Quotes & Sales Report Documentation

1. Overview

The Quotes & Sales Report aggregates **Quotes** and **Sales** metrics in a single call and supports role-aware filtering by **Team**, **Status**, and **Date Range**. When `status=paid-in`, date filtering and timeline grouping use `Receipts.Date`.

2. Endpoint

```
[HttpGet("QuotesAndSalesReport")]
GET /api/report/QuotesAndSalesReport?teamId=<id>&fromDate=dd/MM/yyyy&toDate=dd/MM/yyyy&status=<key>
```

2.1 Parameters

Name	Type	Applied On	Description
`teamId`	string (ObjectId)	`Team._id` / `Team == null`	Restricts invoices/quotes to a team or unassigned.
`fromDate`	string (dd/MM/yyyy)	Invoice `Date` or `Receipts.Date*`	Lower bound (inclusive). *Uses receipts date when* ** `status=paid-in` **.
`toDate`	string (dd/MM/yyyy)	Invoice `Date` or `Receipts.Date*`	Upper bound (inclusive). *Uses receipts date when* ** `status=paid-in` **.
`status`	string	`Status`, `Receipts.Status`, `Company.New`	See Status Options. Default: `active`.

2.2 Practice & Role Logic

- All queries are scoped to the caller's **Practice ID** (`_p`).
- Team managers/admins can query arbitrary teams; non-admins without a team return no data.
- When managing teams and `teamId` is not provided, the user's own team is used.

3. Status Options

Key	Condition (Mongo semantics)
`due`	`Status != Void AND Amount.Due > 0`
`paid`	`Status != Void AND (Amount.Due == 0 OR missing) AND any(Receipts.Status != Pending)`

Key	Condition (Mongo semantics)
`paid-in`	`Status != Void AND Company.New == true` *(groups by* ** `Receipts.Date` **)*
`void`	`Status == Void`
`inprogress`	`any(Receipts.Status == Pending)`
`active` *(default)*	`Status != Void`

4. Aggregation Summary

1. **Total Sales** = `sum( (Amount.Gross - ifNull(Amount.Discount,0)) + ifNull(Amount.Tax,0) )`
2. **Paid Sales** = `sum(Amount.Paid)`
3. **Due Sales** = `Total Sales - Paid Sales - sum(Amount.CreditNoteAdjustment)`
4. **Void Sales** = `sum(Amount.Gross where Status == Void)`
5. **Sales Count** = distinct count of active (non-void) invoices after company join
6. **Quotes** = aggregate `NetAmount` grouped by Status (Sent, Accepted, Rejected)
7. **Timeline** = group by invoice `Date`; when `status=paid-in`, group by `Receipts.Date`

4.1 Server-Side Logic (reference)

- **Scope & Roles**: Restrict all queries by Practice ID (`_p`). If the caller manages teams, allow querying any team; otherwise require the caller's own team. When a `teamId` is provided, include both matching-team and unassigned records.
- **Status Filter**: Map `status` to Mongo conditions:
- `due`: non-void and `Amount.Due > 0`
- `paid`: non-void and `(Amount.Due == 0 OR missing)` and any receipt not `Pending`
- `paid-in`: non-void and `Company.New == true` *(timeline groups by* `Receipts.Date`*)*
- `void`: `Status == Void`
- `inprogress`: any `Receipts.Status == Pending`
- default `active`: non-void
- **Date Field Switch**: Use invoice `Date` for most statuses; when `status=paid-in`, switch all range filters and timeline grouping to `Receipts.Date`.
- **Adjusted Gross (Total Sales)**: `(Amount.Gross - ifNull(Amount.Discount,0)) + ifNull(Amount.Tax,0)`.
- **Paid & Due**: `Paid = sum(Amount.Paid)`; `Due = TotalSales - Paid - sum(Amount.CreditNoteAdjustment)`.
- **Void Sales**: Sum `Amount.Gross` over `Status == Void`.
- **Sales Badge Count**: Distinct count of *active* (post-join) invoices.
- **Timeline Series**:
- *Invoice-date mode*: group by formatted invoice `Date`; output `categories`, `receiptsData`, and `pendingData = AdjGross - Paid` per day.
- *Paid-in mode*: unwind `Receipts`, group by `Receipts.Date`; output `categories`, `receiptsData = sum(Receipts.Amount)`, and set `pendingData = 0`.
- **Quotes Aggregation**: Filter quotes by practice/team/date; group by `Status` to get totals for *Sent*, *Accepted*, *Rejected*.

