Stock Analysis RAG Agent - UI Developer API Guide

Base URL: `https://your-api-domain.com` or `http://localhost:8000`

Quick Reference

Endpoint	Method	Purpose
`/api/v1/chat/`	POST	Send message, get complete response
`/api/v1/chat/stream`	POST	Send message, stream real-time activities
`/api/v1/history/sessions`	GET	List all chat sessions
`/api/v1/history/sessions/{id}`	GET	Get session with all messages
`/api/v1/history/sessions`	POST	Create new session
`/api/v1/history/sessions/{id}`	PATCH	Update session title
`/api/v1/history/sessions/{id}`	DELETE	Delete session

---

1. Chat Endpoints

Regular Chat (Complete Response)

POST `/api/v1/chat/`

Returns complete response after processing.

Request:

```json
{
  "message": "What are the top CSE banking stocks?",
  "session_id": "550e8400-e29b-41d4-a716-446655440000",
  "context_mode": "internal_only",
  "reasoning_mode": "quick",
  "create_alert": false
}
```

Fields:

- `message` (required): User's question (1-10000 chars)
- `session_id` (optional): UUID of existing session, `null` for new
- `context_mode`: `"web_internal"` or `"internal_only"` (default: `"web_internal"`)
- `reasoning_mode`: `"quick"` or `"deep"` (default: `"quick"`)
- `create_alert`: `true` to create alert from conversation (default: `false`)

Response:

json


```
{
 "session_id": "550e8400-e29b-41d4-a716-446655440000",
 "message": {
  "id": "660e8400-e29b-41d4-a716-446655440001",
  "session_id": "550e8400-e29b-41d4-a716-446655440000",
  "role": "assistant",
  "content": "Based on current CSE market conditions...",
  "timestamp": "2024-02-06T10:30:00Z",
  "tokens_used": 450,
  "metadata": {
   "prompt_tokens": 200,
   "completion_tokens": 250,
   "total_tokens": 450
  }
 },
 "activity_feed": [
  {
   "type": "search",
   "message": "Searching internal knowledge base",
   "timestamp": "2024-02-06T10:29:58Z",
   "metadata": {}
  },
  {
   "type": "search",
   "message": "Found 5 relevant documents",
   "timestamp": "2024-02-06T10:29:59Z",
   "metadata": {}
  },
  {
   "type": "generation",
   "message": "Generating response",
   "timestamp": "2024-02-06T10:30:00Z",
   "metadata": {}
  }
 ],
 "alert_created": null
}
```


Activity Types:

- `"search"` - Searching knowledge base
- `"analysis"` - Analyzing data
- `"generation"` - Generating response
- `"alert_creation"` - Creating alert
- `"error"` - Error occurred

Streaming Chat (Real-time Activity Feed)

POST `/api/v1/chat/stream`

Returns Server-Sent Events (SSE) stream with real-time updates.

Request: Same as regular chat

Response: SSE Stream

Event Format:

```javascript
// Activity event
data: {"type":"activity","data":{"activity_type":"search","message":"Searching knowledge base","timestamp":"2024-02-06T10:00:00Z"}

// Response event
data: {"type":"response","data":{"session_id":"...","message":{...},"alert_created":null}}

// Done event
data: {"type":"done"}

// Error event
data: {"type":"error","message":"Error description"}
```

Frontend Integration:

```javascript
const eventSource = new EventSource('/api/v1/chat/stream', {
  method: 'POST',
  headers: { 'Content-Type': 'application/json' },
  body: JSON.stringify({ message: "Your question", context_mode: "internal_only" })
});

eventSource.onmessage = (event) => {
  const data = JSON.parse(event.data);

  if (data.type === 'activity') {
    // Update activity feed UI
    addActivity(data.data);
  } else if (data.type === 'response') {
    // Show final response
    showResponse(data.data);
  } else if (data.type === 'done') {
    eventSource.close();
  }
};
```

---

2. History Endpoints

List Sessions

GET `/api/v1/history/sessions?page=1&page_size=20`

Query Parameters:

- `page`: Page number (default: 1)
- `page_size`: Items per page (default: 20, max: 100)
- `include_inactive`: Include archived sessions (default: false)

