CODEXIS AI API Guide

Essential operations for integrating with CODEXIS AI using GraphQL API.

Table of Contents

Authentication

You need to create account at https://auth.agrp.dev/realms/atlasgroup/login-actions/registration?client_id=codexis (https://auth.agrp.dev/realms/atlasgroup/login-actions/registration?client_id=codexis) (not working at the moment)

Get Access Token from Keycloak

https://www.keycloak.org/securing-apps/token-exchange (https://www.keycloak.org/securing-apps/token-exchange)

Use in all requests:

```
Authorization: Bearer YOUR_ACCESS_TOKEN
```

Endpoints

- **GraphQL:** `https://next.codexis.cz/graphql`
- **WebSocket:** `wss://next.codexis.cz/subscriptions`
- **File Upload:** `https://next.codexis.cz/rest/files`

File Upload

1. Upload Files

Before using files in chat, upload them first:

```
curl -X POST https://next.codexis.cz/rest/files \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -F "files=@document.pdf"
```

Response:

```
[
  {
    "id": "file123",
    "name": "document.pdf",
    "contentType": "application/pdf",
    "size": 1024000
  }
]
```

2. Upload with Preprocessing (OCR)

For scanned documents or images:

```
curl -X POST "https://your-domain/rest/files?preprocess=true" \
  -H "Authorization: Bearer YOUR_TOKEN" \
  -F "files=@scanned.pdf"
```

Core Operations

0. Get Available Custom GPTs

Before creating a chat, you can query available Custom GPTs:

```
query GetCustomGpts {
  customGpts {
    id
    name
    description
    iconUrl
    group {
      id
      name
    }
  }
}
```

Get default Custom GPT:

```
query GetDefaultCustomGpt {
  defaultCustomGpt {
    id
    name
    description
  }
}
```

Response:

```
{
  "data": {
    "defaultCustomGpt": {
      "id": "CUN1c3RvbUdwdAtkZWZhdWx0X2dwdA",
      "name": "CODEXIS AI",
      "description": "Asistent CODEXIS vám pomůže efektivně pracovat v systému CODEXIS. Je optimalizovan pro rychlé vyhledávání v ra
    }
  }
}
```

Get custom GPT groups:

```
query GetCustomGptGroups {
  customGptGroups {
    id
    name
    description
    editable
    customGpts {
      id
      name
    }
  }
}
```

0.1. Create or Update Custom GPT

Create a new custom GPT or update an existing one:

```
mutation SetCustomGpt($input: CustomGptInput!) {
  setCustomGpt(input: $input) {
    id
    name
    description
    instructions
    prompts {
      prompt
    }
    group {
      id
      name
    }
    editable
    pinned
  }
}
```

Note: To create your own custom GPT, you need to set `groupId` to `DkN1c3RvbUdwdEdyb3VwB3ZsYXN0bmk`.

Variables (Create new):

```
{
  "input": {
    "groupId": "DkN1c3RvbUdwdEdyb3VwB3ZsYXN0bmk",
    "name": "My Legal Assistant",
    "description": "Specialized assistant for Czech legal matters",
    "instructions": "You are a legal expert specializing in Czech law. Provide detailed and accurate legal analysis.",
    "prompts": [
      "Analyze this contract",
      "What are the legal implications?",
      "Draft a legal document"
    ],
    "requiresContext": false
  }
}
```

Variables (Update existing):

```
{
  "input": {
    "id": "CUN1c3RvbUdwdAtkZWZhdWx0X2dwdA",
    "groupId": "DkN1c3RvbUdwdEdyb3VwB3ZsYXN0bmk",
    "name": "Updated Legal Assistant",
    "description": "Updated description",
    "instructions": "Updated instructions",
    "prompts": ["New prompt 1", "New prompt 2"],
    "requiresContext": false
  }
}
```

Response:

```
{
  "data": {
    "setCustomGpt": {
      "id": "CUN1c3RvbUdwdAtuZXdfY3VzdG9t",
      "name": "My Legal Assistant",
      "description": "Specialized assistant for Czech legal matters",
      "instructions": "You are a legal expert specializing in Czech law. Provide detailed and accurate legal analysis.",
      "prompts": [
        {
          "prompt": "Analyze this contract"
        },
        {
          "prompt": "What are the legal implications?"
        },
        {
          "prompt": "Draft a legal document"
        }
      ],
      "group": {
        "id": "DkN1c3RvbUdwdEdyb3VwB3ZsYXN0bmk",
        "name": "vlastni"
      },
      "editable": true,
      "pinned": false
    }
  }
}
```

0.2. Get User's Chat History

Get paginated list of user's chats grouped by time:

```
query GetChatHistory($pagination: PaginationInput!, $filter: HistoryUserChatsInput) {
  historyUserChats(pagination: $pagination, filter: $filter) {
    totalCount
    groups {
      title
      chats {
        id
        name
        createdAt
        state
      }
    }
  }
}
```

Variables:

```
{
  "pagination": {
    "offset": 0,
    "limit": 20
  },
  "filter": {
    "searchQuery": ""
  }
}
```

Response:

```json
{
  "data": {
    "historyUserChats": {
      "totalCount": 45,
      "groups": [
        {
          "title": "Dnes",
          "chats": [
            {
              "id": "VXNlckNoYXQKY2hhdDQ1Ng",
              "name": "Nový chat ze dne 13.11.2025",
              "createdAt": "2025-11-13T10:30:00",
              "state": "INACTIVE"
            }
          ]
        },
        {
          "title": "Včera",
          "chats": [
            {
              "id": "VXNlckNoYXQKY2hhdDQ1NQ",
              "name": "Analýza smlouvy",
              "createdAt": "2025-11-12T15:20:00",
              "state": "INACTIVE"
            }
          ]
        }
      ]
    }
  }
}
```

