Jurnal Teknik Informatika (JUTIF)
Vol. 5, No. 4, October 2024, pp. 1460-1478
DOI: https://doi.org/10.52436/1.jutif.2024.5.5.2585
p-ISSN: 2723-3863
e-ISSN: 2723-3871

WEB-BASED IMAGE CAPTIONING FOR IMAGES OF TOURIST ATTRACTIONS IN PURBALINGGA USING TRANSFORMER ARCHITECTURE AND TEXT-TO-SPEECH

Safa Muazam[*1], Yogiek Indra Kurniawan[2], Dadang Iskandar[3]

[1,2,3]Informatics, Engineering Faculty, Universitas Jenderal Soedirman, Indonesia
Email: [1]sfmuazam@gmail.com, [2]yogiek@unsoed.ac.id, [3]dadang.iskandar@unsoed.ac.id

(Article received: July 4, 2024; Revision: July 18, 2024; published: October 29, 2024)

Abstract

Purbalingga, a region in Central Java, offers natural beauty and attractive tourist destinations. Tourists often upload photos of their visits to social media. However, an image can contain a lot of information that may be interpreted differently by individuals. Without captions, people may find it difficult to extract the information. Image captioning addresses this challenge by generating automatic text descriptions. Meanwhile, text-to-speech enhances accessibility for visually impaired people in understanding the information. This research aims to develop an image captioning model for tourist images in Purbalingga using a transformer architecture and ResNet50. The transformer uses an attention mechanism to capture context, while ResNet50 is a convolutional neural network used for image feature extraction. The model was evaluated using the BLEU metric, achieving the best scores at BLEU-{1, 2, 3, 4} = {0.672, 0.559, 0.489, 0.437}. Experiments with various hyperparameters showed that increasing embedding size and layers extended training time but lowered BLEU scores, while changes in the number of heads had minimal effect. The best model was implemented in a web application using the SDLC waterfall method, Flask framework, and MySQL database. The application allows users to upload tourist images, generate automatic descriptions, and listen to them via text-to-speech. Blackbox testing confirmed the system's functionality as expected.

Keywords: *image captioning, text-to-speech, tourist attractions, transformer.*

IMAGE CAPTIONING PADA GAMBAR OBJEK WISATA DI PURBALINGGA MENGGUNAKAN ARSITEKTUR *TRANSFORMER* DAN *TEXT-TO-SPEECH* BERBASIS *WEBSITE*

Abstrak

Purbalingga, sebuah wilayah di Jawa Tengah, menawarkan keindahan alam dan tempat wisata yang menarik. Wisatawan sering mengunggah foto/gambar kunjungannya di media sosial. Namun, sebuah gambar dapat mengandung banyak informasi dan setiap individu dapat menafsirkannya secara berbeda. Tanpa adanya keterangan atau *caption*, manusia akan kesulitan mengurai informasi tersebut. *Image captioning* mampu mengatasi tantangan ini dengan menghasilkan deskripsi teks otomatis. Sementara itu, *text-to-speech* meningkatkan aksesibilitas bagi penyandang tunanetra dalam memahami informasi tersebut. Penelitian ini bertujuan mengembangkan model *image captioning* pada gambar objek wisata di Purbalingga menggunakan arsitektur *transformer* serta ResNet50. *Transformer* menggunakan *attention mechanism* untuk mempelajari konteks, sementara ResNet50 merupakan jaringan konvolusional untuk ekstraksi fitur gambar. Evaluasi model menggunakan metrik BLEU, dengan hasil terbaik BLEU-{1, 2, 3, 4} = {0.672, 0.559, 0.489, 0.437}. Berbagai eksperimen *hyperparameter* menunjukkan penambahan *embedding* dan *layer* meningkatkan waktu pelatihan namun menurunkan skor BLEU, sedangkan perubahan jumlah *head* tidak terlalu memengaruhi hasil. Model terbaik diimplementasikan dalam aplikasi web menggunakan metode SDLC *waterfall*, *framework Flask*, dan basis data *MySQL*. Aplikasi memungkinkan pengguna mengunggah gambar objek wisata, mendapatkan deskripsi otomatis, dan mendengarkan deskripsi tersebut. Pengujian *blackbox* menunjukkan hasil valid sesuai harapan.

Kata kunci: *image captioning, objek wisata, text-to-speech, transformer.*

1. PENDAHULUAN

Purbalingga merupakan sebuah wilayah kabupaten di Provinsi Jawa Tengah yang memiliki keindahan alam serta kekayaan budaya yang menarik. Banyak wisatawan yang mengabadikan momen kunjungannya di Purbalingga melalui foto

atau gambar yang kemudian diunggah di media sosial. Namun, tanpa adanya keterangan atau *caption* yang menyertai gambar-gambar tersebut, banyak informasi yang terkandung dalam gambar sulit untuk dipahami secara maksimal. Hal ini dikarenakan setiap individu dapat menafsirkan gambar secara berbeda, sehingga diperlukan teknologi yang dapat membantu memberikan deskripsi yang lebih jelas dan konsisten.

Salah satu solusi untuk mengatasi tantangan tersebut adalah dengan memanfaatkan teknologi kecerdasan buatan, khususnya dalam bidang *image captioning*. *Image captioning* adalah bidang teknologi yang berfokus pada pemahaman gambar dan deskripsi bahasa suatu gambar[1]. *Image captioning* menggunakan pendekatan *encoder-decoder* untuk menghasilkan deskripsi suatu gambar secara otomatis. Di sisi *encoder*, arsitektur *Convolutional Neural Network* (CNN) digunakan untuk mengekstraksi fitur visual dari gambar[2]. Berbagai arsitektur CNN telah diimplementasikan dalam penelitian sebelumnya, seperti VGG[3], InceptionV3[4], dan ResNet[5]. Di sisi *decoder*, pemodelan bahasa berbasis *Recurrent Neural Network* (RNN) seperti *Long Short-Term Memory* (LSTM)[5] dan *Gated Recurrent Unit* (GRU)[6] sering digunakan untuk menghasilkan *caption* gambar secara sekuensial, kata demi kata, berdasarkan fitur yang diekstraksi oleh CNN.

Penelitian lebih lanjut mengenai *image captioning* menunjukkan bahwa implementasi arsitektur *transformer* dapat menggantikan peran RNN, dengan hasil yang lebih unggul dari LSTM dalam menghasilkan deskripsi teks[7]. Arsitektur *transformer* menggunakan *attention mechanism* yang lebih efektif dalam mempelajari hubungan konteks antara *input* dan *output*, sehingga memberikan keterangan yang lebih akurat dan relevan. Penelitian oleh[7] membuktikan bahwa *transformer* lebih unggul dibandingkan model RNN seperti LSTM, khususnya dalam mengolah data urutan panjang dan kompleks.

Penggunaan *ResNet50* sebagai *encoder* telah terbukti memberikan hasil evaluasi yang lebih baik dibandingkan arsitektur CNN lainnya dalam tugas *image captioning*[8]. Dengan memanfaatkan kombinasi *ResNet50* sebagai *encoder* dan *transformer* sebagai *decoder*, model yang dihasilkan mampu memberikan deskripsi teks otomatis yang lebih akurat dari gambar objek wisata di Purbalingga.

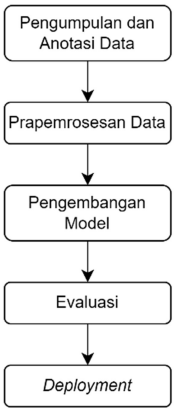
Model *image captioning* tersebut diimplementasikan dalam aplikasi web untuk memudahkan pengguna dalam melakukan tugas *image captioning* objek wisata di Purbalingga. Implementasi berupa *website* dilakukan karena aplikasi berbasis *web* dapat dijalankan pada berbagai perangkat asalkan *browser*-nya kompatibel dan juga tidak perlu dilakukan instalasi aplikasi terlebih dahulu[9].

Meskipun *image captioning* mampu menghasilkan deskripsi teks secara otomatis, penyandang tunanetra tidak dapat membaca teks tersebut secara langsung. Oleh karena itu, diperlukan solusi untuk memberikan aksesibilitas yang lebih baik bagi penyandang tunanetra. *Text-to-speech* menjadi salah satu solusi untuk meningkatkan aksesibilitas dan kemudahan penggunaan. Fitur *text-to-speech* memungkinkan deskripsi gambar yang dihasilkan oleh model *image captioning* untuk dibacakan secara audio, sehingga dapat membantu pengguna yang memiliki keterbatasan penglihatan atau yang ingin mendapatkan informasi secara lebih mudah tanpa harus membaca teks. Teknologi yang bisa digunakan untuk memberikan fitur *text-to-speech* adalah *Web Speech* API. *Web Speech* API merupakan *Application Programming Interface* (API) yang memungkinkan untuk memanipulasi data suara dalam suatu aplikasi web termasuk *text-to-speech*. *Web Speech* API dibangun dengan bahasa *JavaScript* sehingga memudahkan pengguna untuk mengembangkan aplikasi web lebih interaktif khususnya pada perangkat yang memiliki akses terhadap komponen audio[10].

Berdasarkan latar belakang tersebut, penelitian ini bertujuan untuk mengembangkan aplikasi berbasis *web* yang memanfaatkan *image captioning* dengan arsitektur *ResNet50* dan *transformer*, serta dilengkapi dengan fitur *text-to-speech*. Aplikasi memungkinkan pengguna mengunggah gambar objek wisata, mendapatkan deskripsi otomatis dalam bahasa Indonesia, dan mendengarkan deskripsi tersebut. Diharapkan aplikasi ini dapat memberikan manfaat dalam menguraikan deskripsi gambar objek wisata di Purbalingga, serta meningkatkan aksesibilitas bagi penyandang tunanetra.

2. METODE PENELITIAN

Metode yang digunakan dalam penelitian ini ditunjukkan pada gambar 1. Metode dilakukan secara bertahap dan berurutan.



Gambar 1. Metode Penelitian

2.1. Pengumpulan dan Anotasi Data

Data dikumpulkan dari lima objek wisata di Purbalingga, yakni D'Las, Goa Lawa, Owabong, Purbasari, dan Sanggaluri, dengan total sekitar 250 gambar. Setiap gambar dianotasi secara manual dengan tiga keterangan berbeda, sehingga total terdapat sekitar 750 *caption*. *Caption* mempertimbangkan aspek objek, aktivitas, suasana, dan nama objek wisata dalam gambar.

2.2. Prapemrosesan Data

Setelah pengumpulan dan anotasi data, dilakukan prapemrosesan data untuk mendapatkan dataset dengan kualitas baik[11]. Prapemrosesan ini mencakup augmentasi gambar, prapemrosesan *caption*, prapemrosesan gambar, dan pembagian data.

2.2.1. Augmentasi Gambar

Tahapan pertama dalam prapemrosesan data adalah augmentasi gambar. Augmentasi dilakukan dengan melakukan transformasi pada gambar seperti rotasi, perubahan kecerahan, dan kontras, untuk memperbanyak variasi data. *Caption* yang diberikan pada gambar augmentasi sama dengan *caption* pada gambar asli.

2.2.2. Prapemrosesan *Caption*

Pada tahapan ini, dataset dikenakan prapemrosesan *caption*. Proses ini dilakukan dengan beberapa tahapan antara lain, pembersihan teks, penambahan token di awal dan akhir kalimat, pembuatan kamus kosa kata, serta mengubah teks menjadi urutan indeks.

2.2.3. Prapemrosesan Gambar

Pada tahapan ini, dataset dikenakan prapemrosesan gambar. Proses ini dilakukan dengan beberapa tahapan antara lain, mengubah gambar menjadi tensor, menyesuaikan ukuran menjadi 224x224 piksel, dan normalisasi nilai piksel.

2.2.4. Pembagian Data

Setelah prapemrosesan data selesai, dataset dibagi menjadi dua bagian, yaitu data *training* dan data *testing*. Data *training* digunakan sebagai data rujukan dalam perhitungan setiap algoritma, sedangkan data *testing* digunakan untuk menilai penentuan yang dilakukan oleh model sudah tepat atau tidak. Rasio pembagian data adalah 80:20 untuk data *training* dan data *testing* dengan proporsi yang sama antara tiap kategori objek wisata.

