Low Level Design (add EMPLOYEE module)

|  |  |
| --- | --- |
| **Project Code** | GROUP-1 |
| **Project Name** | Expense Management System |

|  |  |  |
| --- | --- | --- |
| **Prepared/Modified by** | **Role** | **Date of Preparation** |
| H.MIDHUN KUMAR. | Software Engineer/ANALYST | 29/08/2018 |
| **Reviewed by** | **Role** | **Date of Review** |
|  | Software Engineer | 29/08/2018 |
| **Approved by** | **Role** | **Date of Approval** |
|  | Training Manager |  |
| **Circulation List** | [nevil.nevil@capgemini.com](mailto:nevil.nevil@capgemini.com)  [midhun.kumar-h@capgemini.com](mailto:midhun.kumar-h@capgemini.com)  [thotta.balaji.kumar@capgemini.com](mailto:thotta.balaji.kumar@capgemini.com)  [aishwarya.mankar@capgemini.com](mailto:aishwarya.mankar@capgemini.com)  [tripti.verma@capgemini.com](mailto:tripti.verma@capgemini.com)  [veena.kamat@capgemini.com](mailto:veena.kamat@capgemini.com)  [faizahmed.mullani@capgemini.com](mailto:faizahmed.mullani@capgemini.com)  [minaj.shaikh@capgemini.com](mailto:minaj.shaikh@capgemini.com)  [pragathi.pragathi@capgemini.com](mailto:pragathi.pragathi@capgemini.com) | **Version Number of the template:1.0** |
| **Version Number** | 1.3 |  |

Table of Contents

[1. Description Of The Unit 3](#_Toc306116213)

[2. References 4](#_Toc306116214)

[3. Dependencies / Assumptions 5](#_Toc306116215)

[4. Class Diagrams 6](#_Toc306116216)

[5. Sequence Diagrams 7](#_Toc306116217)

[6. Design Components 8](#_Toc306116218)

[6.1 List Of UI Components 8](#_Toc306116219)

[6.2 List Of Design Classes 8](#_Toc306116220)

[7. Re-Usable Routines/Components 9](#_Toc306116221)

[8. Messages 10](#_Toc306116222)

[9. Configuration Files 11](#_Toc306116223)

[10. Crud Matrix Table 12](#_Toc306116224)

[11. Data Mapping Matrix 13](#_Toc306116225)

[12. Action Flow Matrix 14](#_Toc306116226)

[13. Stored Procedures 15](#_Toc306116227)

[14. Program Specification / Pseudo Code 16](#_Toc306116228)

[REVISION HISTORY OF THE WORK PRODUCT 17](#_Toc306116229)

# Description Of The Unit

The add employee module allows the admin of the application to add new employees into the database. This module can be used only by the admin of the application.

The insertion of the employee details is done using the Postman and when we view the details in JSP page, we get the details inserted. The admin enters all the details of the employee like employee name, employee designation, employee domain, employee salary, employee emailid, employee gender, and employee password. If the entered employee details are invalid, appropriate error messages are displayed.

After entering all the employee details, the details are inserted into the MySQL database on clicking the Post button in Postman.

# References

| **Reference** | **Description** |
| --- | --- |
| Class Diagram | PLP\PLP Documents\ProjectDocuments\DesignDocuments\HLD\Class Diagram |
| URS | PLP\PLP Documents\ProjectDocuments\Requirements |
| UTP | PLP\PLP Documents\Testing\UTP |
| DB Design | PLP\PLP Documents\ProjectDocuments\DesignDocuments\HLD\DB Design |

