

ITC Institute - Interview Exercise

Hello!

We would like to evaluate our fit based on a real world simulation of what you will do at the company.

To this end we have designed a simple exercise which will allow us to evaluate your energy, independence, and tech competency.

The exercise requires you to implement a simple full stack app with end-to-end testing. The app will contain two simple screens, or web pages.

1. Login screen
2. Information screen: showing results for information retrieved from the database.

We will provide you with a pre-populated database in the form of a docker image.

Once these screens are implemented, you will need to implement an end-to-end test. Note: end-to-end (E2E) tests are separate from in-code unit testing. E2E tests are meant to mimic a user using the browser to interact with a system that resembles (as closely as possible) the real system in production. The idea is to validate that all parts of the system integrate well together.

The exercise has three “levels” in increasing technical level and complexity. You are expected to at least complete the first level, while the next two levels are meant to measure ability and rate of progress.

Technology

The app will be developed using the following Full Stack and testing technologies:

Front End: React, JavaScript

Back End: Node.js, [sequelize.js](#) as an ORM

Database: [Postgres](#)

Testing: [playwright.dev](#)



Setup

[Install Docker.](#)

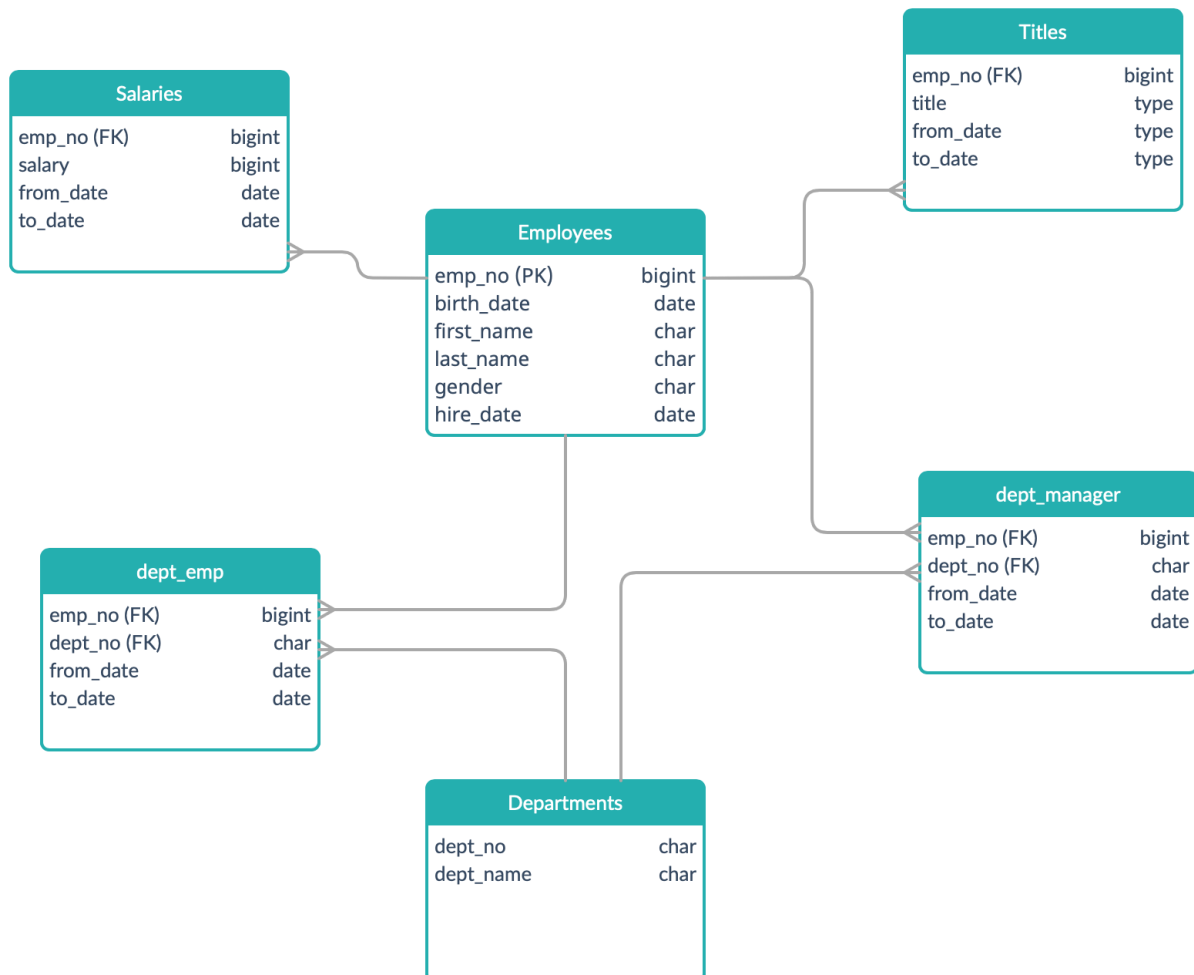
Downloaded and run the database docker image using the following commands:

```
docker run --rm -p 5433:5432 -e POSTGRES_PASSWORD=1234 -d diegmonti/test_db
```

```
psql -U postgres -d postgres -h localhost -p 5433  
(password is 1234)
```

