

RANCANG BANGUN REST API APLIKASI MANAJEMEN TOKO MENGUNAKAN NODEJS PADA CANTIKA PAINT

Muhammad Fiqri Nugroho, Aji Primajaya, Mohamad Jajuli

Informatika, Universitas Singaperbangsa Karawang

Jl. HS. Ronggo Waluyo, Puseurjaya, Telukjambe Timur, Karawang, Jawa Barat, Indonesia

1910631170212@student.unsika.ac.id

ABSTRAK

Cantika Paint adalah toko cat kiloan yang terletak di Karawang, Jawa Barat, Indonesia, dengan total lima cabang yang tersebar di Karawang. Saat ini, pemrosesan data, seperti pelaporan penjualan, dan pemeriksaan stok masih dilakukan secara manual oleh karyawan. Hal ini sangat tidak efisien dan berisiko menyebabkan kesalahan. Pengiriman laporan penjualan secara manual yang dilakukan setiap hari membuat para karyawan kesulitan. Oleh karena itu, sistem REST API aplikasi manajemen toko menggunakan nodeJS telah dibuat dengan pengelolaan dan pemrosesan data disimpan dalam sistem basis data yang bertujuan untuk mencegah redundansi data yang menyebabkan kerugian bagi Cantika Paint dan untuk mempermudah melihat informasi data yang diolah. Metode pengembangan perangkat lunak yang digunakan dalam pengembangan aplikasi ini adalah SDLC (Siklus Hidup Pengembangan Sistem) dengan model air terjun (*waterfall model*). Sistem ini dikembangkan menggunakan bahasa pemrograman nodeJS dengan *framework* expressJS sebagai *backend* dan *API services*, kemudian menggunakan *view engine* EJS (*Embedded JavaScript*) sebagai *front-end*, dan *postgresql* sebagai *database*. Hasil dari pengembangan sistem mampu mengelola data penjualan, stok barang, dan pengiriman lebih aman dan lebih akurat.

Kata kunci: REST API, NodeJS, ExpressJS, PostgreSQL, Metode Waterfall

1. PENDAHULUAN

Perkembangan pada bidang teknologi dan informasi berkembang dengan sangat cepat khususnya dalam pengembangan teknologi komputer untuk berbagai bidang namun yang paling populer adalah pemanfaatan dalam bidang pengolahan data untuk membantu berbagai kegiatan manusia sehari-harinya. Terdapat banyak aktivitas yang dapat diubah menjadi sebuah sistem yang dapat dijalankan menggunakan perangkat komputer, salah satunya adalah mengembangkan sistem untuk bisnis, seperti mengolah data penjualan barang, inventarisasi stok di gudang, dan pengolahan data pembelian barang dari vendor di perusahaan. Cantika Paint merupakan toko yang menjual cat untuk tembok dengan berbagai warna dan jenis. Saat ini toko Cantika Paint sudah memiliki lima cabang yang tersebar di kabupaten karawang, meskipun begitu toko ini masih menggunakan sistem pelaporan manual dengan cara ditulis untuk stok dan hasil penjualan setiap harinya. Dari hasil wawancara terhadap pemilik toko dan lima pegawai didapatkan beberapa kendala dalam proses pengolahan data yang ada, yaitu sulitnya mencari data yang dibutuhkan dan sering kali terjadi kehilangan data karena hilangnya kertas yang digunakan untuk menyimpan data stok dan penjualan, sehingga terpaksa harus melakukan perhitungan ulang barang secara manual. Proses pengolahan data yang masih dilakukan secara manual ini memerlukan waktu yang cukup lama dan menghasilkan akurasi yang kurang memadai, sehingga efisiensi pengolahan data menjadi terhambat. Sedangkan pengolahan data menggunakan komputer dapat mengelola data menjadi lebih efisien. Penerapan komputer dalam sistem informasi adalah langkah yang

tepat terutama jika dihubungkan dengan pengolahan data transaksi [1]. Berdasarkan variabel-variabel tersebut pada penelitian ini akan mengembangkan sebuah aplikasi untuk mengkomputerisasi pengolahan data dari toko Cantika Paint, dimana dengan pembuatan aplikasi ini diharapkan dapat membantu toko Cantika Paint agar dapat memanajemen toko lebih baik, serta mengurangi risiko kehilangan data. Dalam penelitian ini, aplikasi manajemen toko akan dikembangkan dengan menggunakan teknologi *Application Programming Interface* (API) sebagai *Backend Development* yang akan diterapkan pada tampilan *website*, dengan tujuan untuk memudahkan pengembangan aplikasi tersebut. Dengan menggunakan API sebagai *backend* aplikasi ini dapat memiliki lebih dari satu tampilan *frontend* yang terhubung dengan satu *backend*, misalkan toko Cantika Paint membutuhkan aplikasi manajemen toko berbasis android maka sistem API ini dapat digunakan dalam pembangunannya dengan begitu yang perlu dibuat hanya *frontend* saja karena untuk sisi *backend* dapat menggunakan API yang telah dibuat. Selain itu API pada aplikasi ini juga dapat digunakan pada aplikasi lainnya untuk toko Cantika Paint contohnya jika ingin membuat aplikasi pemesanan secara online maka API pada aplikasi ini mampu menyediakan data barang dan stok yang ada secara *real-time*.

