Crowdsourcing Website Template Specifications

I. Summary

This website enables users to complete a survey consisting of a question to obtain the response variable followed by numerous predictor questions. When taking the survey, users have the ability to conveniently add questions that they believe are relevant. Any added questions are emailed to the researcher for approval before they officially appear in the survey. After a question has been added, the researcher can either approve, reject, or modify the submitted question. If the researcher approves or modifies a submitted question, it will appear in the survey for the next participant. Predictor questions are randomly displayed on a webpage one at a time after the response variable question has been answered. After answering all predictor questions, the user is taken to the prediction page that displays the computer's automated prediction along with the actual value of the response variable that the user previously submitted. This website template does not actually calculate an automated prediction, but displays a value on the page in the same way that a true automated prediction would be displayed. After viewing the computer's automated prediction, the user has the ability to add more questions to the survey. Finally, the user is taken to the completion page that thanks the participant for his or her time and provides a link to close the survey.

II. Website Pages

1. Start Page

The first page of the survey is the start page. This page consists of a message telling the user to click the button below to begin the survey and a hyperlink that takes the user to the first

question (response variable question). In the source code, the start page is defined as start.php and the style sheet is defined as start.css.

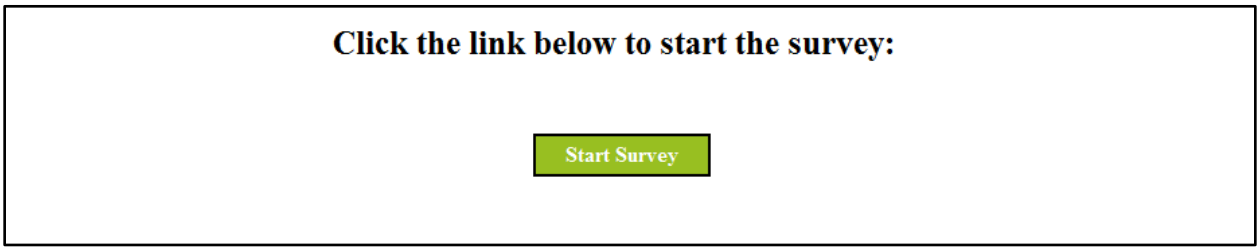
The following code on this page needs to be modified for other crowdsourcing projects:

- **Line 31**

  Start Survey

  Replace "savings.php" with the name of your response variable question page.

The following box shows a screenshot of the start page:



2. Response Variable Question Page

After the user clicks the "Start Survey" hyperlink, he or she is taken to the response variable question page. This page prompts the user to answer the desired response variable question. In this case, the desired response variable is "How much money do you have in your bank account?" After entering a value in the textbox, the program will check to make sure the value is entirely numeric. If the value is not numeric or the value is an empty string, an error message will be displayed when the user clicks the "Next Question" button. In this sample, an error message will also be displayed if the entered value is negative, since you cannot have less than $0 in your bank account. If the entered value is valid after the user clicks the "Next Question" button, the program will add a row to tblUser in the database. The user's

actual response variable will be inserted to the table (fldSavings in this example) and pkUserID will automatically be generated. Before taking the user to the next page, the program will retrieve all approved questions from the database and randomly choose one of these questions to display next. The user will be taken to the first predictor question and the user ID and question ID will be sent through the URL ("user" represents the user ID and "id" represents the question ID). In the source code, the response variable question page is defined as savings.php and the style sheet is defined as savings.css. These page names should be adjusted depending on the desired response variable.

The following code on this page needs to be modified for other crowdsourcing projects:

- **Lines 7 and 8:**

  ```
  $dbUsername = "aberger4";
  $dbUserPass = "aideivim";
  ```

  These variables represent the database's username and password. Replace the values of these variables with your own database's username and password.

- **Line 41**

  ```
  $savings = ""; //personal savings entered by the user
  ```

  This variable holds the response variable entered by the user. Replace the name of this variable to match your desired response variable. Also, replace your newly named variable with all instances of "$savings" throughout the code.

- **Line 52**

  ```
  $savings = $_POST["savings"];
  ```

  This line retrieves the value of the response variable from the textbox. Replace "savings" with whatever you named your textbox.

