

```

1 Lab5. Django Project Automatic Deployment with Jenkins I
2
3 1. Jenkins Repository의 [Lab1.Django 설치하기.pdf]
4 2. Jenkins Repository의 [Lab2.Django Project 시작하기.pdf]의 4번까지 실행
5
6 3. requirements 파일 생성
7 1)프로젝트와 동일한 pip와 pip를 이용한 설치 및 패키지의 버전을 맞추기 위해 requirements.txt 파일을 생성한다.
8 2)최상위 디렉토리(manage.py 파일이 있는 디렉토리)로 이동하여 현재 가상환경에 설치된 pip 목록을 requirements.txt에 저장
9   $ pip freeze > requirements.txt
10
11 3)requirements.txt 목록 확인
12   $ cat requirements.txt
13   asgiref==3.6.0
14   Django==4.2.1
15   sqlparse==0.4.4
16   tzdata==2023.3
17
18 4. Local Repository 생성
19 -상위 디렉토리로 이동
20   $ cd ..
21   $ pwd
22   ~/DjangoHome
23
24   $ ls
25   --Windows의 경우
26   Include      Lib      myproject      Scripts      pyvenv.cfg
27   --macOS의 경우
28   bin          Lib      myproject      pyvenv.cfg
29
30 -git 초기화
31   $ git init
32   -Windows의 경우
33     Initialized empty Git repository in D:/DjangoHome/.git/
34   -macOS의 경우
35     Initialized empty Git repository in /Users/{계정}/DjangoHome/.git/
36
37 -git 필요 환경 설정
38   $ git config user.name 'henry'
39   $ git config user.email 'javaexpert@nate.com'
40
41
42 5. .gitignore 파일 생성
43 1)Github에 push해야 하기 때문에 gitignore 파일이 필요하다.
44 2)아래의 사이트에서 생성한다.
45   https://www.gitignore.io/
46
47 3)Django, venv, Python 입력 후 [생성] 버튼 클릭
48   # Created by https://www.toptal.com/developers/gitignore/api/django,venv,python
49   # Edit at https://www.toptal.com/developers/gitignore?templates=django,venv,python
50
51   ### Django ###
52   *.log
53   *.pot
54   *.pyc
55   __pycache__/
56   local_settings.py
57   db.sqlite3
58   db.sqlite3-journal
59   media
60
61   # If your build process includes running collectstatic, then you probably don't need or want to include staticfiles/
62   # in your Git repository. Update and uncomment the following line accordingly.
63   # <django-project-name>/staticfiles/
64
65   ### Django.Python Stack ###
66   # Byte-compiled / optimized / DLL files
67   *.py[cod]
68   *$py.class
69
70   # C extensions
71   *.so
72
73   # Distribution / packaging
74   .Python
75   build/
76   develop-eggs/
77   dist/
78   downloads/
79   eggs/
80   .eggs/
81   lib/
82   lib64/
83   parts/
84

