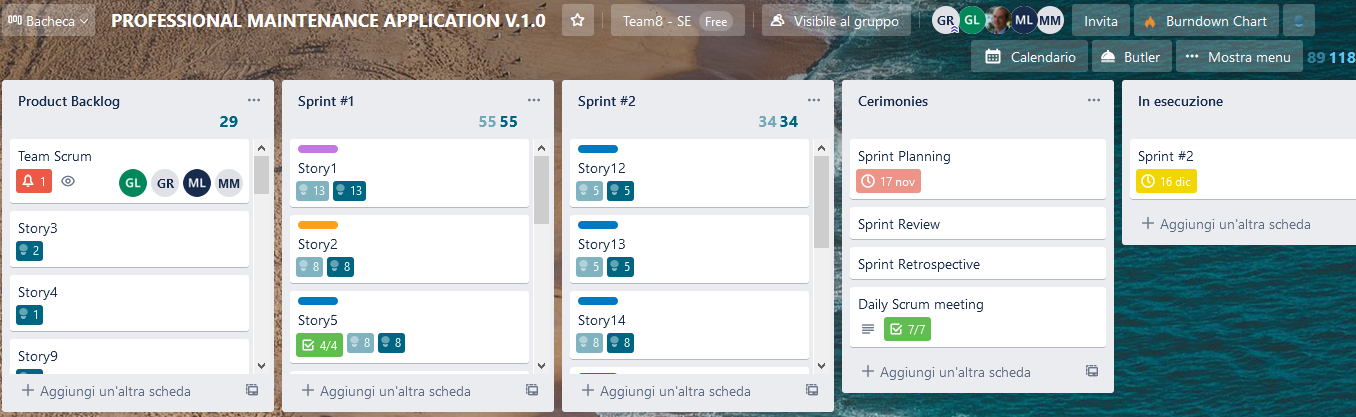
**Progetto Software Engineering – Team 8**

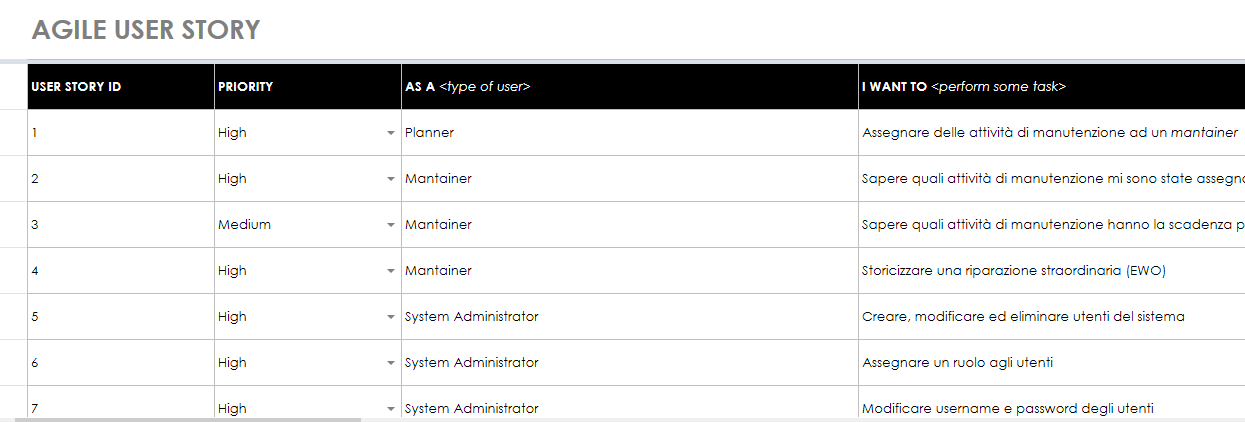
Lo scopo di questo progetto è realizzare un sistema informativo per la gestione delle attività di manutenzione di macchinari dislocati in diversi luoghi. Per realizzare questo sistema, è stato utilizzato Java come linguaggio di programmazione e POSTGRESQL come DBMS per la gestione dei dati.

**Scrum**

Come previsto dalle specifiche del progetto, è stata usata la metodologia agile “Scrum”, corredata dal tool Trello, uno strumento perfetto per il Project Management, il quale gestisce e organizza tutto il lavoro svolto attraverso schede, liste e bacheche. In particolare, risulta utilissimo per distinguere le varie task in maniera semplice ed intuitiva, difatti si possono spostare tramite ‘drop’ da una scheda all’altra (per distinguerli tra task completate, in esecuzione oppure da fare). Inoltre, grazie a questo strumento risulta molto più semplice la collaborazione con altri membri del team. In generale nella prima scheda vi sono tutte le user stories con relativi story points (punteggi da assegnare a ciascun task a seconda della difficoltà), nelle successive due schede sono distinte nel dettaglio le story nei relativi sprint che andremo ad affrontare: nella seguente figura è mostrata la schermata di Trello del progetto.

