BAB II

KAJIAN PUSTAKA

2.1. Kajian Penelitian Sebelumnya

Terdapat beberapa penelitian sebelumnya yang meneliti tentang kasus serupa mengenai implementasi *agile framework* untuk pembuatan Aplikasi Monitoring Jaringan mengunakan *Mikrotik API* dan *Unifi API*. Adapun kajian penelitian akan diuraikan sebagai berikut.

2.1.1 Sistem Monitoring User Hotspot dan Kondisi Jaringan Komputer (Aditya, 2016)

Pada penelitian ini membahas tentang pengembangan sistem *monitoring* jaringan dan *user hotspot* menggunakan Mikrotik API yang ditujukan untuk memberikan informasi atau membantu admin jaringan untuk memudahkan memonitoring sebuah jaringan yang besar. Pengembangan sistem pada penelitian ini berbasis Web.

2.1.2 Rancang Bangun Sistem *Monitoring* Jaringan Menggunakan Mikrotik OS (Lumbantoruan, 2016)

Pada penelitian ini membahas penggunaan MikroTik API yang disediakan oleh MikroTik yang digunakan untuk membuat sebuah sistem yang memonitoring *traffic* jaringan, pembagian *bandwidth* user, log mikrotik, dan notifikasi email. Akan tetapi pada sistem yang

dikembangkan oleh penulis hanya dapat memonitoring traffic jaringan di satu jalur saja, dan hanya dapat memonitoring satu perangkat saja.

2.1.3 Manajemen Tata Kelola Captive Portal Hotspot Mikrotik dan Unifi Controller (Yutanto, 2015)

Pada penelitian ini, *Hotspot MikroTik* dan *Unifi Controller* diimplementasikan untuk memudahkan admin jaringan untuk memonitoring aktivitas *user hotspot* yang sedang aktif melalui mikrotik maupun *unifi controller* serta konfigurasi dan monitoring perangkat *wireless* baik yang terinstall maupun yang baru dapat dilakukan melalui *controller* wifi. Pengembangan sistem pada penelitian ini berbasis web.

2.1.4 Comparison of agile methods : Scrum, Kanban, and Scrumban (Brezočnik & Majer, 2016)

Pada penelitian ini membahas perbandingan *framework* - *framework agile development* yang paling optimal untuk perusahaan penulis. Hasil dari penelitian ini menyatakan bahwa setiap metode tersebut memiliki kelebihan dan kekurangannya sendiri. Scrum merupakan metode yang tepat untuk perusahaan dewasa, perusahaan yang memiliki tim yang berpengalaman lebih dari satu tahun mengembangkan produk atau proyek. Kanban cocok untuk perusahaan dengan produksi berkelanjutan yang membutuhkan respon cepat terhadap perubahan. Sedangkan scrumban cocok untuk perusahaan rintisan atau kecil, karena mengandung fleksibilitas kanban dan karakteristik dasar scrum.

