BCSE498J Project-II

VIDEO EDITING USING ARTIFICIAL INTELLIGENCE

Submitted in partial fulfillment of the requirements for the degree of

Bachelor of Technology

in

Computer Science and Engineering

by

21BCE3417 DIVAKAR REDDY. G

21BCE3738 GURUSHREKAR. M

Under the Supervision of

Dr. KARTHIK. K

Assistant Professor Senior Grade 1

School of Computer Science and Engineering (SCOPE)

April 2025

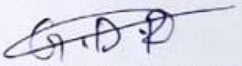
DECLARATION

I hereby declare that the project entitled **"VIDEO EDITING USING ARTIFICIAL INTELLIGENCE"** submitted by me, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering* to VIT is a record of bonafide work carried out by me under the supervision of Prof. Dr. Karthik k.

I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree ordiploma in this institute or any other institute or university.

Place :Vellore

Date :15/04/2025



Signature of the Candidate

CERTIFICATE

This is to certify that the project entitled " **VIDEO EDITING USING ARTIFICIAL INTELLIGENCE**" submitted by Gangireddygari Divakar Reddy (21BCE3417), **School of Computer Science and Engineering**, VIT, for the award of the degree of *Bachelor of Technology in Computer Science and Engineering*, is a record of bonafide work carried out by him under my supervision during Winter Semester 2024-2025, as per the VIT code of academic and research ethics.

The contents of this report have not been submitted and will not be submitted either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university. The project fulfills the requirements and regulations of the University and in my opinion meets the necessary standards for submission.

Place : Vellore

Date : 15/04/2025

VELLORE INSTITUTE OF TECHNOLOGY (VIT) School of Computer Science & Engineering (SCOPE) Vellore-632 014 T.N., India

21/04/2025

Signature of the Guide

21/04/2025

Internal Examiner

External Examiner

Jagadesh.P

Dr. UMADEVI K S

Head – Computer Science and Engineering

EXECUTIVE SUMMARY

Artificial Intelligence (AI) is revolutionizing the video editing industry by automating time-consuming tasks, enhancing creative workflows, and enabling high-quality content production at scale. AI-powered video editing tools leverage machine learning, computer vision, and natural language processing to transform raw footage into polished videos with minimal human intervention.

Key capabilities of AI in video editing include automated scene detection, smart trimming, background noise removal, object tracking, and facial recognition. Advanced tools can even generate subtitles, suggest edits based on content analysis, and apply filters or transitions automatically. These features significantly reduce editing time and enable creators to focus on storytelling and creativity rather than technical details.

AI also supports content personalization and optimization for various platforms. For example, AI can automatically crop and reframe videos for different aspect ratios (e.g., YouTube, Instagram, TikTok), generate highlight reels from long-form content, and even produce multilingual voiceovers or captions using speech synthesis and translation models. The integration of generative AI, such as deep learning models like GPT and diffusion models, further enhances creative possibilities. These tools can generate scripts, storyboard sequences, or even synthesize visual effects and animations, empowering creators and businesses to produce engaging content faster and more cost-effectively.

Industries ranging from marketing and entertainment to education and real estate are adopting AI video editing to scale their content strategies, improve efficiency, and deliver consistent branding. Major platforms like Adobe Premiere Pro, Runway ML, Descript, and Pictory are already incorporating AI features to streamline workflows and enhance user experience.

In conclusion, AI-powered video editing is transforming the content creation landscape by making professional-grade editing accessible, faster, and more intelligent. As technology advances, we can expect even more intuitive, real-time, and creative AI editing solutions to emerge, reshaping how we produce and consume visual content.

ACKNOWLEDGEMENTS

I am deeply grateful to the management of Vellore Institute of Technology (VIT) for providing me with the opportunity and resources to undertake this project. Their commitment to fostering a conducive learning environment has been instrumental in my academic journey. The support and infrastructure provided by VIT have enabled me to explore and develop my ideas to their fullest potential.

My sincere thanks to Dr. Jaisankar N, Dean - School of Computer Science and Engineering (SCOPE), for his unwavering support and encouragement. His leadership and vision have greatly inspired me to strive for excellence.

I express my profound appreciation to Dr. Umadevi K S, the Head of the Scope, for her insightful guidance and continuous support. Her expertise and advice have been crucial in shaping throughout the course. Her constructive feedback and encouragement have been invaluable in overcoming challenges and achieving goals.

I am immensely thankful to my project supervisor, Dr. Karthik K, for his dedicated mentorship and invaluable feedback. His patience, knowledge, and encouragement have been pivotal in the successful completion of this project. My supervisor's willingness to share his expertise and provide thoughtful guidance has been instrumental in refining my ideas and methodologies. His support has not only contributed to the success of this project but has also enriched my overall academic experience.

Thank you all for your contributions and support.

G. Divakar Reddy

Name of the Candidate

TABLE OF CONTENTS

List of Figures

List of Tables

List of Abbreviations

AI	Artificial Intelligence
ASR	Automatic Speech Recognition
BGM	Back Ground Music
FFMPEG	Fast Forward MPEG (video processing tool)
GPU	Graphics Processing Unit
HDR	High Dynamic Range
MP4	MPEG-4 Video Format
NLP	Natural Language Processing
SFX	Sound Effects
UI	User Interface
VQA	Video Quality Assessment
VSR	Video Super Resolution

Symbols and Notations

	Video Content
X	Input video
Y	Target/output video
	Trimming/Cutting Scenes
	Subtitles or transcriptions
	Adding background music
	Blur the face
	Style Transfer

1. INTRODUCTION

Video editing has long been a time-consuming and complex process, requiring skilled professionals to cut, organize, enhance, and transform raw footage into a polished final product. However, with the rapid development of Artificial Intelligence (AI) and machine learning technologies, video editing is undergoing a profound transformation. AI is making it possible to automate many aspects of video production, enabling faster workflows, enhancing creativity, and reducing the need for manual intervention in tasks such as cutting, scene recognition, and audio transcription.

AI-based video editing refers to the application of machine learning, computer vision, and other AI-driven technologies to facilitate or completely automate various stages of video production. By using large datasets and sophisticated algorithms, AI can analyze video content in ways that were once thought to be impossible, offering editors and creators tools that improve productivity, accuracy, and even the creative process itself.

The power of AI in video editing lies in its ability to not only reduce the workload for video editors but also to unlock new creative possibilities. For instance, AI-powered tools can analyze facial expressions, gestures, and even voice tones to suggest video edits that align with the emotional tone or narrative of a project. AI systems can also be personalized, learning from a user's editing style and preferences to provide more tailored suggestions, which can improve editing workflows over time.

While AI promises to streamline and enhance the video editing process, it is still an evolving field, and challenges remain in achieving high-quality outputs across diverse genres and production contexts. Nonetheless, AI's integration into video editing tools represents a significant leap forward in content creation, democratizing video production and making it more accessible for a broader range of users, from independent creators to large-scale production studios.

1.1 BACKGROUND

The video editing industry has traditionally relied on human expertise to curate and refine raw footage into cohesive and engaging visual narratives. For decades, video editors have employed a variety of tools and techniques to piece together footage, add effects, adjust audio, and finalize the production. However, this process is time-consuming and requires meticulous attention to detail, especially as the volume and complexity of video content increase.

In recent years, advancements in Artificial Intelligence (AI) have led to significant innovations in the field of video editing, drastically transforming how editing tasks are approached. AI's ability to process large amounts of data, learn from patterns, and execute tasks with minimal human intervention has opened new possibilities for content creation, especially as the demand for video content has surged globally.

1.2 MOTIVATIONS

AI-powered video editing offers numerous advantages that significantly improve the editing process. It saves time by automating tasks like cutting scenes, colour correction, and shot selection, which would traditionally take hours. This allows editors to focus on more creative aspects of the project. It also boosts creativity by suggesting edits, applying filters, and generating effects based on the footage, providing fresh ideas and inspiration. Furthermore, AI enables personalized content, adjusting the video to appeal to different audiences by changing elements like length, style, and effects. This can make content more engaging and relevant.

One of the key benefits of AI is its ability to lower costs, as it automates tasks that would typically require a skilled editor, reducing the need for expensive professional services. For beginners**,** AI tools make it easier to create high-quality videos, offering a more accessible entry point into video editing without the steep learning curve. AI also ensures consistency, helping maintain a uniform style and tone throughout the video, which is especially beneficial for brands and creators aiming for a cohesive look across their content.

Additionally, AI can automatically add subtitles and translations, expanding the reach of content by making it accessible to a global audience. Its content recognition capabilities allow it to identify faces, objects, and specific scenes, which makes it easier for editors to quickly locate key moments and stay organized. With real-time editing capabilities, AI is enhancing workflows by enabling immediate adjustments, speeding up the process particularly valuable during live events or time-sensitive projects.

1.3 SCOPE OF THE PROJECT

The scope of a project focused on AI-powered video editing encompasses a wide range of objectives and features aimed at streamlining and enhancing the editing process. One major area is the integration of AI in editing tasks, where AI models can automatically detect and segment scenes, suggest or apply cuts, transitions, and color corrections, as well as enhance audio by cleaning up background noise and adjusting levels. AI can also analyze footage to select the best shots based on clarity and composition. In terms of personalization, AI can learn from a user's previous edits to recommend styles, effects, and tailor content to specific audiences, adjusting formats and styles for platforms like social media or ads. Efficiency tools such as real-time editing and automated subtitling and translation allow editors to make instant adjustments and ensure accessibility across multiple languages. Additionally, content recognition and organization through AI can tag objects and faces, making it easier to search and organize clips by automatically generating metadata.

By minimizing manual effort, the system utilizes advanced AI algorithms to perform core editing tasks such as scene detection, background music selection, object removal, colour correction, subtitle generation, and video summarization. Through the integration of deep learning models and computer vision techniques, the system can automatically segment scenes, identify key highlights, and apply cinematic enhancements. It also incorporates speech recognition to generate subtitles and uses mood analysis to pair appropriate background music with video content. Furthermore, it offers features like AI-based object removal using inpainting methods, automatic face detection and blurring for privacy, and optional voice dubbing or translation using voice cloning technologies. The project is not intended to support manual workflows or all video formats, and does not aim to provide real-time editing for lengthy videos. It is designed for content creators, educators, marketers, and professionals seeking efficient, AI-assisted editing solutions.

2. PROJECT DESCRIPTION AND GOALS

2.1 LITERATURE REVIEW

1. Automation in Video Editing Tasks

One of the most significant contributions of AI to video editing is automation. Traditionally, video editing involves numerous manual tasks such as cutting, colour correction, and sound enhancement, which require substantial time and skill. AI has shown promise in automating these tasks, significantly reducing the time required for editing. According to Liu (2020), AI-based systems can perform automatic scene detection and segmentation, making it easier to organize and edit raw footage. Further, Wang (2019) demonstrated that AI models can apply automatic colour grading to match the desired aesthetic of the video, making the editing process faster and more efficient.

In addition to scene segmentation, Xie (2020) explored how AI can assist in shot selection, enabling editors to select the best shots based on predefined quality metrics such as clarity, composition, and motion. This form of automation not only saves time but also enhances the quality of the final product, as AI can often make objective decisions that may be overlooked by human editors.

2. Enhancing Creativity with AI

AI has proven to be an effective tool for enhancing creativity in video editing. Zhang and Sun (2021) proposed an AI-driven tool that helps editors by suggesting edits and creating visual effects based on the footage's content. These tools can also generate transitions or add visual effects like filters and animations, offering creative options that may inspire new directions for a video. In some systems, the AI analyzes video footage and makes recommendations for edits that would enhance the storytelling aspect, like emphasizing emotional moments through visual or audio effects.

Zhou (2022) also highlighted the role of AI in creative content generation, specifically focusing on AI's ability to generate content suggestions that fit within the editor’s vision, while also providing ideas outside the editor’s initial concept. These AI suggestions can help overcome creative blocks and offer fresh perspectives that human editors might not consider on their own.

3. Personalized Video Editing for Diverse Audiences

Another significant area of development is AI's ability to tailor video content to different viewers. Li and Wu (2020) discussed how AI can adjust video formats, styles, and lengths based on user preferences or platform requirements. For instance, AI can adapt the video for various social media platforms such as Instagram, TikTok, or YouTube by adjusting aspects like video length, aspect ratio, and content style. This feature enhances audience engagement by ensuring that videos meet the specific expectations of different platforms.

Moreover, AI can help identify and optimize content for target audiences by analyzing data on viewer behaviour and preferences. Feng (2021) demonstrated how AI could optimize the editing process by adjusting the tone, style, and pacing based on what has been shown to resonate with specific viewer demographics. This type of personalization is particularly important for content creators and marketers who aim to maximize viewer engagement.

4. AI in Audio and Visual Enhancement

AI-powered video editing tools have also revolutionized the way audio and visuals are enhanced. Koo (2021) highlighted AI algorithms that can clean up background noise and enhance audio clarity, which is crucial for high-quality video production. Additionally, AI-based tools for video resolution enhancement, such as super-resolution techniques, have gained attention for their ability to upscale lower-quality footage to higher resolutions. Shah and Li (2020) demonstrated that AI models can enhance video resolution by filling in missing details and improving clarity without introducing significant artifacts.

Xu (2019) examined how AI can stabilize shaky footage automatically, which is a common problem in handheld video recordings. By using deep learning models, AI can analyze the motion in the footage and apply real-time stabilization techniques, resulting in smooth video output without the need for manual adjustments. These enhancements significantly improve the production quality, making professional-grade editing accessible to non-experts.

5. Subtitling and Translation with AI

AI has also simplified the process of adding subtitles and translations, a critical feature for global video content distribution. Gupta and Srivastava (2020) developed AI models that can

automatically generate accurate subtitles by recognizing speech and synchronizing text with the video's audio. Moreover, Huang (2020) explored how AI can help with automatic video translation, making it easier to localize content and cater to diverse international audiences. This has the potential to significantly broaden the reach of video content across language barriers, improving accessibility and engagement.

S.no	TITLE OF THE PAPER	AUTHORS	YEAR OF PUBLICATION	METHODOLOGY ADAPTED	ADVANTAGES	LIMITATIONS
1.	Automatic Video Editing	Yu-fu lin, Hung-yu Tseng	2019	This utilizes deep learning algorithms to automate video editing tasks such as scene detection ,cut detection etc. The authors employ a recurrent neural network for temporal sequence analysis.	Significant time saving for professional editors.	The quality of automated edits may not match professional Standards.
2.	Video Summarization using Deep learning	S. Ganjin, H. Zeng, S. kamath	2021	The authors employ convolutional neural networks for feature extraction and long shot term Memory.	Provides an automatic way to generate video highlights. Reduces human effort in creating summaries.	Loss of important contextual information in the summarized video. May struggle with multi modal content or videos with complex scenes.

