

# التوثيق الفني الشامل - تطبيق مزاد الحيوانات النادرة

المؤلف: Manus AI

يوليو 2025: التاريخ

1.0: الإصدار

## جدول المحتويات

- [مقدمة المشروع](#)
- [البنية التقنية العامة](#)
- [التوثيق الفني للخادم الخلفي](#)
- [دليل المطور للواجهة الأمامية](#)
- [دليل التطبيق الجوال](#)
- [دليل لوحة التحكم الإدارية](#)
- [دليل التثبيت والنشر](#)
- [دليل استكشاف الأخطاء](#)
- [دليل الأمان وأفضل الممارسات](#)
- [دليل المستخدم النهائي](#)

## مقدمة المشروع

تطبيق مزاد الحيوانات النادرة هو منصة متكاملة ومتطورة تهدف إلى توفير بيئة آمنة وموثوقة لتداول الحيوانات النادرة والاستثنائية في المنطقة العربية. يجمع هذا المشروع بين التقنيات الحديثة والتصميم المتطور لتقديم تجربة مستخدم فريدة ومميزة تلبي احتياجات جميع الأطراف المشاركة في عملية المزاد.

## الرؤية والأهداف

تتمحور رؤية المشروع حول إنشاء أول منصة رقمية متخصصة في مزادات الحيوانات النادرة في المنطقة، مع التركيز على الشفافية والأمان والسهولة في الاستخدام. يهدف التطبيق إلى ربط محبي الحيوانات النادرة والمربين المحترفين والهواة في بيئة تفاعلية تضمن حقوق جميع الأطراف وتوفر آليات دفع آمنة ونظام مزايادة عادل وشفاف.

## المميزات الرئيسية

يتميز التطبيق بمجموعة شاملة من المميزات التي تجعله الخيار الأمثل لمزادات الحيوانات النادرة. مما يضمن تحديثات فورية، WebSocket يوفر النظام مزادات مباشرة بالزمن الحقيقي باستخدام تقنية لجميع المزايدات والتغييرات. كما يتضمن نظام دفع متكامل يدعم المحافظ الإلكترونية والتحويلات البنكية مع آليات استرداد تلقائية للمزايد غير الفائزين.

يدعم التطبيق اللغة العربية بشكل كامل مع اتجاه النص من اليمين إلى اليسار، ويوفر واجهات مستخدم متجاوبة تعمل بسلاسة على جميع الأجهزة. النظام مصمم ليكون قابلاً للتوسع والصيانة، مع بنية تقنية حديثة تضمن الأداء العالي والاستقرار.

## نطاق المشروع

يشمل المشروع تطوير أربعة مكونات رئيسية متكاملة: التطبيق الجوال للمستخدمين النهائيين، الواجهة الأمامية للويب، لوحة التحكم الإدارية، والخادم الخلفي مع قاعدة البيانات. كل مكون مصمم ليعمل بتناغم مع الآخرين لتوفير تجربة متسقة وموحدة عبر جميع المنصات.

Android و iOS لضمان التوافق مع أنظمة Expo مع React Native التطبيق الجوال مطور باستخدام لتصميم عصري ومتجاوب Tailwind CSS مع React.js بينما الواجهة الأمامية للويب مبنية باستخدام لوحة التحكم الإدارية توفر أدوات شاملة لإدارة المستخدمين والمزادات والمدفوعات، والخادم الخلفي لضمان الأداء والمرونة MongoDB وقاعدة بيانات Express.js مع Node.js مبنية باستخدام

## البنية التقنية العامة

### نظرة عامة على المعمارية

تعتمد منصة مزاد الحيوانات النادرة على معمارية حديثة ومتطورة تجمع بين أفضل الممارسات في (Microservices Architecture) تطوير التطبيقات الحديثة. النظام مبنية على نمط العمارة المنفصلة مع فصل واضح بين الواجهة الأمامية والخادم الخلفي، مما يضمن المرونة في التطوير والصيانة والتوسع المستقبلي.

(APIs) الخادم الخلفي يعمل كنقطة مركزية لجميع العمليات والبيانات، ويوفر واجهات برمجة تطبيقات موحدة يمكن للواجهات المختلفة الوصول إليها. هذا التصميم يضمن الاتساق في البيانات والعمليات عبر جميع المنصات، ويسهل إضافة منصات جديدة في المستقبل دون الحاجة لتعديل الخادم الخلفي.

## التقنيات المستخدمة

### (Backend) الخادم الخلفي

- Node.js v20.18.0: عالية الأداء JavaScript بيئة تشغيل
- Express.js v4.19.2: إطار عمل ويب سريع ومرن
- MongoDB: قابلة للتوسع NoSQL قاعدة بيانات

- **Mongoose:** مكتبة ODM لـ MongoDB
- **Socket.IO:** للاتصال المباشر والتحديثات الفورية
- **JWT:** للتوثيق والأمان
- **bcryptjs:** لتشفير كلمات المرور
- **Multer:** لرفع الملفات
- **Nodemailer:** لإرسال البريد الإلكتروني

## (Frontend) الواجهة الأمامية

- **React.js v18:** لبناء واجهات المستخدم باستخدام JavaScript مكتبة
- **Vite:** أداة بناء سريعة ومتطورة
- **Tailwind CSS:** للتصميم السريع CSS إطار عمل
- **Axios:** لطلبات HTTP
- **React Router:** للتنقل بين الصفحات
- **Socket.IO Client:** للاتصال المباشر مع الخادم

## (Mobile App) التطبيق الجوال

- **React Native:** إطار عمل لتطوير التطبيقات الجوال
- **Expo:** منصة لتطوير ونشر تطبيقات React Native
- **React Navigation:** للتنقل في التطبيق
- **Expo Vector Icons:** للأيقونات
- **AsyncStorage:** لتخزين البيانات محلياً

## (Admin Dashboard) لوحة التحكم الإدارية

- **React.js v18:** نفس تقنية الواجهة الأمامية للاتساق
- **Material-UI:** متقدمة الـ مكتبة مكونات
- **Chart.js:** للرسوم البيانية والإحصائيات
- **React Table:** لعرض البيانات في جداول

## هيكل قاعدة البيانات

لتوفير المرونة والأداء العالي. النظام يستخدم مجموعات MongoDB قاعدة البيانات مصممة باستخدام منفصلة لكل نوع من البيانات مع روابط مرجعية بينها لضمان تكامل البيانات (Collections).

### :المجموعات الرئيسية

**Users Collection:** تحتوي على بيانات المستخدمين الأساسية مثل الاسم والبريد الإلكتروني ورقم الهاتف ومعلومات التوثيق. كل مستخدم له معرف فريد ودور محدد (مستخدم عادي، مشرف، مدير).

**Animals Collection:** تخزن معلومات الحيوانات المعروضة للمزاد، تشمل الاسم والوصف والفئة والصورة والشهادات الصحية ومعلومات البائع وحالة الموافقة.

**Auctions Collection:** تحتوي على تفاصيل المزادات مثل تاريخ البداية والنهاية والسعر الافتتاحي والحد الأدنى وسعر الشراء الفوري وحالة المزاد.

**Bids Collection:** تسجل جميع المزايدات مع الطوايع الزمنية ومعلومات المزايد والمبالغ المعروضة.

**Payments Collection:** تتبع جميع المعاملات المالية من رسوم العرض ورسوم المزايدة والمدفوعات النهائية والاسترداد.

**Notifications Collection:** تخزن الإشعارات المرسله للمستخدمين مع حالة القراءة والنوع والمحتوى.

## أمان النظام

مع JWT النظام مصمم مع التركيز على الأمان في جميع المستويات. يستخدم التوثيق المبني على salt مع bcrypt انتهاء صلاحية محدد وتجديد تلقائي للرموز. جميع كلمات المرور مشفرة باستخدام HTTPS عشوائي، وجميع الاتصالات محمية بـ.

