| | | |
|---|---|---|
| 1 | CARDS | |
| 1.001 | Get a specify property of a card | |
| | API | @GET /cards/{id}/{field} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br><br>TEST:<br>4. Search for specify property of previously created card using @GET method with parameters:<br><br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>o field = “name”.<br><br>EXPECTED RESULT:<br>5. After executing the method, the application will return JSON object with the value of requested field. In this TCS we asked for name of the card.<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.002 | Get the list a card is in | |
| | API | @GET /cards/{id}/list |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>TEST:<br>4. Search for previously created list where requested card is located, using @GET method with parameters:<br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>o fields = “all”, API will return all fields of requested object.<br>EXPECTED RESULT:<br>5. After executing the method, the application will return the list JSON object where requested card is located.<br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.003 | Get the board a card is on | |
| | API | @GET / cards/{id}/board |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br><br>TEST:<br>4. Search for previously created board where requested card is located, using @GET method with parameters:<br><br>• PATH PARAMS<br>○ id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>○ fields = “all” API will return all fields of requested object.<br><br>EXPECTED RESULT:<br>5. After executing the method, the application will return the board JSON object with where requested card is located.<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.004 | Update a card | |
| | API | @PUT / /cards/{id} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>TEST:<br>4. Update previously created card using @PUT method with parameters:<br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>o name = “XYZ card” (any value, different from current card name).<br>EXPECTED RESULT:<br>5. After executing the method, the application will update the card and return the card JSON object, the card name will be changed to given one.<br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.005 | Add a new comment to a card | |
| | API | @POST / cards/{id}/actions/comments |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br><br>TEST:<br>4. Add a new comment to existing card using @POST method with parameters:<br><br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>o text = “lorem ipsum” (or any other not empty value).<br><br>EXPECTED RESULT:<br>5. After executing the method, the application will return the comment JSON object with given text.<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.006 | Update an existing comment | |
| | API | @PUT / /cards/{id}/actions/{idAction}/comments |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>4. Creating a card comment using @POST method for previously created idCard.<br><br>TEST:<br>5. Update an existing card comment using @PUT method with parameters:<br><br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br>o idAction = idComment, provided from the previous step.<br>• QUERY PARAMS<br>o text = “XYZ”, new card comment (any value different from existing one).<br><br>EXPECTED RESULT:<br>6. After executing the method, the application will return the comment JSON object with updated text.<br><br>TEARDOWN:<br>7. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.007 | Add a new label to a card | |
| | API | @POST /cards/{id}/labels |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br><br>TEST:<br>4. Create a new label and add it to a card using @POST method with parameters:<br><br>• PATH PARAMS<br>○ id = idCard, provided from the previous step.<br>• QUERY PARAMS<br>○ color = “blue”, label color.<br>○ name = “test label”, label name.<br><br>EXPECTED RESULT:<br>5. After executing the method, the application will return the label JSON object with the given color and text.<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. |

