BAB II

TINJAUAN PUSTAKA DAN DASAR TEORI

2.1 Tinjauan Pustaka

Demi menunjang keberlangsungan penelitian berbagai karya ilmiah seperti jurnal, buku, dan skripsi terdahulu dijadikan sebagai tinjauan pustaka.

Studi pertama yang diambil adalah karya yang berjudul *What Aspects of Gacha Games Keep the Players Engaged?*. Penelitian ini meneliti tentang aspek-aspek dalam game *Gacha* yang membuat pemain tetap terlibat. Data dikumpulkan melalui wawancara semi terstruktur dengan lima pemain yang bermain game. Hasil penelitian menunjukkan bahwa daya tarik awal pemain adalah estetika game dan elemen naratifnya. Emosi yang kuat terkait dengan keberhasilan dan kegagalan dalam "tarikan" (*gacha*), dengan koleksionisme sebagai motivasi penting. Kelelahan akibat kurangnya konten baru juga dapat menyebabkan pemain berhenti. (Rentia & Karaseva, 2022).

Studi kedua diambil dari karya yang berjudul Implementasi *Wilcoxon Signed Rank Test* Untuk Mengukur Efektifitas Pemberian Video Tutorial Dan Ppt Untuk Mengukur Nilai Teori. Penelitian ini menggunakan Uji *Wilcoxon Signed Rank* untuk mengukur efektivitas pemberian video tutorial dan PPT untuk mengukur nilai teori. Metode yang digunakan menggunakan *One-Group pretest-posttest Design* dengan mengerjakan soal pilihan ganda sebanyak 20 soal. Hasil yang didapat dari penelitian ini terdapat kenaikan nilai antara sebelum dan sesudah diberikan materi berupa video tutorial dan power point text. (Astuti, Taufiq, & Muhammad, 2021).

Studi ketiga diambil berjudul Terapan *MCRNG* sebagai Generator Acak untuk Game. Dalam jurnal tersebut dijelaskan bagaimana metode *MCRNG* ini bekerja, yaitu dengan cara melakukan perkalian bilangan awal dengan bilangan pengali lalu melakukan modulus dengan bilangan pengumpan. Metode *MCRNG* dapat digunakan untuk melakukan pengacakan terhadap sebuah indeks, namun memiliki kelemahan dimana hasil dari pengacakan dapat memberikan nilai yang sama sehingga kurang cocok apabila diimplementasikan kepada sistem pengacakan yang mengharuskan setiap indeksnya berbeda contohnya pengacakan soal ujian. (Nurhasan & Purnomo 2022).

Studi keempat diambil dari karya yang berjudul *FACTORS THAT INFLUENCE GACHA GAME USERS' DECISION-MAKING IN USING GACHA GAMES.* Dalam jurnal tersebut menjelaskan bahwa keputusan pengguna untuk bermain dan terus menggunakan game *gacha* dipengaruhi oleh berbagai faktor. Jurnal ini menggunakan model *Unified Theory of Acceptance and Use of Technology 2* (UTAUT2) untuk menganalisis faktor-faktor tersebut. UTAUT2 mengidentifikasi beberapa konstruk utama yang memengaruhi niat perilaku dan penggunaan teknologi, termasuk game *gacha*. Faktor-faktor ini meliputi kinerja yang diharapkan, upaya yang diharapkan, pengaruh sosial, kondisi yang memfasilitasi, nilai hedonis, kebiasaan, dan harga. (Indah & Handayani 2024).

Studi kelima diambil dari karya Rizky Ramadhan membangun portal game berbasis web bernama Planet Game ID menggunakan framework Next.js dan arsitektur *microservice*. Portal ini bertujuan untuk memudahkan pemain menemukan dan memainkan game tanpa perlu menginstal atau memikirkan

spesifikasi perangkat. Penelitian ini menyoroti masalah umum pemain game, seperti perangkat dengan spesifikasi rendah dan kesulitan menemukan game berbasis HTML5 yang bisa langsung dimainkan di browser. (Rizky 2022).

Sedangkan penelitian yang diusulkan ini akan membahas pembuatan sistem *gacha* menggunakan metode *Multiplicative Congruential Random Number Generator (MCRNG).* Sistem dibuat menggunakan bahasa pemrograman JavaScript dengan bantuan *framework NextJS* dan *PostgreSQL* sebagai basis data.

