**Survey Engine**

Benjamin Kwon

CSC-17B-44085

June 3, 2019

**Table of Contents**

1. Introduction..........................................................................................................1

2. Description............................................................................................................1

3. Folder Layout........................................................................................................6

4. Concepts...............................................................................................................7

5. Diagrams.............................................................................................................13

6. Code/GitHub Link................................................................................................14

**1. Introduction**

The purpose of this project was to create a functional website containing a utilizable survey engine. The website also contains separate user and admin interfaces. HTML, CSS, JavaScript, PHP, and MySQL queries were used to create the front and back end of the website.

**2. Description**

**Survey Engine**

This project contains a survey engine in which surveys can be created and then sent for other users to take.

Only users that are registered as admins can create surveys. Once an admin has created a survey, a button will appear, allowing them to copy the unique survey link to their clipboard. Any user who receives the link can take the survey after registering and logging in to their account.

The addition or removal of questions and answers for the survey were created to be dynamic. At any point of time when creating the survey, the user can do any the following: add questions, delete questions, add answers to a question, delete answers to a question. The minimum number of questions required is one and each question must have at least two answers.

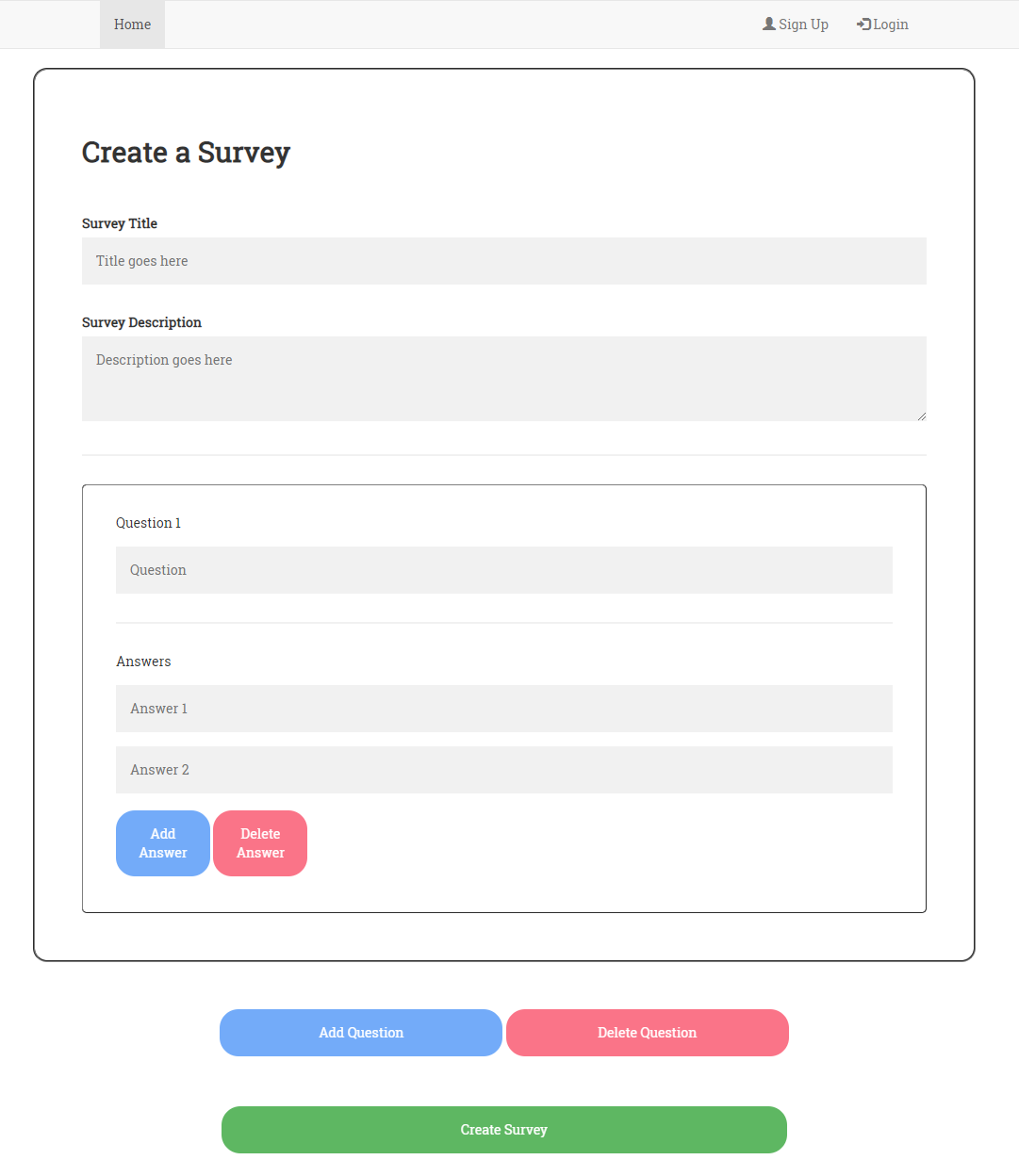
Functionalities such as access to survey results, copying a link to the survey, and the ability to delete the survey can be found in the admin panel. Admins also can modify and delete users.

