1. 빌드 및 배포 방법

1) 사용한 기술 및 IDE 종류와 버전

FE

- VS Code
- TypeScript
- React
- Recoil
- WebSocket (STOMP)
- PWA

BE

- IntelliJ
- JDK 17
- Spring Boot 3.2.4
- Gradle 8.5
- MariaDB 10.3.23 (MySQL 8.0.36)
- MongoDB 5.0.26

Al / BigData

- KoELECTRA
- JDK 8
- Spark 3.5.1
- Python 3.12.2
- FastAPI
- uvicorn

INFRA

- AWS EC2
- AWS S3
- Jenkins
- docker
- nginx 1.18.0

• RabbitMQ 3.8.2

2) 빌드 시 사용되는 환경 변수

BE - application.yml

```
spring:
  data:
    mongodb:
      uri: mongodb+srv://S10P22C108:1znS65MER5@ssafy.ngivl.mongodb.net/S10P22C108
                 ?authSource=admin
  datasource:
    url: jdbc:mysql://stg-yswa-kr-practice-db-master.mariadb.database.azure.com:3306
               /S10P22C108?serverTimezone=UTC&useUnicode=true&characterEncoding=utf8
    username: S10P22C108@stg-yswa-kr-practice-db-master.mariadb.database.azure.com
    password: iF6wrgXGd6
    driver-class-name: com.mysql.cj.jdbc.Driver
  jpa:
    hibernate:
      ddl-auto: update
    show-sql: true
  security:
    oauth2:
      client:
        registration:
          kakao:
            client-id: [kakao_app_key_rest_api_key]
            redirect-uri: https://j10c108.p.ssafy.io/login/oauth2/code/kakao
            client-authentication-method: none
            authorization-grant-type: authorization_code
            scope: profile_nickname, profile_image, account_email
        provider:
          kakao:
            authorization-uri: https://kauth.kakao.com/oauth/authorize
            token-uri: https://kauth.kakao.com/oauth/token
            user-info-uri: https://kapi.kakao.com/v2/user/me
            user-name-attribute: id
    jwt:
      secret-key: [jwt-secret-key]
  rabbitmq:
    username: guest
    password: guest
    host: j10c108.p.ssafy.io
    port: 5672
    queue:
      name: messages
    exchange:
      name: message_exchange
server:
```

```
ssl:
    enabled: true
    enabled-protocols: TLSv1.2
    key-store: classpath:keystore.p12
    key-store-password: ssafy
    key-store-type: PKCS12

fast-api:
    url: http://j10c108a.p.ssafy.io:8881
```

FE - .env

```
# kakao App Key - REST API Key
VITE_KAKAO_CLIENT_ID = ["kakao_app_key_rest_api_key"]

# kakao Auth Code URL
VITE_KAKAO_0AUTH_URL = "https://kauth.kakao.com/oauth"

# kakao Login Redirect URI
VITE_KAKAO_LOGIN_REDIRECT_URI = "https://j10c108.p.ssafy.io/login/oauth2/code/kakao"
VITE_KAKAO_LOGOUT_REDIRECT_URI = "https://j10c108.p.ssafy.io/logout/oauth2/code/kakao"

# BE - Spring Server URL
VITE_API_URL = 'https://j10c108.p.ssafy.io:8080'

# AWS S3 Info
VITE_AWS_ACCESS_KEY_ID=["AWS_S3_ACCESS_KEY_ID"]
VITE_AWS_SECRET_ACCESS_KEY=["AWS_S3_SECRET_ACCESS_KEY"]
VITE_AWS_REGION=ap-northeast-2
```

3) 배포 시 특이사항

INFRA - Fire Wall Port

아래의 표와 같이 각각의 서버에서 포트 접근을 허용하도록 설정해주세요.

[FE/BE Server]		Port #
INFRA	NGINX	80
	Jenkins	8888
FE	React - VITE	3000
BE	Spring Boot	8080
	RabbitMQ	5672 (AMQP), 15672 (RabbitMQ), 61613 (STOMP)
[AI / Big Data Server]		
AI / Big Data	FastAPI	8881

INFRA - Jenkins

젠킨스 실행

```
sudo docker run -d -e JENKINS_OPTS=--httpPort=8888 \
    -p 8888:8888 -v jenkins_home:/var/jenkins_home \
     -v /var/run/docker.sock:/var/run/docker.sock \
     --name jenkins-server jenkins/jenkins:lts-jdk17
```

ienkins docker container 내부에 docker 설치

```
docker exec --user root -it jenkins-server apt-get update
docker exec --user root -it jenkins-server apt-get install -y docker.io
```

도커 그룹을 생성하고 사용자를 해당 그룹에 추가하여 Jenkins 작업에서 도커 명령어를 실행할 수 있습니다. 이를 위해 다음과 같은 단계를 따를 수 있습니다.

도커 그룹 생성:

도커 그룹을 생성하려면 먼저 컨테이너 내부에서 해당 그룹의 GID(GNU IDentifier)를 설정해야 합니다. 일반적으로 호스트와 동일한 GID를 사용하는 것이 좋습니다. 호스트에서 다음 명령을 실행하여 도커 그룹의 GID를 확인할 수 있습니다.

```
getent group docker
```

도커 그룹 생성 및 사용자 추가:

도커 그룹을 생성하고 사용자를 해당 그룹에 추가합니다.

```
docker exec --user root -it jenkins-server groupadd -g <GID> docker
docker exec --user root -it jenkins-server usermod -aG docker jenkins
```

여기서 <GID>는 이전 단계에서 확인한 도커 그룹의 GID입니다.

