Backend Unit Test

UT_n belongs to test file n in the test folder. For example, UT_1 is related to test file test1-connection.js

Unit Test ID	Module	Details	Expected result
UT_1_1	General	Check MongoDB's connection.	• Database connected.
UT_1_2	General	Check it's the test database and NOT the production database.	• The database name is the test database.
USER - TESTING WITHOUT AUTHENTICATION			
UT_2_1	User	Prevent creating a user without a *firstName* field.	• Return an unauthorized error.
UT_2_2	User	Test if a new user is created successfully with all valid fields	• Return a success status 200 and an authorized token.
UT_2_3	User	Prevent creating a user with a *registered username*.	• Return an unauthorized error message.
UT_2_4	User	Prevent creating a user with a *registered email*.	• Return an unauthorized error message.
UT_2_5	User	Prevent logging in with an *incorrect username* or *password*.	• Return an unauthorized error message.
UT_2_6	User	Test if a user is logged in successfully with the *correct username* and *password*.	• Return a success status 200 and an authorized token.
UT_2_7	User	Prevent showing *profile* without logging in.	• Return an unauthorized error message.
UT_2_8	User	Prevent accessing to *searching for other users* without logging in.	• Return an unauthorized error message.
UT_2_9	User	Prevent accessing to *update the account* without logging in.	• Return an unauthorized error message.
UT_2_10	User	Prevent accessing to *delete the account* without logging in.	• Return an unauthorized error message.
USER - TESTING WITH AUTHENTICATED TOKEN - AFTER LOGIN			
UT_2_11	User	Test if it shows the correct *profile* of the current logged in user.	• Return a success status 200 and profile information.
UT_2_12	User	Test to get all the users from the collection using the GET route.	• Return an array of 1 element, which is the currently registered user.
UT_2_13	User	Test if the user can update his *profile*.	• All changed field is correctly updated. • The user data is updated in the database.
UT_2_14	User	The server successfully finds a user with a given id.	• Return a user if that id existed, and that user has all required fields/attributes.
UT_2_15	User	Test if the user can search for another one.	• Return an array of users that match the search query, and that the user has all required fields/attributes.
UT_2_16	User	Test if the user can *delete* his *account*.	• The user data and his contacts/events/relationships are deleted from the database. • Return a message "`Data is deleted successfully!`"
CONTACT - TESTING WITHOUT AUTHENTICATION			
UT_3_1	Contact	Register for an account to be used.	• Return an authorised token.
UT_3_2	Contact	Prevent accessing to *searching for other users* without logging in.	• Return an unauthorized error message.
UT_3_3	Contact	Prevent accessing to *get all current contacts* without logging in.	• Return an unauthorized error message.
UT_3_4	Contact	Prevent accessing to *search for contacts* without logging in.	• Return an unauthorized error message.
CONTACT - TESTING WITH AUTHENTICATED TOKEN - AFTER LOGIN			
UT_3_5	Contact	Prevent creating a contact without the *firstName* field.	• Return a failed status with the message: "`Missing or invalid firstname!`"
UT_3_6	Contact	Test if a contact is created successfully with all valid fields.	• Return a newly created contact and its information.
UT_3_7	Contact	Test if a contact is created successfully with a userId.	• Return a newly created contact and its information that get imported from the provided userId.
UT_3_8	Contact	Test to get all *owned* contacts from the collection.	• Return an array of owned contacts (2): one created manually and one added from a userId.

UT_3_9	Contact	The server successfully finds an *owned* contact with a given id.	• Return a contact if that id existed, and that contact has all required fields/attributes.
UT_3_10	Contact	Test if the user can update an *owned* contact.	• All changed field is correctly updated. • The contact data is updated in the database.
UT_3_11	Contact	Test if the user can search for contacts.	• Return an array of contacts that match the search query, and that contact has all required fields/attributes.
UT_3_12	Contact	Test if the user can *delete* his *owned* contact.	• Data of the contact is deleted from the database. • Return a message "`Data is deleted successfully!`"
EVENT - TESTING WITHOUT AUTHENTICATION			
UT_4_1	Event	Register for an account and add two contacts to be used.	• Return an authorised token and two contacts are created successfully.
UT_4_2	Event	Prevent accessing to *get all current events* without logging in.	• Return an unauthorized error message.
UT_4_3	Event	Prevent accessing to *search for events* without logging in.	• Return an unauthorized error message.
EVENT - TESTING WITH AUTHENTICATED TOKEN - AFTER LOGIN			
UT_4_4	Event	Prevent creating an event without a *name* field.	• Return a failed status with a message: "`Missing event name or event name contains invalid characters!`"
UT_4_5	Event	Test if an event is created successfully with all valid fields.	• Return a newly created event and its information.
UT_4_6	Event	Test to get all *owned* events from the collection.	• Return an array of owned events (1)
UT_4_7	Event	The server successfully finds an *owned* event with a given id.	• Return an event if that id existed, and that event has all required fields/attributes.
UT_4_8	Event	Test if the user can update an *owned* event.	• All changed field is correctly updated. • The event data is updated in the database.
UT_4_9	Event	Test if the user can search for events.	• Return an array of events that match the search query, and that event has all required fields/attributes.
UT_4_10	Event	Test if the user can *delete* his *owned* event.	• Data of the event is deleted from the database. • Return a message "`Data is deleted successfully!`"
RELATIONSHIP - TESTING WITHOUT AUTHENTICATION			
UT_5_1	Relationship	Register for an account and add two contacts to be used.	• Return an authorised token and two contacts are created successfully.
UT_5_2	Relationship	Prevent accessing to *get all current relationships* without logging in.	• Return an unauthorized error message.
UT_5_3	Relationship	Prevent accessing to *search for relationships* without logging in.	• Return an unauthorized error message.
RELATIONSHIP - TESTING WITH AUTHENTICATED TOKEN - AFTER LOGIN			
UT_5_4	Relationship	Prevent creating a relationship without a *people* field or invalid contactId in the *people* field.	• Return a failed status with a message: "`Require 2 people in a relationship!`"
UT_5_5	Relationship	Prevent creating a relationship with duplicated contactId in the *people* field.	• Return a failed status with a message: "`Invalid contact Ids or this relationship has existed!`"
UT_5_6	Relationship	Test if a relationship is created successfully with all valid fields.	• Return a newly created relationship and its information.
UT_5_7	Relationship	Prevent creating a relationship that has already been created.	• Return a failed status with a message: "`Invalid contact Ids or this relationship has existed!`"
UT_5_8	Relationship	Test to get all *owned* relationships from the collection.	• Return an array of owned relationships (1)
UT_5_9	Relationship	The server successfully finds an *owned* relationship with a given id.	• Return a relationship if that id existed, and that relationship has all required fields/attributes.
UT_5_10	Relationship	Test if the user can update an *owned* relationship.	• All changed field is correctly updated. • The relationship data is updated in the database.
UT_5_11	Relationship	Test if the user can search for relationships.	• Return an array of relationships that match the search query, and that relationship has all required fields/attributes.

UT_5_12	Relationship	Test if the user can *delete* his *owned* relationship.	- Data of the relationship is deleted from the database. - Return a message "`Data is deleted successfully!`"