# M5 BiteSwipe Review

## 1. Manual Code Review

a. Code is well documented, good variable names, efficient, no bad design smells, correct error handling: 9/10

- Explanation of the point: There are many API calls to be made in the front end that is TODO. The settings activity is empty and doesn't have any plan on it. What does tvLoggedinuser mean? Why not just loggedinuser? Name is good can make it better.
- Authentication: Good variable names (signInButton, device, ...), catches and logs authentication faliures, and uses hashed nounce. Improvement would be adding documentation and removing the hardcoded WEB_CLIENT_ID.
- Users: All API endpoints are well-documented, with clean implementations. Overal, the deployment guide for setting up frontend, backend and firebase is clear. The code directories are well-structured. And variable names are easy to understand, with error handling logic.

## 2. Manual Test Review

a. Tests are complete (all APIs exposed to the frontend are tested, three main use cases are tested), errors and edge cases are thoroughly tested, correct assertions are used: 7/10 Frontend:

- Explanation of the point: The front end has some redundant tests. The Join group and create group test looks like they are testing the same thing. The front end testing use cases includes login, create/join, and swipe. It may be tested but the testing document is just the same as the original starting document.
- UI tests exist, but no network failure handling.

Backend:

- Authentication: Tests cover base cases for sigin in, could use tests for user creation, authentication faliures, and invalid sign-in responses.
- Users: All user API endpoints are tested, with success and failing cases.

b. Tests implementation matches the requirements and design: 9/10

- Explanation of the point: The swipe is only tested for left and right however it's not tested with restaurants unavailability as mentioned in your requirement doc
- Most tests implementation matches the requirements and design in the API.md file.

c. Tests are well-structured: 10/10

- Explanation of the point:
  - Authentication: Tests are well strucutred and uses proper syntax.
  - User: Tests are named with their testing api name and indicated whether they are mocked or unmocked tests. They are also organized into directories.
  - some tests are redundant.

d. Tests are comprehensive (good coverage): 9/10

- Explanation of the point: Core backend functionality are tested. However, some backend files (e.g. session controllers) are left uncovered and unjustified. Also, since we cannot obtain the results from unmocked tests, it's difficult to determine their tests' validity.
- Authentication: Tests cover the login and nav to homepage, missing faliure scenario test cases.

e. Non-functional requirements are tested well: 5/10

- Explanation of the point: Didn't test two of the non-functional requirements - the usability and response time.

f. All backend tests can be run automatically: 8/10

- Explanation of the point: Mocked tests are set up correctly on Github actions.

## 3. Automated Code Review

a. Codacy runs with the required setup: 10/10

- Explanation of the point: The codacy runs in the codacy dashboard since there are issues shown in the files.

b. All remaining Codacy issues are well-justified: 3/10

- Explanation of the point: There are no codacy issues that are justified.

## 4. Fault: Report One Major Implementation Issue

The report should contain the details about the issue (with screenshots) and the severity of the issue

- Major issue:

- Users' firebase cloud message tokens are easily obtainable using GET - your users are exposed to potential spam attacks like receiving unauthorized notifications.

- Other issues:

  - They say radius is required in the backend in the SessionRoutes.ts (Line 66). However, in the frontend they don't check that if it is empty in CreateGroupPage.kt (Line 118).

  - Moderategrouppage.kt (Line 137):

    - API call is missing an error handler.
    - There is potential of crashing the app

  - User experience: Use more icons rather than text. Make the text font larger, and buttons larger.