من خلال تنظيف المدخلات والتحقق من صحتها. كما XSS و CSRF النظام يتضمن حماية من هجمات لمنع الهجمات المتكررة وحماية الخادم من الحمولة الزائدة rate limiting يستخدم.

## قابلية التوسع والأداء

المعمارية مصممة لتكون قابلة للتوسع الأفقي والعمودي. الخادم الخلفي يمكن توزيعه على عدة خوادم مع موازن تحميل، وقاعدة البيانات تدعم التقسيم والنسخ المتماثل. النظام يستخدم تقنيات التخزين المؤقت لتحسين الأداء وتقليل الحمولة على قاعدة البيانات.

مما يضمن عمل التحديثات الفورية، Redis adapter مكون لدعم التوسع الأفقي باستخدام Socket.IO. حتى مع توزيع التطبيق على عدة خوادم.

## التوثيق الفني للخادم الخلفي

### نظرة عامة على الخادم الخلفي

الخادم الخلفي هو العمود الفقري لمنصة مزاد الحيوانات النادرة، وهو مسؤول عن إدارة جميع يوفر الخادم واجهات، Express.js مع Node.js العمليات التجارية والبيانات والأمان. مبني باستخدام شاملة ومتسقة لجميع العمليات المطلوبة في النظام RESTful برمجة تطبيقات.

الخادم مصمم وفقاً لمبادئ البرمجة النظيفة والمعمارية المتطورة، مع فصل واضح بين طبقات العرض والمنطق التجاري والبيانات. هذا التصميم يضمن سهولة الصيانة والتطوير والاختبار، ويسهل إضافة مميزات جديدة دون التأثير على الوظائف الموجودة.

## هيكل المشروع

```
backend/
├── controllers/           # منطق التحكم في العمليات
│   ├── authController.js  # التوثيق والمصادقة
│   ├── animalController.js # إدارة الحيوانات
│   ├── bidController.js   # إدارة المزادات
│   └── paymentController.js # إدارة المدفوعات
├── models/               # نماذج قاعدة البيانات
│   ├── User.js            # نموذج المستخدم
│   ├── Animal.js          # نموذج الحيوان
│   ├── Bid.js             # نموذج المزايدة
│   ├── Payment.js         # نموذج الدفع
│   └── Notification.js    # نموذج الإشعار
├── routes/               # تعريف المسارات
│   ├── auth.js            # مسارات التوثيق
│   ├── animals.js         # مسارات الحيوانات
│   ├── bids.js            # مسارات المزادات
│   └── payments.js        # مسارات المدفوعات
├── middleware/           # البرمجيات الوسيطة
│   ├── auth.js            # التحقق من التوثيق
│   ├── error.js           # معالجة الأخطاء
│   └── upload.js          # رفع الملفات
├── utils/                # الأدوات المساعدة
│   ├── database.js        # اتصال قاعدة البيانات
│   ├── helpers.js         # دوال مساعدة
│   └── socket.js          # Socket.IO إدارة
├── uploads/              # ملفات مرفوعة
└── server.js             # نقطة دخول التطبيق
```

## (APIs) واجهات برمجة التطبيقات

### التوثيق والمصادقة APIs مجموعة

تسجيل مستخدم جديد في النظام. يتطلب البريد الإلكتروني ورقم الهاتف **POST /api/auth/register** للتحقق من رقم الهاتف قبل تفعيل الحساب OTP وكلمة المرور. النظام يرسل رمز

```
{
  "email": "user@example.com",
  "phone": "+218912345678",
  "password": "securePassword123",
  "username": "اسم المستخدم"
}
```

المرسل للهاتف. يجب إدخال الرمز خلال 10 OTP التحقق من رمز **POST /api/auth/verify-otp** دقائق من الإرسال.

```
{
  "phone": "+218912345678",
  "otp": "123456"
}
```

تسجيل الدخول باستخدام البريد الإلكتروني أو رقم الهاتف وكلمة المرور **POST /api/auth/login**.  
صالح لمدة 24 ساعة JWT token يعيد.

```
{
  "email": "user@example.com",
  "password": "securePassword123"
}
```

refresh token قبل انتهاء صلاحيته. يتطلب JWT token تجديد **POST /api/auth/refresh-token**.  
صالح.

الحصول على معلومات الملف الشخصي للمستخدم المسجل دخوله **GET /api/auth/profile**.  
header صالح في JWT token يتطلب.

تحديث معلومات الملف الشخصي. يمكن تحديث الاسم والصورة الشخصية **PUT /api/auth/profile**.  
ومعلومات الاتصال.

## الحيوانات والمزادات APIs مجموعة

الحصول على قائمة الحيوانات مع إمكانية الفلترة والبحث والترتيب. يدعم **GET /api/animals**.  
للأداء الأمثل pagination.

عدد العناصر في الصفحة: **limit** - رقم الصفحة (افتراضي: 1): **page** - المعاملات المدعومة  
**search** - فلترة حسب حالة المزاد: **status** - فلترة حسب الفئة: **category** - (افتراضي: 10)  
- الحد الأقصى للسعر: **maxPrice** - الحد الأدنى للسعر: **minPrice** - البحث في الاسم والوصف  
**sort**: ترتيب النتائج (price, date, name)

الحصول على تفاصيل حيوان محدد مع تاريخ المزادات والمعلومات الكاملة **GET /api/animals/:id**.

إضافة حيوان جديد للمزاد. يتطلب تسجيل الدخول ودفع رسوم العرض (10 **POST /api/animals**).  
(دينار ليبي).

```
{
  "name": "اسم الحيوان",
  "category": "طيور",
  "description": "وصف مفصل للحيوان",
  "startingPrice": 100,
  "reservePrice": 150,
  "buyNowPrice": 300,
  "auctionDuration": 20,
}
```

```
"images": ["image1.jpg", "image2.jpg"],
"healthCertificate": "certificate.pdf"
}
```

PUT /api/animals/:id تحديث معلومات حيوان. متاح فقط لصاحب الحيوان أو المشرفين.

DELETE /api/animals/:id حذف حيوان من النظام. متاح فقط للمشرفين أو صاحب الحيوان قبل بدء المزاد.

### المزايدات APIs مجموعة

GET /api/bids/animal/:animalId الحصول على تاريخ المزايدات لحيوان محدد مرتبة حسب الوقت.

POST /api/bids وضع مزايدة جديدة. يتطلب دفع رسوم المزايدة (40 دينار ليبي) قابلة للاسترداد.

```
{
  "animalId": "animal_id_here",
  "amount": 200
}
```

GET /api/bids/user الحصول على تاريخ مزايدات المستخدم الحالي.

POST /api/bids/:bidId/confirm تأكيد الفوز بالمزاد ودفع المبلغ النهائي. متاح فقط للفائز في المزاد.

### المدفوعات APIs مجموعة

GET /api/payments/wallet الحصول على رصيد المحفظة الإلكترونية للمستخدم.

POST /api/payments/add-funds إضافة رصيد للمحفظة الإلكترونية.

```
{
  "amount": 500,
  "paymentMethod": "credit_card",
  "paymentDetails": {
    "cardNumber": "****-****-****-1234",
    "expiryDate": "12/25"
  }
}
```

POST /api/payments/withdraw سحب رصيد من المحفظة الإلكترونية.

```
{
  "amount": 200,
  "withdrawMethod": "bank_transfer",
}
```

```
"bankDetails": {  
  "accountNumber": "1234567890",  
  "bankName": "البنك الأهلي"  
}
```

الحصول على تاريخ المعاملات المالية للمستخدم `GET /api/payments/history`.