# Dependencies / Assumptions

**Dependencies:**

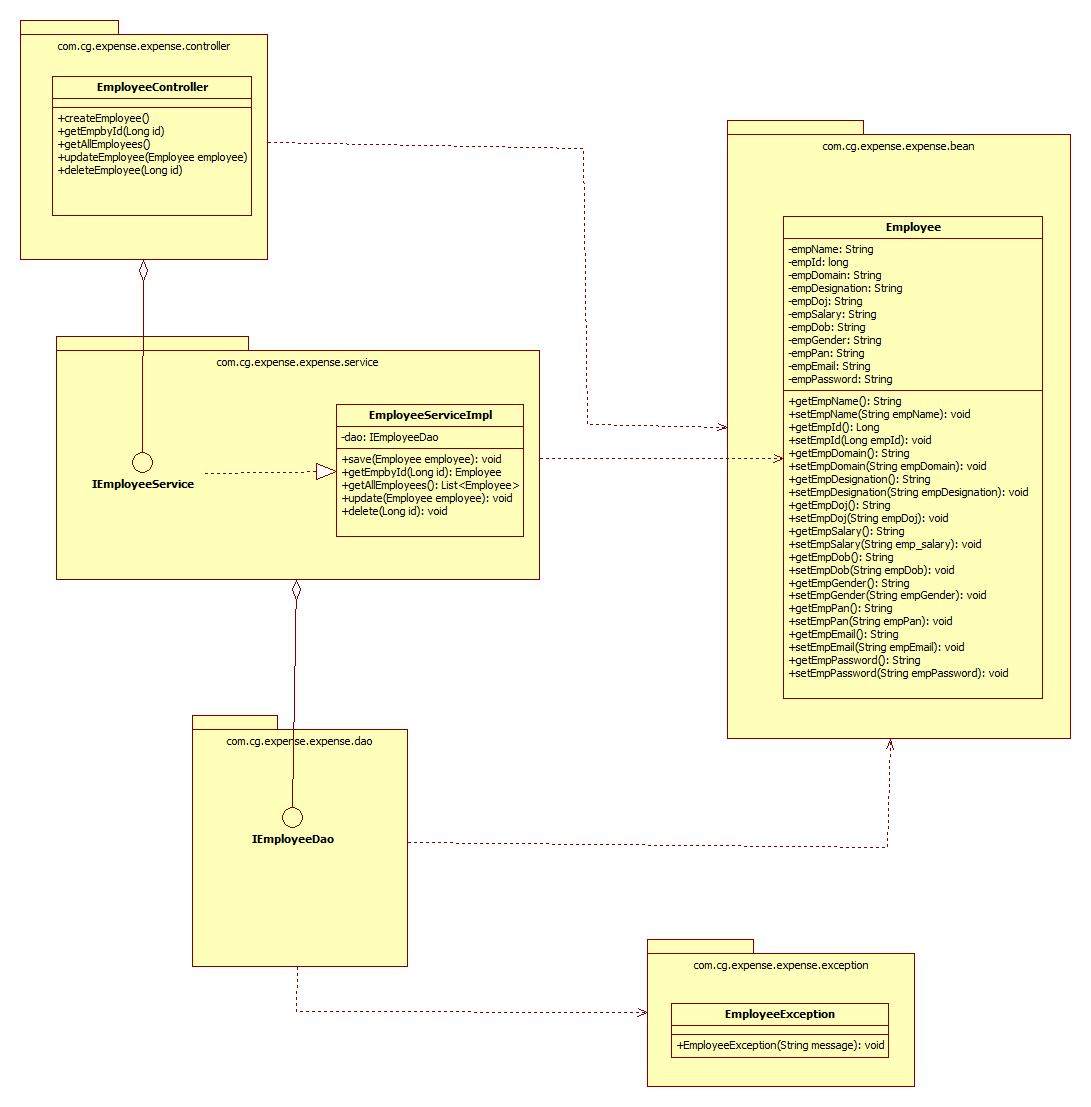
The employee should be logged in.