- **Lines 69-72**

  ```
  else if($savings < 0)
  {
          $errorMessage = "Your total savings cannot be less than $0! Please enter an
          appropriate value!";
  }
  ```

  These lines check to see if the response variable is negative. If your response variable is allowed to be negative, remove this entire section of code.

- **Line 79**

  ```
  mysql_select_db("ABERGER4", $connectID);
  ```

  "ABERGER4" is the name of my database. Replace this value with the name of your database.

- **Line 83**

  ```
  $insertSql = "INSERT INTO tblUser SET fldSavings = '$savings'";
  ```

  Replace "fldSavings" with the name of the field in your database that holds your response variable.

- **Line 114**

  ```
  header("Location:https://www.uvm.edu/~aberger4/CrowdsourcingProject/question.
  php?user=" . $userID . "&id=" . $nextQuestionID);
  ```

  Modify the URL to match the place where you are hosting the website.

- **Line 125**

  ```
  <title>Personal Savings Question</title>
  ```

  Modify the title tag to match your desired response variable.

- **Line 129**

  ```
  <meta name="description" content="Personal Savings Question" />
  ```

  Modify the content tag to match your desired response variable.

- **Line 131**

  <link rel="stylesheet" href="savings.css" type="text/css" media="screen" />

  Modify the stylesheet name “savings.css” to match your desired stylesheet name.

- **Line 165**

  How much money do you have in your bank account?

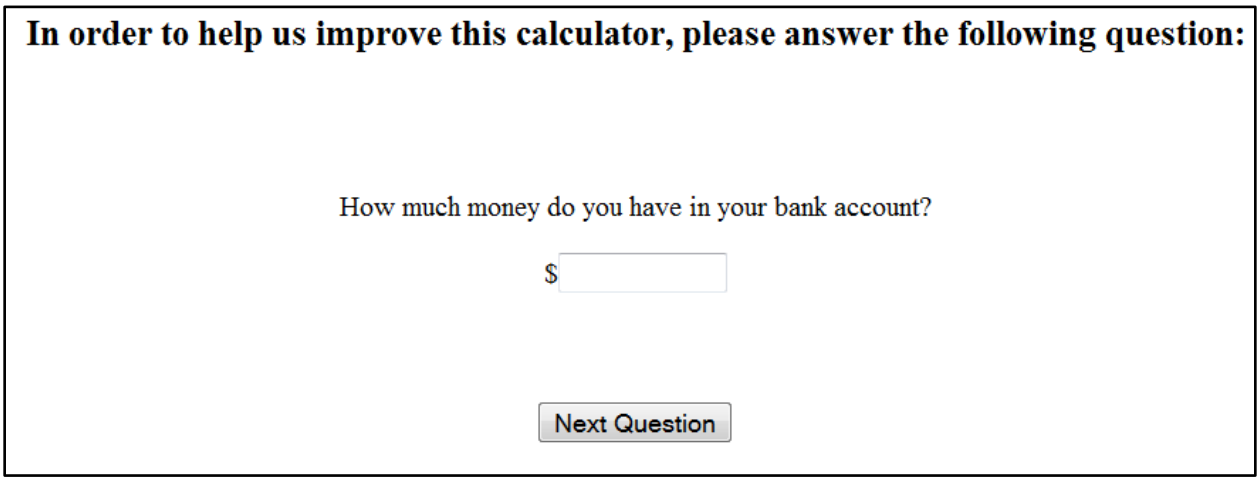
  Replace this text with your desired response variable question.

- **Line 174**

  ** $<input type="text" maxlength="10" size="10" style="font-size: 18px;" value="<?php echo $savings ?>" name="savings" /> **

  Modify the attribute name=”savings” to match your desired response variable. Also, remove the dollar sign ($).

The following box shows a screenshot of the response variable question page:



3. Predictor Question Page

The predictor question page displays the question stored in tblQuestion of the database that corresponds to the “id” variable passed through the URL. The displayed question is followed by a textbox so that the user can type his or her answer. After typing in an answer, the user

can click the “Next” button to go to the next page. If the user does not want to answer the current question, the user can click the “Skip” button. If the user types a value that is not numeric and clicks “Next”, the webpage will display an error message. Otherwise, it will record the user’s response in tblResponse of the database and take the user to the next page. If the user clicks the “Skip” button, a row is still added to tblResponse, but NULL is placed as the value for fldResponse. After recording the user’s response, the program will generate a list of approved questions that the user has not seen yet and randomly select one of these questions to display next. If the user has already seen all of the questions, the program will take the user to the prediction page. Below the “Skip” and “Next” buttons is a large textarea that the user can use to add a question to the survey. If the user adds a question to the survey and clicks the “Add Question to Survey” button, the program will store the question in tblQuestion of the database as unapproved (fldApproved = 0) and send an email to the researcher asking him/her to either approve, reject, or modify the question. In the source code, the predictor question page is defined as question.php and the style sheet is defined as question.css.

