

OSS Term Project

Part 2 Report

23102003 Kim Doyi
23102009 Park Shinhyung
23102015 Oh Eunyoung
23102025 Lee Haneol
23102031 Hwang Yuyoung

Documentation of Process

1 Pre-Development Stage (11/26 - 12/1)

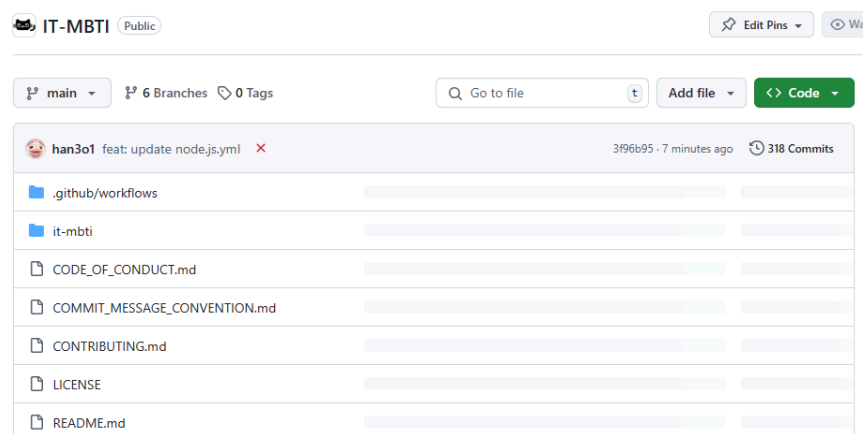
: We worked on Web UI design and creating JSON data.

- **Web UI design (Yuyoung, Shinhyung):** Created UI for the start page, question page, and result page.
- **Create JSON data (Eunyoung, Doyi, Haneol):** Created question and result data.

2 Development Stage (12/1 - 12/17)

1. Initial commits

- **Create documentation files:** Create basic documentation files to set up the structure of the project.



- **README.md:** Basic information including project description, features, installation instructions, etc.
- **CODE_OF_CONDUCT.md:** Code of conduct for project participants.
- **COMMIT_MESSAGE_CONVENTION.md:** Guidelines for writing commit messages.
- **CONTRIBUTING.md:** Contribution guidelines.
- **LICENSE:** Set up the license file.
- **Initial Commit and Core Structure Setup:**
 - > To establish the core structure of the project, the initial commit focused on setting up the basic files and directory structure.
 - > Through this initial commit, we defined the 4 main pages, the components required for each page, and the data files, which helped plan the overall structure and flow of the project.

1) Main Page Structure

The project is composed of 4 main pages, and each page is wrapped with a header and footer. These pages are as follows:

- StartPage: The page that guides the user to start the test.
- QuestionPage: The page where the user can view questions and provide answers.
- LoadingPage: The page that briefly displays a loading state before transitioning to the result page.
- ResultPage: The page that shows the results based on the user's answers.

2) Data Files Setup

- questions.json: This file contains the question data used in the QuestionPage, including each question and its options. It is used to dynamically load the questions and handle user input.
- results.json: This file contains the result data used in the ResultPage, providing the results based on the user's answers. It is used to calculate and display the results.

3) Component Structure

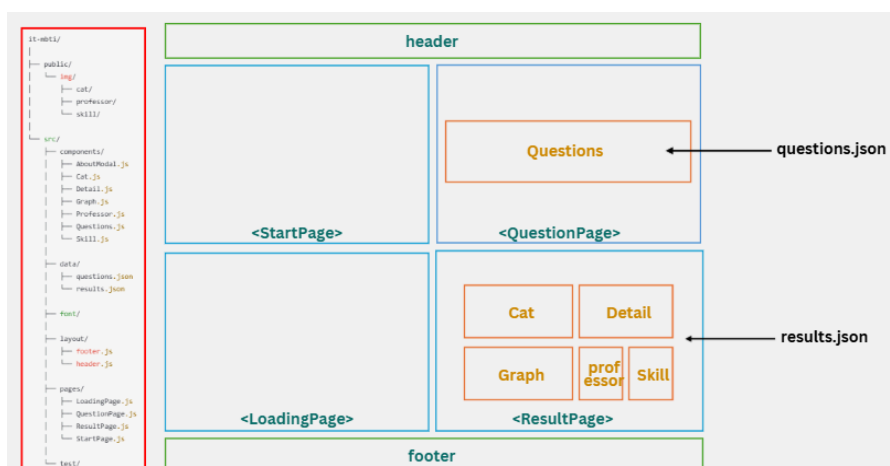
The project's components are set up to independently handle the UI elements required by each page. The main components are as follows:

