ივანე ჯავახიშვილის სახელობის თბილისის

სახელმწიფო უნივერსიტეტი

Ivane Javakhishvili Tbilisi State University

საბაკალავრო ნაშრომი თემაზე:

Thesis on:

პროგრამული სისტემები სპორტის მენეჯმენტში ძიუდოს მაგალითზე

Sports management systems using a judo example

გიორგი მეცხვარიშვილი

Giorgi Metskhvarishvili

ზუსტ და საბუნებისმეტყველო მეცნიერებათა ფაკულტეტი

კომპიუტერული მეცნიერების დეპარტამენტი.

Faculty of Exact and Natural Sciences Department of Computer Sciences

ხელმძღვანელი: პროფესორი მანანა ხაჩიძე

Supervisor: Professor Manana Khachidze

თბილისი 2018

Tbilisi 2018

რეზიუმე - პროგრამული სისტემები თანამედროვე სპორტში აქტიურად გამოიყენება სხვადასხვა სახის მენეჯმენტსა და ადმინისტრაციულ საკითხებში. ისინი მრავალჯერ ამარტივებენ ტურნირების ორგანიზებისა და ჩატარების პროცესს და საშუალებას აძლევენ სპორტულ ფედერაციებს, მოახდინონ სპორტსმენებისა და თანამშრომლების მართვა და სასურველი ინფორმაციის ორგანიზებულად განთავსება და გავრცელება.

ნაშრომში გთავაზობთ ახალ მიდგომას სპორტის მენეჯმენტ სისტემებში ძიუდოს მაგალითზე, რომელიც საშუალებას იძლება ეროვნულ დონეზე ჩატარებულ შეჯიბრებზე შეჯიბრისას საინფორმაციო ტაბლოს მართვასა და ინფორმაციული სახით ჩვენებას. ის ასევე აერთიანებს საქართველოში მოასპარეზე სპორტსმენებისა და ორგანიზებულ ტურნირების ბაზას და სამართავ პანელს. აპლიკაცია შეიცავს ალგორითმს, რომელიც ატარებს ტურნირზე წარდგენილი სპორტსმენების შეწყვილების ცხრილის შედგენა, მოდიფიკაციას, ასევე მესამე ადგილის შეწყვილების ალგორითმს.

პროგრამის მოხმარება შეუძლია არაავტორიზებულ მომხმარებელს საინფორმაციო მიზნებით და ავტორიზებულ მომხმარებელს სხვადასხვა დონეზე სამართავად. აპლიკაცია გამიზნულია რეალურ დროში ტრანზაქციის რეჟიმზე, აქედან გამომდინარე ნებისმიერ მომხმარებელს შეუძლია თვალი ადევნოს ტურნირის მსვლელობას დისტანციურად ასევე ტურნირზე დამსწრე მაყურებელს ტაბლოს სახით.

Abstract - In today's sports world software is widely applied in management and administrative issues. They simplify and fasten the process of organizing and holding a competition while at the same time helping sports' federations to manage sportsmen and staff members, to track, spread and allocate desirable information in a coherent order.

In this thesis, an entirely new approach is offered regarding sports management systems using a judo example. This is mainly to manage information screen by administration and to give information through the screen display to spectators during the contest on a national level. Moreover, the latter forms a database by uniting all the players and held tournaments in Georgia as well as a managing it's panel. An application includes an algorithm that can make a list of paired contestants and afterwards you are given a chance to modify the timetable. In addition to this, the algorithm can match participants to compete for the 3rd position.

This program can be used by a non-authorized person for informational purposes and by an authorized person for governing occasions on several stages. The application is meant to be working in a real time state, which directly reflects into exchanging information asynchronously from administrator to a spectator. This enables every person to follow the news whether they attend or remotely use a display screen.

გასაღები სიტყვები: სპორტის მენეჯმენტ სისტემა ძიუდოს მაგალითზე, ეროვნულ დონეზე ჩატარებული შეჯიბრებები, რეალურ დროში ტრანზაქციის რეჟიმი.

Keywords: Sports management system using a judo example, contest on a national level, exchanging information asynchronously.

სარჩევი

1. შესავალი

თანამედროვე ცხოვრებაში კომპიუტერული ტექნოლოგიების განვითარებასთან ერთად ბევრი მაგალითი გვხვდება იმისა, თუ როგორ გამოიყენება სხვადასხვა პროგრამული სისტემები სპორტის მენეჯმენტში. ეს უკანასკნელი ამცირებს მიწოდების დროს, აუმჯობესებს ხარისხს და ამავდროულად საგრძნობლად ამცირებს განვითარების ხარჯებს. როგოც კვლევები გვიჩვენებს, აპლიკაციები ხშირ შემთხვევაში სხვადასხვა იდეას ან პლატფორმას ეფუძნება. მათი დანიშნულება და სფერო შეუზღუდავია. იგი გვხდება როგორც ფინანსური საკითხების განხრით, ისე იურიდიულის, ადამიანის მენეჯმენტის, საგანმანათლებლო ან სხვა. მას შეიძლება ჰქონდეს ფუნქციური დატვირთვა ან იყოს მხოლოდ ინფორმაციის მიმოცვლის საშუალება. ფუნქციურ დატვირთვასთან ერთად განსხვავდება მათი ფუნქციონალური მხარეც. უმეტეს შემთხვევაში პროგრამული უზრუნველყოფები გამოირჩევა რთული და კომპლექსური მოხმარებით. თუმცა, სპორტის მენეჯმენტის პროგრამული სისტემები შედარებით იოლი მოსახმარია, მისი მიზნებიდან გამომდინარე. მიზნების ჩამონათვალი კი შეიძლება მრავალფეროვანი იყოს. მაგალითად, ძირითადად აქტუალურია სპორტსმენებისა და მომსახურე პერსონალის მართვის, განრიგებისა და ტურნირების შედგენის აპლიკაციები. ასევე სპეციალური ვარჯიშების შედგენისა და კვების რაციონის დასარეგულირებელი პროგრამები.

რაც შეეხება ტექნოლოგიების გამოყენებას, მისი რიცხვიც შეიძლება დიდი აღმოჩნდეს. უწინდელისგან განსხვავებით, დღესდღეობით აქტუალურია ვებ და მობილური აპლიკაციები, რამდენიმე წლის წინ კი desktop პროგრამებს ალტერნატივა არ ჰქონდათ. როგორც სხვა ტიპის პროგრამული უზრუნველყოფების შექმნისას შესაძლო გამოსაყენებელი ტექნოლოგიები აქაც ანალოგიურია. პროგრამული ენებიდან ყველაზე დიდი პოპულარობით C#, Java, Php და Pyton სარგებლობენ. ჩვენს შემთხვევაშიც არჩეული იქნა მათგან ერთ-ერთი - პროგრამული ენა C# რომლის იმპლემენტაციაც Microsoft .NET ტექნოლოგიით მოხდა. .NET ტექნოლოგია თავისი მოქნილობისა და შესაძლებლობებიდან გამომდინარე დიდი მოწონებით სარგებლობს თანამედროვე პროგრამირებაში. შესაბამისად, იგი სწრაფად განვითარებადი პროდუქტია და Microsoft მუდმივად გვთავაზობს სიახლეებს ამ მიმართულებით.

რატომ გამოიყენება პროგრამული სისტემები სპორტის მენეჯმენტში? თანამედროვე აპლიკაციები ეხმარება სხვადასხვა ფედერაციასა და ორგანიზაციას ადამიანური და მატერიალური რესურსების უკეთ მართვასა და გამოყენებაში. მაგალითად, რომელიმე კონკრეტული საქმის გასაკეთებლად ნაკლები დრო და ნაკლები მუშახელია საჭირო. ასევე რადგანაც სისტემა სრულიად აუტომატიზირებულია პერსონალის მხრიდან კონტროლი და დაკვირვება შემცირებულია. გარდა ამისა, ადამიანური რესურსის გამოყენებისას ხშირად ხდება შეცდომები, რომელიც გამოწვეულია ინდივიდის ფსიქოლოგიური თავისებურებებითა და სოციალურ- ეკონომიკური მდგომარეობით. რა თქმა უნდა, კომპიუტერული პროგრამის შემთხვევაშიც ხდება გარკვეული შეუთავსებობები, მაგრამ გარკვეული ფუნქციების დახმარებით დღესდღეობით აპლიკაციებს შეუძლიათ იპოვონ პრობლემა და შეატყობინონ პასუხისმგებელ პირებს.

