BAB II
LANDASAN TEORI

2.1 Udang Vanamei

(Rakasiwi dan Albastomi, 2017) Udang vannnamei atau biasa juga disebut udang vanname (Litopenaeus vannamei) merupakan udang introduksi. Habitat asli udang Vannamei adalah di perairan Pantai Pasifik Barat Amerika Latin. Litopenaeus vannamei yang pada masa awal masuk ke Indonesia sebagian berasal dari Nikaragua dan Meksiko.

Udang vannamei sedang berkembang pesat dibudidayakan di Indonesia akhir-akhir ini. Udang Vannamei memiliki banyak keunggulan dibandingkan jenis udang lain, antara lain lebih tahan serangan penyakit, pertumbuhan lebih cepat, masa pemeliharaan lebih singkat, daya tahan hidup tergolong tinggi selama pemeliharaan, pemberian pakan yang relatif lebih mudah, nilai FCR (Feed Convertion Ratio) cukup rendah sehingga pembudidaya dapat hemat dalam pengeluaran untuk pakan.

Salah satu kunci penunjang keberhasilan usaha budidaya udang vannamei adalah kemampuan memodifikasi lingkungan perairan tambak yang sesuai dengan kebutuhan hidup dan pertumbuhan udang.

Karena itu, pengetahuan tentang sifat udang vannamei perlu dipahami secara mendalam, terutama mengenai morfologi, siklus hidup, tingkah laku, dan seluruh faktor yang mempengaruhi kelangsungan hidup vannamei.

2.2Multimedia

(Arfida, 2015) multimedia adalah suatu kombinasi data atau media untuk menyampaikan suatu informasi sehingga informasi itu tersaji dengan lebih menarik. Sedangkan definisi lainnya menyatakan bahwa multimedia merupakan kombinasi dari teks, grafis, seni, suara, animasi dan video yang dikirimkan oleh komputer atau peralatan elektronik lain. Kata media berasal dari bahasa latin dan merupakan bentuk jamak dari kata *medius* yang berarti tengah, perantara atau pengantar. Dengan kata lain, media adalah komponen sumber belajar atau wahana fisik yang mengandung materi instruksional di lingkungan siswa yang dapat merangsang siswa untuk belajar. Multimedia megandung beberapa media, seperti teks, *audio*, *video*, *image* dan *animation*. Berikut penjelasan tentang objek-objek dalam multimedia:

a. Teks

   Teks adalah bentuk data multimedia yang paling mudah disimpan dan dikendalikan. Kebutuhan teks tergantung pada kegunaan aplikasi multimedia.

b. Grafik

   Grafik menjadi nilai dan unsur tambah suatu penyajian data. Gambar digunakan dalam presentasi multimedia untuk menarik perhatian.

c. Gambar Vektor

   Gambar vektor disimpan sebagai serangkaian instruksi yang digunakan untuk membuat suatu gambar yang dinamakan algoritma, yang menentukan bentuk kurva, garis dan berbagai bangun yang diwakilkan oleh gambar (*picture*). Untuk menyimpan gambar yang tidak terlalu banyak mengandung unsur perubahan warna, gambar vektor adalah pilihan yang lebih tepat.

d. Gambar *Bitmap*

   Gambar *bitmap* adalah gambar yang tersimpan dalam rangkaian *pixel* (titik – titik). Komputer akan mengatur tiap titik di layar sesuai dengan detil warna *bitmap*.

e. Suara (*Audio*)

   Penyampaian sebuah informasi yang sering disertai desain grafis dan teks yang menarik, akan terasa membosankan apabila tidak disertai dengan suara.

f. *Video*

   Video menyediakan sumber daya yang kaya dan membuat aplikasi multimedia lebih hidup. Namun kendala yang dihadapi adalah ukuran file yang terlalu besar. Untuk itu diperlukan software lain untuk memperkecil ukuran file video.

g. Animasi (*Animation*)

   Animasi dalam multimedia merupakan penggunaan komputer untuk menciptakan gerak pada layar.

