Test cases (Deals API)

CRM API baseUrl: `[HIDDEN]`

Authorisation token: `[HIDDEN]`

Deals API

GET

Test case ID	**DG001: Get the list of all deals**
Description	It is possible to retrieve the full list of all existing deals
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal` request
Expected result	• Successful code: 200 OK • List of all deals is returned in response

Test case ID	**DG002: Get the list of all deals - no authorization**
Description	It is not possible to retrieve the list of deals without being authorized
Precondition	Some deals exist in the application, user is NOT authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal` request
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"You do not have permission to get this object"`

Test case ID	**DG003: Get the list of all deals - limit**
Description	It is possible to retrieve the list of deals and set a valid limit value
Precondition	Some deals exist in the application, user is authorized

Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?limit=10` request
Expected result	• Successful code: 200 OK • List of the first 10 deals is returned in response • `"count": 10`
Test case ID	**DG004: Get the list of all deals - negative limit**
Description	Using an invalid limit value returns an empty results set
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?limit=-10` request
Expected result	• Successful code: 200 OK • The system returns an empty result set along with metadata reflecting the invalid request, but without triggering any internal server errors • Example of the expected result: • `{` • `"response": {` • `"cursor": 0,` • `"results": [],` • `"count": -10,` • `"remaining": 64` • `}` `}`
Test case ID	**DG005: Get the list of all deals - cursor**
Description	It is possible to retrieve the list of deals and set a valid cursor value
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?limit=10&cursor=10` request
Expected result	• Successful code: 200 OK • The array containing deals 11 to 20 is returned in response • `"cursor": 10, "count": 10`

Test case ID	**DG006: Get the list of all deals - negative cursor**
Description	Using a negative cursor value is not allowed
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?limit=10&cursor=-10` request
Expected result	• Error code: 400 Bad Request • Informative error message should be displayed. It should give a hint, that negative values are not allowed

Test case ID	**DG007: Get the list of all deals - sorting by company name**
Description	It is possible to sort all deals by company name
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?sort_field=company_name_text&descending=false` request
Expected result	• Successful code: 200 OK • List of all deals displayed, sorted by company name ASC

Test case ID	**DG008: Get the list of all deals - sorting by name**
Description	It is possible to sort all deals by name (DESC)
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?sort_field=name_text&descending=true` request
Expected result	• Successful code: 200 OK • List of all deals displayed, sorted by name DESC

Test case ID	**DG009: Get the list of all deals - sorting by created date**
Description	It is possible to sort all deals by created date (ASC)
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?sort_field=Created Date&descending=false` request
Expected result	• Successful code: 200 OK • List of all deals displayed, sorted by created date ASC
Test case ID	**DG010: Get the list of all deals - filtering deals by specific status**
Description	It is possible to get the list of deals with a specific status (e.g., “Won”)
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?constraints=[{ "key":"status_option_status", "constraint_type": "equals","value":"Won"}]` request
Expected result	• Successful code: 200 OK • List of all deals with `Won` status is displayed
Test case ID	**DG011: Get the list of all deals - filtering deals with notes**
Description	It is possible to get the list of deals with the notes added only
Precondition	Some deals exist in the application, user is authorized
Test data	-
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal?constraints=[{ "key":"note_list_custom_note", "constraint_type": "not empty"}]` request
Expected result	• Successful code: 200 OK • List of all deals with notes is displayed

GET by Id

Test case ID	**DGI001: Get the deal by Id**
Description	It is possible to retrieve an existing deal by Id
Precondition	Some deals exist in the application, user is authorized
Test data	Unique Id of an existing deal
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal/:UniqueId` request
Expected result	• Successful code: 200 OK • Selected deal is returned in response

Test case ID	**DGI002: Get the deal by Id - no authorization**
Description	It is NOT possible to retrieve any deal by Id without being authorized
Precondition	Some deals exist in the application, user is NOT authorized
Test data	Unique Id of an existing deal
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal/:UniqueId` request
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"You do not have permission to get this object"`

