RASISTEM OTOMATISASI RENDERING VIDEO TEMPLATE AFTER EFFECTS BERBASIS WEB DENGAN EXTENDSCRIPT DAN FLASK

WEB-BASED VIDEO TEMPLATE AFTER EFFECTS RENDERING AUTOMATION SYSTEM WITH EXTENDSCRIPT AND FLASK

Ragil Raditya Saputra[1)], Zacky Innova[2)], Epsilona Katiga Capricorna[3)], Bambang Irawan[4)], Ridwan Ramadhan[5)], Muhammad Hafizh Rahman Hakim [6)]

[1), 2),3),4),5),6)] Teknik Informatika, Universitas Esa Unggul
Indonesia
e-mail: ragil.raditya2005@student.esaunggul.ac.id[1)], Zackyinova1@student.esaunggul.ac.id[2)], epsilonakatigacapricorna@student.esaunggul.ac.id[3)], bambang.irawan@esaunggul.ac.id[4)], ramadhanridwan660@student.esaunggul.ac.id[5)], mhafizhrhakim@student.esaunggul.ac.id[6)]

ABSTRAK

Proses pembuatan animasi atau video dalam Adobe After Effects umumnya memerlukan interaksi manual yang berulang dan memakan waktu, sehingga kurang efisien dalam alur kerja multimedia digital. Untuk menjawab tantangan tersebut, penelitian ini merancang sistem otomatisasi rendering berbasis web dengan mengintegrasikan ExtendScript sebagai bahasa scripting internal After Effects dan framework Flask sebagai backend. Antarmuka berbasis HTML memungkinkan pengguna menginput parameter seperti teks, warna, atau pilihan format, yang kemudian diproses oleh Flask dan diteruskan ke After Effects melalui subprocess *lokal. Tak hanya integrasi, kami juga telah membandingkan sistem otomatisasi ini dengan metode manual. Penelitian ini menggunakan pendekatan eksperimental dalam bentuk pengembangan dan pengujian sistem untuk mengevaluasi efektivitasnya. Sistem ini mampu meningkatkan efisiensi kerja hingga 67,78% dibandingkan metode manual tradisional, serta mendukung pemrosesan secara otomatis berdasarkan input pengguna. Sistem ini memberikan solusi praktis dan terjangkau bagi pengguna yang ingin mempercepat produksi konten visual dari template After Effects dengan penyesuaian pada teks dan warna, tanpa mengurangi fleksibilitas dan kualitas hasil akhir. Integrasi teknologi ini membuktikan bahwa otomatisasi proses kreatif dapat dilakukan secara efektif, stabil, dan dapat diakses dengan mudah melalui web jika dikembangkan lebih lanjut.*

***Kata Kunci**: Adobe, Flask, Motion Graphic, Python, Video*

ABSTRACT

The process of creating animations or videos in Adobe After Effects generally requires repetitive manual interaction, making it inefficient in the digital multimedia workflow. To address this challenge, this study designed a web-based automated rendering system by integrating ExtendScript as the internal scripting language of After Effects and Flask as the backend framework. The HTML-based interface allows users to input parameters such as text, color, or format options, which are then processed by Flask and passed to After Effects via a local subprocess. In addition to the integration, we also compared this automated system with the manual method. This study employed an experimental approach through the development and testing of the system to evaluate its effectiveness. The system was able to improve workflow efficiency by up to 67.78% compared to traditional manual methods and supports automated processing based on user input. It offers a practical and affordable solution for user who want to accelerate visual content production from After Effects templates with customized text and color, without compromising the flexibility and quality of the final output. This technological integration demonstrates that creative process automation can be carried out effectively, stably, and can be easily accessed via the web if further developed.