2.3. Pengembangan Model

Pengembangan model dilakukan dengan bahasa *Python* menggunakan *tool Kaggle Notebook*. *Python* dipilih karena mudah dipelajari dan memiliki *library* yang mendukung untuk pengembangan model[12]. Arsitektur model yang dikembangkan adalah *transformer*, yang terdiri dari *encoder* dan *decoder*. Lapisan *encoder* akan memproses gambar dan mengekstraksi fitur, sedangkan *decoder* akan menghasilkan deskripsi teks berdasarkan fitur tersebut. Model akan dilatih menggunakan dataset yang telah diproses sebelumnya.

2.3.1. Ekstrasi Fitur Gambar dengan ResNet50

Ekstraksi fitur merupakan langkah awal dalam pengembangan model *image captioning*. Untuk tahap ini, arsitektur *Residual Networks 50* (ResNet50) digunakan. Tabel 1 menunjukkan lapisan pada arsitektur ResNet50.

Tabel 1. Arsitektur ResNet50

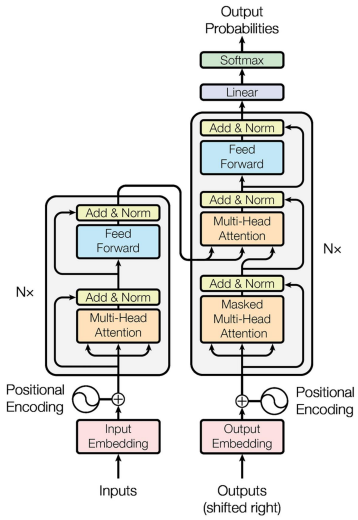
Layer	*Output Size*	*Layer*
Conv1	112×112	$7 \times 7, 64, stride\ 2$
		$3 \times 3 \max pool, stride\ 2$
Conv2	56×56	$\begin{bmatrix} 1 \times 1, 64 \\ 3 \times 3, 64 \\ 1 \times 1, 256 \end{bmatrix} \times 3$
Conv3	28×28	$\begin{bmatrix} 1 \times 1, 128 \\ 3 \times 3, 128 \\ 1 \times 1, 512 \end{bmatrix} \times 4$
Conv4	14×14	$\begin{bmatrix} 1 \times 1, 256 \\ 3 \times 3, 256 \\ 1 \times 1, 1024 \end{bmatrix} \times 6$
Conv5	7×7	$\begin{bmatrix} 1 \times 1, 512 \\ 3 \times 3, 512 \\ 1 \times 1, 2048 \end{bmatrix} \times 3$
	1×1	*Average pool, fully connected layer, softmax*

ResNet50 menggunakan blok *residual* yang mempelajari perbedaan (*residual*) antara *input* dan *output* yang diharapkan, sehingga membuat pelatihan jaringan yang sangat dalam menjadi lebih efisien dan dapat mengurangi masalah *vanishing gradient*.

Sebagai *encoder*, lapisan klasifikasi terakhir dari ResNet50 dihapus sehingga hanya lapisan konvolusi yang digunakan untuk mengekstraksi fitur gambar objek wisata. Fitur-fitur yang dihasilkan dari proses ini selanjutnya digunakan sebagai masukan untuk model *transformer*.

2.3.2. *Transformer*

Transformer adalah arsitektur yang didesain untuk mengatasi keterbatasan dalam model berbasis rekursi dan konvolusi dengan mengandalkan mekanisme *self-attention*. Pendekatan ini memungkinkan model untuk menangani berbagai modalitas, seperti teks, gambar, video, dan audio, dengan cara yang lebih efisien. Gambar 2 merupakan arsitektur *transformer*.



Gambar 2. Arsitektur *Transformer*

Arsitektur *transformer* terdiri dari dua komponen utama yaitu *encoder* dan *decoder*. *Encoder* dan *decoder* masing-masing terdiri dari tumpukan lapisan identik yang berjumlah N. Setiap lapisan *encoder* memiliki dua sub-lapisan: mekanisme *multi-head attention* dan *feed-forward network* berbasis posisi. Lapisan *encoder* juga dilengkapi dengan koneksi *residual* di antara sub-lapisan, diikuti oleh normalisasi lapisan.

Pada *decoder*, terdapat tiga sub-lapisan, yang meliputi mekanisme *multi-head attention*, yang serupa dengan *encoder* dengan tambahan *masking* untuk mencegah model melihat posisi masa depan, *encoder-decoder attention*, dan *feed-forward network*.

Perhitungan *attention* ditunjukkan pada persamaan (1).

$$Attention(Q,K,V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \quad (1)$$

di mana:

- Q adalah *Query*,
- K adalah *Key*,
- V adalah *Value*,
- K^T adalah transpose dari *Key*,
- d_k adalah dimensi *Key*,

Multi-head attention adalah mekanisme yang memperluas kemampuan *attention* dengan menerapkan beberapa *heads* secara paralel. Setiap *head* memproyeksikan *query*, *key*, dan *value* dengan matriks bobot yang berbeda, kemudian menerapkan fungsi *attention* untuk masing-masing proyeksi. Hasil dari semua *heads* digabungkan (*concat*) dan diproyeksikan kembali. Perhitungan untuk *multi-head attention* ditunjukkan pada persamaan (2) dan (3).

$$multihead(Q,K,V) = concat(head_1,\dots,head_n)W^O \quad (2)$$

$$head_i = Attention(QW_i^Q, KW_i^K, VW_i^V) \quad (3)$$

di mana:

- Q, K, V adalah *query*, *key*, dan *value*,
- W_i^Q, W_i^K, W_i^V adalah matriks bobot untuk *query*, *key*, dan *value* di head ke- i,
- W^O adalah matriks bobot untuk *output* dari *concat*.

Karena model *transformer* tidak memiliki unit berulang atau konvolusi, *positional encoding* ditambahkan untuk memberikan informasi tentang posisi token dalam urutan. *Positional encoding* memiliki dimensi yang sama dengan *embedding*, dan dihitung menggunakan fungsi sinus dan kosinus.

Setiap lapisan dalam *encoder* dan *decoder* juga mencakup *feed-forward network* yang sepenuhnya terhubung, yang diterapkan secara terpisah dan identik pada setiap posisi. Jaringan ini terdiri dari dua transformasi linear dengan fungsi aktivasi ReLU di antara keduanya.

2.4. Pelatihan Model

Pelatihan model ini melibatkan komponen seperti *loss function*, *optimizer*, arsitektur pelatihan, dan *hyperparameter*. *Sparse categorical cross-entropy* digunakan sebagai *loss function* untuk mengukur perbedaan antara distribusi prediksi dan target, dengan *masking* untuk mengabaikan token *padding*. *Adam optimizer* diterapkan dengan *learning rate* dinamis yang berubah selama pelatihan. Pelatihan dilakukan di *Kaggle Notebook* menggunakan GPU Tesla T4 dengan *library TensorFlow* dan *Keras*. Nilai *hyperparameter*s untuk pelatihan model secara detail ditunjukkan oleh Tabel 2.

Tabel 2. Nilai *Hyperparameter*

Hyperparameters	**Variabel**	**Nilai**
Dimensi *embedding*	d_model	512
Dimensi *feed forward*	dff	2048
Jumlah *layer*	num_layer	1
Jumlah *head*	num_head	4
Jumlah kosa kata	vocab_size	1733
Ukuran *batch*	batch_size	16
Dropout rate	dropout_rate	0,1
Panjang sekuens	seq_len	108
Epochs	num_epochs	35

Pada penelitian ini dilakukan eksperimen untuk mendapatkan model dengan hasil yang terbaik dengan *hyperparameters tuning*. *Hyperparameter* yang diubah adalah dimensi *embedding*, jumlah *layer*, dan jumlah *head* dengan kombinasi berbagai nilai yang ditunjukkan pada Tabel 3.

Tabel 3. Nilai *Hyperparameter Tuning*

Hyperparameters	**Variabel**	**Nilai**
Dimensi *embedding*	d_model	64, 128, 512, 1024, 2048
Jumlah *layer*	num_layer	1, 2, 4, 6, 8
Jumlah *head*	num_head	1, 2, 4, 8, 16

Dengan menguji berbagai *hyperparameter* yang berbeda pada proses pelatihan, diharapkan dapat ditemukan konfigurasi terbaik dari arsitektur *transformer* untuk menghasilkan *caption* yang akurat dan relevan dari gambar wisata yang ada. Model yang telah dilatih kemudian disimpan dalam format h5 bersama *tokenizer* dalam format *pickle* untuk digunakan pada proses selanjutnya.

2.5. Evaluasi

Evaluasi model dilakukan dengan mengukur skor BLEU (*Bilingual Evaluation Understudy*) untuk menguji kualitas *caption* yang dihasilkan. Skor BLEU yang diuji meliputi BLEU-1, BLEU-2, BLEU-3, dan BLEU-4, yang masing-masing mengukur n-gram *precision* dengan berbagai panjang n. Perhitungan skor BLEU merujuk ke persamaan (4), (5), dan (6).

$$P_n = \frac{\sum_{c\epsilon\{candidates\}} \sum_{n-gra} \epsilon_c count(n-gra)}{\sum_{c'\epsilon\{candidates\}} \sum_{n-gra} \epsilon_c count(n-gra)} \quad (4)$$

Berdasarkan persamaan yang ditunjukkan oleh persamaan (5) dan (6), c merupakan jumlah kata dalam terjemahan referensi, r merupakan jumlah kata dari terjemahan kandidat, p_n merupakan *modified n-gram precision score*, dan *Brevity Penalty* (BP) adalah pinalti untuk mendukung translasi yang terlalu pendek.

$$BP_{BLEU} = \begin{cases} 1 & , c > r \\ exp(1 - \frac{r}{c}), & c \leq r \end{cases} \quad (5)$$

maka, $BLEU = BP.exp\left(\sum_{n=1}^{N} w_n log\ p_n\right)$ (6)

BLEU memiliki skala antara 0 hingga 1, di mana semakin mendekati 1, semakin mirip dengan *caption* referensi maka semakin baik modelnya. Model dievaluasi menggunakan *data testing* dengan menghasilkan *caption* untuk setiap gambar dan membandingkannya dengan *caption* referensi.

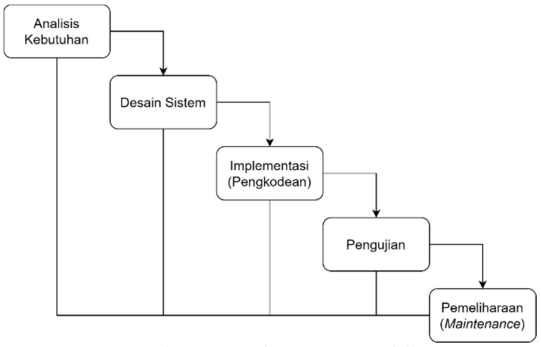
Berbagai kombinasi nilai *hyperparameter* diuji, untuk mendapatkan skor BLEU terbaik. Kemudian dilakukan eksperimen lebih lanjut dengan menguji perubahan *hyperparameter* seperti dimensi *embedding*, jumlah *layer*, dan jumlah *head,* untuk mengamati pengaruhnya terhadap performa model. Percobaan ini dilakukan dengan mempertahankan dua *hyperparameter* tetap dan mengubah nilai satu *hyperparameter*.

2.6. *Deployment*

Setelah melalui tahap evaluasi, model terbaik diimplementasikan menjadi sebuah aplikasi web menggunakan *framework Flask*. *Flask* adalah sebuah *microframework* web berbasis *Python*. *Flask* berfungsi sebagai kerangka kerja aplikasi dan tampilan dari suatu web, memungkinkan pengembang untuk membuat web yang terstruktur dan mengatur perilaku web dengan lebih mudah[13].

