

# **International Institute of Information Technology, Bangalore**

Project Elective Report

## **Personal Library Management**

Under the Guidance of  
Prof. B. Thangaraju



Simran Basu (MT2020145)  
Naga Sri Vaishnavi Dhulipalla (IMT2017514)

## CONTENTS

1	Abstract	2
2	System Configuration and Tools used	2
3	Build - Pipenv	2
4	Testing	3
5	Source Code Management - GitHub	4
6	Containerization - Docker	5
7	Continuous Integration - Jenkins	7
8	Continuous Deployment - Ansible	9
9	Log Management	11
10	Build Jenkins Pipeline	13
11	Continuous Monitoring - ELK	16
12	Screenshots of the Running Application	18
13	Scope for future work	25
13.1	Online Book Clubs . . . . .	25
13.2	Permission Approval . . . . .	25
14	Conclusion	25

## 1 ABSTRACT

Personal library lets people collect and donate books online. It lets readers exchange books and expand their collection online. Every user can have and manage their own library with the online application.

The architecture of our project demands two layers:

- Front end - handled by HTML, CSS, JavaScript, Django-crispy forms and Bootstrap4.
- Back end - handled by Django and sqlite3 as database.

## 2 SYSTEM CONFIGURATION AND TOOLS USED

This project is developed using DevOps practices. DevOps is a set of practices that combines software development (Dev) and IT operations (Ops). It aims to shorten the systems development life cycle and provide continuous delivery with high software quality.

1. **Programming Language:** Python (Django)
2. **Testing:** Django testing
3. **Build:** Virtual Environment for Python(pipenv)
4. **Source Code Management:** GitHub for Git.  
[https://github.com/VaishnaviDhulipalla2902/DevOps\\_Project](https://github.com/VaishnaviDhulipalla2902/DevOps_Project)
5. **Containerization:** Docker.  
[https://hub.docker.com/repository/docker/vaishnavi2902/personal\\_library](https://hub.docker.com/repository/docker/vaishnavi2902/personal_library)
6. **Continuous Integration:** Jenkins
7. **Continuous Deployment:** Ansible
8. **Continuous Monitoring:** ELK Stack

## 3 BUILD – PIPENV

We created a virtual environment for the project so that the dependencies can be exported to a "requirements.txt" file. To do that we need to:

## 1. Install Pipenv

```
1 ~$ pip install pipenv
2
```

## 2. Start the virtual environment

Set the path to the working directory of the project and run the following command.

```
3 ~$ pipenv shell
4
```

## 3. Create a "requirements.txt" file.

Now, we create a requirements.txt file and push it to github.

```
5 ~$ pip freeze > requirements.txt
6
```

Now, we can see the file created inside the working directory. Anyone who wants to run the project on their machine, they just can start the virtual environment and install dependencies using :

```
7 ~$ pip install -r requirements.txt
8
```

```
☰ requirements.txt ×
☰ requirements.txt
1 asgiref==3.3.4
2 coverage==5.5
3 Django==3.2.1
4 django-crispy-forms==1.11.2
5 Pillow==8.2.0
6 pytz==2021.1
7 sqlparse==0.4.1
8 typing-extensions==3.10.0.0
```

Figure 1: Requirements.txt

## 4 TESTING

Jenkins provides us with continuous integration which includes integrated testing, so every time we push code to github it integrates all the different modules of the project and tests their proper functioning.

For testing within python code we use TestCase and SimpleTestCase provided by django.test module. We test if the urls.py, views.py and models.py are working properly. We test

each application - library and users, separately using the following commands:

```
9 ~$ python3 manage.py test library
```

```
10 ~$ python3 manage.py test users
```

To test everything at once, we used a package named coverage :

```
11 ~$ coverage run manage.py test
```

```
PS E:\DevOps_Project> python manage.py test library
Creating test database for alias 'default'...
System check identified no issues (0 silenced).

Ran 10 tests in 0.046s

OK
Destroying test database for alias 'default'...
PS E:\DevOps_Project> python manage.py test users
Creating test database for alias 'default'...
System check identified no issues (0 silenced).
..
Ran 2 tests in 0.052s

OK
Destroying test database for alias 'default'...
PS E:\DevOps_Project>
```

Figure 2: Testing

## 5 SOURCE CODE MANAGEMENT – GITHUB

Source code management (SCM) is used to track modifications to a source code repository. SCM tracks a running history of changes to a code base and helps resolve conflicts when merging updates from multiple contributors. SCM is also synonymous with Version control.

In this step, we create a repository on GitHub and then configure the django project on our local machine and link it to the Git repository.

```
12 ~$ git init
```

Open the command line on the root of the Maven project and type the above command to initialize the Maven project as a Git repository.

```
1 ~$ git remote add origin
```

This command helps to link the Git repository with local repository.

```
1 ~$ git add .
```

This command is to Stage the Django project in Git environment. ("." command specifies to stage all the files in the current folder that are modified or created.)

```
1 ~$ git commit -m "Commit Message"
```

This command is to commit the staged files to GitHub.

```
1 ~$ git push -u origin main
```

This command is to push the changes to the GitHub repository.

## 6 CONTAINERIZATION – DOCKER

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.

### 1. Install docker on the local machine

```
13 ~$ curl -fsSL https://get.docker.com -o get-docker.sh  
14 ~$ sh get-docker.sh
```

### 2. Create a repository in DockerHub

[https://hub.docker.com/repository/docker/vaishnavi2902/personal\\_library](https://hub.docker.com/repository/docker/vaishnavi2902/personal_library)

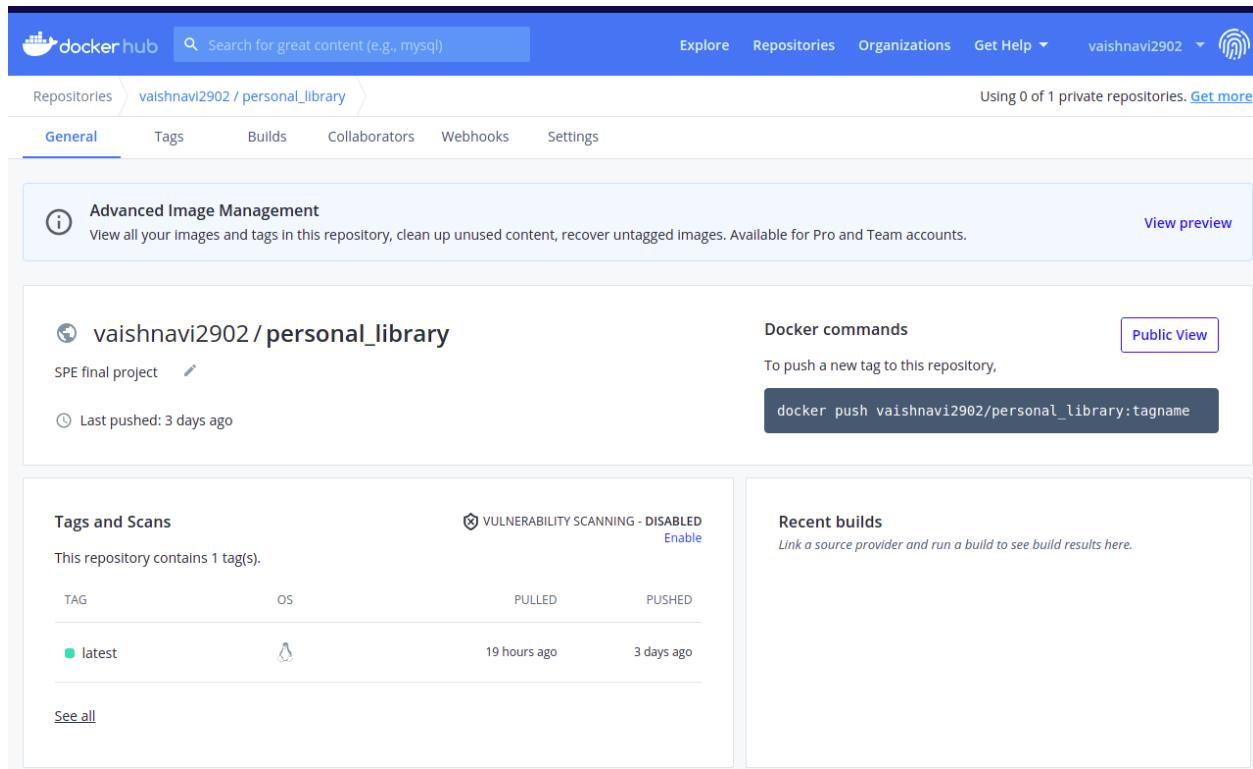


Figure 3: Docker Repository

### 3. Create a Dockerfile

After creating a Dockerfile, push the changes to GitHub.

```

Dockerfile X
Dockerfile > ...
1  FROM python:3
2
3  ENV PYTHONUNBUFFERED 1
4  ENV PYTHONDONTWRITEBYTECODE 1
5
6  RUN mkdir /app
7  WORKDIR /app
8  ADD . /app
9  COPY ./requirements.txt /app/requirements.txt
10 RUN pip3 install -r requirements.txt
11
12 COPY . /app
13
14 EXPOSE 8000
15
16 CMD [ "python", "manage.py", "runserver", "0.0.0.0:9010" ]

```

Figure 4: Docker File

## 7 CONTINUOUS INTEGRATION – JENKINS

### 1. Add Jenkins to the Docker group

```

1  sudo apt install openssh-server
2  sudo su - jenkins
3

```

By executing the above commands, we get logged into jenkins. Here we configure Jenkins to use Docker image via ssh.

```

1  mkdir .ssh
2  cd .ssh
3  ssh-keygen -t rsa
4  ssh-copy-id vaishnavi@localhost
5  ssh vaishnavi@localhost

```

After executing the last command, we get automatically directed outside the Jenkins user.

We can now start Jenkins with the command

```

1  sudo systemctl start jenkins

```

By default, Jenkins starts at port number 8080 so we login on to <http://localhost:8080> on to the browser.

## 2. Manage Plugins

We need to install all the required plugins like Build pipeline, Docker, GitHub, Maven Integration, Ansible etc. After they are all done installing, we need to restart Jenkins and add Docker credentials. In the Jenkins dashboard we add credentials to the Docker Hub repository and we set an unique id which is equal to docker with Registry credentials id in pipeline script.

## 3. Create a New Pipeline on Jenkins

Create new item, and select pipeline.

