

# **PRESENTAZIONE PROGETTO**



# **CLICKFLY**

**PRESENTED BY:**  
**CHIERCHIA GIUSY**  
**PENNARELLA FABIO**

# Introduzione al Progetto – Problem Statement



ClickFly è una piattaforma web per la prenotazione online di voli aerei, progettata per semplificare e migliorare l'esperienza di viaggio degli utenti.

Il sistema si propone come un punto di riferimento affidabile per la ricerca, la prenotazione e la gestione dei voli, offrendo informazioni aggiornate in tempo reale su disponibilità, prezzi e condizioni di viaggio.

In un contesto caratterizzato da un'elevata dinamicità del settore dei trasporti aerei e da una crescente domanda di servizi digitali efficienti, ClickFly fornisce un'interfaccia intuitiva e accessibile che consente agli utenti di pianificare i propri spostamenti in modo rapido e autonomo.

La piattaforma supporta inoltre le attività amministrative, permettendo la gestione centralizzata di voli e prenotazioni, garantendo coerenza, affidabilità dei dati e sicurezza delle transazioni.

# Scopo del Sistema



Fornire una piattaforma web per la prenotazione di voli aerei

Consentire la consultazione di voli aggiornati in tempo reale

Permettere la prenotazione e la gestione autonoma delle prenotazioni

Supportare le attività di utenti e amministratori

Utenti coinvolti

Utente guest: consultazione dei voli

Utente registrato: prenotazione e gestione del profilo

Amministratore: gestione di voli e prenotazioni

# Scenari Analizzati

---

Registrazione nuovo utente

01



Accesso alla piattaforma

02



Ricerca e aggiunta di un volo

03



04

Gestione Carrello



05

Ricarica portafoglio virtuale



06

Acquisto volo,  
Visualizzazione  
Prenotazioni e  
Logout

## **Registrazione di un nuovo utente**

Un utente che visita per la prima volta la piattaforma ClickFly desidera prenotare un volo.

Per poter utilizzare i servizi del sito, accede alla pagina di registrazione e inserisce i propri dati personali (nome, cognome, email, password, numero di telefono).

Una volta completata la registrazione, il sistema salva i dati nel database e consente all'utente di effettuare il login e accedere alla propria area personale.



## **Accesso alla piattaforma (Login)**

Un utente già registrato accede alla piattaforma inserendo email e password.

Il sistema verifica la correttezza delle credenziali e, in caso di esito positivo, consente l'accesso all'area personale, dove l'utente può visualizzare il proprio profilo, il saldo del portafoglio, il carrello e le prenotazioni effettuate.



## **Ricerca di un volo**

Un utente autenticato desidera prenotare un volo.

Accede alla pagina di ricerca e inserisce i parametri desiderati (città di partenza, città di arrivo, data del viaggio).

Il sistema interroga il database e mostra l'elenco dei voli disponibili compatibili con i criteri di ricerca, visualizzando per ciascuno prezzo, compagnia e orari.

## **Aggiunta di un volo al carrello**

Dopo aver visualizzato i risultati della ricerca, l'utente seleziona un volo di suo interesse e indica il numero di passeggeri.

Il sistema aggiunge il volo al carrello dell'utente, che può continuare a cercare altri voli oppure visualizzare il carrello per procedere all'acquisto.



## Gestione del carrello

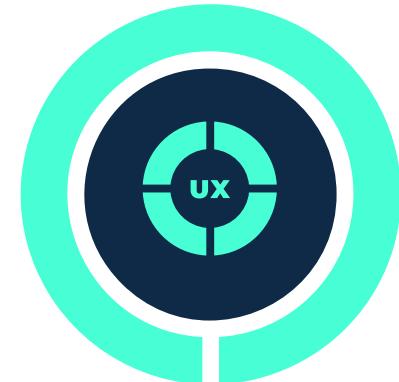
L'utente accede alla pagina del carrello per rivedere i voli selezionati.

Il sistema mostra l'elenco dei voli presenti nel carrello con il relativo prezzo, quantità e il totale complessivo.

## Ricarica del portafoglio virtuale

Prima di effettuare un acquisto, l'utente può accedere alla propria area personale e ricaricare il portafoglio virtuale inserendo l'importo desiderato.

Il sistema aggiorna il saldo associato all'utente, che potrà essere utilizzato per effettuare i pagamenti sulla piattaforma.



## Acquisto dei voli (Checkout)

L'utente, dopo aver verificato il contenuto del carrello, procede al checkout. Il sistema controlla che il saldo del portafoglio sia sufficiente a coprire il costo totale dell'ordine. Se il saldo è sufficiente, il sistema: scala l'importo dal portafoglio dell'utente genera una o più prenotazioni nel sistema svuota il carrello mostra un messaggio di conferma dell'acquisto Se il saldo non è sufficiente, il sistema: notifica l'utente invita a ricaricare il portafoglio



## Visualizzazione delle prenotazioni

L'utente accede alla sezione "Le mie prenotazioni" dalla propria area personale. Il sistema mostra l'elenco di tutte le prenotazioni effettuate, con:

- codice prenotazione
- tratta del volo
- compagnia
- data di prenotazione
- numero di passeggeri
- prezzo totale

## Logout

L'utente può in qualsiasi momento effettuare il logout dalla piattaforma. Il sistema invalida la sessione e riporta l'utente alla pagina principale.



# Functional Requirements

## Registrazione utente

Il sistema deve permettere a un utente non registrato di creare un nuovo account inserendo i propri dati personali (nome, cognome, email, password, telefono).

## Autenticazione utente (Login)

Il sistema deve permettere a un utente registrato di autenticarsi tramite email e password.

## Logout

Il sistema deve permettere all'utente autenticato di terminare la propria sessione in qualsiasi momento.

## Visualizzazione profilo utente

Il sistema deve permettere all'utente autenticato di visualizzare le proprie informazioni personali e il saldo del portafoglio.

## Ricarica del portafoglio virtuale

Il sistema deve permettere all'utente di ricaricare il proprio portafoglio virtuale specificando un importo.

## **Ricerca voli**

Il sistema deve permettere all'utente di cercare voli specificando almeno città di partenza, città di arrivo e data.

## **Visualizzazione risultati di ricerca**

Il sistema deve mostrare all'utente l'elenco dei voli disponibili che soddisfano i criteri di ricerca.

## **Visualizzazione dettagli volo**

Il sistema deve permettere all'utente di visualizzare i dettagli di un volo (compagnia, orari, prezzo, posti disponibili).

## **Aggiunta volo al carrello**

Il sistema deve permettere all'utente di aggiungere uno o più voli al carrello specificando il numero di passeggeri.

## **Visualizzazione carrello**

Il sistema deve permettere all'utente di visualizzare il contenuto del proprio carrello.



## **Modifica carrello**

Il sistema deve permettere all'utente di rimuovere uno o più voli dal carrello.

## **Calcolo del totale del carrello**

Il sistema deve calcolare e mostrare automaticamente il costo totale dei voli presenti nel carrello.

## **Checkout**

Il sistema deve permettere all'utente di procedere all'acquisto dei voli presenti nel carrello.

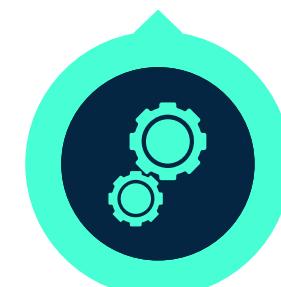
## **Verifica disponibilità fondi**

Durante il checkout, il sistema deve verificare che il saldo del portafoglio dell'utente sia sufficiente a coprire l'importo totale dell'acquisto.

## **Conferma acquisto**

Se il saldo è sufficiente, il sistema deve:

- scalare l'importo dal portafoglio dell'utente
- generare una o più prenotazioni
- svuotare il carrello



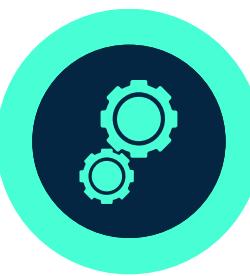
## Gestione errore fondi insufficienti

Se il saldo non è sufficiente, il sistema deve:

bloccare l'operazione

mostrare un messaggio di errore

invitare l'utente a ricaricare il portafoglio



## Visualizzazione prenotazioni

Il sistema deve permettere all'utente di visualizzare lo storico delle proprie prenotazioni.



## Visualizzazione dettagli prenotazione

Il sistema deve mostrare per ogni prenotazione:

- codice prenotazione
- tratta del volo
- compagnia
- data prenotazione
- numero passeggeri
- prezzo totale



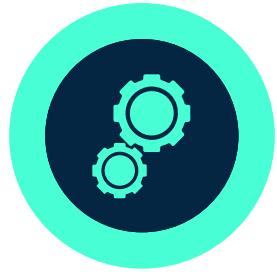
## Persistenza dei dati

Il sistema deve salvare in modo persistente su database:

utenti - voli - carrello - prenotazioni - portafoglio



# Non Functional Requirements



## Prestazioni

Il sistema deve garantire tempi di risposta rapidi per tutte le principali operazioni utente.

- Le pagine principali (ricerca voli, carrello, area personale) devono caricarsi entro 2 secondi in condizioni di carico normale.
- 
- Le operazioni critiche (login, aggiunta al carrello, checkout) devono essere complete entro 3 secondi.
- 
- Gli aggiornamenti del carrello e delle prenotazioni devono essere visibili immediatamente dopo l'operazione.

## Sicurezza

Il sistema deve garantire la protezione dei dati sensibili degli utenti.

- Le password devono essere memorizzate esclusivamente in forma hashata.
- Le sessioni utente devono essere gestite in modo sicuro tramite sessioni HTTP.
- Le pagine riservate (area personale, carrello, prenotazioni) devono essere accessibili solo agli utenti autenticati.
- Tutte le operazioni critiche (checkout, rimozione carrello, visualizzazione prenotazioni) devono verificare l'identità dell'utente.

## **Usabilità**

Il sistema deve essere semplice e intuitivo da usare anche per utenti non esperti.  
L'interfaccia deve essere:

- Chiara
- Intuitiva
- Coerente in tutte le pagine

Il sito deve essere responsive, cioè utilizzabile sia da desktop che da dispositivi mobili.

Le operazioni principali (ricerca voli, acquisto, gestione account) devono essere eseguibili in pochi passaggi chiari.

## **Scalabilità**

Il sistema deve poter crescere nel tempo senza richiedere modifiche strutturali.

Il sistema deve poter gestire:

- Un aumento del numero di utenti
- Un aumento del numero di voli
- Un aumento del numero di prenotazioni
- L'architettura basata su database relazionale e servlet consente l'estensione futura a:
  - Più utenti simultanei
  - Più funzionalità
  - Più moduli applicativi

# Non Functional Requirements

## Affidabilità

Il sistema deve garantire coerenza e correttezza dei dati.

Le operazioni critiche (checkout, creazione prenotazioni) devono essere:

- Atomiche
- Coerenti

In caso di errore:

- Il sistema deve mostrare messaggi chiari all'utente
- Non devono verificarsi stati inconsistenti (es: soldi scalati ma prenotazione non creata)
- I dati nel database devono rimanere sempre consistenti.

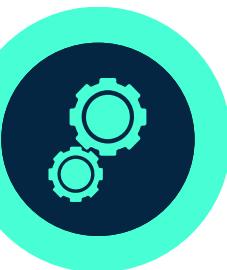


## Manutenibilità

Il sistema deve essere facilmente manutenibile ed estendibile.

Il codice è organizzato secondo il pattern:

- Model
- DAO
- Servlet
- JSP



## Portabilità

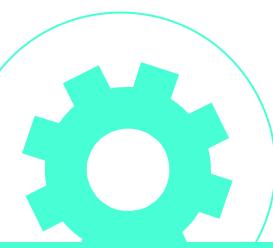
Il sistema deve essere eseguibile su qualsiasi ambiente che supporti:

- Java
- Servlet Container (es: Tomcat)
- MySQL



L'accesso avviene tramite browser web senza necessità di installazione.



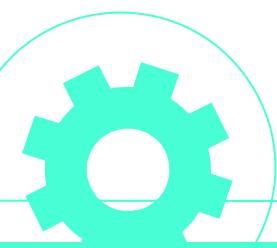


## Tecnologie utilizzate

- Java per la logica lato server
- Servlet/JSP per la gestione delle richieste e delle viste
- MySQL per la persistenza dei dati
- Bootstrap per l'interfaccia grafica responsive
- Maven per la gestione delle dipendenze
- Apache Tomcat come application server

## Deliverables

- Problem Statement
- Requirements Analysis Document (RAD)



- System Design Document (SDD)
- Object Design Document (ODD)
- Test Plan
- Test Case Specification

### **Deadlines**

Il progetto ClickFly è stato sviluppato seguendo una pianificazione a fasi, con consegne intermedie della documentazione e una consegna finale del sistema completo e funzionante entro le scadenze previste dal corso.



# Requirements Analysis

# Scenari

## **Registrazione di un nuovo utente**

Un utente che visita per la prima volta la piattaforma ClickFly desidera prenotare un volo.

Per poter utilizzare i servizi del sito, accede alla pagina di registrazione e inserisce i propri dati personali (nome, cognome, email, password, numero di telefono).

Una volta completata la registrazione, il sistema salva i dati nel database e consente all'utente di effettuare il login e accedere alla propria area personale.

## **Accesso alla piattaforma (Login)**

Un utente già registrato accede alla piattaforma inserendo email e password.

Il sistema verifica la correttezza delle credenziali e, in caso di esito positivo, consente l'accesso all'area personale, dove l'utente può visualizzare il proprio profilo, il saldo del portafoglio, il carrello e le prenotazioni effettuate.

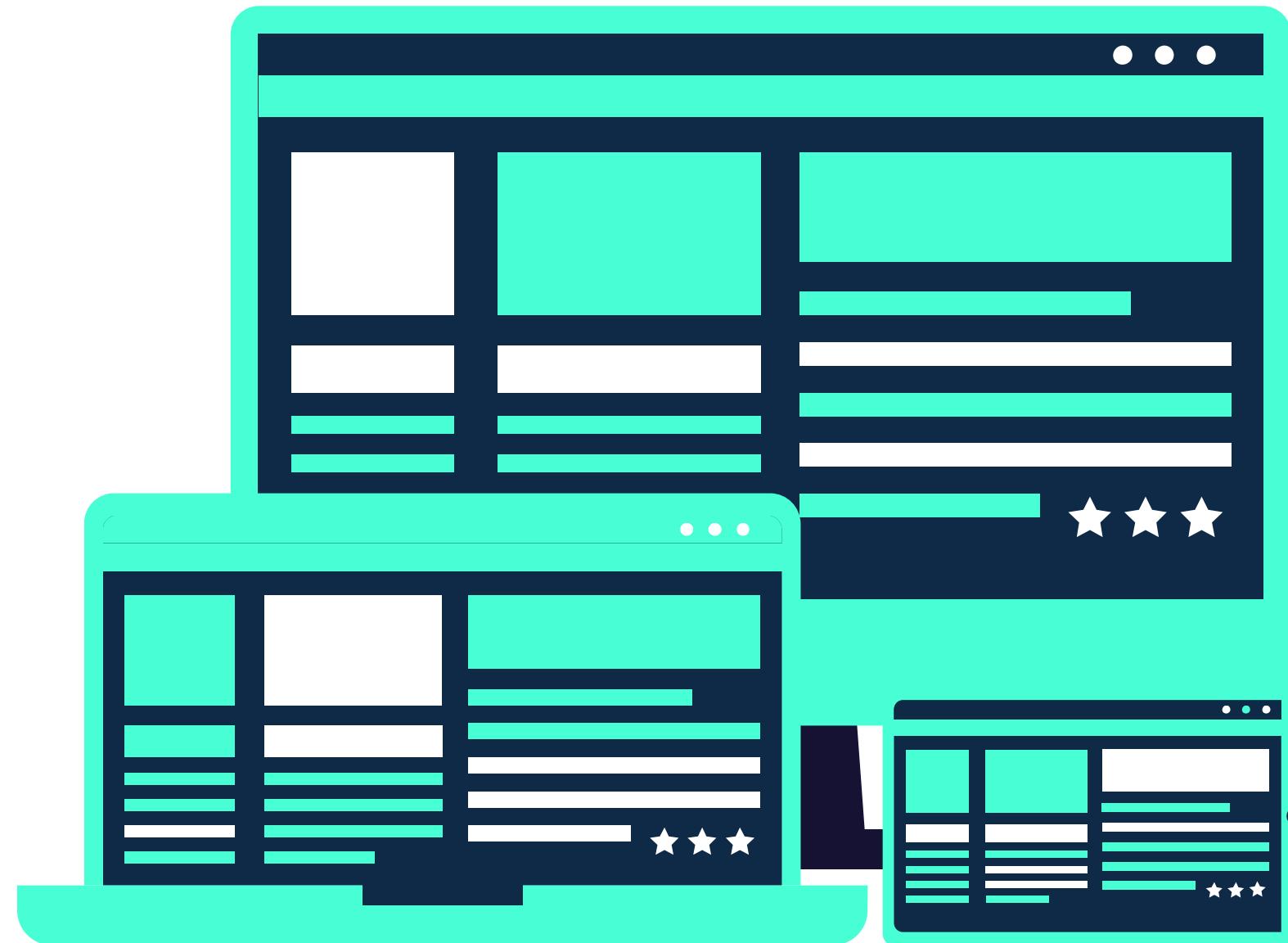
## **Ricerca di un volo**

Un utente autenticato desidera prenotare un volo.

Accede alla pagina di ricerca e inserisce i parametri desiderati (città di partenza, città di arrivo, data del viaggio).

Il sistema interroga il database e mostra l'elenco dei voli disponibili compatibili con i criteri di ricerca, visualizzando per ciascuno prezzo, compagnia e orari.





## Scenario 4: Aggiunta di un volo al carrello

Dopo aver visualizzato i risultati della ricerca, l'utente seleziona un volo di suo interesse e indica il numero di passeggeri.

Il sistema aggiunge il volo al carrello dell'utente, che può continuare a cercare altri voli oppure visualizzare il carrello per procedere all'acquisto.

## Scenario 5: Gestione del carrello

L'utente accede alla pagina del carrello per rivedere i voli selezionati.

Il sistema mostra l'elenco dei voli presenti nel carrello con il relativo prezzo, quantità e il totale complessivo.

L'utente può rimuovere uno o più voli oppure procedere al checkout.

## **Scenario 6: Ricarica del portafoglio virtuale**

Prima di effettuare un acquisto, l'utente può accedere alla propria area personale e ricaricare il portafoglio virtuale inserendo l'importo desiderato. Il sistema aggiorna il saldo associato all'utente, che potrà essere utilizzato per effettuare i pagamenti sulla piattaforma.

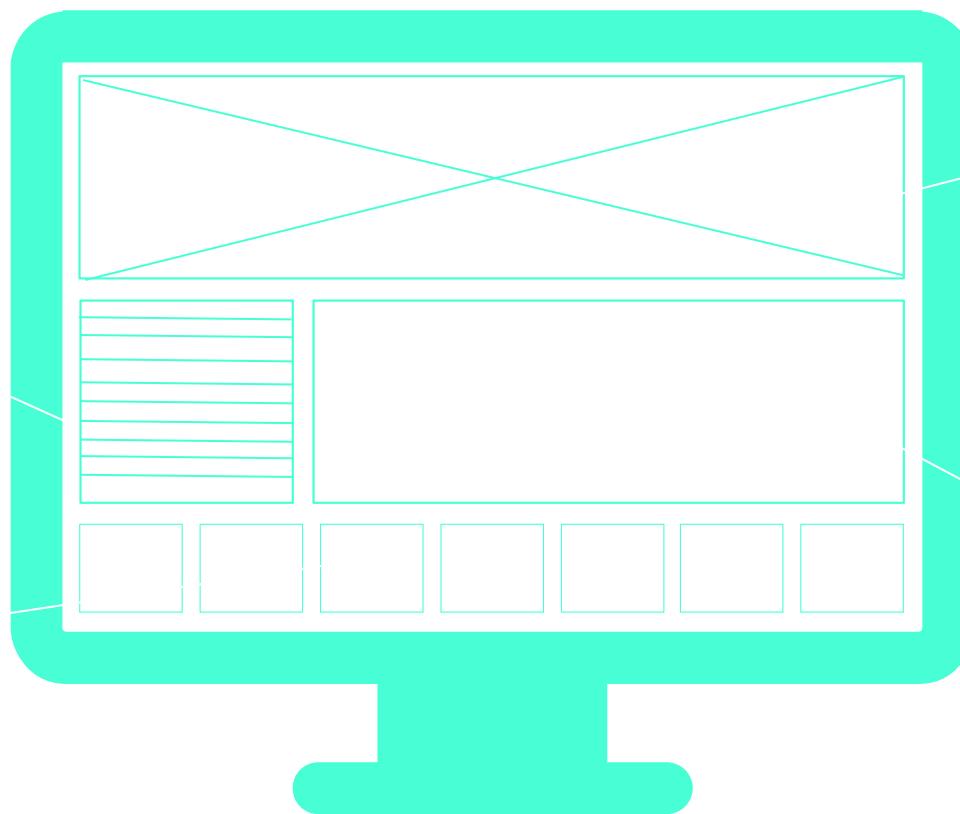
## **Scenario 7: Acquisto dei voli (Checkout)**

L'utente, dopo aver verificato il contenuto del carrello, procede al checkout. Il sistema controlla che il saldo del portafoglio sia sufficiente a coprire il costo totale dell'ordine.

Se il saldo è sufficiente, il sistema:

- scala l'importo dal portafoglio dell'utente
- genera una o più prenotazioni nel sistema
- svuota il carrello
- mostra un messaggio di conferma dell'acquisto
- Se il saldo non è sufficiente, il sistema:
  - notifica l'utente
  - invita a ricaricare il portafoglio

# **Prenotazione di un Volo**



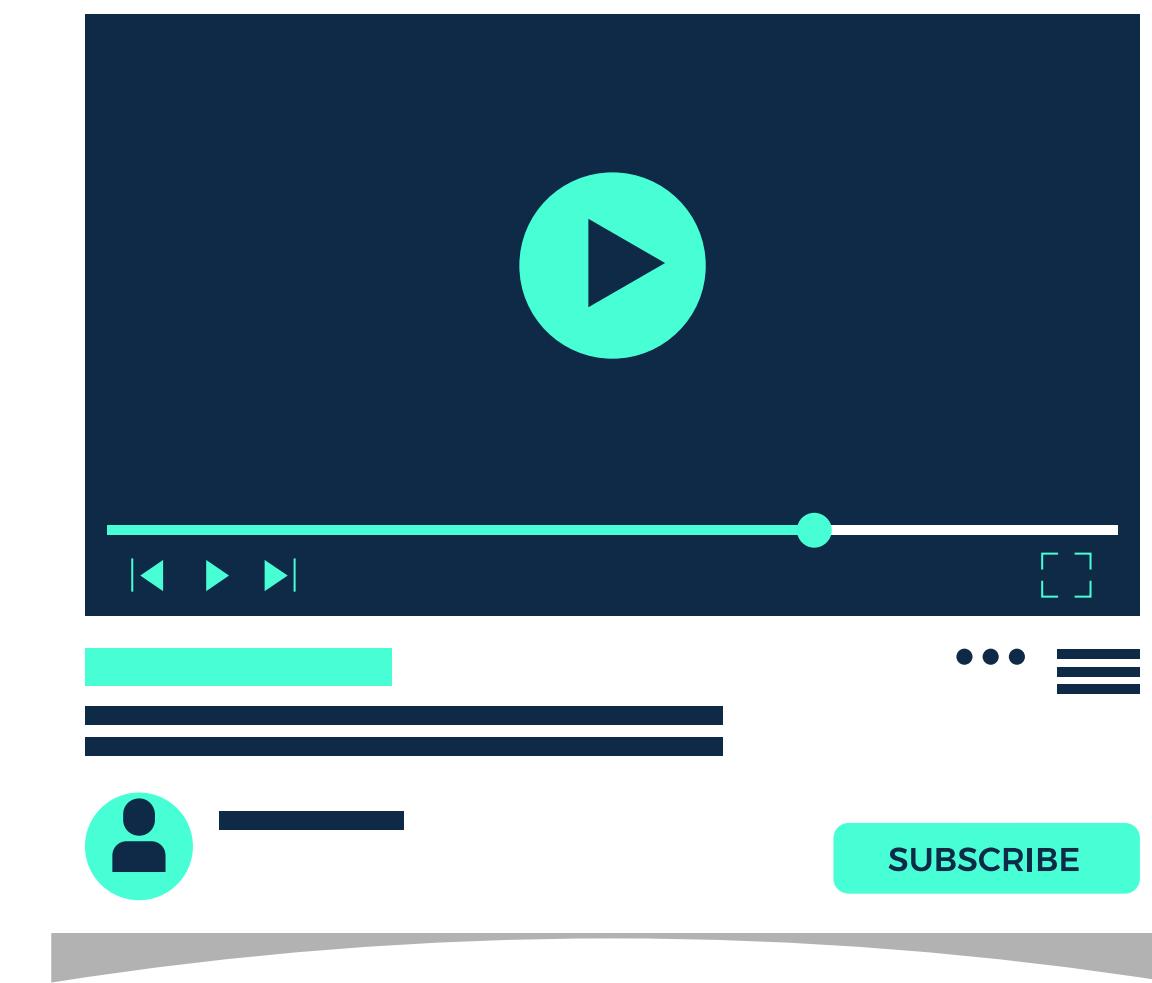
## Scenario 8: Visualizzazione delle prenotazioni

L'utente accede alla sezione "Le mie prenotazioni" dalla propria area personale. Il sistema mostra l'elenco di tutte le prenotazioni effettuate, con:

- codice prenotazione
- tratta del volo
- compagnia
- data di prenotazione
- numero di passeggeri
- prezzo totale

## Scenario 9: Logout

L'utente può in qualsiasi momento effettuare il logout dalla piattaforma. Il sistema invalida la sessione e riporta l'utente alla pagina principale.



