ძიუდოს შეჯიბრის ტაბლოს პროგრამული უზრუნველყოფა

გიორგი მირზაშვილი | გურამ ქოჩლაძე | ირაკლი საბაშვილი

Email: giorgimirzashvili@outlook.com

**რეზიუმე** — ძიუდოს შეჯიბრის ტაბლოს პროგრამული უზრუნველყოფისათვის საჭირო იყო web აპლიკაციის პლატფორმა, ბაზასთან წვდომა და Responsive Layout. ამოცანის კომპლექსურობის გათვალისწინებით, ჩვენმა ჯგუფმა აირჩია შემდეგი ტექნოლოგიები: ASP.NET MVC 4, Entity Framework, Bootstrap, Javascript და JQuery. ამ ტექნოლოგიების კომბინაციით შევქმენით ტაბლო და მისი სამართავი გვერდი, რომლის გამოყენებაც ნებისმიერი კომპიუტერული მოწყობილობისთვისაა გათვალისწინებული. ეს საშუალებას გვაძლევს, რომ ტაბლო გამოისახოს ნებისმიერი ზომის ეკრანზე და ამავდროულად, იგი იმართებოდეს პლანშეტის ან/და პერსონალური კომპიუტერის მეშვეობით. ამოცანის ტიპიდან გამომდინარე, ინფორმაცია ახლდება რეალურ დროში, რაც საშუალებას გვაძლევს მყისიერად ვცვალოთ სამართავი გვერდიდან ტაბლოზე. ინფორმაციის მიმოცვლა არ ხდება მხოლოდ ტაბლოსა და მის სამართავ გვერდს შორის. ყოველი ცვლილება, და შედეგი აისახება მონაცემთა ბაზაში. ყოველივე ამ ტექნოლოგიის შერწყმამ დააკმააყოფილა ამოცანის მოთხოვნები.

**გასაღები სიტვვები -** ASP.NET, MVC, C#, SignalR, JQuery, JavaScript, BootsTrap, Web, MsSql

2017-2018 სასწ.წელი თსუ კომპიუტერულ მეცნიერებათა დეპარტამენტი

—————————— ◆ ——————————

# 1 შესავალი

ძიუდოს შეჯიბრის ტაბლოს პროგრამული უზრუნველყოფის ამოცანაში მოითხოვებოდა ორი გვერდი, უშუალოდ ტაბლო და მისი სამართავი პანელი. სამართავ პანელზე ასახული ცვლილებები ავტომატურად უნდა ასახულიყო ტაბლოზე გვერდის გადატვირთვის გარეშე. ამასთანავე ცვლილებები უნდა განხორციელებულიყო მონაცემთა ბაზაში. შეწყვილების ალგორითმის შედეგად, პროგრამა ირჩევს ორ შესაფერის სპორტსმენს. ტაბლოზე გამოსახულია სპორტსმენის პირადი ინფორმაცია: სახელი, გვარი, რეგიონი და წონითი კატეგორია, აგრეთვე იპონის, ან ვაზარის ამსახველი ქულა. აგრეთვე გვაქვს წამზომი, რომლის ათვლა იწყება 4:00 წუთიდან და კლებით ჩამოდის 0:00-მდე. ამ 4 წუთის განმავლობაში თუ გამარჯვებული არ გამოვლინდა, თვლა იწყება 0:00-დან ზრდადი მიმართულებით, სანამ არ გამოვლინდება გამარჯვებული. თუ ზრდადი დროის პერიოდში რომელიმე სპორტსმენი მოიპოვებს ვაზარის ქულას, იგი ავტომატურად გამოცხადდება როგორც გამარჯვებული. ტაბლოზე არსებობს გაფრთხილების ბარათებიც, რომლებიც თავიდან არ ჩანს. იმ შემთხვევაში, თუ რომელიმე სპორტსმენი დაარღვევს წესს, ბარათი გამოისახება ტაბლოზე. თუ სპორტსმენი მიიღებს სამ გაფრთხილებას, იგი ავტომატურად აგებს. გამარჯვების შემთხვევაში, ტაბლოზე გამოდის ფანჯარა, სადაც წერია გამარჯვებული სპორტსმენის სახელი და გვარი.

