UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Boštjan Hikel

Nadgradnja portala za pomoč pri trženju zavarovalniških produktov

DIPLOMSKO DELO
NA VISOKOŠOLSKEM STROKOVNEM ŠTUDIJU

Mentor: doc. dr. Rok Rupnik

Ljubljana, 2011

Univerza
v Ljubljani

Fakulteta za računalništvo
in informatiko

Tržaška 25
1000 Ljubljana, Slovenija
telefon: 01 476 84 11
faks: 01 426 46 47
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00014/2010
Datum: 01.10.2010

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **BOŠTJAN HIKEL**

Naslov: **NADGRADNJA PORTALA ZA POMOČ PRI TRŽENJU ZAVAROVALNIŠKIH PRODUKTOV**

THE UPGRADE OF PORTAL TO SUPPORT MARKETING OF INSURANCE PRODUCTS

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Speljite nadgradnjo portala za pomoč pri trženju zavarovalniških produktov. Portal naj bo namenjen potencialnim strankam. Poudarek naj bo na tem, da stranka dobi na portalu čim več informacij in da se stranko poveže z lokalnim zavarovalniškim agentom.

Mentor: RRWZ

doc. dr. Rok Rupnik

Dekan:

prof. dr. Nikolaj Zimic

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

IZJAVA O AVTORSTVU

diplomskega dela

Spodaj podpisani Boštjan Hikel,

z vpisno številko 63050041,

sem avtor diplomskega dela z naslovom:

Nadgradnja portala za pomoč pri trženju zavarovalniških produktov

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom prof. dr. Roka Rupnika

- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela

- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 21.03.2011 Podpis avtorja:

Zahvala

Zahvaljujem se svojemu mentorju, doc. dr. Roku Rupniku, za vso strokovno pomoč in nasvete pri izdelavi diplomske naloge. Prav tako se zahvaljujem tudi vsem sodelavcem v podjetju CREA d.o.o., ki so na svoj način pripomogli k nastanku naloge.

Posebno zahvalo namenjam svoji družini in punci za podporo in spodbudne besede v času študija ter potrpežljivost predvsem med izpitnimi obdobji. Hvala, da ste mi priskočili na pomoč, ko sem vas potreboval.

KAZALO VSEBINE

Seznam uporabljenih kratic in simbolov

ASP.NET – spletno aplikacijsko ogrodje za izdelavo spletnih strani in spletnih storitev
CLR – Common Language Runtime – jedro v ogrodju Microsoft .NET za izdelavo aplikacij
FCL – Framework Class Libraries – knjižnica omogoča dostop do funkcionalnosti sistema in je temelj .NET aplikacij, njihovih sestavnih delov in kontrol, ki so vgrajene
C# - Microsoftov programski jezik
XML – Extensible Markup Language – zbirka pravil za grajenje strukturiranih dokumentov
HTML – Hyper Text Markup Language – standardni jezik za razvoj spletnih strani, z njim označujemo in določamo lastnosti besedila
SQL – Structured Query Language – standardni strukturirani jezik za pisanje poizvedb nad podatki v podatkovnih bazah
AJAX – Asynchronous JavaScript and XML – skupina med seboj povezanih metod za razvoj interaktivnih spletnih strani
SOAP – Simple Object Access Protocol – protokol za izmenjavo strukturiranih podatkov pri izvajanju spletnih storitev med računalniškimi omrežji
WSDL – Web Services Description Language – jezik, na osnovi XML, za opisovanje spletnih storitev
VS – VisualStudio – Microsoftovo orodje za urejanje razvojne kode
VSS – VisualSourceSafe – Microsoftovo orodje za hranjenje različic razvojne kode
SSO – SingleSignOn – protokol za enkratno prijavo v sistem z več moduli
SSL – Secure Sockets Layer – protokol za kriptiranje komunikacij v računalniškem omrežju
IR – informacijska rešitev
IT – informacijska tehnologija
HP - HewlettPackard
VMWare – Svetovno znano podjetje, ki se ukvarja z virtualizacijo
SUPB – Sistem za upravljanje s podatkovno bazo – množica programov, namenjena izdelavi, vzdrževanju in nadzoru dostopa do podatkov v podatkovni bazi
SMTP – Simple Mail Transfer Protocol – standardni protokol za prenašanje elektronskih sporočil preko IP omrežij
IP – Internet Protocol – glavni komunikacijski protokol znotraj računalniških omrežij
CGP – celostna grafična podoba (spletne strani)
FS – funkcionalna specifikacija
GUI – Graphical User Interface – grafični uporabniški vmesni oz. tisto kar uporabnik vidi na zaslonu
GUID – Globally Unique IDentifier – unikatna referenčna številka v računalništvu pogosto uporabljena kot identifikator različnih objektov

1 POVZETEK

V diplomski nalogi je predstavljena izdelava dela prenovljenega spletnega portala za priznano slovensko zavarovalniško družbo, ki ji pomaga pri trženju zavarovalniških produktov in pridobivanju novih strank.

Uvodni del diplomske naloge se začne s predstavitvijo zavarovalniške družbe, njenih produktov ter problema prodornosti oz. dostopnosti produktov strankam. V nadaljevanju avtor predstavi še kompleksno aplikacijo, ki jo zavarovalniški agenti uporabljajo na terenu, ter razliko med nekoliko okrnjeno različico, ki bo na voljo strankam na svetovnem spletu.

V glavnem delu, ki je razdeljen na več podpoglavij, je opisan celoten postopek razvoja spletnega portala in delo, ki je bilo opravljeno pred samim razvojem, da bi le-ta potekal tekoče in nemoteno. Izdelava funkcionalne specifikacije ter opredelitev poslovnih zahtev je ena izmed nalog, ki morajo biti opravljene pred razvojem. Poglavje razvoja pa opisuje postopke pri načrtovanju in izdelavi podatkovnega modela ter pozneje podatkovne baze, izbire in izgled uporabniškega vmesnika, opisi izbranih in uporabljenih orodij ter opisi posameznih izbranih komponent na uporabniškem vmesniku. Na koncu sledi še poglavje o varnosti ter vzdrževanju in možnostih nadgradnje portala.

Cilj implementacije take rešitve je zagotoviti pridobitev novih strank, ki jih preko portala napeljemo na kontakt z zavarovalniškim agentom.

Ključne besede: spletni portal, priprave, razvoj, uporabljena orodja, nadgradnje

2 ABSTRACT

The degree addresses the development of a web application for a well-known Slovenian insurance company, which will help marketing of insurance products and gaining new customers.

First part of the degree starts with introduction of the insurance company, its products and problems with their accessibility to the customers. Further on, the author introduces the complexity of existing form application, which is used by agents outside offices and the differences between slightly curtailed version which will be available to customers via Internet.

The main part of the degree, which is divided into several sub-sections, deals the whole development process of the web portal and the work that was done before the actual development to make it run smoothly. Writing of functional specification and defining business requirements is one of the tasks that must be performed prior to development. Development section describes the procedures for designing and constructing the data model and later database, selection and presentation of user interface, descriptions of selected tools and descriptions of selected components on the user interface. The main part is followed by the chapter dealing with safety, maintenance and options for upgrading the web portal.

The objective of implementing such solution is to increase the number of new customers who are directed through the portal to contact an insurance agent.

Key words: web portal, preparation, development, used tools, upgrades

3 UVOD

3.1 Predstavitev problema

Sodobni informacijski sistemi ter tehnološko dovršena spletna stran z velikim naborom funkcionalnosti postajata vse bolj splošna zahteva vsake uspešne organizacije. Uvajanje najnovejših tehnologij in odprtost na splet sta nujna, če želimo pospešiti prodajo in prepoznavnost ter tako zagotoviti rast organizacije. Bliskovit porast ponudbe produktov ter prodaje preko interneta v zadnjih letih je prisilil tako vodilne slovenske kot mednarodne organizacije, da svoje produkte ciljnim strankam posredujejo kar preko spleta, s tem pa je na področje marketinga in prodaje vnesel kar velike spremembe. Zato se je ena izmed vodilnih zavarovalniških družb v Sloveniji odločila svoja zavarovanja in pripadajoče svetovalnice predstaviti tudi na svetovnem spletu ter s tem svojo ponudbo še bolj nazorno predstaviti bodočim in obstoječim strankam.

Zavarovalnice imajo veliko aplikacij v pisarnah in po terenu, veliko zavarovanj ter sovpadajočih produktov, ki so zelo kompleksni in zato težko dostopni oz. razumljivi strankam. Ravno zaradi prodornosti in razumljivosti produktov strankam se zavarovalnice odločajo na splet nalagati zelo okrnjene različice aplikacij, ki so usmerjene k uporabnikom, a po-navadi predstavljajo le del ponudbe.

Omenjena zavarovalniška družba ima že dolgo časa vzpostavljeno spletno poslovanje preko spletnega mesta, na katerem strankam že zagotavljajo vrsto različnih informacij o družbi, zavarovanjih, ponudbah ter del spletnih zavarovalnih storitev: sklepanje in obnova zavarovanj, informativni izračuni ter prijava škod. Ker je zavarovalnica želela nadgraditi sklepanje določenih produktov, izboljšati informativne izračune ter izboljšati prijavo škod je za naročnika potrebno nadgraditi del spletnega portala, na katerem bo predstavljeno zavarovanje doma ter zdravstveno zavarovanje za tujino. Del celotnega spletnega portala, ki ga je potrebno nadgraditi, je razdeljen na oba produkta in ponuja informativni izračun zavarovanja, kar je glavna funkcionalnost aplikacije, poleg informativnega izračuna lahko v delu z zdravstvenim zavarovanjem le-to tudi sklenemo. Portal omogoča registracijo in prijavo, s tem pa uporabnik pridobi nekaj funkcionalnosti, omogoča jim tudi shranjevanje informativnih izračunov ter njihovo pregledovanje. Portal je poleg omenjenega zasnovan tudi za nadgradnjo in predstavitev vseh drugih produktov, ki jih zavarovalniška družba trži.

3.2 Predstavitev rešitve

V diplomski nalogi bom prikazal kako je potekal postopek razvoja informacijskega sistema za del spletnega portala, ki v glavnem skrbi za upravljanje z informativnimi izračuni zavarovanj ter postopek testiranja in predaje aplikacije naročniku. Ker je sistem izpostavljen najrazličnejšim zahtevam in omejitvam, tako poslovnim kot tehnološkim, bom te opisal skozi nalogo.

Diplomska naloga je sestavljena iz štirih delov. Prvi opisuje zajem zahtev, pisanje funkcionalne specifikacije ter pripravo razvojnega okolja z ustrezno strojno in programsko opremo. V nadaljevanju naloge sledi opis razvoja dela spletnega portala, v katerem so predstavljene uporabljene tehnologije in opisi posameznih pomembnih funkcionalnosti

aplikacije. V zaključku drugega dela je zajet tudi opis pomembne komponente poslovanja preko interneta, to je varnost. V tretjem delu naloge je zajet postopek testiranja in nameščanja aplikacije v okolje naročnika, v zadnjem delu pa sta predstavljena izgled in samo delovanje portala, napisane so misli in ugotovitve ter možne oblike vzdrževanja in nadgradnje aplikacije, ki so bile predlagane ob zaključku projekta.

4 AKTIVNOSTI PRED ZAČETKOM RAZVOJA

4.1 Zajem in opredelitev zahtev ter pisanje funkcionalne specifikacije

Po določitvi vsebinskih področij, ki jih mora pokrivati naš del spletnega portala, se je del razvojne hiše, imenovan analitiki, lotil zajema zahtev. V tej začetni fazi se podrobno določijo vse funkcionalnosti portala, ker pa gre v konkretnem primeru za izdelavo aplikacije za znanega naročnika, smo seveda morali poleg splošnih zahtev upoštevati tudi želje naročnikovega marketinga ter njihovega oddelka za informacijske tehnologije in organizacijo. Edini uporabljeni način zajema zahtev je bil v tem primeru pogovor oz. dogovarjanja z zaposlenimi v zgoraj omenjenih sektorjih naročnika. V sklopu zajema tehnoloških zahtev smo skupaj z naročnikom ocenili, da bo sistem informativnega izračuna večinoma uporabljala skupina uporabnikov, ki ima le osnovno znanje uporabe računalnika, zato je potrebno zagotoviti čim enostavnejši izgled in uporabniku prijazen način komuniciranja oz. podajanja zahtevanih podatkov. Ker del zbranih uporabnikovih podatkov predstavlja tudi njegove osebne podatke je potrebno zagotoviti sledljivost dostopov do podatkovne baze v skladu z zahtevami Zakona o varovanju osebnih podatkov.

Že iz samega dejstva, da gre za spletni portal informativnega izračuna zavarovanja, ni težko ugotoviti, da bomo za naročnika razvijali spletno aplikacijo (ang. web application), ki nam prinaša nekatere prednosti. Poleg tega, da sta podatkovni in funkcionalni nivo nameščena zgolj na enem strežniku, kar nam omogoča lažje posodabljanje in varovanje podatkov, je za uporabo dovolj le računalnik s povezavo s svetovnim spletom ter standardno konfiguracijo, namenjeno pisarniški uporabi.

Uporabniške zahteve je priporočljivo, in za končen pravočasni uspeh skoraj nujno potrebno, zapisati v funkcionalno specifikacijo [1], ki predstavlja zaključek zajema zahtev in dogovorov z naročnikom in opisuje delovanje aplikacije iz uporabniškega vidika ter pričakovane vhodne in izhodne parametre.
Vsekakor pa funkcionalna specifikacija ne definira notranjega delovanja sistema, saj ne vključuje natančnejšega načina implementacije posameznih funkcionalnosti. Namesto tega se bolj osredotoči na številne zunanje dejavnike, ki bi lahko sodelovali s sistemom. Tipična funkcionalna specifikacija vsebuje podobne stavke kot: »Ko uporabnik pritisne na potrditveni gumb, se pogovorno okno zapre, aplikacija pa uporabniku ponovno ponudi glavno okno, v stanju, v katerem se je nahajalo, preden je bilo odprto pogovorno okno.« Takšno opisovanje predstavlja stik med uporabnikom in sistemom. Ko uporabnik sistemu s pritiskom na gumb poda vhodne parametre, se aplikacija odzove (oz. naj bi se odzvala) z zapiranjem pogovornega okna.

Nemalokrat pa se zgodi, da se funkcionalna specifikacija v obliki enotnega dokumenta sploh ne pripravi, predvsem pri manjših projektih [2]. Razlog za to je v misli, da bi se čas, namenjen

pisanju dokumenta, lahko koristneje porabil pri samem razvoju. Večji vložek v zajem zahtev v začetni fazi razvoja informacijske rešitve pa občutneje zniža tveganje kasnejšega spreminjanja funkcionalne specifikacije. Ustrezna funkcionalna specifikacija, ki po navadi vsebuje tudi priloženi prototipni produkt ali demo, je pozneje tudi osnova za prevzem in zaključek projekta, za njeno pripravo, v kateri bi bile zajete celovite uporabniške zahteve, pa bi morala biti zainteresirana tako naročnik kot izvajalec, saj tako naročniku kot izvajalcu prinaša številne prednosti:

- naročnik mnogokrat šele po pregledu lastnih zahtev v pisni obliki spozna, ali je rešitev, ki si jo je zamislil, resnično ustrezna ali pa je morda v kakšnem segmentu še nepopolna,
- naročnik lahko preveri, ali je izvajalec pravilno razumel njegove zahteve,
- izvajalcu natančno zapisane funkcionalnosti na začetku nemalokrat predstavljajo varovalko pred širitvijo obsega zahtev v poznejših fazah razvoja,
- izvajalec je v procesu priprave funkcionalne specifikacije že v zgodnji fazi projekta primoran razmišljati o ustreznosti načrtovane tehnične in predstavitvene izvedbe,
- ustrezen popis uporabniških zahtev je temelj uspešnega načrtovanja faze razvoja

4.2 Priprava okolij z ustrezno strojno in programsko opremo

Pri izvajalcu se razvoj in postopki za uspešno dokončanje odvijajo v naslednjih okoljih: razvojno, testno, testno produkcijsko ter produkcijsko okolje. Takšen postopek naj bi veljal za vsak razvoj.

Razvojno okolje mora vzpostaviti izvajalec sam na svoji infrastrukturi, vključno z pripravo distribucije aplikacijskega sistema, ki mora biti skladno (ustrezna arhitektura) s ciljnim tehnološkim okoljem, na katerem bo potekalo testiranje oz. vpeljava aplikacije v produkcijo.

Testno okolje za končno enotno ali za sprotna testiranja med razvojem mora biti ravno tako vzpostavljeno pri izvajalcu.

Testno produkcijsko okolje, ki je po navadi locirano pri naročniku, je ekvivalentno produkcijskemu okolju in je namenjeno testiranju vmesnikov ter funkcionalnosti s strani naročnika.

Produkcijsko okolje je okolje, kjer se izvaja redna produkcija po določilih, skladnih z urnikom - načeloma 24 ur na dan, 7 dni v tednu, je pri naročniku, na tem okolju pa je fizično nameščena aplikacija s svojo podatkovno bazo.

Vsa okolja se med seboj razlikujejo, če ne drugače, po različnih nastavitvenih datotekah, različnih naslovih za klicanje zunanjih spletnih storitev, različnih povezavah na podatkovno bazo.

Strojna oprema, na kateri teče spodaj omenjena programska oprema, je postavljena tako, da ustreza zahtevam po visoki razpoložljivosti in zanesljivosti osnovnih informacijskih storitev, sestavljajo pa jo naslednje komponente:

- Strežnik SuperMicro SuperServer 6025W-URB
- Strežnik HP Integrity rx2800 i2 Server
- Strežniške rezine RAID polje tipa 10
- Strežniške rezine HP ProLiant BL490c G7 L5630

4.2.1 Strežniki [3]

Strežnik je naprava, ki skupaj s programsko opremo, ki jo poganja, nudi določene storitve ostalim napravam oz. odjemalcem v omrežju. Fizično je strežnik zgrajen tako, da je čim bolj odporen na incidente, ki se lahko pripetijo v organizaciji. Diski, na katerih teče operacijski sistem, so zaradi zaščite povezani v RAID polje. Zaradi zaščite pred pregrevanjem pa so strežniki nameščeni v posebnih, ohlajenih sobah. Dodatno zaščito nudijo tudi ventilatorji posameznih komponent. Strežniki imajo po navadi dva napajalnika, ki sta priključena na različna vira napetosti, poleg tega pa imajo strežniki tudi več mrežnih kartic, priključenih na različna omrežna stikala. Vse to nam omogoča visoko razpoložljivost in zanesljivost storitev v organizaciji.

Portal je v razvojnem in testnem okolju tekel na strežniku SuperMicro, ki podpira Intelov Quad-Core procesor, ki ima do 64 GB pomnilnika, posamezna velikost pomnilniške reže pa je lahko 8GB. Strežnik SuperServer omogoča priklop šestih SATA diskov, ki jih lahko povežemo v RAID polje tipa 0, 1, 5 ali 10, ob okvari enega izmed diskov pa ga lahko s pomočjo sistema Hot-Swap zamenjamo brez izklopa strežnika. Poleg naštetega vsebuje tudi zelo zmogljivo grafično kartico in naprednejši sistem hlajenja, priložena programska oprema pa nam omogoča boljši pregled delovanja posameznih komponent.

