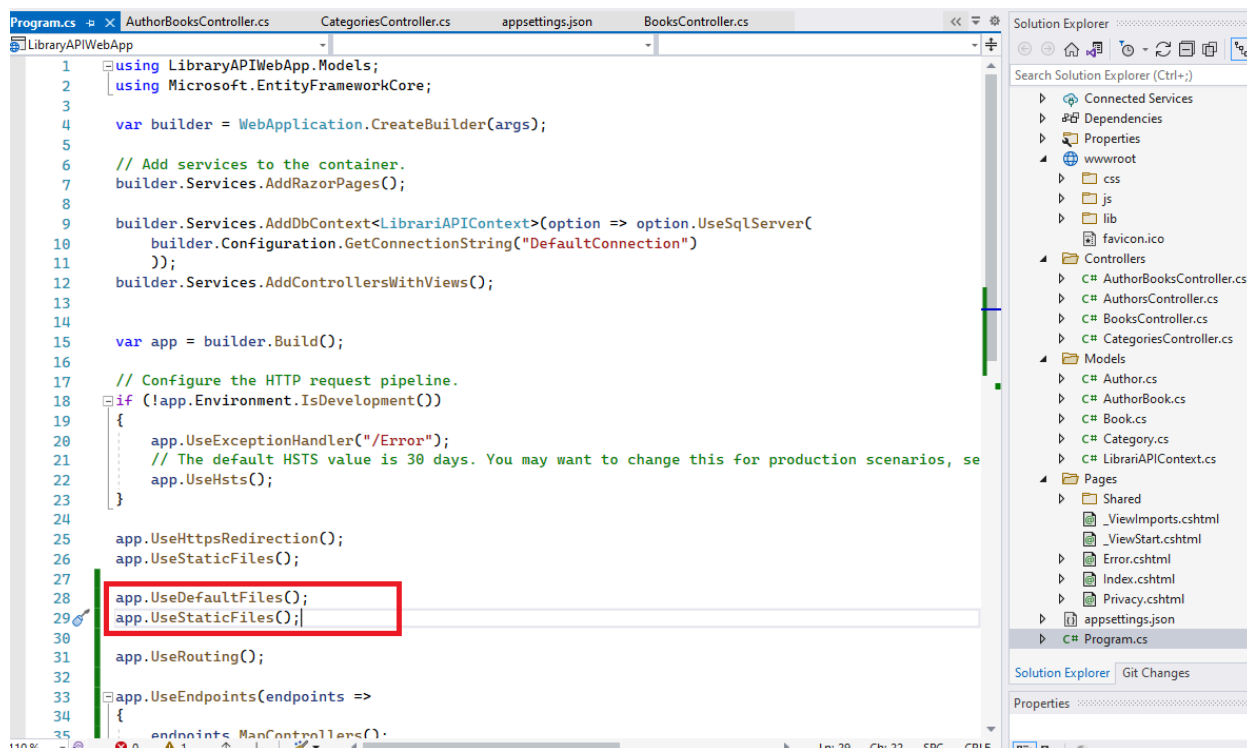


Додамо HTML-сторінку, яка містить форми для створення і адміністрування категорій. Обробники подій приєднуються до елементів на сторінці. При використанні обробників подій створюються запити HTTP до методів дії веб-API. Функція Fetch API fetch ініціює кожен такий запит HTTP.

Функція fetch повертає об'єкт Promise, який містить відповідь HTTP, представлений у вигляді об'єкта Response. Поширеного підходу є отримання тексту відповіді JSON шляхом виклику функції json в об'єкті Response. JavaScript змінює сторінку, використовуючи відомості з відповіді API. Детальніша інформація про функцію fetch (<https://learn.javascript.ru/fetch>).