دفع رسوم عرض حيوان للمزاد (10 دينار ليبي) `POST /api/payments/listing-fee/:animalId`.

الدفع النهائي للحيوان بعد الفوز في المزاد `POST /api/payments/final-payment/:animalId`.

## للتحديثات الفورية Socket.IO نظام

لتوفير تحديثات فورية للمزايدات والإشعارات. عند اتصال المستخدم، يتم Socket.IO النظام يستخدم تسجيله في غرف مخصصة للمزادات التي يتابعها.

### الأحداث المدعومة:

الانضمام لمتابعة مزاد محدد `join_auction` عند اتصال مستخدم جديد `connection`  
إشعار بانتهاء `auction_ended` إشعار بمزايدة جديدة `new_bid` ترك متابعة مزاد `leave_auction`  
إشعار بتجاوز مزايدة المستخدم `bid_outbid` مزاد

## معالجة الأخطاء والتسجيل

النظام يتضمن معالجة شاملة للأخطاء مع تسجيل مفصل لجميع العمليات. الأخطاء مصنفة حسب النوع والخطورة، مع رسائل واضحة للمستخدمين ومعلومات تقنية مفصلة للمطورين.

### أنواع الأخطاء:

- 400 Bad Request: طلب غير صحيح أو بيانات ناقصة
- 401 Unauthorized: غير مصرح بالوصول
- 403 Forbidden: ممنوع الوصول
- 404 Not Found: المورد غير موجود
- 409 Conflict: تضارب في البيانات
- 429 Too Many Requests: تجاوز حد الطلبات
- 500 Internal Server Error: خطأ في الخادم

## الأمان والحماية

الخادم محمي بعدة طبقات أمان:

- JWT Authentication: توثيق مبني على الرموز المميزة
- Rate Limiting: تحديد عدد الطلبات لكل IP

- **Input Validation:** التحقق من صحة جميع المدخلات
- **SQL Injection Protection:** حماية من حقن قواعد البيانات
- **XSS Protection:** حماية من البرمجة النصية المتقاطعة
- **CORS Configuration:** تكوين مشاركة الموارد عبر المصادر
- **Helmet.js:** HTTP حماية إضافية للرؤوس

## متغيرات البيئة

```
NODE_ENV=development
PORT=5000
MONGODB_URI=mongodb://127.0.0.1:27017/rare_animals_auction
JWT_SECRET=your_jwt_secret_key_here
JWT_EXPIRE=24h
REFRESH_TOKEN_SECRET=your_refresh_token_secret
REFRESH_TOKEN_EXPIRE=7d
EMAIL_HOST=smtp.gmail.com
EMAIL_PORT=587
EMAIL_USER=your_email@gmail.com
EMAIL_PASS=your_app_password
SMS_API_KEY=your_sms_api_key
UPLOAD_PATH=./uploads
MAX_FILE_SIZE=5242880
LISTING_FEE=10
BIDDING_FEE=40
PLATFORM_COMMISSION=0.05
```

## دليل المطور للواجهة الأمامية

### نظرة عامة على الواجهة الأمامية

Tailwind كأداة بناء، و Vite مع React.js الواجهة الأمامية لمنصة مزاد الحيوانات النادرة مبنية باستخدام للتصميم. التطبيق مصمم ليكون سريع الاستجابة ومتجاوب مع جميع أحجام الشاشات، مع دعم CSS كامل للغة العربية واتجاه النص من اليمين إلى اليسار.

الواجهة تتبع مبادئ التصميم الحديث مع التركيز على تجربة المستخدم وسهولة الاستخدام. النظام مع إدارة حالة محلية ومشاركة لضمان الأداء الأمثل Component-Based Architecture يستخدم نمط والصيانة السهلة.

### هيكل المشروع

```
auction-frontend/
├── public/
│   ├── index.html
│   └── favicon.ico
```

# الملفات العامة  
# الرئيسي HTML ملف  
# أيقونة الموقع

src/	# الكود المصدري
components/	# المكونات القابلة لإعادة الاستخدام
Header.jsx	# رأس الصفحة
Footer.jsx	# تذييل الصفحة
AnimalCard.jsx	# بطاقة عرض الحيوان
BidTimer.jsx	# عداد وقت المزاد
SearchFilter.jsx	# مكون البحث والفلتر
pages/	# صفحات التطبيق
Home.jsx	# الصفحة الرئيسية
Auctions.jsx	# صفحة المزادات
AnimalDetail.jsx	# تفاصيل الحيوان
Profile.jsx	# الملف الشخصي
Login.jsx	# تسجيل الدخول
hooks/	# React Hooks مخصصة
useAuth.js	# إدارة التوثيق
useSocket.js	# إدارة Socket.IO
useApi.js	# طلبات API
services/	# خدمات التطبيق
api.js	# تكوين Axios
auth.js	# خدمات التوثيق
socket.js	# تكوين Socket.IO
utils/	# الأدوات المساعدة
constants.js	# الثوابت
helpers.js	# دوال مساعدة
validation.js	# التحقق من البيانات
styles/	# ملفات التصميم
globals.css	# الأنماط العامة
App.jsx	# المكون الرئيسي
main.jsx	# نقطة دخول التطبيق
package.json	# تبعيات المشروع
vite.config.js	# تكوين Vite

## المكونات الرئيسية

### Header.jsx مكون

مكون رأس الصفحة يحتوي على شعار التطبيق وقائمة التنقل الرئيسية وأزرار تسجيل الدخول. المكون متجاوب ويتكيف مع أحجام الشاشات المختلفة.

```
import React, { useState } from 'react';
import { Link, useNavigate } from 'react-router-dom';
import { useAuth } from '../hooks/useAuth';

const Header = () => {
  const [isMenuOpen, setIsMenuOpen] = useState(false);
  const { user, logout } = useAuth();
  const navigate = useNavigate();

  const handleLogout = () => {
```

```

logout();
navigate('/');
};

return (
  <header className="bg-white shadow-lg sticky top-0 z-50">
    <div className="container mx-auto px-4">
      <div className="flex justify-between items-center py-4">
        {/* الشعار */}
        <Link to="/" className="flex items-center space-x-2 rtl:space-x-reverse">
          
          <span className="text-xl font-bold text-primary">مزااد الحيوانات النادرة</span>
        </Link>

        {/* قائمة التنقل */}
        <nav className="hidden md:flex space-x-8 rtl:space-x-reverse">
          <Link to="/" className="text-gray-700 hover:text-primary transition-colors">
            الرئيسية
          </Link>
          <Link to="/auctions" className="text-gray-700 hover:text-primary transition-colors">
            المزادات
          </Link>
          <Link to="/how-it-works" className="text-gray-700 hover:text-primary transition-colors">
            يعمل كيف
          </Link>
          <Link to="/contact" className="text-gray-700 hover:text-primary transition-colors">
            بنا اتصل
          </Link>
        </nav>

        {/* أزرار المستخدم */}
        <div className="flex items-center space-x-4 rtl:space-x-reverse">
          {user ? (
            <div className="flex items-center space-x-4 rtl:space-x-reverse">
              <Link to="/profile" className="text-gray-700 hover:text-primary">
                حسابي
              </Link>
              <button onClick={handleLogout} className="btn-secondary">
                الخروج تسجيل
              </button>
            </div>
          ) : (
            <div className="flex items-center space-x-4 rtl:space-x-reverse">
              <button onClick={handleLogin} className="btn-primary">
                تسجيل
              </button>
              <button onClick={handleRegister} className="btn-secondary">
                انشاء حساب
              </button>
            </div>
          )}
        </div>
      </div>
    </div>
  </header>
);

