1 Introduzione

1.1Scopo del sistema

Lo scopo del sistema è quello di fornire supporto all'azienda AutoErre S.r.l. consentendo una gestione quanto più semplice possibile del processo di noleggio auto. Ad oggi un sistema similare non esiste, alcuni competitor usano un sistema molto più semplificato basato sulla singola azione di visualizzazione delle auto.
L'esigenza di EasyLease nasce dal bisogno da parte dell'azienda AutoErre di semplificare tutto il processo di noleggio di un'auto che va dalla scelta di essa da parte di un cliente, alla stipulazione finale di un contratto con il relativo pagamento. Inoltre, con EasyLease l'azienda vuole migliorare ed ampliare la visibilità del proprio business.

EasyLease da un lato si prefigge lo scopo di rendere il meccanismo di noleggio auto da parte di un cliente il più semplice e veloce possibile senza l'esigenza di recarsi fisicamente in concessionaria portando avanti l'intera operazione online, dall'altro lato ha l'obbiettivo di permette ai consulenti dell'azienda di avere una gestione dei clienti più semplice e chiara.

Il sistema progettato è una web app alla quale avranno accesso gli amministratori dell'azienda, i consulenti di essa e i vari utenti che vorranno utilizzarla. Il sistema può essere suddiviso in quattro aree:

- Area Amministratore: vista di auto e consulenti, aggiunta auto, consulenti, eliminazione auto, consulenti, modifica auto.
- Area Consulenti: vista auto, gestione e vista richieste preventivi e ordini.
- Area Cliente: vista auto, gestione preventivi, ordini.
- Area Utente: ricerca e vista auto.

Il sistema deve fornire un metodo di autenticazione sicuro in modo che un utente non possa aver effettuato più di un accesso contemporaneamente e che un cliente non finisca nell'area riservata di un consulente o amministratore e viceversa. Il sistema inoltre dovrà essere facile da apprendere e intuitivo da utilizzare, consentendo una navigazione fluida e permettendo l'utilizzo del sistema anche senza il consulto della documentazione.

1.2 Design Goals

I Design Goals sono organizzati in cinque categorie: Performance, Dependability, Cost, Maintenance, End User and Criteria. I Design Goals identificati nel nostro sistema sono i seguenti:

Criteri di performance

- Tempo di risposta:
  Il software deve consentire una navigazione rapida a tutti i tipi di utenti, quindi tempi di risposta minimi nello svolgimento delle funzionalità da esso offerte.
- Memoria
  La memoria complessiva del sistema dipenderà dalla memoria utilizzata per il mantenimento del Database.

Criteri di affidabilità

● Robustezza:

Il sistema informerà l'utente di eventuali input errati attraverso messaggi di errore.

● Affidabilità:

Il sistema deve garantire l'affidabilità dei servizi proposti. I risultati visualizzati saranno sempre attendibili. Per quanto riguarda le auto che sono visibili sul sistema esse rispecchieranno la reale disponibilità da parte dell'azienda AutoErre. Ad esempio, se l'azienda ha in magazzino un nuovo tipo di auto essa verrà aggiunta al sistema, così come se un'auto presente sul sistema non è più disponibile dall'azienda essa non verrà visualizzata.

Il processo di login da parte di tutti gli utenti sarà gestito in modo affidabile, assicurando il corretto funzionamento del sistema, e garantendo l'accesso alle relative aree private, e l'utilizzo di determinate funzioni solo al tipo di utente per cui quell'area o quella funzione è riservata.

● Disponibilità:

Una volta online, il sistema sarà disponibile a tutti gli utenti (sia essi registrati o meno), i consulenti abilitati da un amministratore e a tutti gli amministratori del sistema.

● Tolleranza ai guasti:

Il sistema potrebbe riscontrare fallimenti dovuti a varie cause tra cui un sovraccarico di dati nel database. Per risolvere il problema, periodicamente sarà previsto un salvataggio dei dati sotto forma di codice SQL necessario alla rigenerazione del Database e periodicamente verrà effettuata un'eliminazione all'interno del database degli ordini scaduti, dei preventivi vecchi un dato periodo e degli ordini e/o preventivi che sono stati rifiutati dall'utente e/o consulente.

