# SiteProc API Documentation

**Version 1.0** | Last Updated: November 2025

## Table of Contents

## Getting Started

The SiteProc API allows you to integrate construction management functionality into your own applications.

### Base URL

```
https://api.siteproc.com/v1
```

### API Versioning

The API is versioned using URL path versioning. Current version: `v1`

### Content Type

All requests and responses use JSON:

```
Content-Type: application/json
```

## Authentication

### API Keys

1. **Generate API Key**
   - Log into SiteProc
   - Go to Settings → Integrations → API
   - Click "Generate New API Key"

   - Copy and store securely

2. **Authentication Header**

   ```
   Authorization: Bearer YOUR_API_KEY
   ```

### Example Request

```
curl -X GET \
  https://api.siteproc.com/v1/projects \
  -H 'Authorization: Bearer sk_live_abc123...' \
  -H 'Content-Type: application/json'
```

### Security Best Practices

☑ **Do:**

- Store API keys securely (environment variables)
- Use separate keys for dev/production
- Rotate keys regularly
- Use HTTPS only

✖ **Don't:**

- Commit keys to version control
- Share keys publicly
- Use production keys in development
- Hardcode keys in applications

---

## Rate Limiting

### Limits

- **Standard Plan:** 1,000 requests/hour
- **Professional Plan:** 5,000 requests/hour
- **Enterprise Plan:** Custom limits

### Rate Limit Headers

```
X-RateLimit-Limit: 1000
X-RateLimit-Remaining: 950
X-RateLimit-Reset: 1699380000
```

### Handling Rate Limits

When rate limited, API returns:

```
{
  "error": {
    "type": "rate_limit_exceeded",
    "message": "Rate limit exceeded. Try again in 45 seconds.",
    "retry_after": 45
  }
}
```

**HTTP Status:** `429 Too Many Requests`

---

# Endpoints

## Projects

### List Projects

```
GET /projects
```

**Parameters:**

| Parameter | Type | Description |
|---|---|---|
| status | string | Filter by status: active, completed, archived |
| limit | integer | Number of results (1-100, default: 20) |
| offset | integer | Pagination offset |

**Response:**

```
{
  "data": [
    {
      "id": "prj_abc123",
      "name": "Downtown Office Renovation",
      "status": "active",
      "budget": 50000.00,
      "spent": 25000.00,
      "start_date": "2025-01-15",
      "end_date": "2025-06-30",
      "client": {
        "id": "cli_xyz789",
        "name": "ABC Company"
      },
      "created_at": "2025-01-10T10:00:00Z",
      "updated_at": "2025-11-07T14:30:00Z"
    }
  ],
  "pagination": {
    "total": 45,
    "limit": 20,
    "offset": 0,
    "has_more": true
  }
}
```

#### Get Project

```
GET /projects/{project_id}
```

**Response:**

```
{
  "id": "prj_abc123",
  "name": "Downtown Office Renovation",
  "description": "Complete renovation of 5th floor office space",
  "status": "active",
  "budget": 50000.00,
  "spent": 25000.00,
  "committed": 15000.00,
  "remaining": 10000.00,
  "start_date": "2025-01-15",
  "end_date": "2025-06-30",
  "completion_percentage": 45,
  "client": {
    "id": "cli_xyz789",
    "name": "ABC Company",
    "contact_email": "john@abccompany.com",
    "contact_phone": "+1234567890"
  },
  "location": {
    "address": "123 Main St, Suite 500",
    "city": "New York",
    "state": "NY",
    "zip": "10001",
    "country": "US",
    "coordinates": {
      "latitude": 40.7128,
      "longitude": -74.0060
    }
  },
  "team": [
    {
      "id": "usr_def456",
      "name": "John Smith",
      "role": "project_manager",
      "email": "john@company.com"
    }
  ],
  "created_at": "2025-01-10T10:00:00Z",
  "updated_at": "2025-11-07T14:30:00Z"
}
```

#### Create Project

```
POST /projects
```

**Request Body:**

```
{
  "name": "New Office Building",
  "description": "Construction of 3-story office building",
  "status": "planning",
  "budget": 500000.00,
  "start_date": "2025-12-01",
  "end_date": "2026-12-31",
  "client_id": "cli_xyz789",
  "location": {
    "address": "456 Oak Avenue",
    "city": "San Francisco",
    "state": "CA",
    "zip": "94102"
  },
  "team_members": ["usr_def456", "usr_ghi789"]
}
```

