eYSIP2016

Question bank Management

Bhalchandra Naik
and Tushar Shah
Mentors : Shubham Gupta, Uttam Kumar Gupta and Yogita
Mali
Duration of Internship: 10/06/2016-24/07/2016

2016, e-Yantra Publication

Question Bank Management

Abstract

The aim of this project is to organize and ease the process of question bank management through a web-based application. Question Bank Management can mainly be bifurcated into two important tasks viz. Question Creation and Question Reviewal.

The application simplifies the task of adding elements like diagram, equations and code to questions by providing interfaces for the same. For better discretion of solving the questions, elements like difficulty level, Category, tags and expected solving time can also be added to question.

The applicaton shall automate the task of evenly distributing the questions amongst the users. Revision history to keep track of all changes made to a question and control the versions of a question, is also maintained by this application.

The ultimate goal of this application is to reduce the fuss created in a closed group during the conventional question bank management process and give the process a drive(which the conventional process may lack) that is required for efficient and harmonious performance of the outfit.

Completion status

- Task Accomplished

  1. Freezing the SRS (Software Requirements Specification)
  2. Set up and study Laravel and Database Management
  3. Schema Development
  4. Account management (including access management)
  5. Revision History of Question
  6. Difficulty level of questions and expected solving time
  7. Converting question text to image
  8. Adding diagrams to questions
  9. Adding questions with choices
  10. Equation box (latex to image)
  11. Coding window (code text to image)
  12. Adding difficulty level tags and category tags
  13. Navigating Questions
  14. Documentation

- All specified tasks have been accomplished

1.1 Hardware parts

- No Hardware parts used

1.2 Software used

1. Linux environment

   - Setting up the system:
     (a) Software Used: Ubuntu
     (b) Software version: 16.04
     (c) download link
   - Setting up the server
     (a) Software used :
       - Apache
       - PHP
     (b) Versions:
       - Apache 2.4.12
       - PHP 5.6.11
     (c) Installation:
       - Apache
       - PHP
   - Setting up the PHP framework
     (a) Software Used :
       - Composer
       - Laravel
     (b) Version:
       - Laravel 5.2.39
       - Composer 1.1.2
     (c) Installation:
       - Composer
       - Laravel
     (d) Conifguring Apache :
       - Ensure that the project folder has proper permissions.
       - Now go to the /etc/apache2/sites-available directory and use the following command to create a configuration file for our laravel install
       - Now add the following content to the file and close it after saving.

2. Revisionable Trait

- Revisionable Trait named sofaRevisionable Trait is used
- Refer the link of the trait for importing link here
- According to the need the Revisionabletrait.php was modified and used

1.3 Assembly of hardware

No Hardware has been used

1.4 Software and Code

Github link

1.4.1 Database Schema

The database schema used in the application has been described in detail in the follwoing sections

1. *equations(exp_id,exp_latex,exp_image,created_at,updated_at)*
   - exp_id : auto-incrementing primary key
   - exp_latex : expression of the equation in LaTeX
   - exp_image : link to the image stored in the file-system
   - created_at : timestamp of creation
   - updated_at : timestamp of updation
2. *codes(code_id, code_description, code_image_path, created_at, updated_at)*

- code_id : auto-incrementing primary key
- code_description : description of code
- code_image_path : link to the image stored in the file-system
- created_at : timestamp of creation
- updated_at : timestamp of updation

3. *diagram(<u>diagram_id</u>, path, created_at, updated_at)*

    - diagram_id : auto-incrementing primary key
    - path : link to the diagram stored in the file-system
    - created_at : timestamp of creation
    - updated_at : timestamp of updation

4. *category(<u>key</u>, name, created_at, updated_at)*

    - key : auto-incrementing primary key
    - name : name of the category(Quantitative, Programming or Electronics)
    - created_at : timestamp of creation
    - updated_at : timestamp of updation

5. *difficulty(<u>key</u>, name, created_at, updated_at)*

    - key : auto-incrementing primary key
    - name : difficulty name(Easy, Medium or Hard)
    - created_at : timestamp of creation
    - updated_at : timestamp of updation

6. *maths_symbols(<u>id</u>, code, description, type, created_at, updated_at)*

    - id : auto-incrementing primary key
    - code : HTML UTF-8 code of the mathematical symbols
    - description : short description of the symbol
    - type : used to point to the subject matter where the symbol is commonly used in
    - created_at : timestamp of creation
    - updated_at : timestamp of updation

The data stored in this table has not been entered manually. A Map-Reduce process using Hadoop was implemented to process the data present on the web-page of w3schools.com(link here) and filter it into a form conforming with structure of the table implemented. JAR file of the program used is in the project folder titled *MathSymbolsProcessor.jar*.

7. *math_symbols_group(id, group_name, div_id, created_at, updated_at)*
   - id : auto-incrementing primary key
   - group_name : Name of the topic pertaining to which special symbols have been stored
   - div_id : used in assigning IDs to divisions created in views
   - created_at : timestamp of creation
   - updated_at : timestamp of updation
8. *options(option_id, q_id, revision, option_no, description, created_at, updated_at)*
   - option_id : auto-incrementing primary key
   - q_id : key of the question whose options are being stored
   - revision : the version of the options (changed when the options are changed)
   - option_no : the serial number of the option in the list of options
   - description : Content of each option
   - created_at : timestamp of creation
   - updated_at : timestamp of updation
9. *tags(id, name, created_at, updated_at)*
   - id : auto-incrementing primary key
   - name : name of the tag with unique constraint
   - created_at : timestamp of creation
   - updated_at : timestamp of updation
10. *q_tag_relations(key, q_id, tag_revision, tag_id, created_at, updated_at)*
    - key : auto-incrementing primary key
    - q_id : key of the question whose tags are being stored

- tag_revision : the version of the tags (changed when the tags are changed)
- tag_id : ID of the tag whose nam is stored in *tags* table
- created_at : timestamp of creation
- updated_at : timestamp of updation

11. *q_tables(q_id, description_id, exp_id, created_by, last_edited_by, diagram_id,current_revision, options, code_id, difficulty, time, category, tag_revision, created_at, updated_at)*

    - q_id : auto-incrementing primary key to identify each question in the database uniquely
    - description_id : contains a reference to a record in descriptions table which has the description details of the question
    - exp_id : contains a reference to a record in the equations table which stores all the details pertaining to the created equations
    - created_by : contains a reference to a user in the users table who has created that particular question
    - last_edited_by : contains a reference to a user in the users table who has most recently made any changes to the question(be it review or edit)
    - diagram_id : contains a reference to diagrams table where details of the diagram stored in the file system are stored
    - current_revision : stores the current version of the question
    - options : stores the current version of options being used
    - code_id : Stores a reference to a record in the codes table where details regarding the code being used in the questions is stored
    - difficulty : stores the difficulty level of question
    - time : stores the time required to solve a particular question
    - category : category of each question(Quantitative, Electronics or aptitude)
    - tag_revision : the current version of the tags (changed when the tags are changed)
    - created_at : timestamp of creation
    - updated_at : timestamp of updation

1.4.2 Features

1. User Authentication and Registration

    - The application supports 2 types of users :
        (a) Administrator
        (b) Normal User
    - Normal users may further have two roles creator(user who creates the question) and reviewer(who reviews questions allotted to him).
    - The Application houses and serves a closed group of users, due to which common users are unable to register onto the application.
    - Thus the Admin has the authority to add new users to the application
    - A user wont be able to access any of the utilities of the application unless she/he has been authenticated or unless she/he has a user account registered for him
    - Password reset procedure has been created for normal users, where a email containing a password reset link is sent to the users on his email-ID
    - The admin can also delete users if need be from the application.

2. Question Creation

    - Question description and Coding box :
        - Question description being typed by the user is transformed into image in real-time and previewed below accordingly.
        - The description being typed is entered on an HTML hidden canvas which is then used to generate an image, whose URL is updated in the image present on every *'onkeyup'* event
        - The same coding logic applies for the code to image conversion, only wth variation in colors and text font used to give llokk and feel consistent with the field of coding
    - Equation Box :
        - The LaTeXcode for equation typed in this window is used to obtain the image URL generated of the equation photo generated by the open API of code-cogs. CodeCogs equation editor
        - The URL of the image generated is dependent on the equation typed.

