Pembangunan Aplikasi Alat Bantu Proses Anotasi Menggunakan Progressive Web Apps

I Gede Bagus Artha Suryawan[1], Y. Sigit Purnomo WP.[2], Ernawati[3]
[1,2,3]Program Studi Teknik Informatika, Fakultas Teknologi Industri
Universitas Atma Jaya Yogyakarta
Jl. Babarsari No 43, Yogyakarta, 55281, Daerah Istimewa Yogyakarta, Indonesia Email:
[1]x1.artha@gmail.com, [2]sigit.purnomo@uajy.ac.id, [3]ernawati@uajy.ac.id

Masuk: 29 Juli 2019 ; Direvisi: 26 Oktober 2019 ; Diterima: 29 Oktober 2019

Abstract. *Dataset is the key to natural language processing. Datasets can be created by giving information to a text or sentence (annotation process). The process can be done either manually or automatically. The automatic one is considered easier and it takes a shorter period of time compared to the manual process. This study is done to develop an application of data annotation (AADT) using Progressive Web Apps (PAW). This application is built by implementing text information extraction to support NER (Named Entity Recognition) approach. The result shows that AADT successfully helped the annotation process of the text data. PWA application sustains the data storage which is continually accessed with small change rate. Besides, PWA is able to keep this application stable. NER (Named Entity Recognition) implementation in this application successfully produced the dataset desired by its users.*
Key words: *Natural Language Processing, Named Entity Recognition, Annotation, Progressive Web Apps.*

Abstrak. *Dataset adalah komponen utama dalam pemrosesan bahasa alami. Dataset dapat dibuat dengan melakukan pemberian informasi pada sebuah teks ataupun sebuah kalimat (proses anotasi). Pembuatan dataset dapat dilakukan secara manual dan otomatis. Pembuatan dataset secara otomatis lebih mudah dan membutuhkan waktu yang lebih singkat daripada pembuatan secara manual. Penelitian ini bertujuan membuat aplikasi alat bantu proses anotasi AADT (Aplikasi Anotasi Data Teks) dengan menggunakan PWA (Progressive Web Apps). Aplikasi ini dibangun dengan menerapkan information extraction terhadap teks untuk mendukung pendekatan NER (Named Entity Recognition). Hasil penelitian menunjukkan bahwa aplikasi ini berhasil membantu proses anotasi data teks. Penggunaan PWA pada aplikasi ini membantu dalam penampungan data yang akan diakses secara terus menerus dengan tingkat perubahan yang kecil. Selain itu PWA membuat aplikasi tetap stabil. Penerapan NER pada aplikasi ini berhasil menghasilkan dataset yang sesuai dengan keinginan pengguna.*
Kata Kunci: *Pemrosesan Bahasa Alami, Named Entity Recognition, Anotasi, Progressive*

1. Pendahuluan

Pengolahan bahasa alami adalah pengembangan berbagai teknik komputasi untuk menganalisis dan menampilkan teks dalam bahasa alami pada satu atau lebih tingkat analisis linguistik untuk mencapai tujuan manusia dalam hal bahasa yaitu menyelesaikan berbagai tugas atau aplikasi [1]. Untuk melakukan pemrosesan bahasa alami sangat membutuhkan namanya *dataset*. *Dataset* merupakan kumpulan data yang ditampung dalam tabel tunggal di mana setiap kolomnya mewakili nilai tertentu. *Dataset* biasanya digunakan dalam *model* pemrosesan bahasa alami ataupun pembelajaran mesin [2].

Ada beberapa situs sebagai penyedia *dataset*. Kaggle menyediakan banyak jenis *dataset* yang dapat diakses oleh semua pengguna, namun *dataset* untuk pemrosesan bahasa alami masih terbatas [11]. Maka untuk bisa melakukan pemrosesan bahasa alami dibuatlah dataset yang sesuai dengan model mereka yang dilakukan dengan proses anotasi. Dengan proses anotasi dapat menghasilkan entitas pada setiap informasi. Dengan WebAnno merupakan aplikasi alternatif yang membantu proses anotasi. Dengan menggunakan

pendekatan NER (*Named Entity Recognition*) aplikasi ini berhasil melakukan proses anotasi. NER adalah bagian dari proses *text mining* dan *natural language processing* yang sangat berguna untuk menemukan dan menentukan jenis *named entity* pada teks [3]. Namun dari aplikasi yang ada masih membuat pengguna kesulitan dalam proses anotasi seperti, proses instalasi yang rumit membuat proses anotasi menjadi terhambat. Selain itu tampilan yang disediakan terbilang rumit untuk digunakan proses anotasi membuat pekerjaan menjadi terhambat. Selain itu beratnya aplikasi dapat mengganggu dan menghambat proses anotasi dikarenakan lambatnya dalam mengolah informasi.

