Spitch.ch Integration Technical Specification

PowerOn Platform | *Version 2.0* | *Date: September 2025*

📋 Executive Summary

Technical specification for integrating spitch.ch telephone service into PowerOn platform. Enables external mandates to call companies via phone, interact with AI using mandate-specific data, and stores call transcripts in PowerOn database.

1. Project Overview

1.1 Integration Scope

- New "spitch" router for external service communication
- Enhanced mandate management with spitch-specific data
- New call transcript data model and storage
- Frontend UI for mandate-user management and call permissions
- API key authentication for external service endpoints
- **NEW:** Mandate registration workflow for telephone service
- **NEW:** Document content extraction engine integration

1.2 Service Logic Summary

PowerOn Side:

- Stores mandate-specific data (general info, staff, services, FAQs)
- Receives and stores call transcripts from spitch.ch
- Provides mandate data to spitch.ch via API endpoints
- Manages user permissions for call data access
- Maintains mandate isolation and security
- **NEW:** Generates AI-powered documents for Spitch using document content extraction engine
- **NEW:** Manages mandate registration workflow for telephone service

Spitch.ch Side:

- Receives mandate data from PowerOn for AI conversation context
- Handles phone calls and AI conversations with mandates
- Sends call transcripts back to PowerOn for storage
- Uses mandate-specific data to personalize mandate interactions
- **NEW:** Creates technical SIP numbers for mandates
- **NEW:** Operates calls independently as a service

1.3 Product Manager View - Mandate Journey

🎯 **Streamlined Mandate Journey (Confirmed by Spitch PM Vadim)**

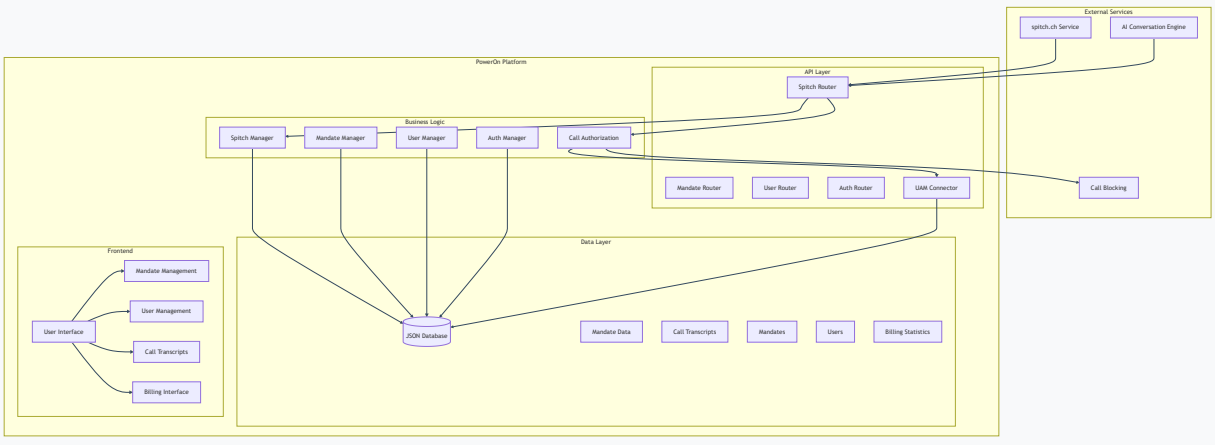
1. **Mandate Registration:** Mandate registers with PowerOn for telephone service with company profile
2. **Initial Setup:** Initially manual, final workflow-driven where mandate drops documents into the box and provides links and text fragments
3. **Integration:** This becomes another connector like Outlook or SharePoint
4. **Profile Transfer:** PowerOn sends new mandate profile to Spitch
5. **Technical Setup:** Spitch initializes mandate and creates technical SIP number
6. **Call Forwarding:** Mandate sets up call forwarding to the technical number (can remove anytime if issues arise - ensuring BCM safety)
7. **Document Generation:** PowerOn AI generates documents for Spitch (already active, document content extraction engine)
8. **Ready State:** Mandate can switch their phone number to the technical number (SIP) anytime, saving telephony costs

1.4 Data Synchronization Approach

🔄 **Data Flow Strategy (Updated per Meeting 2 Sept 2025)**

- **Call Authorization:** Spitch checks with PowerOn before each call to verify mandate is active/authorized
- **PowerOn-Initiated Updates:** All mandate data changes (FAQ, employees, services) initiated by PowerOn
- **Offline Data Management:** Spitch serves pre-configured data, no real-time data building during calls
- **Real-time Transcripts:** Spitch feeds call transcripts to PowerOn API after each call
- **JSON Document Format:** Spitch receives a JSON document per mandate, no files
- **Failsafe Mechanism:** If PowerOn is unreachable, calls are blocked with appropriate messaging