The following shows a list of accounts that can be used that have already been stored into the database schema.

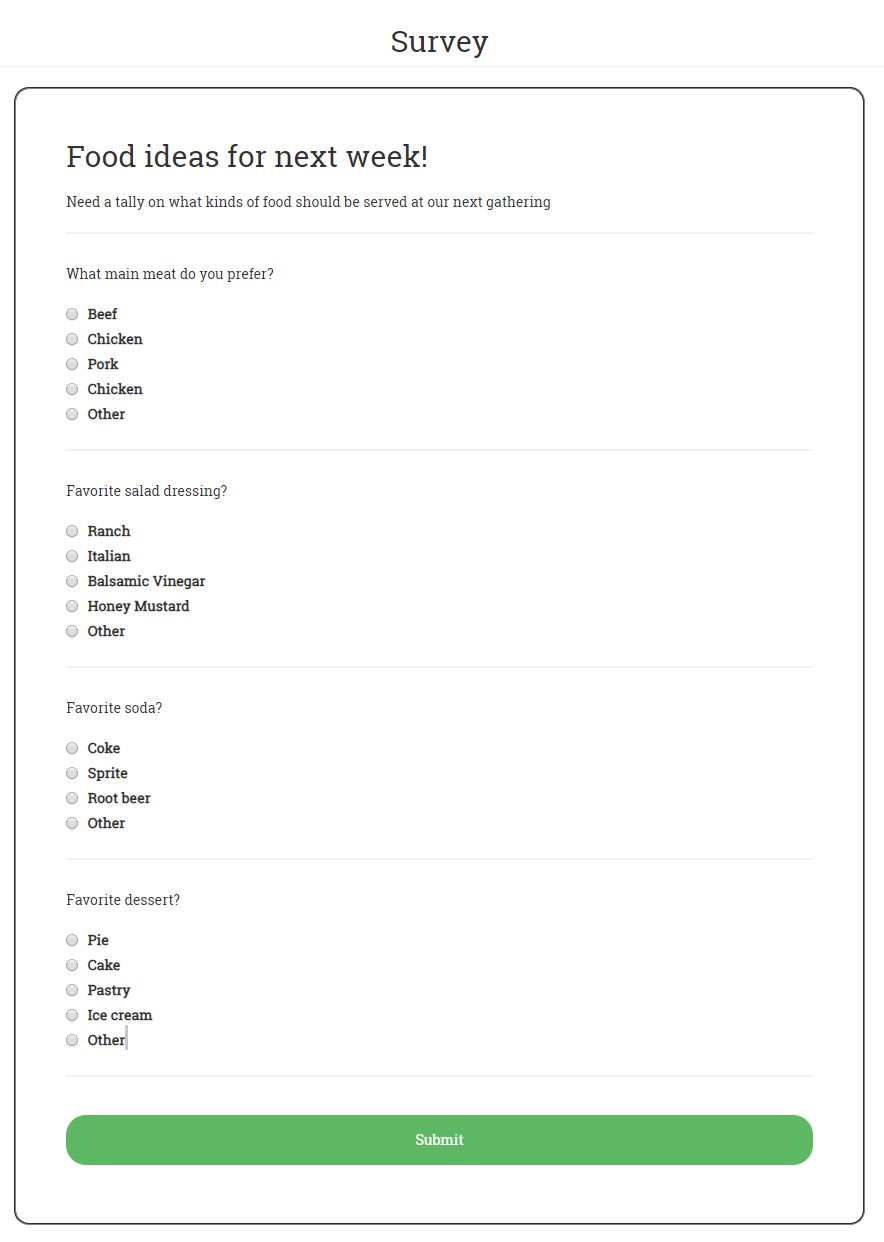
*Table 1. Pre-stored accounts with their email and password*

|  |  |  |
| --- | --- | --- |
| **Username** | **Password** | **Role** |
| admin | admin | Admin |
| user1 | Pass123! | User |
| user2 | Pass123! | User |
| user3 | Pass123! | User |
| user4 | Pass123! | User |
| user5 | Pass123! | User |
| auser1 | Pass123! | User |

