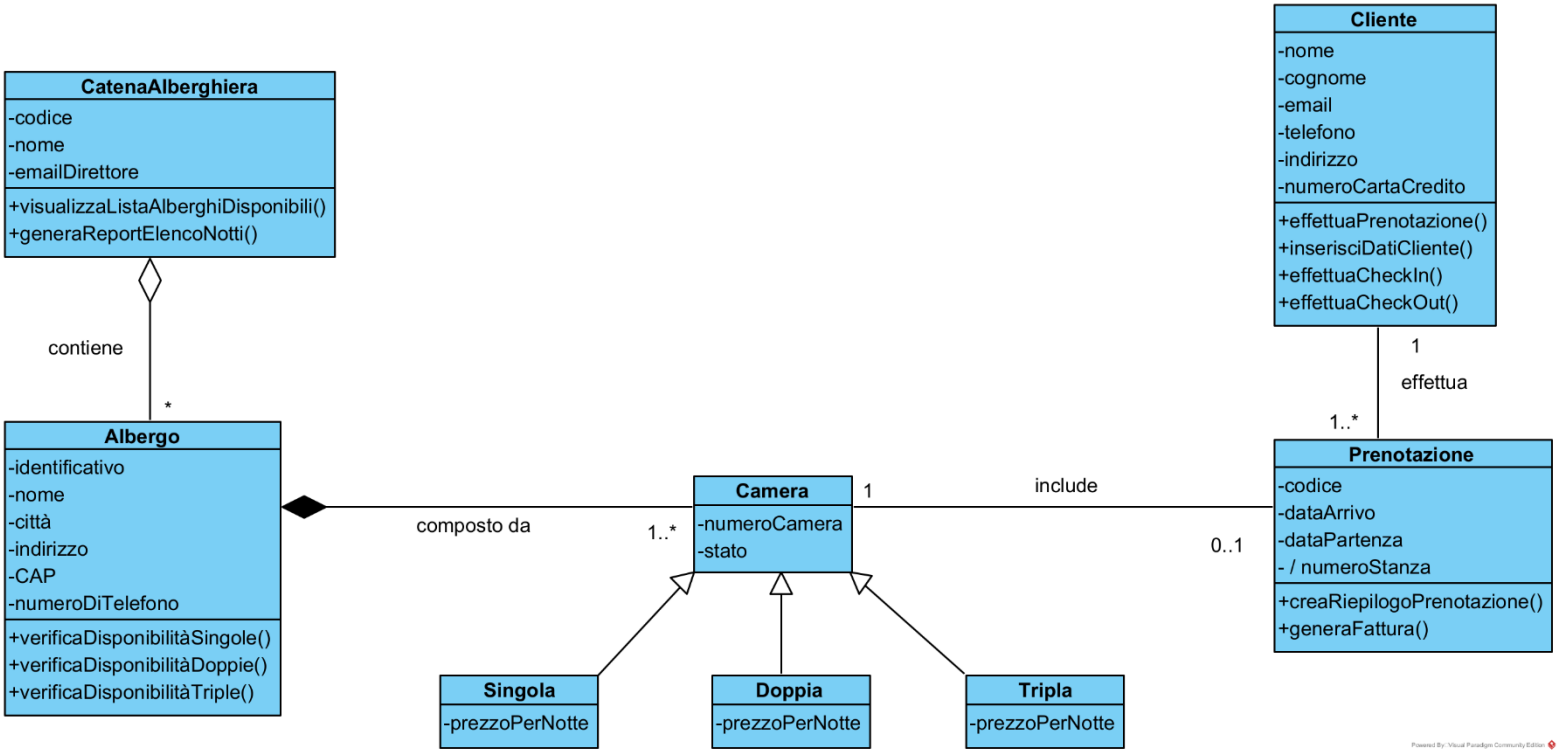


Diagramma delle classi raffinato:



2.7. Diagrammi di sequenza

Diagramma di sequenza di analisi per il caso d'uso **UC1 : VerificaDisponibilitàCamera**

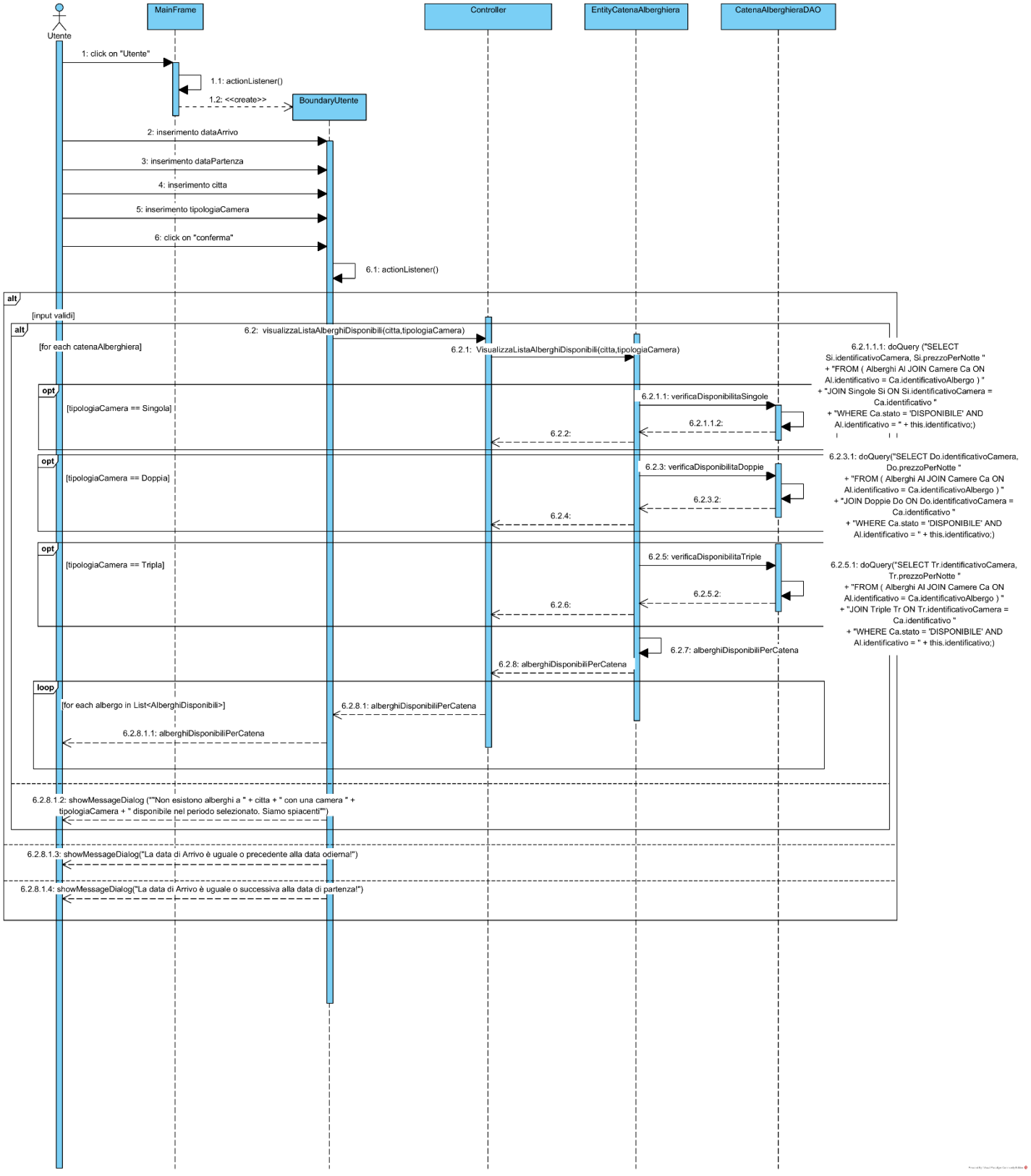


Diagramma di sequenza di analisi per il caso d'uso **UC2 : EffettuaPrenotazione**

