**Project Report**

**“QuestionsAreWelcome – An online Question Answer portal“**

Summary:

QuestionsAreWelcome is a question-and-answer platform wherein questions are asked, answered and edited by the users. Here users can get answers of any questions in defined domain. Users can post their questions related to specific domains like Space, History, Literature, Places, World Affairs and Health and Fitness. Once approved, users who are expert in that domain can give the answers. A question once posted can be edited or deleted till it is not answered. An answer can be edited or deleted at any time. Every question/answer must go through an approver, who will approve or reject the question/answer. If approver finds that question is not of the domain specified, inappropriate or can hurt the sentiments of person, he/she has right to reject it with a justified reason. As soon as a question/answer is approved/rejected owner of the question/answer receives an email stating the status of it. Every user can like/dislike the question/answer. Users can give n number of answers to a question, but they can’t post same question or answer again. Admin can view dashboards and print the pdf of dashboards.

Roles supported:

**User:** Users need to register. Once registered, they can login to post a question. Users can view other users’ questions and answer them. They can edit/delete their questions and answers.

**Approver:** Approver can approve the question/answer or reject question/answer with rejection reason.

**Admin:** Admin is the owner/controller of the application. Admin can add approvers and view dashboards as well as download the pdf of dashboard.

Functionalities:

1. Admin:

* Admin Login
* Add Approver and its details
* View Dashboards and download the pdf of dashboard. HashMaps are used to view dashboards and AbstractPdfView class is used to created PDF.

1. Approver:

* Approver Login
* View posts
* Approve/Reject posts. An email is sent by the system once a question/answer is approved/rejected. JavaMail and SMTP is used to send email.
* If rejected, add rejection reason specific to post
* Update Profile

1. User:

* User Registration
* User Login
* Post a question
* Edit/Delete the question
* Answer the questions
* Edit/Delete the answers
* Like/Dislike any question/answer
* Update Profile

1. OneToOne, OneToMany and other associations are used to relate different Models.
2. Filter is used to prevent SQL injection.
3. Checks are made for duplicate users, questions and answers.
4. Front-end and back-end validations are performed for all user inputs.

Technology stack:

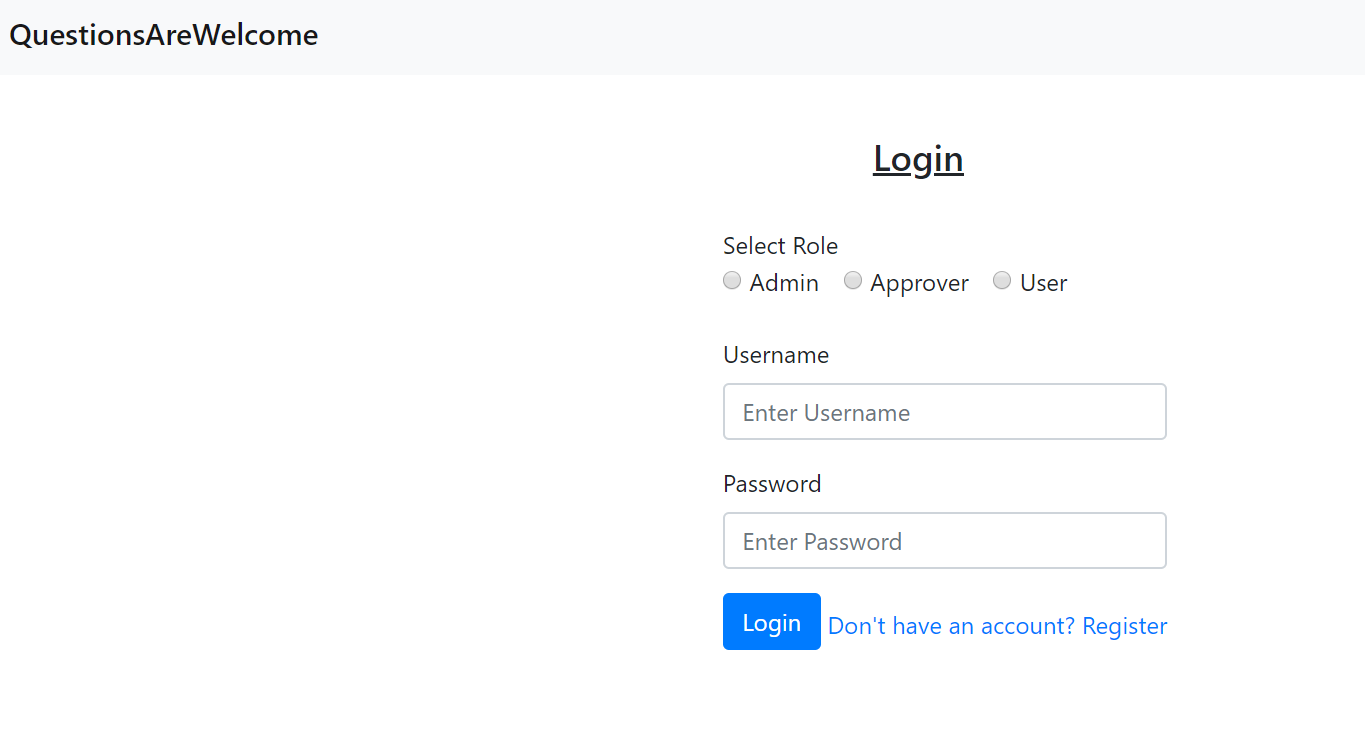
Technology used is Spring MVC and Hibernate.

**Front End (Views):** HTML5, CSS3, Bootstrap, JSP, Spring MVC Views, EL, JSTL and Custom Tags

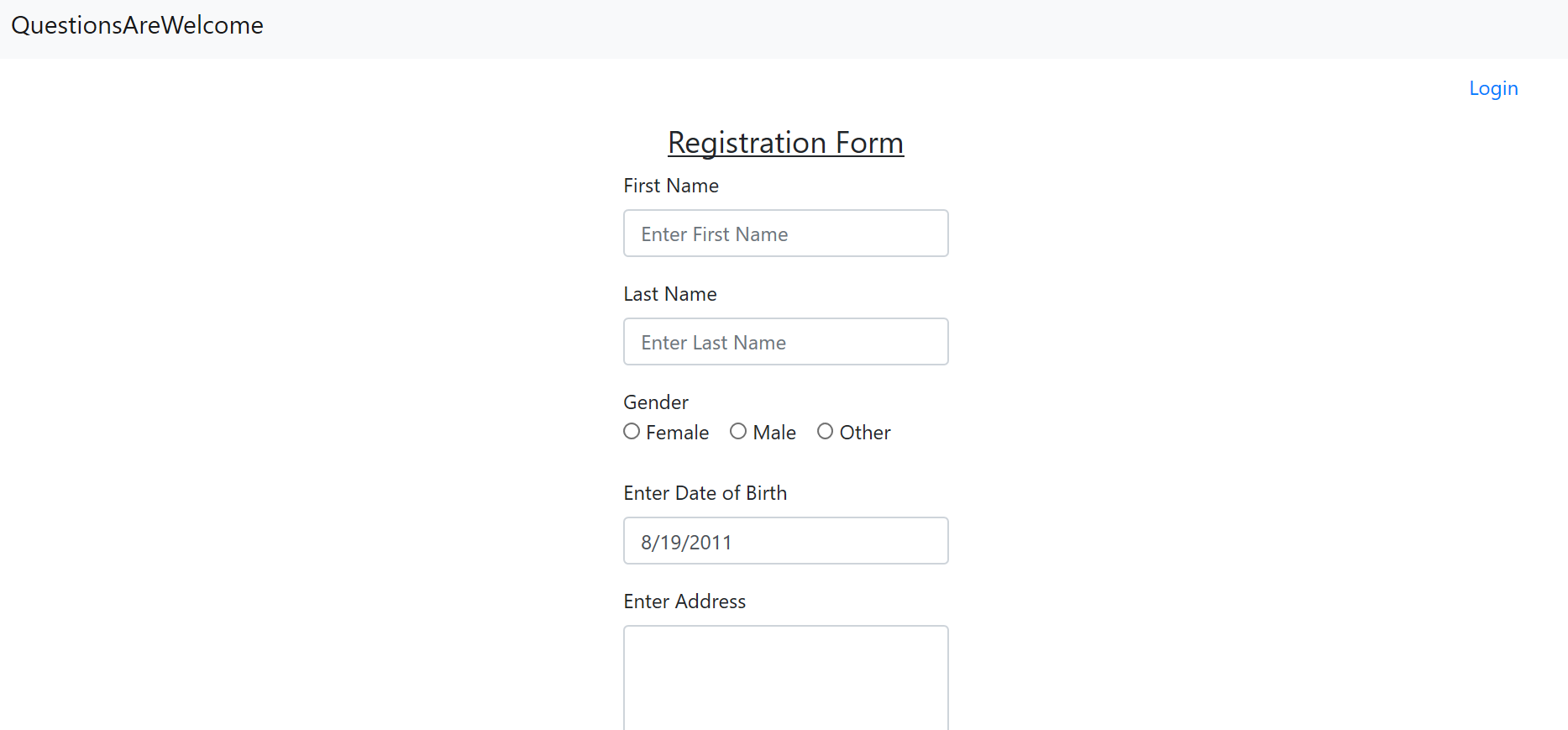
**Back End:** Spring MVC Controllers, Hibernate, Spring Criteria, Spring Annotations

**Database:** Relational Database – MySQL

Screenshots:

**Login Page**

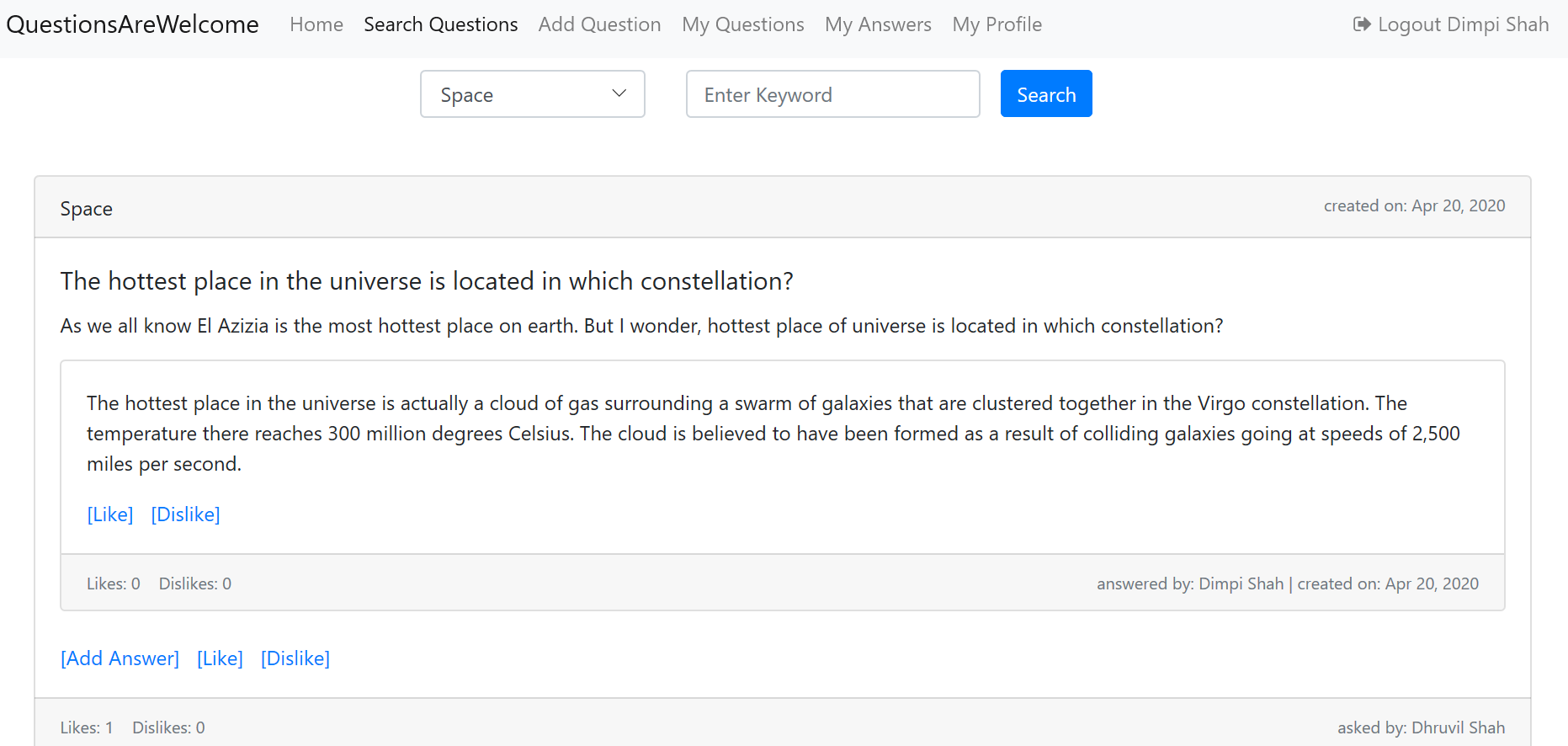
**User Registration**

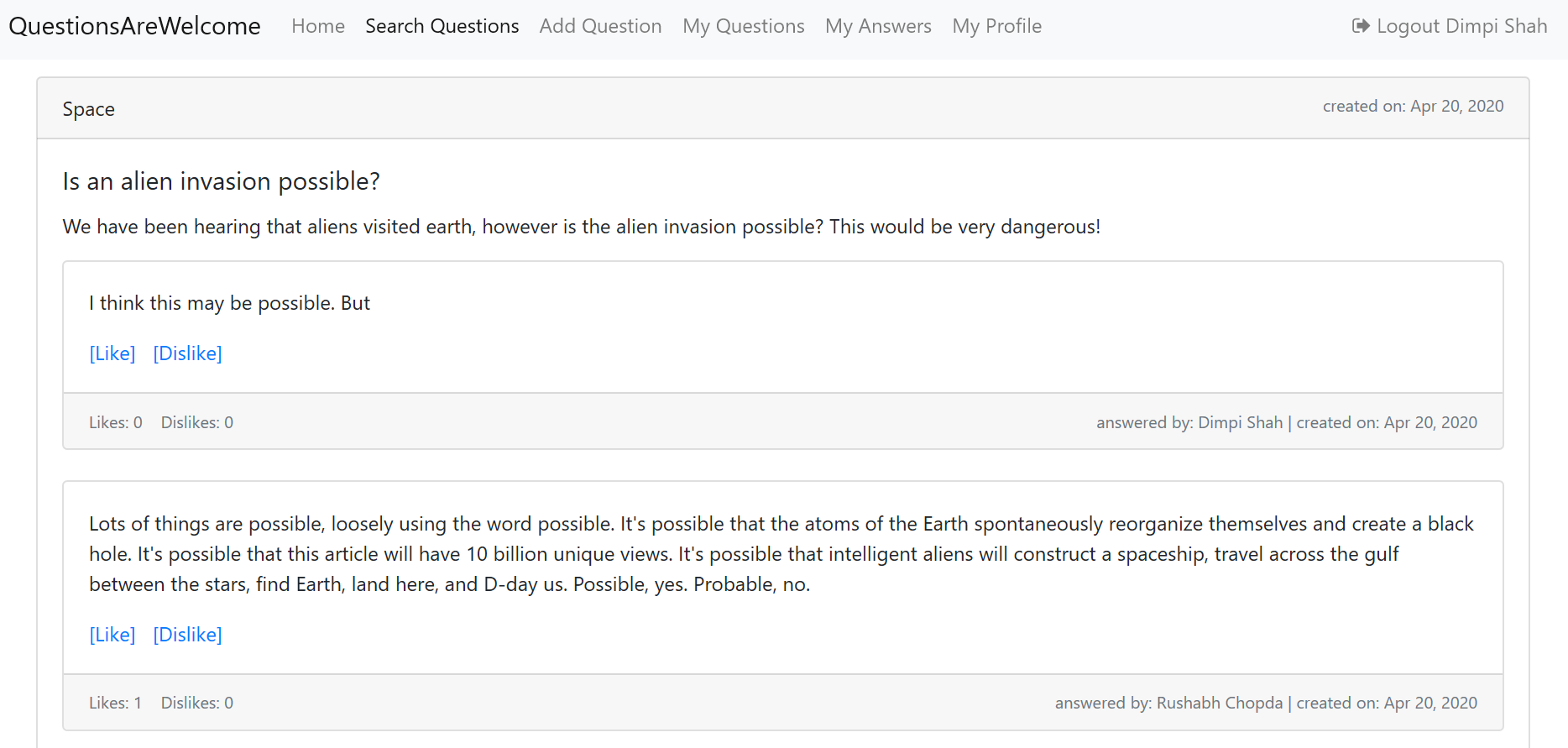


**User Home Page**

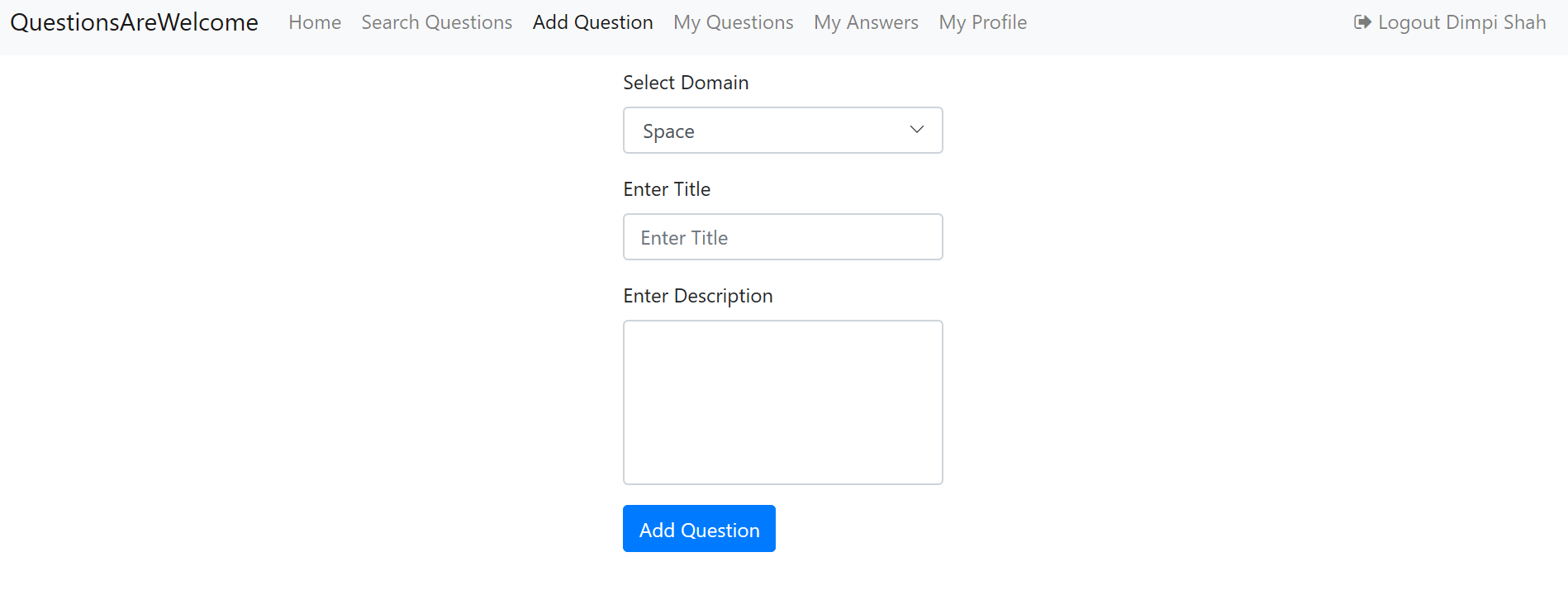


**User Search Questions Page**

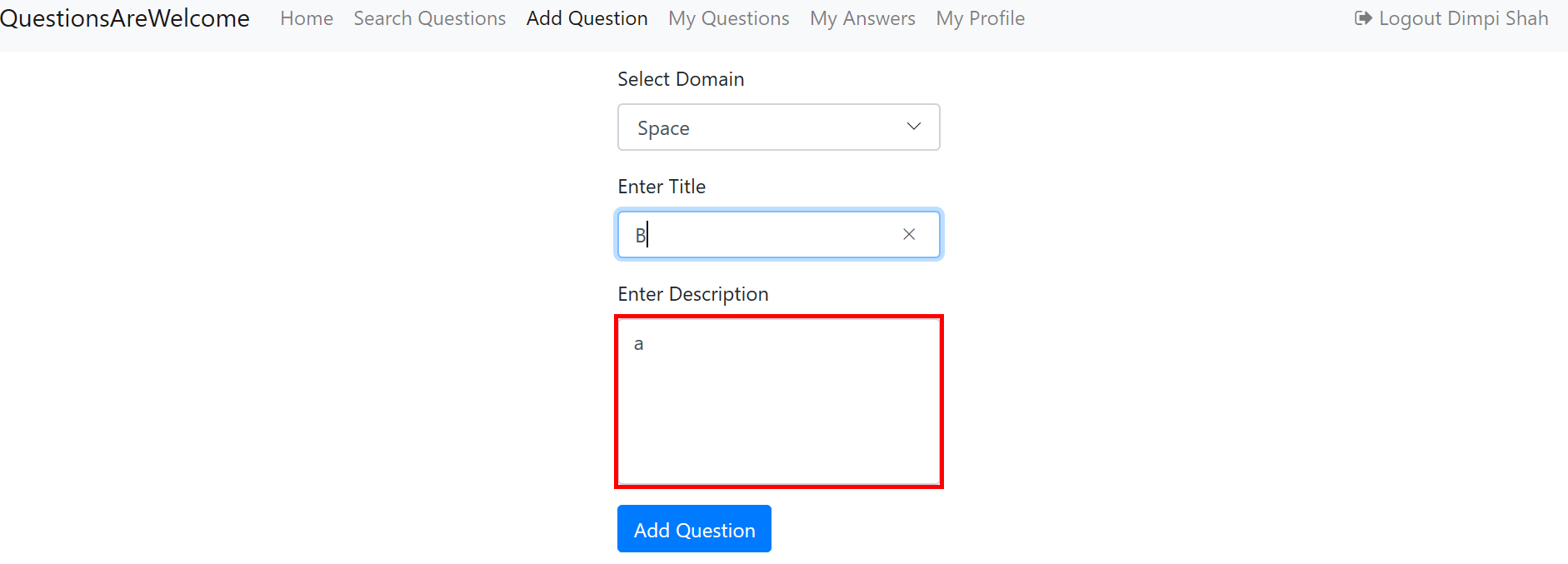




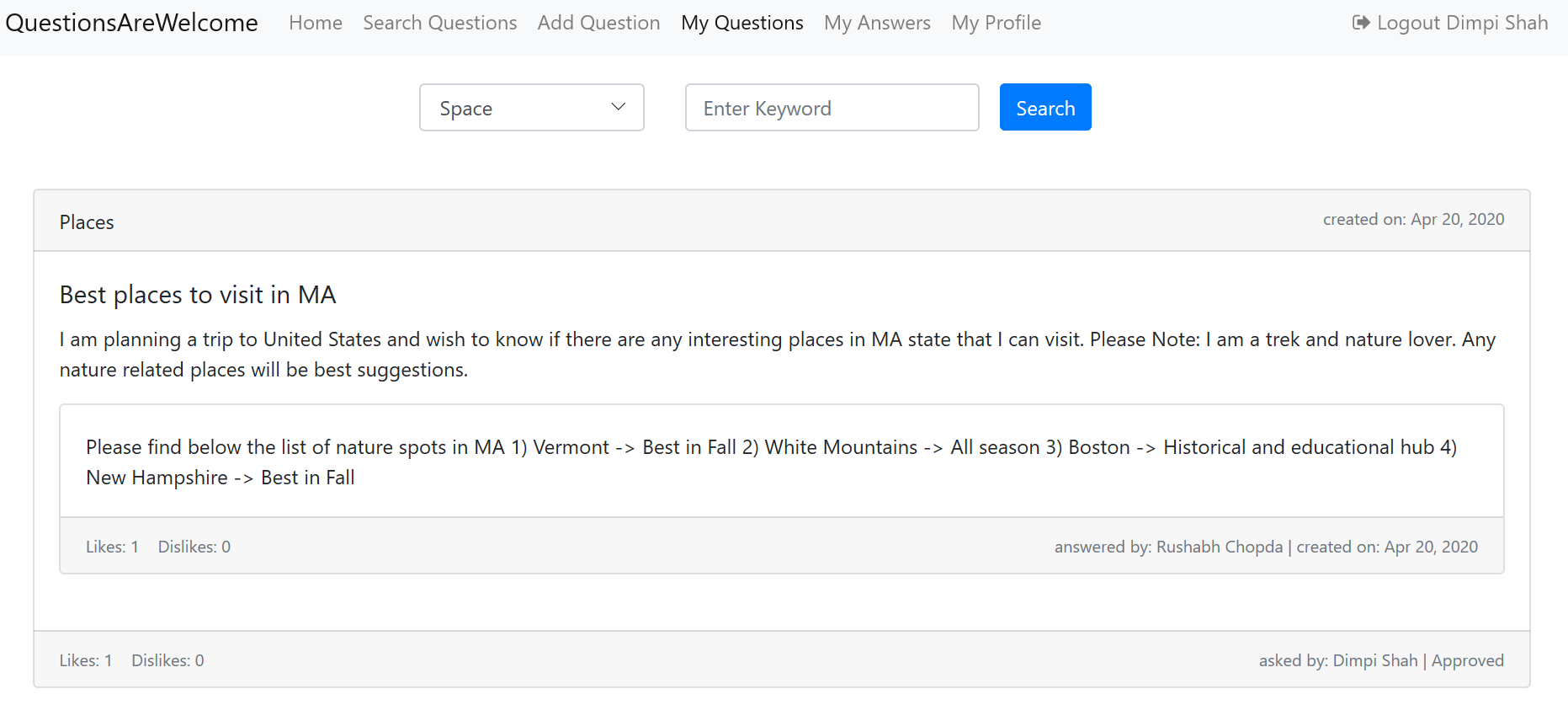
**User Add Question Page**

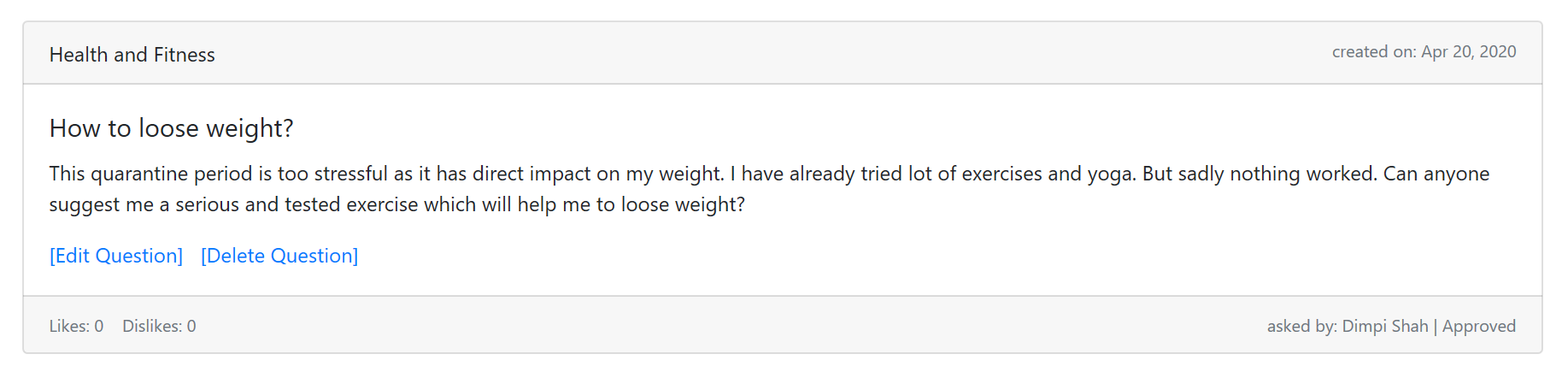


**User Add Question Validations**

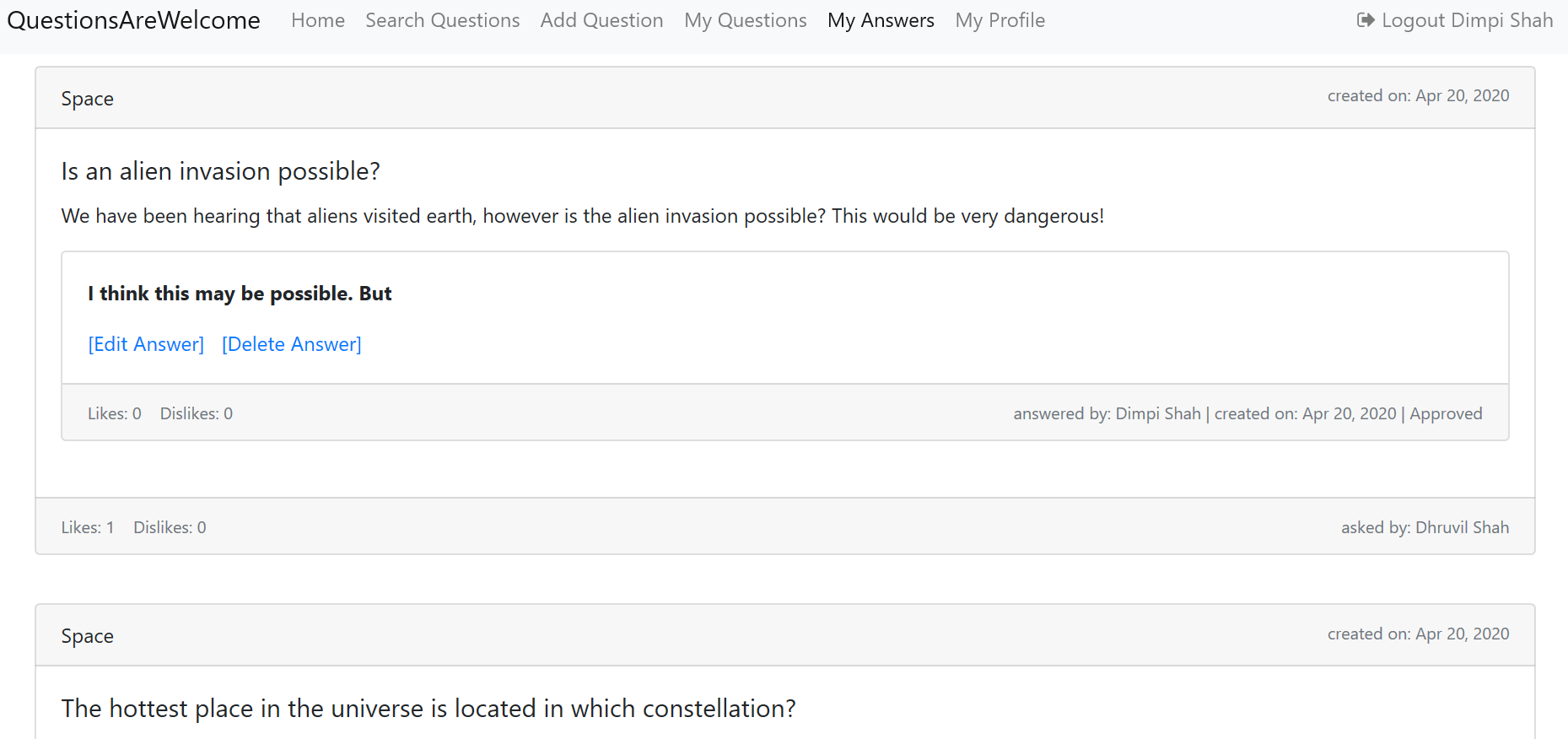


**User My Question List Page**

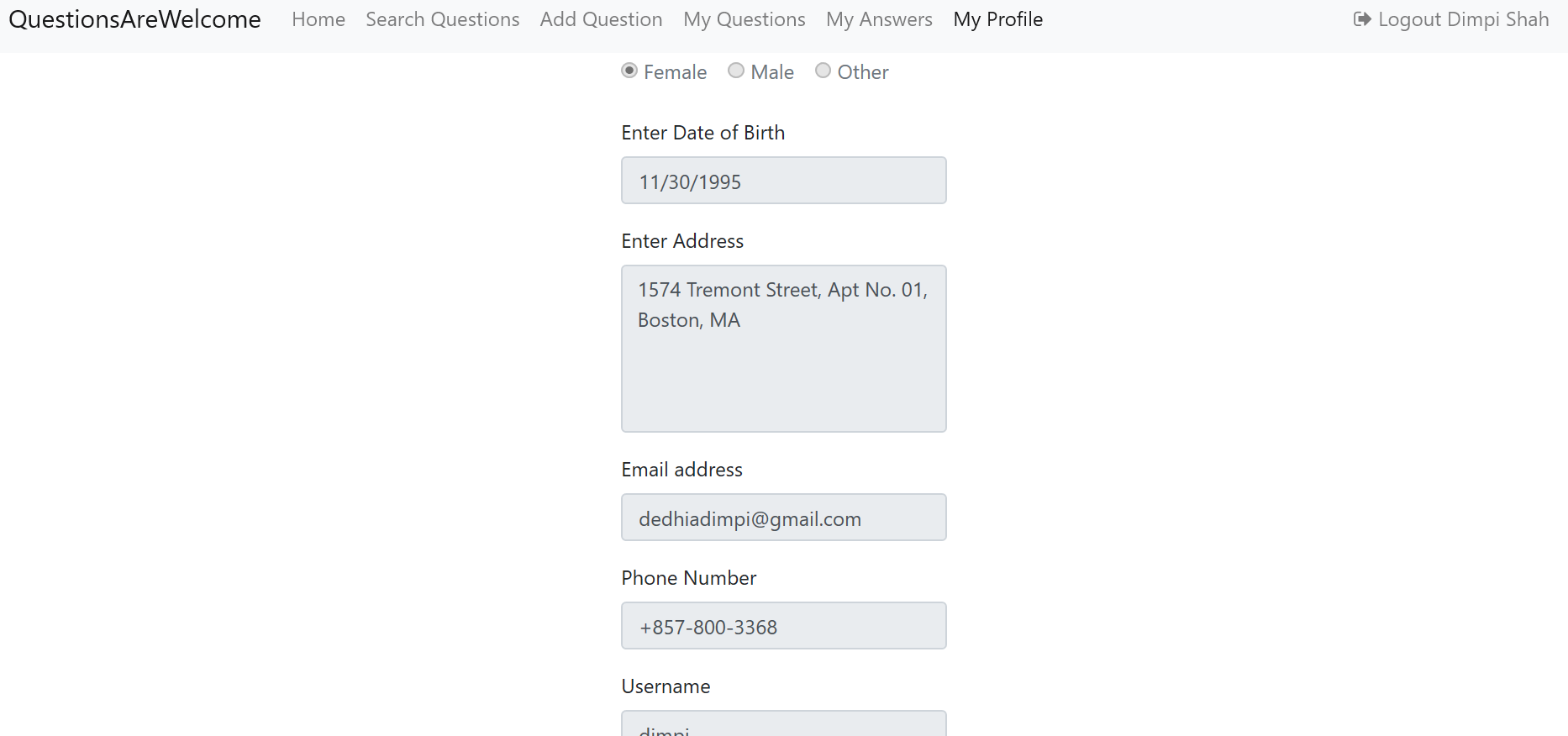


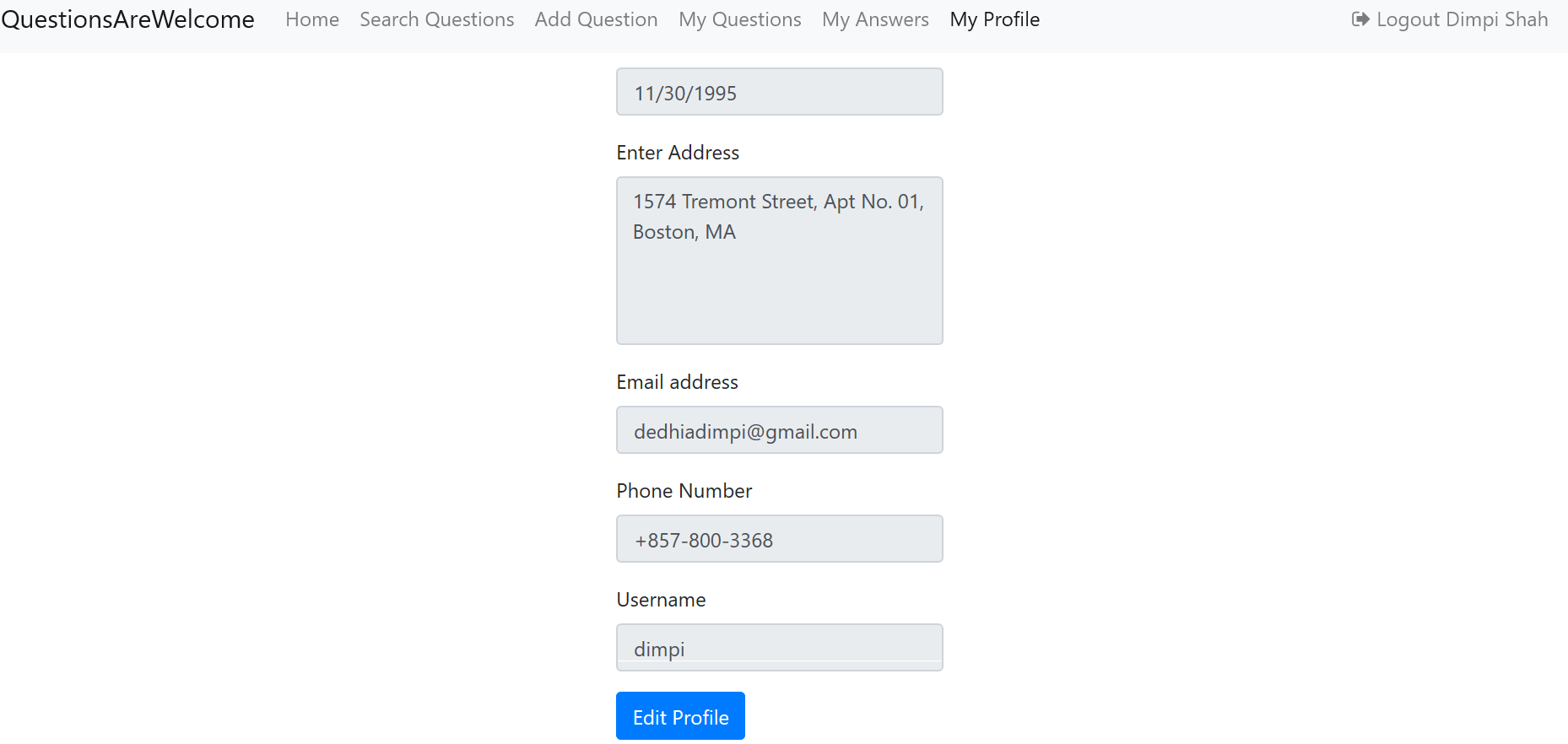


**User My Answer List Page**

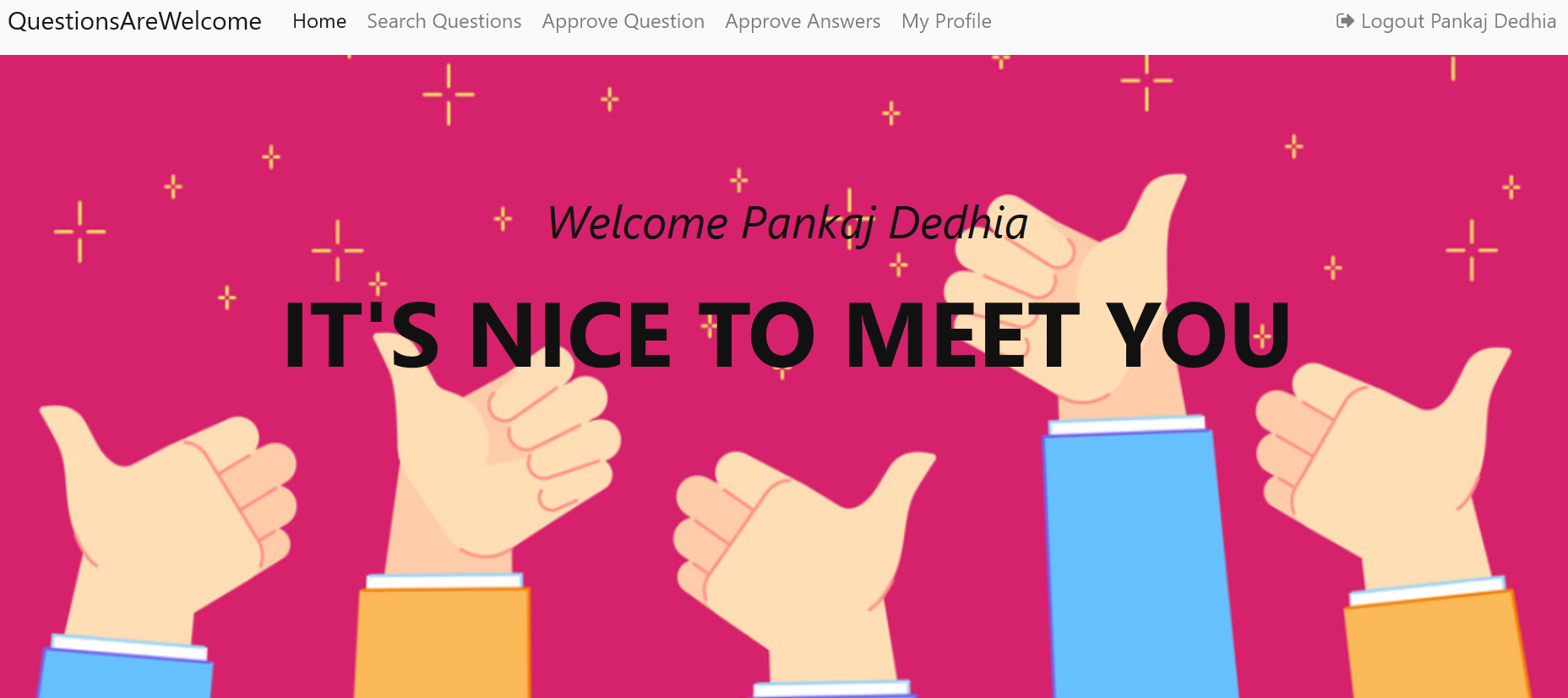


**User My Profile Page**

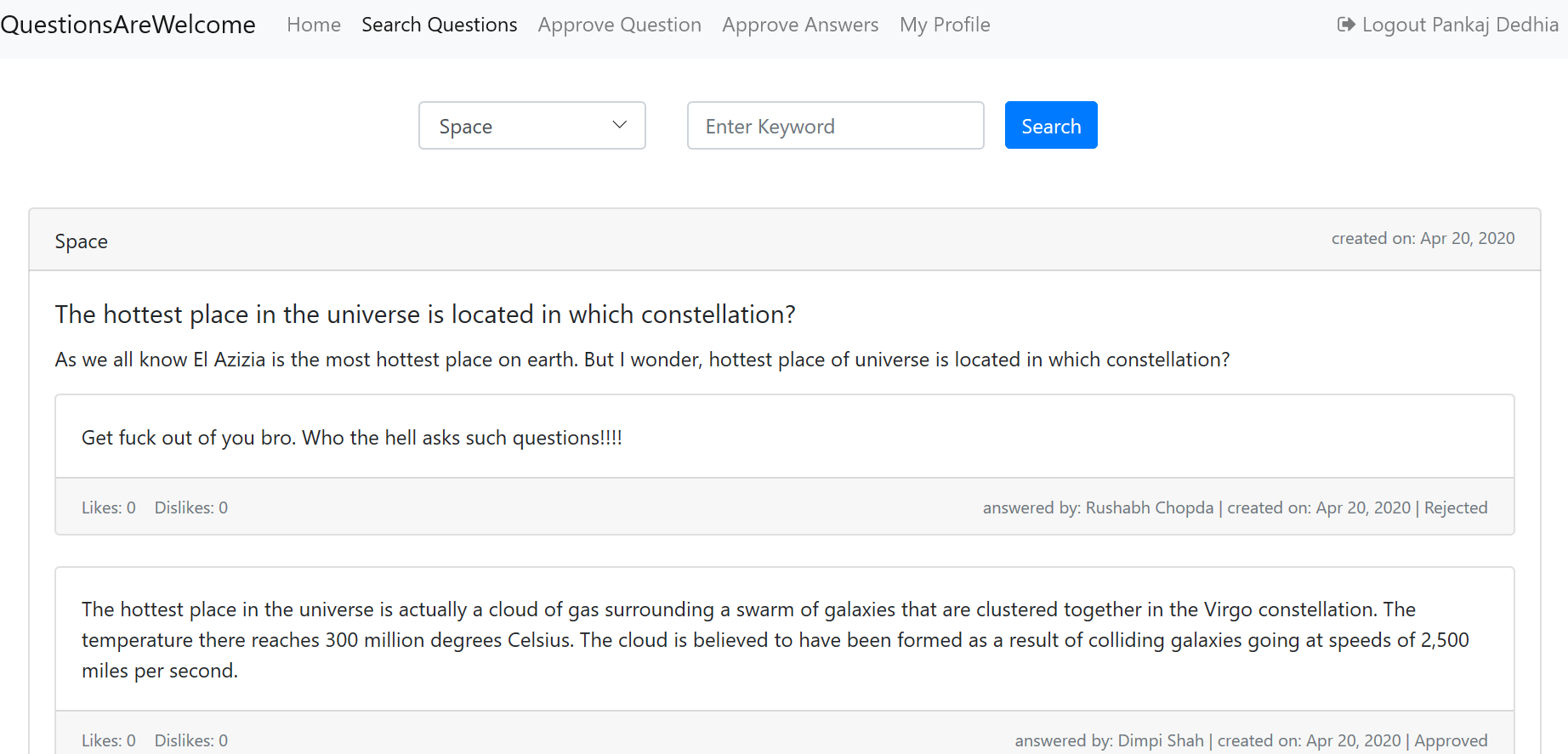


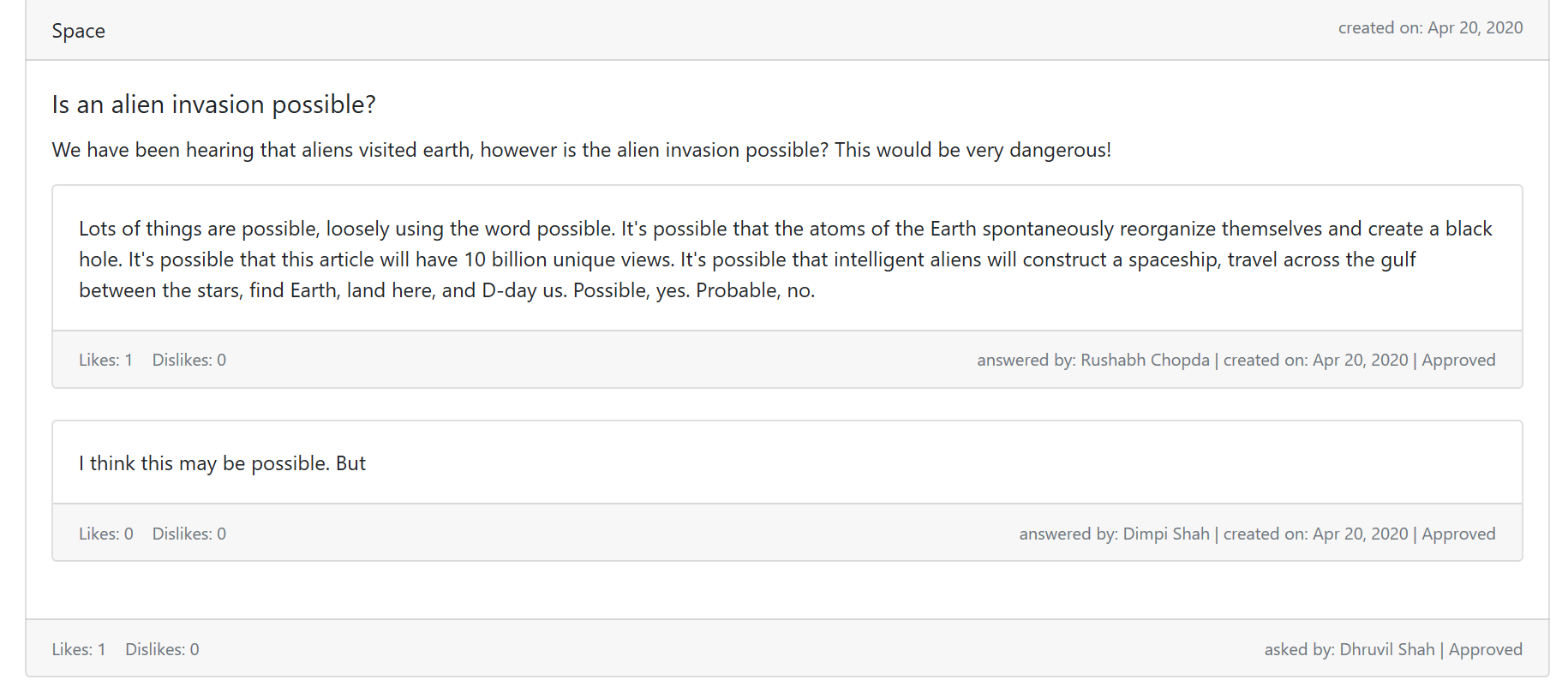


**Approver Home Page**

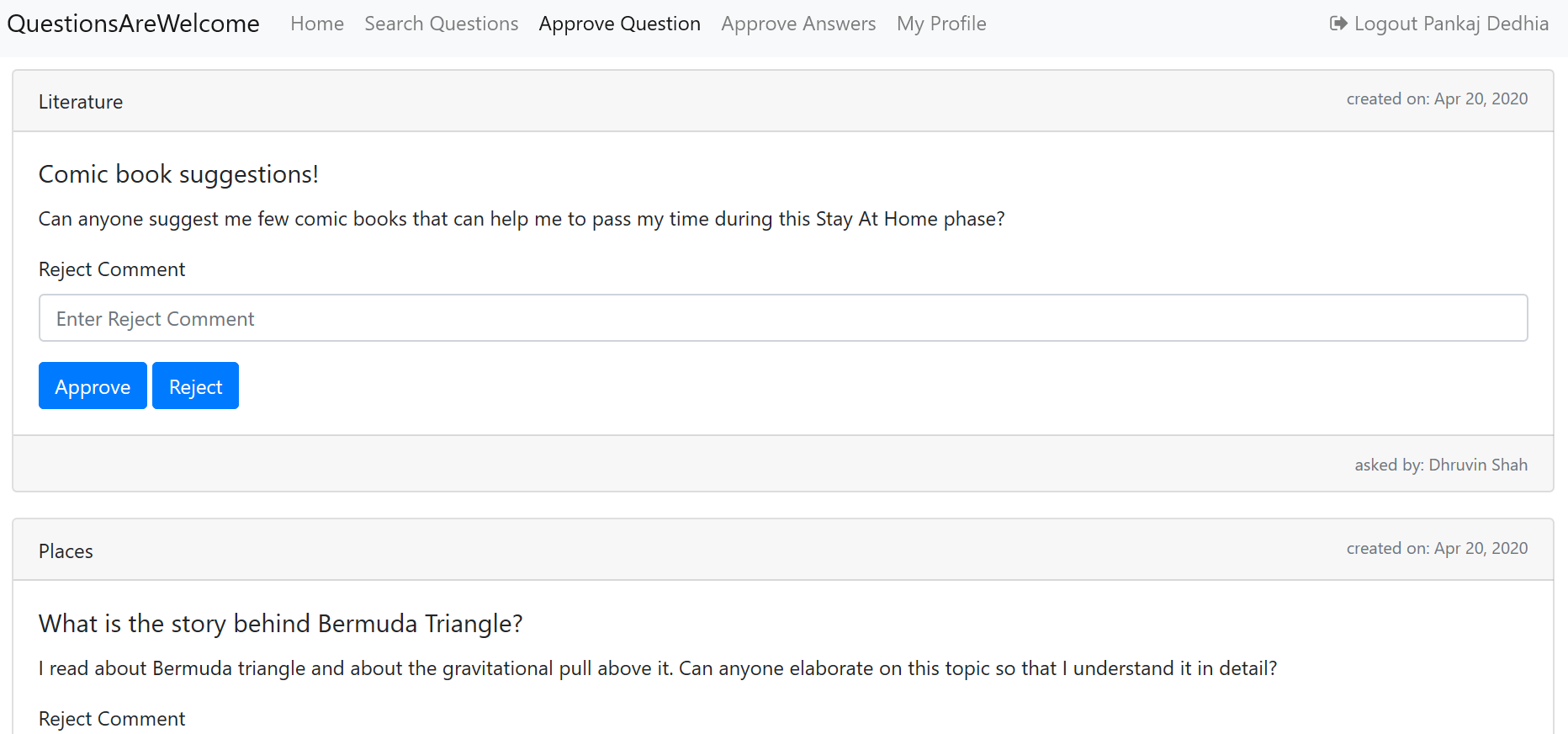


**Approver Question & Answer List Page**

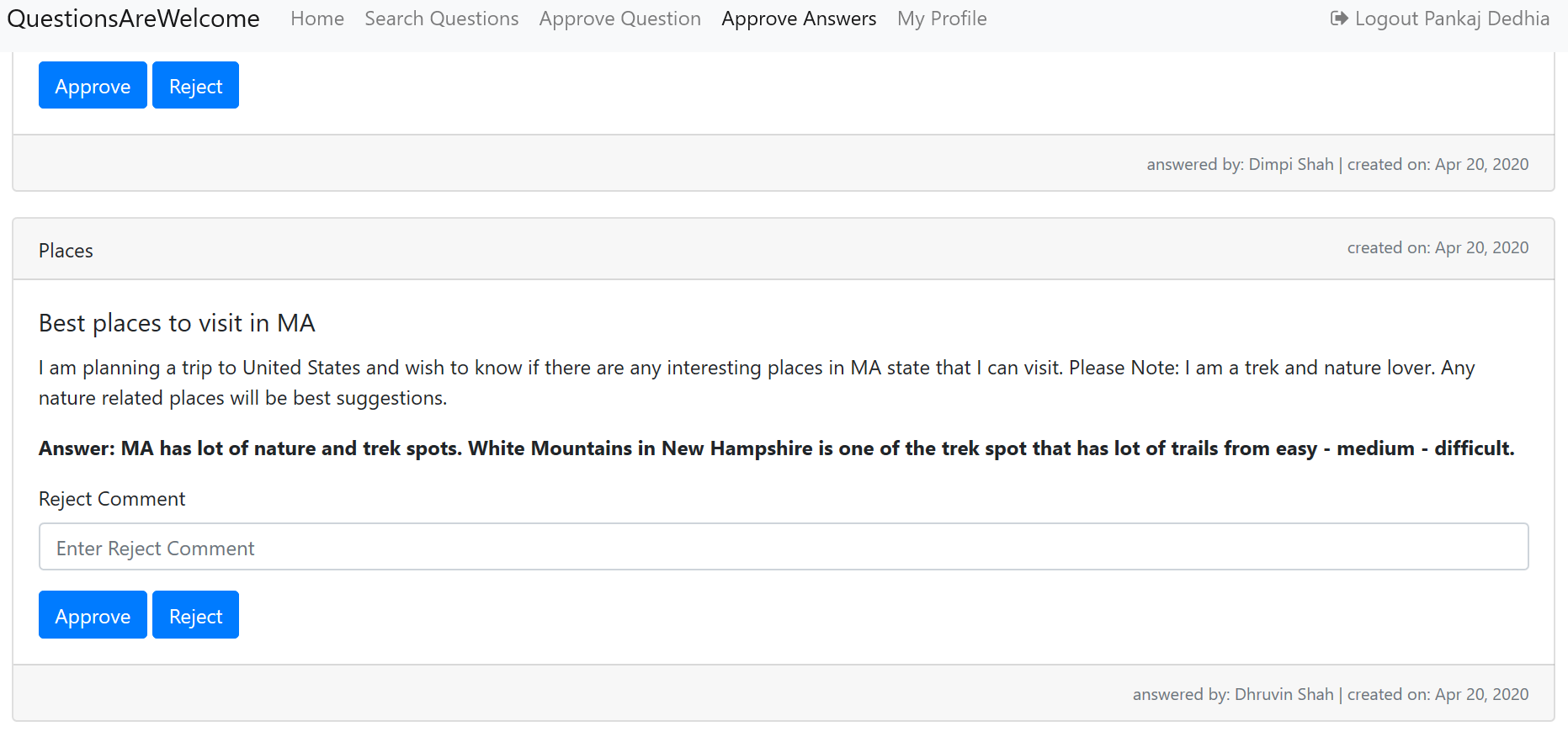




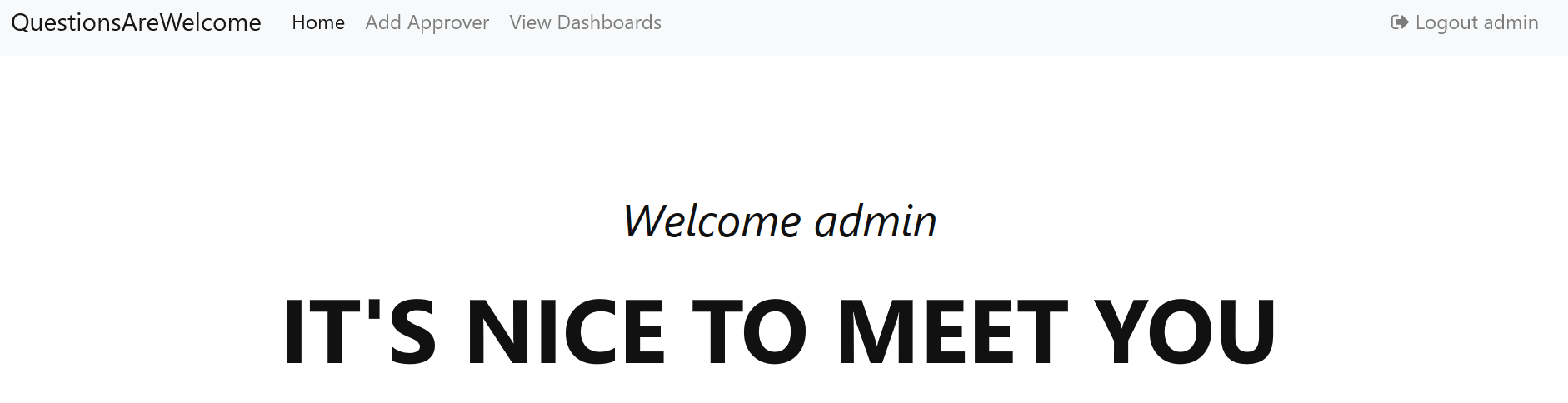
**Approver Approve Question Page**



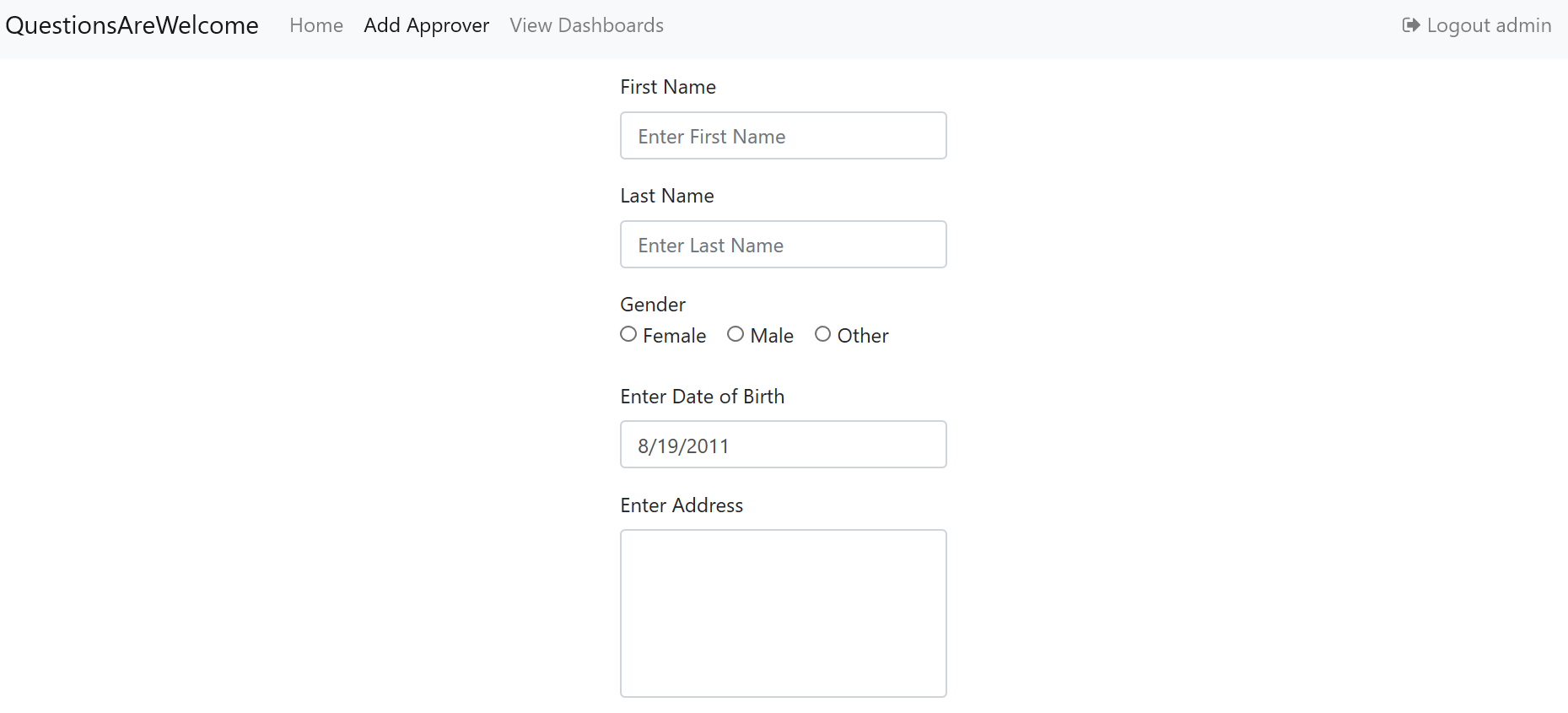
**Approver Approve Answer Page**

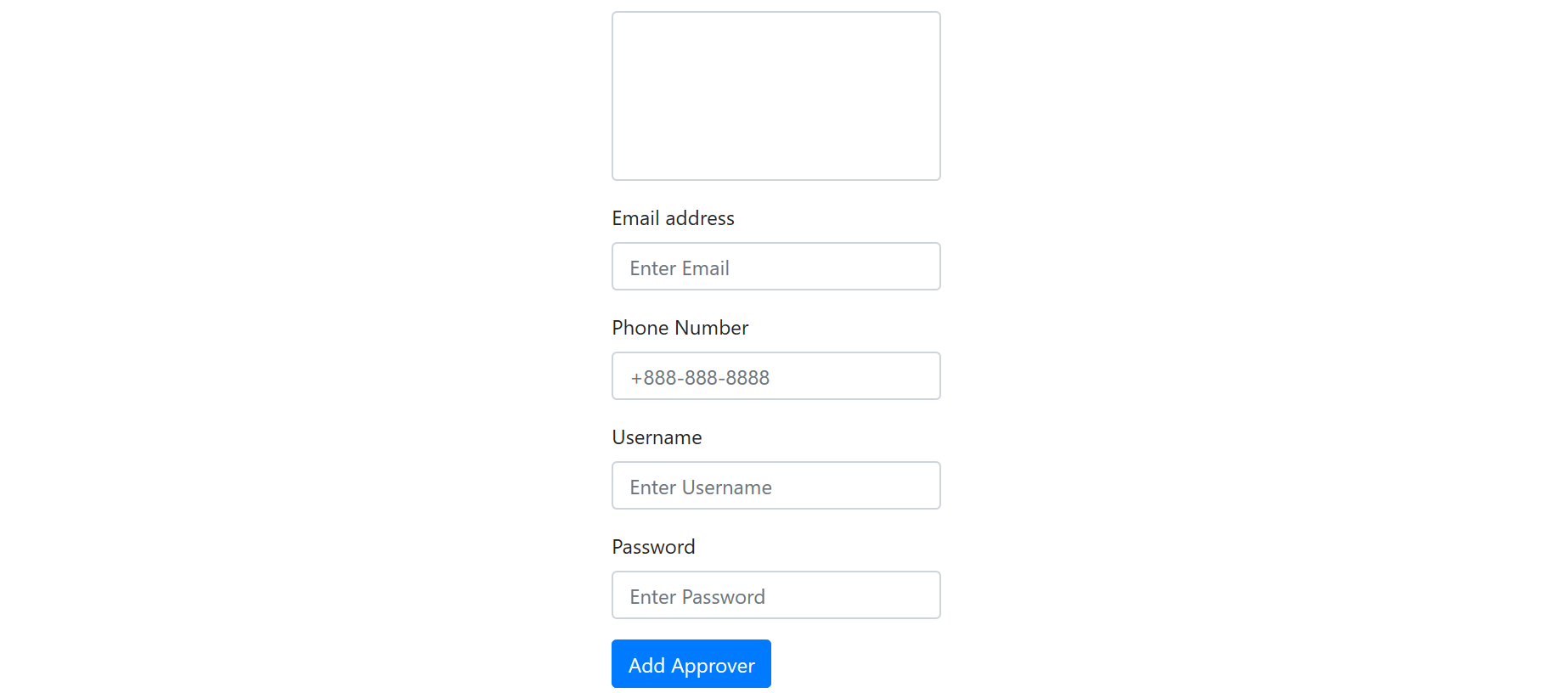


**Admin Home Page**

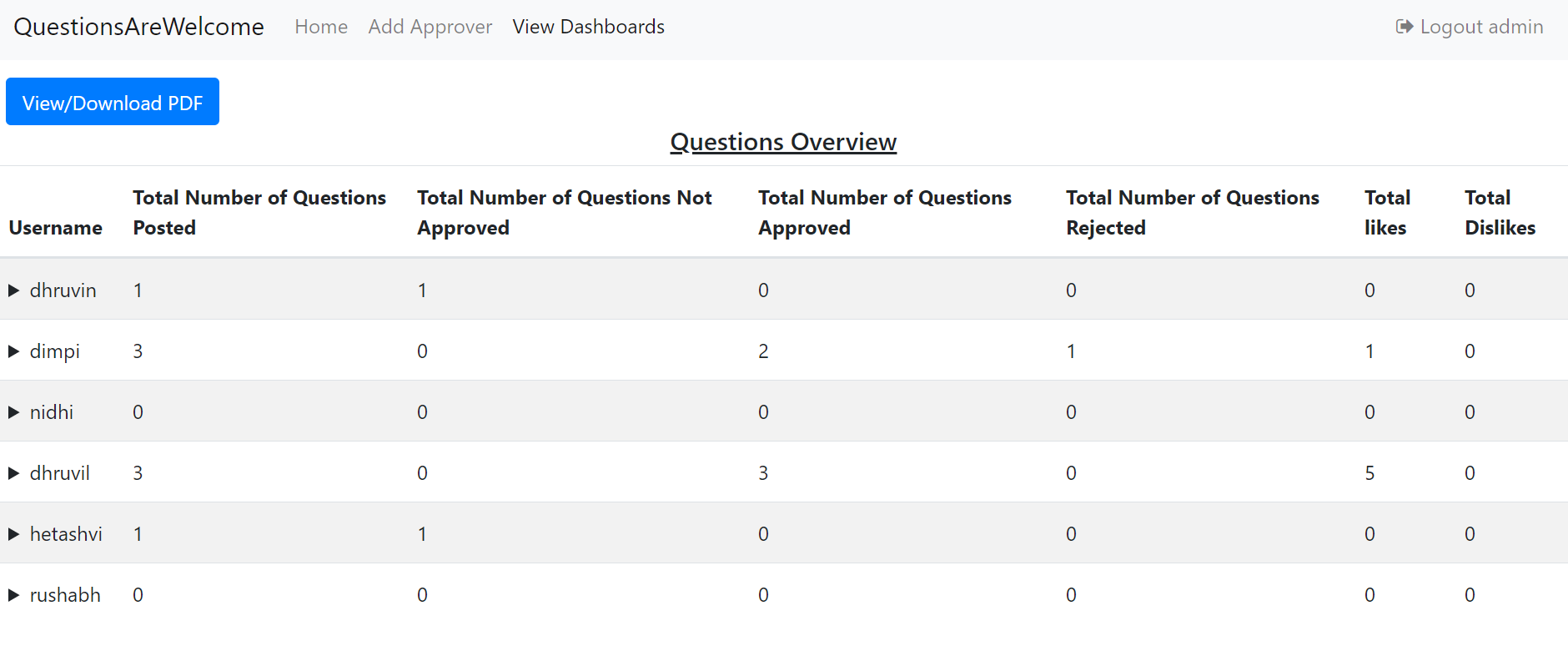


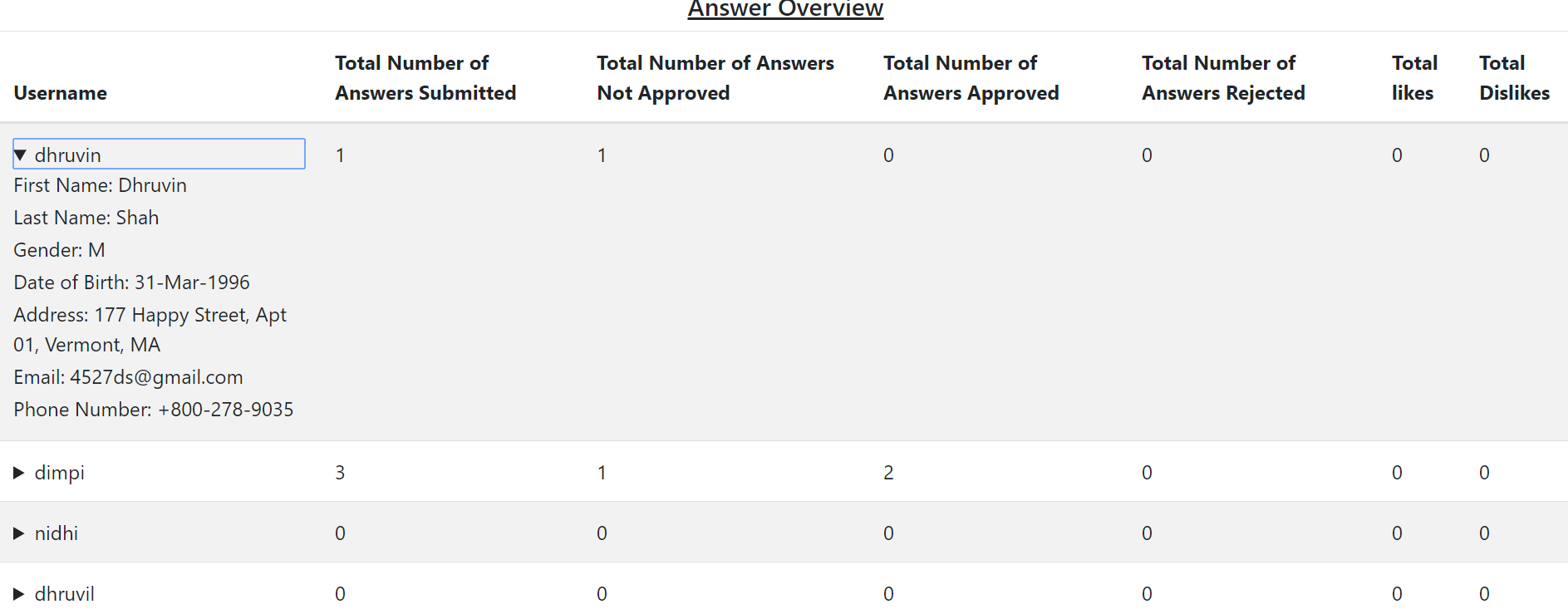
**Admin Add Approver Page**

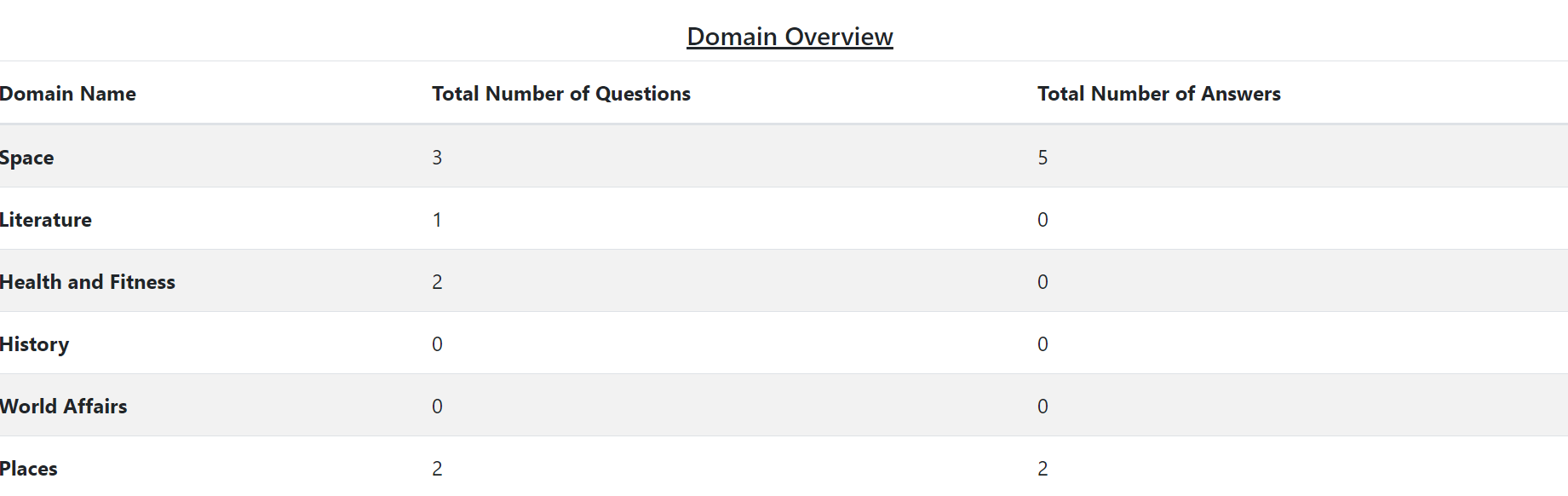




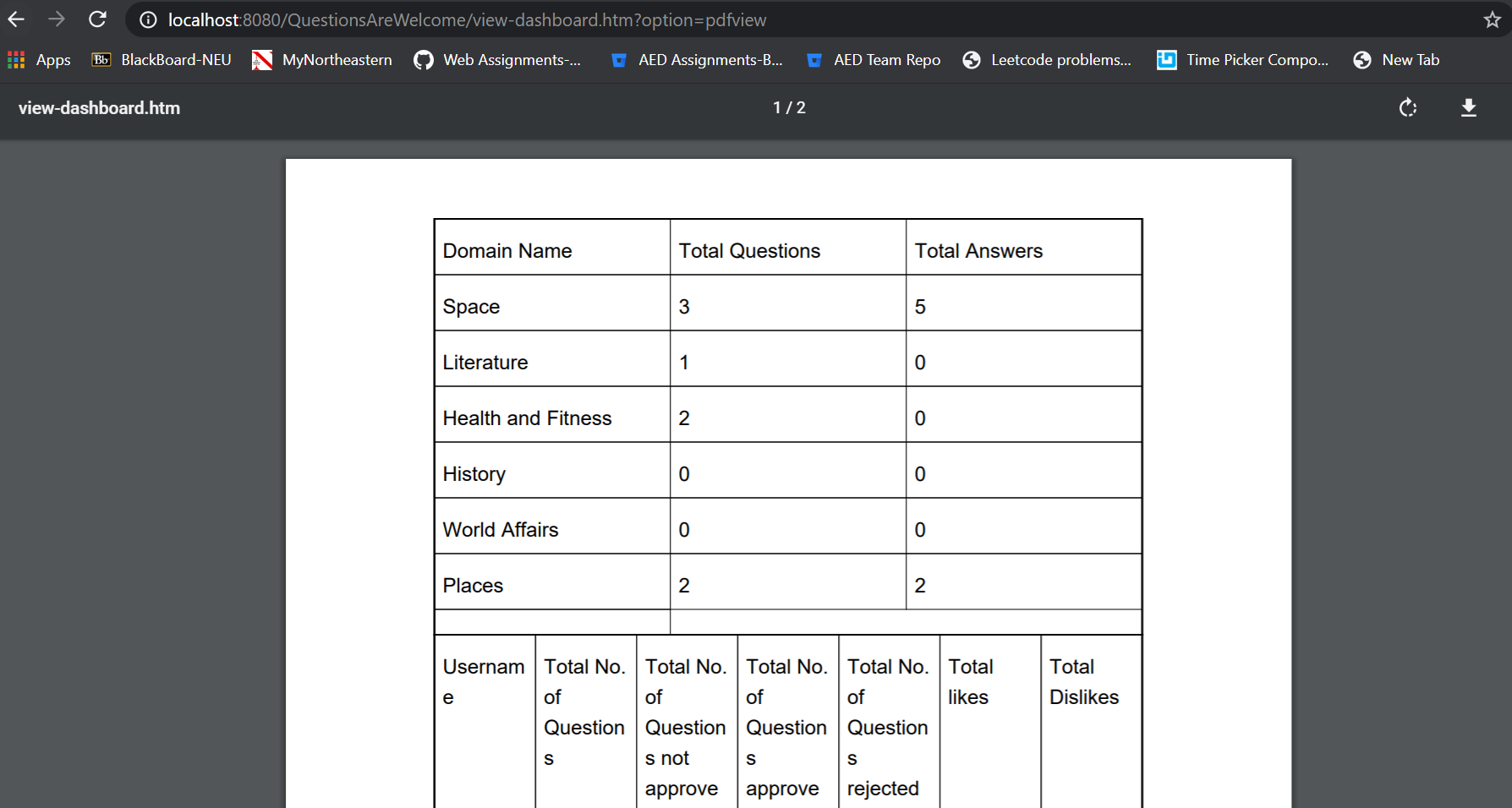
**Admin View Dashboard Page**







**Admin PDF View of Dashboard**



Controller Code:

1. **Login Controller**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controllers.LoginController;

import Helper.Validation;

import Models.Admin.Admin;

import Models.Approver.Approver;

import LoginDao.LoginDao;

import Models.User.User;

import java.io.PrintWriter;

import java.text.SimpleDateFormat;

import java.util.ArrayList;

import java.util.Date;

import java.util.Map;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.AbstractController;

/\*\*

\*

\* @author dedhi

\*/

public class LoginController extends AbstractController{

@Override

protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {

ModelAndView mv = null;

Validation validation = new Validation();

String option = request.getParameter("option") == null ? "" : request.getParameter("option");

if (option == null || option == "") {

mv = new ModelAndView("login");

return mv;

} else if(option.equals("login")){

String role = request.getParameter("role") == null? "none": request.getParameter("role");

String username = request.getParameter("username");

String password = request.getParameter("password");

LoginDao loginDao = new LoginDao();

if(role.equals("admin")){

ArrayList<Admin> admin = loginDao.getAdminLoginDetails(username, password);

if(admin.size() > 0){

request.getSession().setAttribute("user", admin.get(0));

request.getSession().setAttribute("role", "admin");

mv = new ModelAndView("admin-home", "role", admin.get(0));

return mv;

}else{

mv = new ModelAndView("login");

mv.addObject("error","User or Password is Invalid. Kindly Register if you don't have an account!");

return mv;

}

}else if(role.equals("approver")){

ArrayList<Approver> approver = loginDao.getApproverLoginDetails(username, password);

if(approver.size() > 0){

request.getSession().setAttribute("user", approver.get(0));

request.getSession().setAttribute("role", "approver");

mv = new ModelAndView("approver-home", "role", approver.get(0));

return mv;

}else{

mv = new ModelAndView("login");

mv.addObject("error","User or Password is Invalid. Kindly Register if you don't have an account!");

