SERVICE LAYER

Tracy Mazelin

RE-DESIGN DECISIONS

I was fairly thorough in my service layer planning and it already included my stretch features. There are no changes needed to my service layer at this time.

EMPLOYEE

1. Get a single employee:

- Purpose: This endpoint will be used by an employee to view their profile on the View Profile page of the application
- Request: GET {base_url}/employee/{id}
- Reponse code: HTTP 200
- Response body:

```
{
   "employee_id": 123,
   "manager_id": 123,
   "email": "test@test.com",
   "first_name": "Jo",
   "last_name": "Smith",
   "start_date": "01-01-2022",
 }
```

- Errors:
  - 404 Not found: returned when the employee id passed is not found
  - 401 Unauthorized: returned when the user is not authorized to view the employee id passed

2. Add an employee:

- Purpose: Used by an administrator to add a new employee
- Request: POST {base_url}/employee
- Response code: HTTP 201 Created
- Response body: true
- Request body:

```
{
   "manager_id": 123,
   "email": "test@test.com",
   "first_name": "Jo",
   "last_name": "Smith",
   "start_date": "01-01-2022",
   "is_admin": false,
 }
```

- Errors:
    - 400 Bad request: returned when the data passed does not pass validation checks
    - 401 Unauthorized: returned when the user does not have permission to create an employee

3. Update a single employee:

- Purpose: Used by an administrator to edit an employee
- Request: PUT {base_url}/employee/{id}
- Response body: true
- Response code: HTTP 200 OK
- Request body:

```
{
   "employee_id": 123,
   "manager_id": 123,
   "first_name": "Jo",
   "last_name": "Smith",
   "start_date": "01-01-2022",
   "is_admin": false,
}
```

- Response body:

```
{
   "employee_id": 123,
   "manager_id": 123,
   "user_id": 123,
   "email": "test@test.com",
   "first_name": "Jo",
   "last_name": "Smith",
   "start_date": "01-01-2022",
   "is_admin": false,
}
```

- Errors:

    - 400 Bad request: returned when the data passed does not pass validation checks
    - 401 Unauthorized: returned when the user does not have permission to create an employee

4. Delete an employee:
    - Purpose: Used by an administrator to remove an employee
    - Request: DELETE {base_url}/employee/{id}
    - Response code: HTTP 204 No content
    - Errors:
        - 400 Bad request: returned when the id passed does not exist
        - 401 Unauthorized: returned when the user does not have permission to delete an employee

5. Get a list of employees:
    - Purpose: Used by an hr manager to get a list of employees
    - Request: GET {base_url}/employees
    - Response code: HTTP 200 OK
    - Response body:

```
{
   "employees": [{
           "employee_id": 123,
           "manager_id": 123,
           "user_id": 123,
           "email": "test@test.com",
           "first_name": "Jo",
           "last_name": "Smith",
           "start_date": "01-01-2022",
           "is_admin": false,
           "created_date": "01-01-2022"
       },
       {
           "employee_id": 123,
           "manager_id": 123,
           "user_id": 123,
           "email": "test@test.com",
           "first_name": "Jo",
           "last_name": "Smith",
           "start_date": "01-01-2022",
           "is_admin": false,
           "created_date": "01-01-2022"
       }
   ]
}
```

- Errors:
  - 400 Bad request: returned when the id passed does not exist
  - 401 Unauthorized: returned when the user does not have permission to delete an employee

LEAVE TYPES

6. Get a list of leave types:

- Purpose: Used by an an employee to get a list of leave types when creating a leave request
- Request: GET {base_url}/leave_types
- Response code: HTTP 200 OK
- Request body: none
- Response body:

```
{
   "leave_types": [{
          "id": 1,
          "name": "PTO"
      }, {
          "id": 2,
          "name": "Sick"
      },
      {
          "id": 3,
          "name": "Bereavement"
      },
      {
          "id": 4,
          "name": "Jury Duty"
      }
   ]
}
```

- Errors:
    - 401 Unauthorized: returned when the user does not have permission to get leave types

LEAVE REQUEST

7. Create a leave request

- Purpose: Used by an employee to create a request for time off
- Request: POST {base_url}/employee/{id}/leave_request
- Response code: HTTP 201 Created
- Response body: true
- Request body:

```
{
   "employee_id": 123,
   "leave_type_id": 1,
   "approval_status_id": 3,
   "start_date": "2022-03-15",
   "end_date": "2022-03-18",
   "comment": "Family vacation"
}
```

