🧾 Arabic Invoice Generator API - MVP

Professional bilingual invoice generator for freelancers and businesses

🌟 Features

✅ **Bilingual PDF Generation** - Professional invoices in Arabic and English
✅ **JWT Authentication** - Secure user authentication
✅ **Email Sending** - Send invoices directly to clients
✅ **QR Code Integration** - Generate QR codes for easy payment
✅ **Multi-Currency Support** - MAD, USD, EUR, SAR, AED
✅ **Payment Links** - Unique payment links per user
✅ **RESTful API** - Clean and documented API
✅ **RapidAPI Ready** - Ready to publish on RapidAPI

🚀 Quick Start

Prerequisites

- Python 3.11+
- pip

Installation (2 minutes)

```bash
# Clone repository
git clone https://github.com/yourusername/invoice-api.git
cd invoice-api

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt

# Configure environment
cp .env.example .env
# Edit .env with your settings

# Initialize database
python -c "from app.database import init_db; init_db()"

# Run the API
uvicorn app.main:app --reload
```

🎉 **API is running at:** http://localhost:8000
📚 **Documentation:** http://localhost:8000/docs

📚 Documentation

- **Quick Start Guide** - Get started in 10 minutes
- **API Documentation** - Complete API reference
- **Deployment Guide** - Production deployment
- **Postman Collection** - Import and test

🎯 Example Usage

1. Register & Login

```bash
# Register
curl -X POST http://localhost:8000/auth/register \
  -H "Content-Type: application/json" \
  -d '{
    "email": "john@example.com",
    "username": "john_doe",
    "password": "securepass123",
    "company_name": "John Consulting"
  }'

# Login
curl -X POST http://localhost:8000/auth/login \
  -H "Content-Type: application/json" \
  -d '{
    "username": "john_doe",
    "password": "securepass123"
  }'
```

2. Create Invoice

```bash
TOKEN="your_jwt_token"

curl -X POST http://localhost:8000/invoices/generate \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "client_name": "ACME Corp",
    "client_email": "billing@acme.com",
    "language": "ar",
    "currency": "MAD",
    "items": [
      {
        "name": "Web Development",
        "quantity": 1,
        "price": 15000
      }
    ],
    "tax_rate": 20
  }'
```

3. Send Invoice via Email

```bash
curl -X POST http://localhost:8000/invoices/1/send-email \
  -H "Authorization: Bearer $TOKEN"
```

📁 Project Structure

```
invoice-api/
├── app/
│   ├── main.py              # FastAPI application
│   ├── config.py            # Configuration
│   ├── database.py          # Database setup
│   ├── models/              # SQLAlchemy models
│   ├── schemas/             # Pydantic schemas
│   ├── api/                 # API endpoints
│   ├── services/            # Business logic
│   ├── templates/           # Invoice templates
│   └── utils/               # Helper functions
├── tests/                   # Unit tests
├── static/                  # Generated files
├── requirements.txt         # Dependencies
├── .env.example             # Environment template
└── docker-compose.yml       # Docker configuration
```

🛠️ Tech Stack

- **Framework:** FastAPI 0.109.0
- **Database:** SQLAlchemy + SQLite/PostgreSQL
- **PDF Generation:** WeasyPrint + Jinja2
- **QR Codes:** qrcode + segno
- **Email:** aiosmtplib
- **Authentication:** JWT (python-jose)
- **Validation:** Pydantic

🌐 API Endpoints

Authentication

- `POST /auth/register` - Register new user
- `POST /auth/login` - Login and get JWT

Invoices

- `POST /invoices/generate` - Create new invoice
- `GET /invoices/{id}` - Get invoice by ID
- `GET /invoices/` - List all invoices
- `GET /invoices/{id}/download` - Download PDF
- `POST /invoices/{id}/send-email` - Send via email
- `PUT /invoices/{id}` - Update invoice
- `DELETE /invoices/{id}` - Delete invoice

Users

- `GET /users/me` - Get user profile
- `PUT /users/me` - Update profile
- `GET /users/me/stats` - Get statistics

