|  |
| --- |
|  |
| Report 1 Part 2 Team 16 |
| System design and implementation strategy |

|  |
| --- |
|  |

System design - Product breakdown structure diagram of the different components

A diagram of a company

Description automatically generated

The diagram follows on that each of the subtasks should be an individual page on the website.

State machine diagram to model the system’s behaviour.

A screenshot of a computer

Description automatically generated

**State machine overview**

Once a user has logged in with an existing account, they will be taken to one of the 3 different types of users being employees, manager and team leader. The diagram illustrates the different actions each user can perform. The once the user is finished with the system, they would sign out via the button on the settings page.

**Implementation strategy**

Evolution from design

As the client already liked the overall look and navigation of the prototype, the team decided that it would be a good idea to evolve the prototype into the eventual delivered product (feature of an evolutionary model). This saves on development time as the components are being reused in the final implementation and ensures that the aesthetics of the website don’t change too much ensuring validation from the user.

After breaking down the problem into individual and easier to manage problems using the product breakdown diagram, it is now easier to develop an implementation strategy. With multiple people working on the project but on different aspects, it suffices that mostly using a waterfall approach for the reasons detailed below.

Suitability of the waterfall approach

The waterfall model is focused on activities. This works well as the product breakdown structure above easily splits up each section of the project into easier to manage and simpler sub tasks. Each team member can now work on a different sub task to increase the productivity of each member and thus the whole team. “Typically, the system is coded in smaller components, or units, before being put together” (Lutkevich, 2024).

The project has well defined requirements so that the requirements don’t need to be revisited and the client doesn’t have to be too involved which further speeds up the development process as time is saved from not interacting with the client. Furthermore, requirements aren’t also added throughout the development process which means time is further saved from modifying the requirements via validation from the original specification (apart from the added requirements from the demonstration).

Use of GitHub

As each simpler subtask can be worked on by a different team member it is sufficient to use GitHub to host the repository of the team to allow collaboration and version control of the system. We have used issue tracking to notify other members of the team about issues we have faced that may halt progression to enable good communication within the team.

“Typically, you might create a new branch from the default branch of your repository. You can then work on this new branch in isolation from changes that other people are making to the repository” (Github, Inc, 2024). My team will ensure each team member creates a branch to implement changes so they can be reviewed before going in the master branch so that the master branch remains error free and won’t stop the rest of the team from working if there’s an error.

Use of meetings

Medium sized projects like the make-it-it all system tend well towards waterfall as low levels of complexity and organisational skills are needed. Our team of 6 members had weekly meetings to discuss how the project was moving forward communicating on difficulties and challenges we faced and as a result required no manager to oversee progress of each member. For every meeting we made a meeting summary so that the points that were talked about weren’t forgotten about. This uses ideas from a Scrum methodology where the team is self-organizing, and the team has daily meetings (in our case daily meetings were too little a time gap as little progress had been made each day) to discuss progress and setbacks contributing to continuous improvement. “…The main aim of these meetings is to briefly discuss task statuses and any hindrances and ensure the goals of the scrum team”. (MeetingNotes.com, 2024)

Progressing effectively

Due to its sequential nature, Waterfall facilitates easy progress tracking and provides a clear indication if the project is running on time. I have made an activity on Node diagram to help with progress indication which will be discussed in weekly meetings that is based upon the MoSCoW method and the activity on node diagram.

Waterfall minimizes the risk of scope creep and gold plating, promoting team focus. The MoSCoW method is used to clarify tasks and their prioritized order for the entire team.

Big bang approach

The website is being deployed using a ‘big bang’ approach so all the features will be deployed at the end and each webpage will work together as required by the specification. A system that that is deployed in increments would take longer to develop since each increment would need to be tested upon release, slowing down development time. Big bang requires thorough testing at the end of the project, which is crucial when many team members are working partially independently and not fully testing how different elements of the website work together.

Planning and prioritising tasks: Using the MoSCoW method.

“Central to MoSCoW is its acronym, denoting four priority categories – Must have, should have, could have, and won’t have. This classification is crucial in managing stakeholder expectations, directing the project team’s focus towards critical elements, and charting a clear course for project advancement.” (Learn6Sigma, 2024)

As a result, the MoSCoW method would be a sufficient method to prioritise tasks. The only feature that is not needed from the MoSCoW method is the ‘won’t’ requirements due to the project being a small size, completing all tasks is achievable.

**Must**

* **Some projects are confidential, only users that are part of the confidential project can access the posts related to the project (requested in demonstration)**

It would be a security risk if other users not part of the team could access confidential posts.