Aplikasi yang dikembangkan memungkinkan pengguna untuk mengunggah gambar objek wisata di Purbalingga, yang kemudian dianalisis oleh model *image captioning* untuk menghasilkan deskripsi teks. Selain itu, aplikasi dilengkapi dengan fitur *text-to-speech*, di mana deskripsi teks ini akan dibacakan secara otomatis melalui *Web Speech API* yang diimplementasikan dengan *JavaScript*.

Pengembangan aplikasi ini mengikuti metodologi SDLC (*Software Development Life Cycle*) dengan metode *waterfall*, yang terdiri dari tahapan analisis, desain, implementasi, dan pengujian. Metode *waterfall* dipilih karena pendekatannya yang sistematis dan berurutan, di mana setiap tahap harus diselesaikan sebelum melanjutkan ke tahap berikutnya, mirip dengan aliran air terjun dalam pengembangan perangkat lunak[12]. Tahapan SDLC *waterfall* ditunjukkan pada Gambar 3.



Gambar 3. Metode SDLC *Waterfall*

2.6.1. Analisis

Tahap analisis dilakukan untuk menentukan kebutuhan atau *requirement* pengguna. Tahap ini mengidentifikasi permasalahan dan kebutuhan terkait sistem informasi yang akan dikembangkan[14].

2.6.2. Desain

Setelah diperoleh kebutuhan pengguna dan sistem, selanjutnya dilakukan perancangan atau desain sistem. Rancangan yang dibuat menggunakan *Unified Modeling Language* (UML), antara lain *use case diagram*, *sequence diagram*, *Entity Relationship Diagram* (ERD), serta *class diagram*.

Use case diagram menggambarkan interaksi antara satu atau lebih aktor dengan suatu sistem yang akan dibangun[15]. *Entity Relationship Diagram* (ERD) merupakan suatu diagram yang mengidentifikasi tipe dari entitas di dalam suatu sistem yang diuraikan dalam data dengan atributnya, dan menjelaskan hubungan atau relasi diantara entitas tersebut[16]. *Class diagram* bertujuan untuk memvisualisasikan kelas yang akan dibuat saat implementasi berlangsung.

2.6.3. Implementasi

Pada tahap ini dilakukan implementasi desain atau rancangan sistem yang sudah dibuat ke dalam kode program. Sistem dikembangkan menggunakan *framework Flask* yang berbasis *Python* pada sisi server. Sedangkan pada sisi klien menggunakan HTML, CSS, dan *JavaScript* dengan *template engine Jinja2*. *Database* yang digunakan adalah *MySQL*.

2.6.4. Pengujian

Tahap pengujian bertujuan untuk mengetahui kualitas dari sistem[17]. Pengujian dilakukan menggunakan metode pengujian *blackbox*. Pengujian *blackbox* bertujuan untuk menguji apakah aplikasi berjalan sesuai dengan yang diharapkan, yaitu dengan melakukan uji dengan beberapa *test case*[18].

3. HASIL DAN PEMBAHASAN

Pada bagian ini dapat diuraikan mengenai hasil dan pembahasan dari penelitian yang dilakukan.

3.1. Pengumpulan dan Anotasi Data

Dalam penelitian ini, 258 gambar telah dikumpulkan dari lima objek wisata, yaitu D'Las, Goa Lawa, Owabong, Purbasari, dan Sanggaluri. Persebaran jumlah gambar untuk masing-masing objek wisata ditunjukkan pada Tabel 4.

Tabel 4. Jumlah Gambar Setiap Objek Wisata

Wisata	Jumlah Gambar
D'Las	55
Goa Lawa	50
Owabong	51
Purbasari	50
Sanggaluri	52
Total	**258**

Setiap gambar dianotasi dengan tiga *caption* berbeda, menghasilkan total 774 *captions*. Proses anotasi ini menggunakan aplikasi Notepad, di mana setiap baris berisi informasi nama file gambar, indeks *caption*, dan *caption* itu sendiri. Anotasi ini kemudian disimpan dalam file teks (.txt). Tabel 5 menyajikan contoh gambar tempat wisata beserta anotasinya.

Tabel 5. Anotasi pada Gambar

Gambar	*Caption*
 dlas38.jpg	dlas38.jpg#0#perempuan sedang memotret pria yang sedang menggendong anaknya di depan replika triceratops dinosaurus bertanduk dengan pohon dan rerumputan di sekitarnya di dlas dino land
	dlas38.jpg#1#suasana di Dlas Dino Land dengan fokus pada patung Triceratops yang besar. Triceratops tersebut memiliki tiga tanduk yang menonjol dan kulit dengan tekstur yang tampak nyata. Di sebelah patung, terdapat papan informasi yang memberikan penjelasan tentang Triceratops. Di depan patung, ada seorang pria yang menggendong anak kecil dengan topi biru dan pakaian bergaris-garis, sementara seorang wanita berjilbab hitam dengan baju merah muda sedang memotret mereka. Latar belakangnya adalah hutan yang rimbun dengan tanaman hijau yang subur, serta ada rantai putih yang mengelilingi patung sebagai pembatas.
	dlas38.jpg#2#di Dlas Dino Land, menampilkan patung dinosaurus jenis Triceratops. Dinosaurus tersebut terlihat besar dengan tiga tanduk di kepalanya dan tekstur kulit yang tampak kasar dan bersisik. Di sebelah kanan patung, terdapat papan informasi mengenai dinosaurus tersebut. Di depan patung, seorang pria sedang menggendong anak kecil yang mengenakan topi biru dan baju garis-garis. Seorang wanita yang mengenakan jilbab hitam dan baju garis-garis merah muda sedang mengambil foto mereka dengan ponsel. Latar belakang menunjukkan pepohonan hijau yang rimbun, dan rantai putih mengelilingi area patung dinosaurus untuk pembatas

3.2. Prapemrosesan Data

Setelah data dikumpulkan dan dianotasi, dilakukan prapemrosesan data. Prapemrosesan ini mencakup augmentasi gambar, prapemrosesan *caption*, prapemrosesan gambar, dan pembagian data.

3.2.1. Augmentasi Gambar

Pada tahap ini, dilakukan transformasi pada gambar untuk meningkatkan variasi data. Transformasi ini meliputi rotasi dengan sudut antara -10 hingga 10 derajat, perubahan kecerahan dengan faktor antara 0.8 dan 1.2, serta perubahan kontras dengan faktor antara 0.75 dan 1.5. Setiap gambar mengalami augmentasi sebanyak empat kali sehingga total gambar menjadi 1.290 gambar dengan total 3.870 *caption*. Tabel 6 menunjukkan contoh gambar asli dan hasil augmentasinya.

Tabel 6. Augmentasi Gambar

Gambar Asli	Hasil Augmentasi
 dlas38.jpg	
 golaga33.jpg	

3.2.2. Prapemrosesan *Caption*

Pada tahapan ini, dataset dikenakan prapemrosesan *caption*. Proses ini dilakukan dengan beberapa tahapan sebagai berikut.

a. Pembersihan teks

Prapemrosesan *caption* diawali dengan pembersihan teks. Langkah-langkah yang dilakukan meliputi mengubah semua kalimat menjadi berhuruf kecil, menghapus tanda baca, menghapus karakter tunggal, dan menghapus nilai numerik. Tabel 7 menunjukkan *caption* awal dan *caption* sesudah dilakukan pembersihan teks.

Tabel 7. Pembersihan Teks pada *Caption*

***Caption* Awal**	***Caption* Sesudah**
di Dlas Dino Land, menampilkan patung dinosaurus jenis Triceratops. Dinosaurus tersebut terlihat besar dengan tiga tanduk di kepalanya dan tekstur kulit yang tampak kasar dan bersisik. Di sebelah kanan patung, terdapat papan informasi mengenai dinosaurus tersebut. Di depan patung, seorang pria sedang menggendong anak kecil yang mengenakan topi biru dan baju garis-garis. Seorang wanita yang mengenakan jilbab hitam dan baju garis-garis merah muda sedang mengambil foto mereka dengan ponsel. Latar belakang menunjukkan pepohonan hijau yang rimbun, dan rantai putih mengelilingi area patung dinosaurus untuk pembatas	di dlas dino land menampilkan patung dinosaurus jenis triceratops dinosaurus tersebut terlihat besar dengan tiga tanduk di kepalanya dan tekstur kulit yang tampak kasar dan bersisik di sebelah kanan patung terdapat papan informasi mengenai dinosaurus tersebut di depan patung seorang pria sedang menggendong anak kecil yang mengenakan topi biru dan baju garis garis seorang wanita yang mengenakan jilbab hitam dan baju garis garis merah muda sedang mengambil foto mereka dengan ponsel latar belakang menunjukkan pepohonan hijau yang rimbun dan rantai putih mengelilingi area patung dinosaurus untuk pembatas

b. Pemberian token di awal dan akhir kalimat

Setiap *caption* yang telah dibersihkan kemudian diberikan token <start> di awal dan token <end> akhir kalimat untuk membantu model memahami batasan setiap kalimat. Tabel 8 menunjukkan *caption* sesudah ditambahkan token.

Tabel 8. Pemberian Token pada *Caption*

***Caption* Sesudah Ditambahkan Token**
<start> di dlas dino land menampilkan patung dinosaurus jenis triceratops dinosaurus tersebut terlihat besar dengan tiga tanduk di kepalanya dan tekstur kulit yang tampak kasar dan bersisik di sebelah kanan patung terdapat papan informasi mengenai dinosaurus tersebut di depan patung seorang pria sedang menggendong anak kecil yang mengenakan topi biru dan baju garis garis seorang wanita yang mengenakan jilbab hitam dan baju garis garis merah muda sedang mengambil foto mereka dengan ponsel latar belakang menunjukkan pepohonan hijau yang rimbun dan rantai putih mengelilingi area patung dinosaurus untuk pembatas <end>

c. Pembuatan kamus kosa kata

Selanjutnya, dilakukan pembuatan kamus kosa kata yang berisi semua kata unik yang ditemukan dari dataset *caption* beserta indeksnya. Tabel 9 menunjukkan kosa kata beserta indeksnya.

Tabel 9. Kamus Kosa Kata Beserta Indeksnya

Kata	**Indeks**	**Kata**	**Indeks**
<pad>	0	…..	…..
<unk>	1	struktural	1728
yang	2	menyajikan	1729
di	3	stalakmit	1730
dan	4	dibangun	1731
dengan	5	mengganggu	1732
<start>	6	diperkaya	1733
<end>	7	pelajar	1734
…..	…..	turun	1735

d. Representasi teks ke dalam urutan indeks

Setelah kamus kosa kata dibuat, setiap *caption* diubah menjadi urutan indeks sesuai dengan kamus tersebut. Hal ini memungkinkan model untuk memproses data teks secara efisien selama pelatihan. Tabel 10 menunjukkan contoh representasi *caption* dalam urutan indeks.

Tabel 10. Representasi *Caption* dalam Urutan Indeks

Caption	**Representasi**
<start> di dlas dino land menampilkan patung dinosaurus jenis triceratops dinosaurus tersebut terlihat besar dengan tiga tanduk di kepalanya dan tekstur kulit yang tampak kasar dan bersisik di sebelah kanan patung terdapat papan informasi mengenai dinosaurus tersebut di depan patung seorang pria sedang menggendong anak kecil yang mengenakan topi biru dan baju garis garis seorang wanita yang mengenakan jilbab hitam dan baju garis garis merah muda sedang mengambil foto mereka dengan ponsel latar belakang menunjukkan pepohonan hijau yang rimbun dan rantai putih mengelilingi area patung dinosaurus untuk pembatas <end>	[6, 3, 66, 140, 141, 106, 28, 127, 172, 383, 127, 25, 9, 15, 5, 409, 497, 3, 893, 4, 359, 179, 2, 38, 193, 4, 656, 3, 99, 109, 28, 17, 62, 113, 398, 127, 25, 3, 42, 28, 48, 63, 27, 735, 32, 72, 2, 70, 498, 18, 4, 421, 255, 255, 48, 139, 2, 70, 459, 73, 4, 421, 255, 255, 19, 123, 27, 380, 267, 75, 5, 414, 33, 31, 132, 82, 12, 2, 215, 4, 449, 30, 150, 20, 28, 127, 52, 686, 7, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]

3.2.3. Prapemrosesan Gambar

Pada tahapan ini, dataset dikenakan prapemrosesan gambar. Proses ini dilakukan dengan beberapa tahapan sebagai berikut.

