Josue Carlito
CSC17B
4/23/2023

SURVEY ENGINE

Introduction:
Title: Survey Engine

My survey engine program emulates a simple survey creator. It allows a user to take a survey, review their response and submit. Then an admin can go and view the survey data. The data is displayed using "#" to create a type of horizontal bar graph. The admin is also capable of viewing user sign in info, creating surveys, and viewing user responses.

Summary:
Lines of code: ~1200
Variables: ~50
Classes: 4

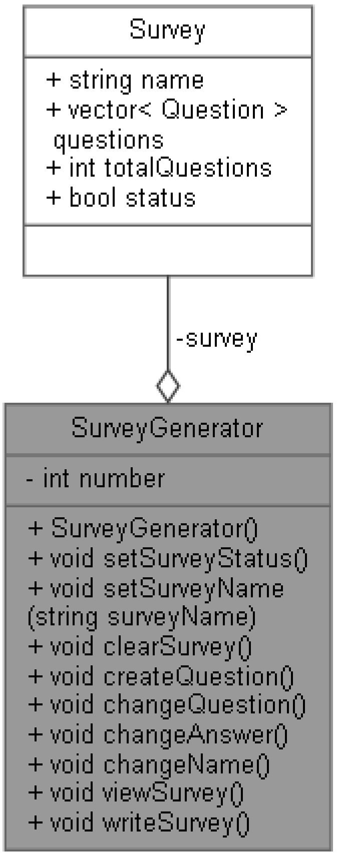
The program is implemented with an OOP design and uses binary files to store information. The normal user can simply take a survey and confirm their response. The admin is capable of creating surveys, viewing the survey data in its entirety as well as specific responses and also capable of editing user sign in info.

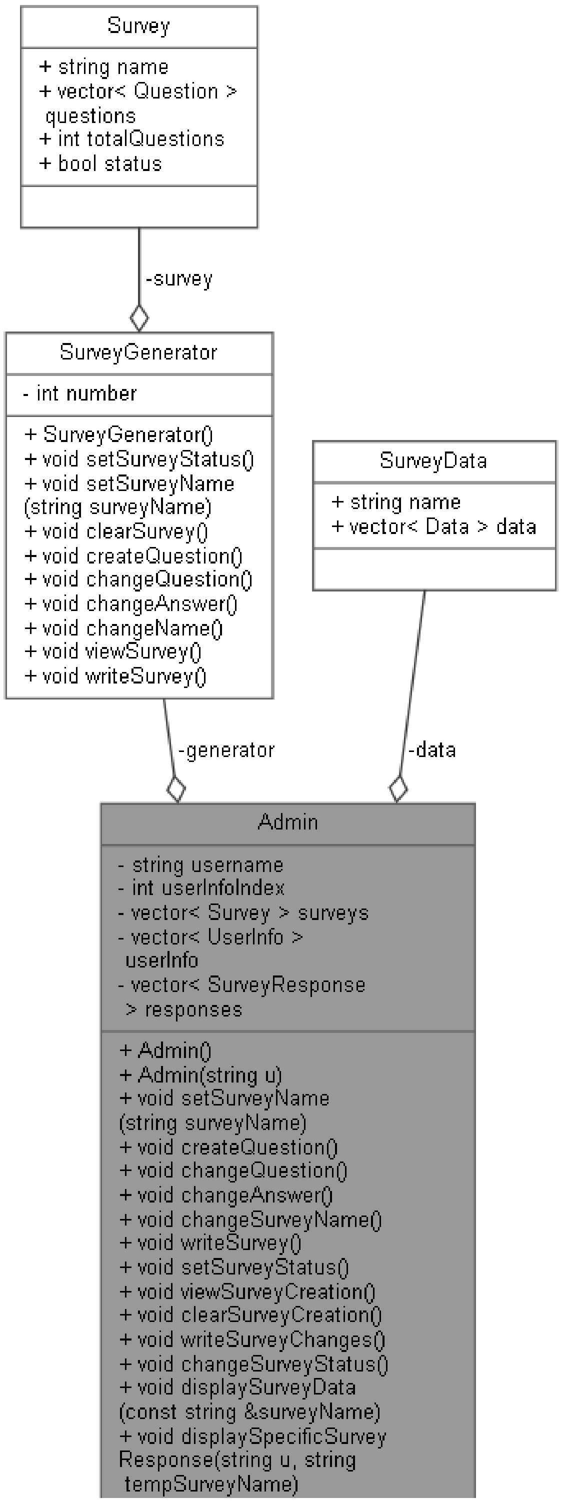
This program in total took me around ~18 hours to code. I coded this after the storefront project so I went into straight OOP design. This resulted in fewer versions. I used the same sign-in class from the storefront program. The most difficult part when programming this project would have to be working with binary files and being able to update them and store information. Another challenge was displaying the survey data and gathering all that information from the files. I look forward to improving this portion of my project in the web version. Overall, I had fun coding this project and working through the UI. Since I did it after the storefront it was fun to continue practicing creating dynamic projects.