# Object Model e Dynamic Model

## Object Model – Class Diagram

### Classi principali:

- Utente
- Volo
- Prenotazione
- Pagamento
- PortafoglioVirtuale
- Amministratore



### Relazioni:

Un Utente può effettuare più Prenotazioni

Ogni Prenotazione è associata a un Volo

Ogni Prenotazione prevede un Pagamento

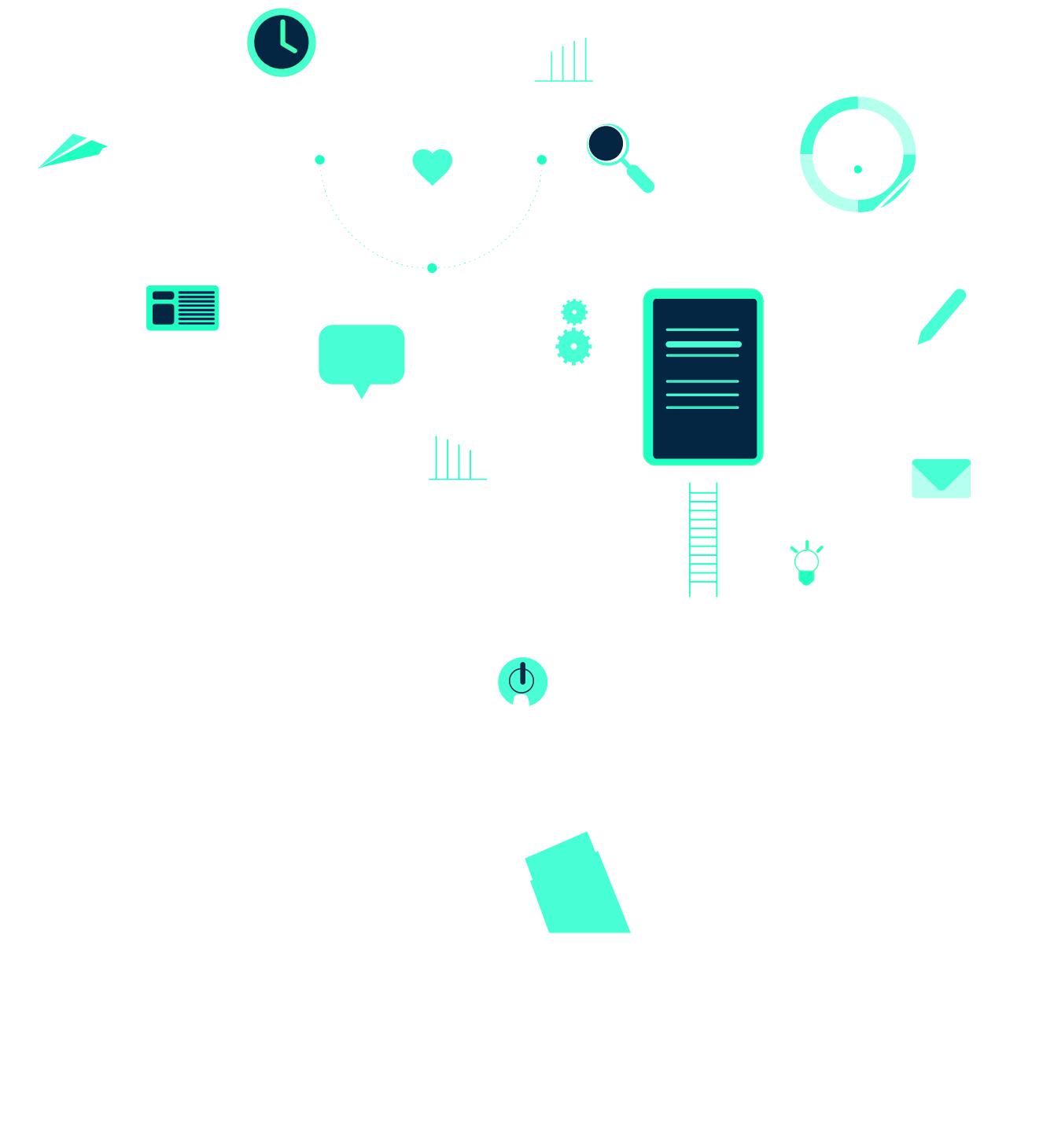
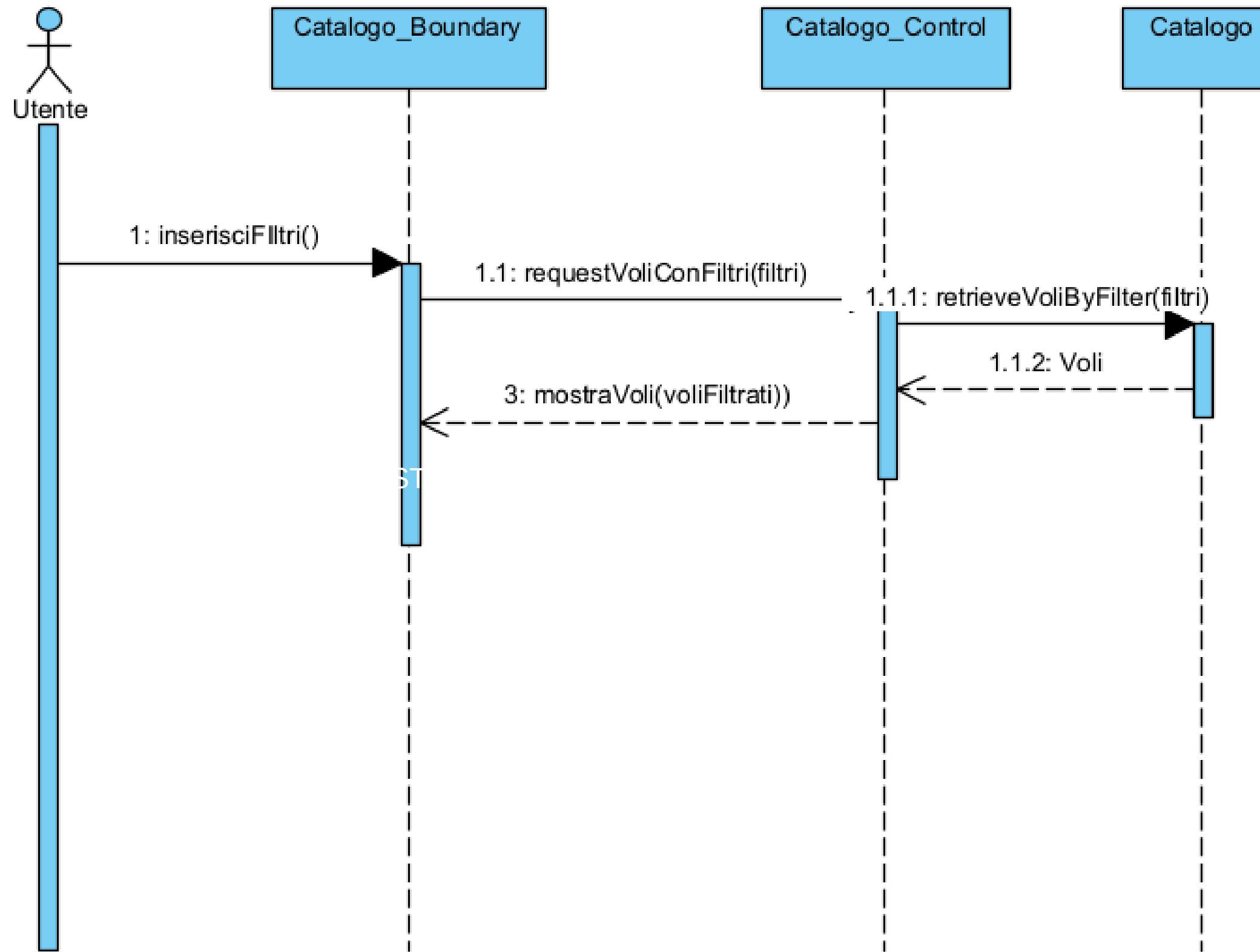
L'Amministratore gestisce Voli e Prenotazioni

## Dynamic Model – Sequence Diagram

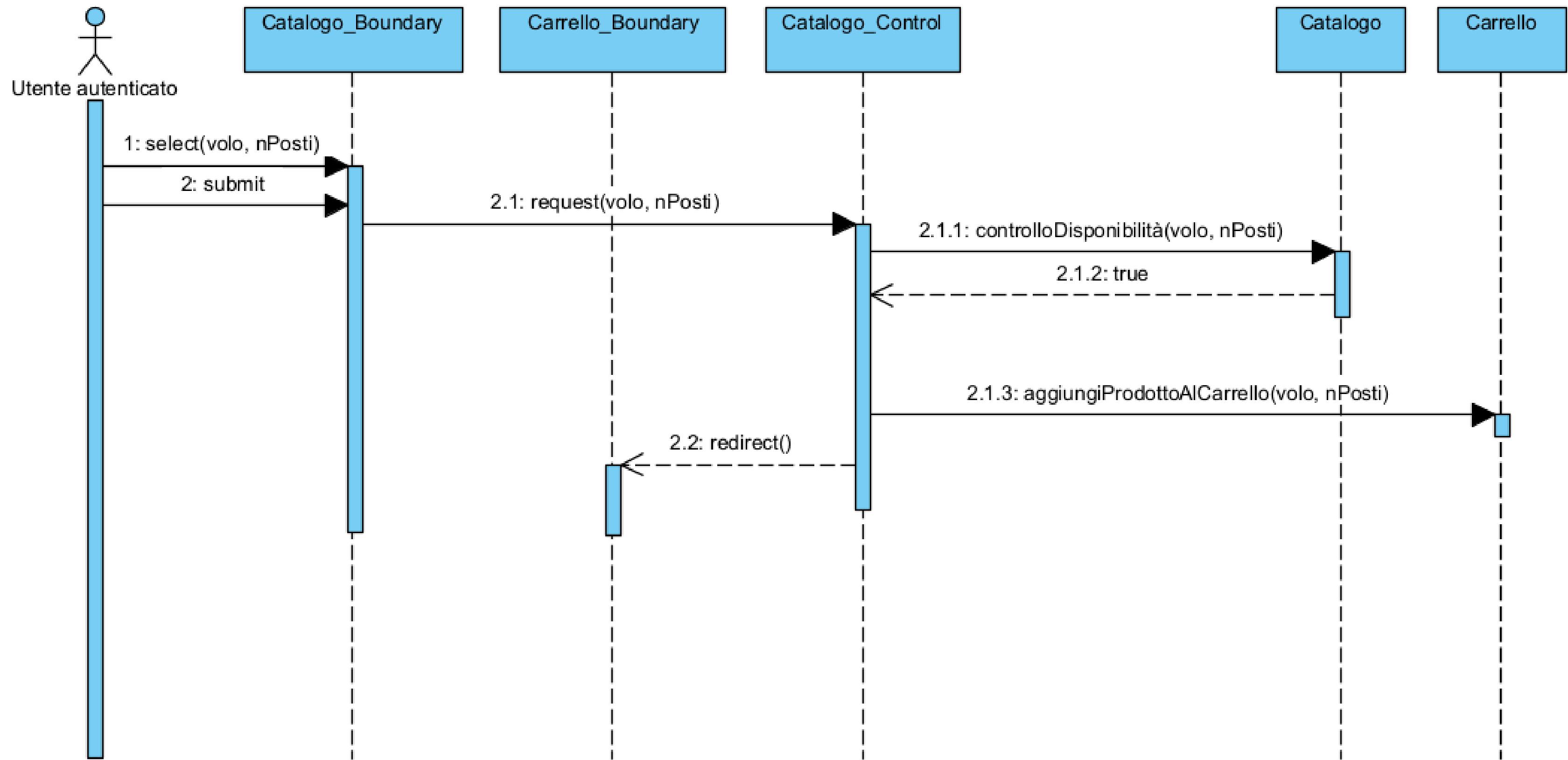
- Inserimento criteri di ricerca
- Interrogazione del database
- Visualizzazione voli disponibili
- Selezione volo e pagamento
- Conferma della prenotazione



