



Gibanje MLADI RAZISKOVALCI KOROŠKE

**Področje: Aplikativni inovacijski predlogi in projekti**

## **REZULTATI ŠPORTNIH DNI – SPLETNA APLIKACIJA**

**Avtorji: Amadej Glasenčnik, Jovana Simeunović,  
Matic Jurač**

**Mentorici: Mateja Aplinc, prof.,  
Janja Kogelnik, prof.**

2023, Ravne na Koroškem

Srednja šola Ravne, Na gradu 4a, 2390 Ravne na Koroškem

# KAZALO

1 UVOD .....	1
2 ZASNOVA IDEJE O REŠITVI PROBLEMA .....	2
3 PRAKTIČNI OPIS SPLETNE REŠITVE .....	3
3.1 GLAVA SPLETNE APLIKACIJE NEPRIJAVLJENIM UPORABNIKOM.....	3
3.2 GLAVA SPLETNE APLIKACIJE PRIJAVLJENIM UPORABNIKOM .....	4
3.3 NOGA SPLETNE APLIKACIJE.....	4
3.4 DOMAČA STRAN – UVODNO BESEDILO .....	5
3.5 DOMAČA STRAN – REZULTATI TEKMOVALNIH SKUPIN.....	5
3.6 STRAN S SEZNAMOM NALOG .....	6
3.7 STRAN S PODROBNOSTMI NALOG .....	6
3.8 STRAN Z VSEMI REZULTATI NALOG .....	7
3.9 PRIJAVNA STRAN .....	7
3.10 NADZORNA PLOŠČA.....	8
3.11 STRAN ZA VNOS REZULTATOV .....	8
3.12 STRAN Z MENIJEM ZA UREJANJE PODATKOV.....	9
3.13 STRAN ZA UREJANJE UVODNEGA BESEDILA DOMAČE STRANI .....	9
3.14 STRAN ZA UREJANJE RAZREDOV .....	10
3.15 STRAN ZA UREJANJE UPORABNIKOV .....	10
3.16 STRAN ZA UREJANJE NALOG .....	11
3.17 STRAN ZA UREJANJE REZULTATOV NALOG .....	11
4 UPORABLJENA TEHNOLOGIJA.....	12
4.1 HTML.....	12
4.2 CSS .....	13
4.2.1 Bootstrap 5.....	13
4.3 JAVASCRIPT .....	14
4.4 C# .....	14

4.5 .NET .....	15
4.6 BLAZOR SERVER .....	15
4.7 ENTITY FRAMEWORK CORE oz. EF CORE .....	16
4.8 SYNCFUSION .....	16
4.9 SQL SERVER.....	17
4.10 VISUAL STUDIO 2022 .....	17
5 IZDELAVA SPLETNE APLIKACIJE .....	18
5.1 USTVARJANJE PODATKOVNE BAZE Z EF CORE .....	18
5.1.1 Ustvarjanje tabele AppData .....	19
5.1.2 Ustvarjanje tabele Class .....	19
5.1.3 Ustvarjanje tabele Exercise .....	19
5.1.4 Ustvarjanje tabele User .....	20
5.1.5 Ustvarjanje tabele UserExercise.....	20
5.1.6 Ustvarjanje tabele GameResult.....	21
5.1.7 Ustvarjanje razreda ApplicationDbContext .....	21
5.2 RAZRED ZA PRIJAVO .....	22
5.3 STRAN ZA VNOS REZULTATOV .....	23
5.4 PRIMER STRANI ZA UREJANJE PODATKOV / UPORABA SYNCFUSION DATAGRID .....	24
6 ZAKLJUČEK.....	26
7 SEZNAM LITERATURE .....	27
8 PRILOGE.....	30

## KAZALO SLIK

Slika 1: Domača stran spletne aplikacije.....	3
Slika 2: Glava spletne aplikacije strani neprijavljenim uporabnikom .....	3
Slika 3: Glava spletne aplikacije prijavljenim uporabnikom .....	4
Slika 4: Noga spletne aplikacije .....	4
Slika 5: Uvodno besedilo domače strani .....	5
Slika 6: Doseženi rezultati po posameznih nalogah .....	5
Slika 7: Seznam nalog .....	6
Slika 8: Podrobnosti naloge .....	6
Slika 9: Prijavna stran.....	7
Slika 10: Nadzorna plošča .....	8
Slika 11: Stran za vnos rezultatov .....	8
Slika 12: Stran z menijem za urejanje podatkov.....	9
Slika 13: Stran za urejanje uvodnega besedila domače strani .....	9
Slika 14: Stran za urejanje razredov/tekmovalnih skupin.....	10
Slika 15: Stran za urejanje uporabnikov .....	10
Slika 16: Stran za urejanje nalog.....	11
Slika 17: Stran za urejanje rezultatov nalog.....	11
Slika 18: Najpogosteje uporabljen logotip HTML5 [2] .....	12
Slika 19: Najpogosteje uporabljen logotip CSS [4].....	13
Slika 20: Najpogosteje uporabljen logotip C# [8] .....	14
Slika 21: Uradni logotip .NET [10] .....	15
Slika 22: Uradni logotip Blazor [12] .....	15
Slika 23: Uradni logotip Syncfusion [15].....	16
Slika 24: Uradni logotip SQL Server [17] .....	17
Slika 25: Uradni logotip Visual Studio 2022 [18].....	17

## **POVZETEK**

Za raziskovalno nalogo MRK smo se odločili, da bomo ustvarili spletno aplikacijo za vodenje športnih dni in prikaz njihovih rezultatov. V to spletno aplikacijo bo skrbnik lahko vpisoval in spreminjal podatke za ekipe, točke in različne dejavnosti športnega dneva. Dodeljeni sodniki bodo lahko vpisovali rezultate za posamezno nalogo določenemu razredu. Za izdelavo naloge smo uporabili najnovejša Microsoftova orodja za razvoj programske opreme (Blazor Server, Visual Studio 2022, Entity Framework Core, programski jezik C#, ogrodje .NET in ASP.NET Core). Uporabili smo tudi opremo za izdelavo spletnih tehnologij (HTML, CSS, Bootstrap in JavaScript). Pri delu smo naleteli na številne težave. Glavno težavo nam je predstavljalo, kako načrtovati, sprogramirati ter preizkusiti program, da bo deloval brez kakršnekoli skrite napake.

Spletno stran smo preizkusili na športnem dnevu Srednje Šole Ravne (na spletni povezavi: <https://sportni-dan.aglasencnik.com>), kjer se je spletna aplikacija odnesla več kot odlično.

Ključne besede: organizacija športnih dni, vpis rezultatov, C#, Blazor Server

## SUMMARY

For our MRK research paper, we have decided to create a web application for hosting sports events and to show the results. In this web application the administrator will be able to upload and edit the points, team names and activities. Chosen judges will also be able to upload results for each specific activity for each individual team. To create the web application we used the most modern tools provided by Microsoft for software development (Blazor Server, Visual Studio 2022, Entity Framework Core, the programming language C#, .NET and ASP.NET). We also used various web design technologies (HTML, CSS, Bootstrap and JavaScript). During the process we encountered many issues. The main challenge was how to plan, program and test the application, so that it would work without any hidden mistakes.

The website was already tested on the sports event for the Secondary School Ravne (<https://sportni-dan.aglasencnik.com>), where the web app performed more than excellently.

Key words: sports event organisation, uploading results, C#, Blazor Server

## 1 UVOD

V današnjem času živimo s tehnologijo z roko v roki. Za skoraj vsako opravilo imamo svojo aplikacijo, ki nam olajša delo. Ker pa smo vpisani v program tehnik računalništva in ker živimo v digitalnem času, je bila za nas organizacija športnih dni težje sprejemljiva.

