Note-
Inside fixture block only we need to have the visit url whether its api or ui.


it("Appraoch2- Dynamically generating json object", ()⇒{
const requestBody={
//math.random returns number between 0 to 1.
//to string - we can pass how many chars needed.
tourist_name: Math.random().toString(5).substring(2),
tourist_email: Math.random().toString(5)
.substring(2)+"@gmail.com",
tourist_location: "Paris"
}
cy.request(
{
method: 'POST',
url: 'http://restapi.adequateshop.com/api/Tourist' ,
body:requestBody
})
.then((response) ⇒{
expect(response.status).to.eq(201)
expect(response.body.tourist_name)
.to.eq(requestBody.tourist_name)
expect(response.body.tourist_email)
.to.eq(requestBody.tourist_email)
expect(response.body.tourist_location)
.to.eq(requestBody.tourist_location)
})
})
codesnap.dev

```
it.only("Approach3- using Fixture", ()=>{

            cy.fixture('tourist').then( (data)=>{
             const requestBody=data;

             cy.request(
                {
                method: 'POST',
                url: 'http://restapi.adequateshop.com/api/Tourist' ,
                //this needed when your website is blocked for launch by
                //click on this to verify if your human.
                failOnStatusCode: false,
                body:requestBody

                })
                .then( (response) =>{
                    expect(response.status).to.eq(201)
                    expect(response.body.tourist_name)
.to.eq(requestBody.tourist_name)
                    expect(response.body.tourist_email)
.to.eq(requestBody.tourist_email)
                    expect(response.body.tourist_location)
.to.eq(requestBody.tourist_location)

                    //validate property name and its value.
                    //two ways to use it.
                    expect(response.body).has
.property('tourist_email',requestBody.tourist_email)
                    expect(response.body).to.
have.property('tourist_email',requestBody.tourist_email)

                })

            })
        } )


    })
```

codesnap.dev

Test data for fixture-

With or without fixture extension it works the same way-

postrequestwithfixture.cy.js | postrequestwithfixture_1.cy.js

cypress > e2e > postrequestwithfixture_1.cy.js > describe('api testing') callback > it('use fixture to generate data') callback

```
/// <reference types="cypress" />

describe('api testing', () => {

    it('use fixture to generate data', () => {

      //lets use fixture without extension.
      //it works with or without extension.
       cy.fixture("touristdetails"). then((data)=>{
           const requestbody=data;
```

```
        const requestbody=data;

     cy.request(
         {
             method: 'post',
             url: 'http://restapi.adequateshop.com/api/Tourist',
             failOnStatusCode: false,
             body: requestbody
         })
```

```
18                  })
19                  .then((response)=>{
20                      expect(response.status).to.equal(201)
21                      expect(response.body.tourist_name).to.eq(requestbody.tourist_name)
22                      expect(response.body.tourist_email).to.eq(requestbody.tourist_email)
23                      expect(response.body.tourist_location).to.eq(requestbody.tourist_location)
24
```

```
24
25                      //validate property name and its value.
26                      //another two ways to check.
27                      expect(response.body).has.property("tourist_email", requestbody.tourist_email)
28                      expect(response.body).to.have.property("tourist_email", requestbody.tourist_email)
29                  })
30              })
31          })
32  })
```