


Baigiamoji praktika III-ojo tarpinio atsiskaitymo ataskaita

Organizacijos praktikos vadovas	<u>programuotojas Gintautas Kučinskas</u> <i>(pareigos, vardas, pavardė, parašas)</i>
Praktikos vadovas universitete	<u>docentas Giedrius Ziberkas</u> <i>(pareigos, vardas, pavardė, parašas)</i>
Studentas:	<u>IFF-7/7, Matas Žilinskas</u>  <i>(grupė, vardas, pavardė, parašas)</i>

1. Projektavimo eiga ir sistemos projektas **1.1. Projektavimo valdymas ir eiga**

Prieš sistemos realizacijos pradžią, klientai pateikė xlsx tipo failą, kuriame buvo išvardinti visi funkciniai reikalavimai.

Kai projekto realizacijos eigoje atsiranda poreikis, klientai nubraižo būsenų diagramą, kuri programuotojų komandai atskleidžia visą objekto gyvavimo eigą.

1.2. Projektavimo technologija

Funkciniai reikalavimai buvo išvardinti su Microsoft „Excel“ įrankiu.
Būsenų diagramoms braižyti buvo naudojamas Microsoft „Visio“ įrankis.

1.3. Programavimo kalbos, derinimo, automatizavimo priemonės, operacinės sistemos

Sistemos naudotojo sąsaja yra realizuojama su TypeScript, CSS bei HTML kalbų kombinacija. Taip pat naudojamos Angular karkaso teikiamais privalumais. Naudojamas kodo redaktorius: Visual Studio Code bei jo plati plėtinių aibė.

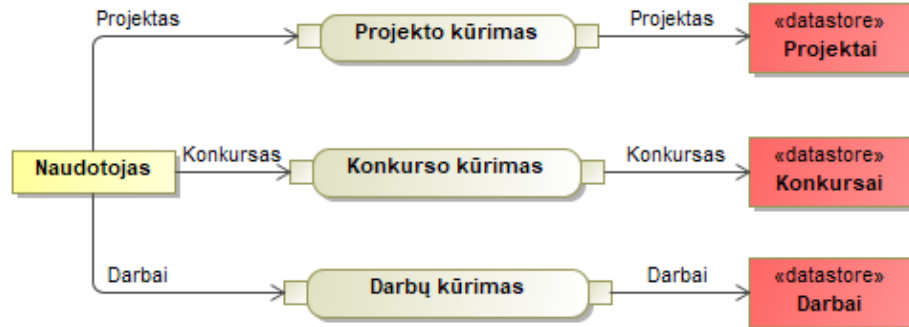
Sistemos serverio funkcionalumas yra programuojamas C# kalba, su .NET karkaso naujausia versija. Naudojama integruota programavimo aplinka: Visual Studio su Resharper plėtiniu.

Naudojama versijavimo kontrolės sistema: Git.

Naudojama operacinė sistema: Windows 10.

1.4. Informacijos srautai

Sistemos informacijos srautų diagrama pateikta 1 pav (diagrama yra supaprastinta siekiant apsaugoti kliento pranašumą rinkoje).



