MSD Reading Questions self-assessment bot – Assignment 2

Kristina Alyabeva, Saveliy Chertkov, Kirill Korikov

October 2024

1 Customer interview

Questions	**Answers**
What is the model for? It will be used only by students for self-examination or by professor for help to assessment?	Students and tutors can use the bot to check the assignment.
Is there a need for a professor's role in program?	Creating roles is unnecessary, a single functional is enough.
Are the questions and articles static or will they change?	Evaluate the text in terms of clarity of presentation.
Will the bot leave comments in the pdf or send a text during the feedback?	It is not so important how the feedback will be presented. You need to specify the disclaimer "You should not completely trust the evaluation of the bot".
Where will the bot be hosted?	It is unlikely that there will be access to the university's servers. It is better to use Docker, the user will deploy everything locally.
What are the requirements of the criterion "too short" "too much"? (number of rows of lines)	There are no exact criteria, it depends. Even two lines are fine.
Is it necessary to take into account the full design of the document (the design of the issues, the design of the document header, the formatting of the document) and how will this be evaluated?	The bot should check the number of questions, the coherence of the text, the citation and formatting check.

Questions	**Answers**
Can you specify the metrics by which you will evaluate how successfully the product will be completed?	1. Usefulness to the student. It would be good to ask the student if the product helped to find mistakes in the work and improve it. It can be difficult to organize. You can think of a proxy dimension. For example, we can take some set of already evaluated works and see what the system writes. and ask them how much the answers match their assessment and how much they find it useful for the student
What is the main idea of the project?	It should work as an anti-plagiarism tool, as a means of final verification by the student before sending it to moodle to catch potential errors.

Tasks from the customer:

1. Divide tasks by importance
2. Interview classmates and tutors

2 Tech stack

The technical stack and description of tools are listed in the table.

Tool	**Description**
Python 3.12	The main language of bot development
python-telegram-bot	Module for interaction with Telegram
NLTK	Basic word processing and text handling
requests	Working with HTTP requests
pyGAM	Package for building Generalized Additive Models
unittest	Unit testing framework
Docker	Project deployment
GitLab	Version control system

The list of team members with tools proficiency is presented in the table below

Team member	**Tools**
Saveliy	Python 3.12 (8), python-telegram-bot (7), NLTK (5), request (8), PyGAM (6), unittest (5), Docker (6), GitLab (6)
Kirill	Python 3.12 (7), GitLab (4), python-telegram-bot (6), unittest (4), request (6)
Kristina	Python 3.12 (7), GitLab (5), NLTK (4)

3 Initial code base

Creation of README.md and Initial code base are presented in the following link.

4 Data model and UI design

4.1 Data Modeling

In agreement with the customer, there will be no database in our project.

The UML class diagram is shown in Figure 1.

The Document class is responsible for the student's work presented in PDF format.

The Report class is responsible for presenting the work report to the student.

The Comment class is responsible for presenting a comment to a part of the document text.

The PDF-reporter class is responsible for checking the document for GPT generation, formatting, and semantic content.

