SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

PassLock

**PROJEKT IZ KOLEGIJA „Sigurnost informacijskih sustava“**

Varaždin, 2018.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

**Fabijan Josip Kraljić**

**Jakov Kristović**

**GitHub repozitorij:** [**github.com/jkristovi/PassLock**](https://github.com/jkristovi/PassLock)

PassLock

**PROJEKT IZ KOLEGIJA „SIgurnost informacijskog sustava“**

Mentor:

Doc dr. sc. Petra Grd

Varaždin, prosinac 2018.

Sadržaj

[1. Ideja projekta 1](#_Toc534640339)

[2. Slične aplikacije 2](#_Toc534640340)

[2.1 LastPass 2](#_Toc534640341)

[2.2 KeePass 3](#_Toc534640342)

[2.3 StickyPassword 4](#_Toc534640343)

[3. Korištene tehnologije 5](#_Toc534640344)

[3.1 Visual Studio 2017 (C#, .NET Framework) 5](#_Toc534640345)

[3.2 SQLite 6](#_Toc534640346)

[3.3 SHA256 hash algoritam 7](#_Toc534640347)

[4. PassLock aplikacija 8](#_Toc534640348)

[5. Zaključak 16](#_Toc534640349)

[6. Literatura 17](#_Toc534640350)

# Opis aplikacijske domene

Tema aplikacije baza podataka za skladište nekog dućana odjeće je automatiziranje rada samog skladišta (naručivanje) te olakšati rad krajnjim zaposlenicima u danom skladištu. Bit aplikacije je unos cjelokupne robe koja se nalazi na skladištu nekog dućana i/ili robne kuće (tipa Zara-e ili H&M-a) po nekakvim kategorijama i/ili vrijednostima, evidentirati njezinu količinu na skladištu. Nadalje, u slučaju da količina zaliha neke stavke sa skladišta padne na određenu razinu tada baza podataka treba sama odrediti koliku količinu robe treba naručiti te na temelju toga sastavi narudžbenicu, bez intervencije krajnjeg korisnika. Takva vrsta naručivanja je izvedena na temelju minimalnih količina te kontinuiranog naručivanja. Osim toga, funkcija same baze je olakšanje pronalaska određene robe na skladištu kao i evidencija ulaznih i izlaznih stavki sa samog skladišta.

Domena pomoću koje je implementirana takva aplikacija te baza podataka obuhvaća Robu, Stanje skladišta, Narudžbenicu te Evidenciju skladišta te pripadajući elementi, a za realizaciju baze podataka korištene su tehnike aktivnih i temporalnih baza podataka koje će u nastavku biti objašnjene.

Razlog za takav rad je sama pomisao o tome kako uopće takvo velike robne kuće i dućani evidentiraju svoju robu, jer sigurno ne naručuju ručnu svoje stavke budući da je cjelokupan asortiman dosta ogroman te je teško pratiti što je na zalihama, a što ne. Zato sam pomislio da bi aktivne baze podataka bile dobre za automatsko naručivanje robe pri nekom stanju zaliha na skladištu, a temporalne za evidenciju narudžbi koje su obrađene, a koje ne.

# Teorijski uvod

U nastavku teksta biti će pobliže objašnjene aktivne baze podataka te temporalne baze podataka, budući da su one implementirane u samu strukturu baze podataka koja se koristi u krajnjoj aplikaciji.

* 1. Aktivne baze podataka

Aktivna baza podataka vrsta konvencionalne baze podataka koja sadrži mehanizme koji reagiraju na nekakve unutrašnje ili vanjske podražaje te se na temelju toga izvri neki niz akcija ovisno o tome što se želi postići.

Za to je potrebno znati pravila pod imenom ECA („Event – Condition – Action“) ili jednostavno „Događaj – Uvjet – Akcija“. Spomenuto možemo interpretirati na sljedeći način, kada dođe do nekog *događaja* (Event) i kada su ispunjeni neki *uvjeti*, onda se izvrši neka *akcija*, u prijevodu automatiziran rad baze dobijemo. (Kornelije Rabuzin, Mirko Maleković, Alen Lovrenčić)

Prednosti takvog pristupa, odnosno korištenjem takve tehnike je što korisnik ne treba voditi nikakvu brigu oko održavanja nekih dijelova baze podataka već ona sama vodu tu brigu. Nedostatak je što lako dođe do nekakvih logičkih pogrešaka koje je teško ispraviti te troši dodatne resurse.

* 1. Temporalne baze podataka

U odnosu na aktivne baze podataka, temporalne baze podataka uzimaju u obzir vremenski aspekt. Tako si možemo zamisliti da da temporalne baze podataka sadrže nekakve povijesne podatke i nekakve trenutne podatke, tj. nekakav interval. Takve baze sadrže nekakvo vrijedeće vrijeme, transakcijsko vrijeme i bitemporalno vrijeme. Vrijedeće vrijeme je vrijeme tijekom koje je neka činjenica istinita, transakcijsko vrijeme je vrijeme tijekom kojeg je neka činjenica pohranjena u bazi podataka, a bitemporalno vrijeme je kombinacija oba spomenuta vremena. Tako neki podaci imaju ugrađeni vremenski aspekt, npr. Narudžbenica je izdana na neki datum te se ona sada treba izvesti, kad se ona izvede imamo datum izvršenja Narudžbenice, tada je vrijeme izdavanja početno vrijeme, a vrijeme izvršenja završno vrijeme, odnosno Narudžbenica ima nekakav svoj vremenski aspekt koji nam je potreban.

# Model baze podataka

Na slici ispod (Slika 4.1) je prikazan model baze podataka koji je korišten prilikom izrade desktop aplikacije. Model baze podataka zovemo još i ERA model, gdje E označava entitet (tablicu), RA označava relaciju (vezu među entitetima). Prikazan model je izrađen u Navicat Premium alatu, grafičkom alatu za upravljanje bazama podataka. Alat omogućuje jednostavno kreiranje tablica i ovisnosti među njima putem grafičkog sučelja što je znatno lakše nego pisanje čistih SQL naredbi. Model baze podataka se sastoji od 11 tablica, tj. entiteta i 4 triggera, a u nastavku je sve pobliže objašnjeno.

# 

[1] Visual Studio 2017, Microsoft [[https://visualstudio.microsoft.com/]](http://complex.zesoi.fer.hr/index.php/en/%5d)

*dostupno 06.12.2018.*

[2] SQLite *[*https://www.sqlite.org/index.html*]*

*dostupno 06.12.2018.*

[3] Wikipedija, SHA-2 algoritam [https://en.wikipedia.org/wiki/SHA-2]

*dostupno 06.12.2018.*

[4] LastPass, *LogMeIn, Inc.* [<https://www.lastpass.com/>]

*dostupno 06.12.2018*

[5] KeePass, *Dominik Reichl* [<https://keepass.com/>]

*dostupno 06.12.2018*

[6] StickyPassword, *Lamantine Software* [<https://www.stickypassword.com/>]

*dostupno 06.12.2018*