VELEUČILIŠTE VERN'

Zagreb

Poslovna informatika

ZAVRŠNI RAD

Projekt izrade rješenja za praćenje rezultata sportskih natjecanja

Filip Ivandić

Zagreb, 2018.

VELEUČILIŠTE VERN'

Preddiplomski stručni studij
Poslovna informarmatika

ZAVRŠNI RAD

Projekt izrade rješenja za praćenje rezultata sportskih natjecanja

Mentor: Dražen Patekar

Student: Filip Ivandić

Zagreb, srpanj 2018.

VELEUČILIŠTE VERN
Zagreb, Trg bana Josipa Jelačića 3
Poslovna informatika

Broj: 3900

ZADATAK ZAVRŠNOG RADA

Kandidat: Filip Ivandić

Tema: Projekt izrade rješenja za praćenje rezultata sportskih natjecanja

U radu je potrebno razraditi sljedeće:

- Opisati procese i proolematiku tradicionalnog praćenja rezultata sportskih natjecanja.
- Definirati funkcionalne i nefunkcionalne zahtjeve.
- Opisati tehnologiju radiofrekvencijske identifikacije i njezinu primjenu.
- Projektirati bazu podataka sustava.
- Projektirati aplikativni sustav i opisati njegove dijelove.
- Opisati postupak implementacije sustava.
- Definirati zaključke i preporuke za praksu.

Napomena: Pri izradbi završnog rada kandidat ima obvezu pridržavati se i uvažavati primjedbe, sugestije i naputke mentora, koristiti i primjenjivati znanja i umijeća stečena tijekom studija, upotrebljavati informacije i podatke prikupljene vlastitim istraživanjem te spoznaje i činjenice iz odgovarajuće znanstvene i stručne literature.

Zadatak zadan: 30.06.2018.

Rok predaje: 04.07.2018.

Mentor	**Voditelj studija**
Dražen Patekar, predavač	mr.sc. Ivo Beroš, viši predavač

MP

PREDGOVOR

Zahvaljujem svojem mentoru Draženu Patekaru, koji je pratio cijeli proces i razvitak ovoga završnoga rada te me usmjeravao kako da riješimo sve probleme i nedaće, na njegovoj pomoći i savjetima.

Želim se zahvaliti i obitelji, djevojci i prijateljima na potpori kroz cijeli proces pisanja i programiranja.

SAŽETAK

U ovome radu opisan je detaljan postupak izrade moderne Windows aplikacije koja koristi tehnologiju RFID u razvojnoj okolini .Net Framework i bazu podataka Access za spremanje i korištenje podataka. Navedene tehnologije bit će korištene u vođenju rezultata u sportskim eventima, kao što su atletika, ali i u drugim sportovima. Glavni alat koji je ovo sve omogućio je Visual Studio Enterprise 2017 u kojem je cijela aplikacija napravljena. Razvojno okruženje koje je bilo bitno je .Net Framework koji nudi veliku stabilnost i puno mogućnosti, sve to uz veliku sigurnost.

Ova aplikacija je namijenjena ubrzanju te poboljšanju načina vođenja rezultata na sportskim događanjima, gdje bi se koristio RFID[1], koji radi na principu komunikacije i razmjene informacija između odašiljača podataka(RFID kartica) i prijemnika kao što su RFID čitači.

Ključne riječi: .NET Framwork, Access, RFID, Visual Studio Entreprise 2017

[1] Definicija se nalazi u poglavlju Razvojne okoline, podpoglavlje tri: RFID

ABSTRACT

This final paper describes a detailed process of making a modern Windows application that uses RFID technology in the .Net Framework developing environment and Access database to save and use data. These technologies will be used to track results in sport events, such as athletics, as well as in other sports. The main tool that has enabled this is Visual Studio Enterprise 2017 where the entire application is made. The developing environment that is essential is .Net Framework that offers stability with plenty of features with great security.

This application is intended to speed up and improve the way in which sports results are tracked where RFID is used, which operates on the principle of communicating and exchanging information between data transmitters (RFID cards) and receiver such as RFID readers.

Keywords: .NET Framwork, Access, RFID, Visual Studio Entreprise 2017

SADRŽAJ

1. UVOD

Glavna ideja ovog završnog rada je izrada moderne aplikacije uz korištenje drugih tehnologija kao što je RFID. Bit će opisani problemi izrade takve aplikacije

Tradicionalni oblik vođenja rezultata atletskih disciplina, kao što su kros i/ili maraton, je korištenje papira. Kros se vodi tako da se na komad drva zabija čavao. Trkači prilikom dolaska na završnu liniju zabijaju papir sa svojim startnim brojem. Nakon utrke, gdje se skupi do sto trkača, ta skupina papira se uzima i rangira tako da se prvi papir s brojem stavlja na prvo mjesto, pa drugi na drugo mjesto i tako dalje. Tu se stvara puno problema. Prvi je velika količina papira koja se printa i reže da bude određenih veličina, te rezultati nisu odmah vidljivi jer se nakon sortiranja moraju unijeti ručno u program ili online, što potraje ako je slaba veza ili nema interneta. Postoji i rizik od ozljede; Prilikom trčanja jedan od kandidata može se nabosti na čavao.

Aplikacija u ovome radu rješava sve te probleme, te povećava sigurnost i kvalitetu utrka. Rezultati se brzo mogu vidjeti, čak i na licu mjesta, zbog metoda i funkcija koje su se koristile, a RFID tehnologija omogućava brzinu i efikasnost. Zbog načina spajanja i komunikacije s računalom, nema potrebe za papirom. Aplikacija ne ovisi o posebnim programima jer je dovoljan samo PDF za objavljivanje podataka i pošto se lokalno sprema u Access, pristupačna je.

U prvom poglavlju bit će objašnjene razvojne okoline koje će se koristiti i funkcioniranje desktop aplikacije korištenjem RFID tehnologije. Zatim, u sljedećem će poglavlju biti spomenuti alati uz pomoć kojih će se izraditi desktop aplikacija i što se s njima može raditi. U zadnjem poglavlju, prije samog zaključka, bit će detaljno opisana arhitektura i izrada, s prikazanim izvornim kodom i komentarima kako kod radi.

2. RAZVOJNE OKOLINE I TEHNOLOGIJE

U ovom poglavlju bit će opisane razvojne okoline i tehnologija koje su iskorištene za izradu moderne aplikacije, te način na koji se koristio RFID i kako je implementiran. Objasnit će se i prednosti njihovog korištenja.

2.1 Microsoft .NET Framework

.NET Framework je okolina za razvoj i izgradnju različitih vrsta aplikacija za web, mobilne uređaje, stolna računala, Windows server. .NET Framework programi se izvršavu u softwareskom okruženju koje se zove Common Language Runtime (CLR), koje pokreće kod, upravlja izvršavanjem programa i nudi usluge koje olakšavaju razvojni proces. Sadrži ogromnu biblioteku koja je kolekcija klasa, sučelja i vrsta vrijednosti i naziva se Framework Class Library (FCL). FCL pruža jezičnu interoperabilnost[2]. Mogu se koristiti različiti programski jezici u .Net Frameworku uključujući i C#, F# i Visual Basic.

