

**DEPARTEMEN ILMU KOMPUTER**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**INSTITUT PERTANIAN BOGOR**

**BOGOR**

**2017**

**NUGROHO AGUNG CAHYONO**

**PENGEMBANGAN MODUL PELAPORAN HARGA**

**PADA APLIKASI DIGITAL TANI**

**PERNYATAAN MENGENAI SKRIPSI DAN  
SUMBER INFORMASI SERTA PELIMPAHAN HAK CIPTA**

Dengan ini saya menyatakan bahwa skripsi berjudul Pengembangan Modul Pelaporan Harga pada Aplikasi Portal Harga adalah benar karya saya dengan arahan dari komisi pembimbing dan belum diajukan dalam bentuk apa pun kepada perguruan tinggi mana pun. Sumber informasi yang berasal atau dikutip dari karya yang diterbitkan maupun tidak diterbitkan dari penulis lain telah disebutkan dalam teks dan dicantumkan dalam Daftar Pustaka di bagian akhir skripsi ini.

Dengan ini saya melimpahkan hak cipta dari karya tulis saya kepada Institut Pertanian Bogor.

Bogor, Agustus 2017

*Nugroho Agung Cahyono*

NIM G64130039

**ABSTRAK**

NUGROHO AGUNG CAHYONO. Pengembangan Modul Pelaporan Harga pada Aplikasi Digital Tani. Dibimbing oleh DEAN APRIANA RAMADHAN.

Informasi yang sulit tersampaikan dari petani kepada pemerintah merupakan permasalahan yang terjadi dalam mengelola dan mengembangkan sektor pertanian. Informasi yang sulit tersampaikan tersebut menyebabkan harga jual hasil panen petani di bawah harga yang ditetapkan oleh pemerintah. Perkembangan Internet dapat dijadikan sebagai peluang untuk mempermudah penyampaian informasi dengan menyediakan media yang mampu menghubungkan penyuluh, petani, masyarakat, pedagang, dan pemerintah. Penelitian ini bertujuan untuk memberikan informasi dan media komunikasi antara pemerintah, petani, masyarakat, dan pedagang. Penelitian inimengembangkan API menggunakan *Framework* Express.jsdan basis data non relasional MongoDB. Pengembangan API menggunakan metode *extreme programming* yang merupakan bagian dari *agile software engineering* yang terbagi menjadi tiga iterasi*.* API yang dikembangkan adalah modul pelaporan harga pada sisi *back end* dengan keluaran dalam bentuk JSON*.* JSON memiliki format yang terdiri atas status, data, *message*, dan token. API yang dikembangkan dapat diakses oleh *front end* berbasis *mobile* maupun web.

Kata kunci: *extreme programming*,modul pelaporan harga, REST API

**ABSTRACT**

NUGROHO AGUNG CAHYONO. Development of Price Reporting Module on Digital Tani Applications. Supervised by DEAN APRIANA RAMADHAN.

The difficulty for delivering an information from farmers to government is a problem that occurs in managing and developing the agricultural sector. It causes the selling price of farmers harvest becomes lower than the price that had been set by the government. The growth of the Internet can be used as an opportunity to facilitate the delivery of information by providing a media which is able to connect agricultural counselor, farmers, society, traders, and government. The purpose of this research is to provide information and communication media between government, farmers, communities, and traders. This API development used Express.js Framework and non relational database (MongoDB). Development of API used extreme programming method that was part of agile software engineering with three iterations. The research developed API for price reporting module on back end side with output in JSON form. JSON had a format consisting of status, data, message, and token. APIs could be accessed by mobile or web application.

Keywords: extreme programming, price reporting module, REST API

Skripsi  
sebagai salah satu syarat untuk memperoleh gelar  
Sarjana Komputer   
pada  
Departemen Ilmu Komputer

**DEPARTEMEN ILMU KOMPUTER**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**INSTITUT PERTANIAN BOGOR**

**BOGOR**

**2017**

**PENGEMBANGAN MODUL PELAPORAN HARGA**

**PADA APLIKASI DIGITAL TANI**

Penguji:

1. Rina Trisminingsih, SKomp MT
2. Husnul Khotimah, SKomp MKom

Judul Skripsi : Pengembangan Modul Pelaporan Harga pada Aplikasi Digital Tani

Nama : Nugroho Agung Cahyono

NIM : G64130039

Disetujui oleh

Dean Apriana Ramadhan, SKomp MKom

Pembimbing

Diketahui oleh

Dr Ir Agus Buono, MSi MKom

Ketua Departemen

Tanggal Lulus:

**PRAKATA**

Puji dan syukur penulis panjatkan kepada Allah *subhanahu wa ta’ala* atas segala karunia-Nya sehingga tugas akhir ini berhasil diselesaikan. Judul yang dipilih dalam penelitian yang dilakukan sejak Februari 2017 ialah Pengembangan Modul Pelaporan Harga Komoditas Pertanian untuk Masyarakat, Pedagang dan Pemerintah Menggunakan REST API. Penelitian ini dilaksanakan di Departemen Ilmu Komputer FMIPA IPB.

Penulis menyadari bahwa selama mengerjakan tugas akhir ini mengalami berbagai kendala. Namun, berkat bantuan, bimbingan, dan dukungan dari berbagai pihak kendala yang dihadapi berhasil diselesaikan. Oleh karena itu, penulis ingin mengucapkan terima kasih kepada:

1. Orang tua dan adik penulis yang memberi doa, semangat, dan dukungan moril maupun materiel.
2. Bapak Dean Apriana Ramadhan, SKomp MKom selaku dosen pembimbing yang telah sabar, ikhlas, meluangkan waktu, tenaga dan pikiran dalam memberi bimbingan, motivasi dan saran selama pengerjaan tugas akhir.
3. Teman-teman satu bimbingan, yaitu Fiqih Nur R, Ryan Baskara, dan Irfan Rafii yang memberikan saran, bantuan dan dukungan selama penelitian berlangsung.
4. Pegawai dan pengajar Departemen Ilmu Komputer IPB.
5. Teman-teman Sonic angkatan 50 Departemen Ilmu Komputer IPB.
6. Semua pihak yang terlibat dalam menyelesaikan tugas akhir ini baik secara langsung maupun tidak langsung yang tidak dapat penulis sebutkan satu per satu.

Bogor, Agustus 2017

*Nugroho Agung Cahyono*

**DAFTAR ISI**

DAFTAR TABEL vi

DAFTAR GAMBAR vi

DAFTAR LAMPIRAN vi

[PENDAHULUAN 1](#_Toc489962053)

[Latar Belakang 1](#_Toc489962054)

[Perumusan Masalah 2](#_Toc489962055)

[Tujuan Penelitian 2](#_Toc489962056)

[Manfaat Penelitian 2](#_Toc489962057)

[Ruang Lingkup Penelitian 2](#_Toc489962058)

[TINJAUAN PUSTAKA 3](#_Toc489962059)

[JSON 3](#_Toc489962060)

[REST API 3](#_Toc489962061)

[MongoDB 3](#_Toc489962062)

[*Roadmap* Penelitian 3](#_Toc489962063)

[METODE 4](#_Toc489962064)

[Data Penelitian 4](#_Toc489962065)

[Arsitektur Sistem 5](#_Toc489962066)

[Tahapan Penelitian 5](#_Toc489962067)

[Lingkungan Pengembangan 7](#_Toc489962068)

[HASIL DAN PEMBAHASAN 7](#_Toc489962069)

[Tahapan Penelitian 7](#_Toc489962070)

[SIMPULAN DAN SARAN 22](#_Toc489962071)

[Simpulan 22](#_Toc489962072)

[Saran 22](#_Toc489962073)

[DAFTAR PUSTAKA 22](#_Toc489962074)

LAMPIRAN 27

RIWAYAT HIDUP 55

**DAFTAR TABEL**

1. [Tahapan pengembangan setiap iterasi 7](#_Toc489962075)
2. [*User story* masyarakat 8](#_Toc489962076)
3. [Hasil pengujian iterasi pertama 12](#_Toc489962077)
4. [Hasil pengujian iterasi kedua 16](#_Toc489962078)
5. [Hasil pengujian iterasi ketiga 20](#_Toc489962079)

**DAFTAR GAMBAR**

1. [*Roadmap* penelitian *e*-*Government* bidang pertanian 4](#_Toc489962080)
2. [Arsitektur Penelitian 5](#_Toc489962081)
3. [Tahapan pada *Extreme Programming* (Pressman 2010) 6](#_Toc489962082)
4. [*Use case diagram* iterasi pertama 8](#_Toc489962083)
5. [CRC *cards* iterasi pertama 9](#_Toc489962084)
6. [*Class diagram* iterasi pertama 9](#_Toc489962085)
7. [*Collections* komoditas 10](#_Toc489962086)
8. [Daftar *collections* di basis dataiterasi pertama 10](#_Toc489962087)
9. [*Unit test* menambah komoditas 11](#_Toc489962088)
10. [Pengujian fungsi melihat laporan harga 12](#_Toc489962089)
11. [*Use case diagram* iterasi kedua 13](#_Toc489962090)
12. [CRC *cards* iterasi kedua 14](#_Toc489962091)
13. [*Class diagram* iterasi kedua 14](#_Toc489962092)
14. [Potongan *code* fungsi getDayLaporan 15](#_Toc489962093)
15. [Pengujian fungsi melihat laporan harga 60 hari sebelumnya 16](#_Toc489962094)
16. [*Use case diagram* iterasi ketiga 17](#_Toc489962095)
17. [CRC *cards* iterasi ketiga 18](#_Toc489962096)
18. [*Class* *diagram* iterasi ketiga 19](#_Toc489962097)
19. [*Code* memasukan data format JSON ke basis data mongoDB 20](#_Toc489962098)
20. [Integrasi fungsi getMailVerifydengan fungsi *register* 20](#_Toc489962099)
21. [Dokumentasi API di Swaggerhub 21](#_Toc489962100)

**DAFTAR LAMPIRAN**

1. [Deskripsi *user story* 24](#_Toc489962101)
2. [*Sequence diagram* iterasi pertama 25](#_Toc489962102)
3. [*Collaboration diagram* iterasi pertama 27](#_Toc489962103)
4. [Daftar *collections* iterasi pertama 28](#_Toc489962104)
5. [Kamus data 29](#_Toc489962105)
6. [*Acceptance test criteria* iterasi pertama 31](#_Toc489962106)
7. [*Sequence diagram* iterasi kedua 34](#_Toc489962107)
8. [*Collaboration diagram* iterasi kedua 38](#_Toc489962108)
9. [*Acceptance test criteria* iterasi kedua 41](#_Toc489962109)
10. [*Sequence diagram* iterasi ketiga 43](#_Toc489962110)
11. [*Collaboration diagram* iterasi ketiga 49](#_Toc489962111)
12. [*Acceptance test criteria* iterasi ketiga 53](#_Toc489962112)

# PENDAHULUAN

## Latar Belakang

Indonesia adalah negara agraris yang berarti bahwa sektor pertanian sebagai sumber mata pencaharian maupun sebagai penopang pembangunan. Sektor pertanian memiliki daya serap tenaga kerja yang tinggi. Sektor pertanian menyerap angkatan kerja sebanyak 35.76 juta jiwa atau 30.2% dari keseluruhan angkatan kerja nasional pada tahun 2014. Rata-rata kontribusi pertanian terhadap PDB mencapai 10.26% dengan pertumbuhan 3.90% selama periode 2010-2014 (Kementan 2014). Pertanian menjadi sektor yang penting di Indonesia karena menjadi dasar untuk penyediaan sandang, papan, dan pangan.

Informasi yang sulit tersampaikan dari petani kepada pemerintah merupakan permasalahan yang terjadi dalam mengelola dan mengembangkan sektor pertanian. Informasi yang sulit tersampaikan tersebut menyebabkan harga jual hasil panen petani di bawah harga yang ditetapkan oleh pemerintah. Tengkulak memanfaatkan kondisi tersebut dengan membeli hasil panen dari petani dengan harga rendah (Romadhan 2009).

Pesatnya perkembangan teknologi informasi dapat digunakan untuk mengatasi masalah yang terjadi di sektor pertanian. Teknologi informasi yang dapat dikembangkan salah satunya, yaitu media Internet atau komunikasi dunia maya (Elian *et al*. 2014). Berdasarkan survey APJII, pengguna Internet di Indonesia sebanyak 88.1 juta jiwa dari total 254.2 juta jiwa dengan tingkat penetrasi 34.9%. Berdasarkan perangkat yang digunakan untuk mengakses Internet, jumlah terbanyak yaitu pengguna Internet dengan menggunakan telepon seluler, laptop atau *notebook, personal computer,* dan tablet(APJII 2016)*.*

Penelitian tentang sistem informasi pelaporan harga sebelumnya telah dilakukan, di antaranya oleh Saputra (2015) dengan membangun aplikasi *Short Message Service* (SMS) *center* untuk informasi harga komoditi hasil pertanian Kabupaten Ogan Ilir. Admin pada sistem SMS *center* tersebut memberikan informasi yang dibutuhkan oleh pengguna. Pengguna mengirimkan SMS ke sistem untuk mendapatkan informasi harga komoditas pertanian, lalu sistem akan memberikan balasan SMS sesuai dengan permintaan yang telah dikirim oleh pengguna. SMS yang masuk maupun keluar dikelola oleh admin, kemudian diintegrasikan dengan dinas terkait sesuai dengan harga komoditas pertanian. Penelitian yang dilakukan oleh Ahmad (2015) tentang perancangan aplikasi komoditas pertanian berbasis Android. Pada sistem tersebut komoditas yang digunakan yaitu padi, jagung, kedelai. Pengguna dapat mengakses sistem untuk mendapatkan informasi komoditas, informasi harga, informasi cuaca, dan berita.

Penelitian ini melakukan *reengineering* penelitian yang dilakukan oleh Setyatama (2016), yaitu pengembangan aplikasi pelaporan harga komoditas pertanian berbasis *mobile* menggunakan REST API. Penelitian tersebut perlu pengembangan token yang dinamis untuk keamanan data (Setyatama 2016). Selain itu penelitian yang dilakukan oleh Mardika (2016) tentang pengembangan REST API modul produsen komoditas pertanian pada aplikasi portal harga perlu pengembangan *email* untuk mengatasi pengguna yang lupa *password*. *Reengineering* dilakukan pada bagian token dan basis data relasional MySQL. Token yang digunakan masih bersifat statis dan diubah menjadi token yang dinamis menggunakan *Json Web Token* (JWT). Basis data yang digunakan dalam penelitian ini adalah basis data non relasional MongoDB. Penelitian ini mengembangkan modul pelaporan harga dengan metode *extreme programming* yang merupakan bagian dari *agile software engineering*. Modul yang dikembangkan diharapkan menjadi media informasi antara *stakeholder* terkait dan mampu menyediakan informasi seputar harga komoditas pertanian kepada *stakeholder* terkait secara *real time.*

## Perumusan Masalah

Perumusan masalah pana penelitian ini yaitu, bagaimana cara mengembangkan modul pelaporan harga komoditas pertanian dengan arsitektur REST API, sehingga API dapat diakses oleh *platform mobile* atau web.

## Tujuan Penelitian

Penelitian ini bertujuan mengembangkan modul pelaporan harga komoditas pertanian menggunakan arsitektur REST API dan basis data non relasionalpada sisi *back end*.

## Manfaat Penelitian

Manfaat dari penelitian ini adalah:

1. Dari sisi pedagang

* 1. Informasi harga komoditas pertanian bisa diketahui secara *real* *time*.
  2. Memasarkan dagangan.

2. Dari sisi masyarakat

* 1. Masyarakat mengetahui harga yang beredar di pasar.
  2. Masyarakat ikut berperan dalam pengendalian harga komoditas pertanian dengan melaporkan harga di pasar secara langsung.

3. Dari sisi pemerintah

a Membuat kebijakan berdasarkan harga komoditas pertanian di pasar.

b Mengetahui daerah yang membutuhkan operasi pasar.

## Ruang Lingkup Penelitian

Ruang lingkup dari penelitian ini adalah:

1. Sistem tidak melakukan validasi untuk setiap data yang dimasukan oleh pengguna.
2. Komoditas pertanian yang digunakan sama dengan penelitian sebelumnya yaitu terdiri atas beras, daging sapi, cabai, kedelai, dan bawang merah.

# TINJAUAN PUSTAKA

## JSON

*Java Script Object Notation* (JSON) adalah format pertukaran data (*lightweight data-interchange format*), mudah diterjemahkan dan dibuat oleh komputer, serta mudah dibaca dan ditulis oleh manusia (JSON 2002). JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun. JSON bersifat interoperabilitas sehingga dapat mengatasi data yang berukuran besar dan untuk berinteraksi dengan berbagai sistem maupun *platform* tanpa batasan akses (JSON 2002). Format data JSON yang digunakan dalam penelitian ini memiliki komponen yang terdiri atas *array*, *boolean*, *number*, *object*, dan *string*.

## REST API

*Representational State Transfer* (REST) merupakan sebuah arsitek untuk aplikasi *web services*, dirancang untuk menggunakan protokol *Hypertext Transfer Protocol* (HTTP) sebagai penghubung komunikasi antara mesin dan mesin, tanpa menggunakan mekanisme yang kompleks. Perintah HTTP yang bisa digunakan adalah fungsi GET, POST, PUT, DELETE (Pautasso 2008). REST bersifat *stateless,* yang berarti setiap *request* yang diterima diproses secara *independent*. Format data yang digunakan oleh REST pada penelitian ini adalah *JavaScript Object Notation* (JSON).