Maka dari masalah yang ada penelitian ini membuat aplikasi agar dapat menyelesaikan masalah pengguna yaitu dalam proses anotasi, dan aplikasi tersebut dinamakan AADT (Aplikasi Anotasi Data Teks). Untuk menyelesaikan masalah diatas maka dibuatlah aplikasi yang menerapkan PWA (Progressive Web Apps) untuk mengoptimalkan proses rendering content agar aplikasi dapat diakses oleh pengguna dengan cepat dan sedikit mengkonsumsi bandwidth dengan menggunakan *service worker*, selain itu ini juga akan memakai fungsi lainnya untuk meningkatkan kualitas pengguna, seperti *web app manifest*, dan *service worker* [13]. PWA (*Progressive Web Apps*) digambarkan sebagai kumpulan dari arsitektur aplikasi yang bekerja secara bersama untuk memberikan sentuhan aplikasi pada sebuah aplikasi web [14]. *Service worker* merupakan script yang berjalan di belakang pramban yang digunakan pengguna [6]. Selain itu aplikasi ini dibangun dengan menerapkan *information extraction* terhadap teks untuk mendukung pendekatan NER (*Named Entity Recognition*). *Information extraction* adalah proses cara mengubah teks tidak terstruktur menjadi informasi yang terstruktur [7].

2. Tinjauan Pustaka

Penelitian yang dilakukan Erick Alfons Lisangan adalah penelitian terhadap Universitas Atma Jaya Makassar untuk mengakomodasi perbedaan bahasa dari pengguna informasi dan komputer untuk memperoleh informasi akademik. Penelitian ini merancang NLP (*Natural Language Processing*) yang dapat mengolah informasi agar dapat memperoleh informasi akademik tanpa terkendala adanya perbedaan bahasa [8]. Dalam penelitian tersebut aplikasi berbasis web tersebut yang dibangung dengan bahasa pemrograman PHP dengan menggunakan DBMS yaitu MySQL. Antarmuka yang diberikan cukup sederhana yaitu berupa masukan dan tombol yang dapat memproses masukan pengguna dan akan memperoleh hasil pemrosesan dalam perintah SQL dan informasi akademik yang diharapkan.

Selanjutnya penelitian yang dilakukan Nisa Kurniasih Wangsenegara yaitu penelitian terhadap implementasi pemrosesan bahasa alami dalam pengukuran ketepatan ejaan EYD pada abstrak skripsi menggunakan *fuzzy logic*. Penelitian ini akan melakukan identifikasi dan menghitung jumlah kesalahan penulisan huruf kapital/kata dan tanda baca [9]. Dalam penelitian tersebut aplikasi yang dibuat dengan berbasis *web* dan dibangun dengan bahasa pemrograman PHP dan dengan menggunakan DBMS berupa MySQL. Antarmuka yang sederhana berupa *text area* yang digunakan untuk memasukan tulisan yang akan diidentifikasi. Selain berupa *text area*, *user* dapat menggunakan fitur *upload* file.

Selanjutnya penelitian yang dilakukan Richard Eckart De Castilho yaitu membangun sebuah aplikasi web untuk dapat melakukan proses anotasi yang dapat dibagikan. Aplikasi yang bernama WebAnno dibangun bertujuan untuk pemrosesan bahasa alami yang memungkinkan melakukan integrasi anotasi semantik dengan anotasi sintaksis [10]. Dalam penelitian memberikan antarmuka yang dapat mempermudah proses anotasi, yaitu berupa *multi-layer annotation* yang cukup efisien dalam proses anotasi, yang dapat membantu menggabungkan dengan skema anotasi yang berbeda. Selain itu *tooltip* yang dapat menampilkan informasi yang dapat membantu dalam proses anotasi [10].

Dengan adanya tinjauan pustaka tersebut dalam penelitian ini, penulis akan melakukan pembangunan aplikasi yang dapat melakukan proses anotasi data teks, yang dibangun dengan bahasa pemrograman *javascript*, menggunakan DBMS MySQL dan menerapkan PWA (*Progressive Web Apps*) untuk dapat melakukan caching terhadap halaman yang diakses. Dengan demikian penulis memperoleh pengetahuan lebih untuk membantu meningkatkan kualitas aplikasi.

3. Metodologi Penelitian

Metode yang digunakan dalam pembangunan perangkat lunak tersebut menggunakan waterfall. Metode ini sangat cocok untuk pengembangan perangkat lunak dengan spesifikasi kebutuhan perangkat lunak yang telah dibuat. Dalam model tersebut, menyediakan beberapa tahap dari sebuah pembangunan perangkat lunak yang dilakukan secara berurut yaitu analisis kebutuhan perangkat lunak, perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak.