2. System Architecture Overview



3. Data Models

3.1 Enhanced Mandate Model

```
class Mandate(BaseModel, ModelMixin):
    id: str = Field(default_factory=lambda: str(uuid.uuid4()))
    name: str = Field(description="Name of the mandate")
    language: str = Field(default="en", description="Default language")
    enabled: bool = Field(default=True, description="Mandate status")
    spitch_enabled: bool = Field(default=False, description="Spitch service enabled")
    spitch_config: Optional[Dict[str, Any]] = Field(default=None, description="Spitch configuration")
    created_at: datetime = Field(default_factory=datetime.now)
    modified_at: datetime = Field(default_factory=datetime.now)
    created_by: str = Field(default="")
    modified_by: str = Field(default="")
```

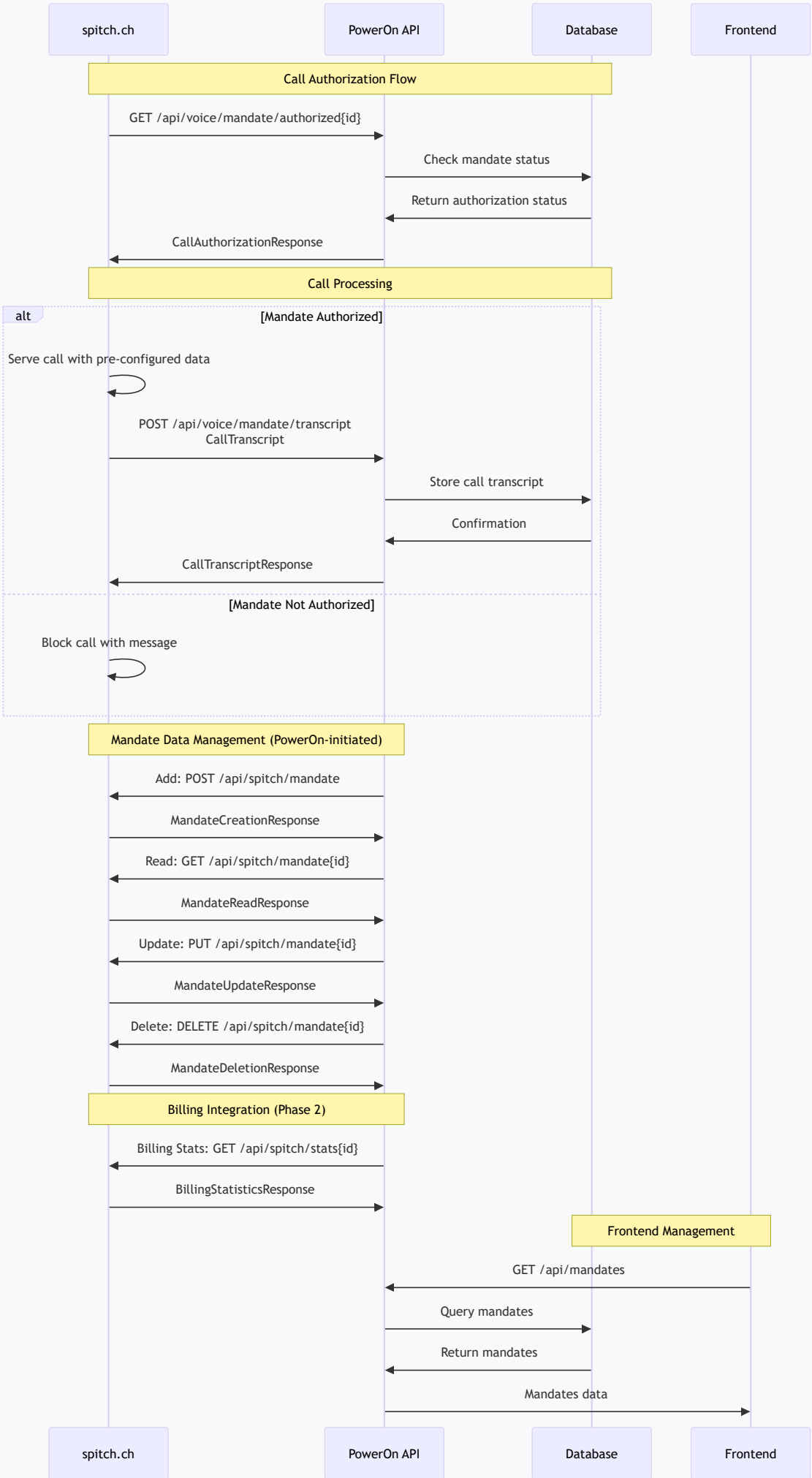
3.2 Spitch Mandate Data Model

```
class SpitchMandateData(BaseModel, ModelMixin):
    id: str = Field(default_factory=lambda: str(uuid.uuid4()))
    mandateId: str = Field(description="Associated mandate ID")
    mandate_general: Dict[str, Any] = Field(description="General mandate information")
    mandate_staff: List[Dict[str, Any]] = Field(description="Staff information")
    mandate_service: List[Dict[str, Any]] = Field(description="Service information")
    mandate_faq: List[Dict[str, Any]] = Field(description="FAQ data")
    mandates: List[Dict[str, Any]] = Field(description="List of mandates with data about main topics")
    contacts: List[Dict[str, Any]] = Field(description="List of contacts with phone numbers and references to
    last_updated: datetime = Field(default_factory=datetime.now)
    created_by: str = Field(description="User ID who created the data")
    modified_by: str = Field(description="User ID who last modified the data")
```

