TEST CASES

Business API Tests

Postman API Testing Guide

Environment Setup

First, create a new environment in Postman with these variables:

- `base_url`: http://localhost:5000
- `token`: (leave empty initially, will be filled after login)

JSON Data:

Test:

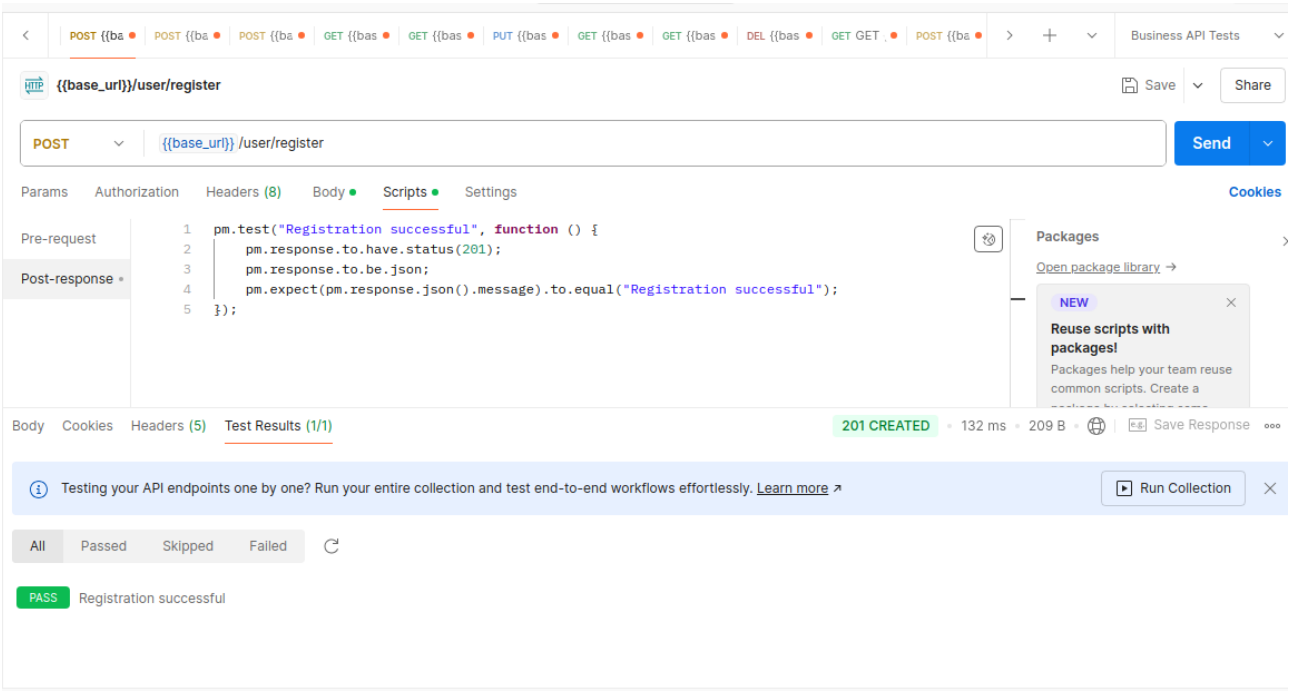
1. User Registration Test

POST {{base_url}}/user/register

Content-Type: application/json

```
{
   "username": "testuser",
   "password": "test123"
}
```

```
// Test Scripts
pm.test("Registration successful", function () {
   pm.response.to.have.status(201);
   pm.response.to.be.json;
   pm.expect(pm.response.json().message).to.equal("Registration successful");
});
```

