using Microsoft.AspNetCore.Mvc;

namespace veeb.Controllers

{

[Route("[controller]")]

[ApiController]

public class PrimitiividController : ControllerBase

{

// GET: primitiivid/hello-world

[HttpGet("hello-world")]

public string HelloWorld()

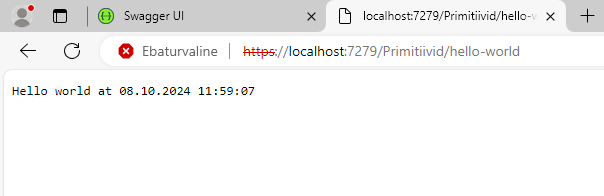
{

return "Hello world at " + DateTime.Now;

}

}

}



[Route("[controller]")]

[ApiController]

public class PrimitiividController : ControllerBase

{

// GET: primitiivid/hello-world

[HttpGet("hello-world")]

public string HelloWorld()

{

return "Hello world at " + DateTime.Now;

}

// GET: primitiivid/hello-variable/mari

[HttpGet("hello-variable/{nimi}")]

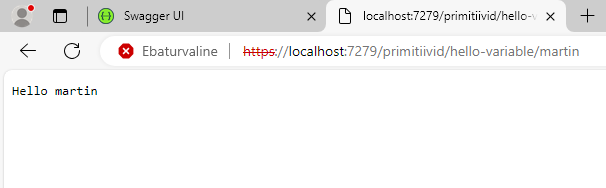
public string HelloVariable(string nimi)

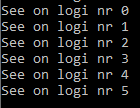
{

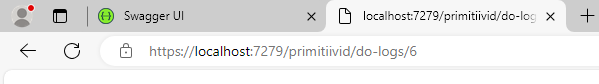
return "Hello " + nimi;

}

}







// GET: primitiivid/do-logs/5

[HttpGet("do-logs/{arv}")]

public void DoLogs(int arv)

{

for (int i = 0; i < arv; i++)

{

Console.WriteLine("See on logi nr " + i);

}

}

// GET: primitiivid/aasta/2000

[HttpGet("aasta/{nr1}")]

public string Synniaasta(string nr1)

{

string AA1970 = "Sina oled vana";

string AA2000 = "Sina oled noor";

string AA2010 = "Sina oled väga noor";

string NoData = "Palun, sisesta arv";

int numericvalue;

bool isNumber = int.TryParse(nr1, out numericvalue);

if (!isNumber)

{

return NoData;

}

if (numericvalue < 1970)

{

return AA1970;

}

else if (numericvalue < 2000)

{

return AA2000;

}

else if (numericvalue < 2010)

{

return AA2010;

}

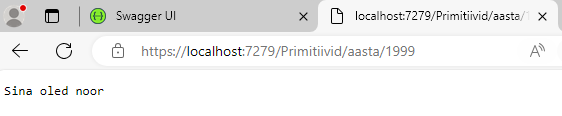
else

{

return NoData;

}

}



// GET: primitiivid/paar-arvud

[HttpGet("paar-arvud")]

public string Paararvud()

{

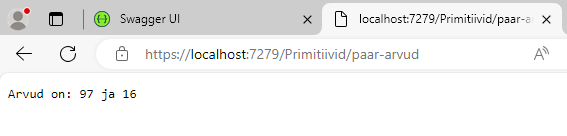
Random rand = new Random();

int Nr1 = rand.Next(1,100);

int Nr2 = rand.Next(1,100);

return $"Arvud on: {Nr1} ja {Nr2}";

}



[Route("[controller]")]

[ApiController]

public class ToodeController : ControllerBase

{

private static Toode \_toode = new Toode(1, "Koola", 1.5, true);

// GET: toode

[HttpGet]

public Toode GetToode()

{

return \_toode;

}

// GET: toode/suurenda-hinda

[HttpGet("suurenda-hinda")]

public Toode SuurendaHinda()

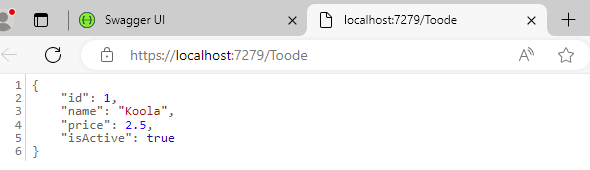
{

\_toode.Price = \_toode.Price + 1;

return \_toode;

}

}



// GET: toode/false-true-false

[HttpGet("false-true-false")]

public Toode FalseTrueFalse()

{

if(\_toode.IsActive == true)

{

\_toode.IsActive = \_toode.IsActive = false;

}

else

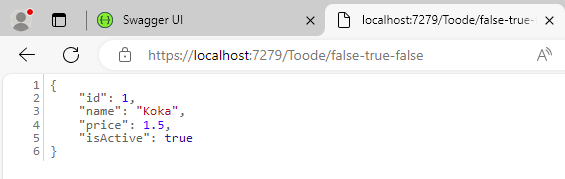
{

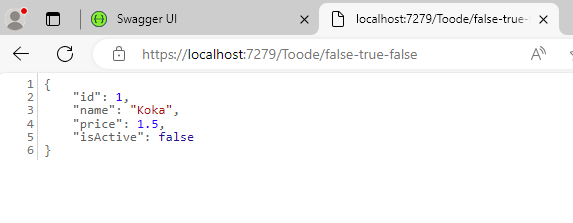
\_toode.IsActive = \_toode.IsActive = true;

