

LAB 4

ASP.NET CORE API

❖ CONTENT

- Create API with ASP.NET Core
- Consume API with VueJS

❖ INSTRUCTION

1. Create new VueJS & ASP.NET Core full-stack web application

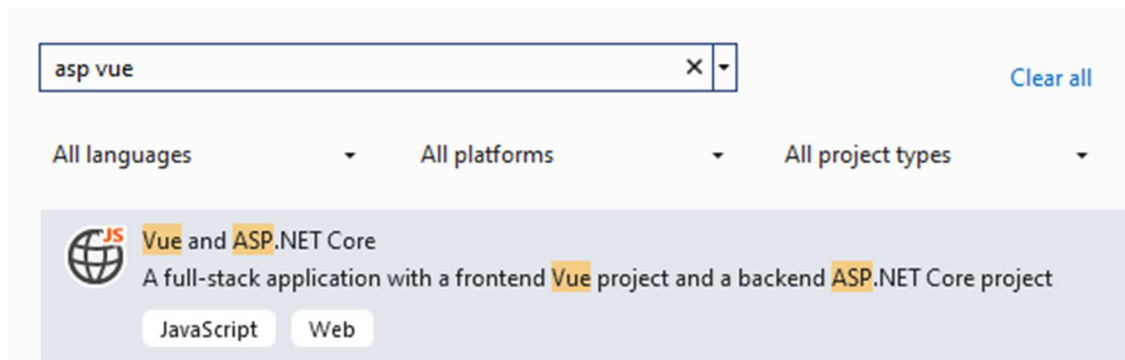


Figure 1 - Create new project (1)

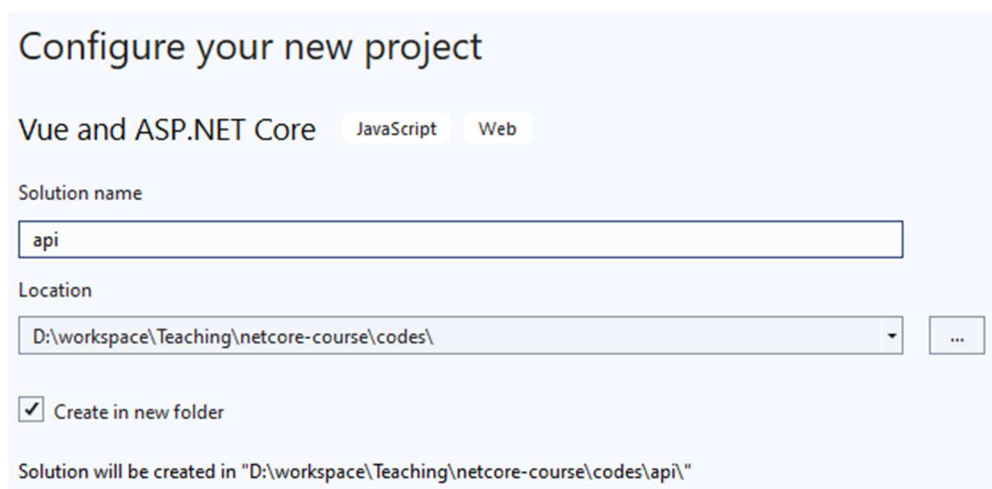


Figure 2 - Create new project (2)

Additional information

Vue and ASP.NET Core JavaScript Web

Framework ⓘ

.NET 8.0 (Long Term Support)

☒ Configure for HTTPS ⓘ

☐ Enable Docker ⓘ

Docker OS ⓘ

Linux

☒ Enable OpenAPI support ⓘ

☐ Do not use top-level statements ⓘ

☒ Use controllers ⓘ

Figure 3 - Create new project (3)

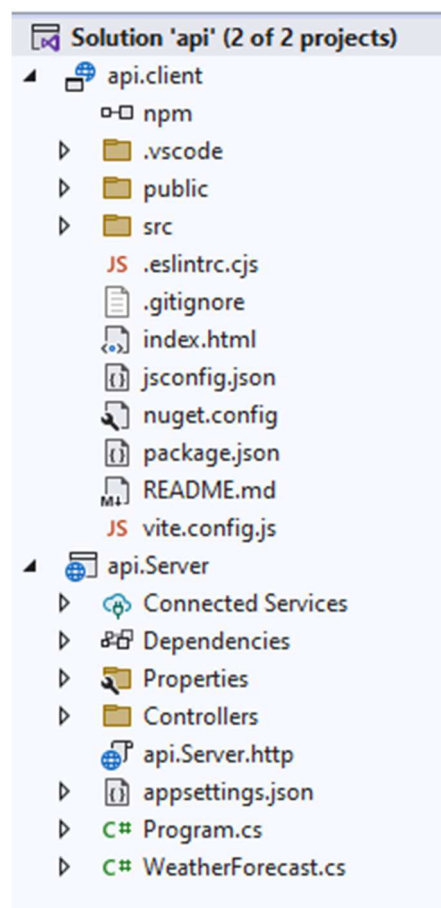


Figure 4 – Web application structure (backend: **api.Server**, frontend: **api.client**)

