**API & Database Test Cases for /articles?include=author**

**Positive API Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TC # | Test Scenario | Test Data | Steps | Expected Result | Actual Result |
| 1 | Verify response contains only allowed fields for articles | { "title": "JSON:API paints my bikeshed!", "body": "The shortest article. Ever." } | Send a GET /articles?include=author request and check attributes | title, body, and author should be present, no extra fields | TBD |
| 2 | Verify response contains only name field for people | { "name": "John" } | Send a GET /articles?include=author request and check included objects | Only name should be present, no extra fields like age or gender | TBD |
| 3 | Verify response when author relationship is present | { "title": "Sample Title", "body": "Sample Body", "author": { "id": "42", "type": "people" } } | Send a GET /articles?include=author request with author relationship present | Response should include author details under relationships | TBD |
| 4 | Verify response when author relationship is missing | { "title": "JSON:API paints my bikeshed!", "body": "The shortest article. Ever." } | Send a GET /articles request without include=author | Articles should be returned without author field, no errors | TBD |
| 5 | Verify API supports multiple authors if allowed | { "title": "Multi-author Article", "body": "Content", "authors": [{"id": "42", "name": "John"}, {"id": "43", "name": "Jane"}] } | Send a GET /articles?include=author request with multiple authors per article | API should return multiple authors or 400 Bad Request | TBD |

**Negative API Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TC # | Test Scenario | Test Data | Steps | Expected Result | Actual Result |
| 6 | Verify response fails when extra fields are present in attributes | { "title": "Sample", "body": "Text", "created": "2025-02-12T14:56:29.000Z" } | Send a GET /articles?include=author request and check attributes | API should return 400 Bad Request or sanitize extra fields | TBD |
| 7 | Verify response fails when extra fields are present in included | { "name": "John", "age": 40, "gender": "male" } | Send a GET /articles?include=author request and check included section | API should return 400 Bad Request or remove extra fields | TBD |
| 8 | Verify response format follows JSON:API specification | { "data": [{"type": "articles", "id": "1"}] } | Send a GET /articles?include=author request and validate format | API should return 400 Bad Request if format is incorrect | TBD |
| 9 | Verify response when ?include= is given an invalid value (e.g., number) | ?include=123 | Send a GET /articles?include=123 request | API should return 400 Bad Request | TBD |

**Database Test Cases**

**Positive Database Test Cases**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| TC # | Test Scenario | Test Data | Steps | Expected Result | Actual Result |
| 10 | Verify API returns empty array when no articles exist | Database has no articles | Send a GET /articles?include=author request when DB is empty | API should return 200 OK with "data": [] | TBD |

**Negative Database Test Cases**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| TC # | Test Scenario | Test Data | Steps | Expected Result | Actual Result | |
| 11 | Verify response when relationships.author.data is missing | { "title": "test", "body": "test" } | Remove relationships.author.data field and check response | API should return 400 Bad Request | | TBD |
| 12 | Verify response when included has incorrect type | { "type": "users", "id": "42", "attributes": { "name": "John" } } | Change included type to users instead of people | API should return 400 Bad Request | | TBD |
| 13 | Verify response when id mismatch occurs | { "relationships": { "author": { "data": { "id": "99", "type": "people" } } }, "included": [{ "id": "42", "type": "people" }] } | Change id in included to not match relationships.author.data | API should return 400 Bad Request | | TBD |
| 14 | Verify response when multiple articles have the same author | { "articles": [{ "title": "A1", "author": "42" }, { "title": "A2", "author": "42" }] } | Send a GET /articles?include=author request with duplicate authors | included should contain unique author entries | | TBD |