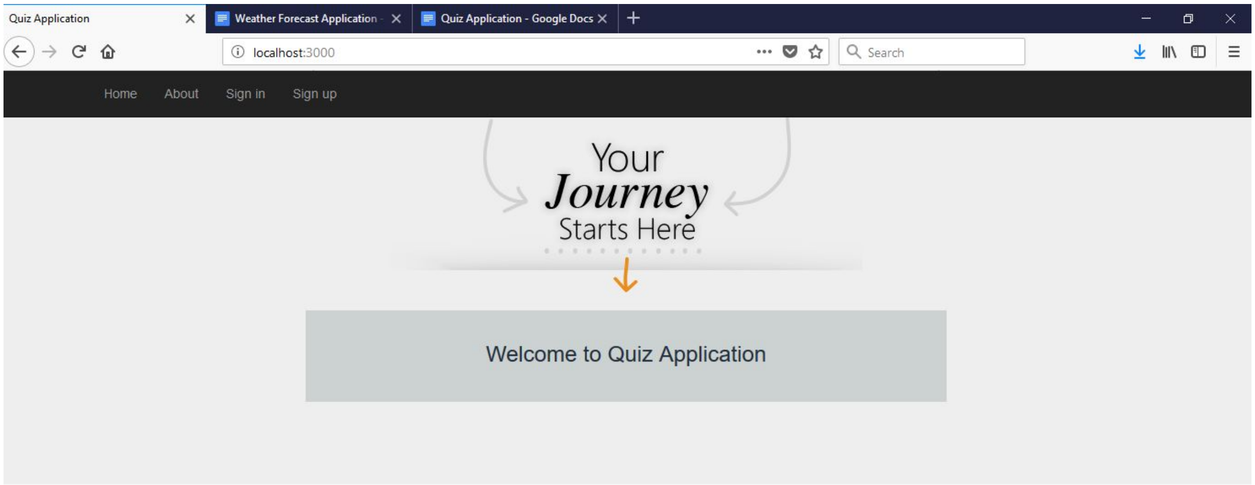
Anagha Kelkar

TestMax

Quiz Application



Welcome to Quiz Application

<u>Software used:</u>

Rails version: *Rails 5.1.5*

Ruby version: *ruby 2.3.3*

Git version: *git 1.9.4*

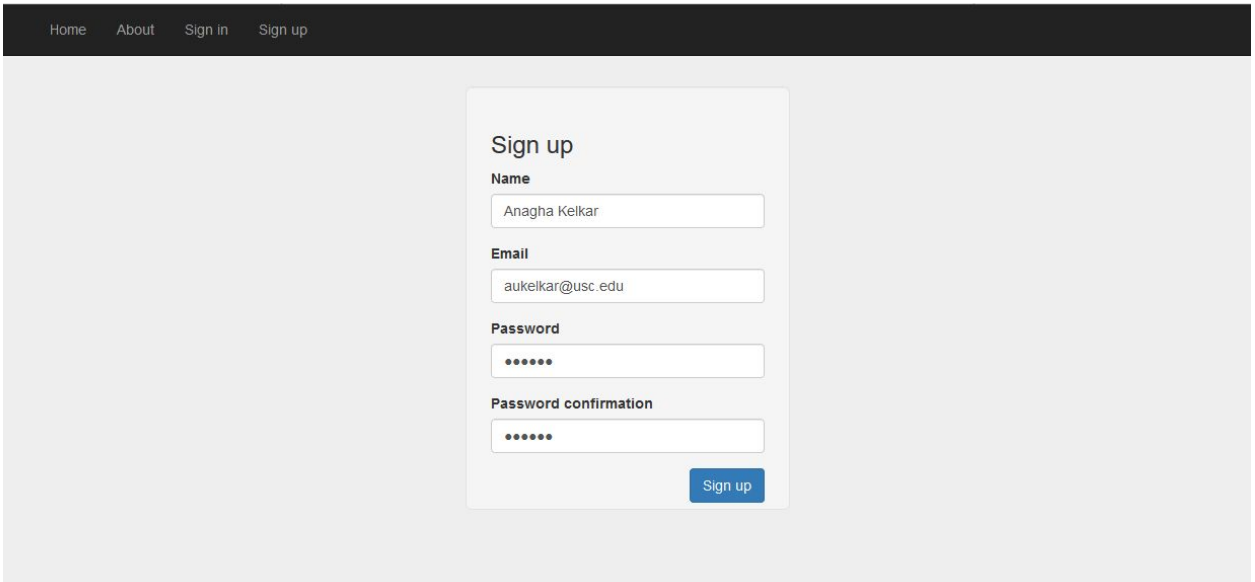
SQLite version: *3.8.7.2*

<u>Note:</u>

Steps to run application successfully, perform steps mentioned in README file which is available in project zip as well as on github.