- StartPage, QuestionPage, ResultPage: These components define the core UI for each page.
- Cat, Graph, Skill: These are various result components used in the ResultPage to display the cat type, job recommendations, graphs, etc.

4) Core Structure Setup

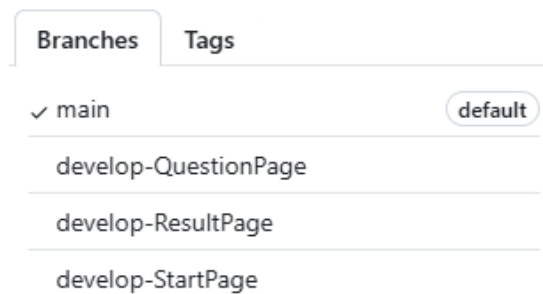
In the initial commit, the core structure of the project was set up to clearly define the relationships between the pages and how each page and component would retrieve the necessary data. This structure ensures that each feature is independently managed, and data can be easily updated when necessary.

-> Thus, through the initial commit, we set up the project's basic structure and core files, specifying how each page and component would interact. This foundation allows us to proceed with feature implementation and additional work.



2. Detailed Implementation: Implement features, content, design, etc.

- 1) **Repository Forking:** Fork the original repository to allow team members to work on their respective tasks.
- 2) **Detailed Implementation:** Each team member specifies and implements their assigned tasks.
- 3) **Branch-specific Pull requests:** After completing their work, each team member creates a Pull request for the respective branch and requests a review.
 - develop-StartPage: Work related to the StartPage.
 - develop-QuestionPage: Work related to the QuestionPage.
 - develop-ResultPage: Work related to the ResultPage.



- 4) **Merging into the main branch:** After reviewing the PR, if there are no issues, merge each task into the main branch.

3. Project Execution

- 1) **Implement essential features and run:** After implementing the necessary features, run the project to verify its actual functionality.
- 2) **Share issues and improvement suggestions:** Share any problems or areas needing improvement that arise during the execution phase and discuss possible solutions with team members.
- 3) **Pull requests:** If additional modifications are needed based on the issues or improvements found during execution, reflect those changes through a PR.

```
oh030@... MINGW64 ~/IT-MBTI/it-mbti (main)
$ git pull

oh030@... MINGW64 ~/IT-MBTI/it-mbti (main)
$ npm start

> it-mbti@0.1.0 start
> react-scripts start
```

4. Code Review and Collaboration:

- PR Process:

- 1) Each team member creates a PR for the committed work and explains the changes.
- 2) Other team members review the PR and provide feedback on the code or changes.
- 3) After incorporating the feedback, if there are no further issues, the PR is merged into the respective branch.

feat: create babel.config.js
#97 by han3o1 was merged 13 minutes ago

feat: add missing Babel plugin for compatibility
#96 by han3o1 was merged 1 hour ago

bug: update Questions.js
#95 by han3o1 was merged 4 hours ago

design: develop each page
#94 by han3o1 was merged 5 hours ago

Develop start page
#93 by eun5young was merged 6 hours ago

bug: update Questions.js #95

Merged yuyoung924 merged 2 commits into `OSS-TeamProjectt:main` from `han3o1:main` 4 hours ago

Conversation 3 -> Commits 2 Checks 0 Files changed 2

han3o1 commented 5 hours ago Member ...