***Keywords**: Adobe, Flask, Motion Graphic, Python, Video*

I. PENDAHULUAN

Adobe After Effects merupakan salah satu perangkat lunak utama yang banyak digunakan dalam industri kreatif untuk menghasilkan animasi, motion graphic, dan efek visual yang kompleks. Meskipun memiliki kemampuan yang sangat luas, proses rendering

pada After Effects masih bersifat manual dan repetitif, seperti pengaturan parameter proyek, pemilihan template, hingga eksekusi render. Kondisi ini menjadi hambatan dalam alur kerja yang membutuhkan efisiensi tinggi, terutama bagi pelaku industri kreatif yang menghadapi tuntutan produksi cepat dan tenggat waktu yang ketat.

Kebutuhan akan otomatisasi dalam produksi multimedia telah menjadi isu penting seiring meningkatnya permintaan konten visual di berbagai platform digital. Secara teoritis, konsep *workflow automation* dalam bidang multimedia telah terbukti mampu mengurangi beban kerja manual, mempercepat proses produksi, serta menekan potensi kesalahan manusia. Namun, belum banyak studi yang mengintegrasikan sistem otomatisasi rendering secara langsung ke dalam platform After Effects dengan pendekatan berbasis web, yang mudah diakses dan ramah pengguna.

Penelitian ini hadir untuk menjawab permasalahan tersebut dengan merancang dan mengimplementasikan sistem otomatisasi rendering video template After Effects yang dapat dijalankan melalui antarmuka web. Sistem ini memanfaatkan ExtendScript bahasa scripting internal Adobe untuk mengatur fungsi rendering secara otomatis, serta framework Flask sebagai backend yang mengelola input pengguna melalui browser. Dengan pendekatan ini, sistem diharapkan dapat menjadi solusi yang praktis, efisien, dan terjangkau, terutama bagi pengguna non-teknis yang ingin mempercepat proses produksi konten tanpa mengorbankan kualitas hasil akhir.

II. STUDI PUSTAKA

Otomatisasi proses produksi multimedia telah menjadi fokus banyak penelitian dalam beberapa tahun terakhir, terutama dalam konteks peningkatan efisiensi dan integrasi teknologi web dengan perangkat lunak kreatif.

Penelitian oleh Bonney et al. (2022) mengembangkan platform digital twin menggunakan Python Flask untuk mendukung operasionalisasi otomatis dalam lingkungan berbasis web, yang menunjukkan fleksibilitas Flask sebagai backend web ringan yang dapat diterapkan dalam berbagai konteks [1].

Wijayanto dan Susetyo (2022) menerapkan Flask dalam pengembangan sistem informasi helpdesk, menunjukkan bagaimana framework ini mampu menangani interaksi pengguna melalui form dan permintaan HTTP secara efisien [2].

Ngantung dan Pakereng (2021) merancang sistem informasi akademik dengan pendekatan Pengguna-centered design dan implementasi Flask, menegaskan kemudahan integrasi antara antarmuka pengguna dan backend berbasis Python [3.]

Dalam konteks integrasi dengan sistem rendering, Susanti dan Mailoa (2020) mengembangkan RESTful API menggunakan Flask untuk pengelolaan data visual melalui skema web service, menunjukkan potensi Flask dalam manajemen data media [4].

Sementara itu, Pratama dan Susetyo (2024) mengembangkan sistem pembaruan cloud menggunakan API Flask, dengan fokus pada konversi data warna dari format HTML ke standar RGB, yang relevan untuk konteks pemrosesan input warna pada sistem kami [5].

Kelima penelitian tersebut menunjukkan bahwa framework Flask merupakan solusi backend yang fleksibel dan adaptif untuk membangun sistem otomasi, termasuk dalam konteks multimedia. Namun, belum banyak penelitian yang secara spesifik mengintegrasikan Flask dengan Adobe After Effects menggunakan ExtendScript untuk tujuan rendering otomatis berbasis web, sehingga menjadi celah penelitian yang dijawab dalam studi ini.

