API REF v1.0

PP AI

API Integration Reference

Connecting ParkPilot AI agents to your park management systems

This document covers the complete integration procedure for connecting ParkPilot AI to ParcVu, Elite Dynamics, and Salesforce via REST API. Intended for technical teams and system administrators.

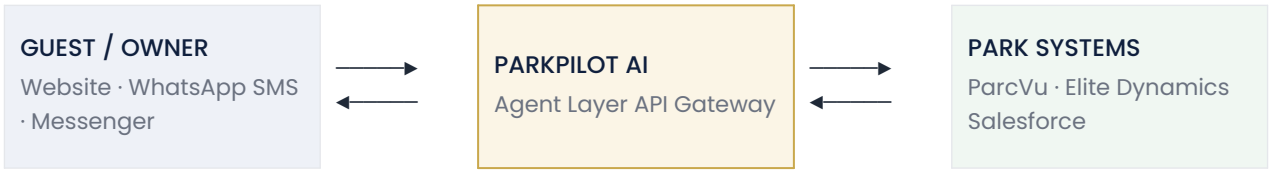
Version 1.0 · parkpilot.uk/docs · February 2026

CONTENTS

SECTION 1

Overview & Architecture

ParkPilot AI connects to your park's existing systems via a secure REST API layer. Rather than replacing your current tools, ParkPilot operates as intelligent middleware — reading live data from your PMS and CRM, processing guest and owner requests through its AI agents, and writing confirmed actions directly back into your systems. No duplicate entry. No manual handover.



Integration scope

Each ParkPilot deployment is scoped individually during onboarding. The integrations you activate depend on which agents are deployed at your park.

Guest Services	Required	Optional	—
Holiday Home Sales	Optional	Required	Optional
Aftersales & Owner Services	—	Required	—
Admin & Compliance	Optional	Required	—

SECTION 2

Authentication

ParkPilot uses API key authentication for all outbound calls to park systems. Keys are provisioned during onboarding and stored encrypted within ParkPilot's secure vault. Your park system credentials never leave your infrastructure unencrypted.

API key format

```http
# Include in all outbound requests
Authorization: Bearer pp_live_xxxxxxxxxxxxxxxxxxx

# Sandbox / testing environment
Authorization: Bearer pp_test_xxxxxxxxxxxxxxxxxxx
```

Key security

API keys are scoped per park and per environment. Never share your live key. If a key is compromised, contact integrations@parkpilot.uk immediately — keys can be rotated without service interruption.

SECTION 3

Base URLs & Environments

ParkPilot operates two environments. Always complete full integration testing in the sandbox environment before requesting promotion to production.

Production	`https://api.parkpilot.uk/v1`	Live park operations
Sandbox	`https://sandbox.api.parkpilot.uk/v1`	Integration testing — no live actions triggered

SECTION 4

ParcVu Integration

ParcVu is the primary integration for parks using it as their property management and bookings platform. ParkPilot connects to ParcVu to check live availability, retrieve guest records, create and modify bookings, and raise maintenance jobs.

> **Before you begin**
> You will need your ParcVu API credentials from your ParcVu account manager. ParkPilot requires read and write permissions on the Bookings, Guests, Availability, and Maintenance modules.

4.1 Check Availability

`GET` `/parcvu/availability` — Check live unit availability

Returns real-time availability for a given accommodation type and date range, applying all park-configured booking rules including minimum stay, arrival day restrictions, and pet policies.

Parameter	Type		Description
park_id	string	required	Your unique ParkPilot park identifier
arrival	date	required	Requested arrival date (YYYY-MM-DD)
departure	date	required	Requested departure date (YYYY-MM-DD)
unit_type	string	optional	lodge · static · touring · glamping · hire
guests	integer	optional	Party size — filters by unit capacity
pets	boolean	optional	Filter to pet-friendly units only

Example response

```json
{
  "available": true,
  "units": [{
    "unit_id":        "LDG-042",
    "name":           "Willow Lodge",
    "type":           "lodge",
    "capacity":       6,
    "pet_friendly":   true,
    "price_per_night": 185.00,
    "total_price":    1295.00,
    "currency":       "GBP"
  }],
  "booking_rules": {
    "min_stay_nights": 3,
    "arrival_days":   ["Friday", "Monday"]
  }
}
```

