Facilitating the Generation of Parametric Questions and their export to Moodle

Francisco de Assis Zampirolli, Valério Ramos Batista, Edson Pimentel, Juliana Braga
Federal University of ABC
09210–580 St. André, Brazil
Emails: {fzampirolli, valerio.batista, edson.pimentel, juliana.braga}@ufabc.edu.br

***Abstract*—[1] Online Education is becoming more and more disseminated in the present day, though it still has a hindrance concerning student assessment. This work presents a solution to create large Question Banks (QB) with parametric questions. This parametric question feature allows you to create more elaborate QB, where there are no duplicates of similar questions; Besides, it allows you to easily generate many variations when rendering an exam for a student. These QB are generated through an innovative solution adapted to the open-source web system MCTest. There one can settle the automatic production of numerous test variations that may even come in thousands. Then MCTest emails the QB to the user in two formats, Aiken and XML, which can be imported by another system, e.g. by Moodle for a quiz. For the second academic term of 2020 a Brazilian federal university decided to offer all of its courses in the distance learning format. One of them is called Networks and Communication (NC), which takes twelve weeks in total, and a professor is applying weekly quizzes on Moodle in order to prepare the students for the main exams. A total of 91 students of NC were assigned to this professor, 45 of them in the morning and 46 in the evening. He reported that the method presented in this article was very satisfactory to mitigate plagiarism, especially in online assessments.**

I. Introduction

Information and Communication Technologies (ICT) are increasingly present in Education, both for distance and classroom learning. The COVID-19 pandemic has led many institutions to adopt the use of ICT as a way to continue classes through Remote Education. Students, teachers, and professors must now adapt to this new reality, a context in which educational resources had to be produced and reused, especially regarding student assessment. One of the obstacles of online assessment is to build large banks of questions that permit to create more differentiated assessments for students.

Parametric questions is a way to automatically generate variations of questions by using *wildcards* that are replaced with random values (texts, numbers, formulas, figures, etc.) when the quiz is taken or randomly assigned by the Learning Management System (LMS) or in MCTest. This is more elaborate than *calculated questions* [1] used in LMS Moodle (moodle.org), whose wildcards are just common variable names (or simple formulas) in curly brackets, like $\{x\}$ and $\{y\}$, replaced with random values when the question is rendered.

[1]The first author grants #2018/23561–1, São Paulo Research Foundation (FAPESP).

MCTest's parametric mode enables the production of Question Banks (QB) by varying items much more than Moodle's limited option, in such a way that repeated question statements become practically impossible. Moreover, that mode easily generates many variations when producing students' tests. The first version of MCTest using Python and LaTeX was presented in [2].

According to [3] online tests are becoming widespread as an evaluation tool in higher education, and this requires a survey on their role in the contemporary learning assessment.

Moodle is one of the most popular free software [4]. Therefore, there is a demand on creating QB in LMS that optimize the evaluation of our students. For example, in [5] the authors utilized Moodle for periodic evaluation of 80 to 100 physics students per year in the programmers Physics, Mathematics and Electronic Engineering at the University of the Basque Country, which follows the Bologna Process [6]. In that paper the authors describe the construction of QB with more than 1,000 multiple-choice questions, which include conceptual and numerical problems, but there was not mentioned whether calculated questions generated the QB. Each periodic test had 10 random multiple-choice questions and was available for one week, with automatic feedback and without timing (no stopwatch for solving the test). Their work detects a positive correlation between the periodic tests and the final exam by considering 673 matriculated students from 2011 to 2015.

Another relevant subject is treated in [4] concerning the calibration of questions on Moodle by scholars, e.g. to endow each question with attributes like an objective in the Bloom's Taxonomy, a weight, etc. Such a weight could represent the question's level of difficulty. That paper details calibration in ten steps, which resort to a combination of the free software programs Moodle, OpenOffice and R. Some psychometric theories supply methods which help in this task, like the Classical Test Theory (CTT) and the Item Response Theory (IRT). Here we cite [7] as another work that used Moodle to calibrate QB. When a question is parametrized the calibration process becomes more tractable, because the exam questions will have the same content though in different texts. Hence any variation will assess the same amount of knowledge and skills for each student.

In [8] the authors propose a multi-agent system for students and teachers by means of Intelligent Tutoring Systems (ITS) applied to Moodle. The two implemented agents were sent to

both teacher and student (tutor). In this system both resources and activities are made available to each student according to their performance, which is classified as basic, intermediate and advanced. The agent Teacher was implemented by following the standard programming on Moodle for adding blocks, and this agent defines both resource levels and activities required in a course or topic. The agent Tutor is responsible for sending motivational messages to the students, according to each one's performance. Though the authors have not included empirical results of tests with the agents their work's ingenuity is surely worth mentioning.