|  |  |  |
|---|---|---|
| 1 | CARDS |  |
| 1.008 | Remove a label from a card |  |
|  | API | @DELETE / cards/{id}/idLabels/{idLabel} |
|  | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>4. Create and add the label using @POST method for previously created idCard.<br><br>TEST:<br>5. Search for previously remove a label from a card using @DELETE method with parameters:<br><br>• PATH PARAMS<br>o id = idCard provided from the previous step.<br>o idLabel = idLabel, the ID of the label to remove.<br><br>EXPECTED RESULT:<br>6. After executing the method, the application will return HTTP code 200.<br><br>TEARDOWN:<br>Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.009 | List the actions on a card | |
| | API | @GET/ cards/{id}/actions |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>4. Add a new comment to existing card using @POST method.<br><br>TEST:<br>5. Get the actions made on a card using @GET method with parameters:<br><br>• PATH PARAMS<br>o id = idCard, provided from the previous step.<br><br>EXPECTED RESULT:<br>6. After executing the method, the application will return the JSON Array (list) of actions made on card. The first object in mentioned list should contain property “type” with value “commentCard”.<br><br>TEARDOWN:<br>Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 1 | CARDS | |
| 1.010 | Find the card with desire value | |
| | API | @GET /search |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>2. Create list using lists @POST method for previously created idBoard.<br>3. Create card using cards @POST method for previously created idList.<br>4. Create another card on the same list with different name using cards @POST method.<br><br>TEST:<br>5. Search for previously created card using different value than idCard. Find the card with desire value using @GET method with parameters:<br><br>• QUERY PARAMS<br>o query = The search query with a length of 1 to 16384 characters, the given value (for example card name).<br>o idBoards = mine, mine or a comma-separated list of board ids<br>o modelTypes = cards, what type or types of Trello objects you want to search. all or a comma-separated list of: actions, boards, cards, members, organizations.<br><br>EXPECTED RESULT:<br>6. After executing the method, the application will return the search JSON object with cards array containing desired card.<br><br>TEARDOWN:<br>7. Delete board created in first step using @DELETE board method. |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.001 | Request a single board. | |
| | API | @GET /boards/{id} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br>TEST:<br>2. Request using @GET method for a single board by its unique ID.<br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>EXPECTED RESULT:<br>3. Request response OK HTTP200<br>4. After executing the method, the application will return JSON object containing BOARD content.<br>5. Returned object has field “name” with value used during board creation<br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.002 | Get a specific property of a board | |
| | API | @GET /boards/{id}/{field} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method with some object fields like name and desc.<br>TEST:<br>2. Request using @GET method for a single board by it’s unique ID.<br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>○ field = name of the desired field<br>EXPECTED RESULT:<br>3. Request response OK HTTP200<br>4. After executing the method, the application will return JSON object containing requested field content<br>TEARDOWN:<br>5. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.003 | Update an existing board by id | |
| | API | @PUT /boards/{id} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method with some object fields like name and desc.<br><br>TEST:<br>2. Request using @PUT method for an update of a board<br><br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>○ name = new name field value<br>○ desc = new desc field value<br>3. Request using @GET method for a single board by its unique ID.<br><br>EXPECTED RESULT:<br>4. Request response OK HTTP200<br>5. After executing the method, the application will return JSON object containing changed field<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.004 | Create new Board | |
| | API | @POST /boards |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method with some object fields like name and desc.<br><br>TEST:<br>2. Request using @GET method for a single board by it’s unique ID provided by previous step<br><br>• QUERY PARAMS<br>○ name = new name field value<br>○ desc = new desc field value<br><br>EXPECTED RESULT:<br>3. Request response OK HTTP200<br>4. After executing the method, the application will return JSON object containing field with correct values used during board creation<br><br>TEARDOWN:<br>5. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.005 | Delete board | |
| | API | @DELETE /boards/{id} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method<br>2. Delete board created in first step using @DELETE board method. ( Response HTTP200)<br><br>TEST:<br>3. Request using @GET method for a single board by its unique ID provided by previous step<br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br><br>EXPECTED RESULT:<br>4. Request response NOT FOUND HTTP404<br>5. After executing the method, the application will return error response with message “The requested resource was not found.”<br><br>TEARDOWN:<br>6. None (object deleted) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.006 | Set emailPosition on Board | |