Tabel tinjauan pustaka dibuat untuk melihat perbandingan penelitian yang diajukan dengan beberapa penelitian sebelumnya yang memiliki kesamaan topik atau metode. Tabel tinjauan pustaka sebagai berikut :

Tabel 2.1 Tinjauan Pustaka

Nama penulis	Topik/Sistem	Metode	Keunggulan	Hasil
Rentia & Karaseva (2022)	*What Aspects of Gacha Games Keep the Players Engaged?*	*Kualitatif Melalui Wawancara Semi-Terstruktur*	Pemahaman Perspektif Mendalam	Aspek-aspek Permainan *Gacha*
Astuti, Taufiq, & Muhammad (2021)	Implementasi *Wilcoxon Signed Rank Test* Untuk Mengukur Efektifitas Pemberian Video Tutorial dan Ppt Untuk Mengukur Nilai Teori	*Wilcoxon Signed Rank Test*	Mengukur data berkelompok dan Berpasangan	Uji kenaikan nilai *pretest* dan *posttest*
Usman Nurhasan, Bagus Purnomo (2022)	Terapan MCRNG sebagai *Generator* Acak untuk *Game*	*Multiplicative CRNG (Congruential Random Number Generator)*	Kemampuan pengacakan pada indeks.	Permainan edukasi menebak gambar hewan vertebrata dan invertebrata berbasis android
Bunga Puspa Indah, Sri Handayani (2024)	*Factors That Influence Gacha Game Users' Decision-Making In Using Gacha Games*	*The Unified Theory of Acceptance and Use of Technolo gy 2 (UTAUT2)*	Validitas tinggi, Prediksi yang Baik, Fleksibilitas	Faktor yang mempengaruhi penggunaan game *gacha*
Rizky Ramadhan (2022)	Rancang Bangun Aplikasi Web Portal Game "Planet Game Id" Menggunakan Next.Js	*SDLC*	Pengembangan dan pengujian yang lebih baik	Web Portal Game "Planet Game Id"
Misbakhul Munir (2025)	Membangun Sistem *Gacha* pada Permainan *Virtual Dressing*	*MCRNG dan Xorshift*	Perhitungan cepat dan memori kecil	Permainan *Virtual Dressing* (V-Dress)

2.2 Dasar Teori

2.2.1 Virtual Dressing (V-Dress)

Virtual Dressing (V-Dress), merupakan permainan berbasis web dimana pemain diharuskan mendandani sebuah karakter dengan berbagai pakaian yang tersedia didalam permainan. V-Dress dibuat oleh Hazart Studio yang dilatarbelakangi oleh maraknya dunia virtual pada saat ini, mulai dari adanya VTuber atau Virtual Youtuber hingga dunia virtual seperti METAVERSE.

Hazart Studio bertujuan memberikan ruang bagi pemain untuk mengekspresikan diri secara virtual melalui permainan V-Dress. Dengan tema fashion sebagai inti permainan, pemain diberi kebebasan untuk menyesuaikan penampilan karakter mereka dengan berbagai opsi pakaian yang disediakan.

Pemain memiliki kesempatan untuk memperoleh berbagai macam pakaian melalui mekanisme *Gacha* yang tersedia dalam permainan Virtual Dressing (V-Dress). Pakaian di dalam permainan akan dikelompokkan ke dalam tiga kategori berdasarkan kualitasnya. Kategori pertama adalah untuk pakaian dengan kualitas rendah yang diberi label “R”, kategori kedua untuk pakaian dengan kualitas menengah yang diberi label “SR”, dan kategori ketiga untuk pakaian dengan kualitas tinggi yang diberi label “SSR”.

Pemain dalam permainan *Gacha* sering menyebut ketiga kategori tersebut sebagai tingkat kelangkaan (rarity). Semakin tinggi tingkat kelangkaan, semakin sulit bagi pemain untuk mendapatkannya.

2.2.2 *Gacha*

Gacha adalah mekanisme lotere untuk memenangkan item virtual yang dikembangkan di Jepang, sebagai elemen permainan peluang dalam mobile game yang digunakan untuk monetisasi dalam model bisnis freemium. Sistem *gacha* ini sendiri pada intinya akan mengacak setiap pembelian yang dilakukan oleh pemain. Jadi pemain disini tidak akan tahu barang apa yang akan mereka dapat ketika melakukan pembelian. Dengan sistem ini, pengembang game menjual apa yang dinamakan suatu “keberuntungan” kepada para pemain. (PM Mufti 2022).

