**ITERAZIONE 2**

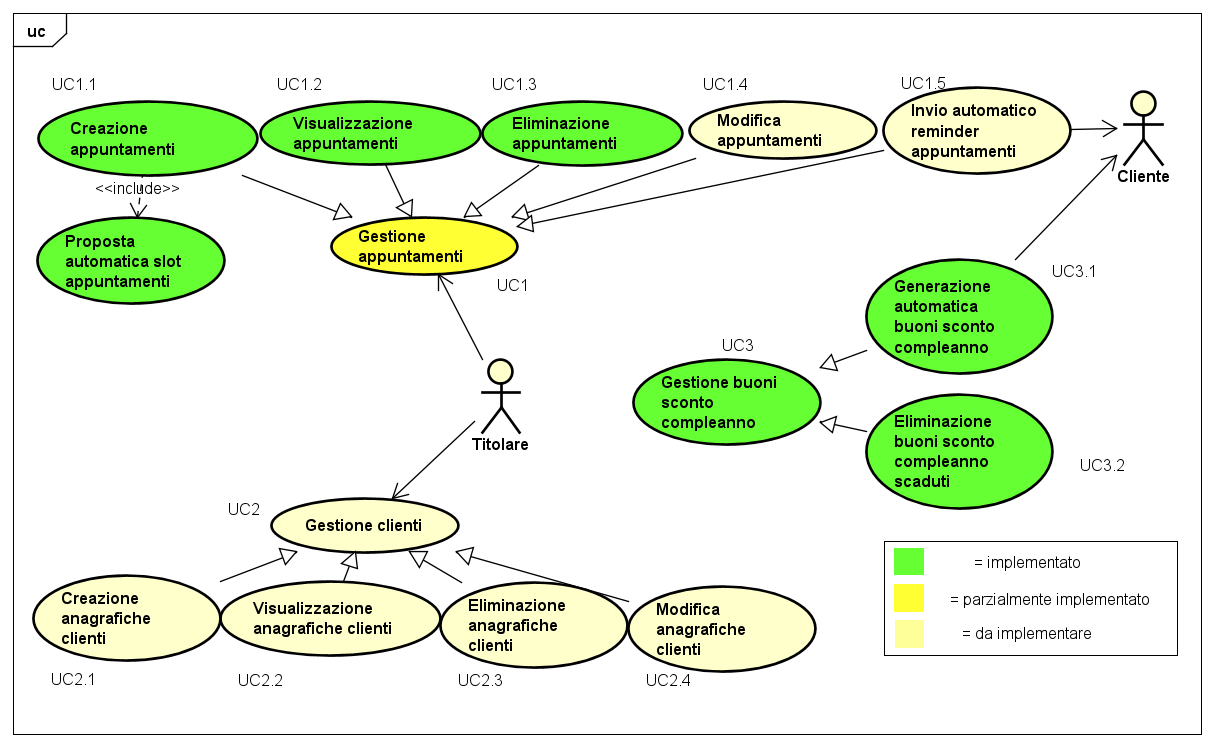
**INTRODUZIONE**

Nella seconda iterazione si è deciso di implementare i seguenti casi d’uso:

* UC1: Gestione appuntamenti
* UC1.1: Creazione appuntamenti;
* UC1.2: Visualizzazione appuntamenti;
* UC1.3: Eliminazione appuntamenti;
* UC3: Gestione buoni sconto compleanno
* UC3.1: Generazione automatica buoni sconto compleanno;
* UC3.2: Eliminazione automatica buoni sconto compleanno scaduti;

Dal diagramma sottostante è possibile analizzare l’evolversi dello stato del progetto rispetto all’iterazione precedente, in particolare si può notare i nuovi casi d'uso implementati di

colore verde, mentre la restante parte non evidenziata rispecchia l’architettura da implementare nella prossima iterazione.



**UC1.1: CREAZIONE APPUNTAMENTI**

*Descrizione*: il titolare deve poter essere in grado di creare nuovi appuntamenti fissati con i clienti: questo comprende avere la possibilità di inserire i dati relativi non solo al cliente ma anche all’appuntamento stesso.

È fondamentale conoscere l’orario di inizio e la tipologia dei trattamenti svolti, al fine di stimare la durata di ogni appuntamento per verificare la disponibilità di eventuali slot temporali in cui inserire il nuovo cliente.

