Sveučilište Jurja Dobrile u Puli

Odjel za informacijsko-komunikacijske tehnologije

**CooKING**

Programsko rješenje

Pula, siječanj 2018. godine

Sadržaj

1. [Uvod](#_Toc505091767) 3

2. [Motivacija](#_Toc505091768) 4

2.1. [SWOT analiza](#_Toc505091769) 5

2.2. [Buduća poboljšanja](#_Toc505091770) 6

3. [Razrada funkcionalnosti](#_Toc505091768) 7

3.1. [Use-Case dijagram](#_Toc505091769) 7

3.2. Dijagram slijeda 9

3.3. [Relacijski model](#_Toc505091769) 12

3.4. Klasni dijagram14

3.5. [Prototip sučelja](#_Toc505091769) 16

4. Implementacija 20

4.1. [Povezanost grafičkog sučelja aplikacije](#_Toc505091769) 20

4.2. Klase 22

4. Korisničke upute 26

# 1. Uvod

CooKING aplikacija se temelji na ideji kuhaj s onime šta imaš. Korisnici će sami unositi namirnice koje su im trenutno pri ruci te bi im aplikacija ponudila sve moguće recepte za koje imaju potrebne namirnice. Unutar aplikacije ima ponuđenih opcija s kojima bi korisnik mogao blago presonalizirati aplikaciju, u početnoj verziji (promjena recepta, promjena vremena kod izvršavanja recepta...). U početnom izdanju aplikacije ne bi bio moguć login sa svojim korisničkim imenom i lozinkom, tako da korisnici neće biti u stanju pratiti svoj progresni stadij odnosno koliko su kuhali, koje vrste jela su kuhali, na kojoj razini su trenutno itd. To sve bi također bilo pridodano u novijoj verziji aplikacije, čime bi pobudili natjecateljski duh u korisnicima i želju za postizanjem ciljeva. S obzirom da su mlađe populacije sve manje zainteresirane za samo kuhanje, a više se oslanjaju na brzu i nezdravu prehranu mislimo da bi ova aplikacija mogla promijeniti taj trend i probuditi želju u njima za domaćom kuhinjom kroz vizualno ugodan i organiziran sadržaj. Pošto danas nisu samo studenti odnosno mlađe generacije financijski ograničene kod kuhanja, aplikacija bi bila namijenjena svima zainteresiranima. Sam izgled i funkcionalnosti su jednostavno implementirane što bi olakšavalo korištenje aplikacije ljudima starije dobi. Nadamo se da bi korisnici kroz ovu aplikaciju postali više zainteresirani za kuhanje te da bi predrasude kako je kuhanje naporno nestale. Također aplikacija u početnoj verziji sadrži u sebi mini-game koji bi u trenu kada korisnik pomisli da je kuhanje dosadno odagnao tu misao i na taj način bi korisniku povezali kuhanje sa zabavom.

# 2. Motivacija

Unutar ciljanog tržišta glavni fokus bi nam bio na mlađe populacije koje su danas nerazdvojni od svojih pametnih uređaja, što bi automatski značilo da će oni lakše pronaći i upravljati našom aplikacijom. Ne bi isključili u potpunosti niti ostatak populacije pošto su i oni prisiljeni na modernizaciju, a imaju veću tendenciju za kuhanjem nego današnja mladež. Svejedno uzeći u obzir faktor dobi smatramo da mladi danas provode više vremena na internetu, a u kasnijem razvoju naše aplikacije povezanost sa internetom bi bila nužna što znači da smo na neki način i primorani gledati na njih kao na primarnu dobnu skupinu ako želimo postići neki uspjeh s ovom aplikacijom. Ova aplikacija ne bi bila prva takve vrste na tržištu. Postoji već nekoliko aplikacija koju su bazirane na sličnoj ideji poput naše, no većina njih je za strane korisnike tj. na stranim jezicima među kojima je naravno najzastupljeniji engleski. Naša aplikacija je na hrvatskom jeziku, te nam je ciljano tržište Hrvatska zajedno sa ostalim zemljama u regiji. Jedna od prednosti je što nema puno aplikacija takvog tipa na hrvatskom jeziku i one koje postoje se razlikuju od naše po principu pretrage recepata. Ponuđene aplikacije sadrže recepte koje se mogu pretraživati po namirnicama, težini pripreme i vrsti jela, ali nijedna nema opciju poput naše da korisnik unese sve namirnice koje su mu na raspolaganju i kao rezultat aplikacija mu prikaže sva jela koja sadrže sve ili neke od tih namirnica. Dosadašnji razvoj aplikacije se odvijao u kratkom vremenskom razdoblju od dva mjeseca. Kao prvo smo napravili sve potrebne dijagrame (Use-case, sequence i class) kako bi vidjeli šta sami želimo od aplikacije i koja bi bila njena ograničenja, nakon određenih prilagodbi koje su nam predstavljale probleme u razvoju krenuli smo sa pisanjem koda. Korisničko sučelje aplikacije smo napravili unutar Microsoft Visual Studia, te imamo bazu podataka koja je napravljena u Microsoft SQL Servera i koja komunicira sa aplikacijom. Ukoliko ima zainteresiranih za aplikaciju sve što treba imati je SQL server za bazu podataka i mora imati instaliran workload NET Framework 4 unutar Visual Studia.

## 2.1. SWOT analiza

|  |  |  |
| --- | --- | --- |
|  | **Pozitivno** | **Negativno** |
| **Unutarnje** | * Pretraga recepata prema namirnicama koje su nam pri ruci * Poticanje ljudi na zdraviju prehranu * Aplikacija je na hrvatskom jeziku, čime osiguravamo da korisnik ne mora posezati za sličnim aplikacijama na stranim jezicima * Specifičan dizajn unutar aplikacije, te zabavni kutak za korisnike * Ostavljeno puno prostora za daljnje napredovanje aplikacije kroz već zapisane ali ne implementirane ideje | * Velika konkurencija na tržištu iz svih zemalja regije * Korisnici u trenutnom izdanju ne mogu pratiti svoj napredak unutar aplikacije * Aplikacija nije povezana na internet, te nije moguće dijeljenje aplikacije na društvene mreže |
| **Vanjsko** | * Suradnja sa proizvođačima ili trgovačkim lancima čije bi se namirnice ili začini nalazili u receptima * S prevođenjem aplikacije na strane jezike bi pokušali privući i strane korisnike, te moguće suradnike * Mogućnost sponzoriranja kulinarskih emisija na tv-u i radiju u svrhu reklamiranja naše aplikacije | * Konkurencija prilagodi svoj način pretraživanja našem * Pojave se nove i modernije aplikacije ovog tipa, uvođenje AI * Aplikacija ne ostvaruje rezultate kakve smo očekivali i za nju nema interesa |

Snage nadvladavaju slabosti, prilike nadvladaju prijetnje – podržava strategiju rasta.

## 2.2. Buduća poboljšanja

U novijim verzijama aplikacije bi dodavali nove funkcionalnosti koje bi se odnosile na poboljšanje korisničkog iskustva na aplikaciji, rješavanje bugova prema korisničkim pritužbama i vlastitom analizom aplikacije. Već spomenuto poboljšanje dodavanja korisničkih imena i lozinki bi omogućilo da svaki korisnik može pratit svoj napredak unutar aplikacije. Na Korisničkom profilu bi se nalazio progres bar na kojem bi bilo vidljivo iskustvo koje je on prikupio prilikom izvršavanja recepata( što bi također ovisilo o težini jela, vrsti jela, vremenu pripreme...), dodjeljivanje bedževa bi funkcioniralo da korisnik dobijem određeni bedž ukoliko često kuha jednu vrstu hrane na primjer korisnik stalno kuha povrće te bi mu bio dodijeljen bedž „zdrave prehrane“ , ako često priprema brzu hranu dodijeljen bi mu bio bedž „junk addict“ itd. Povezali bi aplikaciju sa internetom da možemo pružiti opciju dijeljenja svojih uspjeha sa svojim prijateljima na društvenim mrežama kako bi uključili kompetetivnu komponentu koja bi osigurala veću zanimljivost korisnika i produžila dugotrajnost korištenja aplikacije odnosno čim dulje korisnik aktivno koristi aplikaciju manje su šanse da će odustati od svojih postignuća i od same aplikacije. Kod recepata bi uveli opciju printanja recepata kako bi korisnici mogli imati svoje najdraže recepte u fizičkom obliku i ukoliko ih žele podijeliti sa ljudima koji ne koriste našu aplikaciju, da može uploadati svoje slike kod recepata i da može dodavati svoje bilješke za recept. I neko krajnje poboljšanje koje nam je u ovom trenu zamislivo bi bila da ubacimo statistiku koja bi računala prema podacima svih korisnika najpopularnije recepte. I kasnije ukoliko bi bilo isplativo napravili bi AI koji ispravlja pogreške. Recimo dodao sam 4 žličice soli umjesto 1, a aplikacija zaključi da onda za ispravak okusa treba dodati još vode i brašna.

**4. Razrada funkcionalnosti**

U ovom odjeljku razrađujemo funkcionalnosti aplikacije CooKING. Najprije ćemo objasniti kako funkcionira naša aplikacija na use-case dijagramu i na koji način se odvija interakcija korisnika i aplikacije. Zatim ćemo proći kroz dijagrame slijeda na kojima ćemo objasniti pojedine funkcije ove aplikacije detaljnije. Nakon toga pojasniti će se priloženi prototip aplikacije po kojem je građen dizajn i koji je poslužio kao temelj ideja aplikacije. Na kraju ćemo prikazati klasni dijagram aplikacije te objasniti veze između klasa te relacijski dijagram izveden napravljen po slici baze podataka ove aplikacije.