```

```
85 sdist/
86 var/
87 wheels/
88 share/python-wheels/
89 *.egg-info/
90 .installed.cfg
91 *.egg
92 MANIFEST
93
94 # PyInstaller
95 # Usually these files are written by a python script from a template
96 # before PyInstaller builds the exe, so as to inject date/other infos into it.
97 *.manifest
98 *.spec
99
100 # Installer logs
101 pip-log.txt
102 pip-delete-this-directory.txt
103
104 # Unit test / coverage reports
105 htmlcov/
106 .tox/
107 .nox/
108 .coverage
109 .coverage.*
110 .cache
111 nosetests.xml
112 coverage.xml
113 *.cover
114 *.py,cover
115 .hypothesis/
116 .pytest_cache/
117 cover/
118
119 # Translations
120 *.mo
121
122 # Django stuff:
123
124 # Flask stuff:
125 instance/
126 .webassets-cache
127
128 # Scrapy stuff:
129 .scrapy
130
131 # Sphinx documentation
132 docs/_build/
133
134 # PyBuilder
135 .pybuilder/
136 target/
137
138 # Jupyter Notebook
139 .ipynb_checkpoints
140
141 # IPython
142 profile_default/
143 ipython_config.py
144
145 # pyenv
146 # For a library or package, you might want to ignore these files since the code is
147 # intended to run in multiple environments; otherwise, check them in:
148 # .python-version
149
150 # pipenv
151 # According to pypa/pipenv#598, it is recommended to include Pipfile.lock in version control.
152 # However, in case of collaboration, if having platform-specific dependencies or dependencies
153 # having no cross-platform support, pipenv may install dependencies that don't work, or not
154 # install all needed dependencies.
155 #Pipfile.lock
156
157 # poetry
158 # Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
159 # This is especially recommended for binary packages to ensure reproducibility, and is more
160 # commonly ignored for libraries.
161 # https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
162 #poetry.lock
163
164 # pdm
165 # Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
166 #pdm.lock
167 # pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
168 # in version control.
```

```
169 # https://pdm.fming.dev/#use-with-ide
170 .pdm.toml
171
172 # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
173 __pypackages__/_
174
175 # Celery stuff
176 celerybeat-schedule
177 celerybeat.pid
178
179 # SageMath parsed files
180 *.sage.py
181
182 # Environments
183 .env
184 .venv
185 env/
186 venv/
187 ENV/
188 env.bak/
189 venv.bak/
190
191 # Spyder project settings
192 .spyderproject
193 .spyproject
194
195 # Rope project settings
196 .ropeproject
197
198 # mkdocs documentation
199 /site
200
201 # mypy
202 .mypy_cache/
203 .dmypy.json
204 dmypy.json
205
206 # Pyre type checker
207 .pyre/
208
209 # pytype static type analyzer
210 .pytype/
211
212 # Cython debug symbols
213 cython_debug/
214
215 # PyCharm
216 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
217 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
218 # and can be added to the global gitignore or merged into this file. For a more nuclear
219 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
220 #.idea/
221
222 ### Python ###
223 # Byte-compiled / optimized / DLL files
224
225 # C extensions
226
227 # Distribution / packaging
228
229 # PyInstaller
230 # Usually these files are written by a python script from a template
231 # before PyInstaller builds the exe, so as to inject date/other infos into it.
232
233 # Installer logs
234
235 # Unit test / coverage reports
236
237 # Translations
238
239 # Django stuff:
240
241 # Flask stuff:
242
243 # Scrapy stuff:
244
245 # Sphinx documentation
246
247 # PyBuilder
248
249 # Jupyter Notebook
250
251 # IPython
252
```

```

253 # pyenv
254 # For a library or package, you might want to ignore these files since the code is
255 # intended to run in multiple environments; otherwise, check them in:
256 # .python-version
257
258 # pipenv
259 # According to pypa/pipenv#598, it is recommended to include Pipfile.lock in version control.
260 # However, in case of collaboration, if having platform-specific dependencies or dependencies
261 # having no cross-platform support, pipenv may install dependencies that don't work, or not
262 # install all needed dependencies.
263
264 # poetry
265 # Similar to Pipfile.lock, it is generally recommended to include poetry.lock in version control.
266 # This is especially recommended for binary packages to ensure reproducibility, and is more
267 # commonly ignored for libraries.
268 # https://python-poetry.org/docs/basic-usage/#commit-your-poetrylock-file-to-version-control
269
270 # pdm
271 # Similar to Pipfile.lock, it is generally recommended to include pdm.lock in version control.
272 # pdm stores project-wide configurations in .pdm.toml, but it is recommended to not include it
273 # in version control.
274 # https://pdm.fming.dev/#use-with-ide
275
276 # PEP 582; used by e.g. github.com/David-OConnor/pyflow and github.com/pdm-project/pdm
277
278 # Celery stuff
279
280 # SageMath parsed files
281
282 # Environments
283
284 # Spyder project settings
285
286 # Rope project settings
287
288 # makedocs documentation
289
290 # mypy
291
292 # Pyre type checker
293
294 # pytype static type analyzer
295
296 # Cython debug symbols
297
298 # PyCharm
299 # JetBrains specific template is maintained in a separate JetBrains.gitignore that can
300 # be found at https://github.com/github/gitignore/blob/main/Global/JetBrains.gitignore
301 # and can be added to the global gitignore or merged into this file. For a more nuclear
302 # option (not recommended) you can uncomment the following to ignore the entire idea folder.
303
304 ### Python Patch ###
305 # Poetry local configuration file - https://python-poetry.org/docs/configuration/#local-configuration
306 poetry.toml
307
308 # ruff
309 .ruff_cache/
310
311 # LSP config files
312 pyrightconfig.json
313
314 ### venv ###
315 # Virtualenv
316 # http://iamzed.com/2009/05/07/a-primer-on-virtualenv/
317 [Bb]in
318 [Ii]nclude
319 [Ll]ib
320 [Ll]ib64
321 [Ll]ocal
322 [Ss]cripts
323 pyvenv.cfg
324 pip-selfcheck.json
325
326 # End of https://www.toptal.com/developers/gitignore/api/django,venv,python
327
328 4)생성한 gitignore를 .gitignore 파일에 저장
329 $ nano .gitignore
330
331 5)Commit 및 Push 하기
332 -현재 디렉토리 확인
333 $ pwd
334 ~/DjangoHome
335
336 $ ls

