SVEUČILIŠTE U ZAGREBU

**FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA**

QuickDiary

*Case study*

Objektno oblikovanje

Matej Jurić

Zagreb, siječanj, 2020.

**Sadržaj**

Table of Contents

[1. Design story 3](#_Toc31560129)

[2. Use cases 4](#_Toc31560130)

[2.1. Registracija korisnika 4](#_Toc31560131)

[2.2. Prijava korisnika 5](#_Toc31560132)

[2.3. Dodavanje novog zapisa 5](#_Toc31560133)

[2.4. Prepravljanje zapisa 6](#_Toc31560134)

[2.5. Filtriranje zapisa 7](#_Toc31560135)

[2.6. Odjava korisnika 7](#_Toc31560136)

[2.7. Brisanje korisnika 7](#_Toc31560137)

[2.8. Sortiranje zapisa 8](#_Toc31560138)

[3. Objektni model 9](#_Toc31560139)

[4. Arhitektura 10](#_Toc31560140)

[4.1. MVC 10](#_Toc31560141)

[4.2. Struktura sustava 10](#_Toc31560142)

[4.3. Observer obrazac 12](#_Toc31560143)

[5. GUI 13](#_Toc31560144)

# **Design story**

QuickDiary je aplikacija namijenjena za praćenje svakodnevnih osjećaja, misli i načina življenja. Pomoću nje korisnik će lakše moći pratiti kako se osjećao tijekom godine. Cilj aplikacije je da se nakon dužeg vremena ti podaci mogu analizirati od strane korisnika ili čak nekog stručnjaka. Ljudima je teško u vremenu kad su tužni i osjetljivi sjetiti se vremena kad su bili sretni. Svakodnevnim zapisivanjem navika i običaja koji ljudi imaju pruža se prilika da se otkrije što neku osobu sprječava da bude sretna i zadovoljna. S obzirom na to da većina ljudi nema puno vremena ni volje gubiti vrijeme svaki dan pišući dnevnik ova aplikacija nudi mogućnost da se u jednoj riječi rezimira prethodni dan. Primaran cilj aplikacije je zabilježavanje osjećaja iz dana u dan. Korisnik može označiti jedan osjećaj koji je prevladavao taj dan. Osjećaji koji se nude su:

* + 1. Neutralno
    2. Sretno
    3. Tužno
    4. Ljuto
    5. Razočarano
    6. Stresirano
    7. Zadovoljno

Kako bi se detaljnije opisao osjećaj koji je prevladavao taj dan, korisnik može opisati cijeli dan kao što bi to napravio u običnom dnevniku uz mogućnost unošenja nekih važnijih ključnih riječi. Aplikacija također nudi mogućnost praćenja prehrane, navika spavanja i dnevnih postignuća ili zadataka koje je korisnik obavio. Može se koristiti tijekom dana, no zamišljeno je da se na kraju dana u aplikaciju unesu svi podaci.

Korisnik će uz kronološki prikaz dana moći i filtrirati dane po određenim kriterijima. Kako bi više korisnika moglo koristiti aplikaciju putem jednog računala korisnici se trebaju registrirati i prijaviti prije samog korištenja.

Sudionici:

1. Korisnik
2. Aplikacija
3. Baza podataka

# **Use cases**

* 1. Registracija korisnika
  2. Prijava korisnika
  3. Dodavanje novog zapisa
  4. Prepravljanje zapisa
  5. Filtriranje zapisa
  6. Odjava korisnika
  7. Brisanje korisnika
  8. Sortiranje zapisa

### 2.1. Registracija korisnika

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik je odabrao registraciju

**Postcondition:** Novi korisnik je zabilježen u bazu podataka

**Glavni uspješni scenarij:**

1. Korisnik otvara zaslon za registraciju
2. Korisnik unosi svoje korisničko ime i lozinku
3. Korisnik se uspješno registrira
4. Korisnika se preusmjerava na prijavu

**Alternativni scenarij:**

Korisnik može odustati od registracije čime use case završava neuspješno. Osim odustajanja, korisnik može:

1. Unijeti postojeće korisničko ime
   * + 1. Aplikacija upozorava da se unese drugo korisničko ime
       2. Aplikacija traži ponovni upis
2. Unijeti lozinku kraću od 8 znakova