**Response:** `201 Created`

```
{
  "id": "prj_new123",
  "name": "New Office Building",
  ...
}
```

#### Update Project

```
PATCH /projects/{project_id}
```

**Request Body:**

```
{
  "status": "in_progress",
  "budget": 550000.00,
  "completion_percentage": 25
}
```

**Response:** `200 OK`

#### Delete Project

```
DELETE /projects/{project_id}
```

**Response:** `204 No Content`

---

## Orders

#### List Orders

```
GET /orders
```

**Parameters:**

| Parameter | Type | Description |
|---|---|---|
| project_id | string | Filter by project |
| status | string | pending, acknowledged, delivered, completed |
| supplier_id | string | Filter by supplier |
| from_date | date | Filter by date (YYYY-MM-DD) |
| to_date | date | Filter by date (YYYY-MM-DD) |
| limit | integer | Results per page (1-100) |
| offset | integer | Pagination offset |

**Response:**

```
{
  "data": [
    {
      "id": "ord_abc123",
      "order_number": "PO-2025-001",
      "project_id": "prj_abc123",
      "supplier_id": "sup_xyz789",
      "status": "delivered",
      "order_date": "2025-11-01",
      "delivery_date": "2025-11-05",
      "total_amount": 5000.00,
      "currency": "USD",
      "items": [
        {
          "id": "itm_001",
          "description": "Portland Cement",
          "quantity": 50,
          "unit": "50kg bags",
          "unit_price": 15.00,
          "total": 750.00
        }
      ],
      "created_at": "2025-11-01T09:00:00Z",
      "updated_at": "2025-11-05T14:30:00Z"
    }
  ]
}
```

#### Create Order

```
POST /orders
```

**Request Body:**

```
{
  "project_id": "prj_abc123",
  "supplier_id": "sup_xyz789",
  "delivery_date": "2025-11-15",
  "delivery_address": "123 Construction Site Ave",
  "notes": "Please call before delivery",
  "items": [
    {
      "description": "Steel Rebar 12mm",
      "quantity": 100,
      "unit": "meters",
      "unit_price": 5.50
    },
    {
      "description": "Portland Cement",
      "quantity": 20,
      "unit": "50kg bags",
      "unit_price": 15.00
    }
  ]
}
```

**Response:** `201 Created`

#### Update Order Status

```
PATCH /orders/{order_id}/status
```

**Request Body:**

```json
{
  "status": "delivered",
  "notes": "Delivered successfully at 2:30 PM",
  "proof_of_delivery": {
    "recipient_name": "John Smith",
    "signature_url": "https://...",
    "photo_urls": ["https://...", "https://..."]
  }
}
```

**Response:** `200 OK`

---

## Suppliers

#### List Suppliers

```
GET /suppliers
```

**Response:**

```json
{
  "data": [
    {
      "id": "sup_xyz789",
      "name": "ABC Building Supplies",
      "category": "materials",
      "contact_name": "Jane Doe",
      "contact_email": "jane@abcsupplies.com",
      "contact_phone": "+1234567890",
      "payment_terms": "Net 30",
      "rating": 4.5,
      "active": true
      }
    ]
}
```

#### Create Supplier

```
POST /suppliers
```

**Request Body:**

```
{
  "name": "XYZ Equipment Rental",
  "category": "equipment",
  "contact_name": "Bob Johnson",
  "contact_email": "bob@xyzequipment.com",
  "contact_phone": "+1987654321",
  "address": {
    "street": "789 Industrial Blvd",
    "city": "Chicago",
    "state": "IL",
    "zip": "60601"
  },
  "payment_terms": "Net 15",
  "notes": "Specializes in heavy equipment"
}
```

**Response:** `201 Created`

---

### Deliveries

#### Confirm Delivery

```
POST /deliveries/{delivery_id}/confirm
```

**Request Body:**

```
{
  "received_by": "John Smith",
  "received_at": "2025-11-07T14:30:00Z",
  "condition": "good",
  "notes": "All items received in good condition",
  "photos": ["base64_encoded_image_1", "base64_encoded_image_2"],
  "signature": "base64_encoded_signature"
}
```

**Response:** `200 OK`

---

# Reports

### Generate Report

```
POST /reports
```

**Request Body:**

```
{
  "type": "project_financial",
  "project_id": "prj_abc123",
  "date_from": "2025-01-01",
  "date_to": "2025-11-07",
  "format": "pdf"
}
```

**Response:**

```
{
  "report_id": "rep_abc123",
  "status": "generating",
  "download_url": null,
  "estimated_time": 30
}
```

### Get Report Status

```
GET /reports/{report_id}
```

**Response:**

```
{
  "report_id": "rep_abc123",
  "status": "completed",
  "download_url": "https://api.siteproc.com/reports/download/rep_abc123",
  "expires_at": "2025-11-08T14:30:00Z"
}
```

# Webhooks

## Overview

Webhooks allow you to receive real-time notifications when events occur in SiteProc.