*Attori coinvolti*: titolare, trigger temporale.

**UC1.2: visualizzazione APPUNTAMENTI**

*Descrizione*: il titolare deve poter essere in grado di visualizzare tutti gli appuntamenti: questo comprende avere la possibilità di consultare giorno per giorno il calendario per verificare il numero di clienti previsti per quel determinato giorno, con lo scopo di gestire al meglio le risorse **(sia materiali sia umane)** del saloon, nonché facilitare la verifica di eventuale disponibilità in caso arrivi un nuovo cliente.

*Attori coinvolti*: titolare, trigger temporale.

**UC1.3: eliminazione APPUNTAMENTI**

*Descrizione*: il titolare deve poter essere in grado di eliminare appuntamenti fissati in precedenza con i clienti: questo comprende avere la possibilità di cancellare da calendario un determinato appuntamento e liberare quindi uno slot temporale che potrebbe essere assegnato ad un nuovo cliente.

*Attori coinvolti*: titolare, trigger temporale.

**UC3.1: generazione automatica buoni sconto compleanno**

*Descrizione*: il sistema deve generare automaticamente buoni sconto da inviare in occasione dei compleanni dei clienti.

Sulla base dei dati raccolti relativi ai clienti, il sistema verifica che il giorno attuale corrisponda o meno al compleanno di un cliente.

Nel caso in cui il riscontro avesse avuto successo, a questa data viene aggiunta una durata di 2 mesi che andrà a definire la scadenza del buono, entro la quale il cliente potrà recarsi al saloon per approfittare dello sconto.

Oltre alla scadenza, per ogni buono sconto generato, viene creato anche un codice univoco.

*Attori coinvolti*: trigger temporale.

**UC3.2: eliminazione automatica buoni sconto compleanno scaduti**

*Descrizione*: il sistema, all’accensione, deve eliminare eventuali buoni sconto scaduti.

Il sistema processa tutti i buoni sconto generati fino a quel momento e per ognuno ne verifica la validità andando a controllare se la scadenza precede o meno la data odierna.

Nel momento in cui un buono risulta essere scaduto viene automaticamente rimosso dal sistema.

*Attori coinvolti*: trigger temporale.

**PSEUDOCODICE**

// il seguente algoritmo dovrà essere eseguito una sola volta al giorno

gestioneBuoniSconto(BuonoSconto[] buoniSconto, Cliente[] clienti) void

for each BuonoSconto bS in buoniSconto

if bS.scadenza < adesso

buoniSconto.remove(bS) // supponiamo che questo metodo esista

DateTime nuovaScadenza adesso.aggiungiMesi(2) // supponiamo che questo metodo esista

for each Cliente c in clienti

if c.compleanno = adesso.giorno + "/" + adesso.mese // stringa in formato dd/mm

boolean codiceDuplicato true

while codiceDuplicato

int nuovoCodice random\_int(0, 999999) // supponendo che questa funzione esista (i parametri passati sono i due estremi dell'intervallo di generazione)

codiceDuplicato false

for each BuonoSconto bS in buoniSconto

if bS.codice = nuovoCodice

codicePresente true

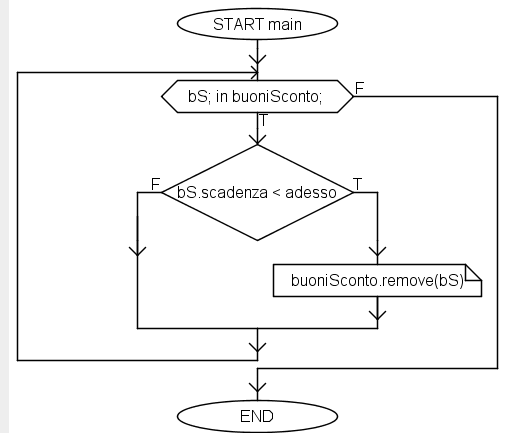
buoniSconto.add(new BuonoSconto(nuovoCodice, nuovaScadenza, c))

//supponiamo che questo metodo esista e che il costruttore di BuonoSconto sia definito