## MongoDB

MongoDB merupakan salah satu jenis basis data non relasional dengan tipe *document store* basis data. MongoDB digunakan dalam penelitian ini untuk meyimpan data yang besar dalam jangka panjang dan *high performance* (MongoDB 2017). Metode *extreme programming* diintegrasikan dengan basis data MongoDB untuk mengoptimalkan data dengan perubahan cepat.

## *Roadmap* Penelitian

*Roadmap* merupakan perencanaan dari seluruh pekerjaan yang akan dikerjakan secara detil dan terperinci dalam jangka waktu tertentu. Pembuatan *roadmap* bertujuan untuk mencapai target dengan waktu yang telah ditentukan. Tiga tahapan perencanaan *roadmap* yang akan dilakukan oleh tim peneliti e-*Goverment* bidang pertanian pada Laboratorium *Software Engineering and Information Science* (SEIS) Departemen Ilmu Komputer FMIPA IPB dalam rentang waktu tahun 2016-2020, yaitu *big data analysis*, *product development*, dan ICT *literacy.* Tiga tahapan tersebut digambarkan secara detil pada Gambar 1.



Gambar 1 *Roadmap* penelitian *e*-*Government* bidang pertanian

Penelitian ini merupakan pelaksanaan *roadmap* pada tahap *product development* bagian *mobile* *and* *web back end development.* Fokus penelitian, yaitu pada bagian *back end* yang menghasilkan API. Fungsi API untuk mengatur dan menjadi sumber aliran data yang dibutuhkan oleh *platform* berbasis web maupun *mobile*. Pengembangan *back end* menggunakan Express.js yang merupakan *framework* Node.js. Basis data yang digunakan adalah basis datanon relasional.

# METODE

## Data Penelitian

Data yang digunakan antara lain, yaitu komoditas, laporan harga, operasi pasar, dan lokasi. Data tersebut merupakan data dari penelitian sebelumnya yang dilakukan oleh Setyatama (2016). Data lokasi diunduh dari https://github.com/cahyadsn/daerah. Data lokasi terdiri dari data provinsi, kabupaten, kecamatan, dan kelurahan, dan jenis.

## Arsitektur Sistem

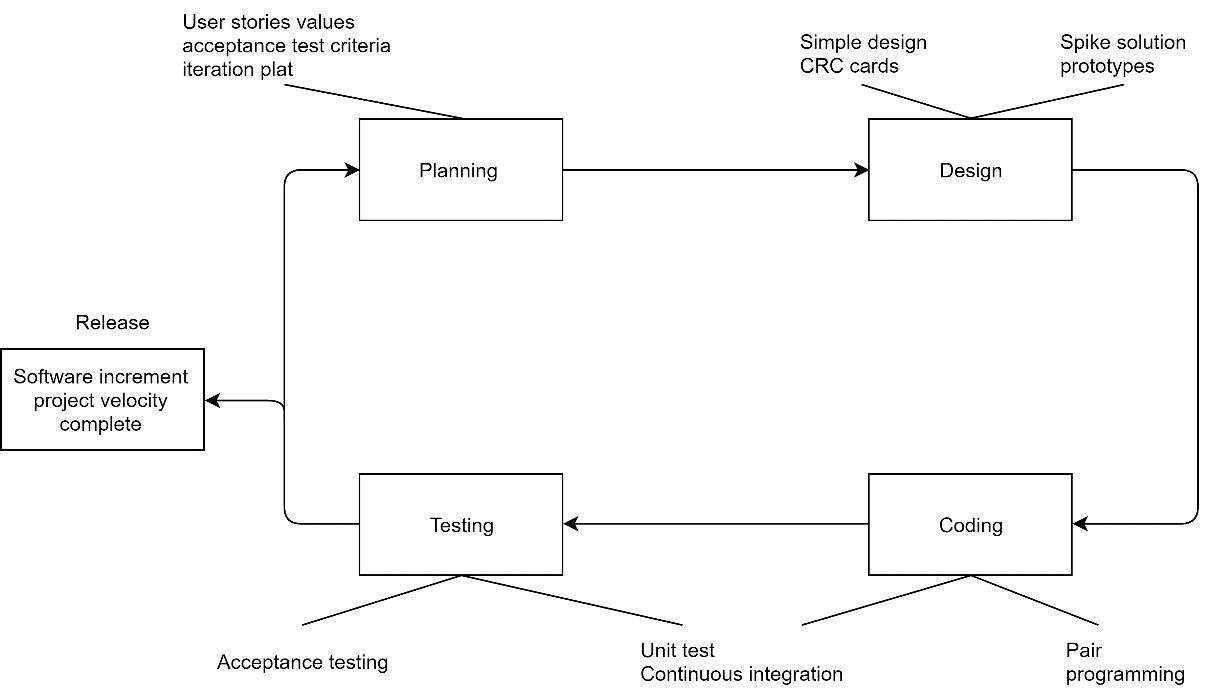
Perancangan arsitektur perangkat lunak dibagi menjadi tiga bagian, yaitu data, *client*, dan *server* (Setiana 2016). Arsitektur perancangan tersebut kemudian dikembangkan dan digunakan selama penelitian. Hasil pengembangan arsitektur penelitian dapat dilihat pada Gambar 2. Bagian data menggunakan basis data non relasional. Bagian *server* menggunakan Google Maps API untuk menandai suatu posisi pengguna berada berdasarkan *latitude* dan *longitude*. *Server* Express.js digunakan untuk pengolahan data dengan format keluaran JSON*.* Bagian *client* dibagi menjadi dua, yaitu *mobile* dan web. Bagian web menggunakan *platform* AngularJS 2 untuk mengakses data dari *server* dan menampilkan informasi daribasis data*,* sedangkan pada *mobile* menggunakan *framework* Ionic 2 dengan *platform* AngularJS 2.



Gambar 2 Arsitektur Penelitian

## Tahapan Penelitian

Metode yang digunakan adalah *extreme programming* yang merupakan bagian dari metode *agile software development* yang berbasis pada pengembangan iteratif dan kolaborasi antar anggota tim yang terorganisir. *Extreme programming* merupakan metodologi pengembangan perangkat lunak yang tanggap terhadap perubahan kebutuhan pengguna, sehingga meningkatkan kualitas perangkat lunak (Pressman 2010). Tahapan dalam *extreme programming*,yaitu *planning, design, coding,* dan *testing* dapat dilihat pada Gambar 3*.*

**

Gambar 3 Tahapan pada *Extreme Programming* (Pressman 2010)

**Perencanaan (*Planning*)**

Iterasi yang akan dilakukan selama penelitian sebanyak tiga kali. Tahap perencanaan dimulai dengan tim menentukan *requirements* yang diperlukan dalam pengembangan sistem. *Requirements* yang sudah ditentukan menjadi dasar untuk membuat *user story, use case diagram,* modul, dan fungsi. *Acceptance test criteria* dibuat untuk masing-masing modul dan fungsi yang direncanakan.

**Desain (*Design*)**

Tahap desain dimulai dengan membuat *class, responsibilities,* dan *collaboration* (CRC) *cards.* *Class* yang sudah dirancang lalu ditentukan atribut beserta dengan fungsi–fungsinya. *Sequence diagram* untuk menggambarkan interaksi dan komunikasi di antara objek dari *class* berdasarkan urutan waktu. Jika mengalami kesulitan *spike solution prototype* dilakukan untuk meminimalkan risiko selama proses pengembangan (Pressman 2010).

**Pengodean (*Coding*)**

Tahap pengodean melakukan *pair programming* dengan anggota tim *back end*, yaitu Fiqih Nur Ramadhan untuk mendukung *continuous integration.* GitHub sebagai media untuk mendukung *pair programming* dan penggabungan *code* pada tim *back end*. Repositori GitHub yang digunakan, yaitu https://github.com/ryanbaskara/backendPH.

**Pengujian (*Testing*)**

Pengujian menggunakan metode *black-box* dan dilakukan secara internal oleh tim. Hasil pengujian kemudian diintegrasikan dengan anggota tim *front end*,yaitu Ryan Baskara dan Irfan Rafii. Pengujian sesuai dengan *acceptance* *test criteria* untuk setiap modul dan fungsi yang diperoleh berdasarkan *user story* pada proses perencanaan (Pressman 2010). Pengujian dilakukan menggunakan aplikasi Postmandengan perintah HTTP (*Hypertext Transfer Protocol*) *method* POST dan GET*.*

## Lingkungan Pengembangan

Spesifikasi perangkat keras yang digunakan pada penelitian ini adalah laptop dengan spesifikasi *Processor* Intel® Core™ i3-3217u, RAM 4 GB, 500 GB HDD, VGA Ge Force 740M 2 Gb. Perangkat lunak yang digunakan adalah sistem operasi Linux Ubuntu 16.04 LTS, *text editor* Visual Studio Code versi 1.13.1, DBMS MongoDB versi 3.2.10, bahasa pemrograman Javascript versi 1.7, *Platform* Node.js versi 6.7.0, *framework*: Express.js 4.14.1, dan *test* API Postman versi 5.0.2.

# HASIL DAN PEMBAHASAN

## Tahapan Penelitian

Proses pengembangan API dimulai pada bulan Februari 2017 dengan menggunakan arsitektur yang telah dikembangkan oleh tim. Metode yang digunakan, yaitu *extreme programming*. API yang dihasilkan dapat diakses oleh *front end* yang merupakan aplikasi berbasis web dan *mobile*. Tahapan yang metode *extreme programming* yang dilakukan oleh tim *back end* yaitu, *planning*, *design*, *coding*, dan *testing*. Keempat tahapan tersebut dilakukan iterasi sebanyak tiga kali.

Tabel 1 Tahapan pengembangan setiap iterasi

|  |  |  |
| --- | --- | --- |
| Iterasi | *Task* | Deskripsi |
| Pertama | *Reengineering* modul laporan harga, komoditas, dan operasi pasar | Perubahan basis data relasional MySQL ke basis data non relasional MongoDB. |
| Kedua | Pengembangan modul *email* | Pengguna mendapatkan notifikasi lupa *password* melalui *email*. |
| Ketiga | Pengembangan modul lokasi | Lokasi yang terdiri atas provinsi, kabupaten, kecamatan, dan kelurahan. |

**Iterasi Pertama**

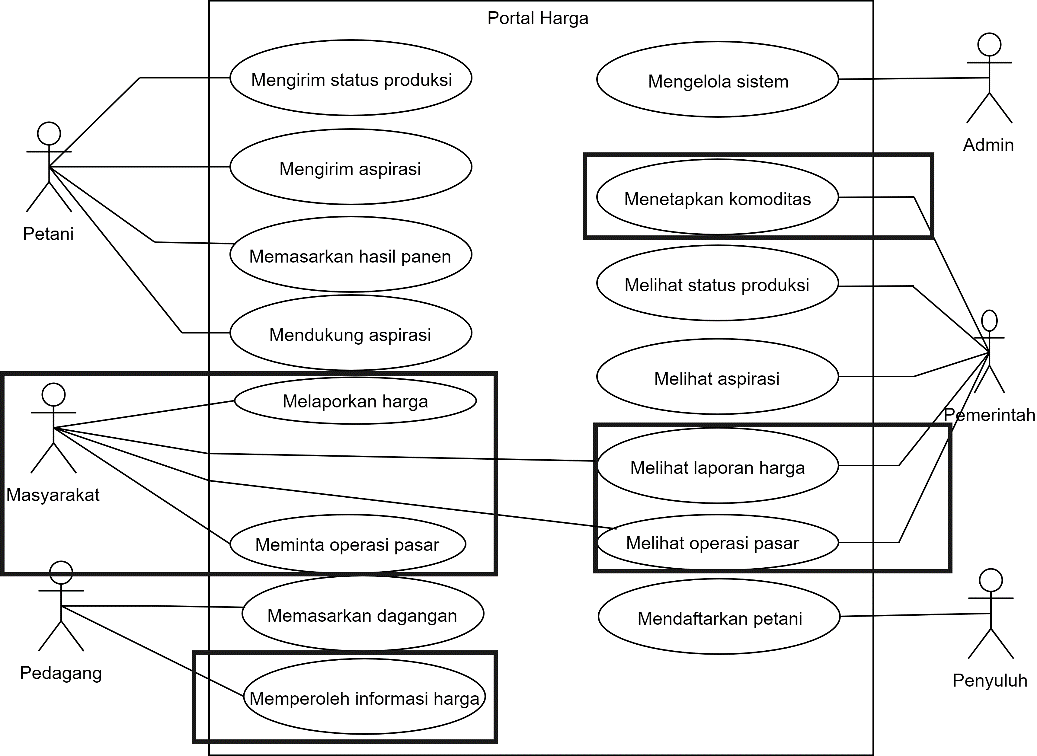
**Perencanaan (*Planning*)**

Iterasi pertama melakukan *reengineering* modul dan fungsi–fungsi dasar yang dilakukan oleh Setyatama (2016)*.* Modul yang dikembangkan penulis, yaitu modul komoditas, laporan harga, dan operasi pasar. Modul tersebut diakses oleh *stakeholder* pemerintah, masyarakat, dan pedagang. Fungsi dasar yang akan dibuat adalah *create*, *read*, *update*, dan *delete* untuk masing-masing modul.

*User story* dan *use case diagram* dibuat sesuai dengan modul dan *stakeholder* terkait. Salah satu *user story* yang dikembangkan oleh Setyatama (2016), yaitu *user story* masyarakat yang dapat dilihat pada Tabel 2. Detil deskripsi hasil pengembangan *user story* dapat dilihat pada Lampiran 1. *User story* tersebut akan menjadi alur penggunaan sistem bagi pengguna. *Use case diagram* untuk mendefinisikan kebutuhan fungsional masing-masing *stakeholder* dapat dilihat pada Gambar 4. Pada *use case diagram* yang diberi kotak hitam merupakan fungsi tiap modul yang mengalami *reengineering* dan pengembangan. *Login* dilakukan oleh setiap *stakeholder* sebelum menjalankan fungsi–fungsi yang ada di dalam sistem. *Stakeholder* yang belum memiliki *account* dapat melakukan *register* untuk membuat *account*.

Tabel 2 *User story* masyarakat

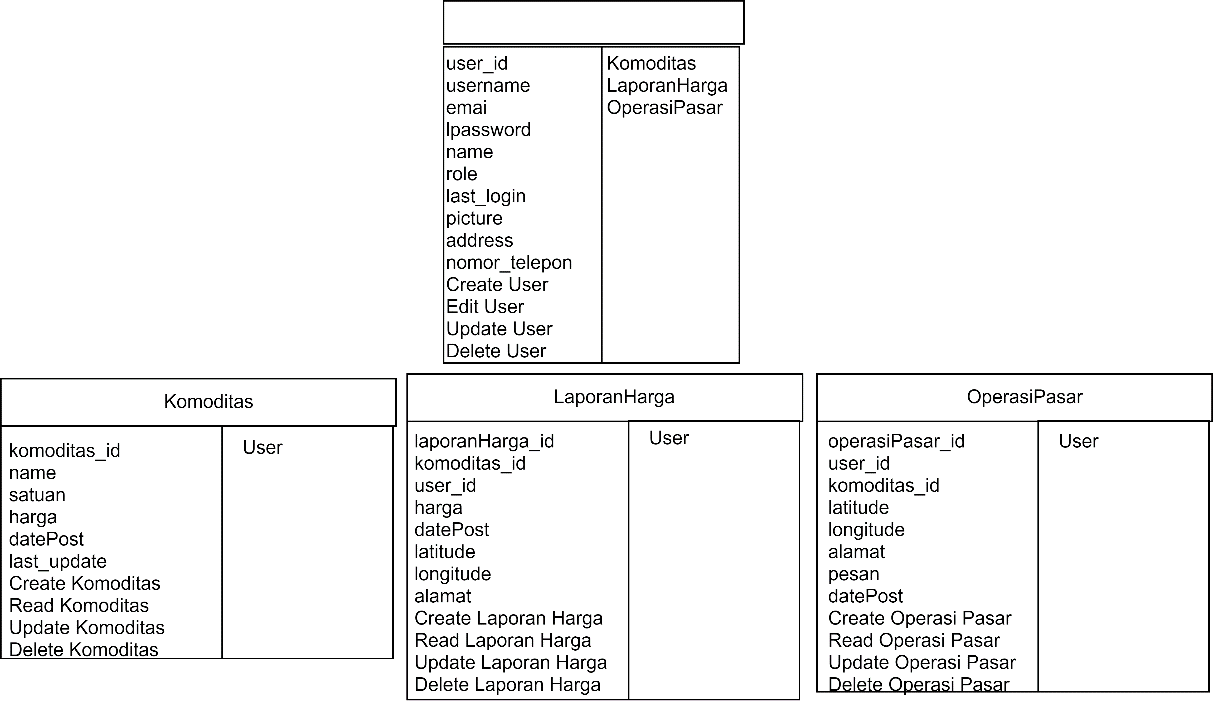
|  |  |  |
| --- | --- | --- |
| *Actor* | *Task* | *User story* |
| Masyarakat | Memberikan laporan harga | Pengguna memberikan laporan harga ketika berada pada suatu lokasi pasar. Pengguna memasukan harga suatu komoditas, sehingga bisa diketahui oleh pengguna lain. |
|  | Melihat laporan harga | Masyarakat memperoleh informasi harga komoditas beserta lokasinya. Informasi tersebut berasal dari masukan pengguna lain yang sudah melaporkan informasi harga pada suatu lokasi, sehingga masyarakat bisa mendapatkan harga yang sesuai dengan kebutuhan. |
|  | Meminta operasi pasar | Harga komoditas yang tinggi pada suatu lokasi, maka masyarakat dapat memberi pesan kepada pemerintah untuk melakukan operasi pasar untuk suatu komoditas pada suatu lokasi tersebut. |
|  | Melihat operasi pasar | Masyarakat dapat melihat operasi pasar yang telah diminta oleh pengguna lain sehingga pengguna tidak akan berbelanja suatu komoditas pada lokasi tersebut karena harga yang tinggi. |