4.2 Create Booking

POST `/parcvu/bookings` — Create a confirmed booking

Creates a confirmed booking in ParcVu directly from an agent conversation. ParcVu returns a booking reference which is relayed immediately to the guest.

```json
{
  "park_id":    "PP-PARK-001",
  "unit_id":    "LDG-042",
  "arrival":    "2026-08-01",
  "departure":  "2026-08-08",
  "guests": {
    "lead_name":  "Sarah Thompson",
    "email":      "s.thompson@email.com",
    "phone":      "+44 7700 900123",
    "party_size": 4,  "pets": 1
  },
  "payment_ref": "PAY-88123",
  "source":      "parkpilot_agent"
}
```

4.3 Raise Maintenance Job

POST `/parcvu/maintenance` — Create a maintenance job

Raised when a guest reports a fault during their stay. The agent captures fault description, unit location, and urgency before creating the job in ParcVu for the maintenance team.

```json
{
  "park_id":     "PP-PARK-001",
  "unit_id":     "LDG-042",
  "booking_ref": "PV-2026-88421",
  "fault":       "Hot tub not heating",
  "urgency":     "medium",
  "reported_by": "guest",
  "notes":       "Guest reports temp stuck at 28C"
}
```

SECTION 5

Elite Dynamics Integration

Elite Dynamics is the primary integration for holiday home sales, aftersales, and owner management. ParkPilot connects to qualify and push ownership leads, book park tours, schedule owner maintenance, and handle compliance queries.

> **Elite Dynamics API access**
> API access is granted via your Elite Dynamics Partner account. Contact your account manager to enable API permissions on the Sales, Owner, and Aftersales modules. ParkPilot requires OAuth 2.0 credentials.

5.1 Push Qualified Lead

POST `/elite/leads` — Create a qualified sales lead

Fires when the Holiday Home Sales Agent qualifies a prospective owner. Full conversation context, lead score, and buyer intent signals are pushed into Elite Dynamics automatically.

```json
{
  "park_id": "PP-PARK-001",
  "lead": {
    "name":         "James & Rachel Whitmore",
    "email":        "jwhitmore@email.com",
    "phone":        "+44 7911 234567",
    "budget":       "£80,000 - £120,000",
    "unit_pref":    "lodge",
    "timeline":     "within_6_months",
    "intent_score": 87,
    "source":       "website_chat",
    "notes":        "Hot tub lodge, south-facing plot"
  },
  "conversation_id": "CONV-20260228-4412"
}
```

5.2 Book Park Tour

POST `/elite/appointments` — Schedule a park tour

Books a park tour directly into the sales team's Elite Dynamics calendar. The agent checks available slots before confirming the appointment with the prospect.

```json
{
  "park_id":    "PP-PARK-001",
  "lead_id":    "ED-LEAD-99142",
  "type":       "park_tour",
  "date":       "2026-03-15",
  "time":       "10:30",
  "sales_rep":  "auto_assign",
  "confirmed_to_lead": true
}
```

5.3 Owner Maintenance Request

POST `/elite/owner-maintenance` — Log an owner maintenance request

Raised when an existing owner contacts the Aftersales Agent to report a fault, book an annual service, or request a warranty repair.

```json
{
  "park_id":        "PP-PARK-001",
  "owner_id":       "ED-OWN-00312",
  "plot":           "B14",
  "type":           "annual_service",
  "preferred_date": "2026-04-10",
  "notes":          "Owner requests morning slot"
}
```

SECTION 6

Salesforce Integration

Salesforce integration is available for parks using it as their enterprise CRM — typically larger parks or group operators. ParkPilot connects via a Salesforce Connected App using OAuth 2.0, pushing qualified leads and guest records into your existing org.

