# 2. LANDASAN TEORI

## 2.1. Tinjauan Pustaka

### 2.1.1. Modul Pembelajaran Digital

Modul adalah suatu unit program pembelajaran yang didesain sedemikian rupa untuk membantu peserta mencapai suatu tujuan pembelajaran (Hernawan et al., 2012). Melalui modul pembelajaran siswa diberikan kesempatan untuk belajar secara mandiri dengan kecepatan dan waktu masing-masing.

Modul pembelajaran digital merupakan sebuah modul ajar yang menggunakan media elektronik sebagai medium penyajian. Melalui media elektronik penyampaian bahan pembelajaran bisa dibawakan dengan lebih menarik dan mudah diakses (Himawan, 2021). Modul pembelajaran digital dapat mencakup bahan audio, video/film, untuk mempermudah siswa mengerti materi pembelajaran di dalamnya.

### 2.1.2. Audio and Visual Learning

Setiap murid memiliki cara belajar yang unik untuk dapat menyerap informasi dengan lebih baik. Murid akan lebih mudah belajar jika situasi dan kondisi yang tepat sudah terpenuhi, sehingga mereka akan cepat mengerti dalam waktu yang lebih singkat (Sreenidhi & Helena, 2017). Salah satu teori model belajar yang umum digunakan adalah Fleming's VAK/VARK model.

Dalam Fleming's VAK/VARK model terdapat 4 model pembelajaran, yaitu visual, aural, read/write, dan kinestetik. 4 model pembelajaran ini saling berkaitan dengan bagaimana masing-masing murid bisa lebih cepat untuk memproses informasi saat mereka belajar.

#### 2.1.2.1. Audio

Murid dengan gaya belajar aural/auditory sangat memperhatikan kata-kata yang disampaikan oleh sang guru. Murid dengan gaya belajar ini bisa lebih cepat belajar dengan mendengar materi yang dibicarakan (Othman & Amiruddin, 2010). Beberapa ciri-ciri yang dimiliki oleh murid dengan gaya belajar ini seperti: mudah terganggu keributan, senang mengulangi apa yang dia dengar dengan lantang, suka berbicara dan berdiskusi, dan sangat pintar mengeja kata-kata.

#### 2.1.2.2. Visual

Murid dengan model belajar visual cenderung akan lebih cepat untuk belajar dengan adanya model atau peraga. Mereka sangat terorganisir dan suka mengamati, namun konsentrasi mereka bisa teralihkan dengan gerakan orang disekitarnya (Sreenidhi & Helena, 2017). Murid dengan gaya belajar ini suka menggunakan gambar maupun simbol dan mudah mereprentasikan ide melalui warna, diagram, dan bagan.

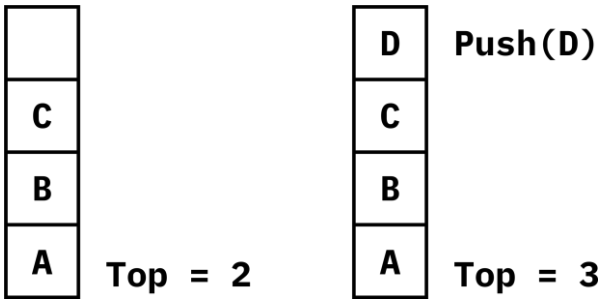
### 2.1.3. Struktur Data

Struktur Data adalah sebuah format untuk memproses, menerima dan menyimpan data (Loshin & Lewis, 2021). Struktur Data mempermudah pengguna untuk mengakses dan menggunakan data yang mereka butuhkan dengan cara yang tepat. Tipe - tipe Struktur Data meliputi sebagai berikut:

