BTech451 Semester Presentation

Dian LIN

Code Runner Extension

Introduction: Push up barriers(code implementations) into Code Runner against cheating.

Motivation: The Functionality of cheating detection is not covered yet by Code Runner. Which may cause:

1. Unfair to hard work students
2. Lost faith in Code Runner
3. Students try to cheating in Code Runner

Application/Tool(Preparation)

- **SandBox**: used to run the series of test cases for input source code under limited time which prevents infinite loops or deadlocks from blocking the system. And effectively prevent malicious codes.(guard system).
- **MySQL**: open-source relational database management system(RDBMS). Allowed users to create relationships between tables by primary keys and foreign keys.

Ideas

- **Similarity checking**: the straightforward idea by checking submissions similarity. (research required)
- **Programming variation**: the answers of Code Runner questions could be obtained online or translated from different languages. (research required)
- **Functionality addition**: more functionality are possible to be implemented into Code Runner. (based on the research results)

Research Process

(Similarity Checking)

Length of Program	Similarity percentage
3 lines (shortest)	100%
3 - 67 lines	68% - 77%
67 lines(longest)	56.6%
Median Length: 30 lines	Median similarity:76.7%(relatively high)

Analysis: High similarity would occur due to short coding length, same question and example provided in lectures.

Conclusion: Similarity checking is not a feasible anti-cheating approach because of above reasons.

Research Process

(Programming variation)

- Question types being selected:

1. recursive question: summing numbers.
2. pre-define question: binary tree.
3. sorting question: bubble-sort, insertion-sort and merge-sort.

- **Analysis**:

1. Solutions of recursive and sorting questions are both easily found online, but pre-define question are not possible since Code Runner define Class for participants.
2. Programming languages translations is an unavoidable personal skill which is not able to be detected by Code Runner.

Conclusion: pre-defined questions become a preferred question creation way but can't prevent copy-paste cheating.

Idea Implementation

- The promising idea to implement for Code Runner is to make sure every participant gets different questions.
- Idea: One Question has more than one variant, such as Boolean question for Odd or Even number.
- The more variants in one question, the more probability of cheating would be reduced.

Proof of Concept

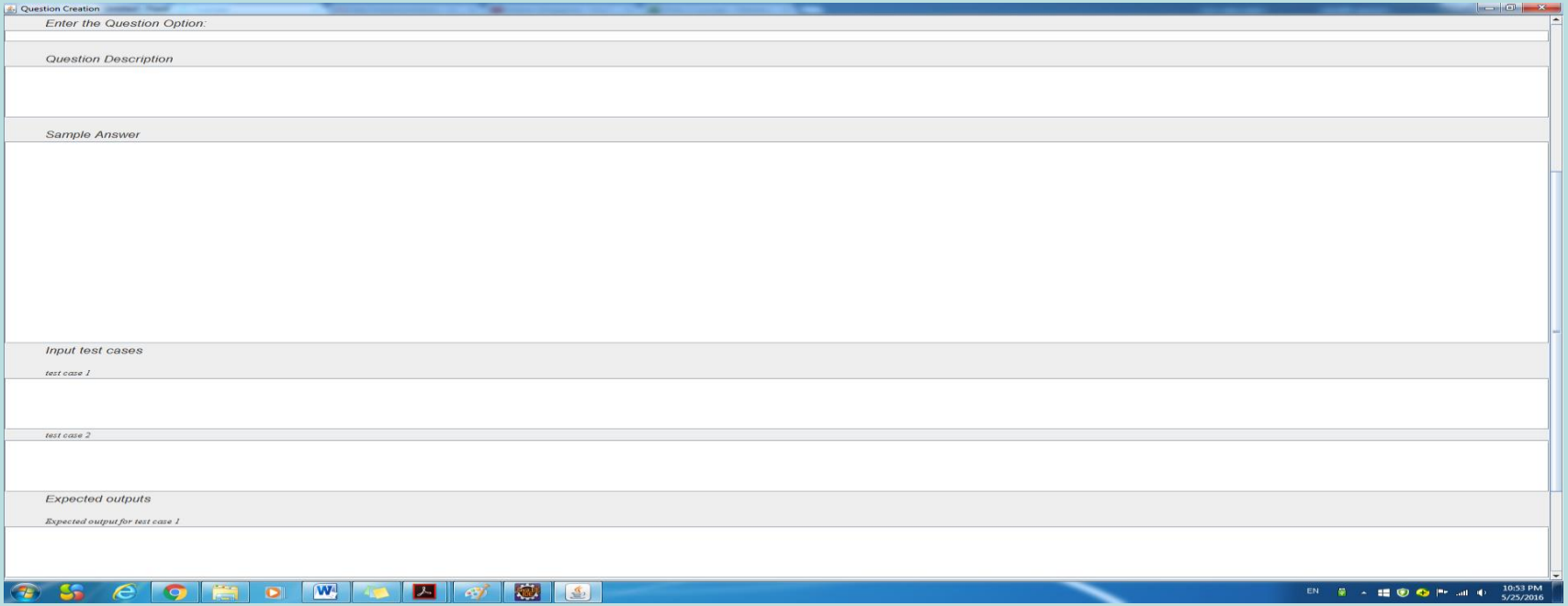
- Idea from last sildes relates to question creation page but other funtionality may remain what they used to be.

