

ITC

Integration Test Case

|  |  |
| --- | --- |
| Riferimento |  |
| Versione | 1.0 |
| Data | 28/01/2021 |
| Destinatario | AutoErre S.r.l |
| Presentato da | Caprio Mattia, Pepe Sara, Torino Francesco Maria, Iodice Michele Attilio, Mori Mattia, Sarro Antonio |
| Approvato da | Di Dario Dario |

# Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Data** | **Versione** | **Descrizione** | **Autori** |
| 26/01/2021 | 1 | Prima stesura ITC | Mori Mattia |

# Sommario

[Revision History 2](#_Toc63113084)

[Sommario 3](#_Toc63113085)

[1. Introduzione 4](#_Toc63113086)

[2. Riferimenti 4](#_Toc63113087)

[3. Test di integrazione 4](#_Toc63113088)

[3.1 Approccio di Integration Testing 4](#_Toc63113089)

[3.2 Componenti da testare 4](#_Toc63113090)

[4. Descrizione dell’esecuzione dei test 5](#_Toc63113091)

[5. Pass/Fail Criteria 6](#_Toc63113092)

# 1. Introduzione

Il Testing di integrazione ha lo scopo di identificare errori e malfunzionamenti nel sistema, eseguendo vari test su tutte le funzionalità del sito e controllando che l’esito atteso e quello risultato dal testing siano lo stesso.

# 2. Riferimenti

La corretta integrazione delle funzionalità da testare è stata indicata nel Test Case Plan, dove sono presenti i Test secondo la suddivisione proposta nel System Design Document. I criteri di accettazione sono stati indicati nello Statement of Work, mentre la priorità della funzionalità è presente nel Requirement Analysis Document.

I documenti di riferimento sono:

* C11\_SDD\_EasyLease\_v2
* C11\_TCS\_EasyLease\_v3
* C11\_RAD\_EasyLease\_v3.5
* C11\_SOW\_EasyLease\_v0.2

# 3. Test di integrazione

## **3.1 Approccio di Integration Testing**

Per il testing di integrazione, è stata adottata una strategia “Bottom-up”, secondo la quale vengono prima collaudati i metodi al livello più basso della gerarchia, passando poi al livello superiore e testandolo sfruttando le funzionalità del livello precedente e risalendo in questo modo fino all’intero sistema.