I finally solved our error of percentage. πππ
Percentage values in the graph now correctly reflect the scores relative to their maximum possible values.
Please check it!

han3o1 added 2 commits 5 hours ago

- feat: update ResultPage.js** Verified 1f9343c
- bug: update Questions.js** Verified ddf5b1e

kimm00 commented 5 hours ago Member ...

Very good!!!! Finally we see the end. Thank you very much! I love you Haneol 🍷🍷

eun5young commented 5 hours ago Member ...

Wow crazy Haneol! I love you~❤️ I'll run the revised version right away!! Thank you!!!

yuyoung924 commented 4 hours ago Member ...

Good job Hanul solved our last problem!!! I Will try unit test

yuyoung924 merged commit `3ec8285` into `OSS-TeamProjectt:main` 4 hours ago Revert

- Issues Process:

- 1) Issues are created for problems that arise during the execution phase or areas that need further clarification. Labels are assigned to specify the type of issue.
- 2) Team members discuss the issues and propose solutions.
- 3) Once the issue is resolved, the issue is closed to indicate the completion of the task.

The order of the questions #68

Closed eun5young opened this issue yesterday · 3 comments



eun5young commented yesterday

Member ...

It would be better to randomize the order of the questions during the test!
Currently, the questions are grouped in sets of three based on the result order, but shuffling the order could make the test more engaging and interesting for users!



eun5young added the **question** label yesterday



han3o1 commented yesterday

Member ...

That's a great suggestion! Randomizing the order of the questions would definitely make the test more dynamic and engaging for users. However, if we implement shuffling, we need to ensure that the logic for calculating the results remains accurate and consistent. Careful adjustments will be necessary to maintain the integrity of the results while improving the user experience. 😊



kimm00 commented yesterday

Member ...

I agree that changing the order of questions can increase user engagement and make the testing experience more dynamic! I think we need to implement this to ensure that the logic for calculating results remains intact and consistent. Any additional considerations? 😊



eun5young commented 6 hours ago

Member Author ...

I updated the order of the questions to make them more varied! I also ensured that the results are logically connected. Thank you for your help~ Please check the updated changes! 😊



eun5young closed this as completed 6 hours ago

Graph Component: a problem of percentage is not displayed as 0

Closed shinh09 opened this issue 8 hours ago · 3 comments



shinh09 commented 8 hours ago

Member ...

The percentage should be marked 0 when there is no score in the Graph component, but the function is failing in the unit test.

```
Test Suites: 2 failed, 3 passed, 5 total
Tests: 1 failed, 13 passed, 14 total
Snapshots: 0 total
Time: 1.708 s
Ran all test suites.
```

• Graph Component › If there is no score, make sure the percentage is displayed as 0



shinh09 added the **bug** label 8 hours ago



han3o1 commented 8 hours ago · edited

Member ...

I've tried several modifications but none of them worked well. ㅠㅠ
Here is my recent try.

```
function Graph({ scores }) {
  const maxScorePerType = 30;
  const graphData = scores.map(([{type, score}] => {
    const percentage = Math.round((score / maxScorePerType) * 100);
    return {
      label: type,
      percentage: percentage,
      color: fixedColors[type],
    };
  }));
};
```



I think we need to modify it outside of the Graph.js file.

3 Examination & Documentation (12/13 - 12/16)

- **Unit Test** (Yuyoung): Write unit tests for key features to ensure that all functions work as expected.
- **CI Pipeline** (Haneol): Set up GitHub Actions to automatically run tests whenever code changes are made.
- **PPT (Presentation)** (Eunyoung, Shinhyung): Prepare a PowerPoint presentation summarizing the project overview, key features, development process, test results, and contributions.
- **Documentation (Report)** (Doyi, Eunyoung): Write a report documenting the implementation process, community interactions, challenges and solutions, and key learnings.