```

```

        </div>
      ) : (
        <div className="flex items-center space-x-2
rtl:space-x-reverse">
          <Link to="/login" className="btn-secondary">
            الدخول تسجيل
          </Link>
          <Link to="/register" className="btn-primary">
            حساب إنشاء
          </Link>
        </div>
      )}
    </div>
  </div>
</div>
</header>
);
};

export default Header;

```

## AnimalCard.jsx مكون

بطاقة عرض الحيوان تحتوي على الصورة والمعلومات الأساسية والسعر الحالي ووقت انتهاء المزاد.

```

import React from 'react';
import { Link } from 'react-router-dom';
import { formatPrice, formatTimeRemaining } from '../utils/
helpers';
import BidTimer from './BidTimer';

const AnimalCard = ({ animal }) => {
  const {
    _id,
    name,
    category,
    images,
    currentBid,
    auctionEndTime,
    status,
    bidCount
  } = animal;

  return (
    <div className="bg-white rounded-lg shadow-md overflow-
hidden hover:shadow-lg transition-shadow">
      { /* صورة الحيوان */ }
      <div className="relative h-48 bg-gray-200">
        {images && images.length > 0 ? (
          <img

```

```

        src={images[0]}
        alt={name}
        className="w-full h-full object-cover"
      />
    ) : (
      <div className="flex items-center justify-center h-
full text-gray-400">
        <span>صورة توجد لا</span>
      </div>
    )}

    {/* شارة الحالة */}
    <div className={`absolute top-2 right-2 px-2 py-1
rounded text-xs font-semibold ${
      status === 'active' ? 'bg-green-100 text-green-800' :
'bg-red-100 text-red-800'
    }`} >
      {status === 'active' ? 'نشط' : 'منتهي'}
    </div>
  </div>

  {/* معلومات الحيوان */}
  <div className="p-4">
    <div className="flex justify-between items-start mb-2">
      <h3 className="text-lg font-semibold text-gray-900
truncate">{name}</h3>
      <span className="text-sm text-gray-500 bg-gray-100
px-2 py-1 rounded">
        {category}
      </span>
    </div>

    {/* السعر الحالي */}
    <div className="mb-3">
      <span className="text-sm text-gray-600">المزايدة
الحالية</span>
      <div className="text-xl font-bold text-primary">
        {formatPrice(currentBid)} ل.د
      </div>
    </div>

    {/* معلومات المزاد */}
    <div className="flex justify-between items-center mb-4">
      <div className="text-sm text-gray-600">
        <span>{bidCount} مزايدة</span>
      </div>

      {status === 'active' && (
        <BidTimer endTime={auctionEndTime} />
      )}
    </div>
  </div>

```

```

    {/* أزرار العمل */}
    <div className="flex space-x-2 rtl:space-x-reverse">
      <Link
        to={`/animals/${_id}`}
        className="flex-1 btn-secondary text-center"
      >
        التفاصيل عرض
      </Link>

      {status === 'active' && (
        <Link
          to={`/animals/${_id}/bid`}
          className="flex-1 btn-primary text-center"
        >
          الآن زايد
        </Link>
      )}
    </div>
  </div>
</div>
);
};

export default AnimalCard;

```

## مكون BidTimer.jsx

عداد تنازلي يعرض الوقت المتبقي لانتهاء المزاد مع تحديث كل ثانية.

```

import React, { useState, useEffect } from 'react';

const BidTimer = ({ endTime }) => {
  const [timeRemaining, setTimeRemaining] = useState('');
  const [isExpired, setIsExpired] = useState(false);

  useEffect(() => {
    const updateTimer = () => {
      const now = new Date().getTime();
      const end = new Date(endTime).getTime();
      const difference = end - now;

      if (difference <= 0) {
        setTimeRemaining('انتهى');
        setIsExpired(true);
        return;
      }

      const hours = Math.floor(difference / (1000 * 60 * 60));
      const minutes = Math.floor((difference % (1000 * 60 *
60)) / (1000 * 60));

```

```

    const seconds = Math.floor((difference % (1000 * 60)) /
1000);

    setTimeRemaining(`${hours}:$
{minutes.toString().padStart(2, '0')}:$
{seconds.toString().padStart(2, '0')}`);
  };

  updateTimer();
  const interval = setInterval(updateTimer, 1000);

  return () => clearInterval(interval);
}, [endTime]);

return (
  <div className={`text-sm font-semibold ${
    isExpired ? 'text-red-600' : 'text-orange-600'
  }`} >
    <span className="block">الوقت المتبقي</span>
    <span className="text-lg">{timeRemaining}</span>
  </div>
);
};

export default BidTimer;

```

## Hooks إدارة الحالة وال

### useAuth Hook

مخصص لإدارة حالة التوثيق والمستخدم Hook.

```

import { useState, useEffect, createContext, useContext } from
'react';
import { authService } from '../services/auth';

const AuthContext = createContext();

export const AuthProvider = ({ children }) => {
  const [user, setUser] = useState(null);
  const [loading, setLoading] = useState(true);

  useEffect(() => {
    const token = localStorage.getItem('token');
    if (token) {
      authService.getProfile()
        .then(userData => setUser(userData))
        .catch(() => localStorage.removeItem('token'))
        .finally(() => setLoading(false));
    } else {

```

```

        setLoading(false);
    }
}, []);

const login = async (credentials) => {
    const response = await authService.login(credentials);
    localStorage.setItem('token', response.token);
    setUser(response.user);
    return response;
};

const logout = () => {
    localStorage.removeItem('token');
    setUser(null);
};

const value = {
    user,
    login,
    logout,
    loading
};

return (
    <AuthContext.Provider value={value}>
        {children}
    </AuthContext.Provider>
);
};

export const useAuth = () => {
    const context = useContext(AuthContext);
    if (!context) {
        throw new Error('useAuth must be used within AuthProvider');
    }
    return context;
};

```

## useSocket Hook

والتحديثات الفورية Socket.IO لإدارة اتصال Hook.

```

import { useEffect, useRef } from 'react';
import io from 'socket.io-client';

export const useSocket = (url, options = {}) => {
    const socketRef = useRef(null);

    useEffect(() => {
        socketRef.current = io(url, {

```

```

    ...options,
    transports: ['websocket']
  });

  return () => {
    if (socketRef.current) {
      socketRef.current.disconnect();
    }
  };
}, [url]);

const emit = (event, data) => {
  if (socketRef.current) {
    socketRef.current.emit(event, data);
  }
};

const on = (event, callback) => {
  if (socketRef.current) {
    socketRef.current.on(event, callback);
  }
};

const off = (event, callback) => {
  if (socketRef.current) {
    socketRef.current.off(event, callback);
  }
};

return { emit, on, off };
};

```

## التصميم والأنماط

مع تخصيصات للدعم الكامل للغة العربية. الألوان والخطوط مختارة Tailwind CSS التطبيق يستخدم بعناية لتوفير تجربة بصرية مريحة ومتسقة.

