|  |  |
| --- | --- |
| **User stories serial** | **View required to accomplish the task** |
|  | Page: 2 combo-boxes to for selecting training program as well as associated employees. The combo-boxes should be able to create a many to many relationships between the employees and training programs. Search bar needs to be there for both employees and training programs (in case of large lists). The search will use AJAX to update the available information. The page is to be built inside common layout for HR.  Input: A list of available training programs and a list of eligible employees  Output: Insert query or equivalent function call to map the selected employees to the selected training programs. Data shall be inserted into 2 different tables, one containing training program and associated employees, another containing employee and associated training programs.  Entities involved: Employee, TrainingPrograms, EmployeeAndTraining, TrainingAndEmployee |
|  | Page: A search bar on top left, a list of training programs on the left side of the page. Tick-boxes for each training program. On selecting a training program (with or without clicking tick-box), display training program details on the right side of the page. In the details of the training program, display training name, training period and requirements (if any) for attending the training session. On key press in the search bar, use AJAX to update the training program list. Disable tick-boxes in the list for programs that the user is not eligible for. The page is to be built inside common layout for employees.  Input: Employee ID, a list containing training programs  Output: Query or equivalent function call to update the list of training applications  Entities involved: TrainingPrograms, EmployeeTrainingApplication |
|  | Page: Multiple tables one below the other. Each table represents a training program. The associated employee names shall form the rows while the performance evaluation metric form the columns. The page is to be built inside common layout for employees.  Input: Training programs list and their associated employees.  Output: Display of the incoming inputs  Entities involved: TrainingAndEmployee |
|  | Page: A search bar on top left. A table below the search bar. The table shall contain name of training program and associated budget. On selecting a training program, display training program details on the right side of the page. In the details of the training program, display training name, training period and budget (in big/distinct manner) with an approve button. Pressing the approve button opens an alert window with yes/no choice. On key press in the search bar, use AJAX to update the training program list. The page is to be built inside common layout for HR.  Input: A list of training programs  Output: Update query or equivalent function call to change the status of training program to approved.  Entities involved: TrainingPrograms |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: 2 buttons on top separated by at least 20 scaled pixels. One button shall be “Legal reports” while the other button should be terms and policies. An empty div should occupy the rest of the page. Clicking on Legal reports button should fetch the list of legal reports and display them in a list view inside the div. The ListView should contain a heading and a part of the description. Clicking on a ListView item shall open a sub window (pop-up window) and display details of the legal report. Clicking on Terms and policies should fill the empty div with a text area and a copy button. The text area shall contain the “Terms and Policies” of the organization while the copy button shall copy the terms and policies to the clipboard. The page is to be built under common layout of legal representative.  Input: A list of legal reports, Terms and Policies List  Output: No special output  Entities involved: LegalReports, TermsAndPolicies |
|  | Page: Search bar on top left. A ListView below the search bar on the left side. The ListView should contain a minified picture of the employee, employee name, employee age, salary and a review button. Clicking on an employee shall display the details of the employee (Picture, Name, Age, Salary, Service age, contact info, description etc.) and a review button below. Clicking on Review will move the employee status to under review. Typing in search bar shall update the listview asynchronously. The page is to be built under common layout of legal representative.  Redirect to appropriate employee management page  Input: List of employees  Output: Update query or equivalent function call to change status of an employee.  Entities involved: Employees |
|  | Page: A list-view with an add button below it. On page load, the list-view is populated with existing requests. Pressing add button opens up a sub-window with a field for number of employees, a field for experience level of employees, a field for preferred gender and a field for the earliest time at which the employees are needed and a field for urgency of the requests, a field for description of expertise required, a field of number of years of experience required and a confirm button. Pressing confirm button does a function call for insertion of data and returns to parent page while updating the info in parent page.  Input: List of employee requests  Output: No special output  Entities Involved: EmployeeRequests |
|  | Page: A form for the entire page along with submit button. The form fills up transport details such as Transport name, Type, Model, Cost of purchase, assign driver(nullable) and status (‘Ordered’,’Active’,’Busy’). “Assign driver” field should be a dropdown list containing the names of available drivers. On submit button click, validate the form and also check if any name in DB matches the transport name. Submit form or display user message as per validation and verification. The page is to be built under common employee layout with and additional link for this page.  Input: A form collection and a list of employees  Output: An insert query or equivalent function call to add the transport to the Transports list  Entities involved: Employees, Transports |
