Software Architecture: AI-Powered Image Captioning and Social Media Posting Software

Table of Contents

Product Description

Overview

The Captioning App, formerly known as CapgenAIze, harnesses the power of artificial intelligence, large language models (LLM), and advanced image recognition to generate captions for images and videos (Reels). These captions are optimized for sharing on social media platforms such as Instagram, Facebook, X (formerly Twitter), and LinkedIn. The app is also ideal for product marketing purposes.

Features

- **AI-Powered Caption Generation:** Utilizes cutting-edge AI and LLM to create relevant and engaging captions.
- **Image and Video Recognition:** Analyzes both images and videos (up to 30 seconds) to understand the content and context.
- **Social Media Optimization:** Generates captions tailored for various social media platforms, ensuring maximum engagement.

How to Use

1. **Upload Media:**
   - **Image:** Simply upload your image.
   - **Video:** Upload a video (up to 30 seconds in length).
2. **Select Preferences:**
   - **Tone:** Choose from various tones such as Artistic, Professional, etc.
   - **Caption Size:** Select the desired size of the caption: small, medium, or large.
   - **Context (Optional):** Provide additional context for the image or video if desired.
3. **Caption Generation:**
   - **For Images:** Captions are generated within seconds.
   - **For Videos:** Captions are generated within 5-10 seconds.

Benefits

- **Efficiency:** Quickly generate captions without manual effort.
- **Customization:** Tailor captions to fit different tones and sizes to match your needs.
- **Engagement:** Enhance social media posts with AI-driven, relevant captions.

Applications

- **Social Media:** Perfect for Instagram, Facebook, X, LinkedIn, and more.
- **Product Marketing:** Create compelling captions for marketing materials.

The Captioning App streamlines the process of creating captions, enabling users to focus on sharing and promoting their content more effectively.

Development Requirements

To develop the Captioning App, the following requirements must be satisfied:

Software Requirements

1. **Python Version:**
   - Python 3.10 (All dependencies are not compatible with Python 3.11)
2. **Python Libraries:**
   - Flask==2.2.3
   - openai==0.27.2
   - opencv_python==4.7.0.72
   - Pillow==9.5.0
   - streamlit==1.21.0

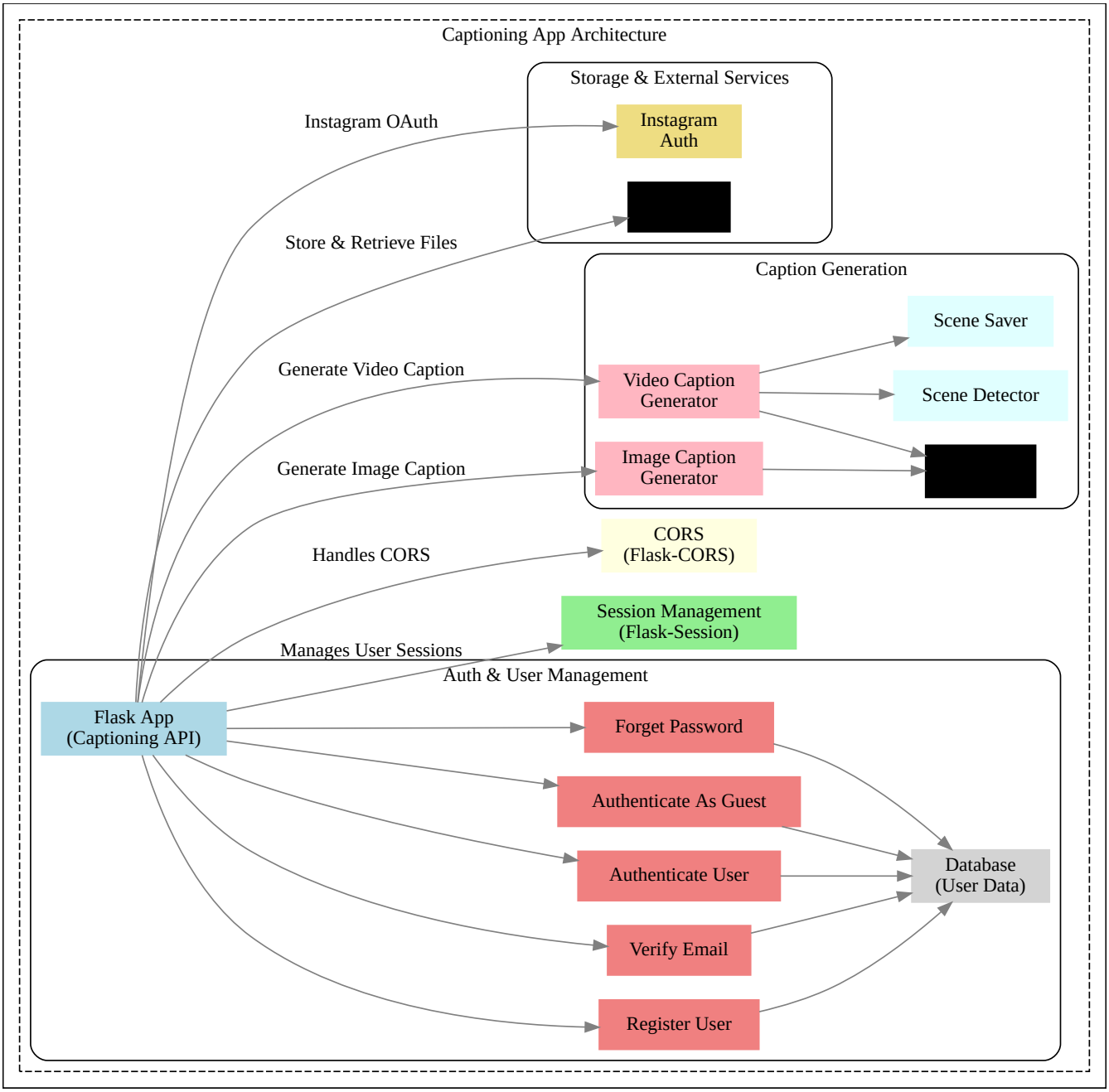
- torch==2.0.0
- transformers==4.27.4
- boto3
- pytest
- pytest-mock
- dnspython
- email-validator
- PyJWT