2.2 Microsoft Access

Microsoft Access je Microsoftov program koji služi za upravljanje bazama podataka. Glavne karakteristike kojima se Access služi su „Jet[3]“ i grafičko sučelje koje ima svoje alate za kreiranje i uređivanje, čak i alati za razvoj softwera.

Za rad koristi različite objekte kao što su tablice, upite, obrasci, stranice, izvještaji, *macros*[4] i moduli. Tablice služe za pohranu podataka. Pomoću upita, iz tablica se izvlače podaci. Upit je pitanje koje će iz jedne ili više tablica izvući one podatke koji zanimaju korisnika. Pomoću Accessa kreira se baza koja se iskoristila kao izvor podataka za ovaj rad.

[2] Language interoperability - *svaki jezik može koristiti kod koji je napisan na drugim jezicima*
[3] *Jet (Microsoft Jet Database Engine) je mehanizam za rad s bazama.*
[4] *Macros-objekt koji služi za automatiziranje radnji.*

2.3 RFID

RFID (Radio Frequency IDentification) odnosi se na tehnologiju kojom digitalni podaci, kodirani u RFID tagovima, ulaze u čitač putem radio valova. Znači, RFID koristi radijsku frekvenciju da bi jedinstveno identificirao neki objekt.

RFID sustavi se sastoje od tri komponente:

- RFID Tag (Transponder[5])
- RFID čitač
- antena

RFID tag sadrži integrirani krug i antenu, koji se koriste za prijenos podataka u RFID čitač. Čitač tada pretvara radiovalove u korisnije oblike podataka. Podaci prikupljeni iz taga zatim se prenose putem komunikacijskog sučelja na računalni sustav, gdje se podaci mogu pohraniti. Postoje dva tipa RFID taga - aktivni i pasivni. Aktivni tip ima svoj izvor napajanja, najčešće bateriju, dok pasivni ne zahtjeva bateriju kao izvor napajanja, RFID, već crpi elektromagnetsku energiju koju šalje RFID čitač.

Kada se jedan tag nalazi u blizini čitača koji preko antene proizvodi elektromagnetsko polje, tada njegov mikročip dobije energije, što mu omogućava bezkontaktno slanje podataka prema čitaču.

[5] Transponder *je uređaj za bežičnu komunikaciju, nadzor ili upravljanje, koji pokreće i automatski reagira na dolazni signal. Termin transponder je kombinacija riječi odašiljač i odzivnik (transmitter and responder) .*

Slika 2.1 Prikaz RFID čitača i tagova

Rad autora

Naravno, ovakva tehnologija ima i mana. Jedna od mana je i sami osnovni princip korištenja radiovalova, jer, ako bi se stavio neki materijal koji reflektira, kao aluminijska folija, cijeli bi sustav postao beskoristan. Problem stvara i vrijeme učitavanja. Ovisno o verziji čitača i kvaliteti, to vrijeme čitanja podataka se mijenja. I sama daljina stvara probleme. Jednostavni čitači učitavaju na par centimetara udaljenosti. Ako bi se željela veća udaljenost čitanja, treba se nabaviti antena ili pojačivač signala.

U radu će se koristiti RFID kartice. Domet čitanja RFID čitača je do deset centimetara zbog niske frekvencije(120–150 kHz[6]).

[6] kHz – *mjerna jedinica za frekvenciju*

3. RAZVOJNI ALATI

U ovome će poglavlju biti prikazani alati koji su bili iskorišteni za izradu aplikacije. Objasnit će se i prednosti korištenja i kako funkcioniraju.

3.1 Visual Studio

Visual Studio je takozvani integrated development environment (IDE) i pruža pomoć za pisanje koda i korištenje kontrola. Visual Studio je integrirano razvojno okruženje koje pripada Microsoftu. Koristi se za razvoj računarskih programa za Windowse, Linuxe, Androide i druge operativne sustave. Sadrži mnoge korisne stvari koje nude programerima ugodno iskustvo u programiranju i ubrzani razvoj programa. Ima puno grafičkih alata i kontrola koje zajedno čine korisničko sučelje.

Visual Studio podržava različite programerske jezike, a neki od njih, koji su već ugrađeni, su C++, C, C#, F# i Visual Basic. Naravno, podržava i druge programske jezike uz dodatnu instalaciju paketa. Pomoću tih instalacija, moguće je koristiti i druge jezike kao što su Python, R, Ruby... Podržava i alate za razvoj web stranice i uređivanje dokumenata kao što su XML[7], HTML[8], JavaScript [9] i CSS[10].

3.2 NuGet Package Manager

NuGet je alat za upravljanje paketa koji se preuzimaju i podešavaju komponente koje se stalno koriste. NuGet Package Manager služi se za skidanje dodatnih alata, ekstenzija koji bi određene zadatke olakšali. Tijekom izrade projekta javila se potreba za dodatnim alatima, ADGV. ADGV stoji za Advende DataGridView koji ima korisničko sučelje nalik Excelu sa opcija filtriranja i sortiranja.

[7] XML (EXtensible Markup Language) - *jezik za označavanje podataka*
[8] HTML *(HyperText Markup Language)* – prezentacijski jezik za izradu web stranica.
[9] JavaScript - skriptni programski jezik.
[10] CSS (Cascading Style Sheets) – stilski jezik za vizualni dizajn web stranice.

4. IZRADA APLIKACIJE

U ovom poglavlju bit će u koracima objašnjen način izrade aplikacije. Prvo će biti objašnjena konačna tražena arhitektura aplikacije. Nakon toga, redom, izrada baze podataka koja će služiti kao primjer, implementacija razvojnih okolina i testiranje aplikacije. U koracima će biti prikazani i bitni dijelovi koda uz sporedne forme.

4.1 Podatkovni modeli

„Model podataka je formalno određena struktura prikazivanja podataka, programski alata za grafičko oblikovanje baze podataka kojim se, neovisno o sklopovlju, drugim programima ili performansama sustava prikazuju svojstva podataka i njihova međusobna povezanost.“ (Kiš,2002). U ovome će dijelu biti prikazan dijagram i objašnjeno općenito kako je baza postavljena koji su entiteti i kako funkcionira.

4.1.1 Entiteti

Ovdje će biti detaljno prikazani svi entiteti i atributi. Entitet je neka posebnost, koncept ili objekt interesa. Može biti stvaran ili apstraktan događaj ili predmet u kojem se pamte podaci. Atributi su podatci koji su jedinstvena obilježja entiteta. Među entitetima koji imaju svoje atribute postoji veza koja čini odnos dva entiteta. Prikazat će se svi entiteti baze sa svojim atributima.

