M5: Testing and Code Review

1. Change History

Change Date	Modified Sections	Rationale
04/02/2025	2.1.1, 2.3, 2.4	Added interface and tests
04/02/2025	5	Codacy errors were fixed!
04/02/2025	4.1	Seperated front end tests

2. Back-end Test Specification: APIs

2.1. Locations of Back-end Tests and Instructions to Run Them

2.1.1. Tests

Interface	Describe Group Location, No Mocks	Describe Group Location, With Mocks
POST /user/	`tests/mockFree/userRoutes.mockFree.test.ts#L34`	`tests/mocked/userRoutes.mockFed.test.ts#L16`
GET /user/:householdId	`tests/mockFree/userRoutes.mockFree.test.ts#L60`	`tests/mocked/userRoutes.mockFed.test.ts#L85`
GET /user/specific-user/:email	`tests/mockFree/userRoutes.mockFree.test.ts#L81`	`tests/mocked/userRoutes.mockFed.test.ts#L127`
PATCH /user/update-household/:email	`tests/mockFree/userRoutes.mockFree.test.ts#L104`	`tests/mocked/userRoutes.mockFed.test.ts#L171`
PATCH /user/:email	`tests/mockFree/userRoutes.mockFree.test.ts#L129`	`tests/mocked/userRoutes.mockFed.test.ts#L219`
DELETE /user/:email	`tests/mockFree/userRoutes.mockFree.test.ts#L166`	`tests/mocked/userRoutes.mockFed.test.ts#L279`
POST /household/create	`tests/mockFree/houseRoutes.mockFree.test.ts#L38`	`tests/mocked/houseRoutes.mockFed.test.ts#L19`
POST /pet/	`tests/mockFree/petRoutes.mockFree.test.ts#L38`	`tests/mocked/petRoutes.mockFed.test.ts#L88`
GET /pet/:householdId	`tests/mockFree/petRoutes.mockFree.test.ts#L67`	`tests/mocked/petRoutes.mockFed.test.ts#L141`
PATCH /pet/:petName/feed	`tests/mockFree/petRoutes.mockFree.test.ts#L91`	`tests/mocked/petRoutes.mockFed.test.ts#L174`
DELETE /pet/:petName	`tests/mockFree/petRoutes.mockFree.test.ts#L165`	`tests/mocked/petRoutes.mockFed.test.ts#L269`
POST /log/:petName	`tests/mockFree/logRoutes.mockFree.test.ts#L39`	`tests/mocked/logRoutes.mockFed.test.ts#L24`
GET /log/pet/:petId	`tests/mockFree/logRoutes.mockFree.test.ts#L115`	`tests/mocked/logRoutes.mockFed.test.ts#L101`
GET /log/household/:householdId	`tests/mockFree/logRoutes.mockFree.test.ts#L171`	`tests/mocked/logRoutes.mockFed.test.ts#L142`
GET /log/:user/:userEmail	`tests/mockFree/logRoutes.mockFree.test.ts#L134`	`tests/mocked/logRoutes.mockFed.test.ts#L174`
POST /analytics/anomalies/:householdId	`tests/mockFree/analyticsRoutes.mockFree.test.ts#L34`	`tests/mocked/analyticsRoutes.mockFed.test.ts#L`
GET /analytics/rankings/:householdId	`tests/mockFree/analyticsRoutes.mockFree.test.ts#L131`	`tests/mocked/analyticsRoutes.mockFed.test.ts#L`
POST /notify/:email	`tests/mockFree/notificationsRoutes.mockFree.test.ts#L38`	`tests/mocked/notificationsRoutes.mockFed.test.`

2.1.2. Commit Hash Where Tests Run

`6854e9f52ca117583f545456127759eb804490f3`

2.1.3. Explanation on How to Run the Tests

1. **Clone the Repository**:
   - Open your terminal and run:

   ```
   git clone https://github.com/matthewfung04/the-animals-are-starving.git
   cd the-animals-are-starving/Backend
   ```

2. **Environment File for Running the Server**:
   - If you want to run the server locally, create a .env file with the following:

```
DB_URI=mongodb://localhost:27017
PORT=5000
```

   - If you wish to run the firebase notifications, you will need to create a key or get ours, but we recommend just using our server.