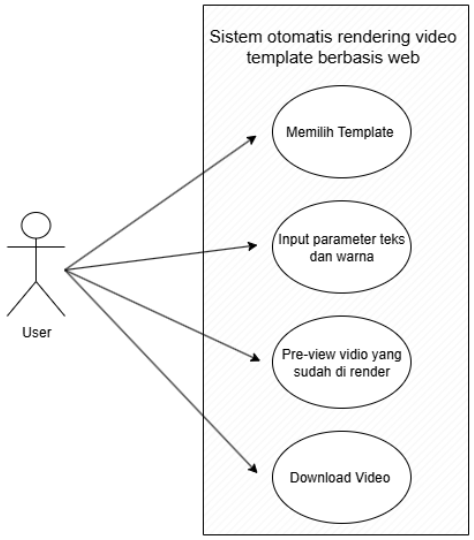
III. METODE PENELITIAN

Penelitian ini menggunakan pendekatan eksperimental dalam bentuk pengembangan dan pengujian sistem otomatisasi rendering video berbasis web. Sistem dirancang dengan menggunakan bahasa pemrograman dan scripting untuk mengintegrasikan backend web (Flask) dengan perintah lokal (ExtendScript) dalam menjalankan proses rendering.

A. Alur Sistem Otomatisasi

Pengguna memberikan input berupa teks dan warna melalui antarmuka HTML sebagaimana ditampilkan pada Gambar 4. Input tersebut kemudian diproses oleh Flask untuk membangkitkan file ExtendScript yang diterapkan pada proyek After Effects, sebagaimana diperlihatkan pada Gambar 6. Setelah proses rendering selesai, pengguna akan menerima hasil video secara otomatis melalui halaman hasil yang ditunjukkan pada Gambar 7. Gambar berikut

memperlihatkan alur kerja sistem mulai dari input parameter oleh pengguna hingga proses rendering selesai dan hasil video diterima kembali oleh pengguna.



Gambar 1. Diagram Alur Otomatisasi Proses Rendering Video

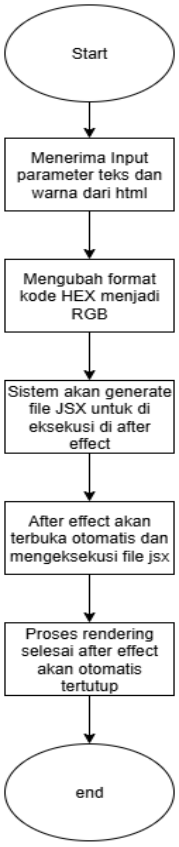
B. Perbandingan Metode Manual dan Otomatis

Metode manual dalam proses pembuatan video memerlukan keterlibatan pengguna yang cukup intensif dan memakan waktu. Pada pendekatan ini, pengguna harus membuka aplikasi After Effects secara langsung, kemudian melakukan serangkaian langkah yang repetitif seperti mengubah teks, menyesuaikan warna, dan mengonfigurasi pengaturan render. Setiap kali ada perubahan yang diperlukan, pengguna harus mengulangi seluruh proses ini secara manual, yang tidak hanya membutuhkan waktu yang lama tetapi juga meningkatkan risiko kesalahan manusia. Selain itu, metode ini memerlukan pemahaman teknis yang mendalam tentang interface After Effects dan workflow yang kompleks.

Sebaliknya, metode otomatis menawarkan pendekatan yang jauh lebih efisien dan user-friendly. Dengan sistem ini, pengguna hanya perlu mengisi form web sederhana yang berisi parameter yang dibutuhkan seperti teks dan warna. Setelah data disubmit, seluruh proses mulai dari input data, pembuatan script, eksekusi di After Effects, hingga rendering final dilakukan secara otomatis dengan intervensi manual sedikit. Sistem otomatis ini tidak hanya menghemat waktu secara signifikan, tetapi juga mengurangi kemungkinan kesalahan dan memungkinkan pengguna tanpa keahlian teknis After Effects untuk menghasilkan video berkualitas profesional dengan mudah.