- **Response Mapping**: Populate `sentQuotes`, `acceptedQuotes`, `pendingQuotes`, `totalQuotesAmount`, `totalSales`, `paidSales`, `dueSales`, `voidSales`, `salesBadgeCount`, plus `categories`, `receiptsData`, `pendingData`.

5. Response Types

5.1 C#

```
public class QuotesAndSalesReportResponse
{
    // Quotes
    public decimal sentQuotes { get; set; }
    public decimal acceptedQuotes { get; set; }
    public decimal pendingQuotes { get; set; }
    public decimal totalQuotesAmount { get; set; }
    public decimal totalQuotes { get; set; }
    public int badgeCount { get; set; }

    // Sales
    public decimal totalSales { get; set; }
    public decimal paidSales { get; set; }
    public decimal dueSales { get; set; }
    public decimal voidSales { get; set; }
    public int salesBadgeCount { get; set; }

    // Daily breakdown
    public List<decimal> receiptsData { get; set; } = new();
    public List<decimal> pendingData  { get; set; } = new();
    public List<string> categories    { get; set; } = new();
}
```

5.2 TypeScript

```
export type QuotesAndSalesReportResponse = {
  // Quotes
  sentQuotes: number;
  acceptedQuotes: number;
  pendingQuotes: number;
  totalQuotesAmount: number;
  totalQuotes: number;
  badgeCount: number;
  salesBadgeCount: number;

  // Sales
  paidSales: number;
  dueSales: number;
  voidSales: number;
  totalSales: number;
```

```
  // Chart
  receiptsData: number[];
  pendingData: number[];
  categories: string[];
};
```

6. Example

```
GET /api/report/QuotesAndSalesReport?
teamId=671f6a13f3&fromDate=01/10/2025&toDate=31/10/2025&status=paid
```

When `status=paid-in`, date filtering and timeline grouping apply on `Receipts.Date` instead of invoice `Date`.

7. Filters Quick Reference

Name	Field(s)	Notes
Practice Filter	`_p`	Implicitly set from authenticated user
Team Filter	`teamId → Team._id` or `Team == null`	Includes team-matched and unassigned records
Status Filter	`status`	See Status Options
Date Range	`fromDate`, `toDate`	Invoice `Date` or `Receipts.Date` (when `paid-in`)

8. Client API (Service)

```
getQuotesAndSalesReport = (fromDate?: string, toDate?: string, teamId?:
string) =>
  ApiUtility.getResult<QuotesAndSalesReportResponse>(`${this.route}/
QuotesAndSalesReport`, {
    fromDate,
    toDate,
    teamId,
  });
```

9. Frontend Usage

9.1 SalesCard

- Fetches `QuotesAndSalesReportResponse` and renders KPIs (**Total sales**, **Receipts**, **Pending**, **Void**, **badgeCount**).
- Chooses daily vs monthly timeline based on whether `from` and `to` fall in the same calendar month.

- Builds `categories` as ISO `YYYY-MM-DD` (daily) or `YYYY-MM` (monthly); aggregates month buckets when needed.
- ECharts bar chart: `Receipts` and `Pending` series; y-axis uses a “nice” ceiling for headroom.

9.2 QuotesCard

- Fetches the same endpoint; donut chart reflects **Accepted** and **Pending** quotes.
- Center label shows **Total quotes**; badge on header shows **Quotes badgeCount**.