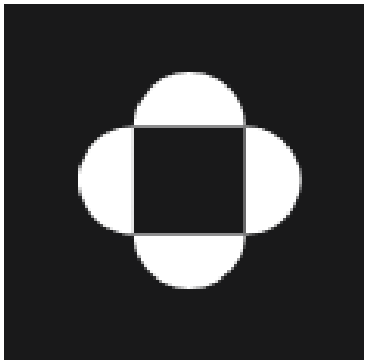

SOAR Test Plan

Comprehensive Test Plan: Strategy, Approach, and Execution for System Validation

This test plan outlines the strategy, approach, and execution for validating critical system functionalities. It prioritizes risk-based testing, ensuring secure logins, smooth upgrades, loan notifications, and third-party integrations. The execution strategy includes functional, integration, and UAT testing, managed via tools like JIRA, to deliver a reliable and user-friendly product.

Project overview

- Enhance user account security and authentication.
- Implement premium account upgrade workflows.
- Streamline loan approval processes and notifications.
- Enable corporate registration validation using third-party integrations.

Scope

Sprint 1	● User Authentication ● Device Management
Sprint 2	● Upgrade to Premium
Sprint 3	● Loan Notifications Approvals
Sprint 4	● Corporate Registration Verification
Out of scope	● Non-production environments not replicating production settings. ● External services or modules not directly impacting the current scope.

Testing strategy

The primary objective of this test strategy is to ensure the system's functionality, integration, and performance align with user requirements and business goals. The tasks and responsibilities are as follows:

Objectives:

- Validate workflows for login, premium account upgrades, loan approvals, and corporate registration.
- Ensure integration with third-party services and SMS notifications.
- Identify and mitigate risks in high-impact features.
- Confirm data security, usability, and scalability.

Tasks:

- **Requirement Analysis**: Review user stories, acceptance criteria, and technical specifications.
- **Test Design:** Create test cases for functional, integration, and performance scenarios.
- **Test Execution:** Perform functional, integration, regression, and performance testing.
- **Defect Management:** Log, track, and retest defects in JIRA or equivalent tools.
- **Reporting:** Provide daily status reports, defect trends, and final test summary reports.

Responsibilities:

- **QA Team:** Validate features, ensure defect resolution, and maintain test data integrity.
- **Developers:** Fix defects and support environment setup
- **Test Execution:** Perform functional, integration, regression, and performance testing.
- **Test Manager:** Oversee the testing process, manage resources, and ensure timely completion.
- **Stakeholders:** Review and approve test outcomes.

Assumptions

- Requirements and user stories are clearly defined, approved, and not subject to significant changes during testing.
- Test environments mimic the production setup, including mock services for external integrations.
- All dependent modules and third-party services are functional and accessible during testing.
- Adequate test data will be available in QA and staging environments.
- Test resources (e.g., tools, environments) are allocated and functional before testing begins.

a. Level of testing

Level	Description	Focus Areas	Entry Criteria	Exit Criteria
Unit Testing	Tests individual components or modules in isolation.	Functions and methods.	Development of individual modules is complete.	All unit test cases pass with no critical defects.
Integration Testing	Ensures that modules and components interact seamlessly with each other.	Data flow between modules. External API integration.	All relevant modules are unit-tested and integrated.	Integration points validated successfully.
Functional Testing	Verifies that the system functions as per the defined requirements.	Login and session management. Notifications. Document upload and workflows.	Functional specifications are finalized.	All functional test cases pass with no major defects.
Regression Testing	Ensures that new changes do not impact existing features.	Features impacted by new changes. Critical workflows.	Code changes are implemented, and previously tested features are stable.	Regression suite executed with no critical issues.
Performance Testing	Validates the system's ability to handle high traffic and load.	Bulk login attempts. High-volume notifications.	Functional testing is complete. Performance environment is set up.	System meets defined performance benchmarks.
Security Testing	Ensures the system is secure and protects sensitive data.	Login and OTP validation. Data encryption.	Functional testing is complete. Security requirements are defined.	System meets security standards with no critical vulnerabilities.
Usability Testing	Evaluates the user experience and ease of use.	User workflows (e.g., premium upgrades, registration validation).	UI designs and workflows are finalized.	System is intuitive and meets user expectations.
User Acceptance Testing (UAT)	Validates that the system meets business requirements and is ready for deployment.	End-to-end workflows. Critical business processes.	Functional and integration testing are complete. UAT environment is ready.	Stakeholders approve the system for deployment.
Edge Case Testing	Validates the system's behaviour under unusual or extreme conditions.	Invalid inputs. Uncommon user behaviours.	Functional testing is complete.	System handles edge cases without breaking.
Risk-Based Testing	Prioritizes testing efforts based on the potential impact and likelihood of failures.	High-risk workflows like login, authentication, and third-party integrations.	Risk assessment is complete and test cases are prioritized.	High-risk areas are validated with minimal critical defects.

