PENERAPAN NODEJS DAN POSTGRESQL SEBAGAI BACKEND PADA APLIKASI ECOMMERCE LOCALLA

Siti Sauda[1], M Barokah[2]
[1,2]Program Studi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Bina Darma
Email: [1]siti_sauda@binadarma.ac.id, [2]181420161@student.binadarma.ac.id

ABSTRAK
Perkembangan teknologi di era serba digital ini sangat berpengaruh untuk kegiatan jual beli. Salah satunya ialah ecommerce, kemudahan dan kenyamanan berbelanja dari ecommerce dimanfaatkan hampir semua kalangan terutama pelaku usaha. Dari sisi pelaku usaha, ecommerce sangat efisien dari segi waktu, pencarian informasi produk dan transaksi dapat lebih cepat dan akurat, tidak perlu membuka toko fisik serta bisa memasarkan produknya dari mana saja. Localla merupakan sebuah nama aplikasi yang akan dikembangkan dalam penelitian ini. Localla berfokus pada penjualan produk lokal atau produk buatan dalam negeri. Tujuan dari penelitian adalah mengembangkan aplikasi ecommerce pada sisi backend menggunakan Nodejs dan PostgreSQL. Hasil dan kesimpulan dari penelitian ini adalah backend berhasil dibangun dengan baik dan REST API dapat dikonsumsi oleh frontend.
Kata Kunci: Backend, Ecommerce, Nodejs, PostgreSQL, REST API

1. PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi di era serba digital ini, dimanfaatkan oleh banyak orang sebagai peluang bisnis. Baik menggunakan media sosial maupun menggunakan *website. E-commerce* merupakan salah satu bisnis yang menggunakan *internet. E-commerce* adalah penggunaan *World Wide Web* (WWW), *internet*, dan *mobiles apps* agar bisa bertransaksi secara digital antara organisasi dan individu (Handayani, 2018). Melalui *e-commerce*, pembeli bisa melihat langsung barang apa saja yang sedang ditawarkan. Kemudian pembeli bisa membeli barang tanpa harus mendatangi langsung tempat pusat pembelanjaan. Oleh karena itu, diperlukannya suatu aplikasi *e-commerce* untuk memanfaatkan peluang tersebut.

Dalam membangun sebuah *backend* pada aplikasi, pemilihan bahasa pemrograman dan *database* meupakan bagian yang penting. Nodejs merupakan turunan dari bahasa pemrograman javascript yang bersifat *non-blocking*, sehingga nodejs dapat menyelesaikan banyak *request* secara bersamaan (Mubariz et al., 2020). Untuk pembuatan *database* skala besar, postgresql menjadi solusi yang lebih baik karena beberapa *query* lebih unggul dibandingkan MySQL (Praba & Safitri, 2020) .

Localla merupakan aplikasi yang akan dikembangkan oleh penulis dan teman-teman. Localla terdiri dari *frontend* dan *backend.* Localla bertujuan sebagai tempat untuk penjualan produk lokal, sehingga membantu penjual dan konsumen mencari produk-produk lokal. Dengan adanya aplikasi tersebut, diharapkan dapat meningkatkan daya dorong masyarakat agar lebih menggunakan produk dalam negeri.

1.2. Tinjuan Pustaka

a. *Backend*

Backend merupakan sisi *server* dan *database* yang bekerja di balik layar dalam suatu aplikasi. Hal yang akan dikembangkan pada bagian *backend* antara lain metode HTTP (*post, get, put,* dan *delete*), dan API (Syaftiaan et al., 2021).

b. *Application Programming Interface* (API)

Application Programming Interface merupakan perangkat yang mengintegrasikan berbagai aplikasi secara bersamaan untuk bertukar data. Sehingga fitur yang serupa tidak akan dibuat ulang karena telah disediakan oleh API yang diakses. Misalnya: integrasi *login* menggunakan google, mengirim pesan dari aplikasi melalui gmail, dan lain-lain. Salah satu dari desain arsitektru di dalam API disebut REST API. REST API berfungsi antara *client* dengan *server* bisa saling bertukar informasi atau data (Zein, 2018).

c. *REpresentional State Transfer* (REST)

REST ialah gaya arsitektur untuk melakukan pengiriman informasi melalui HTTP. Umumnya cara kerjanya seperti *website*. Klien mengirim *request* ke *server* kemudian *server* memberikan *response* kepada klien (Kurniawan et al., 2020).