**4.1. Use-Case dijagram**

Sljedeća slika prikazuje Use-case dijagram naše aplikacije. On je bio prvi izrađen kada smo počeli izrađivati dijagrame i na njemu se temelje svi ostali dijelovi aplikacije. Na dijagramu su u grubo prikazane funkcionalnosti od kojih se aplikacija sastoji.

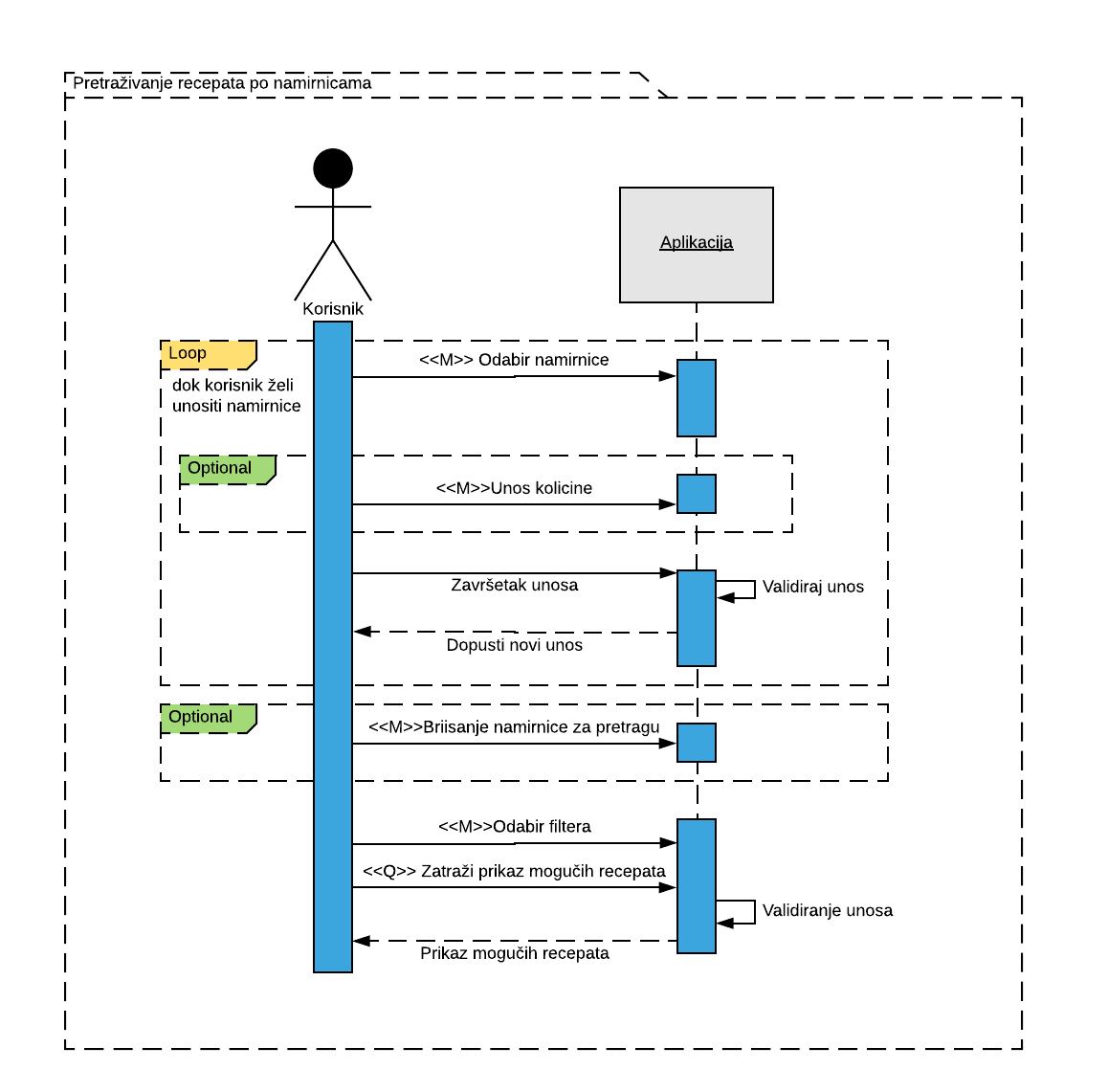


*Slika 1. Use-Case dijagram*

Naša se aplikacija sastoji od 5 temeljnih funkcionalnosti koje se kasnije granaju odnosno proširuju te temeljne funkcionalnosti. Za početak možemo pozornost obratiti na funkcionalnost pod imenom „*Dodavanje podataka*“. U aplikaciji korisniku je omogućeno da dodaje recepte i namirnice po svojem izboru, odnosno da po želji unosi podatke o novom receptu koji uključuju unos sastojaka i koraka te po izboru dodavanje fotografija, te unos novih namirnica od kojih se formiraju sastojci(u paru sa količinom koja ide u recept). Korisnik može dodavati namirnice u procesu stvaranja novog recepta ili može dodavati nove namirnice neovisno o tome je li u procesu dodavanja recepta ili nije. Zatim možemo razmotriti funkcionalnosti „*Prikaz Recepta*“ i „*Prikaz namirnica*“. Te funkcionalnosti omogućuju korisniku da pregledava namirnice koje postoje u bazi podataka i koje se mogu iskoristiti u receptima te također omogućava pregled svih postojećih recepata koji se mogu pripremati. Pri pregledu pojedinog recepta korisnik može vidjeti koji se sastojci nalaze u tom receptu, koji koraci, te ostale informacije o receptu kao što su vrijeme kuhanja, vrsta jela itd. Ako se odlučimo obrisati recept ili namirnicu to možemo učiniti iz prikaza recepata ili namirnica. Nadalje imamo glavnu funkcionalnost naše aplikacije a to je *„Pretraživanje recepata po namirnicama“.* Ova funkcionalnost omogućava da korisnik upisuje namirnice po njihovom imenu i količini od kojih će se formirati oni recepti koji koriste te namirnice(koje je moguće napraviti od tih sastojaka koje je korisnik upisao). Pod *„Filtriranje recepata“* se smatra da korisnik može filtrirati rezultate recepata po kategorijama kao što su vrsta jela, težina pripreme itd. Nakon što se prikažu rezultati pretrage recepata korisnik može *„kuhati uz navođenje“,* odnosno može pratiti svoj progres kroz recept dok kuha, uključivati tajmere koji odbrojavaju vrijeme, pratiti cjelokupno vrijeme kuhanja itd. Posljednje imamo funkcionalnost *„igranje mini-game“.* Ova funkcionalnost omogućava da korisnik u procesu kuhanja(ili van tog procesa) igra mini-game memori igru. Ta igra se temelji na spajanju istih sličica u polju 4x4 i služi primarno za zabavu prilikom primjerice čekanja da tajmer odbroji svoje vrijeme.

**4.2. Dijagrami slijeda**

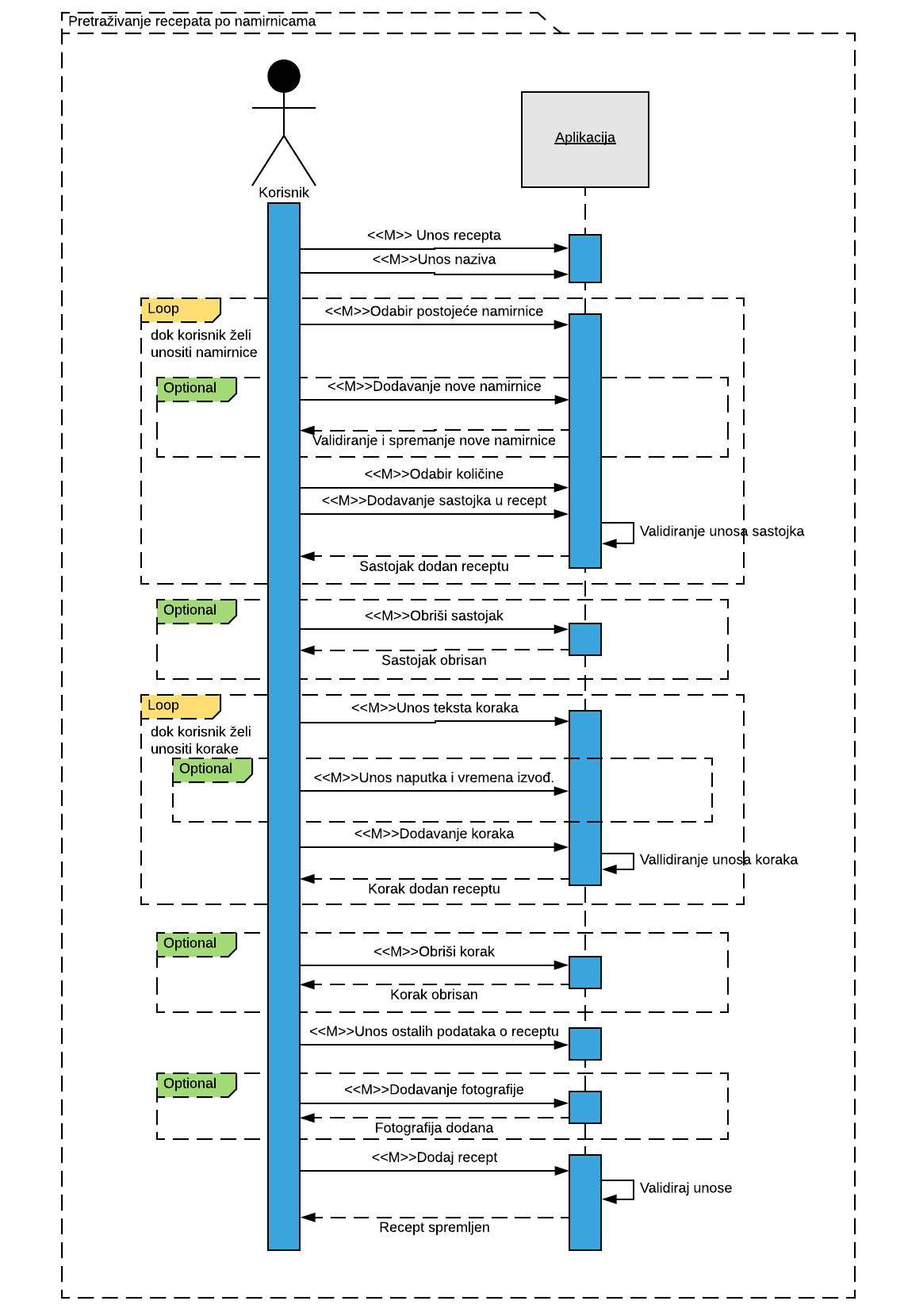
Slijedi nekoliko dijagrama slijeda koji opisuju bitnije procese ove aplikacije. Prvo imamo dijagram koji opisuje kako se odvija dodavanje recepta u detaljnijem prikazu interakcije korisnika i aplikacije.