#### 2.1.3.1. *Stack*

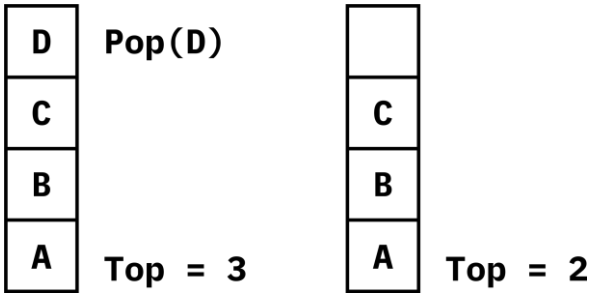
Sebuah tipe Struktur Data berbentuk linier dan mengikuti urutan tertentu. Dalam konsep sebuah *Stack* urutan yang digunakan adalah *Last In First Out* (LIFO), yaitu data terakhir yang masuk akan menjadi data pertama yang keluar (Maliyekkel & M, 2019). Beberapa keunggulan sebuah *Stack* seperti dapat mengelola data secara efisien, dan alokasi memori yang lebih terkontrol. Cara untuk memasukkan data ke dalam *Stack* adalah dengan menggunakan fungsi Push() dan untuk mengeluarkan data dari *Stack* maka digunakan fungsi Pop() (Marcus Zakarria & Prijono, 2006).

Operasi Push() adalah operasi untuk memasukkan data ke dalam *Stack*. Contoh operasi Push() dalam *Stack* dapat dilihat pada Gambar 2.1.




Gambar 2.1 Contoh operasi Push() pada *Stack*.

Operasi Pop() adalah operasi untuk mengeluarkan data dari dalam *Stack*. Contoh operasi Pop() dalam *Stack* dapat dilihat pada Gambar 2.2.

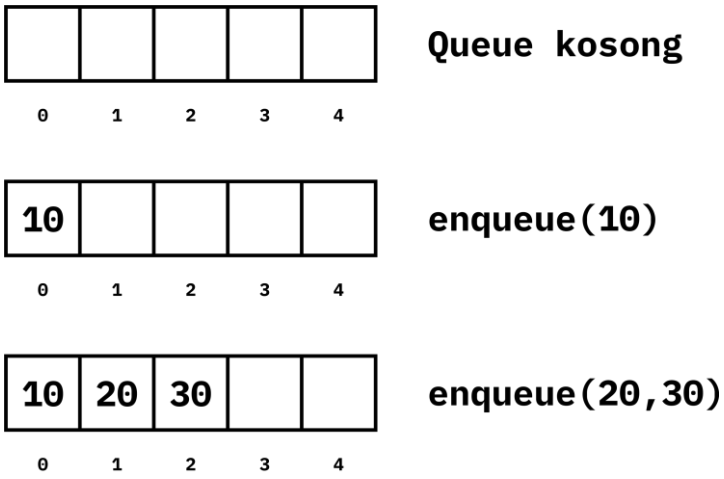



Gambar 2.2 Contoh operasi Pop() pada *Stack*.

#### 2.1.3.2. *Queue*

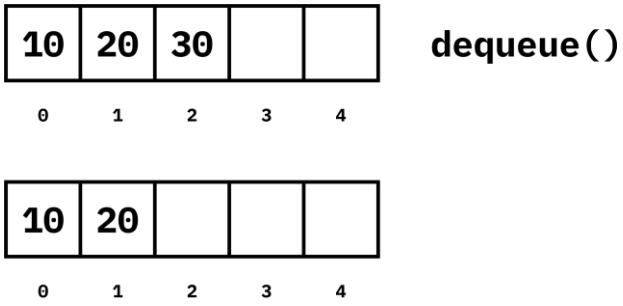
Sebuah metode penyimpanan data yang memiliki konsep mirip dengan *Stack*. Namun perbedaan dalam *Queue* itu sendiri adalah operasi data dimulai pada data pertama dalam *Queue*. *Queue* menggunakan sistem *First In First Out* (FIFO) (Maliyekkel & M, 2019).

Operasi untuk memasukkan data ke dalam Queue disebut *enqueue()* dan operasi untuk menghapus data dari Queue disebut *dequeue()*. Contoh dari *enqueue()* bisa dilihat pada Gambar 2.3.




