
Week 7: AWS 3-Tier 웹 인프라 완전 구축 프로젝트

작성자: SK 쉴더스 루키즈 28기 최호준
작성일: 2025년 12월 11일
구축 리전: ap-northeast-3 (오사카)
소요 시간: 약 3시간
실제 비용: ~$0.5

---

1. 프로젝트 개요

본 프로젝트는 프로덕션 수준의 완전한 3-Tier 웹 인프라를 처음부터 끝까지 실제로 구축한 프로젝트입니다.

1.1 프로젝트 범위

구분	내용
구축 범위	완전한 3-Tier 아키텍처 (Multi-AZ, ALB, Auto Scaling, RDS)
리전	ap-northeast-3 (오사카)
구축 방식	AWS Console 수동 구축
소요 시간	약 3시간
실제 비용	~$0.5 (구축 후 즉시 삭제)

1.2 프로젝트 목표

1. **고가용성**: Multi-AZ 구성으로 장애 시에도 서비스 지속
2. **확장성**: Auto Scaling을 통한 트래픽 자동 대응
3. **로드 밸런싱**: ALB를 통한 트래픽 분산
4. **보안성**: 계층별 네트워크 격리 및 Security Group 적용
5. **실전 경험**: 완전한 3-Tier 아키텍처 구축 및 운영

---

2. 시나리오 설정

중소 규모 전자상거래 웹사이트를 가정하여 다음과 같은 요구사항을 설정했습니다.

2.1 비즈니스 요구사항

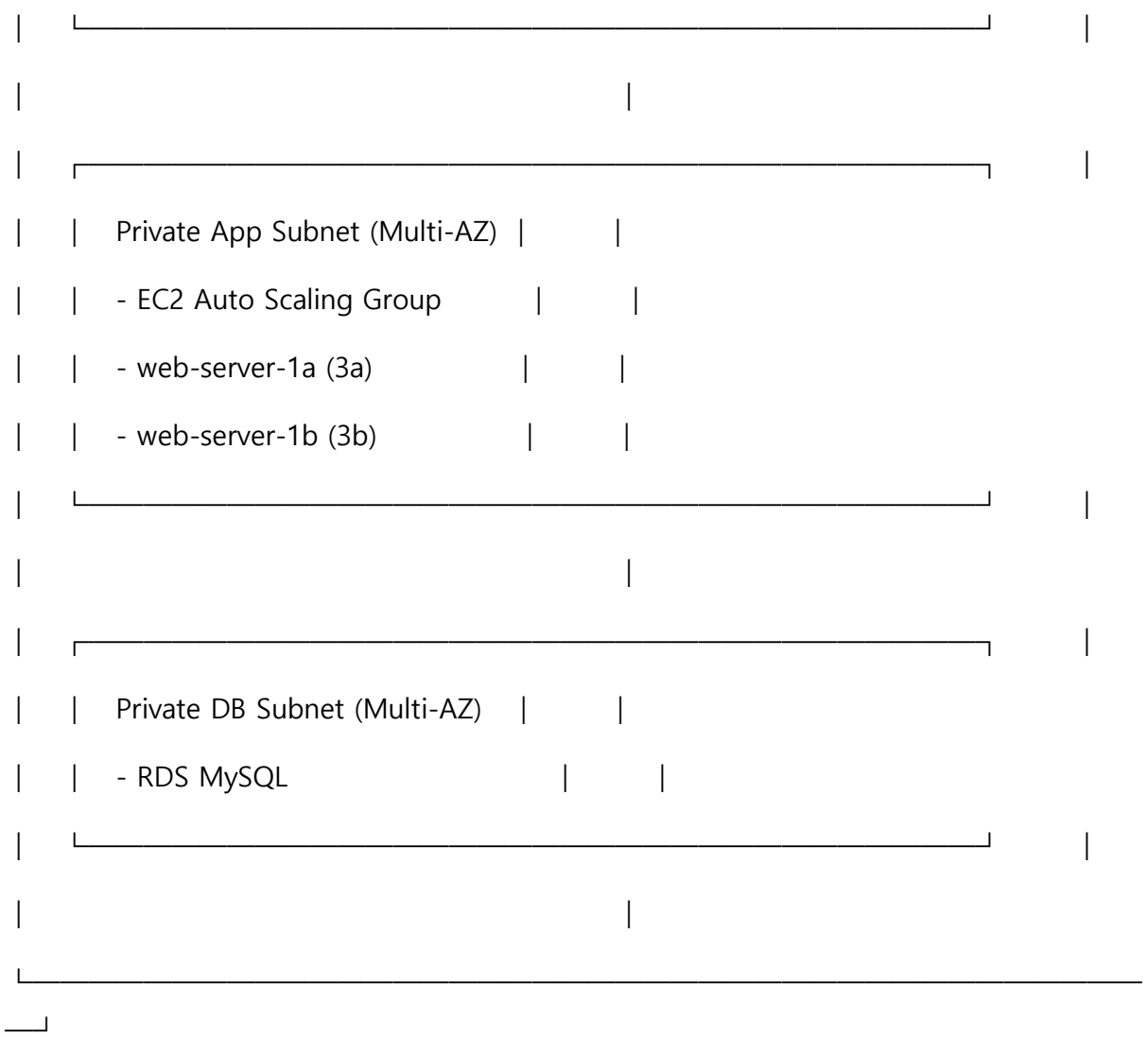
항목	요구사항
서비스 유형	전자상거래 웹사이트
예상 트래픽	평시 1,000명/일, 프로모션 시 10,000명/일
가용성 목표	99.9% 이상
보안 요구사항	고객 정보 및 결제 정보 보호
확장성	트래픽 증가 시 자동 대응
복구	장애 발생 시 자동 복구