## 4. Setup a Jenkins Pipeline

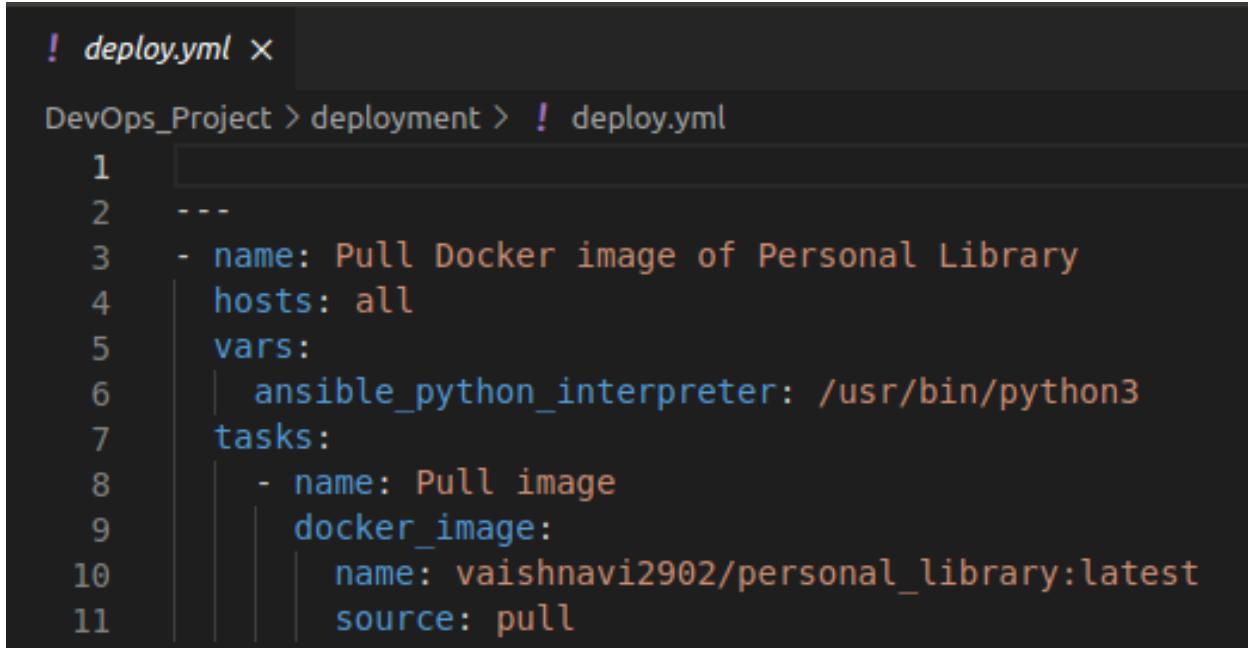
Write a Pipeline Script which includes steps like Git Clone, Build, Building a Docker image, Pushing the docker image to docker hub and Ansible Deployment. After we are done setting up every aspect of our DevOps tool chain, we may now build the jenkins job. This job for now can be manually triggered but for other SCM like GitLab it can be triggered based on events such as new push or new pull but for GitHub we might have set up a webhook. This can be done via port forwarding; we have to make jenkins port 8080 open on wide internet so it can be triggered based on event. But for now we perform this job manually.

```

1 Jenkinsfile ×
2 DevOps_Project > Jenkinsfile
3
4 pipeline{
5     environment{
6         docker_image = ""
7     }
8     agent any
9     stages{
10         stage('Step 1: Git Clone'){
11             steps{
12                 git branch: 'main', url: 'https://github.com/VaishnaviDhulipalla2902/DevOps_Project.git'
13             }
14         }
15         stage('Step 2: Build'){
16             steps{
17                 sh 'pip3 install --upgrade -r requirements.txt'
18             }
19         }
20         stage('Step 3: Testing'){
21             steps{
22                 sh 'python3 ./manage.py test library'
23                 sh 'python3 ./manage.py test users'
24             }
25         }
26         stage('Step 4: Build docker image') {
27             steps {
28                 script {
29                     docker_image = docker.build "vaishnavi2902/personal_library:latest"
30                 }
31             }
32         }
33         stage('Step 5: Push docker image to hub') {
34             steps {
35                 script {
36                     docker.withRegistry('', 'docker') {
37                         docker_image.push()
38                     }
39                 }
40             }
41         }
42         stage('Step 6: Ansible Deployment'){
43             steps{
44                 ansiblePlaybook becomeUser: null,
45                 colorized: true,
46                 credentialsId: 'docker',
47                 installation: 'Ansible',
48                 disableHostKeyChecking: true,
49                 inventory: 'deployment/inventory',
50                 playbook: 'deployment/deploy.yml',
51                 sudoUser: null
52             }
53         }
54     }
55 }
56 }
57 }
58 }
59 }
60 }
61 }
62 }
63 }
64 }
65 }
66 }
67 }
68 }
69 }
70 }
71 }
72 }
73 }
74 }
75 }
76 }
77 }
78 }
79 }
80 }
81 }
82 }
83 }
84 }
85 }
86 }
87 }
88 }
89 }
90 }
91 }
92 }
93 }
94 }
95 }
96 }
97 }
98 }
99 }
100 }
101 }
102 }
103 }
104 }
105 }
106 }
107 }
108 }
109 }
110 }
111 }
112 }
113 }
114 }
115 }
116 }
117 }
118 }
119 }
120 }
121 }
122 }
123 }
124 }
125 }
126 }
127 }
128 }
129 }
130 }
131 }
132 }
133 }
134 }
135 }
136 }
137 }
138 }
139 }
140 }
141 }
142 }
143 }
144 }
145 }
146 }
147 }
148 }
149 }
150 }
151 }
152 }
153 }
154 }
155 }
156 }
157 }
158 }
159 }
160 }
161 }
162 }
163 }
164 }
165 }
166 }
167 }
168 }
169 }
170 }
171 }
172 }
173 }
174 }
175 }
176 }
177 }
178 }
179 }
180 }
181 }
182 }
183 }
184 }
185 }
186 }
187 }
188 }
189 }
190 }
191 }
192 }
193 }
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 }
259 }
260 }
261 }
262 }
263 }
264 }
265 }
266 }
267 }
268 }
269 }
269 }
270 }
271 }
272 }
273 }
274 }
275 }
276 }
277 }
278 }
279 }
279 }
280 }
281 }
282 }
283 }
284 }
285 }
286 }
287 }
288 }
289 }
289 }
290 }
291 }
292 }
293 }
294 }
295 }
296 }
297 }
298 }
299 }
299 }
300 }
301 }
302 }
303 }
304 }
305 }
306 }
307 }
308 }
309 }
309 }
310 }
311 }
312 }
313 }
314 }
315 }
316 }
317 }
318 }
319 }
319 }
320 }
321 }
322 }
323 }
324 }
325 }
326 }
327 }
328 }
329 }
329 }
330 }
331 }
332 }
333 }
334 }
335 }
336 }
337 }
338 }
339 }
339 }
340 }
341 }
342 }
343 }
344 }
345 }
346 }
347 }
348 }
349 }
349 }
350 }
351 }
352 }
353 }
354 }
355 }
356 }
357 }
358 }
359 }
359 }
360 }
361 }
362 }
363 }
364 }
365 }
366 }
367 }
368 }
369 }
369 }
370 }
371 }
372 }
373 }
374 }
375 }
376 }
377 }
378 }
379 }
379 }
380 }
381 }
382 }
383 }
384 }
385 }
386 }
387 }
388 }
389 }
389 }
390 }
391 }
392 }
393 }
394 }
395 }
396 }
397 }
398 }
399 }
399 }
400 }
401 }
402 }
403 }
404 }
405 }
406 }
407 }
408 }
409 }
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 }
779 }
779 }
780 }
781 }
782 }
783 }
784 }
785 }
786 }
787 }
788 }
789 }
789 }
790 }
791 }
792 }
793 }
794 }
795 }
796 }
797 }
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 }
878 }
878 }
879 }
880 }
881 }
882 }
883 }
884 }
885 }
886 }
887 }
888 }
889 }
889 }
890 }
891 }
892 }
893 }
894 }
895 }
896 }
897 }
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 }
978 }
978 }
979 }
980 }
981 }
982 }
983 }
984 }
985 }
986 }
987 }
988 }
989 }
989 }
990 }
991 }
992 }
993 }
994 }
995 }
996 }
997 }
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 }
1078 }
1078 }
1079 }
1080 }
1081 }
1082 }
1083 }
1084 }
1085 }
1086 }
1087 }
1087 }
1088 }
1089 }
1090 }
1091 }
1092 }
1093 }
1094 }
1095 }
1096 }
1097 }
1097 }
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 }
1178 }
1178 }
1179 }
1180 }
1181 }
1182 }
1183 }
1184 }
1185 }
1186 }
1187 }
1188 }
1188 }
1189 }
1190 }
1191 }
1192 }
1193 }
1194 }
1195 }
1196 }
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 }
1278 }
1278 }
1279 }
1280 }
1281 }
1282 }
1283 }
1284 }
1285 }
1286 }
1287 }
1288 }
1288 }
1289 }
1290 }
1291 }
1292 }
1293 }
1294 }
1295 }
1296 }
1297 }
1297 }
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 }
1378 }
1378 }
1379 }
1380 }
1381 }
1382 }
1383 }
1384 }
1385 }
1386 }
1387 }
1388 }
1388 }
1389 }
1390 }
1391 }
1392 }
1393 }
1394 }
1395 }
1396 }
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 }
1476 }
1477 }
1478 }
1478 }
1479 }
1480 }
1481 }
1482 }
1483 }
1484 }
1485 }
1486 }
1487 }
1488 }
1488 }
1489 }
1490 }
1491 }
1492 }
1493 }
1494 }
1495 }
1496 }
1497 }
1497 }
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 }
1578 }
1578 }
1579 }
1580 }
1581 }
1582 }
1583 }
1584 }
1585 }
1586 }
1587 }
1588 }
1588 }
1589 }
1590 }
1591 }
1592 }
1593 }
1594 }
1595 }
1596 }
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 }
1678 }
1678 }
1679 }
1680 }
1681 }
1682 }
1683 }
1684 }
1685 }
1686 }
1687 }
1688 }
1688 }
1689 }
1690 }
1691 }
1692 }
1693 }
1694 }
1695 }
1696 }
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 }
1778 }
1778 }
1779 }
1780 }
1781 }
1782 }
1783 }
1784 }
1785 }
1786 }
1787 }
1788 }
1788 }
1789 }
1790 }
1791 }
1792 }
1793 }
1794 }
1795 }
1796 }
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 }
1878 }
1878 }
1879 }
1880 }
1881 }
1882 }
1883 }
1884 }
1885 }
1886 }
1887 }
1888 }
1888 }
1889 }
1889 }
1890 }
1891 }
1892 }
1893 }
1894 }
1895 }
1896 }
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 }
1978 }
1978 }
1979 }
1980 }
1981 }
1982 }
1983 }
1984 }
1985 }
1986 }
1987 }
1988 }
1988 }
1989 }
1989 }
1990 }
1991 }
1992 }
1993 }
1994 }
1995 }
1996 }
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 }
2078 }
2078 }
2079 }
2080 }
2081 }
2082 }
2083 }
2084 }
2085 }
2086 }
2087 }
2088 }
2088 }
2089 }
2089 }
2090 }
2091 }
2092 }
2093 }
2094 }
2095 }
2096 }
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 }
2152 }
2153 }
2154 }
2155 }
2156 }
2157 }
2158 }
2159 }
2159 }
2160 }
2161 }
2162 }
2163 }
2164 }
2165 }
2166 }
2167 }
2168 }
2169 }
2169 }
2170 }
2171 }
2172 }
2173 }
2174 }
2175 }
2176 }
2177 }
2178 }
2178 }
2179 }
2180 }
2181 }
2182 }
2183 }
2184 }
2185 }
2186 }
2187 }
2188 }
2188 }
2189 }
2189 }
2190 }
2191 }
2192 }
2193 }
2194 }
2195 }
2196 }
2197 }
2198 }
2198 }
2199 }
2199 }
2200 }
2201 }
2202 }
2203 }
2204 }
2205 }
2206 }
2207 }
2208 }
2209 }
2209 }
2210 }
2211 }
2212 }
2213 }
2214 }
2215 }
2216 }
2217 }
2218 }
2219 }
2219 }
2220 }
2221 }
222
```