- We don't know yet if the idea is possible to be implemented and successfully working with MySQL Database, prototype should be made instead of directly moving to Code Runner.

- The GUI mock-up should be same as Code Runner but additional 'option' (variant) functionality needs to be added.
- All question details and test cases in prototype should be store into same database as Code Runner does.

GUI Mock-up Implementation (Java)

Question Creation page:



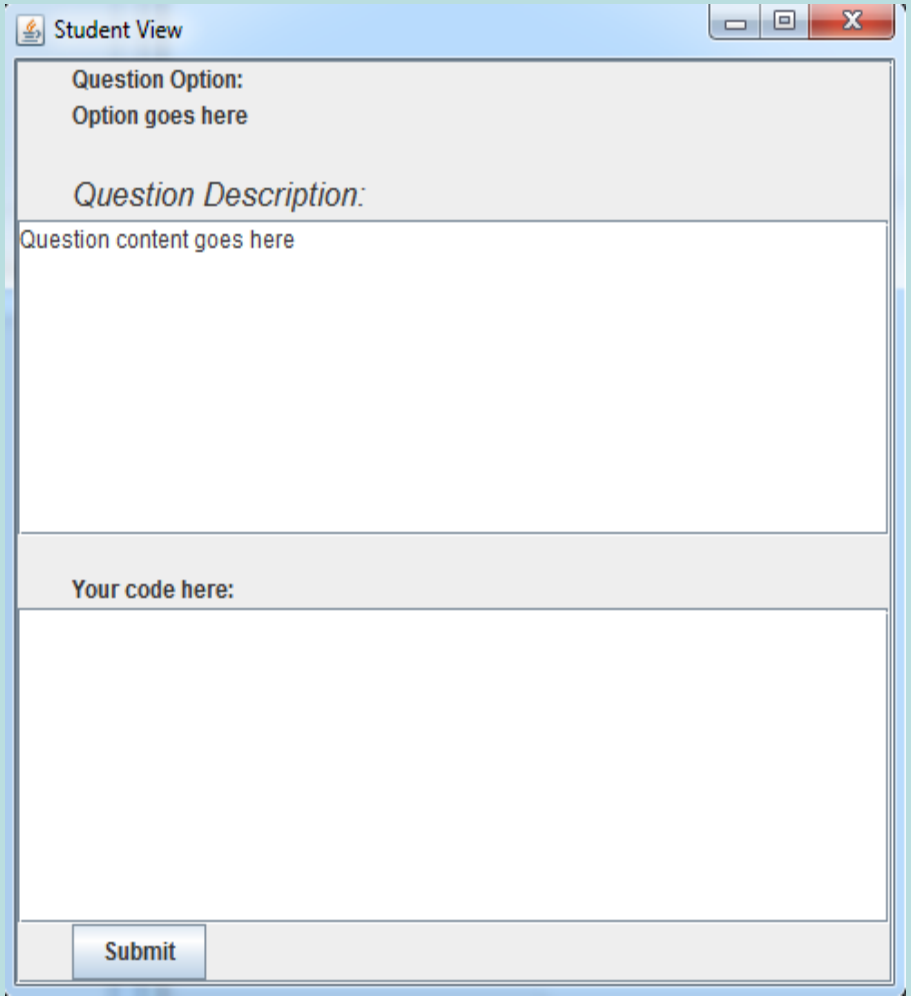
Administrator/Lectures variants view (after Odd and Even variants being added):

Question table below after first question being created

A	B	C	D	E	F	G
Option	Question Descripti...	Sample Code	TestCase1	Expected output1	TestCase2	Expected output1
Odd	This is Odd test	public static boole...	System.out.println(...	false	System.out.println(...	true
Even	This is even test	public static boole...	System.out.println(...	true	System.out.println(...	false

