

MỤC TIÊU:

Kết thúc bài thực hành này bạn có khả năng

- ✓ Cài đặt postman tool và thực hiện kiểm thử crud
- ✓ Tạo ứng dụng Web APIs - ASP.NET Core
- ✓ Sử dụng postman kiểm thử Web APIs - ASP.NET Core

Bài 1 (3 điểm) Cài đặt Postman và thực hiện các thao tác cơ bản crud với Api Ulr: <https://jsonplaceholder.typicode.com/users>

Bài 2 (4 điểm): Tạo ứng dụng Web Api Asp.net core áp dụng DI design pattern cung cấp các chức năng Crud cho model sau:

```
public class Reservation
{
    public int Id { get; set; }
    public string Name { get; set; }
    public string StartLocation { get; set; }
    public string EndLocation { get; set; }
}
```

Gợi ý thiết kế interface

```
public interface IRepository
{
    //tra ve all Reservations
    2 references
    IEnumerable<Reservation> Reservations { get; }
    //tra ve Reservation theo id
    2 references
    Reservation this[int id] { get; }
    //them mới Reservations
    4 references
    Reservation AddReservation(Reservation reservation);
    //cập nhật Reservation
    2 references
    Reservation UpdateReservation(Reservation reservation);
    //xóa Reservation theo id
    2 references
    void DeleteReservation(int id);
}
```

Gợi ý lớp implement interface

```

public class Repository : IRepository
{
    private Dictionary<int, Reservation> items;
    0 references
    public Repository()
    {
        items = new Dictionary<int, Reservation>();
        new List<Reservation> {
            new Reservation {Id=1, Name = "Thepv",
                StartLocation = "Ha Noi", EndLocation="Can Tho" },
            new Reservation {Id=2, Name = "Fpoly",
                StartLocation = "Sai Gon", EndLocation="Tay Nguyen" }

        }.ForEach(r => AddReservation(r));
    }
    2 references
    public Reservation this[int id] => items.ContainsKey(id) ? items[id] : null;

    2 references
    public IEnumerable<Reservation> Reservations => items.Values;

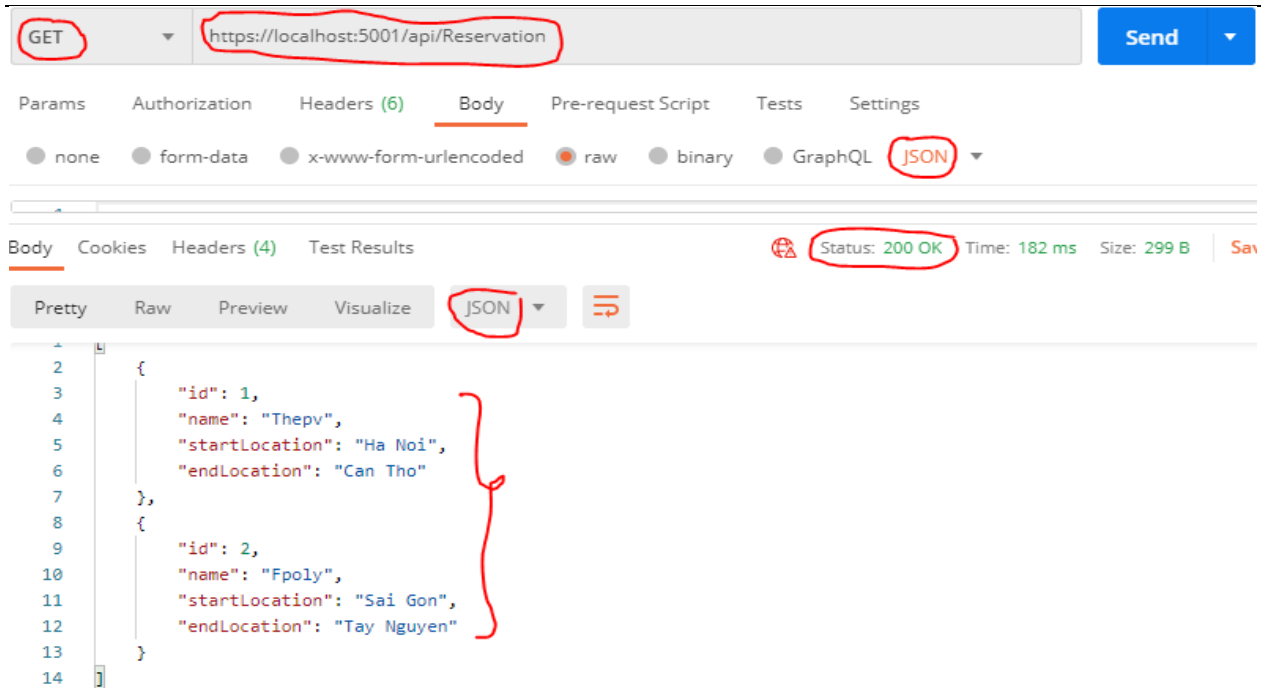
    public Reservation AddReservation(Reservation reservation)
    {
        if (reservation.Id == 0)
        {
            int key = items.Count;
            while (items.ContainsKey(key)) { key++; };
            reservation.Id = key;
        }
        items[reservation.Id] = reservation;
        return reservation;
    }
    2 references
    public void DeleteReservation(int id) => items.Remove(id);
    2 references
    public Reservation UpdateReservation(Reservation reservation) =>
        AddReservation(reservation);
    }
    }
    
```

