

Deploying Machhi Fish Delivery App to Oracle Cloud Free Tier

Prerequisites

- Oracle Cloud Account (Free Tier)
- GitHub Repository (for CI/CD)
- Domain name (optional, but recommended)

Architecture Overview

```
Internet → Load Balancer → Compute Instance (Ubuntu)
                        ↓
                    PostgreSQL Database
                        ↓
                Object Storage (Images)
```

Step 1: Set Up Oracle Cloud Infrastructure

1.1 Create Virtual Cloud Network (VCN)

```
# Oracle Cloud Console → Networking → Virtual Cloud Networks
# Create VCN with:
# - CIDR: 10.0.0.0/16
# - Public Subnet: 10.0.0.0/24
# - Internet Gateway attached
```

1.2 Create Security List

```
# Allow inbound traffic:
# - SSH (22) - from your IP
# - HTTP (80) - from anywhere
# - HTTPS (443) - from anywhere
# - PostgreSQL (5432) - from VCN only
```

Step 2: Launch Compute Instance

2.1 Create Ubuntu Instance

```
# Oracle Cloud Console → Compute → Instances
# Choose:
```

```
# - Ubuntu 22.04 (Canonical)
# - VM.Standard.A1.Flex (Free Tier eligible)
# - 4 OCPU, 24 GB RAM (within free limits)
# - Boot Volume: 200 GB
# - SSH Keys: Add your public key
```

2.2 Initial Server Setup

```
# SSH into your instance
ssh ubuntu@<public-ip>

# Update system
sudo apt update && sudo apt upgrade -y

# Install essential packages
sudo apt install -y curl wget git htop ufw fail2ban

# Configure firewall
sudo ufw allow ssh
sudo ufw allow 80
sudo ufw allow 443
sudo ufw --force enable
```

Step 3: Set Up PostgreSQL Database

3.1 Install PostgreSQL

```
# Install PostgreSQL
sudo apt install -y postgresql postgresql-contrib

# Start and enable PostgreSQL
sudo systemctl start postgresql
sudo systemctl enable postgresql

# Create database and user
sudo -u postgres psql

# In PostgreSQL shell:
CREATE DATABASE machhi_db;
CREATE USER machhi_user WITH ENCRYPTED PASSWORD 'your_secure_password';
GRANT ALL PRIVILEGES ON DATABASE machhi_db TO machhi_user;
ALTER USER machhi_user CREATEDB;
\q
```

3.2 Configure PostgreSQL for Remote Access

```
# Edit postgresql.conf
sudo nano /etc/postgresql/14/main/postgresql.conf
# Change: listen_addresses = '*'

# Edit pg_hba.conf
sudo nano /etc/postgresql/14/main/pg_hba.conf
# Add: host      machhi_db      machhi_user      10.0.0.0/16      md5

# Restart PostgreSQL
sudo systemctl restart postgresql
```

Step 4: Set Up Application Directory

4.1 Clone and Configure

```
# Create application directory
sudo mkdir -p /var/www/machhi
sudo chown ubuntu:ubuntu /var/www/machhi
cd /var/www/machhi

# Clone your repository
git clone https://github.com/yourusername/machhi.git .
cd machhi

# Install Node.js 18
curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

# Install PM2 for process management
sudo npm install -g pm2

# Install dependencies
cd backend && npm install
cd ../frontend && npm install
```

4.2 Configure Environment Variables

```
# Backend .env
cd /var/www/machhi/backend
nano .env

# Add these variables:
NODE_ENV=production
PORT=5000
JWT_SECRET=your_super_secure_jwt_secret_here
DATABASE_URL=postgresql://machhi_user:your_secure_password@localhost:5432/machhi_db
UPLOAD_PATH=/var/www/machhi/uploads
```

```
# Frontend .env
cd /var/www/machhi/frontend
nano .env

# Add:
REACT_APP_API_URL=http://localhost:5000
```

Step 5: Set Up Object Storage for Images

5.1 Create Object Storage Bucket

```
# Oracle Cloud Console → Storage → Object Storage
# Create bucket: machhi-images
# Make it public for read access
```

5.2 Install OCI CLI

```
# Install OCI CLI
curl -L https://raw.githubusercontent.com/oracle/oci-cli/master/scripts/install/install.sh | bash
source ~/.bashrc

# Configure OCI CLI
oci setup config
# Follow prompts to configure with your Oracle Cloud credentials
```

5.3 Create Upload Script

```
# Create upload script
nano /var/www/machhi/upload-to-oci.sh

#!/bin/bash
# Upload file to OCI Object Storage
FILE_PATH=$1
OBJECT_NAME=$2

oci os object put \
  --bucket-name machhi-images \
  --file $FILE_PATH \
  --name $OBJECT_NAME \
  --force

echo "https://objectstorage.us-ashburn-1.oraclecloud.com/n/your-namespace/b/machhi-images/o/$OBJECT_NAME"
```

Step 6: Build and Deploy Application

6.1 Build Frontend

```
cd /var/www/machhi/frontend
npm run build
```

6.2 Set Up Nginx

```
# Install Nginx
sudo apt install -y nginx

# Create Nginx configuration
sudo nano /etc/nginx/sites-available/machhi

server {
    listen 80;
    server_name your-domain.com www.your-domain.com;

    # Frontend (React)
    location / {
        root /var/www/machhi/frontend/build;
        index index.html index.htm;
        try_files $uri $uri/ /index.html;
    }

    # Backend API
    location /api {
        proxy_pass http://localhost:5000;
        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;
    }

    # Static files
    location /uploads {
        alias /var/www/machhi/uploads;
        expires 1y;
        add_header Cache-Control "public, immutable";
    }
}

# Enable site
sudo ln -s /etc/nginx/sites-available/machhi /etc/nginx/sites-enabled/
```

```
sudo nginx -t  
sudo systemctl reload nginx
```