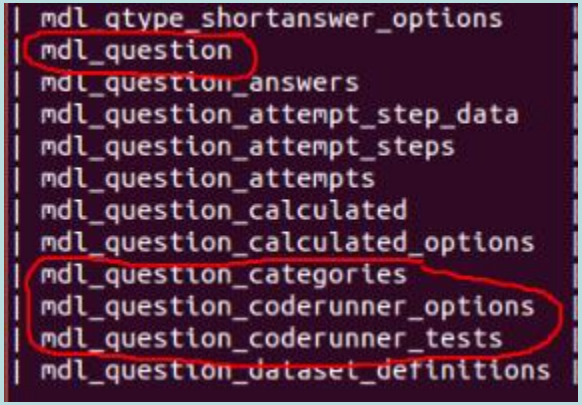
GUI Mock-up Implementation (Java)

- Student question view:



MySQL Database lookup

- All data displayed can be retrieved from Code Runner database, and what we need is to review what tables in the database relate to question creation. Records from following 4 circled tables are detected to be changed when created new questions.



Therefore, MySQL database is connected, and these 4 tables are used to store new records from my prototype.

MySQL Database lookup

mdl_question schema

Field	Type	Null	Key
id	bigint(10)	No	PRI
category	bigint(10)	No	MUL
name	varchar(255)	No	
questiontext	longtext	No	
qtype	varchar(20)	No	

mdl_question_categories schema

Field	Type	Null	Key
id	bigint(10)	No	PRI
name	varchar(255)	No	
contextid	bigint(10)	No	MUL