}

return \_toode;

}





// GET: toode/ümbernimetamine

[HttpGet("ümbernimetamine")]

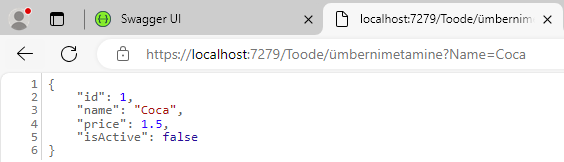
public Toode FalseTrueFalse(string Name)

{

\_toode.Name = Name;

return \_toode;

}



// GET: toode/uushind

[HttpGet("uushind")]

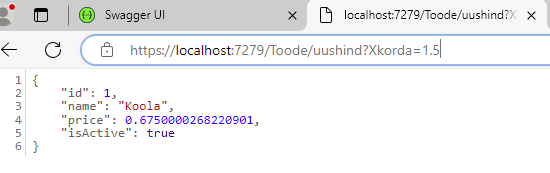
public Toode Uushind(float Xkorda)

{

\_toode.Price \*= Xkorda;

return \_toode;

}



// https://localhost:port/tooted

[HttpGet]

public List<Toode> Get()

{

return \_tooted;

}



[HttpGet("kustuta/{index}")]

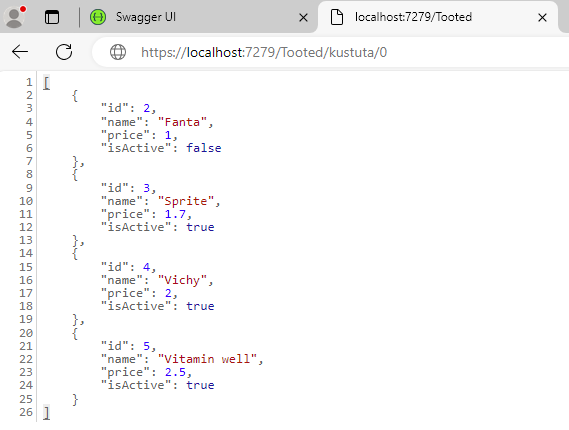
public List<Toode> Delete(int index)

{

\_tooted.RemoveAt(index);

return \_tooted;

}



[HttpGet("kustuta2/{index}")]

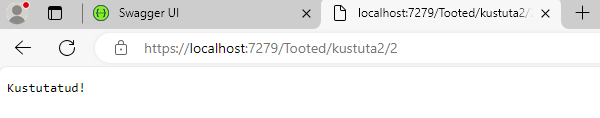
public string Delete2(int index)

{

\_tooted.RemoveAt(index);

return "Kustutatud!";

}



[HttpGet("lisa/{id}/{nimi}/{hind}/{aktiivne}")]

public List<Toode> Add(int id, string nimi, double hind, bool aktiivne)

{

Toode toode = new Toode(id, nimi, hind, aktiivne);

\_tooted.Add(toode);

return \_tooted;

}



[HttpGet("lisa")] // GET /tooted/lisa?id=1&nimi=Koola&hind=1.5&aktiivne=true

public List<Toode> Add2([FromQuery] int id, [FromQuery] string nimi, [FromQuery] double hind, [FromQuery] bool aktiivne)

{

Toode toode = new Toode(id, nimi, hind, aktiivne);

\_tooted.Add(toode);

return \_tooted;

}



[HttpGet("hind-dollaritesse/{kurss}")] // GET /tooted/hind-dollaritesse/1.5

public List<Toode> Dollaritesse(double kurss)

{

for (int i = 0; i < \_tooted.Count; i++)

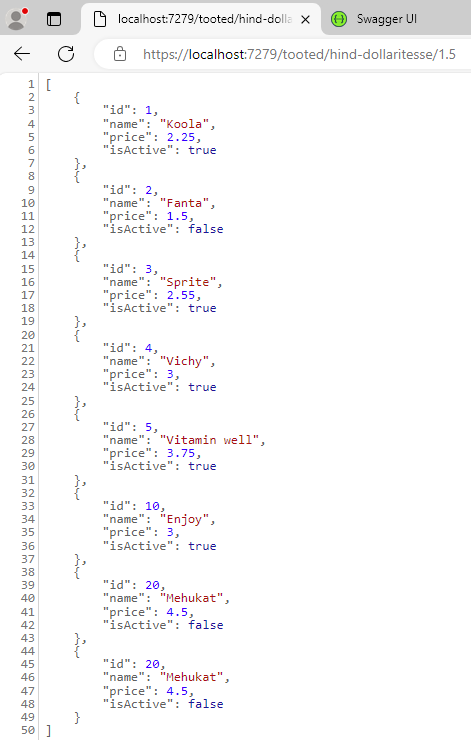
{

\_tooted[i].Price = \_tooted[i].Price \* kurss;