## Tailwind CSS تكوين

```

// tailwind.config.js
module.exports = {
  content: [
    './index.html',
    './src/**/*..{js,ts,jsx,tsx}',
  ],
  theme: {
    extend: {
      colors: {
        primary: {

```

```

        50: '#f0f9f4',
        100: '#dcf2e3',
        500: '#22c55e',
        600: '#16a34a',
        700: '#15803d',
    },
    secondary: {
        50: '#fef7ed',
        100: '#fdefd7',
        500: '#f97316',
        600: '#ea580c',
    }
},
fontFamily: {
    arabic: ['Cairo', 'Tajawal', 'sans-serif'],
},
direction: {
    rtl: 'rtl',
}
},
},
plugins: [
    require('@tailwindcss/forms'),
    require('@tailwindcss/typography'),
],
}

```

## الأنماط العامة

```

/* globals.css */
@import 'tailwindcss/base';
@import 'tailwindcss/components';
@import 'tailwindcss/utilities';

@import url('https://fonts.googleapis.com/css2?family=Cairo:wght@300;400;600;700&display=swap');

* {
    box-sizing: border-box;
    margin: 0;
    padding: 0;
}

html {
    direction: rtl;
    font-family: 'Cairo', sans-serif;
}

body {
    background-color: #f8fafc;
}

```

```

    color: #1e293b;
    line-height: 1.6;
}

/* مكونات مخصصة */
.btn-primary {
    @apply bg-primary-500 hover:bg-primary-600 text-white font-
semibold py-2 px-4 rounded-lg transition-colors duration-200;
}

.btn-secondary {
    @apply bg-gray-200 hover:bg-gray-300 text-gray-800 font-
semibold py-2 px-4 rounded-lg transition-colors duration-200;
}

.card {
    @apply bg-white rounded-lg shadow-md p-6;
}

.input-field {
    @apply w-full px-3 py-2 border border-gray-300 rounded-md
focus:outline-none focus:ring-2 focus:ring-primary-500
focus:border-transparent;
}

/* تحسينات للغة العربية */
.rtl {
    direction: rtl;
    text-align: right;
}

.ltr {
    direction: ltr;
    text-align: left;
}

```

## الأداء والتحسين

التطبيق محسن للأداء من خلال عدة تقنيات:

- **Code Splitting:** تقسيم الكود لتحميل أسرع
- **Lazy Loading:** تحميل المكونات عند الحاجة
- **Image Optimization:** ضغط وتحسين الصور
- **Caching:** تخزين مؤقت للبيانات والطلبات
- **Bundle Optimization:** تحسين حجم الملفات النهائية

## Lazy Loading مثال على

```
import { lazy, Suspense } from 'react';

const AnimalDetail = lazy(() => import('./pages/AnimalDetail'));
const Profile = lazy(() => import('./pages/Profile'));

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route
          path="/animals/:id"
          element={
            <Suspense fallback={<div>التحميل جاري</div>}>
              <AnimalDetail />
            </Suspense>
          }
        />
        <Route
          path="/profile"
          element={
            <Suspense fallback={<div>التحميل جاري</div>}>
              <Profile />
            </Suspense>
          }
        />
      </Routes>
    </Router>
  );
}
```

## دليل التطبيق الجوال

### نظرة عامة على التطبيق الجوال

مما يضمن التوافق، Expo مع React Native تطبيق مزاد الحيوانات النادرة الجوال مطور باستخدام التطبيق يوفر تجربة مستخدم سلسة ومتطورة مع دعم كامل للغة. Android و iOS الكامل مع أنظمة العربية والتحديثات الفورية للمزادات.

التطبيق مصمم ليكون سريع الاستجابة ومحسن للأداء على الأجهزة المحمولة، مع واجهة مستخدم بديهية تسهل على المستخدمين تصفح المزادات والمشاركة في المزادات وإدارة حساباتهم بكل سهولة.

## هيكل التطبيق الجوال

```
mobile-app/
├── App.js           # المكون الرئيسي
├── app.json         # تكوين Expo
├── package.json     # تبعيات المشروع
├── src/
│   ├── screens/    # شاشات التطبيق
│   │   ├── HomeScreen.js      # الشاشة الرئيسية
│   │   ├── LoginScreen.js     # شاشة تسجيل الدخول
│   │   ├── AuctionsScreen.js  # شاشة المزادات
│   │   ├── AnimalDetailScreen.js # تفاصيل الحيوان
│   │   ├── BiddingScreen.js   # شاشة المزايده
│   │   └── ProfileScreen.js   # الملف الشخصي
│   ├── components/ # المكونات القابلة لإعادة الاستخدام
│   │   ├── AnimalCard.js      # بطاقة الحيوان
│   │   ├── BidTimer.js        # عداد المزاد
│   │   ├── SearchBar.js       # شريط البحث
│   │   └── LoadingSpinner.js  # مؤشر التحميل
│   ├── navigation/ # إعدادات التنقل
│   │   └── AppNavigator.js    # التنقل الرئيسي
│   ├── services/      # خدمات التطبيق
│   │   ├── api.js       # طلبات API
│   │   ├── auth.js      # خدمات التوثيق
│   │   └── socket.js     # Socket.IO
│   ├── constants/     # الثوابت
│   │   └── theme.js     # ألوان وأنماط
│   └── utils/          # الأدوات المساعدة
│       ├── helpers.js   # دوال مساعدة
│       └── storage.js    # تخزين محلي
```

## الشاشات الرئيسية

### (HomeScreen) الشاشة الرئيسية

الشاشة الرئيسية تعرض نظرة عامة على المزادات النشطة والإحصائيات الأساسية مع إمكانية الوصول السريع للمميزات الرئيسية.

```
import React, { useState, useEffect } from 'react';
import {
  View,
  Text,
  ScrollView,
  RefreshControl,
  StyleSheet,
  TouchableOpacity
} from 'react-native';
import { SafeAreaView } from 'react-native-safe-area-context';
```

```

import { Ionicons } from '@expo/vector-icons';
import { COLORS, FONTS, SIZES } from '../constants/theme';
import AnimalCard from '../components/AnimalCard';
import { apiService } from '../services/api';

const HomeScreen = ({ navigation }) => {
  const [activeAuctions, setActiveAuctions] = useState([]);
  const [stats, setStats] = useState({});
  const [refreshing, setRefreshing] = useState(false);
  const [loading, setLoading] = useState(true);

  useEffect(() => {
    loadHomeData();
  }, []);

  const loadHomeData = async () => {
    try {
      const [auctionsResponse, statsResponse] = await
Promise.all([
      apiService.getActiveAuctions({ limit: 5 }),
      apiService.getStats()
    ]);

      setActiveAuctions(auctionsResponse.data);
      setStats(statsResponse.data);
    } catch (error) {
      console.error('Error loading home data:', error);
    } finally {
      setLoading(false);
      setRefreshing(false);
    }
  };

  const onRefresh = () => {
    setRefreshing(true);
    loadHomeData();
  };

  if (loading) {
    return (
      <SafeAreaView style={styles.container}>
        <View style={styles.loadingContainer}>
          <Text style={styles.loadingText}>التحميل جاري...</
Text>
        </View>
      </SafeAreaView>
    );
  }

  return (
    <SafeAreaView style={styles.container}>
      <ScrollView