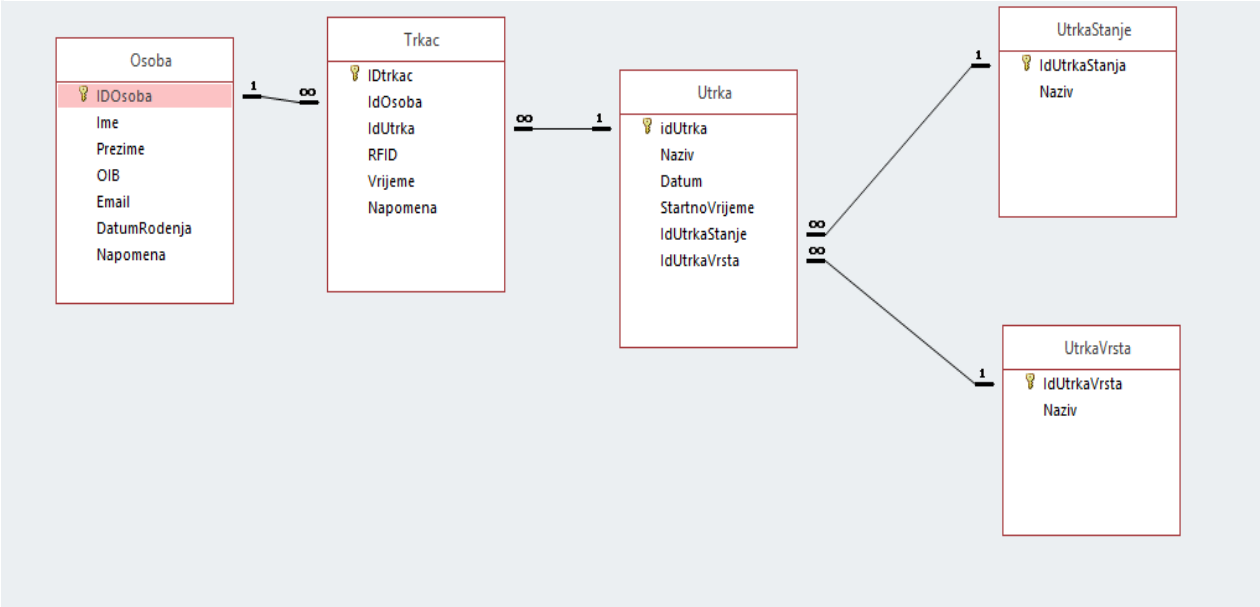
Osoba (idOsoba, Ime, Prezime, OIB, Email, Datumrođenja i Napomena) Trkac (IdTrkac, IdOsoba, IdUtrka, RFID, Vrijeme, Napomena).

Utrka (idUtrka, Naziv, Datum, Startno vrijeme, idUtrkaStanje, idUtrkaVrsta.) utrkaStanje(Naziv, IdUtrkaStanje) , UtrkaVrsta(IdUtrkaVrsta, Naziv) . Veza postoji između Osoba i Trkac, Trkac i Utrka, Utrka posebno sa UtrkaStanje i UtrkaVrsta.

4.1.2 Dijagram baze podataka

Kada želimo opisati neki sustav primjenjujemo grafički prikaz podataka, dijagram baze podataka. Dijagram prikazuje tok podataka, jer se pri korištenju običnog tekstualnog zapisa pojavljuje se niz nedostataka, a najveći problem je održavanje i prikazivanje podataka. S obzirom na te činjenice bolje je grafički prikazati sredstvo za modeliranje i prezentaciju procesa sustava. Prikazat će se dijagram baze podataka iz Accessa.

Slika 4.1 Dijagram toga podataka



Rad autora

Baza koja je korištena sastoji se od 5 tablica (tables) koje služe za spremanje podataka. Glavne tablice su Osoba, Trkač i Utrka, dok su pomoćne UtrkaVrsta i UtrkaStanje. Pomoćne tablice u sebi sadrže samo naziv i ne koriste se pisanje, nego samo za čitanje podataka.

4.1.3 Opis

Baza podataka koja je korištena sastoji se od pet tablica. U tablicu Osoba unose se osnovni podaci(atributi) kao što su ime, prezime, datum rođenja, email i postoji posebna rubrika napomena koja je zamišljena za upis posebnih informacija koje bi mogle utjecati na osobu. Sljedeća tablica je Trkač koja se spaja na osobu preko veze jedan na više gdje dobiva idOsobe koja se kasnije koristi da se toj osobi dodjeli RFID, a i kasnije završno vrijeme nakon utrke.

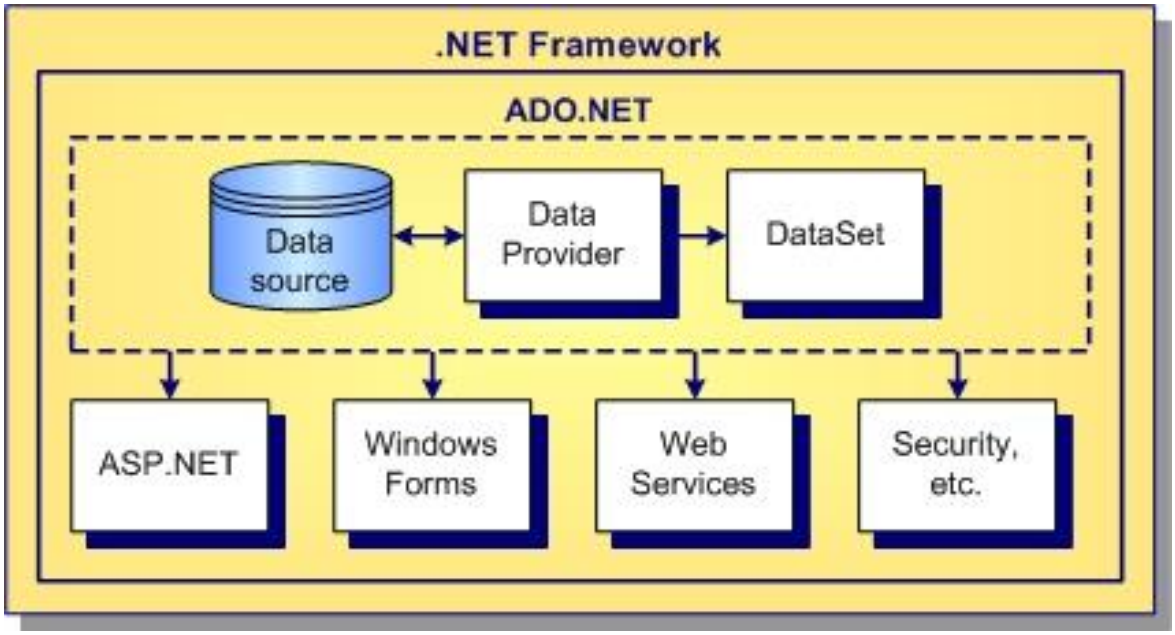
Tablica Utrka u sebi sadrži atribute kao što su vrijeme(misli se na startno vrijeme određene utrke) i datum utrke. Ona na sebe veže dodatne tablice UtrkaVrsta i UtrkaStanje.

4.1.4 Povezivanje na bazu podataka

Kako bi aplikacija imala podatke za raditi i koristiti, potreban je izvor podataka i mjesto za spremanje. Za to se koristila baza podataka Access. Baza je spojena na Visual Studio preko ADO.NET koji omogućava korištenje baze i njezinih podataka. ADO .NET je set klasa koji pruža pristup podacima za .NET Framework, najizravniju metodu pristupa podacima unutar .NET Frameworka. Nudi mnogo mogućnosti kao što su veze, čitanje i pisanje u i iz baze. Definira se i posebna klasa DataSet koja je kolekcija objekata DataTable. DataTable je unutarmemorijska kopija nekih tabličnih podataka. Naravno, može se koristiti zasebno ili za objekte. Stvara se pomoću pripadajućeg DataTable konstruktora[11]. Dok DataSet je klasa koja je praktičan način učitavanja mali podskup sadržaja baze podataka u klijentski kod. S tim se omogućava čitanje i pregled informacija iz baze podataka i uz to sadržava neke osnovne klijentsko obradu.

