PERANCANGAN TYPING GAME BERBASIS WEB DENGAN PENERAPAN METODE AGILE SOFTWARE DEVELOPMENT

[1]Sandi Tendean, [2]Krisyesika, [3]Vernando Saputra
[123]Program Studi Informatika Fakultas Teknologi Informasi Universitas Widya Dharma Pontianak
e-mail: [1]sandi_t@widyadharma.ac.id, [2]krisyesika@widyadharma.ac.id,
[3]19421318_vernando_s@widyadharma.ac.id

Abstract

This research was conducted because the author aimed to create a game that could make typing learning activities more interactive, improve speed, accuracy, and reduce errors in typing. The objective of this research was to implement the linear search algorithm to match the letters typed by players in a web-based typing game. In this study a system modeling language called Unified Modeling Language (UML) was applied to define the functional and non-functional requirements within the game. Furthermore, application development techniques that implemented agile software development were employed to enhance flexibility and effectiveness in software development. The outcome of this research is a typing game genre that incorporates the linear search algorithm. This algorithm is used to match the input, consisting of letters, with a randomly selected word within the game. The results of this research demonstrate that the linear search algorithm performs well in matching the letters typed by players. The research shows that the use of the linear search algorithm in a typing game is capable of providing efficient word validation. The implementation of UML as a system modeling language helps in modeling the system and understanding the interactions between components in the game. The agile development method allows for iterative development and responsiveness to user needs and feedback.

Keywords: *game, typing, linear search algorithm, software development,GDD*

Abstrak

Penelitian ini dilakukan karena penulis ingin menghasilkan permainan yang dapat membuat kegiatan belajar mengetik menjadi lebih interaktif, meningkatkan kecepatan, akurasi dan mengurangi kesalahan seseorang dalam mengetik. Tujuan penelitian ini adalah menerapkan algoritma *linear search* untuk mencocokkan huruf yang diketik oleh pemain dalam *typing game* berbasis web. Dalam penelitian ini menerapkan bahasa pemodelan sistem dengan menggunakan *Unified Modeling Language* (UML) yang digunakan untuk mendefinisikan kebutuhan fungsional dan non-fungsional di dalam permainan. Selanjutnya, teknik pengembangan aplikasi yang menerapkan metode *agile software development,* metode ini diterapkan untuk meningkatkan fleksibilitas dan efektivitas dalam pengembangan perangkat lunak. Hasil dari penelitian ini adalah permainan bergenre *typing* yang menerapkan algoritma *linear search.* Algortima ini digunakan untuk mencocokkan *input* yang berupa huruf dengan sebuah kata yang dipilih secara acak di dalam permainan. Hasil penelitian ini, menunjukkan bahwa algortima *linear search* dapat berfungsi dengan baik dalam mencocokkan huruf yang diketikkan oleh pemain. Penelitian ini menunjukkan bahwa penggunaan algoritma *linear search* dalam *typing game* mampu memberikan validasi kata yang efisien. Implementasi UML sebagai bahasa pemodelan sistem yang membantu dalam memodelkan sistem dan memahami interaksi antara komponen dalam permainan. Metode pengembangan agile memungkinkan adanya pengembangan yang iteratif dan responsif terhadap kebutuhan dan umpan balik pengguna.

Kata Kunci: permainan, *typing*, algoritma *linear search*, pengembangan perangkat lunak, GDD

1. PENDAHULUAN

Mengetik merupakan keterampilan dasar yang digunakan dalam pengoperasian komputer, keterampilan yang dimaksud adalah kecepatan serta keakuratan dalam mengetik yang dikombinasikan dengan tidak melihat tombol-tombol pada *keyboard* secara langsung pada saat mengetik. Mempunyai keterampilan mengetik cepat dan akurat dapat mempengaruhi beberapa hal, diantaranya berupa peningkatan kualitas tulisan yang dihasilkan, mengurangi kerja otak, dan juga dapat berpikir dengan cepat. Mengetik cepat juga berarti mengurangi waktu untuk menyelesaikan suatu tulisan. Namun, belajar atau latihan mengetik terkadang membosankan bagi sebagian orang. Setiap orang memiliki caranya masing-masing untuk mengatasi rasa bosan tersebut, salah satunya adalah dengan bermain *game*.

Game berasal dari Bahasa Inggris yang berarti permainan. *Game* adalah sebuah aktivitas yang dilakukan untuk mencari kesenangan dengan melibatkan pemain di dalamnya. Pemain yang terlibat diharuskan untuk mengikuti situasi atau aturan yang sudah dibuat di dalam *game* tersebut. Dalam dunia digital, *game* adalah

perangkat lunak yang dirancang, dibuat dan dikembangkan untuk dimainkan pada perangkat komputer, *smartphone*, konsol *game* atau platform lainnya.

Typing game adalah permainan yang dirancang untuk mengasah kecepatan dan akurasi mengetik. Pemain harus mengetikkan suatu kata atau kalimat yang ditampilkan dilayar secepat dan setepat mungkin. Biasanya, permainan ini memiliki waktu yang terbatas dan semakin lama pemain bertahan, semakin cepat dan sulit kata-kata yang harus diketik. *Typing game* umumnya digunakan untuk tujuan pendidikan dan hiburan, serta untuk membantu orang mengembangkan keterampilan mengetik yang lebih baik.