return mv;

}

}else if(role.equals("user")){

ArrayList<User> user = loginDao.getUserLoginDetails(username, password);

if(user.size() > 0){

request.getSession().setAttribute("user", user.get(0));

request.getSession().setAttribute("role", "user");

mv = new ModelAndView("user-home", "role", user.get(0));

return mv;

}else{

mv = new ModelAndView("login");

mv.addObject("error","User or Password is Invalid. Kindly Register if you don't have an account!");

return mv;

}

}else{

mv = new ModelAndView("login");

mv.addObject("error","Please specify a role!");

return mv;

}

}else if(option.equals("register")){

LoginDao loginDao = new LoginDao();

String firstname = request.getParameter("firstname");

String lastname = request.getParameter("lastname");

String gender = request.getParameter("gender") == null ? "" : request.getParameter("gender");

Date date = null;

String address = request.getParameter("address");

String email = request.getParameter("email");

String phone = request.getParameter("phone");

String username = request.getParameter("username");

String password = request.getParameter("password");

try{

date = new SimpleDateFormat("yyyy-MM-dd").parse(request.getParameter("dateofbirth").substring(0, 9));

}catch(Exception e){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter correct date!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new Date());

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

if(date.compareTo(new Date()) > 0){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter a date less than today's date!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

if(gender.equals("")){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter your Gender!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.onlyLettersValidation(firstname) || firstname.trim().length() < 0 || firstname.trim().length() > 50){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter valid First Name! First name cannot have more than 50 letters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.onlyLettersValidation(lastname) || lastname.trim().length() < 0 || lastname.trim().length() > 50){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter valid Last Name! Last name cannot have more than 50 letters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(address.trim().length() < 8 || address.trim().length() > 100){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter valid Address! Address cannot be less than 8 letters or more than 100 letters!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.phoneNumberValidation(phone) || phone.trim().length() > 20){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter valid Phone Number!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(username.trim().length() < 5 || username.trim().length() > 10){

mv = new ModelAndView("register-user");

mv.addObject("error","Username should be minimum of 5 characters and maximum of 10 characters!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(validation.usernameWithSpacesValidation(username)){

mv = new ModelAndView("register-user");

mv.addObject("error","Please enter valid Username!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.passwordPatternCorrect(password) || password.trim().length() < 6 || password.trim().length() > 10){

mv = new ModelAndView("register-user");

mv.addObject("error","Password should have 1 small-case letter, 1 Capital letter, 1 digit, 1 special character and the length should be between 6-10 characters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