Execution Strategy

b. Entry criteria

Entry Criteria	Description
Requirements Finalized	All user stories, acceptance criteria, and functional specifications are clearly defined and approved.
Test Cases Prepared	Test cases are written, reviewed, and approved, covering all functional, integration, and edge-case scenarios.
Test Environment Ready	The test environment replicates production and is configured with all necessary tools, integrations, and data.
Test Data Prepared	Required test data is created or loaded in the test environment, including valid, invalid, and edge cases.
Dependencies Resolved	All dependent systems, APIs, and third-party integrations are functional and accessible.
Code Freeze for Testing Scope	Code for the features being tested is complete, stable, and deployed to the test environment.
Defect Tracking Tool Configured	A defect management tool (e.g., JIRA) is set up, and the workflow is defined for logging, assigning, and tracking defects.
Automation Framework Ready	Test automation scripts are developed, reviewed, and ready for execution.
Team Availability	QA team, developers, and stakeholders are available for defect triaging and resolution.

c. Exit criteria

Exit Criteria	Description
All Planned Test Cases Executed	All functional, integration, performance, and regression test cases have been executed.
Critical Defects Resolved	All critical and high-severity defects are resolved, and workarounds (if any) are documented.
Defect Closure Rate	The defect closure rate meets the agreed-upon threshold, with no open critical issues.
Test Results Reviewed and Approved	Test execution results are reviewed and approved by stakeholders, ensuring acceptance criteria are met.
Regression Tests Passed	All regression test cases have been executed successfully to ensure no adverse effects from recent changes.
Performance Benchmarks Met	The system meets the defined performance benchmarks, such as response times and scalability requirements.
Security Validation Complete	Security tests are complete, with no critical vulnerabilities present.
User Acceptance Testing (UAT) Completed	Stakeholders and end-users have validated the system, and UAT sign-off has been obtained.
Test Artifacts Delivered	All testing documentation, including test cases, reports, defect logs, and metrics, is delivered and archived.
Deployment Readiness Confirmed	The system is deemed stable, with no blocking issues, and is ready for implementation.

d. Validation and Defect Management

To ensure the user stories are fully validated, the test cases and scenarios will follow a structured validation process:

i. **Execution of Test Cases:**
   1. All test cases associated with the user stories will be executed in the planned testing cycles.
   2. Test cases will be executed sequentially, starting with high-priority and high-risk features.

ii. **Validation Process:**
   1. Validate each test case against the acceptance criteria defined in the user stories.
   2. Record the results for every test case (Pass, Fail, Blocked).
   3. For failed scenarios, capture detailed evidence such as screenshots, logs, and steps.

iii. **Edge Case Validation:**
   1. Ensure edge cases for each user story are tested, such as invalid inputs, unexpected behaviours, and boundary conditions.
   2. Document the system's response to these scenarios.

iv. **Test Artifacts:**
   1. Collect and store artifacts such as screenshots, logs, and results for test execution.
   2. Maintain these artifacts in a centralized repository for review and audit purposes.

v. **Review and Sign-Off:**
   1. The test manager will review the executed test cases to ensure completeness.
   2. Stakeholders will sign off on the validated user stories upon successful testing.

vi. **Defect Logging:**
   1. All defects identified during the execution of test cases will be logged in a **Defect Tracker** (e.g. JIRA).
   2. Each defect entry must include:
   3. **Defect ID**: A unique identifier for tracking.
   4. **Summary**: Brief description of the defect.
   5. **Steps to Reproduce**: Detailed steps to replicate the issue.
   6. **Expected vs. Actual Results**: Describe the deviation from expected behaviour.
   7. **Severity**: Categorized as Critical, High, Medium, or Low.
   8. **Priority**: Assigned based on the defect's business impact.
   9. **Attachments**: Screenshots or logs as evidence.
   10. **Environment**: Specify where the defect was identified (e.g., browser, mobile, staging).

vii. **Defect life-cycle:**
   1. **New:** Logged by the tester and awaiting triage.
   2. **Assigned:** Assigned to a developer for resolution.

