Project name: backend .net api

Author: Oliver Tallo

Creation Date: 20-Jul-2024

Last Updated:

Version: 1.0

1. **Title, Subject, Last Updated Date, Reference Number**, **and** **Version** are marked by a Word Bookmark so that they can be easily reproduced in the header and footer of documents. When you change any of these values, be careful not to accidentally delete the bookmark. **You can make bookmarks visible by selecting Tools->Options…View and checking the Bookmarks option in the Show region.**

Contents

1. **Introduction**
2. **Design Overview**

* Technical Overview
* Functional Overview

1. **Test Run of a .NET Web API Endpoints through Swagger UI**

* Set MedDashboardAPI project in Visual Studio 2022 as a Startup Project
* Accessing Endpoints through Swagger UI



## Introduction

A backend .NET API that keeps track of patients, with attributes describing the patient's first name, last name, city, and active status. The data will be stored in memory for portability. Every time the app starts, the in-memory database will be empty. It has the ability to add, update, retrieve, filter, and sort the results.

## Design Overview

1. Web API for adding a new patient.
2. Web API for updating an existing patient.
3. Web API for deleting a patient.
4. Web API for getting all patients.
5. Web API that retrieves a page, filtered, and sorted list of patients.

### Technical Overview

This document defines the technical components for the Project i.e., Tools used, DB, Programming language etc., Frontend and methodology used.

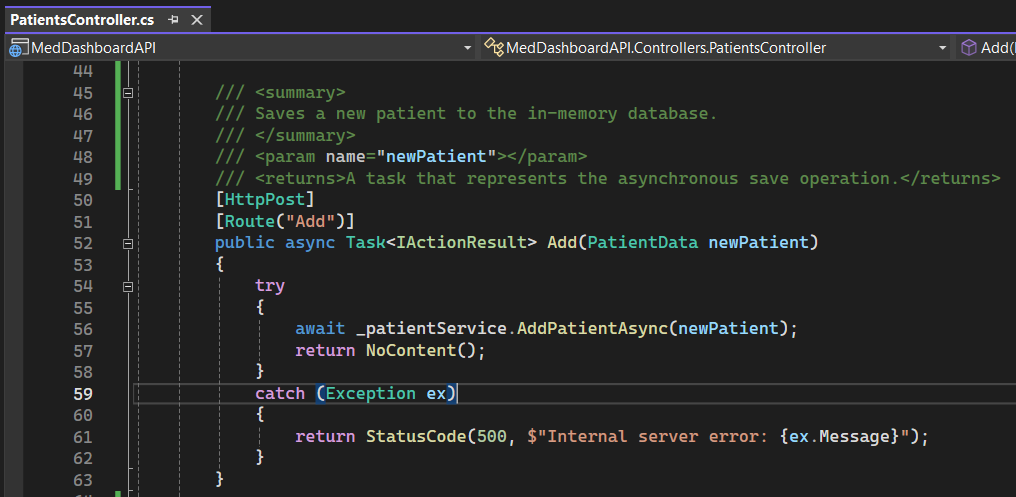
| Technology/Tool Name | Description (Frontend/Backend) | Details |
| --- | --- | --- |
|  |  |  |
| **Visual Studio 2022** | Backend: **In-Memory database** | .ASP.NET Core Web API (.NET 8.0)  Controllers, Models, Data, Interface, Repositories, and Services |
|  | Frontend: **Swagger UI** | <https://localhost:7020/swagger/index.html> (Endpoints: Get, Add, Update, Delete, and GetPaged) |
|  |  |  |
|  |  |  |
|  |  |  |

### Functional Overview

1. **Web API for adding a new patient:** Used for adding a new patient with attributes describing the patient's first name, last name, city, and active status. The data will be stored in memory for portability.

Note: Every time the app starts, the in-memory database will be empty.