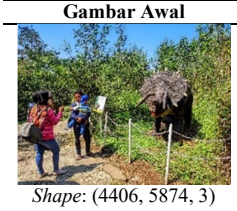
a. Mengubah gambar menjadi tensor

Gambar yang dikumpulkan diubah menjadi tensor untuk memudahkan pemrosesan lebih lanjut dalam model *transformer*.

b. Mengubah ukuran gambar

Selanjutnya. gambar diubah ukurannya menjadi 224x224 piksel agar sesuai dengan *input layer* model ResNet50. Tabel 11 menunjukkan gambar awal dan gambar setelah diubah ukurannya.

Tabel 11. Perubahan Ukuran Gambar

Gambar Awal	**Gambar Sesudah *Resize***
 *Shape*: (4406, 5874, 3) *dtype*: uint8	 *Shape*: (224, 224, 3) *dtype*: uint8

c. Normalisasi gambar

Selanjutnya, gambar dinormalisasi dengan mengubah nilai pikselnya ke rentang -1 hingga 1. Tipe data gambar juga diubah dari uint8 menjadi float32. Tabel 12 menunjukkan piksel awal dan juga piksel sesudah normalisasi.

Tabel 12. Normalisasi Gambar

Piksel Awal	Piksel Sesudah Normalisasi
[[[20 39 88] [121 182 252] [10 45 105] ... [101 169 252] [102 168 255] [101 169 254]] [[2 53 155] [125 189 255] [112 166 253] ... [104 172 255] [103 170 251] [45 85 81]]] ... [[[96 105 110] [102 111 116] [107 116 121] ... [60 37 22] [132 101 72] [101 73 59]] [[110 119 124] [2 4 10] [83 92 97] ... [50 37 0] [148 121 73] [84 66 42]]]	[[[-0.8454586 -0.69076884 -0.3119359] [-0.04715967 0.43101895 0.9748677] [-0.9209735 -0.65086704 -0.1771074] ... [-0.20542711 0.3275962 0.9752878] [-0.20000005 0.31764698 0.99999976] [-0.20693094 0.3264023 0.99306905]] [[-0.98562586 -0.58361495 0.2163415] [-0.01892859 0.48262072 1.0000002] [-0.12175274 0.30300987 0.98160374] ... [-0.18269646 0.34948158 0.999177] [-0.18977636 0.33571386 0.9710078] [-0.64890194 -0.33520377 -0.36238986]]] ... [[[-0.24870813 -0.1781199 -0.13890415] [-0.1976924 -0.12710422 -0.08788848] [-0.1612311 -0.09064281 -0.05142725] ... [-0.53028065 -0.7067081 -0.8311036] [0.0328536 -0.2095772 -0.43732202] [-0.20922577 -0.42965037 -0.5376769]] [[-0.13896751 -0.06837934 -0.0291636] [-0.98346156 -0.9654037 -0.9240715] [-0.34584773 -0.27525955 -0.23604387] ... [-0.60696125 -0.71066165 -0.9971784] [0.15902817 -0.05438316 -0.42975867] [-0.34239602 -0.48140788 -0.67182714]]]
Shape: (224, 224, 3) *dtype*: uint8	*Shape*: (224, 224, 3) *dtype*: float32

3.2.4. Pembagian Data

Dataset dibagi menjadi dua bagian, yaitu data *training* dan data *testing*. Data dibagi dengan proporsi yang sama antara kategori objek wisata pada kedua set data tersebut. Rasio pembagian data adalah 80:20 untuk data *training* dan data *testing*. Tabel 13 menunjukkan jumlah pembagian data antara data *training* dan data *testing*.

Tabel 13. Pembagian Data

Tipe Data	Jumlah Gambar	Jumlah *Caption*
Data *training*	1032	3096
Data *testing*	258	774
Total	1290	3870

3.3. Pengembangan Model

Pengembangan model dimulai dengan ekstraksi fitur dari gambar yang telah dipraproses menggunakan ResNet50 dengan bobot dataset ImageNet. Lapisan klasifikasi terakhir dari ResNet50 dihilangkan untuk memperoleh fitur yang kemudian diubah menjadi representasi tensor dengan dimensi yang sesuai untuk model *transformer* menggunakan *embedding layer*.

Langkah selanjutnya adalah meneruskan tensor fitur tersebut ke lapisan *encoder transformer*. Proses dalam lapisan *encoder* meliputi *embedding*, penambahan *positional encoding*, M*ulti-Head Attention* (MHA), dan *Feed Forward Network* (FFN). Tensor kemudian diteruskan ke lapisan *encoder* selanjutnya hingga mencapai lapisan *encoder* terakhir, dan kemudian diteruskan ke *decoder transformer*.

Pada tahap *decoder*, tensor diteruskan melalui *embedding*, *positional encoding*, *masked multi-head attention*, *encoder-decoder attention*, *dan feed forward network* (FFN) secara berulang hingga mencapai lapisan *decoder* terakhir. Setelah melalui semua lapisan *decoder*, tensor diteruskan ke lapisan final yang terdiri dari *linear layer* dan *softmax* untuk mengubah skor menjadi distribusi probabilitas atas seluruh kata dalam target kamus. Tensor hasil akhir berupa logit distribusi probabilitas yang digunakan untuk acuan memilih kata dengan probabilitas tertinggi sebagai prediksi teks dari model.

3.4. Pelatihan Model

Setelah melakukan pengembangan model, langkah berikutnya adalah melatih model dengan konfigurasi dan arsitektur yang ditentukan. Pelatihan model dilakukan selama 35 *epoch*, dengan memantau nilai *loss function* untuk meminimalkan kesalahan. Model menggunakan *optimizer Adam* yang memanfaatkan gradien untuk memperbarui bobot model secara iteratif pada setiap langkah pelatihan, dengan tujuan mencapai nilai loss serendah mungkin.

3.5. Evaluasi

Evaluasi dilakukan menggunakan skor BLEU untuk mengukur kualitas *caption* yang dihasilkan model.

3.5.1. Evaluasi Model Terhadap Data Uji

Model yang telah dilatih kemudian dievaluasi menggunakan data uji untuk mengukur performanya. Hasil evaluasi model dengan berbagai kombinasi nilai *hyperparameter* yang diuji ditunjukkan pada Tabel 14.

Tabel 14. Hasil Evaluasi Model

Dimensi *Embedding*	Jumlah *Layer*	Jumlah *Head*	Skor BLEU-{1, 2, 3, 4}
128	2	2	{0.672, 0.559, 0.489, 0.437}
128	2	4	{0.671, 0.553, 0.480, 0.425}
128	1	1	{0.654, 0.530, 0.452, 0.395}
128	1	2	{0.649, 0.526, 0.450, 0.393}
64	1	1	{0.623, 0.499, 0.421, 0.361}
512	1	4	{0.634, 0.501, 0.419, 0.358}
128	4	4	{0.534, 0.386, 0.303, 0.246}
512	4	4	{0.439, 0.285, 0.217, 0.177}
512	4	8	{0.406, 0.245, 0.175, 0.137}
2056	8	8	{0.042, 0.006, 0.004, 0.003}

Berdasarkan hasil evaluasi pada Tabel 14, didapatkan skor BLEU-4 yang terbaik yaitu 0.437. Model tersebut menggunakan dimensi *embedding* 128, jumlah *layer* 2, dan jumlah *head* 2. Nilai *hyperparameter* tersebut akan digunakan sebagai dasar untuk menguji pengaruh perubahan dimensi *embedding*, jumlah *layer*, atau jumlah *head*, dengan menjaga nilai dua *hyperparameter* lainnya tetap.

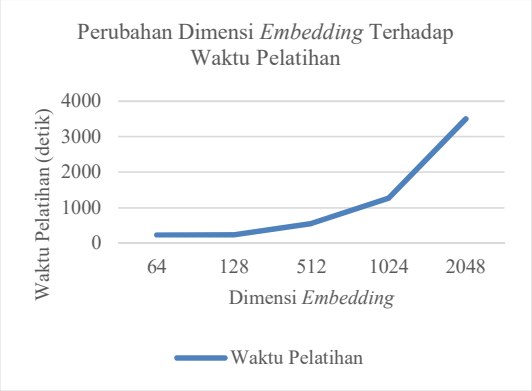
3.5.2. Pengaruh Perubahan Dimensi *Embedding*

Percobaan dilakukan dengan melatih model *image captioning* menggunakan *hyperparameters* sesuai Tabel 2, namun dengan jumlah *layer* 2, jumlah *head* 2, serta berbagai nilai dimensi *embedding* yang diuji, antara lain 64, 128, 512, 1024, dan 2048. Tujuan percobaan ini adalah untuk mengetahui pengaruh perubahan dimensi *embedding* terhadap hasil evaluasi model *image captioning*. Hasil evaluasi model untuk berbagai nilai dimensi *embedding* ditunjukkan pada Tabel 15.

Tabel 15. Hasil Evaluasi untuk Berbagai Dimensi *Embedding*

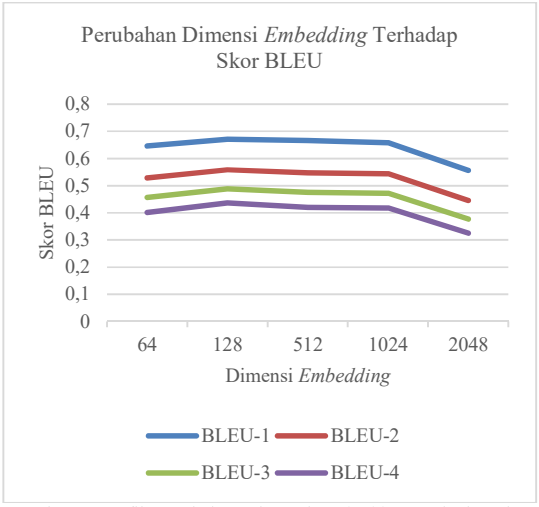
Dimensi *Embedding*	Waktu Pelatihan (detik)	BLEU			
		1	2	3	4
64	238	0.646	0.529	0.457	0.401
128	240	0.672	0.559	0.489	0.437
512	560	0.666	0.547	0.475	0.420
1024	1268	0.658	0.544	0.472	0.418
2048	3505	0.556	0.446	0.377	0.325
Rata-rata		0.6396	0.525	0.454	0.4002

Berdasarkan hasil evaluasi model yang ditunjukkan pada Tabel 15, dari lima kali percobaan dengan jumlah *layer* 2 dan jumlah *head* 2, diuji berbagai nilai dimensi *embedding* yaitu 64, 128, 512, 1024, dan 2048. Rata-rata skor BLEU-4 yang didapatkan adalah 0.4002. Skor BLEU-4 tertinggi adalah 0.437, yang dihasilkan saat dimensi *embedding* 128.



Gambar 4. Grafik Perubahan Dimensi *Embedding* Terhadap Waktu Pelatihan

Berdasarkan grafik yang ditunjukkan pada Gambar 4, waktu pelatihan meningkat seiring dengan bertambahnya dimensi *embedding*. Pada dimensi *embedding* kecil (64 hingga 128), waktu pelatihan relatif singkat. Waktu pelatihan meningkat signifikan pada dimensi *embedding* sedang (512 hingga 1024). Dimensi *embedding* yang sangat besar (2048) menyebabkan waktu pelatihan meningkat drastis. Hal ini menunjukkan bahwa semakin besar dimensi *embedding*, semakin lama waktu yang dibutuhkan untuk melatih model.



Gambar 5. Grafik Perubahan Dimensi *Embedding* Terhadap Skor BLEU

Berdasarkan grafik yang ditunjukkan pada Gambar 5, perubahan dimensi *embedding* memiliki pengaruh terhadap skor BLEU. Pada dimensi *embedding* yang lebih kecil (64 hingga 128), skor BLEU cenderung meningkat. Skor BLEU stabil dalam rentang dimensi *embedding* 128 hingga 512, menunjukkan performa model yang optimal. Namun, ketika dimensi *embedding* meningkat dari 512 ke 2048, skor BLEU mengalami penurunan, yang mungkin disebabkan oleh *overfitting* atau kompleksitas model yang berlebihan.

