Employee management system

The employees on this system are assigned to different states.

The states (State machine) for A given Employee are:

- **ADDED**
- **IN-CHECK**
- **APPROVED**
- **ACTIVE**

Initially when an employee is added it will be assigned "**ADDED**" state automatically.

The allowed state transitions are:

ADDED -> **IN-CHECK** *-> **APPROVED** -> **ACTIVE**

Furthermore, **IN-CHECK** state is special and has the following orthogonal child substates:

- **SECURITY_CHECK_STARTED**
- **SECURITY_CHECK_FINISHED**
- **WORK_PERMIT_CHECK_STARTED**
- **WORK_PERMIT_CHECK_FINISHED**

with allowed state transitions:

- **SECURITY_CHECK_STARTED** -> **SECURITY_CHECK_FINISHED**
- **WORK_PERMIT_CHECK_STARTED** -> **WORK_PERMIT_CHECK_FINISHED**

Employee management system

This means that a complete state of an employee in the IN_CHECK state could look like (**IN_CHECK**, **SECURITY_CHECK_STARTED**, **WORK_PERMIT_CHECK_FINISHED**).

Examples of permitted transition:

- (**IN_CHECK**, **SECURITY_CHECK_STARTED**, **WORK_PERMIT_CHECK_STARTED**) -> (**IN_CHECK**, **SECURITY_CHECK_FINISHED**, **WORK_PERMIT_CHECK_STARTED**)
- (**IN_CHECK**, **SECURITY_CHECK_STARTED**, **WORK_PERMIT_CHECK_STARTED**) -> (**IN_CHECK**, **SECURITY_CHECK_STARTED**, **WORK_PERMIT_CHECK_FINISHED**)

Transition from **IN_CHECK** state to **APPROVED** state happens automatically when the complete state is (**IN_CHECK**, **SECURITY_CHECK_FINISHED**, **WORK_PERMIT_CHECK_FINISHED**). Transition from **IN_CHECK** state to **APPROVED** without meeting the condition above is not allowed.

Examples

Here is an example how a sequence of requests might look like:

1. create employee
2. Update state of an employee to **IN_CHECK**
3. Update substate of **IN_CHECK** state of an employee to **SECURITY_CHECK_FINISHED**
4. Update substate of **IN_CHECK** state an employee to **WORK_PERMIT_CHECK_FINISHED** (employee is auto-transitioned to **APPROVED** state)
5. Update state of an employee to **ACTIVE**

Another possible sequence of requests is:

1. create employee
2. Update state of an employee to **IN_CHECK**
3. Update substate of **IN_CHECK** state an employee to **WORK_PERMIT_CHECK_FINISHED**
4. Update substate of **IN_CHECK** state an employee to **SECURITY_CHECK_FINISHED** (employee is auto-transitioned to **APPROVED** state)
5. Update state of an employee to **ACTIVE**

Requirements

Our backend stack is:

- Java
- Spring Framework

Your task is to build Restful API doing the following:

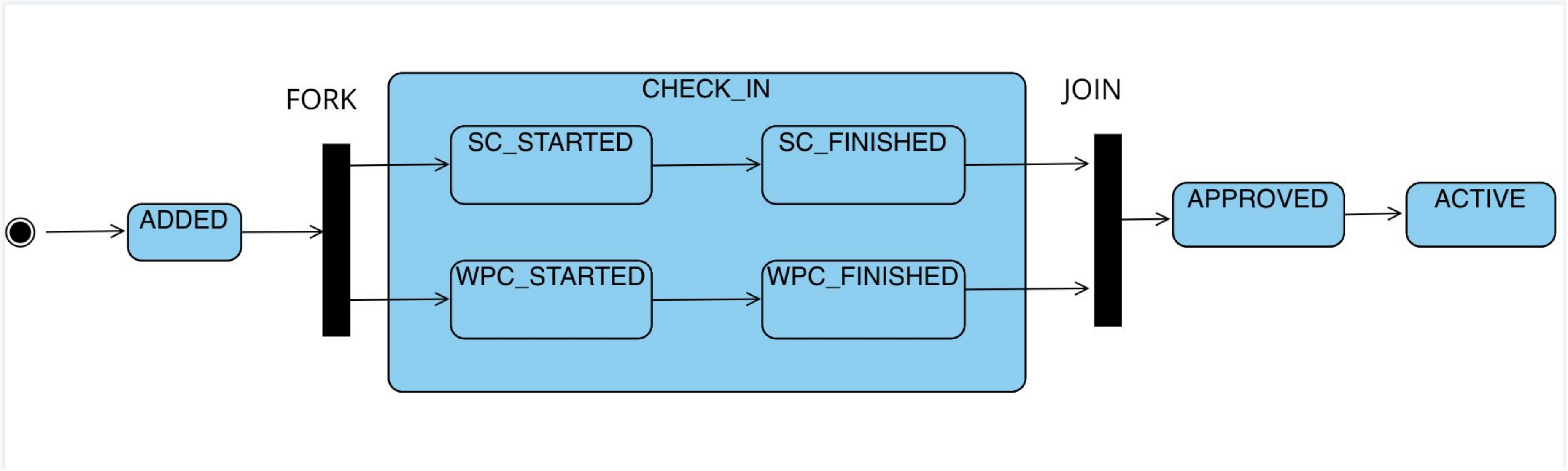
- An Endpoint to support adding an employee with very basic employee details including (name, contract information, age, you can decide.) With initial state "**ADDED**" which indicates that the employee isn't active yet.
- Another endpoint to change the state of a given employee to any of the states defined above in the state machine respecting the transition rules
- An Endpoint to fetch employee details

Please provide a solution with the above features with the following consideration.

- Being simply executable with minimum effort. Ideally using Docker and docker-compose or any similar approach
- For state management (State machine) you can use any library or data structure you consider appropriate
  - Some suggestions from our side (these are only suggestions, feel free to use something else if you want):
    - ENUM with states
    - Stateless4j library: https://github.com/stateless4j/stateless4j
    - Spring state machine: https://projects.spring.io/spring-statemachine/
- Please provide testing for your solution
- Providing an API Contract e.g. OPENAPI spec. is a big plus

Solution

Java 14, Spring State Machine, H2, Swagger


FORK
CHECK_IN
JOIN
ADDED
SC_STARTED
SC_FINISHED
WPC_STARTED
WPC_FINISHED
APPROVED
ACTIVE