The following code on this page needs to be modified for other crowdsourcing projects:

- **Lines 7 and 8**

  ```
  $dbUsername = "aberger4";
  $dbUserPass = "aideivim";
  ```

  These variables represent the database’s username and password. Replace the values of these variables with your own database’s username and password.

- **Lines 132, 179, 243, and 351**

  mysql_select_db("ABERGER4", $connectID);

  "ABERGER4" is the name of my database. Replace this value with the name of your database.

- **Lines 160, 166, 207, 213, 265-267**

  Modify these URLs to match the place where you are hosting the website.

- **Line 260**

  $to = "aberger4@uvm.edu";

  This variable represents the researcher's email address that is used to send an email to the researcher when the user adds a question to the survey. Replace this email address with your own email address.

The following box shows a screenshot of the predictor question page:

4. Prediction Page

The prediction page displays the computer's automated prediction followed by the correct answer that was provided by the user's response to the first question. In this example, the program will always display $10,000 as the computer's automated prediction. After displaying these values, there is a "Continue" hyperlink that takes the user to the completion page. Under this hyperlink is another opportunity for the user to add a question to the survey. In the source code, the prediction page is defined as prediction.php and the style sheet is defined as prediction.css.

The following code on this page needs to be modified for other crowdsourcing projects:

- **Lines 7 and 8**

  ```
  $dbUsername = "aberger4";
  $dbUserPass = "aideivim";
  ```

  These variables represent the database's username and password. Replace the values of these variables with your own database's username and password.

- **Lines 73 and 168**

  ```
  mysql_select_db("ABERGER4", $connectID);
  ```

  "ABERGER4" is the name of my database. Replace this value with the name of your database.

- **Lines 95 -97**

  Modify these URLs to match the place where you are hosting the website.

- **Line 90**

  $to = "aberger4@uvm.edu";

  This variable represents the researcher's email address that is used to send an email to the researcher when the user adds a question to the survey. Replace this email address with your own email address.

- **Line 120**

  <title>Personal Savings Prediction</title>

  Modify the title tag to match your desired response variable.

- **Line 155**

  <h1>Based on your responses to the previous questions, the computer will predict your personal savings:</h1>