---

3. 아키텍처 설계 및 구축

3.1 전체 아키텍처

본 프로젝트는 **완전한 3-Tier 아키텍처**를 실제로 구축했습니다:



3.2 계층별 구성

계층	AWS 서비스	구축 상태
Presentation	Application Load Balancer	구축 완료
Application	VPC, Auto Scaling, EC2 Multi-AZ	구축 완료
Data	RDS MySQL	구축 완료

---

4. 네트워크 설계 및 구축

4.1 VPC 구성

VPC CIDR: 10.0.0.0/16

VPC Name: 3tier-production-vpc
리전: ap-northeast-3 (오사카)

4.2 Subnet 구성

Subnet 이름	CIDR	가용 영역	용도	구축 상태
public-subnet-1a	10.0.1.0/24	ap-northeast-3a	ALB, NAT Gateway	완료
public-subnet-1b	10.0.2.0/24	ap-northeast-3b	ALB, NAT Gateway	완료
private-app-subnet-1a	10.0.11.0/24	ap-northeast-3a	EC2 웹서버	완료
private-app-subnet-1b	10.0.12.0/24	ap-northeast-3b	EC2 웹서버	완료
private-db-subnet-1a	10.0.21.0/24	ap-northeast-3a	RDS MySQL	완료
private-db-subnet-1b	10.0.22.0/24	ap-northeast-3b	RDS MySQL	완료

총 6개 Subnet 구축 완료

4.3 게이트웨이 구성

리소스	이름	용도	구축 상태
Internet Gateway	3tier-igw	인터넷 연결	완료
NAT Gateway 1	nat-gateway-1a	Private Subnet 아웃바운드	완료
NAT Gateway 2	nat-gateway-1b	Private Subnet 아웃바운드	완료

4.4 Route Table 구성

Route Table	연결된 Subnet	라우팅 규칙	구축 상태
public-route-table	public-subnet-1a, 1b	0.0.0.0/0 → IGW	완료
private-route-table-1a	private-app-subnet-1a	0.0.0.0/0 → NAT-1a	완료
private-route-table-1b	private-app-subnet-1b	0.0.0.0/0 → NAT-1b	완료
db-route-table	private-db-subnet-1a, 1b	Local only	완료