- The URL shall have a fixed part and an variable part which varies with every equation
- This is the permanent part :
- As said earlier the variable part is dependent on equation being typed which is obtained in the following manner:
  (a) if the number of adjoining spaces is more than one then they are replaced with just one single space using the following regular expression :
  (b) then variable part is URI encoded and then concatenated to the fixed part which generates the entire link, which can now be updated in the *src* of the image preview tag and the value of the hidden form field

- Options :
  Options are being created on an *onkeyup* event on the number field specified. There can be 2-6 options currently
- Diagram:
  Users can upload the diagrams, from their local file-systems to add to the question.
- Tags and Category :
  - Each question must fall into either of the following categories pertaining to the subject matter
    (a) Quantitative
    (b) Electronics
    (c) Programming
  - Each question must have certain tags to it pertaining to the concepts of topics covered in the question
    eg: turing machine, integrtion, C etc
- Time Required and difficulty level of questions
  - For better discretion of solving the question bank creators must give attributes to questions like Time Required(minimum 30 seconds) and Difficulty level of questions (easy, medium or hard).
- Special symbols keyboard
  - Data stored in the tables maths_symbols and math_symbols_group about the symbols is used to render the keyboard.

3. Picking a Question

- While browsing the question bank the users may want to compose a new question by making minor changes to an existing question
- The users shall be redirected to page with the form similar to that of the Compose interface, with fields set to the value of the initial question.
- on submit the post request is sent to the Controller closure of the Create a question feature and a new question is composed with the creator set to the user who picked the question

4. Editing a Question

   - While browsing his own questions int the 'Home' the user may find that the content of any question he/she created is inconsistent with the subject matter
   - The user can then edit that question and correct it. For editing the user shall be directed to an interface similar to that of compose with fields set current attributes of the question.
   - After making the changes, the post request for that page shall be sent to controller closure where the received values from that form are compared with the existing values, if the values are the same, no updates are made.

5. Navigating the Question Bank

   (a) Currently searching the questions is based on tags and string search on the question description. Search box for entering the string and a multiple select box for the selecting the tags is provided. This feature has been implemented in the *Home* and *Browse* interface

   (b) *Home*

      - Here the User can browse the questions created by him
      - Search queries sent from here are give results of the questions created by that user, that satisfy the submitted queries

   (c) *Browse*

      - Here the User can browse the entire questions
      - Search queries sent from here are give results of the questions that satisfy the submitted queries

6. Review

(a) Criteria : For Distribution of questions to reviewers

- Each questions should have only 2 reviewers.
- The Creator of the questions should not get his/her own created questions for reviewing.
- The Distribution of questions should be on fairness and efficiency basis.

(b) Process : Alloting questions to review

- The process runs a special algorithm to submit questions for review
- The algorithm distributes questions to the different users efficiently and in such a way that the user is paired with all the other users
- From the set of all possible pairs, random pairs is given a question to review individually
- The pair should not be repeated until all the combinations of pairs of different users are alloted a question
- Once the question is alloted, the flag of allotment is set which indicates the status whether 'alloted' or not
- After the question is reviewed,the flag for allotment is set which indicates whether reviewed or not.

(c) Algorithm :

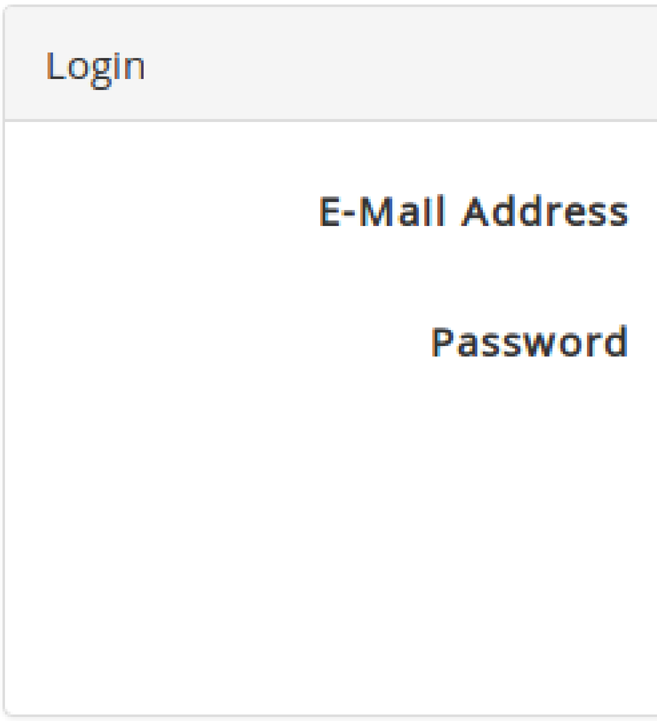
- The List of users who created question are fetched from the database
- for each user all the questions are fetched which are to be reviewed
- The matrix is formed corresponding to the self-cartesian product of users who haven't created any question
- step 1 foreach question, a pair($(row_m, column_n n)$ refers to the index of users) in the upper triangular part of the matrix is alloted a question
- step 2 if the pair in the upper triangular matrix is finished then shuffle the sequence of users and reform a matrix corresponding to the self-cartesian product and repeat step 2
- step 3 if no of questions gets exhausted then break the current iteration and go to the outer loop
- iterate or end inner loop
- iterate or end outer loop

(d) Review :
   - Given to the users, the question to be reviewed, the user might want to modify or might not want to modify
   - If modified then there is a history on the update of the question and questions get removed from the reviewers list
   - If not modified then the reviewers submits no changes and the question gets removed

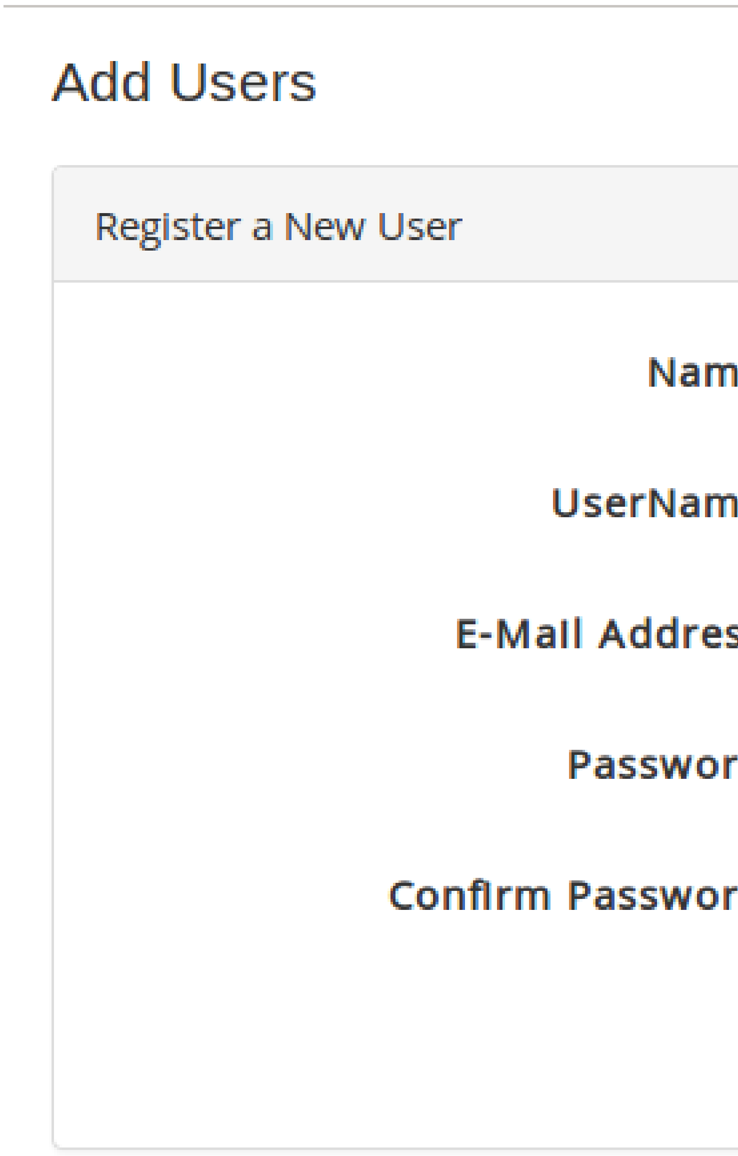
7. Review interface
   (a) Given to the users, the question to be reviewed, the user might want to modify or might not want to modify
   (b) It contains the following button
      - No Change : Whenever the reviewers does not want to modify considering it as appropriate
      - Modify : Whenever the reviewers wants to modify the edit on question method is called in the new browser
   (c) If modified then there is a history on the update of the questions and questions get removed from the reviewers list
   (d) If not modified then the reviewers submits no changes and the questions get removed

8. History interface
   (a) This Section shows the latest version of all questions and the no of version a question has in the history
   (b) Every question has a button indicating the version no, provided that the question has the version history
   (c) After clicking on the version no button the user is redirected to a new page where there will be display of latest version and n^{th} version of the question
   (d) It contains the following button
      - Cancel : Redirects the users to the previous page
      - Restore : Restores the current version to the version
   (e) According to restore the current version is thereof displayed