Dalam *typing game* dibutuhkan sebuah algoritma yang digunakan untuk melakukan pencocokan huruf yang diketik oleh pemain. Hal ini dapat dilakukan dengan bantuan algoritma *Linear Search* atau yang biasa dikenal juga dengan nama *Sequential Search*. Algoritma *Linear Search* merupakan proses pencarian data yang dilakukan dengan cara mencocokkan data yang akan dicari dengan semua data yang ada di dalam sebuah *array*. Proses pencarian dilakukan satu persatu sampai elemen yang dicari ditemukan.

Berdasarkan latar belakang yang telah diuraikan sebelumnya, tujuan dari penelitian ini adalah menerapkan algoritma *Linear Search* dalam pembuatan *typing game* berbasis web. Algoritma *Linear Search* akan digunakan untuk mencocokkan huruf yang tampil di dalam *game* dan huruf yang diketik oleh pemain. Dengan dibuatnya *typing game* ini diharapkan dapat membuat kegiatan belajar mengetik menjadi lebih interaktif, meningkatkan kecepatan, mengurangi kesalahan dan meningkatkan akurasi seseorang dalam mengetik.

2. METODE PENELITIAN

2.1 Metode Pengumpulan Data

2.1.1 Studi Literatur

Studi literatur merupakan teknik pengumpulan data dengan cara menganalisis sumber literatur yang berupa referensi, literatur atau teori yang mendukung serta memiliki kaitan dengan topik penelitian.

2.1.2 Observasi (Pengamatan)

Observasi merupakan teknik pengumpulan data melalui pengamatan terhadap sumber yang berkaitan dengan topik penelitian. Observasi dimulai dari mengamati permainan dengan genre yang sama, mencoba beberapa permainan tersebut agar mendapatkan gambaran mengenai penelitian yang akan dibuat.

2.2 Teknik Analisis Sistem

Merupakan proses untuk menentukan apa saja yang perlu dipelajari, serta mendefinisikan kebutuhan fungsional dan non-fungsional di dalam *Typing game* berbasis web dengan menggunakan diagram *Unified Modeling Language* (UML).

2.3 Teknik Perancangan Aplikasi

Merupakan proses perancangan yang terdiri dari berbagai elemen seperti tampilan antarmuka, kebutuhan fungsional dan non-fungsional. Hal ini dilakukan untuk memastikan bahwa aplikasi yang dirancang dapat berfungsi dengan baik, memenuhi kebutuhan dan sesuai dengan yang sudah direncanakan dengan menggunakan *Game Design Document* (GDD).

2.4 Teknik Pengembangan Aplikasi

Dalam pengembangan aplikasi metode yang digunakan adalah metode pengembangan perangkat lunak *agile*. Metode *agile* digunakan untuk meningkatkan fleksibilitas dan efektivitas dalam pengembangan untuk menghasilkan perangkat lunak dengan kualitas terbaik.

2.5 Landasan Teori

2.5.1 Game

Game berasal dari Bahasa Inggris yang berarti permainan. Dalam setiap *game* terdapat peraturan yang berbeda-beda untuk memulai permainannya sehingga membuat jenis *game* semakin bervariasi. Karena salah satu fungsi *game* sebagai penghilang rasa stres atau rasa jenuh maka hampir setiap orang senang bermain *game* baik anak kecil, remaja maupun dewasa, mungkin hanya berbeda dari jenis *game* yang dimainkannya saja[1]. *Game* adalah suatu cara untuk menghilangkan kepenatan dengan melakukan suatu kegiatan yang dilalui menggunakan kecerdasan berpikir dan strategi yang harus digunakan untuk berinteraksi dengan sistem dan konflik yang direkayasa secara sengaja untuk menimbulkan keseruan dalam bermain[2].

2.5.2 Linear Search

Sequential Search atau yang disebut pencarian linear merupakan metode pencarian yang paling sederhana. *Sequential Search* merupakan proses pencarian data dengan metode pencarian langsung. Pencarian dilakukan dengan cara mencocokkan data yang akan dicari dengan semua data yang ada dalam kelompok data[3]. *Sequential Search* adalah proses membandingkan setiap elemen *array* satu persatu secara beruntun dimulai dari elemen pertama hingga elemen yang dicari ditemukan atau hingga elemen terakhir dari *array*[4].

2.5.3 Metode Agile

Metode *agile* merupakan induk dari scrum. Jika scrum adalah kerangka kerja, maka *agile* adalah pelaksanaan proyek secara keseluruhan yang berskala besar. Metode ini tergolong modern, karena menekankan pada improvisasi dan adaptasi[5]."

