A2A CR Box - Enterprise Content Discovery Agent

A professional enterprise content discovery assistant powered by Gemini, specializing in Box content search and enterprise document management.

🚀 Features

Core Capabilities

- **Box Content Search**: Find files, documents, and folders in Box
- **Box AI Ask**: Ask intelligent questions about specific file content
- **Box Hub Ask**: Automatically discover and use the most relevant Box Hub
- **Enterprise Content Discovery**: Locate specific documents, regulatory files, reports, and business content
- **Professional Communication**: Business-appropriate responses with enterprise focus

Tools Available

1. **Box Search** - Find documents and files across your Box enterprise
2. **Box AI Ask** - Ask questions about specific file content using Box AI
3. **Box Hub Ask** - Automatically discover and use the most relevant knowledge hub

🏗️ Architecture

- **Google ADK (Agent Development Kit)**: Powers the GeminiAgent
- **Box Python SDK**: Handles Box API authentication and operations
- **JWT Authentication**: Secure server-to-server authentication with Box
- **Cloud Run**: Deployed as a scalable containerized service
- **AgentSpace Integration**: Ready for Discovery Engine registration

🔧 Setup

Prerequisites

- Python 3.9+
- Google Cloud Platform account
- Box Developer account with JWT app credentials

Installation

1. **Clone the repository**

   ```
   git clone <your-repo-url>
   cd A2A_CR_Box
   ```

2. **Install dependencies**

   ```
   pip install -r requirements.txt
   ```

3. **Configure Box JWT Authentication**

   Option A: Local JWT Config (Development)

   - Copy `box_jwt_config.example.json` to `box_jwt_config.json`
   - Fill in your Box JWT app credentials:
     - `clientID`
     - `clientSecret`
     - `enterpriseID`
     - `publicKeyID`
     - `privateKey`
     - `passphrase`

   Option B: Google Secret Manager (Production - Recommended)

   This is the secure, production-ready approach that stores your JWT credentials in Google Secret Manager instead of local files.

   Step 1: Enable Secret Manager API

   ```
   gcloud services enable secretmanager.googleapis.com --project=your-project-id
   ```

   Step 2: Create Box JWT Secrets

```
# Create secret for Box Client ID
echo -n "your_client_id_here" | gcloud secrets create box-client-id --data-file=- -

# Create secret for Box Client Secret
echo -n "your_client_secret_here" | gcloud secrets create box-client-secret --data-

# Create secret for Box Public Key ID
echo -n "your_public_key_id_here" | gcloud secrets create box-public-key-id --data-

# Create secret for Box Private Key (the entire PEM content)
gcloud secrets create box-private-key --data-file=path/to/your/private_key.pem --pr

# Create secret for Box Private Key Passphrase
echo -n "your_passphrase_here" | gcloud secrets create box-private-key-passphrase -

# Create secret for Box Enterprise ID
echo -n "your_enterprise_id_here" | gcloud secrets create box-enterprise-id --data-
```

Step 3: Use the Setup Script (Easiest Method)

Save this script as `box_jwt_secrets_setup.sh` :

```bash
#!/bin/bash
# box_jwt_secrets_setup.sh

echo "🔐 Setting up Box JWT secrets in Google Secret Manager..."
echo ""

# Prompt for credentials
read -p "Enter Box Client ID: " BOX_CLIENT_ID
read -p "Enter Box Client Secret: " BOX_CLIENT_SECRET
read -p "Enter Box Public Key ID: " BOX_PUBLIC_KEY_ID
read -p "Enter Box Enterprise ID: " BOX_ENTERPRISE_ID
read -s -p "Enter Box Private Key Passphrase: " BOX_PASSPHRASE
echo ""
read -p "Enter path to Box Private Key PEM file: " BOX_PRIVATE_KEY_PATH

# Set project
PROJECT_ID="your-project-id"

echo ""
echo "📝 Creating secrets in project: $PROJECT_ID"
echo ""

# Create secrets
echo "Creating box-client-id..."
echo -n "$BOX_CLIENT_ID" | gcloud secrets create box-client-id --data-file=- --proj

echo "Creating box-client-secret..."
echo -n "$BOX_CLIENT_SECRET" | gcloud secrets create box-client-secret --data-file=

echo "Creating box-public-key-id..."
echo -n "$BOX_PUBLIC_KEY_ID" | gcloud secrets create box-public-key-id --data-file=

echo "Creating box-enterprise-id..."
echo -n "$BOX_ENTERPRISE_ID" | gcloud secrets create box-enterprise-id --data-file=

echo "Creating box-private-key-passphrase..."
echo -n "$BOX_PASSPHRASE" | gcloud secrets create box-private-key-passphrase --data

echo "Creating box-private-key..."
gcloud secrets create box-private-key --data-file="$BOX_PRIVATE_KEY_PATH" --project

echo ""
```

```
echo "✅ All Box JWT secrets created successfully!"
echo "🔒 Your credentials are now securely stored in Google Secret Manager"
```