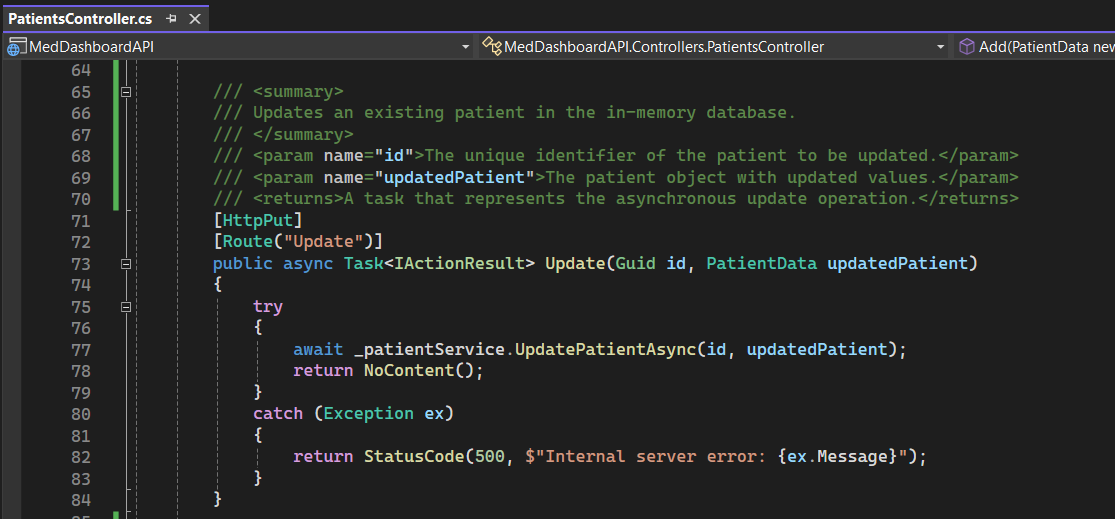
**The screenshot below refers to the API which saves a new patient to the in-memory database.**



1. **Web API for updating an existing patient:** Used for updating an existing patient. A patient’s ID will be passed to get that record to be updated. And the available patient’s data that can be changed are patient’s first name, last name, city, and active status.

Note: Every time the app starts, the in-memory database will be empty.

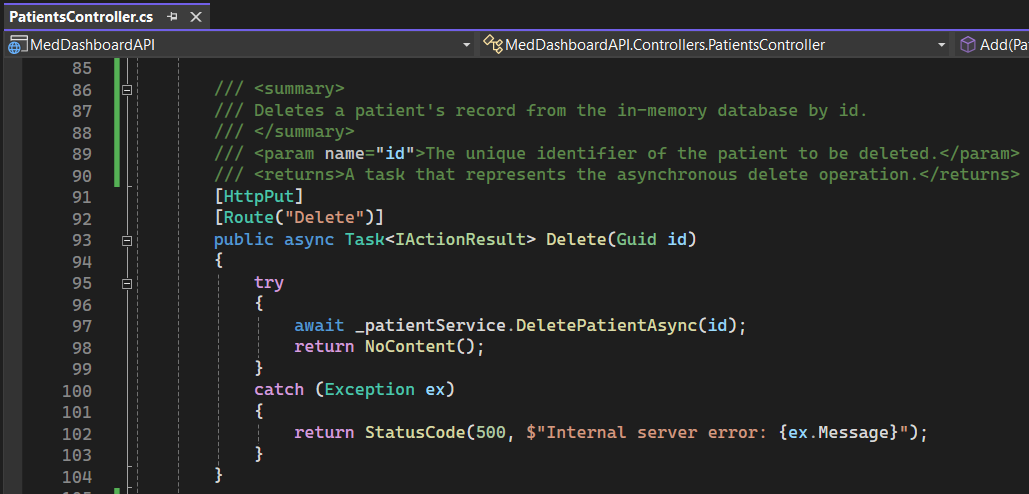
**The screenshot below refers to the API which updates an existing patient’s record.**



1. **Web API for deleting a patient:** Used for deleting a patient. A patient’s ID will be passed to delete a particular patient’s record.