3. **In Progress:** Developer is actively working on the fix.
4. **Resolved:** Fix implemented and ready for retesting.
5. **Closed:** Tester confirms resolution and closes the defect.
6. **Reopened:** If the fix does not resolve the issue, the defect is reopened.
7. **Rejected:** a bug report that has been marked as not a defect by the person responsible for verifying it.

Defect management

Defect Logging:

1. All defects identified during the execution of test cases will be logged in a **Defect Tracker** (e.g. JIRA).
2. Each defect entry must include:
3. **Defect ID**: A unique identifier for tracking.
4. **Summary**: Brief description of the defect.
5. **Steps to Reproduce**: Detailed steps to replicate the issue.
6. **Expected vs. Actual Results**: Describe the deviation from expected behaviour.
7. **Severity**: Categorized as Critical, High, Medium, or Low.
8. **Priority**: Assigned based on the defect's business impact.
9. **Attachments**: Screenshots or logs as evidence.
10. **Environment**: Specify where the defect was identified (e.g., browser, mobile, staging).

Defect life-cycle:

1. **New:** Logged by the tester and awaiting triage.
2. **Assigned:** Assigned to a developer for resolution.
3. **In Progress:** Developer is actively working on the fix.
4. **Resolved:** Fix implemented and ready for retesting.
5. **Closed:** Tester confirms resolution and closes the defect.
6. **Reopened:** If the fix does not resolve the issue, the defect is reopened.
7. **Rejected:** a bug report that has been marked as not a defect by the person responsible for verifying it.

Severity	Impact
1 (Critical)	Functionality is blocked and no testing can proceed Application/program/feature is unusable in the current state
2 (High)	Functionality is not usable and there is no workaround but testing can proceed
3 (Medium)	Functionality issues but there is workaround for achieving the desired functionality
4 (Low)	Unclear error message or cosmetic error which has minimum impact on product use.

Environment Requirements

Environments

Environment	Purpose	Key Features	Requirements
QA Environment	Primary environment for functional, regression, and performance testing.	Mirrors production setup. Preloaded with test data for functional validation.	Application deployed in a staging setup. Secure database with anonymized test data.
UAT Environment	Used by stakeholders and end-users for acceptance testing.	Mirrors production with realistic scenarios and data. Stable and bug-free environment.	Complete feature set with resolved critical defects. Data reflecting production-like scenarios.
Development (Dev) Environment	Used by developers for initial code integration and unit testing.	Quick deployment cycles for testing early-stage builds.	Access to debugging tools and logs. Minimal test data setup.
System Integration Environment	Validate interactions between modules and external systems (e.g., third-party services).	Integration with external APIs, databases, and notifications.	Fully functional external APIs and simulated responses for testing.
Mobile Environment	Validate application behaviour on mobile devices (Android & iOS).	Device compatibility testing for different screen sizes and OS versions.	Physical devices or emulators for Android and iOS. Support for browser and native app testing.

Dependencies

Dependency	Description	Impact	Mitigation Strategy
Test-Item Availability	Features and functionalities must be developed and deployed to the test environment before testing.	Delayed feature development may impact the testing schedule.	Prioritize feature development based on testing needs; conduct partial testing for available features.
Test Data	Availability of valid, invalid, and edge-case data for executing test scenarios.	Lack of prepared data may block test case execution.	Preload test data or create reusable data sets for testing scenarios.
Test Environments	QA, UAT, Integration, and Mobile environments must be stable and configured.	Unstable or unavailable environments may delay test execution.	Ensure early setup and validation of environments; use mock services for unavailable systems.
Third-Party Services	Integration with services for API testing and corporate registration validation.	Downtime or unavailability of third-party services can block integration testing.	Use mock APIs or simulate responses for testing until services are available.
Development Support	Availability of developers to resolve defects identified during testing.	Delayed defect resolution may impact timelines.	Schedule dedicated developer support during active testing cycles.
Test Tools	Access to tools like JIRA, Appium, Postman	Lack of tools or licenses can hinder test execution and defect tracking.	Ensure all tools and licenses are procured and configured before testing begins.
Deadlines	Testing must align with sprint timelines and release deadlines.	Insufficient time for testing may lead to incomplete coverage or unresolved issues.	Plan buffer time for defect resolution; prioritize critical tests in early cycles.
UAT Schedule	Stakeholder availability for User Acceptance Testing.	Delays in UAT sign-off may impact deployment schedules.	Schedule UAT in advance; involve stakeholders early in test cycles for feedback.