int r = loginDao.checkUser(email, username);

if(r == 1){

mv = new ModelAndView("register-user");

mv.addObject("error","User already exist!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

int result = loginDao.registerUser(firstname, lastname, gender, date, address, email, phone, username, password);

if(result == 1){

mv = new ModelAndView("login","error","User registered successfully!");

return mv;

}else {

mv = new ModelAndView("register-user");

mv.addObject("error","Something went wrong. Please try again");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

}else if(option.equals("logout")){

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

request.getSession().invalidate();

mv = new ModelAndView("login");

return mv;

}else if(option.equals("addApprover")){

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

LoginDao loginDao = new LoginDao();

String firstname = request.getParameter("firstname");

String lastname = request.getParameter("lastname");

String gender = request.getParameter("gender");

Date date = new Date();

// Date date = new SimpleDateFormat("yyyy-MM-dd").parse(request.getParameter("dateofbirth").substring(0, 9));

String address = request.getParameter("address");

String email = request.getParameter("email");

String phone = request.getParameter("phone");

String username = request.getParameter("username");

String password = request.getParameter("password");

try{

date = new SimpleDateFormat("yyyy-MM-dd").parse(request.getParameter("dateofbirth").substring(0, 9));

}catch(Exception e){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter correct date!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new Date());

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

if(date.compareTo(new Date()) > 0){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter a date less than today's date!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

if(gender.equals("")){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter your Gender!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.onlyLettersValidation(firstname) || firstname.trim().length() < 0 || firstname.trim().length() > 50){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter valid First Name! First name cannot have more than 50 letters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.onlyLettersValidation(lastname) || lastname.trim().length() < 0 || lastname.trim().length() > 50){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter valid Last Name! Last name cannot have more than 50 letters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(address.trim().length() < 8 || address.trim().length() > 100){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter valid Address! Address cannot be less than 8 letters or more than 100 letters!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.phoneNumberValidation(phone) || phone.trim().length() > 20){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter valid Phone Number!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(username.trim().length() < 5 || username.trim().length() > 10){

mv = new ModelAndView("add-approver");

mv.addObject("error","Username should be minimum of 5 characters and maximum of 10 characters!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(validation.usernameWithSpacesValidation(username)){

mv = new ModelAndView("add-approver");

