Iteration 2 will expand on the work in [Iteration 1](https://canvas.wisc.edu/courses/193673/assignments/723589), adding features, tests, and fixing bugs according (flexibly) to your planned development schedule.

This assignment also requires teams to **use code coverage tools** to estimate the quality of their tests. Code coverage tools determine which lines of your code are being executed, and will have been covered in lecture before this project is due. You should measure code coverage, use it to identify at least one important new test to implement, and then measure code coverage again and **demonstrate an increase**. It is up to you to determine what code coverage tool is appropriate for your development environment and how to use it. Use the [class discussion forum](https://canvas.wisc.edu/courses/193673/discussion_topics) to reach out to classmates and course staff if you need help installing and running coverage tools.

## Key things to do

1. Connecting (server) backend and frontend
2. Navigation bar
3. JUnit tests

* <https://developer.android.com/training/testing/unit-testing/local-unit-tests>
* <https://developer.android.com/studio/test>

1. Code coverage

* <https://medium.com/android-testing-daily/code-coverage-dc1e3d0c70>
* <https://android.jlelse.eu/get-beautiful-coverage-reports-in-your-android-projects-ce9ba281507f>
* <https://stackoverflow.com/questions/18683022/how-to-get-code-coverage-using-android-studio>

1. Progress report

## ToDo: Priority

1. **Create a branch for each person. (Do not push to the master before tested by more than two people)**
2. Set server and database + JUnit test on server

* Write instruction of server on README

1. Navigation bar
2. Write backend structure (**main, category, report, goal, Login(create user)**) + including JUnit test method structure

* Backend structure includes…
  + functions
  + Objects
  + Note for what each function do
  + Ready to write backend codes

**Server is ready: everyone has full understanding on server & how to test with sample dataset** → at least by 4/7 & **meeting** so everyone can see how server works

1. Write backend codes (main, category, report, goal, Login(create user)) + fill in JUnit tests while writing backend codes (record what test cases you used)

* Keep testing while developing codes

1. Test whole project → at least by 4/12 & meeting if needed
2. Progress report

|  |  |
| --- | --- |
| **4/5** | Ali: Login & Categories backend structure  Na: keep testing database queries & creating sample dataset  Joon: Main & Goal backend structure  Andy: Finish connector class and make sure it works with the server & README for client & server  Anna: Navigation bar & Report backend structure + Calculator  Jack: README for client & server |
| **4/6** |
| **4/7** |
| **4/8** |  |
| **4/9** |  |
| **4/10** |  |
| **4/11** |  |
| **4/12** | Meeting (final check in) |
| **4/13 DUE** | Working on Report |

## Iteration 2: Cross-out if ~~done~~

Iteration 2 will focus on finalizing setting up queries and request handlers for the server as well as adding handlers for buttons and linking the front-end pages to the back-end data. User functionality will come together here, as by the end of the iteration, they will be fully linked to begin testing.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Task | Priority (1-5) | Difficulty (1-5) | Time Units (1-5) | Dependency | Person Responsible |
| Implement Back End for “Create User”, “Login”, “Report”, “Goals”, and “Category” pages so ensure that they link with the front-end properly and store the correct data. | 3 | 4 | 2 | Implementing Front End for Pages | Andy/Na/Anna |
| Add Handling for invalid usernames and passwords and if profile already exists as well as changing passwords | 1 | 1 | 1 | Implementing Back end of the login page | Andy/Na |
| Add a Handler for a “Create Account” Button | 1 | 1 | 1 | Will need Create Account Button to be developed | Joon/Ali |
| Implement Back end for calculating current expense, current balance, getting categories, adding transaction, getting daily graph. | 1 | 2 | 2 | None | Andy/Na |
| Add Back end Logic for editing a users profile, sending notifications, user guide, getting application file | 1 | 2 | 4 | None | Andy/Na |
| Implement the functionality of the connecting class between the front-end and the back-end so that they properly interact with the rest of the code | 1 | 4 | 3 | Will need back-end data along with front-end data handlers to ensure this is done properly | Jack/Na |
| Add handlers to link the front-end and back-end for category and goals page | 1 | 1 | 1 | Both goals and categories front-end and back-end must exist | Joon/Ali/Anna |
| Implement handler to add and edit the categories | 1 | 1 | 1 | Edit categories button must already exist | Joon/Ali |
| Implement table using the info from the database to be able to display the categories and the user’s income | 1 | 2 | 2 | Categories page must already exist and data must be already stored in the back-end | Andy/Na/Anna |
| Add post account creation information filling, and notification for user to go to user guide | 1 | 2 | 2 | Account must already be created and stored in the back-end along with user data | Ali/Joon |
| Integrate database with logging in backend logic | 3 | 3 | 2 | Database must be defined | Andy/Na |
| Implement Database management code for deleting older data for both MySQL and the server | 3 | 2 | 2 | Data must already be stored in the database to do this. | Jack/Anna |