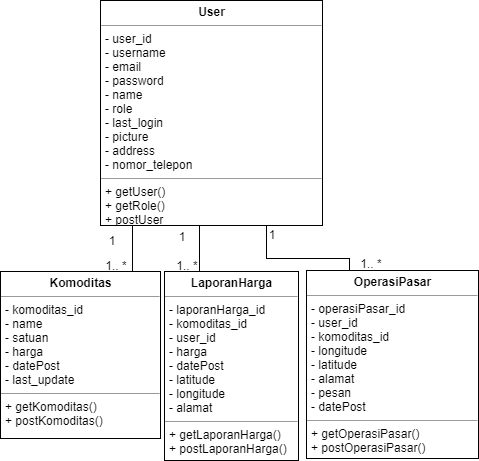
Gambar 4 *Use case diagram* iterasi pertama

**Desain (*Design*)**

*Class Responsibility Collaborator* (CRC) *cards* dirancang menjadi *class diagram* sesuai dengan modul dan *stakeholder* yang sudah ditentukan yang dapat dilihat pada Gambar 5. CRC *cards* memberikan gambaran atribut yang akan dimiliki oleh sebuah *class,* apa saja yang mampu dilakukan oleh sebuah *class,* dan *class* lain yang dapat mengakses suatu *class.*



Gambar 5 CRC *cards* iterasi pertama



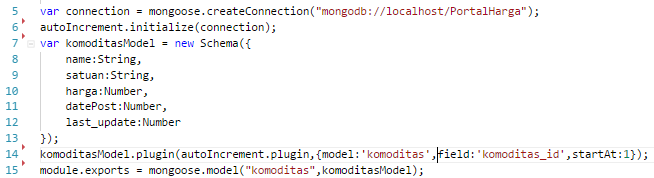
Gambar 6 *Class diagram* iterasi pertama

Tahap desain juga melakukan perancangan *sequence diagram* dan *collaboration diagram*.*Sequence diagram* menunjukkan aliran fungsionalitas yang disusun berdasarkan urutan waktu. Detil *sequence diagram* iterasi pertama dapat dilihat pada Lampiran 2. *Collaboration diagram* menunjukkan informasi yang sama persis dengan *sequence diagram*, namun lebih menunjukkan pada hubungan antar *object*. *Collaboration diagram* iterasi pertama dapat dilihat di Lampiran 3.

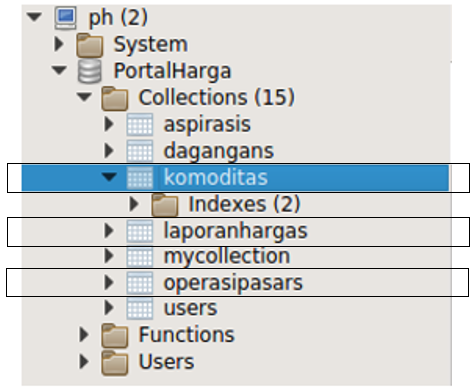
*Class diagram* iterasi pertama terdiri atas *class* komoditas, laporan harga, dan operasi pasar. *Class diagram* iterasi pertama dapat dilihat pada Gambar 6. *Collections* komoditas memiliki *field* komoditas\_id, *name*, satuan, harga, datePost, dan last\_update yang dapat dilihat pada Gambar 7. Komoditas\_id merupakan *field primary key* yang bersifat *auto increment*. Atribut *name*, satuan, dan harga merupakan nama komoditas, satuan komoditas, dan harga komoditas yang berasal dari masukan yang dilakukan oleh pemerintah. DatePost adalah waktu saat memasukan data. Last\_update adalah waktu terakhir saat memperbarui data komoditas. Daftar *collections* iterasi pertama dapat dilihat pada Lampiran 4.

*Class diagram* diimplementasikan pada basis data non relasional MongoDB. Teknik yang digunakan untuk mengubah *class* menjadi *collections* adalah *Object Document Mapper* (ODM). *Class* yang didefinisikan mewakili sebuat *document* yang otomastis akan tersimpan ke dalam sebuah *collections* (MongoDB 2017). Gambar daftar *collections* iterasi pertama dapat dilihat pada Gambar 8.

Penentuan atribut pada sebuah *class* menggunakan konsep *Big*-*O*. Hal tersebut bertujuan untuk mengatasi segala kemungkinan atribut yang dimiliki oleh *object* termasuk turunan dari sebuah *class*. *Class user* atributnya menggunakan konsep *Big*-*O* untuk menampung segala kemungkinan atribut yang dimiliki oleh setiap jenis *user.* Detil penjelasan atribut pada setiap *collections* dapat dilihat di kamus data pada Lampiran 5.



Gambar 7 *Collections* komoditas



Gambar 8 Daftar *collections* di basis dataiterasi pertama

**Pengodean (*Coding*)**

*Method* yang digunakan untuk mengakses REST API, yaitu POST dan GET. *Port* yang digunakan untuk mengakses REST API adalah 5000. Penggunaan *method* POST dan GET diatur di dalam *route*. *Method* POST digunakan untuk memasukan data ke dalam *basis data*, sedangkan *method* GET untuk memperoleh data dari basis data. *Method* POST data yang dimasukan diletakkan di *body.* Format keluaran setelah mengakses API dengan *method* POST atau GET terdiri JSON *array* yang diawali dan diakhiri dengan kurung siku (“[]”) dan JSON *object* yang diawali dan diakhiri dengan kurung kurawal (“{}”). Format keluaran JSON terdiri dari status, data, *message*, dan token. Status merupakan respon yang diberikan oleh server setelah dilakukan *request*. Status berisi nilai *integer* yang merupakan respon yang diberikan oleh *server* terhadap *request* yang dilakukan, 200 berarti suskes, 204 berarti data kosong, 400 berarti *bad request* karena kesalah yang dilakukan pengguna saat memasukan data, 401 berarti *unauthorized* yang disebabkan oleh token yang tidak valid, dan 403 berarti *forbidden* karena pengguna tidak mempunyai hak untuk akses API*.*

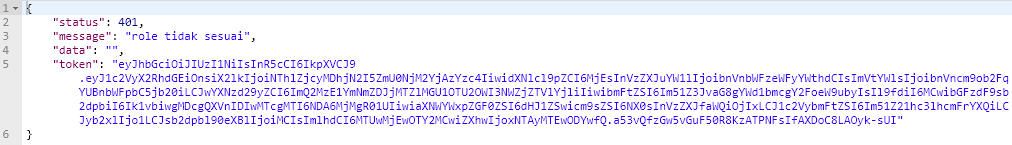
Tahap *coding* melakukan pemisahan token dengan *string* Bearer. Token yang dikirim oleh *front end* berbasis web maupun *mobile* ketika mengakses API terdapat *string* Bearer. *String* Bearer muncul karena token yang dikirim untuk mengakses API berada pada *header* dengan *key authorization.*

Penelitian sebelumnya yang dilakukan oleh Setyatama (2016) menggunakan token yang masih bersifat statis, maka pada penelitian ini digunakan token JWT (JSON Web Token) yang bersifat dinamis. Token berfungsi untuk *security* data diletakkan di *header*. Token pertama kali diberikan ketika *user* melakukan *register* atau *login*. Token yang diberikan merupakan hasil *encode* dari data yang diperlukan untuk *request* API. *Front end mobile* memiliki status *login\_type* = 1 yang berarti token tidak ada waktu kedaluwarsanya tetapi ketika *user logout* dari sistem maka token akan masuk ke dalam daftar *blacklist*. Nilai *front end website* status *login\_type* = 0 yang berarti token memiliki waktu kedaluwarsa enam puluh menit, dan setiap kali melakukan *request* mendapatkan token baru. Token baru diberikan ketika *login\_type* = 1, sedangkan untuk *login\_type* = 0 nilai token baru yaitu “-”.

**Pengujian (*Testing*)**

Pengujian yang dilakukan, yaitu pengujian fungsi *create*, *read*, *update*, dan *delete* untuk setiap modul. Pengujian pada iterasi pertama dilakukan sesuai dengan *acceptance test criteria* iterasi pertama yang dapat dilihat pada Lampiran 6. Hasil pengujian pada iterasi pertama dapat dilihat pada Tabel 3.

*Unit test* dilakukan pada pengujian *role* pengguna. Berdasarkan hasil diskusi dengan tim, *role* setiap *stakeholder* sebagai berikut: *role* 1 adalah admin, *role* 2 adalah pemerintah, *role* 3 adalah penyuluh, *role* 4 adalah petani, *role* 5 adalah masyarakat, dan *role* 6 adalah pedagang*.* Salah satu pengujian *role* yaitu menambah komoditas dengan *role* masyarakat yang dapat dilihat pada Gambar 9. *Role* 3 tidak dapat menambah komoditas karena *role* yang berhak untuk menambah komoditas yaitu *role* 1 atau *role* 2.



Gambar 9 *Unit test* menambah komoditas

Salah satu pengujian pada modul laporan harga, yaitu pengujian fungsi melihat laporan harga yang dapat dilihat pada Gambar 10. Pengujian fungsi melihat laporan harga dengan mengakses URL https://ph.yippytech.com:5000/laporanHarga/get menggunakan *method* GET. Format keluaran berupa JSON yang terdiri atas status, *message*, data, dan token. Data berisi semua laporan harga yang masing-masing laporan harga terdiri atas laporanHarga\_id, datePost, user\_id, komoditas\_id, harga, *latitude*, *longitude*, alamat, namaKomoditas, satuan, dan nama pengirim laporan harga.



Gambar 10 Pengujian fungsi melihat laporan harga

Tabel 3 Hasil pengujian iterasi pertama

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class* | *Method* | Fungsi | URL | Status |
| Komoditas | POST | Menambah komoditas | komoditas/add | Berhasil |
|  | GET | Melihat komoditas | komoditas/get | Berhasil |
| Komoditas | POST | Mengubah komoditas | komoditas/update | Berhasil |
|  | POST | Menghapus komoditas | komoditas/delete | Berhasil |
| Laporan harga | POST | Menambah laporan harga | laporanHarga/add | Berhasil |
|  | GET | Melihat laporan harga | laporanHarga/get | Berhasil |
|  | POST | Mengubah laporan harga | laporanHarga /update | Berhasil |
|  | POST | Menghapus laporan harga | laporanHarga  /delete | Berhasil |
| Operasi pasar | POST | Menambah operasi pasar | operasiPasar/add | Berhasil |
|  | GET | Melihat operasi pasar | operasiPasar/get | Berhasil |
|  | POST | Mengubah operasi pasar | operasiPasar  /update | Berhasil |

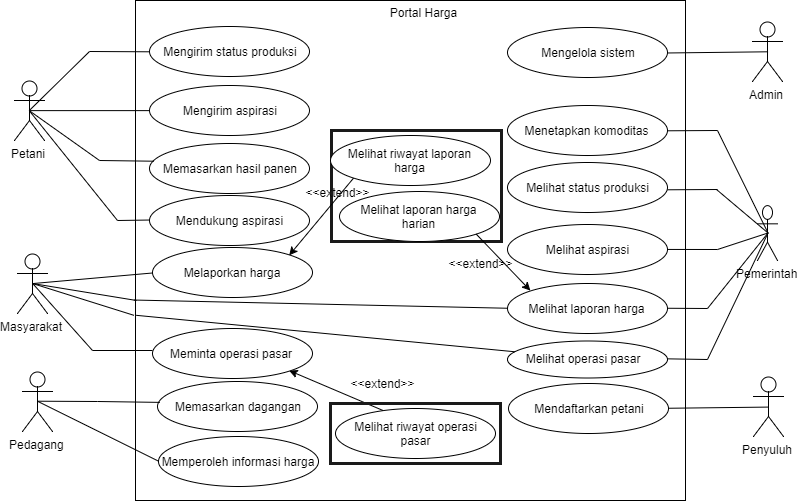
Tabel 3 Lanjutan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class* | *Method* | Fungsi | URL | Status |
| Operasi pasar | POST | Menghapus operasi pasar | operasiPasar  /delete | Berhasil |

**Iterasi Kedua**

**Perencanaan (*Planning*)**

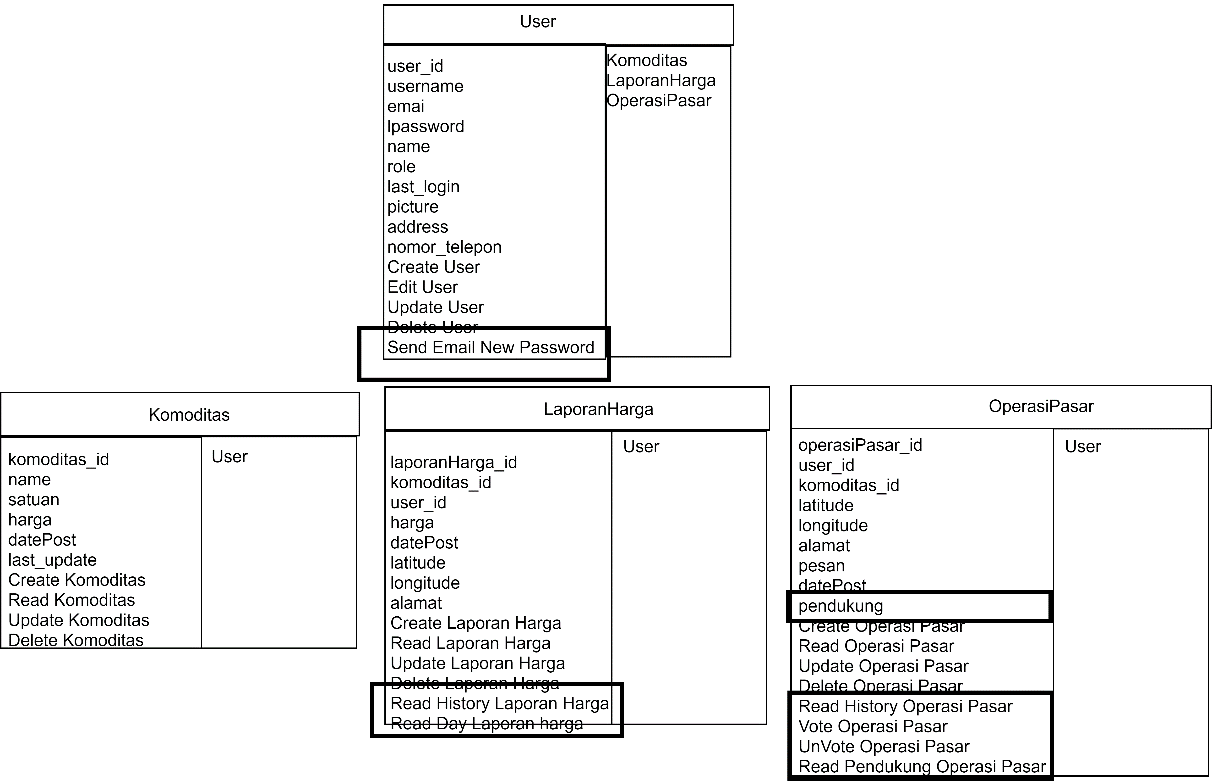
Modul operasi pasar dikembangkan sehingga pengguna dapat mendukung operasi pasar lain. Modul laporan harga dikembangkan sehingga dapat melihat laporan harga pada beberapa hari sebelumnya. Modul operasi pasar dan laporan harga dapat melihat *history* pengguna. Penelitian sebelumnya yang dilakukan oleh Mardika (2016) memiliki kekurangan, yaitu kendala apabila pengguna lupa *password*. Pengembangan modul *email* untuk mengatasi kekurangan tersebut. Perubahan *use case diagram* pada iterasi kedua dapat dilihat pada Gambar 11. Fungsi tiap modul yang dikembangkan di iterasi kedua pada *use case diagram* diberi kotak hitam.



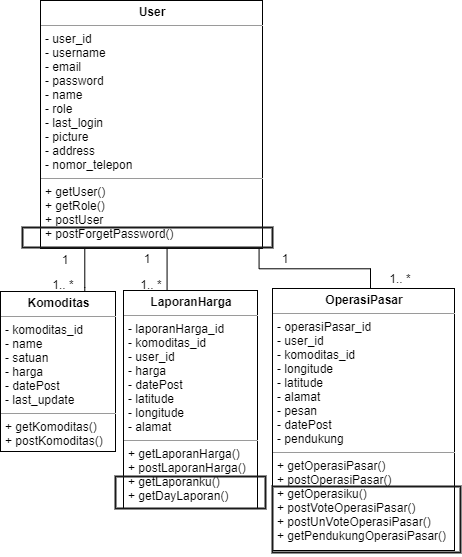
Gambar 11 *Use case diagram* iterasi kedua

**Desain (*Design*)**

*Class Responsibility Collaborator* (CRC) *cards* dirancang menjadi *class diagram* sesuai dengan modul pada tahap perencanaan yang dapat dilihat pada Gambar 12.Perancangan *class* *diagram* sesuai dengan CRC *cards*. *Class* diagram iterasi kedua dapat dilihat pada Gambar 13. Pembuatan *sequence* *diagram* disesuaikan dengan fungsi yang ditambahkan pada *class* *diagram*. Detil *sequence diagram* iterasi kedua dapat dilihat pada Lampiran 7. *Collaboration diagram* untuk menunjukkan hubungan antar *object* yang ada pada *sequence diagram* dapat dilihat pada Lampiran 8.