Response:

json


```
{
  "sessions": [
    {
      "id": "550e8400-e29b-41d4-a716-446655440000",
      "title": "What are the top performing stocks?",
      "created_at": "2024-02-06T10:25:00Z",
      "updated_at": "2024-02-06T10:35:00Z",
      "message_count": 8,
      "last_message_preview": "Based on the analysis..."
    }
  ],
  "total": 25,
  "page": 1,
  "page_size": 20
}
```

Get Session Details

GET `/api/v1/history/sessions/{session_id}`

Response:

json


```
{
 "id": "550e8400-e29b-41d4-a716-446655440000",
 "title": "Stock analysis session",
 "created_at": "2024-02-06T10:25:00Z",
 "updated_at": "2024-02-06T10:35:00Z",
 "is_active": true,
 "metadata": {},
 "message_count": 8,
 "messages": [
  {
   "id": "660e8400-e29b-41d4-a716-446655440001",
   "session_id": "550e8400-e29b-41d4-a716-446655440000",
   "role": "user",
   "content": "What is JKH stock price?",
   "timestamp": "2024-02-06T10:25:00Z",
   "tokens_used": 0,
   "metadata": {}
  },
  {
   "id": "660e8400-e29b-41d4-a716-446655440002",
   "session_id": "550e8400-e29b-41d4-a716-446655440000",
   "role": "assistant",
   "content": "John Keells Holdings (JKH.N0000) is currently...",
   "timestamp": "2024-02-06T10:25:05Z",
   "tokens_used": 320,
   "metadata": {}
  }
 ]
}
```

Create Session

POST /api/v1/history/sessions

Request:

json


```
{
 "title": "Banking sector analysis",
 "metadata": {}
}
```


Response:

```json
{
  "id": "550e8400-e29b-41d4-a716-446655440000",
  "title": "Banking sector analysis",
  "created_at": "2024-02-06T10:25:00Z",
  "updated_at": "2024-02-06T10:25:00Z",
  "is_active": true,
  "metadata": {},
  "message_count": 0
}
```

Update Session

PATCH `/api/v1/history/sessions/{session_id}`

Request:

```json
{
  "title": "Updated title",
  "is_active": true
}
```

Delete Session

DELETE `/api/v1/history/sessions/{session_id}`

Response:

```json
{
  "message": "Session deleted successfully",
  "session_id": "550e8400-e29b-41d4-a716-446655440000"
}
```

---

3. Ticker Format

CRITICAL: All ticker symbols must be in full CSE format.

Format: `{SYMBOL}.{EXCHANGE_CODE}`

Examples:

- ✅ `JKH.N0000` - John Keells Holdings
- ✅ `COMB.N0000` - Commercial Bank
- ✅ `DIAL.N0000` - Dialog Axiata
- ❌ `JKH` - Incomplete (will cause errors)

Exchange Codes:

- `.N0000` - Main Board
- `.X0000` - Diri Savi Board

Complete ticker list available in `ticker_mapping.json`

---

4. Error Handling

Error Response Format:

```
json

{
 "detail": "Error message"
}
```

Common Status Codes:

- `400` - Bad Request (invalid parameters)
- `404` - Not Found (session/resource not found)
- `429` - Too Many Requests (rate limit exceeded)
- `500` - Internal Server Error

Rate Limit Response:

```
json

{
 "error": "Rate limit exceeded",
 "detail": "Too many requests. Please try again later."
}
```

Rate Limits:

- Chat endpoints: 20/minute
- Read endpoints: 60/minute
- Write endpoints: 30/minute

