

Venue Management Tool (BeOnGame)**Vinoth Punniyamoorthy(001225656)****Abstract:**

Online portal for finding and reserving available venue for specific type of outdoor / Indoor game like Badminton, Tennis, Table Tennis, etc. The application can aid to reserve the venue for the day based on end user. On the available list of Venues provided by the service providers, end user can book and reserve the venues. On blocking a venue, the number of slots available for the day goes down and email confirmation is sent to the end-user based on the reservation.

Roles:

1. **Admin:** Admin registers the service providers.
2. **Service Providers:** Adds or Removes the venues slots per sport.
3. **Users:** A user can search and reserve an available sports services based on the game he wants to play.

Features:

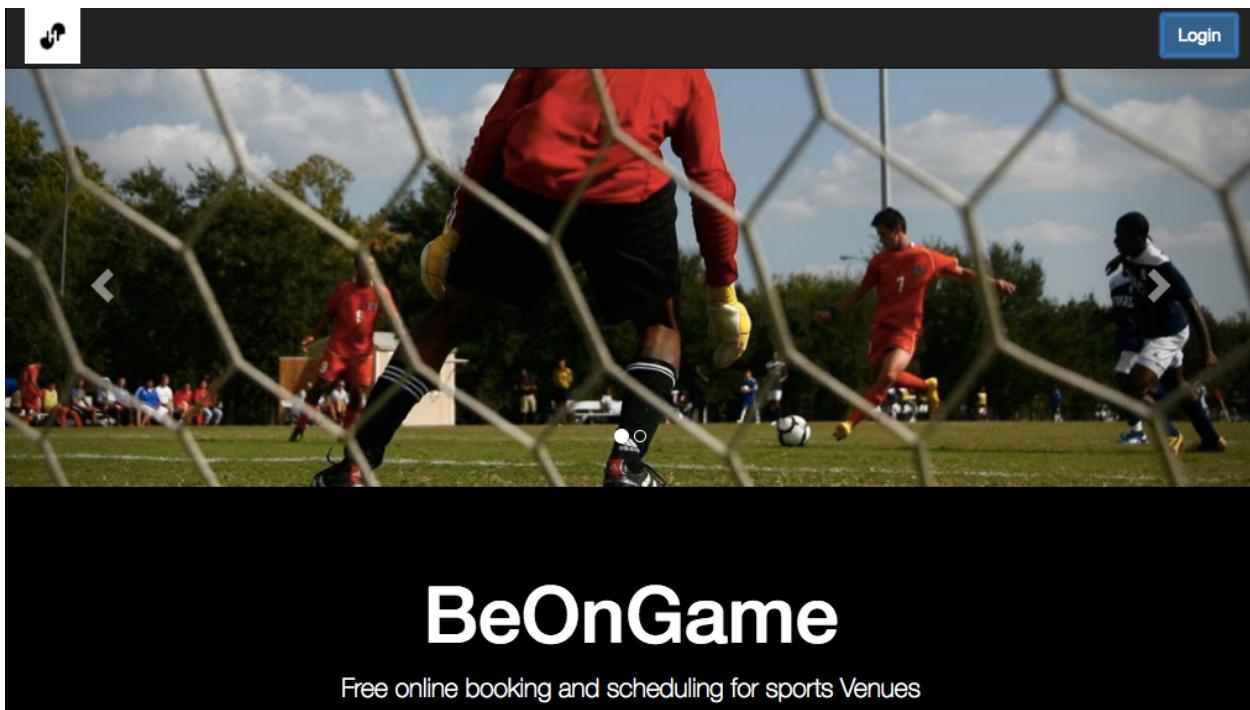
- Manage and show availability of venue to play specific indoor and outdoor sports.
- Provide a frictionless booking process for your users.
- Control your venue from anywhere
- Customize your venue setup.

Key Concepts Included:

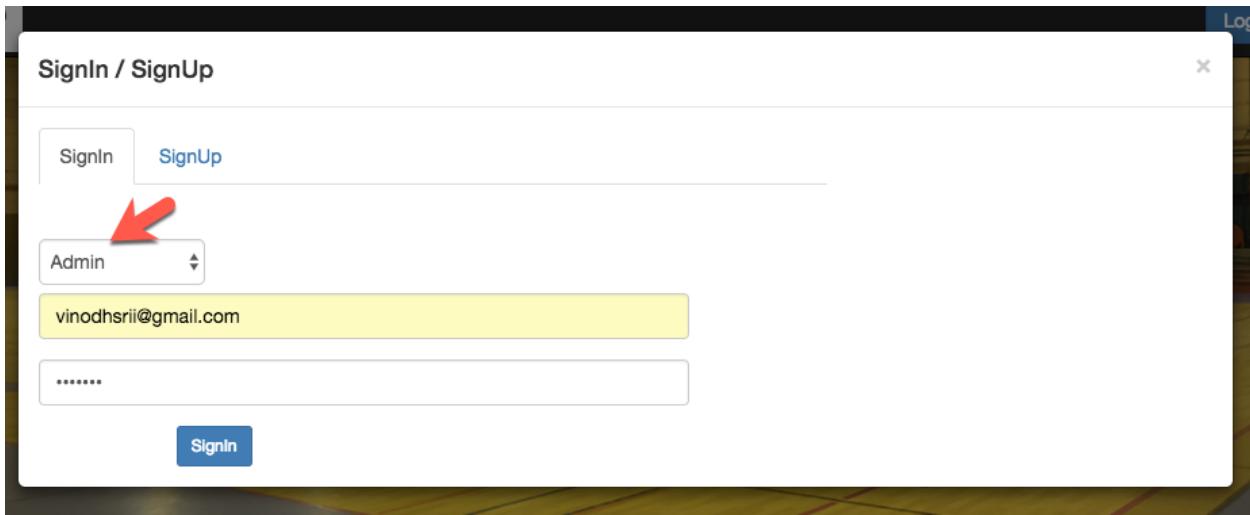
- Annotated POJO Model
- Annotated Controllers
- Hibernate
- JSP View Pages
- Velocity View Page
- Ajax's Concepts including JSON
- Authorization Securities.

Work Flow:

Home Screen:

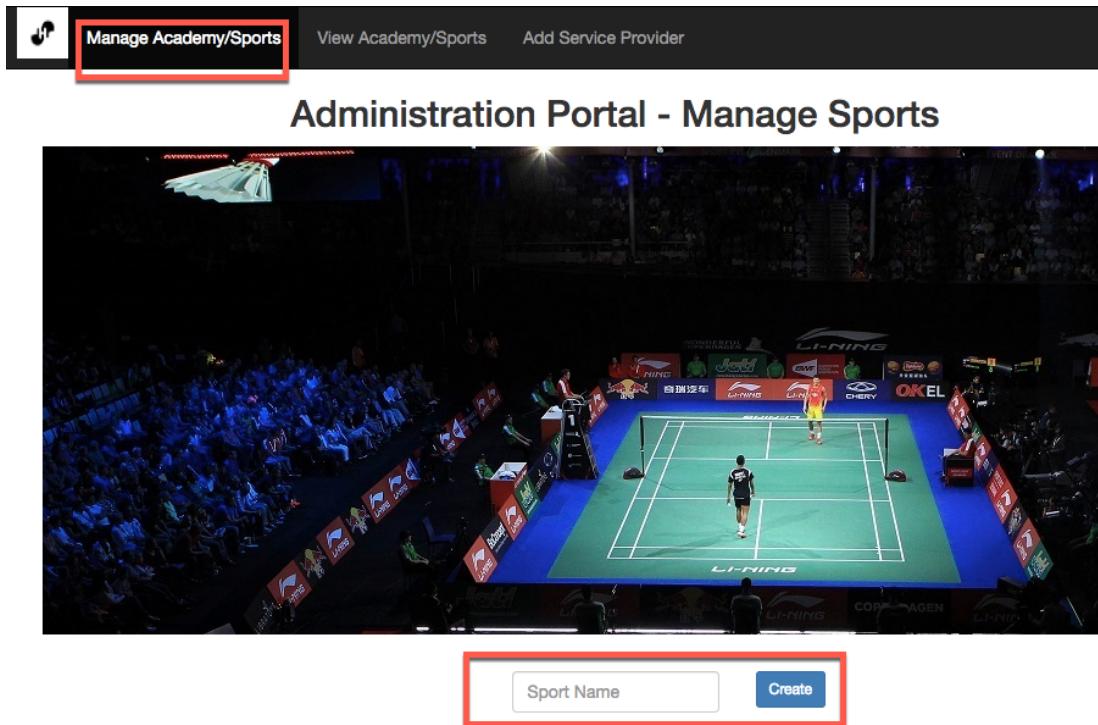


Step 1: Admin Login



Step 2: Admin has 3 functionalities

- Manage Sports Academy** - Adds Sports that Service Providers can include.



Manage Academy/Sports

View Academy/Sports

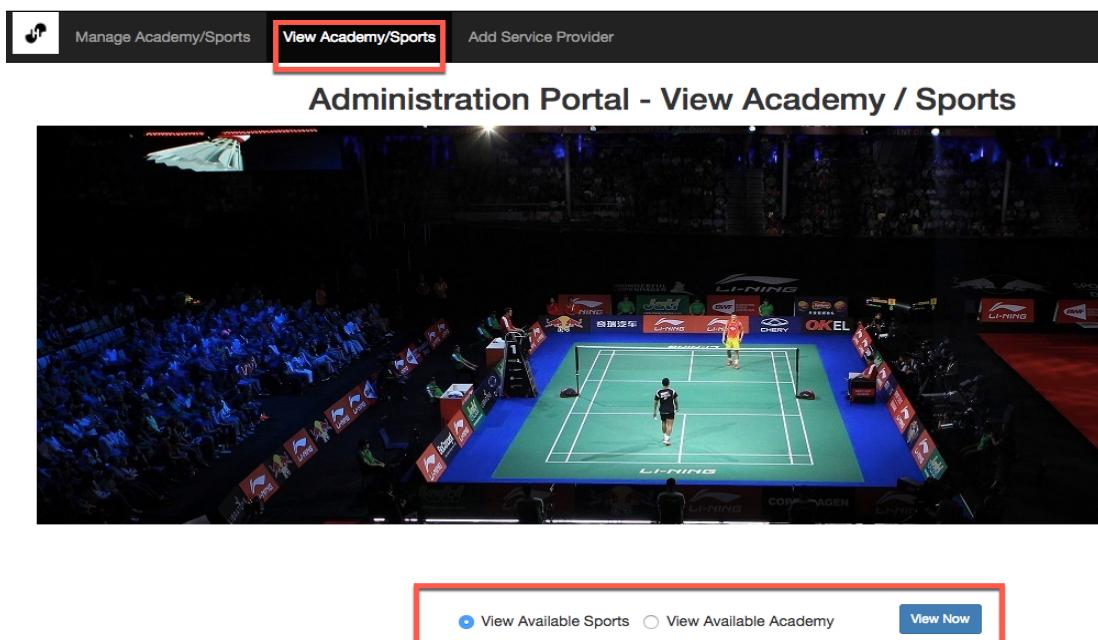
Add Service Provider

Administration Portal - Manage Sports

Sport Name

Create

- View Academy / Sports** – Can have a look at all Sports and Academies.