1.5 Use and Demo



Some glimpses of the application Login Page


Add Users

Register a New User

Nam

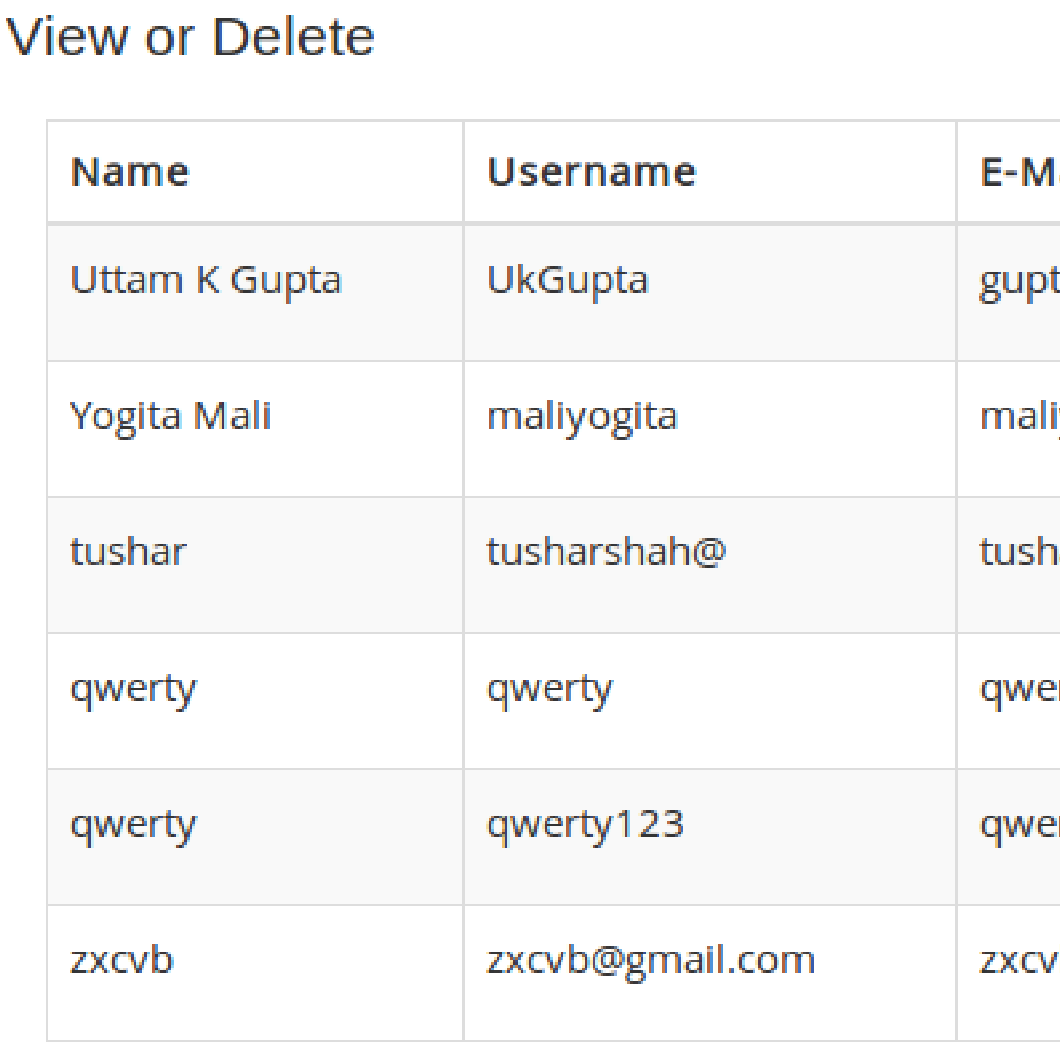
UserNam

E-Mail Addres

Passwor

Confirm Passwor

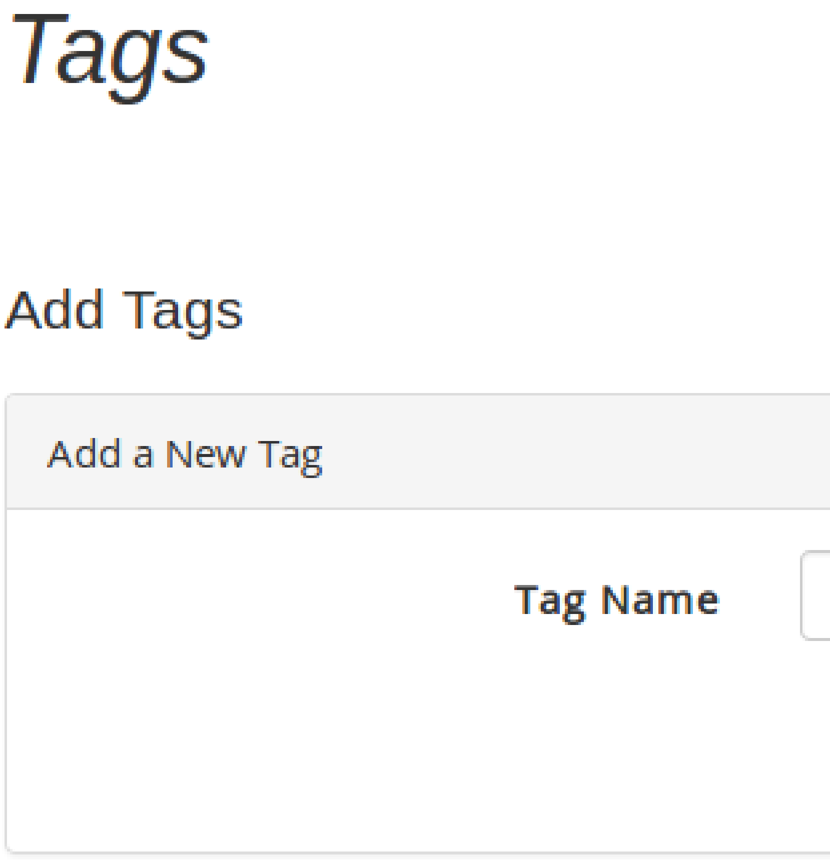
New User Registration on Admin Panel



View or Delete

Name	Username	E-M
Uttam K Gupta	UkGupta	gupt
Yogita Mali	maliyogita	mali
tushar	tusharshah@	tush
qwerty	qwerty	qwe
qwerty	qwerty123	qwe
zxcvb	zxcvb@gmail.com	zxcv

