Progetto gestione carte "SWENG"

Ingegneria del Software

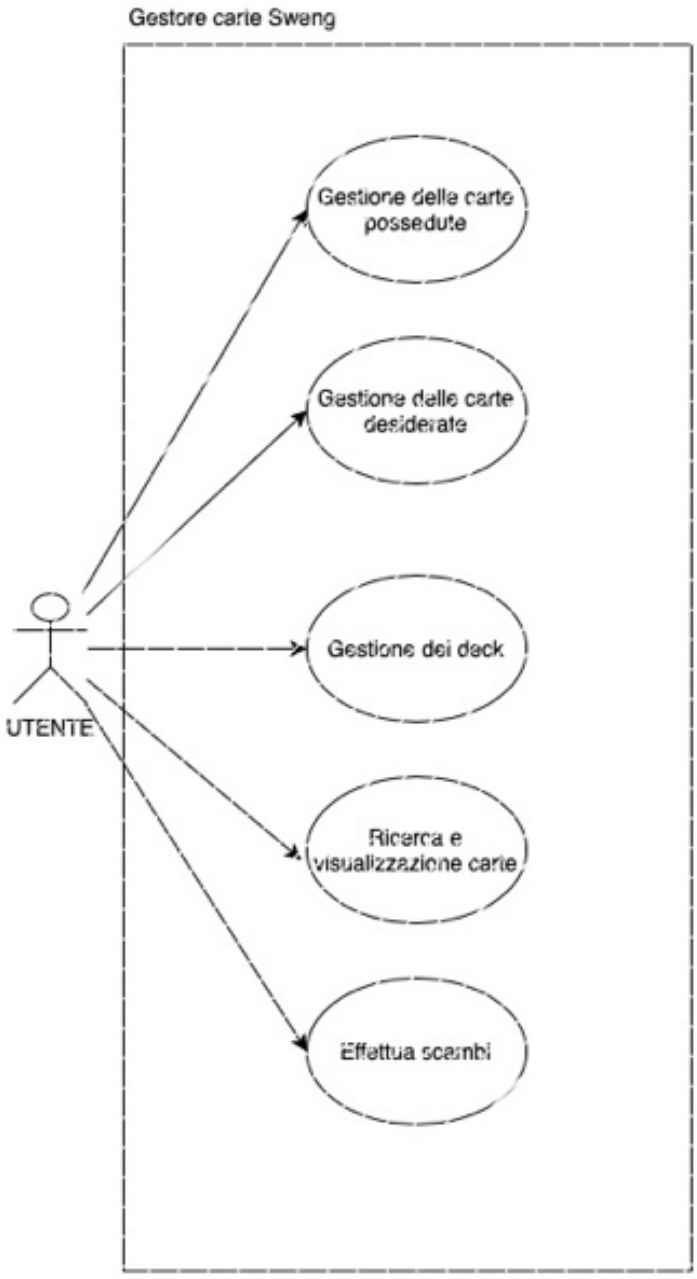
2022-2023

Informatica per il management, Università di Bologna

Membri del gruppo:

Laura Specchiulli, matricola n°: 988074
Filippo Berveglieri, matricola n°: 974754
Gabriele Centonze, matricola n°: 971019

DIAGRAMMA DEI CASI D'USO



NB: si tenga presente che le linee del diagramma non sono tratteggiate volontariamente, ma sono state modificate durante la fase di esportazione, bisogna considerarle dunque come linee continue.

ID: Gestione delle carte possedute
Actor: Utente
Pre-condition: Carte possedute esistenti
Main sequence:
1) Utente apre il gestore carte
2) Utente effetua login
3) Clicca su carta interessata
4) Aggiunge/modifica/cambia stato
Alternative sequence:
- Utente fallisce autenticazione
- Utente non possiede carte
- Carte non caricate sul gestore
Post-conditions: Carta aggiunta/rimossa/cambiata di stato

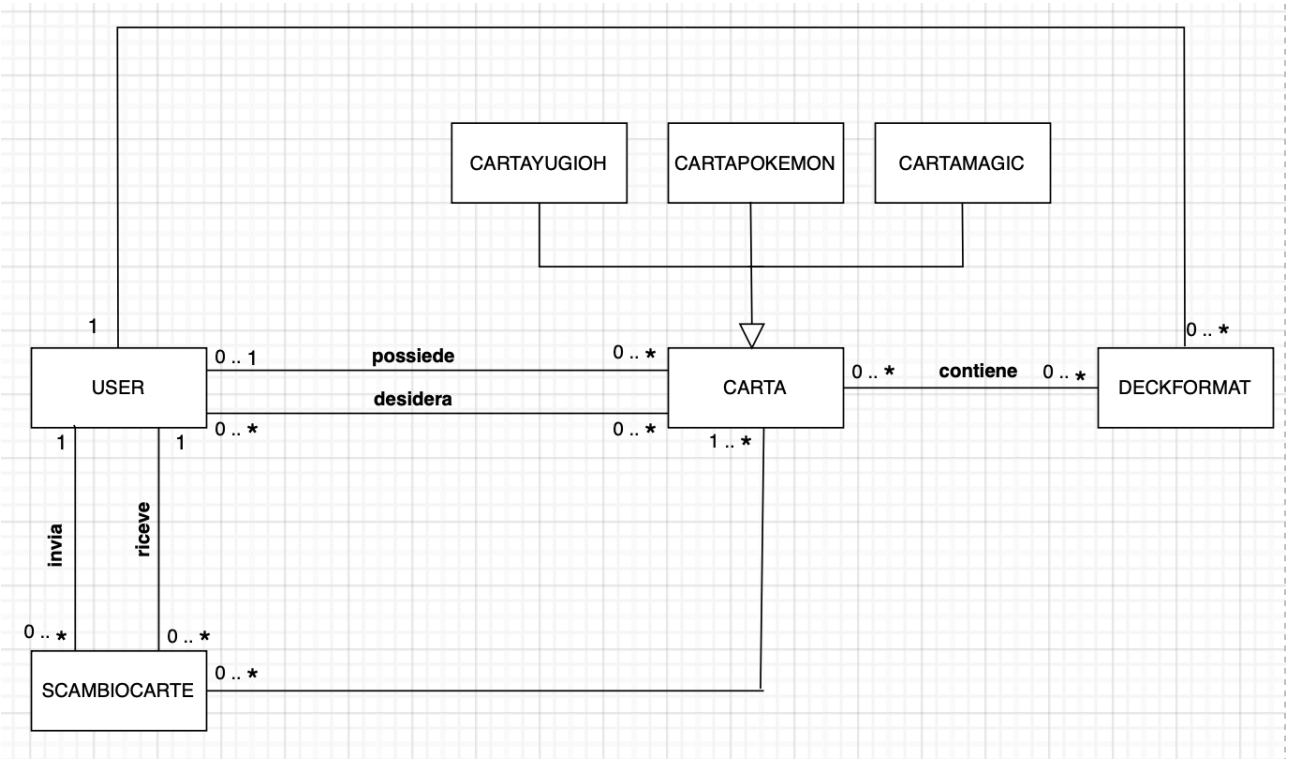
ID: Gestione delle carte desiderate
Actor: Utente
Pre-condition: Carte disponibili esistenti
Main sequence:
1) Utente apre il gestore carte
2) Utente effetua login
3) Clicca su carta desiderata
4) Aggiunge/Rimuove da carte desiderate
Alternative sequence:
- Utente fallisce autenticazione
- Carte disponibili inesistenti
Post-conditions: Carta aggiunta/rimossa dalle carte desiderate

