# Regular Payment API

## Overview

The Regular Payment API accepts a machine to machine transaction containing regular payment information in the form of the subject loan number and payment amount. An Effective Backdate can also be applied to the payment. Once received the payment is processed via Strategy’s Lockbox programming using standard lockbox processing.

**NOTE:** This document outlines how the Strategy API processes regular payments and the requirements for creating the payload. Your organization will be responsible for gathering the data and creating the data in the prescribed format.

## Regular Payment Data Format Requirements

Access the Loan Entry API using http://<your-portal-domain-name>/API/ regularPayment

The API uses JSON as the markup language and the UTF-8 character set. Use camel case for naming JSON keys. The HTTP Method is POST for all actions.

The API action is declared in the JSON request body. Each API action is for a single transaction only (one payment at a time).

## Required fields for API to perform successful upload.

Loan Number and Payment Amount are required fields. Effective Backdate is optional.

Loan Number – decimal (9,0)

Payment Amount – decimal (15,2)

Effective Backdate - Date

## Process Flow

* The API gathers Header Info and establishes a data base connection.
* Determines if Strategy is available, and updates the
* If available updates the Header Log (PAPIH) Status to Received ‘R’
* Verifies that HTTP Method is correct and Authenticates the User exists and has appropriate Strategy task rights to the Regular Payment API.
* Parses the JSON request, verifying the JSON format, and validating against the schema. Creates object if valid format/schema, sends error if not.
* Processes the request by inserting a record in Lockbox file. If insert fails, respond with error, otherwise respond with success.
* Record results in the API Header.

Once a record is inserted, standard Lockbox processing for Strategy is used to edit the transaction and move to the daily transaction file for processing. The accepted transactions will show in Transaction Processing on the Change Transaction Information tab.

Review the Transactions received, accepted and rejected using the GR418A report for Lockbox. Transactions that are rejected because the transaction failed the Lockbox edits will display the reason in the Comments section of the report.

### Schema

The following object contains the schema for Loan Entry API

{

"$schema": "http://json-schema.org/draft-07/schema#",

"title": "Incoming Regular Payment",

"definitions": {

"loanNumber": {"type": "string", "pattern": "^[0-9]{1,9}?$"},

"dateFormat": {"type": "string", "format": "date"},

"paymentFormat": {"type": "string", "pattern": "^[0-9]{0,13}(\\.[0-9]{0,2})?$"}

},

"type": "object",

"properties": {

"action": {"enum": ["insert"]}

},

"allOf": [

{

"if": {

"properties": {"action": {"const": "insert"}}

},

"then": {

"properties": {

"loanNumber": {"$ref": "#/definitions/loanNumber"},

"paymentAmount": {"$ref": "#/definitions/paymentFormat"},

"effectiveBackDate": {"$ref": "#/definitions/dateFormat"}

},

"required": ["loanNumber", "paymentAmount"]

}

}

]

}

### Sample Requests

|  |  |
| --- | --- |
| Example body request without Effective Back Date | |
| Request | Response |
| {  "action": "insert",  "loanNumber":"123456789",  "paymentAmount":"123456789012345.67"  } | {  "action": "insert",  "status": "success",  "recordCount": 1,  "message": "1 records inserted"  } |
| Example body request with Effective Back Date | |
| Request | Response |
| {  "action": "insert",  "loanNumber":"123456789",  "paymentAmount":"123456789012345.67",  "effectiveBackdate":"2022-06-01"  } | {  "action": "insert",  "status": "success",  "recordCount": 1,  "message": "1 records inserted"  } |

## Headers

### Request Headers

The API request must include the following headers.

|  |  |
| --- | --- |
| X-Request-ID | Unique ID generated by client for identifying the request transaction. |
| X-Data-Library | Data library to execute the request against. |

### Response Headers

API responses Headers will contain at a minimum the following:

|  |  |
| --- | --- |
| X-Request-ID | Unique ID generated by client for identifying the request transaction. |
| X-Response-ID | Unique ID generated by API for identify the transaction. |
| X-Received-Timestamp | Timestamp for when request was received. |
| X-Response-Timestamp | Timestamp indicating when a response was created and sent. |

## Response Messages

### Successful HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| HTTP Status Code | HTTP Status Description |  |
| 200 | Ok | Successful request |
| 204 | No Content | No content is found in the request body |

### Unsuccessful HTTP Status Codes

|  |  |  |
| --- | --- | --- |
| HTTP Status Code | HTTP Status Description |  |
| 400 | Bad Request | * Malformed JSON * Missing required fields * Missing required headers |
| 401 | Unauthorized | Missing, invalid or expired values in basic authentication header |
| 405 | Method not allowed | Invalid HTTP method in request |
| 422 | Unprocessable Entity | JSON is correct and passes validation but field values make the request invalid; ex. attempting to insert a record that already exists. |
| 500 | Interval Server Error | An error with the application server or API process has occurred which should be reported to ASP Support. |
| 503 | Service Unavailable | Strategy is locked |

## Security

User ID performing the upload should be a valid Strategy user where password is not expired.

User ID should have execute authority on “Strategy APIs [W]” and Write authority on “Regular Payment API [W]” task ID (190070).

|  |  |
| --- | --- |
|  | A screenshot of a computer  Description automatically generated |