**Test Strategy for Acko Website**

**1. Purpose**

**To define the high-level approach, testing scope, and quality objectives for the Acko website.**

**2. Scope**

**In Scope**

* **End-to-end web flows: registration, login, insurance quotation, purchasing, policy management, claim filing & tracking, customer support.**
* **API and backend validation supporting web functions.**
* **UI and UX validation across supported browsers and devices.**
* **Performance and security (web vulnerabilities, compliance).**
* **Accessibility for differently-abled users.**

**Out of Scope**

* **Deprecated/legacy modules being removed.**
* **Third-party integrations not owned by Acko.**

**3. Quality Goals & Objectives**

* **Deliver a defect-free, reliable, and secure platform.**
* **Meet regulatory compliance (data privacy, accessibility).**
* **Ensure smooth insurance purchase and claim process.**
* **Achieve 99.5%+ uptime post-release.**

**4. Test Levels & Types**

| **Level/Test Type** | **Description** | **Tools/Approach** |
| --- | --- | --- |
| **Unit** | **Isolated code units (dev owned)** | **JUnit, Mocha, PyTest** |
| **Integration** | **Service/data flow between modules** | **Postman, REST-assured** |
| **System** | **E2E user and insurance flows** | **Selenium, Cypress** |
| **Regression** | **All core and critical flows** | **Automated/manual** |
| **Performance** | **Load, stress, scalability** | **JMeter, Locust** |
| **Security** | **OWASP Top 10, input validation, etc.** | **OWASP ZAP, Burp Suite** |
| **Usability/Accessibility** | **Color contrast, tab order, screen readers** | **Axe, manual review** |

**5. Test Design Techniques**

* **Equivalence partitioning**
* **Boundary value analysis**
* **State transition**
* **Scenario-based, risk-based, exploratory**

**6. Environments**

* **DEV, QA, UAT: Production-like, with anonymized data.**

**7. Entry/Exit Criteria**

**Entry**

* **Requirements/stories marked “Ready for Test.”**
* **Stable build deployed to test environment.**

**Exit**

* **All priority defects fixed/closed.**
* **All functional and regression test cases executed and passed as per Definition of Done.**

**8. Test Data Management**

* **Use masked/anonymized but realistic customer and policy data.**
* **Seed and refresh before major cycles.**

**9. Risks and Mitigations**

| **Risk** | **Mitigation** |
| --- | --- |
| **Changing requirements** | **Continuous grooming and sprint reviews** |
| **Environment instability** | **Monitor, rollback plan, redundant servers** |

**10. Tools**

| **Category** | **Examples** |
| --- | --- |
| **Test Management** | **JIRA, TestRail, QA Touch** |
| **Automation** | **Cypress, Selenium, Playwright** |
| **API Testing** | **Postman, REST-assured** |
| **Performance** | **JMeter, Locust** |
| **Security** | **OWASP ZAP, Burp Suite** |

**1. Requirements for the Acko Website**

Functional Requirements:

* User registration and login (account creation, multi-channel login like Google/OTP).
* Shopping insurance products (car, bike, health, travel, gadget).
* Premium calculation and quick quotes.
* Online policy purchase with digital payment (cards, UPI, netbanking).
* Manage policies (view details, renew, download documents).
* Initiate, submit, and track claims (file new, upload documents, monitor status).
* Access to personalized dashboard (profile, notifications, transaction history).
* Customer support (FAQ, chatbot, contact form, toll-free).
* Product and plan search, with filtering and sorting.
* Accessibility and usability for differently-abled users.
* Security features: HTTPS, mobile/email OTP, session management.

Non-Functional Requirements:

* Fast page loading (≤3 seconds for core flows).
* Cross-browser/device compatibility.
* Data privacy and compliance (GDPR/India DPDP).
* High availability (uptime >99.5%).
* Secure payment gateways.
* Scalable architecture for traffic spikes.

**2. Test Strategy for Acko Website**

Test Objectives:

* Ensure flawless and seamless user journey from registration to claims.
* Validate business-critical and regulatory insurance functions.
* Verify robustness, security, and usability for all modules and flows.

In Scope:

* End-to-end functional flows for registration, login, shopping, purchase, management, claims, and support.
* Usability across devices/browsers.
* API/backend testing.
* Performance (load, stress).
* Security (vulnerabilities, compliance).
* Regression and smoke testing each sprint.

