Univerzitet u Nišu

Elektronski Fakultet



Predmet:

**Stručna praksa**

**Izveštaj o realizaciji stručne prakse**

Student:

Pavle Aleksov, 16466

Niš, 2021.

Sadržaj

[Uvod 3](#_Toc72629766)

[Izveštaj 4](#_Toc72629767)

[Dan 1 – „ukrcavanje“ (onboarding) na projekat 4](#_Toc72629768)

[Dan 2 – dokumentacija i rad ES sistema 4](#_Toc72629769)

[Dan 3 – postavka projekta 5](#_Toc72629770)

[Dan 4 – implementacija 6](#_Toc72629771)

[Dan 5 – logovanje na FTP server 6](#_Toc72629772)

[Dan 6 – logovanje u SQL Server bazu 6](#_Toc72629773)

[Dan 7 – testiranje 7](#_Toc72629774)

[Dan 8 – ispravljanje bagova 7](#_Toc72629775)

[Dan 9 – refaktor koda 8](#_Toc72629776)

[Dan 10 – launch support 8](#_Toc72629777)

[Zaključak 9](#_Toc72629778)

# Uvod

Praksa je održana u firmi Codemancy[[1]](#footnote-1) DOO u Nišu, i trajala je od 06.07.2020. do 17.07.2020. Firma se bavi full-stack web programiranjem (web dizajn, backend programiranje, frontend programiranje, UX). Moja uloga je bila programiranje .NET aplikacija korišćenjem C# programskog jezika, rad sa SQL Server bazama podataka, testiranje aplikacija…

U ovom izveštaju neće biti detalja o implementaciji zbog potpisanog NDA ugovora, već će biti opisani procesi razmišljanja kod rešavanja određenih zadataka, kao i generalne ideje oko projektovanja rešenja. Pored toga biće opisani i problemi koji su se javljali u toku izvršavanja zadataka.

# Izveštaj

## Dan 1 – „ukrcavanje“ (onboarding) na projekat

Prvog dana, pored upoznavanja sa radnim prostorom, dobio sam i prve zadatke. Najpre su me upoznali sa potrebama klijenta (biznis logika), zatim sa sistemom koji će se koristiti za implementaciju biznis logike, i na kraju sa mojom ulogom u svemu tome.

Klijentu je potreban e-commerce sajt na kome će se korisnicima prodavati njegov proizvod, uz mogućnost da korisnici mogu da pozivaju druge potencijalne korisnike da se pridruže sajtu, i uz to ostvare određene beneficije. Ova vrsta biznis logike često se svrstava u MLM (Multilevel marketing).

Za implementaciju biznis logike se koristi sistem koji već pruža većinu funkcionalnosti potrebnih za jedan MLM biznis. Ovaj sistem pruža mogućnost proširenja. Na taj način klijent (odnosno klijentov tim programera) može proširiti funkcionalnosti sistema po potrebi. Moja uloga jeste bila programiranje tih dodatnih funkcionalnosti kroz proširenje sistema. U daljem tekstu, sistem će se oslovljavati sa ES, a njegovo proširenje – „ekstenzija“.

Ostatak dana proveo sam istraživanjem mape sajta kompanije koja je realizovala taj sistem, kao i sa detaljnijim istraživanjem klijentovih potreba.

## Dan 2 – dokumentacija i rad ES sistema

Drugog dana sam se upoznavao sa ES sistemom, kao i njegovim načinom komuniciranja sa ekstenzijom. To je jedino bilo moguće čitanjem njihove developer dokumentacije. S obzirom da je u tom trenutku ES bio relativno nov sistem, njegova dokumentacija nije uvek mogla da isprati njegov funkcionalni razvitak, pa je to često predstavljalo prepreku u mojem razumevanju njegovog rada. U tom trenutku sam suštinski shvatio značaj dokumentovanja softvera. Ipak, programeri koji razvijaju ES su često bili dostupni za bilo kakvu pomoć vezanu za njihov sistem i način razvitka ekstenzije.

Pre svega je potrebno pomenuti da se ekstenzija kompajlira kao DLL fajl i direktno uključuje u projekat ES sistema vezan za određenog klijenta.

