GUÍA DE IMPLEMENTACIÓN RUNPOD - ECOSISTEMA MENTALIA

CONFIGURACIÓN INICIAL DE RUNPOD

1. Requisitos de Hardware Recomendados

```
 GPU Pod Configuration:
- GPU: NVIDIA RTX 4090 o superior (para modelos de IA)
- vCPU: 16+ cores
- RAM: 64GB+
- Storage: 500GB+ NVMe SSD
- Network: 1Gbps+

Configuración Mínima:
- GPU: NVIDIA RTX 3080
- vCPU: 8 cores
- RAM: 32GB
- Storage: 200GB SSD
- Network: 500Mbps
```

2. Script de Configuración Automática

```
 #!/bin/bash
# install-mentalia-ecosystem.sh

set -e

echo "🚀 Iniciando instalación del Ecosistema MENTALIA en RunPod..."

# Variables de configuración
DOMAIN=${1:-"mentalia.runpod.io"}
EMAIL=${2:-"admin@mentalia.ai"}
OPENAI_API_KEY=${3:-""}

if [ -z "$OPENAI_API_KEY" ]; then
    echo "❌ Error: OPENAI_API_KEY es requerido"
    echo "Uso: ./install-mentalia-ecosystem.sh [DOMAIN] [EMAIL] [OPENAI_API_KEY]"
    exit 1
fi

# Actualizar sistema
```

```
echo "   Actualizando sistema..."
apt-get update && apt-get upgrade -y

# Instalar dependencias
echo "   Instalando dependencias..."
apt-get install -y \
  curl \
  wget \
  git \
  unzip \
  nginx \
  certbot \
  python3-certbot-nginx \
  htop \
  tree \
  jq

# Instalar Docker
echo "   Instalando Docker..."
curl -fsSL https://get.docker.com -o get-docker.sh
sh get-docker.sh
systemctl start docker
systemctl enable docker

# Instalar Docker Compose
echo "   Instalando Docker Compose..."
curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$
(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
chmod +x /usr/local/bin/docker-compose

# Instalar Node.js
echo "   Instalando Node.js..."
curl -fsSL https://deb.nodesource.com/setup_20.x | bash -
apt-get install -y nodejs

# Crear directorio del proyecto
echo "   Configurando estructura de proyecto..."
mkdir -p /opt/mentalia
cd /opt/mentalia

# Clonar repositorio (reemplazar con repo real)
echo "   Descargando código fuente..."
git clone https://github.com/mentalia/ecosystem.git .

# Configurar variables de entorno
echo "   Configurando variables de entorno..."
cat > .env << EOF
# Configuración de producción
NODE_ENV=production
ENVIRONMENT=production
```

```
# Dominio y SSL
PUBLIC_DOMAIN=${DOMAIN}
ADMIN_EMAIL=${EMAIL}

# Base de datos
POSTGRES_PASSWORD=$(openssl rand -base64 32)
POSTGRES_USER=mentalia
POSTGRES_DB=mentalia_main

# Redis
REDIS_PASSWORD=$(openssl rand -base64 32)

# API Gateway
KONG_DB_PASSWORD=$(openssl rand -base64 32)

# Autenticación
JWT_SECRET=$(openssl rand -base64 64)
SESSION_SECRET=$(openssl rand -base64 32)

# APIs de IA
OPENAI_API_KEY=${OPENAI_API_KEY}
ANTHROPIC_API_KEY=${ANTHROPIC_API_KEY:-""}

# Monitoreo
GRAFANA_PASSWORD=$(openssl rand -base64 16)

# RunPod específico
RUNPOD_POD_ID=${RUNPOD_POD_ID:-""}
RUNPOD_PUBLIC_IP=${RUNPOD_PUBLIC_IP:-""}

# Configuración de correo
SMTP_HOST=${SMTP_HOST:-"smtp.gmail.com"}
SMTP_PORT=${SMTP_PORT:-"587"}
SMTP_USER=${SMTP_USER:-""}
SMTP_PASS=${SMTP_PASS:-""}

# Configuración de almacenamiento
STORAGE_TYPE=local
STORAGE_PATH=/opt/mentalia/storage

# Configuración de logs
LOG_LEVEL=info
LOG_FORMAT=json
EOF

# Configurar Nginx
echo "   Configurando Nginx..."
cat > /etc/nginx/sites-available/mentalia << EOF
server {
    listen 80;
    server_name ${DOMAIN};
    return 301 https://\$server_name\$request_uri;
```

```
}

server {
    listen 443 ssl http2;
    server_name ${DOMAIN};

    # SSL Configuration
    ssl_certificate /etc/letsencrypt/live/${DOMAIN}/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/${DOMAIN}/privkey.pem;

    # Security headers
    add_header X-Frame-Options DENY;
    add_header X-Content-Type-Options nosniff;
    add_header X-XSS-Protection "1; mode=block";
    add_header Strict-Transport-Security "max-age=31536000; includeSubDomains" always;

    # Frontend
    location / {
        proxy_pass http://localhost:3000;
        proxy_http_version 1.1;
        proxy_set_header Upgrade \$http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;
        proxy_cache_bypass \$http_upgrade;
    }

    # API Gateway
    location /api/ {
        proxy_pass http://localhost:8000/;
        proxy_http_version 1.1;
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;
    }

    # WebSocket support
    location /ws/ {
        proxy_pass http://localhost:8000/ws/;
        proxy_http_version 1.1;
        proxy_set_header Upgrade \$http_upgrade;
        proxy_set_header Connection "upgrade";