3.3 Call Transcript Model

```
class CallTranscript(BaseModel, ModelMixin):
    id: str = Field(default_factory=lambda: str(uuid.uuid4()))
    mandateId: str = Field(description="Associated mandate ID")
    start_datetime: datetime = Field(description="Call start time")
    finish_datetime: datetime = Field(description="Call end time")
    caller_phone: str = Field(description="Caller's phone number")
    recipient_phone: str = Field(description="Recipient's phone number")
    transcript_text: str = Field(description="Call transcript content")
    subject: str = Field(description="Call subject/topic")
    tags: List[str] = Field(default_factory=list, description="Call tags")
    spitch_call_id: str = Field(description="External spitch call identifier")
    created_at: datetime = Field(default_factory=datetime.now)
    created_by: str = Field(default="")
```

3.4 Data Flow Diagram


spitch.ch
PowerOn API
Database
Frontend
Call Authorization Flow
GET /api/voice/mandate/authorized{id}
Check mandate status
Return authorization status
CallAuthorizationResponse
Call Processing
alt
[Mandate Authorized]
Serve call with pre-configured data
POST /api/voice/mandate/transcript
CallTranscript
Store call transcript
Confirmation
CallTranscriptResponse
[Mandate Not Authorized]
Block call with message
Mandate Data Management (PowerOn-initiated)
Add: POST /api/spitch/mandate
MandateCreationResponse
Read: GET /api/spitch/mandate{id}
MandateReadResponse
Update: PUT /api/spitch/mandate{id}
MandateUpdateResponse
Delete: DELETE /api/spitch/mandate{id}
MandateDeletionResponse
Billing Integration (Phase 2)
Billing Stats: GET /api/spitch/stats{id}
BillingStatisticsResponse
Frontend Management
GET /api/mandates
Query mandates
Return mandates
Mandates data
spitch.ch
PowerOn API
Database
Frontend

3.5 API Format Contract Specification

All API calls use API key authentication via Authorization Bearer header, with user/system identification automatically determined from API key context and ID parameter required in all request/response data objects for mandate isolation.

3.5.1 Spitch-Initiated API Calls (Spitch → PowerOn)

Call Authorization Request

```
GET /api/voice/mandate/authorized{id}
Headers: Authorization: Bearer {api_key}
```

CallAuthorizationResponse

```
{
    "id": "uuid-string",
    "authorized": true,
    "reason": "Mandate is active and authorized",
    "checked_at": "2025-09-02T10:30:00Z",
    "expires_at": "2025-09-02T11:30:00Z"
}
```

Call Transcript Storage Request

```
POST /api/voice/mandate/transcript
Headers: Authorization: Bearer {api_key}
Content-Type: application/json

{
    "id": "uuid-string",
    "start_datetime": "2025-09-02T10:30:00Z",
    "finish_datetime": "2025-09-02T10:45:00Z",
    "caller_phone": "+41 987 654 321",
    "recipient_phone": "+41 123 456 789",
    "transcript_text": "Full conversation transcript...",
    "subject": "Service Inquiry",
    "tags": ["service", "inquiry", "new-mandate"],
    "spitch_call_id": "spitch-call-uuid-123"
}
```

CallTranscriptResponse

```
{
    "id": "uuid-string",
    "status": "success",
    "transcript_id": "transcript-uuid-456",
    "stored_at": "2025-09-02T10:45:30Z"
```

```
}
```

3.5.2 PowerOn-Initiated API Calls (PowerOn → Spitch)

Mandate Creation Request

```
POST /api/spitch/mandate
Headers: Authorization: Bearer {api_key}
Content-Type: application/json

{
    "id": "uuid-string",
    "mandate_general": {
        "company_name": "Company Name",
        "industry": "Industry Type",
        "contact_info": {
            "email": "contact@company.com",
            "phone": "+41 123 456 789",
            "address": "Company Address"
        },
        "business_hours": "9:00-17:00",
        "timezone": "Europe/Zurich"
    },
    "mandate_staff": [...],
    "mandate_service": [...],
    "mandate_faq": [...],
    "mandates": [...],
    "contacts": [...]
}
```