Features:

1. First step is to create an account to access Quiz.



Sign up

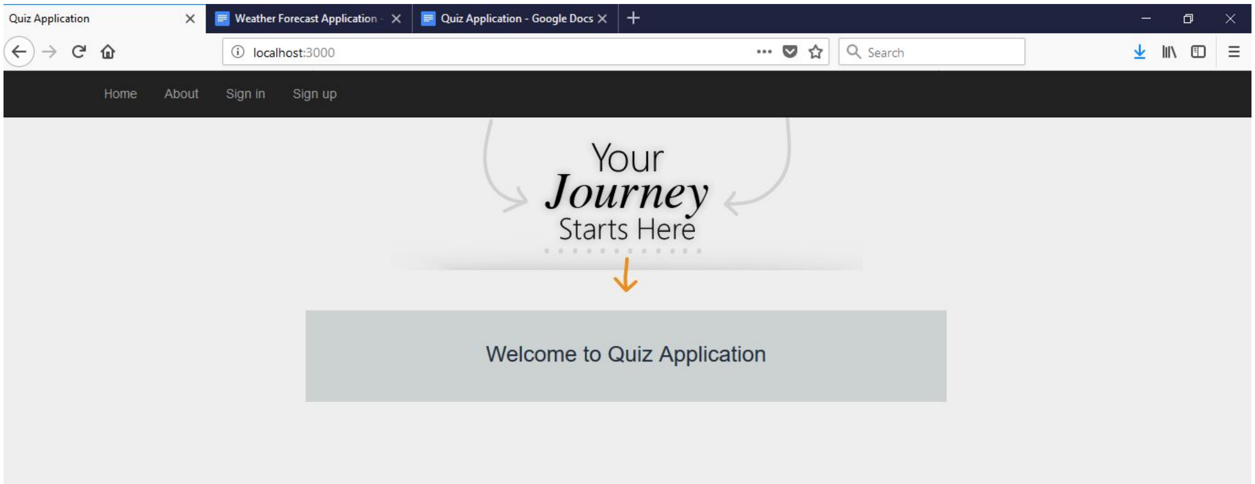
Name: Anagha Kelkar

Email: aukelkar@usc.edu

Password: ••••••

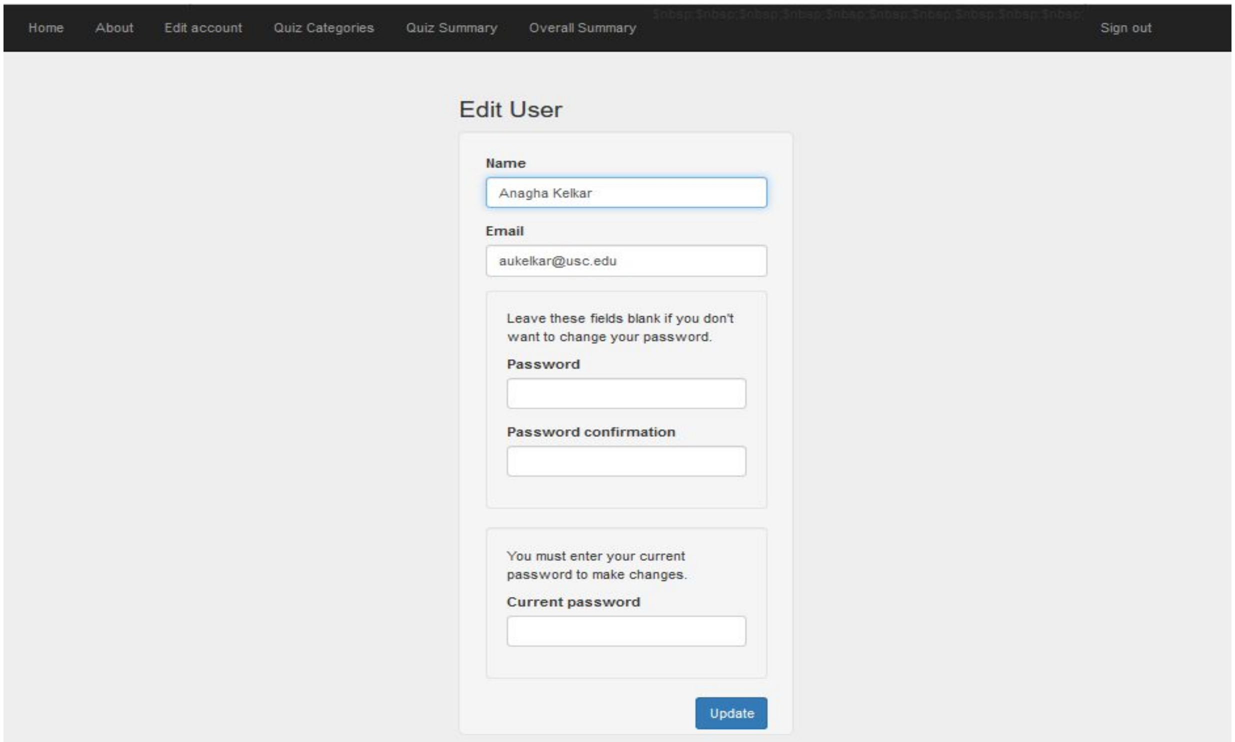
Password confirmation: ••••••

2. Once you create account, you will automatically login the account.