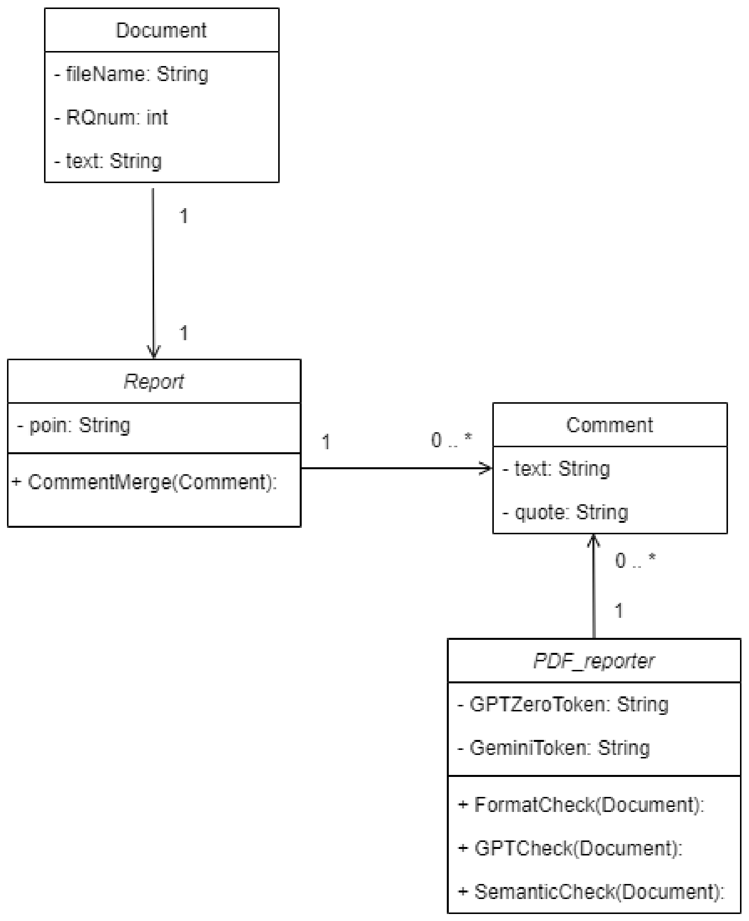


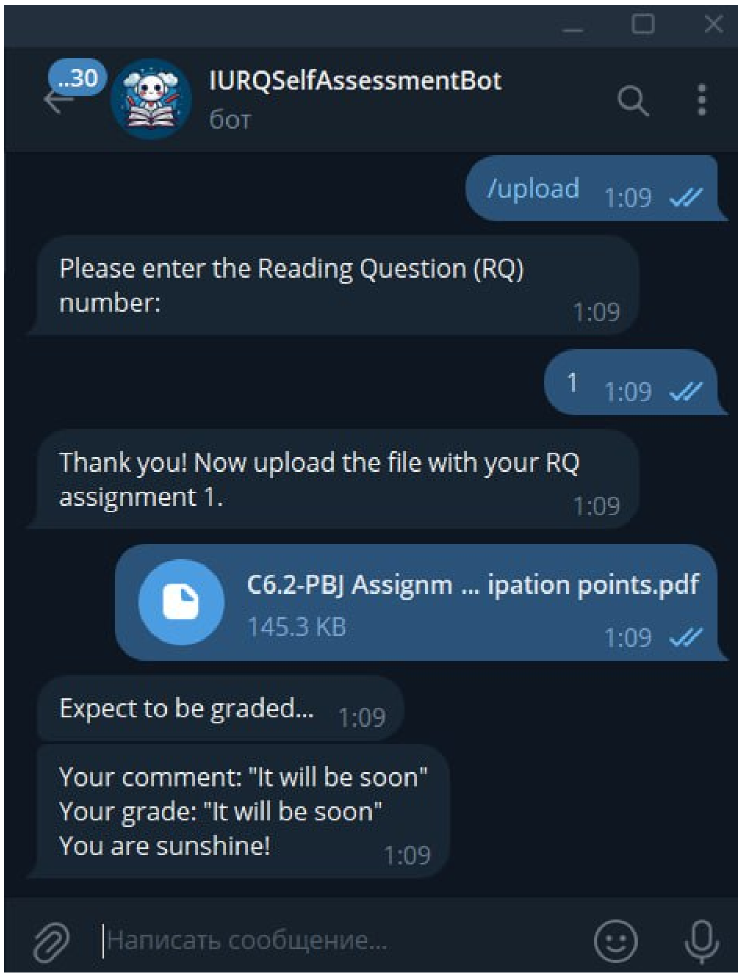
Figure 1: UML Class diagram

4.2 UI Prototyping

In Figure 2, you can see the user interface of the Bot.

The user enters the file download command, enters the RQ number and downloads the file.

The bot checks the file and sends a report to the user.