[11] *Konstruktor(constructors)*- omogućuju programeru postavljanje zadanih vrijednosti, ograničenje instanciranje i pisanje koda koji je fleksibilan i jednostavan za čitanje

Slika 4.2 Prikaz arhitekture ADO.Net



Izvor : Preuzeto 23.6.2018 sa stranice: http://docs.embarcadero.com/products/rad_studio/radstudio2007/RS2007_helpupdates/HUpdate3/EN/html/devnet/adonetov_xml.html

Prije spajanja izvor podataka mora biti jasno definiran. U projektu se koristila Access baza, ali mogao se koristiti i Microsoft SQL Server, Oracle. Kada se dodaje baza podataka, odlučuje se o tipu izvoru podataka.

U ovom slučaju, Završni.mdb je izvor podataka. Spaja se preko DataSeta, a pružatelj ovisi o kakvome je izvoru riječ.

Koristio se OLEDB (Object Linking And Embedding), API [12] koji je dizajniran da pruža pristup podacima iz raznih izvora. OLEDB API pruža niz sučelja implementiranih pomoću COM[13](Component Object Model).

[12] API (Application programming interface)- skup funkcija, procedura i metoda kojima se programeri mogu služiti za pristup uslugama ili resursima programa.
[13] COM - koristi se za stvaranje međuproduktnih komunikacijskih objekata u velikom rasponu programskih jezika.

Slika 4.3 Dio koda

```
<?xml version="1.0" encoding="utf-8"?>
<configuration>
    <configSections>
    </configSections>
    <connectionStrings>
        <add name="Zavrsni_rad.Properties.Settings.ZavrsniConnectionString"
connectionString="Provider=Microsoft.Jet.OLEDB.4.0;Data
Source=|DataDirectory|\Zavrsni.mdb" providerName="System.Data.OleDb" />
    </connectionStrings>
    <startup>
        <supportedRuntime version="v4.0" sku=".NETFramework,Version=v4.6.1" />
    </startup>
```

Rad autora

Nakon ustanovljene veze generira se kod unutar App.config koji sadrži informacije o samoj aplikaciji: tko je pružatelj usluge, vrsta veze, verziju .NET Frameworka.

4.2 Arhitektura aplikacije

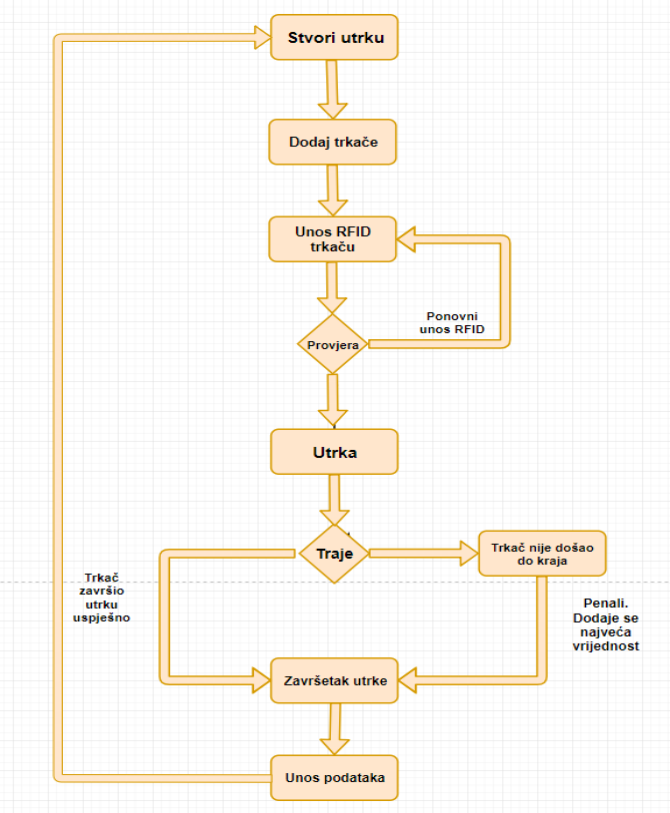
U ovome dijelu bit će riječi o arhitekturi aplikacije u kojoj se radila. U projektu se koristio Windows Forms (WinForms). WinForms je grafička (GUI[14]) biblioteka klasa koja je dio .NET FrameWork-a, ali više će biti rečeno u opisu arhitekture.

4.2.1 Konceptualno

Namjera završnog rada i same aplikacije je modernizacija vođenja rezultata sa RFID tehnologijom. Aplikacija je zamišljena tako da bi se registrirani trkači učitali u bazu i vodili. Svaki trkač bi dobio svoj startni broj i RFID karticu koja bi se koristila prilikom utrke. Prije utrke bi se stvorila utrka s potrebnim atributima. Utrka bi funkcionirala na principu unosa vremena koji bi bio registriran prilikom očitavanja RFID i tako bi se rangirali trkači.

[14] GUI(graphical user interface)- grafičko korisničko sučelje

Slika 4.4 Prikaz dijagram toka



Rad autora

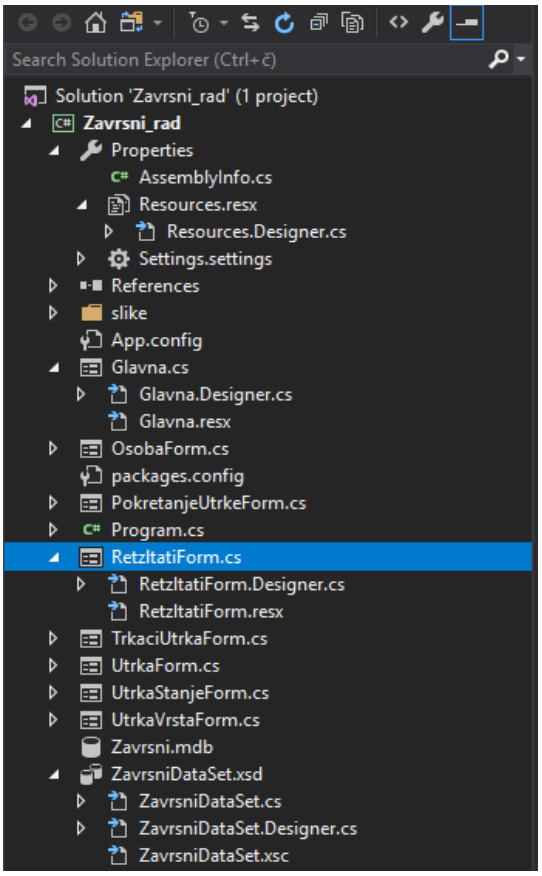
U dijagramu se detaljnije prikazuje odvijanje cijelog procesa, od stvaranje utrke do unosa podataka. Na kraju utrke, kada bi se kreirala nova utrka, proces bi se odvijao od početka.

4.2.2 Opis arhitekture

WinForms je grafička (GUI) biblioteka klasa koja je dio .NET FrameWork-a, jezgra same aplikacije jednostavne arhitekture bazirane na događajima(eventima), što znači da cijelo vrijeme aplikacija očekuje događaj klijenta koji je unaprijed predodređen. Ako se stvori jedna osnovna kontrola, kao što je gumb, može se dodati događaj koji ovisi o načinu korištenja. Tako se može napraviti da prilikom klika na gumb otvara novu formu, promjeni tekst, izbriše i mnogo drugih stvari. Kada se napravi, WinForms je prazan(blank) s par generiranih kodova da se može pokrenuti, a sve ostalo je na programeru da promisli o izgledu i funkcionalnosti.