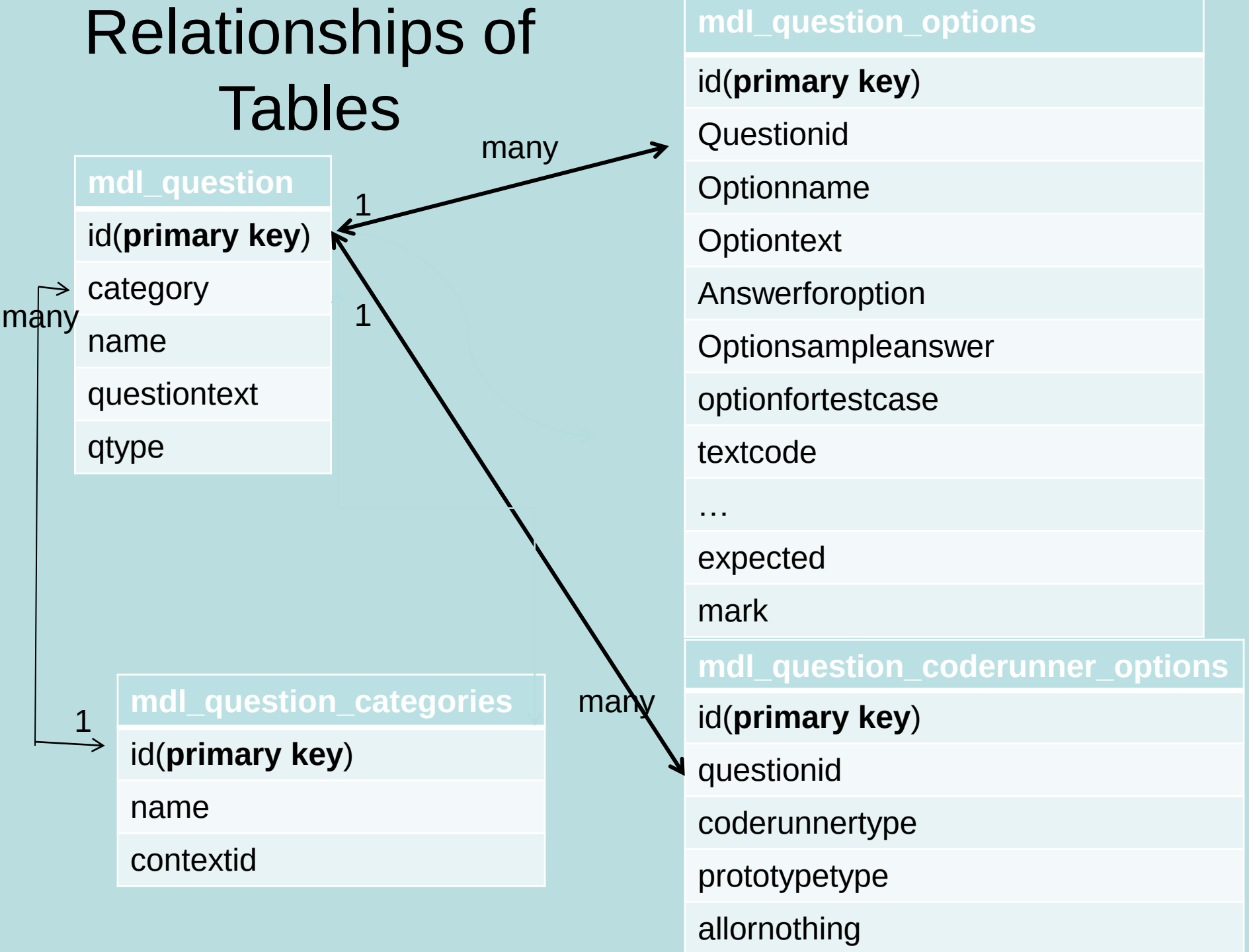
mdl_question_coderunner_tests schema

Field	Type	Null	Key
id	bigint(10)	No	PRI
questionid	bigint(10)	No	MUL
testcode	longtext	YES	
stdin	longtext	YES	
expected	longtext	YES	
mark	decimal(8,3)	No	

mdl_question_coderunner_options schema

Field	Type	Null	Key
id	bigin(10)	No	PRI
questionid	bigint(10)	No	MUL
coderunnertype	varchar(255)	No	MUL
prototypetype	tinyint(1)	No	MUL
allornothing	tinyint(1)	No	

