CS673 Software Engineering

Team 3 - PlanningJam

Software Test Document

Team Member	Role(s)	Signature	Date
Jason	QA	*JasonLee*	09/21/2025
David Metraux	Team Leader	*David Metraux*	*9/21/2025*
Ashley Sachdeva	Requirements Leader	*Ashley Sachdeva*	9/21/2025
Donjay Barit	Configuration Leader	*Donjay Barit*	9/21/2025
Haolin	Design and Implementation Leader	*Haolin Yang*	*9/21/2025*

Revision history

Version	**Author**	**Date**	**Change**
0.0.0	**Ashley Sachdeva**	**09/21/2025**	**Added additional frontend testing steps**
1.0.0	**Ashley Sachdeva**	**10/5/2025**	**Added automated test section and extended frontend**

			manual testing section
1.0.1	**David Metraux**	**10/6/2025**	**Added manual tests for friends page**
1.0.2	**Jason Lee**	**10/5/2025**	**Added to testing summary, backend automated testing report, and testing metrics**
1.0.3	**Donjay Barit**	**10/6/2025**	**Added manual test for plans api query**

• Testing Summary

In this section, you will summarize what was tested, who is involved in testing, when to test, testing techniques used, and testing result. You may have the following tests

- Unit Testing
- Integration testing
- System Testing
- Acceptance Testing
- Regression Testing
  On elements in the frontend, you often have to check if other

What was tested

- Authentication & User APIs (registration, login, profile, user listing)
- Friends API (send/respond/remove friend requests, reciprocal auto-accept logic)
- Plans API (CRUD operations for plans in MongoDB)
- Frontend authentication flows (login, signup, logout, token expiration)
- Frontend plans functionality (create, edit, delete)
- UI responsiveness (cross-browser and viewport testing)

Who is involved

- Backend testing: Team members responsible for Django REST Framework API
- Frontend testing: Team members responsible for React + Vite application

- Integration/system tests: Performed jointly via Postman and browser

<u>When testing occurs</u>

- Unit and integration tests: Continuous, as features are developed
- System/acceptance tests: Before sprint demos and release candidates
- Regression tests: Run after significant refactors, migrations, or merges

<u>Testing techniques used</u>

- Unit Testing: Django pytest for backend, Vitest + React Testing Library for frontend
- Integration Testing: Postman API collection tests, end-to-end flows
- System Testing: Browser/manual exploratory testing
- Acceptance Testing: Full workflow scenarios (signup → login → add friend → create plan)
- Regression Testing: Automated test suites rerun on feature updates

<u>Test Environment</u>

- Backend: Django, DRF, SimpleJWT, django-mongodb-backend
- Database: PostgreSQL (users/friends), MongoDB (plans collection)
- Frontend: React + Vite + TypeScript
- Tools: Postman, Vitest, pytest, React Testing Library

• Manual Testing Report

In this section, you will give a detailed description of each manual test case performed and the result. If this is a previous You shall list what are existing tests developed in the previous semester and what are new tests developed currently.

Here is a sample template that can be used for each test case. For system tests or acceptance tests, you may also include some screenshots.

- Test case ID, name

- New or old:
- Test items: (what do you test)
- Test priority (high/medium/low)
- Dependencies (to other test case/requirement if any):
- Preconditions: (if any)
- input data:
- Test steps:
- Postconditions:
- Expected output:
- Actual output:
- Pass or Fail:
- Bug id/link: (this should link to your github issue id)
- Additional notes:

(You can use an additional spreadsheet for this section as well)

- Test case 1: **Browser testing**
- Test items: Using Chrome or other devtools, verify that items are displayed correctly on different browsers and on different viewpoint sizes..
- Test priority (high)
- Dependencies (to other test case/requirement if any): N/A
- Preconditions: (if any)
- input data: Depends on the item
- Test steps:

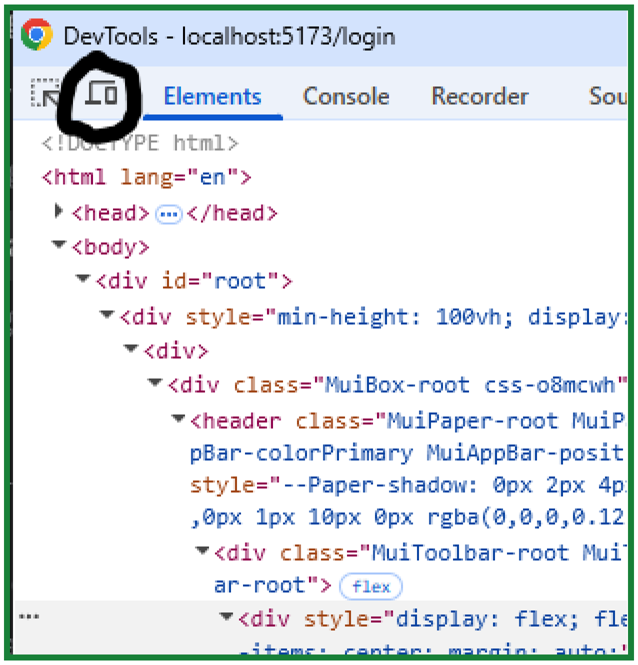
  1. Load location where element will be viewable on a browser

  2. Determine it is still can be viewed and interacted with at a variety of viewpoint sizes (via toggle device toolbar)

  3. Attempt again on a different browser
- Postconditions:
- Expected output:

  The element to look as desired
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes:

  This is how I can check various viewport sizes



```
<!DOCTYPE html>
<html lang="en">
  <head> ··· </head>
  <body>
    <div id="root">
      <div style="min-height: 100vh; display:
        <div>
          <div class="MuiBox-root css-o8mcwh"
            <header class="MuiPaper-root MuiP
            pBar-colorPrimary MuiAppBar-posit
            style="--Paper-shadow: 0px 2px 4p
            ,0px 1px 10px 0px rgba(0,0,0,0.12
              <div class="MuiToolbar-root Mui
              ar-root"> flex
...             <div style="display: flex; fle
                -items: center; margin: auto;'
```