ID: Gestione dei deck
Actor: Utente
Pre-condition: L'utente possiede carte
Main sequence:
1) Utente apre il gestore carte
2) Utente effetua login
3) Entra nella sezione deck
4a) Crea/Elimina un deck
4b)Aggiunge carte da un deck
Alternative sequence:
- Utente fallisce autenticazione
- Nessuna carta posseduta
- Nessuna carta nel deck (per la rimozione della carta)
- Nessun deck esiste (per la rimozione del deck)
Post-conditions: Carta aggiunta dal deck. Deck creato/rimosso

ID: Ricerca/visualizzazione carte
Actor: Utente
Pre-condition: Le carte son caricate sul gestore
Main sequence:
1) Utente apre il gestore carte
2) Utente effetua login
3) Digita il nome e la tipologia di gioco della carta che intende ricercare
4) Clicca sul nome della carta per visualizzarla
Alternative sequence:
- Utente fallisce autenticazione
- Nessuna carta caricata sul gestore
Post-conditions: Carta ricercata e visualizzata

ID: Scambio
Actor: Utente
Pre-condition: L'utente desidera una carta e ne vuole cedere un'altra
Main sequence:
1) Utente apre il gestore carte
2) Utente effetua login
3) Entra nella sezione scambi
4) Utente controlla la lista di carte da cedere e da richiedere
5) Invia la proposta di scambio
Alternative sequence:
- Utente fallisce autenticazione
- Utente non ha carte desiderate
- Utente non ha carte che desidera cedere
Post-conditions: Scambio proposto

DIAGRAMMA DI DOMINIO



GLOSSARIO

USER	Entità che rappresenta l'utente, con le seguenti proprietà: email e password.
CARTA	Entità che rappresenta una carta con le seguenti proprietà: nome, descrizione utente e stato
SCAMBIOCARTE	Entità che rappresenta uno scambio di carte che può avvenire tra due utenti, con le seguenti proprietà: email mittente ed email destinatario, carta mittente(le carte che il mittente decide di cedere) e carta destinatario(le carte che il mittente vuole ricevere) e chiave scambio che rappresenta il codice univoco dello scambio.
DECKFORMAT	Entità che rappresenta un mazzo di carte con le seguenti proprietà: lista delle carte al suo interno, nome, tipologia gioco appartenente.
CARTAYUGIOH	Entità che specializza la classe carta con gli attributi: tipo, descrizione, razza, immagine e immagine small.
CARTAPOKEMON	Entità che specializza la classe carta con gli attributi: varianti, illustratore, immagine, tipo e descrizione.
CARTAMAGIC	Entità che specializza la classe carta con gli attributi: artista, testo, tipo, e i booleani "has foil", "Is Alternative", "Is Full Art", "Is Promo" , "Is Reprint".

MANUALE DELLO SVILUPPATORE

Seguendo il link sottostante della repository GitHub è possibile ottenere il codice sorgente.

Link repository github: https://github.com/gabritech01/sweng

Successivamente, per poter avviare il progetto, è necessaria l'installazione del framework GWT all'interno del proprio ambiente. Nello specifico abbiamo scelto di utilizzare Eclipse come ambiente di sviluppo, in modo da facilitarne l'installazione e l'utilizzo. Una volta scaricato il framework bisogna aprire il progetto e selezionare la cartella "…" con il tasto destro del mouse selezionando poi la voce "Run as" e successivamente la voce "GWT Development mode with Jetty".

La directory principale del progetto è stata suddivisa in 3 parti:

CLIENT: In client vi è una suddivisione tra la parte logica e la parte di interfaccia grafica, nei rispettivi file .java e nei file ui.xml. Le varie pagine vengono caricate all'interno del componente container presente nel file Sweng.html della cartella war tramite il metodo RootPanel.get("container").add(). La classe DbService implementa rpc.RemoteService per la gestione delle chiamate al db lato client. DbServiceAsync è la rispettiva rappresentazione con chiamate async.

SHARED: La cartella shared fa da ponte tra il client e il server per cui contiene le classi che sono condivise da entrambi nella nostra applicazione

SERVER: La classe AppServletContextListener si occupa di chiamare la classe DatabaseInitializer all'avvio del server, quest'ultima si occupa, solo se non è stato fatto in precedenza, di popolare il dbCarte dopo la lettura dei file json. La classe DbServiceImpl si occupa di realizzare i metodi che erano stati descritti nei due file del client DbService e DbServiceAsync. Infine, abbiamo implementato una logica di database separati per ogni collezione di dati che necessita la nostra applicazione nei rispettivi file db.

MANUALE DELL'UTENTE

All'apertura del gestore di carte Sweng, per prima cosa, viene visualizzata la pagina di login.

L'utente, se già in possesso di credenziali, inserirà email e password per poi cliccare su 'login'. In caso l'utente non fosse in possesso di un account potrà crearne uno inserendo le credenziali desiderate e cliccando su 'signin'.

Una volta autenticato, l'utente si trova nella homepage del gestore di carte.
Nella homepage si può ricercare una carta oppure visualizzare le carte possedute. Per ricercare una carta è necessario inserirne il nome e cliccare su 'cerca'. È, inoltre, possibile affinare la ricerca inserendo il gioco a cui è relativa la carta desiderata. È possibile anche vedere tutte le carte relative ad un determinato gioco: per farlo è sufficiente selezionare il gioco senza scrivere nulla nella barra di ricerca e cliccare 'cerca'.