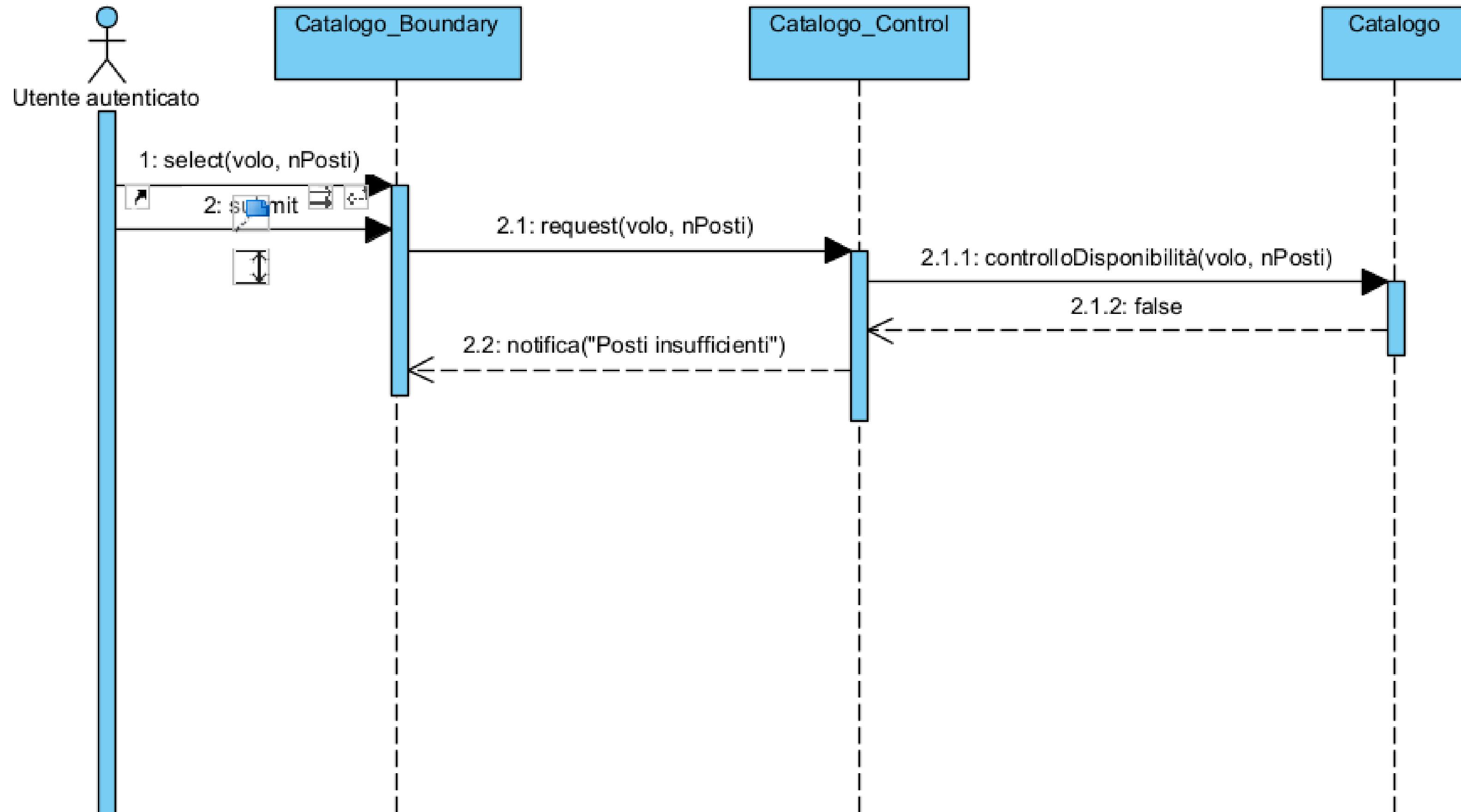
# Ricerca Volo



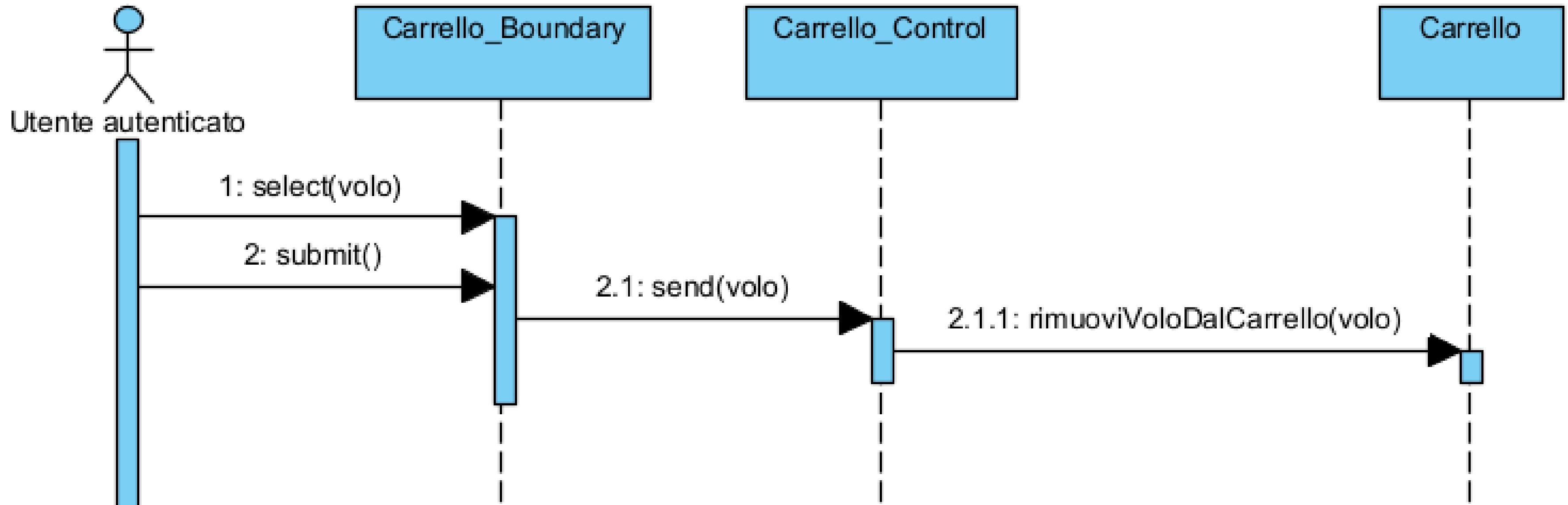
# Aggiunta volo al carrello



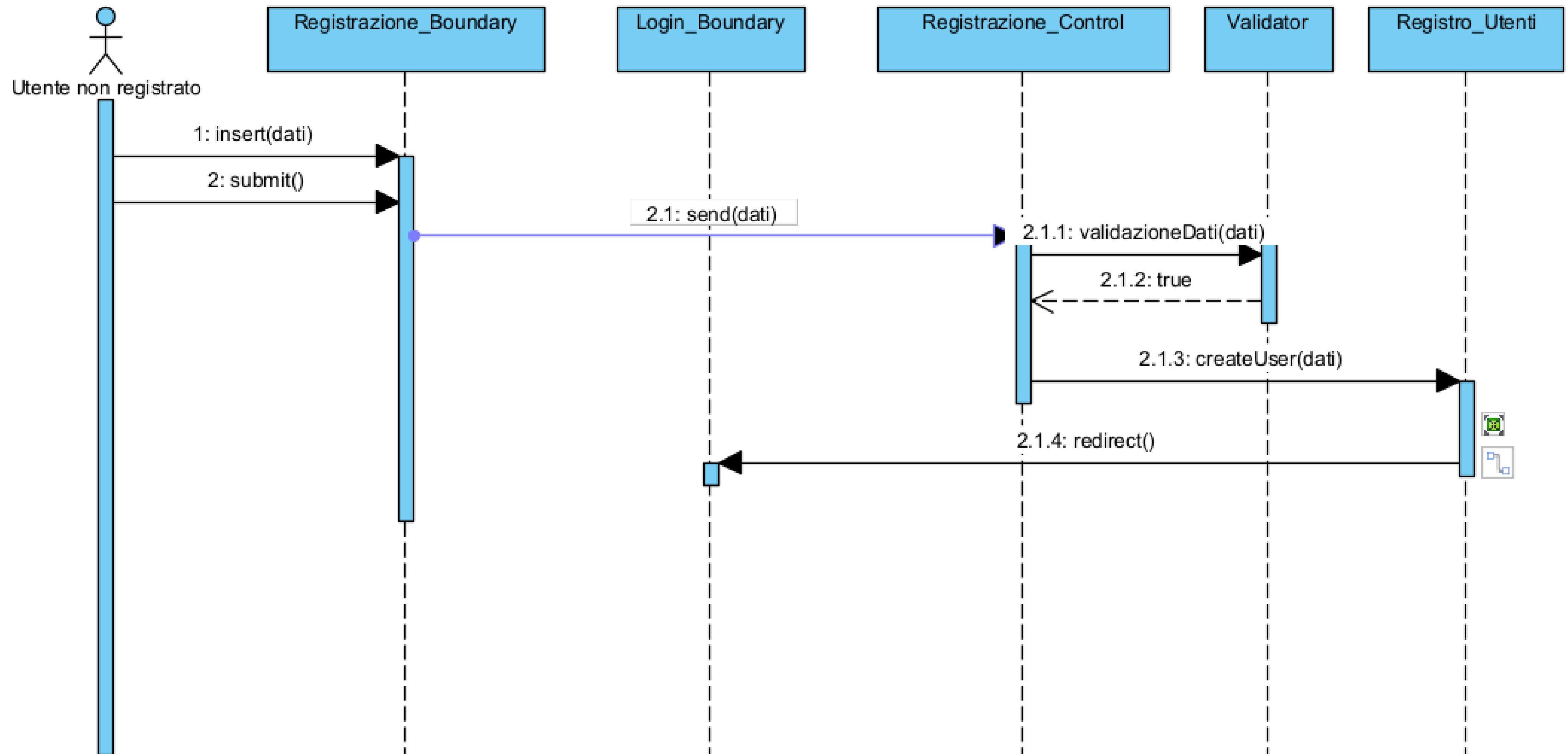
# Aggiunta volo al carrello - Errore



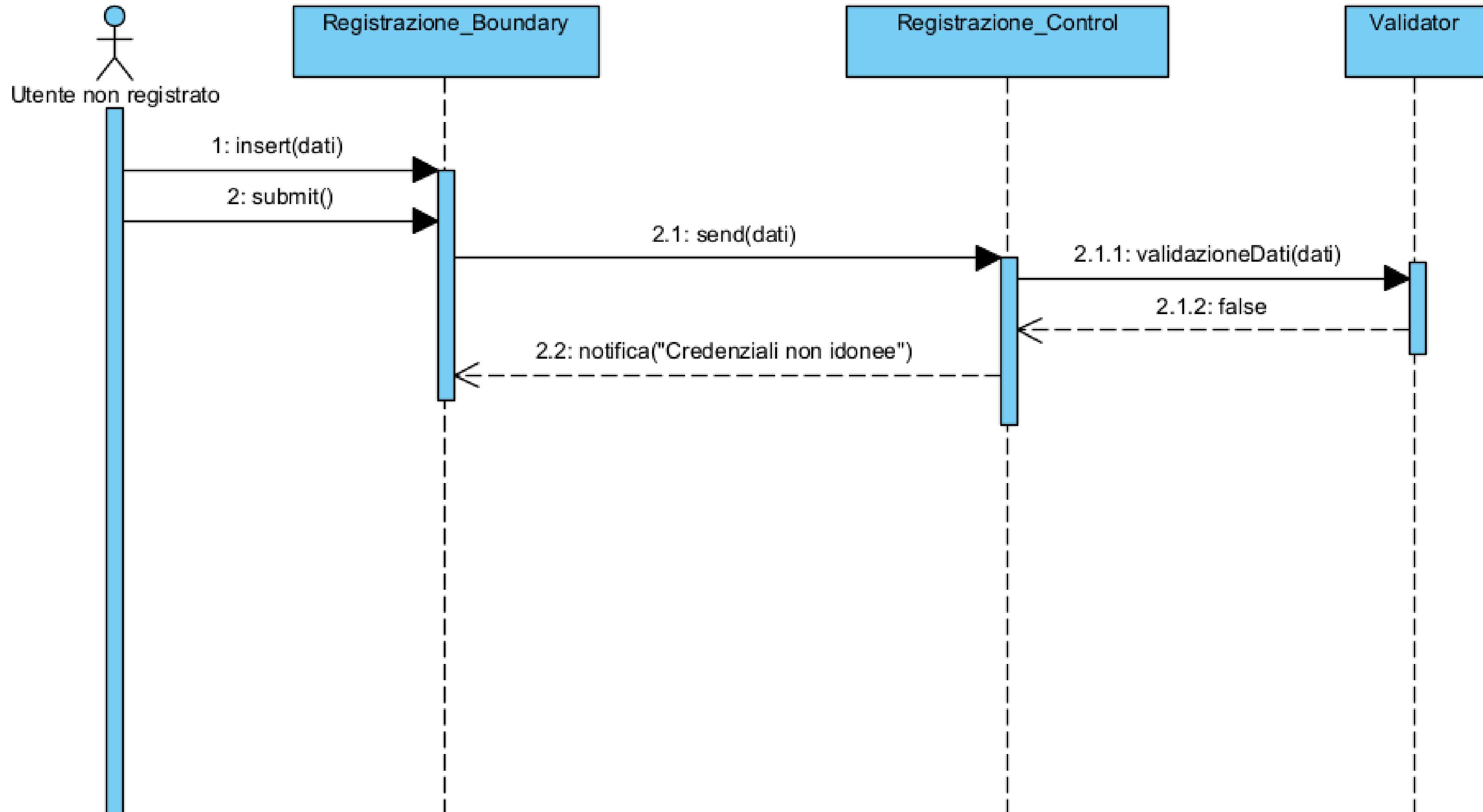
# Rimozione di un volo dal carrello

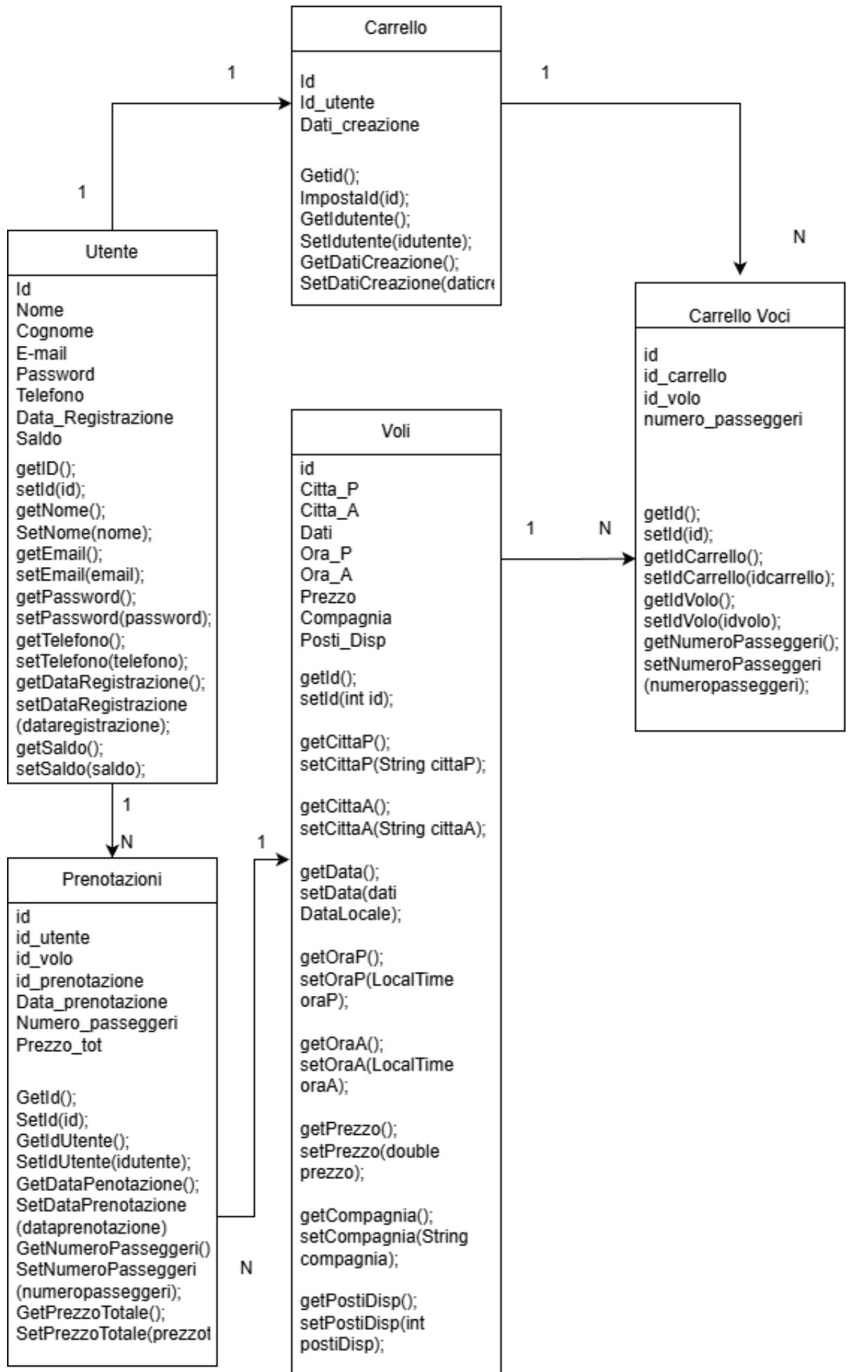


# Registrazione



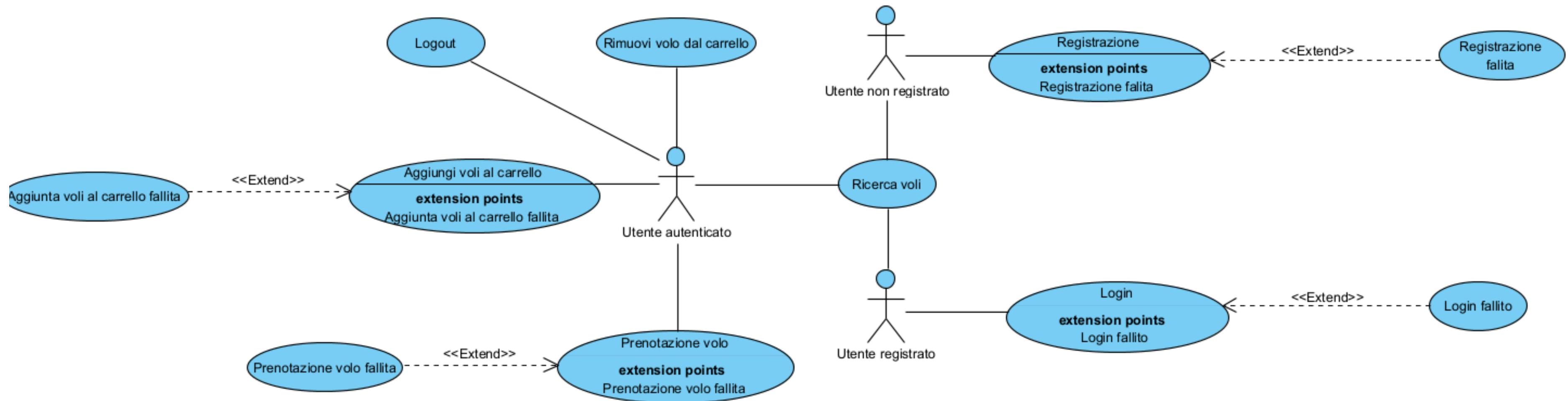
# Registrazione fallita





# Modello a Oggetti

# Diagramma dei casi d'uso





# System Design

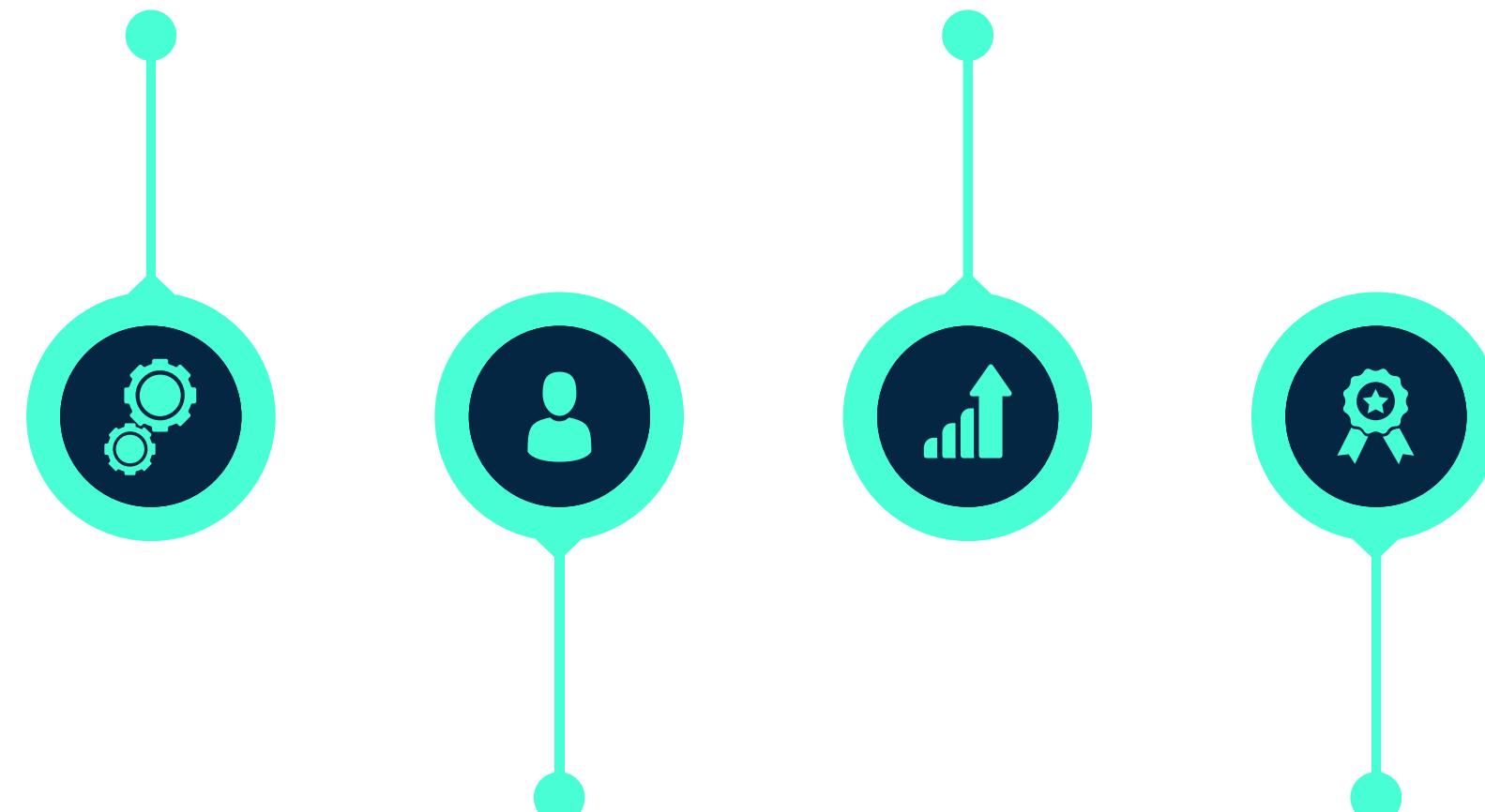
# System Design

---

In questa fase vengono descritti i dettagli tecnici del design del sistema ClickFly.

La sezione illustra l'architettura software adottata, gli obiettivi di progetto e la suddivisione del sistema in sottosistemi, evidenziando le scelte architettoniche effettuate per garantire scalabilità, sicurezza e manutenibilità.

Vengono inoltre definite la mappatura hardware/software, il controllo degli accessi e le boundary conditions, al fine di assicurare il corretto funzionamento del sistema anche in presenza di errori o situazioni anomale.



## Usabilità

Interfaccia intuitiva e facilmente navigabile

Ricerca dei voli semplificata tramite filtri (destinazione, data, prezzo, classe)

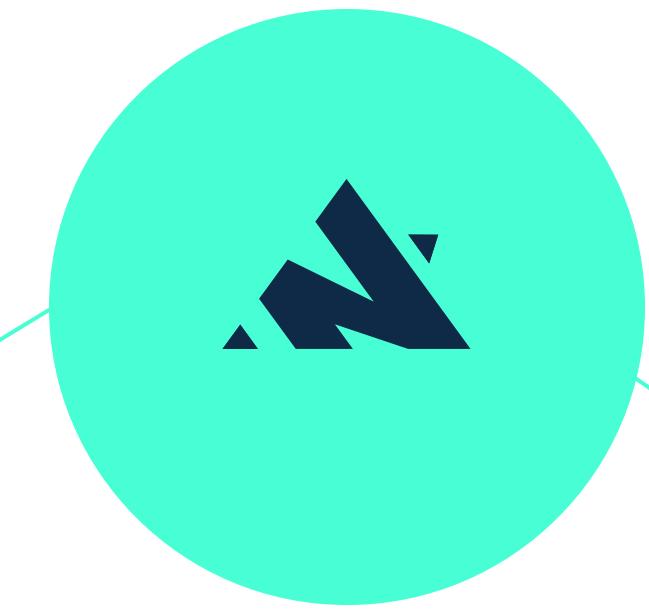
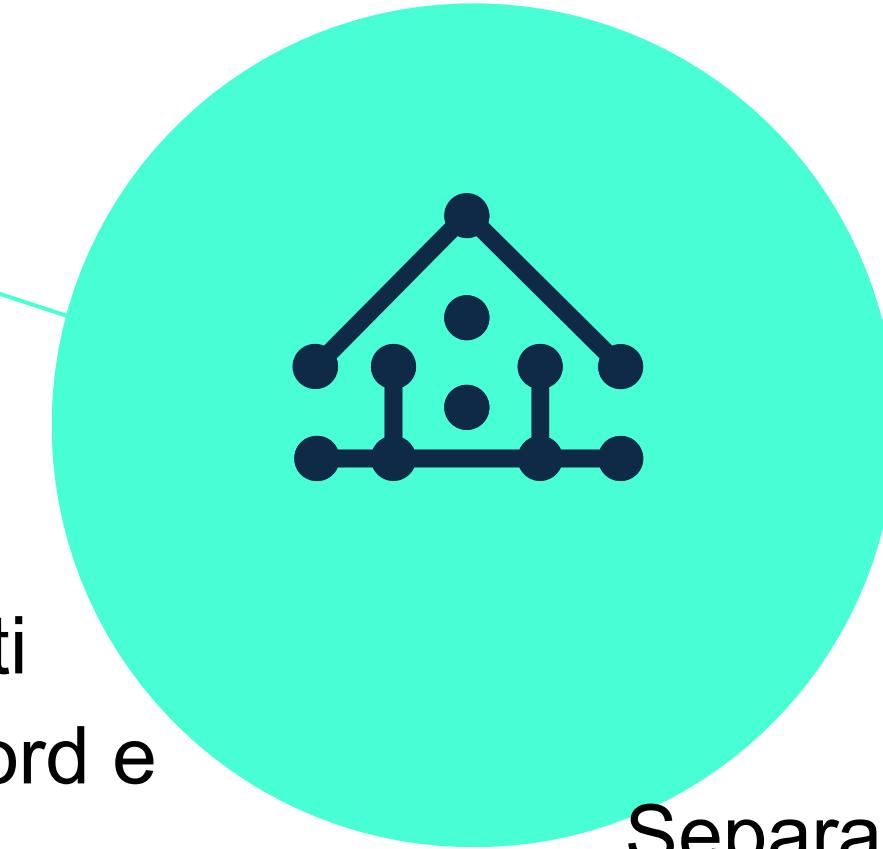
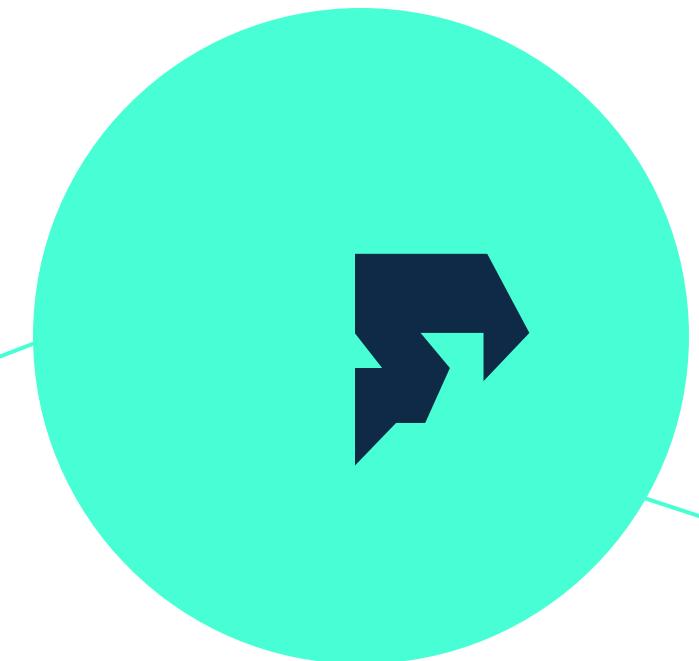
Design responsive per dispositivi desktop e mobile

## Affidabilità

Gestione coerente delle prenotazioni e dei pagamenti

Conferme automatiche via email

Prevenzione di prenotazioni inconsistenti



## Sicurezza

Autenticazione sicura degli utenti

Protezione dei dati sensibili (password e informazioni di pagamento)

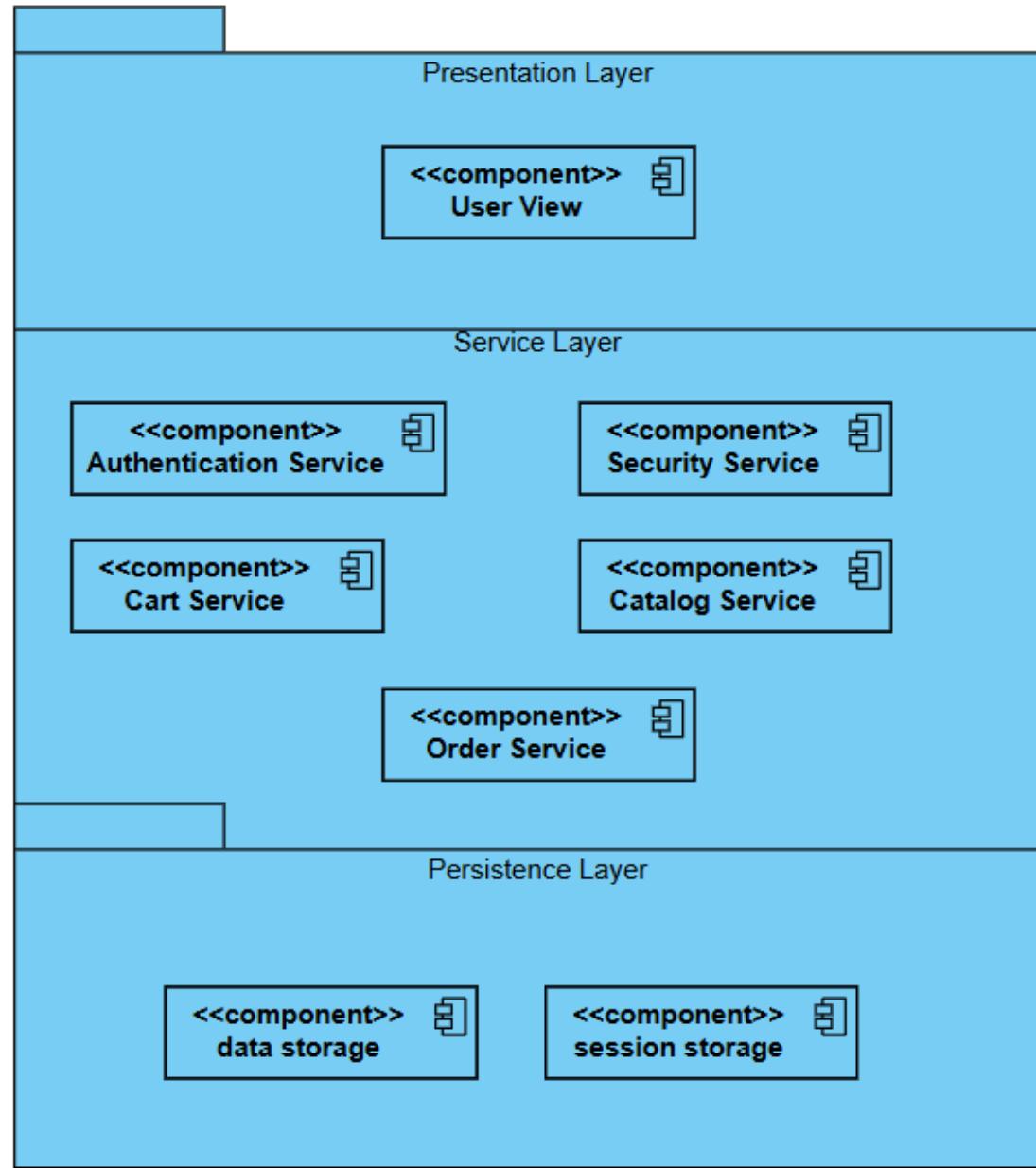
Comunicazioni cifrate tramite protocollo HTTPS

## Manutenibilità ed Estendibilità

Architettura a livelli

Separazione tra logica di business e persistenza  
Facilità di integrazione di nuove funzionalità future

# Architettura Software



L'architettura software di ClickFly è strutturata secondo un modello multilayer, che consente una chiara separazione delle responsabilità e facilita la manutenzione e l'evoluzione del sistema.

**Layer principali:**  
Presentation Layer  
Service Layer  
Persistence Layer

Questa suddivisione permette una gestione modulare delle funzionalità e riduce l'accoppiamento tra i componenti del sistema.

# Presentation Layer

Il Presentation Layer è responsabile dell'interazione tra il sistema e l'utente finale.