The database contains information about an organization, with employees, titles, departments, management structure, and salary information.

The database implements to the following design diagram

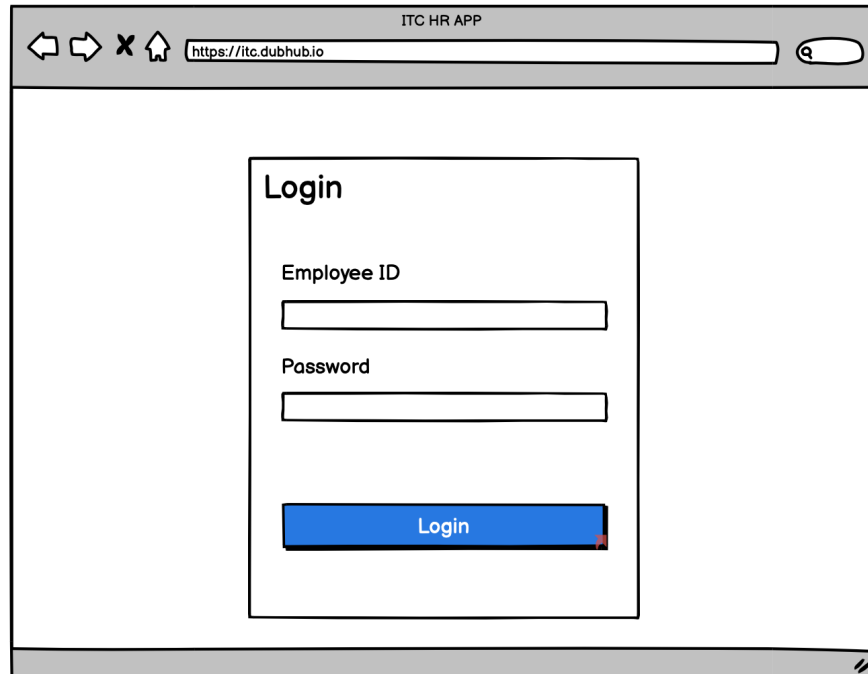


App Requirements

For the first stage, the app is just a login page and an information page.

Login Screen

The login screen requires a user to input their employee id (emp_no) as a user name and their birthday as a password.