2. პროგრამული სისტემები სპორტის მენეჯმენტი

პროგრამული სისტემები ჩვენს გარშემო მრავლად არის წარმოდგენილი. სპორტის მენეჯმენტში იგი აქტიურად გამოიყენება და დღეს დღეისობით იგი მისი განუყოფელი ნაწილია.

წარმოგიდგენთ ერთ-ერთ ასეთ მაგალითს, როცა კოლეჯის სპორტული საინფორმაციო მენეჯმენტი მჭიდროდ არის დაკავშირებული სასწავლო პროცესთან. ამ შემთხვევაში პროგრამული უზრუნველყოფა ხელმეორედ გამოიყენება(Software reuse). ეს არის პროცესი, როდესაც აპლიკაცია უკვე არსებულ პროგრამულ კომპონენტებს იყენებს. სისტემა ეფექტურად აკავშირებს სხვადასხვა დეპარტამენტს, იძლევა გარანტიას რომ სტუდენტის ინფორმაცია იქნება დაცული და მთლიანი. ამავდროულად აპლიკაციის დახმარებით, სკოლა არის სრულიად ინფორმირებული. მისი დახმარებით მასწავლებლებს სწრაფად და მოსახერხებლად შეუძლიათ სტუდენტის ინფორმაციის გახსნა. სისტემა არის ძალიან პროდუქტიული, საჭიროებს ნაკლებ მუშახელს და მატერიალურ რესურსებს. აგრეთვე, ამცირებს სწავლის პროცესთან დაკავშირებულ ხარჯებს. ეს მოდელი განსაკუთრებით მნიშვნელოვანია ინფორმაციული მენეჯმენტის სისტემის განვითარებისათვის.

არსებობს პროგრამული სისტემები, რომლებიც სპორტულ ორგანიზაციას ეხმარება აკონტროლოს ბიუჯეტი და ამავდროულად ქმნის სპორტული თამაშების განრიგს. პროექტები ეხმარება დეპარტამენტს, მართოს ხალხი, აკონტროლოს და განაცხადოს ინფორმაცია მონაცემთა ბაზის მეშვეობით. შედეგად, ხდება კალმისა და ფურცლის ჩანაცვლება. ასეთი სისტემის მეშვეობით პერსონალს შეუძლია კლასიფიცირებული ინფორმაციის უკეთესად კონტროლი ყოველდღიურ საქმიანობაში.

არსებობს აპლიკაციები საინფორმაციო სპორტული მენეჯმენტის პლატფორმისათვის, რომელიც B/S არქიტექტურაზეა დაფუძნებული. რესურსების მენეჯმენტი ყველა სახის ინფორმაციას სპორტული ღონისძიებებთან დაკავშირებით ასახავს გრაფიკულად, კომპიუტერულად აკეთებს სამეცნიერო ანალიზს და აგენერირებს სტატისტიკურ ანგარიშებსა და სტატისტიკურ სქემებს. აგრეთვე, ამავე მოდულში შესაძლებელია ინტერნეტის გამოყენება ნებისმიერი სახის პორტული რესურსების ატრიბუტული ინფორმაციის ჩასაწერად.

2.1. TeamSnap

TeamSnap 21-ე საუკუნის ერთ-ერთი ყველაზე პოპულარული პროგრამაა სპორტულ დარგში. ათი მილიონი მწვრთნელი, ადმინისტრატორები, მოთამაშეები და ასევე მშობლები იყენებენ TeamSnap-ის ვებ და მობილურ აპლიკაციებს. პირველ რიგში ისინი რეგისტრირდებიან, იყენებენ აპლიკაციას როგორც საკომუნიკაციო საშუალებას. გეგმავენ და კოორდინირებენ ყველაფერს გუნდის, კლუბისა და მთლიანი სეზონისათვის.

2.2. TeamUnify

TeamUnify ძალიან ცნობილი გუნდის მენეჯმენტის პროგრამული სისტემის პროვაიდერია. შექმნილია ინოვაციური ოჯახი ვებისა და მობილური პროგრამული პროდუქტების, რომლებიც რევოლუციური გზით მართავენ კონკურენტუნარიან ცურვის გუნდებს. მათში გაერთიანებულია ოთხი ათასზე მეტი მოცურავეების ჯგუფები, დაახლოებით ერთი მილიონი აქტიური ადმინისტრატორი, მშობლები და ცურვის მოყვარულები მსოფლოს ყველა კუთხიდან. ყოველი გამოწერა შეიცავს შვიდ დღიან მხარდაჭერას და ულიმიტო განახლებას მუდმივად მზარდი და განვითარებადი სპორტული გუნდის მენეჯმენტის პროგრამიდან.

2.3. Manage Your League

Manage Your League უზრუნველყოფს ვებზე დაფუძნებულ რეგისტრაციას, ვებსაიტს, განრიგის შექმნას, მსაჯის მენეჯმენტს. League-ს ვებსაიტზე არ არის განთავსებული შემაწუხებელი რეკლამები, რაც მობილურ აპლიკაციას ძალიან მარტივად და სწრაფად გამოსაყენებელს ხდის. მათი სარეგისტრაციო მენეჯმენტ სისტემა ყველაზე სრულყოფილი და მორგებულია მარკეტზე. ამას გარდა, პროგრამას შეუძლია მსაჯებისა და არბიტრების განრიგის შექმნაც. ასევე ეს უკანასკნელი გადასახადების სისტემასაც შეიცავს.

2.4. **Самботур**

Самботур არის რუსული Desktop აპლიკაცია, რომელიც გამოიყენება სამბოს ტურნირების ჩატარებისას. იგი იძლევა საშუალებას, რომ ადმინისტრატორმა მართოს, ხოლო დამსწრე მაყურებელმა საინფორმაციო ეკრანის მეშვეობით თვალი ადევნოს ჭიდაობის მიმდინარე ქულებს. აღსანიშნავია, რომ აღნიშნული აპლიკაცია ძალიან გავს ჩვენს მიერ შემუშავებულ სისტემის ტაბლოს მოდულს, რადგანაც სამბო და ძიუდო ერთი სტილის სპორტის სახეობებია. მიუხედავად ამისა, Самботур ბევრად პრიმიტიულია და გამოიყენება ერთჯერადად. ამასთანავე ტექნილოგიები, რომლებსაც ის იყენებს მოძველებულია და განახლებას საჭიროებს. ვიზუალური მხარე კი, არ ერგება ყველა ტიპის და ზომის მონიტორს. აქვე უნდა ითქვას, რომ აღნიშნული აპლიკაციის ინტერფეისის ენის შეცვლილ ვერსიას იყენებს საქართველოს სამბოს ფედერაცია, რაც აპლიკაციას კიდევ ერთ ნაკლოვანებას სძენს - იგი არ არის მორგებული ქართულ წესებზე და ამასთან ჩამორჩება საერთაშორისო წესების ცვლილებებსაც.

3. პროგრამულ-ტექნიკური ინსტრუმენტები მენეჯმენტის სისტემების შემუშავებაში

3.1. მონაცემთა ბაზა

პლათფორმა ამუშავებს და ცვლის მონაცემებს MS SQL ბაზაში, რისი მენეჯმენტიც ხორციელდება როგორც MS SQL Management Studio-დან, ისე პროგრამული უზრუნველყოფის დატა-ლეიერიდან.