ჩვენმა ჯგუფმა, მოცემული დავალება დაყო სამ ნაწილად: ტაბლოს და სამართავი პანელის დიზაინი, ფუნქციონალი და ბაზასთან ურთიერთობა.

ტაბლოს დიზაინისთვის ავირჩიეთ Bootstrap ტექონლოგია, რომელიც პროპორციულად ცვლის ინფორმაციას ნებისმიერი ზომის ეკრანისთვის. მოცემული ამოცანისთვის მხოლოდ Bootstrap ტექნოლოგია არ გვეყო, ამიტომ გამოვიყენეთ HTML5 და CSS3-ის Responsive თვისებებიც.

ლოგიკისათვის გამოვიყენეთ ASP.NET Razor სინტაქსი, JavaScript, jQuery და ASP.NET SignalR ტექონოლოგიები. ეს უკანასკნელი კი საშუალებას გვაძლევს მონაცემები ვცვალოთ რეალურ დროში.

ბაზასთან ურთიერთობისათვის გამოვიყენეთ ASP.NET Entoty Framework. ეს იყო ყველაზე ოპტიმალური მიდგომა, ვინაიდან ავირჩიეთ Code First, რომელიც მდგომარეობს იმაში, რომ, პირველ რიგში, იწერება აპლიკაცია და იქმნება მოდელები და ამის შემდგომ გენერირდება ბაზა. ეს Code First მიდგომა საბაზისოდ იყენებს Entity Framework-ს. შესაბამისად, ამ Framework-ის გამოყენება ყველაზე ოპტიმალური გამოსავალი იყო.

ASP.NET-ამ ეს პლატფორმამ საშუალება მოგვცა, რომ ყველა ზემოთხსენებულ ტექნოლოგიას ემუშავა სინქრონულად და ამოცანის ყოველი პუნქტი დაეკმაყოფილებინა.

# 2 განხორციელებული სამუშაო (არსებული მდგომარეობა)

როგორც ზემოთ აღვნიშნეთ, ჩვენმა ჯგუფმა სამუშაო პროცესი დაყო სამ ნაწილად: ტაბლოს და სამართავი პანელის დიზაინი, ფუნქციონალი და ბაზასთან ურთიერთობა.

ტაბლოს და პანელის დიზაინისთვის უმეტეს წილად გამოვიყენეთ Bootstrap 3 ტექნოლოგია. ეს ტექნოლოგია ავირჩიეთ იქიდან გამომდინარე, რომ ყველაზე მოსახერხებელ და ოპტიმალურ ტექნოლოგიად გვესახებოდა მოცემული სამუშაოს შესასრულებლად. ძირითადი მიზეზი რაშიც Bootstrap გამოგვადგა, იყო Responsive დიზაინი, რაც საშუალებას გვაძლევს, რომ გამოსახულება შესაბამისად მოერგოს სხვადასხვა რეზოლუციის მქონე კომპიუტერულ მოწყობილობას, იქნება ეს ტაბლო, პლანშეტი, თუ პერსონალური კომპიუტერი. მაგრამ, მიუხედავად იმისა, რომ Bootstrap ტექნოლოგია ბევრ ასპექტში დაგვეხმარა, იგი საკმარისი არ იყო ზოგიერთი სტრუქტურული და სტილისტური დეტალების შესასრულებლად, რისთვისაც უკვე გამოვიყენეთ HTML5 და CSS3 ტექნოლოგიები. სტრუქტურა ძირითადად აწყობილია HTML5 და მასში ინტეგრირებული Bootstrap 3 ტექნოლოგიებით. ეს უკანასკნელი გვაძლევს საშუალებას კომპაქტურად და მოსახერხებლად ავაგოთ დიზაინის სტრუქტურა, ნაკლები კოდის გამოყენებით. ყველაზე მეტად დაგვეხმარა იმაში, რომ დიზაინი ყოფილიყო მაქსიმალურად Responsive. Bootstrap 3-ით, ასევე სტილისტურად გავაფორმეთ აგებული სტრუქტურა, მაგრამ პატარა დეტალების გამო, რომელთა უზრუნველყოფაც მან ვერ შეძლო, გამოვიყენეთ СSS 3 ტექნოლოგია. CSS3-ის მეშვეობით მოვახდინეთ ტექსტის ზომების სწორად შერჩევა და პოზიციონირება, ფონის ფერების შერჩევა, ტექსტის Responsive-ობა vw ერთეულებით, ბარათების გამოსახვა, პოზიციონირება და ფერები, წონის და ტაიმერის ცენტრში პოზიციონირება.