}

return \_tooted;

}



// või foreachina:

[HttpGet("hind-dollaritesse2/{kurss}")] // GET /tooted/hind-dollaritesse2/1.5

public List<Toode> Dollaritesse2(double kurss)

{

foreach (var t in \_tooted)

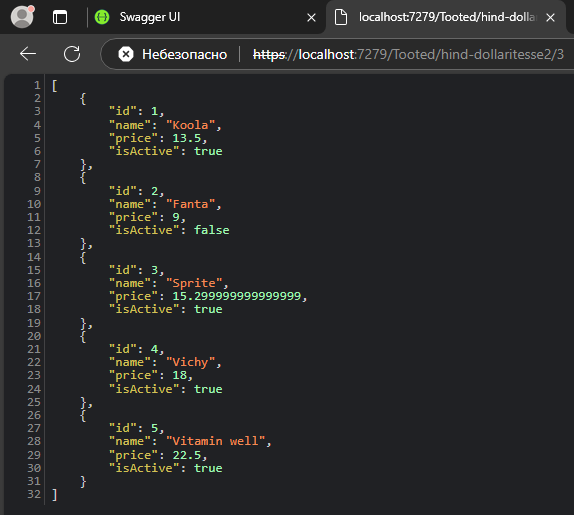
{

t.Price = t.Price \* kurss;

}

return \_tooted;

}



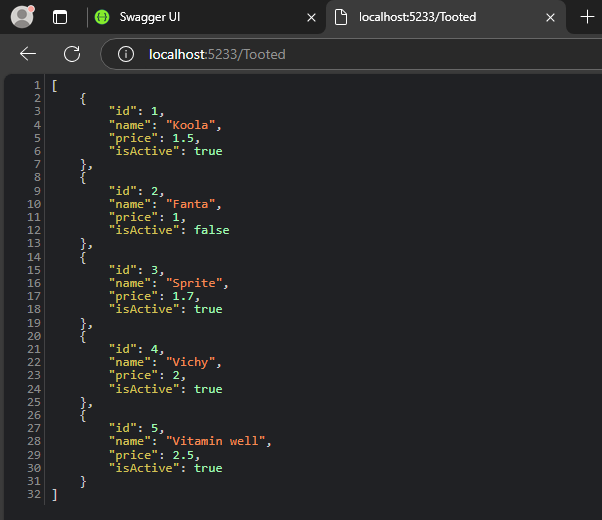
[HttpGet("kustuta-tooded")] // GET /tooted/kustuta-tooded

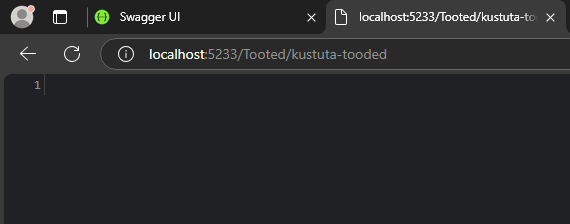
public List<Toode> Kustutakoik()

{

\_tooted.Clear();

return \_tooted;

}



[HttpGet("aktiivsus-väär")] // GET /tooted/hind-dollaritesse/1.5

public List<Toode> Aktiivsus\_vaar()

{

foreach (var t in \_tooted)

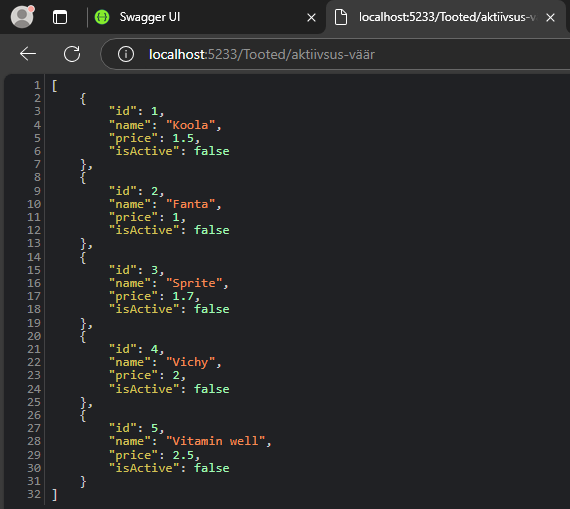
{

t.IsActive = false;

}

return \_tooted;

}



[HttpGet("näita/{id}")] // GET /tooted/näita/id

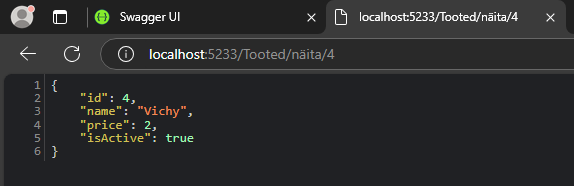
public ActionResult<Toode> Naita(int id)

{

var toode = \_tooted.Find(t => t.Id == id);

return toode != null ? toode : NotFound();

}



[HttpGet("max-hind")] // GET /tooted/max-hind

public ActionResult<Toode> Maxhind()

{

var toode = \_tooted.OrderByDescending(t => t.Price).First();

return toode != null ? toode : NotFound();

}