Our present paper shows how to create QB with numerous variations by means of parametric questions, which will help improve evaluations with already existing ICT. It is organized as follows: Section II describes some related works; our studies are detailed in Section III, afterwards we explain and discuss our results in Section IV; finally, Section V is devoted to conclusions and future works.

II. Related Works

Moodle is a popular LMS with many resources like excellent usability and diversification at creating questions for a quiz, a preparatory exam, reinforcement exercises, etc. On Moodle we can make a timed quiz by drawing questions from QB and allow the student to go only forward. With parametric questions that produce numerous variations we obtain large QB, which makes it difficult to cheat the exam and also evaluates students homogeneously because each exam question will have the same level of difficulty.

On Moodle a *Calculated question type* is what we call a parametric one, which helps produce numerous variations of that question, but it must have at least one *wildcard*, namely a random variable.

As an example we comment the question "Calculate the area of a circle with a radius = {`radius`}, considering pi = {`=pi()`}". The solution is defined by `pi()*pow(radius,2)`, and when the student accesses that question it will come with a random value that replaces `radius`. Figure 1 illustrates this example on Moodle, which however restricts its calculated questions by resorting only to mathematical functions in PHP. Here are these of a single argument: `abs`, `acos`, `acosh`, `asin`, `asinh`, `atan`, `atanh`, `ceil`, `cos`, `cosh`, `deg2rad`, `exp`, `expm1`, `floor`, `log`, `log10`, `log1p`, `rad2deg`, `round`, `sin`, `sinh`, `sprt`, `tan`, `tanh`. And of at least two arguments: `atan2`, `pow`, `min`, `max`. In our example we used this one without arguments `pi()`.

Section IV presents an example of parametric questions created in MCTest that cannot be created as calculated questions in Moodle.

In [9] the authors detailed the production of QB applied to Moodle for a course called Introduction to Logic and Computer Science. In order to classify each question of the QB they used the Mathematical Assessment Hierarchy (MATH) taxonomy [10]. MATH is an adaptation of Bloom's Taxonomy [11], [12]. This most recent paper revises Bloom's

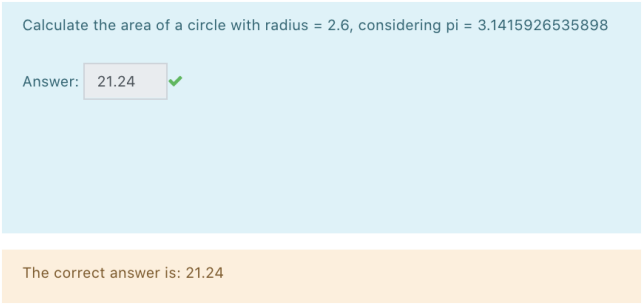

Fig. 1. Cutout of the Moodle page showing the calculated question.

Taxonomy by changing the names of its six levels that classify knowledge and skills of a student, basically by inverting the last two levels as follows: 5-Evaluate and 6-Create.

The QB created in [9] covered the three first levels of Bloom's Taxonomy, namely the student must 1-Remember, 2-Understand and 3-Apply some concept of the course (the subsequent level is 4-Analyze). The QB's creation process was semiautomatic: they implemented a program in Python to generate variations in YAML-format for each question, including many right and wrong answers with feedback. This study was applied to freshmen in Computer Engineering, who sat in total 15 weekly tests of up to 15min each, and these involving theory and resolution of problems. The final exam consisted of 10 theoretical and 23 practical questions. Only the students that passed the weekly tests were allowed to sit the final exam. They were 59 out of 150 in 2017-2018, and 82 out of 168 in 2018-2019. After some filtering the authors considered 119 students in order to compare results, in which 80 (67.2%) scored as much as, or more than, in the final exam. The authors neither detailed their process of creating scripts, nor mentioned their means to produce questions, hence we could not evaluate the technical difficulty that other professors would face at using their method. MCTest also accepts Bloom's taxonomy, presents a didactic way of creating questions on the web, and also uses Python, as we are going to see in the next section.

A new standard of question and test specification was introduced in [13], named Question and Test Interoperability (QTI) standards. This paper summarized the already existing standards by citing Global Learning Consortium (GLC), which is based on Question Markup Language (QML) and uses XML; and Reusable Competency Definition (RDC), which is adopted world-wide and defines a model to describe, reference and share competencies of online and distributed learning. The authors propose a QTI-based system developed in Java, in which they incorporate level of difficulty of questions, evaluated concepts, target-group (age, gender, time and degree of specialization, etc.). However, the article does not describe the use of parametric questions. MCTest stores questions and exams (including student responses) in an MYSQL Database. Questions are created using web interfaces or by importing from a file in TXT format, in a specific pattern [14]). MCTest

can also export questions in JSON format. When users create an exam, MCTest sends questions to their email in XML and Aiken formats, compatible with Moodle, as detailed in the next section.

