|  |  |
| --- | --- |
| Ask and answer question in flask application :-  App.py   1. asking the question.(by the user / student)   @app.route('/ask',  methods = ["POST", "GET"])  def ask():      user = get\_current\_user()      db = get\_db()      if request.method == 'POST':  # we will insert the question asked by user and experts name in questions table.          db.execute('insert into questions (question\_text, asked\_by\_id, expert\_id) values (?, ?,  ?)', [request.form['question'], user['id'], request.form['expert']])          db.commit()          return redirect(url\_for('index'))   # functions name for the first page.      # get all the expert users from the database and show them in ask.html page..      expert\_cur = db.execute('select \* from users where expert = 1')      all\_experts = expert\_cur.fetchall()      return render\_template('ask.html', user=user, all\_experts = all\_experts)   1. seeing all the questions asked by the normal user/ student , only expert user can see the unanswered question and answer them. So we have to login as expert user in order to see the unanswered questions and to answer them.   @app.route('/unanswered')  def unanswered():      user = get\_current\_user()      # get all questions which are unanwered, means where answer\_text is null,  from db,  questions table.      db = get\_db()      question\_cur = db.execute('select questions.id, questions.question\_text, users.name from  questions join users on  users.id = questions.asked\_by\_id  where  questions.answer\_text is  null and questions.expert\_id = ?', [user['id']])      all\_questions = question\_cur.fetchall()      return render\_template('unanswered.html', user=user, questions = all\_questions)   1. answering the unanswered question by the expert in the answer.html page.   @app.route('/answer/<question\_id>' , methods = ["POST", "GET"])  def answer(question\_id):      user = get\_current\_user()      db = get\_db()      if request.method == 'POST':          # store the answer given by the expert into the database by doing an update query.          db.execute('update questions set answer\_text = ? where id = ?', [request.form['answer'], question\_id])          db.commit()          return redirect(url\_for('unanswered'))      # get the question asked by the student and show it in the answer.html page to answer.      question\_cur =  db.execute('select id,  question\_text from questions where id = ?',[question\_id])      question = question\_cur.fetchone()      return render\_template('answer.html', user=user, question=question)        # now show this question\_asked in the answer.html page.   1. Question route to see the question and its answers, person asked and experts name, when any one clicks on the question.   @app.route('/question<question\_id>')  def question(question\_id):      user = get\_current\_user()      db = get\_db()      # query to show all the ansered questions and who asked them and who answered them.      question\_cur = db.execute('select questions.question\_text, questions.answer\_text,  askers.name as asker\_name, experts.name as expert\_name from questions join users as askers  on askers.id= questions.asked\_by\_id join users as experts on  experts.id = questions.expert\_id where questions.id = ?', [question\_id] )      question = question\_cur.fetchone()      # pass this question into the question.html template      return render\_template('question.html', user=user, question = question)   1. Finally showing all the questions and answers asked by the users and answered by the experts name in the home page itself.   @app.route('/')  def index():      user = get\_current\_user()      # fquery to show all the questions and answers in the home page.      db = get\_db()      question\_cur = db.execute('select questions.id as question\_id, questions.question\_text, askers.name as asker\_name, experts.name as expert\_name from questions join users as askers  on askers.id= questions.asked\_by\_id join users as experts on  experts.id = questions.expert\_id where questions.answer\_text is not null' )      questions\_result = question\_cur.fetchall()      # now pass the questions\_result into the home page to view in home.html page      return render\_template('home.html', user=user, questions = questions\_result) | Html code for the appropriate routes  Ask.html    <body>    <form action = "/ask" method = "POST">    <textarea name = "question"></textarea>          <span class="help-block">Ask away!</span>    <select name = "expert" id="select">    {% for expert in all\_experts %}            <option value = "{{expert['id']}}">{{ expert['name'] }}</option>      {% endfor %}     </select>  <button type="submit" class="btn btn-primary">Submit</button>    </form>  </body></html>  Unanswered.html  Showing all the unanswered questions from the db, questions table  And displaying in the unanswered.html page. And creating a route for the html anchor tag to go the page to answer that question which gets clicked by the expert , a new html page will open with that question displayed at the top. And a text area below to answer the question.   <div class="list-group">    {% for question in questions %}      <a href="{{ url\_for('answer', question\_id=question['id'])}}">      <h4>{{ question['question\_text'] }}</h4>      <p>Asked by: {{ question['name'] }}</p>      </a>  {% endfor %}  </div>  Answer.html    <div class="page-header">          <h1>{{ question['question\_text'] }}</h1> # this shows the question    </div>  # form to write the answer by the expert user.  <form action = "{{url\_for('answer', question\_id=question['id']) }}" method = "POST">    <textarea name = "answer" id="textArea"></textarea>  <button type="submit">Submit Answer</button>  </form>  Question.html  <div class="jumbotron">    <h1>{{ question['question\_text']}}</h1>    <p>{{ question['answer\_text']}}</p>    <p><a>Asked By: {{question['asker\_name']}}</a></p>    <p><a>Answered By: {{ question['expert\_name']}}</a></p>  </div>  Navigation link setup for all the html pages.  <ul> <li><a href="{{ url\_for('index') }}">Home</a></li>  {% if not user %}              <li><a href="{{ url\_for('login') }}">Login</a></li>              <li><a href="{{ url\_for('register') }}">Register</a></li>    {% endif %}  {% if user and user['admin'] == 0 and user['expert'] == 0 %}            <li><a href="{{ url\_for('ask') }}">Ask Question</a></li>  {% endif %}  {% if user and user['expert'] == 1 %}              <li><a href="{{ url\_for('unanswered') }}">Answer Questions</a></li>  {% endif %}  {% if user and user['admin'] == 1 %}              <li><a href="{{ url\_for('users') }}">User Setup</a></li>  {% endif %}  {% if user %} <li><a href="{{ url\_for('logout') }}">Logout</a></li>{% endif %}  </ul>  Home.html  <div class="list-group">  {% for question in questions %}  <a href="{{ url\_for('question', question\_id=question['question\_id'])}}">  <h4>{{ question['question\_text']}}</h4>       <p>Asked by: - {{ question['asker\_name']}}.</p>        <p>Answered by: - {{ question['asker\_name']}}.</p>      </a>  {% endfor %}  </div> |

|  |  |
| --- | --- |
| 1. Checking for duplicate username in the database while registering and prompting user to choose different username if it already exists in the database.   @app.route('/register', methods=['GET', 'POST'])  def register():      user = get\_current\_user()      if request.method == 'POST':          db = get\_db()          # checking for same usersname already exists or not          existing\_user\_cur = db.execute('select id from users where name = ?', [request.form['name']])          existing\_user = existing\_user\_cur          if existing\_user:              return render\_template('register.html', user = user, error = 'Username already taken, Try different username.')  # we will show this error message in the register.html page. if username is already in db.          hashed\_password = generate\_password\_hash(request.form['password'], method='sha256')          db.execute('insert into users (name, password, expert, admin) values (?, ?, ?, ?)', [request.form['name'], hashed\_password, '0', '0'])          db.commit()          session['user'] = request.form['name']          return redirect(url\_for('index'))      return render\_template('register.html', user=user)  Showing the error message on the register.html page.  Register.html :- we will display the error message sent from register route in a if condition.  <fieldset>{% if error %} <div>{{ error }}</div> {% endif %}  <input type="text" name="name"> <input type="password" name="password">  <button type="submit">Register</button> </fieldset>   1. Protecting all the routes , for compulsory login into the application in order to use the pages.   Preventing from showing the users setup page if the no user Is logged in to the application.  @app.route('/answer/<question\_id>' , methods = ["POST", "GET"])  def answer(question\_id):      user = get\_current\_user()      # to answer the questions for experts they have to be logged in . or go to login page.      if not user:          return redirect(url\_for('login'))      if user['expert'] == 0:          return redirect(url\_for('index')) # only experts can answer the questions.      db = get\_db()      if request.method == 'POST':          # store the answer given by the expert into the database by doing an update query.          db.execute('update questions set answer\_text = ? where id = ?', [request.form['answer'], question\_id])          db.commit()          return redirect(url\_for('unanswered'))      question\_cur =  db.execute('select id,  question\_text from questions where id = ?',[question\_id])      question = question\_cur.fetchone()      return render\_template('answer.html', user=user, question=question)  @app.route('/ask',  methods = ["POST", "GET"])  def ask():      user = get\_current\_user()      if not user:          return redirect(url\_for('login'))     # only logged in user can ask question, else navigate to login page.      db = get\_db()      if request.method == 'POST':          # we will insert the question asked by the user and the expert persons name in the questions table.          