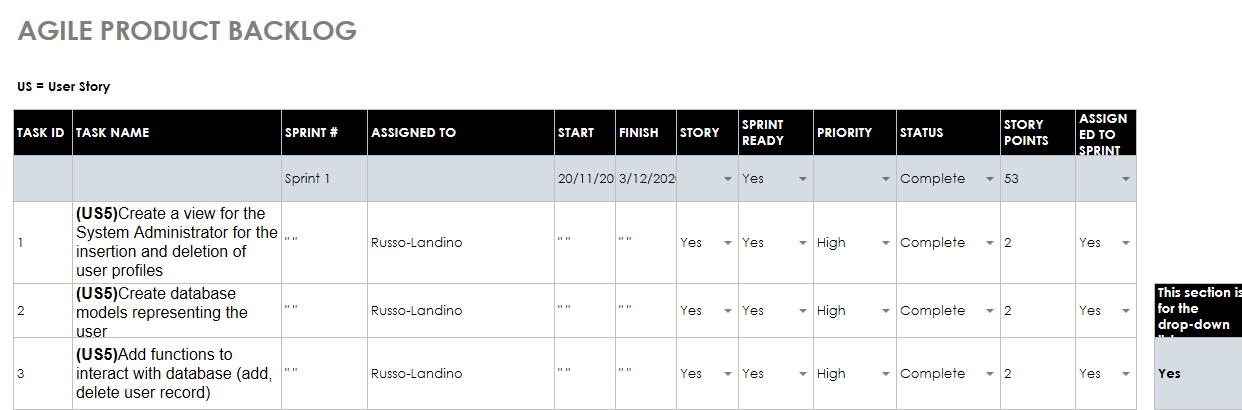


Il lavoro iniziale è stato quello di individuare dal documento SRS le user stories necessarie per il completamento del progetto e dividerle in due sprint che sarebbero state svolte nel corso di questo lavoro: la seguente figura è una vista sull’insieme di tutte le US previste (*UserStories.xlsx*).



Le principali user stories (sono 31) individuate, ovvero quelle a priorità più alta, sono quelle relative alla manipolazione delle attività di manutenzione, alla manipolazione degli addetti alla manutenzione e delle loro competenze, all’assegnazione di un’attività di manutenzione a un addetto specifico e alla gestione degli utenti del database.

La seguente figura, invece, è una schermata sulla suddivisione delle user stories nelle due sprint (*SprintBacklog.xlsx)*.

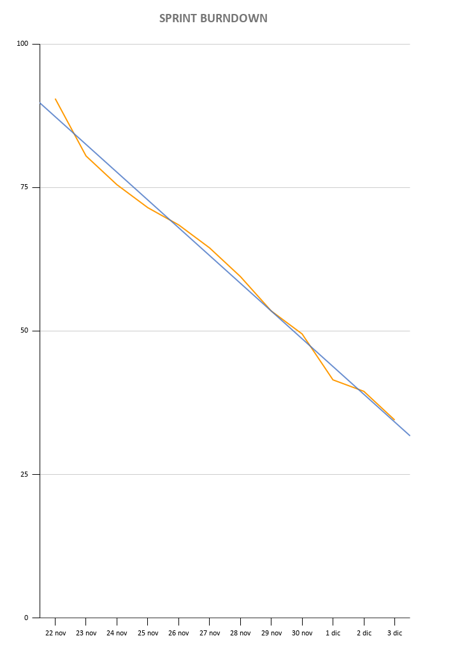


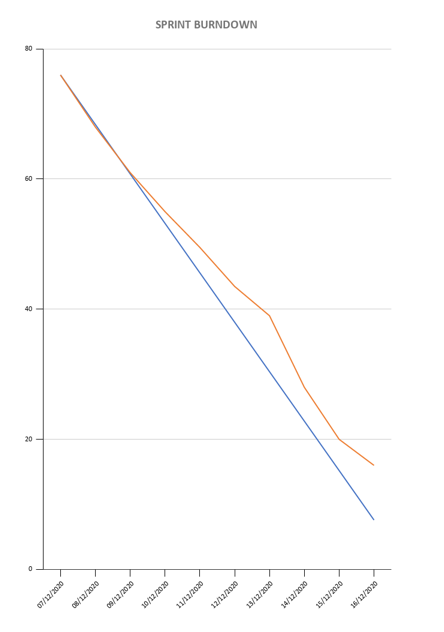
Le diverse user stories sono state divise in task atomici e facilmente risolvibili a cui è stato assegnato un punteggio in story points secondo la funzione di Fibonacci e attraverso il processo di Planning poker. I task a loro volta sono stati raggruppati in varie sprint, ma soltanto le prime due sprint verranno effettivamente rilasciate alla fine del lavoro. Si può notare che la maggior parte dei task della prima sprint sono relativi alla creazione del database e alle relative schermate GUI: infatti, quest’ultimo è il cuore dell’applicativo poiché memorizza tutti i pianificatori, tutti i manutentori e tutte le attività di manutenzione.

La maggior parte dei task previsti nella seconda sprint sono relativi alla scrittura di informazioni relative ai luoghi, agli strumenti, alle documentazione sulle procedure e altre informazioni secondarie all’interno del database e alle schermate GUI che lo permettono.