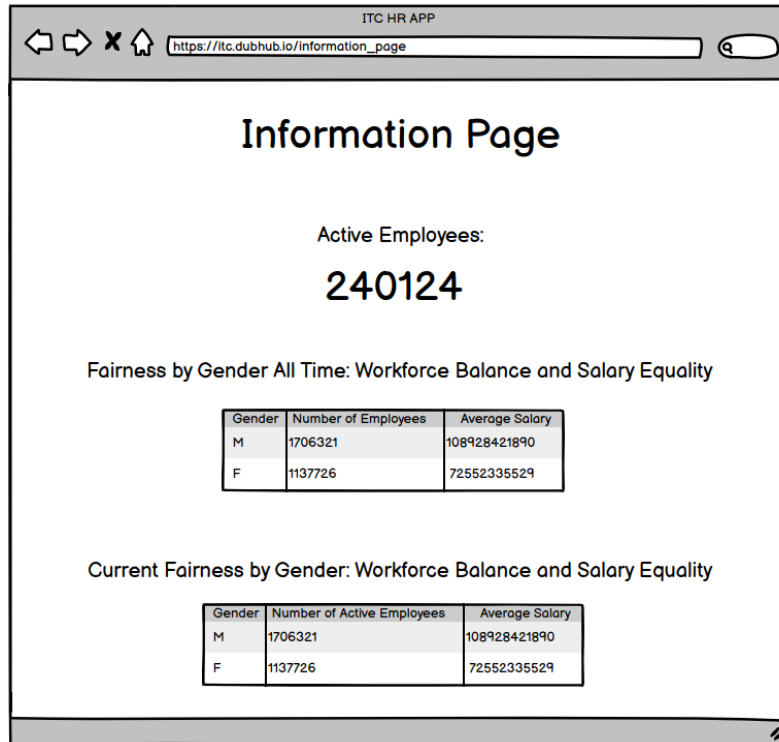
The image shows a web browser window titled "ITC HR APP". The address bar contains "https://itc.dubhub.io". The main content area displays a "Login" form. The form has two input fields: "Employee ID" and "Password". Below these fields is a blue "Login" button. The browser window also shows standard navigation icons (back, forward, home, search) and a search bar.

Information Page

The information page is shown after login. It will contain information about the following topics, and answer the following management questions:

1. Active Employees : What is the total number of employees currently employed?
2. Equality (History) : What is the breakdown of average salary by gender, for all employees (including those that are not currently employed)?
3. Equality (Currently) : What is the breakdown of average salary per gender, for all employees (currently employed)?

Below is a mockup of the information page.



Playwright

Use playwright.dev to test all HTML elements found on the login screen and mock a passed and failed login.

Solution Levels

Level 1 - Basic Implementation - Minimum

A basic solution is expected to have the following implemented:

- Login Screen
- Information Page (initial version, as above)
- Playwright tests
 - Login pass & fail cases
 - Correct data in the information page

We will be looking at the quality of your code on all levels; from how you name variables, to how you compartmentalize components, and most importantly, how thorough your playwright tests will be.

Level 2 - Payroll Forecast - Bonus

Report a list of departments and the total amount of salary they need to pay in the next month + applicable playwright tests.

Playwright tests should validate that the data in the payroll forecast table matches the result of directly querying the database. If data changes in the database and it is correctly reflected in the web page, the test should not fail.

See below screen mockup for an example

ITC HR APP

https://itc.dubhub.io/information_page

Information Page

Active Employees:

240124

Fairness by Gender All Time: Workforce Balance and Salary Equality

Gender	Number of Employees	Average Salary
M	1706321	108928421890
F	1137726	72552335529

Current Fairness by Gender: Workforce Balance and Salary Equality

Gender	Number of Active Employees	Average Salary
M	1706321	108928421890
F	1137726	72552335529

Payroll Forecast

Department Name	Department Num	Active Employee	Number of Salaries Expe	Upcoming Monthly Payr	Yearly Payrol
Marketing	d001	14842	14842	99019452.833333333333	118823343
Finance	d002	12437	12437	81420828.000000000000	97704993
Human Resource	d003	12898	12898	68705388.666666666666	8244646
Production	d004	53304	53304	301359947.41666667	361631936
Development	d005	61386	61386	346104087.50000000	41532490
Quality Managem	d006	14546	14546	79326603.000000000000	95191923
Sales	d007	37701	37701	279153816.83333333	33498458
Research	d008	15441	15441	87387535.250000000000	10486504
Customer Service	d009	17569	17569	98511184.083333333333	118213420

Level 3 - Extra Bonus - E2E Performance Benchmark Reporting

All playwright tests should run multiple times. The exact number of runs should be configurable, but start with 10.

Generate a table with a row for each test reporting the following information per test:

- Number of successful runs
- Average + standard deviation time of all runs
- Time of first run
- Time of worst run, and which run it was