Pada tahap analisis kebutuhan perangkat lunak,setiap fitur yang nantinya akan dibuat, diidentifikasi, dan difokuskan agar sesuai untuk proses anotasi. Pengembang melakukan identifikasi perangkat lunak untuk menentukan fitur yang sesuai dengan kebutuhan pada perangkat lunak ataupun perangkat keras yang digunakan. Tahap selanjutnya dari analisis ialah perancangan perangkat lunak. Pada tahap ini penjabaran terhadap setiap fungsi dan fitur yang ada pada perangkat lunak dilakukan sehingga nantinya pengembang dapat mengetahui setiap aspek yang ada pada perangkat lunak yang dibangun. Tahap ini menghasilkan rancangan desain aplikasi dalam bentuk mockup sehingga bisa terfokus pada pembuatan tampilan sesuai dengan proses anotasi dan memaksimalkan fitur pada perangkat lunak yang akan dibuat.

Selanjutnya adalah tahapan implementasi perangkat lunak (*coding)* dalam pembuatan *backend* dan *frontend*, serta perancangan basis data pada aplikasi ini. Aplikasi yang digunakan pada tahap pembangunan perangkat lunak tersebut, yaitu Visual Studio Code, ExpressJS, TypeORM, ReactJS, dan MySQL.

Tahap yang terakhir ialah pengujian perangkat lunak. Tahap ini melakukan uji *testing* perangkat lunak terhadap setiap fungsi yang sudah dibuat. Tujuannya ialah memperoleh kesesuaian fungsionalitas serta memaksimalkan hasil pada perangkat lunak. Pengujian ini dapat dilakukan secara sistematis sehingga dapat memperoleh hasil yang baik.

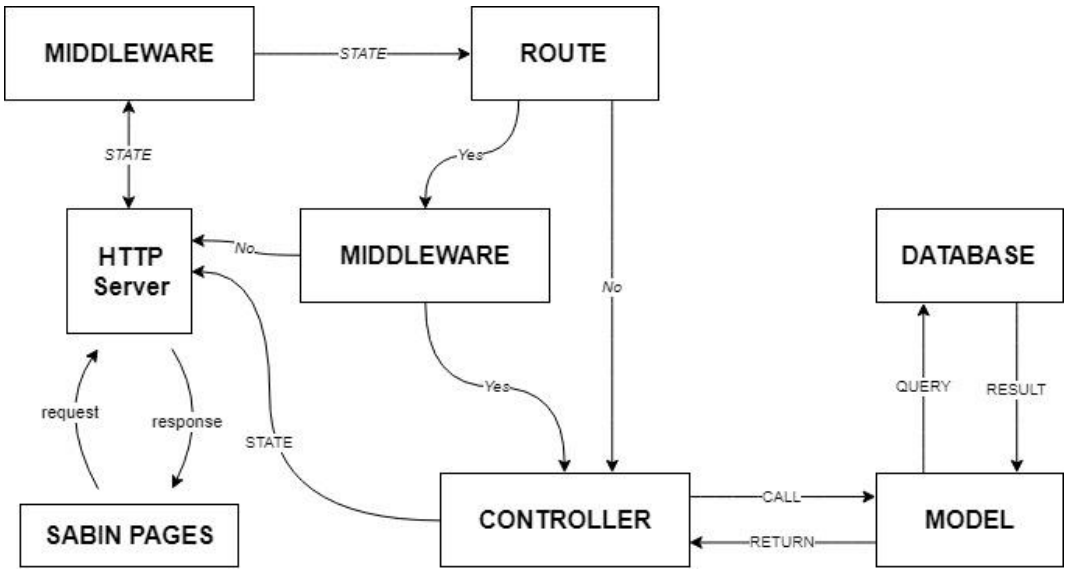
4. Hasil dan Diskusi

Aplikasi anotasi data teks dibangun pada aplikasi berbasis web. Sistem ini mampu melakukan anotasi data teks. Implementasi sistem akan mendeskripsikan beberapa bagian penting dalam aplikasi ini. Berikut arsitektur dan implementasi yang diterapkan pada aplikasi ini.