ტაბლოს ფუნქციონალისათვის საჭირო გახდა გამოგვეყენებინა რეალურ დროში მონაცემთა გაცვლისა და ვიზუალიზაციის ტექნოლოგია, ASP.NET Razor სინტაქსი, Javascript და JQuery.

არსებობს რამოდენიმე გზა, რათა განხორციელდეს რეალურ დროში კომუნიკაცია Web აპლიკაციებში, როგორიცაა: Long Pooling, Interval Pooling, Server Sent Events, Web Sockets და სხვა. მათ შორის HTML5 Web Sockets არის ერთ-ერთი ყველაზე პოპულარული გზა. Web Sockets-მა, რომ გამართულად იმუშაოს მას სჭირდება როგორც კლიენტური მხარდაჭერა, ასევე სერვერულიც. თუ რომელიმე მხარს არ უჭერს Web Sockets-ს, რეალურ დროში კომუნიკაცია არ დამყარდება. ასეთ შემთხვევაში ჩვენ გვჭირდება PolyFill, რომელიც მოიცავს რამდენიმე Fallback მექანიზმს. ASP.NET SignalR უზრუნველყოფს ამ ფუნქციას უფასოდ. თუ ჩვენ გამოვიყენებთ SignalR-ს, ჩვენ აღარ მოგვიწევს ხელით აღმოვაჩინოთ ტექნოლოგიები, რომლებიც გაშვებულ პლათფორმას მხარდაჭერას უწევენ. SignalR-ი ყველა არხულ ლოგიკას აბსტრაგირებს, რომ აღმოაჩინოს ყველა მათგანი და აირჩიოს საუკეთესო. SignalR-ი ამოწმებს და იყენებს ერთ-ერთ ხერხს ქვემოთ მოცემული სიიდან:

* Web Sockets: HTML5 რეალურ დროში კომუნიკაციის ფუნქცია, რომელიც იყენებს სპეციალურ პროტოკოლს ‘ws’
* Server Sent Events: HTML5 რეალურ დროში კომუნიკაციის ფუნქცია HTTP-სა და HTTPS-ზე
* Forever Frames: ამატებს დამალულ iFrames-ს გვერდზე და მართავს კომუნიკაციას ამ Frame-ის გამოყენებით
* Long Pooling: კლიენტი გამუდმებით რაიმე ინტერვალით ითხოვს ინფორმაციას ბაზიდან, რათა მუდმივად უახლეს ინფორმაციას ფლობდეს.

როგორც უკვე ავღნიშნეთ, ყველა თანამედროვე Browser-ს აქვს კარგი გრაფიკული მხარდაჭერა Canvas და SVG ფუნქციონალის გამოყენებით. JavaScript API გვაძლევს საშუალებას Browser-ზე ვმართოთ გრაფიკული გამოსახულებები კოდის მეშვეობით ისე, როგორც ჩვენ გვსურს. ახალი ტექნოლოგიის ათვისების დროს, API ძალიან მოსახერხებელი ჩანს, მაგრამ იგი ძალიან მოუქნელია, როდესაც მარტივი გამოსახულების შექმნიდან გადავდივართ რთულ გრაფიკულ რენდერირებაზე. ეს ის შემთხვევაა, როდესაც საჭიროა აბსტრაქცია, რომელიც ჩვენს საქმეს ამარტივებს. D3 არის SVG-ს აბსტრაქცია. D3 ამარტივებს SVG-ს გამოყენებას და უზრუნველყოფს მრავალფეროვანი გრაფიკის შექმნას.