2.1.5 Perbedaan penelitian ini terhadap kajian penelitian sebelumnya

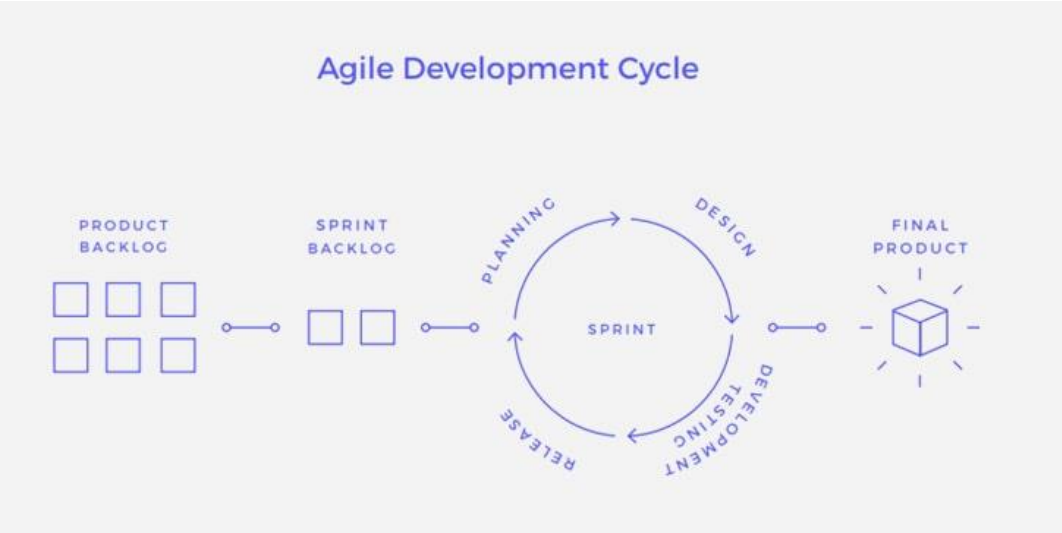
Perbedaan penelitian ini dengan kajian penelitian sebelumnya adalah pada penelitian ini merupakan gabungan dari kajian sebelumnya yaitu pengembangkan apilkasi yang dapat memonitoring jaringan, *user hotspot* serta tambahan fitur *monitoring* perangkat - perangkat jaringan berupa router MikroTik serta access point Unifi ke dalam satu aplikasi monitoring jaringan dengan memanfaatkan API yang telah disediakan oleh MikroTik dan Ubiquiti Unifi. Dalam pengembangan Aplikasi dalam penelitian ini menggunkan metode *agile development* dengan *framework* kanban.

2.2 Dasar Teori

2.2.1 Agile Development

Agile Development Methods adalah sekelompok metode pengembangan perangkat lunak yang didasarkan pada prinsip-prinsip yang sama atau pengembangan sistem yang memerlukan adaptasi cepat terhadap perubahan dalam bentuk apapun. *Agile Development Methode* bersifat cepat, ringan dan bebas bergerak dan waspada sehingga dalam pengembangan perangkat lunak dengan menggunakan metode *agile* diperlukan inovasi dan *responsibility* yang baik antar *developer team* dan klien agar perangkat lunak yang dihasilkan bagus. *Agile Development* memiliki beberapa model kerangka kerja antara lain seperti : *Acceptence Test Driven Development* (ATDD), *Adaptive Software Development* (ASD), *Agile Unified Proccess* (AUP),

Continuous Integration (CI), *Crystal Clear*, *Crystal Methods, Dynamic System Development Method* (DSDM), *Extreme Programming* (XP), *Feature Driven Development* (FDD), *Graphical System Design* (GSD), *Kanban, Lean Software Development*, *Scrum*, *Scrumban*, *Test Driven Development* (TDD), *Velocity Tracking*, dan *Software Development Rhythms*. (Alexandra, 2017)



Gambar 2.1 *Agile Development Cycle*

2.2.2 Kanban

Kanban diperkenalkan di dunia industri manufaktur di Jepang pada tahun 1950-an. Kanban dalam Bahasa Jepang berarti papan nama, dan digunakan dalam system penjadwalan. Kanban pertama kali dikembangkan oleh Toyota. Metode Kanban dalam pengembangan sebuah perangkat lunak berawal tahun 2004, ketika David J. Anderson membantu tim IT kecil Microsoft yang sedang terpuruk. Metode Kanban dalam pengembangan perangkat lunak mendorong tim proyek untuk mengambarkan atau memvisualisasikan alur kerja, membatasi pekerjaan yang sedang berjalan atau *Work In Progress* (WIP) pada setiap alur kerja, dan mengatur siklus waktu (Ahmad et al.,

2013). *Kanban Board* memberikan gambaran tentang alur pengembangan sebuah perangkat lunak, karena menunjukan perkerjaan yang ditugaskan kepada masing–masing *developer* serta menjelaskan prioritas tugas mana yang harus dikerjakan terlebih dahulu dan metode ini juga dapat melacak jika dalam pengembangan terdapat *bottleneck*. Selain itu tujuan penggunaan Kanban yaitu untuk menimalkan *Work In Progress*, yaitu hanya mengerjakan perkerjaan yang diminta dan agar *developer* hanya fokus pada beberapa pekerjaan dalam waktu tertentu.