4.1. Arsitektur Sistem

4.1.1. Arsitektur API

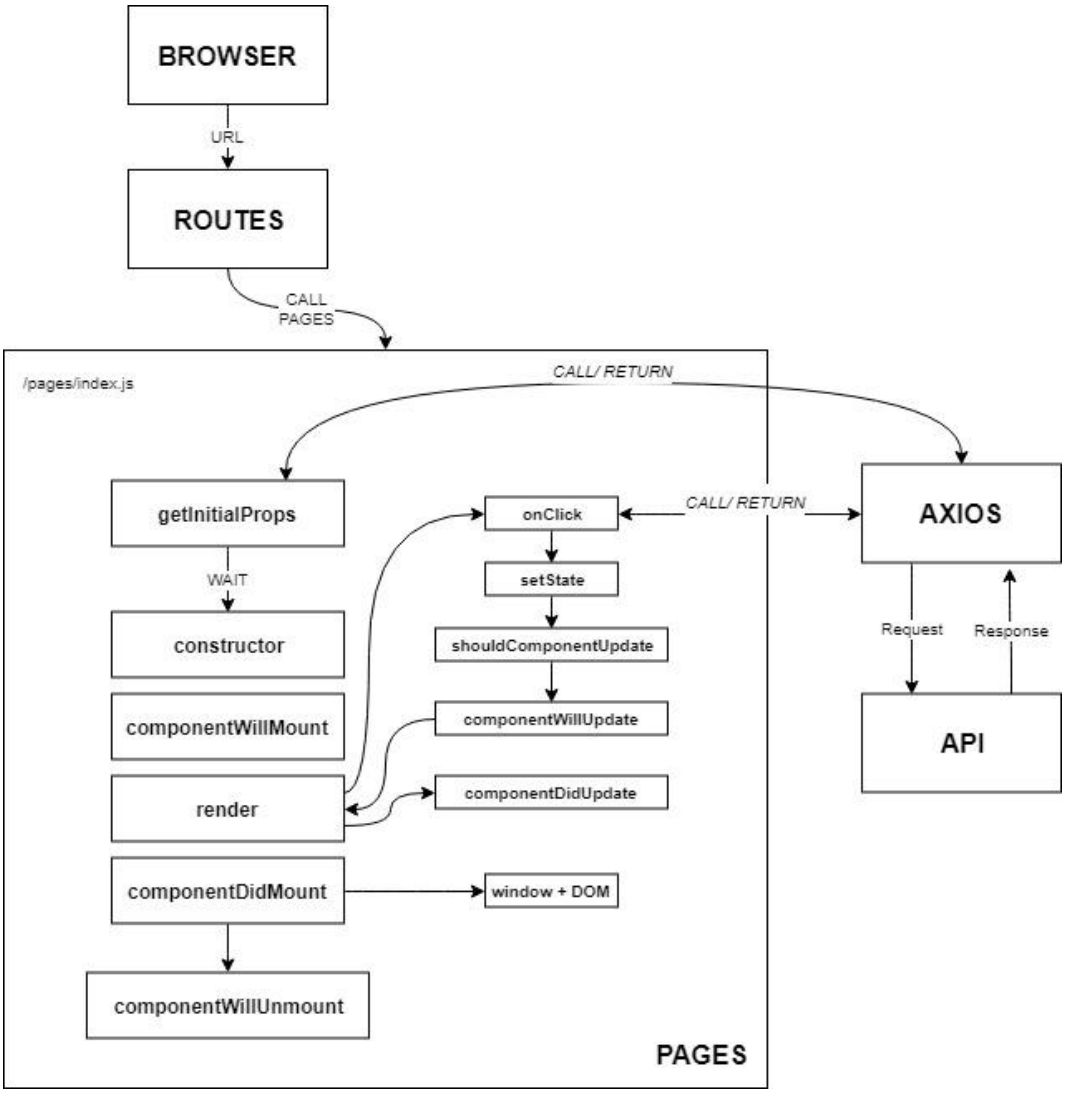
Hasil RESTful API dibuat dengan menggunakan bahasa *javascript* yang dibantu dengan *framework* ExpressJS, adapun *library* yang digunakan untuk membantu proses pengelolaan basis data yaitu dengan TypeORM. TypeORM membantu aplikasi agar dapat memanipulasi basis data dengan mudah, di mana *query* sudah disiapkan dalam bentuk fungsi. Pada gambar 1 merupakan hasil arsitektur API pada AADT.



Gambar 1. Arsitektur API

4.1.2. Arsitektur Pages

ReactJS ini digunakan untuk membuat komponen Pages. ReactJS merupakan framework dari javascript yang membantu dalam pembuatan halaman web dengan bahasa pemrograman *javascript* [15]. Pages merupakan antarmuka yang ditampilkan saat pengguna mengakses aplikasi. Pages dibuat dengan menggunakan bahasa pemrograman *javascript*, adapun *library* yang membantu aplikasi untuk dapat mengakses API yaitu dengan *axios*. Axios berfungsi sebagai *http-client* untuk browser dan NodeJS. Pada gambar 2 merupakan arsitektur pada Pages.



