

## Gambaran Umum Aplikasi

Aplikasi merupakan sistem berbasis web yang digunakan untuk pemesanan tiket event secara online oleh customer, serta pengelolaan transaksi dan data peserta oleh admin. Sistem melibatkan integrasi dengan metode pembayaran QRIS serta pengiriman e-tiket melalui email dan WhatsApp.

## Key Requirements & Functional Mapping Assumptions

Berdasarkan analisis ruang lingkup aplikasi, berikut adalah pemetaan kebutuhan fungsional (customer journey & core transaction flow) serta beberapa asumsi teknis yang akan menjadi dasar strategi pengujian:

Aplikasi adalah web-based ticketing system untuk event, dengan user role Customer (end-user) dan Admin.

- **Customer Flow:** Mulai dari
  - *landing page website* →
  - lihat *detail event* →
  - pilih *kategori tiket* (contoh: VIP/Regular) →
  - *pesan tiket* (mengisi data seperti nama, email, dan No. HP) →
  - Dialihkan ke halaman *pembayaran tiket* →
  - menggunakan metode *QRIS* menerima →
  - *e-tiket* melalui *email* dan *WhatsApp* →
  - status transaksi pada tiket berubah menjadi **Paid**, 1 tiket untuk 1 orang dan 1 kali penggunaan (scan).
- **Admin Flow:** Login dengan credential →
- Masuk ke **Halaman Login Admin** →
- lihat *riwayat transaksi* →
- lihat *data peserta terdaftar* (detail customer berdasarkan event).

**Integrasi eksternal:** Implementasi pembayaran melalui payment gateway yang menggunakan metode QRIS yang terintegrasi secara real-time (contoh: Midtrans), email service (SMTP), dan WhatsApp API untuk pengiriman e-tiket secara otomatis.

**Requirement umum:** Responsive mobile/desktop, keamanan data, handle concurrency (multiple bookings), validasi input (contoh: email valid, awalan no telepon +62 / 08), callback pembayaran sukses, gagal, terlambat, sistem harus bisa handle callback yang masuk lebih dari 1 kali (duplicate callback), datang setelah user menutup halaman pembayaran.

## Beberapa potensi risiko utama (*main risks*) pada aplikasi beserta *mitigation strategy*

Berdasarkan arsitektur aplikasi, risiko utama dikategorikan sebagai berikut:

1. **Financial integrity:** Gagal *payment* dan/atau *double booking* bisa menyebabkan kehilangan *revenue* atau kondisi *overbooking* event, Tiket terjual lebih dari kuota. Hal ini berkaitan langsung dengan Access Validation, dimana sistem harus menjamin dengan menerapkan pengecekan/*preventive* bahwa hanya transaksi dengan status "Paid" yang berhak menerbitkan tiket untuk mencegah kuota terjual melebihi kapasitas.
2. **Security & data privacy:** Kebocoran data pribadi customer (email, no. HP) via email/WA, atau vulnerability di QRIS callback, data transaksi yang tidak sinkron antara customer dan admin.
3. **Operational & service monitoring:** Delay kirim e-tiket yang disebabkan oleh gangguan pada *service* dapat membuat *customer frustrasi*, terutama jika event real-time, callback pembayaran tidak diterima atau diproses dengan benar, status yang tidak konsisten ('Paid' di payment gateway, 'Pending' di sistem internal), pengiriman e-tiket gagal dikirim ke email/wa karena service down.

4. **Scalability & fraud:** Kalau event populer, lonjakan *traffic* dapat mengakibatkan *crash* pada *server* dan *performance* yang lambat, terdapat risiko *reusable* QR Code yang dapat dipindai berkali-kali dapat menyebabkan kerugian (*fraud*), maka sistem wajib memastikan validasi ketat agar satu tiket hanya berlaku untuk satu kali pemindaian (*single-use scan*)
5. **Compliance & consistency:** Alur refund/cancel yang tidak ter-*handle* dengan benar, diperlukan fitur riwayat transaksi yang akurat untuk mendukung kepatuhan operasional dan audit.

#### Area yang paling rawan bug:

- **Payment integration (QRIS):** *Callback* gagal mengupdate status, mismatch amount atau timeout scan QR, user bayar tapi *callback* terlambat, *timeout request*, *payload callback* yang tidak lengkap / *valid*.
- **Data consistency:** Transaksi sukses tapi e-tiket tidak terkirim, atau data peserta di *admin view* tidak *sync* dengan *customer view*, *Refresh* halaman setelah QR muncul.
- **User input validation:** Minim/tidak adanya penjaan/validasi (*missing validation*) pada seluruh *entry point* termasuk penanganan karakter khusus, validasi format nomor telepon (+62/08), serta konsistensi format tanggal dan angka. Field seperti email invalid, jumlah ticket exceeding stock, atau special characters membuat crash formulir.
- **Notification system:** Email/WA gagal kirim karena API error, salah query select, atau konten e-tiket salah (contoh: salah event), mengirim ke email / WA yang salah, e-tiket terkirim lebih dari 1 kali, e-tiket yang terkirim sebelum pembayaran benar-benar selesai.
- **Admin dashboard:** Filter riwayat transaksi error, atau unauthorized access jika session management buruk, status paid di *back-end* tapi di *front-end* masih "pending", dapat disebabkan oleh cache / polling status tidak berjalan.
- **Compatibility:** Versi OS yang tidak mendukung (*outdated*), device dengan low RAM, atau arsitektur CPU tertentu tidak ter-*handle* (crash pada saat splash screen).
- **Session/Token:** Token lama tidak valid tapi alur *refresh token* atau *re-login* tidak ter-*handle* dengan baik sehingga user stuck di loading, *Handling case* user menutup halaman sebelum pembayaran selesai.
- **Rate limiting / overload (related to traffic management & system resilience):** Jika banyak user serentak melakukan akses, aplikasi mengalami *stuck loading* dan tidak memberikan respons error yang informatif ketika *limit request* API tercapai.
- **Inconsistent event data:** Event yang sudah *sold out* atau *expired* di *back-end* namun masih ditampilkan di *front-end/UI* saat diakses atau di klik baru muncul *error*.
- **Seat / quota race condition:** Let say, misal ada dua user ambil seat/kuota yang sama, usernya bisa memesan namun payment kemudian dibatalkan atau order jadi "Ghost Order" (kuota terpesan tapi tidak ada pembayaran) atau "Double Payment" yang disebabkan oleh kondisi ketika dua user secara simultan mengakses sisa kuota terakhir -> perlu mekanisme *database locking*.
- **Format standardization:** Formatting tanggal/angka pada laporan transaksi dan detail e-tiket.
- **Financial logic & reconciliation, price calculation/formula inconsistency:** Tidak adanya sinkronisasi perhitungan antara *front-end* dan *back-end* terkait komponen biaya seperti diskon, kode promo, biaya admin, dan pajak untuk mencegah *mismatch* nominal pembayaran.
- **Failed ticket generation:** Order sudah paid tapi *job* yang melakukan *generate* QR/Barcode dan kirim email/whatsapp error atau antrean (*queue*) *job* macet.
- **Handling refund/protection:** Jika ada fitur refund, *logic* penentuan kapan tiket dibatalkan vs tetap aktif bisa keliru dan membingungkan user.

