BAB II
TINJAUAN PUSTAKA

2.1 Landasan Teori

2.1.1. Sistem Pemesanan Berbasis Web

Sistem pemesanan berbasis web merupakan aplikasi yang memungkinkan pengguna melakukan proses reservasi atau pemesanan layanan melalui media internet secara daring[17]. Sistem ini dirancang untuk memberikan kemudahan akses, efisiensi waktu, serta fleksibilitas bagi pengguna dalam melakukan pemesanan tanpa harus datang langsung ke lokasi[5]. Dalam konteks layanan hiburan seperti billiard, sistem pemesanan berbasis web berfungsi untuk menampilkan informasi ketersediaan meja, jadwal penggunaan, serta memfasilitasi proses *booking* secara real-time[18].

Penerapan sistem pemesanan berbasis web juga memungkinkan pengelolaan data pemesanan, pengguna, dan jadwal secara konsisten, sehingga mengurangi kesalahan pencatatan dan memudahkan pembuatan laporan operasional[19]. Namun, sistem yang berjalan secara *real-time* dan diakses oleh banyak pengguna secara bersamaan memiliki tantangan tersendiri, terutama terkait konsistensi data. Tanpa mekanisme pengendalian yang tepat, kondisi tersebut berpotensi menimbulkan konflik pemesanan yang dapat menurunkan kualitas layanan[20].

2.1.2. Concurrency Control

Concurrency control merupakan mekanisme yang digunakan dalam sistem basis data dan sistem terdistribusi untuk mengatur akses terhadap data ketika terdapat lebih dari satu proses atau pengguna yang melakukan operasi secara bersamaan[12]. Tujuan utama dari *concurrency control* adalah menjaga konsistensi, integritas, dan keandalan data agar tetap valid meskipun terjadi transaksi simultan pada *resource* yang sama[21].

Dalam lingkungan sistem *multi-user*, *concurrency control* berperan penting untuk mencegah terjadinya konflik transaksi seperti *lost update, dirty read*, maupun *race condition*[13]. Untuk mengatasi permasalahan tersebut, Beberapa pendekatan *concurrency control* yang umum digunakan antara lain *pessimistic locking* dan *optimistic concurrency control*. Pemilihan pendekatan tersebut disesuaikan dengan karakteristik sistem, tingkat konkurensi pengguna, serta kebutuhan performa aplikasi[22].

2.1.3. Race Condition

Race condition merupakan kondisi ketika dua atau lebih proses mengakses dan memodifikasi data yang sama secara bersamaan tanpa adanya mekanisme sinkronisasi yang memadai. Akibat dari kondisi ini, hasil akhir proses menjadi tidak dapat diprediksi dan sangat bergantung pada urutan eksekusi masing-masing proses[23].

Pada sistem pemesanan berbasis web, *race condition* sering terjadi ketika beberapa pengguna mencoba melakukan pemesanan terhadap resource yang sama dalam waktu yang hampir bersamaan. Pada sistem pemesanan meja billiard, *race condition* dapat menyebabkan terjadinya *double booking*, yaitu satu meja dan satu waktu pemesanan tercatat lebih dari satu kali[10]. Kondisi tersebut tidak hanya mengakibatkan inkonsistensi data, tetapi juga dapat menurunkan tingkat kepuasan pengguna serta mengganggu performa keaandalan sistem[24].

2.1.4. Firebase Realtime Database

Firebase Realtime Database merupakan layanan basis data NoSQL berbasis cloud yang dirancang untuk mendukung sinkronisasi data secara real-time[25]. Data disimpan dalam format JSON dan dapat diakses secara simultan oleh banyak klien melalui koneksi internet. Setiap perubahan data yang terjadi akan langsung disinkronkan ke seluruh klien yang terhubung, sehingga sistem dapat menampilkan data terbaru tanpa memerlukan proses *refresh* manual [26].

Meskipun memiliki keunggulan dalam hal kecepatan dan sinkronisasi *real-time*, *Firebase Realtime Database* menerapkan prinsip *eventual consistency*.