권한 변경:

생성된 도커 그룹의 권한을 변경하여 Jenkins 사용자가 도커 소켓에 액세스할 수 있도록 합니다.

```
docker exec --user root -it jenkins-server chown root:docker /var/run/docker.so ck
docker exec --user root -it jenkins-server chmod 660 /var/run/docker.sock
```

이제 Jenkins 사용자가 도커 그룹에 속해 있고 도커 그룹은 도커 소켓에 대한 읽기 및 쓰기 권한을 갖도록 설정되었습니다. 따라서 Jenkins 작업에서 도커 명령을 실행할 수 있게 되었습니다.

back configure:

gure	
-	General Enabled 💽
eral Inced Project Options	설명
line	
	Plain test (PPM2)
	Do not allow concurrent builds
	Do not allow the pipeline to resume if the controller restarts Gitab Connection
	SILEO COMPRODON
	Use alternative credential
	Gitlab Repository Nome 👚
	Pipeline apeed/durability coveride (*)
	Preserve stashes from completed bailds ②
	☐ Throttle builds ②
	_ 오래된 별도 약체 (중)
	Build Triggers
	Build after other projects are built
	Build periodically
	Enabled Gittab triggers Push Events **
	Push Events in case of branch delete (*) Opened Merge Request Events (*)
	☐ Build only if new commits were pushed to Merge Request.
	Accepted Merge Request Events (*) Closed Merge Request Events (*)
	Rebuild open Merge Requests 👚
	On push to source or target branch
	Approved Marge Requests (EE-cm)d ® Comments ®
	Comment (regre) for triggering a build (®)
	back
	20 ^
	Endble (cl-skip) (*) Ignore WP Merge Requests (*)
	Labels that launch a build if they are added (comma-separated)
	Set build description to build cause (eg. Merge request or Git Push) Build on successful pipeline events
	Pending build name for pipeline (2)
	Cancel pending merge request builds on update ® Allowed branches
	Allow all branches to trigger this job (§)
	Filter branches by name (2) Filter branches by roger (2)
	Filter merge request by label
	Secret token (T)
	38338caa68d5b47eAn5cl/bc65ca654e7
	George
	Clear
	Generic Webbook Trigger (*)
	Poll SCM ®
	Pel SCM 중 Quar presot 중 설도용 중격으로 구함 (에 스크립로 사용) 중
	Pel SCM 중 Gar prend 중 설도용 단적으로 되었다. 사용) 중 설도용 단적으로 되었다. 사용) 중 Advanced Project Options
	Pel SCM 중 Quar presot 중 설도용 중격으로 구함 (에 스크립로 사용) 중
	Pel SCM 중 Gar prend 중 설도용 단적으로 되었다. 사용) 중 설도용 단적으로 되었다. 사용) 중 Advanced Project Options
	Pel SCM ® Gar prend ® Mark Branc SW (et an ME Are) ® Advanced Project Options 28 ∨ Pipeline Outrition
	Pat XX4 ® Quate princed Quate princed
	Pel SCM ® Gar prend ® Mark Branc SW (et an ME Are) ® Advanced Project Options 28 ∨ Pipeline Outrition
	Pal KAN # Quart period (\$ SEE #ERIOL REP (# CARDE 48) # Municipal Project Options 28 \rightarrow Pipoline Definition Pales soles from SOM SOM #
	Del XXM ® Caste princed Size & BETACLE REG (A CASTE ARE) Advanced Project Options
	Del KAM III Quite prient III Quite prient III Quite prient III Quite III Quite III Quite III Advanced Project Options
	Del XXM ® Caste princed Size & BETACLE REG (A CASTE ARE) Advanced Project Options
	Del SCAL ® Qual primed (® SEE 8 TRUE RED (R CARDE 148) ® Multiple RED RED RED (R CARDE 148) ® Advanced Project Options 28
	Del SCAL SE Caste princed SE SE SE SERCE SE (II A. SE
	Dat SAM # Calar princid SER # \$100.0 R (4 A.0.0 R + 48) Advanced Project Options
	Dat SCAL SE Clast princed SE SE SE SEAL RE RE (A ELECT PRE) SE Advanced Project Options 28
	Data KAM # Clark prived SER # \$100.0 R (4 0 R H H H H H H H H H H H H H H H H H H
	Data KAM # Quart prived SEE # 8 H2 QUE (4 A A SEE # 19) Marie # 8 H2 QUE (4 A A SEE # 19) Advanced Project Options 20 * Pipeline Data See Pipeline Data See Fipeline
	Dat KM ® Quit priend ® See # 2012 RB (# 1.2015 MB) ® Advanced Project Options BB V Pipeline Deferior Protes oxige from SSM V SSM ® GR V Repositories ® Repositories & Repos
	Data KAM # Caster primed SEE # \$100.0 R M (# A.D.M. # # # # # # # # # # # # # # # # # #
	Dat KAM ® Quate primed ® Primed
	Data KAM # Caster primed SEE # \$100.0 R M (# A.D.M. # # # # # # # # # # # # # # # # # #
	Dat KAL SE Quate primed S Quate primed S SE SE PLOS ARE (4) (A) (B) S Advanced Project Options 20 V Pipeline Didition Projection SCM SCM S Reportion S Add to contain S Report S Add to contain S Add to contai
	Dat KM ® Quit priend ® Quit priend ® Advanced Project Options 30
	Data State State Control of State prince Control of State Control of State prince Cont

GitLab에서 코드 변경이 있을 경우, 자동으로 빌드하도록 설정합니다.

