

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2050  
**ANALIZATOR DNEVNIKA  
INFORMACIJSKOG SUSTAVA**

Mislav Marković

Zagreb, veljača 2020.

SVEUČILIŠTE U ZAGREBU  
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 2050  
**ANALIZATOR DNEVNIKA  
INFORMACIJSKOG SUSTAVA**

Mislav Marković

Zagreb, veljača 2020.

## IZVORNIK ZADATKA

Zahvaljujem mentoru prof. dr. sc. Krešimiru Fertalju na iskazanom strpljenju i pruženoj podršci prilikom izrade ovoga rada.

# Sadržaj

1. Uvod.....	1
2. Pregled postojećih rješenja.....	2
2.1. ManageEngine EventLog Analyzer.....	2
2.2. Loggly.....	4
3. Specifikacija zahtjeva.....	8
3.1. Korisnički zahtjevi.....	8
3.2. Zahtjevi sustava .....	8
3.2.1. Funkcionalni zahtjevi .....	8
3.2.2. Nefunkcionalni zahtjevi.....	9
4. Dnevnik informacijskog sustava .....	10
4.1. Dnevnik sustava SQL Server.....	11
4.2. Normalizacija dnevnika .....	11
5. Korelacija događaja.....	13
5.1. Prvi korak.....	13
5.2. Drugi korak .....	16
6. Arhitektura sustava.....	18
6.1. Struktura sustava.....	19
6.2. Model podataka.....	20
6.3. Web poslužitelj .....	32
6.3.1. SawMill.WebApi.....	33
6.3.2. SawMill.Processor .....	38
6.3.3. SawMill.Data.....	43
6.4. Web klijent.....	46
6.4.1. Komponente .....	48
6.4.2. Spremišta .....	51
6.5. Servis za prikupljanje dnevnika .....	51

7. Funkcionalnosti sustava .....	53
7.1. Sustav.....	53
7.2. Komponenta.....	57
7.3. Značajni vremenski interval.....	59
7.4. Korelacijska grupa .....	62
7.5. Zapisi dnevnika.....	64
7.6. Izvještaj sustava .....	78
7.7. Postavke analizatora .....	79
8. Korištene tehnologije i alati .....	81
8.1. Alati .....	81
8.1.1. Visual Studio Community 2017 .....	81
8.1.2. Visual Studio Code.....	81
8.1.3. Microsoft SQL Server Managment Studio 2017.....	81
8.2. Tehnologije .....	81
8.2.1. ASP.NET Core 2.2 .....	81
8.2.2. RestSharp.....	81
8.2.3. Topshelf.....	82
8.2.4. Vue.js.....	82
8.2.5. Vuex .....	82
8.2.6. Vue Router.....	82
8.2.7. Vuetify .....	82
8.2.8. Axios.....	83
8.2.9. NuGet .....	83
8.2.10. NPM.....	83
8.2.11. Microsoft SQL Server.....	83
9. Zaključak.....	84
Sažetak.....	85

Abstract.....	86
Literatura .....	87
Dodatak: Popis slika .....	90
Dodatak: Popis isječaka koda .....	93

# 1. Uvod

Povećanjem kompleksnosti programske potpore pokazuje se potreba za detaljnim bilježenjem dnevnika izvođenja (engl. *log file*) kako bi se nakon pojave pogreške u radu sustava ona mogla pronaći i ispraviti. Dnevnik informacijskog sustava je od ključne važnosti za uspješno održavanje informacijskog sustava. Osim pomoći pri pronalasku grešaka u radu informacijskog sustava dnevnik informacijskog sustava može poslužiti u analizi efikasnosti rada sustava. Sustav za analizu dnevnika informacijskog sustava omogućava korisnicima da steknu razumijevanje o ponašanju svoje programske podrške u produkciji te da pravovremeno uoče moguće probleme.

Tema ovog rada je analizator dnevnika informacijskog sustava, to jest, razvoj programske podrške koja pomaže u praćenju stanja informacijskog sustava provodeći automatiziranu analizu njegovog dnevnika.



## 2. Pregled postojećih rješenja

Analiza dnevnika informacijskog sustava je čest problem u razvoju programske podrške pa danas postoji niz komercijalnih i besplatnih alata koji nam mogu pomoći u tom zadatku.

### 2.1. ManageEngine EventLog Analyzer

ManageEngine nudi programsku podršku za niz područja koja su ključna za rad informacijskog odjela bilo koje organizacije. EventLog Analyzer je proizvod koji pruža mogućnosti prikupljanja i analize raznih dnevnika. EventLog Analyzer je usmjeren na praćenje stanja u mreži te prikupljanjem dnevnika mrežnih usmjernika (eng. router), web poslužitelja, baza podataka i ostalih uređaja spojenih na mrežu pokušava otkriti zloćudne napade kao i nepravilan rad mreže. Nakon prikupljanja dnevnika na centralno mjesto EventLog Analyzer obavlja analizu prema postojećim pravilima korelacije te tako pokušava otkriti nepravilnosti u radu mreže. Proizvod ima prednost jer je dio većeg skupa aplikacija koje rješavaju probleme u sličnoj domeni te može iskoristiti postojeće baze podataka u svojoj analizi. Programska podrška nudi velik broj unaprijed definiranih formata dnevnika. Korisniku se nudi mogućnost filtriranja dnevnika kako bi se olakšala analiza [1].

Slika 2.1 prikazuje tablicu s zapisima dnevnika. Prikazane su informacije o vremenu događaja, uređaju događaja, identifikator vrste događaja, vrijednost prikaza imena uređaja, izvor događaja i ozbiljnost događaja.

Time	Device	Event ID	DisplayName	Source	Severity
2020-02-09 19:04:39	DESKTOP-U303OM7	4798	DESKTOP-U303OM7	Microsoft-Windows-Security-Auditing	Success
2020-02-09 19:02:46	DESKTOP-U303OM7	<a href="#">4798</a>	DESKTOP-U303OM7	Microsoft-Windows-Security-Auditing	Success

Slika 2.1 Prikaz zapisa dnevnika [1]

Slika 2.2 prikazuje formu za dodavanje novog korelacijskog pravila. Korelacijsko pravilo sastoji se od imena, opisa i pravila za korelaciju. Pravila su definirana upozorenja koja se postavljaju na određena svojstva zapisa i povezuju se u grupu.

Add New Rule
×

\*Rule Name

Logon failure

Description

Becomes active on logon failure

Rule Criteria :

Alert Profile

▼

Equals

▼

Excessive logon failures

▼

+

OR

▼

Device Name

▼

Equals

▼

DESKTOP-U303OM7

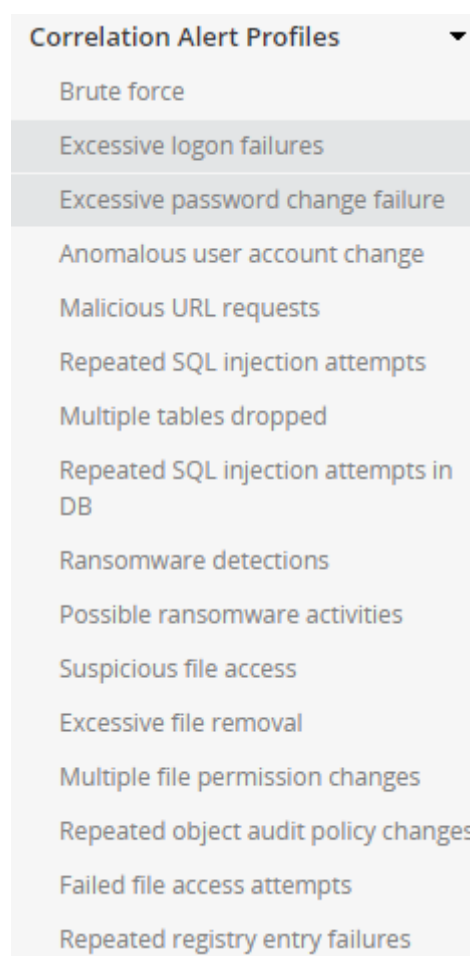
▼

+

−

Slika 2.2 Dodavanje novog korelacijskog pravila [1]

Slika 2.3. prikazuje dio upozorenja koje je moguće koristiti prilikom definiranja korelacijskih pravila.



Slika 2.3 Predefinirana upozorenja [1]

Slika 2.4 prikazuje izvještaj učestalosti tipova događaja i njihove ozbiljnosti.

Source	Failure	Success	Information	Warning	Error
application error	0	0	0	0	2
application hang	0	0	0	0	1
bthusb	0	0	2	0	0
desktop window manager	0	0	1	0	0
esent	0	0	10	0	5
eventlog	0	0	8	0	0
eventloganalyzer	0	0	115	0	0

Slika 2.4 Izvještaj događaja [1]

## 2.2. Loggly

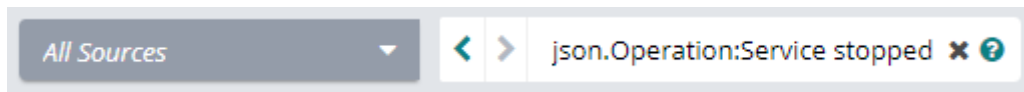
Loggly je rješenje za prikupljanje i analizu dnevnika raznih aplikacija, sustava, uređaja i infrastrukture. Osim prikupljanja dnevnika Loggly nudi mogućnosti nadziranja sustava te analizu dnevnika pomoću raznih izvještaja i grafova koji pomažu u razumijevanju stanja kompleksnog sustava. Glavna prednost Logglya je prikupljanje velike količine podataka iz neograničenog broja heterogenih sustava te brza obrada tih podataka. Loggly podržava integraciju s Jiom, Slackom, HipChatom i drugim aplikacijama kako bi korisniku izdao upozorenje uočeno prema korisničkim pravilima. Upozorenja služe kako bi korisnik bio obaviješten o određenom stanju u sustavu. Upozorenje sadrži pravilo prema kojem ispituje događaje, ukoliko određeni broj događaja zadovolji pravilo unutar zadanog vremenskog okvira Loggly obavještava korisnika. Korisnik obavijest može primiti elektronskom poštom ili obavijest može biti poslana na definiranu pristupnu točku (engl. *endpoint*). Pravila za upozorenja se izvršavaju periodički prema korisnikovim postavkama [2].

Slika 2.5 prikazuje primjer zapisa dnevnika. Unutar Logglya moguće je odrediti koja svojstva dnevnika želimo prikazati.

2020-02-09 19:35:26.458	INFO	MS_NT_EVENTLOG_PROVIDER provider started with result code 0x0. HostProcess = wmiprivse.exe;
2020-02-09 19:34:36.667	DEBUG	WNP Transport Layer received command for the 0x504E472031383520434F4E0037390D0A4D532D43563
2020-02-09 19:34:36.667	DEBUG	WNP Transport Layer received command for the 0x4D532D43563A204D6F384649465A2F3345696336554

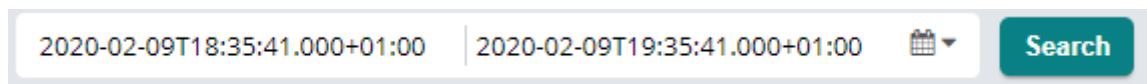
Slika 2.5 Prikaz zapisa dnevnika [2]

Prikazane zapise moguće je pretraživati. Slika 2.6 prikazuje uvjet pretraživanja prema vrijednosti korisničkog svojstava *Operation*. Također moguće je odrediti izvor zapisa koje pretražujemo.



Slika 2.6 Pretraga zapisa - uvjet pretrživanja [2]

Pretraživanje je moguće ograničiti na vremenski interval kako je prikazano na slici 2.7.



Slika 2.7 Pretraga zapisa - vremenski raspon [2]

Osim svojstava strukture zapisa koja su predefinirana u Loggly programskoj podršci moguće je definirati vlastita svojstva. Vlastito svojstvo možemo definirati kao par ključ-vrijednost, vrijednost učvršćena (engl. *anchored*) između početne i završne vrijednosti ili kao regularni izraz. Slika 2.8 prikazuje definiciju vlastitog svojstva regularnim izrazom. Vlastito svojstvo se primjenjuje na poruku događaja i poprima vrijednost opisa operacije koju taj događaj predstavlja. Vlastito svojstvo ne mora imati vrijednost u svakom događaju.

The image shows a 'Define rule' dialog box. It has a title bar with a minus icon and the text 'Define rule'. Inside, there are two main sections. The first section is labeled 'Event matches RegEx:' and contains a text input field with the value '^Operation:(.+)?\$'. The second section is labeled 'Extract Field From:' and contains a dropdown menu with 'json.Message' selected. Below these sections, there is a checkbox labeled 'Limit rule scope (recommended - improves accuracy)' which is checked, followed by three radio buttons: 'All Hosts' (selected), 'All Applications', and 'All Tags'. At the bottom of the dialog, there are three buttons: 'Test rule on Events' (green), 'Continue »' (green), and 'Cancel' (gray).

Slika 2.8 Definiranje vlastitog svojstva atributa [2]

Slika 2.9 prikazuje formu za stvaranje novog upozorenja. Upozorenje se sastoji od imena, opisa, termina pretrage, uvjeta učestalosti, vremenskog okvira i načina obavještanja korisnika.

The image shows a web form for creating a new alert in Loggly. It is divided into several sections:

- Name:** A text input field containing "Operation alert" with a small icon to its right.
- Description:** A text input field containing "Alerts when operation is service stopped".
- Saved Search:** A section with a note: "Visit the search page to save a search. 'Custom Search Context' means this alert was not created from a saved search." Below this is a dropdown menu showing "Service stopped" and a text field containing the search terms "terms: json.Operation:Service stopped".
- Alert if:** A section with a dropdown menu showing "Count", followed by "is", a dropdown menu showing ">", a text input field with "1", a link "(use standard deviation instead)", the text "within the last", a text input field with "1", and a dropdown menu showing "hours".
- Then:** A section with two options:
  - ☒ **Send an email**: Below this is a text input field containing "mislav (mislav.markovic95@gmail.com)" with a small 'x' icon to its right.
  - ☐ **Send to an endpoint**: Below this is a note: "Configure your own service endpoint."

Slika 2.9 Forma za stvaranje novog upozorenja [2]

Nakon što Loggly sustav otkrije zapise koji zadovoljavaju definirano upozorenje korisnik se obavještava na način definiran u tom upozorenju. Slika 2.10 prikazuje obavijest elektroničkom poštom. Obavijest sadrži opis upozorenja, vremenski interval u kojem je upozorenje otkriveno i uvjet korišten za ispitivanje odgovaraju li zapisi upozorenju.

**Operation alert (4 hits)**

View events

Suppress

**Description:**  
Alerts when operation is service stopped

**Trigger timespan:**  
Feb 09 17:35:41 to Feb 09 18:35:41 (UTC)

**Source Group(s):**  
N/A

**Query Terms:**  
json.Operation:Service stopped

...

Showing **4 of 4** alert events.

...

Alert ID: 63fabb86-a089-419d-814c-0169842098a2

Slika 2.10 Obavijest upozorenja [2]

## 3. Specifikacija zahtjeva

Specifikacija zahtjeva određuje koje mogućnosti sustav mora posjedovati kao i ograničenja koja sustav ne smije prekršiti. Zahtjeve možemo podijeliti na korisničke i zahtjeve sustava.

### 3.1. Korisnički zahtjevi

Korisnički zahtjevi su izjave u prirodnom jeziku koje opisuju usluge koje se očekuju od sustava i ograničenja pod kojima se sustav izvodi. Zahtjevi moraju biti pisani jasno, bez tehničkih termina te opisuju samo vanjsko ponašanje sustava [3].

- Korisnik mora moći definirati više sustava sastavljenih od više komponenti.
- Korisnik mora moći opisati strukturu formata dnevnika komponente.
- Korisnik mora moći opisati korelacijske grupe sustava.
- Korisnik mora moći unijeti zapise dnevnika ručno i automatizirano.

### 3.2. Zahtjevi sustava

Zahtjevi sustava u detalje opisuju funkcionalnosti, usluge i operacijska ograničenja koja implementacija sustava mora zadovoljiti. Zahtjevi sustava dijele se na funkcionalne i nefunkcionalne zahtjeve [3].

#### 3.2.1. Funkcionalni zahtjevi

Funkcionalni zahtjevi su izjave o mogućnostima koje sustav pruža, kako sustav reagira na određene ulaze i kako se sustav ponaša u specifičnim situacijama. Nekada funkcionalni zahtjevi opisuju što specifično sustav ne smije raditi [3].

- Sustav mora omogućiti definiciju korisničkog sustava.
- Sustav mora omogućiti stvaranje više komponenti.
- Sustav mora omogućiti pridjeljivanje više komponenti jednom sustavu.
- Sustav mora omogućiti definiranje pravila za normalizaciju dnevnika komponente.
- Sustav mora omogućiti definiranje vlastitih svojstava zapisa dnevnika komponente.
- Sustav mora imati mogućnost primanja zapisa dnevnika određene komponente ručnim unosom zapisa.
- Sustav mora imati mogućnost automatiziranog prijenosa dnevnika komponenti.
- Sustav mora moći normalizirati zapise dnevnika različitih formata.
- Sustav mora imati mogućnost prikaza normaliziranih zapisa dnevnika za sustav.
- Sustav mora omogućiti filtriranje prikazanih normaliziranih zapisa.

- Sustav mora imati mogućnost definiranja pravila za otkrivanje značajnih vremenskih intervala unutar dnevnika jedne komponente.
- Sustav mora prikazati otkrivene značajne vremenske intervale za komponentu.
- Sustav mora omogućiti definiranje pravila za korelacijske grupe sustava.
- Sustav mora moći prikazati otkrivene korelacijske grupe sustava.
- Sustav mora moći generirati izvještaj za značajne vremenske intervale komponenti sustava i otkrivene korelacijske grupe sustava.
- Sustav mora omogućiti parametrizaciju učestalosti izvođenja algoritma za stvaranje korelacijskih grupa.

### **3.2.2. Nefunkcionalni zahtjevi**

Nefunkcionalni zahtjevi su zahtjevi i ograničenja koja se postavljaju na sustav kao cjelinu i obično se ne tiču pojedinačnih mogućnosti sustava.

- Sustav mora podržati višekorisnički rad.
- Korisničko sučelje sustava mora biti implementirano kao jednostranična aplikacija (engl. *single page application*, skraćeno SPA).
- Sustav mora podržavati regularne izraze u definiciji pravila za obradu dnevnika.
- Sustav mora omogućiti automatizirani prijenos dnevnika više komponenti koje se nalaze na istom uređaju.



## 4. Dnevnik informacijskog sustava

Dnevnik informacijskog sustava je datoteka koja sadrži zapise događaja koji su se dogodili u nekom sustavu. Dnevnik je ključan dio svakog sustava, a njegova važnost vidljiva je prilikom pokušaja rekonstrukcije stanja sustava u nekom trenutku. Prilikom razvoja sustava moguće je promatrati izvođenje sustava pod kontroliranim uvjetima, čak i promatrati sustav kako se izvodi naredbu po naredbu. Nakon što sustav bude pušten u stvarnu uporabu, odnosno produkciju, gubimo tu mogućnost. Činjenica je da niti jedan informacijski sustav nije savršen te je neizbježno pojavljivanje grešaka u radu sustava. Nakon što je otkriveno da je u nekom vremenskom periodu sustav krivo radio, potrebno je istražiti uzrok greške koja je onemogućila ispravan rad sustava. Kako bi se otkrio uzrok greške moramo na neki način rekonstruirati stanje u kojem se sustav nalazio prilikom pojavljivanja greške te redoslijed događaja koji su sustav doveli u pogrešno stanje. Dnevnik informacijskog sustava tu ima ključnu ulogu. On se zapisuje tijekom izvođenja sustava te detaljnom istragom vremenskog perioda u kojem sustav nije radio ispravno možemo otkriti uzrok problema. Kompleksniji informacijski sustavi često imaju više od jednog dnevnika, na primjer sustav baze podataka zapisuje svoje dnevnike, mrežni poslužitelj svoj dnevnik, mrežna aplikacija i servisi dodatne dnevnike. Broj dnevnika u sustavu povećava kompleksnost istrage problema jer je potrebno istragu obaviti kroz više komponenti gdje svaka ima svoj kontekst izvođenja, a moguće je da dnevnicima nisu zapisani u istom formatu što osobi koja provodi istragu dodatno otežava posao. Uzrok greške koju istražujemo može se nalaziti u dnevniku druge komponente sustava što nije lako otkriti.

Formati dnevnika informacijskih sustava prikazuju događaje na različite načine, ipak možemo iz više formata izvući ključna svojstva događaja. To su vremenski trenutak u kojem se događaj dogodio, poruka koja opisuje sam događaj i vrijednost koja označava ozbiljnost događaja. Načina zapisa vremena je previše da bi se svi naveli, ali najčešće razlike su u poretku dana, mjeseca, godine i u načinu zapisa sata u danu (12 satni način ili 24 satni način). Ozbiljnosti događaja također variraju između formata dnevnika. Neki formati koriste numeričke vrijednosti gdje različiti rasponi brojeva predstavljaju različitu ozbiljnost događaja dok drugi formati koriste definirane ključne riječi kako bi opisali ozbiljnost događaja. Dnevnik može sadržavati dodatna svojstva koja daju dodatni kontekst događaju zapisanom u njemu, primjeri takvih svojstava su identifikator dretve, identifikator procesa,

IP adresa itd. Kako bi lakše uspoređivali događaje cijelog sustava potrebno je različite formate raznih dnevnika prikazati na jedan način.

## 4.1. Dnevnik sustava SQL Server

Kao primjer dnevnika jedne komponente informacijskog sustava razmotriti ćemo format dnevnika baze podataka SQL Server. Razmatramo dnevnik izvođenja komponente, a ne transakcijski dnevnik. Dnevnik izvođenja sadrži informacije o pokretanju i gašenju poslužitelja, arhiviranju podataka, greškama u radu poslužitelja i drugim akcijama [4]. Format ovog dnevnika je sljedeći: „Vrijeme Izvor Poruka“ i prikazan je u isječku koda 4.1.

```
2020-01-29 21:05:36.03 spid6s      Starting up database 'msdb'.
```

Isječak koda 4.1 Zapis dnevnika

Vrijeme je dano u formatu „yyyy-MM-dd HH-mm-ss.ff“. Izvor događaja je proces koji je zapisao događaj u dnevnik, u danom primjeru to je „spid6s“ odnosno „SQL Process ID 6s“. U ovom slučaju poruka je jednostavna i predstavlja opis događaja. Ukoliko se radi o zapisu greške događaj sadrži nekoliko dodatnih informacija. Isječak koda 4.2 prikazuje zapis s dodatnim informacijama.

```
2020-01-30 02:28:52.47 spid20s      Error: 17054, Severity: 16, State: 1.
```

Isječak koda 4.2 Primjer zapisa greške u dnevniku SQL servera

Vrijeme i izvor zapisa ostaju isti ali poruka sada sadrži dodatnu strukturu oblika „Broj greške Ozbiljnost greške Stanje greške“. Broj greške je jedinstveni identifikator prijavljene greške [5]. Ozbiljnost greške je broj u rasponu od 0 do 24 koji nam govori koliki je utjecaj prijavljene greške na ostatak poslužitelja i baze podataka [6]. Stanje greške je oznaka uvjeta koji su bili prisutni kada je greška prijavljena i služi za lakše otkrivanje uzroka greške [5].

## 4.2. Normalizacija dnevnika

Svođenje različitih formata u jedan zajednički format nazivamo procesom normalizacije dnevnika informacijskog sustava. Zajednički odnosno normalizirani oblik vrijeme zapisuje u formatu „dd/MM/yyyy hh:mm:ss“.

Za ozbiljnost događaja predefinirane su sljedeće razine:

- **Debug.** Označava događaje koji su korisni prilikom razvoja sustava, ali nisu poželjni nakon što je sustav pušten u produkciju te se tada isključuju.
- **Trace.** Razina ozbiljnosti koja predstavlja događaje koji su zapisani kako bi se zabilježilo da je u tom trenutku sustav izvršavao određeni dio programskog koda, zbog svoje učestalosti i male korisnosti najčešće se isključuju prilikom normalnog rada sustava.
- **Info.** Razina za događaje koji su zapisani prilikom redovnog izvršavanja sustava, uglavnom sadrže informacije o akcijama koje sustav obavlja te promjenama stanja sustava.
- **Warning.** Razina događaja koji su potencijalni problemi u daljnjem izvršavanju sustava, ali u trenutku je moguće automatski se oporaviti i nastaviti s izvođenjem.
- **Error.** Razina događaja predstavlja nemogućnost sustava da uspješno izvrši ili dovrši operaciju, ali sam sustav može nastaviti s radom.
- **Fatal.** Razina najveće ozbiljnosti događaja i predstavlja problem koji uzrokuje prestanak rada sustava.

Poruka u normaliziranom obliku ostaje kao opis događaja i ne obrađuje se posebno.

