

Esame di Ingegneria del Software – 10 CFU – 19 aprile 2024

- Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola.
- Non è consentito consultare appunti, libri, colleghi, né qualunque dispositivo elettronico, pena l'immediato annullamento della prova.
- **L'Esercizio 1, gli esercizi Modulo A e gli Esercizi Modulo B devono essere svolti su fogli differenti.**
- Tempo a disposizione: 3 ore.

Si vuole realizzare un sistema per la gestione di prestiti di libri. L'obiettivo è fornire agli utenti una piattaforma intuitiva e conveniente per trovare, richiedere e prendere in prestito libri disponibili in una biblioteca. Gli utenti avranno la possibilità di registrarsi al sito inserendo le proprie informazioni personali come nome, cognome, e-mail e password. Ogni utente avrà un profilo univoco nel sistema. Una volta registrati, gli utenti potranno cercare libri nel catalogo inserendo il titolo, l'autore o il genere desiderato. Il sistema mostrerà quindi i dettagli dei libri trovati, inclusi titolo, autore, anno di pubblicazione e stato attuale (disponibile o in prestito). Gli utenti avranno la possibilità di richiedere il prestito di un libro disponibile. Una volta selezionato il libro di cui richiedere il prestito, l'utente potrà procedere con la richiesta di prestito, che sarà completata una volta effettuato il pagamento del contributo fisso di € 3,00. Per il pagamento, l'utente deve inserire il numero della propria carta di credito, con intestatario della stessa, scadenza, e codice CCV. Le informazioni vengono sul pagamento vengono inviate al servizio esterno UninaPay, che processa il pagamento. Se il pagamento va a buon fine, l'utente visualizza il codice di conferma della sua prenotazione, e potrà successivamente recarsi in biblioteca per ritirare il libro richiesto. In caso di problemi con il pagamento, viene invece mostrato un messaggio di errore.

Esercizio 1

- (a) Si modellino **tutti** i requisiti del sistema descritto sopra utilizzando uno Use Case Diagram;
- (b) Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di richiesta di un prestito.
- (c) Dettagliare il caso d'uso relativo alla funzionalità di richiesta di un prestito, per mezzo descrizioni testuali strutturate secondo il formalismo di Cockburn. Usare la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.
- (d) A partire dai mock-up definiti al punto (b), realizzare uno statechart per modellare il funzionamento dell'interfaccia grafica.

Modulo A - Esercizio 2A

L'azienda *GlobalSport* ha intenzione di lanciare un nuovo servizio (*SportySpotlight*) che permette agli utenti, previa autenticazione, di visualizzare lo stato di diverse competizioni e eventi sportivi (automobilismo, ciclismo, calcio, tennis, basket, pallavolo, etc...), e commentare ciascun evento interagendo con altri utenti.

Da una prima analisi effettuata dagli esperti di Marketing, è emersa l'esistenza di una significativa potenziale base di utenti distribuita in tutto il mondo e concentrata in Europa, Stati Uniti, e Sud America. L'analisi ha evidenziato anche che ci si aspetta una distribuzione del carico del sistema altamente irregolare, con picchi di traffico elevatissimi raggiunti in corrispondenza delle principali competizioni (e.g.: finale dei 100m alle Olimpiadi, finale dei mondiali di calcio, etc.), e periodi di scarso utilizzo (quando non si svolge alcuna competizione di rilievo).

In qualità di Software Architect della *GlobalSport*, vi è richiesto di redigere una relazione di al più una pagina in cui si espone al *Chief Technology Officer (CTO)* una descrizione delle soluzioni architetturali possibili per la messa in opera del sistema *SportySpotlight*. Le soluzioni possibili vanno confrontate rispetto a criteri di affidabilità, tolleranza ai guasti e possibilità di garantire prestazioni adeguate anche durante i picchi di carico, minimizzando i costi operativi, nello scenario prospettato dagli esperti di Marketing.

Modulo A – Esercizio 3A

```
1 public static boolean isPalindrome(int num) {  
2     if (num < 0) {  
3         throw new IllegalArgumentException("Negative numbers are not accepted!");  
4     }  
5     int original = num;  
6     int reverse = 0;  
7     while (true) {  
8         if(num==0){  
9             break;  
10        }  
11        int digit = num % 10;  
12        reverse = reverse * 10 + digit;  
13        num /= 10;  
14    }  
15    return original == reverse;  
16 }
```

Un numero è palindromico se può essere letto allo stesso modo sia da sinistra a destra che da destra a sinistra. Ad esempio, 121, 12321 e 4884 sono numeri palindromici.

1. Rappresentare il CFG del metodo isPalindrome;
2. Scrivere tre test JUnit con strategia White Box per il metodo isPalindrome, indicando per ciascuno di essi quale cammino copre nel CFG. Si richiede che almeno un test copra uno scenario di errore (input non valido) e che, ove possibile, i test JUnit coprano cammini distinti nel CFG;
3. Qual è la Test Effectiveness Ratio (TER), relativamente alla copertura di nodi del CFG, della suite di tre test sviluppata al punto (2)? Motivare la risposta.