*Slika 2. Dijagram slijeda – Pretraživanje recepata po namirnicama*

Dijagram na slici 2 prikazuje dijagram slijeda procesa pretraživanja recepata po namirnicama. Iz ovog dijagrama vidljivo je da korisnik u procesu pretrage namirnica može unositi koliko god namirnica želi u pretragu s time da je unos količine opcionalan. Opcionalan je iz razloga što u trenutku odabira filtera korisnik može odabrati da se zanemare mjere i količine pri filtriranju recepata. Nakon što korisnik odabere namirnicu program validira unesene podatke i ako su točno uneseni pušta unos duge namirnice a ovu unesenu stavlja u listu namirnica za pretraživanje recepata. Također jednom kada se namirnica stavi u listu namirnica za pretragu ona se može i obrisati iz te liste. Nakon što se odaberu namirnice po kojima će se vršiti filtriranje recepata odabiru se i filteri. Postoji nekoliko različitih filtera po kojima korisnik može filtrirati rezultate primjerice odabir težine, vrste i podvrste te biranje hoće li se pri izvlačenju odgovarajućih recepata u obzir uzeti mjere i količine. Nakon što korisnik odabere filtere aplikacija od svih recepata izabire samo one koji odgovaraju filterima a onda ovisno o tome da li je filter za količine i mjere uključen ili ne prikazuje samo recepte koji imaju barem 1 od navedenih namirnica u sebi. Ako imamo sve sastojke recepta ti recepti će se prikazivati na vrhu liste a ispod njih slijede recepti kojima nedostaju namirnice.

Sljedeći dijagram je dijagram slijeda unosa novog recepta. Ovo je bitan korak jer se u njemu povezuju sve relacije u bazama podataka i sve klase ovog programa.

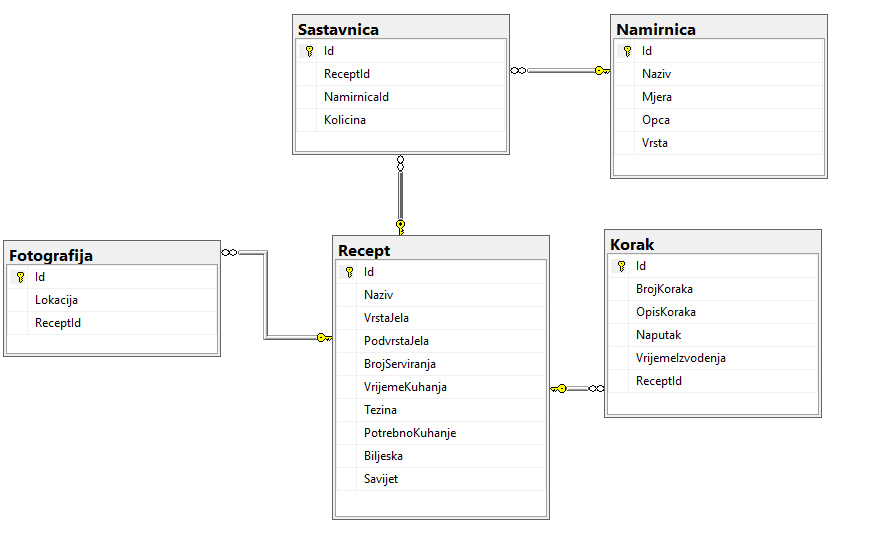


*Slika 3. Dijagram slijeda – Dodavanje recepta*

Dijagram sa slike 3 prikazuje interakciju aplikacije i korisnika u procesu dodavanja novog recepta u program. Kada korisnik klikne na dodavanje novog recepta otvara se novi prozor u kojem se upisuju podaci o receptu. Prvo se upisuje ime recepta a zatim se dodaju namirnice u recept. Korisniku se prikazuju sve postojeće namirnice i on od njih bira one koje će staviti u recept. Naravno uz to mora napisati i količinu namirnice koja ide u recept. Mjeru za namirnicu ne dodaje korisnik nego je ona fiksna i odabire se onda kada se dodaje nova namirnica. Razlog tomu je što pri pretrazi baze ako je uključen filter količina i mjera onda uspoređujemo količinu namirnice koju je unio korisnik s količinom namirnica u receptima u bazi. Ako je mjerna jedinica različita onda se rezultati neće točno prikazati. Kao poboljšanje aplikacije jednoga dana bi bilo dobro uvesti pretvarač jedinica.  
Kada se klikne na dodavanje namirnica validira se njena količina i dodaje u listu sastavnica. Ako u bilo kojem trenutku poželimo obrisati sastavnice to možemo učiniti. Sličan postupak je i sa dodavanjem koraka u recept. Korisnik upisuje tekst koraka recepta i dodatno uz taj korak može(ako želi) dodati naputak za izvođenje i vrijeme izvođenja koraka(od njega se formira tajmer). Koraci se u svakom trenutku mogu brisati. Prati se redoslijed koraka počevši od 1 pa na dalje. Nakon toga korisnik unosi ostale podatke o receptu a oni su vrijeme pripreme, broj serviranja, vrsta i podvrsta, da li je potrebno kuhanje itd. Na temelju tih informacija se izvodi filtriranje recepata kasnije. Isto tako korisnik može svom receptu dodati fotografiju sa računala. Nakon što korisnik klikne na spremi recept aplikacija mora validirati sve unose i uvjeriti se da recept ima barem jedan korak i barem jedan sastojak jer bez toga se ne može kreirati novi recept. Isto tako ako ime recepta već postoji u bazi ne može se kreirati novi recept.

**4.3.** **Relacijski model**

Sljedeći model prikazuje kako je naša aplikacija povezana u bazi podataka. Prikazani su primarni ključevi svake relacije i svi atributi koji su vezani za entitete. Ono što sve relacije imaju zajedničko je to da sve imaju kao primarni ključ Id koji se automatski sekvencijalno generira pri svakom dodavanju retka u tablice. Na taj način smo spriječili situacije u kojima bismo morali paziti koje primarne ključeve mi generiramo trigerima i izbjegli potencijalnu prijetnju od remečenja (violation) primarnog ključa.



*Slika 4. Relacijski model*

Strani ključevi nisu vidljivi na ovom modelu ali se mogu zaključiti iz poveznica između relacija. Primjerice vidimo da fotografija u sebi sadržava svoj Id, lokaciju gdje se ona nalazi u računalu i sadržava ReceptId koji je po svojoj prilici strani ključ koji potiče iz relacije Recept. To znači da svaka fotografija ima u sebi id recepta kojemu ona pripada. Isto vrijedi i za korake. Što se tiče Sastavnice ona se sastoji od 2 strana ključa-od id namirnice i od id recepta. To je potrebno iz razloga što bismo ovdje gledali na vezu više-na-više da nismo uveli relaciju sastavnica.