Komunikacija između ES i ekstenzije moguća je na dva načina:

1. Hook
2. API

Hook je koncept sličan webhook-ovima, pa sam najpre to istraživao. Webhook-ovi se najčešće koriste kada jedna web aplikacija treba da odgovori na neki događaj koji se odigrao u nekoj drugoj web aplikaciji. Prva aplikacija onda treba da napravi HTTP endpoint koji će se pozivati iz druge aplikacije kada se određeni događaj odigra. Taj endpoint je potrebno na neki način proslediti drugoj aplikaciji kako bi ona znala šta treba da „gađa“. To se najčešće radi nekom vrstom registrovanja endpointa u drugu aplikaciju.

Ono što razlikuje klasične webhook-ove od hook-ova ES sistema jeste, najpre, način povezivanja. S obzirom da se ekstenzija direktno uključuje u ES projekat klijenta, nije potrebno praviti neki remote endpoint. ES je to rešio kroz Dependency Injection projektni obrazac koji se jednostavno uvodi u ASP.NET aplikacije. Potrebno je svaki hook „registrovati“ kao servis koji će se kasnije koristiti od strane ES sistema:

public class ExtensionEntry : IExtension

{

public void ConfigureServices(IServiceCollection services)

{

//NOTE: These are examples of how to implement a custom Api Endpoint and a custom hook.

//services.AddSingleton<IExampleService, ExampleService>();

//services.AddScoped<IApiEndpoint, ApiExample>();

//services.AddTransient<IHook<IsEmailAvailableHookRequest, IsEmailAvailableHookResponse>, IsEmailAvailableHook>();

}

}

Iz primera koda možemo videti da se i API registruje na sličan način. Razlika između hook-a i API-a je što će se hook pozivati automatski od strane ES sistema svaki put kada se događaj vezan za taj hook odigra, dok API moramo pozivati ručno, sa frontend-a (s obzirom da ES nudi i mogućnost promene frontend koda).

## Dan 3 – postavka projekta

Trećeg dana je počelo programiranje ekstenzije. Najpre je bilo potrebno instalirati potrebne NuGet pakete (među kojima je i paket ES sistema koji nudi klase za rad sa hook-ovima i API endpointovima). Jedna custom API klasa u stvari označava jedan endpoint koji se može pozivati.

Postavljen je jedan API endpoint koji će se koristiti na sajtu za prikaz poena koje su korisnici zaradili prilikom svojih kupovina, kao i kupovina svojih downline-a (korisnika koji su se registrovali pod njegovim linkom). Pored toga postavljen je i API endpoint koji će se koristiti od strane kupaca za povratak poena u obliku novca.

Postavljeni su i hook-ovi čija je uloga dodela samih poena pri kupovini. To su uglavnom hook-ovi vezani za porudžbine:

* Hook koji se okida kada se porudžbina kreira – s obzirom da se porudžbine mogu praviti sa različitih stranica („prodavnica“), ovaj hook će se koristiti za identifikaciju stranice sa koje je napravljena porudžbina (tačnije za identifikaciju „prodavnice“). Kasnije će se taj identifikator proslediti i ostalim hook-ovima vezanim za porudžbinu.
* Hook koji se okida kada se porudžbina plati – biće zadužen za dodelu samih poena, u zavisnosti od „prodavnice“ u kojoj je napravljena porudžbina, kao i tipa kupca koji je napravio tu porudžbinu.
* Hook koji se okida kada se porudžbina otkaže – zadužena je za oduzimanje poena koji su prethodno dodeljeni, ukoliko se porudžbina otkaže, kako se ne bi zloupotrebljavao sistem dodele poena

Pored tih, imamo i hook vezan za registraciju. Korisnici će i pri registraciji dobiti određen broj poena u zavisnosti od tipa korisnika pod kojim se registruju.

## Dan 4 – implementacija

Četvrtog dana je konačno krenula sama implementacija biznis logike na ekstenziji. Za realizaciju dodele poena često su korišćeni servisi iz ES NuGet paketa poput onih za pribavljanje informacija o kupcu, za pribavljanje informacija o korisnicima povezanim sa kupcem, za dodelu i oduzimanje samih poena,… Sama logika dodele poena u početku nije bila kompleksna, međutim problemi su se javljali kod korišćenja ES servisa. Bili su slabo dokumentovani pa je bilo potrebno rešavati probleme „grubom silom“, odnosno isprobavati razne opcije dok se ne dođe do rešenja. U nekim slučajevima bilo je potrebno komunicirati sa ES programerima, čiji odgovor dobijamo uglavnom sledećeg dana zbog razlike u vremenskim zonama.