| | API | @PUT /boards/{id}/myPrefs/emailPosition |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method<br><br>TEST:<br>2. Set email position using PUT method<br>3. Get Board MyPrefs<br>4. Assert email position bottom<br>5. Set email position using PUT method<br>6. Get Board MyPrefs<br>7. Assert email position top<br><br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>• QUERY PARAMS<br>○ value = Valid values: bottom, top. Determines the position of the email address.<br><br>EXPECTED RESULT:<br>8. Email position changed after each PUT command<br><br>TEARDOWN:<br>9. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.007 | Set showSidebar on Board | |
| | API | @PUT /boards/{id}/myPrefs/showSidebar |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method<br><br>TEST:<br>2. Get Board MyPrefs<br>3. Assert showSidebar is true as default value<br>4. Set showSidebar item using PUT command to false<br>5. Get Board MyPrefs<br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>• QUERY PARAMS<br>○ value = Determines whether to show the side bar<br><br>EXPECTED RESULT:<br>6. showSidebar value is equal false<br><br>TEARDOWN:<br>7. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.008 | Update an existing board | |
| | API | @PUT /boards/{id} |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method<br><br>TEST:<br>1. Get Board MyPrefs<br>2. Assert showSidebar is true as default value<br>3. Set showSidebar item using PUT command to true<br>4. Get Board MyPrefs<br><br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>• QUERY PARAMS<br>○ Desc = new description for the board<br>○ labelNames/blue= name of blue label<br><br>EXPECTED RESULT:<br>5. New description for the board and Name of blue label<br><br>TEARDOWN:<br>6. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.009 | Set showListGuide on Board | |
| | API | @PUT/boards/{id}/myPrefs/showListGuide |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method<br><br>TEST:<br>2. Request using @POST method for a single board by it’s unique ID provided by previous step<br>3. Assert  showListGuide is true as default value<br>4. Set showListGuide item using PUT command<br>5. Get MyPrefs<br>• PATH PARAMS<br>○ id = id of the board, provided from the previous step<br>• QUERY  PARAMS<br>○ value= determines whether to show the list guide<br><br>EXPECTED RESULT:<br>6. showListGuide value is equal true<br><br>TEARDOWN:<br>7. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 2 | BOARDS | |
| 2.010 | Search Board | |
| | API | @GET/search |
| | Test steps: | SETUP:<br>1. Create board using boards @POST method.<br><br>TEST:<br>2. Request using @GET method for a single board by its unique ID.<br>3.  Assert search is true as default value using GET command<br>• QUERY  PARAMS<br>○ query = the search query<br><br>EXPECTED RESULT:<br>4. Search boards value is equal true<br><br>TEARDOWN:<br>5. Delete board created in first step using @DELETE board method. ( Response HTTP200) |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.001 | Get list by id – positive TC | |
| | API | @GET /lists/id |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Get previously created list using @GET method with parameters:<br>- id:<<id from second step>><br>-fields: <<all>><br><br>EXPECTED RESULT<br>4. List info returned as a JSON response<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.002 | Get list by invalid value – negative TC | |
| | API | @GET /lists/id |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Get previously created list using @GET method with parameters:<br>- id:<<id from second step>><br><br>EXPECTED RESULT<br>4. Message about invalid value in id<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.003 | Get list with no id – negative TC | |
| | API | @GET /lists/id |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Get previously created list using @GET method with parameters:<br>id:<<id from second step>><br><br>EXPECTED RESULT<br>4. Message about id value = NULL.<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.004 | List the cards in a list with no ID of the list –negative TC | |
| | API | @GET /lists/{id}/cards |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using @POST method for previously created idList.<br><br>TEST<br>4. Search for list of cards which is connected with list from third Setup step<br><br>EXPECTED RESULT<br>5. Message about value of ID of the list =NULL<br><br>TEARDOWN<br>6. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.005 | List the cards in a list with invalid value of ID of the list –negative TC | |
| | API | @GET /lists/{id}/cards |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using @POST method for previously created idList.<br><br>TEST<br>4. Search for list of cards which is connected with list from third Setup step<br><br>EXPECTED RESULT<br>5. Message about invalid value of ID of the list<br><br>TEARDOWN<br>6. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.006 | List the cards in a list – positive TC | |
| | API | @GET /lists/{id}/cards |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using @POST method for previously created idList.<br><br>TEST<br>4. Search for list of cards which is connected with list from third Setup step<br><br>EXPECTED RESULT<br>5. List info returned as a JSON response<br><br>TEARDOWN<br>6. Delete board created in first step using @DELETE method |