Gambar 2.3 Contoh proses *enqueue()* pada *Queue*.

Operasi untuk menghapus data disebut dequeue(). Pada operasi ini data yang pertama dimasukkan dalam Queue akan dihapus terlebih dahulu. Contoh operasi dequeue() bisa dilihat pada Gambar 2.4.

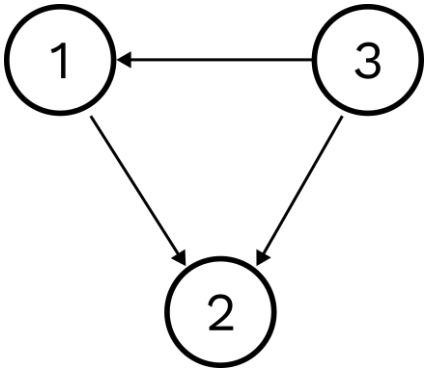



Gambar 2.4 Contoh proses *dequeue()* pada *Queue*.

#### 2.1.3.3. *Graph*

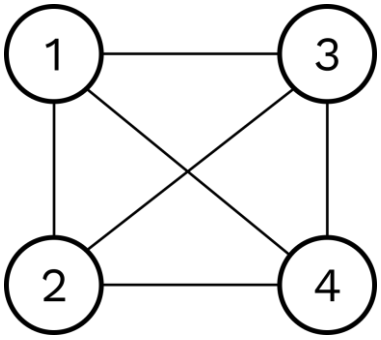
Sebuah *Graph* menyimpan koleksi item secara tidak linear. *Graph* terbuat dari kumpulan *node* yang terbatas, atau disebut *vertices*, dan kumpulan garis yang menghubungkan mereka, atau disebut *edges* (Kudase & Bane, 2016). Graph terdiri dari sejumlah verteks/*vertices* (V) dan sejumlah *edge* (E), dimana setiap *edge* akan saling dihubungkan oleh verteks. *Graph* memiliki banyak contoh dan kegunaan di dunia nyata, salah satunya adalah jaringan internet. *Graph* terbagi menjadi *Directed Graph* dan *Undirected Graph.*

*Directed Graph* merupakan jenis Graph dimana setiap verteksnya mengarah dari satu edge ke edge lainnya. Urutan dari edge pada directed graph termasuk penting, sehingga digambarkan dengan tanda panah untuk mengindikasikan arahnya. Contoh Directed Graph bisa dilihat pada Gambar 2.5.




Gambar 2.5 Directed Graph.

Undirected Graph merupakan jenis Graph dimana setiap verteksnya memiliki dua arah. Urutan dari edge pada graph dianggap tidak penting dan verteks pada graph ini digambarkan dengan garis lurus. Contoh Undirected Graph bisa dilihat pada Gambar 2.6.



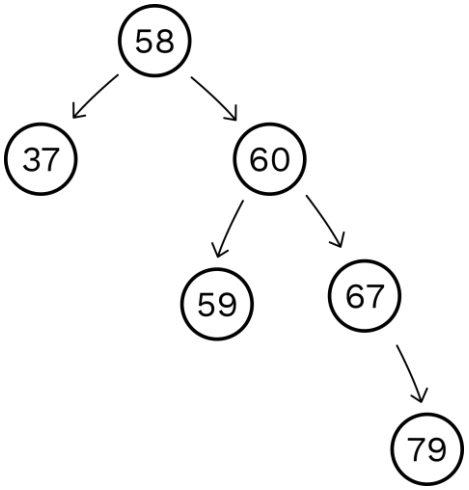

Gambar 2.6 Undirected Graph.