Gacha adalah istilah dari bahasa Jepang, merujuk pada mekanisme di permainan video yang memungkinkan para pemain membayar untuk membuka "*gacha*" atau virtual box yang berisi item, karakter, atau pakaian dalam game. Sistem *gacha* ini sering beroperasi menggunakan mata uang dalam permainan atau mata uang premium yang dapat dibeli dengan uang sungguhan.

Saat seorang melakukan *gacha*, mereka berharap mereka dapat memenangkan berbagai barang langka, barang mewah kadang sangat sulit didapatkan oleh beberapa orang atau malah sering kali dengan mudah diraih tergantung seberapa sering pemain melakukan *gacha*. Mekanisme *gacha* dalam permainan video berbeda dari mekanisme *gacha* yang terdapat di wahana hiburan seperti mesin box atau bola keberuntungan. Jika *gacha* dalam sebuah mesin di wahana hiburan kita tidak tahu apa yang akan kita dapatkan dalam kotak atau bola keberuntungan, maka *gacha* dalam permainan video ini kita tahu persis item yang akan kita dapatkan, hanya saja kapan atau dalam berapa tarikan item itu akan muncul yang kita tidak tahu.

Salah satu kunci dalam mekanisme *gacha* adalah tingkat kelangkaan (rarity) item. Item langka tentu lebih sulit didapatkan dibandingkan item umum. Untuk meningkatkan peluang mendapatkan item langka, pengembang seringkali menerapkan sistem rate up. Dalam periode rate up, peluang untuk mendapatkan item tertentu akan dinaikkan untuk kurun waktu tertentu.

Konsep 50:50 juga sering ditemui dalam *gacha*, terutama saat pemain berhadapan dengan pilihan antara dua karakter atau item dengan tingkat kelangkaan yang sama. Sistem ini memberikan peluang 50% untuk mendapatkan salah satu dari dua pilihan tersebut.

Selain rate up, ada pula istilah rate on dan rate off. Rate on merupakan kondisi dimana seorang pemain akan dipastikan mendapatkan item yang sedang rate up, sedangkan Rate off merupakan kondisi dimana pemain harus bergelut dengan peluang 50:50 antara item rate up dan item standar.

Adapun fitur *pity* yang sering kali digunakan oleh beberapa sistem *gacha* untuk membantu player mendapatkan item langka yang diinginkan dalam beberapa tarikan tertentu yang telah ditetapkan oleh permainan tersebut.

Secara singkat sistem *gacha* ini merupakan inovasi dari sistem penjualan in-game konvensional berupa produk etalase menjadi sistem *gacha* yang memberikan pengalaman berbeda kepada para pemain dalam melakukan pembelian item di dalam permainan. Dalam sistem *gacha* pemain akan melakukan percobaan untuk melakukan tarikan *gacha* dengan menggunakan mata uang di dalam permainan, dan secara otomatis sistem akan memberikan item secara acak yang sudah disediakan sesuai tingkat kelangkaan.

2.2.3 Fitur *Pity*

Dalam konteks permainan daring dengan sistem *gacha*, istilah *pity* merujuk kepada mekanisme jaminan yang memastikan para pemain mendapatkan hadiah langka. Umumnya karakter atau item dengan tingkat kelangkaan tertinggi (misalnya bintang 5 atau SSR) akan pemain dapatkan setelah melakukan sejumlah percobaan *gacha* dalam jumlah tertentu.

Sistem *pity* ini hadir untuk mengatasi faktor keberuntungan yang dominan dalam sistem *gacha*, juga berguna untuk mencegah pemain mengalami frustasi akibat kegagalan berulang kali. Jumlah yang dibutuhkan untuk mencapai pity bervariasi di berbagai permainan dan biasanya tertera pada detail setiap *banner gacha*.

Terdapat juga konsep *"soft pity"* dan *"hard pity"*. *Soft pity* menandai peningkatan peluang mendapatkan hadiah langka beberapa pull sebelum mencapai angka *hard pity*, yang merupakan jaminan mutlak mendapatkan hadiah langka. Contohnya jika pemain menyentuh angka 70 *pity* maka persentase untuk mendapatkan item langka akan naik secara bertahap hingga ke batas *pity*. Setelah pemain menerima hadiah langka melalui sistem *pity*, hitungan akan direset kembali ke nol. Mekanisme ini penting dalam menjaga keseimbangan permainan dan memberikan pengalaman yang lebih adil bagi pemain, dengan memberikan kepastian meskipun faktor keberuntungan sedang tidak berpihak.