6.1 OAuth 2.0 Setup

Your Salesforce administrator creates a Connected App and provides ParkPilot with the consumer key and secret during onboarding. ParkPilot handles token refresh automatically.

```http
# Token endpoint
POST https://login.salesforce.com/services/oauth2/token

Content-Type: application/x-www-form-urlencoded

grant_type=client_credentials
client_id={YOUR_CONSUMER_KEY}
client_secret={YOUR_CONSUMER_SECRET}
```

6.2 Push Lead to Salesforce

POST `/salesforce/leads` — Create a Salesforce Lead record

Creates a Lead record in Salesforce when the Sales Agent qualifies a prospect. Custom fields are mapped to your Salesforce org configuration during onboarding.

```json
{
  "object": "Lead",
  "fields": {
    "FirstName":       "Rachel",
    "LastName":        "Whitmore",
    "Email":           "jwhitmore@email.com",
    "Phone":           "+44 7911 234567",
    "LeadSource":      "ParkPilot AI",
    "Description":     "Budget £80k-£120k. Lodge preference.",
    "PP_IntentScore__c": 87
  }
}
```

SECTION 7

Webhook Configuration

ParkPilot can push real-time webhook events to your systems when key actions occur. Configure your endpoints in the ParkPilot dashboard under Settings Integrations Webhooks.

`booking.confirmed`	Booking created in PMS	Booking ref, guest details, unit, dates, total
`lead.qualified`	Sales lead qualified by agent	Lead details, intent score, conversation ID
`appointment.booked`	Park tour scheduled	Lead ID, date, time, sales rep assigned
`maintenance.raised`	Maintenance job created	Job ID, unit, fault description, urgency level
`handoff.requested`	Agent escalates to human	Reason, conversation transcript, channel
`owner.query.resolved`	Owner query handled by agent	Owner ID, query type, resolution summary

Webhook security
All webhook payloads are signed with HMAC-SHA256 using your webhook secret. Always verify the X-ParkPilot-Signature header before processing any incoming payload.

SECTION 8

Error Codes & Handling

ParkPilot returns standard HTTP status codes. All error responses include a machine-readable error code and a human-readable message.

400	bad_request	Missing or invalid parameters. Check all required fields.
401	unauthorised	Invalid or missing API key. Verify your Bearer token.
403	forbidden	API key lacks permission for this action. Check module scopes.
404	not_found	Resource does not exist. Verify park_id and unit_id.
409	conflict	Booking conflict — unit no longer available. Re-check availability.
422	unprocessable	Request valid but failed booking rule validation.
429	rate_limit_exceeded	Too many requests. Retry after the Retry-After header value.
500	internal_error	ParkPilot server error. Retry with exponential back-off.
503	service_unavailable	Upstream system unreachable. Check status.parkpilot.uk.

Error response format

```json
{
  "error": {
    "code":       "conflict",
    "message":    "Unit LDG-042 is no longer available",
    "details": {
      "unit_id":   "LDG-042",
      "arrival":   "2026-08-01",
      "departure": "2026-08-08"
    },
    "request_id": "req_8x2k9qmn4p"
  }
}
```

SECTION 9

Rate Limits & Quotas

Rate limits are applied per API key per environment. Sandbox limits are intentionally lower to encourage efficient testing patterns before production.

Production	120	Unlimited	200 over 10 seconds
Sandbox	20	5,000	40 over 10 seconds

When a rate limit is exceeded, ParkPilot returns HTTP 429 with a Retry-After header indicating how many seconds to wait. Implement exponential back-off in your integration layer.

SECTION 10

Testing & Go-Live

All integrations must be fully validated in the sandbox environment before go-live. ParkPilot's onboarding team confirms readiness and promotes your credentials to production.

1. **Obtain sandbox credentials**
   Request your pp_test_ API key and sandbox park_id from your ParkPilot onboarding contact. These are provisioned within one business day of onboarding.

2. **Configure endpoints**
   Point all integration calls to https://sandbox.api.parkpilot.uk/v1. No live data or real bookings will be created in the sandbox environment.

3. **Run the test suite**
   Use our Postman collection (available at parkpilot.uk/docs/postman) to run all endpoint tests against your specific park configuration and unit inventory.

4. **Validate webhook delivery**
   Use a tool such as webhook.site to inspect incoming webhook payloads and confirm event structure matches your system's expectations before wiring to production.

5. **Sign-off & go-live**
   Once all tests pass, your ParkPilot onboarding contact promotes credentials to production and actively monitors the first 48 hours of live operation with you.

Integration support
For technical assistance, contact integrations@parkpilot.uk and include your park_id and the request_id from any error response. The ParkPilot technical team responds within one business day.