Risk-Based Testing (RBT)

Risk-Based Testing (RBT) prioritizes testing efforts based on the potential impact and likelihood of failure for each feature or user story. This ensures that high-risk areas receive the most attention during testing, optimizing resource utilization and minimizing risks to the system.

Key Components of Risk-Based Testing

Risk Assessment:

Each feature or user story is analysed to determine its risk level based on:

- **Impact:** The effect of a failure on business operations, users, or compliance.
- **Likelihood:** The probability of a defect occurring, influenced by complexity, dependencies, and historical data.

Prioritization:

Features are categorized into High, Medium, and Low risk levels, and testing efforts are prioritized accordingly.

Focus Areas:

High-risk features undergo thorough testing, including functional, edge case, and regression scenarios.

Medium and low-risk features are tested based on time and resource availability.

Risk Categorization

Risk Level	Description
High	Features critical to business operations or user experience with high failure likelihood.
Medium	Features important to workflows but with a moderate impact or likelihood of failure.
Low	Features with minimal business impact or unlikely to fail.

Application of RBT to User Stories

User Story	Risk Level	Rationale	Testing Approach
User Login from a New Device	High	Login is critical for accessing the system. Issues can affect security and user experience.	1.Test positive and negative login scenarios. 2. Validate session management. Simulate OTP delays and retries.
Premium Account Upgrade	Medium	Important for premium users but has limited impact on core functionality.	1.Validate eligibility criteria and document upload workflows. 2.Test edge cases for invalid uploads.
Loan Notifications	High	Approval/rejection notifications are crucial for user communication and workflow tracking.	1.Test notification triggers for approvals and rejections. 2. Simulate bulk notification scenarios.
Third-Party Integration	High	Integration issues can block corporate registration validation, affecting business workflows.	1.Validate responses for valid/invalid inputs. 2. Test API timeouts and fallback mechanisms.
Departmental Approval Workflows	Medium	Important for approvals but limited direct user impact.	1.Validate approval processes across departments. 2. Test concurrent approval scenarios.
UI Consistency and Minor Enhancements	Low	Minimal business impact; mostly aesthetic.	1.Perform cross-browser/device compatibility testing. 2. Validate visual alignment and accessibility.

Test Coverage by Risk Level

Risk Level	Coverage Approach
High	1. 100% test coverage. 2. Includes functional, integration, edge cases, and regression testing.
Medium	1. 80% test coverage. 2.Focus on critical functional paths and edge cases.
Low	1. 50-70% test coverage. 2. Prioritize based on available time and resources, focusing on core functionality.

Risk Review and Adjustments

- **Periodic Reviews**: Risks are reviewed at the start of each sprint and adjusted based on development progress and findings.
- Risk Metrics:
    - **Defect Density by Risk Level**: Tracks the number of defects in high, medium, and low-risk areas.
    - **Time Spent on Risk Levels**: Ensures balanced focus across risk categories.

Testcases

User Story 1: User Login from a New Device

Test Case ID	Scenario	Steps	Expected Outcome	Preconditions
TC-001	Successful login from a new device	1. Navigate to the login page. 2. Enter valid phone number and password. 3. Enter OTP. 4. Click "Login".	User logs in and is redirected to the home page. Notification is sent.	User account exists, and OTP is valid.
TC-002	Incorrect password	1. Enter valid phone number. 2. Enter incorrect password. 3. Click "Login".	Error message: "Incorrect password. Please try again."	User account exists.
TC-003	Expired OTP	1. Request OTP. 2. Wait for OTP to expire. 3. Enter expired OTP. 4. Click "Login".	Error message: "OTP expired. Please request a new OTP."	OTP is generated but expired.
TC-004	Multiple failed OTP attempts	1. Enter valid phone number and password. 2. Enter invalid OTP multiple times. 3. Click "Login".	Account is temporarily locked after 3 invalid attempts.	OTP generated but not used correctly.
TC-005	Logout from other devices	1. Log in from Device A. 2. Log in from Device B. 3. Observe Device A's session.	Session on Device A is terminated.	User logged in from multiple devices

User Story 2: Premium Account Upgrade

Test Case ID	Scenario	Steps	Expected Outcome	Preconditions
TC-006	Select eligibility criteria	1. Navigate to "Upgrade to Premium". 2. Select eligibility criteria. 3. Click "Next".	User is redirected to the document upload page.	User is a verified individual.
TC-007	Document upload with valid file	1. Upload a valid document. 2. Click "Submit".	Success message: "Document uploaded successfully."	File meets format and size requirements.
TC-008	Document upload with invalid file	1. Upload an unsupported file format. 2. Click "Submit".	Error message: "Invalid file format. Please upload a valid file."	File format is unsupported.
TC-009	Compliance approval notification	1. Submit documents. 2. Wait for compliance approval. 3. Observe notifications.	Compliance team receives approval notification.	Documents uploaded successfully.
TC-010	Session timeout during document upload	1. Log in. 2. Start uploading a document. 3. Wait for session timeout. 4. Submit.	Error message: "Session expired. Please log in again."	User logged in but session expired.