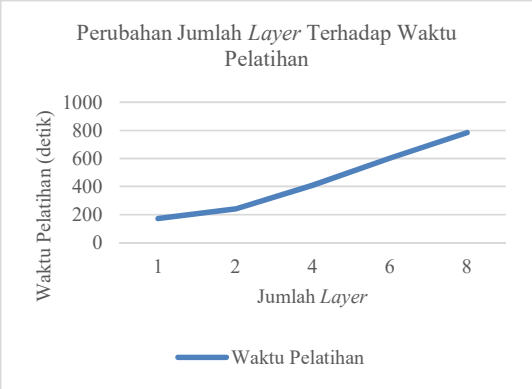
3.5.3. Pengaruh Perubahan Jumlah *Layer*

Percobaan dilakukan dengan melatih model *image captioning* menggunakan *hyperparameters* sesuai Tabel 2, namun dengan dimensi *embedding* 128, jumlah *head* 2, serta berbagai jumlah *layer* yang diuji, antara lain 1, 2, 4, 6, dan 8. Tujuan percobaan ini adalah untuk mengetahui pengaruh perubahan jumlah *layer* terhadap hasil evaluasi model *image captioning*. Hasil evaluasi model untuk berbagai jumlah *layer* ditunjukkan pada Tabel 16.

Tabel 16. Hasil Evaluasi untuk Berbagai Jumlah *Layer*

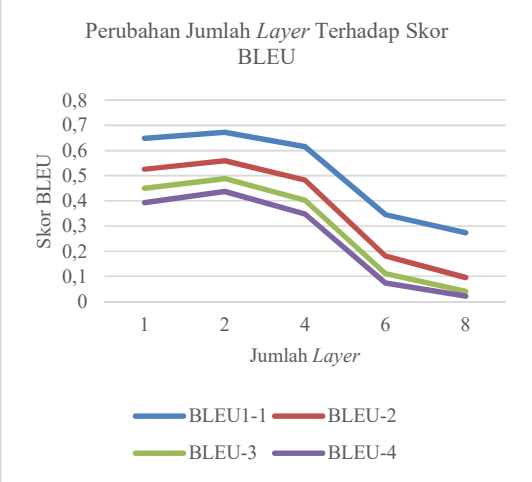
Jumlah *Layer*	Waktu Pelatihan (detik)	BLEU			
		1	2	3	4
1	172	0.649	0.526	0.450	0.393
2	240	0.672	0.559	0.489	0.437
4	409	0.615	0.482	0.403	0.347
6	602	0.345	0.182	0.111	0.074
8	785	0.273	0.096	0.041	0.022
Rata-rata		0.5108	0.369	0.2988	0.2546

Berdasarkan hasil evaluasi model yang ditunjukkan pada Tabel 16, dari lima kali percobaan dengan dimensi *embedding* 128 dan jumlah *head* 2, diuji berbagai jumlah *layer* yaitu 1, 2, 4, 6, dan 8. Rata-rata skor BLEU-4 yang didapatkan adalah 0.2546. Skor BLEU-4 tertinggi adalah 0.437, yang dihasilkan saat jumlah *layer* 2.



Gambar 6. Grafik Perubahan Jumlah *Layer* Terhadap Waktu Pelatihan

Berdasarkan grafik yang ditunjukkan pada Gambar 6, waktu pelatihan meningkat seiring dengan bertambahnya jumlah *layer* dalam model. Pada satu *layer*, waktu pelatihan adalah yang paling singkat. Saat jumlah *layer* bertambah menjadi 2, waktu pelatihan sedikit meningkat. Peningkatan jumlah *layer* menjadi 4 menyebabkan waktu pelatihan meningkat lebih signifikan. Jumlah *layer* yang lebih banyak lagi, yaitu 6 dan 8, menyebabkan waktu pelatihan meningkat banyak. Hal ini menunjukkan bahwa semakin banyak *layer*, semakin lama waktu yang dibutuhkan untuk melatih model, walaupun peningkatannya tidak sesignifikan dibandingkan pada perubahan dimensi *embedding*.



Gambar 7. Grafik Perubahan Jumlah *Layer* Terhadap Skor BLEU

Berdasarkan grafik yang ditunjukkan pada Gambar 7, perubahan jumlah *layer* memiliki pengaruh signifikan terhadap skor BLEU. Pada jumlah *layer* yang sedikit (1 hingga 2), skor BLEU cenderung meningkat, menunjukkan bahwa penambahan *layer* awal dapat membantu model menangkap lebih banyak informasi. Pada jumlah *layer* yang sedang (2 hingga 4), skor BLEU relatif stabil, menunjukkan bahwa jumlah *layer* ini memberikan performa optimal. Namun, ketika jumlah *layer* ditingkatkan lebih dari 4, skor BLEU mengalami penurunan yang signifikan, yang mungkin disebabkan oleh *overfitting* atau kompleksitas model yang berlebihan. Selain itu, ada kemungkinan bahwa informasi penting bisa hilang dikarenakan *layer* yang terlalu banyak, yang dapat mengakibatkan penurunan performa model.

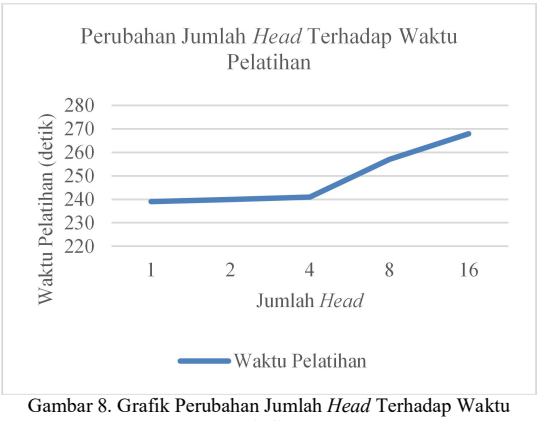
3.5.4. Pengaruh Perubahan Jumlah *Head*

Percobaan dilakukan dengan melatih model *image captioning* menggunakan *hyperparameters* sesuai Tabel 2, namun dengan dimensi *embedding* 128, jumlah *layer* 2, serta berbagai jumlah *head* yang diuji, antara lain 1, 2, 4, 8, dan 16. Tujuan percobaan ini adalah untuk mengetahui pengaruh perubahan jumlah *head* terhadap hasil evaluasi model *image captioning*. Hasil evaluasi model untuk berbagai jumlah *head* ditunjukkan pada Tabel 17.

Tabel 17. Hasil Evaluasi untuk Berbagai Jumlah *Head*

Jumlah *Head*	Waktu Pelatihan (detik)	BLEU			
		1	2	3	4
1	239	0.670	0.557	0.484	0.429
2	240	0.672	0.559	0.489	0.437
4	241	0.671	0.553	0.480	0.425
8	257	0.664	0.542	0.467	0.412
16	268	0.652	0.530	0.455	0.401
Rata-rata		0.6658	0.5482	0.475	0.4208

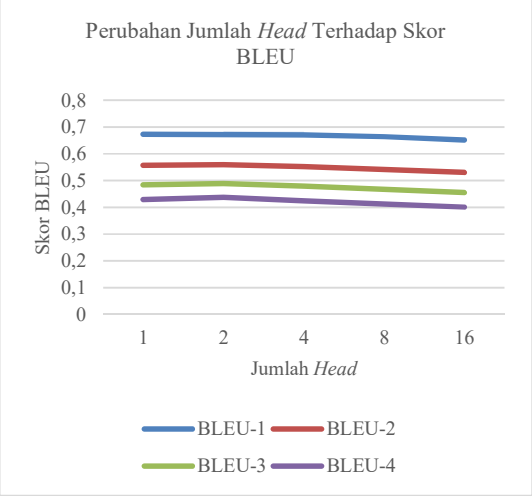
Berdasarkan hasil evaluasi model yang ditunjukkan pada Tabel 17, dari lima kali percobaan dengan dimensi *embedding* 128 dan jumlah *layer* 2, diuji berbagai jumlah *head* yaitu 1, 2, 4, 8, dan 16. Rata-rata skor BLEU-4 yang didapatkan adalah 0.4208. Skor BLEU-4 tertinggi adalah 0.437, yang dihasilkan saat jumlah *head* 2.



Gambar 8. Grafik Perubahan Jumlah *Head* Terhadap Waktu Pelatihan

Berdasarkan grafik yang ditunjukkan pada Gambar 8, perubahan jumlah *head* memiliki pengaruh terhadap waktu pelatihan. Pada jumlah *head* yang lebih kecil (1 hingga 4), waktu pelatihan cenderung stabil dengan sedikit peningkatan. Ketika jumlah *head* meningkat dari 4 hingga 16, waktu pelatihan memang meningkat, tetapi hanya beberapa detik. Ini menunjukkan bahwa meskipun ada peningkatan kompleksitas komputasi dengan bertambahnya jumlah *head*, dampaknya terhadap

waktu pelatihan tidak signifikan. Hal ini disebabkan walaupun terdapat banyak *head*, komputasi dilakukan secara paralel sehingga tidak terlalu banyak memakan waktu.



Gambar 9. Grafik Perubahan Jumlah *Head* Terhadap Skor BLEU

Berdasarkan grafik yang ditunjukkan pada Gambar 9, perubahan jumlah *head* memiliki pengaruh terhadap skor BLEU, tetapi pengaruhnya tidak signifikan. Pada jumlah *head* yang lebih kecil (1 hingga 2), skor BLEU cenderung stabil dengan sedikit peningkatan. Ketika jumlah *head* meningkat menjadi 16, skor BLEU berangsur-angsur mengalami penurunan walaupun tidak terlalu signifikan. Hal ini menunjukkan bahwa meskipun ada variasi dalam skor BLEU, perubahan jumlah *head* tidak memberikan dampak signifikan pada performa model.

3.5.5. Inferensi Model

Tabel 18 menunjukkan beberapa contoh *caption* yang dihasilkan oleh model pada beberapa gambar.

Tabel 18. Inferensi Model

Gambar	*Caption* Prediksi
	seorang ibu bersama dua anaknya yang berdiri di samping patung pohon dengan wajah manusia di dlas patung ini memiliki ekspresi wajah yang unik dengan mata besar hidung kecil dan mulut merah terbuka ibu tersebut mengenakan pakaian panjang berwarna hijau muda dan jilbab bermotif sementara salah satu anaknya duduk di stroller mengenakan kaos putih dan celana biru dan anak yang lainnya mengenakan baju berwarna biru dan kerudung biru tua di belakang mereka terdapat pagar kayu dan pemandangan pepohonan hijau
	pintu masuk ke area wisata sanggaluri yang dihiasi dengan patung besar semar karakter wayang jawa patung tersebut berwarna oranye dengan detail hitam dan putih menampilkan ekspresi wajah yang unik gerbang masuk berada di bawah patung ini memiliki warna warna warni dan putih yang di bawah patung terdapat patung besar dengan desain tradisional jawa yang terlihat jelas di atas tebing buatan langit biru cerah dengan sedikit awan memberikan suasana yang ceria dan ceria pada gambar ini pengunjung di sekitar patung tampak berinteraksi dan menikmati suasana

Secara umum model sudah bisa melakukan *captioning* gambar objek wisata di Purbalingga antara lain D'Las, Goa Lawa, Owabong, Purbasari, dan Sanggaluri. Kalimat yang dihasilkan cukup bermakna walaupun masih terdapat kesalahan tata bahasa. Kata yang dihasilkan terbatas pada kamus kosa kata yang terdapat pada dataset *caption*. Jika gambar di luar lingkup dataset, hasilnya cenderung kurang akurat atau bahkan tidak benar. Namun, pada beberapa gambar objek wisata dengan prediksi *caption* yang salah, prediksi tersebut masih bisa dalam menentukan salah satu dari objek pada gambar, nama tempat wisata, aktivitas yang ditunjukkan, atau suasana pada gambar.