2.2.4 Probabilitas dalam *Gacha*

Dalam sistem *gacha* pada game online, probabilitas adalah kunci utama yang menentukan peluang pemain mendapatkan item atau karakter tertentu. Sistem

ini bekerja secara acak, di mana setiap pemain memiliki kesempatan yang sama untuk memperoleh hadiah yang berbeda. Probabilitas ini biasanya dinyatakan dalam persentase dan dipengaruhi oleh beberapa faktor, seperti tingkat kelangkaan item atau karakter, jenis *gacha* yang diikuti, dan event dalam game.

Item atau karakter yang lebih langka biasanya memiliki probabilitas yang lebih rendah untuk didapatkan. Misalnya, item legendaris mungkin hanya memiliki probabilitas 1%, yang berarti hanya ada 1 peluang dari 100 percobaan untuk mendapatkannya.

2.2.5 Uji A/B

Pengujian A/B atau juga dikenal sebagai pengujian A/B/n adalah metode membandingkan kinerja berbagai strategi melalui indikator objektif untuk memilih strategi terbaik.

Ketika pengujian A/B diterapkan pada produk internet, pada dasarnya pengujian ini merupakan layanan komprehensif untuk melakukan eksperimen online berskala besar yang mengintegrasikan *front-end, back-end,* data, dan umpan balik pengguna. (Sheng, J.; Liu, H.; Wang, B. 2023).

Pada dasarnya pengujian A/B adalah metode untuk membandingkan dua buah sistem (A dan B) melihat dari variabel-variabel yang telah ditentukan. pengujian A/B dapat dilakukan untuk menentukan halaman web, tampilan iklan, tampilan produk dan yang lainnya, pengujian itu bertujuan untuk melihat versi mana yang berkinerja lebih baik dalam mencapai tujuan tertentu.

2.2.6 *Javascript*

Javascript adalah sebuah bahasa script dinamis yang dapat dipakai untuk membangun interaktivitas pada halaman-halaman HTML statis. (Siahaan & Rismon, 2020).

Javascript adalah bahasa pemrograman yang biasa diletakkan bersama kode HTML untuk menentukan suatu tindakan. (Kadir & Triwahyuni, 2013:325).

Javascript adalah bahasa skrip (*Scripting Language*), yaitu kumpulan instruksi perintah yang digunakan untuk mengendalikan beberapa bagian dari sistem operasi. (Sibero, 2013:150).

2.2.7 *Framework*

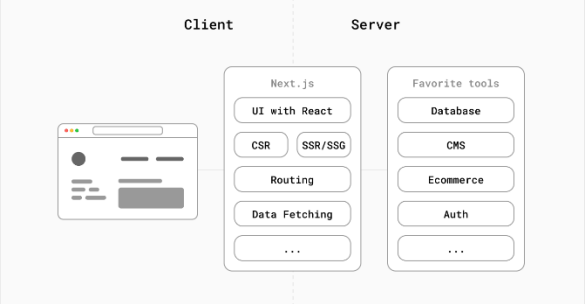
Framework adalah koleksi atau kumpulan potongan-potongan program yang disusun atau diorganisasikan sedemikian rupa, sehingga dapat digunakan untuk membantu membuat aplikasi utuh tanpa harus membuat kodenya dari awal. (Hakim, 2010:3).

Framework adalah suatu kumpulan kode berupa pustaka (*library*) dan alat (*tool*) yang dipadukan sedemikian rupa menjadi satu kerangka kerja (*framework*) guna memudahkan dan mempercepat proses pengembangan aplikasi web. (Raharjo, 2015:2).

2.2.8 *NextJS*

Dikutip dari laman *NextJS*.org, "*Next.js* adalah kerangka *React* yang memberi Anda blok penyusun untuk membuat aplikasi web.

Yang kami maksud dengan kerangka kerja adalah *Next.js* yang menangani perkakas dan konfigurasi yang diperlukan untuk *React*, dan menyediakan struktur, fitur, dan pengoptimalan tambahan untuk aplikasi Anda.”



Gambar *2. 1* Arsitektur *NextJS*

2.2.9 Uji Validitas

Uji validitas data dilakukan untuk mengetahui valid atau tidaknya data yang digunakan dalam penelitian.