3. **Install the relevant packages**:
   - First install the packages needed with the command below. This may take a while (multiple minutes) as some packages are large.

   ```
   npm install
   ```

4. **Running the Test(s)**
   - To simply run all the tests, use:

   ```
   npm test
   ```

   - To run with coverage, use:

   ```
   npm run test:coverage
   ```

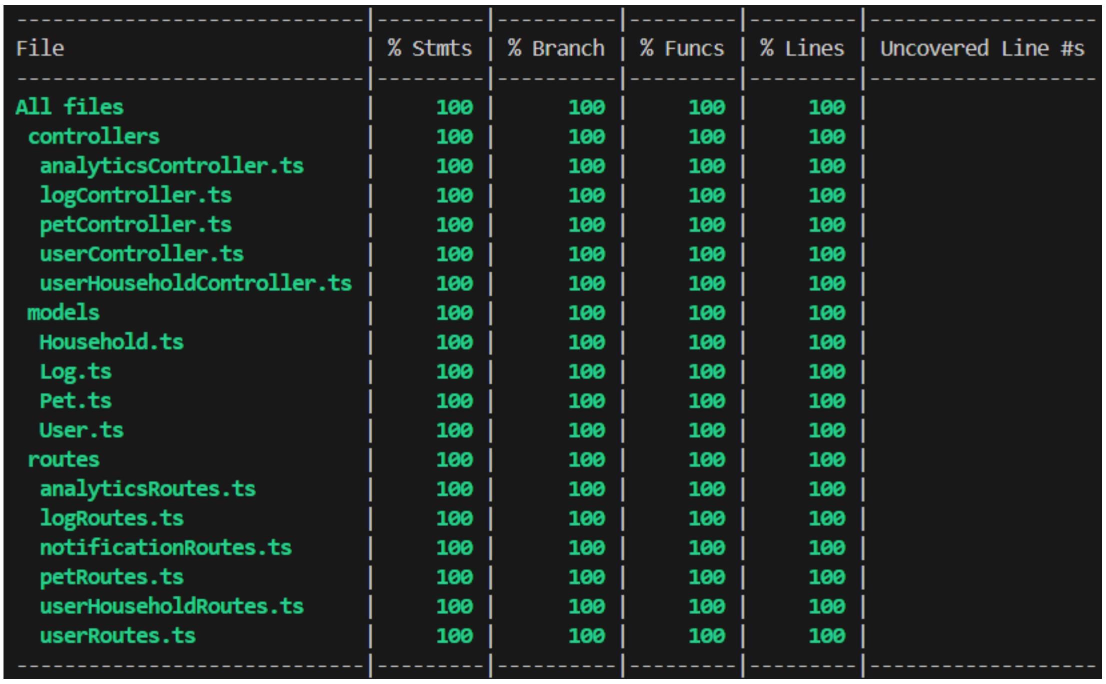
   - To run a specific folder (i.e., mocked or mockFree) or singular file. Just add the path to either command as follows:

   ```
   npm test [path]
   ```

2.2. GitHub Actions Configuration Location

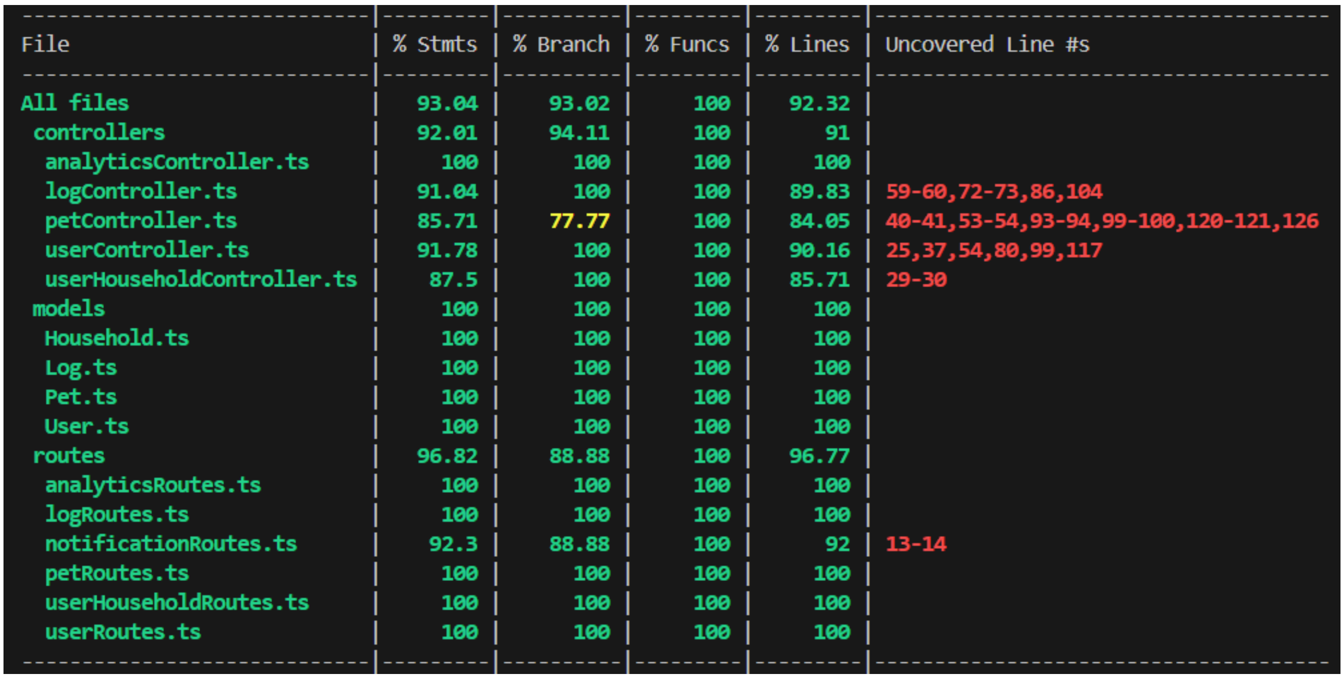
`~/.github/workflows/jest-tests.yml`

2.3. Jest Coverage Report Screenshots With Mocks



```
-----------------------------|---------|----------|---------|---------|-------------------
File                         | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-----------------------------|---------|----------|---------|---------|-------------------
All files                    |     100 |      100 |     100 |     100 |
 controllers                 |     100 |      100 |     100 |     100 |
  analyticsController.ts     |     100 |      100 |     100 |     100 |
  logController.ts           |     100 |      100 |     100 |     100 |
  petController.ts           |     100 |      100 |     100 |     100 |
  userController.ts          |     100 |      100 |     100 |     100 |
  userHouseholdController.ts |     100 |      100 |     100 |     100 |
 models                      |     100 |      100 |     100 |     100 |
  Household.ts               |     100 |      100 |     100 |     100 |
  Log.ts                     |     100 |      100 |     100 |     100 |
  Pet.ts                     |     100 |      100 |     100 |     100 |
  User.ts                    |     100 |      100 |     100 |     100 |
 routes                      |     100 |      100 |     100 |     100 |
  analyticsRoutes.ts         |     100 |      100 |     100 |     100 |
  logRoutes.ts               |     100 |      100 |     100 |     100 |
  notificationRoutes.ts      |     100 |      100 |     100 |     100 |
  petRoutes.ts               |     100 |      100 |     100 |     100 |
  userHouseholdRoutes.ts     |     100 |      100 |     100 |     100 |
  userRoutes.ts              |     100 |      100 |     100 |     100 |
-----------------------------|---------|----------|---------|---------|-------------------
```