- Test case 2: **Login**
- Test items: Login flow for user
- Test priority: High
- Dependencies (to other test case/requirement if any): Must have an existing account
- Preconditions: (if any): user is registered
- input data: email: username: asach8888 pwd: mimic123
- Test steps:
  - Navigate to login
  - Enter username
  - Enter password
  - Click login
  - Clicking signup will redirect you to the signup page
- Postconditions: Successfully login, redirect to home page
- Expected output:
- Actual output:
- Pass or Fail:
- Bug id/link: (this should link to your github issue id)
- Additional notes:
  - If a user enters incorrect username/password correct error message is shown
  - If a user tries to access login/signup from there they get redirected to homepage until they logout

- Test case 3: **Signup**
- Test items: Signup flow for user
- Test priority: High
- Dependencies (to other test case/requirement if any): none
- Preconditions: (if any):
- input data: email: username: xxx, email: xxx@gmail.com, pwd: mimic12345, confirm pws
- Test steps:
  - Navigate to signup
  - Fill out form
  - Click sign up
  - Clicking Login will redirect you to login
- Postconditions: Successfully registered and logged in, redirect to home page
- Expected output:
- Actual output:
- Pass or Fail:
- Bug id/link: (this should link to your github issue id)
- Additional notes:
  - Filling out an existing username will fail validation
  - Incorrect email format will fail validation
  - Passwords being too short, common, or all numerical will fail validation
  - If a user tries to access login/signup from there they get redirected to homepage until they logout

- Test case 4: **Logout**
- Test items: Logout flow for user
- Test priority: medium
- Dependencies (to other test case/requirement if any): none
- Preconditions: (if any):
- input data: None
- Test steps:
  - Navigate to /logout
- Postconditions: Successfully logged out, redirected to login page
- Expected output:
- Actual output:
- Pass or Fail:

- Bug id/link: (this should link to your github issue id)
- Additional notes:
  - Navigating to home page redirects them back to login

- Test case 5: **Token Expiration**
- Test items: Expired token for user authentication
- Test priority: high
- Dependencies (to other test case/requirement if any): Logged in user
- Preconditions: (if any): User is logged in for 60 minutes
- input data: None
- Test steps:
  - If the user is logged in for 60+ minutes on the application they will be logged out
  - Login for 1 hour or change JWT in settings.py to something shorter like 10 seconds
  - Refresh screen, should be redirected to login
- Postconditions: Successfully logged out, redirected to login page
- Expected output:
- Actual output:
- Pass or Fail:
- Bug id/link: (this should link to your github issue id)
- Additional notes:
  - Navigating to home page redirects them back to login

Test Case 6: **Add Plan**

- Test items: Add new plan functionality
- Test priority: High
- Dependencies (to other test case/requirement if any): User must be logged in
- Preconditions (if any): Valid authenticated session, user on /plans/add
- Input data: plan title, description, location name, address, city, state, zipcode, start and end time
- Test steps:
  - Click “Add Plan” button on home page
  - Fill in only some of the data
  - Click “Save”
  - Ensure error message shows to fill out other input data
  - Click cancel button
  - Ensure it redirects to home page
  - Fill rest of input data

    - Click Save button
- Postconditions:
    - New plan is created and visible on the home page
- Expected output: The new plan appears immediately in the plans list with correct information
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes: UI refreshes automatically after adding a plan; no manual reload needed.

Test Case 7: **Edit Plan**

- Test items: Edit new plan functionality
- Test priority: High
- Dependencies (to other test case/requirement if any): User must be logged in
- Preconditions (if any): Valid authenticated session, user has plan already created
- Input data: plan title, description, location name, address, city, state, zipcode, start and end time
- Test steps:
    - Click “Edit Plan” button on plan created by user
    - The plan inputs are pre-filled
    - Click cancel button
    - Ensure it redirects to home page
    - Edit some input data
    - Click Save button
- Postconditions:
    - Edited plan is created and visible on the home page
- Expected output: The edited plan appears immediately in the plans list with correct information
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes: UI refreshes automatically after editing a plan; no manual reload needed.

Test Case 8: **Delete Plan**

- Test items: delete new plan functionality
- Test priority: High
- Dependencies (to other test case/requirement if any): User must be logged in

- Preconditions (if any): Valid authenticated session, user has plan already created
- Input data: none
- Test steps:
    - Click “Delete Plan” button on plan created by user
    - Ensure a confirm deletion modal shows
    - Click cancel button
    - Ensure modal is closed
    - Click delete plan button on plan again
    - Click confirm delete button
- Postconditions:
    - Plan is no longer visible on home page
- Expected output: The plan disappears immediately in the plans list
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes: UI refreshes automatically after deleting a plan; no manual reload needed.

Test Case 9: **Send Friend Request**

- Test items: Friend Request initiation
- Test priority: High
- Dependencies (to other test case/requirement if any): Two registered users
- Preconditions (if any): 1st user logged in
- Input data: none
- Test steps:
    - POST /api/friends/request/<bob_id>/
- Postconditions:
    - A friend request is sent
- Expected output: Success 201 Created status = pending
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes:

Test Case 10: **Respond to Friend Request**

- Test items: Accepting Friend Request
- Test priority: High
- Dependencies (to other test case/requirement if any): Pending request exists
- Preconditions (if any): 2nd user logged in
- Input data: none
- Test steps:
    - POST /api/friends/respond/<request_id>/ with {"action":"accept"}
- Postconditions:
    - A friendship is bloomed
- Expected output: 200 OK, status = accepted
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes:

Test Case 11: **Remove Friend**

- Test items: Friend Removal
- Test priority: High
- Dependencies (to other test case/requirement if any):
- Preconditions (if any): A souring friendship exists
- Input data: none
- Test steps:
    - DELETE /api/friends/remove/<request_id>/
- Postconditions:
    - A broken heart
- Expected output: 204 No Content
- Actual output: N/A
- Pass or Fail: N/A
- Bug id/link: N/A
- Additional notes:

