EPIGNOSIS

PHP developer assignment

Purpose-Disclaimer

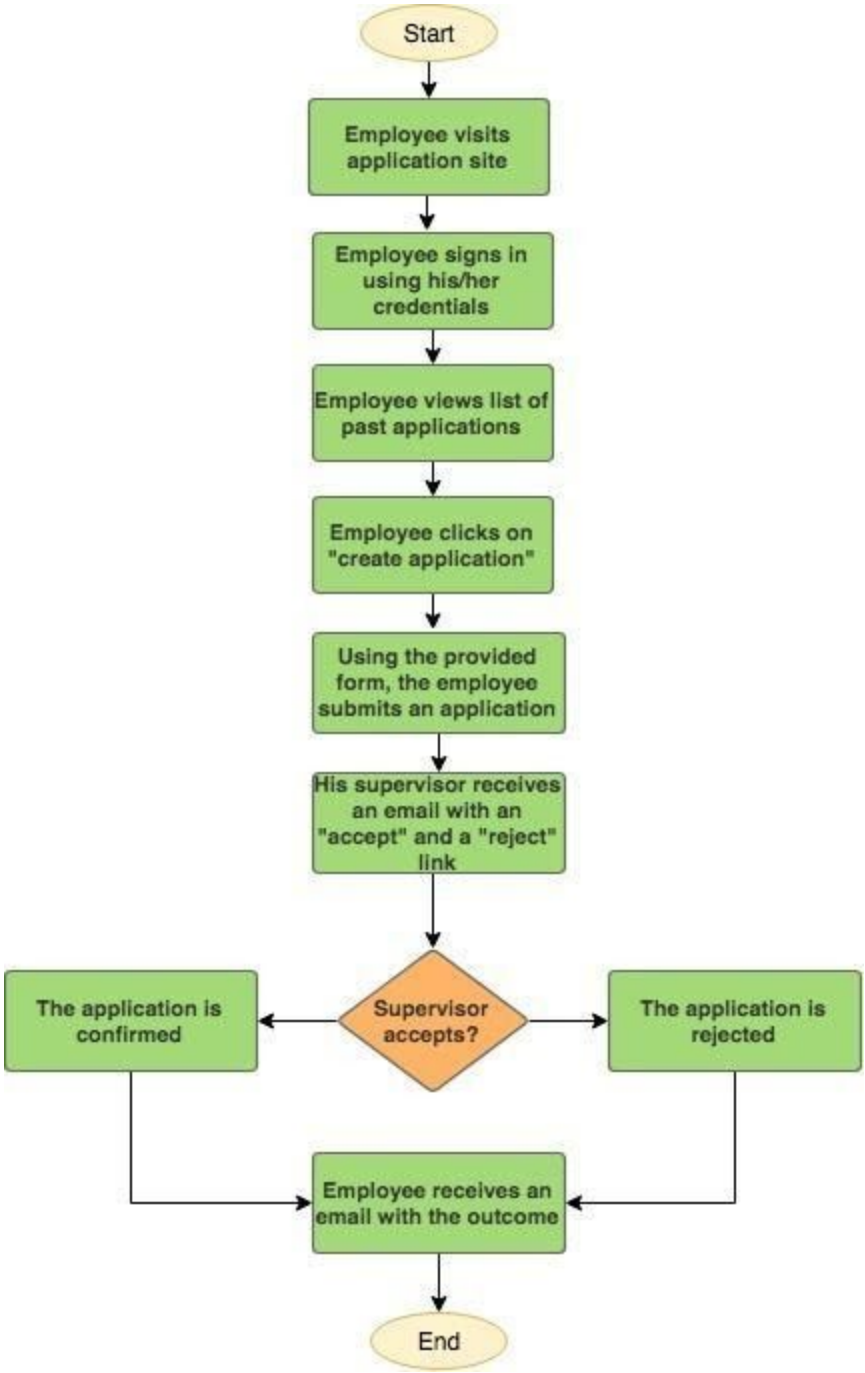
The objective of this assignment is to gauge your technical skills, as well as to give us some talking points for your technical interview. Note that the scope of the assignment is purely fictional. It is in no capacity related to any part of our products and/or business nor will it ever be used commercially or otherwise in any form.

Assignment overview

You work for a company that has tripled in size over the past few years and the way the vacation process works is no longer efficient, as it requires a combination of hand written applications, approvals, storage and maintenance. You are asked to create a portal where employees can request their vacation online, the manager receives a notification to approve or decline that request, and the information (time used, balances) are stored within the portal.

Detailed description

Application process - workflow



Application process - summary

1. The employee signs into the portal
2. A list of past applications is displayed, sorted by submission date (descending) including the following fields:
    a. Date submitted
    b. Dates requested (vacation start - vacation end)
    c. Days requested
    d. Status (pending/approved/rejected)
3. A button “submit request” appears above the list. The employee clicks on the button to visit the submission form
4. The submission form includes the following fields:
    a. Date from (vacation start)
    b. Date to (vacation end)
    c. Reason (textarea)
    d. Submit button
5. After the employee fills-in the fields and clicks on “submit”, he/she is taken back to the list of applications
6. Upon submitting the application, an email is sent to the portal administrator. The email includes the following body:
    “*Dear supervisor, employee {user} requested for some time off, starting on {vacation_start} and ending on {vacation_end}, stating the reason: {reason}*
    Click on one of the below links to approve or reject the application: {approve_link} - {reject_link}”
7. The administrator (who acts as a supervisor as well) clicks on one of the “approve” or “reject” links to mark the application accordingly
8. As soon as the administrator makes a selection, another email goes out to the user notifying him/her of the application outcome, with the following body:
    “*Dear employee, your supervisor has {accepted/rejected} your application submitted on {submission_date}.*”

User provisioning process

The portal includes an administration page where the designated administrator can create and edit users. The process can be summarized as follows:

1. The administrator signs in with his/her credentials
2. He/she views a list of existing users, with the following fields:
    a. User first name
    b. User last name
    c. User email
    d. User type (employee/admin)
3. On top of the page there is a button to create a user. Clicking on it takes the admin to the user creation page, which includes a form with the following fields:
    a. First name
    b. Last name
    c. Email
    d. Password
    e. Confirm password
    f. User type (drop down, admin/employee)
    g. Create button
4. In the list of users, each entry is clickable. Clicking on it takes the admin to the user properties page, which includes the same form as the “creation” page, only this time all fields are pre-populated with the user’s entries (except for the password and the confirm password fields) and the create button is now an update button. The administrator can change the user’s properties by clicking on the update button.

User login process

When an employee visits the portal’s homepage, a “login” form displays, prompting him/her to enter his/her email and password, to sign in.

Technical specifications

1. The portal must be created using PHP 7.
2. The portal must be based on MySQL or MariaDB for the data storage.
3. The portal's source code must be hosted on github.
4. You must not use a PHP framework for the backend, such as Laravel. You are allowed however to use one for the frontend.

Deliverables

1. An EER diagram of the database schema that will be used.
2. Source code for the application.
3. Documentation.
4. A dump of the database.