პროგრამის დაგეგმვის და შემუშავების პირველი ეტაპი სწორედ ამ უზრუნველყოფის მომხმარებლების და მათი ინტერესების შესწავლა და შესაბამისად, საჭირო არსების და მათ შორის არსებული კავშირების დაგეგმვა იყო.

ჩვენი პორტალის მონაცემთა ბაზა აერთიანებს მაქსიმალურად განტვირთული სუფთა დომენური მოდელების შესაბამის არსებს, მათთან დაკავშირებულ ატრიბუტებს და რელაციურ კავშირებს.

ელექტრონული სწავლების პორტალის უმთავრეს არსებს წარმოადგენენ მომხმარებლები, რომლებიც იყოფიან არაავტორიზებულ მომხმარებლებად, ავტორიზებულ მომხმარებლებად, ადმინისტრატორებად და სუპერ-ადმინისტრატორებად, აგრეთვე სპორტსმენი, ტურნირი, შეჯიბრი.

3.2. არქიტექტურა

მონაცემების სპეციფიკისა და დომენური სიმრავლის გააზრების შემდეგ, დაიგეგმა პროგრამული უზრუნველყოფის არქიტექტურა. მრავალი სახის მიდგომებიდან ამორჩა პორტალის დანიშნულებისთვის ყველაზე ოპტიმალური- SOA- Service Oriented Architecture, რადგან ჩვენი პლატფორმა წარმოადგენს მომხმარებლებისთვის მოქნილ და დახვეწილ სერვისს, რომელიც იყოფა მნიშვნელოვან ქვესერვისებად. შესაბამისად, გვაქვს ზოგადი ანგარიშის, მომხმარებლის, ადმინისტრატორის, სუპერ ადმინისტრატორის, სპორტსმენის, ტურნირის, შეჯიბრის და სხვა დამხმარე და საზიარო სერვისები. პროგრამული უზრუნველყოფა, როგორც მთლიანი, ინტერგირებადი და გაფართოებადი სოლუშენი შევქმენით პროგრამული ინჟინერიის წამყვანი მიდგომების გათვალისწინებით. ამავე დროს, სოლუშენში გამოვყავით კონკრეტული პასუხისმგებლობისა და ერთმანეთთან განტვირთული ურთიერთკავშირის მქონე ლეიერები:

- Domain Layer - C# Library. დომენურ და ვიზუალურ მოდელთა სიმრავლე, რომელსაც რეფერენსით ხედავს ყველა ქვედა ფენა. მთავარი მსხვილი ფოლდერებია User, Role, Sportsman, Gender, Region, Tournament, Wrestle, TournamentSporstman, TournamentWrestle.
- Common Layer - C# Library. საზიარო კლასთა და მიკროსერვისთა სიმრავლე, რომელსაც რეფერენსით ხედავს ყველა ქვედა ფენა.
- Core Layer – C# Librabry. საზიარო კლასთა და მიკროსერვისთა სიმრავლე, რომლებიც წარმოადგენენ ბირთ სერვისებს აპლიკაციისთვის და რომელსაც რეფერენსით ხედავს ყველა ქვედა ფენა. ჩვენს შემთხვევაში აღნიშნული ლეიერი აერთიანებს Ninject სერვისებს.
- Data Access Layer - C# Library. მონაცემთა ბაზასთან ურთიერთობისთვის განკუთვნილი ფენა, ბიბლიოთეკა, რომელიც აერთიანებს დატა-რეპოზიტორებს დომენური მოდელების და მათი რელაციური კავშირების CRUD ოპერაციების იმპლემენტაციას. აღსანიშნავია, დატა ლეიერი მხოლოდ მონაცემთა გაცვლასა და CRUD ოპერაციებზეა პასუხისმგებელი და პირდაპირი რეფერენსი Domain-თან აქვს. მსხვილი რეპოზიტორებია: GenericRepository, RegionRepository,

SportsmanRepository, TournamentRepository, WrestleRepository, TournamentSportsmanRepository, TournamentWrestleRepository, RoleRepository, UserRepository, UserRoleRepository.

- Service layer - SOA Core, Business Logics layer - C# Library. წარმოადგენს პროგრამული უზრუნველყოფის გულს, რომლის დანიშნულება მიღებული და გადასაცემი მონაცემების სპეციფიური ბიზნეს ლოგიკით დამუშავებაა, რაც SOA პრინციპით ხორციელდება მთავარი სერვისების გამოყენებით (GenericBll, RegionBll, SportsmanBll, TournamentBll, WrestleBll, TournamentSportsmanBll, TournamentWrestleBll, RoleBll, UserBll, UserRoleBll…) სერვისების ფენა პასუხისმგებელია, რომ მომხმარებლისგან მიღებული მონაცემები სწორად დაამუშავოს და გადასცეს დატა-ლეიერს, აგრეთვე დატა-ლეირიდან დაბრუნებული მონაცემები კორექტული ბიზნეს ლოგიკით გადმოსცეს მომხმარებელს. სერვისების ფენას პირდაპირი კავშირი აქვს მონაცემთა და საზიარო ფენებთან.
- Client layer – Web layer - ASP.NET MVC Project. მომხმარებლის ინტერფეისის მხარეა, რომელიც პასუხისმგებელია მომხმარებლის დაცულობაზე, მონაცემებზე წვდომის კონტროლზე, მონაცემთა ვალიდური სახით სერვისებისთვის გადაცემაზე და სერვისით დაბრუნებულ მონაცემების ვებ-ჰელფერების დამუშავებით ვიუმოდელებში გამოტანაზე. ვებ ლეიერს კავშირი აქვს სერვისებისა და საზიარო ფენსთან. მომხმარებლის მხარისთვის მნიშვნელოვანია უსაფრთხოების, ინიციალიზების, კონფიგურაციისა და ვებ(ვიუ) მოდელების ჰელფერები.

3.3. ტექნოლოგიები, ფრეიმვორქები

პროგრამული უზრუნველყოფა წარმოადგენს .NET Framework-ზე შექმნილი პროექტების ერთობლიობას - რეფერენსებით შეკრულ ბიბლიოთეკების ერთობლიობას, როგორც სოლუშენს, სადაც აღწერილი არქიტექტურული ლეიერები წარმოადგენს ჩვენს მიერ დაწერილ C# ბიბლიოთეკებს, ხოლო მომხმარებლის მხარე არის ASP.NET MVC პროექტი. შესაბამისად, სერვერის მხარის პროგრამირება ხორციელდება ობიექტზე ორიენტირებული ენის- C#- ის საშუალებით .NET გარემოში, ხოლო კლიენტის მხარეს საჭირო ფუნქციონალს ასრულებს Pure Javascript, jQuery, AJAX - ვიზუალური მხარის ჩვენებას კი HTML, CSS, Razor Engine. ვიზუალური მხარის სხვადასხვა ზომის ეკრანზე ეფექტურად მორგებისთვის კი გამოყენებულია Bootstrap. მომხმარებლის მხარეს გამოყენებულია ვალიდაციისათვის ASP.Net Remote Validation, ხოლო ბაზათან ურთიერთობისთვის საკმაოდ სწრაფი და ოტიმიზირებული, ასევე აპრობირებული და Microsoft-ისგან რეკომენდირებული ფრეიმვორქი- ASP.NET Entity Framework.