Test Case 12: **Friends page**

- Test items: Friend functions
- Test priority: High

- Dependencies (to other test case/requirement if any):
- Preconditions (if any): At least two users in the program
- Input data: Two users
- Test steps:
    - Sign in two users, each in a different browser
    - have a user send a friend request to another user
    - Have the receiving user delete the friend request
    - Have the first user send a friend request again
    - Have the receiving user accept the friend request
    - Have either user delete the other as a friend.
- Postconditions:
- Expected output: The UI to change smoothly
- Actual output: So far, I often need to click twice for the UI to update
- Pass or Fail: Sort of fail
- Bug id/link: N/A
- Additional notes:

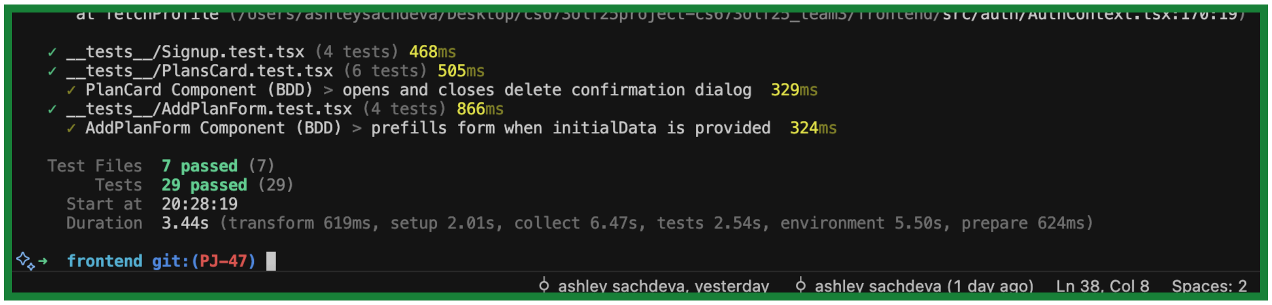
Test Case 13: **Plans API filter query**

- Test items: Query get plans api
- Test priority: High
- Dependencies (to other test case/requirement if any):
- Preconditions (if any): Plans created by two or more users and having a friends relationship with each other or none
- Input data: Set optional values for query filters: start time, end time, friends
- Test steps:
    - Send a GET request to /api/plans?start_time=<stime>&end_time=<etime>&friends=1
- Postconditions: None
- Expected output: A list of plans that is created by the user or their friends with a date range
- Actual output: The list of expected plans for the given user and their friends between the date ranges
- Pass or Fail: Pass
- Bug id/link: N/A
- Additional notes:

• Automated Testing Report

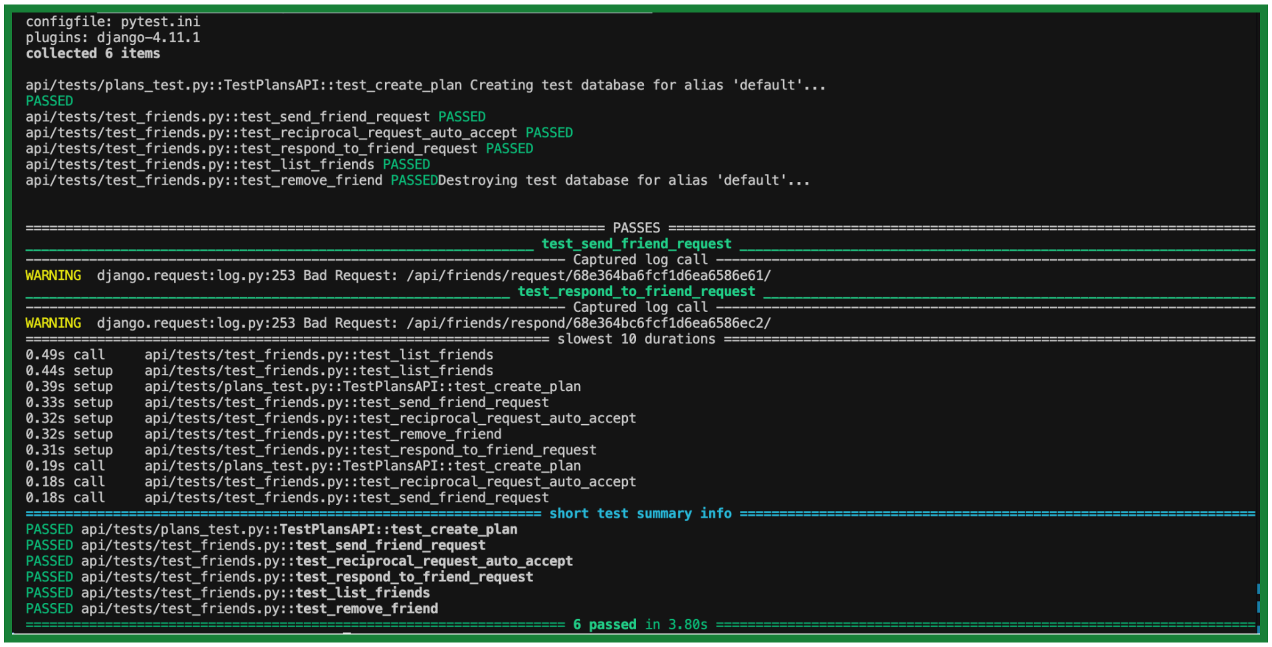
Describe briefly the automated testing you have done, including where the test code resides in your code repository, what test frameworks are used, and the screen shots or generated testing report.

For the frontend of the application, all test files are stored under the __tests__/ directory. Each component and page (e.g., SignUp, PlansHeader, PlanCard, AddPlanForm, AddPlanPage, Home) has its corresponding test file (e.g., SignUp.test.tsx). We use the Vitest test runner and react testing library to both mock and assert. These tools allow realistic user interaction testing for React components while maintaining isolation from backend dependencies.