| 3 | LISTS | |
|---|---|---|
| 3.007 | Archive a list – positive TC | |
| | API | @PUT/lists/{id}/closed |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Use ID of the list of the previously created list in SETUP step 2 to archive a list using @PUT method.<br>4. Check result using @GET method.<br><br>EXPECTED RESULT<br>5. The attribute of the list is <<true>> (shown up in the archive).<br><br>TEARDOWN<br>6. Delete board created in first step using @DELETE method |

| 3 | LISTS | |
|---|---|---|
| 3.008 | Archive a list – negative case | |
| | API | @PUT/lists/{id}/closed |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Use ID of the list of the previously created list in SETUP step 2 to archive a list using @PUT method.<br>4. Check result using @GET method.<br><br>EXPECTED RESULT<br>5. Message about the attribute of the list <<false>> ( shown up on the board ).<br><br>TEARDOWN<br>6. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.009 | Create a new list on a board – positive TC | |
| | API | @POST /lists/ |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br><br>TEST<br>2. Create list using list @POST method for previously created idBoard with parameters:<br>- idBoard:<<id from first step>><br>- name: <<wish list>><br><br>EXPECTED RESULT<br>3. List info returned as a JSON response with specific idBoard from step 2 and specific name ‘wish list’<br><br>TEARDOWN<br>4. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.010 | Create a new list on a board with no name – negative TC | |
| | API | @POST /lists/ |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Create list using list @POST method for previously created idBoard with parameters:<br>- idBoard:<<id from first step>><br>- name: <<wish list>><br><br>EXPECTED RESULT<br>4. Message about name value = NULL<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.011 | Create a new list on a board with invalid idBoard – negative TC | |
| | API | @POST /lists/ |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Create list using list @POST method for previously created idBoard with parameters:<br>- idBoard:<<id from first step>><br>- name: <<wish list>><br><br>EXPECTED RESULT<br>4. Message about invalid idBoard<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 3 | LISTS | |
| 3.012 | Update the properties of a list– positive TC | |
| | API | @PUT//lists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Change name of a list using @PUT method with parameters:<br>-name <<must have list>><br>-id of the list <<id from second step>><br><br>EXPECTED RESULT<br>4. List info returned as a JSON response with specific name ‘must have list from step 2 and specific id of the list<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| 3 | LISTS | |
|---|---|---|
| 3.013 | Update the properties of a list with no id of the list– negative TC | |
| | API | @PUT//lists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br><br>TEST<br>3. Change name of a list using @PUT method with parameters:<br>-name <<must have list>><br>-id of the list <<id from second step>><br><br>EXPECTED RESULT<br>4. Message about value of id of the list = NULL.<br><br>TEARDOWN<br>5. Delete board created in first step using @DELETE method |

| 4 | CHECKLISTS | |
|---|---|---|
| 4.001 | Get checklist by id of a checklist – positive TC | |
| | API | @GET /checklists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Get previously created checklist from step 4.<br><br>EXPECTED RESULT<br>6. List info returned as a JSON response<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

|  |  |  |
|---|---|---|
| 4 | CHECKLISTS |  |
| 4.002 | Get checklist by invalid ID of a checklist – negative TC |  |
|  | API | @GET /checklists/{id} |
|  | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Get previously created checklist from step 4.<br><br>EXPECTED RESULT<br>6. Message about invalid ID<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

|  |  |  |
|---|---|---|
| 4 | CHECKLISTS |  |
| 4.003 | Get checklist by no ID of a checklist – negative TC |  |
|  | API | @GET /checklists/{id} |
|  | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Get previously created checklist from step 4.<br><br>EXPECTED RESULT<br>6. Message about value ID = NULL<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

|  |  |  |
|---|---|---|
| 4 | CHECKLISTS |  |
| 4.004 | Delete a checklist by id – positive TC |  |
|  | API | @DELETE /checklists/{id} |
|  | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Delete a checklist from step 4.<br><br>EXPECTED RESULT<br>6. List info returned as a JSON response<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