```

```

337 ※.git 디렉토리는 보이지 않는 것이 정상
338 --Window의 경우
339 Include Lib myproject Scripts .gitignore pyvenv.cfg
340 --macOS의 경우
341 bin Lib myproject .gitignore pyvenv.cfg
342

```

```

343 -git add
344 $ git add .
345

```

```

346 -git commit
347 $ git commit -m 'Django Project Create'
348

```

```

349 -Github에서 DjangoHome Repository 생성
350 --README.md 생성 하지 않음.
351 --Public
352

```

```

353 -Github에 remote 설정하고 push 하기
354 $ git remote add origin https://github.com/gitinstructor/DjangoHome.git
355 $ git branch -M main
356 $ git push -u origin main
357 Enumerating objects: 17, done.
358 Counting objects: 100% (17/17), done.
359 Delta compression using up to 8 threads
360 Compressing objects: 100% (16/16), done.
361 Writing objects: 100% (17/17), 5.25 KiB | 2.62 MiB/s, done.
362 Total 17 (delta 3), reused 0 (delta 0), pack-reused 0
363 remote: Resolving deltas: 100% (3/3), done.
364 To https://github.com/gitinstructor/DjangoHome.git
365 * [new branch] main -> main
366 branch 'main' set up to track 'origin/main'.
367

```

```

368 6)Github에서 확인
369 https://github.com/{Github계정}/DjangoHome
370 -myproject
371 -.gitignore
372
373

```

6. EC2 Instance에 Tabby로 접속하기

7. Github Repository Clone 및 몇 가지 설정하기

```

376 1)Github의 DjangoHome Repository Clone 하기
377 $ git clone https://github.com/gitinstructor/DjangoHome.git
378 Cloning into 'DjangoHome'...
379 Username for 'https://github.com': gitinstructor
380 Password for 'https://gitinstructor@github.com': <----Personal Access Token
381 remote: Enumerating objects: 12, done.
382 remote: Counting objects: 100% (12/12), done.
383 remote: Compressing objects: 100% (10/10), done.
384 remote: Total 12 (delta 1), reused 12 (delta 1), pack-reused 0
385 Receiving objects: 100% (12/12), 4.83 KiB | 706.00 KiB/s, done.
386 Resolving deltas: 100% (1/1), done.
387

```

2)몇 가지 설정하기

```

389 -날짜 시간 설정
390 $ sudo timedatectl set-timezone 'Asia/Seoul'
391

```

```

392 -Ubuntu Update
393 $ sudo apt-get update
394 $ sudo apt-get upgrade -y
395 $ sudo apt-get dist-upgrade
396

```

```

397 -Nginx 설치
398 $ sudo apt install -y nginx
399

```

```

400 -pip upgrade
401 $ python --version
402 $ pip list
403 $ python -m pip install --upgrade pip
404

```

```

405 -Green unicorn package 설치
406 $ pip install gunicorn
407 Defaulting to user installation because normal site-packages is not writeable
408 Collecting gunicorn
409 Downloading gunicorn-20.1.0-py3-none-any.whl (79 kB)
410 _____ 79.5/79.5 kB 2.5 MB/s eta 0:00:00
411 Requirement already satisfied: setuptools>=3.0 in /usr/lib/python3/dist-packages (from gunicorn) (59.6.0)
412 Installing collected packages: gunicorn
413 Successfully installed gunicorn-20.1.0
414

