Long time ago in a galaxy far away...

...there was an API endpoint to get list of users with permissions assigned to them.

Permissions are assigned to users through `api_users` wrapper: every user has single api_user record which holds references to multiple permissions assigned to that user.

Then a decision was made to group permissions into `user_role`.
Now `api_user` record holds references both to permissions assigned individually and to permissions assigned trough groups of permissions i. e. via `user_role`.

Changes into given listing endpoint were made in order to return not only directly assigned permissions but also a permissions assigned via user_role. And requirement is to keep distinction between those two kinds of permissions. So permissions that come through user_role should have a flag `role_based: true` and directly assigned permissions should have this flag as `false`.

Such changes were made by a developer. Changes were successfully tested in their development environment with a small set of test data. Changes were then deployed.

And then…

Users report erratic behavior for department `58884ff19c5d396319ef9c09` – role_based and non role_based permission are all mixed up and some times permissions are duplicated (and users, naturally, do not give any other details).

Now you have sufficient set of data. You need to find where data returned from endpoint is wrong and to fix bugs that causing incorrect behavior.

Instructions:

1. Find a Python Flask application in `test_02` folder. Configure your development environment to be able to run a project.
2. Restore data set provided in a mongodump archive `test_mongo_dump_archive` into your MongoDB.
3. Use GIT and commit changes you are making. Make sure GIT commit history is neat and commits represent actual work you've done: have an initial commit that contains original assignment code and then commit your code changes.
4. There is a pytest test under `test_result_correct.py`. You can run it from application folder calling:

```
$ python -m pytest
```

when test passes successfully – consider this challenge is done.
Test relies on `fixture_data` and is sensitive to API return value. So do not alter fixture data or endpoint interface.

When challenge is complete:

Email your GIT