```
 1  using System;
 2  using System.Collections.Generic;
 3  using System.Linq;
 4  using System.Threading.Tasks;
 5  using Microsoft.AspNetCore.Http;
 6  using Microsoft.AspNetCore.Mvc;
 7  using Microsoft.EntityFrameworkCore;
 8  using DNPExercises10DataAccess;
 9  using DNPExercises10DataAccess.Data.Entities;
10
11  namespace DNPExercises10DataAccess.Controllers
12  {
13      [Route("api/cats")]
14      [ApiController]
15      public class CatsController : ControllerBase
16      {
17          private readonly ApplicationDbContext _context;
18
19          public CatsController(ApplicationDbContext context)
20          {
21              _context = context;
22          }
23
24          // GET: api/Cats
25          [HttpGet]
26          public IEnumerable<Cat> Getcats()
27          {
28
29              //Query syntax:
30              IEnumerable<Cat> catQuery =
31                  from cats in _context.cats
32                  select cats;
33
34              if (catQuery == null)
35              {
36                  return null;
37              }
38
39              return catQuery;
40
41          }
42
43          // GET: api/Cats/5
44          [HttpGet("{id}")]
45          public IActionResult GetCat([FromRoute] int id)
46          {
47              if (!ModelState.IsValid)
48              {
49                  return BadRequest(ModelState);
50              }
51
52              //Query syntax:
53              //IEnumerable<Cat> catQuery =
```

```
54              //    from cats in _context.cats
55              //    where cats.Id == id
56              //    select cats;
57
58              //Method syntax:
59              IEnumerable<Cat> catQuery = _context.cats.Where(cats =>
                  cats.Id==id);
60
61
62              if (catQuery == null)
63              {
64                  return NotFound();
65              }
66
67              return Ok(catQuery);
68          }
69
70
71          // PUT: api/Cats/5
72          [HttpPut("{id}")]
73          public async Task<IActionResult> PutCat([FromRoute] int id,
              [FromBody] Cat cat)
74          {
75              if (!ModelState.IsValid)
76              {
77                  return BadRequest(ModelState);
78              }
79
80              if (id != cat.Id)
81              {
82                  return BadRequest();
83              }
84
85              _context.Entry(cat).State = EntityState.Modified;
86              try
87              {
88                  await _context.SaveChangesAsync();
89              }
90              catch (DbUpdateConcurrencyException)
91              {
92                  if (!CatExists(id))
93                  {
94                      return NotFound();
95                  }
96                  else
97                  {
98                      throw;
99                  }
100             }
101             return NoContent();
102         }
103
104         // POST: api/Cats
```

```
105         [HttpPost]
106         public async Task<IActionResult> PostCat([FromBody] Cat cat)
107         {
108             if (!ModelState.IsValid)
109             {
110                 return BadRequest(ModelState);
111             }
112 
113             _context.cats.Add(cat);
114             await _context.SaveChangesAsync();
115 
116             return CreatedAtAction("GetCat", new { id = cat.Id }, cat);
117         }
118 
119         // DELETE: api/Cats/5
120         [HttpDelete("{id}")]
121         public async Task<IActionResult> DeleteCat([FromRoute] int id)
122         {
123             if (!ModelState.IsValid)
124             {
125                 return BadRequest(ModelState);
126             }
127 
128             var cat = await _context.cats.FindAsync(id);
129             if (cat == null)
130             {
131                 return NotFound();
132             }
133 
134             _context.cats.Remove(cat);
135             await _context.SaveChangesAsync();
136 
137             return Ok(cat);
138         }
139 
140         private bool CatExists(int id)
141         {
142             return _context.cats.Any(e => e.Id == id);
143         }
144     }
145 }
```