Note: Every time the app starts, the in-memory database will be empty.

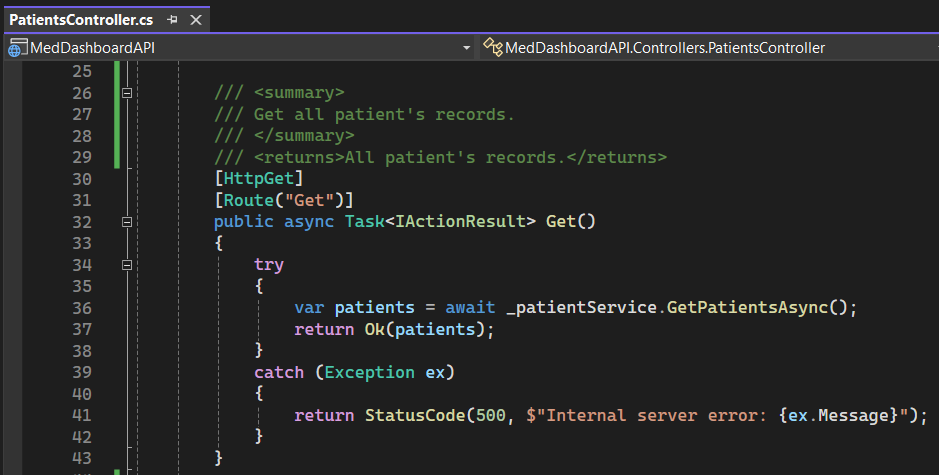
**The screenshot below refers to the API which deletes a patient’s record.**



1. **Web API for getting all patient records:** All patient records will be retrieved.

Note: Every time the app starts, the in-memory database will be empty.

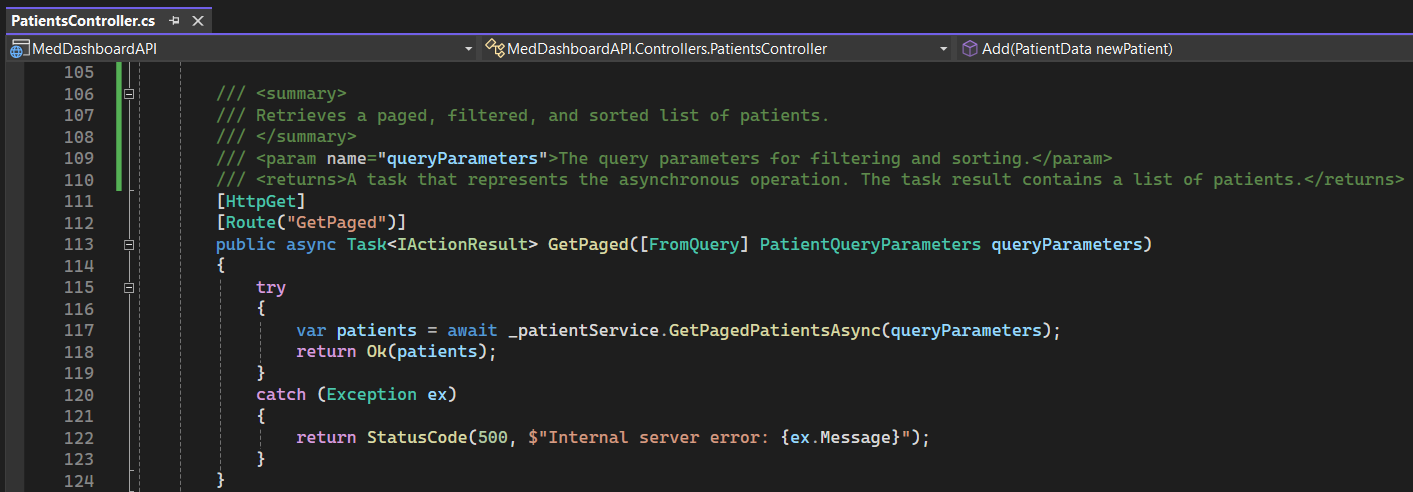
**The screenshot below refers to the API which retrieves all patient records.**



1. **Web API for getting filtered patient records:** Patient records are filtered and sorted either in ascending or descending order.