Športni dnevi na srednjih šolah potekajo brez uporabe informacijske tehnologije. Prav tako je bilo tudi na naši Srednji šoli Ravne. Z organizacijo atletskega športnega dne je bilo veliko dela. Razredi so med seboj tekmovali v različnih igrah na petindvajsetih postajah. Sodniki so morali podatke zapisovati ročno na za to pripravljene liste papirja. Profesor, organizator športnega dne, je moral počakati do zaključka le-tega. Nato je od sodnikov pridobil liste s podatki, podatke ročno vnesel ter jih obdelal v wordovih preglednicah. Ta pristop ima ogromne pomanjkljivosti, kot so: veliko dela, ogromna količina papirja in zabeleženih podatkov, možne napake pri izračunih in razvrščanju mest, možna izguba podatkov, dolgotrajen proces organizacije ...

Zato smo se vprašali: Kako bi ustvarili inovativno spletno aplikacijo z uporabo IKT-tehnologije in s tem uporabo mobilnih naprav? Bila bi ekološko sprejemljivejša, organizatorju (skrbniku/administratorju) bi omogočala hitrejše zbiranje, urejanje in evidentiranje podatkov. Udeležencem bi prikazovala rezultate z uvrstitvami. Hkrati pa bi imeli večletni statistični pregled o športnih dneh v različnih oblikah (grafikoni, tabele, diagrami ...).

V raziskovalni nalogi so zastavljene naslednje hipoteze:

1. hipoteza: Spletna rešitev bo inovativna oziroma še ne obstaja.
2. hipoteza: Spletna aplikacija je ekološko sprejemljiva.
3. hipoteza: Spletna aplikacija omogoča ažuren vnos, urejanje in ogled podatkov.
4. hipoteza: Spletna rešitev zagotavlja statistični pregled podatkov za prejšnja leta.

## **2 ZASNOVA IDEJE O REŠITVI PROBLEMA**

Še preden smo lahko začeli z izdelovanjem naše spletne aplikacije, smo morali preučiti, katere zahteve naj sploh naša spletna aplikacija izpolni.

Kasneje pa bomo vsako od teh strani prikazali ter opisali njeno nalogo.

Ugotovili smo, da mora spletna aplikacije na začetku vsebovati domačo stran, kjer bo lahko zunanja oseba ali tudi kdorkoli drug pogledal skupne rezultate tekmovanj. Naslednja stran bi vsebovala seznam nalog, kjer bi ob kliku na gumb uporabnika pripeljalo do samih podrobnosti naloge, za katero se je odločil, da jo bo podrobno pogledal. Zadnja stran, ki bi jo lahko zunanji uporabniki dostopali, bi bila uporabljena za prikaz vseh rezultatov posameznih nalog.

Za administratorja in sodnike pa bi bilo na voljo tudi več njim namenjenih strani, katere, pa bi lahko dostopali ob prijavi. Naredili smo gumb, kjer se ob kliku pojavi nova stran, kjer se uporabnik prijavi z uporabniškim imenom in geslom, ki ga je prejel od skrbnika oz. administratorja.

Ob prijavi bi se uporabnikom prikazala nova stran, ki pa bi bila različna glede na to, ali je oseba administrator ali pa samo sodnik.

Sodnik bi v tem oknu lahko videl dva gumba. Pri kliku na prvega bi ga aplikacija peljala do naloge, katero ocenjuje, da si lahko tam ogleda podrobnosti (navodila, rezultate). Klik na drugi gumb pa bi ga popeljal do strani za vnos rezultatov, kjer pa bi sodnik nato moral vnesti rezultate glede na svoja navodila naloge, spletna aplikacija bi pa te rezultate obdelala ter prikazala obiskovalcem.

Administrator bi videl tudi tretji gumb, ki pa bi ga popeljal do novih strani, ki bi bile namenjene urejanju podatkov spletne aplikacije. V tem razdelku bi lahko administrator urejal podatke domače strani, lahko bi dodajal, urejal ali brisal razrede oz. tekmovalne ekipe, sodnike, naloge ter vpisane rezultate nalog. V naslednjem poglavju si bomo ogledali tehnologije, katere so bile uporabljene za izdelavo aplikacije.



## 3 PRAKTIČNI OPIS SPLETNE REŠITVE

Za začetek bomo prikazali in opisali vse strani oz. komponente spletne aplikacije, ki so bile narejene v procesu izdelave spletne aplikacije. Domača stran je prikazana spodaj na sliki 1.

ŠPORTNI DAN 2022 OKTOBER

To je spletna stran za športni dan 2022, ki ga prireja Srednja Šola Ravne na Koroškem.

Športni dan se bo odvijal v sredo, 5. oktobra 2022.

Vsi razredi se bodo pomerili v 25 igrah, ki bodo razporejene v okolici šole.

SE VIDIMO!

Spodaj je prikazana tabela, z rezultati razredov po nalogah. Prikazana so mesta v posameznih nalogah.

DOSEŽENO MESTO V POSAMEZNI NALOZI (ŠT. NALOGE)

MESTO	ŠTEVILO TOČK	RAZRED	1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	175	4. DE + DM	17	15	6	3	2	6	15	4	10	5	6	10	5	5
2	182	3. BT	1	17	7	4	5	9	4	8	1	1	3	9	1	8
3	211	2. AT	3	7	11	2	8	9	25	13	2	3	7	3	8	10

Slika 1: Domača stran spletne aplikacije

### 3.1 GLAVA SPLETNE APLIKACIJE NEPRIJAVLJENIM UPORABNIKOM

Slika 2: Glava spletne aplikacije strani neprijavljenim uporabnikom

Na sliki 2 je prikazana glava naše spletne aplikacije, ki se prikaže neprijavljenim uporabnikom. Kot lahko vidimo, imamo v levem kotu naslov spletne aplikacije. Torej, ko je bila aplikacija objavljena za testiranje, je bil naslov »ŠPORTNI DAN 2022 OKTOBER«. Desno od naslova pa lahko vidimo tri povezave, ki uporabnika popeljejo na različne strani v naši spletni aplikaciji. Leva ga pelje nazaj na domačo stran. Sredinska povezava ga pelje do strani, kjer je dostopen seznam z nalogami. Desna povezava pa ga pelje do strani z vsemi rezultati nalog. Čisto na desni strani pa imamo gumb, ki uporabnika popelje do prijavnice strani. Ta gumb je namenjen sodnikom in administratorjem.

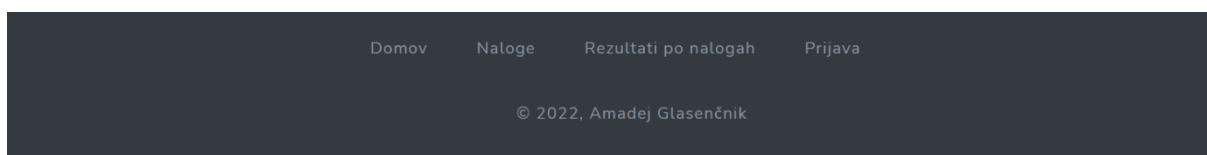
## 3.2 GLAVA SPLETNE APLIKACIJE PRIJAVLJENIM UPORABNIKOM



Slika 3: Glava spletne aplikacije prijavljenim uporabnikom

Slika 3 prikazuje glavo naše spletne aplikacije, ki se prikaže prijavljenim uporabnikom. Vidimo lahko, da je skoraj identična glavi, ki jo vidi neprijavljen uporabnik, le da imamo nekaj dodatnih elementov. Na mesto, da imamo gumb za prijavo, je tam sedaj gumb za odjavo. Zraven na levi strani pa se nam na začetku pojavi besedilo, ki pozdravi osebo, ki se je prijavila, tako da izpiše njegovo uporabniško ime. Zraven pa je gumb, ki prijavljenega uporabnika popelje do nadzorne plošče, ki pa jo bomo opisali malo kasneje.