2.3 Aplikasi

(Sari Y.P, 2016), *mobile phone* adalah salah satu perangkat yang bergerak seperti telepon seluler atau komputer bergerak yang digunakan untuk mengakses jasa jaringannya. Pada *mobile application* juga digunakan untuk mendeskripsikan aplikasi internet yang berjalan pada *smartphone* serta piranti *mobile* lainnya

2.4 Android

(Sari, Y. P., & Ali, R. 2019). Android adalah sebuah sistem operasi untuk *smartphone* dan tablet. Sistem operasi dapat diilustrasikan sebagai ‘jembatan’ antara piranti (*device)* dan penggunanya, sehingga pengguna bias berinteraksi degan *device*-nya dan menjalankan aplikasi-aplikasi yang tersedia pada *device*. *Mobile Phone* adalah salah satu perangkat yang bergerak seperti telepon seluler atau computer bergerak yang digunakan untuk mengakses jasa jaringanya. Pada *mobile application* juga digunakan untuk mendeskripsikan aplikasi internet yang berjalan pada *smartphone* serta piranti *mobile* lainya.

2.5 Perangkat Lunak yang Digunakan

2.5.1 Adobe Flash Profesional CS6

(Agus Supandi,2016), *Adobe Flash Profesional CS6* adalah salah satu perangkat lunak komputer yang merupakan produk unggulan *Adobe System*. *Adobe Flash Profesional CS6* merupakan *software* yang digunakan untuk menciptakan animasi dan konten multimedia. Desain pengalaman *immersive* interaktif yang hadir secara konsisten di seluruh *desktop* dan beberapa perangkat, termasuk *tablet, smartphone, dan televisi*. Dengan *Adobe Flash Profesional CS6* kita dapat dengan mudah menggabungkan beberapa simbol dan urutan animasi menjadi lembaran *sprite* tunggal dan dioptimalkan untuk alur kerja yang lebih baik, dibuat lebih menarik dengan konten menggunakan ekstensi asli untuk mengakses kemampuan perangkat secara spesifik. *Adobe Flash Profesional CS6* telah membuktikan dirinya sebagai program animasi dua dimensi berbasis vector dengan kemampuan professional. Dalam perkembangannya, *Adobe Flash* selalu melakukan banyak penyempurnaan pada setiap versinya. *Adobe Flash Profesional CS6* menghadirkan fitur-fitur baru yang menjadikan *flash* semakin diakui sebagai program yang handal.

2.5.2 Action Script

(Mudiyanto Setiawan, 2016) *Action Script* terdiri dari 2 kata, yaitu : action (aksi) dan *script* (tulisan/naskah) yang beraksi. *Action scipt* adalah bahasa pemrograman yang digunakan di *Flash* dan hingga saat ini sudah mencapai 3 versi.

- *Action Script* 1.0 (tahun 2000 – tahun 2003) mulai dipergunakan pada *Flash* 5 dengan minimal dimainkan di *Flash player 5*
- *Action Script* 2.0 (tahun 2003 – tahun 2006) mulai dipergunakan pada *Flash MX* 2004 dengan minimal dimainkan di *Flash player 7*
- *Action Script* 3.0 (tahun 2006 – sampai sekarang) mulai dipergunakan pada *Flash CS3* dengan minimal dimainkan di *Flash player 9*

Di *Flash Action Script* ditulis pada panel *actions*. Penulisan *action script* di panel *action* dapat dilakukan pada 3 tempat yaitu *movie clip, button*, dan *frame*. *Action*

Script adalah bahasa pemrograman yang dibuat berdasarkan *ECMAScript*, yang digunakan dalam pengembangan situs *web* dan perangkat lunak menggunakan *platform Adobe Flash Player*.

Action script juga dipakai pada beberapa aplikasi basis data, seperti *Alpha Five*. Bahasa ini awalnya dikembangkan oleh *Macromedia*, tapi kini sudah dimiliki dan dilanjutkan perkembangannya oleh *Adobe*, yang membeli *Macromedia* pada tahun 2005. Action dibagi dalam berbagai kategori, yaitu:

- *Basic Actions*: Kategori ini menampung *action* sederhana yang sering sekali digunakan untuk *Movie Flash*, seperti navigasi dan perilaku tombol.
- *Actions*: Kategori ini meliputi *Basic Actions* ditambah dengan banyak action lain yang lebih kompleks.
- *Operators*: Kategori ini berisi simbol yang digunakan misalnya untuk operasi logika dan matematika, seperti tambah, kurang, kali.
- *Functions*: Berisi action yang dapat menerima data tertentu untuk kemudian menghasilkan informasi yang dapat kita gunakan.
- *Properties*: Kategori ini berisi properti objek yang dapat dimodifikasi. Sebagian besar properti ini digunakan untuk objek klip *movie*.
- *Object*: *Flash* memiliki kelas objek yang sudah didefinisikan (*predefined class*). Kelas-kelas ini berada dalam kategori *Objects* di *Action Script*.