2. TINJAUAN PUSTAKA

2.1. *Application Programming Interface* (API)

API merupakan suatu *interface* yang berfungsi untuk mengintegrasikan data dan menghubungkan aplikasi yang berjalan di berbagai *platform* agar dapat saling terhubung. Dengan menggunakan API,

pengguna dapat bertukar data di berbagai *platform* yang berbeda dan juga mempercepat proses pengembangan aplikasi dengan menyediakan fungsi-fungsi terpisah sehingga para pengembang tidak perlu membuat fitur yang sama dari awal.

Representational State Transfer atau biasa disingkat menjadi REST yang merupakan bagian dari web API adalah sebuah gaya arsitektur yang dapat diterapkan dalam membantu membuat dan mengatur sistem yang terdistribusi [2]. REST API merupakan sebuah API yang dibangun berdasarkan aturan atau gaya arsitektur REST, dan memungkinkan interaksi dengan layanan web yang mematuhi prinsip RESTful.

2.2. NodeJS

NodeJS adalah *platform* pengembangan aplikasi berbasis *server* yang berjalan di atas mesin *JavaScript* Google V8 yang akan menjalankan kode *JavaScript* secara *server-side*. NodeJS dapat membuat pengembangan aplikasi web dengan cepat karena didukung dengan model pemrograman *event-driven*, *non-blocking I/O (asynchronous)*,

NodeJS memiliki *library* yang sangat banyak dan beragam seperti *expressJS*. *expressJS* adalah *backend web application framework open source* di bawah lisensi MIT untuk *nodeJS*, *expressJS* di desain untuk membangun *web application* dan API [3].

2.3. PostgreSQL

PostgreSQL merupakan sistem manajemen basis data relasional (RDBMS) *open source* yang dikembangkan oleh *Berkeley Computer Science Department*. PostgreSQL adalah *database* yang banyak digunakan pada *web app*, aplikasi *mobile*, dan aplikasi *analytics* [4].

PostgreSQL atau umumnya dikenal sebagai Postgres adalah salah satu dari banyak *database* hebat yang menawarkan skalabilitas, fleksibilitas, dan kinerja tinggi. PostgreSQL mendukung SQL (*Structured Query Language*) yang dapat melakukan *transactions*, *subqueries*, *triggers*, dan lain-lain [8]. Bagi masyarakat TI (Teknologi Informasi) di Indonesia, Postgres telah digunakan pada berbagai aplikasi sistem informasi diantaranya pada *web*, *billing system*, dan sistem informasi besar lainnya.

2.4. View Engine EJS (Embedded JavaScript)

EJS (*Embedded JavaScript*) adalah *library* template *engine* yang digunakan pada aplikasi web dengan menggunakan bahasa pemrograman *nodeJS*. EJS memungkinkan pengembang untuk menjalankan kode *JavaScript* ke dalam kode HTML yang dapat di-render di sisi *server* sebelum dikirimkan ke klien [9].

Dengan menggunakan EJS, pengembang dapat membuat tampilan *website* yang dinamis dan bisa berubah sesuai dengan data yang dikirimkan oleh *server*. EJS memudahkan pengguna untuk menambahkan variabel atau data dinamis ke dalam template HTML dengan menambahkan *sintaks* tertentu di dalam tag HTML seperti `<% %>`, `<%= %>`,

dan `<%# %>`. EJS juga memungkinkan pengembang untuk membuat struktur template yang terpisah dari file *JavaScript* yang berisi logika program.

2.5. Swagger

Swagger merupakan sebuah *framework* untuk membuat, mendokumentasikan, dan menguji Web API. *Framework* ini mempermudah developer dan pengguna API untuk memahami cara menggunakan dan memanggil API dengan lebih mudah. Swagger menggunakan standar OpenAPI untuk mendeskripsikan API. OpenAPI merupakan sebuah standar untuk mendefinisikan RESTful API menggunakan format JSON atau YAML, yang bertujuan untuk menghasilkan API dengan dokumentasi yang terstruktur dan lebih mudah dibaca oleh manusia.

Swagger juga memungkinkan integrasi dengan *tools* lain seperti Postman, git, dan Jenkins. Swagger bersifat *real-time* sehingga klien dan sistem dokumentasi bergerak bersamaan dengan *server*. Deskripsi metode, parameter, dan model terintegrasi erat kedalam kode *server*, sehingga sinkronisasi API dan dokumentasinya dapat terjaga [6].

