use App\Models\User;

use App\Models\Transaction;

use Illuminate\Support\Facades\DB;

class balanceController extends Controller

{

//Calculate user closing balance for 90 days.

public function calculateUserClosingBalance($userId)

{

// Initialize variables for daily closing balance, total balance, and transaction count

$dailyClosingBalances = [];

$totalBalance = 5; // Initial balance

$transactionCount = 0;

// Iterate through 90 days

for ($i = 0; $i < 90; $i++) {

$currentDate = now()->subDays($i);

// Calculate transaction after balance for the current date

$totalBalance = $this->calculateTransactionAfterBalance($userId, $currentDate, $totalBalance);

// Store daily closing balance

$dailyClosingBalances[$currentDate->format('Y-m-d')] = $totalBalance;

// Count transactions for the current date

$transactionCount += Transaction::where('user\_id', $userId)

->where('trans\_date', $currentDate->format('Y-m-d'))

->count();

}

// Calculate 90 days average balance

$averageBalance = array\_sum($dailyClosingBalances) / count($dailyClosingBalances);

// Calculate average closing balance for the first 30 days and last 30 days

$first30DaysAverage = array\_slice($dailyClosingBalances, 0, 30);

$last30DaysAverage = array\_slice($dailyClosingBalances, 60, 30);

$first30DaysAverageClosing = array\_sum($first30DaysAverage) / count($first30DaysAverage);

$last30DaysAverageClosing = array\_sum($last30DaysAverage) / count($last30DaysAverage);

// Calculate last 30 days income except category id 18020004

$last30DaysIncome = $this->calculateLast30DaysIncome($userId);

// Calculate debit transaction count in the last 30 days

$debitTransactionCount = $this->calculateDebitTransactionCount($userId);

// Sum of debit transactions on Friday/Saturday/Sunday

$sumWeekendDebit = $this->sumDebitTransactionsOnWeekends($userId);

// Sum of income with transaction amount > 15

$sumIncomeAmountGreaterThan15 = $this->sumIncomeAmountGreaterThan15($userId);

// Return the results

return [

'dailyClosingBalances' => $dailyClosingBalances,

'averageBalance' => $averageBalance,

'first30DaysAverageClosing' => $first30DaysAverageClosing,

'last30DaysAverageClosing' => $last30DaysAverageClosing,

'last30DaysIncome' => $last30DaysIncome,

'debitTransactionCount' => $debitTransactionCount,

'sumWeekendDebit' => $sumWeekendDebit,

'sumIncomeAmountGreaterThan15' => $sumIncomeAmountGreaterThan15,

];

}

/\*\*

\* Calculate transaction after balance for a given user and date.

\* @param int $userId

\* @param \Carbon\Carbon $currentDate

\* @param float $totalBalance

\* @return float

\*/

private function calculateTransactionAfterBalance($userId, $currentDate, $totalBalance)

{

// Logic to fetch transactions for the user on the current date

$transactions = Transaction::where('user\_id', $userId)

->where('trans\_date', $currentDate->format('Y-m-d'))

->orderBy('created\_at')

->get();

// Iterate through transactions and calculate transaction after balance

foreach ($transactions as $transaction) {

if ($transaction->trans\_type === 'income') {

$totalBalance += $transaction->trans\_amount;

} elseif ($transaction->trans\_type === 'debit') {

$totalBalance -= $transaction->trans\_amount;

}

}

return $totalBalance;

}

/\*\*

\* Calculate last 30 days income except category id 18020004.

\* @param int $userId

\* @return float

\*/

private function calculateLast30DaysIncome($userId)

{

return Transaction::where('user\_id', $userId)

->where('trans\_type', 'income')

->where('trans\_date', '>', now()->subDays(30))

->where('category\_id', '!=', 18020004)

->sum('trans\_amount');

}

/\*\*

\* Calculate debit transaction count in the last 30 days.

\* @param int $userId

\* @return int

\*/

private function calculateDebitTransactionCount($userId)

{

return Transaction::where('user\_id', $userId)

->where('trans\_type', 'debit')

->where('trans\_date', '>', now()->subDays(30))

->count();

}

/\*\*

\* Sum of debit transactions on Friday/Saturday/Sunday in the last 30 days.

\* @param int $userId

\* @return float

\*/

private function sumDebitTransactionsOnWeekends($userId)

{

return Transaction::where('user\_id', $userId)

->where('trans\_type', 'debit')

->whereIn(DB::raw('DAYOFWEEK(trans\_date)'), [5, 6, 7]) // 5: Friday, 6: Saturday, 7: Sunday

->where('trans\_date', '>', now()->subDays(30))

->sum('trans\_amount');

}

/\*\*

\* Sum of income with transaction amount > 15.

\* @param int $userId

\* @return float

\*/

private function sumIncomeAmountGreaterThan15($userId)

{

return Transaction::where('user\_id', $userId)

->where('trans\_type', 'income')

->where('trans\_amount', '>', 15)

->sum('trans\_amount');

}

}