Berdasarkan analisis di atas, fokus utama pengujian QA berada pada validasi end-to-end flow pemesanan tiket, integrasi pembayaran QRIS, konsistensi status transaksi serta keandalan pengiriman e-tiket sebelum aplikasi dirilis ke production.

Test Case ID	Test Scenario	Type Test	Test Steps	Expected Result
TC-CUST-01	Customer berhasil memesan tiket dengan data valid	Positive	1. Buka landing page 2. Pilih event aktif 3. Pilih kategori tiket tersedia 4. Isi nama, email valid, No. HP valid 5. Klik tombol checkout	Sistem berhasil memproses pesanan dan menampilkan halaman pembayaran
TC-CUST-02	Customer melihat detail event dengan benar	Positive	1. Buka landing page 2. Klik salah satu event	Halaman detail event tampil dengan informasi <b>nama, tanggal, lokasi, harga</b> dan <b>kuota</b>
TC-CUST-03	Customer memilih lebih dari 1 tiket dalam satu transaksi	Positive	1. Pilih event 2. Pilih kategori tiket 3. Pilih jumlah tiket > 1 4. Lanjutkan checkout	Total harga dihitung sesuai jumlah tiket yang dipilih
TC-CUST-04	Customer checkout tanpa mengisi semua data wajib	Negative	1. Pilih event dan tiket 2. Kosongkan field email atau No. HP 3. Klik checkout	Sistem menampilkan validasi error pada field yang kosong
TC-CUST-05	Customer mengisi email dengan format tidak valid	Negative	1. Pilih event dan tiket 2. Isi email dengan format salah (reza@.com) 3. Klik checkout	Sistem menolak input dan menampilkan pesan error email tidak valid
TC-CUST-06	Customer mengisi No. HP dengan format tidak valid	Negative	1. Pilih event dan tiket 2. Isi No. HP tanpa awalan +62 / 08 3. Klik Checkout	Sistem menampilkan pesan error format No.HP tidak valid
TC-CUST-07	Customer mencoba memesan tiket melebihi kuota tersedia	Negative	1. Pilih event dengan kuota terbatas 2. Pilih jumlah tiket melebihi sisa kuota 3. Klik Checkout	Sistem menolak pesanan dan menampilkan pesan kuota tidak mencukupi
TC-CUST-08	Customer memesan tiket untuk event yang sudah sold out	Edge	1. Akses event sold out dari URL 2. Pilih tiket 3. Klik Checkout	Sistem menampilkan informasi event sold out dan tidak dapat melanjutkan checkout
TC-CUST-09	Customer refresh halaman setelah memilih tiket	Edge	1. Pilih event dan tiket 2. Refresh halaman browser	Data tiket tetap konsisten atau user diminta memilih ulang dengan jelas
TC-CUST-10	Customer klik tombol checkout lebih dari satu kali	Edge	1. Isi data valid 2. Klik tombol checkout berulang kali dengan cepat	Sistem hanya membuat satu transaksi dan mencegah duplicate order
TC-CUST-11	Customer menggunakan karakter spesial pada field nama	Edge	1. Isi nama dengan karakter spesial (contoh: @#\$%) 2. Lanjutkan checkout	Sistem menolak atau mensanitize input sesuai aturan
TC-CUST-12	Customer menutup browser sebelum checkout selesai	Edge	1. Isi data dan pilih tiket 2. Tutup browser sebelum klik checkout	Tidak ada transaksi yang tercatat atau status tetap aman (tidak ghost order)

Test Case ID	Test Scenario	Type Test	Test Steps	Expected Result
TC-PAY-01	Customer berhasil melakukan pembayaran QRIS	Positive	1. Selesaikan checkout tiket 2. Sistem menampilkan QR code pembayaran 3. Scan QR menggunakan e-wallet / mobile banking 4. Selesaikan pembayaran	Status transaksi berubah menjadi Paid dan sistem menampilkan konfirmasi pembayaran berhasil
TC-PAY-02	Customer membayar dengan nominal sesuai total transaksi	Positive	1. Tampilkan QR pembayaran 2. Lakukan pembayaran sesuai nominal	Sistem menerima pembayaran dan tidak terjadi error amount
TC-PAY-03	Customer tidak menyelesaikan pembayaran hingga waktu habis	Negative	1. Tampilkan QR pembayaran 2. Tunggu hingga melewati batas waktu pembayaran	Status transaksi berubah menjadi Expired / Failed
TC-PAY-04	Customer membayar dengan nominal yang tidak sesuai	Negative	1. Scan QR pembayaran 2. Ubah nominal pembayaran (jika memungkinkan di e-wallet)	Sistem menolak atau menandai transaksi sebagai gagal
TC-PAY-05	Callback pembayaran QRIS diterima lebih dari satu kali	Edge	1. Selesaikan pembayaran QRIS 2. Simulasikan callback duplicate dari payment gateway	Status transaksi tetap <b>Paid</b> dan tidak terjadi duplicate e-tiket
TC-PAY-06	Callback pembayaran QRIS terlambat diterima sistem	Edge	1. Customer menyelesaikan pembayaran 2. Callback masuk setelah beberapa menit	Status transaksi tetap diperbarui menjadi <b>Paid</b> tanpa error.
TC-PAY-07	Customer menutup halaman pembayaran sebelum callback diterima	Edge	1. Scan QR dan bayar 2. Tutup halaman browser sebelum redirect	Transaksi tetap diproses dan status diperbarui menjadi <b>Paid</b>
TC-PAY-08	Payment gateway mengirim callback dengan payload tidak lengkap	Negative	1. Simulasikan callback tanpa field wajib	Sistem menolak callback dan mencetak error dengan aman
TC-PAY-09	Payment gateway mengirim status pembayaran gagal	Negative	1. Scan QR 2. Batalkan pembayaran di e-wallet	Status transaksi menjadi <b>Failed</b> dan user mendapat notifikasi
TC-PAY-10	Customer refresh halaman pembayaran setelah QR ditampilkan	Edge	1. Tampilkan QR pembayaran 2. Refresh browser	QR tetap valid atau sistem menampilkan status transaksi terbaru