Налаштуйте в додатку обслуговування статичних файлів і включіть зіставлення файлів за замовчуванням. Вставте в файл Program.cs наступний виділений код:



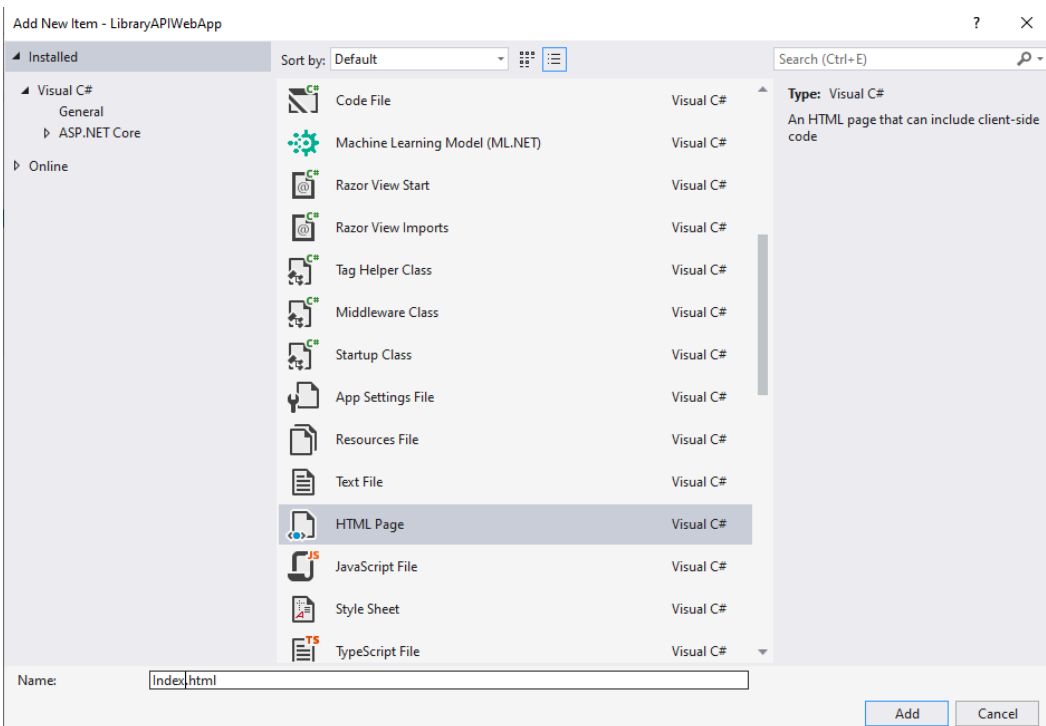
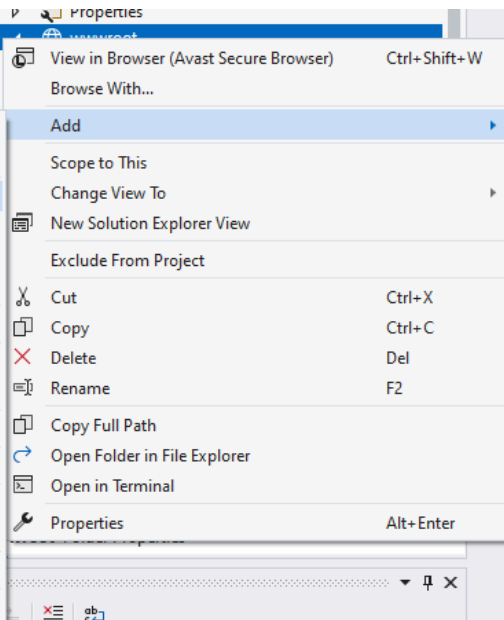
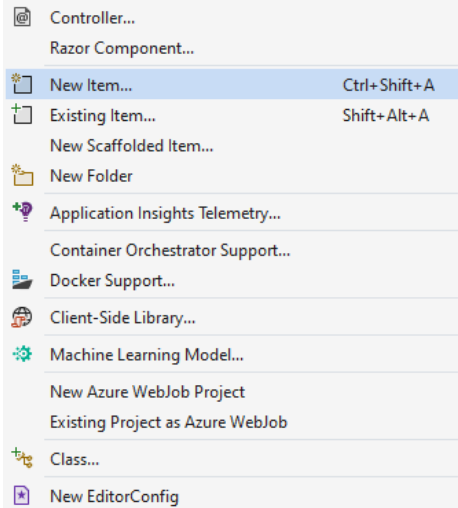
```
1 using LibraryAPIWebApp.Models;
2 using Microsoft.EntityFrameworkCore;
3
4 var builder = WebApplication.CreateBuilder(args);
5
6 // Add services to the container.
7 builder.Services.AddRazorPages();
8
9 builder.Services.AddDbContext<LibrariAPIContext>(option => option.UseSqlServer(
10     builder.Configuration.GetConnectionString("DefaultConnection")
11 ));
12 builder.Services.AddControllersWithViews();
13
14 var app = builder.Build();
15
16 // Configure the HTTP request pipeline.
17 if (!app.Environment.IsDevelopment())
18 {
19     app.UseExceptionHandler("/Error");
20     // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.
21     app.UseHsts();
22 }
23
24 app.UseHttpsRedirection();
25 app.UseStaticFiles();
26
27 app.UseDefaultFiles();
28 app.UseStaticFiles();
29
30 app.UseRouting();
31
32 app.UseEndpoints(endpoints =>
33 {
34     endpoints.MapControllers();
35 });
```