/upload 1:09

Please enter the Reading Question (RQ) number: 1:09

1 1:09

Thank you! Now upload the file with your RQ assignment 1. 1:09

C6.2-PBJ Assignm ... ipation points.pdf
145.3 KB 1:09

Expect to be graded... 1:09

Your comment: "It will be soon"
Your grade: "It will be soon"
You are sunshine! 1:09

Figure 2: Bot UI Design

4.3 Command API

Our bot's commands are listed below:

- /start
  Starts the bot
- /help
  Information on how to use the bot and what it can do
- /upload
  Provides access to RQ number selection and document download

5 Strategic plan

5.1 Success measures

We decided to formulate a threshold of success for our project using the GQM format. the goal then is formulated as following:

Analyze	the RQ Self-assessment bot
For the purpose of	measuring its value
With respect to	proximity of the bot's assessment to the TA/Professor's
From the viewpoint of	SE student
In the context of	MSD course

With the following questions and metrics being used to measure current state of the project relative to the goal:

Question	**Metrics**
How efficient bot is at finding mistakes?	**Number** of mistakes not found by the bot but found by the TA/Prof
Are bot's comments comparable to the human's common sense?	**Evaluation** by the students on the level from 1 to 5, where 1 — bot's commentary does not make any sense, and 5 — bot's commentary is totally valid
How close bot's comments to the TA/professor ones?	**Evaluation** of bot's comments by the TA/professors, whether they agree with the provided comments or not; **Number** of mistakes not found by the bot but found by the TA/Prof

5.2 Roadmap

Project will be managed using **iterative** model. At the end of each iteration there will be process of collecting feedback from the students and TAs/Professors whenever possible

List of planned steps formulated as following:

1. **Iteration 1**
   (a) Creation of a bot in Telegram
   (b) Model trained to inspect answers for adherence to the following criteria: length, coherence
   (c) Bot can return comments with the inspection result
   (d) Feedback collection
2. **Iteration 2**

   (a) Bot can parse the provided text to the "question-answer" pairs, given the text is appropriate (i.e. common person can define what is the question and what is the answer in this text)
   (b) Model trained to inspect the relevance of the answer to the question and any missed references
   (c) Feedback collection

3. **Iteration 3**
   (a) Bot can accept the PDF files of the RQs
   (b) Model trained to inspect the page limit, title, references page
   (c) Feedback collection

4. **Iteration 4**
   (a) Ability for any student to download the source code of the service and host it on their machine
   (b) **MVP milestone:** all previous steps are completed, students can use the bot by themselves to assess their RQs.
   (c) Feedback collection

5.3 Monitoring

We plan to give the MSD course students (as our potential customers) the access to our product after every iteration. This way we can collect the feedback and comment on how to improve our product Whenever possible, we also plan to ask the TA/Professors, to assess, how close our bot's performance (according to the aforementioned success metrics) to their own judgement

5.4 Risks and contingencies

Possible risks include:

- drastic changes in the grading criteria,
- students abusing the model to provide answers for them,
- falling behind the development schedule.

Possible countermeasures for these risks are:

- model will be trained with regards to possible changes, with a focus on more generalized inspection of the answers,
- secure any potential exposure of the model's API, don't accept any prompts from the students, account for possible prompt injections in the answers, train the model to only comment (maybe stick to TAs approach, making very broad comments like "why", "define" and etc.).

6 Tactical plan

Our team decided to choose the **SCRUM** process framework. SCRUM is lightweight, easy to adapt. Given the iterative strategy, this framework seems as an approppriate choice.

Other candidates were eXtreme Programming (XP) and Team Unified Process (TUP).

XP is similar to SCRUM in many ways, but relies too much on developers experience and constant availability for real-life communication. Both cases do not relate to our team given the experience with the chosen tech stack.

TUP is approppriate for small teams, with a focus on team members' growth. But it is too heavy on overhead tasks, which may become problematic, given the strict deadline including a lot of several projects, team has to manage until the deadline.

SCRUM includes following concepts out team is going to adopt:

- Daily meetings - it's better to not tailor the processes from the start, but in this case, daily meetings won't be efficient because our team members have other duties and won't be able to dedicate every day to this project.
- Sprints - usually, SCRUM sprints take few weeks. Appropriate duration would be 2 weeks to implement features and take time to collect feedback
- Reviews - after each iteration it highly important to assess, whether we have reached the planned goals and how valuable our product to the customer.

6.1 Tasks

Tracking will be done with the Kanban framework. It is simple and popular. We suppose, that this will be enough.

Verification will be done with the peer review and feedback collection process in the end of each iteration.

Tasks assignment will be done during each scrum planning phase with respect to each team member strengths, briefly mentioned in the team responsibilities distribution.