3.6. *Deployment*

Pada tahap ini dilakukan *deployment* model, yaitu proses implementasi dari model yang telah dilatih ke dalam aplikasi web. Model yang digunakan adalah model dengan skor BLEU-4 terbaik, yaitu model dengan dimensi *embedding* 128, jumlah *layer* 2, dan jumlah *head* 2. Proses ini mengikuti metode SDLC *waterfall* yang meliputi tahap analisis kebutuhan, desain sistem, implementasi, dan pengujian.

3.6.1. Analisis

Tahap ini bertujuan untuk mengidentifikasi permasalahan dan kebutuhan yang diperlukan dalam sistem. Pengumpulan data dilakukan menggunakan studi literatur dan analisis observasi untuk mengetahui kebutuhan. Tabel 19 menunjukkan kebutuhan pengguna.

Tabel 19. Kebutuhan Pengguna

Role	Kode	Kebutuhan
Admin	URS-A-01	Pengguna dapat mengakses sistem
	URS-A-02	Pengguna dapat melakukan *login*
	URS-A-03	Pengguna dapat mengunggah file gambar
	URS-A-04	Pengguna dapat mengambil gambar menggunakan kamera
	URS-A-05	Pengguna dapat melihat hasil *image captioning*
	URS-A-06	Pengguna dapat mendengarkan *caption* yang disuarakan dalam bahasa Indonesia
	URS-A-07	Pengguna dapat melihat dan menghapus riwayat *image captioning*
	URS-A-08	Pengguna dapat melakukan *logout*
Guest	URS-B-01	Pengguna dapat mengakses sistem
	URS-B-02	Pengguna dapat mengunggah file gambar
	URS-B-03	Pengguna dapat mengambil gambar menggunakan kamera
	URS-B-04	Pengguna dapat melihat hasil *image captioning*
	URS-B-05	Pengguna dapat mendengarkan *caption* yang disuarakan dalam bahasa Indonesia

Analisis kebutuhan sistem dilakukan untuk merinci fitur-fitur yang terdapat pada sistem. Tabel 20 menunjukkan kebutuhan sistem.

Tabel 20. Kebutuhan Sistem

Kode	Kebutuhan
SRS-01	Sistem menyediakan fungsi *login* dan *logout*
SRS-02	Sistem mampu membatasi fungsi sesuai dengan hak akses
SRS-03	Sistem menyediakan fungsi untuk menyalakan dan menggunakan kamera
SRS-04	Sistem mampu menangani *input* gambar dari kamera atau file
SRS-05	Inputan file pada formulir harus berupa gambar jpg/jpeg/png
SRS-06	Sistem menyediakan fungsi untuk memproses gambar dan mengembalikan *caption*
SRS-07	Sistem menyediakan fungsi untuk mengubah teks menjadi suara

3.6.2. Desain

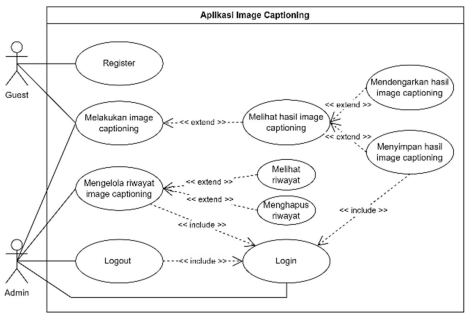
Pada tahap ini dilakukan perancangan berdasarkan analisis kebutuhan pada tahap sebelumnya. Tahap ini meliputi perancangan *use case diagram*, *activity diagram*, *sequence diagram*, *Entity Relationship Diagram* (ERD), serta *class diagram*. Desain ERD dan diagram UML dibuat menggunakan *draw.io*.

3.6.2.1. *Use Case Diagram*

Use case diagram sistem disajikan pada Gambar 10. Terdapat dua jenis pengguna (aktor) yang berinteraksi dengan sistem ini, yaitu *guest* (pengguna umum) dan admin (pengguna yang *login*).

Aktor *guest* dapat melakukan *image captioning*, di mana mereka dapat melihat dan mendengarkan hasil dari *image captioning* tersebut. *Guest* juga memiliki opsi untuk mendaftar jika ingin mendapatkan akses lebih lanjut.

Aktor admin dapat melakukan *login* untuk mengakses fitur tambahan. Sama seperti *guest*, admin juga dapat melakukan *image captioning*, melihat hasilnya, dan mendengarkan hasil *image captioning*. Selain itu, admin dapat menyimpan hasil *image captioning* tersebut. Admin juga memiliki akses untuk mengelola riwayat *image captioning*-nya. Dalam fitur ini, admin dapat melihat riwayat *image captioning* pribadi dan menghapus riwayat tersebut.



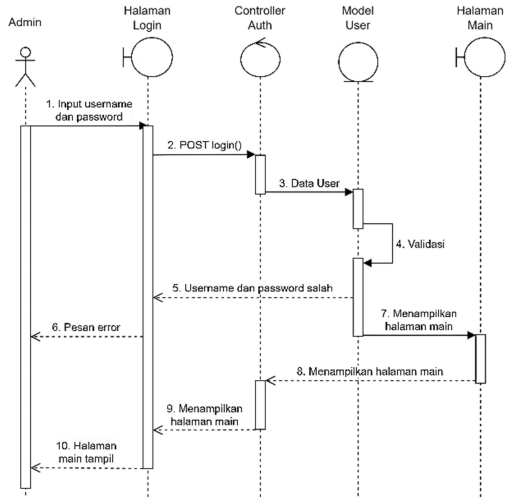
Gambar 10. *Use Case Diagram* Sistem

3.6.2.2. *Sequence Diagram*

Sequence diagram sistem dirancang berdasarkan *use case diagram* pada Gambar 42. Berikut merupakan penjelasan untuk masing-masing *sequence diagram* yang dirancang.

a. *Login*

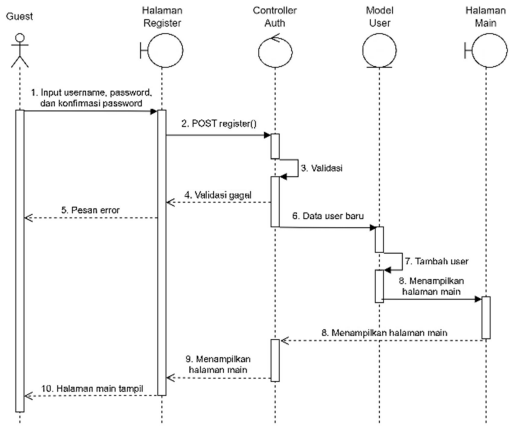
Gambar 11 menunjukkan *sequence diagram login*. Proses dimulai saat admin memasukkan *username* dan *password* di halaman *login*, yang kemudian mengirimkan permintaan ke *Controller Auth*. *Controller Auth* meminta data *user* dari *Model Users* untuk validasi. Jika validasi berhasil, admin akan diarahkan ke halaman utama. Jika gagal, pesan kesalahan akan ditampilkan.



Gambar 11. *Sequence Diagram Login*

b. *Register*

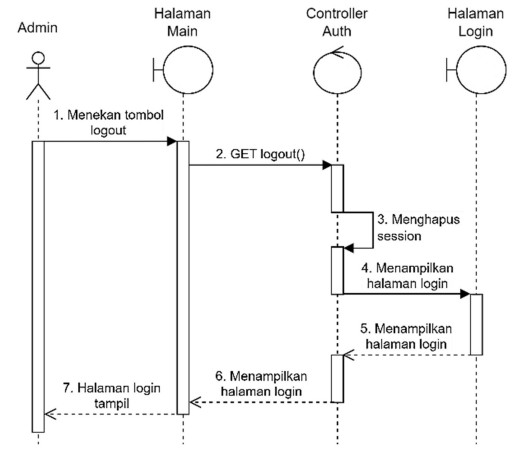
Gambar 12 menunjukkan *sequence diagram register*. Proses dimulai saat *guest* memasukkan informasi di halaman *register*. *Controller Auth* memvalidasi data. Jika validasi berhasil, data user baru ditambahkan ke *Model Users*, dan halaman utama ditampilkan. Jika gagal, pesan kesalahan akan ditampilkan.



Gambar 12. *Sequence Diagram Register*

c. *Logout*

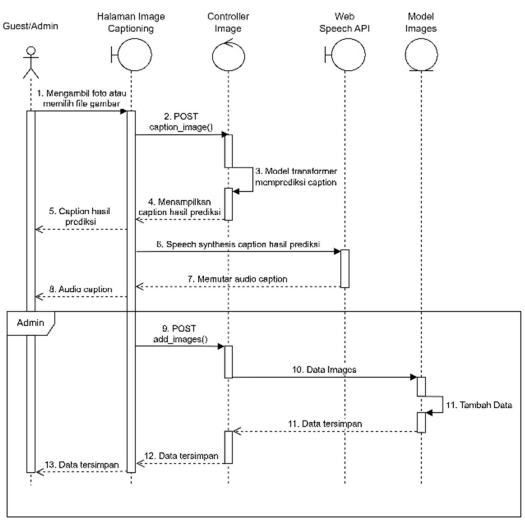
Gambar 13 menunjukkan *sequence diagram logout*. Proses dimulai saat admin menekan tombol *logout*, yang mengirimkan permintaan ke *Controller Auth*. *Controller Auth* menerima permintaan, menghapus sesi admin, dan menampilkan halaman *login* kembali, menandakan bahwa *logout* berhasil.



Gambar 13. *Sequence Diagram Logout*

d. *Image Captioning*

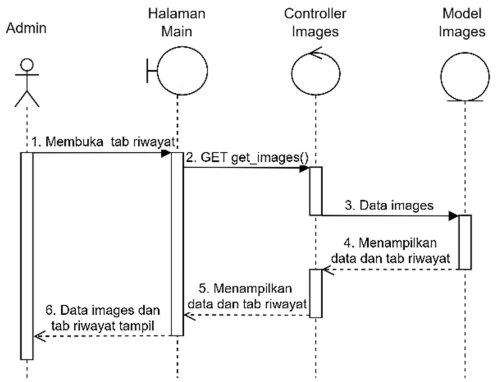
Gambar 14 menunjukkan *sequence diagram* untuk proses *image captioning*. Proses dimulai saat pengguna (*guest* atau admin) mengunggah gambar di halaman *image captioning*, yang mengirim permintaan ke *Controller Images*. *Controller Images* menggunakan model *transformer* untuk menghasilkan prediksi *caption* gambar, kemudian mengirim kembali hasil *caption*-nya. Halaman ini kemudian mengirim teks *caption* ke *Web Speech* API untuk diubah menjadi audio dan memutar hasil audio tersebut. Jika admin ingin menyimpan gambar beserta *caption*-nya, permintaan dikirim ke *Controller Images* untuk menyimpan data tersebut ke *database* melalui *Model Images*.



Gambar 14. *Sequence Diagram Image Captioning*

e. Melihat Riwayat *Image Captioning*

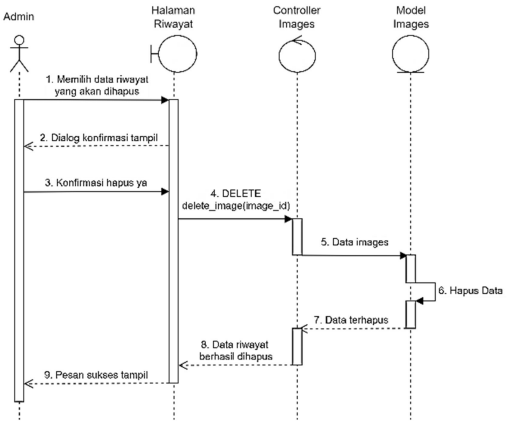
Gambar 15 menunjukkan *sequence diagram* untuk melihat riwayat *image captioning*. Proses dimulai saat admin membuka tab riwayat di halaman utama, yang mengirim permintaan *Controller Images* untuk memperoleh data riwayat *image captioning*. *Controller Images* kemudian meminta data dari *Model Images*, yang mengembalikan data *images* ke *Controller Images* yang kemudian ditampilkan di halaman utama.