Gambar 12 CRC *cards* iterasi kedua



Gambar 13 *Class diagram* iterasi kedua

*Class* operasi pasar memiliki empat fungsi yang dikembangkan, yaitu fungsi *vote*, *unvote*, getPendukung, dan operasiku. Fungsi *vote* dan *unvote* untuk mendukung dan membatalkan dukungan. Fungsi getPendukung untuk melihat siapa saja yang mendukung suatu operasi pasar. Fungsi operasiku untuk melihat *history* operasi pasar pengguna. *Class* laporan harga terdapat dua fungsi yang dikembangkan, yaitu fungsi laporanku dan getDayLaporan. Fungsi laporanku untuk melihat *history* laporan harga pengguna. Fungsi getDayLaporan untuk mendapatkan laporan harga beberapa hari sebelumnya. *Class email* dikembangkan satu fungsi yaitu forgetPassword yang memberikan *password* baru kepada pengguna yang dikirimkan melalui *email*.

**Pengodean (*Coding*)**

Iterasi kedua menambahkan *field* pendukung pada *collections* operasi pasar dengan tipe data *array* untuk menampung daftar pendukung. Ketika seorang pengguna mendukung suatu operasi pasar maka *user\_id* pengguna tersebut akan dimasukan ke dalam *field* *array* pendukung. Apabila seorang pengguna batal mendukung suatu operasi pasar maka *user\_id* pengguna tersebut akan dikeluarkan dari *field array* pendukung.

Tahap pengodean menggunakan *framework* Express.js yang bersifat *asynchronous*. *Asynchronous* dapat diatasi dengan menggunakan *time out* yang dapat dilihat pada Gambar 14*.* Salah satu fungsi yang menggunakan *time out,* yaitu fungsi getDayLaporan dengan *time out* selama 400 *miliseconds*. *Time out* selama 300 *miliseconds* untuk mendapatkan id\_laporan pada beberapa hari sebelumnya kemudian dimasukan ke dalam *array*. Sisa *time out* 100 *miliseconds* untuk mendapatkan data laporan harga sesuai id\_laporan yang sudah disimpan dalam *array*.

//mengambil laoranHarga beberapa hari yang lalu, komoditasnya apa aja selama hari itu

var dayLaporan = function(req,res){

    //ambil semua laporanHarga di sorting sesuai dengan tanggal post

    laporanHarga.find({},'-\_id -\_\_v',{sort:{datePost:-1}},function(err,all){

        if(all==null){

            res.json({status:204,message:err,data:"",token:req.token});

        }else{

            //tanggal sekarang

            var dateNow = new Date();

            //tanggal sekarang di kurangi hari yang diinginkan, hari nya

            dateNow.setDate(dateNow.getDate() - req.params.day);

            //hari yang diinginkan dalam format, hari, tanggal, bulan, dan tahun

            var getDate = dateFormat(dateNow, "dddd , mmmm dS , yyyy");

            //console.log(getDate);

            //buat variabel parsing yang akan menerima laporanHarga\_id pada hari itu

            var parsing = [];

            var number = [];

            var counter = 0;

            for(var i=0;i<all.length;i++){

                if(dateFormat(all[i].datePost, "dddd , mmmm dS , yyyy")==getDate){

                    number.push(all[i].laporanHarga\_id);

                };

            }

            //time out 300 miliseconds

            setTimeout(function () {

                for(var i=0;i<number.length;i++){

          laporanHarga.findOne({laporanHarga\_id:number[i]},'-\_id –

\_\_v').lean().exec(function(err,laporan){

                        komoditas.findOne({komoditas\_id:laporan.komoditas\_id}).exec(function(err,k

omo){

                            user.findOne({user\_id:laporan.user\_id},function(err,masyarakat){

                                laporan.namaKomoditas=komo.name;

                                laporan.satuan = komo.satuan;

                                laporan.nama = masyarakat.name;

                                laporan.datePost = moment(laporan.datePost).format("YYYY-MM-DD");;

                                parsing.push(laporan);

                            })

                        })

                    })

                }

            }, 300);

            //time out 400 mili seconds

            setTimeout(function () {

                res.json({

                    status:200,

                    message:"sukses mendapat laporan harga " + req.params.day + ' hari sebelumnya',

                    data:parsing,

                    token:req.token

                });

            }, 400);

Gambar 14 Potongan *code* fungsi getDayLaporan

Implementasi modul *email* menggunakan *nodemailer* yang merupakan Gmail Google. Fungsi forgetPassword ketika diakses akan membuat *password* baru dan memperbarui *password* pengguna yang lama. *Password* baru dikirimkan ke alamat *email* pengguna yang merupakan hasil *random* *string*. *Random string* yang dihasilkan sepanjang lima belas karakter.

**Pengujian (*Testing*)**

Fungsi yang dikembangkan pada modul operasi pasar, laporan harga, dan *email* dilakukan pengujian. Pengujian pada iterasi kedua dilakukan sesuai dengan *acceptance test criteria* iterasi kedua yang dapat dilihat pada Lampiran 9. Contoh pengujian, yaitu fungsi forgetPassword yang dilakukan dengan *login* menggunakan *password* baru yang dikirimkan melalui *email*. Hasil pengujian pada iterasi kedua dapat dilihat pada Tabel 4.

Salah satu pengujian pada modul laporan harga, yaitu pengujian fungsi melihat laporan harga beberapa hari sebelumnya yang dapat dilihat pada Gambar 15. Contoh pengujian, yaitu melihat laporan harga 61 hari sebelumnya dengan mengakses URL https://ph.yippytech.com:5000/laporanHarga/get/day/61 menggunakan *method* GET. Angka 60 merupakan sebuah parameter berapa hari sebelumnya yang dapat diganti sesuai dengan kebutuhan. Data berisi laporan harga yang dilaporkan 61 hari sebelumnya yang masing-masing laporan harga terdiri atas laporanHarga\_id, datePost, user\_id, komoditas\_id, harga, *latitude*, *longitude*, alamat, namaKomoditas, satuan, dan nama pengirim laporan harga.



Gambar 15 Pengujian fungsi melihat laporan harga 61 hari sebelumnya

Tabel 4 Hasil pengujian iterasi kedua

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class* | *Method* | Fungsi | URL | Status |
| Laporan harga | POST | Mendukung operasi pasar | operasiPasar/pendukung/add | Berhasil |

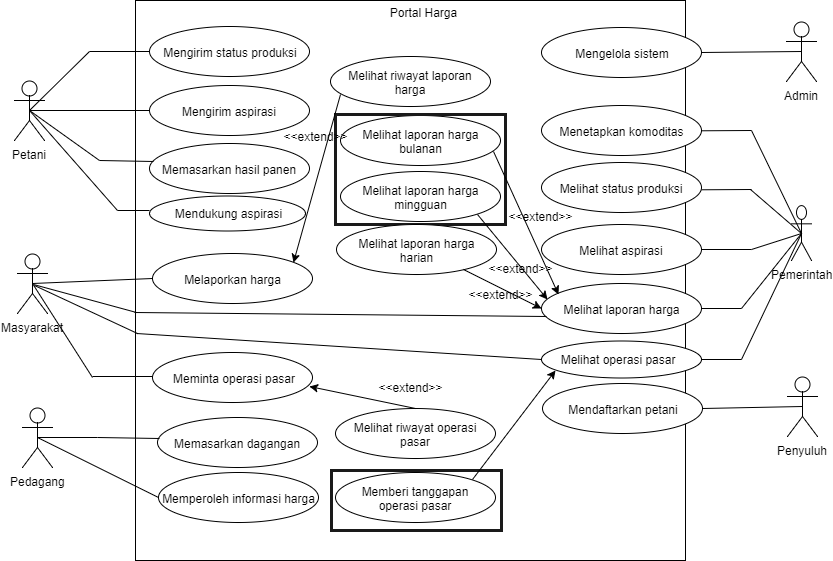
Tabel 4 Lanjutan

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class* | *Method* | Fungsi | URL | Status |
|  | POST | Membatalkan dukung operasi pasar | operasiPasar/pendukung/delete | Berhasil |
|  | GET | Melihat pendukung operasi pasar | operasiPasar/pendukung/get/:operasiPasar\_id | Berhasil |
|  | GET | Melihat *history* laporan harga | operasiPasar/operasi/get/:user\_id | Berhasil |
| Operasi pasar | GET | Melihat *history* operasi pasar | laporanHarga/get/laporan/:user\_id | Berhasil |
|  | GET | Melihat laporan harga beberapa hari sebelumnya | laporanHarga/get/day/:day | Berhasil |
| Mail | POST | Mengirim password | email/forgetPassword | Berhasil |

**Iterasi Ketiga**

**Perencanaan (*Planning*)**

Iterasi ketiga mengembangkan laporan harga, operasi pasar, *email*, dan merancang modul lokasi. Modul laporan harga dapat melihat laporan harga selama satu minggu ataupun selama satu bulan. Modul operasi pasar pengguna dapat memberikan tanggapan. Modul lokasi bertujuan untuk memberikan pilihan lokasi kepada pengguna pada saat *register* atau saat memberikan informasi laporan harga atau operasi pasar sesuai dengan lokasi pengguna berada.

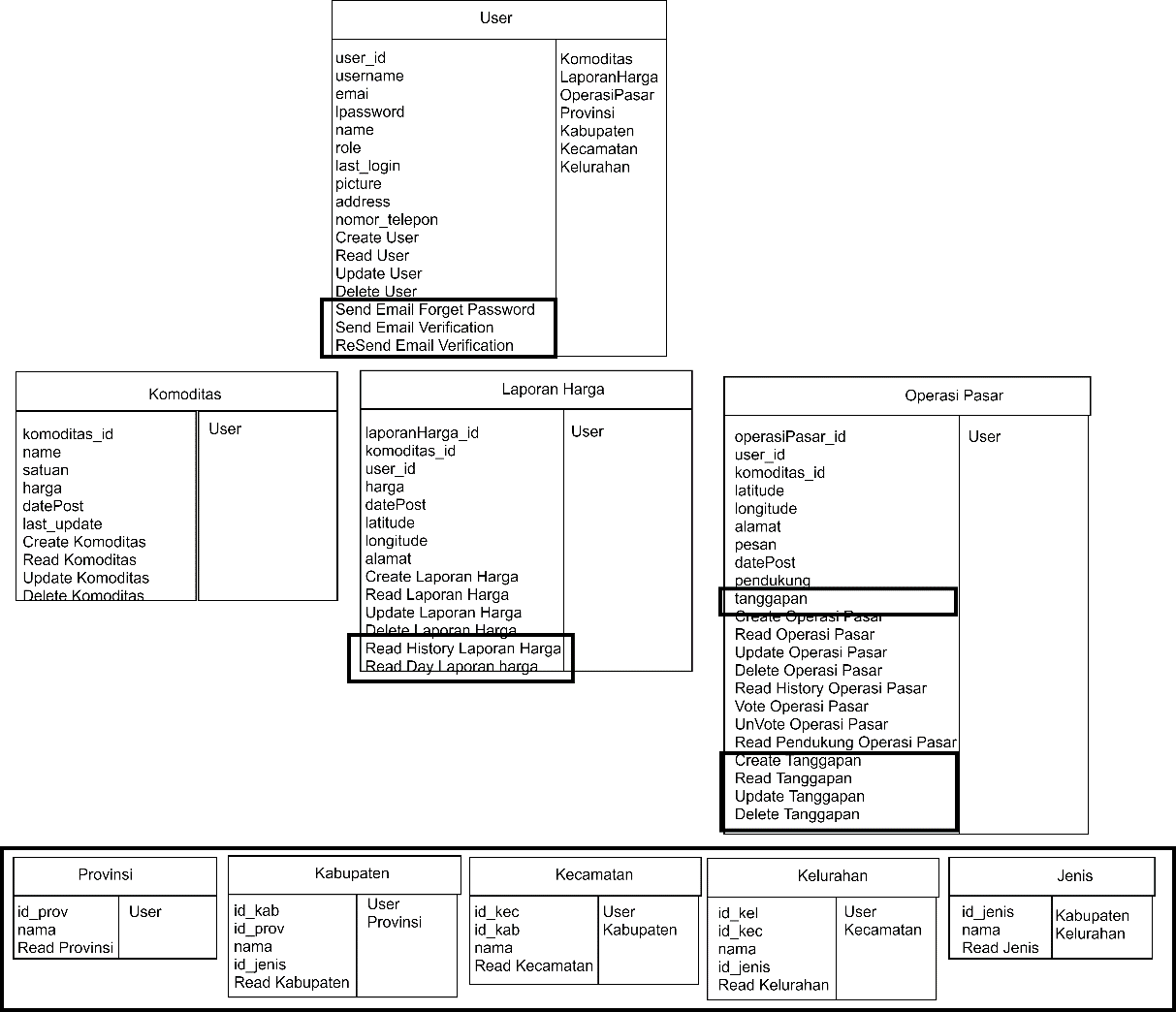


Gambar 16 *Use case diagram* iterasi ketiga

Modul *email* dikembangkan untuk verifikasi *account*. Verifikasi *account* dikirim melalui *email* dan jika tidak diverifikasi maka tidak bisa *login*. Pengguna yang melakukan *register* secara otomatis akan mendapatkan *email* yang berisi *link* untuk verifikasi *account*. Apabila selama satu jam pengguna tidak memverifikasi *account* maka *link* akan kedaluwarsa. Pengguna dapat meminta kembali *email* yang berisi *link* verifikasi dengan memasukan *username*. Detil *use case diagram* iterasi ketiga dapat dilihat pada Gambar 16. Fungsi tiap modul yang dikembangkan di iterasi ketiga pada *use case diagram* diberi kotak hitam.

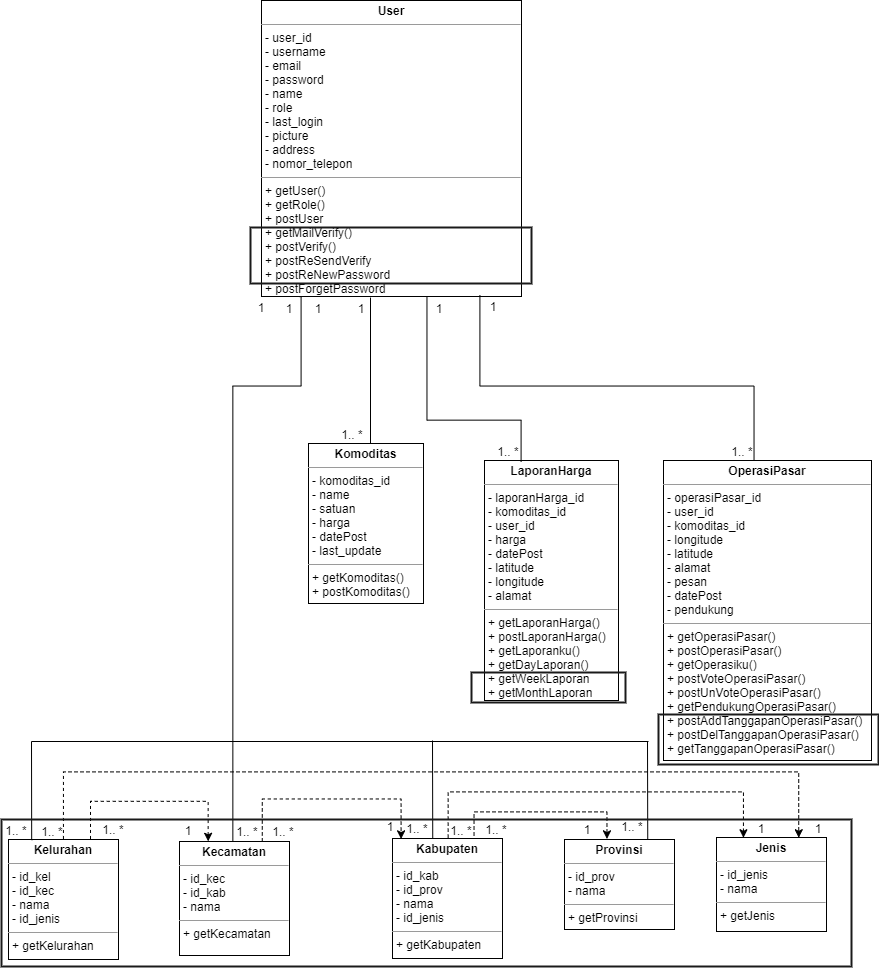
**Desain (*Design*)**

CRC *cards* dirancang menjadi *class diagram* sesuai modul yang dikembangkan pada tahap perencanaan yang dapat dilihat pada Gambar 17. *Class* *diagram* iterasi ketiga dapat dilihat pada Gambar 18. *Sequence diagram* dirancang sesuai dengan hubungan antar *object–object* yang dikembangkan pada *class diagram*. Detil *sequence* *diagram* iterasi ketiga dapat dilihat pada Lampiran 10. *Collaboration diagram* iterasi ketiga dirancang sesuai dengan *object-object* yang ada pada *sequence diagram*. Detil *Collaboration diagram* iterasi ketiga dapat dilihat pada Lampiran 11.