2.5.3 Adobe AIR

(Ikhsan Ra'id 2016), *Adobe AIR (Adobe Integrated Runtime)* adalah sebuah *cross operating system runtime* yang dikembangkan oleh *Adobe* sehingga memungkinkan pengembang memanfaatkan keterampilan mereka (seperti *Flash, Flex, HTML, Javascript,* dan *PDF*) untuk membangun *RIA* (*Rich Internet Application*) dan contentnya ke dalam *platform* baru.

2.5.4 Animasi

(Uswatun Hasanah, 2017) Animasi adalah simulasi gerakan yang dihasilkan dengan menayangkan rentetan Frame ke layar. Fram adalah satu Gambar tunggal pada rentetan Gambar yang membentuk animasi. Animate adalah untuk membuat sesuatu hidup, sebagian orang mengira bahwa animasi itu sama dengan motion (gerakan), tetapi animasi mencakup semua yang mengandung efek visual sehingga animasi mencakup perubahan posisi terhadap waktu, bentuk, warna, struktur, tekstur dari sebuah objek, posisi kamera, pencahayaan, orientasi dan fokus dan perubahan dalam teknik rendering.

2.6 *UML*

(Purwati, N., Halimah, H., & Rahardi, A. , 2018) mendefinisikan *Unified Modelling Language (UML)* adalah sebuah "bahasa" yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak, *UML* menawarkan sebuah standar untuk merancang model sebuah sistem. Tujuan Penggunaan *UML* yaitu untuk memodelkan suatu sistem yang menggunakan konsep berorientasi objek dan menciptakan bahasa pemodelan yang dapat digunakan baik oleh manusia maupun mesin.

2.6.1 Use Case Diagram

(Purwati, N., Halimah, H., & Rahardi, A. , 2018), Menjelaskan, *Use Case* Diagram adalah gambar dari beberapa atau seluruh aktor dan *use case* dengan tujuan yang mengenali interaksi mereka dalam suatu sistem. *Use case* diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem, yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* mempresentasikan sebuah interaksi antara *actor* dan sistem.

Simbol-simbol yang ada pada *use case diagram* dapat dilihat pada tabel 2.1 berikut:

Tabel 2.1 *Simbol Use Case Diagram*

No	Simbol	Deskripsi
1.	*Use Case* 	Fungsionalitas yang disediakan system sebagai unit-unit yang saling nertukar pesan anter unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama *use case*.
2.	Aktor /*Actor*  Nama aktor	Orang, proses, atau system lain yang berinteraksi dengan system informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor beum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal rase nama aktor.
3.	Asosiasi / *Association* 	Komunikasi antara aktor dan *use case* yang berpartisipasi pada *use case* memiliki interaksi dengan aktor.
4.	Ekstensi / *Extend* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu mirip dengan prinsip *inhetirance* pada pemrograman berorientasi objek, biasanya *use case* tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan.

5.	Generalisasi/ *Generalization* 	Hubungan generalisasi dan spesialisasi(umum-khusus) antara dua buah use case dimana fungsi yang satu adalah fungsi yang lebih umum dari lainya.
6.	Menggunakan/ *Include / Uses* 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat dijalankan use case ini. *Include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan.

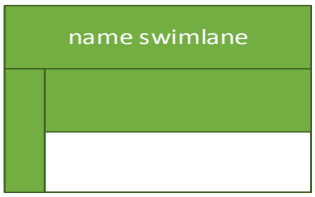
2.6.2 Activity Diagram

(Purwati, N., Halimah, H., & Rahardi, A. , 2018) Mendefinisikan, Activity diagram menggambarkan rangkaian aliran dari aktifitas, digunakan untuk mendeskripsikan aktivitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi. *Activity Diagram* berupa *flowchart* yang digunakan untuk memperlihatkan aliran kerja dari sistem.

Simbol-simbol yang ada pada *activity diagram* dapat dilihat pada tabel 2.2 berikut:

Tabel 2.2 *Simbol Activity Diagram*

No.	Simbol	Deskripsi
1.	Status Awal 	Status awal aktivitas system, sebuah diagram aktivitas memiliki sebuah status awal.