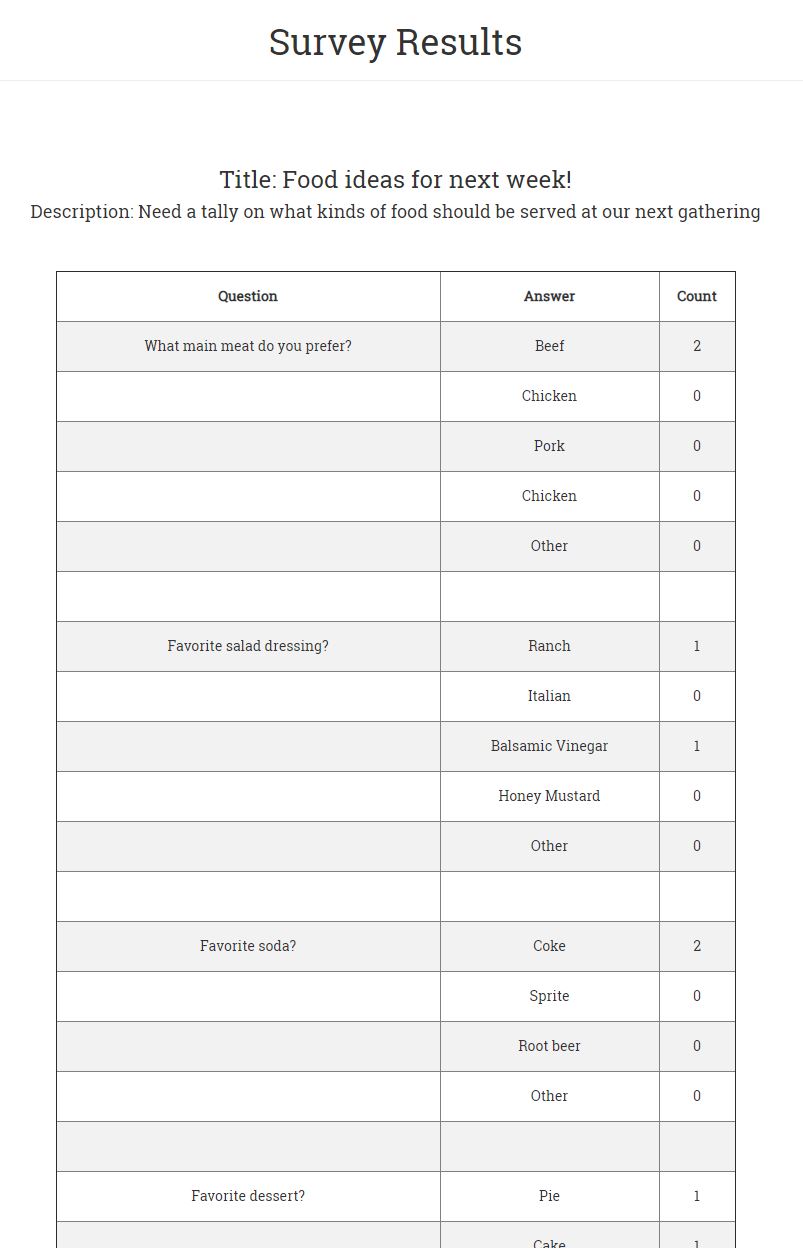
In addition, two surveys have been created by the admin, in which one has results available (two users have already taken the survey).



*A view of the survey creation page*

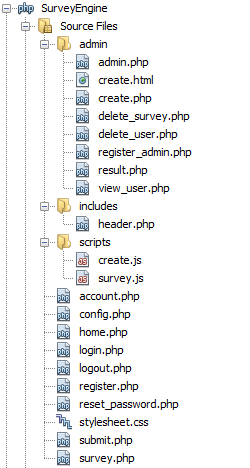
****

*Example of a created survey from the user’s view*

****

*Example of a results from a survey from the admin’s view*

**3. Folder Layout**

****

Pages that are specific to admins are in a separate folder titled ‘admin.’ The folder ‘includes’ contains the header for pages. The folder ‘scripts’ contains JavaScript scripts used in some of the pages. The rest of the files are webpages that can be accessed by any user.

**4. Concepts**

**Model-View-Controller**

*Model:*

The ‘survey’ database contains tables representing data such as survey information, questions to a survey, and answers to a question. User information such as usernames and passwords are also stored. After receiving a command, the model updates, which can then be displayed by the view.

