Software Design Document

Sistem informatic distribuit penry gestionarea stocurior si vanzarilor pentru o firma care inchiriaza masini

|  |  |  |
| --- | --- | --- |
| Dumitru Cristina-Nicoleta | Grupa CEN 3.1B | cristinadumitru1998@gmail.com |
| Capisizu Cosmin Louis | Grupa CEN 3.1A | capisizucosmin@gmail.com |
| Dumitru Mihai Madalin | Grupa CEN 3.1B | duumstering@gmail.com |
| Dieaconu Danut Nicusor | Grupa CEN 3.1B | dan.eu42@yahoo.com |

|  |  |
| --- | --- |
| Instructor: | Stoica Spahiu Cosmin |
| Curs: | Proiect 2 “Use of DataBases” |
| Sectiunea de laborator: | *Software Engineering* |
| Asistent: | Sbora Mihai-Catalin |

|  |  |
| --- | --- |
| Data: | 26.03.2020 |

**CUPRINS**

1. INTRODUCERE
   1. Scop
   2. Domeniu
   3. Prezentare geneala
   4. Material de referinta
   5. Defenitii si acronime
2. PREZENTAREA GENEALA A SISTEMULUI
3. ARHITECTURA DE SISTEM
   1. Design Arhitectural
   2. Decrierea Descompunerii
   3. Design rational
4. DATA DESIGN
   1. Descrierea datelor
   2. Dictionar de date
5. COMPONENTE DE DESIGN
6. INTERFATA DESIGH UTILIZATOR
   1. Prezentarea geneala a interfetei utilizatorului
   2. Screen Images
   3. Screen Objects and Actions
7. MATRICEA CERINTELOR
8. APPENDICES
9. **INTRODUCERE**

Mijloacele de transport reprezinta o problemă pentru acei oameni care nu dispun de transport personal. Fie ca vor sa plece intr-o excursie, fie ca vor sa inchirieze o limuzina pentru un eveniment important sau doar au nevoie de ceva ca sa se deplaseze, tot mai multi oameni din mediul urban incep sa se confrunte cu aceste probleme.

Astfel, dezvoltarea unei aplicatii care sa ofere servicii atat pentru cei care doresc sa ofere masini spre inchiriat, cat si pentru cei care doresc sa inchirieze, este necesara. In ziua de astazi, cu doar cateva click-uri, putem obtine tot ce ne dorim chiar de acasa; deja suntem familiarizati cu shopping-ul online, e-banking etc. In mod similar, **(nume aplicatie)** este o platforma online unde poti inchiria o masina, avand o gama varianta din care sa alegi. Nu multi oameni isi permit sa isi cumpere o masina; pentru acesti oameni, aceasta platforma ar putea fi raspunsul pentru problemele lor. Acest sistem include o multitudine de tipuri de masini.