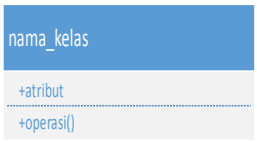
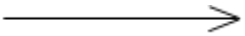
2.	Aktivitas 	Aktivitas yang dilakukan system, aktivitas biasanya diawali dengan kata kerja.
3.	Percabangan / *Decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
4.	Penggabungan / *Join* 	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
5.	Status akhir 	Status akhir yang dilakukan system, sebuah diagram aktivitas memiliki sebuah status akhir.
6.	Swimline 	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi.

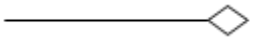
2.6.3 *Class Diagram*

(Purwati, N., Halimah, H., & Rahardi, A. , 2018) *Class Diagram* menggambarkan struktur data dan deskripsi *class*, *package* dan objek beserta hubungan satu sama lain. *Class Diagram* berfungsi untuk menjelaskan tipe dari objek sistem dan hubungannya dengan objek yang lain.

Simbol-simbol yang ada pada *class diagram* dapat dilihat pada tabel 2.3 berikut:

Tabel 2.3 *Simbol Class Diagram*

No.	Simbol	Deskripsi
1.	Kelas 	Kelas pada struktur sistem
2.	Antarmuka / *Interface* 	Sama dengan konsep *interface* dalam pemrograman berorientasi objek.
3.	Asosiasi / *Association*	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
4.	Asosiasi berarah / *Directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*.
5.	Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
6.	Kebergantungan / *Dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas.

7.		Relasi antar kelas dengan makna semua-bagian (*whole-part).*

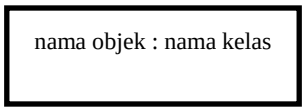
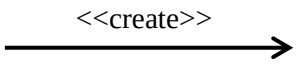
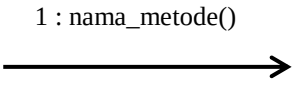
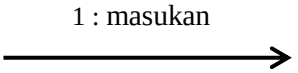
2.6.4 *Sequence Diagram*

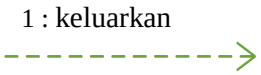
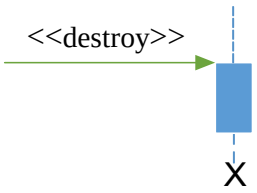
Sequence diagram menggambarkan kolaborasi dinamis antara sejumalah dan untuk menunjukkan rangkaian pesan yang dikirim antar objek juga interaksi antar objek, sesuatu yang terjadi pada titik tertentu dalam eksekusi sistem. *Sequence diagram* menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Dalam *sequence diagram* terdapat dua simbol yaitu *Actor* (untuk menggambarkan pengguna system) dan *Lifeline* (untuk menggambarkan kelas dan objek)

Simbol-simbol yang ada pada *Sequence diagram* dapat dilihat pada tabel 2.4 berikut:

Tabel 2.4 Simbol *Sequence Diagram*

No.	Simbol	Deskripsi
1.	Aktor  **Nama aktor** atau Nama aktor tanpa waktu aktif	Orang, proses, atau system lain yang berinteraksi dengan system informasi yang akan dibuat, jadi walaupun simbol dari aktor adalah gambar orang, tapi aktor beum tentu merupakan orang, biasanya dinyatakan menggunakan kata benda di awal rase nama aktor.

2.	Garis hidup / *Lifeline*	Menyatakan kehidupan suatu objek
3.	Objek 	Menyatakan objek yang berinteraksi pesan
4.	Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah sebuah tahapan yang dilakukan didalamnya
5.	Pesan tipe *create* 	Menyatakan suatu objek membuat objek lain, arah panah mengarah pada objek yang dibuat.
6.	Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri.
7.	Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainya, arah panah mengarah pada objek yang dikirimi.

8.	Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang menerima kembalian.
9.	Pesan tipe destroy 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada destroy.