4.2.2 Strežniške rezine [4]

Strežniške rezine (ang. Server Blades) so kompaktni strežniki in so nameščeni v posebno ohišje imenovano strežniška omara (ang. Server Rack). Vsaka rezina ima svojo procesno enoto in pomnilnik, druge komponente (omrežna kartica, diskovna enota) pa si deli z ostalimi rezinami v omari, ki ima vgrajen sistem za nadzor delovanja posamezne komponente, kot tudi posamezne strežniške rezine. Morebitne okvare ali nepričakovane spremembe temperature pa ustrezno beleži in preko posebnega sistema sporoča sistemskemu administratorju. Za usmerjanje omrežnega prometa med sistemi, ki tečejo v omari, skrbi kar omara sama, ki je zgrajena tako, da ni odvisna le od enega električnega ali omrežnega vira. Električni viri in omrežne povezave so redundantne, kar pomeni, da v primeru izpada enega vira sistem neovirano teče dalje, z uporabo rezervnega vira. Promet po omrežju pa je primerno razporejen, da ne prihaja do preobremenitev le na eni povezavi.

V strežniško omaro je namesto strežniških rezin mogoče namestiti tudi diskovno polje (RAID), v večini primerov pa gre tu kar za zunanja diskovna polja. Polje se razdeli na več posameznih logičnih diskov, ti pa se nato povežejo na strežniško rezino, na kateri teče operacijski sistem.

Kaj razumemo pod pojmom RAID? (ang. Redundant Array Of Independant Disks)
[5, 6, 7]
Je tehnologija, ki zagotavlja večjo zanesljivost za shranjevanje podatkov s pomočjo redundance in združuje več relativno poceni, manj zanesljivih diskovnih komponent, v logično enoto, kjer so vsi diski v polju neodvisni med seboj. Pojem RAID se v računalniških sistemih uporablja kot krovni izraz za poimenovanje podatkovnih shem, ki lahko replicira in razdeli podatke med več diskov. Podatkovne sheme so poimenovane z besedo RAID, kateri sledi številka (npr. RAID 0). Različne izvedbe RAID sistemov vključujejo dva bistvena cilja: povečati zanesljivost podatkov in vhodno/izhodne zmogljivosti. Ko je več fizičnih diskov nastavljenih za uporabo RAID tehnologije, pravimo, da so diski v RAID polju. To polje podatke distribuira na več diskov, le-to pa je operacijskemu sistemu predstavljeno kot en sam disk. RAID je mogoče nastaviti tako, da služi več različnim namenom.
Nekaj pogostih načinov povezovanja več fizičnih diskov v RAID polje:

- **RAID 0:** Za povezavo diskov v RAID 0 polje potrebujemo najmanj dva fizična diska. Zagotavlja nam izboljšano zmogljivost in dodaten prostor, vendar ni odporen na napake. Ravno to pa je njegova slabost, saj ob okvari enega diska izgubimo podatke na vseh diskih. Polje RAID 0 deluje tako, da podatke, ki jih prejme preko vhodno/izhodnih operacij, razdeli na enako velike bloke, ki jih enakomerno porazdeli med vse diske v polju. Tak način porazdelitve diskov odlikuje visoka hitrost zapisovanja in branja podatkov, saj vsak disk prevzame del dela, ki ga mora opraviti polje. V tako polje ni nujno povezati enako velikih diskov, vendar pa je prostor, ki ga pridobimo v polje, velik le toliko, kot je velik najmanjši disk v polju. Zanesljivost polja RAID 0 je enaka zanesljivosti posameznega diska, ki jo delimo s številom diskov v polju.

- **RAID 1:** Tako kot pri predhodni različici tudi tukaj potrebujemo najmanj dva diska, v večini primerov pa gre za polje dveh diskov, kjer so podatki identično zapisani na oba diska (zrcalna množica). Polje zagotavlja toleranco napak, zato z njim lahko operiramo, dokler v zrcalni množici deluje vsaj en disk. Z ustrezno podporo operacijskega sistema lahko izboljšamo zmogljivost branja podatkov, pri tem pa le minimalno zmanjšamo zmogljivost pisanja, na splošno pa je hitrost branja in pisanja podatkov določena z hitrostjo posameznega fizičnega diska v polju. Uporabi polja RAID 1 z ločenim krmilnikom za vsak disk pravimo duplexing.

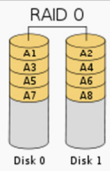
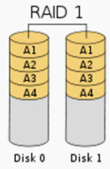
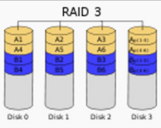
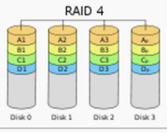
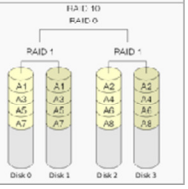
- **RAID 3:** Za delovanje take nastavitve našega diskovnega polja potrebujemo najmanj 3 fizične diske, omogoča pa nam zelo visoke hitrosti prenosa podatkov, ki so izmenično porazdeljeni na dva diska. Na tretjem pa je prostor rezerviran za paritetne podatke, ki se ob okvari enega izmed prvih dveh diskov uporabijo za obnovo podatkov na le-tem. Podatki se obnovijo s pomočjo XOR operacije med pariteto in delujočimi diski. Med okvaro podatki niso na voljo za vhodno/izhodne operacije.

- **RAID 4:** Tudi za to nastavitev potrebujemo najmanj tri diske, delovanje polja pa je podobno nastavitvi RAID 5, le da so vsi paritetni podatki shranjeni na enem disku, kar pa lahko ustvari performančno ozko grlo. Diskovno polje RAID 5 posamezne datoteke porazdeli na več diskov, vsak disk pa za sebe deluje neodvisno od drugih, kar nam omogoča vzporedno izvajanje vhodno/izhodnih zahtev, čeprav lahko prenos podatkov trpi na račun tipa paritete. Algoritem za ugotavljanje napak je shranjen v ločeni diskovni enoti, uporablja pa namenske paritetne podatke.

- **RAID 5:** Paritetni podatki so porazdeljeni skupaj s podatki, pridobljenimi preko vhodno/izhodnih operacij. Polje za svoje delovanje potrebuje vse razen enega diska, ob okvari pa so podatki zaradi paritete na posameznih diskih dosegljivi, celotno polje pa na ta način ni uničeno. Posamezni izpad diska bo imel za svojo posledico zmanjšano delovanje celotnega polja, dokler se okvarjen disk ne zamenja ali obnovi. Polje bo izgubilo podatke ob okvari drugega diska in je ranljivo, dokler se podatki na okvarjenem disku ne obnovijo in zapišejo na nadomestni disk.

- **RAID 6:** Diskovno polje za svoje delovanje potrebuje najmanj štiri fizične diske, kar nam zagotavlja toleranco napak oz. nemoteno delovanje ob okvari dveh diskov v polju. Taka nastavitev večje RAID skupine naredi bolj praktične, še posebej za sisteme visoke razpoložljivosti. To postane zelo pomembno, saj se vedno večjim diskom povečuje tudi čas, ki je potreben za obnovo po tem, ko se zgodi okvara ali napaka na disku. RAID različice z enim paritetnim diskom so občutljive na izgubo podatkov, tako kot RAID 0 različica, dokler se okvarjen disk ne zamenja, njegova vsebina pa obnovi. Obnovitveni čas je odvisen od velikosti diska, večji je disk, večje je trajanje obnove podatkov. RAID različice z dvema paritetnima diskoma nam zagotavljajo čas potreben za obnovo, brez ogrožanja podatkov, če se med obnovo okvari dodaten disk.

- **RAID 10:** Je kombinacija oz. gnezdena različica polja RAID 1 in RAID 0, za delovanje potrebujemo najmanj dva diska, vendar pa se običajno uporabijo kar štirje, ker s tem pridobimo na hitrosti. V tako različico polja lahko povežemo le sodo število diskov, zgrajeno pa je iz zrcalnih množic znotraj črtaste množice, kar nam omogoča toleranco napak in izboljšano zmogljivost, vendar povečuje kompleksnost. Glavna prednost takega polja je črtasta množica, ki vsebuje več zrcalnih množic, na ta način pa je polje ob okvari še vedno delujoče. Toleranca napak je tako visoka, da tako polje preživi tudi več okvar hkrati, oz. toliko časa dokler ima zrcalna množica vsaj en delujoč disk. Ker polje vsebuje zrcalne množice je taka različica varna in skrbi za zaščito podatkov, poleg tega pa omogoča velike zmogljivosti pri branju in pisanju podatkov. Tak način povezovanja diskov v RAID polje uporabljata tudi izvajalec in naročnik.

Level	Description	Minimum # of disks	Space Efficiency	Fault Tolerance	Read Benefit	Write Benefit	Image
RAID 0	Block-level striping without parity or mirroring.	2	1	0 (none)	nX	nX	
RAID 1	Mirroring without parity or striping.	2	1/n	n-1 disks	nX	1X	
RAID 3	Byte-level striping with dedicated parity.	3	1 - 1/n	1 disk			
RAID 4	Block-level striping with dedicated parity.	3	1 - 1/n	1 disk			
RAID 5	Block-level striping with distributed parity.	3	1 - 1/n	1 disk	(n-1)X	variable	
RAID 6	Block-level striping with double distributed parity.	4	1 - 2/n	2 disks			
RAID 10	Mirrored sets in a striped set	2	(N / 2) * Smin	n-1 disks	nX	nX	

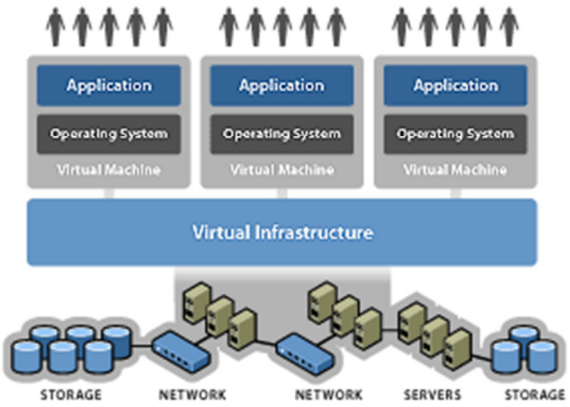
Slika 4.1 Najbolj pogosti načini povezovanja fizičnih diskov v RAID polja in njihove lastnosti.

Taka arhitektura je primerna za poganjanje virtualnih okolij, s katerim prihranimo na prostoru, času in predvsem denarju, pridobimo pa na zmogljivosti. Tako kot na naročnikovem, tudi na izvajalčevem diskovnem polju teče virtualno okolje VMWare ESX 3.5. V virtualno okolje je postavljenih več virtualnih strežnikov, eden izmed njih pa poganja naš portal. Na nekem drugem virtualnem strežniku se nahaja podatkovna baza portala ter druge podatkovne baze in servisi, ki pa niso del te diplomske naloge.

4.2.3 Virtualizacija [8]

Visoka rast poslovnih zahtev in števila sprememb narekuje informacijskim organizacijam čim večjo odzivnost in prilagodljivost. Hkrati od njih zahteva stalen razvoj novih funkcionalnosti, ki ga zavirajo predvsem visoki stroški in čas, porabljen za vzdrževanja obstoječih sistemov in aplikacij. V IT oddelkih so se do sedaj s temi izzivi spopadali z organizacijskimi in tehnološkimi pristopi, ki pa niso dovolj učinkoviti. V zadnjih letih pa je v oddelke IT prodrla ideja virtualizacije, ki bistveno poenostavi obvladovanje sistemov v različnih informacijskih okoljih.

Zniževanje operativnih stroškov ter izboljšanje operativne učinkovitosti in prilagodljivosti. Presežek strežniške konsolidacije in namestitev standardne virtualizacijske platforme, ki nam avtomatizira celotno IT infrastrukturo. Izkoristek moči virtualizacije za boljše upravljanje IT zmogljivost, zagotovitev boljše storitvene ravni in poenostavitev IT procesov. VMWare je skoval izraz za virtualizacijo IT infrastrukture - VIRTUALNA INFRASTRUKTURA.



Slika 4.2 Osnovna ideja postavitve in delovanja virtualne infrastrukture.

Ločevanje okolja programske opreme od okolja strojne opreme lahko združi več strežnikov, podatkovnih skladišč in omrežij v bazenski vir za skupno uporabo. Iz tega izhaja dinamično ter varno in zanesljivo zagotavljanje virov aplikacijam, ki jih potrebujejo. Ta pionirski pristop omogoča uporabo gradnikov cenovno ugodnih strežnikov za izgradnjo samooptimizacijskega podatkovnega centra in zagotavlja visoko stopnjo izkoriščenosti, razpoložljivosti, avtomatizacije in prilagodljivosti.
Virtualna infrastruktura omogoča delitev več fizičnih virov v celotni infrastrukturi. Navidezni stroj omogoča delitev deleža virov iz enega fizičnega računalnika v več virtualnih strojev, s tem pa pridobimo na učinkovitosti. Ti viri se delijo na več virtualnih strojev in aplikacij. Poslovne potrebe so gonilna sila za dinamičnim spajanjem fizičnih virov z aplikacijami, čeprav se te potrebe spreminjajo in razvijajo. Optimizacija virov z združevanjem strežnika z omrežjem in podatkovnimi skladišči v enoten bazen IT virov, katerega aplikacije se potem lahko uporabljajo po potrebi, se kaže v večji fleksibilnosti organizacije ter zmanjšanju celotnih operativnih stroškov.

Virtualna infrastruktura je sestavljena iz:

- Hipervizorjev, ki omogočajo popolno virtualizacijo vsakega fizičnega računalnika.
- Virtualno infrastrukturnih storitev, kot je upravljanje z viri in konsolidirano ustvarjanje varnostnih kopij za optimizacijo razpoložljivih sredstev med virtualnimi stroji.
- Avtomatizirane rešitve, ki omogočajo posebno zmogljivost za optimizacijo predvsem IT procesov (npr.: provizioniranje ter odprava napak).

S pomočjo virtualizacije lahko izboljšamo učinkovitost in razpoložljivost IT sredstev in naših aplikacij. Seveda pa se moramo najprej znebiti starega načela »en strežnik, ena aplikacija«, v katerem so sistemski viri zelo slabo izkoriščeni, ter začeti uporabljati več virtualnih strojev na enem samem fizičnem računalniku oz. strežniku. Hkrati pa s tem rešimo naše administratorje, ki bi porabili toliko časa samo za upravljanje in odpravljanje težav, ki bi nam jih predstavljali strežniki, namesto da bi ta čas bolj koristno uporabili za morebitne inovacije. Kot primer lahko navedem, da se v organizaciji, ki ne uporablja virtualizacije, za vzdrževanje obstoječe infrastrukture porabi okoli 70 % predvidenega proračuna, namenjenega IT oddelku, in tako za inovacije ostane zelo malo sredstev in posledično tudi časa. Uporaba virtualnega okolja zmanjša potrebo po prostoru in električni energiji, poleg tega pa omogoča centralno upravljanje in boljšo konsolidacijo informacijske infrastrukture.

Virtualizacija močno poveča izkoriščenost fizične infrastrukture, hkrati pa brez potrebe po dodatnih stroških omogoča ločevanje aplikacij na različne virtualne stroje, ki tečejo na enem samem fizičnem stroju. Vsak virtualni stroj pa si deli vire prav tega fizičnega računalnika preko več različnih okolij. Različni virtualni stroji lahko poganjajo različne operacijske sisteme in veliko aplikacij na tem fizičnem računalniku.

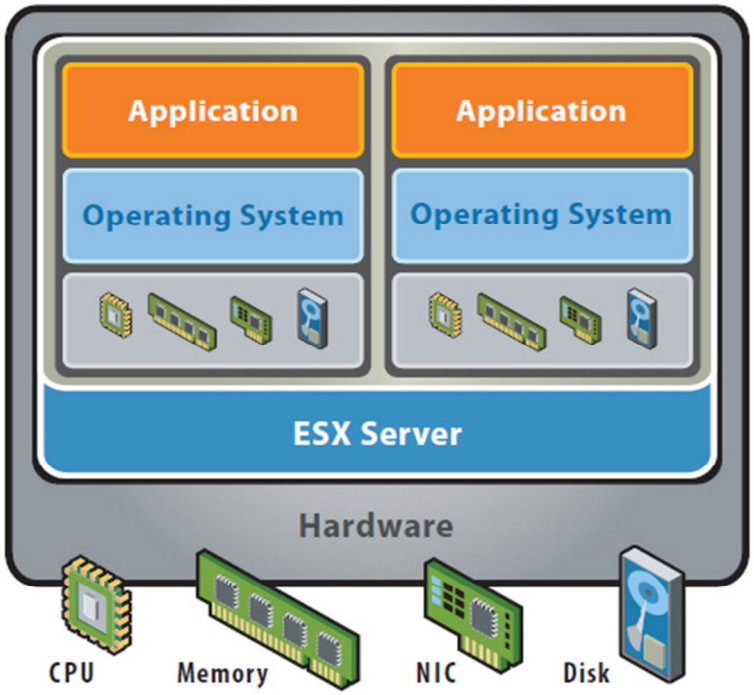
VMWare virtualizacijska platforma je zgrajena na poslovno pripravljeni arhitekturi. Uporaba programske opreme za preoblikovanje ali virtualizacijo strojne opreme računalnika, vključno z procesorjem, pomnilnikom, trdim diskom in mrežno kartico, nam omogoča, da si ustvarimo popolnoma funkcionalen virtualni stroj, na katerem lahko teče svoj operacijski sistem, ki poganja aplikacije tako kot čisto pravi računalnik. Vsak posamezni virtualni stroj vsebuje popoln sistem, kar nam omogoča odpravo morebitnih konfliktov. VMWare virtualizacija deluje tako, da vstavi tanek sloj programske opreme neposredno na računalnikovo strojno opremo ali na gostiteljski operacijski sistem. Ta sloj pa vsebuje virtualni monitor ali »hipervizor«, ki dodeljuje strojne vire dinamično in pregledno med navidezne stroje. Več operacijskih sistemov deluje hkrati na enem fizičnem računalniku, ki si delijo vire strojne opreme med seboj. Z enkapsulacijo celotnega stroja, vključno s procesorjem, pomnilnikom, operacijskim sistemom in omrežnimi napravami, je ta virtualni stroj popolnoma združljiv z vsemi standardnimi operacijskimi sistemi, aplikacijami ter gonilniki naprav. Tako lahko povsem varno poganjamo več operacijskih sistemov in aplikacij istočasno na enem fizičnem računalniku, vsak pa ima omogočen dostop do virov, ki jih potrebuje, takrat ko jih potrebuje.

Virtualizacija fizične infrastrukture nam zmanjšuje stroške, medtem pa nam zagotavlja povečanje učinkovitosti, uporabe in prilagodljivosti že obstoječih sredstev. Po vsem svetu organizacije vseh velikosti pridobivajo na koristi z virtualizacijo, predvsem iz naslednjih razlogov:

1. **Ker pridobimo več iz obstoječih virov:** zajezitev skupnih infrastrukturnih virov in odcepitev od načela ena aplikacija na enem strežniku, s konsolidacijo strežnika.
2. **Ker zmanjšamo stroške z zmanjševanjem fizične infrastrukture ter izboljšanjem strežnikov:** manj strežnikov in s tem povezane strojne opreme pomeni zmanjšanje poslovnega prostora in porabe energije. Boljša orodja za upravljanje omogočajo izboljšanje administracije na strežniku, zato so znižane tudi potrebe po osebju.
3. **Ker povečamo razpoložljivost strojne opreme in aplikacij za boljšo poslovno kontinuiteto:** izdelovanje varnostnih kopij ter selitve celotnega virtualnega okolja brez prekinitve storitev. Odpravljanje načrtovanih izpadov in takojšnja vzpostavitev stanja po nepričakovani napaki.
4. **Ker pridobimo operacijsko prilagodljivost:** odzivanje na spremembe na trgu z dinamičnim upravljanjem virov, hitrejše provizioniranje strežnika in izboljšana namestitev aplikacij.
5. **Ker izboljšamo upravljanje in varnost:** nameščanje, upravljanje in spremljanje varnih okolij, do katerih lahko uporabnik dostopa lokalno ali preko oddaljenega dostopa, z ali brez povezave v omrežje, iz skoraj vseh standardnih namiznih, prenosnih ali tabličnih računalnikov.