2.6. Black Box Testing

Black box testing merujuk pada pengujian terhadap fungsi-fungsi dalam aplikasi guna mengetahui apakah sistem sudah sesuai dengan skenario uji yang telah dipersiapkan [7]. *Black box testing* dilakukan dengan fokus pada masukan (*input*) dan keluaran (*output*) yang dihasilkan oleh program pada berbagai situasi dan kondisi.

Pengujian yang dilakukan pada *black box testing* dilakukan dengan cara memberikan masukan tertentu pada program dan mengecek keluaran yang di hasilkan. Tujuannya adalah untuk menemukan kesalahan atau *bug* dalam perangkat lunak dengan melihat bagaimana program merespon masukan yang diberikan.

3. METODE PENELITIAN

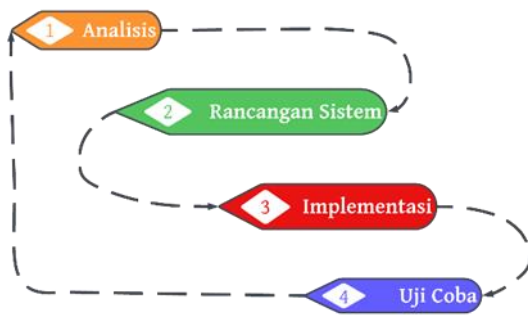
3.1. Metode Pengumpulan Data

Metode pengumpulan data yang digunakan pada penelitian ini yaitu wawancara dan observasi. Wawancara dilakukan kepada pemilik dan lima pegawai, untuk menemukan masalah yang dihadapi oleh Toko Cantika Paint. Kemudian observasi dilakukan dengan mengamati sistem yang saat ini sedang berjalan untuk memahami sistem yang dibutuhkan, tujuannya adalah untuk memberikan landasan yang solid dalam merekomendasikan solusi sistem yang sesuai dengan kebutuhan dan mampu mengatasi masalah yang dihadapi Toko Cantika Paint.

3.2. Metode Pengembangan Sistem

Penelitian ini menggunakan metode *waterfall* untuk melakukan pengembangan sistem, metode ini bersifat serial dan mengikuti urutan tahapan. Tahapan

SDLC *waterfall* yang digunakan pada penelitian ini dapat dilihat pada gambar di bawah ini :



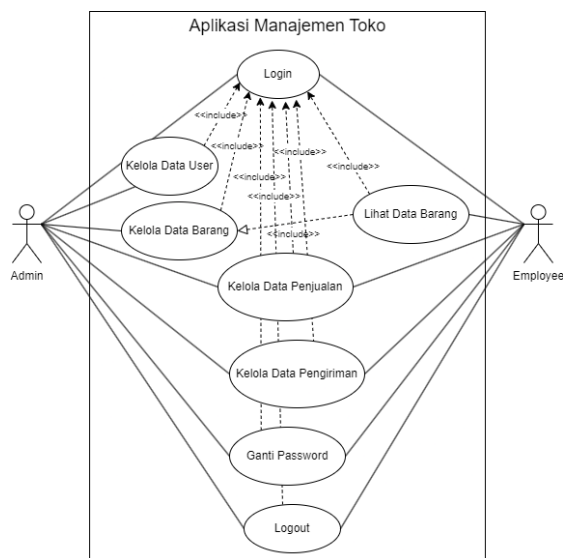
Gambar 1. Metode penelitian model *waterfall*

Metode SDLC *waterfall* merupakan sebuah pendekatan yang terstruktur, dimulai dari tahap analisis, rancangan sistem, implementasi, dan uji coba/testing [5]. Model ini juga menekankan pentingnya dokumentasi sehingga cocok untuk proyek yang mengedepankan kualitas [10].

3.3. Desain Sistem

3.3.1. Use Case Diagram

Untuk memodelkan fungsionalitas sistem berdasarkan kebutuhan pengguna dibuatlah sebuah *use case diagram* yang akan memberikan gambaran visual tentang bagaimana aktor-aktor berinteraksi dengan sistem dalam skenario yang berbeda.

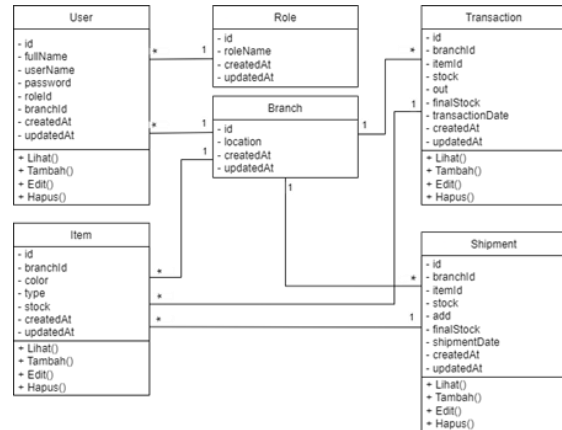


Gambar 2. Use case diagram

Berdasarkan hasil dari analisis sistem yang saat ini berjalan, sistem membutuhkan dua aktor, yaitu admin (pemilik) dan *employee* (pegawai) dengan hak akses yang berbeda.