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;
    }

    # Monitoring (protegido)
```

```
    location /monitoring/ {
        auth_basic "Área Restringida";
        auth_basic_user_file /etc/nginx/.htpasswd;
        proxy_pass http://localhost:3001/;
        proxy_set_header Host \$host;
        proxy_set_header X-Real-IP \$remote_addr;
        proxy_set_header X-Forwarded-For \$proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto \$scheme;
    }
}
EOF

# Habilitar sitio
ln -sf /etc/nginx/sites-available/mentalia /etc/nginx/sites-enabled/
rm -f /etc/nginx/sites-enabled/default

# Configurar SSL
echo "   Configurando SSL..."
if [ "${DOMAIN}" != "mentalia.runpod.io" ]; then
    certbot --nginx -d ${DOMAIN} --non-interactive --agree-tos --email ${EMAIL}
else
    echo "⚠ Usando dominio por defecto, SSL manual requerido"
fi

# Crear usuario para monitoreo
echo "   Configurando autenticación para monitoreo..."
htpasswd -cb /etc/nginx/.htpasswd admin $(openssl rand -base64 12)

# Construir imágenes Docker
echo "   Construyendo imágenes Docker..."
docker-compose build

# Inicializar base de datos
echo "   Inicializando base de datos..."
docker-compose up -d postgres-main redis
sleep 10

# Ejecutar migraciones
echo "   Ejecutando migraciones..."
docker-compose run --rm backend npm run migrate

# Iniciar todos los servicios
echo "   Iniciando servicios..."
docker-compose up -d

# Configurar scripts de mantenimiento
echo "   Configurando scripts de mantenimiento..."

# Script de health check
cat > /opt/mentalia/health-check.sh << 'EOF'
#!/bin/bash
```

```
echo "=== MENTALIA ECOSYSTEM HEALTH CHECK ==="
echo "Timestamp: $(date)"
echo

# Verificar servicios Docker
echo "   Docker Services:"
docker-compose ps

echo
echo "   System Resources:"
echo "CPU: $(top -bn1 | grep "Cpu(s)" | awk '{print $2}' | awk -F'%' '{print $1}')%"
echo "Memory: $(free -m | awk 'NR==2{printf "%.1f%%", $3*100/$2 }')%"
echo "Disk: $(df -h / | awk 'NR==2 {print $5}')"

echo
echo "   Service Health:"
curl -s -o /dev/null -w "Frontend: %{http_code}\n" http://localhost:3000/health
curl -s -o /dev/null -w "API Gateway: %{http_code}\n" http://localhost:8000/health

echo
echo "   Database:"
docker-compose exec -T postgres-main psql -U mentalia -d mentalia_main -c "SELECT
'Connected' as status;" 2>/dev/null || echo "Database: ERROR"

echo "=== END HEALTH CHECK ==="
EOF

chmod +x /opt/mentalia/health-check.sh

# Script de backup
cat > /opt/mentalia/backup.sh << 'EOF'
#!/bin/bash
BACKUP_DIR="/opt/mentalia/backups"
DATE=$(date +%Y%m%d_%H%M%S)

mkdir -p $BACKUP_DIR

echo "   Backing up database..."
docker-compose exec -T postgres-main pg_dumpall -U mentalia > $BACKUP_DIR/
db_backup_$DATE.sql

echo "   Backing up configuration..."
tar -czf $BACKUP_DIR/config_backup_$DATE.tar.gz .env docker-compose.yml nginx/

echo "   Cleaning old backups..."
find $BACKUP_DIR -name "*.sql" -mtime +7 -delete
find $BACKUP_DIR -name "*.tar.gz" -mtime +7 -delete

echo "   Backup completed: $DATE"
EOF

chmod +x /opt/mentalia/backup.sh
```

```
# Configurar cron jobs
echo "   Configurando tareas programadas..."
(crontab -l 2>/dev/null; echo "*/5 * * * * /opt/mentalia/health-check.sh >> /var/log/mentalia-health.log 2>&1") | crontab -
(crontab -l 2>/dev/null; echo "0 2 * * * /opt/mentalia/backup.sh >> /var/log/mentalia-backup.log 2>&1") | crontab -

# Configurar logrotate
cat > /etc/logrotate.d/mentalia << EOF
/var/log/mentalia-*.log {
    daily
    missingok
    rotate 30
    compress
    delaycompress
    notifempty
    create 644 root root
}
EOF

# Reiniciar Nginx
systemctl restart nginx

echo "   ¡Instalación completada!"
echo
echo "   URLs de acceso:"
echo "   Frontend: https://${DOMAIN}"
echo "   API: https://${DOMAIN}/api"
echo "   Monitoreo: https://${DOMAIN}/monitoring"
echo
echo "   Credenciales:"
echo "   Monitoreo: admin / $(grep admin /etc/nginx/.htpasswd | cut -d: -f2)"
echo "   Grafana: admin / $(grep GRAFANA_PASSWORD .env | cut -d= -f2)"
echo
echo "   Comandos útiles:"
echo "   Health check: /opt/mentalia/health-check.sh"
echo "   Backup: /opt/mentalia/backup.sh"
echo "   Logs: docker-compose logs -f"
echo "   Restart: docker-compose restart"
echo
echo "   Archivos importantes:"
echo "   Configuración: /opt/mentalia/.env"
echo "   Logs: /var/log/mentalia-*.log"
echo "   Backups: /opt/mentalia/backups/"
```