Gambar 2. Arsitektur Pages

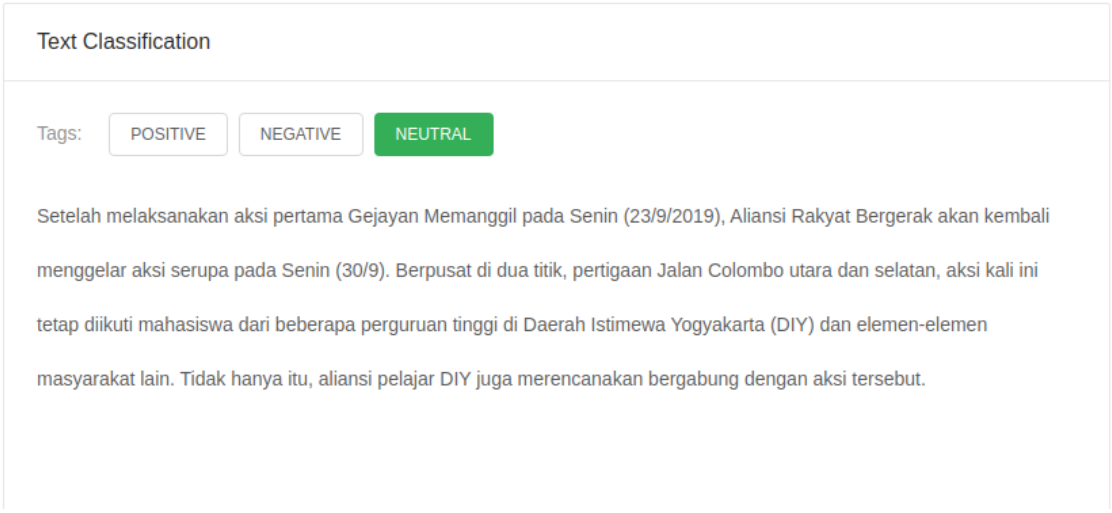
4.2. Implementasi Fitur

4.2.1. Fitur Anotasi

Pada bagian hasil antarmuka anotasi, akan dibahas proses pengelolaan anotasi pada AADT. Aplikasi ini juga mendukung beberapa anotasi yaitu *classifier*, *extractor*, dan *pattern extractor*. Setiap anotasi dapat dilakukan dengan memilih jenis anotasi yang ingin dipakai. Fungsi *classifier* melakukan anotasi berdasarkan *source* yang ingin diberi entitas, sedangkan fungsi *extractor* melakukan anotasi terhadap kalimat yang ingin diberi entitas, dan *pattern extractor* melakukan anotasi terhadap kalimat yang dipilih lalu diberi entitas. Berikut merupakan penjelasan bagaimana proses implementasi anotasi *classifier*, *extractor*, dan *pattern extractor*. Berikut penjelasan perbedaan dari anotasi yang disediakan.

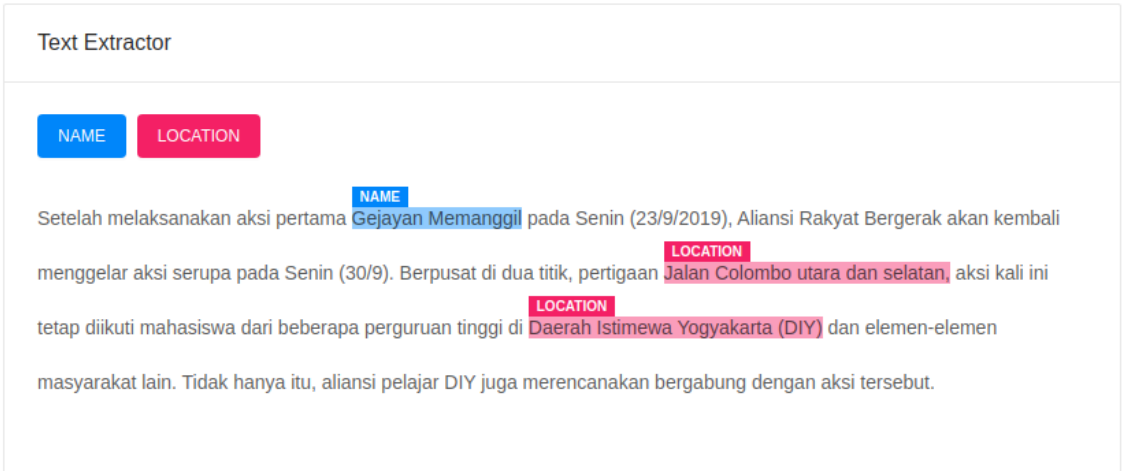
Tabel 1. Tabel Deskripsi Anotasi

Tipe	Deskripsi
Classifier	Anotasi dengan *classifier* merupakan anotasi yang berjenis klasifikasi terhadap teks dengan menentukan label pada *source*, biasanya digunakan pembuatan dataset untuk pemrosesan bahasa alami analisis sentimen.
Extractor	Anotasi dengan *extractor* merupakan anotasi yang dapat melakukan pemberian label pada sebuah kata ataupun kalimat yang diinginkan, biasannya anotasi ini digunakan untuk pembuatan dataset pada pemrosesan bahasa alami *information extraction.*
Pattern Extractor	Anotasi dengan *pattern extractor* merupakan anotasi yang dapat memilih kalimat pada *source* dan menentukan teks yang akan digunakan untuk anotasi dan ingin diberi label, contoh penggunaanya sama seperti *extractor* perbedaanya hanya dapat melakukan pemilihan teks yang digunakan