ეს ორი ტექნოლოგია ძალიან მომხიბვლელია ნებისმიერი დეველოპერისათვის, მაგრამ ისმის კითხვა, აქვს თუ არა ამ ტექნოლოგიებს მინუსი? ყოველთვის ოპტიმალურია მათი გამოყენება? ASP.NET SignalR-ი საკმაოდ მოსახერხებელი და საიმედო მიდგომაა რეალურ დროში კომუნიკაციებისათვის, მაგრამ მას აქვს უარყოფითი მხარეებიც, ესენია:

* არ გვაძლევს 100%-იან საიმედოობას კომუნიკაციისას.
* არ უნდა გამოვიყენოთ ეს ტექნოლოგია, თუ შეტყობინების გარანტირებული მიწოდება პრიორიტეტულია. (საბანკო ტრანსფერი არის ამის ნათელი მაგალითი).
* ეს ტექნოლოგია დინამიკურია , ამიტომ ზოგიერთი სინტაქსური შეცდომა მხოლოდ კოდის გაშვების შემდეგ იჩენს თავს.
* დიდი კოდის დებაგირების დროს უჭირს მუშაობა.
* შეუძლია შექმნას მხოლოდ 10 კავშირი , როდესაც იყენებს IIS სერვერს.
* Web კლიენტი და სერვერი იყენებს სხადასხვა ტექნოლოგიებს, ამიტომ ისინი არ იმეორებენ ერთი და იგივე კოდს.

D3.js თანამედროვე და ერთ-ერთი საუკეთესო გზაა მონაცემთა ვიზუალიზაციისათვის, მაგრამ არც ის აღმოჩნდა უნაკლო, მისი უარყოფითი მხარეებია:

* DOM-ის მანიპულაციები ძალიან ნელია დიდი ზომის მასალის რენდერირებისას.
* SVG-ს აქვს დატვირთვის ლიმიტები (დიდი ზომის ინფორმაციის დამუშავებისას).

ASP.NET Razor სინტაქსი საშუალებას გვაძლევს მარტივად და ეფექტურად ვმართოთ web აპლიკაციის View გვერდები. მისი გამოყენებით HTML-ში შეგვიძლია ჩავაშენოთ C# ლოგიკა, რაც ბევრად ამარტივებს თანამედროვე დონის აპლიკაციების შექმნას. Javascript და მისი ბიბლიოთეკა jQuery გამოიყენება დინამიკური web პროგრამირებისათვის. დღევანდელი web საიტები მათ გარეშე წარმოუდგენელია.

ტაბლოს ფუნქციონალისათვის კომბინირებულად გამოვიყენეთ ASP.NET Razor, C#, Javascript და jQuery. მოთხოვნილებებიდან გამომდინარე ტაბლოს სამართავი გვერდსა და უშუალოდ ტაბლოზე ინფორმაცია უნდა განახლებულიყო რეალურ დროში და ასახულიყო ცვლილებები ბაზაში. ამის მისაღწევათ გამოვიყენეთ ASP.NET SignalR ტექნოლოგია, რომელმაც წარმატებით გადჭრა დასმული პრობლემა. ასევე Javascript-ის საშუალებით შევქმენით წამზომი, რომელსაც შეუძლია გააჩეროს, გააგრძელოს და გაანულოს მიმდინარე დრო. იგი იწყებს ათვლას ოთხი წუთიდან და უკუსვლით მუშაობს. თუ ოთხი წუთის განმავლობაში გამარჯვებული არ გამოვლინდება, წამზომი ამვტომატურად იწყებს დადებითი მიმართულებით ათვლას და იცვლება გამარჯვებისათვის საჭირო პირობებიც. კონკრეტულად მოსაგებად საკმარისი ხდება ვაზარის ქულის მოპოვება. Razor სინტაქსის მეშვეობით ვიზუალიზაცია გავუკეთეთ ინფორმაციას ბაზიდან და მისი მეშვეობით ვმართეთ საჯარიმო ბარათები. ასევე გამოვიყენეთ Remodal ფუნქცია. მისი მეშვეობით ეკრანზე გამოდის ფანჯარა სადაც მითითებულია გამარჯვებული სპორტსმენი. ყოველივე ზემოთ აღწერილმა ტექნოლოგიამ მოგვცა საშუალება დაგვეკმაყოფილებინა ის ფუნქციონალი, რაც მოთხოვნილი იყო.

ჩვენი ამოცანა იყო რელაციური ბაზიდან ობიექტზე-ორიენტირებულ ენაში ინფორმაციის მიმოცვლა. ამ ამოცანას, კი გარკვეულ წინააღმდეგობებთან მივყავართ, ვინაიდან არ არსებობს ერთი კონკრეტული და დომინანტური გზა. არჩევანის გაკეთება მოგვიწია ჩვენი მოთხოვნებიდან და შესასრულებელი სამუშაოდან გამომდინარე.