List of Users



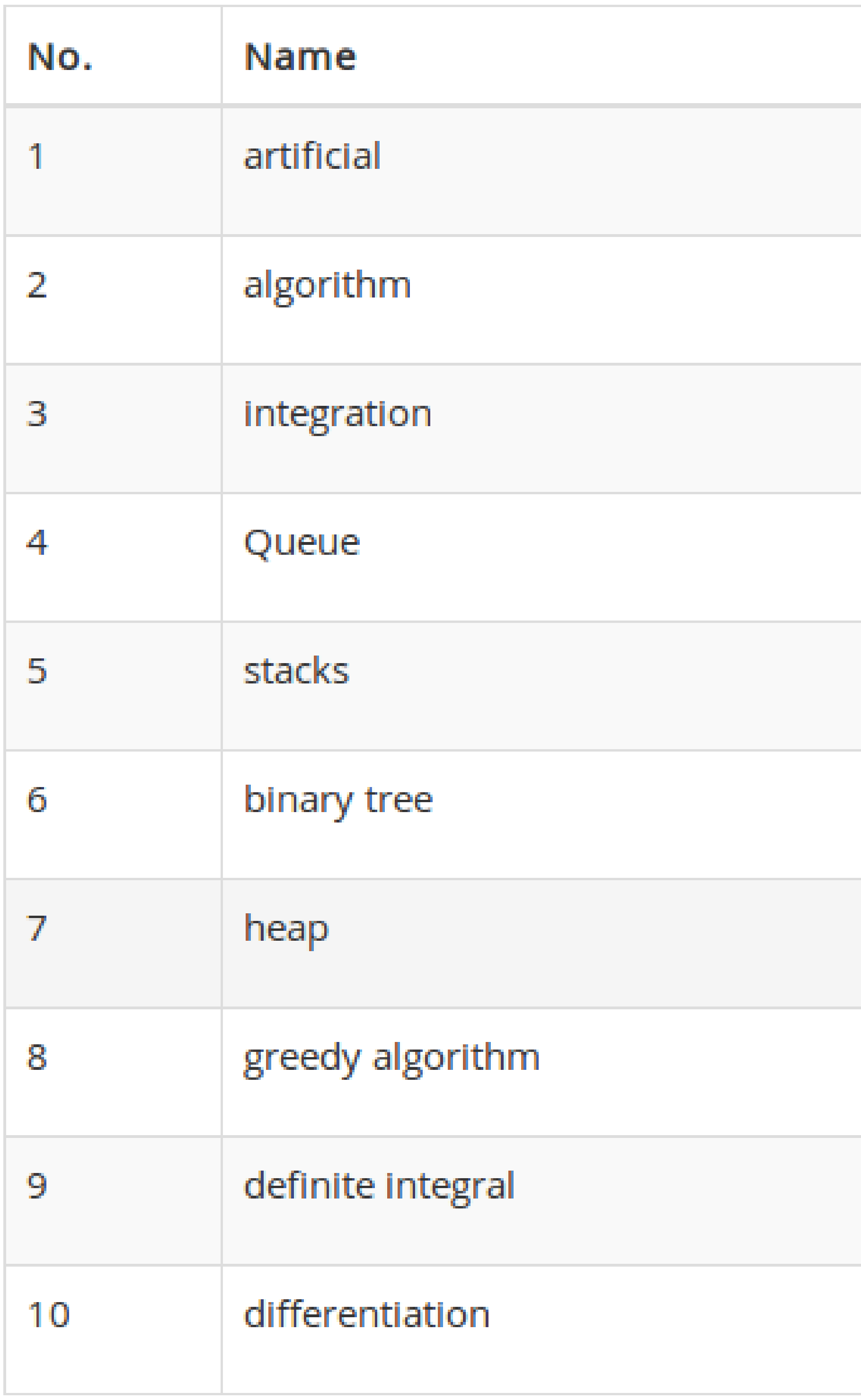
Tags

Add Tags

Add a New Tag

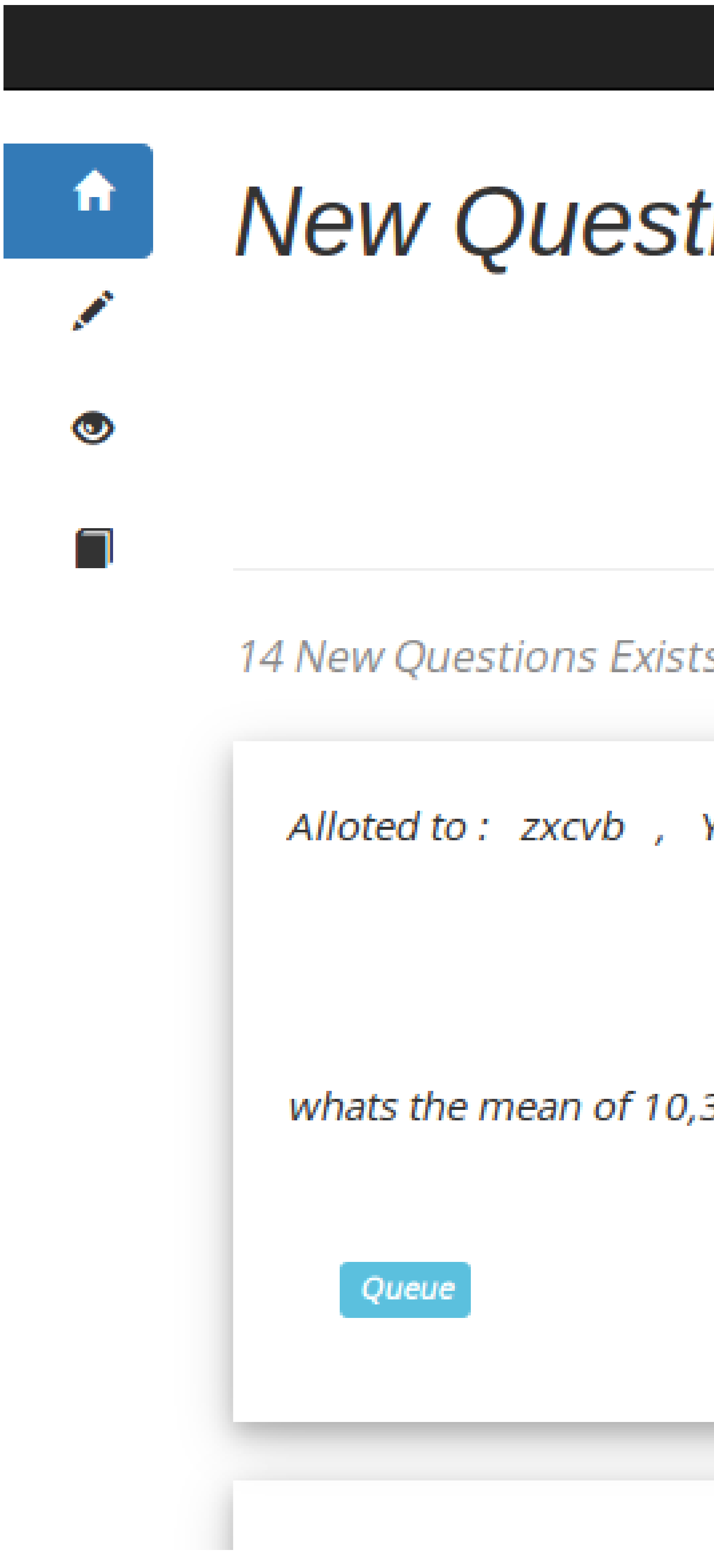
Tag Name

Adding new Tags by Admin



No.	Name
1	artificial
2	algorithm
3	integration
4	Queue
5	stacks
6	binary tree
7	heap
8	greedy algorithm
9	definite integral
10	differentiation