**Burndown Chart**

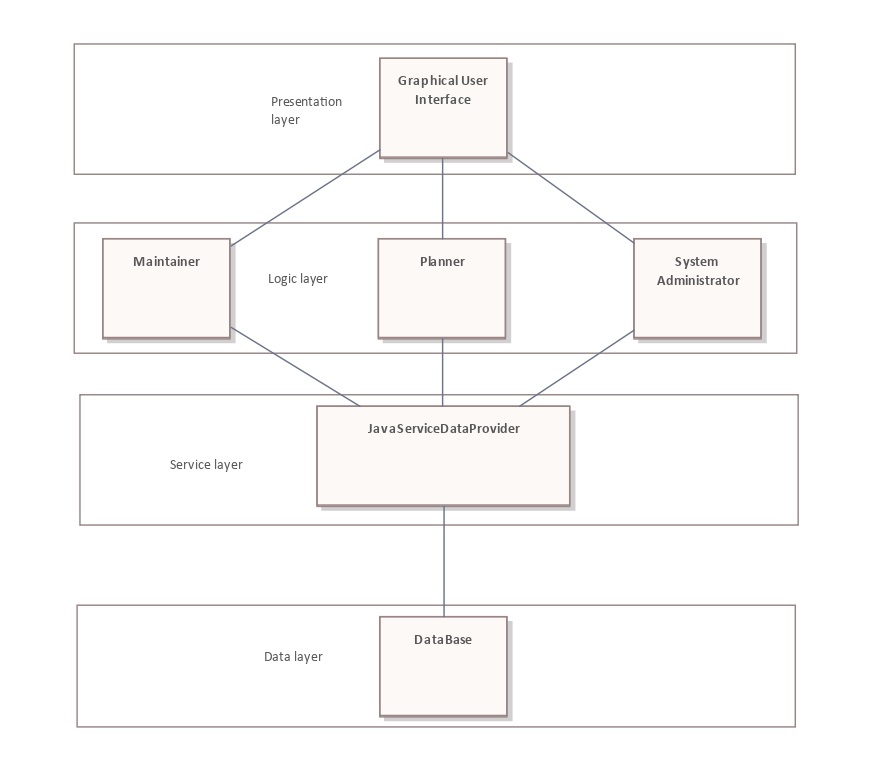
Un burndown chart è una rappresentazione grafica del lavoro da fare su un progetto nel tempo. Il lavoro rimanente (o backlog) è indicato sull'asse verticale (sotto forma di *story point*) e il tempo sull'asse orizzontale. Il diagramma rappresenta una serie storica del lavoro da fare. Esso è utile per prevedere quando avverrà il completamento del lavoro. Di seguito vengono mostrati i burndown chart relativi alle due Sprint effettuate.