2.4. Jest Coverage Report Screenshots Without Mocks



```
-----------------------------|---------|----------|---------|---------|------------------------------------
File                         | % Stmts | % Branch | % Funcs | % Lines | Uncovered Line #s
-----------------------------|---------|----------|---------|---------|------------------------------------
All files                    |   93.04 |    93.02 |     100 |   92.32 |
 controllers                 |   92.01 |    94.11 |     100 |      91 |
  analyticsController.ts     |     100 |      100 |     100 |     100 |
  logController.ts           |   91.04 |      100 |     100 |   89.83 | 59-60,72-73,86,104
  petController.ts           |   85.71 |    77.77 |     100 |   84.05 | 40-41,53-54,93-94,99-100,120-121,126
  userController.ts          |   91.78 |      100 |     100 |   90.16 | 25,37,54,80,99,117
  userHouseholdController.ts |    87.5 |      100 |     100 |   85.71 | 29-30
 models                      |     100 |      100 |     100 |     100 |
  Household.ts               |     100 |      100 |     100 |     100 |
  Log.ts                     |     100 |      100 |     100 |     100 |
  Pet.ts                     |     100 |      100 |     100 |     100 |
  User.ts                    |     100 |      100 |     100 |     100 |
 routes                      |   96.82 |    88.88 |     100 |   96.77 |
  analyticsRoutes.ts         |     100 |      100 |     100 |     100 |
  logRoutes.ts               |     100 |      100 |     100 |     100 |
  notificationRoutes.ts      |    92.3 |    88.88 |     100 |      92 | 13-14
  petRoutes.ts               |     100 |      100 |     100 |     100 |
  userHouseholdRoutes.ts     |     100 |      100 |     100 |     100 |
  userRoutes.ts              |     100 |      100 |     100 |     100 |
-----------------------------|---------|----------|---------|---------|------------------------------------
```

2.5 Missing Coverage Justification (Without Mocks)

Analytics Routes

- (21-22) An error in the route which we'd need a mock to test
- (74-75) A catch block from a database error, again requires a mock

Log Routes

- (54-55, 67-68, 81, 99) All catch blocks which require a mock to test

Notifications Routes

- (13-14) Catch block from a database error, requires a mock

UserHousehold Routes

- (29-30) A catch block from a database error, requires a mock

User Routes

- (25, 37, 54, 80, 99, 117) All catch blocks which require a mock to test

Pet Routes

- (40-41, 53-54, 99-100, 126) All catch blocks which require a mock to test
- (93-94, 120-121) Other type of db error where petId does not align with the one found from Pet.findOne, would need a mock to test.

---

3. Back-end Test Specification: Tests of Non-Functional Requirements

3.1. Test Locations in Git

Non-Functional Requirement	Location in Git
Performance (Response Time)	`tests/non-functional/performance.test.js`

3.2. Test Verification and Logs

- **Performance (Response Time)**
  - **Verification:** This test suite simulates multiple concurrent API calls using Jest along with a load-testing utility to mimic real-world user behavior. The focus is on key endpoints such as retrieving user and pet data as well as household creation to ensure that each call completes within the target response time of 2 seconds under the worst loads for an app of our size (100 concurrent calls). The test logs capture metrics such as average response time, maximum response time, and error rates. These logs are then analyzed to identify any performance bottlenecks, ensuring the system can handle expected traffic without degradation in user experience.
  - **Log Output**

```
PASS tests/non-functional/performance.test.ts (6.24 s)
Performance Non-Functional Tests
√ should respond quickly for concurrent calls to user endpoints (417 ms)
√ should respond quickly for concurrent calls to pet endpoints (233 ms)
√ should respond quickly for concurrent calls to household endpoints (344 ms)
Test Suites: 1 passed, 1 total
Tests:       3 passed, 3 total
Snapshots:   0 total
Time:        6.358 s, estimated 7 s
```

- Note: our second non-functional requirement was tested on the front-end

---

4. Front-end Test Specification

4.1. Location in Git of Front-end Test Suite:

`src/app/src/androidTest/java/com/example/theanimalsarestarving/HistoryManagementTest.java`
`src/app/src/androidTest/java/com/example/theanimalsarestarving/LogFeedingTest.java`
`src/app/src/androidTest/java/com/example/theanimalsarestarving/NotificationsTest.java`
`src/app/src/androidTest/java/com/example/theanimalsarestarving/ThreeClickTest.java`