Aplikacija upozorava da se upiše jača lozinka

Aplikacija traži ponovni upis

### 2.2. Prijava korisnika

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik je odabrao prijavu

**Postcondition:** Korisnik se s prijave preusmjerava na početni zaslon aplikacije

**Glavni uspješni scenarij:**

1. Korisnik otvara zaslon za prijavu
2. Korisnik unosi korisničko ime i lozinku
3. Korisnika se preusmjerava na početni zaslon aplikacije

**Alternativni scenarij:**

Korisnik može odustati od prijave čime use case završava neuspješno. Osim odustajanja, korisnik može:

1. Unijeti krive podatke

Aplikacija upozorava o krivom korisničkom imenu ili lozinki

Aplikacija traži ponovni upis

### 2.3. Dodavanje novog zapisa

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Korisnik se uspješno prijavio u aplikaciju

**Postcondition:** Korisnik uspješno dodaje novi zapis

**Glavni uspješni scenarij:**

1. Korisnik odabire unos novog zapisa
2. Korisnik unosi podatke u formu i sprema zapis

**Alternativni scenarij:**

Korisnik može odustati od stvaranja novog zapisa čime use case završava neuspješno. Osim odustajanja, korisnik može:

1. Unijeti postojeći datum
   1. Korisnika se upozorava da datum postoji
   2. Korisnik treba unijeti drugi datum
2. Zaboraviti unijeti osjećaj
   1. Korisnika se upozorava da nije unio osjećaj
   2. Korisnik treba odabrati jedan od osjećaja

### 2.4. Prepravljanje zapisa

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik odabire zapis koji želi prepraviti

**Postcondition:** Promjene korisnika su uspješno pohranjene

**Glavni uspješni scenarij:**

1. Korisnik odabire zapis
2. Korisnik unosi promjene
3. Korisnik uspješno sprema promijenjeni zapis

**Alternativni scenarij:**

Korisnik može odustati od prepravljanja zapisa čime use case završava neuspješno.

### 2.5. Filtriranje zapisa

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik odabire filtriranje

**Postcondition:** Korisniku su prikazani filtrirani podaci

**Glavni uspješni scenarij:**

1. Korisnik unosi filter
2. Korisniku se prikazuju filtrirani podaci

**Alternativni scenarij:**

Korisnik može odustati od filtriranja čime use case završava neuspješno.

### 2.6. Odjava korisnika

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik odabire odjavu

**Postcondition:** Korisnik je odjavljen i preusmjeren na prijavu

**Glavni uspješni scenarij:**

1. Korisnik odabire odjavu korisničkog računa
2. Korisnika se uspješno odjavljuje i preusmjerava na zaslon za prijavu

**Alternativni scenarij:**

Korisnik može odustati od odjave korisničkog računa čime use case završava neuspješno.

### 2.7. Brisanje korisnika

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik odabire brisanje korisničkog računa

**Postcondition:** Korisnik je izbrisan i preusmjeren na prijavu

**Glavni uspješni scenarij:**

1. Korisnik odabire brisanje korisničkog računa
2. Korisnika se upozorava da potvrdi svoju odluku
3. Korisnika se još jednom upozorava da nema povratka nakon potvrde
4. Korisnički račun i svi vezani podaci se brišu iz baze podataka

**Alternativni scenarij:**

Korisnik može odustati od brisanja korisničkog računa čime use case završava neuspješno.

1. Korisnik može na upozorenje o potvrdi brisanja odustati
   1. Korisnika se vraća na prethodni zaslon

### 2.8. Sortiranje zapisa

**Primary actor:** Korisnik

**Stakeholders:**

* Baza podataka

**Precondition:** Sustav je pokrenut i korisnik odabire stupac po kojem želi sortirati

**Postcondition:** Korisniku je prikazana sortirana lista zapisa

**Glavni uspješni scenarij:**

1. Korisnik odabire stupac za sortiranje
2. Korisniku se prikazuju sortirana list zapisa

**Alternativni scenarij:**

Korisnik može odustati od sortiranja zapisa čime use case završava neuspješno.