Uz navedene nužne dijelove normaliziranog oblika dnevnika svaki format sadrži dodatna svojstva koja nisu uvijek prisutna u zapisu.

## 5. Korelacija događaja

Korelacija događaja postupak je obrade niza događaja u svrhu otkrivanja grupe događaja koja se pojavljuje u otprije poznatom vremenskom okviru. Pojavljivanje definirane grupe događaja ima dodatno značenje u usporedbi s pojavljivanjem individualnih događaja iz grupe samostalno. Važno je primijetiti da je korelacijska grupa određena vremenskim okvirom, ali nije određena izvorom događaja što znači da događaji proizašli iz različitih dijelova sustava mogu sudjelovati u korelacijskoj grupi. Korelacija događaja povezuje događaje s prepoznatljivim i značajnim uzorcima u izvršavanju sustava. Kako bi proveli postupak korelacije događaja potrebno je zapise događaja svesti na isti oblik, odnosno normalizirati ih. Korisnost korelacije događaja vidljiva je u smanjenju vremena potrebnog za pronalaženje određenih uzoraka. Automatizirana korelacija događaja osigurava da nećemo propustiti određeno pojavljivanje grupe događaja zbog prevelike količine zabilježenih događaja. U ovom radu korelacija događaja temelji se na zadanim pravilima. Korisnik zadaje pravila pomoću kojih se identificiraju ključni događaji te se od njih formiraju korelacijske grupe.

### 5.1. Prvi korak

Nakon normalizacije, korelacija se obavlja u dva koraka. **Prvi korak** obrađuje normalizirane zapise jedne komponente koristeći pravila za prepoznavanje značajnih vremenskih intervala. Pravilo za prepoznavanje značajnih vremenskih intervala sastoji se od regularnog izraza, minimalnog praga pojavljivanja, vremenskog intervala, i odabranog svojstva zapisa. Regularni izraz koristi se za testiranje podudarnosti vrijednosti zapisa događaja s pravilom. Minimalni prag pojavljivanja određuje koliko puta pravilo mora biti zadovoljeno da bi događaje u intervalu smatrali značajnima. Vremenski interval je raspon u kojem pokušavamo naći minimalni prag podudaranja s pravilom. Odabrano svojstvo zapisa određuje koju vrijednost zapisa koristimo za testiranje s regularnim izrazom. Svojstvo može biti poruka događaja ili neko od vlastitih svojstava opisanog od strane korisnika za format dnevnika komponente nad kojom pokušavamo pronaći značajne vremenske intervale. Rezultat pronalaženja značajnih vremenskih intervala je vrijeme prvog zapisa u intervalu i vrijeme posljednjeg zapisa u intervalu.

Postoje dvije varijante prepoznavanja značajnih vremenskih intervala.

- Prva varijanta samo zahtjeva da je regularni izraz pravila pronašao podudaranje s vrijednošću svojstva zapisa, ali vrijednosti ne moraju biti konstantne između zapisa u intervalu.
- Druga varijanta vremenski interval smatra značajnim samo kada je minimalni broj pojavljivanja zadovoljen istim vrijednostima u zadanom vremenskom intervalu.

Pravilo definirano u isječku koda 5.1 primijenjeno u varijanti koja ne zahtjeva konstantnost vrijednosti na primjer zapisa u isječku koda 5.2 pronalazi značajne vremenske intervale [2020-02-04 00:25:39.214, 2020-02-04 00:25:42.232] i [2020-02-04 00:29:22.044, 2020-02-04 00:29:23.123] dok bi isto pravilo primijenjeno u varijanti koja zahtjeva konstantnost vrijednosti pronašla samo interval [2020-02-04 00:25:39.214, 2020-02-04 00:25:42.232].

```
Regularni izraz: Login failed for user with id [\d]+$
Prag pojavljivanja: 3
Vremenski interval: 30s
Svojstvo zapisa: Poruka
```

Isječak koda 5.1 Primjer pravila značajnog događaja

```
2020-02-04 00:25:39.214 +01:00 [INF] Login failed for user with id 1
2020-02-04 00:25:41.671 +01:00 [INF] Login failed for user with id 1
2020-02-04 00:25:42.232 +01:00 [INF] Login failed for user with id 1

2020-02-04 00:29:22.044 +01:00 [INF] Login failed for user with id 0
2020-02-04 00:29:22.584 +01:00 [INF] Login failed for user with id 1
2020-02-04 00:29:23.123 +01:00 [INF] Login failed for user with id 2
```

Isječak koda 5.2 Primjer dnevnika komponente

**Algoritam za određivanje značajnih vremenskih intervala** provodi se nad skupom normaliziranih zapisa dnevnika jedne komponente. Algoritam uzima u obzir mogućnost da su zapisi ranijih i kasnijih događaja od danog skupa potencijalno dostupni. Pokušati će pronaći značajne vremenske intervale na prijelazima trenutačnog skupa zapisa i od prije poznatih zapisa. Ako je *prviDogađaj* vrijeme najranijeg zapisa danog skupa zapisa, a *posljednjiDogađaj* vrijeme najkasnijeg zapisa danog skupa zapisa u skup zapisa uključujemo sve događaje koji se nalaze u vremenskom rasponu [*prviDogađaj* – *vremenski interval pravila*, *posljednjiDogađaj* + *vremenski interval pravila*]. Algoritam u svakom trenutku razmatra vremenski okvir koji je manji ili jednak vremenskom intervalu pravila. Ako je unutar vremenskog okvira otkriven dovoljan broj zapisa koji se podudaraju s pravilom

algoritam stvara novi značajni vremenski interval za to pravilo, resetira brojač i nastavlja od sljedećeg zapisa. Ukoliko dovoljan broj odgovarajućih zapisa nije pronađen početak vremenskog okvira pomiče se na sljedeći zapis koji odgovara pravilu te algoritam nastavlja s radom. Pseudokod varijante algoritma za nekonstantne vrijednosti dan je u isječku koda 5.3. Varijabla *brojDogađaja* je brojač podudarajućih zapisa u vremenskom okviru koji razmatramo, *početakIntervala* označava prvi podudarajući zapis i on je početno vrijeme vremenskog okvira unutar kojeg brojimo podudarajuće zapise, kraj vremenskog okvira uvijek je vrijeme trenutnog zapisa. Pseudokod varijante za konstantne vrijednosti dan je u isječku koda 5.4. Primijenjena je ista ideja samo u ovom slučaju imamo istovremeno više aktivnih vremenskih okvira. Varijabla *brojVrijednosti* je rječnik vrijednosti i broj pojavljivanja te vrijednosti u njenom vremenskom okviru. Varijabla *početnoVrijemeVrijednosti* je rječnik vrijednosti i vremena prvog pojavljivanja te vrijednosti. Vrijeme prvog pojavljivanja označava početak vremenskog okvira za tu vrijednost. Vremenski okviri za sve vrijednosti imaju isti kraj, vrijeme trenutnog zapisa,

```

Inicijaliziraj brojDogađaja na 0
Inicijaliziraj početakIntervala na praznu vrijednost

Za svaki zapis u proširenom skupu zapisa:
    Ako se zapis podudara s pravilom:
        Uvećaj brojDogađaja za 1
    Ako početakIntervala ima praznu vrijednost:
        Postavi početakIntervala na vrijeme zapisa
    Ako vrijeme zapisa - početakIntervala veće od intervala pravila:
        Postavi početakIntervala na vrijeme prvog sljedećeg zapisa
        koji odgovara pravilu
        Umanji brojDogađaja za 1
    Ako brojDogađaja veći ili jednak minimalnom pragu pravila:
        Stvori novi značajni interval
        Postavi početakIntervala na praznu vrijednost
        Postavi brojDogađaja na 0

```

Isječak koda 5.3 Algoritam značajnih vremenskih intervala nekonstantne vrijednosti

```

Inicijaliziraj rječnik brojVrijednosti
Inicijaliziraj rječnik početnoVrijemeVrijednosti

Za svaki zapis u proširenom skupu zapisa:
    Ako se zapis podudara s pravilom:
        Uvećaj brojVrijednosti[vrijednost] za 1
    Ako početnoVrijemeVrijednosti[vrijednost] ima praznu vrijednost:
        Postavi početnoVrijemeVrijednosti[vrijednost] na vrijeme
        zapisa
    Ako vrijeme zapisa - početnoVrijemeVrijednosti[vrijednost] veće od
    intervala pravila:
        Postavi početnoVrijemeVrijednosti[vrijednost] na vrijeme
        prvog sljedećeg zapisa iste vrijednosti koji odgovara pravilu
        Umanji brojVrijednosti[vrijednost] za 1
    Ako brojVrijednosti[vrijednost] veći ili jednak minimalnom pragu
    pravila:
        Stvori novi značajni interval za vrijednost
        Postavi početnoVrijemeVrijednosti[vrijednost] na praznu
        vrijednost
        Postavi brojVrijednosti[vrijednost] na 0

```

Isječak koda 5.4 Algoritam za značajne vremenske intervale konstante vrijednosti

## 5.2. Drugi korak

**Drugi korak** korelacije događaja obavlja se na razini cijelog sustava i obrađuje rezultate **algoritma za određivanje značajnih vremenskih intervala** provedenog nad zapisima svake komponente sustava. **Algoritam za stvaranje korelacijskih grupa** radi sa značajnim vremenskim intervalima svih komponenti povezanih u jedan sustav. Korisnik zadaje pravilo za korelacijsku grupu koje povezuje značajne vremenske intervale i određuje redoslijed u kojem se moraju pojaviti. Korelacijska grupa također ima vremenski interval i stvoriti će se samo ako je razlika kraja zadnjeg i početka prvog značajnog vremenskog intervala grupe manja od zadanog vremenskog intervala. Pravilo za stvaranje korelacijske grupe sastoji se od poredane liste pravila značajnih vremenskih intervala i vremenskog okvira. Algoritam primjenjuje određene heuristike u pokušaju stvaranja točnijih grupa. Prva heuristika je da se značajni vremenski interval pridjeljuje samo jednoj grupi. Ovaj pristup sprječava da broj značajnih vremenskih intervala konstanto raste kako novi zapisi pristižu u sustav. Druga heuristika je da će nakon izgradnje liste kandidata grupa algoritam stvoriti prvo grupe čije je ukupno trajanje najkraće. Ideja ovog pristupa je da događaji koji su vremenski bliži jedan

drugome imaju veću šansu zapravo biti korelirani. Pseudokod algoritma dan je u isječku koda 5.5. Pomoćne strukture algoritma su *listaPravila*, *kandidati*, *listaIskorištenihIntervala*. *listaPravila* je poredana lista prema kojoj se određuje koji je redoslijed značajnih vremenskih intervala unutar korelacijske grupe. *kandidati* je lista lista koja sadrži ispravne redoslijede značajnih vremenskih intervala i potencijalno će se od njih stvoriti grupa. Grupa se stvara ako se zadovolje uvjeti da je ukupno trajanje grupe kraće od vremenskog intervala zadanog u pravilu korelacijske grupe i ako niti jedan značajni vremenski interval nije pridijeljen grupi tijekom rada algoritma. *listaIskorištenihIntervala* nam služi kako bi pratili koji su značajni vremenski intervali već pridijeljeni grupi tijekom rada algoritma i sprječava nas da jedan značajni interval pridijelimo dvije grupe.

```
Inicijaliziraj listaPravila s listom pravila
Inicijaliziraj kandidati pridjeljujući im značajni vremenski interval
prvog pravila listePravila koji nije pridijeljen grupi
Inicijaliziraj listaIskorištenihIntervala

Za svako pravilo iz listaPravila:
    Svakom kandidatu pridijeli značajni vremenski interval pravila koji
    se pojavljuje prvi nakon zadnjeg vremenskog intervala u tom
    kandidatu
Za svakog kandidata poredanog prema trajanju:
    Ako kandidat sadrži značajne vremenske intervale za svako pravilo
    Ako kandidat ne sadrži vremenski interval iz
    listaIskorištenihIntervala i trajanje kandidata je kraće od
    vremenskog intervala pravila grupe
        Stvori korelacijsku grupu
        Dodaj iskorištene vremenske intervale u
        listaIskorištenihIntervala
```

Isječak koda 5.5 Algoritam za stvaranje korelacijskih grupa

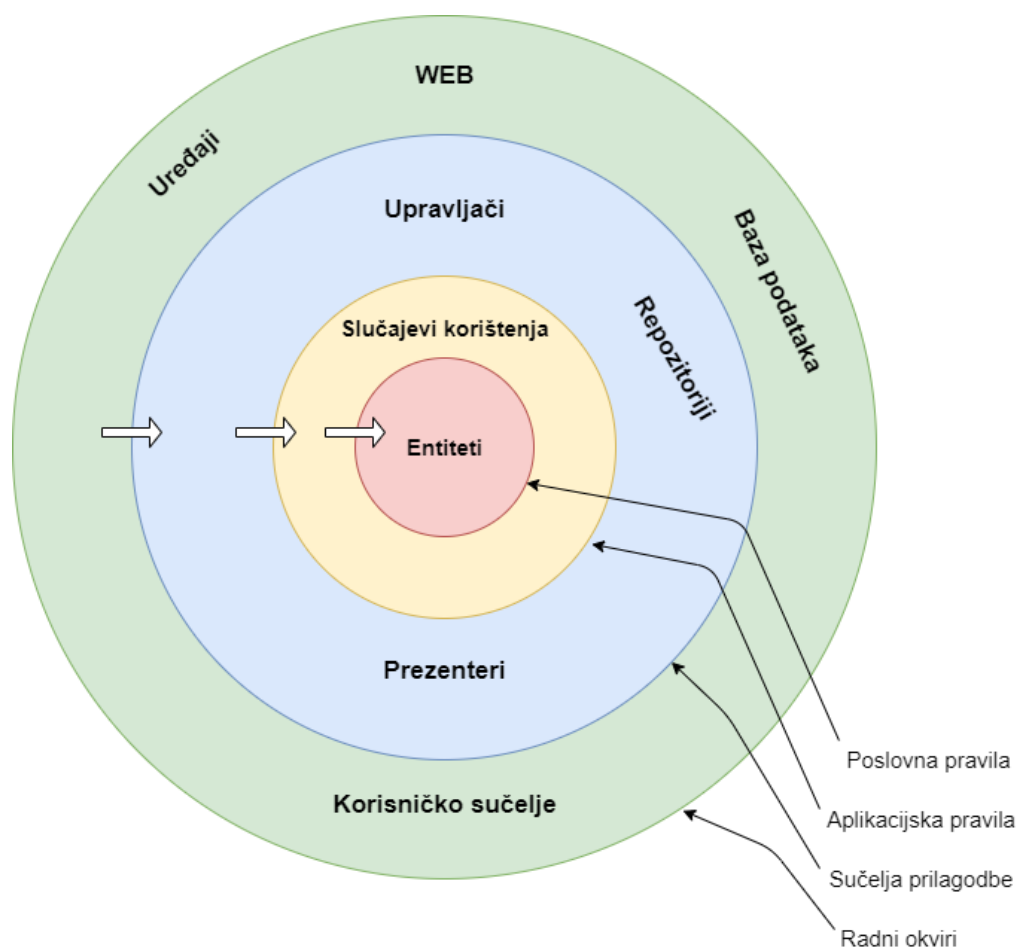


## 6. Arhitektura sustava

Arhitektura programske podrške je dizajn sustava visoke razine te opisuje osnovni strukturni radni okvir unutar kojeg identificiramo glavne komponente sustava i komunikaciju između tih komponenti. Arhitektura utječe na svojstva sustava poput performansi, robusnosti, raspodijeljenosti i održivosti [3]. Prilikom razvoja arhitekture sustava korišten je pristup čiste arhitekture (engl. *clean architecture*). Glavni cilj čiste arhitekture je odvajanje briga (engl. *separation of concerns*) odnosno izolacija domene sustava od vanjskih radnih okvira, tehnologija i knjižnica. Izolacija domene postiže se odvajanjem komponenti programske podrške u slojeve. Principi čiste arhitekture su:

- Nezavisnost o radnim okvirima. Arhitektura ne ovisi o postojanju vanjske knjižnice ili mogućnostima vanjske programske podrške.
- Ispitljivost (engl. *testability*). Poslovna pravila mogu se testirati bez korisničkog sučelja, baze podataka, web poslužitelja ili bilo koje druge vanjske komponente.
- Nezavisnost o korisničkom sučelju. Korisničko sučelje je lako promjenjivo bez mijenjanja ostatka sustava.
- Nezavisnost o bazi podataka. Poslovna pravila nisu vezana uz jednu bazu podataka [7].

Čista arhitektura razvrstava komponente u četiri razine apstrakcije. Najapstraktnija razina su entiteti (engl. *entities*) i oni enkapsuliraju poslovna pravila i poslovne objekte sustava. Entiteti predstavljaju općenita pravila visoke razine koja se najmanje mijenjanju kao posljedica vanjske promijene. Sljedeća razina apstrakcije su slučajevi korištenja (engl. *use cases*). Slučajevi korištenja sadrže aplikacijska pravila i upravljaju tokom podataka iz entiteta i u entitete. Promijene u ovom sloju ne uzrokuju promjenu u entitetima. Također promijene u vanjskim komponentama ne uzrokuju promijene na ovoj razini apstrakcije. Sučelja prilagodbe (engl. *interface adapters*) je sloj koji sadrži pretvarače (engl. *adapters*) formata podataka. Pretvarači uzimaju podatke u obliku koji je najpogodniji za rad entitetima i slučajevima korištenja te ih pretvaraju u oblik koji odgovara bazi podataka ili korisničkom sučelju. Radni okviri i uređaji su posljednji sloj apstrakcije i on se sastoji od radnih okvira i alata poput baze podataka ili web radnih okvira. Glavno pravilo čiste arhitekture je pravilo ovisnosti (engl. *dependency rule*). Pravilo ovisnosti nam govori da ovisnosti izvornog koda moraju teći od najmanje apstraktnog sloja prema najapstraktnijem sloju [7]. Slojevi čiste arhitekture i smjer ovisnosti prikazan je na slici 6.1.



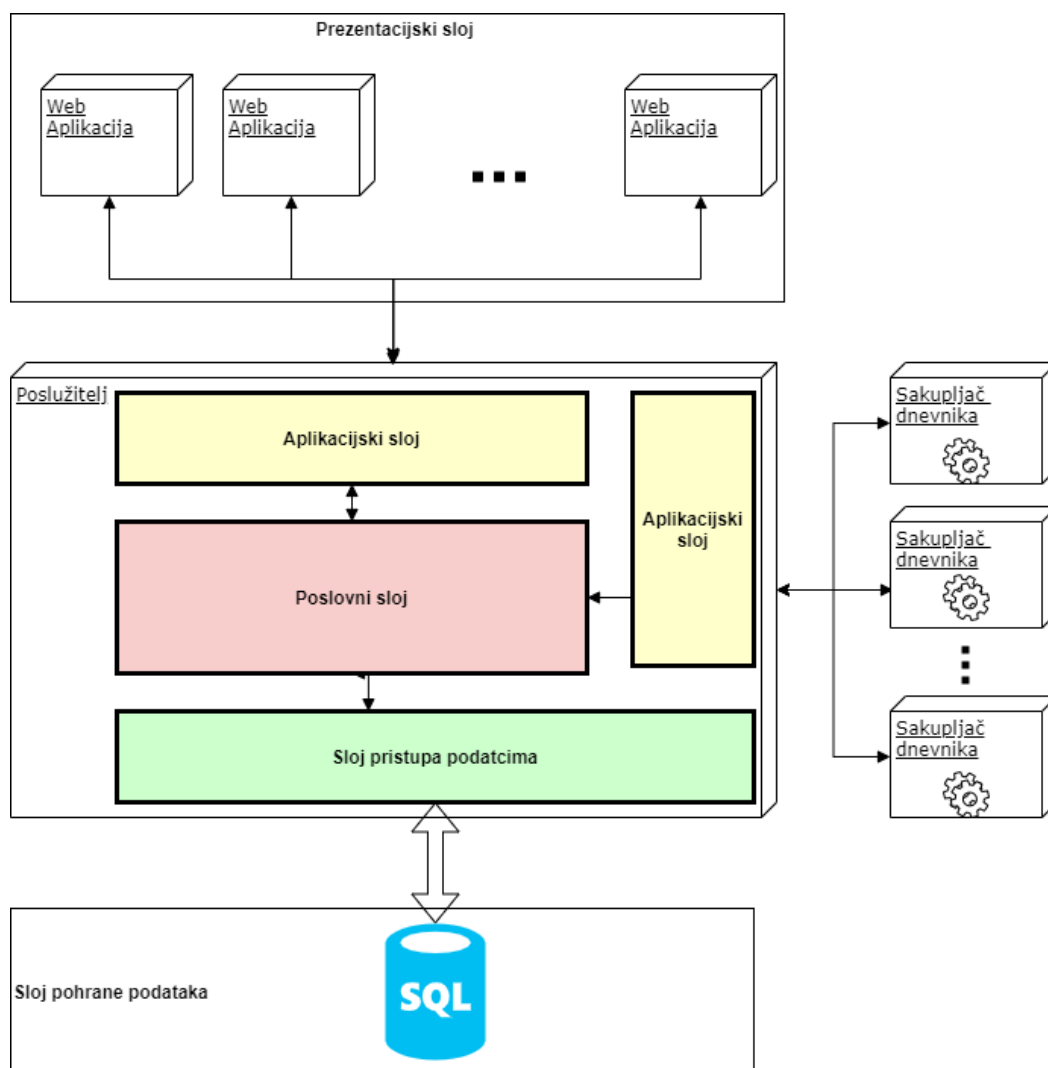
Slika 6.1 Slojevi čiste arhitekture i pravilo ovisnosti [7]

## 6.1. Struktura sustava

Razvijeni sustav ostvaren je prema modelu klijent-poslužitelj. Klijent-poslužitelj model arhitekture je model sustava gdje je sustav organiziran kao skup usluga i asociranih poslužitelja i klijenata koji pristupaju i koriste servise. Dijelovi ovog modela su poslužitelj, skup klijenata i mreža. Poslužitelj pruža usluge drugim komponentama sustava. Skup klijenata poziva poslužitelj kako bi konzumirali ponuđene usluge. Mreža omogućava komunikaciju između klijenata i poslužitelja [3]. Arhitektura sustava prikazana je na slici 6.2. Sastoji se od jednog poslužitelja i dvije vrste klijenata. Prvi klijent je web aplikacija koja implementira korisničko sučelje. Drugi klijent je servis za prikupljanje dnevnika. Sustav možemo prikazati pomoću sljedećih slojeva:

- Sloj pohrane podataka. Podatci se pohranjuju u relacijsku bazu podataka.
- Sloj pristupa podacima. Sloj koji je zadužen za komunikaciju s bazom podataka te za stvaranje, uređivanje, brisanje i čitanje podataka.
- Poslovni sloj. Sloj koji sadrži poslovnu logiku sustava.

- Aplikacijski sloj. Sloj sustava zadužen za komunikaciju poslovnog sloja i prezentacijskog sloja.
- Prezentacijski sloj. Sloj koji sadrži logiku prikaza podataka na korisničkom sučelju.

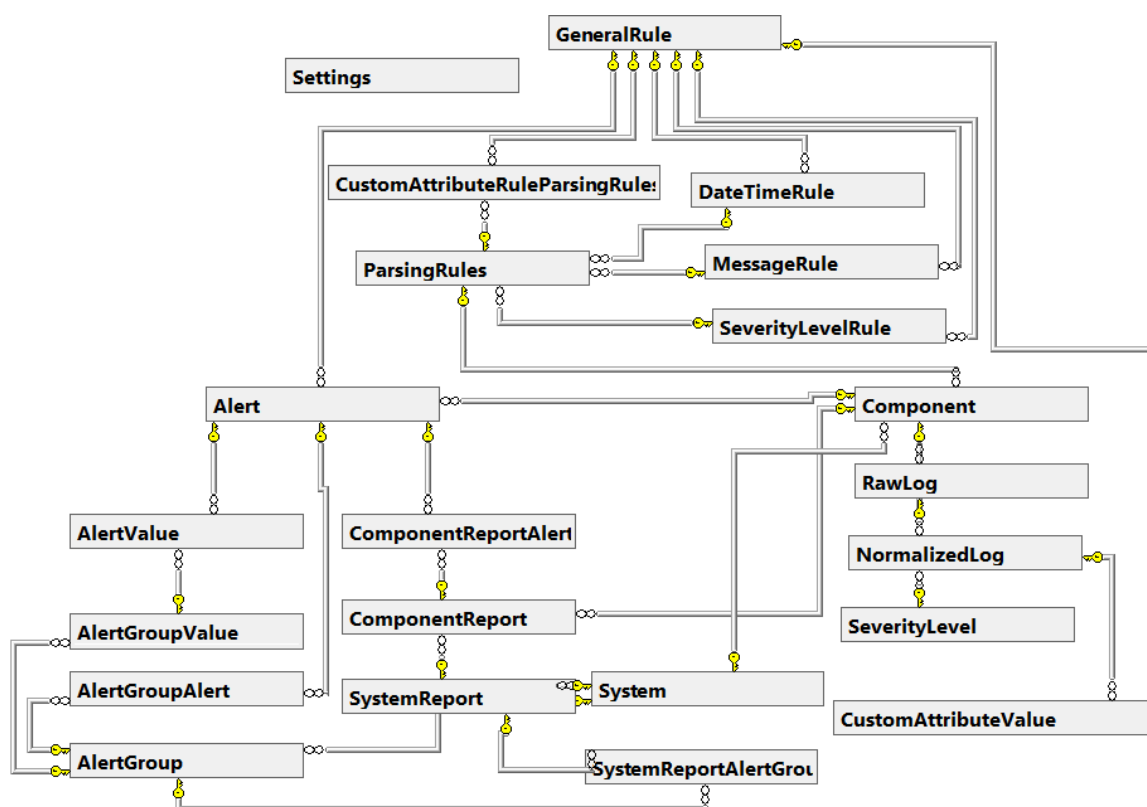


Slika 6.2 Arhitektura sustava

## 6.2. Model podataka

Stanje aplikacije pohranjeno je u bazu podataka. Odabrana je relacijska baza podataka jer se model podataka može opisati vezama između entiteta. Kao sustav baze podataka izabran je Microsoft SQL Server.