Modulo B - Esercizio 3B

In riferimento alla traccia dell'esercizio di modulo A, allo scopo di valutare l'usabilità a posteriori del sistema proposto, lo studente identifichi un test di scenario da proporre a 5 utenti e simuli le risposte e le performances osservate. Lo studente definisca una metrica per quantificare l'usabilità.

Esame di Ingegneria del Software – 10 CFU – 23 gennaio 2024

- Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola.
- Non è consentito consultare appunti, libri, colleghi, né qualunque dispositivo elettronico, pena l'immediato annullamento della prova.
- **L'Esercizio 1, gli esercizi Modulo A e gli Esercizi Modulo B devono essere svolti su fogli differenti.**
- Tempo a disposizione: 3 ore.

Un'azienda di trasporto ferroviario vi commissiona la realizzazione di un sistema informativo per la gestione di avvisi. Gli addetti dell'azienda, previa autenticazione, possono inserire nuovi avvisi. Ciascun avviso è caratterizzato da un titolo, da una descrizione in testo libero, dalla data di pubblicazione, ed è associato ad una o più delle linee ferroviarie gestite dall'azienda. Gli avvisi possono inoltre essere di diverso tipo: avvisi di sciopero (che riportano anche la data e le fasce orarie per cui è indetto lo sciopero), guasti (che riportano anche una data indicativa di quando la circolazione ferroviaria riprenderà correttamente), ritardi (riportanti l'entità media, in minuti, del ritardo sulla linea e una data stimata in cui la circolazione tornerà regolare), oppure semplici informazioni su variazioni del servizio. Quando viene inserito un nuovo avviso, il sistema invia automaticamente, utilizzando le API del servizio di messagistica Unigram, un messaggio contenente i dettagli dell'avviso all'interno del canale pubblico dedicato agli avvisi. I viaggiatori possono visualizzare l'elenco degli avvisi dal più recente al meno recente, filtrando eventualmente soltanto gli avvisi relativi alle linee ferroviarie di proprio interesse. I viaggiatori, inoltre, possono inviare segnalazioni di guasti e/o disservizi, che verranno successivamente visualizzate dagli addetti dell'azienda.

Esercizio 1

- Si modellino tutti i requisiti del sistema descritto sopra utilizzando un Use Case Diagram;
- Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di inserimento di un nuovo avviso.
- Dettagliare il caso d'uso relativo alla funzionalità di inserimento di un nuovo avviso, per mezzo descrizioni testuali strutturate secondo il formalismo di Cockburn. Usare la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.
- A partire dai mock-up definiti al punto (b), realizzare uno statechart per modellare il funzionamento dell'interfaccia grafica.

Modulo A - Esercizio 2A

In un videogioco di ruolo, il metodo statico `computePrice` della classe `ShopUtils` viene utilizzato per calcolare il prezzo (espresso in Septim, la valuta di gioco) di un item in vendita in un negozio gestito da un NPC. Il metodo prende in input i seguenti parametri:

- `int itemLevel`: indica il livello dell'oggetto e può assumere valori compresi tra 1 e 99;
- `String rarity`: indica il livello di rarità dell'item ("RARE", "EPIC", oppure "LEGENDARY");
- `boolean isPlayerMerchant`: indica se è il giocatore ha correntemente attivo il tratto "Merchant", che gli permette di avere prezzi più vantaggiosi quando acquista da NPC.

Se i parametri non sono validi, il metodo solleva una `IllegalArgumentException`. In caso contrario, ritorna il prezzo da mostrare al giocatore. Il prezzo è determinato in maniera pseudocasuale all'interno di intervalli determinati come mostrato nella tabella seguente. Gli intervalli sono da considerarsi con estremi inclusi.

	Rare		Epic		Legendary	
	Merchant	Non Merchant	Merchant	Non Merchant	Merchant	Non Merchant
Level < 50	5-10	10-15	100-150	200-250	500-700	800-1000
50 <= Level < 80	100-150	200-250	500-700	800-1000	2500-5000	3000-7000
80 <= Level < 100	500-700	800-1000	2500-5000	3000-7000	10000-20000	15000-30000

Per esempio, qualsiasi valore compreso tra 2500 e 5000 Septim sarebbe valido per un item Legendary di livello 51 da vendere a un giocatore con il tratto *Merchant* attivo.

- Indicare, per ciascuno dei parametri del metodo `getQuote`, le classi di equivalenza individuate.
- Scrivere quattro test JUnit con strategia Black Box per il metodo `computePrice`, indicando per ciascuno di essi quali classi di equivalenza copre. Si richiede inoltre che un test corrisponda a scenari in cui i parametri non sono validi, e che i restanti tre corrispondano a scenari in cui i parametri sono validi.
- Quanti test sono necessari per testare il metodo con strategia N-WECT? Motivare la risposta.

Modulo A - Esercizio 3A

Descrivere vantaggi e svantaggi (qualora ve ne siano) di **alta coesione** e **basso accoppiamento** nel contesto di architetture basate su micro-servizi.

Modulo B - Esercizio 3B

Descrivere i cicli di vita della progettazione dell'interfaccia nelle fasi di ideazione, prototipazione e testing, illustrando in particolare in quali condizioni si attiva l'iterazione a ritroso o quella del passaggio alla fase successiva.