Dalam mengembangkan perangkat lunak dengan Metode *Agile*, terdapat beberapa tahapan yang harus dilalui antara lain:

a. Perencanaan: merupakan langkah di mana tim pengembang dan juga klien merancang apa saja yang dibutuhkan dalam suatu perangkat lunak yang hendak dibuat.

b. Implementasi: merupakan tahapan di mana para tim pemrograman melakukan pengkodean pada suatu perangkat lunak.
c. Tes Perangkat Lunak: pada tahap ini, perangkat lunak yang telah diproduksi akan dites atau dicek, yang menjadi tanggung jawab bagian kontrol kualitas supaya bug yang masih ditemukan diperbaiki agar kualitas perangkat lunak tersebut tetap terjaga.
d. Dokumentasi: jika tahap tes perangkat lunak sudah selesai, kemudian dilanjutkan dengan proses dokumentasi yang mana tahap ini dimaksudkan untuk memberi kemudahan terhadap proses pemeliharaan atau maintenance ke depannya.
e. Deployment: merupakan tahap yang dilakukan untuk menjamin kualitas perangkat lunak yang diciptakan dengan menguji kualitas sistem. Jika sistem yang diproduksi telah memenuhi syarat, perangkat lunak tersebut nantinya sudah siap untuk dikembangkan.
f. Pemeliharaan: tahapan terakhir yang dilakukan dalam Metode Agile adalah pemeliharaan atau maintenance. Tahap ini ditujukan supaya tidak ada lagi bug yang mengganggu perangkat lunak. Maka dari itu, pemeliharaan ini merupakan tahap yang sangat penting dan harus dilakukan secara berkala agar kualitas selalu terjaga[6].

2.5.4 Game Design Document (GDD)

A game design document (GDD) is a design specification of a video game. It describes the game rules, its goals and mechanics, the ways the game can be played, and everything else concerning the way it should work." (Game Design Document (GDD) adalah sebuah spesifikasi rancangan video game. GDD menjelaskan aturan permainan, tujuan permainan dan mekanismenya, cara permainan bisa dimainkan, dan segala hal tentang caranya bekerja.)[7]

Game Design Document is a formal, living document describing the software for a video game as it is being developed. (Game Design Document adalah dokumen formal yang menjelaskan perangkat lunak untuk video game saat dalam pengembangan.)[8]

3. HASIL DAN PEMBAHASAN

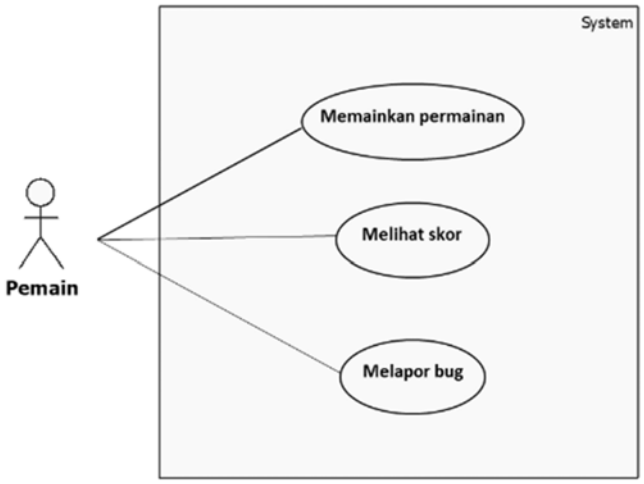
Dalam pembahasan ini akan menerapkan tahap-tahap dari metode *agile software development*. Adapun tahapan yang diterapkan adalah tahap perencanaan, implementasi, tes perangkat lunak dan dokumentasi.

3 1. Perencanaan

Pada tahap ini dilakukan perencanaan sistem yang akan dikembangkan dan akan dilakukan desain sistem atau pemodelan sistem menggunakan diagram *Unified Modeling Language* (UML). Dalam pemodelan diagram UML akan membahas dua diagram yaitu, diagram *use case* dan diagram *class*.

