Name: Hyunjae Kim

Term: Summer 2022

Previous Team Projects

- What types of projects have your worked on?

  In the past, I worked on course projects. To be specific, I worked on CS291 (Web development) making Tic-Tac-Toe website. Moreover, I worked on ECE 271/2 (Digital Logic Design) project making FPGA random hexadecimal dice.

- What was your role on those projects?

  In Web Development course project, my role was making Tic-Tac-Toe game by using HTML and JavaScript.

  In Digital Logic Design, my role was making timer which initialize the random hexadecimal numbers, and writing part of project report.

- What are some the difficulties that arose and what do you feel was the cause?

  The difficulties in Web Development project were that optimizing my Tic-Tac-Toe game with other parts of the website including general HTML design, and Tic-Tac-Toe game scoring system. I felt that the cause of these difficulties was not frequently collaborating with other team member's code, and not considering the component of the code which will going to be connected.

  The difficulties in Digital Logic Design project were similar to the Web Development project. The difficulties were optimizing my counting clock with combining random number generator, and random number indicator. Similar to the reason of the Web Development project, I think the causes were not considering the entire functionality of the FPGA random dice, and not frequently reviewing and collaborating with other team members.

Working with Continuous Integration

- Did you have any initial reservations about using a Continuous Integration workflow? If so, what were they and how did you plan on overcoming them?

  I had an initial reservation about using a continuous integration workflow. I thought I am doing the continuous integration workflow first time so that I am not good at it and take long time to complete. However, when I start the workflow, it was the thing I experienced from past course project and assignments. After realizing these, I overcome the initial reservation I had.

- What was your experience with the mandatory Code Review process? What techniques did you employ to make your Code Reviews more effective? As a Code Reviewee, how did you approach pull requests?

  In code review process, I checked the code is easy to read, has consistent naming conventions, and it is enough pythonic.

  The techniques I employed to make my code was checking the "environment.yml" on GitHub. It automatically checks the Linting Errors so that I only need to review the naming convention, readability, and etc.

  As a code reviewee, I approached my pull requests having good readability, consistent naming, and pythonic style. The reason is that by using "environment.yml", I can check my linting errors in the code by myself so that I was able to request to review the style, naming and if my code is enough pythonic.

- How important did you find daily commits when working with Continuous Integration? If you had trouble with this, how did you set out to remedy the situation?

  I found that daily commits are significantly important when working with continuous integration. I did the assignments about two days, and this made me realize I need to do daily commits. By cramming the commits on GitHub, I was confused about my workflow. However, by finding each commits step by step, it was able to remedy my confusion on the workflow. Therefore, these are the reason why I think daily commits are important, and the method I remedy my trouble.

- Did you and your team follow a Test Driven Development Process? If so, what was your motivations and how successful were you?

  I followed the TDD process after I made the function converting decimals into hexadecimals in Function3. The reason is that in order to do TDD process on Function3, the fundamental feature had to be established, and that fundamental feature was converting decimals into hexadecimals.

The motivation TDD process was that I was not sure to make fully activating ‘conv_endian’ function without checking any edge cases. By using TDD process, it was easy to find the edge case of the function, and revise the function to fit that edge case. As a result, the function passed all the test case which was able to generate expected output from the all of the testcases.

- How do you feel you grew as a developer?

  I feel rewarding that I grew as a developer after completing the group project. From the group project, I learned a lot what is important for a developer. It is consistent, and soft skills.

- How do you feel you grew as a team member?

  I feel also rewarding and fruitful that I grew as a team member after completing the group project. From this group project, I learned that soft skills including communication is important in team work, and work as a developer.

- How do you feel you grew as a mentor?

  I am satisfied the I grew as a mentor. I thought growing as a mentor needs some hard work. However, after completing this project, I realized everyone can be a mentor if they have mind to help other team members or colleagues.

Lessons for the Future

- How does working with Continuous Integration facilitate better software development?

  Continuous Integration is building concrete pillar of software. To be specific, continuous integration starts from building small part, and then solve the small problem from the built small part. By repeating this process, the software become dense, and the software is able to activate as the expectation. Therefore, these are the reasons why continuous integration facilitates better software.

- How does having mandatory Code Reviews help individuals become better developers?

  Having mandatory Code Reviews help individuals to learn things where they missed on the process of writing codes, and new methods or concrete their knowledge by reviewing the code. These are the reasons why I think mandatory Code Reviews help individuals become better developers.

- If you used Test Driven Development, how does this lead to better software?

  Similar to continuous integration, TDD also starts with small part of the software, and solve the small problem from built small part, and so on. Moreover, by using certain criteria, it is efficient to build small parts and integrate them. Therefore, TDD leads to better software in effective and efficient way.

- Why is it so important to have a solid test suite when working on a shared code repository?

  If there is no solid test suite, then some edge cases can be checked multiple times, and other edge cases never been checked. Therefore, in order to prevent this problem from happening, it is so important to have a solid test suite when working on a shared code repository.