Gambar 17 CRC *cards* iterasi ketiga

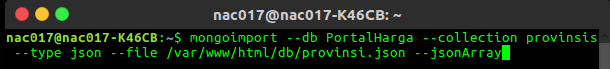
Modul lokasi terdiri atas *class* jenis, provinsi, kabupaten, kecamatan, dan kelurahan. *Class* kelurahan memiliki ketergantungan terhadap *class* kecamatan, *class* kecamatan memiliki ketergantungan terhadap *class* kabupaten, dan *class* kabupaten memiliki ketergantungan terhadap *class* provinsi*.* Modul laporan harga terdapat dua fungsi yang dikembangkan, yaitu fungsi getWeek dan getMont. Modul operasi pasar terdapat tiga fungsi yang dikembangkan, yaitu addTanggapan untuk memberikan tanggapan, delTanggapan untuk menghapus tanggapan, dan getTanggapan melihat siapa saja yang memberikan tanggapan. Modul *email* terdapat empat fungsi yang dikembangkan, yaitu getMailVerify, postVerify, postReSendVerify, dan postReNewPassword. Fungsi getVerify untuk mengirim *email* verifikasi.



Gambar 18 *Class* *diagram* iterasi ketiga

**Pengodean (*Coding*)**

Iterasi ketiga membuat API untuk modul lokasi menggunakan data lokasi seluruh Indonesia dalam format SQL. Data format SQL kemudian diubah menjadi format JSON untuk masing-masing tabel menggunakan phpMyAdmin. Data dalam format JSON kemudian dimasukan ke dalam basis data MongoDB dengan menggunakan *terminal* untuk masing-masing tabel menjadi *collections* yang bersesuaian. *Query* yang digunakan untuk memasukan data ke dalam *collections* MongoDB adalah *mongoimport.* Contoh *query* memasukan data provinsi ke dalam basis data MongoDB yaitu dapat dilihat pada Gambar 19. *Query* tersebut memasukan data provinsi dalam format JSON yang berada dalam folder var/www/html/db ke dalam *collections* provinsis di basis data PortalHarga.



Gambar 19 *Code* memasukan data format JSON ke basis data mongoDB

*Pair programming* dilakukan antara fungsi getMailVerify dengan fungsi *register* yang dapat dilihat pada Gambar 20. *Email* yang dikirim ke pengguna berisi *link* yang diintegrasikan dengan fungsi postVerify. Fungsi postMailVerifyuntuk mengubah *field* isValidatepada *collections user* agar bernilai *true* yang memiliki nilai *default false.*



Gambar 20 Integrasi fungsi getMailVerifydengan fungsi *register*

**Pengujian (*Testing*)**

Pengujian pada modul lokasi untuk mendapatkan provinsi seluruh Indonesia, kabupaten atau kota berdasarkan provinsi yang dipilih, kecamatan berdasarkan kabupaten atau kota yang dipilih, dan kelurahan berdasarkan kecamatan yang dipilih. Modul *email* dilakukan pengujian dengan memverifikasi *account* melalui *link* yang dikirim ke pengguna melalui *email*. Pengujian pada iterasi ketiga dilakukan sesuai dengan *acceptance test criteria* iterasi ketiga yang dapat dilihat pada Lampiran 12. Detil pengujian iterasi ketiga dapat dilihat pada Tabel 5.

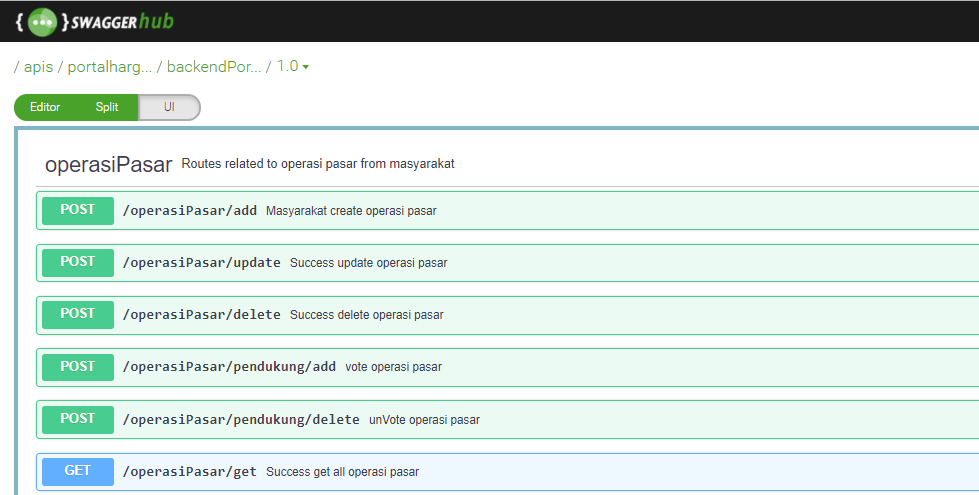
Salah satu pengujian pada modul lokasi, yaitu pengujian fungsi melihat provinsi di Indonesia. Pengujian fungsi melihat provinsi di Indonesia dengan mengakses URL https://ph.yippytech.com:5000/lokasi/provinsi menggunakan *method* GET. Data berisi informasi provinsi yang masing-masing terdiri dari id\_prov dan nama provinsi. API yang sudah dikembangkan didokumentasikan di https://app.swaggerhub.com/apis/portalharga/backendPortalHarga/1.0. Potongan dokumentasi API yang dapat dilihat pada Gambar 21.

Tabel 5 Hasil pengujian iterasi ketiga

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| *Class* | *Method* | Fungsi | URL | Status |
| Provinsi | GET | Melihat provinsi | lokasi/provinsi | Berhasil |
| Kabupaten | GET | Melihat kabupaten | lokasi/kabupaten/:id\_prov | Berhasil |
| Kecamatan | GET | Melihat kecamatan | lokasi/kecamatan/:id\_kab | Berhasil |
| Kelurahan | GET | Melihat kelurahan | lokasi/kelurahan/:id\_kec | Berhasil |

Tabel 5 Lanjutan

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| *Class* | *Method* | | Fungsi | URL | | Status |
| Mail | GET | | Mengirim email verifikasi | /user/email/getMailVerify | | Berhasil |
|  | POST | | Mengirim ulang email verifikasi | /user/email/validate/resend | | Berhasil |
| Mail | POST | Mengirim *link* ke halaman *forget password* | | /user/email/forgetPassword | Berhasil | |
|  | POST | Memperbarui *password* | | /user/email/reNewPassword | Berhasil | |
| Laporan harga | GET | Melihat laporan harga minggu ini | | laporanHarga/getWeek | Berhasil | |
|  | GET | Melihat laporan harga bulan ini | | laporanHarga/getMonth | Berhasil | |
| Operasi pasar | POST | Memberi tanggapan operasi pasar | | operasiPasar/tanggapan/add | Berhasil | |
|  | POST | Menghapus tanggapan operasi pasar | | operasiPasar/tanggapan/detele | Berhasil | |
|  | GET | Melihat siapa saja yang memberi tanggapan | | operasiPasar/get/:operasiPasar | Berhasil | |



Gambar 21 Dokumentasi API di Swaggerhub

# SIMPULAN DAN SARAN

## Simpulan

Penelitian ini berhasil mengembangkan REST API menggunakan Node.js dan basis data non relasional dengan menggunakan metode *Extreme Programming* dan dilakukan iterasi sebanyak tiga kali. Pengembangan API mampu menghubungkan enam aktor, yaitu admin, pemerintah, penyuluh, petani, masyarakat, dan pedagang. API yang dihasilkan sudah terintegrasi dengan *email* dan lokasi seluruh Indonesia. API sudah dapat diakses oleh *front end* berbasis *mobile* maupun web*.*

## Saran

Penelitian selanjutnya pada tahap pengodean perlu dianalisis efisiensi algoritme agar *execution times* lebih cepat. Penambahan modul *region* agar pemerintah bisa memberikan patokan harga yang sesuai berdasarkan wilayahnya, melihat laporan harga sesuai dengan lokasi pengguna berada. Selain itu, penelitian selanjutnya dapat mengimplementasikan fitur JOIN yang ada pada basis data non relasional.

# DAFTAR PUSTAKA

Ahmad. 2015. Perancangan aplikasi komoditas pertanian berbasis android. *Jurnal CSRID*. 7(3): 190–200.

[APJII] Asosiasi Penyelenggara Jasa Internet Indonesia. 2016. Profil pengguna internet Indonesia 2016 [Internet]. [diunduh 2017 Aug 7]. Tersedia pada: https://apjii.or.id/downfile/file/BULETINAPJIIEDISI02Oktober2016.pdf.

Elian N, Lubis DP, Rangkuti PA. 2014. Penggunaan internet dan pemanfaatan informasi pertanian oleh penyuluh pertanian di Kabupaten Bogor wilayah Barat. *Jurnal Komunikasi Pembangunan.* 12(2):105–106.

[JSON] JavaScript Object Notation. 2002. Introduction JSON [internet]. [diunduh 2017 Jul 27]. Tersedia pada: http://json.org/json-id.html.

[Kementan] Kementrian Pertanian. 2015. Renstra kementan tahun 2015-2019 [Internet]. [diunduh 2017 Mei 13]. Tersedia pada: [http://www.pertanian.go.id/file/RENSTRA\_2015–2019.pdf](http://www.pertanian.go.id/file/RENSTRA_2015-2019.pdf).

Mahardika SB. 2016. Pengembangan REST API modul produsen komoditas pertanian pada aplikasi portal harga [skripsi]. Bogor (ID): Institut Pertanian Bogor.

[MongoDB]. 2017. MongoDB Documentation Release 3.0.7. [internet] [diunduh 2017 Aug 7] Tersedia pada https://docs.mongodb.com/v3.4/mongodb-manual-v3.4.epub.

Pautasso C. 2008. REST vs SOAP Making the Right Architectural Decision. Amsterdam (NL): SOA Symposium.

Pressman RS. 2010. *Software Engineering: A Practitioner's Approach.* Ed ke-7*.* Boston (US): Mc Graw Hill.

Romadhan M. 2009. Pola hubungan tengkulak dan petani [skripsi]. Medan (ID): Universitas Sumatera Utara.

Saputra ZR. 2015. Aplikasi SMS *Center* untuk informasi harga komoditi hasil pertanian Kabupaten Ogan Ilir. *Jurnal SIGMATA*. 4(2): 3–6.

Setiady H. 2013. Sistem informasi pemesanan dan penjualan berbasis web pada dewi florist [skripsi]. Palembang (ID): Sekolah Tinggi Ilmu Manajemen Informatika dan Komputer *Global Information* Multi Data Palembang.

Setiana D. 2016. Pengembangan modul pelaporan harga komoditas pertanian pada sisi pengguna admin dan pemerintah menggunakan REST API [skripsi]. Bogor (ID): Institut Pertanian Bogor.

Setyatama D. 2016. Pengembangan aplikasi pelaporan harga komoditas pertanian berbasis *mobile* menggunakan REST API [skripsi]. Bogor (ID): Institut Pertanian Bogor.

Lampiran 1 Deskripsi *user story*

1. Pemerintah

|  |  |  |
| --- | --- | --- |
| *Actor* | *Task* | *User story* |
| Pemerintah | Menetapkan komoditas | Pemerintah menentukan komoditas apa saja yang boleh masuk ke dalam sistem beserta harga dan satuannya |
|  | Melihat status produksi | Status produksi yang dikirimkan oleh petani dapat dilihat oleh pemerintah, sehingga produksi yang dihasilkan oleh petani dapat dipantau oleh pemerintah |
|  | Melihat operasi pasar | Pemerintah dapat melihat operasi pasar yang diminta oleh masyarakat sehingga pemerintah bisa mengatasi harga suatu komoditas di suatu wilayah tertentu |
|  | Melihat laporan harga | Laporan harga yang dikirim oleh masyarakat dapat dilihat oleh pemerintah untuk menentukan kebijakan harga selanjutnya |
|  | Melihat aspirasi | Aspirasi petani yang dikirim dapat dilihat oleh pemerintah sehingga pemerintah segera menangani kebutuhan petani yang ada di lapangan |

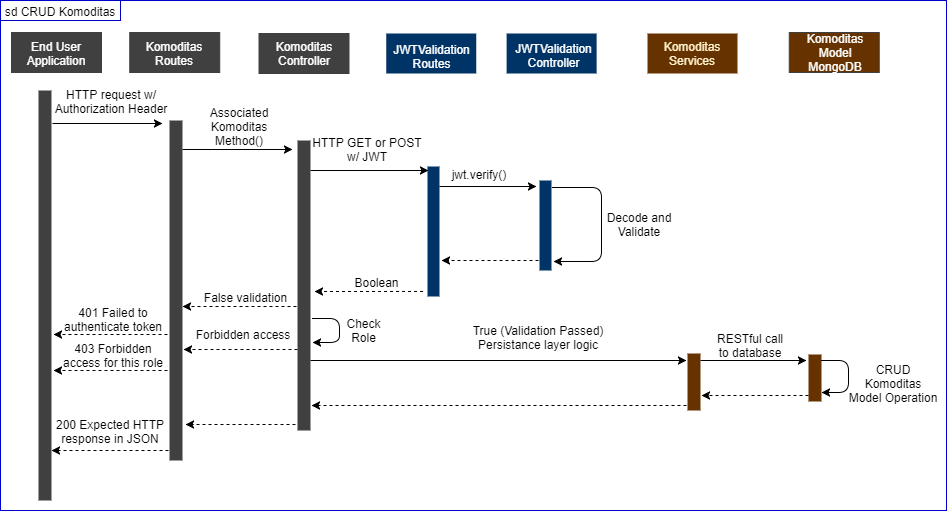
1. Pedagang

|  |  |  |
| --- | --- | --- |
| *Actor* | *Task* | *User story* |
| Pedagang | Melihat laporan harga | Pedagang dapat melihat laporan harga yang dikirimkan oleh masyarakat sehingga pedagang bisa menentukan harga jual untuk suatu komoditas |
|  | Memasarkan dagangan | Pedagang bisa memasarkan komoditas yang mereka dagangkan |

Lampiran 2 *Sequence diagram* iterasi pertama

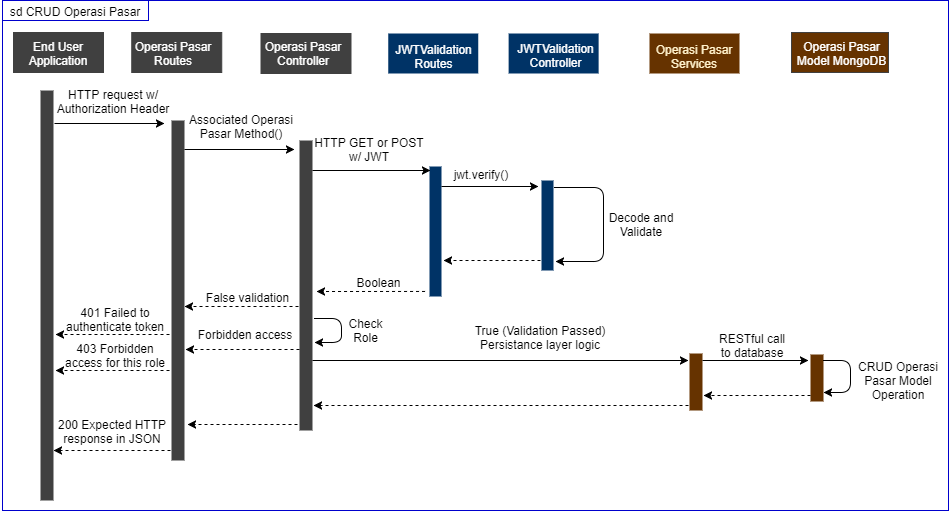
**Komoditas**

1. CRUD Komoditas



**Operasi pasar**

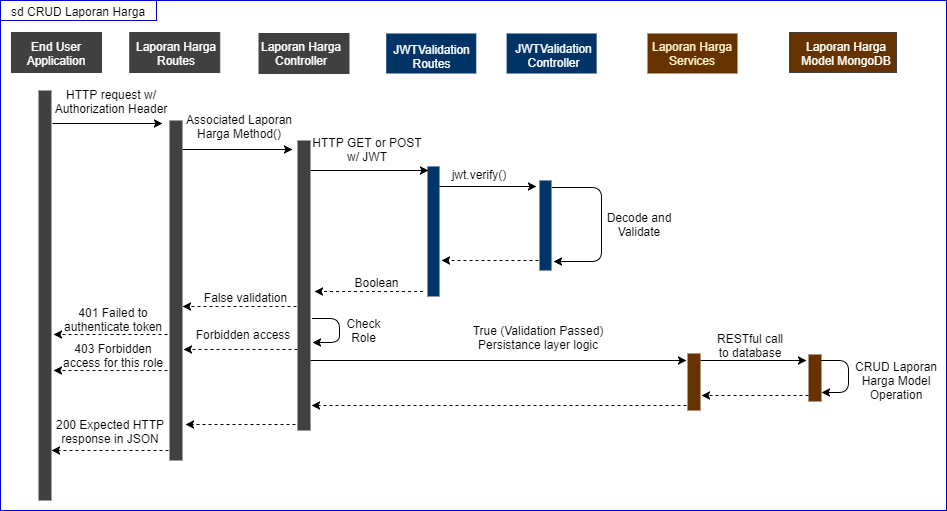
1. CRUD Operasi pasar



Lampiran 2 *Sequence diagram* iterasi pertama (lanjutan)