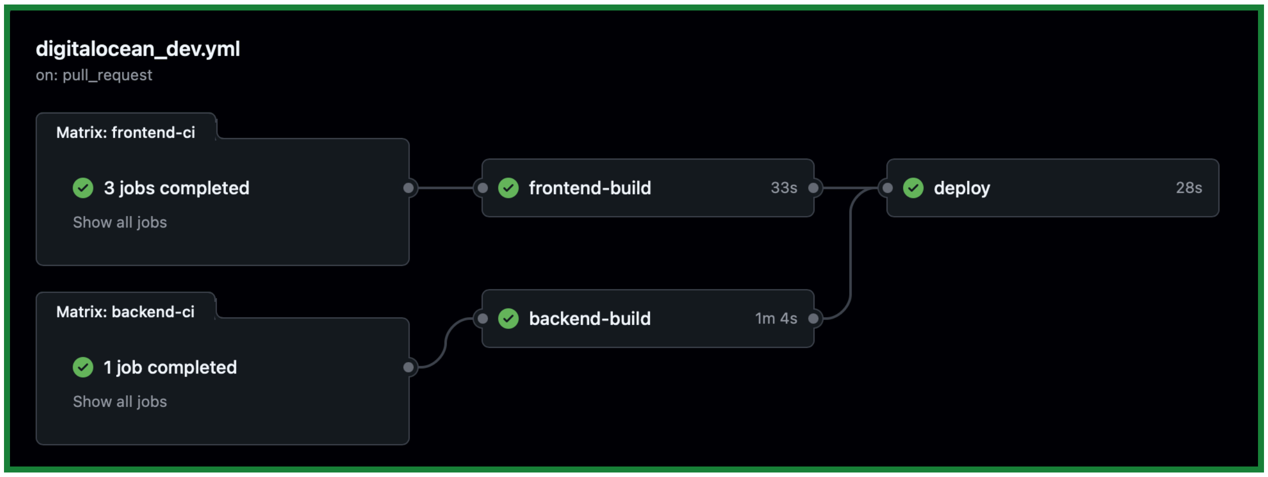
```
    at fetchProfile (/Users/ashleysachdeva/Desktop/cs6750tf25project-cs6750tf25_team3/frontend/src/auth/AuthContext.tsx:170:19)

 ✓ __tests__/Signup.test.tsx (4 tests) 468ms
 ✓ __tests__/PlansCard.test.tsx (6 tests) 505ms
   ✓ PlanCard Component (BDD) > opens and closes delete confirmation dialog  329ms
 ✓ __tests__/AddPlanForm.test.tsx (4 tests) 866ms
   ✓ AddPlanForm Component (BDD) > prefills form when initialData is provided  324ms

 Test Files  7 passed (7)
      Tests  29 passed (29)
   Start at  20:28:19
   Duration  3.44s (transform 619ms, setup 2.01s, collect 6.47s, tests 2.54s, environment 5.50s, prepare 624ms)

  frontend git:(PJ-47) █
                                  ashley sachdeva, yesterday   ashley sachdeva (1 day ago)   Ln 38, Col 8   Spaces: 2
```

Backend and frontend tests validate core API and UI behavior: backend tests (backend/api/tests/, e.g., test_friends.py, plans_test.py) exercise friend relationships and plans CRUD using pytest + pytest-django with the DRF test client, while frontend tests cover React components and user interactions using Vitest + React Testing Library. The Postman collection was updated (postman/planning-jam-api-postman.json) to reflect API examples, and optional machine readable reports can be generated with pytest in the terminal



```
configfile: pytest.ini
plugins: django-4.11.1
collected 6 items

api/tests/plans_test.py::TestPlansAPI::test_create_plan Creating test database for alias 'default'...
PASSED
api/tests/test_friends.py::test_send_friend_request PASSED
api/tests/test_friends.py::test_reciprocal_request_auto_accept PASSED
api/tests/test_friends.py::test_respond_to_friend_request PASSED
api/tests/test_friends.py::test_list_friends PASSED
api/tests/test_friends.py::test_remove_friend PASSEDDestroying test database for alias 'default'...


========================================================================= PASSES ==========================================================================
_________________________________________________________________ test_send_friend_request _________________________________________________________________
--------------------------------------------------------------------- Captured log call ---------------------------------------------------------------------
WARNING  django.request:log.py:253 Bad Request: /api/friends/request/68e364ba6fcf1d6ea6586e61/
______________________________________________________________ test_respond_to_friend_request ______________________________________________________________
--------------------------------------------------------------------- Captured log call ---------------------------------------------------------------------
WARNING  django.request:log.py:253 Bad Request: /api/friends/respond/68e364bc6fcf1d6ea6586ec2/
=================================================================== slowest 10 durations ===================================================================
0.49s call     api/tests/test_friends.py::test_list_friends
0.44s setup    api/tests/test_friends.py::test_list_friends
0.39s setup    api/tests/plans_test.py::TestPlansAPI::test_create_plan
0.33s setup    api/tests/test_friends.py::test_send_friend_request
0.32s setup    api/tests/test_friends.py::test_reciprocal_request_auto_accept
0.32s setup    api/tests/test_friends.py::test_remove_friend
0.31s setup    api/tests/test_friends.py::test_respond_to_friend_request
0.19s call     api/tests/plans_test.py::TestPlansAPI::test_create_plan
0.18s call     api/tests/test_friends.py::test_reciprocal_request_auto_accept
0.18s call     api/tests/test_friends.py::test_send_friend_request
================================================================= short test summary info =================================================================
PASSED api/tests/plans_test.py::TestPlansAPI::test_create_plan
PASSED api/tests/test_friends.py::test_send_friend_request
PASSED api/tests/test_friends.py::test_reciprocal_request_auto_accept
PASSED api/tests/test_friends.py::test_respond_to_friend_request
PASSED api/tests/test_friends.py::test_list_friends
PASSED api/tests/test_friends.py::test_remove_friend
==================================================================== 6 passed in 3.80s ====================================================================
```