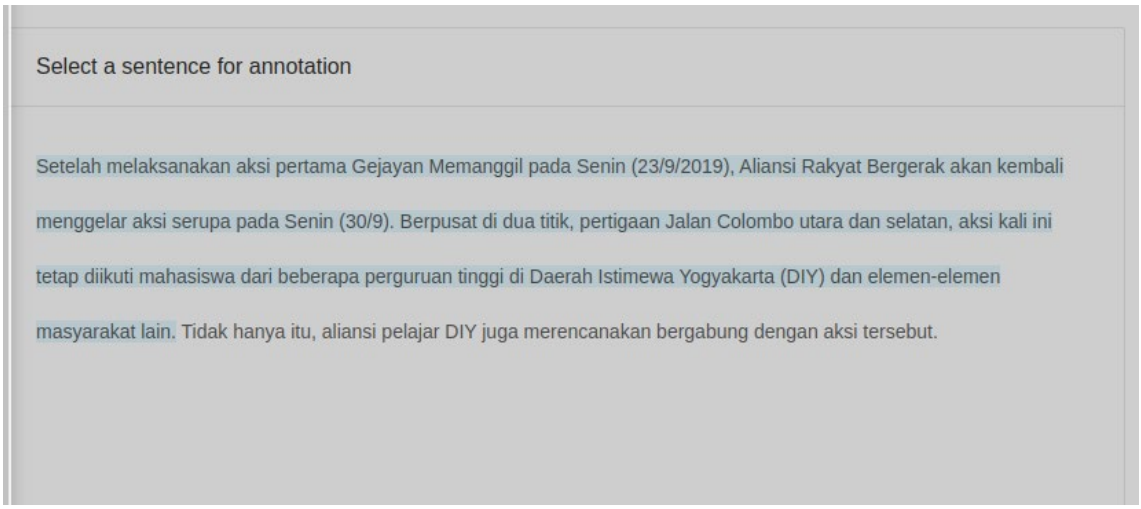
Gambar 3. Antarmuka Anotasi *Classifier*

Gambar 3 menunjukan bagaimana antarmuka dari anotasi *classifier*. Anotasi ini memberi label pada teks yang diberikan dengan tujuan untuk melakukan klasifikasi apakah teks berikut *positive*, *negative*, atau *neutral*.



Gambar 4. Antarmuka Anotasi *Extractor*

Gambar 4 menunjukan bagaimana anotasi *extractor* bekerja, dimana pengguna dapat melakukan seleksi pada teks yang ingin diberi label. Terdapat 2 label pada *model* ini yang bisa dipakai, yaitu *name* dan *location*. Proses anotasi ini mengandalkan *text selection* untuk menandai teks yang ingin diberi label sistem, lalu mendapatkan lokasi dari *index* yang ditandai.



Gambar 5. Antarmuka Anotasi *Pattern Extractor Yang Teks Terpilih*

Gambar 6. Antarmuka Anotasi *Pattern Extractor Yang Teks Ingin Dilakukan Anotasi*

Gambar 5 dan 6 menunjukan bagaimana antarmuka anotasi *pattern extractor* tertampil dan bekerja. Dengan ini pengguna dapat melakukan memilih teks yang ingin dilakukan anotasi, dan dapat melakukan *grouping* anotasi berdasarkan teks yang diseleksi diawal. Dalam proses penandaan teks sangat menyerupai dengan *text extractor* hanya memiliki hasil anotasi yang berpola. Adapun beberapa fungsi yang disediakan untuk membantu proses anotasi, yaitu:

1. ***Import Spreadsheet***

Untuk dapat melakukan proses anotasi aplikasi membutuhkan *source* yang berisi kumpulan teks ataupun berita. Untuk memudahkan pengguna untuk memasukan kedalam aplikasi, maka aplikasi mendukung pengambilan *source* dengan *file .xls* ataupun *.csv* agar data semakin mudah tersortir, dan dimana setiap cell pada file *.xls* ataupun *.csv* mewakili satu berita/kalimat.