Displaying current tags



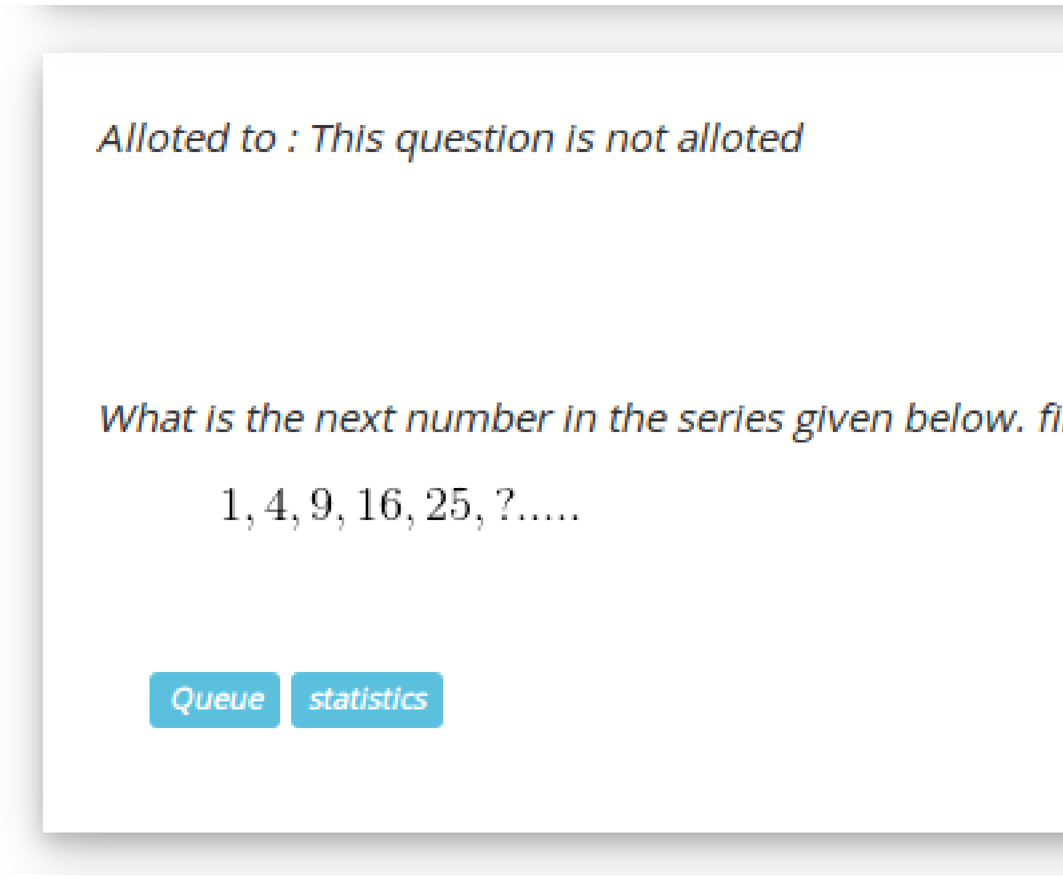
New Quest

14 New Questions Exists

Alloted to : zxcvb ,

whats the mean of 10,3

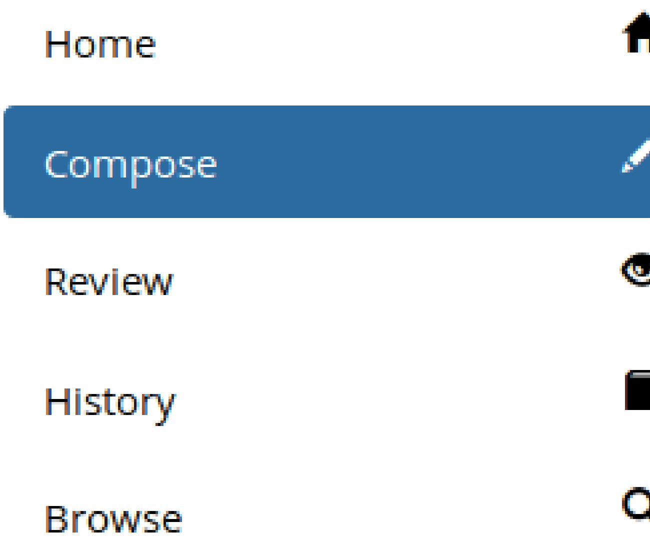
New Question for review on Admin Panel


Alloted to : This question is not alloted

What is the next number in the series given below. fi

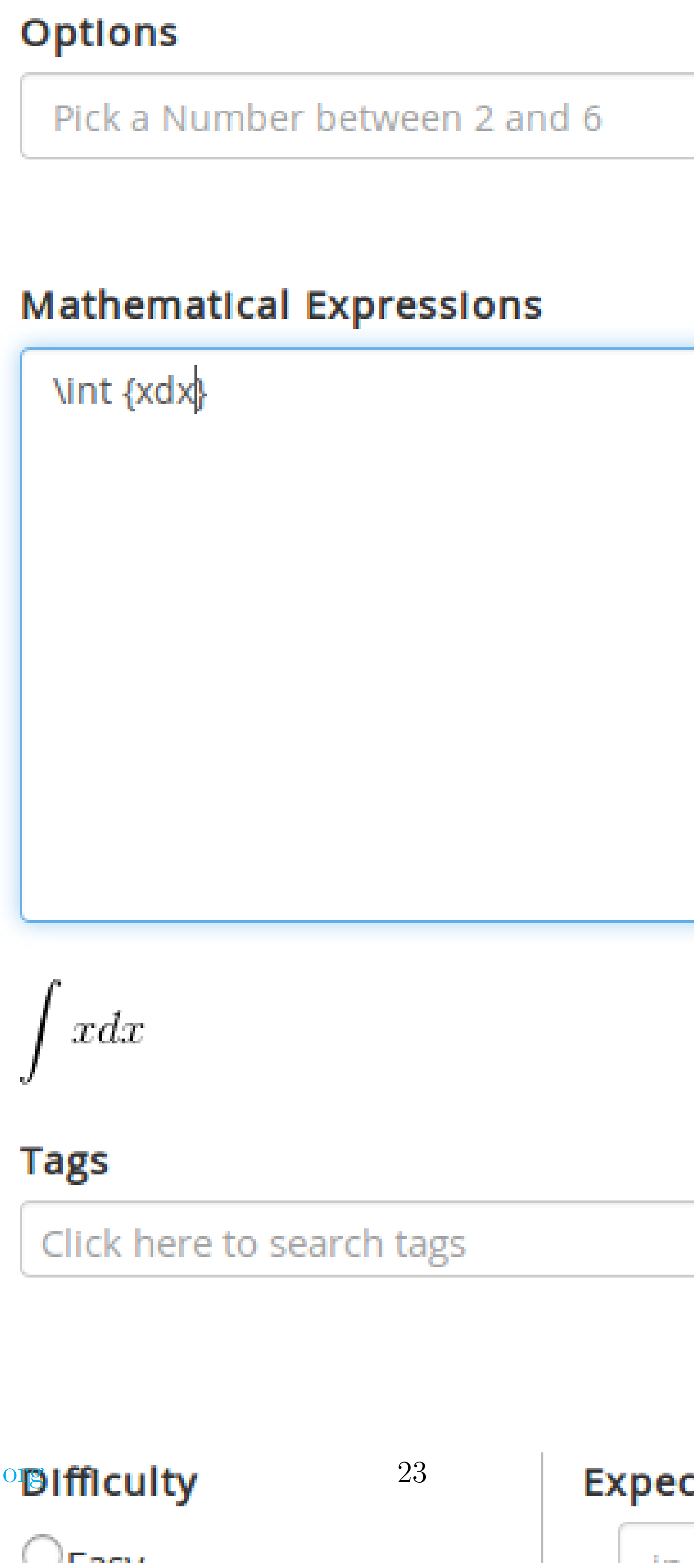
$1, 4, 9, 16, 25, ?.....$

Alloted or not