The process described in [9] is quite similar to the one we introduce here. The main difference is that our study includes a combination with a web system that spares the elaboration of individual scripts for each question, whereas those authors generated their QB directly on Moodle.

III. Using Adapted MCTest

Here we explain the two steps that combine MCTest and Moodle to achieve a test that retrieves questions from a large QB. However, this section shows a simple example that could even be prepared directly on Moodle, whereas the contribution of MCTest's capabilities will be detailed in Section IV. As we have already mentioned, MCTestis endowed with complex built-in algorithms that enable the elaboration of sophisticated questions, a resource that Moodle still lacks.

A. Creating QB

We generate QB with the open source system MCTest available on GitHub. With MCTest one can either create or resort to already-existing entities Institution, Program, Course, Question, Class and Exam (see vision.ufabc.edu.br for details). We create and export QB as follows: on the subpage Exam the user chooses questions, classes and many other attributes of a test, like its division between Multiple-Choice and Written Response parts, its number of variations, etc. For example, in the case of a class with one hundred students the user can even ask for *many* hundreds of variations. Notice that random generation can occasionally repeat values, so the more questions the less probable it is that Moodle will retrieve the same test variation on two different occasions.

In the case of parametric questions MCTest generates a new question for each variation (for details on parametric questions in MCTest, see [14]). For the Multiple-Choice part it shuffles both the order of questions and of the respective alternatives, even for the non-parametric ones. Moreover, MCTest takes all involved classes and students to draw a test question for each student, and returns them all in a single PDF to the user for approval. This automatic step can be repeated until reaching an approved PDF, in which case MCTest emails the QB to the user in two formats, Aiken and XML, both supported by Moodle. For instance, typical QB-filenames for them are `report_Exam_139_variations_DB_aiken.txt` and `report_Exam_139_variations_DB.xml`, respectively. In this example 139 is an ID-number for the exam in the Database. Figure 2 shows a cutout of subpage Exam with options for files to be sent to the professor's email.

MCTest's version available on GitHub (latest access May 8 2021) is the most recent one, which accepts two types of questions: Multiple-Choice (QM) and Written Response (QT). On Moodle QT is called *essay*, which can only be corrected manually. In MCTest's case, if the question requires a short answer (either text or number), then it allows for automatic correction. This will also be performed by Moodle if one chooses *shortanswer* instead of *essay*. Either type can use parameters, and on MCTest even *many* thereof.

On the subpage Exam we create variations of a test that may include QM and/or QT questions, but the QT will always come after the QM in the generated questions. Each question is quantified with an integer level of difficulty from 1 to 5. Though MCTest generates random questions, in any of them all questions of difficulty δ will come before the ones of difficulty $1+\delta$, $\delta = 1, \ldots, 4$. Lest this arrangement ought not to appear the user can toggle them all to the same level. Anyway, the order of questions will be forwarded to the Aiken format explained in Section III-A1. Moreover, we can take a small universe of QM to select random subsets of fixed size for a test, e.g. 5 out of 10 QM for each question. This makes tests more uniform in content but also of great variety due to parametrization.

Now we explain the formats Aiken and XML generated by MCTest, and these can be imported by Moodle.

1) Aiken Format: Here MCTest simply replaces the aforementioned PDF with `Aiken.TXT`, so that each variation of the exam will come in this new format: variation 1 with its questions; variation 2 with its questions; etc. In this format, a question is identified firstly by its description, then by its alternatives, each consisting of a letter followed by either ")" or ".", a blank space and a text. In the following example the correct answer is D.

```
What is the square root of 50 with three
decimals?
A) 7.073
B) 7.072
C) 7.074
D) 7.071
ANSWER: D
```

2) XML Format: Moodle uses XML with various *tags* that are computer-readable, not human-readable as in Aiken. For MCTest to create categories of questions in XML, its source code was extended so that now MCTest rearranges all questions of all exam variations in the following sequence: Type (QT and QM); Topic; Difficulty; Short Description. For example, in Figure 3 we see five variations of the question `multichoice` (QM) with short description `add two vectors` and difficulty 1. Hence in XML this question becomes the sequence of categories `multichoice/1-vector/diff1/add two vectors`. On MCTest a category is called topic, but there are not subtopics. In this example `1-vector` is a topic of a course.

