



# **BHASKARACHARYA NATIONAL INSTITUTE FOR SPACE APPLICATIONS AND GEO-INFORMATICS**

**WEEKLY PROGRESS REPORT (29/03/2024 – 04/04/2024)**

**WEEK 17**

<b>PROJECT NAME</b>	<b>Intern Management System</b>
---------------------	---------------------------------

**PROJECT DESCRIPTION :**

An Intern Management System (IMS) is a software application designed to automate the processes associated with managing interns within an organization.

**GROUP MEMBER :**

**NILKANTHKUMAR PATEL , ROHITKUMAR PRAJAPATI, KRISHNA CHAPLA**

**GROUP ID :**

**24RH4**

**GROUP GUIDE :**

**SIDHDHARTH PATEL**

**GROUP COORDINATOR :**

**SIDHDHARTH PATEL**

**COLLEGE GUIDE NAME :**

**Prof. Niyati Buch**

**COLLEGE GUIDE No. :**

**9428700496**

**COLLEGE GUIDE EMAIL. :**

**niyatibuch.ce@ddu.ac.in**

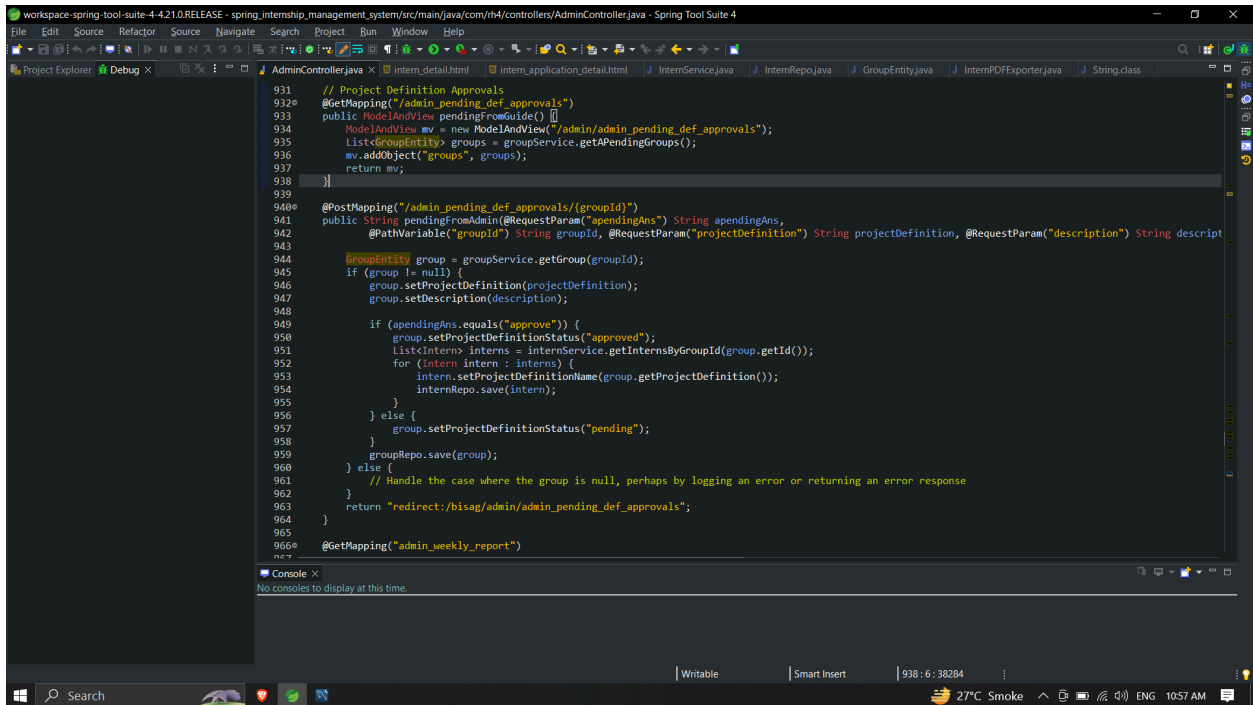
**COLLEGE NAME :**

**DHARMSINH DESAI UNIVERSITY**

<b>29/03/2024</b>	<b>Holiday (Good Friday)</b>
<b>30/03/2024</b>	<ul style="list-style-type: none"> <li>• Worked on project definition changes where admin can edit project definition and description. ( KRISHNA CHAPLA )</li> <li>• Made required changes in frontend. ( NILKANTH PATEL &amp; ROHIT PRAJAPATI )</li> </ul>
<b>31/03/2024</b>	<b>Holiday (Sunday)</b>
