📚 Invoice Generator API - Complete Documentation

🚀 Base URL

```
Development: http://localhost:8000
Production: https://api.yourdomain.com
```

🔐 Authentication

All endpoints (except `/auth/register` and `/auth/login`) require JWT authentication.

Headers

```http
Authorization: Bearer {your_jwt_token}
Content-Type: application/json
```

📋 API Endpoints

🔑 Authentication

1. Register User

POST `/auth/register`

Create a new user account.

Request Body:

```json
{
  "email": "user@example.com",
  "username": "john_doe",
  "password": "securepass123",
  "full_name": "John Doe",
  "company_name": "Doe Consulting",
  "phone": "+212600000000",
  "address": "123 Main St, Casablanca"
}
```

Response: `201 Created`

```json
{
  "id": 1,
  "email": "user@example.com",
  "username": "john_doe",
  "full_name": "John Doe",
  "company_name": "Doe Consulting",
  "phone": "+212600000000",
  "address": "123 Main St, Casablanca",
  "payment_link": "https://pay.yourdomain.com/pay/john_doe-1",
  "is_active": true,
  "is_verified": false
}
```

Errors:

- `400`: Email or username already exists

---

2. Login

POST `/auth/login`

Login and receive JWT access token.

Request Body:

```json
{
  "username": "john_doe",
  "password": "securepass123"
}
```

Response: `200 OK`

```json
{
  "access_token": "eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...",
  "token_type": "bearer",
  "expires_in": 1800
}
```

Errors:

- `401`: Invalid credentials
- `403`: User account inactive

🧾 Invoices

3. Create Invoice

POST `/invoices/generate`

Generate new invoice with PDF and QR code.

Request Body:

```json
{
  "client_name": "ACME Corporation",
  "client_email": "billing@acme.com",
  "client_phone": "+212600111222",
  "client_address": "456 Business Ave, Rabat",
  "language": "ar",
  "currency": "MAD",
  "items": [
    {
      "name": "Web Development",
      "description": "E-commerce website development",
      "quantity": 1,
      "price": 15000
    },
    {
      "name": "SEO Optimization",
      "description": "3-month SEO package",
      "quantity": 3,
      "price": 2000
    }
  ],
  "tax_rate": 20,
  "discount_rate": 10,
  "due_date": "2025-11-01T00:00:00",
  "notes": "Payment due within 30 days. Bank transfer preferred."
}
```

Response: `201 Created`

json

```json
{
  "id": 1,
  "invoice_number": "INV-20251002-1234",
  "user_id": 1,
  "client_name": "ACME Corporation",
  "client_email": "billing@acme.com",
  "client_phone": "+212600111222",
  "client_address": "456 Business Ave, Rabat",
  "language": "ar",
  "currency": "MAD",
  "items": [
    {
      "name": "Web Development",
      "description": "E-commerce website development",
      "quantity": 1,
      "price": 15000,
      "total": 15000
    },
    {
      "name": "SEO Optimization",
      "description": "3-month SEO package",
      "quantity": 3,
      "price": 2000,
      "total": 6000
    }
  ],
  "subtotal": 21000,
  "tax_rate": 20,
  "tax_amount": 3780,
  "discount_rate": 10,
  "discount_amount": 2100,
  "total": 22680,
  "issue_date": "2025-10-02T10:30:00",
  "due_date": "2025-11-01T00:00:00",
  "pdf_path": "./static/invoices/invoice_INV-20251002-1234.pdf",
  "qr_code_path": "./static/qr_codes/invoice_INV-20251002-1234_qr.png",
  "payment_link": "https://pay.yourdomain.com/pay/john_doe-1?invoice=INV-20251002-1234",
  "status": "draft",
  "is_sent_email": false,
  "email_sent_at": null,
  "notes": "Payment due within 30 days. Bank transfer preferred.",
  "created_at": "2025-10-02T10:30:00",
  "updated_at": "2025-10-02T10:30:00"
}
```

Validation:

- `client_name`: Required, min 1 char
- `client_email`: Required, valid email
- `items`: Required, min 1 item
- `quantity`: Must be > 0
- `price`: Must be >= 0
- `tax_rate`: 0-100
- `discount_rate`: 0-100

---

4. Get Invoice

GET `/invoices/{invoice_id}`

Retrieve specific invoice by ID.

Response: `200 OK`

```json
{
  "id": 1,
  "invoice_number": "INV-20251002-1234",
  ...
}
```

Errors:

- `404`: Invoice not found

---

5. List Invoices

GET `/invoices?page=1&page_size=10&status=sent`

List all user's invoices with pagination.

Query Parameters:

- `page`: Page number (default: 1)
- `page_size`: Items per page (default: 10, max: 100)
- `status`: Filter by status - draft, sent, paid, cancelled

Response: `200 OK`

```json
{
  "invoices": [
    {
      "id": 1,
      "invoice_number": "INV-20251002-1234",
      ...
    },
    {
      "id": 2,
      "invoice_number": "INV-20251001-5678",
      ...
    }
  ],
  "total": 25,
  "page": 1,
  "page_size": 10
}
```

---

6. Download Invoice PDF

GET `/invoices/{invoice_id}/download`

Download invoice as PDF file.