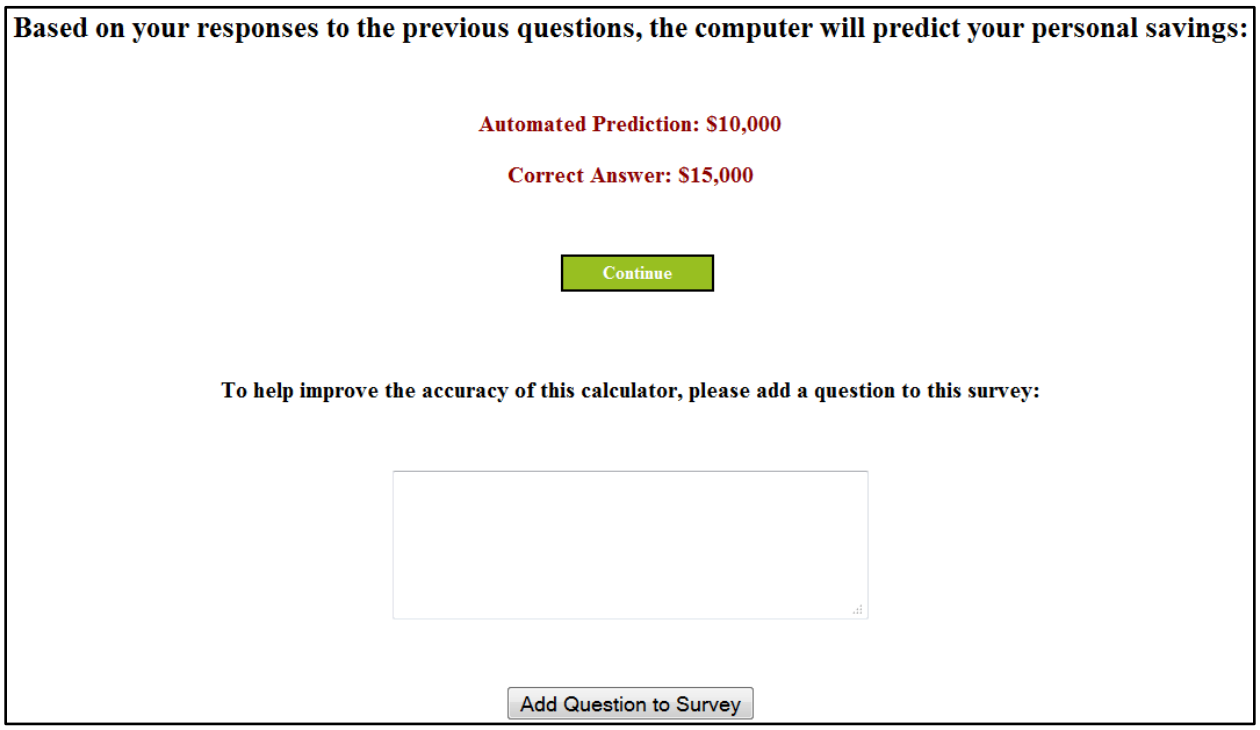
  Modify this message to match your desired response variable.

- **Line 172 – 188**

```
$savingsSql = "SELECT fldSavings FROM tblUser WHERE pkUserID =
'$userID'";
$savingsData = mysql_query($savingsSql, $connectID);
$savings = ""; //the user's savings
while ($row = mysql_fetch_row($savingsData))
{
        $savings = $row[0];
}
echo "<p class='message'>Automated Prediction: $" . number_format(10000) .
"<br /></p>";
echo "<p class='message'>Correct Answer: $" . number_format($savings) . "</p>";
```

  Modify the following variable and attribute names: $savingsSql, fldSavings, $savingsData, and $savings to match your desired response variable.

The following box shows a screenshot of the predictor question page:

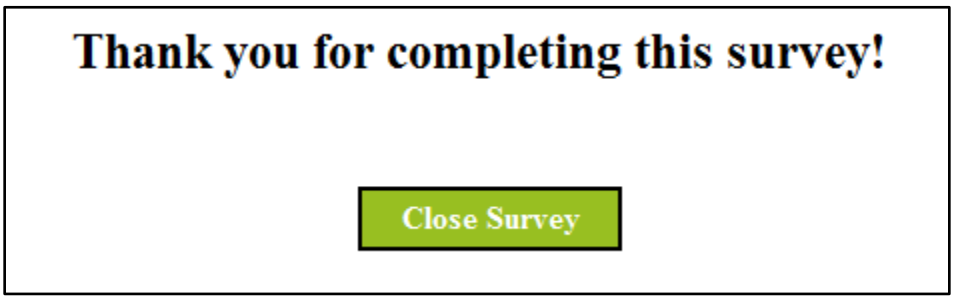


5. Completion Page

The completion page displays a message thanking the user for completing the survey followed by a link to close the survey. When the user clicks the link, the window will close. In Firefox, the link will only work if the window was opened previously by the browser. In Internet Explorer, this link will always work. In the source code, the completion page is defined as completion.php and the style sheet is defined as completion.css.

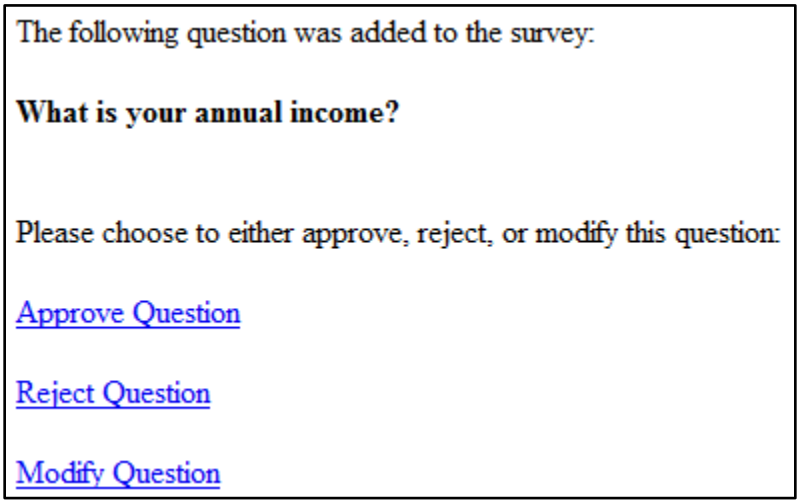
No code on this page needs to be modified for other crowdsourcing projects