3. Monitoreo y Mantenimiento

```
# Comandos de administración frecuentes

# Ver estado de servicios
docker-compose ps

# Ver logs en tiempo real
docker-compose logs -f

# Reiniciar servicio específico
docker-compose restart [servicio]

# Actualizar código
git pull && docker-compose build && docker-compose up -d

# Verificar recursos del sistema
htop

# Verificar espacio en disco
df -h

# Ver conexiones de red
netstat -tulpn

# Backup manual
/opt/mentalia/backup.sh

# Health check manual
/opt/mentalia/health-check.sh
```

4. Solución de Problemas Comunes

```
# Problema: Servicio no responde
# Solución: Reiniciar servicio específico
docker-compose restart [servicio]

# Problema: Base de datos corrupta
# Solución: Restaurar desde backup
docker-compose stop postgres-main
docker volume rm mentalia_postgres_data
docker-compose up -d postgres-main
# Esperar 30 segundos
cat /opt/mentalia/backups/db_backup_[fecha].sql | docker-compose exec -T postgres-main
psql -U mentalia

# Problema: Espacio en disco lleno
# Solución: Limpiar logs y backups antiguos
```

```
docker system prune -f
find /var/log -name "*.log" -mtime +30 -delete
find /opt/mentalia/backups -mtime +30 -delete

# Problema: SSL expirado
# Solución: Renovar certificado
certbot renew --nginx

# Problema: Alto uso de CPU
# Solución: Verificar procesos y escalar si es necesario
htop
docker-compose up -d --scale [servicio]=3
```

5. Escalabilidad y Optimización

```
 # docker-compose.override.yml para producción
version: '3.8'

services:
 frontend-main:
  deploy:
   replicas: 3
   resources:
    limits:
     cpus: '1.0'
     memory: 1G
    reservations:
     cpus: '0.5'
     memory: 512M

 educacion-service:
  deploy:
   replicas: 2
   resources:
    limits:
     cpus: '2.0'
     memory: 2G
    reservations:
     cpus: '1.0'
     memory: 1G

 postgres-main:
  deploy:
   resources:
    limits:
     cpus: '4.0'
     memory: 8G
    reservations:
     cpus: '2.0'
     memory: 4G
```

```
    command: postgres -c max_connections=200 -c shared_buffers=2GB -c
  effective_cache_size=6GB
```

Esta guía proporciona todo lo necesario para desplegar el ecosistema MENTALIA en RunPod de manera robusta y escalable.