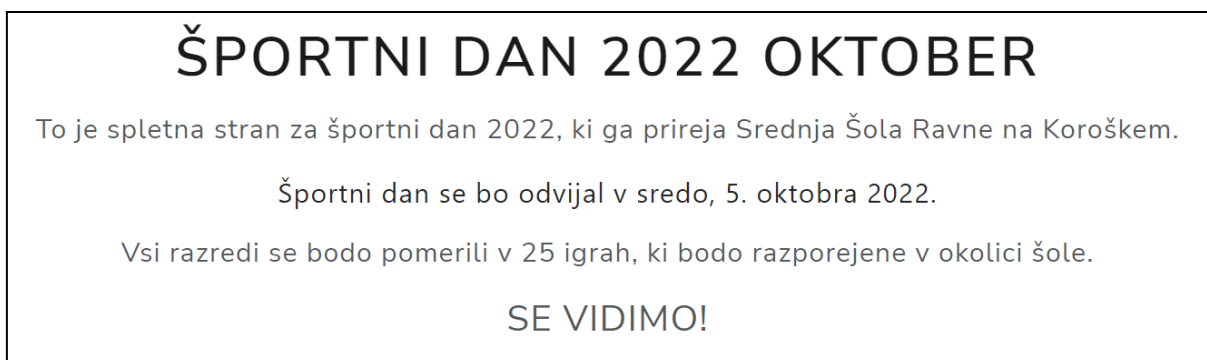
## 3.3 NOGA SPLETNE APLIKACIJE



Slika 4: Noga spletne aplikacije

Na sliki 4 je prikazana noga naše spletne aplikacije, ki se prikaže uporabnikom čisto na spodnjem delu spletne strani. Slika je v tem primeru približana. Kot lahko vidimo, so povezave enake kot v glavi. Torej imamo štiri povezave. Edina razlike je, da se po prijavi uporabnika spremeni le besedilo na povezavi za prijavo na odjavo. Spodaj pa imamo samo napis avtorskih pravic.

### 3.4 DOMAČA STRAN – UVODNO BESEDILO



Slika 5: Uvodno besedilo domače strani

Slika 5 prikazuje besedilo, ki je prikazano na domači strani. To besedilo je ob prihodu na domačo stran prebrano iz podatkovne baze, saj ga lahko kadarkoli uredi administrator na strani za urejanje domače strani (več o tej strani kasneje).

### 3.5 DOMAČA STRAN – REZULTATI TEKMOVALNIH SKUPIN

DOSEŽENO MESTO V POSAMEZNI NALOGI (ŠT. NALOGE)															
MESTO	ŠTEVILO TOČK	RAZRED	1	2	3	4	5	6	7	8	9	10	11	12	13
1	175	4. DE + DM	17	15	6	3	2	6	15	4	10	5	6	10	5
2	182	3. BT	1	17	7	4	5	9	4	8	1	1	3	9	1
3	211	2. AT	3	7	11	2	8	9	25	13	2	3	7	3	8

Slika 6: Doseženi rezultati po posameznih nalogah

Slika 6 prikazuje tabelo na domači strani. Tabela razvrsti vse tekmovalne skupine po mestih, izpiše, koliko točk ima vsaka skupina (skupen seštevek mest, torej manj je bolje), naprej pa so izpisana mesta, katera je dosegla ta ekipa pri vsaki nalogi posebej. Na sliki se zaradi pomanjkljivosti prostora vidijo le omejeni podatki.

## 3.6 STRAN S SEZNAMOM NALOG



Slika 7: Seznam nalog

Slika 7 prikazuje stran s seznamom vseh nalog. Tukaj spletna aplikacija iz podatkovne baze prebere vse naloge, ki jih je vnesel administrator ter prikaže tukaj nekaj osnovnih podatkov o tisti nalogi, kot so št. naloge, naslov, lokacija in rekviziti.

## 3.7 STRAN S PODROBNOSTMI NALOG



Slika 8: Podrobnosti naloge

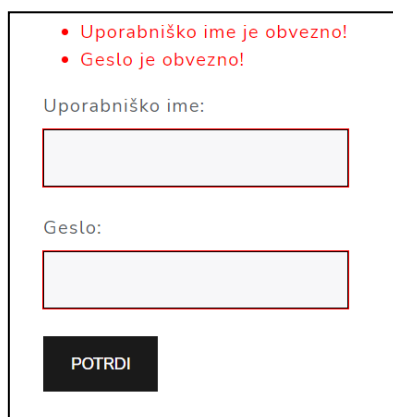
Na sliki 8 je prikazana stran, ki se pojavi ob kliku na gumb, na strani s seznamom nalog. Na tej strani izvemo vse podrobnosti o posamezni nalogi

ter tudi rezultate tekmovalnih skupin. Na samem vrhu je prikazan naslov ter številka naloge, nato sledijo lokacija, pravila in rekviziti. Pod temi podatki pa je potem prikazana tabela, ki prikazuje mesta, ki so jih dosegle tekmovalne skupine, zraven pa je tudi prikazan rezultat, ki ga je imela posamezna skupina.

### 3.8 STRAN Z VSEMI REZULTATI NALOG

Stran z vsemi rezultati nalog je zelo podobna strani, ki smo jo opisali prej. Torej na tej strani so prikazani enaki podatki, kot na strani s podrobnostmi nalog, le da so podatki prikazani za vse naloge in ne le za eno.

### 3.9 PRIJAVNA STRAN



The image shows a login form with the following elements:

- Two red error messages at the top: "Uporabniško ime je obvezno!" and "Geslo je obvezno!".
- A label "Uporabniško ime:" followed by a text input field.
- A label "Geslo:" followed by a text input field.
- A black button with the text "POTRDI" in white.

Slika 9: Prijavna stran

Na sliki 9 je prikazana prijavna stran. V tem primeru že ime pove, da se lahko uporabnik tukaj prijavi, da pridobi avtorizacijo do urejanja podatkov.

Uporabnik v polja vpiše svoje podatke (torej uporabniško ime in geslo) ter nato klikne na gumb POTRDI. Če katero od polj ni izpolnjeno, se napiše napaka, ki je vidna zgoraj. Enako se izpiše tudi napaka, če je uporabnik vnesel napačno uporabniško ime ali geslo.

### 3.10 NADZORNA PLOŠČA



Slika 10: Nadzorna plošča

Slika 10 prikazuje stran nadzorne plošče, ki se pokaže uporabniku ob prijavi. V tem primeru smo prijavljeni kot administrator, saj drugače ne bi videli zgornjega gumba. Zgornji gumb uporabnika popelje do strani za urejanje podatkov o sami aplikaciji. Sredinski gumb uporabnika popelje do ogleda vseh nalog. Zadnja pa ga popelje do strani za vpis rezultatov.

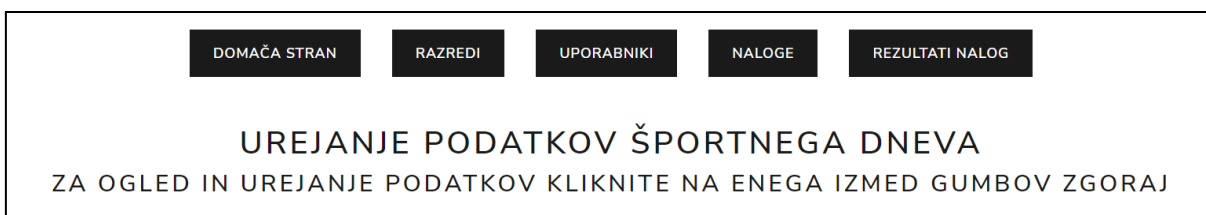
### 3.11 STRAN ZA VNOS REZULTATOV

The image shows a form for entering results. It contains a dropdown menu labeled 'Razred:' with a downward arrow. Below it is a text input field with the placeholder text 'Čas zapišite tako (primer: 43,3)'. Underneath is another text input field labeled 'Rezultat:' containing the number '0'. At the bottom left of the form is a dark button with the text 'SHRANI' in white.