|  |  |  |
|---|---|---|
| 4 | CHECKLISTS |  |
| 4.005 | Delete a checklist with no ID of a checklist – negative TC |  |
|  | API | @DELETE /checklists/{id |
|  | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Delete a checklist from step 4.<br><br>EXPECTED RESULT<br>6. Message about value of ID of a checklist = NULL<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| 4 | CHECKLISTS | |
|---|---|---|
| 4.006 | Delete a checklist with invalid ID of a checklist – negative TC | |
| | API | @DELETE /checklists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Delete a checklist from step 4.<br><br>EXPECTED RESULT<br>6. Message about invalid value of ID of a checklist<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| 4 | CHECKLISTS | |
|---|---|---|
| 4.007 | Update an existing checklist – positive TC | |
| | API | @PUT /checklists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Change name of a checklist using @PUT method with parameters:<br>-name <<a laptop>><br>-id of the checklist <<id from fourth step>><br><br>EXPECTED RESULT<br>6. List info returned as a JSON response with specific name ‘a laptop’ from step 5 and specific id of the list from step 5<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.008 | Update an existing checklist with no ID of a checklist – negative TC | |
| | API | @PUT /checklists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Change name of a checklist using @PUT method with parameters:<br>-name <<a laptop>><br>-id of the checklist <<id from fourth step>><br><br>EXPECTED RESULT<br>6. Message about value of a checklist = NULL<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.009 | Update an existing checklist with invalid ID of a checklist – negative TC | |
| | API | @PUT /checklists/{id} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Change name of a checklist using @PUT method with parameters:<br>-name <<a laptop>><br>-id of the checklist <<id from fourth step>><br><br>EXPECTED RESULT<br>6. Message about invalid value of ID of a checklist<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.010 | Create a new checkItem in a checklist – positive TC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>- id:<<id from fourth step>><br>- name: << malpa>><br><br>EXPECTED RESULT<br>6. List info returned as a JSON response<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.011 | Create a new checkItem in a checklist with a wrong name–negativeTC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: <<string's length is 16385>><br><br>EXPECTED RESULT<br>6. Message: string's length of the name is too long<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| 4 | CHECKLISTS | |
|---|---|---|
| 4.012 | Create a new checkItem in a checklist with no name–negativeTC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br><br>EXPECTED RESULT<br>6. Message about value of the name = NULL<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| 4 | CHECKLISTS | |
|---|---|---|
| 4.013 | Create a new checkItem in a checklist with no ID of a checklist–negativeTC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br><br>EXPECTED RESULT<br>6. Message about value of ID of a checklist = NULL<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.014 | Create a new checkItem in a checklist with invalid ID of a checklist–negativeTC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br><br>EXPECTED RESULT<br>6. Message about invalid value of ID of a checklist<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.015 | Create a new checkItem in a checklist with invalid name–negativeTC | |
| | API | @POST /checklists/{id}/checkItems |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br><br>TEST<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br><br>EXPECTED RESULT<br>6. Message about invalid name<br><br>TEARDOWN<br>7. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.016 | Get an item from a checklist – positive TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checkItem using checkItem @POST method for previously created id with parameters:<br>d:<<id from fourth step>><br>name: << malpa>><br>TEST<br>6. Get an item from a checklist from step 5.<br>EXPECTED RESULT<br>7. List info returned as a JSON response<br>TEARDOWN<br>8. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.017 | Get an item from a checklist – positive TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checklist using checklist @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br>fields: <<all>><br>6. Create checkItem using @POST method for previously created checkList.<br>TEST<br>7. Get an item from a checklist from step 5.<br>EXPECTED RESULT<br>8. List info returned as a JSON response<br>TEARDOWN<br>9. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.018 | Get an item from a checklist with no ID of a checklist – negative TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checklist using checklist @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br>6. Create checkItem using @POST method for previously created checkList.<br><br>TEST<br>7. Get an item from a checklist from step 5.<br><br>EXPECTED RESULT<br>8. Message about value of ID of a checklist = NULL<br><br>TEARDOWN<br>9. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.019 | Get an item from a checklist with invalid ID of a checklist – negative TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checklist using checklist @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br>6. Create checkItem using @POST method for previously created checkList.<br>TEST<br>7. Get an item from a checklist from step 5.<br><br>EXPECTED RESULT<br>8. Message about invalid value of ID of a checklist<br><br>TEARDOWN<br>9. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.020 | Get an item from a checklist with invalid idCheckItem of a checklist – negative TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checklist using checklist @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br>6. Create checkItem using @POST method for previously created checkList.<br>TEST<br>7. Get an item from a checklist from step 5.<br>EXPECTED RESULT<br>8. Message about invalid value of ID of a checkItem<br>TEARDOWN<br>9. Delete board created in first step using @DELETE method |