Nella sezione, sottostante, invece, sono elencate le carte possedute dall'account con cui si è effettuato l'accesso suddivise per gioco.

Oltre alle carte possedute, nella homepage, è possibile visualizzare le carte desiderate che sono quelle carte che l'utente non possiede ma che vorrebbe avere.

Cliccando sulla carta, sia che questo avvenga dalla barra di ricerca piuttosto che da "le tue carte possedute" o "le tue carte desiderate" si verrà indirizzati alla pagina di dettaglio della carta.
In questa pagina è possibile visualizzare l'immagine della carta e le sue caratteristiche, compreso lo stato. Lo stato rappresenta l'usura della carta ed è rappresentato su una scala da 1 (molto rovinata) a 5 (in perfette condizioni).

Nella pagina del dettaglio della carta è, inoltre, possibile consultare l'elenco degli account che possiedono quella carta e l'elenco degli account che la cercano.

Un'altra funzionalità selezionabile tramite la barra di navigazione è la sezione "scambi". Cliccando su "scambi" comparirà una pagina divisa in due parti: una parte ("Bozza scambio") adibita alla proposta di uno scambio e un'altra ("Richieste di scambio").

Per proporre uno scambio ad un'altra persona è necessario digitare la mail del destinatario dello scambio nello spazio e cliccare sul tasto "invia scambio".

Per far si che una carta compaia in automatico nelle "carte da cedere" è necessario aprire il dettaglio della carta dalle "carte possedute" e cliccare sul tasto "cedo questa carta". Simil discorso per aggiungerla alle "carte da prendere", dove occorre aprire il dettaglio della carta dalla barra di ricerca o dalle "carte desiderate" e cliccare su "voglio questa carta".

Nella parte sottostante invece è presente la parte di gestione degli scambi che son stati proposti all'account. Compare il mittente dello scambio, le carte prese in considerazioni e i 2 tasti "accetta" e "rifiuta" che, appunto, ci permettono di accettare o rifiutare uno scambio.

Nella sezione della barra di navigazione troviamo anche il tasto "I tuoi deck" dove, cliccandoci sopra, si aprirà una pagina in cui è possibile gestire i propri mazzi di carte.

Accedendoci l’utente può creare da zero un nuovo deck, assegnandogli un nome e selezionando la categoria di carte nel quale sarà inserito (deck YuGiOh, deck Magic o deck Pokemon). Una volta creato un mazzo, esso comparirà sotto alla propria categoria e cliccandoci sopra l’utente visualizzerà una schermata con la lista di carte da poter aggiungere al nuovo deck appena creato, carte prese dalla sezione “carte possedute”. Inoltre, con un pulsante, l’utente può eliminare il deck creato.

DIARIO DEL PROGETTO

Per organizzare meglio il lavoro abbiamo adottato la metodologia Scrum con i suoi sistemi di Product backlog e Sprint backlog per il coordinamento del lavoro.

Per una maggiore coordinazione all'interno del gruppo abbiamo deciso di dividerci e ruotare al cambio di ogni Sprint, nei 3 diversi ruoli di Product Owner, Scrum Master e Development Team, realizzando nel totale 3 sprint.
Per decidere come dividerci il lavoro totale in tasks da elaborare in seguito abbiamo effettuato diversi Sprint Planning, confrontandoci ogni giorno sul lavoro svolto e sui task da completare o ancora eseguire tramite dei Daily Scrum effettuati tramite call di gruppo su Teams dove abbiamo esposto le eventuali problematiche e soluzioni.

Oltre ai Daily Scrum giornalieri, al termine di ogni sprint veniva organizzata una Sprint Review e una Sprint Retrospective per mostrare delle demo del prodotto realizzato verificando eventuali problemi e dettagli da sistemare.

Per cercare di coordinare e di gestire al meglio possibile il progetto ci siamo serviti di GitHub, applicazione web che ci permette di dividere il lavoro in tramite l'Issues tracking system, nel quale è possibile assegnare a ogni componente del team più tasks da completare.

Per una questione di comodità oltre allo stato del task (todo, done, progress) noi abbiamo deciso di personalizzarci le label ed assegnarle ai vari task. Questo perché lo stato nella homepage delle issue non veniva mostrato mentre la label compariva immediatamente affianco al titolo della issue.

Le 4 label da noi create sono:
1)DA FARE —> label appartenente a tutti i tasks che devono essere implementati
2)PARZIALE —> label appartenente ai tasks in lavorazione da parte del team di sviluppo
3)FATTO —> label appartenente ai tasks completati da parte del Team
4) NON FUNZIONANTE —> label da assegnare eventualmente alle tasks che non si sono concluse con successo

SPRINT N° 1

PERIODO: 16/08/2023 – 22/08/2023

Product Owner: Laura Specchiulli
Scrum Master: Filippo Berveglieri
Development Team: Gabriele Centonze

Stato della board ad inizio 1° sprint: Tutte le issues erano nella sezione “to do”
Stato della board a fine 1° sprint: **Vedi immagine 1 e 2**

SPRINT N° 2

PERIODO: 22/08/2023 – 26/08/2023

Product Owner: Gabriele Centonze
Scrum Master: Laura Specchiulli
Development Team: Filippo Berveglieri