Virtualizacija nam skozi dejstva in številke omogoča:

- ✓ od 50 do 70 odstotkov večjo izkoriščenost strojne opreme,
- ✓ do 40 odstotno znižanje stroškov za strojno in programsko opremo
- ✓ od 50 do 70 odstotkov nižje operativne stroške



Slika 4.3 Osnovna ideja virtualizacije ter prikaz virtualnega ESX okolja.

Naročnik in izvajalec imata postavljena dva VMWare ESX 3.5 strežnika, ki ju poganja ena strežniška rezina. Vsakemu izmed njiju je dodeljen po en virtualni disk iz dveh različnih diskovnih polj, ki je zgrajeno v RAID 10 načinu. V virtualnem ESX 3.5 okolju teče več virtualnih strežnikov, ki so namenjeni različnim funkcionalnostim, katere zavarovalnica uporablja za svoje trženje in delovanje, na enem izmed teh pa teče naš portal za podporo trženju produktov. Ostali strežniki, ki tečejo v virtualnem okolju, so namenjeni arhiviranju in podatkovnim bazam. Tako ločevanje podatkovne baze od sistemskega diska in operacijskega sistema nam zagotavlja optimalno branje in pisanje podatkov ter najvišje možne performančne rezultate.

5 Izdelava aplikacije

5.1 Predstavitev uporabljenih tehnologij in orodij

Ker izvajalec v okviru razvoja projektov teži k spletnim aplikacijam, ki jih naročniki lahko uporabijo za svoje zaposlene preko lokalnega intraneta ali se odločijo aplikacijo predstaviti svetu kar po internetu, smo tudi predstavljeni projekt razvili v okolju Visual Studio.NET, ki s svojim objektno orientiranim programiranjem omogoča lažjo ter hitrejšo izgradnjo aplikacije, saj ima odlično podporo najnovejših spletnih tehnologij. Visual Studio podpira delo z različnimi, ne samo za splet orientiranimi programskimi jeziki, kot so Visual Basic, C++, Java in C#. Glede na orientiranost podjetja k uporabi spletnih tehnologij in namembnosti aplikacije, ki jo je potrebno razviti, se ni bilo težko odločiti za zadnji omenjeni programski jezik ter njegove sovpadajoče spletne tehnologije. Ker v veliki večini k aplikaciji sodi tudi ustrezna podatkovna baza, smo se odločili le to postaviti na strežnik SQL Server 2005, saj je ta najbolj združljiv z orodjem Visual Studio.NET. Poleg omenjenih smo za popolnost delovanja uporabili še nekaj drugih tehnologij, ki so združljive in najbolj primerne za uporabo oz. kombiniranje z zgoraj omenjenima tehnologijama. V sklopu predstavitve orodij je potrebno in pomembno omeniti tudi drugo smer orientiranosti izvajalca, in sicer uporabo izključno licenčnih ali odprto-kodnih programov za razvoj rešitev. Glede na to, da je podjetje začelo delovati na osnovi Microsoftovih programov, se ta trend nadaljuje še danes, kar pa vodi do uporabe nekaterih programov, ki bi jih bilo smiselno zamenjati za druge, mogoče bolj popularne, in take, ki imajo enostavnejšo uporabo.

5.1.1 Microsoft Visual SourceSafe (VSS)

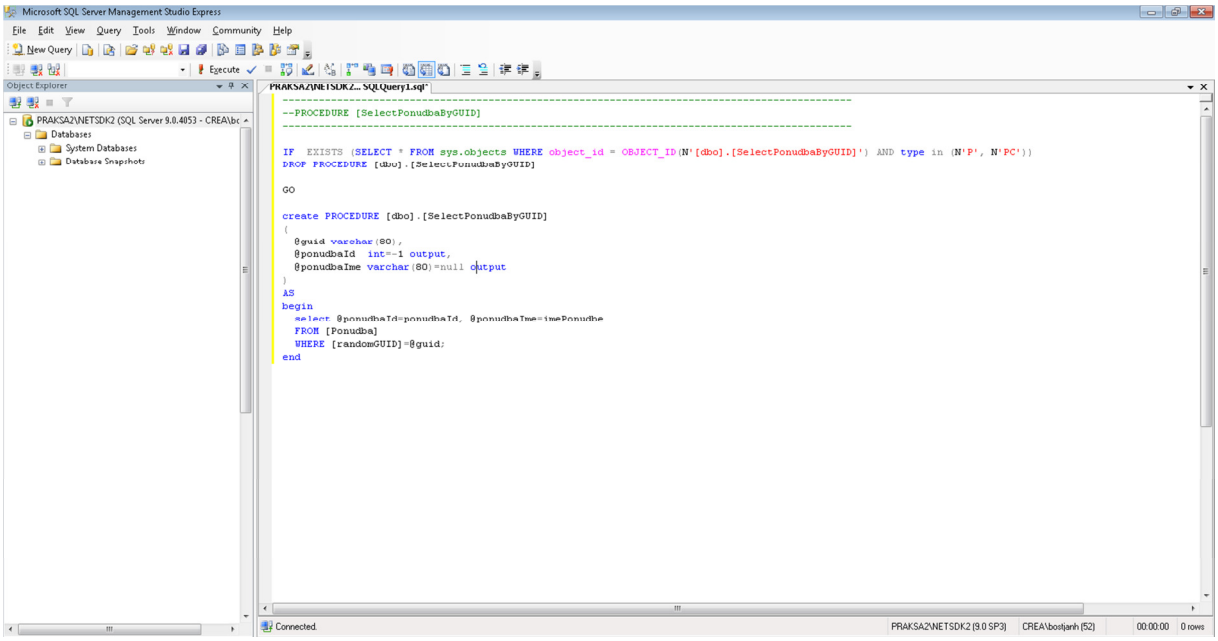
Microsoft Visual SourceSafe je sistem za nadzor različic na ravni datotek, ki mnogim organizacijam zagotavlja obnovitveno točko ter omogoča delo na več različicah istega projekta v istem časovnem obdobju. Ta zmožnost delovanja je še posebej učinkovita v okolju za razvoj programske opreme, kjer se uporablja za ohranjanje vzporednih različic programske kode. Vsekakor pa se program lahko uporablja tudi za ohranjanje datotek v kateri koli drugi panogi. Zaradi projektno usmerjenega nadzora različic in bogate integracije z orodjem Visual Studio, Visual SourceSafe tako individualnim razvijalcem kot razvojnim ekipam omogoča uporabo orodij za varnejše spreminjanje obstoječe programske kode in spremljanje sprememb le-te skozi čas na različnih projektih in različnih uporabnikov. VSS omogoča hitro in učinkovito izmenjavo datotek med projekti, organiziranost le-teh pa koordinacijo razvojne ekipe naredi zelo intuitivno. Ob dodajanju nove datoteke v sistem je ta shranjena v podatkovno bazo in takoj na voljo vsem drugim uporabnikom, vse narejene spremembe se beležijo, tako da lahko vsak uporabnik kadar koli obnovi staro različico datoteke. Člani razvojne ekipe lahko vidijo zadnjo različico datoteke, delajo spremembe lokalnih kopij in v podatkovno bazo shranjujejo najnovejše različice. Sistem za nadzor različic uvaja sprostitveno-rezervacijski (ang. check-in, check-out) model, v katerem si posameznik prilasti datoteko, naredi spremembe, nato pa to shrani nazaj v podatkovno bazo. Ob tem so po-navadi ostali razvijalci brez možnosti shranjevanja iste datoteke, saj si to pravico nekdo že lasti. Sistem prav tako omogoča prekinitev sprememb in razveljavitev zadnje različice, ki bi pozneje lahko povzročala kakršne koli težave. Program podpira tudi združevanje različic datotek, prav tako pa vsebuje mehanizme za razreševanje konfliktov združevanja. Operacije združevanja različic programske kode omogoča neodvisno delo posameznika, ne da bi morali sinhronizirati spremembe, ki so jih naredili drugi člani razvojne ekipe.

5.1.2 SQL Server 2005

Glede na naročnikovo zahtevo po stabilnosti podatkovne strukture, smo se pri razvoju odločili za uporabo Microsoft SQL Serverja, ki nudi integrirano platformo za analizo in upravljanje s podatki, kar organizacijam omogoča zanesljivo upravljanje s kritičnimi poslovnimi podatki in uporabo kompleksnih poslovnih aplikacij. Zagotavlja nam osnovno podporo XML strukture ter internetne poizvedbe. Strežnik SQL Server 2005 povečuje zahteve po zanesljivosti in ponuja inovativne možnosti, ki zvišujejo učinkovitost zaposlenih, vključuje heterogene ekosisteme informacijske tehnologije ter povečuje finančne in operativne proračune. Grafična orodja in čarovniki poenostavijo nastavitve, načrtovanje podatkovnih zbirk in nadziranje učinkovitosti delovanja, kar skrbniku podatkovne baze omogoča, da se osredotoči le na izpolnjevanje strateških poslovnih potreb. Gre za relacijsko podatkovno bazo, njen glavni jezik za poizvedovanje pa je T(ransact)-SQL. Glavna funkcionalnost strežnika je zajem podatkov iz baze, pri čemer strežnik s pomočjo načrta poizvedbe poizkuša poiskati najhitrejši način za pridobitev želenih podatkov.

5.1.3 Sql Management Studio

S pomočjo Microsoftovega brezplačnega grafičnega orodja SQL Management Studio je iz obstoječe podatkovne baze mogoče izdelati podatkovni diagram, iz tega pa se lahko celo generira nova podatkovna baza. Ena od mnogokrat uporabljenih možnosti je tudi pisanje direktnih poizvedb nad podatki, ki se nahajajo v bazi. To se dogaja v enem izmed poizvedben oken, ki jih aplikacija ponuja, kar prikazuje spodnja slika.



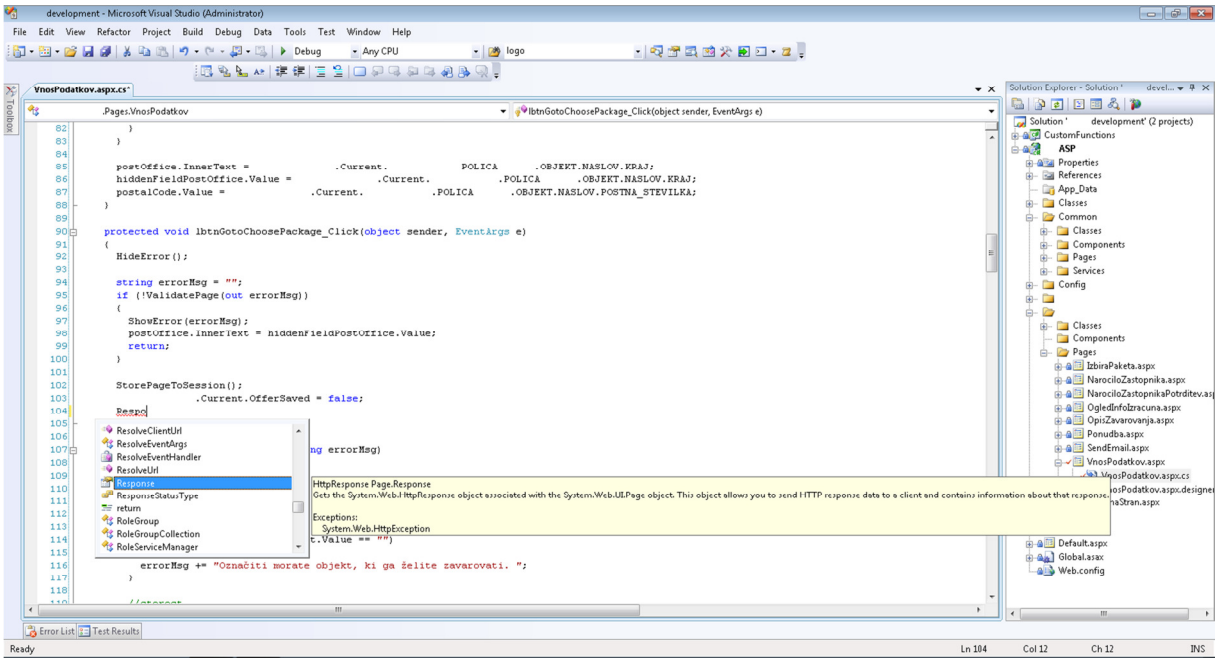
Slika 5.1 Sql Management Studio z odprtim poizvedbenim oknom.

5.1.4 Microsoft Office Visio Professional 2003

Orodje je eno izmed glavnih aplikacij, ki jih svetovno znana podjetja uporabljajo za izdelavo poslovnih in tehničnih diagramov. Gre za izjemno mogočen program, ki je zmožen opravljati nalogo projektiranja tlorisa hiše, ali pa kaj manjšega, na primer, načrt manjšega omrežja. Orodje nam ponuja več kot 60 v-naprej pripravljenih predlog ter številne priročne povleci in spusti (ang. drag and drop) ikone in predmete. Visio smo uporabili za načrtovanje podatkovnega modela, ki smo pozneje pretvorili v bazne skripte. Poleg te predloge vsebuje Visio tudi predlogo za obrazce, predlogo za tloris hiše, predlogo za konceptualni načrt spletne strani, predlogo organizacijske strukture, ...

5.1.5 Microsoft Visual Studio 2008

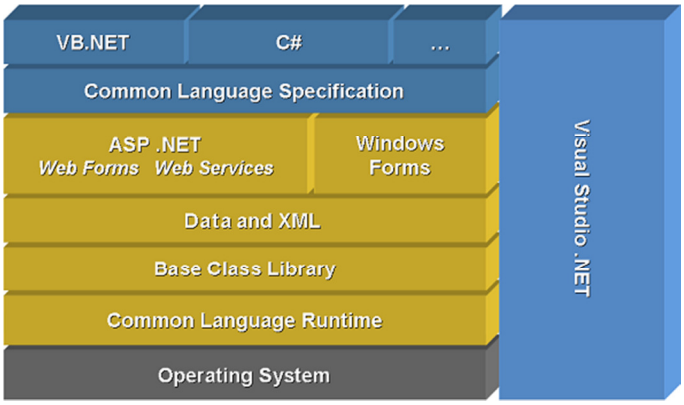
Je integrirano razvojno okolje (IDE) oz. orodje za razvoj programske kode. S takim orodjem je mogoče razviti tako konzolne kot grafične aplikacije, ne zaostaja pa niti razvoj »Windows Forms« aplikacij, spletnih strani (ang. web sites) in spletnih storitev (ang. web services). Visual Studio vsebuje okno z urejevalnikom programske kode, kateremu je v veliko pomoč orodje IntelliSense (orodje za samodejno dokončevanje napisanega dela kode), vgrajen razhroščevalnik (ang. debuger) pa deluje tako na ravni programske kode, kot tudi na strojni ravni, večkrat uporabljeno orodje pa je tudi grafični urejevalnik vnosnih mask (ang. form designer), ki nam je v veliko pomoč pri razvoju GUI aplikacij. Velika prednost je tudi sposobnost povezovanja s sistemi nadzora različic, na primer VSS. Microsoft Visual Studio podpira različne programske jezike, s pomočjo jezikovnih storitev pa sta urejevalnik programske kode in razhroščevalnik sposobna predelati skoraj vse jezike, če le obstaja taka jezikovna storitev. Vgrajeni jeziki so C/C++, VisualBasic.NET ter C#. Podporo za druge jezike (npr. Python ali Ruby) je mogoče ločeno namestiti preko zunanjih jezikovnih storitev. Seveda pa Visual Studio podpira tudi različne tehnologije spletnega razvoja, kot so XML/XSLT, HTML/XHTML, JavaScript ter CSS. Za nadzor vseh podprtih tehnologij Visual Studio uporablja zbirko knjižnic .NET Framework, ki omogoča hitrejše in enostavnejše razvijanje programske kode.



Slika 5.2 Urejevalnik kode Microsoft Visual Studio 2008 z odprtim IntelliSensom.

5.1.6 .NET Framework [9]

Microsoft .NET Framework je programski okvir, namenjen operacijskim sistemom Microsoft Windows. Vsebuje veliko knjižnico (.NET Framework Base Class Library), ki podpira več programskih jezikov, kar razvijalcu omogoča interoperabilnost programskih jezikov (vsak programski jezik lahko uporabi programsko kodo, ki je napisana v nekem drugem programskem jeziku). Ta knjižnica je na voljo vsem programskim jezikom, ki jih podpira .NET, ponuja pa uporabniški vmesnik, povezljivost s podatkovnimi bazami in dostop do njihovih podatkov, kriptografijo, razvoj spletnih aplikacij, numerične algoritme ter povezovanje z omrežjem. Razvijalci uporabljajo knjižnico kot razred v kombinaciji le-te z lastno programsko kodo za izdelavo aplikacij. Druga komponenta .NET Frameworka pa je The Common Language Runtime (CLR). Vsi programi, napisani v okviru .NET, se izvajajo v posebnem programu oz. aplikaciji CLR, ki deluje kot nekakšen navidezni stroj, s tem pa razvijalcu ni potrebno upoštevati zmožnosti procesorja, ki bo izvedel naš program. CLR prinaša tudi druge pomembne storitve, kot so varnost, upravljanje s pomnilnikom in obvladovanje izjem.



Slika 5.3 Zgradba .NET Frameworka.

5.1.7 C#

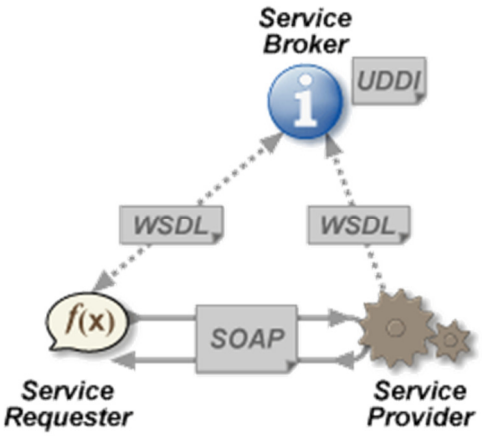
Programski jezik C# je relativno nov, saj je svet ugledal šele leta 2000, potem ko podjetje Sun (Java) ni dovolilo sprememb v njihovem programskem jeziku, v katerih bi zajeli vse to, kar sedaj poznamo pod imenom C#. Od vsega začetka se Microsoft trudi izboljšati funkcionalnost in predvsem varnost jezika, tako da imamo od aprila 2010 na tržišču že četrto različico tega jezika. C# je zelo preprost, moderen, objektno orientiran, predvsem pa splošno namenski programski jezik, saj je v tem jeziku napisanih kar precej računalniških iger, prevajalnikov, orodij in celo operacijski sistem. Seveda pa ne smemo pozabiti niti na spletne aplikacije, ki tečejo na asp.net platformah. Poleg naštetega je programski jezik C# v zadnjem času tudi zelo priljubljen, namreč njegova uporaba se je precej povečala, verjetno predvsem zaradi enostavnosti uporabe in lepo strukturirane programske kode, ki je napisana v tem jeziku, saj morajo biti vse metode in spremenljivke zaprte v razrede, kar nam omogoča preglednost in lahko berljivost. Prednost pa mu dajejo tudi striktni boolean podatkovni tip bool, stavek finally ob lovljenju izjem, samodejni zbiralec odvečnih spremenljivk (ang. garbage collector), ob katerih se sprošča pomnilnik, ki je na voljo našemu programu, ter preprosto spreminjanje podatkovnih tipov v objekte (ang. typecasting).