Note: Every time the app starts, the in-memory database will be empty.

**The screenshot below refers to the API which retrieves filtered and sorted patient records.**



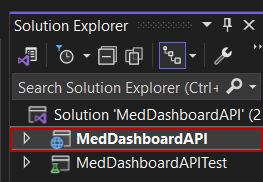
1. If the Functional Design included a Technical Overview section, duplicate that information here and add additional details.

## Test Run of a .NET Web API Endpoints through Swagger UI

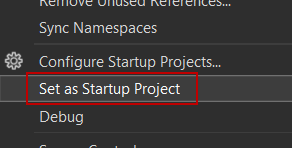
1. Web API for adding a new patient (/api/Patients/Add)
2. Web API for getting all patients (/api/Patients/Get)
3. Web API for updating an existing patient (/api/Patients/Update)
4. Web API that retrieves a page, filtered, and sorted list of patients. (/api/Patients/GetPaged)
5. Web API for deleting a patient (/api/Patients/Delete)

### Set MedDashboardAPI Project in Visual Studio 2022 as a Startup Project

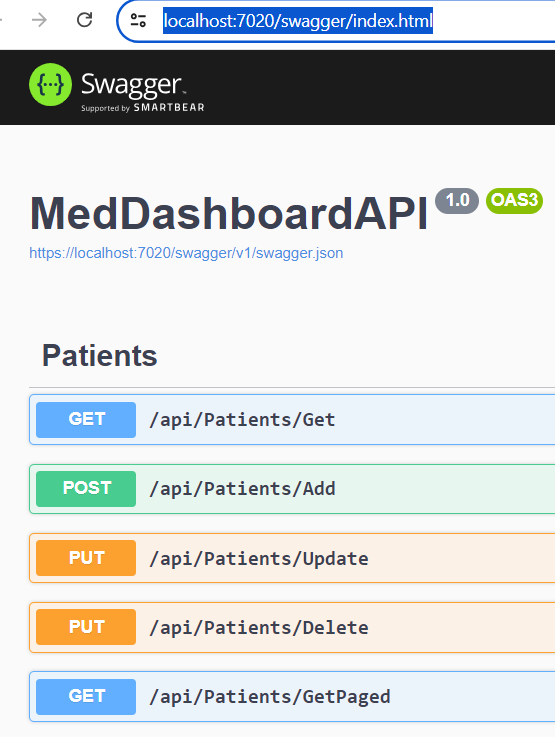
1. **MedDashboardAPI as a Startup Project:** Set as a Startup Project from the Solution Explorer. And click “F5” to execute the project. The Swagger UI will display (https://localhost:7020/swagger/index.html).
2. Right-click on “MedDashboardAPI”