MandateCreationResponse

```
{
    "id": "uuid-string",
    "status": "success",
    "mandate_id": "spitch-mandate-uuid-789",
    "phone_number": "+41 44 123 4567",
    "created_at": "2025-09-02T10:30:00Z"
}
```

Mandate Read Request

```
GET /api/spitch/mandate{id}
Headers: Authorization: Bearer {api_key}
```

MandateReadResponse

```
{
    "id": "uuid-string",
    "status": "success",
```

```
    "mandate_id": "spitch-mandate-uuid-789",
    "mandate_general": {...},
    "mandate_staff": [...],
    "mandate_service": [...],
    "mandate_faq": [...],
    "mandates": [...],
    "contacts": [...],
    "last_updated": "2025-09-02T10:30:00Z"
}
```

Mandate Update Request

```
PUT /api/spitch/mandate{id}
Headers: Authorization: Bearer {api_key}
Content-Type: application/json

{
    "id": "uuid-string",
    "mandate_id": "spitch-mandate-uuid-789",
    "mandate_general": {...},
    "mandate_staff": [...],
    "mandate_service": [...],
    "mandate_faq": [...],
    "mandates": [...],
    "contacts": [...]
}
```

MandateUpdateResponse

```
{
    "id": "uuid-string",
    "status": "success",
    "mandate_id": "spitch-mandate-uuid-789",
    "updated_at": "2025-09-02T10:35:00Z",
    "changes_applied": ["mandate_faq", "mandate_staff"]
}
```

Mandate Deletion Request

```
DELETE /api/spitch/mandate{id}
Headers: Authorization: Bearer {api_key}
Query Parameters: id={id}
```

MandateDeletionResponse

```
{
    "id": "uuid-string",
    "status": "success",
    "mandate_id": "spitch-mandate-uuid-789",
    "deleted_at": "2025-09-02T10:40:00Z"
```

```
}
```

Billing Statistics Request (Phase 2)

```
GET /api/spitch/stats{id}?id={id}&datetimefrom={datetimefrom}&datetimeto={datetimeto}
Headers: Authorization: Bearer {api_key}
```

BillingStatisticsResponse

```
{
    "id": "uuid-string",
    "status": "success",
    "mandate_id": "spitch-mandate-uuid-789",
    "period_start": "2025-09-01T00:00:00Z",
    "period_end": "2025-09-30T23:59:59Z",
    "generated_at": "2025-10-01T00:00:00Z",
    "billing_positions": [
        {
            "position_id": "pos-001",
            "position_type": "call_minutes",
            "description": "AI Call Minutes",
            "quantity": 125.5,
            "unit": "minutes",
            "unit_price_chf": 0.15,
            "total_amount_chf": 18.83
        },
        {
            "position_id": "pos-002",
            "position_type": "external_twilio",
            "description": "Twilio Voice API Costs",
            "quantity": 1,
            "unit": "service",
            "unit_price_chf": 15.75,
            "total_amount_chf": 15.75
        },
        {
            "position_id": "pos-003",
            "position_type": "external_aws",
            "description": "AWS Transcribe Costs",
            "quantity": 1,
            "unit": "service",
            "unit_price_chf": 8.25,
            "total_amount_chf": 8.25
        },
        {
            "position_id": "pos-004",
            "position_type": "spitch_service",
            "description": "Spitch Platform Service Fee",
            "quantity": 1,
            "unit": "service",
            "unit_price_chf": 10.00,
            "total_amount_chf": 10.00
        }
    ],
    "summary": {
        "total_positions": 4,
        "total_amount_chf": 52.83,
        "external_costs_chf": 24.00,
```

```
        "spitch_fees_chf": 28.83
    }
}
```

3.5.3 Error Response Format

```
{
    "id": "uuid-string",
    "status": "error",
    "error_code": "MANDATE_NOT_FOUND",
    "error_message": "Mandate ID does not exist",
    "error_details": {
        "provided_id": "invalid-uuid"
    },
    "timestamp": "2025-09-02T10:30:00Z",
    "request_id": "req-12345"
}
```

3.5.4 Common Error Codes

Error Code	HTTP Status	Description	Resolution
UNAUTHORIZED	401	Invalid or missing API key	Check API key configuration
MANDATE_NOT_FOUND	404	Mandate ID does not exist	Verify mandate ID is correct
RATE_LIMIT_EXCEEDED	429	API rate limit exceeded	Wait and retry request
VALIDATION_ERROR	400	Request data validation failed	Check request format and required fields
INTERNAL_ERROR	500	Internal server error	Contact support with request_id