<b>01/04/2024</b>	<ul style="list-style-type: none"> <li>• Worked on file upload and manage sections and made other changes in frontend and backend. ( NILKANTH PATEL &amp; ROHIT PRAJAPATI )</li> <li>• Full Day Leave. ( KRISHNA CHAPLA )</li> </ul>
<b>02/04/2024</b>	<ul style="list-style-type: none"> <li>• Worked on group ID generation such that when new year begins then count starts from the beginning. Eg: '2025G001'. ( KRISHNA CHAPLA )</li> <li>• Added searching and sorting in all tables and improved group section design. ( NILKANTH PATEL &amp; ROHIT PRAJAPATI )</li> </ul>
<b>03/04/2024</b>	<ul style="list-style-type: none"> <li>• Worked on weekly report and final report deadline management. ( KRISHNA CHAPLA &amp; NILKANTH PATEL )</li> <li>• Designed need help pages for all dashboards. ( ROHIT PRAJAPATI )</li> </ul>
<b>04/04/2024</b>	<ul style="list-style-type: none"> <li>• Worked on the intern details update section for approved interns. ( NILKANTH PATEL )</li> <li>• Designed a 404 error page and worked on a report. ( ROHIT PRAJAPATI )</li> <li>• Full Day Leave. ( KRISHNA CHAPLA )</li> </ul>

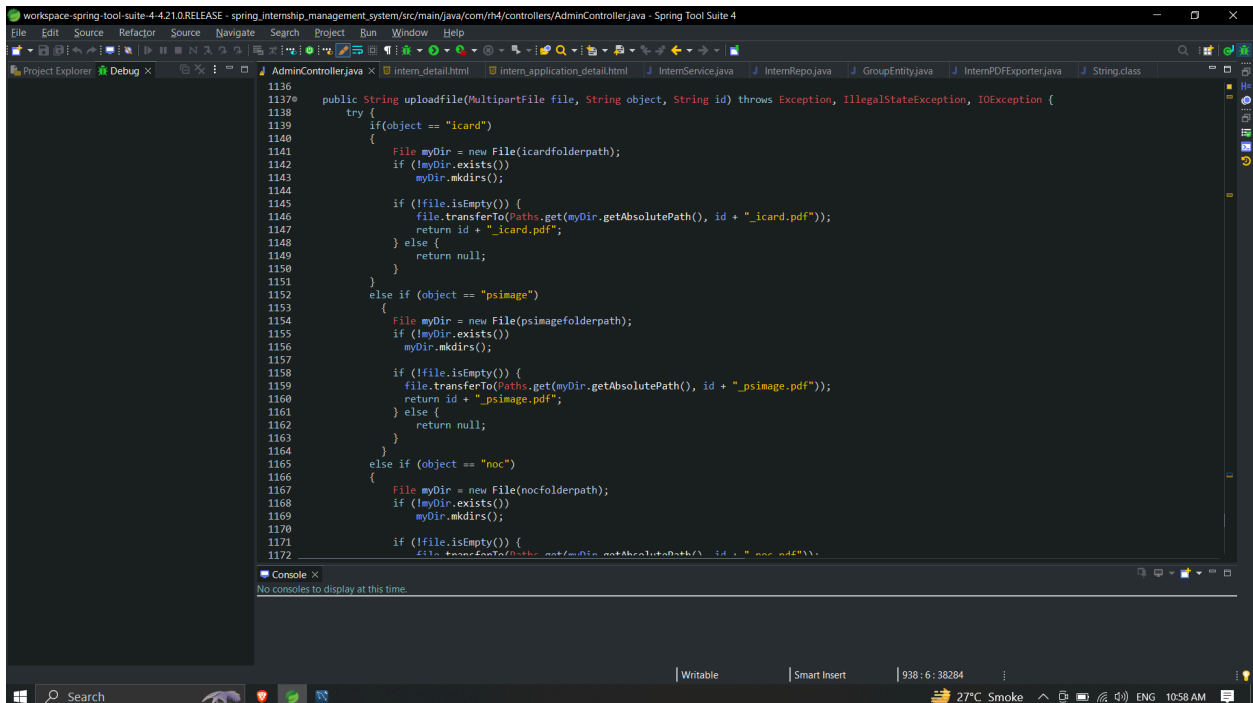
<b>Next Week Plan</b>	<ul style="list-style-type: none"> <li>• We will complete all kinds of the pending changes.</li> </ul>
-----------------------	--