$rxy = (\Sigma xy - \Sigma x.\Sigma y) / \sqrt{(n\Sigma x^2 - \Sigma x)^2(n\Sigma y^2 - \Sigma y)^2}$

Keterangan:

rxy = koefisien korelasi antara X dan Y

n = jumlah sampel

$\Sigma \bar{X}\bar{Y}$ = jumlah data XY

ΣX = jumlah data variable X

ΣY = jumlah data variabel Y

2.2.10 Uji Reliabilitas

Uji reliabilitas dilakukan untuk menentukan reliabilitas suatu data dalam sebuah penelitian.

$\alpha = (N / (N-1)) * (1 - (\Sigma Vi / \Sigma Vt))$

Keterangan:

α = Koefisien Cronbach's Alpha

N = Jumlah item dalam skala

ΣVi = Jumlah varians setiap item

ΣVt = Total varians skala

2.2.11 Random Number Generator (RNG)

Random Number Generator (RNG) adalah sebuah program atau alat untuk menghasilkan urutan angka atau simbol secara tidak teratur. (T. R. Hidayat, 2010). Beberapa contoh metode dalam pembangkit bilangan acak diantaranya adalah:

a. Metode fisik

Metode paling pertama untuk menghasilkan angka secara acak adalah menggunakan dadu, koin, rolet, dan lain sebagainya. Sampai saat ini, metode ini masih cukup sering digunakan, terutama di dalam game perjudian. Karena metode ini dianggap terlalu lambat, pengaplikasiannya untuk statistika dan kriptografi kurang begitu popular untuk saat ini.

Dasar dari metode fisik adalah fenomena fisika atomic atau subatomic acak yang tidak dapat diprediksi dapat dilacak dengan menggunakan mekanika kuantum.

b. Metode distribusi probabilitas

Metode ini menggunakan fungsi densitas probabilitas. Metode ini bekerja cukup baik untuk menghasilkan *pseudo-random* dan *true random number*. Salah satu metode, yaitu metode inverse, mengintegralkan are lebih dari sama dengan bilangan acak.

Metode kedua, *acceptance-rejection,* memilih antara nilai x dan y, lalu membandingkan apakah fungsi x lebih dari nilai y. apabila fungsi x lebih dari nilai y, maka nilai x akan diterima. Jika sebaliknya, makan nilai x akan ditolak dan algoritmanya akan mencoba ulang.

c. Metode komputasi

Metode ini menggunakan algoritma bernama *Pseudo-random number generator* yang secara otomatis menghasilkan serangkaian angka acak yang memiliki kualitas baik. Nilai yang dihasilkan oleh algoritma tersebut secara umum ditentukan dengan sebuah konstanta yang disebut *seed*. Salah satu *PRNG* yang umum adalah *Linear Congruential Generator*.

MCRNG (Multiplicative Congruential Random Number Generator) adalah metode untuk menghasilkan angka acak melalui perhitungan matematika. Prinsipnya adalah dengan menggunakan sebuah angka awal ("bibit"), yang kemudian dikalikan dengan angka lain ("pengali"), hasilnya dibagi dengan angka ketiga ("modulus"), dan sisa pembagiannya menjadi angka acak pertama.

Hasil angka acak pertama lalu digunakan kembali sebagai bibit untuk perhitungan selanjutnya, menghasilkan urutan angka acak. Meskipun mudah diimplementasikan dan hemat memori, *MCRNG* memiliki kekurangan, yaitu periodenya yang terbatas (urutan angka akan berulang) dan distribusinya yang kurang merata. Oleh karena itu, dibutuhkan dukungan atau kolaborasi dengan metode lainnya misalkan metode *Xorshift*.

2.2.12 Multiplicative Congruential Random Number Generator

MCRNG, atau Pembangkit Bilangan Acak Kongruen Perkalian, adalah salah satu algoritma yang umum digunakan untuk menghasilkan bilangan acak semu (*pseudorandom*). Algoritma ini termasuk dalam keluarga *Linear Congruential Generator (LCG),* tetapi dengan sedikit perbedaan utama.

Perbedaan Utama dengan LCG Biasa:

Dalam LCG biasa, rumus yang digunakan adalah:

$X(i+1) = (a * Xi + c) \bmod m$

di mana c adalah increment. Pada MCRNG, nilai c selalu 0. Sehingga rumusnya menjadi:

$X(i+1) = (a * Xi) \bmod m$

Keunggulan MCRNG:

a. Kecepatan: Karena tidak ada operasi penambahan (c), MCRNG cenderung lebih cepat daripada LCG biasa.
b. Penggunaan Memori: MCRNG hanya memerlukan sedikit memori untuk menyimpan nilai sebelumnya (Xi).