front configure:

rd > front > Configuration		
figure	General	Enabled 🕢
eneral	48	
dvanced Project Options peline		
	Plain text: 미리보기	a a
	Do not allow concurrent builds	
	Do not allow the pipeline to resume if the controller restarts	
	GitLab Connection	
		~
	Use alternative credential GitLab Repository Name (?)	
	uttao nepository Name f	
	☐ Pipeline speed/durability override ②	
	Preserve stashes from completed builds	
	Throttle builds 🔞	
	오래된 별도 삭제 ②	
	이 별도는 매개변수가 있습니다 (?)	
	Build Triggers	
	Build after other projects are built Build periodically Build periodically	
	 Build when a change is pushed to GitLab. GitLab webhook URL: http://j10c108.p.ssafyiois888/project/front ? Enabled GitLab triggers 	
	✓ Push Events ②	
	Push Events in case of branch delete ② Opened Merge Request Events ②	
	Build only if new commits were pushed to Merge Request ?	
	✓ Accepted Merge Request Events ② Closed Merge Request Events ②	
	Rebuild open Merge Requests ?	
	On push to source branch	¥
	Approved Merge Requests (EE-only) ✓ Comments ?	
	Comment (regex) for triggering a build (?)	
	front	
	□ ZB ✓ □ Generic Webhook Tripper (*) □ Poll Scott (*) □ Out seeten! (*)	
	Genetic Webhook Trigger 중 Poll SCM (중) Quiet period 중 발표를 참적으로 유럽 (에: 스크립트 사용) 중 Advanced Project Options	
	Genetic Webhook Tripper ® Poll SCM (중) Quiet period (중) 생도를 원적으로 유발 (약. 스크링트 사용) 중)	
	Genetic Webhook Trigger 중 Poll SCM (중) Quiet period 중 발표를 참적으로 유럽 (에: 스크립트 사용) 중 Advanced Project Options	
	□ Genetic Webhook Tripper ② □ Poli SCM ③ □ Quiet period ② □ WEB RAICE 유발 (N: △田田트 사용) ② Advanced Project Options □ 대 □ V	
	□ Genetic Webhook Tripper ② □ Poli SCM ③ □ Quiet period ② □ WEB RAI CE 유발 (N: △田田트 사용) ② Advanced Project Options □ □ ✓ Pipeline	v
	□ Genetic Webhook Tripper ② □ Poll SCM ③ □ Quiet period ② □ WE 是 점 의으로 유발 (特: 스크램트 사용) ② Advanced Project Options ② ② Pipeline Definition Pipeline soriet from SCM.	
	□ Genetic Webhook Tripper ② □ Poll SCM ② □ Quiet period ② □ 版 是	v
	□ Genetic Webhook Tripper ② □ Red SCM ② □ Quiet period ② □ 版 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图 图	
	□ Genetic Webhook Tripper ② □ Poll SCM ② □ Quiet period ② □ 版 是	v (8
	□ Genetic Webhook Tripper ② □ Poll SCM ② □ Quiet period ② □ 版生音 원지으로 유행 (株 - 二祖星 사용) ② Advanced Project Options 교육 ∨ Pipeline Definition Pipeline scopt from SCM SCM ② □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □ □	v (8
	□ Genetic Webhook Tripper ② □ Pol SCM ② □ Quiet period ② □ W 医唇唇可足 中枢(性 - 二母型 - 八名型 - 八名	v (8
	□ Genetic Webhook Trigger ② □ Pal SCM ③ □ Quiet period ② □ WE書 점적으로 숙발 (박 스크림트 사용) ② Advanced Project Options □급 ~ Pipeline Definition Pipeline copy from SCM SCM ② □ Got Repositorly URL ② Repositorly URL ③ Repositorly URL ③ Repositorly URL ③ Repositorly URL ④ Repositorly URL ⑥ Reposi	▼ ●
	□ Genetic Webhook Trigger 学 □ Pol SCM (学 □ Valet period で □ Valet period で □ Valet Project Options コネッ Pipeline Definition Epinition Fipeline Scotia で □ Option SCM SCM で □ Option SCM Fipeline Scotia で Properties (学)	▼ ●
	□ Genetic Webhook Tripper ② □ Poll SCM ② □ Quiet period ② □ 恒星音 ROM ② □ 二祖星 사용)② Advanced Project Options 교육 ▼ Pipeline Definition Pipeline script from SCM SCM ② □ □ Git Repository URL ③ □ https://wib.sas/y.com/10-big/date-dist-sub2/516P22C108.git Corderviol ③ ■ Websock/Subgachon.ac.ky******* + Add ■	▼ ●
	□ Genetic Webhook Tripper 学 □ Pol SCM (学 □ Quiet period 学 □ WEB 등적으로 유행 (特. 스크림트 사용) 学 Advanced Project Options 교급 ~ Pipeline Defenition Pipeline script from SCM SCM (学 □ Got Repositorly URL (于 □ Interpr/lab.sos/scom/110-bigdate-dist-sub2/516922C108.ght Credential Wesson/scom/110-bigdate-dist-sub2/516922C108.ght Credential Wesson/scom/110-bigdate-dist-sub2/516922C108.ght Credential Wesson/scom/110-bigdate-dist-sub2/516922C108.ght	▼ ●
	□ Genetic Webhook Trigger 学 □ Pol SCM (学 □ Quiet period 学 □ 場合者を全産 会管 (特に会議性 小谷) 学 Advanced Project Options ユヨ マ Pipeline Definition Pipeline soly from SCM SCM (学 □ Git □ Repository URL (学) □ Hottps://doi.box.org/com/10-bigsidate-dist-sub2/530P22C108.git □ Condential は □ Vegocolysiogachon.ac.lc/***********************************	▼ ●