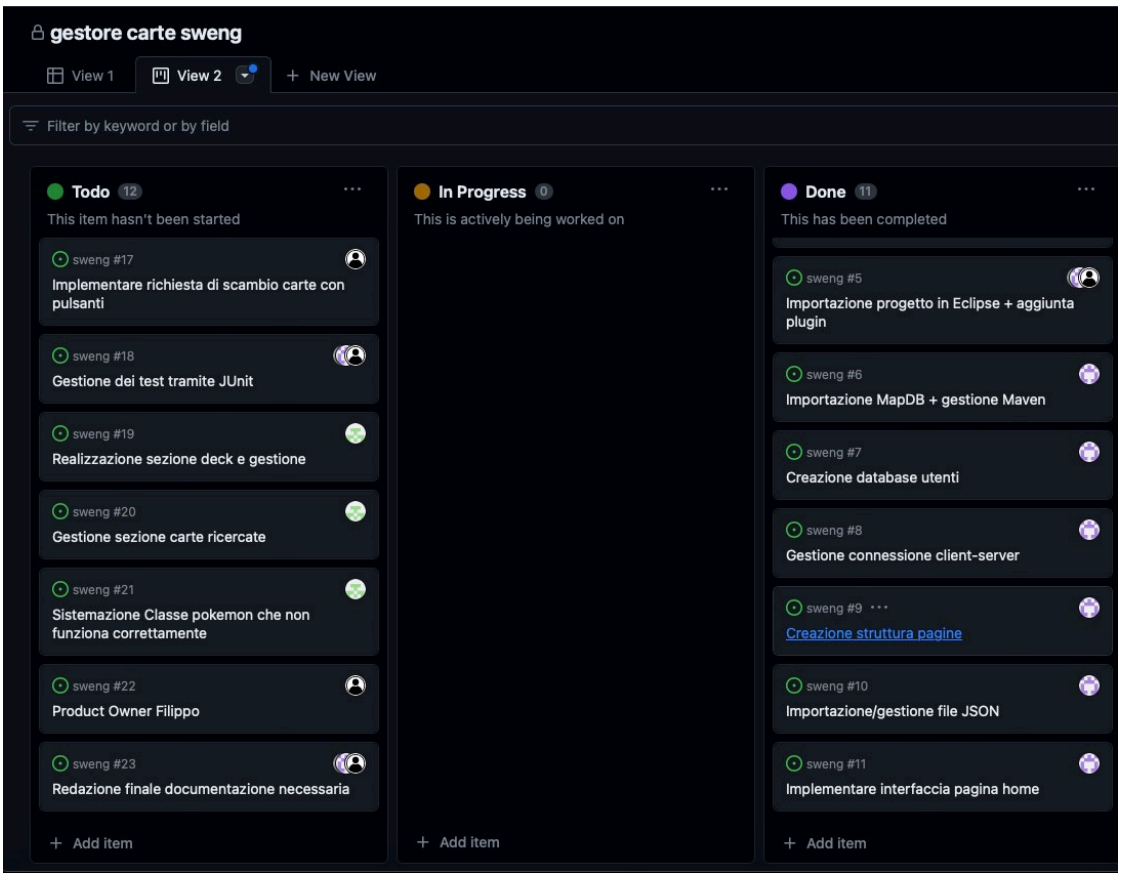
Stato della board ad inizio 2° sprint: **Vedi immagine 1 e 2**
Stato della board a fine 2° sprint: **Vedi immagine 3 e 4**

SPRINT N° 3

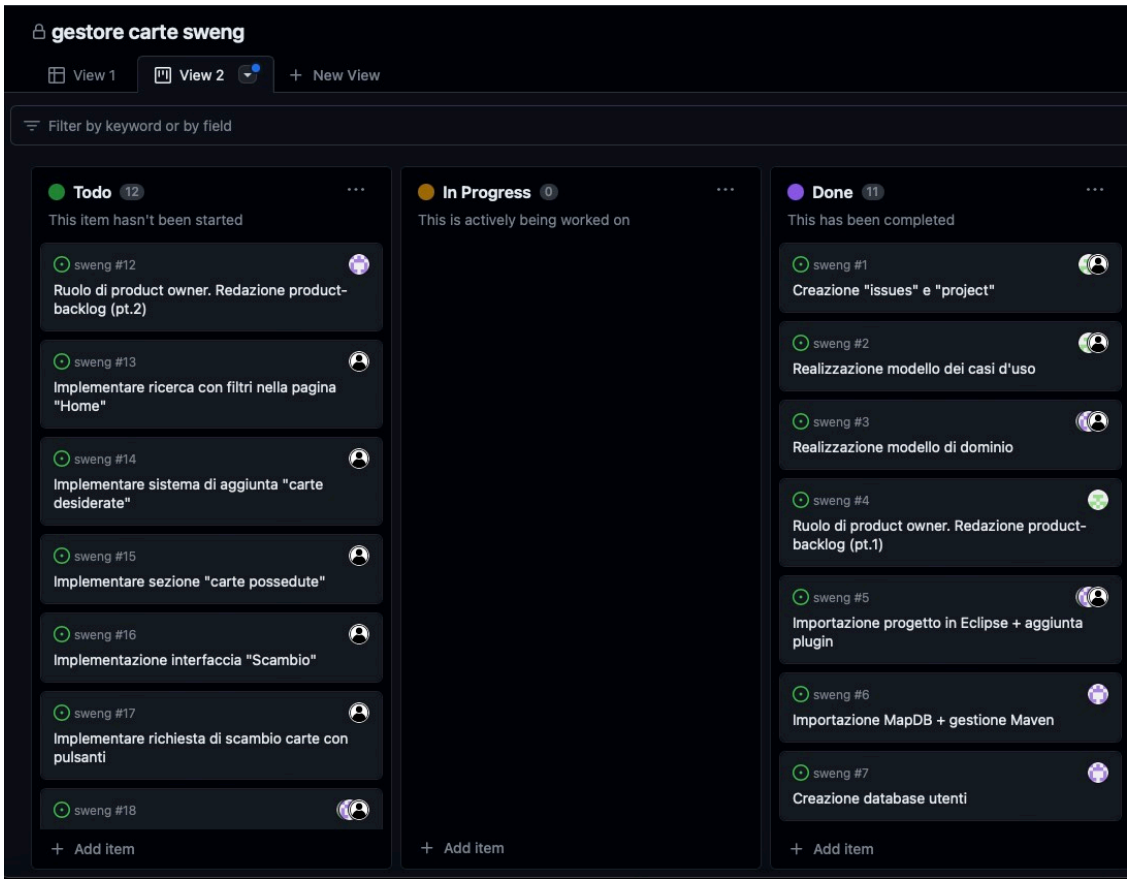
PERIODO: 27/08/2023 – 30/08/2023

Product Owner: Filippo Berveglieri
Scrum Master: Gabriele Centonze
Development Team: Laura Specchiulli