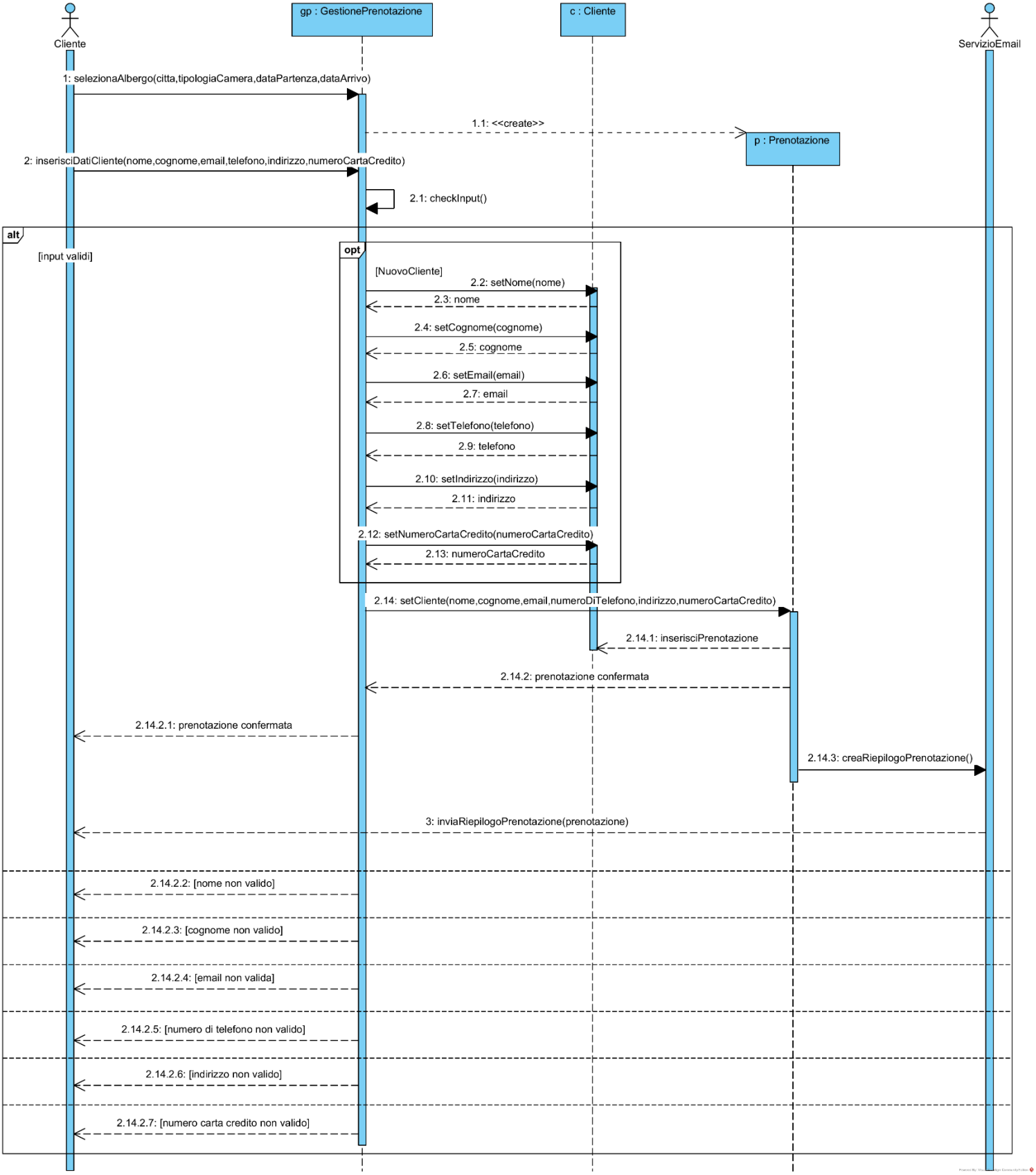


Diagramma di sequenza di analisi per il caso d'uso **UC3 : EffettuaCheckIn**

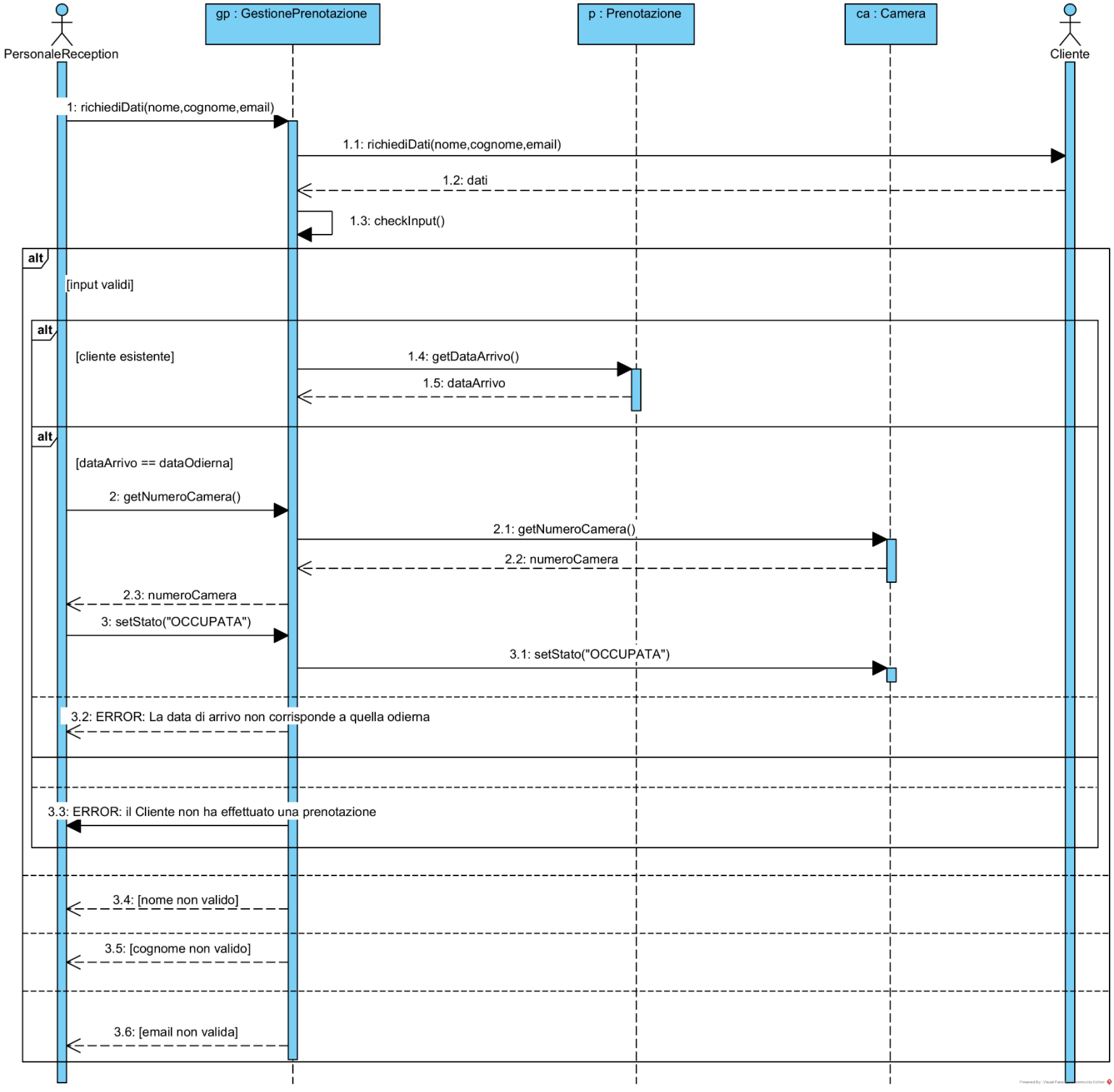


Diagramma di sequenza di analisi per il caso d'uso **UC4 : EffettuaCheckOut**

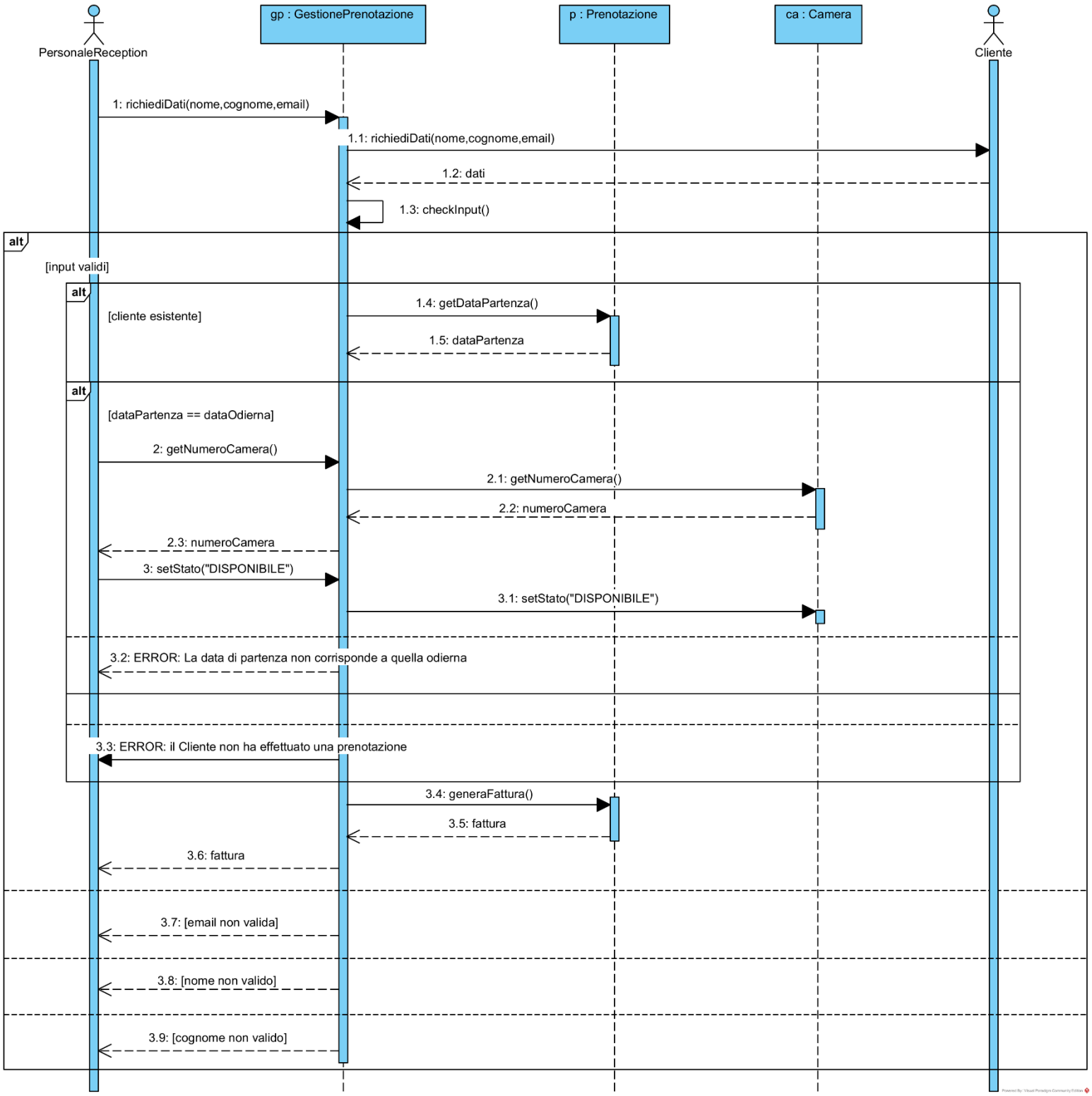
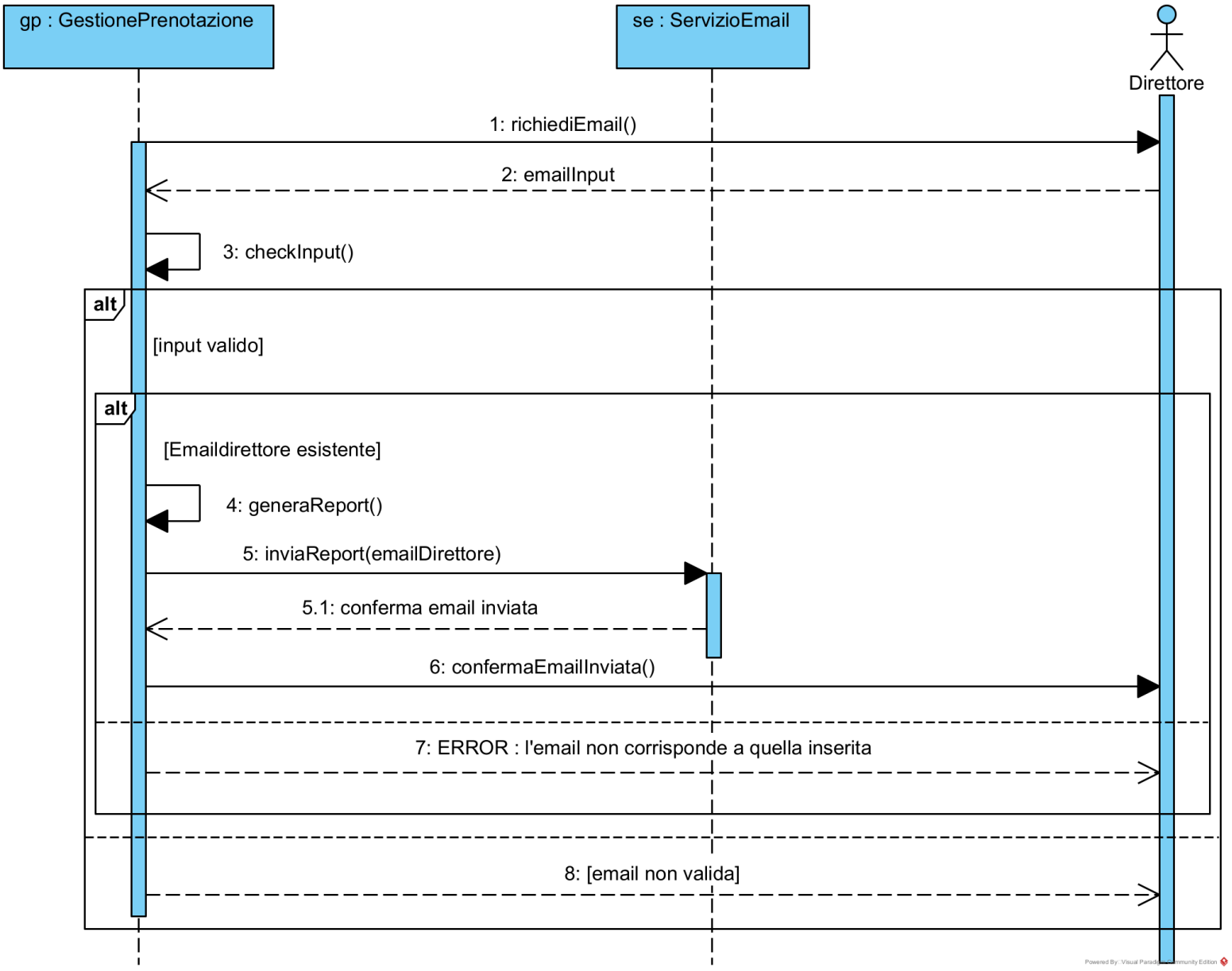


Diagramma di sequenza di analisi per il caso d'uso **UC5 : GeneraReportElencoNotti**



2.8. Verifica della completezza dei requisiti

Legenda: UCD = Use Case Diagram, CD = Class Diagram, SD = Sequence Diagram

- **RF01** è modellato nell' UCD con l'attore "Utente" e con il caso d'uso UC1.
- **RF02** è modellato nell' UCD con gli attori "Cliente" (primario) e "ServizioEmail" (secondario) e con il caso d'uso UC2.
- **RF03** è modellato nel SD "VerificaDisponibilitaCamera" con i parametri della funzione "visualizzaDisponibilitaCamera".
- **RF04** è modellato nel SD "EffettuaPrenotazione" non appena si verifica la condizione "input validi".
- **RF05** è modellato nel UCD con l'attore "Utente" e con il caso d'uso UC6.
- **RF06** è modellato nel UCD con l'attore "Cliente" e con il caso d'uso UC7.
- **RF07** è modellato nel UCD con gli attori "Tempo" (primario) e "ServizioEmail" (secondario) e con il caso d'uso UC8.
- **RF08** è modellato nel UCD con l'attore "PersonaleReception" e con il caso d'uso UC3.
- **RF09** è modellato nel UCD con l'attore "PersonaleReception" e con il caso d'uso UC4.
- **RF10** è modellato nel UCD con l'attore "PersonaleReception" e con il caso d'uso UC9.
- **RF11** è modellato nel UCD con gli attori "Tempo" (primario) e "Direttore" e "ServizioEmail" (secondari) e con il caso d'uso UC5.
- **RD01** è modellato nel CD con la classe "CatenaAlberghiera".
- **RD02** è modellato nel CD con la relazione di contenimento (aggregazione) tra la classe "CatenaAlberghiera" (contenitore) e la classe "Albergo" (contenuto).
- **RD03** è modellato nel CD con la classe "Albergo".
- **RD04** è modellato nel CD con la relazione di contenimento (composizione) tra la classe "Albergo" (contenitore) e la classe "Camera" (contenuto).
- **RD05** è modellato nel CD con la relazione di generalizzazione-specializzazione tra la superclasse "Camera" e le sottoclassi "Singola", "Doppia" e "Tripla".
- **RD06**, **RD08** sono modellate nel CD con la classe "Camera".
- **RD07** è modellato nel CD con le sottoclassi "Singola", "Doppia", "Tripla".

3. STIMA DEI COSTI

Tabella di riferimento per le complessità di dati e transazioni:

	SEMPLICE	MEDIO	COMPLESSO
NILF	3	4	6
NEIF	4	5	7
NEI	3	4	6
NEO	7	10	15
NEQ	5	7	10

Tabella elenco dei fattori correttivi (il cui valore è compreso tra 0 e 5):

FATTORI CORRETTIVI
COMUNICAZIONE DATI
DISTRIBUZIONE ELABORAZIONE
PRESTAZIONI
UTILIZZO INTENSIVO CONFIGURAZIONE
FREQUENZA DELLE TRANSAZIONI
INSERIMENTO DATI INTERATTIVO
EFFICIENZA PER L'UTENTE FINALE
AGGIORNAMENTO INTERATTIVO
COMPLESSITÀ ELABORATIVA
RIUSABILITÀ
FACILITÀ INSTALLAZIONE
FACILITÀ GESTIONE OPERATIVA
MOLTEPLICITÀ DI SITI
FACILITÀ DI MODIFICA

VERIFICA DISPONIBILITÀ CAMERA

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOTALE
NILF	2		4	6	10
NEIF	0				
NEI	4	3			12
NEO	0				
NEQ	1	5			5

- **NILF:** Ricerca delle catene alberghiere che si trovano nella città indicata dall'utente [1 medio], ricerca degli alberghi per ogni catena alberghiera [1 complesso]
- **NEIF:** -
- **NEI:** Città, Data Partenza, Data Arrivo, Tipologia Camera [4 semplici]
- **NEO:** -
- **NEQ:** Controllo della tipologia di camera desiderata dall'utente. [1 semplice]

UFP	27
LLOC	1431

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	5
DISTRIBUZIONE ELABORAZIONE	2
PRESTAZIONI	4
UTILIZZO INTENSIVO CONFIGURAZIONE	1
FREQUENZA DELLE TRANSAZIONI	5
INSERIMENTO DATI INTERATTIVO	3
EFFICIENZA PER L'UTENTE FINALE	4
AGGIORNAMENTO INTERATTIVO	0
COMPLESSITÀ ELABORATIVA	2
RIUSABILITÀ	2
FACILITÀ INSTALLAZIONE	2

FP	27,27
JAVA	1445

FACILITÀ GESTIONE OPERATIVA	4
MOLTEPLICITÀ DI SITI	0
FACILITÀ DI MODIFICA	2
Totale	**36**

- **COMUNICAZIONE DATI:** Incidenza essenziale. È fondamentale che l'utente visualizzi la lista di alberghi disponibili
- **DISTRIBUZIONE ELABORAZIONE:** Incidenza moderata. L'utente inserisce dei dati che andranno validati e conteranno nell'elaborazione finale; tuttavia, l'azione principale è di pura visualizzazione dati.
- **PRESTAZIONI:** Incidenza significativa. È importante che l'utente non attenda eccessivamente per la visualizzazione degli alberghi.
- **UTILIZZO INTENSIVO CONFIGURAZIONE:** Incidenza scarsa. La funzione presa in analisi non richiede una configurazione intensiva.
- **FREQUENZA DELLE TRANSAZIONI:** Incidenza essenziale. Tutte le elaborazioni effettuate (ricerca di un albergo nella città richiesta, interrogazione sulla tabella della camera selezionata dall'utente) si basano sugli input dell'utente.
- **INSERIMENTO DATI INTERATTIVO:** Incidenza media. L'utente inserisce dati ogni volta che deve ricercare una camera; tuttavia, ciò viene fatto una sola volta in tutta la funzione.
- **EFFICIENZA PER L'UTENTE FINALE:** Incidenza significativa. È importante che l'utente ottenga risultati precisi per la visualizzazione degli alberghi, e che possa farlo attraverso una grafica intuitiva.
- **AGGIORNAMENTO INTERATTIVO:** Incidenza ininfluente. Non è richiesta una frequenza di aggiornamento alta.
- **COMPLESSITÀ ELABORATIVA:** Incidenza moderata. Per l'elaborazione dei dati è necessario accedere a due entità separate.
- **RIUSABILITÀ:** Incidenza moderata. La funzione viene utilizzata sia singolarmente che nel caso in cui si effettui una prenotazione.
- **FACILITÀ INSTALLAZIONE:** Incidenza moderata. Per il corretto funzionamento, è necessario solamente avere la lista di catene alberghiere e di alberghi.
- **FACILITÀ GESTIONE OPERATIVA:** Incidenza significativa. È importante che l'utente possa usare con facilità i dati ottenuti (la lista di alberghi disponibili).
- **MOLTEPLICITÀ DI SITI:** Incidenza ininfluente. Il sistema è centralizzato, ovvero i dati utili all'elaborazione della lista di alberghi non dipendono da enti esterni.
- **FACILITÀ DI MODIFICA:** Incidenza moderata. Sono previste solo due macro-elaborazioni (l'utente inserisce i dati e poi visualizza la lista di alberghi); se l'utente si

accorge di aver commesso un errore, non è necessario implementare funzioni particolari per il rollback, basta che l'utente riavvii il processo e reinserisca gli input.

