### **Exhaustive API Test Cases: Payment Uploader - Payment Module**

**Objective:** To provide an exhaustive and highly detailed suite of API test cases for the System Integration Testing (SIT) of the Payment Module. These test cases are designed to validate every aspect of the API's functionality, including data validation, business logic, status transitions, security, and integration with downstream systems.

#### **1. User & Authentication (/user/auth)**

| **Test Case ID** | **API Endpoint & Method** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- | --- |
| **AUTH-001** | POST /user/auth/login | **Happy Path: Login as Maker** | { "username": "paymentMaker" } | 200 OK | Returns a JSON object with a data.userToken (JWT). |
| **AUTH-002** | POST /user/auth/login | **Happy Path: Login as Checker** | { "username": "paymentChecker" } | 200 OK | Returns a JSON object with a data.userToken (JWT). |
| **AUTH-003** | POST /user/auth/login | **Negative: Login with Non-existent User** | { "username": "invalidUser" } | 404 Not Found | Error message: "User Not Found". |
| **AUTH-004** | POST /user/auth/login | **Validation: Blank Username** | { "username": "" } | 400 Bad Request | Error message: "username cannot be empty." |
| **AUTH-005** | GET /user/auth/current | **Happy Path: Get Current User Permissions** | **Header:** user-token: {{maker\_access\_token}} | 200 OK | Returns a JSON array of menu permissions for the Maker role. |
| **AUTH-006** | GET /user/auth/current | **Security: Get Permissions with Invalid Token** | **Header:** user-token: invalidtoken | 401 Unauthorized | Standard unauthorized error response. |

#### **2. Standing Order Management (/StandingOrder)**

##### **2.1. Create Standing Order (POST /StandingOrder)**

| **Test Case ID** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- |
| **SO-POST-001 - 031** | **Happy Path: Create SO for All 31 Billers** | For each of the 31 billers in Detail biller SYB.csv:<br>{ "productCode": "...", "billingId": "...", "accountNumberDebit": "...", "accountNumberCharge": "..." } | 200 OK | Returns the newly created Standing Order object with approvalStatus: "PendingApproval". |
| **SO-POST-032** | **Happy Path: accountNumberCharge is auto-filled** | { ..., "accountNumberDebit": "ACCT\_A", "accountNumberCharge": null } | 200 OK | The created SO object in the DB has accountNumberCharge set to "ACCT\_A". |
| **SO-POST-033** | **Validation: Missing billingId** | { "productCode": "PDAM-BTM", "accountNumberDebit": "..." } | 400 Bad Request | Error message: "billingId is a required field." |
| **SO-POST-034** | **Validation: Missing productCode** | { "billingId": "...", "accountNumberDebit": "..." } | 400 Bad Request | Error message: "productCode is a required field." |
| **SO-POST-035** | **Validation: Missing accountNumberDebit** | { "productCode": "PDAM-BTM", "billingId": "..." } | 400 Bad Request | Error message: "accountNumberDebit is a required field." |
| **SO-POST-036** | **Validation: Empty billingId** | { "billingId": "", ... } | 400 Bad Request | Error message: "billingId cannot be empty." |
| **SO-POST-037** | **Validation: Invalid Data Type for startDebitDate** | { "startDebitDate": "abc", ... } | 400 Bad Request | Error message: "startDebitDate must be a valid number." |
| **SO-POST-038** | **Validation: Out of Range startDebitDate** | { "startDebitDate": 32, ... } | 400 Bad Request | Error message: "startDebitDate must be between 1 and 31." |
| **SO-POST-039** | **Integration: Invalid productCode** | { "productCode": "INVALID-CODE", ... } | 404 Not Found | Error message: "Product with code 'INVALID-CODE' not found." |
| **SO-POST-040** | **Integration: Invalid billingId (Biller Rejects)** | { "billingId": "999999999", ... } | 400 Bad Request | Error message from Biller Aggregator: "Billing ID not found." |
| **SO-POST-041** | **Integration: Invalid accountNumberDebit (OVS-BAS)** | { "accountNumberDebit": "DORMANT-ACCT", ... } | 400 Bad Request | Error message: "Account number is invalid or dormant." |
| **SO-POST-042** | **Integration: Biller Aggregator Timeout** | (Configure simulator to not respond) | 504 Gateway Timeout | Error message: "The request to the biller aggregator timed out." |
| **SO-POST-043** | **Integration: OVS-BAS Timeout** | (Configure simulator to not respond) | 504 Gateway Timeout | Error message: "The request to the core banking system timed out." |
| **SO-POST-044** | **Security: Unauthorized (No Token)** | (No user-token header) | 401 Unauthorized | Standard unauthorized error response. |
| **SO-POST-045** | **Security: Forbidden (Checker Role)** | **Header:** user-token: {{checker\_access\_token}} | 403 Forbidden | Error message: "User does not have permission to create this resource." |