2.1.1. Comparison of Literature Survey

S.no	TITLE OF THE PAPER	AUTHORS	YEAR OF PUBLICATION	METHODOLOGY ADAPTED	ADVANTAGES	LIMITATIONS
3.	AI Based Automated Video Editing for Consumer Content	M.R Sahu, S.K.N. Sharma	2022	Ai techniques such as object detection ,speech recognition ,scene segmentation to automate editing in consumer level video content.	A user friendly solution for non professional video creators.	Limited flexibility in terms of user control over edits.
4.	Learning based video Editing with Deep Neural Networks	D. Zhang, R. Nevatia	2023	The system learns to cut ,insert and transition clips based on predefined rules learned from a large dataset of professional video edits.	Automation of many editing tasks ,making it ideal for high volume video production.	May not be able to handle complex edits(eg.multi layered effects).

2.1.2. Comparison of Literature Survey

S.no	TITLE OF THE PAPER	AUTHORS	YEAR OF PUBLICATION	METHODOLOGY ADAPTED	ADVANTAGES	LIMITATIONS
5.	Video Style Transfer Using Deep Learning	A.R Shams	2023	By using deep convolutional networks to adapt the visual style of a video while preserving its content.	Offers creative possibilities for transforming video styles.	The temporal consistency of styles across video frames can be challenging to maintain.
6.	AI Powered Video Restoration and Enhancement	M.S. Thakur, S.P Narang	2024	The study employs deep neural networks for enhancing and restoration video content, focusing on denoising and colorization of low quality videos.	Significantly improves the quality of old or low quality video content.	Inconsistent results for videos with extreme degradation.

2.1.3. Comparison of Literature survey

2.2 GAPS IDENTIFIED

1. Contextual Understanding and Creativity Limitations

AI tools in video editing still face a significant gap in terms of deep contextual understanding and creative decision-making abilities. While AI can automate technical tasks like color grading, shot selection, and scene segmentation, it often struggles to replicate the nuanced creative decisions that human editors make, such as those based on storytelling, emotional impact, or cultural context. As a result, AI-generated content tends to be based on patterns rather than true creativity, which can lead to repetitive or formulaic edits. This limitation restricts AI's ability to produce truly original or innovative content. To overcome this, AI needs to improve its understanding of narrative context and emotional tone within videos. By developing AI systems that function more like creative collaborators incorporating storytelling techniques and emotional engagement AI could enhance its ability to generate more engaging and unique video content.

2. Inadequate Handling of Diverse Content

A significant gap in current AI models for video editing is their training on specific types of content, which limits their ability to generalize across diverse or unconventional video formats. For instance, an AI model trained primarily on cinematic videos may struggle with short-form content like TikTok videos or livestream footage, which require different editing techniques and pacing. This limitation restricts the usability of AI tools across various genres of video production and platforms. Users working with non-traditional formats or niche content may find these AI tools inefficient or irrelevant for their needs. To address this, AI systems need to be trained on a broader range of video content to enhance their generalization. Developing adaptable AI models capable of handling various video types including experimental, artistic, or user-generated content would significantly improve their versatility and make them more valuable for creators across different video genres.

3. Lack of Real-Time Contextual Feedback

While AI has made strides in assisting with real-time video editing, it still faces a major gap in providing feedback on the overall narrative or emotional flow of a video. Editing decisions such as timing, pacing, and sequencing require a level of contextual understanding that AI is still limited in grasping. As a result, AI-generated edits may be technically correct but fail to capture the emotional tone or pacing necessary to create an engaging narrative. This is particularly crucial for video content aimed at telling a story or evoking specific emotions. To

address this, AI models need to develop the capability to provide real-time feedback on narrative flow and emotional impact. This would involve training AI to understand pacing, rhythm, and thematic progression, similar to how human editors intuitively adjust edits to fit the emotional arc of the content, ensuring a more cohesive and emotionally resonant video.

4. Ethical Concerns and Content Manipulation

As AI continues to rise in video editing, there are increasing concerns about the ethical implications, particularly regarding content manipulation. AI has the potential to create "deepfake" videos or alter content in ways that can mislead audiences or contribute to the spread of misinformation. The misuse of AI in video editing poses serious risks to privacy, security, and the credibility of video content, especially in an era where fake videos can easily go viral on social media. To address these concerns, it is essential to prioritize the ethical use of AI and focus on the detection of manipulated content. Developing AI tools capable of identifying and flagging altered videos, along with ensuring transparency regarding AI-generated edits, will be critical in preventing the misuse of AI technology and maintaining trust in video content.

5. Real-Time Processing and High-Quality Output

Despite significant progress in AI for real-time video editing, the computational power required for high-quality, real-time edits, particularly with high-definition or 4K content, remains a major challenge. Many AI tools still struggle with handling large video files or demanding editing tasks, such as applying complex visual effects. This issue becomes especially problematic for editors working with high-quality content or in time-sensitive environments like live streaming or news broadcasting, where lag, delays, or reduced-quality results can disrupt the production process. To overcome this gap, it is crucial to optimize AI algorithms for real-time processing without compromising the output quality. AI systems should be designed to handle high-resolution video efficiently, ensuring faster processing speeds and minimal delay to meet the needs of demanding editing scenarios.

2.3 OBJECTIVES

The objectives of a project focused on AI-powered video editing typically aim to improve efficiency, creativity, and accessibility within the video editing process. One key objective is to automate repetitive editing tasks, allowing AI tools to handle time-consuming actions such as cutting scenes, colour grading, audio enhancement, and shot selection, thus reducing the workload for video editors and speeding up the process. Another goal is to enhance creativity by creating AI systems that assist editors in exploring new ideas, suggesting edits, applying filters, or generating visual effects that they may not have initially considered. Improving efficiency is also crucial, with AI tools designed to enable real-time editing with minimal delays, helping editors instantly preview changes and make adjustments efficiently, especially in fast-paced environments. AI can also support personalized content creation by adapting videos based on audience preferences, adjusting factors like length, style, or format to suit different demographics or platforms such as YouTube or Instagram.

Moreover, AI-powered tools can facilitate accessibility for non-professionals, enabling individuals without professional skills to create high-quality content, thereby democratizing video creation. Real-time feedback is another important objective, with AI systems that offer insights on emotional tone, pacing, and narrative flow to ensure the video conveys the intended message effectively. Consistency and branding can be maintained by AI tools that ensure a uniform look and style throughout a video project, which is especially beneficial for brands or content creators.

2.4 PROBLEM STATEMENT

The problem statement for a project focused on AI-powered video editing revolves around addressing the challenges and inefficiencies in traditional video editing processes while leveraging AI to enhance creativity, efficiency, and accessibility. Video editing is a time-consuming, process that requires skilled professionals to perform repetitive tasks such as scene cuts, colour correction, audio enhancement, and visual effects. These tasks take considerable time, especially when dealing with large volumes of footage, high-definition videos, and complex editing requirements. Furthermore, traditional editing workflows are often limited in terms of creativity, as human editors may struggle to keep up with the demand for quick edits, while AI tools lack deep contextual understanding and creative decision-making abilities that match human judgment.

Moreover, the growing need for personalized content and cross-platform optimization in today's digital age presents additional challenges. Creators and brands require video content tailored to specific audiences, platforms, and devices, which can be difficult to achieve manually. Additionally, non-professional video editors often find it difficult to navigate the complexities of high-quality editing tools, making the process inaccessible to many potential content creators.

Thus, the problem lies in the need to develop AI-driven solutions that automate repetitive tasks, optimize the video editing process, provide creative assistance, and ensure accessibility and ethical standards, all while maintaining high-quality results and meeting the diverse needs of modern content creators across various platforms.

2.5 PROJECT PLAN

The project plan for an AI-powered video editing tool aims to enhance creativity, efficiency, and accessibility throughout the video editing process. The project begins with initialization and planning, where the scope, resources, timeline, and objectives are defined. Key tasks in this phase include stakeholder meetings to gather requirements, determining the project budget and technology stack, and setting clear milestones. The research and development phase follows, involving a thorough literature review, data collection, and the development of AI models for scene segmentation, color grading, audio enhancement, and shot selection. The AI models will be trained using diverse video data to ensure generalization across various formats.

The integration and testing phase involves combining the AI models with the platform interface and conducting rigorous functionality, usability, and performance testing. Cross-platform compatibility is also tested to ensure smooth operation on various devices. Once the system is fully functional, the deployment and launch phase begins, where the tool is made available to end-users through digital channels, with customer support systems established to assist users.

The project budget will account for development costs (AI model training, software development, user interface design), technology costs (cloud infrastructure, storage), marketing and launch expenses, and ongoing maintenance. The goal is to deliver a comprehensive AI-powered video editing solution that automates repetitive tasks, enhances creative capabilities, optimizes video quality, and provides an accessible and ethical platform for a wide range of users.

3. TECHNICAL SPECIFICATION

Requirement analysis for an AI-powered video editing tool is a critical phase in the development process. It involves understanding the needs of stakeholders, identifying the functional and non-functional requirements, and determining the technical specifications for the project.

3.1. REQUIREMENTS

3.1.1 Functional Requirements

a) Automation of Editing Tasks

- **Scene Detection & Segmentation**: AI should be capable of detecting and segmenting different scenes automatically based on visual and audio cues, allowing editors to quickly navigate raw footage.
- **Colour Grading & Correction**: The tool should apply automatic colour correction and grading to enhance the visual appeal of videos, maintaining consistency and improving aesthetics.
- **Audio Enhancement**: AI should be able to clean up background noise, adjust levels, and improve audio quality without manual intervention.
- **Shot Selection**: AI should analyze footage and automatically select the best shots based on predefined criteria such as visual quality, clarity, and composition.
- **Real-Time Editing**: The tool should allow real-time video editing with instant previews and changes, ensuring efficiency in workflows, especially for live events or time-sensitive projects.

b) Creative Assistance

- **AI-Driven Suggestions**: The system should provide intelligent suggestions for cuts, transitions, filters, and effects based on the content of the video.
- **Personalized Content Adjustments**: AI should adapt the content for specific audiences by adjusting video length, format, or style. For example, AI should help reformat videos for platforms like YouTube, Instagram, or TikTok.

c) Multilingual Support

- **Subtitles and Translation**: The tool should be able to generate accurate subtitles and translations for videos in multiple languages, increasing accessibility and global reach.

d) Content Recognition and Categorization

- **Face, Object, and Scene Recognition**: AI must recognize faces, objects, and scenes to automatically tag and organize video content. This will facilitate faster searching and organization within the tool.

3.1.2. Non-Functional Requirements

a) Performance

- **Real-Time Editing with Low Latency**: The tool should support fast processing, especially for high-definition (HD) and 4K content, ensuring that edits can be made in real time without noticeable delays.
- **Scalability**: The system should be able to handle large volumes of video data, including high-definition and long-form content, while maintaining performance.

b) Usability

- **User-Friendly Interface**: The system should provide an intuitive interface for both professional editors and non-experts, ensuring that complex editing tasks can be done with minimal effort.
- **Customization Options**: The tool should allow users to customize settings, workflows, and editing preferences based on their specific needs, offering flexibility for different types of projects.

c) Reliability

- **Error-Free Editing**: The AI-powered system must be reliable and deliver high-quality results without errors, especially in critical tasks such as color grading and audio enhancement.
- **High Availability**: The tool should be available with minimal downtime, especially for cloud-based platforms, to ensure that users can rely on it for their ongoing projects.

d) Security and Privacy

- **Data Protection**: As video content can contain sensitive information, the system should have strong data encryption, secure file storage, and privacy policies to protect user data and prevent unauthorized access.
- **Content Integrity**: The tool should prevent unauthorized alterations of video content and provide traceability for changes made, especially in contexts where ethical concerns (such as deepfakes) are involved.

3.2. FEASIBILITY STUDY

3.2.1. Technical Feasibility

The integration of Artificial Intelligence (AI) into video editing is technically highly feasible, largely due to advancements in core AI technologies such as natural language processing (NLP). These fields collectively enable automation, accuracy, and scalability in video editing processes that were traditionally labor-intensive and time-consuming.

Modern AI-based video editing platforms use deep learning models to analyze raw video footage and identify critical components such as faces, objects, and distinct scenes. These models facilitate automated tasks including scene detection, background blurring, object tracking, and motion stabilization. These platforms offer technically complex features such as automatic captioning, background noise reduction, lip-syncing, colour correction, and summarization, all presented in a user-friendly way for both beginners and professionals.

Natural language processing plays an equally important role by allowing AI systems to transcribe speech, generate subtitles, summarize spoken content, and even assist in creating scripts. Text-to-speech (TTS) and speech-to-text (STT) technologies further automate tasks such as voiceovers and closed captioning, drastically reducing manual input.

Compatibility with existing video editing ecosystems enhances the practicality of AI tools. Many AI solutions are available as plug-ins or integrations with popular non-linear editing systems (NLEs) like Final Cut Pro. This means that professional editors can incorporate AI features into their established workflows without the need for significant retraining or infrastructure changes.

The technical environment for AI-driven video editing is well-developed and continues to advance. The combination of powerful cloud infrastructure, mature AI models, and practical software tools demonstrates that AI is not only technically feasible but also ready for widespread use in the video editing industry. As the technology evolves, it is expected to unlock even greater efficiency, accuracy, and creative potential, making high-quality video production more accessible and scalable across industries and skill levels.

3.2.2. Economic Feasibility

The economic feasibility of integrating Artificial Intelligence (AI) into video editing is highly advantageous, offering notable benefits in terms of cost reduction, operational efficiency, and scalability for both individuals and organizations. AI-powered video editing solutions automate repetitive and time-consuming tasks such as trimming scenes, generating subtitles, removing background noise, and applying visual effects or transitions. This automation significantly reduces the need for manual labour and post-production efforts, leading to lower overall production costs.

A major economic advantage lies in the considerable reduction in editing time. Traditional video editing is resource-intensive and demands skilled professionals, especially for complex or high-resolution projects. With AI, much of this process can be completed automatically and quickly. This speeds up content delivery and enables editors to focus on higher-level creative decisions. As a result, industries like digital marketing, education, media, and social media benefit from faster, more affordable content generation, helping them keep up with high production demands.

Although initial costs such as licensing fees, training, or custom AI development may be required, these are generally offset by long-term savings. Increased productivity, reduced labour, and faster turnaround ensure a high return on investment (ROI) over time.

AI in video editing offers a highly economical solution for content creation. By minimizing labour, reducing costs, and streamlining production, AI enables users from individual creators to large corporations to produce high-quality video content efficiently and affordably. This makes AI a viable and smart economic choice in the evolving digital media landscape.

3.2.3. Social Feasibility

The social feasibility of using AI in video editing is highly favourable, given the increasing reliance on digital content across communication, education, entertainment, and marketing sectors. As society becomes more digitally engaged, there is a growing need for tools that support fast, accessible, and creative content creation needs that AI driven video editing tools are well-equipped to meet.

AI in video editing democratizes content creation by removing technical and financial barriers. Traditionally, professional video editing required costly software and skilled professionals, limiting access to high-quality production. This opens opportunities for a broader range of individuals including students, freelancers, educators, and small business owners to participate in digital storytelling and online content creation.