- Create an Ansible playbook** We make a new directory in our working directory to create the Ansible playbook. We need to be careful about giving the correct path of docker image and the correct python version for which we can do

```
15 ~$ which python
```

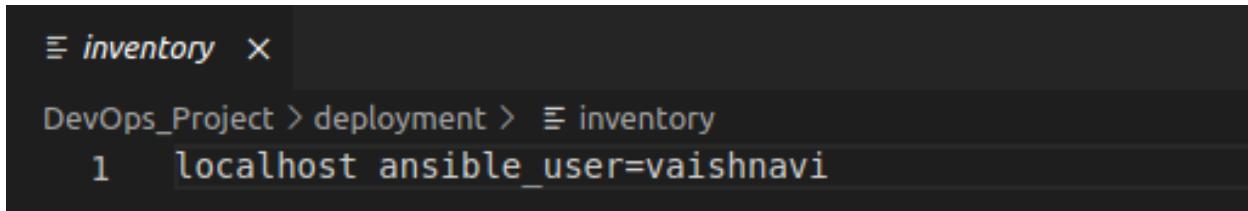


```
! deploy.yml ×

DevOps_Project > deployment > ! deploy.yml

1
2   ---
3   - name: Pull Docker image of Personal Library
4     hosts: all
5     vars:
6       ansible_python_interpreter: /usr/bin/python3
7     tasks:
8       - name: Pull image
9         docker_image:
10           name: vaishnavi2902/personal_library:latest
11           source: pull
```

Figure 6: Ansible Playbook



```
≡ inventory ×

DevOps_Project > deployment > ≡ inventory

1   localhost ansible_user=vaishnavi
```

Figure 7: Inventory File

- Configure Ansible in Jenkins** We need to go to Jenkins -> Dashboard -> Manage Jenkins -> Global Tool Configuration and add Ansible there. Here, we also need to give the correct path to Ansible.

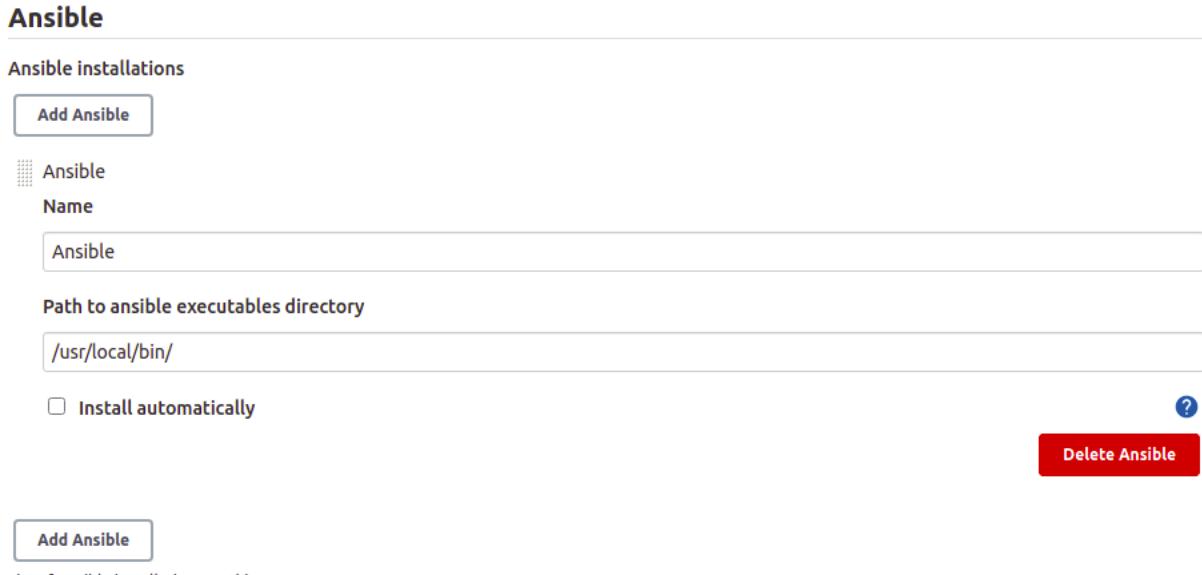


Figure 8: Ansible Configuration in Jenkins

### 3. Add Ansible Stage in the Jenkins pipeline script

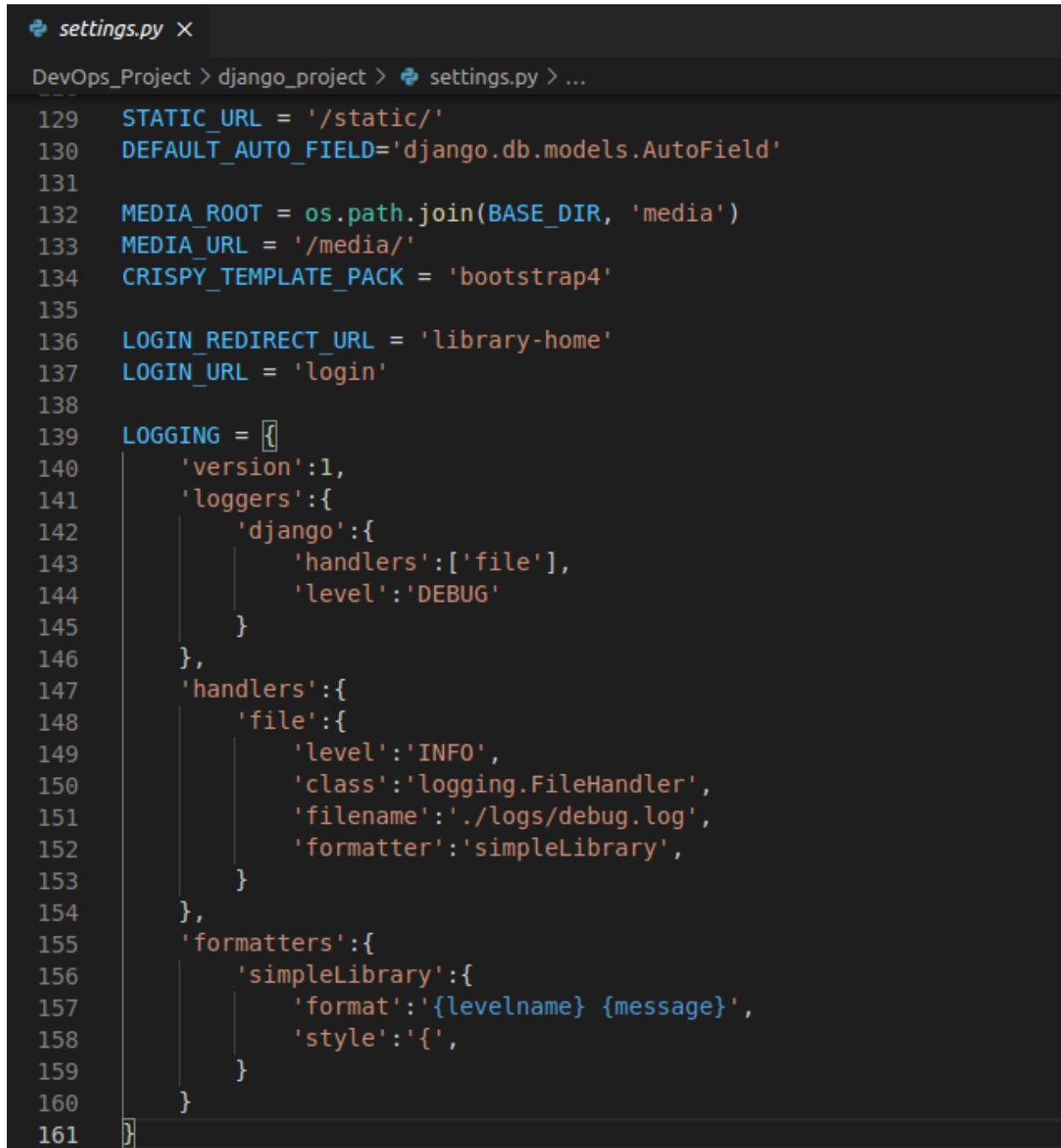
```

16 stage('Step 5: Ansible Deployment') {
17     steps{
18         ansiblePlaybook becomeUser: null ,
19             colorized: true ,
20             credentialsId: 'docker' ,
21             installation: 'Ansible'
22             disableHostKeyChecking: true ,
23             inventory: 'deployment/inventory' ,
24             playbook: 'deployment/deploy.yml' ,
25             sudoUser: null
26     }
27 }
```

## 9 LOG MANAGEMENT

Logging keeps track of all the operations that were performed in the application, warnings, debugging information, errors etc. The log folder contains a file debug.log which stores all the logging information.