Model svih tablica baze podataka i njihovih veza prikazan je na slici 6.3. Zbog preglednosti tablice su prikazane samo svojim nazivom. U ostatku poglavlja opisuju se tablice baze podataka u detalje.



Slika 6.3 Model baze podataka

## Tablica Alert

Tablica Alert prikazana na slici 6.4 modelira pravila za otkrivanje značajnih vremenskih intervala unutar dnevnika komponente. Atributi tablice Alert su:

- AlertId – primarni ključ tablice.
- Name – ime pravila.
- Description – opis pravila.
- Timespan – vremenski raspon u sekundama.
- Threshold – minimalni prag pojavljivanja događaja.
- Value – regularni izraz prema kojem se određuje odgovara li događaj pravilu.
- GeneralRuleId – strani ključ na pravilo koje je korišteno za stvaranje vrijednosti iz zapisa dnevnika. Vrijednost stvorena ovim pravilom će biti uspoređena s vrijednošću pravila.
- HasConstantValue – oznaka zahtjeva li pravilo pojavljivanje istih vrijednosti kako bi bilo zadovoljeno ili je dovoljno da vrijednosti samo odgovaraju regularnom izrazu pravila, a međusobno se mogu razlikovati.
- ComponentId – strani ključ na komponentu za koju se primjenjuje ovo pravilo.

	Column Name	Data Type	Allow Nulls
▶	AlertId	int	<input type="checkbox"/>
	Name	nvarchar(MAX)	<input checked="" type="checkbox"/>
	Description	nvarchar(MAX)	<input checked="" type="checkbox"/>
	Timespan	bigint	<input type="checkbox"/>
	Threshold	int	<input type="checkbox"/>
	Value	nvarchar(MAX)	<input type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>
	HasConstantValue	bit	<input type="checkbox"/>
	ComponentId	int	<input type="checkbox"/>
			<input type="checkbox"/>

Slika 6.4 Tablica Alert

## Tablica AlertGroup

Tablica AlertGroup prikazana je na slici 6.5. Ova tablica modelira pravilo za stvaranje korelacijskih grupa. Atributi tablice AlertGroup su:

- AlertGroupId – primarni ključ tablice.
- Name – ime pravila za stvaranje korelacijskih grupa.
- Description – opis pravila za stvaranje korelacijskih grupa.
- CorrelationWindow – vremenski raspon u sekundama kojim se određuje maksimalno trajanje korelacijske grupe.
- SystemId – strani ključ na sustav za koji se primjenjuje pravilo korelacijskih grupa.

	Column Name	Data Type	Allow Nulls
▶	AlertGroupId	int	<input type="checkbox"/>
	Name	nvarchar(MAX)	<input checked="" type="checkbox"/>
	Description	nvarchar(MAX)	<input checked="" type="checkbox"/>
	CorrelationWindow	bigint	<input type="checkbox"/>
	SystemId	int	<input type="checkbox"/>
			<input type="checkbox"/>

Slika 6.5 Tablica AlertGroup

## Tablica AlertGroupValue

Tablica AlertGroupValue prikazana je na slici 6.6. Tablica modelira stvorenu korelacijsku grupu. Atributi tablice AlertGroupValue su:

- AlertGroupValueId – primarni ključ tablice.

- AlertGroupId – strani ključ na pravilo korelacijskih grupa prema kojem je stvorena korelacijska grupa.
- TimespanStart – početno vrijeme prvog značajnog vremenskog intervala koji se nalazi u grupi.
- TimespanEnd – završno vrijeme posljednjeg značajnog vremenskog intervala koji se nalazi u grupi.

	Column Name	Data Type	Allow Nulls
▶	AlertGroupValueId	int	<input type="checkbox"/>
	AlertGroupId	int	<input type="checkbox"/>
	TimespanStart	datetime	<input type="checkbox"/>
	TimespanEnd	datetime	<input type="checkbox"/>
			<input type="checkbox"/>

Slika 6.6 Tablica AlertGroupValue

## Tablica AlertGroupAlert

Tablica AlertGroupAlert prikazana je na slici 6.7. Ova tablica je spojna tablica između AlertGroup tablice i Alert tablice te s njome ostvarujemo više na više vezu (engl. *many to many*). Veza između ove dvije tablice označava koja pravila značajnih vremenskih intervala i kojim redoslijedom čine pravilo korelacijske grupe. Atributi tablice su:

- AlertGroupId – strani ključ na tablicu pravila korelacijske grupe. Dio je kompozitnog primarnog ključa tablice.
- AlertId – strani ključ na pravilo značajnih vremenskih intervala. Dio kompozitnog primarnog ključa.
- Position – određuje poziciju pravila značajnog vremenskog intervala unutar korelacijske grupe.


	Column Name	Data Type	Allow Nulls
▶	AlertGroupId	int	<input type="checkbox"/>
▶	AlertId	int	<input type="checkbox"/>
	Position	int	<input type="checkbox"/>

Slika 6.7 Tablica AlertGroupAlert

## Tablica AlertValue

Tablica AlertValue prikazana je na slici 6.8. Tablica modelira pronađeni značajni vremenski interval. Atributi tablice su:

- AlertValueId – primarni ključ tablice.
- AlertId – strani ključ na pravilo značajnog vremenskog intervala prema kojem je stvoren ovaj interval.
- ConstantValue – neobavezan atribut koji sadrži vrijednost ukoliko je pravilo značajnog vremenskog intervala zahtijevalo konstante vrijednosti, tada je konstanta vrijednost prema kojoj je stvoren interval vrijednost ovog atributa.
- TimespanStart – vrijeme početka intervala.
- TimespanEnd – vrijeme kraja intervala.
- AlertGroupValueId – strani ključ na korelacijsku grupu kojoj ovaj interval pripada.

	Column Name	Data Type	Allow Nulls
	AlertValueId	int	<input type="checkbox"/>
	AlertId	int	<input type="checkbox"/>
	ConstantValue	nvarchar(MAX)	<input checked="" type="checkbox"/>
	TimespanStart	datetime	<input type="checkbox"/>
	TimespanEnd	datetime	<input type="checkbox"/>
	AlertGroupValueId	int	<input checked="" type="checkbox"/>
			<input type="checkbox"/>

Slika 6.8 Tablica AlertValue

## Tablica Component

Tablica Component prikazana je na slici 6.9. Ova tablica modelira komponentu sustava. Atributi tablice su:

- ComponentId – primarni ključ tablice.
- ComponentName – naziv komponente.
- ComponentDescription – opis komponente.
- SystemId – strani ključ na sustav kojem ova komponenta pripada.
- ParsingRulesId – strani ključ na grupu pravila prema kojima se obrađuje i normalizira dnevnik ove komponente.

	Column Name	Data Type	Allow Nulls
PK	ComponentId	int	<input type="checkbox"/>
	ComponentName	varchar(256)	<input type="checkbox"/>
	ComponentDescription	nvarchar(MAX)	<input checked="" type="checkbox"/>
	SystemId	int	<input checked="" type="checkbox"/>
	ParsingRulesId	int	<input checked="" type="checkbox"/>

Slika 6.9 Tablica Component

## Tablica ComponentReport

Tablica ComponentReport prikazana je na slici 6.10. Ova tablica modelira izvještaj učestalosti tipova značajnih vremenskih intervala za komponentu. Atributi tablice su:

- ComponentReportId – primarni ključ tablice.
- ComponentId – strani ključ na komponentu.
- SystemReportId – strani ključ na izvještaj sustava.
- Timestamp – vrijeme izrade izvještaja.

	Column Name	Data Type	Allow Nulls
PK	ComponentReportId	int	<input type="checkbox"/>
	ComponentId	int	<input type="checkbox"/>
	SystemReportId	int	<input type="checkbox"/>
	Timestamp	datetime	<input type="checkbox"/>


Slika 6.10 Tablica ComponentReport

## Tablica ComponentReportAlert

Tablica ComponentReportAlert prikazana je na slici 6.11. Ova tablica modelira statistiku pojavljivanja jednog tipa značajnog vremenskog intervala u određenom izvještaju komponente. Atributi tablice su:

- ComponentReportAlertId – primarni ključ tablice.
- AlertId – strani ključ na pravilo značajnog vremenskog intervala.
- Count – broj pronađenih intervala prema zadanom pravilu.
- ComponentReportId – strani ključ na izvještaj komponente.





	Column Name	Data Type	Allow Nulls
	ComponentReportAlertId	int	<input type="checkbox"/>
	AlertId	int	<input type="checkbox"/>
	Count	int	<input type="checkbox"/>
	ComponentReportId	int	<input type="checkbox"/>

Slika 6.11 Tablica ComponentReportAlert

## Tablica CustomAttributeRuleParsingRules

Tablica CustomAttributeRuleParsingRules prikazana je na slici 6.12. Spojna tablica modelira više na više vezu između općenitog pravila obrade zapisa dnevnika i grupe pravila obrade zapisa dnevnika. Atributi tablice su:

- ParsingRulesId - strani ključ na grupu pravila za obradu dnevnika. Dio kompozitnog primarnog ključa tablice.
- GeneralRuleId – strani ključ na pravilo obrade zapisa dnevnika. Dio kompozitnog primarnog ključa tablice.

	Column Name	Data Type	Allow Nulls
	ParsingRulesId	int	<input type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>

Slika 6.12 Tablica CustomAttributeRuleParsingRules

## Tablica CustomAttributeValue

Tablica CustomAttributeValue prikazana je na slici 6.13. Tablica modelira dobivenu vrijednost iz zapisa dnevnika primjenom pravila za obradu zapisa dnevnika. Atributi tablice su:

- CustomAttributeValueId – primarni ključ tablice.
- Value – vrijednost dobivena primjenom pravila.
- NormalizedLogId – strani ključ na normalizirani zapis.
- GeneralRuleId – strani ključ na pravilo čijom primjenom je dobivena vrijednost.

	Column Name	Data Type	Allow Nulls
PK	CustomAttributeValueId	int	<input type="checkbox"/>
	Value	varchar(2048)	<input type="checkbox"/>
	NormalizedLogId	int	<input type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>

Slika 6.13 Tablica CustomAttributeValue

## Tablica GeneralRule

Tablica GeneralRule prikazana je na slici 6.14. Tablica modelira pravilo za obradu zapisa dnevnika. Atributi tablice su:

- GeneralRuleId – primarni ključ tablice.
- GeneralRuleName – naziv pravila obrade zapisa dnevnika.
- GeneralRuleDescription – opis pravila obrade zapisa dnevnika.
- GeneralRuleMatcher – regularni izraz čijom primjenom na zapis dnevnika dobivamo vrijednost pravila.
- GeneralRuleStartAnchor – početni niz znakove u zapisu dnevnika. Dio zapisa prije početnog niza se ignorira.
- GeneralRuleEndAnchor – završni niz znakova u zapisu dnevnika. Dio zapisa dnevnika nakon posljednjeg pojavljivanja ovog niza se ignorira.

	Column Name	Data Type	Allow Nulls
PK	GeneralRuleId	int	<input type="checkbox"/>
	GeneralRuleName	varchar(512)	<input type="checkbox"/>
	GeneralRuleDescription	varchar(4096)	<input type="checkbox"/>
	GeneralRuleMatcher	varchar(2048)	<input type="checkbox"/>
	GeneralRuleStartAnchor	varchar(1024)	<input checked="" type="checkbox"/>
	GeneralRuleEndAnchor	varchar(1024)	<input checked="" type="checkbox"/>

Slika 6.14 Tablica GeneralRule

## Tablica DateTimeRule

Tablica DateTimeRule prikazana je na slici 6.15. Ova tablica je specijalizacija tablice GeneralRule i modelira pravilo za obradu vremena iz zapisa dnevnika. Atributi tablice su:

- DateTimeRuleId – primarni ključ tablice.
- GeneralRuleId – strani ključ na tablicu pravila obrade zapisa dnevnika.

- DateTimeFormat – format u kojem se vrijednost dobivena primjenom pravila interpretira kao vremenski trenutak.

	Column Name	Data Type	Allow Nulls
▶	DateTimeRuleId	int	<input type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>
	DateTimeFormat	varchar(256)	<input type="checkbox"/>

Slika 6.15 Tablica DateTimeRule

## Tablica MessageRule

Tablica MessageRule prikazana je na slici 6.16. Tablica je specijalizacija GeneralRule tablice i moderira pravilo za obradu poruke događaja iz zapisa dnevnika. Atributi tablice su:

- MessageRuleId – primarni ključ tablice.
- MaxLength – maksimalna dužina poruke koja se uzima kao vrijednost pravila.
- GeneralRuleId – strani ključ na tablicu pravila obrade zapisa dnevnika.

	Column Name	Data Type	Allow Nulls
▶	MessageRuleId	int	<input type="checkbox"/>
	MaxLength	int	<input checked="" type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>

Slika 6.16 Tablica MessageRule

## Tablica SeverityLevelRule

Tablica SeverityLevelRule prikazana je na slici 6.17. Tablica je specijalizacija tablice GeneralRule i modelira pravilo za obradu ozbiljnosti događaja iz zapisa dnevnika. Atributi tablice su:

- SeverityLevelRuleId – primarni ključ tablice.
- GeneralRuleId - strani ključ na tablicu pravila obrade zapisa dnevnika.
- Trace – vrijednost koja određuje radi li se o Trace razini ozbiljnosti.
- Debug - vrijednost koja određuje radi li se o Debug razini ozbiljnosti.
- Info - vrijednost koja određuje radi li se o Info razini ozbiljnosti.
- Warning - vrijednost koja određuje radi li se o Warning razini ozbiljnosti.
- Error - vrijednost koja određuje radi li se o Error razini ozbiljnosti.
- Fatal - vrijednost koja određuje radi li se o Fatal razini ozbiljnosti.

	Column Name	Data Type	Allow Nulls
▶ 🔑	SeverityLevelRuleId	int	<input type="checkbox"/>
	GeneralRuleId	int	<input type="checkbox"/>
	Trace	varchar(256)	<input type="checkbox"/>
	Debug	varchar(256)	<input type="checkbox"/>
	Info	varchar(256)	<input type="checkbox"/>
	Warning	varchar(256)	<input type="checkbox"/>
	Error	varchar(256)	<input type="checkbox"/>
	Fatal	varchar(256)	<input type="checkbox"/>

Slika 6.17 Tablica SeverityLevelRule

## Tablica SeverityLevel

Tablica SeverityLevel prikazana je na slici 6.18. Tablica je šifarnik koji sadrži vrijednosti razina ozbiljnosti. Atributi tablice su:

- SeverityLevelId – primarni ključ tablice.
- SeverityLevelValue – vrijednost razine ozbiljnosti događaja.


	Column Name	Data Type	Allow Nulls
▶ 🔑	SeverityLevelId	int	<input type="checkbox"/>
	SeverityLevelValue	nvarchar(MAX)	<input checked="" type="checkbox"/>

Slika 6.18 Tablica SeverityLevel

## Tablica ParsingRules

Tablica ParsingRules prikazana je na slici 6.19. Tablica modelira grupu pravila obrade zapisa dnevnika prema kojima se zapis dnevnika pretvara u normalizirani oblik. Atributi tablice su:

- ParsingRulesId – primarni ključ tablice.
- DateTimeRuleId – strani ključ na pravilo za obradu vremena događaja iz zapisa dnevnika.
- MessageRuleId – strani ključ na pravilo za obradu poruke događaja iz zapisa dnevnika.
- SeverityLevelRuleId – strani ključ na pravilo za obradu ozbiljnosti zapisa događaja iz zapisa dnevnika.

	Column Name	Data Type	Allow Nulls
	ParsingRulesId	int	<input type="checkbox"/>
	DateTimeRuleId	int	<input type="checkbox"/>
	MessageRuleId	int	<input type="checkbox"/>
	SeverityLevelRuleId	int	<input type="checkbox"/>


Slika 6.19 Tablica ParsingRules

## Tablica RawLog

Tablica RawLog prikazana je na slici 6.20. Tablica modelira neobrađeni zapis dnevnika.

Atributi tablice su:

- RawLogId – primarni ključ tablice.
- Message – cijeli zapis iz dnevnika.
- ComponentId – strani ključ na komponentu čijem dnevniku pripada ovaj zapis.

	Column Name	Data Type	Allow Nulls
	RawLogId	int	<input type="checkbox"/>
	Message	varchar(4096)	<input type="checkbox"/>
	ComponentId	int	<input type="checkbox"/>

Slika 6.20 Tablica RawLog

## Tablica NormalizedLog

Tablica NormalizedLog prikazana je na slici 6.21. Tablica modelira normalizirani zapis dnevnika. Atributi tablice su:

- NormalizedLogId – primarni ključ tablice.
- RawLogId – strani ključ na neobrađeni zapis.
- ComponentId – strani ključ na komponentu čijem dnevniku pripada normalizirani zapis.
- Message – vrijednost poruke događaja dobivene primjeno pravila za obradu poruke događaja.
- Timestamp – vrijednost vremena događaja dobivena primjenom pravila za obradu vremena događaja.
- SeverityLevelId – strani ključ na šifrarnik razine ozbiljnosti.

	Column Name	Data Type	Allow Nulls
▶	NormalizedLogId	int	<input type="checkbox"/>
	RawLogId	int	<input type="checkbox"/>
	ComponentId	int	<input type="checkbox"/>
	Message	varchar(2048)	<input type="checkbox"/>
	Timestamp	datetime	<input type="checkbox"/>
	SeverityLevelId	int	<input type="checkbox"/>

Slika 6.21 Tablica NormalizedLog

## Tablica Settings

Tablica Settings prikazana je na tablici 6.22. Tablica Settings modelira postavku učestalosti izvršavanja analizatora. Atributi tablice su:

- SettingsId – primarni ključ tablice.
- Name – naziv postavke.
- Frequency – broj sekundi koliko analizator čeka ponovno izvršavanje nakon završetka prošlog izvršavanja.

	Column Name	Data Type	Allow Nulls
▶	SettingsId	int	<input type="checkbox"/>
	Name	nvarchar(MAX)	<input checked="" type="checkbox"/>
	Frequency	bigint	<input type="checkbox"/>

Slika 6.22 Tablica Settings

## Tablica System

Tablica System prikazana je na slici 6.23. Tablica modelira jedan informacijski sustav. Atributi tablice su:

- SystemId – primarni ključ tablice.
- SystemName – naziv informacijskog sustava.
- SystemDescription – opis informacijskog sustava.

	Column Name	Data Type	Allow Nulls
▶	SystemId	int	<input type="checkbox"/>
	SystemName	varchar(256)	<input type="checkbox"/>
	SystemDescription	varchar(2048)	<input type="checkbox"/>

Slika 6.23 Tablica System

## Tablica SystemReport

Tablica SystemReport prikazana je na slici 6.24. Tablica modelira izvještaj sustava koji sadrži učestalost pojavljivanja tipova korelacijskih grupa za sustav. Atributi tablice su:

- SystemReportId – primarni ključ tablice.
- SystemId – strani ključ sustava za koji je izvještaj generiran.
- Timestamp – vrijeme generiranja izvještaja.

	Column Name	Data Type	Allow Nulls
PK	SystemReportId	int	<input type="checkbox"/>
	SystemId	int	<input type="checkbox"/>
	Timestamp	datetime	<input type="checkbox"/>

Slika 6.24 Tablica SystemReport

## Tablica SystemReportAlertGroup

Tablica SystemReportAlertGroup prikazana je na slici 6.25. Tablica modelira učestalost tipa korelacijske grupe za određeni izvještaj sustava. Atributi tablice su:

- SystemReportAlertGroupId – primarni ključ tablice.
- SystemReportId – strani ključ na izvještaj sustava.
- AlertGroupId – strani ključ na pravilo korelacijske grupe čiju učestalost prebrojavamo.
- Count – učestalost tipa korelacijske grupe.

	Column Name	Data Type	Allow Nulls
PK	SystemReportAlertGroupId	int	<input type="checkbox"/>
	SystemReportId	int	<input type="checkbox"/>
	AlertGroupId	int	<input type="checkbox"/>
	Count	int	<input type="checkbox"/>

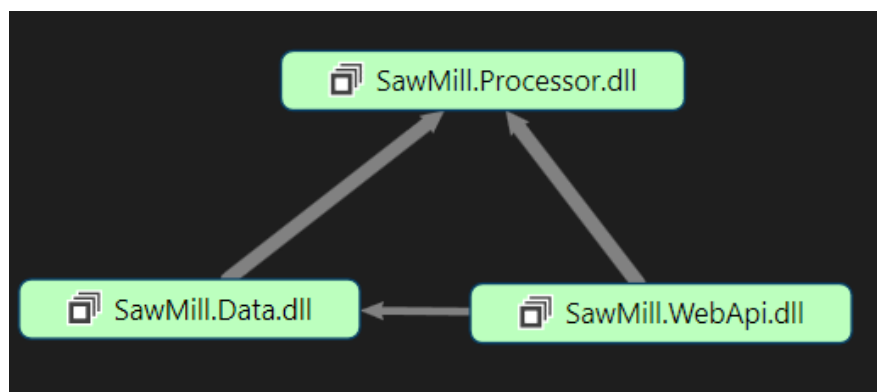
Slika 6.25 Tablica SystemReportAlertGroup

## 6.3. Web poslužitelj

Aplikacija na poslužitelju je glavni dio sustava jer sadrži poslovnu logiku, obrađuje klijentske zahtjeve i komunicira s bazom podataka. Poslužitelj veliki dio svoga posla obavlja komunicirajući s vanjskim komponentama preko mreže. S jedne strane preko pristupnih

točaka prima zahtjeve i odgovara na njih, a s druge strane šalje upite bazi podataka te čeka odgovor. Ovakve situacije uzrokuju puno vremena gdje aplikacija čeka na nečiji odgovor, kako bi se to vrijeme bolje iskoristilo primijenjeno je asinkrono programiranje. Programski jezik C# u kojem je razvijen poslužitelj ima ugrađenu podršku za asinkrono programiranje [8]. Prednost asinkronog programiranja je mogućnost da se za vrijeme čekanja dugotrajnog odgovora bez kojeg metoda ne može nastaviti aplikacija može izvršavati metodu koja trenutačno ne mora čekati. Posljedica ovog pristupa je vidljiva u razvijenim sučeljima, gotovo sve metode kao povratnu vrijednost imaju razred „Task<T>“. „Task<T>“ je razred koji označava da izvođenje metode može potrajati te pozivatelju pruža mogućnost da asinkrono čeka na njen završetak. Kada metoda s ovim povratnim razredom završi vraća rezultat koji je tipa generičkog parametra razreda „Task<T>“.

Poslužitelj je razdvojen u tri komponente prikazane na slici 6.26. Komponente poslužitelja su „SawMill.Data“, „SawMill.Processor“ i „SawMill.WebApi“. Strelice na slici 6.26 označavaju smjer ovisnosti gdje komponenta iz koje izlazi strelica ovisi o komponenti u koju strelica pokazuje.



