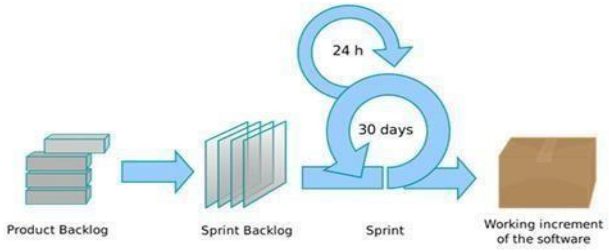
BAB II TINJAUAN PUSTAKA

2.1 Metode Pengembangan Perangkat Lunak

Metode pengembangan yang digunakan dalam pembuatan aplikasi ini yaitu dengan menggunakan metode *scrum.*



Gambar 2.1 Metode *Scrum*

Model *scrum* merupakan metode pengembangan perangkat lunak secara cepat (*agile*). Model *scrum* memiliki beberapa kelebihan seperti, mampu mentransformasikan proses bisnis yang sulit menjadi mudah dikembangkan, dengan model *scrum* mampu memonitoring dan mengontrol aktivitas pada proses pengembangan sistem Metode pengembangan sistem mengacu pada metode *Scrum* yang merupakan sebuah metode yang mudah dikontrol, fleksibel, memuat strategi pengembangan menyeluruh dimana seluruh tim bekerja sebagai satu unit untuk mencapai goal yang sama Proses pengembangan menggunakan metode scrum terdapat empat tahapan pengembangan yaitu : ((1) *product backlog*, (2) *sprint Backlog*, (3) *sprint log*, (4) *Increment*.

Adapun kerangka kerja dalam model scrum terdiri dari:

a.*Product Backlog*

Tahap *Product Backlog* lebih mengarah kepada pengumpulan kebutuhan, pembaruan, pemeliharaan, dan deskripsi singkat tentang fungsi-fungsi yang dinginkan pada saat aplikasi akan dibangun.

b.*Sprint Backlog*

Tahap *Sprint backlog* dilakukan untuk sebuah proses pemenuhan kebutuhan sesuai dengan yang dinginkan pada proses *Product Backlog* sebelumnya.

c. *Sprint Log*

Tahap *Sprint Log* merupakan proses dimana paparan aplikasi dalam bentuk sebuah *prototype* dan

pemaparan dalam bentuk hal teknis baik berupa *tools* yang dibutuhkan untuk mengembangkan aplikasi.([2]M. Efniasari, A. Wantoro, dan E. R. Susanto)

d. *Increment*

*Tahap Increment me*rupakan hasil akhir dari tahap *Product Backlog* yang telah selesai dikembangkan pada saat tahapan *Sprint Log*. Pada tahap ini, diharapkan tahapan Increment telah selesai dilakukan sehingga mampu untuk digunakan sesuai dengan yang dinginkan.

2.2 Metode Haversine

Metode *Haversine* merupakan sebuah metode yang digunakan dalam sistem navigasi dimana metode ini akan menghasilkan sebuah perhitungan jarak antara dua titik dari garis bujur (*longitude*) dan garis lintang (*latitude*). Rumus *Haversine* adalah persamaan penting dalam proses navigasi, *Haversine* Formula memberikan jarak lingkaran besar (radius) antara dua titik pada permukaan bumi berdasarkan bujur dan lintang. Penggunaan formal ini mengasumsikan pengabaian efek *ellipsoidal* (diasumsikan bumi tidak bulat sempurna melainkan lebih mendekati bentuk telur dengan permukaan yang tidak rata). Ini merupakan bentuk persamaan khusus dari trigonometri bola, *law of haversines*, mencari hubungan sisi dan sudut pada segitiga dalam bidang bola.

Rumus *Harvisine* Formula sebagai berikut :

x = (lon2-lon1) * cos ((lat1+lat2)/2); y= (lat2-lat1); d= sqrt(x*x+y*y)*R

Keterangan: x = Longitude (Lintang) y =

Latitude (Bujur) d = Jarak

R = Radius Bumi

2.3 Rest Api

Representational State Transfer (REST) adalah sebuah gaya arsitektur untuk pendistirbusian sistem *hypermedia*. Arsitektur *REST* adalah arsitektur klien – *server* dimana klien mengirim *request* pada *server* dan *server* memproses request dan mengembalikan sebuah response (transaksi). Setiap transaksi bersifat independen dan tidak terkait dengan transaksi lainnya (stateless). Hal ini yang membuat aplikasi REST sederhana dan ringan. *RESTful web service*

adalah sebutan untuk aplikasi web yang menggunakan arsitektur REST. *RESTful web service* menggunakan metode *http GET, POST, PUT, dan DELETE* untuk menerima, membuat, memperbarui dan menghapus resource.

Application Programming Interface (API) adalah sebuah *tools* yang membuat beberapa sistem dapat saling terhubung. Pada *API* terdapat dua bagian, yaitu *server yang* berfungsi sebagai penyedia dari *API* dan klien yang berbentuk sebuah program yang mengetahui data apa yang tersedia pada API dan dapat memanipulasi data tersebut sesuai *request* pengguna. Pada API berbasis website dibagi menjadi dua yaitu *REST API* dan *SOAP API*. *REST API adalah* API berbasis website yang menggunakan teknologi *REST* dan menggunakan *format JSON (JavaScript Object Notation)*.