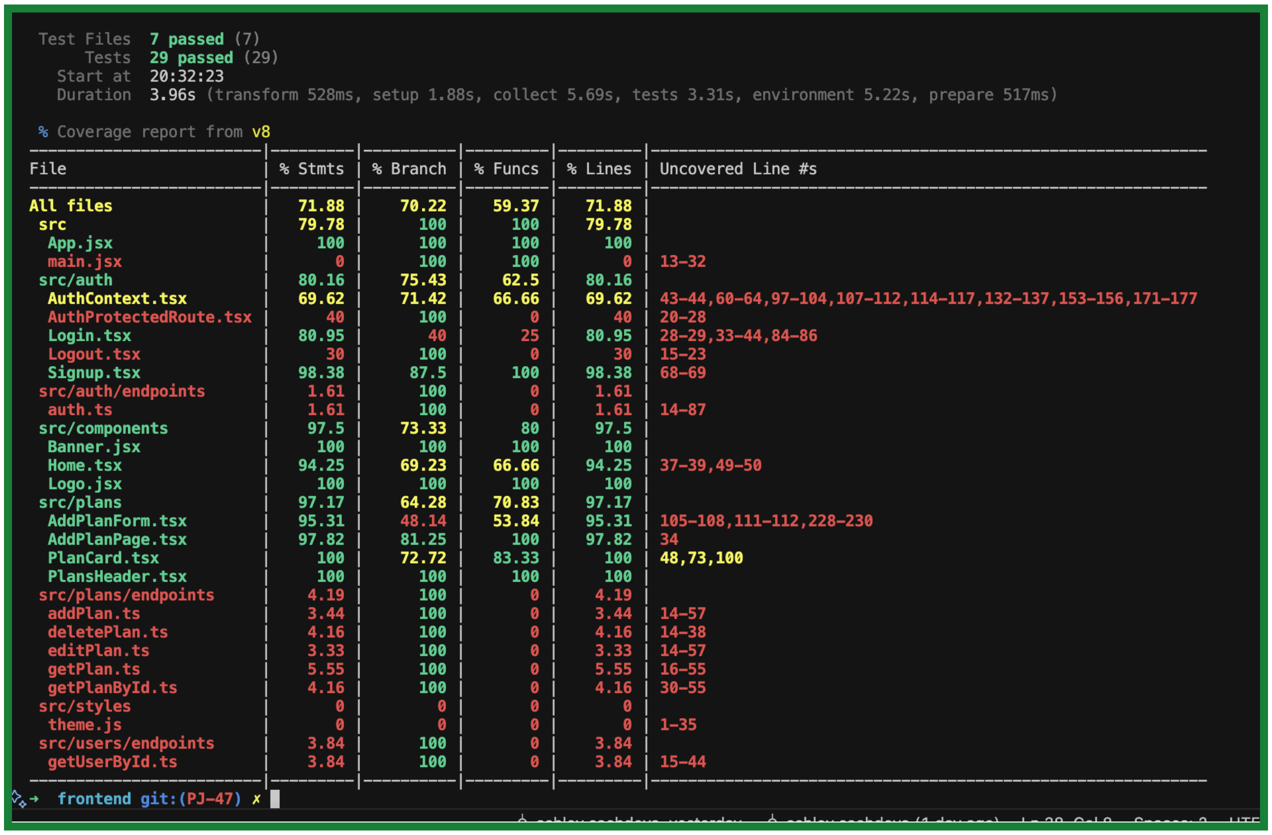
In addition, automated testing and deployment are integrated using GitHub Actions and DigitalOcean workflows (`digitalocean.yml`, `digitalocean_dev.yml`). On every PR, the CI pipeline runs `pytest` for backend tests and `vitest` for frontend unit tests. Only after all tests pass does the workflow trigger deployment to the DigitalOcean development environment (via `digitalocean_dev.yml`) or production environment (`digitalocean.yml`), ensuring no broken code reaches production.



● Testing Metrics

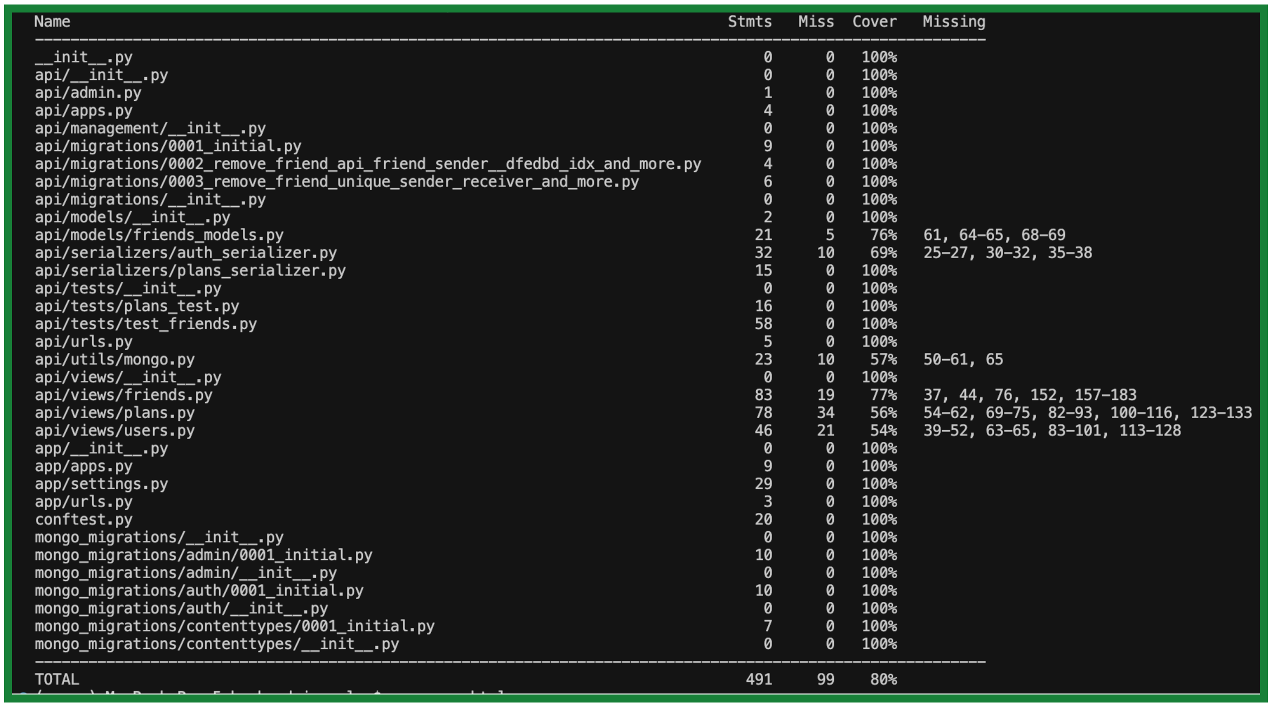
In this section, you shall report any metrics used for the evaluation, e.g. # of test cases, test coverage, defects rate, etc.