Dalam arsitekturnya, REST *server* menyediakan *resource* sedangkan REST *client* menerima *resource* tersebut. *Resource* tersebut dapat berbentuk JSON, XML, atau teks.

d. Nodejs

Nodejs merupakan *runtime environment* platform *server-side* yang bersifat *open source* dan *cross* platform yang menggunakan *chrome* V8 *javascript engine*, sehingga nodejs dapat menjalankan javascript diluar *browser* (Subramanian, 2019).

e. *Javascript Object Notation* (JSON)

JSON merupakan turunan javascript, namun bisa digunakan pada bahasa pemrograman lainnya. Saat

standalone ekstensi yang digunakan dalam JSON yaitu .json. Sedangkan saat didefinisikan dalam format lain, JSON dimasukkan ke dalam sebuah variabel (Hasanuddin et al., 2022).

f. Sequelize

Sequelize merupakan *Object Relational Mapping* (ORM) nodejs untuk MySQL, PostgreSQL, dan SQL lainnya.

g. JSON Web Token (JWT)

JSON Web Token adalah sebuah token yang berbentuk string panjang untuk digunakan dalam melakukan sistem autentikasi dan pertukaran informasi (Rahmatulloh et al., 2018). Pada aplikasi *website* umumnya menggunakan *session* untuk melakukan *login*, tapi didalam API hanya menggunakan JWT. JWT terbagi menjadi tiga bagian yang dipisah oleh tanda titik, yaitu: *Header* untuk algoritma *encoding* yang digunakan, *Payload* untuk data-data informasi, dan *Signature* untuk nilai *hash* dari komponen-komponen *header*, *payload*, dan *secret key*.

h. Node Package Manager (NPM)

NPM merupakan pengelola paket yang dipakai untuk platform nodejs. Dengan menggunakan *package manager* akan mempersingkat pembuatan fitur, semakin komplek suatu aplikasi semakin banyak NPM yang digunakan (Nasution & Iswari, 2021).

i. Heroku

Heroku adalah sebuah *cloud platform as a service* yang berfungsi sebagai *web service* untuk mengelola informasi dan *knowledge base* (Firdausillah & Arieansyah, 2019).

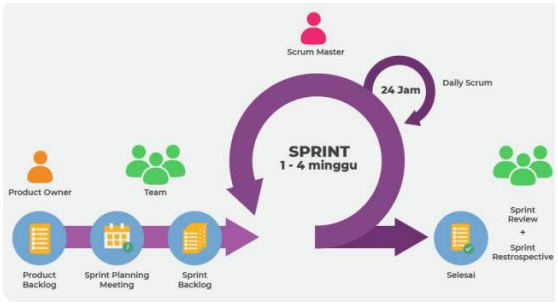
1.3. Metodologi Penelitian

a. Pengumpulan Data

Pengumpulan data pada penelitian ini yaitu observasi, mengumpulkan data dengan cara pengamatan terhadap aplikasi terdahulu yang serupa. Kemudian studi pustaka, mengumpulkan data-data dengan cara membaca dan mempelajari buku-buku, jurnal, ataupun referensi lain yang berkaitan dengan penelitian yang dibahas.

b. Metode Pengembangan

Metode pengembangan yang digunakan adalah Scrum. Metode scrum merupakan salah satu metode pengembangan perangkat lunak yang termasuk dalam *agile software development* (Andipradana & Dwi Hartomo, 2021).



Gambar 1. Metode scrum

Berikut merupakan tahapan-tahapan dalam metode scrum:

a) *Product Backlog* adalah daftar fitur yang diprioritaskan, berisi dekripsi singkat dari semua fungsionalitas yang diinginkan dalam produk. *Product backlog* dibiarkan tumbuh dan berubah seiring semakin banyak yang dipelajari tentang produk dan pelanggan.
b) *Sprint Backlog* adalah daftar tugas yang diidentifikasi oleh *scrum team* untuk diselesaikan selama *sprint*. Selama *sprint meeting planning*, *developer* memilih sejumlah item *product backlog*, dan mengidentifikasi tugas yang diperlukan untuk menyelesaikan setiap *user story*.
c) *Sprint* adalah batasan waktu di mana pekerjaan harus diselesaikan, biasanya berdurasi selama satu bulan atau kurang.
d) *Sprint Review* adalah pertemuan yang diadakan di akhir *sprint*, di mana *developer* menunjukkan apa yang telah dicapai.