## Dan 5 – logovanje na FTP server

S obzirom da se kod izvršava isključivo kao dodatak u ES klijentov projekat, nije bilo nikakvih mogućnosti debagiranja. Zbog problema koje smo imali generalno sa ES sistemom i servisima, i dužinom trajanja međusobne komunikacije, koja je bila neretka, odlučeno je da dodamo sopstveni logging sistem koji bi nam pomogao u pronalasku grešaka, kao i prepoznavanju uspešno izvršenih hook-ova i API-a.

Naša firma ima svoje FTP servere, pa je najpre odlučeno da se logovi nakon kreiranja okače na neki od FTP servera, kako bi mogli da se pregledaju u bilo kom trenutku. Logovi bi bili obični tekstualni fajlovi koji bi sadržali informacije o ključnim tačkama izvršavanja koda (flow aplikacije). Za pregled samog fajl sistema FTP servera možemo koristiti bilo kojeg klijenta. Ja sam lično odlučio za FileZilla[[2]](#footnote-2)-u.

Za rad sa FTP serverom kroz C# kod, odlučeno je da se koristi FluentFTP[[3]](#footnote-3) NuGet paket. Fajlovi su se pravili po jednom hook/API request-u, i nakon završetka svakog, bilo uspešno ili neuspešno, slali su se na FTP server. Ono što nismo znali u tom trenutku jesu privilegije koje je naša ekstenzija imala na mašini na kojoj se izvršava ES kod klijenta. Povratne poruke grešaka su ukazivale na to da nemamo privilegije za pisanje na disk mašine gde je ES kod. S obzirom da je bio poslednji dan u nedelji, ostalo nam je da to iskomuniciramo sa ES programerima, i pogledamo njihov odgovor sledeće nedelje.

## Dan 6 – logovanje u SQL Server bazu

Odgovor od ES-a nije bio pozitivan. Pretpostavke su bile tačne, nismo imali privilegije za upis na disk, a nismo ni mogli da dobijemo tu privilegiju. S toga je ceo FTP logging sistem morao da otpadne. Umesto toga, ES nam je ponudio upis u njihovu bazu, tačnije kreiranje novih tabela u drugoj šemi. Ovo je, naravno, i pre FTP servera bila bolja opcija, međutim nismo ni za to imali privilegije, ali smo mogli da ih dobijemo.

Sada je logging sistem bilo potrebno preneti na komunikaciju sa SQL Server bazom. Većina logging koda morala je da se obriše s obzirom da većina sadrži samu komunikaciju sa FTP serverom. Same linije poruka za logovanje su mogle da ostanu takve kakve jesu.

Najpre je dodat servis koji će direktno komunicirati sa bazom. Odlučeno je da se za C# komunikaciju sa bazom koristi Dapper[[4]](#footnote-4) ORM. U tom servisu je metoda za upis u log tabelu naše novo-kreirane šeme.

Pored toga dodat je i servis koji će da predstavlja „omotač“ (wrapper) oko ovog servisa koji upisuje u bazu. Životni ciklus tog servisa će biti scoped, što omogućava korišćenje istog loggera u istom HTTP kontekstu. Samim tim, ako se flow promeni na neku drugu klasu, dodavanje novih linija u konačnu log poruku će sadržati i sve poruke iz „pozivajuće“ klase.

## Dan 7 – testiranje

Testiranje se odvija kroz dve faze:

1. Pisanje test slučajeva
2. Izvođenje testiranja

To je često tako odrađeno da bi mogle različite osobe da se podele po fazama. Pre zvaničnog testiranja, često je i poželjno da sam programer koji je pisao kod prođe kroz bar jedan od slučajeva, i najbolje uz debager. To u ovom slučaju nije bilo moguće zbog samog načina rada ekstenzije. Međutim, s obzirom na malu kompleksnost logike, kao i mojeg dobrog poznavanja sadržaja projekta, moja je dužnost bila odraditi obe faze testiranja. Samim tim se može smatrati da je programer prošao kroz kod koji je pisao.