System Requirements

1. **Docker:** To containerize and manage the application.
2. **curl:** For transferring data with URLs.

Ensure all the above dependencies and tools are installed and properly configured in your development environment to successfully build and deploy the Captioning App.

Basic Diagram



Description

Flask App (Captioning API) - Light Blue

- The core component managing all application endpoints and coordinating requests and responses.
- Handles routes for user registration, login, media uploads, and caption generation.

Session Management (Flask-Session) - Light Green

- Manages user sessions to maintain state across different requests.
- Ensures secure session handling.

CORS (Flask-CORS) - Light Yellow

- Manages Cross-Origin Resource Sharing to control resource access across different domains.

Auth & User Management - Light Coral

- **Register User:** Handles user registration.
- **Verify Email:** Manages email verification.
- **Authenticate User:** Manages user login.
- **Authenticate As Guest:** Allows guest login.
- **Forget Password:** Handles password reset requests.
- **Database:** Stores user information and authentication data.

Caption Generation - Light Pink

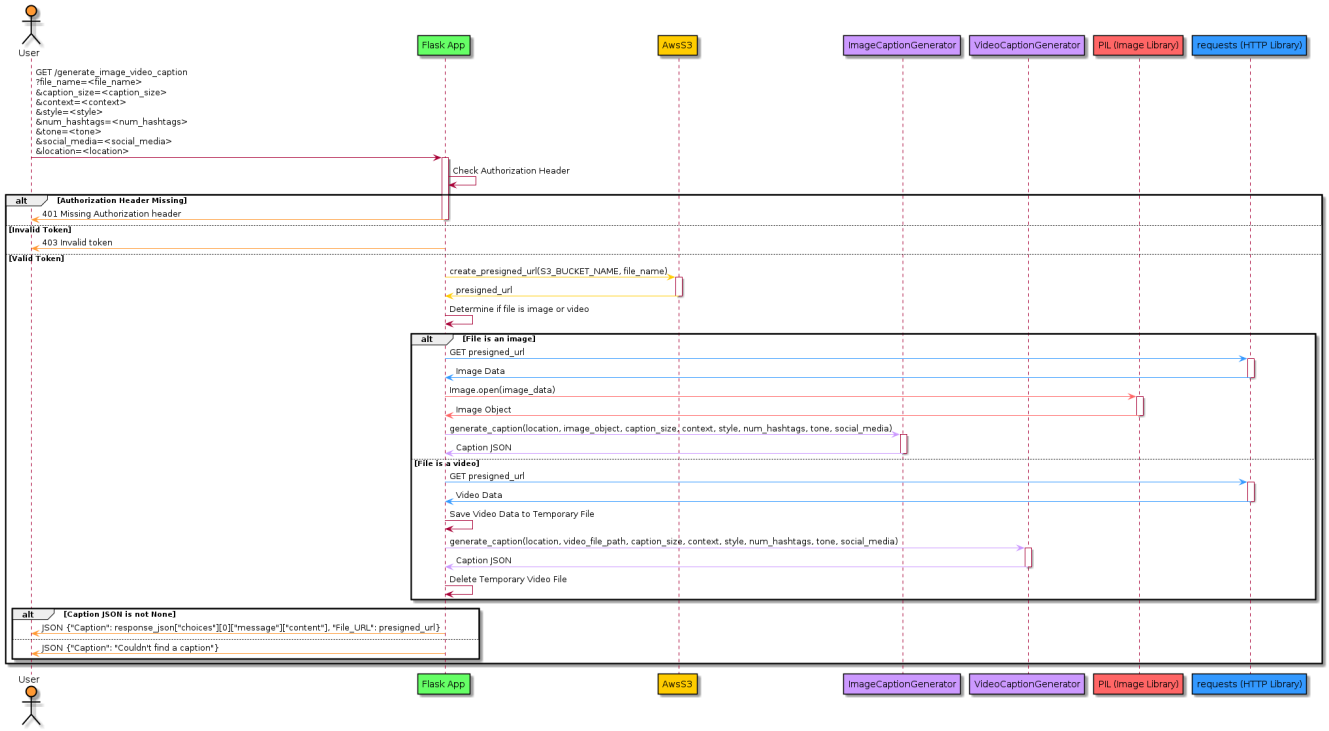
- **Image Caption Generator:** Generates captions for images.
- **Video Caption Generator:** Generates captions for videos.
- **Scene Detector:** Detects scenes in videos for captioning.
- **Scene Saver:** Saves detected scenes.
- **Chatbot (OpenAI):** Provides AI-driven language models for caption generation.