Keterbatasan:

a. Periode Pendek: MCRNG dapat memiliki periode pengulangan yang lebih pendek dibandingkan dengan LCG biasa, terutama jika parameter a dan m tidak dipilih dengan hati-hati. Ini berarti urutan bilangan acak yang dihasilkan akan berulang setelah beberapa iterasi.

b. Korelasi: Bilangan acak yang dihasilkan oleh MCRNG mungkin memiliki korelasi tertentu, yang dapat mempengaruhi hasil jika digunakan dalam simulasi atau aplikasi yang memerlukan keacakan yang tinggi.

2.2.13 *Xorshift*

Xorshift bekerja dengan serangkaian operasi *bitwise XOR (exclusive or)* dan *shift* (pergeseran bit). Algoritma ini menggunakan beberapa variabel internal yang disebut *state*. Setiap kali sebuah bilangan acak perlu dihasilkan, algoritma ini melakukan serangkaian operasi XOR dan *shift* pada *state*, dan kemudian mengembalikan salah satu nilai dari state yang telah dimodifikasi sebagai bilangan acak yang baru.

Sementara operasi *shift* memindahkan bit-bit di dalam sebuah kata komputer dengan sebuah *shift count*. Kedua operasi dalam kombinasi ini menghasilkan urutan bilangan pseudo-acak dengan periode yang sangat panjang dan distribusi yang cukup untuk banyak aplikasi. Implementasi Xorshift biasanya menunjukkan bahwa *state* terdiri dari satu atau lebih dari satu kata komputer. Misalnya, generasi dengan *state* khusus 32-bit dapat menghasilkan 2 ^ 32 – 1 bilangan acak sebelum terulang kembali.

x ^= x << a; x ^= x >> b; x ^= x << c;

Perkembangan dari *Xorshift* yang memiliki ukuran state lebih besar, seperti 64-bit atau lebih, dapat menghasilkan panjang yang jauh lebih lama. Marsaglia sendiri telah merekomendasikan xorshiro 256 * sebagai yang umum digunakan karena periode yang cukup lama dan distribusi outputnya yang cukup baik.

Xorshiro 256 * berbeda dari varian yang lain karena penambahan operasi rotasi pada algoritmanya.

Keunggulan *Xorshift*:

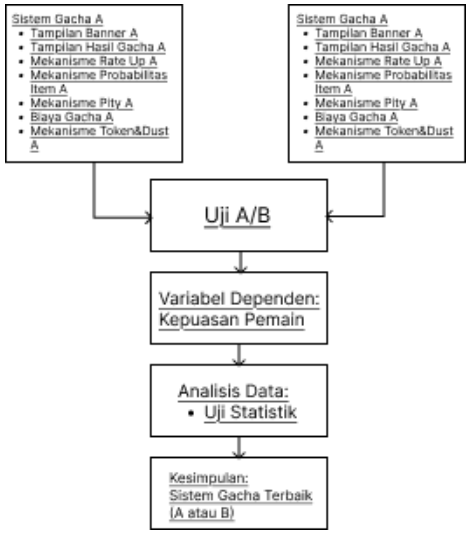
a. Kecepatan: *Xorshift* sangat cepat karena hanya menggunakan operasi *bitwise* yang efisien.
b. Sederhana: Kode implementasi *xorshift* relatif pendek dan mudah dipahami.
c. Kualitas: Meskipun sederhana, *xorshift* dapat menghasilkan urutan bilangan acak yang cukup baik untuk banyak aplikasi.

Kelemahan *Xorshift*:

a. Periode Pendek: Beberapa varian xorshift memiliki periode pengulangan yang relatif pendek, yang berarti urutan bilangan acak akan berulang setelah beberapa saat.
b. Ketergantungan pada *State* Awal: Jika state awal tidak dipilih dengan hati-hati, urutan bilangan acak yang dihasilkan mungkin tidak acak dengan baik.

2.2.14 Kerangka Berpikir

Kerangka berpikir penelitian ini disajikan pada Gambar 2.2.