1 pav. Informacijos srautų diagrama

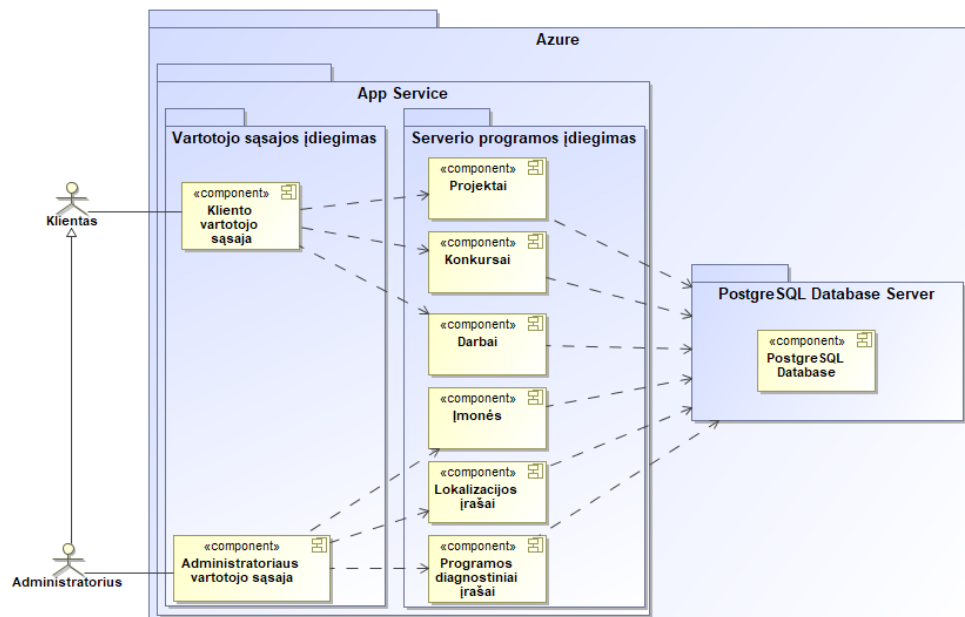
1.5. Sistemos sudėtis

Kuriamą sistemą sudaro trys pagrindinės dalys:

1. Duomenų bazė, kurioje yra saugoma visa statinė informacija.
2. Serveris, kuris turi tiesioginę prieigą prie duomenų bazės ir gali atlikti tarpinius veiksmus, jei tokių reikia.
3. Vartotojo sąsaja, per kurią vartotojas mato visą reikiamą informaciją bei gali daryti užklausas į serverį.

1.6. Komponentai

Sistemos komponentų diagrama pateikta 2 pav (diagrama yra supaprastinta siekiant apsaugoti kliento pranašumą rinkoje).



2 pav. Komponentų diagrama

1.7. Sąsajos įvertinimas pagal vartotojo patyrimą, profesinę terminologiją

Dabartiniame projekto realizavimo etape labiau rūpinamės baziniu funkcionalumu negu vartotojo sąsajos patyrimu. Kaip to pasekmė, vartotojo sąsaja naujiems vartotojams gali būti gana nepatogi. Visų pirma, kai kurios bazinių objektų formos yra apkrautos labai dideliu informacijos kiekiu, kuris gali eiti net per kelias korteles. Visų antra, kai kurie mygtukai neturi pavadinimų – tik ikonas. Kartais galima nesuprasti, ką tam tikra ikona reiškia.

Šiuo metu sistema pritaikyta jau patyrusiems vartotojams, kurie žino, kaip tokiomis sistemomis naudotis. Visgi, ateityje yra numatyta vartotojo patyrimo pusę patobulinti.

1.8. Duomenų kontrolė

Kiekvienas veiksmas yra validuojamas serveryje ir, jeigu įmanoma, vartotojo sąsajoje. Jei tam tikrų duomenų trūksta, arba įvesti duomenys yra nevalidūs, vartotojui parodomas pranešimas, kuriame tiksliai parašyta, kas negerai.

Duomenų kontrolės pavyzdys:

Jeigu vartotojas mėgins sukurti projektą neįrašęs jam jokio pavadinimo, sistema jau vartotojo sąsajos lygmenyje jam išmes klaidą – raudonai paryškins pavadinimo laukelį bei raudonomis raidėmis parašys žinutę „Pavadinimas yra privalomas“.

2. Testavimo eiga ir rezultatai

Šiame projekto realizacijos etape atliekame tik vienetų testavimą. Testuojame visus servisų viešus metodus, stengiamės ištestuos kiek įmanoma daugiau kelių.

Vienetų testavimui naudojama „xUnit“ biblioteka, pritaikyta C# kalba parašytoms programoms.

Vietoj reliacinės duomenų bazės, naudojame „Entity Framework“ bibliotekos suteiktą „In Memory Database“ alternatyvą, kuri imituoja reliacinę duomenų bazę darbinėje atmintyje.

Taip pat aktyviai naudojamos „MoQ“ biblioteka, kuri leidžia imituoti bet kokią sąsają – pavyzdžiui, jeigu servisas reikalauja el. laiškų siuntimo serviso, tačiau mes nenorime siųsti tikro el. laiško kiekvieną kartą, kai vykdome vienetų testus.

Šiuo metu sistemai parašyti maždaug 800 vienetų testų.

3. Dokumentacija

Dokumentacija programuotojui:

1. Įdiegti Visual Studio Code.
2. Įdiegti NodeJS.
3. Įdiegti NPM CLI.
4. Įdiegti .NET 5.0 SDK,
5. Įdiegti PostgreSQL duomenų bazės serverį.
6. Susikonfigūruoti Microsoft User Secrets.
7. Sukompiliuoti programą.
8. Paleisti programą.

Dokumentacija vartotojui:

1. Nueiti į svetainę (adreso neminėsiu dėl konfidencialumo).
2. Užsiregistruoti per rekvizitus.
3. Prisijungti.
4. Prisijungus galima patekti į bet kurį klientui skirtą puslapį.