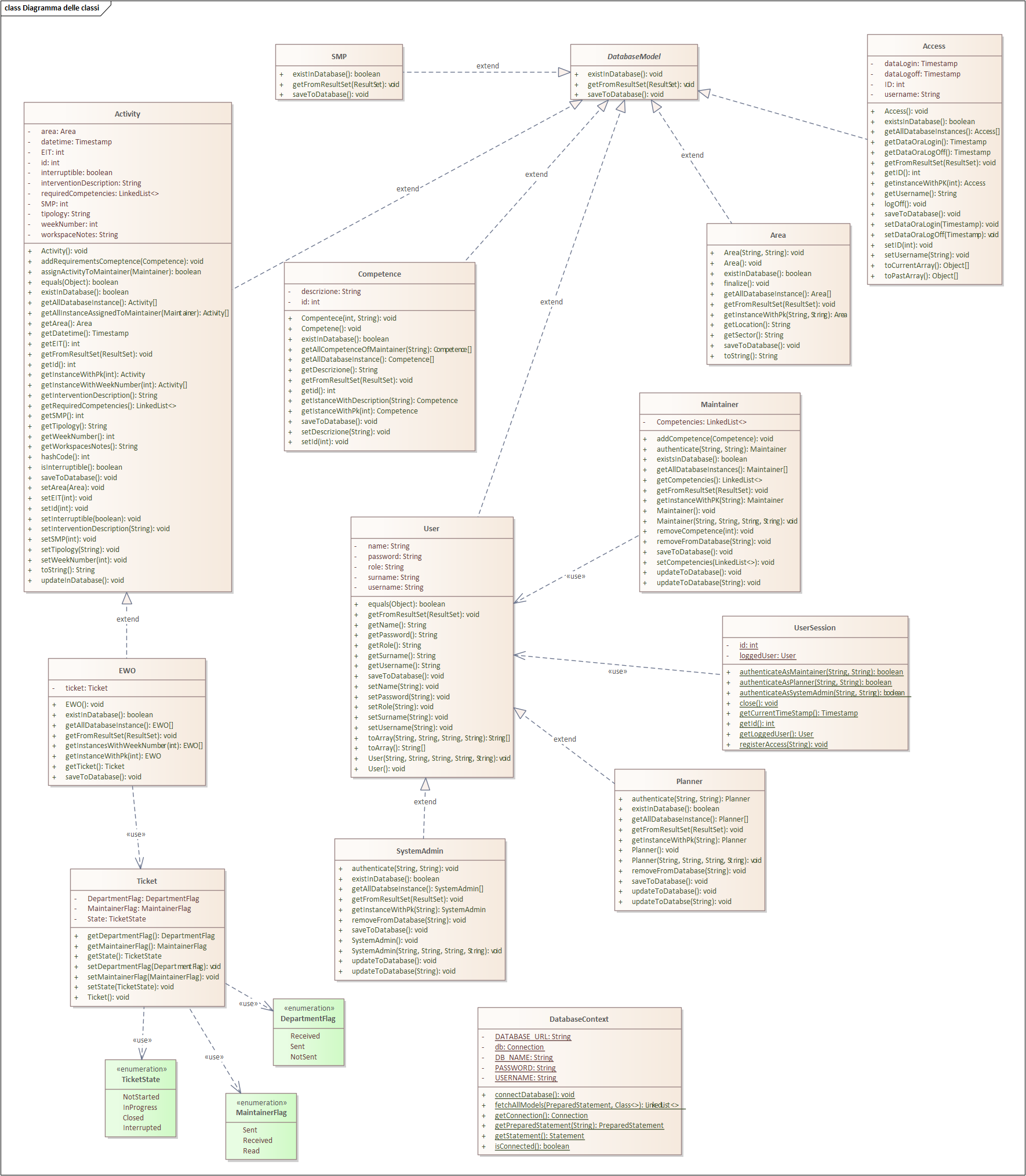


**Architettura dell’applicazione**

Si è deciso di usare un’architettura di tipo layered: le operazioni sul database sono effettuate dal componente JavaServiceDataProvider, che è possibile utilizzare attraverso una GUI, secondo il punto di vista di una delle tre tipologie di utenti, ovvero Maintainer, Planner e System Administrator (è colui che ha il privilegio di aggiungere, rimuovere e modificare gli utenti dell’applicativo).



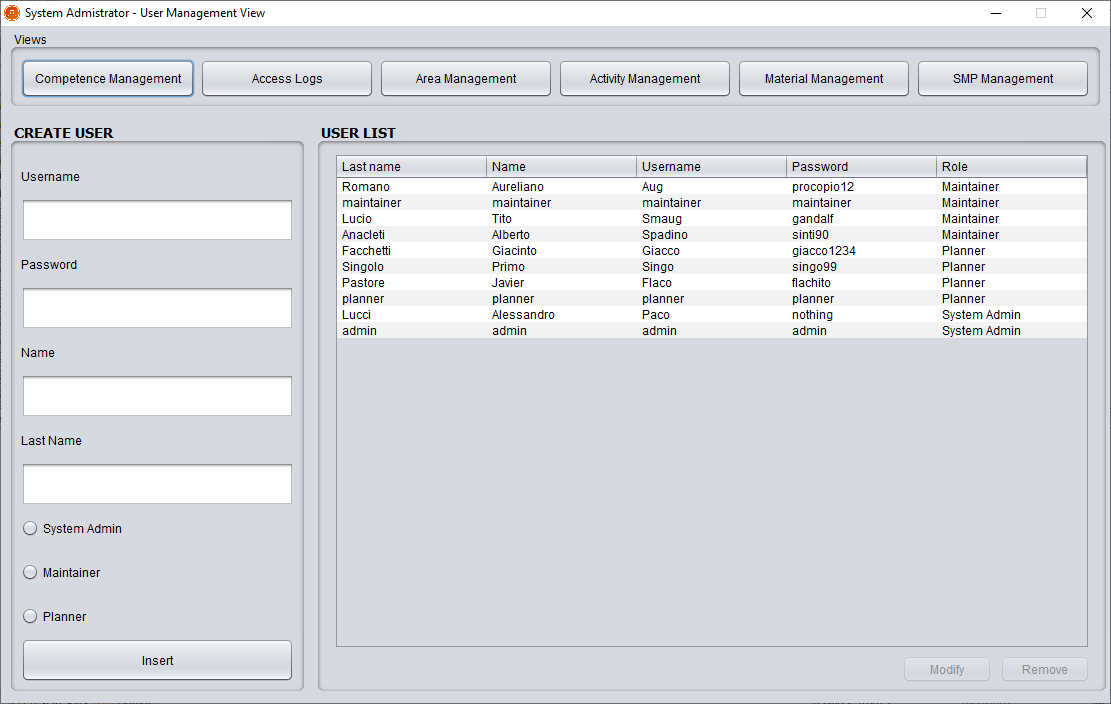
Per avere una visione completa di tutte le classi Java, è stato realizzato, tramite il software Enterprise Architect, un diagramma delle classi in linguaggio UML in cui vengono mostrate le relazioni tra le varie classi. Ogni classe ha una sua funzionalità, ad esempio, la classe *DatabaseContext* gestisce le operazioni di collegamento tra l’applicazione e il database POSTGRESQL; la classe *UserSession* gestisce le operazioni di login e di logout di una determinata tipologia di utente; per quanto riguarda le classi del package *it.unisa.team8se.models*, esse permettono di fare operazioni di manipolazione e di ricerca (query) sul database; il package it.unisa.team8se.gui contiene le classi che rappresentano le diverse schermate dell’applicazione e i diversi metodi per interagire col database



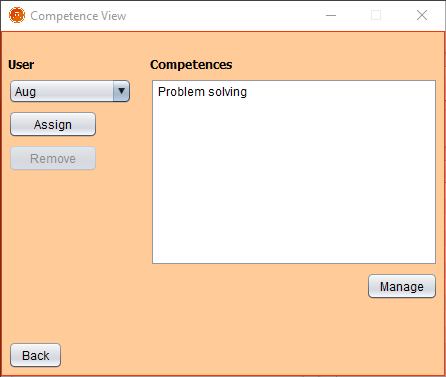
Per poter accedere all’applicazione è necessario autenticarsi attraverso la schermata di login iniziale in cui bisogna inserire username e password e specificare il ruolo dell’utente attraverso un menu a tendina situato sotto l’etichetta ROLE.