Test Case ID	Test Scenario	Type Test	Test Steps	Expected Result
TC-ADM-01	Admin dapat melihat daftar riwayat transaksi	Positive	1. Login sebagai admin 2. Akses menu Riwayat Transaksi	Daftar transaksi tampil dengan informasi dasar (ID, event, status, tanggal, total)
TC-ADM-02	Admin melihat detail transaksi dengan status Paid	Positive	1. Buka Riwayat Transaksi 2. Klik salah satu transaksi Paid	Detail transaksi tampil lengkap dan sesuai dengan data customer
TC-ADM-03	Admin memfilter transaksi berdasarkan status	Positive	1. Akses Riwayat Transaksi 2. Pilih filter status ( <b>Paid/Pending/Failed</b> )	Daftar transaksi tampil sesuai filter yang dipilih
TC-ADM-04	Admin melihat transaksi dengan status Pending	Edge	1. Buka Riwayat Transaksi 2. Pilih transaksi <b>Pending</b>	Informasi transaksi tetap dapat diakses dan status ditampilkan dengan benar
TC-ADM-05	Admin melihat transaksi yang gagal	Edge	1. Akses Riwayat Transaksi 2. Buka transaksi Failed	Transaksi ditandai sebagai Failed tanpa e-tiket terasosiasi
TC-ADM-06	Admin mengakses riwayat transaksi tanpa login	Negative	1. Akses URL riwayat transaksi langsung tanpa login	Sistem mengarahkan ke halaman login admin
TC-ADM-07	Data transaksi admin konsisten dengan status pembayaran backend	Edge	1. Selesaikan transaksi sebagai customer 2. Login admin 3. Cek transaksi tersebut	Status transaksi di admin sesuai dengan status backend
TC-ADM-08	Admin melakukan refresh halaman riwayat transaksi	Edge	1. Buka Riwayat Transaksi 2. Refresh halaman	Data transaksi tetap konsisten dan tidak terjadi error

Backend_Test_Cases				
Test Case ID	Test Scenario	Type Test	Test Steps	Expected Result
TC-BE-01	Backend berhasil membuat transaksi checkout dengan request valid	Positive	1. Kirim request checkout dengan event ID valid, kategori tiket tersedia, data customer lengkap	API merespons sukses dan transaksi tercatat dengan status <b>pending</b>
TC-BE-02	Backend menolak checkout dengan data customer tidak lengkap	Negative	1. Kirim request checkout tanpa email atau No. HP	API mengembalikan error validasi (400) dengan pesan field wajib
TC-BE-03	Backend menolak checkout jika melebihi kuota	Negative	1. Kirim request checkout dengan jumlah tiket melebihi kuota	API menolak transaksi dan tidak membuar order
TC-BE-04	Backend memastikan satu transaksi hanya tercatat satu kali	Edge	1. Kirim request checkout yang sama secara bersamaan (double submit)	Hanya satu transaksi yang tercatat, tidak ada duplicate order
TC-BE-05	Backend menerima callback pembayaran QRIS sukses	Positive	1. Kirim callback pembayaran dengan status sukses dan payload valid	Status transaksi berubah menjadi <b>Paid</b>
TC-BE-06	Backend menolak callback pembayaran dengan signature tidak valid	Negative	1. Kirim callback QRIS dengan signature salah	Callback ditolak dan status transaksi tidak berubah
TC-BE-07	Backend menangani callback pembayaran duplicate	Edge	1. Kirim callback pembayaran sukses lebih dari satu kali	Status transaksi tetap <b>paid</b> dan tidak terjadi duplicate proses
TC-BE-08	Backend menangani callback pembayaran terlambat	Edge	1. Kirim callback sukses setelah beberapa menit dari pembayaran	Status transaksi tetap diperbarui menjadi <b>Paid</b>
TC-BE-09	Backend menangani callback pembayaran gagal	Negative	1. Kirim callback dengan status gagal	Status transaksi berubah menjadi <b>Failed</b>
TC-BE-10	Backend generate e-tiket setelah transaksi paid	Positive	1. Transaksi berstatus paid 2. Trigger proses generate e-tiket	E-tiket berhasil dibuat dan dikaitkan dengan transaksi
TC-BE-11	Backend tidak generate e-tiket jika transaksi belum paid	Negative	1. Trigger generate e-tiket untuk transaksi pending	Sistem menolak proses generate e-tiket
TC-BE-12	Backend menangani kegagalan pengiriman e-tiket	Edge	1. Simulasikan kegagalan email/WA service	Status transaksi tetap <b>Paid</b> dan error pengiriman tercatat untuk retry

## Exploratory Testing

### Pendekatan

Exploratory testing dilakukan dengan mensimulasikan perilaku pengguna nyata dan kondisi tidak terduga pada alur pemesanan tiket, pembayaran, e-tiket, dan admin dashboard, tanpa mengikuti test case formal.

### Temuan Exploratory Testing

1. Status transaksi tidak langsung terupdate setelah pembayaran  
**Area eksplor:** *Halaman pembayaran QRIS & Status transaksi*  
**Temuan / Potensi Bug:**  
Setelah customer menyelesaikan pembayaran QRIS, status transaksi di halaman customer masih menampilkan **Pending** selama beberapa waktu.  
**Dampak:**  
Customer ragu apakah pembayaran berhasil dan berpotensi melakukan pembayaran ulang atau menghubungi CS.
2. e-Tiket terkirim ganda saat callback duplicate  
**Area eksplor:** *Callback pembayaran & pengiriman e-tiket*  
**Temuan / Potensi Bug:**  
Jika payment gateway mengirim callback lebih dari satu kali, sistem berpotensi mengirim e-tiket lebih dari sekali ke email atau WhatsApp customer.  
**Dampak:**  
Customer bingung karena menerima beberapa tiket untuk transaksi yang sama dan meningkatkan risiko penyalahgunaan tiket.
3. Ghost Order Akibat User Menutup Halaman Pembayaran  
**Area eksplor:** *Checkout & pembayaran*  
**Temuan / Potensi Bug:**  
User menutup browser setelah checkout namun sebelum pembayaran selesai, transaksi tetap tercatat sebagai **pending** tanpa mekanisme auto-expire.  
**Dampak:**  
Kuota tiket tertahan oleh transaksi yang tidak valid sehingga mengurangi ketersediaan tiket untuk user lain.
4. Data admin tidak sinkronisasi dengan status backend  
**Area eksplor:** *Admin dashboard - riwayat transaksi*  
**Temuan / Potensi Bug:**  
Status transaksi di admin dashboard masih **pending** meskipun backend sudah menerima callback pembayaran sukses.  
**Dampak:**  
Admin salah membaca kondisi transaksi dan berpotensi salah mengambil keputusan (misalnya: menolak klaim customer)
5. QR code tiket masih bisa digunakan lebih dari satu kali  
**Area eksplor:** *Validasi e-tiket / QR code*  
**Temuan / Potensi Bug:**  
QR code e-tiket tetap valid meskipun sudah digunakan sekali untuk masuk event.  
**Dampak:**  
Risiko fraud tinggi karena satu tiket dapat digunakan oleh lebih dari satu orang.
6. Kegagalan pengiriman e-tiket tidak ditangani dengan retry  
**Area eksplor:** *Email & WhatsApp notification*  
**Temuan / Potensi Bug:**  
Jika email atau WhatsApp service sedang down, sistem tidak memiliki mekanisme retry atau penanda pengiriman gagal.  
**Dampak:**  
Customer tidak menerima tiket meskipun transaksi sudah **paid**, menurunkan kepercayaan user.