3.1.1. Unified Modeling Language (UML)

3.1.1.1. Use Case Diagram

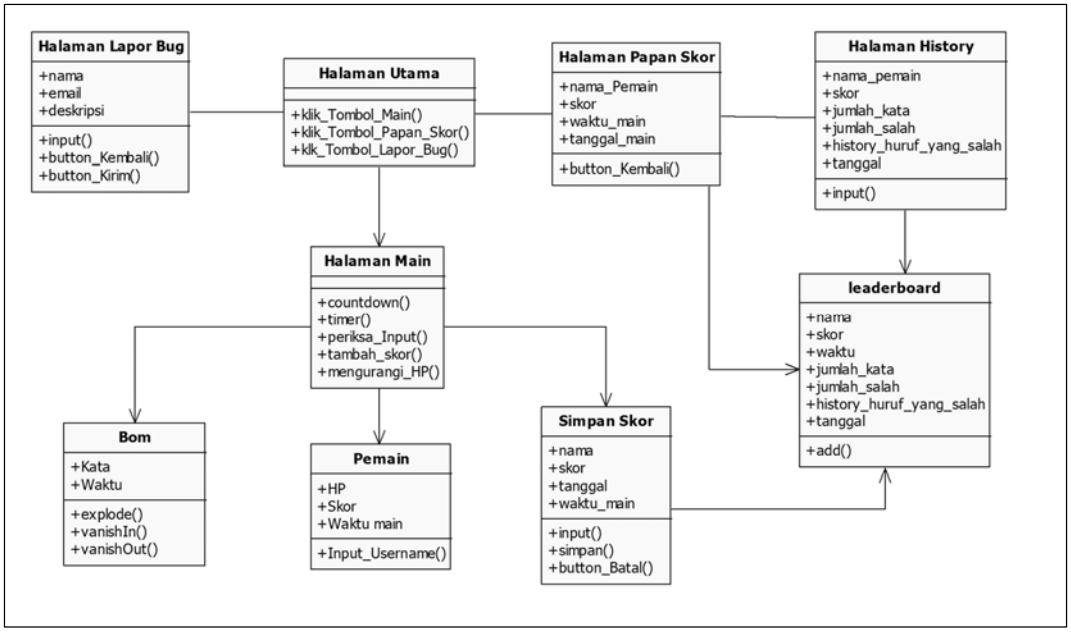


Gambar 1. Diagram *Use Case* Permainan

Pada Gambar 1, menggambarkan interaksi yang dapat dilakukan oleh pemain adalah memainkan permainan, melihat skor dan melapor *bug*.

3.1.1.2. Class Diagram

Pada Gambar 2 menunjukkan halaman utama, halaman main, halaman papan skor, lapor *bug* dan simpan skor merupakan *interface* dari sistem yang dirancang. Pemain dan bom adalah kelas-kelas yang yang berinteraksi dengan sistem pada saat bermain. Sedangkan *leaderboard* merupakan kelas yang berupa tabel untuk menampung nama, skor, waktu main dan tanggal pada saat pemain menjalankan fungsi simpan.



Gambar 2. Diagram *Class* Sistem

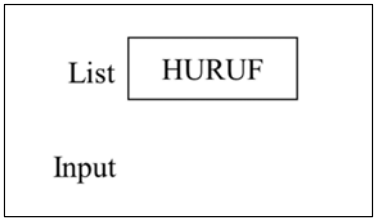
3 2. Implementasi

Pada tahap ini dilakukan penerapan atau implementasi algoritma *Linear Search* dalam pembuatan *Typing Game*. Algoritma *Linear Search* merupakan salah satu algoritma yang digunakan untuk melakukan pencarian suatu elemen seperti huruf, angka, atau posisi dari elemen tersebut. Algoritma *Linear Search* digunakan di dalam perancangan permainan ini dikarenakan algoritma ini cocok digunakan untuk melakukan pencocokan huruf yang diketik oleh pemain dengan kata yang akan muncul di dalam permainan.

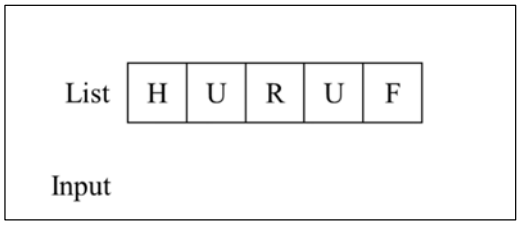
3.2.1. Ilustrasi Algoritma Linear Search yang Akan Diterapkan

Sebelumnya sudah membahas mengenai algoritma *Linear Search*, algoritma ini membutuhkan nilai dari suatu *list* dan nilai target yang akan dicari di dalam list tersebut. Algoritma linear search akan digunakan untuk mencocokkan *input* atau ketikkan dari pemain dengan kata yang telah dipilih secara acak dari sebuah *array* dengan cara membandingkan huruf yang diketik dengan huruf dari kata yang sudah terpilih.

Setelah mendapatkan sebuah kata, selanjutnya kata tersebut akan dibagi atau dipisah satu-persatu berdasarkan panjang kata tersebut. Berikut adalah ilustrasi dari penjelasan di atas:



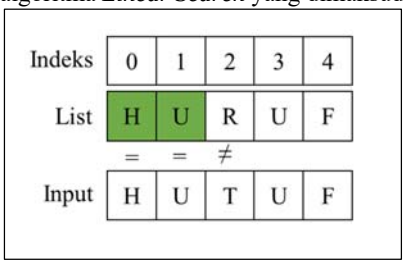
Gambar 3. Ilustrasi Kata yang Diambil Secara Acak



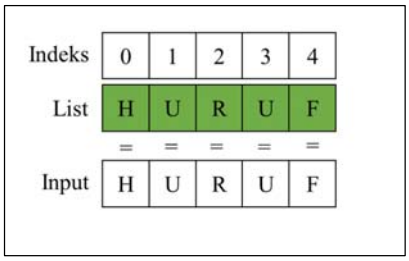
Gambar 4. Ilustrasi Kata Dipisah Perhuruf

Setiap huruf pada kata yang akan diketikkan oleh pemain akan dibandingkan dengan setiap huruf dari kata yang terpilih. Setelah dibandingkan, permainan akan memberikan tanda pada huruf yang diketikkan oleh pemain, jika huruf tersebut sama dengan huruf dari kata yang terpilih. Jika tidak sama maka permainan tidak akan memberikan tanda kepada huruf tersebut dan akan menjalankan fungsi lain.

