МИНИСТЕРСТВО ОБРАЗОВАНИЯ РЕСПУБЛИКИ БЕЛАРУСЬ

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

МЕЖГОСУДАРСТВЕННОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ «БЕЛОРУССКО-РОССИЙСКИЙ УНИВЕРСИТЕТ»

Кафедра «Автоматизированные системы управления»

Лабораторная работа №23

«Работа с базами данных с использованием технологии ASP.NET»

Выполнил:

ст. гр. АСОИ-181

Остапенко Александр Константинович

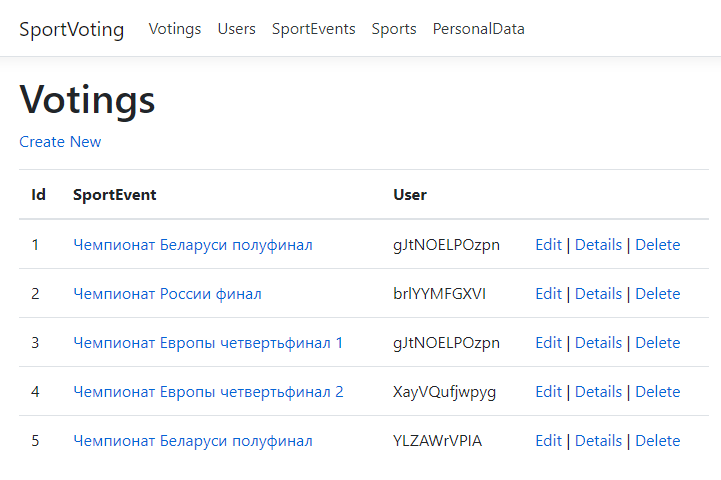
Проверил:

Мрочек Татьяна Владимировна

Цель: изучение основ использования технологии ASP.NET при работе с БД.

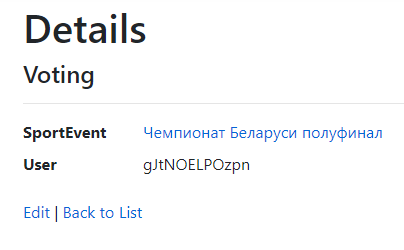
**Задание**

Необходимо создать web-форму для доступа к таблицам базы данных с использованием технологии ASP.NET MVC или ASP.NET Core.

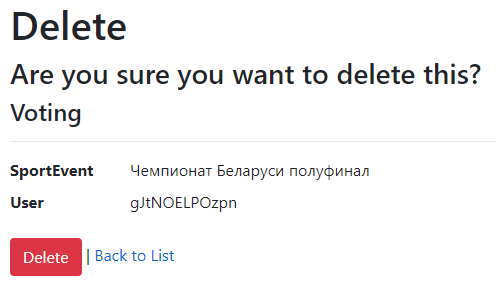


Можно открывать страницы удаления, редактирования и страницы с деталями голосований. Так же можно добавить новое голосование.

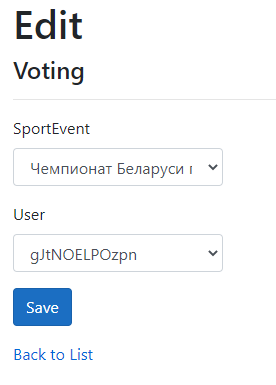
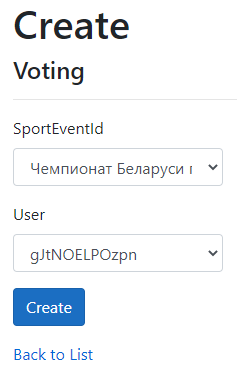
Страница с деталями голосования:



Страница подтверждения удаления голосования:



Страница редактирования голосования: Страница добавления голосования:

Подключение к базе:

protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)

{

if (!optionsBuilder.IsConfigured)

{

optionsBuilder.UseSqlServer("Data Source=woland-a\\mssqlserver01;Initial Catalog=SportVoting;Integrated Security=True");

}

}

Код контроллера:

// GET: Votings

public async Task<IActionResult> Index()

{

return View(await \_context.Votings.Include(v => v.SportEvent).Include(v => v.User).ToListAsync());

}

// GET: Votings/Details/5

public async Task<IActionResult> Details(int? id)

{

if (id == null)

{

return NotFound();

}

var voting = await \_context.Votings

.Include(v => v.SportEvent)

.Include(v => v.User)

.FirstOrDefaultAsync(m => m.Id == id);

if (voting == null)

{

return NotFound();

}

return View(voting);

}

// GET: Votings/Create

public IActionResult Create()

{

ViewBag.SportEvents = new SelectList(\_context.SportsEvents.AsEnumerable(), "Id", "Name");

ViewBag.Users = new SelectList(\_context.Users.Where(u => u.Role.Name == "Organizer" || u.Role.Name == "Admin").AsEnumerable(), "Id", "Login");

return View();

}

// POST: Votings/Create

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Create([Bind("Id,UserId,SportEventId")] Voting voting)

{

if (ModelState.IsValid)

{

\_context.Add(voting);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

return View(voting);

}

// GET: Votings/Edit/5

public async Task<IActionResult> Edit(int? id)

{

if (id == null)

{

return NotFound();

}

var voting = await \_context.Votings

.Include(v => v.SportEvent)

.Include(v => v.User)

.FirstOrDefaultAsync(m => m.Id == id);

if (voting == null)

{

return NotFound();

}

ViewBag.SportEvents = new SelectList(\_context.SportsEvents.AsEnumerable(), "Id", "Name", voting.SportEventId);

ViewBag.Users = new SelectList(\_context.Users.Where(u => u.Role.Name == "Organizer" || u.Role.Name == "Admin").AsEnumerable(), "Id", "Login", voting.UserId);

return View(voting);

}

// POST: Votings/Edit/5

[HttpPost]

[ValidateAntiForgeryToken]

public async Task<IActionResult> Edit(int id, [Bind("Id,UserId,SportEventId")] Voting voting)

{

if (id != voting.Id)

{

return NotFound();

}

if (ModelState.IsValid)

{

try

{

\_context.Update(voting);

await \_context.SaveChangesAsync();

}

catch (DbUpdateConcurrencyException)

{

if (!VotingExists(voting.Id))

{

return NotFound();

}

else

{

throw;

}

}

return RedirectToAction(nameof(Index));

}

return View(voting);

}

// GET: Votings/Delete/5

public async Task<IActionResult> Delete(int? id)

{

if (id == null)

{

return NotFound();

}

var voting = await \_context.Votings

.Include(v => v.SportEvent)

.Include(v => v.User)

.FirstOrDefaultAsync(m => m.Id == id);

if (voting == null)

{

return NotFound();

}

return View(voting);

}

// POST: Votings/Delete/5

[HttpPost, ActionName("Delete")]

[ValidateAntiForgeryToken]

public async Task<IActionResult> DeleteConfirmed(int id)

{

var voting = await \_context.Votings.FindAsync(id);

\_context.Votings.Remove(voting);

await \_context.SaveChangesAsync();

return RedirectToAction(nameof(Index));

}

private bool VotingExists(int id)

{

return \_context.Votings.Any(e => e.Id == id);

}