## **Componenti principali**

Pagine web sviluppate con JSP, HTML, CSS e Bootstrap

## **Componenti grafici**

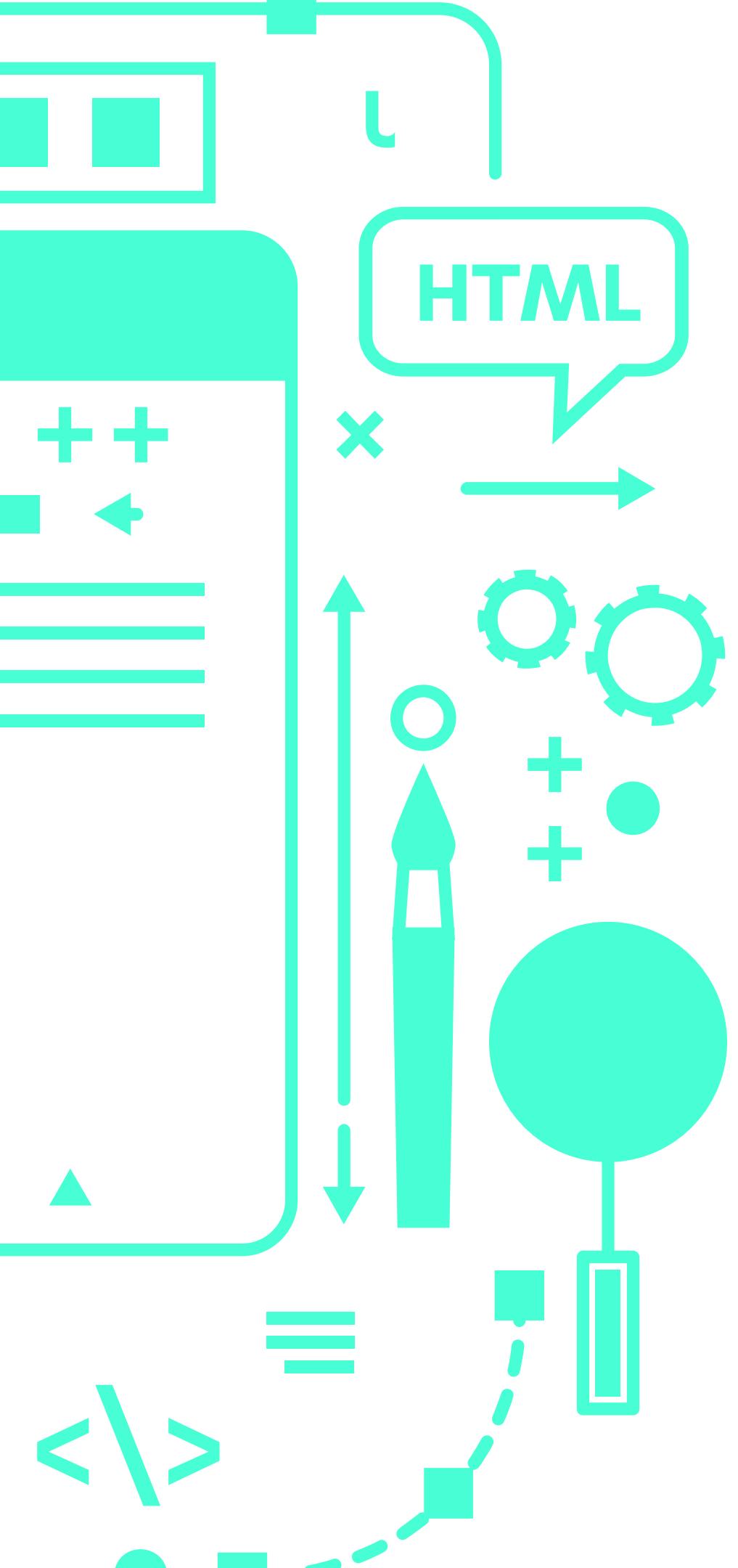
per la visualizzazione di voli, carrello e prenotazioni  
Form per login, registrazione, pagamento e gestione del profilo

## **Caratteristiche chiave**

Interfaccia responsive

Validazione lato client tramite JavaScript

Navigazione semplice e coerente tra le pagine



# Service Layer

Il Service Layer rappresenta il cuore della logica applicativa e coordina le operazioni tra Presentation Layer e Persistence Layer.

Componenti principali

**Authentication Service:** login, logout, gestione delle sessioni e dei ruoli

**Flight Service:** ricerca voli e gestione della disponibilità dei posti

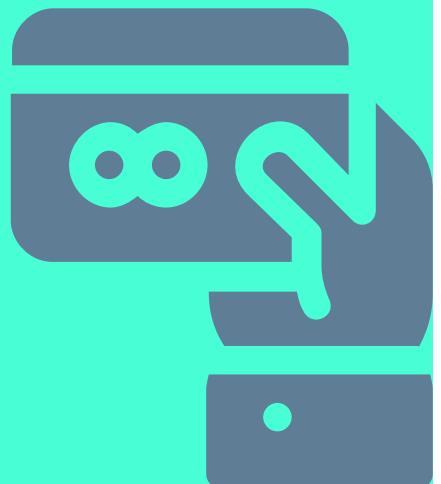
**Booking Service:** creazione, modifica e cancellazione delle prenotazioni

**Payment Service:** gestione dei pagamenti e del portafoglio virtuale

**Notification Service:** invio di email di conferma e aggiornamento

**Caratteristiche chiave**

- Gestione delle transazioni
- Controllo della coerenza dei dati
- Centralizzazione della logica di business





# Persistence Layer

Gestisce la memorizzazione e il recupero dei dati del sistema.

## Componenti principali

DAO (Data Access Object) per:

- Utenti
- Voli
- Prenotazioni
- Pagamenti
- Connessione al database tramite JDBC
- Database relazionale MySQL

## Caratteristiche chiave

- Supporto alle transazioni ACID
- Integrità referenziale dei dati
- Separazione tra accesso ai dati e logica applicativa

# Mappatura Hardware / Software

## Client

- Browser web (desktop o mobile)
- Interazione tramite protocollo HTTP/HTTPS

## Application Server

- Apache Tomcat
- Esecuzione delle Servlet Java e delle JSP
- Gestione della logica applicativa

## Database Server

- MySQL
- Memorizzazione persistente di utenti, voli e prenotazioni

La comunicazione tra i componenti avviene tramite protocolli standard, garantendo affidabilità e sicurezza.



# Controllo degli Accessi

## Ruoli del sistema

### Guest

- Ricerca e visualizzazione dei voli

### Utente Registrato

- Prenotazione dei voli
- Gestione delle prenotazioni
- Accesso all'area personale

### Amministratore

- Gestione dei voli
- Visualizzazione delle prenotazioni degli utenti

### Gestore Ordini

- Aggiornamento dello stato delle prenotazioni

### Sicurezza

- Autenticazione tramite credenziali
- Password memorizzate in forma cifrata
- Accesso alle funzionalità limitato in base al ruolo

Oggetti Attori	Catalogo	Ordini	Carrello
Cliente	visualizza	crea, visualizza (proprio)	aggiunge, rimuove, modifica, visualizza
Admin	aggiunge, modifica, elimina, visualizza (tutti)		
Gestore ordine	visualizza	visualizza gestisce stato (tutti)	
Guest	visualizza		aggiunge, rimuove, modifica (temporaneo)



# Boundary Conditions

## Esempi di condizioni gestite

Tentativo di prenotazione con posti non disponibili

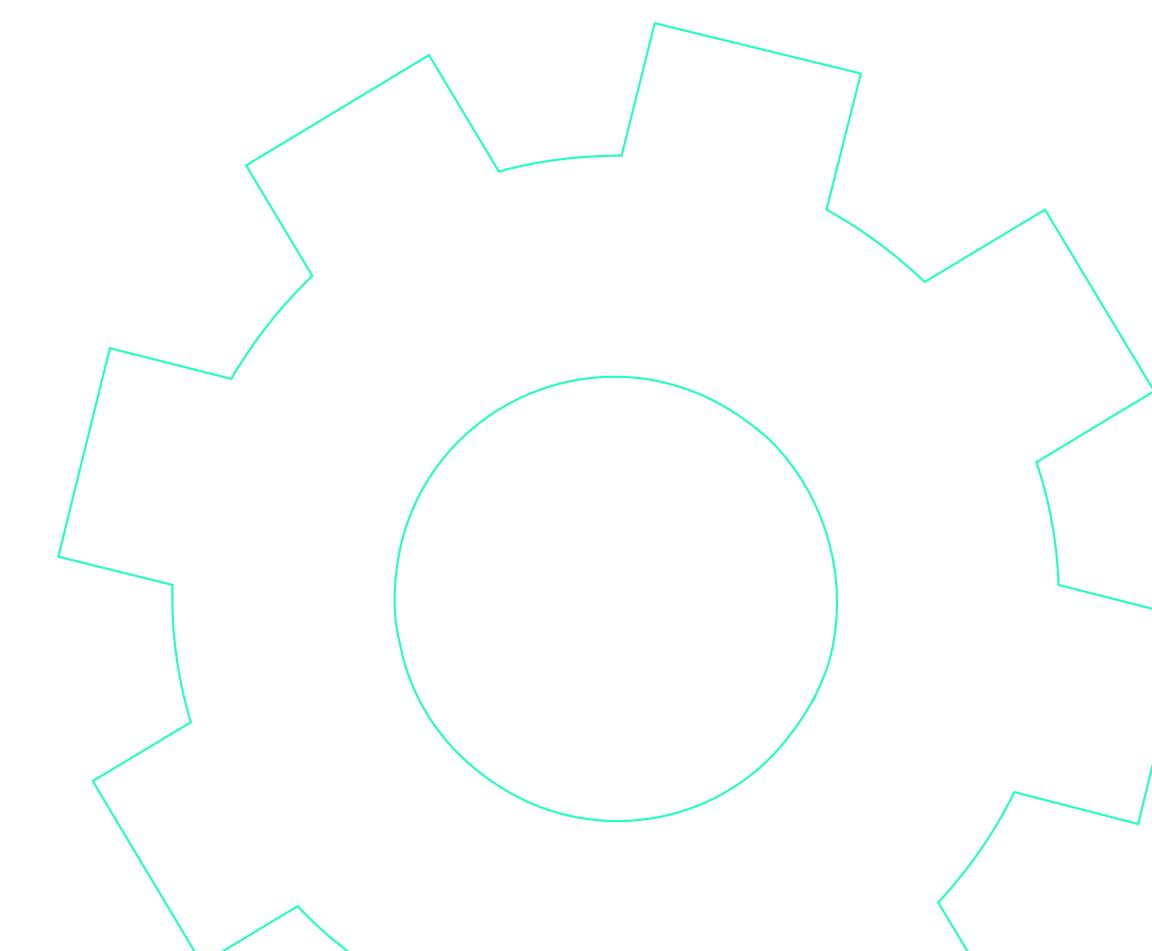
**Pagamento fallito** → prenotazione non confermata

Sessione utente scaduta durante il checkout

Errore di connessione al database

Modifica o cancellazione di una prenotazione non consentita

In tutti i casi, il sistema mantiene uno stato consistente e fornisce messaggi di errore chiari e comprensibili all'utente.



# Servizi dei sottosistemi

## Authentication

- Verifica delle credenziali utente
- Gestione login / logout
- Assegnazione e controllo dei ruoli (utente, admin, gestore ordini)

## Flight Catalog

- Gestione dei dettagli dei voli (compagnia, tratta, orari, prezzo, posti disponibili)
- Ricerca e filtraggio dei voli (partenza, arrivo, data, prezzo, compagnia)

## Security

Protezione dei dati sensibili degli utenti (hashing delle password, gestione sicura delle sessioni)

Controllo degli accessi ai sottosistemi in base al ruolo

Comunicazioni sicure tramite HTTPS

## Validation

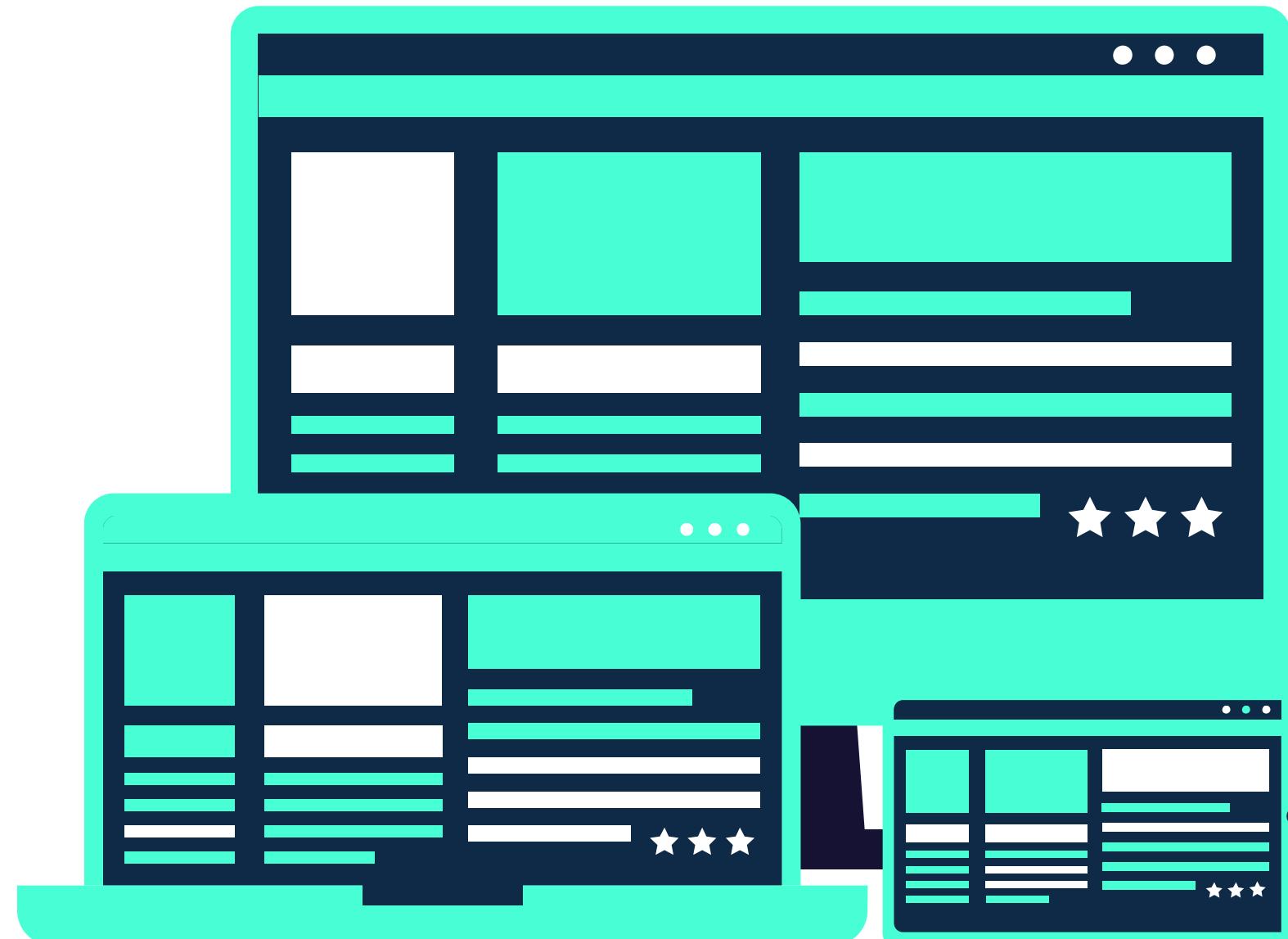
- Validazione degli input utente
- (filtri di ricerca, dati passeggeri, dati di pagamento)
- Verifica della disponibilità dei posti
- Validazione dei dati durante il checkout

## Cart (Carrello Prenotazioni)

- Aggiunta, rimozione e modifica dei voli nel carrello
- Aggiornamento del numero di posti selezionati
- Visualizzazione del contenuto del carrello

## Booking / Order

- Creazione delle prenotazioni
- Gestione del pagamento
- Aggiornamento dello stato della prenotazione
- Tracciamento delle prenotazioni e storico ordini



# Object Design

# Object Design

L'Object Design di ClickFly definisce la struttura interna degli oggetti software, le loro responsabilità, le interazioni e i principali compromessi progettuali adottati.

Il design è stato realizzato seguendo il pattern architettonale MVC (Model–View–Controller), con l'obiettivo di garantire una chiara separazione delle responsabilità, migliorare la manutenibilità del codice e favorire l'estendibilità del sistema nel tempo.

## Object Design Trade-Offs

### Buy vs Build

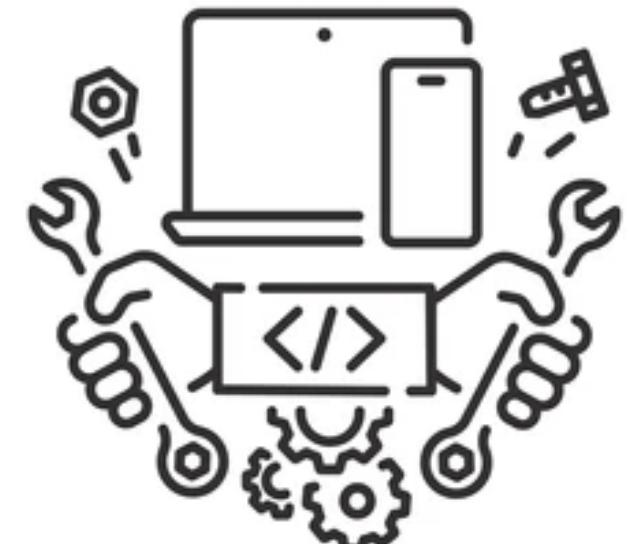
Per ClickFly è stato adottato un approccio ibrido:

#### Buy

Sono stati utilizzati framework e tecnologie consolidate, come Bootstrap per il front-end e Apache Tomcat come application server, al fine di ridurre i tempi di sviluppo e aumentare l'affidabilità complessiva del sistema.

#### Build

Le funzionalità core del sistema (ricerca voli, gestione del carrello, prenotazioni e pagamenti) sono state sviluppate internamente, garantendo pieno controllo sulla logica applicativa e sulle regole di business.



# Struttura dei Package

Il software ClickFly è organizzato secondo una struttura a livelli (layered architecture) che riflette il pattern MVC e favorisce manutenibilità, riuso ed estendibilità.

Livelli dell'architettura

## **Presentation Layer**

Gestisce l'interazione con l'utente

Comprende pagine JSP per la visualizzazione di:

- catalogo voli
- carrello
- checkout
- profilo utente
- Servlet (Controller) che ricevono le richieste HTTP e coordinano la logica applicativa
- Non contiene logica di business né accesso diretto al database

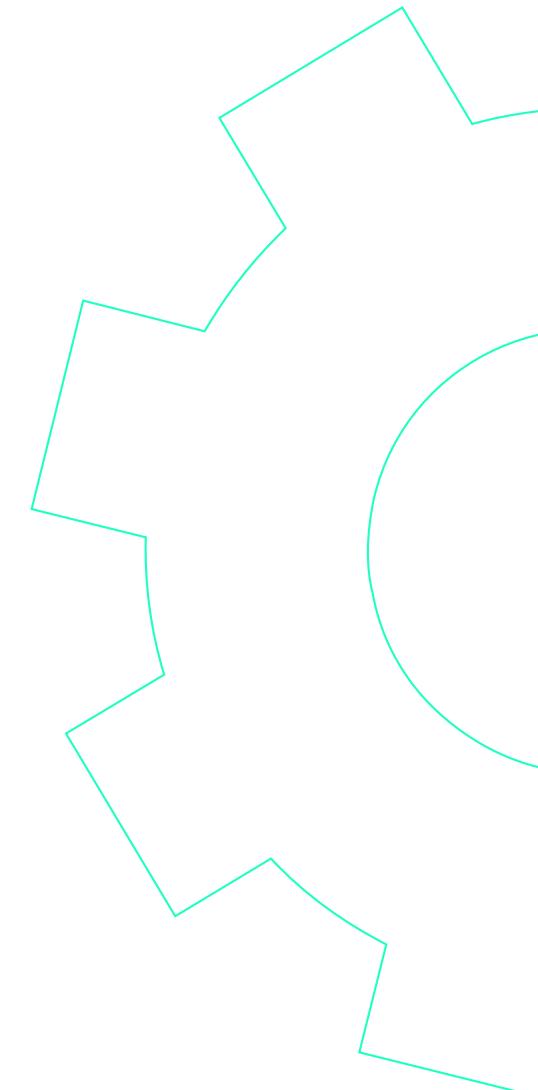


## **Service Layer**

- Contiene la logica di business dell'applicazione
- Responsabilità principali:
- Autenticazione e gestione utenti
- Ricerca e gestione dei voli
- Gestione del carrello
- Creazione e aggiornamento delle prenotazioni
- Gestione dei pagamenti
- Espone servizi utilizzati dai controller, mantenendo basso accoppiamento

## **Persistence Layer**

- Gestisce l'accesso ai dati persistenti
- Include:
- DAO (Data Access Object) per:
  - utenti
  - voli
  - prenotazioni
  - pagamenti
- Isola la logica di persistenza dal resto del sistema
- Implementato tramite JDBC (o equivalente)



# Organizzazione dei Package

L'organizzazione del progetto è suddivisa logicamente nei seguenti package:

## Model

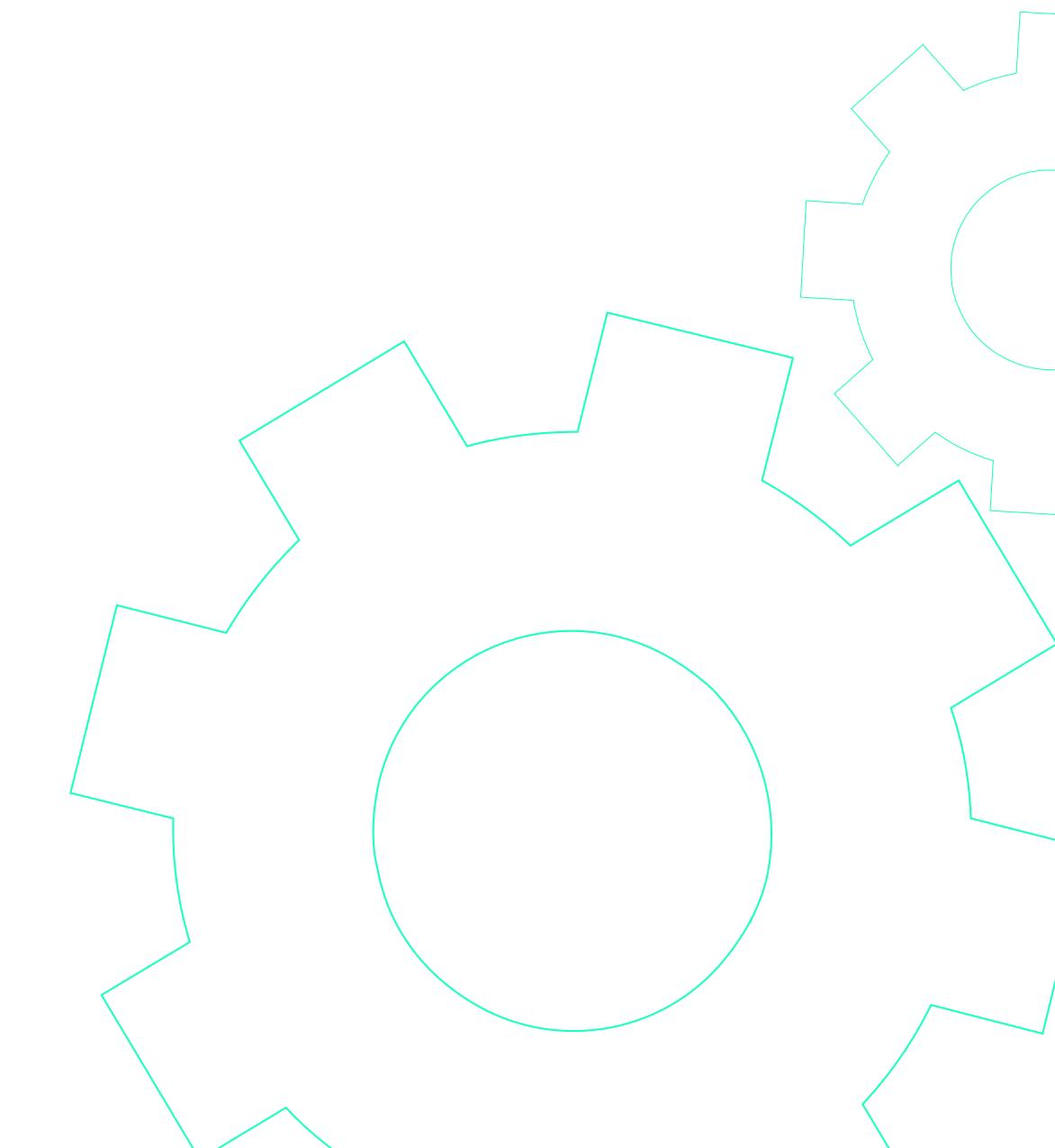
- Contiene i Java Beans che rappresentano le entità del dominio:
- Utente
- Volo
- Prenotazione
- Carrello
- Pagamento
- Le classi riflettono il Class Diagram e encapsulano i dati

## Service

- Contiene le classi che implementano la logica di business
- Coordina le operazioni tra controller e DAO
- Espone interfacce per facilitare estensioni future

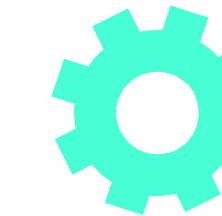
## Persistence (DAO)

- Contiene:
  - Interfacce DAO
  - Implementazioni concrete dei DAO
- Fornisce i metodi di accesso ai dati utilizzati dal Service Layer



## Controller

- Contiene le Servlet
- Gestisce le richieste HTTP e:
- invoca i servizi
- prepara i dati
- inoltra la risposta alle JSP



## WebApp

Contiene tutte le risorse web dell'applicazione:

### Pagine JSP

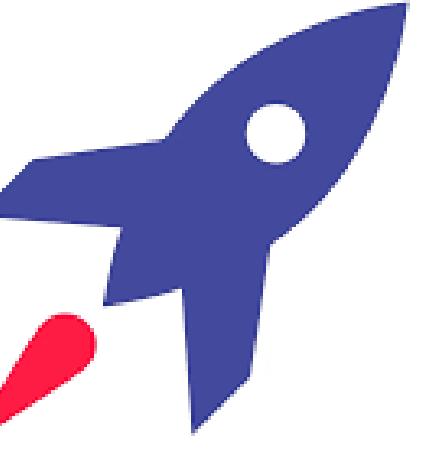
- Cartelle META-INF e WEB-INF per configurazioni e sicurezza
- Cartella resources con:
  - fogli di stile CSS
  - immagini
  - template riutilizzabili (header e footer)

## Gestione del progetto

- Utilizzo di Maven per:
  - gestione delle dipendenze
  - organizzazione del build

*Suddivisione chiara delle responsabilità per migliorare:*

- manutenibilità
- leggibilità
- scalabilità del progetto



# Memory Space vs Response Time

Il sistema privilegia tempi di risposta rapidi rispetto al consumo di memoria.