**Assumptions:**

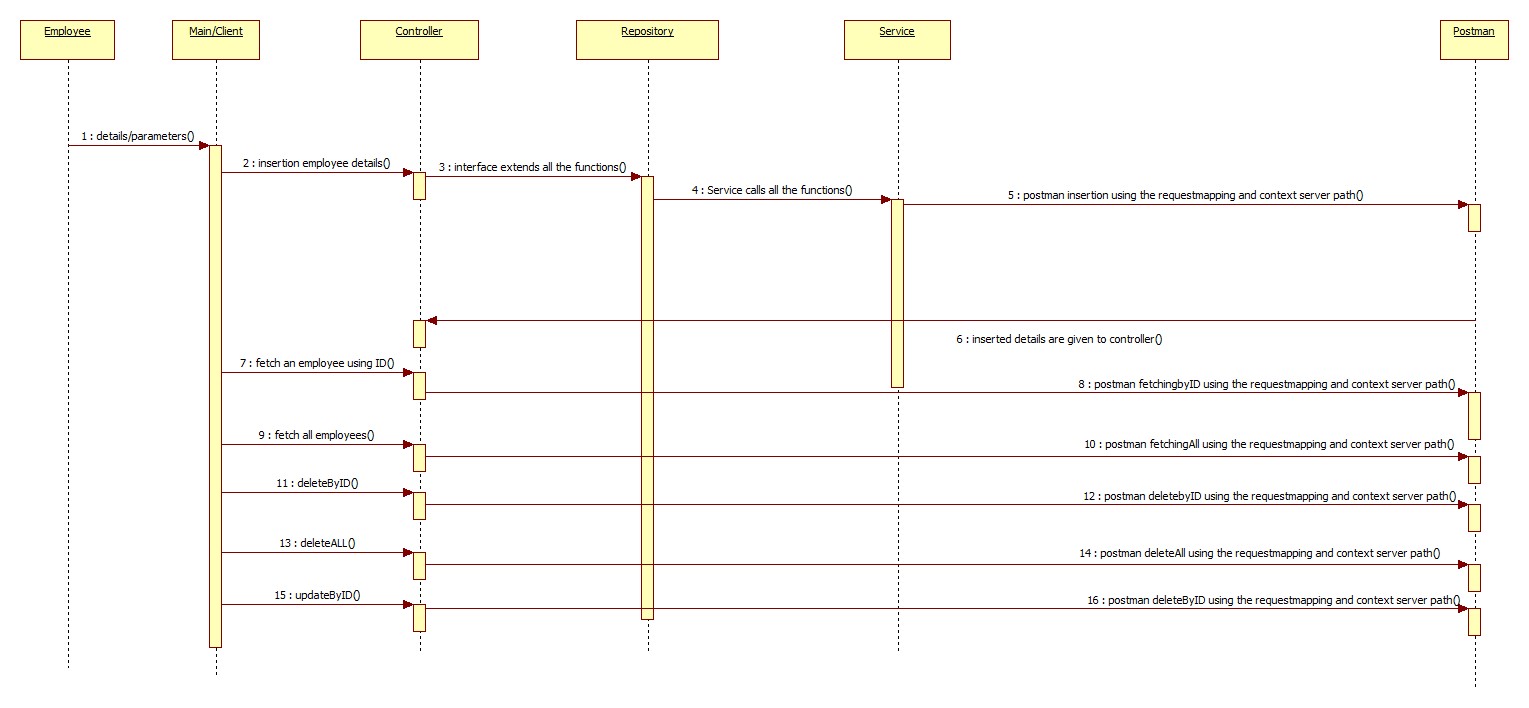
The employee details are stored in the MySQL database.

# Class Diagrams

[Class Diagram](../../HLD/Class%20Diagrams)