4. Pilot Mandate Data

4.1 Initial Pilot Mandates

Pilot Mandate 1: ValueOn AG

- **Company:** ValueOn AG
- **Phone Number:** +41 44 9437040
- **Provider:** Swisscom Fixnet AG
- **Status:** Ready for spitch.ch integration

Pilot Mandate 2: PamoCreate AG

- **Company:** PamoCreate AG

- **Phone Number:** +41 55 2462345
- **Provider:** iWay AG
- **Status:** Ready for spitch.ch integration

4.2 Pilot Mandate Setup Requirements

- Mandate creation in PowerOn with spitch_enabled = True
- Mandate data population (general info, staff, services, FAQs)
- Phone number mapping configuration in spitch.ch system
- Test call validation and transcript storage verification
- User permission setup for call data access

5. Spitch.ch Service Perspective

5.1 Data Received from PowerOn per Mandate

```
{
    "mandateId": "uuid-string",
    "mandate_general": {
        "company_name": "Company Name",
        "industry": "Industry Type",
        "contact_info": {
            "email": "contact@company.com",
            "phone": "+41 123 456 789",
            "address": "Company Address"
        },
        "business_hours": "9:00-17:00",
        "timezone": "Europe/Zurich"
    },
    "mandate_staff": [
        {
            "name": "Staff Member Name",
            "role": "Position/Role",
            "department": "Department",
            "contact": "+41 123 456 789",
            "expertise": ["Area 1", "Area 2"]
        }
    ],
    "mandate_service": [
        {
            "service_name": "Service Name",
            "description": "Service Description",
            "pricing": "Price Information",
            "availability": "Availability Details",
            "requirements": ["Requirement 1", "Requirement 2"]
        }
    ],
    "mandate_faq": [
        {
            "question": "Frequently Asked Question",
            "answer": "Standard Answer",
            "category": "Question Category",
            "keywords": ["keyword1", "keyword2"]
        }
    ],
    "mandates": [
        {
```

```
            "mandate_id": "uuid-string",
            "name": "Mandate Name",
            "main_topics": ["Topic 1", "Topic 2", "Topic 3"],
            "description": "Mandate description",
            "status": "active",
            "priority": "high"
        }
    ],
    "contacts": [
        {
            "name": "Contact Person Name",
            "phone_number": "+41 123 456 789",
            "email": "contact@company.com",
            "role": "Contact Role",
            "mandate_references": ["uuid-string-1", "uuid-string-2"],
            "is_primary": true
        }
    ]
}
```

5.2 Data Delivered to PowerOn per Call

```
{
    "mandateId": "uuid-string",
    "start_datetime": "2025-01-15T10:30:00Z",
    "finish_datetime": "2025-01-15T10:45:00Z",
    "caller_phone": "+41 987 654 321",
    "recipient_phone": "+41 123 456 789",
    "transcript_text": "Full conversation transcript...",
    "subject": "Service Inquiry",
    "tags": ["service", "inquiry", "new-mandate"],
    "spitch_call_id": "spitch-call-uuid-123"
}
```

5.3 Call Processing Workflow

1. Mandate calls spitch.ch service
2. spitch.ch identifies mandate based on phone number
3. **spitch.ch checks mandate authorization with PowerOn API**
4. **If authorized:** spitch.ch serves call using pre-configured data
5. **If not authorized or PowerOn unreachable:** Call is blocked with appropriate message
6. AI uses pre-configured mandate data for personalized conversation
7. Call concludes, transcript generated
8. spitch.ch sends transcript to PowerOn for storage

6. API Endpoints

6.1 Spitch Router Endpoints (Spitch → PowerOn)

Endpoint	Method	Description	Authentication	Rate Limit	Direction
/api/voice/mandate/authorized{id}	GET	Check if mandate is authorized for calls	Bearer Token	200/minute	Spitch → PowerOn
/api/voice/mandate/transcript	POST	Store call transcript for a mandate	Bearer Token	50/minute	Spitch → PowerOn

6.2 PowerOn-Initiated Endpoints (PowerOn → Spitch)

Endpoint	Method	Description	Authentication	Rate Limit	Direction
/api/spitch/mandate	POST	Create new mandate in Spitch system	Bearer Token	10/minute	PowerOn → Spitch
/api/spitch/mandate{id}	GET	Read mandate data from Spitch system	Bearer Token	50/minute	PowerOn → Spitch
/api/spitch/mandate{id}	PUT	Update mandate data (FAQ, employees, services)	Bearer Token	20/minute	PowerOn → Spitch
/api/spitch/mandate{id}	DELETE	Delete mandate from Spitch system	Bearer Token	5/minute	PowerOn → Spitch
/api/spitch/stats{id}	GET	Get billing statistics for mandate (Phase 2)	Bearer Token	10/minute	PowerOn → Spitch