Slika 11: Stran za vnos rezultatov

Slika 11 prikazuje primer strani, ki se pojavi sodniku, ko klikne na gumb za vnos rezultatov igre. Vsakemu sodniku se pokaže drugačna stran, saj ima vsak sodnik dostop do drugačne naloge. V vsakem primeru mora sodnik najprej izbrati v zgornjem seznamu razred oz. tekmovalno skupino, nato pa vnesti rezultat v spodnje polje, nato pa klikniti gumb SHRANI. Če je bilo shranjevanje uspešno, se mu to tudi izpiše.

## 3.12 STRAN Z MENIJEM ZA UREJANJE PODATKOV



Slika 12: Stran z menijem za urejanje podatkov

Na sliki 12 imamo prikazano stran, ki jo lahko vidi administrator po kliku na gumb za urejanje podatkov o spletni aplikaciji. Stran je samo enostaven meni, ki poziva uporabnika, da izbere med enim od gumbov zgoraj, glede na katero stvar hoče ta urejati v sami spletni aplikaciji. Na voljo ima urejanje domače strani, razredov, uporabnikov, nalog ter rezultatov nalog.

## 3.13 STRAN ZA UREJANJE UVODNEGA BESEDILA DOMAČE STRANI



Slika 13: Stran za urejanje uvodnega besedila domače strani

Slika 13 prikazuje stran za urejanje uvodnega besedila domače strani. Administrator preko okna uredi besedilo po želji, nato pa klikne na gumb SHRANI SPREMEMBE.

### 3.14 STRAN ZA UREJANJE RAZREDOV

UREJANJE RAZREDOV	
+ Dodaj Uredi Izbrisi	
ID razreda	Ime razreda
1	1. AT
2	1. AVS
3	1. BR
4	1. CR
5	1. DE
6	1. EM
7	1. OK
8	1. SA + 2. SA
9	2. AT
10	2. AVS

« < 1 2 3 > » 1 of 3 pages (25 items)

Slika 14: Stran za urejanje razredov/tekmovalnih skupin

Na sliki 14 je prikazana stran za urejanje razredov. Uporabnik mora klikniti na gumb dodaj za pojavitev okna za dodajanja razreda oz. tekmovalne skupine. S selekcijo enega izmed njih ter klikom na gumb uredi ali pa izbriši pa lahko izbran element tudi izbriše.

### 3.15 STRAN ZA UREJANJE UPORABNIKOV

UREJANJE UPORABNIKOV			
+ Dodaj Uredi Izbrisi			
ID uporabnika	Uporabniško ime	Geslo	Tip uporabnika
1	admin	nzX528vH#7	Admin
2	organizator	7#Ff5s6\$32	Admin
3	sodnik1	!c39A8YkY!	User
4	sodnik2	umMw82\$4K!	User
5	sodnik3	Rw@2z8S72&	User
6	sodnik4	5wu@E\$3e9*	User
7	sodnik5	*cBRGzz639	User
8	sodnik6	9kqS&8C9v6	User
9	sodnik7	52k4C#Ej6m	User
10	sodnik8	e8Sw33aEb#	User

« < 1 2 3 > » 1 of 3 pages (27 items)

Slika 15: Stran za urejanje uporabnikov

Slika 15 prikazuje stran za urejanje uporabnikov. Sem torej spadajo sodniki in administratorji. Uporabnika lahko doda novega uporabnika, pri katerem mora vnesti vse pomembne podatke. Lahko pa ga ob selekciji enega izmed njih tudi uredi ali pa celo izbriše. Uporabnikom se lahko dodelita dva naziva, to sta lahko: sodnik/user ali pa Admin.



### 3.16 STRAN ZA UREJANJE NALOG

UREJANJE NALOG								
+ Dodaj   Uredi   Izbriši								
ID naloge	Številka nalo...	Naslov	Lokacija	Rekviziti	Tip naloge	Razvrstitev	Št. igralcev	Pravila
1	1	TEK 4 x 10 MET...	STADION	štafetna palica, ...	Time	Ascending	1	<ul><li style="..."
2	2	STRELJANJE Z Z...	STADION	puška, metki, pl...	Points	Descending	4	<ul><li style="..."
3	3	SKOK V DALJIN...	STADION	meter	Distance	Descending	3	<ul><li style="..."
4	4	NOŠENJE ŽOGLI...	STADION	badminton lopa...	Time	Ascending	1	<ul><li style="..."
5	5	SUVANJE KROG...	STADION	krogla, meter	Distance	Descending	5	<ul><li style="..."
6	6	HOKEJ - STRELI ...	STADION	mali gol, 2x sto...	Points	Descending	4	<ul><li style="..."
7	7	TEK 4 x 60 MET...	STADION	štafetna palica, ...	Time	Ascending	1	<ul><li style="..."
8	8	PRESKOKI NIZK...	STADION	nizka klop, štop...	Points	Descending	4	<ul><li style="..."
9	9	PRENAŠANJE TE...	MODRA TRIBUN...	težka žoga, što...	Time	Ascending	1	<ul><li style="..."
10	10	PRENAŠANJE TE...	RDEČA TRIBUN...	težka žoga, što...	Time	Ascending	1	<ul><li style="..."

1 of 3 pages (25 items)

Slika 16: Stran za urejanje nalog

Na sliki 16 je prikazana stran za urejanje nalog. Tukaj lahko uporabnik doda uredi ali izbriše naloge po enakem postopku, kot je lahko prejšnje elemente v straneh za urejanje, le da mora vnesti pravilne podatke.

### 3.17 STRAN ZA UREJANJE REZULTATOV NALOG

UREJANJE REZULTATOV IGER			
+ Dodaj   Uredi   Izbriši			
ID rezultata igre	Razred	Št. naloge	Rezultat
4	1. SA + 2. SA	8	193
5	3. AVS	20	18
6	4. DT	24	51
7	1. AT	1	54,4
8	1. AVS	2	10
9	3. AT	15	83
10	3. BT	16	15
11	4. CR	22	2
12	3. CR	17	15
13	2. AVS	10	22,8

1 of 63 pages (625 items)

Slika 17: Stran za urejanje rezultatov nalog

Na zgornji sliki je prikazana stran za urejanje rezultatov iger. Tukaj lahko spet uporabnik po istem sistemu kot prej doda, uredi ali izbriše rezultate posameznih iger. Tukaj se nahaja najbolj osnovni seznam vseh rezultatov, za vsako nalogo vsakega razreda.

## 4 UPORABLJENA TEHNOLOGIJA

V tem poglavju bomo opisali vso tehnologijo oz. programska orodja, ki smo jih uporabili pri izdelavi naše spletne aplikacije. Odločali smo se med večimi različnimi orodji. Izbrali smo orodja, ki so bila nam najbližja, poznana, uporabna in dostopna.