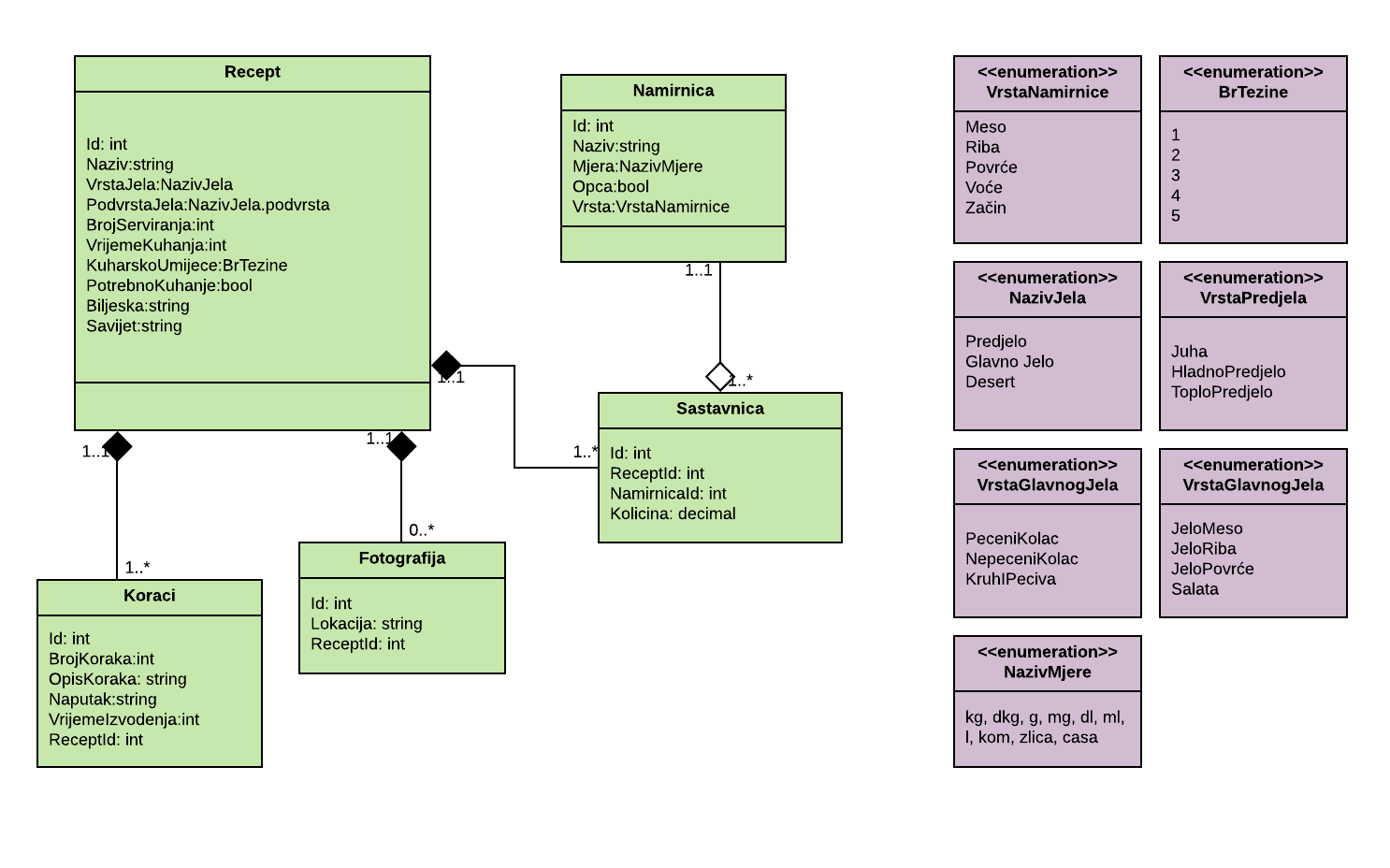
Svaka relacija u ovom modelu odgovara klasi u klasnom dijagramu. No razlike postoje u primjerice načinu na koji recept sadržava korake u bazi od načina na koji sadržava korake u programu. U bazi svaki korak ima id recepta kojem on pripada. U programu bismo to isto tako mogli napraviti ali onda bismo morali pri svakom čitanju koraka recepta prolaziti kroz sve korake svih recepta i po id-u recepta tražiti korake koji pripadaju receptu. Zato smo to u programu riješili tako da svaki recept ima listu tipa Korak koja sadržava sve korake koji pripadaju tom receptu. Isto je s namirnicama i sa fotografijama.

Još jedna stavka na koju je bitno obratiti pozornost je ta da se pri brisanju recepta kaskadno brišu svi koraci, sve fotografije i sve sastavnice koje pripadaju tom receptu ali ne brišu se namirnice.

**4.4 Klasni dijagram**

Klasni dijagram opisuje način na koji su klase povezane u našem programu. Kao što je prije navedeno postoje razlike između relacijskog modela i klasnog dijagrama iako na prvi pogled to možda ne izgleda tako. U ovom dijagramu vidimo da se atributi klasa također poklapaju sa atributima relacija. Sve klase imaju Id-eve koji se povlače iz baze i po njima se može pretraživati jednako točno kao i u bazama podataka. Jednom kada iz baze povučemo sve podatke više ništa ne moramo iz baze čitati. Novi podaci se jednostavno stavljaju u liste podataka koji su pročitani iz baze. Radi lakšeg komuniciranja s bazom putem dodatka pod nazivom Dapper, isti nazivi atributa su u klasama kao i u relacijama.

Ovo je klasni dijagram aplikacije:



*Slika 5. Klasni dijagram*

Ako počnemo gledati od recepta vidimo da se sve klase povezuju sa tom klasom. Oko klase recept se okreće cijela aplikacija počevši od dodavanja recepata pa sve do praćenja koraka u kuhanju. Skoro svi procesi se baziraju na podacima o receptu. Sada se možemo osvrnuti na odnose između relacija u relacijskom modelu gdje se redci relacije Fotografija, Korak i Sastavnica kaskadno brišu ako se obriše stavka iz relacije Recept čiji se Id nalazi kao atribut jedne od te tri relacije. Naš klasni dijagram izgleda gotovo isto kao i relacijski model ali sa tom razlikom da u programu svaki objekt tipa Recept ima 3 liste: Fotografije, Koraci i Namirnice. Te tri liste služe za prikazivanje podataka na ekranu korisnika i oslobađaju nas stalnog kopanja po bazi podataka kako bi izvukli dodatne informacije o receptu. Svaki korak kao i svaka fotografija sadrži IdRecepta i iako se možda čini nepotrebno da imaju taj atribut(pošto objekt Recept sadrži liste samo svojih koraka i fotografija), to nam je svakako korisno jer u trenucima kada odvajamo recepte od koraka ili fotografija uvijek znamo kojem receptu ta fotografija pripada. Nadalje pitamo se zašto se u receptu nalazi lista namirnica a ne lista sastavnica. Kada bi se nalazila lista sastavnica onda bismo imali samo id namirnice koju on sadrži plus količinu. Ovako imamo sve podatke o toj namirnici koje možemo iskoristiti na način da pratimo koje je ona vrste i je li must-have ili ne, ali uz nedostatak količine koja ide u recept. Prednost je također ta što kod pretraživanja recepata po namirnicama ne moramo za svaki recept prolaziti kroz sastavnice i onda kroz sve namirnica kako bi ustanovili da li se određena namirnica nalazi u receptu nego moramo samo pogledati listu namirnica(nazive) koji se nalaze u listi namirnica koje pripadaju receptu. Sljedeće možemo promotriti vrste veza između klasa. Kompoziciju primjećujemo na tri mjesta: vezanje koraka za recept, vezanje fotografije za recept i vezanje sastavnice za recept. Ta veza označava da se pri brisanju recepta brišu i svi koraci i fotografije koje se nalaze u tom receptu(u programu). Međutim pri brisanju recepta ne brišu se njemu pripadne namirnice jer one mogu biti sastavnice nekog drugog recepta. Kardinalnosti veze su same po sebi jasne. Naša aplikacija koristi i nekoliko enumeracija koje služe ponajprije lakšem filtriranju i pronalaženju podataka no i kako bi se recepti logički odvojili po određenim principima.

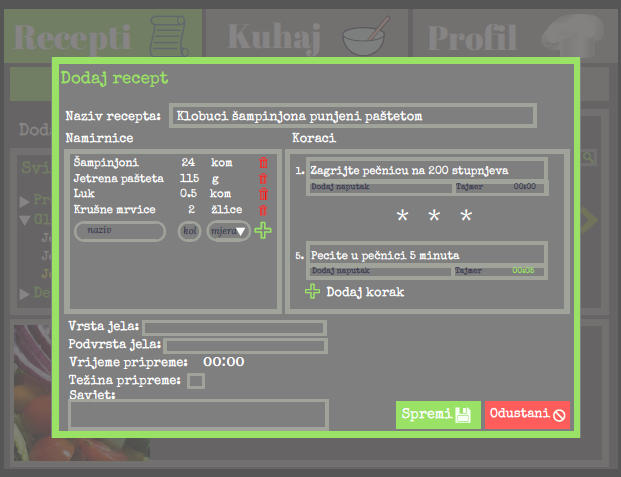
**4.5. Prototip sučelja**

Ideja za dizajn aplikacije potekla je iz prototipa sučelja međutim prototip je dao ideje i za funkcionalnosti aplikacije i uvelike ubrzao pisanje koda. Na sljedećim slikama prikazan je prototip sučelja odnosno prvotna ideja kako bi aplikacija trebala izgledati i kakve bi funkcionalnosti ona trebala imati.



*Slika 6. Prototip „Svi Recepti“*

Gornja slika prikazuje početni prozor aplikacije pri pokretanju. Gotova aplikacija izgleda uvelike kao i prototip uz manje izmjene. Aplikacija se doima moderno iz razloga što se koristio *flat-style* princip koji se bazira na kreiranju prozora, gumbova i ostalih widgeta na način da se oni potpuno personaliziraju i odvoje od postavljenog microsoft windows izgleda prozora. Uklonili smo okvire prozora što nam je ostavilo goli prozor bez gumba, na koji smo dodavali prilagođene widgete. Na ovom prozoru su se koristili različiti widgeti od picturebox-eva koji drže slike recepata preko tree-view-a koji sadrži recepte po kategorijama do list-view-a koji sadrži korake recepta i prikazuje njihove detalje.



*Slika 7. Prototip – Dodavanje recepta*

Prototip dodavanje recepta je jako sličan prozoru za dodavanje recepta koji je trenutno u aplikaciji.



*Slika 8. Prototip – Praćenje kuhanja*

Prozor koji odgovara slici broj 8 je malo više izmijenjen nego ostali prozori s obzirom na izgled prototipa. U praćenje kuhanja ubačena je cirkularna traka za napredak te su ostali widgeti razmješteni drugačije.



*Slika 10. Prototip – Pretraga recepata po namirnicama*

Slika 10 prikazuje prototip prozora za dodavanje recepata po namirnicama, koji je vrlo sličan završnom prozoru i uključuje sve funkcionalnosti koje se mogu vidjeti na ovom prozoru.



