Test Strategy Document

UK Banking & Financial Solution Application

1. Introduction

This document defines the overall test strategy for the UK Banking & Financial Solution Application. The purpose is to ensure the application meets functional, security, performance, and compliance requirements in alignment with UK FCA (Financial Conduct Authority), PSD2 (Payment Services Directive 2), and GDPR regulations.

PSD2 Overview: PSD2 is a UK/EU regulation designed to make online payments more secure, enable third-party financial service providers via secure APIs, and encourage innovation in the banking sector. It mandates Strong Customer Authentication (SCA) and secure data sharing through open banking API.

2. Scope

2.1 In-Scope

- Functional testing of retail, corporate, and investment banking modules.
- API and integration testing with payment gateways, credit bureaus, and financial data providers.
- Regression Testing
- Accessibility, Cross browser & responsiveness testing
- Regulatory compliance testing (FCA, PSD2, GDPR).
- Security testing for authentication, authorization, and data encryption.
- Performance, load, and stress testing.
- Penetration testing

PSD2 Compliance Checklist:

1. Strong Customer Authentication (SCA): Validate 2FA or MFA during payments.
2. Secure API Access: Verify OAuth2 or similar token-based authentication for third-party apps.

3. Consent Management: Ensure customers can give and revoke consent to third parties.
4. Transaction Integrity: Confirm data is not altered in transit.
5. Timeouts & Session Management: Enforce time-based session expiry.
6. Audit Logs: Ensure all access and transactions are logged for compliance.

2.2 Out-of-Scope

- Third-party system internal testing outside integration points.
- Hardware testing not related to application deployment.

3. Test Objectives

- Validate all functional requirements as per business specifications.
- Ensure compliance with UK banking regulations.
- Validate system security against OWASP (Open web application security project) Top 10 vulnerabilities.
- Confirm system stability and scalability under expected transaction loads.
- Ensure compatibility with supported browsers, devices, and operating systems.

4. Test Design Approach

- **Select Design Techniques**
  - **Equivalence Partitioning** – Grouping test cases to reduce redundancy.
  - **Boundary Value Analysis** – Focusing on limits of input ranges.
  - **Decision Table Testing** – For complex business rules.
  - **State Transition Testing** – For workflows and transaction states.
  - **Use Case Testing** – Based on real-world user interactions.
- **Create Test Scenarios**
  - High-level functional flows.
  - Include cross-functional and end-to-end process scenarios.
- **Write Detailed Test Cases**
  - Steps, expected results, preconditions, test data.
  - Ensure **reusability** and **clear pass/fail criteria**.
- **Review & Sign-off**
  - Peer review by QA team.
  - Business Analyst validation for requirement coverage.
  - Automation team review for script readiness.
- **Traceability**
  - Maintain **Requirement Traceability Matrix** linking requirements to scenarios and cases.
  - Ensure coverage gaps are visible and addressed.
- **Version Control**
  - Store test cases in a centralized, version-controlled repository.

- Maintain historical records for audits (important in BFSI).

5. Test Execution Approach

5.1 RACI Matrix for Testing Levels

Testing Levels	Responsible (R)	Accountable (A)	Consulted (C)	Informed (I)
Unit Testing	Developer	Developer	Business Analyst	Test Manager, Test Lead, QA/Tester, End User
Integration Testing	QA/Tester, Developer	Test Lead	Test Manager, Business Analyst	End User
System Testing	QA/Tester	Test Lead	Test Manager, Developer, Business Analyst	End User
Acceptance Testing	QA/Tester, End User	End User	Test Lead, Business Analyst	Test Manager, Developer
Regression Testing	QA/Tester	Test Lead	Test Manager	Developer, Business Analyst, End User
Performance Testing	Performance Tester	Test Lead	Test Manager	Developer, Business Analyst, End User
Security Testing	QA/Tester	Test Lead	Test Manager	Developer, Business Analyst, End User
Compliance Testing	QA/Tester	Test Lead	Business Analyst	Test Manager, Developer, End User
Penetration Testing	Security Tester / External Pen Test Vendor	Test Lead / Security Manager	Test Manager, Business Analyst	Developer, End User
API Testing	QA/API Tester	Test Lead	Test Manager	Developer, Business Analyst, End User

5.2 Testing Types

The following testing types will be executed to ensure complete coverage of functional and non-functional aspects:

- **Functional Testing** – Validates that the application behaves according to business requirements and specifications.
- **Regression Testing** – Ensures that new code changes do not introduce defects into previously working functionality.
- **API Testing** – Verifies that all internal and external APIs function correctly, return accurate data, and handle errors gracefully.
- **Security Testing** – Identifies vulnerabilities, ensures data protection, and verifies compliance with OWASP Top 10 security standards.
- **Performance Testing** – Evaluates system responsiveness, scalability, and stability under normal and peak load conditions.
- **Accessibility Testing** – Ensures compliance with WCAG (Web Content Accessibility Guidelines) so the application is usable by individuals with disabilities.
- **Cross-Browser Testing** – Confirms consistent functionality and UI rendering across supported browsers and devices.
- **Compliance Testing** – Validates adherence to industry regulations such as FCA, PSD2, and GDPR.