```

```

        contentContainerStyle={styles.scrollContent}
        refreshControl={
          <RefreshControl refreshing={refreshing}
onRefresh={onRefresh} />
        }
      >
        { /* الرأس */ }
        <View style={styles.header}>
          <Text style={styles.welcomeText}>مرحباً في بك</Text>
          <Text style={styles.appTitle}>النادرة الحيوانات مزاد</
Text>
          <Text style={styles.subtitle}>في الحيوانات أندر اكتشف</
العالم</Text>
        </View>

        { /* الإحصائيات */ }
        <View style={styles.statsContainer}>
          <View style={styles.statCard}>
            <Icons name="hammer" size={24}
color={COLORS.primary} />
            <Text style={styles.statNumber}
>{stats.activeAuctions || 0}</Text>
            <Text style={styles.statLabel}>نشط مزاد</Text>
          </View>

          <View style={styles.statCard}>
            <Icons name="people" size={24}
color={COLORS.secondary} />
            <Text style={styles.statNumber}>{stats.totalUsers
|| 0}</Text>
            <Text style={styles.statLabel}>مستخدم</Text>
          </View>

          <View style={styles.statCard}>
            <Icons name="trophy" size={24}
color={COLORS.accent} />
            <Text style={styles.statNumber}
>{stats.completedAuctions || 0}</Text>
            <Text style={styles.statLabel}>مكتمل مزاد</Text>
          </View>
        </View>

        { /* أزرار الإجراءات السريعة */ }
        <View style={styles.quickActions}>
          <TouchableOpacity
            style={[styles.actionButton, { backgroundColor:
COLORS.primary }]}
            onPress={() => navigation.navigate('Auctions')}
          >
            <Icons name="search" size={20} color="white" />
            <Text style={styles.actionButtonText}>تصفح
المزادات</Text>

```

```

        </TouchableOpacity>

        <TouchableOpacity
            style={[styles.actionButton, { backgroundColor:
COLORS.secondary }]}
            onPress={() => navigation.navigate('AddAnimal')}
        >
            <Icons name="add-circle" size={20}
color="white" />
            <Text style={styles.actionButtonText}>أضف حيوان</
Text>
        </TouchableOpacity>
    </View>

    { /* المزايدات النشطة */ }
    <View style={styles.section}>
        <View style={styles.sectionHeader}>
            <Text style={styles.sectionTitle}>النشطة المزايدات</
Text>

            <TouchableOpacity onPress={() =>
navigation.navigate('Auctions')}>
                <Text style={styles.seeAllText}>عرض الكل</Text>
            </TouchableOpacity>
        </View>

        {activeAuctions.length > 0 ? (
            activeAuctions.map((animal) => (
                <AnimalCard
                    key={animal._id}
                    animal={animal}
                    onPress={() =>
navigation.navigate('AnimalDetail', { id: animal._id })}
                />
            ))
        ) : (
            <View style={styles.emptyState}>
                <Icons name="hourglass-outline" size={48}
color={COLORS.gray} />
                <Text style={styles.emptyStateText}>مزايدات توجد لا</Text>
                <Text>نشطة حالياً</Text>
            </View>
        )}
    </View>
</ScrollView>
</SafeAreaView>
);
};

const styles = StyleSheet.create({
    container: {
        flex: 1,
        backgroundColor: COLORS.background,

```

```
},
scrollContent: {
  paddingBottom: 20,
},
loadingContainer: {
  flex: 1,
  justifyContent: 'center',
  alignItems: 'center',
},
loadingText: {
  fontSize: SIZES.medium,
  color: COLORS.gray,
  fontFamily: FONTS.medium,
},
header: {
  padding: SIZES.large,
  alignItems: 'center',
  backgroundColor: COLORS.primary,
},
welcomeText: {
  fontSize: SIZES.medium,
  color: 'white',
  fontFamily: FONTS.regular,
},
appTitle: {
  fontSize: SIZES.xLarge,
  color: 'white',
  fontFamily: FONTS.bold,
  textAlign: 'center',
  marginVertical: 5,
},
subtitle: {
  fontSize: SIZES.small,
  color: 'rgba(255,255,255,0.8)',
  fontFamily: FONTS.regular,
  textAlign: 'center',
},
statsContainer: {
  flexDirection: 'row',
  justifyContent: 'space-around',
  paddingHorizontal: SIZES.medium,
  paddingVertical: SIZES.large,
},
statCard: {
  backgroundColor: 'white',
  padding: SIZES.medium,
  borderRadius: 12,
  alignItems: 'center',
  minWidth: 80,
  elevation: 2,
  shadowColor: '#000',
  shadowOffset: { width: 0, height: 2 },
}
```

```
    shadowOpacity: 0.1,
    shadowRadius: 4,
  },
  statNumber: {
    fontSize: SIZES.large,
    fontFamily: FONTS.bold,
    color: COLORS.dark,
    marginTop: 5,
  },
  statLabel: {
    fontSize: SIZES.small,
    fontFamily: FONTS.regular,
    color: COLORS.gray,
    marginTop: 2,
  },
  quickActions: {
    flexDirection: 'row',
    paddingHorizontal: SIZES.medium,
    gap: SIZES.small,
  },
  actionButton: {
    flex: 1,
    flexDirection: 'row',
    alignItems: 'center',
    justifyContent: 'center',
    padding: SIZES.medium,
    borderRadius: 12,
    gap: 8,
  },
  actionButtonText: {
    color: 'white',
    fontSize: SIZES.medium,
    fontFamily: FONTS.semiBold,
  },
  section: {
    padding: SIZES.medium,
  },
  sectionHeader: {
    flexDirection: 'row',
    justifyContent: 'space-between',
    alignItems: 'center',
    marginBottom: SIZES.medium,
  },
  sectionTitle: {
    fontSize: SIZES.large,
    fontFamily: FONTS.bold,
    color: COLORS.dark,
  },
  seeAllText: {
    fontSize: SIZES.medium,
    fontFamily: FONTS.medium,
    color: COLORS.primary,
```

```

    },
    emptyState: {
      alignItems: 'center',
      paddingVertical: SIZES.xLarge,
    },
    emptyStateText: {
      fontSize: SIZES.medium,
      color: COLORS.gray,
      fontFamily: FONTS.regular,
      marginTop: SIZES.small,
      textAlign: 'center',
    },
  },
});

```

```

export default HomeScreen;

```

## شاشة تسجيل الدخول (LoginScreen)

شاشة تسجيل الدخول توفر واجهة بسيطة وآمنة للمستخدمين للوصول إلى حساباتهم.

```

import React, { useState } from 'react';
import {
  View,
  Text,
  TextInput,
  TouchableOpacity,
  StyleSheet,
  Alert,
  KeyboardAvoidingView,
  Platform,
  ScrollView
} from 'react-native';
import { SafeAreaView } from 'react-native-safe-area-context';
import { Ionicons } from '@expo/vector-icons';
import { COLORS, FONTS, SIZES } from '../constants/theme';
import { authService } from '../services/auth';

const LoginScreen = ({ navigation }) => {
  const [formData, setFormData] = useState({
    email: '',
    password: ''
  });
  const [loading, setLoading] = useState(false);
  const [showPassword, setShowPassword] = useState(false);

  const handleInputChange = (field, value) => {
    setFormData(prev => ({
      ...prev,
      [field]: value
    }));
  };

```

```

};

const handleLogin = async () => {
  if (!formData.email || !formData.password) {
    Alert.alert('خطأ', 'يرجى ملء جميع الحقول');
    return;
  }

  setLoading(true);
  try {
    const response = await authService.login(formData);
    // التنقل للشاشة الرئيسية بعد تسجيل الدخول بنجاح
    navigation.replace('Main');
  } catch (error) {
    Alert.alert('خطأ في تسجيل الدخول', error.message || 'حدث خطأ غير متوقع');
  } finally {
    setLoading(false);
  }
};

return (
  <SafeAreaView style={styles.container}>
    <KeyboardAvoidingView
      behavior={Platform.OS === 'ios' ? 'padding' : 'height'}
      style={styles.keyboardView}
    >
      <ScrollView contentContainerStyle={styles.scrollContent}
    >
      </* الرأس */>
      <View style={styles.header}>
        <Icons name="paw" size={60}
color={COLORS.primary} />
        <Text style={styles.title}>بعودتك مرحباً</Text>
        <Text style={styles.subtitle}>للمتابعة دخولك سجل</Text>
      </View>

      </* نموذج تسجيل الدخول */>
      <View style={styles.form}>
        <View style={styles.inputContainer}>
          <Text style={styles.label}>البريد الإلكتروني</Text>
          <View style={styles.inputWrapper}>
            <Icons name="mail-outline" size={20}
color={COLORS.gray} />
            <TextInput
              style={styles.input}
              placeholder="أدخل بريدك الإلكتروني"
              value={formData.email}
              onChangeText={(value) =>
handleInputChange('email', value)}
              keyboardType="email-address"

