Fully dressed use case for add, update and delete employee

Use Case Name:

New Employee Registration

Primary Actor:

HR Manager

Secondary Actors:

IT Administrator, Department Manager

Stakeholders and Interests:

- **New Employee:** Wants their profile set up and access granted to relevant systems.
- **HR Manager:** Needs to ensure employee details are recorded accurately and onboarding is smooth.
- **IT Administrator:** Needs to provide necessary access credentials and work tools.
- **Department Manager:** Wants the employee registered correctly for role assignment.

Preconditions:

- The new employee has accepted the job offer.
- All mandatory documents have been submitted.
- The HR Manager is logged into the system with appropriate permissions.
- The system is functioning properly.

Postconditions:

- Employee data is saved in the system.
- Employee is assigned a unique ID.

- Login credentials are generated.
- Notifications are sent to IT and the Department Manager.
- The onboarding checklist is initiated.

Main Success Scenario (Basic Flow):

- **Trigger:** The HR Manager initiates the process after receiving a signed offer letter.
- **HR Input:**
    - Logs into the system and navigates to the Employee Management section.
    - Selects “Add New Employee.”
    - Enters personal details, job title, department, and documents.
- **System Action:**
    - Validates mandatory fields.
    - Assigns a unique employee ID.
    - Stores data in the HRMS database.
- **IT Notification:**
    - System sends notification to IT for system setup.
    - IT assigns email, access credentials, and work tools.
- **Manager Notification:**
    - System sends notification to Department Manager about new hire.
- **Onboarding Setup:**
    - System generates onboarding checklist (training, documentation, etc.).
- **Confirmation:**
    - HR receives confirmation and sends welcome email to the employee.

Alternative Flows (Extensions):

- **Missing Document:**
    - Step 2A: If documents are missing, system halts process and prompts to upload.
- **Duplicate Employee:**
    - Step 3A: If employee with same ID exists, system shows a duplicate warning.
- **IT Setup Delays:**
    - Step 4A: If IT doesn't acknowledge setup, reminder is auto-triggered.

Exception Flows:

- **System Error:**
    - System fails during registration, HR is alerted and logs the issue.
- **Permission Denied:**
    - Unauthorized users attempting registration are denied access.

Trigger:

- HR initiates upon final hiring confirmation.

Special Requirements:

- Secure handling of personal data.
- Compliance with labor laws.
- Role-based access controls.

Use Case Name:

Delete Employee Record

Primary Actor:

HR Manager

Secondary Actors:

Admin

Stakeholders and Interests:

- **HR Manager:** Wants to remove employee data cleanly.
- **Admin:** Ensures proper auditing and no data loss.

Preconditions:

- Employee has officially left the organization.
- All exit formalities and clearances are complete.
- HR has appropriate access rights.

Postconditions:

- Employee status marked as "Inactive" or "Deleted."
- Records are archived.
- IT access is revoked.
- Notifications are sent to Admin and relevant departments.

Main Success Scenario (Basic Flow):

- **Trigger:** HR receives final clearance form.
- **HR Action:**
    - Logs in and navigates to Employee Management.
    - Searches for employee by ID.
    - Selects “Delete Employee” option.
- **System Prompts:**
    - Confirms last working day and exit reason.

    - Validates all dues are cleared.
- **Access Revocation:**
    - Notifies IT to deactivate accounts and tools.
- **Archiving:**
    - Moves data to archive for audit purposes.
- **Notification:**
    - Sends update to Admin and Manager.

Alternative Flows:

- **Pending Dues:**
    - Step 3A: Process paused until dues cleared.
- **Exit Not Approved:**
    - Step 2A: System alerts if no exit approval attached.

Exception Flows:

- **Record Not Found:**
    - HR is prompted to recheck ID.
- **System Crash:**
    - Logs error and halts deletion.

Trigger:

- Exit clearance form submission.

Special Requirements:

- GDPR compliance for data deletion.
- Proper access logs of deletion action.

Use Case Name:

Update Employee Details

Primary Actor:

HR Executive

Secondary Actors:

Department Manager

Stakeholders and Interests:

- **HR Executive:** Needs to ensure up-to-date employee data.
- **Department Manager:** Wants current data for work management.
- **Employee:** Expects accurate personal and professional records.

Preconditions:

- Employee already exists in the system.
- HR is logged in with permissions.
- Reason for update is valid (e.g., promotion, address change).