- Errors:
    - 400 Bad request: returned when the data passed does meet validation requirements
    - 401 Unauthorized: returned if the user does not have permission to create a leave request for the given employee

8. Get leave requests by employee

- Purpose: Used by an employee to see the list of leave requests made
- Request: GET {base_url}/employee/{id}/leave_requests
- Response code: HTTP 200 OK
- Response body:

```
{
   "leave_requests": [{
       "id": 123,
       "employee_id": 123,
       "leave_type_id": 1,
       "approval_status_id": 3,
       "start_date": "2022-03-15",
       "end_date": "2022-03-18",
       "comment": "Family vacation",
        "created_date": "2022-03-15"
    }, {
        "id": 123,
        "employee_id": 123,
        "leave_type_id": 2,
        "approval_status_id": 1,
        "start_date": "2022-03-15",
        "end_date": "2022-03-18",
        "comment": "Not feeling well",
        "created_date": "2022-03-15"
    }]
}
```

- Errors:
    - 401 Unauthorized: returned if the user does not have permission to view the leave requests of the given employee

9. Get leave requests by manager

- Purpose: Used by a manager to see the list of leave requests for employees on the team
- Request: GET {base_url}/manager/{id}/leave_requests
- Response code: HTTP 200 OK
- Response body:

```
{
    "leave_requests": [{
        "id": 123,
        "employee_id": 123,
        "leave_type_id": 1,
        "approval_status_id": 3,
        "start_date": "2022-03-15",
        "end_date": "2022-03-18",
        "comment": "Family vacation",
        "created_date": "2022-03-15"
    }, {
        "id": 123,
        "employee_id": 123,
        "leave_type_id": 2,
        "approval_status_id": 1,
        "start_date": "2022-03-15",
        "end_date": "2022-03-18",
        "comment": "Not feeling well",
        "created_date": "2022-03-15"
    }]
}
```

- Errors:
    - 401 Unauthorized: returned if the user does not have permission to view the leave requests of the given manager

10. Approve or deny a leave request

- Purpose: Used by a supervisor to approve or deny a time off request
- Request: PUT {base_url}/leave_request/{id}
- Response body: true
- Response code: HTTP 200 OK
- Request body:

```
{
   "leave_request_id": 1,
   "leave_request_status": 2
}
```

- Errors:
    - 401 Unauthorized: returned if the user does not have permission to approve or deny a leave request

APPROVAL STATUSES

11. Get a list of approval statuses:

- Purpose: Used by a manager when approving or denying a leave request
- Request: GET {base_url}/approval_statuses
- Response code: HTTP 200 OK
- Response body:

```
{
   "approval_status": [{
           "id": 1,
           "name": "Pending"
       }, {
           "id": 2,
           "name": "Approved"
       },
       {
           "id": 3,
           "name": "Denied"
       }
   ]
}
```

- Errors:
    - 401 Unauthorized: returned if the user does not have permission to get a list of approval statuses

12. Get a list of managers

- Purpose: Used by an an hr administrator to get a list of managers when creating new employees
- Request: GET {base_url}/managers
- Response code: HTTP 200 OK
- Response body:

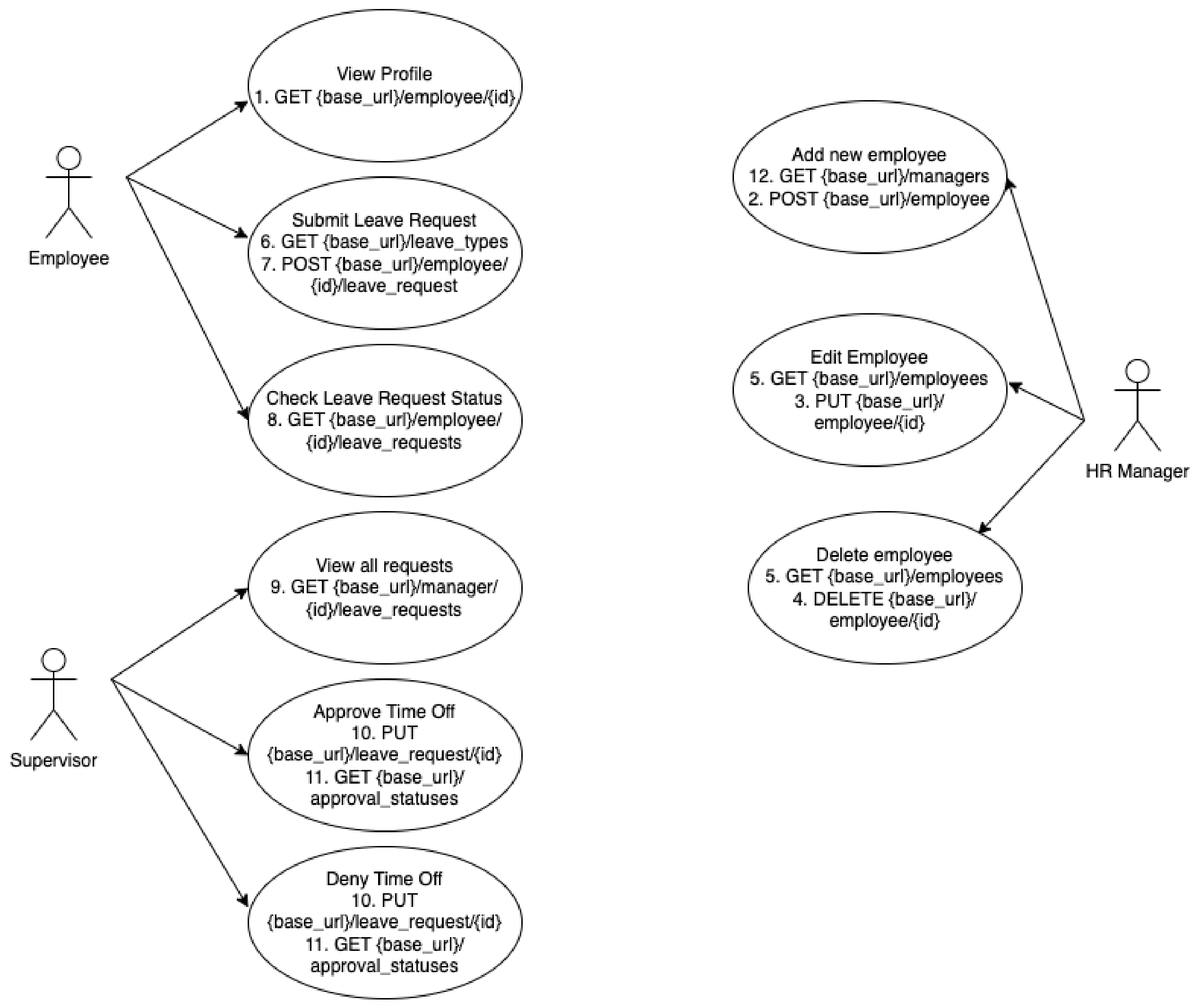
```
{
   "managers": [{
            "id": 1,
            "name": "Steve Smith"
        }, {
            "id": 2,
            "name": "Sarah Jones"
        },
        {
            "id": 3,
            "name": "Claire Baker"
        }
    ]
}
```

- Errors:
    - 401 Unauthorized: returned if the user does not have permission to get a list of managers

NOTE: I will be utilizing the flask_login package to handle authentication. Therefore, these methods are not included in this service layer document because they will be handled by the package.

DIAGRAM OF PAGES USING THE ABOVE SERVICE ENDPOINTS

NOTE: The numbers correspond to the above endpoint descriptions


Employee
View Profile
1. GET {base_url}/employee/{id}
Submit Leave Request
6. GET {base_url}/leave_types
7. POST {base_url}/employee/{id}/leave_request
Check Leave Request Status
8. GET {base_url}/employee/{id}/leave_requests
Supervisor
View all requests
9. GET {base_url}/manager/{id}/leave_requests
Approve Time Off
10. PUT {base_url}/leave_request/{id}
11. GET {base_url}/approval_statuses
Deny Time Off
10. PUT {base_url}/leave_request/{id}
11. GET {base_url}/approval_statuses
HR Manager
Add new employee
12. GET {base_url}/managers
2. POST {base_url}/employee
Edit Employee
5. GET {base_url}/employees
3. PUT {base_url}/employee/{id}
Delete employee
5. GET {base_url}/employees
4. DELETE {base_url}/employee/{id}