Testing and deployment

Testing

Current Test Setup

The project currently has limited automated tests. Main test file:

- `referral_system_test.js` - Tests for referral system calculations

Running Tests

```
npm test
```

Manual Testing

1. **Use Postman/Insomnia**
   - Import OpenAPI spec from `docs/openapi.json`
   - Test endpoints manually
2. **Use Scalar API Reference**
   - Access `http://localhost:3610/api/kitab-suci` in development
   - Test directly from browser
3. **WebSocket Testing**
   - Use Socket.IO client library
   - Or use tools like Socket.IO Client Tool

Recommended Test Coverage

Future test implementation should cover:

- Unit tests for utilities and services
- Integration tests for API endpoints
- E2E tests for critical flows (checkout, payment)
- Load testing for concurrent operations

Deployment

Production Checklist

1. **Environment Configuration**
   - ☐ Set `ENV=PRODUCTION`
   - ☐ Configure production database credentials
   - ☐ Set secure JWT secret
   - ☐ Configure production API keys (Xendit, GoSend)
   - ☐ Set CORS origins
   - ☐ Configure email settings
2. **Database**
   - ☐ Run database migrations
   - ☐ Set up database backups
   - ☐ Configure connection pooling
   - ☐ Set up read replicas (if needed)
3. **Security**
   - ☐ Enable HTTPS
   - ☐ Configure Helmet security headers
   - ☐ Set up rate limiting
   - ☐ Configure firewall rules
   - ☐ Set up SSL certificates
4. **Performance**
   - ☐ Enable Redis caching
   - ☐ Configure CDN for static assets
   - ☐ Set up load balancer
   - ☐ Configure process manager (PM2)
5. **Monitoring**
   - ☐ Set up logging (Winston/Bunyan)
   - ☐ Configure error tracking (Sentry)
   - ☐ Set up uptime monitoring
   - ☐ Configure performance monitoring (New Relic/DataDog)

Deployment Options

Option 1: VPS/Dedicated Server

```
# Install PM2
npm install -g pm2

# Start application
pm2 start index.js --name onmarket-api

# Configure PM2 startup
pm2 startup
pm2 save

# Monitor
pm2 monit
```

Option 2: Docker

Create `Dockerfile`:

```
FROM node:16-alpine

WORKDIR /app

COPY package*.json ./
RUN npm ci --only=production

COPY . .

EXPOSE 3610

CMD ["node", "index.js"]
```

Create `docker-compose.yml`:

```
version: '3.8'
services:
  api:
    build: .
```

```
    ports:
      - "3610:3610"
    env_file:
      - .env
    depends_on:
      - postgres
      - mongo
      - redis

  postgres:
    image: postgres:14
    environment:
      POSTGRES_DB: onmarket_db
      POSTGRES_USER: onmarket
      POSTGRES_PASSWORD: password
    volumes:
      - postgres-data:/var/lib/postgresql/data

  mongo:
    image: mongo:5
    volumes:
      - mongo-data:/data/db

  redis:
    image: redis:7-alpine
    command: redis-server --requirepass yourpassword

volumes:
  postgres-data:
  mongo-data:
```

Run with Docker:

```
docker-compose up -d
```

Option 3: Cloud Platforms

AWS:

- Use Elastic Beanstalk
- Or ECS with Fargate

- RDS for PostgreSQL
- DocumentDB for MongoDB
- ElastiCache for Redis

Google Cloud:

- Cloud Run
- Cloud SQL
- Memorystore

Heroku:

```
heroku create onmarket-api
heroku addons:create heroku-postgresql
heroku addons:create mongolab
heroku addons:create heroku-redis
git push heroku main
```

Nginx Configuration

```
server {
    listen 80;
    server_name api.yourdomain.com;

    location / {
        proxy_pass http://localhost:3610;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection 'upgrade';
        proxy_set_header Host $host;
        proxy_cache_bypass $http_upgrade;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }

    # WebSocket support
    location /socket.io/ {
        proxy_pass http://localhost:3610;
        proxy_http_version 1.1;
        proxy_set_header Upgrade $http_upgrade;
        proxy_set_header Connection "upgrade";
```

```
        proxy_set_header Host $host;
    }
}
```

Revision #1
Created 24 February 2026 09:29:42 by ondeliveloper
Updated 24 February 2026 09:30:04 by ondeliveloper