Manage Academy/Sports

View Academy/Sports

Add Service Provider

Administration Portal - View Academy / Sports

View Available Sports View Available Academy

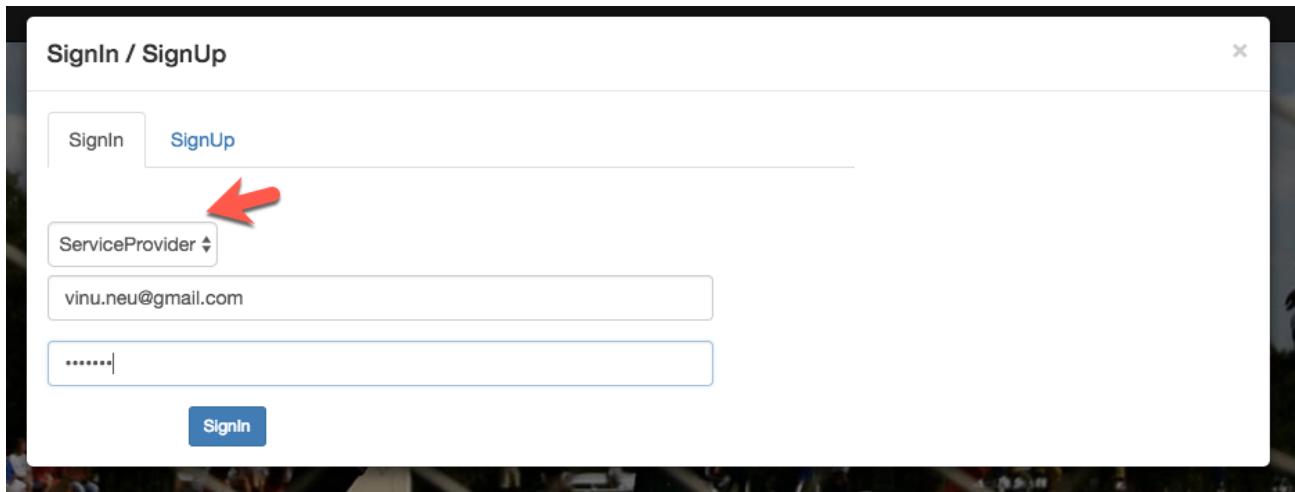
View Now

- c. **Add Service Provider** - Adds Service Providers to the Portal / get registered.



The screenshot shows the 'Administration Portal - Add Service Providers' interface. At the top, there are three buttons: 'Manage Academy/Sports', 'View Academy/Sports', and 'Add Service Provider', with the last one being highlighted by a red box. Below the buttons is a banner featuring a badminton match. The main form area contains six input fields arranged in two rows of three: 'ServiceProvider', 'Email-ID', 'Password' (all in the first row); 'First Name', 'Last Name', 'Age' (all in the second row); and 'Academy Name', 'Academy Location', 'Mobile' (all in the third row). A 'Create' button is located at the bottom left of the form.

Step 3: Service Provider Login



The screenshot shows the 'SignIn / SignUp' page. It features two tabs: 'SignIn' (selected) and 'SignUp'. Below the tabs is a dropdown menu labeled 'ServiceProvider' with a red arrow pointing to it. The main form includes fields for 'Email-ID' (containing 'vinu.neu@gmail.com') and 'Password' (containing '.....'). A 'SignIn' button is located at the bottom left of the form.

Step 4: Service Provider has 2 Functionalities

Manage Sports

Service Provider Portal - View Sports

Add Slots to your Academy Remove Slots from your Academy

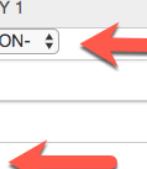
[View Now](#)

a. Add Slots / Sports to his /her Academy

Create Slots Sports

Service Provider Portal - Manage Sports

AcademyID	1
AcademyName	ACADEMY 1
Sports	BADMINTON- ↴
Price	200
Slots Count	2
	Create

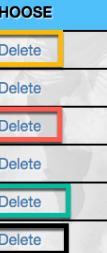


b. Remove Slots /sports to his/her Academy

Service Portal - List of Available Slots



ACADEMY NAME	ACADEMY LOCATION	SPORTS NAME	PRICE	AVAILABILITY	CHOOSE
ACADEMY 1	BOSTON	BADMINTON	500	N	Delete
ACADEMY 1	BOSTON	BADMINTON	500	N	Delete
ACADEMY 1	BOSTON	ICEHOCKEY	200	N	Delete
ACADEMY 1	BOSTON	ICEHOCKEY	200	N	Delete
ACADEMY 1	BOSTON	BADMINTON	75	A	Delete
ACADEMY 1	BOSTON	BADMINTON	75	A	Delete



Step 4: End-user Login

SignIn / SignUp

SignIn SignUp

User

mailpoorani@gmail.com

.....

SignIn



Step 6: End-user has 3 Functions

- Choose a sport of his/her interest.

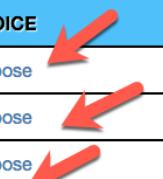
Search Your Sport

- Choose the Academy of his/her choice.

User Portal - List of Academies



ACADEMY NAME	ACADEMY LOCATION	CHOICE
ACADEMY 3	TRICHY	Choose
ACADEMY 2	BANGALORE	Choose
ACADEMY 1	BOSTON	Choose



c. Reserve a slot.

User Portal - List of Available Slots



ACADEMY NAME	ACADEMY LOCATION	SPORTS NAME	PRICE	SLOT NO	AVAILABILITY	CHOOSE
ACADEMY 1	BOSTON	BADMINTON	500	2	N	Occupied
ACADEMY 1	BOSTON	BADMINTON	500	1	N	Occupied
ACADEMY 1	BOSTON	ICEHOCKEY	200	2	N	Occupied
ACADEMY 1	BOSTON	ICEHOCKEY	200	1	N	Occupied
ACADEMY 1	BOSTON	BADMINTON	75	2	A	Pick
ACADEMY 1	BOSTON	BADMINTON	75	1	A	Pick

Snap Shots of controllers



1. Admincontroller

untitled

```
1 package com.beongame.top.controllers;
2
3 import java.util.List;
4
5 import javax.servlet.http.HttpServletRequest;
6 import javax.servlet.http.HttpSession;
7
8 import org.springframework.beans.factory.annotation.Autowired;
9 import org.springframework.beans.factory.annotation.Qualifier;
10 import org.springframework.http.MediaType;
11 import org.springframework.stereotype.Controller;
12 import org.springframework.web.bind.annotation.RequestMapping;
13 import org.springframework.web.bind.annotation.RequestMethod;
14 import org.springframework.web.bind.annotation.ResponseBody;
15 import org.springframework.web.servlet.ModelAndView;
16
17 import com.beongame.top.dao.AcademyDAO;
18 import com.beongame.top.dao.ServiceproviderDAO;
19 import com.beongame.top.dao.SportsDAO;
20 import com.beongame.top.exception.UserException;
21 import com.beongame.top.pojo.Academy;
22 import com.beongame.top.pojo.Email;
23 import com.beongame.top.pojo.Person;
24 import com.beongame.top.pojo.Serviceprovider;
25 import com.beongame.top.pojo.Sports;
26 import com.beongame.top.random.EmailDetails;
27
28 @Controller
29 @RequestMapping("/admin/*")
30 public class Admincontroller {
31
32     @Autowired
33     @Qualifier("svcprDAO")
34     ServiceproviderDAO svcprDAO;
35
36     @Autowired
37     @Qualifier("sportsDAO")
38     SportsDAO sportsDAO;
39
40     @Autowired
41     @Qualifier("academyDAO")
42     AcademyDAO academyDAO;
43
44     // // Add Sports - Get
45     @RequestMapping(value = "admin/sportsCreates", method = RequestMethod.GET, produces = MediaType.TEXT_PLAIN_VALUE)
46     protected @ResponseBody String createSports(HttpServletRequest request) throws Exception {
47         Sports check = sportsDAO.checkSports(request.getParameter("aid"));
48         if (check != null) {
49             return "Sports already exists";
50         } else {
51             return "";
52         }
53     }
54 }
```

Line 315, Column 1

Tab Size: 4 Java

```
untitled untitled UNREGISTERED
54
55     // Add Sports - Get
56     @RequestMapping(value = "admin/sportsCreate", method = RequestMethod.GET)
57     protected String createSport(HttpServletRequest request) throws Exception {
58         HttpSession session = (HttpSession) request.getSession();
59
60         if (session.getAttribute("ltype") == null) {
61             return "redirect-Page";
62         } else {
63             String ltype = (String) session.getAttribute("ltype");
64             String ladmin = "Admin";
65             if (ltype.equals(ladmin)) {
66                 Person user = (Person) session.getAttribute("cuser");
67                 if (user != null) {
68                     session.setAttribute("SuccessMessage", "Sports added successfully");
69                     return "success";
70                 } else {
71                     return "redirect-Page";
72                 }
73             } else {
74                 return "redirect-Page";
75             }
76         }
77     }
78
79     // Add Sports - Post
80
81     @RequestMapping(value = "admin/sportsCreate", method = RequestMethod.POST)
82     protected String createSports(HttpServletRequest request) throws Exception {
83         HttpSession session = (HttpSession) request.getSession();
84         try {
85             Sports check = sportsDAO.checkSports(request.getParameter("spName"));
86             if (check == null) {
87                 Sports game = sportsDAO.addSports(request.getParameter("spName"));
88                 if (game == null) {
89                     session.setAttribute("errorMessage", "Issue while adding Sports at this point .Try again Later");
90                     return "error";
91                 }
92
93                 session.setAttribute("sports", request.getParameter("spName"));
94                 session.setAttribute("SuccessMessage", "Sports added successfully");
95                 return "success";
96             } else {
97                 session.setAttribute("errorMessage", "Sorry ! This sport already exists in the DB.");
98                 return "error";
99             }
100         } catch (UserException e) {
101             System.out.println("Exception: " + e.getMessage());
102             session.setAttribute("errorMessage", "error while Adding Sports");
103             return "error";
104         }
105     }
106 }

```

Line 315, Column 1 Tab Size: 4 Java

```
untitled untitled UNREGISTERED
109
110     // Email check
111     @RequestMapping(value = "admin/serviceprovidersignups", method = RequestMethod.GET, produces = MediaType.TEXT_PLAIN_VALUE)
112     protected @ResponseBody String emailcheck(HttpServletRequest request) throws Exception {
113
114         Person p = svcpvDAO.checkEmail(request.getParameter("aid1"));
115
116         if (p != null) {
117             return "EmailID Not Available";
118         } else {
119             return "";
120         }
121
122     }
123
124     // Academy check
125     @RequestMapping(value = "admin/serviceprovidersignups", method = RequestMethod.GET, produces = MediaType.TEXT_PLAIN_VALUE)
126     protected @ResponseBody String academyccheck(HttpServletRequest request) throws Exception {
127
128         Academy check = academyDAO.checkAcademy1(request.getParameter("aid2"));
129         if (check != null) {
130             return "Academy Already Exists";
131         } else {
132             return "";
133         }
134
135     }
136
137     // Add ServiceProvider - Get
138     @RequestMapping(value = "admin/serviceprovidersignup", method = RequestMethod.GET)
139     protected ModelAndView createServiceProv(HttpServletRequest request) throws Exception {
140         HttpSession session = (HttpSession) request.getSession();
141         if (session.getAttribute("ltype") == null) {
142             return new ModelAndView("redirect-Page");
143         } else {
144
145             String ltype = (String) session.getAttribute("ltype");
146             String ladmin = "Admin";
147             if (ltype.equals(ladmin)) {
148                 Person user = (Person) session.getAttribute("cuser");
149                 if (user != null) {
150                     session.setAttribute("SuccessMessage", "Sports added successfully");
151                     return new ModelAndView("success");
152                 } else {
153                     return new ModelAndView("redirect-Page");
154                 }
155             } else {
156                 return new ModelAndView("redirect-Page");
157             }
158         }
159     }
160
161     // Add ServiceProvider - Post