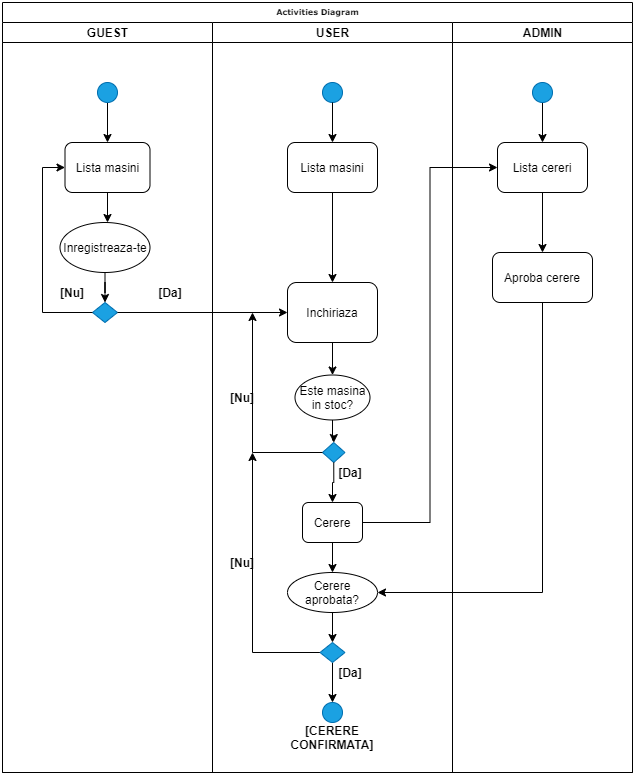
* 1. **SCOP**

Dezvoltarea tehnologiei a imbunatatit in mod considerabil numeroase afaceri, comunicarea intre companii (service provider) si clientii lor, printre care si industria inchirierilor de masini. Acest sistem de inchiriere al masinilor este conceput avaind in vedere urmatoarele:

* Imbunatatirea proceselor de afaceri:
* Rezervarea vehiculelor online - unealta prin care clientii pot rezerva masinile disponibile online conform termenilor de ridicare
* Inregistrarea utilizatorului - un portal de inregistrare unde sunt stocate informatiile clientului, unde sunt monitorizate tranzactiile astfel incat sa le putem oferi servicii imbunatatite
* Rezervari de grup - le permite clientilor sa faca rezervari pentru un grup in cazul nuntilor sau intalnirilor corporatiste
* Declaratie problema - inchirierea unei masini presupune imprumutarea unei masini pentru un termen limitat, in schimbul unei taxe de bani pe o perioada determinata
* Scopul produsulu - acest proiect intreprinde mai multe zone, plecand de la concepte de business pana la programare si este necesar ca pentru acesta sa se realizeze cercetari, tocmai pentru a asigura obiectivele proiectului. Zonele acoperite de proiect sunt:
  + Industria inchirierilor de masini: Aceasta include studii despre cum functioneaza aceasta, procesul implicat si oportunitati care exista pentru imbunatatire.
  + Tehnologia PHP folosita pentru dezvoltarea aplicatiei
  + Clienti generali, precum si angajatii unei companii vor putea sa foloseasca acest sistem in mod eficient
  + Platforma web – acest sistem va fi disponibil 24/7, exceptie facand probleme tehnice ale serverului, care speram a fi minimale.
* Scopuri si obiective. Scopurile specifice sunt:
  + Realizarea unui sistem web care permite clientilor inregistrarea pe site si rezervarea online a unei masini
  + Sa simplifice sarcina unui utilizator intotdeauna cand acesta doreste sa inchirieze o masina.
  1. **DOMENIU**

In AcrivityDiagram sunt explicate principalele procese logice pentru fiecare tip de utilizator in parte.

* **Guest** va putea doar vizualiza lista de masini. Pentru a beneficia de restul serviciilor puse la dispozitie Guest-ul va fi diectionar spre pagina de Login.
* **User** va fi persoasa deja log-ata pe un cont. Va avea acces la toate serviciile dispuse pe site.
* **Admin-** un va avea scopul logic de gestionare al cererilor.



* 1. **PREZENTARE GENEALA**

Acest document este scris in comformitate cu standardele Software Design Documentation explicate in “IEEE Recommended Practice for Software Design Documentation”.In capitolul 2 este prezentarea generala a sistemului. Capitolele 3 – 5 contin descrierea arhitecturii de sistem si a componentelor de design. In capitolul 6 sunt prezentate si explicate sectiuni de UI ale sistemului iar capitolul 7 contine diagramele cu clasele.

* 1. **MATERIAL DE REFERINTA**
  2. **DEFENITII SI ACRONIME**

1. **PREZENTAREA GENEALA A SISTEMULUI**

Proiectul consta intr-o aplicatie WEB ce are rolul de a se ocupa cu gestionarea stocurilor si vanzarilor pentru o firma care inchiriaza masini. Dezvoltarea aplicatiei se face folosind platforma .NET Core si model de proiectare software MVC .

.NET Core este o platformă de dezvoltare cu sursa publica fiind axata pe dezvoltarea aplicatiilor in mod general. Este intretinuta de Microsoft și de comunitatea .NET de pe GitHub. Este disponibila Windows, macOS și Linux.