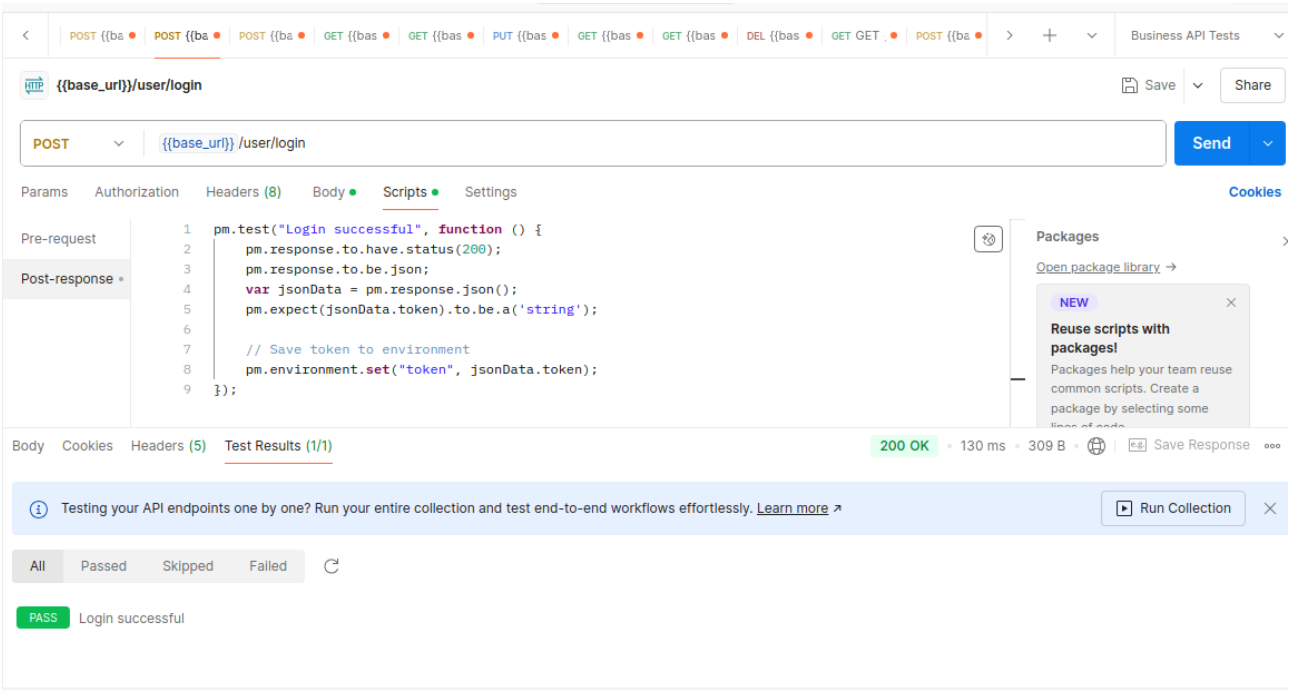


2. User Login Test

POST {{base_url}}/user/login
Content-Type: application/json

```
{
   "username": "testuser",
   "password": "test123"
}
```

```
// Test Scripts
pm.test("Login successful", function () {
   pm.response.to.have.status(200);
   pm.response.to.be.json;
   var jsonData = pm.response.json();
   pm.expect(jsonData.token).to.be.a('string');

   // Save token to environment
   pm.environment.set("token", jsonData.token);
});
```



3. Add Business Test

POST {{base_url}}/businesses
Authorization: Bearer {{token}}
Content-Type: application/json