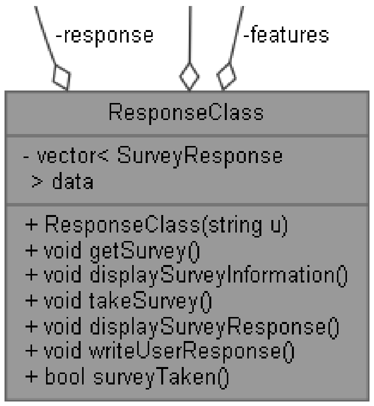
Description:
The program is prompt heavy and requires the user to read each line carefully before they enter an input. The program begins with a sign in page where the user is capable of creating a normal user account, an admin account, or if they have already registered for an account then they can sign in. *To create an admin account the user must enter an admin key which is "1234"* I have already created an admin account with the username and password being "mlehr". If a user signs in as a normal user they are the surveys and how many questions each survey has. Then they type in the survey name exactly and take and submit the survey. If a user signs in as an admin, they have the option to choose from viewing previous surveys and creating a survey, survey data, and user information options (user sign in info). The previous surveys and creating a survey page allows the admin to to create a survey, review their creation and then save it. It also allows them to view all the active surveys and review their questions. If they do not like an old survey then they can make it inactive. The survey data page allows an admin to view an overview of how many surveys users have taken, view all survey responses in their entirety, view a all of a specific user's responses, or view their response to a specific survey and final view the surveys overall data presented in a horizontal bar graph using "#"s. Finally, the user information options allow you to edit sign in information. You can view all the user's sign in info, change the user's password or account status, and then write the changes. Those are the complete admin functionalities.

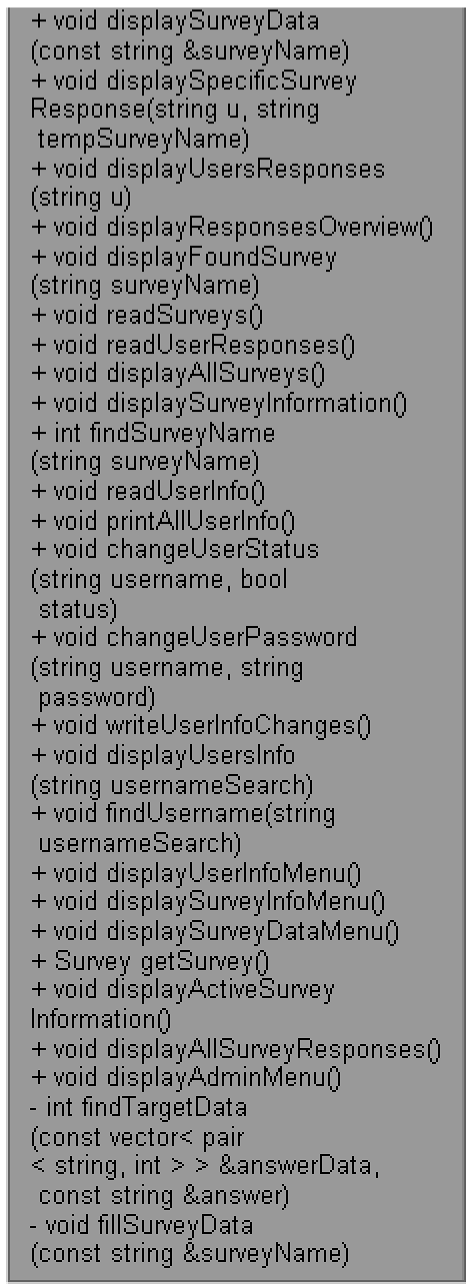
Documentation:

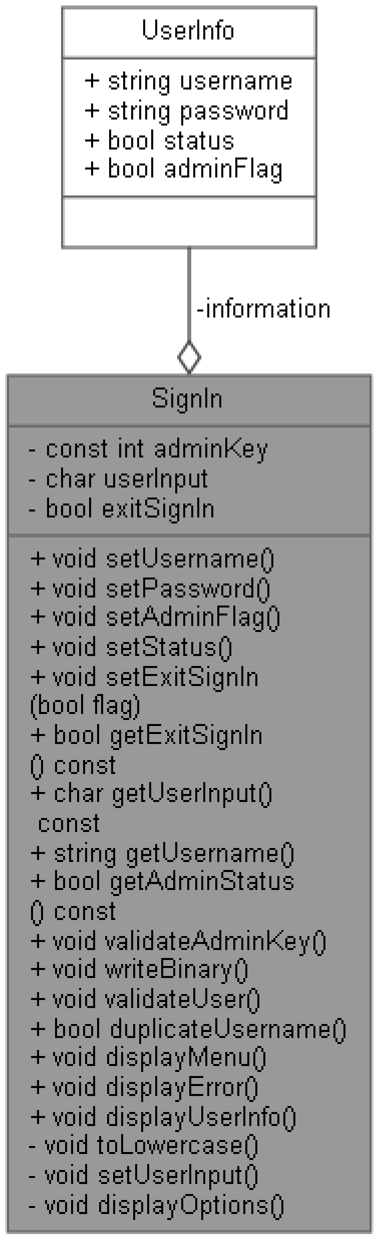
Doxygen was used on version 2 (the final version). There is a doxygen folder in the repository. Click on the index.html file to be brought to the homepage and then navigate throughout the html pages to view the project's documentation.






-response
-features
ResponseClass
- vector< SurveyResponse > data
+ ResponseClass(string u)
+ void getSurvey()
+ void displaySurveyInformation()
+ void takeSurvey()
+ void displaySurveyResponse()
+ void writeUserResponse()
+ bool surveyTaken()


+ void displaySurveyData
(const string &surveyName)
+ void displaySpecificSurvey
Response(string u, string
tempSurveyName)
+ void displayUsersResponses
(string u)
+ void displayResponsesOverview()
+ void displayFoundSurvey
(string surveyName)
+ void readSurveys()
+ void readUserResponses()
+ void displayAllSurveys()
+ void displaySurveyInformation()
+ int findSurveyName
(string surveyName)
+ void readUserInfo()
+ void printAllUserInfo()
+ void changeUserStatus
(string username, bool
status)
+ void changeUserPassword
(string username, string
password)
+ void writeUserInfoChanges()
+ void displayUsersInfo
(string usernameSearch)
+ void findUsername(string
usernameSearch)
+ void displayUserInfoMenu()
+ void displaySurveyInfoMenu()
+ void displaySurveyDataMenu()
+ Survey getSurvey()
+ void displayActiveSurvey
Information()
+ void displayAllSurveyResponses()
+ void displayAdminMenu()
- int findTargetData
(const vector< pair
< string, int > > &answerData,
const string &answer)
- void fillSurveyData
(const string &surveyName)


UserInfo
+ string username
+ string password
+ bool status
+ bool adminFlag
-information
SignIn
- const int adminKey
- char userInput
- bool exitSignIn
+ void setUsername()
+ void setPassword()
+ void setAdminFlag()
+ void setStatus()
+ void setExitSignIn
(bool flag)
+ bool getExitSignIn
() const
+ char getUserInput()
const
+ string getUsername()
+ bool getAdminStatus
() const
+ void validateAdminKey()
+ void writeBinary()
+ void validateUser()
+ bool duplicateUsername()
+ void displayMenu()
+ void displayError()
+ void displayUserInfo()
- void toLowercase()
- void setUserInput()
- void displayOptions()