2.2.2.1 User Stories

Pada metode *agile*, rincian pernyataan dinegosiasikan melalui diskusi yang terjadi secara berulang selama pengembangan aplikasi. *User stories* merupakan hasil dari diskusi yang dilakukan oleh *Product Owner* dengan *Stakeholder* yang akan menggunakan aplikasi yang dikembangkan yang nantinya digunakan untuk membuat *Product Backlog*. Dalam *user stories* berisi nama *user* pengguna aplikasi, fitur yang menjadi kebutuhan dan tujuan dari fitur yang direncanakan. Dengan adanya *user stories* memudahkan komunikasi antara tim teknis dan tim bisnis karena bahasa yang digunakan sangat umum (Kurniawan & Sani, 2019).

2.2.2.2 Product Backlog

Product Backlog merupakan daftar pekerjaan-pekerjaan yang dituliskan dan dijelaskan oleh *Product Owner* atau terjemahan dari *user stories* ke dalam bahasa yang sedikit teknis. Pada *product backlog*, *Product Owner* menentukan prioriatas dalam pengerjaan setiap *backlog*.

Kebutuhan *customer* atau kebutuhan teknikal sebagai acuan untuk menentukan prioritas pengerjaan. Selain itu penentuan prioritas juga dapat ditentukan berdasarkan nilai krusial atau nilai penting fitur yang akan dibuat. Dalam penentuan prioritas *Product Owner* dapat meminta tim *developer* atau *stakeholders* (Kurniawan & Sani, 2019).

2.2.2.3 Kanban Board & Work In Progress

Kanban Board merupakan sebuah papan dimana papan tersebut berfungsi untuk menvisualisasikan atau mengambarkan pekerjaan pada berbagai tahap proses dengan menggunakan *post-it* atau kartu untuk mewakili item pekerjaan atau *backlog*, dan kolom untuk mewakili setiap tahap proses. Setiap kolom tahap proses memiliki batasan maksimal jumlah item pekerjaan atau *backlog* yang dapat ditampung (*limit Work In Progress*). Manfaat *kanban board* antara lain untuk memudahkan manajement tugas atau *task*, *stay on top priority*, mengetahui apa yang sedang atau telah dilakukan oleh kolaborator/ rekan kerja ketika bekerja dengan orang lain dalam suatu *project*.

***Gambar 2.2** Kanban Board*

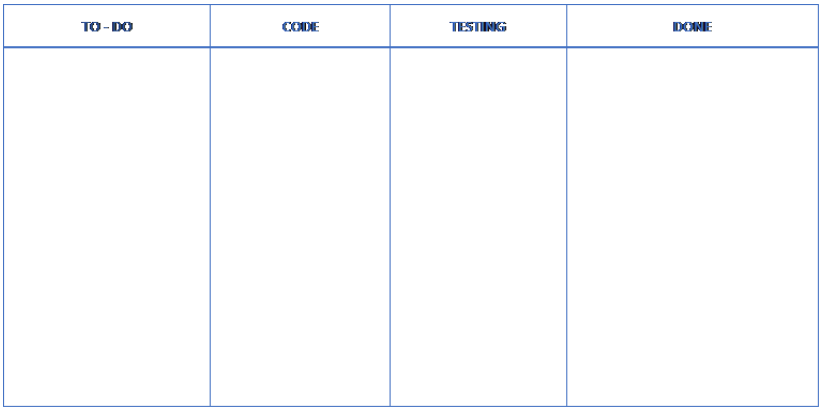
2.2.2.4 Pengunaan Kanban

Berikut proses singkat implementasi *Kanban Software Development* **(Gustino, 2018)**:

a. Memilih *Agile Coach*, *agile coach* bertanggung jawab memimpin seluruh anggota agar bekerja dengan sesuai dengan aturan dan etika yang baik.

