**Scenario 1: New User Registers, Browses Books, and Places an Order**

Flow:

**Check API Status:**

Send a GET request to /status to ensure the API is operational.

Verify the response indicates the API is "available."

**Register API Client:**

Send a POST request to /api-clients/ with the following request body:

{

"clientName": "Ha",

"clientEmail": "bruh@example.com"

}

Verify the response contains an access token: eb400a2b5238875f218acbcf6c880cad0113ade57302099afcbcd298abb93d0b

Store the access token for subsequent authenticated requests.

**Browse Available Books:**

Send a GET request to /books to retrieve a list of all available books.

Optionally, use query parameters to filter by type (e.g., type=fiction) or limit the number of results (e.g., limit=5).

Verify the response contains a list of books in the expected format.

Present the list of books to the user for selection.

**View Details of a Specific Book:**

Let the user select a book from the list.

Send a GET request to /books/:bookId with the selected bookId.

Verify the response contains detailed information about the book, including its title, type, and availability.

Display the detailed information to the user.

**Submit an Order:**

After the user decides to order the book, send a POST request to /orders with the following request body:

{

  "books": [

    {"bookId": 1},

    {"bookId": 2},

    {"bookId": 3},

    {"bookId": 4}

  ],

  "customerName": "Lmao"

}

{

"bookId": 6,

"customerName": "Lmao"

}

Include the Authorization header with the token obtained earlier:

Authorization: Bearer <YOUR TOKEN>

Verify the response contains the order ID, confirming the order was successfully placed.

Display a confirmation message to the user with the order details.

**View All Orders:**

Send a GET request to /orders to retrieve a list of all orders placed by the user.

Verify the response contains the recently placed order.

Present the list of orders to the user for reference.