Community interactions


Filters ▾		Q is:issue is:open			Labels 12	Milestones 0	New issue		
<input type="checkbox"/>	🕒 10 Open	✓ 2 Closed		Author ▾	Label ▾	Projects ▾	Milestones ▾	Assignee ▾	Sort ▾
<input type="checkbox"/>	🕒	Graph Component: a problem of percentage is not displayed as 0	bug		#74 opened 5 hours ago by shinh09				3
<input type="checkbox"/>	🕒	Percentage Calculation Bug on Result Page	bug		#69 opened yesterday by kimm00				3
<input type="checkbox"/>	🕒	Update Result Page Features and Fix Errors	bug		#67 opened yesterday by han3o1	3 tasks done			2
<input type="checkbox"/>	🕒	Idea Suggestion for [Brief Feature Description]	good first issue idea		#60 opened yesterday by yuyoung924				
<input type="checkbox"/>	🕒	Error in the results provided by the result page	bug		#59 opened 2 days ago by eun5young				1
<input type="checkbox"/>	🕒	Diverse Field Data Ideas	good first issue idea		#28 opened 5 days ago by shinh09				2
<input type="checkbox"/>	🕒	Type Results Return Questions	question		#27 opened 5 days ago by shinh09				7
<input type="checkbox"/>	🕒	Develop list - StartPage	to do		#13 opened last week by shinh09	3 tasks done			
<input type="checkbox"/>	🕒	To do list - Data	to do		#11 opened 2 weeks ago by kimm00	10 tasks done			6
<input type="checkbox"/>	🕒	To do list - design	to do		#10 opened 2 weeks ago by shinh09	13 tasks done			2

1 Type Results Return Questions]

: Description of the most active issue topic

Type Results Return Questions #27

 Open **shinh09** opened this issue 5 days ago · 7 comments





shinh09 commented 5 days ago

Member


...

Results page related questions! The way you calculate the results is simply the sum of your answers. By the way, if the sum results are the same, what type of results should you give? $\pi\pi$





shinh09 added the **question** label 5 days ago



kimm00 commented 4 days ago


Member


...

Why don't we use additional data for the same case? For example, add weight to a specific question or consider an additional priority question.

Like in UI/UX Designer, we can weight this question 'Enjoys solving problems with creative ideas.'

-> It's because the core of UI/UX design is creative problem solving and user-centered design.







han3o1 commented 2 days ago Member ...

Great point to consider!
Instead of solely relying on weighted questions to resolve ties, how about giving users a choice in the process?

For instance, if multiple results have the same score, you could present an additional question like, "Which activity do you prefer more?" to help further differentiate the results. This approach could make the experience more interactive for the user and increase their trust in the final result.


By involving users in the decision-making process, it shifts from being purely calculation-based to feeling more personalized and engaging. Great work so far! 🙌






yuyoung924 commented 2 days ago Member ...


If the score is tied, I think we'll have to think about what results to return. Let's put all those results on the results page so that users can check them. What do you think?






eun5young commented 2 days ago Member ...

I think this is a good idea!! As Yuyoung said, it would be a good idea to show the users all the results with the most points! Why don't we make it so that the users can turn the page and see the next results?







shinh09 commented 2 days ago Member Author ...

If the score is tied, I think we'll have to think about what results to return. Let's put all those results on the results page so that users can check them. What do you think?


I think this way too.. I think the user will like to show all the results that they might be interested in.






han3o1 commented 2 days ago Member ...


I completely agree! Displaying all tied results on the results page would be a great way to ensure users can see all their top matches. This approach feels fair and user-friendly. Then, let's move forward with this idea!





kimm00 commented 2 days ago Member ...

I agree that showing all tied results on the results page, along with the ability to navigate through them, would make the experience more engaging and user-friendly. Let's proceed with implementing this idea, and we can refine it further if needed during the process. Great teamwork, everyone!





: As a result of the exchange of various opinions to Shinhung's question, Yuyoung's opinion that all the results should be displayed in the window was accepted when there were several tests tied.

It is not limited to just one job, but provides users with an opportunity to consider various jobs.

[2] Process of generating Pull Request and Merging]


feat: add Detail.js file #16

 Merged yuyoung924 merged 3 commits into [OSS-TeamProjectt:develop-ResultPage](#) from [yuyoung924:main](#)  last week

 Conversation 4

 Commits 3

 Checks 0

 Files changed 2




yuyoung924 commented last week

Member ...