Welcome to Quiz Application

3. You can edit account from Edit Account Tab.



Edit User

Name

Anagha Kelkar

Email

aukelkar@usc.edu

Leave these fields blank if you don't want to change your password.

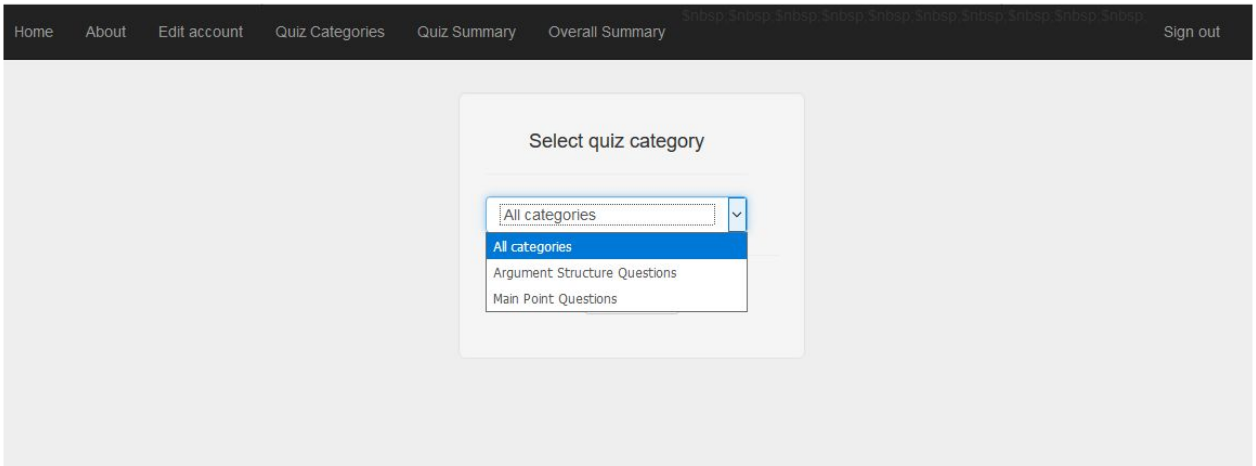
Password

Password confirmation

You must enter your current password to make changes.