#### 2.1.3.4. *Tree*

Sesuai dengan namanya, *Tree* menyimpan data secara hirarki dan memiliki bentuk seperti pohon. Setiap *Tree* direpresentasikan sebagai sebuah kumpulan dari node yang berhubungan dan memiliki isi, dan setiap node tersebut tidak memiliki refrensi yang sama dan tidak ada yang mengarah ke *root* dari *Tree* (Dhankar et al., 2014). Data dalam *Tree* disusun dalam berbagai level. Setiap *node* dalam *Tree* saling berhubungan membentuk beberapa tipe (GeeksforGeeks, 2023), yaitu:

a. *Child Node*: Node yang merupakan penerus langsung dari sebuah node disebut node anak dari node tersebut.
b. *Root Node*: Node paling atas dari pohon atau node yang tidak memiliki simpul induk disebut simpul akar.
c. *Leaf Node* atau *External Node*: Node yang tidak memiliki node anak disebut node daun.
d. *Ancestor Node*: Setiap simpul pendahulu di jalur akar ke simpul itu disebut Leluhur dari simpul itu.
e. *Descendant Node*: Setiap node penerus di jalur dari node daun ke node itu.
f. *Sibling*: Anak-anak dari simpul induk yang sama disebut saudara kandung.
g. *Node Level*: Hitungan tepi pada jalur dari simpul akar ke simpul itu. Node root memiliki level 0.
h. *Internal node*: Sebuah node dengan setidaknya satu anak disebut Node Internal.
i. *Neighbour Node*: Parent atau child node dari node tersebut disebut tetangga dari node tersebut.
j. *Sub-tree*: Setiap simpul pohon bersama dengan keturunannya.

Contoh dari *Tree* dapat dilihat pada Gambar 2.7.




Gambar 2.7 Contoh *Tree*.

#### 2.1.3.5. *AVL Tree*

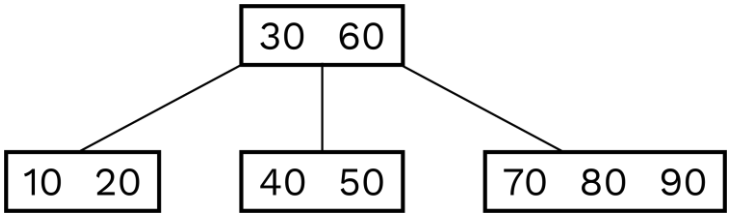
*AVL Tree* bisa didefinisikan sebagai self-balancing Binary Search *Tree*, dimana setiap nodes yang ada dalam *Tree* ini diasosiasikan dengan sebuah balance factor (perbedaan tinggi *sub-tree* kiri dan kanan) (Deepa et al., 2017). Untuk menghitung balance factor, yang perlu dilakukan adalah mengurangi tinggi *sub-tree* kanan dengan *sub-tree* kiri. Hasil yang didapat setelah pengurangan tersebut tidak boleh lebih dari 1 (berada dalam rentang -1 hingga 1).

#### 2.1.3.6. *B-Tree*

Berbeda dari jenis-jenis *Tree* lainnya, *B-Tree* merupakan sebuah tipe *Tree* yang bisa menyimpan banyak data dalam satu nodenya (GeeksforGeeks, 2023a). *B-Tree* dapat dikategorikan sebagai self-balancing *Tree*, dimana setiap nodes memiliki jumlah minimal data yang harus disimpan di dalamnya. Beberapa karateristik yang dimiliki *B-Tree* yaitu:

a. Semua leaf berada pada level yang sama.
b. *B-Tree* didefinisikan dengan istilah "minimum degree", biasa disebut 't'.
c. Setiap node kecuali root harus berisi setidaknya t-1 key.
d. Root mungkin berisi minimal 1 key.
e. Semua node (termasuk root) dapat berisi paling banyak (2*t-1) key.
f. Jumlah anak dari sebuah node sama dengan jumlah key di dalamnya ditambah 1.

Beberapa operasi yang ada pada sebuah *B-Tree* yaitu searching (mencari data), insert (memasukkan data) dan delete (menghapus data) (Koruga & Bača, 2010). Contoh dari *B-Tree* bisa dilihat pada Gambar 2.8.




Gambar 2.8 Contoh B-Tree.