```

3)가상환경 생성 후 해당 프로젝트로 이동

```

416 -가상환경 생성
417 $ virtualenv DjangoHome
418 created virtual environment CPython3.10.6.final.0-64 in 501ms
419 creator CPython3Posix(dest=/home/ubuntu/DjangoHome, clear=False, no_vcs_ignore=False, global=False)
420 seeder FromAppData(download=False, pip=bundle, setuptools=bundle, wheel=bundle, via=copy,

```

```

421     app_data_dir=/home/ubuntu/.local/share/virtualenv)
422     added seed packages: pip==23.1.2, setuptools==67.6.1, wheel==0.40.0
423     activators BashActivator,CShellActivator,FishActivator,NushellActivator,PowerShellActivator,PythonActivator
424 -가상환경 활성화하기
425     $ source ./DjangoHome/bin/activate
426
427 -해당 디렉토리로 이동하기
428     $ cd DjangoHome
429
430 4)git 설정
431     $ ls -al
432     ※.git은 보이지 않을 수 있음.
433     bin      lib      myproject    pyvenv.cfg    .git          .gitignore
434
435     $ git config user.name 'henry'
436     $ git config user.email 'javaexpert@nate.com'
437
438
439 5)필요 패키지 설치
440     $ cd myproject
441
442     $ ls
443     manage.py      myproject      requirements.txt
444
445     $ pip install -r requirements.txt
446     Collecting asgiref==3.6.0 (from -r requirements.txt (line 1))
447       Using cached asgiref-3.6.0-py3-none-any.whl (23 kB)
448     Collecting Django==4.2.1 (from -r requirements.txt (line 2))
449       Using cached Django-4.2.1-py3-none-any.whl (8.0 MB)
450     Collecting sqlparse==0.4.4 (from -r requirements.txt (line 3))
451       Using cached sqlparse-0.4.4-py3-none-any.whl (41 kB)
452     Collecting tzdata==2023.3 (from -r requirements.txt (line 4))
453       Using cached tzdata-2023.3-py2.py3-none-any.whl (341 kB)
454     Installing collected packages: tzdata, sqlparse, asgiref, Django
455     Successfully installed Django-4.2.1 asgiref-3.6.0 sqlparse-0.4.4 tzdata-2023.3
456
457 6)Django Project 시작하기
458     $ python manage.py runserver 0:8000
459     Watching for file changes with StatReloader
460     Performing system checks...
461
462     System check identified no issues (0 silenced).
463
464     You have 18 unapplied migration(s). Your project may not work properly until you apply the migrations for app(s): admin,
465     auth, contenttypes, sessions.
466     Run 'python manage.py migrate' to apply them.
467     May 17, 2023 - 05:07:24
468     Django version 4.2.1, using settings 'myproject.settings'
469     Starting development server at http://127.0.0.1:8000/
470     Quit the server with CONTROL-C.
471
472 7)Web Browser로 확인하기
473     http://ec2-43-201-250-26.ap-northeast-2.compute.amazonaws.com:8000
474
475 8)기본테이블 생성하기
476     $ python manage.py migrate
477     Operations to perform:
478       Apply all migrations: admin, auth, contenttypes, sessions
479     Running migrations:
480       Applying contenttypes.0001_initial... OK
481       Applying auth.0001_initial... OK
482       Applying admin.0001_initial... OK
483       Applying admin.0002_logentry_remove_auto_add... OK
484       Applying admin.0003_logentry_add_action_flag_choices... OK
485       Applying contenttypes.0002_remove_content_type_name... OK
486       Applying auth.0002_alter_permission_name_max_length... OK
487       Applying auth.0003_alter_user_email_max_length... OK
488       Applying auth.0004_alter_user_username_opts... OK
489       Applying auth.0005_alter_user_last_login_null... OK
490       Applying auth.0006_require_contenttypes_0002... OK
491       Applying auth.0007_alter_validators_add_error_messages... OK
492       Applying auth.0008_alter_user_username_max_length... OK
493       Applying auth.0009_alter_user_last_name_max_length... OK
494       Applying auth.0010_alter_group_name_max_length... OK
495       Applying auth.0011_update_proxy_permissions... OK
496       Applying auth.0012_alter_user_first_name_max_length... OK
497       Applying sessions.0001_initial... OK
498
499 9)기본테이블 확인하기
500     $ ls
501     db.sqlite3      manage.py      myproject      requirements.txt
502
503 10)관리자 계정 생성