● Sicurezza:

L'accesso al sistema sarà garantito mediante e-mail e password; mentre l'accesso all'area privata utente è concessa solo agli utenti che hanno eseguito il processo di registrazione sulla medesima piattaforma; l'accesso all'area riservata degli amministratori è consentita ai soli utenti identificati come amministratori, mentre l'accesso all'area riservata dei consulenti è consentita solo agli utenti che sono stati aggiunti da un amministratore come consulenti.

Criteri di costi

- Costo di sviluppo:

  È stimato un costo complessivo di 350 ore per la progettazione e lo sviluppo del sistema (50 per ogni membro del progetto).

Criteri di manutenzione

- Estensibilità:

  È possibile aggiungere nuove funzionalità al sistema, in seguito di nuove esigenze da parte dell'azienda o dall'avvento di nuove tecnologie che migliorino il sistema.

- Adattabilità:

  Il sistema è fatto su misura per l'azienda AutoErre e quindi funziona solo per essa e tutti gli utenti che intendono interagire con essa.

- Portabilità:

  L'interazione con il sistema avviene attraverso un browser a prescindere da quale esso sia, quindi possiamo definire il sistema come portabile. Poiché il sistema viene sviluppato come una web application, esso è accessibile da qualunque dispositivo, che sia esso mobile o fisso, che abbia un sistema operativo windows o altro, purché abbia un browser installato. Questa caratteristica garantisce la portabilità dello stesso.

- Tracciabilità dei requisiti:

  La tracciabilità dei requisiti sarà possibile grazie ad una matrice di tracciabilità, attraverso la quale sarà possibile retrocedere al requisito associato ad ogni parte del progetto. La tracciabilità sarà garantita dalla fase di progettazione fino al testing.

Criteri di usabilità

- Usabilità:

  Il sistema sarà di facile comprensione e utilizzo, permettendo di effettuare in modo semplice e immediato le varie operazioni grazie a un'interfaccia intuitiva, di facile comprensione e utilizzo. L'intuitività del sistema è garantita da un'ottima prevedibilità di quelle che sono le azioni dell'utente, cioè la risposta del sistema ad un'azione utente sarà corrispondente alle aspettative. Inoltre, tramite l'utilizzo di apposite immagini il sistema sarà ancora più chiaro e intuitivo.

- Utilità:

  Il lavoro dell'utente verrà supportato nel miglior modo possibile dal sistema, infatti l'utente compirà le operazioni necessarie per un leasing comodamente dal proprio dispositivo senza alcun bisogno di recarsi in concessionari. Faciliterà e velocizzerà il lavoro dei consulenti e amplierà la visibilità sul mercato dell'azienda AutoErre S.r.l.

1.2.1 Design Trade-off

Performance vs Memoria:

Il sistema deve poter garantire risposte rapide anche se con un uso della memoria eccessivo. Ciò significa che verranno introdotte delle ridondanze per evitare interrogazioni costose in termini di performance e non ci saranno delle eventuali immediate eliminazioni di dati.

Tempo di risposta vs **Affidabilità**

Il sistema sarà implementato in modo tale da preferire l'affidabilità al tempo di risposta, in modo tale da garantire una risposta del sistema consistente e non errata, contenente le giuste informazioni a discapito dal tempo impiegato per produrla (soprattutto nel caso della compilazione dei preventivi).

Disponibilità vs Tolleranza ai guasti

Il sistema deve essere sempre disponibile all'utente in caso di errore di una funzionalità a media o bassa priorità, anche a costo di rendere non disponibile quest'ultima per un lasso di tempo, informando l'utente con delle opportune notifiche.

Ovviamente se questa è una funzionalità Core, il sistema verrà messo in manutenzione fin quando il guasto non verrà risolto.

Criteri di manutenzione vs Criteri di performance

Il sistema nella sua implementazione preferirà la manutenibilità alla performance in questo modo sarà facilitato il processo di aggiornamento del software agli sviluppatori anche se a discapito delle performance del sistema.

Di seguito è riportata una tabella che mostra i design goal preferiti nei trade off. Il grassetto indica la preferenza.