### 4.1 HTML

HTML je kratica za HyperText Markup Language, ki je označevalni jezik za izdelavo spletnih strani. Z njim opišemo gradiva, ki jih želimo objaviti na spletu. Poleg lahko spreminjamo tip pisave, velikost in stil, vključujemo lahko slike, povezave, tabele, obrazce ipd. [1]

HTML lahko pišemo v vsakem urejevalniku besedil (kot npr. beležnici, Notepad++, Sublime Text, itd.), saj je zapisan v obliki HTML-elementov, ki so sestavljeni iz oznak, zapisanih v koničastih oklepajih (npr. kot **<p>** ) znotraj vsebine spletne strani. HTML oznake so običajno zapisane v parih, kot npr. **<p>** in **</p>**. Prva oznaka se imenuje tudi začetna značka in druga oznaka končna značka. Znotraj para oznak lahko oblikovalec spletne strani vpiše poljubno besedilo in tudi druge oznake (gnezdenje oznak). [1]

Osnova vseh spletnih strani je HTML. Tudi dinamične spletne strani, ki omogočajo interakcijo med uporabnikom in spletno stranjo, so kombinacija programskega jezika in jezika HTML. Trenutna različica HTML je HTML5. [1]



Slika 18: Najpogosteje uporabljen logotip HTML5 [2]

## 4.2 CSS

Kaskadne slogovne predloge (angl. Cascading Style Sheets), poznane pod kratico CSS, so predloge, predstavljene v obliki preprostega slogovnega jezika, ki skrbi za prezentacijo spletnih strani. [3]

CSS omogoča izdelovalcem spletnih strani, da predpišejo obliko posameznim elementom. Večina elementov v HTML je namreč namenjena logičnemu oblikovanju, kjer samo določimo, kakšne vrste je posamezen element (slika, tabela, vrstica v tabeli, celica v vrstici, seznam, točka seznama, odstavek, indeks, eksponent, naslov, aktivna povezava ...), brskalnik pa te elemente oblikuje po svoje. [3]

Z uporabo stilov CSS lahko elementom določimo celo vrsto oblikovnih lastnosti, med katere spadajo ozadje, robovi, razmiki, odmiki, pisava, poravnava, barva ... CSS nam omogoča, da oblikovne lastnosti določimo ločeno od vsebine, kar poveča preglednost napisane kode. [3]



Slika 19: Najpogosteje uporabljen logotip CSS [4]

### 4.2.1 Bootstrap 5

Bootstrap je brezplačen odprtokodni front-end razvojni okvir za ustvarjanje spletnih mest in spletnih aplikacij. Bootstrap je zasnovan tako, da omogoča odziven razvoj mobilnih spletnih mest. Bootstrap nam ponuja zbirko sintakse za predloge, ki so kasneje uporabljeni v naših spletnih straneh. Spletni razvijalci, ki uporabljajo Bootstrap, lahko gradijo spletna mesta veliko hitreje, ne da bi porabili čas za skrb glede osnovnih ukazov in funkcij [5]. Bootstrap-ova spletna stran z vso potrebno razvojno dokumentacijo se nahaja na naslednji spletni strani: <https://getbootstrap.com/>.

## 4.3 JAVASCRIPT

JavaScript je programski jezik, ki spada v skupino tolmačev. To pomeni, da se napisana koda programa ne prevede v program, ki bi ga lahko shranili v obliki izvršilne datoteke, pač pa poseben tolmač po vrsti razčlenjuje in izvršuje ukaze, ki smo jih napisali. [6]

JavaScript je jezik, ki dopolnjuje obstoječe jezike. V kombinaciji s PHP-jem se izkaže za zelo učinkovito orodje ravno zaradi osveževanja spletne strani. Dogodki se namreč izvajajo na uporabnikovem računalniku in ne na strežniku. To ima za posledico vidnost izvorne kode, ki v tem primeru, za razliko od PHP-ja, ni skrita očem uporabnikom. [6]

## 4.4 C#

C# (izgovarja se kot "See Sharp") je sodoben, objektno usmerjen in tipsko varen programski jezik. C# razvijalcem omogoča izdelavo številnih vrst varnih in robustnih aplikacij, ki delujejo v .NET. C# ima svoje korenine v družini jezikov C in ga bodo programerji C, C++, Java in JavaScript takoj poznali. [7]

C# je objektno usmerjen, komponentno usmerjen programski jezik. C# ponuja jezikovne konstrukcije za neposredno podporo tem konceptom, zaradi česar je C# naraven jezik za ustvarjanje in uporabo programskih komponent. Od svojega nastanka je C# dodal funkcije za podporo novim delovnim obremenitvam in nastajajočim praksam načrtovanja programske opreme. V svojem bistvu je C# objektno usmerjen jezik, saj vi določite vrste spremenljivk in njihovo vedenje. [7]



Slika 20: Najpogosteje uporabljen logotip C# [8]

## 4.5 .NET

.NET (izg. dotnet) je brezplačna odprtokodna razvijalska platforma za različne platforme za gradnjo različnih vrst aplikacij. Z .NET lahko uporabite več jezikov, urejevalnikov in knjižnic za izdelavo za splet, mobilne naprave, namizne računalnike, igre, IoT in še več. Aplikacije .NET lahko pišete v C#, F# ali Visual Basic. Ne glede na to, ali delate v C#, F# ali Visual Basic, bo vaša koda izvirno delovala v katerem koli združljivem operacijskem sistemu. Z .NET lahko ustvarite veliko vrst aplikacij. Nekateri so medplatformski, nekateri pa ciljajo na določen nabor operacijskih sistemov in naprav. [9]



Slika 21: Uradni logotip .NET [10]

## 4.6 BLAZOR SERVER

Blazor je spletno ogrodje za gradnjo komponent spletnega uporabniškega vmesnika (komponente Razor), ki jih je mogoče gostiti na različne načine. Komponente Razor se lahko izvajajo na strani strežnika v ASP.NET Core (Blazor Server) ali na strani odjemalca v brskalniku v izvajalnem okolju .NET, ki temelji na WebAssembly (Blazor WebAssembly, Blazor WASM). Komponente Razor lahko gostite tudi v izvornih mobilnih in namiznih aplikacijah, ki se upodabljajo v vdelan nadzor spletnega pogleda (Blazor Hybrid). Ne glede na model gostovanja je način gradnje komponent Razor enak. Iste komponente Razor lahko uporabljate s katerim koli modelom gostovanja nespremenjeno. [11]



Slika 22: Uradni logotip Blazor [12]

## 4.7 ENTITY FRAMEWORK CORE oz. EF CORE

Entity Framework Core je nova različica Entity Framework-a po EF 6.x. Je odprtokodna, lahka, razširljiva in večplatformska različica tehnologije dostopa do podatkov omenjenega ogrodja. [13]

Le-to je ogrodje objektnega/relacijskega preslikave (O/RM). Je izboljšava ADO.NET, ki daje razvijalcem avtomatiziran mehanizem za dostop in shranjevanje podatkov v bazi podatkov. [13]

EF Core je namenjen uporabi z aplikacijami .NET Core. Lahko pa se uporablja tudi s standardnimi aplikacijami, ki temeljijo na ogrodju .NET 4.5+. [13]

Ogrodje podpira dva razvojna pristopa 1) Code-First 2) Database-First. EF Core je v glavnem usmerjen na pristop najprej koda in nudi malo podpore za pristop najprej baza podatkov, ker vizualni oblikovalec ali čarovnik za model DB ni podprt od EF Core 2.0. [13]

## 4.8 SYNCFUSION

Syncfusion Blazor Components je sodobna izvorna knjižnica komponent uporabniškega vmesnika podjetja za ustvarjanje Blazor WebAssembly in strežniških aplikacij. Knjižnica je bila zgrajena že od začetka, tako da je lahka, odzivna, modularna in prijazna do dotika. Komponente Syncfusion Blazor so izvorne komponente Blazor-ja in ne ovojni komponent EJ2. [14]



Slika 23: Uradni logotip Syncfusion [15]

## 4.9 SQL SERVER

Microsoft SQL Server je sistem za upravljanje relacijskih baz podatkov, ki ga je razvil Microsoft. Kot strežnik baz podatkov je programski izdelek s primarno funkcijo shranjevanja in pridobivanja podatkov, kot jih zahtevajo druge programske aplikacije – ki se lahko izvajajo na istem računalniku ali na drugem računalniku v omrežju (vključno z internetom). Microsoft trži vsaj ducat različnih izdaj strežnika Microsoft SQL Server, namenjenih različnim občinstvom in za delovne obremenitve, ki segajo od majhnih aplikacij z enim samim strojem do velikih aplikacij, usmerjenih v internet, s številnimi sočasnimi uporabniki. [16]



Slika 24: Uradni logotip SQL Server [17]

## 4.10 VISUAL STUDIO 2022

Visual Studio je Microsoftovo integrirano razvojno okolje (IDE). Uporablja se za razvoj računalniških programov, vključno s spletnimi mesti, spletnimi aplikacijami, spletnimi storitvami in mobilnimi aplikacijami. Program uporablja Microsoftove platforme za razvoj programske opreme, kot so Windows API, Windows Forms, Windows Presentation Foundation, Windows Store in Microsoft Silverlight. Visual Studio vključuje urejevalnik kode, ki podpira IntelliSense (komponento za dokončanje kode) in preoblikovanje kode. Integrirani razhroščevalnik in še veliko več. [18]



Slika 25: Uradni logotip Visual Studio 2022 [18]

## 5 IZDELAVA SPLETNE APLIKACIJE

V tem poglavju se bomo sprehodili skozi kodo naše spletne aplikacije. Zaradi obširnosti same aplikacije ne bomo opisali ter vključili vse kode ter opisali celotnega postopka izdelave aplikacije, ampak bomo opisali le nam najpomembnejše komponente in dele kode. Ta bo na voljo na spletni povezavi, ki bo dostopna v poglavju: PRILOGE.