Response: `200 OK`

- Content-Type: `application/pdf`
- File: `invoice_{number}.pdf`

Errors:

- `404`: Invoice or PDF not found

---

7. Send Invoice by Email

POST `/invoices/{invoice_id}/send-email`

Send invoice PDF to client via email.

Rate Limit: 5 emails per hour per user

Request Body (Optional):

```json
{
  "to_email": "override@example.com",
  "subject": "Custom subject",
  "message": "Custom message to include in email"
}
```

Response: `200 OK`

```json
{
  "message": "Email is being sent",
  "recipient": "billing@acme.com",
  "invoice_number": "INV-20251002-1234"
}
```

Errors:

- `404`: Invoice not found
- `400`: PDF not generated
- `429`: Rate limit exceeded

---

8. Update Invoice

PUT `/invoices/{invoice_id}`

Update invoice details (client info, status, notes, due date).

Request Body:

```json
{
  "client_name": "Updated Name",
  "status": "paid",
  "notes": "Updated notes",
  "due_date": "2025-12-01T00:00:00"
}
```

Response: `200 OK` (Returns updated invoice)

Errors:

- `404`: Invoice not found

---

9. Delete Invoice

DELETE `/invoices/{invoice_id}`

Permanently delete invoice and associated files.

Response: `204 No Content`

Errors:

- `404`: Invoice not found

👤 User Management

10. Get Current User

GET `/users/me`

Get current user's profile information.

Response: `200 OK`

```json
{
  "id": 1,
  "email": "user@example.com",
  "username": "john_doe",
  "full_name": "John Doe",
  "company_name": "Doe Consulting",
  "phone": "+212600000000",
  "address": "123 Main St, Casablanca",
  "payment_link": "https://pay.yourdomain.com/pay/john_doe-1",
  "is_active": true,
  "is_verified": false
}
```

---

11. Update User Profile

PUT `/users/me`

Update user profile information.

Request Body:

```json
{
  "full_name": "John Updated Doe",
  "company_name": "New Company Name",
  "phone": "+212611222333",
  "address": "New Address",
  "email": "newemail@example.com"
}
```

Response: `200 OK` (Returns updated user)

Errors:

- `400`: Email already registered

---

12. Get User Statistics

GET `/users/me/stats`

Get user's invoice statistics.

Response: `200 OK`

```json
{
  "total_invoices": 25,
  "status_breakdown": {
    "draft": 5,
    "sent": 12,
    "paid": 7,
    "cancelled": 1
  },
  "total_revenue": 125000.00,
  "pending_amount": 45000.00,
  "payment_link": "https://pay.yourdomain.com/pay/john_doe-1"
}
```

📝 Example Workflows

Complete Invoice Creation Flow

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

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

# Save the token
TOKEN="your_jwt_token_here"

# 3. Create Invoice
curl -X POST http://localhost:8000/invoices/generate \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json" \
  -d '{
    "client_name": "Client Corp",
    "client_email": "client@example.com",
    "language": "ar",
    "currency": "MAD",
    "items": [
      {
        "name": "Consulting Services",
        "quantity": 10,
        "price": 500
      }
    ],
    "tax_rate": 20
  }'

# 4. Send Email
curl -X POST http://localhost:8000/invoices/1/send-email \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json"

# 5. Download PDF
curl -X GET http://localhost:8000/invoices/1/download \
  -H "Authorization: Bearer $TOKEN" \
  -H "Authorization: Bearer $TOKEN" \
  --output invoice.pdf
```

🌐 Supported Languages & Currencies

Languages

- `ar`: Arabic (RTL)
- `en`: English (LTR)

Currencies

- `MAD`: Moroccan Dirham
- `USD`: US Dollar
- `EUR`: Euro
- `SAR`: Saudi Riyal
- `AED`: UAE Dirham

⚠️ Error Responses

All errors follow this format:

```json
{
  "detail": "Error message description"
}
```

HTTP Status Codes

- `200`: Success
- `201`: Created
- `204`: No Content (successful deletion)
- `400`: Bad Request (validation error)
- `401`: Unauthorized (invalid/missing token)
- `403`: Forbidden (inactive account)
- `404`: Not Found
- `429`: Too Many Requests (rate limit)
- `500`: Internal Server Error

🔒 Security Best Practices

1. **Always use HTTPS** in production
2. **Keep JWT tokens secure** - never expose in URLs
3. **Tokens expire after 30 minutes** - implement refresh logic
4. **Rate limits apply** - max 5 emails per hour
5. **Use strong passwords** - min 8 characters

📊 Rate Limits

Endpoint	Limit
Email Sending	5 per hour per user
API Requests	No limit (MVP)

🛠️ Integration Examples

Python