In the following XML-format we first specify the category, then the variations of a question from that category. Here MCTest's purpose is to generate QB with variations, hence for each new question we once again specify category and variations in the XML-format. For the sake of concision we indicate it by ellipsis in our example.

```
<question type="category">
```

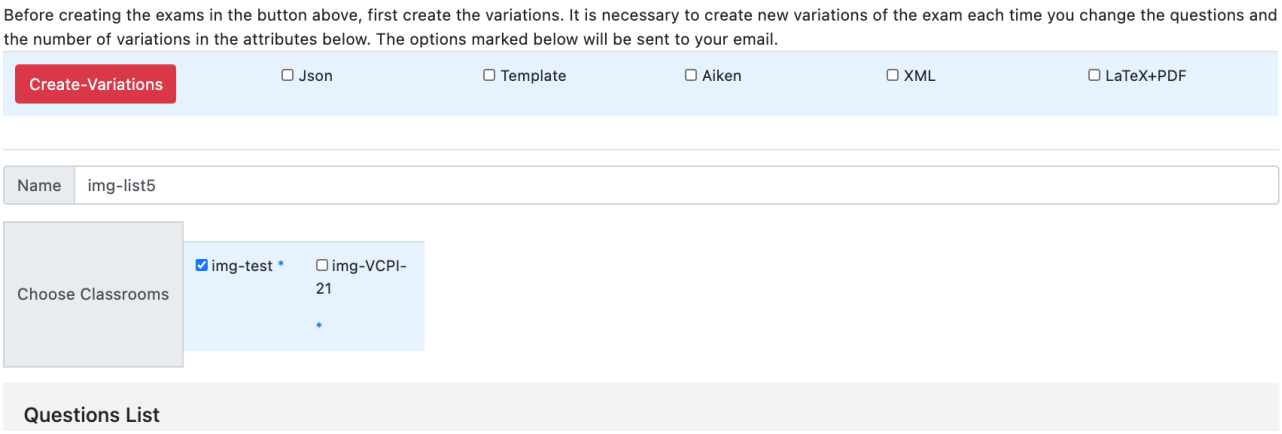

Fig. 2. Cutout of subpage Exam showing the creation part of variations of the exam, including files to be sent by e-mail to the professor in the formats: Json (for integration of the MCTest+Moodle+VPL, see [15] for details), Template (for automatic correction of questions outside MCTest, see [16] for details), Aiken, XML and LaTeX+PDF.

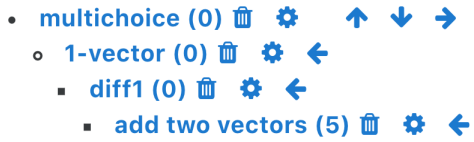

Fig. 3. Cutout of a Moodle subpage that shows the hierarchy of categories of a QB in XML-format. This QB was generated by MCTest and consists of five variations of a question. As indicated they all have difficulty 1, cover the topic `1-vector` and are of Multiple-Choice. The last category is a short description of the question: `add two vectors`.

```
  <category>
  <text>
    $course$/top/multichoice/1-vector/diff1/
    add two vectors
  </text>
  </category>
</question>
<question type="multichoice">
  <name>
  <text>Topic: 1-vector Difficulty: 1</text>
  </name>
  <questiontext format="moodle_auto_format">
  <text>What is the square root of 50 with
  three decimals?</text>
  </questiontext>
  <shuffleanswers>true</shuffleanswers>
  <answer fraction="0" format=
    "moodle_auto_format">
    <text>7.073</text>
    <feedback></feedback>
  </answer>
  <answer fraction="0" format=
    "moodle_auto_format">
    <text>7.072</text>
    <feedback></feedback>
  </answer>
  <answer fraction="0" format=
    "moodle_auto_format">
    <text>7.074</text>
    <feedback></feedback>
  </answer>
  <answer fraction="100" format=
    "moodle_auto_format">
    <text>7.071</text>
  </answer>
</question>
...
```

B. *Creating a Quiz from QB*

In order to import on Moodle a QB-file generated by MCTest, e.g. `report_Exam_139_variations_DB.xml`, we had to settle the following steps valid for Moodle 3.5.8: on a subpage Course click on the upper right cogwheel icon and choose `More...` With `Question bank` choose `Import`, `Moodle XML format` and move the XML-file to the region `You can drag and drop files here to add them`. Finally click on `Import` and `Continue`. The QB is then ready for access in a quiz.

Now we must create a quiz on Moodle. On the aforementioned subpage Course click again on the upper right cogwheel icon but now choose `Turn edition on`, then `Add an activity or resource` and `Quiz`. Give a name to the test, then `Save and display`. On the subsequent page take the steps `Edit quiz`, then `Add` and `a random question`. For example, choose the category `add two vectors`. On the subsequent page one can choose the number of questions to be drawn from this category, let us say 2 for `Number of random questions`. Then click on `Add random question`. This will prompt Figure 4, whence questions are mixed with `Shuffle`.

