training - Bug #3002

Implement Payables UI – Vendor Payments

08/18/2026 07:53 AM - Yashaditya Singh

Status:	New	**Start date:**	08/18/2026
Priority:	Normal	**Due date:**	
Assignee:	Yashaditya Singh	**% Done:**	0%
Category:		**Estimated time:**	0.00 hour
Target version:		**Spent time:**	0.00 hour

Description

Implement the frontend UI for the Payables module, focusing on Vendor Payments.

1. Vendor Payments Listing

Create a Vendor Payments page with:

Vendor Payments table
Payment/Voucher number
Vendor name
Vendor Bill number
Payment date
Payment mode
Amount paid
Created date
View payment details
Search/filter functionality
Pagination

Supported payment modes:

CASH
BANK
UPI

The UI should consume the Vendor Bill Payments listing API and support filtering by vendor or vendor bill.

2. Record Vendor Payment

Add a Record Payment button on the Vendor Payments page.

Create a payment form containing:

Vendor selection
Payment date
Payment mode
Narration
Outstanding vendor bills
Bill-wise payment allocation
Total payment amount

The user should be able to select multiple outstanding bills belonging to the same vendor and enter the amount to be applied against each bill. The API supports multiple bill allocations within a single payment.

3. Outstanding Bills

When a vendor is selected, display the vendor's posted outstanding bills in a table.

Display:

Field Description
Bill Number Vendor bill number
Bill Date Bill date

Total Amount Original bill amount
Amount Paid Amount already paid
Balance Due Remaining amount
Payment Amount Amount to apply

Only bills that can receive payment should be selectable.

The UI should prevent the user from entering a payment amount greater than the bill's balance due. The backend also validates this condition.

4. Payment Details

After successfully recording a payment, display:

Accounting voucher number
Payment date
Vendor
Payment mode
Narration
Total payment amount
Allocated bills
Amount applied against each bill
Updated amount paid
Updated balance due
Updated payment status

Payment statuses should be displayed as:

UNPAID
PARTIALLY_PAID
PAID

The payment response contains the updated bill balance and payment status after allocation.

5. Payment History

Add a payment history section/page where users can view payments made against a specific vendor bill.

Display:

Payment/Voucher number
Bill number
Amount applied
Payment date
Created by

The existing API supports filtering payment history by financeVendorId and vendorBillId.

6. UI Validation

Implement frontend validation for:

Vendor is required.
Payment date is required.
Payment mode is required.
At least one bill must be selected.
Payment amount must be greater than zero.
Payment amount cannot exceed the bill's balance due.
Selected bills must belong to the selected vendor.
Total payment amount should automatically update based on allocations.
7. UI/UX Requirements
Follow the existing EVCSMS MUI theme.
Reuse existing table, filter, pagination, dialog and form components.
Use a consistent accounting/finance UI design.
Show loading indicators while API requests are in progress.
Show success/error notifications.
Show confirmation before recording the payment.

Disable the submit button while the payment is being submitted.
Display appropriate empty states when no outstanding bills are available.
Handle API validation and permission errors properly.
8. Required API Integration

Integrate the following Payables APIs:

Record Payment

POST /api/v1/master-data/purchasing/vendor-bill-payments

List Payments

GET /api/v1/master-data/purchasing/vendor-bill-payments

Use the existing Finance Vendor and Vendor Bill APIs for vendor selection and outstanding bill information.

Acceptance Criteria
Vendor Payments menu/page is available under Payables.
User can view existing vendor payments.
User can filter payments by vendor and vendor bill.
User can select a vendor and view its outstanding posted bills.
User can allocate payment amounts against one or multiple bills.
Payment total is calculated automatically.
Overpayment is prevented through frontend validation.
Successful payment updates the displayed payment status and balance.
Payment details can be viewed after creation.
Payment history can be viewed for a vendor bill.
Loading, validation, success and error states are handled properly.
UI follows the existing EVCSMS application theme and component structure.