5.1.8 jQuery

Najnovejša spletna tehnologija na tržišču, ki pa je v bistvu hitra, kratka in jedrnata knjižnica še ene zelo priljubljene spletne tehnologije – JavaScript. Ravno zato je jQuery v zadnjem času tako zelo priljubljen, njegova uporaba pa ima ekstremno visok porast, gledano v odstotkih. Na tržišču se je jQuery pojavil šele avgusta leta 2006, danes pa je s pomočjo te spletne tehnologije napisanih skoraj polovica od 10.000 najbolj obiskanih spletnih strani na svetu. JQuery poenostavlja spreminjanje HTML dokumentov, obvladovanje akcij, animacij ter AJAX interakcij za hitrejši razvoj spletnih strani in aplikacij. JQuery je zasnovan zato, da bi spremenil način pisanja JavaScripta.

5.1.9 Spletne storitve (ang. Web Services) [10]

Spletno storitev lahko razumemo kot način komunikacije med dvema elektronskima napravama. V računalniškem svetu pa to pomeni vmesnik za programiranje aplikacij, ki so dosegljive prek različnih protokolov in izvršene na oddaljenem računalniku oz. strežniku. W3C te spletne vmesnike definira kot programski sistem, narejen za podporo interoperabilnih interakcij med dvema računalnikoma prek omrežja. Razmeroma nova tehnologija je olajšala kompleksno komunikacijo z oddaljenimi sistemi, v zadnjem času pa je to zelo priljubljen način prenosa informacij, predvsem tam, kjer je nujno potrebna interoperabilnost. Spletni portal eAS tako pri komunikaciji s sistemom INIS za shranjevanje zavarovalnih polic uporablja spletne storitve, ki uporabljajo sporočila XML, narejena po standardnem protokolu SOAP (ang. Simple Object Acess Protocol). Spletni strežnik za shranjevanje polic nam nudi opis svojih spletnih storitev preko WSDL (ang. Web Service Description Language) dokumenta, ta pa nam služi za samodejno generiranje programske kode na strani odjemalca.



Slika 5.4 Struktura spletnih storitev.

5.1.9.1 ASWS

Je spletna storitev izvajalca, ki skrbi za komunikacijo med aplikacijo in zalednim sistemom. Poleg tega ASWS vsebuje tudi nekaj drugih funkcionalnosti, kot so predlogi za samodejno dokončevanje imen krajev ter poštnih številk in pridobivanje podatkov o imenih ulic. Glavna funkcionalnost te spletne storitve je izmenjava podatkov o prijavljenem uporabniku ter pridobivanje podatkov za uporabnikove zahteve v aplikaciji, ki pa ni del te diplomske naloge.

5.1.9.2 MegaPOS

Je spletna storitev zunanjega izvajalca za avtorizacijo številke kreditne kartice, ki pozneje tudi izvede plačilo premije zavarovalne police, ki jo je uporabnik sklenil na našem delu spletnega portala. Dodatna prednost je tudi ta, da se celotno plačilo izvaja na strani zunanjega izvajalca, tako da nam ni potrebno skrbeti za shranjevanje kreditnih kartic, varnost podatkov in tako dalje, še vseeno pa imamo nadzor nad samim plačilom, ko interno komuniciramo z MegaPOS sistemom in tako točno vemo, kaj se s samim plačilom dogaja in kakšen je trenutni status plačila. Kratek opis delovanja spletne storitve MegaPOS bi bil, da uporabnika po uspešno zaključeni sklenitvi police in izbranem načinu plačila na našem spletnem portalu preusmerimo na MegaPOS varni (SSL) URL, od koder ga po opravljenem plačilu ta preusmeri nazaj na naš spletni portal, kjer nato v ozadju preverimo, ali je bilo plačilo uspešno ter ga dokončno avtoriziramo in uporabniku sporočimo status njegovega plačila.

5.2 Načrtovanje in izdelava podatkovne baze

Tako kot pri vsakem projektu, se razvoj začne z načrtom in fizično izdelavo podatkovne baze. Seveda se mora načrtovalec, ki je v podjetju zadolžen za posvetovanje z razvijalcem aplikacije ter skrbnikom podatkovnih baz, da uskladijo svoja mnenja in poglede na zastavljeni problem. Ker gre v večini primerov za razvijalca, ki bo naredil kar obe fazi, se ta korak ponavadi izpusti, preden pa se loti načrtovanja, si mora razvijalec v glavi, ali še bolje na listu papirja, zamisliti, kakšen bo približen končni rezultat. V fazi zbiranja in analize zahtev podatkovne baze, si razvijalec ustvari sliko oz. segment realnega sveta ('mini svet‘), ki ga bo predstavil z modelom, nato pa te ideje spravi na papir. Po končani začetni fazi mora razvijalec dojeti in razumeti naslednje stvari:

- katere podatke je potrebno hraniti v podatkovni bazi
- kako so podatki med seboj povezani
- katere operacije se izvajajo nad podatki
- kako bodo uporabniki uporabljali podatkovno bazo

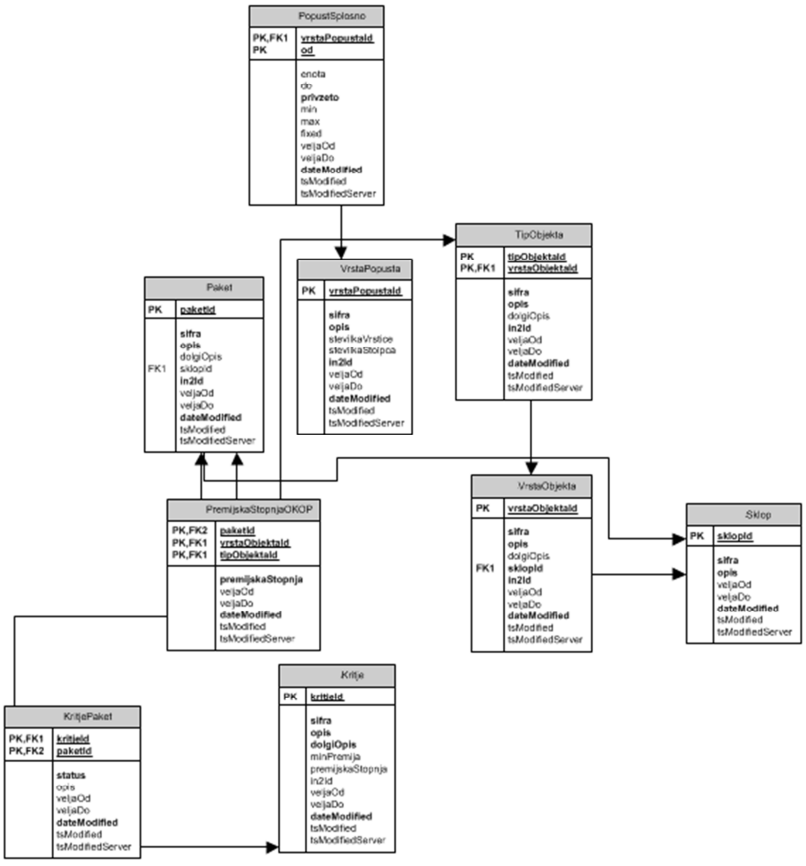
Sledi faza konceptualnega načrtovanja, katerega cilj je izdelati konceptualni načrt oz. konceptualno shemo podatkovne baze. Ta načrt zajema tudi omejitve, ki se nanašajo na podatke. Običajno za predstavitev konceptualnega načrta uporabljamo:

- entitetno – relacijske diagrame
- diagrame razredov
- diagrame objekt – vloga

Pred logičnim načrtovanjem se razvijalec in skrbnik podatkovnih baz dogovorita, kateri sistem za upravljanje podatkovne baze bi bil glede na karakteristike projekta in zahteve naročnika najbolj primeren za dotični projekt. Po izmenjavi mnenj in dogovoru sledi transformacija konceptualne sheme v podatkovni model, ki ga podpira izbrani SUPB. Opravi se tudi natančen pregled dobljene logične sheme, pri katerem se iščejo morebitne napake ali pomanjkljivosti modela. Pred prehodom na naslednjo fazo je potrebno preveriti tudi, ali je podatkovna baza normalizirana (preverijo se funkcionalne odvisnosti med atributi, če mogoče obstaja nepotrebno podvajanje podatkov, ...).
Cilj fizičnega načrtovanja podatkovne baze je izboljšanje delovanje končnega sistema. Določijo se indeksi in parametri shranjevanja podatkovne baze. Identificirajo se različne skupine uporabnikov, nato pa se preveri ustreznost delovanja podatkovne baze glede na potrebe in zahteve le-teh. Ob zaključku te faze dobimo fizično podatkovno bazo, ki jo ponavadi namestimo na poseben, namenski strežnik. Ta faza se imenuje implementacija podatkovne baze, po koncu katere se lahko začne razvoj aplikacije.

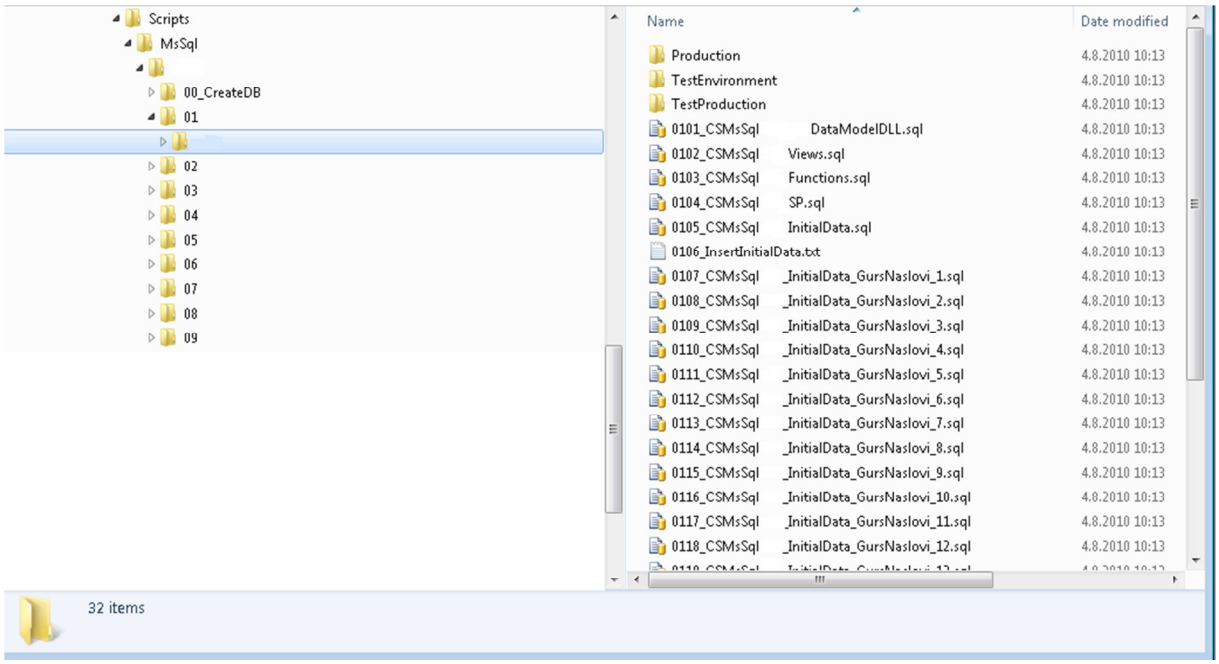
5.2.1 Podatkovna baza spletnega portala

Na podlagi naročnikovih zahtev, ki so zapisane v funkcionalni specifikaciji, smo najprej na list papirja narisali okvirno število tabel in njihovih atributov, po posvetu med sodelavci pa smo se lotili načrtovanja konceptualnega modela, v katerem smo uporabili entitetno-relacijske diagrame. Konceptualni model je bil izdelan v orodju Microsoft Office Visio Professional 2003, s pomočjo katerega smo konceptualni model transformirali v fizični model, nato pa smo iz tega dobili še bazno skripto.



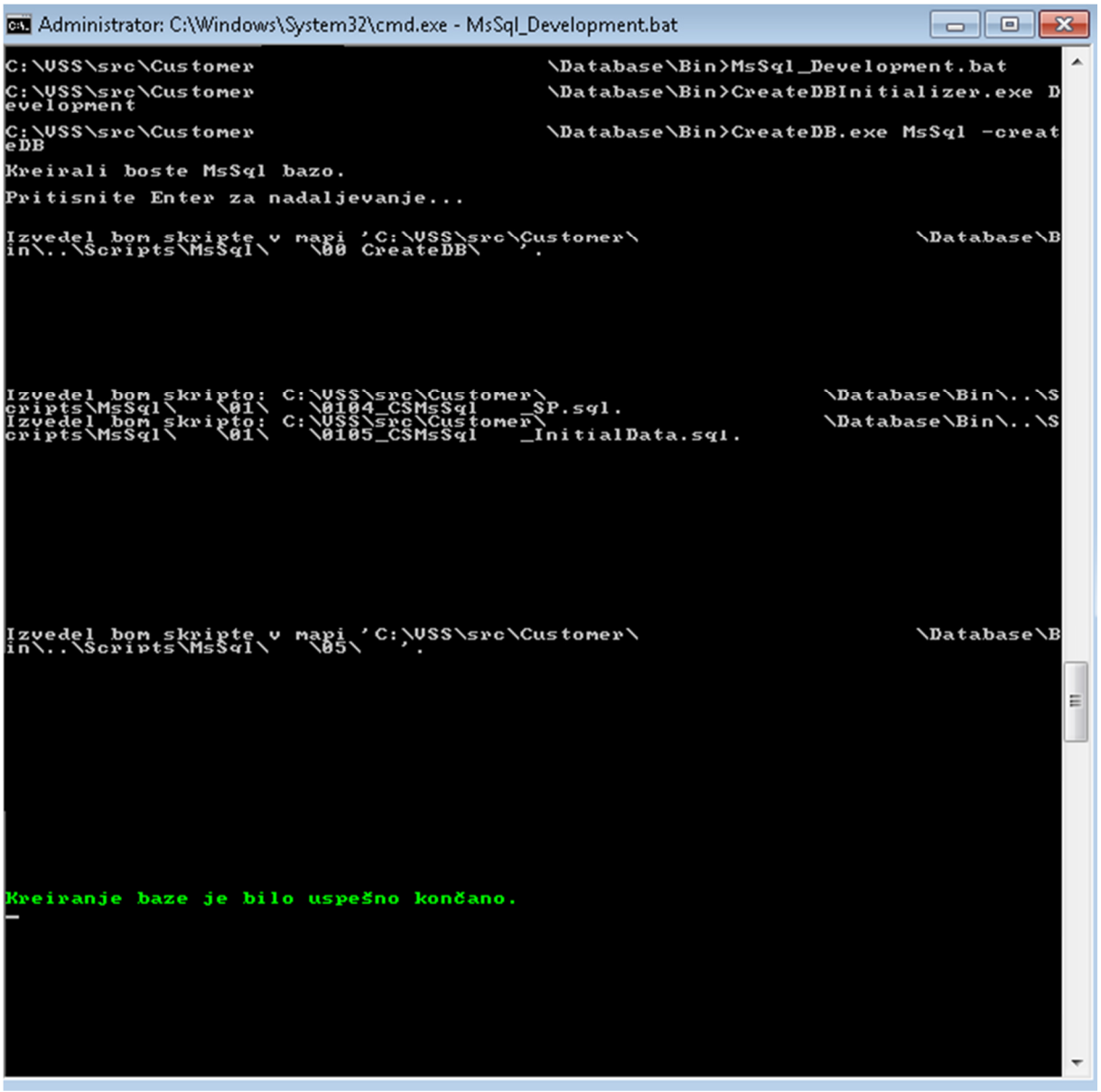
Slika 5.5 Koncept podatkovnega modela dela spletnega portala.

Bazno skripto je bilo pred nadaljevanjem potrebno še prirediti in razdeliti v štiri skupine, v vsaki skupini pa se je bazna skripta razdelila še na tri dele. S tem smo jo naredili kompatibilno z izvajalčevim programom, ki se imenuje »CreateDB«. Program poskrbi za fizično izdelavo baze, saj požene v-naprej pripravljene bazne skripte in na posebnem strežniku ustvari podatkovno bazo. V-naprej pripravljena struktura baznih skript mora biti razdeljena v štiri skupine, kar pomeni, da imamo za vsako izmed okolij izvajanja aplikacije in baze svojo različico. Tako se štiri skupine imenujejo kar po okoljih – razvojno, testno, testno produkcijo in produkcijsko okolje, zato se tudi klic programa izvede vedno drugače. Kot že rečeno, se v vsaki izmed štirih skupin nahajajo še tri skripte. Prva nam ustvari bazo, druga nam ustvari tabele, tretja pa poskrbi za vstavljanje inicialnih podatkov, ki se med seboj razlikujejo v vseh okoljih. V razvojnem okolju so seveda drugačni podatki, kot v produkcijskem. Pozneje, ob razvijanju aplikacije, se po-navadi zgodi, da vsaka izmed omenjenih skupin vsebuje tudi skripto s shranjenimi procedurami (ang. stored procedures), ki vsebuje poizvedbe na podatkovno bazo. Poleg te skripte lahko okolje oz. skupina vsebuje tudi skripti za ustvarjanje pogledov (ang. views) ter ustvarjanje indeksov (ang. indexes). Še pozneje, ko je bila aplikacija skupaj s podatkovno bazo že predana in nameščena k naročniku, lahko pride tudi do sprememb v obstoječih skriptah, po-navadi so to procedure in pogledi, ki jih zapišemo v dodatno strukturo, imenuje pa se različica 2 (ang. version 02), podatkovne baze dela spletnega portala.



Slika 5.6 Struktura baznih skript, pripravljenih za program »CreateDB«.

Ko imamo urejeno strukturo baznih skript, lahko preko Windows ukazne vrstice (ang. command prompt) poženemo interni program za ustvarjanje fizične podatkovne baze, »CreateDB«. Znotraj ukazne vrstice odpremo imenik, kjer se nahajajo bazne skripte, in poženemo datoteko, ki nam pripravi klic internega programa, odvisen od izvajalnega okolja. Program se sprehodi preko imenika in izvrši skripte, ki se nahajajo v njem. Tako se najprej generira podatkovna baza, nato se v bazo vstavijo tabele, v tabele se potem vstavijo inicialni podatki, nato pa se ustvarijo še procedure in pogledi. Kot že omenjeno, se lahko pozneje nabere kar nekaj različic podatkovne baze, zato se izvede vsaka različica posebej, v primeru podvojenih imen pogledov ali procedur pa mora programer zagotoviti, da se ohranja samo tiste iz zadnje različice. Na koncu razvijalec oz. skrbnik podatkovne baze dobi prijazno sporočilo o uspešno ustvarjeni podatkovni bazi, v primeru napak pa program sporoči, kaj je šlo narobe.