Postconditions:

- Updated details are stored.
- Timestamped audit record is created.
- Relevant departments are notified.

Main Success Scenario (Basic Flow):

- **Trigger:** Employee submits request or HR receives update mandate.
- **HR Action:**
    - Logs in and locates employee record.
    - Clicks on “Edit” and makes necessary changes.

- **System Validation:**
    - Checks for valid inputs (e.g., phone number format).
- **Audit Logging:**
    - Saves a snapshot of old vs. new data.
- **Save Changes:**
    - Confirms and commits changes.
- **Notifications:**
    - Sends update to relevant parties (e.g., payroll, IT).

Alternative Flows:

- **Unauthorized Update:**
    - Step 2A: If user doesn’t have edit rights, system blocks changes.
- **Invalid Inputs:**
    - Step 3A: Prompts HR to correct data format.

Exception Flows:

- **Update Conflict:**
    - Concurrent update attempt results in merge conflict.
- **Missing Justification:**
    - System prompts for update reason before saving.

Trigger:

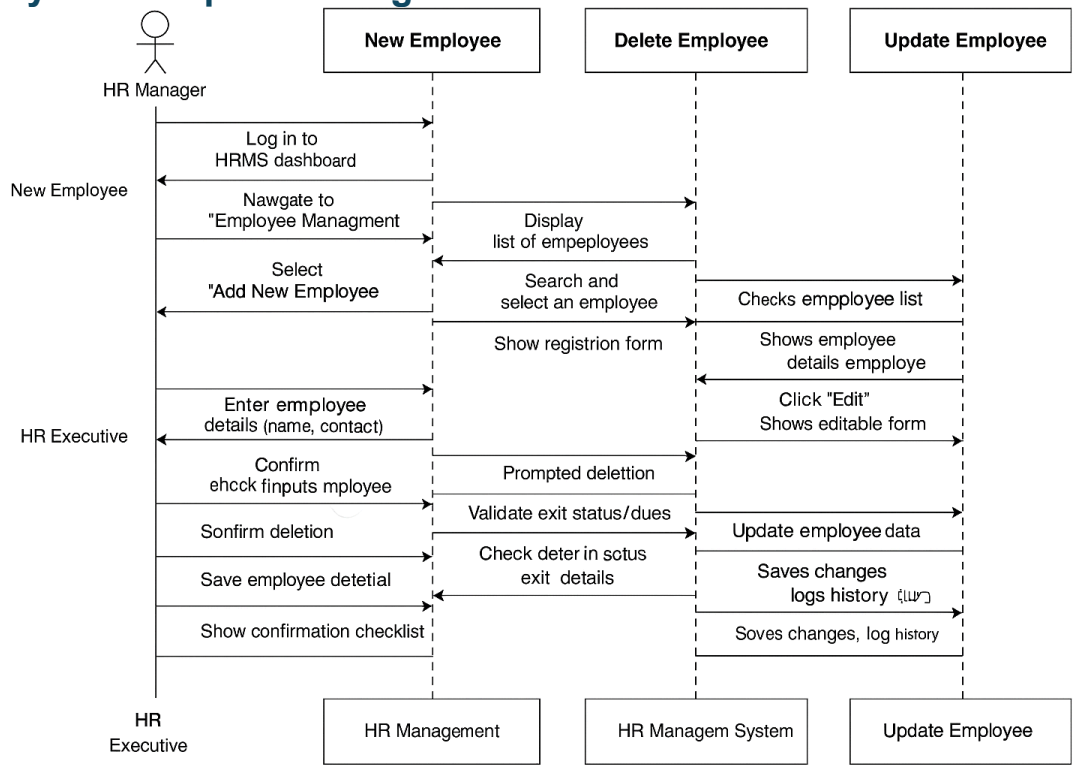
- Employee request or organizational change.

Special Requirements:

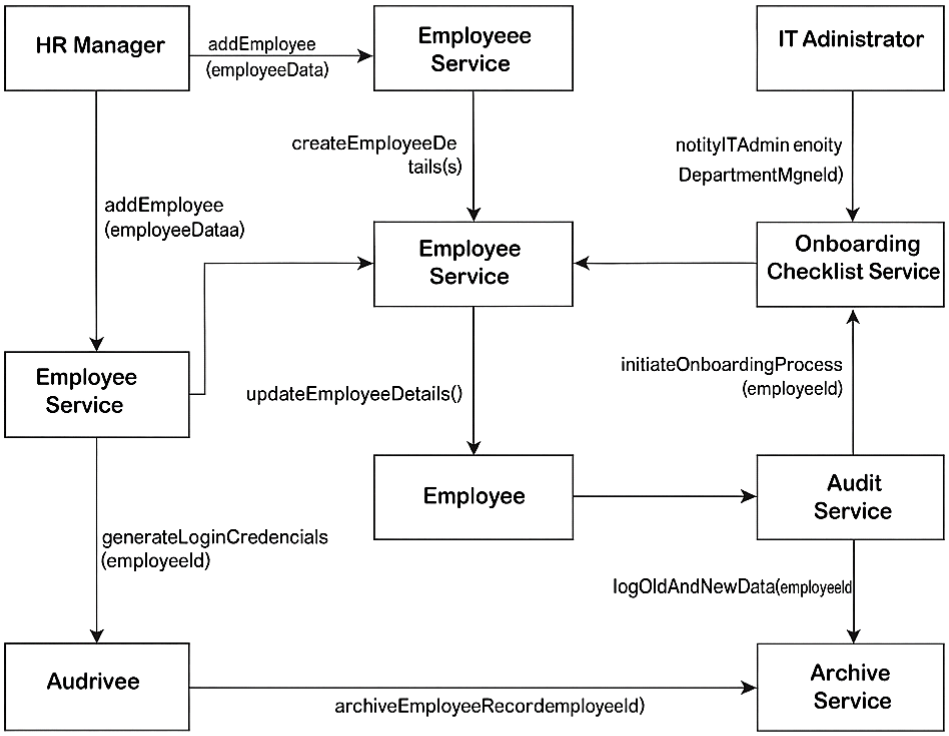
- Editable logs for compliance.
- Restricted updates (e.g., salary changes require approval).

- Backup before every major change.

System Sequence Diagram:

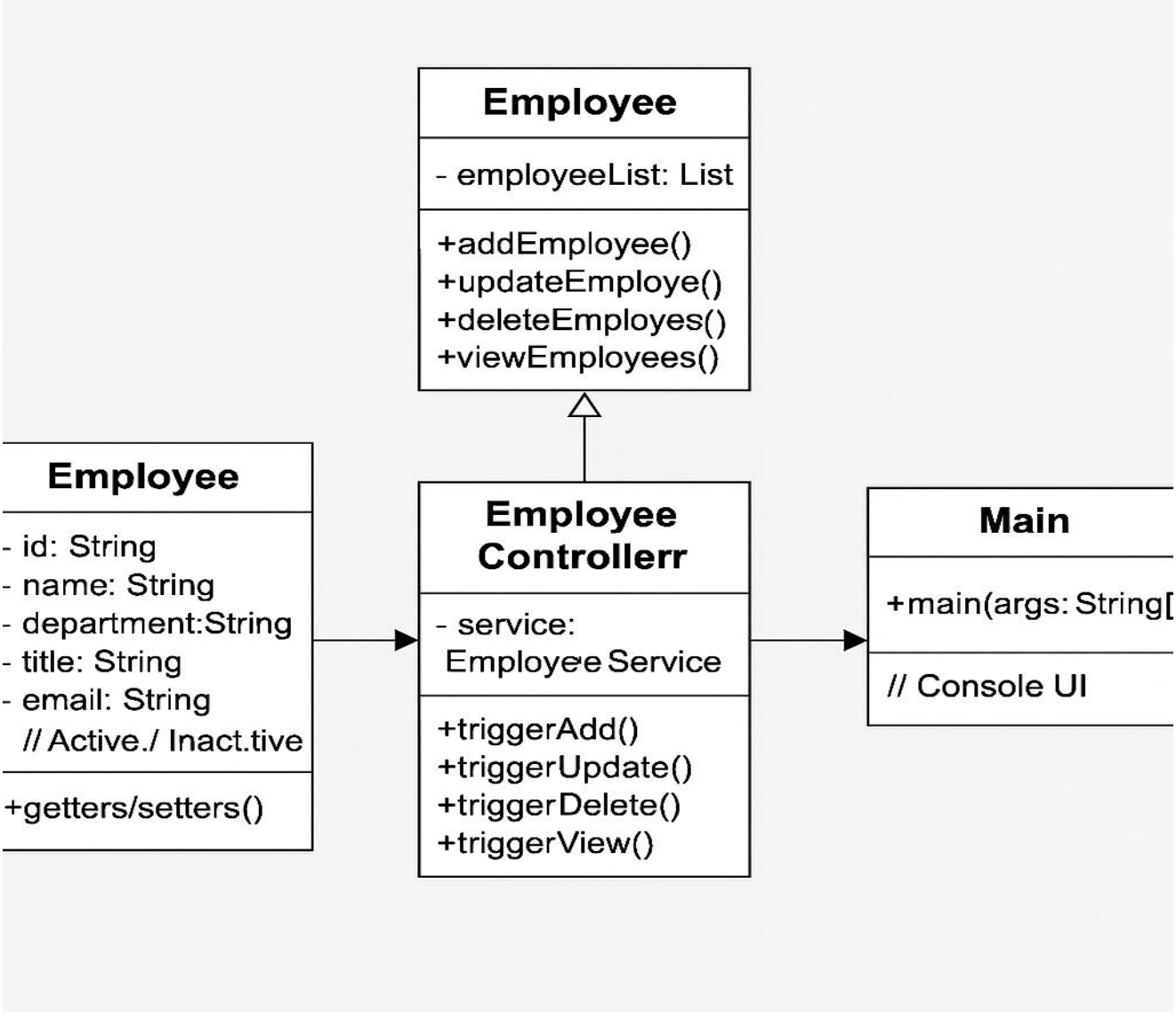


System Event Design via collaboration diagram:

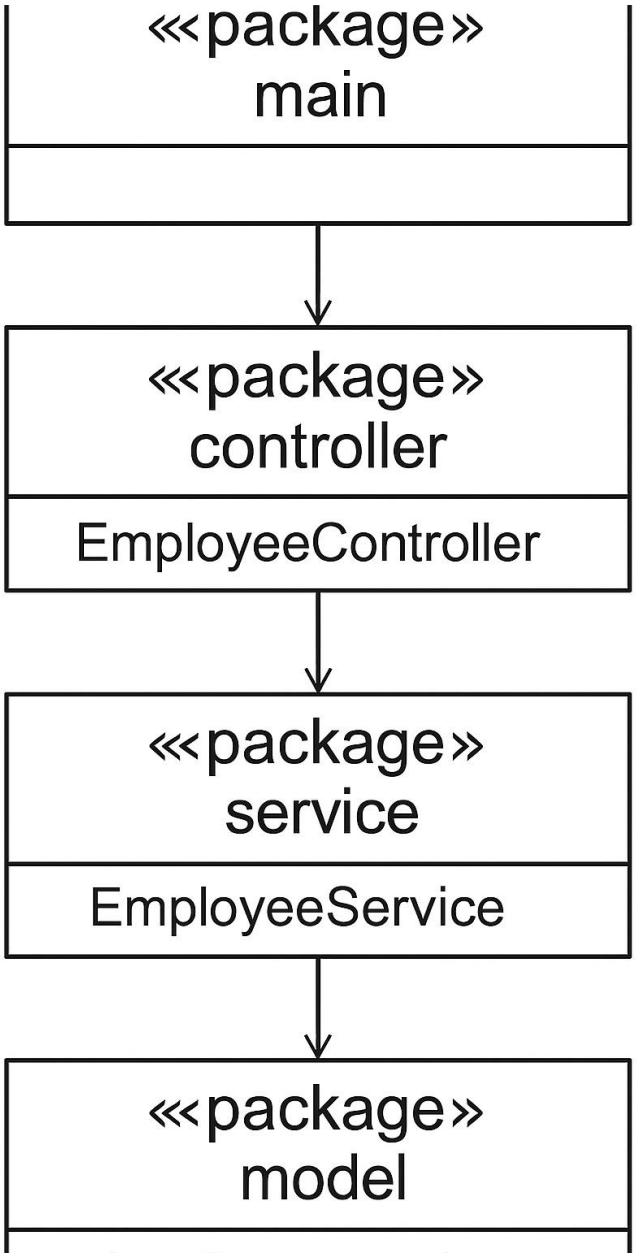


System Event Design via Collaboration Design (GRASP)