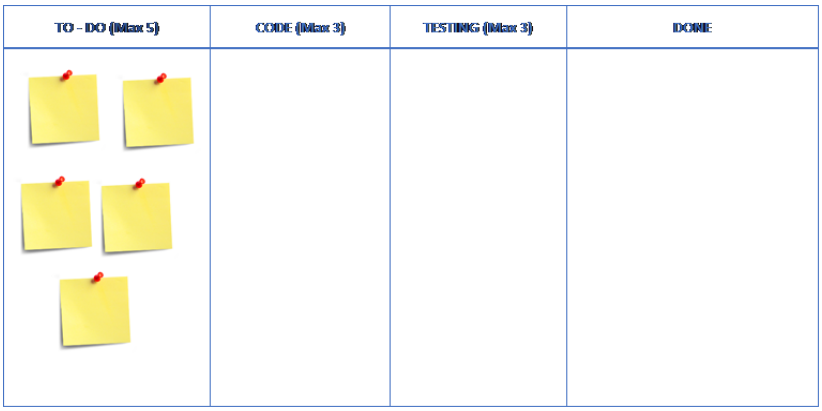
b. Menyiapkan *Kanban Board*, dan beberapa kertas yang nantinya ditempelkan di *Kanban Board*. Atau juga dapat menggunakan aplikasi pihak ketiga seperti trello sebagai alternatif.

c. Dev Team dan Agile Coach membagi beberapa tahapan di Kanban Board atau Aplikasi Kanban Board, biasanya pembagian berbentuk seperti gambar berikut :



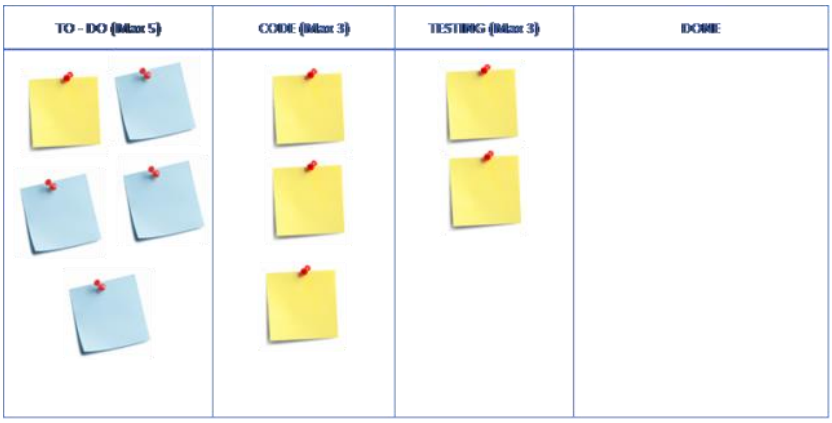
Gambar 2.3 *Pembagian tahapan-tahapan di Kanban Board*

d. Dev Team dan Agile Coach menempelkan tugas – tugas apa saja yang harus diselesaikan dan menentukan jumlah maksimal tugas di setiap tahapan seperti gambar berikut :



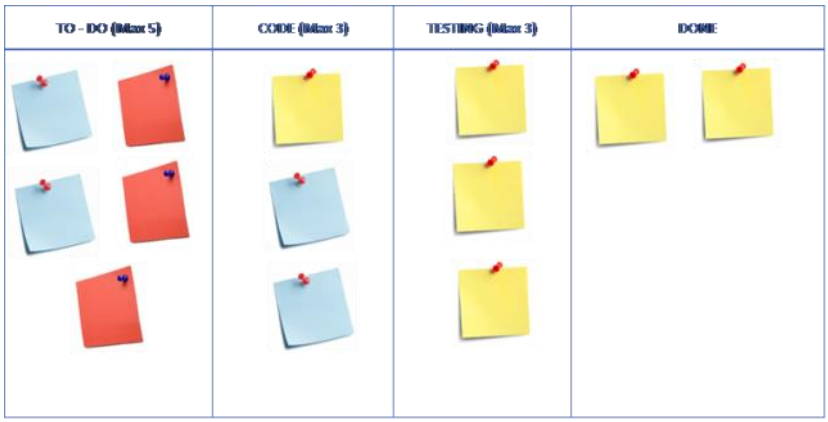
Gambar 2.4 *Penulisan tugas dan menentukan limit di setiap tahapan*