Sama aplikacija strukturirana je pomoću formi koje u sebi sadrže svoju podklasu, izgled forme(Designer) i samu funkcionalnost forme.

Slika 4.5 Prikaz strukture Visual Studio



Rad autora

U glavnome direktoriju nalaze se Forme. U njemu je vizualno sučelje, gdje se prikazuju informacije bitne klijentu. Svaka forma sadrži kontrole, kako je već rečeno, koje ovise o događajima i ponašanju klijenta.

Uz pomoć Visual Studia, koji pruža IDE pri izradi koda, i .NET Frameworka i njegovih kontrola omogućena je izrada funkcionalnih i jako korisnih aplikacija.

4.3 Opis osnovnih dijelova i funkcionalnost aplikacije

Ovdje će biti objašnjeni osnovni dijelovi sustava i sama funkcionalnost aplikacije. Sam rad sa Visual Studiom jako je brz i efikasan u kombinaciji s WinForms i njegovim bibliotekama. Sadržava puno klasa i alata za izradu vrlo kvalitetnih aplikacija, te Access kao bazu podataka.

4.3.1 Aplikacija

Aplikacija je napravljena da sadrži forme. Forme su prilikom izrade prazne, pa se pomoću kontrola omogućava njihova funkcionalnost.

Svaka forma ima mogućnost korištenja već instaliranih klasa, metoda i kontrola . Da bi se više govorilo o aplikaciji, potrebno je proći neke kontrole koje su korištene u radu.

Tablica 4.1 Kontrole korištene u radu

Naziv kontrole	Funkcija
ToolStrip Control (alatna traka)	Kontejner u koju se mogu postaviti druge klase
Button Control	Jednostavna kontrola koja ovisi o događajima klijenta
TextBox Control	Kontrola koja se koriti za prikaz i unos podataka ili vrijednosti.
DateTimePicker Control	Kontrola kojom se klijentu omogućava odabir datuma i vremena
ComboBox Control	Kontrola koja nudi klijentu popis opcija
Label Control	Kontrola za ispis podataka ili vrijednosti
DataGridView Control	Kontrola za prikaz podataka u tabličnom obliku

Rad autora

U tablici su prikazane kontrole. Svaka kontrola ima svoju funkciju i namjenu. U TextBoxu se prikazuju podaci ili se traži od klijenata da upiše podatke. Gumbi se koriste, pošto je WinForms po arhitekturi ovisan o događajima, da se proces ili radnja odvija dok je gumb kliknut. Jedna od kontrola u aplikaciji je DataGridView. Ona se koristi za prikaz podataka u tabličnome obliku, a Label se koristi za ispis podataka. DateTimePicker se koristi za odabir vremena i datuma, ali ima svoje vlastito formatiranje, pa se može formatirati po potrebi.

Aplikacija se sastoji od osam formi. Izgled formi je jednostavan i sastoji se od dva zasebna kontejnera. Desni se sastoji od polja za upis podataka i alatne trake s osnovnim funkcijama kao što su: dodaj novu stavku, spremi promjene, uredi promjene, zatvori formu i posebnoga pregleda podataka koji se nalazi s lijeve strane. Pratio se jedan dizajn kroz cijelu aplikaciju.

Prva, ali i početna forma je GlavnaForm koja je odmah i glavni izbornik koji ima mogućnost otvaranja drugih forma, a to su Osoba i Utrka. U formi Osoba mogu se dodavati i uređivati osobe.

U formi Utrka dodaju se nove utrke i ona je izbornik za daljnje forme koje se koriste. Te forme su Trkači, Pokretanje utrke i Rezultati, koji služe za ispis podataka.

4.3.2 DataGridView

Obratit će se pažnja na jednu posebnu kontrolu i pomoću nje objasniti princip povezivanja podataka. Kontrola o kojoj je riječ je DataGridView koja se koristi za prikaz podataka.

Dizajnirana je kao fleksibilan, proširiv sustavi za prikaz i uređivanje tabličnih podataka. U radu se najviše koristila zbog toga svojstva. Kontrola podržava vezivanje podataka, tako da će se povezati s različitim izvorima. Najčešće će se povezati na BindingSource. BindingSource je komponenta koja prezentira izvor podataka. Kada je odabran, BindingSource Visual Studio generira TableAdaptor u poseban događaj Load koji se javlja kada se određena forma pokrene. TableAdaptor komponenta je koja puni DataSet sa podacima iz baze podataka, te je baziran na upitima.

TableAdaptor generira osnovni upit Fill,GetData(). Upit koji izabire(Select) podatke vezane za tablicu. Upit se generira čim se stvori. Svaki TableAdaptor ima svoj set Fill, GetData(). Ostali upiti koji se isto generiraju pri izradi veze su: ažuriraj(Update), obriši(Delete), ubaci(Insert) i već spomenuti izaberi(Select). Naravno, postoji mogućnost stvaranja vlastitih upita, ovisno o potrebi.

DataGridview ima i posebna svojstva. Nakon spajanja na izvor podataka, podaci se pokazuju u tablicama. Te tablice su zadane na početku kao kontrola za tekst(vidi tablicu 4.1), ali mogu se mijenjati u druge kontrole kao što su Button, Label i Combobox. Najzanimljivija je Combobox kontrola jer ima opciju povezivanja s drugim tablicama preko BindingSourca. To omogućava da se prikazuje umjesto nekoga određenog podatka u drugi pomoću DisplayMember svojstva za prikazivanje podataka.

4.3.3 Osnovne funkcije

Prikazat će se osnovne funkcije unutar aplikacije koje pridonose normalnome tijeku korištenja aplikacije. Imaju jednostavne radnje kao što su: otvori drugu formu, unesi novi podatak i spremi podatak. Sve funkcije rade s ugrađenim ADO.Net upitima. Određene funkcije ovise o upitima. Forme koje će se gledati su Osoba, UtrkaStanje i UtrkaVrsta, GlavnaForm.

GlavnaForm, koja služi kao izbornik aplikacije, ima dvije funkcije koje otvaraju druge forme. Slika prikazuje jednostavnu funkciju za otvranje forme Osoba, koja ovisi o događaju klik. Uz to, Glavna forma ima istu takvu funkciju za otvaranje forme Utrka.

Slika 4.6 Dio koda

```
private void btnOsoba_Click(object sender, EventArgs e)
        {
            var forma = new Osoba();
            forma.ShowDialog();

        }
```

Rad autora

Sljedeća forma s više funkcija je OsobaForma. U formi Osoba nalaze se funkcije za dodavanje novih podataka i spremanje tih podataka. Kod koji je prikazan služi za dodavanje osobe (Slika 4.7 Dio koda).

Funkcija za dodavanje osobe reagira na događaj klik i, prilikom toga, kontrole koje nisu dopuštene (txtIme.Enabled[15]=false;) omogućavaju da se upisuje u njih. Nakon toga dodaje se novi red(Row) unutar DataGridView, koji je na početku prazan i stavlja se fokus na TextBox kontrolu. Pomoću toga, otvara se prazan prostor za ispunu podacima.

