Sprint Review

Date: Dec 12, 2022 3:40 PM to 4:40 PM
Place: Discord Call

Roles:

- Tanush Nadimpalli: Product Owner **[present]**
- Rohan Shiva: Scrum Master **[present]**
- Ramki Pitchala: Developer **[present]**

Sprint Goal Met 🥳

- All our tasks from the sprint backlog were completed.

Updated Product Backlog

[Epic 1] As an instructor, I want to create “fill in the blank” parson’s problems from code files, so that I can test my student’s ability to complete the missing ideas in code. [Problem 1] **[MET]**

[Epic 2] As an instructor, I want to generate multiple-choice style parson’s problems from code files, so that I can test my student's ability to discern incorrect parts of code blocks. [Problem 1]

[Epic 3] As an instructor, I want to generate 1 problem per problem type for each student so that each student will have an independent set of different types of problems to holistically test their understanding of the course content. [Problem 2] **[MET]**

[Epic 4] As an instructor, I want to be able to export my generated problem sets to Canvas, so that I can efficiently create Canvas Quizzes for my students by simply importing the exported problem sets. [Problem 3] **[PARTIALLY MET - Added functionality to export to JSON file]**

[Epic 5] **[NEW]** As an instructor, I want to be able to export my generated problem sets to Blackboard, so that I can efficiently create Blackboard Quizzes for my students by simply importing the exported problem sets.

[Epic 6] **[NEW]** As an instructor, I want to be able to export my generated problem sets to Moodle, so that I can efficiently create Moodle Quizzes for my students by simply importing the exported problem sets.

[Epic 7] **[NEW]** As an instructor, I want to be able to access previously generated problem sets, so that I can re-export/modify generated problem sets.

[EPIC 8] **[NEW]** As an instructor, I want to be able to generate matching style parson’s problems from a code file, so that I can test my student with more interactive questions.

Updated Sprint Backlog

[Task 1] Create a `FillInTheBlankProblemGenerator` class

- Assignee: **Ramki**
- Status: DONE
- User Story: **[Epic 1]**

[Task 2] Write unit tests for `FillIntheBlankProblemGenerator`

- Assignee: **Ramki**
- Status: DONE
- User Story: **[Epic 1]**

[Task 3] Modify `ProblemSetGenerator` to use `FillInTheBlankProblemGenerator` to generate “fill in the blank” problems

- Assignee: **Ramki**
- Status: DONE
- User Story: **[Epic 1]**

[Task 4] Modify `GenerationForm` to allow instructors to select “fill in the blank” problem type

- Assignee: **Tanush**
- Status: DONE
- User Story:**[Epic 1]**

[Task 5] Modify `GenerationForm` tests to ensure “fill in the blank” problem type can be selected in GenerationForm

- Assignee: **Tanush**
- Status: DONE
- User Story:**[Epic 1]**

[Task 6] : Modify GenerationForm to allow the # of problems per problem type to be set by the user

- Assignee: **Rohan**
- Status: DONE
- User Story: **[Epic 3]**

[Task 7] Modify `GenerationForm` tests to ensure # of problems per problem type implementation is respected

- Assignee: **Rohan**
- Status: DONE
- User Story: **[Epic 3]**

[Task 8] Modularize `GenerationResult` component and display the “fill in the blank” problem type. Update tests to make sure they reflect the new changes.

- Assignee: **Tanush**
- Status: DONE
- User Story: **[Epic 3]**

[Task 9] Add functionality in `GenerationResult` to export a generated problem set as a JSON file.

- Assignee: **Rohan**
- Status: DONE

- User Story: **[Epic 4]**

[Task 10] Modify `GenerationResult` tests to ensure export button to export generated problem set as a JSON file functions properly.

- Assignee: Rohan
- Status: DONE
- User Story: **[Epic 4]**