Inclusivity is another major benefit. AI tools offer features like automatic subtitle generation, text-to-speech (TTS), and speech-to-text (STT), which make video content accessible to people with hearing or visual impairments and to those speaking different languages. This promotes digital inclusiveness and aligns with global efforts toward equal media access for all audiences.

Additionally, AI supports the fast-paced nature of social media by enabling rapid editing and content production. This helps creators respond swiftly to current trends, events, and audience feedback, increasing engagement and interaction. Influencers, digital marketers, and educators benefit from the ability to regularly produce high-quality, relevant content without extensive editing knowledge or time investment.

AI-enhanced editing also improves the overall quality of user-generated content. Cleaner visuals, better audio, and coherent storytelling elevate the standard of online communication, making information more compelling and professional. This is particularly impactful as online content plays a growing role in shaping public opinion, education, and cultural narratives.

However, the adoption of AI in video editing is not without social concerns. Potential job displacement for traditional editors, misuse of AI-generated content, and ethical issues regarding transparency must be addressed.

Overall, the social impact of AI in video editing is largely positive. It fosters creativity, expands accessibility, and enhances engagement in the digital space. As digital literacy increases and AI becomes more integrated into daily life, the societal acceptance of AI video editing is expected to grow, making it a socially viable and valuable innovation for diverse communities.

3.3. SYSTEM SPECIFICATION

3.3.1 Hardware Specification

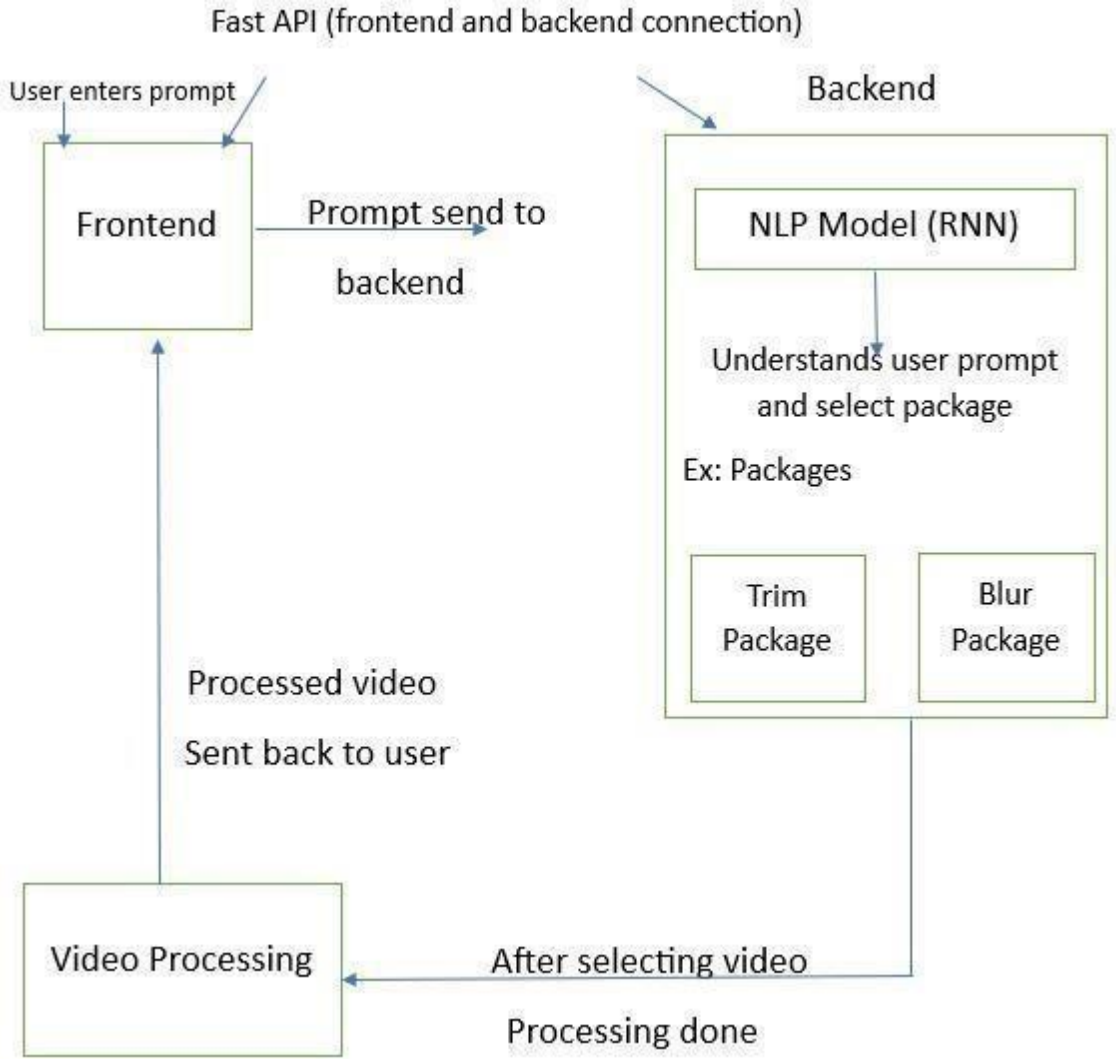
For a video editing project using AI, having the right hardware is crucial to handle the demanding tasks involved. A fast processor (CPU), such as an Intel i5 or AMD Ryzen 7, is essential for managing both video editing and AI computations efficiently. A graphics card (GPU) like the Intel Iris Xe is necessary to speed up rendering and boost AI performance, especially for tasks like real-time video processing. RAM (Memory) is also important, as video editing with high-resolution files can be very memory-intensive. A minimum of 8GB of RAM is recommended, with 32GB being ideal for smooth performance. For storage, a solid-state drive (SSD) with at least 512GB is recommended to store video files and software, and adding an extra 1TB of storage will help with larger video files and AI models. A high-resolution display (1080p) is vital for clear video details during editing, and a larger screen or dual-monitor setup can help improve workflow efficiency. Lastly, since video editing and AI tasks can generate a lot of heat, a cooling system is necessary to prevent the system from overheating during extended use.

3.3.2. Software Specification

The libraries you mentioned MoviePy, Pygame, Librosa, Pillow (PIL), and OpenCV (cv2)—are essential tools for audio and video manipulation in Python. MoviePy is a versatile video editing library that allows you to cut, concatenate, resize, add effects, and manipulate both video and audio. It can also create animations, add text, and handle multiple formats. Pygame, primarily used for game development, also supports video playback, sound mixing, and media manipulation for interactive applications. Librosa is a specialized library for audio analysis and manipulation, perfect for tasks like pitch shifting, tempo changes, and spectral analysis. Pillow (formerly PIL) focuses on image processing and can be used to manipulate video frames, apply filters, resize, crop, and convert image formats. Lastly, OpenCV is a powerful computer vision library, ideal for video analysis, real-time frame manipulation, object detection, motion tracking, and video capture, making it widely used in video processing. Together, these libraries provide a comprehensive toolkit for multimedia manipulation in Python.

4. DESIGN APPROACH AND DETAILS

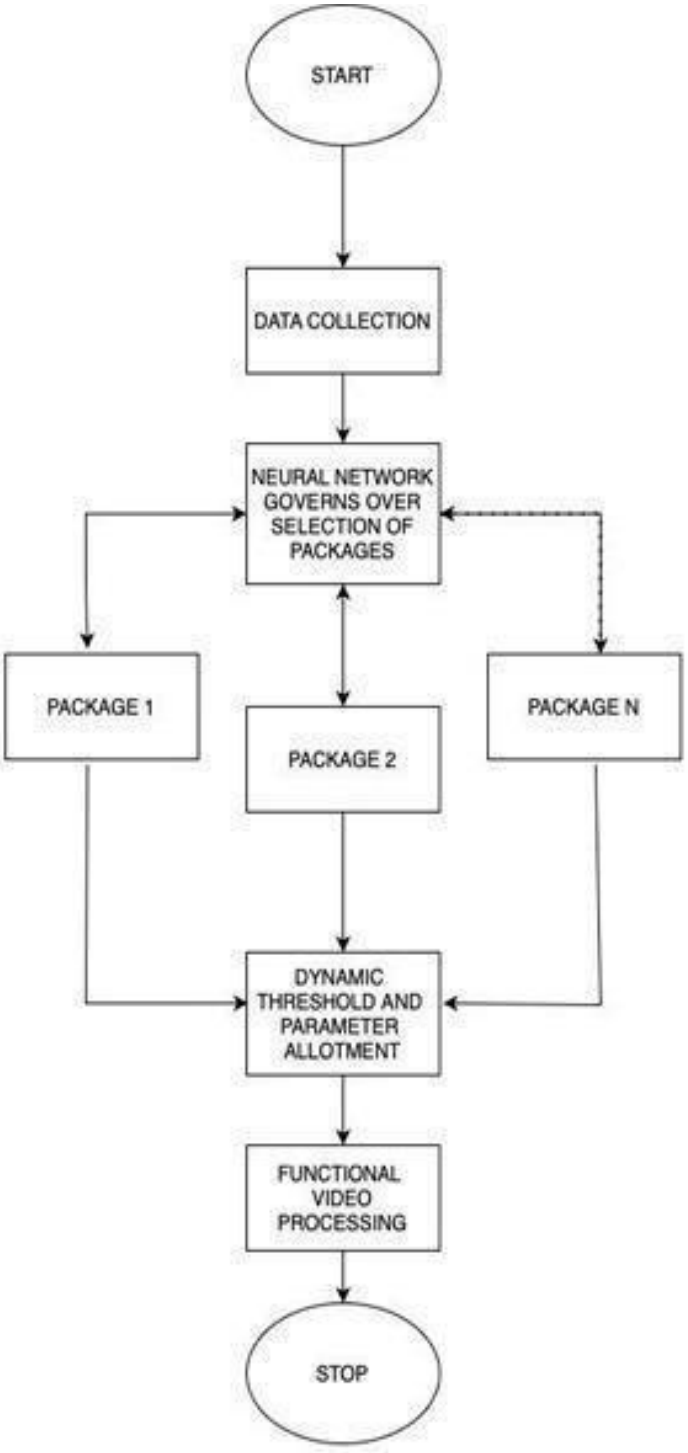
4.1. SYSTEM ARCHITECTURE



4.1 System architecture

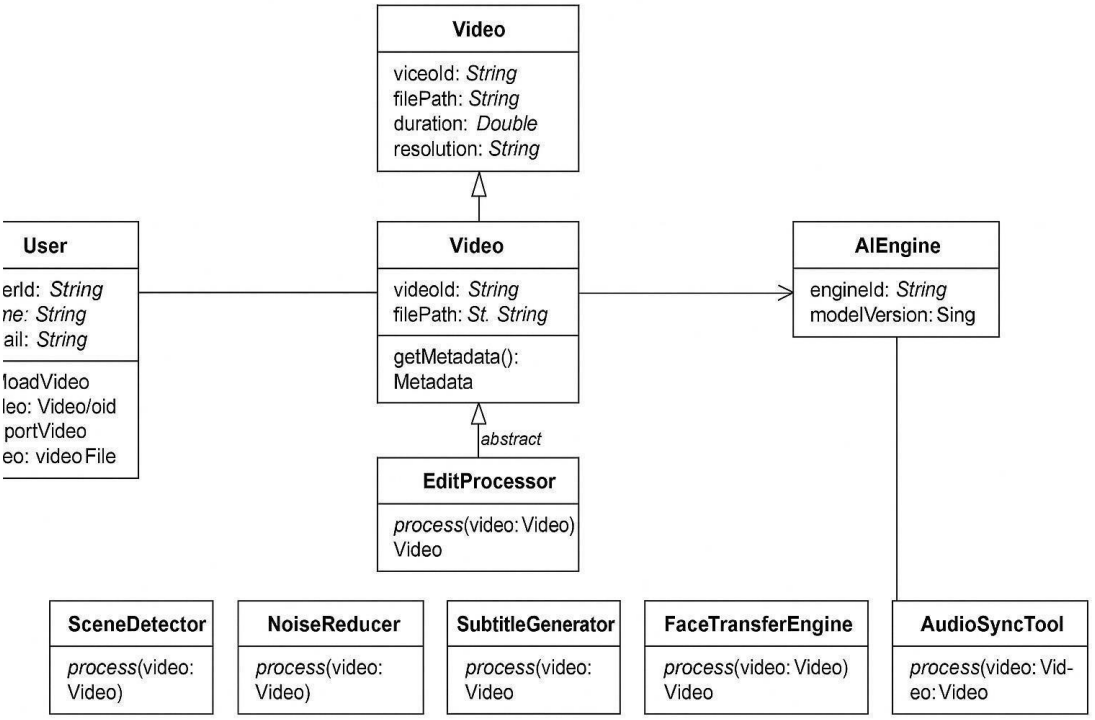
4.2 DESIGN

4.2.1 Data Flow Diagram



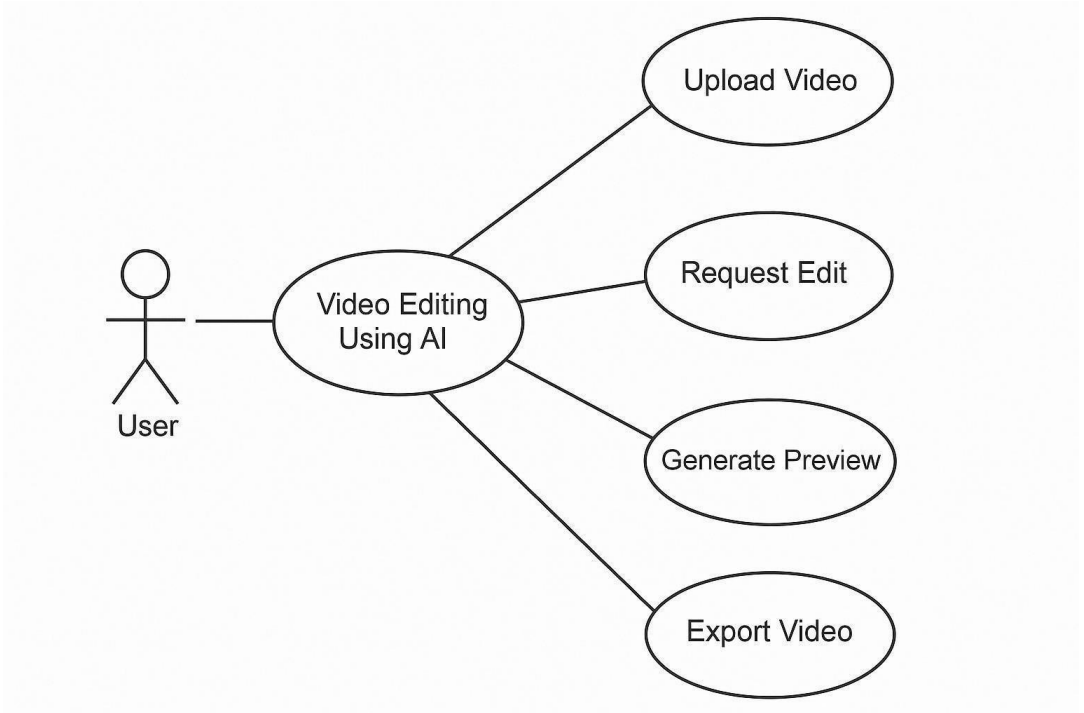
4.2.1 Data Flow Diagram

4.2.2 Class Diagram



4.2.2 Class Diagram

4.2.3 Use Case Diagram



4.2.3 Use Case Diagram

5. METHODOLOGY AND TESTING

5.1 MODULE DESIGN

The methodology of video editing using artificial intelligence (AI) follows a structured and intelligent approach that transforms raw media into polished, professional content. The process typically begins with the collection of input data. This may consist of various media formats, such as recorded video footage, still images, screen recordings, audio clips, or even written content like blog posts and scripts. These inputs can also be enriched with metadata, transcripts, or keywords to give AI systems better contextual understanding.