No description provided.



 han3o1 and others added 3 commits [last week](#)

-   add: Create questions.json Verified 37896a7
-   Merge pull request [OSS-TeamProjectt#15](#) from han3o1/main ... Verified bed5c2a
-   feat: add detail.js 05ce4a4

 yuyoung924 merged commit **4e9cf1b** into [OSS-TeamProjectt:develop-ResultPage](#) last week

Revert



shinh09 commented last week • edited ▾

Member ...

Don't you have a styling file..?



yuyoung924 commented last week

Member Author ...

Oh,, sorry... I changed it right away. Thank you!



  yuyoung924 changed the title ~~Add Detail.js file~~ feat: add Detail.js file last week



han3o1 commented last week

Member ...

Can you change it to English and post it?



yuyoung924 commented last week

Member Author ...

Okay! I'll update that ASAP! Thank you



: It shows the real-time detection of the other person's mistakes, feedback, and immediate acceptance. This allowed smooth work to continue without much difficulty.

It is very important to check the other person's code and comment before doing a merge.

[3] Issue the errors and check them in real time]

Update Result Page Features and Fix Errors #67

🔒 Closed

🔄 3 tasks done

han3o1 opened this issue yesterday · 2 comments



han3o1 commented yesterday · edited by kimm00 ▾

Member ...

I will share the current situation of our project!

Modifications:

- ✅ Modify the result page to allow users to view all tied results when multiple scores are the same.
- ✅ Connect professor images to their corresponding external links.
- ✅ Fix the percentage calculation errors in the result display.

Feel free to share any additional thoughts or suggestions on how to improve the result page further. Your feedback is highly valued and will help us make the user experience even better! 😊



🏷️ kimm00 added the **bug** label yesterday



kimm00 commented yesterday

Member ...

Thank you for sharing the updates on the result page! I would like to work on fixing the percentage calculation errors in the result display as it seems crucial for providing accurate results to users.

I'll review the related code and ensure that the calculations handle all tied results correctly. Once I make the changes, I'll submit a PR for further review. Let me know if you have any additional feedback or considerations for this task! 😊



han3o1 commented yesterday

Member Author ...

That's a good point! I'll reflect on that.



: By representing the problem as a to-do list, users could easily grasp the problem. And by commenting on which users are trying to solve which problem, the problem duplication could not occur.

If the problem is solved, it's good to easily let us know that it's solved with just one click (check the box)!

Unit Test

1. Components Unit Test

1) AboutModal.test.js

1. Verify that modals are rendered normally
2. Verify the close button is rendered and the text is correct
3. Verify that the onClose function is invoked when the Close button is clicked

2) Cat.test.js

1. Verify that components are rendered normally
2. Verify that the correct data is displayed according to the Title prop
3. Verify that the default text is displayed when the type is not in results.json
4. Verify that the image is rendered correctly

3) Detail.test.js

1. Verify that components are rendered normally
2. Verify that the data that fits the type prop is displayed correctly
3. Verify that the default value is displayed when type is not in results.json

4) Graph.test.js

1. Verify that components are rendered normally
2. Verify the percentage is displayed correctly based on the score
3. If there is no score, make sure the percentage is displayed as 0%
4. Verify that a component is empty when it is in an empty score array

5) Professor.test.js

1. Verify that components are rendered normally
2. Verify that the correct faculty for the type prop is displayed correctly
3. Verify that the link to the professor profile and the image are rendered correctly
4. Verify that the default text is displayed when type is not in results.json

6) Question.test.js

1. Verify that the question text is rendered correctly
2. Verify that all option buttons are rendered
3. Check to see if the selection status changes when the button is clicked

7) Skill.test.js

1. Verify that components are rendered normally
2. Verify that the skill that fits the type prop is displayed correctly
3. Verify skill images and links are rendered correctly
4. Verify that the default text is displayed when type is not in results.json

2. layout Unit Test

1) footer.test.js

1. Verify that the logo image is rendered
2. Verify that the university name and project title are rendered

2) header.test.js

1. Verify that navigation links are rendered
2. Verify that About button is rendered and opens modal on click
3. Verify that modal closes when the Close button is clicked
4. Verify that navigation links are rendered
5. Verify that About button is rendered and opens modal on click
6. Verify that modal closes when the Close button is clicked