2. PEMBAHASAN

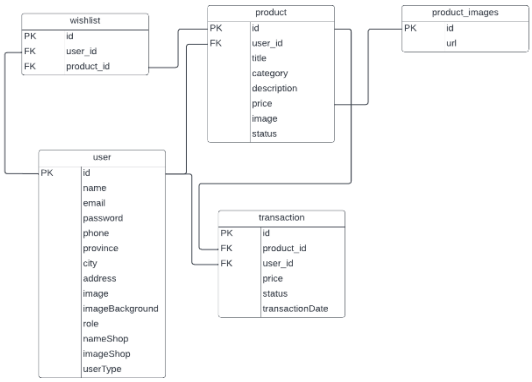
2.1. *Product Backlog*

Pada tahap ini menentukan fitur-fitur berdasarkan kebutuhan pengguna oleh *product owner*. Daftar fitur-fitur dapat dilihat pada tabel 1 sebagai berikut.

Tabel 1. *Product backlog*

No.	Nama *backlog*	Prioritas
1	Perancangan basis data	Tinggi
2	*Login, register, auth user controller*	Tinggi
3	*Seller controller*	Tinggi
4	*Search feature*	Sedang
5	*Image upload helper*	Rendah
6	Google oauth	Rendah
7	JWT *security*	Tinggi
8	Heroku *deployment*	Tinggi
9	Swagger *Documentation*	Tinggi

Pada *product backlog* yang terdapat di tabel 1, berfokus pada pengembangan *backend* pada aplikasi menggunakan nodejs dan postgresql.



Gambar 2. Perancangan basis data

2.2. *Sprint*

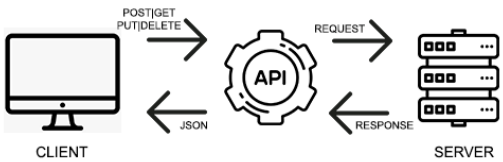
Tahapan ini mempertimbangkan estimasi waktu pekerjaan *product backlog* yang akan dikerjakan.

Tabel 2. *Sprint*

Tugas	Estimasi (Minggu ke-)			
	1	2	3	4
Perancangan basis data				
Coding				
Testing				
Heroku deployment & swagger documentation				

2.3. *Sprint Review*

Tahapan terakhir yaitu meninjau apa yang dikerjakan dari semua *sprint* yang selanjutnya mengevaluasi apakah sudah sesuai kebutuhan.



Gambar 3. Alur REST API

Pada gambar 3, *client* memanggil URL API atau biasa disebut *endpoint* menggunakan HTTP *method* (*GET, POST, PUT, DELETE*). Setelah itu, *server* menerima request berdasarkan HTTP *method* dan *endpoint*. Selanjutnya, *server* mengembalikan hasil *request* kepada *client* dalam format *response* JSON dan HTTP *response code*. HTTP *response code* berfungsi untuk mengidentifikasi *request* yang diterima. Seperti *code* 200-208 yang menunjukkan *request* berhasil, *code* 400-499 ketika ada kesalahan pada *client*, dan 500-511 menunjukkan ada kesalahan pada *server*.

Tabel 3. Proses Auntetikasi

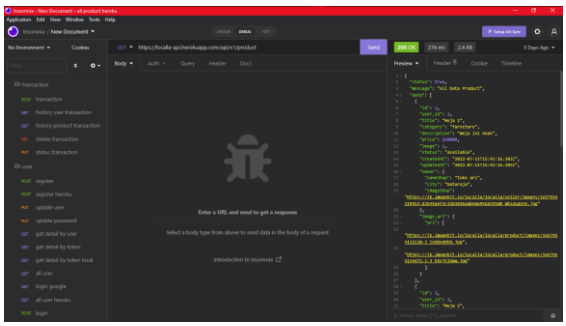
Komponen	Data
Method	Create: POST Read: GET Update: PUT Delete: DELETE
URL	URL API

Data	{
	"email": "oka@gmail.com",
	"password": "12345678"
	}

Success Response	Code: 200 Status: True Message: “Login Success” Data: {data}, tokenAccessJWT
Error Response	Code: 400 Status: False Message: “Email is not found” Data: null
	Code: 500 Status: False Message: “"function is not defined Data: null

Pada tabel 3, *create*, *update*, dan *delete* menggunakan HTTP *method POST, PUT, DELETE*, sedangkan *read* menggunakan *GET*.