### 5.1 USTVARJANJE PODATKOVNE BAZE Z EF CORE

V tem poglavju si bomo pogledali, kako smo ustvarili našo podatkovno bazo z orodjem Entity Framework Core v SQL Server.

Prvi korak pri delu z EF Core-om je namestitev potrebnih paketov (nuget packages). Ti so:

- Microsoft.EntityFrameworkCore
- Microsoft.EntityFrameworkCore.Design
- Microsoft.EntityFrameworkCore.SqlServer
- Microsoft.EntityFrameworkCore.Tools

Po namestitvi paketov lahko začnemo s programiranjem. Najprej si po primeru dobre prakse ustvarimo nekaj map. Te bodo imenovane Models in Data. Vse tabele bodo shranjene v mapi Models in bodo imele tudi ime: imeTabele**Model**.cs. V mapi Data pa bo naš razred ApplicationDbContext.cs, ki bo služil kot neke vrste povezava med C# in podatkovno bazo.

Ko imamo te mape ustvarjene in to datoteko narejeno, moramo v glavni datoteki Program.cs dodati nekaj kode:

```
builder.Services.AddDbContext<ApplicationDbContext>(opt => opt.UseSqlServer(  
    builder.Configuration.GetConnectionString("DefaultConnection")  
));
```

Ta koda nam bo dodala povezavo od baze v program, tako da lahko začnemo delati z bazo, še prej pa moramo v datoteko appsettings.json vključiti naš niz s podatki za povezavo do baze. Ko bomo naredili naše tabele in jih vključili v ApplicationDbContext, pa moramo v Package Manager Console vedno zagnati naslednja ukaza:



- add-migration <ime-migracije> – S tem kreiramo novo migracijo (znotraj so vse spremembe, ki jih hočemo narediti z bazo).
- Update-database – Ta ukaz pa nam potem zažene vse nove migracije in jih uporabi na podatkovni bazi.

### 5.1.1 Ustvarjanje tabele AppData

```
[Table("AppData")]
public class AppDataModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public DataType DataType { get; set; }

    [Required]
    public string Value { get; set; } = "";
}

public enum DataType
{
    None,
    HomePageHtml
}
```

Zgoraj lahko vidimo kodo, ki je uporabljena za ustvarjanje tabele **AppData**. V njej hranimo »nastavitve« spletna aplikacije. Kot lahko vidimo, pa moramo ustvariti tudi **Enum**, ki bo shranjeval tip podatkov.

### 5.1.2 Ustvarjanje tabele Class

```
[Table("Class")]
public class ClassModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public string Name { get; set; } = null!;

    public ICollection<GameResultModel> GameResults { get; set; } = null!;
}
```

Zgornja koda je uporabljena za ustvarjanje tabele **Class**. V tej tabeli hranimo podatke o razredih. V predzadnji vrstici pa je del kode, ki je namenjen predstavljanju odnosov do drugih tabel, torej v našem primeru **GameResults**, kjer nakažemo povezavo 1-M (1 proti Mnogo).

### 5.1.3 Ustvarjanje tabele Exercise

Na naslednji strani je prikazana koda, ki je uporabljena za ustvarjanje tabele **Exercise**. Kot pove že samo ime, bomo v tej tabeli shranjevali vse podatke o samih nalogah, vključno z njihovim tipom (čas, dolžina, točke).

```

[Table("Exercise")]
public class ExerciseModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public int ExerciseNumber { get; set; }

    [Required]
    public string Title { get; set; } = null!;

    [Required]
    public string Location { get; set; } = null!;

    [Required]
    public string Rules { get; set; } = null!;

    [Required]
    public string Props { get; set; } = null!;

    [Required]
    public ExerciseType ExerciseType { get; set; }

    [Required]
    public OrderType OrderType { get; set; }

    [Required]
    public int NumberOfPlayers { get; set; }

    public ICollection<UserExerciseModel>
    UserExercises { get; set; } = null!;
}

public enum ExerciseType
{
    Time,
    Distance,
    Points
}

public enum OrderType
{
    Ascending,
    Descending
}
    
```

## 5.1.4 Ustvarjanje tabele User

```

[Table("User")]
public class UserModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public string Username { get; set; } = null!;

    [Required]
    public string Password { get; set; } = null!;

    [Required]
    public UserType UserType { get; set; }

    public ICollection<UserExerciseModel>
    UserExercises { get; set; } = null!;
}

public enum UserType
{
    User,
    Admin
}
    
```

Prikazana koda je uporabljena za ustvarjanje tabele **User**. V njej bomo hranili vse podatke o uporabnikih oz. sodnikih, ki jih bo vnesel administrator.

## 5.1.5 Ustvarjanje tabele UserExercise

Koda na naslednji strani je uporabljena za ustvarjanje tabele **UserExercise**. V njen bodo shranjeni podatki o tem, katere naloge lahko dostopa kateri sodnik zato, da ti ne bi mogli spreminjati rezultatov nalogam, za katere niso pooblašeni.

```
[Table("UserExercise")]
public class UserExerciseModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public int UserId { get; set; }

    [Required]
    public int ExerciseId { get; set; }

    public UserModel User { get; set; } = null!;
    public ExerciseModel Exercise { get; set; } = null!;
}
```

## 5.1.6 Ustvarjanje tabele **GameResult**

```
[Table("GameResult")]
public class GameResultModel
{
    [Key]
    public int Id { get; set; }

    [Required]
    public int ClassId { get; set; }

    [Required]
    public int ExerciseId { get; set; }

    [Required]
    public double Result { get; set; }

    public string PlayerResults { get; set; } = null!;

    public ClassModel Class { get; set; } = null!;
    public ExerciseModel Exercise { get; set; } = null!;
}
```

Koda zgoraj je uporabljena za ustvarjanje tabele **GameResult**. V tej tabeli bomo shranjevali vse rezultate iger. Shranjevali bomo torej razred, ki je tekmoval. Nato pa tudi nalogo ter njihov dosežen rezultat.

## 5.1.7 Ustvarjanje razreda **ApplicationDbContext**

Že v prejšnjih poglavjih smo omenjali razred `ApplicationDbContext` kot neke vrste povezavo do naše podatkovne baze. V njem vključimo vse tabele zato, da bodo te v naslednjem koraku tudi v resnici narejene na SQL Server.