Istruzioni: Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola. Durante la prova, non è consentito consultare materiale didattico, né interagire in alcun modo con dispositivi elettronici o colleghi, pena l'immediato annullamento della prova. Il tempo a disposizione per completare la prova è di **3 ore**.

Esercizio 1

Vi è commissionata la realizzazione di UninAdvisor, un sistema che permette agli studenti di valutare, in maniera anonima, i docenti di un noto ateneo. Il sistema permette ad un utente (non, necessariamente, autenticato) di ricercare un docente per cognome e nome, e di visualizzare tutte le recensioni inserite nel sistema per quel docente. Un utente registrato, invece, può lasciare al più una recensione per ogni singolo docente. Per fare ciò, l'utente indica una valutazione (da 1 a 5 stelle), un titolo, e una breve descrizione (tutte i campi vanno compilati). Quando un utente inserisce una recensione, il sistema esterno ModeratorGPT viene invocato per analizzare il titolo e la descrizione della recensione, e determinare se questa è offensiva verso il docente o meno. Nel caso in cui la recensione sia ritenuta offensiva, il sistema non permette di salvarla, e mostra un messaggio di errore. Le recensioni non offensive, invece, vengono salvate correttamente. Gli utenti registrati, inoltre, possono inserire anche nuovi docenti nel sistema, qualora i docenti che vogliono recensire non fossero già presenti. Nell'inserimento di un nuovo docente, è necessario indicare nome, cognome, corso di studi cui appartiene, e link al sito web istituzionale. Gli amministratori del sito web, inoltre, previa autenticazione, possono accedere a tutte le funzionalità riservate agli utenti registrati, e in aggiunta possono eliminare qualsiasi review o docente, a loro discrezione. Infine, un utente non registrato può segnalare una recensione ritenuta offensiva, e gli amministratori possono visualizzare tutte le segnalazioni.

Si svolgano i seguenti punti, usando la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.

- ✓ 1. Modellare tutti i requisiti funzionali del sistema descritto sopra utilizzando uno Use Case Diagram;
- ✓ 2. Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di invio di una recensione da parte di un utente.
- ✓ 3. Dettagliare il caso d'uso relativo alla funzionalità di invio di una recensione, per mezzo descrizioni testuali strutturate secondo un formalismo tabellare come quello proposto da A. Cockburn.

Esercizio 2

Descrivere brevemente le caratteristiche, la struttura e la finalità degli schemi "Personas". Dopodiché fornire due esempi di Personas con riferimento al sistema descritto nell'Esercizio 1.

Esercizio 3

Il controller di ricarica di uno smartphone supporta due modalità di protezione della batteria: BASE e MASSIMA, che si attivano durante il processo di ricarica, ovvero quando il cavo di ricarica è collegato. Nella modalità di protezione di base, il controller ricarica la batteria fino al 100%. Una volta che la batteria arriva al 100%, la ricarica viene sospesa, salvo riattivarsi quando la batteria scende al di sotto del 95%. Nella modalità di protezione massima, invece, il controller sospende la ricarica della batteria quando il livello di carica raggiunge l'80%, salvo riattivarsi quando il livello di carica scende al di sotto del 75%.

Modellare con uno statechart il comportamento del controller di ricarica sopra descritto.

Esercizio 4

Il metodo statico `calculatePrize` della classe `SwEngGames` viene utilizzato in una università di Seoul per calcolare il montepremi finale da destinare ai singoli studenti che superano tutte le dure prove predisposte nell'ambito di un gioco a premi a tema Ingegneria del Software. Il metodo accetta i seguenti argomenti:

- `int nEliminated`: numero (non negativo) di studenti eliminati durante le prove;
- `int nWinners`: numero (non negativo) di studenti che hanno superato tutte le prove (vincitori);
- `String gameEdition`: stringa non null che indica che tipo di edizione del gioco a premi si sta considerando. Il parametro può assumere i seguenti valori: `"XMAS"` (versione natalizia), `"EASTER"` (versione pasquale), `"DEFAULT"` (versione base). Qualsiasi altra stringa diversa da null deve essere interpretata come `"DEFAULT"`.

Se almeno uno dei parametri è non valido, il metodo solleva una `IllegalArgumentException`. In caso contrario, il metodo ritorna un numero float rappresentante il montepremi (in Won, la valuta ufficiale della Corea del Sud) che sarà destinato a ciascuno studente vincitore. Tale budget è calcolato come descritto di seguito. Innanzitutto, si calcola il budget complessivo, utilizzando la formula:

$$\text{overallBudget} = (\text{nEliminated} \times \text{coeffPrize})$$

Il coefficiente `coeffPrize` viene determinato in base all'edizione del gioco a premi, come da tabella seguente:

coeffPrize		
XMAS	EASTER	DEFAULT
10.000.000	2.000.000	1.000.000

Infine, il budget complessivo viene diviso in parti uguali tra tutti gli studenti che hanno superato tutte le prove. Per esempio, una invocazione del metodo con 10 studenti eliminati durante le prove, 2 studenti che hanno superato le prove e l'edizione del gioco `"DEFAULT"`, risulterà in un budget complessivo di 10 milioni di Won, da suddividere tra due vincitori, portando a un risultato di 5 milioni di Won per studente vincitore.