*Slika 9. Prototip – Rezultati pretrage po namirnicama*

Ono što je u slici 9 različito od stvarnog prozora jest prikaz rezultata. Ovdje se vidi da su rezultati prikazani kao sličice sa informacijama o rezultatima pretrage a pritiskom na sliku dobijemo detalje recepta. U aplikaciji rezultati su prikazani u list-boxu poredani prvo oni koji imaju sve sastojke zatim oni kojima sastojci nedostaju.

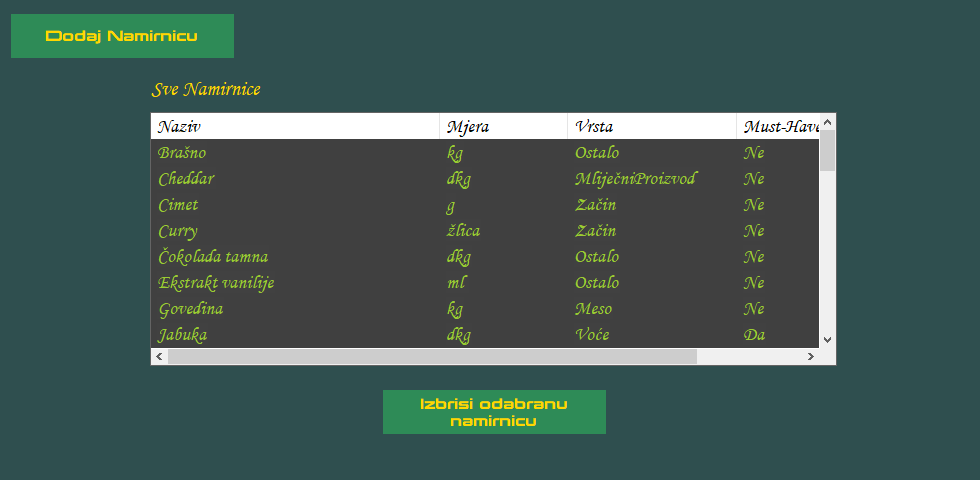
**5. Implementacija**

U ovom odjeljku opisati ćemo kako smo implementirali ovu aplikaciju kroz kod. Prvo ćemo se osvrnuti na načine na koji su prozori aplikacije povezani, zatim objasniti osnovne klase čijim objektima metode aplikacije upravljaju, te opisati pomoćne klase. Opisati ćemo izradu grafičkog sučelja te na posljetku objasniti kako funkcionira povezanost s bazom podataka.

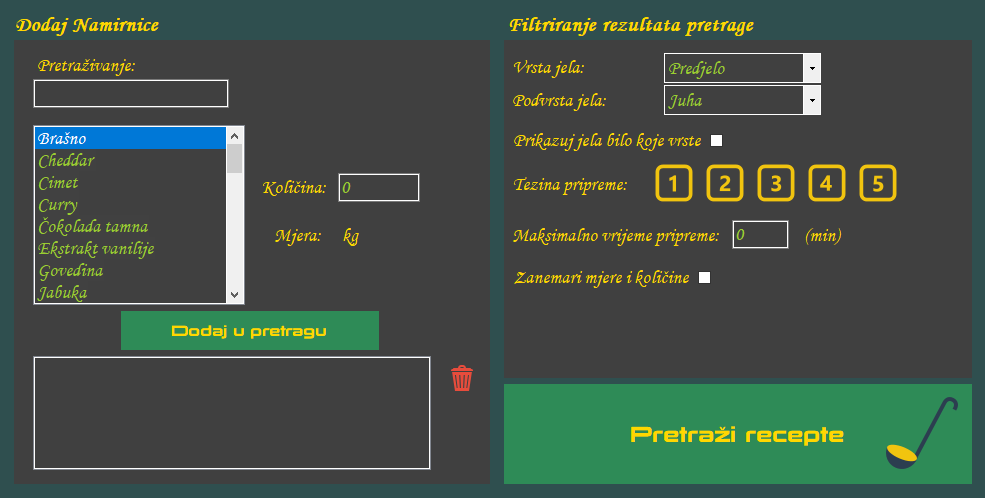
**5.1. Povezanost prozora aplikacije**

Naša aplikacija sastoji se od glavnog prozora i od ostalih prozora. Glavni prozor sadržava *User Control* komponente koje primarno služe za grupiranje widgeta u cjeline za koje se onda smatra da su jedna jedinstvena komponenta. Ovim načinom stvorili smo dojam da se sve odvija na jednom prozoru aplikacije što i je točno ali ne u cijelosti. Kada se klikne na gumbe koji su umetnuti u glavni prozor(Recepti, Kuhaj, Minigame..) eventi koji se onda okinu jednostavno pozivaju određeni user control na površinu prozora. Naša aplikacija se sastoji od četiri user control-a a to su: UCNamirnice(koji sadržava widgete kojima je funkcija prikaz svih postojećih namirnica iz baze te koji omogućuju dodavanje i brisanje namirnica), UCSviRecepti(koji omogućava pregled svih recepata, dodavanje recepta i brisanje recepta), UCKuhajPretrazivanjePoNamirnicama(koji korisniku omogućava da unosi namirnice po kojima želi da se recepti prikazuju te filtriranje recepta) te UCKuhajRezultatiPretrage(koji prikazuje rezultate pretrage po namirnicama. Kada kliknemo na bilo koji gumb na glavnom prozoru koji sadržava te user control-ove određeni user control se postavi na površinu prozora metodom *BringToFront().* Inače svi su oni naslagani jedan iza drugog na istom prozoru i čekaju da se pozovu. Ta metoda nam omogućava da sve te komponente budu aktivne u svakom trenutku(za razliku od prozora koji se otvaraju i zatvaraju po potrebi) što nam omogućava da se progres ostvaren na bilo kojoj komponenti ne briše ako se neka druga komponenta izvuče na površinu prozora. Tako primjerice ako smo unijeli neke namirnice za pretragu, namjestili filtere pretrage i kliknuli na *Pretraži*, UCKuhajRezultatiPretrage će se popeti na vrh i prikazati rezultate ali sve sa prethodnog UC-a će ostati nepromijenjeno odnosno kada kliknemo *Natrag* iz UC-a gdje se prikazuju rezultati ponovno ćemo doći na isti prozor gdje smo prvotno upisivali namirnice po kojima pretražujemo recepte te vidjeti da su te namirnice još uvijek tamo i da su filteri u stanju kakvom smo ih ostavili. To omogućava brži prolazak kroz prozore i štedi korisnika ponovnom upisivanju filtera i namirnica za pretragu. Isto funkcionira i s ostalim dijelovima aplikacije.

Svaki UC se ponaša kao posebni prozor što znači da on ima poseban Dizajner i poseban dio za pisanje koda koji se odnosi na widgete koji su u njemu. To znači da bilo koji atribut(lista ili varijabla) koji se nalazi u tom UC-u se ne vidi u niti jednom drugom UC-u, čak i ako je varijabla javna. Sve liste koje su nam krucijalne za funkcioniranje aplikacije moraju biti vidljive i na ostalim UC-ovima a to smo postigli na način da smo u UC kojem su potrebne varijable nekog drugog UC-a postavili varijablu koja će sadržavati referencu na taj UC. Tako primjerice UCKuhajPretrazivanjePoNamirnicama za svoje funkcioniranje treba listu svih recepata koja se nalazi u UCSviRecepti. S obzirom da on ne vidi tu listu morali smo omogućiti da se pri inicijalizaciji UCKuhajPretrazivanjePoNamirnicama kao njegov partner postavi UCSviRecepti. Na taj način mi možemo pristupiti listi SviRecepti(partner.SviRecepti) i koristiti tu listu. Ovo je također vrlo korisno jer mi imamo prividne prozore što znači da su nam svi prozori(UC-ovi) ubiti otvoreni u isto vrijeme. Kada dodajemo primjerice novu namirnicu u UCNamirnice mi moramo osvježiti listu namirnica automatski i u UCPretrazivanjePoNamirnicama. To je omogućeno partnerstvom pri čemu se nakon dodavanja nove namirnice automatski poziva funkcija koja osvježava listu namirnica i u tom UC-u. Sljedeće slike prikazuju 2 UC-a.



*Slika 10. UCNamirnice*



*Slika 11. UCPretrazivnjePoNamirnicama*

**5.2. Klase**

Ova aplikacija se bazira na klasama koje se podudaraju s bazama podataka te klasama koje služe kao pomoćne da bi se podaci prezentirali korisniku,

Slijedi tablica glavnih klasa i njihovih sastavnica kao i funkcionalnosti koje one omogućuju.

|  |  |  |
| --- | --- | --- |
| Naziv klase | Sastavnice | Funkcionalnosti |
| Fotografija.cs | public string Id  public string Lokacija  public int ReceptId | * Služi za punjenje liste fotografija u objektu tipa Recept |
| Koraci.cs | public int Id  public int BrojKoraka  public string OpisKoraka  public string Naputak  public int VrijemeIzvodenja  public int ReceptId | * Služi za punjenje liste koraka u objektu tipa Recept |
| Namirnice.cs | public int Id  public string Naziv  public Enumeracije.ENazivMjere Mjera  public bool Opca  public Enumeracije.EVrstaNamirnice Vrsta | * Služi za punjenje liste namirnica u objektu tipa Recept * Koristi se u listi za držanje svih namirnica iz baze podataka |
| Recept.cs | public int Id  public string Naziv  public Enumeracije.EVrstaJela VrstaJela  public int PodvrstaJela  public int BrojServiranja  public int VrijemeKuhanja  public Enumeracije.EBrTezine Tezina  public bool PotrebnoKuhanje  public string Biljeska  public string Savijet  public List<Koraci> KoraciRecepta  public List<Namirnice> NamirniceRecepta  public List<Fotografija> FotografijeRecepta | * Koristi se u listi svih recepata koja drži sve recepte iz baze. * Koristi se u listi koja drži recepte koji odgovaraju pretrazi |
| Sastavnica.cs | public int Id  public int ReceptId  public int NamirnicaId  public decimal Kolicina | * Lista koja drži sve sastavnice iz baze |

