BAB II

TINJAUAN PUSTAKA

2.1 Penelitian Terkait

Smart Timer Application merupakan aplikasi Kesehatan Digital berbasis *mobile*. Aplikasi ini digunakan untuk mengatur waktu penggunaan aplikasi sesuai dengan kebutuhan pengguna, melihat waktu penggunaan *smartphone*, pengingat durasi penggunaan *smartphone*, dan notifikasi peringatan untuk menutup aplikasi jika sudah pada batas yang ditentukan pengguna. Beberapa platform telah membuat aplikasi yang berguna untuk memberikan informasi mengenai penggunaan *smartphone* diantaranya telah terdapat di *Google Play Store* seperti aplikasi Kesehatan Digital, *Action Dash*: Kesehatan Digital, dan *Stay Free* – Kesehatan Digital. Aplikasi tersebut secara umum telah memberikan fitur berupa waktu penggunaan *smartphone*, riwayat peluncuran aplikasi, riwayat pemberitahuan, mode tidur, mode fokus, dan batas penggunaan aplikasi. Sama halnya dengan beberapa pengabdian berikut yang menjadi referensi penelitian yang akan dilakukan.

Pada penelitian yang dilakukan oleh Arief Witjaksono dkk meliputi pemeriksaan refraksi subjektif dan objektif, survei prevalensi kelainan refraksi, penyuluhan dan edukasi kesehatan mata khususnya yang berkaitan dengan dampak durasi bermain *game* yang terlalu lama terhadap kesehatan mata. Dalam penelitian tersebut menyatakan bahwa jika terlalu sering berhadapan di depan monitor dapat merusak mata akibat sinar radiasi yang

dipaparkan oleh layar monitor yang terlalu lama dilihat maka sinar – X, sinar ultraviolet, gelombang mikro (*microwave*), radiasi elektromagnetik frekuensi sangat rendah (*Very Low Frequency*/VLF) dan radiasi elektromagnetik frekuensi amat sangat rendah (*Extremely Low Frequency*/ELF) tersebut akan ditangkap oleh kornea mata, selanjutnya cahaya tersebut diteruskan ke lensa, lensa tersebut dapat rusak khususnya lensa mata pada anak usia sekolah karena secara fisiologis saraf [5].

Begitupun dengan penelitian yang dilakukan oleh Sara Prot dkk menyatakan dampak negatif dari *game online* diantaranya mata yang terlalu lama terpapar pencahayaan layar komputer berpotensi mengalami *Computer Vision Syndrome* (CVS) yang berpengaruh pada tajam penglihatan. *American Refraksi Optimist Association* mendefinisikan CVS sebagai kumpulan gejala gangguan penglihatan akibat penggunaan komputer yang terlalu lama. Gejala CVS dapat berupa mata lelah, sakit kepala, mata kering, penglihatan buram, nyeri pada leher dan bahu. Berat ringannya keluhan yang dilaporkan sebanding dengan lamanya waktu yang digunakan di depan komputer [6].

Selain itu, pada penelitian yang dilakukan oleh Windy Patadungan dkk melalui observasional analitik menggunakan pendekatan *cross sectional*, hasil penelitian tersebut menunjukkan bahwa intensitas lama paparan cahaya *smartphone* (jam) dalam sehari tidak berpengaruh terhadap ketajaman penglihatan atau visus (p-*value* = 0,769 > 0,05). Tidak adanya pengaruh tersebut diduga dipengaruhi oleh adanya jeda waktu atau waktu dimana otot mata beristirahat dalam menggunakan *smartphone* sehari sehingga mata

dapat terhindar dari kelelahan mata [4].

2.2 Landasan Teori

2.2.1 Sistem

Menurut Sutarman (2012:13) bahwa “Sistem adalah kumpulan elemen yang saling berhubungan dan berinteraksi dalam satu kesatuan untuk menjalankan suatu proses pencapaian suatu tujuan utama”. Menurut Fatansyah (2015:11) bahwa “Sistem adalah sebuah tatanan (keterpaduan) yang terdiri atas sejumlah komponen fungsional (dengan satuan fungsi dan tugas khusus) yang saling berhubungan dan secara bersama-sama bertujuan untuk memenuhi suatu proses tertentu”. Dari beberapa pengertian di atas penulis dapat menyimpulkan bahwa sistem merupakan sekumpulan elemen, himpunan dari suatu unsur, komponen fungsional yang saling berhubungan dan berinteraksi satu sama lain untuk mencapai tujuan yang diharapkan [7].

2.2.2 Alarm

Alarm merupakan alat yang digunakan untuk menetapkan aktivitas. Alarm sangat mudah digunakan dan ditemui pada *smartphone* maupun jam alarm [8]. Alarm juga dapat digunakan untuk membatasi penggunaan *smartphone* yang difungsikan untuk mengurangi durasi penggunaan *smartphone*.

2.2.3 Flutter SDK

Flutter adalah SDK untuk pengembangan aplikasi *mobile* dengan kinerja tinggi, aplikasi untuk IOS dan Android, dari satu *codebase* (basis kode) yang di buat oleh *google* dengan lisensi *open source*. Bertujuan untuk menghadirkan aplikasi yang berkinerja tinggi pada platform yang berbeda [9].

