|  |  |
| --- | --- |
| **Login** | http://localhost:8080/login  Use admin for both un and pw, as startup login credentials |
|  | |

|  |  |
| --- | --- |
| **Roles** | http://localhost:8080/api/roles/  1. Your application should be able to add roles in the database dynamically in the db. |
|  | |

|  |  |
| --- | --- |
| **User Add** | http://localhost:8080/api/users/  2. Your application should be able to add Users in the db which can be used for authentication purposes. **(ROLE\_ADMIN)** |
|  | |
| **User Add** | http://localhost:8080/api/users/  2. Your application should be able to add Users in the db which can be used for authentication purposes. **(ROLE\_USER)** |
|  | |

|  |  |
| --- | --- |
| **Logout** | http://localhost:8080/logout  Logout of the super admin user |
|  | |

|  |  |
| --- | --- |
| **Login** | http://localhost:8080/login  Login of the newly created user with ROLE\_ADMIN |
|  | |

|  |  |
| --- | --- |
| **Employee Add** | http://localhost:8080/api/employees/  3. Now Your application should be able to add employees data in the db if and only if the authenticated user is ADMIN |
|  | |

|  |  |
| --- | --- |
| **List Employees** | http://localhost:8080/api/employees/  4. Your application should provide an endpoint to list all the employees stored in the database. |
|  | |

|  |  |
| --- | --- |
| **Fetch Employee** | http://localhost:8080/api/employees/3  5. Your application should provide endpoint to fetch an employee record specifically based on the id of that employee |
|  | |

|  |  |
| --- | --- |
| **Update Employee** | http://localhost:8080/api/employees/  Your application should provide an endpoint to update an existing employee with the given updated json object |
|  | |

|  |  |
| --- | --- |
| **Delete Employee** | http://localhost:8080/api/employees/1  7. Your application to have endpoint to delete an existing employee record based on the id of the employee |
|  | |

|  |  |
| --- | --- |
| **First name Fetch** | http://localhost:8080/api/employees/by-first-name/arn  8. Your application should fetch employee by his/her first name and if found more than one record then list them all |
|  | |

|  |  |
| --- | --- |
| **Employee Sort List** | http://localhost:8080/api/employees/?sort=asc  9a. Your application should be able to list all employee records sorted on their first name in **ascending order.** |
|  | |

|  |  |
| --- | --- |
| **Employee Sort List** | http://localhost:8080/api/employees/?sort=desc  9a. Your application should be able to list all employee records sorted on their first name in **descending order.** |
|  | |

|  |  |
| --- | --- |
| **ROLE\_USER** | http://localhost:8080/api/employees/  Establishing that a login with a ROLE\_USER is "Forbidden". |
|  | |

|  |  |
| --- | --- |
| **H2 Database** | H2 Console  H2 showing all the relevant databases |
|  | |