# **Objektni model**

A screenshot of a social media post

Description automatically generated

Slika 1. Objektni model

# **Arhitektura**

## **4.1. MVC**

QuickDiary je realizacija MVC odnosno Model-View-Controller obrasca arhitekture. View predstavlja prikaze podataka korisniku kroz različite ekrane (*screen*). Controller je zadužen za primanje i procesiranje različitih zahtjeva, definiranje ponašanja sustava i mapiranje korisničkih akcija u promjene modela. Model je enkapsulacija stanja sustava koji odgovara na promjene stanja i otkriva funkcionalnosti sustava. Veza između Modela i Viewa je tu kako bi se promjene u modelu mogle direktno očitovati u Viewu. To ponašanje ostvaruje se pomoću *Observer* obrasca.

A screenshot of a cell phone

Description automatically generated

Slika 2. MVC

## **4.2. Struktura sustava**

Aplikacija je podijeljena u 6 logičkih cjelina, uz cjelinu za testove, prikazanih na sljedećoj slici.

A screenshot of a cell phone

Description automatically generated

Slika 3. Struktura aplikacije

AppStarter je aplikacija koja će pokrenuti cijeli sustav. Ona sadrži reference na sve ostale cjeline. Unutar Main metode jedinog razreda u AppStarteru, inicijalizira se *WindowsFormsFactoryi*, te *MainFormController* kojem se prosljeđuje instanca *WindowsFormFactory*, te instance *UserRepositorya, RecordRepositorya* i *CurrentUserRepositorya*. Nakon početnih inicijalizacija prikazuje se forma za registraciju.

BaseLib je *class library* u kojem se nalaze sučelja za forme, *MainController*, te sučelja za Observer obrazac.

Controllers sadrži implementacije glavnog *MainFormControllera*, kao i implementacije svih ostalih kontrolera korištenih u sustavu.

MemoryBasedDAL sadrži implementacije tri repozitorija, *CurrentUserRepository*, *RecordRepository* i *UserRepository*. Sučelja ovih repozitorija definirana su unutar Model cjeline.

Model sadrži implementacije korisnika (*User*)i zapisa (*Record*). Uz to definirana su sučelja repozitorija koji se implementiraju unutar MemoryBasedDAL-a.

PresentationLayer sadrži sve korištene forme, te razred *WindowsFormsFactory* koji je zadužen za stvaranje pojedine forme.

Na sljedećoj slici prikazan je način na koji su svaka od cjelina povezane međusobno.

A close up of a map

Description automatically generated

Slika 4. Reference cjelina sustava

Za sve akcije koje korisnik napravi unutar forme odnosno View-a, pozivaju se metode unutar *MainFormControlleru*, koji zatim te akcije prosljeđuje kontroleru zaduženom za tu određenu formu. Taj specifični kontroler procesira korisničke akcije, te može ako je potrebno, mijenjati podatke i preusmjeriti korisnika na drugu formu.

## **4.3. Observer obrazac**

Kako bi se promjene mogle očitovati u Viewu, kao što je prethodno napomenuto koristi se *Observer* obrazac. Subjekt odnosno *Observable* je repozitoriji čijom se promjenom stanja treba promijeniti stanje u View. U slučaju ovog sustava *Observable* je *RecordRepository*, a *Observer* je *HomeForm*, odnosno početni ekran aplikacije gdje su prikazani zapisi. *Observer* obrazac je realiziran kao što je prikazano na sljedećoj slici.

Napomena: *IRecordRepository* nema prikazane sve metode koje to sučelje ima, kako ne bi zauzelo previše prostora, također metode definirane u tom sučelju nemaju ulogu u ostvarenju *Observer* obrasca.

A screenshot of a cell phone

Description automatically generated

Slika 5. Observer obrazac

# **GUI**

A screenshot of a cell phone

Description automatically generated

Slika 6. Registration screen

A screenshot of a cell phone

Description automatically generated

Slika 7. Login screen

A screenshot of a cell phone

Description automatically generated

Slika 8. Home screen

Odabirom *Add new* otvara se ekran prikazan na slici 9. Dvostrukim pritiskom na datum jednog od zapisa otvara se ekran prikazan na slici 10. Jednim pritiskom na datum, te zatim pritiskom na gumb *Edit* otvara se ekran prikazan na slici 11.

A screenshot of a social media post

Description automatically generated

Slika 9. Dodaj novi zapis screen

A screenshot of a cell phone

Description automatically generated

Slika 10. Detalji zapisa screen

A screenshot of a cell phone

Description automatically generated

Slika 11. Prepravi zapisa screen