e. Dev Team bekerja dan jika terdapat pekerjaan yang selesai, kertas atau stiker yang bertuliskan pekerjaan yang selesai dipindahkan ke tahap selanjutnya. Dan jika tugas di tahap ‘To-Do’ mulai sedikit, Dev Team menambahkan tugas dari *Product Backlog.* Dev Team akan melakukan Daily Stand-Up setiap akan memulai pekerjaan untuk membicarakan progress pekerjaan.



Gambar 2.5 *Pemindahan tugas ke tahap selanjutnya.*

f. Jika tugas – tugas yang terdapat di tahap ‘*Done*’ sudah cukup, maka Dev Team akan melakukan demo di depan stakeholder, dan merelease produk tersebut.

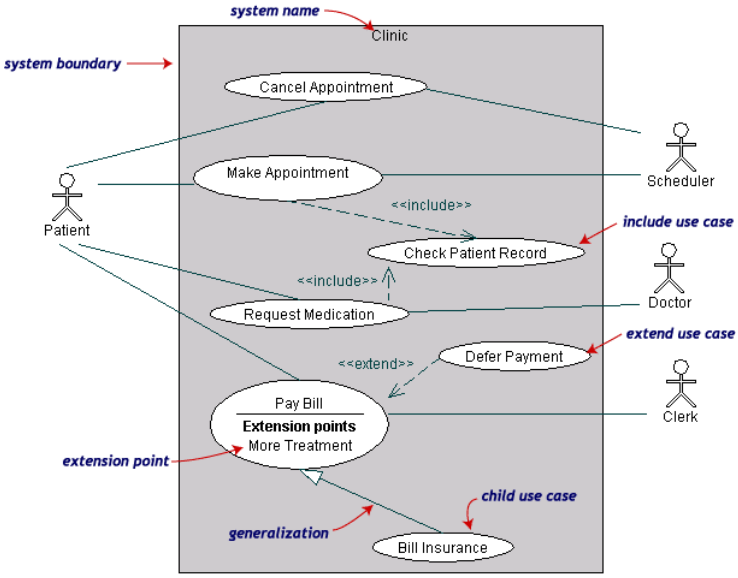


Gambar 2.6 *Tugas ditahap 'Done' akan dipresentasikan ke Stakeholder.*

g. Kegiatan tersebut akan diulang-ulang sampai produk tersebut selesai.

