ChatLive Pro - دردشة فيديو احترافية

تطبيق دردشة فيديو احترافي باللغة العربية يستخدم WebRTC و Socket.io لتوفير تجربة دردشة مرئية سلسة وآمنة.

🌟 المميزات

- 📹 **دردشة فيديو عالية الجودة** باستخدام WebRTC
- 💬 **رسائل نصية فورية** مع Socket.io
- 🌍 **مطابقة حسب الدولة والاهتمامات**
- 📱 **تصميم متجاوب** يعمل على جميع الأجهزة
- 🔒 **آمن ومشفر** بأحدث معايير الأمان
- 📊 **إحصائيات مباشرة** للمستخدمين والغرف
- 🎨 **واجهة عربية جميلة** وسهلة الاستخدام

📋 المتطلبات

النظام

- **Node.js** (الإصدار 16.0.0 أو أحدث)
- **npm** أو **yarn**
- **MongoDB** (محلي أو سحابي)

المتصفحات المدعومة

- Chrome 80+
- Firefox 75+
- Safari 13+
- Edge 80+

🚀 التثبيت السريع

1. تحميل المشروع

bash

```
# استنساخ المشروع (إذا كان على Git)
git clone https://github.com/your-username/chatlive-pro.git
cd chatlive-pro

# أو إنشاء مجلد جديد
mkdir chatlive-pro
cd chatlive-pro
```

2. إنشاء ملفات المشروع

قم بإنشاء الملفات التالية في مجلد المشروع:

أ) ملف package.json

bash

```
npm init -y
```

ثم استبدل محتوى الملف بالكود المطلوب.

ب) تثبيت المكتبات

bash

```
npm install express socket.io cors dotenv bcryptjs jsonwebtoken mongoose helmet rate-limiter-fl
npm install -D nodemon
```

3. إنشاء هيكل المجلدات

bash

```
mkdir config models utils public
```

4. إعداد قاعدة البيانات

محلي MongoDB:

bash

```
# تثبيت MongoDB على Ubuntu/Debian
sudo apt-get install mongodb

# بدء خدمة MongoDB
sudo systemctl start mongodb
sudo systemctl enable mongodb
```

MongoDB سحابي (MongoDB Atlas):

1. اذهب إلى MongoDB Atlas
2. أنشئ حساب مجاني
3. أنشئ Cluster جديد
4. احصل على Connection String

5. إعداد متغيرات البيئة

أنشئ ملف `.env` في المجلد الجذر:

bash

```
touch .env
```

6. تشغيل التطبيق

bash

```
# في وضع التطوير
npm run dev

# في وضع الإنتاج
npm start
```

📁 هيكل المشروع

```
chatlive-pro/
├── config/
│   └── database.js          # إعداد قاعدة البيانات
├── models/
│   └── User.js              # نموذج بيانات المستخدم
├── utils/
│   └── roomManager.js       # مدير الغرف
├── public/
│   └── index.html           # الواجهة الأمامية
├── server.js                # السيرفر الرئيسي
├── package.json             # إعدادات المشروع
├── .env                     # متغيرات البيئة
└── README.md                # هذا الملف
```

⚙️ إعداد الإنتاج

1. إعداد Nginx

تثبيت Nginx:

bash

```
sudo apt update
sudo apt install nginx
```

إعداد ملف الموقع:

bash

```
sudo nano /etc/nginx/sites-available/chatlive-pro
```

ملف الإعداد:

nginx

```nginx
server {
    listen 80;
    server_name yourdomain.com;
    return 301 https://$server_name$request_uri;
}

server {
    listen 443 ssl http2;
    server_name yourdomain.com;

    ssl_certificate /etc/letsencrypt/live/yourdomain.com/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/yourdomain.com/privkey.pem;

    ssl_session_timeout 1d;
    ssl_session_cache shared:SSL:50m;
    ssl_stapling on;
    ssl_stapling_verify on;

    location / {
        proxy_pass http://localhost:3001;
        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;
        proxy_read_timeout 86400;
    }
}
```

تفعيل الموقع:

bash

```bash
sudo ln -s /etc/nginx/sites-available/chatlive-pro /etc/nginx/sites-enabled/
sudo nginx -t
sudo systemctl reload nginx
```

2. الحصول على شهادة SSL

باستخدام Let's Encrypt:

```bash
# تثبيت Certbot
sudo apt install certbot python3-certbot-nginx

# الحصول على الشهادة
sudo certbot --nginx -d yourdomain.com

# تجديد تلقائي
sudo crontab -e
# إضافة السطر التالي:
# 0 12 * * * /usr/bin/certbot renew --quiet
```

3. استخدام PM2 لإدارة العملية

تثبيت PM2:

```bash
sudo npm install -g pm2
```

إنشاء ملف ecosystem.config.js:

```javascript
module.exports = {
  apps: [{
    name: 'chatlive-pro',
    script: 'server.js',
    instances: 'max',
    exec_mode: 'cluster',
    env: {
      NODE_ENV: 'development'
    },
    env_production: {
      NODE_ENV: 'production',
      PORT: 3001
    }
  }]
};
```

تشغيل التطبيق:

bash

```bash
# بدء التطبيق
pm2 start ecosystem.config.js --env production

# حفظ الإعدادات
pm2 save

# تشغيل تلقائي عند البدء
pm2 startup
```

🐳 نشر باستخدام Docker

إنشاء Dockerfile:

dockerfile

```dockerfile
FROM node:16-alpine

WORKDIR /app

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

COPY . .

EXPOSE 3001

USER node

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

إنشاء docker-compose.yml:

yaml