Untuk melakukan perbandingan huruf-huruf tersebut, digunakan algoritma *Linear* Search. Berikut adalah ilustrasi dari penerapan algoritma *Linear Search* yang dimaksud:



Gambar 5. Ilustrasi *Input* Huruf Salah



Gambar 6. Ilustrasi *Input* Huruf Benar

3.2.2. Penerapan Algortima Linear Search

```
typed = String.fromCharCode(e.keyCode);
 if(e.keyCode >= 65 && e.keyCode <= 90){
    for (var i = 0; i < spans.length; i++) {
      if (spans[i].innerHTML == typed) {
        if (spans[i].classList.contains("bg")) {
          continue;
        } else if (spans[i].classList.contains("bg") == false &&
          spans[i-1] == undefined || spans[i-1].classList.contains
          ("bg") != false ) {
          spans[i].classList.add("bg");
          break;
        }
      } else if(spans[i].classList.contains("bg") == false &&
        spans[i].innerHTML != typed){
        salahInputAudio();
        decreaseHP(1);
      }
    }
```

Gambar 7. Penerapan Algoritma *Linear Search*

Sebelum menerapkan algoritma *Linear Search*, dibutuhkan suatu variabel yang digunakan untuk menampung *input*-an atau ketikkan dari pemain. Pada Gambar 7, dibaris kode yang diberikan warna merah yang digunakan untuk menampung *input*-an pemain. Selanjutnya, *input*-an pemain di dalam permainan akan dibatasi jadi pemain hanya mengetikkan alfabet yang berupa huruf A sampai Z. Selain membatasi *input*-an dari *keyboard* dibutuhkan juga fungsi yang digunakan untuk membandingkan huruf-huruf yang sudah ditampung dengan input-an dari pemain. Pada Gambar 7, dibaris kode yang diberikan warna biru digunakan untuk membatasi *input*-an pemain, melakukan perulangan dan perbandingan huruf-huruf yang sudah ditampung dengan input-an dari pemain.

Selanjutnya dibutuhkan penanda untuk huruf yang sudah dibandingkan dengan huruf yang belum dibandingkan. Hal ini dapat dilakukan seperti pada baris kode berwarna ungu pada Gambar 7. Selain itu hal terakhir yang dibutuhkan adalah ketika pemain salah *input* atau ketik, maka HP pemain berkurang sebanyak sisa huruf yang belum diketikkan. Pada Gambar 7, dibaris kode yang diberikan warna cokelat digunakan untuk memanggil fungsi yang dapat mengurangi HP pemain.

3 3. Tes Perangkat Lunak

Pada tahap ini, perangkat lunak yang telah selesai diproduksi akan diuji dengan menggunakan metode pengujian *black box*. Dalam tahapan pengujian ini akan dibagi menjadi beberapa pembahasan, yaitu menetapkan spesifikasi kebutuhan, melakukan test case, membandingkan hasil uji dengan hasil yang diharapkan dan kesimpulan dari hasil pengujian.

3.3.1. Spesifikasi Perangkat Lunak

Spesifikasi perangkat merupakan detail teknis yang menggambarkan kemampuan dan karakteristik dari suatu perangkat. Berikut adalah beberapa perangkat yang digunakan dalam pengujian dan spesifikasi dari perangkat tersebut:

a. Processor Intel® Celeron® CPU N3350 @ 1.10Ghz, 1101 Mhz, 2 logical processor(s).
b. Memori fisik internal (RAM) 4 GB .
c. Resolusi layar 1366 x 768 (32 bit) (60Hz).
d. Keyboard standar PS/2 *keyboard.*
e. Sistem operasi Microsoft Windows 8.1 Pro.
f. Web *browser* Google Chrome.

3.3.2. Test Case dan Hasil Uji

Tabel 1. Test Case dan Hasil Uji Dalam Permainan

Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian	Kesimpulan
Menekan tombol *enter*	Menjalankan fungsi *start*() Memuat *hit point* (HP), bom, *timer*, skor dan kata	Berhasil menjalankan fungsi *start*().Berhasil memuat HP, bom, *timer*, skor dan kata.	Berhasil
Menjalankan animasi bom, *timer* dan kata.	Memunculkan bom, *timer* dan kata dengan animasi.	Berhasil memunculkan bom, *timer* dan kata dengan animasi.	Berhasil
Menjalankan hitung mundur *timer*.	*Timer* berkurang setiap detiknya.	Berhasil mengurangi *value timer* setiap detiknya.	Berhasil
Jika *input* huruf sesuai.	Memberikan tanda pada huruf yang di-*input.*	Berhasil memberikan tanda pada huruf yang di-*input*-kan.	Berhasil
Jika *input* huruf tidak sesuai.	Mengurangi HP dan memainkan efek suara *input* salah.	Berhasil mengurangi HP dan memainkan efek suara *input* salah.	Berhasil
Jika *input* huruf dari suatu kata selesai.	Memberikan animasi keluar pada bom, *timer* dan kata. Menambah skor	Berhasil memberikan animasi keluar pada bom, *timer* dan kata. Berhasil menambah skor.	Berhasil
Jika waktu habis.	HP pemain berkurang sebanyak 10, dan memainkan efek suara ledakan.	Berhasil mengurangi HP dan memainkan efek suara ledakan.	Berhasil
Jika HP berkisar 60 persen sampai 30 persen	Menampilkan bar HP dengan warna kuning	Berhasil memvisualisasikan bar HP menjadi warna kuning	Berhasil
Jika HP dibawah 30 persen	Menampilkan bar HP dengan warna merah	Berhasil memvisualisasikan bar HP menjadi warna merah	Berhasil
Jika HP habis	Menampilkan notifikasi untuk menyimpan skor	Berhasil menampilkan notifikasi untuk menyimpan skor.	Berhasil
Jika pemain memilih menyimpan skor.	Menampilkan *form* untuk meng-*input* nama pemain.	Berhasil menampilkan form untuk meng-*input* nama pemain.	Berhasil
Jika pemain menyimpan skor tanpa meng-*input*-kan nama.	memberikan warning bahwa *field* nama harus diisi.	Berhasil memberikan *warning*.	Berhasil
Pemain *input* nama dan menekan *button* simpan.	Menyimpan skor ke *database*.	Skor berhasil tersimpan ke *database.*	Berhasil
Jika pemain memilih tidak menyimpan skor.	Menampilkan notifikasi dan kembali ke halaman utama.	Berhasil menampilkan notifikasi dan kembali ke halaman utama.	Berhasil