|  |  |
| --- | --- |
| Test case ID | TC\_API\_001 |
| Test case title | Verify API status endpoint returns a successful response. |
| Test description | Ensure the /status endpoint confirms that the API is operational. |
| Pre-conditions | None |
| Test steps | HTTP Method: GET  Endpoint: /status  Headers: None  Request Body: None  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: {"status": "available"}  Response Headers: Standard HTTP headers |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_001.1 |
| Test case title | Verify response when the API is unavailable. |
| Test description | Ensure the /status endpoint returns the correct error response when the API is down. |
| Pre-conditions | The API is intentionally stopped or unavailable. |
| Test steps | HTTP Method: GET  Endpoint: /status  Headers: None  Request Body: None  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 503 Service Unavailable  Response Body: {"error": "Service Unavailable"}  Response Headers: Standard HTTP headers |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_002 |
| Test case title | Verify a new API client can register successfully. |
| Test description | Ensure a user can register and receive a valid access token. |
| Pre-conditions | None |
| Test steps | HTTP Method: POST  Endpoint: /api-clients/  Headers: Content-Type: application/json  Request Body:  {  "clientName": "Ha",  "clientEmail": "bruh@example.com"  }  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: Contains an access\_token  Response Headers: Standard HTTP headers. |
| Post-conditions | Ensure the access\_token is saved for subsequent requests. |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_002.1 |
| Test case title | Verify error when required fields are missing in registration. |
| Test description | Ensure the API returns an error if clientName or clientEmail is missing in the registration payload. |
| Pre-conditions | None |
| Test steps | HTTP Method: POST  Endpoint: /api-clients/  Headers: Content-Type: application/json  Request Body:  {  "clientEmail": "lmao@example.com"  }  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 400 Bad Request  Response Body: {"error": "Invalid or missing client name/email."}  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_002.3 |
| Test case title | Verify error when email in registration is already in use by another user. |
| Test description | Ensure the API returns an error if email is duplicated. |
| Pre-conditions | None |
| Test steps | HTTP Method: POST  Endpoint: /api-clients/  Headers: Content-Type: application/json  Request Body:  {  "clientEmail": "valentin@example.com"  }  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 409 Conflict  Response Body: {"error": "API client already registered. Try a different email."}  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_003 |
| Test case title | Verify the list of books is retrieved successfully without query params. |
| Test description | Ensure a user can browse available books using the /books endpoint. |
| Pre-conditions | None |
| Test steps | HTTP Method: GET  Endpoint: /books  Headers: None  Request Body: None  Parameters:  Optional: type=fiction  Optional: limit=20 |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: A list of books with valid properties (bookId, name, type).  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_003.1 |
| Test case title | Verify the list of books is retrieved successfully using both valid query params. |
| Test description | Ensure a user can browse available books using the /books endpoint. |
| Pre-conditions | None |
| Test steps | HTTP Method: GET  Endpoint: /books  Headers: None  Request Body: None  Parameters:  type=fiction  limit=20 |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: A list of books with valid properties (bookId, name, type).  Response Headers: Standard HTTP headers (Content-Type: application/json). |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_003.2 |
| Test case title | Verify error when query parameter is invalid. |
| Test description | Ensure the API returns an error if type has an invalid value or limit exceeds the allowed range (1–20). |
| Pre-conditions | None |
| Test steps | HTTP Method: GET  Endpoint: /books?limit=50  Headers: None  Request Body: None  Parameters: Query Parameter: limit=0, 21 / type: fixion |
| Input data | N/A |
| Expected results | Status Code: 400 Bad Request  Response Body: {"error": "Invalid value for query parameter 'limit'. Cannot be greater than 20."} /  {"error": "Invalid value for query parameter 'type'. Must be one of: fiction, non-fiction."}  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_004 |
| Test case title | Verify a user can view details of a specific book. |
| Test description | Ensure a user can retrieve detailed information about a book using /books/:bookId |
| Pre-conditions | A list of books is available, and An existed bookId is selected. |
| Test steps | HTTP Method: GET  Endpoint: /books/:bookId  Headers: None  Request Body: None  Parameters:  Path Variable: bookId 4 |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: Contains properties such as bookId, name, type, and availability.  Response Headers: Standard HTTP headers |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_005 |
| Test case title | Verify a new order can be placed successfully. |
| Test description | Ensure a user can submit an order for a selected book using /orders. |
| Pre-conditions | User is authenticated, and a valid bookId is selected. |
| Test steps | HTTP Method: POST  Endpoint: /orders  Headers:  Content-Type: application/json  Authorization: Bearer <YOUR TOKEN>  Request Body:  {  "bookId": 1,  "customerName": "John Doe"  }  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 201 Created  Response Body: Contains an orderId.  Response Headers: Standard HTTP headers |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_005.1 |
| Test case title | Verify error when submitting an order without authentication. |
| Test description | Ensure the API returns an error if a user tries to place an order without a valid access token. |
| Pre-conditions | None |
| Test steps | HTTP Method: POST  Endpoint: /orders  Headers:  Content-Type: application/json (No Authorization header).  Request Body:  {  "bookId": 1,  "customerName": "Ha"  }  Parameters: None |
| Input data | N/A |
| Expected results |  |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_005.2 |
| Test case title | Verify error when submitting an order with multiple books at once. |
| Test description | Ensure the API returns an error if a user tries to place an order with an invalid request body (nested book object). |
| Pre-conditions | User is authenticated. |
| Test steps | HTTP Method: POST  Endpoint: /orders  Request Body:  {  "books": [  {"bookId": 1},  {"bookId": 2},  {"bookId": 3},  {"bookId": 4}  ],  "customerName": "Lmao"  }  Parameters: None |
| Input data |  |
| Expected results | Status Code: 401 Unauthorized  Response Body: {"error": "Authentication required"}  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_005.3 |
| Test case title | Verify error when submitting an order with an out of range bookId. |
| Test description | Ensure the API returns an error if a user tries to place an order with an invalid bookId. |
| Pre-conditions | User is authenticated. |
| Test steps | HTTP Method: POST  Endpoint: /orders  Headers:  Content-Type: application/json  Authorization: Bearer <YOUR TOKEN>  Request Body:  {  "bookId": 0/7,  "customerName": "John Doe"  }  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 400 Bad Request  Response Body: {"error": " Invalid or missing bookId."}  Response Headers: Standard HTTP headers. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_006 |
| Test case title | Verify a user can retrieve all their orders. |
| Test description | Ensure a user can view a list of their orders using /orders |
| Pre-conditions | User is authenticated, and at least one order exists. |
| Test steps | HTTP Method: GET  Endpoint: /orders  Headers:  Authorization: Bearer <YOUR TOKEN>  Request Body: None  Parameters: None |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: A list of orders, including the recently placed order.  Response Headers: Standard HTTP headers (e.g., Content-Type: application/json). |
| Post-conditions |  |

**Scenario 2: Modify and Delete an Existing Order**

|  |  |
| --- | --- |
| Test case ID | TC\_API\_007 |
| Test case title | Verify a user can modify an existing order. |
| Test description | Ensure a user can update the customerName of an order. |
| Pre-conditions | User is authenticated with a valid token, and the order exists. |
| Test steps | HTTP Method: PATCH  Endpoint: /orders/:orderId  Headers: Authorization: Bearer <TOKEN>  Request Body:  {  "customerName": "Lmao"  }  Parameteres: orderId: {existingOrderId} |
| Input data | N/A |
| Expected results | Status Code: 200 OK  Response Body: Updated order details. |
| Post-conditions |  |

|  |  |
| --- | --- |
| Test case ID | TC\_API\_008 |
| Test case title | Verify a user can delete an existing order. |
| Test description | Ensure a user can successfully delete an order. |
| Pre-conditions | User is authenticated with a valid token, and the order exists. |
| Test steps | HTTP Method: DELETE  Endpoint: /orders/:orderId  Headers: Authorization: Bearer <TOKEN>  Request Body: None  Parameteres: orderId: {nonexistingOrderId} |
| Input data | N/A |
| Expected results | Status Code: 204 No Content |
| Post-conditions |  |