Modelul MVC este folosit pntru a imparti responsabilitatea sistemului in subsisteme responasbile de anumite actiuni.

Proiectul isi propunesa acopere urmatoarele functionaliati:

Indentificare, folosita pentru a stoca date despre clienti si pentru a personaliza interactiunea cu aplicatia. De aceasta functionalitate se ocupa sistemul de logare si baza de date construita pentru acesta.

Prezentarea produselor, folosita pentru a furniza datele necesare si personalizate clientilor care beneficiaza de serviciile oferite. Aceasta functionalitate este indeplinita de sistemul de distribuire a informatiilor care are scopul de a memora, afisa sau prezenta datele despre serviciile disponibile.

Administrare, aceasta functionalitate permite administratorlor editarea informatiilor si accesul la datele clientilor in cazul in care acestia sunt interesati sa achizitioneze un serviciu.

1. **ARHITECTURA DE SISTEM**
   1. **DESIGN ARHITECTURAL**

Develop a modular program structure and explain the relationships between the modules to achieve the complete functionality of the system. This is a high level overview of how responsibilities of the system were partitioned and then assigned to subsystems. Identify each high level subsystem and the roles or responsibilities assigned to it. Describe how these subsystems collaborate with each other in order to achieve the desired functionality. Don’t go into too much detail about the individual subsystems. The main purpose is to gain a general understanding of how and why the system was decomposed, and how the individual parts work together. Provide a diagram showing the major subsystems and data repositories and their interconnections. Describe the diagram if required.

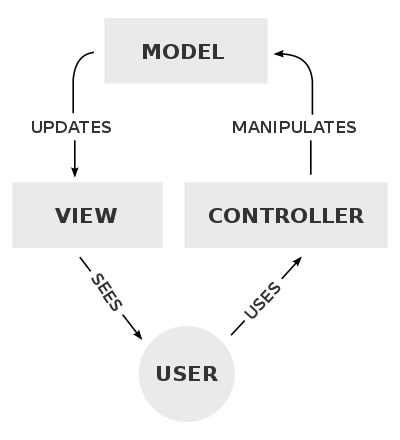
Dezvoltați o structură de program modulară și explicați relațiile dintre module pentru a obține funcționalitatea completă a sistemului. Aceasta este o imagine de ansamblu la nivel înalt a modului în care responsabilitățile sistemului au fost repartizate și apoi atribuite subsistemelor. Identificați fiecare subsistem la nivel înalt și rolurile sau responsabilitățile care îi sunt atribuite. Descrieți modul în care aceste subsisteme colaborează între ele pentru a obține funcționalitatea dorită. Nu intrați în prea multe detalii despre subsistemele individuale. Scopul principal este de a obține o înțelegere generală despre cum și de ce a fost descompus sistemul și modul în care părțile individuale lucrează împreună. Oferiți o diagramă care prezintă principalele subsisteme și depozite de date și interconexiunile acestora. Descrieți diagrama dacă este necesar.

Deoarece modelul MVC este folosit aplicatia este impartita in trei categorii de sisteme Model, View si Controller.

Model-ul stochiaza datele necesare cum ar fi adatele clientilor, datele masinilor si datele personalului care intretine pagina.

View-ul este interfata cu care interactioneaza utilizatorul atnci cand foloseste aplicatia. Componentele de acest fel sunt folosite pentru a rtansmite informatii si pentru a obtine informati si actiuni de la utilizatori.

Controller-ul are rolul de as se ocupa cu actiunile pe care aplicatie trebuie sa le indeplineasca cum ar fi incarcatul datelor, modificarea datelor, salvarea datelor. Tot odata are rolul de a trimite utilizatorilor View-ul necesar si de a prelucra datele obtiunute de catre Model sau View.



Modelele folosite sunt clasele: Car, User, History.

Controlerele folosite sunt clasele: NavigationController, UserController, CarControler, HistoryController, DataController.

Paginile de vizualizare folosite sunt paginile: Home, Login/Register, Cars, My History.