## SCREEN SHOTS :



This screenshot shows the Spring Tool Suite 4 IDE with the `AdminController.java` file open. The code is in the `src/main/java/com/rh4/controllers` package. It features a `@GetMapping("/admin_pending_def_approvals")` method that returns a `ModelAndView` with a list of `GroupEntity` objects. Below this, there is a `@PostMapping("/admin_pending_def_approvals/{groupId}")` method that handles group updates, including setting project definitions and statuses. The code is well-commented and includes error handling for null groups.

```
931 // Project Definition Approvals
932 @GetMapping("/admin_pending_def_approvals")
933 public ModelAndView pendingFromGuide() {
934     ModelAndView mv = new ModelAndView("/admin/admin_pending_def_approvals");
935     List<GroupEntity> groups = groupService.getAPendingGroups();
936     mv.addObject("groups", groups);
937     return mv;
938 }
939
940 @PostMapping("/admin_pending_def_approvals/{groupId}")
941 public String pendingFromAdmin(@RequestParam("apendingAns") String apendingAns,
942     @PathVariable("groupId") String groupId, @RequestParam("projectDefinition") String projectDefinition, @RequestParam("description") String description) {
943     GroupEntity group = groupService.getGroup(groupId);
944     if (group != null) {
945         group.setProjectDefinition(projectDefinition);
946         group.setDescription(description);
947         if (apendingAns.equals("approve")) {
948             group.setProjectDefinitionStatus("approved");
949             List<Intern> interns = internService.getInternsByGroupId(group.getId());
950             for (Intern intern : interns) {
951                 intern.setProjectDefinitionName(group.getProjectDefinition());
952                 internRepo.save(intern);
953             }
954         } else {
955             group.setProjectDefinitionStatus("pending");
956         }
957         groupRepo.save(group);
958     } else {
959         // Handle the case where the group is null, perhaps by logging an error or returning an error response
960         return "redirect:/bisag/admin/admin_pending_def_approvals";
961     }
962 }
963
964 @GetMapping("/admin_weekly_report")
965 // ...
966 }
```