2.4 *UML (Unified Modeling Language)*

Unified Modeling Language (UML) adalah sebuah bahasa yang berdasarkan gambar untuk menvisualisasikan menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis objek. *UML* bukanlah merupakan bahasa pemrograman tetapi model-model yang tercipta berhubungan langsung dengan berbagai macam bahasa pemrograman, sehingga memungkinkan melakukan pemetaan (*mapping*) langsung dari modelmodel yang dibuat dengan *UML* degan bahasa-bahasa pemrograman berorientasi obyek, seperti *Java*

2.5 *Use Case Diagram*

Merupakan pemodelan untuk melakukan *(behavior)* sistem informasi yang akan dibuat. *Usecase* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. secara kasar usecase digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi itu. Simbol dan keterangan *use case diagram*.

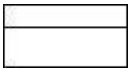
Tabel 2.1 *Use Case* Diagram

No	Gambar	Nama	Keterangan
1		*Actor*	Menspesifikasikan himpuan peran yang pengguna mainkan Ketika berinteraksi dengan *use case*.
2		*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempengaruhi elemen yang bergantung padanya elemen yang tidak mandiri(*independent*).
3		*Generalization*	Hubungan dimana objek anak (*descendent*) berbagi perilaku dan struktur data dari objek yang ada di atasnya objek induk (*ancestor*).
4		*Include*	Menspesifikasikan bahwa *usecase* sumber secara *eksplisit*.
5		*Extend*	Menspesifikasikan bahwa *use case* target memperluas perilaku dari *use case* sumber pada suatu titik yang diberikan.
6		*Association*	Apa yang menghubungkan antara objek satu dengan objek lainnya.

2.6 *Class Diagram*

Class diagram merupakan gambaran struktur sistem dari segi pendefinisian kelaskelasyang akan dibuat untuk membangunsistem. *Class diagram* terdiri dari atribut danoperasi dengan tujuan pembuat pembuat program dapat membuat hubungan antara dokumentasi perancangan dan perangkat lunak sesuai.

Tabel 2.2 *Class* Diagram

No		Gambar	Nama	Keterangan
1			*Generalization*	Hubungan dimana objek anak *(descendent)*berbagi perilaku dan struktur data dari objekyang ada di atasnya objek induk (*ancestor*).
2			*Nary Association*	Upaya untuk menghindari asosiasi denganlebih dari 2 objek.
3			*Class*	Himpunan dari objek-objek yang berbagiatribut serta operasi yang sama.
4			*Collaboration*	Deskripsi dari urutan aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang rukur bagi suatu aktor
5			*Realization*	Operasi yang benar-benar dilakukan olehsuatu objek.

6			*Dependency*	Hubungan dimana perubahan yang terjadi pada suatu elemen mandiri *(independent)* akan mempegaruhi elemen yang bergantung padanya elemen yang tidak mandiri

2.7 *Activity Diagram*

Activity Diagram merupakan diagram yang menggambarkan *workflow* atau aktivitas dari sebuah sistem yang ada pada perangkat lunak.

Tabel 2.3 *Activity* Diagram

No	Gambar	Nama	Keterangan
1		***Actifity***	emperlihatkan bagaimana masingmasing kelas antarmuka saling berinteraksi satu sama lain
2		***Action***	State dari sistem yang mencerminkan eksekusi dari suatu aksi
3		***Initial Node***	Bagaimana objek dibentuk atau diawali.
4		***Actifity FinalNode***	Bagaimana objek dibentuk dan dihancurkan

5		***Fork Node***	Satu aliran yang pada tahap tertentu berubahmenjadi beberapa aliran

2.8 Penelitian Terkait

Dalam penyusunan skripsi ini, peneliti terinspirasi dan mereferensi dari penelitian penelitian sebelumnya yang berkaitan dengan skripsi ini. Daftar penelitian terkait adalah sebagai beriku

Tabel 2.4 Penelitian Terkait

Peneliti	Judul	Kesimpulan
1.Theo Rapi Ridwan, Anton Ariyanto, Alfi Rahmi	ANALISIS PERPIPAAN AIR BERSIH MENGGUNAKAN APLIKASI E-PANET 2.0 Studi Kasus di Universitas Pasir Pengaraian	:analisis sistem pendistribusian air pada gedung yang beragam, agar kontinuitas kebutuhan air setiap lantai dapat terpenuhi untuk mempermudah analisa sistem plumbing
2.Fajar Antono dan Saruni Dwiasnati	Implementasi Absensi Karyawan Menggunakan Algoritma Haversine dengan Global Posisitioning System Berbasis Android	Metode Haversine digunakan untuk menghitung jarak antara titik di permukaan bumi menggunakan garis lintang (longitude) dan garis bujur (latitude) sebagai variabel inputan. Haversine formula adalah persamaan penting pada navigasi, memberikan jarak lingkaran besar antara dua titik pada permukaan bola (bumi). berdasarkan bujur dan lintang.

		Dengan mengasumsikan bahwa bumi berbentuk bulat sempurna dengan jari-jari R 6.367, 45 km, dan lokasi dari 2 titik di koordinant bola (lintang dan bujur) masing-masing adalah lon1, lat1, dan lon2, lat2
3. Clarissa Chandra dan Joko Susilo	PERANCANGAN SISTEM PEMESANAN *ONLINE* JASA *PRINT OUT* BERBASIS ANDROID MENGGUNAKAN METODE *SCRUM*	Metode pengembangan adalah sebuah cara yang tersistem atau teratur yang bertujuan untuk melakukan analisa pengembangan suatu sistem agar sistem tersebut dapat memenuhi kebutuhan.