2. ***Export Annotation***

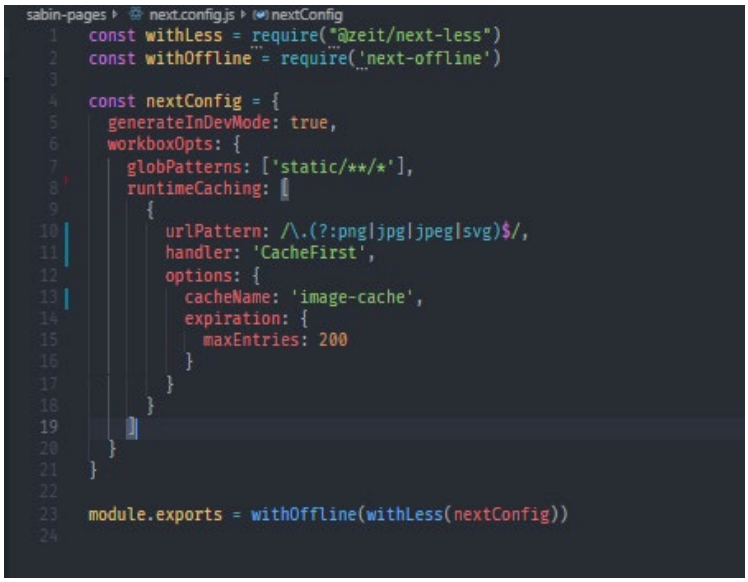
Dengan proses anotasi yang sudah dilakukan pastinya pengguna ingin mengambil hasil anotasi mereka kembali untuk dapat digunakan pada model pemrosesan bahasa alami mereka. Dengan itu aplikasi menyiapkan fungsi *export* untuk mengolah data anotasi yang sudah dilakukan oleh pengguna dan bisa diunduh dan menghasilkan *file .csv* yang isinya sesuai dengan label dan hasil mereka.

3. ***Model Sharing***

Dengan melakukan pembuatan dataset yang akan memakan waktu pengguna. Dengan model sharing membuat pengguna lain dapat berkontribusi dalam model yang dimiliki dengan tujuan mempercepat proses anotasi tanpa harus takut adanya kehilangan data. Dengan dibantu fitur pengamanan pada model, membuat data yang dapat ditampilkan pada model dapat atur oleh pengguna sendiri, seperti: informasi lengkap, dataset, verifikasi anotasi, dan statistik.

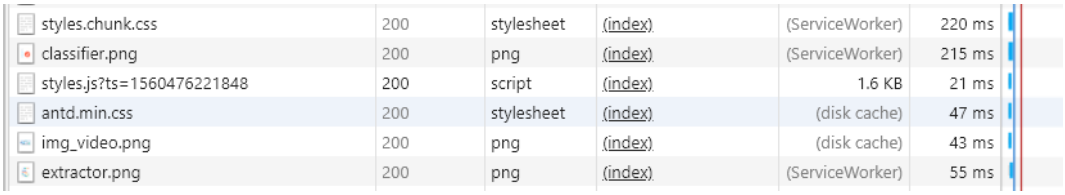
4.2.2. Implementasi PWA

Untuk hasil PWA pada AADT, ada beberapa *library* yang membantu agar ReactJS dengan PWA bisa berjalan dengan baik yaitu *next-offline* dan *workbox*. Pada *next-offline* yaitu digunakan untuk mengaktifkan *service worker* pada aplikasi. Lalu pada *workbox* merupakan *library* yang membantu pembuatan config *service worker* secara otomatis [12]. *Service worker* memampukan aplikasi dapat penyimpanan assets pada browser yang berfungsi untuk meringankan aplikasi saat diakses.



Gambar 7. Potongan Kode: Pemasangan PWA

Gambar 7 merupakan implementasi PWA. Pada variabel *workboxOpts* berisi fitur yang ingin diaktifkan pada workbox. Lalu pada *runtimeCaching* melakukan *caching* berdasarkan *handler* yang dipilih dan ini bekerja saat browser mengakses *file* gambar lalu membuat aplikasi melakukan *caching* jika *file* belum pernah diakses oleh browser.



Gambar 8. Hasil Progressive Web Apps

Gambar 8 merupakan hasil dari PWA pada Pages. Pada pengolahan *file* yang berada folder static akan disimpan secara otomatis pada pramban dan hasilnya setiap pengaksesan *file* pada kolom *size* akan berisi *(ServiceWorker)*. Pada saat browser mengakses *file* gambar akan melakukan melakukan *caching* jika *file* belum pernah diakses oleh browser.

5. Kesimpulan

Berdasarkan hasil analisis, perancangan, implementasi hingga pengujian sistem yang telah dibuat, diperoleh kesimpulan yaitu aplikasi ini berhasil dapat membantu proses anotasi data teks, dimana dengan menerapkan NER (*Named Entity Recognition*) dapat membantu dalam memilih sebuah teks yang ingin dilakukan anotasi. Lalu untuk mengatasi masalah proses eksekusi yang berat dalam proses anotasi membuat aplikasi ini dibangun dengan ReactJS, dengan ini membuat aplikasi menjadi lebih ringan, stateful, dan *reusable.* Dan masalah pengelolaan data yang besar membuat aplikasi melambat bisa dibantu PWA (*Progressive Web Apps*) membuat proses mengelola *caching files*.