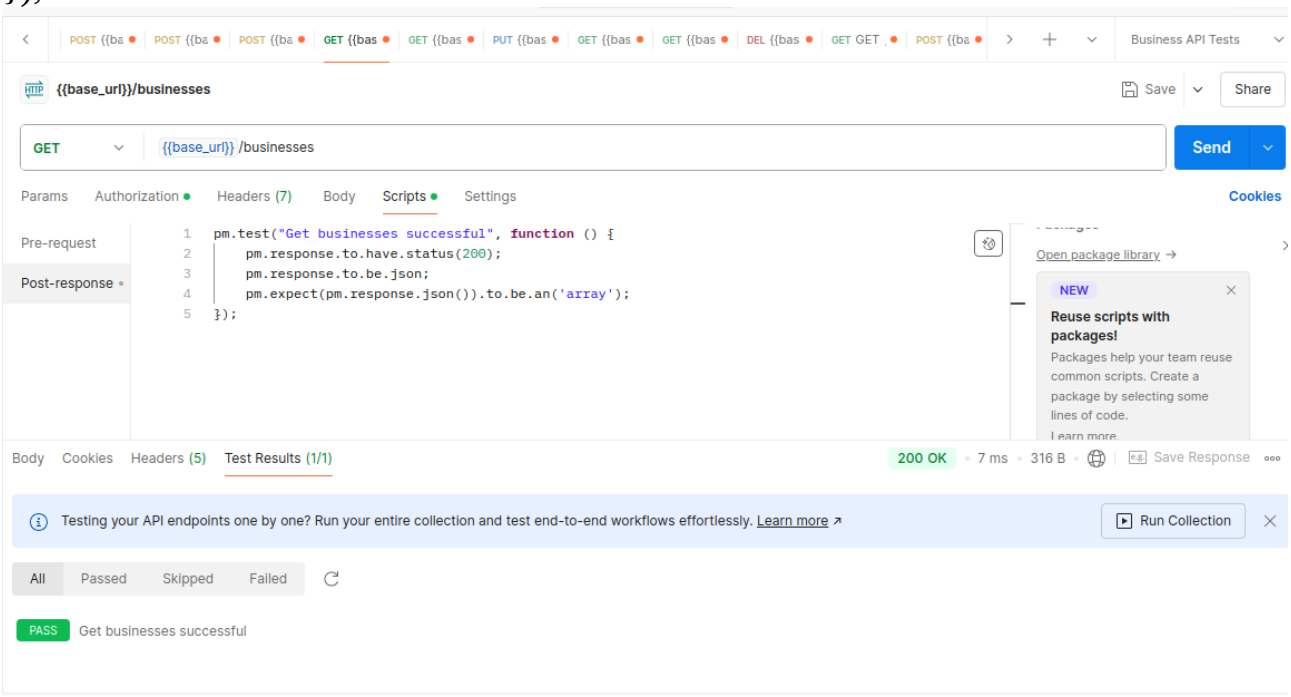
```
{
   "name": "Test Business",
   "address": "123 Test Street",
   "type": "Retail"
}
```

```
// Test Scripts
pm.test("Business added successfully", function () {
   pm.response.to.have.status(201);
   pm.response.to.be.json;
   pm.expect(pm.response.json().message).to.equal("Business added");
});

// Save business ID for later tests
if (pm.response.status === 201) {
   pm.environment.set("business_id", pm.response.json()._id);
}
```

4. Get All Businesses Test

GET {{base_url}}/businesses
Authorization: Bearer {{token}}

// Test Scripts
pm.test("Get businesses successful", function () {
pm.response.to.have.status(200);
pm.response.to.be.json;
pm.expect(pm.response.json()).to.be.an('array');
});



5. Get Single Business Test

GET {{base_url}}/businesses/{{business_id}}
Authorization: Bearer {{token}}

// Test Scripts
pm.test("Get single business successful", function () {
    pm.response.to.have.status(200);
    pm.response.to.be.json;
    pm.expect(pm.response.json()).to.have.property('name');
    pm.expect(pm.response.json()).to.have.property('address');
    pm.expect(pm.response.json()).to.have.property('type');
});