Current password

3. You can start quiz by clicking on Quiz Categories.

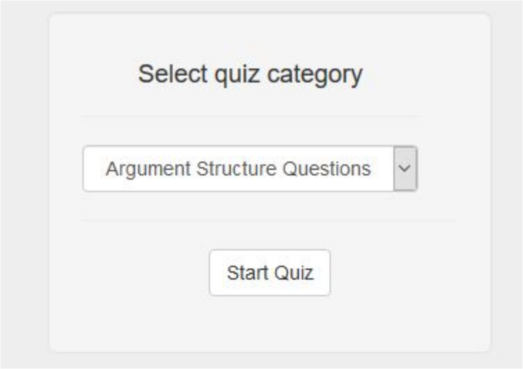


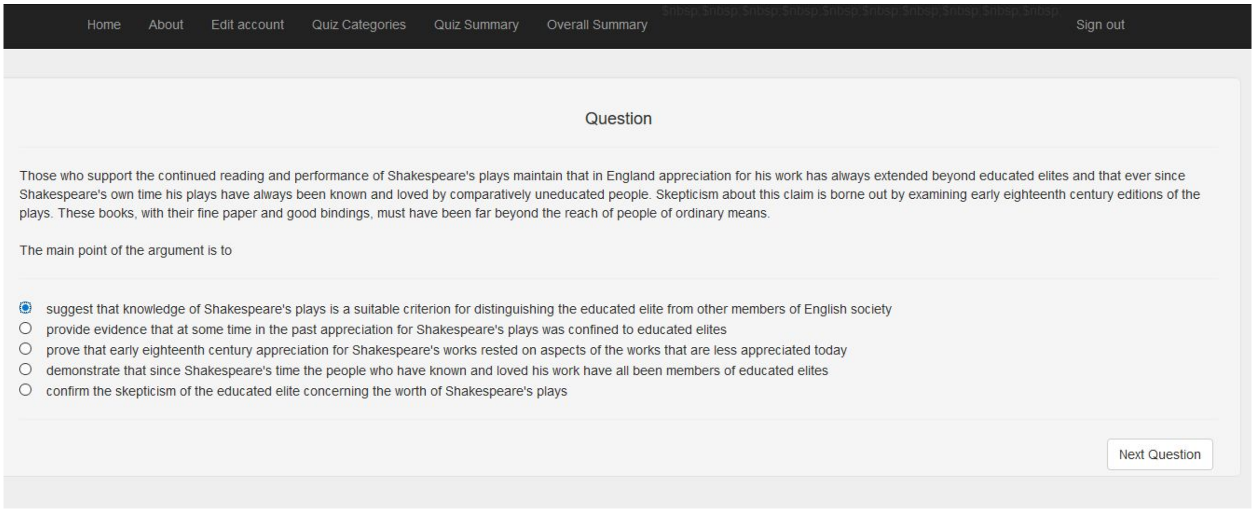
Select quiz category

All categories

You can take category-specific quiz or all categories quiz.

5. Consider you choose “Argument Structure Questions”.





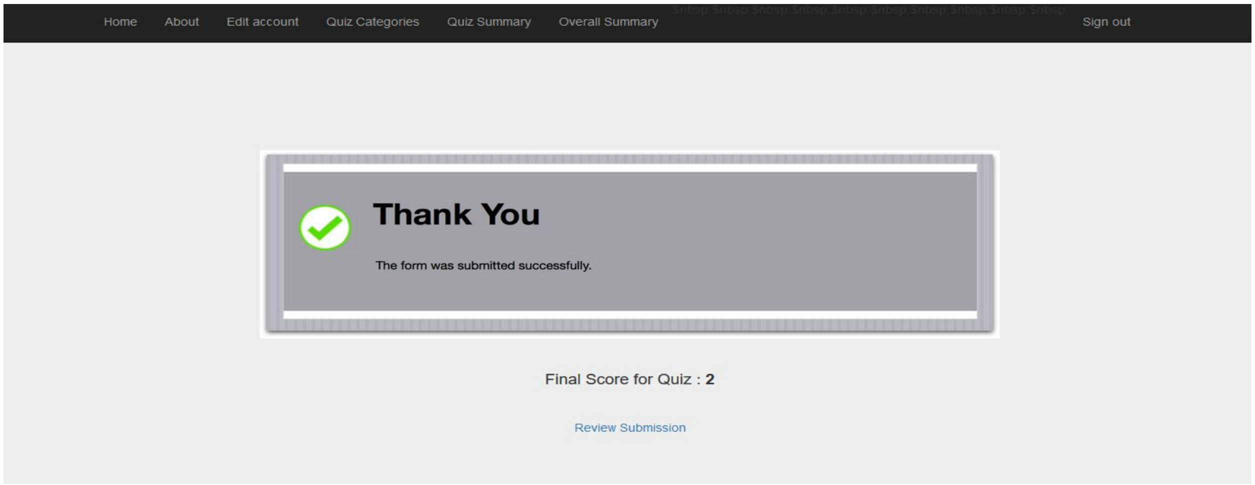
Those who support the continued reading and performance of Shakespeare's plays maintain that in England appreciation for his work has always extended beyond educated elites and that ever since Shakespeare's own time his plays have always been known and loved by comparatively uneducated people. Skepticism about this claim is borne out by examining early eighteenth century editions of the plays. These books, with their fine paper and good bindings, must have been far beyond the reach of people of ordinary means.