Relationships of Tables

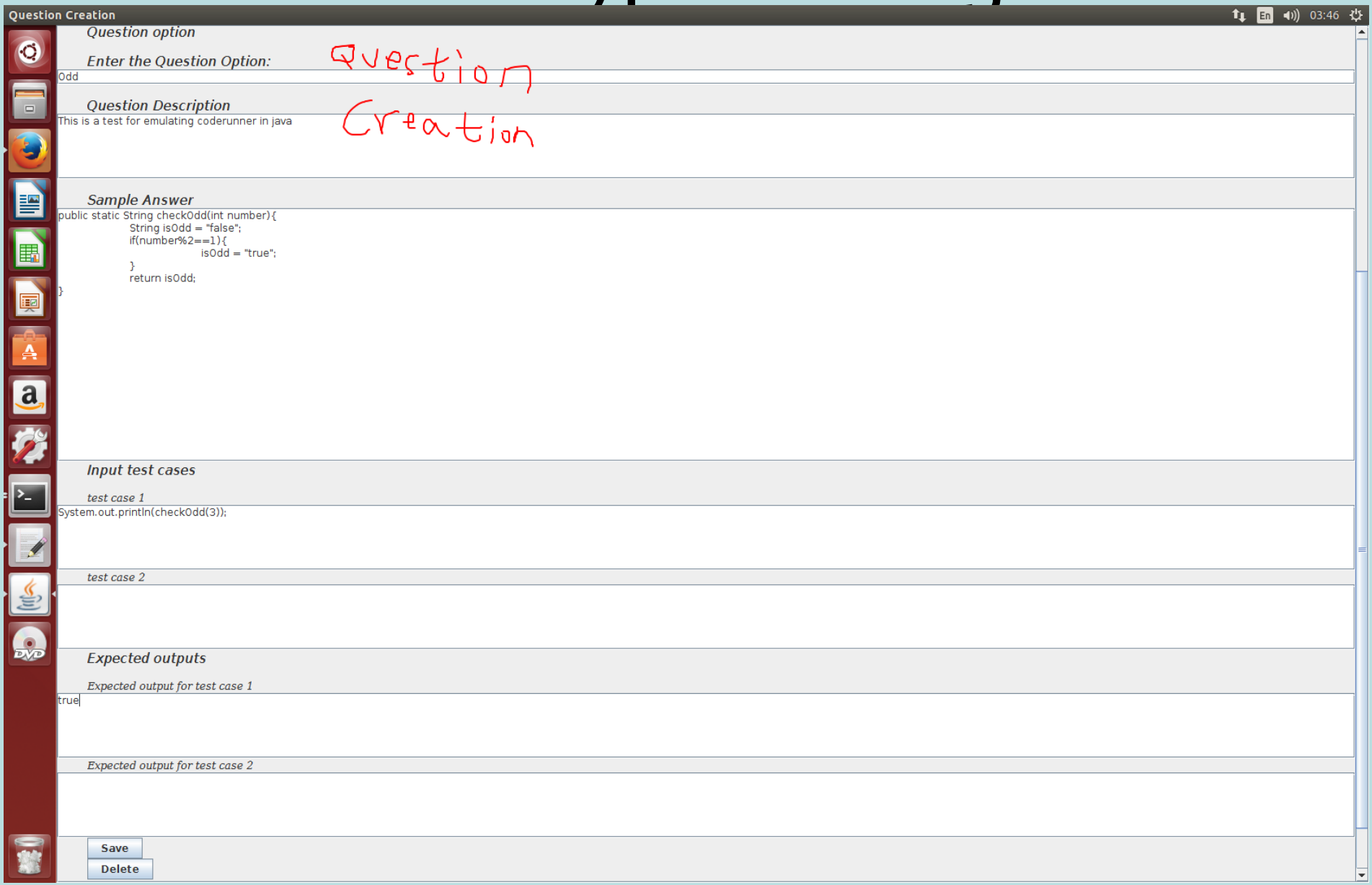


Prototype Testing

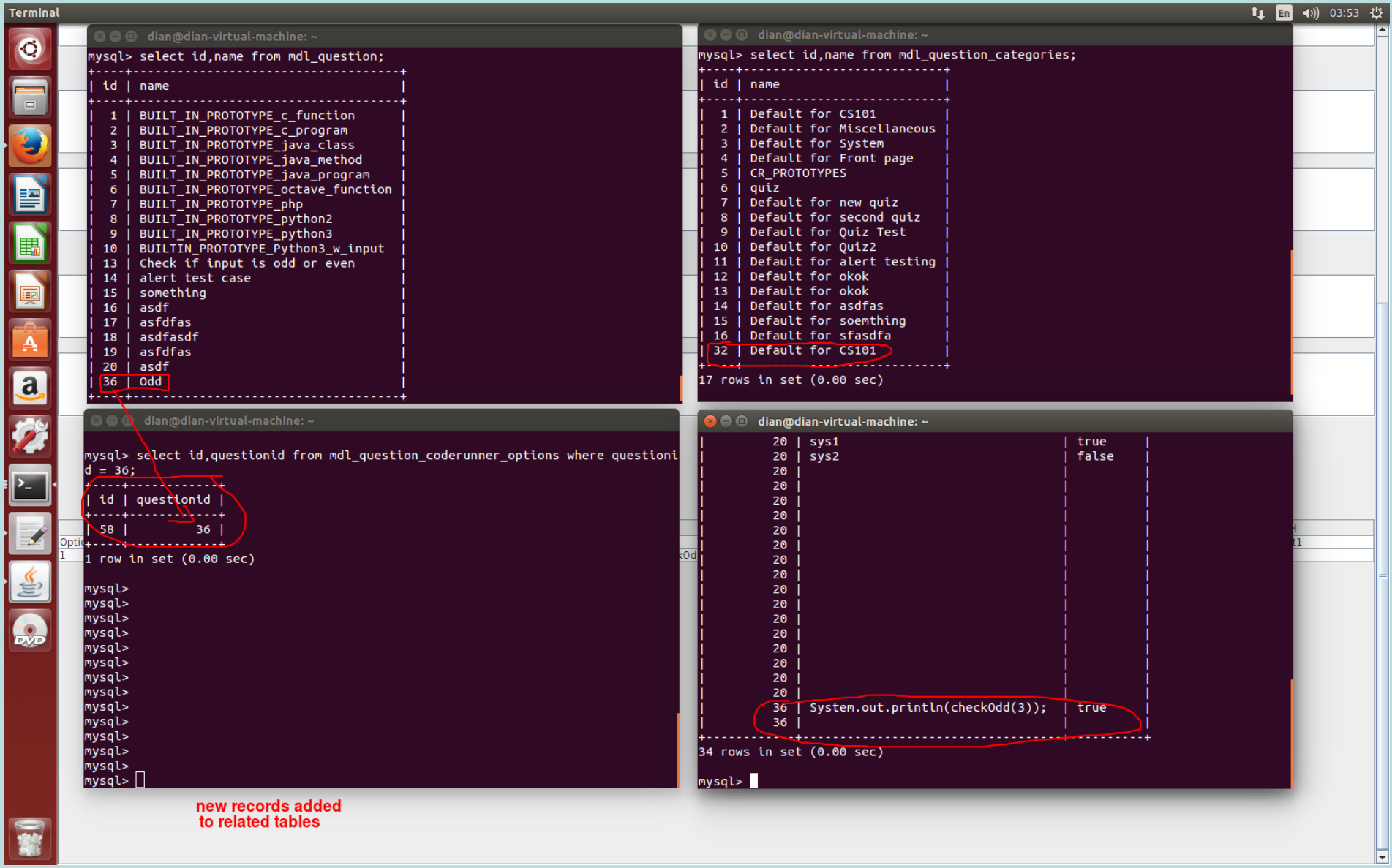
- After GUI Mock-up and functionality implemented, we need to test if it is working as Code Runner does.