🔐 Environment Variables

env

```
# Database
DATABASE_URL=sqlite:///./invoices.db

# Security
SECRET_KEY=your-super-secret-key
ALGORITHM=HS256
ACCESS_TOKEN_EXPIRE_MINUTES=30

# Email (SendGrid example)
EMAIL_HOST=smtp.sendgrid.net
EMAIL_PORT=587
EMAIL_USERNAME=apikey
EMAIL_PASSWORD=your-api-key
EMAIL_FROM=noreply@yourdomain.com

# App
DEBUG=true
ALLOWED_ORIGINS=http://localhost:3000
```

🐳 Docker Deployment

```bash
# Build and run
docker-compose up -d

# View logs
docker-compose logs -f

# Stop
docker-compose down
```

🧪 Testing

```bash
# Run all tests
pytest

# With coverage
pytest --cov=app tests/

# Specific test file
pytest tests/test_invoices.py -v
```

📊 Features Roadmap

✅ MVP (Current)

- [x] Bilingual PDF generation
- [x] JWT authentication
- [x] Email sending
- [x] QR codes
- [x] Multi-currency

🔜 Phase 2

- [ ] Payment gateway integration (Stripe, PayPal)
- [ ] Webhook notifications
- [ ] Recurring invoices
- [ ] Invoice templates customization
- [ ] Multi-language expansion

🔮 Phase 3

- [ ] Mobile SDK
- [ ] Advanced analytics
- [ ] Team collaboration
- [ ] API rate limiting tiers
- [ ] White-label solution

💰 Pricing Tiers (Suggested)

Tier	Invoices/Month	Price	Email Sending
Free	10	$0	5/day
Basic	100	$9.99	50/day
Pro	500	$29.99	200/day
Enterprise	Unlimited	Custom	Unlimited

🤝 Contributing

Contributions are welcome! Please follow these steps:

1. Fork the repository
2. Create feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit changes (`git commit -m 'Add AmazingFeature'`)
4. Push to branch (`git push origin feature/AmazingFeature`)
5. Open Pull Request

📄 License

This project is licensed under the MIT License - see the LICENSE file for details.

🙏 Acknowledgments

- FastAPI for the amazing framework
- WeasyPrint for PDF generation
- The open-source community

📞 Support

- **Documentation:** Full Docs
- **Email:** support@yourdomain.com
- **Issues:** GitHub Issues
- **Discord:** Join our community

🌟 Show Your Support

If this project helped you, give it a ⭐ on GitHub!

Made with ❤️ for freelancers and small businesses in MENA

Get Started | API Docs | Deploy

```
invoice-api/
├── app/
│   ├── __init__.py
│   ├── main.py              # FastAPI app entry point
│   ├── config.py            # Configuration & environment variables
│   ├── database.py          # Database connection
│   │
│   ├── models/
│   │   ├── __init__.py
│   │   ├── user.py          # User database model
│   │   └── invoice.py       # Invoice database model
│   │
│   ├── schemas/
│   │   ├── __init__.py
│   │   ├── user.py          # User Pydantic schemas
│   │   ├── invoice.py       # Invoice Pydantic schemas
│   │   └── auth.py          # Authentication schemas
│   │
│   ├── api/
│   │   ├── __init__.py
│   │   ├── auth.py          # Authentication endpoints
│   │   ├── invoices.py      # Invoice CRUD endpoints
│   │   └── users.py         # User management endpoints
│   │
│   ├── services/
│   │   ├── __init__.py
│   │   ├── auth_service.py  # JWT & password handling
│   │   ├── pdf_service.py   # PDF generation
│   │   ├── qr_service.py    # QR code generation
│   │   └── email_service.py # Email sending
│   │
│   ├── templates/
│   │   ├── invoice_ar.html  # Arabic invoice template
│   │   └── invoice_en.html  # English invoice template
│   │
│   └── utils/
│       ├── __init__.py
│       ├── dependencies.py  # FastAPI dependencies
│       └── helpers.py       # Helper functions
│
├── tests/
│   ├── __init__.py
│   ├── test_auth.py
│   ├── test_invoices.py
│   └── test_services.py
│
├── alembic/                 # Database migrations
│   ├── versions/
│   └── env.py
```

```
│
├── static/                 # Generated PDFs & QR codes
│   ├── invoices/
│   └── qr_codes/
│
├── .env.example            # Environment variables template
├── .gitignore
├── requirements.txt
├── alembic.ini
├── README.md
└── docker-compose.yml      # Optional: for deployment
```