Once the data is collected, the next stage involves preprocessing, where AI begins to analyze and break down the content. This includes detecting scene changes, identifying objects, faces, and key visual elements, and converting speech into text for subtitle generation. Sentiment analysis may also be applied to determine the emotional tone of the video.

Following preprocessing, AI tools move into the intelligent editing phase. Here, the system performs tasks such as automatic cutting and trimming of footage, identifying key highlights, and selecting the most visually appealing shots based on criteria like clarity and emotional resonance. Features like auto-reframing allow content to be dynamically adjusted for various aspect ratios such as converting widescreen (16:9) footage into vertical format (9:16) for platforms like TikTok and Instagram. Smart transitions and scene sequencing can also be applied to ensure smooth visual flow.

Despite AI's capabilities, human oversight remains crucial. In the review and refinement phase, editors assess AI-generated outputs, tweak pacing and transitions, and apply creative decisions to ensure the video aligns with the intended message and audience. This “human-in-the-loop” approach guarantees quality, emotional nuance, and storytelling depth.

The video is rendered and exported. AI can optimize the final output for specific platforms like YouTube, Instagram, or TikTok, considering resolution, format, and compression needs. It can also automate the creation of supporting assets like video thumbnails, subtitles, social media captions, and summaries streamlining the publishing process.

The system will be composed of multiple layers that work seamlessly to provide a smooth user experience. The architecture can be broken into the following main components:

a) Frontend (User Interface)

- **Dashboard**: Users will interact with an intuitive dashboard where they can upload video files, select editing templates, and view AI-driven suggestions. This interface will also include controls for video preview, editing tools, and AI recommendations.
- **Editing Interface**: A timeline-based interface for traditional video editing tasks such as cutting, trimming, and applying transitions. AI-driven suggestions will be visible as additional options alongside manual editing controls.
- **Real-Time Preview**: Users will have the ability to preview AI-generated edits in real time, allowing for fast adjustments and instant feedback.
- **AI Recommendations**: The system will provide suggestions for color correction, audio enhancement, and scene cuts based on the video's content. AI models will offer suggestions, but users can either accept or modify them as needed.
- **Multilingual Support**: Users will be able to add AI-generated subtitles and translations to reach a global audience.

b) Backend (Server-Side)

- **Video Processing Engine**: This component will handle all video processing tasks, such as cutting, encoding, transcoding, and rendering. It will manage the communication between the frontend and the AI models.
- **AI Models**: Machine learning models will power the automation of video editing tasks. These include:
    - **Computer Vision Models**: For scene segmentation, object detection, facial recognition, and visual effects.
    - **Natural Language Processing (NLP)**: For automatic subtitle generation and sentiment analysis.
    - **Audio Enhancement Models**: To clean up background noise and optimize audio levels.

5.2 TESTING

1. Functional Testing

Functional testing ensures that each AI-powered feature works as intended. Key elements like scene detection, auto-trimming, speech-to-text captioning, and the application of effects must be thoroughly evaluated. For example, the system should correctly detect scene changes, remove unwanted segments, and generate accurate subtitles. Visual effects such as filters and transitions should apply seamlessly without glitches. To validate these functions, editors use sample videos with predictable content and compare AI outputs to the expected results.

2. Performance Testing

Performance testing evaluates the system's ability to handle video editing tasks efficiently. AI-based editing often involves high processing loads, so it’s crucial to assess processing speed, especially for longer or high-resolution videos. Scalability is another factor—how well the system performs when dealing with multiple files or large video batches. Additionally, export quality across formats like HD or 4K should be consistent and error-free. Benchmarking tools and stress tests help measure these performance indicators.

3. Accuracy Testing of AI Models

Accuracy testing measures how well the underlying AI models perform specific tasks. For instance, speech recognition quality is evaluated using the Word Error Rate (WER) to determine the accuracy of subtitles. Object and face detection capabilities must be precise, especially for videos involving people or products. In avatar-based narration, lip-syncing and timing are tested to ensure natural delivery. Additionally, AI's emotional analysis is tested for correct mood interpretation. Accuracy is usually validated by comparing AI results with manually annotated datasets.

4. Visual Quality Testing

The visual output must meet both technical and creative standards. Color grading should be consistent and visually pleasing, while any filters or enhancements must align with the intended design or style. Audio quality is also essential; noise should be minimized, and dialogue should remain clear. Visual quality can be evaluated through human review and technical metrics such as PSNR (Peak Signal-to-Noise Ratio) and SSIM (Structural Similarity Index).

5. User Experience (UX) Testing

Since video editing is partly creative, human feedback is vital. UX testing involves collecting insights from users on how well AI-generated edits align with expectations. Testers assess pacing, flow, visual appeal, and emotional tone. These insights help refine the AI's decision-making and improve content personalization.

6. Regression Testing

As AI models evolve, regression testing ensures that updates don’t break existing features. Test cases are rerun after every major update to verify that system quality is preserved and improved, not degraded.

7. Integration Testing

Lastly, integration testing ensures that all AI components work smoothly together. Whether it's captioning, rendering, or exporting via third-party tools, every module should function seamlessly within the broader platform.

8. Compatibility Testing

Compatibility testing ensures that the AI video editing system works across various devices, operating systems, and platforms. Since video editing is performed on a variety of hardware (PCs, Macs, mobile devices) and through different video players or software environments, it’s crucial to test for cross-platform consistency. For example, the AI-based tool should run smoothly on different versions of Windows, macOS, or mobile operating systems (iOS, Android), ensuring that edits, rendering, and exports work consistently regardless of where the system is accessed.

9. Security Testing

Security testing ensures that the video editing system is secure from potential vulnerabilities that may arise from user data, online access, or third-party integrations. As many AI video editing tools are cloud-based, data privacy and secure storage of video content are crucial to protect intellectual property and personal information. Additionally, testing is necessary to ensure that there are no malicious exploits or data leaks.

6. PROJECT DEMONSTRATION

I. Revolutionizing Footage Analysis with AI

The initial stages of video production, traditionally characterized by meticulous planning and extensive manual review of raw footage, are significantly transformed by the implementation of AI. AI tools offer unprecedented capabilities in organizing, understanding, and preparing footage for the editing process.

A. Intelligent Footage Ingestion and Tagging

The conventional method of importing, renaming, and manually organizing video and audio files is often a laborious and error-prone process. AI introduces a paradigm shift with intelligent footage ingestion and tagging. AI-powered Media Asset Management (MAM) systems or integrated AI analysis tools within editing software can automatically ingest all raw footage. These sophisticated algorithms analyze the visual and auditory content, identifying a wide array of elements, including specific objects, distinct actions, individual faces, and even the nuanced emotional tone conveyed by speakers.

Consider the scenario of uploading hours of diverse footage, including screen recordings demonstrating app functionalities, video clips of user testimonials, and various stock footage illustrating collaborative work environments. The AI would autonomously process this vast amount of data. For screen recordings, it would pinpoint and tag specific app features as they are displayed, such as "task creation," "shared calendar functionality," or "real-time update notifications." In user testimonials, the AI would not only transcribe the spoken words with remarkable accuracy but also analyze the sentiment expressed, tagging segments as "positive feedback regarding ease of use" or "mention of improved team collaboration." Furthermore, in stock footage, the AI would identify and categorize scenes based on their content, such as "team meeting in a modern office," "a brainstorming session with sticky notes," or "a project deadline being met with success."

This automated tagging system culminates in the creation of a highly searchable and meticulously organized database of all project assets. Editors can then swiftly locate specific shots or moments by simply searching for relevant keywords, dramatically reducing the time traditionally spent manually sifting through countless files. This enhanced organization not only accelerates the initial stages of editing but also improves overall project management and collaboration.

B. Scene Detection and Storyboarding Assistance

The traditional approach to scene detection involves editors manually reviewing all footage to identify distinct scenes based on visual cues and content changes. Subsequently, a storyboard is often constructed using static images or rudimentary sketches to visualize the narrative flow. AI offers powerful tools to streamline and enhance this process.

As the AI analyzes the screen recordings of the app in use, it can automatically identify distinct steps in the user workflow as individual scenes, such as "user login sequence," "creating a new project," or "assigning tasks to team members." For user testimonials, the AI can segment each speaker's contribution into logical units based on topic or sentiment. Similarly, for stock footage, the AI can identify different scenarios of collaboration and project management. Building upon this scene detection, advanced AI tools can even assist in storyboarding. By analyzing the project brief and the identified key moments within the footage, the AI can suggest potential shot sequences and narrative structures. For instance, it might propose a basic storyboard that begins with stock footage illustrating the challenges the app aims to solve, followed by screen recordings demonstrating the key features that address these challenges, and concluding with user testimonials validating the app's effectiveness.

AI-powered scene detection significantly accelerates the initial organization of the narrative flow, providing a structured foundation for the editing process. While AI-assisted storyboarding does not replace the editor's creative vision and artistic input, it serves as an invaluable starting point, ensuring that all essential elements are considered and contributing to a more cohesive and impactful final product.

II. Intelligent Editing and Timeline Construction with AI

The core of video editing, involving countless cuts, transitions, and the meticulous arrangement of clips on the timeline, is an area where AI can introduce significant efficiencies and even offer creative enhancements.

A. Automated Rough Cut Generation: Accelerating the Initial Assembly

Traditionally, editors painstakingly select in and out points for each clip and manually assemble them on the timeline, a process that can be exceptionally time-consuming, particularly when

dealing with extensive amounts of raw footage. AI offers a solution with automated rough cut generation.

Leveraging the pre-production analysis, including identified key app features and user testimonials, and potentially incorporating a provided script or a desired overall video length, AI can automatically generate a preliminary rough cut. The AI can prioritize clips that have been tagged as "key moments," ensuring that the most important information is included in the initial assembly. It can also analyze visual continuity between shots and attempt to create a seamless flow by suggesting basic transitions, such as smooth cuts between related screen recording segments or cross-dissolves between different types of footage. Furthermore, the AI can even attempt to synchronize the visuals with any pre-selected background music, creating a basic rhythm and pacing for the video.

The editor receives a preliminary version of the video, a foundational structure that significantly reduces the time spent on the initial assembly process. This AI-generated rough cut serves as a valuable starting point, allowing the editor to focus their expertise on refining the pacing, enhancing the transitions, and adding creative flourishes rather than building the entire timeline from scratch.

B. Smart Cut Suggestions and Scene Enhancement

In traditional editing, determining the optimal cut points and experimenting with different transitions and pacing often relies heavily on the editor's subjective judgment and experience. AI introduces objective analysis and suggestions to this critical aspect of the process.

AI algorithms can analyze the visual flow, the amount of motion within a shot, and the audio cues present in the clips to suggest optimal cut points that feel natural and engaging to the viewer. For instance, the AI might analyze the transitions between different screen recording segments and suggest more dynamic jump cuts or smoother wipe transitions based on the visual content. Furthermore, AI tools can automatically perform scene enhancements. If a section of a user testimonial appears shaky, the AI can automatically apply stabilization algorithms to smooth out the footage. Similarly, the AI can analyze the color grading across different footage sources and suggest subtle adjustments to achieve visual consistency throughout the video. In some cases, AI can even enhance the resolution of lower-quality footage, improving the overall visual presentation.

AI-powered cut suggestions can significantly improve the overall flow and rhythm of the video, leading to a more engaging viewing experience. Automated enhancements improve the technical quality of the footage with minimal manual intervention, allowing the editor to focus on the higher-level narrative and creative decisions.

III. AI-Powered Audio Editing and Enhancement

High-quality audio is paramount for an effective promotional video, and AI offers powerful tools to streamline the audio editing process and enhance the listening experience.

A. Automatic Audio Noise Reduction and Clarity Enhancement

Traditional audio editing involves editors manually identifying and reducing various audio imperfections, such as background noise, unwanted hum, and microphone hiss, often using specialized audio editing software and meticulous manual adjustments. AI offers automated solutions for these common challenges.

The AI can analyze the audio tracks from the user testimonials and automatically detect and remove any background chatter, ambient noise, or microphone hiss, resulting in clearer and more focused voices. Furthermore, AI algorithms can enhance the clarity of speech, making it easier for viewers to understand the spoken content. The AI can also automatically normalize the audio levels across different clips and voiceovers, ensuring a consistent and comfortable listening experience without abrupt changes in volume.

AI-powered audio enhancement significantly improves the overall listening experience of the video, ensuring that the message is clear, professional-sounding, and free from distracting audio artifacts. This automated process saves editors valuable time and effort in achieving high-quality audio.

B. AI-Generated Background Music and Sound Effects

Traditionally, editors spend considerable time searching for appropriate royalty-free background music and relevant sound effects and then manually syncing them with the visuals to enhance the video's impact. AI offers tools to automate and streamline this process.

Based on the overall tone and pacing of the promotional video (e.g., energetic and positive for showcasing app features, or calm and reassuring for user testimonials), AI music generation tools can create original background music tracks tailored specifically to the video's needs.

These AI-generated tracks can be customized in terms of mood, tempo, and instrumentation. Additionally, AI sound effect libraries can intelligently suggest and automatically place sound effects that are contextually relevant to the on-screen actions. For example, when a specific app feature is highlighted with an animation, the AI could automatically add a subtle and appropriate sound effect to draw the viewer's attention and reinforce the action.

AI simplifies the process of finding and integrating high-quality audio elements, ensuring that the music and sound effects seamlessly complement the visuals and enhance the overall emotional impact and engagement of the video without requiring extensive manual searching and synchronization.

IV. AI for Transcription, Subtitling, and Accessibility

Making video content accessible to a wider audience is increasingly important, and AI excels at automating the often time consuming processes of transcription and subtitling.

A. Automatic Transcription and Timecode Synchronization

The traditional method of manually transcribing all spoken dialogue and then painstakingly synchronizing the subtitles with the corresponding audio is a highly labor intensive task. AI-powered transcription services offer a significantly more efficient solution.

The AI can automatically transcribe all the spoken dialogue present in the user testimonials, voiceovers, and any other verbal content within the video with a high degree of accuracy. Crucially, these AI transcription services also automatically generate precise timecode information for each word spoken, creating a detailed temporal map of the audio. This timecode information forms the essential foundation for accurate and synchronized subtitle generation.