1. Click on “Set as Startup Project”

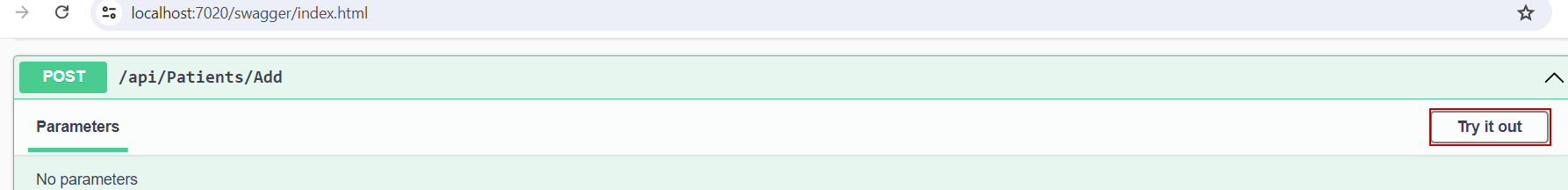


1. Press “F5” key to execute the project. This will display the Swagger UI.

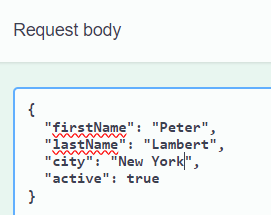


### Accessing Endpoints through Swagger UI

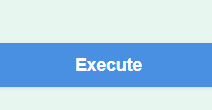
1. **/api/Patients/Add:** Add new patient
2. Click “Try it out”



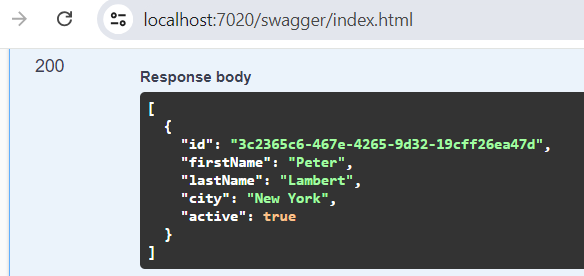
1. In the Request Body, fill up the firstName, lastName, city, and active fields.



1. Click “Execute” button for saving into the In-Memory database.

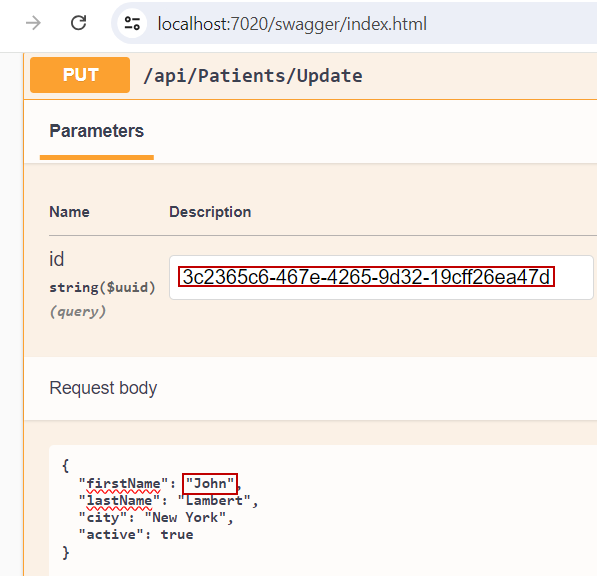


1. **/api/Patients/Get:** Get all patient records.
2. Click “Try it out”
3. Click “Execute” button to retrieve all patient records.