3.3.2. Class Diagram

Class diagram mengilustrasikan struktur sistem yang telah dibentuk di dalam basis data serta relasi antara kelas-kelas tersebut. *Class diagram* yang akan menjadi gambaran untuk pembuatan *database*.



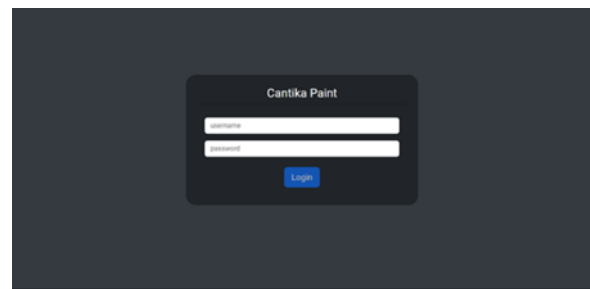
Gambar 3. Class diagram

4. HASIL DAN PEMBAHASAN

Dalam penelitian ini menghasilkan aplikasi manajemen Toko Cantika Paint untuk karyawan dan pemilik yang dapat mengelola data barang serta melakukan pelaporan untuk penjualan maupun pengiriman barang, sehingga pengolahan data bisa lebih aman dan efisien.

4.1. Tampilan Halaman Login

Halaman *login* berfungsi untuk masuk kedalam aplikasi agar pengguna dapat mengakses fitur yang ada pada aplikasi berdasarkan hak akses yang dimiliki. Untuk masuk atau *login* ke aplikasi ini dilakukan dengan cara pengguna mengisi *username* dan *password* pada *form login* yang ada pada halaman ini, kemudian sistem akan melakukan validasi jika data yang di isi valid maka sistem akan menampilkan halaman utama, jika tidak valid maka akan muncul pesan gagal.

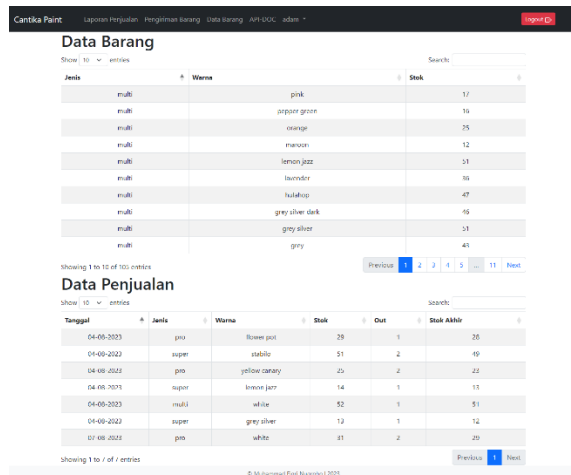


Gambar 4. Tampilan halaman *login*

4.2. Tampilan Halaman Utama

Pada tampilan halaman utama terdapat perbedaan antara pengguna dengan hak akses admin dan pengguna dengan hak akses pegawai yaitu pada bagian navbar untuk admin terdapat tambahan menu edit *user*, selain itu data yang di tampilkan untuk admin dan pegawai juga memiliki perbedaan untuk admin dapat melihat seluruh data barang dan data penjualan dari semua cabang sedangkan untuk pegawai hanya dapat melihat data dari cabang tokonya masing-masing, kemudian untuk data yang di tampilkan dapat di urutkan sesuai kebutuhan serta pengguna dapat

melakukan pencarian atau filter dengan memasukan kata kunci pada bagian *searchbar*.



Data Barang

Jenis	Warna	Stok
multi	pink	17
multi	pepper green	16
multi	orange	25
multi	maroon	12
multi	lemon jazz	11
multi	lavender	16
multi	hulalop	47
multi	grey silver dark	16
multi	grey silver	11
multi	grey	41

Showing 1 to 10 of 101 entries

Data Penjualan

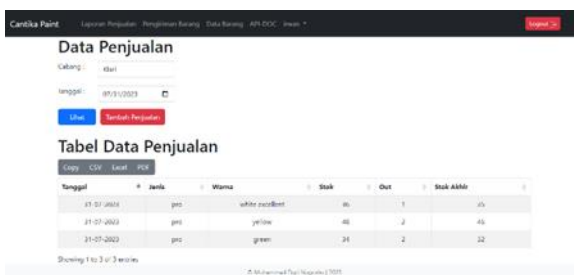
Tanggal	Jenis	Warna	Stok	Out	Stok Akhir
04-06-2023	pro	Brown pot	28	1	28
04-09-2023	super	stabilo	51	2	49
04-09-2023	pro	yellow candy	25	2	23
04-09-2023	super	limestone jar	14	1	13
04-09-2023	multi	white	52	1	51
04-09-2023	super	grey silver	13	1	12
07-08-2023	pin	white	31	2	29

Showing 1 to 7 of 7 entries

Gambar 5. Tampilan halaman utama

4.3. Tampilan Halaman Laporan Penjualan

Halaman ini menampilkan data laporan penjualan pada hari ini atau sesuai dengan cabang dan tanggal yang dipilih. Data yang di tampilkan dapat di *export* sesuai dengan *format file* yang di butuhkan.



