MrmCogsComponent

August 18, 2026

MrmCogsComponent

Kind: Class

Source: `Frontend/src/app/components/dashboard/dashboard-libary/widgets/mrm-cogs/mrm-cogs.component.ts` (line 78)

Part of: Frontend

`MrmCogsComponent` is an Angular dashboard widget for loading, initializing, aggregating, and saving cost-of-goods data. It responds to component lifecycle changes, checks feature subscription access, calculates totals by year or month, and formats month and price values for display.

Extends: `BaseWidgetComponent`

Methods

Method	Signature	Returns
`checkFeatureSubscription`	`checkFeatureSubscription()`	`boolean`
`ngOnInit`	`ngOnInit()`	`void`
`ngOnChanges`	`ngOnChanges()`	`void`
`getDashboardData`	`getDashboardData()`	`void`
`initalizingData`	`initalizingData(value: BranchData[])`	`void`
`saveDashboardData`	`saveDashboardData()`	`void`
`calculateSumByYear`	`calculateSumByYear()`	`void`
`calculateSumByMonth`	`calculateSumByMonth(monthIndex: number)`	`void`
`translateMonthName`	`translateMonthName(month: MonthData)`	`string`
`getpriceFormatted`	`getpriceFormatted(x: undefined)`	`string`
`onResize`	`onResize()`	`void`

Method	Signature	Returns
`setChaching`	`setChaching()`	`void`
`getChaching`	`getChaching()`	`void`
`switchWithTaxableHandler`	`switchWithTaxableHandler()`	`void`
`switchChartTypeHandler`	`switchChartTypeHandler()`	`void`
`switchBasedOnTendencyHandler`	`switchBasedOnTendencyHandler()`	`void`
`comparingWithPrevMonth`	`comparingWithPrevMonth(month: BranchData, bIndex: number, monthIndex: number, propertyName: string)`	`string`
`comparingBookingWithPrevSumm`	`comparingBookingWithPrevSumm(month: MonthsSum, index: number, propertyName: string)`	`string`
`isCurrentMonth`	`isCurrentMonth(month: MonthData)`	`boolean`
`getpriceForecastAmount`	`getpriceForecastAmount(month: MonthData)`	`number`
`expectedCurrencyChange`	`expectedCurrencyChange(e: undefined)`	`void`
`selectText`	`selectText(e: undefined)`	`void`
`mouseoverRowSpan`	`mouseoverRowSpan(e: undefined, bIndex: number)`	`void`
`mouseoutRowSpan`	`mouseoutRowSpan(e: undefined, bIndex: number)`	`void`
`editSelectedMonthPrepare`	`editSelectedMonthPrepare(month: MonthData, supplierId: number)`	`void`
`openPopup`	`openPopup(n: string, t: string, i: string, btnTxt: string)`	`void`
`savePopup`	`savePopup()`	`void`
`ClosewithDonotSave`	`ClosewithDonotSave()`	`void`
`OnOpenedPopup`	`OnOpenedPopup()`	`void`
`closePopup`	`closePopup()`	`void`

Properties

Property	Type
`sharedPopup`	`SharedPopupComponent`

Property	Type
`_widget`	`Widget`
`_widgetType`	`string`
`yearId`	`number`
`periodId`	`number`
`countOfEditablePrevMonths`	`number`
`withoutOptionBar`	`boolean`
`editMode`	`boolean`
`bookingOrSales`	`number`
`withNominal`	`boolean`
`withTaxable`	`boolean`
`selectedTeam`	`number`
`forecastRuleId`	`number`
`data`	`BranchData[]`
`allowedtoEdit`	`boolean`
`allowedtoEnableEdting`	`boolean`
`widgetEditRequest`	`any`
`widgetEditEnd`	`any`
`forecastRules`	`any[]`
`selectedYear`	`number`
`numYearsDatasource`	`any[]`
`todayWidgetsEnum`	`any`
`cachingWidgetData`	`CachingWidgetData`
`originalData`	`BranchData[]`
`selectedData`	`BranchData[]`
`singleMonths`	`MonthData`
`branchCurrency`	`any`
`basedOnTendency`	`boolean`
`SeriseRange`	`any[]`