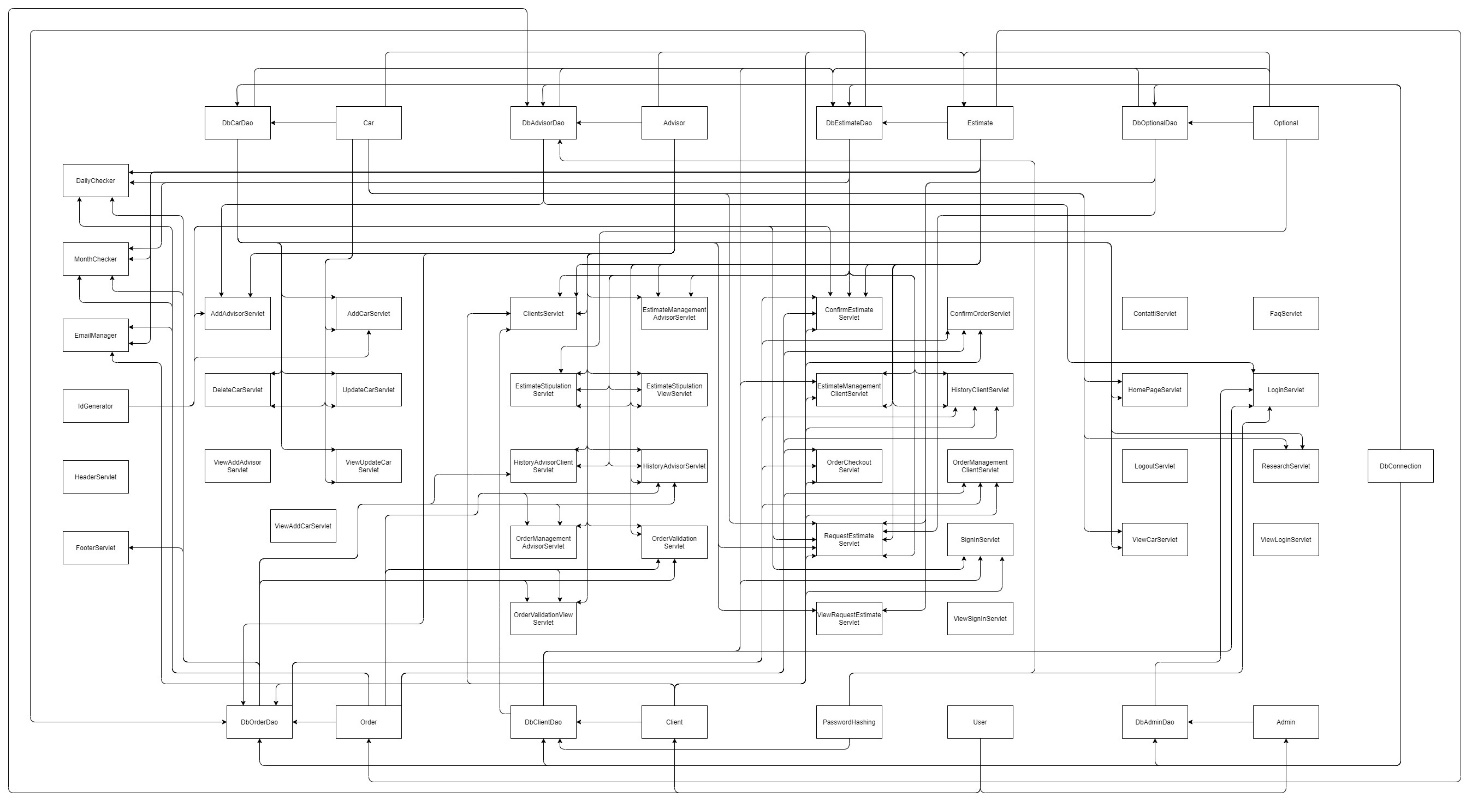
## **3.2 Componenti da testare**

* **Model**
  + Admin
  + DbAdminDao
  + Advisor
  + DbAdvisorDao
  + Car
  + DbCarDao
  + Client
  + DbClientDao
  + DbConnection
  + Estimate
  + DbEstimateDao
  + Optional
  + DbOptionalDao
  + Order
  + DbOrderDao
* **Control** 
  + AddAdvisorServlet
  + AddCarServlet
  + DeleteCarServlet
  + UpdateCarServlet
  + ViewAddAdvisorServlet
  + ViewAddCarServlet
  + ViewUpdateCarServlet
  + ClientsServlet
  + EstimateManagementAdvisorServlet
  + EstimateStipulationServlet
  + EstimateStipulationViewServlet
  + HistoryAdvisorClientServlet
  + HistoryAdvisorServlet
  + OrderManagementAdvisorServlet
  + OrderValidationServlet
  + OrderValidationViewServlet
  + ConfirmEstimateServlet
  + ConfirmOrderServlet
  + EstimateManagementClientServlet
  + HistoryClientServlet
  + OrderCheckoutServlet
  + OrderManagementClientServlet
  + RequestEstimateServlet
  + SignInServlet
  + ViewRequestEstimateServlet
  + ViewSignInServlet
  + FooterServlet
  + HeaderServlet
  + ContattiServlet
  + FaqServlet
  + HomePageServlet
  + LoginServlet
  + LogoutServlet
  + ResearchServlet
  + ViewCarServlet
  + ViewLoginServlet
  + DailyChecker
  + MonthChecker
  + EmailManager
  + IdGenerator
  + PasswordHashing

# 4. Descrizione dell’esecuzione dei test

Per l’implementazione dei Test di integrazione abbiamo usato Mockito, un framework che permette di generare un Mock di una classe, dichiarandone il comportamento e facilitando alcuni tipi di test: è stato nel nostro caso principalmente utilizzato per simulare componenti impossibili da testare (come HttpServletRequest, HttpServletResponse, HttpSession, ServletContext e RequestDispatcher), utilizzando il vero database ed includendo opportune funzioni di rollback per farlo tornare allo stato iniziale nei casi in cui vengono apportate modifiche.

Il precedente diagramma, per una migliore visibilità, è visionabile nel file **C11\_IntegrationTestPlanDiagram\_EasyLease** ed è stato caricato su Google Drive al seguente link:

<https://drive.google.com/file/d/1l-PEV6zQjY9F63Ck6Cpb7tubeuZNtCMX/view?usp=sharing>

# 5. Pass/Fail Criteria

L’esito dei test sarà Passed se il loro risultato atteso non corrisponde a quello ottenuto con il testing, identificando dunque dei malfunzionamenti, mentre sarà Failed se i due risultati corrispondono e non vengono dunque riscontrati errori.

Verrà considerato valido un testing completo che rispetta le seguenti caratteristiche:

* Raggiungimento di una branch coverage pari o superiore al 75%
* Testing di tutte le funzionalità indicate come ad Alta Priorità