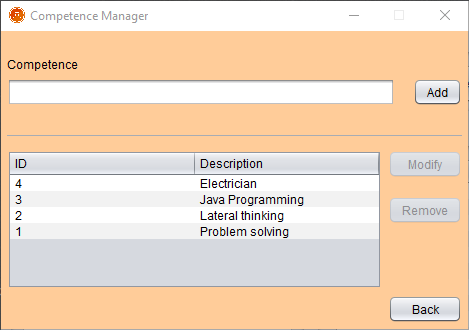


Se si accede all’applicazione con il ruolo di System Admin, la prima schermata che appare mostra tutte le operazioni che un amministratore può compiere, infatti, egli può aggiungere un nuovo utente, modificare un utente, rimuovere un utente. Nella parte superiore di questa schermata, chiamata *User Management View*, sono presenti dei bottoni che permettono di raggiungere le altre schermate dell’applicazione le quali permettono una gestione dei dati e un’organizzazione completa delle attività da svolgere.

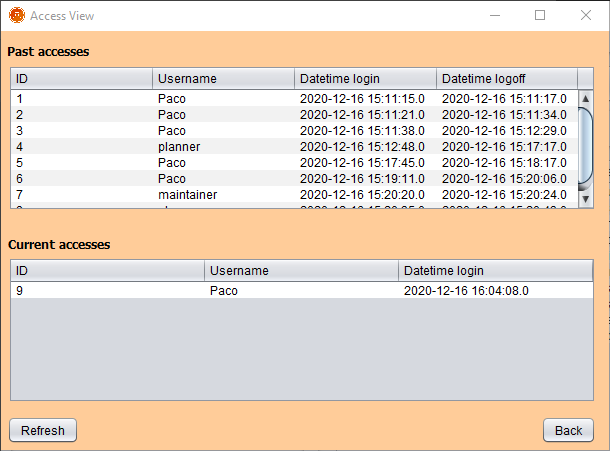


Dalla schermata *Competence View* è possibile visualizzare le diverse competenze di ogni utente, aggiungere o rimuovere nuove competenze ad un utente. Il bottone *Manage* conduce ad una nuova schermata chiamata *Competence Manager* che permette di prevedere nuove competenze o di modificare quelle già previste

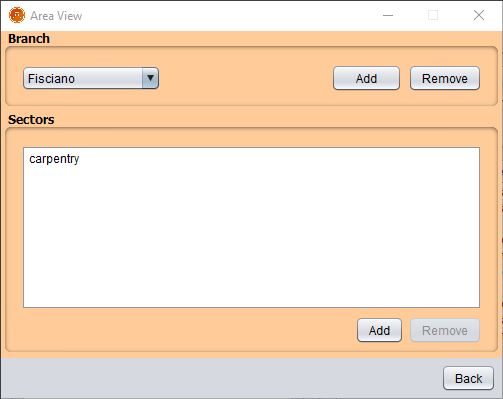




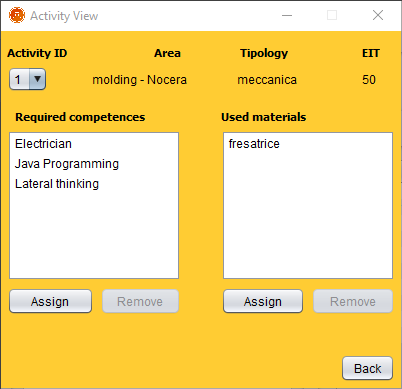
Ogni utente ha uno storico proprio di accessi effettuati, sia passati che attuali, visibile tramite due tabelle nella schermata *Access View* denominate rispettivamente *Past accesses* e *Current accesses*. Inoltre, vengono specificate data e ora di login e di logoff per gli accessi passati, mentre, per gli accessi correnti viene mostrata solo data e ora di login.



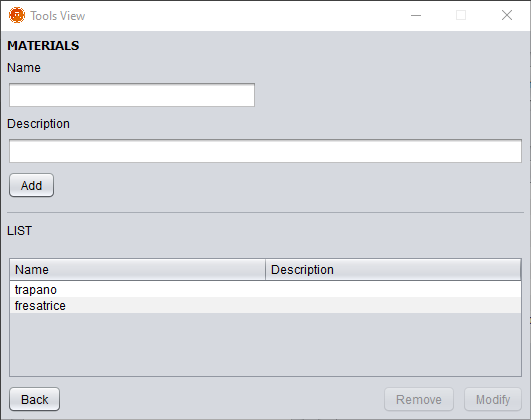
In *Area View* si aggiungono o rimuovono nuove filiali con i relativi settori di competenza i quali possono anch’essi essere aggiunti o rimossi in base alle esigenze dell’utente.