```yaml
version: '3.8'

services:
  app:
    build: .
    ports:
      - "3001:3001"
    environment:
      - NODE_ENV=production
      - MONGODB_URI=mongodb://mongo:27017/videochat
    depends_on:
      - mongo
    restart: unless-stopped

  mongo:
    image: mongo:5.0
    ports:
      - "27017:27017"
    volumes:
      - mongo_data:/data/db
    restart: unless-stopped

  nginx:
    image: nginx:alpine
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf
      - /etc/letsencrypt:/etc/letsencrypt
    depends_on:
      - app
    restart: unless-stopped

volumes:
  mongo_data:
```

تشغيل Docker:

bash

```
# بناء وتشغيل
docker-compose up -d

# مراقبة اللوجز
docker-compose logs -f
```

☁️ نشر على الخدمات السحابية

Heroku

1. تثبيت Heroku CLI:

bash

```
npm install -g heroku
```

2. تسجيل الدخول وإنشاء التطبيق:

bash

```
heroku login
heroku create your-app-name
```

3. إضافة قاعدة البيانات:

bash

```
heroku addons:create mongolab:sandbox
```

4. ضبط متغيرات البيئة:

bash

```
heroku config:set NODE_ENV=production
heroku config:set JWT_SECRET=your-secret-key
```

5. نشر التطبيق:

bash

```
git add .
git commit -m "Deploy to Heroku"
git push heroku main
```

DigitalOcean

1. إنشاء Droplet:

- اختر Ubuntu 20.04
- حجم مناسب (2GB RAM على الأقل)
- إضافة SSH Key

2. الاتصال والإعداد:

bash

```
ssh root@your-server-ip

# تحديث النظام
apt update && apt upgrade -y

# تثبيت Node.js
curl -fsSL https://deb.nodesource.com/setup_16.x | sudo -E bash -
apt-get install -y nodejs

# تثبيت MongoDB
wget -qO - https://www.mongodb.org/static/pgp/server-5.0.asc | sudo apt-key add -
echo "deb [ arch=amd64,arm64 ] https://repo.mongodb.org/apt/ubuntu focal/mongodb-org/5.0 multiv
apt-get update
apt-get install -y mongodb-org

# تفعيل MongoDB
systemctl start mongod
systemctl enable mongod
```

3. نشر التطبيق:

bash

```
# إنشاء مستخدم جديد
adduser chatapp
usermod -aG sudo chatapp
su - chatapp

# رفع الملفات
git clone your-repo-url
cd chatlive-pro
npm install --production

# إعداد متغيرات البيئة
cp .env.example .env
nano .env

# تشغيل التطبيق
npm start
```

🔧 استكشاف الأخطاء

مشاكل شائعة وحلولها:

1. خطأ في الاتصال بقاعدة البيانات:

bash

```
# التحقق من تشغيل MongoDB
sudo systemctl status mongod

# فحص لوجز MongoDB
sudo tail -f /var/log/mongodb/mongod.log
```

2. مشاكل في WebRTC:

- تأكد من استخدام HTTPS في الإنتاج
- تحقق من إعدادات STUN/TURN servers
- اختبر على متصفحات مختلفة

3. مشاكل في Socket.io:

```javascript
// إضافة تشخيص في الكود
this.socket.on('connect_error', (error) => {
    console.error('فشل اتصال Socket.io:', error);
});
```

4. مشاكل في الأداء:

```bash
# مراقبة استخدام الموارد
htop
iotop

# فحص لوجز التطبيق
pm2 logs chatlive-pro
```

📊 المراقبة والتحليلات

إعداد Monitoring:

1. PM2 Monitoring:

```bash
pm2 install pm2-server-monit
```

2. استخدام Grafana + Prometheus:

```yaml
# إضافة إلى docker-compose.yml
  prometheus:
    image: prom/prometheus
    ports:
      - "9090:9090"
    volumes:
      - ./prometheus.yml:/etc/prometheus/prometheus.yml

  grafana:
    image: grafana/grafana
    ports:
      - "3000:3000"
    environment:
      - GF_SECURITY_ADMIN_PASSWORD=admin
```

🔒 الأمان

نصائح هامة للأمان:

1. **استخدم HTTPS** **دائماً** في الإنتاج
2. **غيّر المفاتيح السرية** في .env
3. **فعّل Rate Limiting** لمنع الهجمات
4. **استخدم Helmet.js** لحماية Headers
5. **راقب اللوجز** بانتظام
6. **حدّث المكتبات** دورياً

إعدادات جدار الحماية:

```bash
# Ubuntu UFW
sudo ufw allow ssh
sudo ufw allow 80
sudo ufw allow 443
sudo ufw enable
```

📈 التحسينات

تحسين الأداء:

1. **استخدم CDN** للملفات الثابتة
2. **فعّل Gzip compression**
3. **استخدم Redis** للـ Sessions
4. **حسّن قاعدة البيانات** بالفهارس
5. **استخدم Load Balancer** للحمولة العالية

🆘 الدعم

الحصول على المساعدة:

1. **فحص اللوجز** أولاً
2. **البحث في Documentation**
3. **التأكد من إصدارات المكتبات**
4. **اختبار على بيئة تطوير منفصلة**

معلومات مفيدة للدعم:

- إصدار Node.js: `node --version`
- إصدار npm: `npm --version`
- نظام التشغيل: `uname -a`
- إصدار MongoDB: `mongod --version`

📝 الترخيص

هذا المشروع مرخص تحت رخصة MIT. راجع ملف LICENSE للتفاصيل.

🤝 المساهمة

نرحب بالمساهمات! يرجى قراءة CONTRIBUTING.md للمزيد من التفاصيل.

ملاحظة مهمة: تأكد من اختبار التطبيق جيداً في بيئة التطوير قبل النشر في الإنتاج.