6. Saran

Dari proses analisis, perancangan, implementasi hingga pengujian sistem pada pembuatan penelitian, didapatkan beberapa saran untuk pengembangan lebih lanjut dari aplikasi anotasi data teks ini. Untuk kebutuhan anotasi sebagai penyempurnaan maupun penambahan fitur-fitur yang bisa meningkatkan kinerja aplikasi dan proses anotasi, seperti *relation entity* yang mendukung pada fitur anotasi untuk menghubungkan setiap *named entity* yang saling berhubungan. Selain itu dapat mengimplementasikan *database offline* agar aplikasi dapat berjalan secara *offline* dengan maksimal.

Referensi

[1] A. Wibowo and S. Hartati, “Aplikasi Pengolah Bahasa Alami untuk Query Basis Data Jalan dan Lalu Lintas dalam Format XML,” *J. Teknol. Inf. Din.*, vol. 18, no. 1, pp. 65–79, 2013.

[2] A. Rachmat and Y. Lukito, “Sentipol : Dataset Sentimen Komentar Pada Kampanye Pemilu Presiden Indonesia 2014 Dari Facebook Page,” *Konf. Nas. Teknol. Inf. dan Komun. (KNASTIK 2016)*, no. December, 2016.

[3] D. W. Wulandari, P. P. Adikara, and S. Adinugroho, “Named Entity Recognition (NER) Pada Dokumen Biologi Menggunakan Rule Based dan Naïve Bayes Classifier,” *J. Pengemb. Teknol. Inf. dan Ilmu Komput. Univ. Brawijaya*, vol. 2, no. 11, pp. 4555–4563, 2018.

[4] H. Hamad, M. Saad, and R. Abed, “Performance Evaluation of RESTful Web Services,” *International Arab Journal of e-Technology*, vol. 1, no. 3, pp. 72–78, 2010.

[5] A. Kumar and R. K. Singh, “COMPARATIVE ANALYSIS OF ANGULARJS AND REACTJS,” *International Journal of Latest Trends in Engineering and Technology*, vol. 7, no. 4, pp. 225–227.

[6] Google Developer, “Web Fundamentals,” *Google*, 2018. [Online]. Available: https://developers.google.com/web/fundamentals/. [Accessed: 16-Nov-2018].

[7] E. Susanti and K. Mustofa, “Ekstraksi Informasi Halaman Web Menggunakan Pendekatan Bootstrapping pada Ontology-Based Information Extraction,” *IJCCS (Indonesian J. Comput. Cybern. Syst.*, vol. 9, no. 2, p. 111, 2017.

[8] E. A. Lisangan, F. T. Informasi, U. Atma, and J. Makassar, "Natural Language Processing Dalam Memperoleh Informasi Akademik Mahasiswa Universitas Atma Jaya Makassar," *J. Temat.*, vol. 1, no. May, pp. 1–9, 2013.

[9] N. K. Wangsanegara and B. Subaeki, "IMPLEMENTASI NATURAL LANGUAGE PROCESSING DALAM PENGUKURAN KETEPATAN EJAAN YANG DISEMPURNAKAN (EYD) PADA ABSTRAK SKRIPSI MENGGUNAKAN ALGORITMA FUZZY LOGIC," *J. Tek. Inform.*, vol. 8, no. 2, pp. 1–6, 2015.

[10] Y. Seid Muhie, G. Iryna, R. Eckart de Castilho, and C. Biemann, "WebAnno: A Flexible, Web-based and Visually Supported System for Distributed Annotations," *Ger. Inst. Educ. Res. Educ. Inf.*, pp. 1–6, 2013.

[11] Kaggle Documentation, "Notebooks Documentation," Kaggle, 2019. [Online]. Available: https://www.kaggle.com/docs/kernels/. [Accessed: 18-Oct-2019].

[12] Google Developers, "Workbox," Google, 2019. [Online]. Available: https://developers.google.com/web/tools/workbox. [Accessed: 28-Oct-2019].

[13] Google Developer, "Progressive Web Apps," *Google*, 2018. [Online]. Available: https://developers.google.com/web/progressive-web-apps/checklist. [Accessed: 02-Nov-2018]

[14] R. S. Mishra, "Progressive WEBAPP : Review," *Int. Res. J. Eng. Technol.*, vol. 3, no. 6, pp. 3028–3032, 2016.

[15] J. Voutilainen, "Evaluation of Front-end JavaScript Frameworks for Master Data Management Application Development," no. December, 2017.