3.3.3. Kesimpulan

Berdasarkan pengujian menggunakan spesifikasi perangkat di atas, dapat disimpulkan bahwa , fungsi yang ada di dalam permainan bekerja sesuai dengan yang diharapkan.

3.4. Dokumentasi

Tahapan dokumentasi *typing game* ini akan menerapkan format *game design document*. *Game design document* akan dibagi menjadi lima bagian utama, yaitu ringkasan umum, *gameplay*, mekanisme permainan, elemen serta aset dari permainan yang akan dibuat.

3.4.1. Ringkasan Umum

3.4.1.1. Judul game

Judul dari permainan ini adalah “Typer Bomb”.

3.4.1.2. Konsep Game

Typer bomb adalah *game* bergenre *typing* yang bertujuan untuk menguji dan meningkatkan kecepatan serta ketepatan mengetik pemain. Pemain akan menghadapi serangkaian tantangan mengetik yang intens dan harus mengetikkan kata-kata yang muncul dengan cepat dan benar sebelum waktu habis.

3.4.1.3. Target Pemain

Target utama dari permainan ini adalah untuk semua orang yang ingin meningkatkan keterampilan mengetik, baik itu meningkatkan kecepatan atau ketepatan dalam mengetik.

3.4.1.4. Code Editor

Game dibuat dengan menggunakan Visual Studio Code sebagai *code editor* menggunakan JavaScript dan PHP sebagai bahasa pemrograman.

3.4.1.5. Platform Tujuan

Game ini dirancang hanya berbasis *website* dan tampilan yang dirancang khusus untuk ukuran layar desktop dengan sistem operasi yang universal (Windows atau MacOS).

3.4.2. Gameplay

3.4.2.1. Tujuan

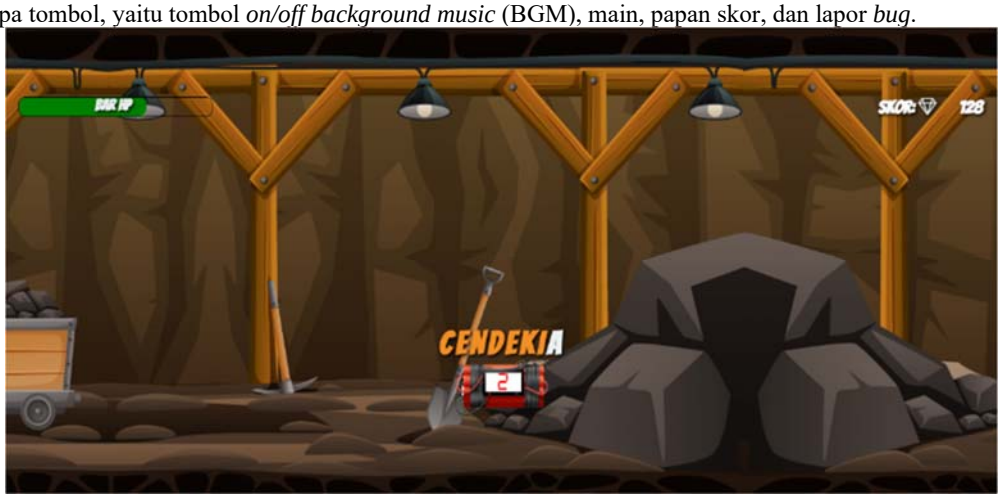
Permainan ini memiliki objektif utama yaitu mengetikan setiap kata yang muncul sebelum waktu habis dan mendapatkan skor sebanyak-banyaknya.

3.4.2.2. Desain Graphical User Interface (GUI)

Berikut ini merupakan tampilan dari halaman utama:

Gambar 8. Tampilan Halaman Utama

Pada Gambar 8 menampilkan halaman utama. Pada halaman utama pemain dapat berinteraksi dengan beberapa tombol, yaitu tombol *on/off background music* (BGM), main, papan skor, dan lapor *bug*.



Gambar 9. Tampilan Halaman Main

Pada Gambar 9 menampilkan halaman main, pada halaman ini pemain harus mengetik setiap kata yang muncul sebelum waktu habis dan memperoleh skor sebanyak-banyaknya sebelum *hit point* (HP) pemain habis.