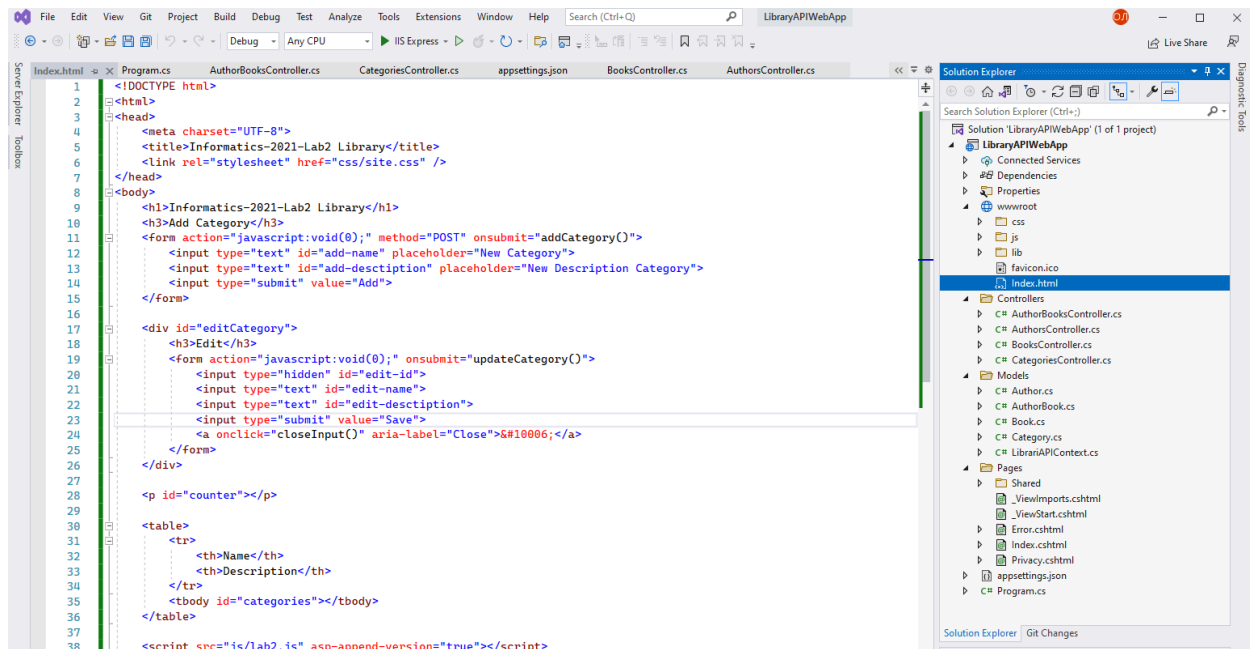
Знайдіть папку wwwroot в кореновому каталозі проекту.