- ✓ Indicare, per ciascuno dei parametri del metodo `calculatePrize`, un partizionamento in classi di equivalenza per testing black-box;
- ✓ Scrivere quattro test JUnit con strategia Black Box per il metodo `calculatePrize`, indicando per ciascuno di essi quali classi di equivalenza copre. Si richiede inoltre che un test corrisponda a scenari in cui almeno uno dei parametri è non valido, e che i restanti tre corrispondano a scenari in cui tutti i parametri sono validi. Ove possibile, i test devono coprire classi di equivalenza distinte.
- ✓ Quanti test sono necessari per testare il metodo con strategia R-WECT? Motivare la risposta, descrivendo brevemente le diverse strategie.

Esame di Ingegneria del Software – 10 CFU – 13 Giugno 2024

- Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola.
- Non è consentito consultare appunti, libri, colleghi, né qualunque dispositivo elettronico, pena l'immediato annullamento della prova.
- L'Esercizio 1, gli esercizi Modulo A e gli Esercizi Modulo B devono essere svolti su fogli differenti.
- Tempo a disposizione: 3 ore.

Si richiede la realizzazione di un sistema per gestire corsi di formazione organizzati da dipendenti per dipendenti, in contesti aziendali. Il sistema deve permettere ai dipendenti di creare un nuovo corso di formazione, specificando la tipologia del corso (online o in presenza), una breve descrizione del corso e le date previste di inizio e fine. I corsi online sono caratterizzati da un URL a cui collegarsi per seguire le lezioni, mentre quelli in presenza dal codice di una stanza in cui il corso si svolge. Il dipendente deve anche indicare il formatore responsabile del corso, che può essere scelto tra i vari dipendenti dell'azienda o può essere il creatore stesso del corso. Facoltativamente, il dipendente può indicare fino a due assistenti formatori, selezionabili tra i dipendenti dell'azienda. Il sistema deve inoltre permettere ai dipendenti di visualizzare i corsi disponibili e richiedere l'iscrizione a uno di essi. I responsabili del corso possono visualizzare le richieste ricevute e decidere se accettarle o respingerle. In caso di approvazione, il sistema deve registrare la data in cui la richiesta è stata approvata. In caso di rifiuto, i responsabili del corso devono fornire una breve descrizione delle motivazioni del rifiuto. Sia in caso di accettazione che di rifiuto, il sistema deve inviare una notifica via e-mail, utilizzando le API del servizio esterno "EasyMail", a tutti gli interessati: il dipendente richiedente, i formatori responsabili e gli eventuali assistenti formatori.

Esercizio 1

- ☒ Si modellino tutti i requisiti del sistema descritto sopra utilizzando uno Use Case Diagram; MOD A
- ☒ Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di creazione di un corso di formazione. MOD B
- ☒ Dettagliare il caso d'uso relativo alla funzionalità di richiesta d'iscrizione ad un corso formativo, per mezzo descrizioni testuali strutturate secondo il formalismo di Cockburn. Usare la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti. MOD A
- ☒ A partire dai mock-up definiti al punto (b), realizzare uno statechart per modellare il funzionamento dell'interfaccia grafica. MOD B

Modulo A - Esercizio 2A

L'azienda presso cui lavorate ha deciso di migrare il proprio sistema di gestione documentale in cloud. Attualmente, l'azienda utilizza un server fisico in loco che gestisce i documenti aziendali e supporta 200 utenti. Questo server funziona 24 ore al giorno, 7 giorni su 7, e comporta un costo di manutenzione mensile di 2000€, oltre ai costi di energia elettrica stimati di 500€ al mese.

La consulenza effettuata precedentemente ha evidenziato che, per supportare lo stesso carico di lavoro in cloud, l'azienda avrebbe bisogno di 5 macchine virtuali con le seguenti caratteristiche:

- Le macchine virtuali devono essere sempre attive per garantire la disponibilità continua dei servizi.
- Ogni macchina virtuale costa 0,10€/ora.

L'azienda ha anche considerato di adottare un modello ibrido, in cui i dati meno critici sono archiviati in 3 macchine virtuali che possono essere accese solo durante l'orario di lavoro (dalle 8:00 alle 18:00, dal lunedì al venerdì), mentre i dati critici rimangono su 2 macchine virtuali sempre attive. Dare una stima del costo mensile per mantenere tutte le macchine virtuali sempre attive e una stima nel caso si scelga di adottare un modello ibrido. Inoltre, descrivere i vantaggi e gli svantaggi (se ve ne sono) di migrare il sistema di gestione documentale in cloud rispetto a mantenere il server fisico in loco, considerando anche aspetti organizzativi e di sicurezza.

Modulo A - Esercizio 3A

La classe `StreamingService` ha un metodo `calculateSubscription` che viene utilizzato per calcolare il prezzo mensile da offrire ad un cliente per accedere a un servizio di streaming. Il metodo prende in input i seguenti parametri:

- `Int maxDevice`: indica il numero massimo di dispositivi che possono essere utilizzati contemporaneamente.
- `String quality`: indica la qualità dello streaming richiesta dal cliente ("SD", "HD", "UHD").
- `Boolean familyPlan`: indica se il pacchetto include un piano famiglia, che permette di creare profili separati per ogni membro della famiglia.