```python
import requests

API_URL = "http://localhost:8000"

# Login
response = requests.post(f"{API_URL}/auth/login", json={
    "username": "user",
    "password": "pass"
})
token = response.json()["access_token"]

# Create invoice
headers = {"Authorization": f"Bearer {token}"}
invoice_data = {
    "client_name": "Client",
    "client_email": "client@example.com",
    "language": "en",
    "currency": "USD",
    "items": [{"name": "Service", "quantity": 1, "price": 100}]
}

response = requests.post(
    f"{API_URL}/invoices/generate",
    json=invoice_data,
    headers=headers
)
invoice = response.json()
print(f"Invoice created: {invoice['invoice_number']}")
```

JavaScript/Node.js

javascript

```javascript
const axios = require('axios');

const API_URL = 'http://localhost:8000';

// Login
const login = async () => {
  const response = await axios.post(`${API_URL}/auth/login`, {
    username: 'user',
    password: 'pass'
  });
  return response.data.access_token;
};

// Create invoice
const createInvoice = async (token) => {
  const response = await axios.post(
    `${API_URL}/invoices/generate`,
    {
      client_name: 'Client',
      client_email: 'client@example.com',
      language: 'en',
      currency: 'USD',
      items: [
        { name: 'Service', quantity: 1, price: 100 }
      ]
    },
    {
      headers: { Authorization: `Bearer ${token}` }
    }
  );
  return response.data;
};

// Usage
(async () => {
  const token = await login();
  const invoice = await createInvoice(token);
  console.log(`Invoice created: ${invoice.invoice_number}`);
})();
```

PHP

```php
<?php

$apiUrl = 'http://localhost:8000';

// Login
$response = file_get_contents($apiUrl . '/auth/login', false, stream_context_create([
    'http' => [
        'method' => 'POST',
        'header' => 'Content-Type: application/json',
        'content' => json_encode([
            'username' => 'user',
            'password' => 'pass'
        ])
    ]
]));

$token = json_decode($response)->access_token;

// Create invoice
$invoiceData = [
    'client_name' => 'Client',
    'client_email' => 'client@example.com',
    'language' => 'en',
    'currency' => 'USD',
    'items' => [
        ['name' => 'Service', 'quantity' => 1, 'price' => 100]
    ]
];

$response = file_get_contents($apiUrl . '/invoices/generate', false, stream_context_create([
    'http' => [
        'method' => 'POST',
        'header' => "Authorization: Bearer $token\r\nContent-Type: application/json",
        'content' => json_encode($invoiceData)
    ]
]));

$invoice = json_decode($response);
echo "Invoice created: " . $invoice->invoice_number;
?>
```

🚀 RapidAPI Integration

This API is designed to be published on RapidAPI. Here's how to integrate:

RapidAPI Configuration

1. **Base URL**: Your production URL
2. **Authentication**: JWT Bearer token
3. **Rate Limiting**: Managed by RapidAPI tiers

Pricing Tiers (Suggested)

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

📖 FAQs

How do I generate an invoice in Arabic?

Set `"language": "ar"` in the invoice creation request.

Can I customize the PDF template?

Yes, edit the HTML templates in `app/templates/`.

How long are JWT tokens valid?

Tokens are valid for 30 minutes by default.

What happens if I exceed the email rate limit?

You'll receive a `429 Too Many Requests` error. Wait one hour before sending more emails.

Can I use my own payment gateway?

Yes! The `payment_link` field is customizable. Integration with real payment gateways will be added in future versions.

How do I backup my invoices?

All invoice data is stored in the database. Regular database backups are recommended.

🐛 Troubleshooting

Issue: "Could not validate credentials"

Solution: Your JWT token has expired. Login again to get a new token.

Issue: "Email rate limit exceeded"

Solution: Wait one hour. Each user can send max 5 emails per hour.

Issue: "PDF not generated"

Solution: Ensure WeasyPrint dependencies are installed. Check server logs for details.

Issue: "Invoice not found"

Solution: Verify the invoice ID and ensure you're the owner of the invoice.

📞 Support & Contact

- **Documentation**: http://localhost:8000/docs
- **GitHub Issues**: [Your Repo URL]
- **Email**: support@yourdomain.com
- **Discord**: [Your Discord Server]

🔄 Changelog

Version 1.0.0 (MVP)

- ✅ User authentication (JWT)
- ✅ Invoice generation (PDF + QR)
- ✅ Email sending capability
- ✅ Multi-language support (AR/EN)
- ✅ Multi-currency support
- ✅ RapidAPI ready

Planned Features

- 🔜 Payment gateway integration (Stripe, PayPal)
- 🔜 Webhook notifications
- 🔜 Invoice templates customization
- 🔜 Recurring invoices
- 🔜 Multi-language expansion
- 🔜 Mobile app SDK

Made with ❤️ for freelancers and small businesses