- ASP.NET MVC არის Microsoft- ის მიერ შემუშავებული ვებ-პროგრამის ფრეიმვორკი, რომელიც ახდენს Model - View- Controller (MVC) ფრეიმვორქის იმპლემენტაციას. MVC მოდელი განისაზღვრება 3 ლოგიკური ლეიერით: Model (ბიზნეს ლეიერი); View (საჩვენებელი ლეიერი); Controller (მაკონტროლებელი ლეიერი). მოდელი წარმოადგენს აპლიკაციის კონკრეტული ასპექტის მდგომარეობას. კონტროლერი ახორციელებს ლოგიკურ მოქმედებებს და მათ საფუძველზე აახლებს მოდელს, შემდეგ კი ინფორმაციის აწვდის view-ს. view იღებს საჭირო ინფორმაციას კონტროლერისგან და აწვდის მომხმარებლის ინტერფეისს, რომ ეს ინფორმაცია გამოაქვეყნოს. აღნიშნული ლეიერები ფუნქციონირებენ დამოუკიდებლად, რაც საშუალებას გვაძლევს, რომ მათი ტესტირება იყოს ინდეფერენტული.
- ASP.NET Entity Framework არის ADO.NET- ის ტექნოლოგიების კომპლექტი, რომელიც მხარს უჭერს მონაცემებზე ორიენტირებული პროგრამების განვითარებას. მონაცემებზე ორიენტირებული აპლიკაციების არქიტექტორებსა და დეველოპერებს ჩვეულებრივ ორი განსხვავებული ამოცანის

გადაჭრა სჭირდებათ. მათ უნდა შექმნან შესაბამისი მოდელები არსებისთვის, კავშირებისთვისა და ბიზნესის პრობლემებისთვის და მათ ასევე უნდა იმუშაონ იმ მონაცემთა შესანახ სისტემებზე, რომლებიც გამოიყენება მონაცემების მისაღებად და შესანახად, ამავდროულად მონაცემები შეიძლება ზიარდებოდეს მრავალრიცხოვან შენახვის სისტემებში; Entity Framework საშუალებას აძლევს დეველოპერებს მონაცემთან იმუშაონ დომენური ობიექტებისა და თვისებების საშუალებით ისე რომ არ ქონდეთ ურთიერთობა მონაცემთა ბაზების ცხრილებსა და სვეტებთან, სადაც ეს მონაცემები ინახება. მონაცემებთან ურთიერთობისას Entity Framework -ის გამოყენებით დეველოპერებს შეუძლიათ იმუშაონ შედარებით მაღალი დონის აბსტრაქციასთან და შექმნან მონაცემებზე ორიენტირებული აპლიკაციები ნაკლები კოდის გამოყენებით, ვიდრე ტრადიციულ პროგრამებში.

- HTML - HyperText Markup Language არის ჰიპერტექსტური აღწერის ენა, რომელიც განკუთვნილია ვებ-გვერდების და საიტების შესაქმნელად და ინფორმაციის გასავრცელებლად ინტერნეტის საშუალებით. HTML ელემენტები წარმოადგენენ ყველა საიტის შემადგენელ ნაწილს. HTML-ის მეშვეობით დასაშვებია სურათებისა და ობიექტების გვერდზე ასახვა, ისევე, როგორც ინტერაქტიული ფორმების შექმნა. შესაძლებელია სტრუქტურული დოკუმენტების შექმნა, რაც ხდება ტექსტის სტრუქტურული სემანტიკის აღნიშვნით. მაგალითად, აბზაცებით, სიებით, სათაურებით, ბმულებით, ციტირებებით და სხვა ელემენტებით.
- Razor - არის ASP.NET პროგრამირების სინტაქსი, რომელიც გამოიყენება დინამიური ვებ გვერდების შესაქმნელად C# ის გამოყენებით.
- Bootstrap - არის უფასო ფრონტი-ენდ ფრეიმვორქი ვებსაიტებისა და ვებ პროგრამების დიზაინის შესაქმნელად. იგი შეიცავს HTML-სა და CSS-ზე დაფუძნებულ ინტერფეისის კომპონენტებს, როგორებიცაა: ტიპოგრაფია, ფორმები, ღილაკები, ნავიგაცია და სხვა.
- JavaScript — პროგრამირების ერთ-ერთი ფართოდ გავრცელებული ენაა. იგი შეიქმნა ორგანიზაციაში Netscape Communications Corporations, Brendan Eich-ის მიერ, HTML გვერდებისათვის დინამიური ფუნქციის დამატების მიზნით. ამ ორგანიზაციის ერთ-ერთ პროდუქტს წარმოადგენს XX საუკუნის 90-იან წლებში არსებული საკმაოდ პოპულარული ბრაუზერი Netscape Navigator, რომელმაც ვერ გაუძლო ბრაუზერების ომს და დღეისათვის თითქმის არ გამოიყენება.
- jQuery — JavaScript ბიბლიოთეკაა, რომელიც ფოკუსირდება JavaScript-ის და HTML-ის ურთიერდამოკიდებულებაზე. jQuery ბიბლიოთეკა აადვილებს ნებისმიერ DOM-ის ელემენტთან დაშვებას, მიმართვას DOM ელემენტის ატრიბუტებთან და შინაარსთან, მათით მანიპულირებას. აგრეთვე, jQuery ბიბლიოთეკა წარმოადგენს მოსახერხებელ API-ის Ajax-თან სამუშაოდ.
- Ajax — არ არის პროგრამირების ენა, მაგრამ ის არის უკეთესი, უფრო სწრაფი და ინტერაქტიული ვებ პროგრამების შექმნის ტექნოლოგია. Ajax-ის დახმარებით JavaScript-ს შეუძლია დაუკავშირდეს სერვერს XMLHttpRequest ობიექტის გამოყენებით. ამ ობიექტის დახმარებით, JavaScript-ს შეუძლია ივაჭროს ვებ სერვერით, გვერდის განახლების გარეშე. Ajax იყენებს ასინქრონულ მონაცემთა გადაცემას ბრაუზერსა და ვებ სერვერს შორის და საშუალებას აძლევს ვებ გვერდს სერვერიდან მოითხოვოს ინფორმაციის მხოლოდ ის ბიტები, რომლის მიმოხილვაც სურს მომხმარებელს. Ajax ტექნოლოგია ინტერნეტ პროგრამებს აქცევს უფრო პატარა, სწრაფ და მომხმარებელზე მორგებულ პროგრამებად.

3.4. პატერნები, პროგრამირების აპრობირებული პრინციპები

ჩვენი გუნდისთვის მნიშვნელოვანი არა მხოლოდ შესრულებადი და კომპილირებადი კოდი, არამედ მაღალ დონეზე გააზრებული ინჟინერია და მარტივად გაფართოებადი და მოქნილი კოდია, სწორედ ამ მიზნით გამოვიყენეთ აპრობირებული მიდგომები კოდის და კლასების ორგანიზებისას, როგორიცაა :

IOC – Inversion of control by Dependency Injecton - კლასებსა და მეთოდებს შორის პასუხისმგებლობათა განტვირთვისთვის. შესაბამისი კონტეინერი კონფიგურირდება და ინიციალიზდება ვებ-პროექტის გაშვებისას. ჩვენს შემთხვევაში გამოყენებული იქნება Ninject.

SOA Principles - იმისთვის, რომ ფუნქციონალი იყოს დამოუკიდებელი გამომძახებელი მხარისგან და მოქნილი

Repository Pattern - დომენური მოდელების კლასების განტვირთვა ბაზასთან ურთიერთობის მეთოდებისგან.

დაცულია **OOP** საფუძვლები და პროგრამირების კარგი სტილის ისეთი პრინციპები, როგორიცაა **SOLID (Single responsibility, Open/closed principle, Liskov substitution, Interface segregation, Dependency inversion)**, **DRY**-Do not repeat yourself, **KISS**-Keep it short and simple, **YAGNI**- You are not gonna need it.