1. **Manage logging in settings.py** LOGGING in settings.py handles the logs and stores them in debug.log.



```

    settings.py ×

DevOps_Project > django_project > settings.py > ...

129     STATIC_URL = '/static/'
130     DEFAULT_AUTO_FIELD='django.db.models.AutoField'
131
132     MEDIA_ROOT = os.path.join(BASE_DIR, 'media')
133     MEDIA_URL = '/media/'
134     CRISPY_TEMPLATE_PACK = 'bootstrap4'
135
136     LOGIN_REDIRECT_URL = 'library-home'
137     LOGIN_URL = 'login'
138
139     LOGGING = [
140         'version':1,
141         'loggers':{
142             'django':{
143                 'handlers':['file'],
144                 'level':'DEBUG'
145             }
146         },
147         'handlers':{
148             'file':{
149                 'level':'INFO',
150                 'class':'logging.FileHandler',
151                 'filename': './logs/debug.log',
152                 'formatter':'simpleLibrary',
153             }
154         },
155         'formatters':{
156             'simpleLibrary':{
157                 'format':'{levelname} {message}',
158                 'style':'{',
159             }
160         }
161     ]

```

Figure 9: LOGGING

## 2. Add Logger functions

We need to add logger functions in the main application code. Add the below line inside view.py.

```

1     import logging
2     logger = logging.getLogger(__name__)

```

3

And add some lines inside the functions to generate logs.

```
4     logger.info("Your Message");
5
```

```
@login_required
def profile(request):
    if request.method == 'POST':
        u_form = UserUpdateForm(request.POST, instance=request.user)
        p_form = ProfileUpdateForm(request.POST,
                                    request.FILES,
                                    instance=request.user.profile)
        if u_form.is_valid() and p_form.is_valid():
            u_form.save()
            p_form.save()
            messages.success(request, f'Your account has been updated')
            # DEBUG
            logger.debug('debug')
            # INFO
            logger.info('Update Successful')
            # WARNING
            logger.warning('warning')
            # ERROR
            logger.error(['Error Occured'])
    return redirect('profile')
```

Figure 10: LOGGING

## 10 BUILD JENKINS PIPELINE

Now, we need to build the project on Jenkins. While we build it, we might encounter a lot of errors which we fix along the way. The full stage pipeline looks like,

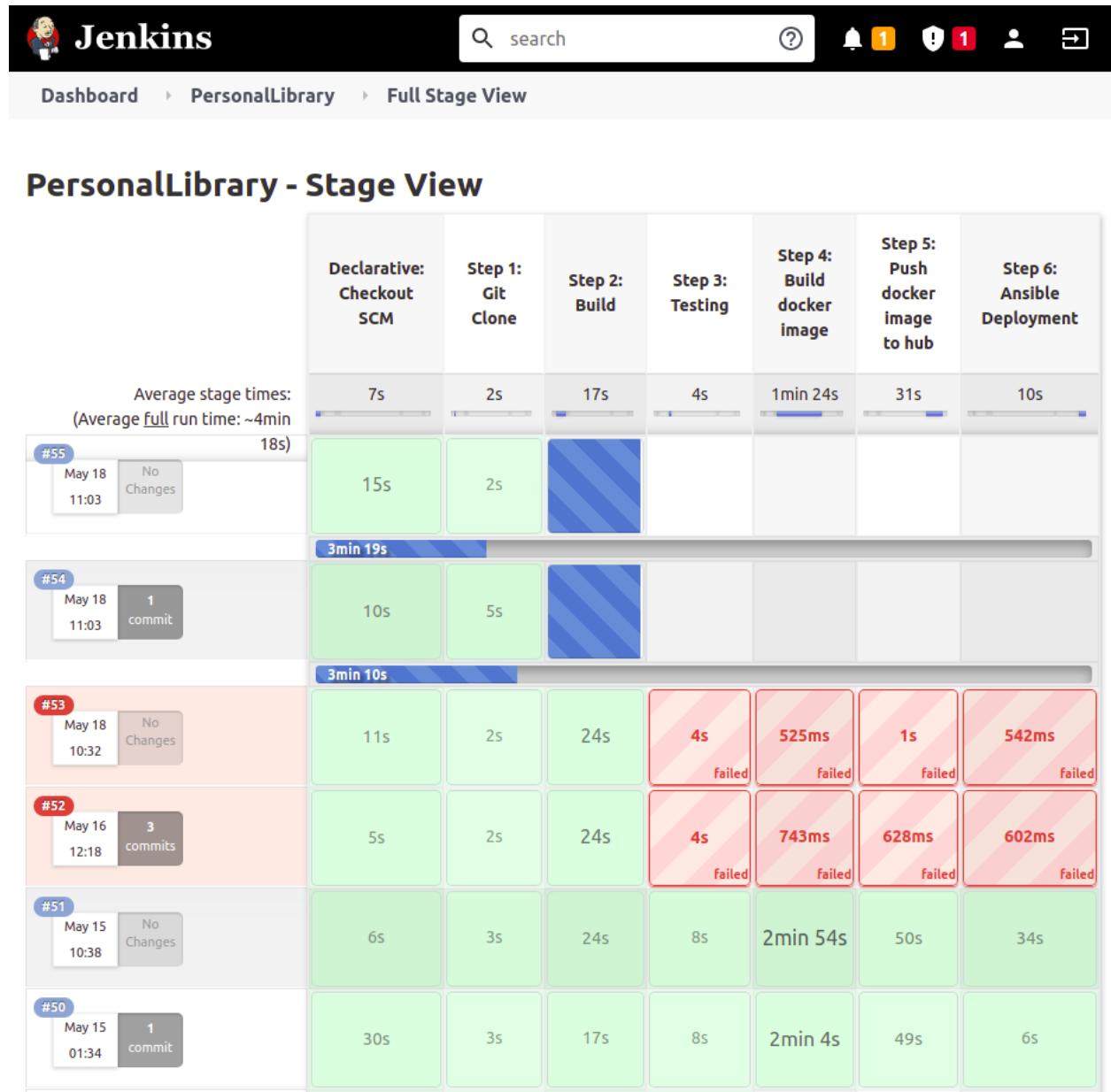


Figure 11: Full Stage Pipeline Jenkins

We may encounter some errors but we can fix them by looking up the error logs in jenkins. A successful build would look like:

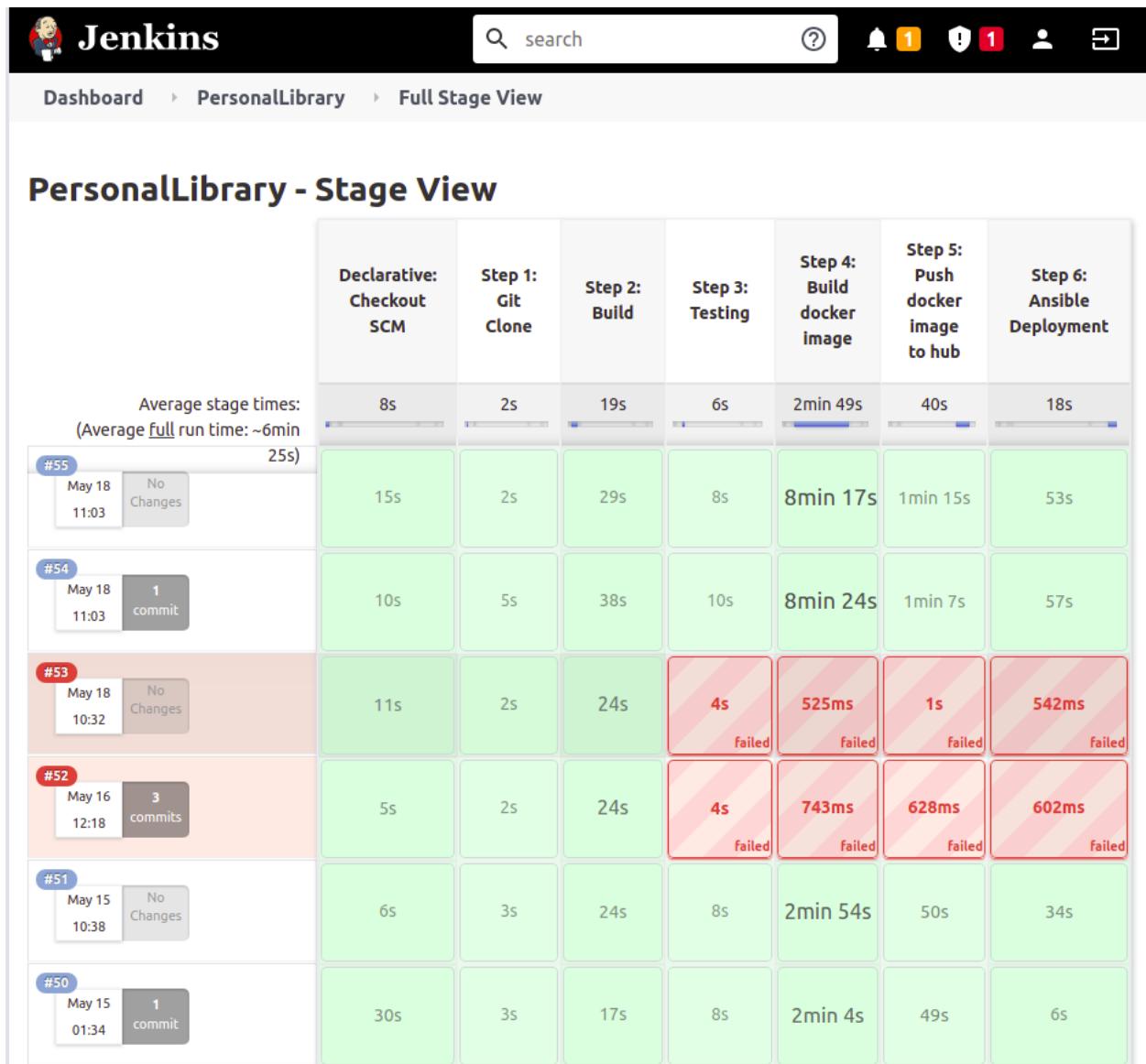


Figure 12: Full Stage Pipeline Jenkins

There are 5 stages in my Jenkins pipeline:

1. Git Clone
2. Build
3. Testing
4. Build Docker Image
5. Push Docker Image to Hub
6. Ansible Deployment

After doing all this, the docker image is pulled on to the local machine. We can run this image on the local machine to generate logs which act as input to the ELK monitoring.

```
28 ~$ docker images
29 ~$ docker run --name personal_library -d -p 9010:9010 vaishnavi2902/
   personal_library:latest
```

Starting and stopping the container is done as follows:

```
30 ~$ docker ps -a
31 ~$ docker start <container_id>
32 ~$ docker stop <container_id>
```

Now we know that the application executes smooth and we run the application for a few times to generate the log files.

## 11 CONTINUOUS MONITORING – ELK

"ELK" is the acronym for three open source projects: Elasticsearch, Logstash, and Kibana. ELK stack gives us the ability to aggregate logs from all the systems and applications, analyze these logs, and create visualizations for application and infrastructure monitoring, faster troubleshooting, security analytics, and more.

