Vacation Request Management API

A comprehensive REST API for managing employee vacation requests built with Spring Boot. This system allows employees to submit vacation requests and managers to review, approve, or reject them while tracking vacation day balances and detecting schedule conflicts.

Features

Employee Features

- Submit Vacation Requests - Create new vacation requests with date validation
- View Personal Requests - See all submitted requests with status filtering
- Track Vacation Days - Monitor remaining vacation days (30-day annual limit)
- Status Filtering - Filter requests by status (pending, approved, rejected)

Manager Features

- Request Overview - View all employee vacation requests
- Individual Employee Reports - See vacation history for specific employees
- Approve/Reject Requests - Process requests with proper authorization
- Overlap Detection - Identify conflicting vacation schedules
- Advanced Filtering - Filter by status combinations

System Features

- 30-Day Annual Limit - Automatic validation of vacation day limits
- Role-Based Access - Separate endpoints for employees and managers
- Overlap Detection - Prevent scheduling conflicts
- Data Persistence - H2 in-memory database with JPA
- Input Validation - Comprehensive request validation
- RESTful Design - Clean, intuitive API endpoints

Quick Start

1. Clone and Setup

git clone https://github.com/okwandiswa/vacation-request-api.git

cd vacation-request-api

2. Run the Application

Using VS Code:

1. Open the project in VS Code

2. Navigate to VacationApiApplication.java
3. Click the "Run" button above the main method

3. Verify Setup

The application will start on http://localhost:8080

Check initial data:

curl http://localhost:8080/api/employees

You should see 5 users:

- **Employees:** John Smith, Jane Doe, Alice Johnson
- **Managers:** Bob Manager, Sarah Director

API Documentation

Employee Endpoints

-Create Vacation Request

POST /api/employees/{employeeId}/vacation-requests

Content-Type: application/json

```
{
  "vacationStartDate": "2025-12-25",
  "vacationEndDate": "2025-12-30",
  "reason": "Christmas vacation"
}
```

-View Employee's Requests

GET /api/employees/{employeeId}/vacation-requests

GET /api/employees/{employeeId}/vacation-requests?status=pending

-Get Vacation Summary

GET /api/employees/{employeeId}/vacation-summary

Manager Endpoints

-View All Requests

GET /api/managers/{managerId}/vacation-requests

GET /api/managers/{managerId}/vacation-requests?status=pending

GET /api/managers/{managerId}/vacation-requests?status=pending,approved

-View Employee Overview

GET /api/managers/{managerId}/employees/{employeeId}/vacation-requests

-Detect Overlapping Requests

GET /api/managers/{managerId}/overlapping-requests?startDate=2025-12-20&endDate=2025-12-30

-Process Requests

PUT /api/managers/{managerId}/vacation-requests/{requestId}?action=approve

PUT /api/managers/{managerId}/vacation-requests/{requestId}?action=reject

General Endpoints

-Get All Employees

GET /api/employees

Request/Response Examples

Vacation Request Response


```
{
  "id": 1,
  "author": {
    "id": 1,
    "name": "John Smith",
    "email": "john.smith@company.com",
    "department": "Engineering",
    "role": "EMPLOYEE"
  },
  "status": "PENDING",
  "resolved_by": null,
  "request_created_at": "2025-08-20T10:30:00.123456",
  "vacation_start_date": "2025-12-25",
  "vacation_end_date": "2025-12-30",
```

```
  "reason": "Christmas vacation"
}
```

Vacation Summary Response

```
{
  "employeeId": 1,
  "employeeName": "John Smith",
  "totalVacationDays": 30,
  "usedVacationDays": 6,
  "remainingVacationDays": 24
}
```

Testing

Using VS Code REST Client

The project includes an api-tests.http file with pre-configured test requests. Install the "REST Client" extension in VS Code and click "Send Request" above each HTTP request.

Example Test Workflow

1. Create vacation request for Employee 1
2. Check vacation summary (should show 30 days available)
3. Manager views request (should show pending status)
4. Manager approves request
5. Check vacation summary again (should show 24 days remaining)

Using curl

```
# Create vacation request
curl -X POST http://localhost:8080/api/employees/1/vacation-requests \
  -H "Content-Type: application/json" \
  -d '{"vacationStartDate":"2025-12-25","vacationEndDate":"2025-12-30","reason":"Christmas vacation"}'

# Approve request (as manager)
curl -X PUT "http://localhost:8080/api/managers/4/vacation-requests/1?action=approve"
```

Project Structure

src/main/java/com/example/vacationapi/

–VacationApiApplication.java # Main application class

–Employee.java # Employee entity

–VacationRequest.java # Vacation request entity

–RequestStatus.java # Status enum

–Role.java # Employee role enum

–EmployeeRepository.java # Employee data access

–VacationRequestRepository.java # Vacation request data access

– VacationService.java # Business logic

–VacationController.java # Employee endpoints

–ManagerController.java # Manager endpoints

–EmployeeController.java # Employee info endpoints

– VacationRequestDTO.java # Request data transfer object

–VacationSummaryDTO.java # Summary data transfer object

–OverlapReportDTO.java # Overlap report object

-DataInitializer.java # Initial data setup

src/main/resources/

-application.properties # Application configuration

api-tests.http # HTTP test requests

Configuration

Database Configuration

The application uses H2 in-memory database by default. Configuration in application.properties:

```
# H2 Database
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.username=sa
spring.datasource.password=

# H2 Console (for development)
```

spring.h2.console.enabled=true

Access H2 Console: http://localhost:8080/h2-console

- JDBC URL: jdbc:h2:mem:testdb
- Username: sa
- Password: (leave empty)

Server Configuration

server.port=8080

Business Rules

Vacation Day Management

- Each employee has 30 vacation days per year
- Only approved requests count toward used days
- Requests cannot exceed available days

Request Workflow

1. Employee submits request (status: PENDING)
2. Manager reviews request
3. Manager approves (status: APPROVED) or rejects (status: REJECTED)
4. Approved requests update employee's used vacation days

Authorization Rules

- Employees can only view/create their own requests
- Managers can view all requests and approve/reject any request
- Only users with MANAGER role can process requests

Validation Rules

- End date must be after start date
- Employee must exist
- Manager must exist (for processing)
- Cannot exceed 30-day annual limit

Common Issues & Solutions

Application Won't Start

- Ensure Java 17+ is installed

- Check that port 8080 is available
- Verify all dependencies are downloaded

400 Bad Request Errors

- Check JSON format in requests
- Ensure required fields are included
- Verify date format (YYYY-MM-DD)

Validation Errors

- Ensure vacation dates are in the future
- Check that employee/manager IDs exist
- Verify request doesn't exceed vacation day limit