3. Page Unit Test

1) LoadingPage.test.js

1. Verify that loading text and spinner are rendered
2. Verify that it redirects to /result after 5 seconds

2) QuestionPage.test.js

1. Verify that the first question is display
2. Verify that progress bar updates when answering a question
3. Verify that navigating to /loading when last question is answered

3) ResultPage.test.js

1. Verify that components are rendered correctly with the highest score
2. Verify that Next button changes the result type

4) StartPage.test.js

1. Verify that the titles and description are rendered correctly
2. Verify that the image is rendered correctly
3. Verify that clicking the Start button navigates to /question

Challenges and Solutions

We've got some issues during the project.

1 Collision in the Process of Generating Pull Request

Problem:

Merge failed due to conflict after updating the code and generating a pull request. After reviewing the codebase, we found that fragments of existing and rewritten codes overlap, resulting in redundancy and complexity.

Solution:

We were able to work the affected parts again to resolve conflicts, simplify implementations, and ensure smooth and successful merging. This approach not only improves code quality but can also contribute to long-term maintenance possibilities. This problem provided an opportunity to simplify the code for better readability and efficiency.

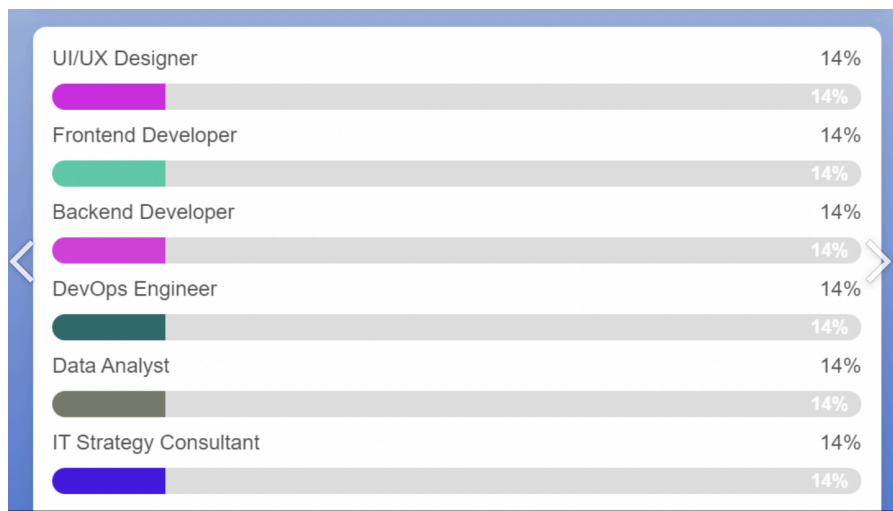
💡 2 Percentage Calculation Bug on Result Page 💡

Problem:

The current code calculates the percent of each type based on the sum of the scores and the percent for each type, but the calculation process is not scaled properly and the correct percentage value is not reflected. This leads to an error that actually outputs **60%**, even though the total percentage should be **50%** when a neutral answer is chosen.

The root cause of this problem is that the answer score is fixed to the **integer value (1, 2, 3, 4, 5)** only. The neutral value (3) is excessively influential in the overall percentage calculation, as the score of the neutral answer is not reflected at the same rate as the other scores. This results in the percentage of the score not matching the actual total.

In addition, the percent calculation process identified a problem where the ratio was incorrectly reflected due to poor normalization. As a result, the ratio of each type of score to the absolute value distorted the relative ratio to the total score, which leads to incorrect results for the user. (e.g. The result was significantly lower than 100% even though everyone chose 'Strongly Agree'.)



-> This is an example of an incorrect result that only 14% of the time comes out, not up to 100%, even though all of the 'Strongly Agree' are pressed.

Solution:

We introduced score scaling, using new values of 0, 2.5, 5, 7.5 and 10. Since the neutral response was scaled to a value of 5, we set it so that the normalized sum can be accurately reflected in the ratio calculation.