```

```

        autoCapitalize="none"
        autoCorrect={false}
      />
    </View>
  </View>

  <View style={styles.inputContainer}>
    <Text style={styles.label}>المرور كلمة</Text>
    <View style={styles.inputWrapper}>
      <Icons name="lock-closed-outline" size={20}
color={COLORS.gray} />
      <TextInput
        style={styles.input}
        placeholder="أدخل كلمة المرور"
        value={formData.password}
        onChangeText={(value) =>
handleInputChange('password', value)}
        secureTextEntry={!showPassword}
        autoCapitalize="none"
      />
      <TouchableOpacity
        onPress={() => setShowPassword(!showPassword)}
        style={styles.eyeIcon}
      >
        <Icons
          name={showPassword ? 'eye-outline' : 'eye-
off-outline'}

          size={20}
          color={COLORS.gray}
        />
      </TouchableOpacity>
    </View>
  </View>

  <TouchableOpacity
    style={styles.forgotPassword}
    onPress={() =>
navigation.navigate('ForgotPassword')}
  >
    <Text style={styles.forgotPasswordText}>كلمة نسيت؟</Text>
  </TouchableOpacity>

  <TouchableOpacity
    style={[styles.loginButton, loading &&
styles.loginButtonDisabled]}
    onPress={handleLogin}
    disabled={loading}
  >
    <Text style={styles.loginButtonText}>
      {loading ? 'تسجيل الدخول...' : 'تسجيل
الدخول'}
    </Text>
  </TouchableOpacity>

```

```

        </Text>
      </TouchableOpacity>
    </View>

    { /* رابط إنشاء حساب */ }
    <View style={styles.footer}>
      <Text style={styles.footerText}>حساب لديك ليس</
Text>
      <TouchableOpacity onPress={() =>
navigation.navigate('Register')}>
        <Text style={styles.signupLink}>إنشاء حساب جديد</
Text>
      </TouchableOpacity>
    </View>
  </ScrollView>
</KeyboardAvoidingView>
</SafeAreaView>
);
};

const styles = StyleSheet.create({
  container: {
    flex: 1,
    backgroundColor: COLORS.background,
  },
  keyboardView: {
    flex: 1,
  },
  scrollContent: {
    flexGrow: 1,
    justifyContent: 'center',
    padding: SIZES.large,
  },
  header: {
    alignItems: 'center',
    marginBottom: SIZES.xLarge,
  },
  title: {
    fontSize: SIZES.xLarge,
    fontFamily: FONTS.bold,
    color: COLORS.dark,
    marginTop: SIZES.medium,
  },
  subtitle: {
    fontSize: SIZES.medium,
    fontFamily: FONTS.regular,
    color: COLORS.gray,
    marginTop: SIZES.small,
  },
  form: {
    marginBottom: SIZES.large,
  },
});

```

```
inputContainer: {
  marginBottom: SIZES.medium,
},
label: {
  fontSize: SIZES.medium,
  fontFamily: FONTS.medium,
  color: COLORS.dark,
  marginBottom: SIZES.small,
},
inputWrapper: {
  flexDirection: 'row',
  alignItems: 'center',
  backgroundColor: 'white',
  borderRadius: 12,
  paddingHorizontal: SIZES.medium,
  borderWidth: 1,
  borderColor: COLORS.lightGray,
},
input: {
  flex: 1,
  paddingVertical: SIZES.medium,
  paddingHorizontal: SIZES.small,
  fontSize: SIZES.medium,
  fontFamily: FONTS.regular,
  textAlign: 'right',
},
eyeIcon: {
  padding: SIZES.small,
},
forgotPassword: {
  alignSelf: 'flex-end',
  marginBottom: SIZES.large,
},
forgotPasswordText: {
  fontSize: SIZES.small,
  fontFamily: FONTS.medium,
  color: COLORS.primary,
},
loginButton: {
  backgroundColor: COLORS.primary,
  paddingVertical: SIZES.medium,
  borderRadius: 12,
  alignItems: 'center',
},
loginButtonDisabled: {
  backgroundColor: COLORS.gray,
},
loginButtonText: {
  fontSize: SIZES.medium,
  fontFamily: FONTS.semiBold,
  color: 'white',
},
```

```

    footer: {
      flexDirection: 'row',
      justifyContent: 'center',
      alignItems: 'center',
    },
    footerText: {
      fontSize: SIZES.medium,
      fontFamily: FONTS.regular,
      color: COLORS.gray,
    },
    signupLink: {
      fontSize: SIZES.medium,
      fontFamily: FONTS.semiBold,
      color: COLORS.primary,
    },
  });

export default LoginScreen;

```

## المكونات القابلة لإعادة الاستخدام

### AnimalCard مكون

```

import React from 'react';
import {
  View,
  Text,
  Image,
  TouchableOpacity,
  StyleSheet
} from 'react-native';
import { Ionicons } from '@expo/vector-icons';
import { COLORS, FONTS, SIZES } from '../constants/theme';
import { formatPrice, formatTimeRemaining } from '../utils/helpers';

const AnimalCard = ({ animal, onPress }) => {
  const {
    name,
    category,
    images,
    currentBid,
    auctionEndTime,
    status,
    bidCount
  } = animal;

  return (
    <TouchableOpacity style={styles.container} onPress={onPress}>

```

```

    { /* صورة الحيوان */ }
    <View style={styles.imageContainer}>
      {images && images.length > 0 ? (
        <Image source={{ uri: images[0] }}
style={styles.image} />
      ) : (
        <View style={styles.placeholderImage}>
          <Icons name="image-outline" size={40}
color={COLORS.gray} />
        </View>
      )}

    { /* شارة الحالة */ }
    <View style={[
      styles.statusBadge,
      { backgroundColor: status === 'active' ?
COLORS.success : COLORS.error }
    ]}>
      <Text style={styles.statusText}>
        {status === 'active' ? 'نشط' : 'منتهي'}
      </Text>
    </View>
  </View>

  { /* معلومات الحيوان */ }
  <View style={styles.content}>
    <View style={styles.header}>
      <Text style={styles.name} numberOfLines={1}>{name}</
Text>

      <Text style={styles.category}>{category}</Text>
    </View>

    <View style={styles.bidInfo}>
      <Text style={styles.bidLabel}>المزايدة</Text>
      <Text style={styles.bidAmount}>
        >{formatPrice(currentBid)} ل.د.</Text>
    </View>

    <View style={styles.footer}>
      <View style={styles.bidCount}>
        <Icons name="people-outline" size={16}
color={COLORS.gray} />
        <Text style={styles.bidCountText}>{bidCount}</Text>
      </View>

      {status === 'active' && (
        <View style={styles.timeRemaining}>
          <Icons name="time-outline" size={16}
color={COLORS.warning} />
          <Text style={styles.timeText}>
            {formatTimeRemaining(auctionEndTime)}

```

```

        </Text>
      </View>
    })
  </View>
</View>
</TouchableOpacity>
);
};