The main point of the argument is to

● suggest that knowledge of Shakespeare's plays is a suitable criterion for distinguishing the educated elite from other members of English society
○ provide evidence that at some time in the past appreciation for Shakespeare's plays was confined to educated elites
○ prove that early eighteenth century appreciation for Shakespeare's works rested on aspects of the works that are less appreciated today
○ demonstrate that since Shakespeare's time the people who have known and loved his work have all been members of educated elites
○ confirm the skepticism of the educated elite concerning the worth of Shakespeare's plays

At a time, quiz contains 5 questions from the category you selected and unanswered questions.

6. Once you submit quiz, you will get a score of correct answers. You can also review submission.

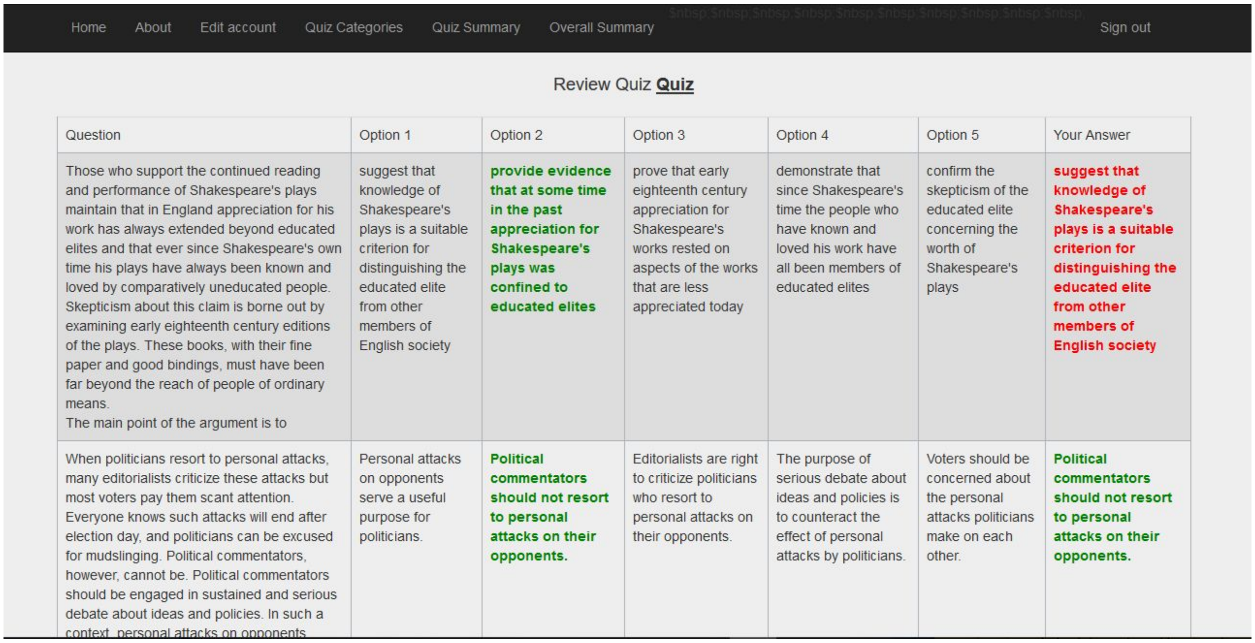


Thank You

The form was submitted successfully.