Gợi	ý	viết	controller
-----	---	------	------------

```

[ApiController]
[Route("api/[controller]")]
1 reference
public class ReservationController : ControllerBase
{
    private IRepository repository;
    0 references
    public ReservationController(IRepository repo) => repository = repo;
    [HttpGet]
    0 references
    public IEnumerable<Reservation> Get() => repository.Reservations;
    [HttpGet("{id}")]
    1 reference
    public ActionResult<Reservation> Get(int id)
    {
        if (id == 0)
            return BadRequest("Value must be passed in the request body.");
        return Ok(repository[id]);
    }
    [HttpPost]
    0 references
    public Reservation Post([FromBody] Reservation res) =>
        repository.AddReservation(new Reservation
        {
            Name = res.Name,
            StartLocation = res.StartLocation,
            EndLocation = res.EndLocation
        });
    [HttpPut]
    0 references
    public Reservation Put([FromBody] Reservation res) => repository.UpdateReservation(res);
    [HttpPatch("{id}")]
    0 references
    public StatusCodeResult Patch(int id, [FromBody] JsonPatchDocument<Reservation> patch)
    {
        var res = (Reservation)((OkObjectResult)Get(id).Result).Value;
        if (res != null)
        {
            patch.ApplyTo(res);
            return Ok();
        }
        return NotFound();
    }
    [HttpDelete("{id}")]
    0 references
    public void Delete(int id) => repository.DeleteReservation(id);
    
```

Gợi ý sử dụng postman kiểm tra các Api



The screenshot shows a Postman interface for a GET request to `https://localhost:5001/api/Reservation`. The request is successful, returning a `200 OK` status. The response body is displayed in JSON format, showing a list of two reservation objects. Red circles highlight the `GET` method, the URL, the `JSON` format selection, and the `200 OK` status. A red bracket groups the two JSON objects in the response body.

```

{
  "id": 1,
  "name": "Thepv",
  "startLocation": "Ha Noi",
  "endLocation": "Can Tho"
},
{
  "id": 2,
  "name": "Fpoly",
  "startLocation": "Sai Gon",
  "endLocation": "Tay Nguyen"
}

```

Bài 3 (3điểm)

Tạo ứng dụng Web Api Asp.net core áp dụng DI design pattern cung cấp các chức năng Crud sao cho

- Các chức năng CRU được gọi theo url: <https://localhost:5000/api/product>
- Dữ liệu trả về từ việc đọc Api có định dạng json như hình

JSON	Raw Data	Headers
Save	Copy	
▼ 0:		
ProductName:	"ERP Solution Package"	
Price:	60000	
▼ 1:		
ProductName:	"Pendrive"	
Price:	1000	
▼ 2:		
ProductName:	"Denim-Men"	
Price:	1000	
▼ 3:		
ProductName:	"Denim-Women"	
Price:	1000	
▼ 4:		
ProductName:	"Pringles"	
Price:	200	

- c. Chức năng Delete được gọi theo url : <https://localhost:5000/api/product/id>