### 2.1.4. Blackbox Testing

Metode *Blackbox* merupakan sebuah bentuk pengujian yang dilakukan dengan mengamati hasil *input* dan *output* dari perangkat lunak tanpa adanya pengetahuan tentang kode dari perangkat lunak tersebut. Pengujian *Blackbox* biasanya dilakukan pada akhir tahap pembuatan perangkat lunak untu memastikan agar perangkat lunak tersebut bisa berjalan dan berfungsi dengan baik. Pengujian ini diharapkan bisa mengidentifikasi kendala yang dapat menghambat pengguna dalam mengerjakan tugasnya (Setiawan et al., 2020).

## 2.2. Tinjauan Studi

### 2.2.1. Rancangan Website Pembelajaran Terintegrasi dengan Modul Digital Fisika Menggunakan 3D PageFlip Professional (Bakri et al., 2016)

1. Masalah yang diangkat dalam penelitian ini adalah pemindahan modul Fisika ke media berbasis online untuk membantu proses pembelajaran yang lebih efektif dan efisien.
2. Metode yang digunakan dalam penelitian ini dengan menggunakan pengembangan berbasis tahapan 4D (Define, Design, Develop, dan Dissemination). Define yaitu mendefinisikan syarat yang harus dipenuhi oleh produk yang dihasilkan. Design dimana sang penulis merancang modul online berbasis 3D PageFlip Professional. Pada tahap Develop, penulis mengembangkan 3D PageFlip Professional yang akan ditampilkan dalam bentuk website pembelajaran. Dan tahap terakhir Disseminate untuk mempromosikan produk kepada pengguna.
3. Hasil dari penelitian ini tercipta sebuah website pembelajaran Fisika yang berbasis 3D PageFlip Professional. Semua modul dari 3D PageFlip dapat berfungsi dengan baik saat website dijalankan.
4. Perbedaan penelitian yang dilakukan dalam skripsi ini dijelaskan pada Tabel 2.1.

Tabel 2.1 Tabel perbandingan

| Penelitian Terdahulu | Yang dikerjakan dalam penelitian ini |
|---|---|
| Menggunakan 3D PageFlip Professional untuk mengembangkan modul pembelajaran. | Membuat *website* yang di-*coding* dari awal sehingga memungkinkan tampilan dan isi *website* untuk lebih fleksibel. |
| Memfokuskan integrasi mata pelajaran Fisika dengan modul *e-learning* berbasis *website*. | Memfokuskan pada pengembangan *website* berisi video animasi pembelajaran mata kuliah Struktur Data. |

### 2.2.2. Rancangan dan Implementasi Sistem E-Learning Berbasis Web (Indrawan & Nugraha, 2020)

1. Masalah yang diangkat dalam penelitian ini adalah kurangnya sarana pendukung materi pembelajaran di SMP Negeri 1 Singaraja yang dapat diakses dan digunakan kapan saja. Hal ini menghambat proses pembelajaran antara guru dan murid di sekolah tersebut.
2. Metode yang digunakan dalam penelitian ini adalah metode pengembangan perangkat lunak model *waterfall*. Tahapan dalam metode ini meliputi: rekayasa dan pemodelan, analisis, desain, implementasi, pengujian dan pemeliharaan.
3. Hasil dari penelitian ini berupa sebuah *e-learning* berbentuk *website*. Penggunaan *e-learning.*
4. *g* berkontribusi positif pada kualitas pembelajaran karena sudah disesuaikan dengan kurikulum dan kebutuhan siswa.
5. Perbedaan penelitian yang digunakan pada skripsi ini terdapat pada Tabel 2.2.

Tabel 2.2 Tabel perbandingan.

| Penelitian terdahulu | Yang dikerjakan dalam penelitian ini |
|---|---|
| Membuat sebuah *Learning Management System* (LMS) berdasarkan kurikulum Sekolah Menengah Atas (SMA). | Fokus untuk membuat *website* berisi video animasi pembelajaran berdasarkan kurikulum Struktur Data yang ada. |