Gambar 2.2 Kerangka Berfikir

Kerangka berpikir ini menggambarkan hubungan antara variabel independen, yaitu sistem *gacha* A dan B, dengan variabel dependen, yaitu tingkat kepuasan pemain

Perbedaan-perbedaan dalam desain sistem *gacha* ini diuji melalui metode uji A/B. Partisipan penelitian akan memainkan kedua sistem *gacha*. Setelah menggunakan sistem *gacha*, tingkat kepuasan partisipan akan diukur menggunakan kuesioner dengan skala likert 1 sampai 5. Data yang diperoleh kemudian dianalisis secara statistik menggunakan uji *Wilcoxon Signed-Rank* untuk menentukan apakah terdapat perbedaan yang signifikan dalam kepuasan partisipan antara kedua kelompok. Hasil analisis ini akan digunakan untuk menarik kesimpulan tentang sistem *gacha* mana yang dipilih untuk diimplementasikan ke dalam permainan web Virtual Dressing menurut tingkat kepuasan pengguna.

2.2.15 UUID *(Universally Unique Identifier)*

UUID *(Universally Unique Identifier)* adalah standar untuk membuat pengenal unik 128-bit. UUID digunakan secara luas dalam ilmu komputer untuk berbagai tujuan, karena menjamin bahwa identifikasi yang dihasilkan secara praktis unik. Ini berarti bahwa dua UUID yang berbeda hampir pasti mewakili dua entitas yang berbeda, bahkan jika dibuat di sistem yang berbeda atau pada waktu yang berbeda.

Karakteristik Utama UUID:

a. Unik: Kemungkinan dua UUID yang berbeda bernilai sama sangat kecil (dapat diabaikan). Inilah yang membuatnya sangat berguna untuk identifikasi.

b. Terdistribusi: UUID dapat dihasilkan tanpa memerlukan koordinasi terpusat. Ini penting untuk sistem terdistribusi di mana berbagai komponen mungkin perlu membuat pengenal unik secara independen.
c. Standar: UUID didefinisikan oleh standar RFC 4122, yang memastikan interoperabilitas antar sistem yang berbeda.
d. 128-bit: Ukurannya 128bit memberikan ruang yang sangat besar untuk kemungkinan UUID yang berbeda, membuatnya praktis unik.

UUID biasanya direpresentasikan sebagai string 36 karakter, termasuk tanda hubung, dalam format berikut:

xxxxxxxx-xxxx-xxxx-xxxx-xxxxxxxxxxxx

di mana setiap 'x' adalah digit heksadesimal (0-9 dan A-F).

Ada beberapa versi UUID, masing-masing dengan cara pembuatan yang berbeda:

a. Versi 1 (*Time-based*): Didasarkan pada waktu dan alamat *MAC host*. Ini memiliki potensi kecil untuk bentrokan jika jam sistem tidak akurat atau alamat MAC duplikat ada.
b. Versi 2 (*DCE Security*): Mirip dengan versi 1, tetapi menggunakan domain keamanan DCE. Kurang umum digunakan.
c. Versi 3 *(Name-based (MD5))*: Dihasilkan dengan hashing nama dan namespace menggunakan algoritma MD5. *Deterministic*, artinya nama dan namespace yang sama akan selalu menghasilkan UUID yang sama.
d. Versi 4 (*Random*): Dihasilkan sepenuhnya secara acak. Ini adalah jenis UUID yang paling umum digunakan karena kesederhanaan dan jaminan keunikannya.

e. Versi 5 *(Name-based (SHA-1))*: Mirip dengan versi 3, tetapi menggunakan algoritma SHA-1.

UUID digunakan dalam berbagai aplikasi, termasuk:

a. *Database*: Sebagai kunci utama untuk tabel.

b. Sistem Terdistribusi: Untuk mengidentifikasi objek, pesan, atau transaksi.

c. Perangkat Lunak: Sebagai pengenal unik untuk objek atau komponen.

d. Web: Sebagai pengenal unik untuk sumber daya atau sesi.

Penggunaan di *NextJS:*

```
import { v4 as uuidv4 } from 'uuid';
function MyComponent() {
  const generateUUID = () => {
    const newUUID = uuidv4();
    console.log(newUUID);
    return newUUID;
  };
  return (
    <div>
      <button onClick={generateUUID}>Generate UUID</button>
    </div>
  );
}
export default MyComponent;
```

Gambar 2.3 Penggunaan *UUID* dalam *NextJS*

2.2.16 *Convenience Sampling*

Sugiyono (2017) menyatakan bahwa *convenience sampling* adalah teknik penentuan sampel berdasarkan kebetulan, yaitu siapa saja yang secara kebetulan

bertemu dengan peneliti dapat digunakan sebagai sampel, bila dipandang oleh peneliti orang yang kebetulan ditemui itu cocok sebagai sumber data.