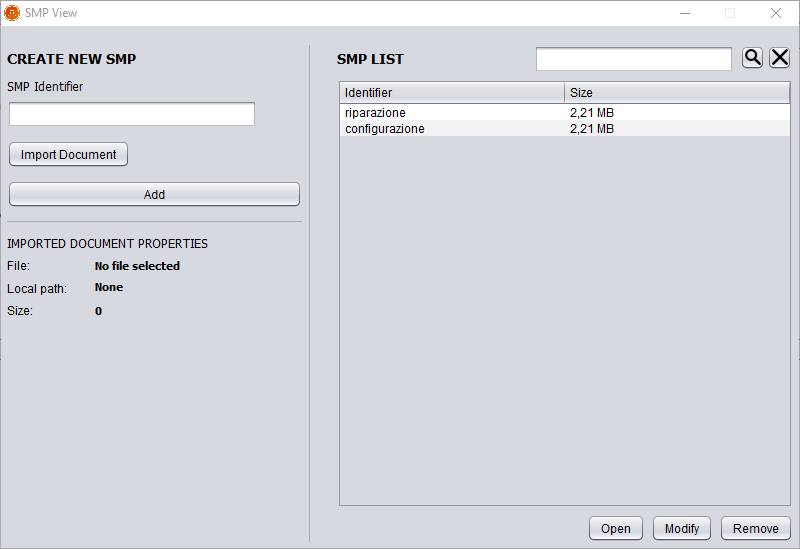
In *Activity View* è possibile assegnare o rimuovere una competenza da un’attività e scegliere quali materiali devono essere utilizzati per svolgerla o rimuovere quelli che non servono.



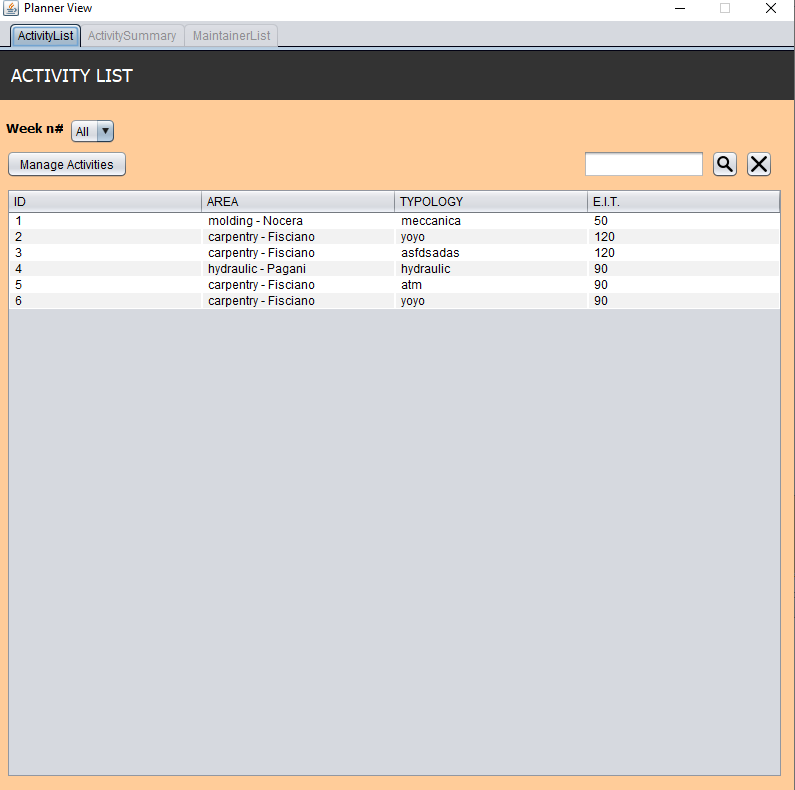
Nuovi materiali possono essere aggiunti nella schermata *Tools View*. Inoltre, è possibile sia modificare sia rimuovere un materiale.



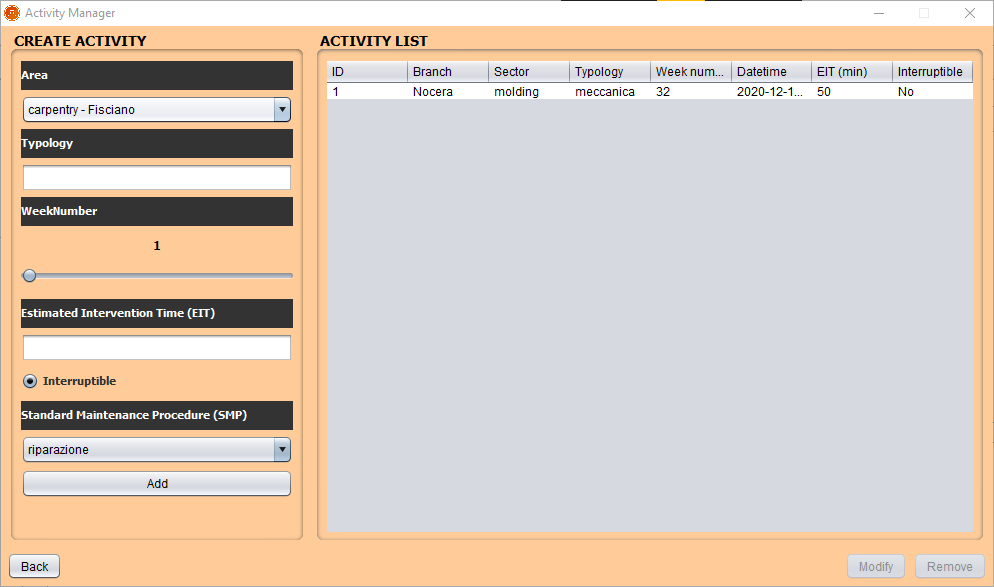
Infine, come ultima schermata di gestione dati accessibile sempre dall’interfaccia *User Management View*, abbiamo una visione generale dei file SMP, con la possibilità di importare un nuovo documento SMP, di aprire, modificare o rimuovere un documento secondo le esigenze dell’utente.