Slika 5.7 Ukazna vrstica, ki prikazuje ustvarjanje podatkovne baze za del spletnega portala v razvojnem okolju.

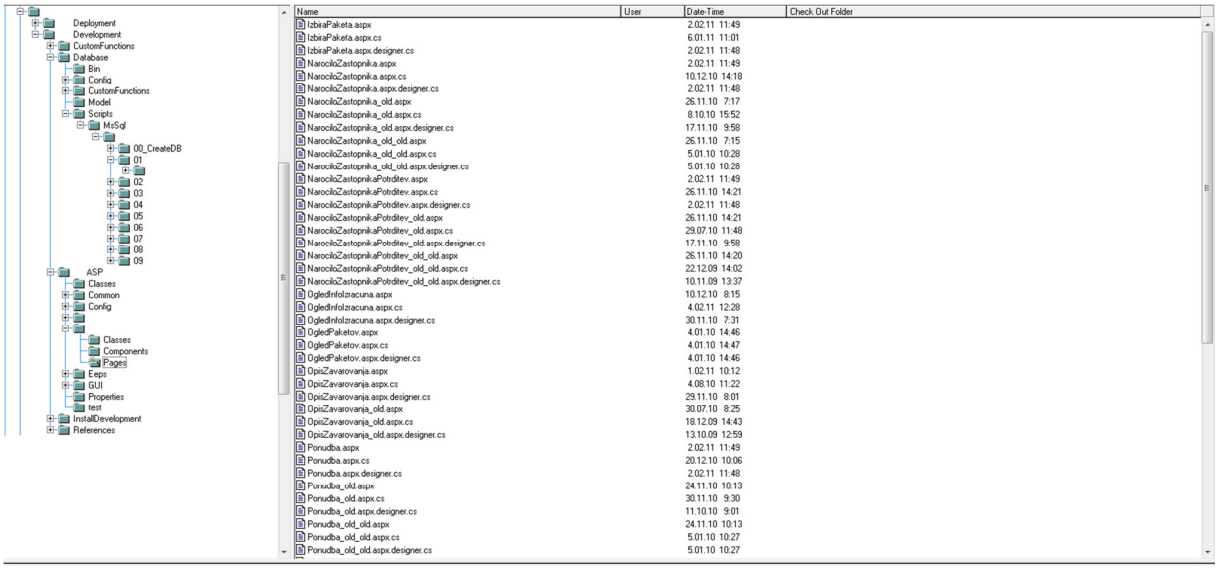
5.3 Načrtovanje in izdelava spletnega portala

Po natančnem pregledu funkcionalne specifikacije smo po posvetu s sodelavci morali najprej poimenovati celoten projekt, katerega ime bo najbolj jasno opredeljeval naročnika ter namen projekta. Ker gre v tem primeru za znano zavarovalniško družbo in spletni portal ter na nek način tudi za delček spletnega poslovanja, smo se odločili za ime s predpono »e«, tej pa smo dodali še ime zavarovalnice. Pred začetkom razvoja je bilo potrebno, v skladu s pravili, kreirati in urediti nov projekt s prej omenjenim imenom, in sicer s pomočjo internega dokumenta, v katerem so zapisana pravila o organizaciji izvorne kode, v vizualnem orodju za hranjenje izvorne kode projektov, Microsoft Visual Source Safe.

Le-ta se razdeli v dva imenika, development in deployment, kar pomeni, da v podstrukturo prvega spada razvojna programska koda, v drugega pa tista, ki je namenjena testerjem in naročniku. Ker se bo projekt vodil in razvijal v orodju Microsoft Visual Studio se v glavnem imeniku poleg omenjenih nahaja še samostojna datoteka z končnico *.sln, ki predstavlja rešitev in vsebuje projekte, ki spadajo v naš spletni portal (ang. Solution). Seveda ima tudi notranja struktura posameznega imenika že vnaprej določen videz, in sicer mora vsebovati imenik s projektom v Visual Studiu, katerega ime smo v skladu z internim dokumentom določili kot ime celotnega projekta, plus pripona ASP (ker se uporablja tehnologija ASP.Net), znotraj katerega pa se nahajajo podimeniki, ki vsebujejo programsko kodo. V skladu z že omenjenim internim dokumentom za ureditev programske kode se morajo ti podimeniki imenovati »Classes«, »Common«, »Config« ter »GUI«. Nato sledijo še imeniki podprojektov, ki so vključeni v del spletnega portala, njihovo število pa je odvisno od želje naročnika, koliko svojih zavarovalniških produktov bo ponujal na svoji spletni strani. Trenutno naročnik uporablja dva, v pripravi pa je že tretji produkt, ki pa ni del te diplomske naloge. Tako smo v projekt dodali tudi podimenika za zavarovanje doma ter zdravstvenega zavarovanja. Prvi od omenjenih šestih vsebuje razrede, ki so skupni vsem podprojektom, drugi vsebuje skupne strani, kar se po-navadi izkaže za pozdravno stran, katera vsebuje navigacijo po posameznih produktih. Tretji imenik, Config, vsebuje nastavitvene datoteke za vsako od štirih okolij, v katerem se bo uporabljalo portal. Podimenik GUI pa, kar nam že samo ime pove, vsebuje datoteke, ki skrbijo za grafični uporabniški vmesnik in njegov izgled, tu se nahajajo še css datoteke s stili, vse slikovne datoteke (ikone) ter JavaScript datoteke.

Struktura posameznega podprojekta pa mora biti sestavljena iz treh imenikov. Components, Classes in Pages. Že intuitivno poimenovanje nam pove, kaj se skriva v njih. HTML koda strani, razredi in komponente, ki so del določenega podprojekta.

V tem trenutku je pomembno omeniti tudi to, da projekt, v orodju za ohranjanje različic programske kode, ne vsebuje samo projekta iz razvojnega orodja Visual Studio, ampak tudi imenika »InstallDevelopment« ter »References«, ki vsebujeta programe, potrebne za nemoten razvoj, in jih je potrebno namestiti na delovno postajo pred razvojem, ter sklice na razvojne knjižnice, ki niso del Visual Studia. Tukaj se nahaja tudi imenik z imenom Database, katerega strukturo in vlogo smo spoznali v prejšnjem poglavju.



Slika 5.8 Struktura razvojne kode, pogled v orodju Microsoft VisualSourceSafe.

Ker je zavarovalništvo velik pojem in ima posamezna zavarovalnica ogromno poslovnih enot, kar skupaj z njimi prinese zapletene programe, namenjene samo agentom v pisarnah, agentom na terenu ter aplikacije za nadrejene, se je ena izmed vodilnih zavarovalnic v Sloveniji odločila, da bo svojim strankam ponudila aplikacijo, ki bo namenjena samo njim. Na zahtevo naročnika je zato v sklopu prenove spletnega mesta potrebno, namesto zelo podrobnega programa, z veliko funkcionalnosti, ki je namenjen agentom, izdelati kar precej okrnjen del celotnega spletnega portala, ki bo ponujal samo osnovne informacije njihovih zavarovalniških produktov. Ker si zavarovalnica želi odprtega sistema, je bila ena od zahtev tudi ta, da mora portal vsebovati anonimno sprehajanje po straneh, kar pa obenem vabi uporabnike k registraciji, s katero pridobijo nekaj funkcionalnosti. Portal bo razdeljen na dva dela, prvi bo imel skupne lastnosti vseh produktov in bo povezoval korporativnih del zavarovalnice z zalednim sistemom, drugi del pa bo namenjen vsakemu posameznemu produktu posebej, tako da je predmet te diplomske naloge razvoj prvega sklopa dela portala ter razvoj funkcionalnosti za dva produkta zavarovalnice.

Ko smo dovolj dobro razumeli funkcionalno specifikacijo in ko so bile odpravljene vse nejasnosti v zvezi s tem, smo pričeli z razvojem posameznih funkcionalnosti portala. Prva stvar, ki jo je moral po zahtevah naročnika vsebovati del portala, je bila enotna začetna stran, ki je skupna vsem produktom, ki so predstavljeni na portalu. Hkrati pa je že vnaprej pripravljena tako, da omogoča dodajanje novih produktov brez kakršnih koli omejitev. Kot že omenjeno, nam ravno ta funkcionalnost trenutno omogoča implementacijo tretjega produkta.
Za popolno skupno stran je bilo potrebno implementirati možnost dinamičnega dodajanja novih ikon, ki pomenijo nove produkte oz. nove funkcionalnosti strani. Izdelan je bil tudi prijavno-registracijski obrazec, ki še neregistriranemu uporabniku omogoča, da v novem oknu postane prijavljen uporabnik, po prijavi pa ima možnost pregledovanja shranjenih informativnih izračunov ter sklenjenih zdravstvenih zavarovanj. Tudi pregledovanje je implementirano dinamično in je zato v primeru takega dodatnega produkta zmožno takoj prikazovati shranjene predmete uporabnika. Seveda imajo prijavljeni uporabniki na voljo tudi možnost odjave.

Del portala, ki je namenjen posameznim produktom je bil razvit kar tako kot teče sam postopek izdelave informativnega izračuna za zavarovanje doma oz. sklenitve novega zdravstvenega zavarovanja za tujino. To pomeni, da so bile strani dodane v projekt in implementirane ena za drugo, poleg tega je bilo potrebno zagotoviti tudi ohranjanje podatkov med prehodi strani, kar smo zagotovili z uporabo seje. Prisotnih je veliko razredov in objektov, ki jih prav tako hranimo kar v seji, seveda pa brez poizvedb v podatkovni bazi portal ne bi bil takšen, kakršen je. Na koncu smo za zaključek sklenitve zavarovanja za tujino s pomočjo zunanje spletne storitve implementirali tudi plačilo preko spleta.

5.4 Opisi izbranih funkcionalnosti

5.4.1 Enotna registracija/prijava

Kot smo že omenili, je bila ena izmed naročnikovih zahtev enoten način prijave. To pomeni, da je bila tako podatkovna baza dela spletnega portala, kakor tudi sam portal zasnovan tako, da je možno dodajati nove module oz. produkte, prav tako pa je bilo potrebno zagotoviti konsistentnost podatkov med moduli. Na ta način so bili pozneje grajeni tudi moduli. Enotna registracija uporabniku omogoča, da si ustvari en uporabniški račun, preko katerega ima možnost dostopati do vseh modulov, ki jih ponuja del spletnega portala. Postopek registracije je enostaven in kot obvezen podatek zahteva le elektronski naslov in geslo. Novemu uporabniku so na voljo tudi vnosna polja za vpis osebnih podatkov (ime, priimek, EMŠO, datum rojstva) in podatkov o prebivališču. Omeniti velja tudi funkcionalnost samodejnih predlog krajev in poštnih številk ter ulic in njihovih hišnih številk, kar je implementirano s pomočjo interne spletne storitve. Po uspešno zaključeni registraciji uporabnik na podani elektronski naslov dobi potrditveno sporočilo, preko katerega aktivira svoj uporabniški račun, s tem pa si omogoči prijavo v spletni portal. Poleg takega načina registracije in prijave je možno spletni portal uporabljati tudi preko potrdila certifikata. Uporabnik v tem primeru vnese samo svoj elektronski naslov, na katerega dobi sporočilo z aktivacijsko povezavo.

Prijava je implementirana na način enkratne prijave za celotni sistem (ang. SSO – SingleSignOn), ki pomeni lastnost nadzora nad dostopanjem do več med seboj povezanih, vendar neodvisnih sistemov. S to lastnostjo se uporabnik prijavi le enkrat in pridobi dostop do vseh sistemov, ne da bi se moral na vsakem od njih prijaviti še enkrat. Obratno lastnost ima »SingleSignOff«, s katero se od vseh sistemov odjavimo le z enim klikom. Tak način prijave ima kar nekaj prednosti, saj smo s tem precej zmanjšali možnost ribarjenja (ang. phishing), ker večina uporabnikov vpisuje svoja gesla tudi tam, kjer ni potrebno. Ena izmed prednosti je tudi zmanjšana možnost, da bi uporabnik pomešal uporabniška imena in gesla, prav tako pa je uporabniku prihranjen čas, potreben za ponovno prijavo v drug sistem z istim uporabniškim imenom. Seveda pa ima tudi SSO svoje negativne lastnosti. Poglavitni problem takega načina prijave se imenuje »grajski ključ« (ang. keys to the castle). Ker nam SSO omogoča dostop do vseh sistemov, potem ko smo se uspešno prijavili, to povečuje negativen odziv, v primeru, da so bili prijavni podatki na voljo drugim osebam in hkrati zlorabljeni. Rešitev za ta problem bi lahko bila uporaba enkratnih gesel ali pametnih kartic. Drugi problem, ki se lahko pojavi, pa je nedelovanje storitve, ki upravlja prijavo, kar lahko pripelje do nedostopnosti vseh sistemov, ki jih imamo pod SSOjem, zato je tak način prijave precej nezaželen v sistemih, ki morajo nujno delovati ves čas.

5.4.2 Pregled informativnih izračunov/sklenjenih zavarovanj

Po uspešni prijavi ima uporabnik na začetni skupni strani na voljo seznam vseh informativnih izračunov, ki jih je shranil med postopkom izdelave, ter vseh sklenjenih zavarovanj, ki se shranjujejo samodejno. Seznam shranjenih predmetov je ločen po produktih, vsak sklop pa je sestavljen iz tabele s stolpci z imenom shranjenega predmeta, višino premije in datumom shranjenega predmeta. V sklopu, kjer so izpisana sklenjena zdravstvena zavarovanja za tujino, ima uporabnik na voljo tudi stolpec, v katerem je izpisan status plačila premije. V kolikor plačilo še ni bilo izvršeno, lahko uporabnik s klikom na gumb ta status osveži. Gumb za osvežitev statusa je prikazan samo takrat. Uporabnik lahko posamezne elemente seznama odpre na posebni, ločeni strani, kjer lahko pregleda postavke, višino premije in ostale podatke na polici. V primeru, da uporabnik nima shranjenih oz. sklenjenih zavarovanj, dobi izpisano ustrezno sporočilo, tabela pa ni prikazana.

5.4.3 Shrani informativni izračun

Da se posamezni info izračun uporabniku prikaže na skupni strani, ga mora predhodno tudi shraniti. Postopek shranjevanja izračuna je ločen na prijavljenega in anonimnega uporabnika, saj pri prvem sistem ne potrebuje dodatnih podatkov in njegov informativni izračun shrani direktno v svojo podatkovno bazo. Pri anonimnih uporabnikih pa sistem uporabnika najprej opozori, da se bo izračun shranil le za čas trenutne seje, potem pa bo izginil, hkrati pa s tem uporabnika prepriča, da se registrira in postane prijavljen uporabnik. Ko se uporabnik odloči shraniti izračun, mu sistem ponudi prednastavljeno ime izračuna, ki vsebuje ime kraja, v katerem se nahaja zavarovana nepremičnina, datum in višino premije. Ime lahko seveda spremeni po lastni želji, po kliku na gumb pa sistem izračuna še unikatno kombinacijo številk in znakov (ang. GUID), za poznejše iskanje in odpiranje shranjenega elementa, nato pa element shrani v ustrezno strukturo (baza ali seja).

5.4.4 Sklenitev in plačilo premije zdravstvenega zavarovanja za tujino

Shranjevanje zdravstvenih zavarovanj za tujino je, kot rečeno, samodejno, vendar pa mora uporabnik za to pripeljati postopek sklenitve do konca in izvesti plačilo premije. Ob zaključku je zavarovanje sklenjeno, ob dejanski izvršitvi plačila pa tudi veljavno in začne veljati od datuma, podanega med postopkom sklenitve. Plačilo premije se izvaja s pomočjo zunanje spletne storitve MegaPOS, na katero sistem preusmeri uporabnika, tam pa izbere vrsto kreditne kartice in vpiše številko kartice ter njeno veljavnost. Storitev nato avtorizira podatke in izvede plačilo ter uporabnika preusmeri nazaj na naš portal, kjer se izpiše ustrezna informacija o statusu plačila. V tem trenutku se sklenjeno zavarovanje tudi shrani v podatkovno bazo, uporabnika pa se preusmeri nazaj na prvo stran, kjer ima možnost videti status plačila. Omeniti pa je potrebno tudi to, da imajo funkcionalnost sklepanja zavarovanja na voljo le prijavljeni uporabniki.

5.4.5 Pošiljanje elektronskega sporočila o uspešnem zaključku

Po uspešnem zaključku sklepalnega ali informativnega dela se uporabniku na podani elektronski naslov pošljejo ustrezni podatki o polici in njenih postavkah. Uporabnik v elektronskem sporočilu dobi tudi spletno povezavo, na kateri si lahko ogleda shranjeno polico. Omeniti velja tudi to, da se v modulu informativnih izračunov zavarovanj doma ob dokončanju postopka anonimnega uporabnika shranjen element iz seje premakne oz. shrani v podatkovno bazo, z namenom njegovega poznejšega prikazovanja. Za implementacijo te funkcionalnosti je bilo potrebno nastaviti tudi poseben SMTP strežnik, ki skrbi za pošiljanje elektronskih sporočil uporabnikom.

5.5 Varnost

Internet kot nova oblika virtualnega prostora, v katerem je oblika anonimnosti in svobode razmeroma velika, predstavlja nove izzive na področju varnosti. Zaradi svoje zasnove ga je nemogoče v celoti obvladovati, zato je na vsakem posameznem udeležencu odgovornost za zagotavljanje varnosti in zasebnosti podatkov, ki jih oddaja oz. prejema. Žal z razvojem metod za varen prenos in zaščito podatkov poteka tudi vzporeden razvoj metod za prestrezanje in poneverbo podatkov, slednji marsikdaj prehiteva prvega. Spletno poslovanje z občutljivimi finančnimi transakcijami predstavlja še poseben izziv pri zaščiti.