Karakteristik Utama

a. Kemudahan Akses: Faktor utama dalam pemilihan peserta adalah seberapa mudah peneliti dapat menjangkau mereka.
b. Non-Probabilitas: Karena pemilihan peserta tidak acak, setiap anggota populasi tidak memiliki peluang yang sama untuk terpilih.
c. Cepat dan Murah: *Convenience sampling* seringkali merupakan metode pengambilan sampel yang paling cepat dan paling murah.

Kelebihan:

a. Praktis: Metode ini sangat praktis dan mudah diimplementasikan.
b. Cepat: Pengumpulan data dapat dilakukan dengan cepat karena peneliti tidak perlu melakukan proses seleksi yang rumit.
c. Biaya Terjangkau: Biaya yang dikeluarkan untuk pengambilan sampel relatif rendah.

Kekurangan:

a. Bias Seleksi: Sampel mungkin tidak representatif dari populasi yang lebih besar, karena peserta dipilih berdasarkan ketersediaan dan kemudahan.
b. Generalisasi Terbatas: Sulit untuk menggeneralisasikan temuan penelitian ke populasi yang lebih luas karena sampel tidak representatif.

Contoh Penggunaan:

Contoh penggunaan convenience sampling dalam penelitian tentang preferensi konsumen terhadap suatu produk. Peneliti dapat mewawancarai konsumen yang kebetulan ditemui di pusat perbelanjaan atau pasar.

2.2.17 Uji non-parametik *Chi-Square*

Uji *Chi-square* adalah uji statistik non-parametrik yang digunakan untuk menganalisis data kategorik. Data kategorik adalah data yang berupa kategori atau kelompok, seperti jenis kelamin, warna, atau preferensi. Uji *Chi-square* bekerja dengan membandingkan frekuensi observasi dengan frekuensi harapan. Frekuensi observasi adalah frekuensi yang sebenarnya terjadi dalam data, sedangkan frekuensi harapan adalah frekuensi yang diharapkan jika tidak ada hubungan antara variabel.

Rumus Uji *Chi-square*

$\chi^2 = \Sigma [(O - E)^2 / E]$

di mana:

χ^2 adalah nilai *Chi-square*

O adalah frekuensi observasi

E adalah frekuensi harapan

Uji *Chi-square* dapat digunakan untuk menguji berbagai macam hipotesis, antara lain:

a. Uji Kesesuaian (*Goodness of Fit Test*): Uji ini digunakan untuk menguji apakah distribusi data observasi sesuai dengan distribusi teoritis yang diharapkan.
b. Uji Independensi (*Test of Independence*): Uji ini digunakan untuk menguji apakah dua variabel kategorik saling bebas atau tidak.

c. Uji Homogenitas (*Test of Homogeneity*): Uji ini digunakan untuk menguji apakah beberapa populasi memiliki proporsi yang sama untuk variabel kategorik tertentu.

Syarat Penggunaan Uji *Chi-square*:

a. Data harus berupa data kategorik (nominal atau ordinal).
b. Data harus berupa data frekuensi atau data cacahan.
c. Tidak ada sel dengan frekuensi harapan kurang dari 5.
d. Ukuran sampel harus cukup besar.

Cara Melakukan Uji *Chi-square*:

a. Menentukan Hipotesis: Tentukan hipotesis nol (H0) dan hipotesis alternatif (H1) yang akan diuji.
b. Menyusun Tabel Kontingensi: Sajikan data dalam bentuk tabel kontingensi yang berisi frekuensi observasi dan frekuensi harapan.
c. Menghitung Nilai *Chi-square*: Hitung nilai *Chi-square* menggunakan rumus
d. Menentukan Derajat Kebebasan (df): Hitung derajat kebebasan menggunakan rumus: df = (jumlah baris - 1) x (jumlah kolom - 1)
e. Menentukan Tingkat Signifikansi (α): Pilih tingkat signifikansi yang akan digunakan (biasanya 0,05).
f. Mencari Nilai *Chi-square* Tabel: Cari nilai *Chi-square* tabel pada tabel distribusi *Chi-square* dengan derajat kebebasan dan tingkat signifikansi yang telah ditentukan.

g. Membandingkan Nilai *Chi-square* Hitung dengan *Chi-square* Tabel: Jika nilai *Chi-square* hitung lebih besar dari nilai *Chi-square* tabel, maka hipotesis nol ditolak.
h. Menarik Kesimpulan: Tarik kesimpulan berdasarkan hasil perbandingan nilai *Chi-square*.