Test case ID	**DGI003: Get the deal by Id - non-existing Id**
Description	It is NOT possible to retrieve a non-existing deal by Id
Precondition	Some deals exist in the application, user is authorized
Test data	Unique Id of a NON-existing deal *Example*: 1625048131815x332253337528303000
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal/:UniqueId` request
Expected result	• Error code: 404 Not Found • Error message displayed: `"Missing object of type deal: object with id 1625048131815x332253337528303000 does not exist"`

Test case ID	**DGI004: Get the deal by Id - wrong Id type**
Description	It is NOT possible to retrieve a deal by wrong type Id
Precondition	Some deals exist in the application, user is authorized
Test data	UniqueId with a wrong data value *Example*: boolean (true or false) instead of string
Test steps	1. Send **GET** `{{baseUrl}}/obj/deal/:UniqueId` request
Expected result	• Error code: 400 Bad Request • Error message displayed: `"Invalid data type for 'UniqueId': expected string, received boolean."`

POST + GET

Test case ID	**DPOG001: Create a new deal with mandatory fields only and check the created object**
Description	It is possible to create a new deal with mandatory fields only and observe if it was created successfully
Precondition	Valid user Ids are present in the system

Test data	<pre>{
	  "assignee__user": "1625047362532x398219546419355650",
	  "company_name_text": "Team 5 mandatory fields",
	  "deal_value_estimation_number": 700000,
	  "name_text": "Haribo Goldbears - Promo campaign",
	  "order_number": 5,
	  "status_option_status": "In progress",
	  "visible_to_list_user": [
	  "1625047362532x398219546419355650"
	  ]
	}</pre>

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request using body from test data, but with mandatory fields only 2. Copy id of the created deal from the response 3. Send **GET** `{{baseUrl}}/obj/deal/:UniqueID`, set correct UniqueID (taken from step 2)

Expected result	Step 1: Successful code: 201 Created, deal is created Step 3: Successful code: 200 OK, correct deal is found, data corresponds to input
Test case ID	**DPOG002: Create a new deal with maximum set of fields and check the created o**
Description	It is possible to create a new deal with maximum set of fields and observe if it is was created succ
Precondition	Valid Ids of users, funnels, leads, notes and stages are present in the system

Test data	`{`
	`  "assignee__user": "1625047373081x778685405315052900",`
	`  "company_name_text": "Team 5 all fields",`
	`  "deal_value_estimation_number": 500,`
	`  "description_text": "This is a test deal",`
	`  "file_list_custom_data_file": [`
	`  "1647262794715x376755327337758700",`
	`  "1647262830545x113983646886789120"`
	`  ],`
	`  "funnel_custom_funnel1": "1724677207945x687761705721445400",`
	`  "lead_custom_lead": "1725012477904x962236867638862600",`
	`  "logo_image":`
	`"https://s3.amazonaws.com/appforest_uf/f1625060020121x284193349144459940/Scree`
	`06-30%20at%2016.33.31.png",`
	`  "name_text": "Haribo Watermelon - Promo campaign",`
	`  "note_list_custom_note": [`
	`  "1678718690483x188510294479732740"`
	`  ],`
	`  "order_number": 33,`
	`  "stage_custom_stage": "1724754232592x426388216980834400",`
	`  "status_option_status": "Won",`
	`  "visible_to_list_user": [`
	`  "1625047373081x778685405315052900"`
	`  ]`
	`}`

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request using body from test data 2. Copy id of the created deal from the response 3. Send **GET** `{{baseUrl}}/obj/deal/:UniqueID`, set correct UniqueID (taken from step 2
Expected result	Step 1: Successful code: 201 Created, deal is created Step 3: Successful code: 200 OK, correct deal is found, data corresponds to input

POST

Test case ID	**DPO001: Create a new deal - no authorization**

Description	It is NOT possible to create a new deal without being authorized. Note: We will only complete the required fields
Precondition	Valid user Ids are present in the system