Gambar 10. Tampilan Halaman Papan Skor

Pada Gambar 10, halaman papan skor menampilkan skor-skor yang sudah disimpan oleh pemain sebelumnya. Pada halaman ini hanya menampilkan top 10 skor tertinggi. Pada halaman ini pemain hanya dapat berinteraksi dengan dua tombol, yaitu tombol kembali untuk kembali ke halaman utama dan tombol *on/off* BGM untuk mengaktifkan dan menonaktifkan BGM.

Gambar 11. Tampilan Halaman History

Pada Gambar 4.11, halaman history menampilkan nama, skor, jumlah kata, jumlah salah, history huruf yang salah dan tanggal yang sudah disimpan oleh pemain sebelumnya. Pada halaman ini pemain hanya dapat berinteraksi dengan dua tombol dan satu bar pencarian, yaitu tombol kembali untuk kembali ke halaman utama dan tombol on/off BGM untuk mengaktifkan dan menonaktifkan BGM. Bar pencarian digunakan untu mencari history berdasarkan nama yang di-input oleh pemain.

Gambar 12. Tampilan Halaman Lapor Bug

Pada Gambar 12, halaman lapor *bug* menampilkan *form* yang harus diisi oleh pemain pada saat ingin melapor *bug*. Pemain wajib mengisi semua *field* jika ingin melaporkan *bug*. Jika ada *field* kosong, maka pemain akan diminta untuk mengisi *field* kosong tersebut. Selain itu pemain juga dapat berinteraksi dengan dua tombol, yaitu tombol kembali untuk kembali ke halaman utama dan tombol *on/off* BGM untuk mengaktifkan dan menonaktifkan BGM.

3.4.3. Mekanika

3.4.3.1. Aturan Permainan

a. Setiap kata yang muncul harus diketik oleh pemain sebelum waktu habis.
b. Jika pemain berhasil meng-*input* semua huruf di dalam kata yang muncul, maka pemain akan mendapatkan skor.
c. Jika pemain salah mengetik atau salah *input* huruf, maka *hit point* (HP) akan berkurang sesuai dengan panjang kata atau sisa dari huruf yang belum diketik.
d. Jika pemain kehabisan waktu, maka HP akan berkurang 10.
e. Waktu akan semakin berkurang ketika pemain mencapai skor tertentu.
f. Permainan akan berakhir jika HP pemain habis.

3.4.3.2. Cara Bermain

Pemain hanya perlu mengetikkan kata yang muncul dengan cepat, akurat dan benar sebelum kehabisan waktu. Dalam permainan ini pemain hanya perlu mendapatkan skor sebanyak-banyaknya sebelum HP pemain habis.

3.4.4. Elemen Permainan

3.4.4.1. Skor

Skor pemain bertambah berdasarkan waktu yang sudah ditentukan. Berikut perhitungan yang digunakan untuk perolehan skor pemain:

Skor = ⌊sisa_waktu * 0.3 + panjang_kata * 0.3⌋
Misalkan:
sisa_waktu = 9
panjang_kata = 12
Jadi:
Skor = ⌊9 * 0.3 + 12 * 0.3⌋
Skor = ⌊2.7 + 3.6⌋
Skor = ⌊6.3⌋
Skor = 6

Dapat dilihat pada Tabel 2, sisa waktu dapat mempengaruhi skor yang diperoleh. Jika sisa waktu lebih dari atau sama dengan 7 detik, maka pemain akan memperoleh skor sebanyak 5 detik. Jika sisa waktu lebih dari atau sama dengan 5 detik, maka pemain akan memperoleh skor sebanyak 4. Jika sisa waktu kurang dari 5 detik, maka pemain akan memperoleh skor sebanyak 3.

3.4.4.2. Desain Level

Level atau tingkat kesulitan di dalam permainan ini adalah waktu bom yang semakin cepat. Berikut tabel untuk memperjelas waktu bom berdasarkan skor yang sudah diperoleh.

Tabel 3. Detail Tingkat Kesulitan

Skor	Perolehan Waktu
Skor di bawah 50	panjang_kata * 100%
Skor di bawah 100	panjang_kata * 80%
Skor di bawah 200	panjang_kata * 70%
Skor di bawah 400	panjang_kata * 60%
Skor di bawah 800	panjang_kata * 50%
Skor di atas 800	panjang_kata * 40%

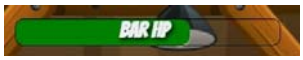
Berdasarkan Tabel 4.11, skor dapat memengaruhi perolehan waktu dari bom yang muncul. Perhitungan perolehan waktu bom dapat dilihat pada perhitungan berikut:

Jika skor dibawah 100
Maka, rumus perhitungannya adalah:
Waktu = ⌊panjang_kata * 0.8⌋
Misalkan:
panjang_kata = 12
Jadi:
Waktu = ⌊12 * 0.8⌋
Waktu = ⌊9.6⌋
Waktu = 9

3.4.5. Aset Permainan

3.4.5.1. Objek Permainan

Tabel 4. Objek Permainan

Nama	Objek	Keterangan
Bom		Bom merupakan musuh utama di dalam permainan ini. Bom akan muncul bersamaan dengan waktu dan kata yang harus diketik.
Waktu		Waktu merupakan parameter utama yang menentukan skor. Selain itu jika pemain kehabisan waktu, maka HP akan berkurang sebanyak 10.
Kata		Kata merupakan objek yang menjadi tantangan di dalam permainan. Setiap kata yang muncul harus diketikkan oleh pemain sebelum waktu yang ditentukan habis. Jika pemain kehabisan waktu atau melakukan kesalahan dalam mengetik, maka HP pemain akan berkurang. Jika pemain berhasil mengetikkan kata maka pemain akan memperoleh skor.
Bar HP		*Bar* HP merupakan indikator yang menunjukkan berapa HP pemain yang tersisa.
Skor		Skor merupakan angka yang menunjukkan berapa skor yang sudah diperoleh oleh pemain.

