| Test number | Resource | Task | Files | Input | Expected Output | Actual Output |
|---|---|---|---|---|---|---|
| | | **Client side CRUD operations Development Side** | | | | |
| 1 | expenses | view all expenses | expense_routes, expense_controller | GET on http://localhost:3000/expenses | Response includes all expenses<br>Status 200 | Response included all expenses<br>Status 200 |
| 2 | expenses | view by id | expense_routes, expense_controller | GET on http://localhost:3000/expenses/ :id | Response to include one expense with id | Error regarding date object |
| 3 | expenses | view by id | expense_routes, expense_controller | GET on http://localhost:3000/expenses /:id | Response to include one expense with id | Response to include one expense with id |
| 4 | expenses | create new expense | expense_routes, expense_controller | POST on http://localhost:3000/expenses | Return new expense with status 201 | Return new expense with status 201, but date object unexpected formatting. |
| 5 | expenses | create new expense | expense_routes, expense_controller | POST on http://localhost:3000/expenses | Return new expense with status 201 | Return new expense with status 201 |
| 6 | expenses | delete expense | expense_routes, expense_controller | DELETE on http://localhost:3000/expenses/:id | Delete expense by id, return successful message status 204 | Removed expense from database, and successful message |
| 7 | expenses | update expense | expense_routes, expense_controller | PUT on http://localhost:3000/expenses/ :id | Update expense by id, return updated expense with status 204 | Response with updated expense data |
| | | **Testing register route development** | | | | |
| 1 | auth/register | register new user | auth_routes, auth_controller | POST on http://localhost:3000/auth/register with correct details | User object with username,email,password,salt and hash field. Also return satus 200<br>including id generated by the mongo db | Return status as 200 with user object as we expected<br>.(username,email,password,salt,hash and id) |
| 2 | auth/register | register new user | auth_routes,auth_controller | POST on http://localhost:3000/auth/register with missing one field | Return error with status 400 | Return error with status 400 |
| 3 | auth/login | login new user | auth_routes, auth_controller | POST on http://localhost:3000/auth/login<br>with correct username,email and password | Return user object with status 200 | Returned "ReferenceError: req is not defined" with status 500 |
| 4 | auth/login | login new user | auth_routes, auth_controller | POST on http://localhost:3000/auth/login<br>with correct username,email and password | Return user object with status 200 | Returned User object with status 200 |
| 5 | auth/logout | logout user | auth_routes, auth_controller | GET on http://localhost:3000/auth/logout | Return 200 status | Returned error:"Cannot GET /auth/logout " with status 404 |
| 6 | auth/logout | logout user | auth_routes, auth_controller | GET on http://localhost:3000/auth/logout | Return 200 status | Return 200 status |
| 7 | auth/register | register new user | auth_routes,auth_controller | POST on http://localhost:3000/auth/register.<br>with already existing username | Return status 500 and error object with error message as "error": {<br>"name": "UserExistsError",<br>"message": "A user with the given username is already registered"<br>}" | Return status 500 and error object with error message as "error": {<br>"name": "UserExistsError",<br>"message": "A user with the given username is already registered"<br>}" |
| 8 | auth | update current user | auth_controller | PUT on http://localbost:3000/auth | update current user details, status 200 | update current user details, status 200 (once testing correct user id) |
| 9 | auth | delete user | auth_controller | DELTEE on http://localbost:3000/auth | delete current user details, status 200 | delete current user details, status 200 |
| | | **Testing authentication on development** | | | | |
| 1 | expenses | authenticate user in create expense | expense_routes, expense_controller | POST on http://localhost:3000/expenses<br>trying to create new expense without login user | Return 403 status-forbidden | Return 403-forbidden |
| 2 | expenses | authenticate user in delete action | expense_routes, expense_controller | DELETE on http://localhost:3000/expenses/:id<br>trying to delete expense without login user | Return 403 status-forbidden | Return 403 status-forbidden |
| 3 | expenses | authenticate user in update action | expense_routes, expense_controller | PUT on http://localhost:3000/expenses/:id<br>trying to update expense without login user | Return 403 status-forbidden | Return 403 status-forbidden |
| 4 | expenses | authenticate user in viewing all expense | expense_routes, expense_controller | GET on http://localhost:3000/expenses<br>trying to get all expenses without login | Return 403 status-forbidden | Return 403 status-forbidden |
| 5 | expenses | authenticate user in view a single expense | expense_routes, expense_controller | GET on http://localhost:3000/expenses/:id<br>trying to get single expense without login | | |
| 6 | expenses | logged-in user view expenses, only for current l | auth_controller, expenses | GET on http://localhost:300/expenses | return only expenses made by logged in user | Return 403 status-forbidden |
| | | **Testing on production side** | | | | |
| 1 | auth/register | register new user with<br>already existing username | auth_routes, auth_controller | POST on https://obscure-basin-70004.herokuapp.com/auth/register | Return error message with status 500 | Return status of 500 and error object with error message as "error": {<br>"name": "UserExistsError",<br>"message": "A user with the given username is already registered"} |
| 2 | auth/register | register new user with user details | auth_routes, auth_controller | POST on https://obscure-basin-70004.herokuapp.com/auth/register | User object with username,email,password,salt and hash field. Also return satus 200<br>including id generated by the mongo db | Return status as 200 with user object as we expected .<br>(username,email,password,salt,hash and id) |
| 3 | expenses | add new expense to logged in user | expense_routes,expense_controller,auth routes | POST https://obscure-basin-70004.herokuapp.com/expenses | Status code 201 with expense details that we are added and the username | Status code 201 with expense details that we are added and the username |
| 4 | expenses | add new expense without login | expense_routes,expense_controller,auth_routes | POST https://obscure-basin-70004.herokuapp.com/expenses | 403 forbidden | 403 forbidden |
| 5 | expenses | get all expenses of loggedin user | expense_routes,expense_controller,auth_routes | GET https://obscure-basin-70004.herokuapp.com/expenses | 200 status with all the expenses that is created by the loggedIn user | 200 status with all the expenses that is created by the loggedIn user |
| 6 | expenses | delete an expense by id (logged in user) | expense_routes,expense_controller,auth_routes | DELETE https://obscure-basin-70004.herokuapp.com/expenses/'id' | remove expense with defined id from database | remove expense with defined id from database, confirmed in mogoDB Atlas |
| 7 | expenses | update an existing expense (logged in user) | expense_routes,expense_controller,auth_routes | UPDATE https://obscure-basin-70004.herokuapp.com/expenses/'id' | update data for expense with defined id, status 200 | update data for expense with defined id, status 200, confirmed in mongoDB Atlas |