EFFETTUA PRENOTAZIONE

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOTALE
NILF	2	3	4		7
NEIF	1	4			4
NEI	5	3			15
NEO	0				
NEQ	2	5			10

- **NILF:** Creazione di un nuovo cliente se non esiste [1 semplice]. Il sistema invia il riepilogo della prenotazione [1 medio]
- **NEIF:** Il cliente seleziona l'albergo tra quelli presenti [1 semplice]
- **NEI:** Nome, Cognome, Email, Numero di Telefono, Numero Carta di Credito [5 semplici]
- **NEO:** -
- **NEQ:** Controllo della tabella clienti per verificare l'esistenza del cliente [1 semplice], conferma dell'avvenuta prenotazione [1 semplice]

UFP	36
LLOC	1908

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	5
DISTRIBUZIONE ELABORAZIONE	4
PRESTAZIONI	5
UTILIZZO INTENSIVO CONFIGURAZIONE	1
FREQUENZA DELLE TRANSAZIONI	5
INSERIMENTO DATI INTERATTIVO	4
EFFICIENZA PER L'UTENTE FINALE	4
AGGIORNAMENTO INTERATTIVO	0

FP	37,08
JAVA	1965

COMPLESSITÀ ELABORATIVA	2
RIUSABILITÀ	0
FACILITÀ INSTALLAZIONE	1
FACILITÀ GESTIONE OPERATIVA	5
MOLTEPLICITÀ DI SITI	0
FACILITÀ DI MODIFICA	2
Totale	**38**

- **COMUNICAZIONE DATI:** Incidenza essenziale. E' fondamentale che il cliente visualizzi l'avvenuta prenotazione della camera.
- **DISTRIBUZIONE ELABORAZIONE:** Incidenza significativa. Il cliente si occuperà dell'inserimento dei dati, utili nelle fasi successive di elaborazione. Se il cliente fornisce dei dati sbagliati non si potrà proseguire nella prenotazione.
- **PRESTAZIONI:** Incidenza essenziale. È importante che il cliente non attenda eccessivamente per l'invio della prenotazione.
- **UTILIZZO INTENSIVO CONFIGURAZIONE:** Incidenza scarsa. La funzione presa in analisi non richiede una configurazione intensiva.
- **FREQUENZA DELLE TRANSAZIONI:** Incidenza essenziale. Tutte le elaborazioni effettuate (scelta dell'albergo, interrogazione sulla tabella del cliente, invio riepilogo) si basano sugli input dell'utente.
- **INSERIMENTO DATI INTERATTIVO:** Incidenza significativa. L'utente inserisce tutti i dati richiesti ogni volta che deve prenotarsi.
- **EFFICIENZA PER L'UTENTE FINALE:** Incidenza significativa. È importante che l'utente ottenga risultati precisi per la validazione della prenotazione, e che possa farlo attraverso una grafica intuitiva.
- **AGGIORNAMENTO INTERATTIVO:** Incidenza inifluente. Non è richiesta una frequenza di aggiornamento alta.
- **COMPLESSITÀ ELABORATIVA:** Incidenza moderata. Per l'elaborazione dei dati è necessario accedere a due entità separate.
- **RIUSABILITÀ:** Incidenza inifluente. La funzione non viene riutilizzata successivamente.
- **FACILITÀ INSTALLAZIONE:** Incidenza scarsa. Per il corretto funzionamento, è necessario solamente avere la lista di camere.
- **FACILITÀ GESTIONE OPERATIVA:** Incidenza significativa. È importante che l'utente possa facilmente effettuare una prenotazione.
- **MOLTEPLICITÀ DI SITI:** Incidenza ininfluente. Il sistema è centralizzato, ovvero i dati utili all'elaborazione non dipendono da enti esterni.

- **FACILITÀ DI MODIFICA:** Incidenza moderata. Se l'utente sbaglia nell'inserimento dei dati, è necessario che riavvii il processo e reinserisca gli input.

EFFETTUA CHECK-IN

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOTALE
NILF	1	3			3
NEIF	1	4			4
NEI	3	3			9
NEO	0				
NEQ	2		7		14

- **NILF:** Aggiornamento dello stato della camera. [1 semplice]
- **NEIF:** Controllo del numero della camera da parte del personale [1 semplice]
- **NEI:** Nome, Cognome, E-mail [3 semplici]
- **NEO:** -
- **NEQ:** Controllo del cliente attraverso la prenotazione. Controllo della data di Arrivo coincidente con quella odierna [2 medio].

UFP	30
LLOC	1590

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	5
DISTRIBUZIONE ELABORAZIONE	4
PRESTAZIONI	4
UTILIZZO INTENSIVO CONFIGURAZIONE	1
FREQUENZA DELLE TRANSAZIONI	5
INSERIMENTO DATI INTERATTIVO	2
EFFICIENZA PER L'UTENTE FINALE	2
AGGIORNAMENTO INTERATTIVO	0
COMPLESSITÀ ELABORATIVA	2

FP	30
JAVA	1590

RIUSABILITÀ	1
FACILITÀ INSTALLAZIONE	2
FACILITÀ GESTIONE OPERATIVA	5
MOLTEPLICITÀ DI SITI	0
FACILITÀ DI MODIFICA	2
Totale	**35**

- **COMUNICAZIONE DATI:** Incidenza essenziale. È fondamentale, attraverso il Personale della Reception, che il cliente sia a conoscenza dell'avvenuto check-in.
- **DISTRIBUZIONE ELABORAZIONE:** Incidenza significativa. Il cliente inserisce i dati richiesti dal Personale della Reception, utili nelle fasi successive di elaborazione. Se il cliente fornisce dei dati sbagliati non si potrà proseguire il check-in.
- **PRESTAZIONI:** Incidenza significativa. È importante che il cliente non attenda eccessivamente per effettuare il check-in.
- **UTILIZZO INTENSIVO CONFIGURAZIONE:** Incidenza scarsa. La funzione presa in analisi non richiede una configurazione intensiva.
- **FREQUENZA DELLE TRANSAZIONI:** Incidenza essenziale. Tutte le elaborazioni effettuate (ricerca prenotazione per data, interrogazione tabella camera e update) si basano sugli input dell'utente.
- **INSERIMENTO DATI INTERATTIVO:** Incidenza moderata. L'utente inserisce, attraverso il Personale della Reception, i dati ogni volta che deve effettuare il check-in; tuttavia, ciò viene fatto una sola volta in tutta la funzione.
- **EFFICIENZA PER L'UTENTE FINALE:** Incidenza moderata. È importante che l'utente possa effettuare il check-in, ma non è rilevante il fatto che la grafica sia intuitiva poiché l'interfacciamento con il sistema si ha attraverso il Personale della Reception.
- **AGGIORNAMENTO INTERATTIVO:** Incidenza ininfluente. Non è richiesta una frequenza di aggiornamento alta.
- **COMPLESSITÀ ELABORATIVA:** Incidenza moderata. Per l'elaborazione dei dati è necessario accedere a due entità separate.
- **RIUSABILITÀ:** Incidenza scarsa. La funzione non viene riutilizzata successivamente.
- **FACILITÀ INSTALLAZIONE:** Incidenza moderata. Per il corretto funzionamento, è necessario avere una lista di prenotazioni e una lista di camere.
- **FACILITÀ GESTIONE OPERATIVA:** Incidenza significativa. È importante che l'utente, attraverso il Personale della Reception, effettui il check-in, entrando in possesso della camera.
- **MOLTEPLICITÀ DI SITI:** Incidenza ininfluente. Il sistema è centralizzato, ovvero i dati utili all'elaborazione non dipendono da enti esterni.

- **FACILITÀ DI MODIFICA:** Incidenza moderata. Se l'utente sbaglia nell'inserimento dei dati, è necessario che riavvii il processo e reinserisca gli input.

EFFETTUA CHECK-OUT

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOTALE
NILF	1	3			3
NEIF	1	4			4
NEI	3	3			9
NEO	1		10		10
NEQ	2		7		14

- **NILF:** Aggiornamento dello stato della camera. [1 semplice]
- **NEIF:** Controllo del numero della camera da parte del personale [1 semplice]
- **NEI:** Nome, Cognome, E-mail [3 semplici]
- **NEO:** Stampa della fattura [1 medio]
- **NEQ:** Controllo del cliente attraverso la prenotazione. Controllo della data di Partenza coincidente con quella odierna [2 medio].

UFP	40
LLOC	2120

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	5
DISTRIBUZIONE ELABORAZIONE	4
PRESTAZIONI	4
UTILIZZO INTENSIVO CONFIGURAZIONE	1
FREQUENZA DELLE TRANSAZIONI	5
INSERIMENTO DATI INTERATTIVO	2
EFFICIENZA PER L'UTENTE FINALE	2
AGGIORNAMENTO INTERATTIVO	0
COMPLESSITÀ ELABORATIVA	2

FP	40
JAVA	2120

RIUSABILITÀ	1
FACILITÀ INSTALLAZIONE	2
FACILITÀ GESTIONE OPERATIVA	5
MOLTEPLICITÀ DI SITI	0
FACILITÀ DI MODIFICA	2
Totale	**35**

- **COMUNICAZIONE DATI:** Incidenza essenziale. È fondamentale, attraverso il Personale della Reception, che il cliente sia a conoscenza dell'avvenuto check-out.
- **DISTRIBUZIONE ELABORAZIONE:** Incidenza significativa. Il cliente inserisce i dati richiesti dal Personale della Reception, utili nelle fasi successive di elaborazione. Se il cliente fornisce dei dati sbagliati non si potrà proseguire il check-out.
- **PRESTAZIONI:** Incidenza significativa. È importante che il cliente non attenda eccessivamente per effettuare il check-out.
- **UTILIZZO INTENSIVO CONFIGURAZIONE:** Incidenza scarsa. La funzione presa in analisi non richiede una configurazione intensiva.
- **FREQUENZA DELLE TRANSAZIONI:** Incidenza essenziale. Tutte le elaborazioni effettuate (ricerca prenotazione per data, interrogazione tabella camera e update, stampa della fattura) si basano sugli input dell'utente.
- **INSERIMENTO DATI INTERATTIVO:** Incidenza moderata. L'utente inserisce, attraverso il Personale della Reception, i dati ogni volta che deve effettuare il check-on; tuttavia, ciò viene fatto una sola volta in tutta la funzione.
- **EFFICIENZA PER L'UTENTE FINALE:** Incidenza moderata. È importante che l'utente possa effettuare il check-out, ma non è rilevante il fatto che la grafica sia intuitiva poiché l'interfacciamento con il sistema si ha attraverso il Personale della Reception.
- **AGGIORNAMENTO INTERATTIVO:** Incidenza ininfluente. Non è richiesta una frequenza di aggiornamento alta.
- **COMPLESSITÀ ELABORATIVA:** Incidenza moderata. Per l'elaborazione dei dati è necessario accedere a due entità separate.
- **RIUSABILITÀ:** Incidenza scarsa. La funzione non viene riutilizzata successivamente.
- **FACILITÀ INSTALLAZIONE:** Incidenza moderata. Per il corretto funzionamento, è necessario avere una lista di prenotazioni e una lista di camere.
- **FACILITÀ GESTIONE OPERATIVA:** Incidenza significativa. È importante che l'utente, attraverso il Personale della Reception, effettui il check-out, lasciando la camera.
- **MOLTEPLICITÀ DI SITI:** Incidenza ininfluente. Il sistema è centralizzato, ovvero i dati utili all'elaborazione non dipendono da enti esterni.

- **FACILITÀ DI MODIFICA:** Incidenza moderata. Se l'utente sbaglia nell'inserimento dei dati, è necessario che riavvii il processo e reinserisca gli input.

GENERA REPORT ELENCO NOTTI

	VALORE	SEMPLICE	MEDIO	COMPLESSO	TOTALE
NILF	2		4	6	10
NEIF	0				
NEI	1	3			3
NEO	0				
NEQ	1	7			7

- **NILF:** Elaborazione dei dati e creazione della mail contente il report [1 complesso]. Il sistema invia una mail [1 medio]
- **NEIF:** -
- **NEI:** E-mail [1 semplice]
- **NEO:** -
- **NEQ:** Controllo dell'e-mail del direttore . [1 medio]

UFP	20
LLOC	1060

FATTORI CORRETTIVI	
COMUNICAZIONE DATI	5
DISTRIBUZIONE ELABORAZIONE	4
PRESTAZIONI	4
UTILIZZO INTENSIVO CONFIGURAZIONE	1
FREQUENZA DELLE TRANSAZIONI	5
INSERIMENTO DATI INTERATTIVO	3
EFFICIENZA PER L'UTENTE FINALE	4
AGGIORNAMENTO INTERATTIVO	0
COMPLESSITÀ ELABORATIVA	2

FP	20,2
JAVA	1071

RIUSABILITÀ	1
FACILITÀ INSTALLAZIONE	1
FACILITÀ GESTIONE OPERATIVA	4
MOLTEPLICITÀ DI SITI	0
FACILITÀ DI MODIFICA	2
Totale	**36**

- **COMUNICAZIONE DATI:** Incidenza essenziale. È fondamentale che il Direttore sia a conoscenza riceva il report.
- **DISTRIBUZIONE ELABORAZIONE:** Incidenza significativa. Il Direttore inserisce i dati richiesti, utili nelle fasi successive di elaborazione. Se i dati sono sbagliati non si potrà inviare il report.
- **PRESTAZIONI:** Incidenza significativa. È importante che il Direttore non attenda eccessivamente l'invio del report.
- **UTILIZZO INTENSIVO CONFIGURAZIONE:** Incidenza scarsa. La funzione presa in analisi non richiede una configurazione intensiva.
- **FREQUENZA DELLE TRANSAZIONI:** Incidenza essenziale. Tutte le elaborazioni effettuate (richiesta dati, interrogazione tabella Direttore e invio report) si basano sugli input dell'utente.
- **INSERIMENTO DATI INTERATTIVO:** Incidenza media. Il Direttore inserisce i dati ogni volta che il sistema lo richiede per l'invio del report; tuttavia, ciò viene fatto una sola volta in tutta la funzione.
- **EFFICIENZA PER L'UTENTE FINALE:** Incidenza significativa. È importante che il Direttore possa riceve il report, e che possa farlo attraverso una grafica intuitiva.
- **AGGIORNAMENTO INTERATTIVO:** Incidenza ininfluente. Non è richiesta una frequenza di aggiornamento alta.
- **COMPLESSITÀ ELABORATIVA:** Incidenza moderata. Per l'elaborazione dei dati è necessario accedere a due entità separate.
- **RIUSABILITÀ:** Incidenza scarsa. La funzione non viene riutilizzata successivamente.
- **FACILITÀ INSTALLAZIONE:** Incidenza scarsa. Per il corretto funzionamento, è necessario avere una lista di prenotazioni.
- **FACILITÀ GESTIONE OPERATIVA:** Incidenza significativa. È importante che il Direttore, ricerca il report aggiornato per ogni mese.
- **MOLTEPLICITÀ DI SITI:** Incidenza ininfluente. Il sistema è centralizzato, ovvero i dati utili all'elaborazione non dipendono da enti esterni
- **FACILITÀ DI MODIFICA:** Incidenza moderata. Se il Direttore sbaglia nell'inserimento dei dati, è necessario che riavvii il processo e reinserisca gli input.

4. PIANO DI TEST FUNZIONALE

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ "VerificaDisponibilitaCamera".

Il numero di test da effettuarsi senza particolari vincoli è: **3 * 4 * 2 = 24**

Introduciamo i vincoli [ERROR].

Il numero di test da eseguire per testare singolarmente i vincoli è 6 (2 per Città, 3 per Data Arrivo/Data Partenza, 1 per Tipologia Camera).

Il numero di test risultante è: **(1 * 1 * 1) + 6 = 7**.

