7.1P

Task 4.

1. . In the context of the employee’s CRUD operations on their own record, why the system does not allow employee to perform the “C” and “D” operations? Justify your answers.
2. If the company wants the employees to have a record, then there’s no reason to give them the ability to delete their record.
3. If a third party accessed an employee account, they could delete the employee’s records, or create a new one & attempt to pose as an employee and infiltrate the company
4. If an employee had a disciplinary mark put against their record they could potentially remove it by deleting & recreating their record.
5. . In the context of the employee’s Review operation, what information can be reviewed? Justify your answer.

The employees name, phone number, address, email, bank id & salary.  
This information can be viewed by the employee because it is strictly their own personal information, and could not be used by a third party to fraudulently pass as an employee.

1. In the context of employee’s Review operation (reviewing their own detail), the company decided to implement a DTO which excludes the password being sent to the client. Why password is excluded? Do you think that this is a good practice? Why or Why not? If not, propose an alternative and justify your choice.

The password is excluded because if the employee left the webpage open while logged in, someone could view the password and use it to login to the employees account and change their details.

I think this is good practice. Most computers will used in an environment with many other people, or the computers themselves are shared, and so its not possible to guarantee only one person can access the device.

1. In the context of the employee’s Update operation, What information can be updated? Are these the same as those in 4.2 and 4.3above? Why or Why not? Justify your answer.

The information that the employee should be able to update will be information that is relevant to the employee alone.

So this would be phone, email, address, and password.

The employee must be able to change their password as only the employee should know their password. This way if the administrator wished to alter the employees data they would have to do so via their admin account which would keep them accountable.

1. What information cannot be updated? How would you avoid these data being updated by the employee “accidentally”?

The employee ID, name, app group, and their salary as these fields relate to the business and changes to them would affect other members in the organization.

The name cannot be updated as it’s a relevant piece of information for the employer to authenticate the user’s identity.

These would be presented as read-only in the UI to the user, and on the admin forms the admins password could be required.

1. In the context of the employee’s Update operation, where should the actual change of the employee’s information occur? Do you think this is a good practice? Why or Why not? Justify your answer.

This would occur in a separate set of xml pages for employees only. These would have some fields read-only and only the functions that employees require.

I think this is good practice as it completely separates the user interfaces & with some re-writing of the existing code, separates the beans used.

1. In the context of employee’s Update operation, the company decided to first display the details of a particular employee (if such employee exists after searching through the database via the employee’s id) in the web browser so that the employee could enter the required information. Should the existing password be

(1) sent and displayed to the client?

(2) sent to the client but not displayed?

(3) not sent?

What is your choice? Why or why not? Justify your answer. If your answer is (3), how would you implement the feature that allows the employee to change their own password?

It should be (3) not sent, as its unnecessary and only creates a security risk. The password should only be sent in one direction. From the client, to the database. The only information that should be sent back is whether the password was correct or not (true/false).

I would just copy current existing methods for changing passwords in which you enter the current password, then the new password, and lastly a confirm password. Then if the old password is correct it updates the record with the new password.

1. In the context of deleting employee’s record, the company choose to accept the employee id as the input and then remove the employee record by setting the field “active” to false instead of removing the record from the database. Do you think that this is a good practice? Why or Why not? If not, propose an alternative and justify your choice.

There are several business reasons for keeping a record of employees who have previously been in the organisation. However, by keeping them in the database there is the security risk that an account could be reactivated and used by a third party.

Instead of having the deactivated employees remain in the employee database I would create a second database of ex employees and create a bean to search through and migrate deactivated employees to the database. The deactivated employee database would not have a password column, and so the users previous password would be lost upon migration. To migrate the employee back to the active employee database the administrator would have to manually create a password for the reactivated employee. This would prevent a third party from using any automated process to reactivate a deactivated employee.

1. After reviewing the features provided by the application “ED-Secure”, do you think that the application as is can provide the features listed in the Case Study Section above? Are there any deficiencies? Why and why not? What changes would you suggest to address all features listed?

As it stands the application cannot offer two features:

* The employee can only view & edit their own details

A separate method for retrieving an employees details needs to be created that only a user can access.

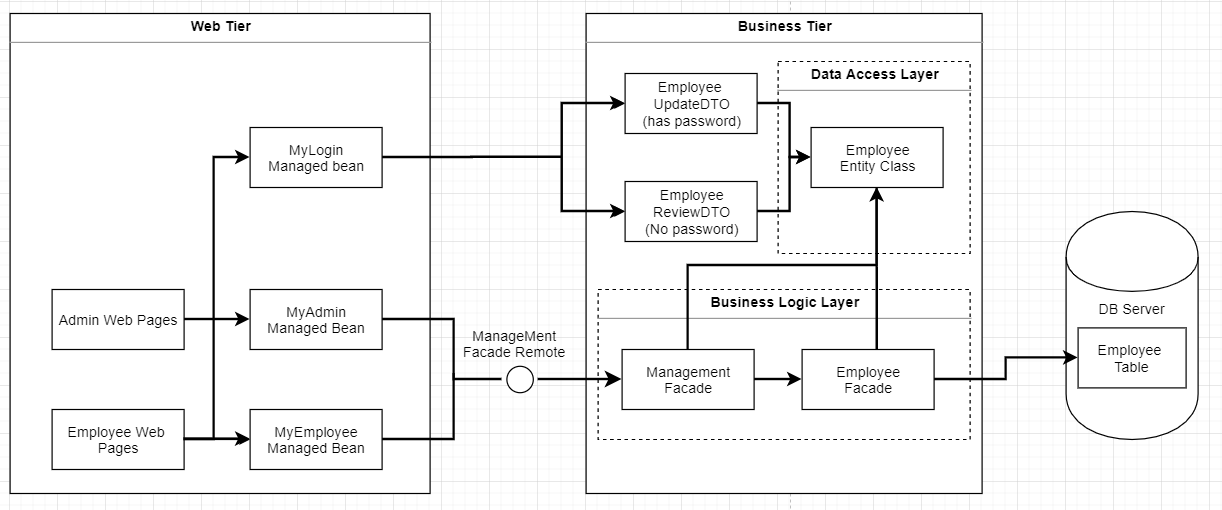
* The application will not display the users password.

The usermanage bean would need to have its functionality separated into 2 beans: one for user & admin tasks, and one for admin only tasks. A separate user DTO would need to be created for updating & retrieving user details.

For BOTH of these a new set of xhtml web pages would need to be created that call their respective beans.

Task 5.

1. UML Diagram



1. Component Descriptions

* Admin Web Pages

The set of jsf webpages that provide the UI for the admin tasks to be completed in the application

* Employee web pages

JSF web pages in which the employee can navigate to review and update their details.

* MyAdmin Managed Bean

Holds session data & provides functionality specific to admin pages.

* MyEmployee Managed Bean

Holds session data & provides functionality specific to employee pages. Notably different to the admin managed bean as it will lack methods for adding & deleting users, and will only retrieve data for the current user.

* Mylogin Managed Bean

Provides functionality for logging the user out of their login session.

* Management Façade