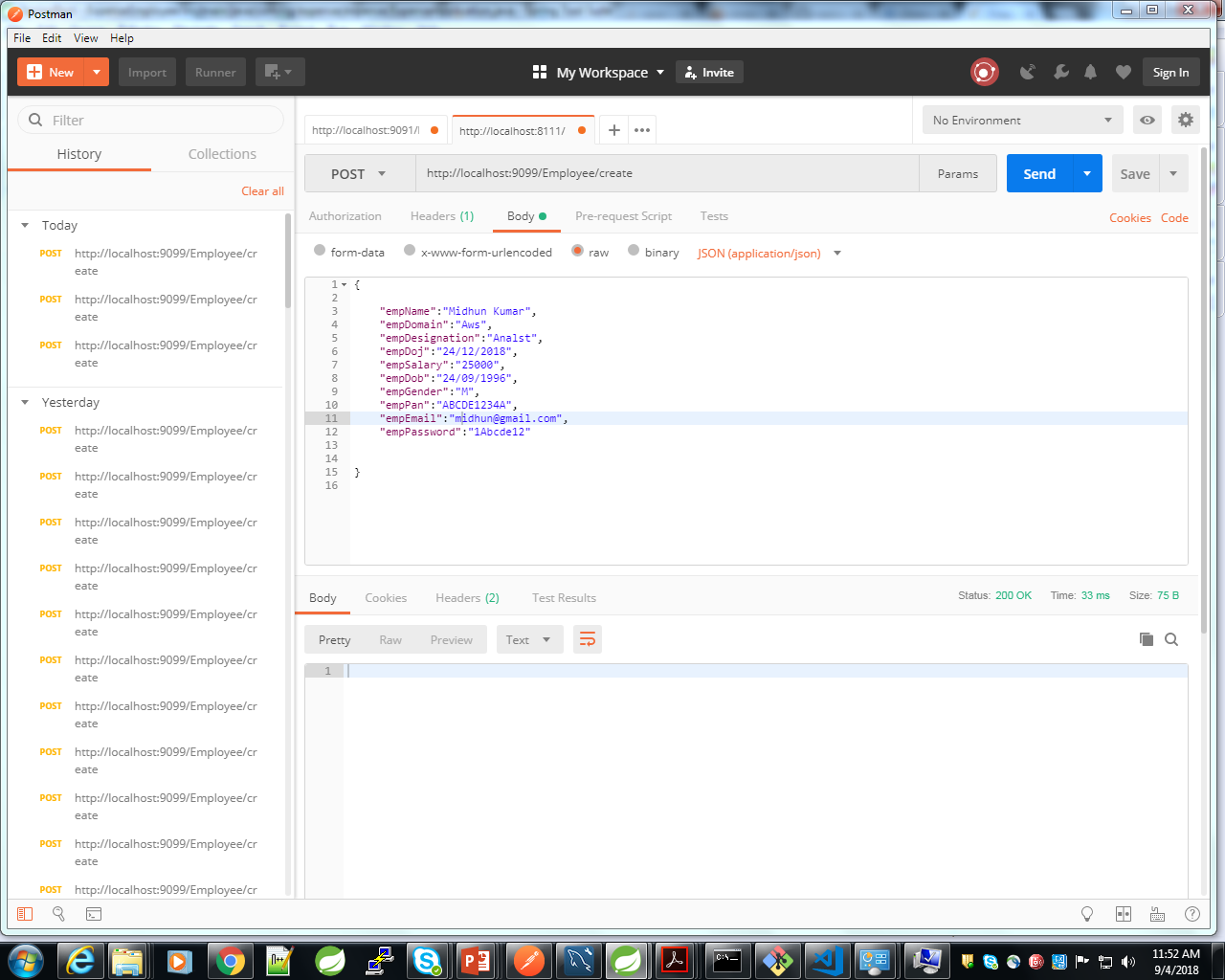


# Sequence Diagrams



# Design Components

## List Of UI Components



## List Of Design Classes

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sr. No.** | **LLD Name** | **Class/Interface Name** | **Method Name** | **Method Description** |
| 1 | Add EmployeeModule | Employee | Appropriate getters and setters methods | Getters retrieve the property values and setters set the property values |
| 2 | Add EmployeeModule | EmployeeController | **public** **void** createEmployee(@RequestBody ExpApp expapp) | Sets the model attribute.  The controller function is called by the EmployeeServiceImpl and the repository IEmployeeDao acts as an interface.Using Postman, we insert the details of the employee. |
| 3 | Add EmployeeModule | EmployeeServiceImpl | **public** **void** save(ExpApp expapp) | Calls the DAO layer method to insert the employee details in to the database |
| 4 | Add EmployeeModule | IEmployeeDao | **public** **interface** IEmployeeDao **extends** JpaRepository<ExpApp, Long> | Inserts the employee information into the database and returns the result to EmployeeServiceImpl  . |

# Re-Usable Routines/Components

Re-Usable Routines/Components

* JSP files
* Pom.xml files
* EmployeeException class
* Database-MySQL properties files
* Logger properties files
* Postman

# Messages

|  |  |  |
| --- | --- | --- |
| **#.** | **Type** | **Message Text** |
| M1 | Error | Name should be atleast Three Characters |
| M2 | Error | Employee Domain should Start with a capital letter followed by combination of lowercase and uppercase letters within range of 3 to 20 characters. |
| M3 | Error | Employee Designation should Start with a capital letter followed by combination of lowercase and uppercase letters within range of 3 to 30 characters. |
| M4 | Error | Salary should be in positive numbers |
| M5 | Error | Gender should be valid Either (M,F,O) or (Male or Female or Other) |
| M6 | Error | PAN number should be in format ABCDE1234A |
| M7 | Error | EMAIL should be in format of [abc@capgemini.com] |
| M8 | Error | Must contain at least one number and one uppercase and lowercase letter, and at least 8 or more characters |
| M9 | Error | Validation Failed |

# Configuration Files

|  |  |  |
| --- | --- | --- |
| File Name | Path | Description |
| application.properties | /ExpenseEmployee/src/main/resources/application.properties | To configure database connection. |
| pom.xml | /ExpenseEmployee/pom.xml | To configure the bean properties. |
| Postman | localhost:9098/Employee/employees | To configure the request mapping and view resolver. |

# Crud Matrix Table

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **#** | **Table Name** | **Create** | **Read** | **Update** | **Delete** |
| 1. | Employee | Y | N | N | N |