As we have already mentioned, MCTest allows for numerous variations of complex parametric questions, and this is precisely the advantage of our study. In the next section we detail how to produce these variations.

IV. Results and Discussions

In the course Introduction to Computer Science, a typical example of question is “write a code that calculates the scalar product of any pair of vectors”, illustrated in Figures 5, 6 and

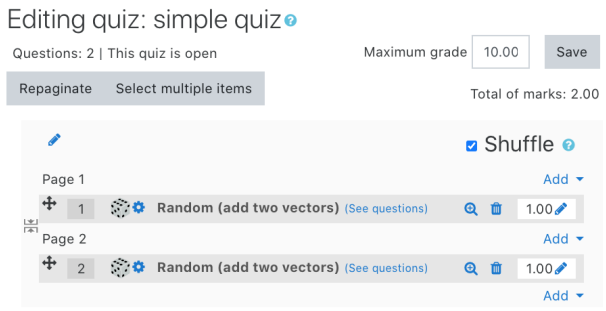

Fig. 4. Cutout of the Moodle subpage that creates a quiz.

7. When elaborated on MCTest the most important part of this question will be the field `Description` in Figure 5. There the statement is written in LaTeX and Python, this one for parametrization. The first part describes the question and the second is the Python code between `[[def:` and `]]`. The first part also includes a random variable for the size of the vectors defined by `[[code:n]]`. Right after `[[def:` one attributes a value to the parameter `n`.

Figure 6 shows the second step to create/update a question. In case the user toggles `Type` as "Multiple-Choice Question", the field `Answer Text` will appear and at least the token `[[code:correctAnswer]]` must be included there because the question is parametric. In Figure 5 the last but one line of `Description` is `correctAnswer = np.dot(a,b)`, whence the function `dot` of the numpy Python library computes the scalar product of the vectors `a` and `b`. These two arguments were defined by the function `randint` of the same library, and their `n` entries vary between 0 and 8. As already mentioned, right after `[[def:` we have `n=np.random.randint(5,9)`, which sets `n` between 5 and 8 to define the size of the vectors.

In Figure 6 the user also includes wrong alternatives, for instance `[[code:correctAnswer-1]]`. Further wrong alternatives are added similarly, and even without the token (for N/A, an impossible result like 650, etc.) By clicking on `Create-PDF` in Figure 5 we get the layout of a question illustrated in Figure 7. Finally, Figure 8 shows the question uploaded to Moodle in the XML file generated by MCTest.

We can profit this example of scalar product to create another question with parameters, which will increase the number of variations and make cheating even less feasible. For instance, by introducing the parameter `order = np.random.choice(['increasing', 'decreasing'])` and adapting the question for the student to sort either vector a or b, or even both, and then compute the scalar product. Hence each alternative will come out with a scalar product and the sorted vector(s). Sorting may be asked up to a certain position $k < n$, among other possibilities. Therefore, the code required by the question will have to include sorting besides just the use of a function like `dot` for the scalar product. Namely, it is quite improbable for the student to find such a code already implemented on a web page. At least they would have to merge codes carefully in order to solve the question.

Now we present an experience report of a professor that is already using our method.

A. Experience Report

For the second academic term of 2020 a Brazilian federal university decided to offer all of its courses in the distance learning format. One of them is called Networks and Communication (NC), which takes twelve weeks in total, and the professor is applying weekly quizzes on Moodle in order to prepare the students for the main exams. Each exam is preceded by a formative test that contributes as a bonus of up to 5% on the final grade. This professor is following the same strategy of distance student evaluation presented in [16], which was already incorporated by MCTest and therefore counts on many facilities to apply individual exams through the students' institutional emails. They use a Google Form to upload their solutions, which are automatically sent to a spreadsheet in the professor's Google Drive.

B. Context of the Experiments

Distance student evaluation is of course much less reliable than exams applied in a classroom under the supervision of one or more examiners. But the coronavirus pandemic created a state of exception, and therefore we resort to methods that enable learning activities.

NC belongs to the programme Bachelor in Science and Technology (BST), which takes three years with three trimesters each. One of its first courses is called Foundations in Computer Science, followed by Nature of Information and Processing of Information in the upcoming academic terms. NC is lectured in the fourth academic term, which occasionally started at the end of September 2020. By the way, that was when we invited the professor to use our method in order to prepare quizzes on Moodle.