mv.addObject("error","Please enter valid Username!");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}else if(!validation.passwordPatternCorrect(password) || password.trim().length() < 6 || password.trim().length() > 10){

mv = new ModelAndView("add-approver");

mv.addObject("error","Password should have 1 small-case letter, 1 Capital letter, 1 digit, 1 special character and the length should be between 6-10 characters");

mv.addObject("firstname",firstname);

mv.addObject("lastname",lastname);

mv.addObject("date", new SimpleDateFormat("yyyy-MM-dd").format(date));

mv.addObject("address",address);

mv.addObject("email",email);

mv.addObject("phone",phone);

mv.addObject("username",username);

mv.addObject("password",password);

return mv;

}

int result = loginDao.addApprover(firstname, lastname, gender, date, address, email, phone, username, password);

if(result == 1){

mv = new ModelAndView("add-approver");

mv.addObject("firstname","");

mv.addObject("lastname","");

mv.addObject("date", new Date());

mv.addObject("address","");

mv.addObject("email","");

mv.addObject("phone","");

mv.addObject("username","");

mv.addObject("password","");

mv.addObject("error","Approved registered successfully!");

return mv;

}else {

mv = new ModelAndView("add-approver");

mv.addObject("firstname","");

mv.addObject("lastname","");

mv.addObject("date", new Date());

mv.addObject("address","");

mv.addObject("email","");

mv.addObject("phone","");

mv.addObject("username","");

mv.addObject("password","");

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}

return mv;

}

}

1. **Admin Controller**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controllers.AdminController;

import Dao.DashboardDao.DashboardDao;

import Models.User.User;

import java.util.ArrayList;

import java.util.Map;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.AbstractController;

/\*\*

\*

\* @author dedhi

\*/

public class AdminController extends AbstractController{

@Override

protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

DashboardDao dashboardDao = new DashboardDao();

Map<String,ArrayList<Object>> questionMap = dashboardDao.getUserDetailsWithQuestions();

Map<String,ArrayList<Object>> answerMap = dashboardDao.getUserDetailsWithAnswers();

Map<String, ArrayList<Integer>> domainMap = dashboardDao.getDomainDetails();

if(request.getParameter("option").equals("view")){

mv = new ModelAndView("view-dashboard");

mv.addObject("questionMap",questionMap);

mv.addObject("answerMap",answerMap);

mv.addObject("domainMap",domainMap);

return mv;

}if(request.getParameter("option").equals("pdfview")){

mv = new ModelAndView("pdfview-dashboard");

mv.addObject("questionMap",questionMap);

mv.addObject("answerMap",answerMap);

mv.addObject("domainMap",domainMap);

return mv;

}

return mv;

}

}

1. **Approver Controller**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controllers.ApproverController;

import Dao.ApproverDao.ApproverDao;

import Helper.ApproveEmail;

import Models.Answer.Answer;

import Models.Approver.Approver;

import Models.Question.Question;

import Models.User.User;

import java.io.PrintWriter;

import java.util.ArrayList;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.context.ConfigurableApplicationContext;

import org.springframework.context.support.ClassPathXmlApplicationContext;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.AbstractController;

/\*\*

\*

\* @author dedhi

