The most important aspect of creating a successful application is to get its Database right. Creating an effective database schema and defining relations are very important for a web application.

Relational tables for “Employee Vacation/Leave” Request System.

**Employee table**  **Holidays**

1. Employee id (PK) 1) weekid (PK)
2. Employee name 2) Employee Id (FK)
3. Employee department 3) Total hours worked until now
4. Employee Joining Date 4) carried over holidays of previous year
5. Employee Department 5) Total Holidays earned in the year
6. Employee Manager id etc….
7. Employee role

Etc…..

**Leave table**

1. Leave id (PK)
2. Employee id (FK)
3. Start date of holiday
4. End date of holiday
5. Type of Holiday
6. Status of leave
7. Holidays remaining
8. Record-Status
9. Timestamp
10. Date of request
11. Comments.
12. Comments from manager

Etc …..

**Step 1)**  Create the business requirements like rules and restrictions of the Employee vacation/leave. For example Max number of vacation days a person can take at one time or if the person exceeds over the paid time leave what approach should the manager take etc.

Based upon the above rules, create the relational tables or entities. For this example

I am using only 3 tables to make it look simple and to keep track of the system (employee, holidays and leave table). In real functional application it would take much more tables than these 3.

**Employee** table keeps track of employees and **Holiday** table keeps track of holidays earned until now and finally **leave** table keeps track of all the leave requests and status of the requests.

**Step 2)** design a prototype for the front-end with necessary fields as per the requirements.

In this case I would create 2 views one for the Manager and one for the Employee.

An employee would login to the portal and either applies for the leave or track the status of the leave.

A manager would login and can do the same but also can approve or disapprove leave for the employees who work under her.

Example

Employee name :: leave id ::

Department :: Date ::

No of leaves left ::

Starting date of Vacation :: calendar

Ending Date of Vacation :: calendar

Reason ::

Request to Manager :: drop down

Etc …………………………….

And also we create a view with respect to manager:: showing the list of people who applied for vacation that works under the respective manager.

Determine whether the application needs synchronous or asynchronous calls to the server. In this case, I would prefer synchronous calls as I can see only forms to be submitted to the server.

So here I go with just JSP, Java script, JQuery and CSS for front end technologies.

**Step 3:** Choosing a language for coding, in this case I prefer java and use spring MVC which I am comfortable with and also try to use ORM framework or JDBC for persisting the data into relational tables.

1. After choosing a framework, divide the application into web layer, service layer, Business layer, persistence layer and reporting layer this way coding will be much easier and modular.
2. Identify the no of users for the application. In this case I would be prepared for the maximum users of around 3000. And also decide on how many concurrent users can access this application and determine whether caching is required or not.
3. Application security should be well established for both internet and intranet applications.
4. Make the application as light weight as possible.

If it’s a standalone application I prefer to make it as much light weight as possible, without the need of application server. IF it’s not a standalone application then we can incorporate into existing application if we are already using it.

Service layer can be exposed either as a Soap webservice, restfull webservice or java Service. In this case I would prefer to use java service as we do not require to provide services to 3rd party outside the company.

Step 4:: make it extensible and modular.