# Data Mapping Matrix

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Sr. No.** | **Screen Label** | **Field Source** | **Field Type** | **Data Type** | **Max Length** | **Data Flow** | **Mandatory** |
| 1 | Employee ID | Employee.emp\_id | Label | Number | NA | UI🡪D | Yes |
| 2 | EmployeeName | Employee.emp\_name | Label | Varchar2 | 100 | UI🡪D | Yes |
| 3 | EmployeeDesignation | Employee.emp\_designation | Label | Varchar2 | 100 | UI🡪D | Yes |
| 4 | EmployeeDomain | Employee.emp\_domain | Label | Varchar2 | 100 | UI🡪D | Yes |
| 5 | EmployeeDOJ | Employee.emp\_doj | Label | Date | NA | UI🡪D | Yes |
| 6 | Employee Salary | Employee.emp\_salary | Label | Number | NA | UI🡪D | Yes |
| 7 | EmployeeDOB | Employee.emp\_dob | Label | Date | NA | UI🡪D | Yes |
| 8 | EmployeeGender | Employee.emp\_gender | Label | Varchar2 | 100 | UI🡪D | Yes |
| 9 | EmployeePAN | Employee.emp\_pan | Label | Varchar2 | 100 | UI🡪D | Yes |
| 10 | EmployeeMailID | Employee.emp\_mailid | Label | Varchar2 | 100 | UI🡪D | Yes |
| 11 | EmployeePassword | Employee.emp\_password | Label | Varchar2 | 100 | UI🡪D | Yes |
| 13 | Create | Postman | Button | NA | NA | PM🡪UI | Yes |
| 14 | Cancel | Postman | Button | NA | NA | PM🡪UI | Yes |

**Legend**

D 🡪 UI (Database to User Interface)

UI 🡪 D (User Interface to Database)

UI (User Interface Only)

PM🡪UI (Postman to User Interface)

# Action Flow Matrix

|  |  |  |  |
| --- | --- | --- | --- |
| **Sr. No.** | **Action** | **Description** | **Process** |
| 1 | On Running the code in the console. | Spring Application starts. | The admin clicks on the Run button in the Spring and console generates with the success message.Then open the Postman. |
| 2 | On changing the action panel to Post and entering the details. | The employee details are inserted into the database. | Data is entered in the fields which gets inserted into the database using Postman. |

# Stored Procedures

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **#** | **Stored Procedure Name**  **(with Description)** | **Input/Out Parameters** | **Type** | **Size** |
| 1. | NA | NA | NA | NA |

# Program Specification / Pseudo Code

## <ACTION :Create\_Employee >

**Pseudo Code:**

1. **Client Side Scripting:**

None

1. **Server Side Scripting:**

**BEGIN**

**Controller:**

@PostMapping("/employees")

**public** **void** createEmployee(@Valid @RequestBody Employee Employee) {

service.save(Employee);

}

**Service:**

@Transactional

**public** **void** save(Employee Employee) **throws** EmployeeException{

dao.save(Employee);

}

**END**

**14. 2 <ACTION: CreateEmployee\_CLICK>**

**Pseudo Code:**

1. **Client Side Scripting:**

IF invalid EmployeeName

Print Error Message M1

IF invalid EmployeeDomain

Print Error Message M2

IF invalid EmployeeDesignation

Print Error Message M3

IF invalid EmployeeSalary

Print Error Message M4

IF invalid EmployeeGender

Print Error Message M5

IF invalid EmployeePAN

Print Error Message M6

IF invalid EmployeeEmail

Print Error Message M7

IF invalid EmployeePassword

Print Error Message M8

IF invalid Validations

Print Error Message M9

1. **Server Side Scripting:**

**BEGIN**

Call createEmployee(@Valid @RequestBody Employee Employee) from controller to service

**BEGIN**

Call save(Employee Employee) **throws** EmployeeException to Repository interface from Service Layer

**BEGIN**

Insertion of details are done through Postman.

**END**

Return Details. to Service Layer.

**END**

Return Details to Controller Layer.

**END**

IF error during Insertion occurs, then show error message.

**14. 2 <ACTION: CANCEL\_CLICK>**

**Pseudo Code:**

**1. Client Side Scripting:**

None

**2. Server Side Scripting:**

**BEGIN**

Call createEmployee(@Valid @RequestBody Employee Employee) from controller to service

Insertion of details are done through Postman.

**END**

REVISION HISTORY OF THE WORK PRODUCT

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | Date | Version # | Section Changed | Details of changes made | Approved By |
| 1 | 29/08/2018 | 1.0 | LLD prepared | N/A |  |
| 2 | 29/08/2018 | 1.1 | Reusable components,  Pseudo code | Reusable components added.  The pseudo code is modified as per the specified flow. |  |
| 3 | 29/08/2018 | 1.2 | Reusable components,  References,  Data Mapping Matrix. | Reusable Components added.  File name with relative path is added in the references section.  The name of text boxes(screen label is changed). |  |
| 4 | 29/08/2018 | 1.3 | Configuration files,UI Components | Paths of the configuration files is updated and UI components(Screen Shots) added. |  |