Data Penjualan

Cabang: idari

Tanggal: 07/07/2023

Tabel Data Penjualan

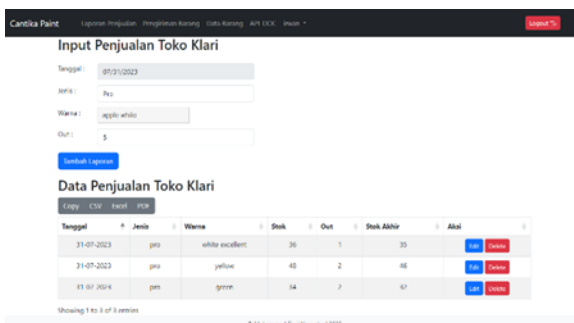
Tanggal	Jenis	Warna	Stok	Out	Stok Akhir
31-07-2023	pro	white excellent	46	1	45
31-07-2023	pro	yellow	48	2	46
31-07-2023	pro	green	34	2	32

Showing 1 to 3 of 3 entries

Gambar 6. Tampilan halaman laporan penjualan

4.4. Tampilan Halaman Input Penjualan

Setelah mengklik tombol tambah penjualan pada halaman laporan penjualan, maka sistem akan membawa pengguna ke halaman *input* penjualan. Pada halaman ini pengguna dapat menambahkan laporan penjualan dengan mengisi jenis, warna, dan jumlah barang keluar. Selain itu pengguna dapat menghapus atau mengedit data yang ada.



Input Penjualan Toko Klari

Tanggal: 07/07/2023

Jenis: pro

Warna: apple white

Out: 5

Data Penjualan Toko Klari

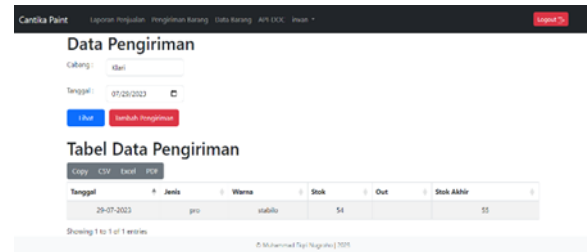
Tanggal	Jenis	Warna	Stok	Out	Stok Akhir	Aksi
31-07-2023	pro	white excellent	36	1	35	[Edit] [Delete]
31-07-2023	pro	yellow	48	2	46	[Edit] [Delete]
31-07-2023	pin	green	34	2	32	[Edit] [Delete]

Showing 1 to 3 of 3 entries

Gambar 7. Tampilan halaman *input* penjualan

4.5. Tampilan Halaman Laporan Pengiriman

Halaman ini menampilkan data pengiriman pada hari ini atau sesuai dengan cabang dan tanggal yang dipilih. Data yang di tampilkan dapat di *export* sesuai dengan *format file* yang di butuhkan. Selain itu jika ingin menambahkan laporan pengiriman bisa mengklik tombol tambah pengiriman setelah mengisi cabang dan tanggal.



Data Pengiriman

Cabang: idari

Tanggal: 07/07/2023

Tabel Data Pengiriman

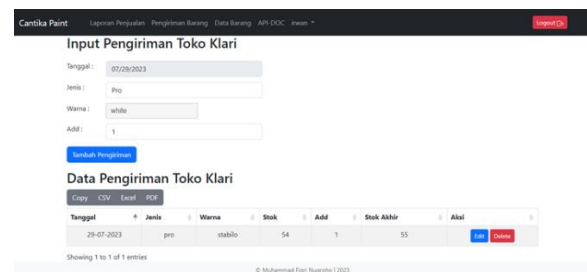
Tanggal	Jenis	Warna	Stok	Out	Stok Akhir
29-07-2023	pro	stabilo	54	1	53

Showing 1 to 1 of 1 entries

Gambar 8. Tampilan halaman laporan pengiriman

4.6. Tampilan Halaman Input Pengiriman

Setelah mengklik tombol tambah pengiriman pada halaman laporan pengiriman, maka sistem akan menampilkan halaman *input* pengiriman, halaman ini berfungsi untuk mengelola pengiriman sesuai dengan cabang dan tanggal yang dipilih. Pada halaman ini pengguna dapat menambahkan laporan pengiriman dengan mengisi jenis, warna, dan jumlah barang masuk. Selain itu pengguna dapat menghapus atau mengedit data yang ada, dengan cara memilih data yang diinginkan pada tabel data pengiriman.