For the frontend we have above 70% coverage on statements, branches and lines and slightly below 60% coverage on functions.



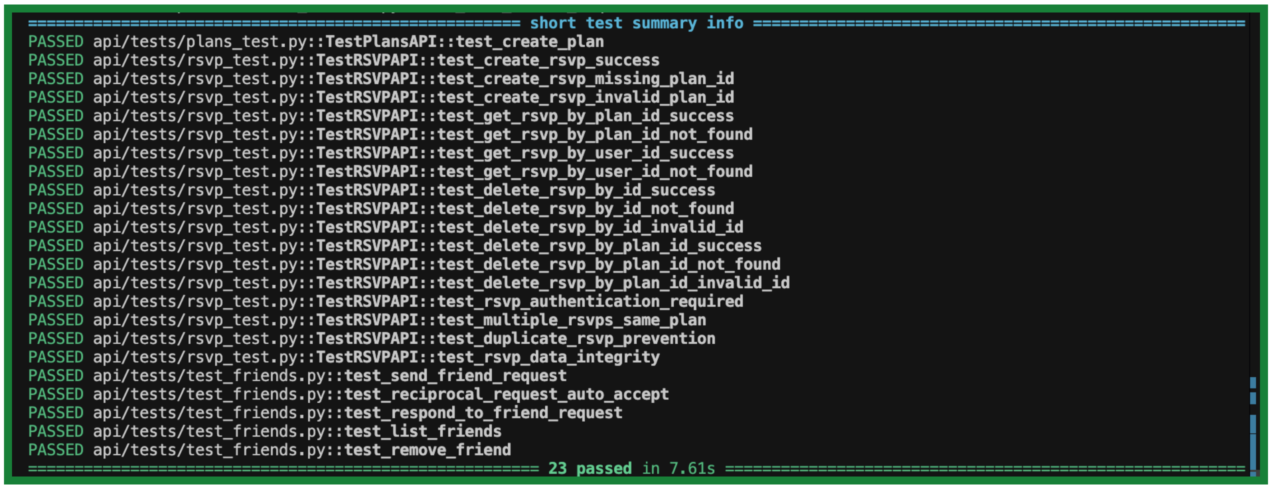
```
 Test Files  7 passed (7)
      Tests  29 passed (29)
   Start at  20:32:23
   Duration  3.96s (transform 528ms, setup 1.88s, collect 5.69s, tests 3.31s, environment 5.22s, prepare 517ms)

 % Coverage report from v8
-------------------------|---------|----------|---------|---------|-------------------------------------------------------------
File                     | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-------------------------|---------|----------|---------|---------|-------------------------------------------------------------
All files                |   71.88 |    70.22 |   59.37 |   71.88 |
 src                     |   79.78 |      100 |     100 |   79.78 |
  App.jsx                |     100 |      100 |     100 |     100 |
  main.jsx               |       0 |      100 |     100 |       0 | 13-32
 src/auth                |   80.16 |    75.43 |    62.5 |   80.16 |
  AuthContext.tsx        |   69.62 |    71.42 |   66.66 |   69.62 | 43-44,60-64,97-104,107-112,114-117,132-137,153-156,171-177
  AuthProtectedRoute.tsx |      40 |      100 |       0 |      40 | 20-28
  Login.tsx              |   80.95 |       40 |      25 |   80.95 | 28-29,33-44,84-86
  Logout.tsx             |      30 |      100 |       0 |      30 | 15-23
  Signup.tsx             |   98.38 |     87.5 |     100 |   98.38 | 68-69
 src/auth/endpoints      |    1.61 |      100 |       0 |    1.61 |
  auth.ts                |    1.61 |      100 |       0 |    1.61 | 14-87
 src/components          |    97.5 |    73.33 |      80 |    97.5 |
  Banner.jsx             |     100 |      100 |     100 |     100 |
  Home.tsx               |   94.25 |    69.23 |   66.66 |   94.25 | 37-39,49-50
  Logo.jsx               |     100 |      100 |     100 |     100 |
 src/plans               |   97.17 |    64.28 |   70.83 |   97.17 |
  AddPlanForm.tsx        |   95.31 |    48.14 |   53.84 |   95.31 | 105-108,111-112,228-230
  AddPlanPage.tsx        |   97.82 |    81.25 |     100 |   97.82 | 34
  PlanCard.tsx           |     100 |    72.72 |   83.33 |     100 | 48,73,100
  PlansHeader.tsx        |     100 |      100 |     100 |     100 |
 src/plans/endpoints     |    4.19 |      100 |       0 |    4.19 |
  addPlan.ts             |    3.44 |      100 |       0 |    3.44 | 14-57
  deletePlan.ts          |    4.16 |      100 |       0 |    4.16 | 14-38
  editPlan.ts            |    3.33 |      100 |       0 |    3.33 | 14-57
  getPlan.ts             |    5.55 |      100 |       0 |    5.55 | 16-55
  getPlanById.ts         |    4.16 |      100 |       0 |    4.16 | 30-55
 src/styles              |       0 |        0 |       0 |       0 |
  theme.js               |       0 |        0 |       0 |       0 | 1-35
 src/users/endpoints     |    3.84 |      100 |       0 |    3.84 |
  getUserById.ts         |    3.84 |      100 |       0 |    3.84 | 15-44
-------------------------|---------|----------|---------|---------|-------------------------------------------------------------
➜  frontend git:(PJ-47) ✗ 
```

There is around 80% coverage on backend statements currently:



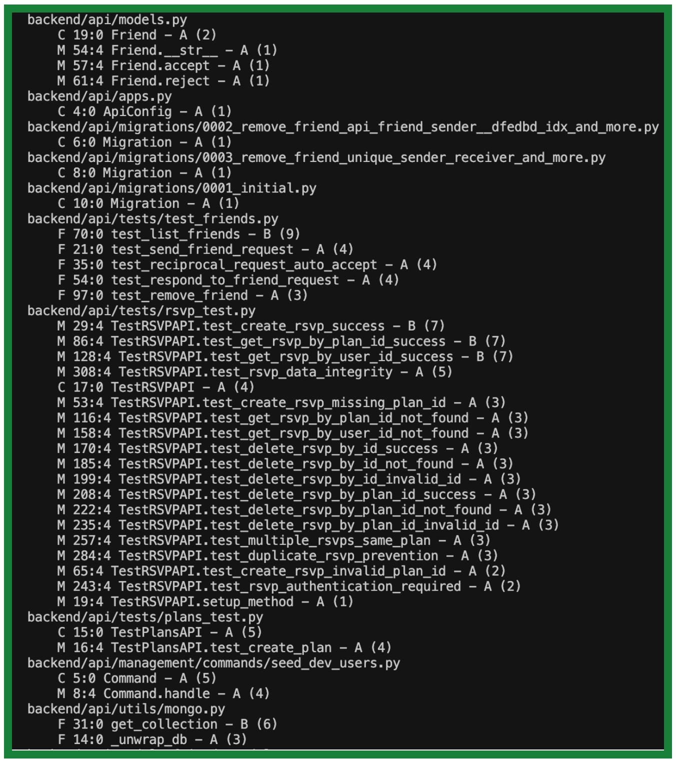
```
Name                                                                         Stmts   Miss  Cover   Missing
-----------------------------------------------------------------------------------------------------------
__init__.py                                                                      0      0   100%
api/__init__.py                                                                  0      0   100%
api/admin.py                                                                     1      0   100%
api/apps.py                                                                      4      0   100%
api/management/__init__.py                                                       0      0   100%
api/migrations/0001_initial.py                                                   9      0   100%
api/migrations/0002_remove_friend_api_friend_sender__dfedbd_idx_and_more.py      4      0   100%
api/migrations/0003_remove_friend_unique_sender_receiver_and_more.py             6      0   100%
api/migrations/__init__.py                                                       0      0   100%
api/models/__init__.py                                                           2      0   100%
api/models/friends_models.py                                                    21      5    76%   61, 64-65, 68-69
api/serializers/auth_serializer.py                                              32     10    69%   25-27, 30-32, 35-38
api/serializers/plans_serializer.py                                             15      0   100%
api/tests/__init__.py                                                            0      0   100%
api/tests/plans_test.py                                                         16      0   100%
api/tests/test_friends.py                                                       58      0   100%
api/urls.py                                                                      5      0   100%
api/utils/mongo.py                                                              23     10    57%   50-61, 65
api/views/__init__.py                                                            0      0   100%
api/views/friends.py                                                            83     19    77%   37, 44, 76, 152, 157-183
api/views/plans.py                                                              78     34    56%   54-62, 69-75, 82-93, 100-116, 123-133
api/views/users.py                                                              46     21    54%   39-52, 63-65, 83-101, 113-128
app/__init__.py                                                                  0      0   100%
app/apps.py                                                                      9      0   100%
app/settings.py                                                                 29      0   100%
app/urls.py                                                                      3      0   100%
conftest.py                                                                     20      0   100%
mongo_migrations/__init__.py                                                     0      0   100%
mongo_migrations/admin/0001_initial.py                                          10      0   100%
mongo_migrations/admin/__init__.py                                               0      0   100%
mongo_migrations/auth/0001_initial.py                                           10      0   100%
mongo_migrations/auth/__init__.py                                                0      0   100%
mongo_migrations/contenttypes/0001_initial.py                                    7      0   100%
mongo_migrations/contenttypes/__init__.py                                        0      0   100%
-----------------------------------------------------------------------------------------------------------
TOTAL                                                                          491     99    80%
```

Along with 23 passed tests (not including the 6 automated CI pipeline tests run to each Pull Request ticket)

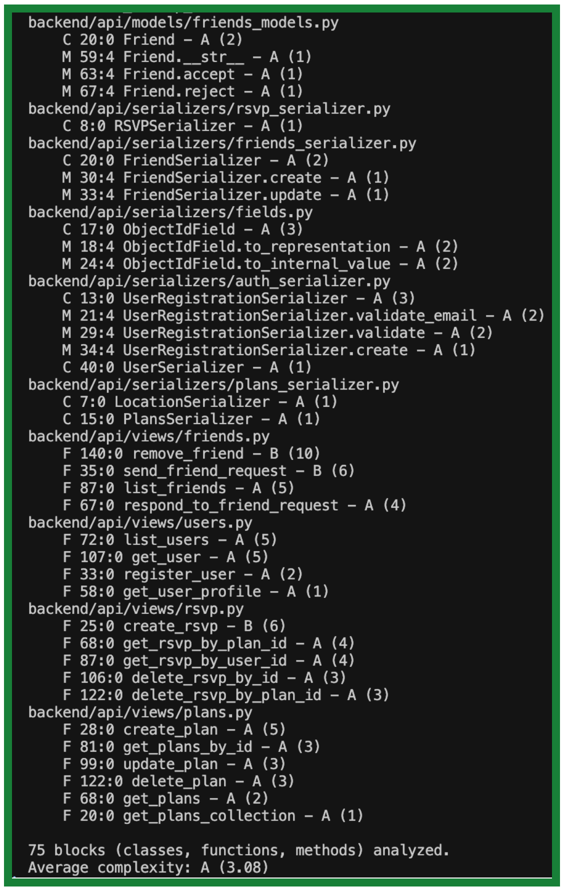


```
======================================================= short test summary info ========================================================
PASSED api/tests/plans_test.py::TestPlansAPI::test_create_plan
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_create_rsvp_success
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_create_rsvp_missing_plan_id
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_create_rsvp_invalid_plan_id
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_get_rsvp_by_plan_id_success
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_get_rsvp_by_plan_id_not_found
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_get_rsvp_by_user_id_success
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_get_rsvp_by_user_id_not_found
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_id_success
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_id_not_found
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_id_invalid_id
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_plan_id_success
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_plan_id_not_found
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_delete_rsvp_by_plan_id_invalid_id
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_rsvp_authentication_required
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_multiple_rsvps_same_plan
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_duplicate_rsvp_prevention
PASSED api/tests/rsvp_test.py::TestRSVPAPI::test_rsvp_data_integrity
PASSED api/tests/test_friends.py::test_send_friend_request
PASSED api/tests/test_friends.py::test_reciprocal_request_auto_accept
PASSED api/tests/test_friends.py::test_respond_to_friend_request
PASSED api/tests/test_friends.py::test_list_friends
PASSED api/tests/test_friends.py::test_remove_friend
========================================================= 23 passed in 7.61s ==========================================================
```