Test data	<pre>{
	  "assignee__user": "1625047362532x398219546419355650",
	  "company_name_text": "Team 5 no auth",
	  "deal_value_estimation_number": 700001,
	  "name_text": "Haribo Starmix - Promo campaign",
	  "order_number": 55,
	  "status_option_status": "In progress",
	  "visible_to_list_user": [
	  "1625047362532x398219546419355650"
	  ]
	}</pre>

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"Permission denied: cannot create this object"`
Test case ID	**DPO002: Create a new deal - without mandatory fields**
Description	It is NOT possible to create a new deal without mandatory fields
Precondition	Valid Ids of funnels, leads, notes and stages are present in the system

Test data	<pre>{
	  "description_text": "Team 5 Test no mandatory fields",
	  "funnel_custom_funnel1": "1724748016077x923297551481141200",
	  "lead_custom_lead": "1725010563849x249896618869343900",
	  "note_list_custom_note": [
	  "1627393624856x199214596987551740"
	  ],
	  "stage_custom_stage": "1724754232312x8364179569360064 00"
	}</pre>

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request using body from test data, but without mandatory fields
Expected result	• Error code: 400 Bad Request • Deal is not created, error message about missing mandatory fields is displayed
Test case ID	**DPO003: Create a new deal - wrong user value**

Description	It is NOT possible to create a new deal with incorrect user value (e.g., number instead of string)
Precondition	Valid user Ids are present in the system

Test data	<pre>{
	  "assignee__user": 123,
	  "company_name_text": "Team 5 wrong user value",
	  "deal_value_estimation_number": 501,
	  "name_text": "Haribo - Promo campaign",
	  "order_number": 35,
	  "status_option_status": "Lost",
	  "visible_to_list_user": [
	  "1625047373081x778685405315052900"
	  ]
	}</pre>

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request, defining assignee__user as (for example) 123
Expected result	• Error code: 400 Bad Request • Deal is not created, error message `"Invalid data for field assignee_user: expected a string, but got a number"` is displayed
Test case ID	**DPO004: Create a new deal - wrong status value**
Description	It is NOT possible to create a new deal with incorrect status value (NOT: Won, In progress or Lost)
Precondition	Valid Ids of users, funnels, leads, notes and stages are present in the system

Test data	<pre>{
	  "assignee__user": "1625047373081x778685405315052900",
	  "company_name_text": "Team 5 wrong status",
	  "deal_value_estimation_number": 500,
	  "description_text": "This is a test deal 2",
	  "funnel_custom_funnel1": "1724677207945x687761705721445400",
	  "lead_custom_lead": "1725012477904x962236867638862600",
	  "name_text": "Haribo Peaches - Promo campaign",
	  "order_number": 34,
	  "stage_custom_stage": "1724754232592x426388216980834400",
	  "status_option_status": "None",
	  "visible_to_list_user": [
	  "1625047373081x778685405315052900"
	  ]
	}</pre>

Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request using body from test data, defining status_option_status (e.g., None)

Expected result	• Error code: 400 Bad Request • Deal is not created, error message `"Invalid data for field status_option_status: could not parse this as a Status"` is displayed
Test case ID	**DPO005: Create a new deal - wrong deal estimation value**
Description	It is NOT possible to create a new deal with incorrect deal estimation value (e.g., boolean instead of number)
Precondition	Valid user Ids are present in the system
Test data	`{ "assignee__user": "1625047373081x778685405315052900", "company_name_text": "Team 5 wrong estimation value", "deal_value_estimation_number": false, "name_text": "Haribo 2 - Promo campaign", "order_number": 36, "status_option_status": "Lost", "visible_to_list_user": ["1625047373081x778685405315052900"] }`
Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request, defining deal_value_estimation_number as boolean (e.g., false)
Expected result	• Error code: 400 Bad Request • Deal is not created, error message `"Invalid data for field deal_value_estimation_number: Expected a number, but got a boolean (original data: false)"` is displayed
Test case ID	**DPO006: Create a new deal - empty body**
Description	It is NOT possible to create a new deal with an empty body request
Precondition	-
Test data	`{ }`
Test steps	1. Send **POST** `{{baseUrl}}/obj/deal`request with an empty request body
Expected result	• Error code: 400 Bad Request • Deal is not created, informative error message is displayed