C. Perancangan Sistem Backend

Backend sistem dikembangkan dengan flask yang bertanggung jawab untuk menangani permintaan HTTP, pengolahan parameter input. Selain itu, *ExtendScript (.jsx)* digunakan untuk mengontrol jalannya proses di Adobe After Effects secara otomatis [6].



Gambar 2. Alur Sistem Backend

Contoh fungsi route utama pada backend yang ditunjukkan oleh Gambar 2 tentang menerima input, sebagai berikut:

```
@app.route('/render', methods=['POST'])
def render_motion():
text = request.form['text']
color_hex = request.form['color'].lstrip('#')
```

Kode di atas memperlihatkan bagaimana sistem menerima input text dan color dari form HTML, terutama pada nilai warna yang awalnya berupa kode HEX menjadi format RGB [5], agar After Effects dapat membaca warna dari masukan pengguna. Seperti Gambar 2 tentang mengubah format kode HEX ke RGB

```
r = int(color_hex[0:2], 16) / 255
g = int(color_hex[2:4], 16) / 255
b = int(color_hex[4:6], 16) / 255
```

Setelah itu, sistem membangkitkan skrip JSX secara dinamis berdasarkan parameter input pengguna yang ditunjukkan pada Gambar 2. *ExtendScript* tersebut akan mengatur lapisan teks dan warna pada komposisi dalam proyek .aep:

```
_comp.layer("Text Layer").property("Source Text").setValue("{text}");
_comp.layer("Control Layer").property("Effects").property("Color Control").property("Color").setValue([{r}, {g}, {b}]);
```

Skrip JSX tersebut kemudian disimpan sementara dan dieksekusi oleh After Effects yang otomatis terbuka menggunakan perintah `subprocess` dari Python:

```
subprocess.call([afterfx_path, aep_path, "-r", jsx_path])
```

Proses ini sepenuhnya berjalan secara otomatis, dimulai dari pengambilan input pengguna, pembuatan skrip, setelah eksekusi file JSX After Effects, hingga hasil rendering yang disimpan secara otomatis di direktori `output` dengan penamaan berdasarkan text yang sudah dimasukan pengguna:

```
output_name = f"render_{text}.mp4"
```

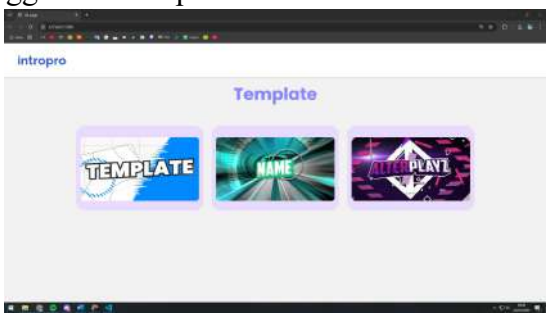
Setelah rendering selesai, pengguna diarahkan ke halaman preview dan dapat mengunduh hasil video secara langsung dari web.

Dengan sistem ini, proses yang sebelumnya dilakukan secara manual di After Effects kini dapat dilakukan dalam satu klik melalui antarmuka web, menghemat waktu dan mengurangi potensi kesalahan input.

D. Langkah – Langkah Penelitian

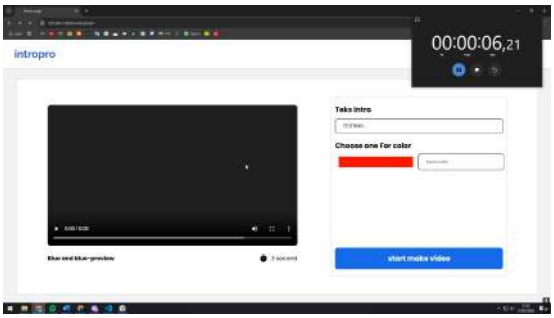
Setelah tahap perancangan dan pengembangan sistem otomatisasi rendering video berbasis web selesai dilakukan, langkah berikutnya adalah melakukan proses penelitian melalui pengujian terhadap sistem yang telah dibangun. Penelitian ini difokuskan pada pembandingan antara dua metode rendering video, yaitu metode otomatis berbasis web dengan metode manual di After Effects langsung.