|  | Page: A dropdown-list for area. Selecting an item in dropdown list shows the number of employees in the area and the current transport capacity in the area on the right side of the page.. Below the Number of employees, a list view displays transports that are assigned to the area. Below the listview, an “Add Transports” button is displayed. Pressing “Add Transports” opens a subwindow containing available transport names in a dropdown list and a disabled confirm button. Selecting a transport name in sub-window shows the number of seats in that transport and enable confirm button. Pressing confirm button navigates back to the main page and updates the listview with the newly assigned transport name and also increase the current transport capacity in the area. Pressing “confirm” on subpage also makes function call to add the entry to AreaTransports table. If the function call of “confirm” button in subwindow does not work, do not add the transport name to the list but display an error message instead. The page is to be built under common employee layout with and additional link for this page.  Input: A list containing number of employees in each area, A list of transports assigned to each area, a list of available transports  Output: No specific output other than the function call from the sub page.  Entities involved: Employees, AreaTransports, Transports |
|  | Page: A dropdown-list containing names of all vendors (and their id’s in hidden html tag). Selecting a name from the list displays 2 tables one below the other. First table contains list of all equipment’s etc. provided by the vendor with an “Add Equipment” button below the table. Pressing “Add Equipment” opens up a subpage with the vendor name on top and a form for adding equipment details with a “Confirm” button below. Pressing confirm button performs function call for insert query and updates the main page first table with new entry. Second table is similar to first table with “Vendor Transactions” in place of “Vendor equipments”.  Input: 3 lists containing vendors, equipments provided by vendors and transactions involved with vendors  Output: No output.  Entities involved: Vendors, VendorEquipments, VendorTransactions |
|  |  |
|  | Page: A dropdown list with 4 items (‘Canteen food’, ‘Raw materials’,’Expenses’,’Transactions’) with an empty div below for diplaying additional information. Selecting ‘Canteen food’ and ‘Raw materials’ displays the respective lists in a table. Selecting expenses displays a listview of expense title, description and cost. Selecting Transactions displays a listview of transactions with and add button below. Pressing add button opens up a sub-page with form for adding transaction and a confirm button. Pressing confirm performs a function call for insert query and updates the list of transactions in the main page.  Input: A list of available canteen food, a list of raw materials in canteen, a list of all canteen transactions  Output: Insert query of the subpage  Entities involved: CanteenFood, CanteenRawMaterials, CanteenTransactions |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: A table containing list of all employees. Employee names are in rows while their performance metric are in columns. An extra column is present with recommend button. Pressing recommend button removes the button and sends an asynchronous request for update of employee recommendation. The table should be sort-able by any column.  Input: List of employee performance metric  Output: No special output  Entities Involved: Employees, EmployeeEvaluationMetric |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: A table containing record of all employees on leave along with leave related information and a link of the leave application. The information should be grouped by leave type.  Input: A list of employees on leave and associated data  Output: No special output  Entities Involved: Employees, EmployeeLeave |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: A List-View of all leave applications along with associated employee and eligibility of the employee. The List-View should have button for approval. Clicking a List-View item opens up a popup window containing the leave application. Pressing confirm button removes the List-View item and sends an asynchronous request to update the employee leave status and associated data.  Input: List of all leave application  Output: No special input  Entities Involved: Employees, EmployeeLeaveRequests |
|  | The previous entry does the work |
|  | Attendance is to be recorded automatically |
|  | Page: A select containing choices (‘Weekly attendance’, ‘Monthly attendance’, ‘Per Year Attendance’, ‘Yearly Attendance’). Selecting weekly, monthly and yearly attendance opens up a new select containing values for respective week/month/year. Changing select value generates a table containing associated data. Employee names are in rows and attendance related info is in columns.  Input: List of employee attendance record.  Output: No special output  Entities Involved: Employees, EmployeeAttendance |
|  | Page: Same as item #8 with implementation for salesmen  Input: Refer item 8  Output: Refer item 8  Entities Involved: Refer item 8 |
|  | Page: A table view of all employees who are work for sales department. The table should contain all employee related information  Input: A list containing employee information  Output: No special output  Entities Involved: Employees |