Bug_Reporting Frontend									
Bug ID	Title	Environment	Precondition	Steps to Reproduce	Expected Result	Actual Result	Severity	Priority	Evidence
BR-FR-01	Status transaksi tidak otomatis terupdate setelah pembayaran QRIS berhasil	Web (Chrome) Staging	Customer sudah checkout dan halaman QRIS tampil	1. Scan QR pembayaran. 2. Selesaikan pembayaran. 3. Kembali ke aplikasi tanpa refresh.	Status berubah menjadi <b>Paid</b> secara otomatis	Status tetap Pending hingga refresh manual	Medium	High	Status Pending masih tampil setelah pembayaran sukses
BR-FR-02	E-tiket terkirim lebih dari satu kali ke email dan WhatsApp	Web (Chrome) Staging	Transaksi Paid dan callback diterima lebih dari satu kali	1. Selesaikan pembayaran QRIS. 2. Sistem menerima callback duplicate	E-tiket dikirim satu kali	E-tiket dikirim lebih dari satu kali	Medium	Medium	Customer menerima e-tiket yang sama berulang



Bug_Reporting Backend									
Bug ID	Title	Environment	Precondition	Steps to Reproduce	Expected Result	Actual Result	Severity	Priority	Evidence
BR-BE-01	Duplicate callback pembayaran memicu proses ganda (e-tiket terikirim ganda)	Staging / Sandbox QRIS (Payment Gateway)	A. Sudah ada transaksi checkout (status pending) B. Pembayaran QRIS berhasil C. Gateway mengirim callback sukses > 1x (duplicate)	1. Checkout tiket sampai muncul QRIS 2. Lakukan pembayaran QRIS hingga sukses 3. Simulasikan callback sukses dikirim 2x (same_order_id/transaction_id) 4. Cek proses pasca bayar (update status, generate tiket, kirim email/WA)	1. Status transaksi menjadi Paid 2. Callback bersifat idempotent: tidak ada proses pasca bayar yang berjalan 2x 3. E-tiket hanya dibuat & dikirim 1x	Callback duplicate dapat memicu proses pasca bayar lebih dari sekali (contoh: e-tiket terikirim ganda ke email/WA)	High	P1	Exploratory Testing: temuan #2; TC_PAY_05; TC_BE_07
BR-BE-02	Ghost order: transaksi pending tidak auto expire saat user menutup halaman pembayaran	Staging / Sandbox QRIS (Payment Gateway)	A. Transaksi checkout dibuat (status pending) B. User sudah masuk halaman pembayaran QRIS C. User menutup tab/browser sebelum proses selesai	1. Checkout tiket sampai halaman pembayaran QRIS 2. Tutup tab/browser sebelum redirect/konfirmasi 3. Tidak melakukan pembayaran 4. Tunggu melewati batas waktu pembayaran (TTL QR) 5. Cek status transaksi & kuota tiket	1. Setelah TTL terlewat, suatu transaksi menjadi Expired/Failed 2. Kuota/reservasi tiket dilepas kembali 3. Tidak ada transaksi menggantung yang menahan kuota	Transaksi tetap pending tanpa mekanisme auto-expire sehingga kuota tiket tertahan (ghost order)	High	P1	Exploratory testing: temuan #3; TC-PAY-03; (handle user menutup halaman pembayaran)

## Automation Awareness

Pada bagian ini, Saya memilih fitur **Login** dan **Checkout** karena keduanya merupakan critical flow pada aplikasi pemesanan tiket dan paling sering terkena regression ketika ada perubahan sistem.

### A. Fitur Login

#### 1. Test Scenario yang layak di-Automation

Untuk fitur login, Saya meng-automation scenario yang:

- Bersifat **stabil**
- Dieksekusi **berulang**
- Menjadi **entry point** ke fitur lain

Contoh scenario yang layak di-automation:

- Login dengan email dan password valid → user berhasil masuk ke halaman utama
- Login gagal karena password salah atau email tidak terdaftar
- Validasi field mandatory (email / password kosong)
- Validasi format email tidak valid
- Akses halaman utama tanpa login → redirect ke halaman login
- Session masih aktif setelah refresh halaman

Alasan: Login adalah core feature, dan automation di area ini membantu mencegah blocker bug yang bisa menghentikan seluruh flow aplikasi.

#### 2. Test scenario yang tidak layak di-Automation

Scenario yang tidak Saya automation antara lain:

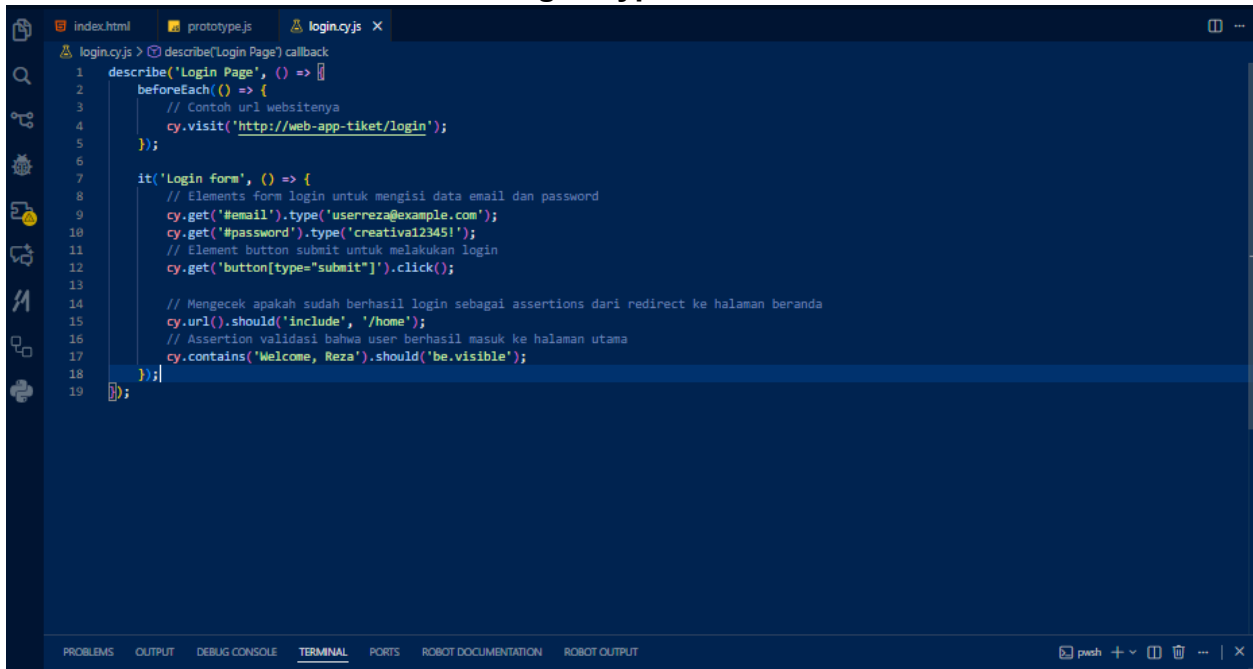
- Validasi UI/UX pesan error (bagian copywriting, warna, layout)
- Exploratory testing terkait brute force / rate limiting
- CAPTCHA atau mekanisme keamanan tambahan (Jika memang ada)

Alasan: Scenario di atas cenderung tidak stabil, sering berubah, dan lebih efektif diuji secara manual atau exploratory.