🚀 Quick Start

1. Installation

```bash
# Clone the repository
git clone <your-repo-url>
cd invoice-api

# Create virtual environment
python -m venv venv
source venv/bin/activate  # On Windows: venv\Scripts\activate

# Install dependencies
pip install -r requirements.txt
```

2. Configuration

```bash
# Copy environment template
cp .env.example .env

# Edit .env with your settings
nano .env
```

3. Database Setup

```bash
# Initialize database
alembic upgrade head
```

4. Run the API

```bash
# Development mode
uvicorn app.main:app --reload --host 0.0.0.0 --port 8000

# Production mode
uvicorn app.main:app --host 0.0.0.0 --port 8000 --workers 4
```

5. Access API Documentation

- **Swagger UI**: http://localhost:8000/docs
- **ReDoc**: http://localhost:8000/redoc

🔑 Environment Variables

```env
# Database
DATABASE_URL=sqlite:///./invoices.db
# For PostgreSQL: postgresql://user:password@localhost/dbname

# Security
SECRET_KEY=your-super-secret-key-change-this-in-production
ALGORITHM=HS256
ACCESS_TOKEN_EXPIRE_MINUTES=30

# Email (SendGrid example)
EMAIL_HOST=smtp.sendgrid.net
EMAIL_PORT=587
EMAIL_USERNAME=apikey
EMAIL_PASSWORD=your-sendgrid-api-key
EMAIL_FROM=noreply@yourdomain.com

# App Settings
APP_NAME=Invoice Generator API
APP_VERSION=1.0.0
DEBUG=true
```

📚 API Endpoints

Authentication

- `POST /auth/register` - Register new user
- `POST /auth/login` - Login and get JWT token
- `POST /auth/refresh` - Refresh access token

Invoices

- `POST /invoices/generate` - Create new invoice
- `GET /invoices/{invoice_id}` - Get invoice by ID
- `GET /invoices/` - List all user invoices
- `GET /invoices/{invoice_id}/download` - Download PDF
- `POST /invoices/{invoice_id}/send-email` - Send invoice via email

Users

- `GET /users/me` - Get current user info
- `PUT /users/me` - Update user profile

🧪 Testing

```bash
# Run all tests
pytest

# Run with coverage
pytest --cov=app tests/

# Run specific test file
pytest tests/test_invoices.py -v
```

🐳 Docker Deployment

```bash
# Build and run with Docker Compose
docker-compose up -d

# View logs
docker-compose logs -f

# Stop services
docker-compose down
```

📖 Features

✅ MVP Features (Stage 1)

- [x] Bilingual PDF generation (Arabic + English)
- [x] JWT Authentication
- [x] Invoice CRUD operations
- [x] QR Code generation
- [x] Unique payment links
- [x] Email sending capability
- [x] PDF download
- [x] RapidAPI ready

🔮 Future Features

- [ ] Payment gateway integration (Stripe, PayPal)
- [ ] Multi-language support
- [ ] Subscription tiers
- [ ] Advanced analytics
- [ ] Webhook notifications

🤝 Contributing

1. Fork the repository
2. Create feature branch (`git checkout -b feature/AmazingFeature`)
3. Commit changes (`git commit -m 'Add some AmazingFeature'`)
4. Push to branch (`git push origin feature/AmazingFeature`)
5. Open Pull Request

📄 License

MIT License - feel free to use for commercial projects

💬 Support

- Documentation: `/docs`
- Email: support@yourdomain.com
- Issues: GitHub Issues

Made with ❤️ for freelancers and small businesses in MENA