Input Pengiriman Toko Klari

Tanggal: 07/07/2023

Jenis: pro

Warna: white

Addr: 1

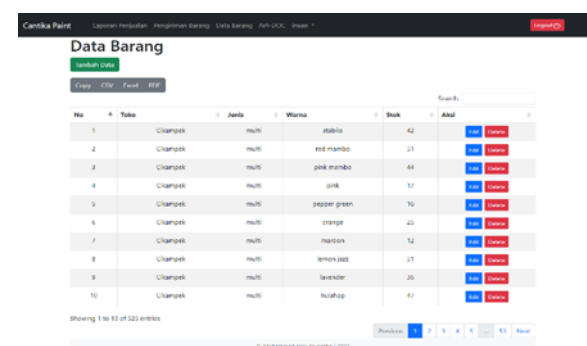
Data Pengiriman Toko Klari

Tanggal	Jenis	Warna	Stok	Add	Stok Akhir	Aksi
29-07-2023	pro	stabilo	54	1	55	[Edit] [Delete]

Showing 1 to 1 of 1 entries

Gambar 9. Tampilan halaman *input* pengiriman

4.7. Tampilan Halaman Data Barang



Data Barang

Tabel Data Barang

No	Toko	Jenis	Warna	Stok	Aksi
1	Clampok	multi	stabilo	42	[Edit] [Delete]
2	Clampok	multi	red mamba	11	[Edit] [Delete]
3	Clampok	multi	pink mamba	44	[Edit] [Delete]
4	Clampok	multi	pink	17	[Edit] [Delete]
5	Clampok	multi	pepper green	16	[Edit] [Delete]
6	Clampok	multi	orange	25	[Edit] [Delete]
7	Clampok	multi	maroon	12	[Edit] [Delete]
8	Clampok	multi	lemon jazz	11	[Edit] [Delete]
9	Clampok	multi	lavender	16	[Edit] [Delete]
10	Clampok	multi	hulalop	47	[Edit] [Delete]

Showing 1 to 10 of 102 entries

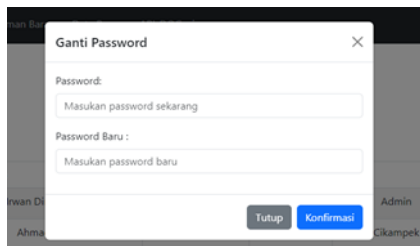
Gambar 10. Tampilan halaman data barang

Tampilan halaman data barang ini menampilkan data barang sesuai dengan cabang pengguna, namun jika pengguna merupakan admin data yang

ditampilkan adalah data barang seluruh cabang selain itu admin dapat menambah data barang dengan cara mengklik tombol tambah data dan mengisi *form* tambah data barang, selain itu admin juga dapat mengedit atau menghapus data barang.

4.8. Tampilan Form Ganti Password

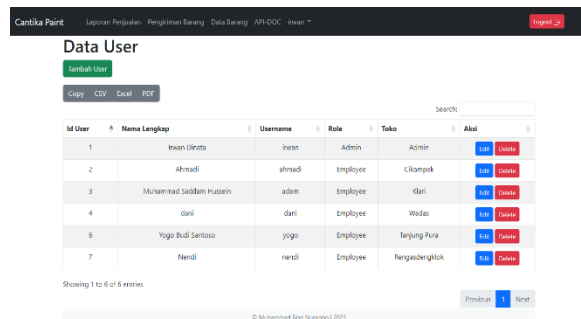
Formulir untuk mengganti *password* dibuat pada sebuah modal, pengguna dapat mengganti *password* dengan mengisi *password* baru dan *password* lama, jika sesuai maka *password* baru akan disimpan kedalam *database*, dan pengguna akan otomatis *logout* dan di arahkan ke halaman *login*.



Gambar 11. Tampilan form ganti password

4.9. Tampilan Halaman Edit User

Halaman edit *user* yang hanya dapat di akses dan dikelola oleh pengguna dengan hak akses admin, halaman ini memungkinkan admin untuk menambahkan *user* dengan cara mengklik tombol “tambah *user*”, selain itu admin juga dapat mengedit dan menghapus data *user* yang sudah ada.