5.3 Methodology

A risk-based testing approach will be adopted, prioritizing test execution based on the likelihood and potential business impact of defects.

- High-risk areas (e.g., payment processing, customer data security) will receive maximum coverage and early test execution.
- Testing activities will follow an iterative process, with continuous feedback loops between development, QA, and business teams.
- Agile principles will be followed where applicable, allowing early defect detection and quicker adaptation to changes in requirements.

5.4 Defect Management

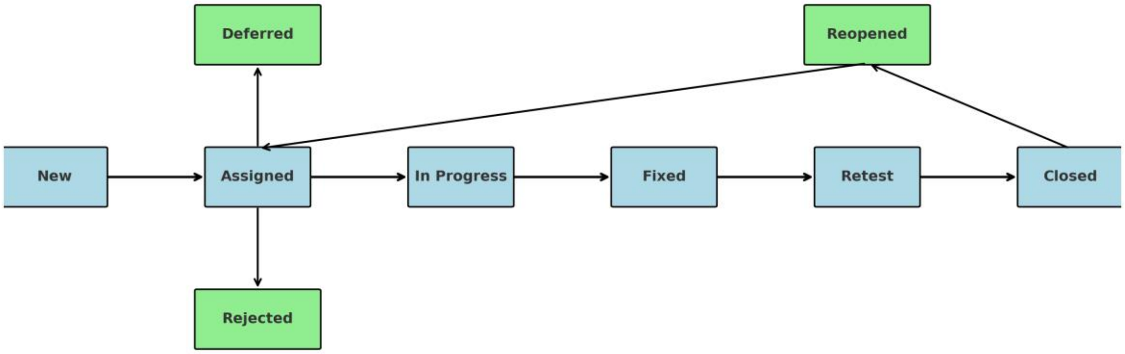
Defect management will be performed using standardized processes to ensure transparency and traceability:

- All defects will be logged and tracked in a defect/test management tool such as **JIRA, TestRail, DevPlus, or Jile**.
- Defects will be triaged based on **severity and priority**, ensuring critical issues are addressed first.
- Defect lifecycle stages (New → Assigned → In Progress → Fixed → Retested → Closed) will be strictly followed.
- Detailed defect reports will be shared with stakeholders during daily stand-ups and weekly status meetings.

5.4.1 Defect Severity & Priority Definitions

Severity	Description	Priority	Description
Critical	System crash, major functionality unavailable,s no workaround possible.	**P1 – Urgent**	Must be fixed immediately; blocks testing or production.
High	Major functionality broken but workaround exists; impacts business significantly.	**P2 – High**	Fix in the current or next build; high business impact.
Medium	Minor functionality issue; workaround available; limited business impact.	**P3 – Medium**	Fix before project closure; moderate impact.
Low	Cosmetic/UI issues or enhancements; no impact on functionality.	**P4 – Low**	Fix if time permits; lowest priority.

5.4.2 Defect lifecycle stages



Stage	Responsible	Description
New	Tester	Identifies a defect and logs it with full details, screenshots, and steps to reproduce.
Assigned	Test Lead / Project Manager	Assigns defect to the relevant developer or development team for resolution.
In Progress	Developer	Starts working on fixing the defect.
Fixed	Developer	Marks the defect as fixed and provides the build for testing.
Retest	Retest	Verifies the fix in the specified build to ensure the defect no longer occurs.

Closed	Retest	Confirms the defect is resolved and no related issues exist; marks it as closed.
Reopened	Retest	Reopens defect if it reoccurs after being marked fixed or closed.
Deferred	Project Manager / Product Owner	Postpones defect for a future release due to low impact or constraints.
Rejected	Developer / Test Lead	Marks defect as invalid or not reproducible.

5.5 Automation

Automation will be used to improve efficiency and ensure repeatable, consistent testing:

- **Regression and Smoke Testing** – Automated using **Selenium WebDriver with Java** and integrated with TestNG for execution management.
- **API Testing** – Automated using **Postman** and command-line execution via **Newman**, with assertions for both functional and negative test cases.
- **Performance Testing** – Conducted using **Apache JMeter** to simulate concurrent user loads and monitor system response times.
- Automation scripts will be version-controlled in a Git repository, with CI/CD integration for scheduled or on-demand execution.

5.6 Test Data

Test data will be carefully managed to ensure both relevance and compliance:

- **Masked Production Data** – Real production data will be anonymized to remove personally identifiable information (PII) in compliance with **GDPR**.
- **Synthetic Test Data** – Artificially generated data sets will be created for edge cases, negative testing, and scenarios not found in production.
- Test data refresh cycles will be defined to ensure accuracy and consistency across test environments.
- Secure storage and controlled access policies will be applied to all test data.