A total of 91 students of NC were assigned to this professor, 45 of them in the morning and 46 in the evening. Since both classes use exactly the same material, the professor records the weekly live meetings for both periods and chooses the best parts to be uploaded as video lectures on his YouTube channel. Each lecture is recorded in different parts that separate theory interposed by related to exercises.

For example, in order not to affect students who have little access to the internet, every exam must allow 72 hours for them to finish. For Moodle activities the professor chose weekly quizzes elaborated with our method. Students can simply use a smartphone to read the questions and write their answers.

C. Experiments

As we have mentioned in Section III, some quizzes can be directly prepared on Moodle but this platform has limitations. For instance, it fails to interpret some LaTeX commands like variables inside a text, which come between a left and a right $. Moodle quizzes cannot embed either graphs or figures in the

Fig. 5. Cutout of the subpage on MCTest to create/update a question. In this first step one must fill out the fields `Choose Topic` and `Description` (see [14] for details on the text contained in this description), optionally also `Short Description` and `Group` (in this case only one question per group will come out in each exam question).

statement, hence the professor avoided them but wanted to use MCTest because he was going to update the quizzes with these resources in order to prepare exams. See how to do it with MCTest in [14]. Moreover, this professor finds Moodle useful for keeping lecture notes, lists of exercises and a centralized contact with his students. But if they want to recommend his video lectures then YouTube is better because their access to Moodle is restricted to the university.

For similar reasons the professor prefers to prepare all of his evaluations in MCTest, a platform exclusively devoted to student assessment, and there he counts on many tools to produce almost any kind of test. Namely, he prefers not to unify the whole teaching activity on a single resource, but to use the best of four different ones: Moodle (for static teaching material), YouTube (for dynamic material), Zoom (for live lectures) and MCTest (for class evaluations). Students do not have access to MCTest, and there he was able to fully adapt old exams and lists of exercises already written in LaTeX. In 2019 MCTest was improved by allowing embedded Python codes together with LaTeX statements, as described in [14]. The professor has been using this facility to generate numerous variations of activities, so that they come individualized for each student.

Another experience report of the method presented herein was given by a professor who lectured Functions of a Unique Variable (FUV), a course that introduces derivatives and integrals. In [16] we presented an adaptation of MCTest whose exams in PDF are each emailed to the respective student, and on that occasion there were one hundred matriculated in FUV. The automatic correction was carried out by Google Form & Spreadsheet. According to [16], even with numerous variations of the exam its Multiple-Choice part was biased towards high scores, but the Written Response question was well-balanced. It required uploading an image of the complete student's handwritten solution, including signature and ID. That professor is now using the same parametric questions on MCTest to generate QB in XML, and then export it to Moodle, because this platform unifies access for students and professors. But he had to adapt the statements, since LaTeX reserved words cannot be handled by Moodle. It however accepts formulas between "\(" and "\)" in a sentence, or even centred by "$$".

V. Conclusions and Future Works

We have just presented an original and promising study to generate QB in which one can have numerous variations