GET tidak perlu mengirimkan data, cukup memanggil URL API. Sedangkan *POST, PUT, DELETE* memperlukan data, karena *method* tersebut melakukan perubahan dalam *database*. Jika valid, maka akan menampilkan *success response*. Jika tidak valid atau terjadi kesalahan dalam *backend*, akan menampilkan *error response* berdasarkan *error* tersebut.



Gambar 4. *Testing* API menggunakan insomnia

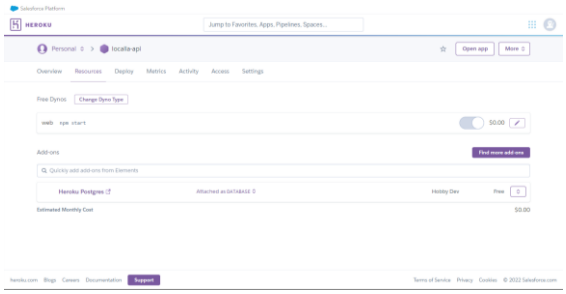
Pada gambar 4, API yang telah dibuat selanjutnya dilakukan pengujian untuk membuktikan apakah berfungsi sesuai dengan hasil yang diinginkan. Hasil dari pengujian pada dilihat pada tabel berikut.

Tabel 4. Pengujian dengan Insomnia

Method	URI	Hasil Uji
POST	/auth/login	Berhasil
GET	/auth/google	Berhasil
POST	/auth/forgot-password	Berhasil
POST	/auth/reset-password/{user_id}	Berhasil
PUT	/user	Berhasil
GET	/user	Berhasil
GET	/user/profile	Berhasil
GET	/user/{user_id}	Berhasil
POST	/user/register	Berhasil
PUT	/user/image	Berhasil
PUT	/user/updatepassword	Berhasil
PUT	/seller	Berhasil
POST	/product	Berhasil
GET	/product	Berhasil
GET	/product/{product_id}	Berhasil
DELETE	/product/{product_id}	Berhasil
PUT	/product/{product_id}	Berhasil
GET	/product/seller/{seller_id}	Berhasil
POST	/product/wishlist/{product_id}	Berhasil
GET	/transaction	Berhasil
GET	/transaction/user	Berhasil
GET	/transaction/product/{product_id}	Berhasil
POST	/transaction/{product_id}	Berhasil
DELETE	/transaction/delete/{transaction_id}	Berhasil
PUT	/transaction/update/{transaction_id}	Berhasil

Setelah seluruh API telah diuji dan sesuai harapan. Langkah selanjutnya menentukan *cloud server*, sehingga API yang dibuat dapat dikonsumsi oleh *frontend.*

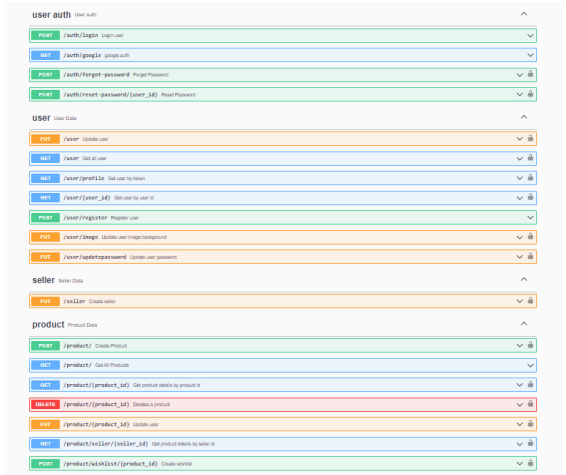
Heroku digunakan sebagai *cloud server*, karena Heroku mendukung nodejs, konfigurasi yang mudah, mempunyai Heroku Postgres, dan gratis.



Gambar 5. Heroku deployment

Langkah terakhir yaitu membuat dokumentasi API. Dengan adanya dokumentasi API, *developer* tidak mengalami kesusahan dalam mengakses API tersebut.