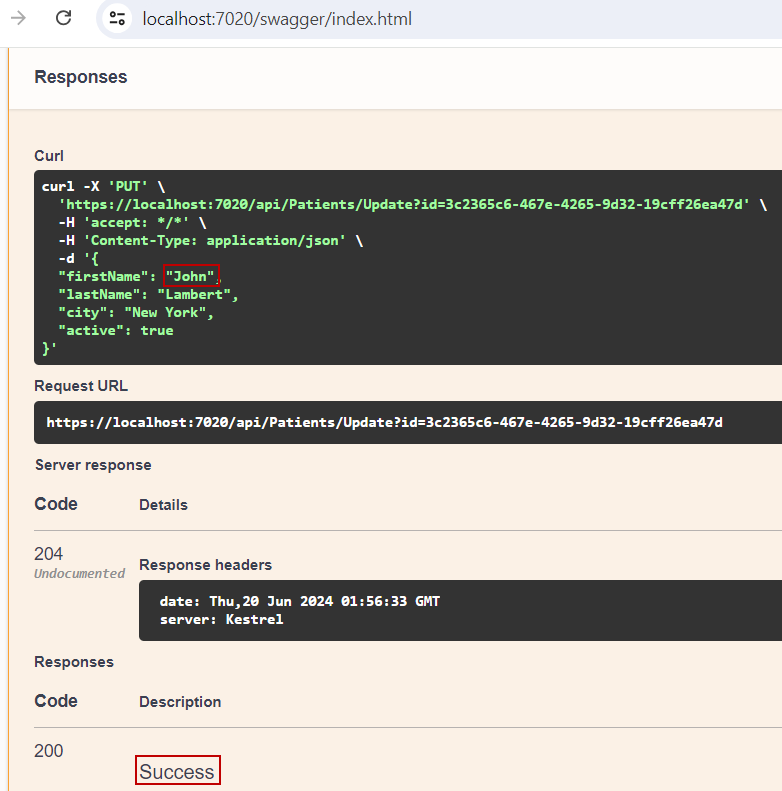


Note: Copy the id 3c2365c6-467e-4265-9d32-19cff26ea47d

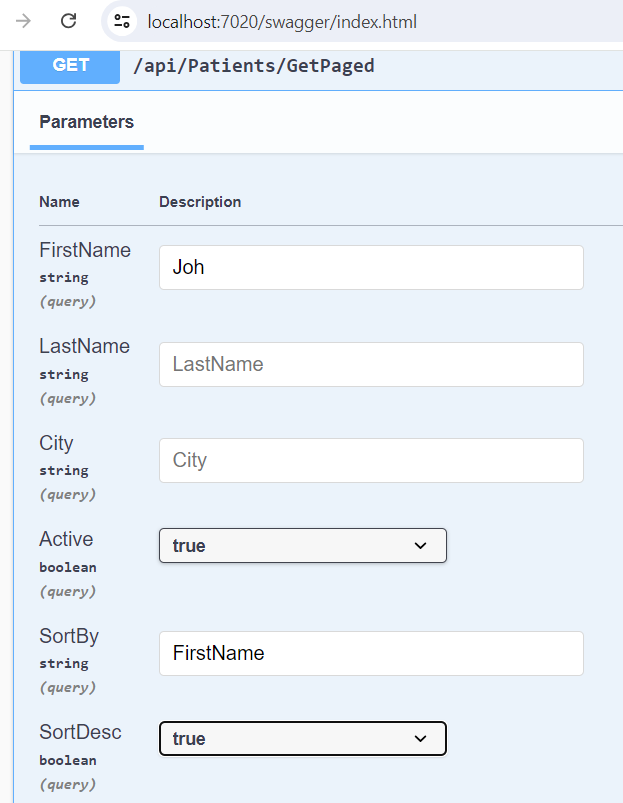
1. **/api/Patients/Update:** Update an existing patient by id.
2. Click “Try it out”
3. Paste the ID into the ID input field, change “Peter” to “John”, and click “Execute” button to update.