* 1. **DESCRIEREA DESCOMPUNERII**

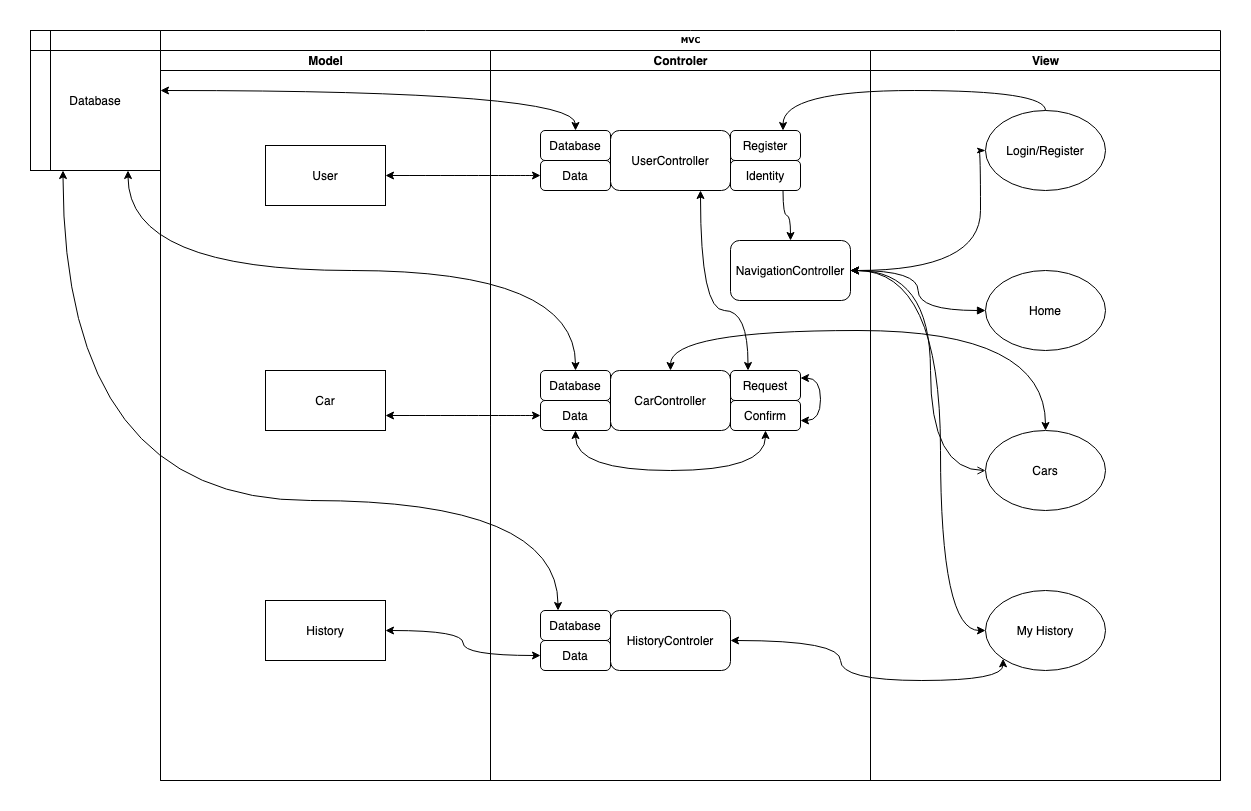
Provide a decomposition of the subsystems in the architectural design. Supplement with text as needed. You may choose to give a functional description or an object¬oriented description. For a functional description, put top¬level data flow diagram (DFD) and structural decomposition diagrams. For an OO description, put subsystem model, object diagrams, generalization hierarchy diagram(s) (if any), aggregation hierarchy diagram(s) (if any), interface specifications, and sequence diagrams here.

Asigurați o descompunere a subsistemelor în proiectarea arhitecturală. Supliment cu text după cum este necesar. Puteți alege să oferiți o descriere funcțională sau o descriere obiectorizată. Pentru o descriere funcțională, puneți diagrama de flux a datelor (DFD) și diapozitive structurale de descompunere. Pentru o descriere OO, puneți modelul subsistemului, diagramele obiectelor, diagrama (jurnalele) ierarhiei de generalizare (dacă există), diagrama (jurnalele) de agregare (dacă există), specificațiile interfeței și diagramele de secvență aici.