The following box shows a screenshot of the completion page:



6. Question Approval Page

When the user adds a question to the survey, an email is sent to the researcher. The email will look like this:



If the researcher clicks the "Approve Question" hyperlink, the question becomes approved in the database (fldApproved is switched to 1). If the researcher clicks the "Reject Question" hyperlink, the question is removed entirely from the database. If the researcher clicks the "Modify Question" hyperlink, the researcher can modify the text of the question before it is approved. After each of these hyperlinks is clicked, a window will open that displays the appropriate content for each command. For the "Approve Question" hyperlink, the page will just display a message saying that the question has been added to the survey. For the "Reject

Question" hyperlink, the page will just display a message saying that the question has been removed from the database. For the "Modify Question" hyperlink, the page will display a textarea containing the user's question that the researcher can use to modify its text and a button to submit the question. All three of these scenarios are controlled by the same page of source code (question approval page). The question approval page is defined as questionApproval.php and the style sheet is defined as questionApproval.css. When one of the approval hyperlinks is clicked, the following two variables will be sent through the URL: the new question's ID (variable name is "id") and the approval decision (variable name is "choice"). The approval decision variable indicates whether the researcher would like to approve, reject, or modify the submitted question.

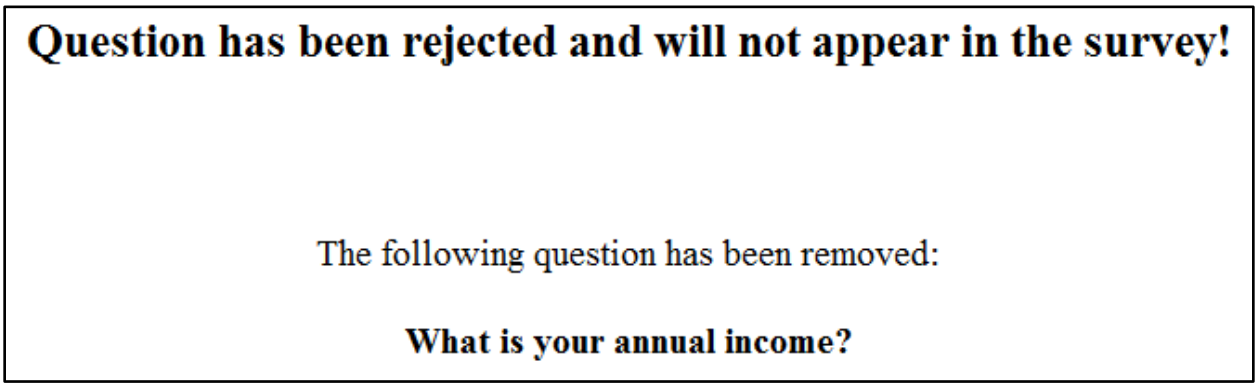
The following box shows a screenshot of the question approval page when the user clicks the "Approve Question" hyperlink:

Question has been approved and will now appear in the survey!