- SOLID: S-Single responisibility - დაწყებული მარტივი მეთოდით დასრულებული არქიტექტურული ლეირით, ყველაფერს აქვს თავისი უნიკალური დატვირთვა და გამიჯნული პასუხისმგებლობა, O-Open/Closed principle- კლასები ღიია გაფართოებისთვის, მაგრამ წვდომა მეთოდებზე დაცულია, L-Liskov substitution - მშობელი და მემკვიდრე კლასების ოპტიმალური გამოყენება, I-interface segregation - პროგრამული მომსახურე კლასები ასრულებს შესაბამის ინტერფეისებს, რაც საშუალებას იძლევა გამოვიყენოთ პროგრამული ინჟინერიის უმნიშვნელოვანესი ასპექტები - დაბალი ურთიერთკავშირი და შეჯაჭვულობა კლასებს შორის IOC მიდგომით. ასევე ინტერფეისები გვეხმარება სამომავლო Unit testing- ის მარტივად განხორციელებაში. D-Dependency inversion - განხილული კლასებსა და მეთოდებს შორის დამოკიდებულების განტვირთვა და პასუხიმგებლობათა სწორი გადანაწილება.
- DRY: Don’t repeat yourself - საერთო მეთოდები, კლასები, ვალიდატორები გატანილია ცალკე და გამოიყენება საჭიროებისდა მიხედით მრავალ ადგილას კორექტულად, გარდა ამისა, დაცულია პასუხისმგებლობათა მკაცრი გადანაწილება როგორც მეთოდებში, ასევე არქიტექტურული ფენების დონეზე.
- KISS: keep it short and simple. რადგან ვცდილობთ არ გავიმეოროთ ერთი და იგივე კოდი და კლასები, ვიყენებთ მემკვიდრეობითობას და მეთოდების პასუხისმგებლობების მიხედვით გამიჯვნას.
- YAGNI: You are not gonna need it- ვიყენებთ ზომიერად ზოგად კლასებს და მეთოდებს. ვპასუხობთ კონკრეტულ მოთხოვნებს ოპტიმალური გადაწყვეტილებებით და განტვირთული პასუხისმგებლობებით.

გათვალისწინებულია გუნდურ მუშაობაზე მორგებული გაიდლაინები არქიტექტურული პატერნების პასუხისმგებლობათა მკაცრი დაცვის, მეთოდებს შორის დაბალი ურთიერთკავშირისა და დამოკიდებულებების განტვირთვის გათვალისწინებით.

პროგრამაში მაღალ დონეზეა ასევე გათვალისწინებული კიბერ-უსაფრთხოების პრინციპები.

3.5. დამხმარე ბიბლიოთეკები

SignalR-ი არის ASP.NET უახლესი ბიბლიოთეკა. რეალურ დროში მომუშავე FrameWork-ი, რომელიც აძლევს საშუალებას სერვერს გამოიყენოს მეთოდები კლიენტებზე. რა არის რეალურ დროში ტრანზაქცია? ამ კომუნიკაციის დახმარებით მომხმარებელს აქვს საშუალება, მყისიერად დაუკავშირდეს სერვერსა და მასზე განთავსებულ ინფორმაციას, მიიღოს ან გააგზავნოს იგი. ამ ტექნოლოგიის გამოყენების გარეშე კლიენტს არ აქვს სერვერთან დაუყოვნებლივ დაკავშირების საშუალება და იძულებულია ელოდის მას, როდის გამოუყოფს მისთვის სასურველ ინფორმაციას. რა თქმა უნდა, ეს უკანასკნელი უზრუნველყოფს ორმხრივ კავშირს კლიენტსა და სერვერულ მხარეს შორის. SignalR-ის ულიმიტოდ გამოყენება შესაძლებელია იმ აპლიკაციაში, რომელიც იყენებს JavaScript-ს ან .Net Framework-ს. მოკლედ რომ ვთქვათ, SignalR- არის ახალი ბიბლიოთეკა .NET -სთვის, რომელიც შექმნილია ASP.NET აპლიკაციების რეალურ დროში სამართავად. შესაბამისად, მყარდება წმინდა კავშირი კლიენტსა და სერვერს შორის. უამრავი პლატფორმისთვის არსებობს SignalR-ის კლიენტები და თითოეული კლიენტური მხარის მეთოდისათვის არსებობს სერვერული მეთოდიც, რაც ხელს უწყობს მის მრავალფეროვნებას.

Kendo UI არის ყოვლისმომცველი ფრეიმვორქი, რომელიც შეიცავს 70+ UI- ს ბიბლიოთეკის ვიჯეტს მონაცემთა ვიზუალიზაციის უამრავ გაჯეტს, კლიენტის მხარის მონაცემთა წყაროსა და ჩაშენებულ MVVM (Model-View-ViewModel) ბიბლიოთეკას. ის უზრუნველყოფს AngularJS და Bootstrap ინტეგრაციას და ასევე ნაწილდება პროდუქტთა რამდენიმე ერთეულად, რათა მომხმარებელმა შეძლოს დამოკიდებული პროდუქტის არჩევა პროექტის მოთხოვნებიდან გამომდინარე. Kendo UI- ი მოიცავს საწარმოთა კლასის ხაზის ბიზნეს-პროგრამებისთვის განკუთვნილ ვიჯეტებს და განკუთვნილია პროფესიონალური საიტების შექმნისათვის, რომელებიც საჭიროებს დროდადრო საჭიროებს ტექნიკურ მხარდაჭერას.

Microsoft Azure (ყოფილი Windows Azure) — Microsoft-ის ღრუბლოვანი სერვისების პლატფორმა, რომლის მეშვეობითაც შესაძლებელია კომპანიის "ღრუბლოვან" დატაცენტრებში აპლიკაციების განთავსება და ვირტუალურად მაშტაბირება. Windows Azure ანხორციელებს Platform as service (PaaS) მოდელს, როდესაც პლატფორმა კლიენტისთვის წარმოდგენილია როგორც სერვისი. Windows Azure წარმოადგენს აპლიკაციების შემუშავებისა და შესრულების შესაძლებლობას, ასევე ის ითავსებს მონაცემთა შენახვას სერვერებზე, რომლებიც განთავსებულია გარკვეულ დატაცენტრებში. პლატფორმა, როგორც სერვისი ასევე ითვალისწინებს ინფრასტრუქტურას როგორც სერვისს (Infrastructure as a Service, IaaS), რომლის შესაძლებლობებმა დიდი ცვლილებები განიცადეს 2012 წლის 7 ივნისის შემდეგ. Windows Azure სრულიად ახდენს 2 "ღრუბლოვანი" მოდელის რეალიზებას- პლატფორმას როგორც სერვიისს (PaaS) და ინფრასტრუქტურას როგორც სერვისს (IaaS). Windows Azure-ის პლატფორმის ქმედუნარიანობას უზრუნველყოფენ Microsoft-ის 8 გლობალური დატაცენტრი. საჯარო ღრუბლის შემოთავაზებაში კლიენტი იხდის მხოლოდ რესურსების და სიმძლავრის საფასურს, რომლებიც ამოქმედებულია აპლიკაციაში (პროგრამაში). კლიენტი ასევე დამატებით იხდის აპლიკაციის გამოყენების ფაქტობრივი დროის საფასურს. აღნიშნული მოდელის ძირითადი განსხვავებებია:

- მხოლოდ გამოყენებული რესურსების საფასურის გადახდა;
- საერთო, მრავალნაკადიანი გამოთვლის სტრუქტურა;
- ინფრასტრუქტურისგან აბსტრაქცია.

4. ძიუდოს სპორტული მენეჯმენტის პროგრამული უზრუნველყოფის სისტემის ლოგიკური სქემა

4.1. ზოგადი მიმოხილვა

დღეს დღეისობით ყოველგვარი საქმიანობა დაფუძნებულია პროგრამული სისტემების გამოყენებაზე. საერთაშორისო მასშტაბით სპორტის მენეჯმენტში უამრავი განსხვავებული სტილის და ფუნქციონალის აპლიკაციები არსებობს. მიუხედავად ამისა, მოთხოვნებიდან გამომდინარე მათ ხშირი ცვლილება თუ გაუმჯობესება სჭირდებათ. სამწუხაროდ, საქართველო მსგავსი პროგრამული სისტემების სიმრავლით არ გამოირჩევა და ხშირად გვიწევს გამოვიყენოთ უცხო აპლიკაციები, რომლებიც ჩამორჩენილია თანამედროვეობას ან ზუსტად არ ერგება ჩვენს სპეციფიკურ მოთხოვნებს. სწორედ ამ სახის პრობლემის გადაწყვეტის გზას გვთავაზობს ჩვენს მიერ შექმნილი პროგრამული უზრუნველყოფა. მისი გამოყენება შეიძლება მრავალმხრივ საქართველოს ძიუდოს ფედერაციის მიერ ან სხვა კერძო სტრუქტურის მიერ, რომლის საქმიანობაც ძიუდოს ტურნირების ჩატარებას გულისხმობ.