2.7 Pengujian *Black boxs*

(Purwati, N., Halimah, H., & Rahardi, A. , 2018) Black box testing adalah pengujian perangkat lunak dari segi spesifikasi fungsional tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak sesuai dengan spesifikasi yang dibutuhkan. Pengujian black box dilakukan dengan membuat kasus uji yang bersifat mencoba semua fungsi dengan memakai perangkat lunak apakah sesuai dengan spesifikasi yang dibutuhkan. Pengujian black box berusaha menemukan kesalahan dalam kategori:

1. Fungsi-fungsi yang tidak benar atau hilang.
2. Kesalahan interfase
3. Kesalahan dalam struktur data atau akses database eksternal
4. Kesalahan Kinerja
5. Inisialisasi dan kesalahan terminasi

2.8 Metode Pengumpulan Data

(Tamagola,Raka, dan Puput Budi Wintoro) metode pengumpulan data ini dilakukan untuk memudahkan dalam mendapatkan data yang diperlukan. Berikut beberapa metode pengumpulan data yang digunakan :

a. Observasi

Observasi (Pengamatan Langsung) merupakan salah satu teknik pengumpulan data yang tidak hanya mengukur sikap dari responden (wawancara dan angket) namun juga dapat digunakan untuk merekam berbagai fenomena yang terjadi (situasi, kondisi).

b. Wawancara

(Amnah, 2016) menjelaskan, teknik wawancara merupakan teknik pengumpulan data atau fakta yang dilakukan dengan cara menanyakan langsung kepada bagian yang terkait sesuai yang dibutuhkan dalam proses penelitian skripsi.

c. Studi Pustaka

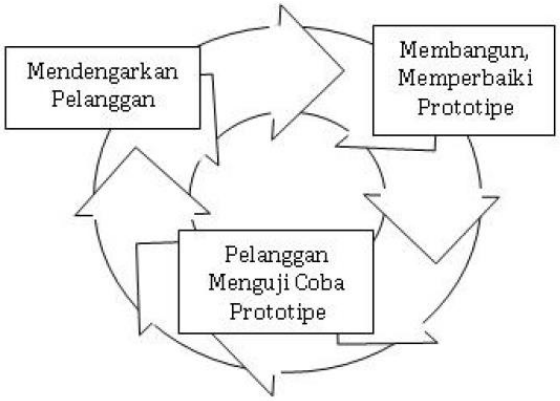
(Amnah, 2016) menjelaskan, studi pustaka mempelajari buku-buku serta literatur-literatur yang ada pada perpustakaan, mempelajari beberapa alur karya ilmiah yang berkaitan dengan judul yang diangkat, dan mempelajari bentuk-bentuk data pengolahan data sebagai dasar informasi.

2.9 Metode Pengembangan Perangkat Lunak

Metode Pengembangan Perangkat Lunak yang digunakan dalam penelitian ini adalah proyotype, berikut penjelasan prototype.

2.9.1 Prototype

(Artaye, 2018) menjelaskan, *prototype* model dimulai dengan mengumpulkan kebutuhan, pengembang dan pelanggan bertemu dan mendifinisikan objek keseluruhan dari perangkat lunak, mengidentifikasi segala kebutuhan yang diketahui dan kemudian melakukan "perancangan kilat". Perancangan kilat berfokus pada penyajian dari aspek-aspek perangkat lunak tersebut yang akan nampak bagi pelanggan atau pemakai (contohnya pendekatan input dan format output). Perancangan kilat membawa kepada kontruksi sebuah *prototype. Prototype* tersebut dievaluasi oleh pelanggan dan dipakai untuk menyaring kebutuhan pengembangan perangkat lunak, seperti gambar 2.4 berikut :



Gambar 2.4 Ilustrasi Model Prototipe

(Kadapi, 2018), uraian dari tahapan Model *Prototype* adalah sebagai berikut:

1) Pengumpulan kebutuhan

   Developer dan klien bertemu untuk menentukan tujuan umum, kebutuhan yang diketahui dan gambaran bagian-bagian yang akan dibutuhkan berikutnya. Selanjutnya melakukan analisis terhadap data apa saja yang dibutuhkan

2) Perancangan

   Perancangan dilakukan dengan cepat dan rancangan mewakili semua aspek software yang diketahui, dan rancangan ini menjadi dasar pembuatan *prototype*.

3) Evaluasi *prototype*

   Calon pengguna mengevaluasi *prototype* yang dibuat dan digunakan untuk memperjelas kebutuhan *software*. *Software* yang sudah dijalankan, dilakukan perbaikan apabila kurang memuaskan.