Storage & External Services - Light Orange and Light Goldenrod

- **AWS S3:** Stores and retrieves media files.
- **Instagram Auth:** Manages OAuth authentication with Instagram.

This enhanced diagram and description offer a comprehensive view of the Captioning App's architecture, highlighting the interactions and functionalities of each component.

Sequence Diagram



Detailed Explanation of the Sequence Diagram

1. **User Request**:
   - The user sends a GET request to the `/generate_image_video_caption` endpoint on the `FlaskApp` with parameters including `file_name`, `caption_size`, `context`, `style`, `num_hashtags`, `tone`, `social_media`, and `location`.
2. **FlaskApp Checks Authorization Header**:
   - The `FlaskApp` checks the authorization header of the request.

```
**Authorization Missing**:
- If the authorization header is missing, `FlaskApp` responds with a `401
Missing Authorization header` error and deactivates itself.
```

```
**Invalid Token**:
- If the authorization token is invalid, `FlaskApp` responds with a `403 Invalid
token` error and deactivates itself.
```

```
**Valid Token**:
- If the authorization token is valid, `FlaskApp` proceeds to create a presigned
URL for the file from `AwsS3`.
```

3. **Creating Presigned URL**:
   - `FlaskApp` requests `AwsS3` to create a presigned URL for the specified file.

- AwsS3 returns the presigned URL to FlaskApp.

4. **Determine File Type (Image or Video)**:
   - FlaskApp determines if the file is an image or a video based on its extension.

```
**File is an Image**:
- `FlaskApp` sends a GET request to the presigned URL to fetch the image data
using `Requests`.
- `Requests` returns the image data to `FlaskApp`.
- `FlaskApp` uses `PIL` to open the image data and convert it into an image
object.
- `PIL` returns the image object to `FlaskApp`.
- `FlaskApp` sends the image object to `ImageCaptionGenerator` to generate a
caption.
- `ImageCaptionGenerator` returns the generated caption JSON to `FlaskApp`.
```

```
**File is a Video**:
- `FlaskApp` sends a GET request to the presigned URL to fetch the video data
using `Requests`.
- `Requests` returns the video data to `FlaskApp`.
- `FlaskApp` saves the video data to a temporary file.
- `FlaskApp` sends the video file path to `VideoCaptionGenerator` to generate a
caption.
- `VideoCaptionGenerator` returns the generated caption JSON to `FlaskApp`.
- `FlaskApp` deletes the temporary video file.
```

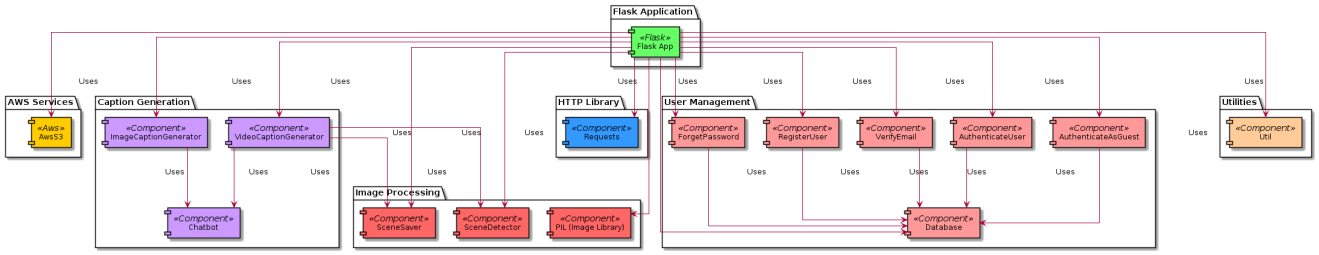
5. **Return Caption to User**:
   - If the caption JSON is not None, FlaskApp responds to the user with the generated caption and the presigned URL.
   - If the caption JSON is None, FlaskApp responds to the user with an error message stating "Couldn’t find a caption".

6. **End**:
   - FlaskApp deactivates itself after completing the process.

Summary

The sequence diagram outlines the process of generating captions for images or videos. The user sends a request to the FlaskApp, which checks for authorization and then interacts with AwsS3 to fetch the file. Depending on the file type, the FlaskApp uses either ImageCaptionGenerator or VideoCaptionGenerator to generate a caption and then returns the caption to the user.

Component Diagram — Complete Application



Usecase Diagram