This screenshot shows the Spring Tool Suite 4 IDE with the `AdminController.java` file open, displaying a different section of the code. It features a `public String uploadfile(MultipartFile file, String object, String id) throws Exception, IllegalStateException, IOException` method. This method handles file uploads for different categories: "icard", "psimage", and "noc". It checks if the directory exists, creates it if necessary, and then transfers the file to the appropriate path. The code is well-commented and includes error handling for file operations.

```
1137 public String uploadfile(MultipartFile file, String object, String id) throws Exception, IllegalStateException, IOException {
1138     try {
1139         if (object == "icard") {
1140             File myDir = new File(icardfolderpath);
1141             if (!myDir.exists()) {
1142                 myDir.mkdirs();
1143             }
1144             if (!file.isEmpty()) {
1145                 file.transferTo(Paths.get(myDir.getAbsolutePath(), id + "_icard.pdf"));
1146                 return id + "_icard.pdf";
1147             } else {
1148                 return null;
1149             }
1150         } else if (object == "psimage") {
1151             File myDir = new File(psimagefolderpath);
1152             if (!myDir.exists()) {
1153                 myDir.mkdirs();
1154             }
1155             if (!file.isEmpty()) {
1156                 file.transferTo(Paths.get(myDir.getAbsolutePath(), id + "_psimage.pdf"));
1157                 return id + "_psimage.pdf";
1158             } else {
1159                 return null;
1160             }
1161         } else if (object == "noc") {
1162             File myDir = new File(nocfolderpath);
1163             if (!myDir.exists()) {
1164                 myDir.mkdirs();
1165             }
1166             if (!file.isEmpty()) {
1167                 file.transferTo(Paths.get(myDir.getAbsolutePath(), id + "_noc.pdf"));
1168                 return id + "_noc.pdf";
1169             } else {
1170                 return null;
1171             }
1172         }
1173     } catch (Exception e) {
1174         // ...
1175     }
1176 }
```

```
workspace-spring-tool-suite-4-4.21.0.RELEASE - spring_internship_management_system/src/main/java/com/rh4/controllers/AdminController.java - Spring Tool Suite 4
File Edit Source Refactor Source Navigate Search Project Run Window Help
Project Explorer Debug AdminController.java intern_detail.html intern_application_detail.html InternService.java InternRepo.java GroupEntity.java InternPDFExporter.java String.class
162 public String generateGroupId() {
163     // Check if the current year has changed
164     int year = getCurrentYear();
165     SimpleDateFormat yearFormat = new SimpleDateFormat("yyyy");
166     String currentYear = yearFormat.format(new Date());
167     int currentYearInt = Integer.parseInt(currentYear);
168     int serialNumber = generateSerialNumberForGroup();
169     if (year != currentYearInt) {
170         currentYearInt = year; // Update the current year
171         serialNumber = 0; // Reset the serial number to 0
172     }
173     // Increment the serial number
174     serialNumber++;
175     // Format the serial number
176     String formattedSerialNumber = String.format("%03d", serialNumber);
177     // Combine the parts to form the custom group ID
178     String groupId = currentYear + "G" + formattedSerialNumber;
179     return groupId;
180 }
181 // Method to get the current year