Spletni portal za svoje delovanje uporablja različne načine zagotavljanja varnosti svojih uporabnikov in njihovega poslovanja, eden izmed njih pa je uporaba certifikatov. Poleg tega mora biti uporabnik tudi sam precej pazljiv, kdaj in kako vpisuje svoje podatke na internetu, saj večina napadov na spletno poslovanje danes temelji na zavajanju uporabnika, pri čemer napadalci uporabniku odtujijo podatke, potrebne za prijavo v spletno poslovanje. Nekatere od metod, s katerimi napadalci poizkušajo pridobiti podatke so:

- spletno ribarjenje (ang. phishing),
- zvabljanje (ang. pharming),
- napad XSS (ang. cross-site scripting),
- beleženje (ang. keystroke logging),
- trojanski konj (ang. trojan horse)

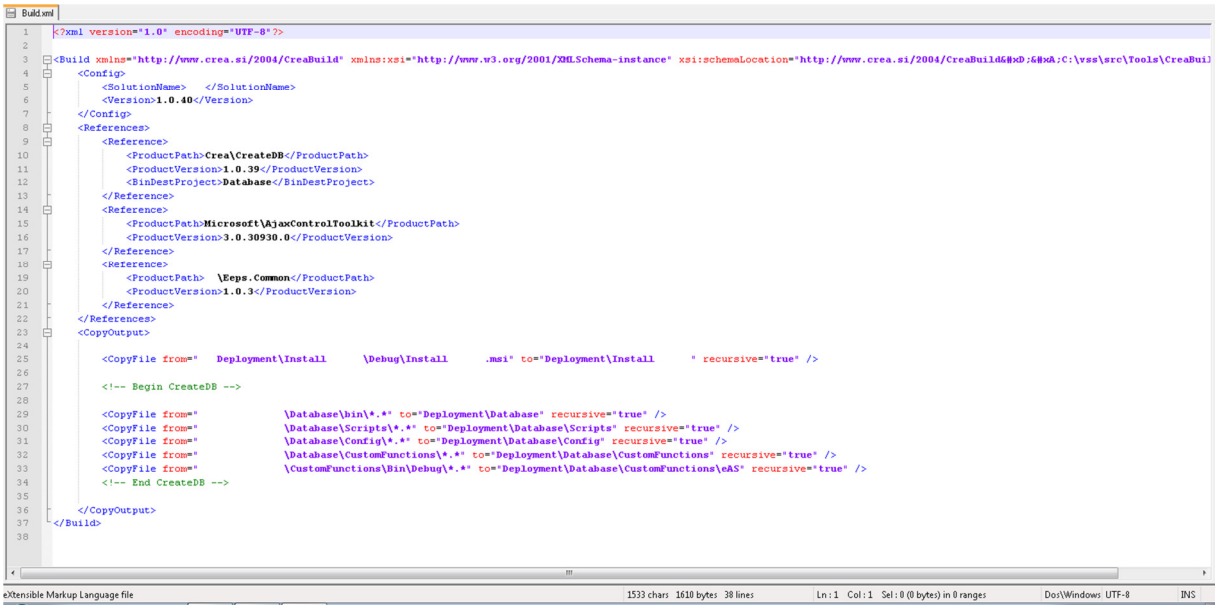
Tem napadom je skupno, da z različnimi metodami uporabnika prevarajo, da zlonamernemu programu zaupajo in posredujejo podatke, potrebne za prijavo v spletno poslovanje. Ti programi nato te podatke zlorabijo za vstop, v večini takih primerov, v spletno banko, kjer se denarna sredstva preusmerijo na račun lastnika programa. Uporabnik se proti temu lahko zavaruje z uporabo varnega operacijskega sistema in spletnega brskalnika ter skrbnega varovanja svojih prijavnih podatkov. Metode za zagotavljanje varnosti so osnovne in preproste – izbira varnih komponent, redno posodabljanje programske opreme, ki preprečuje zlonamerna dejanja, pazljivost pri prejemanju in poganjanju nepoznane programske opreme. Kljub vsem naprednim varnostnim mehanizmom se v praksi izkaže, da je najšibkejši člen v varnostni verigi ravno uporabnikova neizkušenost in naivnost.

Del spletnega portala z informativnimi izračuni, uporablja za preverjanje uporabnika poleg gesla tudi kvalificirano digitalno potrdilo (certifikat), ali na kratko KDP, javnega overitelja, ki je pooblaščen za izdajo certifikatov. Certifikat tako enolično predstavlja uporabnika in je v primerih poslovanja s spletnimi bankami tudi nujno potreben za vstop. Portal zavarovalnice je v-naprej pripravljen tako, da bo v prihodnosti mogoče police podpisovati tudi elektronsko, s pomočjo posebnega vtičnika za spletni brskalnik, podpisne komponente.

5.6 Izdelava namestitvenega paketa

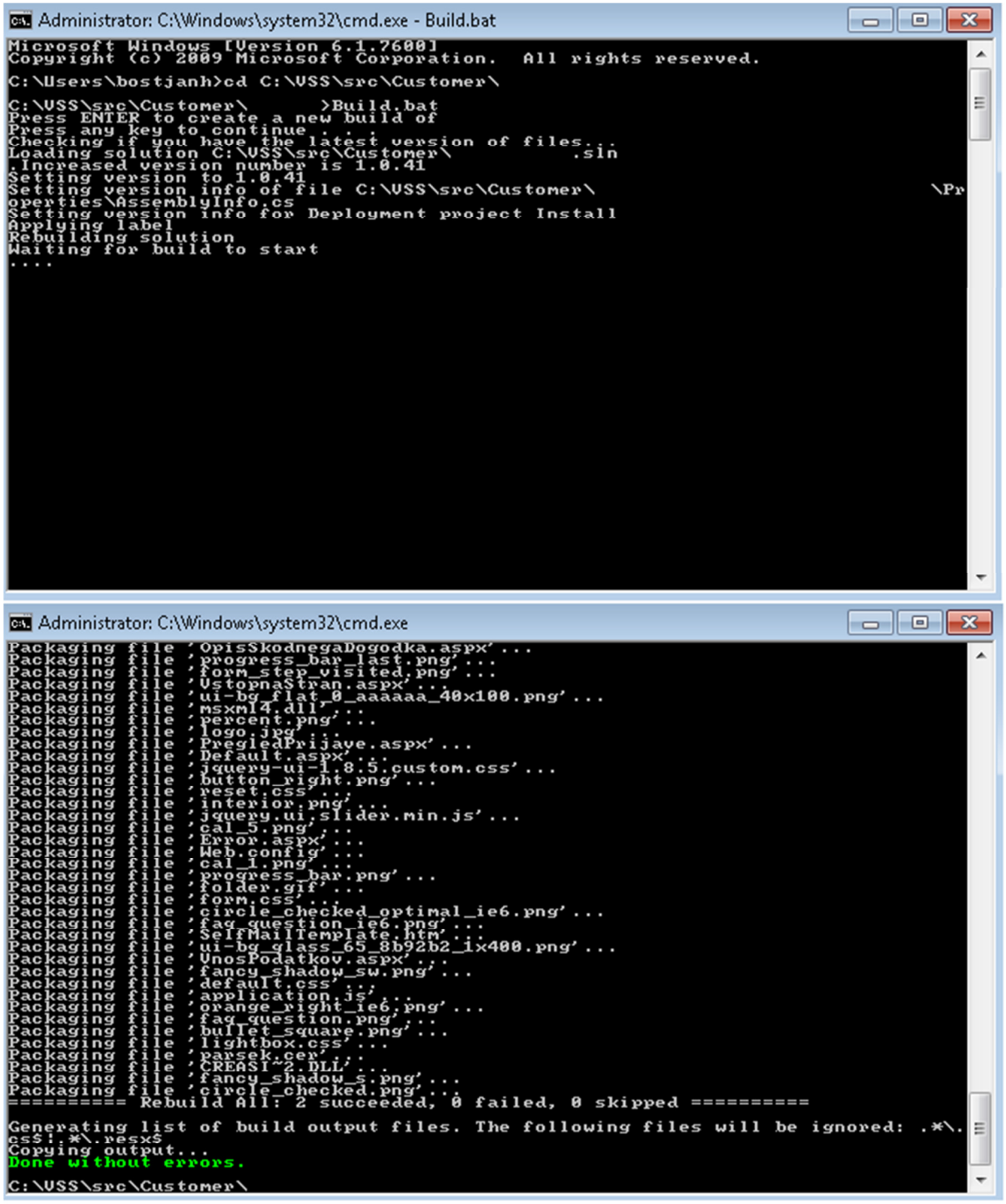
Po končanem razvoju vseh funkcionalnosti, ki so bile zapisane v funkcionalni specifikaciji, se razvijalec loti hitrega testa in odpravi ugotovljene napake. Veliko pozornosti se posveča predvsem pravilnemu delovanju preverjanja podatkov na obrazcih, pri katerih se skuša omejiti čim več morebitnih ranljivosti delovanja aplikacije, na primer: vpisovanje črk namesto številk (in obratno), vpisovanje negativnih števil (v polja, kjer je predviden vnos zneskov), … Večkrat pa že kar med razvojem onemogočimo vnašanje podatkov v določena polja, v primeru, ko niso vneseni vsi predhodni potrebni podatki, če le to ni bilo zahtevano drugače.

Po odpravi napak je potrebno rešitev pripraviti na namestitev. Ker izvajalec uporablja interni program, ki rešitev predela in izdela namestitveni paket, je bilo potrebno rešitev, tako kot bazne skripte, malo prilagoditi. Ustvarili smo poseben projekt znotraj rešitve, z imenom »Installer«. V projektu smo podali tri možnosti izbire, preko katerih povemo, v katero okolje se bo aplikacija namestila, v naslednjem koraku imamo možnost izbrati lokacijo namestitve na datotečnem sistemu, na zadnjem koraku pa namestitev še potrdimo s klikom na gumb »dokončaj«. Ker gre za standardni namestitveni paket s končnico »*.msi«, ki je s svojim, uporabniku prijaznim, grafičnim vmesnikom precej razumljiv, smo razvoj s tem zaključili. Preden pa uporabimo interni program za predelavo rešitve, »Build«, moramo ustvariti še nastavitveno datoteko. V skladu z internim poimenovanjem datotek bomo ustvarili datoteko končnice *.xml in ji dali enako ime, kot ga ima program – »Build«. V tej datoteki so zapisane vse potrebne datoteke, ki se morajo predelati iz rešitve v namestitveni paketek, poleg teh se v namestitev predelajo tudi bazne skripte in mogoče reference iz drugih projektov ali knjižnic.



Slika 5.9 Nastavitvena datoteka Build.xml.

Ko imamo pripravljeno datoteko Build.xml, lahko preko Windows ukazne vrstice poženemo program »Build«. Program prebere nastavitveno datoteko in izvede vse, kar smo napisali vanjo, potem pa požene še projekt znotraj rešitve, ki smo ga omenili prej, preko katerega se ustvari namestitveni paket. Na koncu program še skopira vse datoteke na poseben interni strežnik, kjer se hranijo vsi namestitveni paketi in potrebne datoteke za namestitev in poganjanje aplikacij.



Slika 5.10 Ukazna vrstica, preko katere poženemo interni program »Build«, na koncu vidimo sporočilo o uspešni izdelavi namestitvenega paketa.

6 POSTOPEK TESTIRANJA, NAMEŠČANJE K NAROČNIKU, PREDAJA V UPORABO

6.1 TESTIRANJE

Potem, ko je bilo razvojno testiranje uspešno dokončano in se je uspešno izvedel tudi program Build, podporna ekipa iz internega strežnika vzame namestitveni paket in ga prenese v datotečni sistem strežnika, ki je namenjen internemu testnemu okolju. Podporna ekipa s pomočjo razvijalca izvede namestitev aplikacije, kjer med postopkom nameščanja na grafičnem vmesniku namestitvenega paketa izbere ustrezni gumb s testnim okoljem. Ko se je namestitev aplikacije končala, je potrebno ustvariti še podatkovno bazo. Postopek je precej podoben tistemu iz razvoja, le da je tukaj potrebno v Windows ukazni vrstici pognati interni program »CreateDB«, s parametrom »test environment«. Ko sta nameščeni obe potrebni stvari, se interni tehnolog – tester, s pomočjo razvijalca loti testiranja aplikacije.

Testiranje se opravi s pomočjo v-naprej pripravljenih testnih scenarijev, preverijo pa se vse funkcionalnosti, ki so zapisane v funkcionalni specifikaciji. Ob zaključenem testu mora tester razvijalcu predati izhodni dokument – zapisnik testiranja, v katerem so zapisane vse ugotovljene napake, tako funkcionalne kot pravopisne. Skupaj nato dokument pregledata, si izmenjata mnenja, potem pa mora razvijalec te napake odpraviti. Ko so ugotovljene napake odpravljene, se ponovi postopek ustvarjanja namestitvenega paketa, namestitve in ponovnega testiranja. Če je med odpravljanjem napak prišlo tudi do spremembe v baznih skriptah je med ponovnim nameščanjem potrebno na novo ustvariti tudi podatkovno bazo. Tester nato pregleda, če so bile ugotovljene napake odpravljene, nemalokrat pa se v tem primeru zgodi, da je razvijalec med opravljanjem napak ugotovljeno sicer odpravil, vendar s tem pokvaril kakšno drugo funkcionalnost, zato se testni cikel lahko ponovi tudi večkrat.

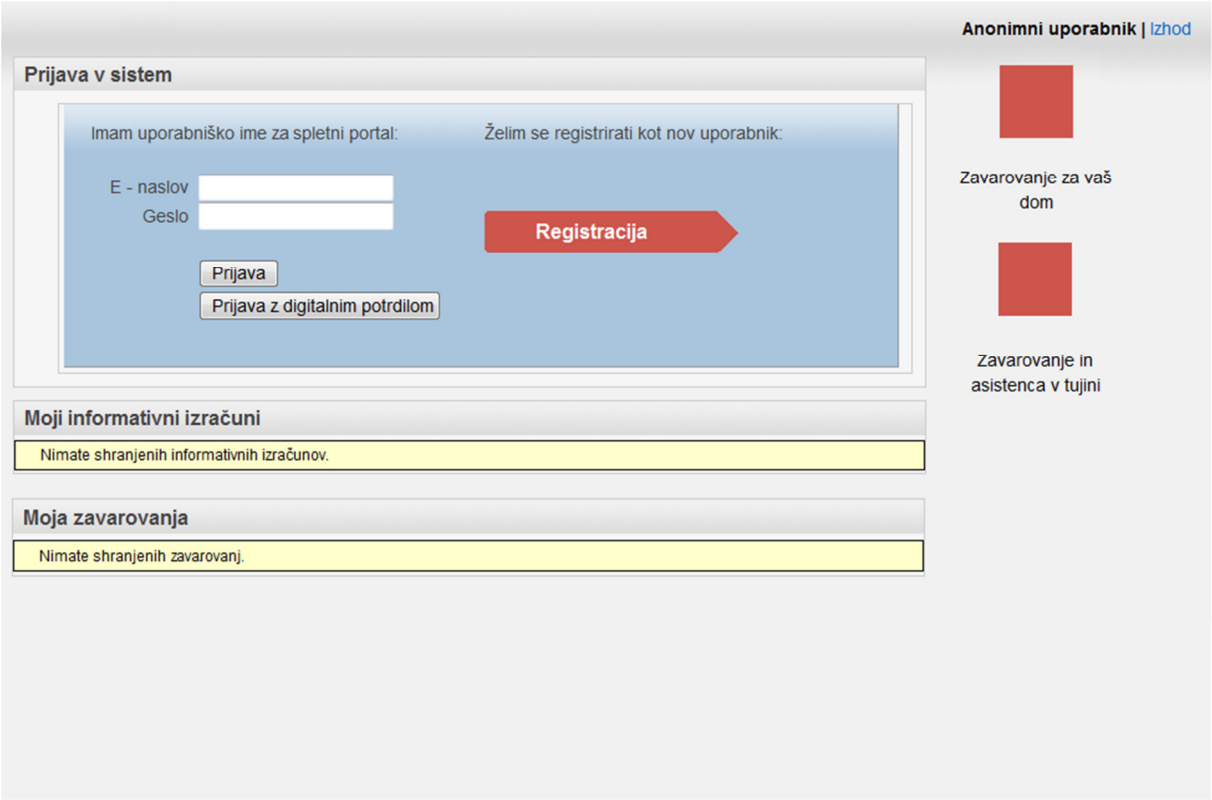
6.2 PRODUKCIJA

Ko so odpravljene vse napake in je testni cikel zaključen, se lahko aplikacija namesti k naročniku. Najprej se jo namesti v naročnikovo testno okolje, postopek namestitve pa je ponovno precej podoben zgoraj omenjenemu, le da je tukaj potrebno na grafičnem vmesniku namestitvenega paketa izbrati naročnikovo testno okolje, za namestitev oz. ustvarjanje podatkovne baze pa se program »CreateDB« pokliče s parametrom »test production«. Naročnikovi tehnologi imajo nato dovolj časa, da vse skupaj podobno kot interni testerji tudi sami preizkusijo. Ob koncu se prav tako posreduje zapisnik testiranja, nato pa se ob morebitnih ugotovljenih napakah le-te odpravijo, ustvari se nov namestitveni paket in aplikacija se ponovno namesti v naročnikovo testno okolje. Po pregledu odpravljenih napak naročnik izvajalcu sporoči ustreznost aplikacije, nato pa skupaj podpišejo prevzemni zapisnik, aplikacija in podatkovna baza se namestita v naročnikovo produkcijsko okolje, kjer je nato portal pripravljen za uporabo preko interneta. S tem naročnik tudi opravi uradni prevzem aplikacije.

7 VIDEZ IN DELOVANJE

Uporabniki širom sveta preko spleta uporabljajo veliko število aplikacij, pa naj bodo to aplikacije socialnih omrežij ali pa neke specifične namenske aplikacije. Tako se lahko tudi uporabniki našega spletnega portala le-tega poslužujejo preko spleta, do aplikacije pa jih vodijo povezave iz uradne spletne strani zavarovalniške družbe. Uporabnik ima na voljo dve možnosti dostopa, prva je preusmeritev na skupno stran, kjer nato izbira možnosti med podprtimi zavarovalniškimi produkti, drugi način dostopa pa je preusmeritev iz podprtega zavarovalniškega modula znotraj uradne spletne strani, od koder se preusmerimo na začetno stran modula dela spletnega portala z informativnimi izračuni.

Začetna skupna stran vsebuje navigacijo podprtih modulov in pregled shranjenih predmetov prijavljenega uporabnika, anonimnemu uporabniku pa namesto pregleda ponudi možnost registracije, kar prikazujeta sliki 7.1 in 7.2.



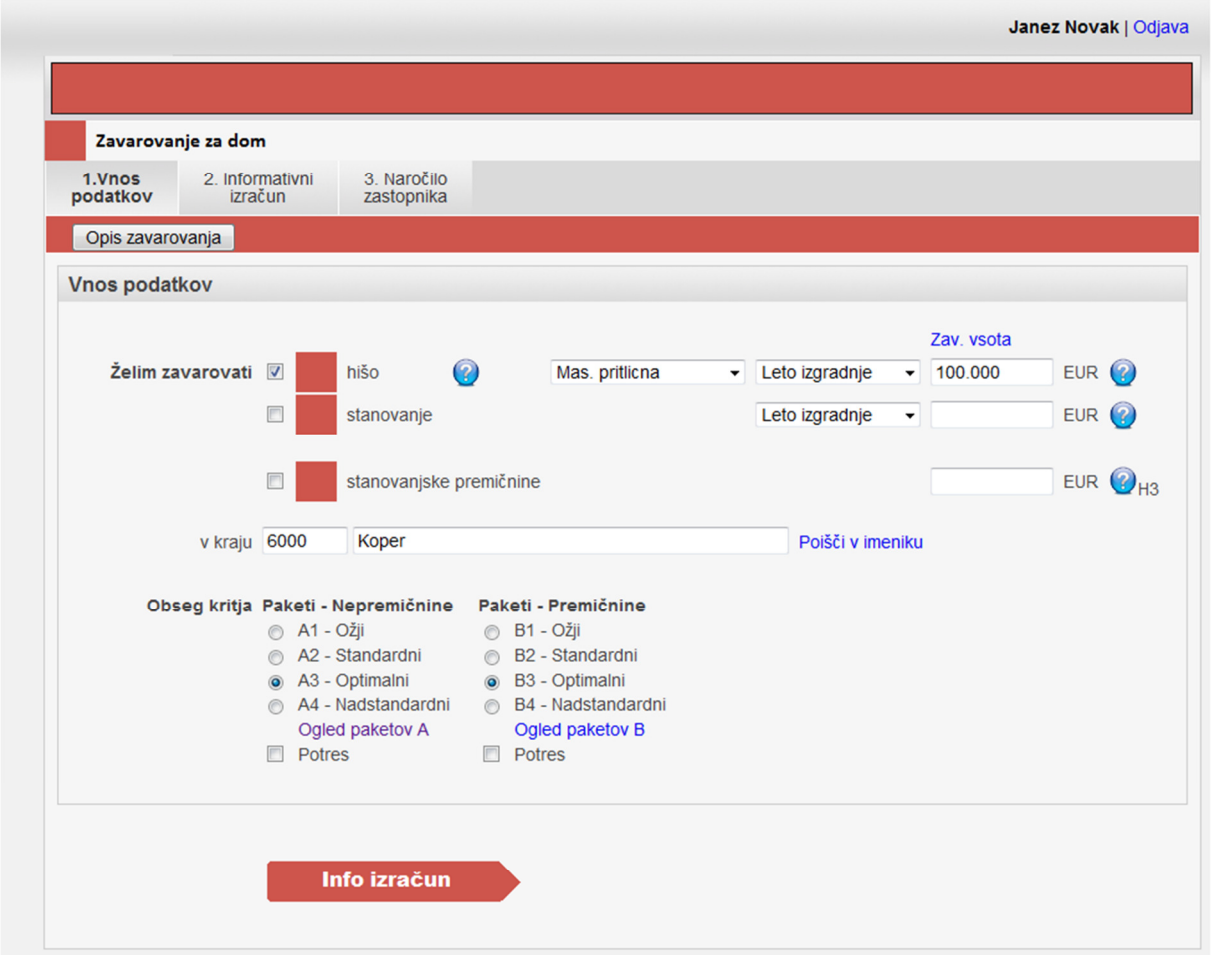
Slika 7.1 Primer skupne strani, kot jo vidi anonimni uporabnik.