Property	Type
`echartsIntance`	`any`
`totalBookingMonths`	`MonthsSum[]`
`tempMonthSum`	`MonthsSum`
`totalTargets`	`number`
`yearBooking`	`number`
`totalForecasts`	`number`
`tempTotalTarget`	`TotalTargets`
`allGroupsTotalSales`	`number`
`totalPercentageOfAchievedSales`	`number`
`totalCumulativeSales`	`number`
`totalDirectSales`	`number`
`totalTotalDirectSales`	`number`
`totalDirectService`	`number`
`totalDirectSalesWithScrap`	`number`
`totalTotalDirectSalesWithScrap`	`number`
`totalCommission`	`number`
`_totalSalesAfterCommission`	`number`
`popup`	`any`
`searchText`	`string`
`pastDiff`	`number`
`urrenFeature`	`number`

Where it refuses work

- `MrmCogsComponent` stops the work with an early return when `this.dashboardLibraryService.checkFeatureSubscription(this.widget.chartEnumId)`.
- `MrmCogsComponent` stops the work with an early return when `month.MonthId == 12`.
- `MrmCogsComponent` stops the work with an early return when `month[propertyName] < this.data[bIndex].MrmCogsGroups[monthIndex - 1][propertyName]`.

- `MrmCogsComponent` stops the work with an early return when `month[propertyName] < this.totalBookingMonths[index - 1][propertyName]`.
- `MrmCogsComponent` stops the work with an early return when `AppSettings.calculateMonthsBeforeToday(month.YearId, month.MonthId -1) == 0 || AppSetting⋯`.

Diagram

```
graph LR
  A[Angular lifecycle] --> B[MrmCogsComponent]
  B --> C[checkFeatureSubscription]
  B --> D[getDashboardData]
  D --> E[initalizingData]
  E --> F[calculateSumByYear]
  E --> G[calculateSumByMonth]
  F --> H[getpriceFormatted]
  G --> I[translateMonthName]
  B --> J[saveDashboardData]
```

Usage

```
import { ComponentFixture, TestBed } from '@angular/core/testing';
import { MrmCogsComponent } from './mrm-cogs.component';

describe('MrmCogsComponent', () => {
  let fixture: ComponentFixture<MrmCogsComponent>;
  let component: MrmCogsComponent;

  beforeEach(() => {
    TestBed.configureTestingModule({
      declarations: [MrmCogsComponent],
    });

    fixture = TestBed.createComponent(MrmCogsComponent);
    component = fixture.componentInstance;
  });

  it('loads and formats dashboard data', () => {
    component.ngOnInit();

    if (component.checkFeatureSubscription()) {
      component.getDashboardData();
      component.calculateSumByYear();
      component.calculateSumByMonth();

      const formattedPrice = component.getpriceFormatted();
```

```
      const monthName = component.translateMonthName();

      expect(formattedPrice).toBeDefined();
      expect(monthName).toBeDefined();
    }
  });
});
```

AI Coding Instructions

- Call `checkFeatureSubscription()` before requesting or displaying dashboard data that depends on subscription access.
- Keep data loading in `getDashboardData()` and data setup in `initalizingData()` so lifecycle methods remain focused on orchestration.
- Recalculate yearly and monthly sums after dashboard data changes before rendering formatted values.
- Use `translateMonthName()` and `getpriceFormatted()` for display output instead of duplicating formatting logic in templates.
- Preserve the existing lifecycle flow between `ngOnInit()`, `ngOnChanges()`, data initialization, and dashboard persistence.

Used by

3 references from 3 files. Each is a place in this repository where the symbol is actually used — go read one rather than trusting an example.

Imported by (3)

- `DashboardTabComponent` — `Frontend/src/app/components/dashboard/dashboard-libary/dashboard-tab/dashboard-tab.component.ts`:1
- `DashboardWidgetsModule` — `Frontend/src/app/components/dashboard/dashboard-widgets.module.ts`:1
- `ReportsTabComponent` — `Frontend/src/app/components/reports/reports-library/reports-tab/reports-tab.component.ts`:1