2.2.3 Use Case

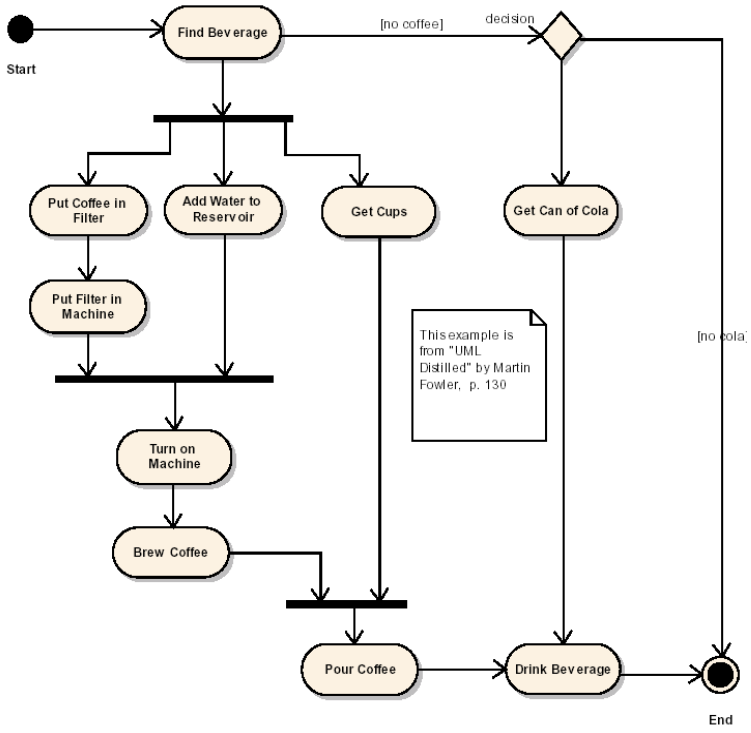
Use case diagram merupakan diagram yang menggambarkan fungsionalitas yang diharapkan dari sebuah system. Use case mempresentasikan sebuah interaksi antara aktor dengan sistem atau pekerjaan apa saja yang dapat dilakukan oleh aktor pada sistem tersebut. Sebagai contoh aktor dapat login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang / sebuah aktor adalah mempresentasikan sebuah entitas manusia atau mesin yang berinterasi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.(Dharwiyanti, 2003)



Gambar 2.7 *Contoh Use Case*

2.2.4 Activity Diagram

Activity diagram merupakan diagram yang menggambarkan alir aktivitas atau alir proses dalam sistem yang sedang dirancang. *Activity* diagram mempresentasikan sebuah alir berawal, *decision* atau keputusan yang mungkin terjadi, dan bagaimana sebuah alir berakhir (Dharwiyanti, 2003).



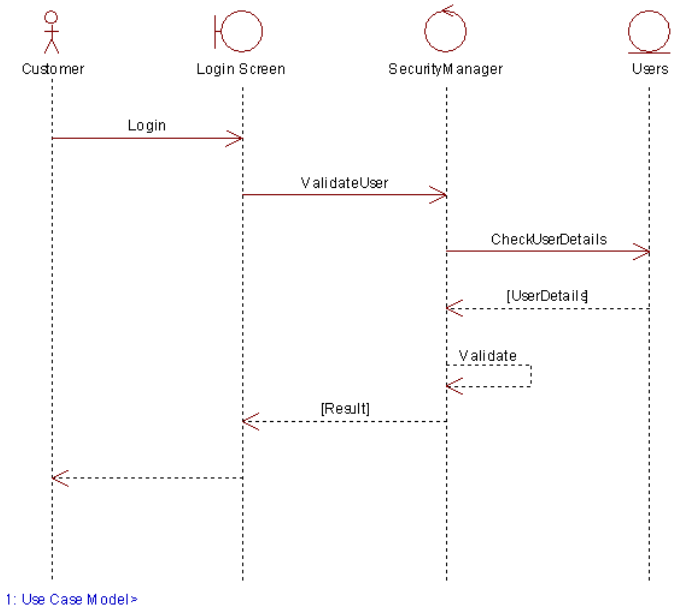
Gambar 2.8 *Contoh Activity Diagram*

2.2.5 Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence* diagram terdiri atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). *Sequence* diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan *output* apa yang dihasilkan (Dharwiyanti, 2003).

Masing-masing objek, termasuk aktor, memiliki *lifeline* vertikal. *Message* digambarkan sebagai garis berpanah dari satu objek ke objek

lainnya. Pada fase desain berikutnya, *message* akan dipetakan menjadi operasi/metoda dari *class*. *Activation* bar menunjukkan lamanya eksekusi sebuah proses, biasanya diawali dengan diterimanya sebuah *message*. Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan *icon* khusus untuk objek *boundary*, *controller* dan *persistent entity*.