2. Create new model as table

```
public class Book
{
    public int BookId { get; set; }
    public required string BookTitle { get; set; } //required: compulsory field
    public double BookPrice { get; set; }
    public int BookQuantity { get; set; }
    public string? BookCover { get; set; } //? : nullable field
}
```

Figure 5 – api.Server/Models/Book.cs

3. Generate API Controller by adding new scaffolded item

Add New Scaffolded Item

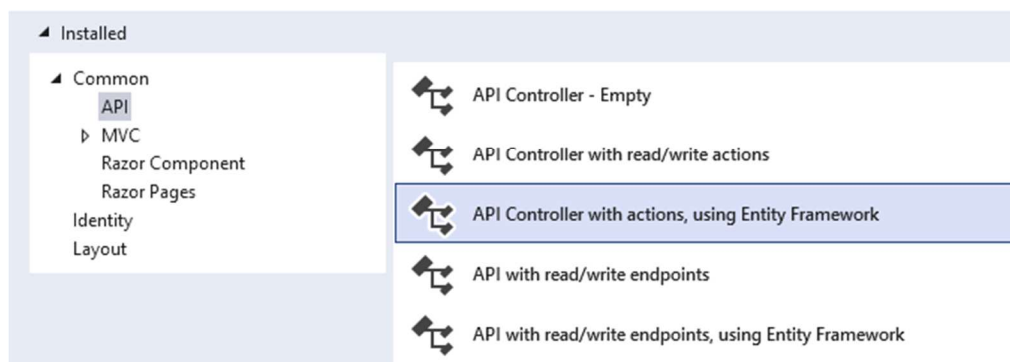


Figure 6 - Generate API Controller with Scaffolding

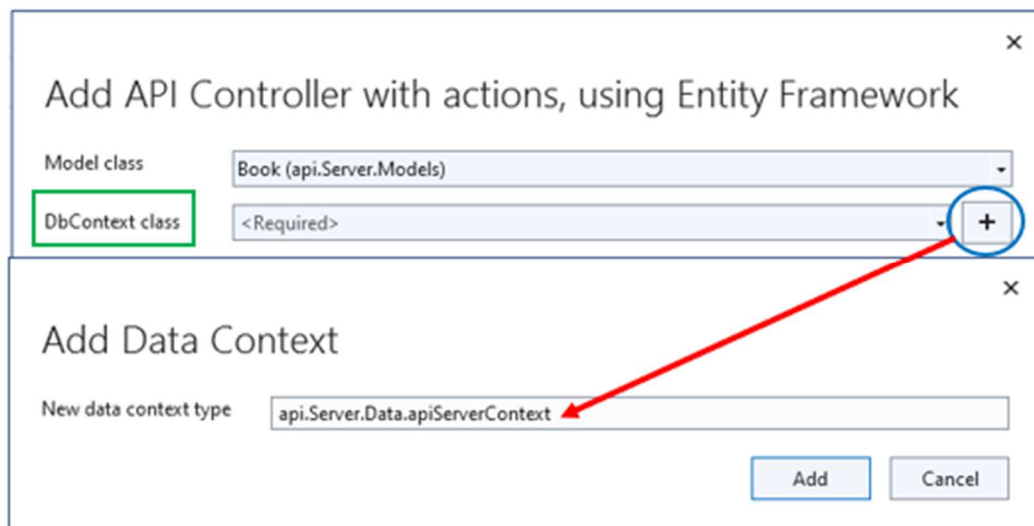


Figure 7 - Add DbContext class (1)

Add API Controller with actions, using Entity Framework

Model class: Book (api.Server.Models)

DbContext class: api.Server.Data.apiServerContext

Database provider: SQL Server

Controller name: BooksController

Add Cancel

Figure 8 - Add DbContext class (2)

4. Seed book data when running project

```
public apiServerContext(DbContextOptions<apiServerContext> options)
    : base(options)
{
}

public DbSet<Book> Book { get; set; }

protected override void OnModelCreating(ModelBuilder builder)
{
    base.OnModelCreating(builder);
    SeedBook(builder);
}
```

Figure 9 – api.Server/Data/apiServerContext.cs (1)

```
private void SeedBook(ModelBuilder builder)
{
    builder.Entity<Book>().HasData(
        new Book
        {
            BookId = 1,
            BookTitle = "The Full Stack Developer",
            BookPrice = 25,
            BookQuantity = 10,
            BookCover = "https://m.media-amazon.com/images/I/61svWgrmTOL._AC_UF1000,1000_QL80_.jpg"
        },
        new Book
        {
            BookId = 2,
            BookTitle = "ASP.NET Core Application Development",
            BookPrice = 35,
            BookQuantity = 20,
            BookCover = "https://m.media-amazon.com/images/I/81iua0Wh34L._AC_UF1000,1000_QL80_.jpg"
        },
        new Book
        {
            BookId = 3,
            BookTitle = "Getting to know Vue.js",
            BookPrice = 45,
            BookQuantity = 30,
            BookCover = "https://media.springernature.com/full/springer-static/cover-hires/book/978-1-4842-3781-6"
        }
    );
}
```

Figure 10 – api.Server/Data/apiServerContext.cs (2)

```

PM> Add-Migration First
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> Update-Database
Build started...
Build succeeded.