- Question creation test(record insertion)
  - In GUI Mock-up, type new question details, test case and expected output
  - all data above should be inserted into MySQL database as new records
  - question details should be able to display in both lecture's view and student's view

Prototype Testing



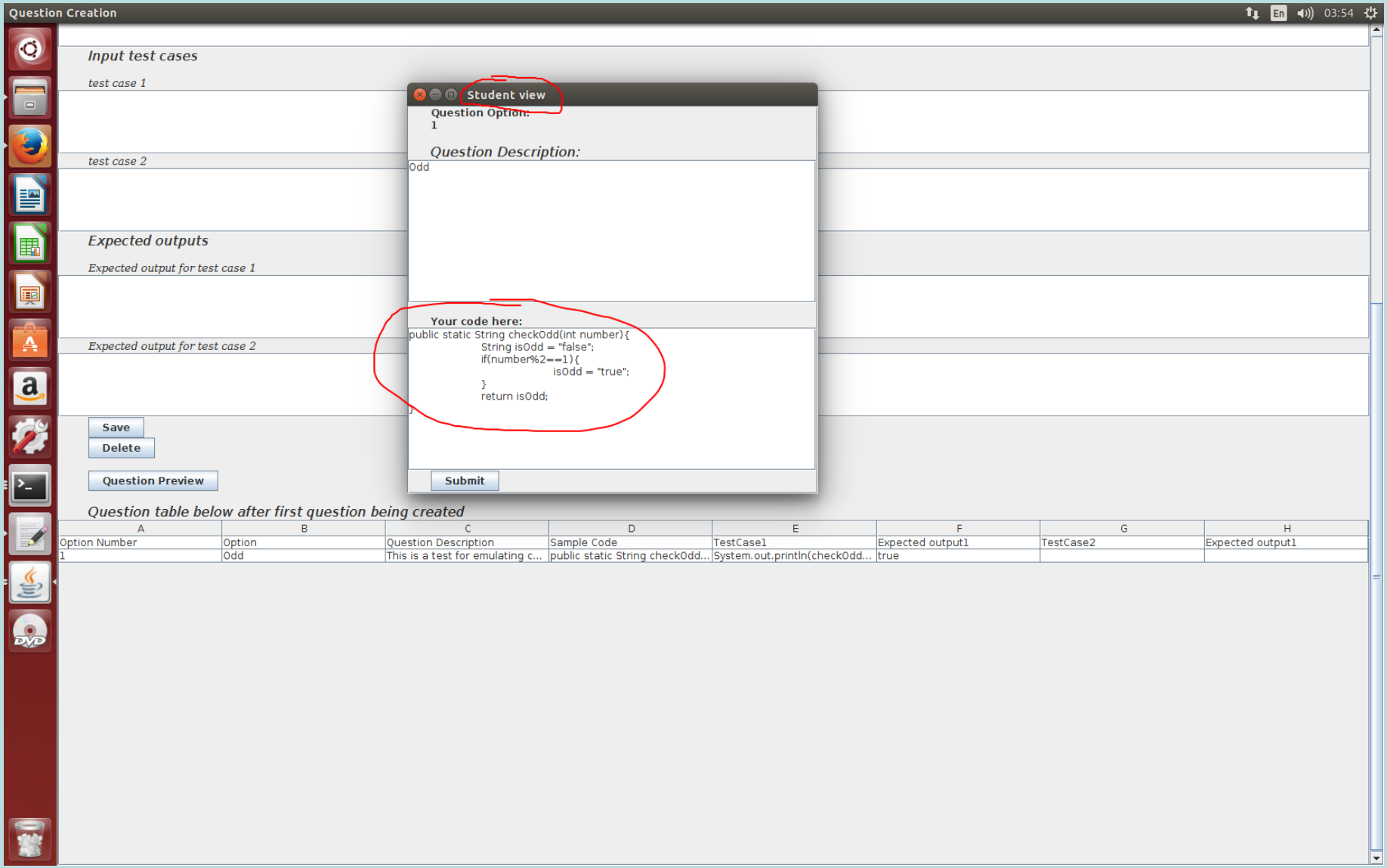
Prototype Testing