CITTÀ	DATA ARRIVO / PARTENZA	TIPOLOGIA CAMERA
➢ Stringa di caratteri di lunghezza <= 50 ➢ Stringa di caratteri di lunghezza > 50 [ERROR] ➢ Stringa che contiene simboli che non sono caratteri [ERROR]	➢ Data con formato valido (dd/mm/yyyy) ➢ Data con formato non valido [ERROR] ➢ Data di partenza precedente alla data di arrivo [ERROR] ➢ Data di arrivo successiva alla data di partenza [ERROR]	➢ Stringa che equivale a "Singola", "Doppia" o "Tripla" ➢ Stringa che non sia "Singola", "Doppia" o "Tripla" [ERROR]

TEST SUITE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE
1	Tutti input validi	Città valida, Data Arrivo/ Partenza valida, Tipologia camera valida	L'utente inserisce città, data di arrivo, data di partenza, tipologia camera per cercare la camera desiderata. Il sistema inserisce l'albergo nella lista di quelli disponibili nella città e periodo indicato e la mostra all'utente	{Città: Napoli, DataArrivo: 19/05/2023, DataPartenza: 24/05/2023, Tipologia Camera: Singola}	Camera disponibile.	Si visualizza la lista di alberghi con camere disponibili di tutte le catene alberghiere.
2	Città stringa > 50 caratteri	Città [ERROR], Data Arrivo/ Partenza valida, Tipologia camera valida	Nessuna	{Città: "napoliromafirenzeudinemilanopalermogenovaveneziapadova", DataArrivo: 11/01/2022, DataPartenza: 19/01/2022, Tipologia Camera: Doppia}	Messaggio di errore - Nome della città troppo lungo!	-
3	Città stringa con simboli	Città stringa con simboli [ERROR], Data Arrivo/ Partenza valida,	Nessuna	{Città: "€%^!?", DataArrivo: 07/02/2018, DataPartenza: 13/02/2018, Tipologia Camera: Tripla}	Messaggio di errore - Nome della	-

		Tipologia camera valida			città non valido!	
4	Data formato non valido	Città valida, Data Arrivo/ Partenza [ERROR], Tipologia camera valida	Nessuna	{Città: “Firenze”, DataArrivo: 1207/302/20185, DataPartenza: 613/02/20186, Tipologia Camera: Singola}	Messaggio di errore - Data non valida!	-
5	Data di partenza precedente a quella di arrivo	Città valida, Data di partenza precedente a quella di arrivo [ERROR], Tipologia camera valida	Nessuna	{Città: “Firenze”, DataArrivo: 27/03/2018, DataPartenza : 15/03/2018, Tipololgia Camera: Doppia}	Messaggio di errore - Data partenza precedente a data arrivo!	-
6	Data di arrivo successiva a quella di arrivo	Città valida, Data di arrivo successiva a quella di partenza [ERROR], Tipologia camera valida	Nessuna	{Città: “Firenze”, DataArrivo: 15/03/2018, DataPartenza : 12/03/2018, Tipololgia Camera: Doppia}	Messaggio di errore - Data arrivo successiva a data partenza!	-
7	Stringa che non sia “Singola”,” Doppia” o “Tripla” [ERROR]	Città valida, Data Arrivo/ Partenza valida, Tipologia Camera [ERROR]	Nessuna	{Città: “Milano”, DataArrivo: 24/04/2021, DataPartenza: 30/04/2021, Tipologia Camera: pippo}	Messaggio di errore - Tipologia camera non valido!	-

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ "EffettuaPrenotazione".

Il numero di test da effettuarsi senza particolari vincoli è: **3 * 3 * 3 * 2 * 4 = 216**

Introduciamo i vincoli [ERROR].

Il numero di test da eseguire per testare singolarmente i vincoli è 10 (2 per Nome/Cognome, 2 per E-mail, 2 per Numero Di Telefono, 1 per Indirizzo, 3 per numero carta di credito).

Il numero di test risultante è: **(1 * 1 * 1 * 1 * 1) + 10 = 11**.

NOME / COGNOME	E-MAIL	NUMERO DI TELEFONO	INDIRIZZO	NUMERO CARTA DI CREDITO
➢ Stringa di caratteri di lunghezza <= 50 ➢ Stringa di caratteri di lunghezza > 50 [ERROR] ➢ Stringa che contiene simboli che non sono caratteri [ERROR]	➢ Stringa in cui è presente il simbolo @ ➢ Stringa in cui non è presente il simbolo @ [ERROR] ➢ Stringa in cui il dominio dell'e-mail non è valido [ERROR]	➢ Stringa di caratteri <=20 ➢ Stringa di caratteri >20 [ERROR] ➢ Stringa di caratteri che contiene simboli/caratteri [ERROR]	➢ Stringa di caratteri <=150 ➢ Stringa di caratteri >150 [ERROR]	➢ Stringa = a 16 caratteri numerici ➢ Numero carta di credito con stringa > di 16 caratteri [ERROR] ➢ Numero carta di credito con stringa < di 16 caratteri [ERROR] ➢ Numero carta di credito contente caratteri o simboli [ERROR]

TEST SUITE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE
1	Tutti input validi	Nome/Cognome valido, E-mail valida, Telefono valido, Indirizzo valido, Numero Carta Credito valido	La camera è disponibile e il sistema inserisce l'albergo nella lista di quelli disponibili nella città e periodo indicato. Il cliente può effettuare la prenotazione inserendo i dati personali.	{Nome: MariaElena, Cognome: Lecci, E-mail: mariaelenatacco@email.it, Telefono:0812334412, Indirizzo: Via Nuova Agnano, NumeroCartaCredito: 123456789101345}	Camera Prenotata	Si ricevono per email i dettagli della prenotazione.
2	Stringa di caratteri > 50	Nome/Cognome [ERROR], E-mail valida, Telefono valido, Indirizzo valido, Numero Carta Credito valido	Nessuna	{Nome: ritamartinariccardoalessandroantonioluciafrancesca, Cognome: castaldiborgstromcampanellabrescialucianadifrancia, E-mail : raffaelemina@email.it, Telefono: 0812334412, Indirizzo: Via Sardegna, NumeroCartaCredito: 2319335126912931}	Messaggio di errore - Nome o Cognome non validi!	-
3	Stringa con numeri /	Nome/Cognome [ERROR], E-mail	Nessuna	{Nome: rita232, Cognome: castaldi?23, E-mail :	Messaggio di errore - Nome o	-

	simboli	valida, Telefono valido, Indirizzo valido, Numero Carta Credito valido		paolostrigaro@gmail.com, Telefono:0812334412, Indirizzo: Via Sassuolo, NumeroCartaCredito : 2319335126912931}	Cognome non validi!	
4	Stringa in cui non è presente il simbolo @	Nome/Cognome valido, E-mail [ERROR], Telefono valido, Indirizzo valido, Numero Carta Credito valido	Nessuna	{Nome: Michele, Cognome: Di Meo, E-mail: micheledimeo_email.it, Telefono:0812334412, Indirizzo: Via Pisa Centro, NumeroCartaCredito : 2319335126911573}	Messaggio di errore - E-mail non valida!	-
5	Stringa in cui il dominio dell'e-mail non è valido	Nome/Cognome valido, E-mail [ERROR], Telefono valido, Indirizzo valido, Numero Carta Credito valido	Nessuna	{Nome: MariaElena, Cognome: Tacco, E-mail: mariaelenatacco@lapadula.mi, Telefono: 0812334412, Indirizzo: Via Aniello Falcone, NumeroCartaCredito: 1234567891013457}	Messaggio di errore - Dominio dell'e-mail non riconosciuto!	-
6	Stringa di caratteri > 20	Nome/Cognome valido, E-mail valida, Telefono [ERROR], Indirizzo valido, Numero Carta Credito valido	Nessuna	{ Nome: Lorenzo, Cognome: Coti, E-mail : lorenzocoti@gmail.com, Telefono: 12345678901234567890, Indirizzo: Via Petrazzuolo, NumeroCartaCredito: 2319335199922900}	Messaggio di errore - Numero di telefono non valido!	-
7	Stringa di caratteri che contiene	Nome/Cognome valido, E-mail valida, Telefono [ERROR], Indirizzo	Nessuna	{Nome: Lucia, Cognome: Gallo, E-mail: luciagallo@gmail.com, Telefono: vittoria, Indirizzo: Via	Messaggio di errore - Numero di telefono non valido!	-

	simboli/caratteri	valido, Numero Carta Credito valido		sicilia, NumeroCartaCredito: 2319335199922900}		
8	Stringa di caratteri > 150	Nome/Cognome valido, E-mail valida, Telefono valido, Indirizzo [ERROR], Numero Carta Credito valido		{Nome: Pasquale, Cognome: Mecca, Email: pasqualemecca@gmail.com, Telefono:0812334412, Indirizzo: 1234 Via delle Magnolie, Appartamento 5, Palazzo Rossi, Piazza Garibaldi, Via Roma, Piazza della Libertà, Via Verdi, Via Mazzini, Via Cavour, Via Carducci, Via Manzoni, Via Dante, Via Leopardi, Via Donizetti, Via Monteverdi, Via Vivaldi, Via Bach, Via Beethoven, Via Mozart, Via Chopin, Via Schubert, Via Strauss, Via Brahms, Via Wagner, Via Debussy, NumeroCartaCredito: 2319335123122931}	Messaggio di errore - Indirizzo non valido troppo lungo!	-
9	Numero carta di credito > di 16 caratteri significative	Nome/Cognome valido, E-mail valida, Telefono valido, Indirizzo valido, Numero Carta Credito [ERROR]		{Nome: Rita, Cognome: Culla, E-mail: rita.culla@gmail.com, Telefono:0812334412, Indirizzo: Via Pisa Centro, NumeroCartaCredito: “2319335126911573683 93”}	Messaggio di errore - Numero di carta di credito troppo grande!	-
10	Numero carta di credito < di	Nome/Cognome valido, E-mail		{Nome: Mario, Cognome: Coccia, E-mail: mario.cocc1@gmail.com,	Messaggio di errore - Numero	-

	16 caratteri significative	valida, Telefono valido, Indirizzo valido, Numero Carta Credito [ERROR]		Telefono:3318922344, Indirizzo: Via Quattro Giornate, NumeroCartaCredito: “2319335126”}	di carta di credito troppo piccolo!	
11	Numero carta di credito contente caratteri o simboli	Nome/Cognome valido, E-mail valida, Telefono valido, Indirizzo valido, Numero Carta Credito [ERROR]		{Nome: Riccardo, Cognome: Gandolfo, E-mail: riccardogandolfi@email.it, Telefono:0812334112, Indirizzo: Via Napoli , NumeroCartaCredito: “sadsd”}	Messaggio di errore - Numero carta di credito non riconosciuto!	-

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “EffettuaCheckIn”.

Il numero di test da effettuarsi senza particolari vincoli è: **3 * 3 * 2 * 2 = 36**

Introduciamo i vincoli [ERROR].

Il numero di test da eseguire per testare singolarmente i vincoli è 6 (2 per Nome / Cognome, 2 per E-mail, 1 per Data Arrivo, 1 per Stato).

Il numero di test risultante è: **(1 * 1 * 1 * 1) + 6 = 7**.

NOME / COGNOME	E-MAIL	DATA ARRIVO	STATO
➢ Stringa di caratteri di lunghezza <= 50 ➢ Stringa di caratteri di lunghezza > 50 [ERROR] ➢ Stringa che contiene simboli che non sono caratteri [ERROR]	➢ Stringa in cui è presente il simbolo @ ➢ Stringa in cui non è presente il simbolo @ [ERROR] ➢ Stringa in cui il dominio dell'e-mail non è valido [ERROR]	➢ Data con formato valido (dd/mm/yyyy) ➢ Data con formato non valido [ERROR]	➢ Stringa di caratteri uguale a "PRENOTATA" ➢ Stringa di caratteri diversa da "PRENOTATA" [ERROR]

TEST SUITE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE
1	Tutti input validi	Nome valido, Cognome valido, E-mail valida, Data Arrivo valida, Stato valido	Lo stato delle camere deve essere “PRENOTATA”	{Nome: “Arianna” Cognome: “Luccina” E-mail: arianna_luccina@email.it Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Controllo valido	Il personale della Reception ha effettuato il check-in.
2	Stringa di caratteri > 50	Nome non valido [ERROR], Cognome valido, E-mail valida, Data Arrivo valida, Stato valido	Nessuna	{Nome: “ritamartinariccardoalessandroantonioluciafrancesca” Cognome: “Pioni” E-mail: arianna_luccina@email.it Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Messaggio di errore - Nome non valido	Impossibile effettuare il check-in.
3	Stringa con numeri / simboli	Nome non valido [ERROR], Cognome valido, E-mail valida, Data	Nessuna	{Nome: “Luca2?” Cognome: “Finetta” E-mail: lucafinetta0@outlook.it,	Messaggio di errore - Nome non valido	Impossibile effettuare il check-in.

		Arrivo valida, Stato valido		Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }		
4	Stringa di caratteri > 50	Nome valido, Cognome non valido [ERROR], E-mail valida, Data Arrivo valida, Stato valido	Nessuna	{Nome: “Luca2?” Cognome: “Finetta” E-mail: lucafinetta0@outlook.it Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Messaggio di errore - Cognome non valido	Impossibile effettuare il check-in.
5	Stringa con numeri / simboli	Nome valido, Cognome non valido [ERROR], E-mail valida, Data Arrivo valida, Stato valido	Nessuna	{Nome: “Rita” Cognome: “Cino!”!”” E-mail: cinorita@gmail.com Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Messaggio di errore - Cognome non valido	Impossibile effettuare il check-in.
6	Stringa in cui non è presente il simbolo @	Nome valido, Cognome valido, E-mail non valida [ERROR], Data Arrivo valida, Stato valido	Nessuna	{Nome: “Antonio” Cognome: “Colle” E-mail: antonio.collegmail.com Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Messaggio di errore - E-mail non valida	Impossibile effettuare il check-in.
7	Stringa in cui il dominio dell’e-mail non è valido	Nome valido, Cognome valido, E-mail non valida [ERROR], Data Arrivo valida, Stato valido	Nessuna	{Nome: “Pasquale” Cognome: “Cercone” E-mail: pasqualecercone@gmmmail.com Data Arrivo: 2024/06/20 Stato: “PRENOTATA” }	Messaggio di errore - E-mail non valida	Impossibile effettuare il check-in.
8	Formato data non valido	Nome valido, Cognome valido, E-mail valida, Data		{Nome: “Franco” Cognome: “Rucci” E-mail: franco1rucci@email.it Data	Messaggio di errore - Data Arrivo non valida	Impossibile effettuare il check-in.

		Arrivo non valida, Stato valido		Arrivo: 20240620 Stato: “PRENOTATA” }		
9	Stato diverso da “PRENOTATA” [ERROR]	Nome valido, Cognome valido, E-mail valida, Data Arrivo valida, Stato non valido		{Nome: “Piera” Cognome: “Linito” E-mail: pierolinito@libero.it Data Arrivo: 2024/06/20 Stato: “LIBERA” }	Messaggio di errore - Stato non valido	Impossibile effettuare il check-in.

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “EffettuaCheckOut”.

Il numero di test da effettuarsi senza particolari vincoli è: **3 * 3 * 2 * 2 = 36**

Introduciamo i vincoli [ERROR].

Il numero di test da eseguire per testare singolarmente i vincoli è 6 (2 per Nome / Cognome, 2 per E-mail, 1 per Data Partenza, 1 per Stato).

Il numero di test risultante è: **(1 * 1 * 1 * 1) + 6 = 7**.

NOME / COGNOME	E-MAIL	DATA PARTENZA	STATO
➢ Stringa di caratteri di lunghezza <= 50 ➢ Stringa di caratteri di lunghezza > 50 [ERROR] ➢ Stringa che contiene simboli che non sono caratteri [ERROR]	➢ Stringa in cui è presente il simbolo @ ➢ Stringa in cui non è presente il simbolo @ [ERROR] ➢ Stringa in cui il dominio dell'e-mail non è valido [ERROR]	➢ Data con formato valido (dd/mm/yyyy) ➢ Data con formato non valido [ERROR]	➢ Stringa di caratteri uguale a "OCCUPATA" ➢ Stringa di caratteri diversa da "OCCUPATA" [ERROR]

TEST SUITE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE
1	Tutti input validi	Nome valido, Cognome valido, E-mail valida, Data Partenza valida, Stato valido	Lo stato delle camere deve essere “OCCUPATA”	{Nome: “Arianna” Cognome: “Luccina” E-mail: arianna_luccina@email.it Dat Partenza: 2024/06/20 Stato: “OCCUPATA” }	Controllo valido	Il personale della Reception ha effettuato il check-out.
2	Stringa di caratteri > 50	Nome non valido [ERROR], Cognome valido, E-mail valida, Data Partenza valida, Stato valido	Nessuna	{Nome: “ritamartinariccardoalessandroantonioluciafrancesca” Cognome: “Pioni” E-mail: arianna_luccina@email.it Data Partenza: 2024/06/20 Stato: “OCCUPATA” }	Messaggio di errore - Nome non valido	Impossibile effettuare il check-out.
3	Stringa con numeri / simboli	Nome non valido [ERROR], Cognome valido, E-mail valida, Data	Nessuna	{Nome: “Luca2?” Cognome: “Finetta” E-mail: lucafinetta0@outlook.it,	Messaggio di errore - Nome non valido	Impossibile effettuare il check-out.