აღნიშნული პროგრამული უზრუნველყოფა იყოფა რამდენიმე ლოგიკურ მოდულად ფუნქციონალის და მომხმარებლის გამოყენების უფლებების მიხედვით. სისტემაში არსებობს არაავტორიზებული და ავტორიზებული მომხმარებლები. არაავტორიზებულ მომხმარებელს პროგრამის მოხმარება შეუძლია საინფორმაციო მიზნებით და ავტორიზებულ მომხმარებელს სხვადასხვა დონეზე მის სამართავად. არაავტორიზებული მომხმარებელი ეცნობა ინფორმაციას სპორტსმენების და ტურნირების შესახებ, ასევე მას შეუძლია თვალყური ადევნოს ნებისმიერი შეჯიბრის ტაბლოს ინფორმაციას პირდაპირ რეჟიმში.

ავტორიზებული მომხმარებლები იყოფიან 3 განსხვავებული როლის მიხედვით User, Administrator და Super Administrator. User-ს აქვს ნებართვა და სპეციალური პანელი იმისთვის, რომ მართოს ტაბლოზე განთავსებული ინფორმაცია. Administrator-ს აქვს ნებართვა და დამატებით ფუნქციონალი, რომ აკონტროლოს და მართოს სისტემაში არსებული User-ები, ასევე რომ დაამატოს, შეცვალოს ან წაშალოს სპორტსმენის ან ტურნირის შესახებ ინფორმაცია და შექმნას ახალი ტურნირი, დაამატოს მასში სპორტსმენები და შექმნას ან მოდიფიკაცია გაუკეთოს შეწყვილების ცხრილს. Super Administrator-ს აქვს ნებართვა და ყველა ზემოთ ჩამოთვლილი ფუქციონალი და ამას დამატებით შეუძლია მართოს სისტემაში არსებული Administrator-ები.

4.2. ძირითადი ამოცანები და მოდულები

სისტემის ორგანიზება:

სისტემაში არსებობს მონაცემთა ბაზა, სადაც შენახულია ინფორმაცია სპორტსმენებისა და ტურნირების შესახებ. სისტემის ადმინისტრატორს შეუძლია დაამატოს, შეცვალოს ან წაშალოს ამ სახის ნებისმიერი ინფორმაცია.

ტურნირის შედგენის ამოცანა:

საწყის საფეხურზე იქმნება ტურნირი, შემდგომ აღნიშნულ ტურნირს ემატება მონაწილეები. დამატებული მონაწილეების რაოდენობის და მათი რეიტინგის მიხედვით ჩვენს მიერ შექმნილი შეწყვილების ალგორითმის გამოყენებით იქმნება შეწყვილების ცხრილი, რის მიხედვითაც ტარდება ჭიდაობები.

ტაბლოს მართვა:

ტაბლოს მართვა დაფუძნებულია ძიუდოს შეჯიბრის პრინციპზე. შეჯიბრი მიმდინარეობს ორ სპორტსმენს შორის ძირითადი 4 წუთის განმავლობაში. თუ ძირითადი 4 წუთის ამოწურვის შემდგომ ვერ გამოვლინდა გამარჯვებული, ათვლას დაიწყებს დამატებითი დრო გამარჯვებულის გამოვლენამდე. გამარჯვება განისაზღვრება ორი ტიპის ქულათა სისტემით. 1. იპონი - მისი მნიშვნელობა შეიძლება იყოს 1 ან 0. თუ რომელიმე სპორტსმენი გააკეთებს იპონს შეჯიბრი სრულდება და გამარჯვებულად ცხადდება ზემოაღნიშნული მოჭიდავე. 2. ვაზარი - მისი მნიშვნელობა შეიძლება იყოს 0 ან ნებისმიერი ნატურალური რიცხვი. თითო ილეთის გაკეთება აისახება 1 ქულის მომატების სახით. გამარჯვებულია ის მონაწილე რომელიც მეტ ვაზარის ქულას დააგროვებს. ამას დამატებით ჯარიმის სახით თითოეულმა მონაწილემ შეიძლება მიიღოს მაქსიმუმ 3 გაფრთხილება. მესამე გაფრთხილებისას მონაწილე ეთიშება შეჯიბრს და გამარჯვებულად ცხადდება მისი ოპონენტი. მონაწილე შეიძლება ასევე დისკვალიფიცირდეს ტურნირის მიმდინარეობისას წონითი კატეგორიის დარღვევის გამო. ამ შემთხვევაშიც, გამარჯვებულად ცხადდება მისი ოპონენტი სპორტსმენი.

4.3. ფუნქციონალის დეტალური აღწერა

მთავარი გვერდი

აღწერა:

მთავარ გვერდს აქვს ვიზუალური დატვირთვა. მასზე განთავსებულია ანიმაციური სლაიდერი, რომელზეც გამოსახულია ძიუდოსთან დაკავშირებული სურათები.

უფლებები:

გვერდზე დაიშვებიან ავტორიზებული და არაავტორიზებული მომხმარებლები;

ხილვადობა თავსართში:

- SuperAdmin: მთავარი, ტურნირები, სპორტსმენები, რეგიონები, მომხმარებლები;
- Admin: მთავარი, ტურნირები, სპორტსმენები, რეგიონები, მომხმარებლები;
- User: მთავარი, ტურნირები, სპორტსმენები;
- არაავტორიზებული მომხმარებელი: მთავარი, ტურნირები, სპორტსმენები.

დამატებითი ინფორმაცია:

თავსართის ზემოთ განთავსებულია კონტეინერი რომელზეც გამოსახულია მომხმარებლის სახელი, მომხმარებლიდან გამოსვლა და პაროლის შეცვლა იმ შემთხვევაში, თუ მომხმარებელი ავტორიზებულია.

რეგიონების გვერდი

აღწერა:

რეგიონების გვერდზე განთავსებულია საქართველოში არსებული ყველა ქალაქის ჩამონათვალი, რომლებსაც შემდგომში სპორტსმენების სისტემაში დამატებისას მივუთითებთ.

ფუნქციონალი:

- რეგიონის დამატება;
- არსებული რეგიონის ჩასწორება;
- არსებული რეგიონის წაშლა.

უფლებები:

გვერდზე შესვლა და ფუნქციონალის გამოყენება შეუძლია მხოლოდ SuperAdmin-სა და Admin-ს.

ვიზუალიზაცია:

ინფორმაციის გამოსატანად და მათ დასამუშავებლად გამოყენებულია Telerik UI Grid Inline Editing - ის ფუნქციონალით.

ვალიდაცია:

Client Side ვალიდაციისთვის გამოყენებულია Remote Validation, რომელიც ამოწმებს რომ არ მოხდეს ქალაქის სახელის დუბლირება.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი (Add, Edit, Delete).

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- დასახელება: string Name

ყველა ველი სავალდებულოა

სპორტსმენების გვერდი

აღწერა:

სპორტსმენების გვერდზე განთავსებულია საქართველოში მოასპარეზე ყველა სპორტსმენის ჩამონათვალი, რომლებსაც შემდგომში ტურნირების სისტემაში სპორტსმენების დამატებისას მივუთითებთ. სიის თითოეული ელემენტი შეიცავს სპორტსმენის სურათს, პირად ნომერს, სახელს, გვარს, ასაკს, სქესს, რეგიონს, წონას, რეიტინგს. სია იყოფა გვერდებად და სიის ბოლოს განთავსებულია გვერდის ცვლილების შესაძლებლობა. სიის თავში დამატებულია ძებნის ფუნქციონალი. შესაძლებელია ფილტრის დადება (ასაკით, სქესით, წონით(დიაპაზონი), რეგიონით, რეიტინგით(დიაპაზონი).