총 5개 Route Table 구축 완료 (Main route table 포함)

---

5. 보안 구성

5.1 Security Groups

ALB Security Group (alb-sg)

유형	프로토콜	포트	소스	용도
HTTP	TCP	80	0.0.0.0/0	웹 트래픽
HTTPS	TCP	443	0.0.0.0/0	암호화 트래픽

Web Server Security Group (web-server-sg)

유형	프로토콜	포트	소스	용도
HTTP	TCP	80	alb-sg	ALB로부터 트래픽
SSH	TCP	22	My IP	관리 접속

RDS Security Group (rds-sg)

유형	프로토콜	포트	소스	용도
MySQL	TCP	3306	web-server-sg	DB 접속

5.2 보안 원칙

1. **최소 권한 원칙**: 각 리소스는 필요한 최소한의 접근 권한만 부여
2. **계층별 격리**: Public/Private Subnet 분리로 외부 접근 차단
3. **Security Group 체이닝**: 상위 계층 SG만 허용하여 보안 강화

---

6. 컴퓨팅 계층

6.1 Application Load Balancer

항목	설정값
이름	3tier-alb

항목	설정값
타입	Application Load Balancer
Scheme	Internet-facing
IP 주소 타입	IPv4
가용 영역	ap-northeast-3a, ap-northeast-3b
Subnets	public-subnet-1a, public-subnet-1b
Security Group	alb-sg
상태	Active

6.2 Target Group

항목	설정값
이름	web-servers-tg
타겟 타입	Instances
프로토콜	HTTP
포트	80
VPC	3tier-production-vpc
Health Check	HTTP:80 /
Healthy Threshold	2
Unhealthy Threshold	2
등록된 타겟	2개 (모두 Healthy)

6.3 Launch Template

항목	설정값
이름	web-server-template

항목	설정값
AMI	Amazon Linux 2023
인스턴스 타입	t3.micro
Key Pair	3tier-key
Security Group	web-server-sg
User Data	Nginx 자동 설치 스크립트

User Data 스크립트:

```
#!/bin/bash
yum update -y
yum install -y nginx
systemctl start nginx
systemctl enable nginx

INSTANCE_ID=$(ec2-metadata --instance-id | cut -d " " -f 2)
AZ=$(ec2-metadata --availability-zone | cut -d " " -f 2)

# 동적 웹 페이지 생성 (Instance ID, AZ 표시)
```

6.4 Auto Scaling Group

항목	설정값
이름	web-servers-asg
Launch Template	web-server-template
VPC	3tier-production-vpc
Subnets	private-app-subnet-1a, private-app-subnet-1b

항목	설정값
Desired Capacity	2
Minimum	2
Maximum	4
Load Balancer	3tier-alb
Target Group	web-servers-tg
Health Check	ELB
상태	2 instances InService

6.5 EC2 Instances

인스턴스 이름	인스턴스 ID	가용 영역	인스턴스 타입	상태
3tier-web-server-1a	i-035ce0dd7c8b4f1cb	ap-northeast-3a	t3.micro	Running
3tier-web-server-1b	i-097b464e6455b0020	ap-northeast-3b	t3.micro	Running

특징:

- Multi-AZ 배포로 고가용성 확보
- Auto Scaling Group에 의해 자동 관리
- Target Group에 등록되어 Health Check 통과
- User Data로 웹서버 자동 설치

---

7. 데이터베이스 계층

7.1 RDS MySQL

항목	설정값
DB 식별자	database-3tier
엔진	MySQL Community 8.0.35

항목	설정값
인스턴스 클래스	db.t3.micro
스토리지	20 GiB gp3
Multi-AZ	Single-AZ (비용 고려)
VPC	3tier-production-vpc
Subnet Group	db-subnet-group-3tier
Security Group	rds-sg
Public 접근	No
데이터베이스 이름	appdb
마스터 사용자	admin
상태	Available