| | | |
|---|---|---|
| 4 | CHECKLISTS | |
| 4.021 | Get an item from a checklist with no idCheckItem of a checklist – negative TC | |
| | API | @GET /checklists/{id}/checkItems/{idCheckItem} |
| | Test steps: | SETUP<br>1. Create board using board @POST method.<br>2. Create list using list @POST method for previously created idBoard.<br>3. Create a card using card @POST method for previously created idList.<br>4. Create checklist using checklist @POST method for previously created idCard.<br>5. Create checklist using checklist @POST method for previously created id with parameters:<br>id:<<id from fourth step>><br>name: << HP Pavilion 15-CW0008NW>><br>6. Create checkItem using @POST method for previously created checkList.<br>TEST<br>7. Get an item from a checklist from step 5.<br>EXPECTED RESULT<br>8. Message about value of ID of a checkItem = NULL<br>TEARDOWN<br>9. Delete board created in first step using @DELETE method |

|  |  |  |
|---|---|---|
| 5 | LABELS |  |
| 5.001 | Create a new label on a board |  |
|  | API | @POST /labels/{id}/name |
|  | Test steps: | SETUP:<br>1. Create a board using boards @POST method with some object fields like name and description<br><br>TEST:<br>2. Create a label on the board from STEP 1 using labels @POST method with some object fields like name, color and idBoard<br><br>QUERY PARAMS<br>name - Name for the label<br>color - The color for the label. See fields for color options<br>idBoard - The ID of the board to create the label on<br><br>EXPECTED RESULT:<br>3. Request response OK HTTP 200.<br>4. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>5. Delete the board created in the first step using @DELETE board method (Response HTTP 200) |

|  |  |  |
|---|---|---|
| 5 | LABELS |  |
| 5.002 | Delete a label on a board |  |
|  | API | @DELETE /labels/{id} |
|  | Test steps: | SETUP:<br>1. Create a board using boards @POST method with some object fields like name and description<br>2. Create a label on the board from STEP 1 using labels @POST method with some object fields like name, color and idBoard<br><br>TEST:<br>3. Delete the label on the board using labels @DELETE method by its ID<br><br>PATH PARAMS<br>Id - The ID of the label to delete<br><br>EXPECTED RESULT:<br>4. Request response OK HTTP 200.<br>5. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>6. Delete the board created in the first step using @DELETE board method (Response HTTP 200) |

|  |  |  |
|---|---|---|
| 5 | LABELS |  |
| 5.003 | Get an information about a label by ID |  |
|  | API | @GET /labels/{id} |
|  | Test steps: | SETUP:<br>1. Create a board using boards @POST method with some object fields like name and description<br>2. Create a label on the board from STEP 1 using labels @POST method with some object fields like name,  color and idBoard<br><br>TEST:<br>3. Get an information about the label using labels @GET method by its ID<br><br>PATH PARAMS<br>Id – id of the label<br><br>QUERY PARAMS<br>fields - all or a comma-separated list of fields<br><br>EXPECTED RESULT:<br>4. Request response OK HTTP 200.<br>5. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>6. Delete board created in the first step using @DELETE board method (Response HTTP 200) |

|  |  |  |
|---|---|---|
| 5 | LABELS |  |
| 5.004 | Update the color of a label by ID |  |
|  | API | @PUT /labels/{id}/color |
|  | Test steps: | SETUP:<br>1. Create a board using boards @POST method with some object fields like name and description<br>2. Create a label on the board from STEP 1 using labels @POST method with some object fields like name, color and idBoard<br><br>TEST:<br>3. Update the color of the the label using labels @PUT method by its ID<br><br>PATH PARAMS<br>id - The id of the label<br><br>QUERY PARAMS<br>value - The new color for the label<br><br>EXPECTED RESULT:<br>4. Request response OK HTTP 200.<br>5. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>6. Delete an board created in the first step using @DELETE board method (Response HTTP 200) |