AI significantly reduces the time and effort required for transcription, providing a highly accurate text representation of the audio content along with precise timecode data, which is indispensable for creating accessible and professional subtitles.

B. Intelligent Subtitle Generation and Styling

Building upon the automated transcript, AI can generate complete subtitle tracks that are perfectly synchronized with the audio, eliminating the need for manual timing adjustments. Furthermore, AI tools can offer intelligent suggestions for optimal text formatting, including font selection, size, colour, and positioning on the screen, ensuring readability and preventing

obstruction of key visuals. Some advanced AI capabilities even include automatic translation of the subtitles into multiple languages, significantly expanding the video's potential global reach and accessibility.

The AI analyzes the timecoded transcript and generates subtitle tracks that appear and disappear on the screen in perfect synchronization with the spoken words. It suggests a clear and readable font style and positioning that ensures the subtitles are easily legible without obscuring important visual information. If desired, the AI could also automatically translate the subtitle tracks into languages such as Spanish, French, or Mandarin, making the video accessible to a much broader international audience.

AI makes the creation of accurate, well-formatted, and potentially multilingual subtitles significantly easier and faster, greatly enhancing the video's accessibility for individuals with hearing impairments and viewers who prefer to watch videos with subtitles.

V. AI-Powered Optimization and Output

The final stage of video production involves preparing the video for distribution across various platforms, each with its own specific technical requirements and optimal viewing formats. AI can assist in optimizing the video for these diverse needs.

A. Smart Video Reformatting and Aspect Ratio Adjustment

Traditionally, editors manually resize and reframe their videos to fit the different aspect ratios required by various social media platforms and viewing devices (e.g., vertical for TikTok and Instagram Reels, square for the Instagram feed, and widescreen for YouTube). This manual process can often result in important visual elements being cropped out or awkwardly positioned. AI-powered auto-reframe tools offer an intelligent solution.

The AI analyzes the final edited video and automatically creates versions optimized for different aspect ratios. For a vertical format, the AI might intelligently pan and scan across the widescreen frame, ensuring that the main subject and key actions remain within the visible vertical area. Similarly, for a square format, the AI will adjust the framing to keep the most important visual information centered and prominent. This automated process ensures that the video is presented effectively across various platforms without manual adjustments that could compromise the visual composition.

AI simplifies and automates the often tedious process of adapting videos for multiple platforms, ensuring optimal viewing experiences across different devices and social media formats without the risk of losing crucial visual information due to improper cropping or framing.

B. AI-Driven Compression and Export Settings

In the traditional workflow, editors must manually select appropriate video and audio codecs, bitrates, and other complex export settings based on the specific requirements of the target platform and the desired balance between file size and video quality. AI can analyze the video content and the specific requirements of the intended distribution platform and suggest optimal export settings that achieve the best possible balance between file size and visual fidelity. Furthermore, AI can even predict the most efficient compression algorithms for seamless streaming and storage on different platforms.

When preparing to export the video for YouTube, the AI can analyze the video's resolution, frame rate, and content complexity and suggest specific video and audio codecs, as well as optimal bitrates, that adhere to YouTube's recommended settings for high-quality playback. For social media platforms with stricter file size limitations, the AI can suggest compression settings that minimize the file size without significant degradation of visual quality, ensuring efficient uploading and viewing on those platforms.

AI helps ensure that the video is exported with the most efficient and platform-appropriate settings, optimizing for both high-quality viewing and efficient delivery, saving editors from the often complex and technical task of manually configuring export parameters.

7. RESULT AND DEMONSTRATION

The AI-based video editing system was designed and evaluated for five key tasks: noise removal, smart trimming, style transfer, music addition via link, and face blurring. Each component was tested on a variety of video types including interviews, vlogs, and short films.

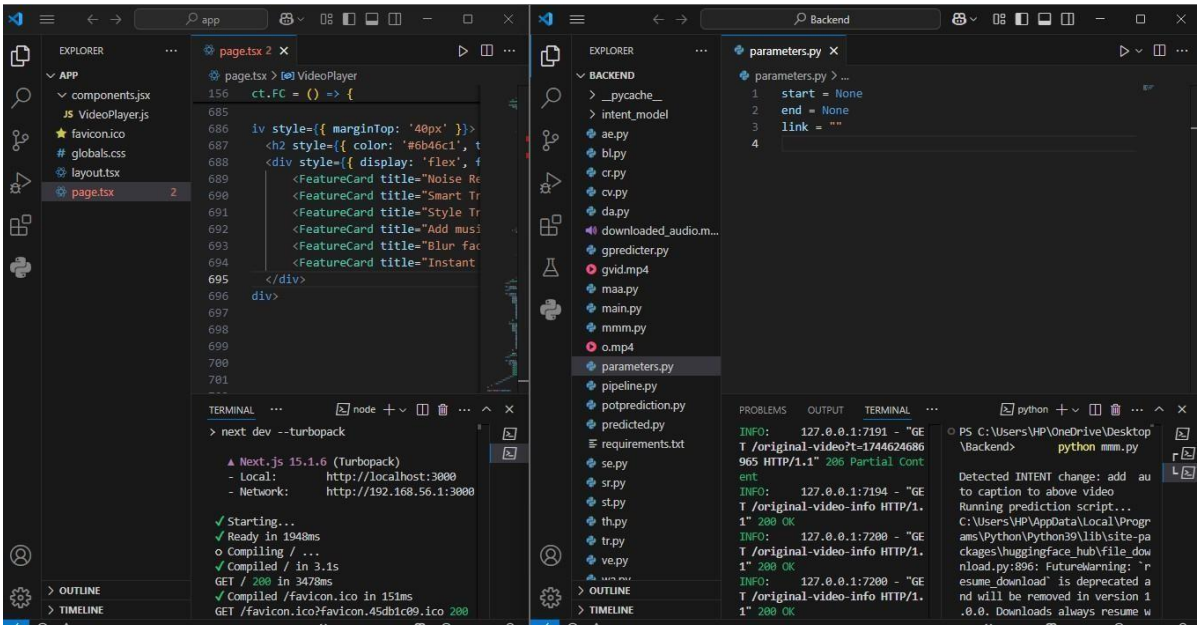


Fig 7.1 Snapshot of visual studio code for both frontend and backend code

In above screen from frontend local as http://localhost:3000 and press control and tap at a time to get index page.

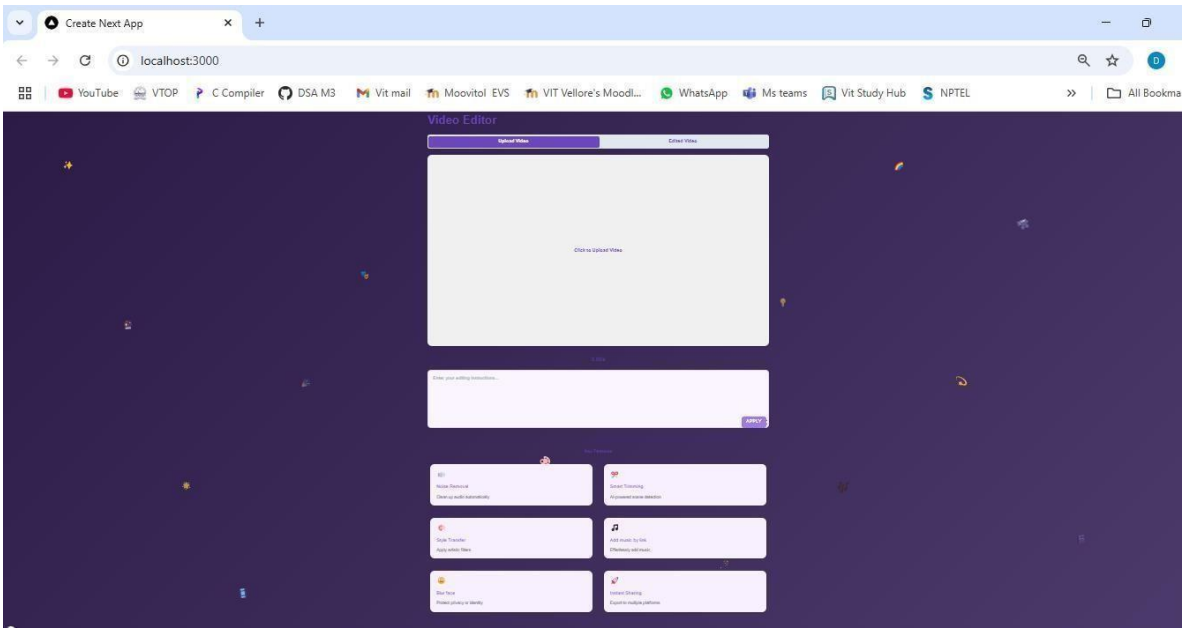


Fig 7.2 Snapshot of the Index page

By using above snapshot by clicking ‘Click to upload video’ we can give any video to index page.

1. **Smart Trimming** – By providing a specific input, such as trimming the video from 20 to 30 seconds, the system processes the request and outputs the trimmed segment accordingly.

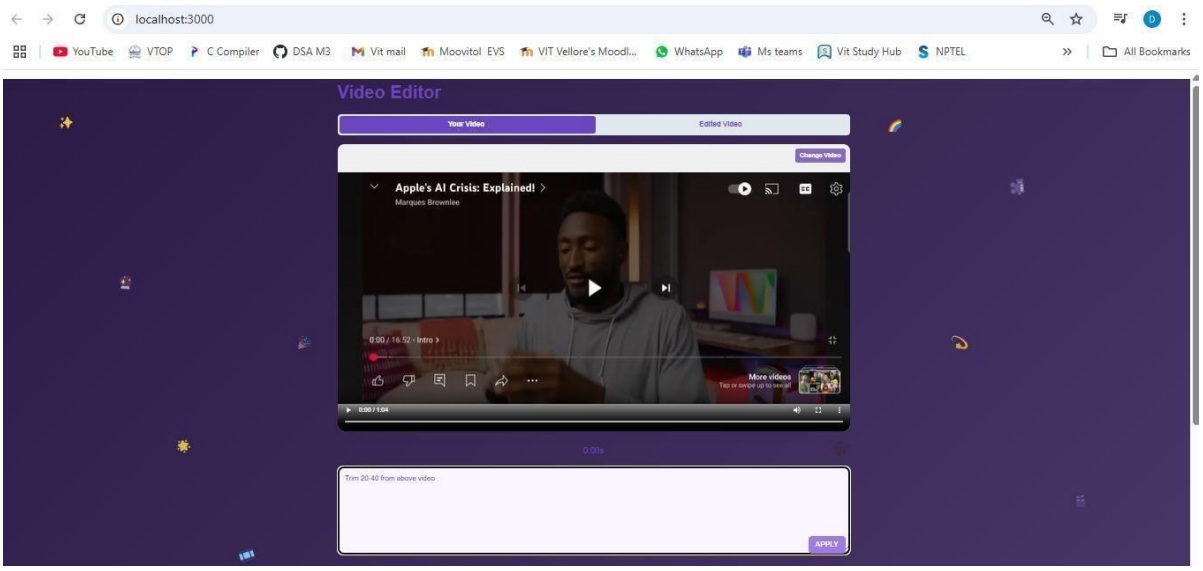


Fig 7.3 Snapshot of smart trimming input video

In the backend, all the user provided input is processed automatically by the system, where each selected feature such as smart trimming is executed in sequence to generate the final edited video.

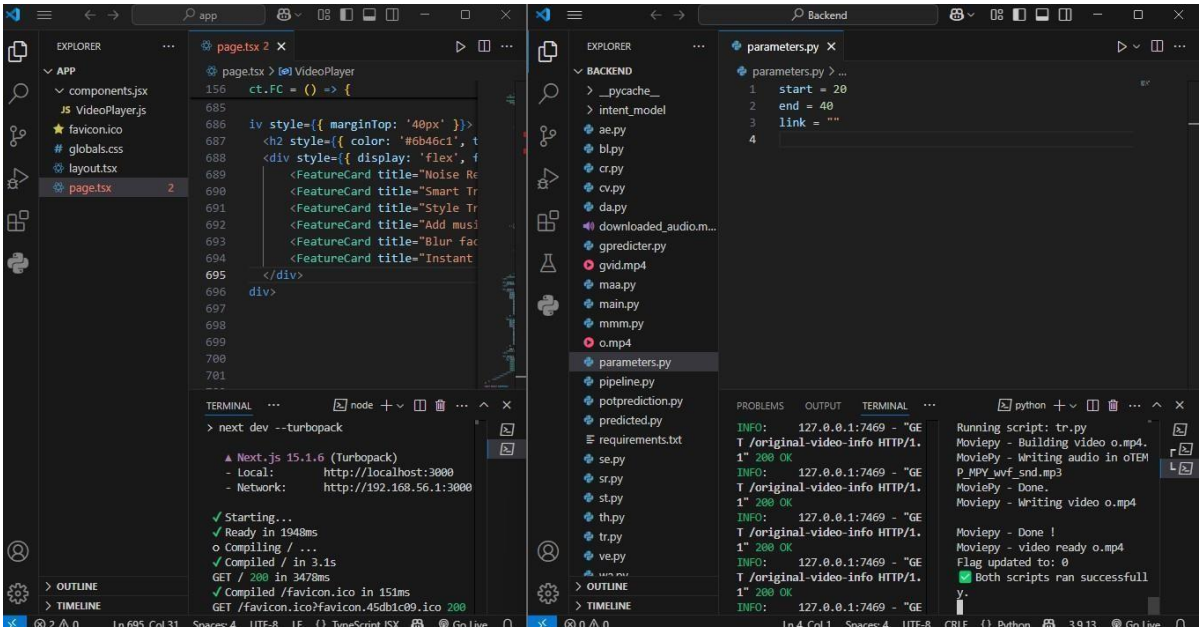


Fig 7.3.1 Snapshot of processing backend output video

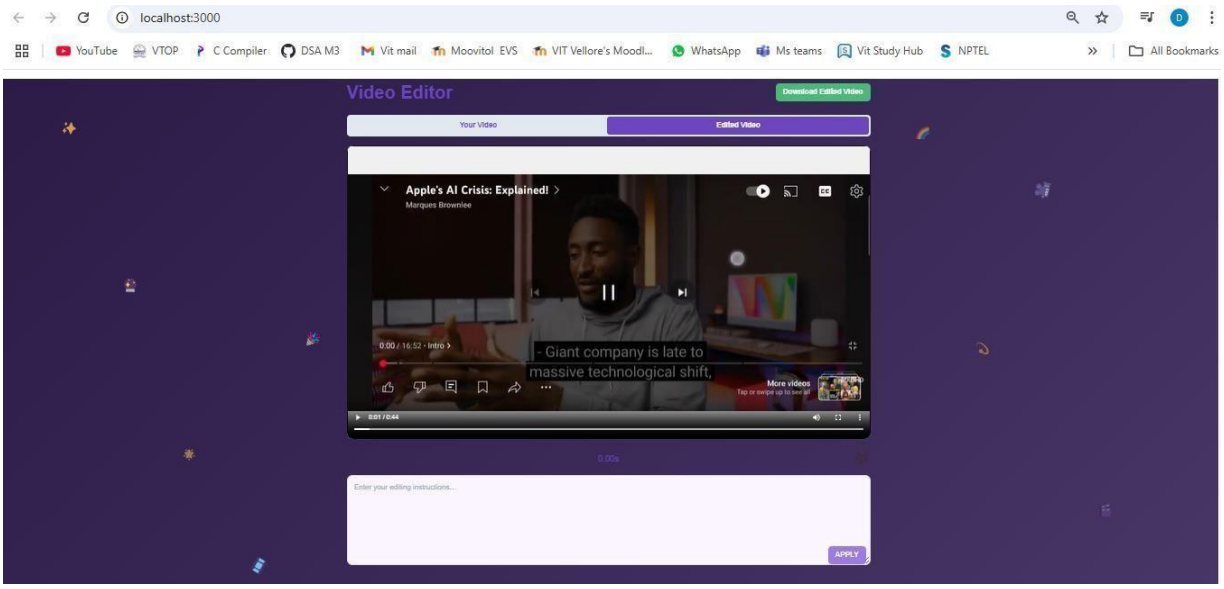


Fig 7.3.2 Snapshot of smart trimming output video

2. Style Transfer - By providing an input such as 'apply black and white filter to the video,' the system processes the request and applies the black and white effect to the entire video accordingly.