## Setup

1. Go to Settings → Integrations → Webhooks
2. Add your endpoint URL
3. Select events to subscribe to
4. Save webhook

## Webhook Events

| Event | Description |
|---|---|
| `order.created` | New order created |
| `order.acknowledged` | Supplier acknowledged order |
| `order.delivered` | Order delivered |
| `order.completed` | Order completed with POD |
| `delivery.confirmed` | Delivery confirmed by site |
| `project.created` | New project created |
| `project.updated` | Project details updated |
| `budget.alert` | Budget threshold reached |
| `payment.received` | Payment received |

## Webhook Payload

```
{
  "id": "evt_abc123",
  "type": "order.delivered",
  "created_at": "2025-11-07T14:30:00Z",
  "data": {
    "order_id": "ord_abc123",
    "project_id": "prj_abc123",
    "supplier_id": "sup_xyz789",
    "delivered_at": "2025-11-07T14:00:00Z",
    "delivered_by": "John Driver",
    "total_amount": 5000.00
  }
}
```

### Webhook Security

**Verify Signatures:**

Each webhook includes a signature header:

```
X-SiteProc-Signature: sha256=abc123def456...
```

**Verification Code (Node.js):**

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

function verifyWebhook(payload, signature, secret) {
  const hmac = crypto.createHmac('sha256', secret);
  const digest = hmac.update(payload).digest('hex');
  return `sha256=${digest}` === signature;
}

// Usage
app.post('/webhook', (req, res) => {
  const signature = req.headers['x-siteproc-signature'];
  const isValid = verifyWebhook(
    JSON.stringify(req.body),
    signature,
    process.env.WEBHOOK_SECRET
  );

  if (!isValid) {
    return res.status(401).send('Invalid signature');
  }

  // Process webhook
  console.log('Event:', req.body.type);
  res.status(200).send('OK');
});
```

---

## Error Handling

### Error Response Format

```json
{
  "error": {
    "type": "invalid_request",
    "message": "The project_id field is required.",
    "code": "missing_required_field",
    "field": "project_id"
  }
}
```

### Error Types

| Status Code | Error Type | Description |
|---|---|---|
| 400 | `invalid_request` | Request parameters invalid |
| 401 | `authentication_error` | Invalid API key |
| 403 | `permission_denied` | Insufficient permissions |
| 404 | `not_found` | Resource not found |
| 409 | `conflict` | Resource conflict |
| 429 | `rate_limit_exceeded` | Too many requests |
| 500 | `server_error` | Internal server error |

### Error Handling Example

```
try {
  const response = await fetch('https://api.siteproc.com/v1/projects', {
    headers: {
      'Authorization': `Bearer ${apiKey}`,
      'Content-Type': 'application/json'
    }
  });

  if (!response.ok) {
    const error = await response.json();

    switch (error.error.type) {
      case 'authentication_error':
        console.error('Invalid API key');
        break;
      case 'rate_limit_exceeded':
        const retryAfter = response.headers.get('Retry-After');
        console.log(`Rate limited. Retry after ${retryAfter}s`);
        break;
      default:
        console.error('API error:', error.error.message);
    }

    return;
  }

  const data = await response.json();
  console.log('Projects:', data);

} catch (err) {
  console.error('Network error:', err);
}
```

---

## Code Examples

### JavaScript/Node.js

```javascript
const fetch = require('node-fetch');

const API_KEY = 'sk_live_your_api_key';
const BASE_URL = 'https://api.siteproc.com/v1';