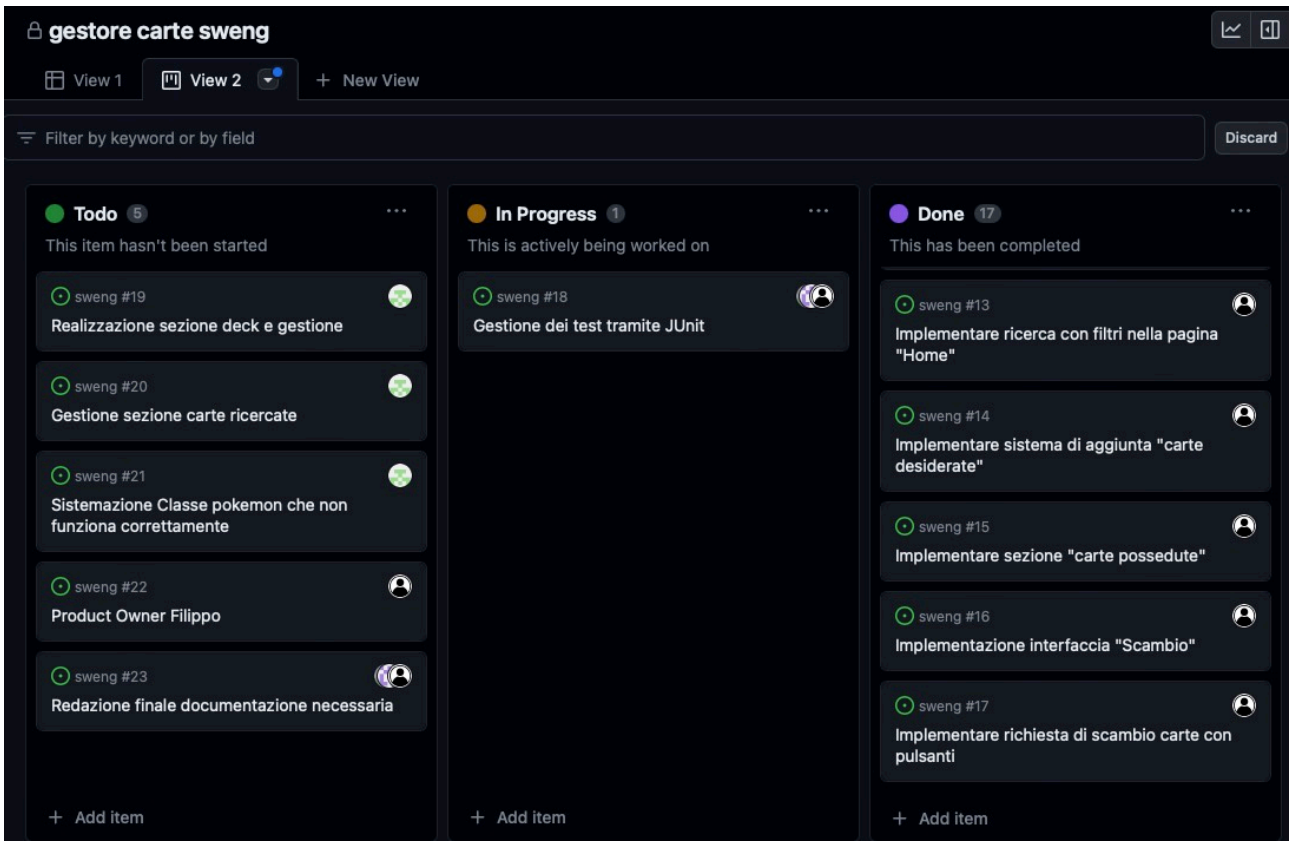
Stato della board ad inizio 2° sprint: **Vedi immagine 3 e 4**
Stato della board a fine 3° spint: Tutte le issues erano nella sezione “done” (**Vedi immagine 5 e 6**)


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Todo 12
This item hasn't been started
sweng #17
Implementare richiesta di scambio carte con pulsanti
sweng #18
Gestione dei test tramite JUnit
sweng #19
Realizzazione sezione deck e gestione
sweng #20
Gestione sezione carte ricercate
sweng #21
Sistemazione Classe pokemon che non funziona correttamente
sweng #22
Product Owner Filippo
sweng #23
Redazione finale documentazione necessaria
Add item
In Progress 0
This is actively being worked on
Add item
Done 11
This has been completed
sweng #5
Importazione progetto in Eclipse + aggiunta plugin
sweng #6
Importazione MapDB + gestione Maven
sweng #7
Creazione database utenti
sweng #8
Gestione connessione client-server
sweng #9
Creazione struttura pagine
sweng #10
Importazione/gestione file JSON
sweng #11
Implementare interfaccia pagina home
Add item