Text to image conversion and options

Special Symbol keyboard `compose4.png`



Options

Pick a Number between 2 and 6

Mathematical Expressions

```
\int {xdx}
```

$$\int xdx$$

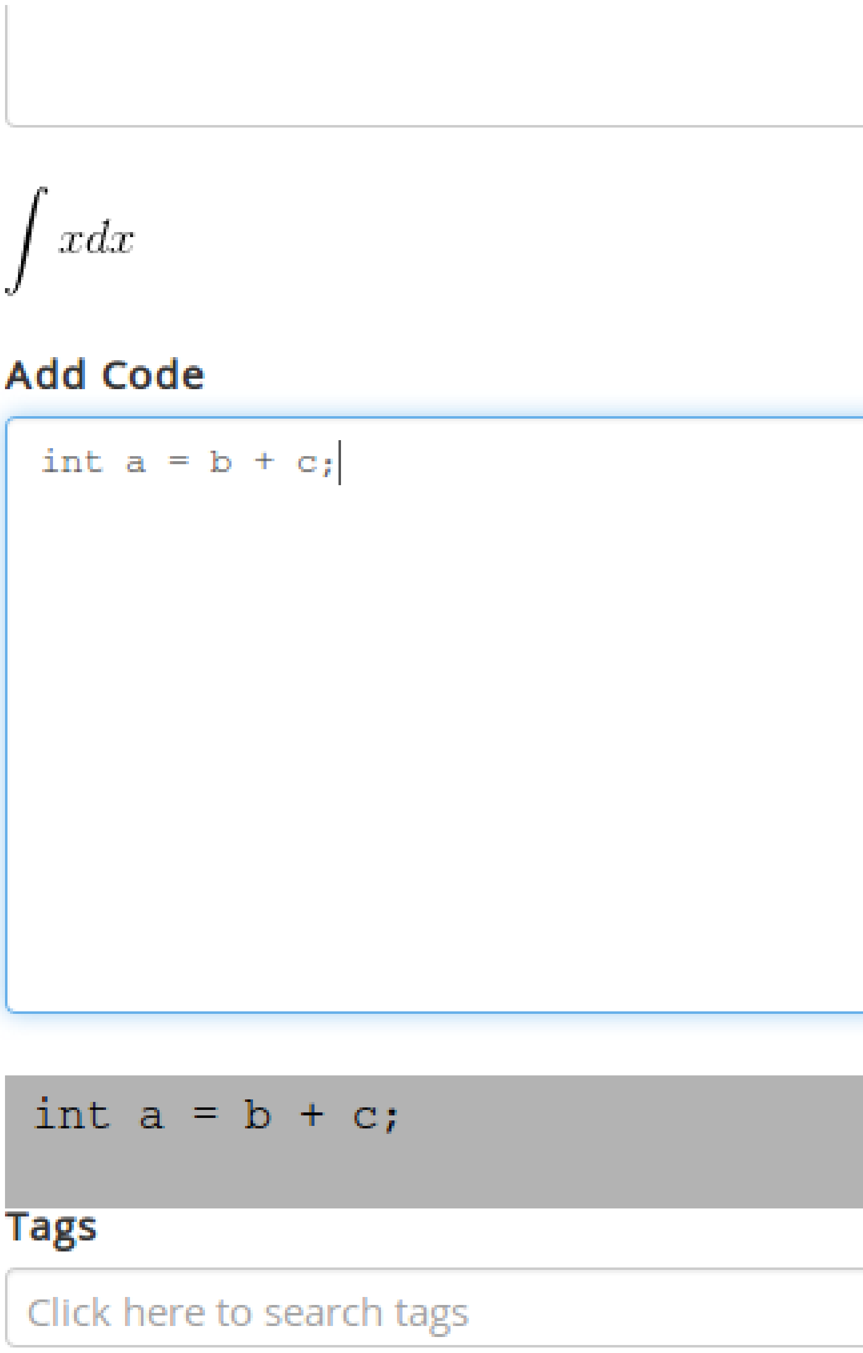
Tags

Click here to search tags

Difficulty

Expec

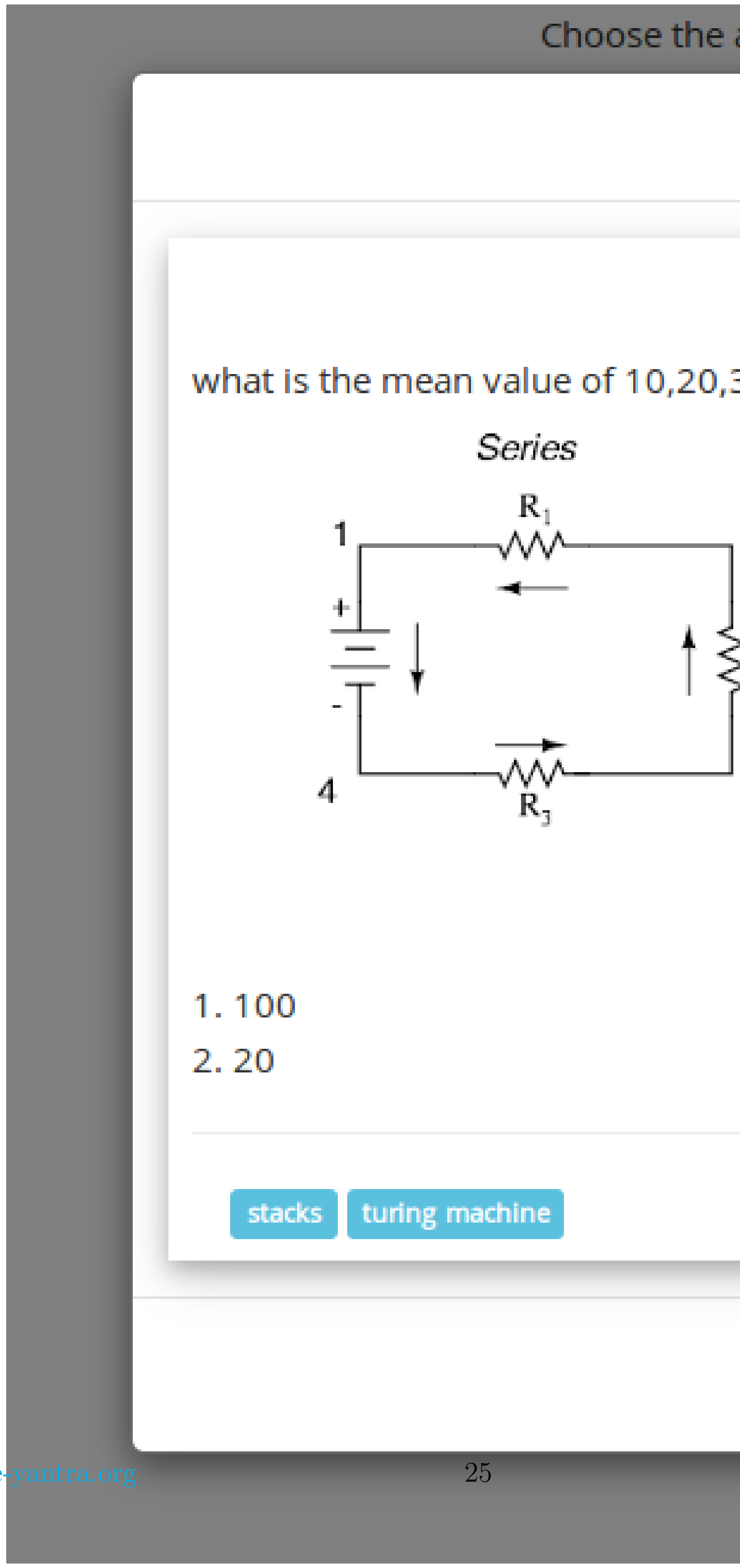
LaTeX to image conversions


$$\int x dx$$

```
int a = b + c;
```

```
int a = b + c;
```