Додайте в неї файл Index.html

```
er.Services.AddRazorPages();
```

```
er.Services.AddDbContext<LibraryAPIApp  
builder.Configuration.GetConnection
```





```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Informatics-2021-Lab2 Library</title>
  <link rel="stylesheet" href="css/site.css" />
</head>
<body>
  <h1>Informatics-2021-Lab2 Library</h1>
  <h3>Add Category</h3>
  <form action="javascript:void(0);" method="POST" onsubmit="addCategory()">
    <input type="text" id="add-name" placeholder="New Category">
    <input type="text" id="add-description" placeholder="New Description Category">
    <input type="submit" value="Add">
  </form>

  <div id="editCategory">
    <h3>Edit</h3>
    <form action="javascript:void(0);" onsubmit="updateCategory()">
      <input type="hidden" id="edit-id">
      <input type="text" id="edit-name">
      <input type="text" id="edit-description">
      <input type="submit" value="Save">
      <a onclick="closeInput()" aria-label="Close">&#10006;</a>
    </form>
  </div>

  <p id="counter"></p>

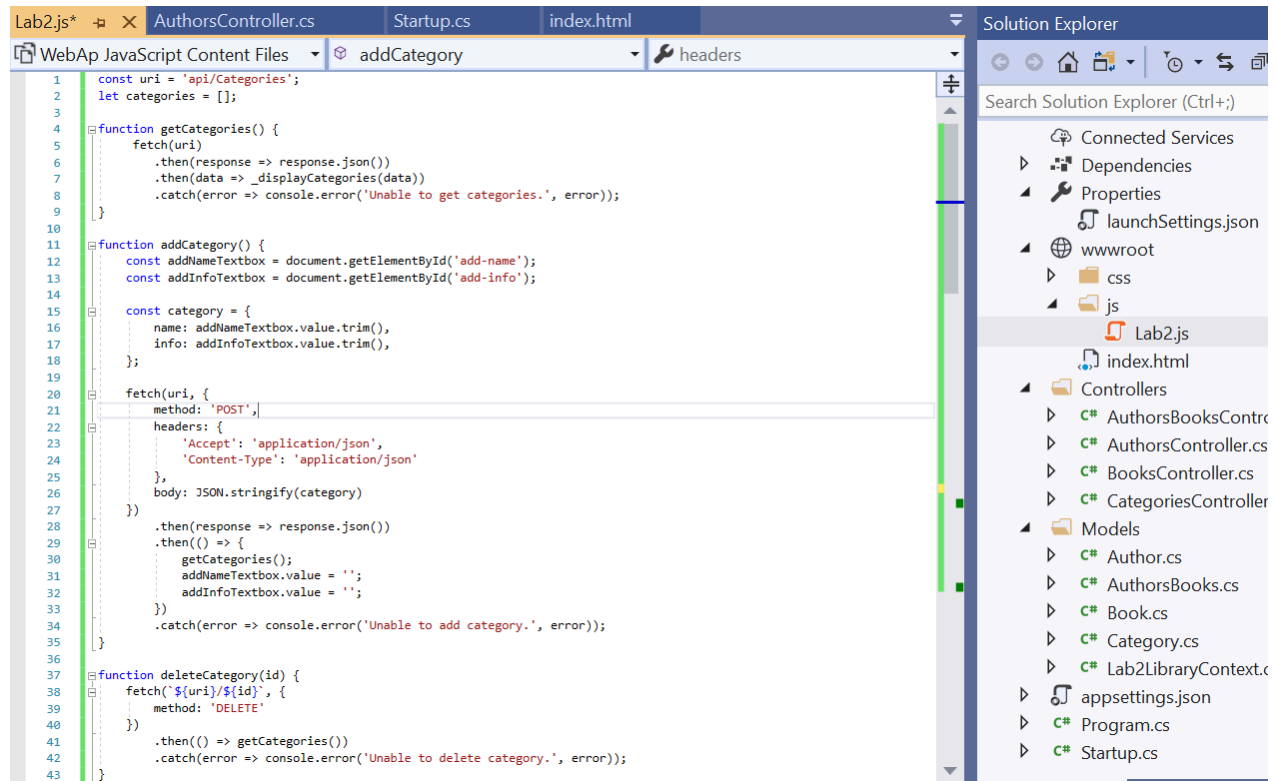
  <table>
    <tr>
      <th>Name</th>
      <th>Description</th>
    </tr>
    <tbody id="categories"></tbody>
  </table>
```

```

<script src="js/lab2.js" asp-append-version="true"></script>
<script type="text/javascript">
    getCategories();
</script>
</body>
</html>