		Partenza valida, Stato valido		Data Partenza: 2024/06/20 Stato: “OCCUPATA” }		
4	Stringa di caratteri > 50	Nome valido, Cognome non valido [ERROR], E-mail valida, Data Partenza valida, Stato valido	Nessuna	{Nome: “Luca2?” Cognome: “Finetta” E-mail: lucafinetta0@outlook.it, Data Partenza: 2024/06/20 Stato: “OCCUPATA” }	Messaggio di errore - Cognome non valido	Impossibile effettuare il check-out.
5	Stringa con numeri / simboli	Nome valido, Cognome non valido [ERROR], E-mail valida, Data Partenza valida, Stato valido	Nessuna	{Nome: “Rita” Cognome: “Cino!”!”” E-mail: cinorita@gmail.com, Data Partenza: 2024/06/20 Stato: “OCCUPATA” }	Messaggio di errore - Cognome non valido	Impossibile effettuare il check-out.
6	Stringa in cui non è presente il simbolo @	Nome valido, Cognome valido, E-mail non valida [ERROR], Data Partenza valida, Stato valido	Nessuna	{Nome: “Antonio” Cognome: “Colle” E-mail: antonio.collegmail.com, Data Partenza: 2024/06/20 Stato: “OCCUPATA” }	Messaggio di errore - E-mail non valida	Impossibile effettuare il check-out.
7	Stringa in cui il dominio dell’e-mail non è valido	Nome valido, Cognome valido, E-mail non valida [ERROR], Data Partenza valida, Stato valido	Nessuna	{Nome: “Pasquale” Cognome: “Cercone” E-mail: pasqualecercone@gmmmail.com, Data Partenza: 2024/06/20 Stato: “OCCUPATA” }	Messaggio di errore - E-mail non valida	Impossibile effettuare il check-out.
8	Formato data non valido	Nome valido, Cognome valido, E-mail valida, Data		{Nome: “Franco” Cognome: “Rucci” E-mail: franco1rucci@email.it Data	Messaggio di errore - Data	Impossibile effettuare il check-out.

		Partenza non valida, Stato valido		Partenza: 20240620 Stato: “OCCUPATA” }	Partenza non valida	
9	Stato diverso da “PRENOTATA” [ERROR]	Nome valido, Cognome valido, E-mail valida, Data Partenza valida, Stato non valido		{Nome: “Piera” Cognome: “Linito” E-mail: pierolinito@libero.it Data Partenza: 2024/06/20 Stato: “LIBERA” }	Messaggio di errore - Stato non valido	Impossibile effettuare il check-out.

PIANO DI TEST UTILIZZANDO IL METODO DEL *CATEGORY-PARTITION TESTING* PER LA FUNZIONALITÀ “GeneraReportElencoNotti”.

Il numero di test da effettuarsi senza particolari vincoli è: **3 = 3**

Introduciamo i vincoli [ERROR].

Il numero di test da eseguire per testare singolarmente i vincoli è 2 (2 per E-mail).

Il numero di test risultante è: **(1) + 2 = 3**.

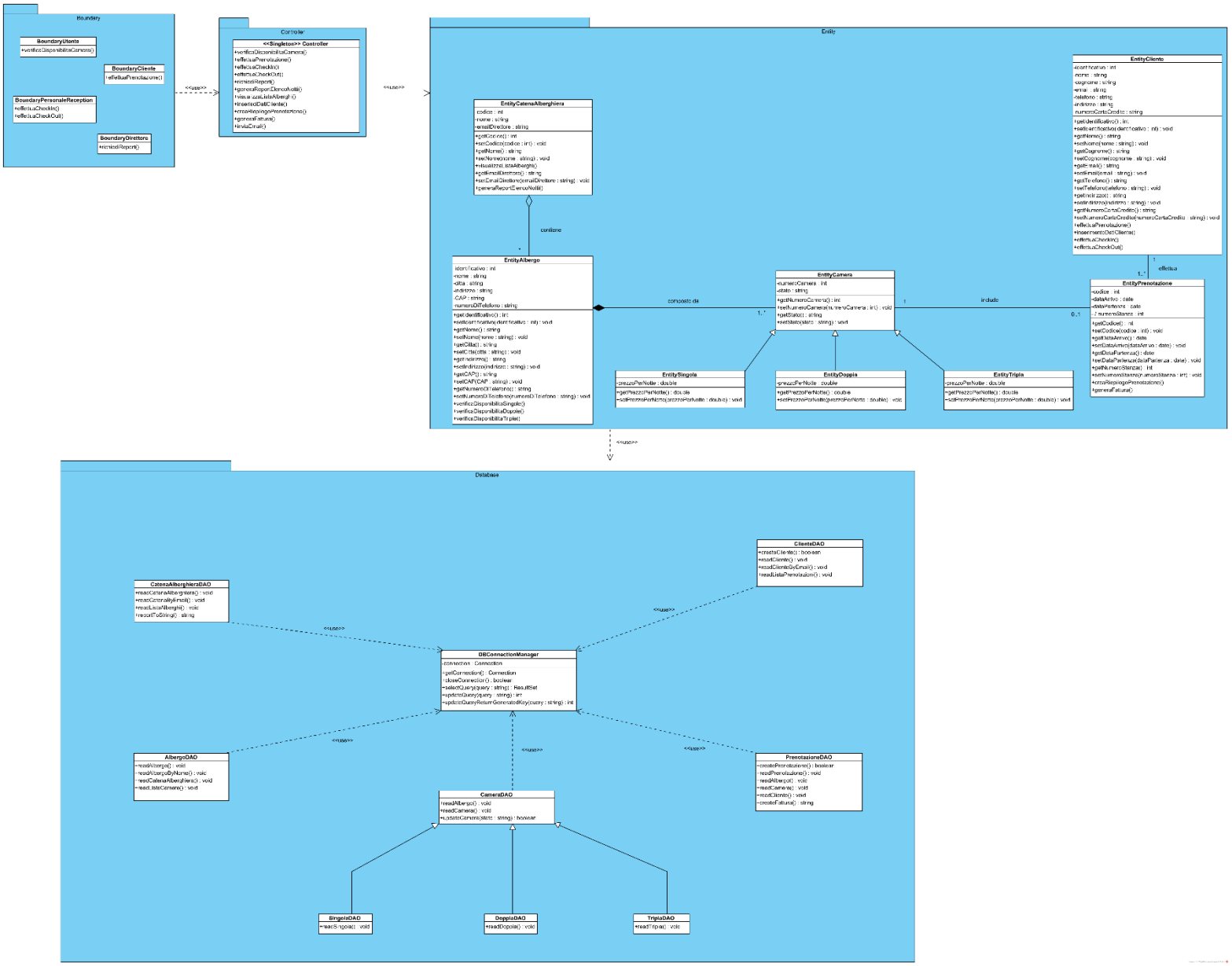
EMAIL
➢ Stringa in cui è presente il simbolo @ ➢ Stringa in cui non è presente il simbolo @ [ERROR] ➢ Stringa in cui il dominio dell’e-mail non è valido [ERROR]

TEST SUITE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE
1	Tutti input validi	E-mail valida	Il primo giorno di ogni mese il sistema invia per e-mail al Direttore della catena alberghiera un report con l'elenco delle notti in cui ciascuna camera è stata occupata nel mese precedente	{ E-mail: mariaelenatacco@email.it}	E-mail inviata con successo!	Il direttore della catena alberghiera riceve l'e-mail con l'allegato
2	Stringa in cui non è presente il simbolo @	E-mail [ERROR],	Nessuna	{E-mail: micheledimeo_email.it }	Messaggio di errore - E-mail non valida!	-
3	Stringa in cui il dominio dell'e-mail non è valido	E-mail [ERROR]	Nessuna	{ E-mail: mariaelenatacco@ lapadula.fug}	Messaggio di errore - Dominio dell'e-mail non riconosciuto!	-

5. PROGETTAZIONE

5.1. Diagramma delle classi



5.2. Diagrammi di sequenza

Diagramma di sequenza di progettazione per il caso d'uso **UC1 : VerificaDisponibilitaCamera**

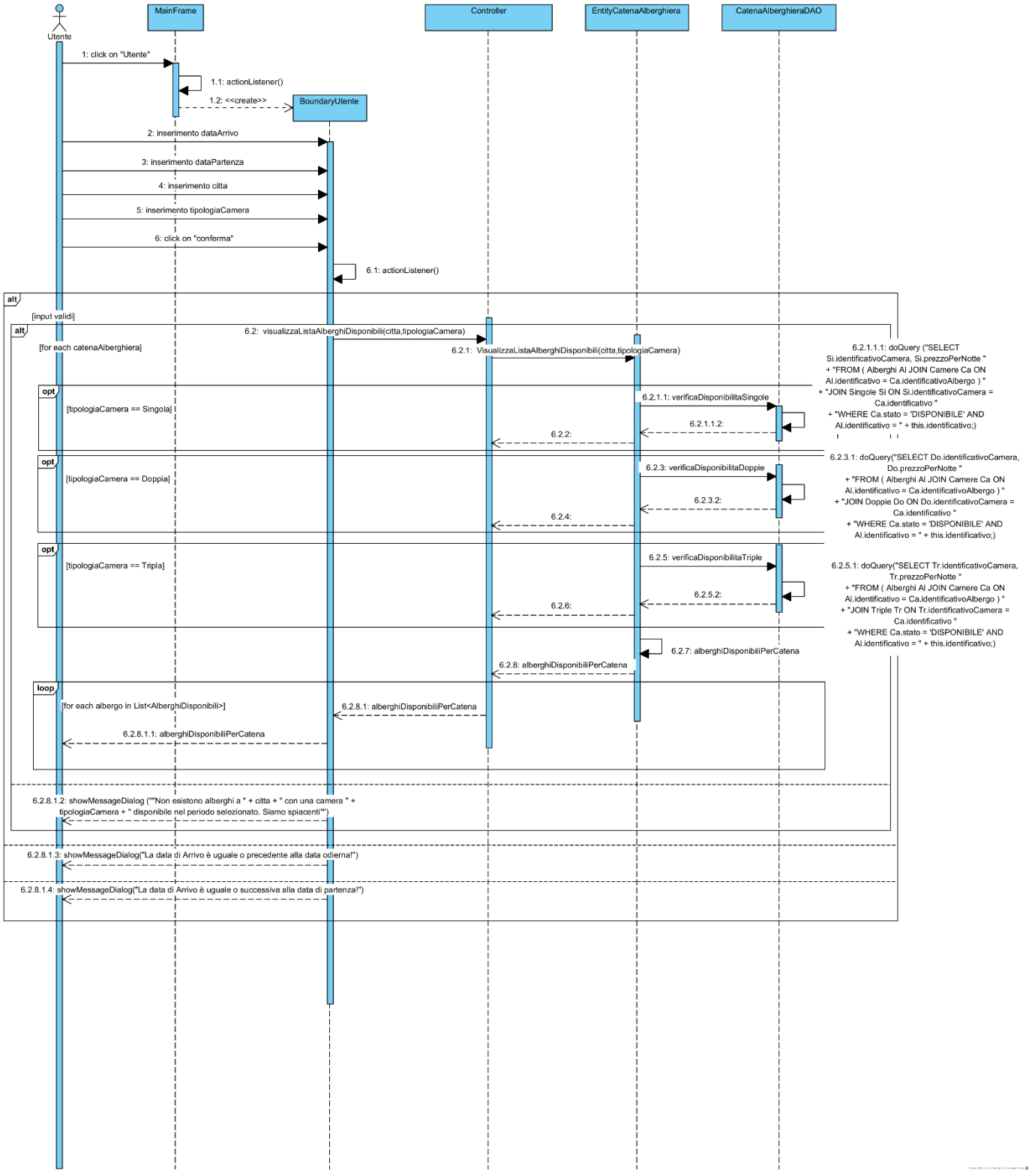


Diagramma di sequenza di progettazione per il caso d'uso **UC3 : EffettuaCheckIn**

