BGC WorkLoad Management System

Requirements and Specification Document
07/09/2021, version 1
Iteration 1: 🟦
Iteration 2: 🟥
Iteration 3: 🟩

Project Abstract - BGC Workload Management System

The goal of this project is to develop a web application that manages the planned and actual work capacity of BGC Engineering's various Software Development teams, and compare them to the planned workload. Through the information and insights provided through our application, the user will be able to make better decisions regarding the allocation of resources within the company. On our application, users will have access to a headcount graph which will show both a projected and employed headcount for BGC to see how they are doing in terms of hiring. There will also be a working capacity graph which will show work capacity vs workload to determine BGC's hiring needs. Users will be able to create, edit, and delete both project and employee information to adjust the graphs as needed. We are seeking to improve upon the previous implementation of a simple excel document by creating a more organized, intuitive, and fully-featured application.

Customer

The customer is a Software Development Manager within the company BGC Engineering. Other members of the management team will also have access to the application.

Competitive Analysis

Our project will be competing with BGC's previous implementation: an excel document. This document was difficult to navigate and often glitchy, so our project will need to be an improvement over this current document. We will improve the useability of the application by automating many tasks such as creating the ramp-up time automatically rather than having the user manually input each time period. Furthermore, the utilization of a database will allow for more convenient and intuitive CRUD of information, as everything will be organized within form inputs rather than having to edit each cell of an excel document. Lastly, we plan to vastly improve the data visualization by using various visual graph representations of the data, which the predecessor failed to render consistently. Redundant elements of the excel document will be removed for usability (e.g. team, resource, custom, and working days columns).

Scope and Epics

Our project features can be categorized into several main epics:

- Employee database and view

    - Data Structure and Data Table for Employees
    - Various graphs for data visualization
    - Ability to CRUD from front-end interface given proper credential
- Login
    - Custom login with no API; Created separate data table in database
- Project workload database and view
    - Data Structure and Data Table for Projects
- Work Capacity vs Work comparison view
- Proper Timetable/Calendar view (upcoming projects, employee start dates, etc.)
    - Ability to assign employees to projects
- User Interface

User Interface Requirements

The application UI layout can be found here:
https://276group.invisionapp.com/freehand/BGC-WorkLoad-Management-System-PExE8GEcs

Pages

User Login

Managers and read-only users will both use the same login page to access the application. User authentication levels are stored in the user database. Those who are unregistered will not be able to access the application.

Headcount Graph

Displays employee headcount over a changeable period of time.

Employee Entry Form

Allows users to add employee information. Upon saving, the user will return to 'Employees' page.

Current Employees

Displays current employees with a column visualization at the bottom of the screen. Users can add or edit employee information, or return to the 'Headcount Graph'.

Work Capacity vs Workload Graph

Displays work capacity vs workload graph, will also be able to view and add employees and projects.

Project List and Timetable

Will be able to search for current projects, as well as edit and add new projects. Will also display a list of projects and a time table.

Project Entry Form

Allows users to add a project with various attributes such as name, role, team, resource, funding, number of people, etc.

1st Iteration Requirements

For the first iteration, we are looking to implement a restricted user login built from scratch, and create a working database with PostgreSQL for the employee headcount. The application will be initialized using the Spring framework and Maven as the project manager. A barebones front-end with database connection will be the goal. This will include proper display of the employee list and all related information in an intuitive and easy to read format, proper form inputs and backend connection to CRUD individual employees, and lastly a bonus feature of added data visualization using Chart.js or another JavaScript graphing library.

Beyond Iteration 1

As it stands, BGC's current implementation of the Workload Management System is a "large and hard to maintain" excel spreadsheet, as described by the client. We are hoping to improve vastly over the predecessor by implementing a clean, concise, but detailed application to meet the user's needs of viewing their work capacity in comparison to work load. Thus, the first core feature to be implemented will be the Employee database, where we will keep track of active vs planned hires, their work schedule, work capacity, and their assignments to the current projects. If feasible, additional context such as team, official role/title,and other information may be added. The second core feature to be implemented will be the project workload database, where the active/future projects will be listed. The workload, deadlines, scheduled work sprints, ramp up/work capacity are prioritized, but more information such as funding, unit cost, etc may be included upon the client's request and time given.