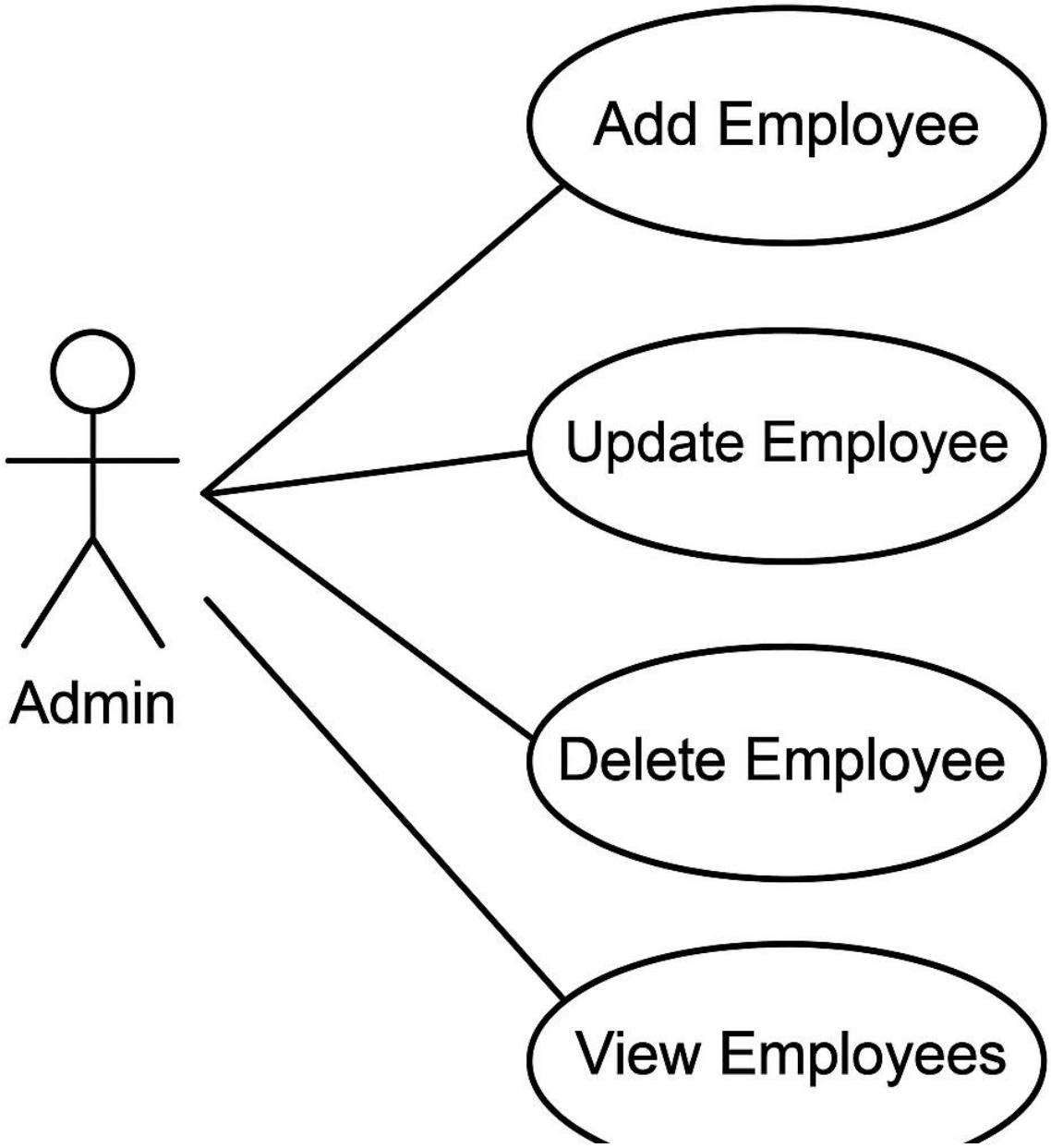
Class diagram:



Package diagram:



Use case diagram:



Prototype(example coding) – Employee Management:

```
// === MODEL ===
class Employee {
    private String id;
```

```
    private String name;
    private String department;
    private String title;
    private String email;
    private String status; // Active / Inactive

    // Constructor, Getters, Setters
}

// === SERVICE ===
class EmployeeService {
    // Store employees in a simple list or map
    // Create new employee
    // Update existing employee
    // Delete (or mark as inactive)
    // View all employees
}

// === CONTROLLER ===
class EmployeeController {
    private EmployeeService service = new EmployeeService();

    // Methods to trigger add, update, delete, and view operations
}
```

```
// === VIEW (for console interaction, basic UI) ===
public class Main {
    public static void main(String[] args) {
        // Initialize controller
        // Menu-driven interaction for:
        // 1. Add Employee
        // 2. Update Employee
        // 3. Delete Employee
        // 4. View Employees
        // 0. Exit
    }
}
```