Il carrello e le informazioni di autenticazione vengono mantenuti nella sessione utente.

La disponibilità dei voli viene verificata in modo efficiente per ridurre accessi ripetuti al database.

Questo approccio comporta un maggiore utilizzo di memoria, considerato accettabile in favore di una migliore esperienza utente.

## Semplicità ed Estendibilità / Classi principali

La progettazione degli oggetti privilegia una struttura semplice e modulare:

- Utilizzo di interfacce per i servizi principali
- Basso accoppiamento tra le classi
- Facilità di estensione del sistema (es. nuovi metodi di pagamento o integrazione di nuove compagnie aeree) senza modifiche invasive alle componenti esistenti

# Classi principali del Model

## Utente

Rappresenta un utente del sistema.

Attributi: id, nome, email, password, ruolo

## Responsabilità:

- Autenticazione
- Accesso alle funzionalità in base al ruolo

# Classi: Volo, Carrello, Prenotazione

## Volo

Rappresenta un volo disponibile alla prenotazione.

Attributi: idVolo, partenza, destinazione, data, orario, postiDisponibili, prezzo, compagnia

## Responsabilità:

- Fornire informazioni sul volo
- Gestire la disponibilità dei posti



## **Carrello**

Contiene i voli selezionati dall'utente.

### **Attributi:**

listaVoli, postiSelezionati

### **Responsabilità:**

Aggiunta e rimozione dei voli

Aggiornamento dei posti selezionati

Verifica della disponibilità

## **Prenotazione**

Rappresenta una prenotazione confermata.

### **Attributi:**

idPrenotazione, utente, voli, stato, dataPrenotazione

### **Responsabilità:**

Gestione dello stato della prenotazione

Collegamento tra utente e voli prenotati



## **Service Objects**

AuthenticationService

login()

logout()

verificaRuolo()

Gestisce l'autenticazione degli utenti e il controllo degli accessi.

## **CartService**

aggiungiVolo()

aggiornaPosti()

rimuoviVolo()

## **Post-condizioni corrette:**

Se il volo non è presente nel carrello → viene inserito con i posti richiesti

Se il volo è già presente → viene aggiornata la quantità dei posti (nei limiti della disponibilità)  
L'operazione restituisce true se va a buon fine,  
false altrimenti

# Booking, Payment e DAO

## BookingService

- creaPrenotazione()
- annullaPrenotazione()
- confermaPrenotazione()

Gestisce il flusso di prenotazione coordinando carrello, disponibilità e pagamento.

## PaymentService

- effettuaPagamento()

Garantisce l'atomicità del processo di pagamento:

- Se il pagamento fallisce → la prenotazione non viene confermata
- La disponibilità dei posti rimane invariata

## Persistence Objects (DAO) – ClickFly

- UserDAO
- FlightDAO
- BookingDAO
- PaymentDAO

Ogni DAO isola la logica di accesso ai dati dalla logica applicativa, seguendo il pattern DAO.



# Invarianti e Boundary Conditions

## Invarianti principali

- Un utente non autenticato non può effettuare prenotazioni
- Non è possibile confermare una prenotazione se i posti non sono disponibili
- Ogni pagamento confermato corrisponde a una prenotazione valida
- Le prenotazioni devono essere sempre consistenti con la disponibilità dei voli

## Boundary Conditions (Object Level)

Tentativo di aggiungere un volo esaurito al carrello

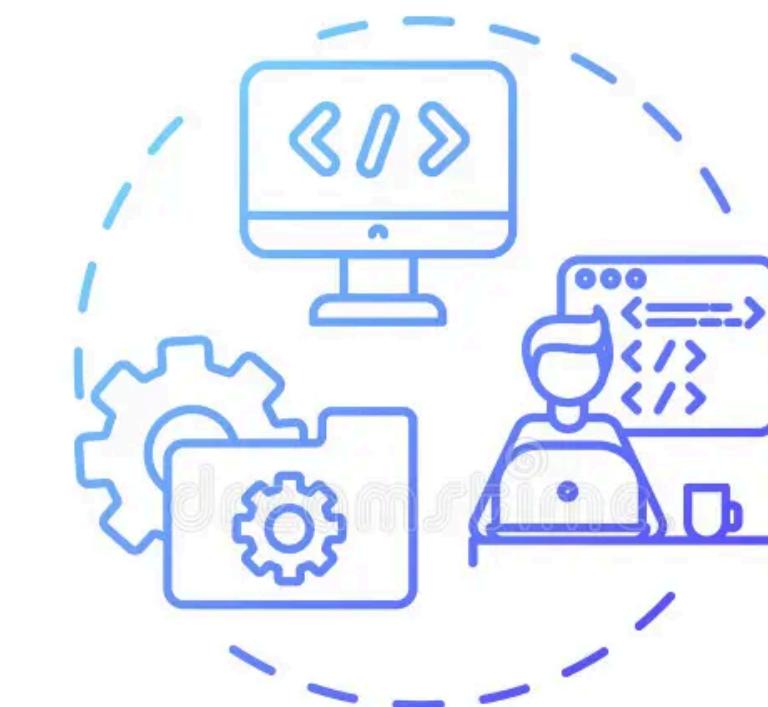
- Modifica dei posti oltre la disponibilità residua
- Pagamento fallito durante il checkout
- Accesso a funzionalità riservate senza permessi

## Pattern architetturali utilizzati

**MVC:** separazione tra Model, View e Controller

**DAO:** accesso ai dati strutturato e indipendente

**Service Layer:** coordinamento della logica di business

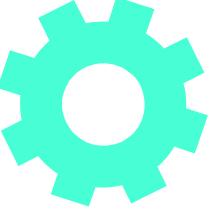




# Test Plan

# Test Plan

---



Il piano di testing descrive le strategie e gli approcci adottati per verificare la qualità del sistema ClickFly e garantirne correttezza, stabilità e affidabilità.

## Obiettivi del piano di test

- Verificare il corretto funzionamento delle funzionalità principali
- Validare le interazioni tra i diversi moduli del sistema
- Identificare difetti sia in situazioni standard che in casi eccezionali

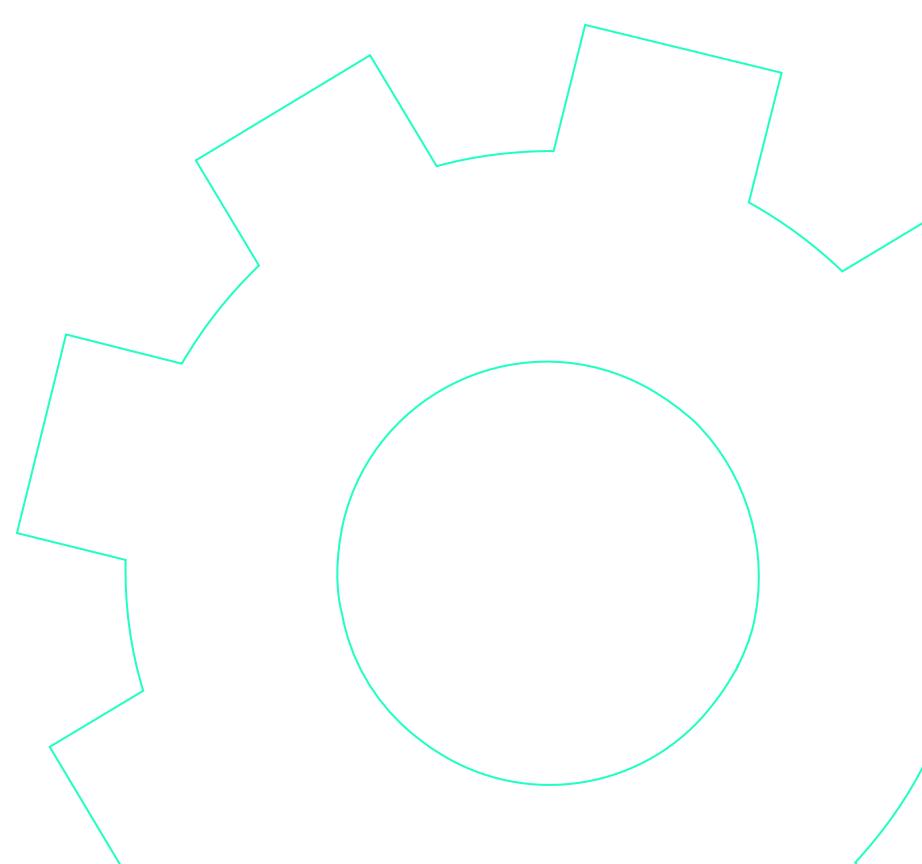
## Scenari Considerati

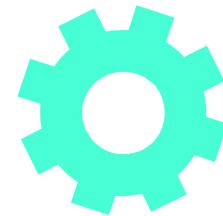
- Operazioni standard
  - ricerca e visualizzazione voli
  - gestione del carrello
  - prenotazione e pagamento

## Eventi eccezionali

- pagamenti falliti
- posti non disponibili
- sessioni scadute
- errori di validazione o autorizzazione

L'obiettivo è individuare errori e comportamenti anomali del sistema.





## Collegamenti ad altri documenti

Il piano di test si basa sui seguenti documenti di progetto:

- **RAD** (Requirements Analysis Document)

Utilizzato per derivare i casi di test a partire dai requisiti funzionali e non funzionali.

- **SDD** (System Design Document)

Fornisce informazioni sull'architettura del sistema, utili per progettare test di integrazione e di sistema.

- **ODD** (Object Design Document)

Describe classi e interazioni, consentendo di verificare il corretto comportamento dei componenti software.

## Panoramica del Sistema

ClickFly è un'applicazione web per la prenotazione di voli, sviluppata **in Java** e progettata per ambienti desktop e mobile.

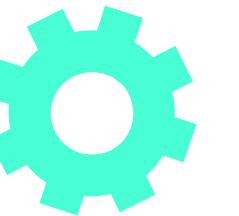
## Tecnologie utilizzate

**Application Server:** Apache Tomcat 9

**Back-end:** Servlet e logica applicativa Java

**Front-end:** JSP, HTML, CSS, Bootstrap

**Database:** MySQL



## Funzionalità Principali

### Gestione utenti

- registrazione
- login/logout

### Catalogo voli

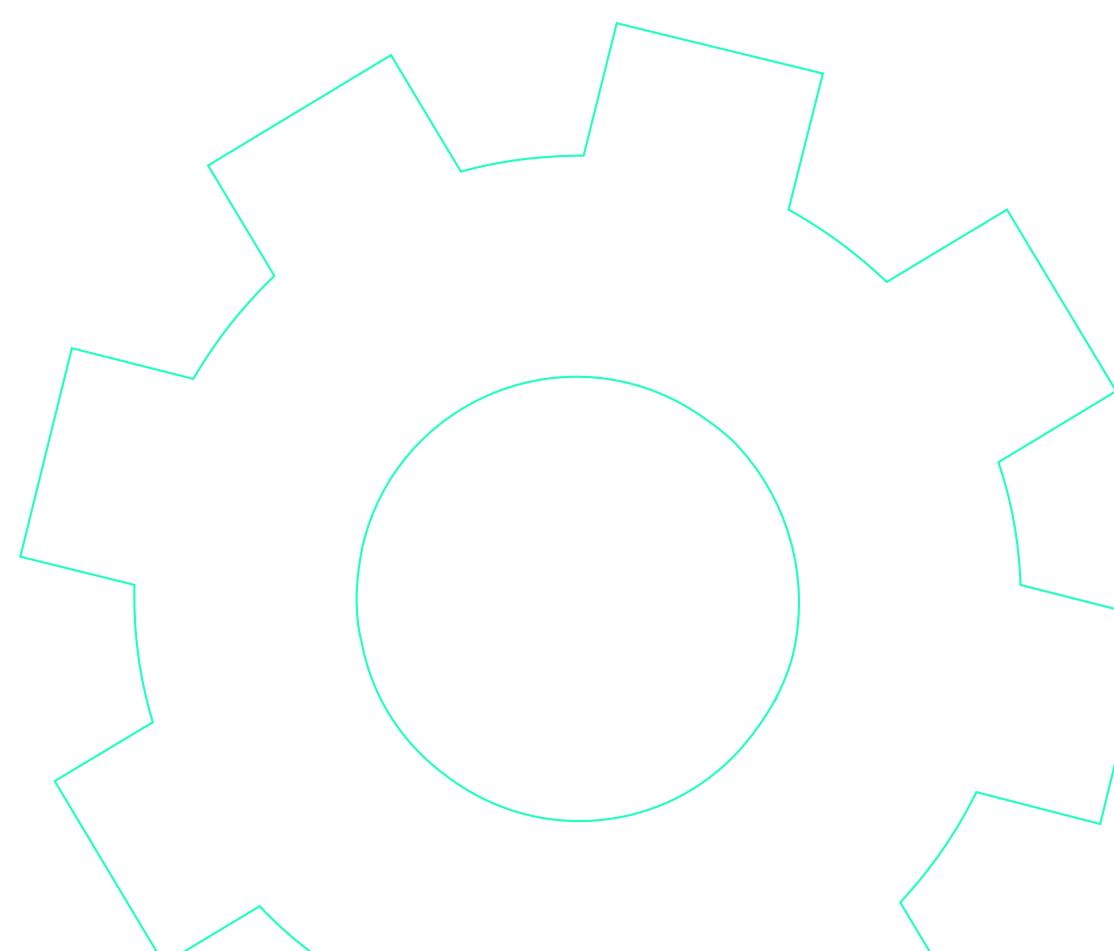
- ricerca e filtraggio
- visualizzazione dettagli

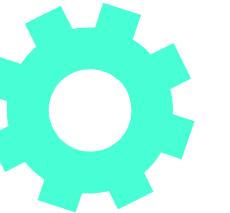
### Carrello

- aggiunta/rimozione voli
- modifica numero di posti

### Prenotazioni

- checkout e pagamento
- Dashboard amministrativa
- gestione voli
- gestione prenotazioni





## Caratteristiche da Testare

### Utente non autenticato (Guest)

- Registrazione
- Navigazione del catalogo voli

### Utente registrato

- Autenticazione
- Gestione carrello
- Prenotazione voli
- Consultazione storico prenotazioni

### Amministratore

- Inserimento, modifica e rimozione voli
- Visualizzazione e aggiornamento stato prenotazioni

## Criteri di Successo e Fallimento

### Successo

- Il sistema produce risultati conformi ai requisiti
- Le funzionalità critiche operano correttamente
- Nessun errore bloccante rilevato

### Fallimento

- Presenza di difetti durante l'esecuzione dei test
- Comportamenti non conformi ai requisiti specificati

# Approccio al Testing

## Obiettivi del Testing

- Validare la conformità ai requisiti
- Individuare e correggere difetti
- Verificare sicurezza, prestazioni e stabilità

## Metodologia

**Test di unità:** verifica delle singole funzioni

**Test di integrazione:** interazione tra moduli

**Test di sistema:** funzionamento complessivo dell'applicazione

## Risorse e Strumenti

### Requisiti Hardware

- PC con almeno 8 GB di RAM
- Processore multi-core



## Requisiti Software

Windows 11

Browser moderni (Chrome, Firefox, Edge)

## Strumenti di Testing

- JUnit e Mockito per test di unità
- MySQL per verifiche sui dati
- Selenium per l'automazione dei test funzionali

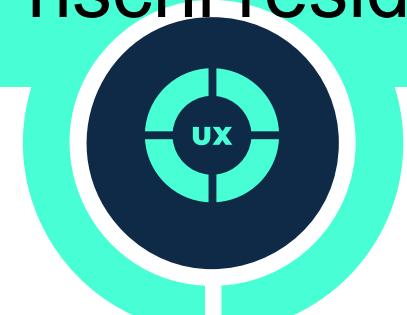
## Criteri di Ingresso e Uscita

### Criteri di Ingresso

- Requisiti approvati
- Ambiente di test correttamente configurato
- Dati di test disponibili

### Criteri di Uscita

- Copertura completa delle funzionalità testate
- Assenza di difetti critici
- Accettazione o mitigazione dei rischi residui



# Packages – Livelli Logici

## Gestione dei Difetti

### Classificazione dei Difetti

**Critici:** bloccano il sistema

**Alti:** devono essere risolti prima del rilascio

**Medi/Bassi:** risolvibili in fasi successive

### Processo

- Registrazione del difetto
- Assegnazione priorità
- Correzione
- Nuova validazione
- Chiusura

### Metriche di Test

### Misurazioni Utilizzate

Percentuale di test superati

Numero di difetti per gravità

- Copertura dei requisiti tramite casi di test

### Scopo

- Monitorare l'efficacia del piano di test
- Migliorare la qualità complessiva del sistema

### Riassunto del Piano di Test

Il piano di test di ClickFly adotta un approccio strutturato per garantire:

- Corretta validazione delle funzionalità critiche
- Individuazione e gestione dei difetti
- Affidabilità e qualità globale del sistema

# Packages – Livelli Logici

## Metodologie

- Black-box testing
- Bottom-up testing
- Test di integrazione e di sistema

## Strumenti

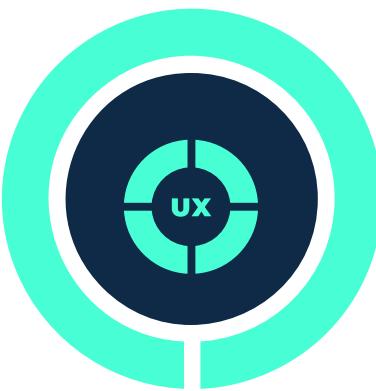
- JUnit
- Selenium
- Strumenti di monitoraggio e tracciamento dei difetti



# Test Case Specification

# Scopo del Documento

Questo documento definisce il piano di collaudo del sistema ClickFly, un'applicazione web progettata per la ricerca, selezione e prenotazione di voli online.



Il collaudo ha lo scopo di verificare che:

le funzionalità implementate siano conformi ai requisiti definiti nel RAD;

il sistema sia privo di errori critici;

i flussi principali e alternativi siano gestiti correttamente.

La specifica dei test consente di individuare tempestivamente eventuali difetti, contribuendo a migliorare affidabilità, stabilità e qualità complessiva del software.

## Strategia di Test

La strategia di test adottata per ClickFly si basa sul Category Partition Method, una tecnica che permette di progettare i casi di test suddividendo i parametri di input in categorie rappresentative.



Questo approccio consente di:

coprire in modo sistematico gli scenari principali;

includere casi limite ed eccezionali;

evitare ridondanze non necessarie;

garantire un'elevata copertura funzionale con un numero controllato di test.



# Identificazione dei Parametri

Per ciascuna funzionalità del sistema ClickFly vengono individuati i parametri rilevanti, utilizzati come input per i test.



## Parametri principali

### Registrazione / Autenticazione

- email
- password



### Ricerca voli

- aeroporto di partenza
- aeroporto di arrivo
- data del volo

### Prenotazione

- volo selezionato
- numero di posti



### Pagamento

- metodo di pagamento
- dati di pagamento

Tali parametri determinano il comportamento del sistema e l'esito dei test in base alle combinazioni considerate.

# Categorie dei Parametri

Ogni parametro è suddiviso in categorie di valori, che rappresentano differenti situazioni di input:

## Valori corretti

rispettano i requisiti funzionali

## Valori limite

testano condizioni ai margini della validità (es. ultimo posto disponibile)

## Valori non validi

verificano la corretta gestione degli errori e delle eccezioni

Questa suddivisione permette di valutare la robustezza del sistema in condizioni normali e critiche.

## Frame di Test

Un frame di test è una combinazione logica di categorie di parametri che rappresenta uno specifico scenario di collaudo.

## Esempio – Autenticazione

Email:

- formato corretto
- formato errato

Password:

- corretta
- errata



## Esempio – Prenotazione volo

Numero di posti:

- disponibile
- non disponibile



Pagamento:

- riuscito
- fallito

Questo approccio consente di modellare sistematicamente tutti i comportamenti possibili del sistema.

## Casi di Test

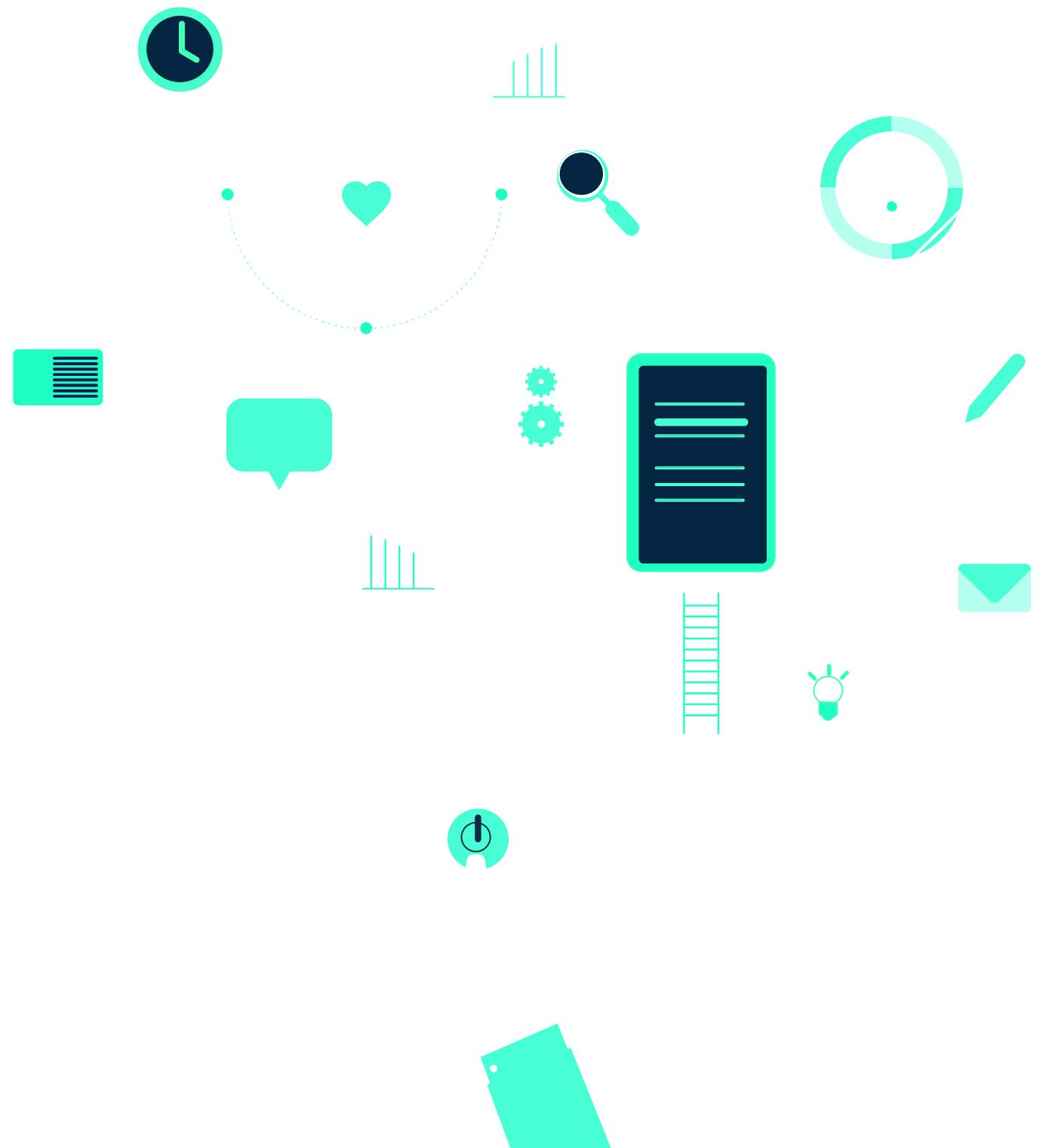
Per ciascun modulo di ClickFly, dopo la definizione dei relativi test frame, verranno forniti:

- un caso di test riuscito
- un caso di test fallito al fine di dimostrare:
- il corretto funzionamento del sistema in condizioni valide;
- la gestione appropriata degli errori e delle situazioni anomale.