	□ Genetic Webhook Tripper ② □ Pol SCM ③ □ Quiet period ② □ W 医唇 함께 C 스크림트 사용 ② Advanced Project Options 교육 ➤ Pipeline Definition Pipeline script from SCM SCM ④ Got Repositories ③ Repositories ③ Repositories ③ Repositories ③ Add Repository URL ② Intropy/Abb.ass/y.com/110-bigdate-dist-sub2/516F22C108.git Credentials — Add Repository Add Repository	▼▼▼
	□ Central Control Project Project Project Options Advanced Project Options 正日 Pipeline Definition Pipeline Definition Fipeline script from SCM SCM ② Got Repositoriy URL ② Interprivate script from SCM SCM ② Repositoriy URL ② Interprivate script from SCM Repositoriy URL ② Interprivate script from SCM Repositoriy URL ② Interprivate script from SCM Repositoriy URL ③ Interprivate script from SCM Repositoriy URL ③ Repositoriy URL ③ Interprivate script from SCM Repositoriy URL ③ Repositoriy URL ⑥ Re	▼▼▼
	□ Genetic Webhook Trigger 学 □ Pol SCM (学 □ Quiet period 学 □ 場合 報告を表現 (単に会議性 人名 会議 (単に会議性 人名 会議性 (単に会議性 (▼▼▼
	□ Genetic Webhook Trigger 学 □ Pol SCM (学 □ Vaset period 学 □ Vaset period 学 □ Vaset Project Options Advanced Project Options ココマ Pipeline Definition Fipeline SCM SCM 学 □ Ott Repository URL 学 Haptor/Massay.com/10-bigsdate-dist-sub2/510P22C108.git Cordentials 学 □ Vased Project Options (学) Add Repository URL 学 Haptor/Massay.com/10-bigsdate-dist-sub2/510P22C108.git Cordentials 学 □ Value Option (Passay Com/10-bigsdate-dist-sub2/510P22C108.git Add Repository Branch Specifier (Stark for Smy) 学 Yydowslop Add Branch Specifier (Stark for Smy) 学 Yydowslop Add Branch Specifier (Stark for Smy) 学	▼▼▼
	□ Generic Wethook Tripper ② □ Pol SCM ③ □ Quiet period ② □ WE 를 함적으로 유명 (에. 스크림트 사용) ② Advanced Project Options 교육 ➤ Pipeline Definition Fipeline script from SCM SCM ③ □ Repository URL ③ Introv/Abbosdy.com/10-bigdate-dist-ub/2/510F22C10ligit Cendentials ③ Wessoch@apachon.ac.kr/****** **Add ** #################################	v
	□ Generic Wethook Trigger 学 □ Pol SCM (学 □ Vacet period 学 □ Vacet period Project Options ② ② Pipeline Definition Fipeline solution Fipeline solution Fipeline country from SCM SCM (学 □ Opt □ Perpository URL (学 □ Introv/Alb basely/com/Alb bigdate-dist-aub2/2510P22C108 git □ Cordential (*) □ Vacedential (*) □ Vaced	v
	□ Genetic Webhock Trigger 学 □ Pol SCM (学 □ Valet period 学 □ Valet period 学 □ Valet period 学 □ Valet period 学 □ Valet Period Project Options □ □ V Pipeline Definition □ Pepline solget from SCM SCM (学) □ Integral/fish assign com/10 big data-data-sub2/516P22C108.git Peppolitories (学) Peppolitories (学) Peppolitories (学) Peppolitories (**) Peppolitorie	v
	□ Central Control Project □ Pal SCAN (章 □ Quiet period (章) □ ME 를 등적으로 유병 (에- 스크림트 사용) (章) Advanced Project Options 교급 ➤ Pipeline Definition Fipeline script from SCAM SCAN (章 □ Got Repository USL (章 □ Wesschild Spark or an Activation of the State of the St	v
	□ Generic Webhook Trigger 学 □ Pol SCM (学 □ Vacet period 学 □ WES 등 점 적으로 수행 (ペニュ 田田 사용) 学 Advanced Project Options ユョン Pipeline Definition Fipeline script from SCM SCM (学 □ Ott Repository URL (学) Haptor/lab lady com/10 - big date-dist-sub2/510F22C108 git Condensite (学) Wissochylogachonac.b://***********************************	v
	□ Generic Webhook Trigger 学 □ Pol SCM (*) □ Quiet period ** □ WES 중에 (** △田田트 사용) ** Advanced Project Options 교육 ~ Pipeline Definition Pipeline SCM (**) □ Git □ Repository URL (**) □ Hattps://do.sady.com/10-big/date-dist-sub2/510P22C10ligit □ Credential (**) □ viglooc/pi/Diparhomack.pr***** + Add ** □ 교육 ~ □ 교육 ~ □ Add Repository □ Respectively (blank for 'any) (**) □ Viderelop Add Repository Institute (blank for 'any) (**) □ Viderelop Add Repository URL (**) □ Project (blank for 'any) (**) □ Viderelop Add Struck □ Repository browser (**) □ (718) □ Additional Behaviours □ Add ** □ Additional Behaviours □ Additional Senarchian (**) □ Additional Sena	v