182 private int getCurrentYear() {
183     SimpleDateFormat yearFormat = new SimpleDateFormat("yyyy");
184     String yearString = yearFormat.format(new Date());
185     return Integer.parseInt(yearString);
186 }
187 // Method to generate the serial number for group
188 private int generateSerialNumberForGroup() {
189     String id = groupService.getHostRecentGroupId();
190     if (id == null || id.isEmpty()) {
191         return 0;
192     }
193     return Integer.parseInt(id);
194 }
195 }
196 }
197 }
198 }
199 }
200 }
201 }
202 }
203 }
204 }
205 }
206 }
207 }
208 }
209 }
210 }
211 }
212 }
213 }
214 }
215 }
216 }
217 }
218 }
219 }
220 }
221 }
222 }
223 }
224 }
225 }
226 }
227 }
228 }
229 }
230 }
231 }
232 }
233 }
234 }
235 }
236 }
237 }
238 }
239 }
240 }
241 }
242 }
243 }
244 }
245 }
246 }
247 }
248 }
249 }
250 }
251 }
252 }
253 }
254 }
255 }
256 }
257 }
258 }
259 }
260 }
261 }
262 }
263 }
264 }
265 }
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 }
400 }
401 }
402 }
403 }
404 }
405 }
406 }
407 }
408 }
409 }
410 }
411 }
412 }
413 }
414 }
415 }
416 }
417 }
418 }
419 }
420 }
421 }
422 }
423 }
424 }
425 }
426 }
427 }
428 }
429 }
430 }
431 }
432 }
433 }
434 }
435 }
436 }
437 }
438 }
439 }
440 }
441 }
442 }
443 }
444 }
445 }
446 }
447 }
448 }
449 }
450 }
451 }
452 }
453 }
454 }
455 }
456 }
457 }
458 }
459 }
460 }
461 }
462 }
463 }
464 }
465 }
466 }
467 }
468 }
469 }
470 }
471 }
472 }
473 }
474 }
475 }
476 }
477 }
478 }
479 }
480 }
481 }
482 }
483 }
484 }
485 }
486 }
487 }
488 }
489 }
490 }
491 }
492 }
493 }
494 }
495 }
496 }
497 }
498 }
499 }
500 }
501 }
502 }
503 }
504 }
505 }
506 }
507 }
508 }
509 }
510 }
511 }
512 }
513 }
514 }
515 }
516 }
517 }
518 }
519 }
520 }
521 }
522 }
523 }
524 }
525 }
526 }
527 }
528 }
529 }
530 }
531 }
532 }
533 }
534 }
535 }
536 }
537 }
538 }
539 }
540 }
541 }
542 }
543 }
544 }
545 }
546 }
547 }
548 }
549 }
550 }
551 }
552 }
553 }
554 }
555 }
556 }
557 }
558 }
559 }
560 }
561 }
562 }
563 }
564 }
565 }
566 }
567 }
568 }
569 }
570 }
571 }
572 }
573 }
574 }
575 }
576 }
577 }
578 }
579 }
580 }
581 }
582 }
583 }
584 }
585 }
586 }
587 }
588 }
589 }
590 }
591 }
592 }
593 }
594 }
595 }
596 }
597 }
598 }
599 }
600 }
601 }
602 }
603 }
604 }
605 }
606 }
607 }
608 }
609 }
610 }
611 }
612 }
613 }
614 }
615 }
616 }
617 }
618 }
619 }
620 }
621 }
622 }
623 }
624 }
625 }
626 }
627 }
628 }
629 }
630 }
631 }
632 }
633 }
634 }
635 }
636 }
637 }
638 }
639 }
640 }
641 }
642 }
643 }
644 }
645 }
646 }
647 }
648 }
649 }
650 }
651 }
652 }
653 }
654 }
655 }
656 }
657 }
658 }
659 }
660 }
661 }
662 }
663 }
664 }
665 }
666 }
667 }
668 }
669 }
670 }
671 }
672 }
673 }
674 }
675 }
676 }
677 }
678 }
679 }
680 }
681 }
682 }
683 }
684 }
685 }
686 }
687 }
688 }
689 }
690 }
691 }
692 }
693 }
694 }
695 }
696 }
697 }
698 }
699 }
700 }
701 }
702 }
703 }
704 }
705 }
706 }
707 }
708 }
709 }
710 }
711 }
712 }
713 }
714 }
715 }
716 }
717 }
718 }
719 }
720 }
721 }
722 }
723 }
724 }
725 }
726 }
727 }
728 }
729 }
730 }
731 }
732 }
733 }
734 }
735 }
736 }
737 }
738 }
739 }
740 }
741 }
742 }
743 }
744 }
745 }
746 }
747 }
748 }
749 }
750 }
751 }
752 }
753 }
754 }
755 }
756 }
757 }
758 }
759 }
760 }
761 }
762 }
763 }
764 }
765 }
766 }
767 }
768 }
769 }
770 }
771 }
772 }
773 }
774 }
775 }
776 }
777 }
778 }
779 }
780 }
781 }
782 }
783 }
784 }
785 }
786 }
787 }
788 }
789 }
790 }
791 }
792 }
793 }
794 }
795 }
796 }
797 }
798 }
799 }
800 }
801 }
802 }
803 }
804 }
805 }
806 }
807 }
808 }
809 }
810 }
811 }
812 }
813 }
814 }
815 }
816 }
817 }
818 }
819 }
820 }
821 }