|  | Page: Available funds from company displayed on top. A List-View below the funds figure containing the hire requests sorted by urgency level. Each List-View item should display approximate cost of the hire request and a confirmation button. Clicking on confirmation button removes the List-View item and sends an asynchronous call to perform appropriate update operations.  Input: List of all hire requests  Output: No special output  Entities Involved: EmployeeRequests, ApprovedEmployeeRequests |
|  | Page: A list of all approved hiring requests with a textbox beside each item. Each item should also have an “allocate” button. Pressing allocate button allocates the amount specified in textbox to the hire request and deducts from available funds using asynchronous request. The total available funds are displayed on top.  Input: A list of all approved hiring request  Output: No special output  Entities Involved: ApprovedEmployeeRequests |
|  | Page: A select list containing names of available layouts for circulars. Selecting a name opens an instance of the circular in an empty div open for editing. A button below the layout, “Create” is present. Pressing “Create” button creates the circular as per the data entered (by sending an asynchronous request) and moves to next page. The next page contains a List-View of advertisers with more prominent/useful ones at the top and “Add” button for each advertiser. The available funds for the recruitment is displayed on top. Pressing “Add” in the list item removes the item from the list and deducts funds if applicable from available funds. An asynchronous call is sent to update the advertisers list for the particular job advertisement.  Input: A list of available job circular layouts, A list of advertisers.  Output: No special output  Entities Involved: Circulars, LayoutForCirculars, AdvertisersForCirculars, Advertisers |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: A table view of all advertisers and edit button corresponding to each advertisers.  Input: List of all advertisers  Output: An update query or equivalent function call to update the list of advertisers  Entities Involved: Advertisers |
|  | Page:  Input:  Output:  Entities Involved: |
|  | Page: Number of applicants awaiting slot assignment for interview displayed at the top. Below that, a list-view of all upcoming interview slots. Hovering mouse over each item shows the number of applicants assigned to that slot and their names. Each list item has an edit button. Pressing edit button allows changing slot timing and changing applicants for that slot in a sub window. An add button below the list-view is present. Pressing add button allows addition of new slots in a sub window. Adding new slots checks for time clash. On pressing confirm in sub window, the new slot is added to the main page and an update operation performed using asynchronous function call.  Input: A list of applicants, a list of time slots  Output: No special output  Entities Involved: InterviewSlots, SlotApplicants, ShortListedCandidates |
|  | Page: An add page for the entity employees  Input: A form collection object  Output: An insert query or equivalent function call to update employees list  Entities Involved: Employees |
|  | Page: A table containing employee names and respective amount/funds they are entitled to. The employee names are in rows while items like basic salary, annual bonus, provident fund etc. are in columns.  Input: A list of employees, list of company benefits  Output: No special output  Entities Involved: Employees, SalaryStructure |
|  | Page: A list of all employees who have not been paid in the current month as well as the amount they are supposed to get. List items should have a “Pay” button which performs asynchronous call to pay employee salary and remove the item from the list.  Input: A list of all employees, a list of all employees who have been paid  Output: If the database does not have a table for the current month/week payment, the table will be created automatically.  Entities Involved: Employees, EmployeeSalary*Month/WeekName* |
|  | Page: 2 select tabs with multiple options for time period (month to month or week to week or year to year). Changing select values generate appropriate table and displays below in a table.  Input: Multiple lists involved in the payment of salary at different time periods, List of all employees  Output: No special output  Entities Involved: Employees, one or more EmployeeSalary*Month/WeekName* entities |
|  | Page: A list-view of all current projects under the department of the viewing user. The list-view should be capable of CRUD operations. Display available funds, total funds and assigned funds above the project names. List items should display project name, time period, and budget assigned. Hovering mouse button over the list item displays all employees assigned to the project.  Input: A list of all projects, a list of employees assigned to the projects  Output: No special output  Entities Involved: Employees, ProjectEmployees, Projects |
|  | Page: A select list containing names of all active projects under the department of the viewing user. Selecting a project name display the name of all employees involved in the project with an add button below to add more employees to the project. Pressing add button opens a sub window with a bar for writing employee name and a confirm button. Writing employee name displays the employee info such as name, department, supervisor and projects involved in. Pressing confirm button adds the employee to the project using an asynchronous call and updates the info in main page.  Input: List of all employees, list of all projects, list of all employees involved in the project, list of all projects involved with the employee  Output: No special output  Entities Involved: Employees, EmployeeProjects, ProjectEmployees, Projects |