*Tablica 1. Glavne klase aplikacije*

Gore navedene klase su klase koje omogućuju ponajprije čuvanje podataka koje dohvaćamo iz baze. Nakon što podatke dohvatimo iz baze i pohranimo u odgovarajuće objekte, moramo ih nekako prikazati korisniku. Ponekad je dovoljno izvući podatke iz objekata glavnih klasa no međutim nekada nam je potreban objekt koji će se sastojati od podataka iz više od jednog objekta glavnih klasa. Za tu svrhu uvodimo pomoćne klase prikazane u sljedećoj tablici:

|  |  |  |
| --- | --- | --- |
| Naziv klase | Sastavnice | Funkcionalnosti |
| PomocnaNamirnice.cs | public string Naziv  public decimal Kolicina  public Enumeracije.ENazivMjere Mjera  public string Display {get {  return string.Format(„{0} {1} {2}“, Naziv, Kolicina, Mjera); } | * Ova klasa nam omogućava da prikazujemo korisniku sastojke recepta tako da uzima naziv i mjeru iz objekta tipa Namirnica a količinu iz objekta tipa Sastavnice |
| Enumeracije.cs | public enum EvrstaJela { Predjelo, GlavnoJelo, Desert};  public enum Epredjelo { Juha, HladnoPredjelo, ToploPredjelo};  public enum EGlavnoJelo { JeloMeso, JeloRiba, JeloProvrće, Salata};  public enum EDesert { PečeniKolač, NepečeniKolač, KruhIPeciva};  public enum EBrTezine { Jedan, Dva, Tri, Četiri, Pet};  public enum ENazivMjere { kg, dkg, g, mg, dl, ml, l, kom, žlica, čaša, prstohvat, žličica};  public enum EVrstaNamirnice { Meso, Riba, Voće, Povrće, Začin, MliječniProizvod, Žitarica, OrašastoVoće, Ostalo, Masnoća}; | * Ova klasa sadržava sve enumeracije koje se koriste u programu. Kao usporedbu možemo pogledati klasni dijagram u kojem su enumeracije naznačene ljubičastim okvirima. |
| ReceptiFilter.cs | public int Id  public string Naziv  public bool ImaSve  public List<Namirnice> NedostajuNam  public List<string> ImaNamirnice | * Ova pomoćna klasa služi kako bi se prikazali rezultati pretrage recepta u ListView-u. Njezin objekt se sastoji od komponenata objekta tipa Recept te od Namirnica filtriranih po objektu tipa Sastavnica. |

*Tablica 2. Pomoćne klase*

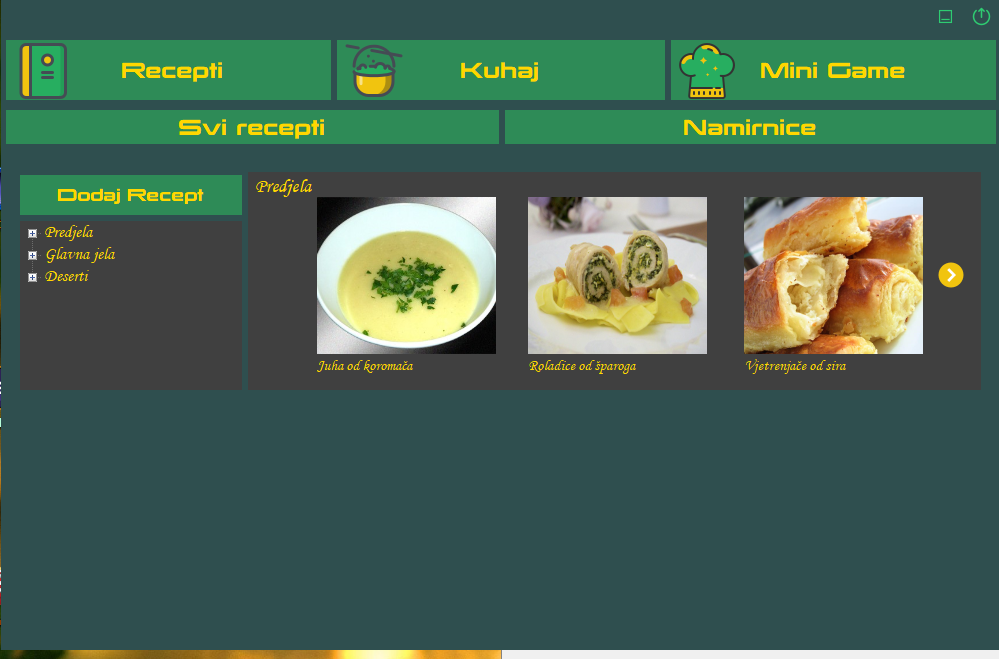
Sada nam još samo ostaje opisati klase koje imaju svoju posebnu svrhu u radu aplikacije. One ne služe za prikaz podataka niti za držanje podataka već se u njima nalaze posebne funkcije koje koristimo tijekom rada aplikacije.

|  |  |  |
| --- | --- | --- |
| Naziv klase | Sastavnice | Funkcionalnosti |
| SqlKonekcija.cs |  | * Ova klasa je statična i služi povezivanju aplikacije s bazom podataka koristeći paket *Dapper.* Sadrži metode koje se mogu pozvati iz bilo kojeg kuta aplikacije i služe kako bi pozivale različite procedure iz baze podataka predajući(ili uzimajući) odgovarajuće parmetre |
| IDodNamContract  (interface) | void PopuniListuNamirnica();  void UcitajUListView(); | * Između svih klasa imamo i jedan interface koji osigurava da sve klase koje pozivaju *DodajNamirnicuForm* u sebi sadržavaju ove dvije funkcije. Te funkcije služe za ponovno popunjavanje ListView-ova i lista koje sadržavaju namirnice nakon što korisnik unese novu namirnicu u bazu. |

*Tablice 3. Ostale klase*

**6. Korisničke upute**

Kada otvori aplikaciju i uđe u glavni izbornik, korisnik ima nekoliko mogućnosti. Može pregledavati recepte, kuhati prema njima, igrati mini-game dok npr. čeka da se neki korak u receptu izvrši (primjerice da nešto zakuha), zatim može pregledati postojeće namirnice te dodati recept.



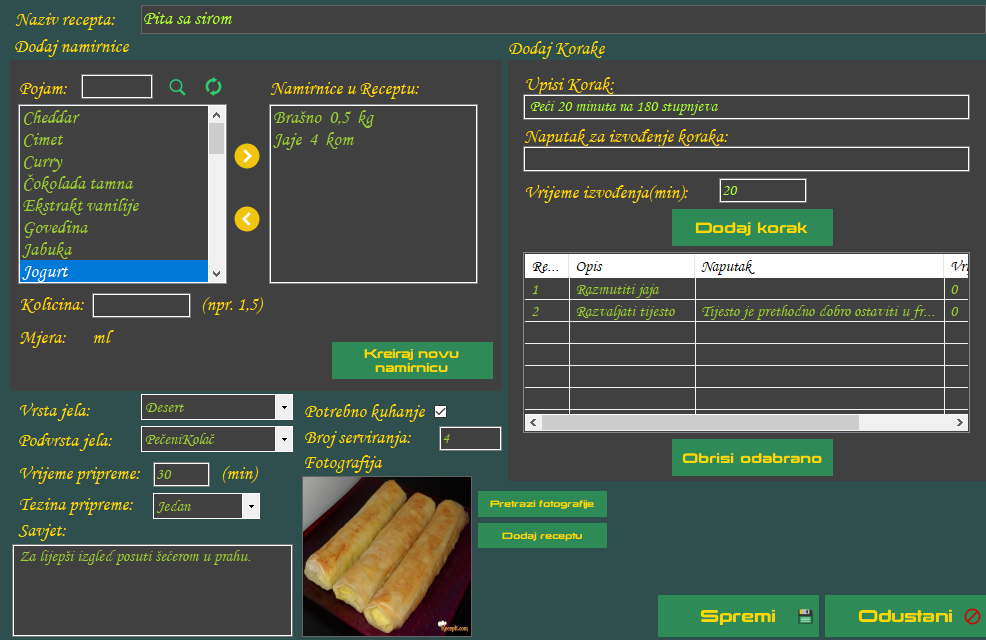
*Slika 12. Početni prozor*

Ispod opcije za dodavanje recepta imamo ponuđene tri glavne vrste jela po kojima su recepti raspoređeni. Svaka vrsta jela ima još nekoliko pod-vrsti (npr. predjela su podjeljena na juhe te hladna i topla predjela). Kada označimo neku vrstu jela, recepti te kategorije se prikazuju sa desne strane. A kada kliknemo na određeni recept on se otvara na dnu prozora. U tom receptu vidimo sliku jela, korake pripreme, sastojke, te informacije o tom receptu kao što su vrijeme pripreme,težina pripreme i sl. Usput nam se nudi i opcija za brisanje tog istog recepta.