How To Run Tests

- Make sure the backend is running with `npm run dev` in the `backend` folder
- Enter a `mongosh` shell in the `backend` folder
- Type `use pet-tracker` and then `load("initdb.mongo")`
- Run the class for the desired test from Android Studios while an emulator is active

4.2. Tests

- **Use Case: Log Feeding (Test Success)**
  - **Expected Behaviors:**

Scenario Steps	Test Case Steps
1. User scrolls through the list of pets on the base page to find the pet being fed.	Check that the feed button is present and click it. Locate the desired pet on the page and ensure it exists.
2. User presses the corresponding “Feed Pet” button to confirm that the pet has been fed.	Locate the feed button with accociated pet and press it.
3. System updates the feeding log with the pet's ID, user ID, date, and amount of food.	Exit feeding menu. Checks log button exists. Clicks log button. Checks that new log appeared.
4. User is prompted with a success message indicating that the log has been updated successfully.	While on feed screen, check clicked pet for "FED" text.

  No failure scenarios as they all relate to server issues which requires mocking.

- **Use Case: Requesting Others to do Feeding (Test Success)**
  - **Expected Behaviors:**

Scenario Steps	Test Case Steps
1. Sender begins at the home page and presses "Notify Other Users".	Launch MainActivity and click notiy_button
2. The sender then presses the “Notify” button next to the name of the recipient.	First check that the a row with "Bob" appears with a "Notify" button.
3. App sends a request to the server to notify the recipient.	Verify that the button is pressable by clicking "Nofity"
4. User 2 (the recipient) receives a notification indicating that they are responsible for feeding the pet.	This cannot be tested explicitly as we only have one device

  No failure scenarios as they all relate to server issues which requires mocking.

- **Use Case: History Management (Test Success)**
  - **Expected Behaviors:**

Scenario Steps	Test Case Steps

Scenario Steps	Test Case Steps
1. Household Manager clicks the “View History” button on the main screen	Check that the log button is present and click it.
2. The history is retrieved and the user is directed to a new screen displaying the feeding history in the household	Check to see if existing logs in the backend appear on screen.

- **Use Case: History Management (Test Failure)**
  - **Expected Behaviors:**

Scenario Steps	Test Case Steps
1. Household Manager clicks the “View History” button on the main screen - 1a. User is not a Household Manager - 1a1. User is not prompted with the "View History" button	Set current user to a non manager user. Check that the view log button does not exist. Set current user back to manager user.

- **Test Logs:**

```
com.example.theanimalsarestarving.EspressoTest   26.92 s
passed testLogFeedingUseCase   10.59 s
passed testNotifications    3.92 s
passed testHistoryManagementUseCaseSuccess   2.85 s
passed testHistoryManagementUseCaseFailure   1.24 s
```

4.3. Front-end Non-Functional Test

Non-Functional Requirement	Location in Git
Accessability (# of Clicks)	src/app/src/androidTest/java/com/example/theanimalsarestarving/EspressoTest.java

4.3.1. Test Verification and Logs

- **Accessability (# of Clicks)**
  - **Verification:** This test suite simulates a how a real-world user would access the use-case with the longest sequence of clicks. The focus is on making sure that the app should be usable by all members of the household, including those with impaired vision, language barriers, and of all mental faculties. Thus, users should be able to complete the action in 3 clicks or less. We did this by running tests starting from the home screen, and ensuring that we can feed a pet and view history by using less than three perform(click()) calls.
  - **Log Output**

```
com.example.theanimalsarestarving.EspressoTest   26.92 s
passed threeClickTest   8.32 s
```