6.3 Enhanced Mandate Endpoints

Endpoint	Method	Description	Authentication
/api/mandates/{mandateId}/spitch-config	PUT	Update spitch configuration for mandate	JWT (Admin/SysAdmin)
/api/mandates/{mandateId}/mandate-data	GET	Get mandate data for mandate	JWT
/api/mandates/{mandateId}/mandat-data	PUT	Update mandate data for mandate	JWT (Admin/SysAdmin)

7. Security Implementation

7.1 API Key Authentication

```
# New security middleware for spitch endpoints
class SpitchAPIKeyAuth:
    def __init__(self, api_keys: List[str]):
        self.api_keys = set(api_keys)

    async def __call__(self, request: Request):
        api_key = request.headers.get("X-API-Key")
        if not api_key or api_key not in self.api_keys:
            raise HTTPException(
                status_code=status.HTTP_401_UNAUTHORIZED,
                detail="Invalid API key"
            )
        return api_key
```

7.2 Rate Limiting

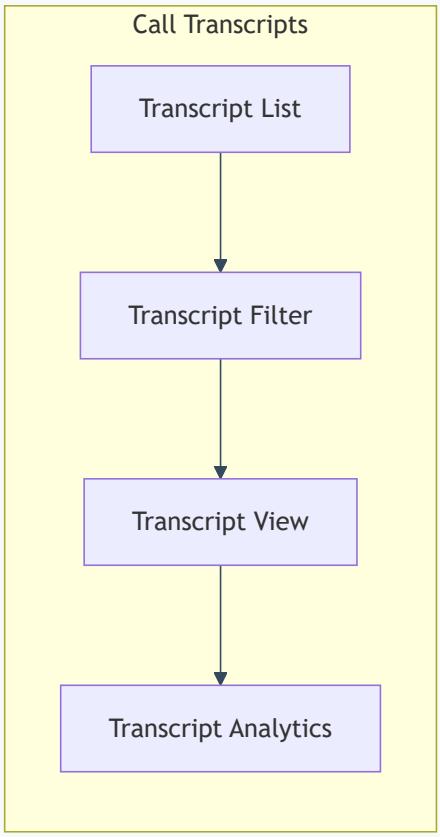
- **Mandate Data Retrieval:** 100 requests per minute per API key
- **Transcript Storage:** 50 requests per minute per API key
- **IP-based Limiting:** Additional protection against abuse

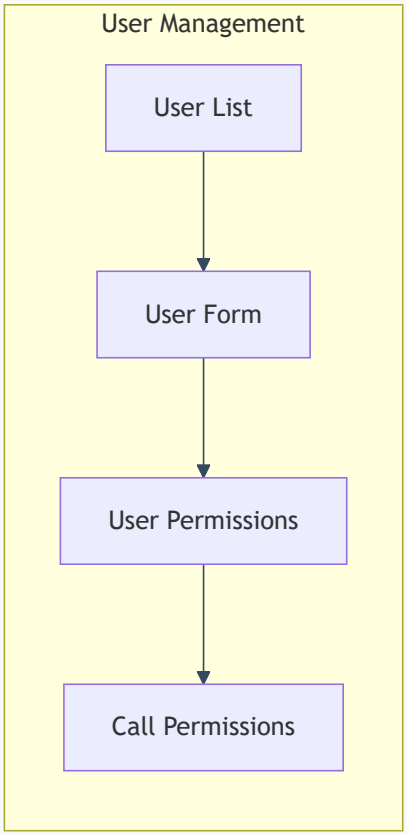
7.3 Mandate Isolation

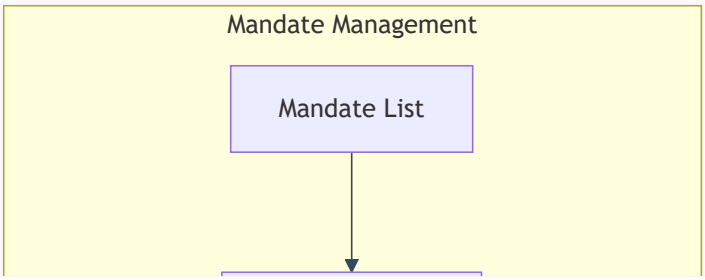
- All data queries filtered by mandate ID
- Users can only access data from their assigned mandate
- API key requests validated against mandate permissions