```

```
503 $ python manage.py createsuperuser
504 Username (leave blank to user 'ubuntu') : admin
505 Email address: javaexpert@nate.com
506 Password : *****
507 Password (again) : *****
508 This password is too common.
509 Bypass password validation and create user anyway? [y/N] : y
510 Superuser created successfully.
511
```

512 8. 관리자모드 접속하기

```
513 -웹브라우저에서 확인
514 $ python manage.py runserver 0:8000
515
516 http://{EC2 퍼블릭 IPv4 DNS}:8000/admin
517 Username: admin
518 Password :
```

519 9. Django Project에 main App 생성하기

```
520 1)manage.py가 있는 최상위 디렉토리에서 다음과 같이 main App을 생성한다.
521 $ python manage.py startapp main
```

522 2)myproject > settings.py 수정하기

```
523 ...
524 INSTALLED_APPS = [
525     'main',          <----여기
526     'django.contrib.admin',
527     'django.contrib.auth',
528     'django.contrib.contenttypes',
529     'django.contrib.sessions',
530     'django.contrib.messages',
531     'django.contrib.staticfiles',
532 ]
533 ...
534 LANGUAGE_CODE = 'ko-kr'
535
536 TIME_ZONE = 'Asia/Seoul'    <---TimeZone 설정
537 ...
```

538 3)index.html 파일 생성하기

```
539 -새로 생성된 main 디렉토리로 이동
```

```
540 $ cd ..
541 $ cd main
```

```
542 -templates 디렉토리 생성 후 이동
```

```
543 $ mkdir templates
544 $ cd templates
```

```
545 -main 디렉토리 생성 후 이동
```

```
546 $ mkdir main
547 $ cd main
```

```
548 -index.html 파일 생성
```

```
549 $ nano index.html
550 <!DOCTYPE html>
551 <html lang="en">
552 <head>
553     <meta charset="UTF-8">
554     <meta http-equiv="X-UA-Compatible" content="IE=edge">
555     <meta name="viewport" content="width=device-width, initial-scale=1.0">
556     <title>Welcome to My Django Project</title>
557 </head>
558 <body>
559     <h1>Hello Django Project</h1>
560 </body>
561 </html>
```

562 4)main > views.py 수정하기

```
563 $ nano views.py
564 from django.shortcuts import render
565
566 # Create your views here.
567 def index(request):
568     return render(request, "main/index.html")
```