Fiecare componenta are la randul ei alte componente sau date folosita de aceasta componenta.

Modelele nu au alte subcomponente deoarece rolul lor este sa stocheze date, ele au proprietati cum ar fi name in cazul clasei User sau registrationNumber in cazul clasei Car.

Controlerele au mai multe subsisteme. Controlerul Users are subsistemul Identity care se ocupa cu identificarea utilizatorului. Sau Register care se ocupa cu salvarea unui nou utilizator in baza de date. Controlerul CarControler se ocupa cu gestiunarea masinilor dar are si componente care se ocupa cu afisarea celor mai relevante si cu confirmarea serviciilor si modificarea disponibilitatii unei masini deja inchiriate.



* 1. **DESIGN RATIONAL**

Arhitectura MVC a fost aleasa deoarece ofera o separare clara a responsabilitatilor si permite imbunatatirea independenta a componentelor fara a afecta alte componente din sistem.

Dezavantajele arhitecturii sunt cresterea complexitatii proiectului si impartirea unei actiuni in mai multe componente astfel un proiect MVC este mai greu ne navigat.

In cazul proiectului nostru aceasta arhitectura isi gaseste locul fiind fexibila si de viitor.

1. **DATA DESIGN**
   1. **DESCRIEREA DATELOR**

Transformarea datelor poate fi împãrtitã în urmãtoarele etape, fiecare aplicabil dupã cum este necesar, pe baza complexitãtii transformãrii necesare : descoperirea datelor, maparea datelor, generarea codului, executarea codului, revizuirea datelor.

Pasii pot fi descrisi astfel:

Descoperirea datelor este primul pas în procesul de transformare a datelor. În mod obisnuit, datele sunt profilate folosind instrumente de profilare pentru a întelege mai bine structura si caracteristicile datelor.

Maparea datelor este procesul de definire a modului în care câmpurile individuale sunt mapate, modificate, unite, filtrate, agregate etc. pentru a produce rezultatul final dorit.

Generarea codului este procesul de generare a codului executabil (de ex. SQL, Python) care va transforma datele pe baza regulilor de mapare a datelor dorite ?i definite.

Executia codului este pasul prin care se executã codul generat pe baza datelor pentru a crea iesirea doritã.

Revizuirea datelor este ultima etapã a procesului, care se concentreazã pe asigurarea datelor de iesire care îndeplinesc cerintele de transformare.

* 1. **DICTIONAR DE DATE**

1. **COMPONENTE DE DESIGN**

Cerintele non-functionale, asa cum prevede si titlul, sunt cerinte care nu sunt direct luate in vedere atunci cand sunt oferite spre folosire. Acestea se pot raporta la proprietatile urgente ale sistemului, timpul de raspuns, etc. Alternativ, ele pot defini constrangeri ale implementarii sistemului, de exemplu: capacitatile dispozitivelor Intrare / Iesire sau reprezentarea datelor folosite in interfate cu alte sisteme. Cerintele non-functionale, precum securitatea, sau valabilitatea, de obicei specifica sau constrang caracteristicile unui sistem ca intreg.

* Utilizarea

Sistemul furnizeaza ajutor si meniu de support in toate interfatele pentru utilizator, ca acesta sa interactioneze cu sistemul. Utilizatorul poate folosi astfel sistemul pentru a contacta agentii companiei din cadrul departamentului de asistenta.

* Securitatea:

Sistemul ofera nume de utilizator si parola, tocmai pentru a preveni ca persoanale neautorizate sa realizeze inchirieri. Parola va trebui sa aibe o dimensiune ce depaseste opt caractere. Subsistemul ar trebui sa confere un nivel ridicat de securitate si integritate a datelor; persoanele care doresc sa creeze un cont sunt obligate sa atribuie contului si CNP-ul lor, acesta fiind confirmat prin solicitarea atasarii unei poze cu actul personal. Astfel, prin limitarea unei persoane la un singur cont, accesul neautorizat este redus.