Slika 6.26 Komponente poslužitelja

### 6.3.1. SawMill.WebApi

Komponenta „SawMill.WebApi“ izrađena je pomoću radnog okvira ASP.Net Core te pruža programibilno aplikacijsko sučelje (engl. *application programmable interface*, skraćeno API) klijentima. Projekt sadrži upravljače (engl. *controllers*) pomoću kojih aplikacija zaprima HTTP (engl. *HyperText Transfer Protocol*) zahtjev. Nakon zaprimanja zahtjeva upravljač poziva servis koji implementira aplikacijsku logiku te obavlja traženu akciju nad modelima aplikacije. Ovisno o rezultatu rada servisa upravljač vraća prikladan odgovor pozivatelju. Format podataka koje upravljač prima kroz zahtjev i vraća kao odgovor



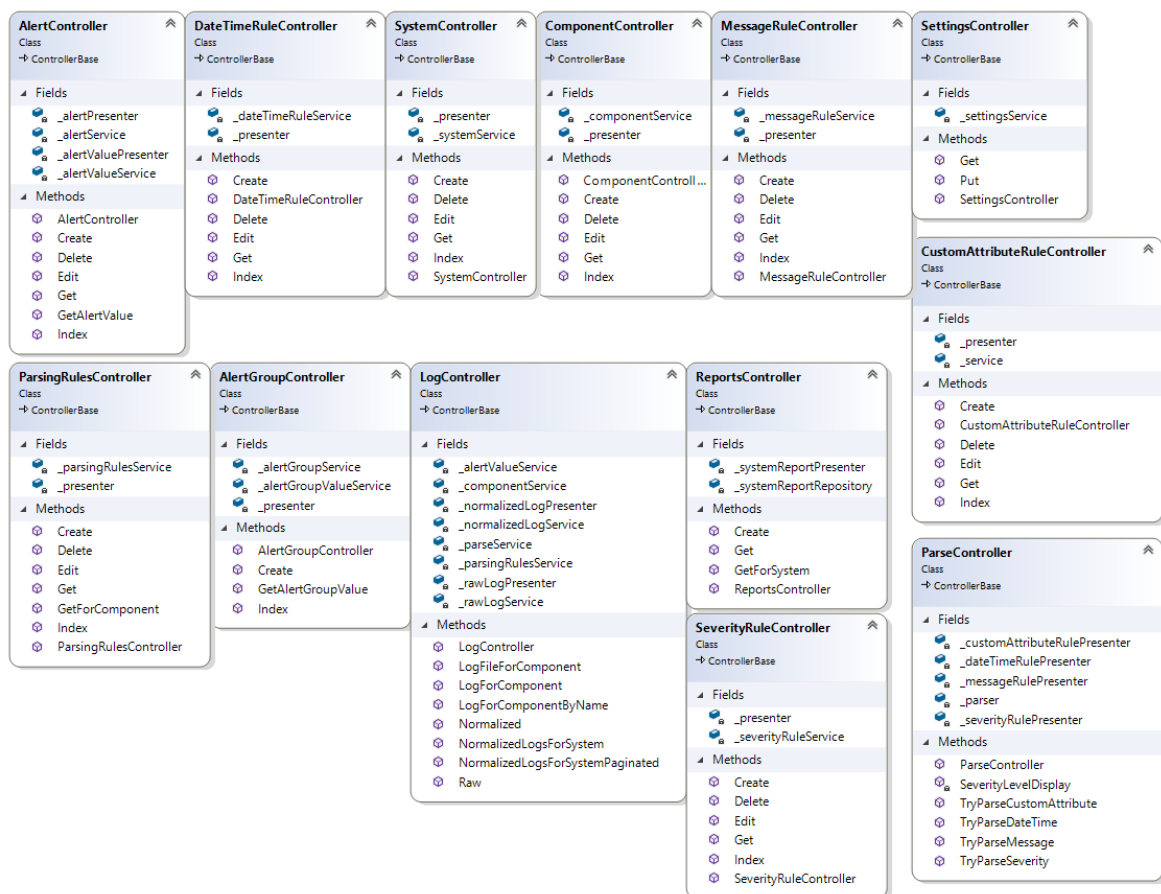
definiran je modelima pogleda (engl. *view model*). Prezenter (engl. *presenters*) su odgovorni za pretvaranje modela pogleda u poslovne modele i obrnuto. Projekt je zadužen za obavljanje injekcije zavisnosti (engl. *dependency injection*, skraćeno DI) čime postizemo inverziju kontrole (engl. *inversion of control*, skraćeno IoC). Injekcija zavisnosti ostvaruje se tako da svi razredi objekte o kojima su zavisni prime kroz konstruktor. Ovom tehnikom postizemo bolje odvajanje razreda (engl. *decoupling*). Kako bi injekcija zavisnosti bila moguća potrebno je sučelja i njihove implementacije registrirati u kontejner servisa (engl. *service container*). ASP,Net Core dolazi s ugrađenim kontejnerom koji je dostupan kroz „IServiceProvider“ sučelje [9]. Registracija razreda u kontejner obavlja se u „ConfigureServices“ metodi „Startup“ razreda. Registracija razreda prikazana je isječkom koda na slici 6.27.

```
// db context
services.AddScoped<SawMillDbContext>();

// repository
services.AddScoped<ISystemRepository, SystemRepository>();
services.AddScoped<IComponentRepository, ComponentRepository>();
```

Slika 6.27 Registracija razreda

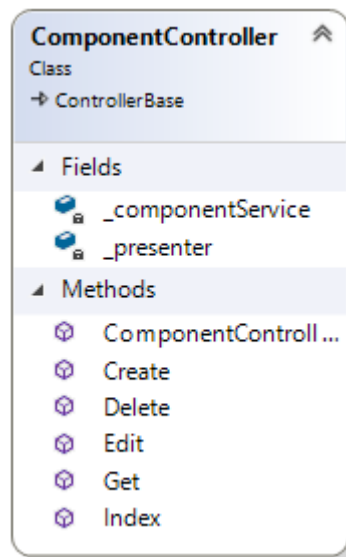
Projekt sadrži 13 upravljača gdje svaki upravljač sadrži metode koje predstavljaju akcije koje je moguće poduzeti nad modelima aplikacije. Metode upravljača predstavljaju pristupne točke (engl. *endpoints*) *web API*a. Svi upravljači prikazani su na slici 6.28.



Slika 6.28 Dijagram razreda upravljača

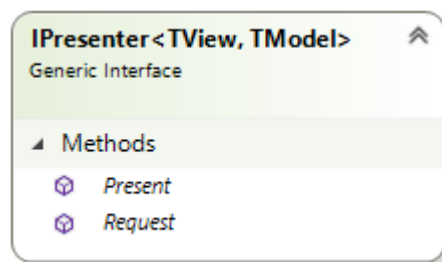
Na slici 6.29 detaljnije je prikazan upravljač za komponentu „ComponentController“. Upravljač zavisi o dva sučelja „IComponentService“ i „IPresenter<ComponentViewModel, Component>“ čije implementacije dobiva kroz konstruktor tehnikom injektiranja zavisnosti. Metoda upravljača vezana je na određenu adresu i HTTP glagol zahtjeva. Upravljač preko svojih metoda definira sljedeće pristupne točke *web API*a:

- POST `domena/api/component` – pristupna točka za stvaranje nove komponente.
- DELETE `domena/api/component/{id}` – pristupna točka za brisanje komponente.
- GET `domena/api/component/{id}` – pristupna točka za dohvaćanje jedne komponente.
- GET `domena/api/component` – pristupna točka za dohvaćanje svih komponenti
- PUT `domena/api/component/{id}` – pristupna točka za ažuriranje jedne komponente.

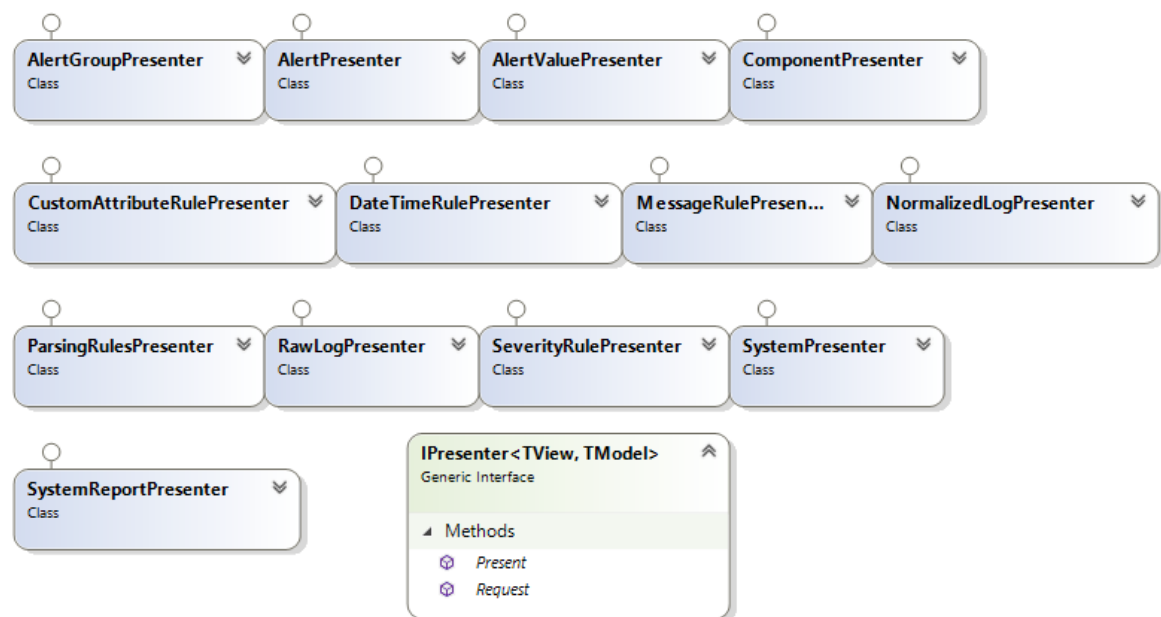


Slika 6.29 Upravljač za komponentu

Prezenter su definirani generičkim sučeljem „**IPresenter<TView, TModel>**“ prikazanim na slici 6.30. Sučelje koristi dvije generičke varijable „**TView**“ koja predstavlja model pogleda i „**TModel**“ koja predstavlja poslovni model. Metoda „**Present**“ pretvara poslovni model u model pogleda dok metoda „**Request**“ pretvara model pogleda u poslovni model. Razredi koji implementiraju ovo sučelje moraju specificirati generičke parametre ovisno o tome koju pretvorbu rade. Svi razredi koji implementiraju ovo sučelje i obavljaju funkciju prezentera prikazani su na slici 6.31.

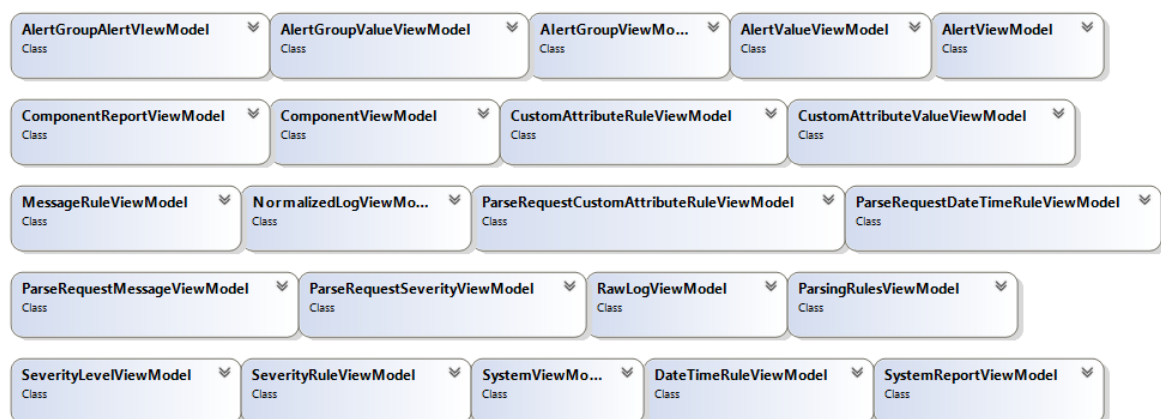


Slika 6.30 Bazno sučelje prezentera



Slika 6.31 Dijagram prezentera

Modeli pogleda su format podataka prilagođen za prikazivanje na korisničkom sučelju, u obzir je uzeta i činjenica da se modeli pogleda vraćaju kao odgovor na pozvane *web API* metode. Svi modeli pogleda prikazani su na slici 6.32.

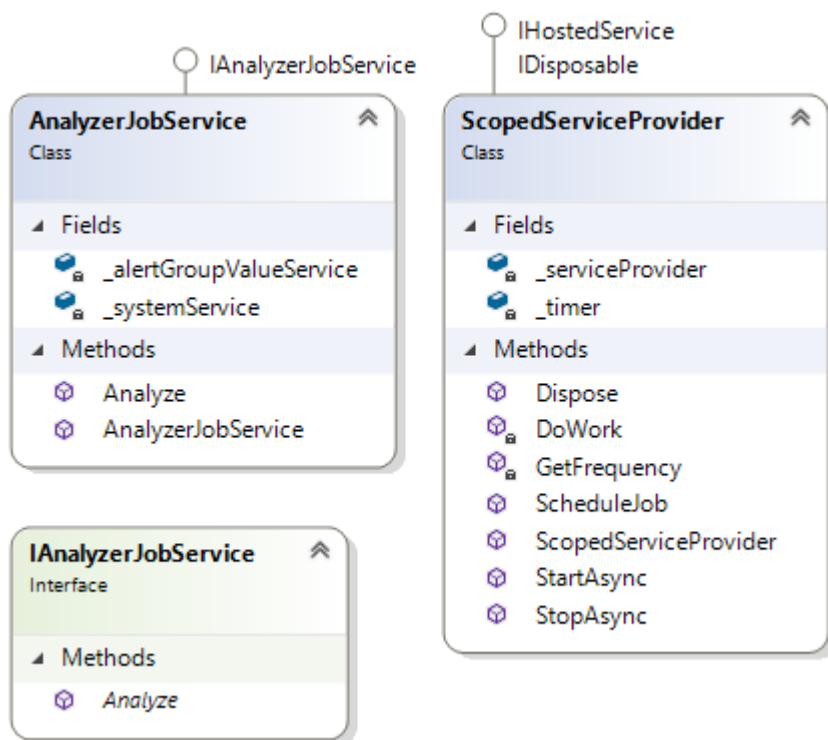


Slika 6.32 Dijagram razreda modela pogleda

Osim obrađivanja pristiglih *HTTP* zahtjeva aplikacija izvršava pozadinski zadatak koji u zadanim vremenskim intervalima obavlja posao stvaranja korelacijskih grupa. ASP.Net Core nudi mogućnost registracije servisa koji se pokreću prilikom pokretanja aplikacije [10]. Registracija servisa prikazana je isječkom koda na slici 6.33. Razredi i sučelja koja čine pozadinski servis prikazani su na slici 6.34.

```
// background services
services.AddScoped<IAnalyzerJobService, AnalyzerJobService>();
services.AddHostedService<ScopedServiceProvider>();
```

Slika 6.33 Isječak koda za registraciju servisa



Slika 6.34 Dijagram razreda pozadinskog servisa

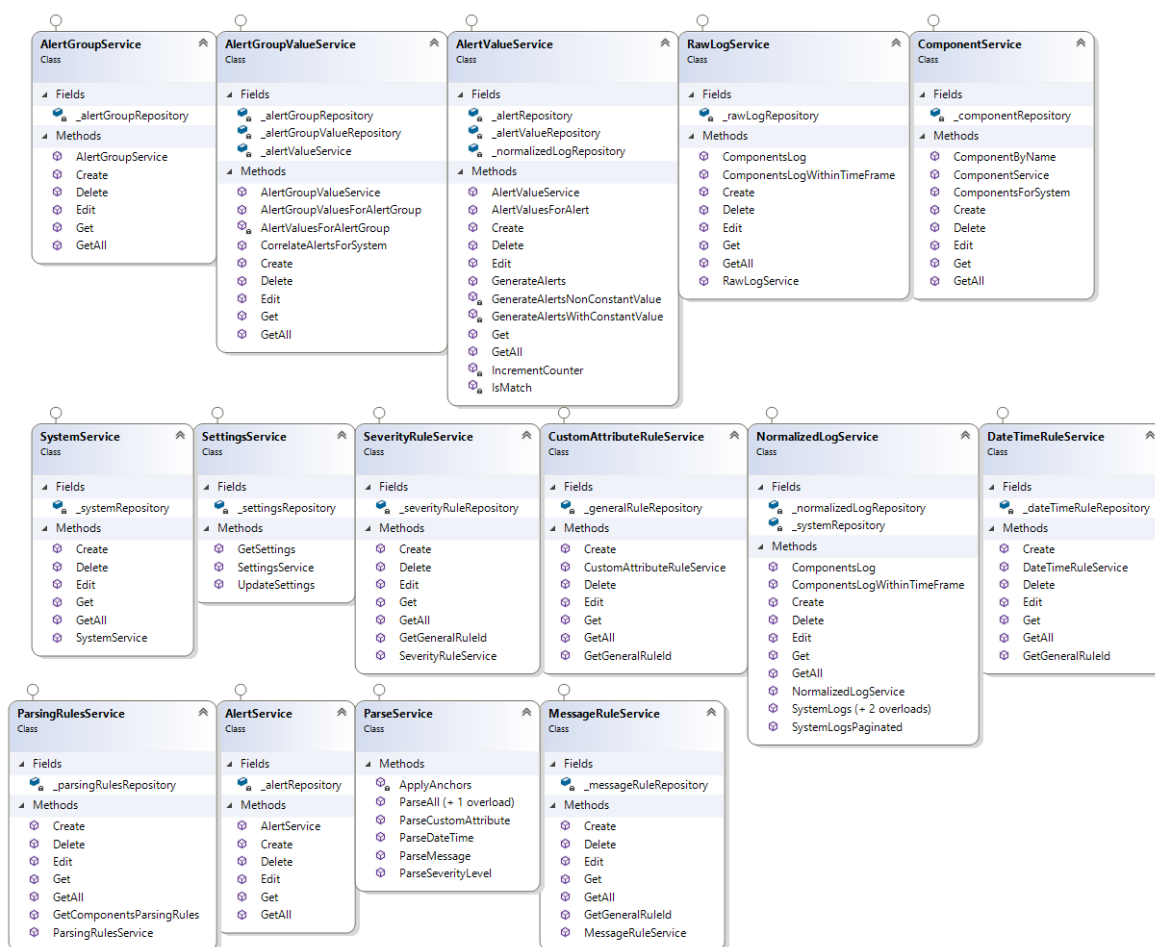
### 6.3.2. SawMill.Processor

Komponenta „SawMill.Processor“ je .Net Core knjižnica koja predstavlja sloj poslovne logike u arhitekturi. Knjižnica sadrži sučelja i implementacije servisa, poslovne modele i sučelja repozitorija. Pristup podacima ostvaren je oblikovnim obrascem repozitorija (engl. *repository*). Tim pristupom implementacija pristupa bazi podataka i podatkovni modeli izolirani su od poslovnih modela i poslovne logike. Servisi su zaduženi za obradu poslovnih modela, a putem sučelja repozitorija komuniciraju s bazom podataka. Sučelja servisa nasljeđuju generičko sučelje „ICrudServiceAsync<T>“ koje definira metode za stvaranje, dohvaćanje, uređivanje i brisanje modela. Generičko sučelje prikazano je na slici 6.36. Svako sučelje koje naslijedi generičko sučelje mora specificirati nad kojim modelom ostvaruje CRUD metode. Po potrebi sučelja servisa definiraju dodatne metode. Dijagram razreda servisa prikazan je na slici 6.35.

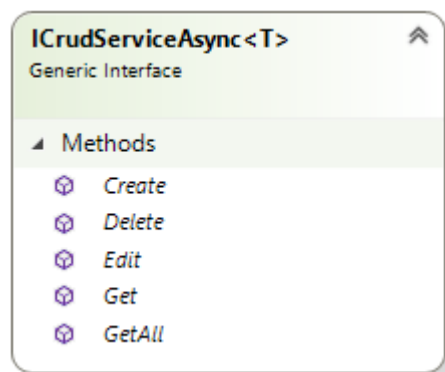
- **AlertGroupService** – sadrži CRUD metode za pravila korelacijske grupe.

- AlertGroupValueService – Sadrži CRUD metode za korelacijsku grupu, metodu za stvaranje korelacijskih grupa za zadani sustav i metodu za dohvaćanje stvorenih korelacijskih grupa prema jednom pravilu.
- AlertService – sadrži CRUD metode za pravila značajnog vremenskog intervala.
- AlertValueService – Sadrži CRUD metode za pronađene značajne vremenske intervale, metodu za pronalaženje značajnih vremenskih intervala u kolekciji zapisa dnevnika i metodu za dohvaćanje pronađenih značajnih vremenskih intervala prema zadanom pravilu.
- ComponentService – sadrži CRUD metode za komponente, metodu za dohvaćanje komponenti jednog sustava i metodu za dohvaćanje komponente prema imenu.
- CustomAttributeRuleService – sadrži CRUD metode za korisnička svojstva zapisa dnevnika.
- DateTimeRuleService – sadrži CRUD metode za pravilo parsiranja vremena iz zapisa dnevnika.
- MessageRuleService – sadrži CRUD metode za pravilo parsiranja poruke iz zapisa dnevnika.
- NormalizedLogService – sadrži CRUD metode za normalizirane zapise dnevnika, metodu za dohvaćanje normaliziranih zapisa jedne komponente unutar vremenskog intervala, metodu za dohvaćanje svih normaliziranih zapisa komponente, metodu za dohvaćanje normaliziranih zapisa sustava unutar zadanog vremenskog intervala, metodu za dohvaćanje ograničenog broja normaliziranih zapisa sustava unutar zadanog vremenskog intervala, metodu za dohvaćanje zadani broj normaliziranih zapisa sustava koji dolaze nakon danom normaliziranog zapisa.
- ParseService – servis s metodama za obradu zapisa dnevnika prema danim pravilima, sadrži metode za parsiranje vremena iz danog zapisa prema danom pravilu, metodu za parsiranje razine ozbiljnosti zapisa prema zadanom pravilu, metodu za parsiranje poruke zapisa prema zadanom pravilu, metodu za parsiranje korisničkog svojstva iz zapisa prema danom pravilu, metodu za primjenu svih pravila na zapis kako bi se dobio zapis u normaliziranom obliku.
- ParsingRulesService – sadrži CRUD metode za grupu pravila obrade zapisa dnevnika i metodu za dohvaćanje grupe pravila za zadanu komponentu.
- RawLogService – sadrži CRUD metode za neobrađene zapise.

- SettingsService – servis za dohvaćanje i ažuriranje postavke učestalosti pokretanja analizatora korelacijskih grupa.
- SeverityRuleService – sadrži CRUD metode za pravilo parsiranja ozbiljnosti iz zapisa dnevnika.
- SystemService – sadrži CRUD metode za sustav.



Slika 6.35 Dijagram razreda servisa



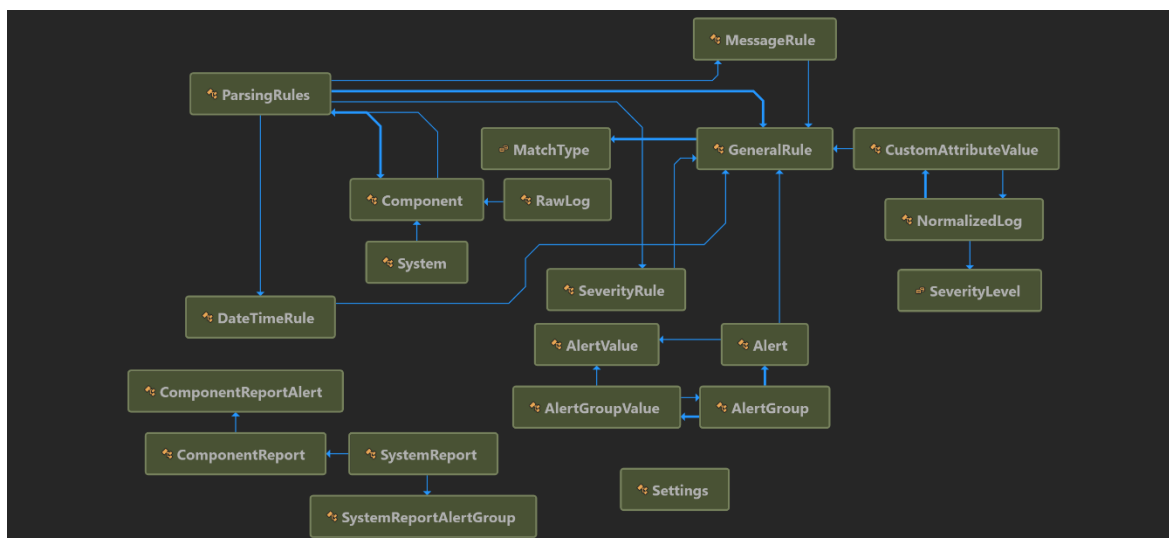
Slika 6.36 Bazno sučelje sučelja servisa

Slika 6.37 prikazuje dijagram razreda poslovnih modela.