First, we run ElasticSearch(from the directory it is installed in) using,

```
33 ~$ /bin/elasticsearch
```

Elastic Search runs on <http://localhost:9200>.

Next run Kibana(from the directory it is installed in) using,

```
34 ~$ /bin/kibana
```

Kibana runs on <http://localhost:5601>.

Next run Logstash(from the directory it is installed in) using,

```
35 ~$ /bin/logstash -f <path_conf_file>
```

<path\_conf\_file> is the logstash configuration file: logstash.conf. It contains various plugins for input, output and filter. Here, we have used 2 plugins: grok and date. Grok helps to define a search and extract parts of our log line into structured fields and works by combining text patterns into something that matches the logs. The date filter is used for parsing dates from fields, and then using that date or timestamp as the logstash timestamp for the event. Logstash runs on <https://localhost:9600>.

```
logstash.conf ×  
DevOps_Project > logstash.conf  
1   input {  
2     stdin {  
3       beats {  
4         port => 5044  
5         ssl => true  
6       }  
7     }  
8   }  
9   filter {  
10     plugin {  
11       grok {  
12         match => { "message" => "%{TIMESTAMP_ISO8601:timestamp}" }  
13       }  
14       date {  
15         locale => "en"  
16         match => [ "timestamp", "YYYY-MM-dd HH:mm:ss" ]  
17         target => "@timestamp"  
18       }  
19     }  
20   }  
21 }  
22 output [  
23   elasticsearch {  
24     hosts => ["localhost:9200"]  
25     index => "indexforlogstash"  
26   }  
27 ]
```

Figure 13: logstash.conf

The screenshot shows a Microsoft Edge browser window displaying the Kibana interface. The URL in the address bar is `localhost:5601/app/management/kibana/indexPatterns/patterns/9d551c70-b51a-11eb-ae00-7b8d4c0f371f#/?_a=(tab: indexedFields)`. The main content area is titled "logstash" and displays a table of fields. The table has columns for Name, Type, Format, Searchable, Aggregatable, and Excluded. The fields listed are:

Name	Type	Format	Searchable	Aggregatable	Excluded
@timestamp	date	Date	●	●	<input type="button" value="edit"/>
@version	string	String	●	●	<input type="button" value="edit"/>
_id	string		●	●	<input type="button" value="edit"/>
_index	string		●	●	<input type="button" value="edit"/>

The left sidebar contains navigation links for Ingest, Data, Alerts and Insights, and Kibana. The bottom status bar shows system information like temperature (38°C), battery level (17%), and date/time (17-05-2021).

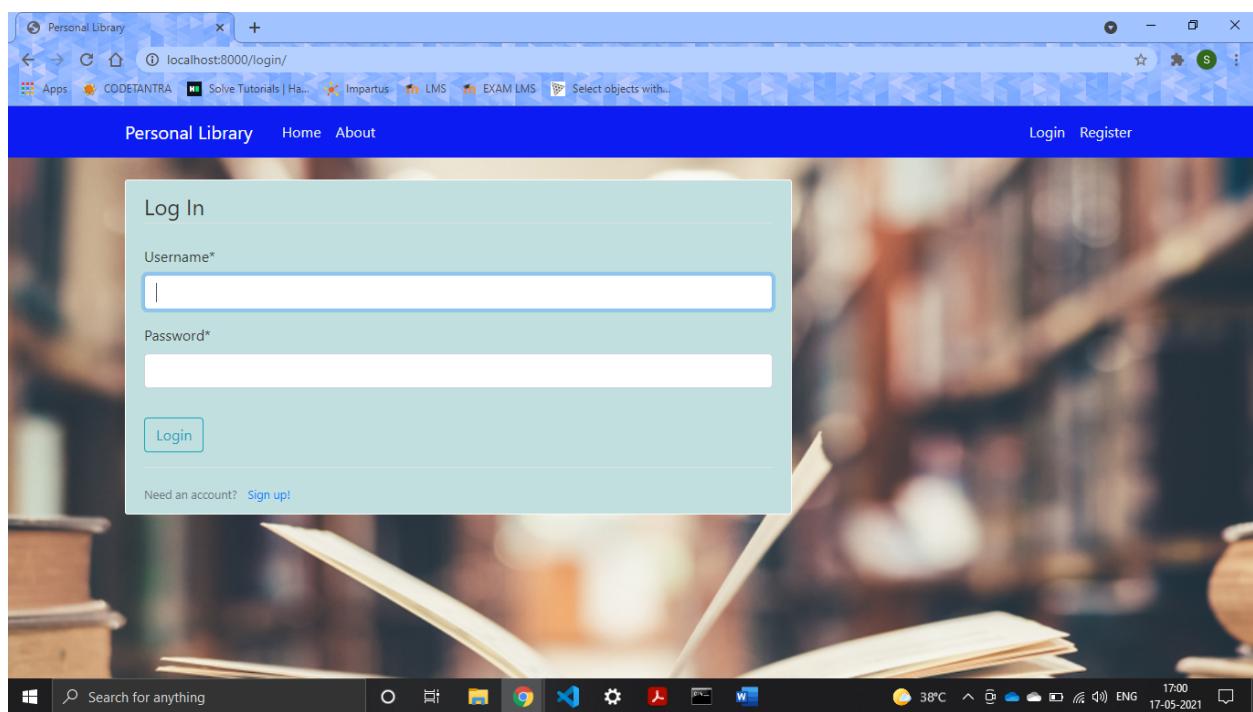
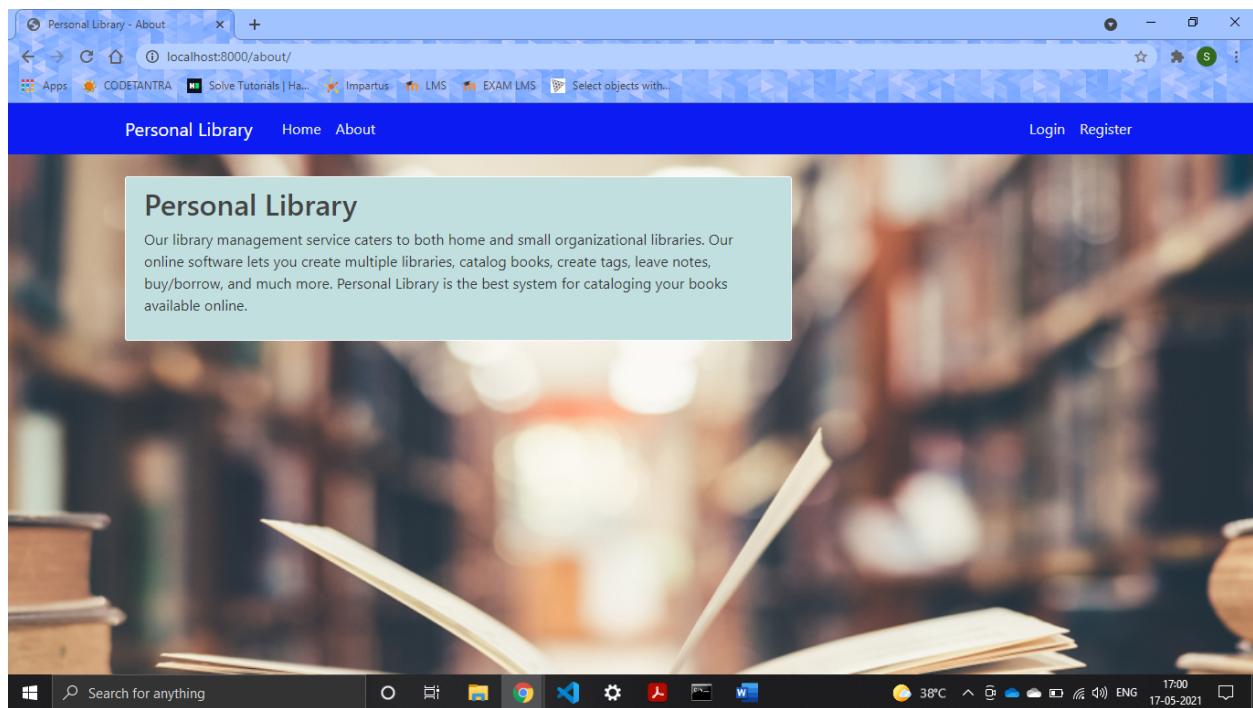
Figure 14: kibana dashboard