6.3 Set Up SSL with Let's Encrypt

```
# Install Certbot  
sudo apt install -y certbot python3-certbot-nginx  
  
# Get SSL certificate  
sudo certbot --nginx -d your-domain.com -d www.your-domain.com  
  
# Set up auto-renewal  
sudo crontab -e  
# Add: 0 12 * * * /usr/bin/certbot renew --quiet
```

6.4 Start Backend with PM2

```
cd /var/www/machhi/backend  
  
# Seed database  
npm run seed  
npm run seed-products  
  
# Start with PM2  
pm2 start index.js --name "machhi-backend"  
pm2 startup  
pm2 save  
  
# Check status  
pm2 status  
pm2 logs machhi-backend
```

Step 7: Set Up CI/CD (Optional but Recommended)

7.1 GitHub Actions Workflow

```
# .github/workflows/deploy.yml  
name: Deploy to Oracle Cloud  
  
on:  
  push:  
    branches: [ main ]  
  
jobs:  
  deploy:  
    runs-on: ubuntu-latest
```

```

steps:
  - uses: actions/checkout@v3

  - name: Deploy to Oracle Cloud
    uses: appleboy/ssh-action@master
    with:
      host: ${ secrets.ORACLE_HOST }
      username: ubuntu
      key: ${ secrets.ORACLE_SSH_KEY }
      script: |
        cd /var/www/machhi
        git pull origin main
        cd backend && npm install && pm2 restart machhi-backend
        cd ../frontend && npm install && npm run build
        sudo systemctl reload nginx

```

Step 8: Monitoring and Maintenance

8.1 Set Up Monitoring

```

# Install monitoring tools
sudo apt install -y htop iotop ncd

# PM2 monitoring
pm2 monit

# Check logs
pm2 logs machhi-backend --lines 100
sudo tail -f /var/log/nginx/access.log

```

8.2 Backup Strategy

```

# Create backup script
nano /var/www/machhi/backup.sh

#!/bin/bash
DATE=$(date +%Y%m%d_%H%M%S)
pg_dump machhi_db > /var/www/backups/db_backup_${DATE}.sql
tar -czf /var/www/backups/uploads_${DATE}.tar.gz /var/www/machhi/uploads/

# Upload to Object Storage
oci os object put --bucket-name machhi-backups --file
/var/www/backups/db_backup_${DATE}.sql
oci os object put --bucket-name machhi-backups --file
/var/www/backups/uploads_${DATE}.tar.gz

```

8.3 Security Hardening

```
# Disable root login
sudo sed -i 's/#PermitRootLogin yes/PermitRootLogin no/' /etc/ssh/sshd_config
sudo systemctl restart ssh

# Set up fail2ban
sudo systemctl enable fail2ban
sudo systemctl start fail2ban

# Regular updates
sudo apt update && sudo apt upgrade -y
```

Step 9: Domain and DNS Configuration

9.1 Point Domain to Oracle Cloud

```
# In your domain registrar, create A records:
# @ → <your-instance-public-ip>
# www → <your-instance-public-ip>
```

9.2 Oracle Cloud DNS (Alternative)

```
# Oracle Cloud Console → DNS Management
# Create zone for your domain
# Add A records pointing to instance IP
```

☒ Step 10: Final Verification

10.1 Test Deployment

```
# Test frontend
curl -I http://your-domain.com

# Test API
curl http://your-domain.com/api/categories

# Test SSL
curl -I https://your-domain.com

# Check PM2 status
pm2 status

# Check Nginx status
sudo systemctl status nginx
```


10.2 Performance Optimization

```
# Enable gzip compression in Nginx
sudo nano /etc/nginx/nginx.conf
# Add: gzip on; gzip_types text/plain text/css application/json
application/javascript text/xml application/xml application/xml+rss
text/javascript;

# Restart Nginx
sudo systemctl restart nginx

# Database optimization
sudo -u postgres psql -d machhi_db
# Run: CREATE INDEX CONCURRENTLY idx_products_category ON products("categoryId");
# Run: CREATE INDEX CONCURRENTLY idx_orders_user ON orders("userId");
```

Free Tier Limits Reminder

- **Compute:** 2 AMD-based VMs (up to 1/8 OCPU and 1 GB memory each) OR 1 AMD-based VM (up to 1 OCPU and 6 GB memory)
- **Block Storage:** 200 GB total
- **Object Storage:** 20 GB
- **Autonomous Database:** 2 ADB instances (up to 1 OCPU and 20 GB storage each)
- **Load Balancer:** 1 instance (up to 10 Mbps)

Important Notes

1. **Always Free Resources:** Stick to free tier limits to avoid charges
2. **Security:** Regularly update your instance and monitor for vulnerabilities
3. **Backups:** Set up automated backups of your database and uploads
4. **Monitoring:** Use PM2 and Oracle Cloud monitoring tools
5. **Scaling:** Consider upgrading to paid tiers as your app grows

Troubleshooting

Common Issues:

Nginx not serving React app:

```
sudo nginx -t
sudo systemctl reload nginx
```

PM2 process not starting:

```
cd /var/www/machhi/backend  
pm2 delete machhi-backend  
pm2 start index.js --name "machhi-backend"
```

Database connection issues:

```
sudo -u postgres psql  
# Check if database exists: \l  
# Check user permissions: \du
```

SSL certificate issues:

```
sudo certbot certificates  
sudo certbot renew
```

Your Machhi fish delivery app is now successfully deployed on Oracle Cloud Free Tier! 🐟