Final Score for Quiz : **2**

7. Click on Review Submission to review quiz.

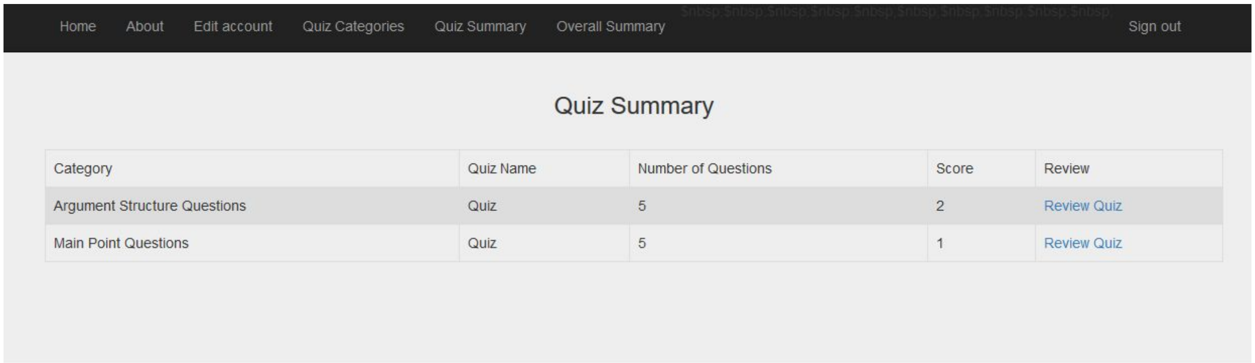


Review Quiz **Quiz**

Question	Option 1	Option 2	Option 3	Option 4	Option 5	Your Answer
Those who support the continued reading and performance of Shakespeare's plays maintain that in England appreciation for his work has always extended beyond educated elites and that ever since Shakespeare's own time his plays have always been known and loved by comparatively uneducated people. Skepticism about this claim is borne out by examining early eighteenth century editions of the plays. These books, with their fine paper and good bindings, must have been far beyond the reach of people of ordinary means. The main point of the argument is to	suggest that knowledge of Shakespeare's plays is a suitable criterion for distinguishing the educated elite from other members of English society	**provide evidence that at some time in the past appreciation for Shakespeare's plays was confined to educated elites**	prove that early eighteenth century appreciation for Shakespeare's works rested on aspects of the works that are less appreciated today	demonstrate that since Shakespeare's time the people who have known and loved his work have all been members of educated elites	confirm the skepticism of the educated elite concerning the worth of Shakespeare's plays	**suggest that knowledge of Shakespeare's plays is a suitable criterion for distinguishing the educated elite from other members of English society**
When politicians resort to personal attacks, many editorialists criticize these attacks but most voters pay them scant attention. Everyone knows such attacks will end after election day, and politicians can be excused for mudslinging. Political commentators, however, cannot be. Political commentators should be engaged in sustained and serious debate about ideas and policies. In such a context, personal attacks on opponents	Personal attacks on opponents serve a useful purpose for politicians.	**Political commentators should not resort to personal attacks on their opponents.**	Editorialists are right to criticize politicians who resort to personal attacks on their opponents.	The purpose of serious debate about ideas and policies is to counteract the effect of personal attacks by politicians.	Voters should be concerned about the personal attacks politicians make on each other.	**Political commentators should not resort to personal attacks on their opponents.**

- Green colored text from Options is the correct answer.
- If Your answer is red colored text that means your answer does not match correct answer, otherwise green.

8. If you want to review all the Quiz you have taken before, click on Quiz Summary Tab.



Quiz Summary

Category	Quiz Name	Number of Questions	Score	Review
Argument Structure Questions	Quiz	5	2	Review Quiz
Main Point Questions	Quiz	5	1	Review Quiz

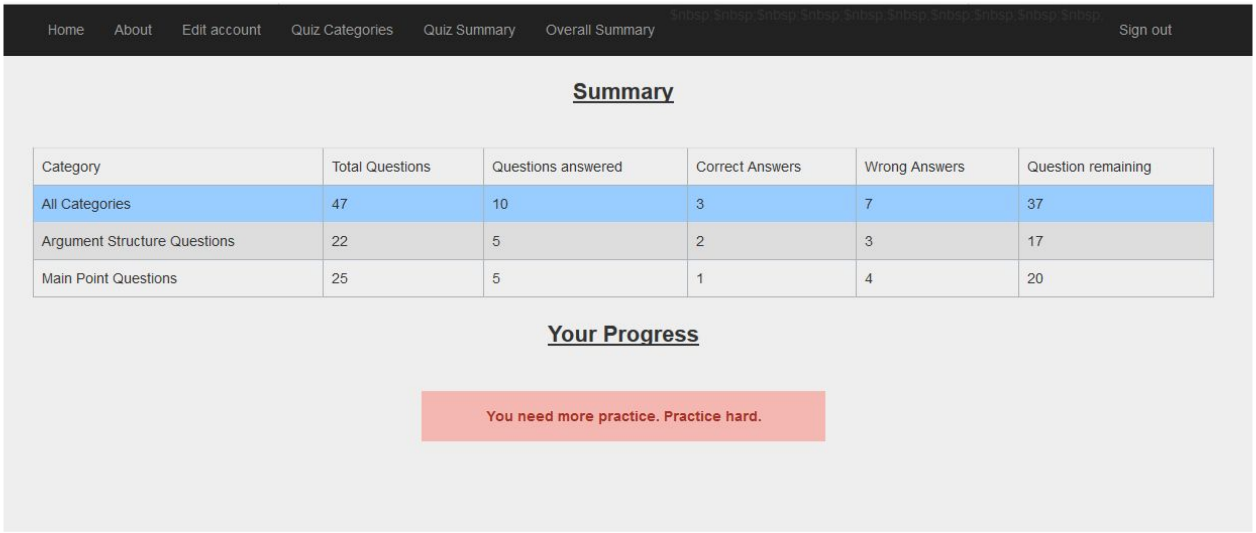
Quiz Summary tab displays all the quiz taken before with Review facility.