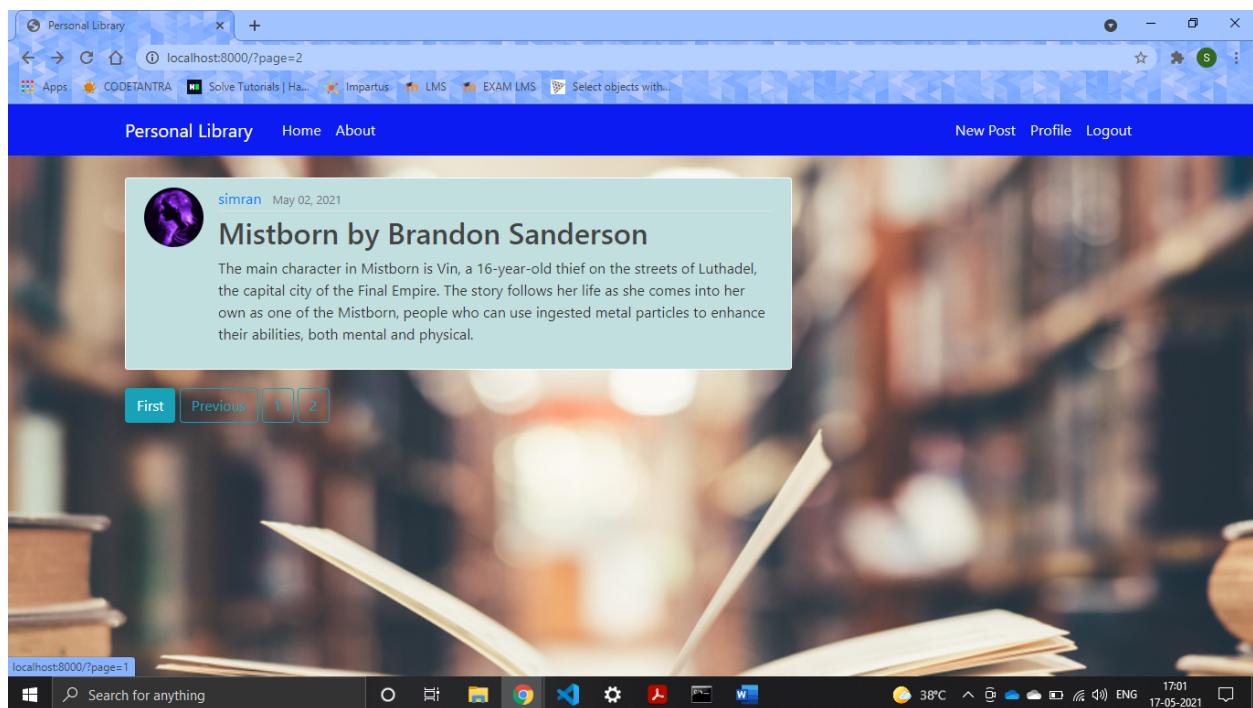
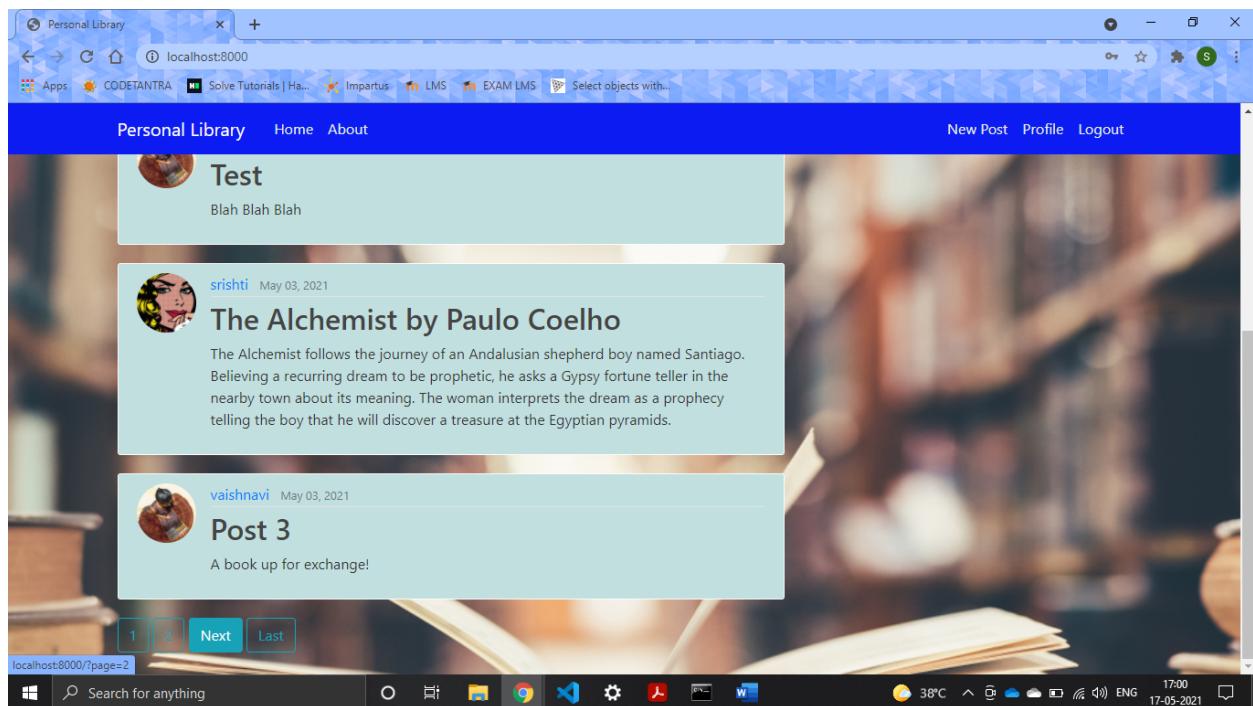
## 12 SCREENSHOTS OF THE RUNNING APPLICATION

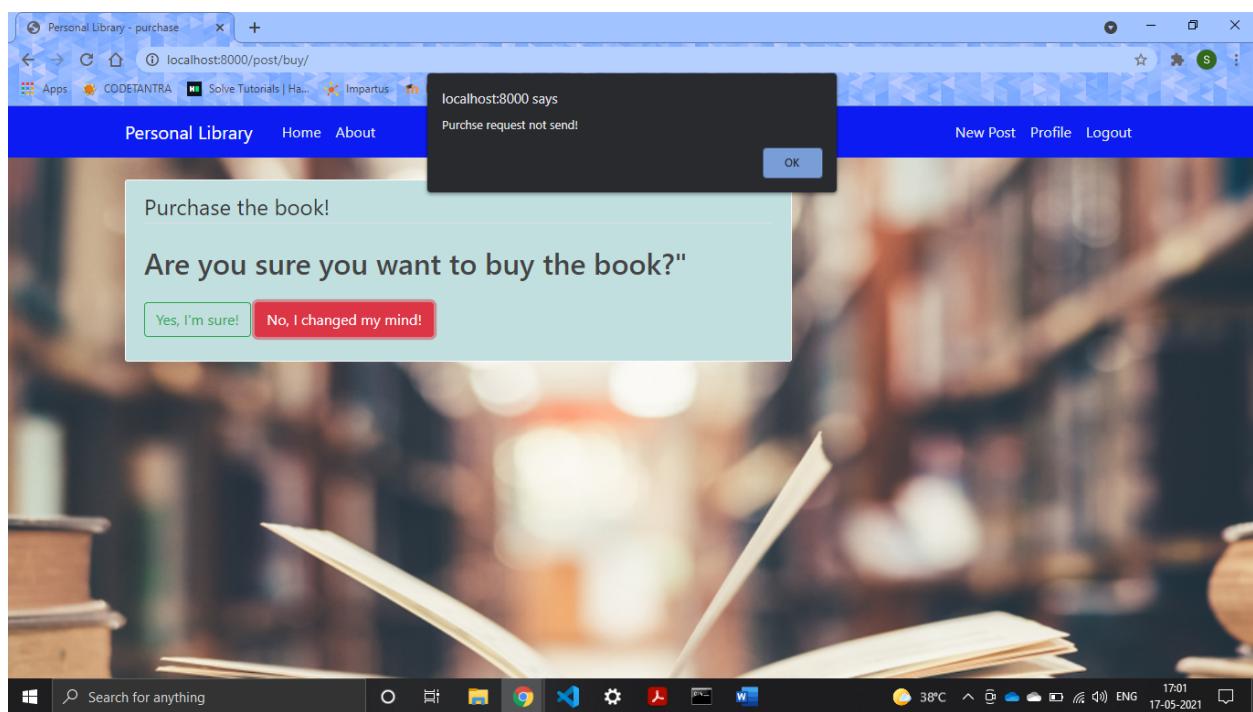
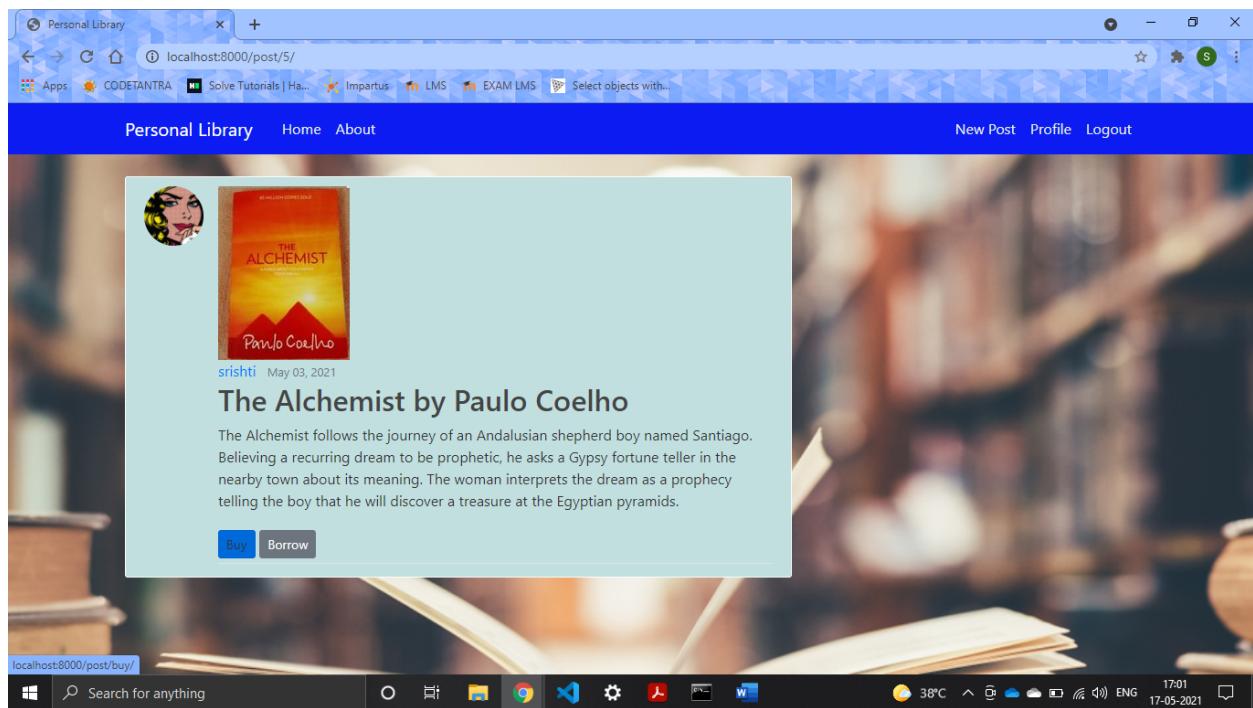
The screenshot shows a Microsoft Edge browser window displaying a running application. The URL in the address bar is `localhost:8000`. The main content area is titled "Personal Library" and shows a list of posts. The posts are:

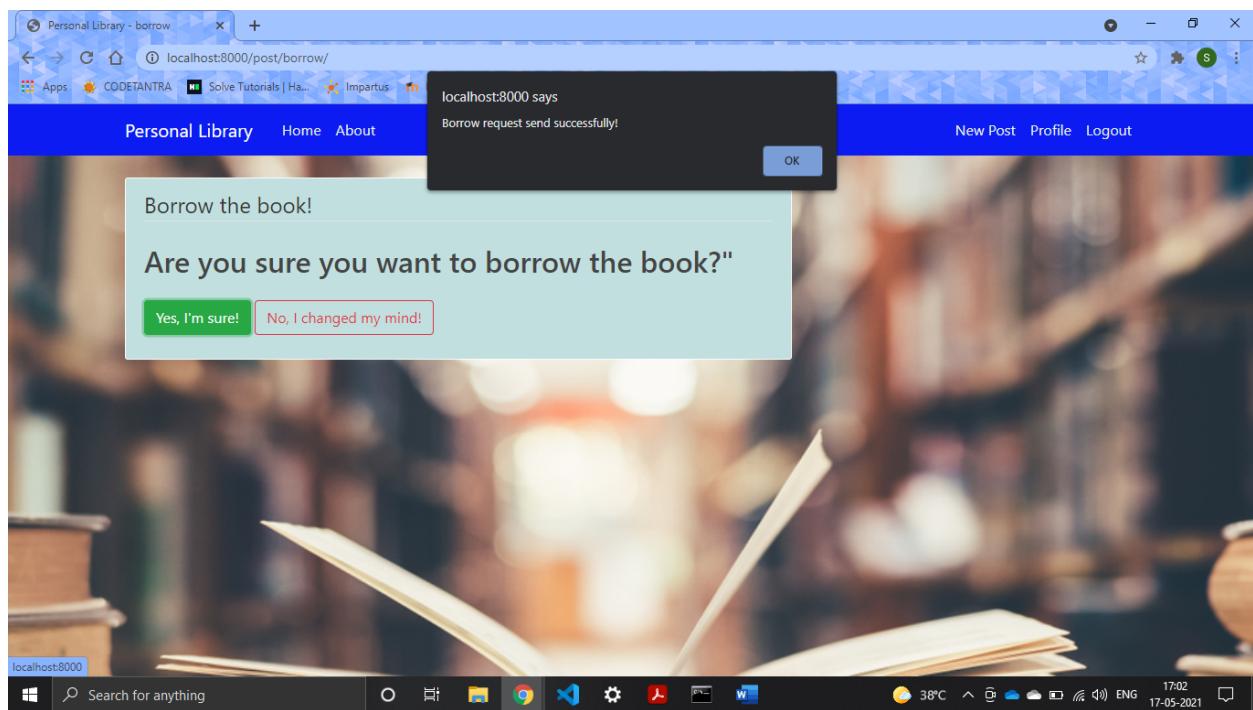
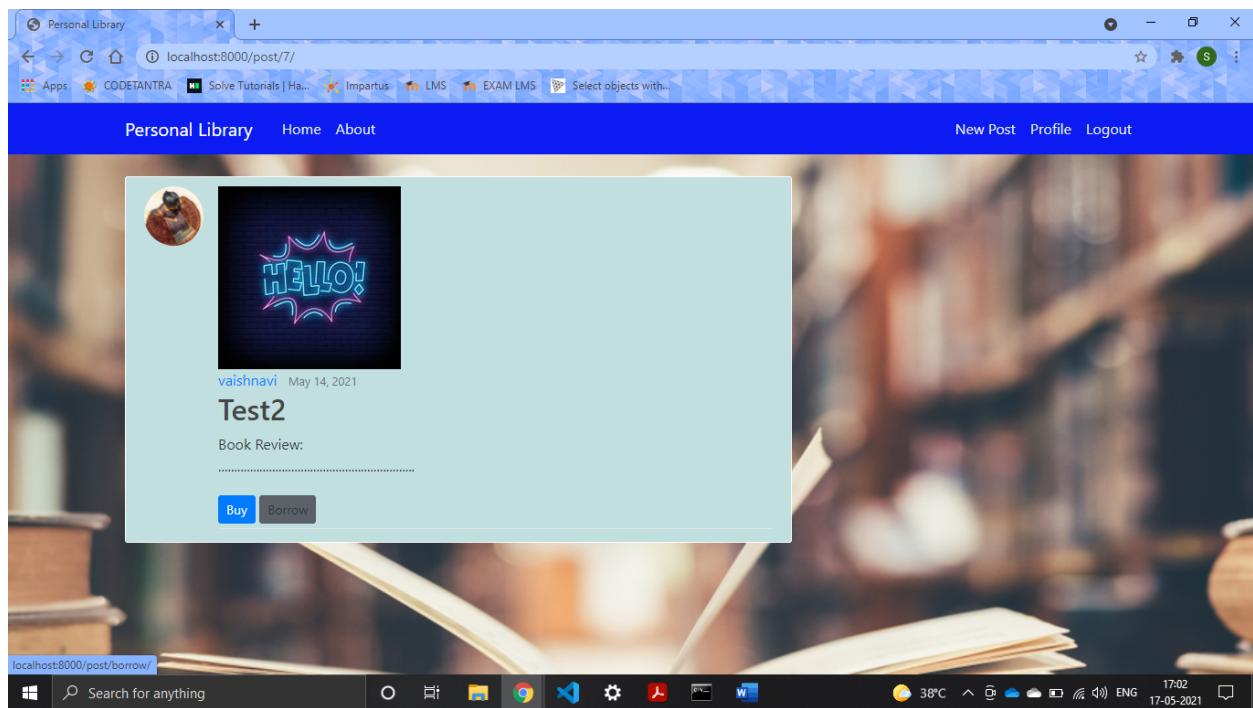
- Test** by Blah Blah Blah
- The Alchemist by Paulo Coelho** by srishti on May 03, 2021. Description: The Alchemist follows the journey of an Andalusian shepherd boy named Santiago. Believing a recurring dream to be prophetic, he asks a Gypsy fortune teller in the nearby town about its meaning. The woman interprets the dream as a prophecy telling the boy that he will discover a treasure at the Egyptian pyramids.
- Post 3** by vaishnavi on May 03, 2021. Description: A book up for exchange!

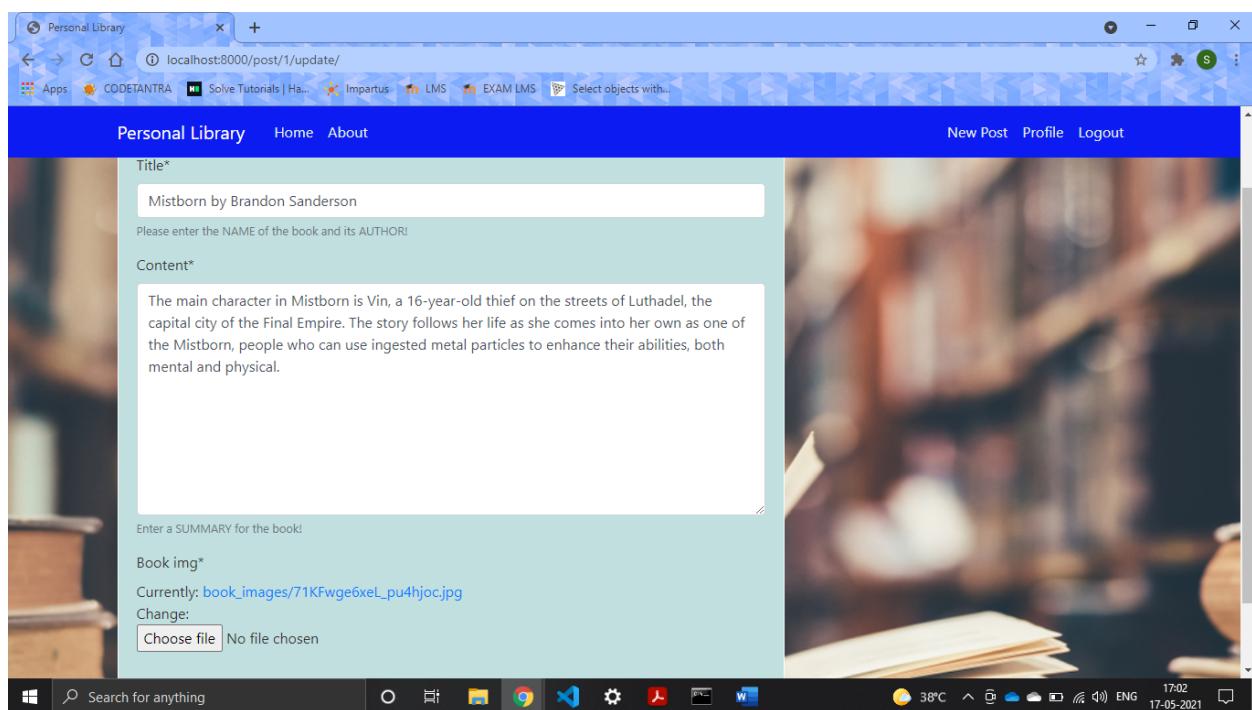
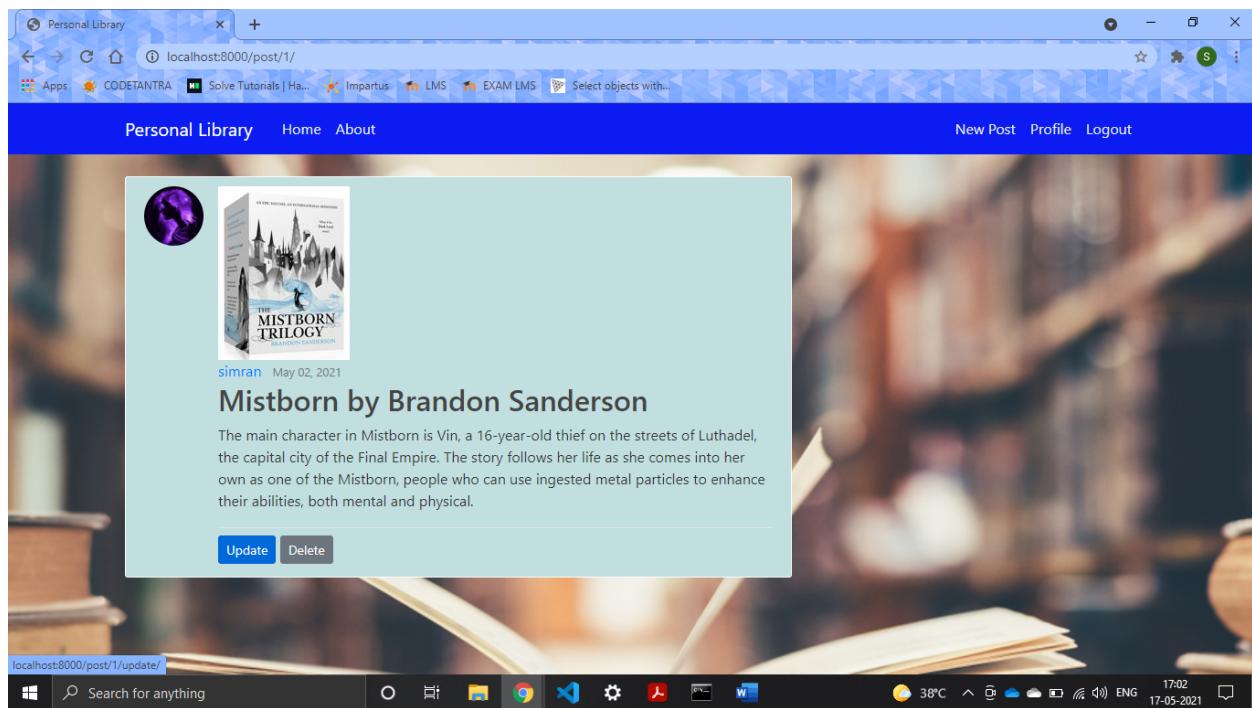
The bottom of the screen shows a navigation bar with buttons for 1, 2, Next, and Last. The bottom status bar shows system information like temperature (38°C), battery level (17%), and date/time (17-05-2021).