Slika 4.7 Dio koda

```
private void tsbDodaj_Click(object sender, System.EventArgs e)
        {
            txtIme.Enabled = true;
            txtPrezime.Enabled = true;
            txtOIB.Enabled = true;
            txtxEmail.Enabled = true;
            rtbNapomena.Enabled = true;
            tsbUredi.Enabled = true;
            tsbSpremi.Enabled=true;
            dtmDatum.Enabled = true;

            var row = zavrsniDataSet.Osoba.NewOsobaRow();
            zavrsniDataSet.Osoba.AddOsobaRow(row);


            osobaBindingSource.MoveLast();

            txtIme.Focus();
         }
```

Rad autora

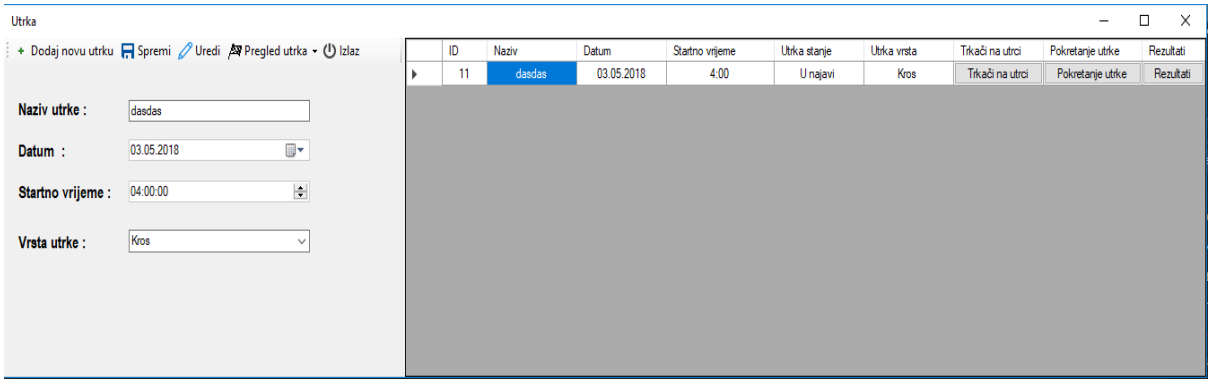
Druga funkcija koja se koristila je spremanje. Kod funkcionira tako da stopira trenutačne promjene, ubaci podatke u bazu i ažurira ju. Unutar tih formi uvela se i mogućnost korištenja tipkovnice i prečaca. Kada se klikne Ctrl+N dodaje se novo prazno polje. Ctrl+S sprema promjene, a Alt+F4 zatvara trenutačno otvoren prozor. Svi ti prečaci omogućeni su i na drugim formama.

[15] Enabled- Svojstvo kontrole da se reagira na korisnikovo interakciju

4.3.4 Funkcije UtrkaForm

U ovome dijelu rada bit će prikaza UtrkaForm forma. Njezina uloga je upis novih utrka, a DataGridView služi za prikaz informacija i prikaz ugrađenog izbornika s kojim se mogu otvarati forme Trkač, PokrenutaUtrka i Rezultati. Sljedeća slika (4.8) prikazuje formu UtrkaForm.

Slika 4.8 Dio koda



Rad autora

Utrka forma sastoji se od osnovnih funkcija za dodavanje i spremanje utrka. Funkcija koja otvara druge forme je dio DataGridViewa. Napravilo se tako da su određene ćelije formatirane kao gumb. Pritiskom na ćeliju otvara određenu formu. Sljedeći kod prikazat će kako je to napravljeno.

Slika 4.9 Dio koda

```
private void dgvUtrka_CellClick(object sender, DataGridViewCellEventArgs e)
        { if (e.ColumnIndex == 6)
            {
                TrkaciUtrkaForm forma = new TrkaciUtrkaForm();
                forma.IdUtrkas =
Convert.ToInt32(dgvUtrka.CurrentRow.Cells[0].Value.ToString());
                forma.lblId.Text = dgvUtrka.CurrentRow.Cells[1].Value.ToString();

                DateTime shp =
DateTime.Parse(dgvUtrka.CurrentRow.Cells[3].Value.ToString().ToString());
                forma.lblSvTr.Text = shp.ToString("t");

                DateTime shp2 =
DateTime.Parse(dgvUtrka.CurrentRow.Cells[2].Value.ToString().ToString());
                forma.lblDTrk.Text = shp2.ToString("dd.MM.yyyy");
                forma.Show(); }
        }
```

Rad autora

Prilikom klika na određenu ćeliju (u ovome primjeru ćelija 6) otvara se forma TrkaciUtrkaForm. Prilikom otvaranja određeni podaci prelaze s UtrkaForm na

TrkacForm. Kako bi se to omogućilo, u formi Trkač stvoreno je posebno svojstvo(property).

Slika 4.10 Dio koda

```
public int IdUtrkas
        {
            get { return _idUtrka; }
            set { _idUtrka = value; }
        }
```

Rad autora

Stvorivši to svojstvo, koje služi za prebacivanje idUtrka u druge forme i čita unutar nje, unutar UtrkaForm ono se dodijeli u kod. Podaci kao datum i startno vrijeme prebacuju se u Label u željenom formatu.

4.3.5 Funkcije TrkaciUtrkaForm

TrkaciUtrkaForm forma je u kojoj se registriraju trkači. Osim funkcija za upis, dodavanje i spremanje trkača, nudi i dodatne funkcije. Za vrijeme upisa podataka, pomoću RFID kartice, upisuje se RFID u TextBox koji je namijenjen tome. Kako je spomenuto, u prošloj formi je uvedeno svojstvo za prijenos podataka. U ovoj formi, TrkaciUtrkaForm, to se svojstvo koristi na više načina. Prvi je da se prikazuju samo trenutačna utrka koja se sprema u Label.

Druga funkcija toga svojstva je da se uz trkača automatski upisuje utrka kako bi se lakše sortiralo. Zadnja je funkcija da se prikazuju samo trkači koji su na toj utrci. Kako bi se to uspjelo napraviti, bilo je potrebna napraviti posebni upit. Upiti se rade pomoću ugrađenih alata za kreiranje upita.

Slika 4.11 Dio koda

```
SELECT     IDtrkac, IdOsoba, IdUtrka, RFID, Vrijeme, Napomena
FROM       Trkac
WHERE      (IdUtrka = ?)
```

Rad autora

Upit ima naziv FillByIDUtrka s parametrima IdUtrka koji prikazuje sve podatke iz tablice Trkač, gdje je traženi parametar idUtrka koji se dobiva pomoću svojstva iz forme Utrka.

4.3.6 Funkcije PokretanjeUtrkeForm

U ovoj formi PokretanjeUtrkeForm prati se i upisuje vrijeme. Za prikaz vremena trebalo se upotrebiti:

- System.Diagnostics namespeca[16] - Stopwatch. Služi za mjerenje vremena unutar aplikacije.
- Timer – komponenta koja podiže događaj u definiranim intervalima.
- Osnovna komponenta za početak i kraj vremena
- Prikaz vremena u definiranom formatu

Prije ugradnje funkcija, trebao se oblikovati sam prikaz vremena. Vrijeme se prikazuje u obliku stringa. Funkcija koristi TimeSpan [17]svojstva. Vrijeme se pokazuje u obliku minuta, sekundi, milisekundi.