Koda za vključevanje tabel je prikazana spodaj:

```
public DbSet<ClassModel> Class { get; set; } = null!;
public DbSet<UserModel> User { get; set; } = null!;
public DbSet<ExerciseModel> Exercise { get; set; } = null!;
public DbSet<UserExerciseModel> UserExercise { get; set; } = null!;
public DbSet<AppDataModel> AppData { get; set; } = null!;
public DbSet<GameResultModel> GameResult { get; set; } = null!;
```

## 5.2 RAZRED ZA PRIJAVO

```

public async Task LoginAsync(UserModel loginFormModel)
{
    var (userInDatabase, isSuccess) = LookUpUser(loginFormModel.Username, loginFormModel.Password);
    var principal = new ClaimsPrincipal();

    if (isSuccess)
    {
        var identity = CreateIdentityFromUser(userInDatabase);
        principal = new ClaimsPrincipal(identity);
        await _protectedLocalStorage.SetAsync("identity", JsonConvert.SerializeObject(userInDatabase));
    }

    NotifyAuthenticationStateChanged(Task.FromResult(new AuthenticationState(principal)));
}

public async Task LogoutAsync()
{
    await _protectedLocalStorage.DeleteAsync("identity");
    var principal = new ClaimsPrincipal();
    NotifyAuthenticationStateChanged(Task.FromResult(new AuthenticationState(principal)));
}

private ClaimsIdentity CreateIdentityFromUser(UserModel user)
{
    return new ClaimsIdentity(new Claim[]
    {
        new (ClaimTypes.Name, user.Username),
        new (ClaimTypes.Hash, user.Password),
        new (ClaimTypes.Role, user.UserType.ToString())
    }, "sportni-dan-v2");
}

private (UserModel, bool) LookUpUser(string username, string password)
{
    var result = _dataProviderService.User.FirstOrDefault(u => username == u.Username && password == u.Password);
    return (result, result is not null);
}

```

Eden izmed razredov, ki se nam je zdel zelo pomemben, da ga omenimo, je razred, ki omogoča prijavo in odjavo uporabnikov. Ta razred se imenuje: `AuthenticationService`. Zaradi velikosti datoteke bomo vključili in opisali le funkcijo glavnih metod, ki jih imamo zapisane v tem razredu.

Prva metoda se uporablja takrat, ko se uporabnik poskusi prijaviti. Metoda v svojem delovanju kliče metodo, ki je zapisana zadnja v naši kodi ter preveri, če se uporabniško ime in geslo ujemata s katerim od uporabnikov v podatkovni bazi. Če se, potem prijavi uporabnika ter zapiše njegove podatke v lokalno shrambo, kar pa naredi s pomočjo tretje metode po vrsti.

Druga metoda po vrsti je uporabljena za odjavo uporabnikov, kjer pa metoda samo izbriše podatke, ki so zapisani v lokalni shrambi.

## 5.3 STRAN ZA VNOS REZULTATOV

```

@inherits InsertResultsBase
@page "/insertresults/{ExerciseId:int}"

<pageTitle>@($"{Vpis {Exercise.ExerciseNumber}. Naloga")</pageTitle>

<div class="row">
  <h1 class="text-center">@($"{Exercise.ExerciseNumber}. NALOGA: {Exercise.Title}")</h1>
  <div class="col-md-4">
    <EditForm Model="inputModel" OnValidSubmit="@FormSubmit" Context="EditForm">
      <DataAnnotationsValidator />
      <ValidationSummary />

      <div class="form-group mt-4">
        <label class="input-label">Razred:</label>
        <div class="wrapper mt-2">
          <input select @bind-Value="inputModel.ClassId" class="form-select input-border"
onfocus='this.size=5;' onblur='this.size=1;' onchange='this.size=1; this.blur();'>
            @foreach (var item in Classes)
            {
              <option value="@item.Id">@item.Name</option>
            }
          </input select>
        </div>
      </div>

      @if (Exercise.ExerciseType == ExerciseType.Time)
      {
        <p class="mt-4">Čas zapišite tako (primer: 43,3)</p>
      }
      else
      {
        <p class="mt-4">Tudi če je rezultat 0, to vseeno vpišite.</p>
      }

      @if (Exercise.NumberOfPlayers > 1)
      {
        @foreach (var i in Enumerable.Range(0, Exercise.NumberOfPlayers))
        {
          var ii = i;
          int num = i + 1;

          <div class="form-group mt-4">
            <label class="input-label">@($"{Dijak {num}:}")</label>
            <div class="wrapper mt-2">
              <input type="number" value="@PlayerInputsList[ii]" @oninput="(e) => resum(ii, e)"
step=".1" min="0" class="form-control input-border" placeholder="0,0" required />
            </div>
          </div>
        }
      }

      <div class="form-group mt-4">
        <label class="input-label">Rezultat:</label>
        <div class="wrapper mt-2">
          <input number @bind-Value=inputModel.Result @bind-Value:format="F2" step=".1" min="0"
placeholder="0,0" class="form-control input-border" />
        </div>
      </div>

      <div class="mt-4">
        <button type="submit" class="btn btn-primary">Shrani</button>
      </div>
      <p class="text-danger error-message mt-4">@errorMessage</p>
    </EditForm>
  </div>
</div>

```

Zgornja koda je uporabljena za vnos rezultatov. To kodo tudi imenujemo »front-end«, kjer sprogramiramo le del, ki ga vidi nato uporabnik. V drugem delu kode, ki je na naslednji strani, pa je prikazana koda, ki je uporabljena za samo logiko in shranjevanje rezultatov. Na tej strani, torej iz podatkovne baze prejmemo podatke o sami nalogi in te podatke tudi prikažemo v spletni

aplikaciji. Nato se glede na podatke ustvarijo polja, ki jih mora sodnik vnesti, s tem pa kličemo metode, ki so prikazane na kodi spodaj.

```
protected async Task FormSubmit()
{
    try
    {
        if (inputModel != null && inputModel.ClassId != 0)
        {
            var existingResult = db.GameResult.FirstOrDefault(x => x.ClassId == inputModel.ClassId &&
x.ExerciseId == Exercise.Id);

            string playerResults = "";

            foreach (var item in PlayerInputsList)
            {
                playerResults += $"{item.ToString()};";
            }

            if (existingResult != null)
            {
                existingResult.Result = inputModel.Result;
                existingResult.PlayerResults = playerResults;
            }
            else
            {
                await db.GameResult.AddAsync(new GameResultModel()
                {
                    ExerciseId = Exercise.Id,
                    ClassId = inputModel.ClassId,
                    Result = inputModel.Result,
                    PlayerResults = playerResults
                });
            }

            await db.SaveChangesAsync();

            ResetInputs();

            toast.ShowSuccess("Vpis naloge uspešen!");
        }
        else errorMessage = "Nepopoln vpis!";
    }
    catch
    {
        toast.ShowError("Napaka pri vpisu!");
    }
}
```

V tej kodi torej najprej preverimo, da so vsa polja bila izpolnjena. Po prvem preverjanju nato tudi preverimo, če že rezultat te naloge obstaja za razred, ki ga je vnesel sodnik. Če obstaja, potem rezultat samo spremenimo, v nasprotnem primeru pa ga dodamo. Na koncu pa izpišemo sporočilo, da je bilo shranjevanje podatkov uspešno.

## 5.4 PRIMER STRANI ZA UREJANJE PODATKOV / UPORABA SYNCFUSION DATAGRID

Za konec poglavja o izdelavi spletne aplikacije bomo prikazali še primer strani za urejanje razredov oz. tekmovalnih skupin, ki jih lahko ureja administrator. S tem bomo prikazali delovanje Syncfusion komponent.

Za začetek moramo namestiti nuget paket: **Syncfusion.Blazor**.