- Alert – model pravila za otkrivanje značajnih vremenskih intervala u dnevniku komponente.
- AlertValue – pronađeni značajni vremenski interval.
- AlertGroup – korelacijska pravila za sustav.
- AlertGroupValue – stvorene korelacijske grupe.
- Component – model komponente u sustavu.
- ComponentReport – izvještaj učestalosti tipova značajnih vremenskih intervala komponente.
- ComponentReportAlert – izvještaj učestalosti jednog tipa značajnog vremenskog intervala komponente.
- CustomAttributeValue – vrijednost korisničkog svojstva za normalizirani zapis.
- DateTimeRule – pravilo za parsiranje vremena događaja iz njegovog zapisa.
- GeneralRule – svojstva zajednička svim pravilima za obradu zapisa.
- MessageRule – pravilo za parsiranje poruke događaja iz njegovog zapisa.
- NormalizedLog – normalizirani zapis.
- ParsingRules – skup pravila za obradu zapisa komponente.
- RawLog – model neobrađenog zapisa komponente.
- Settings – model postavki intervala izvršavanja analizatora.
- SeverityRule – model pravila za parsiranje razine ozbiljnosti događaja iz njegovog zapisa.
- System – model sustava.
- SystemReport – model izvještaja učestalosti pojavljivanja korelacijskih grupa za sustav.



- SystemReportAlertGroup – model izvještaja učestalosti jedne korelacijske grupe u sustavu.



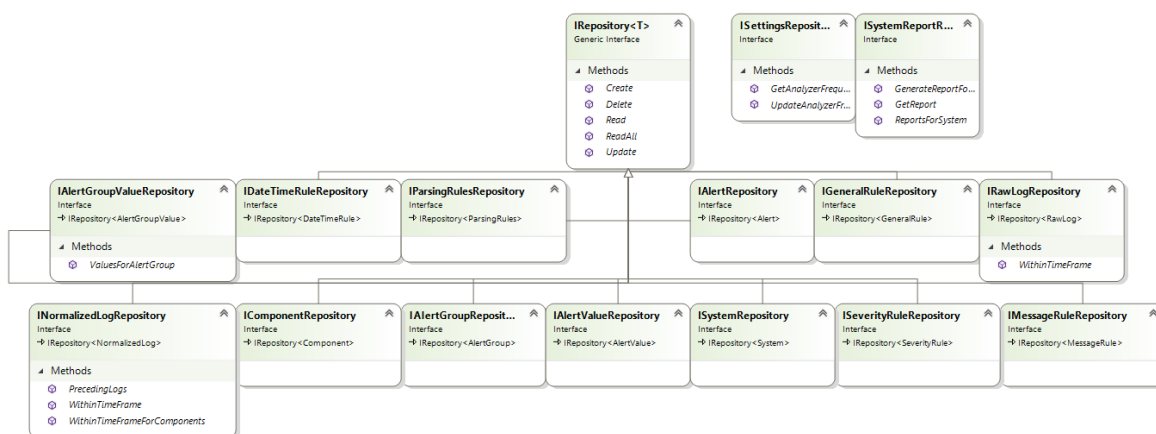
Slika 6.37 Dijagram razreda poslovnih modela

Sučelje „IRepository<T>“ je generičko sučelje koje definira osnove metode za rad s jednim modelom. Ostala sučelja repozitorija nasljeđuju iz ovog sučelja te specificiraju generički parametar u ovisnosti za koji model implementacija sučelja nudi CRUD metode. Po potrebi sučelje definira dodatne metode čija implementacija je zahtijevala bliži pristup spremištu podataka. Slika 6.38 prikazuje sučelja repozitorija.

- IAlertGroupRepository – definira CRUD metode za AlertGroup model.
- IAlertGroupValueRepository – definira CRUD metode za AlertGroupValue model i metodu za dohvaćanje svih podataka prema stranom ključu na AlertGroup model.
- IAlertRepository – definira CRUD metode za Alert model.
- IAlertValueRepository – definira CRUD metode za AlertValue model.
- IComponentRepository – definira CRUD metoda za Component model.
- IDateTimeRuleRepository – definira CRUD metode za DateTimeRule model.
- IGeneralRuleRepository – definira CRUD metode za GeneralRule model.
- IMessageRuleRepository – sadrži CRUD metode za MessageRule model.
- INormalizedLogRepository – sadrži CRUD metode za NormalizedLog model, metodu za dohvaćanje podataka prema stranom ključu na komponentu koji zadovoljavaju uvjet da se nalaze unutar zadanog vremenskog intervala, metodu za dohvaćanje zadanog broja podataka koji imaju strani ključ na jednu od danih komponenti i dogodili su se unutar zadanog vremenskog intervala. Dodatno definira

metodu za dohvaćanje zadanog broja podataka koji imaju strani ključ na jednu od zadanih komponenti te prethode zadanom normaliziranome zapisu.

- IParsingRulesRepository – sadrži CRUD metode za ParsingRules model.
- IRawLogRepository – sadrži CRUD metode za RawLog model.
- ISettingsRepository – sadrži metodu za dohvaćanje postavki analizatora i metodu za ažuriranje postavki analizatora.
- ISeverityRuleRepository – sadrži CRUD metode za SeverityRule model.
- ISystemReportRepository – definira metode za generiranje izvještaja za sustav s zadanim primarnim ključem, metodu za dohvaćanje svih izvještaja jednog sustava i metodu za dohvaćanje izvještaja sustava prema primarnom ključu.
- ISystemRepository – definira CRUD metode za System model.

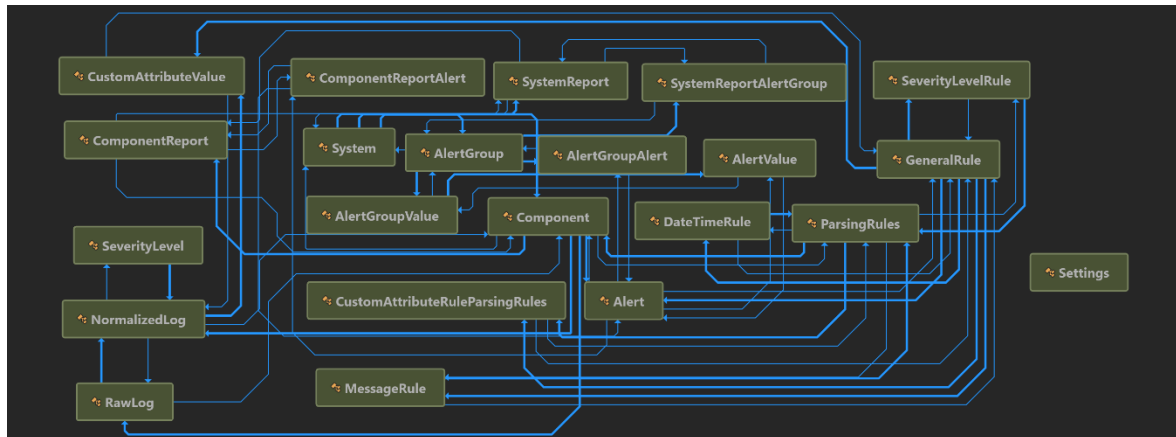


Slika 6.38 Dijagram sučelja repozitorija

### 6.3.3. SawMill.Data

Komponenta „SawMill.Data“ je .Net Core knjižnica koja u arhitekturi ima ulogu sloja za pristup podacima. Pristup podacima u bazi podataka ostvaren je radnim okvirom „Entity Framework Core“. Knjižnica sadrži podatkovne modele, implementacije sučelja repozitorija, metode za pretvorbu poslovnih modela u podatkovne modele i obrnuto te konfiguraciju objektno-relacijskog pretvaranja (engl. *object-relational mapping*). Podatkovni modeli predstavljaju tablice u bazi podataka, prikazani su na slici 6.39. Radni okvir pristup bazi omogućava nasljeđivanjem iz razreda „DbContext“ te je to ostvareno u razredu „SawMillDbContext“. „SawMillDbContext“ obavlja ulogu konfiguracije objektno-relacijskog pretvaranja povezivanjem svojstava podatkovnih modela s atributima tablice baze podataka. Primjer konfiguracije pretvaranja za podatkovni model prikazan je isječkom koda na slici 6.40. Pretvorba podatkovnih modela u poslovne modele ostvarena je skupom

statičkih metoda „ToBusinessModel“. Ulazni parametar ove metode je tipa podatkovnog modela koji pretvaramo, a izlazni tip metode je poslovni model u koji pretvaramo. Pretvorba poslovnih modela u podatkovne ostvarena je skupom statičkih metoda „ToDomainModel“. Ulazni parametar metode je poslovni model koji pretvaramo, a izlazi parametar je podatkovni model u koji pretvaramo. Pretvarači su prikazani na slici 6.41.



Slika 6.39 Dijagram podatkovnih modela

```
modelBuilder.Entity<Component>(entity =>
{
    entity.Property(c => c.ComponentId).ValueGeneratedOnAdd();

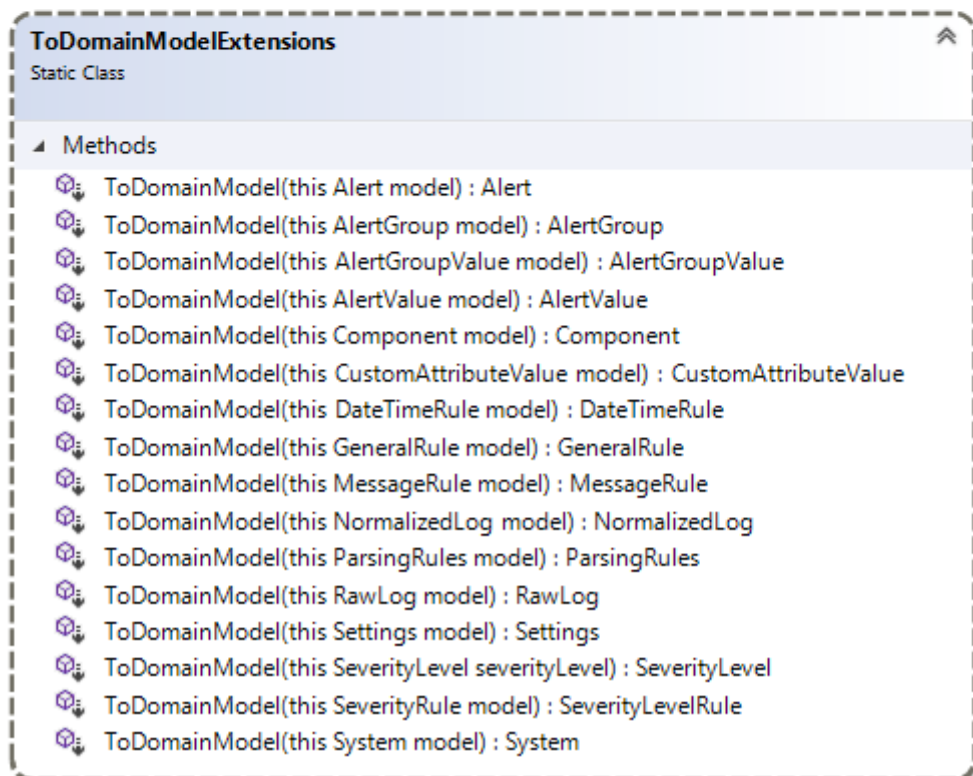
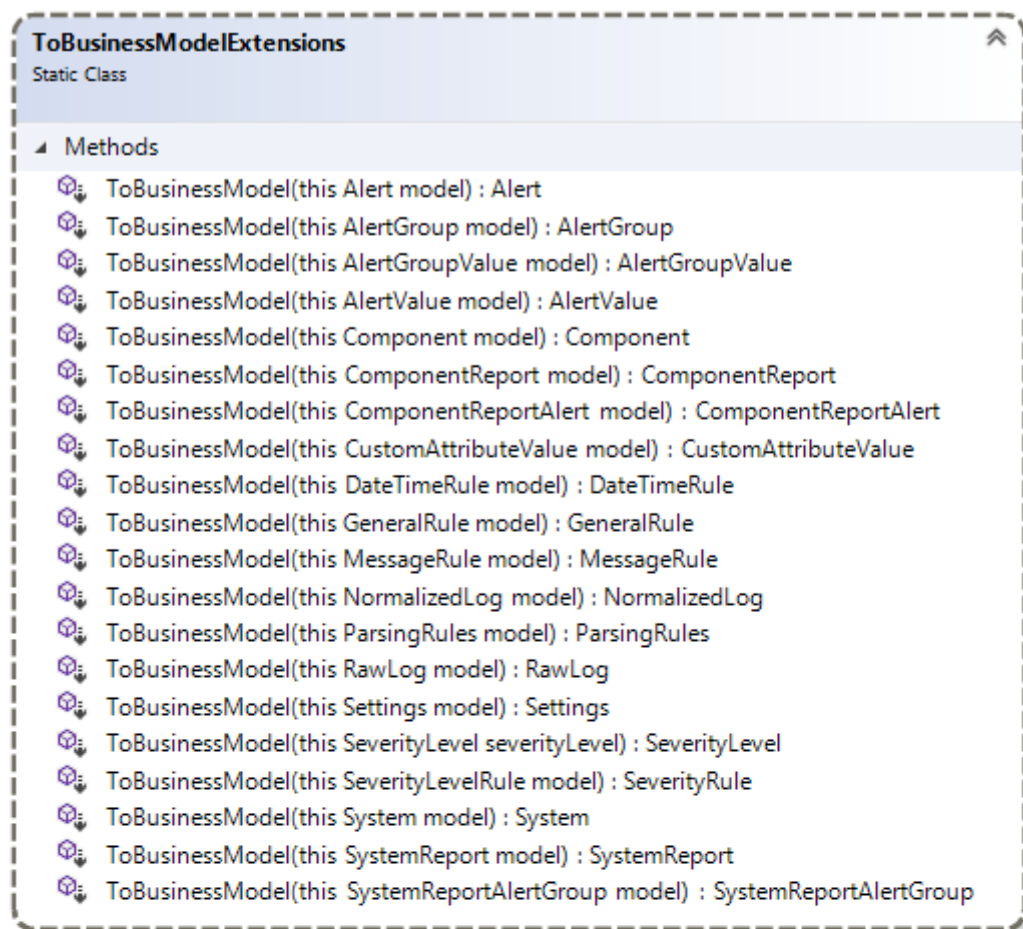
    entity.ForSqlServerHasIndex(e => e.ComponentName).IsUnique();

    entity.Property(e => e.ComponentName)
        .IsRequired()
        .HasMaxLength(256)
        .IsUnicode(false);

    entity.HasOne(navigationExpression: d => d.System)
        .WithMany(navigationExpression: p => p.Component)
        .HasForeignKey(d => d.SystemId)
        .IsRequired(false)
        .OnDelete(DeleteBehavior.ClientSetNull)
        .HasConstraintName("FK_Component_System");

    entity.HasOne(navigationExpression: d => d.ParsingRules)
        .WithMany(navigationExpression: d => d.Components)
        .HasForeignKey(d => d.ParsingRulesId)
        .IsRequired(false).OnDelete(DeleteBehavior.ClientSetNull)
        .HasConstraintName("FK_Component_ParsingRules");
});
```

Slika 6.40 Konfiguracija pretvaranja Component podatkovnog modela

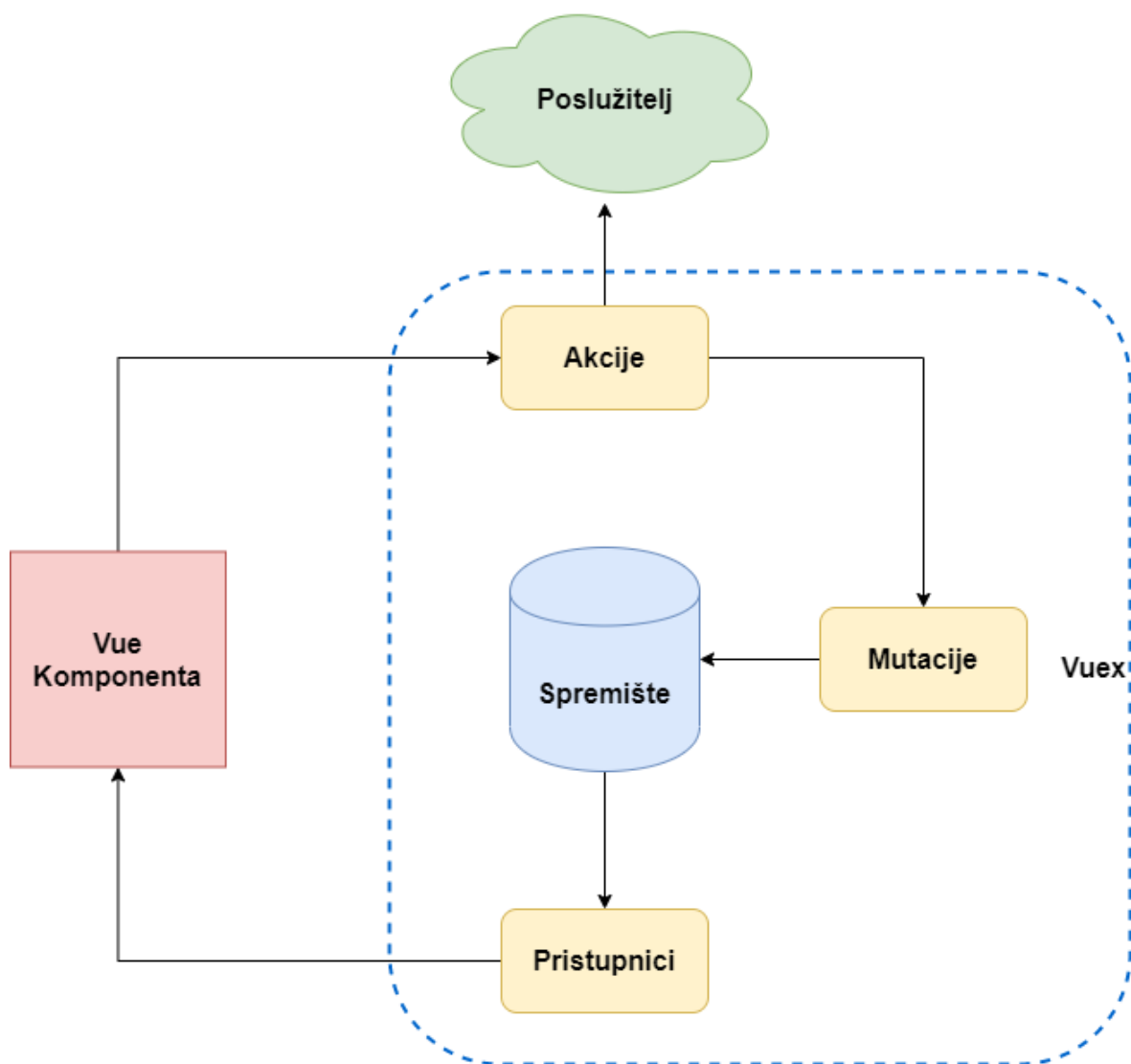


Slika 6.41 Dijagram pretvarača

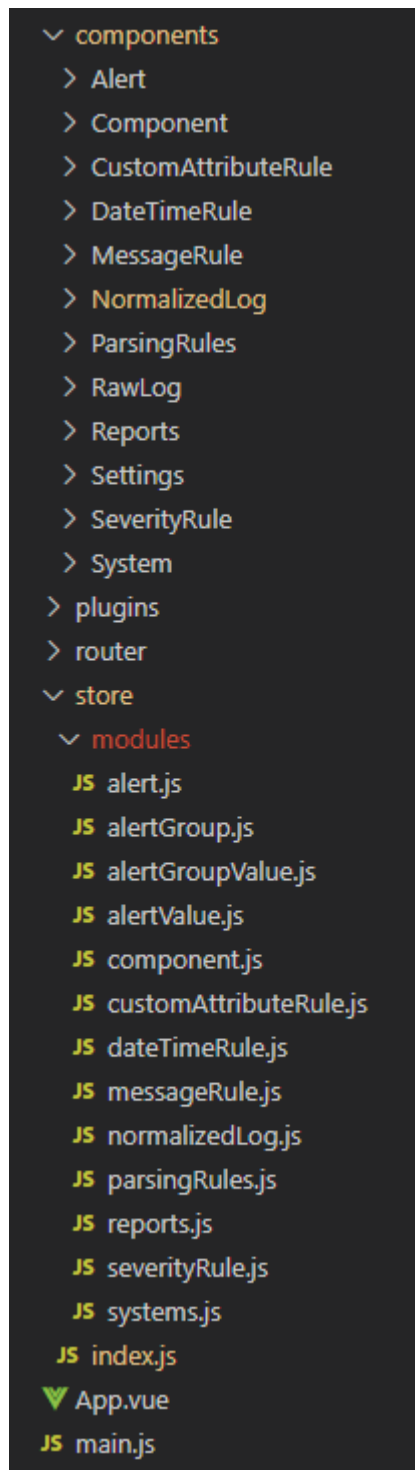
## 6.4. Web klijent

Web klijent (SawMill.Frontend) predstavlja prezentacijski sloj u arhitekturi sustava te je ostvaren kao jednostranična aplikacija kojoj se može pristupiti putem web preglednika (engl. *browser*). Web aplikacija ostvarena je radnim okvirom Vue.js. Web aplikacija komunicira s poslužiteljem kako bi dohvatila podatke potrebe za prikaz te kako bi obavila određene korisničke akcije.

Arhitektura aplikacije primjenjuje oblikovni obrazac upravljanja stanjem (engl. *state managment pattern*). Obrazac je primijenjen korištenjem knjižnice Vuex. Kada više komponenti prikazuje i mijenja iste podatke nastaje problem ažuriranja svih komponenti koje koriste te podatke. Obrazac upravljanja stanjem rješava taj problem uvođenjem globalnog stanja koje postaje jedini izvor podataka te se njihova promjena odražava u svim komponentama koje su koristile te podatke. Podatci se nalaze u spremištu (engl. *store*), a s njima radimo kroz nekoliko vrsta funkcija. Pomoću funkcija pristupa (engl. *getters*) čitamo podatke u spremištu. Funkcije pristupa omogućavaju nam da iz spremišta vraćamo podatke nakon što na njih primijenimo dodatnu logiku, važno je primijetiti da primjena dodatne logike ne mijenja same podatke u spremištu. Funkcije akcija (engl. *actions*) služe za obavljanje asinkronih operacija. One ne mijenjaju podatke u spremištu izravno već za to pozivaju mutacije. Funkcije mutacija (engl. *mutations*) su sinkrone funkcije koje mijenjaju podatke u spremištu. Mutacije su jedine funkcije koje mijenjaju sadržaj spremišta. Arhitektura web aplikacije s Vuex knjižnicom obrasca za upravljanje stanjem prikazana je na slici 6.42. Struktura aplikacije sastoji se od komponenti (engl. *components*), spremišta, adresa (engl. *routes*) i početnog dijela aplikacije. Struktura aplikacije prikazana je na slici 6.43.



Slika 6.42 Arhitektura web aplikacije [11]



Slika 6.43 Struktura web aplikacije

### 6.4.1. Komponente

Komponente su višestruko iskoristive, imenovane instance Vue objekata [12]. Svaka komponenta sastoji se od predloška (engl. *template*), koda (engl. *script*) i stila (engl. *style*). Komponente su organizirane hijerarhijski.

- App.vue – korijenska komponenta koja učitava potrebne resurse poput fontova i stilova, postavlja komponentu za navigaciju, postavljaju se komponente po URL adresama.
- SystemIndex.vue – komponenta koja prikazuje sve korisnikove sustave kao i opcije za stvaranje novih sustava. Sadrži komponente „System.vue“ i „SystemForm.vue“.
- System.vue – komponenta koja prikazuje opisne podatke o sustavu, komponente koje su mu pridijeljene. Sadrži opcije za uređivanje sustava, stvaranje nove komponente sustava, brisanje sustava i navigaciju na detalje sustava. Sadrži komponente „SystemForm.vue“ i „ComponentForm.vue“.
- SystemDetail.vue – prikazuje detalje sustava, korelacijske grupe sustava, normalizirane zapise sustava, popis postojećih izvještaja sustava. Nudi mogućnosti unosa zapisa dnevnika neke od komponenti sustava, dodavanjem korelacijskih grupa sustavu, generiranja izvještaja sustava. Sadrži komponente „SystemLog.vue“, „System.vue“, „RawLogForm.vue“, „AlertGroup.vue“, „AlertGroupForm.vue“.
- SystemForm.vue – komponenta s formom za stvaranje novog sustava ili uređivanje postojećeg sustava. Sadrži komponentu „ComponentForm.vue“.
- Component.vue – komponenta koja prikazuje detalje jedne komponente sustava. Prikazuje pravila za obradu dnevnika komponente, pravila značajnih vremenskih intervala komponente, mogućnost za uređivanje navedenih pravila. Sadrži komponente „ParsingRules.vue“, „Alert.vue“ i „AlertForm.vue“.
- ComponentForm.vue – komponenta za stvaranje nove ili ažuriranje postojeće komponente.
- Alert.vue – prikazuje pravilo za otkrivanje značajnih vremenskih intervala i otkrivene vremenske intervale. Za svaki otkriveni značajni vremenski interval moguće je pogledati zapise koji se nalaze u njemu. Nudi mogućnost uređivanja pravila. Sadrži komponente „AlertForm.vue“, „CustomAttributeRule.vue“ i „SystemLog.vue“.
- AlertForm.vue – komponenta za stvaranje novog pravila otkrivanja značajnih vremenskih intervala ili uređivanje postojećeg pravila.
- AlertGroup.vue – prikazuje pravilo korelacijske grupe kao i grupe stvorene prema ovom pravilu. Sadrži komponentu „SystemLog.vue“.
- AlertGroupForm.vue – komponenta za stvaranje novog pravila korelacijske grupe.



