**Identifikasi Test Case untuk Fitur Pemesanan Makanan**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **No.** | **Test Case** | **Tujuan Pengujian** | **Langkah - langkah uji coba** | **Data yang diperlukan** | **Hasil yang diharapkan** |
| 1. | Login Berhasil | Memastikan pengguna dapat login ke aplikasi dengan kredensial yang valid. | 1. Buka aplikasi FoodFast. 2. Masukkan email dan password yang valid. 3. Klik tombol "Login." | * Email: user@example.com * Password: password123 | Pengguna berhasil login dan diarahkan ke halaman utama. |
| 2. | Pencarian Restoran | Memastikan fitur pencarian dapat menampilkan hasil yang relevan. | 1. Login ke aplikasi. 2. Masukkan nama restoran atau jenis makanan pada kolom pencarian. 3. Klik tombol "Cari." | Kata kunci: "Burger" | Restoran yang sesuai dengan kata kunci muncul dalam hasil pencarian. |
| 3. | Menambahkan Makanan ke Keranjang | Memastikan pengguna dapat menambahkan makanan ke keranjang belanja. | Login dan cari restoran.  Pilih makanan dari daftar menu.  Klik tombol "Tambah ke Keranjang." | Nama makanan: "Cheese Burger" | Makanan berhasil ditambahkan ke keranjang dengan jumlah yang sesuai. |
| 4. | Proses Checkout | Memastikan pengguna dapat menyelesaikan proses checkout. | 1. Buka keranjang belanja. 2. Klik tombol "Checkout." 3. Pilih metode pembayaran (e-wallet). 4. Konfirmasi pesanan. | Metode pembayaran: e-wallet (contoh: Dana) | Pengguna diarahkan ke halaman konfirmasi pembayaran. |
| 5. | Notifikasi Pesanan Berhasil | Memastikan pengguna menerima notifikasi setelah pesanan berhasil dibuat. | 1. Selesaikan proses checkout. 2. Periksa notifikasi di aplikasi. | Pesanan aktif. | Notifikasi "Pesanan berhasil dibuat" muncul di layar. |
| 6. | Validasi Jumlah Item di Keranjang | Memastikan jumlah item di keranjang sesuai dengan input pengguna. | 1. Tambahkan dua makanan ke keranjang. 2. Periksa jumlah item di keranjang. | Dua makanan berbeda, atau makanan yang sama berjumlah 2 item atau lebih. | Jumlah item di keranjang sesuai dengan jumlah yang ditambahkan. |
| 7. | Validasi Input Metode Pembayaran | Memastikan aplikasi memvalidasi input metode pembayaran dengan benar. | 1. Login ke aplikasi. 2. Tambahkan makanan ke keranjang. 3. Pilih metode pembayaran tanpa memasukkan informasi yang valid (contoh: nomor e-wallet kosong). 4. Klik tombol "Konfirmasi Pembayaran." | Metode pembayaran: E-wallet tanpa nomor valid. | Sistem menampilkan pesan error seperti "Metode pembayaran tidak valid." |
| 8. | alidasi Ketersediaan Menu | Memastikan aplikasi menampilkan pesan jika makanan yang dipilih habis. | 1. Login ke aplikasi. 2. Cari restoran dan pilih makanan yang stoknya kosong. 3. Tambahkan makanan tersebut ke keranjang. | Makanan yang stoknya kosong: "Cheese Burger." | Sistem menampilkan pesan "Menu ini saat ini tidak tersedia." |
| 9. | Membatalkan Pesanan Sebelum Checkout | Memastikan pengguna dapat membatalkan pesanan di keranjang sebelum menyelesaikan checkout. | 1. Login ke aplikasi. 2. Tambahkan makanan ke keranjang. 3. Buka keranjang dan klik tombol "Hapus" pada item tertentu. | Makanan di keranjang: "Nasi Goreng." | Item yang dipilih berhasil dihapus dari keranjang. |
| 10. | Uji Ketahanan Sistem untuk Pesanan Massal | Memastikan aplikasi mampu menangani banyak pesanan dalam waktu bersamaan. | 1. Simulasikan 100 pengguna memesan makanan dari restoran yang sama secara bersamaan. 2. Periksa respons aplikasi dan waktu pemrosesan. | Restoran yang sama untuk semua pengguna. | Aplikasi tetap berjalan tanpa error, dan semua pesanan berhasil diproses. |

**Analisis Potensi Bug**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No. | Bug | Penyebab Bug | Dampak | Saran Perbaikan |
| 1. | Login Gagal | Masalah pada validasi kredensial atau koneksi server. | Pengguna tidak dapat mengakses aplikasi. | Tambahkan pesan error yang jelas dan pastikan server login stabil. |
| 2. | Pencarian Tidak Menampilkan Hasil | Kata kunci pencarian tidak terindeks atau adanya error pada backend. | Pengguna kesulitan menemukan restoran. | Perbaiki algoritma pencarian dan pastikan data restoran terindeks dengan benar. |
| 3. | Keranjang Tidak Mengupdate Item | Error pada proses penyimpanan item ke keranjang di sisi frontend atau backend. | Pengguna tidak bisa melanjutkan ke checkout. | Lakukan pengujian mendalam pada API yang menangani keranjang dan pastikan data sinkron. |
| 4. | Stok Makanan Tidak Terupdate | * Sinkronisasi data antara aplikasi dan database restoran yang tidak berjalan dengan baik. * Proses pembaruan stok berjalan lambat atau gagal pada saat pesanan dibuat. | Pengguna bisa memesan makanan yang sebenarnya sudah habis, dan ini akan menyebabkan ketidakpuasan pelanggan. | Implementasikan sistem pembaruan stok secara real-time.  Tambahkan validasi stok saat pengguna menekan tombol "Checkout." |
| 5. | Kesalahan pada Kalkulasi Total Harga | * Kesalahan dalam algoritma penghitungan total harga, seperti diskon atau pajak yang tidak diaplikasikan dengan benar. * Data menu atau harga makanan tidak diperbarui. | Pengguna mungkin membayar lebih atau kurang dari harga yang seharusnya. | * Uji integritas data harga pada database secara berkala. * Tambahkan unit test untuk semua skenario kalkulasi harga, termasuk diskon dan pajak. |
| 6. | Error pada Jaringan Lemah | Aplikasi tidak menangani situasi jaringan lambat atau terputus dengan baik. | Proses pemesanan terhenti atau data pesanan tidak tersimpan, menyebabkan frustrasi pengguna. | * Implementasikan fitur offline mode yang memungkinkan pengguna menyelesaikan pesanan saat jaringan kembali tersedia. * Tambahkan indikator progres dan pesan informasi saat jaringan lambat. |
| 7. | Duplikasi Pesanan | Pengguna tidak menerima notifikasi segera setelah memesan, lalu mencoba memesan ulang.  Sistem backend tidak memvalidasi ID pesanan dengan benar sebelum menyimpan. | Terjadi penggandaan pesanan, yang dapat membebani restoran dan mengurangi pengalaman pengguna. | * Tambahkan mekanisme pencegahan duplikasi pesanan di backend dengan memeriksa ID pesanan sebelum menyimpannya. * Kirim notifikasi instan meskipun proses backend belum selesai sepenuhnya. |