PATCH + GET

Test case ID	**DPAG001: Modify a deal and check the data**
Description	It is possible to update an existing deal
Precondition	DPOG001 is successfully completed, user is authorized

Test data	`{`
	`"company_name_text": "Team 5 do not delete - PATCH",`
	`"name_text": "Haribo Watermelon"`
	`}`

Test steps	1. Send **PATCH** `{{baseUrl}}/obj/deal/:UniqueID` , set correct UniqueID (taken from step 2, DPOG001) 2. Send **GET** `{{baseUrl}}/obj/deal/:UniqueID`, set correct UniqueID (taken from previous step)
Expected result	Step 1: Successful code: 204 No Content, update was successful Step 2: Successful code: 200 OK, correct deal is found, company and name values were successfully updated

PATCH

Test case ID	**DPA001: Modify a deal - no authorization**
Description	It is NOT possible to update an existing deal without being authorized
Precondition	Some deals exist in the application, user is NOT authorized

Test data	UniqueID of an existing deal
	
	`{`
	`"company_name_text": "Team 5 do not delete - PATCH_2",`
	`"name_text": "Haribo Watermelon_2"`
	`}`

Test steps	1. Send **PATCH** `{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"You do not have permission to modify this object"`
Test case ID	**DPA002: Modify a deal - wrong data type values**

Description	It is NOT possible to update an existing deal with an incorrect data values (e.g., company name value - boolean instead of string; order value - string instead of number)
Precondition	Some deals exist in the application, user is authorized

Test data	UniqueID of an existing deal
	
	`{`
	`"company_name_text": true,`
	`"order_number": "Team 5"`
	`}`

Test steps	1. Send **PATCH** `{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 400 Bad Request • Error message displayed: `"Invalid data for field company_name_text: Expected a string, but got a boolean (original data: true), invalid data for field order_number: Expected a number, but got a string (original data: "Team 5")"`

PUT + GET

Test case ID	**DPUG001: Replace a deal and check the data**
Description	It is possible to replace an existing deal
Precondition	DPOG002 is successfully completed, user is authorized

Test data	<pre>{
	  "assignee__user": "1625125825767x718816571317085800",
	  "company_name_text": "Team 5 replace",
	  "deal_value_estimation_number": 100,
	  "description_text": "Testing PUT method",
	  "funnel_custom_funnel1": "1724748016077x923297551481141200",
	  "lead_custom_lead": "1725011886770x256041149107739460",
	  "name_text": "Team 5",
	  "note_list_custom_note": [
	  "1678718571924x727404269818871800"
	  ],
	  "order_number": 0,
	  "stage_custom_stage": "1724754233044x396443822862285300",
	  "status_option_status": "Lost",
	  "visible_to_list_user": [
	  "1625125825767x718816571317085800"
	  ]
	}</pre>

Test steps	1. Send **PUT** `{{baseUrl}}/obj/deal/:UniqueID` , set correct UniqueID (taken from step 2, DPOG002) 2. Send **GET** `{{baseUrl}}/obj/deal/:UniqueID`, set correct UniqueID (taken from previous step)
Expected result	Step 1: Successful code: 204 No Content, update was successful Step 2: Successful code: 200 OK, correct deal is found, all fields were successfully replaced with new data

PUT

Test case ID	**DPU001: Replace a deal - no authorization**
Description	It is NOT possible to replace an existing deal without being authorized
Precondition	Some deals exist in the application, user is NOT authorized