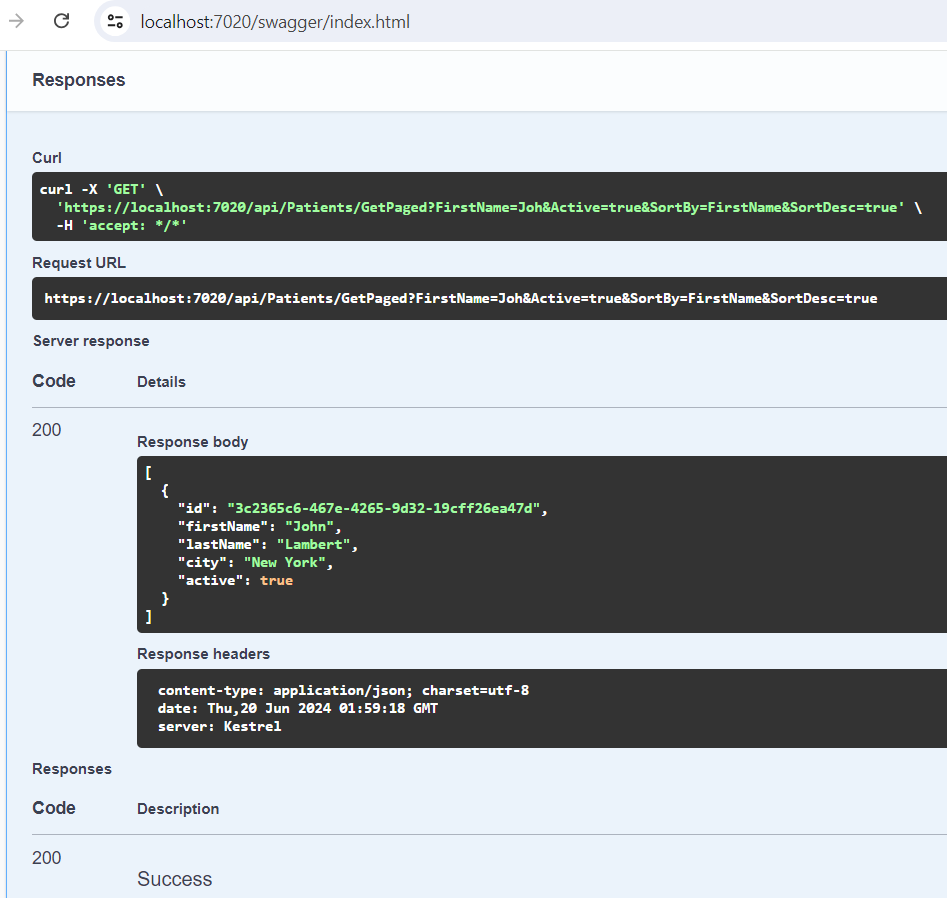
1. Result



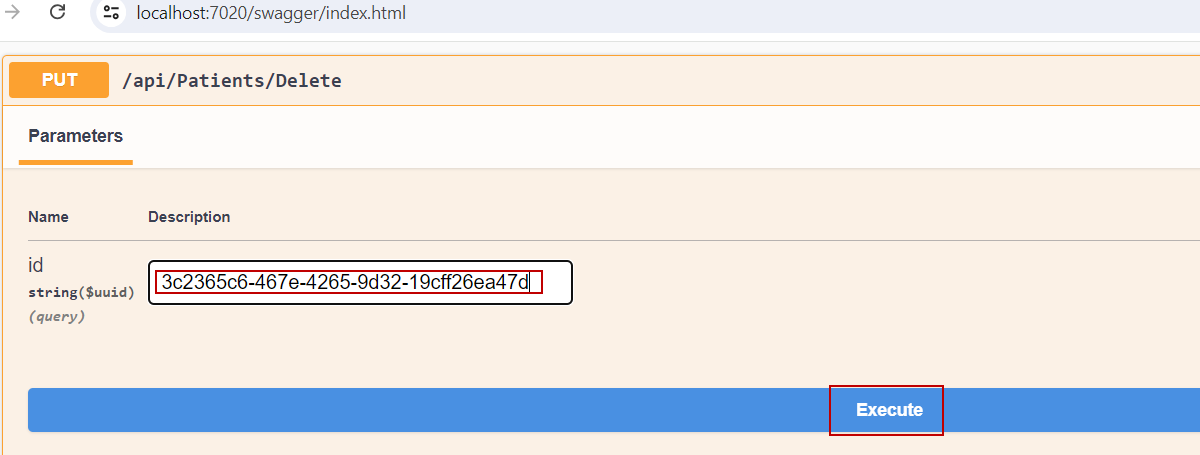
1. **/api/Patients/GetPaged:** Getting a patient record using a filter with an option for sorting records.
2. Click “Try it out”
3. Fill up the filter and sort fields



1. Result



1. **/api/Patients/Delete:** Deleting patient record by Id
2. Click “Try it out”
3. Paste the ID into the ID input field and click “Execute” button to delete.



1. Result