Metode otomatis yang telah dikembangkan memungkinkan pengguna melakukan rendering video hanya dengan mengisi formulir teks dan warna melalui antarmuka web. Selanjutnya, proses render dilakukan secara otomatis oleh Adobe After Effects tanpa perlu interaksi manual. Gambar berikut menunjukkan tampilan antarmuka utama dari sistem berbasis web yang telah dikembangkan menggunakan stopwatch.



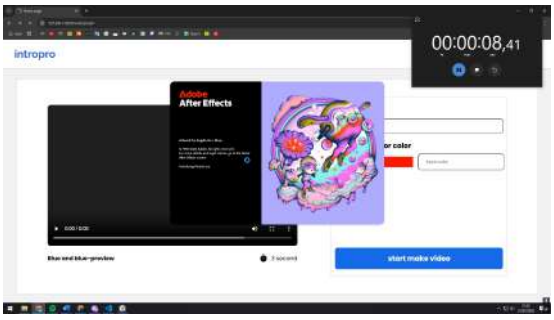
Gambar 3. Main Menu

Pada Gambar 3 Pengguna dapat memilih dari 3 template yang tersedia sesuai use case diagram di Gambar 1.

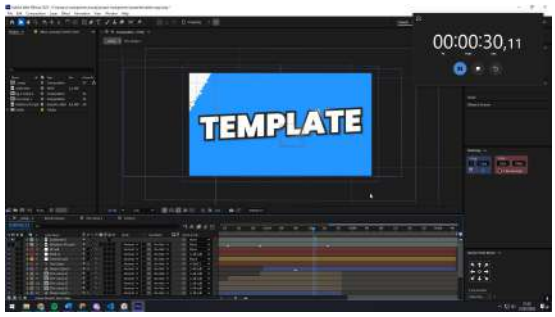


Gambar 4. *Pengguna* Mengisi Teks dan Warna

Di Gambar 4 tersedia form untuk Pengguna mengisi parameter teks dan warna yang diinginkan.

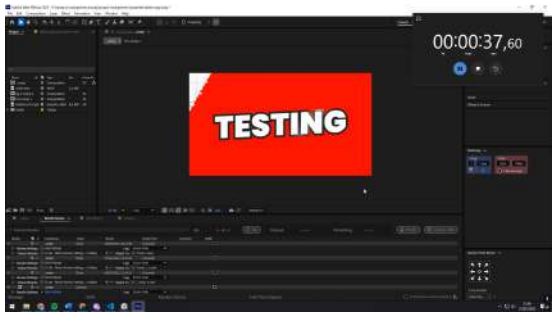


Gambar 5. Masuk Otomatis ke After Effects Untuk Eksekusi File JSX

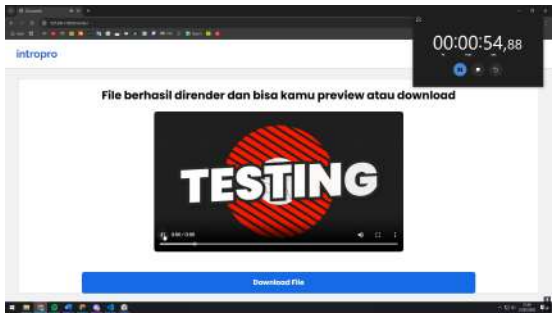


Gambar 6. Eksekusi File JSX

Saat masuk di dalam After Effects file JSX akan dieksekusi menggunakan shortcut, seperti alur sistem backend di Gambar 2. Setelah file JSX di eksekusi, parameter yang ada akan otomatis terganti dengan parameter yang sudah isi sebelumnya oleh pengguna dan langsung render di After Effects secara otomatis.