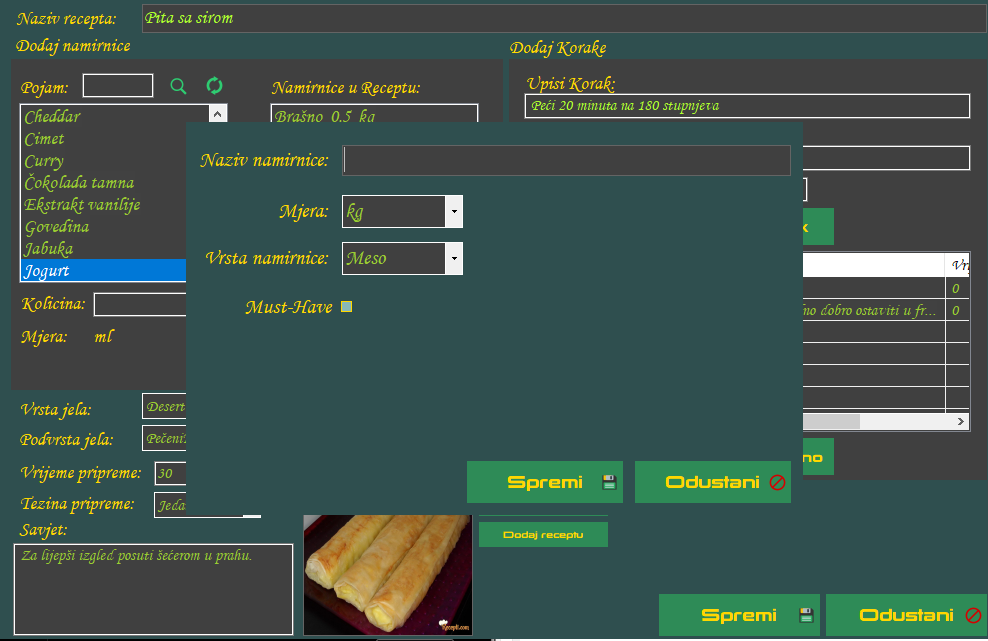
*Slika 13. Detalji recepta*

Kada odaberemo “Dodavanje recepta”, u prvom retku nam se nudi da upišemo ime recepta. Zatim dolazimo do dodavanja namirnica. One su automatski sve izlistane, no možemo ih i pretraživati u tražilici koja se automatski prepravlja s onim što je upisano. Namirnicu koju želimo dodati u recept označimo, upišemo joj željenu količinu, te pritisnemo strelicu koja pokazuje desno da bismo ju prebacili u listu “Namirnice u receptu”. U slučaju da količina nije upisana, biti će nemoguće pritisnuti na strelicu. Prebačene namirnice nestaju iz liste sa lijeve strane, a u slučaju da smo namirnicu slučajno dodali lako je možemo vratiti pritiskom na strelicu u lijevo.



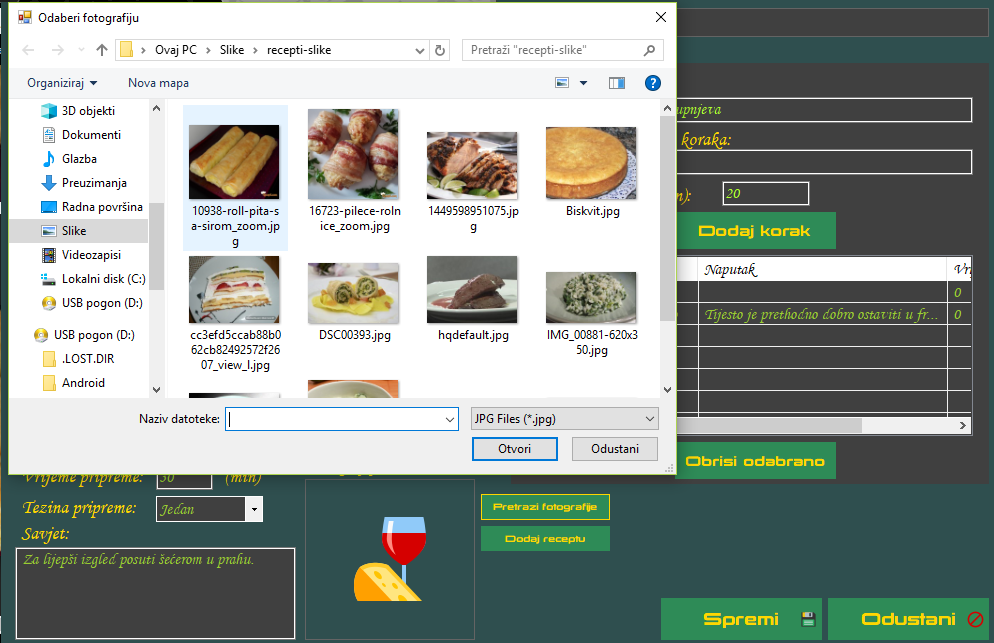
*Slika 14. Dodavanje recepta*

Ako namirnica koju želimo dodati ne postoji, imamo mogućnost dodavanje nove namirnice. Novoj namirnici upisujemo ime, odabiremo mjeru u kojoj će se prikazivati te upisujemo količinu. Usput imamo i mugućnost da namirnicu označimo kao “must have”, a to se odnosi na one namirnice koje bi stalno trebli imati (npr sol).



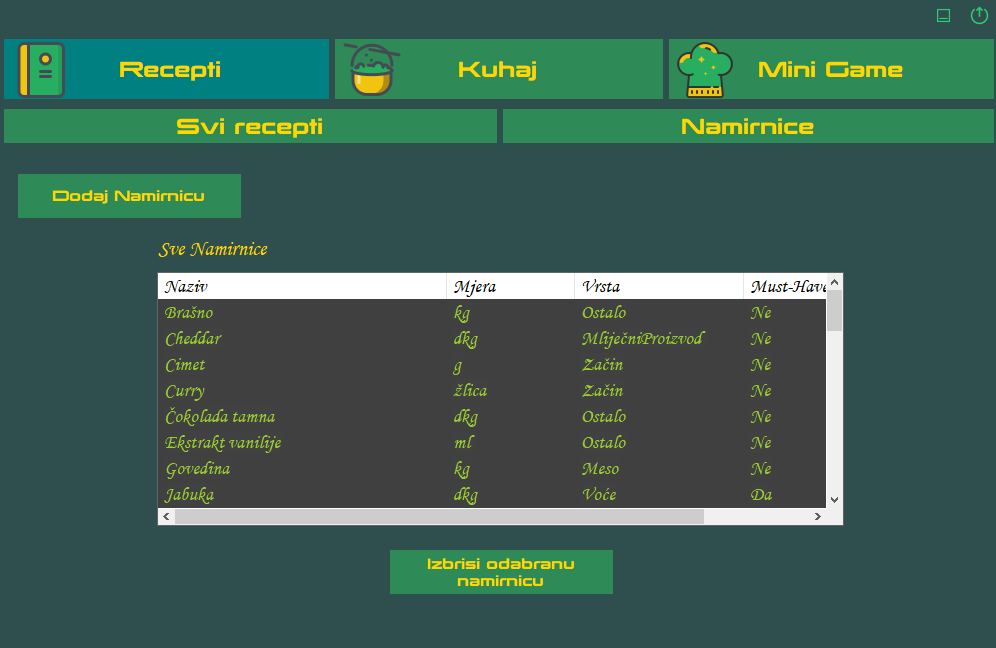
*Slika 15. Dodavanje namirnice unutar Dodavanja recepta*

Nakon što smo dodali željene namirnice odabiremo vrstu i podvrstu jela, upisujemo potrebno vrijeme pripreme, odabiremo težinu pripreme (od 1 do 5), upisujemo broj serviranja, upisujemo savjet ako želimo, dodajemo fotografiju (otvara se OpenFileDialog prozor i učitavamo sliku sa računala) te dodajemo korake. Kada dodajemo korake oni se spremaju u listu s rednim brojevima po redu kako smo ih dodavali. Za svaki korak upisujemo vrijeme njegovog izvođenja, a možemo dodati i naputak.



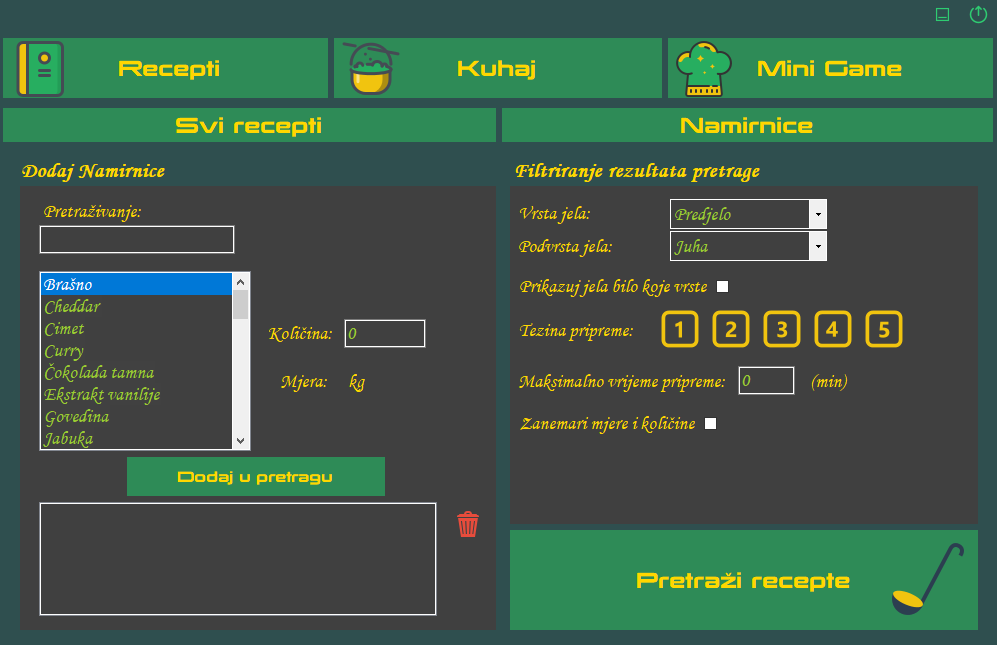
*Slika 16. Dodavanje fotografije*

Kada kliknemo na “Namirnice” u glavnom prozoru, otvara nam se lista sa svim namirnicama koju možemo razvrstati prema nazivu, vrsti, ili mjeri. Tu nam se nude i opcije dodavanja te brisanja namirnica. U slučaju da je neka namirnica navedena u nekom receptu kao sastojak, njeno brisanje neće biti moguće. Isto tako ne možemo dodati namirnicu koja već postoji.