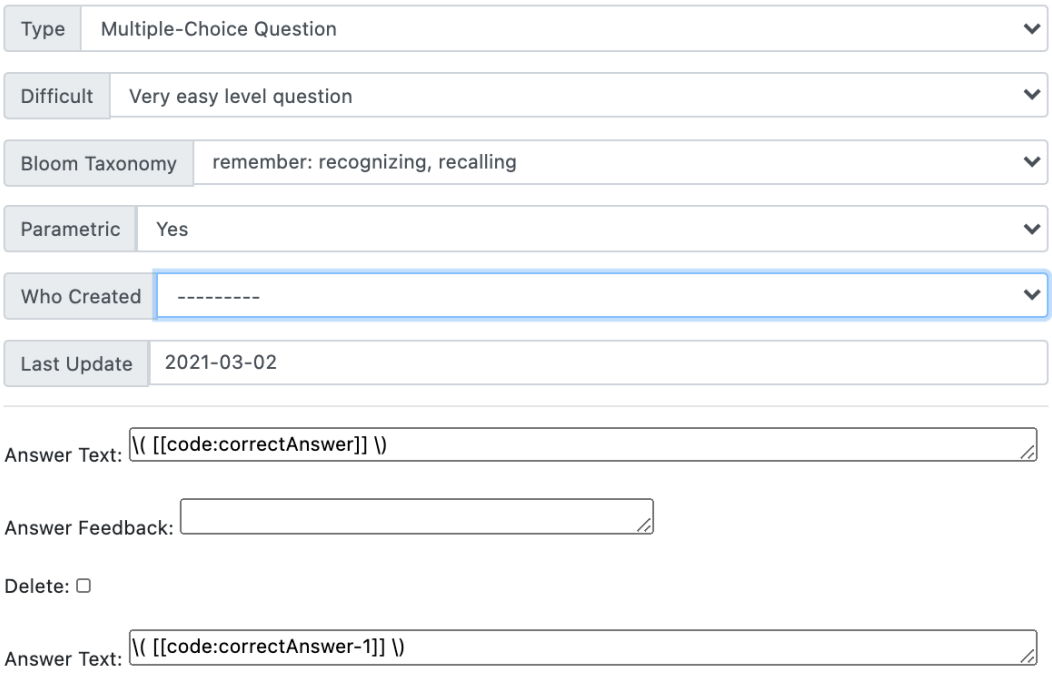

Fig. 6. Cutout of the subpage on MCTest to create/update a question. In this second step one must toggle `Type` as either Multiple-Choice or Written Response (in which case the question will become similar to the one in Figure 1) and choose one of the five levels in `Difficult`. The fields `Bloom Taxonomy`, `Who Created` and `Last Update` are optional, but not toggling `Parametric` to Yes will make MCTest compile `Description` as if it were all in LaTeX. Then the user must give alternatives (always put the right one in the first field), two of them already depicted in this example.

MCTest

Topic: 5-vector
Group:
Short Description: scalar product
Type: QM
Difficulty: 1
Bloom taxonomy: remember
Last update: 2021-03-02
Who created:

#1663 1. Write a program that, given two vectors $a = [a_1, a_2, ..., a_n]$ and $b = [b_1, b_2, ..., b_n]$, both with n values, perform the following operation, know as scalar product:

$$\sum_{i=1}^{n} a_i b_i = a_1 b_1 + a_2 b_2 + \cdots + a_n b_n.$$

Considering the vectors below (size $n = 7$ each), what is the scalar product between them?
a = [2,7,5,1,6,4,8]
b = [6,3,4,2,8,8,2]

A.*3156 B.*1150 C.#0151 D.*2146

Fig. 7. Cutout of the subpage of MCTest right after clicking on `Create-PDF` in Figure 5. This question has C as the correct alternative, which came from the first field `Answer Text` in Figure 6, here located with #0.

of a same question, providing parametric tokens are used adequately. These QB were generated through the web open source system MCTest available at GitHub for free. Hence anyone can download it, make changes and install either for their institution or just for themselves.

The QB are generated in both formats Aiken and XML, the former being human-readable and the latter machine-readable with many *tags* to give the best characterization to each question. Files in these formats can be imported by Moodle, and we get categorized QB if XML is adopted. After importing the QB, the teacher/professor can create a quiz by means of several resources available on Moodle, like timed questions, with or without feedback, going only forwards, etc.

Future works could include and adapt some capabilities of

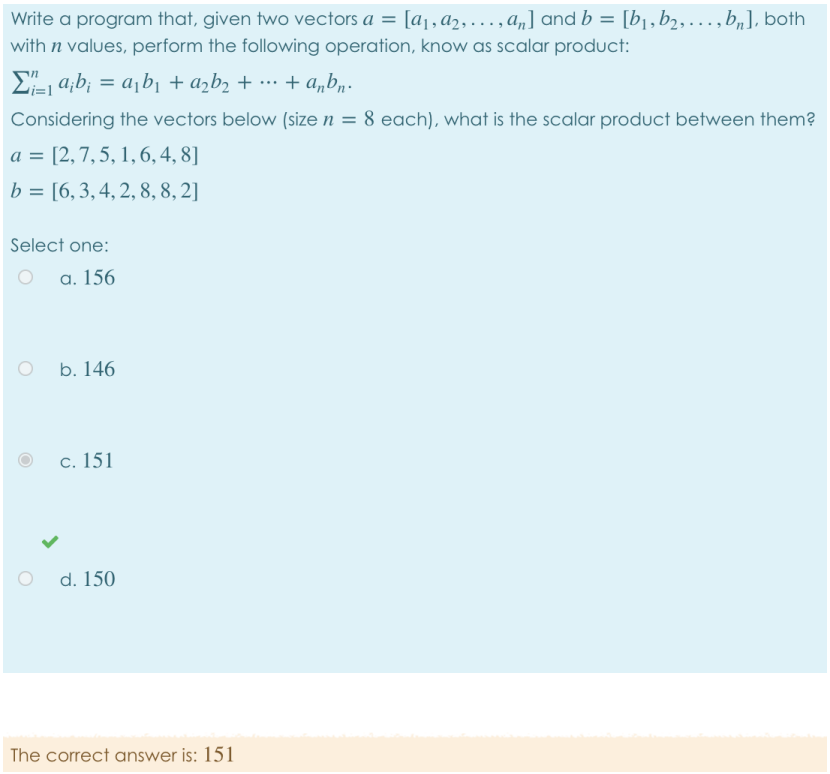

Fig. 8. Cutout of the Moodle page showing a quiz from imported XML-file generated by MCTest, more specifically the question presented in Figures 5 to 7, such file with numerous shuffled alternatives.