```

Line 315, Column 1 Tab Size: 4 Java

```
untitled UNREGISTERED
161 // Add ServiceProvider - Post
162 @RequestMapping(value = "admin/serviceprovidersignup", method = RequestMethod.POST)
163 protected String createServiceProvider(HttpServletRequest request) throws Exception {
164     HttpSession session = (HttpSession) request.getSession();
165     try {
166         Academy check = academyDAO.checkAcademy(request.getParameter("acadName"), request.getParameter("acadLoc"));
167         if (check == null) {
168             Person p = svcrDAO.scheck(request.getParameter("emailId"));
169             if (p == null) {
170                 Academy acad = academyDAO.addAcademy(request.getParameter("acadName"),
171                     request.getParameter("acadLoc"));
172                 if (acad == null) {
173                     System.out.println("Issue while creating Service Provider at this point .Try again Later");
174                     session.setAttribute("errorMessage",
175                         "Issue while creating Service Provider at this point .Try again Later");
176                     return "error";
177                 } else {
178                     Serviceprovider sp = svcrDAO.signup(request.getParameter("firstName"),
179                         request.getParameter("lastName"), request.getParameter("age"),
180                         request.getParameter("emailId"), request.getParameter("contactNo"),
181                         request.getParameter("password"),
182                         request.getParameter("logInType"), acad);
183
184                     if (sp == null) {
185                         System.out.println("Issue While Creating Service Provider at this point .Try again Later");
186                         session.setAttribute("errorMessage",
187                             "Issue While Creating User at this point .Try again Later");
188                         return "error";
189                     }
190
191                     String message = "BeOnGame welcomes you. Thanks for registering with Beongame.Please find your credentials. "
192                         + " EmailId: " + request.getParameter("emailId") + " and " + " Password :"
193                         + " " + request.getParameter("password");
194
195                     // Send Mail
196
197                     Email emailDetails = new Email();
198                     emailDetails.setFromEmail("vinodhsriri@gmail.com");
199                     emailDetails.setToEmail(request.getParameter("emailId"));
200                     emailDetails.setSubject("SignIn Credentials");
201                     emailDetails.setMessage(message);
202                     EmailDetails email = new EmailDetails();
203                     email.sendEmail(emailDetails);
204
205                     session.setAttribute("spuser", sp);
206                     session.setAttribute("SuccessMessage", "Service Provider added successfully");
207                     return "success";
208                 }
209             } else {
210                 session.setAttribute("errorMessage",
211                     "Sorry ! Service Provider / User Details already exists in DB");
212             }
213         }
214     }
215     else {
216         session.setAttribute("errorMessage", "Sorry ! Academy already exists");
217         return "error";
218     }
219 }
220 } catch (UserException e) {
221     System.out.println("Exception: " + e.getMessage());
222     session.setAttribute("errorMessage", "error While Adding Service Provider");
223     return "error";
224 }
225 }
226 }
227 }
228 // View Sports / Academy - Get
229 @RequestMapping(value = "admin/view", method = RequestMethod.GET)
230 protected ModelAndView views(HttpServletRequest request) throws Exception {
231     HttpSession session = (HttpSession) request.getSession();
232     if (session.getAttribute("ltype") == null) {
233         return new ModelAndView("redirect:Page");
234     } else {
235         String ltype = (String) session.getAttribute("ltype");
236         String ladmin;
237         if (ltype.equals(ladmin)) {
238             Person user = (Person) session.getAttribute("cuser");
239             if (user == null) {
240                 try {
241                     String viewValue = (String) session.getAttribute("optradio");
242                     if (viewValue.equalsIgnoreCase("sp")) {
243                         List<Sports> sports = sportsDAO.list();
244                         if (sports == null) {
245                             System.out.println("Issue while retrieving Sports at this point .Try again Later");
246                             session.setAttribute("errorMessage",
247                                 "Issue while adding Sports at this point .Try again Later");
248                             return new ModelAndView("error");
249                         }
250
251                         return new ModelAndView("search-result", "spList", sports);
252                     } else {
253                         List<Academy> academy = academyDAO.list();
254                         if (academy == null) {
255                             System.out.println("Issue while retrieving Academy at this point .Try again Later");
256                             session.setAttribute("errorMessage",
257                                 "Issue while adding Academy at this point .Try again Later");
258                             return new ModelAndView("error");
259                         }
260                         return new ModelAndView("academyList", "acList", academy);
261                     }
262                 } catch (UserException e) {
263                     System.out.println("Exception: " + e.getMessage());
264                     session.setAttribute("errorMessage", "error while Pulling data from DB");
265                     return new ModelAndView("error");
266                 }
267             }
268         }
269     }
270 }
271 }
272 }
273 }
274 }
275 }
276 }
277 }
278 }
279 }
280 }
281 }
282 }
283 }
284 }
285 }
286 }
287 }
288 }
289 }
290 }
291 }
292 }
293 }
294 }
295 }
296 }
297 }
298 }
299 }
300 }
301 }
302 }
303 }
304 }
305 }
306 }
307 }
308 }
309 }
310 }
311 }
312 }
313 }
314 }
315 }
316 }
317 }
318 }
319 }
320 }
321 }
322 }
323 }
324 }
325 }
326 }
327 }
328 }
329 }
330 }
331 }
332 }
333 }
334 }
335 }
336 }
337 }
338 }
339 }
340 }
341 }
342 }
343 }
344 }
345 }
346 }
347 }
348 }
349 }
350 }
351 }
352 }
353 }
354 }
355 }
356 }
357 }
358 }
359 }
360 }
361 }
362 }
363 }
364 }
365 }
366 }
367 }
368 }
369 }
370 }
371 }
372 }
373 }
374 }
375 }
376 }
377 }
378 }
379 }
380 }
381 }
382 }
383 }
384 }
385 }
386 }
387 }
388 }
389 }
390 }
391 }
392 }
393 }
394 }
395 }
396 }
397 }
398 }
399 }
399 }
400 }
401 }
402 }
403 }
404 }
405 }
406 }
407 }
408 }
409 }
410 }
411 }
412 }
413 }
414 }
415 }
416 }
417 }
418 }
419 }
419 }
420 }
421 }
422 }
423 }
424 }
425 }
426 }
427 }
428 }
429 }
429 }
430 }
431 }
432 }
433 }
434 }
435 }
436 }
437 }
438 }
439 }
439 }
440 }
441 }
442 }
443 }
444 }
445 }
446 }
447 }
448 }
449 }
449 }
450 }
451 }
452 }
453 }
454 }
455 }
456 }
457 }
458 }
459 }
459 }
460 }
461 }
462 }
463 }
464 }
465 }
466 }
467 }
468 }
469 }
469 }
470 }
471 }
472 }
473 }
474 }
475 }
476 }
477 }
478 }
479 }
479 }
480 }
481 }
482 }
483 }
484 }
485 }
486 }
487 }
488 }
489 }
489 }
490 }
491 }
492 }
493 }
494 }
495 }
496 }
497 }
498 }
499 }
499 }
500 }
501 }
502 }
503 }
504 }
505 }
506 }
507 }
508 }
509 }
509 }
510 }
511 }
512 }
513 }
514 }
515 }
516 }
517 }
518 }
519 }
519 }
520 }
521 }
522 }
523 }
524 }
525 }
526 }
527 }
528 }
529 }
529 }
530 }
531 }
532 }
533 }
534 }
535 }
536 }
537 }
538 }
539 }
539 }
540 }
541 }
542 }
543 }
544 }
545 }
546 }
547 }
548 }
549 }
549 }
550 }
551 }
552 }
553 }
554 }
555 }
556 }
557 }
558 }
559 }
559 }
560 }
561 }
562 }
563 }
564 }
565 }
566 }
567 }
568 }
569 }
569 }
570 }
571 }
572 }
573 }
574 }
575 }
576 }
577 }
578 }
579 }
579 }
580 }
581 }
582 }
583 }
584 }
585 }
586 }
587 }
588 }
589 }
589 }
590 }
591 }
592 }
593 }
594 }
595 }
596 }
597 }
598 }
599 }
599 }
600 }
601 }
602 }
603 }
604 }
605 }
606 }
607 }
608 }
609 }
609 }
610 }
611 }
612 }
613 }
614 }
615 }
616 }
617 }
618 }
619 }
619 }
620 }
621 }
622 }
623 }
624 }
625 }
626 }
627 }
628 }
629 }
629 }
630 }
631 }
632 }
633 }
634 }
635 }
636 }
637 }
638 }
639 }
639 }
640 }
641 }
642 }
643 }
644 }
645 }
646 }
647 }
648 }
649 }
649 }
650 }
651 }
652 }
653 }
654 }
655 }
656 }
657 }
658 }
659 }
659 }
660 }
661 }
662 }
663 }
664 }
665 }
666 }
667 }
668 }
669 }
669 }
670 }
671 }
672 }
673 }
674 }
675 }
676 }
677 }
678 }
679 }
679 }
680 }
681 }
682 }
683 }
684 }
685 }
686 }
687 }
688 }
689 }
689 }
690 }
691 }
692 }
693 }
694 }
695 }
696 }
697 }
698 }
699 }
699 }
700 }
701 }
702 }
703 }
704 }
705 }
706 }
707 }
708 }
709 }
709 }
710 }
711 }
712 }
713 }
714 }
715 }
716 }
717 }
718 }
719 }
719 }
720 }
721 }
722 }
723 }
724 }
725 }
726 }
727 }
728 }
729 }
729 }
730 }
731 }
732 }
733 }
734 }
735 }
736 }
737 }
738 }
739 }
739 }
740 }
741 }
742 }
743 }
744 }
745 }
746 }
747 }
748 }
749 }
749 }
750 }
751 }
752 }
753 }
754 }
755 }
756 }
757 }
758 }
759 }
759 }
760 }
761 }
762 }
763 }
764 }
765 }
766 }
767 }
768 }
769 }
769 }
770 }
771 }
772 }
773 }
774 }
775 }
776 }
777 }
778 }
778 }
779 }
780 }
781 }
782 }
783 }
784 }
785 }
786 }
787 }
788 }
788 }
789 }
789 }
790 }
791 }
792 }
793 }
794 }
795 }
796 }
797 }
798 }
798 }
799 }
799 }
800 }
801 }
802 }
803 }
804 }
805 }
806 }
807 }
808 }
809 }
809 }
810 }
811 }
812 }
813 }
814 }
815 }
816 }
817 }
818 }
819 }
819 }
820 }
821 }
822 }
823 }
824 }
825 }
826 }
827 }
828 }
829 }
829 }
830 }
831 }
832 }
833 }
834 }
835 }
836 }
837 }
838 }
839 }
839 }
840 }
841 }
842 }
843 }
844 }
845 }
846 }
847 }
848 }
849 }
849 }
850 }
851 }
852 }
853 }
854 }
855 }
856 }
857 }
858 }
859 }
859 }
860 }
861 }
862 }
863 }
864 }
865 }
866 }
867 }
868 }
869 }
869 }
870 }
871 }
872 }
873 }
874 }
875 }
876 }
877 }
877 }
878 }
878 }
879 }
879 }
880 }
881 }
882 }
883 }
884 }
885 }
885 }
886 }
886 }
887 }
887 }
888 }
888 }
889 }
889 }
890 }
891 }
892 }
893 }
894 }
895 }
896 }
897 }
898 }
898 }
899 }
899 }
900 }
901 }
902 }
903 }
904 }
905 }
906 }
907 }
908 }
909 }
909 }
910 }
911 }
912 }
913 }
914 }
915 }
916 }
917 }
918 }
919 }
919 }
920 }
921 }
922 }
923 }
924 }
925 }
926 }
927 }
928 }
929 }
929 }
930 }
931 }
932 }
933 }
934 }
935 }
936 }
937 }
938 }
939 }
939 }
940 }
941 }
942 }
943 }
944 }
945 }
946 }
947 }
948 }
949 }
949 }
950 }
951 }
952 }
953 }
954 }
955 }
956 }
957 }
958 }
959 }
959 }
960 }
961 }
962 }
963 }
964 }
965 }
966 }
967 }
968 }
969 }
969 }
970 }
971 }
972 }
973 }
974 }
975 }
976 }
977 }
977 }
978 }
978 }
979 }
979 }
980 }
981 }
982 }
983 }
984 }
985 }
985 }
986 }
986 }
987 }
987 }
988 }
988 }
989 }
989 }
990 }
991 }
992 }
993 }
994 }
995 }
996 }
997 }
998 }
998 }
999 }
999 }
1000 }
1001 }
1002 }
1003 }
1004 }
1005 }
1006 }
1007 }
1008 }
1009 }
1009 }
1010 }
1011 }
1012 }
1013 }
1014 }
1015 }
1016 }
1017 }
1018 }
1019 }
1019 }
1020 }
1021 }
1022 }
1023 }
1024 }
1025 }
1026 }
1027 }
1028 }
1029 }
1029 }
1030 }
1031 }
1032 }
1033 }
1034 }
1035 }
1036 }
1037 }
1038 }
1039 }
1039 }
1040 }
1041 }
1042 }
1043 }
1044 }
1045 }
1046 }
1047 }
1048 }
1049 }
1049 }
1050 }
1051 }
1052 }
1053 }
1054 }
1055 }
1056 }
1057 }
1058 }
1059 }
1059 }
1060 }
1061 }
1062 }
1063 }
1064 }
1065 }
1066 }
1067 }
1068 }
1069 }
1069 }
1070 }
1071 }
1072 }
1073 }
1074 }
1075 }
1076 }
1077 }
1077 }
1078 }