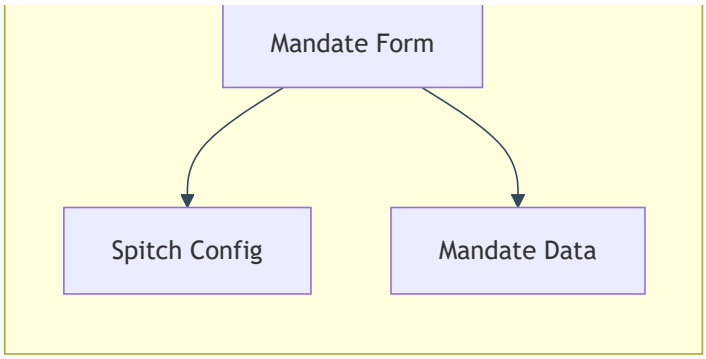
8. Frontend Implementation

8.1 Enhanced Mandate Management UI


Call Transcripts
Transcript List
Transcript Filter
Transcript View
Transcript Analytics


User Management
User List
User Form
User Permissions
Call Permissions


Mandate Management
Mandate List



8.2 New UI Components

Component	File	Description	Dependencies
SpitchConfigPanel	js/modules/spitchConfig.js	Mandate spitch configuration management	formGeneric.js, apiCalls.js
MandateDataManager	js/modules/mandateData.js	Mandate data CRUD operations	formGeneric.js, apiCalls.js
CallTranscriptsView	js/modules/callTranscripts.js	Call transcript display and filtering	formGeneric.js, apiCalls.js
UserCallPermissions	js/modules/userCallPermissions.js	User permissions for call data access	formGeneric.js, apiCalls.js

9. Implementation Approach

9.1 Two-Phase Implementation Strategy

📋 Phase 1: MVP Implementation (2 Pilot Mandates)

Duration: 6-8 weeks | **Scope:** Basic integration for 2 pilot mandates

- Basic call authorization and blocking mechanism
- Mandate data management (create, update, delete)
- Call transcript storage and retrieval
- Simple UI for mandate management
- API key authentication
- Basic error handling and logging

📋 Phase 2: Production Implementation (Full Scale)

Duration: 12-16 weeks | **Scope:** Full production system

- **Multi-tenant Architecture:** Support for hundreds of mandates
- **Billing Integration:** Complete billing statistics and cost tracking
- **Usage Tracking:** Per-minute conversation tracking and reporting
- **Third-party Cost Management:** Twilio and AWS cost calculation
- **Fault Tolerance:** High availability and disaster recovery
- **Security & Compliance:** Full security audit and compliance
- **CI/CD Pipeline:** Automated deployment and testing
- **Monitoring & Analytics:** Comprehensive monitoring and reporting

9.2 Implementation Sprints

Sprint 1: Backend Foundation (2 weeks)

- Create new data models (SpitchMandateData, CallTranscript)
- Enhance existing Mandate model with spitch fields
- Implement spitch router with basic endpoints
- Add API key authentication middleware
- Create database migration scripts

Package Price: 18,000 CHF

Sprint 2: Core API Implementation (2 weeks)

- Implement mandate data retrieval endpoints
- Implement call transcript storage endpoints
- Add rate limiting and security measures
- Create comprehensive error handling
- Write API documentation and tests

Package Price: 20,000 CHF

Sprint 3: Frontend Development (2 weeks)

- Create spitch configuration UI components
- Implement mandate data management interface
- Build call transcript viewing and filtering
- Add user permission management for call data
- Integrate with existing mandate/user management

Package Price: 22,000 CHF

Sprint 4: Testing & Integration (2 weeks)

- End-to-end testing with spitch.ch service
- Performance testing and optimization
- Security audit and penetration testing
- User acceptance testing
- Production deployment preparation

Package Price: 20,000 CHF

9.3 Billing Integration (Phase 2)

9.3.1 Billing Data Model

```
class BillingStatistics(BaseModel, ModelMixin):
    id: str = Field(default_factory=lambda: str(uuid.uuid4()))
    mandate_id: str = Field(description="Associated PowerOn mandate ID")
    period_start: datetime = Field(description="Billing period start")
    period_end: datetime = Field(description="Billing period end")
    total_call_minutes: float = Field(description="Total conversation minutes")
    total_calls: int = Field(description="Total number of calls")
    twilio_costs: float = Field(description="Twilio usage costs")
    aws_costs: float = Field(description="AWS Bedrock costs")
    total_cost: float = Field(description="Total external service costs")
    created_at: datetime = Field(default_factory=datetime.now)
```

9.3.2 Billing Workflow

💰 **Billing Process (Phase 2)**

1. **Data Collection:** Spitch tracks all usage metrics (minutes, calls, external costs)
2. **Periodic Reporting:** PowerOn requests billing data from Spitch (daily/weekly/monthly)
3. **Cost Calculation:** PowerOn calculates final billing amounts for mandates
4. **Mandate Billing:** PowerOn handles all mandate billing and communication
5. **Spitch Billing:** PowerOn bills Spitch for service usage