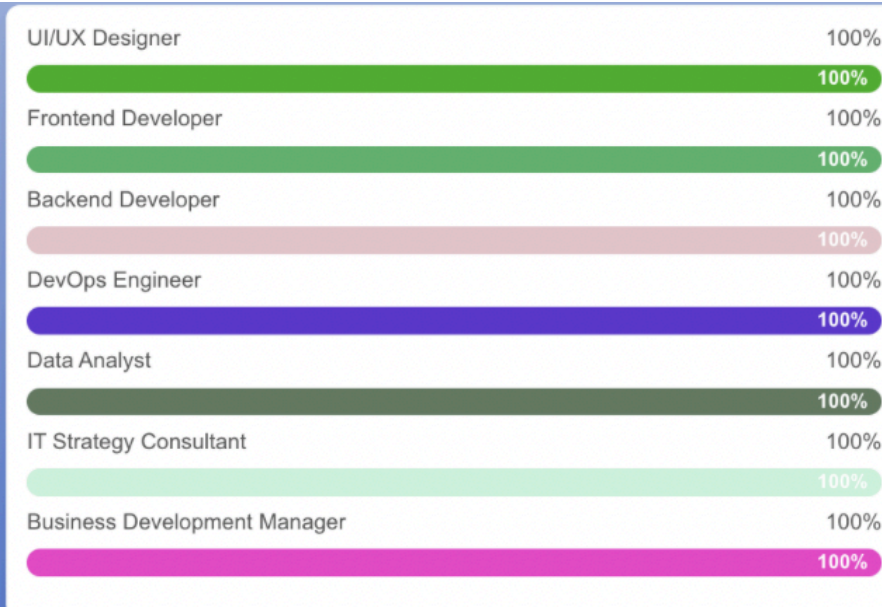
For example, if only neutral (5) is selected for all questions, the total percentage is correctly calculated as **50%**.

[Modify Question.js]

```
it-mbti/src/components/Questions.js +34 -19

47 47      return (
48 48          <QuestionContainer>
49 49              <QuestionText>{questionText}</QuestionText>
50 50              <OptionsContainer>
51 -          <OptionButton $isSelected={selectedOption === 1} onClick={() => handleClick(0)}>
52 -              Strongly Disagree
53 -          </OptionButton>
54 -          <OptionButton $isSelected={selectedOption === 2} onClick={() => handleClick(2.5)}>
55 -              Disagree
56 -          </OptionButton>
57 -          <OptionButton $isSelected={selectedOption === 3} onClick={() => handleClick(5)}>
58 -              Neutral
59 -          </OptionButton>
60 -          <OptionButton $isSelected={selectedOption === 4} onClick={() => handleClick(7.5)}>
61 -              Agree
62 -          </OptionButton>
63 -          <OptionButton $isSelected={selectedOption === 5} onClick={() => handleClick(10)}>
64 -              Strongly Agree
65 -          </OptionButton>
```

```
51 + <IconButton
52 +   $isSelected={selectedOption === 0}
53 +   onClick={() => handleClick(0)}
54 + >
55 +   Strongly Disagree
56 + </IconButton>
57 + <IconButton
58 +   $isSelected={selectedOption === 2.5}
59 +   onClick={() => handleClick(2.5)}
60 + >
61 +   Disagree
62 + </IconButton>
63 + <IconButton
64 +   $isSelected={selectedOption === 5}
65 +   onClick={() => handleClick(5)}
66 + >
67 +   Neutral
68 + </IconButton>
69 + <IconButton
70 +   $isSelected={selectedOption === 7.5}
71 +   onClick={() => handleClick(7.5)}
72 + >
73 +   Agree
74 + </IconButton>
75 + <IconButton
76 +   $isSelected={selectedOption === 10}
77 +   onClick={() => handleClick(10)}
78 + >
79 +   Strongly Agree
80 + </IconButton>
66 81 </OptionsContainer>
67 82 </QuestionContainer>
68 83 );
.....
↓
```



-> This is what the result page looks like when we modified the code and ran it again and it came out right!