*View:*

The view allows the visualization of the model state. Through HTML, CSS, and JavaScript, the user can see the data displayed from the model in a presentable way. For example, admins can see a maintained list of survey results or user information from the database.

*Controller:*

By reading user inputs such as keyboard inputs, PHP can take the data from methods such as HTTP GET and POST requests. The data can then be manipulated and/or stored into the model through commands. Thus, the controller manages any user interaction with the model. After a survey is created, the inputs would then be stored into the database

**Objects**

File location: /SurveyEngine/scripts/create.js

Using the recently implemented JavaScript class syntax, a Question class and Survey class were constructed. The question class includes variables that store the question string, the number of answers to the question, and an array of the answers. The survey class contains variables that store the survey title, the description, the number of questions, and an array that holds each question object. The information is passed from a JavaScript object to a PHP object by using cookies.

**Reading/Writing files/JSON**

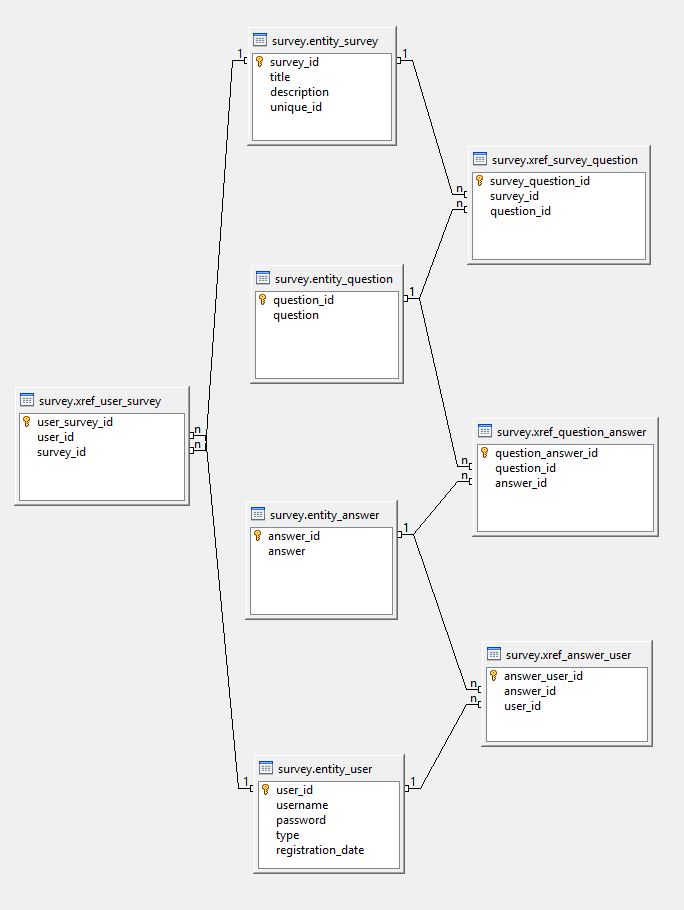
File location: /SurveyEngine/scripts/create.js (lines 161-164)

/SurveyEngine/admin/create.php (line 29)

When a survey is created and the “Create Survey” button is clicked, a function is called that will store the text written by the user into an object of the Survey class. JSON is then used to stringify the object. This string is then stored to a cookie, which will later be read into a server with a PHP script.

**Databases SQL**

Diagram of database tables (entities, xrefs, and enums):



**Form Validation – Regular Expressions**

File location: /Survey/register.php (lines 10-23)

Regular expressions were used for input validation when a user registers an account on the website. Details are as follows:

Username – 3 to 15 alphanumeric characters only

Password – 6 to 20 characters that includes at least one character of each of the following: digit, lower case character, upper case character, symbol

**User-Admin-Login**

File location: /SurveyEngine/admin/register\_admin.php (Register admin account)

