**.NET Core for the backend and Vue.js for the frontend:**

**Backend (API using .NET Core):**

1. Set up the project:
   * Create a new ASP.NET Core Web API project using your preferred development environment.
2. Define the Todo model:
   * Create a new class called **Todo** with properties such as **Id**, **Title**, **Description**, **Completed**, etc.
   * Add the necessary attributes and data annotations to the properties for validation purposes.
3. Create a restful API.
4. Configure the CORS policy:
   * Set up the CORS policy in the **Startup.cs** file to allow requests from the frontend application's domain.
5. Connect to a database.
6. Implement the repository or service:
   * Create a repository or service class that interacts with the database.
   * Define methods for CRUD operations and implement them using the database context.
7. Test the API endpoints:
   * Use tools like Postman or curl to test the API endpoints and ensure they are working correctly.

**Frontend (Vue.js Composition API):**

1. Set up the project:
   * Install Node.js and npm (Node Package Manager) if not already installed.
   * Create a new Vue.js project using the Vue CLI by running the command: **Vue create to-do-app**.
   * Choose the default configuration or customize it based on your preferences.
2. Define the components:
   * Create components for displaying the to-do list, adding a new to-do, editing a to-do, etc.
   * Set up the necessary data properties, methods, and event handlers within each component.
3. Create API service:
   * Create a JavaScript file (e.g., **api.js**) to handle API requests using the **axios** library.
   * Define methods for making HTTP requests to the backend API endpoints (GET, POST, PUT, DELETE).
4. Build the frontend UI:
   * Design and implement the user interface using HTML, CSS, and Vue.js template syntax.
   * Use the created components and the API service to fetch and display the to-dos.
   * Implement forms and input fields for adding/editing to-dos.
5. Connect the UI with the API service:
   * Call the API service methods within the Vue component methods and event handlers.
   * Update the UI based on the API responses.
6. Test the application:
   * Start the development server using the command: **npm run serve**.
   * Open the application in a web browser and test adding, editing, deleting, and marking to-dos as completed.

Remember to install the necessary dependencies for both the backend (using NuGet) and the frontend (using npm or yarn). Also, adapt the project structure and code according to your specific requirements and preferences.