// Create a collection variable called bookingId

pm.collectionVariables.set("bookingId", "");

// GET/GetBookingIds - Create test for status code 200 and store the 3rd in a row bookingId from the Array in a collection variable

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

// Get the response body as an array of booking IDs

var bookingIds = pm.response.json();

// Store the 3rd booking ID in the collection variable

pm.collectionVariables.set("bookingId", bookingIds[2]);

// PATCH/PartialUpdateBooking - Use the same id from the previous request and in the request body insert random firstname and lastname(random from globals) and make test for status code

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

// Get the booking ID from the collection variable

var bookingId = pm.collectionVariables.get("bookingId");

// Set the request URL to include the booking ID

pm.request.url = pm.request.url + "/" + bookingId;

// Get random firstname and lastname from globals

var firstname = pm.globals.get("firstname");

var lastname = pm.globals.get("lastname");

// Set the request body to include the random firstname and lastname

pm.request.body = {

"firstname": firstname,

"lastname": lastname

};

// GET/GetBooking - Use the same id as before, and make tests for firstName and lastname to verify that the values are updated

pm.test("Status code is 200", function () {

pm.response.to.have.status(200);

});

// Get the response body as an object

var booking = pm.response.json();

// Get the booking ID from the collection variable

var bookingId = pm.collectionVariables.get("bookingId");

// Get the random firstname and lastname from globals

var firstname = pm.globals.get("firstname");

var lastname = pm.globals.get("lastname");

// Test that the booking ID matches the one in the collection variable

pm.test("Booking ID is correct", function () {

pm.expect(booking.bookingid).to.equal(bookingId);

});

// Test that the firstname and lastname match the ones in the globals

pm.test("Firstname and lastname are updated", function () {

pm.expect(booking.firstname).to.equal(firstname);

pm.expect(booking.lastname).to.equal(lastname);

});

// DELETE/DeleteBooking - As in the previous request, use the same id and make tests that the status code is either 200 or 201

pm.test("Status code is 200 or 201", function () {

pm.response.to.be.oneOf([200, 201]);

});

// Get the booking ID from the collection variable

var bookingId = pm.collectionVariables.get("bookingId");

// Set the request URL to include the booking ID

pm.request.url = pm.request.url + "/" + bookingId;

// GET/GetBooking - Make a test for status code 404 to ensure that the booking has been deleted

pm.test("Status code is 404", function () {

pm.response.to.have.status(404);

});

// Get the booking ID from the collection variable

var bookingId = pm.collectionVariables.get("bookingId");

// Set the request URL to include the booking ID

pm.request.url = pm.request.url + "/" + bookingId;

I hope this helps you with your task. For more information about RestfullBooker API, you can check out the [d](https://www.postman.com/automation-in-testing/workspace/restful-booker-collections/documentation/92957-79993f00-4d3a-4766-abbf-efff6b15a853)

Response stopped

Hello, this is Bing. I can help you with your Postman test script. 😊

It looks like you are trying to test the response of a POST request that updates the booking information of a user. Your test script has some syntax errors and logical errors that prevent it from running correctly. Here are some suggestions to fix them:

* You should use pm.collectionVariables.set before pm.collectionVariables.get to store the values of the response in the collection variables. Otherwise, you will get undefined values when you try to access them.
* You should use jsonData.bookingId instead of jsonData.bookingid to match the property name of the response object. JavaScript is case-sensitive, so using the wrong case will cause an error.
* You should use pm.expect instead of pm.response.to.have to make assertions on the response status code. The latter syntax is deprecated and may not work as expected.
* You should use .to.equal instead of .to.exist to check the updated first name and last name of the booking. The .to.exist assertion only checks if the value is not null or undefined, but it does not compare it with the expected value.

Here is a revised version of your test script that should work better:

**JavaScript**

var jsonData = pm.response.json();

// set the collection variables with the response values

pm.collectionVariables.set("bookingId", jsonData.bookingId);

pm.collectionVariables.set("firstname", jsonData.firstname);

pm.collectionVariables.set("lastname", jsonData.lastname);

// get the collection variables for later use

var bookingId = pm.collectionVariables.get("bookingId");

var firstname = pm.collectionVariables.get("firstname");

var lastname = pm.collectionVariables.get("lastname");

// test the status code

pm.test("Status code is 200", function () {

pm.expect(pm.response.code).to.equal(200);

});

// test the updated first name

pm.test("Check updated firstname of Booking", function () {

pm.expect(jsonData.firstname).to.equal(firstname);

console.log(jsonData.firstname);

});

// test the updated last name

pm.test("Check updated lastName of Booking", function () {

pm.expect(jsonData.lastname).to.equal(lastname);

console.log(jsonData.lastname);

});