9.3.3 Billing API Endpoints

Endpoint	Method	Description	Direction
/api/spitch/billing-stats/{mandateId}?period={period}	GET	Get billing statistics for mandate	PowerOn → Spitch
/api/spitch/billing-stats/bulk?mandates={ids}&period={period}	GET	Get billing statistics for multiple mandates	PowerOn → Spitch

9.4 Project Pricing

Sprint Package Pricing

Sprint	Duration	Package Price (CHF)	Description
Sprint 1	2 weeks	18,000	Backend Foundation
Sprint 2	2 weeks	20,000	Core API Implementation
Sprint 3	2 weeks	22,000	Frontend Development
Sprint 4	2 weeks	20,000	Testing & Integration
Total Project	**8 weeks**	**80,000**	**Complete Integration**

Investment Schedule

- **Sprint 1:** 18,000 CHF (upon completion)
- **Sprint 2:** 20,000 CHF (upon completion)
- **Sprint 3:** 22,000 CHF (upon completion)
- **Sprint 4:** 20,000 CHF (upon completion)

Project Scope

- Complete backend API implementation
- Full frontend UI development
- Comprehensive testing and security audit
- Production deployment and documentation
- Post-deployment support (2 weeks)

10. Project Management

10.1 Risk Assessment

10.1.1 Technical Risks

- **API Integration Complexity:** External service dependencies may cause reliability issues
- **Data Volume:** High call volume could impact database performance
- **Security Vulnerabilities:** New endpoints increase attack surface
- **Call Authorization Failures:** PowerOn downtime could block all calls
- **Billing Data Accuracy:** Complex cost tracking across multiple services

10.1.2 Mitigation Strategies

- Implement comprehensive error handling and fallback mechanisms
- Add database indexing and archiving strategies for call transcripts
- Regular security audits and penetration testing
- Implement monitoring and alerting for all new endpoints

10.2 Success Metrics

10.2.1 Technical Metrics

- API response time < 200ms for mandate authorization checks
- 99.9% uptime for spitch integration endpoints
- Zero data loss for call transcripts
- Successful integration with spitch.ch service
- Call authorization response time < 100ms
- Billing data accuracy > 99.5%

10.2.2 Business Metrics

- Reduced mandate service response time
- Improved mandate satisfaction through AI-powered phone support
- Increased mandate engagement and retention
- Cost savings through automated mandate interactions

10.3 Project Conclusion

The spitch.ch integration extends PowerOn platform capabilities for external telephone service integration while maintaining existing security and mandate isolation principles. Implementation follows established patterns and leverages existing infrastructure for maintainability and scalability.

Total project cost: CHF 80,000 for complete integration delivered in 8 weeks across 4 sprints.

10.4 Meeting Results & Project Status

> 📞 **Meeting Results (September 2, 2025)**
>
> **Status:** All technical requirements clarified and architecture confirmed. Project ready to proceed with two-phase approach.

10.4.1 Key Decisions & Requirements

- **Call Authorization:** Spitch must check with PowerOn before each call - if PowerOn is unreachable, calls are blocked
- **API Direction:** PowerOn initiates all mandate data changes; Spitch only calls PowerOn for authorization and transcript storage
- **Two-Phase Approach:** MVP for 2 pilot mandates, then full production implementation
- **Billing Responsibility:** PowerOn handles all mandate billing; Spitch provides raw usage data
- **Data Management:** All mandate data changes happen offline via PowerOn-initiated calls
- **Failsafe Mechanism:** Call blocking with appropriate messaging when PowerOn is unavailable
- **Mandate Journey:** Streamlined workflow - mandate registration → profile transfer → SIP setup → call forwarding
- **Data Format:** JSON document per mandate (no files) with enhanced mandates[] and contacts[] attributes
- **Integration Approach:** PowerOn as another connector (like Outlook/SharePoint) with document content extraction engine

10.4.2 Technical Architecture Confirmed

- User Access Management (UAM) connector for mandate authorization

- Call blocking mechanism with mandate messaging
- PowerOn-initiated mandate lifecycle management
- Billing statistics collection and reporting (Phase 2)
- Multi-tenant architecture for production scale (Phase 2)

10.4.3 Implementation Timeline

1. **Phase 1 (Weeks 1-8):** MVP implementation for 2 pilot mandates
2. **Phase 2 (Weeks 9-24):** Full production implementation with billing
3. **Ongoing:** Support, monitoring, and continuous improvement

10.4.4 Key Benefits

- **Mandate Experience:** Seamless integration with existing PowerOn workflows
- **Technical Efficiency:** Leverages existing document content extraction engine
- **Business Continuity:** BCM-safe with removable call forwarding
- **Cost Optimization:** Potential telephony cost savings through SIP integration
- **Scalability:** Framework for additional external service integrations

---