Moodle, such as adding more information about each question through labelling it, specially the ones students have already answered. For this purpose we could resort to multi-agent systems [8], IRT [4], [7] and recommendation systems [17].

References

[1] "Calculated question type," [Online; accessed 19-July-2021]. [Online]. Available: https://docs.moodle.org/35/en/Calculated_question_type.

[2] F. A. Zampirolli, V. R. Batista, and J. A. Quilici-Gonzalez, "An automatic generator and corrector of multiple choice tests with random answer keys," in *2016 IEEE Frontiers in Education Conference (FIE)*. IEEE, 2016, pp. 1–8.

[3] B. Boitshwarelo, A. K. Reedy, and T. Billany, "Envisioning the use of online tests in assessing twenty-first century learning: a literature review," *Research and Practice in Technology Enhanced Learning*, vol. 12, no. 1, p. 16, 2017.

[4] C. Presedo, A. J. Arméndariz, J. López-Cuadrado, and T. A. Pérez, "Calibración de ítems vía expertos utilizando moodle," *Revista Iberoamericana de Educação*, vol. 69, no. 1, pp. 117–132, 2015.

[5] G. A. López, J. Sáenz, A. Leonardo, and I. G. Gurtubay, "Use of the moodle platform to promote an ongoing learning when lecturing general physics in the physics, mathematics and electronic engineering programmes at the university of the basque country upv/ehu," *Journal of Science Education and Technology*, vol. 25, no. 4, pp. 575–589, 2016.

[6] D. Kennedy, A. Hyland, and N. Ryan, "Learning outcomes and competences: Bologna handbook, introducing bologna objectives and tools, 1–18," *Prieiga per internet*, 2009.

[7] J. M. Azevedo, E. P. Oliveira, and P. D. Beites, "Using learning analytics to evaluate the quality of multiple-choice questions," *The International Journal of Information and Learning Technology*, 2019.

[8] P. Giuffra, E. Cecilia, and R. A. Silveira, "A multi-agent system model to integrate virtual learning environments and intelligent tutoring systems," *International Journal of Interactive Multimedia and Artificial Intelligence*, 2013.

[9] M. Bakó and L. Aszalós, "The creation and application of a question bank for an introductory logic module." *Acta Didactica Napocensia*, vol. 12, no. 2, 2019.

[10] G. Smith, L. Wood, M. Coupland, B. Stephenson, K. Crawford, and G. Ball, "Constructing mathematical examinations to assess a range of knowledge and skills," *International Journal of Mathematical Education in Science and Technology*, vol. 27, no. 1, pp. 65–77, 1996.

[11] D. R. Krathwohl, B. S. Bloom, and B. B. Masia, *Taxonomy of educational objectives: The classification of educational goals*. David McKay Company, Incorporated, 1956.

[12] L. W. Anderson and D. A. Krathwohl, "A taxonomy for learning, teaching, and assessing: A revision of bloom's taxonomy of educational objectives," 2001.

[13] M. H. Zedan and H. A. Hassan, "An aligned assessment item authoring environment based on interoperability standards," *International Journal of Modern Education and Computer Science*, vol. 5, no. 12, p. 1, 2013.

[14] F. A. Zampirolli, F. Teubl, and V. R. Batista, "Online generator and corrector of parametric questions in hard copy useful for the elaboration of thousands of individualized exams." in *CSEDU (1)*, 2019, pp. 352–359.

[15] F. A. Zampirolli, P. H. Pisani, J. M. Josko, G. Kobayashi, F. Fraga, D. Goya, and H. R. Savegnago, "Parameterized and automated assessment on an introductory programming course," in *Anais do XXXI Simpósio Brasileiro de Informática na Educação*. SBC, 2020, pp. 1573–1582.

[16] F. A. Zampirolli, V. R. Batista, E. Arrazola, and I. A. Junior, "Online assessments with parametric questions and automatic corrections: an improvement for mctest using google forms and sheets," in *Anais do XXXI Simpósio Brasileiro de Informática na Educação*. SBC, 2020, pp. 51–60.

[17] H. Zhang, T. Huang, Z. Lv, S. Liu, and H. Yang, "Moocrc: A highly accurate resource recommendation model for use in mooc environments," *Mobile Networks and Applications*, vol. 24, no. 1, pp. 34–46, 2019.