ფუნქციონალი:

- სპორტსმენის დამატება;
- არსებული სპორტსმენის ჩასწორება;
- არსებული სპორტსმენის წაშლა.

უფლებები:

გვერდზე დაიშვებიან ავტორიზებული და არაავტორიზებული მომხმარებლები; ფუნქციონალის გამოყენება შეუძლია მხოლოდ SuperAdmin-სა და Admin-ს.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი (Add, Edit, Delete).

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- პირადი ნომერი: string PersonalNumber
- სახელი: string Name
- გვარი: string Surname
- წონა: double Weight
- რეიტინგი: int Rank
- რეგიონი: RegionId

ყველა ველი სავალდებულოა

ტურნირების გვერდი

აღწერა:

ტურნირების გვერდზე განთავსებულია საქართველოს ძიუდოს ფედერაციის მიერ საქართველოში ორგანიზებული ყველა ტურნირის ჩამონათვალი. ტაბების სახით გამოიყოფილია მიმდინარე, მომავალი და უკვე ჩატარებული ტურნირები. ადმინისტრატორს უნდა შეეძლოს ტურნირის მართვა (სპორტსმენების დამატება და შეწყვილების ცხრილის აგება), User-ს და არაავტორიზებულ მომხმარებელს მიმდინარე და მომავალი ტურნირის დეტალების(ინფორმაციის და შედგენილი შეწყვილების ცხრილის) დათვალიერება.

ფუნქციონალი:

- ტურნირის დამატება;
- არსებული ტურნირის ჩასწორება;
- არსებული ტურნირის წაშლა;
- არსებული ტურნირის მართვა;
- არსებული ტურნირის დეტალური ინფორმაციის ნახვა.

უფლებები:

გვერდზე დაიშვებიან ავტორიზებული და არაავტორიზებული მომხმარებლები;

- SuperAdmin: მიმდინარე (შესაძლებელია მოდიფიკაცია) , მომავალი (შესაძლებელია მოდიფიკაცია) და უკვე ჩატარებული ტურნირები;
- Admin: მიმდინარე (შესაძლებელია მოდიფიკაცია) , მომავალი (შესაძლებელია მოდიფიკაცია) და უკვე ჩატარებული ტურნირები;
- User: მიმდინარე და მომავალი ტურნირები;
- არაავტორიზებული მომხმარებელი: მიმდინარე და მომავალი ტურნირები.

ვიზუალიზაცია:

ინფორმაციის გამოსატანად გამოყენებულია 3 tab-ი (მომდინარე, მომავალი და უკვე ჩატარებული ტურნირები)-თვის. თითოეულისთვის გამოტანილია შესაბამისი კონტენტი ჩამონათვალის სახით. მომხმარებლისთვის, რომლისთვისაც დაშვებულია მოდიფიკაცია, სიის ელემენტზე განთავსებულია მართვის ღილაკი. ასევე თითოეულ მათგანი შეიცავს ტურნირის დასახელებას, აღწერას, თარიღს და წონით კატეგორიას.

ვალიდაცია:

Client Side ვალიდაციისთვის გამოყენებული იქნება Remote Validation, რაც შეამოწმებს რომ არ მოხდეს ტურნირის სახელის დუბლირება.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი (Add, Edit, Delete).

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- დასახელება: string Name
- აღწერა: string Description
- თარიღი: DateTime Date
- წონოთი კატეგორია: int Weight
- სტატუსი: int StatusId

ყველა ველი სავალდებულოა

ტურნირზე მონაწილე სპორტსმენების გვერდი

აღწერა:

ტურნირზე მონაწილე სპორტსმენების გვერდზე გადასვლა ხდება ტურნირების გვერდიდან კონკრეტული ტურნირზე განთავსებული მართვის ღილაკზე დაჭერით. გვერდზე განთავსებულია აღნიშნულ ტურნირზე მონაწილე ყველა სპორტსმენის ჩამონათვალი და შესაძლებელია მათი მოდიფიკაცია ტურნირის დაწყებამდე. სპორტსმენების დამატების შემდგომ შესაძლებელია ტურნირის შედგენა ღილაკზე დაჭერით, რითიც მოხდება სპორტსმენების დაწყვილება და შეწყვილების ცხრილის შედგენა სპეციალური ალგორითმის მეშვეობით.

ფუნქციონალი:

- არსებულ ტურნირზე სპორტსმენის დამატება, განთესვა;
- არსებულ ტურნირზე სპორტსმენის წაშლა;
- არსებული ტურნირის მართვა;
- არსებული ტურნირის დეტალური ინფორმაციის ნახვა;
- არსებული ტურნირის შეწყვილების ცხრილის აგება.

უფლებები:

გვერდზე შესვლა და ფუნქციონალის გამოყენება შეუძლია მხოლოდ SuperAdmin-სა და Admin-ს.

ვალიდაცია:

Client Side ვალიდაციისთვის გამოყენებულია Remote Validation, რაც შეამოწმებს რომ არ მოხდეს ტურნირზე სპორტსმენის დუბლირება.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი (Add, Edit, Delete).

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- ტურნირის იდენტიფიკატორი: int TournamentId
- სპორტსმენის იდენტიფიკატორი: int SportsmanId
- განთესვა: bool Seeded

ყველა ველი სავალდებულოა

ტურნირის შემადგენელი შეჯიბრებების გვერდი

აღწერა:

ტურნირის შემადგენელი შეჯიბრებების გვერდზე განთავსებულია დეტალური ინფორმაცია ტურნირის შესახებ და ასევე შეწყვილების ცხრილი, თუ ის უკვე შედგენილია. მასზე გადმოსვლა ხდება ტურნირების გვერდიდან კონკრეტული ტურნირზე განთავსებული დეტალების ღილაკზე დაჭერით. იმ შემთხვევაში, თუ ტურნირი ჯერ არ დაწყებულა ადმინისტრატორის მიერ შესაძლებელია შეწყვილების ცხრილის შედგენა ტურნირზე მონაწილე სპორტსმენების გვერდზე „ტურნირის შედგენის“ ღილაკზე დაჭერით. ასევე აღნიშნულ გვერდზე შესაძლებელია სპორტსმენების გადაადგილება შეწყვილების ცხრილში.

ფუნქციონალი:

არსებულ შეწყვილების ცხრილში სპორტსმენების გადაადგილება.

უფლებები:

გვერდზე დაიშვებიან ავტორიზებული და არაავტორიზებული მომხმარებლები; ფუნქციონალის გამოყენება შეუძლია მხოლოდ SuperAdmin-სა და Admin-ს.

ვიზუალიზაცია:

ინფორმაციის სახით უნდა გამოჩნდეს ტურნირის შესახებ დეტალური ინფორმაცია და აიგოს შეწყვილების ცხრილის ვიზუალური მხარე.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი.

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- ტურნირის იდენტიფიკატორი: int TournamentId
- შეჯიბრის იდენტიფიკატორი: int WrestleId

ყველა ველი სავალდებულოა

შეჯიბრის კონტროლის გვერდი

აღწერა:

შეჯიბრის კონტროლის გვერდზე განთავსებულია ორ სპორტსმენს შორის შეჯიბრის მართვის ფუნქციონალი და ინფორმაციის გამოტანა.

ფუნქციონალი:

- სპორტსმენებისთვის იპონის ქულის დამატება/დაკლება/გამოჩენა;
- სპორტსმენებისთვის ვაზარის ქულის დამატება/დაკლება/გამოჩენა;
- სპორტსმენებისთვის გაფრთხილების ქულის დამატება/დაკლება/გამოჩენა;
- შეჯიბრის დროის დაწყება, გაჩერება, განულება;
- მოქმედების ტექსტის დამატება/წაშლა/გამოჩენა;
- მოქმედების ტექსტის ფაილად გენერირება.

