**Testing Documentation**

**Survey System Anywhere**

**Lizard League Software**

## Testing Plan

The purpose of this document is to design and execute the test plans and verify that the outputs meet the requirements. The testing is divided into two categories: Unit testing and Interface testing.

### Unit Testing

Unit testing is carried by executing the public functions of the classes that carry out the core functionalities. Each function is executed and its output is verified against expected output. The class holding the tests can be found in the source code.

## Class Durvey\_Db

|  |  |  |  |
| --- | --- | --- | --- |
| **Function** | **Scenario** | **Expected Output** | **Actual Output** |
| Check4Db | Surveydatabase.db exists | True | True |
|  | Surveydatabase.db does not exist | False | False |
| createDb |  | Tables (SURVEY, SURVEY\_QUESTION, POSSIBLE\_ANSWERS, ANSWERS) are created | Database file along with necessary tables were created |
| addNewSurvey |  | A record in SURVEY table is created | A record in SURVEY table was created |
| addNewQuestionWithAnswer |  | - A new record in SURVEY\_QUESTION - Records in POSSIBLE\_ANSWERS for the above question | A new record in SURVEY\_QUESTION and several records in POSSIBLE\_ANSWERS were created |
| deleteDB |  | File “surveydatabase.db” is deleted | File “surveydatabase.db” was deleted |
| getSurveyID |  | Survey\_ID of given survey name | Survey ID was returned correctly |
| addNewRespondent |  |  |  |
| addResults |  | Response for the questions should be created in ANSWERS table | Records were created in ANSWERS table for the given question |
| getSurveyNames |  | List of survey names | List of survey names was retrieved |
| getSurveyQuestions |  |  |  |
| getNumQuestions |  | Number of questions of a particular survey | Number of questions of a particular survey was returned correctly |
| getSurveyQuestionsAnswer |  | All questions and their answers for a particular survey | List of questions along with their answer keys and answer text were retrieved |
| checkUniqueSurveyName |  | True if the given survey name is not in the database. Otherwise, False | True when survey name was not in the table SURVEY and False when the survey name was not in the table |
| getAnswerID |  | ID of an answer for a particular question |  |
|  |  |  |  |

## Class Survey\_Actions

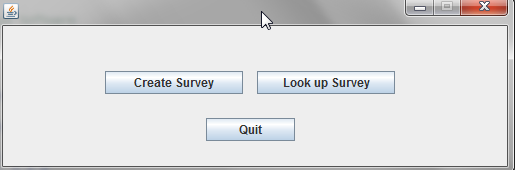
All of the functions in this class call the functions defined in survey\_db and these are just wrapper functions with no new statement execution. The testing method implemented was same as with survey\_db class.

|  |  |  |
| --- | --- | --- |
| **Function** | **Expected Output** | **Actual Output** |
| addNewSurvey | A record in SURVEY table is created | A record in SURVEY table was created |
| addNewQuestionWithAnswer | - A new record in SURVEY\_QUESTION - Records in POSSIBLE\_ANSWERS for the above question | A new record in SURVEY\_QUESTION and several records in POSSIBLE\_ANSWERS were created |
| addResults | Response for the questions should be created in ANSWERS table | Records were created in ANSWERS table for the given question |
| addResults | Response for the questions should be created in ANSWERS table | Records were created in ANSWERS table for the given question |
| getSurveyID | Survey\_ID of given survey name | Survey ID was returned correctly |
| getNumberOfQuestions | Number of questions of a particular survey | Number of questions of a particular survey was returned correctly |
| getSurveyQuestionsAnswers | All questions and their answers for a particular survey | List of questions along with their answer keys and answer text were retrieved |
| checkUniquesurveyName | True if the given survey name is not in the database. Otherwise, False | True when survey name was not in the table SURVEY and False when the survey name was not in the table |
| getSurveyNames | List of survey names | List of survey names was retrieved |
|  |  |  |

### GUI Testing

GUI testing deals with testing whether or not the GUI components react to the user inputs as expected.

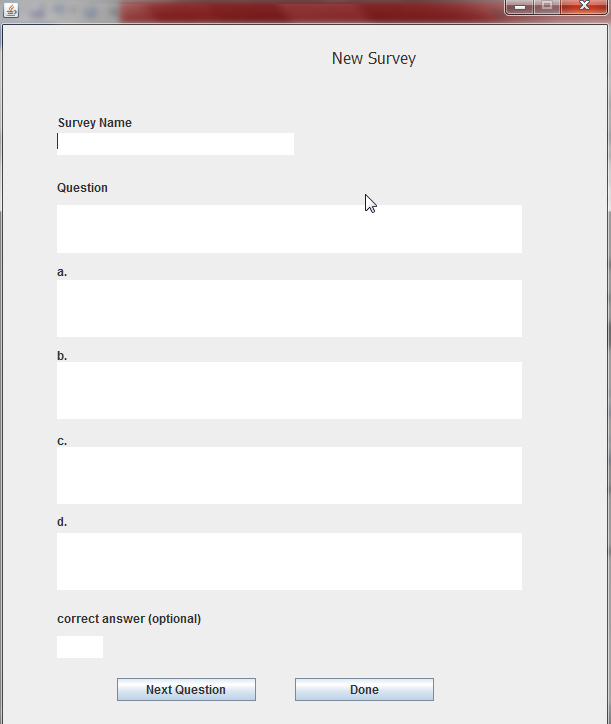
**Test cases for main application window**



1. Create Survey button

When clicked users should be shown a window that allows them to add new survey along with questions and answers.