# Registrazione

Test frames:

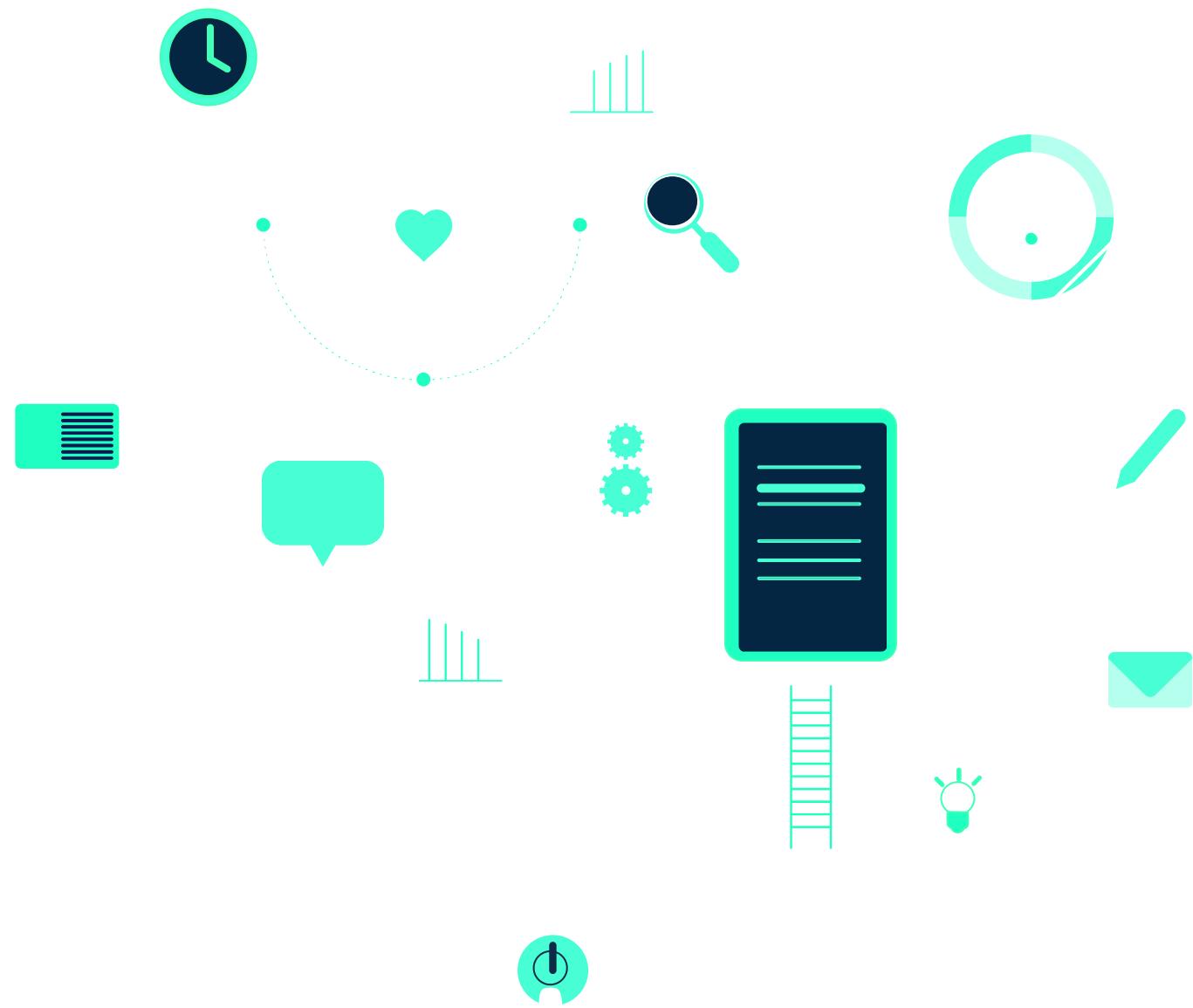
Parametri	email, username, password, confirm Password
Necessità ambientali	Database
Categorie:	Scelte:
lunghezza username:	US1: lunghezza<1 US2: lunghezza>=1
formato email	FM1: formato email corretto FM2: formato email non corretto.
email presente nel database	EP1: email presente. EP2: email non presente.



matching password	MA1: password e confirmPassword non combaciano. MA2: password e confirmPassword combaciano.
formato password	PA1: formato password non corretto. PA2: formato password corretto.
properties e selettori:	FM1: formato email corretto FM2: formato email non corretto. [ERROR]
	EP1: email presente. [ERROR] EP2: email non presente.
	MA1: password e confirmPassword non combaciano. [ERROR] MA2: password e confirmPassword combaciano.
	PA1: formato password non corretto. [ERROR] PA2: formato password corretto.
	US1: lunghezza<1 [ERROR] US2: lunghezza>=1

#### Test Cases:

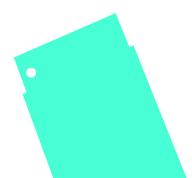
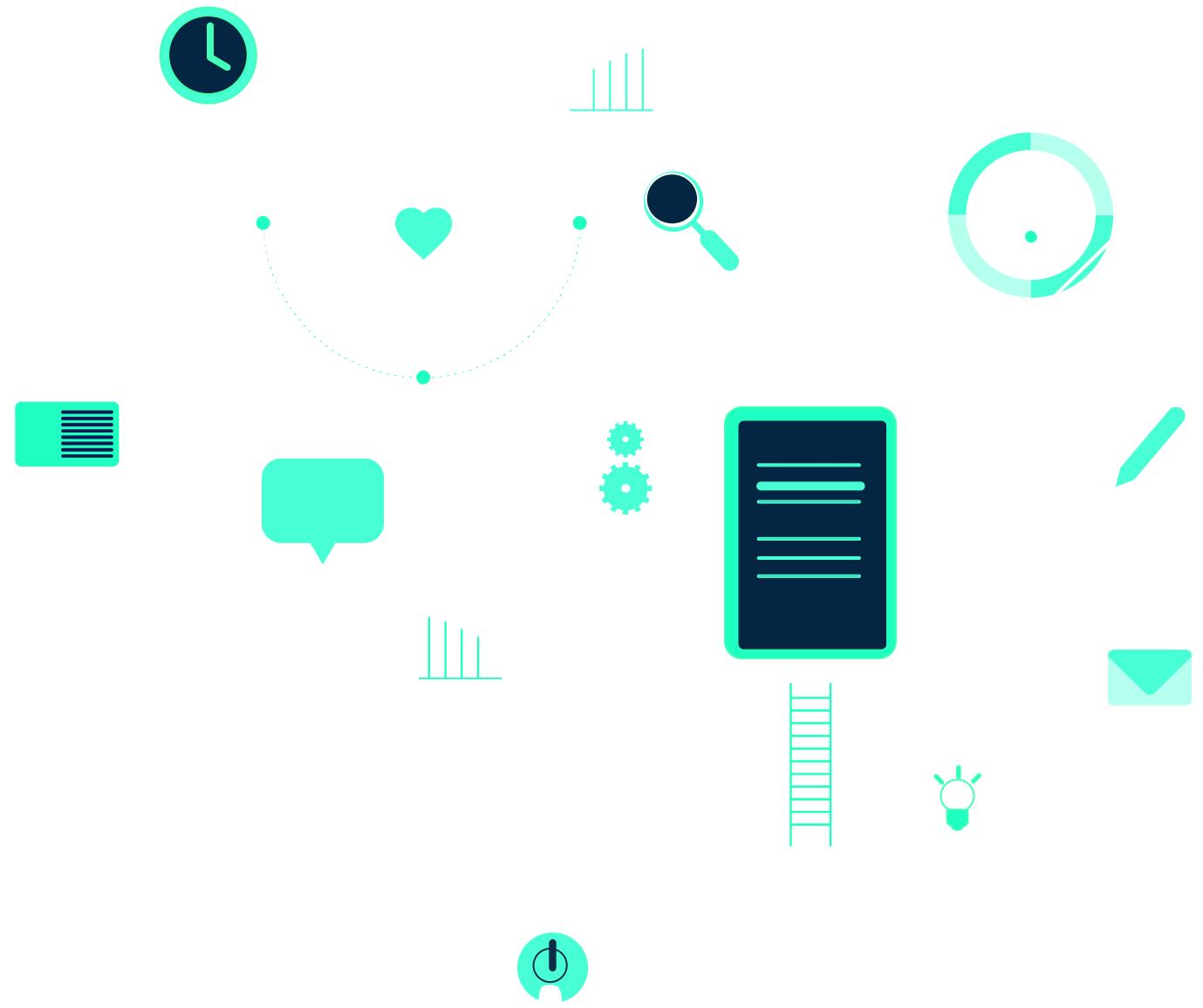
TC1 Registrazione valida	Test frames: FM1, EP2, MA2, PA2, US2
INPUT	
Email	user@gmail.com
Username	utente123



matching password	MA1: password e confirmPassword non combaciano. MA2: password e confirmPassword combaciano.
formato password	PA1: formato password non corretto. PA2: formato password corretto.
properties e selettori:	FM1: formato email corretto FM2: formato email non corretto. [ERROR]
	EP1: email presente. [ERROR] EP2: email non presente.
	MA1: password e confirmPassword non combaciano. [ERROR] MA2: password e confirmPassword combaciano.
	PA1: formato password non corretto. [ERROR] PA2: formato password corretto.
	US1: lunghezza<1 [ERROR] US2: lunghezza>=1

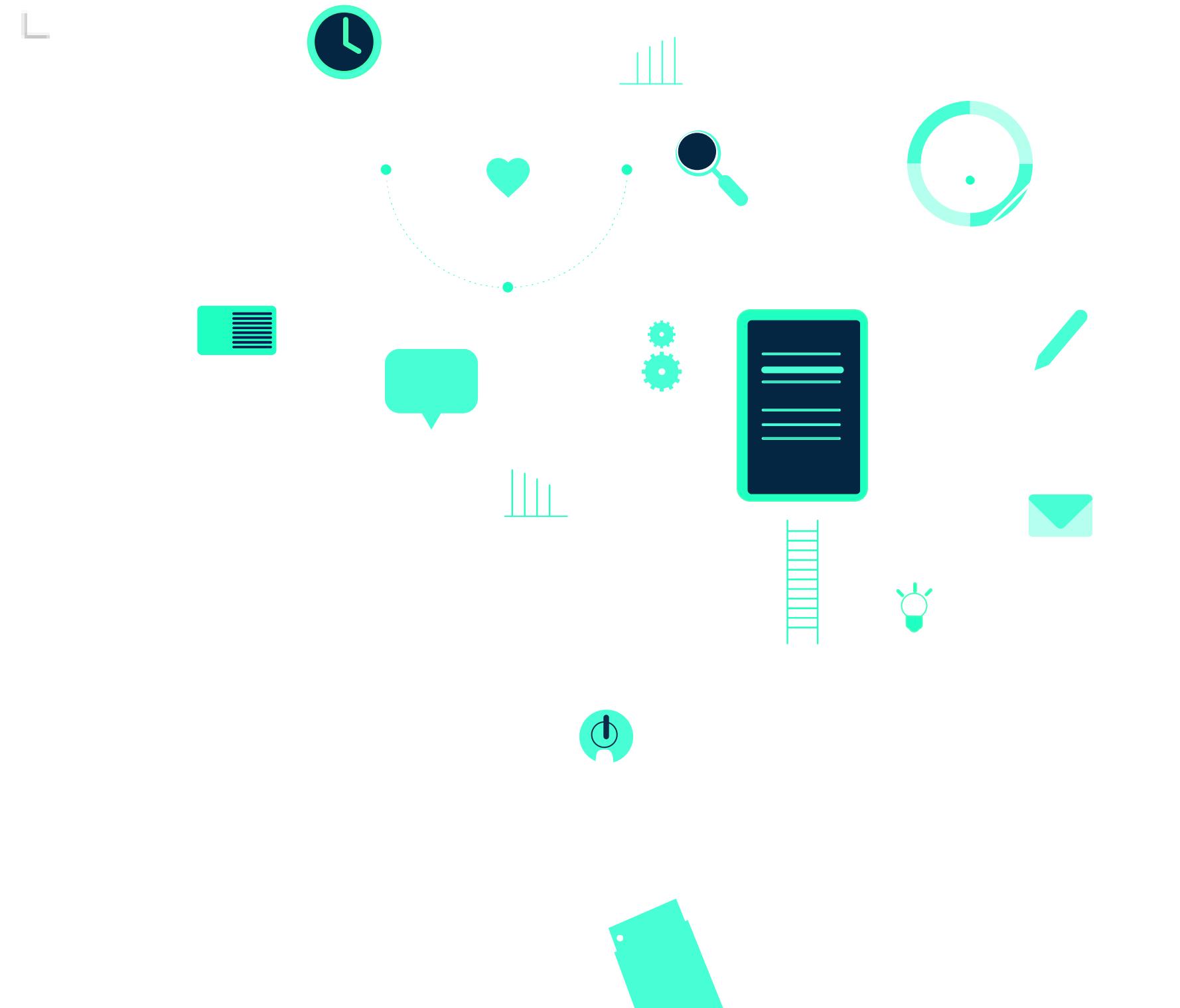
### Test Cases:

TC1 Registrazione valida	Test frames: FM1, EP2, MA2, PA2, US2
<b>INPUT</b>	
Email	user@gmail.com
Username	utente123



Password	M@rcoP@ssw0rd
Confirm Password	M@rcoP@ssw0rd
ORACOLO	
<i>Registrazione completata con successo.</i>	

TC2 Email non corretta	Test frames: EP2, MA2, PA2, US2, FM2
INPUT	
Email	user@nomail
Username	utente123
Password	M@rcoP@ssw0rd
Confirm Password	M@rcoP@ssw0rd
ORACOLO	

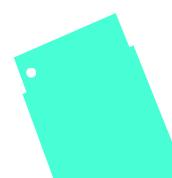
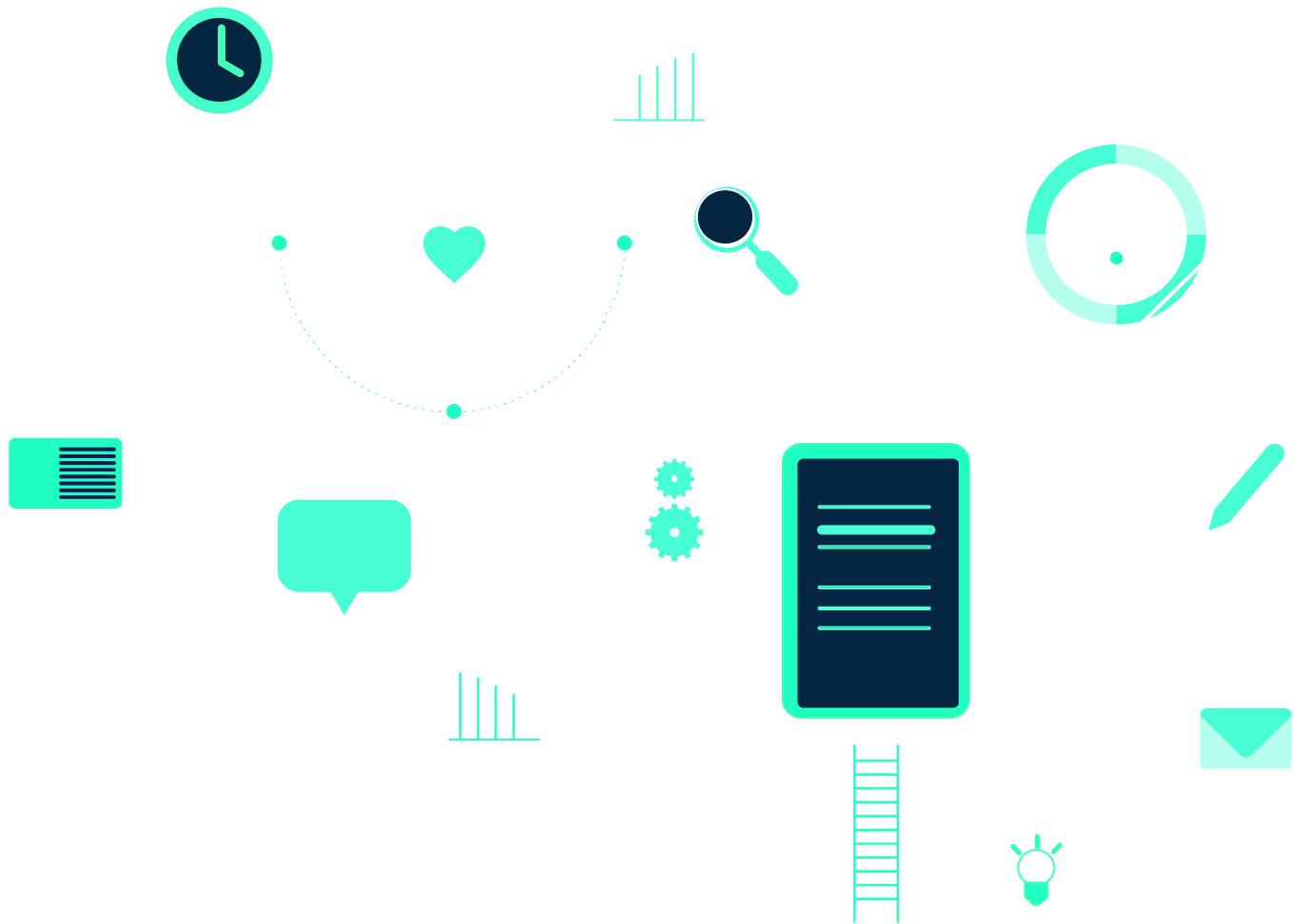


TC3 Username vuoto	Test frames: FM1, EP2, MA2, PA2, US1
--------------------	--------------------------------------

	Ingegneria del Software	Pagina 8 di 40
--	-------------------------	----------------

INPUT	
Email	user@gmail.com
Username	
Password	M@rcoP@ssw0rd
Confirm Password	M@rcoP@ssw0rd
ORACOLO	
<i>Registrazione non completata con successo e messaggio di errore "L'username non può essere una stringa vuota".</i>	

TC4 Password non valida	Test frames: FM1, EP2, MA2, US2, PA1
INPUT	
Email	user@gmail.com
Username	utente
Password	Marco
Confirm Password	Marco
ORACOLO	



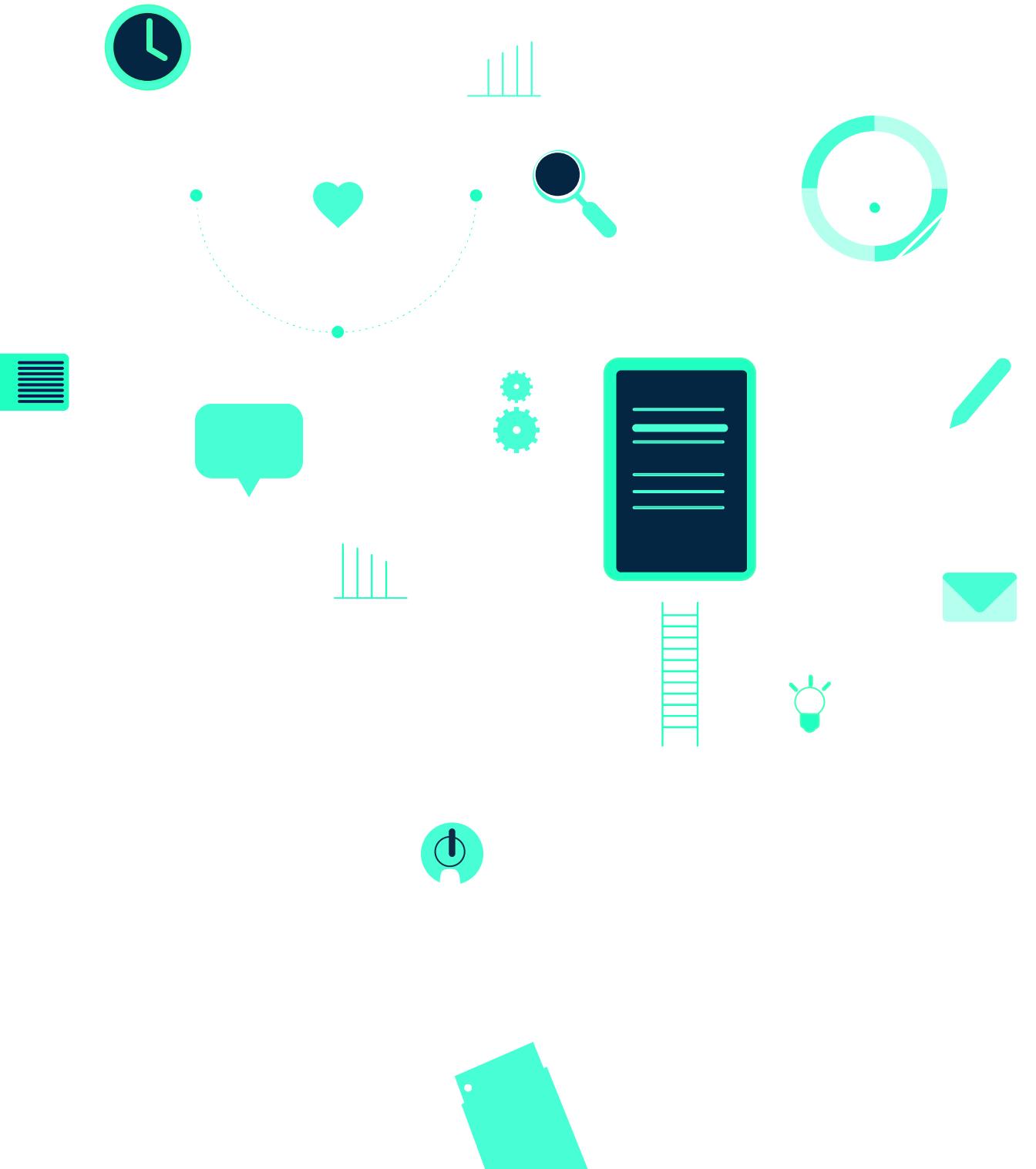
Registrazione non completata con successo e messaggio di errore "La password deve essere di almeno 8 caratteri, con almeno un numero, una maiuscola, una minuscola e un carattere speciale".