*Slika 17. Prikaz namirnica*

Kada u glavnom izborniku odaberemo “Kuhaj” otvoriti će nam se mogućnost pretrage recepata po određenim namirnicama (moguće je i upisati potrebnu količinu namirnice). A sa desne strane imamo mogućnost filtriranja rezutata pretrage. Jela možemo filtrirati prema vrsti i podvrsti ili zanemariti vrstu jela stavljanjem kvačice na “Prikazuj jela bilo koje vrste”. Zatim možemo odabrati težinu pripreme. Težina pripreme se odabire na način da kada stavimo kvačicu na primjerice 2, prikazivati će se samo jela težine dva, a kada bismo kvačicama označili 2 i 4, prikazivati će se sva jela između težina 2 i 4. Zadnje dvije mogućnosti kod filtriranja su upisivanje maksimalnog vremena potrbnog za pravljenje jela te mogućnost da zanemarimo potrebne količine i mjere u slučaju da nas samo zanima koje se namirnice koriste, a ne i koliko ih je potrebno.

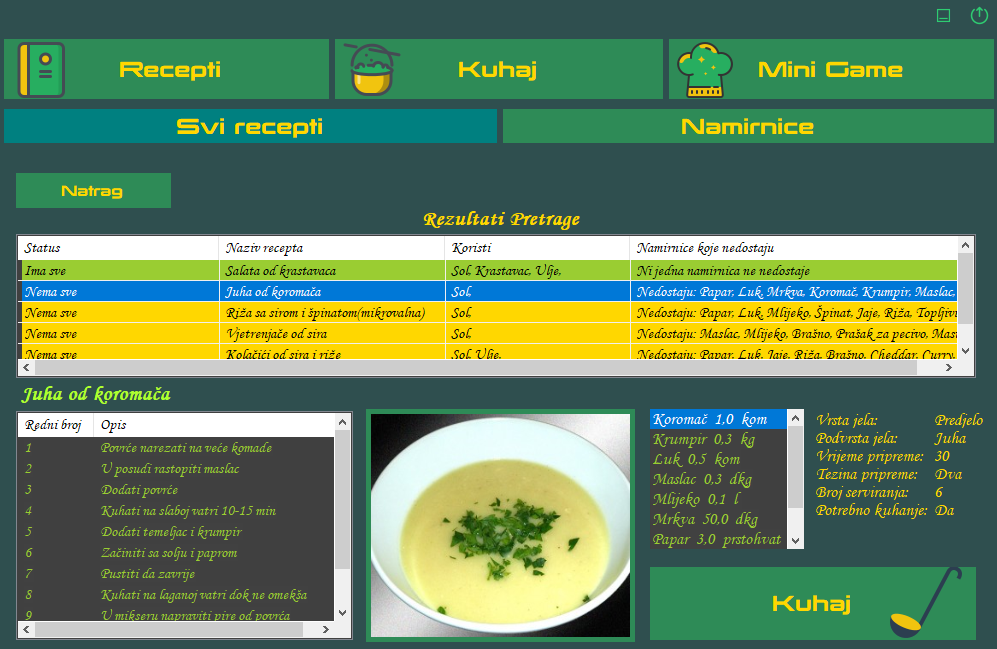


*Slika 18. Pretraživanje po namirnicama 1*



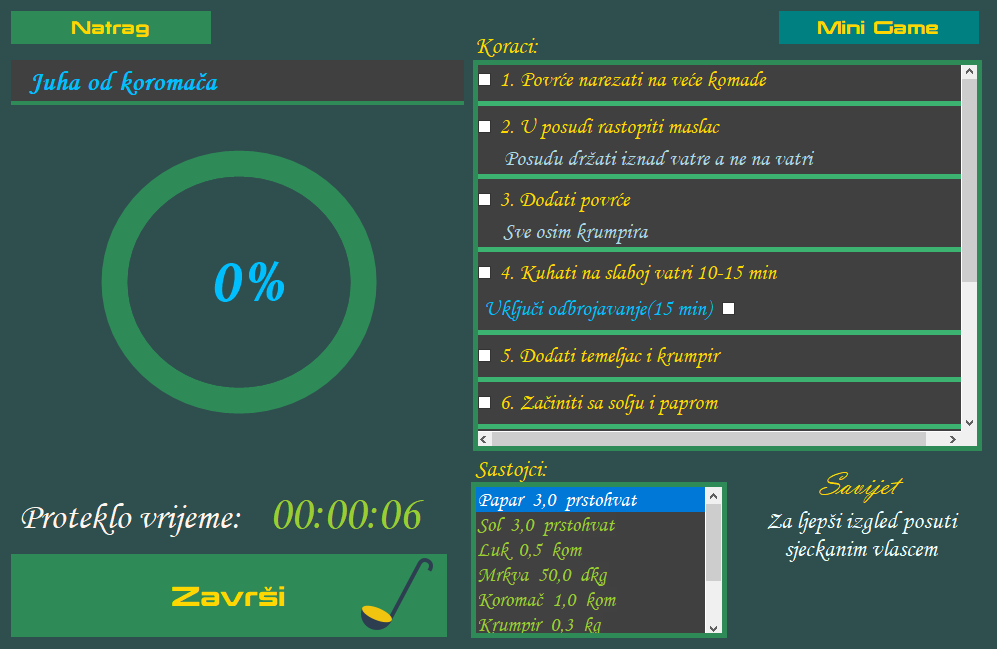
*Slika 18. Pretraživanje po namirnicama 2*

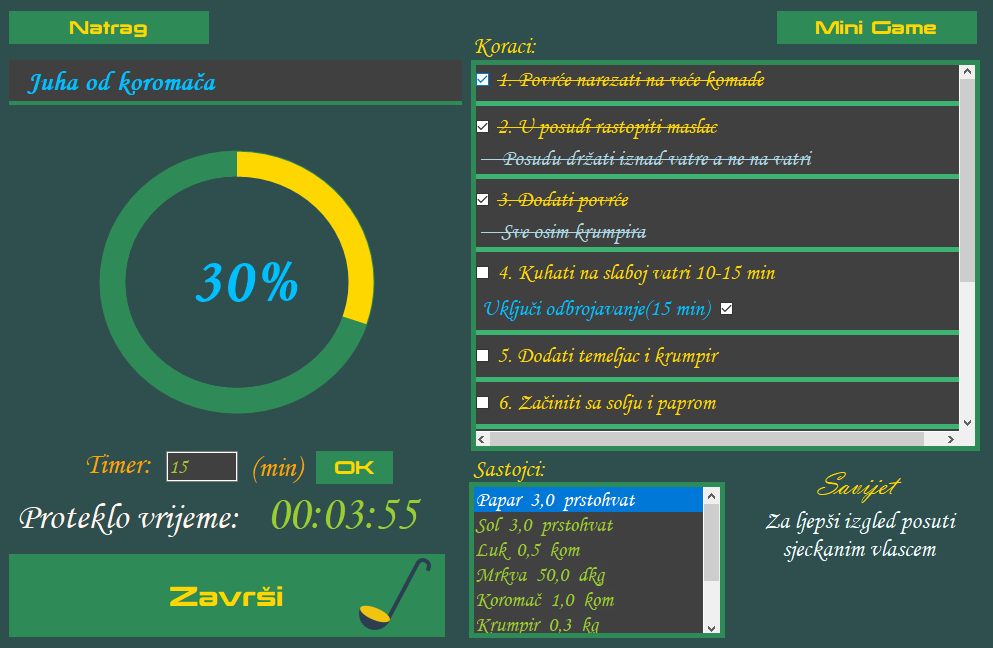
Nakon što pritisnemo “Pretraži recepte” izbacuje nam se lista sa receptima koji u potpunosti ili djelomično zadovoljavaju kriterije filtriranja. Rezultati prikazuju status recepta (ima sve/ nema sve sastojke), naziv recepta, namirnice koje se koriste, te namirnice koje nedostaju. Recepti koji imaju status “ima sve” su obojeni zeleno dok su recepti sa statusom “nema sve” obojeni žuto. Kada kliknemo na željeni recept, on se otvara na dnu prozora te se prikazuju koraci pripreme, slika recepta, potrebne namirnice, informacije o jelu te gumb za početak kuhanja.



*Slika 19. Rezultati pretrage*

Kada pritisnemo na “Kuhaj” u rezultatima pretrage, prikazuju nam se postotak završetka jela, proteklo vrijeme od početka te svi koraci i naputci ako ih neki korak ima. Svaki korak i naputak možemo označiti kada smo gotovi s njime te se on tada precrta. Neki koraci mogu sadržavati i opciju odbrojavanja koje je automatski podešeno na 15 minuta, no može se podesiti i na željeno vrijeme. Kada odbrojavanje završi pali se alarm. Tu se još nalazi i mogućnost igranja mini igre memory dok primjerice čekamo da tajmer završi odbrojavanje.







*Slike 20, 21, 22. Praćenje recepta u kuhanju i mini-game memori*

Na kraju kada pritisnemo “Završi”, iskače nam MessageBox sa porukom “Bon Appetit, jelo je gotovo”.



*Slika 23. Poruka pri završetku kuhanja*