569 5)myproject > urls.py 수정하기

```
570 $ nano urls.py
571 """
572 URL configuration for myproject project.
```

```
573
574 The `urlpatterns` list routes URLs to views. For more information please see:
575 https://docs.djangoproject.com/en/4.2/topics/http/urls/
```

```

587 Examples:
588 Function views
589     1. Add an import: from my_app import views
590     2. Add a URL to urlpatterns: path("", views.home, name='home')
591 Class-based views
592     1. Add an import: from other_app.views import Home
593     2. Add a URL to urlpatterns: path("", Home.as_view(), name='home')
594 Including another URLconf
595     1. Import the include() function: from django.urls import include, path
596     2. Add a URL to urlpatterns: path('blog/', include('blog.urls'))
597 """
598 from django.contrib import admin
599 from django.urls import path
600 from main import views as main_views          <---여기 수정
601
602 urlpatterns = [
603     path("", main_views.index, name='index'), <---여기 수정
604     path('admin/', admin.site.urls),
605 ]
606
607 6)기존 Model들을 migrate하고, Django Project 시작하기
608 $ python manage.py makemigrations && python manage.py migrate
609 No changes detected
610 Operations to perform:
611   Apply all migrations: admin, auth, contenttypes, sessions
612 Running migrations:
613   No migrations to apply.
614
615 $ python manage.py runserver 0:8000
616
617 7)Web Browser를 통해 확인하기
618 http://{EC2 퍼블릭 IPv4 DNS}:8000
619
620
621 10. 변경사항 Github에 push
622 $ git status
623 On branch main
624 Your branch is up to date with 'origin/main'.
625
626 Changes not staged for commit:
627   (use "git add <file>..." to update what will be committed)
628   (use "git restore <file>..." to discard changes in working directory)
629     modified:   myproject/myproject/settings.py
630     modified:   myproject/myproject/urls.py
631
632 Untracked files:
633   (use "git add <file>..." to include in what will be committed)
634     myproject/main/
635
636 no changes added to commit (use "git add" and/or "git commit -a")
637
638 $ git add .
639 $ git status
640 On branch main
641 Your branch is up to date with 'origin/main'.
642
643 Changes to be committed:
644   (use "git restore --staged <file>..." to unstage)
645     new file:   myproject/main/__init__.py
646     new file:   myproject/main/admin.py
647     new file:   myproject/main/apps.py
648     new file:   myproject/main/migrations/__init__.py
649     new file:   myproject/main/models.py
650     new file:   myproject/main/templates/main/index.html
651     new file:   myproject/main/tests.py
652     new file:   myproject/main/views.py
653     modified:   myproject/myproject/settings.py
654     modified:   myproject/myproject/urls.py
655
656 $ git commit -m 'Add main App to Django Project'
657 [main f11b7f7] Add main App to Django Project
658 10 files changed, 37 insertions(+), 2 deletions(-)
659 create mode 100644 myproject/main/__init__.py
660 create mode 100644 myproject/main/admin.py
661 create mode 100644 myproject/main/apps.py
662 create mode 100644 myproject/main/migrations/__init__.py
663 create mode 100644 myproject/main/models.py
664 create mode 100644 myproject/main/templates/main/index.html
665 create mode 100644 myproject/main/tests.py
666 create mode 100644 myproject/main/views.py
667
668 $ git push origin main
669 Username for 'https://github.com': gitinstructor
670 Password for 'https://gitinstructor@github.com':

```



```
671 Enumerating objects: 21, done.
672 Counting objects: 100% (21/21), done.
673 Compressing objects: 100% (13/13), done.
674 Writing objects: 100% (16/16), 1.82 KiB | 930.00 KiB/s, done.
675 Total 16 (delta 2), reused 0 (delta 0), pack-reused 0
676 remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
677 To https://github.com/gitinstructor/DjangoHome.git
678   cf69166..f11b7f7  main -> main
679
680
681 11. uWSGI 서버 연결하기
682   1) Django repository의 [Lab.uWSGI 서버 연결하기.pdf]를 참조
683
684   2) 성공하면 Github에 Push한다.
685     $ git status
686       On branch main
687       Your branch is up to date with 'origin/main'.
688
689       Untracked files:
690         (use "git add <file>..." to include in what will be committed)
691         myproject/.config/
692
693       nothing added to commit but untracked files present (use "git add" to track)
694
695     $ git add .
696
697     $ git status
698       On branch main
699       Your branch is up to date with 'origin/main'.
700
701       Changes to be committed:
702         (use "git restore --staged <file>..." to unstage)
703         new file:   myproject/.config/uwsgi/myproject.ini
704
705     $ git commit -m "Add myproject.ini File"
706       [main 6d34994] Add myproject.ini File
707       1 file changed, 16 insertions(+)
708       create mode 100644 myproject/.config/uwsgi/myproject.ini
709
710     $ git push origin main
711       Username for 'https://github.com': gitinstructor
712       Password for 'https://gitinstructor@github.com':
713       Enumerating objects: 8, done.
714       Counting objects: 100% (8/8), done.
715       Compressing objects: 100% (4/4), done.
716       Writing objects: 100% (6/6), 646 bytes | 646.00 KiB/s, done.
717       Total 6 (delta 1), reused 0 (delta 0), pack-reused 0
718       remote: Resolving deltas: 100% (1/1), completed with 1 local object.
719       To https://github.com/gitinstructor/DjangoHome.git
720         f11b7f7..6d34994  main -> main
721
722
723 12. nginx 연결하기
724   1) Django repository의 [Lab.nginx 연결하기.pdf]를 참조
```