Se i parametri non sono validi, il metodo solleva un `IllegalArgumentException`. In caso contrario, ritorna il prezzo mensile da proporre al cliente. Il prezzo mensile di base è calcolato moltiplicando il numero di dispositivi per il costo mensile per dispositivo determinato in base alla qualità dello streaming richiesta, come da tabella seguente. Inoltre, se è incluso il piano famiglia, la somma di 10.00 € va aggiunta al totale.

Qualità dello streaming	Costo per dispositivo (€)
SD	5
HD	10
UHD	15

- ☒ Indicare, per ciascuno dei parametri del metodo `calculateSubscription`, le classi di equivalenza individuate.
- ☒ Scrivere quattro test JUnit con strategia Black Box per il metodo `calculateSubscription`, indicando per ciascuno di essi quali classi di equivalenza copre. Si richiede inoltre che un test corrisponda a scenari in cui i parametri non sono validi, e che i restanti tre corrispondano a scenari in cui i parametri sono validi.
- ☒ Quanti test sono necessari per testare il metodo con strategia WECT? Motivare la risposta.

Modulo B - Esercizio 3B

Per verificare l'usabilità di una specifica applicazione, i progettisti hanno definito un test di compito articolato in 5 task da assegnare a 15 utenti. Gli utenti sono divisi in tre gruppi, 1) esperti di dominio, 2) esperti utilizzatori di sistemi informatici, ma estranei al dominio, 3) persone sia esperte del dominio che dell'uso delle tecnologie.

Invece di ricorrere al semplice criterio "pass", "fail" o "successo parziale", visto che tutti gli utenti hanno ottenuto il valore "pass", i progettisti hanno deciso di valutare: a) il tempo necessario per completare ogni task, b) il numero di tocchi (o di click) sullo schermo, c) il numero di richieste di chiarimenti rivolte al valutatore osservatore. Lo studente, illustrando con dettagli ogni passaggio, definisca una metrica per integrare questi parametri in un'unica valutazione e provi a fare una previsione delle performance relativamente al confronto fra i tre gruppi di utenti. Lo studente, inoltre, indichi quali altri parametri si sarebbero potuti prendere in considerazione, in alternativa a quelli proposti.

Esercizio 1

Vi è commissionata la realizzazione di **FantaInformatica**, un sistema che permette agli studenti di creare una propria squadra virtuale di docenti e competere in una lega basata su eventi legati alla didattica. Ciascuna lega (o campionato) dura per un intero anno accademico.

Il sistema consente a qualsiasi utente (anche non autenticato) di visualizzare la classifica generale e l'albo d'oro con i vincitori delle passate edizioni. Un utente registrato, invece, può creare una squadra selezionando tre docenti diversi tra quelli disponibili nel sistema. Per creare una squadra, è necessario anche indicare un capitano tra i docenti selezionati, e un nome (univoco) della squadra. Le squadre create possono quindi essere successivamente iscritte ad un campionato, prima dell'inizio di ciascun anno accademico.

Durante l'anno accademico, le squadre iscritte guadagnano punti in base al verificarsi di particolari eventi, come ad esempio "*Nessun bocciato ad una prova scritta*: +10 punti", "*Compiti corretti entro un giorno*: +10 punti", "*Spostamento di una lezione su richiesta degli studenti*: +7 punti", "*Mancata correzione di una prova scritta entro la data promessa*: -5 punti", "*Ritardo di più di 5 minuti a una lezione*: -10 punti", etc... Gli eventi attribuiti al capitano di ciascuna squadra valgono un punteggio doppio.

Gli amministratori del sistema possono invece gestire la lista dei docenti disponibili, modificare il regolamento di un campionato (personalizzando eventi e punteggi relativi), eliminare squadre che violano le regole (e.g.: che contengono nomi offensivi) e validare gli eventi, indicando se si sono verificati o meno in una certa data.

Si svolgano i seguenti punti, usando la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.

- ✗ Modellare tutti i requisiti funzionali del sistema descritto sopra utilizzando uno Use Case Diagram;
- ✗ Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di creazione di una nuova squadra.
- ✗ Dettagliare il caso d'uso relativo alla funzionalità di creazione di una squadra, per mezzo descrizioni testuali strutturate secondo un formalismo tabellare come quello proposto da A. Cockburn.

Esercizio 2

Che cosa sono le *affordances* nel contesto dell'interazione uomo-macchina? In che modo influenzano il design delle interfacce utente e che impatto hanno sull'usabilità di una interfaccia utente? Fornire un esempio concreto di *affordance*, relativo a un sistema come quello descritto in Esercizio 1.

Esercizio 3

Un termostato intelligente regola la temperatura di un'aula in base a due modalità di funzionamento: NORMALE e RISPARMIO ENERGETICO. In modalità NORMALE, il termostato attiva il riscaldamento quando la temperatura scende sotto i 20°C, e lo disattiva quando la temperatura raggiunge i 23°C. In modalità RISPARMIO ENERGETICO, il riscaldamento si attiva solo quando la temperatura scende sotto i 18°C, e si spegne quando la temperatura raggiunge i 21°C. La modalità può essere cambiata in qualsiasi momento dagli addetti, interagendo con un apposito controllo a parete. Inoltre, gli addetti possono spegnere il termostato e disattivare interamente il sistema di riscaldamento, premendo un apposito pulsante rosso.