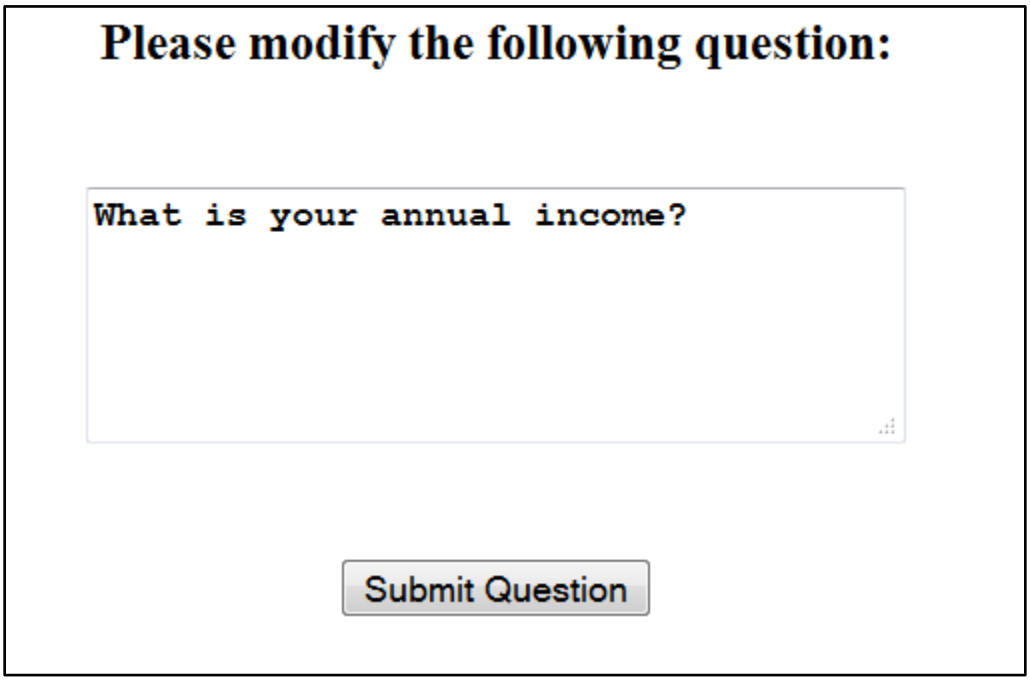
The following question has been added:

What is your annual income?

The following box shows a screenshot of the question approval page when the user clicks the "Reject Question" hyperlink:



The following box shows a screenshot of the question approval page when the user clicks the "Modify Question" hyperlink:



The following code on this page needs to be modified for other crowdsourcing projects:

- **Lines 7 and 8**

  $dbUsername = "aberger4";
  $dbUserPass = "aideivim";

  These variables represent the database's username and password. Replace the values of these variables with your own database's username and password.

- **Lines 49, 93, 118, and 143**

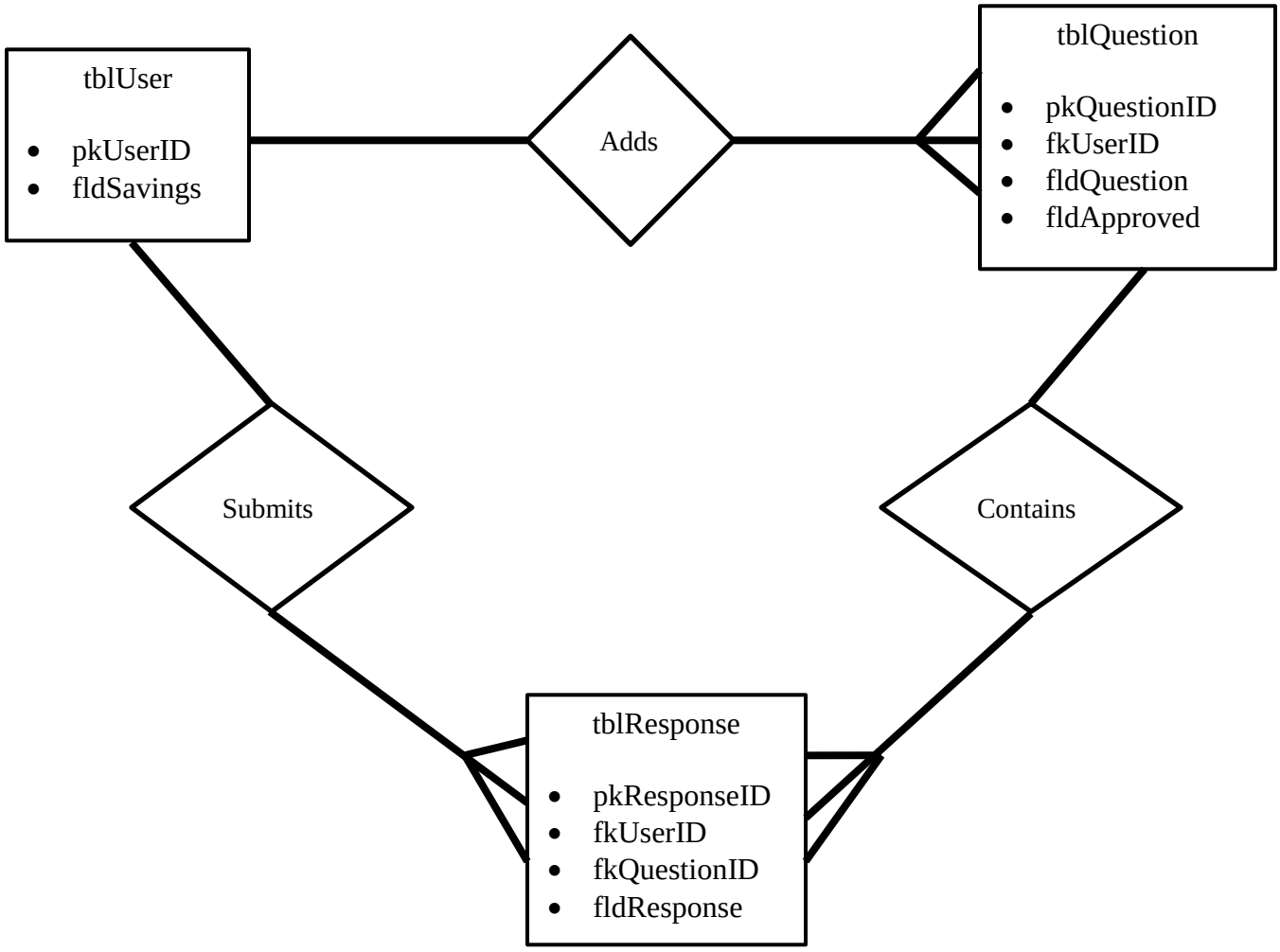
  mysql_select_db("ABERGER4", $connectID);

