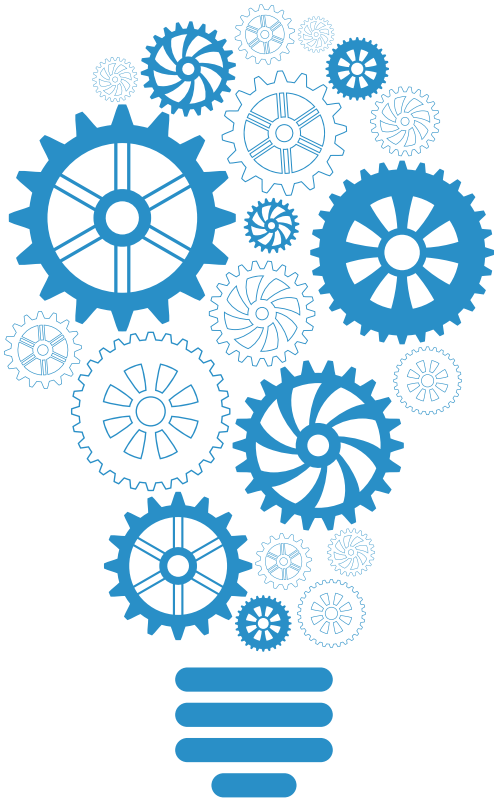
CRUSHBANK

The only business-grade AI
watsonx™

CrushBank API For Developers

The **CrushBank Universal API** provides developers with a streamlined way to **ingest data** from various sources, search within the CrushBank platform, and interact programmatically with stored data. It eliminates the need for **custom integrations** for each system, ensuring a **secure, scalable, and efficient** data transfer process.



Key Capabilities

- **Data Ingestion:** Import tickets, CRM records, configurations, and documents from external systems.
- **Search & Discovery:** Query indexed data to enhance IT support workflows.
- **Feedback & Tracking:** Rate documents, track clicks, and improve AI-driven search results.
- **Authentication & Security:** Secure access using API keys with support for **clientId** and **clientSecret** authentication.

1 Authentication

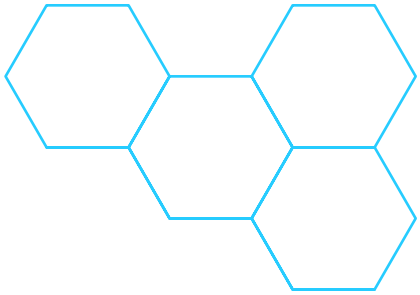
Developers must authenticate API requests using **clientId** and **clientSecret**.
These credentials can be retrieved from the **CrushBank admin panel**.

Example: Authentication Request

```
{
  “clientId”: “your-client-id”,
  “clientSecret”: “your-client-secret”
}
```

API Endpoint:
POST https://externalauthentication.crushbank.com/authenticationtoken/authentication/external
The response contains an **accessToken**, which must be included in all subsequent API calls.

2 Data Ingestion



Developers can push structured data into the CrushBank Data Lake. Supported ingestion types include:

- **Tickets** (POST /api/Ingestion/v1/ticket)
- **CRM Contacts** (POST /api/Ingestion/v1/crm)
- **Configurations** (POST /api/Ingestion/v1/config)
- **Documents** (POST /api/Ingestion/v1/document)

Authentication Request Example:

```
{
  “Context”: {
    “IntegrationUuid”: “your-integration-id”
  },
  “Content”: {
    “CreatedBy”: “User Name”,
    “CreatedDate”: “2024-02-11T10:00:00Z”,
    “UpdatedBy”: “Technician”,
    “UpdatedDate”: “2024-02-11T12:00:00Z”,
    “ClientName”: “Company A”,
    “Identifier”: “TICKET123”,
    “Title”: “Server outage issue”,
    “Properties”: {
      “Status”: “Open”,
      “Type”: “Incident”,
      “Priority”: “High”
    },
    “Description”: “Client reports a server outage affecting multiple users.”,
    “WorkNotes”: [ ],
    “TimeEntries”: [ ]
  },
  “Permissions”: [ ]
}
```

API Endpoint:
POST https://api.crushbank.com/api/ingestion/v1/ticket

Database Integration Example (SQL Server)

Developers can **automate data ingestion** by extracting records from **SQL-based systems** and sending them to CrushBank.

```
import pyodbc
import requests
import json

auth_url = "https://externalauthentication.crushbank.com/authenticationtoken/authentication/external"
auth_payload = {
    "clientId": "your-client-id",
    "clientSecret": "your-client-secret"
}
auth_response = requests.post(auth_url, json=auth_payload)
access_token = auth_response.json()['accessToken']

# Connect to SQL Server
connection_string = "DRIVER={ODBC Driver 17 for SQL Server};SERVER=your-server;DATABASE=your-db;UID=user;PWD=password"
conn = pyodbc.connect(connection_string)
cursor = conn.cursor()

# Fetch ticket records
cursor.execute("SELECT TicketID, CreatedBy, CreatedDate, Status, Description FROM Tickets")
tickets = cursor.fetchall()

# Ingest each ticket into CrushBank
for ticket in tickets:
    ticket_payload = {
        "Context": {"IntegrationUuid": "your-integration-id"},
        "Content": {
            "Identifier": ticket[0],
            "CreatedBy": ticket[1],
            "CreatedDate": ticket[2],
            "Status": ticket[3],
            "Description": ticket[4]
        }
    }
    headers = {"Authorization": f"Bearer {access_token}", "Content-Type": "application/json"}
    response = requests.post("https://api.crushbank.com/api/ingestion/v1/ticket", headers=headers, json=ticket_payload)
    print(f"Ticket {ticket[0]} uploaded: {response.status_code}")
```

3 Search & Querying

Developers can **retrieve stored data** using search API endpoints.

- **Basic Search**: (POST /api/Ask/v1/search)
- **Filtered Search (Date)**: (POST /api/Ask/v1/search?filter=date)
- **Filtered Search (Company)**: (POST /api/Ask/v1/search?filter=company)

Example: Searching for Tickets

```
{
  “query”: “server outage”,
  “filters”: {
    “date”: “last 7 days”,
    “company”: “Company A”
  }
}
```

API Endpoint:
POST https://api.crushbank.com/api/Ask/v1/search

4 Tracking & Feedback

To enhance AI-powered search results, developers can track user interactions.

- **Rate Documents**: (POST /api/Ask/v1/document/{documentId}/rate)
- **Track Document Selection**: (POST /api/Ask/v1/document/selected)
- **Report Poor Search Results**: (POST /api/Ask/v1/search/nogoodresults)

---

5 Integration Considerations

Supported Data Sources

- **External SaaS applications**
- **SQL & NoSQL databases**
- **CSV dumps & structured data exports**

Security & Compliance

- **Encryption**: Controlled by the client during API transmission.
- **Throughput & Rate Limits**: No hard limit on data ingestion, but performance varies based on volume.
- **Real-Time vs. Scheduled Imports**: Developers can set up scheduled or real-time ingestion.

Conclusion

The **CrushBank API** provides a **robust and flexible** way to **ingest, search, and interact** with data. Developers can **automate ticket ingestion**, integrate with external SaaS platforms, and enhance AI-powered search by tracking user behavior.

For full API documentation, visit: **Swagger API Reference**