Modellare con uno statechart il comportamento del termostato descritto sopra.

Esercizio 4

Il metodo `isCharInString` riportato di seguito viene utilizzato per verificare se un certo carattere è presente all'interno di una data stringa.

```
1 public static boolean isCharInString(String str, char target) {  
2     if (str == null) {  
3         throw new IllegalArgumentException("Invalid String!");  
4     }  
5  
6     if (str.isEmpty()) {  
7         throw new IllegalArgumentException("Invalid String!");  
8     }  
9  
10    int i = 0;  
11    while (true) {  
12        if (i >= str.length() || str.charAt(i) == target) {  
13            break;  
14        }  
15        i++;  
16    }  
17  
18    return i < str.length();  
19 }
```

- X Rappresentare il CFG del metodo `isCharInString`.
- X Scrivere quattro test JUnit con strategia White-Box per il metodo `isCharInString`, indicando per ciascuno di essi quale cammino copre nel CFG. Si richiede che almeno un test copra uno scenario di errore (lancio di una `IllegalArgumentException`) e che, ove possibile, i test coprano cammini distinti.
- X Qual è la *Test Effectiveness Ratio* (TER), relativamente alla copertura di nodi del CFG, di ciascuno dei test sviluppati al punto X? Qual è, invece, la TER relativa all'intera suite di test prodotta al punto X? Motivare la risposta, descrivendo brevemente cosa si intende per TER e a cosa serve tale metrica.

Istruzioni: Scrivere immediatamente, su ogni foglio che vi è stato consegnato, cognome, nome, numero di matricola. Durante la prova, non è consentito consultare materiale didattico, né interagire in alcun modo con dispositivi elettronici o colleghi, pena l'immediato annullamento della prova. Il tempo a disposizione per completare la prova è di 3 ore.

Esercizio 1

Si vuole realizzare una piattaforma digitale per facilitare il tutoring *peer-to-peer* tra studenti. Quando uno studente si registra al servizio, riceve automaticamente un credito gratuito iniziale che può utilizzare per richiedere una sessione di tutoraggio di un'ora. Gli studenti che hanno bisogno di aiuto possono quindi inserire una richiesta di tutoraggio specificando la materia, l'argomento specifico, una durata (in ore) e una data/orario in cui intendono svolgere il tutoraggio. Al momento dell'inserimento della richiesta, il sistema blocca automaticamente i crediti necessari nell'account del richiedente per garantire che il pagamento possa sempre avvenire al termine della sessione. Il sistema pubblica le richieste inserite, rendendole visibili a tutti gli altri studenti registrati. Un altro studente che si sente preparato nella materia può quindi accettare la richiesta, impegnandosi a erogare la sessione di tutoring tramite strumenti di videoconferenza nell'orario/data indicati. Ogni ora di lezione ricevuta costa un credito. Al termine della sessione di tutoring, il tutorando conferma che la lezione è avvenuta e il tutor riceve automaticamente i crediti guadagnati nel proprio account. I tutorandi, inoltre, possono aprire un reclamo, se la sessione non è andata in maniera soddisfacente dal loro punto di vista. Gli studenti possono acquistare crediti aggiuntivi tramite pagamento online (utilizzando il servizio esterno UNINAPAY) oppure convertire i crediti accumulati in sconti per servizi universitari come stampe o buoni mensa. Gli amministratori della piattaforma visualizzano le transazioni, i reclami degli utenti, e possono emettere rimborsi per le lezioni non ritenute soddisfacenti.

Si svolgano i seguenti punti, usando la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.

- Modellare tutti i requisiti funzionali del sistema descritto sopra utilizzando uno Use Case Diagram;
- Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di inserimento di una richiesta di tutoring;
- Dettagliare il caso d'uso relativo alla funzionalità di inserimento di una richiesta di tutoring, per mezzo descrizioni testuali strutturate secondo un formalismo tabellare come quello proposto da A. Cockburn.