3.4.5.2. Efek Suara

Efek suara yang akan digunakan di dalam permainan antara lain:

a. Efek suara pada saat pemain salah mengetik.
b. Efek suara pada saat pemain selesai mengetik suatu kata.
c. Efek suara *timer*.
d. Efek suara pada saat waktu bom habis.
e. Efek suara pada saat pemain berinteraksi dengan sistem.

3.4.5.3. Background Music (BGM)

BGM merupakan musik latar belakang yang digunakan di dalam permainan agar permainan tidak terkesan monoton dan membosankan.

4. KESIMPULAN

Berdasarkan hasil perancangan, dan pengujian yang telah dilakukan, maka diperoleh kesimpulan sebagai berikut:

a. Algoritma *linear search* dapat dengan tepat membandingkan *input*-an pemain dengan huruf yang harus diketik.
b. Penerapan metode *agile* dalam penelitian ini membuat pengembangan perangkat lunak menjadi lebih flesksibel dan lebih efektif, karena semua tahapan yang diterapkan memungkinkan adanya pengembangan yang iteratif dan responsif terhadap kebutuhan dan umpan balik pengguna.
c. Berdasarkan pengujian yang sudah dilakukan, dapat disimpulkan bahwa permainan akan berfungsi sesuai dengan yang diharapkan pada perangkat komputer dengan web *browser* yang sudah *up to date*. Jika permainan dijalankan pada web *browser* dengan versi yang sudah ketinggalan maka, permainan tidak akan berjalan sesuai dengan yang diharapkan.
d. Tahapan dokumentasi yang dilakukan dalam metode agile menerapkan format *Game Design Document* (GDD). Desain dan pengembangan Typer Bomb menjadi lebih terstruktur dan mudah dipahami untuk pengembangan berikutnya.

5. SARAN

Berdasarkan hasil penelitian yang sudah dibahas pada bab-bab sebelumnya, berikut beberapa saran untuk pengembangan typer bomb ini kedepannya:

a. Membuat Typer Bomb menjadi permainan yang kompetitif dengan menambah mode *multiplayer*.
b. Menambahkan pilihan mode baru, misalnya permainan menyediakan mode dalam bahasa Inggris.
c. Menambah fitur akun, agar pada saat menyimpan skor pemain tidak perlu mengisi nama berulang-ulang.
d. Meningkatkan kemampuan algoritma *linear search* dalalm membandingkan *input* dari pemain dengan dua buah kata atau lebih.

UCAPAN TERIMA KASIH

Kami mengucapkan terima kasih yang tulus atas dukungan, serta teman-teman seangkatan dan civitas akademi Universitas Widya Dharma Pontianak atas dukungan dan arahan terhadap penelitian ini.

DAFTAR PUSTAKA

[1] Ridoi, Mokhammad. (2018). *Cara Mudah Membuat Game Edukasi dengan Construct 2*. Penerbit Maskha. Malang.
[2] Sandy, Teguh Arie dan Wahyu Nur Hidayat. (2019). *Game Mobile Learning*. CV. Multimedia Edukasi. Malang.
[3] Rerung, Rintho Rante. (2020). *Algoritma dan Struktur Data untuk Perguruan Tinggi*. Insan Cendekia Mandiri. Solok.
[4] Buana, I Komang Setia, Herman Setiawan dan Prasetyo Adi Wibowo Putro. (2021). *Pemrograman Terstruktur*. Syiah Kuala University Press. Banda Aceh.
[5] Ndaumanu, Ricky Imanuel, Suwarti, Kristina, Jose Augusto Duarte Guterres, Ratna Dewi, Amna, Frederikus Suarezsaga, Wilda Susanti, Mohammad Sofyan S. Thayf, Marlina dan Arden Simeru. (2022). *Tahapan-Tahapan Rekayasa Perangkat Lunak.* Media Sains Indonesia. Bandung.
[6] Romindo, Reska Mayefis, Tri Yusnanto, Nono Heryana, Jamaludin, Allans Prima Aulia, Angga Aditya Permana, Sitti Aisa, Jhoni S Pasaribu, Wahyuddin S, Fredy AH Sihombing. (2023). *Rekayasa Perangkat Lunak*. PT GLOBAL EKSEKUTIF TEKNOLOGI. Padang.
[7] Cossu, Sebastiano. (2019). *Game Development with GameMaker Studio 2*. Apress. London.
[8] Lanzinger, Franz. (2020). *2D Game Development with Unity*. CRC Press. Boca Raton, Florida.