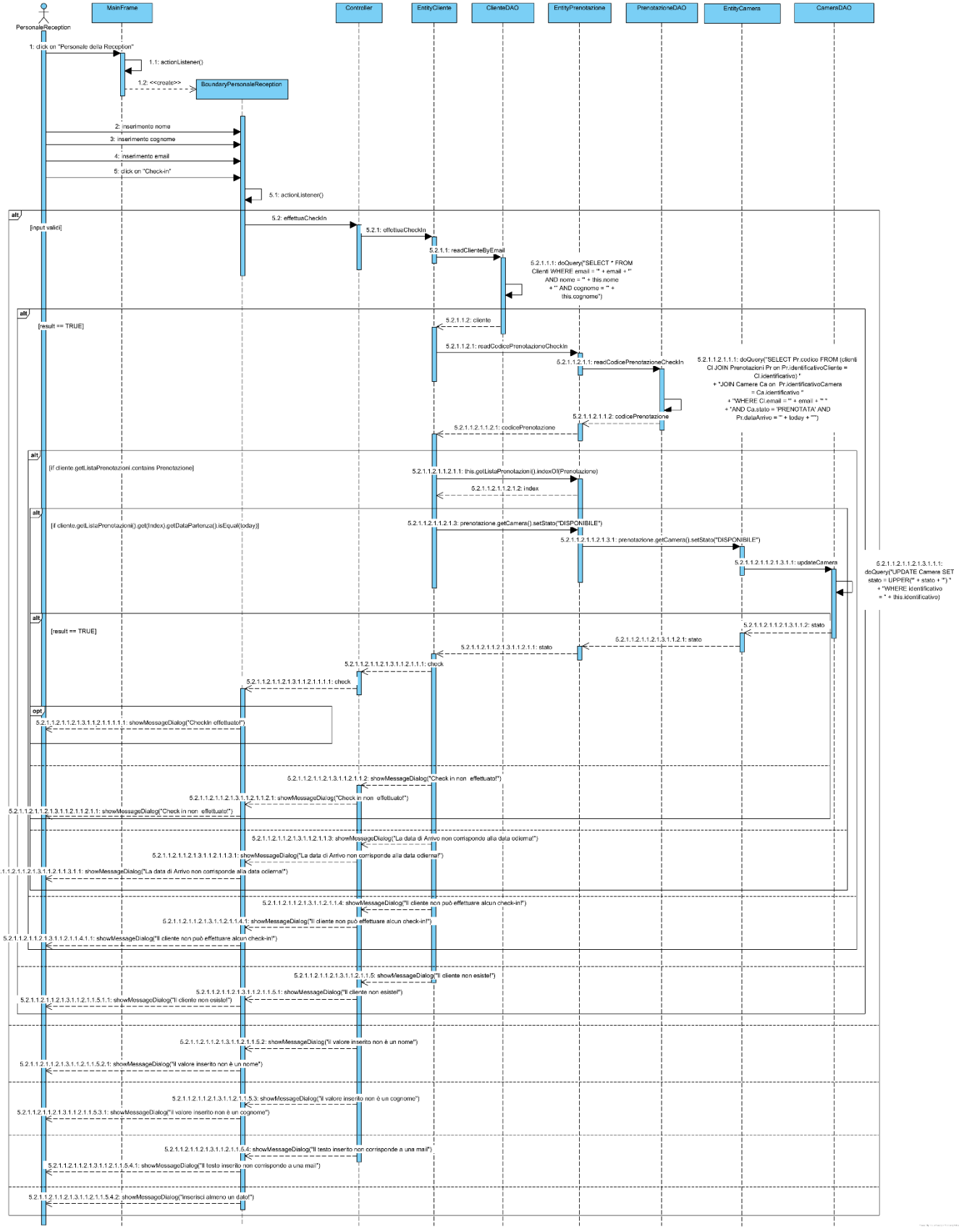
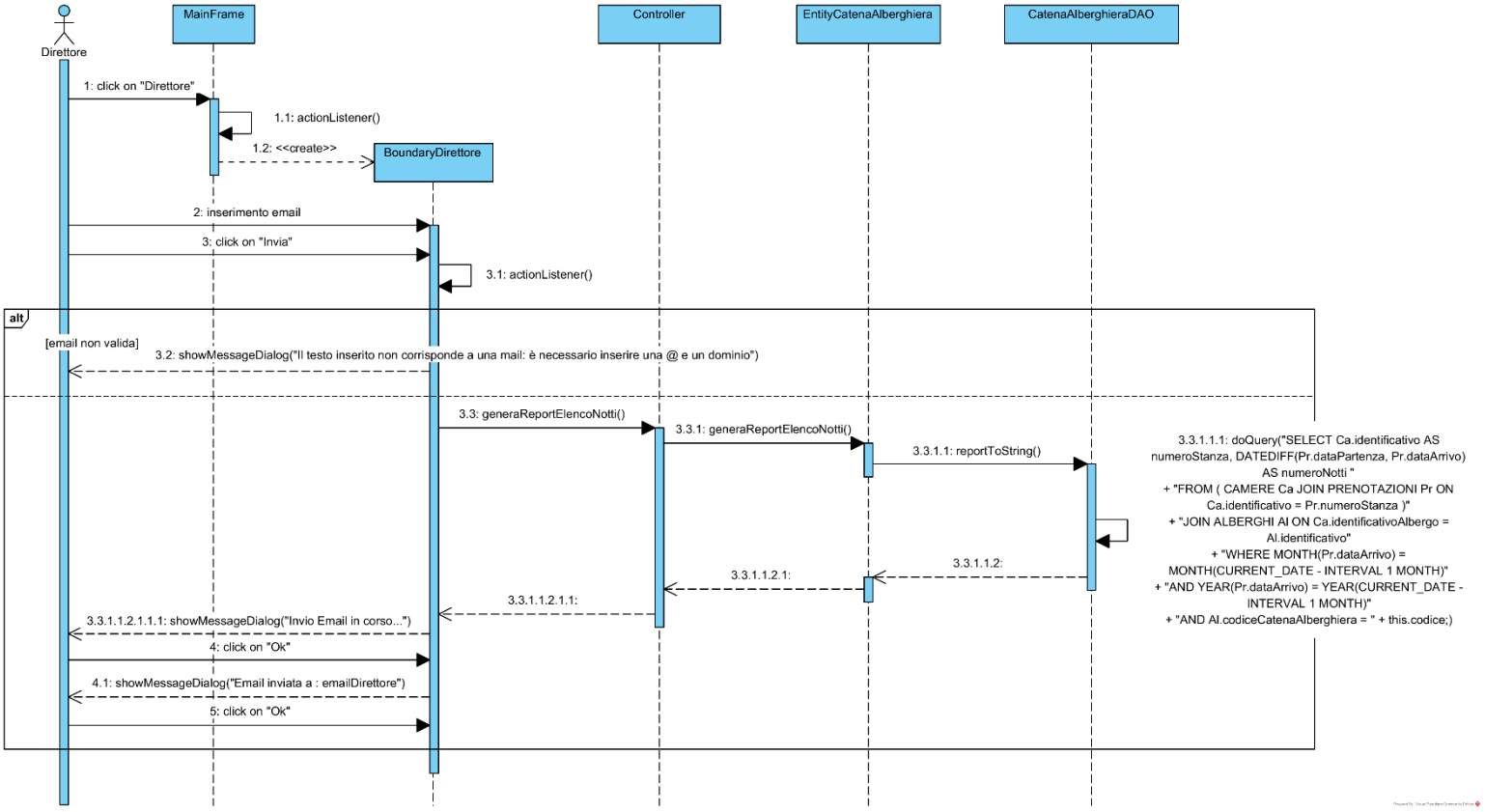


Diagramma di sequenza di progettazione per il caso d'uso **UC5 : GeneraReportElencoNotti**



6. IMPLEMENTAZIONE

PACKAGE

In fase di implementazione si è deciso di utilizzare un approccio BCED con l'utilizzo dei package:

- **Boundary**, si occupa dell'interfacciamento con l'utente;
- **Control**, si occupa di implementare le responsabilità del sistema (identificate nella fase di analisi dei casi d'uso);
- **Entity**, si occupa di implementare le classi precedentemente definite nel diagramma delle classi;
- **Database**, si occupa dell'interfacciamento diretto con il database.

Oltre ad essi, si è deciso di utilizzare il package **Exception**, nel quale sono state definite delle eccezioni personalizzate. Infine, l'ultimo package utilizzato è quello delle **Images**, sfruttate a livello JFrame.

CLASSI

Si analizza il contenuto dei singoli package.

- All'interno del package **Boundary** sono state definite le classi:
  - MainFrame, che rappresenta la principale interfaccia grafica;
  - BoundaryCliente, BoundaryDirettore, BoundaryPersonaleReception, BoundaryUtente, responsabili della logica di presentazione delle funzionalità disponibili rispettivamente di Clenti, Direttori, Personali della Reception e Utenti.
- All'interno del package **Controller** è stata definita la classe:
  - Controller, che controlla l'esecuzione del processo. Per quest'ultimo, si è deciso di utilizzare il design pattern Singleton, esso permette di definire e utilizzare una singola istanza, che verrà creata una sola volta e richiamata quando deve essere usata.
- All'interno del package **Entity**, sono state definite le classi:
  - EntityAlbergo, è responsabile di gestire e conservare i dati relativi a un albergo;
  - EntityCamera, è responsabile di gestire e conservare i dati relativi a una camera;

- EntityCatenaAlberghiera, è responsabile di gestire e conservare i dati relativi a una catena alberghiera;
- EntityCliente, è responsabile di gestire e conservare i dati relativi a un cliente;
- EntityDoppia, è responsabile di gestire e conservare i dati relativi a una camera doppia;
- EntityPrenotazione, è responsabile di gestire e conservare i dati relativi a una prenotazione;
- EntityServizioEmail, è responsabile dell'invio delle mail;
- EntitySingola, è responsabile di gestire e conservare i dati relativi a una camera singola;
- EntityTripla, è responsabile di gestire e conservare i dati relativi a una tripla;

- All'interno del package **Database**, sono state definite le classi:

  - DBConnectionManager, è responsabile di gestire la connessione col Database;
  - AlbergoDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a un albergo dal Database;
  - CameraDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una camera dal Database;
  - CatenaAlberghieraDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una catena alberghiera dal Database;
  - ClienteDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a un cliente dal Database;
  - DoppiaDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una camera doppia dal Database;
  - PrenotazioneDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una prenotazione dal Database;
  - SingolaDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una camera singola dal Database;
  - TriplaDAO, è responsabile dell'estrazione, lettura e aggiornamento dei dati relativi a una camera tripla dal Database.

ECCEZIONI

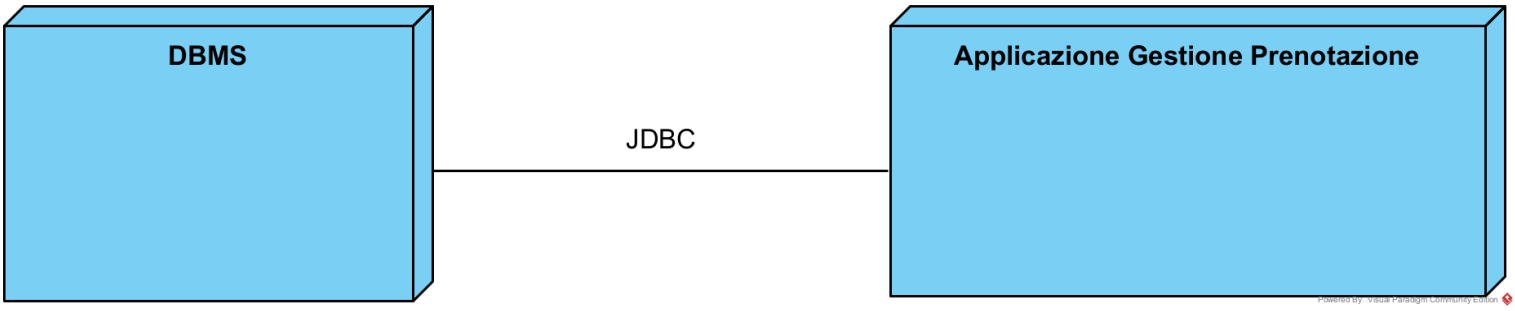
All'interno del package **Exception**, sono state definite le seguenti eccezioni:

1. DataNotValidException, si scatena se il formato della data non è valido;
2. InvalidCameraTypeException, si scatena se il tipo della camera non è valido;
3. NoDirectorFoundException, si scatena se non è stato trovato nessun direttore;
4. NoRoomException, si scatena se non vengono trovate delle camere;
5. NotFirstDayException, si scatena se non è il primo giorno del mese.

LIBRERIE

Per l'installazione e l'esecuzione del programma, è necessario includere il file my-sql-connector-java-8.0.33.jar per realizzare la connessione con il Database creato in MySQLServer. Si è deciso, inoltre, di utilizzare le librerie javax.mail-1.6.2.jar per l'invio delle e-mail tramite servizio di posta esterno.

DIAGRAMMA DI DEPLOYMENT



LOC – LLOC

- VerificaDisponibilitaCamera:

LOC: 80 (funzioneBase) + 267 (validazioneStringa) + 38 (validazioneData) + 44 (visualizzaListaAlbergiDisponibili) = 429
LLOC: 65 (funzioneBase) + 142 (validazioneStringa) + 26 (validazioneData) + 23 (visualizzaListaAlberghiDisponibili) = 256

- EffettuaPrenotazione:

LOC: 53 (funzioneBase) + 267 (validazioneStringa) + 33 (capitalizeString) + 112 (effettuaPrenotazione) + 17 (checkUniqueEmail) + 30 (readClienteByAllData) + 17 (getIdentificativoNuovoCliente) + 9 (inserimentoDatiCliente) + 21 (createCliente) + 18 (getCodiceNuovaPrenotazione) + 21 (updateCamera) + 20 (createPrenotazione) + 8 (generaPrezzoComplessivo) + 25 (creaRiepilogoPrenotazione) + 53 (inviaEmail) = 704
LLOC: 36 (funzioneBase) + 142 (validazioneStringa) + 14 (capitalizeString) + 66 (effettuaPrenotazione) + 13(checkUniqueEmail) + 26 (readClienteByData) + 13 (getIdentificativoNuovoCliente) + 9 (inserimentoDatiCliente) + 17 (createCliente) + 13 (getCodiceNuovaPrenotazione) + 16 (updateCamera) + 17 (createPrenotazione) + 6 (generaPrezzoComplessivo) +19(creaRiepilogoPrenotazione) + 28(inviaEmail) = 435

- EffettuaCheckIn

LOC: 20 (funzioneBase) + 267 (validazioneStringa) + 48 (effettuaCheckIn) + 24 (readCodicePrenotazioneCheckIn) + 21 (updateCamera) = 380
LLOC: 15 (funzioneBase) + 142 (validazioneStringa) + 39 (effettuaCheckIn) + 19 (readCodicePrenotazioneCheckIn) + 16 (updateCamera) = 231

- EffettuaCheckOut

LOC: 18 (funzioneBase) + 267 (validazioneStringa) + 57 (effettuaCheckOut) + 22 (readCodicePrenotazioneCheckOut) + 21 (updateCamera) = 385
LLOC: 13 (funzioneBase) + 142 (validazioneStringa) + 39 (effettuaCheckOut) + 17 (readCodicePrenotazioneCheckOut) + 16 (updateCamera) = 227

- GeneraReportElencoNotti

LOC: 42 (funzioneBase) + 13 (isFirstDayOfMonth) + 267 (validazioneStringa) + 80 (reportMonthToString) + 53 (inviaEmail) + 5 (generaReportElencoNotti) + 41 (reportToString) = 501
LLOC: 30 (funzioneBase) + 9 (isFirstDayOfMonth) + 142 (validazioneStringa) + 49 (reportMonthToString) + 28 (inviaEmail) + 4 (generaReportElencoNotti) + 31 (reportToString) = 293

La stima dei costi aveva previsto un numero di LLOC per le funzioni analizzate sempre maggiore; quindi, lo scostamento in positivo indica che la stima effettuata in fase di analisi non è stata ecceduta.