// Get all projects
async function getProjects() {
  const response = await fetch(`${BASE_URL}/projects`, {
    headers: {
      'Authorization': `Bearer ${API_KEY}`,
      'Content-Type': 'application/json'
    }
  });

  return await response.json();
}

// Create new order
async function createOrder(orderData) {
  const response = await fetch(`${BASE_URL}/orders`, {
    method: 'POST',
    headers: {
      'Authorization': `Bearer ${API_KEY}`,
      'Content-Type': 'application/json'
    },
    body: JSON.stringify(orderData)
  });

  return await response.json();
}

// Usage
const orderData = {
  project_id: 'prj_abc123',
  supplier_id: 'sup_xyz789',
  delivery_date: '2025-11-15',
  items: [
    {
      description: 'Cement',
      quantity: 50,
      unit: '50kg bags',
      unit_price: 15.00
    }
  ]
};

createOrder(orderData)
  .then(order => console.log('Order created:', order))
  .catch(err => console.error('Error:', err));
```

### Python

```python
import requests

API_KEY = 'sk_live_your_api_key'
BASE_URL = 'https://api.siteproc.com/v1'

headers = {
    'Authorization': f'Bearer {API_KEY}',
    'Content-Type': 'application/json'
}

# Get all projects
def get_projects():
    response = requests.get(f'{BASE_URL}/projects', headers=headers)
    return response.json()

# Create new order
def create_order(order_data):
    response = requests.post(
        f'{BASE_URL}/orders',
        headers=headers,
        json=order_data
    )
    return response.json()

# Usage
order_data = {
    'project_id': 'prj_abc123',
    'supplier_id': 'sup_xyz789',
    'delivery_date': '2025-11-15',
    'items': [
        {
            'description': 'Cement',
            'quantity': 50,
            'unit': '50kg bags',
            'unit_price': 15.00
        }
    ]
}
```

```
order = create_order(order_data)
print(f'Order created: {order}')
```

### cURL

```
# List projects
curl -X GET \
  'https://api.siteproc.com/v1/projects?limit=10' \
  -H 'Authorization: Bearer sk_live_your_api_key'

# Create order
curl -X POST \
  'https://api.siteproc.com/v1/orders' \
  -H 'Authorization: Bearer sk_live_your_api_key' \
  -H 'Content-Type: application/json' \
  -d '{
    "project_id": "prj_abc123",
    "supplier_id": "sup_xyz789",
    "delivery_date": "2025-11-15",
    "items": [
      {
        "description": "Cement",
        "quantity": 50,
        "unit": "50kg bags",
        "unit_price": 15.00
      }
    ]
  }'

# Update order status
curl -X PATCH \
  'https://api.siteproc.com/v1/orders/ord_abc123/status' \
  -H 'Authorization: Bearer sk_live_your_api_key' \
  -H 'Content-Type: application/json' \
  -d '{
    "status": "delivered",
    "notes": "Delivered successfully"
  }'
```

---

## SDKs

### Official SDKs

**JavaScript/TypeScript**

```
npm install @siteproc/sdk
```

#### Python

```
pip install siteproc-sdk
```

#### PHP

```
composer require siteproc/sdk
```

### SDK Usage Example

```
const SiteProc = require('@siteproc/sdk');

const client = new SiteProc({
  apiKey: 'sk_live_your_api_key'
});

// Get projects
const projects = await client.projects.list();

// Create order
const order = await client.orders.create({
  projectId: 'prj_abc123',
  supplierId: 'sup_xyz789',
  deliveryDate: '2025-11-15',
  items: [
    {
      description: 'Cement',
      quantity: 50,
      unit: '50kg bags',
      unitPrice: 15.00
    }
  ]
});

// Subscribe to webhooks
client.webhooks.on('order.delivered', (event) => {
  console.log('Order delivered:', event.data);
});
```

---

## Best Practices

### Performance

☑ **Use pagination** for large datasets ☑ **Cache responses** when appropriate ☑ **Use webhooks** instead of polling ☑ **Batch requests** when possible ☑ **Compress responses** (gzip)

### Security

✅ **Store API keys securely** ✅ **Use HTTPS only** ✅ **Validate webhook signatures** ✅ **Rotate keys regularly** ✅ **Use environment variables**

### Error Handling

✅ **Implement retries** with exponential backoff ✅ **Handle rate limits** gracefully ✅ **Log errors** for debugging ✅ **Validate inputs** before API calls

---

## Support

### API Support

📧 **Email:** api-support@siteproc.com 💬 **Discord:** https://discord.gg/siteproc 📚 **Docs:** https://docs.siteproc.com/api

### Report Issues

🐛 **GitHub:** https://github.com/siteproc/api-issues 📊 **Status Page:** https://status.siteproc.com

---

API Version: 1.0 Last Updated: November 2025