# 4. Funzionalità da testare e non testare

Di seguito sono elencate tutte le funzionalità da testare in relazione ai sottosistemi a cui appartengono:

1. Package Utente:
   * LoginControl.java
   * PasswordControl.java
   * RegistrationControl.java
2. PackageGestoremarketing:
   * InsPrezzoControl.java
3. PackageMagazziniere:
   * MagazziniereControl.java
4. PackageProprietario:
   * AdminControl.java
5. PacakgeAcquisto:
   * CartControl.java
6. Package Cata:
   * AggiungiCarta.java
7. PackageIndirizzo:
   * AggiungiIndirizzo.java
8. PackageProdotto:
   * ProductControl.java

# 10. Pianificazione dei test

Il team dedito alla compilazione del testing deve essere composto da persone che hanno una completa conoscenza del dominio applicativo e del dominio delle soluzioni del sistema. Inoltre, devono conoscere in maniera completa tutte le tecniche di testing utilizzate e nominate all’interno del Test Plan e del Test case specification. Le attività che comportano tale fase dovranno essere fatte nei tempi, nei costi e nei vincoli di qualità specificati.

Il sistema revisionato dovrà, successivamente alla correzione, essere verificato attraverso altri casi di test per consentire di assicurarsi che le modifiche sono state compiute in maniera corretta e verificare se tali cambiamenti hanno introdotto nuovi errori. L’attività di testing è fondamentale nello sviluppo di un sistema software in quanto la mancanza di tale attività o una cattiva gestione della stessa può portare al completo fallimento del sistema e, in casi estremi, dell’intero progetto. Data l’importanza del testing, la schedulazione delle sue attività sono fondamentali.

## 10.1 Determinazione dei ruoli

Tutta l’attività di testing viene eseguita da Giuseppe Caiazzo. Le attività relative al testing di unità come correzioni e cambiamenti verranno delegate agli sviluppatori che si occuperanno delle modifiche a livello implementativo. Tale organizzazione serve principalmente per alleggerire il carico di lavoro da parte del tester che potrà dedicarsi maggiormente sul lavoro di testing funzionale.

## 10.2 Determinazione dei rischi

Nel vaso in cui vi sia un’elevata quantità di errori, il sistema rischia di essere consegnato in ritardo. Per sovvenire a questa situazione, nel caso in cui si verifichi, si è deciso di effettuare una pianificazione verticale dei testi funzionale. Tale approccio consente di rilasciare un numero minore di funzionalità nei tempi previsti, ma in maniera completa e funzionale.

## 10.3 Decomposizione gerarchica del sistema

La divisione gerarchica del sistema è stata mappata in 3 livelli gerarchici come nel seguente diagramma:

## 10.4 Organizzazione delle attività di testing

Le attività di testing verranno organizzate secondo uno schema che effettuerà una divisone funzionale di tipo verticale. In questo modo al termine di ogni attività si avrà una funzionalità completamente testata nei suoi livelli gerarchici. I vantaggi principali sono che, in caso di ritardi dovuti al ritrovamento di numerosi failure, il sistema verrà rilasciato con meno componenti, ma interamente testate e funzionanti.