Project Outline: Integral Calculator with Manim Animation

This project aims to create a system that takes a mathematical integral written in LaTeX, computes the solution with human-style steps, and then automatically generates an animated video of the solution process using Manim. The system combines symbolic computation with programmatic animation to provide clear, visual explanations of integration techniques.

Key Components:

- Input Parsing: Accept a LaTeX expression of an integral. Convert LaTeX into a symbolic form that can be processed by a Computer Algebra System (CAS).
- Symbolic Computation: Use SymPy (Python-based CAS) to compute the integral. Employ SymPy’s 'manualintegrate' module to extract step-by-step integration methods (substitution, integration by parts, etc.).
- Step Representation: Transform SymPy’s integration steps into a sequence of human-readable forms (LaTeX).
- Manim Animation Engine: Generate Manim scenes that animate each integration step. Start with the integral expression, apply transformations, and finish with the antiderivative.
- Video Export: Automatically render the Manim animation into an MP4 video, producing a visual explanation of the integral.

Workflow:

1. User provides an integral written in LaTeX.
2. The LaTeX is parsed into a symbolic expression (SymPy).
3. SymPy computes the integral and provides a breakdown of steps.
4. Each step is converted back to LaTeX for display.
5. A Manim script is automatically generated to animate the process.
6. Manim renders the animation into a video file.

Possible Extensions:

- Handle definite integrals by showing area under the curve and evaluating bounds step by step.
- Allow user to customize colors, pacing, and visual style of animations.
- Deploy as a web app (Flask/Streamlit) where users type in integrals and download videos.
- Add numeric fallback for definite integrals if symbolic solution is not found.

This project bridges symbolic mathematics and visual storytelling. It makes complex integration techniques more approachable by transforming symbolic steps into intuitive animations. By leveraging SymPy for computation and Manim for rendering, the system provides a practical and educational tool for students and teachers alike.