1078 }
1079 }
1079 }
1080 }
1081 }
1082 }
1083 }
1084 }
1085 }
1085 }
1086 }
1086 }
1087 }
1087 }
1088 }
1088 }
1089 }
1089 }
1090 }
1091 }
1092 }
1093 }
1094 }
1095 }
1095 }
1096 }
1096 }
1097 }
1097 }
1098 }
1098 }
1099 }
1099 }
1100 }
1101 }
1102 }
1103 }
1104 }
1105 }
1106 }
1107 }
1108 }
1109 }
1109 }
1110 }
1111 }
1112 }
1113 }
1114 }
1115 }
1116 }
1117 }
1118 }
1119 }
1119 }
1120 }
1121 }
1122 }
1123 }
1124 }
1125 }
1126 }
1127 }
1128 }
1129 }
1129 }
1130 }
1131 }
1132 }
1133 }
1134 }
1135 }
1136 }
1137 }
1138 }
1139 }
1139 }
1140 }
1141 }
1142 }
1143 }
1144 }
1145 }
1146 }
1147 }
1148 }
1149 }
1149 }
1150 }
1151 }
1152 }
1153 }
1154 }
1155 }
1156 }
1157 }
1158 }
1159 }
1159 }
1160 }
1161 }
1162 }
1163 }
1164 }
1165 }
1166 }
1167 }
1168 }
1169 }
1169 }
1170 }
1171 }
1172 }
1173 }
1174 }
1175 }
1176 }
1177 }
1177 }
1178 }
1178 }
1179 }
1179 }
1180 }
1181 }
1182 }
1183 }
1184 }
1185 }
1185 }
1186 }
1186 }
1187 }
1187 }
1188 }
1188 }
1189 }
1189 }
1190 }
1191 }
1192 }
1193 }
1194 }
1195 }
1195 }
1196 }
1196 }
1197 }
1197 }
1198 }
1198 }
1199 }
1199 }
1200 }
1201 }
1202 }
1203 }
1204 }
1205 }
1206 }
1207 }
1208 }
1209 }
1209 }
1210 }
1211 }
1212 }
1213 }
1214 }
1215 }
1216 }
1217 }
1218 }
1219 }
1219 }
1220 }
1221 }
1222 }
1223 }
1224 }
1225 }
1226 }
1227 }
1228 }
1229 }
1229 }
1230 }
1231 }
1232 }
1233 }
1234 }
1235 }
1236 }
1237 }
1238 }
1239 }
1239 }
1240 }
1241 }
1242 }
1243 }
1244 }
1245 }
1246 }
1247 }
1248 }
1249 }
1249 }
1250 }
1251 }
1252 }
1253 }
1254 }
1255 }
1256 }
1257 }
1258 }
1259 }
1259 }
1260 }
1261 }
1262 }
1263 }
1264 }
1265 }
1266 }
1267 }
1268 }
1269 }
1269 }
1270 }
1271 }
1272 }
1273 }
1274 }
1275 }
1276 }
1277 }
1277 }
1278 }
1278 }
1279 }
1279 }
1280 }
1281 }
1282 }
1283 }
1284 }
1285 }
1285 }
1286 }
1286 }
1287 }
1287 }
1288 }
1288 }
1289 }
1289 }
1290 }
1291 }
1292 }
1293 }
1294 }
1295 }
1295 }
1296 }
1296 }
1297 }
1297 }
1298 }
1298 }
1299 }
1299 }
1300 }
1301 }
1302 }
1303 }
1304 }
1305 }
1306 }
1307 }
1308 }
1309 }
1309 }
1310 }
1311 }
1312 }
1313 }
1314 }
1315 }
1316 }
1317 }
1318 }
1319 }
1319 }
1320 }
1321 }
1322 }
1323 }
1324 }
1325 }
1326 }
1327 }
1328 }
1329 }
1329 }
1330 }
1331 }
1332 }
1333 }
1334 }
1335 }
1336 }
1337 }
1338 }
1339 }
1339 }
1340 }
1341 }
1342 }
1343 }
1344 }
1345 }
1346 }
1347 }
1348 }
1349 }
1349 }
1350 }
1351 }
1352 }
1353 }
1354 }
1355 }
1356 }
1357 }
1358 }
1359 }
1359 }
1360 }
1361 }
1362 }
1363 }
1364 }
1365 }
1366 }
1367 }
1368 }
1369 }
1369 }
1370 }
1371 }
1372 }
1373 }
1374 }
1375 }
1376 }
1377 }
1377 }
1378 }
1378 }
1379 }
1379 }
1380 }
1381 }
1382 }
1383 }
1384 }
1385 }
1385 }
1386 }
1386 }
1387 }
1387 }
1388 }
1388 }
1389 }
1389 }
1390 }
1391 }
1392 }
1393 }
1394 }
1395 }
1395 }
1396 }
1396 }
1397 }
1397 }
1398 }
1398 }
1399 }
1399 }
1400 }
1401 }
1402 }
1403 }
1404 }
1405 }
1406 }
1407 }
1408 }
1409 }
1409 }
1410 }
1411 }
1412 }
1413 }
1414 }
1415 }
1416 }
1417 }
1418 }
1419 }
1419 }
1420 }
1421 }
1422 }
1423 }
1424 }
1425 }
1426 }
1427 }
1428 }
1429 }
1429 }
1430 }
1431 }
1432 }
1433 }
1434 }
1435 }
1436 }
1437 }
1438 }
1439 }
1439 }
1440 }
1441 }
1442 }
1443 }
1444 }
1445 }
1446 }
1447 }
1448 }
1449 }
1449 }
1450 }
1451 }
1452 }
1453 }
1454 }
1455 }
1456 }
1457 }
1458 }
1459 }
1459 }
1460 }
1461 }
1462 }
1463 }
1464 }
1465 }
1466 }
1467 }
1468 }
1469 }
1469 }
1470 }
1471 }
1472 }
1473 }
1474 }
1475 }
1475 }
1476 }
1476 }
1477 }
1477 }
1478 }
1478 }
1479 }
1479 }
1480 }
1481 }
1482 }
1483 }
1484 }
1485 }
1485 }
1486 }
1486 }
1487 }
1487 }
1488 }
1488 }
1489 }
1489 }
1490 }
1491 }
1492 }
1493 }
1494 }
1495 }
1495 }
1496 }
1496 }
1497 }
1497 }
1498 }
1498 }
1499 }
1499 }
1500 }
1501 }
1502 }
1503 }
1504 }
1505 }
1506 }
1507 }
1508 }
1509 }
1509 }
1510 }
1511 }
1512 }
1513 }
1514 }
1515 }
1516 }
1517 }
1518 }
1519 }
1519 }
1520 }
1521 }
1522 }
1523 }
1524 }
1525 }
1526 }
1527 }
1528 }
1529 }
1529 }
1530 }
1531 }
1532 }
1533 }
1534 }
1535 }
1536 }
1537 }
1538 }
1539 }
1539 }
1540 }
1541 }
1542 }
1543 }
1544 }
1545 }
1546 }
1547 }
1548 }
1549 }
1549 }
1550 }
1551 }
1552 }
1553 }
1554 }
1555 }
1556 }
1557 }
1558 }
1559 }
1559 }
1560 }
1561 }
1562 }
1563 }
1564 }
1565 }
1566 }
1567 }
1568 }
1569 }
1569 }
1570 }
1571 }
1572 }
1573 }
1574 }
1575 }
1576 }
1577 }
1577 }
1578 }
1578 }
1579 }
1579 }
1580 }
1581 }
1582 }
1583 }
1584 }
1585 }
1585 }
1586 }
1586 }
1587 }
1587 }
1588 }
1588 }
1589 }
1589 }
1590 }
1591 }
1592 }
1593 }
1594 }
1595 }
1595 }
1596 }
1596 }
1597 }
1597 }
1598 }
1598 }
1599 }
1599 }
1600 }
1601 }
1602 }
1603 }
1604 }
1605 }
1606 }
1607 }
1608 }
1609 }
1609 }
1610 }
1611 }
1612 }
1613 }
1614 }
1615 }
1616 }
1617 }
1618 }
1619 }
1619 }
1620 }
1621 }
1622 }
1623 }
1624 }
1625 }
1626 }
1627 }
1628 }
1629 }
1629 }
1630 }
1631 }
1632 }
1633 }
1634 }
1635 }
1636 }
1637 }
1638 }
1639 }
1639 }
1640 }
1641 }
1642 }
1643 }
1644 }
1645 }
1646 }
1647 }
1648 }
1649 }
1649 }
1650 }
1651 }
1652 }
1653 }
1654 }
1655 }
1656 }
1657 }
1658 }
1659 }
1659 }
1660 }
1661 }
1662 }
1663 }
1664 }
1665 }
1666 }
1667 }
1668 }
1669 }
1669 }
1670 }
1671 }
1672 }
1673 }
1674 }
1675 }
1676 }
1677 }
1677 }
1678 }
1678 }
1679 }
1679 }
1680 }
1681 }
1682 }
1683 }
1684 }
1685 }
1685 }
1686 }
1686 }
1687 }
1687 }
1688 }
1688 }
1689 }
1689 }
1690 }
1691 }
1692 }
1693 }
1694 }
1695 }
1695 }
1696 }
1696 }
1697 }
1697 }
1698 }
1698 }
1699 }
1699 }
1700 }
1701 }
1702 }
1703 }
1704 }
1705 }
1706 }
1707 }
1708 }
1709 }
1709 }
1710 }
1711 }
1712 }
1713 }
1714 }
1715 }
1716 }
1717 }
1718 }
1719 }
1719 }
1720 }
1721 }
1722 }
1723 }
1724 }
1725 }
1726 }
1727 }
1728 }
1729 }
1729 }
1730 }
1731 }
1732 }
1733 }
1734 }
1735 }
1736 }
1737 }
1738 }
1739 }
1739 }
1740 }
1741 }
1742 }
1743 }
1744 }
1745 }
1746 }
1747 }
1748 }
1749 }
1749 }
1750 }
1751 }
1752 }
1753 }
1754 }
1755 }
1756 }
1757 }
1758 }
1759 }
1759 }
1760 }
1761 }
1762 }
1763 }
1764 }
1765 }
1766 }
1767 }
1768 }
1769 }
1769 }
1770 }
1771 }
1772 }
1773 }
1774 }
1775 }
1776 }
1777 }
1777 }
1778 }
1778 }
1779 }
1779 }
1780 }
1781 }
1782 }
1783 }
1784 }
1785 }
1785 }
1786 }
1786 }
1787 }
1787 }
1788 }
1788 }
1789 }
1789 }
1790 }
1791 }
1792 }
1793 }
1794 }
1795 }
1795 }
1796 }
1796 }
1797 }
1797 }
1798 }
1798 }
1799 }
1799 }
1800 }
1801 }
1802 }
1803 }
1804 }
1805 }
1806 }
1807 }
1808 }
1809 }
1809 }
1810 }
1811 }
1812 }
1813 }
1814 }
1815 }
1816 }
1817 }
1818 }
1819 }
1819 }
1820 }
1821 }
1822 }
1823 }
1824 }
1825 }
1826 }
1827 }
1828 }
1829 }
1829 }
1830 }
1831 }
1832 }
1833 }
1834 }
1835 }
1836 }
1837 }
1838 }
1839 }
1839 }
1840 }
1841 }
1842 }
1843 }
1844 }
1845 }
1846 }
1847 }
1848 }
1849 }
1849 }
1850 }
1851 }
1852 }
1853 }
1854 }
1855 }
1856 }
1857 }
1858 }
1859 }
1859 }
1860 }
1861 }
1862 }
1863 }
1864 }
1865 }
1866 }
1867 }
1868 }
1869 }
1869 }
1870 }
1871 }
1872 }
1873 }
1874 }
1875 }
1876 }
1877 }
1877 }
1878 }
1878 }
1879 }
1879 }
1880 }
1881 }
1882 }
1883 }
1884 }
1885 }
1885 }
1886 }
1886 }
1887 }
1887 }
1888 }
1888 }
1889 }
1889 }
1890 }
1891 }
1892 }
1893 }
1894 }
1895 }
1895 }
1896 }
1896 }
1897 }
1897 }
1898 }
1898 }
1899 }
1899 }
1900 }
1901 }
1902 }
1903 }
1904 }
1905 }
1906 }
1907 }
1908 }
1909 }
1909 }
1910 }
1911 }
1912 }
1913 }
1914 }
1915 }
1916 }
1917 }
1918 }
1919 }
1919 }
1920 }
1921 }
1922 }
1923 }
1924 }
1925 }
1926 }
1927 }
1928 }
1929 }
1929 }
1930 }
1931 }
1932 }
1933 }
1934 }
1935 }
1936 }
1937 }
1938 }
1939 }
1939 }
1940 }
1941 }
1942 }
1943 }
1944 }
1945 }
1946 }
1947 }
1948 }
1949 }
1949 }
1950 }
1951 }
1952 }
1953 }
1954 }
1955 }
1956 }
1957 }
1958 }
1959 }
1959 }
1960 }
1961 }
1962 }
1963 }
1964 }
1965 }
1966 }
1967 }
1968 }
1969 }
1969 }
1970 }
1971 }
1972 }
1973 }
1974 }
1975 }
1976 }
1977 }
1977 }
1978 }
1978 }
1979 }
1979 }
1980 }
1981 }
1982 }
1983 }
1984 }
1985 }
1985 }
1986 }
1986 }
1987 }
1987 }
1988 }
1988 }
1989 }
1989 }
1990 }
1991 }
1992 }
1993 }
1994 }
1995 }
1995 }
1996 }
1996 }
1997 }
1997 }
1998 }
1998 }
1999 }
1999 }
2000 }
2001 }
2002 }
2003 }
2004 }
2005 }
2006 }
2007 }
2008 }
2009 }
2009 }
2010 }
2011 }
2012 }
2013 }
2014 }
2015 }
2016 }
2017 }
2018 }
2019 }
2019 }
2020 }
2021 }
2022 }
2023 }
2024 }
2025 }
2026 }
2027 }
2028 }
2029 }
2029 }
2030 }
2031 }
2032 }
2033 }
2034 }
2035 }
2036 }
2037 }
2038 }
2039 }
2039 }
2040 }
2041 }
2042 }
2043 }
2044 }
2045 }
2046 }
2047 }
2048 }
2049 }
2049 }
2050 }
2051 }
2052 }
2053 }
2054 }
2055 }
2056 }
2057 }
2058 }
2059 }
2059 }
2060 }
2061 }
2062 }
2063 }
2064 }
2065 }
2066 }
2067 }
2068 }
2069 }
2069 }
2070 }
2071 }
2072 }
2073 }
2074 }
2075 }
2076 }
2077 }
2077 }
2078 }
2078 }
2079 }
2079 }
2080 }
2081 }
2082 }
2083 }
2084 }
2085 }
2085 }
2086 }
2086 }
2087 }
2087 }
2088 }
2088 }
2089 }
2089 }
2090 }
2091 }
2092 }
2093 }
2094 }
2095 }
2095 }
2096 }
2096 }
2097 }
2097 }
2098 }
2098 }
2099 }
2099 }
2100 }
2101 }
2102 }
2103 }
2104 }
2105 }
2106 }
2107 }
2108 }
2109 }
2109 }
2110 }
2111 }
2112 }
2113 }
2114 }
2115 }
2116 }
2117 }
2118 }
2119 }
2119 }
2120 }
2121 }
2122 }
2123 }
2124 }
2125 }
2126 }
2127 }
2128 }
2129 }
2129 }
2130 }
2131 }
2132 }
2133 }
2134 }
2135 }
2136 }
2137 }
2138 }
2139 }
2139 }
2140 }
2141 }
2142 }
2143 }
2144 }
2145 }
2146 }
2147 }
2148 }
2149 }
2149 }
2150 }
2151
```

```

263     } catch (UserException e) {
264         System.out.println("Exception: " + e.getMessage());
265         session.setAttribute("errorMessage", "error while Pulling data from DB");
266         return new ModelAndView("error");
267     }
268 } else {
269     return new ModelAndView("redirect-Page");
270 }
271 } else {
272     return new ModelAndView("redirect-Page");
273 }
274 }
275 }
276 }
277 }
278 // View Sports / Academy - Post
279 @RequestMapping(value = "admin/view", method = RequestMethod.POST)
280 protected ModelAndView view(HttpServletRequest request) throws Exception {
281     HttpSession session = (HttpSession) request.getSession();
282     try {
283         String viewValue = request.getParameter("optradio");
284         if (viewValue.equalsIgnoreCase("sp")) {
285             List<Sports> sports = sportsDAO.list();
286             if (sports == null) {
287                 System.out.println("Issue while retrieving Sports at this point .Try again Later");
288                 session.setAttribute("errorMessage", "Issue while adding Sports at this point .Try again Later");
289                 return new ModelAndView("error");
290             }
291             // session.setAttribute("sportset", sports);
292             session.setAttribute("optradio", viewValue);
293             session.setAttribute("url", "https://drive.google.com/drive/my-drive");
294             return new ModelAndView("search-result", "spList", sports);
295         } else {
296             List<Academy> academy = academyDAO.list();
297             if (academy == null) {
298                 System.out.println("Issue while retrieving Academy at this point .Try again Later");
299                 session.setAttribute("errorMessage", "Issue while adding Academy at this point .Try again Later");
300                 return new ModelAndView("error");
301             }
302             // session.setAttribute("academyset", academy);
303             session.setAttribute("optradio", viewValue);
304             return new ModelAndView("academyList", "aList", academy);
305         }
306     } catch (UserException e) {
307         System.out.println("Exception: " + e.getMessage());
308         session.setAttribute("errorMessage", "error While Pulling data from DB");
309         return new ModelAndView("error");
310     }
311 }
312 }
313 }
314 }
315 }