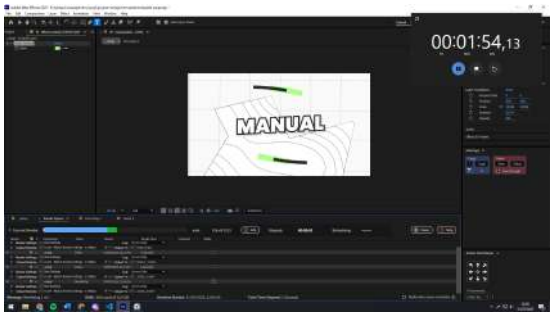
Gambar 7. Proses Rendering setelah eksekusi file *JSX*



Gambar 8. Proses Rendering Selesai

Gambar 8 menjelaskan tentang halaman yang berubah otomatis menjadi video preview dengan parameter yang sudah sesuai dengan parameter teks dan warna yang sudah diisi oleh pengguna sebelumnya. Video juga dapat diunduh oleh pengguna untuk disimpan, sesuai dengan use case diagram Gambar 1.

Sedangkan metode manual mengharuskan pengguna membuka proyek After Effects secara langsung, kemudian mengganti parameter seperti teks dan warna satu per satu secara manual, mengatur ulang komposisi, dan mengeksekusi render secara manual pada setiap perubahan. Seperti di Gambar 9.



Gambar 9. Proses Rendering Manual

Proses yang telah dijelaskan sebelumnya merupakan salah satu contoh dari lima kali percobaan yang dilakukan dalam penelitian ini. Percobaan tersebut dilakukan untuk membandingkan performa metode otomatis berbasis web dengan metode manual tradisional di Adobe After Effects. Perbandingan ini bertujuan untuk mengukur efisiensi waktu dan beban kerja pengguna pada masing-masing metode, serta mengevaluasi sejauh mana sistem otomatisasi yang dibangun mampu meningkatkan produktivitas dibandingkan proses manual tradisional di Adobe After Effects.

IV. HASIL DAN PEMBAHASAN

Penelitian ini berhasil mengembangkan sistem otomatisasi proses rendering video template berbasis web yang mengintegrasikan ExtendScript dan framework Flask. Sistem yang dikembangkan terdiri dari beberapa komponen utama: antarmuka web berbasis HTML, backend Flask dengan Python, dan scripting ExtendScript untuk kontrol Adobe After Effects.

A. Analisis Efisiensi Sistem

Pengujian menggunakan komputer os windows 10 yang menggunakan spesifikasi GTX 1650 super, software After Effects 2025 dan dilakukan sebanyak 5 kali menggunakan template yang sama dengan variasi parameter. Waktu yang dicatat adalah total waktu mulai dari input hingga file berhasil diproses. Tabel berikut menunjukkan data waktu (dalam detik) dari masing-masing metode:

Tabel 1. Data waktu *rendering*

No.	Waktu Manual (detik)	Waktu Otomatis (detik)
1	184	61
2	174	54
3	188	60
4	174	55
5	184	61
avg	180.80	58.20

Berdasarkan Tabel 1, rata-rata waktu proses rendering secara manual adalah 180.80 detik, sedangkan waktu rendering dengan sistem otomatis hanya membutuhkan 58.20 detik. Untuk mengukur seberapa besar peningkatan efisiensi yang diperoleh dari penggunaan sistem otomatisasi rendering dibandingkan dengan metode manual, digunakan rumus efisiensi sebagai berikut:

$$Ef = \left(\frac{waktu_m - waktu_o}{waktu_m}\right) \times 100\ \% \qquad (1)$$

di mana:

- Ef adalah nilai efisiensi dalam bentuk persentase (%),
- $waktu_m$ merupakan rata-rata waktu rendering menggunakan metode manual (dalam detik),
- $waktu_o$ merupakan rata-rata waktu rendering menggunakan sistem otomatisasi (dalam detik).