9. Overall Summary tab displays overall details of your progress. It displays following details:

Total Questions	Questions Answered	Correct Questions
Wrong Questions	Questions Unanswered	Your Progress

Your progress parameters filters progress based on percentage of number of correct questions.

If no questions is attempted	No progress - Yet to start quiz
If greater than or equal to 60%	You are doing great!
If less than 60%	You need to improve



Summary

Category	Total Questions	Questions answered	Correct Answers	Wrong Answers	Question remaining
All Categories	47	10	3	7	37
Argument Structure Questions	22	5	2	3	17
Main Point Questions	25	5	1	4	20

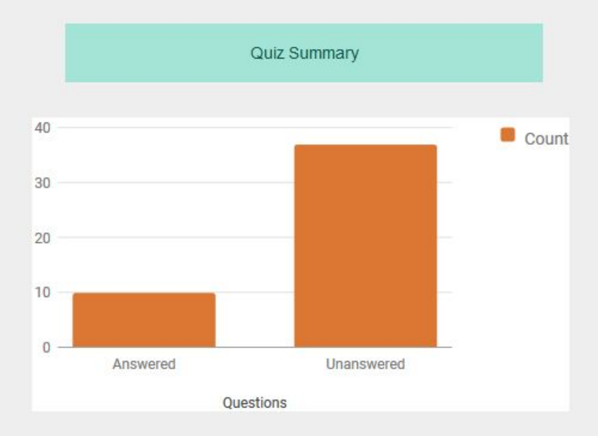
Your Progress

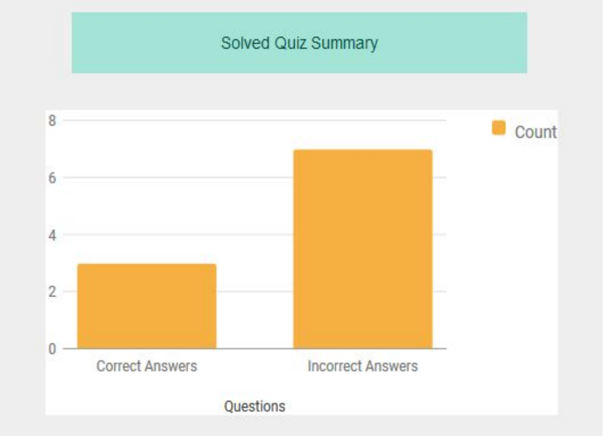
You need more practice. Practice hard.

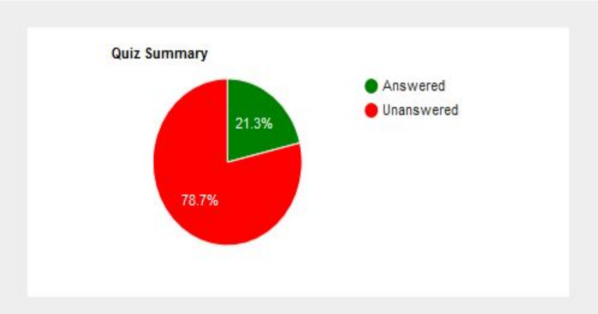
10. Graphical Summary of Quiz

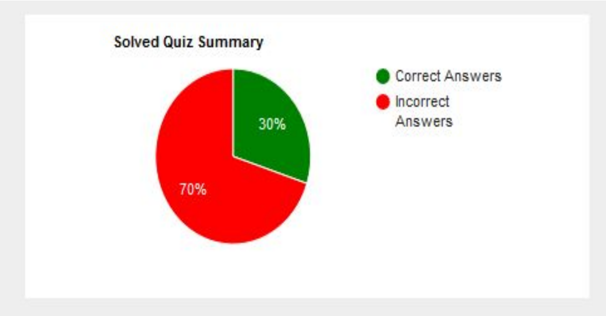
- Quiz Summary - Total number of questions answered / unanswered
- Solved Quiz Summary - Total number of questions correct / incorrect

I have implemented Pie chart and Bar graph to demonstrate visual representation.









Google charts:

For graphical representation, I have used google charts to display Bar graph and Pie chart.