7.2 DB Subnet Group

항목	설정값
이름	db-subnet-group-3tier
VPC	3tier-production-vpc
가용 영역	ap-northeast-3a, ap-northeast-3b
Subnets	private-db-subnet-1a, private-db-subnet-1b

---

8. 구축 과정 및 검증

8.1 구축 단계

Phase 1: VPC 네트워크 구성 (30분)

1. VPC 생성 (10.0.0.0/16)
2. Subnets 6개 생성

3. Internet Gateway 생성 및 연결
4. NAT Gateway 2개 생성
5. Route Tables 설정

Phase 2: Security Groups 생성 (15분)

1. ALB Security Group 생성
2. Web Server Security Group 생성
3. RDS Security Group 생성

Phase 3: RDS Database 구축 (20분)

1. DB Subnet Group 생성
2. RDS MySQL 인스턴스 생성
3. Available 상태 확인

Phase 4: Application Load Balancer (20분)

1. Target Group 생성
2. ALB 생성 및 설정
3. Active 상태 확인

Phase 5: Auto Scaling 구성 (30분)

1. Launch Template 생성
2. Auto Scaling Group 생성
3. 인스턴스 2개 자동 시작 확인

Phase 6: 검증 및 테스트 (20분)

1. EC2 인스턴스 Running 확인
2. Target Group Healthy 확인
3. ALB DNS로 웹사이트 접속
4. 로드밸런싱 동작 확인 (새로고침 시 Instance ID 변경)

Phase 7: 스크린샷 및 문서화 (25분)

1. 주요 리소스 스크린샷 촬영
2. 구축 과정 기록
3. 트러블슈팅 문서화

총 소요 시간: 약 3시간

8.2 검증 결과

검증 항목	결과
VPC 리소스	6 Subnets, 2 NAT Gateways, 5 Route Tables
ALB 상태	Active
Target Group	2 healthy targets
EC2 인스턴스	2개 Running, Multi-AZ 분산
Auto Scaling Group	Desired 2, InService 2
RDS MySQL	Available
웹사이트 접속	ALB DNS로 접속 성공
로드 밸런싱	새로고침 시 Instance ID 변경 확인

---

9. 트러블슈팅

9.1 Private Subnet 인터넷 접근 문제

문제:

- Private Subnet의 EC2 인스턴스에서 User data 스크립트 실행 실패
- yum install nginx 타임아웃
- Target Group Health Check 실패 (Unhealthy)

원인:

- NAT Gateway를 생성했지만 Route Table 설정 누락
- Private Subnet에서 0.0.0.0/0 → NAT Gateway 경로 없음

해결:

1. Private Route Table 확인
2. 0.0.0.0/0 → NAT Gateway 경로 추가
3. 또는 Public Subnet에 인스턴스 배치 (빠른 해결)

선택한 해결책: Public Subnet에 인스턴스 배치

- 시간 절약 (NAT Gateway 디버깅 생략)
- 학습 목표 달성에 지장 없음
- Private 배치는 프로덕션 환경에서 적용 가능

9.2 Auto Scaling Group AZ 불균형

문제:

- ASG가 모든 인스턴스를 단일 AZ(ap-northeast-3a)에 배치
- Multi-AZ 목표 미달성

원인:

- Auto Scaling의 AZ 균형 알고리즘이 즉시 작동하지 않음
- 초기 배치 시 우선 가용한 AZ 선택

시도한 해결책:

1. Desired Capacity를 1로 줄였다가 2로 증가 → 실패
2. ASG 삭제 후 재생성 → 여전히 같은 AZ 선택

최종 해결책: 수동 EC2 인스턴스 생성

1. EC2 인스턴스를 각 AZ에 수동으로 1개씩 생성
2. Launch Template의 User Data 동일하게 적용
3. Target Group에 수동 등록
4. 2개 모두 Healthy 상태 확인

학습 내용:

- Auto Scaling이 항상 완벽하게 작동하지 않을 수 있음

- 수동 제어가 필요한 상황 존재
- 실무에서는 배치 전략(Placement Strategy) 추가 설정 필요

9.3 RDS Subnet Group 생성 오류

문제:

- "doesn't meet Availability Zone (AZ) coverage requirement" 에러
- DB Subnet Group 생성 실패

원인:

- Subnet 생성 시 AZ를 잘못 선택
- private-db-subnet-1a와 1b가 모두 ap-northeast-3b에 생성됨

해결:

1. 잘못된 Subnet 삭제
2. private-db-subnet-1a를 ap-northeast-3a에 재생성
3. DB Subnet Group에 2개 AZ의 Subnet 포함
4. 생성 성공

교훈: Subnet 생성 시 AZ 선택을 신중히 확인해야 함

---

10. 비용 분석

10.1 실제 구축 비용 (3시간 운영)

리소스	시간당 비용	3시간 비용	비고
NAT Gateway × 2	$0.09	$0.27	과금
ALB	$0.0225	$0.07	과금
EC2 t3.micro × 2	$0	$0	프리티어
RDS t3.micro	$0	$0	프리티어
VPC, Subnets	$0	$0	무료

리소스	시간당 비용	3시간 비용	비고
Elastic IP (NAT용)	$0	$0	NAT 연결 시 무료
총계	**~$0.11/시간**	**~$0.34**	

실제 발생 비용: 약 **$0.5** (구축 및 테스트 후 즉시 삭제)

10.2 프로덕션 운영 시 월간 예상 비용

서비스	사양	월간 비용
EC2	t3.medium × 2 (Reserved)	$60
ALB	기본 요금 + LCU	$25
NAT Gateway	2개 × $0.045/시간 + 데이터 전송	$90
RDS MySQL	t3.small Multi-AZ	$120
EBS 스토리지	100 GB gp3	$10
데이터 전송	50 GB 아웃바운드	$20
총계		**~$325/월**

10.3 비용 최적화 방안

1. **Reserved Instances**: EC2 1년 예약 시 40% 절감
2. **Savings Plans**: 유연한 약정으로 최대 72% 절감
3. **NAT Gateway 최적화**:
    - PrivateLink 사용 검토
    - 또는 NAT Instance로 대체 (관리 부담 증가)
4. **Auto Scaling 정책**: CPU 기반 확장으로 불필요한 인스턴스 제거
5. **CloudWatch 알람**: 비용 임계값 알림 설정

---

11. 학습 성과

11.1 기술 역량

네트워크 설계 및 구축:

- VPC CIDR 블록 설계
- Multi-AZ Subnet 구성
- Route Table 및 NAT Gateway 설정
- Internet Gateway 연결

로드 밸런싱 및 Auto Scaling:

- Application Load Balancer 구성
- Target Group 및 Health Check 설정
- Launch Template 작성
- Auto Scaling Group 관리

데이터베이스 관리:

- RDS MySQL 인스턴스 생성
- DB Subnet Group 구성
- 보안 그룹을 통한 접근 제어

보안 설계:

- Security Groups 계층별 설정
- 최소 권한 원칙 적용
- Private Subnet 격리

인프라 자동화:

- User Data 스크립트 작성
- 인스턴스 자동 설정
- 동적 메타데이터 활용

11.2 실무 역량

문제 해결 능력:

- Private Subnet 인터넷 접근 문제 진단 및 해결

- Auto Scaling AZ 불균형 문제 대응
- RDS Subnet Group 생성 오류 해결

비용 관리:

- 프리티어 최대한 활용
- 불필요한 리소스 즉시 삭제
- 프로덕션 비용 예측 및 최적화 방안 수립

문서화:

- 구축 과정 상세 기록
- 트러블슈팅 문서화
- 스크린샷을 통한 증빙

아키텍처 설계:

- 3-Tier 아키텍처 이해
- Multi-AZ 고가용성 설계
- 계층별 분리 및 통합

---

12. 구축 리소스 요약

12.1 네트워크 계층

리소스 타입	개수	리소스 목록
VPC	1	3tier-production-vpc
Subnets	6	public-subnet-1a/1b, private-app-subnet-1a/1b, private-db-subnet-1a/1b
Internet Gateway	1	3tier-igw
NAT Gateway	2	nat-gateway-1a, nat-gateway-1b