Gambar 15. *Sequence Diagram* Melihat Riwayat *Image Captioning*

f. Menghapus Riwayat *Image Captioning*

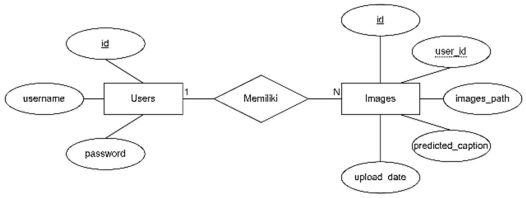
Gambar 16 menunjukkan *sequence diagram* untuk menghapus riwayat *image captioning*. Proses dimulai saat admin memilih data riwayat yang akan dihapus pada halaman riwayat. Halaman riwayat menampilkan dialog konfirmasi kepada admin. Setelah admin mengonfirmasi penghapusan, halaman riwayat mengirim permintaan ke *Controller Images*. *Controller Images* meminta *Model Images* untuk menghapus data dari *database*. *Model Images* menghapus data dan mengirimkan konfirmasi kembali ke *Controller Images*, yang kemudian menampilkan pesan sukses.



Gambar 16. *Sequence Diagram* Menghapus Riwayat *Image Captioning*

3.6.2.3. *Entity Relationship Diagram* (ERD)

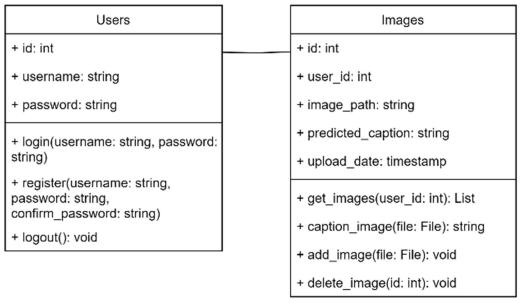
ERD sistem ditunjukkan pada Gambar 17. Terdapat dua entitas yang saling berhubungan yaitu *users* dan *images*. Relasi yang terjadi antara *users* dengan *images* adalah *users* memiliki *images*. Kardinalitas dari relasi ini adalah *one to many* di mana satu *users* bisa memiliki banyak *images*.



Gambar 17. *Entity Relationship Diagram* (ERD) Sistem

3.6.2.4. *Class Diagram*

Class diagram bertujuan untuk memvisualisasikan kelas yang akan dibuat saat pengembangan berlangsung. *Class diagram* sistem disajikan pada Gambar 18.



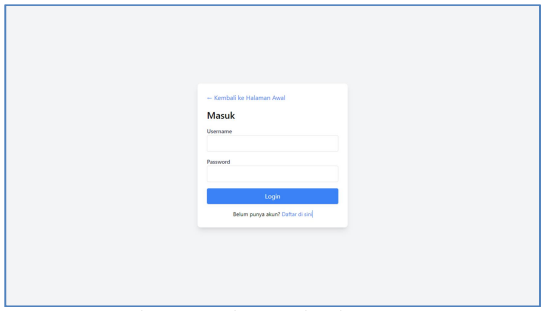
Gambar 18. *Class Diagram* Sistem

3.6.3. Implementasi

Sistem ini dikembangkan menggunakan *framework Flask* yang berbasis *Python* pada sisi server. Sedangkan pada sisi klien menggunakan HTML, CSS, dan *JavaScript* dengan *template engine Jinja2*. *Database* yang digunakan adalah *MySQL*. Berikut merupakan penjelasan untuk masing-masing halaman yang terdapat pada sistem.

3.6.3.1. Halaman *Login*

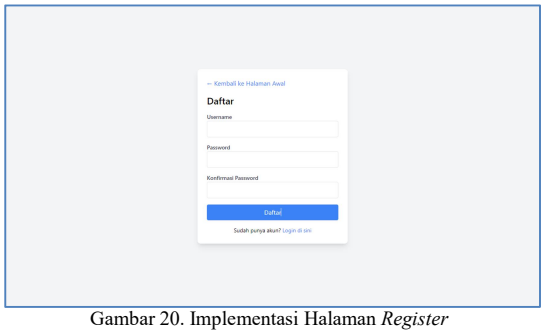
Pada halaman *login*, pengguna harus memasukkan *username* dan *password* yang terdaftar. Jika akun terverifikasi, pengguna diarahkan ke halaman utama. Jika tidak, pesan kesalahan ditampilkan. Gambar 19 merupakan tampilan implementasi halaman *login*.



Gambar 19. Implementasi Halaman *Login*

3.6.3.2. Halaman *Register*

Pada halaman *register*, pengguna mengisi *username*, *password*, dan konfirmasi *password*. Jika validasi berhasil, registrasi akun baru berhasil dan pengguna diarahkan ke halaman utama. Jika validasi gagal, pesan kesalahan ditampilkan. Gambar 20 merupakan tampilan implementasi halaman *register*.



Gambar 20. Implementasi Halaman *Register*

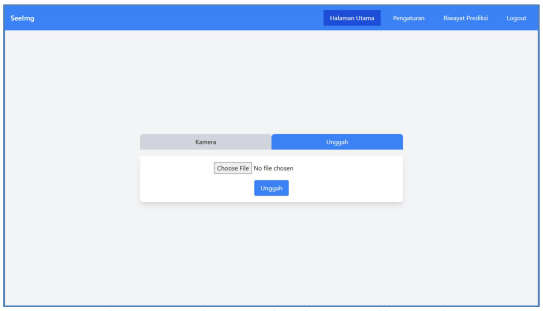
3.6.3.3. Tab *Input* Kamera

Di halaman utama, pengguna dapat memilih tab kamera untuk mengambil foto langsung. Setelah foto diambil, gambar akan diproses untuk menghasilkan *caption*. Gambar 21 merupakan tampilan implementasi tab *input* kamera.

Gambar 21. Implementasi Tab *Input* Kamera

3.6.3.4. Tab *Input* File Gambar

Pada tab unggah gambar, pengguna dapat mengunggah gambar dari perangkat mereka. Gambar 22 merupakan tampilan implementasi tab *input* file gambar.



Gambar 22. Implementasi Tab *Input* File Gambar

Setelah memilih file, nama file dan pratinjau gambar akan ditampilkan. Gambar 23 menunjukkan pratinjau gambar yang dipilih. Pengguna dapat mengklik tombol unggah untuk memproses gambar dan menghasilkan *caption*.

Gambar 23. Pratinjau Gambar yang Dipilih

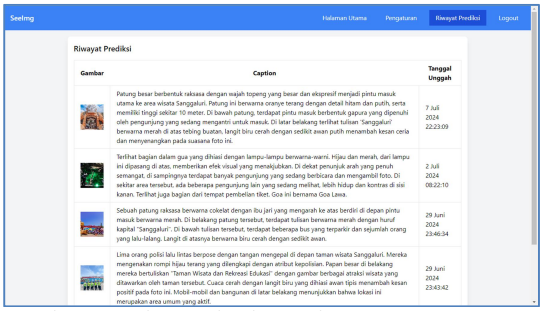
3.6.3.5. Tab *Output*

Setelah pengunggahan, gambar akan diproses oleh model *transformer* untuk menghasilkan *caption*. *Caption* ditampilkan di bawah gambar dan dibacakan oleh *Web Speech API*. Terdapat tiga tombol interaktif antara lain, putar ulang, hentikan suara, dan tutup tab *output*. Gambar 24 merupakan tampilan implementasi tab *output.*

Gambar 24. Implementasi Tab *Output*

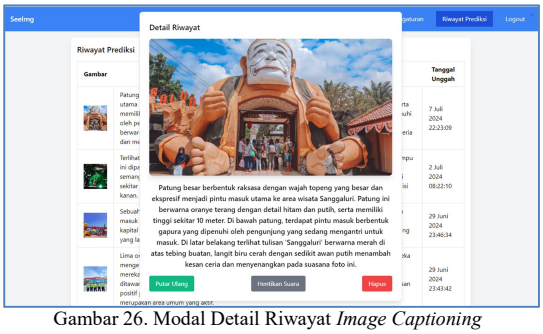
3.6.3.6. Tab Riwayat Hasil *Image Captioning*

Tab riwayat hasil *image captioning* hanya dapat diakses oleh admin, yang menampilkan daftar gambar yang telah diproses beserta hasil *caption* dan tanggal unggahnya. Gambar 25 merupakan tampilan implementasi halaman riwayat *image captioning*.



Gambar 25. Implementasi Halaman Riwayat *Image Captioning*

Admin dapat melihat detail gambar dalam modal, memutar audio *caption*, dan menghapus riwayat gambar. Gambar 26 menunjukkan modal detail riwayat *image captioning*.



Gambar 26. Modal Detail Riwayat *Image Captioning*

3.6.4. Pengujian

Pengujian dilakukan menggunakan metode pengujian *blackbox*. Pengujian *blackbox* bertujuan untuk mengetahui apakah sistem berjalan sesuai dengan yang diharapkan Hasil pengujian *blackbox* disajikan pada Tabel 21.

Tabel 21. Hasil Pengujian *Blackbox*

Modul	*Input*	*Output* yang Diharapkan	Hasil
Login	Memasukkan *username* dan password yang valid	Berhasil *login* dan diarahkan ke halaman *main*	Valid
	Memasukkan *username* atau password yang tidak valid	Gagal *login* dan menampilkan pesan *error*	Valid
Register	Memasukkan *username*, password, dan konfirmasi password yang valid	Berhasil *register* dan diarahkan ke halaman *main*	Valid
	Memasukkan *username*, password, dan konfirmasi password yang tidak valid	Gagal *register* dan menampilkan pesan *error*	Valid
Halaman	Menekan tab kamera	Sistem menampilkan tab kamera dan menyalakan kamera	Valid

awal	Menekan tab unggah	Sistem menampilkan tab unggah dan formulir *input* file	Valid
	Mengambil foto atau mengunggah file gambar yang valid	Berhasil unggah, sistem memproses gambar, menampilkan tulisan prediksi *caption* dari gambar inputan, serta membacakan *caption* tersebut	Valid
	Mengambil foto atau mengunggah file gambar yang tidak valid atau terdapat kesalahan pada server	Gagal memproses gambar dan menampilkan pesan *error*	Valid
Halaman utama	Menekan tab kamera	Sistem menampilkan tab kamera dan menyalakan kamera	Valid
	Menekan tab unggah	Sistem menampilkan tab unggah dan formulir *input* file	Valid
	Mengambil foto atau mengunggah file gambar yang valid	Berhasil unggah, sistem memproses gambar, menampilkan tulisan prediksi *caption* dari gambar inputan, membacakan *caption* tersebut, serta menyimpan riwayat ke *database*	Valid
	Mengambil foto atau mengunggah file gambar yang tidak valid atau terdapat kesalahan pada server	Gagal memproses gambar dan menampilkan pesan *error*	Valid
Riwayat prediksi	Menekan tab riwayat prediksi	Sistem menampilkan tab dan tabel riwayat prediksi pengguna	Valid
	Menekan gambar di tabel riwayat prediksi	Menampilkan modal detail gambar dan *caption*-nya	Valid
	Menekan tombol hapus pada modal detail gambar	Menampilkan konfirmasi penghapusan, jika konfirmasi ya, maka data akan dihapus dan menampilkan pesan sukses	Valid
Modal *output*	Menekan tombol putar ulang	Membaca ulang *caption* yang ditampilkan	Valid
	Menekan tombol hentikan suara	Menghentikan suara	Valid
Logout	Menekan tombol *logout*	Pengguna akan keluar sistem dan diarahkan ke halaman *login*	Valid

Hasil pengujian *blackbox* menunjukkan seluruh kasus uji bernilai valid yang mana dapat ditarik kesimpulan bahwa sistem sudah sesuai dengan kebutuhan yang diinginkan dan dapat digunakan.