ბაზასთან ურთიერთობისთვის ავირჩიეთ Entity Framework, ვინაიდან თავად აპლიკაცია დაიწერა Code-First მიდგომით. ეს ნიშნავს, რომ ჯერ იწერება კოდი და ობიექტები, შემდეგ კი გენერირდება ბაზა ობიექტების და მათი კავშირების შესაბამისად. ამისთვის ASP.NET იყენებს Entity Framework-ს ან ADO.NET-ს. არჩევანი გაკეთდა პირველზე, რადგან ჩვენი აპლიკაციისთვის უფრო პრაქტიკული აღმოჩნდა.

აპლიკაცია საჭიროებდა ინფორმაციის მიმოცვლას სამართავ პანელსა და ტაბლოს შორის. სამართავ პანელში, თამაშის დროს შეყვანილი ინფორმაცია ინახება მონაცემთა ბაზაში. ასევე ყველა მონაცემის წამოღება ხდება უშუალოდ ბაზიდან, როგორც სამართავ პანელში ასევე ტაბლოზე. ეს ყოველივე მიმდინარეობს Live რეჟიმში, პანელში ქულის მომატების დროსაც ქულა ჯერ ინახება ბაზაში, შემდეგ ბაზაში განახლებული ინფორმაცია აისახება ტაბლოზე. ამ ცვლილებას უზრუნველყოფს Entity Framework და Javascript.

მონაცემთა შეცვლისას Javascript-ით ინფორმაცია გადაეწოდება კლას ფაილს, სადაც Entity Framework შეინახავს მონაცემთა ბაზაში, შემედეგ უკვე განახლებული მონაცემები ბრუნდება და გადაეწოდება ტაბლოს.

# 3 სისტემური მოდელი

ძიუდოს შეჯიბრის ტაბლოს პროგრამული უზრუნველყოფის ამოცანა, პირველ რიგში, თემატურად დავყავით სამ ნაწილად: ტაბლოს და სამართავი პანელის დიზაინი, ფუნქციონალი და ბაზასთან ურთიერთობა. ამის შემდეგ თითოეული ნაწილი დავამუშავეთ და შევქმენით მათი მოდელები.

დიზაინის მხარეს, ფოტოშოპში შევქმენით კოცეპტუალური მოდელი და შემდეგ ეს მოდელი განვახორციელეთ HTML5, CSS3 და Bootstrap 3-ის მეშვეობით.

ტაბლოსა და სამართავ პანელზე გამოტანილია სპორტსმენის და თამაშის მონაცემები. ამ მონაცემებს ვიღებთ მონაცემთა ბაზიდან. სამართავ პანელში ხდება თამაშიდან გამომდინარე შესაბამისი ცვლილება. მონაცემთა ბაზაში არსებული მონაცემების ცვლილება აისახება სამართავ პანელზე და ტაბლოზე ერთდროულად. შესაბამისად სამართავ პანელში შეცვლილი მონაცემები აგრეთვე ახლდება ბაზაში.

პირობითად, მატჩის დროს ინფორმაცია გამოდის სამართავ პანელზე, შემდეგ, თამაშის პროცესში, მომხარებლის მიერ შეტანილი ცვლილებები აისახება ბაზაში და შესაბამისად განახლდება ტაბლოზე გამოტანილი ინფორმაციაც.

# 4 ამოცანის დასმა

ჩვენი ჯგუფის დავალება მდგომარეობდა შემდეგში: შეგვექმნა ტაბლო თავისი სამართავი პანელით, რომელზეც გამოსახული იქნებოდა სპორტსმენისა და თამაშის შესაბამისი ინფორმაცია. ტაბლოსა და სამართავ პანელზე ინფორმაცია უნდა განახლებულიყო რეალურ დროში და ამავდროულად შეცვლილი ინფორმაცია უნდა ასახულიყო ბაზაშიც. ასევე ნებისმიერი რეზოლუციის ეკრანისთვის, ტაბლოს დიზაინი უნდა ყოფილიყო თავსებადი.