Gambar 2.9 *Contoh Sequence Diagram*

2.2.6 Application Programming Interface (API)

Antarmuka pemograman aplikasi API (*Application Programming Interface*) merupakan sebuah *interface* yang berisi sekumpulan kode, sekumpulan fungsi dan sekumpulan *protocol* yang memiliki akses ke aplikasi lain atau *system operasi*. Sehingga dapat memanfaatkan atau menggunakan fitur yang terdapat di aplikasi lain tanpa harus membuat program atau fungsi yang serupa.

2.2.7 MikroTik

MikroTik merupakan sebuah perusahaan pembuat perangkat jaringan seperti *router* & *access point* yang berkantor pusat di Riga, Latvia. Mikrotik didirikan pada tahun 1996 yang pada awalnya berfokus di bidang *Wireless Internet Service Provider* atau penyedia internet nirkabel. Pada tahun 1997 MikroTik memulai mengembangkan RouterOS yaitu sebuah sistem operasi jaringan (*operating system network*) yang berfungsi untuk *firewall* atau *router network* berbasis linux yang dapat. Pada tahun 2002 MikroTik memulai mengembangkan RouterBOARD yaitu perangkat jaringan *router* yang terinstall dengan sistem operasi RouterOS yang dikembangkan oleh MikroTik.

2.2.7.1 Mikrotik RouterOS

MikroTik RouterOS merupakan sistem operasi yang diperuntukkan untuk *router* buatan MikroTik, atau juga dapat diinstall di Komputer/*Personal Computer*. MikroTik routerOS sendiri adalah sistem operasi dan perangkat lunak (software) berbasis linux yang dapat menjadikan komputer bisa atau *Personal Computer* sebagai perangkat jaringan *router*. RouterOS memiliki fitur-fitur *networking* diantaranya : *Firewall* & *NAT*, *Routing*, *Hotspot*, *DHCP Server*, *Tunneling Protocol*, *DNS Server*, *Bridging* dan masih banyak lagi.

2.2.7.2 RouterOS API (ROS Api)

ROS Api merupakan *application programming interface* yang berisi program atau kumpulan fungsi – fungsi yang berfungsi untuk

menghubungkan atau memberi perintah ke *router* mikrotik hanya dengan memanggil fungsi yang terdapat di program atau API tersebut. Hal-hal yang dapat diakses dengan menggunakan routerOS API antara lain, melihat *ipaddress*, melihat daftar user *hotspot*, melihat daftar *hotspot profile*, melihat penggunaan *resource*, menambah *user hotspot*, menghapus *user hotspot*, dan masih banyak lagi.

2.2.8 Unifi Controller

Unifi controller merupakan *software proprietary* dari produk ubiquiti, software tersebut digunakan untuk mengatur atau memanajemen produk – produk perangkat jaringan yang diproduksi oleh Ubiquiti seperti *Switch*, Access Point dan masih banyak lagi (Yutanto, 2015). Unifi Controller memiliki berbagai fitur antara lain mengkonfigurasi perangkat-perangkat jaringan, seperti adopting Access Point, mengatur SSID, frekuensi, *channel bandwidth*, VLAN, dan masih banyak lagi. Selain untuk mengkonfigurasi perangkat jaringan, Unifi Controller juga dapat maintenance perangkat jaringan seperti *reboot, upgrade* OS perangkat jaringan, dan momonitoring aktivitas user yang terhubung ke perangkat jaringan tersebut. Unifi Controller ini dapat diinstall di computer biasa atau PC dan juga diinstall di Server. Software ini tersedia dan dapat diinstall di berbagai sistem operasi, seperti Windows, Linux, dan Mac OS.

2.2.9 Unifi API

Unifi Api merupakan *application programming interface* yang berisi program atau kumpulan fungsi – fungsi yang berfungsi untuk menghubungkan ke unifi controller hanya dengan memanggil fungsi yang terdapat di program atau API tersebut. Hal-hal yang dapat diakses dengan menggunakan Unifi API antara lain, melihat daftar perangkat-perangkat unifi (*access point* dan *switch*), melihat total *client* yang terhubung via *wireless*, melihat jumlah *client* yang terhubung ke sebuah *access point* (AP), dan masih banyak lagi.