6. Update Business Test

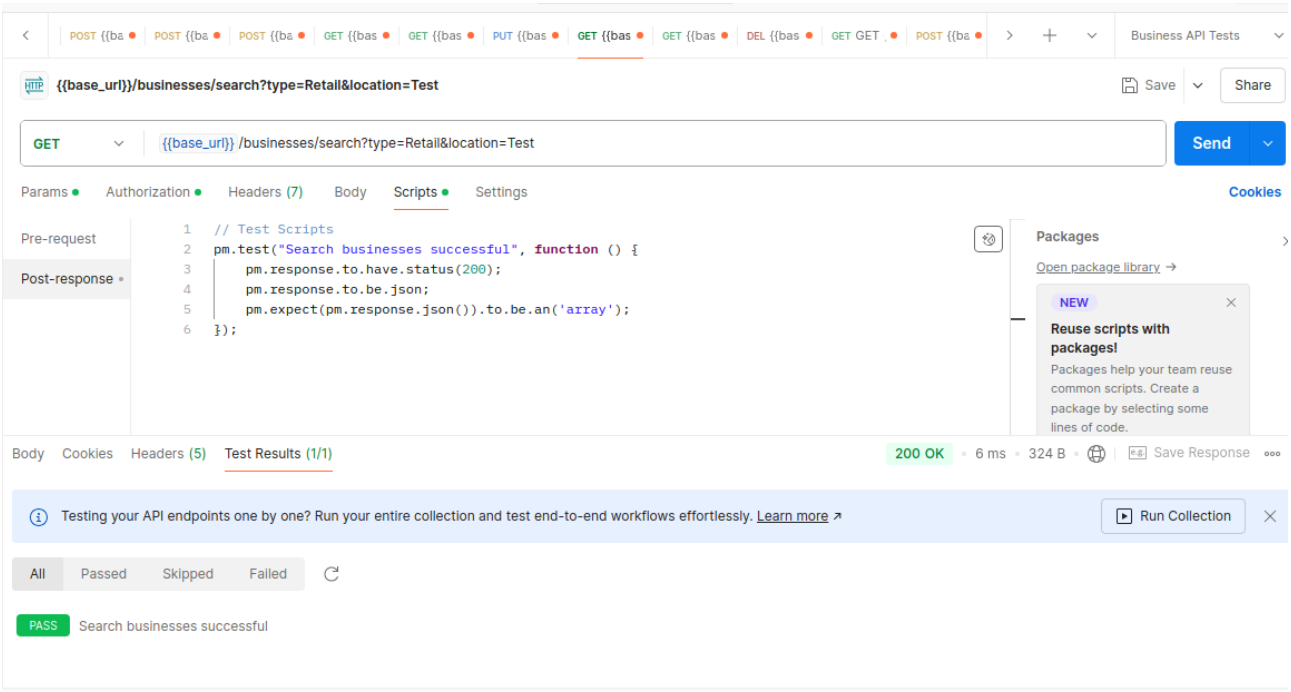
PUT {{base_url}}/businesses/{{business_id}}
Authorization: Bearer {{token}}
Content-Type: application/json

{
    "name": "Updated Test Business",

```
    "address": "456 Test Avenue",
    "type": "Retail"
}

// Test Scripts
pm.test("Update business successful", function () {
    pm.response.to.have.status(200);
    pm.response.to.be.json;
    pm.expect(pm.response.json().message).to.equal("Business updated");
});
```

7. Search Businesses Test

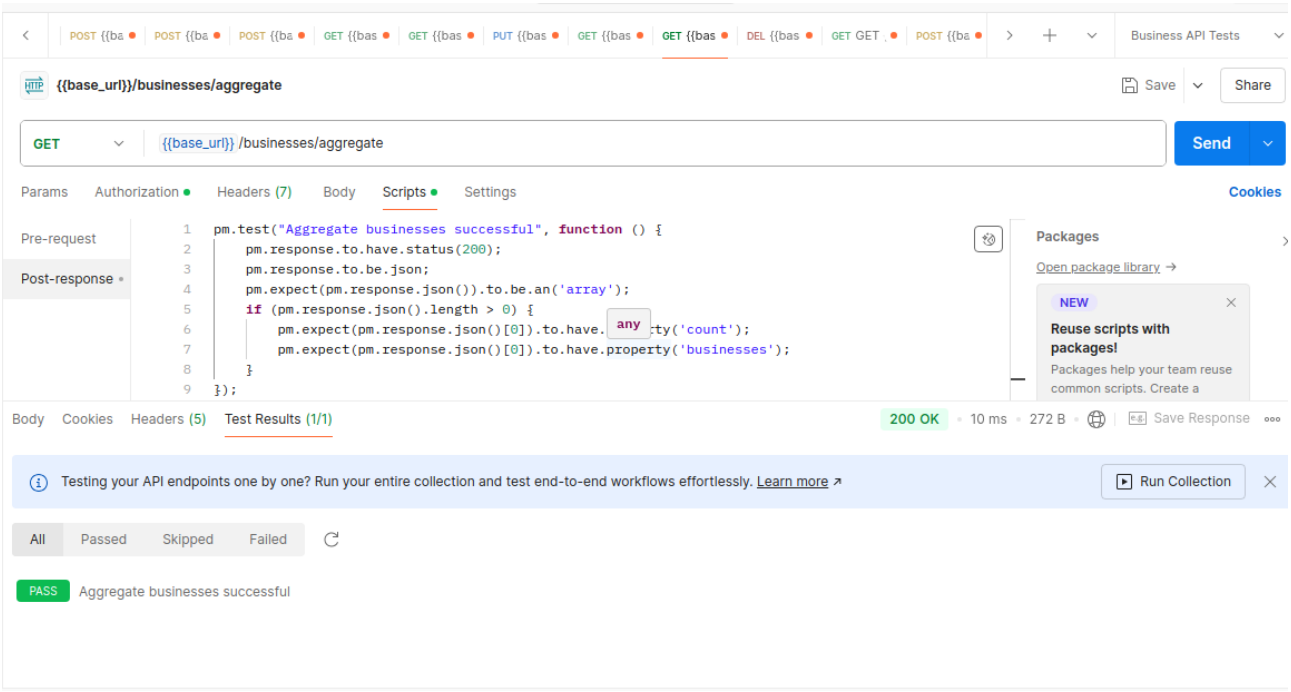
```
GET {{base_url}}/businesses/search?type=Retail&location=Test
Authorization: Bearer {{token}}

// Test Scripts
pm.test("Search businesses successful", function () {
    pm.response.to.have.status(200);
    pm.response.to.be.json;
    pm.expect(pm.response.json()).to.be.an('array');
});
```