2.2.4 Dart

Dart adalah bahasa pemrograman yang dikembangkan oleh *google* untuk kebutuhan dalam membuat aplikasi android atau *mobile*. Dart merupakan konsep *Object Oriented Programing* (OOP) yang memiliki struktur kode berada dalam *class* yang didalamnya berisi method maupun variabel. Dart sendiri menggunakan *C-Style syntax* sehingga mekanisme dart mirip dengan bahasa pemrograman C, java, javascript, dan swift [10].

2.2.5 Kotlin

Kotlin adalah bahasa pemrograman yang dikembangkan oleh Jetbrains. Kotlin dianggap bahasa pemrograman yang mudah dipelajari dibandingkan dengan java untuk pengembangan aplikasi android. Kotlin bisa digunakan untuk aplikasi *non-smartphone* atau biasa disebut dengan *native* [11].

2.2.6 Visual Studio Code

Visual Studio Code adalah aplikasi teks *editor* gratis di kembangkan oleh *Microsoft* yang dapat digunakan di semua bahasa

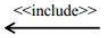
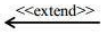
pemrograman yang ada tanpa perlu berganti aplikasi *editor*, serta dapat dijalankan di berbagai *platform Operating Sistem (OS)* seperti *windows, linux* ,dan *mac OS.* Visual Studio Code memudahkan para *programmer* saat berganti bahasa pemrograman tanpa perlu berganti aplikasi *editor* serta memahami dan konfiguarasi *tools* kembali di aplikasi *editor* barunya. Visual Studio Code juga memberikan kebebasan kepada penggunanya dalam tema, *debugger*, *extension*, dan lainnya [12].

2.2.7 UML (*Unified Modelling Language*)

UML (*Unified Modelling Language*) merupakan penyedia bahasa pemodelan visual yang dapat digunakan oleh pengembang untuk membuat *blueprint* yang mudah dimengerti serta dilengkapi dengan mekanisme efektif untuk *sharing* dan mengkomunikasikan rancangan tersebut dengan yang lain. UML memudahkan dalam merancang *software* berorientasi objek [13]. Berikut beberapa jenis dari UML yang digunakan beserta simbol dan keterangannya:

1. *Use Case* Diagram.

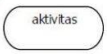
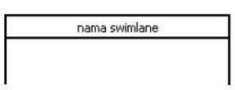
   Use case diagram menggambarkan interaksi sistem dengan *actor*.

Simbol	Keterangan
	Aktor : Mewakili peran orang, sistem yang lain, atau alat ketika berkomunikasi dengan *use case*
	*Use case* : Abstraksi dan interaksi antara sistem dan aktor
	Association : Abstraksi dari penghubung antara aktor dengan use case
	Generalisasi : Menunjukkan spesialisasi aktor untuk dapat berpartisipasi dengan use case
	Menunjukkan bahwa suatu use case seluruhnya merupakan fungsionalitas dari use case lainnya
	Menunjukkan bahwa suatu use case merupakan tambahan fungsional dari use case lainnya jika suatu kondisi terpenuhi

Gambar 1. *Use Case* Diagram [14]

2. *Activity* Diagram.

   Activity diagram digunakan untuk melakukan pemodelan dari proses yang terjadi pada sistem.

Simbol	Deskripsi
status awal 	status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal
aktivitas 	aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja
percabangan / *decision* 	asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu
penggabungan / *join*	asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu
status akhir 	status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir
swimlane 	memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi
fork, 	digunakan utk menunjukkan kegiatan yg dilakukan secara paralel
join, 	digunakan utk menunjukkan kegiatan yg digabungkan

Gambar 2. *Activity* Diagram [14]

2.2.8 Pengujian Sistem

1. *White Box Testing*

   Pengujian *white box* digunakan berdasarkan sumber kode dan mampu menentukan bagian mana kode yang tidak berfungsi dengan baik dan benar. Beberapa kelebihan *white box testing* diantaranya mampu menghapus bagian asing dari kode-kode yang tersembunyi, melakukan pengujian secara menyeluruh karena seluruh bagian dari struktur atau logika dieksplorasi. Membantu dalam mengoptimalkan kode, dan pengujian *white box testing* dapat dimulai walaupun GUI masih dalam masa pengembangan [15]. Dalam melakukan pengujian pada aplikasi *Smart Timer* dengan metode *white box testing* penguji akan melihat kode secara keseluruhan kemudian melakukan testing terhadap setiap kode yang ada jika kode tidak berfungsi maka bisa dihapus atau digantikan dengan kode yang difungsikan sebagaimana mestinya.

2. *Black Box Testing*

   Pengujian *black box* disebut sebagai pengujian perilaku. Struktur interior, logika perangkat lunak yang diuji tidak diketahui oleh penguji. Penguji didasarkan kepada spesifikasi kebutuhan dan tidak perlu dilakukan analisis kode. Kelebihan *black box testing* diantaranya membantu dalam hal penemuan aspek yang tidak terpenuhi dari spesifikasi kebutuhan yang diberikan dalam

pengembangan perangkat lunak. Sedangkan kekurangan *black box testing* diantaranya pengujian tidak dapat dilakukan sepenuhnya dikarenakan pengetahuan penguji terbatas tentang perangkat lunak yang diuji [15]. Dalam pengujian pada aplikasi *Smart Timer* menggunakan metode *black box testing* penguji hanya melihat *interface* aplikasi kemudian dilakukan pengujian pada fungsi dari spesifikasi aplikasi.