GitLab에서 코드 변경이 있을 경우, 자동으로 빌드하도록 설정합니다.

bigdata configure:

Jenkins	Q. 검색 (CTRL+K)	② Q Q Q duck ∨ ⊖ 로
shboard > bigdata > Configuration		
Configure	General	Enabled
③ General	설명	
Advanced Project Options	#5	
Pipeline		
	Plain text 미리보기	A
	Do not allow concurrent builds	
	Do not allow the pipeline to resume if the controller restarts	
	Gittab Connection	
		~
	Use alternative credential	
	GitLab Repository Name 🔞	
	Pipeline speed/durability override ?	
	Preserve stashes from completed builds ?	
	Throttle builds ?	
	으래된 별도 삭제 ②	
	이 벨트는 매개변수가 있습니다 ?	
	Build Triggers	
	Build after other projects are built (?)	
	Build periodically ? Build when a change is pushed to GitLab. GitLab webhook URL: http://j10c108.p.ssafyicx8888/project/foigdata ?	
	Enabled GitLab triggers	
	✓ Push Events ③ Push Events in case of branch delete ②	
	Opened Merge Request Events 3	
	Build only if new commits were pushed to Merge Request ? Accepted Merge Request Events ?	
	Closed Merge Request Events (?)	
	Rebuild open Merge Requests 🔞	
	Never	~
	Approved Merge Requests (EE-only) ✓ Comments ?	
	Comment (reger) for triggering a build ?	
	bigdata	
	Quiet period ② 발트를 점적으로 유발 (박 스크립트 사용) ③ Advanced Project Options	
	2₹ ✓	
	Pipeline	
	Definition	
	Pipeline script from SCM	~
	SCM ?	~ 3
	Repositories ®	×
	Repository URL https://lab.ssafy.com/s10-bigdata-dist-sub2/510P22C108.git	*
	Credentials ② wjssoqis@gachon.ac.kz/******	~
	+ Add *	
	28 V	
	2.0	
	Add Repository	
	Branches to build (2)	
	Branch Specifier (blank for 'any') 🔞	*
	*/develop	
	Add Branth	
	Repository browser 🔞	
	(A18)	~
	Additional Behaviours	
	Add ~	
	Contra Data (M	
	Script Path ? /bigdstat/Jenkinsfile	
	✓ Lightweight checkout ?	
	Pipeline Syntax	
	시청 Apply	

자동으로 Docker Image를 빌드하고 배포하기 위해 Web Trigger를 설정하였습니다. 트리거에는 Push, Merge Event, Comment를 지정하였습니다.

Git Lab과의 연결은 아이디와 비밀번호를 입력해 인증하였습니다. 타겟 브랜치는 develop으로 설정했습니다.



파이프라인과 연결한 Hook들은 gitLab에서 확인이 가능 합니다.

Publish over SSH:



SSH 통신을 통해 명령어를 보내고 파일 실행을 하기 위해 설정한 타겟 서버의 정보입니다.

FE - React

```
stage('.env download') {
            steps {
                withCredentials([file(credentialsId: 'env', variable: 'env')]) {
                    script {
                        sh 'cp -f ${env} ./front/'
                    }
                }
            }
        }
        stage('Build Image') {
            steps {
                script {
                    dockerImage = docker.build("frontend",
                    "-f front/Dockerfile ./front") // Dockerfile 경로 수정
            }
        }
        stage("Kill existing frontend container") {
            steps {
                script {
                    def existingContainers =
                    sh(script: "docker ps -aq -f name=frontend",
                         returnStdout: true).trim()
                    if (existingContainers) {
                        sh "docker rm -f ${existingContainers}"
                    }
                }
            }
        }
        stage("Run frontend container") {
            steps {
                script {
                    sh 'docker rm -f frontend' // Remove container if exists
                    sh "docker run -d --restart=always -p 3000:3000 \
                    --name frontend frontend"
                }
            }
        }
        stage('Clean Common Image') {
            steps {
                script {
                    sh "docker system prune -af"
            }
        }
   }
}
```

```
# 기존 멀티 스테이지 빌드의 builder 단계 제거 (필요에 따라)
FROM node:lts-alpine
```

```
WORKDIR /front

# package.json과 package-lock.json을 먼저 복사
COPY package.json .
COPY package-lock.json .

# 프로젝트 의존성 설치
RUN rm -rf node_modules
RUN npm install

# 프로젝트 소스 복사
COPY . .

EXPOSE 3000/tcp

# 애플리케이션 실행
CMD ["npm", "run", "dev"]
```

BE - Spring

```
pipeline {
    environment {
        dockerImage = ''
   }
    agent any
    stages {
        stage('Checkout') {
           steps {
                script {
                    // 코드 체크아웃
                    checkout scm
                }
           }
        }
        stage('keystore delete') {
            steps {
                script{
                    sh 'rm -f ./back/src/main/resources/keystore.p12'
                }
           }
       }
        stage('keystore download') {
                withCredentials([file(credentialsId: 'keystore',
                variable: 'keystoreFile')]) {
                    script {
                        sh 'cp -f $keystoreFile
                        ./back/src/main/resources/keystore.p12'
                    }
```

```
}
}
stage('secret.yml delete') {
    steps {
        script{
            sh 'rm -f ./back/src/main/resources/application.yml'
   }
}
stage('secret.yml download') {
    steps {
        withCredentials([file(credentialsId: 'application2',
        variable: 'applicationFile')]) {
            script {
                sh 'cp -f $applicationFile
                ./back/src/main/resources/application.yml'
            }
        }
   }
}
stage('Build Common-Server .JAR') {
    steps {
        script {
            // 변경된 브랜치와 머지 대상 브랜치 출력
            // 현재 빌드 중인 브랜치명 확인
            def currentBranch = env.BRANCH_NAME
            echo "Merge Target Branch: \${CHANGE_TARGET}"
            echo "Current Branch: \${currentBranch}"
            echo "back end 입니다."
            // gradlew 빌드
            //rwx
            sh 'chmod +x ./back/gradlew'
            sh '''
                cd ./back
                ./gradlew clean build
        }
   }
}
stage('Build Common Image') {
    steps {
        script {
            dockerImage = docker.build("backend",
             "-f back/Dockerfile ./back")
        }
   }
}
```

```
stage("Kill exist container") {
            steps {
                script {
                    def existingContainers
                    = sh(script: "docker ps -aq -f name=backend",
                     returnStdout: true).trim()
                    if (existingContainers) {
                        sh "docker rm -f ${existingContainers}"
//
                       sh "docker system prune -af"
                }
            }
        }
        stage("run container") {
            steps {
                script {
                    sh 'docker rm -f backend'
                    sh "docker run -d --restart=always -p 8080:8080
                    --name backend backend"
                }
            }
        }
        stage('Clean Common Image') {
            steps {
                script {
                    sh "docker system prune -af"
            }
        }
   }
}
```

```
FROM openjdk:17.0.2-oracle

ARG JAR_FILE=build/libs/*.jar

COPY ${JAR_FILE} ./app.jar

EXPOSE 8080

ENV TZ=Asia/Seoul

ENTRYPOINT ["java","-jar","./app.jar"]
```

BE - RabbitMQ

RabbitMQ Image file 을 Docker에 설치/실행하기 위해 아래의 명령어를 실행한다.