### 3. Contoh Test Case Manual - Login

Test Case ID	Scenario	Steps to produce flow	Expected Result
TC-LGN-001	Login dengan kredensial yang valid	1. User membuka halaman login 2. Input email dan password yang valid 3. Klik tombol login	User berhasil login dan dialihkan ke halaman utama dengan session login aktif

## 4. Pseudo Code Automation Login Cypress

The image shows a code editor with a dark theme. The file explorer on the left shows 'index.html', 'prototype.js', and 'login.cy.js'. The main editor area shows the content of 'login.cy.js'. The code is a Cypress test suite for a login page. It starts with a 'describe' block for 'Login Page' and a 'beforeEach' hook. Inside the hook, it uses 'cy.visit' to go to 'http://web-app-tiket/login'. The main test block 'it' is for the 'Login form'. It contains several steps: getting the email input, typing 'userreza@example.com', getting the password input, typing 'creativai2345!', clicking the submit button, and then assertions to check the URL includes '/home' and the page contains 'Welcome, Reza'.

```
1 describe('Login Page', () => {  
2   beforeEach(() => {  
3     // Contoh url websitenya  
4     cy.visit('http://web-app-tiket/login');  
5   });  
6  
7   it('Login form', () => {  
8     // Elements form login untuk mengisi data email dan password  
9     cy.get('#email').type('userreza@example.com');  
10    cy.get('#password').type('creativai2345!');  
11    // Element button submit untuk melakukan login  
12    cy.get('button[type="submit"]').click();  
13  
14    // Mengecek apakah sudah berhasil login sebagai assertions dari redirect ke halaman beranda  
15    cy.url().should('include', '/home');  
16    // Assertion validasi bahwa user berhasil masuk ke halaman utama  
17    cy.contains('Welcome, Reza').should('be.visible');  
18  });  
19 });
```

## B. Fitur checkout tiket

### 1. Test scenario yang layak di-Automation

Untuk checkout, Saya fokus meng-automation bagian **sebelum pembayaran eksternal**, karena ini masih berada dalam kontrol sistem internal.

Scenario yang layak di-automation diantaranya:

- User memilih kategori tiket dan quantity
- Validasi data diri (nama, email, no HP)
- Validasi format email dan nomor HP
- Checkout berhasil → redirect ke halaman pembayaran QRIS
- Perhitungan total harga sesuai jumlah tiket
- Prevent checkout jika stok tiket habis

Alasan: checkout adalah **business-critical flow** dan rawan regression, terutama terkait validasi data dan konsistensi harga.

### 2. Test scenario yang tidak layak di-Automation

Scenario yang tidak saya automation:

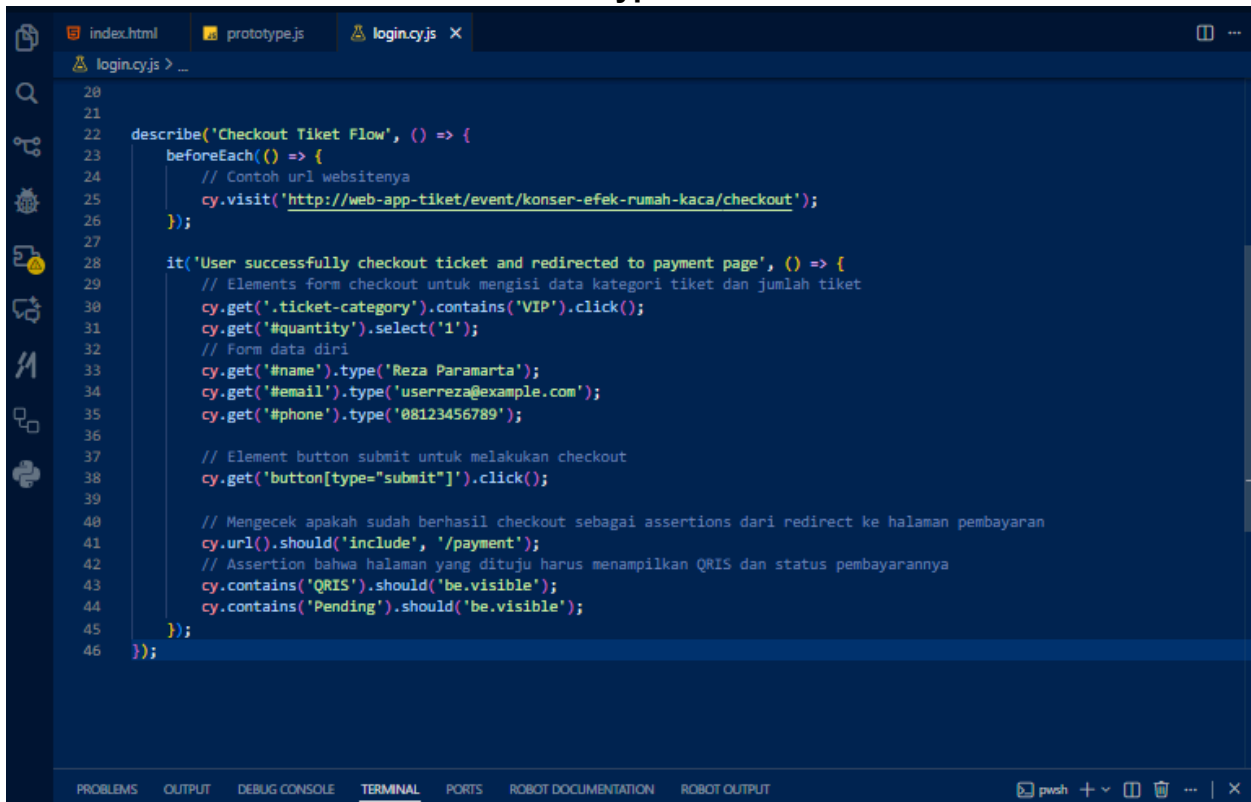
- Proses pembayaran QRIS secara real (Scan QR)
- Simulasi payment timeout real dari payment gateway
- Exploratory testing ghost order akibat user menutup halaman

Alasan: Scenario ini bergantung pada **external service** dan kondisi real-time, sehingga lebih tepat diuji melalui manual testing, mock, atau API testing terpisah.