**Laporan harga**

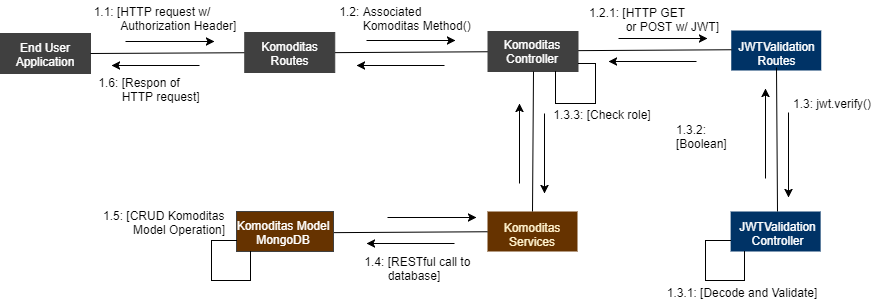
1. CRUD Laporan harga



Lampiran 3 *Collaboration diagram* iterasi pertama

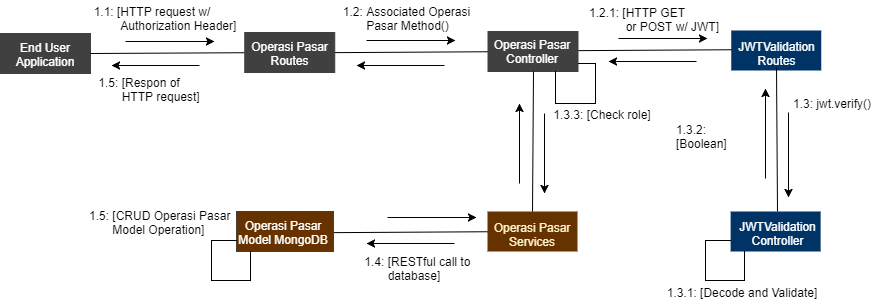
**Komoditas**

1. CRUD Komoditas



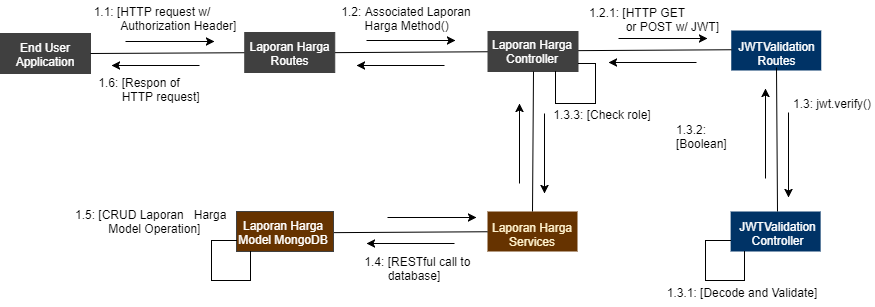
**Operasi pasar**

1. CRUD Operasi pasar



**Laporan harga**

1. CRUD Laporan harga

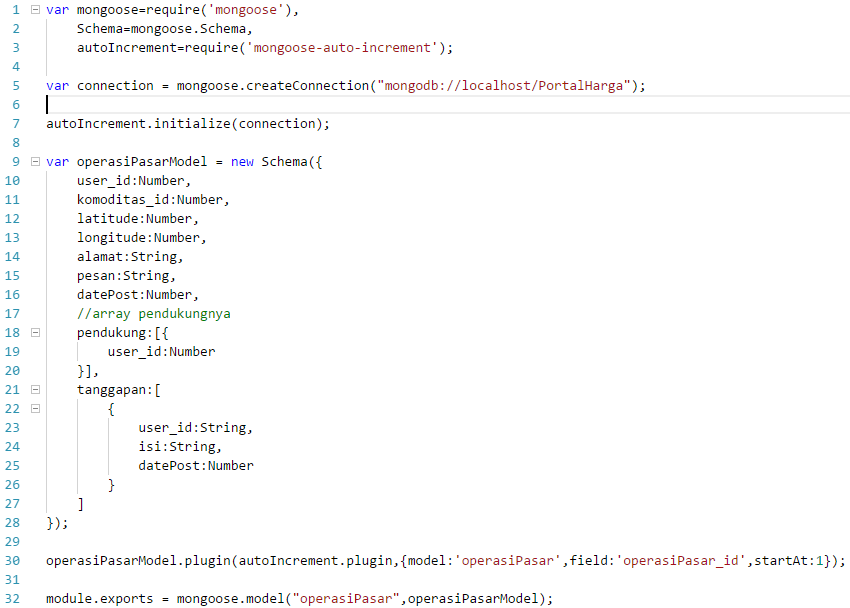


Lampiran 4 Daftar *collections* iterasi pertama

1. *Collections* laporan harga



1. *Collections* operasi pasar



Lampiran 5 Kamus data

**Komoditas**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field name | Caption | Data Type | Key Type | Constraints | Reference  Collections |
| komoditas\_ id |  | Autonumber | Primary Key |  |  |
| name | Name komoditas | String |  | Not Null |  |
| satuan | Satuan komoditas | String |  | Not Null |  |
| harga | Harga komoditas | Number |  | Not Null |  |
| datePost | Date input | Number |  |  |  |
| last\_update | Date update | Number |  |  |  |

**Laporan harga**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field name | Caption | Data Type | Key Type | Constraints | Reference Collections |
| laporanHarga\_id |  | Autonumber | Primary Key |  |  |
| komoditas\_id | Name  komoditas | Number | Foreign key | Not Null | komoditas |
| user\_id | Satuan komoditas | Number | Foreign key | Not Null | users |
| harga | Harga  komoditas | Number |  | Not Null |  |
| datePost | Date input | Number |  |  |  |
| latitude | Coordinate latitude | Number |  | Not Null |  |
| longitude | Coordinate longitude | Number |  | Not Null |  |
| alamat | Location details | String |  | Not Null |  |

**Operasi pasar**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field name | Caption | Data Type | Key Type | Constraints | Reference Collections |
| operasiPasar\_ id |  | Autonumber | Primary Key |  |  |
| user\_id | Name komoditas | Number | Foreign key | Not Null | users |

Lampiran 5 Kamus data (lanjutan)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Field name | Caption | Data Type | Key Type | Constraints | Reference Collections |
| komoditas\_id | Satuan komoditas | Number | Foreign key | Not Null | komoditas |
| latitude | Coordinate latitude | Number |  | Not Null |  |
| longitude | Coordinate longitude | Number |  | Not Null |  |
| alamat | Location details | String |  | Not Null |  |
| pesan | Message user | String |  | Not Null |  |
| datePost | Date input | Number |  |  |  |
| pendukung | Vote of users | Array |  | Not Null |  |
| tanggapan | Response | Array |  | Not Null |  |

Lampiran 6 *Acceptance test criteria* iterasi pertama

1. Komoditas

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Kd.1 | Menambah komoditas | * Memilih *method* POST * Klik tombol *send* | * token * nama komoditas * satuan komoditas * harga komoditas | * status * *message* * data komoditas * token |
| Kd.2 | Melihat semua komoditas | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data komoditas * token |
| Kd.3 | Melihat detil satu komoditas | * Memilih *method* GET * Klik tombol *send* | * token * komoditas\_id | * status * *message* * data komoditas * token |
| Kd.4 | Mengubah informasi komoditas | * Memilih *method* POST * Klik tombol *send* | * token * komoditas\_id * nama komoditas * satuan komoditas * harga komoditas | * status * *message* * data komoditas * token |
| Kd.5 | Menghapus komoditas | * Memilih *method* POST * Klik tombol *send* | * token * komoditas\_id | * status * *message* * data komoditas * token |

Lampiran 6 *Acceptance test criteria* iterasi pertama (lanjutan)

1. Laporan harga

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Lp.1 | Menambah laporan harga | * Memilih *method* POST * Klik tombol *send* | * token * komoditas\_id * user\_id * harga * latitude * longitude * alamat | * status * *message* * data laporan harga * token |
| Lp.2 | Melihat semua laporan harga | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data laporan harga * token |
| Lp.3 | Melihat detil satu laporan harga | * Memilih *method* GET * Klik tombol *send* | * token * laporanharga\_id | * status * *message* * data laporan harga * token |
| Lp.4 | Mengubah informasi laporan harga | * Memilih *method* POST * Klik tombol *send* | * token * laporanharga\_id * harga komoditas | * status * *message* * data laporan harga * token |
| Lp.5 | Menghapus laporan harga | * Memilih *method* POST * Klik tombol *send* | * token * laporanharga\_id | * status * *message* * data laporan harga * token |

Lampiran 6 *Acceptance test criteria* iterasi pertama (lanjutan)

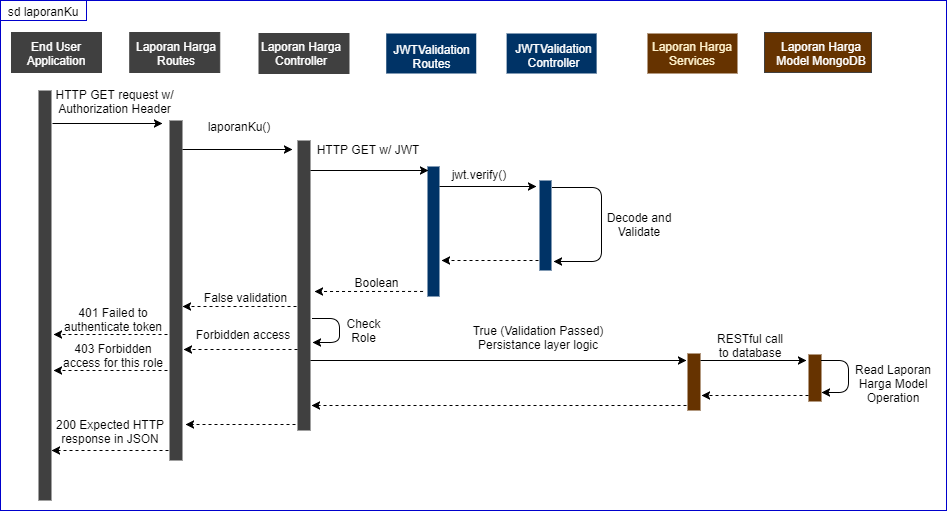
1. Operasi pasar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Op.1 | Menambah operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * user\_id * komoditas\_id * latitude * longitude * alamat * pesan | * status * *message* * data operasi pasar * token |
| Op.2 | Melihat semua operasi pasar | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data operasi pasar * token |
| Op.3 | Melihat detil satu operasi pasar | * Memilih *method* GET * Klik tombol *send* | * token * operasiPasar\_id | * status * *message* * data operasi pasar * token |
| Op.4 | Mengubah informasi operasi pasar | * Memilih *method* POST * Klik tombol *send* | * Token * operasiPasar\_id * pesan | * status * *message* * data operasi pasar * token |
| Op.5 | Menghapus operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * operasiPasar\_id | * status * *message* * data operasi pasar * token |

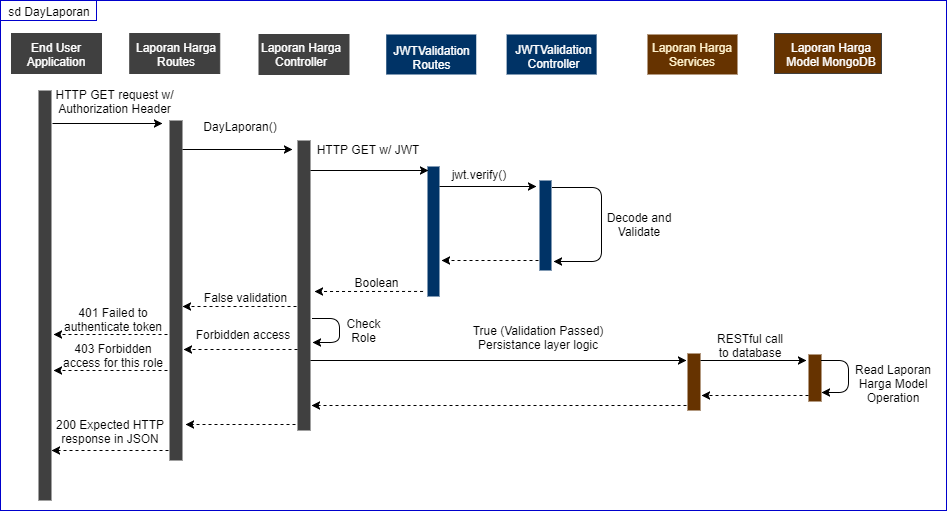
Lampiran 7 *Sequence diagram* iterasi kedua

**Laporan Harga**

1. *History* pengguna



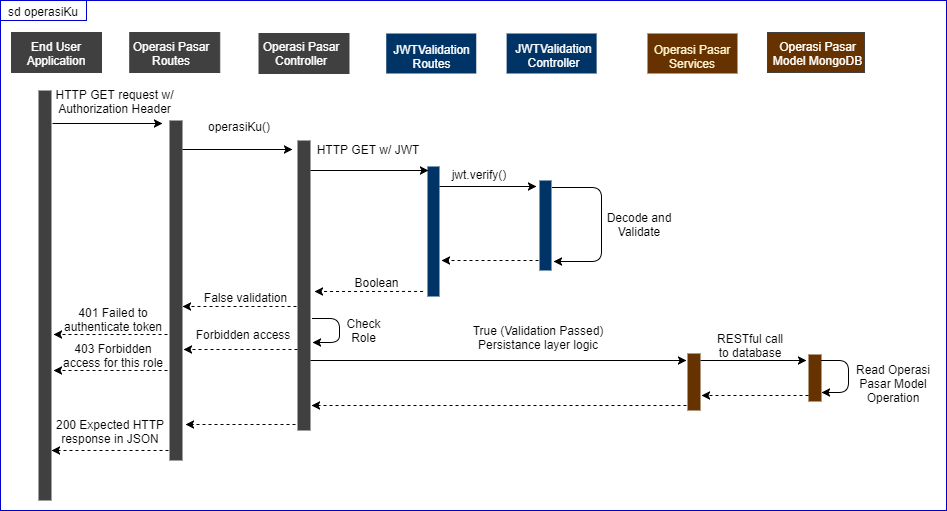
1. *Day* laporan



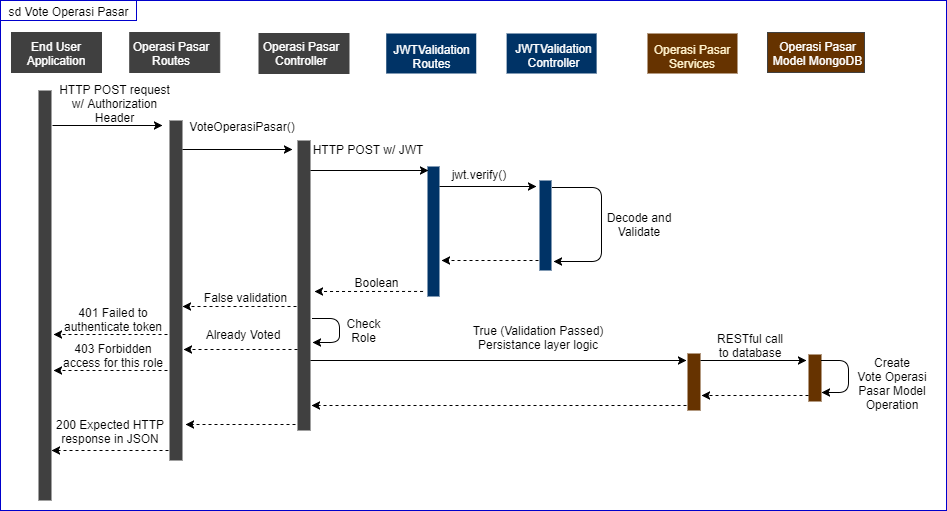
Lampiran 7 *Sequence diagram* iterasi kedua (lanjutan)

**Operasi pasar**

1. *History* pengguna

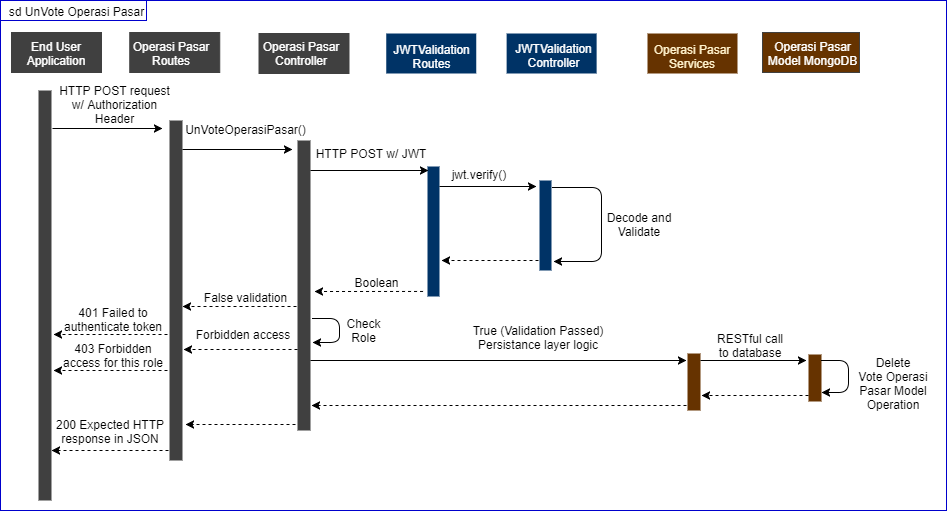


1. *Vote* operasi pasar

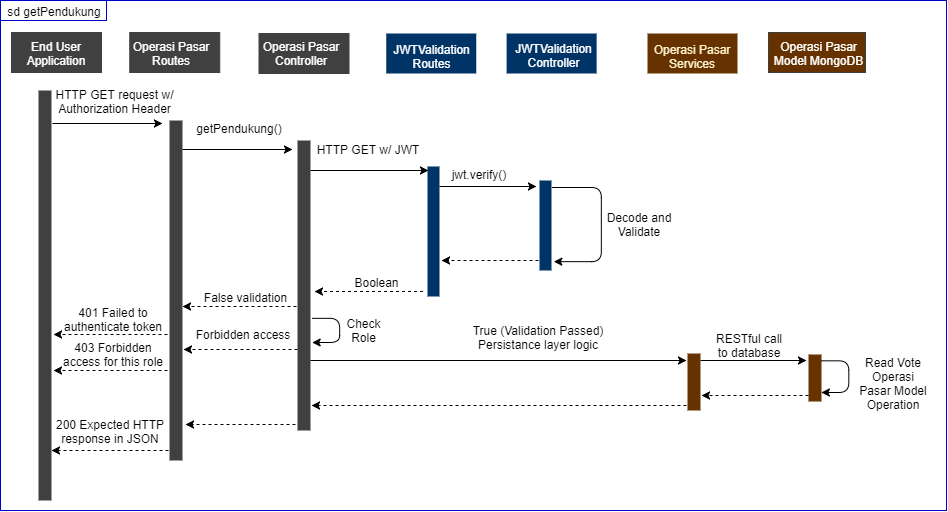


Lampiran 7 *Sequence diagram* iterasi kedua (lanjutan)