db.execute('insert into questions (question\_text, asked\_by\_id, expert\_id) values (?, ?, ?)', [request.form['question'], user['id'], request.form['expert']])          db.commit()          return redirect(url\_for('index'))   # functions name for the first page.      # get all the expert users from the database and show them in ask.html page..      expert\_cur = db.execute('select \* from users where expert = 1')      all\_experts = expert\_cur.fetchall()      return render\_template('ask.html', user=user, all\_experts = all\_experts)  @app.route('/unanswered')  def unanswered():      user = get\_current\_user()      if not user:          return redirect(url\_for('login'))      if user['expert'] == 0:          return redirect(url\_for('index')) # only experts see unanswered questions.      # get all the questions which are unanwered, means answer\_text is null,  from the db, questions table.      db = get\_db()      question\_cur = db.execute('select questions.id, questions.question\_text, users.name from questions join users on  users.id =  questions.asked\_by\_id  where  questions.answer\_text is null and questions.expert\_id = ?', [user['id']])      all\_questions = question\_cur.fetchall()      return render\_template('unanswered.html', user=user, questions = all\_questions)  @app.route('/users')  def users():      user = get\_current\_user()      if not user:          return redirect(url\_for('login'))  #if user is not logged in it will go to login page.      if user['admin'] == 0:          return redirect(url\_for('index'))  #if user is not admin then no user-setup page.      db = get\_db()      users\_cur = db.execute('select id, name, expert, admin from users')      users\_results = users\_cur.fetchall()      return render\_template('users.html', user=user, users=users\_results) | only admins can use the promote page and make any one expert users  @app.route('/promote/<user\_id>')  def promote(user\_id):      user = get\_current\_user()      if not user:          return redirect(url\_for('login'))      if user['admin'] == 0:          return redirect(url\_for('index'))  # if he is not admin then no update setup page for him.      db = get\_db()      db.execute('update users set expert = 1 where id = ?', [user\_id])      db.commit()   1. Showing login failure message if username or password is incorrect in the login route , login() function :-   @app.route('/login', methods=['GET', 'POST'])  def login():      user = get\_current\_user()      error = None      if request.method == 'POST':          db = get\_db()          name = request.form['name']          password = request.form['password']          user\_cur = db.execute('select id, name, password from users where name = ?', [name])          user\_result = user\_cur.fetchone()          if user\_result:              if check\_password\_hash(user\_result['password'], password):                  session['user'] = user\_result['name']                  return redirect(url\_for('index'))              else:                  error = 'Username or password did not match. Try again.'  else :          error = 'Username or password did not match. Try again.'      return render\_template('login.html', user=user , error = error)  showing this error message in the html page on login.html (the login screen).  Login.html page  <fieldset>{% if error %} <div>{{ error }}</div> {% endif %}  <input type="text" name="name"> <input type="password" name="password">  <button type="submit">Register</button> </fieldset>   1. Adding link macro code and importing/ inheriting the whole navigation bar instead of writing in all the html pages   For this we will create a new html page called as show\_links.html and put the entire navigation inside this page  Show\_links.html  {% macro show\_links(user) %}  <ul class="nav navbar-nav">      <li class="active"><a href="{{ url\_for('index') }}">Home</a></li>      {% if not user %}      <li><a href="{{ url\_for('login') }}">Login</a></li>      <li><a href="{{ url\_for('register') }}">Register</a></li>      {% endif %}      {% if user and user['admin'] == 0 and user['expert'] == 0 %}      <li><a href="{{ url\_for('ask') }}">Ask Question</a></li>      {% endif %}      {% if user and user['expert'] == 1 %}      <li><a href="{{ url\_for('unanswered') }}">Answer Questions</a></li>      {% endif %}      {% if user and user['admin'] == 1 %}      <li><a href="{{ url\_for('users') }}">User Setup</a></li>      {% endif %}      {% if user %}      <li><a href="{{ url\_for('logout') }}">Logout</a></li>      {% endif %}    </ul>  {% endmacro %}  Now in all the html pages , import this file and in place of ul to ul , the whole navigation you can write the functions name in jinja code  Example : - register.html page  {% from "show\_links.html" import show\_links %}  <html> <body>      <nav class="navbar navbar-inverse navbar-fixed-top">        <div class="container">          <div class="navbar-header">            <a class="navbar-brand" href="#">Questions & Answers</a>          </div>          <div id="navbar" class="navbar-collapse collapse">                {{  show\_links(user)  }}          </div><!--/.nav-collapse -->        </div>      </nav>  Do this in all the html pages. |