TC5 Password non combaciano	Test frames: FM1, EP2, PA2, US2, MA1
<b>INPUT</b>	
Email	user@gmail.com
Username	utente
Password	M@rcop@ssw0rd

	Ingegneria del Software	Pagina 9 di 40
--	-------------------------	----------------

Confirm Password	M@rcop@ssw0rd22
<b>ORACOLO</b>	
<i>Registrazione non completata con successo e messaggio di errore "Le password non combaciano".</i>	

TC6 Email già presente nel database	Test frames: FM1, EP1, MA2, PA2, US2
<b>INPUT</b>	
Email	user@gmail.com
Username	utente123



Password	M@rcoP@sswOrd
Confirm Password	M@rcoP@sswOrd
ORACOLO	
<i>Registrazione non completata con successo e messaggio di errore "Email già presente nel database".</i>	

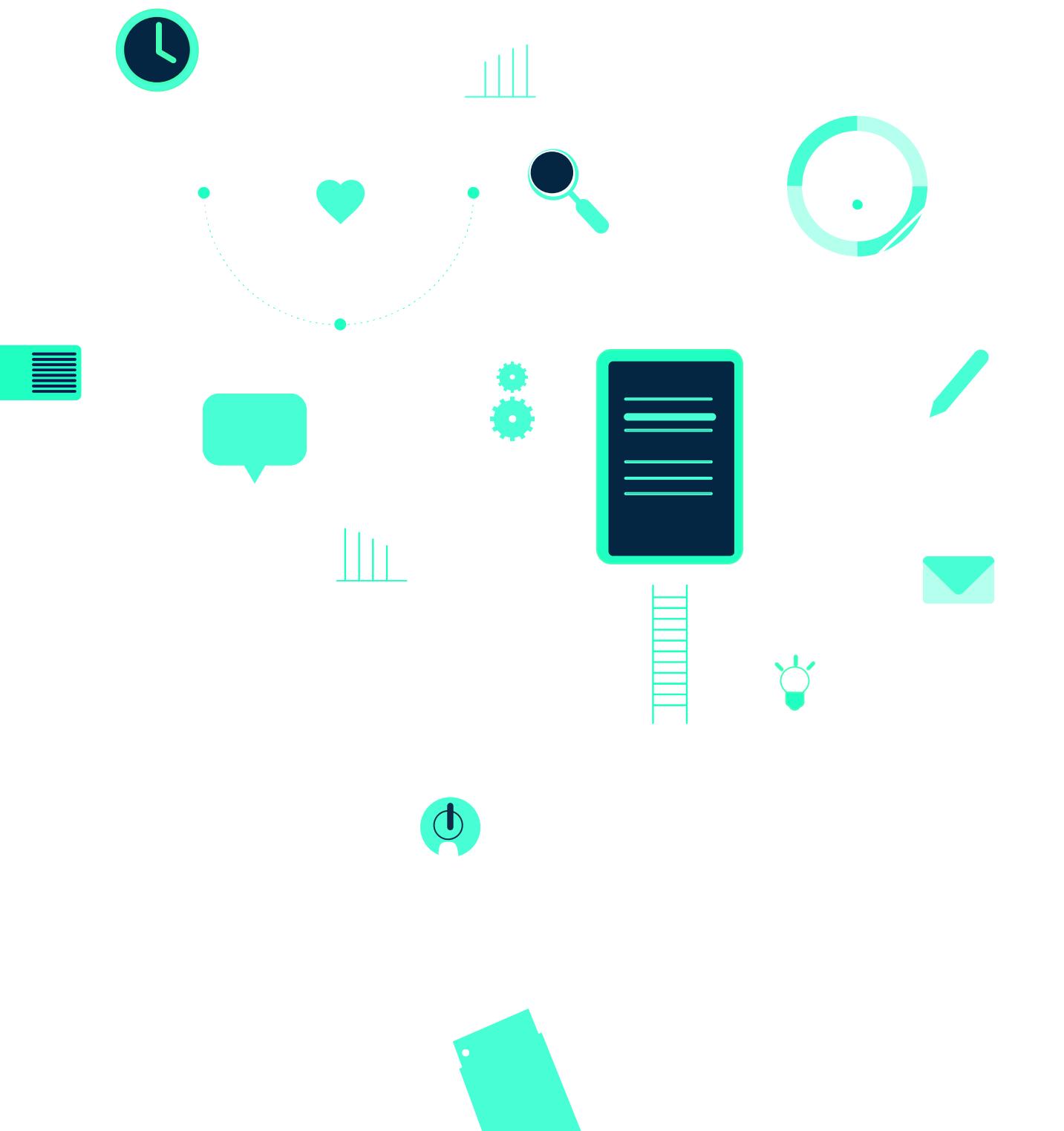
# Autenticazione

## Test frames:

Parametri	email, password
Necessità ambientali	Database
Categorie:	Scelte:
username esistente:	ES1: email presente nel database ES2: email non presente nel database
password associata:	PSA1: password associata. PSA2: password non associata.
properties e selettori:	ES1: email presente nel database

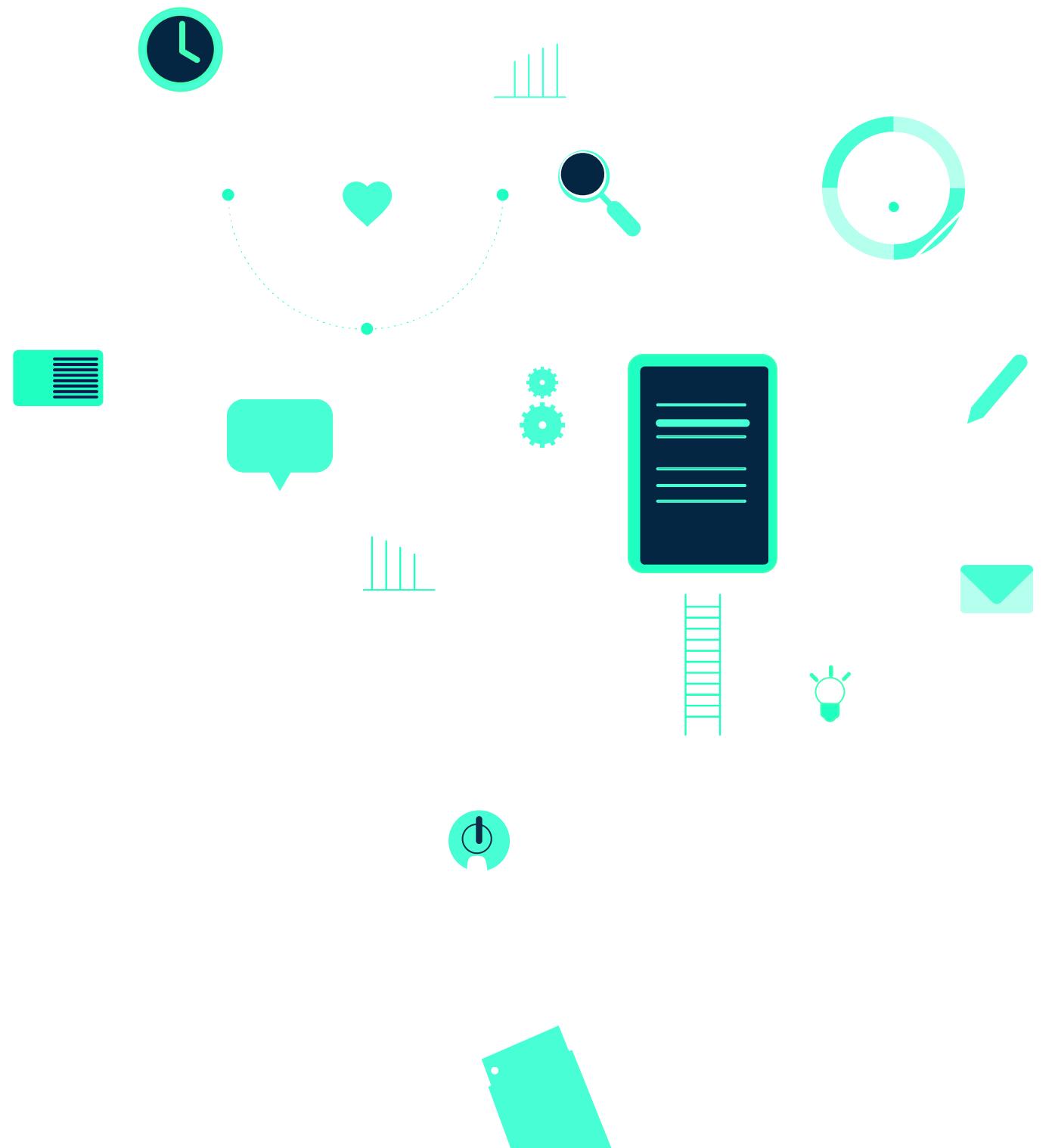
	Ingegneria del Software	Pagina 10 di 40
--	-------------------------	-----------------

	ES2: email non presente nel database [ERROR]
	PSA1: password associata. PSA2: password non associata. [ERROR]



### Test cases:

TC7 Autenticazione riuscita	Test frames: UP1, PSA1
<b>INPUT</b>	
Email	utente123@gmail.com
Password	M@rcoP@ssw0rd
<b>ORACOLO</b>	
<i>Autenticazione completata con successo.</i>	

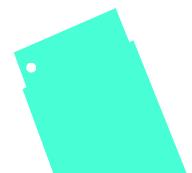
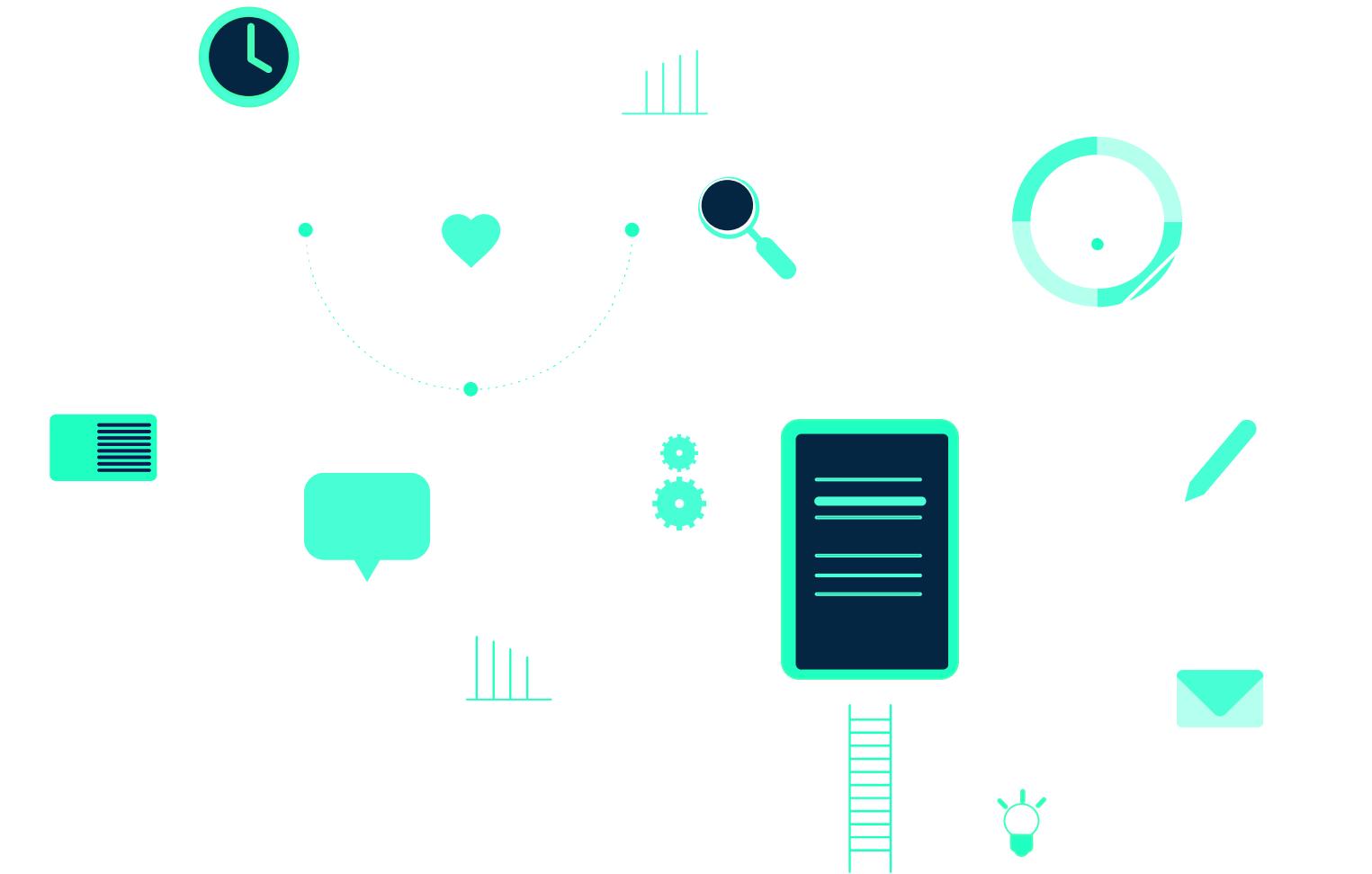


TC8 Email non presente nel database	Test frames: UP2, PSA1
<b>INPUT</b>	
Email	utente123@gmail.com
Password	M@rcoP@ssw0rd
<b>ORACOLO</b>	
<i>Autenticazione fallita e messaggio d'errore "Non esiste nessun account associato a questa username"</i>	

TC9 Password non corretta	Test frames: PSA2, UP1
<b>INPUT</b>	

Email	utente123@gmail.com
Password	Marcolino123@
ORACOLO	
<i>Autenticazione fallita e messaggio d'errore "La password è sbagliata. Ritenta"</i>	

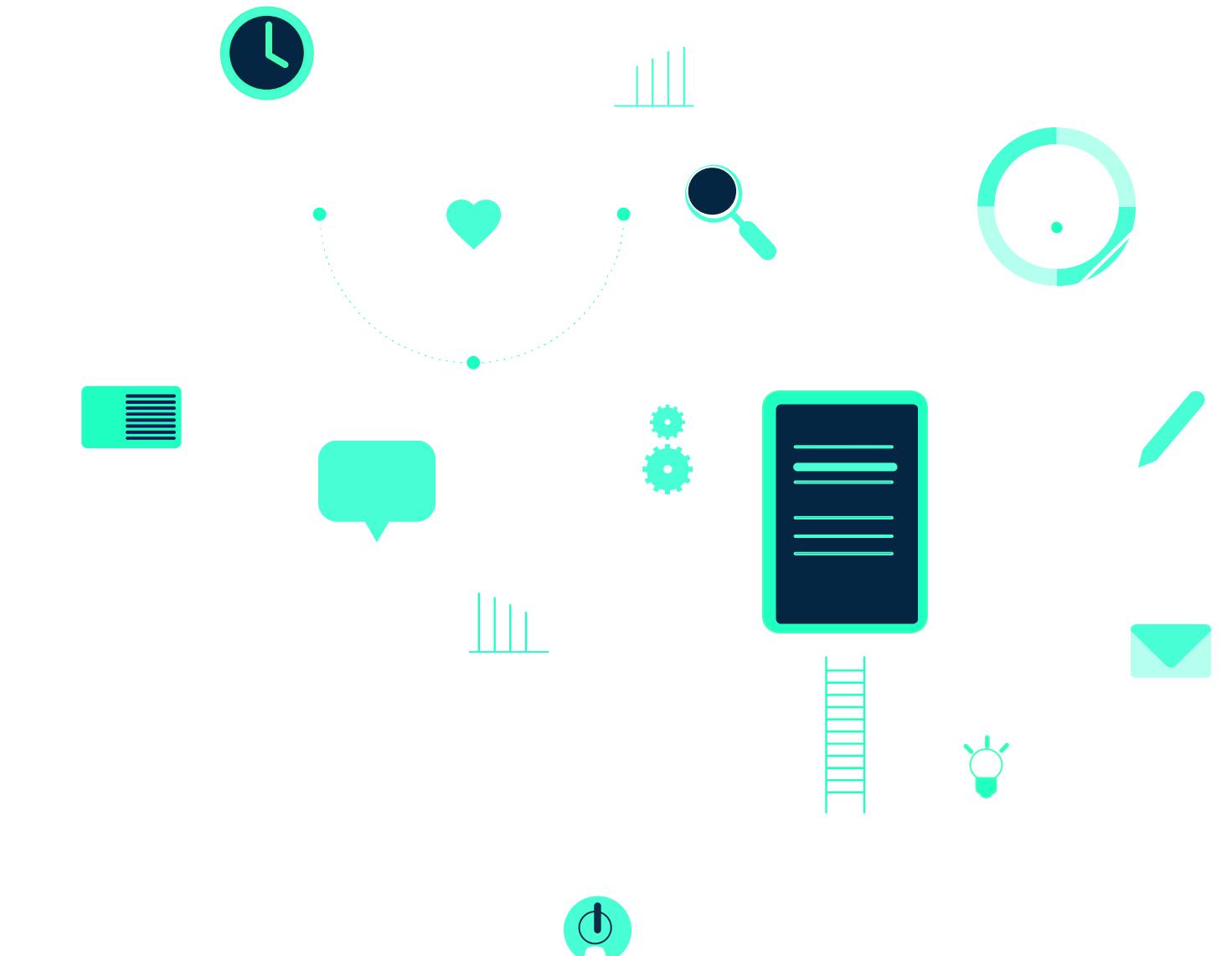
**Aggiunta di un nuovo prodotto al catalogo**



### Test frames:

Parametri	destinazione, idViaggio, prezzo, compagnia aerea,aeroportoPartenza,numeroPosti
Necessità ambientali	Database
Categorie:	Scelte: FD1: formato destinazione corretto FD2: formato destinazione non corretto
formato nome viaggio	FNV1: formato nome viaggio corretto FNV2: formato nome viaggio non corretto
id viaggio presente nel database	IDP1: id viaggio non presente IDP2: id viaggio già presente.
formato id	FID1: formato id corretto. FID2: formato id non corretto.
prezzo	PP1: prezzo >0.00 PP2: prezzo ≤0.00

# Aggiunta di un nuovo prodotto al catalogo



formato compagnia aerea	FB1: formato compagnia corretto. FB2: formato compagnia non corretto.
formato aeroporto di partenza	FAP1: formato aeroporto corretto FAP2: formato aeroporto non corretto
numero posti disponibili	NP1: numero posti > 0 NP2: numero posti ≤ 0

**Test cases:**

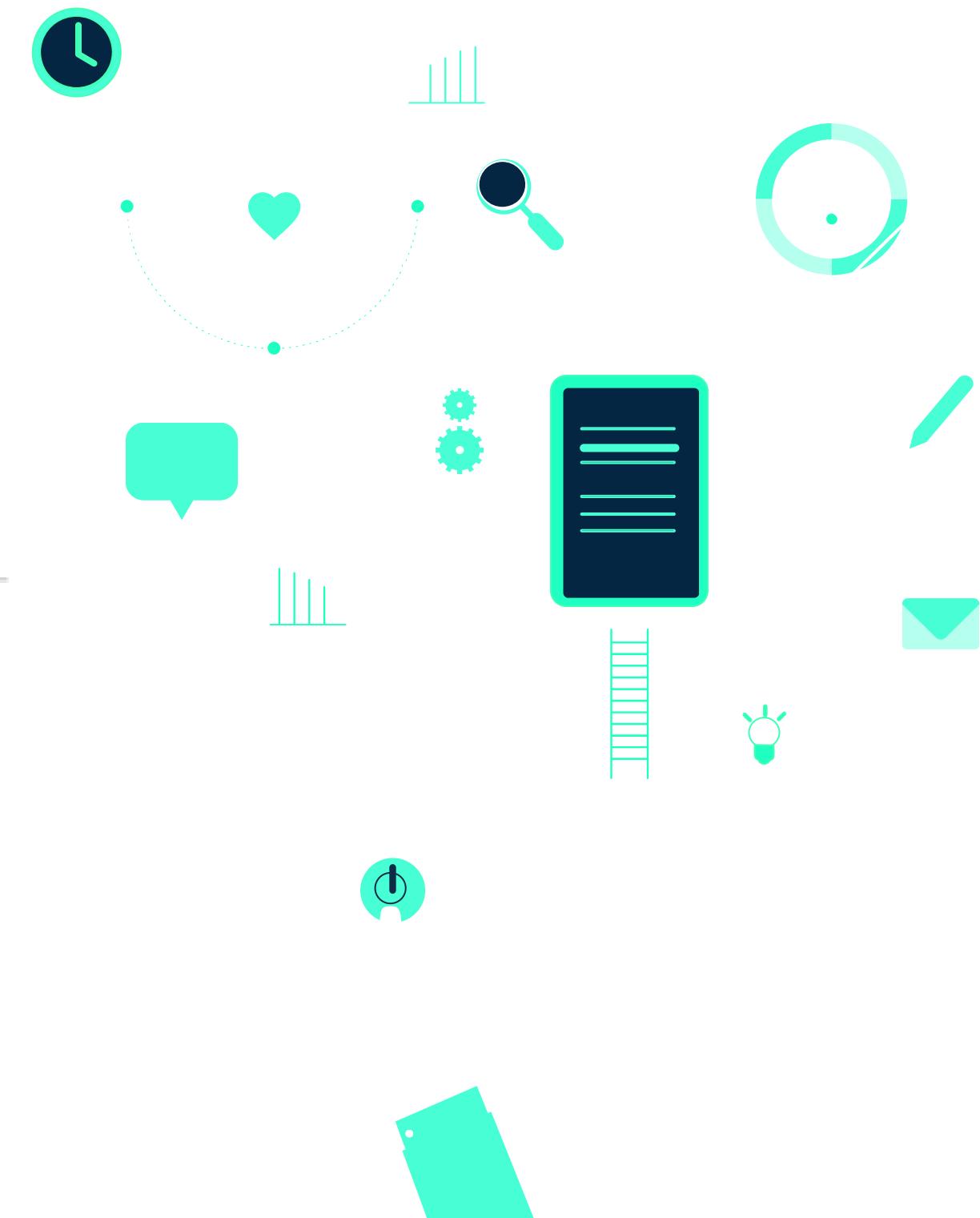
TC10 Aggiunta riuscita

Test frames: FNP1, IDP1, FID1, PP1,  
QP1, FIM1, LD1

	Ingegneria del Software	Pagina 13 di 40
--	-------------------------	-----------------

INPUT	
Destinazione	Parigi
ID Viaggio	CF1856

Prezzo	199.99
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	120
Data Partenza	15/03/2025
Data Ritorno	20/03/2025
Descrizione	Volo diretto da Napoli a Parigi con bagaglio incluso e servizi standard a bordo.
ORACOLO	
<i>Il viaggio viene inserito correttamente nel sistema e reso disponibile nel catalogo di ClickFly.</i>	



## ORACOLO

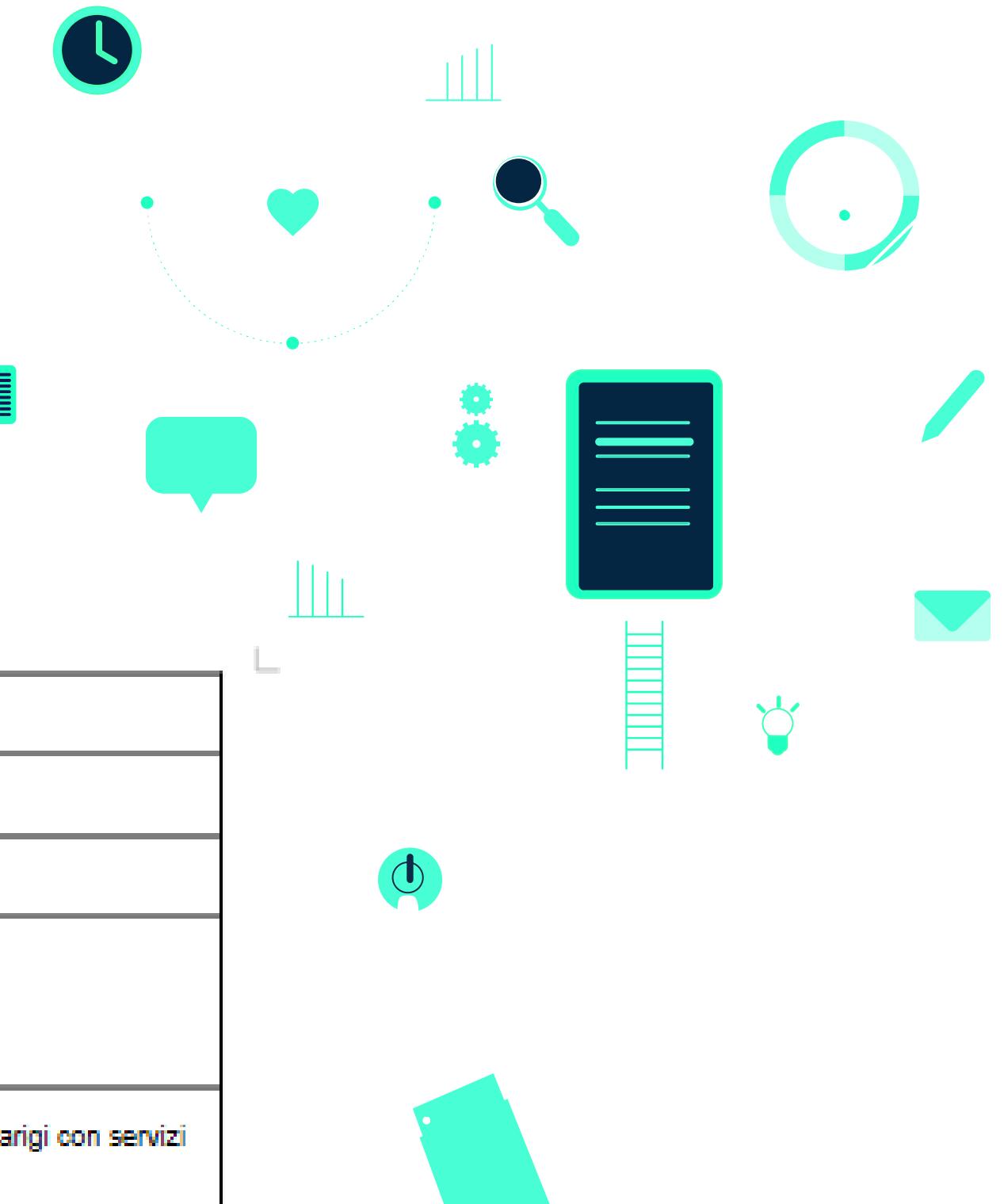
Il viaggio viene inserito correttamente nel sistema e reso disponibile nel catalogo di ClickFly.