1. Create New Chat

```graphql
mutation NewChat(
    $customGptId: ID!
    $prompt: String!
    $item: ID
    $fileIds: [String!]
) {
    newGptChatPrompt(
        customGptId: $customGptId
        prompt: $prompt
        item: $item
        fileIds: $fileIds
    ) {
        id
        name
        state
        messages {
            id
            author
            message
            status {
                state
                message
            }
        }
    }
}
```

Variables (Basic chat):

```
{
  "customGptId": "CUN1c3RvbUdwdAtkZWZhdWx0X2dwdA",
  "prompt": "Analyze this document",
  "fileIds": ["file123"]
}
```

Variables (Chat with specific judikat - Czech):

```
{
  "customGptId": "custom gpt Judikatura ČR AI id here",
  "prompt": "What is the main ruling in this case?",
  "item": "JUDIKAT_ID_HERE",
  "fileIds": []
}
```

Variables (Chat with specific judikat - EU):

```
{
  "customGptId": "custom gpt Judikatura EU AI id here",
  "prompt": "Analyze this EU case law",
  "item": "JUDIKAT_ID_HERE",
  "fileIds": []
}
```

Note: The `item` parameter allows you to create a chat with a specific document/judikat as context. This is useful for:

- Analyzing specific case law (judikatura)
- Asking questions about a particular legal document
- Getting AI insights on a specific judikat

For judikat analysis, use specialized Custom GPT models:

- **Judikatura ČR AI**
- **Judikatura EU AI**

Response:

```
{
  "data": {
    "newGptChatPrompt": {
      "id": "VXNlckNoYXQKY2hhdDQ1Ng",
      "name": "Nový chat ze dne 13.11.2025",
      "state": "ACTIVE",
      "messages": [
        {
          "id": "Q2hhdE1lc3NhZ2UKbXNnMA",
          "author": "SYSTEM",
          "message": "<p style=\"white-space: pre-wrap\">Jsem AI asistent v systému CODEXIS. Jsem odborník na právo v České republice
          "status": {
            "state": "COMPLETED",
            "message": null
          }
        },
        {
          "id": "Q2hhdE1lc3NhZ2UKbXNnMQ",
          "author": "USER",
          "message": "<p style=\"white-space: pre-wrap\">Analyze this document</p>",
          "status": {
            "state": "COMPLETED",
            "message": null
          }
        },
        {
          "id": "Q2hhdE1lc3NhZ2UKbXNnMg",
          "author": "AI",
          "message": "",
          "status": {
            "state": "PROCESSING",
            "message": "Přemýšlím..."
          }
        }
      ]
    }
  }
}
```

2. Send Message to Existing Chat

```
mutation SendMessage(
  $chatId: ID!
  $prompt: String!
  $fileIds: [String!]
) {
  promptChat(
    chatId: $chatId
    prompt: $prompt
    fileIds: $fileIds
  ) {
    id
    state
    messages {
      id
      author
      message
      status {
        state
      }
    }
  }
}
```

Variables:

```
{
  "chatId": "VXNlckNoYXQKY2hhdDQ1Ng",
  "prompt": "What are the key points?",
  "fileIds": []
}
```

3. Get Chat

```
query GetChat($chatId: ID!) {
  node(id: $chatId) {
    ... on UserChat {
      id
      name
      state
      messages {
        id
        author
        message
        status {
          state
          message
        }
        files {
          id
          name
        }
      }
    }
  }
}
```

4. Get Message

```
query GetMessage($messageId: ID!) {
  node(id: $messageId) {
    ... on ChatMessage {
      id
      author
      message
      status {
        state
        message
      }
    }
  }
}
```

5. Create Feedback for Message

Provide positive or negative feedback for AI message:

```
mutation CreateFeedback($input: FeedbackInput!) {
  createFeedback(input: $input) {
    id
    author
    message
    rating
    status {
      state
      message
    }
  }
}
```

Variables (Positive feedback):

```
{
  "input": {
    "messageId": "Q2hhdE1lc3NhZ2UKbXNnMg",
    "rating": "POSITIVE"
  }
}
```

Variables (Negative feedback with review):

```
{
  "input": {
    "messageId": "Q2hhdE1lc3NhZ2UKbXNnMg",
    "rating": "NEGATIVE",
    "review": {
      "predefinedReviews": ["Incorrect information", "Poor formatting"],
      "message": "The response was not accurate",
      "refundRequested": false
    }
  }
}
```

6. Stop Streaming

```
mutation StopChat($chatId: ID!) {
  stopChat(chatId: $chatId) {
    id
    state
  }
}
```

WebSocket Subscriptions

Subscribe to Chat Updates

```
subscription ChatUpdates($chatId: ID!) {
  userChat(chatId: $chatId) {
    id
    name
    state
    messages {
      id
      author
      message
      status {
        state
      }
    }
  }
}
```

Subscribe to Message Streaming

```
subscription MessageStream($messageId: ID!) {
  messageStreaming(messageId: $messageId) {
    id
    message
    status {
      state
      message
    }
  }
}
```

Streaming updates:

```
// Update 1
{
  "id": "msg2",
  "message": "Based on",
  "status": { "state": "STREAMING" }
}

// Update 2
{
  "id": "msg2",
  "message": "Based on the document,",
  "status": { "state": "STREAMING" }
}

// Final
{
  "id": "msg2",
  "message": "<p>Based on the document, here are the key points...</p>",
  "status": { "state": "COMPLETED" }
}
```