822 }
823 }
824 }
825 }
826 }
827 }
828 }
829 }
830 }
831 }
832 }
833 }
834 }
835 }
836 }
837 }
838 }
839 }
840 }
841 }
842 }
843 }
844 }
845 }
846 }
847 }
848 }
849 }
850 }
851 }
852 }
853 }
854 }
855 }
856 }
857 }
858 }
859 }
860 }
861 }
862 }
863 }
864 }
865 }
866 }
867 }
868 }
869 }
870 }
871 }
872 }
873 }
874 }
875 }
876 }
877 }
878 }
879 }
880 }
881 }
882 }
883 }
884 }
885 }
886 }
887 }
888 }
889 }
890 }
891 }
892 }
893 }
894 }
895 }
896 }
897 }
898 }
899 }
900 }
901 }
902 }
903 }
904 }
905 }
906 }
907 }
908 }
909 }
910 }
911 }
912 }
913 }
914 }
915 }
916 }
917 }
918 }
919 }
920 }
921 }
922 }
923 }
924 }
925 }
926 }
927 }
928 }
929 }
930 }
931 }
932 }
933 }
934 }
935 }
936 }
937 }
938 }
939 }
940 }
941 }
942 }
943 }
944 }
945 }
946 }
947 }
948 }
949 }
950 }
951 }
952 }
953 }
954 }
955 }
956 }
957 }
958 }
959 }
960 }
961 }
962 }
963 }
964 }
965 }
966 }
967 }
968 }
969 }
970 }
971 }
972 }
973 }
974 }
975 }
976 }
977 }
978 }
979 }
980 }
981 }
982 }
983 }
984 }
985 }
986 }
987 }
988 }
989 }
990 }
991 }
992 }
993 }
994 }
995 }
996 }
997 }
998 }
999 }
1000 }
```

```
workspace-spring-tool-suite-4-4.21.0.RELEASE - spring_internship_management_system/src/main/java/com/rh4/controllers/AdminController.java - Spring Tool Suite 4
File Edit Source Refactor Source Navigate Search Project Run Window Help
Project Explorer Debug AdminController.java intern_detail.html intern_application_detail.html InternService.java InternRepo.java GroupEntity.java InternPDFExporter.java String.class
378 @PostMapping("/intern_application/update")
379 public String internApplicationSubmission(@RequestParam long id, InternApplication internApplication, MultipartHttpServletRequest req) throws IllegalStateException {
380     Optional<InternApplication> intern = internService.getInternApplication(id);
381     if (internApplication.getIsActive() != true)
382     {
383         intern.get().setFirstName(internApplication.getFirstName());
384         intern.get().setLastName(internApplication.getLastName());
385         intern.get().setContactNo(internApplication.getContactNo());
386         MyUser user = myUserService.getUserByUsername(intern.get().getEmail());
387         user.setUsername(internApplication.getEmail());
388         userRepo.save(user);
389         intern.get().setEmail(internApplication.getEmail());
390         intern.get().setCollegeName(internApplication.getCollegeName());
391         intern.get().setBranch(internApplication.getBranch());
392         intern.get().setProgrammingLangName(internApplication.getProgrammingLangName());
393         intern.get().setSemester(internApplication.getSemester());
394         intern.get().setJoiningDate(internApplication.getJoiningDate());
395         intern.get().setCompletionDate(internApplication.getCompletionDate());
396         if (req.getFile("icardImageone") != null)
397         {
398             intern.get().setCardImage(uploadfile(req.getFile("icardImageone"), "icard", Long.toString(intern.get().getId())));
399         }
400         if (req.getFile("nocPdfone") != null)
401         {
402             intern.get().setNocPdf(uploadfile(req.getFile("nocPdfone"), "noc", Long.toString(intern.get().getId())));
403         }
404         if (req.getFile("resumePdfone") != null)
405         {
406             intern.get().setResumePdf(uploadfile(req.getFile("resumePdfone"), "resume", Long.toString(intern.get().getId())));
407         }
408     }
409     return "success";
410 }
411 }
412 }
413 }
414 }
415 }
416 }
417 }