Make it executable and run:

```
chmod +x box_jwt_secrets_setup.sh
./box_jwt_secrets_setup.sh
```

Step 4: Update Requirements.txt

Add the Secret Manager client library:

```
google-cloud-secret-manager>=2.0.0
```

Step 5: Update .gitignore

Now you can safely ignore the JWT config file:

```
# Box Configuration (Sensitive)
box_jwt_config.json
*.jwt
*.pem
*.key
*.p12
*.pfx
```

4. **Set environment variables**

```
export GOOGLE_CLOUD_PROJECT="your-project-id"
export GOOGLE_CLOUD_LOCATION="us-central1"
export GOOGLE_GENAI_USE_VERTEXAI="TRUE"
export MODEL="gemini-2.5-flash"
```

🚀 Deployment

Local Development

```
python -m uvicorn agent_executor:app --reload
```

Cloud Run Deployment

```
./deploy.sh <project-id> <service-name>
```

AgentSpace Registration

After deployment, register your agent in Google Cloud Discovery Engine:

```
curl -X POST \
  -H "Authorization: Bearer $(gcloud auth print-access-token)" \
  -H "Content-Type: application/json" \
  "https://discoveryengine.googleapis.com/v1alpha/projects/<PROJECT_ID>/locations/globa
  -d '{
    "name": "Box_Search_Agent",
    "displayName": "Box Search Agent",
    "description": "Enterprise content discovery agent for Box",
    "a2aAgentDefinition": {
      "jsonAgentCard": "{\"provider\": {\"url\": \"<YOUR_CLOUD_RUN_URL>\"},\"name\": \"
    }
  }'
```

📖 Usage Examples

Box Content Search

```
User: "Find regulatory documents"
Agent: [Searches Box and returns organized results with file counts and details]
```

Box AI Ask

```
User: "What are the key points in Capital Call Notice.pdf?"
Agent: [Uses Box AI to analyze the specific file and provide insights]
```

Box Hub Ask

```
User: "What are our company policies?"
Agent: [Automatically discovers relevant hubs and provides answers from the best knowle
```

🔒 Security

Authentication Methods

- **JWT Authentication**: Secure server-to-server authentication with Box
- **Environment Variables**: Sensitive configuration stored securely
- **Box API Permissions**: Minimal required permissions for enterprise access
- **Cloud Run Security**: No unauthenticated access, secure by default

Google Secret Manager Benefits

Using Google Secret Manager for JWT credentials provides several security advantages:

1. 🔒 **Enhanced Security**: Credentials stored in Google's secure Secret Manager instead of local files
2. 🚀 **Easier Deployment**: No need to include sensitive files in deployments
3. 📝 **Version Control**: Secrets can be versioned and rotated without code changes
4. 👥 **Access Control**: IAM controls who can access secrets
5. 🔄 **Easy Updates**: Update secrets without redeploying code
6. 🔍 **Audit Logging**: Track who accessed secrets and when
7. 🌍 **Multi-Region**: Secrets can be replicated across regions for availability

Best Practices

- Use Google Secret Manager for production deployments
- Keep local JWT config only for development/testing
- Regularly rotate JWT credentials
- Use least-privilege access for secret permissions
- Monitor secret access through Cloud Logging

🛠️ Development

Project Structure

```
A2A_CR_Box/
├── agent_executor.py      # Main A2A executor
├── gemini_agent.py        # Gemini agent with tools
├── box_auth.py           # Box JWT authentication
├── box_search.py         # Box content search
├── box_ai_ask.py         # Box AI file analysis
├── box_hub_ask.py        # Box Hub discovery and querying
├── requirements.txt      # Python dependencies
├── deploy.sh            # Cloud Run deployment script
└── README.md            # This file
```

Adding New Tools

1. Create your tool function in a new Python file
2. Import it in `gemini_agent.py`
3. Add it to the `tools` list
4. Update the agent instructions
5. Add corresponding skills

📝 License

This project is licensed under the Box TOS. This project is delivered as is with no commitments to maintanence.

🤝 Contributing

1. Fork the repository
2. Create a feature branch
3. Make your changes
4. Add tests if applicable
5. Submit a pull request

📞 Support

For issues and questions:

- Check the logs in Google Cloud Console
- Review Box API documentation
- Check Google ADK documentation

🔄 Version History

- **v1.0.0**: Initial release with Box search and AI capabilities
- **v1.1.0**: Added Box Hub Ask functionality
- **v1.2.0**: Enhanced enterprise focus and professional communication