Out of Scope:

* 3rd party integrations not owned by Acko (unless specified).
* Legacy modules being deprecated.

Test Types & Tools:

| **Type** | **Examples/Tools** |
| --- | --- |
| UI Automation | Selenium, Cypress, Playwright |
| API Testing | Postman, REST-assured |
| Performance | JMeter, Locust |
| Security | OWASP ZAP, Burp Suite |
| Test Management | QA Touch, JIRA, TestRail |

Environments: DEV, QA, UAT with production-like data (masked/anonymized).

Test Design Techniques:

* Equivalence partitioning & boundary value analysis.
* Risk-based exploratory.
* State transition and scenario-based.

Entry/Exit Criteria:

* Entry: Stable build, user stories “Ready for Test”.
* Exit: All major/critical defects fixed, acceptance criteria met, regression pass rate as per Definition of Done.

**3. Test Plan (Per Sprint – Agile Example)**

**Test Plan Overview (Sprint N)**

| **Item** | **Description** |
| --- | --- |
| Sprint Goal | E.g., Launch new digital claim submission process |
| Scope | User story list, related modules, regression impact |
| Resources | QA Engineers, Dev, PO, Scrum Master |
| Timeline | Sprint days 1–10 (2-week sprint) |
| Environments | QA/UAT, production-similar config/data |

**Test Scenarios (Sprint N Example: Claim Filing Feature)**

1. Register/log in via mobile/email and access dashboard.
2. Navigate to ‘File New Claim’.
3. Fill and submit claim with valid data (documents uploaded).
4. Validate error on missing/invalid documents.
5. Check system emails/SMS sent for claim reference.
6. Track claim status from dashboard.
7. Attempt claim filing on unsupported browsers/devices.
8. Security: CSRF/XSS protection during file uploads.
9. API returns correct claim references, handles failures gracefully.

**Regression Suite (Minimum for Every Sprint)**

* Registration/login.
* Policy search and purchase.
* Payment and renewal flows.
* Policy management and download.
* Customer support access.
* Dashboard access.

**Defect Tracking**

* All found issues logged in JIRA (blockers handled within sprint).

**4. Sample Test Cases (with Scenarios)**

| **TC#** | **Scenario** | **Steps** | **Expected Result** |
| --- | --- | --- | --- |
| TC1 | Register new user w/valid details | Open site → Register → Fill valid info → Submit | Account created, redirected to dashboard |
| TC2 | Register with duplicate email | Open site → Register → Use existing email → Submit | Error: email already exists |
| TC3 | Login with valid credentials | Open site → Login → Enter correct info | Successful login, redirected to dashboard |
| TC4 | Quick insurance quote (car) | Home → Car Insurance → Enter vehicle details → Submit | Quote shown with premium |
| TC5 | Buy policy (logged in) | Login → Select quote → Proceed to buy → Complete payment | Policy issued, soft copy downloadable |
| TC6 | File new claim (with required docs) | My Policies → File Claim → Upload docs → Submit | Claim created, confirmation screen/email sent |
| TC7 | File claim – invalid/missing document | Go to File Claim → Skip required document/upload wrong format | Error message displayed |
| TC8 | Track claim status | My Claims → View Status | Status tracking, timeline shown |
| TC9 | Renew existing policy | My Policies → Select expiring → Renew → Payment | Policy renewed, updated status |
| TC10 | Payment error handling | Proceed to pay → Simulate failed transaction | Proper error, retry option |
| TC11 | Accessibility (tab navigation) | Register/Login page → Tab through form fields | Focus order logical, visually indicated |
| TC12 | Cross-device login | Register/login on mobile/tablet/browser | Experience and layout consistent |
| TC13 | Security – session timeout | Login → Stay idle for session timeout duration | Auto-logout triggered, session ended |
| TC14 | GDPR/policy doc download | My Profile → Download policy/docs | Correct PDF downloaded, privacy info maintained |
| TC15 | API returns correct error for unauthorized access | Attempt API request without auth token | 401/403 returned |

**5. Agile Test Plan Best-Practices (Per Sprint)**

* Test scope kept dynamic: Stories/tasks in sprint backlog are tested.
* Test cases updated/added: For any new or refactored feature.
* Automation scripts refactored: Maintain CI/CD pipeline stability.
* Regressions always run: Risk-based for core modules.
* Demo/Sign-off: Only user stories passing all acceptance, integration, and regression criteria are demoed to the PO/stakeholders.