Swagger merupakan salah satu *tools* yang digunakan untuk membuat dokumentasi API. Selain membuat dokumentasi, swagger bisa melakukan pengujian API.



Gambar 6. *Swagger documentation*

3. KESIMPULAN

Nodejs dan postgresql dari sisi *backend development* dapat diterapkan untuk dikonsumsi pada *client website.* Kemudahan dokumentasi API menggunakan swagger sangat membantu *frontend* dalam mengakses API serta format yang dipakai didalamnya.

Saran terkait hasil penelitian ini adalah API yang telah ada bisa dikonsumsi pada *client mobile*, sehingga sistem akan terintegrasi dengan *client mobile.*

PUSTAKA

Andipradana, A., & Dwi Hartomo, K. (2021). Rancang Bangun Aplikasi Penjualan Online Berbasis Web Menggunakan Metode Scrum. *Jurnal Algoritma*, *18*(1), 161–172. https://doi.org/10.33364/algoritma/v.18-1.869

Firdausillah, F., & Arieansyah. (2019). Implementasi Algoritma Levenshtein Distance Sebagai Chatbot Agen Pariwisata Berbasis Aplikasi LINE. *Seminar Nasional APTIKOM (SEMNASTIK)*, *1*, 377–385.

Handayani, S. (2018). Perancangan Sistem Informasi Penjualan Berbasis E-Commerce Studi KaHandayani, S. (2018). Perancangan Sistem Informasi Penjualan Berbasis E-Commerce Studi Kasus Toko Kun Jakarta. ILKOM Jurnal Ilmiah, 10(2), 182–189. https://doi.org/10.33096/ilkom.v10i2.310. *ILKOM Jurnal Ilmiah*, *10*(2), 182–189.

Hasanuddin, Asgar, H., & Hartono, B. (2022). Rancang Bangun Rest Api Aplikasi Weshare Sebagai Upaya Mempermudah Pelayanan Donasi Kemanusiaan. *Jurnal Informatika Teknologi dan Sains*, *4*(1), 8–14. https://doi.org/10.51401/jinteks.v4i1.1474

Kurniawan, I., Humaira, & Rozi, F. (2020). REST API Menggunakan NodeJS pada Aplikasi Transaksi Jasa Elektronik Berbasis Android. *JITSI : Jurnal Ilmiah Teknologi Sistem Informasi*, *1*(4), 127–132. https://doi.org/10.30630/jitsi.1.4.18

Mubariz, A., Nur, D., Tungadi, E., & Utomo, M. N. Y. (2020). Perancangan Back-End Server Menggunakan Arsitektur Rest dan Platform Node . JS (Studi Kasus : Sistem Pendaftaran Ujian Masuk Politeknik Negeri Ujung Pandang). *Seminar Nasional Teknik Elektro dan Informatika (SNTEI)*, 72–77.

Nasution, & Iswari, L. (2021). Penerapan React JS Pada Pengembangan FrontEnd Aplikasi Startup Ubaform. *Automata*, *2*(2).

Praba, A. D., & Safitri, M. (2020). Studi Perbandingan Performansi Antara Mysql Dan Postgresql. *Jurnal Khatulistiwa Informatika*, *8*(2), 88–93. https://doi.org/10.31294/jki.v8i2.8851

Rahmatulloh, A., Sulastri, H., & Nugroho, R. (2018). Keamanan RESTful Web Service Menggunakan JSON Web Token (JWT) HMAC SHA-512. *Jurnal Nasional Teknik Elektro dan Teknologi Informasi (JNTETI)*, *7*(2). https://doi.org/10.22146/jnteti.v7i2.417

Subramanian, V. (2019). Pro MERN Stack. In *Pro MERN Stack*. https://doi.org/10.1007/978-1-4842-4391-6

Syaftiaan, B., Safira, N. F., & Rizky, F. (2021). *Rancang Bangun Backend Aplikasi Jobbie: Pencarian Dan Penyedia Jasa Lapangan Kerja*. *8*(6), 12441.

Zein, A. (2018). PERAN TEXT PROCESSING DALAM APLIKASI PENERJEMAH MULTI BAHASA MENGGUNAKAN AJAX API GOOGLE. *World Development*, *1*(1), 1–15.