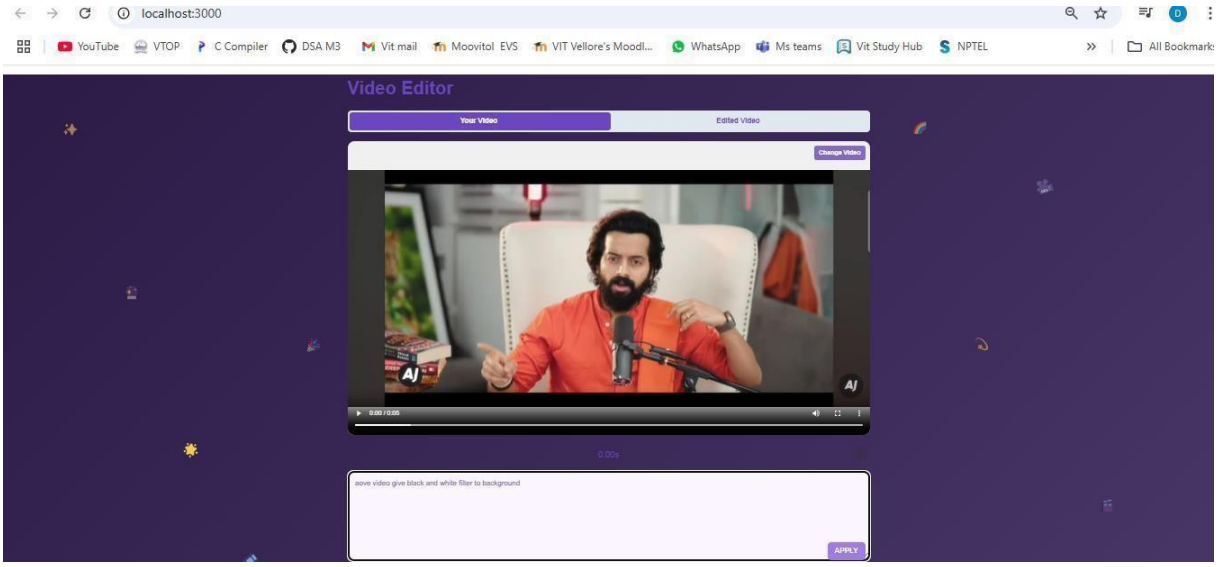


Fig 7.4 Snapshot of style transfer input video

In the backend, all the user provided input is processed automatically by the system, where each selected feature such as style transfer is executed in sequence to generate the final edited video.

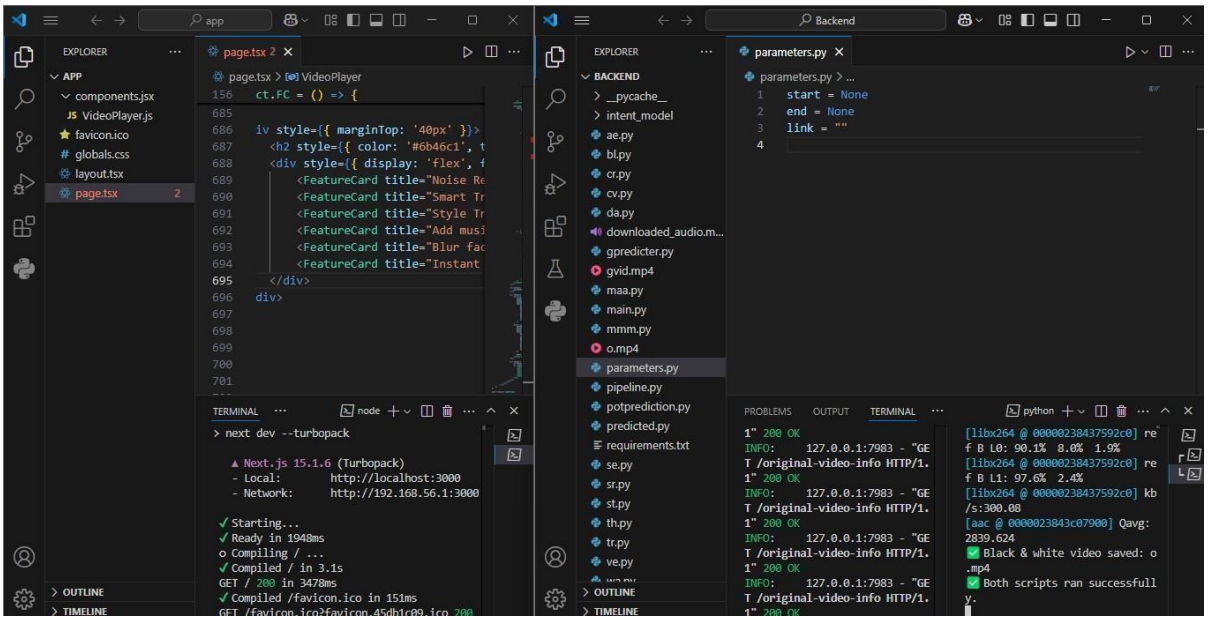


Fig 7.4.1 Snapshot of processing backend output video

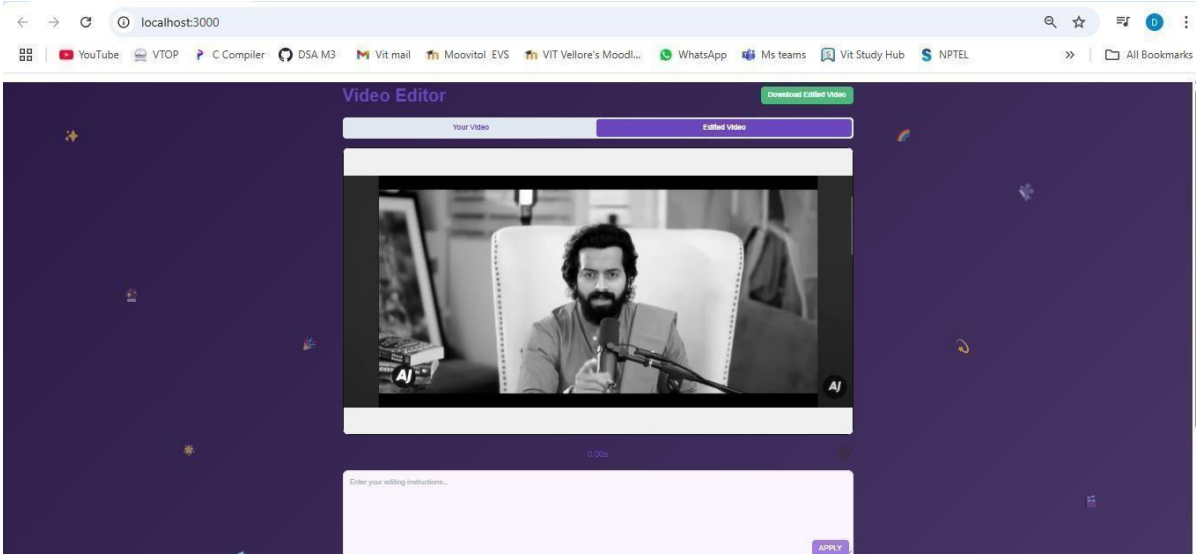


Fig 7.4.2 Snapshot of style transfer output video

3. **Add Music By Link** - By providing an input, such as a music link, the system processes the request and adds the corresponding audio as background music to the video.

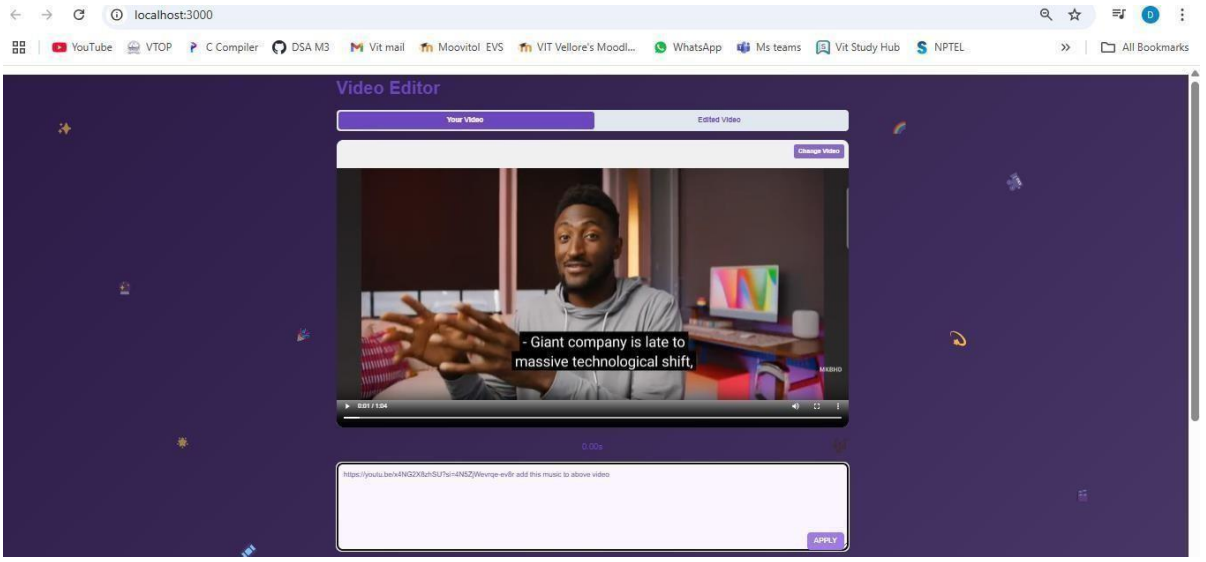


Fig 7.5 Snapshot of add music by link input video

In the backend, all the user provided input is processed automatically by the system, where each selected feature such as add music by link is executed in sequence to generate the final edited video.

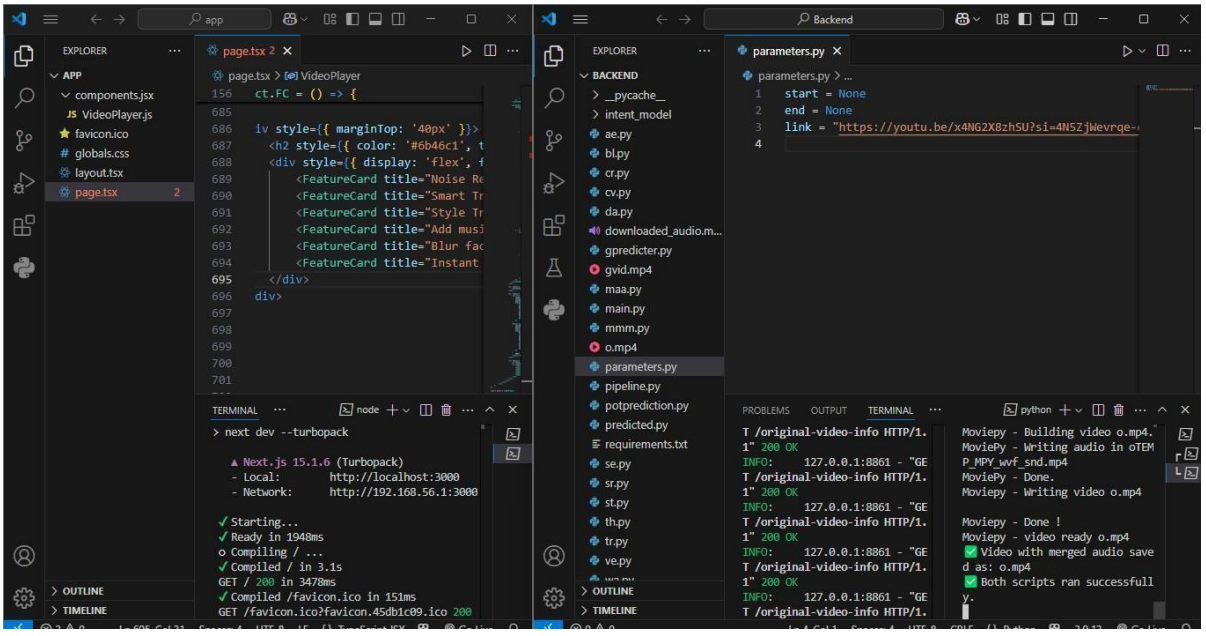


Fig 7.5.1 Snapshot of processing backend output video

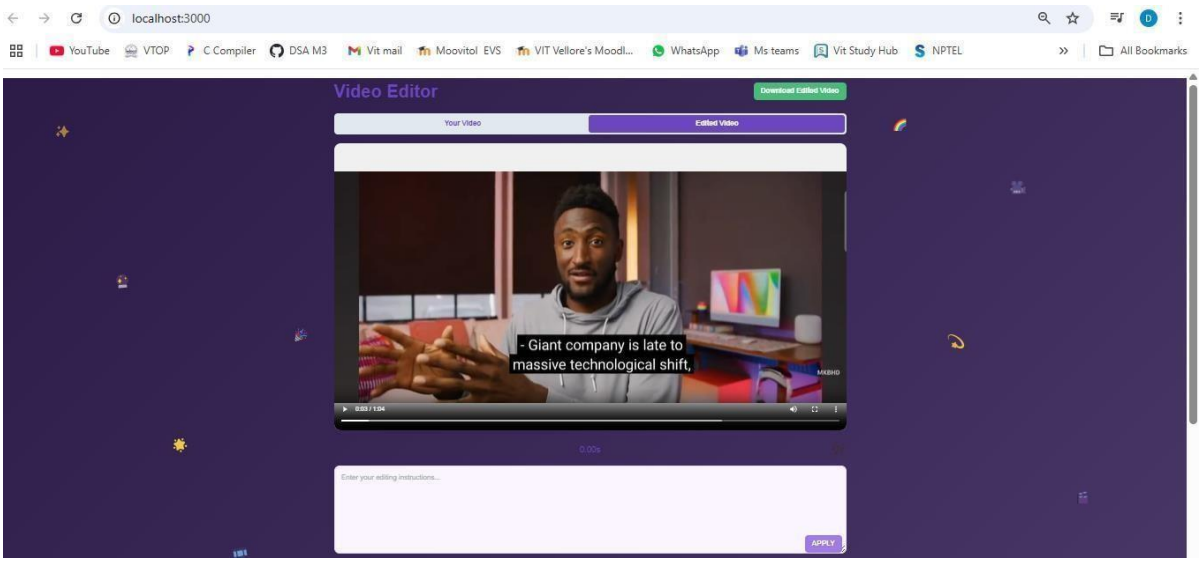


Fig 7.5.2 Snapshot of add music by link output video

4.Blur Face - By providing an input such as 'blur the face in the video,' the system processes the request and automatically detects and blurs all visible faces in the video.