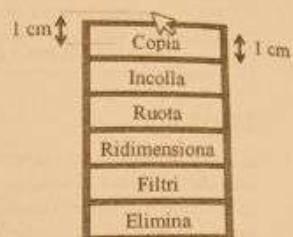
Esercizio 2

In una nota università, droni completamente autonomi supportano i docenti umani nel monitoraggio degli studenti durante le prove d'esame. Quando un docente attiva un drone tramite comando remoto, il sistema inizia automaticamente una fase di calibrazione, durante la quale verifica il corretto funzionamento dei sensori di bordo. Se la calibrazione fallisce, il drone si disattiva immediatamente. Una volta completata con successo la calibrazione, il drone inizia a pattugliare l'aula spostandosi in maniera casuale. Durante il pattugliamento, il sistema analizza l'ambiente circostante attraverso la fotocamera per identificare possibili violazioni del regolamento d'esame. Quando viene rilevata una violazione, il drone cattura un'immagine degli studenti coinvolti e invia una notifica ai docenti, allegando l'immagine acquisita. Completata questa operazione, il drone riprende il normale pattugliamento. Durante tutte le operazioni, se il sensore di batteria rileva un livello inferiore al 5%, il drone sospende tutte le attività e si dirige verso la stazione di ricarica per ricaricare le batterie. Quando il livello di carica supera il 50%, il drone riprende automaticamente le attività di monitoraggio. I docenti possono sospendere temporaneamente il drone in qualsiasi momento tramite comando remoto. Quando questo accade, il sistema memorizza lo stato in cui si trovava. Successivamente, quando il docente riattiva il drone, il sistema riprende l'attività che stava svolgendo prima della sospensione senza dover ripetere la calibrazione. I docenti possono anche disattivare completamente il drone, terminando tutte le operazioni.

Modellare con uno statechart il comportamento di uno dei droni sopra descritti.

Esercizio 3

In un'applicazione di editing fotografico professionale è presente un menu contestuale che appare al click destro con sei opzioni (Copia, Incolla, Ruota, Ridimensiona, Filtri, Elimina), disposte come mostrato nella figura a lato. In seguito a valutazioni sperimentali su utenti reali, è stato determinato che le frequenze di utilizzo di ciascuna delle funzionalità nel menu contestuale sono le seguenti: Elimina (45%), Incolla (25%), Copia (20%), tutte le altre (~3%).



Tenendo conto che l'obiettivo principale per questo software professionale è massimizzare l'efficienza delle interazioni, e analizzando i dati di utilizzo, proporre possibili design alternativi del menu a tendina. Motivare le proprie proposte, facendo riferimento a opportuni modelli e teorie dell'Interazione Uomo-Macchina, e indicando l'impatto che tali design alternativi avrebbero sulla efficienza delle interazioni.

Esercizio 4

Il metodo `calcolaSconto` riportato di seguito viene utilizzato per calcolare lo sconto da applicare a un cliente in base alla sua età, al suo reddito, e al fatto che abbia acquistato o meno un abbonamento *premium*.

```

1 public static int calcolaSconto(int eta, double reddito, boolean isPremium) {
2     if (eta < 18) {
3         throw new IllegalArgumentException("Customers must be 18 or older");
4     } else if (eta >= 65) {
5         if (isPremium) {
6             return 25;
7         } else {
8             return 15;
9         }
10    } else {
11        if (reddito < 20000) {
12            return 10;
13        } else if (reddito > 50000 && isPremium) {
14            return 20;
15        } else {
16            return 5;
17        }
18    }
19 }

```

a. Rappresentare il CFG del metodo `calcolaSconto`.

b. Scrivere quattro test JUnit con strategia White-Box per il metodo `calcolaSconto`, indicando per ciascuno di essi quale cammino copre nel CFG. Si richiede che almeno un test copra uno scenario di errore (lancio di una `IllegalArgumentException`) e che, ove possibile, i test coprano cammini distinti. Si richiede, inoltre, che tutti i test includano asserzioni significative.

c. Qual è la *Test Effectiveness Ratio* (TER), relativamente alla copertura di nodi del CFG, di ciascuno dei test sviluppati al punto (b)? Qual è, invece, la TER relativa all'intera suite di test prodotta al punto (b)? Motivare la risposta, descrivendo brevemente cosa si intende per TER e a cosa serve tale metrica.

Istruzioni: Scrivere immediatamente su ogni foglio cognome, nome e numero di matricola. È consentito utilizzare esclusivamente i fogli forniti dalla commissione. Tutto il materiale personale (quaderni, appunti, dispositivi elettronici), deve essere riposto nello zaino - che deve rimanere chiuso - o depositato sulla cattedra. I dispositivi elettronici (telefoni, smartwatch, fitness band, auricolari, tablet, ecc.) atti alla ricezione di informazioni devono essere spenti e non devono essere mai visibili durante la prova. La prova ha una durata di **3 ore**. Ogni violazione delle regole comporterà l'immediato annullamento della prova.

Esercizio 1

Vi è commissionata la realizzazione di StackGPT, un sistema basato su LLM progettato per supportare ingegneri del software in ogni fase del ciclo di sviluppo, dalla progettazione alla manutenzione del codice. StackGPT funziona come un assistente esperto in ingegneria del software, capace - tra le altre cose - di comprendere e generare codice in linguaggi di programmazione comuni. Gli utenti registrati, possono effettuare il login, accedere al sistema, e eventualmente modificare la password.

StackGPT permette agli utenti registrati di fare richieste inerenti all'Ingegneria del Software. Per fare una richiesta, l'utente indica un titolo e una descrizione dettagliata della richiesta, uno o più tag personalizzati, e seleziona tra due modelli di elaborazione: un modello avanzato (StackGPT-4o) ottimizzato per task complessi e un modello più leggero (StackGPT-3.5) ideale per task più semplici. Ogni utente ha a disposizione una quota giornaliera di 1M di token in input/output per il modello StackGPT-3.5 e di 10k token in input/output per StackGPT-4o. Superata tale soglia, l'utente deve attendere che la quota si re-imposti, nel giorno successivo, per porre nuove richieste. Quando un utente inserisce una richiesta, la risposta di StackGPT viene generata in tempo reale sfruttando le API di OpenAI (che ospita i due modelli utilizzati) e mostrata all'utente, e i token consumati vengono detratti dalla quota disponibile (che gli utenti possono visualizzare tramite una funzionalità apposita).