- CustomAttributeRule.vue – prikazuje pravilo za parsiranje vlastitog svojstva iz zapisa dnevnika.
- CustomAttributeRuleForm.vue – komponenta za stvaranje novog pravila obrade zapisa dnevnika ili ažuriranje postojećeg.
- DateTimeRule.vue – komponenta za prikaz pravila obrade vremena iz zapisa dnevnika.
- DateTimeRuleForm.vue – komponenta za stvaranje novog pravila obrade vremena iz zapisa dnevnika ili ažuriranje postojećeg.
- MessageRule.vue – komponenta za prikaz pravila obrade poruke iz zapisa dnevnika.
- MessageRuleForm.vue – komponenta za stvaranje novog pravila obrade poruke iz zapisa dnevnika ili ažuriranje postojećeg.
- SystemLog.vue – komponenta za prikaz normaliziranih zapisa jednog sustava. Nudi mogućnost filtriranja zapisa i dohvaćanja dodatnih zapisa.
- ParsingRules.vue – komponenta koja prikazuje sva pravila za obradu jednog formata dnevnika. Sadrži komponente „DateTimeRule.vue“, „MessageRule.vue“, „SeverityRule.vue“, „CustomAttributeRule.vue“.
- ParsingRulesForm.vue – komponenta koja nudi mogućnost stvaranja nove grupe pravila za obradu dnevnika jedne komponente ili ažuriranja postojeće grupe pravila. Sadrži komponente „DateTimeRule.vue“, „MessageRule.vue“, „SeverityRule.vue“, „CustomAttributeRule.vue“, „DateTimeRuleForm.vue“, „MessageRuleForm.vue“, „SeverityRuleForm.vue“, „CustomAttributeRuleForm.vue“.
- RawLogForm.vue – komponenta za ručni unos zapisa dnevnika komponente. Nudi mogućnost unosa teksta ili datoteke dnevnika.
- SystemReport.vue – komponenta za prikaz izvještaja učestalosti korelacijskih grupa za sustav i učestalosti vrsta značajnih vremenskih intervala komponenti sustava.
- SettingsForm.vue – komponenta za ažuriranje postavki analizatora.
- SeverityRule.vue – komponenta za prikaz pravila obrade ozbiljnosti događaja iz zapisa dnevnika.
- SeverityRuleForm.vue – komponenta za stvaranje novog pravila obrade ozbiljnosti događaja iz zapisa dnevnika ili ažuriranje postojećeg.

### 6.4.2. Spremišta

- alert.js – spremište s logikom za dohvaćanje, stvaranje i ažuriranje pravila značajnih vremenskih intervala.
- alertGroup.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje korelacijskih grupa.
- alertGroupValue.js - spremište s logikom za dohvaćanje stvorenih korelacijskih grupa.
- alertValue.js - spremište s logikom za dohvaćanje pronađenih značajnih vremenskih intervala.
- component.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje komponenti.
- customAttributeRule.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje proizvoljnih pravila obrade zapisa dnevnika.
- dateTimeRule.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje pravila za obradu vremena iz zapisa dnevnika.
- messageRule.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje pravila za obradu poruke iz zapisa dnevnika.
- normalizedLog.js - spremište s logikom za dohvaćanje normaliziranih zapisa.
- parsingRules.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje grupe pravila za obradu dnevnika komponente.
- reports.js - spremište s logikom za dohvaćanje i stvaranje izvještaja sustava.
- severityRule.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje pravila za obradu ozbiljnosti događaja iz zapisa dnevnika.
- systems.js - spremište s logikom za dohvaćanje, stvaranje i ažuriranje sustava.

## 6.5. Servis za prikupljanje dnevnika

Komponenta „SawMill.Collector“ je .Net Core aplikacija koja sadrži mogućnost praćenja dnevnika komponenti i prebacivanja tih dnevnika na poslužitelj kako bi se provela njihova analiza. Knjižnica „Topshelf“ iskorištena je kako bi se omogućilo instaliranje aplikacije kao servisa operacijskog sustava. Aplikacija koristi konfiguracijsku datoteku „INI“ formata [13]. Primjer formata konfiguracijske dan je isječkom koda 6.1. Konfiguracijska datoteka sadrži tri odjeljka (engl. *section*). Odjeljak „components“ sadrži parove ključ-vrijednost gdje je ključ ime komponente čiji dnevnik se prikupljaju, a vrijednost putanja do mape gdje se

dnevnici te komponente nalaze. Odjeljak „general“ sadrži postavke za ponašanje servisa. Vrijednost ključa „frequency“ određuje period nakon kojeg servis provjerava postoje li novi zapisi u dnevnicima ili postoje li nove datoteke dnevnika. Vrijednost ključa „url“ je adresa na kojoj se nalazi poslužitelj. Odjeljak „patterns“ sadrži parove ključ-vrijednosti gdje ključ mora biti ime komponente navedeno u odjeljku „components“, a vrijednost uzorak prema kojem se određuje koje datoteke su dnevnici komponente.

```
[components]
myComponent="component directory"

[general]
frequency=300
url="http://localhost:52361/api/"

[patterns]
myComponent="filePattern*.log"
```

Isječak koda 6.1 Format konfiguracijske datoteke

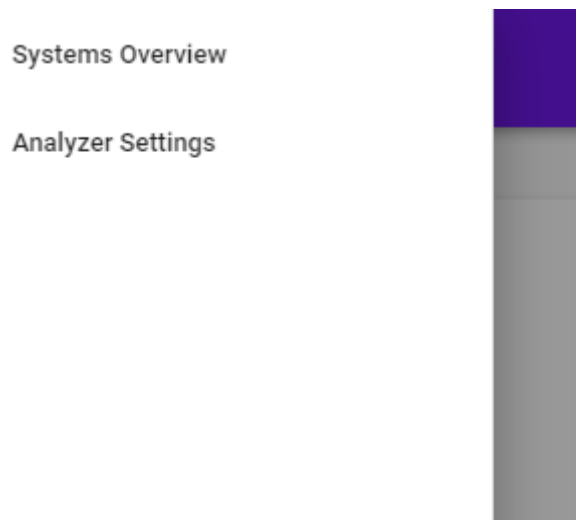
Servis se instalira pokretanjem izvršive datoteke naredbom danom isječkom koda 6.2.

```
Sawmill.collector.exe -install -config:"putanja/config.ini"
```

Isječak koda 6.2 Naredba za instalaciju servisa

## 7. Funkcionalnosti sustava

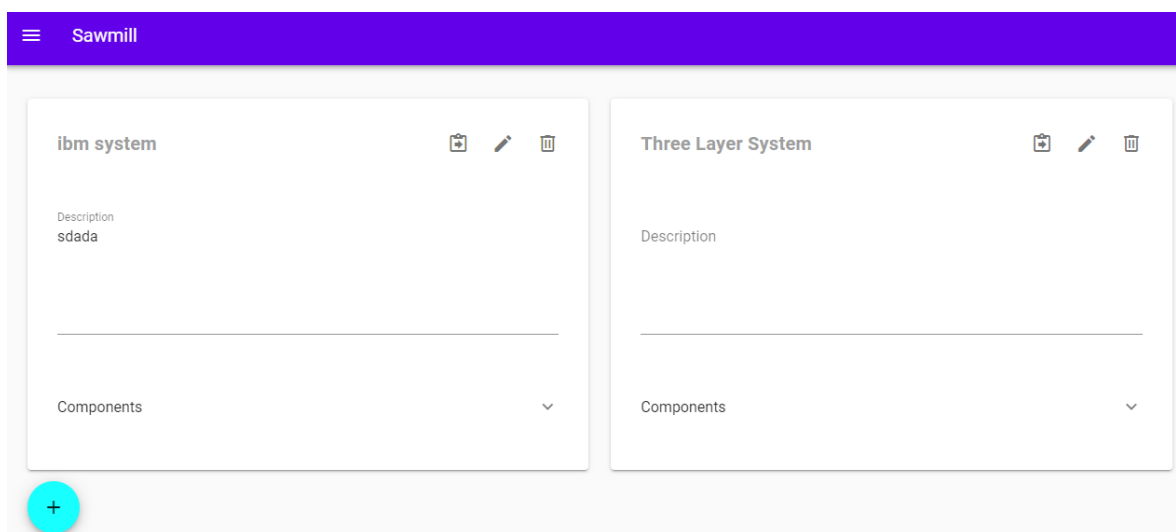
Kao dio rada razvijen je sustav „SawMill“ koji korisniku pruža web sučelje za rad sa sustavom. Navigacijski izbornik dostupan je pomoću trake pri vrhu aplikacije. Slika 7.1 prikazuje navigacijski izbornik koji sadrži dvije opcije. Opcija „System Overview“ korisnika vraća na stranicu za pregled postojećih sustava. Opcija „Analyzer Settings“ prikazuje stranicu za ažuriranje postavki analizatora dnevnika.



Slika 7.1 Navigacijski izbornik

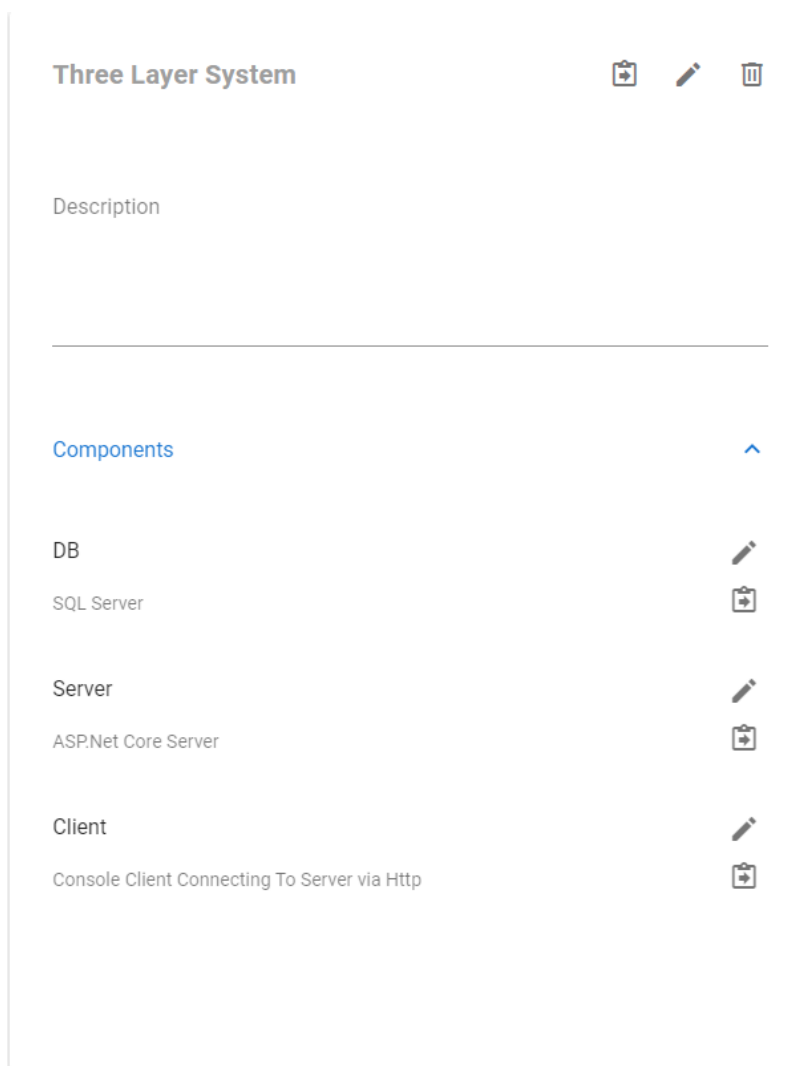
### 7.1. Sustav

Pregled svih definiranih sustava prikazan je na slici 7.2. Stranica prikazuje definirane informacijske sustave. Sustavi su organizirani u kartice koje prikazuju osnovne podatke o sustavu. Gumb u donjem lijevom kutu slike 7.2 otvara formu za unos novog sustava.



Slika 7.2 Početna stranica aplikacije

Slika 7.3 detaljnije prikazuje karticu jednog sustava. Prikazuju se podatci o nazivu sustava, opisu sustava i popisu komponenti sustava. Svaka kartica sadrži poveznicu na stranicu detaljnog prikaza sustava, gumb za uređivanje osnovnih podataka sustava i gumb za brisanje sustava. Popis komponenti sustava prikazuje ime i opis svake komponente. Uz svaku komponentu nalaze se poveznice na stranicu detalja komponente i formu ažuriranja podataka komponente.



Slika 7.3 Detaljniji prikaz kartice jednog sustava

Forma za stvaranje novog sustava prikazana je na slici 7.4. Forma se sastoji od polja za unos imena sustava, polja za unos opisa sustava i padajućeg izbornika za odabir komponenti sustava. Korisnik može odabrati komponente koje nisu pridružene niti jednom sustavu. Gumb „Create Component“ na dnu padajuće liste otvara dijalog s formom za stvaranje nove komponente. Padajuća lista prikazana je na slici 7.5. Gumb „Cancel“ pri dnu forme za stvaranje sustava poništava akciju stvaranja novog sustava i zatvara dijalog. Gumb „Submit“

pokreće akciju stvaranja novog sustava. Akcija komunicira s poslužiteljem i na njegov odgovor prikazuje novostvoreni sustav.

---

Name  
Analyzer test system

---

Description  
Information system for testing analyzer.

---

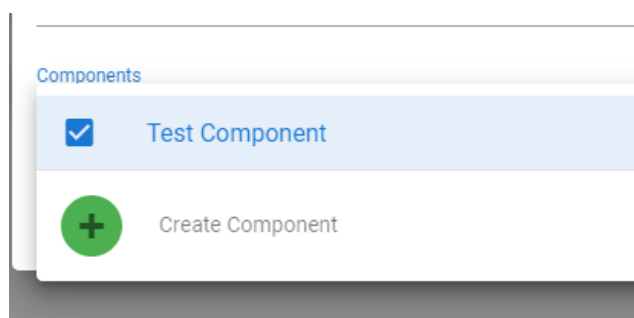
Components  
Test Component

---

SUBMIT

CANCEL

Slika 7.4 Forma za stvaranje sustava



Slika 7.5 Padajući izbornik za izbor komponenti

Slika 7.6 prikazuje formu za uređivanje podataka sustava. Forma se otvara gumbom za uređivanje koji se nalazi na kartici sustava. Forma dopušta uređivanje imena sustava, opisa sustava i komponenti koje pripadaju sustavu. Gumb „Submit“ pokreće akciju ažuriranja sustava na poslužitelju. Gumb „Cancel“ odbacuje promijene i zatvara formu za uređivanje sustava.

ibm system

Name  
ibm system

Description  
IBM OS System

Components  
testcomp

SUBMIT CANCEL

Slika 7.6 Forma za uređivanje sustava

Stranica detalja sustava prikazana je na slici 7.7. Stranica je dostupna putem poveznice na kartici sustava sa stranice pregleda sustava. Gornji dio stranice sustava prikazuje iste podatke prikazane za sustav na početnoj stranici. Donji dio stranice sadrži četiri polja koja se klikom proširuju i prikazuju dodatne podatke. To su podatci o korelacijskim grupama sustava, forma za unos zapisa dnevnika komponenti sustava, izvještaji sustava i normalizirani zapisi sustava.

Three Layer System

Description

Information system composed of three components: Database (MSSQL Server), Web Server and Client that makes request to Web Servers API.

Components

Correlation Groups

Log Input

Reports

Normalized Log

Slika 7.7 Stranica detalja sustava

## 7.2. Komponenta

Dijalog s formom za stvaranje nove komponente prikazan je na slici 7.8. Dijalog je dostupan iz padajućeg izbornika za pridruživanje komponenti sustavu. Forma sadrži polje za unos imena komponente i polje za unos opisa komponente. Gumb „Submit“ šalje zahtjev poslužitelju za stvaranje nove komponente. Gumb „Cancel“ otkazuje stvaranje nove komponente i zatvara dijalog s formom.



Name  
Test Component

Description  
Description of component

SUBMIT CANCEL

Slika 7.8 Forma za stvaranje komponente

Forma za ažuriranje podataka komponente prikazana je na slici 7.9. Forma je dostupna putem poveznice za uređivanje komponente na popisu komponenti sustava. Forma omogućava ažuriranje imena komponente, opisa komponente i izbora sustava kojem komponente pripada putem padajućeg izbornika. Gumb „Submit“ pokreće akciju ažuriranja komponente na poslužitelju. Gumb „Cancel“ odbacuje promijene i zatvara dijalog s formom za ažuriranje komponente.

Name  
DB

Description  
SQL Server

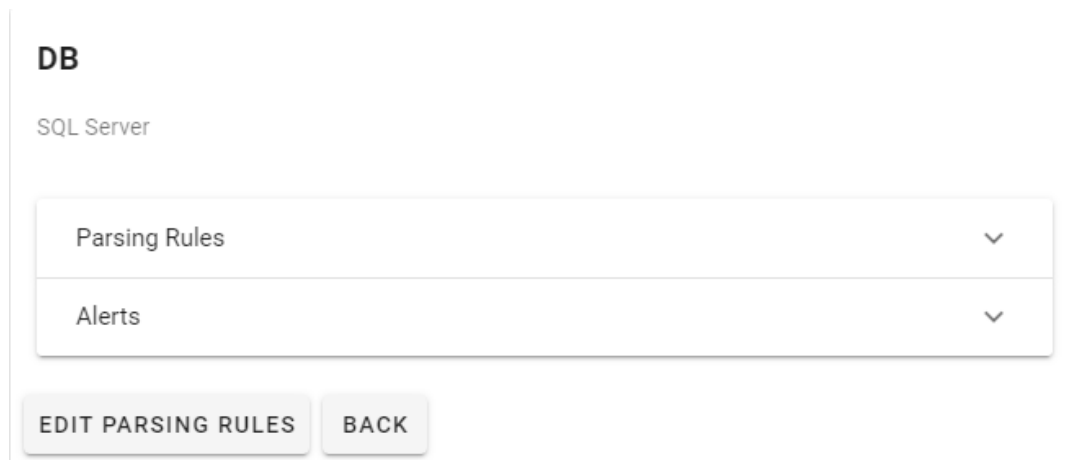
System  
Three Layer System

SUBMIT CANCEL

Slika 7.9 Forma za ažuriranje komponente

Stranica detalja jedne komponente prikazana je na slici 7.10. Stranica je dostupna putem poveznice koja se nalazi pokraj elementa liste komponenti sustava. Stranica prikazuje ime komponente, opis komponente, proširivu karticu za prikaz normalizacijskih pravila dnevnika komponente i proširivu karticu s pravilima značajnih vremenskih intervala komponente. Pri dnu stranice nalazi se gumb „Edit Parsing Rules“ koji pokreće postupak

definiranja normalizacijskih pravila za dnevnik komponente. Gumb „Back“ je poveznica na stranicu s koju je korisnik napustio kako bi došao na stranicu detalja komponente.



Slika 7.10 Stranica detalja komponente

### 7.3. Značajni vremenski interval

Pravila za otkrivanje značajnih vremenski intervali dostupna su na stranici detalja komponente pod proširivom karticom „Alerts“. Kartica „Alerts“ sadrži listu pravila definiranih za komponentu te za svako pravilo prikazuje se popis značajnih intervala otkrivenih prema tome pravilu. Jedno pravilo prikazano je na slici 7.11. Prikazuje se ime pravila, regularni izraz kojim se određuje zadovoljava li zapis pravilo, minimalni prag zapisa koji odgovaraju pravilu unutar vremenskog intervala, vremenski interval pravila u sekundama, radi li se o pravilu konstantne vrijednosti ili o pravilu nekonstantne vrijednosti i svojstvo zapisa čija se vrijednost koristi za utvrđivanje podudaranja s pravilom. Svojstvo zapisa prikazuje ime normalizacijskog pravila koje daje vrijednost tog svojstva, regularni izraz kojim se dobiva vrijednost svojstva, početni učvršćivač (engl. *anchor*) i krajnji učvršćivač. Ispod kartice nalaze se gumb „Edit“ koji otvara formu za ažuriranje pravila i gumb „Delete“ koji briše formu.

## Suspicious login

Value: .\*Login failed for user with id.\*

Threshold: 5

Timespan (in seconds): 120

Is Constant Value: true

Custom Attribute:

### SeriLog Message

Matcher: .+\$

Start Anchor: ]

End Anchor:

EDIT DELETE

Slika 7.11 Primjer pravila za otkrivanje značajnih vremenskih intervala

Slika 7.12 prikazuje formu za stvaranje novog ili ažuriranje postojećeg pravila otkrivanja značajnog vremenskog intervala. Novo pravilo moguće je stvoriti gumbom „Add New Alert“ koji se nalazi na dnu „Alerts“ kartice na stranici detalja komponente. Postojeće pravilo ažurira se gumbom „Edit“ pri dnu kartice koja prikazuje pravilo. Forma sadrži polja za unos imena pravila, opis pravila, regularni izraz koji određuje odgovara li zapis pravilu, minimalni prag pojavljivanja zapisa, gumb za određivanje zahtjeva li pravilo konstantnost vrijednosti unutar vremenskog okvira, padajući izbornik pomoću kojeg se odabire svojstvo zapisa čija vrijednost će se uspoređivati s uvjetom pravila i vremenski interval pravila u sekundama. Gumb „Submit“ pokreće akciju koja stvara navedeno pravilo za komponentu. Gumb „Cancel“ odbacuje unesene podatke i zatvara formu.

Name

Server Client Attack

Description

Alerts when one client does too many requests against server.

Value

.\*Client request id: [\d]+.\*

Threshold

100

☒ Is Value Constant?

Rule on which to alert

Serilog Message

Seconds

30

SUBMIT

CANCEL

Slika 7.12 Forma za stvaranje novog pravila značajnog vremenskog intervala

Slika 7.13 prikazuje primjer otkrivenih značajnih vremenskih intervala. Za svaki interval prikazuje se njegovo početno i završno vrijeme. Gumb „Show Logs“ ispod vremenskog intervala otvara dijalog koji prikazuje normalizirane zapise s postavljenim filterima kako bi se prikazali zapisi relevantni za taj značajni vremenski interval. Postavlja se filter komponente i filter vremenskog raspona. Slika 7.14 prikazuje dijalog sa zapisima otkrivenog značajnog vremenskog intervala.

1/29/2020, 7:07:14 AM - 1/29/2020, 7:08:00 AM

SHOW LOGS

1/29/2020, 7:08:00 AM - 1/29/2020, 7:08:01 AM

SHOW LOGS

1/29/2020, 7:08:01 AM - 1/29/2020, 7:16:06 AM

SHOW LOGS

Slika 7.13 Popis otkrivenih značajnih vremenskih intervala

Component	Timestamp	Severity	Message
<div>Components</div> <div>Server ▼</div>	<div>2020-01-29 07:07:14 ~ 2020-01-29 07:08:00 📅</div>	<div>Severity</div> <div>Debug (+5 others) ▼</div>	<div>Search ...</div>
Server	1/29/2020, 7:07:14 AM	Info	Login failed for user with id 1
Server	1/29/2020, 7:07:56 AM	Info	Login failed for user with id 1
Server	1/29/2020, 7:08:00 AM	Info	Login failed for user with id 1
Server	1/29/2020, 7:08:00 AM	Info	Login failed for user with id 1
Server	1/29/2020, 7:08:00 AM	Info	Login failed for user with id 1

Slika 7.14 Zapisi unutar otkrivenog značajnog vremenskog intervala

## 7.4. Korelacijska grupa

Pregled pravila za korelacijske grupe dostupan je na stranici detalja sustava pod karticom „Correlation Groups“. Klikom na „Correlation Groups“ proširuje se kartica koja prikazuje definirana korelacijska pravila te stvorene grupe za svako pravilo. Kartica također sadrži gumb za dodavanje novog korelacijskog pravila. Postojeće korelacijsko pravilo prikazano je na slici 7.15. Prikazano je ime pravila, opis pravila, vremenski interval pravila u sekundama i pravila značajnih vremenskih intervala te njihove pozicije.

### SQL Attack Attempt

Group that correlates suspicious user queries with server sql exceptions

Timespan (in seconds): 600

Alerts:

Suspicious query	Position: 0
Sql Exception	Position: 1

Slika 7.15 Pravilo korelacijske grupe

Slika 7.16 prikazuje primjer stvorene korelacijske grupe. Za grupu se prikazuje njeno početno i završno vrijeme kao i početna i završna vremena značajnih vremenskih intervala koji čine ovu grupu. Gumb „Show Logs“ otvara dijalog s prikazom normaliziranih zapisa sustava s postavljenim filterom vremenskog intervala. Vremenski interval koji se postavlja kao filter je vremenski interval korelacijske grupe.