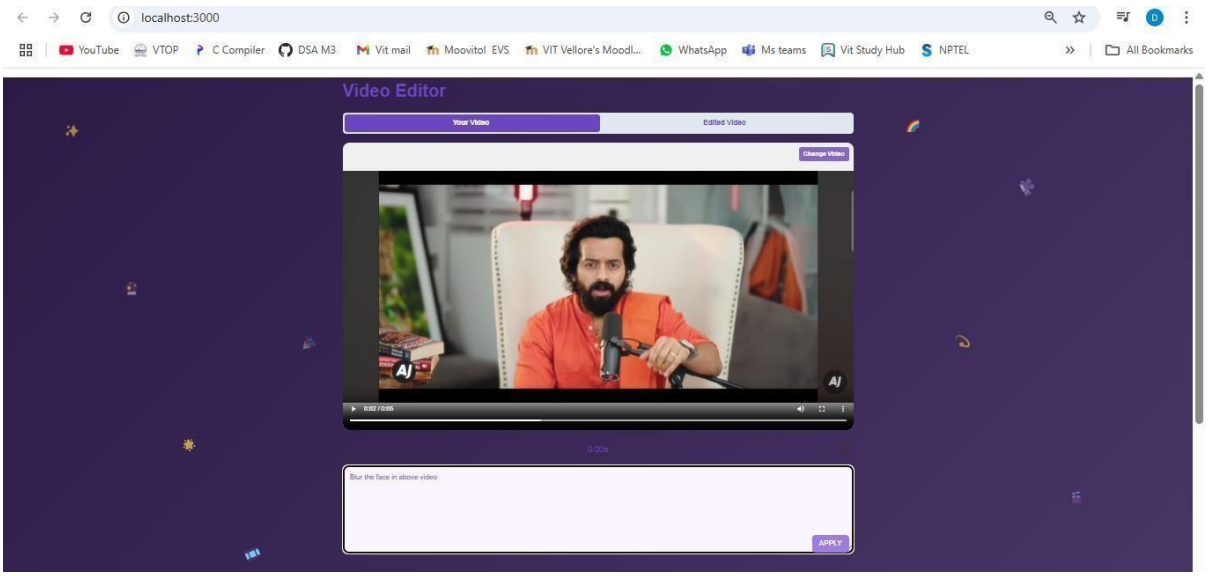


Fig 7.6 Snapshot of blur the face input video

In the backend, all the user provided input is processed automatically by the system, where each selected feature such as blur the face is executed in sequence to generate the final edited video.

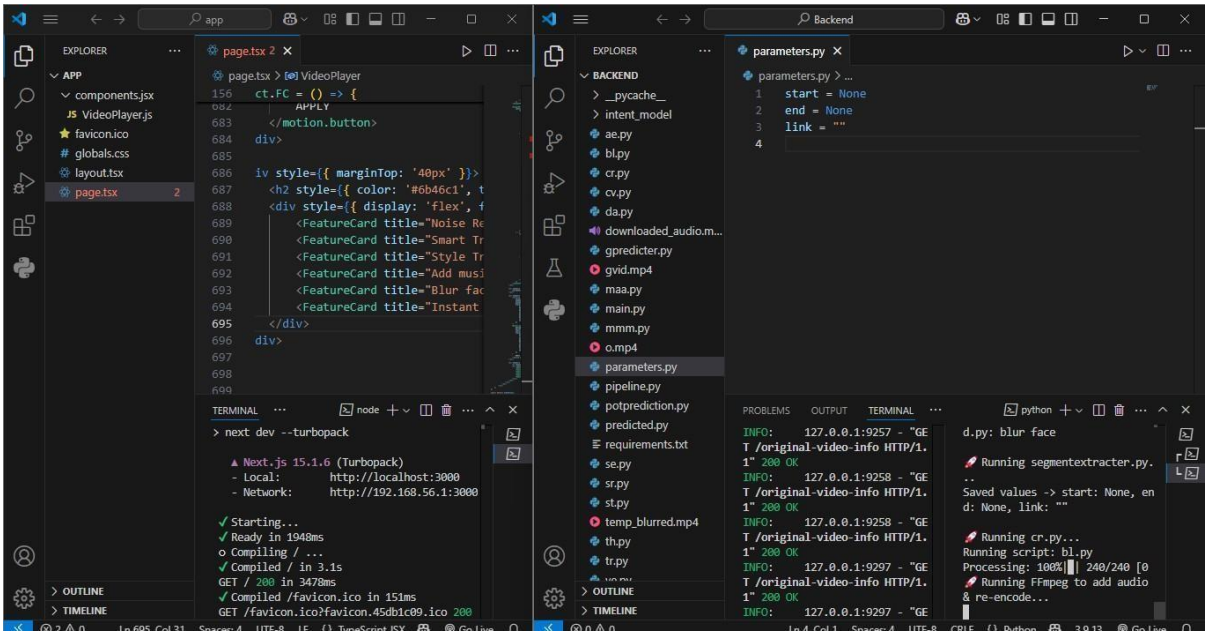


Fig 7.6.1 Snapshot of processing backend output video

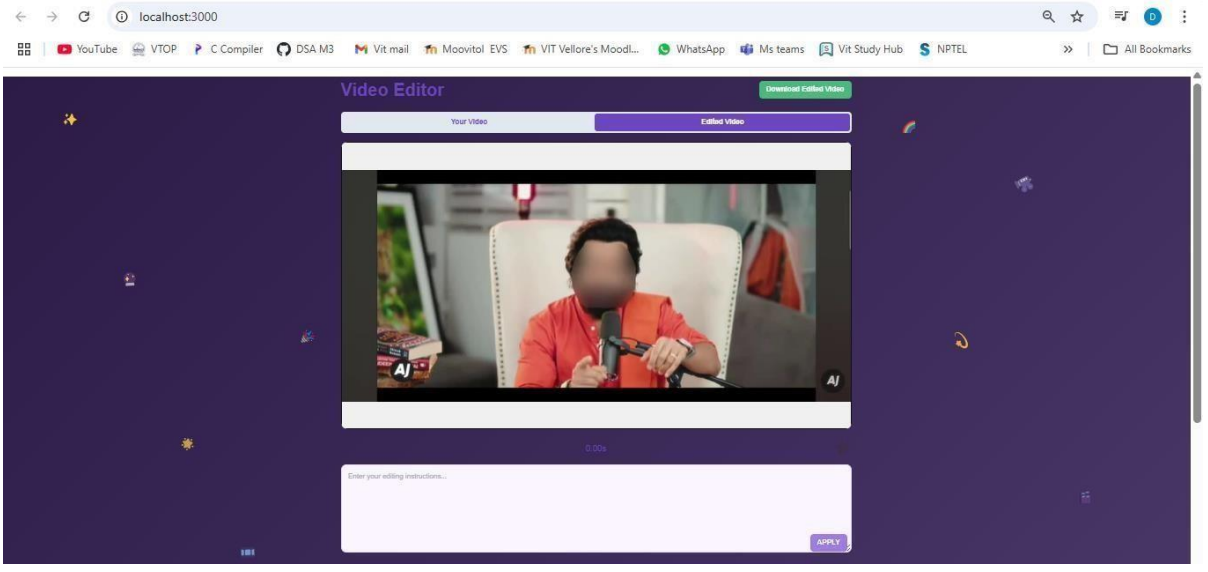


Fig 7.6.2 Snapshot of blur the face output video

8. CONCLUSION AND FUTURE ENHANCEMENTS

This project explored the application of artificial intelligence in automating and enhancing the video editing process. Traditional video editing is often time-consuming and requires significant technical skill, particularly when dealing with large volumes of raw footage or when striving for professional-level outputs. By leveraging AI, the system developed in this project significantly reduces the manual effort involved, streamlines the editing workflow, and makes high-quality video editing accessible to a wider audience.

The AI-based video editor integrated a range of intelligent features such as noise removal, smart trimming, style transfer, music addition via URL, and face blurring. Each of these functionalities was designed to work with minimal user input. For instance, smart trimming could automatically remove unnecessary segments from the video, while face detection algorithms ensured privacy by blurring faces without requiring manual masking. Style transfer models added creative filters to transform the visual aesthetic of the video, and music could be added from online links, automatically synchronized with the video content. These features worked together to provide a seamless editing experience that balanced automation with creative control.

Through testing, the system proved to be both effective and user-friendly. It performed well across different types of content, including interviews, vlogs, and casual recordings. The overall processing time was reduced, and the quality of output was generally high. However, the system was not without limitations. There were occasional issues with audio-video synchronization, and trimming logic sometimes removed important pauses or transitions in narrative-driven videos. Processing high-resolution videos also introduced performance bottlenecks, particularly in tasks involving style transfer and face blurring, which are GPU intensive.

Despite these challenges, the project successfully demonstrates that AI has a strong role to play in the future of video editing. The system may not yet match the precision and creativity of a professional human editor, but it can serve as a valuable assistive tool especially for content creators, educators, marketers, and casual users who need quick and efficient editing solutions.

FUTURE ENHANCEMENTS

The current implementation of AI-powered video editing has proven effective in automating several editing tasks such as noise removal, smart trimming, face blurring, style transfer, and background music addition, there is still significant room for improvement and expansion. Future enhancements can focus on improving the system's accuracy, efficiency, user-friendliness, and adaptability to cater to a wider audience and more complex editing requirements.

One of the most important future enhancements is the integration of context-aware editing. Currently, smart trimming is based on basic rules such as silence detection or lack of motion. However, with advancements in deep learning and natural language processing, future versions can incorporate models that understand the narrative structure, emotional tone, and importance of each scene. This would allow the system to retain meaningful content while trimming only the truly unnecessary parts, especially for storytelling or documentary style videos.

Another critical enhancement is real-time video editing and GPU optimization. Currently, tasks like style transfer and face blurring are resource intensive and may take considerable time to process high-resolution videos. Future implementations could leverage optimized AI inference engines along with dedicated GPU support to enable faster processing, potentially achieving near real-time performance. This would be particularly useful for live-streaming scenarios or situations requiring quick turn around times.

Cloud-based integration is another area of opportunity. By moving the processing to the cloud, users can edit videos without needing powerful local hardware. This would also enable features such as collaborative editing, multi-user access, and real-time updates. Additionally, storing and processing videos in the cloud can simplify version control and allow for seamless integration with platforms like Google Drive, Dropbox, or YouTube.

Multilingual subtitle generation is also an important feature to consider. While the current system might support basic speech-to-text transcription, expanding it to include real-time translations and support for multiple languages would make the platform more globally inclusive. This is particularly beneficial for educational content, international content creators, and social media influencers targeting diverse audiences.

9. REFERENCES

[1]. Chen, D., Yuan, L., Liao, J., Yu, N., & Hua, G. (2017). *StyleBank: An explicit representation for neural image style transfer*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 1897–1906.

[2]. Ronneberger, O., Fischer, P., & Brox, T. (2015). *U-Net: Convolutional networks for biomedical image segmentation*. In International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI) (pp. 234–241). Springer.

[3]. Valin, J. M. (2018). *A hybrid DSP/deep learning approach to real-time full-band speech enhancement*. arXiv preprint arXiv:2005.03760.

[4]. Zhang, K., Zuo, W., Chen, Y., Meng, D., & Zhang, L. (2017). *Beyond a Gaussian denoiser: Residual learning of deep CNN for image denoising*. IEEE Transactions on Image Processing, 26(7), 3142–3155.

[5]. Redmon, J., & Farhadi, A. (2018). *YOLOv3: An incremental improvement*. arXiv preprint arXiv:1804.02767.

[6]. Google. (n.d.). *YouTube Data API v3*. Google Developers. Retrieved from https://developers.google.com/youtube/v3

[7]. OpenAI. (2023). *Whisper: Speech recognition model*. OpenAI. Retrieved from

[8]. NVIDIA. (n.d.). *TensorRT: High-performance deep learning inference optimizer and runtime library*. NVIDIA Developer. Retrieved from

[9]. MediaPipe. (n.d.). *MediaPipe Face Detection*. Google. Retrieved from https://mediapipe.dev/

[10]. Sharma, A., & Gaur, M. (2018). *A deep learning-based approach for image and video captioning: A review*. 2018 IEEE Calcutta Conference (CALCON), 223-228.

[11]. Kim, S., Kim, J., & Kim, H. (2019). *Real-time video object detection with deep learning on GPUs*. Journal of Supercomputing, 75(6), 3469–3487.

[12]. Tao, A., Li, Z., & Han, M. (2017). *Speech enhancement using deep learning for audio and video editing*. In IEEE Transactions on Audio, Speech, and Language Processing,

[13]. Wang, Z., Liu, W., & Zhang, Y. (2018). *Deep learning models for real-time video content enhancement and enhancement in editing workflows*. Journal of Real-Time Image Processing, 15(3), 635–648.

[14]. Wang, T. et al. (2021). "Learning Retargetable Video Representations for Efficient Video Editing." *ICCV*.

[15]. Agerri, R., & Maza, I. (2020). *Real-time video editing using deep learning for media applications*. In Proceedings of the 2020 International Conference on Digital Signal Processing (DSP), 201-206.

[16]. Park, T., Zhu, J. Y., Isola, P., & Efros, A. A. (2017). *Unpaired image-to-image translation using cycle-consistent adversarial networks*. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2223–2232.

[17]. Rastegari, M., Ordonez, V., & Redmon, J. (2016). *XNOR-Net: ImageNet classification using binary convolutional neural networks*. In Proceedings of the European Conference on Computer Vision (ECCV), 525–542.