Le richieste degli utenti e le risposte generate dal sistema sono inoltre pubbliche (ogni utente registrato può visualizzare le richieste di ogni altro utente, effettuando filtri per parola chiave o per contenuto testuale). Gli utenti possono inoltre valutare le risposte del sistema (tramite "mi piace/non mi piace") e lasciare commenti per evidenziare eventuali incompletezze, inconsistenze o errori. Il sistema permette infine agli amministratori della piattaforma di modificare le quote di utilizzo token, e visualizzare i costi di utilizzo mensile in tempo reale, sfruttando le API di OpenAI.

Si svolgano i seguenti punti, usando la propria conoscenza del dominio per derivare dettagli non definiti nei requisiti.

- ✓ a. Modellare tutti i requisiti funzionali del sistema descritto sopra utilizzando uno Use Case Diagram;
- ✓ b. Realizzare i mock-up dell'applicazione descritta, relativamente alla funzionalità di inserimento di una richiesta.
- ✓ c. Dettagliare il caso d'uso relativo alla funzionalità di inserimento di una richiesta, per mezzo descrizioni testuali strutturate secondo un formalismo tabellare come quello proposto da A. Cockburn.

Esercizio 2

L'Art Institute of Chicago vi ha appena assunto come consulenti esperti di Ingegneria del Software per supportarli nella gestione di una fornitura di totem interattivi multimediali (touchscreen). Questi dispositivi saranno distribuiti nelle sale espositive e nelle aree comuni del museo, con lo scopo di: fornire informazioni dinamiche su opere d'arte, mostre temporanee e servizi del museo; guidare i visitatori attraverso percorsi tematici; Supportare funzioni di accessibilità (es. descrizioni audio, traduzioni multilingue); integrarsi con il sistema gestionale del museo per aggiornamenti in tempo reale.

Vi è richiesto di contribuire alla redazione del documento di specifica dei requisiti software per la fornitura di cui sopra, fornendo almeno cinque requisiti non funzionali.

V Esercizio 3

Un'agenzia di cybersecurity ha sviluppato un agente autonomo basato su AI che monitora forum e social media per identificare potenziali minacce (es. frodi, discorsi d'odio, traffico illecito, fake news).

Una volta attivato, l'agente inizia le attività di monitoraggio, scegliendo la prima pagina web da visitare in maniera causale da un pool predefinito di pagine iniziali. La pagina viene quindi caricata e analizzata, cercando contenuti critici. Se non vengono individuate criticità, l'agente sceglie, tra i link presenti nella pagina web corrente, la prossima pagina da visitare, e ripete il ciclo. Se invece vengono ravvisate criticità, il sistema raccoglie ulteriori informazioni di contesto (e.g.: date, luoghi) presenti nella pagina, e trasmette un report alla centrale operativa, prima di riprendere le regolari attività. In ogni momento, se si verificano problemi di connettività, l'agente mette in pausa le operazioni e, quando la connessione ritorna stabile, riprende le attività da dove erano state interrotte.

Si specifichi tramite uno Statechart il comportamento dell'agente AI descritto sopra.

Esercizio 4

Il metodo statico *calcolaPrezzo* della classe *ServiziMuseali* viene utilizzato per calcolare il costo di un biglietto di ingresso in una struttura museale. Il metodo accetta i seguenti parametri:

- *int age*: età della persona che vuole acquistare il biglietto;
- *String type*: stringa che indica il livello di fidelity del cliente. Può assumere valori in "BASE" (clienti regolari), "UNIVERSITY" (per studenti universitari), o "GUIDES" (per guide e accompagnatori);
- *boolean protected*: indica se il cliente appartiene ad una delle categorie protette a norma di legge;

Se almeno uno dei parametri è non valido, il metodo solleva una *IllegalArgumentExcepion*. In caso contrario, il metodo ritorna un numero float rappresentante il prezzo (in Euro) del biglietto. Tale prezzo è calcolato in base alla tipologia di cliente e in base alla sua età, come da tabella seguente.

		age		
		age < 18	18 ≤ age < 65	age ≥ 65
type	BASE	0	10	5
	UNIVERSITY	0	7	0
	GUIDES	0	5	0

Per le categorie protette, inoltre, il biglietto di ingresso è sempre gratuito.

- Indicare, per ciascuno dei parametri del metodo *calcolaPrezzo*, un partizionamento in classi di equivalenza per testing black-box;
- Scrivere tre test JUnit con strategia Black Box per il metodo *calcolaPrezzo*, indicando per ciascuno di essi quali classi di equivalenza copre. Si richiede inoltre che un test corrisponda a scenari in cui almeno uno dei parametri è non valido, e che i restanti due corrispondano a scenari in cui tutti i parametri sono validi. Ove possibile, i test devono coprire classi di equivalenza distinte.
- Quanti test sono necessari per testare il metodo con strategia R-WECT? Motivare la risposta, descrivendo brevemente tale strategia.