Schema:

1. Category:

id	name	created_at	updated_at

2. Quiz:

id	name	created_at	updated_at	category_id

Category --- Quiz

(one) (many)

3. Question:

id	text	created_at	updated_at	category_id	unique_id

Category --- Question

(one) (many)

4. Choice:

id	text	correct	question_id	created_at	updated_at

Question --- Choice

(one) (many)

5. UserInput:

id	user_id	choice_id	question_id	correct_answer	quiz_id

UserInput stores input for

- Particular user
- Choice user has selected for particular question
- Input belongs to which quiz

6. User:

- Default fields created by Devise
- Added new field “Name”

Addition to schema :

Currently as there are only multiple choice questions, I have added just user inputs table in schema, but if there are questions with different types such as checkbox, radiobutton, text, image etc. schema will look like this.

Quiz: name, created_at, updated_at, category_id
Question: text, category_id, created_at, updated_at, **type** **type** will be one of them: ● TextQuestion ● RadiobuttonQuestion ● CheckboxQuestion ● FileQuestion We will handle this by adding association: **has_many :choices, foreign_key: 'question_id'**
Choice: text, correct, question_id, created_at, updated_at
UserInput: (It will make use of **polymorphism**) user_id, question_id, quiz_id, **type** ● TextInput - content, user_input_id, created_id, updated_id ● RadioButtonInput - choice_id, user_input_id, created_at, updated_at ● FileInput - file_default_parameters, user_input_id, created_at, updated_at ● CheckBoxInput - choice_id, user_input_id, created_at, updated_at
Category: name, created_at, updated_at

Setup:

The given excel file which contains Question with its available options and category assigned to it.

Setup in Quiz Application inserts entries in Question, Category and Choice schema.

In public folder, given excel file is present and setup file read the file and store entries in schema.

What more can be implemented?:

- As we are saving available questions, choices and categories in database, we can create a panel which will perform crud operations on Question, Choice and Category for individual or bulk operation.
- Now, questions already answered by user are not visible again in the quiz. User can review them, but can't take them again. But if user wants to appear for those questions, we can keep Revision Section, which will include quiz containing all answered questions.
- Now different random quiz is created for a particular user. *But to reduce database entries, we can create one quiz containing "x" number of questions same for all users. That way number of entries in database will reduce.*
- UserInput schema stores which user stored input for which quiz and which choice they have selected. (No need to store question_id -> to reduce queries I have added it.)
- For generation of graph, json data is created from database entries. For faster retrieval, we can create a json file and perform *caching* on it until user does not take new quiz).
- Forgot password is not implemented. Just added that feature to showcase we can perform that.
- We can further optimize queries by using slow query log mechanism.

- Default time to execute query is 10 seconds. Make it less according to your requirement which is a threshold.
- In log file, we can see how many queries are time consuming and accordingly optimize them.

  Example:

1. Model.sum(&:amount) is expensive query than Model.sum(:amount)
2. Model.all.map(&:user_id) is expensive query than Model.pluck(:user_id)
3. Model.pluck(:user_id).uniq is expensive query than Model.uniq.pluck(:user_id)
4. Use find_each instead of each to iterate over records.

Rspec:

For testing purpose, I have written specs to test certain features for website.

To test User login functionality: (using FactoryBot)

1. Sign in spec - scenarios
    - user cannot sign in if not registered
    - user can sign in with valid credentials
    - user cannot sign in with wrong email
    - user cannot sign in with wrong password
2. Sign out spec - scenario
    - user signs out successfully
3. User delete spec - scenario
    - user can delete own account
4. User edit spec - scenarios
    - user changes email address
    - user cannot cannot edit another user's profile

To test After login functionality: (using FactoryBot)

1. About page spec - scenario
    - Visit the about page
2. Home page spec - scenario
    - visit the home page
3. Navigation spec - scenario
    - view navigation links
4. Sign up spec - scenarios
    - visitor can sign up with valid email address and password
    - visitor cannot sign up with invalid email address
    - visitor cannot sign up without password
    - visitor cannot sign up with a short password
    - visitor cannot sign up without password confirmation
    - visitor cannot sign up with mismatched password and confirmation

To test Quiz basic functionality(Question, Category and Choice creation): (using FactoryBot)

Quiz setup spec - scenario

- returns correct number of questions
- returns correct number of choices
- returns correct number of categories
- returns correct number of questions for category *Argument Structure Questions*
- returns correct number of questions for category *Main Point Questions*
- returns five choices for each question

Additionally, actual quiz implementation (creation of quiz, creation of user inputs etc.) can be checked.

Note: All the screenshots are present in the application’s “assets/images/screenshots” folder.

Github link: https://github.com/AnaghaKelkar/quiz_app