User Stories

There are two defined user types for this project:

1) BGC managers
2) Regular BGC employees

As such, there is an admin view for managers, and a read-only view for regular employees. Both admin and read-only views share general stories, while unregistered users will be unable to access the application in any capacity other than the login page.

GENERAL STORIES FOR MANAGERS AND REGULAR EMPLOYEES

Changing Graph Dates
On the employee dashboard, both managers and regular employees will be able see a headcount graph of BGC employees. The start and end dates of this graph can be changed by the user.

In testing, we found that when no specific dates were chosen, the graph would be empty. We plan to incorporate some input validation to prevent this failure case.

Filtering Employees
Users will be able to sort employee profiles by filtering attributes like name, employee type, team, hiring status, capacity, and starting or ending dates. They will also be able to find specific employees through a general search function.

In testing, we found that the search was case sensitive. For example, if someone was on team “Avengers” searching “avengers” would not return the desired result. This will be rectified in the next iteration.

ADMIN VIEW STORIES FOR MANAGERS

Adding, Editing and Deleting Employees
Managers will be able to create a new employee from the dashboard. When creating an employee, they will be redirected to a form where they will enter information for their name, position, role/title, status, starting date, and ending date. If they give an employee the “Permanent Hire” position in the form, the form will automatically set the end date to an indefinite period (for now it is set to “2999-12-31”).

Managers will also be able to edit any and all of the existing employee attributes mentioned above. They will be able to delete specific employees directly from the dashboard by selecting the delete button next to their name on the list.

In testing, we found that the lack of form validation allowed users to create employees in the database with missing fields, which may interfere with other processes in the application that rely on certain attributes. For the first iteration, we added a placeholder message asking the user to fill in the required field. This will soon be replaced with proper form validation.

Giving User Access
Managers will have the authority to create other users and define their access levels; “edit” (“admin” view equivalent to manager) or “read-only”. When they click “Add New Manager” from the navigation bar, they will be redirected to a page where they will be able to enter a username and password, and also select the access level/user type. Manager user types will have the same privileges and access rights as listed above, whereas read-only users will have more restricted rights and privileges that will be outlined below.

UNREGISTERED USER STORIES

Login
The login page asks for a username and password which either permits admin or read-only view. With the exception of the login page, an unregistered user will be unable to access any of our project's pages and be directed to a page that says "this action is unauthorized".

DEVELOPER STORIES

Delete Data

Currently, there is an option for a user with admin view to delete all data. This is a temporary feature included by the developers for testing clarity, and will be eliminated in later iterations.

Distribution of Work

Early on, the project was divided into features which were assigned to pairs or individuals. Over the duration of the sprint we gradually adopted an Agile approach to our process as we began to collaborate more and ensure our different features were integrated cohesively. We also employed pair programming the most frequently, where one group member focused on reviewing code and providing information on methods and data while the other implemented and pushed code onto github. Weekly meetings with the employer ensured customer participation, allowing us to take feedback into consideration and elaborate on requirements as we developed the features. All members of the group were actively participating in meetings both with and without the employer. As such, the distribution of work was fairly balanced for this first iteration, with our members proactively engaging in not only development activities but also communication and organization of the project development process.

Documentation

Documentation and the outline of the application design and implementation are clearly and explicitly detailed in this document. However, since we developed this iteration in a more Agile fashion, we decided to forego explicit documentation of the codebase with the exception of detailing the data classes we used.

Timeline

In the first week of the iteration, we quickly assigned tasks to individual members and pairs within the group. Our initial plan was to implement a login with the Microsoft Azure Directory API, and have a working database configuration through PostgreSQL for the employees. However, after initial struggles and reconsideration of user requirements, we opted to omit the Azure API and create our own user database. The rest of the features planned were all successfully implemented according to our planned timeline.