TC11 Nome destinazione non valido	Test frames: FNP2, IDP1, FID1, PPI, QP1, FIM1, LD1
<b>INPUT</b>	
Destinazione	Parigi
ID	CF1656
Prezzo	200

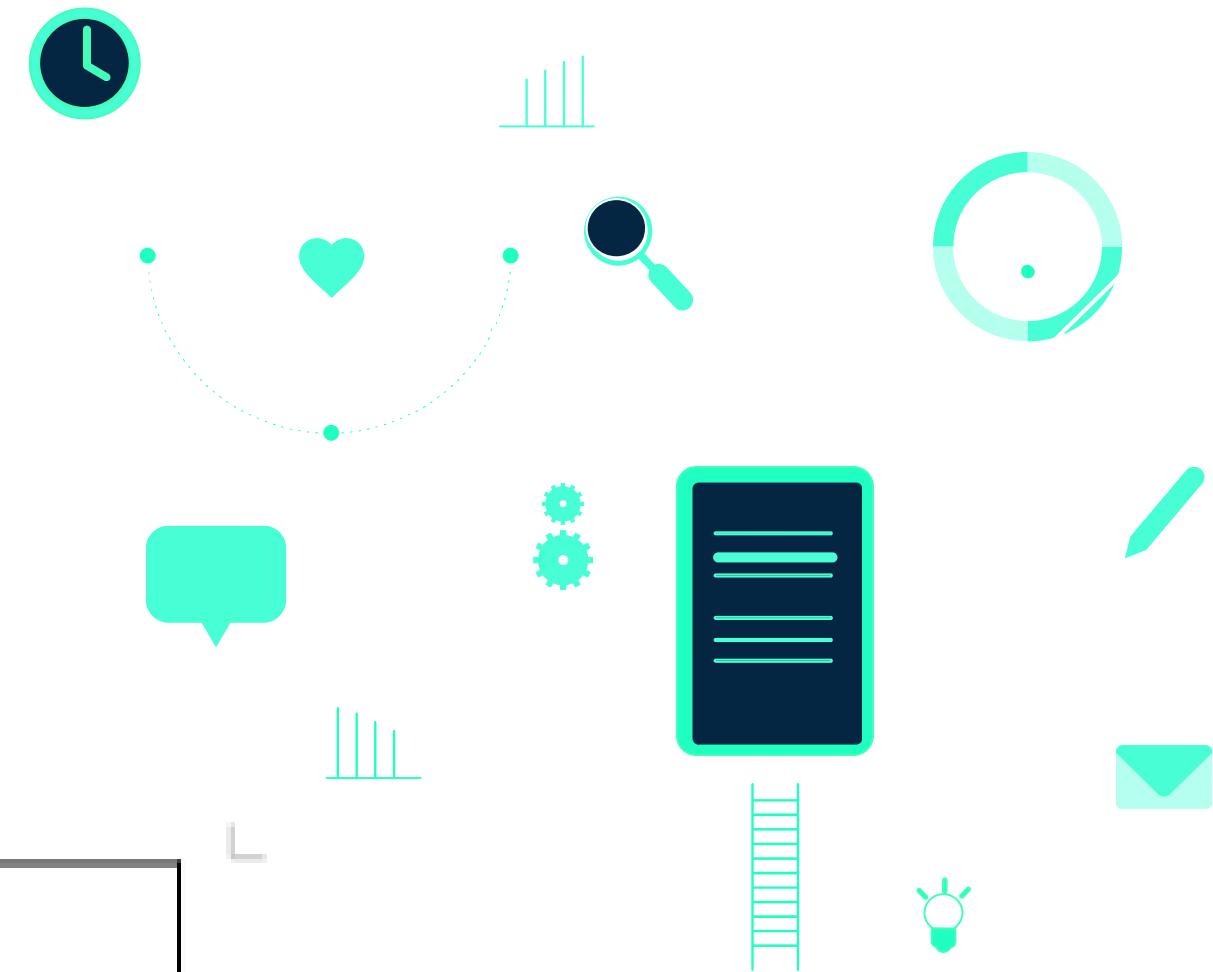
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	120
Documento Allegato	Itinerario.pdf
Descrizione	Volo diretto da Napoli a Parigi con servizi standard a bordo.

## ORACOLO

L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore:  
"Formato destinazione non valido."



TC12   ID presente nel database	Test frames: FNP1, IDP2, FID1, PP1, QP1, FIM1, LD1
<b>INPUT</b>	
Nome viaggio	Volo Napoli – Parigi
ID Viaggio	10
Prezzo	200



Compagnia Aerea	Air France
Aeroporto di partenza	Napoli Capodichino
Numero posti disponibili	10
Documento allegato	itinerario.pdf
Descrizione	Volo diretto da Napoli a Parigi con servizi standard inclusi.
<b>ORACOLO</b>	
<p>Aggiunta del viaggio non completata con successo e il sistema restituisce il seguente messaggio d'errore: <b>"ID del viaggio già presente nel database e associato ad un altro viaggio."</b></p>	

TC13 Formato ID non corretto

Test frames: FNP1, IDP1, RID2, PP1,  
QP1, FIM1, LD1

INPUT

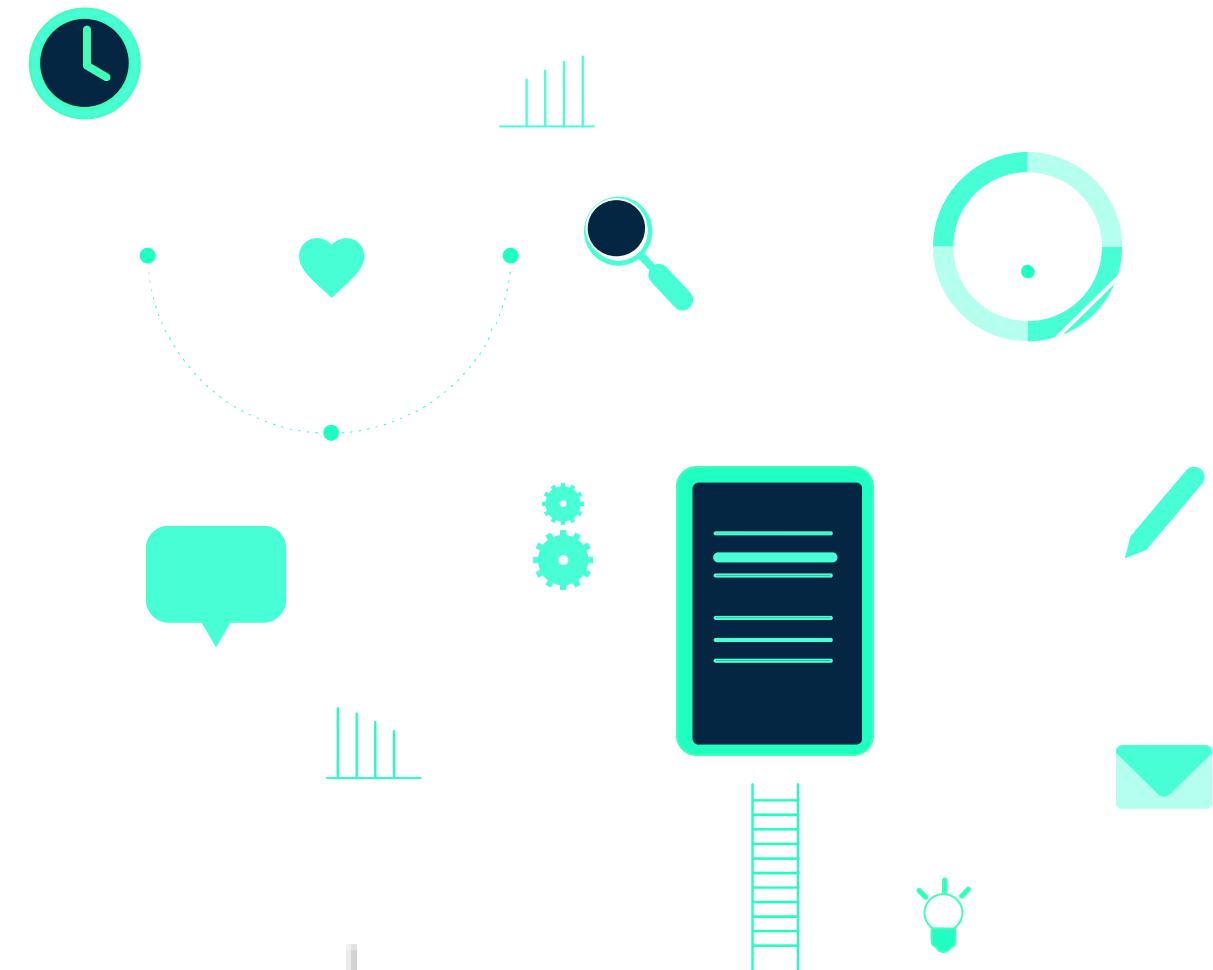
Nome viaggio Volo Napoli – Parigi

Id Viaggio abcd

Prezzo 200

Compagnia aerea Air France

Aeroporto di partenza Napoli Capodichino



Posti disponibili

10

Immagine del viaggio

volo\_roma\_parigi.jpg

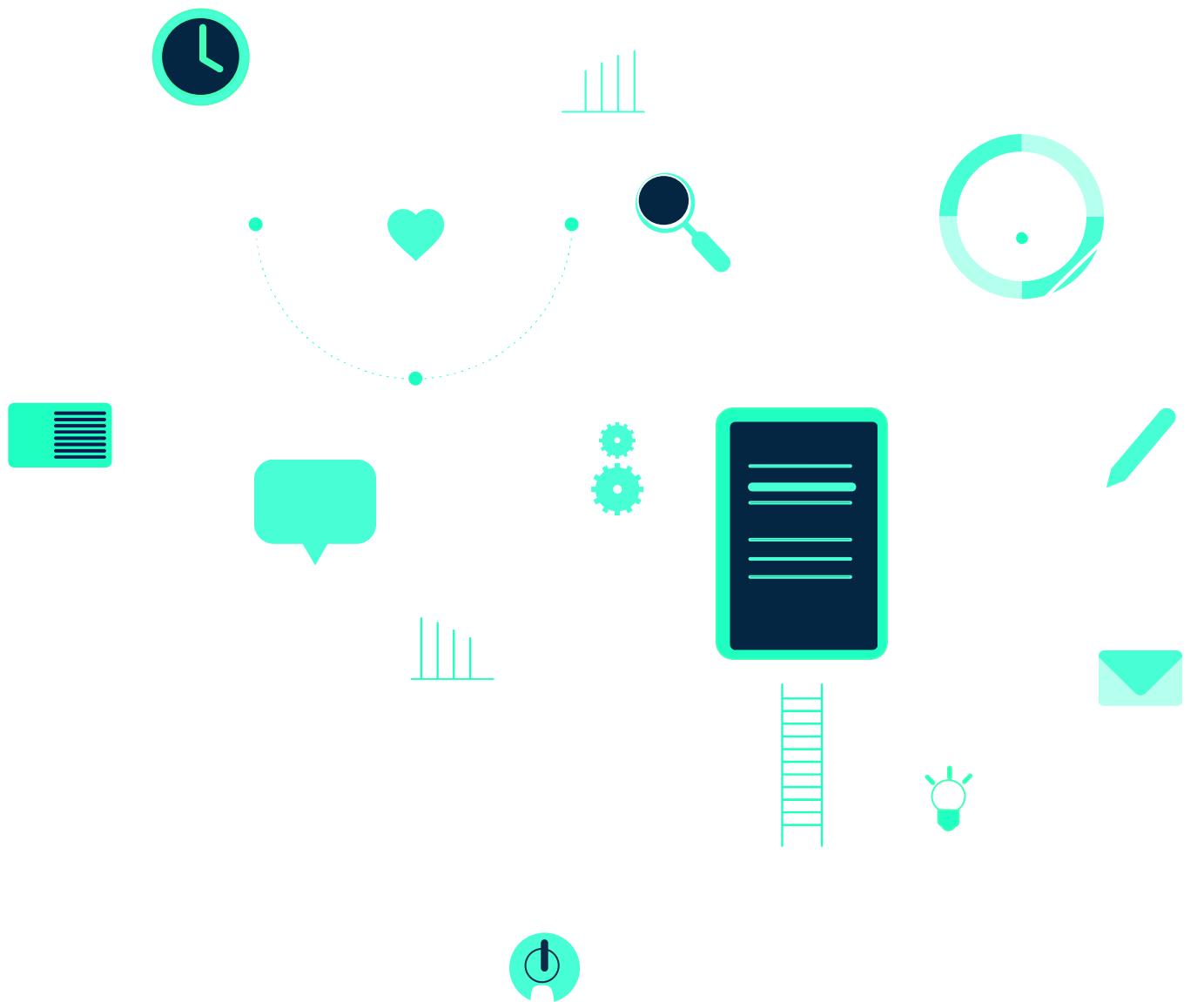
Descrizione

Volo diretto da Roma a Parigi con durata  
di circa 2 ore, comprensivo di bagaglio a  
mano

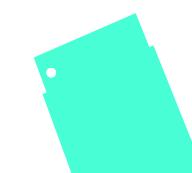
ORACOLO

*Aggiunta del viaggio non completata con successo e il sistema visualizza il  
seguente messaggio d'errore:  
"Il formato dell'ID del viaggio non può contenere caratteri alfabetici."*

TC12 ID Viaggio già presente nel database	Test frames: FNP1, IDP1, FID1, PP2, QP1, FIM1, LD1
<b>INPUT</b>	
Destinazione	Parigi
Id viaggio	10
Prezzo	200
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	10
Documento Allegato	itinerario.pdf
Descrizione	Volo diretto da Napoli a Parigi con servizi standard a bordo.
<b>ORACOLO</b>	
<p>L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore: "ID del viaggio già presente nel database e associato a un altro viaggio."</p>	



TC15 Quantità non valida	Test frames: FNP1, IDP1, FID1, PP1, QP2, FIM1, LD1
<b>INPUT</b>	
Numero di posti non valido	FD1, IDV1, FIDV1, PP1, NP2, FAL1, LD1



Destinazione	Parigi
ID Viaggio	CF1656
Prezzo	200
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	0
Descrizione	Volo diretto da Napoli a Parigi con servizi standard a bordo.
Documento Allegato	itinerario.pdf

### ORACOLO

L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore:  
"Il numero di posti disponibili non può essere nullo o negativo."

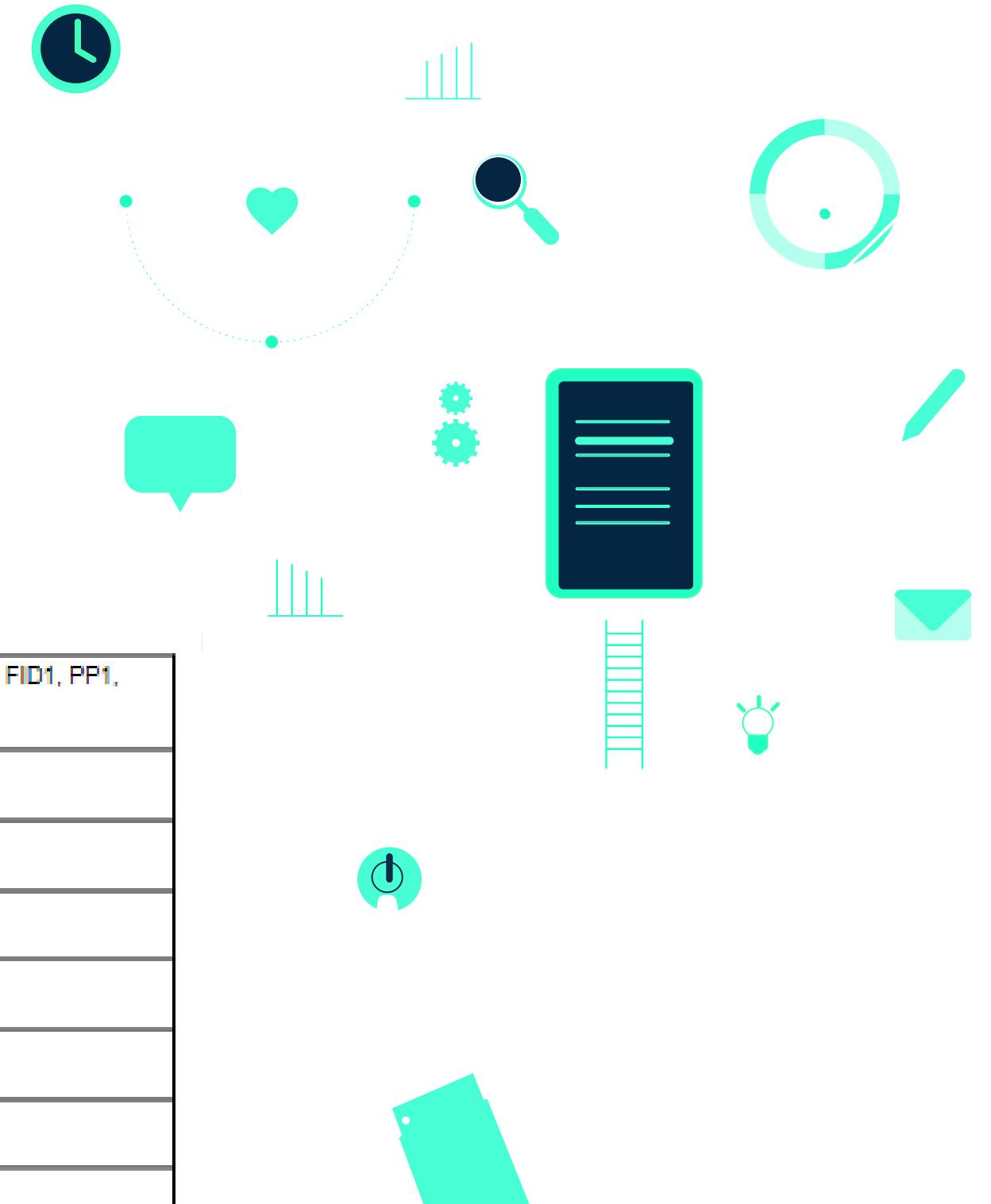
TC16 Formato immagine non supportato  
Test frames: FNP1, IDP1, FID1, PP1,  
QP1, FIM2, LD1

### INPUT

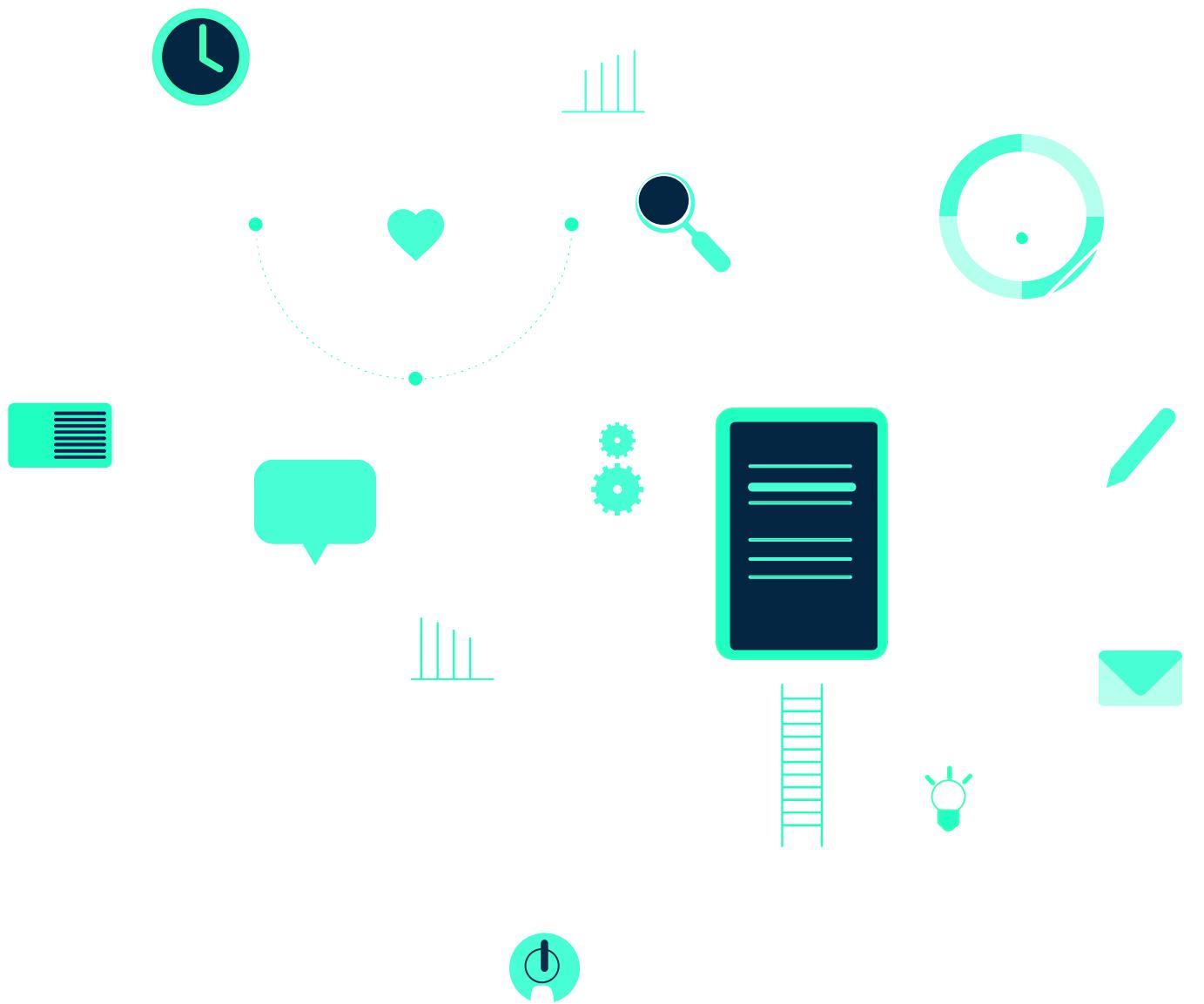
Destinazione	Parigi
Id viaggio	CF1656
Prezzo	200
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	10
Documento Allegato	Programma.exe
Descrizione	Volo diretto da Napoli a Parigi con servizi standard a bordo.

### ORACOLO

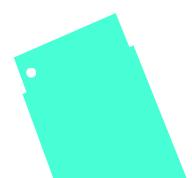
L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore:  
"Il file allegato non è di un formato supportato."



TC16 Formato immagine non supportato	Test frames: FNP1, IDP1, FID1, PP1, QP1, FIM2, LD1
<b>INPUT</b>	
Destinazione	Parigi
Id viaggio	CF1858
Prezzo	200
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	10
Documento Allegato	Programma.exe
Descrizione	Volo diretto da Napoli a Parigi con servizi standard a bordo.
<b>ORACOLO</b>	
<p>L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore:  <b>"Il file allegato non è di un formato supportato."</b></p>	



TC17 Descrizione non valida	Test frames: FNP1, IDP1, FID1, PP1, QP1, FIM1, LD2
<b>INPUT</b>	
Destinazione	Parigi



Id viaggio	CF1656
Prezzo	200
Compagnia Aerea	Air France
Aeroporto di Partenza	Napoli Capodichino
Numero Posti Disponibili	10
Documento Allegato	itinerario.pdf
Descrizione	(campo descrizione assente)
<b>ORACOLO</b>	
<p>L'inserimento del viaggio non viene completato con successo e il sistema restituisce un messaggio di errore: "Descrizione assente o superiore ai 256 caratteri."</p>	