##### **2.2. Get Standing Orders (GET /StandingOrder)**

| **Test Case ID** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- |
| **SO-GET-001** | **Happy Path: Get All Standing Orders** | (No parameters) | 200 OK | Returns a JSON array of all standing order objects within an Items array. |
| **SO-GET-002** | **Filtering: Get by ApprovalStatus** | ?ApprovalStatus=Approved | 200 OK | Returns a JSON array of SOs only with "Approved" status. |
| **SO-GET-003** | **Filtering: Get by ApplicationStatus** | ?ApplicationStatus=Amendment | 200 OK | Returns a JSON array of SOs only with "Amendment" status. |
| **SO-GET-004** | **Filtering: Get by AccountNumberDebit** | ?AccountNumberDebit=... | 200 OK | Returns a JSON array of SOs matching the account number. |
| **SO-GET-005** | **Filtering: Get by ProductCode** | ?ProductCode=PDAM-BTM | 200 OK | Returns a JSON array of SOs for the specified product. |
| **SO-GET-006** | **Filtering: Get by Date Range** | ?ModifiedDate1=...&ModifiedDate2=... | 200 OK | Returns SOs updated within the specified date range. |
| **SO-GET-007** | **Filtering: Combined Filters** | ?ApprovalStatus=Approved&ProductCode=PDAM-BTM | 200 OK | Returns a JSON array of SOs that are both "Approved" AND for the specified product. |
| **SO-GET-008** | **Filtering: No Results** | ?BillingId=NONEXISTENT | 200 OK | Returns an Items array that is empty []. |
| **SO-GET-009** | **Pagination: Get First Page** | ?Page=1&Size=5 | 200 OK | Returns the first 5 SOs. Response body should contain totalItems. |
| **SO-GET-010** | **Pagination: Get Second Page** | ?Page=2&Size=5 | 200 OK | Returns SOs 6 through 10. |
| **SO-GET-011** | GET /StandingOrder/:id | **Happy Path: Get SO by ID** | id: Valid SO ID | 200 OK |
| **SO-GET-012** | GET /StandingOrder/:id | **Negative: Get Non-Existent SO** | id: 99999 | 404 Not Found |
| **SO-GET-013** | GET /StandingOrder/:id | **Validation: Invalid ID format** | id: abc | 400 Bad Request |

##### **2.3. Update/Amend/Terminate/Delete Standing Order**

| **Test Case ID** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- |
| **SO-PUT-001** | **Happy Path: Amend an Approved SO** | PUT /StandingOrder/:id<br>id: Valid Approved SO ID<br>{ "accountNumberDebit": "NEW-ACCT", "applicationStatus": "amend" } | 200 OK | Returns the updated SO object. approvalStatus is now "PendingApproval". applicationStatus is "Amendment". |
| **SO-PUT-002** | **Negative: Update a Non-Existent SO** | PUT /StandingOrder/99999 | 404 Not Found | Error message: "Standing Order with ID 99999 not found." |
| **SO-PUT-003** | **Security: Attempt to Update a Submitted SO** | PUT /StandingOrder/:id<br>id: Valid Submitted SO ID | 400 Bad Request | Error message: "Can't update data because approval is pending." |
| **SO-TERM-001** | **Happy Path: Terminate an Approved SO** | PUT /StandingOrder/:id<br>id: Valid Approved SO ID<br>{ "applicationStatus": "terminate", "finalDebit": "2025-12" } | 200 OK | Returns the updated SO object. approvalStatus is now "PendingApproval". applicationStatus is "Terminate". |
| **SO-DEL-001** | **Happy Path: Delete a Submitted SO** | DELETE /StandingOrder/:id<br>id: Valid Submitted SO ID<br>{ "notes": "User request" } | 204 No Content | The SO is successfully deleted. |
| **SO-DEL-002** | **Validation: Delete without mandatory notes** | DELETE /StandingOrder/:id<br>id: Valid Submitted SO ID<br>{} | 400 Bad Request | Error message: "notes field is mandatory for deletion." |
| **SO-DEL-003** | **Security: Attempt to Delete Approved SO** | DELETE /StandingOrder/:id<br>id: Valid Approved SO ID | 403 Forbidden | Error message: "Approved Standing Orders cannot be deleted." |

##### **2.4. Approve/Reject Standing Order**

| **Test Case ID** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- |
| **SO-APP-001** | **Happy Path: Approve a Pending SO** | PUT /StandingOrder/:id/approved<br>id: Valid Pending SO ID<br>{ "notes": "Verified" } (optional) | 200 OK | Returns the SO object with approvalStatus: "Approved". Triggers fee booking. |
| **SO-APP-002** | **Integration: Fee Debit Fails** | PUT /StandingOrder/:id/approved<br>id: Valid Pending SO ID (for an account with insufficient funds for the fee) | 400 Bad Request | Error message: "Approval failed: Insufficient funds for registration fee." |
| **SO-APP-003** | **Status Transition: Approve an Already Approved SO** | PUT /StandingOrder/:id/approved<br>id: Valid Approved SO ID | 409 Conflict | Error message: "Standing Order is already in 'Approved' state." |
| **SO-APP-004** | **Status Transition: Approve a Rejected SO** | PUT /StandingOrder/:id/approved<br>id: Valid Rejected SO ID | 409 Conflict | Error message: "Cannot approve a 'Rejected' Standing Order. It must be resubmitted." |
| **SO-APP-005** | **Happy Path: Reject a Pending SO** | PUT /StandingOrder/:id/rejected<br>id: Valid Pending SO ID<br>{ "notes": "Incorrect account number provided." } | 200 OK | Returns the SO object with approvalStatus: "Rejected". |
| **SO-APP-006** | **Validation: Reject without mandatory notes** | PUT /StandingOrder/:id/rejected<br>id: Valid Pending SO ID<br>{} | 400 Bad Request | Error message: "notes field is mandatory for rejection." |
| **SO-APP-007** | **Security: Forbidden (Maker Role)** | PUT /StandingOrder/:id/approved<br>**Header:** user-token: {{maker\_access\_token}} | 403 Forbidden | Error message: "User does not have permission to approve." |