Rumus ini digunakan untuk mengetahui seberapa besar waktu yang berhasil dihemat dengan beralih dari proses manual ke proses otomatis. Semakin tinggi nilai Ef, semakin besar peningkatan efisiensi waktu yang diperoleh.

Berdasarkan data hasil percobaan Tabel 1, rata-rata waktu rendering secara manual adalah 180.80 detik, sedangkan dengan sistem otomatis hanya membutuhkan waktu 58.20 detik. Jika dimasukkan ke dalam rumus 1, maka efisiensi yang diperoleh sebesar:

$$Ef = \left(\frac{180.80 - 58.20}{180.80}\right) \times 100\ \% = 67{,}78\ \% \quad (2)$$

Hasil ini menunjukkan bahwa sistem otomatisasi memberikan peningkatan efisiensi waktu kerja hingga 67,78% dibandingkan metode rendering manual. Python flask dapat digunakan untuk membangun antarmuka backend berbasis web yang serupa [7].

B. Keunggulan dan Keterbatasan

Beberapa keunggulan yang ditawarkan oleh sistem ini antara lain adalah efisiensi waktu yang jauh lebih baik dibanding metode manual, kemudahan akses karena sistem berbasis web jika dikembangkan lebih lanjut, serta fleksibilitas tinggi dalam mendukung parameter kustomisasi seperti teks dan warna. Sistem ini juga memudahkan pengguna tanpa keahlian teknis dalam After Effects untuk tetap bisa menghasilkan video secara otomatis [8].

Namun demikian, sistem ini masih memiliki sejumlah keterbatasan [9]. Proses rendering sangat bergantung pada keberadaan dan stabilitas Adobe After Effects di sisi server, yang artinya sistem hanya dapat berjalan jika After Effects terpasang dan aktif. Selain itu, jumlah template yang tersedia masih terbatas dan dapat menjadi hambatan untuk variasi kebutuhan pengguna. Proses rendering juga tetap membutuhkan sumber daya perangkat keras yang tinggi, sehingga server perlu memiliki spesifikasi yang memadai untuk menjaga performa optimal saat banyak pengguna mengakses system [10].

V. KESIMPULAN

Penelitian ini berhasil merancang dan mengimplementasikan sistem otomatisasi proses rendering video berbasis web pada Adobe After Effects dengan mengintegrasikan *ExtendScript* dan framework Flask. Sistem yang dikembangkan terbukti dapat meningkatkan efisiensi proses rendering hingga 67,78% dibandingkan metode manual tradisional [11]. Sistem ini memungkinkan pembuat konten sosial media untuk mempercepat produksi konten visual tanpa mengorbankan kualitas dan fleksibilitas [12].

Integrasi teknologi web dengan software kreatif tradisional membuktikan bahwa otomatisasi proses kreatif dapat dilakukan secara efektif dan stabil [7].

Untuk pengembangan lebih lanjut, sistem dapat ditingkatkan dengan menambahkan dukungan untuk lebih banyak template, parameter kustomisasi yang lebih kompleks, dan integrasi dengan cloud storage untuk penyimpanan hasil render [13]. Selain itu, implementasi real-time monitoring dan notification system dapat meningkatkan pengalaman pengguna dalam memantau proses rendering [14].

Penelitian ini memberikan kontribusi signifikan dalam bidang otomatisasi produksi multimedia dan membuka peluang untuk pengembangan sistem serupa yang dapat diaplikasikan pada software kreatif. Dengan terus berkembangnya teknologi web dan kebutuhan akan efisiensi dalam industri kreatif, sistem seperti ini memiliki potensi besar untuk diadopsi secara luas di masa depan [15].

DAFTAR PUSTAKA

[1] M. S. Bonney *et al.*, "Development of a digital twin operational platform using Python Flask," *Data-Centric Engineering*, vol. 3, no. 1, Jan. 2022, doi: 10.1017/dce.2022.1.

[2] C. Wijayanto and Y. A. Susetyo, "IMPLEMENTASI FLASK FRAMEWORK PADA PEMBANGUNAN APLIKASI SISTEM INFORMASI HELPDESK (SIH)," Sep. 2022.