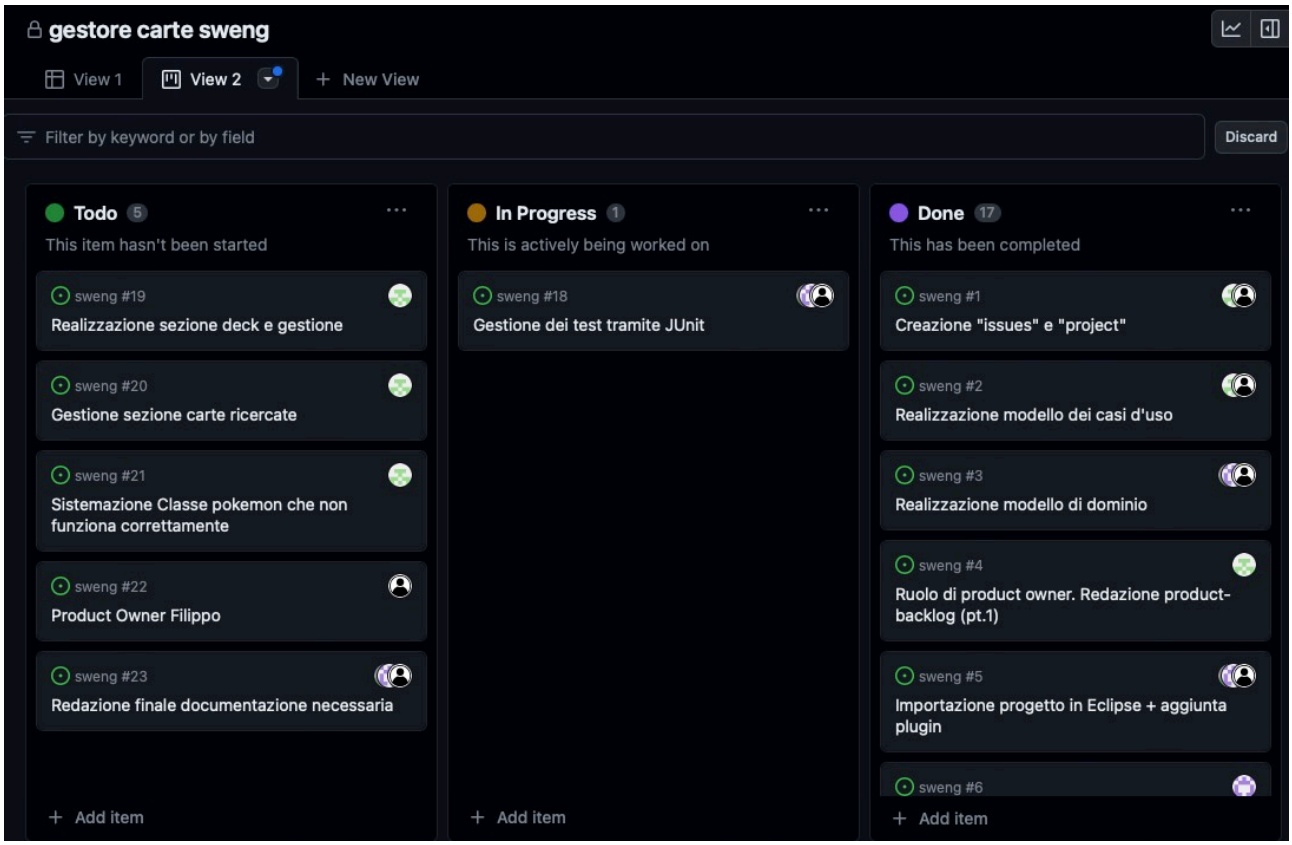
1


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Todo 12
This item hasn't been started
sweng #12
Ruolo di product owner. Redazione product-backlog (pt.2)
sweng #13
Implementare ricerca con filtri nella pagina "Home"
sweng #14
Implementare sistema di aggiunta "carte desiderate"
sweng #15
Implementare sezione "carte possedute"
sweng #16
Implementazione interfaccia "Scambio"
sweng #17
Implementare richiesta di scambio carte con pulsanti
sweng #18
Add item
In Progress 0
This is actively being worked on
Add item
Done 11
This has been completed
sweng #1
Creazione "issues" e "project"
sweng #2
Realizzazione modello dei casi d'uso
sweng #3
Realizzazione modello di dominio
sweng #4
Ruolo di product owner. Redazione product-backlog (pt.1)
sweng #5
Importazione progetto in Eclipse + aggiunta plugin
sweng #6
Importazione MapDB + gestione Maven
sweng #7
Creazione database utenti
Add item

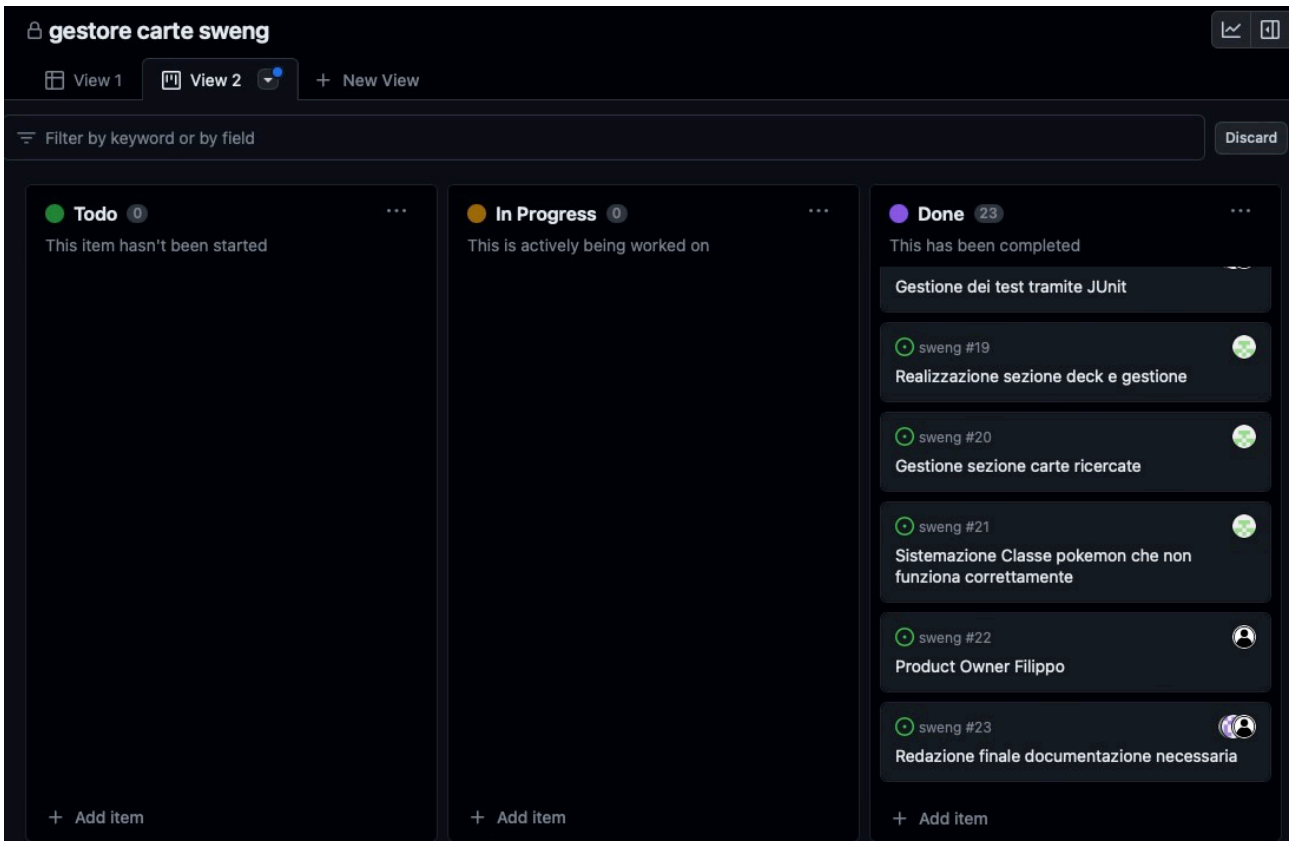
2


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Discard
Todo 5
This item hasn't been started
sweng #19
Realizzazione sezione deck e gestione
sweng #20
Gestione sezione carte ricercate
sweng #21
Sistemazione Classe pokemon che non funziona correttamente
sweng #22
Product Owner Filippo
sweng #23
Redazione finale documentazione necessaria
Add item
In Progress 1
This is actively being worked on
sweng #18
Gestione dei test tramite JUnit
Add item
Done 17
This has been completed
sweng #13
Implementare ricerca con filtri nella pagina "Home"
sweng #14
Implementare sistema di aggiunta "carte desiderate"
sweng #15
Implementare sezione "carte possedute"
sweng #16
Implementazione interfaccia "Scambio"
sweng #17
Implementare richiesta di scambio carte con pulsanti
Add item