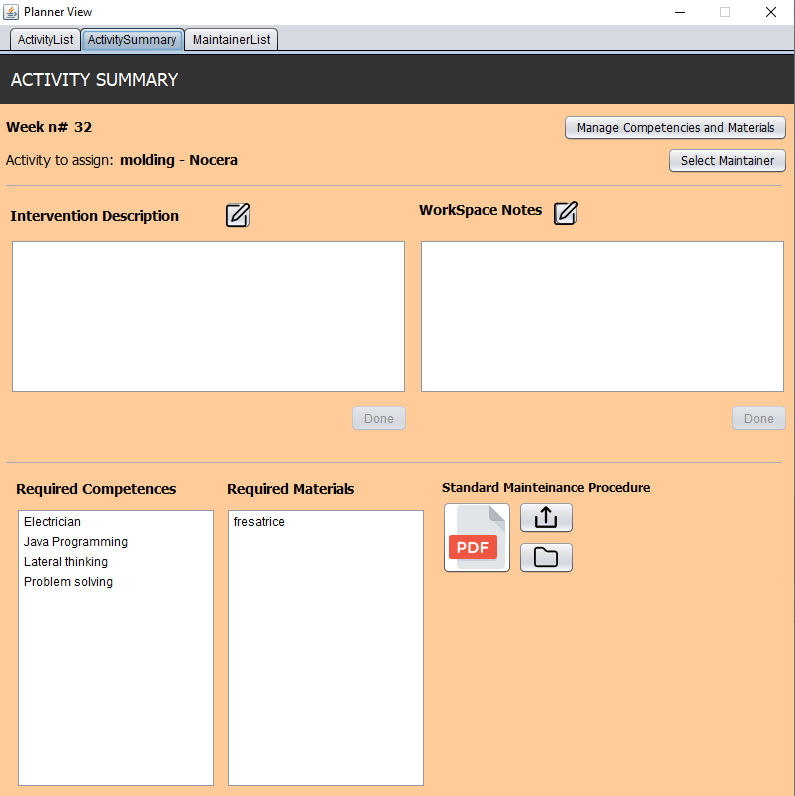
L’utente Planner, appena effettuato l’accesso, ha a disposizione una schermata in cui visualizza tutte le attività disponibili e attraverso il bottone *Manage Activities* viene portato sulla schermata *Activity Manager* da cui può creare una nuova attività.



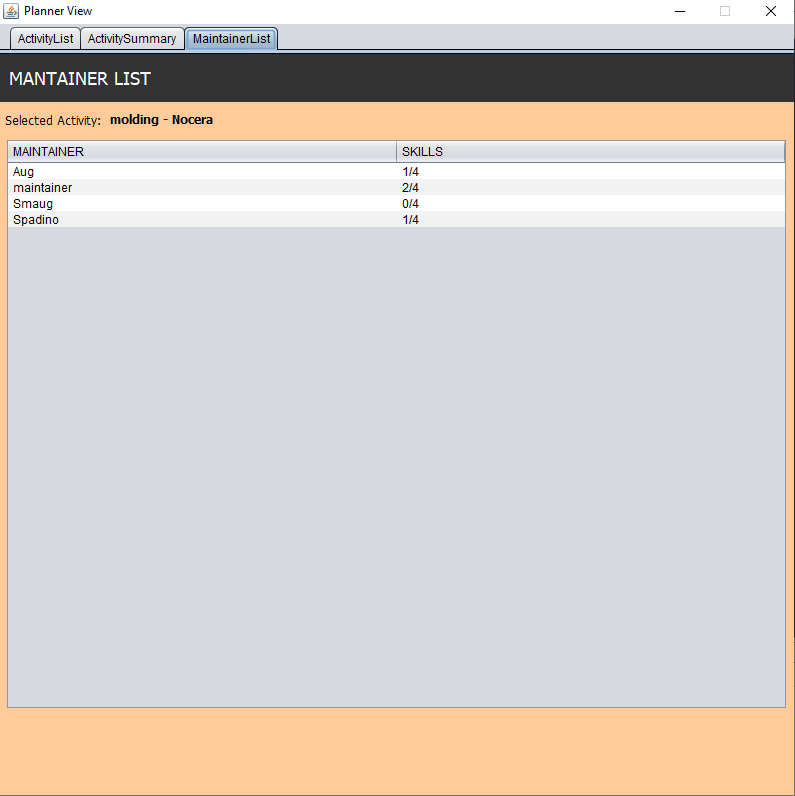
Come anticipato in precedenza, la schermata *Activity Manager* permette di creare una nuova attività specificando tutti i campi necessari per identificarla, infatti, è possibile inserire l’area dove si svolge l’attività, la tipologia, il numero della settimana corrispondente, il tempo stimato per compierla, la possibilità di interromperla e inoltre è possibile aggiungere un documento SMP relativo all’intervento specifico da eseguire.