Slika 7.2 Primer skupne strani, kot jo vidi prijavljen uporabnik.

Ob prihodu na skupno spletno stran uporabniku ponudimo tudi registracijo in v primeru, da se le-ta za to odloči, mora v obrazec vnesti potrebne osebne podatke in potrditi registracijo. Sistem uporabniku na podani elektronski naslov samodejno pošlje elektronsko sporočilo, ki vsebuje povezavo, preko katere lahko uporabnik aktivira svoj uporabniški račun. Postopek registracije uporabniškega računa prikazuje slika 7.3.

Slika 7.3 Primer registracijskega obrazca.

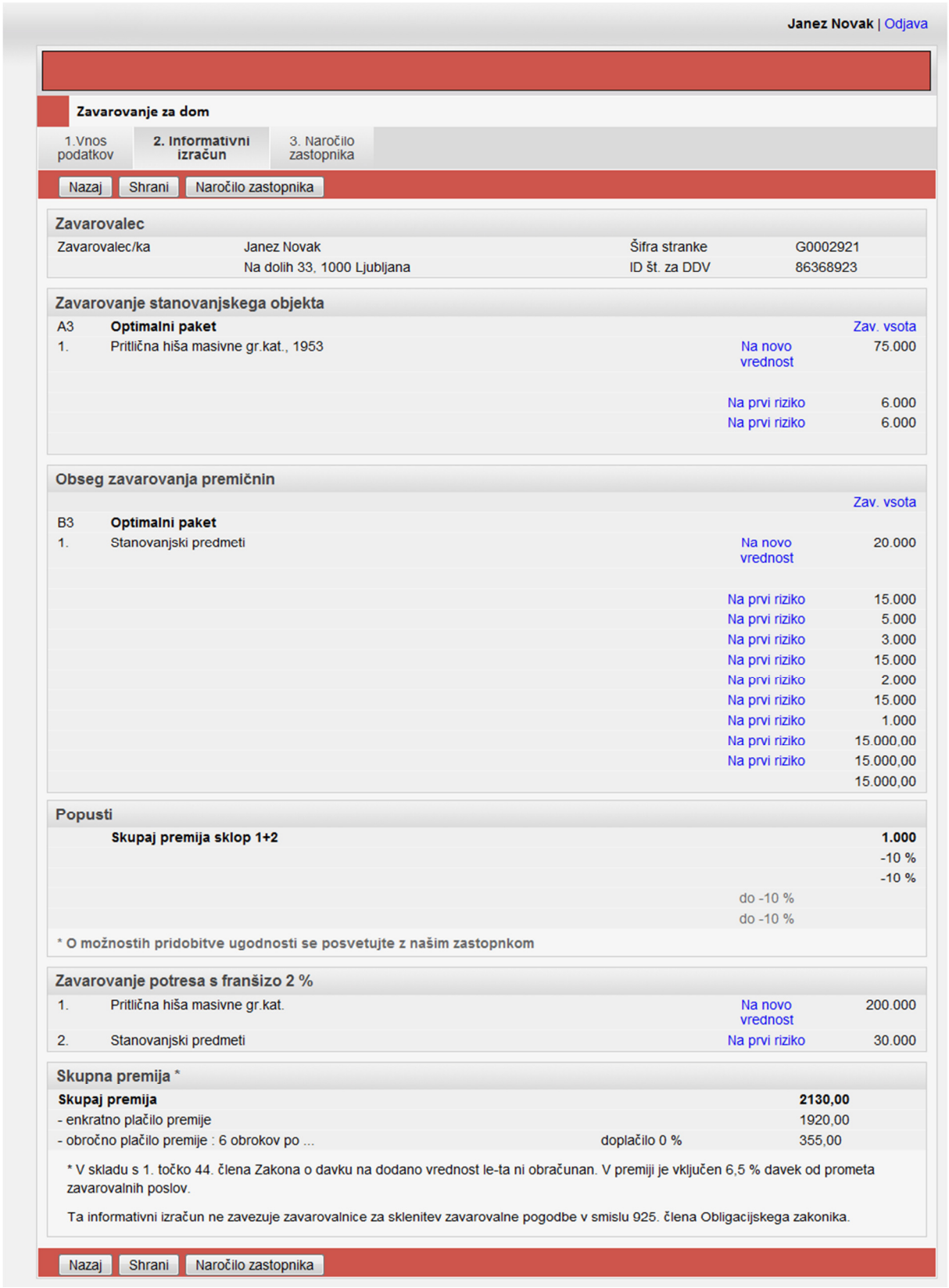
Postopek ustvarjanja informativnega izračuna zavarovanja za dom uporabnik prične z vnosom osnovnih podatkov o (ne)premičnini, kar je razvidno iz slike 7.4. Stran vsebuje prikaz zavihkov, ki predstavljajo korake informativnega izračuna. Na strani mora uporabnik izbrati vrsto zgradbe (hiša ali stanovanje), vnesti kdaj je bil objekt zgrajen ter kakšna je skupna vrednost nepremičnine. Poleg tega lahko zavarujemo tudi stanovanjske premičnine (klavir, umetnine, …), za katere je prav tako potrebno vpisati njihovo vrednost. Na koncu ima uporabnik možnost skleniti tudi dodatno zavarovanje proti potresu, nato izbere še paket kritij in s klikom na gumb »Info izračun« nadaljuje na naslednji korak.



Slika 7.4 Primer strani za vnos osnovnih podatkov o (ne)premičnini.

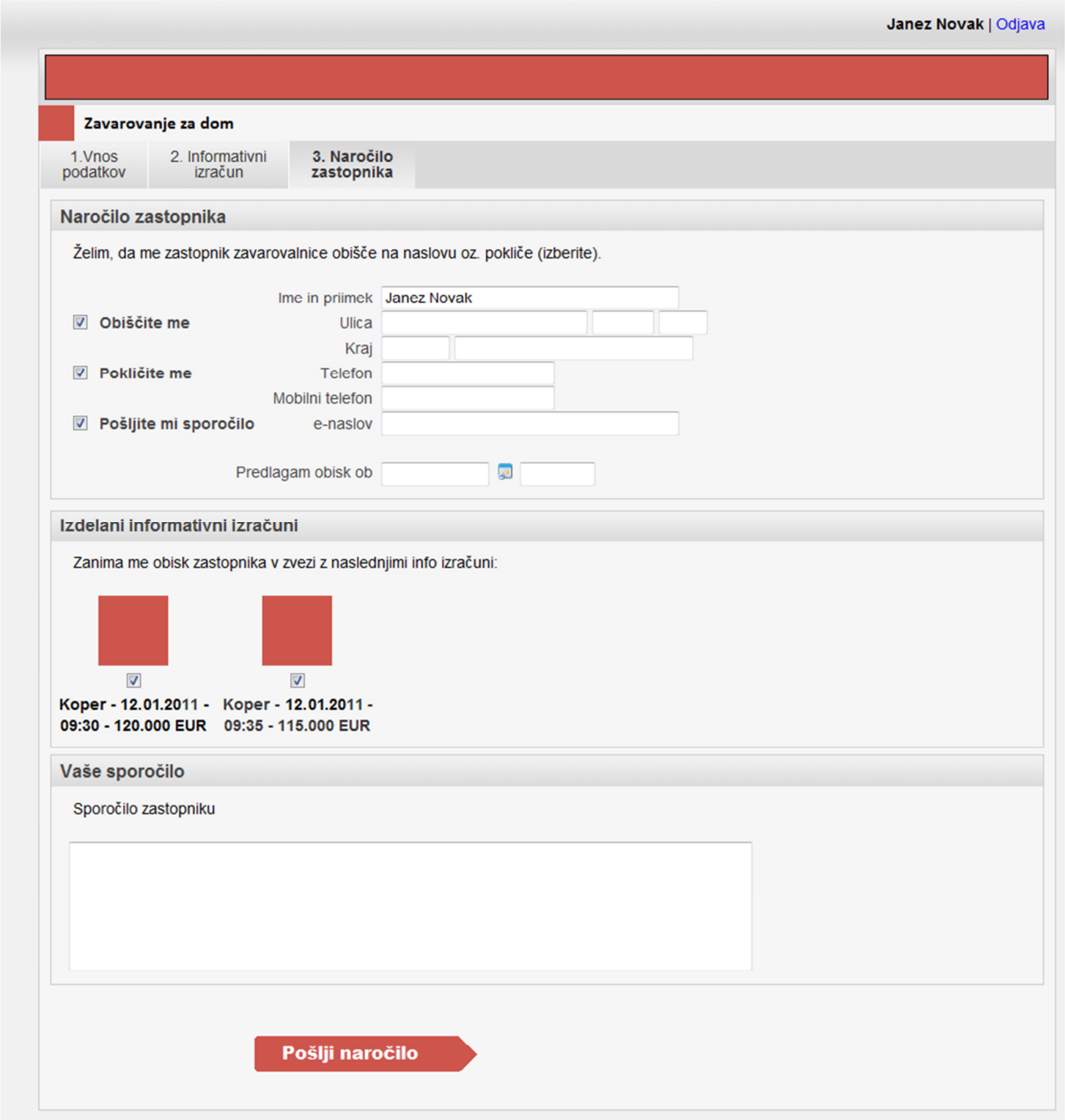
Preden uporabnik izbere obseg kritja, si lahko v samostojnem oknu ogleda tudi primerjavo kritij, ki jih vključujejo posamezni paketi (ne)premičnin. Do primerjave uporabnik lahko pride s klikom na povezavo »Ogled paketov A« oz. »Ogled paketov B«.

Na naslednjem koraku (slika 7.5) so uporabniku prikazane postavke na polici in posamezni zneski ter ostali podatki, na koncu ima izpisano tudi skupno premijo in možnost dodatnih popustov, ki mu jih lahko odobri zavarovalniški agent ob dejanski sklenitvi police.



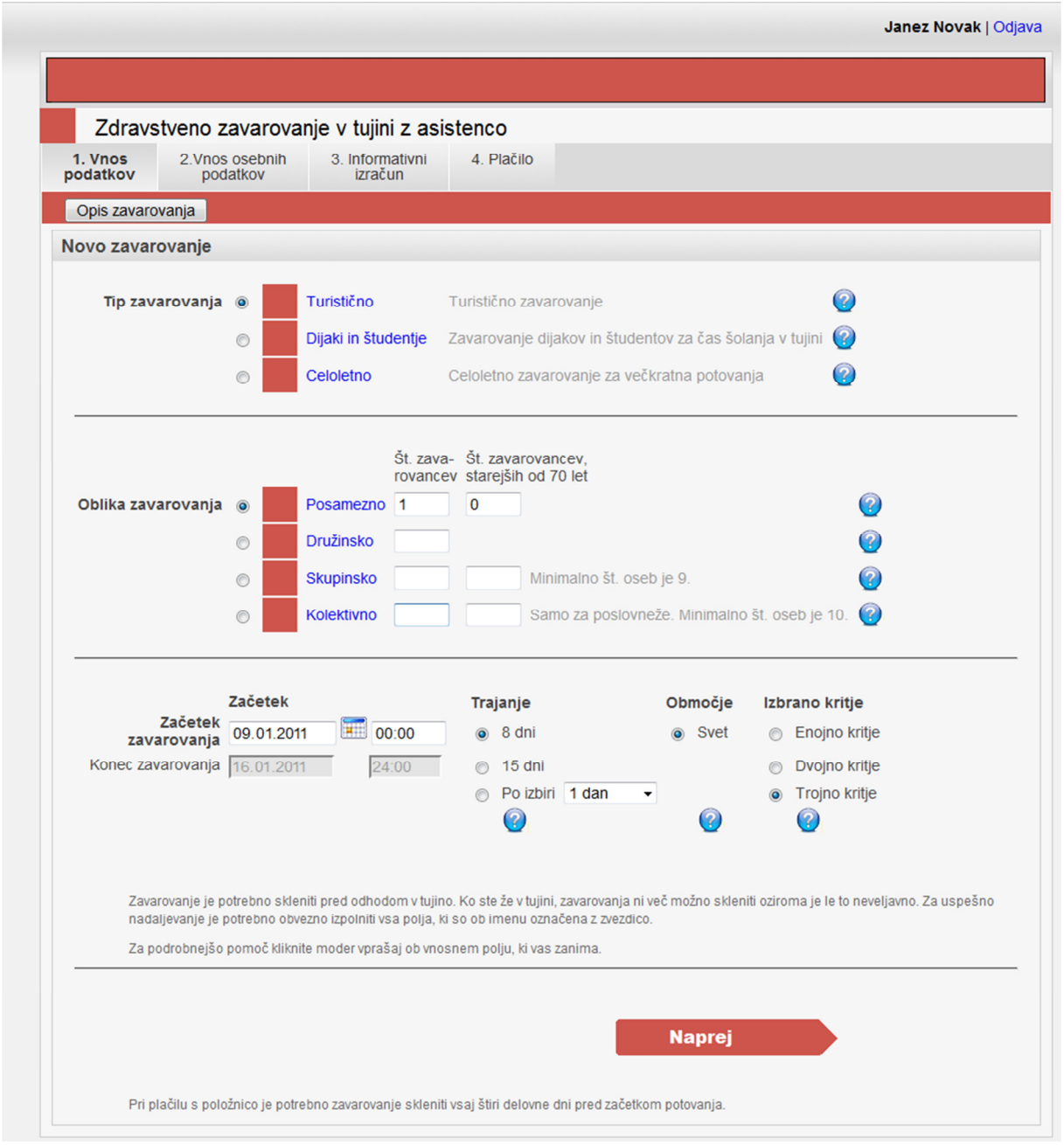
Slika 7.5 Primer strani za pregled postavk in posameznih zneskov na polici.

Na zadnjem koraku (slika 7.6) ustvarjanja informativnega izračuna uporabnik samo še vnese svoje podatke in želeni način kontakta z zastopnikom, nato iz seznama shranjenih informativnih izračunov izbere tistega, o katerem se z zastopnikom želi pogovarjati ter s klikom na gumb »Pošlji naročilo« potrdi naročilo. Uporabniku in agentu, ki je določen za prejemanje obvestil, sistem samodejno pošlje elektronsko sporočilo, uporabniku se izpiše prijazno obvestilo, kaj se je zgodilo v ozadju, nato pa uporabnik lahko svoje delo nadaljuje s preusmeritvijo nazaj na prvo stran portala.



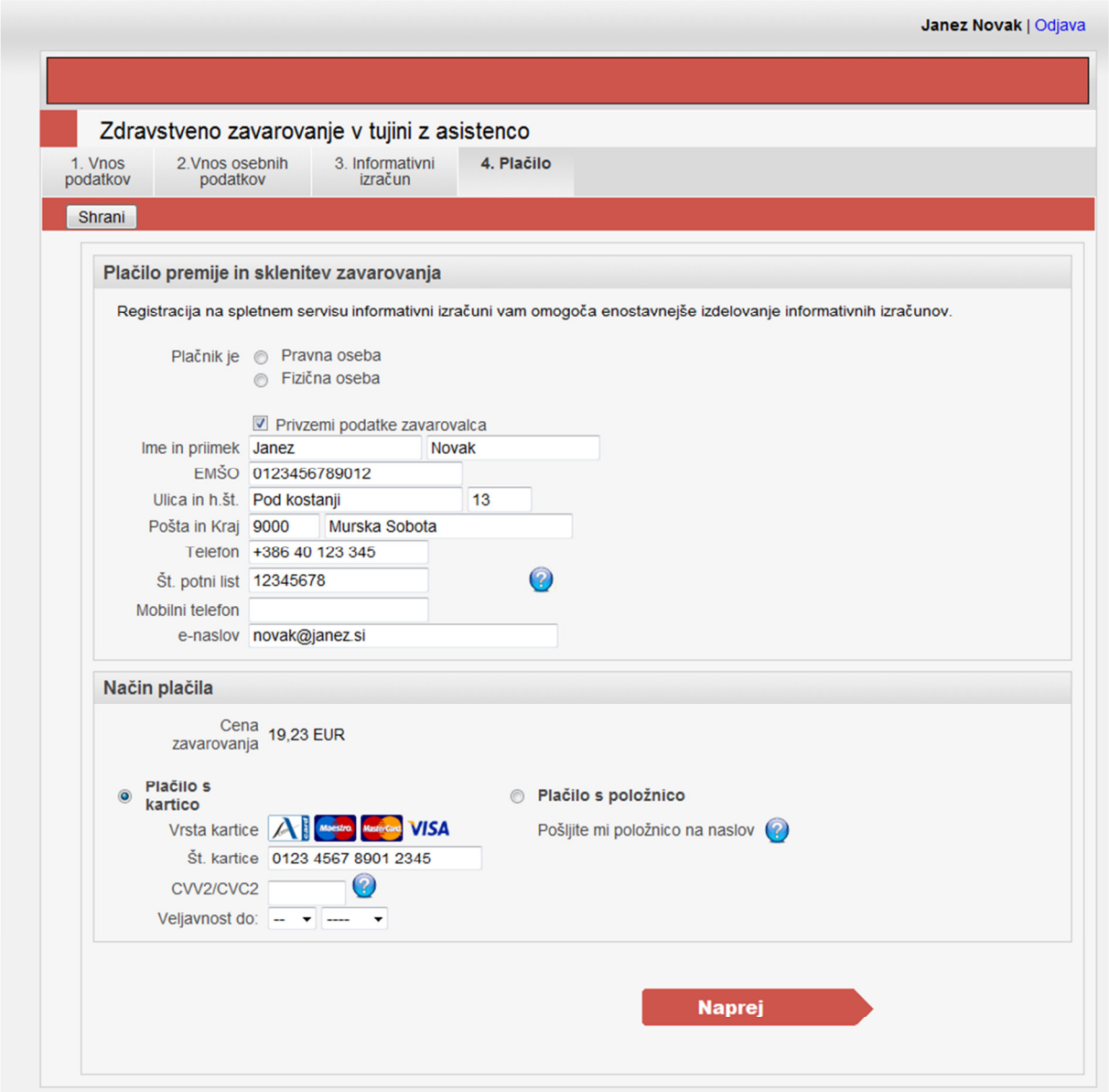
Slika 7.6 Primer strani za naročilo zastopnika.