```

Line 315, Column 1 Tab Size: 4 Java

2. Endusercontroller

```

1 package com.beongame.top.controllers;
2
3 import java.util.ArrayList;
4 import java.util.List;
5
6 import javax.mail.NoSuchProviderException;
7 import javax.servlet.http.HttpServletRequest;
8 import javax.servlet.http.HttpSession;
9
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.beans.factory.annotation.Qualifier;
12 import org.springframework.stereotype.Controller;
13 import org.springframework.web.bind.annotation.RequestMapping;
14 import org.springframework.web.bind.annotation.RequestMethod;
15 import org.springframework.web.servlet.ModelAndView;
16
17 import com.beongame.top.dao.AcademyDAO;
18 import com.beongame.top.dao.BookingDAO;
19 import com.beongame.top.dao.ServiceproviderDAO;
20 import com.beongame.top.dao.SportsDAO;
21 import com.beongame.top.dao.SportsacademyDAO;
22 import com.beongame.top.exception.UserException;
23 import com.beongame.top.pojo.Academy;
24 import com.beongame.top.pojo.Booking;
25 import com.beongame.top.pojo.EmailDetails;
26 import com.beongame.top.pojo.Enduser;
27 import com.beongame.top.pojo.Person;
28 import com.beongame.top.pojo.Serviceprovider;
29 import com.beongame.top.pojo.SportsAcademy;
30 import com.beongame.top.random.EmailDetails;
31
32 @Controller
33 @RequestMapping("/enduser/*")
34 public class Endusercontroller {
35
36     @Autowired
37     @Qualifier("academyDAO")
38     AcademyDAO academyDAO;
39
40     @Autowired
41     @Qualifier("spcDAO")
42     SportsacademyDAO spcDAO;
43
44     @Autowired
45     @Qualifier("bookingDAO")
46     BookingDAO bookingDAO;
47
48     @Autowired
49     @Qualifier("svcpDAO")
50     ServiceproviderDAO svcpDAO;
51
52     // Add Sports
53

```

Line 218, Column 37 Tab Size: 4 Java

```
package com.beongame.top.controllers; UNREGISTERED
untitled package com.beongame.top.controllers;
54     @Autowired
55     @Qualifier("sportsDAO")
56     SportsDAO sportsDAO;
57
58     @RequestMapping(value = "enduser/academySearch", method = RequestMethod.GET)
59     protected ModelAndView searchAcademy(HttpServletRequest request) throws Exception {
60         HttpSession session = (HttpSession) request.getSession();
61
62         if (session.getAttribute("ltype") == null) {
63             return new ModelAndView("redirect-Page");
64         } else {
65
66             String ltype = (String) session.getAttribute("ltype");
67             String enduser = "User";
68
69             if (ltype.equals(enduser)) {
70                 Person user = (Person) session.getAttribute("Enduser");
71                 if (user != null) {
72                     try {
73                         List<SportsAcademy> sportslist = sportsDAO.list();
74                         List<Academy> a = new ArrayList<Academy>();
75                         for (SportsAcademy sa : sportslist) {
76                             if (!a.contains(sa.getAcademy())) {
77                                 String str = session.getAttribute("searchSport");
78                                 String st1 = sa.getGame();
79                                 if (st1.equals(str)) {
80                                     a.add(sa.getAcademy());
81                                 }
82                             }
83                         }
84                         if (a.isEmpty()) {
85                             session.setAttribute("errorMessage",
86                                     "No Academy exists for the given sport. We are working hard to include More academy of your interest");
87                             return new ModelAndView("sorry-Page");
88                         } else {
89                             return new ModelAndView("academySearch", "alist", a);
90                         }
91                     } catch (UserException e) {
92                         System.out.println("Exception: " + e.getMessage());
93                         session.setAttribute("errorMessage", "error while Adding Sports");
94                         return new ModelAndView("error");
95                     }
96                 } else {
97                     return new ModelAndView("redirect-Page");
98                 }
99             } else {
100                 return new ModelAndView("redirect-Page");
101             }
102         }
103     }
104 }
105
106 }
```