### 3. Contoh Test Case Manual - Checkout

Test Case ID	Scenario	Steps to produce flow	Expected Result
TC-PAY-001	Checkout tiket dengan data valid	<ol style="list-style-type: none"> <li>1. User memilih event dan kategori tiket</li> <li>2. User mengisi data diri dengan benar</li> <li>3. Klik tombol checkout</li> </ol>	Sistem menampilkan halaman pembayaran QRIS, total harga yang sesuai dan status transaksi <b>'Pending'</b>

### 4. Pseudo Code Automation Checkout Cypress



```

20
21
22 describe('Checkout Tiket Flow', () => {
23   beforeEach(() => {
24     // Contoh url websitenya
25     cy.visit('http://web-app-tiket/event/konser-efek-rumah-kaca/checkout');
26   });
27
28   it('User successfully checkout ticket and redirected to payment page', () => {
29     // Elements form checkout untuk mengisi data kategori tiket dan jumlah tiket
30     cy.get('.ticket-category').contains('VIP').click();
31     cy.get('#quantity').select('1');
32     // Form data diri
33     cy.get('#name').type('Reza Paramarta');
34     cy.get('#email').type('userreza@example.com');
35     cy.get('#phone').type('08123456789');
36
37     // Element button submit untuk melakukan checkout
38     cy.get('button[type="submit"]').click();
39
40     // Mengecek apakah sudah berhasil checkout sebagai assertions dari redirect ke halaman pembayaran
41     cy.url().should('include', '/payment');
42     // Assertion bahwa halaman yang dituju harus menampilkan QRIS dan status pembayarannya
43     cy.contains('QRIS').should('be.visible');
44     cy.contains('Pending').should('be.visible');
45   });
46 });
  
```

Dengan pendekatan ini, Saya ingin menunjukkan bahwa:

- Automation digunakan untuk regression dan critical flow
- Tidak semua scenario harus di-automation
- Manual testing tetap penting untuk **exploratory**, **edge case**, dan **integrasi eksternal**
- Pemilihan test automation selalu mempertimbangkan **stabilitas**, **value**, dan **maintainability**.

# TEST PLAN

## Web Application - Event Ticketing System

### 1. Document Information

Item	Detail
Project Name	Web-Based Event Ticketing System
Document Type	Test Plan
Version	1.0
Prepared By	Quality Assurance (Reza Paramarta)
Date	8 January 2026
Environment	QA / Staging
Test Level	System Testing

### 2. Purpose & Objective

Tujuan dari test plan ini adalah untuk memastikan bahwa aplikasi web pemesanan tiket event telah memenuhi kebutuhan fungsional, keadaan transaksi, serta konsistensi data sebelum dirilis ke production.

Pengujian difokuskan pada:

- Validasi end-to-end customer journey
- Integrasi payment QRIS
- Konsistensi status transaksi
- Keandalan pengiriman dan validasi e-tiket
- Pencegahan fraud dan inconsistency

### 3. Scope of Testing

#### 3.1 In Scope

Customer Side

- Landing Page
- Halaman Detail Event

- Pemilihan kategori tiket & quantity
- Checkout & input data customer
- Halaman pembayaran QRIS
- Status transaksi (Pending / Paid )
- Pengiriman e-tiket (Email & WhatsApp)
- Validasi QR code (single-use)

#### Admin Side

- Login Admin
- Riwayat transaksi
- Data peserta terdaftar
- Sinkronisasi status transaksi

#### Integration

- Payment Gateway (QRIS)
- Callback pembayaran (success, failed, duplicate, delayed)
- Email service
- WhatsApp API

#### Non-Functional (Basic)

- Data consistency
- Concurrency (kuota tiket)
- Error handling
- Penetration & security testing pada user journeys, API, dan business-critical transactions
- Load testing untuk mengukur performance response times, API latency, error rates, dan connection handling

### 3.2 Out of Scope

- Rekonsiliasi keuangan di sisi bank
- Uji compliance regulasi eksternal

## 4. Test Approach

### 4.1 Manual Testing

Digunakan untuk:

- Exploratory testing
- Edge & negative cases
- Integrasi payment QRIS

- Simulasi ghost order
- Validasi QR Code usage
- UI/UX validation

## 4.2 Automation Testing

Digunakan untuk:

- Regression testing
- Critical & stable flow
- Login
- Checkout sebelum payment eksternal

Tool Automation:

- Cypress (UI automation)

## 4.3 API Testing

Digunakan untuk:

- Callback pembayaran QRIS
- Generate e-tiket
- Validasi status transaksi

## 5. Test Types

- Functional Testing
- Positive & Negative Testing
- Edge Case Testing
- Exploratory
- Regression Testing (Automation)
- Basic Compatibility Testing (Desktop & Mobile)
- Load Testing

## 6. Test Environment

Item	Detail
Environment	QA / Staging
Search Engine	Chrome, Mozilla Firefox, Microsoft Edge
Device	Desktop & Mobile
OS	Windows, macOS, Android
Payment	QRIS Sandbox

Email	SMTP Test
WhatsApp	Sandbox / Mock API

## 7. Status

Status pengujian:

- Todo : Test case belum dilakukan test
- In testing : Test case sedang dilakukan test
- Blocked : Test case mengalami blocker
- Passed : Test case sudah dilakukan test tanpa ada kendala
- Failed : Test case sudah dilakukan test dan terdapat kendala atau error

## 8. Impact/Severity Level

Tingkatan prioritas dari suatu bug/error:

- High : Bug/error tersebut harus segera di fix
- Medium : Bug/error harus segera di fix, dengan catatan tidak ada bug/error yang statusnya high
- Low : Bug/error tersebut hanya perbaikan *cosmetic/usability issue*

## 9. Entry Criteria

Pengujian dapat dimulai apabila:

- Build aplikasi sudah fully completed dan stabil
- Environment testing QA ready dan tersedia
- Requirement dan flow bisnis telah dipahami
- Akses admin dan test account tersedia
- Test plan sudah di-approved
- Tester dan seluruh role yang involved sudah di-briefing dan siap untuk melakukan testing sesuai responsibility area masing-masing

## 10. Exit Criteria

Pengujian dianggap selesai apabila:

- Tidak ada blocker dan critical (high & medium) bug/defect, all reported bugs sudah di-fix dan di-retest dan E2E regression
- Semua critical flow lulus testing
- Status transaksi konsisten antara customer dan admin
- E-tiket berhasil diterbitkan untuk transaksi Paid
- Automation test utama lulus (login & checkout)
- Integration test completed



## 11. Major Tasks, Timeline, & Deliverables

Phase	Task	Target Start	Target Stop	Actual Start	Actual Stop	Deliverable(s)
P1	Requirement Analysis & Test Planning	07/01/26	09/01/26	07/01/26	08/01/26	• Test Plan Document
P2	System Functional Test Case Design	10/01/26	12/01/26	TBD	TBD	• Test Scenario & Test Case
P3	Env Readiness & Test Data Prep	13/01/26	14/01/26	TBD	TBD	• QA Staging Environment Ready
P4	System Functional Testing (Build 1-3)	15/01/26	28/01/26	TBD	TBD	• Bug Report • Execution Log • Exploratory Testing Notes
P5	Baseline Load Test (~1 month, include the following: preparation, execution, diagnostics, analysis, optimization & fine-tuning, re-test, reporting & handover)	31/01/26	28/02/26	TBD	TBD	• Performance & Load Test Results

P6	Regression & Final Go/No Go Sign-off	01/03/26	07/03/26	TBD	TBD	<ul style="list-style-type: none"> <li>• Automation Test Script (Cypress)</li> <li>• Test Summary Report</li> </ul>
----	--------------------------------------	----------	----------	-----	-----	---

## 12. Parameter List (Functional, Usability, & Security Testing Main Items)

Daftar ini memetakan skenario pengujian berdasarkan peran user untuk memastikan validasi yang tepat di sisi Admin maupun Customer.