new records added
to related tables

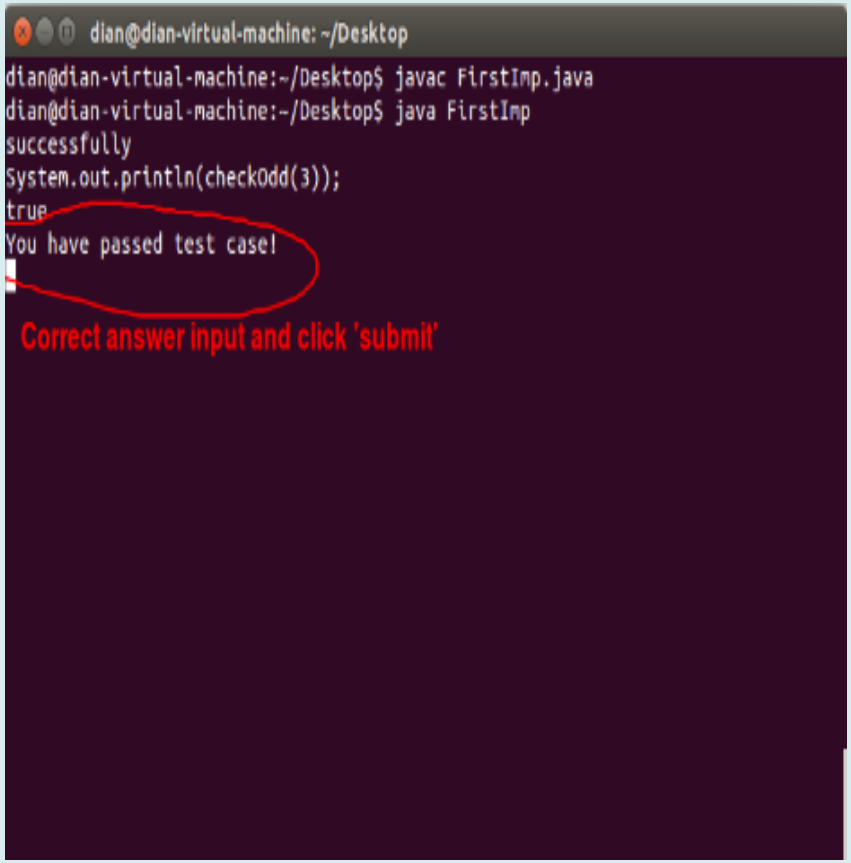
Prototype Testing

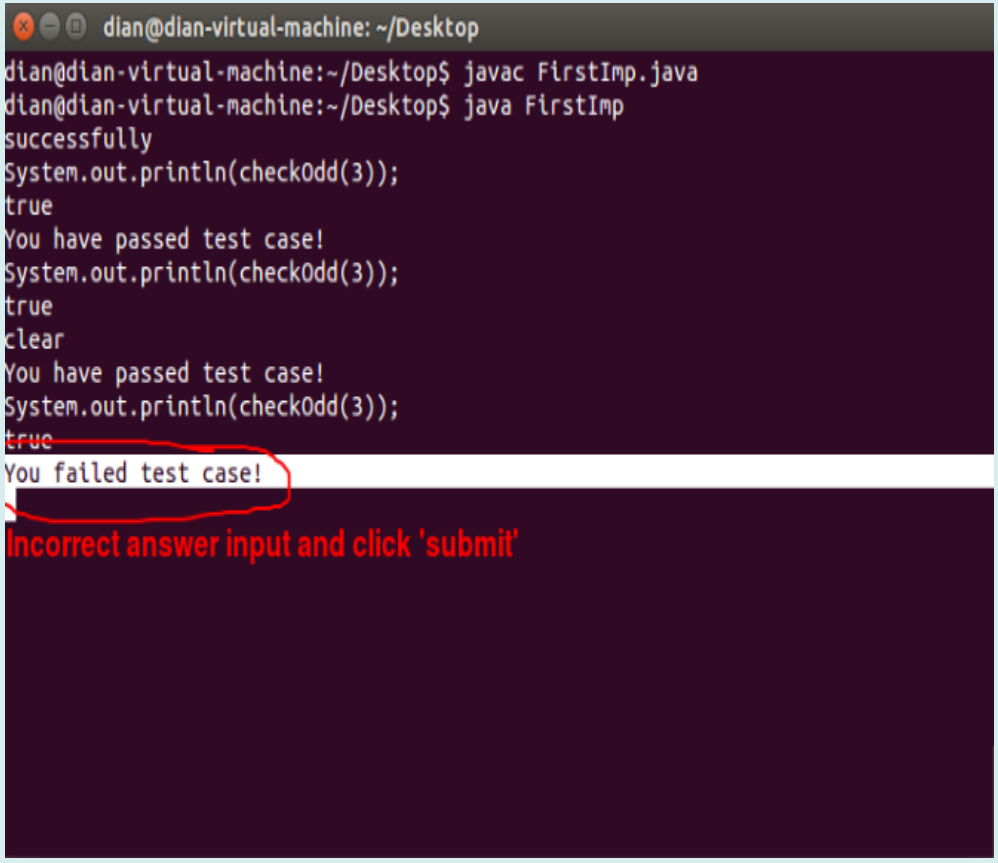
Lecture and student view of the new created question



Prototype Testing

With correct and incorrect answer, prototype is able to run students input as Java file and return the feedback as SandBox does in Code Runner.