Kada se ta funkcija napravila, uveo se Label u kojem se prikazuje prilagođeno vrijeme. Osim dodavanje naljepnice, na formu su se stavila tri gumba: gumb start, gumb pauza i gumb reset. Gumb start prilikom klika koristi metodu Start() koja upali štopericu da mjeri vrijeme, a timer da podiže vrijeme. Gumb pauza koristi metodu Stop() da zaustavi trenutačno vrijeme, a preostali gumb koristi metodu Reset() da postavi vrijeme na početnu vrijednosti. Sve metode pripadaju namespacu System.Dignostics.

Vrijeme se upisuje u DataSet kada određena RFID kartica bude registrirana od strane čitača i učitava se preko Textbox kontrole. Kako bi se vrijeme upisalo, koristio se posebni upit.

[16] System.Diagnostics namespeca omogućuje interakciju s procesima sustava, zapisima događaja i brojačima performansi.
[17] TimeSpan - predstavlja vremenski interval.

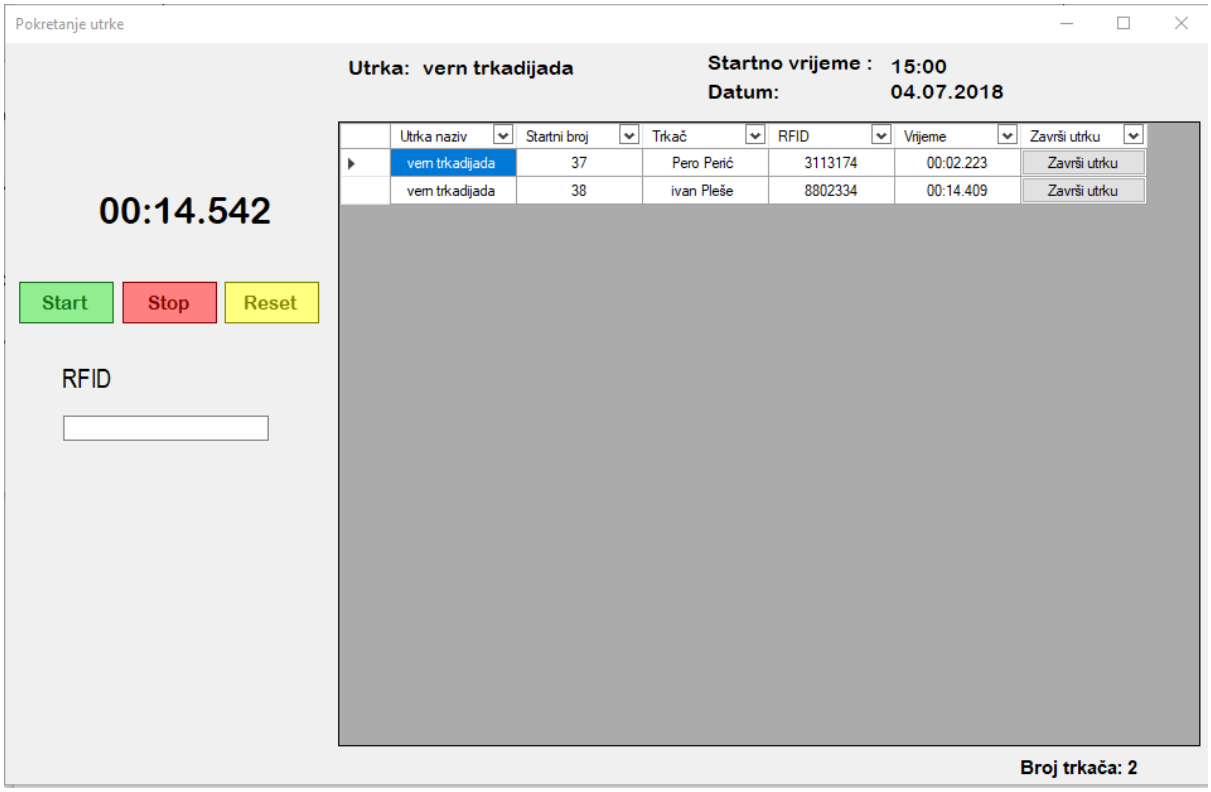
Slika 4.12 Dio koda

```
UPDATE      Trkac
SET          Vrijeme = ?
WHERE       (RFID = ?)
```

Rad autora

Upit ažurira tablicu Trkač i postavlja vrijeme na trenutno vrijeme, koje čita iz naljepnice, koja sadrži vrijeme. Upisuje vrijeme po traženom RFIDu.

Slika 4.13 Prikaz forme



Rad autora

Aplikacija radi tako da se na traženi RFID upiše vrijeme. Kada se upišu sva vremena, štoperica se zaustavlja i gumbi se više ne mogu koristiti. Automatski zaustavlja vremena koristeći upit.

Slika 4.14 Dio koda

```
SELECT      COUNT(*) AS Broj
FROM        Trkac
WHERE       (Vrijeme IS NULL) AND (IdUtrka = ?)
```

Rad autora

Upit traži ukupan broj trkača(Count) iz tablice Trkač, gdje vrijeme nije postavljeno; null. Upit uvijek vraća jedan broj. Broj koji upit vraća iskoristio se za automatsko zaustavljanje aplikacije. Napravila se globalna varijabla naziva count. Kada se broj iz upita izjednači s varijablom count, vrijeme se pauzira. Cijela ta funkcija odvija se dok ima trkača. Učitavanje vremena uspostavljeno je kada se upiše broj i javi enter. Dok se RFID učitava, stavlja se enter na kraj, što je omogućilo da se funkcija vrti u krug.

Slika 4.15 Dio koda

```
txtRFID.KeyPress += (sndr, ev) =>
            {
                if (ev.KeyChar.Equals((char)13))
                {
                    try
                    {
                        int rFID = int.Parse(txtRFID.Text);
                        string lol = lblTimer.Text;
                        trkacTableAdapter.UpdateQuery(lol, rFID);
                        count = count - 1;
                        trkacTableAdapter.Fill(zavrsniDataSet.Trkac);
                               ev.Handled = true;
                    catch
                    {

                    }
                    finally
                    {
                        txtRFID.Clear();
                        txtRFID.Focus();
                    }
                    if (count == 0)
                    {
                        watch.Stop();
                        btStop.Enabled = false;
                        btnStart.Enabled = false;
                        btnReset.Enabled = false;
```

Rad autora

Za svaki slučaj, uvela se i opcija da se zaustavi utrka ako netko od trkača odustane ili ne može završiti utrku. Kada trkač odustane, u ćeliju se upiše vrijednost Nan(not a number) i tako se prikazuje da je odustao ili nije mogao završiti utrku.

4.3.8 Funkcije Rezultatiform

Zadnja forma koja je ostala je prikazivanje rezultata. Nakon što je utrka odrađena, ukupni se podaci moraju prikazati i ispisati. Za tu funkcionalnost koristio se Microsoft Rdlc Report Designer. To je paket koji sadrži alate za kreiranje vlastitih izvještaja.