기본적으로 RabbitMQ는 AMQP를 사용하나, STOMP 사용을 위해 포트를 추가로 열어준다.

docker run -d --name rabbitmq -p 5672:5672 -p 15672:15672 -p 61613:61613 --restart=unless-stopped rabbitmq:3-management

- 5672 (AMQP)
- 15672 (RabbitMQ)
- 61613 (STOMP)

Docker의 Exec에서 STOMP 사용을 위해 Plugin을 설치한다.

rabbitmq-plugins enable rabbitmq_stomp

이후 RabbitMQ 설정/사용을 위해 Exchange와 Queue를 등록하고 Routing-key를 Binding한다.

(Back 폴더 내 파일 참조)

- · StompConfig
- RabbitConfig
- RabbitProperties
- RabbitStompController
- ChatService

AI / Big Data

```
pipeline {
   environment {
       repository = "angryj/bigdata"
       DOCKERHUB_CREDENTIALS = credentials('bigduck')
       // jenkins에 등록해 놓은 docker hub credentials 이름
       dockerImage = ''
   }
   agent any
   stages {
       stage('Checkout') {
           steps {
                script {
                    // 코드 체크아웃
                    checkout scm
                }
           }
       }
       stage('Build Image') {
           steps {
                    dockerImage = docker.build("${repository}:bigdata_${BUILD_NUMBER})
                     "-f bigdata/Dockerfile ./bigdata") // Dockerfile 경로 수정
                }
           }
       }
       stage('DockerHub Login') {
           steps {
                script {
                    sh "echo \ [DOCKERHUB_CREDENTIALS_PSW] | docker login -u \
                    ${DOCKERHUB_CREDENTIALS_USR} --password-stdin"
                }
           }
       stage('Push User Image to DockerHub') {
           steps {
                script {
                    sh "docker push \${repository}:bigdata_\${BUILD_NUMBER}"
                }
```

```
}
        stage("Deploy") {
            steps {
                sshPublisher(
                    continueOnError: false,
                    failOnError: true,
                    publishers: [
                        sshPublisherDesc(
                            configName: "bigdata",
                            verbose: true,
                            transfers: [
                                 sshTransfer(execCommand: "sudo docker rm -f bigdata")
                                 sshTransfer(execCommand: "sudo docker pull "
                                + repository +":bigdata_${BUILD_NUMBER}"),
                                sshTransfer(execCommand: "sudo docker run -d
                                 -p 8881:8881 --name bigdata "
                                + repository +":bigdata_${BUILD_NUMBER}"),
                                sshTransfer(execCommand: "docker system prune -af")
                            ]
                        )
                    ]
                )
            }
        }
        stage('Clean Common Image') {
            steps {
                script {
                    sh "docker system prune -af"
                }
            }
        }
    }
}
```

```
# apt init
ENV LANG=C.UTF-8
ENV TZ=Asia/Seoul
ENV DEBIAN_FRONTEND=noninteractive
RUN apt-get update && \
    apt-get -y upgrade && \
    apt-get install -y --no-install-recommends tzdata \
        g++ git curl vim wget ssh unzip sudo

RUN apt-get update && \
    apt-get -y upgrade

# Install Java
```

```
RUN wget http://www.mirbsd.org/~tg/Debs/sources.txt/wtf-bookworm.sources && \
    sudo mkdir -p /etc/apt/sources.list.d && \
    sudo mv wtf-bookworm.sources /etc/apt/sources.list.d/ && \
    sudo apt update
RUN sudo apt install -y openjdk-8-jdk
# workspace
WORKDIR /app/
COPY ./app /app
SHELL ["/bin/bash", "-c"]
# install spark
RUN wget https://dlcdn.apache.org/spark/spark-3.5.1/spark-3.5.1-bin-hadoop3.tgz && \
    tar xvf spark-3.5.1-bin-hadoop3.tgz && \
    sudo mv spark-3.5.1-bin-hadoop3 /opt/spark
ENV JAVA_HOME=/usr/lib/jvm/java-8-openjdk-amd64
ENV SPARK_HOME=/opt/spark
ENV PATH=$PATH:$SPARK_HOME/bin:$SPARK_HOME/sbin
RUN pip install -r requirements.txt
# Execute FastAPI
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "8881", "--reload"]
```