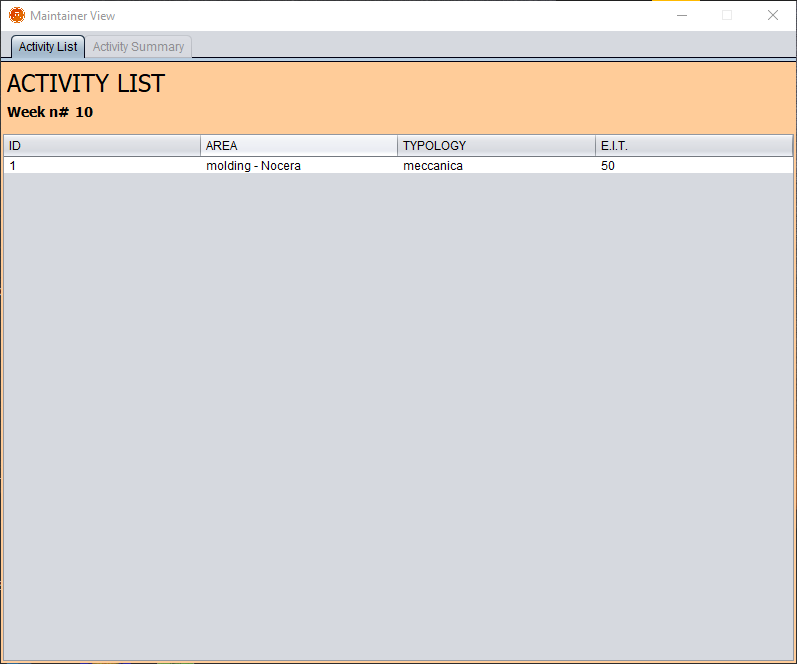
La schermata *Activity Summary*, accessibile dopo aver selezionato un’attività da Activity List, mostra un riepilogo dell’attività selezionata in cui vengono specificate le competenze richieste per svolgere quella determinata attività e i materiali richiesti da utilizzare. Da questa schermata è possibile, tramite il bottone *Manage Competences and Materials*, andare sulla schermata *Activity View* vista in precedenza, con l’unica differenza che ora l’attività non può essere cambiata nel menu a tendina Activity ID e tramite il bottone *Select Maintainer* accedere alla Maintainer List da cui scegliere il maintainer da assegnare a quella attività. Inoltre, è possibile caricare un documento PDF corrispondente al file SMP richiesto per quel determinato intervento. Se si vuole aggiungere una descrizione dell’intervento oppure scrivere delle note riguardanti il lavoro, sono presenti anche due blocchi note che permettono di farlo.

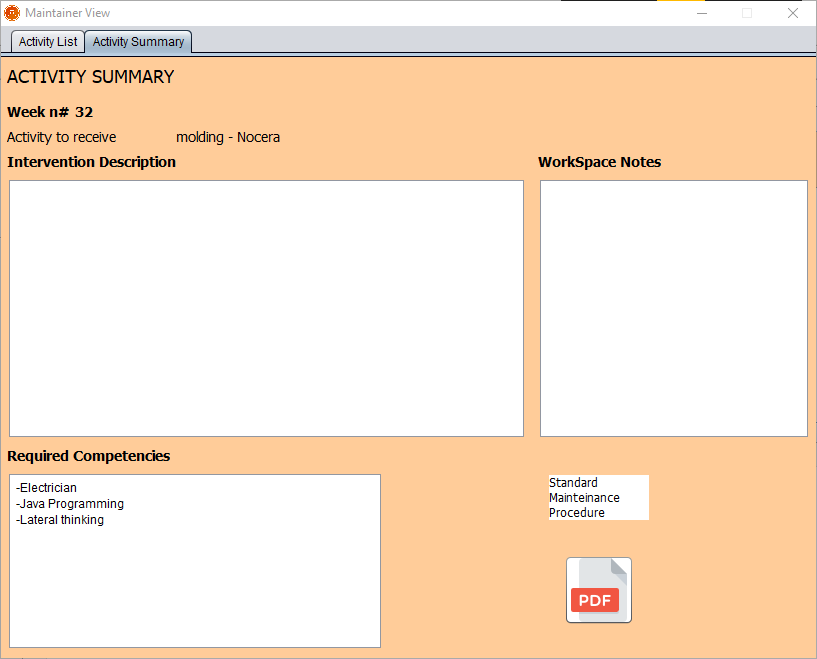


In *Maintainer List*, sono mostrati tutti i maintainer disponibili che possono essere assegnati ad una determinata attività selezionata e la colonna *Skills* indica le *skills* che hanno i maintainer tra quelle richieste per quell’attività.



L’utente Maintainer, dopo aver eseguito il login, accede alla schermata delle attività a lui assegnate e attraverso *Activity Summary* egli può vedere un riepilogo dell’attività, infatti, può visualizzare le competenze richieste per quella determinata attività, scrivere delle note e aprire un file SMP.





**Design Pattern**

Nel progetto sono stati applicati alcuni pattern, in particolare è stato applicato un pattern creazionale chiamato *Singleton*, con il quale è stata creata una sola istanza di esecuzione per tutta la durata del programma e quindi si è scelto di dichiarare un oggetto statico nella classe *DatabaseContext* su cui richiamare tutte le operazioni appartenenti alla classe. Un altro pattern applicato è stato *Decorator*, che appartiene alla classe dei pattern strutturali, visibile quando il contenuto del *TextField* diventa troppo grande (ad esempio vengono inseriti molti oggetti), compare una *scrollbar* che permette di vedere tutti gli oggetti inseriti nel *TextField*.