Step6: 運用・監視・バックアップ・セキュリティ強化

1. 監視（ヘルスチェック）

1-1) AppServer: HTTPヘルスチェック

```
# AppServer
sudo tee /usr/local/bin/healthcheck_web.sh <<'SH'
#!/usr/bin/env bash
URL="http://127.0.0.1"
ts="$(date '+%Y-%m-%d %H:%M:%S')"
if curl -fsS -m 5 "$URL" >/dev/null; then
  echo "$ts OK"
else
  echo "$ts NG"
  systemctl is-active nginx || sudo systemctl restart nginx
  systemctl is-active php-fpm || sudo systemctl restart php-fpm
fi
SH
sudo chmod +x /usr/local/bin/healthcheck_web.sh

# cronで5分ごと
echo "*/5 * * * * root /usr/local/bin/healthcheck_web.sh >> /var/log/healthcheck_web.log 2>&1" | sudo tee /etc/cron.d/health_web
```

1. 監視（ヘルスチェック）

1-2) DBServer: 3306ポート監視

```
# DBServer
sudo tee /usr/local/bin/healthcheck_db.sh <<'SH'
#!/usr/bin/env bash
HOST="127.0.0.1"
PORT=3306
ts="$(date '+%Y-%m-%d %H:%M:%S')"
if (command -v nc >/dev/null && nc -z -w3 $HOST $PORT) || (command -v telnet >/dev/null && echo quit | telnet $HOST $PORT >/dev/null 2>&1); then
  echo "$ts OK"
else
  echo "$ts NG"
  systemctl is-active mariadb || sudo systemctl restart mariadb
fi
SH
sudo chmod +x /usr/local/bin/healthcheck_db.sh
echo "*/5 * * * * root /usr/local/bin/healthcheck_db.sh >> /var/log/healthcheck_db.log 2>&1" | sudo tee /etc/cron.d/health_db
```

2. バックアップ（DBServerで毎日スナップショット取得）

```
# DBServer
sudo mkdir -p /var/backups/mysql
sudo tee /usr/local/bin/backup_mysql.sh <<'SH'
#!/usr/bin/env bash
set -euo pipefail
BACKUP_DIR="/var/backups/mysql"
DB="appdb"
USER="root"
PASS=""   # mysql_config_editor の利用を推奨（下に記載）
DATE="$(date +%F_%H%M%S)"
mkdir -p "$BACKUP_DIR"
mysqldump -u "$USER" ${PASS:+-p"$PASS"} --single-transaction --routines --triggers "$DB" ￥
 | gzip > "$BACKUP_DIR/${DB}_${DATE}.sql.gz"
# 14日以上を削除
find "$BACKUP_DIR" -type f -name "${DB}_*.sql.gz" -mtime +14 -delete
SH
sudo chmod +x /usr/local/bin/backup_mysql.sh

# 深夜3:15に実行
echo "15 3 * * * root /usr/local/bin/backup_mysql.sh >> /var/log/backup_mysql.log 2>&1" | sudo tee /etc/cron.d/backup_mysql
```

2. バックアップ（DBServerで毎日スナップショット取得）

パスワードの安全な扱い（推奨）

DBServerで root 用に資格情報を安全保存

mysql_config_editor set --login-path=localroot --user=root --password

スクリプト内を次へ変更

mysqldump --login-path=localroot --single-transaction --routines --triggers "$DB"

3. ログ管理（読みやすく・溜めすぎない）

Nginx / PHP-FPM / MariaDB は logrotate が同梱されています（/etc/logrotate.d/ を確認）。
アプリ独自ログを出すなら logrotate 設定を追加：

```
# AppServer例：/var/www/app/storage/logs/app.log を日次ローテート
sudo tee /etc/logrotate.d/app_log <<'CONF'
/var/www/app/storage/logs/*.log {
    daily
    rotate 14
    compress
    missingok
    notifempty
    create 0640 nginx nginx
    sharedscripts
    postrotate
        /bin/systemctl reload nginx >/dev/null 2>&1 || true
    endscript
}
CONF
```

4. セキュリティ強化

4-1) SSHの更なる強化（両VM）

すでに鍵認証＆PasswordAuthentication no 済み想定

```
sudo sed -i 's/^#¥?ClientAliveInterval.*/ClientAliveInterval 300/' /etc/ssh/sshd_config
sudo sed -i 's/^#¥?ClientAliveCountMax.*/ClientAliveCountMax 2/' /etc/ssh/sshd_config
sudo systemctl restart sshd
```

4. セキュリティ強化

4-2) firewalld の最小化

AppServer: http, ssh のみ

```
sudo firewall-cmd --permanent --set-default-zone=public
sudo firewall-cmd --permanent --remove-service=https || true
sudo firewall-cmd --permanent --add-service=http
sudo firewall-cmd --permanent --add-service=ssh
sudo firewall-cmd --reload
```

4. セキュリティ強化

4-2) firewalld の最小化

DBServer: ssh と mysql のみ（Host-Only 想定だが最小権限を維持）

```
sudo firewall-cmd --permanent --add-service=ssh
sudo firewall-cmd --permanent --add-service=mysql
sudo firewall-cmd --reload
```

4. セキュリティ強化

4-3) SELinux（有効のまま運用）

- 既に設定済み：httpd_can_network_connect on（App→DB接続）
- 追加で AVC 出たら audit を基に対処（学習用は permissive 化しないのが◎）

4. セキュリティ強化

4-4) fail2ban（SSHブルート対策）

- 両VMとも（Rocky 9 では epel-release が必要）

```
sudo dnf install -y epel-release
sudo dnf install -y fail2ban
sudo systemctl enable --now fail2ban
```

- 最低限のJail（ssh）

```
sudo tee /etc/fail2ban/jail.d/ssh.local <<'JAIL'
[sshd]
enabled = true
bantime = 1h
findtime = 10m
maxretry = 5
JAIL
sudo systemctl restart fail2ban
sudo fail2ban-client status sshd
```