უფლებები:

გვერდზე შესვლა და ფუნქციონალის გამოყენება შეუძლია მხოლოდ ავტორიზებულ მომხმარებელს.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი.

დამატებითი ინფორმაცია:

ფუნქციონალის ქმედებების შესრულებისას ინფორმაცია ინახება ბაზაში და აისახება როგორც შეჯიბრის კონტროლის გვერდზე ასევე შეჯიბრის გვერდზე, SignalR - ის გამოყენებით.

ბაზის მოდელი:

- იდენტიფიკატორი: int Id [Key]
- პირველი სპორტსმენის იდენტიფიკატორი: int SportsmanOneId
- მეორე სპორტსმენის ერთის იდენტიფიკატორი: int SportsmanTwoId
- პირველი სპორტსმენის ვაზარის ქულა: int SporstmanOneWazariPoint
- მეორე სპორტსმენის ვაზარის ქულა: int SporstmanTwoWazariPoint
- პირველი სპორტსმენის იპონის ქულა: bool SporstmanOneIppon
- მეორე სპორტსმენის იპონის ქულა: bool SporstmanTwoIppon
- პირველი სპორტსმენის ჯარიმის ქულა: int SportmanOneFine
- მეორე სპორტსმენის ჯარიმის ქულა: int SportmanTwoFine
- შეჯიბრის დრო: Time time
- დროის მიმართულება: bool Decreesing
- სტატუსი: int statusId
- გამარჯვებული სპორტსმენის იდენტიფიკატორი: int WinnerId

ყველა ველი სავალდებულოა

შეჯიბრის გვერდი

აღწერა:

შეჯიბრის გვერდზე განთავსებულია ორ სპორტსმენს შორის შეჯიბრის ინფორმაციის გამოტანა.
ფუნქციონალი:

- ინფორმაციის გამოჩენა

უფლებები:

გვერდზე დაიშვებიან ავტორიზებული და არაავტორიზებული მომხმარებლები;
მომხმარებლების გვერდი

აღწერა:

მომხმარებლების გვერდზე განთავსებულია სისტემაში არსებული ყველა მომხმარებლების ჩამონათვალი.
ფუნქციონალი:

- მომხმარებლის დამატება;
- არსებული მომხმარებლის რედაქტირება;
- არსებული მომხმარებლის წაშლა;

უფლებები:

გვერდზე შესვლა და ფუნქციონალის გამოყენება შეუძლია მხოლოდ Administrator მომხმარებელს.

- SuperAdmin: ყველა ტიპის მომხმარებელზე მოდიფიკაცია;
- Admin: User ტიპის მომხმარებლებზე მოდიფიკაცია;

ვიზუალიზაცია:

ინფორმაციის გამოსატანად და მათ დასამუშავებლად გამოყენებულა Telerik UI Grid Window Editing - ის ფუნქციონალით.

ვალიდაცია:

Client Side ვალიდაციისთვის გამოყენებულია Remote Validation, რომელიც შეამოწმებს რომ არ მოხდეს მომხმარებლების დუბლირება.

ლოგირება:

Action Log ცხრილში შეინახოს ყველა POST ტიპის მეთოდი (Add, Edit, Delete).

ბაზის მოდელი:

- იდენტიფიკატორი: string Id [Key]
- სახელი: string FirstName
- გვარი: string LastName
- მომხმარებლის სახელი: string UserName
- პაროლი: string PasswordHash
- სტატუსი: bool IsDeleted

ყველა ველი სავალდებულოა

5. დასკვნა

მოგეხსენებათ, თანამედროვე სისტემები აქტიურად გამოიყენება თანამედროვე სპორტსა და მის მენეჯმენტში. ასევე, სხვადასხვ ადმინისტრაციულ საკითხებში. მთავარი მიზეზი გახლავთ ის, რომ ამგვარი სისტემები როგორც შესამჩნევად აიოლებენ შეჯიბრებების ორგანიზებისა და ხელმძღვანელობის პროცესს, ისე საშუალებას აძლევენ სპორტულ ფედერაციებს, მოახდინონ სპორტსმენებისა და თანამშრომლების მართვა და სასურველი ინფორმაციის ორგანიზებულად განთავსება, გავრცელება.

გამომდინარე იქიდან, რომ ამგვარი სისტემები საჭიროა ქართულ სპორტულ მენეჯმენტშიც, ფედერაციები სხვადასხვა სახის რესურსებით ეძიებიან მათ. საბოლოო ჯამში კი - უცხო ქვეყნიდან იღებენ მაგალითს და უშედეგოდ ცდილობენ მათ ქართულ სტრუქტურაზე მორგებას. სწორედ ამიტომ, ეს ნაშრომი გთავაზობთ სრულიად ახალ მიგნებას ქართულ სპორტულ მენეჯმენტში ძიუდოს მაგალითზე. ასეთი სახის პროექტი უნიკალურია საქართველოში და ანალოგი არ აქვს.

ასეთი მიდგომა გვეხმარება, ეროვნულ დონეზე ჩატარებულ შეჯიბრებზე შეჯიბრისას საინფორმაციო ტაბლოს ვმართოთ და ინფორმაციული სახით ვაჩვენოთ. რაც ძალიან მნიშვნელოვანია, ის ასევე აერთიანებს საქართველოში მოასპარეზე სპორტსმენებისა და ორგანიზებული ტურნირების ბაზას. აგრეთვე, მათ სამართავ პანელს. ალგორითმი, რომელსაც აპლიკაცია შეიცავს, ტურნირზე წარდგენილი სპორტსმენების შეწყვილების ცხრილს ადგენა (იძლევა მოდიფიკაციის საშუალებას), ასევე აკეთებს მესამე ადგილის შეწყვილების ალგორითმს.

ამასთან პროექტში გამოყენებული ტექნოლოგიები შერჩეულია თანამედროვეობიდან გამომდინარე და წარმოადგენენ მიმდინარე, სწორ და ეფექტურ მიდგომებს. პროექტის არქიტექტურა სრულიად აკმაყოფილებს პროგრამული უზრუნველყოფის მაჩვენებლებს როგორებიცაა სისწორე, მდგრადობა, გაფართოებითობა, დიზაინის სიმარტივე, დეცენტრალიზაცია, ხელმეორედ გამოყენებითობა, თავსებადობა, ეფექტურობა, გამოყენების სიმარტივე, ფუნქციონალურობა, პორტაბელურობა, დროულობა, შემოწმებადობა, ინტეგრირებულობა და შეკეთებადობა. ასევე აღსანიშნავია ის ფაქტი, რომ პროექტი მოქნილია და შესაძლებელია მისი მრავალმხრივ განვითარება და გაუმჯობესება - ახალი ფუნქციონალის დამატებით.

ლიტერატურა

- https://books.google.ge/books/about/Professional_ASP_NET_MVC_5.html?id=FaQLBAAAQBAJ&source=kp_cover&redir_esc=y
- https://docs.microsoft.com/en-us/aspnet/mvc/overview/getting-started/introduction/getting-started
- https://docs.microsoft.com/en-us/dotnet/csharp/
- https://www.telerik.com/documentation
- https://docs.microsoft.com/en-us/azure/
- https://www.wikipedia.org/
- https://www.pluralsight.com/
- https://docs.microsoft.com/en-us/aspnet/signalr/overview/getting-started/introduction-to-signalr
- http://sdsu-dspace.calstate.edu/bitstream/handle/10211.10/1228/Mirza_Irfan.pdf?sequence=1
- https://pdfs.semanticscholar.org/bae8/b8bababe6c6936556046c8b05929ec65095a.pdf
- http://www.sersc.org/journals/IJHIT/vol8_no3_2015/34.pdf
- https://www.capterra.com/sports-league-software/