* Performanta:

Rata de raspuns a sistemului pentru fiecare instructiune realizata de utilizator nu ar trebui sa depaseasca minimul de 10 secunde. Sistemul ar trebui sa confere un nivel ridicat de performanta atunci cand executa datele de intrare ale utilizatorului si ar trebui si sa ofere un raspuns intr-un termen cat mai scurt de timp, timpul mediu fiind 50 de secunde pentru sarcinile complicate si 20-25 pentru cele mai putin complicate.

* Valabilitate:

Sistemul ar trebui intotdeauna sa fie disponibil 24 de ore, 7 zile pe saptamana. De asemenea, in cazul apariției oricărei defecțiuni majore a sistemului, sistemul ar trebui sa redevina valabila in 1-2 zile lucratoare, asa incat sistemul de afaceri sa nu fie grav afectat.

1. **INTERFATA DESIGH UTILIZATOR**
   1. **PREZENTARE GENEALA A INTERFETEI UTILIZATORULUI**

Toti utilizatorii vor vedea aceeasi paginã atunci când intrã pe acest site web. Aceastã paginã solicitã utilizatorilor un nume de utilizator si o parolã sau sã continue ca invitat. Din perspectiva unui invitat, acesta are optiunile de a cãuta masini, are posibilitatea sã afle dacã masina cãutatã este în stoc sau nu si, de asemenea, are optiunea de a-si crea contul si de a vã autentifica pentru a deveni utilizator .

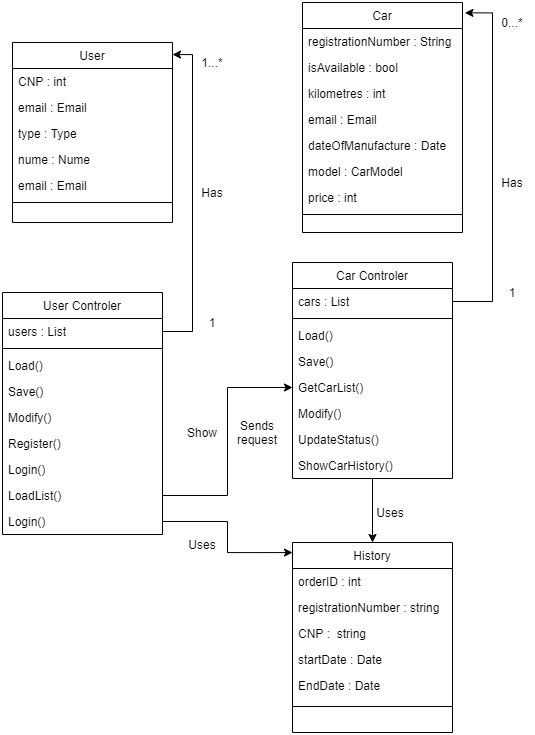
Dupã ce a fost autentificat prin nume de utilizator si parolã corecte, utilizatorul va fi redirectionat cãtre profilul corespunzãtor unde poate face diverse activitãti. Din perspectiva unui utilizator, are optiunile de a cãuta masini, are posibilitatea de a afla dacã masina cãutatã este în stoc sau nu, are optiunea de a închiria o masinã, având posibilitatea sã solicite timp suplimentar, apoi sã trimiteti o cerere si asteptati sã fie confirmata, precum si locul unde sã plãtiti pentru a închiria o masinã, având posibilitatea sã plãtiti pentru a putea folosi masina închiriatã mai mult.

UI va fi simplã si de consecventã, folosind terminologia întelese în mod obisnuit de cãtre utilizatorii intentionati ai sistemului. Sistemul va avea o interfatã simplã, în concordantã cu interfata standard, pentru a elimina nevoia de instruire a utilizatorilor.

* 1. **SCREEN IMAGE**
  2. **SCREEN OBJECTS AND ACTIONS**

1. **MATRICEA CERINTELOR**

Mai jost este diagrama claselor de baza ce vor fi folosite.

****

1. **APENDICES**