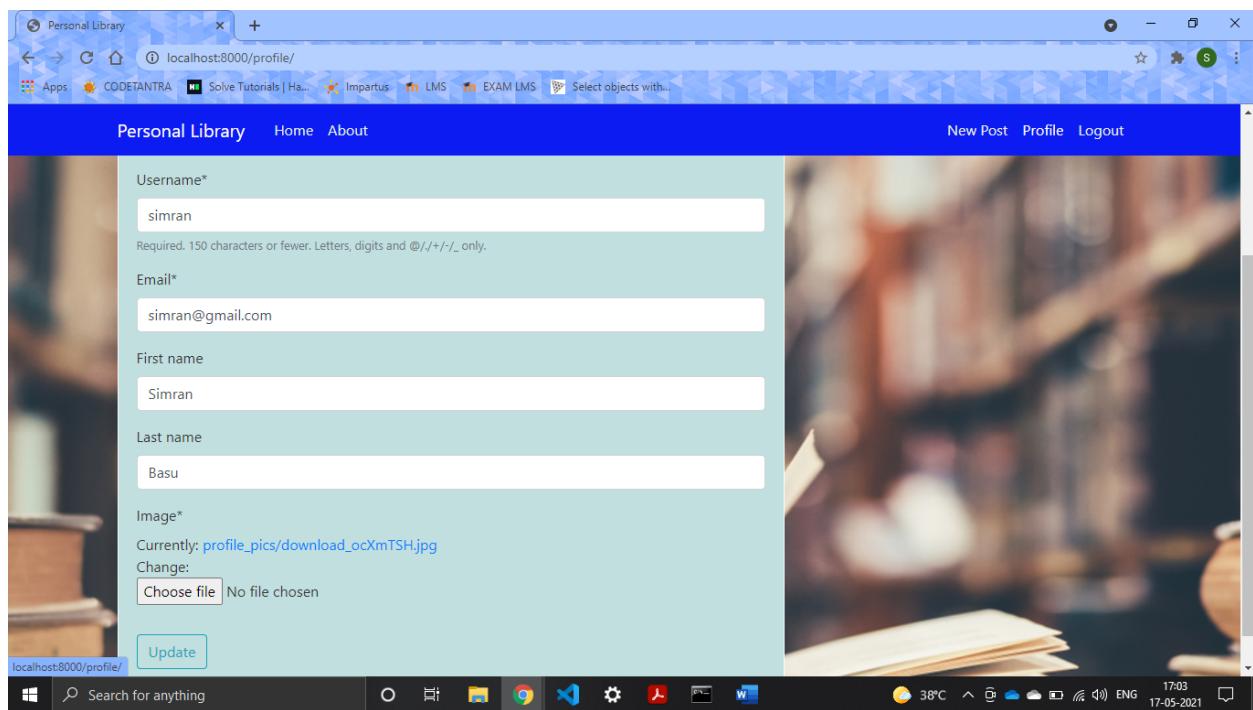
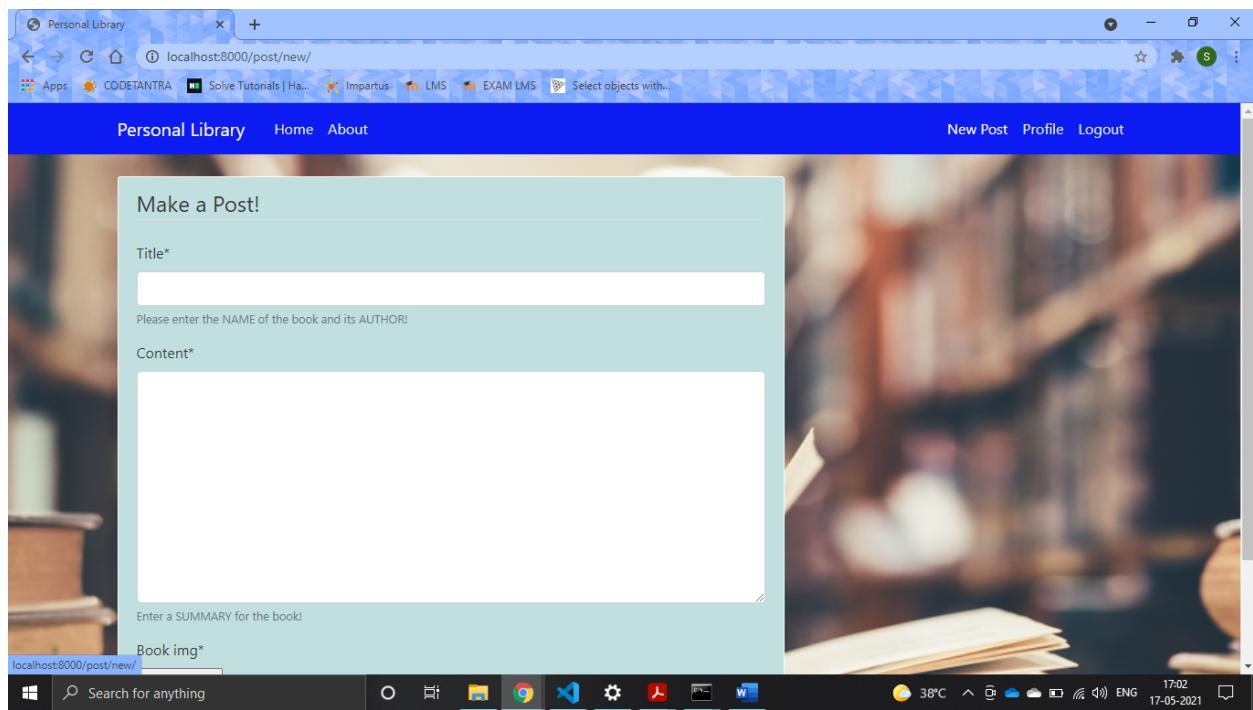


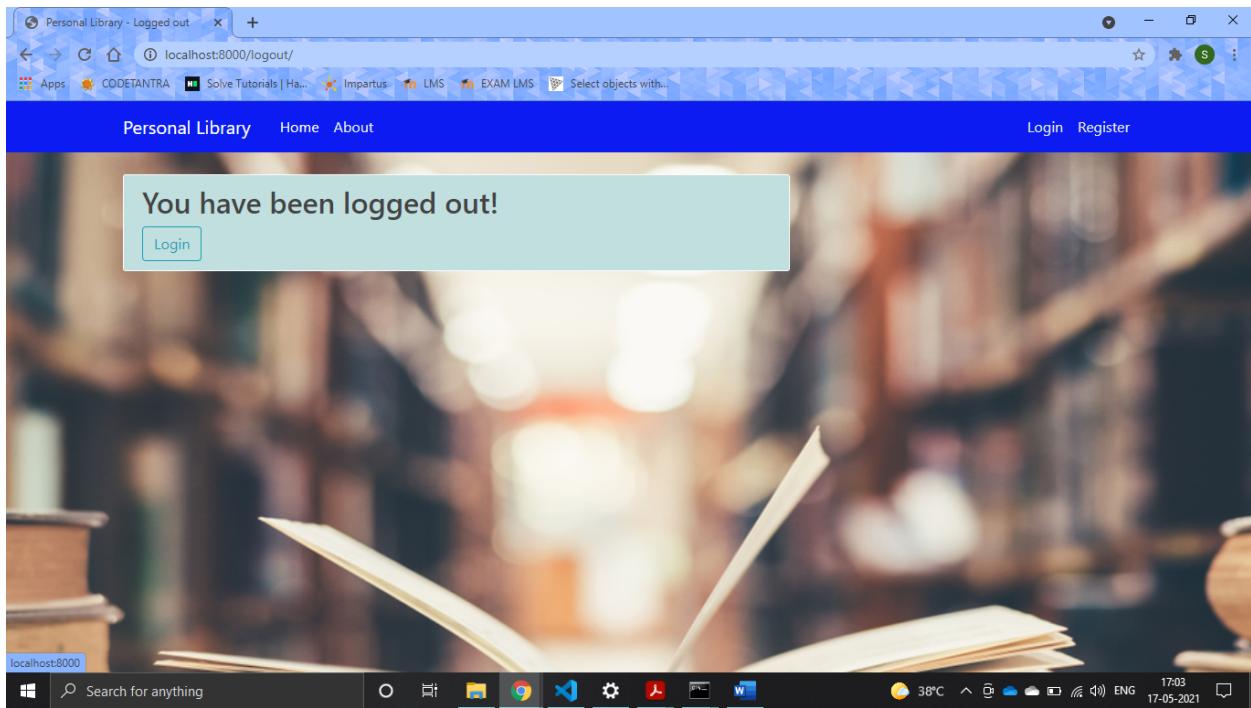












## 13 SCOPE FOR FUTURE WORK

### 13.1 Online Book Clubs

Book club is for a group of users who want to read and discuss a book with other readers. We would like to implement that feature using message queue under a topic.

### 13.2 Permission Approval

We would like to implement a permission approval system for accepting purchase or borrow request when user logs in.

## 14 CONCLUSION

We successfully build a online library platform for all the readers who want to stay connected to reader community using DevOps tools. The tools we used are: GitHub, Jenkins, Docker, Rundeck. These tools are integrated using Jenkins. The entire pipeline has been automated. For deployment of the entire project , it's taking a minimum of 19 minutes and maximum of half an hour.

The Devops methodology and lifecycle tools prove to be better than the Agile methodology in terms of technical, cultural and business benefits. By minimizing friction between independent teams, DevOps enables a collaborative approach for enterprise software

development and delivery that reflects the needs of the entire application lifecycle for today's modern enterprises.

Thus, we can develop, test, deploy and monitor the application easily.

## REFERENCES

- [1] <https://docs.djangoproject.com/en/3.2/intro/reusable-apps/>.
- [2] <https://docs.djangoproject.com/en/3.2/topics/logging/>.
- [3] <https://semaphoreci.com/community/tutorials/dockerizing-a-python-django-web-application>.
- [4] <https://pypi.org/project/django-ansible/>.
- [5] <https://docs.djangoproject.com/en/3.2/topics/testing/>.