```

```

const styles = StyleSheet.create({
  container: {
    backgroundColor: 'white',
    borderRadius: 12,
    marginBottom: SIZES.medium,
    elevation: 3,
    shadowColor: '#000',
    shadowOffset: { width: 0, height: 2 },
    shadowOpacity: 0.1,
    shadowRadius: 4,
  },
  imageContainer: {
    position: 'relative',
    height: 200,
    borderTopLeftRadius: 12,
    borderTopRightRadius: 12,
    overflow: 'hidden',
  },
  image: {
    width: '100%',
    height: '100%',
    resizeMode: 'cover',
  },
  placeholderImage: {
    width: '100%',
    height: '100%',
    backgroundColor: COLORS.lightGray,
    justifyContent: 'center',
    alignItems: 'center',
  },
  statusBadge: {
    position: 'absolute',
    top: 10,
    right: 10,
    paddingHorizontal: 8,
    paddingVertical: 4,
    borderRadius: 6,
  },
  statusText: {
    fontSize: SIZES.small,
    fontFamily: FONTS.semiBold,
    color: 'white',
  },
});

```

```
content: {
  padding: SIZES.medium,
},
header: {
  flexDirection: 'row',
  justifyContent: 'space-between',
  alignItems: 'flex-start',
  marginBottom: SIZES.small,
},
name: {
  flex: 1,
  fontSize: SIZES.medium,
  fontFamily: FONTS.semiBold,
  color: COLORS.dark,
},
category: {
  fontSize: SIZES.small,
  fontFamily: FONTS.regular,
  color: COLORS.gray,
  backgroundColor: COLORS.lightGray,
  paddingHorizontal: 8,
  paddingVertical: 4,
  borderRadius: 6,
  marginLeft: SIZES.small,
},
bidInfo: {
  marginBottom: SIZES.small,
},
bidLabel: {
  fontSize: SIZES.small,
  fontFamily: FONTS.regular,
  color: COLORS.gray,
},
bidAmount: {
  fontSize: SIZES.large,
  fontFamily: FONTS.bold,
  color: COLORS.primary,
},
footer: {
  flexDirection: 'row',
  justifyContent: 'space-between',
  alignItems: 'center',
},
bidCount: {
  flexDirection: 'row',
  alignItems: 'center',
  gap: 4,
},
bidCountText: {
  fontSize: SIZES.small,
  fontFamily: FONTS.regular,
  color: COLORS.gray,
```

```

    },
    timeRemaining: {
      flexDirection: 'row',
      alignItems: 'center',
      gap: 4,
    },
    timeText: {
      fontSize: SIZES.small,
      fontFamily: FONTS.medium,
      color: COLORS.warning,
    },
  }));

export default AnimalCard;

```

## إدارة الحالة والتخزين

لتخزين البيانات محلياً مثل رمز التوثيق وإعدادات المستخدم AsyncStorage التطبيق يستخدم

```

// utils/storage.js
import AsyncStorage from '@react-native-async-storage/async-storage';

export const storage = {
  // حفظ البيانات
  setItem: async (key, value) => {
    try {
      const jsonValue = JSON.stringify(value);
      await AsyncStorage.setItem(key, jsonValue);
    } catch (error) {
      console.error('Error saving data:', error);
    }
  },

  // استرجاع البيانات
  getItem: async (key) => {
    try {
      const jsonValue = await AsyncStorage.getItem(key);
      return jsonValue !== null ? JSON.parse(jsonValue) : null;
    } catch (error) {
      console.error('Error retrieving data:', error);
      return null;
    }
  },

  // حذف البيانات
  removeItem: async (key) => {
    try {
      await AsyncStorage.removeItem(key);
    } catch (error) {

```

```

        console.error('Error removing data:', error);
    }
},

// مسح جميع البيانات
clear: async () => {
    try {
        await AsyncStorage.clear();
    } catch (error) {
        console.error('Error clearing storage:', error);
    }
}
};

// مفاتيح التخزين
export const STORAGE_KEYS = {
    AUTH_TOKEN: 'auth_token',
    USER_DATA: 'user_data',
    SETTINGS: 'app_settings',
    FAVORITES: 'favorite_animals'
};

```

## التنقل والتوجيه

للتنقل بين الشاشات مع دعم كامل للغة العربية React Navigation يستخدم

```

// navigation/AppNavigator.js
import React from 'react';
import { NavigationContainer } from '@react-navigation/native';
import { createBottomTabNavigator } from '@react-navigation/
bottom-tabs';
import { createStackNavigator } from '@react-navigation/stack';
import { Ionicons } from '@expo/vector-icons';
import { COLORS } from '../constants/theme';

import HomeScreen from '../screens/HomeScreen';
import AuctionsScreen from '../screens/AuctionsScreen';
import ProfileScreen from '../screens/ProfileScreen';
import LoginScreen from '../screens/LoginScreen';
import AnimalDetailScreen from '../screens/AnimalDetailScreen';

const Tab = createBottomTabNavigator();
const Stack = createStackNavigator();

const MainTabs = () => {
    return (
        <Tab.Navigator
            screenOptions={({ route }) => ({
                tabBarIcon: ({ focused, color, size }) => {
                    let iconName;

```

```

        if (route.name === 'Home') {
            iconName = focused ? 'home' : 'home-outline';
        } else if (route.name === 'Auctions') {
            iconName = focused ? 'hammer' : 'hammer-outline';
        } else if (route.name === 'Profile') {
            iconName = focused ? 'person' : 'person-outline';
        }

        return <Icons name={iconName} size={size}
color={color} />;
    },
    tabBarActiveTintColor: COLORS.primary,
    tabBarInactiveTintColor: COLORS.gray,
    tabBarStyle: {
        backgroundColor: 'white',
        borderTopWidth: 1,
        borderTopColor: COLORS.lightGray,
        height: 60,
        paddingBottom: 8,
        paddingTop: 8,
    },
    headerShown: false,
  )}
>
  <Tab.Screen
    name="Home"
    component={HomeScreen}
    options={{ tabBarLabel: 'الرئيسية' }}
  />
  <Tab.Screen
    name="Auctions"
    component={AuctionsScreen}
    options={{ tabBarLabel: 'المزادات' }}
  />
  <Tab.Screen
    name="Profile"
    component={ProfileScreen}
    options={{ tabBarLabel: 'حسابي' }}
  />
</Tab.Navigator>
);
};

const AppNavigator = () => {
  return (
    <NavigationContainer>
      <Stack.Navigator
        screenOptions={{
          headerShown: false,
          gestureEnabled: true,
          gestureDirection: 'horizontal-inverted', // للدعم RTL

```

```

    }}
  >
  <Stack.Screen name="Login" component={LoginScreen} />
  <Stack.Screen name="Main" component={MainTabs} />
  <Stack.Screen
    name="AnimalDetail"
    component={AnimalDetailScreen}
    options={{
      headerShown: true,
      title: 'تفاصيل الحيوان',
      headerTitleAlign: 'center',
    }}
  />
</Stack.Navigator>
</NavigationContainer>
);
};

export default AppNavigator;

```

## الأداء والتحسين

التطبيق محسن للأداء من خلال:

- **Lazy Loading:** تحميل الشاشات عند الحاجة
- **Image Caching:** تخزين مؤقت للصور
- **State Management:** إدارة فعالة للحالة
- **Memory Management:** إدارة الذاكرة وتنظيف الموارد
- **Network Optimization:** تحسين طلبات الشبكة

### مثال على تحسين الصور

```

import { Image } from 'expo-image';

const OptimizedImage = ({ source, style, ...props }) => {
  return (
    <Image
      source={source}
      style={style}
      contentFit="cover"
      transition={200}
      cachePolicy="memory-disk"
      {...props}
    />
  );
};

```