**Take-Quiz Module Level Description**

3.One use case describing the principal user interaction with your project component.

**Title:** Take Quiz

**Primary Actor:** Student

**Stakeholders and Interests:**

Student: wants the quiz recorded

Instructor: wants quiz submission

**Precondition**: Student logged in to quiz selection page.

**Main Scenario**:

1. SuD presents quizzes available to Student
2. Student selects quiz
3. SuD presents any quiz special instructions
4. Student starts quiz
5. SuD presents quiz question
6. Student enters answer
7. SuD saves and advances to next question
8. Student repeats 5-8 as desired
9. Student submits quiz solution
10. SuD confirms quiz submission

**Extensions**:

6a. Suspend Quiz

6a1. Student suspends quiz session

6a2. SuD maintains answers for subsequent quiz session

10a. Submission rejected

10a1. SuD says quiz incomplete

10a2. Student forces incomplete submission

Variation: 6b. Student selects any other quiz question

Variation: 6c. Student changes previous answer

4. Clear connection between your module and its responsibilities from a project requirements list.

**Requirement Header 1.3**: User can take a Quiz

**Link to requirements List used**: <http://www.cs.mun.ca/~brown/cs2005/Notes/project/projrequirements.html>

**Requirement1**: Users can take permitted Quiz, Access Limitations are checked

**Method that solves requirement**: checkAccess()

Check Access:

* Student must be logged in.
* Student must be listed as part of student to take the quiz.
* The max attempts for the quiz must not be already exceeded by student.
* The deadline for the quiz must not be exceeded.

**Requirement2**: Users can have multiple attempts.

**Method that solves requirement**: Take quiz constructor\_\_init\_\_()

Constructor:

* Creates a new attempt for that quiz when someone tries to take a quiz again.

**Requirement3**: Users can pause quiz.

**Method that solves requirement**: stopQuiz(), resumeQuiz()

Pause Quiz and Resume Quiz:

Student presses the pause quiz button to attend to an urgent situation

* stopQuiz() adds the students incomplete quiz-attempt to the student quiz-attempt dictionary.
* The dictionary is then persisted
* resumeQuiz() resets the present quiz-attempt to the incomplete quiz-attempt when the user decides to continue the quiz.

5.  pydoc and PEP257 compatible docstrings for your module

* Modules have been documented with their docstrings.