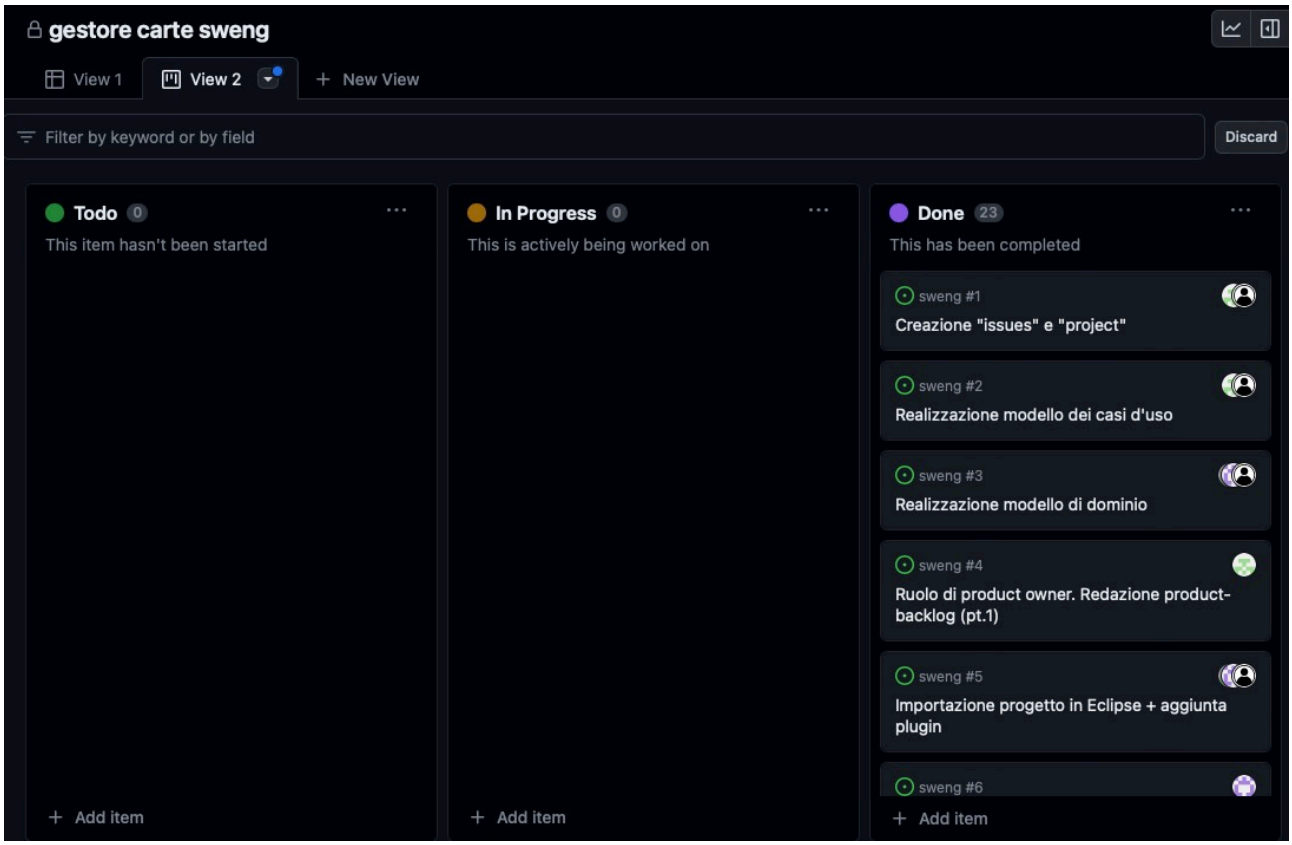
3


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Discard
Todo 5
This item hasn't been started
sweng #19
Realizzazione sezione deck e gestione
sweng #20
Gestione sezione carte ricercate
sweng #21
Sistemazione Classe pokemon che non funziona correttamente
sweng #22
Product Owner Filippo
sweng #23
Redazione finale documentazione necessaria
Add item
In Progress 1
This is actively being worked on
sweng #18
Gestione dei test tramite JUnit
Add item
Done 17
This has been completed
sweng #1
Creazione "issues" e "project"
sweng #2
Realizzazione modello dei casi d'uso
sweng #3
Realizzazione modello di dominio
sweng #4
Ruolo di product owner. Redazione product-backlog (pt.1)
sweng #5
Importazione progetto in Eclipse + aggiunta plugin
sweng #6
Add item

4


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Discard
Todo 0
This item hasn't been started
Add item
In Progress 0
This is actively being worked on
Add item
Done 23
This has been completed
Gestione dei test tramite JUnit
sweng #19
Realizzazione sezione deck e gestione
sweng #20
Gestione sezione carte ricercate
sweng #21
Sistemazione Classe pokemon che non funziona correttamente
sweng #22
Product Owner Filippo
sweng #23
Redazione finale documentazione necessaria
Add item

5


gestore carte sweng
View 1
View 2
New View
Filter by keyword or by field
Discard
Todo 0
This item hasn't been started
Add item
In Progress 0
This is actively being worked on
Add item
Done 23
This has been completed
sweng #1
Creazione "issues" e "project"
sweng #2
Realizzazione modello dei casi d'uso
sweng #3
Realizzazione modello di dominio
sweng #4
Ruolo di product owner. Redazione product-backlog (pt.1)
sweng #5
Importazione progetto in Eclipse + aggiunta plugin
sweng #6
Add item