Sehingga tanpa mekanisme transaksi yang tepat, sistem berpotensi mengalami konflik data atau race condition, ketika terjadi akses data secara bersamaan. Oleh karena itu, penggunaan fitur firebase transaction menjadi penting untuk menjaga konsistensi data pada sistem multi-user[15].

2.1.5. Firebase Transaction

Firebase Transaction merupakan fitur pada *Firebase Realtime Database* yang memungkinkan proses baca dan tulis data dilakukan secara atomik. Melalui mekanisme ini, proses baca dan tulis data dilakukan dalam satu siklus transaksi yang tidak terpisahkan, di mana sistem akan membaca nilai data terbaru, kemudian melakukan validasi ulang sebelum proses penulisan data dilakukan. Apabila selama proses transaksi terdeteksi adanya perubahan data oleh transaksi lain, maka *Firebase* akan menjalankan mekanisme *retry* atau membatalkan transaksi tersebut[14].

Fungsi *runTransaction()* pada *Firebase* digunakan untuk mengimplementasikan mekanisme tersebut ke dalam sistem. Dengan membungkus seluruh proses baca dan tulis data ke dalam satu siklus transaksi atomik, sistem dapat memastikan bahwa hanya satu transaksi yang berhasil menuliskan data pada *resource* tertentu dalam satu waktu. Penerapan mekanisme ini efektif dalam mengatasi permasalahan *race condition* pada sistem pemesanan berbasis web yang berjalan secara *real-time* dan diakses oleh banyak pengguna secara simultan[27].

2.2 Penelitian Terdahulu

Beberapa penelitian sebelumnya telah membahas pengembangan sistem pemesanan berbasis web pada berbagai bidang layanan, seperti pemesanan fasilitas olahraga, reservasi hotel, dan sistem reservasi layanan lainnya. Sebagian penelitian menerapkan mekanisme validasi sisi server atau *locking* sederhana untuk mencegah konflik pemesanan[13]. Namun, pendekatan tersebut belum sepenuhnya mampu menjamin atomisitas transaksi pada lingkungan multi-user yang berjalan secara real-time.

Penelitian lain yang memanfaatkan *Firebase Realtime Database* umumnya berfokus pada aspek sinkronisasi data dan implementasi operasi operasi *Create, Read, Update, Delete* (CRUD), tanpa membahas secara mendalam risiko *race condition* pada skenario transaksi simultan[28]. Selain itu, studi yang secara spesifik mengevaluasi efektivitas *Firebase Transaction* dalam mencegah *double booking* masih relatif terbatas. Oleh karena itu, penelitian ini memiliki perbedaan dengan penelitian terdahulu, yaitu dengan mengimplementasikan dan menguji mekanisme *concurrency control* menggunakan *Firebase Transaction* melalui simulasi *pre-test* dan *post-test* dengan variasi jumlah pengguna simultan.

2.3 Kerangka Pemikiran

Kerangka pemikiran dalam penelitian ini disusun berdasarkan permasalahan yang ditemukan pada sistem pemesanan meja billiard berbasis web, yaitu terjadinya *race condition* yang menyebabkan *double booking* akibat tidak adanya mekanisme *concurrency control* yang memadai. Permasalahan tersebut muncul ketika beberapa pengguna melakukan pemesanan terhadap meja dan waktu yang sama secara bersamaan.

Untuk mengatasi permasalahan tersebut, penelitian ini menerapkan mekanisme *concurrency control* ke dalam sistem menggunakan *Firebase Transaction*. Mekanisme ini memastikan bahwa proses baca dan tulis data dilakukan secara atomik sehingga hanya satu transaksi yang dapat berhasil pada satu *resource* dalam satu waktu. Dengan penerapan mekanisme tersebut, sistem diharapkan mampu menjaga konsistensi data, mencegah terjadinya *race condition* yang mengakibatkan *double booking*, serta meningkatkan keandalan sistem pemesanan dalam lingkungan *multi-user real-time*.