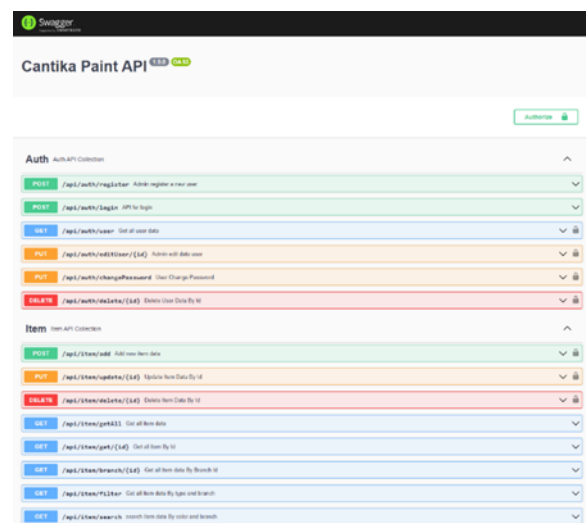


Gambar 12. Tampilan halaman edit user

4.10. Tampilan Halaman Dokumentasi API

Halaman dokumentasi API ini berfungsi untuk *developer* yang akan mengembangkan aplikasi ini

supaya dapat memahami API yang terdapat pada aplikasi manajemen toko cantika paint. *Framework* dokumentasi API yang digunakan adalah swagger, fungsi swagger disini tidak hanya untuk melihat seluruh *endpoint* dari aplikasi, tetapi juga dapat mengujinya langsung dengan mengirim *request* dan menerima *response*. *Endpoint* yang memiliki ikon gembok artinya untuk mengakses *endpoint* tersebut di perlukan *token* yang bisa didapatkan setelah melakukan *login*. *Token* ini didapatkan dengan mengirimkan *request* berisi *username* dan *password* pada *endpoint* *login*. Setelah itu *token* dimasukan kedalam menu *Authorize*, kemudian ikon gembok akan terbuka yang artinya seluruh *endpoint* dapat di akses. Selain itu di dalam dokumentasi ini juga terdapat kumpulan *response* berdasarkan kode status HTTP untuk setiap *endpoint*.



Gambar 13. Halaman dokumentasi API

4.11. Hasil Pengujian Sistem

Pengujian dilakukan dengan menggunakan metode *black box testing* untuk mengevaluasi aspek fungsional dari aplikasi yang telah dikembangkan.

1. Pengujian Terhadap Halaman Login

Pengujian dilakukan dengan mengisi *username* dan *password* dengan data yang valid dan tidak valid. Jika tidak valid, maka sistem akan menampilkan pesan kesalahan dan pengguna akan tetap berada di halaman *login*.

Tabel 1. Hasil pengujian halaman *login*

Skenario Pengujian	Kasus Pengujian	Hasil yang Diharapkan	Ket
Masuk kehalaman <i>login</i>	Membuka aplikasi	Sistem menampilkan halaman <i>login</i>	Sesuai
Mengisi <i>username</i> serta <i>password</i> yang valid	Mengisikan kombinasi <i>username</i> dan <i>password</i> yang valid, lalu mengklik tombol <i>login</i>	Sistem menampilkan halaman utama	Sesuai
Mengisi <i>username</i> yang valid dan <i>password</i> yang tidak valid	Memasukan <i>username</i> yang valid dan <i>password</i> yang tidak valid lalu mengklik tombol <i>login</i>	Sistem menampilkan pesan gagal “ <i>password</i> yang anda masukan salah” dan tidak masuk ke halaman utama.	Sesuai
Mengisi <i>username</i> dan <i>password</i> yang tidak valid	Memasukkan kombinasi <i>username</i> dan <i>password</i> yang tidak valid lalu mengklik tombol <i>login</i>	Sistem menampilkan pesan gagal “ <i>user</i> tidak ditemukan” dan tidak masuk ke halaman utama	Sesuai

2. Pengujian halaman *input* penjualan
 Pengujian pada halaman input penjualan berupa pengujian untuk menambah, mengedit, dan menghapus data penjualan.

Halaman ini dapat diakses setelah memilih cabang dan tanggal di halaman laporan penjualan dan menekan tombol tambah laporan.

Tabel 2. Hasil pengujian halaman *input* penjualan

Skenario Pengujian	Kasus Pengujian	Hasil yang Diharapkan	Ket
Masuk kehalaman <i>input</i> penjualan	Membuka halaman laporan penjualan lalu memilih cabang dan tanggal dan menekan tombol tambah laporan	Sistem menampilkan halaman <i>input</i> penjualan untuk cabang dan tanggal yang dipilih.	Sesuai
Menambahkan data penjualan	Memasukan jenis, warna, dan barang keluar/ <i>out</i> lalu menekan tombol tambah laporan	Berhasil menyimpan data ke <i>database</i> kemudian menampilkan pesan berhasil dan menampilkan data penjualan yang sudah ditambahkan	Sesuai
Mengosongkan jumlah barang keluar/ <i>out</i> pada <i>form input</i> penjualan	Memasukan jenis dan warna lalu menekan tombol tambah laporan	Data tidak disimpan ke <i>database</i> lalu menampilkan pesan gagal "Data Tidak Boleh Kosong"	Sesuai
Memasukan data penjualan yang sudah ada	Memasukan jenis, warna, dan barang keluar/ <i>out</i> yang telah ditambahkan lalu menekan tombol tambah laporan	Data tidak disimpan ke <i>database</i> lalu menampilkan pesan gagal "Data penjualan sudah ada"	Sesuai
Mengedit data penjualan	Memasukan data barang keluar/ <i>out</i> lalu mengklik tombol <i>update</i>	Menyimpan data yang telah di edit ke dalam <i>database</i>	Sesuai
Menhapus data penjualan	Menekan tombol <i>delete</i> pada data di tabel data penjualan	Data berhasil dihapus dan menampilkan pesan "Berhasil Hapus Data".	Sesuai