NGINX 리버스 프록시 설정

certbot으로 Let's Encrypt에서 SSL 인증서를 발급 받은 후, 리버스 프록시 설정을 위해 /etc/nginx/sites-enabled/default을 아래와 같이 수정한다.

```
##

# You should look at the following URL's in order to grasp a solid understanding

# of Nginx configuration files in order to fully unleash the power of Nginx.

# https://www.nginx.com/resources/wiki/start/

# https://www.nginx.com/resources/wiki/start/topics/tutorials/config_pitfalls/

# https://wiki.debian.org/Nginx/DirectoryStructure

#

# In most cases, administrators will remove this file from sites-enabled/ and

# leave it as reference inside of sites-available where it will continue to be

# updated by the nginx packaging team.

#

# This file will automatically load configuration files provided by other

# applications, such as Drupal or Wordpress. These applications will be made

# available underneath a path with that package name, such as /drupal8.

#

# Please see /usr/share/doc/nginx-doc/examples/ for more detailed examples.

##

# Default server configuration

# server {

listen 80 default_server;
```

```
listen [::]:80 default_server;
    server_name j10c108.p.ssafy.io;
    # redirect all http request to https
    return 301 https://$server_name$request_uri;
}
server {
   # listen 80 default_server;
    # listen [::]:80 default_server;
    # SSL configuration
    # listen 443 ssl default_server;
    # listen [::]:443 ssl default_server;
    # Note: You should disable gzip for SSL traffic.
    # See: https://bugs.debian.org/773332
    # Read up on ssl_ciphers to ensure a secure configuration.
    # See: https://bugs.debian.org/765782
    # Self signed certs generated by the ssl-cert package
    # Don't use them in a production server!
    # include snippets/snakeoil.conf;
    root /var/www/html;
    # Add index.php to the list if you are using PHP
    index index.html index.htm index.nginx-debian.html;
    server_name j10c108.p.ssafy.io;
    # location / {
        # First attempt to serve request as file, then
        # as directory, then fall back to displaying a 404.
        # try_files $uri $uri/ =404;
    # }
    # reverse proxy
    # frontend
    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Host $host;
    }
    # backend
    location /api {
        proxy_pass http://localhost:8080;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
```

```
proxy_set_header Host $host;
    # CORS
    add_header 'Access-Control-Allow-Origin' '*';
    add_header 'Access-Control-Allow-Methods' '*';
    add_header 'Access-Control-Allow-Headers' '*';
    add_header 'Access-Control-Allow-Credentials' 'true';
}
# websocket
location /wss {
    proxy_pass http://localhost:8080;
    proxy_http_version 1.1;
    proxy_set_header Upgrade $http_upgrade;
    proxy_set_header Host $host;
}
    # jenkins
    location /jenkins {
            proxy_pass http://j10c108.p.ssafy.io:8888;
            proxy_http_version 1.1;
            proxy_set_header Upgrade $http_upgrade;
            proxy_set_header Host $host;
            # CORS
            add header 'Access-Control-Allow-Origin' '*';
            add_header 'Access-Control-Allow-Methods' '*';
            add_header 'Access-Control-Allow-Headers' '*';
            add_header 'Access-Control-Allow-Credentials' 'true';
    }
# pass PHP scripts to FastCGI server
#location ~ \.php$ {
   include snippets/fastcgi-php.conf;
#
# # With php-fpm (or other unix sockets):
   fastcgi_pass unix:/var/run/php/php7.4-fpm.sock;
# # With php-cgi (or other tcp sockets):
   fastcgi_pass 127.0.0.1:9000;
#}
# deny access to .htaccess files, if Apache's document root
# concurs with nginx's one
#location ~ /\.ht {
    deny all;
#}
listen [::]:443 ssl ipv6only=on; # managed by Certbot
listen 443 ssl; # managed by Certbot
ssl_certificate /etc/letsencrypt/live/j10c108.p.ssafy.io/fullchain.pem; # managed
ssl_certificate_key /etc/letsencrypt/live/j10c108.p.ssafy.io/privkey.pem; # manag
include /etc/letsencrypt/options-ssl-nginx.conf; # managed by Certbot
ssl_dhparam /etc/letsencrypt/ssl-dhparams.pem; # managed by Certbot
```

```
}
# Virtual Host configuration for example.com
# You can move that to a different file under sites-available/ and symlink that
# to sites-enabled/ to enable it.
#server {
# listen 80;
 listen [::]:80;
#
#
# server_name example.com;
# root /var/www/example.com;
# index index.html;
#
# location / {
#
       try_files $uri $uri/ =404;
# }
#}
```

4) DB 접속 정보

Maria DB	User Info	User Name	S10P22C108
		Password	iF6wrgXGd6
	URL	mysql://stg-yswa-kr-practice-db-master.mariadb.database.azure.com:3306/S10P22C108?serverTimezone=UTC&useUnicode=true&characterEncoding=utf8	
Mongo DB	User Info	User Name	S10P22C108
		Password	1znS65MER5
	URI	mongodb+srv://S10P22C108:1znS65MER5@ssafy.ngivl.mongodb.net /S10P22C108?authSource=admin	

2. 카카오 소셜 로그인을 위한 설정 정보

앱 설정 > 플랫폼

Web

Site Domain	http://j10c108.p.ssafy.io:3000
	http://j10c108.p.ssafy.io:8080
	https://j10c108.p.ssafy.io:3000
	https://j10c108.p.ssafy.io:8080
	http://j10c108.p.ssafy.io
	https://j10c108.p.ssafy.io

제품 설정 > 카카오 로그인

카카오 로그인 활성화

Redirect URI

Redirect URI	https://j10c108.p.ssafy.io/login/oauth2/code/kakao
	https://j10c108.p.ssafy.io:8080/api/users/login
	https://j10c108.p.ssafy.io/api/users/login

제품 설정 > 카카오 로그인 > 동의 항목

개인정보

항목 이름	상태
닉네임	필수 동의
프로필 사진	필수 동의
카카오계정(이메일)	필수 동의

3. DB 덤프 파일 최신본

: 'exec' 폴더에 업로드되어 있음.