Code to image conversion

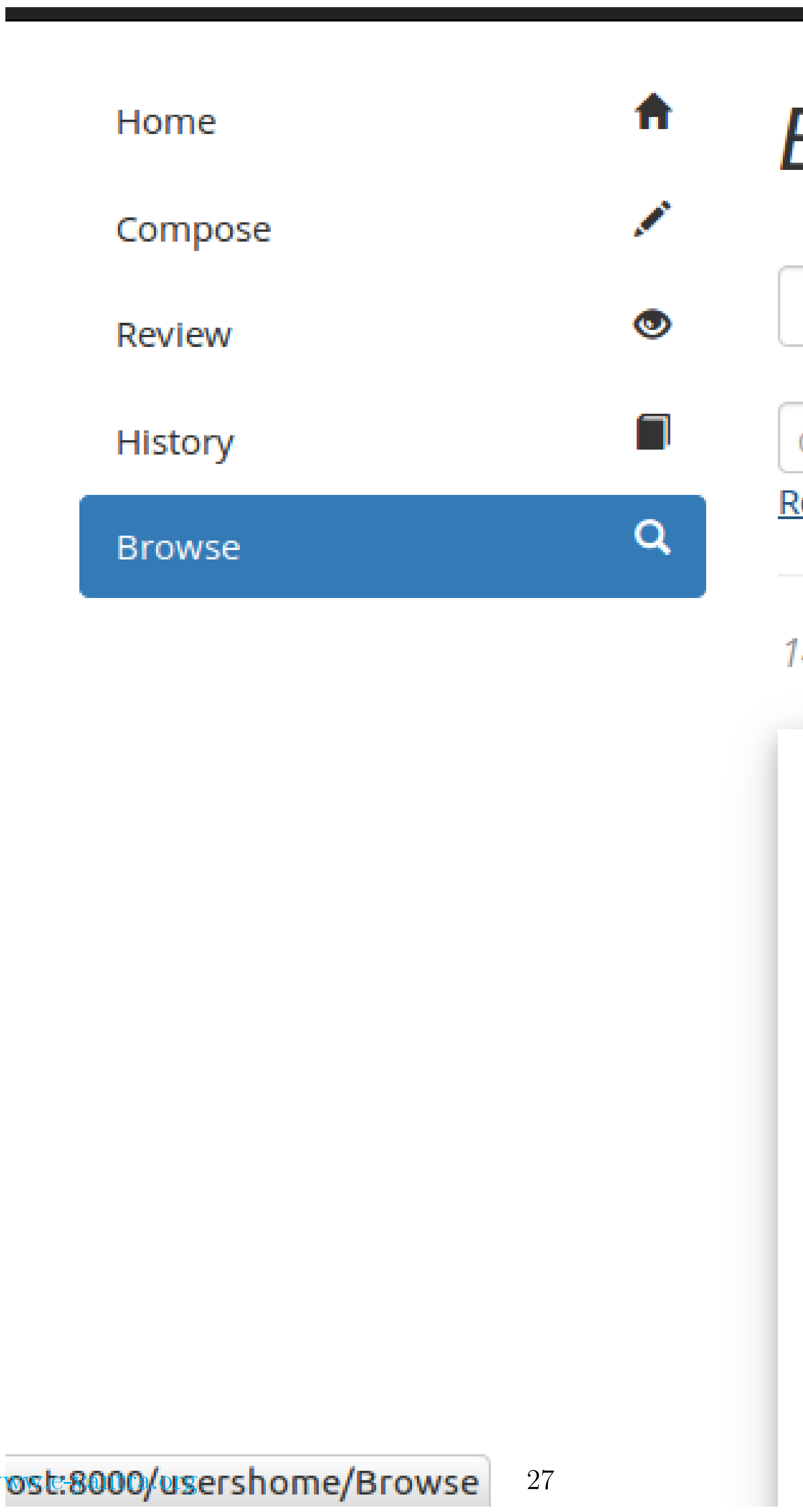


Choose the

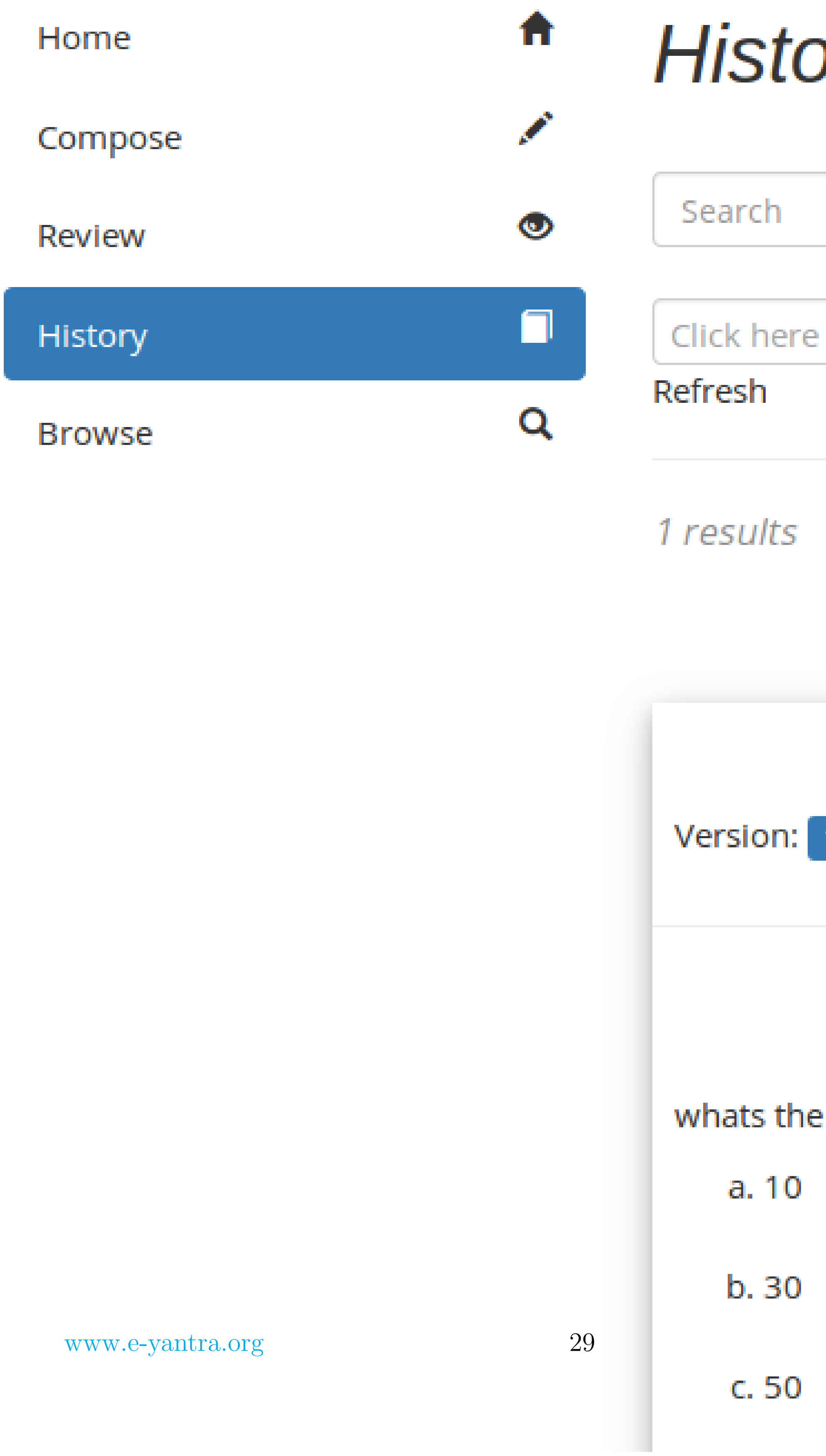
what is the mean value of 10,20,3

1. 100
2. 20

Preview of question befor submit



Browsing the question bank



1 results

Version:

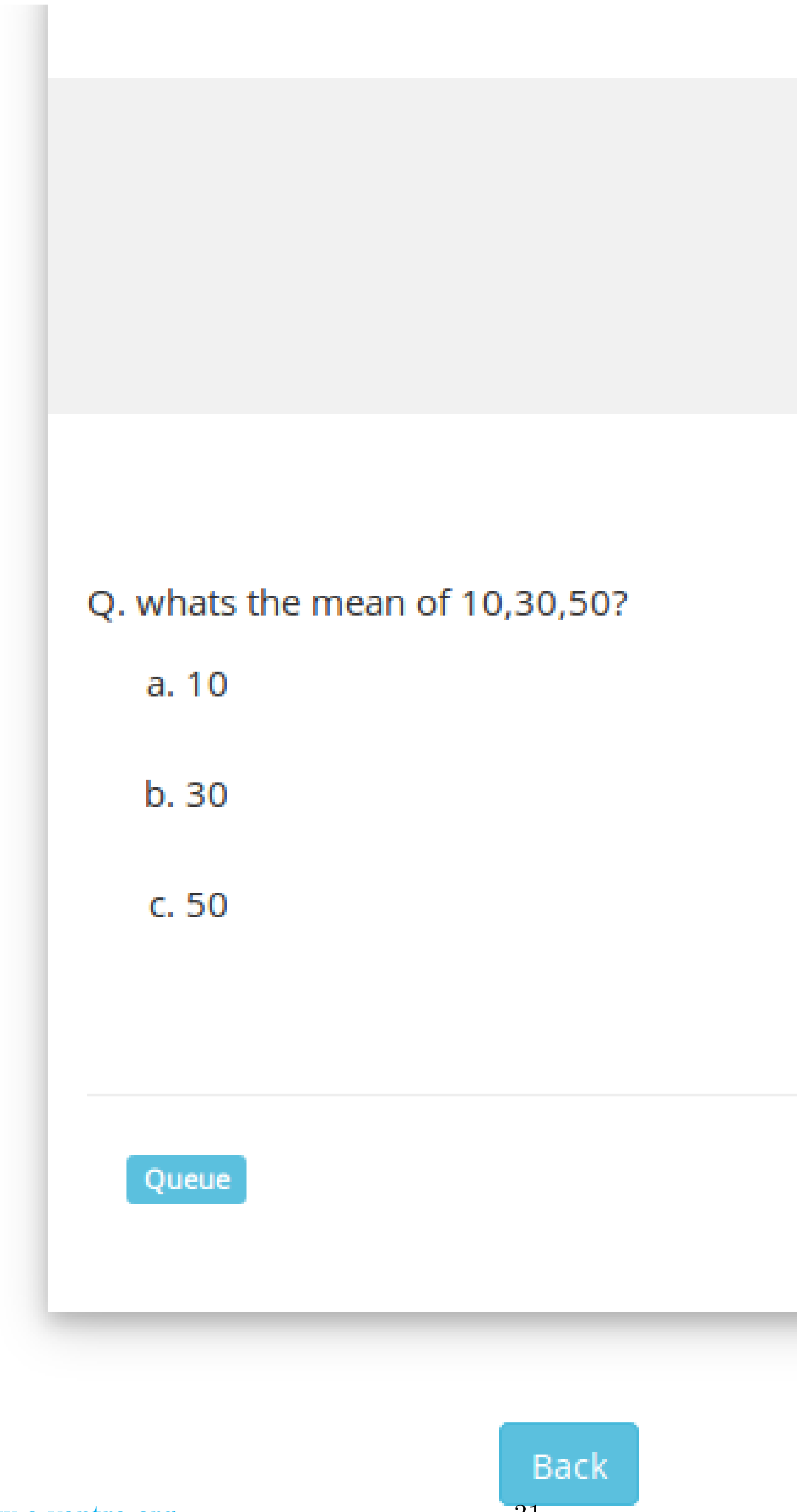
whats the

a. 10

b. 30

c. 50

History interface



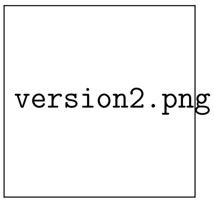
Q. whats the mean of 10,30,50?