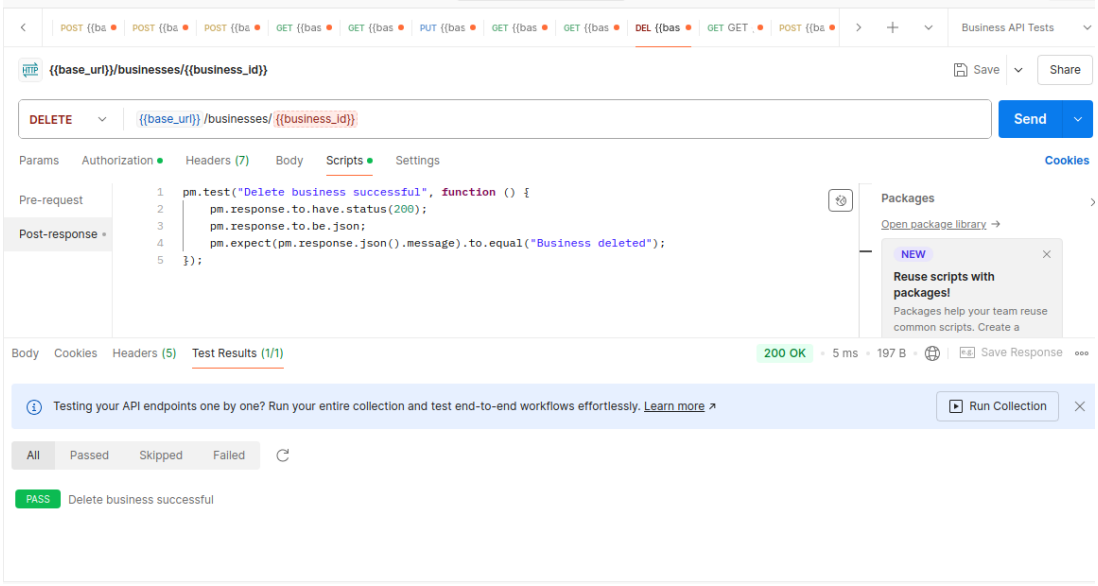
8. Aggregate Businesses Test

GET {{base_url}}/businesses/aggregate
Authorization: Bearer {{token}}

```
// Test Scripts
pm.test("Aggregate businesses successful", function () {
    pm.response.to.have.status(200);
    pm.response.to.be.json;
    pm.expect(pm.response.json()).to.be.an('array');
    if (pm.response.json().length > 0) {
        pm.expect(pm.response.json()[0]).to.have.property('count');
        pm.expect(pm.response.json()[0]).to.have.property('businesses');
    }
});
```



9. Delete Business Test

DELETE {{base_url}}/businesses/{{business_id}}
Authorization: Bearer {{token}}

```
// Test Scripts
pm.test("Delete business successful", function () {
    pm.response.to.have.status(200);
    pm.response.to.be.json;
    pm.expect(pm.response.json().message).to.equal("Business deleted");
});
```