User Story 3: Loan Notifications

Test Case ID	Scenario	Steps	Expected Outcome	Preconditions
TC-011	Loan approval notification	1. Approve a loan request. 2. Verify SMS notification.	Loan requester receives an SMS: "Your loan request is approved."	Loan request is approved.
TC-012	Loan rejection notification	1. Reject a loan request. 2. Provide rejection reasons. 3. Verify SMS notification.	Loan requester receives SMS: "Your loan request is rejected: [Reasons]."	Loan request is rejected.
TC-013	Bulk notifications	1. Approve/reject multiple loan requests. 2. Verify notifications.	Notifications are sent without delay.	Multiple loan requests exist.
TC-014	Invalid contact data	1. Attempt to send notification to a user with invalid contact details.	Error logged: "Notification delivery failed due to invalid contact details."	Loan request exists but contact is invalid.

User Story 4: Third-Party Integration And Departmental Approval Workflows

Test Case ID	Scenario	Steps	Expected Outcome	Preconditions
TC-015	Valid registration number	1. Enter valid registration number. 2. Submit for validation.	System returns: "Registration number is valid."	Registration number exists.
TC-016	Invalid registration number	1. Enter invalid registration number. 2. Submit for validation.	System returns: "Invalid registration number. Please re-enter."	Registration number does not exist.
TC-017	API timeout	1. Submit registration number. 2. Simulate API timeout.	Error message: "Service unavailable. Please try again later."	API service is temporarily unavailable.
TC-018	Operational duration check	1. Enter valid registration number. 2. Submit. 3. Verify company start date.	"<2 years: Notify user." "≥2 years: Proceed to next step."	Valid registration number exists.
TC-019	Single department approval	1. Submit loan request. 2. Approve in a single department.	Request is marked as approved in the department.	Loan request is in review.
TC-020	Concurrent approvals	1. Submit loan request. 2. Approve in multiple departments concurrently.	Approval timestamps recorded for each department.	Loan request is in review.
TC-021	Approval tracking	1. Submit a loan request. 2. View approval history.	Approval details (e.g., approver name, timestamp) are displayed.	Loan request is partially approved.
TC-022	Conflicting approvals	1. Approve requests in one department. 2. Reject in another. 3. Verify final status.	Approval status reflects combined actions with clear history.	Loan request is in review.

Test Run

To ensure the system functions correctly and provides a seamless user experience across multiple mobile devices, browsers, and operating systems.

- Devices:
  - Android (e.g., Samsung Galaxy, Google Pixel, OnePlus)
  - iOS (e.g., iPhone SE, iPhone 13, iPad)
- Browsers:
  - Mobile Chrome, Safari, Firefox, Edge
- Operating Systems:
  - Android versions 10, 11, 12
  - iOS versions 14, 15, 16

Test Run 1: User Login from a New Device

Test Run ID	Associated User Story	Test Cases	Execution Date	Executed By	Status
TR-001	User Login from a New Device	TC-001, TC-002, TC-003, TC-004, TC-005	YYYY-MM-DD	QA Engineer	In Progress

Test Run 2: Premium Account Upgrade

Test Run ID	Associated User Story	Test Cases	Execution Date	Executed By	Status
TR-002	Premium Account Upgrade	TC-006, TC-007, TC-008, TC-009, TC-010	YYYY-MM-DD	QA Engineer	Not Started

Test Run 3: Loan Notifications

Test Run ID	Associated User Story	Test Cases	Execution Date	Executed By	Status
TR-003	Loan Notifications	TC-011, TC-012, TC-013, TC-014	YYYY-MM-DD	QA Engineer	Not Started

Test Run 4: Third-Party Integration And Departmental Approval Workflows

Test Run ID	Associated User Story	Test Cases	Execution Date	Executed By	Status
TR-004	Third-Party Integration And Departmental Approval Workflows	TC-015, TC-016, TC-017, TC-018, TC-019, TC-020, TC-021, TC-022	YYYY-MM-DD	QA Engineer	Not Started