#### **3. Payment Execution & Transaction Management (/Transaction)**

| **Test Case ID** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- |
| **PAY-EXEC-001** | **Happy Path: Trigger Worker Job** | POST /api/payments/execute-job (Inferred endpoint) | { "executionDate": "..." } | 202 Accepted |
| **PAY-GET-001** | **Happy Path: Get All Transactions** | GET /Transaction | (No parameters) | 200 OK |
| **PAY-GET-002 - 012** | **Filtering: Get by each of the 11 PaymentStatus types** | ?PaymentStatus=Paid, ?PaymentStatus=BalanceShort, etc. | 200 OK | Returns a JSON array of transactions only with the specified status. |
| **PAY-GET-013** | **Filtering: Get by UI Status Group "Need Repair"** | ?TransactionType=Need Repair | 200 OK | Returns transactions with statuses "Overdue - Balance Short" and "Overdue - Failed". |
| **PAY-GET-014** | GET /Transaction/:id | **Happy Path: Get Transaction by ID** | id: Valid Transaction ID | 200 OK |
| **PAY-REPAIR-001** | **Happy Path: Repair a "Balance Short" Transaction** | PUT /Transaction/:id<br>id: Valid "BalanceShort" Transaction ID<br>{ "repairCharge": 5000, "notes": "..." } | 200 OK | The transaction is submitted for approval. approvalStatus becomes "PendingApproval". |
| **PAY-REPAIR-002** | **Status Transition: Attempt to Repair a "Paid" Transaction** | PUT /Transaction/:id<br>id: Valid "Paid" Transaction ID | 409 Conflict | Error message: "Cannot repair a transaction that is already 'Paid'." |
| **PAY-UPDATE-001** | **Happy Path: Update an Outstanding Transaction** | PUT /Transaction/:id<br>id: Valid Outstanding Transaction ID<br>{ "startDebitDate": "..." } | 200 OK | The transaction is submitted for approval. approvalStatus becomes "PendingApproval". |
| **PAY-UPDATE-002** | **Status Transition: Attempt to Update a "Paid" Transaction** | PUT /Transaction/:id<br>id: Valid "Paid" Transaction ID | 403 Forbidden | Error message: "Cannot update a transaction that has already been processed." |
| **PAY-APP-001** | **Happy Path: Approve a Repair Request** | PUT /Transaction/:id/approved<br>id: Valid Transaction ID pending repair approval | 200 OK | The transaction approvalStatus becomes "Approved" and the re-execution job is triggered. |
| **PAY-APP-002** | **Happy Path: Reject an Update Request** | PUT /Transaction/:id/rejected<br>id: Valid Transaction ID pending update approval<br>{ "notes": "..." } | 200 OK | The transaction approvalStatus becomes "Rejected". The original data is retained. |

#### **4. Master Data & External Services**

| **Test Case ID** | **API Endpoint & Method** | **Test Scenario** | **Request Payload / Key Parameters** | **Expected HTTP Status** | **Expected Response Body / Outcome** |
| --- | --- | --- | --- | --- | --- |
| **PROD-001** | POST /Product | **Happy Path: Create New Product** | { "productCode": "NEW-PROD", ... } | 200 OK | Returns the newly created product object with approvalStatus: "PendingApproval". |
| **PROD-002** | POST /Product | **Validation: Duplicate productCode** | { "productCode": "2411111", ... } | 409 Conflict | Error message: "Product with code '2411111' already exists." |
| **CHG-001** | POST /Charge | **Happy Path: Create New Charge** | { "chargeKey": "Standing Order Creation", ... } | 200 OK | Returns the newly created charge object with approvalStatus: "PendingApproval". |
| **DWH-001** | POST /AccountNumber | **Happy Path: Check Account Number** | { "accountNumber": "3665CADIOD5998100912" } | 200 OK | Returns account details including accountName and accountStatus. |
| **DWH-002** | POST /workingday | **Happy Path: Get Working Days** | { "startDate": "...", "endDate": "..." } | 200 OK | Returns a JSON array of working day strings. |
| **SYB-001** | POST /externals/check-billing | **Happy Path: Check SYB Billing** | { "ProductCode": "...", "BillingId": "..." } | 200 OK | Returns billing details including customerName and totalBill. |