# 5 პრობლემის გადაწყვეტა

პირველი პრობლემა რაც გადავწყვიტეთ, იყო Responsive-ობა, ანუ ტაბლოს დიზაინის მორგება ნებისმიერი რეზოლუციის მქონე კომპიუტერული მოწყობილობისთვის, იქნება ეს თვითონ ტაბლო, პლანშეტი თუ პერსონალური კომპიუტერი. ამისთვის გამოვიყენეთ Bootstrap 3. ამ ტექნოლოგიით უზრუნველვყავით დიზაინის მოქნილობა და თავსებადობა.

მეორე პრობლემა იყო ის, რომ ტაბლოზე და სამართავ პანელზე ინფორმაცია უნდა განახლებულიყო რეალურ დროში. ამისათვის გამოვიყენეთ ტექნოლოგია SignalR, რათა, გვერდის გადატვირთვის გარეშე ინფორმაცია განახლებულიყო ორივე გვერდზე და ამასთანავე, ეს განახლებული ინფორმაცია ასახულიყო ბაზაში. მორიგი პრობლემა იყო ტაიმერის რეალურ დროში შეცვლა ტაბლოს გვერდზე. ჩვენ უნდა აგვერჩია ისეთი ტექნოლოგია რომელსაც დიდი დატვირთვა არ ექნებოდა, რადგან ყოველ წამს უნდა განახლებულიყო გვერდი რეალურ დროში და ამას შეეძლო გამოეწვია პრობლემებ: გადაეტვირთა აპლიკაცია ან დაეკარგა სიგნალი და გაჩერებულიყო ტაიმერი. ამ პრობლემის აღმოსაფხვრელად ჩვენ გამოვიყენეთ Javascript და SignalR-ის კომბინაცია რათა შეგვექმნა პლაგინი Board Hub.

ბაზასთან ურთიერთობისთვის ჩვენ ავირჩიეთ Entity Framework, რადგან პროექტი იყო Code First და იგი გვაძლევდა ბაზასთან მარტივი ურთიერთობის დამყარების საშუალებას.

# 6 ანალიზი

სამართავი პანელი და ტაბლო Responsive-ია, რაც ნიშშნავს რომ აპლიკაცია გათვლილია ნებისმიერი რეზოლუციის ეკრანისთვის და ეს უზრუნველყოფილია Bootstrap-ით

ASP.NET პლატფორმამ საშუალება მოგვცა მარტივად შეგვესრულებინა ჩვენი დავალება, კერძოდ კი SignalR საშუალებით რეალურ დროში გამოგვეტანა განახლებული ინფორმაცია ეკრანებზე.

დღევანდელ ბაზარზე უმეტესობა მსხვილი კომპანიებისა იყენებს ASP.NET იგი უზრუნველყოფს მაღალი დონის უსაფრთხოებას და გვთავაზობს ყველა იმ ტექნოლოგიების ინტეგრირებას რომლებიც აუცილებელი იყო ჩვენი ამოცანის თითოეული ნაწილის განსახორციელებლად.

# დასკვნა

ამოცანის გადასაჭრელად გამოვიყენეთ Bootstrap 3, ASP.NET, SignalR და Entity Framework. ამ ტექნოლოგიების მეშვეობით უზრუნველვყავით Responsive-ობა, მონაცემების რეალურ დროში გამოტანა, ბაზაში განახლებული ინფორმაციის ასახვა და Code First ბაზასათან ურთიერთობის დამყარება. შედეგად, მივიღეთ მოქნილი და თავსებადი დიზაინი ნებისმიერი კომპიუტერული მოწყობილობისთვის, სპორტსმენთა მონაცემები და წამზომი რომელიც ახლდება რეალურ დროში, ყოველ წამს და ბაზა, რომელშიც აისახება ცვლილებები აპლიკაციის კოდთან მყარი კავშირის საფუძველზე.

# ლიტერატურა

|  |  |
| --- | --- |
| [1] | P. Fletcher, "https://docs.microsoft.com," Microsoft, 10 06 2014. [Online]. Available: https://docs.microsoft.com/en-us/aspnet/signalr/overview/getting-started/introduction-to-signalr. [Accessed 23 12 2017]. |
| [2] | M. Bostock, "https://d3js.org/," Data-Driven Documents, [Online]. Available: https://d3js.org/. |
| [3] | Microsoft, "https://www.asp.net/," ASP.NET. [Online]. |
| [4] | Bootstrap, "https://getbootstrap.com/," BootsTrap. [Online]. |