1. *UnVote* operasi pasar



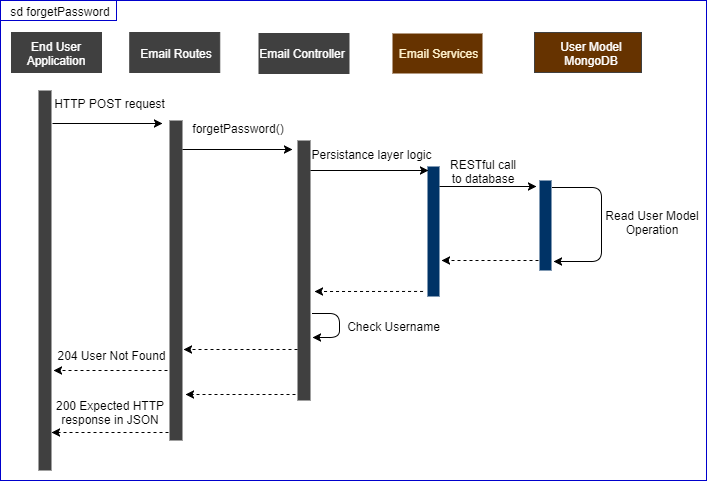
1. *Get* pendukung operasi pasar



Lampiran 7 *Sequence diagram* iterasi kedua (lanjutan)

**Email**

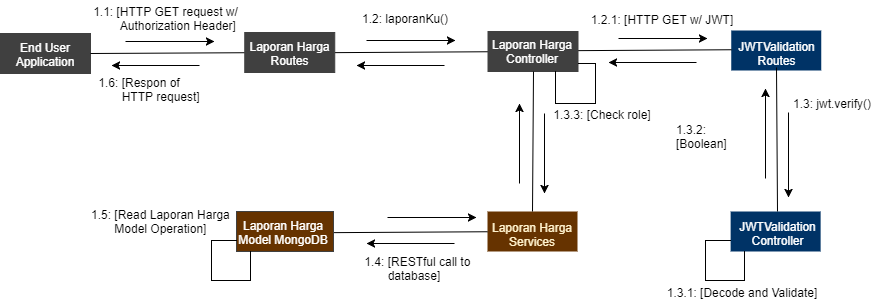
1. *Forget password*



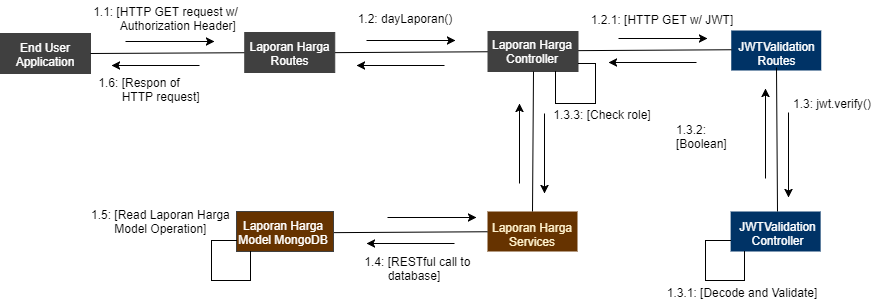
Lampiran 8 *Collaboration diagram* iterasi kedua

**Laporan Harga**

1. Historypengguna

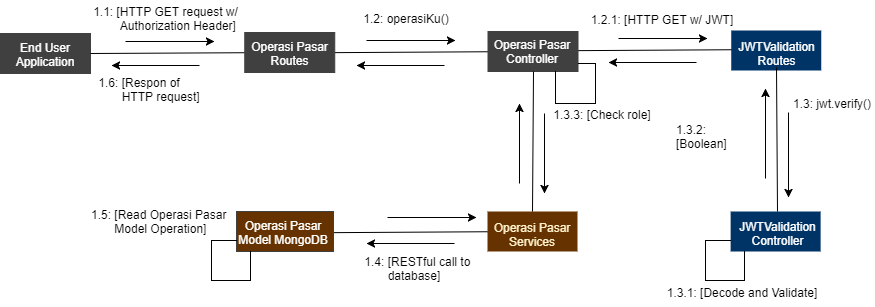


1. *Day* laporan



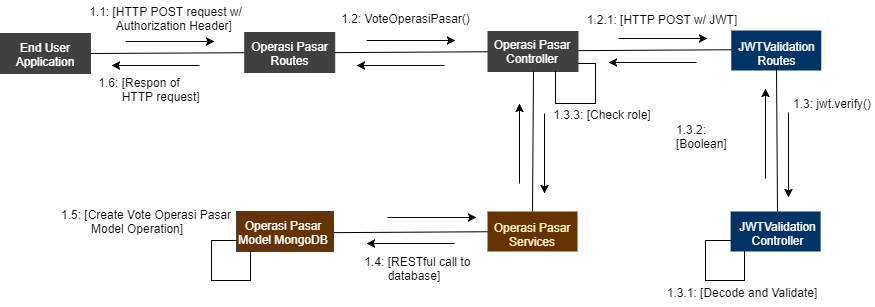
**Operasi pasar**

1. *History* pengguna

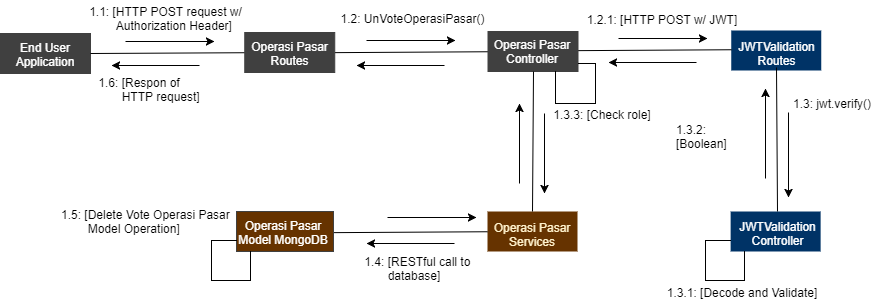


Lampiran 8 *Collaboration diagram* iterasi kedua (lanjutan)

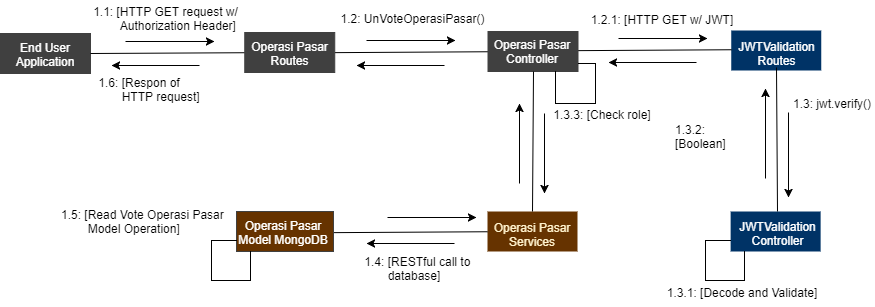
1. *Vote* operasi pasar



1. *UnVote* operasi pasar



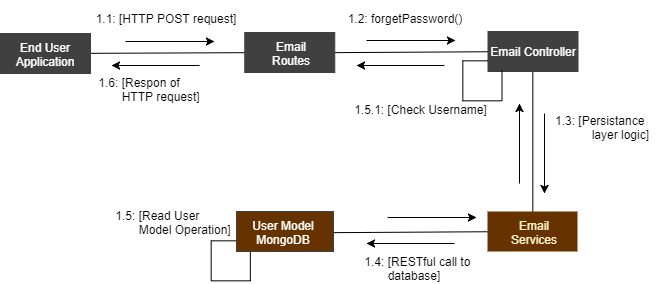
1. *Get* pendukung operasi pasar



Lampiran 8 *Collaboration diagram* iterasi kedua (lanjutan)

**Email**

1. *Forget* *password*



Lampiran 9 *Acceptance test criteria* iterasi kedua

1. Laporan harga

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Lp.6 | Melihat *history* laporan harga seorang *user* | * Memilih *method* GET * Klik tombol *send* | * token * user\_id | * status * *message* * data laporan harga * token |
| Lp.7 | Melihat laporan harga pada suatu hari tertentu | * Memilih *method* GET * Klik tombol *send* | * token * angka berapa hari dari sekarang | * status * *message* * data laporan harga * token |

1. Operasi pasar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Op.6 | Melihat *history* operasi pasar seorang *user* | * Memilih *method* GET * Klik tombol *send* | * token * user\_id | * status * *message* * data operasi pasar * token |
| Op.7 | Mendukung operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * user\_id * operasiPasar\_id | * status * *message* * data operasi pasar * token |
| Op.8 | Membatalkan dukungan operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * user\_id * operasiPasar\_id | * status * *message* * data operasi pasar * token |

Lampiran 9 *Acceptance test criteria* iterasi kedua (lanjutan)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Op.9 | Melihat pendukung operasi pasar | * Memilih *method* GET * Klik tombol *send* | * token * operasiPasar\_id | * status * *message* * data operasi pasar * token |

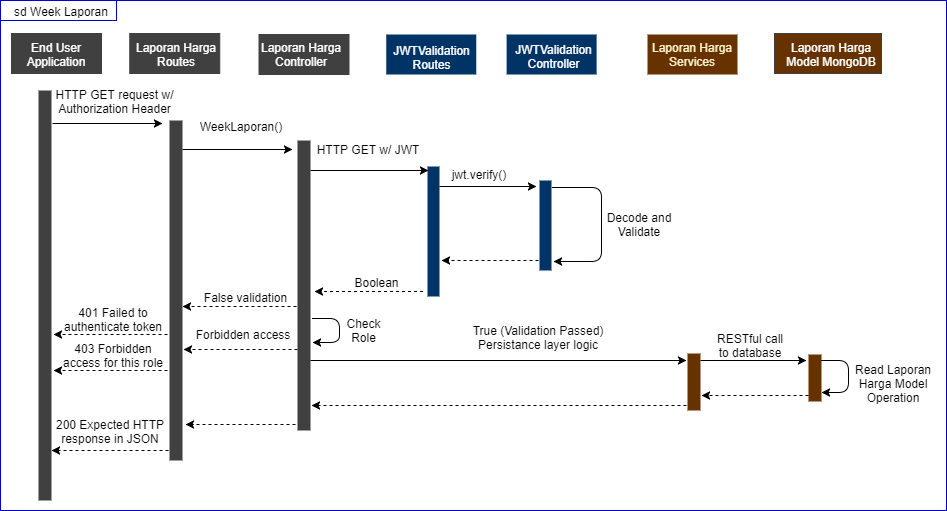
1. Email

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  masukan | Keluaran yang  diharapkan  (JSON) |
| Em.1 | Mengirim password baru | * Memilih *method* POST * Klik tombol *send* | * username * email | * status * *message* * data pengguna |

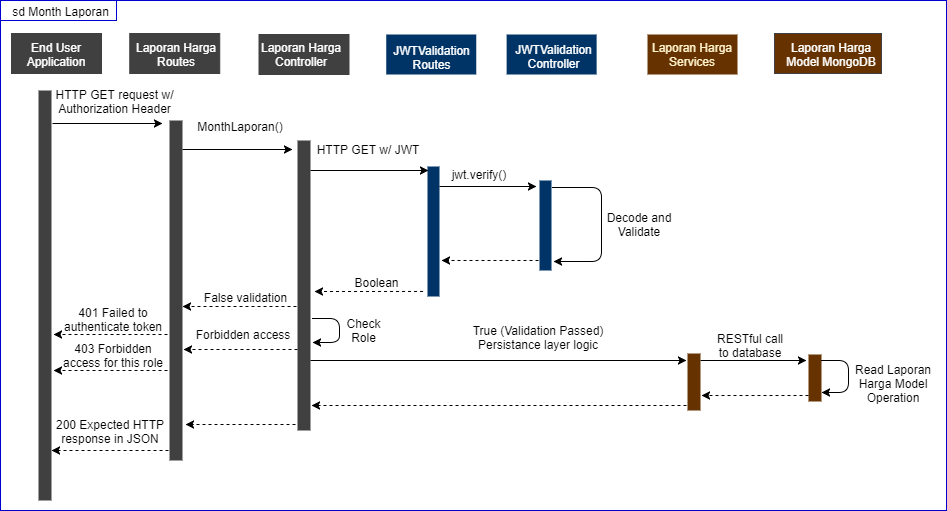
Lampiran 10 *Sequence diagram* iterasi ketiga

**Laporan harga**

1. *Week* laporan harga

**

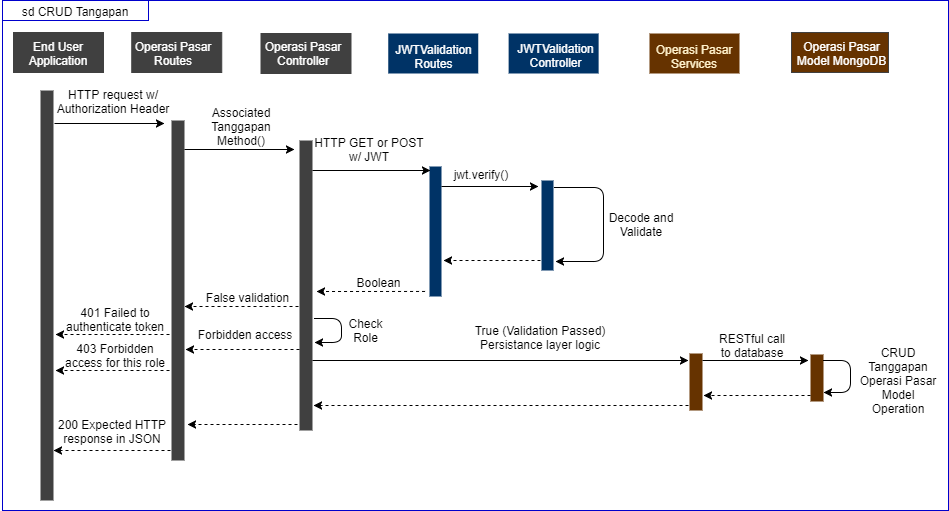
1. *Month* laporan harga

**

Lampiran 10 *Sequence diagram* iterasi ketiga (lanjutan)

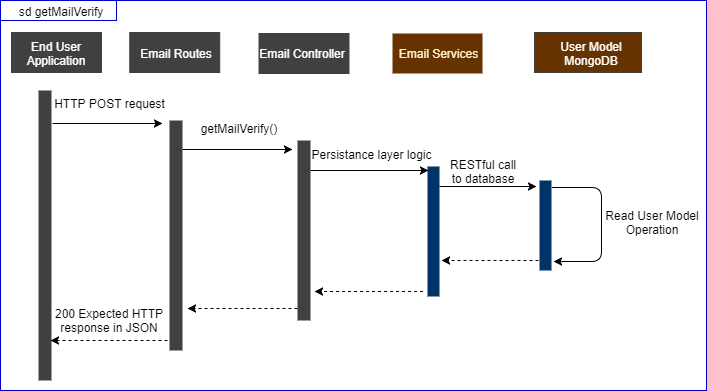
**Operasi pasar**

1. CRUD tanggapan operasi pasar



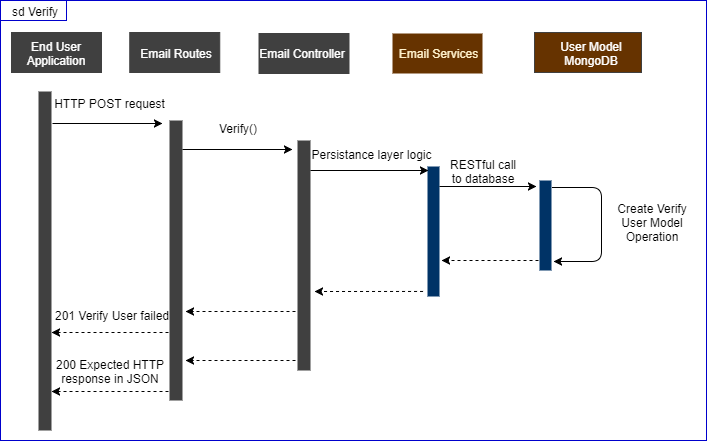
**Email**

1. getMailVerify

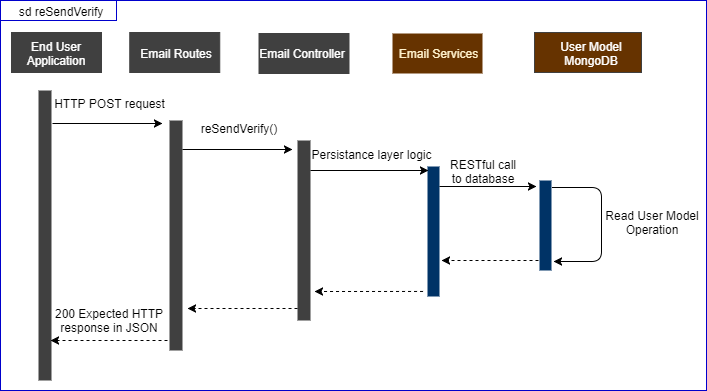
**

Lampiran 10 *Sequence diagram* iterasi ketiga (lanjutan)