```

Додайте файл JavaScript з іменем lab2.js в папку wwwroot/js:



```

const uri = 'api/Categories';
let categories = [];

function getCategories() {
    fetch(uri)
        .then(response => response.json())
        .then(data => _displayCategories(data))
        .catch(error => console.error('Unable to get categories.', error));
}

function addCategory() {
    const addNameTextbox = document.getElementById('add-name');
    const addInfoTextbox = document.getElementById('add-description');

    const category = {
        name: addNameTextbox.value.trim(),
        info: addInfoTextbox.value.trim(),
    };

    fetch(uri, {

```

```

        method: 'POST',
        headers: {
            'Accept': 'application/json',
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(category)
    })
    .then(response => response.json())
    .then(() => {
        getCategories();
        addNameTextbox.value = '';
        addInfoTextbox.value = '';
    })
    .catch(error => console.error('Unable to add category.', error));
}

function deleteCategory(id) {
    fetch(`${uri}/${id}`, {
        method: 'DELETE'
    })
    .then(() => getCategories())
    .catch(error => console.error('Unable to delete category.', error));
}

function displayEditForm(id) {
    const category = categories.find(category => category.id === id);

    document.getElementById('edit-id').value = category.id;
    document.getElementById('edit-name').value = category.name;
    document.getElementById('edit-description').value = category.info;
    document.getElementById('editForm').style.display = 'block';
}

function updateCategory() {
    const categoryId = document.getElementById('edit-id').value;
    const category = {
        id: parseInt(categoryId, 10),
        name: document.getElementById('edit-name').value.trim(),
        info: document.getElementById('edit-description').value.trim()
    };

    fetch(`${uri}/${categoryId}`, {
        method: 'PUT',
        headers: {
            'Accept': 'application/json',
            'Content-Type': 'application/json'
        },
        body: JSON.stringify(category)
    })
    .then(() => getCategories())
    .catch(error => console.error('Unable to update category.', error));

    closeInput();

    return false;
}

function closeInput() {
    document.getElementById('editForm').style.display = 'none';
}

```

```

function _displayCategories(data) {
  const tBody = document.getElementById('categories');
  tBody.innerHTML = '';

  const button = document.createElement('button');

  data.forEach(category => {
    let editButton = button.cloneNode(false);
    editButton.innerText = 'Edit';
    editButton.setAttribute('onclick', `displayEditForm(${category.id})`);

    let deleteButton = button.cloneNode(false);
    deleteButton.innerText = 'Delete';
    deleteButton.setAttribute('onclick', `deleteCategory(${category.id})`);

    let tr = tBody.insertRow();

    let td1 = tr.insertCell(0);
    let textNode = document.createTextNode(category.name);
    td1.appendChild(textNode);

    let td2 = tr.insertCell(1);
    let textNodeInfo = document.createTextNode(category.description);
    td2.appendChild(textNodeInfo);

    let td3 = tr.insertCell(2);
    td3.appendChild(editButton);

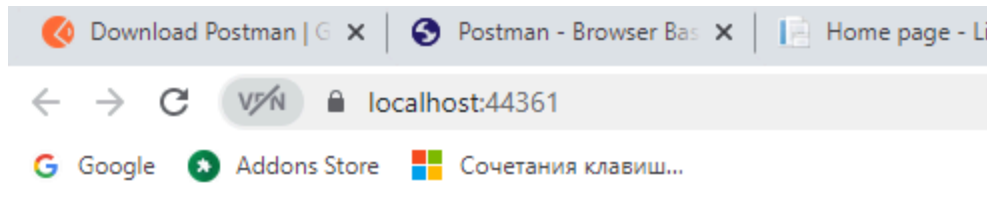
    let td4 = tr.insertCell(3);
    td4.appendChild(deleteButton);
  });

  categories = data;
}

```

За бажання можна додати в папку CSS файл site.css (проте, без цього етап може бути зараховано ☺)

Можна запускати:



Informatics-2021-Lab2 Library

Add Category

Edit

Name	Description		
Логіка	111	<input type="button" value="Edit"/>	<input type="button" value="Delete"/>
Програмування подіол		<input type="button" value="Edit"/>	<input type="button" value="Delete"/>

Для задачі етапу достатньо створити html-сторінку, яка демонструє роботу одного контролера ☺.