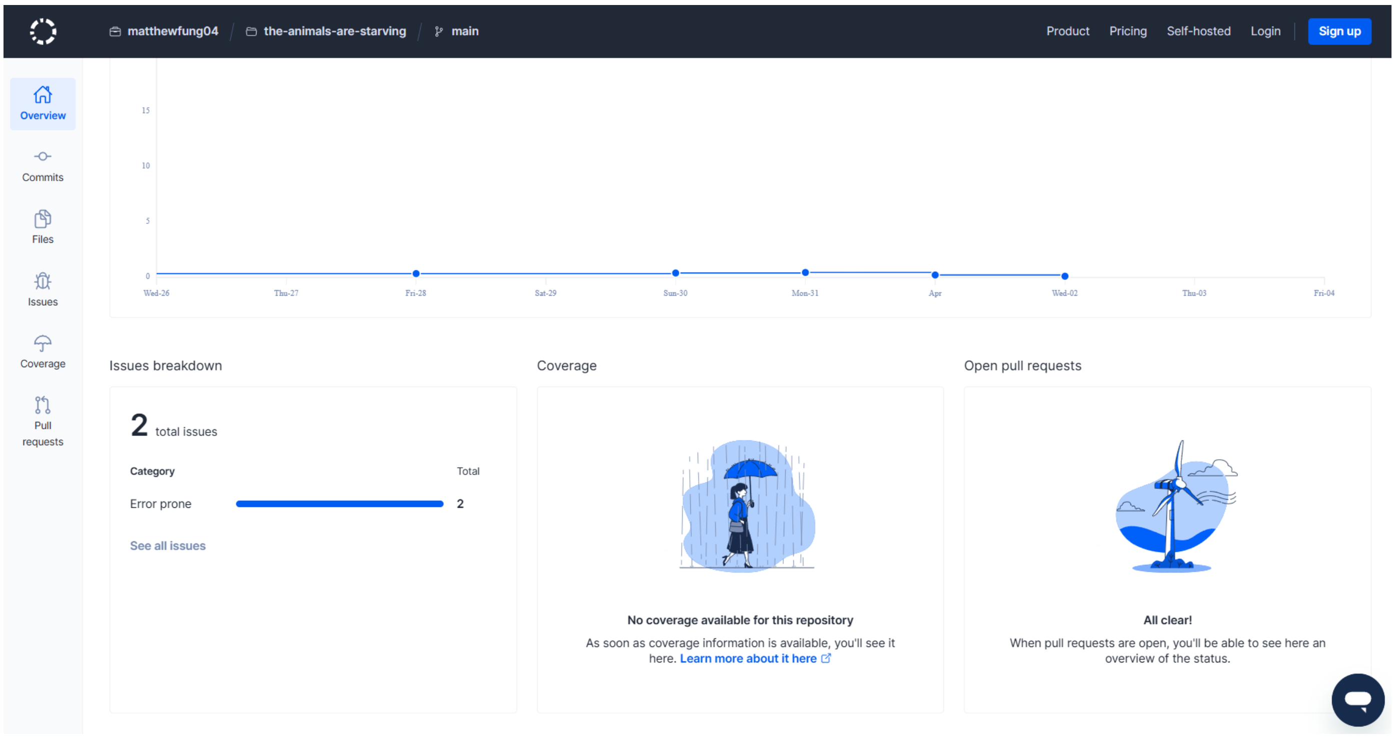
---

5. Automated Code Review Results

5.1. Commit Hash Where Codacy Ran

`6854e9f52ca117583f545456127759eb804490f3`

5.2. Unfixed Issues per Codacy Category



Issues breakdown

2 total issues

Category	Total
Error prone	2

See all issues

Coverage

No coverage available for this repository

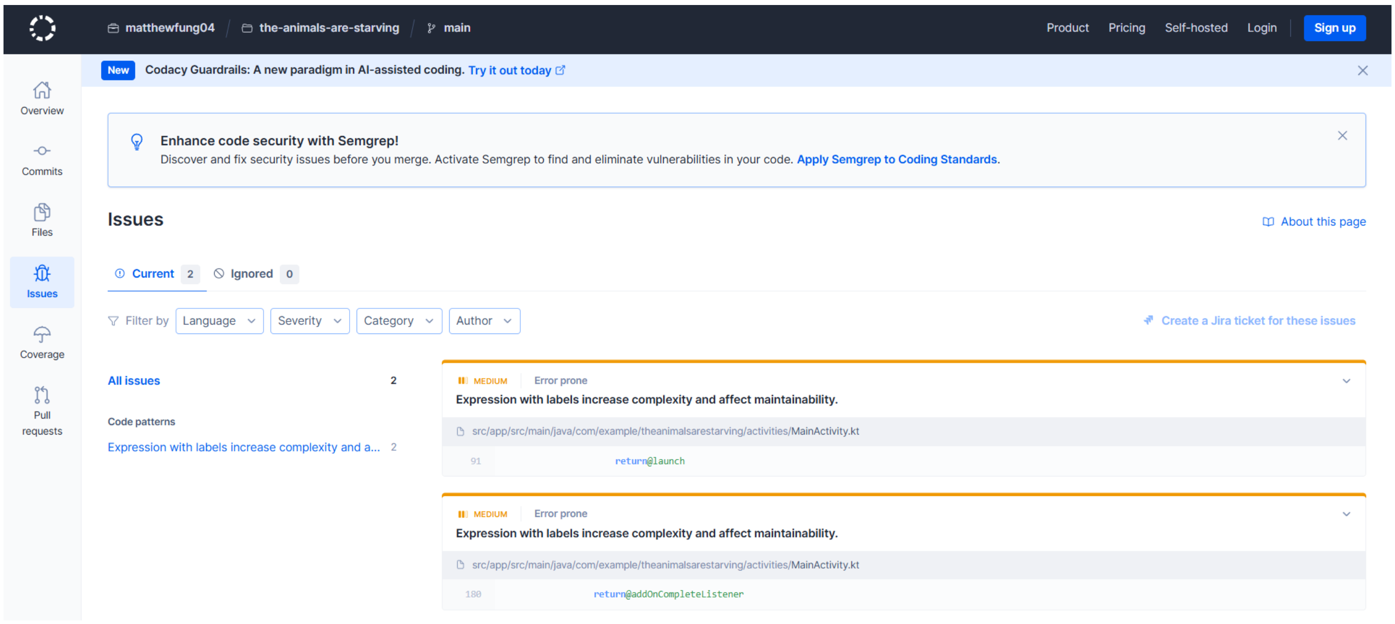
As soon as coverage information is available, you'll see it here. Learn more about it here

Open pull requests

All clear!

When pull requests are open, you'll be able to see here an overview of the status.

5.3. Unfixed Issues per Codacy Code Pattern



New Codacy Guardrails: A new paradigm in AI-assisted coding. Try it out today

Enhance code security with Semgrep!
Discover and fix security issues before you merge. Activate Semgrep to find and eliminate vulnerabilities in your code. Apply Semgrep to Coding Standards.

Issues

All issues 2

Code patterns

Expression with labels increase complexity and a... 2

MEDIUM Error prone
Expression with labels increase complexity and affect maintainability.
src/app/src/main/java/com/example/theanimalsarestarving/activities/MainActivity.kt

```
91                  return@launch
```

MEDIUM Error prone
Expression with labels increase complexity and affect maintainability.
src/app/src/main/java/com/example/theanimalsarestarving/activities/MainActivity.kt

```
180               return@addOnCompleteListener
```

5.4. Justifications for Unfixed Issues

- **Code Pattern: Expression with labels increase complexity and affect maintainability.**
    1. **Expression with labels increase complexity and affect maintainability.**
        - **Location in Git:** `src/app/src/main/java/com/example/theanimalsarestarving/activities/MainActivity.kt#L91`
        - **Location in Git:** `src/app/src/main/java/com/example/theanimalsarestarving/activities/MainActivity.kt#L180`
        - **Justification:** Labels can reduce code redundancy by preventing duplicated logic. Kotlin provides labels as an intentional feature to handle nested control flow; for instance, to simulate a break from a higher-level loop or to return from a lambda, you can label the loop or lambda and use a return at that label. The official Kotlin docs even demonstrate that there's "no direct equivalent for break, but it can be simulated by adding another nesting lambda and non-locally returning from it". (Kotlin Docs) In our case, these breaks were necessary, as they are part of onCreate, and reordering is not possible during setup.