Prvo se stvara nova stavka koja se sprema u .rdlc formatu. Kontrole koje se mogu koristit su tablice, matrice, grafikoni, textbox. Nakon stvorene stavke, mora se, koristeći DataSet, stvoriti veza. Kada je veza stvorena, odabire se koji se podaci prikazuju i u kojem obliku. Podaci koji su se koristili su: ime i prezime iz tablice Osoba, vrijeme iz tablice Trkac. Uveo se i poseban stup Poredak koji u sebi sadrži rangiranje trkača.

U formu se dodaje kontrola koja služi za prikaz bilo kojeg izvještaja, ReportViewer koji se koristi za integriranje i prikaz izvještaja u formatu rdlc. To je mode koji sadržava sve procese i zbirku podataka iz .rdlc stavke. On sadrži biblioteke za prikaz.

Prilikom pokretanja forme Rezultat, ReportViewer prikazuje odabrani izvještaj. Izvještaji u sebi imaju ugrađene funkcije: Izvoz podataka, uvećavanje ili umanjivanje fonta, tražilicu sadržaja.

To su sve funkcionalnosti aplikacije, od jednostavnih do složenijih. Uvijek ima mjesta za napredak i razvoj.

4.4 Implementacija i korištenje

U ovome dijelu rada objasnit će se kako bi se aplikacija implementirala u stvarnim događajima i utrkama. Program je napravljen u Visual Studiju i koristi resurse, procese i servise s Windows operativnoga sustava za koji je namijenjen. Program bi se koristio za sportske evente, točnije, namijenjen je za atletske discipline.

Program se koristi tako da na početnome izborniku bira opciju Osoba koja dovodi klijenta na formu gdje se unose nove osobe.

Pritiskom na gumb „Dodaj novi“ stvara se nova prazna ćelija u kojoj se unose podaci. Podaci se upisuju u kontrole TextBox pazeći da podaci budu važeći i u pravilnome obliku (OIB mora imati jedanaest znamenki i samo u brojčanome zapisu, moraju se unijeti sve tražene informacije). Kada se podaci unesu, pritišće se gumb „Spremi“ gdje se unešene informacije spremaju. Ako je klijent napravio pogrešku, postoji mogućnost „Uredi“ kojom se mijenjaju unešene informacije.

Kada je klijent upisao sve željene osobe, forma se gasi i odlazi na drugu formu Utrka u kojoj se stvara nova utrka. Nova utrka mora sadržavati ime, dan utrke, startno vrijeme i vrstu atletske discipline. Nakon spremanja, utrka se prikazuje unutar kontrole DataGridView. U istome retku DataGridViewa postoje tri gumba koji vode na sljedeće forme: Trkači na utrci, Pokretanje utrke i Rezultati.

Klijent prvo mora pritisnuti gumb Trkači na utrci kako bi registrirao trkače na određenu utrku. Odabire se trkač i, pomoću RFID čitača, unosi se njegov posebni RFID kod. U slučaju da je potrebno, može se dodati neka napomena. Nakon što je trkač upisan i spremljen, dodjeljuje mu se startni broj koji služi za provjeru trkača.

Nakon što su unešeni svi trkači, potrebno je promijeniti u formu Pokretanje utrke. Tu se odrađuje upis vremena i spremanje u DataSet. Kada utrka počne, vrijeme se pali i pojavljuje u Labelu.

Kada trkač dolazi prema završnoj liniji mora prisloniti RFID karticu na čitač da bi se upisao. Samo učitavanje i spremanje broja vremenski potroši do jedne sekunde, što se da popraviti boljim čitačem. Nakon što se kartica prislonila i upisala, trkač je završio utrku i sortira se po svojemu vremenu. Nakon što su svi trkači završili utrku, timer se automatski zaustavlja. Ako bi jedan ili više trkača odustalo ili ne mogu završiti utrku,

postoji i ručno gašenje. Završena utrka nema više mogućnost biti promijenjena, mogu se samo pročitati podaci.

Zadnji gumb odvodi klijenta u formu gdje se samo čitaju i ispisuju podaci. Prvo se prezentiraju svi rezultati u tablici, sortirani po vremenu, od najmanjeg vremena prema najvećem, i po tome su rangirani. Ako su podaci valjani, cijela se tablica ispisuje. Mogućnost ispisa je u Wordu, Excelu ili PDFu. Tako bi se koristila i implementirala aplikacija u stvarnome svijetu.

5. ZAKLJUČAK

Tema ovoga rada bila izrada rješenja za praćenje rezultata sportskih natjecanja, što je i realizirano.

Postigao se glavni cilj, a to je kako iskoristiti RFID u modernizaciji bilježenja rezultata. Do toga se došlo uz korištenje razvojnoga okruženja Visual Studio i .NET Framework, koji su se pokazali izvrsni u rješavanju problema i ostvarenju aplikacije zbog mnogih alata za bolji pregled i pisanje koda.

Pomoću ove aplikacije, mnoga sportska natjecanja mogla bi se voditi puno kvalitetnije i jednostavnije zbog RFID rješenja, koji bi uz pomoć čitača i tagova jako ubrzao proces; Od dodavanja trkača na utrku do samoga bilježenja vremena.

Ovaj rad je rezultirao uspješnom izradom rješenja za praćenje rezultata sportskih natjecanja, ali može poslužiti i kao osnova za daljnji rad na poboljšanju ukupne modernizacije sportskih natjecanja, koja bi se mogla postići uvođenjem drugih tehnologija.

LITERATURA

Knjige:

1. I.Griffitbs, M.Adams i J.Liberty (2010) Programiranje C# 4.0. Zagreb. Dobar plan.

2. M.Pavlić (2010) Informacijski sustavi. Zagreb. Školska knjiga, d.d

Internetski izvori:

1. What is .NET? . Preuzeto 19.6.2018 sa stranice:

https://www.microsoft.com/net/learn/what-is-dotnet

2. Overview of the .NET Framework. Preuzeto 19.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/framework/get-started/overview

3. Windows Forms. Preuzeto 20.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/framework/winforms/

4. Create .NET apps faster with NuGet. Preuzeto 20.6.2018 sa stranice:

https://www.nuget.org

5. Stephen A. Weis (2012) RFID (Radio Frequency Identification): Principles and Applications. Preuzeto 20.6.2018 sa stranice:

http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.182.5224&rep=rep1&type=pdf

6. Visual Studio Enterprise. . Preuzeto 20.6.2018 sa stranice:

https://visualstudio.microsoft.com/vs/enterprise/#Fragment_SystemRequirements

7. Creating a DataTable. Preuzeto 20.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/framework/data/adonet/dataset-datatable-dataview/creating-a-datatable

8. BindingSource Component Overview. Preuzeto 21.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/framework/winforms/controls/bindingsource-component-overview

9. Introduction to LINQ Queries (C#). Preuzeto 21.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/csharp/programming-guide/concepts/linq/introduction-to-linq-queries

. Learn Nuget Package Manager in 10 Minutes. Preuzeto 21.6.2018 sa stranice:

http://www.codedigest.com/quick-start/8/learn-nuget-package-manager-in-10-minutes

11. Timer Class. Preuzeto 22.6.2018 sa stranice:

https://docs.microsoft.com/en-us/dotnet/api/system.windows.forms.timer?view=netframework-4.7.2

12. RFID tagging. Preuzeto 27.6.2018 sa stranice:

https://internetofthingsagenda.techtarget.com/definition/RFID-tagging

POPIS SLIKA

POPIS TABLICA