5. KESIMPULAN DAN SARAN

Aplikasi manajemen toko cantika paint berhasil dibuat dengan menggunakan bahasa pemrograman nodeJS dengan *framework* expressJS sebagai *backend* dan API *services*, *view engine* EJS sebagai *frontend*, dan *postgreSQL* sebagai *database* berhasil dikembangkan dengan menerapkan sistem REST API. Selanjutnya pengujian dilakukan menggunakan *blackbox testing* dengan hasil aplikasi manajemen toko cantika paint dapat berjalan sesuai dengan yang di harapkan. Aplikasi manajemen toko yang telah dibuat berhasil memberikan kontribusi yang signifikan dalam meningkatkan efisiensi dan produktivitas pemilik toko serta pegawai. Dengan sistem pelaporan yang efisien, pemilik toko dan pegawai dapat mengakses informasi hasil penjualan secara *real-time*, membantu mereka untuk mengambil keputusan yang lebih tepat waktu. Selain itu penggunaan aplikasi ini telah membantu meningkatkan pemantauan yang lebih baik atas stok barang yang tersedia untuk menghindari kekurangan atau kelebihan persediaan, yang membantu mengoptimalkan pendapatan dan mengurangi pemborosan. Dengan aplikasi ini, pelaporan hasil penjualan dan pemantauan stok barang menjadi lebih mudah dan akurat. Hal ini akan membantu meningkatkan kualitas layanan toko dan dapat memberikan keuntungan yang lebih besar.

DAFTAR PUSTAKA

- [1] A. Parantoro, "Faktor Penggunaan Komputerisasi Akuntansi Pada UMKM," *INDIKATOR*, vol. 1, no. 2, 2020. doi:10.37753/indikator.v2i1.72
- [2] F. Doglio, "Developing your rest api," *REST API Development with Node.js*, pp. 191–259, 2018. doi:10.1007/978-1-4842-3715-1_7
- [3] "Node.js web application framework," Express, <http://www.expressjs.com/>. (accessed Jun. 26, 2023).
- [4] A. Muhammad, "APA ITU PostgreSQL? Mengenal database postgresql," Niagahoster Blog, <https://www.niagahoster.co.id/blog/postgresql-adalah/> (accessed Jun. 26, 2023).
- [5] M. Usnaini, V. Yasin, and A. Z. Sianipar, "Perancangan Sistem informasi inventarisasi Aset Berbasis web Menggunakan Metode Waterfall," *Jurnal Manajemen Informatika Jayakarta*, vol. 1, no. 1, p. 36, 2021. doi:10.52362/jmijayakarta.v1i1.415
- [6] R. Ratovsky, "Getting started with swagger [I] - what is Swagger?," *swagger.io*, <https://swagger.io/blog/api-development/getting-started-with-swagger-i-what-is-swagger/> (accessed Jun. 25, 2023).
- [7] S. H. Bariyah and K. A. Imania, "Pengembangan Virtual assistant Chatbot Berbasis Whatsapp Pada Pusat Layanan informasi mahasiswa institut pendidikan Indonesia - garut," *JURNAL PETIK*, vol. 8, no. 1, pp. 66–79, 2022. doi:10.31980/jpetik.v8i1.1575
- [8] R. A. Firdaus and A. Ashari, "Basis Data Paralel pada Sistem Multikomputer Menggunakan PostgreSQL-MPI," *SEMNAS TEKNO MEDIA ONLINE*, vol. 2, no. 1, pp. 1–16, Feb. 2014.
- [9] F. Hanafi, "Mengenal Lebih Dekat *view engine* Express.js," *Codepolitan*, <https://www.codepolitan.com/blog/mengenal->

- lebih-dekat-view-engine-expressjs/ (accessed Sep. 20, 2023).
- [10] Y. Firmansyah and J. Jamilah, "Implementasi SDLC waterfall dalam pembuatan game Edukasi Perjuangan indonesia"hisotira" menggunakan RPG maker MV Berbasis Android," *Jurnal Khatulistiwa Informatika*, vol. 6, no. 2, pp. 178–185, 2018. doi:10.31294/khatulistiwa.v6i2.162