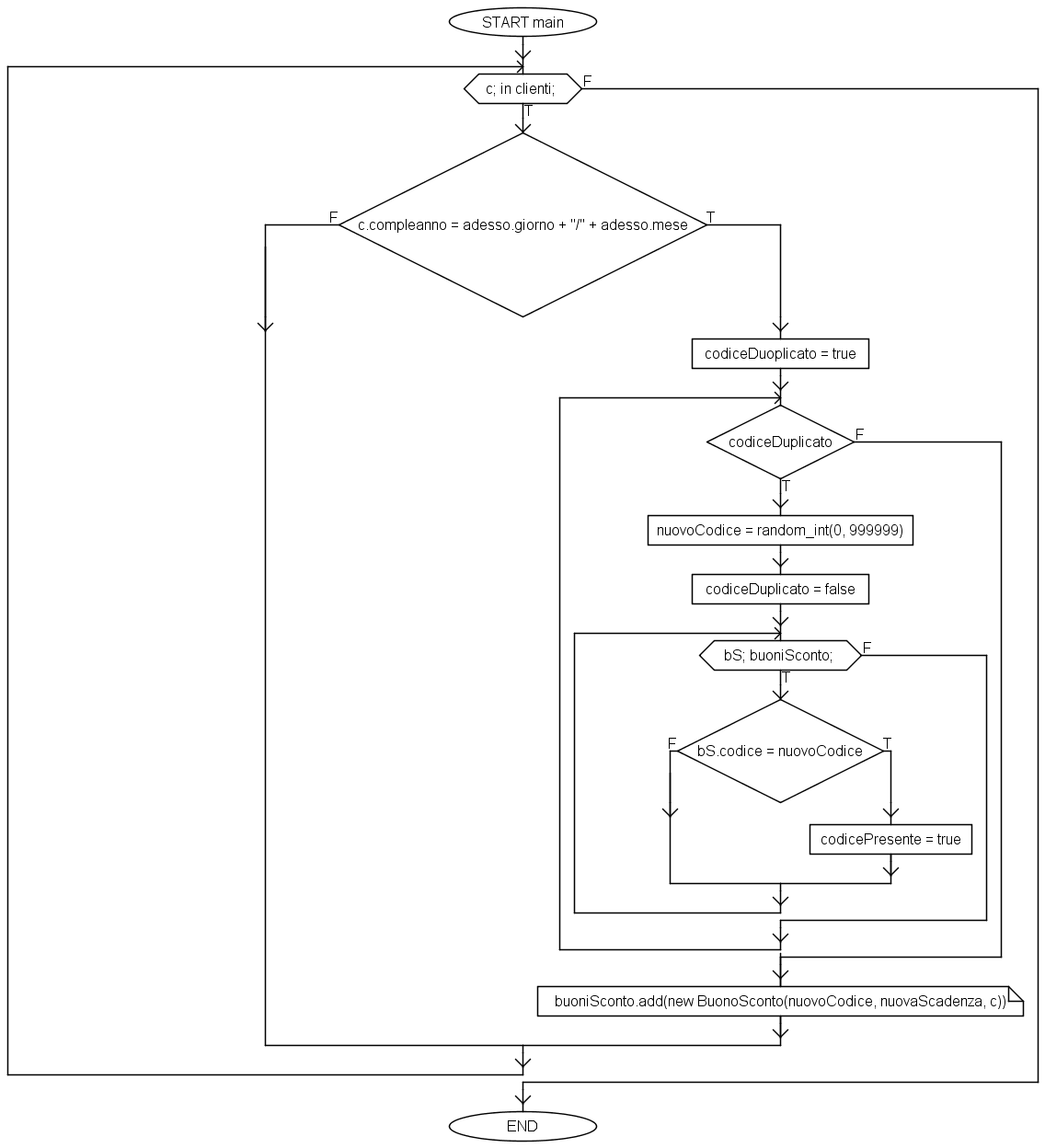
**Flowchart**

Per maggior chiarezza sono stati realizzati anche i flowchart delle parti algoritmiche implementate.

Eliminazione buono sconto:



Creazione buono sconto:



**TESTING**

Per l’iterazione 2 è stato effettuato il test delle funzioni descritte nei nuovi casi d’uso implementati.

I risultati sono riportati di seguito:

**SPAZIO PER IMMAGINI TESTING**

**DOCUMENTATON API**

**COSA METTIAMO????????**