Pisanje test slučajeva obuhvatalo je smišljanje samog slučaja koji treba da se testira - to su uglavnom bili slučajevi dodele poena po svakom tipu kupca i po svakom tipu njihovih veza sa drugim korisnicima; zapisivanje procesa, odnosno potrebnih koraka da bi se izvršio test; i eventualno potrebnih podataka (recimo kreiranje naloga određenih vrsta test kupaca) za izvršavanje testa.

Izvođenje testiranja može biti i jednostavnije od pisanja test slučajeva. S obzirom da već imamo test slučajeve samo je potrebno pratiti korake za njihovo izvršavanje. U ovom slučaju koraci nisu morali da budu vrlo detaljni, s obzirom na moje poznavanje sistema na kome se radi. Kada bi se test slučajevi pisali za drugu osobu, trebalo bi pisati detaljnije kako bi osoba korektno izvršila taj test. Pri izvršavanju testova potrebno je beležiti njihov status (uspešan ili neuspešan, odnosno zadovoljava očekivanja ili ne). Nakon toga sledi i ispravljanje grešaka pronađenih u toku testiranja.

## Dan 8 – ispravljanje bagova

Kod pisan na ekstenziji do tada nije ni jednom bio testiran (zbog nemogućnosti debagovanja). Zbog toga je imao puno sitnih nedostataka, poput grešaka u porukama logging sistema, u određenim brojkama, obrnutim uslovima u if naredbama… Nakon ispravki bi se ponovo testirali određeni slučajevi kako bi se proverila njihova korektnost. Pri svakoj novoj promeni bilo je potrebno ponovo deploy-ovati ekstenziju. Taj proces traje oko 5 minuta, međutim to može delovati vrlo sporo u trenucima kada su promene vrlo sitne. To je predstavljalo još jednu poteškoću u ovakvom sistemu, pa sam često čekajući na deploy neke sitne promene, radio na promeni nekog drugog dela koda.

## Dan 9 – refaktor koda

Nakon svih izmena i popravki, moglo je da se primeti da je kod nepotrebno dug, i na nekim mestima skoro isti. Da bi se lakše tumačio od strane programera koji bi u budućnosti radili na projektu, potrebno je srediti tu situaciju, tačnije potrebno je refaktorisati kod. U ovom slučaju najkritičnije je bilo kopiranje nekih funkcionalnosti na više mesta. Taj problem je rešen odvajanjem tih funkcija u posebne servise. Kod je nakon toga smanjen za oko 300 linija (ne računajući dodavanje tih novih servisa čiji je odnos broja linija i doprinosa samoj arhitekturi neznatan), a sama struktura je bila znatno čitkija.









Nakon i tih promena bilo je potrebno odraditi kratko testiranje, kako bi se utvrdilo da one nisu uticale na samu logiku. Nakon toga kod je bio spreman za produkcionu verziju.

## Dan 10 – launch support

Poslednjeg danja prakse nije bilo mnogo posla. S obzirom da je sve završeno po planu, ostalo je samo ostati prisutan kao podrška pri korišćenju logike na ekstenziji na produkcionoj verziji sajta. Većih problema nije bilo.

# Zaključak

Generalno iskustvo je vrlo zadovoljavajuće. Rad sa ES sistemom je u nekim trenucima podsećao i na low-level programiranje, s obzirom na svoj trial-and-error proces rada. Definitivno nije lako raditi sa slabo dokumentovanim sistemima, ali je sve moguće uz dobru podršku drugih programera.

Nakon ove prakse ostao sam da radim u firmi, gde sam već oko 10 meseci. Neki projekti i danas koriste ES sistem zajedno sa ekstenzijom. Iako je njegovo trenutno stanje daleko od savršenog, može se reći da je ipak poboljšano u odnosu na ono za vreme moje prakse. Uz to su pronađeni i novi načini „preskakanja“ nekih ES prepreka od strane tima, pa je i lakše raditi sa njim.

1. <https://codemancystudio.com/> [↑](#footnote-ref-1)
2. <https://filezilla-project.org/> [↑](#footnote-ref-2)
3. <https://github.com/robinrodricks/FluentFTP> [↑](#footnote-ref-3)
4. <https://dapper-tutorial.net/dapper> [↑](#footnote-ref-4)