Api 1:

app.get('/api/total\_items', (req, res) => {

const { start\_date, end\_date, department } = req.query;

const startDate = new Date(start\_date);

const endDate = new Date(end\_date);

const filter = {

department: department,

purchase\_date: {

$gte: startDate,

$lte: endDate

}

};

db.collection('purchases').aggregate([

{ $match: filter },

{ $group: { \_id: null, totalSeats: { $sum: '$seats' } } }

])

.toArray()

.then(result => {

const totalSeats = result.length > 0 ? result[0].totalSeats : 0;

res.json(totalSeats);

})

.catch(error => {

console.error('Error retrieving total seats:', error);

res.status(500).json({ error: 'An error occurred' });

});

});

Api 2:

app.get('/api/nth\_most\_total\_item', (req, res) => {

const { item\_by, start\_date, end\_date, n } = req.query;

const startDate = new Date(start\_date);

const endDate = new Date(end\_date);

let query;

let orderByColumn;

let responseColumn;

if (item\_by === 'quantity') {

query = `

SELECT item\_name

FROM purchases

WHERE purchase\_date BETWEEN ? AND ?

GROUP BY item\_name

ORDER BY SUM(quantity) DESC

LIMIT ?, 1

`;

orderByColumn = 'quantity';

responseColumn = 'item\_name';

} else if (item\_by === 'price') {

query = `

SELECT item\_name

FROM purchases

WHERE purchase\_date BETWEEN ? AND ?

GROUP BY item\_name

ORDER BY SUM(price) DESC

LIMIT ?, 1

`;

orderByColumn = 'price';

responseColumn = 'item\_name';

} else {

return res.status(400).json({ error: 'Invalid item\_by parameter' });

}

const values = [startDate, endDate, n - 1];

connection.query(query, values, (err, result) => {

if (err) {

console.error('Error retrieving nth most sold item:', err);

return res.status(500).json({ error: 'An error occurred' });

}

if (result.length === 0) {

return res.status(404).json({ error: 'No data found' });

}

const nthItem = result[0][responseColumn];

res.json(nthItem);

});

});

Api 3:

app.get('/api/percentage\_of\_department\_wise\_sold\_items', (req, res) => {

const { start\_date, end\_date } = req.query;

// Convert start\_date and end\_date to JavaScript Date objects

const startDate = new Date(start\_date);

const endDate = new Date(end\_date);

const query = `

SELECT department, (SUM(seats) / (SELECT SUM(seats) FROM purchases WHERE purchase\_date BETWEEN ? AND ?)) \* 100 AS percentage

FROM purchases

WHERE purchase\_date BETWEEN ? AND ?

GROUP BY department

`;

const values = [startDate, endDate, startDate, endDate];

connection.query(query, values, (err, result) => {

if (err) {

console.error('Error retrieving percentage of sold items:', err);

return res.status(500).json({ error: 'An error occurred' });

}

const percentageByDepartment = {};

result.forEach(row => {

const department = row.department;

const percentage = row.percentage.toFixed(2);

percentageByDepartment[department] = `${percentage}%`;

});

res.json(percentageByDepartment);

});

});

Api 4:

app.get('/api/monthly\_sales', (req, res) => {

const { product, year } = req.query;

const query = `

SELECT MONTH(purchase\_date) AS month, SUM(quantity \* price) AS monthlySales

FROM purchases

WHERE product = ? AND YEAR(purchase\_date) = ?

GROUP BY MONTH(purchase\_date)

ORDER BY MONTH(purchase\_date)

`;

const values = [product, year];

connection.query(query, values, (err, result) => {

if (err) {

console.error('Error retrieving monthly sales:', err);

return res.status(500).json({ error: 'An error occurred' });

}

const monthlySales = result.map(row => row.monthlySales);

res.json(monthlySales);

});

});