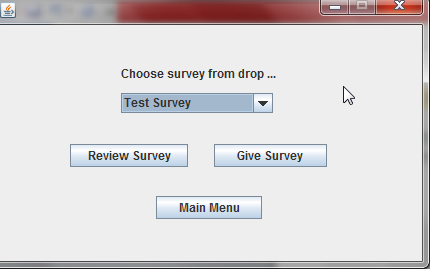
**Output**



1. Lookup Survey button

When clicked, users should be shown a window that allows them to retrieve the surveys in the database either give the survey or review it.

**Output**



1. Quit button

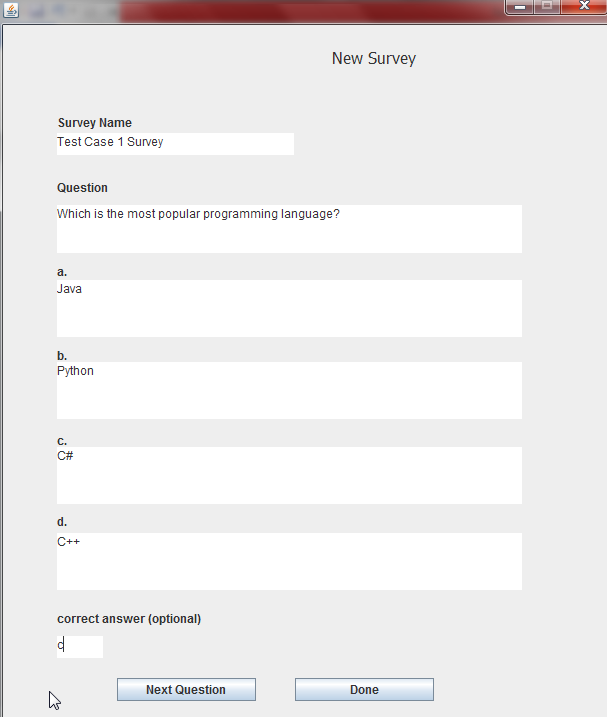
When clicked, the application must close.

**Output**

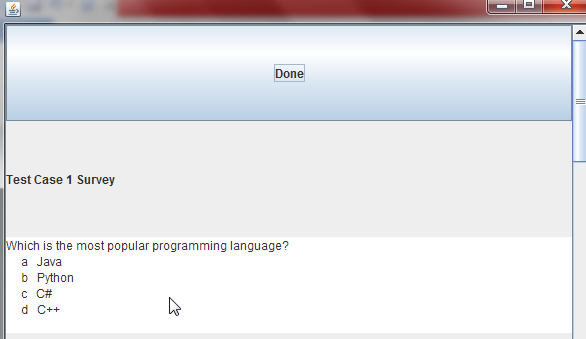
Application terminated.

**Test cases for Create Survey**

“Create Survey” window must allow users to create a new survey and its corresponding questions. When “Next Question” is clicked, the input boxes should be cleared so that users can enter new values. When “Done” is clicked, the survey along with its questions should be added to the database.

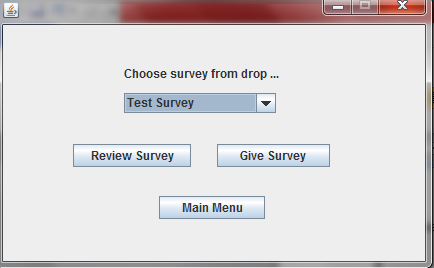


The output can be verified by going to “Lookup Survey”



**Test cases for Lookup Survey**

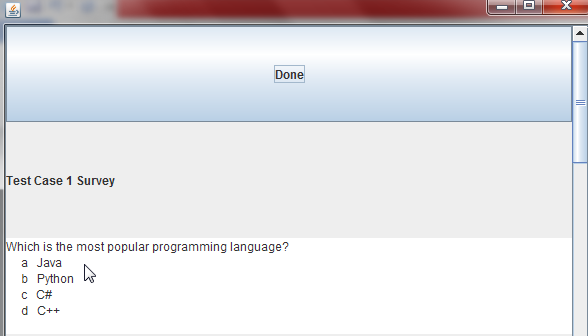
Lookup Survey should allow users to review the surveys in the database, provide an option to give the survey and also go back to the main menu.



1. Test case for Review Survey

Review Survey should display the survey selected from the drop down box and its related questions and associated answers.

**Output**

****

1. Test case for Give Survey

Give Survey should present a user with an interface that allows responding to the survey that they have selected.