APPENDIX A – SAMPLE CODE

Predicter Code

```
import torch

from transformers import BertTokenizer, BertForSequenceClassification

import subprocess

from pipeline import INTENT  # Make sure this import is correct

# Load the trained model and tokenizer

model = BertForSequenceClassification.from_pretrained("./intent_model") # Load your trained model

tokenizer = BertTokenizer.from_pretrained("bert-base-uncased") # Load tokenizer used for training

# Define the intent labels (from your dataset)

intent_labels = {

    0: 'intent', 1: 'video enhance', 2: 'silencer remove', 3: 'stabilize',

    4: 'add music by link', 5: 'audio enhance', 6: 'gif', 7: 'dark',

    8: 'blur face', 9: 'trim', 10: 'black and white', 11: 'warm'

}

# Function to predict intent

def predict_intent(text):

    """Predicts the intent based on input text using the BERT model."""

    # Tokenize the input text

    inputs = tokenizer(text, return_tensors="pt", truncation=True, padding=True, max_length=64)

    # Get the model's output

    with torch.no_grad():  # Disable gradient calculation for inference
```

```
        outputs = model(**inputs)
    # Get the predicted label index
    pred = torch.argmax(outputs.logits, dim=1).item()
    # Map the predicted index to the corresponding intent
    predicted_intent = intent_labels[pred]
    return predicted_intent
# Function to write the predicted intent to predicted.py
def write_prediction_to_file(predicted_intent):
    """Writes the predicted intent to predicted.py and confirms success."""
    try:
        with open("predicted.py", "w") as f:
            f.write(f"predicted_intent = {repr(predicted_intent)}\n")
        print(f" Prediction saved to predicted.py: {predicted_intent}")
        return True
    except Exception as e:
        print(f"+ Error writing to predicted.py: {e}")
        return False
# Function to run segmentextracter.py and cr.py sequentially
def run_post_prediction_scripts():
    """Runs segmentextracter.py and then cr.py after successful prediction."""
    try:
        print("\n Running segmentextracter.py...")
        subprocess.run(["python", "se.py"], check=True)
        print("\n Running  cr.py...")
```

```
        subprocess.run(["python", "cr.py"], check=True)
        p r i n t ( "  Both scripts ran successfully.")
    except subprocess.CalledProcessError as e:
        print(f"+ Error executing scripts: {e}")
# Main execution
try:
    test_sentence = INTENT  # Replace INTENT with actual text input
    predicted_intent = predict_intent(test_sentence)
    print(f"  Predicted Intent: {predicted_intent}")
    # Write the predicted intent to the file
    if write_prediction_to_file(predicted_intent):
        run_post_prediction_scripts() # Run segmentextracter.py and cr.py
    else:
        print("ı. Skipping additional scripts due to failed prediction save.")
except Exception as e:
    print(f"+ Error processing intent: {e}")
```

Backend to Frontend Connection Using Fast Api Code

```
from fastapi import FastAPI, UploadFile, File, HTTPException
from fastapi.middleware.cors import CORSMiddleware
from fastapi.responses import JSONResponse, FileResponse
from pathlib import Path
import os
app = FastAPI()
app.add_middleware(
    CORSMiddleware,
    allow_origins=["*"],
    allow_credentials=True,
    allow_methods=["*"],
    allow_headers=["*"]
)
BASE_DIR = Path(__file__).parent.resolve()
PIPELINE_FILE_PATH = BASE_DIR / "pipeline.py"
USER_VIDEO_PATH = BASE_DIR / "gvid.mp4"  # Changed from user_video.mp4
ORIGINAL_VIDEO_PATH = BASE_DIR / "o.mp4"
original_mtime = 0
def update_pipeline(param: str, value):
    data = {}
    if PIPELINE_FILE_PATH.exists():
        try:
            with open(PIPELINE_FILE_PATH, "r") as file:
```

```
            lines = file.readlines()
        for line in lines:
            if "=" in line:
                key, val = line.strip().split("=", 1)
                key = key.strip()
                val = val.strip()
                if val.startswith("") and val.endswith(""):
                    val = val[1:-1]
                elif val.replace(".", "", 1).isdigit():
                    val = float(val) if "." in val else int(val)
                data[key] = val
    except Exception as e:
        print(f"Error reading pipeline: {e}")
data[param] = value
with open(PIPELINE_FILE_PATH, "w") as file:
    for key, val in data.items():
        if isinstance(val, str):
            file.write(f"{key} = \"{val}\"\n")
        else:
            file.write(f"{key} = {val}\n")
@app.post("/upload-video/")
async def upload_video(file: UploadFile = File(...)):
    try:
```

```
        # Validate file type (optional, can be enhanced further)
        if not file.content_type.startswith('video/'):
            raise HTTPException(400, detail="Invalid file type. Please upload a video file.")
        # Write the new video file, overwriting the existing one
        with USER_VIDEO_PATH.open("wb") as buffer:
            content = await file.read()
            buffer.write(content)
        update_pipeline("user_video", "gvid.mp4") # Updated to gvid.mp4
        return JSONResponse(
            status_code=200,
            content={"filename": "gvid.mp4", "message": "Video uploaded"}
        )
    except Exception as e:
        raise HTTPException(500, detail=f"Upload error: {str(e)}")
    finally:
        await file.close()
@app.get("/user-video")
async def get_user_video():
    if not USER_VIDEO_PATH.exists():
        raise HTTPException(404, detail="User video not found")
    return FileResponse(USER_VIDEO_PATH)
@app.get("/original-video")
async def get_original_video():
    if not ORIGINAL_VIDEO_PATH.exists():
```

```
        raise HTTPException(404, detail="Original video not found")
    return FileResponse(ORIGINAL_VIDEO_PATH)
@app.get("/original-video-info")
async def get_original_video_info():
    global original_mtime
    if ORIGINAL_VIDEO_PATH.exists():
        current_mtime = os.path.getmtime(ORIGINAL_VIDEO_PATH)
        if current_mtime != original_mtime:
            original_mtime = current_mtime
            return {"modified": True}
    return {"modified": False}
@app.post("/set-intent/")
async def set_intent(data: dict):
    try:
        intent = data.get("intent")
        if not intent:
            raise HTTPException(400, detail="Intent required")
        update_pipeline("INTENT", intent)
        return {"message": "Intent updated"}
    except Exception as e:
        raise HTTPException(500, detail=str(e))
@app.post("/set-current-time/")
async def set_current_time(data: dict):
    try:
```

```
        current_time = data.get("current_time")
        if current_time is None:
            raise HTTPException(400, detail="Time required")
        update_pipeline("current_time", current_time)
        return {"message": "Time updated"}
    except Exception as e:
        raise HTTPException(500, detail=str(e))
@app.get("/get-parameters/")
async def get_parameters():
    try:
        data = {}
        if PIPELINE_FILE_PATH.exists():
            with open(PIPELINE_FILE_PATH, "r") as file:
                content = file.read()
                if content.strip():
                    exec(content, {}, data)
        return data
    except Exception as e:
        raise HTTPException(500, detail=f"Parameter error: {str(e)}")
if __name__ == "__main__":
    import uvicorn
    uvicorn.run(app, host="0.0.0.0", port=8000, reload=True)
```

Music Via Link Code

```
import os
import re
import glob
from yt_dlp import YoutubeDL
from moviepy.video.io.VideoFileClip import VideoFileClip
from moviepy.audio.io.AudioFileClip import AudioFileClip
from parameters import link
VIDEO_FILE = "gvid.mp4"  # Replace with your video file path
YOUTUBE_URL = link  # YouTube URL from parameters.py
OUTPUT_FILE = "o.mp4"  # Fixed output file name
def sanitize_filename(filename):
    """Removes or replaces special characters in filenames."""
def download_youtube_audio(url):
    """
    Downloads audio from a YouTube URL using yt-dlp and ensures safe filename handling.
    """
    sanitized_output = sanitize_filename("downloaded_audio")
    ydl_opts = {
        'format': 'bestaudio/best',
        'outtmpl': f"{sanitized_output}.%(ext)s", # Force sanitized filename
        'postprocessors': [{
            'key': 'FFmpegExtractAudio',
            'preferredcodec': 'mp3',
```

```
            'preferredquality': '192',
        }],
        'noplaylist': True,  # Download only a single video
        'nocheckcertificate': True,  # Prevent SSL issues
        'force_overwrites': True,  # Overwrite if file exists
    }
    try:
        with YoutubeDL(ydl_opts) as ydl:
            ydl.extract_info(url, download=True)
        audio_files = glob.glob("downloaded_audio*.mp3")
        if not audio_files:
            print("+ Error: Audio file not found after download!")
            return None
        downloaded_audio = audio_files[0] # Get the first matching file
        p r i n t ( f "  Audio downloaded successfully:
        {downloaded_audio}") return downloaded_audio
    except Exception as e:
        print(f"+ Failed to download audio: {e}")
        return None
def merge_audio_with_video(video_path, audio_path, output_path="op.mp4"):
    """
    Merges a video file with an audio file.
    - If audio is longer than video, trim audio to video length.
    - If video is longer than audio, keep video length (no audio after video ends).
```

```
    """
    if not os.path.exists(audio_path):
        print(f"+ Error: Audio file '{audio_path}' not found!")
        return
    try:
        video_clip = VideoFileClip(video_path)
        audio_clip = AudioFileClip(audio_path)
        # If audio is longer than video, trim audio to video length
        if audio_clip.duration > video_clip.duration:
            print("◆ Audio is longer than video. Trimming audio to match video length.")
            audio_clip = audio_clip.subclip(0, video_clip.duration)
        final_video = video_clip.set_audio(audio_clip)
        final_video.write_videofile(output_path, codec="libx264", audio_codec="aac")
        p r i n t ( f "  Video with merged audio saved as: {output_path}")
    except Exception as e:
        print(f"+ An error occurred while merging: {e}")
if _name___== "_main_":
    print("n Downloading audio from YouTube...")
    downloaded_audio = download_youtube_audio(YOUTUBE_URL)
    if downloaded_audio:
        p r i n t ( " Merging downloaded audio with video...")
        merge_audio_with_video(VIDEO_FILE, downloaded_audio, OUTPUT_FILE)
    else:
        print("+ Failed to download audio. Exiting.")
```

Trim Video Code

```
from moviepy.editor import VideoFileClip, concatenate_videoclips
from parameters import start, end
 # Import cv to modify its flag variable
def update_flag(value: int):
    """
    Updates the flag value in cv.py by writing directly into the file.
    Parameters:
      value: The new value for the flag.
    """
    with open("cv.py", "w") as file:
        file.write(f"flag = {value}\n")
def remove_segment(input_file: str, output_file: str, start_time: float, end_time: float):
    """
    Removes a segment of the video from start_time to end_time and writes the remaining parts
    into a new video file.
    Parameters:
      input_file: Path to the input video file.
      output_file: Path where the resulting video will be saved.
      start_time: The starting time (in seconds) to remove.
      end_time: The ending time (in seconds) to remove.
    """
    with VideoFileClip(input_file) as clip:
```

```
        if start_time < 0 or end_time > clip.duration or start_time >= end_time:
            raise ValueError("Invalid start_time and/or end_time.")
        clips_to_concat = []
        # Extract the segment from the beginning up to start_time, if any.
        if start_time > 0:
            first_clip = clip.subclip(0, start_time)
            clips_to_concat.append(first_clip)
        # Extract the segment from end_time to the end of the video, if any.
        if end_time < clip.duration:
            second_clip = clip.subclip(end_time, clip.duration)
            clips_to_concat.append(second_clip)
        # Check if there's any remaining video to concatenate.
        if not clips_to_concat:
            raise ValueError("The removal segment covers the entire video.")
        # Concatenate the remaining segments.
        remaining_clip = concatenate_videoclips(clips_to_concat)
        remaining_clip.write_videofile(output_file, codec="libx264")
    # Update flag in cv.py after completion
    update_flag(0)  # Set flag to 0 by writing directly into cv.py
    print("Flag updated to:", 0)
if __name__ == "__main__":
    # Example usage: remove the segment between 'start' seconds and 'end' seconds from the video.
    remove_segment("gvid.mp4", "o.mp4", start, end)
```

Blur The Face Code

```
import cv2
import numpy as np
from tqdm import tqdm
import subprocess
import os
class VideoFaceBlurrer:
    def __init__(self):
        self.face_cascade = cv2.CascadeClassifier(cv2.data.haarcascades + 
'haarcascade_frontalface_default.xml')
        self.blur_strength = 85  # Must be odd number
        self.oval_ratio = 0.6
    def detect_faces(self, frame):
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        return self.face_cascade.detectMultiScale(gray, 1.3, 5)
    def oval_blur_effect(self, face_roi):
        h, w = face_roi.shape[:2]
        mask = np.zeros((h, w), dtype=np.uint8)
        cv2.ellipse(mask, (w//2, h//2), (int(w/2), int(h*self.oval_ratio)), 0, 0, 360, 255, -1)
        blurred = cv2.GaussianBlur(face_roi, (self.blur_strength, self.blur_strength), 0)
        return np.where(mask[:,:,None] == 255, blurred, face_roi)
    def process_video(self, input_path, temp_output_path, final_output_path):
        cap = cv2.VideoCapture(input_path)
        if not cap.isOpened():
```

```
    raise ValueError(f"Could not open video: {input_path}")
fps = cap.get(cv2.CAP_PROP_FPS)
width = int(cap.get(cv2.CAP_PROP_FRAME_WIDTH))
height = int(cap.get(cv2.CAP_PROP_FRAME_HEIGHT))
total_frames = int(cap.get(cv2.CAP_PROP_FRAME_COUNT))
# Safe codec
fourcc = cv2.VideoWriter_fourcc(*'mp4v')
out = cv2.VideoWriter(temp_output_path, fourcc, fps, (width, height))
pbar = tqdm(total=total_frames, desc='Processing', unit='frame')
while True:
    ret, frame = cap.read()
    if not ret:
        break
    faces = self.detect_faces(frame)
    for (x, y, w, h) in faces:
        face_roi = frame[y:y+h, x:x+w]
        frame[y:y+h, x:x+w] = self.oval_blur_effect(face_roi)


    out.write(frame)
    pbar.update(1)
pbar.close()
cap.release()
out.release()
cv2.destroyAllWindows()
```

```
        # Merge audio and re-encode
        self.merge_audio_and_encode(input_path, temp_output_path, final_output_path)
    def merge_audio_and_encode(self, original_video, video_no_audio, output_final):
        command = [
            "ffmpeg",
            "-y", # overwrite without asking
            "-i", video_no_audio,
            "-i", original_video,
            "-c:v", "libx264",
            "-preset", "fast",
            "-c:a", "aac",
            "-map", "0:v:0",
            "-map", "1:a:0",
            "-shortest",
            output_final
        ]
        🎬 Running FFmpeg to add audio & re-encode...")
        subprocess.run(command, stdout=subprocess.PIPE, stderr=subprocess.PIPE)
        p r i n t ( f " ✅ Final output saved as: {output_final}")
if __name__== "__main__":
    blurrer = VideoFaceBlurrer()
    input_video = "gvid.mp4"
    temp_video = "temp_blurred.mp4"
    final_video = "o.mp4"
```

```
blurrer.process_video(input_video, temp_video, final_video)

# Optional: clean up temp file

if os.path.exists(temp_video):

    os.remove(temp_video)
```