7. TESTING

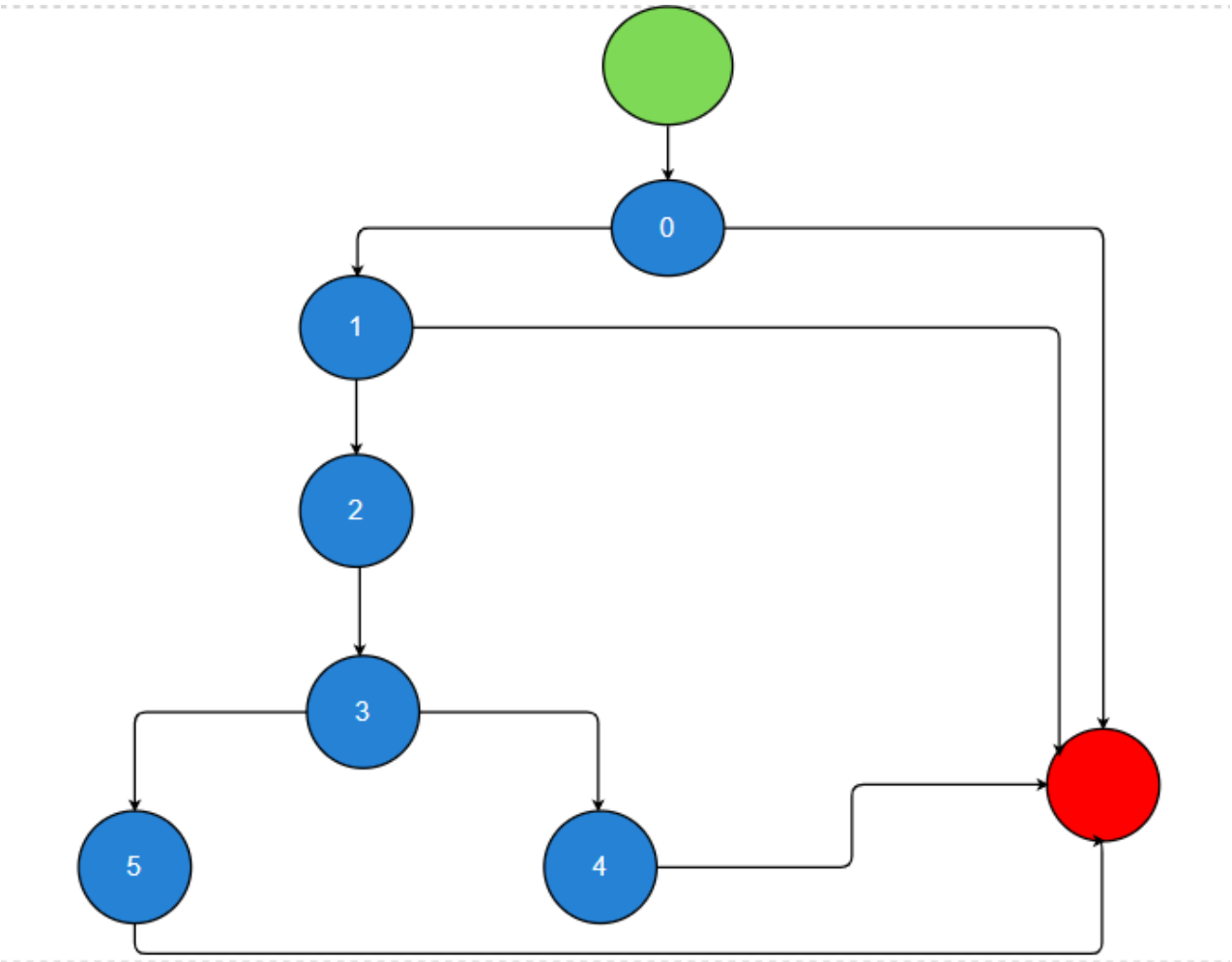
7.1. Test strutturale

7.1.1. Complessità ciclomatica

VERIFICA DISPONIBILITÀ CAMERA

```
public ArrayList<String> verificaDisponibilitaCamera(String citta, String tipologiaCamera, String dataArrivoStringa, String
dataPartenzaStringa, JTextArea textArea, JButton btn) {

            ArrayList<String> nomiAlberghi = new ArrayList<String>();

            validazioneStringa(citta,Contesto.CITTA); //0

            try {
                if(validazioneData(dataArrivoStringa,dataPartenzaStringa)) { //1
                    try {
                        ArrayList<EntityAlbergo> alberghiDisponibiliPerCatena = new ArrayList<EntityAlbergo>(); //2

                        for(int i = 1; i <= numeroCatene; i++) {
                            EntityCatenaAlberghiera catena = new EntityCatenaAlberghiera(i);

                            ArrayList<EntityAlbergo> alberghiDisponibili =
catena.visualizzaListaAlberghiDisponibili(citta,tipologiaCamera);

                            alberghiDisponibiliPerCatena.addAll(alberghiDisponibili);
                        }

                        if(!alberghiDisponibiliPerCatena.isEmpty()) { //3
                            System.out.println("ALBERGHI DISPONIBILI:\n"); //4

                            for(EntityAlbergo albergo : alberghiDisponibiliPerCatena) {
                                System.out.println(albergo);
                                textArea.append("ALBERGHI DISPONIBILI:\n" + albergo.toString());
                            }

                            btn.setVisible(true);
                            btn.setEnabled(true);

                            for(EntityAlbergo albergo : alberghiDisponibiliPerCatena) {
                                nomiAlberghi.add(albergo.getNome());
                            }
```

```
                }else { //6

                    btn.setVisible(false);
                    btn.setEnabled(false);

                    throw new NoRoomException("Non esistono alberghi a " + citta + " con una camera " +
                            tipologiaCamera + " disponibile nel periodo selezionato. Siamo spiacenti");
                }

            }catch(NoRoomException e){

                btn.setVisible(false);
                btn.setEnabled(false);
                // A questo punto non può procedere a effettua prenotazione
                JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
            }
        }else {

            btn.setVisible(false);
            btn.setEnabled(false);
        }
    } catch (DataNotValidException e) {

        btn.setVisible(false);
        btn.setEnabled(false);
        // Qui ritorna alla schermata di inserimento
        JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
    }

    return nomiAlberghi;
```



Numero Ciclomatico di McCabe:

10 (#nArchi) – 8 (#nNodi) + 2 = 4

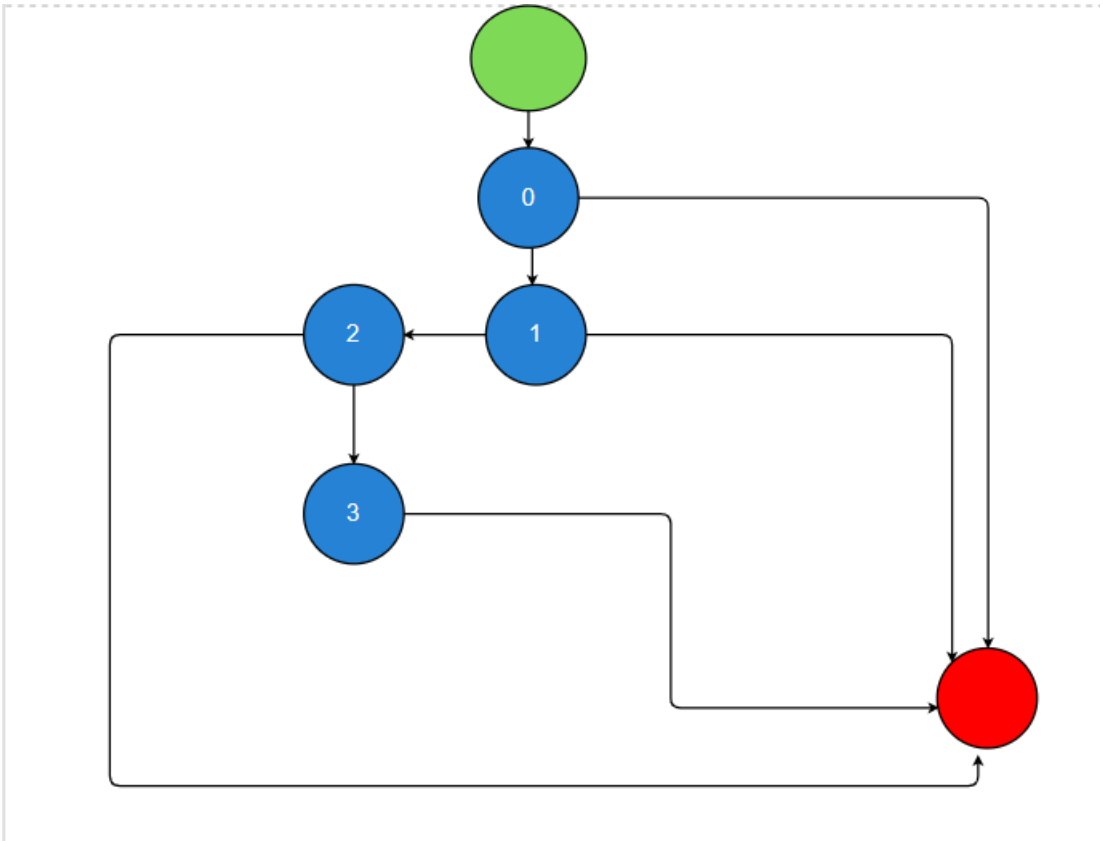
Cammini Indipendenti:

1. **0**
2. **0-1**
3. **0-1-2-3-5**
4. **0-1-2-3-4**

GENERA REPORT ELENCO NOTTI

```
public boolean generaReportElencoNotti(String emailDirettore) {
        boolean check = false;

        try {
            // Lo scenario inizia solo se è il primo giorno del mese (pre-condizione)
            if(isFirstDayOfMonth()) {//0

                try {
                    if(validazioneEmail(emailDirettore)) {//1
                        try {
                            EntityCatenaAlberghiera catena = new EntityCatenaAlberghiera(emailDirettore);

                            if(!catena.getEmailDirettore().isEmpty()) {//2
                                EntityServizioEmail servizioEmail = new EntityServizioEmail();

                                String oggetto = "Report elenco notti - Mese di " + reportMonthToString();

                                servizioEmail.inviaEmail(catena.getEmailDirettore(), oggetto,
catena.generaReportElencoNotti());//3

                                check = true;
                            }else {
                                // Qui torna indietro alla schermata di inserimento dell'e-mail
                            }

                        }catch(NoDirectorFoundException e) {
                            JOptionPane.showMessageDialog(null, "Il direttore è inesistente", "ERRORE",
JOptionPane.ERROR_MESSAGE);
                            e.printStackTrace();
                        }
                    }

                }catch(ArrayIndexOutOfBoundsException | UnknownHostException | IllegalArgumentException e) {
                    JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
                }

            // Throw per identificare in che casistica scatenare l'eccezione NotFirstDayException
            } else {
                throw new NotFirstDayException("Non è il primo giorno del mese: impossibile generare il report");
            }

        // Se non è il primo giorno del mese, si scatena l'eccezione NotFirstDayException
        }catch(NotFirstDayException e) {
            check = true;
```

```
            JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        return check;
    }
```



Numero Ciclomatico di McCabe:

8 (#nArchi) – 6 (#nNodi) + 2 = 4

Cammini Indipendenti:

1. **0**
2. **0-1**
3. **0-1-2**
4. **0-1-2-3**

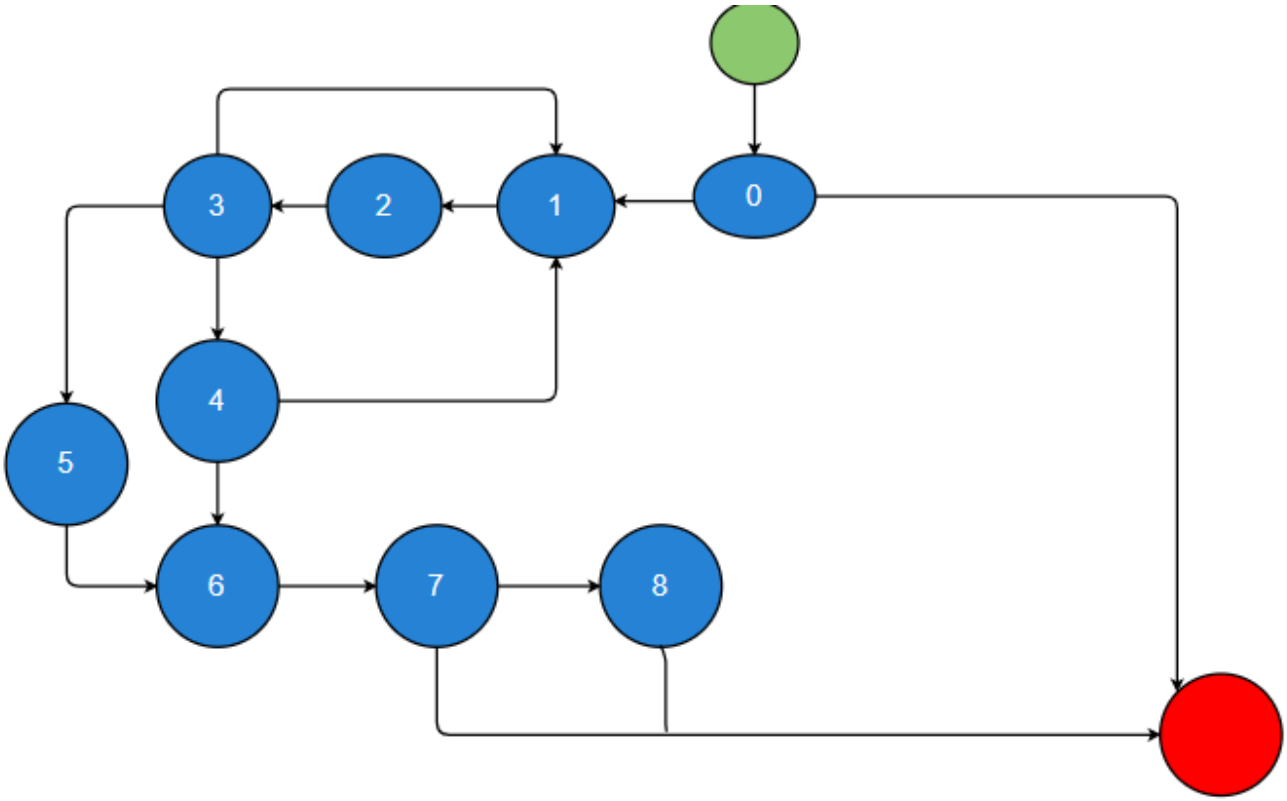
EFFETTUA PRENOTAZIONE

```
public boolean effettuaPrenotazione(String nome, String cognome, String email, String telefono,
                        String indirizzo, String numeroCartaCredito, String dataArrivoStringa,
                        String dataPartenzaStringa, String tipologiaCamera, String nomeAlbergo) {
        boolean check = false;

        try {
            if(validazioneStringa(nome,Contesto.NOME) && validazioneStringa(cognome,Contesto.COGNOME) &&
            validazioneEmail(email) && validazioneTelefono(telefono) && validazioneIndirizzo(indirizzo) &&
            validazioneNumeroCartaCredito(numeroCartaCredito)) { //0

                // Rendo tutte le iniziali maiuscole
                nome = capitalizeString(nome);
                cognome = capitalizeString(cognome);
                indirizzo = capitalizeString(indirizzo);

                EntityCliente cliente = new EntityCliente();

                try {
                    if(cliente.effettuaPrenotazione(nome,cognome,email,telefono,indirizzo,numeroCartaCredito,
                                dataArrivoStringa, dataPartenzaStringa, tipologiaCamera, nomeAlbergo)) {
                        check = true;
                    }
                } catch (NotUniqueEmailException e) {
                    JOptionPane.showMessageDialog(null, e.getMessage(), "REGISTRAZIONE NON EFFETTUATA",
JOptionPane.ERROR_MESSAGE);
                }
            }

        }catch(IllegalArgumentException | ArrayIndexOutOfBoundsException | UnknownHostException e) {
            // Qui poi si torna alla schermata di inserimento perché c'è stato qualche problema
            JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        return check;
    }


        public boolean effettuaPrenotazione(String nome, String cognome, String email, String telefono,
            String indirizzo, String numeroCartaCredito, String dataArrivoStringa, String dataPartenzaStringa,
            String tipologiaCamera, String nomeAlbergo) throws NotUniqueEmailException {
        boolean check = false;

        ClienteDAO clienteDAO = new ClienteDAO(email);
```

```
        // Se la mail inserita non esiste nel sistema e anche i dati del cliente non risultano, creo un nuovo cliente
        if(!clienteDAO.checkUniqueEmail(email) && !clienteDAO.readClienteByAllData(nome,cognome,email,telefono,indirizzo, //1
                numeroCartaCredito)) {

            // Assegno un nuovo identificativo al cliente
            this.identificativo = clienteDAO.getIdentificativoNuovoCliente(); //2

            // Inserimento dati del cliente validati //3
            this.inserimentoDatiCliente(nome,cognome,email,telefono,indirizzo,numeroCartaCredito);

            // Se i dati inseriti non sono presenti nel database, vanno salvati: creo un nuovo cliente
            if(clienteDAO.createCliente(nome,cognome,email,telefono,indirizzo,numeroCartaCredito)) { //4
                JOptionPane.showMessageDialog(null, "\nNuovo cliente registrato!", "REGISTRAZIONE EFFETTUATA",
JOptionPane.INFORMATION_MESSAGE);

                JOptionPane.showMessageDialog(null, "\nDati validi, prenotazione in corso", "PRENOTAZIONE",
JOptionPane.INFORMATION_MESSAGE); //4->7
            }else {
                JOptionPane.showMessageDialog(null, "Cliente non registrato correttamente", "REGISTRAZIONE NON EFFETTUATA",
JOptionPane.ERROR_MESSAGE);
            }

        // Se la mail esiste già e i dati del cliente coincidono, si identifica il cliente registrato
        }else if(clienteDAO.checkUniqueEmail(email) && clienteDAO.readClienteByAllData(nome,cognome,email,telefono,// 5->6
                indirizzo,numeroCartaCredito)) {
            this.identificativo = clienteDAO.getIdentificativo();

            JOptionPane.showMessageDialog(null, "\nBentornato! Siamo lieti che tu sia tornato a prenotare con Italian
Travel!", "CLIENTE GIÀ REGISTRATO", JOptionPane.INFORMATION_MESSAGE);

            JOptionPane.showMessageDialog(null, "\nDati validi, prenotazione in corso", "PRENOTAZIONE",
JOptionPane.INFORMATION_MESSAGE);

        // Se la mail esiste già e i dati del cliente NON coincidono, non si può creare il cliente
        }else if(clienteDAO.checkUniqueEmail(email) && !clienteDAO.readClienteByAllData(nome,cognome,email,telefono,
                indirizzo,numeroCartaCredito)) {
            throw new NotUniqueEmailException("L'e-mail inserita appartiene già ad un altro cliente!");
        }

        // Creo la nuova prenotazione
        EntityPrenotazione prenotazione = new EntityPrenotazione();//  6

        // Setto l'albergo in base al nome
        EntityAlbergo albergo = new EntityAlbergo(nomeAlbergo);

        // Conversione date
```

```
        DateTimeFormatter formatter = DateTimeFormatter.ofPattern("dd/MM/yyyy");

        LocalDate dataArrivo = LocalDate.parse(dataArrivoStringa, formatter);
        LocalDate dataPartenza = LocalDate.parse(dataPartenzaStringa, formatter);

        // Per la creazione della nuova prenotazione nel database
        PrenotazioneDAO prenotazioneDAO = new PrenotazioneDAO();

        prenotazione.setCodice(prenotazioneDAO.getCodiceNuovaPrenotazione());
        prenotazione.setDataArrivo(dataArrivo);
        prenotazione.setDataPartenza(dataPartenza);
        prenotazione.setAlbergo(albergo);
        prenotazione.setCliente(this);

        // Logica di assegnazione delle camere: la prima della lista
        if(tipologiaCamera.equalsIgnoreCase("Singola")) {
            prenotazione.setCamera(albergo.caricaSingoleDisponibili().get(0));
        }else if(tipologiaCamera.equalsIgnoreCase("Doppia")) {
            prenotazione.setCamera(albergo.caricaDoppieDisponibili().get(0));
        }else if(tipologiaCamera.equalsIgnoreCase("Tripla")) {
            prenotazione.setCamera(albergo.caricaTripleDisponibili().get(0));
        }else {
            JOptionPane.showMessageDialog(null, "Camera " + tipologiaCamera + " inesistente nell'albergo "
                    + prenotazione.getAlbergo().getNome() + "!", "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        prenotazione.setNumeroStanza(prenotazione.getCamera().getNumeroCamera());

        // Imposto la camera appena prenotata come "PRENOTATA" //7
        prenotazione.getCamera().setStato("PRENOTATA");

        // Aggiornamento della camera
        CameraDAO camera = new CameraDAO(prenotazione.getCamera().getIdentificativo());

        // Se l'aggiornamento della camera funziona correttamente, procedo all'inserimento della prenotazione
        if(camera.updateCamera(prenotazione.getCamera().getStato())){

            // Creo la prenotazione con tutti i dati necessari nel  database //8
            if(prenotazioneDAO.createPrenotazione(dataArrivo,dataPartenza,prenotazione.getNumeroStanza(),
                    prenotazione.getAlbergo().getIdentificativo(),prenotazione.getCamera().getIdentificativo(),
                    this.identificativo)) {

                JOptionPane.showMessageDialog(null, prenotazione.getCamera(), "INFO PRENOTAZIONE",
JOptionPane.INFORMATION_MESSAGE);

                // Aggiungo la prenotazione alla lista delle prenotazioni
```

```
                this.getListaPrenotazioni().add(prenotazione);

                JOptionPane.showMessageDialog(null, "\nPrenotazione effettuata correttamente", "PRENOTAZIONE EFFETTUATA",
JOptionPane.INFORMATION_MESSAGE);

                // Se la prenotazione è andata a buon fine, il sistema invia al Cliente un report
                // del riepilogo prenotazione
                String oggetto = "Riepilogo prenotazione - Alloggio nell'albergo " + prenotazione.getAlbergo().getNome();

                String testo = prenotazione.creaRiepilogoPrenotazione(prenotazione.getAlbergo(),
                                                prenotazione.getCamera(),tipologiaCamera);

                EntityServizioEmail servizioEmail = new EntityServizioEmail();

                servizioEmail.inviaEmail(this.email, oggetto, testo);

                check = true;

            }else {
                JOptionPane.showMessageDialog(null, "Prenotazione non effettuata correttamente. "
                        + "È necessario ripetere la procedura", "ERRORE", JOptionPane.ERROR_MESSAGE);
            }

        }else {
            // Questo è un caso raro di errore interno, non dovrebbe interessare al Cliente
            JOptionPane.showMessageDialog(null, "Prenotazione non effettuata correttamente. "
                    + "È necessario ripetere la procedura", "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        return check;
    }
```