### Detected in following timespans:

1/29/2020, 8:27:38 AM - 1/29/2020, 8:36:42 AM  
Suspicious query: 2020-01-29T08:27:38.257 - 2020-01-29T08:27:38.353  
Sql Exception: 2020-01-29T08:36:42.483 - 2020-01-29T08:36:42.583

**SHOW LOGS**

Slika 7.16 Stvorene korelacijske grupe

Slika 7.17 prikazuje formu za stvaranje pravila korelacijske grupe. Novo pravilo korelacijske grupe stvaramo gumbom „Add New Correlation Group“ koji se nalazi na kraju kartice „Correlation Groups“. Forma sadrži polja za unos imena pravila korelacijske grupe, opisa korelacijske grupe, vremenskog okvira korelacijske grupe. Padajućom listom možemo odabrati pravilo otkrivanja značajnih vremenskih intervala, klikom na gumb „Add Alert“ dodajemo odabrano pravilo u formu grupe te mu možemo definirati poziciju unutar grupe.

Correlation Group

Description

Seconds

300

Correlation alerts

DB Online

DB Offline

Position

0

DB Online

Position

1

ADD ALERT

SUBMIT

CANCEL

Slika 7.17 Forma za stvaranje novog pravila korelacijske grupe

## 7.5. Zapisi dnevnika

Pravila za normalizaciju zapisa dnevnika komponente prikazuju se na stranici detalja komponente pod proširivom karticom „Parsing Rules“. Klikom na karticu prikazuju se pravila pridijeljena komponenti. Primjer prikaza svih pravila vidljiv je na slici 7.18. Uvijek su prisutna pravila za obradu vremena, poruke i ozbiljnosti zapisa. Dodatna pravila su pravila za obradu svojstava zapisa definiranih od strane korisnika.

Parsing Rules

**SQL Server DateTime**

Matcher: `[d- :]+`

Start Anchor:

End Anchor: `,`

Date Format: `yyyy-MM-dd HH:mm:ss.ff`

**SQL Server Severity**

Matcher: `.`

Start Anchor:

End Anchor:

Debug: Debug

Trace: Trace

Info: `*`

Warning: Warning

Error: Error

Fatal: Fatal

**SQL Server Message**

Matcher: `.*$`

Start Anchor:

End Anchor:

Max Length: 4096

**Error**

Matcher: Error: `[v]+`

Start Anchor:

End Anchor:

**State**

Matcher: State: `[v]+`

Start Anchor:

End Anchor:

**Severity**

Matcher: Severity: `[v]+`

Start Anchor:

End Anchor:

Slika 7.18 Grupa pravila za obradu zapisa komponente

Sva pravila sadrže sljedeće zajedničke atribute: ime, regularni izraz za dobivanje vrijednosti iz zapisa, početni učvršćivač i završni učvršćivač. Učvršćivači služe kako bi se smanjio zapis na koji se primjenjuje regularni izraz pravila. Dijelovi zapisa prije prve pojave početnog učvršćivača i nakon posljednje pojave krajnjeg učvršćivača se ignoriraju prilikom primjene pravila. Učvršćivači nisu obavezni, a mogu sadržavati jedan ili više znakova. Slika 7.19 prikazuje primjer pravila obrade vremena. Pravilo dodatno prikazuje format vremena koji je dobiven primjenom regularnog izraza.

**SQL Server DateTime**

Matcher: `[d- :]+`

Start Anchor:

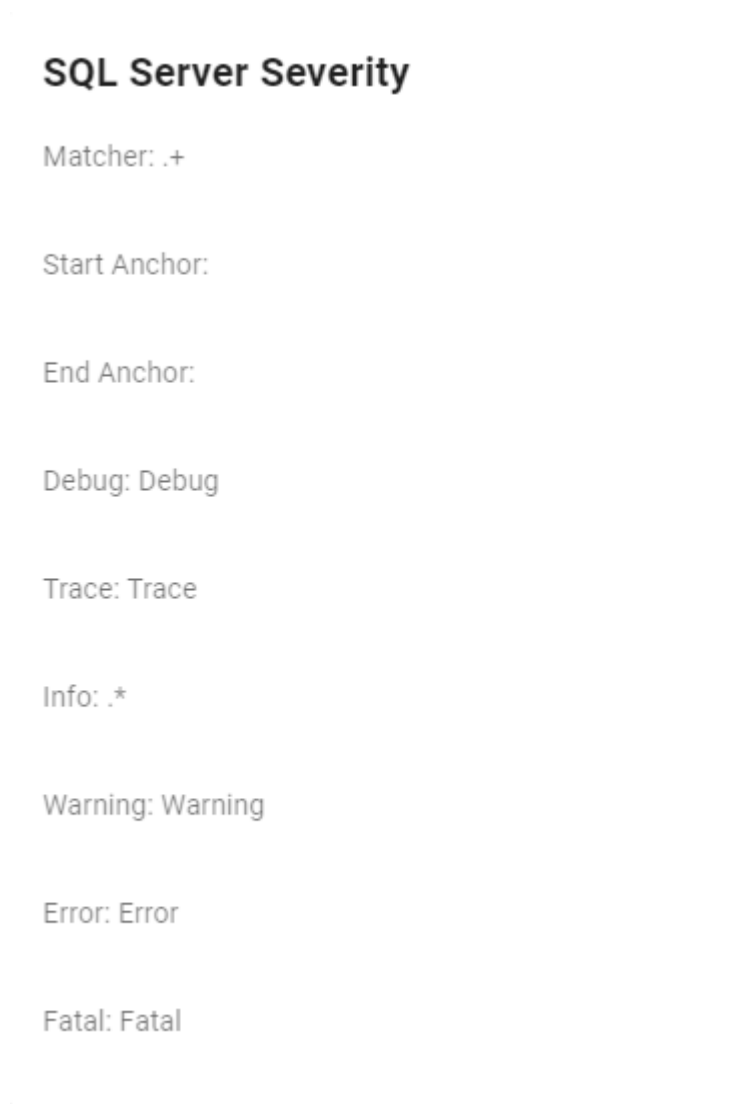
End Anchor: `,`

Date Format: `yyyy-MM-dd HH:mm:ss.ff`

Slika 7.19 Prikaz pravila za obradu vremena događaja



Slika 7.20 prikazuje pravilo za obradu ozbiljnosti iz zapisa. Pravilo dodatno definira regularni izraz za svaku razinu ozbiljnosti. Ozbiljnost događaja utvrđuje se primjenom regularnog izraza razine ozbiljnosti na vrijednost dobivenu pravilom obrade zapisa. Ukoliko vrijednost odgovara regularnom izraz zaključujemo da je događaj te razine ozbiljnosti.



**SQL Server Severity**

Matcher: .+

Start Anchor:

End Anchor:

Debug: Debug

Trace: Trace

Info: .\*

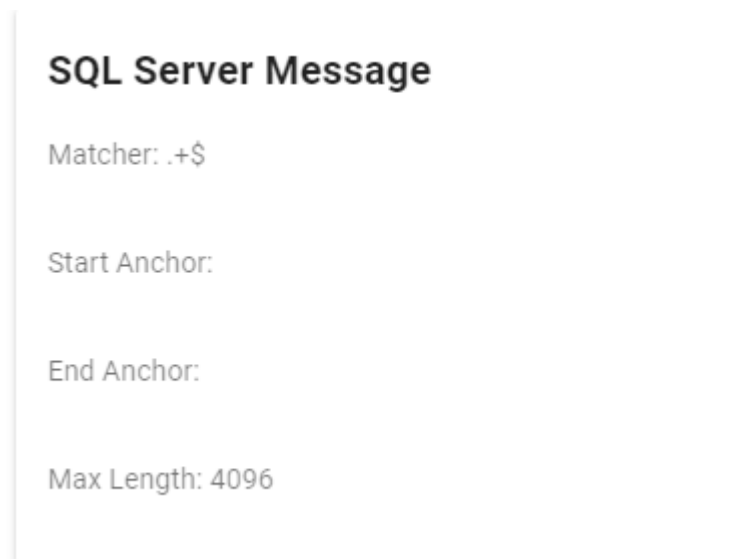
Warning: Warning

Error: Error

Fatal: Fatal

Slika 7.20 Pravilo za obradu ozbiljnosti događaja

Slika 7.21 prikazuje pravilo obrade poruke događaja. Pravilo sadrži dodatno svojstvo za određivanje maksimalne duljine poruke događaja.

The image shows a dialog box titled "SQL Server Message". It contains four labels: "Matcher: .+\$", "Start Anchor:", "End Anchor:", and "Max Length: 4096". The labels are in a light gray font, and the text "Max Length: 4096" is also in light gray. The dialog box has a thin gray border.

Slika 7.21 Pravilo obrade poruke događaja

Pravila obrade dnevnika komponente moguće je definirati gumbom „Parsing Rules Setup“ koji se nalazi pri dnu kartice „Parsing Rules“. Forma za grupu pravila obrade zapisa dnevnika podijeljena je u 4 koraka vidljiva na slici 7.22.



Slika 7.22 Koraci postavljanja pravila obrade zapisa

Korak dodavanja pravila obrade vremena zapisa vidljiv je na slici 7.23. Korak se sastoji od padajućeg izbornika pomoću kojeg možemo odabrati postojeće pravilo i gumba „+“ kojim dodajemo novo pravilo. Nakon odabira pravila, podatci pravila su prikazani ispod padajućeg izbornika zajedno s gumbom „Edit“ pomoću kojeg možemo urediti pravilo. Dno stranice sadrži tekstualno polje u koje je moguće unijeti primjer zapisa te gumbom „Test Rule“ isprobavamo odabrano pravilo. Rezultat obrade pravila vidljiv je unutar polja „Result“. Gumb „Continue“ korisnika vodi na sljedeći korak, a gumb „Cancel“ odbacuje promijene i prekida postupak.

Date Time Rule  
SQL Server DateTime

SQL Server DateTime

Matcher: [\d-:.]+

Start Anchor:

End Anchor: ,

Date Format: yyyy-MM-dd HH:mm:ss.ff

EDIT

CONTINUE

CANCEL

Input log line on which you wish to test out current rule

2020-01-30 02:28:52.47 spid20s    Error: 17054, Severity: 16, State: 1.

Result

2020-01-30T02:28:52.47

TEST RULE

Slika 7.23 Korak dodavanja pravila obrade vremena

Forme za stvaranje novog pravila obrade vremena događaja i uređivanje postojećeg su iste, jedina razlika je u početnim podacima forme. Slika 7.24. prikazuje formu pravila za obradu vremena događaja. Forma sadrži polja za unos imena, opisa, regularnog izraza za podudaranje, početnog učvršćivača, krajnjeg učvršćivača i formata vremena. Gumb „Submit“ pokreće odgovarajuću akciju ovisno o tome radi li se o stvaranju novog pravila ili ažuriranju postojećeg. Gumb „Cancel“ odbacuje promijene i zatvara formu.

Name  
Standard date time format

---

Description  
Describes year first date time format that is anchored between square brackets.

---

Matcher  
[\\d- :]+

---

Start anchor  
[

---

End Anchor  
]

---

yyyy-MM-dd HH:mm:ss

**SUBMIT** **CANCEL**

Slika 7.24 Forma za pravilo obrade vremena

Drugi korak vidljiv je na slici 7.25 i sastoji se od istih dijelova kao i prošli korak s time da je jedina razlika u tome da ovaj korak služi za definiciju pravila obrade poruke događaja.

Message Rule  
SQL Server Message

**SQL Server Message**

Matcher: .+\$

Start Anchor:

End Anchor:

Max Length: 4096

EDIT

CONTINUE

CANCEL

Input log line on which you wish to test out current rule  
2020-01-30 02:28:52.47 spid20s Error: 17054, Severity: 16, State: 1.

Result  
Error: 17054, Severity: 16, State: 1.

TEST RULE

Slika 7.25 Korak dodavanja pravila obrade poruke

Slika 7.26 prikazuje formu za stvaranje ili uređivanje pravila obrade poruke događaja. Forma sadrži polja za unos imena, opisa, regularnog izraza, početnog i krajnjeg učvršćivača te maksimalne duljine poruke. Gumb „Submit“ pokreće odgovarajuću akciju ovisno o tome radi li se o stvaranju novog pravila ili ažuriranju postojećeg. Gumb „Cancel“ odbacuje promijene i zatvara formu.

Name  
End of line message

---

Description  
Message that starts after delimiter and goes to the end of line.

---

Matcher  
.+\$

---

Start anchor  
]

---

End Anchor

---

10000

---

**SUBMIT** **CANCEL**

Slika 7.26 Forma za pravilo obrade poruke događaja

Treći korak služi za odabir pravila obrade ozbiljnosti događaja. Sastoji se od istih dijelova kao i prošli koraci. Slike 7.27 i 7.28 prikazuju ovaj korak.

Severity Rule  
SQL Server Severity

**SQL Server Severity**

Matcher: .+

Start Anchor:

End Anchor:

Debug: Debug

Trace: Trace

Info: .\*

Warning: Warning

Error: Error

Fatal: Fatal

EDIT CONTINUE CANCEL

Slika 7.27 Korak dodavanja pravila obrade ozbiljnosti

Input log line on which you wish to test out current rule

2020-01-30 02:28:52.47 spid20s Error: 17054, Severity: 16, State: 1.

Result

Error

TEST RULE

Slika 7.28 Korak dodavanja pravila ozbiljnosti - primjer zapisa

Forma za stvaranje ili uređivanje pravila obrade ozbiljnosti događaja prikazana je na slici 7.29. Forma se sastoji od unosa imena, opisa, regularnog izraza, početnog i krajnjeg učvršćivača te regularnog izraza za svaku razinu ozbiljnosti. Gumb „Submit“ pokreće odgovarajuću akciju ovisno o tome radi li se o stvaranju novog pravila ili ažuriranju postojećeg. Gumb „Cancel“ odbacuje promijene i zatvara formu.

Name  
Uppercase letter severity

---

Description  
Severity that is written with one uppercase letter between spaces

---

Matcher  
[s].[s]

---

Start anchor

---

End Anchor

---

D

---

T

---

I

---

W

---

E

---

F

---

**SUBMIT** **CANCEL**

Slika 7.29 Forma za pravilo obrade ozbiljnosti događaja

Posljednji korak prikazan je na slici 7.30 i sastoji se od odabira proizvoljnih pravila za obradu zapisa. U posljednjem koraku moguće je odabrati 0 ili više pravila, svako odabrano pravilo moguće je ažurirati i isprobati. Postavljanje pravila obrade zapisa završava se klikom na gumb „Complete Setup“.



Custom attribute Rules  
Error, State, Severity

---

**Error**

Matcher: Error: [\d]+

Start Anchor:

End Anchor:

---

**EDIT**

Input log line on which you wish to test out current rule  
2020-01-30 02:28:52.47 spid20s Error: 17054, Severity: 16, State: 1.

---

Result  
Error: 17054

---

**TEST RULE**

Slika 7.30 Korak dodavanja vlastitih pravila

Slika 7.31 prikazuje formu za stvaranje ili uređivanje pravila korisničkog svojstva. Forma se sastoji od atributa zajedničkih svim pravilima. Gumb „Submit“ pokreće odgovarajuću akciju ovisno o tome radi li se o stvaranju novog pravila ili ažuriranju postojećeg. Gumb „Cancel“ odbacuje promijene i zatvara formu.

Name  
Custom attribute

---

Description  
Parsed user id attribute

---

Matcher  
.\*id [\\d]+\\\$

---

Start anchor  
:

---

End Anchor

---

**SUBMIT** **CANCEL**

Slika 7.31 Forma pravila korisničkog svojstva

Normalizirane zapise cijelog sustava moguće je vidjeti na stranici detalja sustava pod karticom „Normalized Logs“. Kartica se sastoji od tabličnog prikaza zapisa. Stupci tablice su komponenta, vrijeme događaja, ozbiljnost događaja i poruka događaja. Aplikacija dohvaća ograničeni broj zapisa za prikaz, ukoliko korisnik želi prikazati više zapisa to može učiniti gumbom „Load More“ koji se nalazi na dnu prikaza zapisa. Slika 7.32 prikazuje normalizirane zapise i gumb „Load More“-

Client	1/29/2020, 7:36:35 AM	Info	Querying users with category guest
Client	1/29/2020, 7:37:29 AM	Info	Querying users with category ' ; drop table [sawmill.db].[dbo].[User] --
Client	1/29/2020, 7:37:41 AM	Info	Querying users with category admin
LOAD MORE			

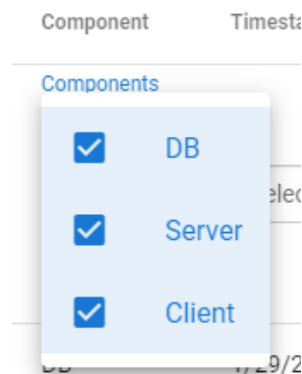
Slika 7.32 Normalizirani zapisi

Normalizirane zapise moguće je filtrirati prema komponenti kojoj pripadaju, vremenu događaja, ozbiljnosti i poruci događaja. Zaglavlje tablice sadrži polja za unos uvjeta filtriranja i prikazano je na slici 7.33.



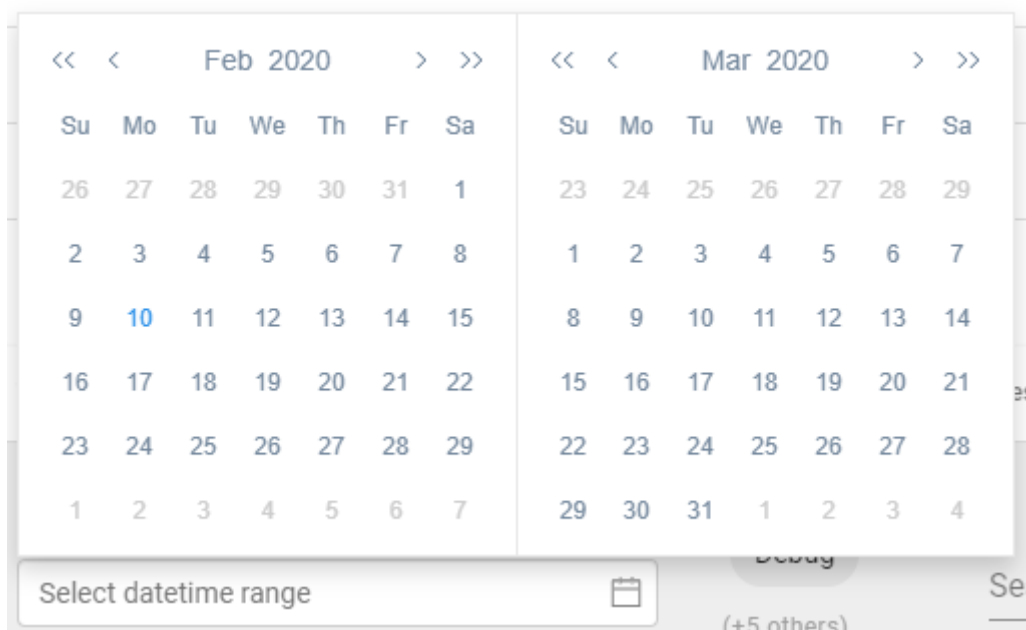
Slika 7.33 Zaglavlje tablice normaliziranih zapisa

Slika 7.34 prikazuje postavljanje filtera po komponenti. Korisniku se prikazuje padajući izbornik u kojem može označiti komponente čije zapise želi uključiti u prikaz.



Slika 7.34 Filtriranje po komponenti

Slika 7.35 prikazuje postavljanje filtera po datumu. Korisniku se prikazuju dva kalendara, odabir na lijevom kalendaru predstavlja početni datum intervala, a odabir na desnom kalendaru krajnji datum intervala. Nakon što je korisnik odabrao oba datuma dodatno može zadati vrijeme unutar dana odabranih dana što je prikazano na slici 7.36.

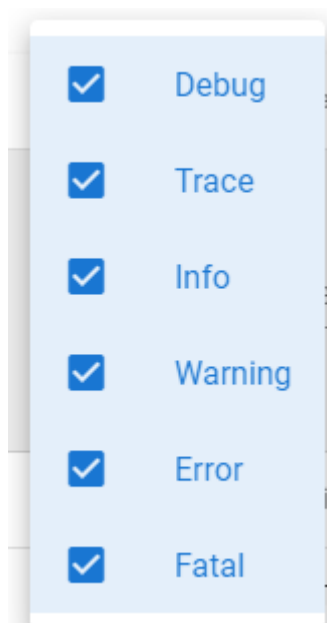


Slika 7.35 Filtriranje po datumu

2020-02-04			2020-03-10		
02	04	03	03	03	02
03	05	04	04	04	03
04	06	05	05	05	04
05	07	06	06	06	05
06	08	07	07	07	06
07	09	08	08	08	07
08	10	09	09	09	08

Slika 7.36 Filtriranje po vremenu

Slika 7.37 prikazuje postavljanje uvjeta za filtriranje po ozbiljnosti događaja. Korisnik iz padajućeg izbornika odabire koje razine ozbiljnosti želi uključiti u prikaz zapisa.



Slika 7.37 Filtriranje po ozbiljnosti događaja

Slika 7.38 prikazuje unos uvjeta za filtriranje prema poruci događaja. Nakon postavljanja ovog uvjeta prikazuju se samo zapisi čija poruka sadrži uneseni tekst.

## Message

[Search Message](#)

Authentication

Slika 7.38 Filtriranje po poruci događaja

„Log Input“ kartica stranice detalja sustava sadrži formu za ručni unos zapisa dnevnika. Forma je podijeljena u dodatne kartice (engl. *tabs*) gdje svaka kartica predstavlja komponentu sustava. Forma sadrži mogućnosti dvije vrste unosa zapisa. Prvi način unosa je tekstualni i prikazan je na slici 7.39.


Log Input

DB	SERVER	CLIENT
<div>Log</div> <div>2020-01-29 08:27:38.052 +01:00 [INF] Querying users with category guest</div> <div>2020-01-29 08:27:38.256 +01:00 [INF] Querying users with category ' ; drop table [sawmill.db].[dbo].[User] --</div> <div>2020-01-29 08:27:38.352 +01:00 [INF] Querying users with category admin</div>		

Slika 7.39 Forma za ručni unos zapisa

Drugi način unosa je odabirom datoteke i prikazan je na slici 7.40. Promjenu načina unosa moguće je napraviti pritiskom gumba „Switch Input“.

[Log File Input](#)

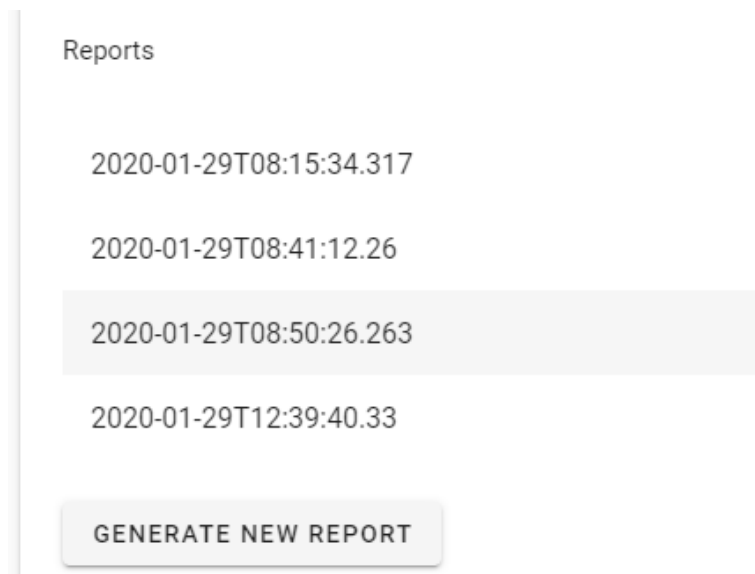
 client20200204.txt (512 B)

SWITCH INPUT SUBMIT

Slika 7.40 Unos zapisa odabirom datoteke

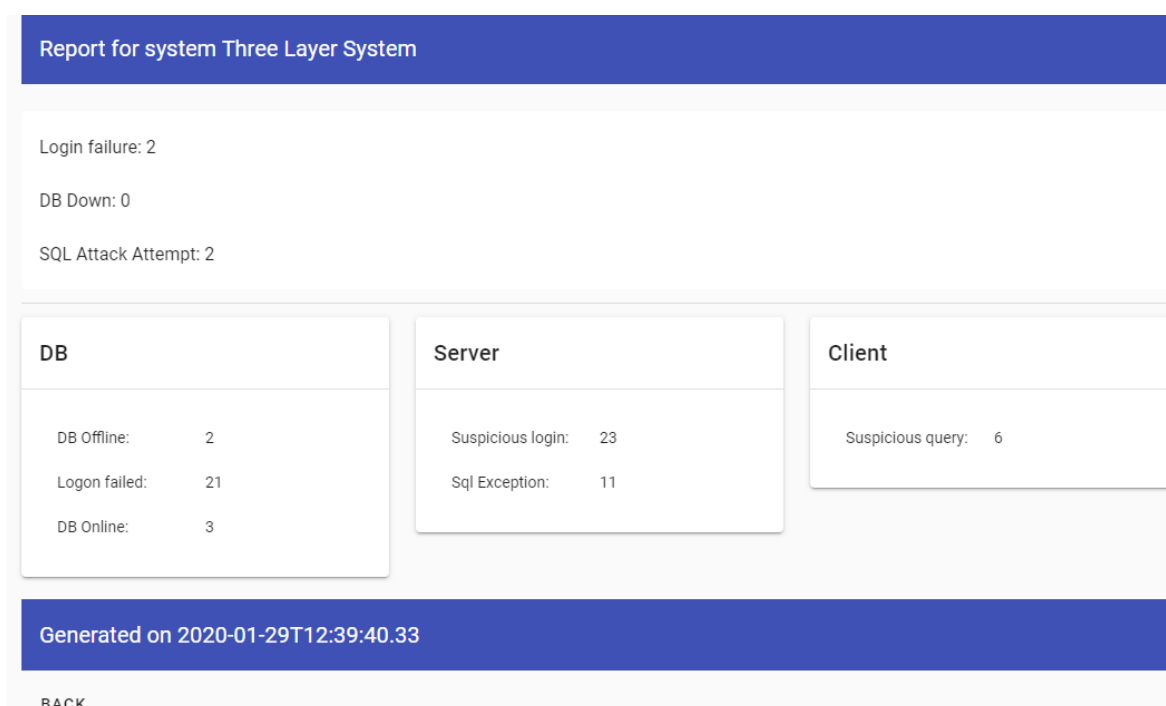
## 7.6. Izvještaj sustava

Kartica „Reports“ stranice detalja sustava sadrži popis generiranih izvještaja. Gumbom „Generate New Report“ pokrenuti će se akcija stvaranja novog izvještaja za trenutačni sustav. Stvoreni izvještaj prikazivati će ažurno stanje učestalosti pojavljivanja korelacijskih grupa i tipova značajnih vremenskih intervala. Popis izvještaja vidljiv je na slici 7.41.