Line 218, Column 37 Tab Size: 4 Java

```
package com.beongame.top.controllers; UNREGISTERED
untitled package com.beongame.top.controllers;
107
108     @RequestMapping(value = "enduser/academySearch", method = RequestMethod.POST)
109     protected ModelAndView searchAcademy(HttpServletRequest request) throws Exception {
110         HttpSession session = (HttpSession) request.getSession();
111         try {
112
113             List<SportsAcademy> sportslist = sportsDAO.list();
114             List<Academy> a = new ArrayList<Academy>();
115             for (SportsAcademy sa : sportslist) {
116                 if (!a.contains(sa.getAcademy())) {
117                     String str = request.getParameter("sportname");
118                     session.setAttribute("searchSport", str);
119                     String st1 = sa.getGame();
120                     if (st1.equals(str)) {
121                         a.add(sa.getAcademy());
122                     }
123                 }
124             }
125             if (a.isEmpty()) {
126                 session.setAttribute("errorMessage",
127                         "No Academy exists for the given sport. We are working hard to include More academy of your interest");
128                 return new ModelAndView("sorry-Page");
129             } else {
130                 return new ModelAndView("academySearch", "alist", a);
131             }
132         } catch (UserException e) {
133             System.out.println("Exception: " + e.getMessage());
134             session.setAttribute("errorMessage", "error While Adding Sports");
135             return new ModelAndView("error");
136         }
137     }
138
139
140     @RequestMapping(value = "enduser/selectSport", method = RequestMethod.GET)
141     protected ModelAndView UpdateSports(HttpServletRequest request) throws NoSuchProviderException {
142         HttpSession session = (HttpSession) request.getSession();
143         if (session.getAttribute("ltype") == null) {
144             return new ModelAndView("redirect-Page");
145         } else {
146             String str1 = request.getParameter("spID");
147             String ltype = (String) session.getAttribute("ltype");
148             String enduser = "User";
149             if (ltype.equals(enduser)) {
150                 Person user = (Person) session.getAttribute("Enduser");
151                 if (user != null) {
152                     try {
153                         List<SportsAcademy> sportslist = sportsDAO.list();
154                         SportsAcademy tmp = new SportsAcademy();
155                         Academy acad = new Academy();
156                         Booking bo = new Booking();
157                         boolean flag = false;
158                         for (SportsAcademy sa : sportslist) {
159                             if (sa.getAcademy().getAcademyID().equals(str1)) {
160                                 tmp = sa;
161                                 flag = true;
162                             }
163                         }
164                         if (flag) {
165                             acad.setAcademyID(tmp.getAcademyID());
166                             acad.setAcademyName(tmp.getAcademyName());
167                             acad.setGame(tmp.getGame());
168                             bo.setAcademy(acad);
169                             bo.setBookingDate(tmp.getBookingDate());
170                             bo.setBookingTime(tmp.getBookingTime());
171                             bo.setBookingStatus(tmp.getBookingStatus());
172                             bo.setBookingTime(tmp.getBookingTime());
173                             bo.setBookingStatus(tmp.getBookingStatus());
174                             bo.setBookingTime(tmp.getBookingTime());
175                             bo.setBookingStatus(tmp.getBookingStatus());
176                             bo.setBookingTime(tmp.getBookingTime());
177                             bo.setBookingStatus(tmp.getBookingStatus());
178                             bo.setBookingTime(tmp.getBookingTime());
179                             bo.setBookingStatus(tmp.getBookingStatus());
180                             bo.setBookingTime(tmp.getBookingTime());
181                             bo.setBookingStatus(tmp.getBookingStatus());
182                             bo.setBookingTime(tmp.getBookingTime());
183                             bo.setBookingStatus(tmp.getBookingStatus());
184                             bo.setBookingTime(tmp.getBookingTime());
185                             bo.setBookingStatus(tmp.getBookingStatus());
186                             bo.setBookingTime(tmp.getBookingTime());
187                             bo.setBookingStatus(tmp.getBookingStatus());
188                             bo.setBookingTime(tmp.getBookingTime());
189                             bo.setBookingStatus(tmp.getBookingStatus());
190                             bo.setBookingTime(tmp.getBookingTime());
191                             bo.setBookingStatus(tmp.getBookingStatus());
192                             bo.setBookingTime(tmp.getBookingTime());
193                             bo.setBookingStatus(tmp.getBookingStatus());
194                             bo.setBookingTime(tmp.getBookingTime());
195                             bo.setBookingStatus(tmp.getBookingStatus());
196                             bo.setBookingTime(tmp.getBookingTime());
197                             bo.setBookingStatus(tmp.getBookingStatus());
198                             bo.setBookingTime(tmp.getBookingTime());
199                             bo.setBookingStatus(tmp.getBookingStatus());
200                             bo.setBookingTime(tmp.getBookingTime());
201                             bo.setBookingStatus(tmp.getBookingStatus());
202                             bo.setBookingTime(tmp.getBookingTime());
203                             bo.setBookingStatus(tmp.getBookingStatus());
204                             bo.setBookingTime(tmp.getBookingTime());
205                             bo.setBookingStatus(tmp.getBookingStatus());
206                             bo.setBookingTime(tmp.getBookingTime());
207                             bo.setBookingStatus(tmp.getBookingStatus());
208                             bo.setBookingTime(tmp.getBookingTime());
209                             bo.setBookingStatus(tmp.getBookingStatus());
210                             bo.setBookingTime(tmp.getBookingTime());
211                             bo.setBookingStatus(tmp.getBookingStatus());
212                             bo.setBookingTime(tmp.getBookingTime());
213                             bo.setBookingStatus(tmp.getBookingStatus());
214                             bo.setBookingTime(tmp.getBookingTime());
215                             bo.setBookingStatus(tmp.getBookingStatus());
216                             bo.setBookingTime(tmp.getBookingTime());
217                             bo.setBookingStatus(tmp.getBookingStatus());
218                             bo.setBookingTime(tmp.getBookingTime());
219                             bo.setBookingStatus(tmp.getBookingStatus());
220                             bo.setBookingTime(tmp.getBookingTime());
221                             bo.setBookingStatus(tmp.getBookingStatus());
222                             bo.setBookingTime(tmp.getBookingTime());
223                             bo.setBookingStatus(tmp.getBookingStatus());
224                             bo.setBookingTime(tmp.getBookingTime());
225                             bo.setBookingStatus(tmp.getBookingStatus());
226                             bo.setBookingTime(tmp.getBookingTime());
227                             bo.setBookingStatus(tmp.getBookingStatus());
228                             bo.setBookingTime(tmp.getBookingTime());
229                             bo.setBookingStatus(tmp.getBookingStatus());
230                             bo.setBookingTime(tmp.getBookingTime());
231                             bo.setBookingStatus(tmp.getBookingStatus());
232                             bo.setBookingTime(tmp.getBookingTime());
233                             bo.setBookingStatus(tmp.getBookingStatus());
234                             bo.setBookingTime(tmp.getBookingTime());
235                             bo.setBookingStatus(tmp.getBookingStatus());
236                             bo.setBookingTime(tmp.getBookingTime());
237                             bo.setBookingStatus(tmp.getBookingStatus());
238                             bo.setBookingTime(tmp.getBookingTime());
239                             bo.setBookingStatus(tmp.getBookingStatus());
240                             bo.setBookingTime(tmp.getBookingTime());
241                             bo.setBookingStatus(tmp.getBookingStatus());
242                             bo.setBookingTime(tmp.getBookingTime());
243                             bo.setBookingStatus(tmp.getBookingStatus());
244                             bo.setBookingTime(tmp.getBookingTime());
245                             bo.setBookingStatus(tmp.getBookingStatus());
246                             bo.setBookingTime(tmp.getBookingTime());
247                             bo.setBookingStatus(tmp.getBookingStatus());
248                             bo.setBookingTime(tmp.getBookingTime());
249                             bo.setBookingStatus(tmp.getBookingStatus());
250                             bo.setBookingTime(tmp.getBookingTime());
251                             bo.setBookingStatus(tmp.getBookingStatus());
252                             bo.setBookingTime(tmp.getBookingTime());
253                             bo.setBookingStatus(tmp.getBookingStatus());
254                             bo.setBookingTime(tmp.getBookingTime());
255                             bo.setBookingStatus(tmp.getBookingStatus());
256                             bo.setBookingTime(tmp.getBookingTime());
257                             bo.setBookingStatus(tmp.getBookingStatus());
258                             bo.setBookingTime(tmp.getBookingTime());
259                             bo.setBookingStatus(tmp.getBookingStatus());
260                             bo.setBookingTime(tmp.getBookingTime());
261                             bo.setBookingStatus(tmp.getBookingStatus());
262                             bo.setBookingTime(tmp.getBookingTime());
263                             bo.setBookingStatus(tmp.getBookingStatus());
264                             bo.setBookingTime(tmp.getBookingTime());
265                             bo.setBookingStatus(tmp.getBookingStatus());
266                             bo.setBookingTime(tmp.getBookingTime());
267                             bo.setBookingStatus(tmp.getBookingStatus());
268                             bo.setBookingTime(tmp.getBookingTime());
269                             bo.setBookingStatus(tmp.getBookingStatus());
270                             bo.setBookingTime(tmp.getBookingTime());
271                             bo.setBookingStatus(tmp.getBookingStatus());
272                             bo.setBookingTime(tmp.getBookingTime());
273                             bo.setBookingStatus(tmp.getBookingStatus());
274                             bo.setBookingTime(tmp.getBookingTime());
275                             bo.setBookingStatus(tmp.getBookingStatus());
276                             bo.setBookingTime(tmp.getBookingTime());
277                             bo.setBookingStatus(tmp.getBookingStatus());
278                             bo.setBookingTime(tmp.getBookingTime());
279                             bo.setBookingStatus(tmp.getBookingStatus());
280                             bo.setBookingTime(tmp.getBookingTime());
281                             bo.setBookingStatus(tmp.getBookingStatus());
282                             bo.setBookingTime(tmp.getBookingTime());
283                             bo.setBookingStatus(tmp.getBookingStatus());
284                             bo.setBookingTime(tmp.getBookingTime());
285                             bo.setBookingStatus(tmp.getBookingStatus());
286                             bo.setBookingTime(tmp.getBookingTime());
287                             bo.setBookingStatus(tmp.getBookingStatus());
288                             bo.setBookingTime(tmp.getBookingTime());
289                             bo.setBookingStatus(tmp.getBookingStatus());
290                             bo.setBookingTime(tmp.getBookingTime());
291                             bo.setBookingStatus(tmp.getBookingStatus());
292                             bo.setBookingTime(tmp.getBookingTime());
293                             bo.setBookingStatus(tmp.getBookingStatus());
294                             bo.setBookingTime(tmp.getBookingTime());
295                             bo.setBookingStatus(tmp.getBookingStatus());
296                             bo.setBookingTime(tmp.getBookingTime());
297                             bo.setBookingStatus(tmp.getBookingStatus());
298                             bo.setBookingTime(tmp.getBookingTime());
299                             bo.setBookingStatus(tmp.getBookingStatus());
299 }
```