6. Test Environment

- **Environments**: Dev, SIT, UAT, Pre-Prod, Prod.
- **Configurations**: Windows/Linux/Mac servers, MS SQL database, AWS cloud hosting.
- **Access Controls:** Role-based access; only authorized users allowed in restricted test environments.

- **Data Management:** GDPR-compliant anonymization for customer data.

7. Testing Tools

Category	Tools
Test Management	JIRA DevPlus TestRail BrowserStack
Automation Testing	Intellij Idea / VS code editor Selenium WebDriver TestNG
API Testing	Postman RestAssured
Performance	Apache JMeter
Security Testing	OWASP ZAP, Burp Suite
Reporting	Confluence Power BI dashboards MS Office (Word, Excel, Power Point etc.)
Accessibility	Wave Lighthouse report
Cross Browser / Responsiveness Testing	BrowserStack
MI Report	Google analytics

8. Risk Management

8.1 Risks

- Tight timelines impacting regression coverage.
- Late changes in regulatory requirements.
- Integration dependency delays.

8.2 Mitigation

- Early engagement with compliance team.
- Incremental regression testing.
- Daily sync with integration partners.
- Buffer time in test schedule.

9. Roles & Responsibilities

Role	Responsibilities
Test Manager	Define strategy, manage schedules, oversee reporting
Test Leads	Plan, assign, and monitor execution.
Test Engineers / QA	Design, execute, and log defects.
Automation Test Engineers	Develop and maintain automated suites.
Security Testers	Conduct penetration and vulnerability testing.
Business Analysts	Provide clarification on requirements
Project Manager	Oversee project delivery, manage timelines, budgets, and stakeholder communication.
Scrum Master	Facilitate Agile ceremonies, remove impediments, and ensure Scrum practices are followed.

10. Test Metrics & Reporting

10.1 Metrics

- Test case execution rate.
- Pass/fail percentage.
- Defect density.
- Defect leakage rate.
- Test coverage %.

10.2 Reporting

- Daily execution status.
- Weekly defect summary.
- Final test closure report.

11. Test Schedule

11.1 High-Level Timeline

Phase	Planned Date	Description
Test Planning	From-To Date	Define scope, approach, timelines, and resource allocation.
Test Design	From-To Date	Create and review test cases, scenarios, and test data.
Test Execution	From-To Date	Run test cases, log defects, and retest fixes.

Regression	From-To Date	Assist business users during User Acceptance Testing.
UAT Support	From-To Date	Assist business users during User Acceptance Testing.
Non-Functional Testing	From-To Date	Validate usability, scalability, and reliability aspects.
Security Testing	From-To Date	Identify vulnerabilities and validate security controls.
Penetration Testing	From-To Date	Simulate real-world attacks to assess system resilience.
Performance Testing	From-To Date	Test system behavior under expected and peak load conditions.
Closure & Sign-off	From-To Date	Prepare test closure report, get formal approvals, and archive artifacts.

12. Entry & Exit Criteria

12.1 Entry criteria

- Approved **Business Requirement Document (BRD)** or **User Stories** are available.
- Test environment is set up and accessible.
- Test data is prepared and meets GDPR compliance.
- Test cases and scenarios are reviewed and signed off.
- Required tools (e.g., JIRA, Selenium, Postman, JMeter) are installed and configured.
- All necessary access permissions are granted to the QA team.
- No critical open defects blocking test execution from earlier phases.

12.2 Exit Criteria

- All **planned test cases** have been executed.
- All **high-severity defects** are closed or have approved workarounds.
- Regression testing is completed with acceptable results.
- Non-functional tests (performance, security, compliance) meet agreed benchmarks.
- UAT feedback is incorporated and retested.
- Test closure report is prepared and approved by stakeholders.
- Regulatory compliance validation (FCA, PSD2, GDPR) is passed.
- All deliverables (test cases, defect logs, reports) are archived.

13. Approval

Prepared By: Rohini Choudhary, QA Manager, August 2025
Reviewed By: [Name, Role, Date]
Approved By: [Name, Role, Date]
Signatures: ___________________________

14. Glossary of Terms

Term	Definition
FCA	Financial Conduct Authority – UK regulator for financial services.
PSD2	Payment Services Directive 2 – EU/UK regulation for secure payments and open banking APIs.
GDPR	General Data Protection Regulation – EU/UK data protection and privacy law.
SCA	Strong Customer Authentication – PSD2 requirement for multi-factor authentication.
OWASP	Open Web Application Security Project – organization providing security best practices and the OWASP Top 10 list.
UAT	User Acceptance Testing – final testing phase where end users validate the system.
API	Application Programming Interface – allows systems to communicate with each other.
SIT	System Integration Testing – testing combined components as a whole.
Defect Density	Number of defects per size of code or functionality tested.
Test Coverage	Percentage of requirements or code tested.
BFSI	Banking, Financial Services, and Insurance.