[3] R. K. Ngantung and M. A. I. Pakereng, "Model Pengembangan Sistem Informasi Akademik Berbasis Pengguna Centered Design Menerapkan Framework Flask Python," *JURNAL MEDIA INFORMATIKA BUDIDARMA*, vol. 5, no. 3, p. 1052, Jul. 2021, doi: 10.30865/mib.v5i3.3054.

[4] E. Susanti and E. Mailoa, "RESTful API Implementation in Making a Master Data Planogram Using the Flask Framework (Case Study: PT Sumber Alfaria Trijaya, Tbk)," 2020. [Online]. Available: www.jitecs.ub.ac.id

[5] R. Nandang Pratama and Y. A. Susetyo, "Implementasi Python API dengan Framework Flask sebagai Cloud Run Service Untuk Proses Update di PT. XYZ," 2024.

[6] J. Qiu, M. Chen, and G. Feng, "MBPPE: A Modular Batch Processing Platform for Electroencephalography," *Applied Sciences (Switzerland)*, vol. 14, no. 2, Jan. 2024, doi: 10.3390/app14020770.

[7] S. Larasati and Y. A. Susetyo, "Development of a Web-Based Trading Term Application Using Flask Framework at PT. XYZ," *International Journal Software Engineering and Computer Science (IJSECS)*, vol. 4, no. 1, pp. 367–376, Apr. 2024, doi: 10.35870/ijsecs.v4i1.2339.

[8] B. Ariyansa and N. Setiyawati, "PERANCANGAN DOMAIN SPECIFIC LANGUAGE PADA PEMBUATAN APLIKASI FRAMEWORK REPORTING DENGAN MENGGUNAKAN PYTHON FLASK," *JIPI (Jurnal Ilmiah Penelitian dan Pembelajaran Informatika)*, vol. 8, no. 4, pp. 1479–1492, Nov. 2023, doi: 10.29100/jipi.v8i4.4043.

[9] D. F. Ningtyas and N. Setiyawati, "Implementasi Flask Framework pada Pembangunan Aplikasi Purchasing Approval Request," *Jurnal Janitra Informatika dan Sistem Informasi*, vol. 1, no. 1, pp. 19–34, Apr. 2021, doi: 10.25008/janitra.v1i1.120.

[10] H. J. Mohammed and K. H. A. Faraj, "A Python-WSGI and PHP-Apache Web Server Performance Analysis by Search Page Generator (SPG)," *UKH Journal of Science and Engineering*, vol. 5, no. 1, pp. 132–138, Jun. 2021, doi: 10.25079/ukhjse.v5n1y2021.pp132-138.

[11] Y. Tamariska Bota and N. Setiyawati, "Pengembangan Sistem Informasi Perantara Bisnis Menggunakan Framework Flask," 2022. [Online]. Available: https://journal-computing.org/index.php/journal-ita/index

[12] Z. Liang, Z. Liang, Y. Zheng, B. Liang, and L. Zheng, "Data analysis and visualization platform design for batteries using flask-based python web service," *World Electric Vehicle Journal*, vol. 12, no. 4, Dec. 2021, doi: 10.3390/wevj12040187.

[13] A. Mateen, S. Y. Nam, M. A. Haider, and A. Hanan, "A dynamic decision support system for selection of cloud storage provider," *Applied Sciences (Switzerland)*, vol. 11, no. 23, Dec. 2021, doi: 10.3390/app112311296.

[14] P. M, A. B. J, and S. E.S, "Real-Time Web Server Monitoring System using Python," *Journal of Artificial Intelligence and Capsule Networks*, vol. 6, no. 3, pp. 332–339, Sep. 2024, doi: 10.36548/jaicn.2024.3.006.

[15] N. Idris, C. Feresa, M. Foozy, and P. Shamala, "A Generic Review of Web Technology: DJango and Flask," 2020.