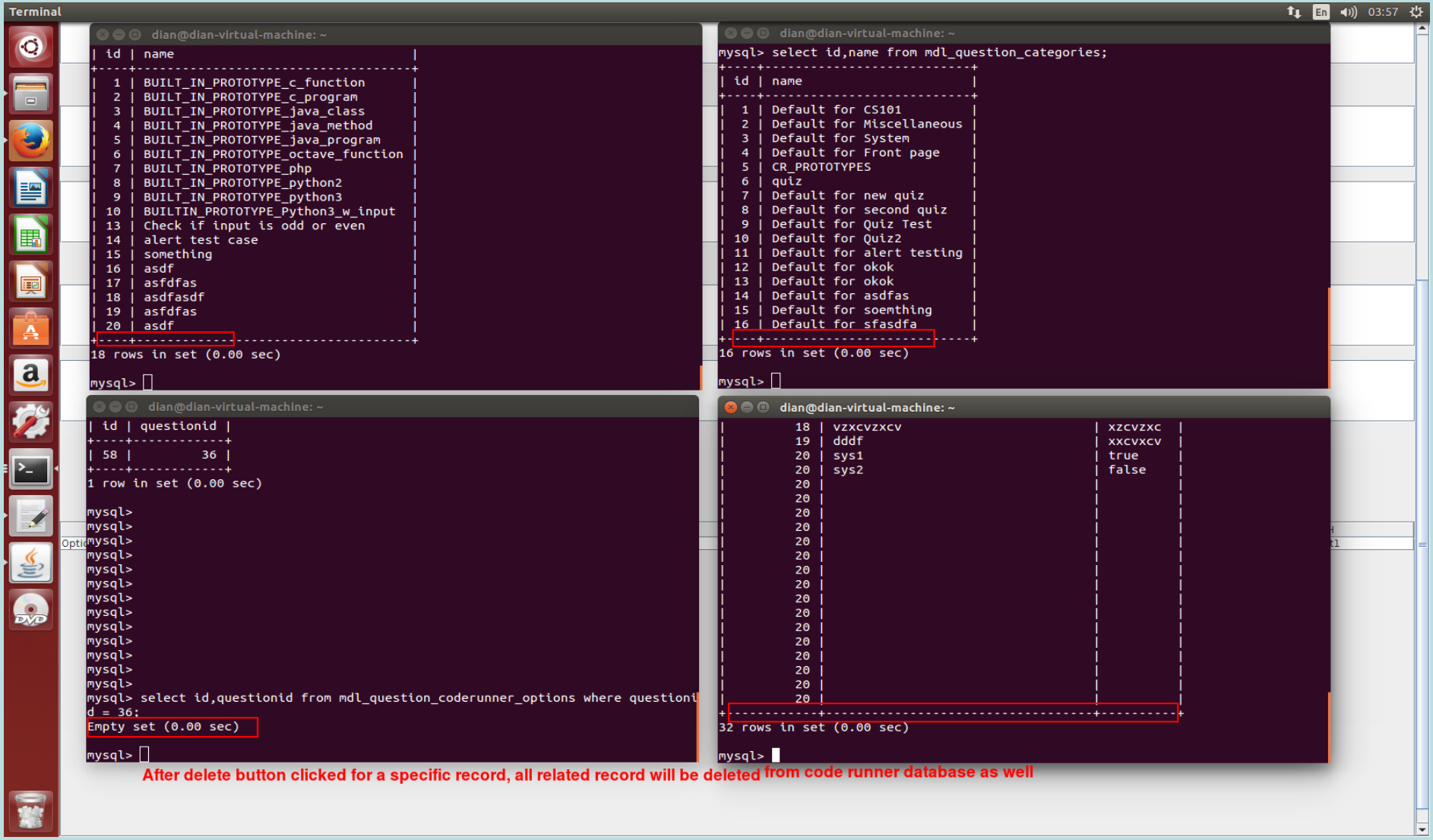


Prototype Testing

- Deletion test:
  - records of the corresponding question will be both deleted from teacher's view and MySQL database
  - Students are no longer to view the deleted question again.

Prototype Testing

The variants 'Odd' that we just created are now deleted from database.



Further work

- So far, proof of concept has been successfully done by using Java programming in my prototype.
- Moving back to Code Runner and implement the concept by PHP programming skills.
- More feasible ideas against cheating need to be generated and implemented into Code Runner.
- Reseach based on particular ideas.

Thank you!

¿Questions?