User Access	Context	Scenario Type	Detailed User Actions	Parameter List	Expected Outcome / Output
Admin	Authentication	True Positive	Login ke dashboard admin menggunakan kredensial valid.	{User: "admin_reza", Pass: "AdminTicket 2026}	Sistem berhasil masuk ke dashboard admin dan menampilkan riwayat transaksi.
Admin	Data Sync	True Positive	Memverifikasi sinkronisasi status transaksi antara sistem dan bank.	{Filter_Date: "2026-01-08", Status: "Paid"}	Status transaksi pada dashboard admin berubah menjadi "Paid" secara <i>real-time</i> setelah pembayaran sukses.
Customer/ Peserta/ Registrant	Ticket Reservation	True Positive	Pemesanan tiket sukses dengan kuota tersedia melalui flow normal.	{Event_ID: "EVT-01", Qty: 2, Email: "cust@mail.com"}	Sistem <i>generate</i> e-tiket dan mengirimkannya via Email/ WhatsApp.

Customer/ Peserta/ Registrant	Input Validation	False Positive	Pendaftaran menggunakan nama sah yang mengandung simbol sensitif (tanda kutip).	{Name: "J'S Nur'ayuni."}	Sistem berhasil memproses pendaftaran tanpa menganggap simbol sebagai ancaman keamanan.
Customer/ Peserta/ Registrant	Invoice Progress	False Positive	Mengakses halaman pembayaran dengan ID transaksi yang tidak terdaftar.	{Transaction_ID: "INV-999-NO TFOUND"}	Sistem menampilkan pesan kesalahan "Invoice Not Found" atau dialihkan kembali ke Landing Page.
Customer/ Peserta/ Registrant	Business Logic	False Positive	Pencarian nama event menggunakan simbol wildcard %%% pada kategori 'All'.	{Search_Key: "%%%", Category: "All"}	Sistem menampilkan pesan "No events found" dan tidak menyebabkan error pada query database.
Customer/ Peserta/ Registrant	Security (XSS)	Negative (malicious)	Mencoba menyuntikkan <i>script</i> berbahaya pada formulir pendaftaran peserta.	{Name: "<script>alert('XSS')</script>"}>"}	Sistem melakukan sanitasi input dan tidak mengeksekusi skrip tersebut (muncul sebagai teks biasa).

<b>Customer/ Peserta/ Registrant</b>	<b>Security</b>	<b>Negative</b>	Manipulasi penulisan tag HTML ( <i>Filter Evasion</i> ) pada kolom pendaftaran.	{Name: "<script>confirm(1)</script>"}	Sistem keamanan tetap mendeteksi dan memblokir upaya injeksi meskipun penulisan dimanipulasi.
--	-----------------	-----------------	---	---------------------------------------	---

### 13. Workload Profile & Parameter List (Load Testing)

Daftar parameter untuk load test dengan fokus pada *critical functions*.

Module/ Functional	Business Context/ Transaction	User Action Type	Parameter List	Approx. Value /Mo	Test Percentage (%)
Login / Auth	Authentication Flow	Data Entry	{User, Pass, Grant_Type, Device_ID, App_Ver}	15,000	15%
Landing Page & Search	Discovery Flow	Data Enquiry	{Search_Key, Category_ID, Sort_By, Limit, Page}	20,000	35%
Reservation	Order Initiation (Ticket Booking)	Data Entry	{Event_ID, Ticket_Type, Qty, Email, Name}	12,000	20%
Checkout	Payment Request	Data Entry	{Order_ID, Pay_Method, Token, Total, Voucher}	8,000	10%
Integration	Payment Callback	Data Update	{Trans_ID, Status_Code, Signature, Amount, Time}	8,000	10%

Notification	Digital Fulfillment	Action Trigger	{Ticket_ID, Recipient, Link, Template_ID}	8,000	10%
--------------	---------------------	----------------	---	-------	-----

## 14. Risk & Mitigation

Risk	Impact	Mitigation
Callback payment terlambat	Status tidak sinkron	API testing & retry logic validation
Ghost order	Kuota tiket tertahan	Validasi auto-expire
Duplicate callback	E-tiket terkirim ganda	Idempotency testing
QR reusable	Fraud	Single-use QR validation
Email/WA gagal kirim	Customer complaint	Retry & failure logging

## 15. Multi-Vendor Responsibility Mapping

Vendor	Specialization	Involvement	Responsibility Area
<b>Vendor 1</b>	Functional QA	40 HPW <i>(Hours per Week)</i>	Lead: Exploratory Testing, Manual Cases, & UX Validation.
<b>Vendor 2</b>	Performance	40 HPW <i>(Hours per Week)</i>	Lead: Load & Stress Test Design, Scripting, & Analysis.
<b>Vendor 3</b>	Security	15 HPW <i>(Hours per Week)</i>	Penetration Testing & Fraud Prevention (Single-use QR).
<b>Vendor 4</b>	Infrastructure	25 HPW <i>(Hours per Week)</i>	Env Readiness, Monitoring Tools Setup, & Cloud Management.

<b>Vendor 5</b>	Automation	30 HPW (Hours per Week)	Cypress Regression Scripts Development.
<b>Vendor 6</b>	API Integration	20 HPW (Hours per Week)	Payment Gateway Validation, Callback Logic, & Idempotency.

## 16. Team Roles & Responsibilities

Name	Role	Responsibilities
Reza Paramarta	QA Manager	Test Planning, Bug reporting, Managing test
Hannah Lee	Tester	Execution (testing on different devices, search engines, and OS), Bug reporting
Andre Le'Baar & Liz Johnson	Developer	Bug fixing & technical clarification
J'S Nur'Ayuni	Product / Stakeholder	Project ownership, requirement clarification & sign-off

## 17. Test Execution Strategy

- Prioritaskan *high-risk areas*:
  - Payment Transaction: : Validasi integrasi QRIS, penanganan callback, dan sinkronisasi status (Pending / Paid / Expired) antara database dan tampilan UI.
  - e-Tiket & QR Code *Generation*: Memastikan e-tiket hanya *generated/printed* untuk transaksi "Paid" dan QR Code valid hanya untuk penggunaan satu kali (*single-use*), tidak *reusable*, kuota tiket berkurang dengan benar saat transaksi terjadi bersamaan (*race condition*).
- Jalankan *exploratory testing* di awal fase untuk menemukan *edge cases* yang belum tercover oleh initial test case standar.
- Automation dijalankan untuk regression, menggunakan Cypress untuk menguji alur/fitur kritis dan stabil (Login, Search, Add to Cart) setiap kali ada perubahan kode serta hindari / *minimize* pengujian manual berulang pada fitur dasar, sehingga tim QA bisa fokus pada area baru atau area yang bermasalah.
- Re-test & regression (pasca bug fix):

- Re-test: Memverifikasi secara spesifik bahwa bug yang ditemukan di Phase 4 telah diperbaiki oleh Developer.
- Regression: Menjalankan kembali skenario pada modul terkait untuk memastikan perbaikan tidak merusak fitur yang sudah ada.
- End-to-End (E2E) Regression: Melakukan pengujian alur lengkap dari awal hingga akhir (dari Login -> Pilih Tipe Tiket -> Bayar -> Terima e-tiket) untuk memastikan seluruh subsistem terintegrasi secara sempurna setelah adanya perubahan kode.