Nato moramo dodati ključ do licence Syncfusion v appsettings.json. Po tem pa moramo dodati sledečo vrstico v Program.cs, da registriramo storitev:

```
SyncfusionLicenseProvider.RegisterLicense(configuration["SyncfusionLicense"]);
```

Nato smo morali v \_Layout.razor dodati še naslednji dve vrstici:

```
<link href="_content/Syncfusion.Blazor/styles/bootstrap5.css" rel="stylesheet" />
<script src="_content/Syncfusion.Blazor/scripts/syncfusion-blazor.min.js" type="text/javascript"></script>
```

S tem pa smo uspešno vključili Syncfusion Blazor komponente v našo spletno aplikacijo.

```
@inherits ClassEditBase
@page "/edit/class"

<PageTitle>Urejanje Razredov – Športni Dan 2022 Oktober</PageTitle>

<EditNavigation CurrentPage=@EditPageType.Class />

<h1 class="text-center my-5">Urejanje Razredov</h1>
<div class="col-lg-12 control-section">
  <div class="content-wrapper">
    <div class="row">
      <SfGrid @ref="Grid" DataSource=@Classes" Toolbar="ToolbarItems" AllowPaging="true"
AllowSorting="true" AllowFiltering="true">
        <GridPageSettings PageSize="10"></GridPageSettings>
        <GridEvents TValue="ClassModel" OnToolbarClick="ToolbarClickHandler"></GridEvents>
        <GridColumns>
          <GridColumn Field=@nameof(ClassModel.Id) HeaderText="ID razreda" IsPrimaryKey="true"
TextAlign="TextAlign.Center"></GridColumn>
          <GridColumn Field=@nameof(ClassModel.Name) HeaderText="Ime razreda"
TextAlign="TextAlign.Center"></GridColumn>
        </GridColumns>
      </SfGrid>
    </div>
  </div>
</div>
```

Zgoraj je prikazana koda za ustvarjanje tabele oz. SfGrid Syncfusion komponente, ki nam naredi tabelo glede na podatke, ki jih dobimo iz podatkovne baze. Nato smo naredili, da lahko ob klikih na gumbe urejamo podatke o tej tabeli oz. v tem primeru razredov.

## 6 ZAKLJUČEK

V raziskovalni nalogi smo izdelali inovativno spletno aplikacijo za vodenje športnih dni in prikaz rezultatov. Naša spletna rešitev deluje na vseh novejših spletnih brskalnikih, pametnih telefonih in tablicah. S tem smo 1. hipotezo (Spletna rešitev mora biti inovativna oziroma da takšna še ne obstaja) tudi potrdili.

S spletno aplikacijo smo zmanjšali ročno prepisovanje podatkov, uporabo pisarniškega materiala ter zmanjšali ogljični odtis na okolje in s tem potrdili 2. hipotezo (Spletna aplikacija mora biti ekološko sprejemljiva).

Skozi proces raziskave smo ugotovili, da aplikacija skrbnikom omogoča ažurno dodajanje in urejanje nalog, sodnikom pa hitrejše dostopanje in vnašanje rezultatov iz različnih lokacij/športnih postaj. Arhivi podatkov niso več potrebni v fizični obliki, saj so shranjeni le v elektronski. Ti podatki potrjujejo 3. hipotezo (Spletna aplikacija mora omogočati ažuren vnos, urejanje in ogled podatkov).

Na koncu našega raziskovanja smo spoznali, da statističnega pregleda podatkov za leta nazaj na naši šoli žal nimamo v elektronski obliki. Spletna aplikacija je nastala v lanskem šolskem letu, zato se 4. hipoteza (Spletna rešitev mora zagotoviti statistični pregled podatkov za prejšnja leta) delno potrdi. Izdelana spletna rešitev je izziv za nadaljnje delo. V prihodnje želimo izboljšati grafične prikaze zbranih podatkov, dodajanje podatkov o posameznikih, razglasitev zmagovalne ekipe in najuspešnejšega posameznika.

V prihodnje smo si zadali cilj, da lahko implementiramo aplikacijo za športni karton, da bi lahko dodali imena in priimke dijakov, ki tekmujejo. Pri tem je pomembno tudi varstvo podatkov.



## 7 SEZNAM LITERATURE

- [1] eNSA: Uvod v HTML. Dostopno na naslovu: <https://nsa-splet.si/html/uvod/html-uvod-01.php>  
(nazadnje obiskano: 15. 3. 2023)
- [2] Fotografija: <https://en.wikipedia.org/wiki/HTML>  
(nazadnje obiskano: 15. 3. 2023)
- [3] eNSA: Kaj je CSS. Dostopno na naslovu: <https://nsa-splet.si/css/uvod/css-uvod-02-kajJeCss.php>  
(nazadnje obiskano: 15. 3. 2023)
- [4] Fotografija: <https://en.wikipedia.org/wiki/CSS>  
(nazadnje obiskano 15. 3. 2023)
- [5] ZOLA, Andrew: What is a Bootstrap and how does it work?  
Dostopno na naslovu:  
<https://www.techtarget.com/whatis/definition/bootstrap>  
(nazadnje obiskano: 15. 3. 2023)
- [6] eNSA: Osnove JavaScripta. Dostopno na naslovu: <https://nsa-splet.si/js/uvod/js-uvod-01.php>  
(nazadnje obiskano: 15. 3. 2023)
- [7] Microsoft: A tour of the C# language. Dostopno na naslovu:  
<https://learn.microsoft.com/en-us/dotnet/csharp/tour-of-csharp/>  
(nazadnje obiskano: 15. 3. 2023)
- [8] Fotografija:  
[https://en.wikipedia.org/wiki/C\\_Sharp\\_%28programming\\_language%29](https://en.wikipedia.org/wiki/C_Sharp_%28programming_language%29)  
(nazadnje obiskano: 15. 3. 2023)

- [9] Microsoft: What is .NET?. Dostopno na naslovu: <https://dotnet.microsoft.com/en-us/learn/dotnet/what-is-dotnet>  
(nazadnje obiskano: 15. 3. 2023)
  
- [10] Fotografija: <https://upload.wikimedia.org/wikipedia/commons/thumb/7/7d/Microsoft .NET logo.svg/2048px-Microsoft .NET logo.svg.png>  
(nazadnje obiskano: 15. 3. 2023)
  
- [11] Microsoft: ASP.NET Core Blazor hosting models. Dostopno na naslovu: <https://learn.microsoft.com/en-us/aspnet/core/blazor/hosting-models?view=aspnetcore-6.0>  
(nazadnje obiskano: 15. 3. 2023)
  
- [12] Fotografija: <https://upload.wikimedia.org/wikipedia/commons/d/d0/Blazor.png>  
(nazadnje obiskano: 15. 3. 2023)
  
- [13] EntityFrameworkTutorial.net: Entity Framework Core. Dostopno na naslovu: <https://www.entityframeworktutorial.net/efcore/entity-framework-core.aspx>  
(nazadnje obiskano: 15. 3. 2023)
  
- [14] Syncfusion: Welcome to Syncfusion Blazor Components. Dostopno na naslovu: <https://blazor.syncfusion.com/documentation/introduction>  
(nazadnje obiskano: 15. 3. 2023)
  
- [15] Fotografija: [https://cdn.syncfusion.com/content/images/company-logos/Syncfusion\\_Logo\\_Image.png](https://cdn.syncfusion.com/content/images/company-logos/Syncfusion_Logo_Image.png)  
(nazadnje obiskano: 15. 3. 2023)

- [16] Wikipedia: Microsoft SQL Server. Dostopno na naslovu:  
[https://en.wikipedia.org/wiki/Microsoft\\_SQL\\_Server](https://en.wikipedia.org/wiki/Microsoft_SQL_Server)  
(nazadnje obiskano: 15. 3. 2023)
  
- [17] Fotografija: <https://www.sqlservertutorial.net/wp-content/uploads/sql-server-tutorial.svg>  
(nazadnje obiskano: 15. 3. 2023)
  
- [18] Wikipedia: Visual Studio. Dostopno na naslovu:  
[https://en.wikipedia.org/wiki/Visual\\_Studio](https://en.wikipedia.org/wiki/Visual_Studio)  
(nazadnje obiskano: 15. 3. 2023)

## **8 PRILOGE**

Celoten projekt naše spletne aplikacije je na voljo na naslednji spletni povezavi: <https://github.com/aglasencnik/SportniDanV2Public>

(Spletna stran, nazadnje obiskano: 15. 3. 2023)