3 Unit Test

Problem:

At first, we created a unit test at the same time as we constructed the project. But in this case, there was a problem with the unit test that was successful before other team members committed. Correcting issues or errors caused the unit test to fail to execute or change the data.

Solution:

When this challenge occurred, we stopped making unit test and completed all project configurations first. Then we completed the unit test, and succeeded in the unit test for a total of 13 files. After constructing the project without errors, we constructed the unit test file that allows you to test the project without errors.

After correcting the code file like this, we have configured the CI pipeline to prevent the unit test from failing.

Once the CI pipeline modifies and commits to a file, it automatically builds and runs the unit test. The unit test must be successful to proceed to the next stage, which notifies code modifiers if the already configured test fails. This process improves unit test and code maintenance.

```
PASS src/test/components/Detail.test.js
PASS src/test/components/Cat.test.js
PASS src/test/components/Graph.test.js
PASS src/test/layout/footer.test.js
PASS src/test/pages/LoadingPage.test.js (5.436 s)
```

```
PASS src/test/components/Professor.test.js
PASS src/test/components/Question.test.js
PASS src/test/layout/header.test.js
```

```
PASS src/test/components/AboutModal.test.js
PASS src/test/pages/StartPage.test.js
```

```
PASS src/test/pages/QuestionPage.test.js
```

```
PASS src/test/components/Skills.test.js
PASS src/test/pages/ResultPage.test.js
```

```
Test Suites: 13 passed, 13 total
Tests:       41 passed, 41 total
Snapshots:   0 total
Time:        5.686 s, estimated 6 s
Ran all test suites.
```

-> This illustrates the success of the unit test!

Key learnings

Key Learning through Collaboration

: Through collaboration, we learned a lot from solving various challenges and problems while working on team projects. In particular, we came to have a deep understanding of the importance of the key role of communication and the importance of open source, not the purpose of development.

1. Importance of Communication

Effective communication was one of the most crucial components of team projects. No matter how skillful the team is, poor exchanges of opinions between the team members can delay the project's progress or reduce the quality of the code.

From the beginning of the project, we were able to deal with the task quickly through clear division of roles, and we helped the people in the same part to improve our understanding of the task.

In addition, when code conflicts or unexpected errors occurred, we communicated quickly with our team members to find solutions. In particular, documentation that records and shares the causes and solutions of problems through GitHub Issue or Pull Request was very helpful.

Through a review of the code written by another team member, we exchanged feedback to find improvements. We were able to improve the quality of the code by cooperating with the solution rather than simply pointing out the problems.

In conclusion, we learned that the success of collaboration depends on clear and open communication rather than mere ability to write code.

2. Importance of Open Source

Through this project, we realized that learning the value and utilization of open sources is important.

Open source can quickly discover and solve problems through the participation of various developers (team members). Through code review, we have improved the quality of code by aiming for continuous improvement rather than rewriting code. We were able to come up with an efficient solution by openly discussing the issues we experienced through Github.

Through this cooperative culture, the problem-solving process itself became an opportunity for learning and development.

Due to the nature of the open source, all codes and work records are transparently disclosed, so it was possible to proceed efficiently based on trust in the collaboration process.

In addition, while working with several people, we naturally learned collaboration tools (a GitHub) and processes, which greatly helped to strengthen teamwork and project management capabilities.

Summary

: Through this team project, we were able to learn in depth the essence of efficient communication and the true value of open source. Beyond simple technological growth, we realized that smooth communication based on teamwork and active utilization and contribution of open source are key drivers for maximizing the project's performance.

This insight will provide a solid foundation for more complex and larger projects in the future, and will be an important foundation for building a sustainable development environment.

We learned that understanding the overall flow of the project, rather than focusing only on individual files, is essential for identifying and solving issues effectively. Revising only the files where problems are found can lead to incomplete solutions or new issues. By reviewing all related files and ensuring the overall code is cohesive and well-connected, we were able to enhance the project's completeness. This process highlighted the importance of active communication between contributors in open source projects.