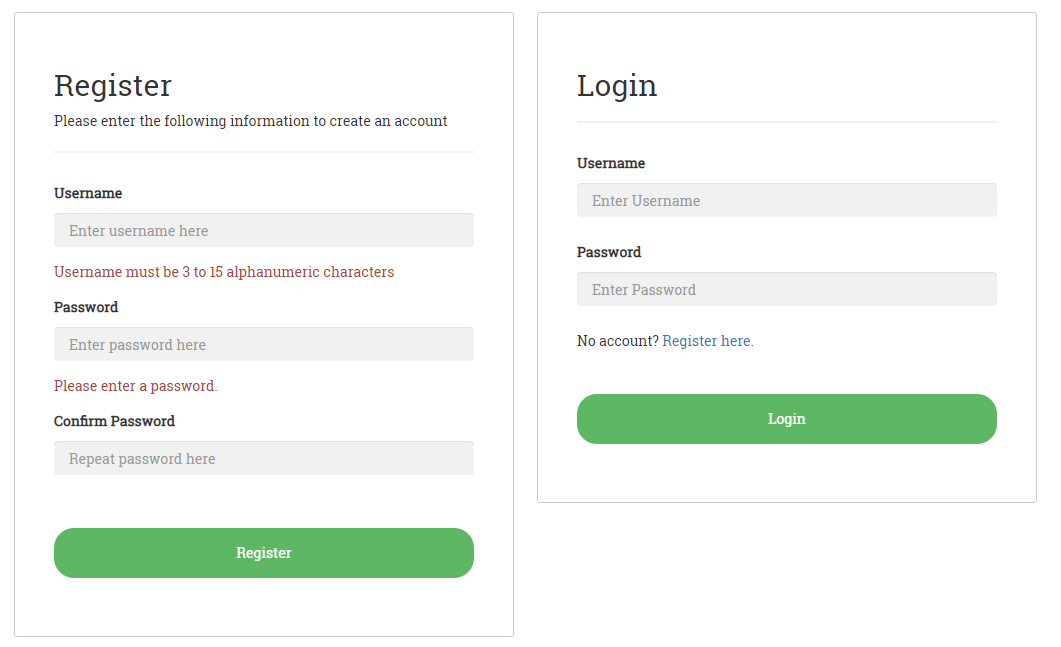
/SurveyEngine/register.php (Register user account)

/SurveyEngine/admin/admin.php (Admin panel)

/SurveyEngine/account.php (User panel)

/SurveyEngine/login.php (Login page)

When no user is logged in, a button for registering an account as well as logging in is displayed on the top navigation bar. When logged in, the register and login buttons disappear and instead show a button that directs the user to their main account page. Here, users can view surveys they have taken and reset their password. On the other hand, admins can view the admin panel which grants access to the creation of surveys and the ability to view and modify any user account.



*A view of the login and register pages*

The `entity\_user` table inside the survey database contains both users and admins. The difference between the two is the type/role column, which signals whether the user is an admin or not. Admins have a wider functionality access such as the ability to directly manipulate database items as well as user information. On the other hand, users can only interact with the allowed elements.

*Table 2. Roles of the two different types of users*

|  |  |
| --- | --- |
| **Admin** | **User** |
| Create surveys | Take surveys (one submit per survey) |
| View results of surveys |  |
| View and modify list of users |  |

**Cookies-Sessions-Securing Pages**

File location: Cookies: /SurveyEngine/create.js & /SurveyEngine/survey.php

Sessions: Most files in project

Securing Pages: Most files in project

In this project, cookies are used to store the survey object after it has been converted to a JSON string. This cookie is then passed to a PHP object after decoding. This information is taken and stored in the database. Thus, the survey, survey questions, answers to the questions, and individual user answers can all be linked through the database when queried.

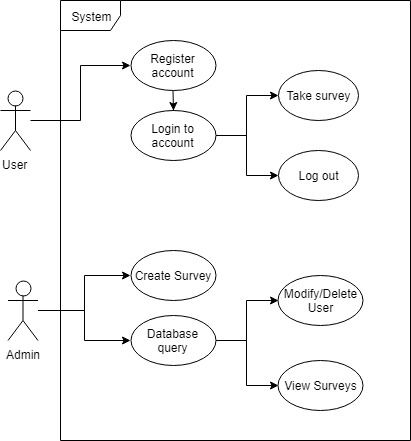
Sessions are used on many pages to hold information for easy tracking such as the user’s identification number.

All pages are secured using sessions as well as valid GET or POST parameters. Depending on the page, users will be redirected to another more appropriate page or be denied access. For example, if a user tries to get into pages only intended for admins, they will be denied access and the script will terminate.

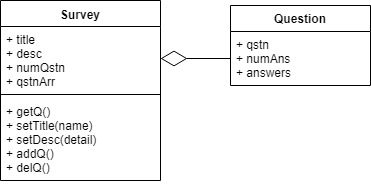
Pages that only grant access to admins are secured by using a session that tracks whether the user has admin privileges. Most of these session variables are set when a user logs in to the system.

**4. Diagrams**

UML Use Case Diagram



UML Class Diagram



**6. Code**

The source code for each file of this project can be found at the following GitHub link: <https://github.com/bhknx3/CSC-17B/tree/master/SurveyEngine>

OR

The code can be viewed in the file labeled SurveyEngine\_Code.docx.