## 18. Approval

Name	Role	Status
Reza Paramarta	Functional Quality Assurance Lead	Approved
Lead Vendor 2	Performance Lead	Approved
Lead Vendor 3	IT Security Lead	Approved
Lead Vendor 4	Infrastructure Lead	Approved
Lead Vendor 5	Automation Lead	Approved
Lead Vendor 6	Integration Lead	Approved
J'S Nur'Ayuni	Stakeholder (Business Process / Product Owner)	Pending

# Testing Mind Map

## 1. Purpose

Mind map testing digunakan untuk:

- Memetakan area pengujian secara menyeluruh
- Mengidentifikasi risiko tersembunyi
- Membantu exploratory testing
- Menjadi referensi saat membuat test scenario dan test case

Mind map ini disusun berdasarkan:

- Alur bisnis aplikasi
- Integrasi eksternal (Payment & notification)
- Risiko yang telah diidentifikasi pada tahap analisis

## 2. Testing Mind Map - Web Event Ticketing Application

Mind map dituliskan dalam bentuk hirarki teks agar mudah dipahami, ditelusuri, dan di-maintain sebagai dokumen QA.

### a. Customer Side

Landing page

- Load halaman
- Event list tampil
- Event expired / sold out
- Responsif (Web & Mobile)

Detail Event

- Informasi event lengkap
- Harga tiket sesuai kategori
- Event sudah expired
- Event sold out
- Konsistensi data front-end vs back-end

Pilih Tiket

- Kategori tiket (VIP / Regular)
- Quantity melebihi stok
- Update harga real time
- Race condition kuota

Checkout

- Validasi field mandatory (nama, email, no. HP)



- Format email tidak valid
- Format nomor HP (+62 / 08)
- SPecial character input
- Total harga sesuai
- Ghost order (user menutup halaman sebelum pembayaran selesai)

#### Payment QRIS

- Redirect ke halaman pembayaran
- Status transaksi pending
- Callback sukses
- Callback gagal
- Callback duplicate
- Callback terlambat
- User melakukan refresh halaman saat QR tampil

#### E-tiket

- Generate QR Code
- Pengiriman e-tiket via email
- Pengiriman e-tiket via whatsapp
- Gagal kirim notifikasi
- Duplicate e-tiket
- Konten e-tiket tidak sesuai event
- QR code reusable (fraud risk)

### **b. Admin Side**

#### Login Admin

- Login dengan kredensial valid
- Login gagal
- Session timeout
- Unauthorized access

#### Riwayat Transaksi

- Status Pending / Paid
- Status transaksi tidak sinkron dengan backend
- Filter transaksi
- Sorting data
- Pagination

#### Data peserta

- Data peserta sesuai transaksi paid
- Data duplikat
- Data tidak sinkron
- Export data peserta (Jika tersedia )

### **c. Integration**

#### Payment Gateway (QRIS)

- Valid callback payload
- Invalid callback payload
- Duplicate callback
- Timeout callback
- Amount mismatch

#### Email Service

- SMTP service down
- Retry mechanism pengiriman
- Logging kegagalan pengiriman

#### WhatsApp API

- API error
- Delay pengiriman
- Retry mechanism

### **d. Non-Functional & Risk Area**

#### Data Consistency

- Konsistensi data antara customer dan admin view
- Konsistensi data front-end dan back-end

#### Security (Basic)

- Session handling
- Token expired
- Unauthorized access

#### Concurrency

- Multiple user melakukan checkout secara bersamaan
- Akses kuota terakhir (race condition)
- Page load time
- Stuck loading
- Error handling yang user-friendly

## **3. How Mind Map is Used in Testing**

Mind map ini digunakan untuk:

- Menentukan prioritas testing berdasarkan risiko
- Menjadi dasar pembuatan test scenario dan test case
- Panduan pelaksanaan exploratory testing
- Referensi diskusi antara QA, Developer, dan Product

## Pendekatan Manual vs Automation Testing

Dalam aplikasi web pemesanan tiket event ini, Saya menerapkan hybrid testing approach, yaitu kombinasi **manual testing** dan **automation testing**, dengan mempertimbangkan alur bisnis, risiko aplikasi, serta stabilitas fitur.

Pendekatan ini dipilih karena aplikasi memiliki critical transaction flow, integrasi payment eksternal (QRIS), serta potensi risiko finansial dan fraud yang tinggi.

### Manual Testing Approach

**Manual testing digunakan sebagai pendekatan utama pada area yang:**

- Memiliki banyak kemungkinan edge case
- Bergantung pada integrasi eksternal
- Membutuhkan observasi dan eksplorasi manusia

**Area yang diuji secara manual:**

- Exploratory testing pada flow checkout dan pembayaran
- Payment QRIS (callback sukses, gagal, duplicate, dan terlambat)
- Ghost order akibat user menutup halaman pembayaran
- Pengiriman e-tiket via email dan whatsapp
- Validasi QR code (single-use, reusable risk)
- Sinkronisasi status transaksi antara customer view dan admin dashboard
- Validasi UI/UX dan pesan error

**Alasan:**

Berdasarkan hasil exploratory testing, banyak resiko muncul dari:

- Callback payment yang tidak konsisten
- Status transaksi yang mismatch
- Integrasi notifikasi yang gagal

Skenario ini kurang stabil untuk di-automation dan lebih efektif diuji secara manual dan eksploratif.

### Automation Testing Approach

Automation testing digunakan pada area yang:

- Bersifat stabil
- Sering dilakukan berulang
- Menjadi critical path aplikasi
- Rentan terhadap regression bug

**Area yang di-automation:**

- Login (valid, invalid, mandatory field, session handling)
- Checkout sebelum pembayaran eksternal
  - Pemilihan tiket
  - Validasi input data customer
  - Perhitungan total harga
  - Redirect ke halaman pembayaran QRIS

**Tools:**

- Cypress untuk UI automation
- API intercept / mocking untuk menjaga stabilitas test

**Alasan:**

Automation membantu memastikan bahwa:

- Core flow selalu berjalan normal
- Perubahan di satu fitur tidak merusak flow lain
- Regression dapat terdeteksi cepat dan konsisten

Namun, automation tidak digunakan untuk proses scan QR real payment karena ketergantungan pada external service.

## Perbandingan Manual vs Automation

Aspek	Manual Testing	Automation Testing
Tujuan	Eksplorasi & Validasi kompleks	Regression & Konsistensi
Cocok untuk	Payment, Callback, QR, e-tiket	Login & checkout
Fleksibilitas	Tinggi	Rendah
Kecepatan jangka panjang	Rendah	Tinggi
Maintenance	Rendah	Tinggi

## Kesimpulan Pendekatan

Pendekatan yang digunakan adalah risk-based hybrid testing:

- Manual testing fokus pada area beresiko tinggi seperti payment, status transaksi, dan e-tiket
- Automation testing fokus pada critical flow yang stabil dan berulang seperti log in dan checkout

Pendekatan ini memastikan kualitas aplikasi tetap terjaga tanpa over-automation, sekaligus efisien dari sisi waktu dan biaya.