Defect counts are inconsistent, we are unable to compute a defect density. Currently however, our LOC is = 1,416,730. And Average Cyclomatic Complexity is: A (3.08)



```
backend/api/models.py
    C 19:0 Friend - A (2)
    M 54:4 Friend.__str__ - A (1)
    M 57:4 Friend.accept - A (1)
    M 61:4 Friend.reject - A (1)
backend/api/apps.py
    C 4:0 ApiConfig - A (1)
backend/api/migrations/0002_remove_friend_api_friend_sender__dfedbd_idx_and_more.py
    C 6:0 Migration - A (1)
backend/api/migrations/0003_remove_friend_unique_sender_receiver_and_more.py
    C 8:0 Migration - A (1)
backend/api/migrations/0001_initial.py
    C 10:0 Migration - A (1)
backend/api/tests/test_friends.py
    F 70:0 test_list_friends - B (9)
    F 21:0 test_send_friend_request - A (4)
    F 35:0 test_reciprocal_request_auto_accept - A (4)
    F 54:0 test_respond_to_friend_request - A (4)
    F 97:0 test_remove_friend - A (3)
backend/api/tests/rsvp_test.py
    M 29:4 TestRSVPAPI.test_create_rsvp_success - B (7)
    M 86:4 TestRSVPAPI.test_get_rsvp_by_plan_id_success - B (7)
    M 128:4 TestRSVPAPI.test_get_rsvp_by_user_id_success - B (7)
    M 308:4 TestRSVPAPI.test_rsvp_data_integrity - A (5)
    C 17:0 TestRSVPAPI - A (4)
    M 53:4 TestRSVPAPI.test_create_rsvp_missing_plan_id - A (3)
    M 116:4 TestRSVPAPI.test_get_rsvp_by_plan_id_not_found - A (3)
    M 158:4 TestRSVPAPI.test_get_rsvp_by_user_id_not_found - A (3)
    M 170:4 TestRSVPAPI.test_delete_rsvp_by_id_success - A (3)
    M 185:4 TestRSVPAPI.test_delete_rsvp_by_id_not_found - A (3)
    M 199:4 TestRSVPAPI.test_delete_rsvp_by_id_invalid_id - A (3)
    M 208:4 TestRSVPAPI.test_delete_rsvp_by_plan_id_success - A (3)
    M 222:4 TestRSVPAPI.test_delete_rsvp_by_plan_id_not_found - A (3)
    M 235:4 TestRSVPAPI.test_delete_rsvp_by_plan_id_invalid_id - A (3)
    M 257:4 TestRSVPAPI.test_multiple_rsvps_same_plan - A (3)
    M 284:4 TestRSVPAPI.test_duplicate_rsvp_prevention - A (3)
    M 65:4 TestRSVPAPI.test_create_rsvp_invalid_plan_id - A (2)
    M 243:4 TestRSVPAPI.test_rsvp_authentication_required - A (2)
    M 19:4 TestRSVPAPI.setup_method - A (1)
backend/api/tests/plans_test.py
    C 15:0 TestPlansAPI - A (5)
    M 16:4 TestPlansAPI.test_create_plan - A (4)
backend/api/management/commands/seed_dev_users.py
    C 5:0 Command - A (5)
    M 8:4 Command.handle - A (4)
backend/api/utils/mongo.py
    F 31:0 get_collection - B (6)
    F 14:0 _unwrap_db - A (3)
```



```
backend/api/models/friends_models.py
    C 20:0 Friend - A (2)
    M 59:4 Friend.__str__ - A (1)
    M 63:4 Friend.accept - A (1)
    M 67:4 Friend.reject - A (1)
backend/api/serializers/rsvp_serializer.py
    C 8:0 RSVPSerializer - A (1)
backend/api/serializers/friends_serializer.py
    C 20:0 FriendSerializer - A (2)
    M 30:4 FriendSerializer.create - A (1)
    M 33:4 FriendSerializer.update - A (1)
backend/api/serializers/fields.py
    C 17:0 ObjectIdField - A (3)
    M 18:4 ObjectIdField.to_representation - A (2)
    M 24:4 ObjectIdField.to_internal_value - A (2)
backend/api/serializers/auth_serializer.py
    C 13:0 UserRegistrationSerializer - A (3)
    M 21:4 UserRegistrationSerializer.validate_email - A (2)
    M 29:4 UserRegistrationSerializer.validate - A (2)
    M 34:4 UserRegistrationSerializer.create - A (1)
    C 40:0 UserSerializer - A (1)
backend/api/serializers/plans_serializer.py
    C 7:0 LocationSerializer - A (1)
    C 15:0 PlansSerializer - A (1)
backend/api/views/friends.py
    F 140:0 remove_friend - B (10)
    F 35:0 send_friend_request - B (6)
    F 87:0 list_friends - A (5)
    F 67:0 respond_to_friend_request - A (4)
backend/api/views/users.py
    F 72:0 list_users - A (5)
    F 107:0 get_user - A (5)
    F 33:0 register_user - A (2)
    F 58:0 get_user_profile - A (1)
backend/api/views/rsvp.py
    F 25:0 create_rsvp - B (6)
    F 68:0 get_rsvp_by_plan_id - A (4)
    F 87:0 get_rsvp_by_user_id - A (4)
    F 106:0 delete_rsvp_by_id - A (3)
    F 122:0 delete_rsvp_by_plan_id - A (3)
backend/api/views/plans.py
    F 28:0 create_plan - A (5)
    F 81:0 get_plans_by_id - A (3)
    F 99:0 update_plan - A (3)
    F 122:0 delete_plan - A (3)
    F 68:0 get_plans - A (2)
    F 20:0 get_plans_collection - A (1)

75 blocks (classes, functions, methods) analyzed.
Average complexity: A (3.08)
```

- References

- Glossary