| | | |
|---|---|---|
| 5 | LABELS | |
| 5.005 | Update the name of a label by ID | |
| | API | @PUT /labels/{id}/name |
| | Test steps: | SETUP:<br>1. Create a board using boards @POST method with some object fields like name and description<br>2. Create a label on the board from STEP 1 using labels @POST method with some object fields like name, color and idBoard<br><br>TEST:<br>3. Update the name of the label using labels @PUT method by its ID<br><br>PATH PARAMS<br>id -The id of the label to update<br><br>QUERY PARAMS<br>value - The new name for the label<br><br>EXPECTED RESULT:<br>4. Request response OK HTTP 200.<br>5. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>6. Delete the board created in the first step using @DELETE board method (Response HTTP 200) |

| | | |
|---|---|---|
| 6 | ORGANIZATIONS | |
| 6.001 | Create a new organization | |
| | API | @POST /organization |
| | Test steps: | SETUP:<br><br>TEST:<br>1. Create an organization using organizations @POST method.<br><br>QUERY PARAMS<br>displayName<br>desc - The description for the team<br>name - A string with a length of at least 3. Only lowercase letters, underscores, and numbers are allowed. Must be unique.<br>website - A URL starting with http:// or https://<br><br>EXPECTED RESULT:<br>2. Request response OK HTTP 200.<br>3. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>4. Delete the organization created in the first step using @DELETE organizations method (Response HTTP 200) |

| | | |
|---|---|---|
| 6 | ORGANIZATIONS | |
| 6.002 | Get an information about the organization by ID | |
| | API | @GET /organizations/{id} |
| | Test steps: | SETUP:<br>1. Create an organization using organizations @POST method with some object fields like name, desc, displayName and website.<br>TEST:<br>2. Get an information about the organization using organizations @GET method by its ID<br>PATH PARAMS<br>id - The ID or name of the organization<br>EXPECTED RESULT:<br>3. Request response OK HTTP 200.<br>4. After executing the method, the application will return JSON object containing requested field content<br>TEARDOWN:<br>5. Delete the organization created in the first step using @DELETE organizations method (Response HTTP 200) |

| | | |
|---|---|---|
| 6 | ORGANIZATIONS | |
| 6.003 | Get an information about the boards in an organization | |
| | API | @GET /organizations/{id}/boards |
| | Test steps: | SETUP:<br>1. Create an organization using organizations @POST method with some object fields like name, desc, displayName and website.<br>TEST:<br>2. Get an information about the boards in an organization using organizations @GET method by its ID<br>PATH PARAMS<br>id - The ID or name of the organization<br>QUERY PARAMS<br>filter - all or a comma-separated list of: open, closed, members, organization, public<br>fields - all or a comma-separated list of board fields<br>EXPECTED RESULT:<br>3. Request response OK HTTP 200.<br>4. After executing the method, the application will return JSON object containing requested field content<br>TEARDOWN:<br>5. Delete the organization created in the first step using @DELETE organizations method (Response HTTP 200) |

|  |  |  |
|---|---|---|
| 6 | ORGANIZATIONS |  |
| 6.004 | Update an organizations description |  |
|  | API | @GET /organizations/{id}/desc |
|  | Test steps: | SETUP:<br>1. Create an organization using organizations @POST method with some object fields like name, desc, displayName and website.<br><br>TEST:<br>2. Update the description of the organization using organizations @PUT method by its ID<br><br>PATH PARAMS<br>id - The id of the organization<br><br>QUERY PARAMS<br>value - The new description for the organization<br><br>EXPECTED RESULT:<br>3. Request response OK HTTP 200.<br>4. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>5. Delete the organization created in the first step using @DELETE organizations method (Response HTTP 200) |

|  |  |  |
|---|---|---|
| 6 | ORGANIZATIONS |  |
| 6.005 | Update an organizations name |  |
|  | API | @GET /organizations/{id}/name |
|  | Test steps: | SETUP:<br>1. Create an organization using organizations @POST method with some object fields like name, desc, displayName and website.<br><br>TEST:<br>2. Update the name of the organization using organizations @PUT method by its ID<br><br>PATH PARAMS<br>id - The id of the organization<br><br>QUERY PARAMS<br>value - The new name for the organization<br><br>EXPECTED RESULT:<br>3. Request response OK HTTP 200.<br>4. After executing the method, the application will return JSON object containing requested field content<br><br>TEARDOWN:<br>5. Delete the organization created in the first step using @DELETE organizations method (Response HTTP 200) |