\*/

public class ApproverController extends AbstractController{

@Override

protected ModelAndView handleRequestInternal(HttpServletRequest req, HttpServletResponse resp) throws Exception {

ModelAndView mv = null;

if(req.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

ApproverDao approverDao = new ApproverDao();

Approver approver = (Approver) req.getSession().getAttribute("user");

if(req.getParameter("option").equals("question")){

if(req.getParameter("action").equals("Approve")){

long question\_id = Long.parseLong(req.getParameter("question\_id"));

Question result = approverDao.approveQuestions(question\_id, approver,"");

ArrayList<Question> questionList = approverDao.getQuestionListForApproval();

mv = new ModelAndView("approve-questions");

mv.addObject("questionList",questionList);

if(result != null){

String configEmail = "email-bean.xml";

ConfigurableApplicationContext context = new ClassPathXmlApplicationContext(configEmail);

// @Service call

ApproveEmail approveEmail = (ApproveEmail) context.getBean("approveEmail");

String toAddr = result.getUser().getEmail();

String fromAddr = "dedhiadimpi@gmail.com";

// email subject

String subject = "Good news from QuestonsAreWelcome!";

// email body

String body = "Hi "+result.getUser().getFirstname()+" "+result.getUser().getLastname()+",\n\nYour Question '"+result.getQuestion\_title()+"' is Approved and Posted.";

approveEmail.sendEmail(toAddr, fromAddr, subject, body);

mv.addObject("error","Question Approved successfully!");

mv.addObject("reject","");

return mv;

}else{

mv.addObject("reject","");

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}else if(req.getParameter("action").equals("Reject")){

String reject\_comment = req.getParameter("reject\_comment");

if(reject\_comment == null || reject\_comment == "" || reject\_comment.trim().length() < 5 || reject\_comment.trim().length() > 4000){

ArrayList<Question> questionList = approverDao.getQuestionListForApproval();

mv = new ModelAndView("approve-questions");

mv.addObject("questionList",questionList);

mv.addObject("reject",reject\_comment);

mv.addObject("error","Please enter a valid rejection reason! Reject reason cannot be less than 5 or more than 4000 characters");

return mv;

}else{

long question\_id = Long.parseLong(req.getParameter("question\_id"));

Question result = approverDao.approveQuestions(question\_id, approver,reject\_comment);

ArrayList<Question> questionList = approverDao.getQuestionListForApproval();

mv = new ModelAndView("approve-questions");

mv.addObject("questionList",questionList);

if(result != null){

String configEmail = "email-bean.xml";

ConfigurableApplicationContext context = new ClassPathXmlApplicationContext(configEmail);

// @Service call

ApproveEmail approveEmail = (ApproveEmail) context.getBean("approveEmail");

String toAddr = result.getUser().getEmail();

String fromAddr = "dedhiadimpi@gmail.com";

// email subject

String subject = "Bad news from QuestonsAreWelcome!";

// email body

String body = "Hi "+result.getUser().getFirstname()+" "+result.getUser().getLastname()+",\n\nYour Question '"+result.getQuestion\_title()+"' is Rejected with following reason\n"+result.getReject\_comment();

approveEmail.sendEmail(toAddr, fromAddr, subject, body);

mv.addObject("reject","");

mv.addObject("error","Question Rejected successfully!");

return mv;

}else{

mv.addObject("reject",reject\_comment);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}