418 }
419 }
420 }
421 }
422 }
423 }
424 }
425 }
426 }
427 }
428 }
429 }
430 }
431 }
432 }
433 }
434 }
435 }
436 }
437 }
438 }
439 }
440 }
441 }
442 }
443 }
444 }
445 }
446 }
447 }
448 }
449 }
450 }
451 }
452 }
453 }
454 }
455 }
456 }
457 }
458 }
459 }
460 }
461 }
462 }
463 }
464 }
465 }
466 }
467 }
468 }
469 }
470 }
471 }
472 }
473 }
474 }
475 }
476 }
477 }
478 }
479 }
480 }
481 }
482 }
483 }
484 }
485 }
486 }
487 }
488 }
489 }
490 }
491 }
492 }
493 }
494 }
495 }
496 }
497 }
498 }
499 }
500 }
501 }
502 }
503 }
504 }
505 }
506 }
507 }
508 }
509 }
510 }
511 }
512 }
513 }
514 }
515 }
516 }
517 }
518 }
519 }
520 }
521 }
522 }
523 }
524 }
525 }
526 }
527 }
528 }
529 }
530 }
531 }
532 }
533 }
534 }
535 }
536 }
537 }
538 }
539 }
540 }
541 }
542 }
543 }
544 }
545 }
546 }
547 }
548 }
549 }
550 }
551 }
552 }
553 }
554 }
555 }
556 }
557 }
558 }
559 }
560 }
561 }
562 }
563 }
564 }
565 }
566 }
567 }
568 }
569 }
570 }
571 }
572 }
573 }
574 }
575 }
576 }
577 }
578 }
579 }
580 }
581 }
582 }
583 }
584 }
585 }
586 }
587 }
588 }
589 }
590 }
591 }
592 }
593 }
594 }
595 }
596 }
597 }
598 }
599 }
600 }
601 }
602 }
603 }
604 }
605 }
606 }
607 }
608 }
609 }
610 }
611 }
612 }
613 }
614 }
615 }
616 }
617 }
618 }
619 }
620 }
621 }
622 }
623 }
624 }
625 }
626 }
627 }
628 }
629 }
630 }
631 }
632 }
633 }
634 }
635 }
636 }
637 }
638 }
639 }
640 }
641 }
642 }
643 }
644 }
645 }
646 }
647 }
648 }
649 }
650 }
651 }
652 }
653 }
654 }
655 }
656 }
657 }
658 }
659 }
660 }
661 }
662 }
663 }
664 }
665 }
666 }
667 }
668 }
669 }
670 }
671 }
672 }
673 }
674 }
675 }
676 }
677 }
678 }
679 }
680 }
681 }
682 }
683 }
684 }
685 }
686 }
687 }
688 }
689 }
690 }
691 }
692 }
693 }
694 }
695 }
696 }
697 }
698 }
699 }
700 }
701 }
702 }
703 }
704 }
705 }
706 }
707 }
708 }
709 }
710 }
711 }
712 }
713 }
714 }
715 }
716 }
717 }
718 }
719 }
720 }
721 }
722 }
723 }
724 }
725 }
726 }
727 }
728 }
729 }
730 }
731 }
732 }
733 }
734 }
735 }
736 }
737 }
738 }
739 }
740 }
741 }
742 }
743 }
744 }
745 }
746 }
747 }
748 }
749 }
750 }
751 }
752 }
753 }
754 }
755 }
756 }
757 }
758 }
759 }
760 }
761 }
762 }
763 }
764 }
765 }
766 }
767 }
768 }
769 }
770 }
771 }
772 }
773 }
774 }
775 }
776 }
777 }
778 }
779 }
780 }
781 }
782 }
783 }
784 }
785 }
786 }
787 }
788 }
789 }
790 }
791 }
792 }
793 }
794 }
795 }
796 }
797 }
798 }
799 }
800 }
801 }
802 }
803 }
804 }
805 }
806 }
807 }
808 }
809 }
810 }
811 }
812 }
813 }
814 }
815 }
816 }
817 }
818 }
819 }
820 }
821 }
822 }
823 }
824 }
825 }
826 }
827 }
828 }
829 }
830 }
831 }
832 }
833 }
834 }
835 }
836 }
837 }
838 }
839 }
840 }
841 }
842 }
843 }
844 }
845 }
846 }
847 }
848 }
849 }
850 }
851 }
852 }
853 }
854 }
855 }
856 }
857 }
858 }
859 }
860 }
861 }
862 }
863 }
864 }
865 }
866 }
867 }
868 }
869 }
870 }
871 }
872 }
873 }
874 }
875 }
876 }
877 }
878 }
879 }
880 }
881 }
882 }
883 }
884 }
885 }
886 }
887 }
888 }
889 }
890 }
891 }
892 }
893 }
894 }
895 }
896 }
897 }
898 }
899 }
900 }
901 }
902 }
903 }
904 }
905 }
906 }
907 }
908 }
909 }
910 }
911 }
912 }
913 }
914 }
915 }
916 }
917 }
918 }
919 }
920 }
921 }
922 }
923 }
924 }
925 }
926 }
927 }
928 }
929 }
930 }
931 }
932 }
933 }
934 }
935 }
936 }
937 }
938 }
939 }
940 }
941 }
942 }
943 }
944 }
945 }
946 }
947 }
948 }
949 }
950 }
951 }
952 }
953 }
954 }
955 }
956 }
957 }
958 }
959 }
960 }
961 }
962 }
963 }
964 }
965 }
966 }
967 }
968 }
969 }
970 }
971 }
972 }
973 }
974 }
975 }
976 }
977 }
978 }
979 }
980 }
981 }
982 }
983 }
984 }
985 }
986 }
987 }
988 }
989 }
990 }
991 }
992 }
993 }
994 }
995 }
996 }
997 }
998 }
999 }
1000 }
```