5. Complete Integration Example

javascript

```javascript
// 1. Start new conversation
async function startConversation() {
  const response = await fetch('/api/v1/chat/', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({
      message: "Analyze JKH stock",
      session_id: null,  // New session
      context_mode: "internal_only",
      reasoning_mode: "deep"
    })
  });

  const data = await response.json();

  // Save session ID for continuation
  const sessionId = data.session_id;

  // Display response
  displayMessage(data.message.content);

  // Show activities
  data.activity_feed.forEach(activity => {
    displayActivity(activity);
  });

  return sessionId;
}

// 2. Continue conversation
async function continueConversation(sessionId, message) {
  const response = await fetch('/api/v1/chat/', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({
      message: message,
      session_id: sessionId,  // Use existing session
      context_mode: "internal_only",
      reasoning_mode: "quick"
    })
  });

  const data = await response.json();
  displayMessage(data.message.content);
}
```

```javascript
// 3. Load chat history
async function loadChatHistory() {
  const response = await fetch('/api/v1/history/sessions?page=1&page_size=20');
  const data = await response.json();

  // Display sessions in sidebar
  data.sessions.forEach(session => {
    displaySessionInSidebar(session);
  });
}

// 4. Load specific session
async function loadSession(sessionId) {
  const response = await fetch(`/api/v1/history/sessions/${sessionId}`);
  const data = await response.json();

  // Display all messages
  data.messages.forEach(msg => {
    displayMessage(msg.content, msg.role);
  });

  return data;
}

// 5. Real-time streaming (recommended for better UX)
function streamChat(message, sessionId) {
  const eventSource = new EventSource('/api/v1/chat/stream', {
    method: 'POST',
    headers: { 'Content-Type': 'application/json' },
    body: JSON.stringify({
      message: message,
      session_id: sessionId,
      context_mode: "internal_only",
      reasoning_mode: "quick"
    })
  });

  eventSource.onmessage = (event) => {
    const data = JSON.parse(event.data);

    switch(data.type) {
      case 'activity':
        // Update activity feed in real-time
        addActivityToFeed(data.data.message, data.data.activity_type);
        break;

      case 'response':
        // Show final response
```

```
      displayMessage(data.data.message.content);
      saveSessionId(data.data.session_id);
      break;

    case 'done':
      eventSource.close();
      hideLoadingIndicator();
      break;

    case 'error':
      showError(data.message);
      eventSource.close();
      break;
  }
};

eventSource.onerror = (error) => {
  console.error('Stream error:', error);
  eventSource.close();
  showError('Connection lost. Please try again.');
};
}
```

---

6. UI Flow Recommendations

Starting a Chat

1. User types message
2. Set `session_id: null` for first message
3. Send to `/api/v1/chat/stream` for real-time feedback
4. Display activities as they arrive
5. Save returned `session_id` for continuation

Continuing a Chat

1. User types message
2. Use saved `session_id`
3. Send to same endpoint
4. Display new message in chat

Loading History

1. On app load, call `/api/v1/history/sessions`
2. Display sessions in left sidebar (like your screenshot)
3. On session click, call `/api/v1/history/sessions/{id}`
4. Display all messages
5. Continue chat with that `session_id`

Activity Feed Display

```javascript
// Map activity types to icons/colors
const activityIcons = {
  'search': '🔍',
  'analysis': '📊',
  'generation': '✨',
  'alert_creation': '🔔',
  'error': '⚠️'
};

function displayActivity(activity) {
  const icon = activityIcons[activity.type];
  const element = `
    <div class="activity ${activity.type}">
      <span class="icon">${icon}</span>
      <span class="message">${activity.message}</span>
    </div>
  `;
  activityFeedContainer.innerHTML += element;

  // Auto-scroll to bottom
  activityFeedContainer.scrollTop = activityFeedContainer.scrollHeight;
}
```

---

7. TypeScript Types (Optional)

typescript

```typescript
// Request types
interface ChatRequest {
  message: string;
  session_id?: string | null;
  context_mode?: 'web_internal' | 'internal_only';
  reasoning_mode?: 'quick' | 'deep';
  create_alert?: boolean;
}

// Response types
interface Message {
  id: string;
  session_id: string;
  role: 'user' | 'assistant' | 'system';
  content: string;
  timestamp: string;
  tokens_used: number;
  metadata: Record<string, any>;
}

interface ActivityFeedItem {
  type: 'search' | 'analysis' | 'generation' | 'alert_creation' | 'error';
  message: string;
  timestamp: string;
  metadata: Record<string, any>;
}

interface ChatResponse {
  session_id: string;
  message: Message;
  activity_feed: ActivityFeedItem[];
  alert_created?: any;
}

interface Session {
  id: string;
  title: string;
  created_at: string;
  updated_at: string;
  message_count: number;
  last_message_preview?: string;
}
```

8. Health Check

GET `/health`

Check API status before starting.

Response:

```json
{
  "status": "healthy",
  "app": "Stock Analysis RAG Agent",
  "version": "1.0.0",
  "environment": "production"
}
```

Need Help?

- **API Issues:** Contact backend team
- **Rate Limits:** Configure in environment variables
- **CORS Issues:** Add your domain to `ALLOWED_ORIGINS`

Last Updated: February 2024