Test data	UniqueID of an existing deal
	
	`{`
	`  "assignee__user": "1625125825767x718816571317085800",`
	`  "company_name_text": "Team 5 replace no auth",`
	`  "deal_value_estimation_number": 101,`
	`  "description_text": "Testing PUT method no auth",`
	`  "funnel_custom_funnel1": "1724748016077x923297551481141200",`
	`  "lead_custom_lead": "1725011886770x256041149107739460",`
	`  "name_text": "Team 5",`
	`  "note_list_custom_note": [`
	`  "1678718571924x727404269818871800"`
	`  ],`
	`  "order_number": 0,`
	`  "stage_custom_stage": "1724754233044x396443822862285300",`
	`  "status_option_status": "Lost",`
	`  "visible_to_list_user": [`
	`  "1625125825767x718816571317085800"`
	`  ]`
	`}`

Test steps	1. Send **PUT** `{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"You do not have permission to replace this object"`
Test case ID	**DPU002: Replace a deal - wrong data type values**

Description	It is NOT possible to replace an existing deal with an incorrect data values (e.g., company name value - boolean instead of string; order value - string instead of number)
Precondition	Some deals exist in the application, user is authorized

Test data	UniqueID of a existing deal
	
	`{`
	`  "assignee__user": "1625125825767x718816571317085800",`
	`  "company_name_text": true,`
	`  "deal_value_estimation_number": 100,`
	`  "description_text": "Testing PUT method",`
	`  "funnel_custom_funnel1": "1724748016077x923297551481141200",`
	`  "lead_custom_lead": "1725011886770x256041149107739460",`
	`  "name_text": "Team 5",`
	`  "note_list_custom_note": [`
	`  "1678718571924x727404269818871800"`
	`  ],`
	`  "order_number": "text1",`
	`  "stage_custom_stage": "1724754233044x396443822862285300",`
	`  "status_option_status": "Won",`
	`  "visible_to_list_user": [`
	`  "1625125825767x718816571317085800"`
	`  ]`
	`}`

Test steps	1. Send **PUT**`{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 400 Bad Request • Error message displayed: `"Invalid data for field company_name_text: Expected a string, but got a boolean (original data: true), invalid data for field order_number: Expected a number, but got a string (original data: "string1")"`

DELETE + GET

Test case ID	**DDG001: Delete a deal and check data**
Description	It is possible to delete an existing deal
Precondition	DPOG002 is successfully completed, user is authorized
Test data	UniqueID of the deal created in DPOG002

Test steps	1. Send **DELETE** `{{baseUrl}}/obj/deal/:UniqueID` , set correct UniqueID (taken from step 2, DPOG002) 2. Send **GET** `{{baseUrl}}/obj/deal/:UniqueID`, set correct UniqueID (taken from previous step)
Expected result	Step 1: Successful code: 204 No Content, delete was successful Step 2: Error code: 404 Not Found, `"status": "MISSING_DATA", "message": "Missing object of type deal: object with id {{UniqueId}} does not exist"`

DELETE

Test case ID	**DD001: Delete a deal - no authorization**
Description	It is NOT possible to delete an existing deal without being authorized
Precondition	Some deals exist in the application, user is NOT authorized
Test data	UniqueID of an existing deal
Test steps	1. Send **DELETE** `{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 401 Unauthorized • Error message displayed: `"You do not have permission to delete this object"`
Test case ID	**DD002: Delete a deal - non-existing Id**
Description	It is NOT possible to delete a deal with a non existent Id, and the system returns an appropriate response
Precondition	Some deals exist in the application, user is authorized
Test data	A non-existingn deal´s UniqueID Example: 1725521410379x283042201399783000
Test steps	1. Send **DELETE** `{{baseUrl}}/obj/deal/:UniqueID`
Expected result	• Error code: 404 Not Found • Error message displayed: `"Missing object of type deal: object with id 1725521410379x283042201399783000 does not exist"`