Line 218, Column 37 Tab Size: 4 Java

```
159 package com.beongame.top.controllers;
160
161     for (SportsAcademy sa : sportslist) {
162         String str2 = String.valueOf(sa.getSpID());
163         if (str2.equals(str1)) {
164             sa.setAvailability("N");
165             spacDAO.save(sa);
166             flag = true;
167             tmp = sa;
168             acad = sa.getAcademy();
169         }
170     }
171     if (flag == true) {
172         Enduser eu = (Enduser) session.getAttribute("Enduser");
173         Serviceprovider spID = svcrpDAO.getObj(acad);
174         bo = bookingDAO.addBooking(eu, tmp, spID);
175         String message = "Congratz! Your booking has been confirmed with "+acad.getAcademyName()+" . Thank you for using BeOnGame.";
176         // Send Mail
177         Email emailDetails = new Email();
178         emailDetails.setFromEmail("vishnudhatri@gmail.com");
179         emailDetails.setToEmail(user.getEmail());
180         emailDetails.setSubject("Booking Confirmation");
181         emailDetails.setMessage(message);
182         EmailDetails email = new EmailDetails();
183         email.sendMail(emailDetails);
184     }
185
186     if (bo == null) {
187         System.out.println("Issue while booking .Try again Later");
188         session.setAttribute("errorMessage", "Issue while booking .Try again Later");
189         return new ModelAndView("sorry-Page");
190     }
191
192
193     session.setAttribute("SuccessMessage", "You successfully made a booking. Just Be -on- Game.");
194     return new ModelAndView("success");
195
196 } catch (UserException e) {
197     System.out.println("Exception: " + e.getMessage());
198     session.setAttribute("errorMessage", "Error while Booking your Slot.Please Try again later");
199     return new ModelAndView("error");
200 }
201 } else {
202     return new ModelAndView("redirect-Page");
203 }
204 } else {
205     return new ModelAndView("redirect-Page");
206 }
207 }
208
209
210 @RequestMapping(value = "enduser/sportsSearch", method = RequestMethod.GET)
211 protected ModelAndView searchSports(HttpServletRequest request) {
}

```

```
196     } catch (UserException e) {
197         System.out.println("Exception: " + e.getMessage());
198         session.setAttribute("errorMessage", "Error while Booking your Slot.Please Try again later");
199         return new ModelAndView("error");
200     }
201     } else {
202         return new ModelAndView("redirect=Page");
203     }
204 } else {
205     return new ModelAndView("redirect=Page");
206 }
207 }
208 }
209
210 @RequestMapping(value = "enduser/sportsSearch", method = RequestMethod.GET)
211 protected ModelAndView searchSports(HttpServletRequest request) {
212     HttpSession session = (HttpSession) request.getSession();
213     if (session.getAttribute("ltype") == null) {
214         return new ModelAndView("redirect=Page");
215     } else {
216         String str1 = request.getParameter("acadID");
217         String ltype = (String) session.getAttribute("ltype");
218         String enduser = "User";
219         if (ltype.equals(enduser)) {
220             Person user = (Person) session.getAttribute("Enduser");
221             if (user != null) {
222                 try {
223                     List<SportsAcademy> sportslist = spadDAO.list();
224                     List<SportsAcademy> sportslistnew = new ArrayList<SportsAcademy>();
225                     for (SportsAcademy sa : sportslist) {
226                         String str2 = String.valueOf(sa.getAcademy().getAcademyID());
227                         if (str2.equals(str1)) {
228                             sportslistnew.add(sa);
229                         }
230                     }
231                     return new ModelAndView("slotsView-user", "splist", sportslistnew);
232                 } catch (UserException e) {
233                     System.out.println("Exception: " + e.getMessage());
234                     session.setAttribute("errorMessage", "error while Adding Sports");
235                     return new ModelAndView("error");
236                 }
237             } else {
238                 return new ModelAndView("redirect=Page");
239             }
240         } else {
241             return new ModelAndView("redirect=Page");
242         }
243     }
244 }
245 }
246 }
247 }
248 }
```

3. Servicecontroller

```
package com.beongame.top.controllers;
import java.util.ArrayList;
import java.util.List;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpSession;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.beans.factory.annotation.Qualifier;
import org.springframework.stereotype.Controller;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.servlet.ModelAndView;
import com.beongame.top.dao.AcademyDAO;
import com.beongame.top.dao.BookingDAO;
import com.beongame.top.dao.EnduserDAO;
import com.beongame.top.dao.ServiceproviderDAO;
import com.beongame.top.dao.SportsDAO;
import com.beongame.top.dao.SportsacademyDAO;
import com.beongame.top.exception.UserException;
import com.beongame.top.pojo.Academy;
import com.beongame.top.pojo.Person;
import com.beongame.top.pojo.Serviceprovider;
import com.beongame.top.pojo.Sports;
import com.beongame.top.pojo.SportsAcademy;
@Controller
@RequestMapping("/serviceprovider/*")
public class Service {
    @Autowired
    @Qualifier("userDAO")
    UserDAO userDAO;
    @Autowired
    @Qualifier("svcpDAO")
    ServiceproviderDAO svcpDAO;
    @Autowired
    @Qualifier("euserDAO")
    EnduserDAO euserDAO;
    @Autowired
    @Qualifier("sportsDAO")
    SportsDAO sportsDAO;
    @Autowired
    @Qualifier("academyDAO")
    AcademyDAO academyDAO;
}
Line 252, Column 1
```

```
package com.beongame.top.controllers;
import com.beongame.top.controllers;
import com.beongame.top.controllers;
@RequestMapping(value = "serviceprovider/bookings", method = RequestMethod.GET)
protected ModelAndView bookings(HttpServletRequest request) throws Exception {
    HttpSession session = (HttpSession) request.getSession();
    if (session.getAttribute("ltype") == null) {
        return new ModelAndView("redirect-Page");
    } else {
        String ltype = (String) session.getAttribute("ltype");
        String serviceProvider = "ServiceProvider";
        if (ltype.equals(serviceProvider)) {
            Person user = (Person) session.getAttribute("puser");
            if (user == null) {
                session.setAttribute("SuccessMessage", "Successfully Created Slots for your academy");
                return new ModelAndView("success");
            } else {
                return new ModelAndView("redirect-Page");
            }
        } else {
            return new ModelAndView("redirect-Page");
        }
    }
}
@RequestMapping(value = "serviceprovider/sportsCreate", method = RequestMethod.GET)
protected ModelAndView signUpse(HttpServletRequest request) throws Exception {
    HttpSession session = (HttpSession) request.getSession();
    if (session.getAttribute("ltype") == null) {
        return new ModelAndView("redirect-Page");
    } else {
        String ltype = (String) session.getAttribute("ltype");
        String serviceProvider = "ServiceProvider";
        if (ltype.equals(serviceProvider)) {
            String str1 = request.getParameter("spID");
            try {
                List<SportsAcademy> sportslist = spcadAO.list();
                for (SportsAcademy sa : sportslist) {
                    String str2 = String.valueOf(sa.getSpID());
                    if (str2.equals(str1)) {
                        spcadAO.delete(sa);
                    }
                }
            }
        }
    }
}
@RequestMapping(value = "serviceprovider/update", method = RequestMethod.GET)
protected ModelAndView update(HttpServletRequest request) throws Exception {
    HttpSession session = (HttpSession) request.getSession();
    if (session.getAttribute("ltype") == null) {
        return new ModelAndView("redirect-Page");
    } else {
        String ltype = (String) session.getAttribute("ltype");
        String serviceProvider = "ServiceProvider";
        if (ltype.equals(serviceProvider)) {
            String str1 = request.getParameter("spID");
            try {
                List<SportsAcademy> sportslist = spcadAO.list();
                for (SportsAcademy sa : sportslist) {
                    String str2 = String.valueOf(sa.getSpID());
                    if (str2.equals(str1)) {
                        spcadAO.delete(sa);
                    }
                }
            }
        }
    }
}
Line 252, Column 1
```

```
159     return new ModelAndView("slotsView-sp", "splist", sportslistnew);
160 } catch (UserException e) {
161     System.out.println("Exception: " + e.getMessage());
162     session.setAttribute("errorMessage", "error while Adding Sports");
163     return new ModelAndView("error");
164 }
165 }
166 } catch (UserException e) {
167     System.out.println("Exception: " + e.getMessage());
168     session.setAttribute("errorMessage", "error while Pulling data from DB");
169     return new ModelAndView("error");
170 }
171 }
172 }
173 }
174 }
175 }
176 }
177 @RequestMapping(value = "serviceprovider/addOrDelete", method = RequestMethod.POST)
178 protected ModelAndView addOrDel(HttpServletRequest request) throws Exception {
179     HttpSession session = (HttpSession) request.getSession();
180     try {
181         String viewValue = request.getParameter("oprtradio");
182         session.setAttribute("oprtradio", viewValue);
183         if (viewValue.equalsIgnoreCase("sp")) {
184             Person person = (Person) session.getAttribute("spuser");
185             ServiceProvider user = svpcrDAO.getObjsp(person.getPersonID());
186             Academy acad = academyDAO.getAcademy(user.getAcadID());
187             if (acad == null) {
188                 session.setAttribute("errorMessage", "Issue with fetching results");
189                 return new ModelAndView("userNotFound");
190             }
191             List<Sports> spList = sportsDAO.list();
192             session.setAttribute("spList", spList);
193             session.setAttribute("Academyobj", acad);
194             return new ModelAndView("serviceProviderCreate", "user", user);
195         } else {
196             Person person = (Person) session.getAttribute("spuser");
197             ServiceProvider user = svpcrDAO.getObjsp(person.getPersonID());
198             Academy acad = academyDAO.getAcademy(user.getAcadID());
199             String str1 = String.valueOf(acad.getAcademyID());
200             try {
201                 List<SportsAcademy> sportslist = spacDAO.list();
202                 List<SportsAcademy> sportslistnew = new ArrayList<SportsAcademy>();
203                 for (SportsAcademy sa : sportslist) {
204                     String str2 = String.valueOf(sa.getAcademy().getAcademyID());
205                     if (str2.equals(str1)) {
206                         sportslistnew.add(sa);
207                     }
208                 }
209             }
210             return new ModelAndView("slotsView-en" "enlist", "sportslistnew");
211         }
212     }
213 }
```

```

package com.beongame.top.controllers;
package com.beongame.top.controllers;
package com.beongame.top.controllers;

200    Academy acad = academyDAO.getAcademy(user.getAcadId());
201    String str1 = String.valueOf(acad.getAcademyID());
202    try {
203        List<SportsAcademy> sportsList = spadAO.list();
204        List<SportsAcademy> sportsListNew = new ArrayList<SportsAcademy>();
205        for (SportsAcademy sa : sportsList) {
206            String str2 = String.valueOf(sa.getAcademyID());
207            if (str2.equals(str1)) {
208                sportsListNew.add(sa);
209            }
210        }
211        return new ModelAndView("slotsView-sp", "splist", sportsListNew);
212    } catch (UserException e) {
213        System.out.println("Exception: " + e.getMessage());
214        session.setAttribute("errorMessage", "error while Adding Sports");
215        return new ModelAndView("error");
216    }
217    } catch (UserException e) {
218        System.out.println("Exception: " + e.getMessage());
219        session.setAttribute("errorMessage", "error While Pulling data from DB");
220        return new ModelAndView("error");
221    }
222}
223}
224
225 // Create slots - Post
226 @RequestMapping(value = "serviceprovider/sportsCreate", method = RequestMethod.POST)
227 protected String signUpUser(HttpServletRequest request) throws Exception {
228     HttpSession session = (HttpSession) request.getSession();
229     int noSlots = Integer.parseInt(request.getParameter("spSlots"));
230     while (noSlots != 0) {
231         try {
232             SportsAcademy spacedAcad = spadAO.addSlots(request.getParameter("spPrice"), String.valueOf(noSlots),
233             request.getParameter("spName"), (Academy) session.getAttribute("Academyobj"));
234             if (spacedAcad != null) {
235                 System.out.println("Issue While Creating User at this point .Try again Later");
236                 session.setAttribute("errorMessage", "Issue While Creating User at this point .Try again Later");
237                 return "error";
238             }
239         } catch (UserException e) {
240             System.out.println("Exception: " + e.getMessage());
241             session.setAttribute("errorMessage", "error while Signup");
242             return "error";
243         }
244         noSlots--;
245     }
246     session.setAttribute("SuccessMessage", "Successfully Created Slots for your academy");
247     return "success";
248 }
249 }
250
251 }
252