}

}else if(req.getParameter("option").equals("answer")){

if(req.getParameter("action").equals("Approve")){

long answer\_id = Long.parseLong(req.getParameter("answer\_id"));

Answer result = approverDao.approveAnswers(answer\_id, approver,"");

ArrayList<Answer> answerList = approverDao.getAnswerListForApproval();

mv = new ModelAndView("approve-answers");

mv.addObject("answerList",answerList);

mv.addObject("reject","");

if(result != null){

String configEmail = "email-bean.xml";

ConfigurableApplicationContext context = new ClassPathXmlApplicationContext(configEmail);

// @Service call

ApproveEmail approveEmail = (ApproveEmail) context.getBean("approveEmail");

String toAddr = result.getUser().getEmail();

String fromAddr = "dedhiadimpi@gmail.com";

// email subject

String subject = "Good news from QuestonsAreWelcome!";

// email body

String body = "Hi "+result.getUser().getFirstname()+" "+result.getUser().getLastname()+",\n\nYour Answer '"+result.getAnswer\_description()+"' for Question '"+result.getQuestion().getQuestion\_title()+"' is Approved and Posted";

approveEmail.sendEmail(toAddr, fromAddr, subject, body);

mv.addObject("error","Answer Approved successfully!");

return mv;

}else{

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}else if(req.getParameter("action").equals("Reject")){

String reject\_comment = req.getParameter("reject\_comment");

if(reject\_comment == null || reject\_comment == "" || reject\_comment.trim().length() < 5 || reject\_comment.trim().length() > 4000){

ArrayList<Answer> answerList = approverDao.getAnswerListForApproval();

mv = new ModelAndView("approve-answers");

mv.addObject("answerList",answerList);

mv.addObject("reject",reject\_comment);

mv.addObject("error","Please enter a rejection reason");

return mv;

}else{

long answer\_id = Long.parseLong(req.getParameter("answer\_id"));

Answer result = approverDao.approveAnswers(answer\_id, approver,reject\_comment);

ArrayList<Answer> answerList = approverDao.getAnswerListForApproval();

mv = new ModelAndView("approve-answers");

mv.addObject("answerList",answerList);

if(result != null){

String configEmail = "email-bean.xml";

ConfigurableApplicationContext context = new ClassPathXmlApplicationContext(configEmail);

// @Service call

ApproveEmail approveEmail = (ApproveEmail) context.getBean("approveEmail");

String toAddr = result.getUser().getEmail();

String fromAddr = "dedhiadimpi@gmail.com";

// email subject

String subject = "Bad news from QuestonsAreWelcome!";

// email body

String body = "Hi "+result.getUser().getFirstname()+" "+result.getUser().getLastname()+",\n\nYour Answer '"+result.getAnswer\_description()+"' for Question '"+result.getQuestion().getQuestion\_title()+"' is Rejected with following reason\n"+result.getReject\_comment();

approveEmail.sendEmail(toAddr, fromAddr, subject, body);

mv.addObject("reject","");

mv.addObject("error","Answer Rejected successfully!");

return mv;

}else{

mv.addObject("reject",reject\_comment);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}

}

}

return mv;

}

}

1. **User Controller**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controllers.UserController;

import Dao.AnswerDao.AnswerDao;

import Dao.ApproverDao.ApproverDao;

import Dao.DomainDao.DomainDao;

import Dao.LikesDao.LikesDao;

import Dao.QuestionDao.QuestionDao;

import Models.Answer.Answer;

import Models.Approver.Approver;

import Models.Domain.Domain;

import Models.Likes.Likes;

import Models.Question.Question;

import Models.User.User;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import java.util.HashSet;

import java.util.List;

import java.util.Set;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.AbstractController;

/\*\*

\*

\* @author dedhi

\*/

public class UserController extends AbstractController{

@Override

protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

String option = request.getParameter("option");

if(option.equals("addQuestion")){

Question question = new Question();

QuestionDao questionDao = new QuestionDao();

DomainDao domainDao = new DomainDao();

String title = request.getParameter("title").trim();

String desc = request.getParameter("desc").trim();

List<Question> qList = questionDao.checkQuestion(title,desc);

if(qList.size() > 0){

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("add-question");

mv.addObject("domainList",domainList);

mv.addObject("title",title);

mv.addObject("desc",desc);

mv.addObject("error","Question already exist!");

return mv;

}

if(title.length() < 5 || title.length() > 500 || desc.length() < 10 || desc.length() > 4000){

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("add-question");

mv.addObject("domainList",domainList);

mv.addObject("title",title);

mv.addObject("desc",desc);

mv.addObject("error","Title cannot be less than 5 and more than 500 characters! Description cannot be less than 10 or more than 4000 characters!");

return mv;

}

User user = (User) request.getSession().getAttribute("user");

question.setQuestion\_title(title.trim());

question.setQuestion\_description(desc.trim());

question.setDomain(domainDao.getDomain(Integer.parseInt(request.getParameter("domain"))));

question.setUser(user);

ApproverDao approverDao = new ApproverDao();

question.setApprover(approverDao.getDummyApprover());

question.setNo\_of\_likes(0);

question.setNo\_of\_dislikes(0);

question.setCreated\_on(new Date());

int result = questionDao.addQuestion(question);

if(result == 1){

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("add-question");

mv.addObject("domainList",domainList);

mv.addObject("title","");

mv.addObject("desc","");

mv.addObject("error","Your Question is sent for approval. You will receive an email once it's approved!");

return mv;

}else{

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("add-question");

mv.addObject("domainList",domainList);

mv.addObject("title",title);

mv.addObject("desc",desc);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}else if(option.equals("addAnswer")){

int question\_id =Integer.parseInt(request.getParameter("question\_id"));

QuestionDao questionDao = new QuestionDao();

AnswerDao answerDao = new AnswerDao();

Question question = questionDao.getQuestion(question\_id);

String desc = request.getParameter("desc").trim();

List<Answer> aList = answerDao.checkAnswer(request.getParameter("desc"));

if(aList.size() >0){

mv = new ModelAndView("add-answer");

mv.addObject("question", question);

mv.addObject("desc",desc);

mv.addObject("error","Answer already exist!");

return mv;

}

if(desc.length() < 10 || desc.length() > 4000){

mv = new ModelAndView("add-answer");

mv.addObject("question", question);

mv.addObject("desc",desc);

mv.addObject("error","Description cannot be less than 10 or more than 4000 characters!");

return mv;

}

Answer answer = new Answer();

answer.setAnswer\_description(request.getParameter("desc"));

answer.setUser((User) request.getSession().getAttribute("user"));

ApproverDao approverDao = new ApproverDao();

answer.setApprover(approverDao.getDummyApprover());

answer.setQuestion(question);

answer.setNo\_of\_likes(0);

answer.setNo\_of\_dislikes(0);

answer.setCreated\_on(new Date());

int result = answerDao.addAnswer(answer);

if(result == 1){

mv = new ModelAndView("add-answer");

mv.addObject("question", question);

mv.addObject("desc","");

mv.addObject("error","Your Answer is sent for approval. You will receive an email once its approved!");

return mv;

}else{

mv = new ModelAndView("add-answer");

mv.addObject("question", question);

mv.addObject("desc",desc);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}else if(option.equals("editAnswer")){

AnswerDao answerDao = new AnswerDao();

int r = 0;

String desc = request.getParameter("desc").trim();

List<Answer> aList = answerDao.checkAnswer(desc);

for(Answer a: aList){

if(a.getAnswer\_id() != Long.parseLong(request.getParameter("answer\_id"))){

r = 1;

break;

}

}

if(r == 1){

Answer answer = answerDao.getAnswer(Long.parseLong(request.getParameter("answer\_id")));

mv = new ModelAndView("edit-answer");

mv.addObject("answer\_id",Long.parseLong(request.getParameter("answer\_id")));

mv.addObject("answer\_desc",answer.getAnswer\_description());

mv.addObject("error","Answer already exist!");

return mv;

}

if(desc.length() < 10 || desc.length() > 4000){

Answer answer = answerDao.getAnswer(Long.parseLong(request.getParameter("answer\_id")));

mv = new ModelAndView("edit-answer");

mv.addObject("answer\_id",Long.parseLong(request.getParameter("answer\_id")));

mv.addObject("answer\_desc",answer.getAnswer\_description());

mv.addObject("error","Description cannot be less than 10 and greater than 4000 characters!");

return mv;

}

int result = answerDao.updateAnswer(Long.parseLong(request.getParameter("answer\_id")), request.getParameter("desc"));

User user = (User) request.getSession().getAttribute("user");

List<Answer> answers = answerDao.getMyAnswerList(user.getUser\_id());

HashMap<Question, ArrayList<Answer>> answerList = new HashMap<>();

for(Answer a : answers){

if(answerList.containsKey(a.getQuestion())){

answerList.get(a.getQuestion()).add(a);

}else{

ArrayList<Answer> ans = new ArrayList<Answer>();

ans.add(a);

answerList.put(a.getQuestion(), ans);

}

}

mv = new ModelAndView("view-my-answers");

mv.addObject("answerList",answerList);

if(result == 1){

mv.addObject("error","Your answer is sent for approval. You will receive an email once approved!");

}else{

mv.addObject("error","Something went wrong. Please try again later!");

}

return mv;

}else if(option.equals("editMyQuestion")){

String title = request.getParameter("title").trim();

String desc = request.getParameter("desc").trim();

QuestionDao questionDao = new QuestionDao();

List<Question> qList = questionDao.checkQuestion(title,desc);

int r = 0;

for(Question q: qList){

if(q.getQuestion\_id() != Long.parseLong(request.getParameter("question\_id"))){

r = 1;

break;

}

}

if( r == 1){

User user = (User) request.getSession().getAttribute("user");

Question question = questionDao.getQuestion(Long.parseLong(request.getParameter("question\_id")));

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("edit-question");

mv.addObject("question", question);

mv.addObject("domainList", domainList);

mv.addObject("error","Question already exist!");

return mv;

}

if(title.length() < 5 || title.length() > 500 || desc.length() < 10 || desc.length() > 4000){

Question question = questionDao.getQuestion(Long.parseLong(request.getParameter("question\_id")));

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("edit-question");

mv.addObject("question", question);

mv.addObject("domainList", domainList);

mv.addObject("error","Title cannot be less than 5 and more than 500 characters! Description cannot be less than 10 or more than 4000 characters");

return mv;

}

int result = questionDao.updateQuestion(Long.parseLong(request.getParameter("question\_id")),request.getParameter("title")

, request.getParameter("desc"), Long.parseLong(request.getParameter("domain")));

if(result == 1){

User user = (User) request.getSession().getAttribute("user");

List<Question> questionList = questionDao.getMyQuestionList(user.getUser\_id());

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "myQuestions");

mv.addObject("domainList",domainList);

// ApproverDao approverDao = new ApproverDao();

// Approver approver = approverDao.getDummyApprover();

// mv.addObject("approver\_id",approver.getApprover\_id());

mv.addObject("error","Your Question is sent for approval. You will receive an email once it's approved!");

return mv;

}else{

User user = (User) request.getSession().getAttribute("user");

Question question = questionDao.getQuestion(Long.parseLong(request.getParameter("question\_id")));

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("edit-question");

mv.addObject("question", question);

mv.addObject("domainList", domainList);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}else{

String select = request.getParameter("option");

User user = (User) request.getSession().getAttribute("user");

QuestionDao questionDao = new QuestionDao();

LikesDao likesDao = new LikesDao();

if(select.equals("like\_question")){

int result = 0;

long question\_id = Long.parseLong(request.getParameter("question\_id"));

Question question = questionDao.getQuestion(question\_id);

Likes like = likesDao.getLikeQuestion(question\_id, user.getUser\_id());

if(like == null){

like = new Likes();

like.setQuestion(question);

like.setUser(user);

like.setType("like");

int likes = question.getNo\_of\_likes();

question.setNo\_of\_likes(likes+1);

result = questionDao.addLikes(question);

}else if(like.getType().equals("dislike")){

like.setType("like");

int dislikes = question.getNo\_of\_dislikes();

question.setNo\_of\_dislikes(dislikes-1);

int likes = question.getNo\_of\_likes();

question.setNo\_of\_likes(likes+1);

result = questionDao.addLikes(question);

}

result = likesDao.addLikes(like);

if(result == 1){

List<Question> questionList = questionDao.getQuestionList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

return mv;

}else{

mv = new ModelAndView("login-fail");

return mv;

}

}else if(select.equals("dislike\_question")){

int result = 0;

long question\_id = Long.parseLong(request.getParameter("question\_id"));

Question question = questionDao.getQuestion(question\_id);

Likes like = likesDao.getLikeQuestion(question\_id, user.getUser\_id());

if(like == null){

like = new Likes();

like.setQuestion(question);

like.setUser(user);

like.setType("dislike");

int dislikes = question.getNo\_of\_dislikes();

question.setNo\_of\_dislikes(dislikes+1);

result = questionDao.addLikes(question);

}else if(like.getType().equals("like")){

like.setType("dislike");

int dislikes = question.getNo\_of\_dislikes();

question.setNo\_of\_dislikes(dislikes+1);

int likes = question.getNo\_of\_likes();

question.setNo\_of\_likes(likes-1);

result = questionDao.addLikes(question);

}

result = likesDao.addLikes(like);

if(result == 1){

List<Question> questionList = questionDao.getQuestionList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

return mv;

}else{

mv = new ModelAndView("login-fail");

return mv;

}

}else if(select.equals("like\_answer")){

int result = 0;

long answer\_id = Long.parseLong(request.getParameter("answer\_id"));

AnswerDao answerDao = new AnswerDao();

Answer answer = answerDao.getAnswer(answer\_id);

Likes like = likesDao.getLikeAnswer(answer\_id, user.getUser\_id());

if(like == null){

like = new Likes();

like.setAnswer(answer);

like.setUser(user);

like.setType("like");

int likes = answer.getNo\_of\_likes();

answer.setNo\_of\_likes(likes+1);

result = answerDao.addLikes(answer);

}else if(like.getType().equals("dislike")){

like.setType("like");

int dislikes = answer.getNo\_of\_dislikes();

answer.setNo\_of\_dislikes(dislikes-1);

int likes = answer.getNo\_of\_likes();

answer.setNo\_of\_likes(likes+1);

result = answerDao.addLikes(answer);

}

result = likesDao.addLikes(like);

if(result == 1){

List<Question> questionList = questionDao.getQuestionList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

return mv;

}else{

mv = new ModelAndView("login-fail");

return mv;

}

}else if(select.equals("dislike\_answer")){

int result = 0;

long answer\_id = Long.parseLong(request.getParameter("answer\_id"));

AnswerDao answerDao = new AnswerDao();

Answer answer = answerDao.getAnswer(answer\_id);

Likes like = likesDao.getLikeAnswer(answer\_id, user.getUser\_id());

if(like == null){

like = new Likes();

like.setAnswer(answer);

like.setUser(user);

like.setType("dislike");

int dislikes = answer.getNo\_of\_dislikes();

answer.setNo\_of\_dislikes(dislikes+1);

result = answerDao.addLikes(answer);

}else if(like.getType().equals("like")){

like.setType("dislike");

int dislikes = answer.getNo\_of\_dislikes();

answer.setNo\_of\_dislikes(dislikes+1);

int likes = answer.getNo\_of\_likes();

answer.setNo\_of\_likes(likes-1);

result = answerDao.addLikes(answer);

}

result = likesDao.addLikes(like);

if(result == 1){

List<Question> questionList = questionDao.getQuestionList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

return mv;

}else{

mv = new ModelAndView("login-fail");

return mv;

}

}

}

return mv;

}

}

1. **Redirect Controller**

/\*

\* To change this license header, choose License Headers in Project Properties.