4. 시연 시나리오 및 기능 설명

: 시연 순서에 따른 site 화면별, 실행별(클릭 위치 등) 상세 설명

1. 파티 생성 및 참가

1-1. 파티 생성 : 화면 하단의 [만들기] 버튼 클릭 → 새로운 페이지에서 파티명 입력 후 [파티열기] 버튼 클릭

1-2. 파티 참가: 화면 중앙에 6자리 참여 코드 입력 후 [참여하기] 버튼 클릭

1-3. 파티 참여 현황 확인 : 화면 상단 [새로고침] 아이콘 클릭

2. 파티 시작 및 삭제

2-1. 파티 주최자

2-1-1. 시작 : 화면 하단의 종료시간을 입력하여 마니또 게임 시작 가능

2-1-2. 삭제: 화면 하단 [파티삭제] 버튼을 클릭하여 파티 삭제 가능

2-2. 파티 참가자

2-2-1. 시작: 새로고침을 통해 파티가 시작했음을 확인

2-2-2. 탈퇴 : 화면 하단 [파티떠나기] 버튼을 클릭 → 안내 확인 후 [OK] 버튼 클릭

2-3. 파티 주최자&파티 참가자 : 화면 상단 [참여코드 공유] 버튼을 클릭하여 참여코드를 클립보드에 복사하고 다른 메신저를 통해 공유 가능

2-3. 힌트 : 파티 시작 직후 힌트 입력 화면으로 이동 → 답변 입력 후 화면 하단 [입력 완료] 버튼 클릭

3. 미션

하단 네비게이션 바 첫 번째 버튼 클릭

3-1. 미션 수행

3-1-1. 미션 인증 사진 업로드 : 화면 중앙 [+] 아이콘 클릭 → 갤러리에서 사진 선택 후 [완료] 버튼 클릭

3-1-2. 미션 인증 사진 제출 : 화면 하단 [제출] 버튼 클릭

3-1-3. 미션 새로고침 : 화면 상단 [새로고침] 아이콘 클릭 → 안내 확인 후 [OK] 버튼 클릭

3-2. 미션 확인 : '미션 수행' 화면에서 우측으로 슬라이드 → 인증 사진 확인 후 화면 하단 [네] 또는 [아니오] 버튼 클릭 → [채팅방으로 이동] 버튼 클릭

4. 채팅: 하단 네비게이션 바 두 번째 버튼 클릭

4-1. 메세지 전송

4-1-1. 채팅 메세지 전송 : 화면 하단 입력 창에 메세지 작성 후 우측 [전송] 아이콘 클릭

4-1-2. 사진 메세지 전송 : 화면 하단의 [+] 아이콘 클릭 → 갤러리에서 사진 선택 후 [완료] 버튼 클릭 → 화면 하단 [전송] 아이콘 클릭

4-2. 채팅방

4-2-1. 그룹 채팅방 : 파티 내 참여 인원 전부와 실명 채팅 가능

4-2-2. 마니또와 대화 : 익명의 마니또와 채팅 가능 4-2-3. 마니띠와 대화 : 실명의 마니띠와 채팅 가능

5. 힌트: 하단 네비게이션 바 세 번째 버튼 클릭

5-1. 파티 진행 중 : 사용자는 자신의 마니또가 수행하지 않은 개수만큼 힌트 확인 가능

5-2. 파티 종료 후 : 사용자는 자신의 마니또에 대한 힌트 전부 확인 가능

6. 투표 : 하단 네비게이션 바 네 번째 버튼 클릭

6-1. 파티 진행 중 : 투표 시간 안내 문구 확인

6-2. 파티 종료 24시간 전 ~ 파티 종료 직전 : 사용자 본인 제외 참가자 목록 확인 → 마니또 선택 → 하단 [투표하기] 버튼 클릭 → 투표 완료 확인 후 [그룹 채팅으로 돌아가기] 버튼 클릭 → 채팅 목록 화면으로 이동

7. 결과

7-1. 미션 결과: 파티 종료 후 하단 네비게이션 바 첫 번째 버튼 클릭

7-1-1. 내가 수행한 미션 : 화면 중앙 날짜를 클릭하여 해당 일에 제출한 미션 인증 사진 확인

7-1-2. 마니또가 수행한 미션 : '내가 수행한 미션' 화면에서 우측으로 슬라이드 → 화면 중앙 날짜를 클릭하여 해당 일에 제출된 미션 인 증 사진 확인

7-2. 분석 결과 : 파티 종료 후 하단 네비게이션 바 네 번째 버튼 클릭

7-2-1. 그룹 마니또 페어 확인: 그룹 내 마니또 페어와 우호도 점수, 마니또 투표 결과 확인 가능

7-2-2. 마니또와 관계 분석 : '그룹 마니또 페어 확인' 화면에서 우측으로 슬라이드 → 우호도 점수, WordCount, 긍정어vs부정어 사용 비율 등 채팅 데이터 기반 분석결과 확인, 진행기간 동안 활동 내역 요약 확인

7-2-3. 마니띠와 관계 분석 : '그룹 마니또와 관계 분석' 화면에서 우측으로 슬라이드 → 우호도 점수, WordCount, 긍정어vs부정어 사용 비율 등 채팅 데이터 기반 분석 결과 확인, 진행 기간 동안 활동 내역 요약 확인

8. 파티 나가기

8-1. 파티 주최자 : 화면 우측 상단 [오리] 아이콘 클릭 → [파티 삭제] 버튼 클릭 → 안내 확인 후 [OK] 버튼 클릭 → '파티 생성 및 참가' 화면으로 이동

8-2. 파티 참가자

8-2-1. 파티 종료 전 : 화면 우측 상단 [오리] 아이콘 클릭 → 안내 메세지 확인

8-2-2. 파티 종료 후 : 서비스 재 입장 시 안내 메세지 확인 → '파티 생성 및 참가' 화면으로 이동