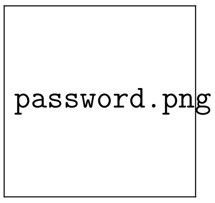
a. 10

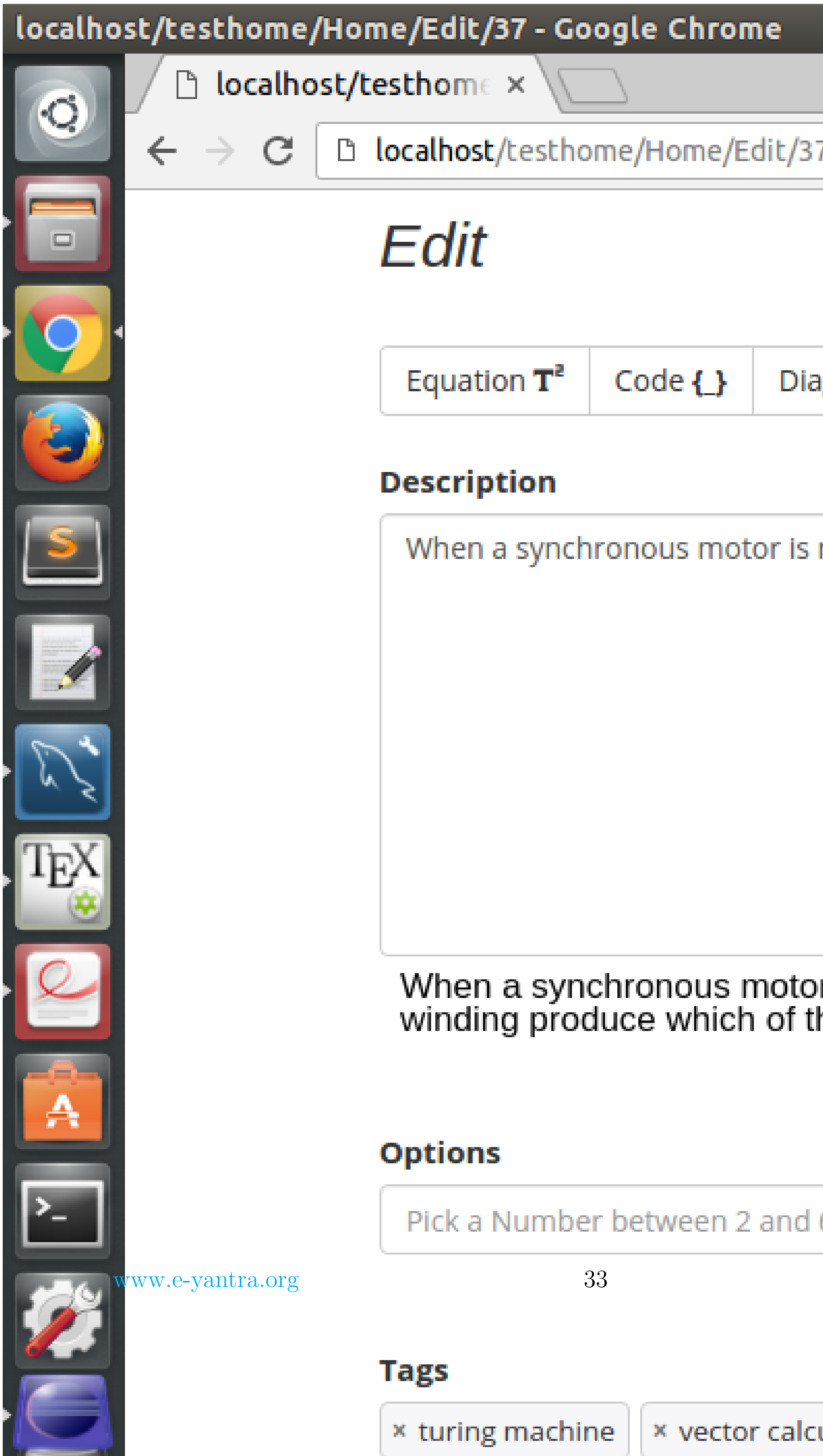
b. 30

c. 50

New Version

Old Version 

Password Reset link 


localhost/testhome/Home/Edit/37 - Google Chrome

localhost/testhom

localhost/testhome/Home/Edit/37

Edit

Equation T² | Code { } | Dia

Description

When a synchronous motor is

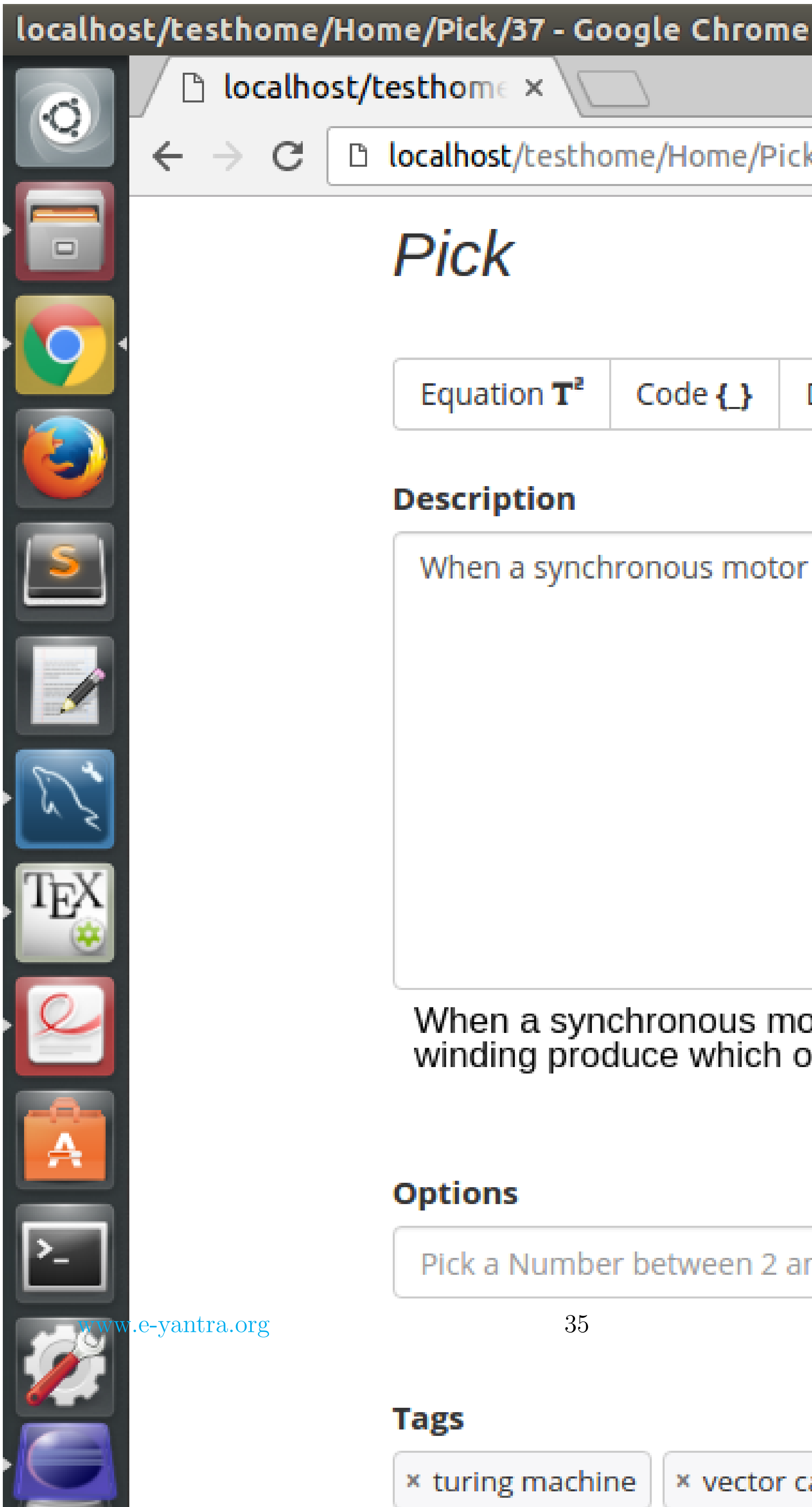
When a synchronous motor
winding produce which of th

Options

Pick a Number between 2 and

Tags

× turing machine × vector calc


localhost/testhome/Home/Pick/37 - Google Chrome

localhost/testhome

localhost/testhome/Home/Pick

Pick

Equation T² Code { }

Description

When a synchronous motor

When a synchronous mo
winding produce which o

Options

Pick a Number between 2 ar

Tags

× turing machine × vector c

1.6 Future Work

- Interesting insights can be gained by performing mining operations on the data generated in the system
- As every application has scope for improvement, studies can be done to develop various algorithms for distribution of questions in more selective manner and organized (as per categories or sub-categories, or difficulty level)
- The application has a coding window. APIs that give results of the code being typed and allow features such as auto-indentation can also be integrated into the existing system
- An algorithm for automating the process of creating question sets using the existing question bank can be created, each of which is of the simillar difficulty level.

1.7 Bug report and Challenges

- Bugs
  1. Small UI flaws may be present
- Challenges
- Designing a good UI
- Since the database is in the Boyce Codd Normal Form, developing efficient queries for merging data was a challenge
- Creating the database that shall not suffer major modifications in the long run
- Text to image conversion

Bibliography

[1] Ad Kamerman and Leo Monteban, *WaveLAN-II: A High-Performance Wireless LAN for the Unlicensed band*, 1997.