6

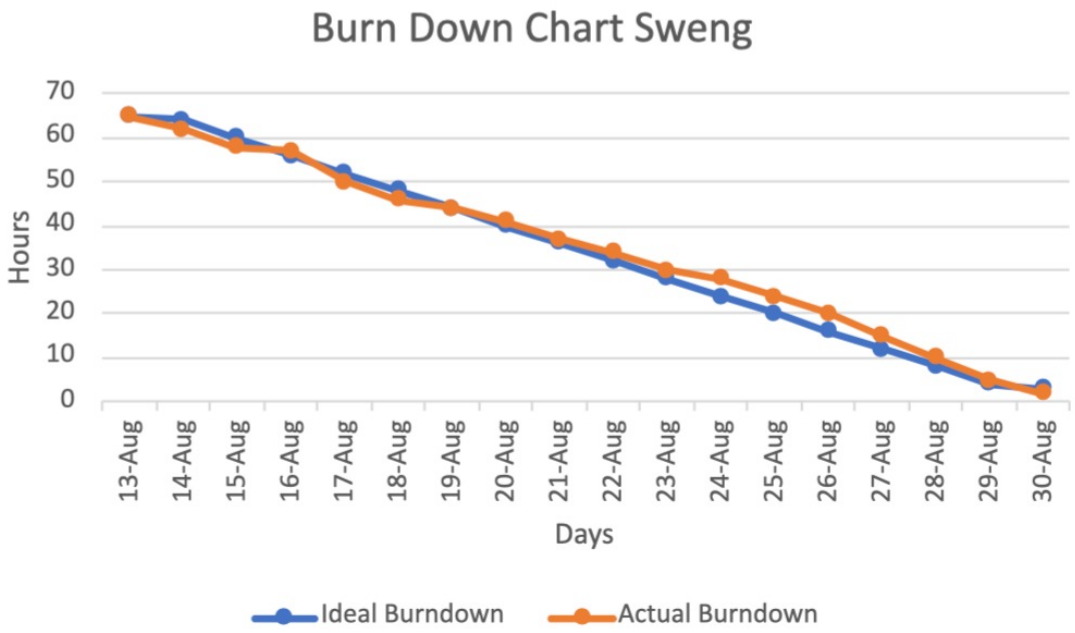
DIARIO GIORNALIERO

Data	Task	Tempo	Ruoli
13/08/2023	Primo meeting di gruppo: conoscenza + definizione obiettivi	1 ora	
13/08/2023	Lettura specifiche del progetto	2 ore	
14/08/2023	Creazione repository	1 ora	
14/08/2023	Installazione GWT + Eclipse + creazione progetto GWT su Eclipse	3 ore	
15/08/2023	Commit iniziale di prova	1 ora	
16/08/2023	Organizzazione lavoro Scrum	3 ore	
16/08/2023	Realizzazione modello dei casi d'uso e modello di dominio	4 ore	
16/08/2023	Inizio realizzazione pagina login + registrazione. Daily scrum	3 ore	Product owner: LS Scrum master: FB Dev. team: GC
17/08/2023	Risoluzione problematiche legate a GWT e Maven + comprensione ambiente GitHub + daily scrum	4 ore	Product owner: LS Scrum master: FB Dev. team: GC
18/08/2023	Sincronizzazione GitHub (creazione project, issue) + daily scrum	2 ore	Product owner: LS Scrum master: FB Dev. team: GC
19/08/2023	Importazione MapDB, gestione Maven e struttura pagine + daily scrum	3 ore	Product owner: LS Scrum master: FB Dev. team: GC
20/08/2023	Creazione database utenti, implementazione json + daily scrum	4 ore	Product owner: LS Scrum master: FB Dev. team: GC
21/08/2023	Implementazione "ricerca con filtri" in homepage + daily scrum	3 ore	Product owner: LS Scrum master: FB Dev. team: GC
22/08/2023	Risoluzione problemi legati a file json e db, lettura db carte, inserimento pagina "card details" + daily scrum	3 ore	Product owner: LS Scrum master: FB Dev. team: GC
22/08/2023	Sprint review e sprint retrospective	1 ora	Product owner: LS Scrum master: FB Dev. team: GC

23/08/2023	Conclusione e perfezionamento ricerca con filtri + daily scrum	2 ore	Product owner: GC Scrum master: LS Dev. team: FB
24/08/2023	Implementazione sezione “carte desiderate” e “carte possedute” + daily scrum	3 ore	Product owner: GC Scrum master: LS Dev. team: FB
25/08/2023	Implementazione pagina scambi + daily scrum	4 ore	Product owner: GC Scrum master: LS Dev. team: FB
26/08/2023	Ultimazione pagina “scambi” + commit + daily scrum	4 ore	Product owner: GC Scrum master: LS Dev. team: FB
26/08/2023	Sprint review e sprint retrospective	1 ora	Product owner: GC Scrum master: LS Dev. team: FB
27/08/2023	Inizio implementazione pagina “deck” + daily scrum	5 ore	Product owner: FB Scrum master: GC Dev. team: LS
28/08/2023	Implementazione pagina “deck” + sezione carte ricercate con filtro nella pagina Home + correzione classe Pokemon + daily scrum	5 ore	Product owner: FB Scrum master: GC Dev. team: LS
29/08/2023	Sistemazione dettaglio foto delle carte+ Testing finale + aggiunta commenti al codice + sprint review e sprint retrospective finale	3 ore	Product owner: FB Scrum master: GC Dev. team: LS
30/08/2023	Dettagli finali + risoluzione alcuni bug + stesura documentazione per consegna	2 ore	

BURN DOWN CHART

Un Burn Down Chart è uno strumento utile per monitorare il progresso di un progetto di sviluppo software nel tempo. Può aiutare a visualizzare se il team sta rispettando i tempi previsti per la consegna delle attività.



Nel nostro caso, dopo il primo meeting ci eravamo prefissati la consegna del lavoro entro la data del 30/08 prevedendo un impiego giornaliero di 4 ore. La scadenza è stata rispettata nonostante, come si evince dal Burn Down Chart, in alcuni giorni son state lavorate meno ore che, però, son state compensate da altri giorni di maggior lavoro.

FASE DI TESTING – JUNIT

JUnit è un framework utile ad implementare programmi di testing nella programmazione in Java.

Nel nostro caso abbiamo utilizzato JUnit senza però adottare un approcio TDD (Test Driven Development). L'approcio TDD prevede che prima di iniziare la fase di stesura del codice vengano create le unità di testing e, in funzione di queste, vengano poi scritti i codici in modo che siano in grado di superare la fase di testing.

Abbiamo però svolto numerosi test tra cui quello sulla creazione degli utenti, sul login, sulla sull registrazione, sulla ricerca di una carta, sull'aggiunta di una carta, sugli scambi,…
I test si son rivelati fondamentali per la fase di sviluppo, soprattutto per facilitare il processo di debugging.