```

Figure 11 - Use **PMC** to add migration & update database

5. Config backend API endpoint

```

server: {
  proxy: {
    '^/api': {
      target,
      secure: false
    }
  },
  port: 5173,
  https: {
    key: fs.readFileSync(keyFilePath),
    cert: fs.readFileSync(certFilePath),
  }
}

```

Figure 12 – **api.client/vite.config.js**

6. Import a CSS framework to decorate web UI (*optional*)

```

<title>ASP.NET Core - VueJS Web App</title>
<link rel="stylesheet" type="text/css" href="semantic/dist/semantic.min.css">
<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-QWTKZyjpPEjISv5WaRU90FeRpok6YctnYmDr5pNlyT2bRjXh0JMHjY6hW+ALEwIH"
crossorigin="anonymous">

```

Figure 13 – **api.client/ index.html**

7. Add methods & views to fetch (GET) & delete (DELETE) books

```
<script>
  var api = "/api/books";

  export default {
    data() {
      return {
        books: []
      }
    },
    methods: {
      async fetchBooks() {
        const response = await fetch(api);
        this.books = await response.json();
        console.log(this.books);
      },
      async deleteBook(id) {
        await fetch(api + "/" + id, {
          method: 'DELETE'
        });
        this.fetchBooks();
      }
    },
    created() {
      this.fetchBooks();
    }
  }
</script>
```

Figure 14 - *api.client/src/App.vue (1)*

```

<template>
  <div class="container text-center">
    <table class="table table-striped">
      <thead>
        <tr>
          <th colspan="6" class="text text-primary">
            <h3>Book List</h3>
          </th>
        </tr>
        <tr>
          <th class="text text-success">Id</th>
          <th class="text text-success">Title</th>
          <th class="text text-success">Price</th>
          <th class="text text-success">Quantity</th>
          <th class="text text-success">Cover</th>
          <th class="text text-success">Menu</th>
        </tr>
      </thead>

```

Figure 15 - *api.client/src/App.vue (2)*

```

      <tbody>
        <tr v-for="book in books" :key="book.bookId">
          <th>{{ book.bookId }}</th>
          <td>{{ book.bookTitle }}</td>
          <td>{{ book.bookPrice }}</td>
          <td>{{ book.bookQuantity }}</td>
          <td>
            
          </td>
          <td>
            <button @click="deleteBook(book.bookId)"
              class="btn btn-danger"
              onclick="return confirm('Are you sure to delete this book ?')">
              Delete
            </button>
          </td>
        </tr>
      </tbody>
    </table>
  </div>
</template>

```

Figure 16 - *api.client/src/App.vue (3)*

8. Check for final results

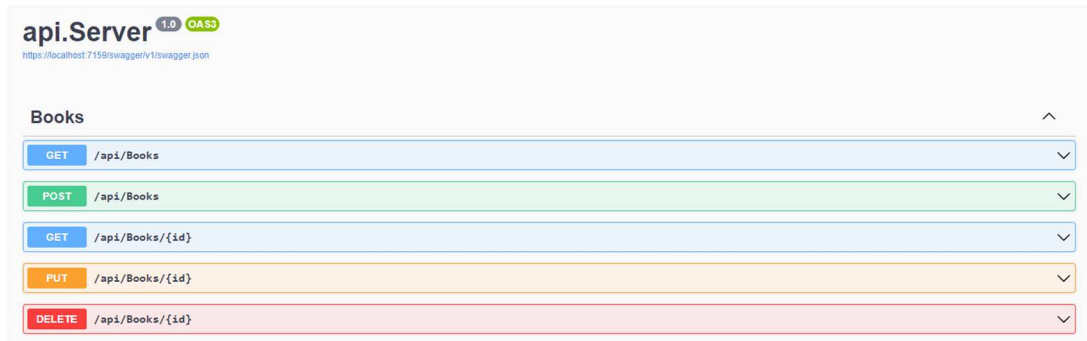


Figure 17 - Test API with Swagger

Book List

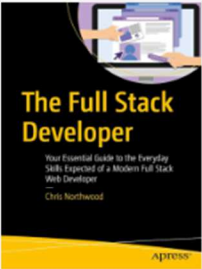
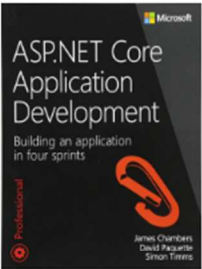
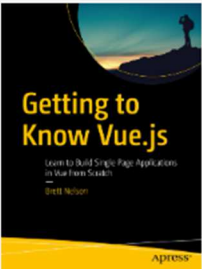
Id	Title	Price	Quantity	Cover	Menu
1	The Full Stack Developer	\$25	10		Delete
2	ASP.NET Core Application Development	\$35	20		Delete
3	Getting to know Vue.js	\$45	30		Delete

Figure 18 - Consume API in Vue