Slika 7.41 Kartica s izvještajima

Slika 7.42 prikazuje jedan izvještaj sustava. Gornji dio izvještaja prikazuje učestalost pojedine korelacijske grupe u sustavu. Donji dio izvještaja podijeljen je na komponente sustava i za svaku komponentu prikazuje se učestalost otkrivenih tipova značajnih vremenskih intervala. Gumb „Back“ korisnika vraća na stranicu detalja sustava.

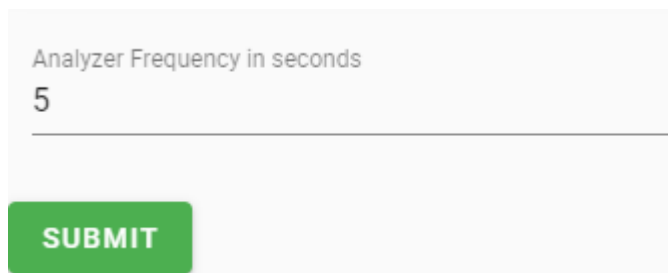


Slika 7.42 Izvještaj sustava

## 7.7. Postavke analizatora

Stranica za konfiguriranje postavi analizatora dostupna je putem navigacijskog izbornika odabirom opcije „Analyzer Settings“. Jedina postavka analizatora koju je moguće ažurirati

je vrijeme izvođenja analizatora. Slika 7.43 prikazuje formu za ažuriranje postavke analizatora. Vrijeme se zadaje u sekundama i pritiskom na gumb „Submit“ postaje aktivno. Postavka određuje periodičnost izvođenja analizatora za stvaranje korelacijskih grupa.

The image shows a web form with a light gray background. At the top, the text "Analyzer Frequency in seconds" is displayed in a small, gray font. Below this text, the number "5" is entered into a text input field. A horizontal line separates the input field from the rest of the form. At the bottom of the form, there is a green rectangular button with the word "SUBMIT" written in white, uppercase letters.

Slika 7.43 Postavka učestalosti izvođenja analizatora

## 8. Korištene tehnologije i alati

Razvoj programskog rješenja je složeni proces koji je zahtjeva uporabu više alata i tehnologija kako bi bio uspješno obavljen.

### 8.1. Alati

#### 8.1.1. Visual Studio Community 2017

Visual Studio je integrirano razvojno okruženje (engl. *Integrated development environment*, skraćeno IDE) razvijeno od strane Microsofta. Koristi se za razvoj mrežnih aplikacija, mrežnih servisa, mobilnih aplikacija te ostalih računalnih programa. Podržava razvoj u više programskih jezika [14]. Razvojno okruženje korišteno je za razvoj servisa za prikupljanje dnevnika (*SawMill.Collector*), knjižnice za pristup podacima (*SawMill.Data*), knjižnice za obradu podataka (*SawMill.Processor*) te mrežnog programibilnog sučelja (*SawMill.WebApi*). Za razvoj navedenih komponenti korišten je C# programski jezik.

#### 8.1.2. Visual Studio Code

Visual Studio Code je uređivač teksta (engl. *text editor*) otvorenog koda za Windows, macOS i Linux operacijske sustave [15]. Korišten je za razvoj klijentske mrežne aplikacije (*SawMill.Frontend*).

#### 8.1.3. Microsoft SQL Server Managment Studio 2017

SQL Server Managment Studio je integrirano okruženje za konfiguriranje, nadziranje i administriranje instanci SQL Servera i baza podataka [16]. Integrirano okruženje korišteno je za stvaranje i upravljanje relacijske baze podataka korištene u programskom rješenju.

### 8.2. Tehnologije

#### 8.2.1. ASP.NET Core 2.2

ASP.NET Core je više-platformski, performantni radni okvir otvorenog koda za izgradnju aplikacija povezanih internetom [17]. Radni okvir korišten je za razvoj aplikacijskog programibilnog sučelja.

#### 8.2.2. RestSharp

RestSharp je knjižnica za .NET koja nudi mogućnosti HTTP (engl. *HyperText Transfer Protocol*) klijenta. Podržava automatsku serijalizaciju i deserijalizaciju podataka, otkrivanje tipa zahtjeva i odgovora, parametrizaciju zahtjeva te brojne druge mogućnosti [18].



RestSharp knjižnica korištena je u servisu za prikupljanje dnevnika (*SawMill.Collector*) kako bi se prikupljeni dnevnici prenijeli na poslužitelj.

### 8.2.3. Topshelf

Topshelf je radni okvir za stvaranje Windows servisa pomoću .NET platforme. Radni okvir nam omogućava pisanje konzolne aplikacije te jednu klasu koja konfigurira radni okvir, a nakon toga Topshelf se pobrine za instalaciju samog servisa [19]. Topshelf je korišten za razvoj prikupljača dnevnika (*SawMill.Collector*) kako bi se on mogao izvoditi kao servis.

### 8.2.4. Vue.js

Vue je progresivni, reaktivni radni okvir (engl. *progressive reactive framework*) za izgradnju korisničkih sučelja [20]. Podržava jezike JavaScript i njegov nad skup TypeScript. Progresivan radni okvir donosi dodatno označavanje samom HTMLu, a reaktivni radni okvir donosi automatsko osvježavanje korisničkog sučelja prilikom ažuriranja podataka na koje je određeni dio korisničkog sučelja vezan.

### 8.2.5. Vuex

Vuex je knjižnica za Vue.js koja implementira obrazac za upravljanje stanjem aplikacije. Vuex nudi središnje mjesto za pohranu podataka koji su dijeljeni među komponentama mrežne aplikacije te pravila koja osiguravaju da je stanje aplikacije ažurirano na predvidivi način [11].

### 8.2.6. Vue Router

Vue Router je knjižnica za Vue.js te je njegov službeni usmjerivač (engl. *router*) s mogućnostima preslikavanja navigacijskih ruta na Vue komponente. Neke od dodatnih mogućnosti ove knjižnice su ugniježdene rute, konfiguracija temeljena na komponentama, parametrizirane rute, poveznice s automatski dodanim CSS razredima i povijest navigacije kroz aplikaciju [21].

### 8.2.7. Vuetify

Vuetify je knjižnica komponenti za izgradnju korisničkih sučelja za Vue.js [22]. Komponente koje nudi Vuetify temelje se na specifikaciji materijalnog dizajna (engl. *material design specification*) [23].

### **8.2.8. Axios**

Axios je JavaScript knjižnica koja implementira HTTP klijenta temeljenog na obećanjima (engl. *promise based HTTP client*). Namijenjena je za korištenje iz mrežnog preglednika i Node.js radnog okruženja (engl. *runtime*) [24].

### **8.2.9. NuGet**

NuGet je upravitelj paketima (engl. *package manager*) za .NET platformu. Sastoji se od klijentskih aplikacija i NuGet galerije. Galerija je središnji repozitorij paketa, a pomoću klijentskih aplikacija moguće je proizvesti i konzumirati pakete [25]. NuGet je korišten prilikom razvoja svih dijelova sustava koji koriste .NET platformu.

### **8.2.10. NPM**

NPM je upravitelj paketima za JavaScript programski jezik. Sastoji se od konzolne klijentske aplikacije i online baze podataka javnih paketa. Pomoću klijentske aplikacije moguće je pretraživati pakete i upravljati istima [26]. NPM upravitelj paketima korišten je prilikom razvoja mrežne klijentske aplikacije (*SawMill.Frontend*)

### **8.2.11. Microsoft SQL Server**

Microsoft SQL Server je sustav za upravljanje relacijskim bazama podataka. SQL Server je korišten kako baza podataka programskog rješenja.

## 9. Zaključak

Informacijski sustavi sastavljeni od više komponenti predstavljaju izazov kod analize pogrešaka sustava otkrivenih u produkcijskom radu sustava. Osoba koja provodi istragu mora istraživati dnevnik različite formate što otežava i ručnu analizu i proces automatizacije analize.

U ovom radu prikazan je problem kompleksnosti analize dnevnika različitih komponenti informacijskog sustava s ciljem otkrivanja uzroka prijavljene pogreške. Predložen je postupak svođenja različitih formata dnevnika u normalizirani oblik. Pojašnjen je primjer formata dnevnika SQL Servera. Predložen je postupak korelacije događaja temeljen na korisničkim pravilima. Postupak korelacije odvija se u dva koraka, prvi korak pronalazi skup događaja koji predstavljaju značajni vremenski interval unutar dnevnika jedne komponente. Drugi korak stvara korelacijske grupe između komponenti primjenjujući pravila na otkrivene značajne vremenske intervale.

Razvijena je programska podrška koja korisniku omogućava opisivanje formata dnevnika komponente kako bi se mogao normalizirati. Definiranje pravila za otkrivanje značajnih vremenskih intervala unutar jedne komponente. Povezivanje komponenti u informacijski sustav. Definiranje pravila za stvaranje korelacijskih grupa od otkrivenih značajnih vremenskih intervala. Generiranje izvještaja o učestalosti tipova korelacijskih grupa i tipova značajnih vremenskih intervala komponenti sustava.

## Sažetak

**Naslov:** Analizator dnevnika informacijskog sustava

**Sažetak:** U radu je opisan problem različitih formata dnevnika komponenti informacijskog sustava i predložen je postupak normalizacije dnevnika informacijskog sustava. Predloženi su algoritmi za korelaciju događaja temeljeni na korisničkim pravilima. Specifikacija zahtjeva temelji se na domeni problema i analizi postojećih rješenja. Model podatka razvijen je kako bi podržao specificirane zahtjeve. Opisana je arhitektura sustava koja slijedi principe čiste arhitekture, te definira slojeve koji izoliraju ovisnosti dijelova izvornog koda. Prikazana je funkcionalnost sustava na primjeru informacijskog sustava s tri komponente. Opisane su tehnologije i alati korišteni tijekom razvoja sustava.

**Ključne riječi:** Programska podrška, dnevnik, korelacija događaja, web aplikacija

# Abstract

**Title:** Information system log analyser.

**Summary:** This thesis describes problems with different log formats of information system components and suggests process of normalization for information system logs. Paper presents algorithms for event correlation based on user defined rules. Requirement specification is based on domain problems and analysis of existing solutions. Domain model is developed so it can support specified requirements. Paper gives overview of system architecture that follows principles of clean architecture and defines layers that achieve separation of concerns in source code dependencies. System functionality is presented on information system with three components. Paper gives overview of technologies and tools used in development process.

**Keywords:** Software, log, event correlation, web application

## Literatura

- [1] ManageEngine, »Discover complex attack patterns with event log correlation,« [Mrežno]. Dostupno na: <https://www.manageengine.com/products/eventlog/event-correlation.html>. [Pokušaj pristupa 9 Veljače 2020].
- [2] Loggly, »Product overview,« [Mrežno]. Dostupno na: <https://www.loggly.com/product/>. [Pokušaj pristupa 9 Veljače 2020].
- [3] I. Sommerville, Software Engineering, Harlow: Pearson Education Limited, 2007.
- [4] B. Woody, »SQL Server Error Logs,« 28 Ožujka 2003. [Mrežno]. Dostupno na: <http://www.informit.com/articles/article.aspx?p=31359&seqNum=242>. [Pokušaj pristupa 3 Veljača 2020].
- [5] Microsoft, »Understanding Database Engine Errors,« Microsoft, 16 Ožujka 2017. [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/understanding-database-engine-errors?view=sql-server-ver15>. [Pokušaj pristupa 3 Veljače 2020].
- [6] Microsoft, »Database Engine Error Severities,« Microsoft, 16 Ožujka 2017. [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/sql/relational-databases/errors-events/database-engine-error-severities?view=sql-server-ver15>. [Pokušaj pristupa 3 Veljače 2020].
- [7] R. Martin, »The Clean Architecture,« 13 Kolovoza 2012. [Mrežno]. Dostupno na: <https://blog.cleancoder.com/uncle-bob/2012/08/13/the-clean-architecture.html>. [Pokušaj pristupa 2 Veljače 2020].
- [8] Microsoft, »Asynchronous programming with async and await,« 18 Ožujka 2019. [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/async/>. [Pokušaj pristupa 5 Veljače 2020].

- [9] Microsoft, »Dependency injection in ASP.NET Core,« 30 Siječnja 2020. [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/dependency-injection?view=aspnetcore-2.2>. [Pokušaj pristupa 3 Veljače 2020].
- [10] Microsoft, »Background tasks with hosted services in ASP.NET Core,« [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/aspnet/core/fundamentals/host/hosted-services?view=aspnetcore-3.1&tabs=visual-studio>. [Pokušaj pristupa 3 Veljače 2020].
- [11] Vuex, »What is Vuex?,« Vuex, [Mrežno]. Dostupno na: <https://vuex.vuejs.org/>. [Pokušaj pristupa 3 Veljače 2020].
- [12] Vue.js, »Components Basics,« [Mrežno]. Dostupno na: <https://vuejs.org/v2/guide/components.html>. [Pokušaj pristupa 5 Veljače 2020].
- [13] Nepoznato, »INI file,« [Mrežno]. Dostupno na: [https://en.wikipedia.org/wiki/INI\\_file](https://en.wikipedia.org/wiki/INI_file). [Pokušaj pristupa 4 Veljače 2020].
- [14] Microsoft, »Visual Studio,« [Mrežno]. Dostupno na: <https://visualstudio.microsoft.com/>. [Pokušaj pristupa 3 Veljače 2020].
- [15] Microsoft, »Visual Studio Code,« Microsoft, [Mrežno]. Dostupno na: <https://code.visualstudio.com/>. [Pokušaj pristupa 3 Veljače 2020].
- [16] Microsoft, »Download SQL Server Management Studio (SSMS),« Microsoft, [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/sql/ssms/download-sql-server-management-studio-ssms?view=sql-server-ver15>. [Pokušaj pristupa 3 Veljače 2020].
- [17] Microsoft, »Introduction to ASP.NET Core,« Microsoft, [Mrežno]. Dostupno na: <https://docs.microsoft.com/en-us/aspnet/core/?view=aspnetcore-3.1>. [Pokušaj pristupa 3 Veljače 2020].
- [18] J. Sheehan, A. Young i A. Zimarev, »RestSharp,« RestSharp, [Mrežno]. Dostupno na: <http://restsharp.org/>. [Pokušaj pristupa 3 Veljače 2020].

- [19] C. Patterson, T. Smith i D. Sellers, »Topshelf Overview,« Topshelf, [Mrežno]. Dostupno na: <https://topshelf.readthedocs.io/en/latest/overview/index.html>. [Pokušaj pristupa 3 Veljače 2020].
- [20] Vue.js, »What is Vue.js?,« Vue.js, [Mrežno]. Dostupno na: <https://vuejs.org/v2/guide/>. [Pokušaj pristupa 3 Veljače 2020].
- [21] Vue.js, »Introduction,« Vue.js, [Mrežno]. Dostupno na: <https://router.vuejs.org/>. [Pokušaj pristupa 3 Veljače 2020].
- [22] Vuetify, »Material Design Component Framework,« Vuetify, [Mrežno]. Dostupno na: <https://vuetifyjs.com/en/>. [Pokušaj pristupa 3 Veljače 2020].
- [23] Material Design, »Design,« [Mrežno]. Dostupno na: <https://material.io/design/>. [Pokušaj pristupa 3 Veljače 2020].
- [24] Axios, »axios,« Axios, [Mrežno]. Dostupno na: <https://github.com/axios/axios#axios>. [Pokušaj pristupa 3 Veljače 2020].
- [25] Microsoft, »What is NuGet?,« Microsoft, [Mrežno]. Dostupno na: <https://www.nuget.org/>. [Pokušaj pristupa 3 Veljače 2020].
- [26] npm, Inc., »About npm,« npm, Inc., [Mrežno]. Dostupno na: <https://docs.npmjs.com/about-npm/>. [Pokušaj pristupa 3 Veljače 2020].



## Dodatak: Popis slika

Slika 2.1 Prikaz zapisa dnevnika [1] .....	2
Slika 2.2 Dodavanje novog korelacijskog pravila [1] .....	3
Slika 2.3 Predefinirana upozorenja [1] .....	3
Slika 2.4 Izvještaj događaja [1] .....	4
Slika 2.5 Prikaz zapisa dnevnika [2] .....	4
Slika 2.6 Pretraga zapisa - uvjet pretrživanja [2] .....	5
Slika 2.7 Pretraga zapisa - vremenski raspon [2] .....	5
Slika 2.8 Definiranje vlastitog svojstva atributa [2] .....	5
Slika 2.9 Forma za stvaranje novog upozorenja [2] .....	6
Slika 2.10 Obavijest upozorenja [2] .....	7
Slika 6.1 Slojevi čiste arhitekture i pravilo ovisnosti [7] .....	19
Slika 6.2 Arhitektura sustava .....	20
Slika 6.3 Model baze podataka .....	21
Slika 6.4 Tablica Alert .....	22
Slika 6.5 Tablica AlertGroup .....	22
Slika 6.6 Tablica AlertGroupValue .....	23
Slika 6.7 Tablica AlertGroupAlert .....	23
Slika 6.8 Tablica AlertValue .....	24
Slika 6.9 Tablica Component .....	25
Slika 6.10 Tablica ComponentReport .....	25
Slika 6.11 Tablica ComponentReportAlert .....	26
Slika 6.12 Tablica CustomAttributeRuleParsingRules .....	26
Slika 6.13 Tablica CustomAttributeValue .....	27
Slika 6.14 Tablica GeneralRule .....	27
Slika 6.15 Tablica DateTimeRule .....	28
Slika 6.16 Tablica MessageRule .....	28
Slika 6.17 Tablica SeverityLevelRule .....	29
Slika 6.18 Tablica SeverityLevel .....	29
Slika 6.19 Tablica ParsingRules .....	30
Slika 6.20 Tablica RawLog .....	30
Slika 6.21 Tablica NormalizedLog .....	31
Slika 6.22 Tablica Settings .....	31

Slika 6.23 Tablica System .....	31
Slika 6.24 Tablica SystemReport .....	32
Slika 6.25 Tablica SystemReportAlertGroup .....	32
Slika 6.26 Komponente poslužitelja .....	33
Slika 6.27 Registracija razreda .....	34
Slika 6.28 Dijagram razreda upravljača .....	35
Slika 6.29 Upravljač za komponentu .....	36
Slika 6.30 Bazno sučelje prezentera .....	36
Slika 6.31 Dijagram prezentera .....	37
Slika 6.32 Dijagram razreda modela pogleda .....	37
Slika 6.33 Isječak koda za registraciju servisa .....	38
Slika 6.34 Dijagram razreda pozadinskog servisa .....	38
Slika 6.35 Dijagram razreda servisa .....	40
Slika 6.36 Bazno sučelje sučelja servisa .....	41
Slika 6.37 Dijagram razreda poslovnih modela .....	42
Slika 6.38 Dijagram sučelja repozitorija .....	43
Slika 6.39 Dijagram podatkovnih modela .....	44
Slika 6.40 Konfiguracija pretvaranja Component podatkovnog modela .....	44
Slika 6.41 Dijagram pretvarača .....	45
Slika 6.42 Arhitektura web aplikacije [11] .....	47
Slika 6.43 Struktura web aplikacije .....	48
Slika 7.1 Navigacijski izbornik .....	53
Slika 7.2 Početna stranica aplikacije .....	53
Slika 7.3 Detaljniji prikaz kartice jednog sustava .....	54
Slika 7.4 Forma za stvaranje sustava .....	55
Slika 7.5 Padajući izbornik za izbor komponenti .....	55
Slika 7.6 Forma za uređivanje sustava .....	56
Slika 7.7 Stranica detalja sustava .....	57
Slika 7.8 Forma za stvaranje komponente .....	58
Slika 7.9 Forma za ažuriranje komponente .....	58
Slika 7.10 Stranica detalja komponente .....	59
Slika 7.11 Primjer pravila za otkrivanje značajnih vremenskih intervala .....	60
Slika 7.12 Forma za stvaranje novog pravila značajnog vremenskog intervala .....	61
Slika 7.13 Popis otkrivenih značajnih vremenskih intervala .....	61

Slika 7.14 Zapisi unutar otkrivenog značajnog vremenskog intervala.....	62
Slika 7.15 Pravilo korelacijske grupe .....	62
Slika 7.16 Stvorene korelacijske grupe .....	63
Slika 7.17 Forma za stvaranje novog pravila korelacijske grupe .....	64
Slika 7.18 Grupa pravila za obradu zapisa komponente .....	65
Slika 7.19 Prikaz pravila za obradu vremena događaja.....	65
Slika 7.20 Pravilo za obradu ozbiljnosti događaja .....	66
Slika 7.21 Pravilo obrade poruke događaja .....	67
Slika 7.22 Koraci postavljanja pravila obrade zapisa.....	67
Slika 7.23 Korak dodavanja pravila obrade vremena.....	68
Slika 7.24 Forma za pravilo obrade vremena .....	69
Slika 7.25 Korak dodavanja pravila obrade poruke .....	70
Slika 7.26 Forma za pravilo obrade poruke događaja .....	71
Slika 7.27 Korak dodavanja pravila obrade ozbiljnosti.....	72
Slika 7.28 Korak dodavanja pravila ozbiljnosti - primjer zapisa .....	72
Slika 7.29 Forma za pravilo obrade ozbiljnosti događaja .....	73
Slika 7.30 Korak dodavanja vlastitih pravila.....	74
Slika 7.31 Forma pravila korisničkog svojstva .....	75
Slika 7.32 Normalizirani zapisi .....	75
Slika 7.33 Zaglavlje tablice normaliziranih zapisa.....	76
Slika 7.34 Filtriranje po komponenti .....	76
Slika 7.35 Filtriranje po datumu .....	76
Slika 7.36 Filtriranje po vremenu .....	77
Slika 7.37 Filtriranje po ozbiljnosti događaja.....	77
Slika 7.38 Filtriranje po poruci događaja .....	78
Slika 7.39 Forma za ručni unos zapisa .....	78
Slika 7.40 Unos zapisa odabirom datoteke.....	78
Slika 7.41 Kartica s izvještajima .....	79
Slika 7.42 Izvještaj sustava.....	79
Slika 7.43 Postavka učestalosti izvođenja analizatora.....	80

## **Dodatak: Popis isječaka koda**

Isječak koda 4.1 Zapis dnevnika .....	11
Isječak koda 4.2 Primjer zapisa greške u dnevniku SQL servera.....	11
Isječak koda 5.1 Primjer pravila značajnog događaja .....	14
Isječak koda 5.2 Primjer dnevnika komponente.....	14
Isječak koda 5.3 Algoritam značajnih vremenskih intervala nekonstantne vrijednosti .....	15
Isječak koda 5.4 Algoritam za značajne vremenske intervale konstante vrijednosti .....	16
Isječak koda 5.5 Algoritam za stvaranje korelacijskih grupa.....	17
Isječak koda 6.1 Format konfiguracijske datoteke .....	52
Isječak koda 6.2 Naredba za instalaciju servisa .....	52