\* To change this template file, choose Tools | Templates

\* and open the template in the editor.

\*/

package Controllers.RedirectController;

import Dao.AnswerDao.AnswerDao;

import Dao.ApproverDao.ApproverDao;

import Dao.DomainDao.DomainDao;

import LoginDao.LoginDao;

import Dao.QuestionDao.QuestionDao;

import Dao.UserDao.UserDao;

import Models.Answer.Answer;

import Models.Approver.Approver;

import Models.Domain.Domain;

import Models.Question.Question;

import Models.User.User;

import java.text.SimpleDateFormat;

import java.util.List;

import java.util.ArrayList;

import java.util.Date;

import java.util.HashMap;

import Helper.Validation;

import javax.servlet.http.HttpServlet;

import javax.servlet.http.HttpServletRequest;

import javax.servlet.http.HttpServletResponse;

import org.springframework.stereotype.Controller;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RequestMethod;

import org.springframework.web.bind.annotation.RequestParam;

import org.springframework.web.servlet.ModelAndView;

import org.springframework.web.servlet.mvc.AbstractController;

/\*\*

\*

\* @author dedhi

\*/

@Controller

public class RedirectController{

//

// @Override

// protected ModelAndView handleRequestInternal(HttpServletRequest request, HttpServletResponse response) throws Exception {

// ModelAndView mv = null;

// mv = new ModelAndView("add-question");

// return mv;

// }

@RequestMapping(value = "/redirect-add-question.htm")

protected ModelAndView getAddQuestionModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("add-question");

mv.addObject("domainList",domainList);

mv.addObject("error","");

mv.addObject("title","");

mv.addObject("desc","");

return mv;

}

@RequestMapping(value = "/redirect-user-home.htm")

protected ModelAndView getUserHomeModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

mv = new ModelAndView("user-home");

return mv;

}

@RequestMapping(value = "/redirect-admin-home.htm")

protected ModelAndView getAdminHomeModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

mv = new ModelAndView("admin-home");

return mv;

}

@RequestMapping(value = "/redirect-approver-home.htm")

protected ModelAndView getApproverHomeModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

mv = new ModelAndView("approver-home");

return mv;

}

@RequestMapping(value="/redirect-get-question-list.htm")

protected ModelAndView getQuestionList(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

QuestionDao questionDao = new QuestionDao();

List<Question> questionList = questionDao.getQuestionList();

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","");

return mv;

}

@RequestMapping(value="/redirect-get-approver-question-list.htm")

protected ModelAndView getApproverQuestionList(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

QuestionDao questionDao = new QuestionDao();

List<Question> questionList = questionDao.getApproverQuestionList();

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "approverQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","");

return mv;

}

@RequestMapping(value="/redirect-get-my-question-list.htm")

protected ModelAndView getMyQuestionList(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

User user = (User) request.getSession().getAttribute("user");

QuestionDao questionDao = new QuestionDao();

List<Question> questionList = questionDao.getMyQuestionList(user.getUser\_id());

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "myQuestions");

mv.addObject("domainList",domainList);

// ApproverDao approverDao = new ApproverDao();

// Approver approver = approverDao.getDummyApprover();

mv.addObject("error","");

return mv;

}

@RequestMapping(value="/redirect-search-list.htm")

protected ModelAndView getSearchQuestionList(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

long domain\_id = Long.parseLong(request.getParameter("domain"));

QuestionDao questionDao = new QuestionDao();

List<Question> questionList = questionDao.searchQuestionList(request.getParameter("keyword"), domain\_id);

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

if(request.getParameter("action").equals("allQuestions")){

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "allQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","");

return mv;

}else if(request.getParameter("action").equals("myQuestions")){

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "myQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","");

return mv;

}else if(request.getParameter("action").equals("approverQuestions")){

List<Question> questions = questionDao.searchApproverQuestionList(request.getParameter("keyword"), domain\_id);

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questions);

mv.addObject("action", "approverQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","");

return mv;

}

return mv;

}

@RequestMapping(value="/redirect-edit-question.htm")

protected ModelAndView editMyQuestion(@RequestParam("question\_id") long question\_id, HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

User user = (User) request.getSession().getAttribute("user");

QuestionDao questionDao = new QuestionDao();

Question question = questionDao.getQuestion(question\_id);

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("edit-question");

mv.addObject("question", question);

mv.addObject("domainList", domainList);

mv.addObject("error","");

return mv;

}

@RequestMapping(value="/redirect-get-my-answer-list.htm")

protected ModelAndView getMyAnswerList(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

User user = (User) request.getSession().getAttribute("user");

AnswerDao answerDao = new AnswerDao();

List<Answer> answers = answerDao.getMyAnswerList(user.getUser\_id());

HashMap<Question, ArrayList<Answer>> answerList = new HashMap<>();

for(Answer a : answers){

if(answerList.containsKey(a.getQuestion())){

answerList.get(a.getQuestion()).add(a);

}else{

ArrayList<Answer> ans = new ArrayList<Answer>();

ans.add(a);

answerList.put(a.getQuestion(), ans);

}

}

mv = new ModelAndView("view-my-answers");

mv.addObject("answerList",answerList);

mv.addObject("error","");

return mv;

}

@RequestMapping(value = "redirect-add-answer.htm")

protected ModelAndView getAddAnswerModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

QuestionDao questionDao = new QuestionDao();

long id = Long.parseLong(request.getParameter("id"));

Question question = questionDao.getQuestion(id);

mv = new ModelAndView("add-answer");

mv.addObject("question", question);

mv.addObject("desc","");

mv.addObject("error","");

return mv;

}

@RequestMapping(value = "redirect-my-profile.htm")

protected ModelAndView getMyProfileModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

String role = (String) request.getSession().getAttribute("role");

if(role.equals("user")){

User myProfile = (User) request.getSession().getAttribute("user");

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","disabled");

mv.addObject("view","true");

mv.addObject("error","");

}else if(role.equals("approver")){

Approver myProfile = (Approver) request.getSession().getAttribute("user");

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","disabled");

mv.addObject("view","true");

mv.addObject("error","");

}

return mv;

}

@RequestMapping(value = "redirect-edit-myProfile.htm")

protected ModelAndView editMyProfileModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

String role = (String) request.getSession().getAttribute("role");

if(role.equals("user")){

User myProfile = (User) request.getSession().getAttribute("user");

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

mv.addObject("error","");

}else if(role.equals("approver")){

Approver myProfile = (Approver) request.getSession().getAttribute("user");

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

mv.addObject("error","");

}

return mv;

}

@RequestMapping(value = "redirect-save-myProfile.htm")

protected ModelAndView saveMyProfileModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

Validation validation = new Validation();

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

long user\_id = Long.parseLong(request.getParameter("user\_id"));

String firstname = request.getParameter("firstname");

String lastname = request.getParameter("lastname");

String gender = request.getParameter("gender");

Date date = new Date();

String address = request.getParameter("address");

String email = request.getParameter("email");

String phone = request.getParameter("phone");

int result = 0;

if(request.getSession().getAttribute("role").equals("user")){

User myProfile = (User) request.getSession().getAttribute("user");

try{

date = new SimpleDateFormat("yyyy-MM-dd").parse(request.getParameter("dateofbirth").substring(0, 9));

}catch(Exception e){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter correct date!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

if(date.compareTo(new Date()) > 0){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter a date less than today's date!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

if(gender.equals("")){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter your Gender!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.onlyLettersValidation(firstname) || firstname.trim().length() < 0 || firstname.trim().length() > 50){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid First Name! First name cannot have more than 50 letters");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.onlyLettersValidation(lastname) || lastname.trim().length() < 0 || lastname.trim().length() > 50){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Last Name! Last name cannot have more than 50 letters");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(address.trim().length() < 8 || address.trim().length() > 100){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Address! Address cannot be less than 8 letters or more than 100 letters!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.phoneNumberValidation(phone) || phone.trim().length() > 20){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Phone Number!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

UserDao userDao = new UserDao();

result = userDao.updateUser(user\_id, firstname, lastname, gender, date, address, email, phone);

}else if(request.getSession().getAttribute("role").equals("approver")){

Approver myProfile = (Approver) request.getSession().getAttribute("user");

try{

date = new SimpleDateFormat("yyyy-MM-dd").parse(request.getParameter("dateofbirth").substring(0, 9));

}catch(Exception e){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter correct date!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

if(date.compareTo(new Date()) > 0){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter a date less than today's date!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

if(gender.equals("")){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter your Gender!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.onlyLettersValidation(firstname) || firstname.trim().length() < 0 || firstname.trim().length() > 50){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid First Name! First name cannot have more than 50 letters");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.onlyLettersValidation(lastname) || lastname.trim().length() < 0 || lastname.trim().length() > 50){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Last Name! Last name cannot have more than 50 letters");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(address.trim().length() < 8 || address.trim().length() > 100){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Address! Address cannot be less than 8 letters or more than 100 letters!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}else if(!validation.phoneNumberValidation(phone) || phone.trim().length() > 20){

mv = new ModelAndView("my-profile");

mv.addObject("error","Please enter valid Phone Number!");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","");

mv.addObject("view","false");

return mv;

}

ApproverDao approverDao = new ApproverDao();

result = approverDao.updateApprover(user\_id, firstname, lastname, gender, date, address, email, phone);

}

String role = (String) request.getSession().getAttribute("role");

if(role.equals("user")){

User myProfile = (User) request.getSession().getAttribute("user");

if(result == 1){

myProfile.setFirstname(firstname);

myProfile.setLastname(lastname);

myProfile.setGender(gender);

myProfile.setDate\_of\_birth(date);

myProfile.setAddress(address);

myProfile.setEmail(email);

myProfile.setPhone\_number(phone);

}

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","disabled");

mv.addObject("view","true");

if(result == 1){

mv.addObject("error","Profile updated successfully!");

}else{

mv.addObject("error","Something went wrong. Please try again later!");

}

}else if(role.equals("approver")){

Approver myProfile = (Approver) request.getSession().getAttribute("user");

if(result == 1){

myProfile.setFirstname(firstname);

myProfile.setLastname(lastname);

myProfile.setGender(gender);

myProfile.setDate\_of\_birth(date);

myProfile.setAddress(address);

myProfile.setEmail(email);

myProfile.setPhone\_number(phone);

}

mv = new ModelAndView("my-profile");

mv.addObject("myProfile", myProfile);

mv.addObject("date",new SimpleDateFormat("yyyy-MM-dd").format(myProfile.getDate\_of\_birth()));

mv.addObject("disabled","disabled");

mv.addObject("view","true");

if(result == 1){

mv.addObject("error","Profile updated successfully!");

}else{

mv.addObject("error","Something went wrong. Please try again later!");

}

}

return mv;

}

@RequestMapping(value = "redirect-edit-answer.htm")

protected ModelAndView editAnswerModel(@RequestParam("answer\_id") long answer\_id, HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

AnswerDao answerDao = new AnswerDao();

Answer answer = answerDao.getAnswer(answer\_id);

mv = new ModelAndView("edit-answer");

mv.addObject("answer\_id",answer\_id);

mv.addObject("answer\_desc",answer.getAnswer\_description());

mv.addObject("error","");

return mv;

}

@RequestMapping(value = "redirect-delete-answer.htm")

protected ModelAndView deleteAnswerModel(@RequestParam("answer\_id") long answer\_id, HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

AnswerDao answerDao = new AnswerDao();

int result = answerDao.deleteAnswer(answer\_id);

User user = (User) request.getSession().getAttribute("user");

List<Answer> answers = answerDao.getMyAnswerList(user.getUser\_id());

HashMap<Question, ArrayList<Answer>> answerList = new HashMap<>();

for(Answer a : answers){

if(answerList.containsKey(a.getQuestion())){

answerList.get(a.getQuestion()).add(a);

}else{

ArrayList<Answer> ans = new ArrayList<Answer>();

ans.add(a);

answerList.put(a.getQuestion(), ans);

}

}

mv = new ModelAndView("view-my-answers");

mv.addObject("answerList",answerList);

if(result == 1){

mv.addObject("error","Answer deleted successfully!");

}else{

mv.addObject("error","Something went wrong. Please try again later!");

}

return mv;

}

@RequestMapping(value = "redirect-delete-question.htm")

protected ModelAndView deleteQuestionModel(@RequestParam("question\_id") long question\_id, HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

QuestionDao questionDao = new QuestionDao();

int result = questionDao.deleteQuestion(question\_id);

if(result == 1){

User user = (User) request.getSession().getAttribute("user");

List<Question> questionList = questionDao.getMyQuestionList(user.getUser\_id());

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "myQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","Question deleted successfully!");

return mv;

}else{

User user = (User) request.getSession().getAttribute("user");

List<Question> questionList = questionDao.getMyQuestionList(user.getUser\_id());

DomainDao domainDao = new DomainDao();

ArrayList<Domain> domainList = domainDao.getDomainList();

mv = new ModelAndView("view-all-questions");

mv.addObject("questionList", questionList);

mv.addObject("action", "myQuestions");

mv.addObject("domainList",domainList);

mv.addObject("error","Something went wrong. Please try again later!");

return mv;

}

}

@RequestMapping(value = "redirect-approve-questions.htm")

protected ModelAndView approveQuestionsModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

ApproverDao approverDao = new ApproverDao();

ArrayList<Question> questionList = approverDao.getQuestionListForApproval();

mv = new ModelAndView("approve-questions");

mv.addObject("questionList",questionList);

mv.addObject("reject","");

if(questionList.isEmpty()){

mv.addObject("error","No pending approvals");

}else{

mv.addObject("error","");

}

return mv;

}

@RequestMapping(value = "redirect-approve-answers.htm")

protected ModelAndView approveAnswersModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

ApproverDao approverDao = new ApproverDao();

ArrayList<Answer> answerList = approverDao.getAnswerListForApproval();

mv = new ModelAndView("approve-answers");

mv.addObject("answerList",answerList);

mv.addObject("reject","");

if(answerList.isEmpty()){

mv.addObject("error","No pending approvals");

}else{

mv.addObject("error","");

}

return mv;

}

@RequestMapping(value = "redirect-register-user.htm")

protected ModelAndView registerUserModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

mv = new ModelAndView("register-user");

mv.addObject("firstname","");

mv.addObject("lastname","");

mv.addObject("date", new Date());

mv.addObject("address","");

mv.addObject("email","");

mv.addObject("phone","");

mv.addObject("username","");

mv.addObject("password","");

mv.addObject("error","");

return mv;

}

@RequestMapping(value = "redirect-add-approver.htm")

protected ModelAndView addApproverModel(HttpServletRequest request, HttpServletResponse response) throws Exception{

ModelAndView mv = null;

if(request.getSession().getAttribute("user") == null){

mv = new ModelAndView("login");

return mv;

}

mv = new ModelAndView("add-approver");

mv.addObject("firstname","");

mv.addObject("lastname","");

mv.addObject("date", new Date());

mv.addObject("address","");

mv.addObject("email","");

mv.addObject("phone","");

mv.addObject("username","");

mv.addObject("password","");

mv.addObject("error","");

return mv;

}

}