```

Line 252, Column 1

Tab Size: 4 Java

4. Usercontroller

```

package com.beongame.top.controllers;
package com.beongame.top.controllers;
package com.beongame.top.controllers;

1 package com.beongame.top.controllers;
2
3 import java.util.List;
4
5 import javax.servlet.http.HttpServletRequest;
6 import javax.servlet.http.HttpSession;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.beans.factory.annotation.Qualifier;
9 import org.springframework.http.MediaType;
10 import org.springframework.stereotype.Controller;
11 import org.springframework.web.bind.annotation.RequestMapping;
12 import org.springframework.web.bind.annotation.RequestMethod;
13 import org.springframework.web.bind.annotation.ResponseBody;
14 import org.springframework.web.servlet.ModelAndView;
15
16 import com.beongame.top.dao.AcademyDAO;
17 import com.beongame.top.dao.BookingDAO;
18 import com.beongame.top.dao.EnduserDAO;
19 import com.beongame.top.dao.ServiceproviderDAO;
20 import com.beongame.top.dao.SportsDAO;
21 import com.beongame.top.dao.SportsacademyDAO;
22 import com.beongame.top.dao.UserDAO;
23 import com.beongame.top.exception.UserException;
24 import com.beongame.top.pojo.Academy;
25 import com.beongame.top.pojo.Booking;
26 import com.beongame.top.pojo.Enduser;
27 import com.beongame.top.pojo.Person;
28 import com.beongame.top.pojo.Serviceprovider;
29 import com.beongame.top.pojo.Sports;
30 import com.beongame.top.pojo.SportsAcademy;
31
32 @Controller
33 @RequestMapping("/user/*")
34 public class Usercontroller {
35     @Autowired
36     @Qualifier("userDao")
37     UserDAO userDAO;
38
39     @Autowired
40     @Qualifier("svcpDAO")
41     ServiceproviderDAO svcpDAO;
42
43     @Autowired
44     @Qualifier("euserDAO")
45     EnduserDAO euserDAO;
46
47     @Autowired
48     @Qualifier("sportsDAO")
49     SportsDAO sportsDAO;
50
51     @Autowired
52     @Qualifier("academyDAO")
53     AcademyDAO academyDAO;

```

Line 256, Column 1

Tab Size: 4 Java

```
package com.beongame.top.controllers;
package com.beongame.top.controllers;
package com.beongame.top.controllers;

@RequestMapping(value = "/user/login", method = RequestMethod.POST)
protected ModelAndView signInUser(HttpServletRequest request) throws Exception {
    HttpSession session = (HttpSession) request.getSession();
    try {
        Booking bk = bookingDAO.get();
        if (bk == null) {
            System.out.println("Booking");
        }
        SportsAcademy spac = spacDAO.get();
        if (spac == null) {
            System.out.println("Booking");
        }

        String loginType = request.getParameter("loginType");
        if (loginType.equalsIgnoreCase("Admin")) {
            Person user = userDao.get(request.getParameter("email"), request.getParameter("password"),
                request.getParameter("loginType"));

            if (user == null) {
                System.out.println("UserName/Password does not exist");
                session.setAttribute("errorMessage", "UserName/Password does not exist");
                return new ModelAndView("userNotFound");
            }
            session.setAttribute("user", user);
            session.setAttribute("type", request.getParameter("loginType"));
            return new ModelAndView("admin", "user", user);
        } else if (loginType.equalsIgnoreCase("User")) {

            List<Sports> spList = sportsDAO.list();
            Enduser user = euserDao.get(request.getParameter("email"), request.getParameter("password"),
                loginType);

            if (user == null) {
                System.out.println("UserName/Password does not exist");
                session.setAttribute("errorMessage", "UserName/Password does not exist");
                return new ModelAndView("userNotFound");
            }
            session.setAttribute("type", request.getParameter("loginType"));
            session.setAttribute("Enduser", user);
            return new ModelAndView("userPage", "spList", spList);
        } else {
            ServiceProvider user = svcpDAO.get(request.getParameter("email"), request.getParameter("password"),
                request.getParameter("loginType"));

            if (user == null) {
                System.out.println("UserName/Password does not exist");
                session.setAttribute("errorMessage", "UserName/Password does not exist");
                return new ModelAndView("userNotFound");
            }
            Academy Acad = acadDAO.getAcad(user.getCategory());
        }
    } catch (Exception e) {
        e.printStackTrace();
    }
}

Line 256, Column 1
```

```
package com.beongame.top.controllers;
package com.beongame.top.controllers;
package com.beongame.top.controllers;

159
160     Academy acad = academyDAO.getAcademy(user.getAcadId());
161     if (acad == null) {
162         session.setAttribute("errorMessage", "Issue in Sigin");
163         return new ModelAndView("userNotFound");
164     }
165
166     List<Sports> spList = sportsDAO.list();
167     session.setAttribute("spList", spList);
168     session.setAttribute("Academobj", acad);
169     session.setAttribute("suser", user);
170     session.setAttribute("ltype", request.getParameter("loginType"));
171     return new ModelAndView("serviceproviderPage", "user", user);
172 }
173
174 } catch (UserException e) {
175     System.out.println("Exception: " + e.getMessage());
176     session.setAttribute("errorMessage", "error while login");
177     return new ModelAndView("userNotFound");
178 }
179
180 // User SignUp - Get
181 @RequestMapping(value = "/user/signup", method = RequestMethod.GET)
182 protected String signUpUser(HttpServletRequest request) throws Exception {
183     HttpSession session = (HttpSession) request.getSession();
184     try {
185         Enduser user = userDAO.signup(request.getParameter("firstName"), request.getParameter("lastName"),
186             request.getParameter("age"), request.getParameter("emailId"), request.getParameter("contactNo"),
187             request.getParameter("userName"), request.getParameter("password"),
188             request.getParameter("loginType"));
189         if (user == null) {
190             System.out.println("Issue While Creating User at this point .Try again Later");
191             session.setAttribute("errorMessage", "Issue While Creating User at this point .Try again Later");
192             return "error";
193         }
194         session.setAttribute("user", user);
195         return "home";
196     }
197     catch (UserException e) {
198         System.out.println("Exception: " + e.getMessage());
199         session.setAttribute("errorMessage", "error while SignUp");
200         return "error";
201     }
202 }
203
204 // Academy check
205 @RequestMapping(value = "/user/signup", method = RequestMethod.GET, produces = MediaType.TEXT_PLAIN_VALUE)
206 protected @ResponseBody String usercheck(HttpServletRequest request) throws Exception {
207     Person user = userDAO.getnew(request.getParameter("aid1"));
208     if (user != null) {
209         return "Email Already Registered";
210     }
211     return "";
212 }
213
214
215
216
217
218
219
220 // User signup - Post
221 @RequestMapping(value = "/user/signup", method = RequestMethod.POST)
222 protected ModelAndView signUpUser(HttpServletRequest request) throws Exception {
223     HttpSession session = (HttpSession) request.getSession();
224     try {
225         Person user = userDAO.getnew(request.getParameter("emailId"));
226         if (user == null) {
227             Enduser user = userDAO.signup(request.getParameter("firstName"), request.getParameter("lastName"),
228                 request.getParameter("age"), request.getParameter("emailId"), request.getParameter("contactNo"),
229                 request.getParameter("userName"), request.getParameter("password"),
230                 request.getParameter("loginType"));
231             if (user == null) {
232                 System.out.println("Issue While Creating User at this point .Try again Later");
233                 session.setAttribute("errorMessage", "Issue While Creating User at this point .Try again Later");
234                 return new ModelAndView("error");
235             }
236             List<Sports> spList = sportsDAO.list();
237             session.setAttribute("ltype", request.getParameter("loginType"));
238             session.setAttribute("Enduser", user);
239             return new ModelAndView("userPage", "spList", spList);
240         }
241         else {
242             session.setAttribute("errorMessage", "Email Already Registered");
243             return new ModelAndView("error");
244         }
245
246     } catch (UserException e) {
247         System.out.println("Exception: " + e.getMessage());
248         session.setAttribute("errorMessage", "error while SignUp");
249         return new ModelAndView("error");
250     }
251 }
252
253
254
255
256

```

```
package com.beongame.top.controllers;
package com.beongame.top.controllers;
package com.beongame.top.controllers;

204 }
205
206 // Academy check
207 @RequestMapping(value = "/user/signup", method = RequestMethod.GET, produces = MediaType.TEXT_PLAIN_VALUE)
208 protected @ResponseBody String usercheck(HttpServletRequest request) throws Exception {
209     Person user = userDAO.getnew(request.getParameter("aid1"));
210     if (user != null) {
211         return "Email Already Registered";
212     }
213     return "";
214 }
215
216
217
218
219
220 // User signup - Post
221 @RequestMapping(value = "/user/signup", method = RequestMethod.POST)
222 protected ModelAndView signUpUser(HttpServletRequest request) throws Exception {
223     HttpSession session = (HttpSession) request.getSession();
224     try {
225         Person user = userDAO.getnew(request.getParameter("emailId"));
226         if (user == null) {
227             Enduser user = userDAO.signup(request.getParameter("firstName"), request.getParameter("lastName"),
228                 request.getParameter("age"), request.getParameter("emailId"), request.getParameter("contactNo"),
229                 request.getParameter("userName"), request.getParameter("password"),
230                 request.getParameter("loginType"));
231             if (user == null) {
232                 System.out.println("Issue While Creating User at this point .Try again Later");
233                 session.setAttribute("errorMessage", "Issue While Creating User at this point .Try again Later");
234                 return new ModelAndView("error");
235             }
236             List<Sports> spList = sportsDAO.list();
237             session.setAttribute("ltype", request.getParameter("loginType"));
238             session.setAttribute("Enduser", user);
239             return new ModelAndView("userPage", "spList", spList);
240         }
241         else {
242             session.setAttribute("errorMessage", "Email Already Registered");
243             return new ModelAndView("error");
244         }
245
246     } catch (UserException e) {
247         System.out.println("Exception: " + e.getMessage());
248         session.setAttribute("errorMessage", "error while SignUp");
249         return new ModelAndView("error");
250     }
251 }
252
253
254
255
256

```