1. Verify

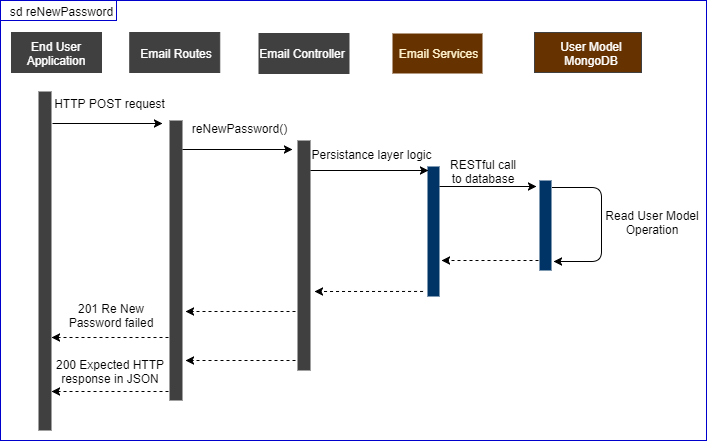


1. reSendMailVerify



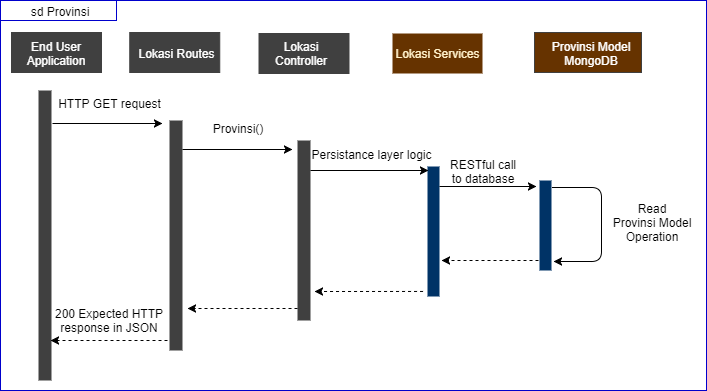
Lampiran 10 *Sequence diagram* iterasi ketiga (lanjutan)

1. reNewPassword



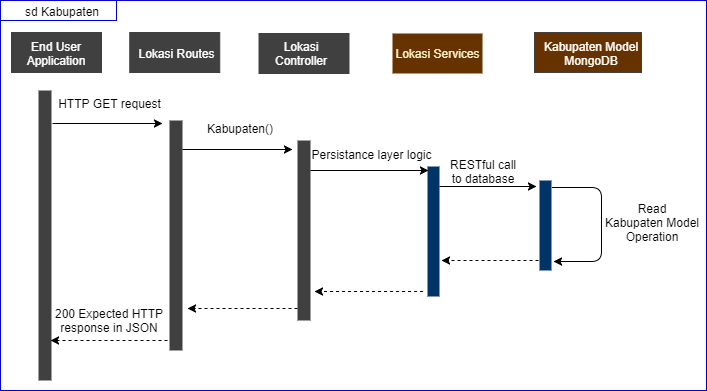
**Lokasi**

1. Provinsi

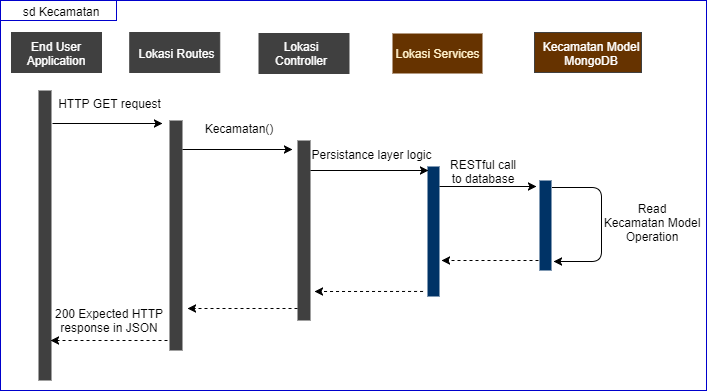


Lampiran 10 *Sequence diagram* iterasi ketiga (lanjutan)

1. Kabupaten

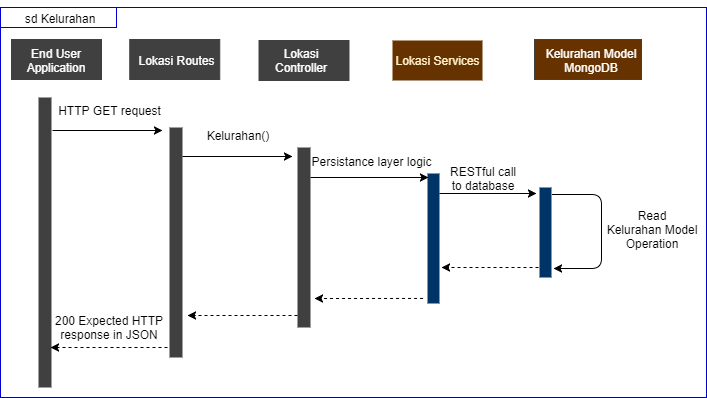


1. Kecamatan



Lampiran 10 *Sequence diagram* iterasi ketiga (lanjutan)

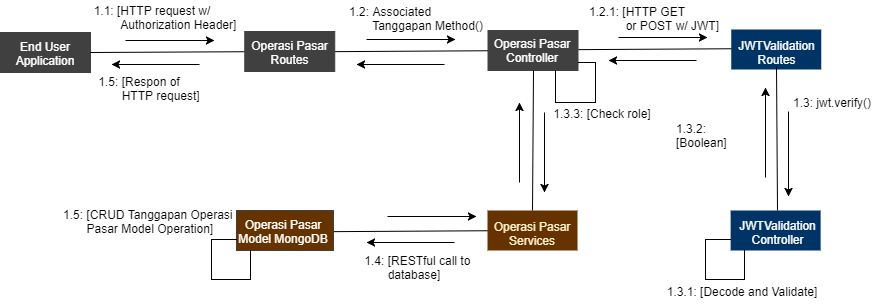
1. Kelurahan



Lampiran 11 *Collaboration diagram* iterasi ketiga

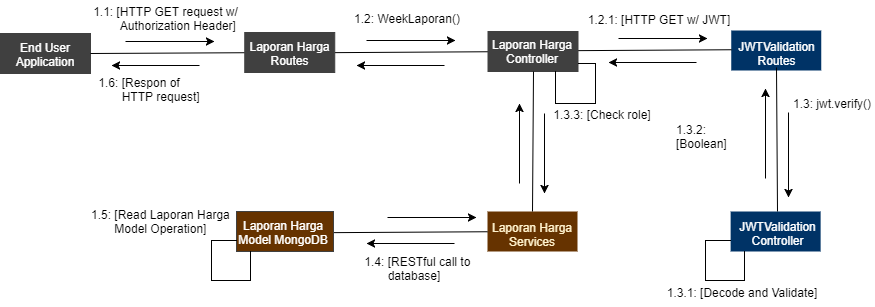
**Operasi pasar**

1. CRUD tanggapan

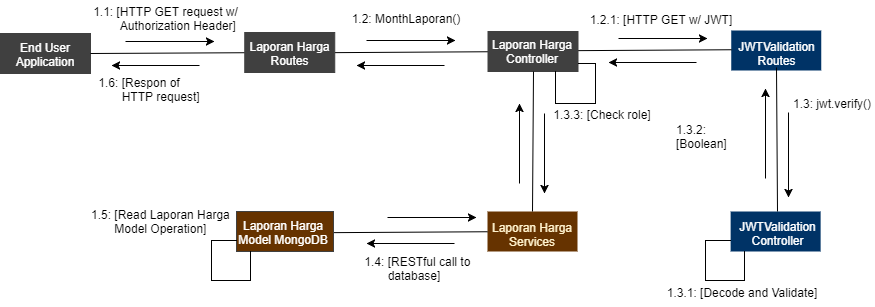


**Laporan harga**

1. *Week* laporan



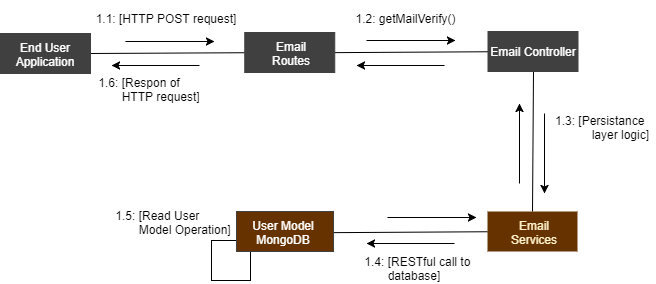
1. *Month* laporan



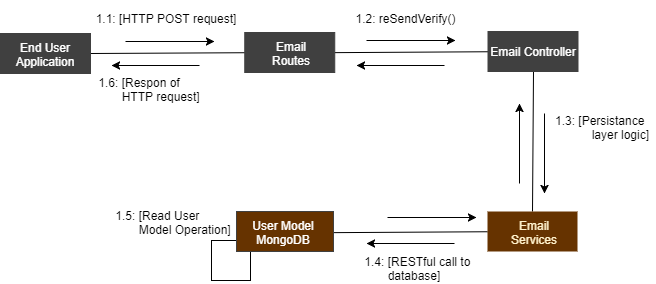
Lampiran 11 *Collaboration diagram* iterasi ketiga (lanjutan)

**Email**

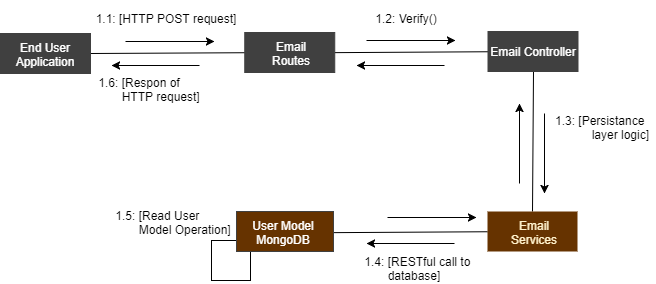
1. getMailVerify



1. reSendVerify

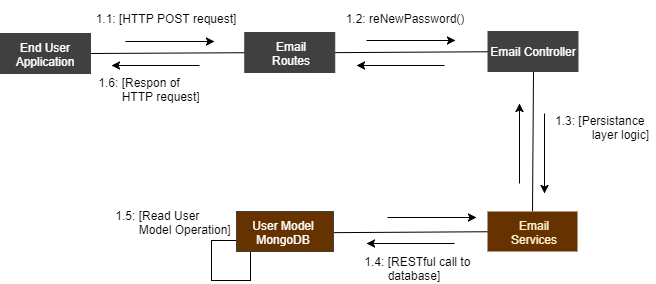


1. Verify



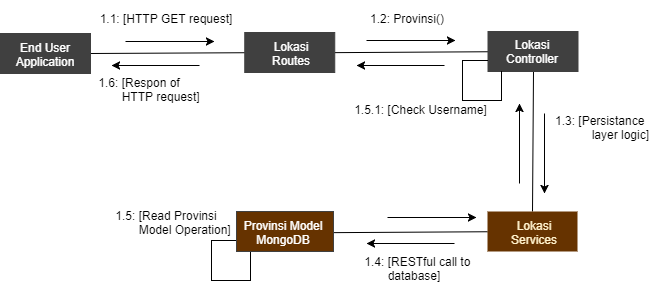
Lampiran 11 *Collaboration diagram* iterasi ketiga (lanjutan)

1. reNewPassword

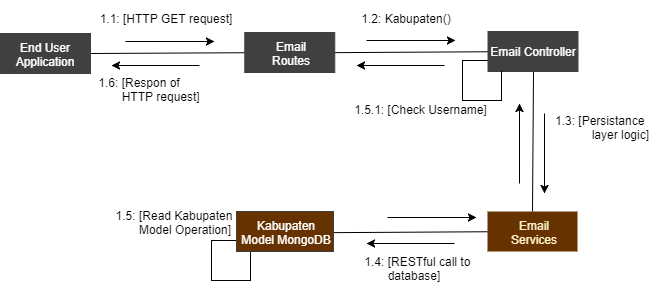


**Lokasi**

1. Provinsi

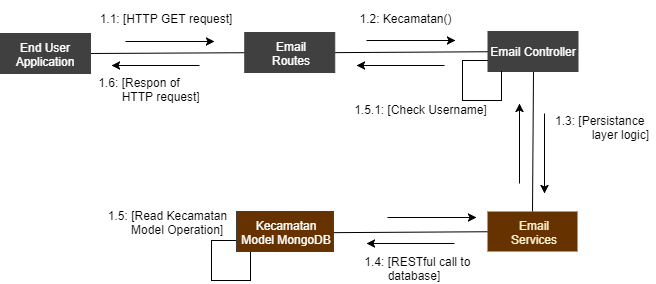


1. Kabupaten

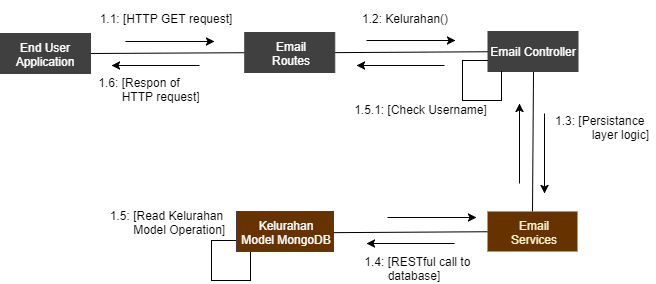


Lampiran 11 *Collaboration diagram* iterasi ketiga (lanjutan)

1. Kecamatan



1. Kelurahan



Lampiran 12 *Acceptance test criteria* iterasi ketiga

1. Laporan harga

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Lp.8 | Melihat laporan harga minggu ini | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data laporan harga * token |
| Lp.9 | Melihat laporan harga bulan ini | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data laporan harga * token |

1. Email

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Em.1 | Mengirim email lupa *password* | * Memilih *method* POST * Klik tombol *send* | * Username | * status * *message* * data pengguna |
| Em.2 | Mengirim email verifikasi *account* | * Memilih *method* POST * Klik tombol *send* | * Username | * status * *message* * data pengguna |
| Em.3 | Mengirim ulang email verifikasi *account* | * Memilih *method* POST * Klik tombol *send* | * Username | * status * *message* * data pengguna |

Lampiran 12 *Acceptance test criteria* iterasi ketiga (lanjutan)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Em.4 | Memverifikasi *account* | * Memilih *method* GET * Klik tombol *send* | * token | * status * *message* * data pengguna |
| Em.5 | Memperbarui *password* | * Memilih *method* POST * Klik tombol *send* | * token * *password* baru | * status * *message* * data pengguna |

1. Operasi pasar

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Op.10 | Memberikan tanggapan operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * operasiPasar\_id * isi | * status * *message* * data operasi pasar * token |
| Op.11 | Menghapus tanggapan operasi pasar | * Memilih *method* POST * Klik tombol *send* | * token * operasiPasar\_id * \_id | * status * *message* * data operasi pasar * token |
| Op.12 | Melihat tanggapan operasi pasar | * Memilih *method* GET * Klik tombol *send* | * token * operasiPasar\_id | * status * *message* * data operasi pasar * token |

Lampiran 12 *Acceptance test criteria* iterasi ketiga (lanjutan)

Lokasi

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| ID Pengujian | Deskripsi Pengujian | Prosedur  Pengujian | Data  Masukan | Keluaran yang  diharapkan  (JSON) |
| Lk.1 | Melihat seluruh provinsi di Indonesia | * Memilih *method* GET * Klik tombol *send* | * provinsi | * status * *message* * data provinsi |
| Lk.2 | Melihat kabupaten yang berada pada suatu provinsi | * Memilih *method* GET * Klik tombol *send* | * kabupaten * id\_prov | * status * *message* * data kabupaten |
| Lk.3 | Melihat kecamatan yang berada pada suatu kabupaten | * Memilih *method* GET * Klik tombol *send* | * kecamatan * id\_kab | * status * *message* * data kecamatan |
| Lk.4 | Melihat kelurahan yang ada pada suatu kecamatan | * Memilih *method* GET * Klik tombol *send* | * kelurahan * id\_kec | * status * *message* * data kelurahan |

**RIWAYAT HIDUP**

Penulis dilahirkan di Sragen pada tanggal 17 Agustus 1996 dari pasangan dari Sunardi dan Rusmiyanti. Penulis adalah putra pertama dari dua bersaudara. Pada tahun 2013, penulis lulus dari SMA Negeri 1 Sragen dan diterima di Departemen Ilmu Komputer, Fakultas Matematika dan Ilmu Pengetahuan Alam, Institut Pertanian Bogor melalui jalur SNMPTN Undangan di tahun yang sama.

Pada tahun 2016, penulis melaksanakan Praktik Kerja Lapangan (PKL) di Komunitas Mandiri Indonesia (KMI) Bogor selama 35 hari kerja. Pada masa perkuliahan, penulis aktif menjadi asisten praktikum mata kuliah Rekayasa Perangkat Lunak.