4. DISKUSI

Penelitian ini berhasil membuat model *image captioning* pada gambar objek wisata di Purbalingga menggunakan arsitektur *transformer*. Model yang dikembangkan mampu menghasilkan deskripsi otomatis dari gambar-gambar objek wisata. Data yang digunakan mencakup gambar dari lima objek wisata di Purbalingga, antara lain D'Las, Goa Lawa, Owabong, Purbasari, dan Sanggaluri. Seluruh proses pengembangan dan implementasi dilakukan menggunakan bahasa pemrograman *Python*. Model arsitektur *transformer* terdiri dari lapisan ekstrasi fitur gambar menggunakan ResNet50 serta beberapa layer *encoder* dan *decoder* dengan *attention mechanism* dan *feed-forward neural network*. Evaluasi model *image captioning* dilakukan menggunakan metrik BLEU, dengan hasil terbaik yang dicapai adalah BLEU-{1, 2, 3, 4} = {0.672, 0.559, 0.489, 0.437}. Nilai ini dicapai dengan menggunakan kombinasi *hyperparameter* dimensi *embedding* 128, jumlah *layer* 2, dan jumlah *head* 2.

Referensi penelitian oleh[19] menunjukkan bahwa arsitektur *transformer*, telah terbukti efektif dalam tugas-tugas seperti *image captioning*, di mana model menghasilkan deskripsi bahasa alami untuk gambar secara otomatis. Penelitian[7] juga menunjukkan bahwa *transformer* memberikan hasil yang lebih baik dibandingkan dengan model *Long Short-Term Memory* (LSTM) dalam tugas *image captioning*. Ini mendukung penggunaan *transformer* dalam penelitian, yang menghasilkan deskripsi gambar dengan skor BLEU yang cukup baik.

Untuk mengetahui pengaruh perubahan *hyperparameter* dimensi *embedding*, jumlah *layer*, dan jumlah *head* terhadap performa model, dilakukan eksperimen dengan berbagai kombinasi nilai *hyperparameter* tersebut. Hasilnya menunjukkan bahwa peningkatan dimensi *embedding* meningkatkan waktu pelatihan secara drastis dan menyebabkan penurunan skor BLEU setelah titik optimal pada dimensi 128, menunjukkan potensi *overfitting*. Penambahan jumlah *layer* meningkatkan waktu pelatihan walau tidak sesignifikan perubahan dimensi *embedding*, dengan performa optimal tercapai pada 2 *layer*, sementara jumlah *layer* lebih dari 4 menyebabkan penurunan skor BLEU secara drastis. Perubahan jumlah *head* memiliki dampak minimal pada waktu pelatihan dan skor BLEU, dengan performa stabil pada jumlah *head* kecil (1 hingga 2) dan sedikit penurunan pada jumlah *head* yang lebih besar.

Model *image captioning* pada gambar objek wisata di Purbalingga diimplementasikan dalam aplikasi berbasis web menggunakan metode SDLC *waterfall*. Sistem ini memiliki dua jenis pengguna, yaitu admin dan *guest*, dengan hak akses berbeda. Desain sistem menggunakan *Unified Modeling Language* (UML), dan implementasi dilakukan menggunakan bahasa pemrograman *Python* dengan *framework Flask* serta basis data *MySQL*. Aplikasi ini memungkinkan pengguna untuk mengunggah file gambar objek wisata, termasuk menangkap foto langsung dengan kamera, dan mendapatkan deskripsi otomatis/*caption* dalam bahasa Indonesia. Fitur *text-to-speech* yang menggunakan *Web Speech* API berbasis bahasa pemrograman *JavaScript* juga disertakan untuk membacakan deskripsi gambar yang dihasilkan. Admin memiliki hak akses tambahan, dapat mengelola dan menyimpan riwayat deskripsi gambar yang diunggah. Hasil pengujian dengan metode *blackbox* menunjukkan hasil valid di setiap kasus uji, menandakan bahwa sistem berjalan sesuai kebutuhan dan layak digunakan. Aplikasi ini diharapkan dapat memberikan manfaat dalam menguraikan deskripsi/*caption* gambar objek wisata

di Purbalingga secara otomatis serta meningkatkan aksesibilitas bagi penyandang tunanetra dalam memahami keterangan gambar tersebut.

5. KESIMPULAN

Penelitian ini berhasil mengembangkan model *image captioning* untuk gambar objek wisata di Purbalingga menggunakan arsitektur Resnet50 dan *transformer*. Model ini mampu menghasilkan deskripsi otomatis untuk gambar-gambar objek wisata seperti D'Las, Goa Lawa, Owabong, Purbasari, dan Sanggaluri. Evaluasi model menggunakan metrik BLEU menunjukkan bahwa skor BLEU-{1, 2, 3, 4} terbaik yang diperoleh adalah {0.672, 0.559, 0.489, 0.437}. Skor ini dicapai dengan kombinasi *hyperparameter* dimensi *embedding* 128, jumlah *layer* 2, dan jumlah *head* 2. Penambahan *embedding* dan *layer* meningkatkan waktu pelatihan dan menurunkan skor BLEU, sementara perubahan jumlah *head* tidak terlalu memengaruhi hasil

Model diimplementasikan dalam aplikasi web menggunakan metode SDLC *waterfall*, *framework Flask*, dan basis data *MySQL*. Aplikasi memungkinkan pengguna mengunggah gambar objek wisata, mendapatkan deskripsi otomatis dalam bahasa Indonesia, dan mendengarkan deskripsi tersebut dengan fitur *text-to-speech* berbasis *Web Speech* API. Pengujian *blackbox* menunjukkan bahwa sistem berjalan sesuai kebutuhan dan layak digunakan. Aplikasi diharapkan dapat memberikan manfaat dalam memberikan deskripsi gambar objek wisata di Purbalingga secara otomatis dan meningkatkan aksesibilitas bagi penyandang tunanetra.

Untuk pengembangan lebih lanjut, disarankan agar dataset diperluas dengan lebih banyak gambar dari berbagai objek wisata dan kondisi yang berbeda, serta mengeksplorasi arsitektur model yang lebih kompleks seperti GPT atau BERT. Penggunaan metrik evaluasi tambahan seperti METEOR atau ROUGE dapat memberikan penilaian yang lebih komprehensif terhadap kualitas deskripsi. Selain itu, penambahan fitur-fitur baru dalam aplikasi, seperti deteksi objek secara *real-time* dan opsi bahasa lain untuk deskripsi, akan meningkatkan pengalaman pengguna secara keseluruhan.

DAFTAR PUSTAKA

[1] M. D. Zakir Hossain, F. Sohel, M. F. Shiratuddin, and H. Laga, "A comprehensive survey of deep learning for image captioning," *ACM Comput. Surv.*, vol. 51, no. 6, 2019, doi: 10.1145/3295748.

[2] D. H. Fudholi, "Image Captioning Approach for Household Environment Visual Understanding," *Int. J. Inf. Syst. Technol.*, vol. 5, no. 36, pp. 292–298, 2021, [Online]. Available: https://ijistech.org/ijistech/index.php/ijistech/article/view/135/pdf

[3] S. S. Rawat, K. S. Rawat, and R. Nijhawan, "A Novel Convolutional Neural Network-Gated Recurrent Unit approach for Image Captioning," in *2020 Third International Conference on Smart Systems and Inventive Technology (ICSSIT)*, 2020, pp. 704–708. doi: 10.1109/ICSSIT48917.2020.9214109.

[4] S. Degadwala, D. Vyas, H. Biswas, U. Chakraborty, and S. Saha, "Image Captioning Using Inception V3 Transfer Learning Model," in *2021 6th International Conference on Communication and Electronics Systems (ICCES)*, 2021, pp. 1103–1108. doi: 10.1109/ICCES51350.2021.9489111.

[5] Y. Chu, X. Yue, L. Yu, M. Sergei, and Z. Wang, "Automatic Image Captioning Based on ResNet50 and LSTM with Soft Attention," *Wirel. Commun. Mob. Comput.*, vol. 2020, 2020, doi: 10.1155/2020/8909458.

[6] R. Khan, M. S. Islam, K. Kanwal, M. Iqbal, M. I. Hossain, and Z. Ye, "A Deep Neural Framework for Image Caption Generation Using GRU-Based Attention Mechanism," no. i, 2022, [Online]. Available: http://arxiv.org/abs/2203.01594

[7] P. Dandwate, C. Shahane, V. Jagtap, and S. C. Karande, "Comparative Study of Transformer and LSTM Network with Attention Mechanism on Image Captioning," *Lect. Notes Networks Syst.*, vol. 720 LNNS, pp. 527–539, 2023, doi: 10.1007/978-981-99-3761-5_47.

[8] R. Mulyawan, A. Sunyoto, and A. H. Muhammad, "Pre-Trained CNN Architecture Analysis for Transformer-Based Indonesian Image Caption Generation Model," *Int. J. Informatics Vis.*, vol. 7, no. 2, pp. 487–493, 2023, doi: 10.30630/joiv.7.2.1387.

[9] M. S. Mohammad Suryawinata, *Buku Ajar Mata Kuliah Pengembangan Aplikasi Berbasis Web*. 2019. doi: 10.21070/2019/978-602-5914-81-2.

[10] M. E. Prastyo, "Aplikasi Text to Speech Berbasis Javascript," *Visualika*, vol. 7, no. 1, pp. 89–101, 2022, [Online]. Available: http://www.jurnas.stmikmj.ac.id/index.php/visualika/article/view/147/72

[11] E. Etriyanti, D. Syamsuar, and N. Kunang, "Implementasi Data Mining Menggunakan Algoritme Naive Bayes Classifier dan C4.5 untuk Memprediksi Kelulusan Mahasiswa," *Telematika*, vol. 13, no. 1, pp. 56–67, 2020,

doi: 10.35671/telematika.v13i1.881.

[12] Adawiyah Ritonga and Yahfizham Yahfizham, "Studi Literatur Perbandingan Bahasa Pemrograman C++ dan Bahasa Pemrograman Python pada Algoritma Pemrograman," *J. Tek. Inform. dan Teknol. Inf.*, vol. 3, no. 3, pp. 56–63, 2023, doi: 10.55606/jutiti.v3i3.2863.

[13] R. Y. Azhari, "Technology and Informatics Insight Journal Web Service Framework: flask dan fastAPI," *Technol. Informatics Insight J.*, vol. 1, no. 1, pp. 80–87, 2020, [Online]. Available: https://jurnal.universitasputrabangsa.ac.id/index.php/tiij

[14] N. Khaerunnisa and N. Nofiyati, "Sistem Informasi Pelayanan Administrasi Kependudukan Berbasis Web Studi Kasus Desa Sidakangen Purbalingga," *J. Tek. Inform.*, vol. 1, no. 1, pp. 25–33, 2020, doi: 10.20884/1.jutif.2020.1.1.9.

[15] S. Julianto and S. Setiawan, "Perancangan Sistem Informasi Pemesanan Tiket Bus Pada Po. Handoyo Berbasis Online," *Simatupang, Julianto Sianturi, Setiawan*, vol. 3, no. 2, pp. 11–25, 2019, [Online]. Available: https://journal.amikmahaputra.ac.id/index.php/JIT/article/view/56/48

[16] F. N. Hasanah and R. S. Untari, *Buku Ajar Rekayasa Perangkat Lunak*. UMSIDA PRESS, 2020. doi: 10.21070/2020/978-623-6833-89-6.

[17] Y. I. Kurniawan, A. Fatikasari, M. L. Hidayat, and M. Waluyo, "Prediction for Cooperative Credit Eligibility Using Data Mining Classification With C4.5 Algorithm," *J. Tek. Inform.*, vol. 2, no. 2, pp. 67–74, 2021, doi: 10.20884/1.jutif.2021.2.2.49.

[18] F. Y. Al Irsyadi, Supriyadi, and Y. I. Kurniawan, "Interactive educational animal identification game for primary schoolchildren with intellectual disability," *Int. J. Adv. Trends Comput. Sci. Eng.*, vol. 8, no. 6, pp. 3058–3064, 2019, doi: 10.30534/ijatcse/2019/64862019.

[19] O. Ondeng, H. Ouma, and P. Akuon, "A Review of Transformer-Based Approaches for Image Captioning," *Appl. Sci.*, vol. 13, no. 19, pp. 1–38, 2023, doi: 10.3390/app131911103.