  "ABERGER4" is the name of my database. Replace this value with the name of your database.

III. Database Structure

In order for the program to work properly, the database must be set up identical to how it is described in this document. The database will need to consist of 3 tables: User table, Question table, and Response table. The Response table is a relationship table between the User table and Question table, which represent a many-to-many relationship.

ER Diagram



1. **User Table**

   The User table is represented in the database as tblUser and contains 2 attributes: pkUserID and fldSavings. pkUserID is the primary key of the table. It is an auto-incrementing integer (surrogate key) whose value is defined after a row has been added. fldSavings is an integer that represents the user's answer to the response variable question. In this case, fldSavings represents the answer to the question "How much money do you have in your bank account?" In another crowdsourcing project, the name of this attribute should be changed to match the desired response variable, though the attribute will still serve the same purpose. The researcher should manually add a row to this table to start. pkUserID can be given a value of -1 and fldSavings should be given a value of NULL.

2. **Question Table**

   The Question table is represented in the database as tblQuestion and contains 4 attributes: pkQuestionID, fkUserID, fldQuestion, and fldApproved. pkQuestionID is the primary key of the table. It is an auto-incrementing integer (surrogate key) whose value is defined after a row has been added. fkUserID is a foreign key and represents the user ID of the participant who added the question to the survey. fldQuestion is a text field (varchar) and represents the actual text of the question. fldApproved is a Boolean variable (tinyint) and indicates whether a question has been approved by the researcher. fldApproved can either have a value of 0 or 1. A value of 0 indicates that the question has not been approved while a value of 1 indicates that the question has been approved. The researcher should manually add the first question to the survey. pkQuestionID can be given a value of -1, fkUserID can be given a value of -1, fldQuestion should be the text of the first question, and fldApproved should be given a value of 1.

3. Response Table

The Response table is represented in the database as tblResponse and contains 4 attributes: pkResponseID, fkUserID, fkQuestionID, and fldResponse. pkResponseID is the primary key of the table. It is an auto-incrementing integer (surrogate key) whose value is defined after a row has been added. fkUserID is a foreign key and represents the user ID of the participant who answered the question. fkQuestionID is another foreign key and represents the question ID of the question that was answered. fldResponse is an integer and represents the actual user response to the current question. This attribute (fldResponse) accepts NULL values. A NULL value would indicate that the user skipped the question.