Trade Off	
Performance	Memoria
Affidabilità	Tempo di risposta
Disponibilità	Tolleranza ai guasti
Criteri di manutenzione	Criteri di performance

1.3 Definizioni, acronimi e abbreviazioni

Definizioni

Cliente: rappresenta un generico cliente, che sia intenzionato ad usare la piattaforma, utilizzando le operazioni descritte sopra per finalizzare un ordine. Ogni cliente ha un proprio account registrato sulla piattaforma in cui saranno specificate le seguenti informazioni:

1. Nome;
2. Cognome;
3. E-mail;
4. Password;
5. Dati per il pagamento;

Consulente: rappresenta un lavoratore dipendente dell'azienda, il quale compito è quello di inoltrare il cliente nella stipulazione di un preventivo, quindi di fargli comprendere il costo a cui andrà in contro finalizzando l'ordine nel momento del pagamento. Ogni consulente avrà un proprio account in cui sono specificate le seguenti informazioni:

1. Id consulente;
2. Nome;
3. Cognome;
4. E-mail;
5. Password;
6. Numero ufficio;

Amministratore: rappresenta un titolare o una figura che ne fa le veci, incaricato di gestire l'intera piattaforma. Ogni amministratore avrà le seguenti informazioni:

1. Nome;
2. Cognome;
3. E-mail;
4. Password;

Autovettura: rappresenta un'automobile e, come tale avrà le seguenti informazioni:

1. Id automobile;
2. Tipo di auto;
3. Marca;
4. Modello;
5. Caratteristiche tecniche;
6. Descrizione;

Preventivo: rappresenta un form che contiene le informazioni sull'auto e sul noleggio di cui un cliente ha fatto richiesta e successivamente un consulente ha compilato

Specificati:

1. Auto
2. Optional
3. Prezzo
4. Consulente
5. Cliente

Ordine: rappresenta un preventivo di cui è avvenuta (o deve avvenire) la conferma sia da parte del consulente che del cliente dove sono specificati:

1. Data inizio
2. Data fine
3. Preventivo a cui fa riferimento

Richiesta preventivo: rappresenta l'azione che effettua il cliente per richiedere il preventivo di una determinata auto.

Conferma preventivo: rappresenta l'azione di conferma di un preventivo da parte di un cliente

Stipulazione preventivo: rappresenta l'azione da parte del consulente di compilare con i relativi costi un preventivo.

Conferma ordine: rappresenta l'azione da parte di un consulente di confermare un ordine fatto da uno dei clienti da lui preso in carico.

Conferma data: rappresenta l'azione da parte di un cliente di confermare la data di consegna di un'auto o meno.

Preventivo preso in carico: rappresenta il fatto che un consulente ha preso in carico il preventivo ed eventualmente il successivo ordine di un determinato cliente da lui scelto.

Quando un utente è "loggato" vuol dire che ha effettuato l'autenticazione.

Acronimi

SDD = System Design Document (Documento di Progettazione del Sistema)

HW = Hardware

SW = Sowftware

DBMS = DataBase Management System

A = Amministratore

AUT = Autenticazione

ACC = Account

CL = cliente

C= consulente

NA = Nessuna

CD = Class Diagram

GUI = Graphical User Interface (Interfaccia utente)

DBA = Data Base Administrator (Amministratore del DataBase)

Il "+" sta per pseudo-requisiti o vincoli del sistema:

- Aggiungere
- Aggiungere

ALTRI ACRONIMI DA AGGIUNGERE IN CORSO D'OPERA.

1.4 Riferimenti

- RAD_V2.docx – Requirements Analysis Document
- http://www.easylease.it
- Bernd Bruegge & Allen H.Dutoit, Object-Oriented Software Engineering: Using UML, Patterns and Java, (2nd edition), Prentice-Hall, 2004.
- Ian Sommerville, Ingegneria del software, (10a edizione), Pearson, 2017.

1.5 Panoramica Documento

Il documento è diviso in quattro sezioni, a ciascuna delle quali è assegnato un compito ben preciso:

1. Introduzione: in questa parte viene fornita una descrizione del sistema, in modo da fornire una visione generale a chiunque legga questo documento. In particolare, viene illustrato lo scopo del sistema ed i suoi design goals. Inoltre, per evitare ambiguità e facilitare la comprensione, vengono illustrati gli acronimi, le definizioni e le abbreviazioni che sono state utilizzate nella stesura del documento. Infine, nell'introduzione è anche presente una sezione dedicata ai riferimenti utili a capire il documento.
2. Architettura del Sistema Corrente: questa sezione contiene la descrizione del sistema corrente. Tuttavia, come illustrato in seguito, ad oggi non esiste un vero e proprio sistema che fa ciò che si propone di fare EasyLease, ma esiste un sistema similare di riferimento.
3. Architettura del Sistema Proposto: in questa parte viene illustrata l'architettura del Sistema Proposto. Questa sezione si apre con una breve panoramica che va ad illustrare il sistema proposto. Successivamente, viene effettuata la decomposizione del sistema generale in sottosistemi ed il mapping. Al punto successivo verrà poi esposto il mapping Hardware/Software. Poi verrà illustrata la gestione dei dati persistenti, il controllo degli accessi e la sicurezza, il controllo del flusso globale del sistema e le condizioni limite.
4. Servizi dei Sottosistemi: contiene la rappresentazione dei servizi dei sottosistemi.

2. Architettura del Sistema Corrente

Attualmente non esiste un sistema che vuole realizzare gli stessi obiettivi di EasyLease: stiamo quindi nel campo della Green-field Engineering, cioè stiamo esplorando funzionalità ancora mai scrutate. Un sistema similare di riferimento è Arval: i punti in comune con il sistema proposto sono la possibilità di ricercare e visualizzare un'auto e le sue caratteristiche e la possibilità di reperire dei contatti telefonici o telematici (e-mail); tuttavia, i due sistemi si diversificano dal fatto che il cliente non può compilare ed inviare una richiesta di preventivo e, per tanto, non esiste una vera e propria gestione del leasing online, ma solo una visualizzazione delle auto.

3. Architettura del Sistema Proposto

3.1 Panoramica

Il sistema proposto rientra nella Green-field Engineering, in quanto stiamo sondando per primi questo campo. EasyLease è una web app che per sua natura ha un ciclo di vita molto lungo (in quanto l'azienda AutoErre vuole mettere a disposizione la piattaforma a tempo indeterminato) e, per tanto, nel tempo il sistema andrà incontro ad un processo di reengineering al fine di aggiungere nuove funzionalità e migliorare quelle già presenti.

Il sistema è rivolto all'azienda AutoErre S.r.l. (quindi a chiunque sia in essa coinvolto, sia esso amministratore o consulente) e a chiunque sia interessato ad effettuare un leasing. La piattaforma mette a disposizione funzionalità diverse, in base a chi ne fa richiesta.

L'utente che si reca sul sito può registrarsi alla piattaforma, diventando così cliente. L'utente si differenzia dal cliente poiché può effettuare solo una parte delle attività che può effettuare il cliente. Infatti, l'utente può ricercare e visualizzare le auto, ma non richiedere un preventivo o, di conseguenza, sottoscrivere un contratto. Il cliente, invece, oltre a ricercare e visualizzare le auto, può anche richiedere un preventivo. Una volta che la sua richiesta di preventivo riceve risposta, egli ha a disposizione un tempo prestabilito per accettare o rifiutare. Se egli decide di accettare, si procede con la sottoscrizione del contratto. Il cliente ha, infine, la possibilità di recarsi nella propria area utente e modificare le proprie informazioni personali.

Il consulente non ha la possibilità di registrarsi, poiché ciò viene fatto dall'amministratore. Egli ha la possibilità di fare l'accesso al sistema, visualizzare tutti gli ordini ad esso associati, visualizzare e rispondere alle richieste di preventivo ancora non associate a nessun consulente. Così come il cliente, il consulente ha la possibilità di andare nella propria area utente e modificare le proprie informazioni personali.

L'amministratore non si registra, poiché le sue credenziali sono già generate alla nascita della piattaforma. Egli ha la possibilità di fare l'accesso al sistema, effettuare l'aggiunta o la rimozione di un consulente. Analogamente, l'amministratore può aggiungere, modificare o rimuovere un'auto.

Lo stile architetturale usato è di tipo repository in quanto i sottosistemi che compongono il software accedono e modificano una singola struttura dati (nel nostro caso un database MySQL). L'architettura implementa un pattern di tipo MVC, diffuso nello sviluppo di interfacce grafiche di sistemi software object-oriented in grado di separare la logica di presentazione dei dati dalla logica di business. Questo tipo di architettura è multi-tier ovvero le funzionalità del sito sono separate e suddivise in più sottosistemi su più livelli in comunicazione tra loro.