Podobno kot prvi modul, se tudi drugi prične z vnosom osnovnih podatkov, ki jih sistem potrebuje za ustvarjanje nove police zdravstvenega zavarovanja za tujino z asistenco. Tudi ta stran vsebuje pregled korakov sklenitve z zavihki. Na strani je potrebno izbrati tip in obliko zavarovanja ter vpisati, kdaj naj zavarovanje začne veljati in koliko časa naj velja. Poleg tega je potrebno izbrati tudi obseg kritja, ki ga upošteva asistenca za tujino. Celotno izbiro prikazuje slika 7.7.



Slika 7.7 Primer strani za vnos osnovnih podatkov o zavarovanju.

Vnosu osnovnih podatkov sledi pregled, nato pa uporabnik s klikom na gumb »Naprej« delo nadaljuje na naslednjem koraku, kjer mora podati svoje osebne podatke in izbrati način plačila (položnica ali kreditna kartica). Nato je v primeru plačila s kreditno kartico preusmerjen na zunanjo spletno storitev, kjer se plačilo dejansko izvede. Po končanem postopku plačila je uporabnik preusmerjen nazaj na pregled police s postavkami in zavarovanimi osebami ter zneskom premije. Uporabnik nato lahko nadaljuje z delom, samodejno pa je preusmerjen nazaj na skupno stran. Postopek pred preusmeritvijo na zunanjo spletno storitev MegaPOS prikazuje slika 7.8.



Slika 7.8 Primer izbire načina plačila.

8 VZDRŽEVANJE, NADGRADNJE

Vzdrževanje opredeljujemo kot fazo, ki praviloma sledi uspešni vpeljavi informacijske rešitve v produkcijsko okolje. Potreba po vzdrževanju izhaja iz sprememb v delovanju rešitve v produkcijskem procesu, te spremembe pa se pojavljajo na področjih delovnih procesov, poslovnih pravil, produkcijskega okolja ter informacijskih tehnologij. Ker vsaka izmed naštetih sprememb vpliva na poseg v rešitev, ima vzdrževalna ekipa kar nekaj nalog, ki se začnejo že med razvojem, te naloge pa morajo biti skrbno načrtovane od samega začetka.

Naloge se po-navadi zapišejo v dokument, imenovan načrt vzdrževanja IR, ki mora vsebovati način, na katerega naročnik preko svojih uporabnikov zahteva spremembe v IR ali prijavlja napake. Poleg tega mora načrt natančno določati tudi oceno sredstev, porabljenih za odziv na vzdrževalni zahtevek. Vzdrževanje IR po-navadi traja približno 5 do 6 let po tem, ko je bila rešitev nameščena v produkcijo oz. predana naročniku.

Šest najbolj pogostih procesov vzdrževanja:

1. Izvajanje procesov vključuje pripravo rešitve in začetek dejavnosti, kot so zasnova in oblikovanje načrta vzdrževanja, pripravo na obravnavanje težav, ugotovljenih med razvojem, in spremljanje upravljanja rešitve.
2. Proces analize težav in sprememb, ki se izvede takoj, ko je rešitev postala del odgovornosti vzdrževalne ekipe. Razvijalec mora analizirati vsako zahtevo, jo potrditi in preveriti njeno veljavnost, jo proučiti in predlagati rešitev, nato pa v ustrezen dokument zapisati ugotovljeno – zahtevo in predlagano rešitev. Na koncu mora razvijalec pridobiti še vsa potreba dovoljenja za vpeljavo spremembe, ki odpravi težavo.
3. Proces, ki zadeva samo uveljavitev spremembe.
4. Proces sprejema spremembe s potrditvijo posameznika, ki je podal zahtevek. S tem se izvajalec prepriča, da je sprememba prinesla željen rezultat.
5. Migracijski proces je izjema in ni del vsakodnevnih nalog vzdrževalne ekipe. Če je potrebno rešitev prenesti na drugo platformo, brez kakršne koli spremembe v funkcionalnosti, se uporabi ta proces, naloga pa se bo verjetno dodelila projektno vzdrževalni ekipi.
6. Proces vzdrževanja, dogodek, ki se ne zgodi vsak dan, je tudi odstranitev dela rešitve iz produkcijskega okolja.

Kakor koli že, pa je poglavitna naloga vzdrževalne ekipe spremljanje delovanja rešitve ter odzivanje na zaznane nepravilnosti, kakor tudi izvajanje potrebne spremembe ali ustrezne dograditve rešitve. Značilnost vzdrževanja je predvsem v tem, da se stalno odvija in je del življenjskega cikla IR.

8.1 Načini vzdrževanja

V teoriji obstajajo štirje najbolj pogosto omenjeni načini vzdrževanja:

1. Korektivno vzdrževanje – reaktivne spremembe rešitve, ki odpravijo ugotovljene napake. Spremembe so vpeljane po predaji rešitve naročniku.
2. Prilagodljivo vzdrževanje – spremembe rešitve, ki se opravijo po predaji naročniku, z namenom ohranitve uporabne rešitve v spremenjenem ali spreminjajočem se okolju.
3. Dovršeno vzdrževanje – sprememba rešitve po predaji zaradi izboljšanega delovanja ali lažjega nadaljnjega vzdrževanja.
4. Preventivno vzdrževanje – sprememba v rešitvi po predaji, z namenom odkrivanja in odpravljanja skritih napak, preden se te dejansko pojavijo.

V praksi pa se pri izvajalcu uporabljata le dva načina, ki pa vseeno delujeta na podlagi prvega zgoraj omenjenega:

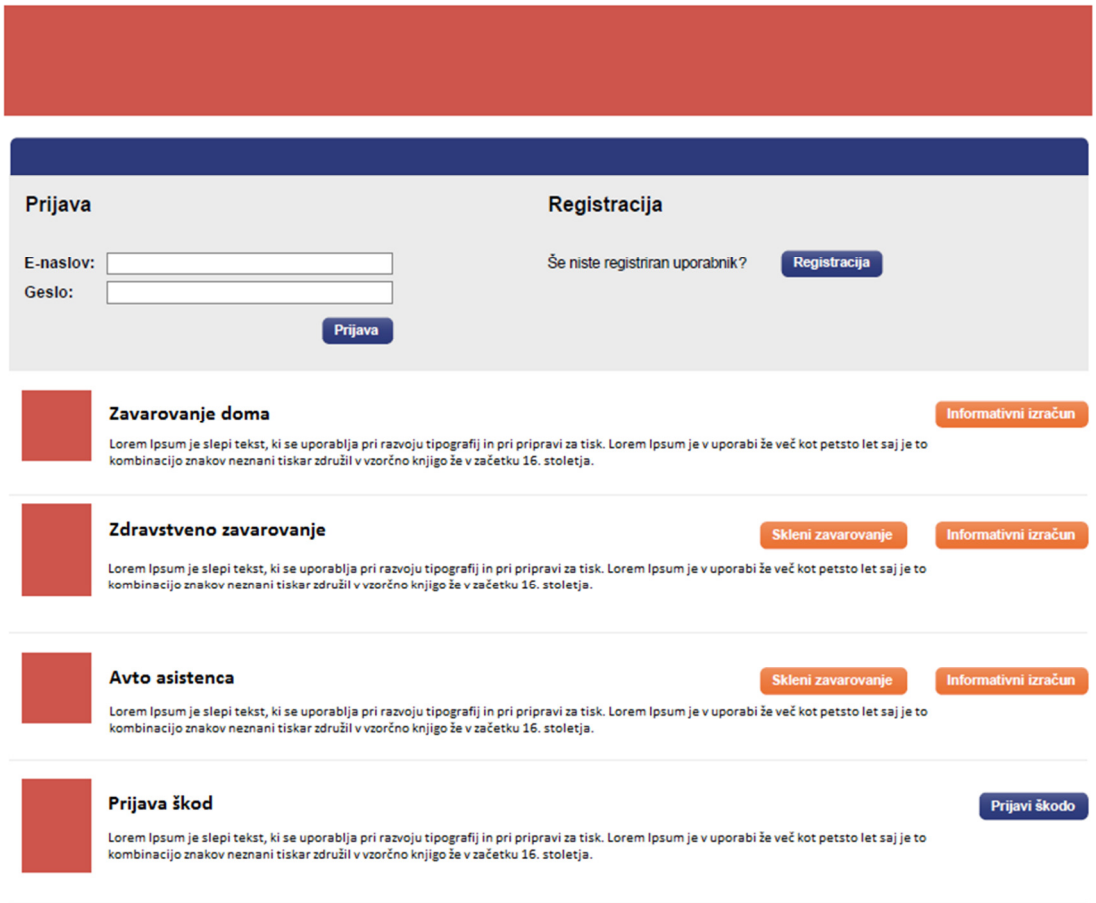
1. Vzdrževanje po pogodbi – naročnik ima z izvajalcem sklenjeno pogodbo, ki se podpiše ob predaji rešitve naročniku, njena veljavnost pa je lahko za določen (pet let) ali nedoločen čas. V okviru takega vzdrževanja naročnik mesečno plačuje določeno vsoto denarja, za katero ima pri izvajalcu zagotovljena dva razvijalca in določeno število ur (na primer 20), ki jih ta dva razvijalca porabita za vse, kar naročnik prijavi (analiza, odprava, namestitev). Napake se prijavljajo preko internega sistema za sporočanje.
2. Vzdrževanje po naročilu – naročnik ob ugotovljeni napaki preko internega sistema sporoči oz. odda zahtevek, nato se ta naloga dodeli razvijalcu, ta pa potem analizira ter odpravi ugotovljeno napako.

Pri izvajalcu kot najbolj pogoste razloge za vzdrževalne postopke štejemo potrebo po delovanju obstoječe rešitve v novih različicah brskalnikov (Internet Explorer 8, Mozilla Firefox 4).

8.2 Nadaljnji razvoj

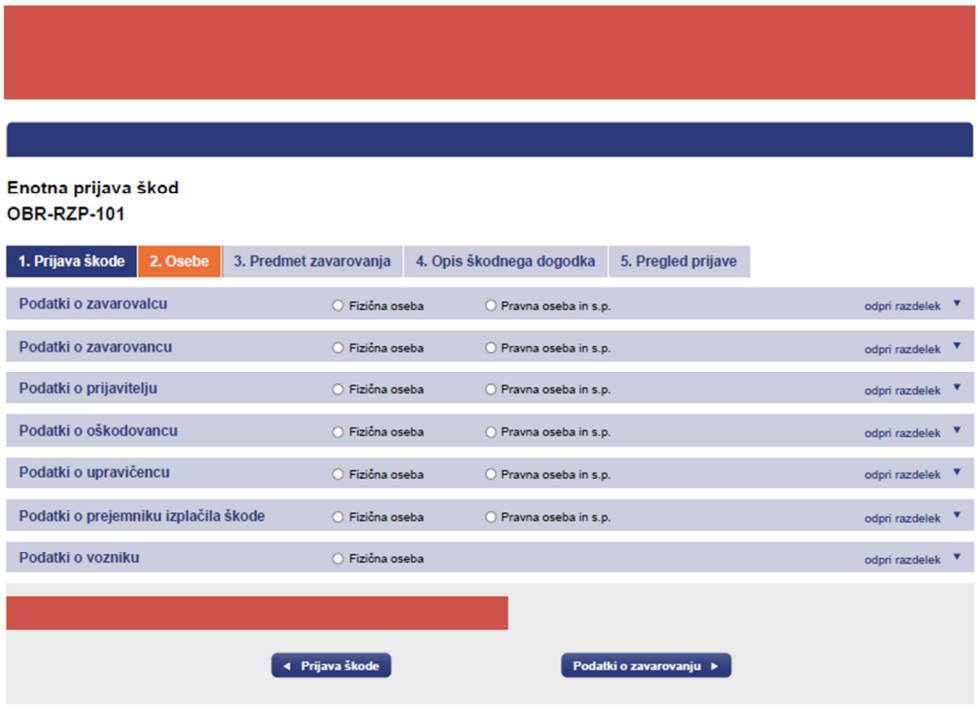
Ker nadgradnja obstoječe rešitve ne šteje kot vzdrževanje, sem jo uvrstil v svoj razdelek, v katerem bom opisal tri nadgradnje spletnega portala, od katerih bo prva kmalu v uporabi, druga je trenutno še v razvoju, tretjo pa bom omenil kot eno izmed možnih nadgradenj rešitve v prihodnosti.

1. Prenovljeni CGP – Zavarovalniška družba se je v začetku leta odločila prenoviti svoje že dokaj zastarelo in tehnološko zaostalo uradno spletno mesto. Ker se del spletnega portala z informativnimi izračuni navezuje na to mesto, je bilo potrebno v sklopu te prenove zagotoviti tudi skladnost dela portala z uradno stranjo. Zaradi tega je del portala dobil popolnoma novo obliko, vključno z barvami in oblikami ikon. Del portala je dobil tudi nekaj tehničnih novosti, saj po novem vključuje enako glavo in nogo kot jo ima uradna stran, poleg tega pa je bila spremenjena tudi navigacija s skupne strani.



Slika 8.1 Konceptni videz skupne strani po prenovi CGP.

2. Uporabnikom spletnega portala bo na voljo tudi tretja storitev, in sicer modul za prijavo škod. Zavarovalnica svojim strankam to storitev na svoji spletni strani sicer že ponuja, vendar se je ob prenovi spletnega mesta odločila za nadgradnjo sistema oddaje škodnih dogodkov. Predvidena je celovita spletna oddaja, uporabnikom pa bodo na voljo tudi vsi obrazci za tiste, ki še vedno raje natisnejo in izpolnijo obrazec ter ga fizično prinesejo na poslovno enoto. Modul je zasnovan na čim bolj enostaven, uporabniku prijazen in razumljiv način. Ker gre za enotno prijavo škod, bodo aplikacijo lahko uporabljali tudi tisti, ki niso zavarovani pri naročniku, so pa bili udeleženi v škodnem dogodku skupaj z naročnikovim zavarovancem. Za uporabo pregleda prijavljenih škodnih dogodkov bo nujno potrebna registracija, saj gre med uporabo za podajanje in pošiljanje podatkov, ki zahtevajo prijavljenega uporabnika. Zaradi tega je bil spremenjen tudi registracijski postopek, ki po novem vključuje tudi sklop, namenjen poslovnim partnerjem zavarovalnice, za lažje dokončanje postopka ter poznejšo uporabo modula.



Slika 8.2 Koncept prijave škodnih dogodkov, ki je še v razvoju, grafično pa je že usklajen s prenovljenim spletnim mestom.

3. Digitalni podpis police – Spletni portal je že od začetka zasnovan tako, da bo v prihodnosti lahko ponudil tudi digitalni podpis police, v modulih, ki bodo ponujali dejansko sklepanje zavarovanj preko interneta. Del, kjer se sklepa zavarovanje za tujino, je prvi na spisku mnogih, ki bi lahko vključevali to funkcionalnost. Za digitalni podpis pa bi uporabnik moral doma imeti posebno tablico, preko katere bi potem podpisal polico.

9 ZAKLJUČNE MISLI IN UGOTOVITVE

Sodobna informacijska infrastruktura in uporaba najnovejših informacijskih tehnologij nam omogočata lažje in hitrejše upravljanje procesov. V diplomski nalogi sem predstavil postopek razvoja nove spletne aplikacije, vključno z vsemi pripravami in dogodki pred pričetkom razvoja ter dejavnostmi, ki se dogajajo med testiranjem in po predaji izdelka naročniku.

Splošno znano je, da se pri izdelavi informacijskih sistemov ali samostojnih aplikacij za znanega naročnika pojavijo težave, ker ta ne zna jasno definirati zahtev. Največkrat se v fazi analize problemskega stanja predstavi samo standardne postopke dela, vsa možna odstopanja pa se pokažejo šele med samim razvojem ali celo testiranjem. Po drugi strani pa tudi na izvajalčevi strani večkrat manjka specifično poznavanje obravnavanega področja, zato so reakcije ob napakah ali napačnih oz. pomanjkljivih informacijah prepozne.

Kljub velikemu vložku časa in sredstev obeh vpletenih strani pa se pozitivne lastnosti že kažejo. Trend uporabe spletnih storitev in elektronskega poslovanja je v velikem porastu in vedno več ljudi bo s takimi malenkostmi poslovalo kar preko spleta. Ker tako poslovanje uporabniki ocenjujejo kot bolj udobno in zanesljivo, je tudi s stališča zavarovalnice težnja po avtomatizaciji postopkov prinesla velike prihranke pri količini dela in porabljenega časa, potrebnega za izvršitev storitve. Iz perspektive izvajalčeve podporne ekipe pa med samim razvojem in predajo ter uporabo ni bilo večjih težav.

Zame, kot razvijalca, je bil ta projekt prvi večji, pri katerem sem sodeloval oz. sem ga razvijal kar sam, od same zasnove, preko izdelave in do predaje naročniku. Projekt mi je bil dodeljen takoj potem, ko sem začel delati v razvojni hiši, kar je pomenilo, da sem se hkrati seznanjal s projektnim delom na splošno, tako kot tudi z novimi tehnologijami, ki jih hiša uporablja, predvsem tistimi, ki jih v času šolanja nismo uspeli spoznati. Novo delovno okolje je torej prineslo kar nekaj novih načinov komunikacije, dokumentiranja in veliko različnih poslovnih praks ter sestankov z naročniki. Izdelava aplikacije je večinoma potekala gladko in brez večjih problemov, začetne težave, ki so bile bolj ali manj posledica privajanja na novo delovno mesto, pa smo na prijateljski način rešili znotraj delovnega kolektiva.

KAZALO SLIK

LITERATURA IN VIRI

[1] – (2010) Dejavniki tveganja razvoja. Dostopno na: http://www.ipmit.si/IPMITstrani/ipmitslo.nsf/V/KCD6F54A791F3DB41C12572C00023AAD4/$file/Laura_Knafeljc_Kozman_Dejavniki_tveganja_razvoja.pdf

[2] – (2011) Funkcionalna specifikacija. Dostopno na: http://en.wikipedia.org/wiki/Functional_specification

[3] – Rene J. Chevance, Server architectures, Burlington, Elsevier digital press, 2005

[4] – Kiran Mani, On the edge: A comprehensive guide to blade server technology, Singapore, John Wiley and Sons, 2007

[5] – (2011) Splošno o RAID poljih. Dostopno na: http://en.wikipedia.org/wiki/RAID

[6] – (2011) Standardno povezovanje v RAID polja. Dostopno na: http://en.wikipedia.org/wiki/Standard_RAID_levels

[7] – (2011) Gnezdeno povezovanje v RAID polja. Dostopno na: http://en.wikipedia.org/wiki/Nested_RAID_levels#RAID_10_.28RAID_1.2B0.29

[8] – (2011) Virtualizacija. Dostopno na: http://www.vmware.com/virtualization/

[9] – (2011) .NET Framework. Dostopno na: http://en.wikipedia.org/wiki/.NET_Framework

[10] – (2011) Spletne storitve. Dostopno na: http://en.wikipedia.org/wiki/Web_service