Numero Ciclomatico di McCabe:

14 (#nArchi) – 11 (#nNodi) + 2 = 4

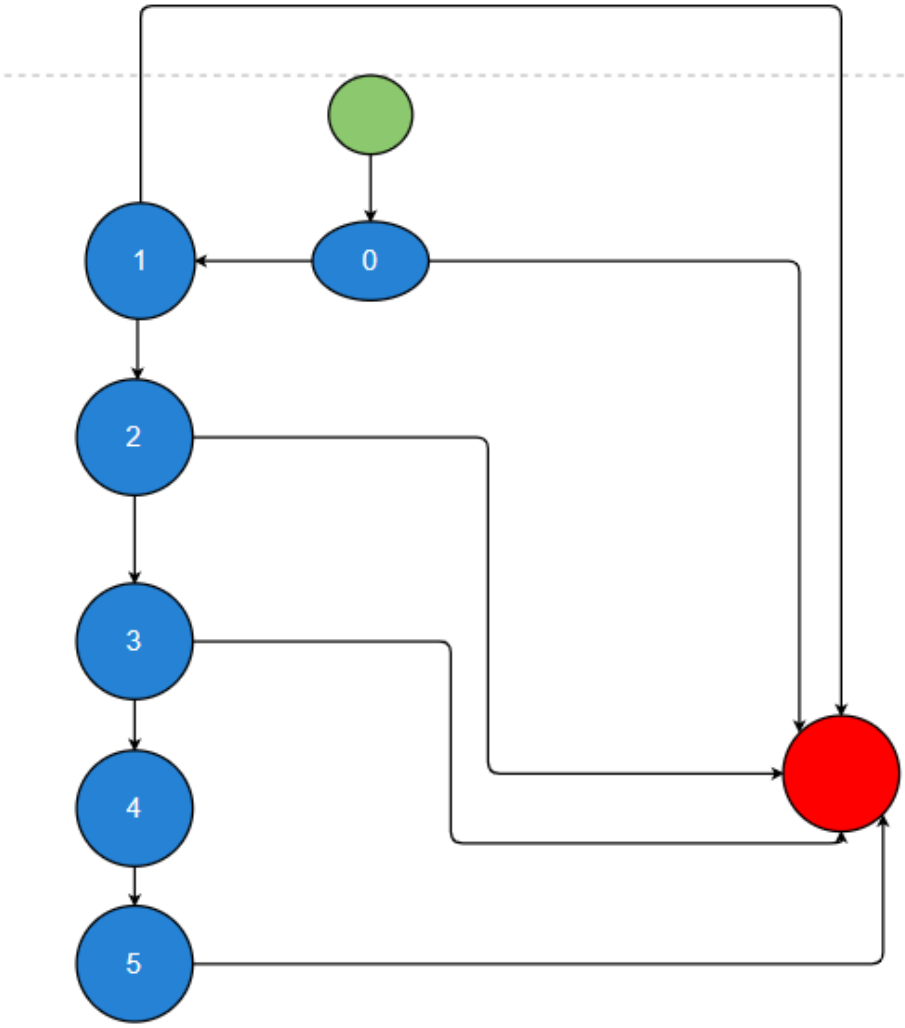
Cammini Indipendenti:

1. **0**
2. **0-1-2-3**
3. **0-1-2-3-4**
4. **0-1-2-3-4-6-7**
5. **0-1-2-3-4-6-7-8**

EFFETTUA CHECK-IN

```
public boolean effettuaCheckIn(String nome, String cognome, String email) {
        boolean check = false;

        try {
            if(validazioneStringa(nome,Contesto.NOME) && validazioneStringa(cognome,Contesto.COGNOME) &&
            validazioneEmail(email)) { //0
                EntityCliente cliente = new EntityCliente(email);

                if(cliente.effettuaCheckIn()) {
                    check = true;
                }
            }

        }catch(IllegalArgumentException | ArrayIndexOutOfBoundsException | UnknownHostException e) {
            // Qui poi si torna alla schermata di inserimento perché c'è stato qualche problema
            JOptionPane.showMessageDialog(null, e.getMessage(), "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        return check;
    }
        public boolean effettuaCheckIn() {
        boolean check = false;

        ClienteDAO clienteDAO = new ClienteDAO(this.email);

        PrenotazioneDAO prenotazioneDAO = new PrenotazioneDAO();

        if(clienteDAO.readClienteByEmail(this.email)) {//1
            EntityPrenotazione prenotazione = new EntityPrenotazione();

            prenotazione.setCodice(prenotazioneDAO.readCodicePrenotazioneCheckIn(this.email));

            if(this.getListaPrenotazioni().contains(prenotazione)) { //2
                JOptionPane.showMessageDialog(null, this.nome + " " + this.cognome +
                        " è presente nel sistema e ha effettuato una prenotazione", "DATI VERIFICATI",
JOptionPane.INFORMATION_MESSAGE);

                int index = this.getListaPrenotazioni().indexOf(prenotazione);

                LocalDate today = LocalDate.now();

                if(this.getListaPrenotazioni().get(index).getDataArrivo().isEqual(today)) { //3
                    // Set dei dati della prenotazione
                    prenotazione.setDataArrivo(this.getListaPrenotazioni().get(index).getDataArrivo());
```

```
                    prenotazione.setDataPartenza(this.getListaPrenotazioni().get(index).getDataPartenza());
                    prenotazione.setNumeroStanza(this.getListaPrenotazioni().get(index).getCamera().getNumeroCamera());
                    prenotazione.setAlbergo(this.getListaPrenotazioni().get(index).getAlbergo());
                    prenotazione.setCamera(this.getListaPrenotazioni().get(index).getCamera());
                    prenotazione.setCliente(this);

                    JOptionPane.showMessageDialog(null, prenotazione, "RIEPILOGO DATI PRENOTAZIONE",
JOptionPane.INFORMATION_MESSAGE);

                    CameraDAO cameraDAO = new CameraDAO(prenotazione.getCamera().getIdentificativo());

                    // Si aggiorna lo stato della camera ad occupata
                    prenotazione.getCamera().setStato("OCCUPATA"); //4

                    // Aggiorno la camera se questa esiste
                    if(cameraDAO.updateCamera(prenotazione.getCamera().getStato())) { //5
                        JOptionPane.showMessageDialog(null, "Check-in effettuato con successo", "OCCUPAZIONE CAMERA",
JOptionPane.INFORMATION_MESSAGE);

                        check = true;
                    } else {
                        JOptionPane.showMessageDialog(null, "Check-in non effettuato", "ERRORE", JOptionPane.ERROR_MESSAGE);
                    }
                } else {
                    JOptionPane.showMessageDialog(null, "La data di arrivo non corrisponde alla data odierna", "ERRORE",
JOptionPane.ERROR_MESSAGE);
                }
            }else {
                JOptionPane.showMessageDialog(null, "\nIl cliente non può effettuare alcun check-in!", "ERRORE",
JOptionPane.ERROR_MESSAGE);
            }

        } else {
            JOptionPane.showMessageDialog(null, "\nIl cliente non esiste!", "ERRORE", JOptionPane.ERROR_MESSAGE);
        }

        return check;
}
```



Numero Ciclomatico di McCabe:

11 (#nArchi) – 8 (#nNodi) + 2 = 4

Cammini Indipendenti:

1. **0**
2. **0-1**
3. **0-1-2-3-4-5**
4. **0-1-2**
5. **0-1-2-3**

7.2. Test funzionale

EFFETTUA PRENOTAZIONE

TEST CASE ID	DESCRIZIONE	CLASSI DI EQUIVALENZA COPERTE	PRE-CONDIZIONI	INPUT	OUTPUT ATTESI	POST-CONDIZIONI ATTESE	OUTPUT OTTENUTI	POST-CONDIZIONI OTTENUTE	ESITO (*FAIL*, *PASS*)
1	Tutti input validi	Nome valido, Cognome valido, E-mail valida, Numero di telefono valido, Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, E-mail: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}	Cliente prenotato	Sono salvati nel sistema i dati del cliente	Cliente prenotato	Sono salvati nel sistema i dati del cliente	PASS
2	Nome contente numeri/simboli	Nome contente numeri/simboli [ERROR], Cognome valido, E-mail valido, Numero di telefono, Indirizzo	Nessuna	{Nome: ”M4r10”, Cognome: “Rossi”, E-mail: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}	Messaggio di errore - Nome non valido	-	Messaggio di errore - Nome non valido!	-	PASS

		valido, Carta di credito valida							
3	Nome stringa di caratteri > 50	Nome stringa di caratteri > 50 [ERROR], Cognome valido, E-mail valido, Numero di telefono, Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mariooooooooooooooooooooooooooooooooooooooooooooooooooooooooooo”, Cognome: “Rossi”, E-mail: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}		-	Messaggio di errore - Nome non valido!	-	**PASS**
4	Cognome contente numeri/simboli	Nome valido, Cognome contente numeri/simboli [ERROR], E-mail valido, Numero di telefono, Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mario”, Cognome: “R0ss1”, Email: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}		-	Messaggio di errore - Cognome non valido!	-	**PASS**

5	Cognome stringa di caratteri > 50	Nome valido, Cognome stringa di caratteri > 50 [ERROR], E-mail valido, Numero di telefono, Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mario”, Cognome: “Rossiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii Iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii iiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiiii”, Email: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}		-	Messaggio di errore - Cognome non valido!	-	**PASS**
6	E-mail con dominio non valido	Nome valido, Cognome valido, E-mail con dominio non valido [ERROR], Numero di telefono, Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, Email: “mario_rossi@gm41l.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561237895674”}		-	Messaggio di errore - E-mail non valida!	-	**PASS**
7	E-mail non contiene una @	Nome valido, Cognome valido, E-mail non contiene una @	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, Email: “mario_rossi_gmail.com”, Numero di Telefono:		-	Messaggio di errore - E-mail non valida!	-	**PASS**

		[ERROR], Numero di telefono, Indirizzo valido, Carta di credito valida		“081872701”, Carta di credito: “8904561237895674”}					
8	Numero di telefono stringa di caratteri di lunghezza > 20	Nome valido, Cognome valido, E-mail valido, Numero di telefono stringa di caratteri di lunghezza > 20 [ERROR], Indirizzo valido, Carta di credito valida	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, Email: “mario_rossi@gmail.com”, Numero di Telefono: “08187270100000000000000”, Carta di credito: “8904561237895674”}		-	Messaggio di errore - Numero di telefono non valido	-	**PASS**
9	Carta di credito stringa di caratteri di lunghezza > 16	Nome valido, Cognome valido, E-mail valido, Numero di telefono, Indirizzo valido, Carta	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, Email: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “8904561290378956 74”}		-	Messaggio di errore - Carta di credito non valida	-	**PASS**

		di credito stringa di caratteri di lunghezza > 16 [ERROR]							
10	Carta di credito stringa di caratteri di lunghezza < 16	Nome valido, Cognome valido, Email valido, Numero di telefono, Indirizzo valido, Carta di credito stringa di caratteri di lunghezza < 16 [ERROR]	Nessuna	{Nome: ”Mario”, Cognome: “Rossi”, Email: “mario_rossi@gmail.com”, Numero di Telefono: “081872701”, Carta di credito: “890895674”}		-	Messaggio di errore - Carta di credito non valida	-	**PASS**