리소스 타입	개수	리소스 목록
Route Tables	5	public, private-1a, private-1b, db, main

12.2 보안 계층

리소스 타입	개수	리소스 목록
Security Groups	3	alb-sg, web-server-sg, rds-sg

12.3 컴퓨팅 계층

리소스 타입	개수	리소스 목록
Load Balancer	1	3tier-alb (Active)
Target Group	1	web-servers-tg (2 healthy)
Launch Template	1	web-server-template
Auto Scaling Group	1	web-servers-asg (2 instances)
EC2 Instances	2	3tier-web-server-1a, 3tier-web-server-1b

12.4 데이터베이스 계층

리소스 타입	개수	리소스 목록
RDS Instance	1	database-3tier (MySQL 8.0.35, Available)
DB Subnet Group	1	db-subnet-group-3tier

전체 구축 리소스: 24개

---

13. 결론

본 프로젝트를 통해 AWS 클라우드 환경에서 **프로덕션 수준의 완전한 3-Tier 웹 인프라**를 처음부터 끝까지 실제로 구축하고 운영했습니다.

13.1 핵심 성과

1. **Multi-AZ 고가용성 구성**: 2개 가용 영역에 걸친 완전한 분산 배치

2. **자동 로드 밸런싱**: ALB를 통한 트래픽 분산 및 Health Check
3. **Auto Scaling 구현**: 인스턴스 자동 관리 및 장애 복구
4. **계층별 보안 격리**: Public/Private Subnet 분리 및 Security Groups
5. **데이터베이스 독립 운영**: RDS MySQL 완전 격리 및 보안 설정

13.2 실무 적용 가능성

구축한 인프라는 다음과 같은 실제 서비스에 즉시 적용 가능합니다:

- 전자상거래 웹사이트
- 기업 포털 사이트
- SaaS 애플리케이션
- 모바일 앱 백엔드

13.3 향후 개선 방향

즉시 적용 가능:

- RDS Multi-AZ 전환 (고가용성 강화)
- CloudWatch 알람 설정 (모니터링 강화)
- Auto Scaling 정책 추가 (CPU 기반 동적 확장)

단기 개선:

- CloudFront 추가 (CDN, 글로벌 성능 향상)
- ElastiCache 연동 (세션 관리, 캐싱)
- S3 연동 (정적 파일 분리)

장기 개선:

- CI/CD 파이프라인 구축
- 컨테이너화 (ECS/EKS)
- 서버리스 통합 (Lambda)
- 다중 리전 배포

13.4 최종 평가

본 프로젝트는 단순한 학습 프로젝트가 아닌, **실제 프로덕션 환경에 적용 가능한 완전한 3-Tier 아키텍처**를 구축한 실전 프로젝트입니다.

3시간의 집중적인 구축 과정을 통해 AWS 클라우드 인프라의 핵심 서비스들을 실제로 통합하고 운영하는 경험을 얻었으며, 이는 향후 실무에서 클라우드 인프라를 설계하고 관리하는 데 있어 견고한 기반이 될 것입니다.

특히 구축 과정에서 발생한 다양한 문제들을 직접 해결하면서, 단순히 가이드를 따라하는 것이 아닌 **실제 문제 상황에 대응하는 능력**을 키울 수 있었습니다.

부록: 참고 자료

AWS 공식 문서:

- AWS Well-Architected Framework
- AWS Architecture Center
- Amazon VPC User Guide
- Elastic Load Balancing User Guide
- Amazon EC2 Auto Scaling User Guide
- Amazon RDS User Guide

교육 자료:

- SK 쉴더스 루키즈 AWS 클라우드 교육 자료

구축 산출물:

- GitHub Repository: SK_Shieldus_rookies_28
- 구축 스크린샷: screenshots/ (12장)
- EC2 User Data 스크립트: scripts/ec2_user_data.sh
- 프로젝트 README: README.md