**All requirements listed in Report 1 Part 1 have ‘Must’ priority due to being directly being sourced from the specification letter and from the clients themselves (requirements also specified in the handout document in the presentation).**

**Should**

* **Change ‘add task’ name to ‘assign task’ (requested in demonstration)**

Would reduce confusion between the difference between as add task is used directly on an employee and is the same as assign task but auto fills the assignee.

* **Edit and delete tasks (improves ease of use)**

Would make tasks that were mistyped easy to change and update or if the task description is unclear can be edited and clarified.

* **New password (improves security of website)**

To ensure sufficient security measures creating a new password for an existing user if their password gets compromised. EG: by social engineering.

* **Method to prevent SQL injection (improves system’s robustness)**

To ensure security measures; stopping a user that has logged in to modify the database illegally.

* **Edit and delete posts (makes system more organised)**

Would make tasks that were mistyped easy to change and update or if the task description is unclear can be edited and clarified. This action can be achieved by Managers and the original post creator.

* **Delete replies (makes system more organised)**

This is used to delete replies that are no longer needed, outdated, or display incorrect information. Managers and the original reply creator can delete the reply.

**Could**

* **Light/dark mode (requested in demonstration)**

A light/dark mode function would improve the aesthetics of the website as requested by the client in the demonstration.

* **Searching and sorting view topics and view posts asynchronously (Would make the website more responsive)**

Would allow a user to easily navigate to which topic and post they want to view without needing to refresh the page.

* **Searching and sorting by in manage employees asynchronously (Would make the website more responsive)**

Would allow a manager to easily navigate to which employee they want to view via searching for their name or by filters such as what user level (manager, team leader or employee) they are.

**Activity on node diagram to show how ‘Must’ tasks are planned and prioritised.**

“Activity on Node (AON) diagrams are a powerful tool for visualizing project schedules and workflow. They provide a clear overview of the activities involved in a project, as well as their dependencies and the critical path.” (kdi-ppi, 2024)

With the project broken down via the product breakdown structure diagram and the tasks prioritised using the MoSCoW method it is now appropriate to start planning the order in which tasks are completed.

A diagram of a company

Description automatically generatedThe approximate critical path for the diagram below is as follows: Database -> Login -> Milestone 1 -> Manager dashboard -> Manage employees -> Manage employees.

**Explanation following the Activity on node diagram.**

(NOTE: it is difficult to give the duration of each task as the team has never completed an endeavour like this, and the task completion time can vary depending on team availability how many bugs there are. Without the duration, having early and late start times is impossible hence why they are omitted in the diagram).

The first task is the **database,** creating the tables which supplies the project with necessary tables containing data that is used in the subsequent tasks.

Once the database has been made the **login** can be created (needs to access the database as a prerequisite to access valid login information) so that the 3 different types of users (Manager, Team Leader, and employee) can be accessed.

Once the login functionality works, **new users can then be created** and added to the database. Along with this, **authentication** functions need to be made to ensure the data is valid and won’t cause bugs and errors to the database when they are needed. This happens when the user needs to sign into the account they have just created.

I found it appropriate to place a milestone after the whole of the database and login section had been complete as this marks a monumental point of completion in the project.

Once the first milestone had been complete, multiple pages can be worked on simultaneously. This includes the **dashboards** (for the Manager, Team Leader, and Employee) and the **settings** page. These pages depend on who signs in, which dictates which dashboard to show. Furthermore, once the login is complete, the knowledge organisation sub system can be developed on, starting with **view topics**. This page can have pre-coded topics inserted directly into the database while the create functionality is not available.

Once all the topics can be viewed, it follows that **posts in topics can be viewed**. After this, users should be able to create posts (which can additionally create topics) to be displayed on the viewing pages.

After the Team Leader and Manager dashboards have been made, an **assign task page** can be created as it is used in both dashboards thus it needs to work and be tested in both places.

The addition of the Manager dashboard allows the **manage employees** and **create project** pages to be made; they are dependent on the dashboard being made first. The pages can be made in parallel as their functionalities are exclusive of one another.

The **settings** page is just dependent on the login being created. The only essential feature here is sign out. This also includes features such as changing password and light/dark mode that aren’t essential to how the system operates.

# References

Github, Inc, 2024. *About Branches.* [Online]   
Available at: https://docs.github.com/en/pull-requests/collaborating-with-pull-requests/proposing-changes-to-your-work-with-pull-requests/about-branches  
[Accessed 17 02 2024].

kdi-ppi, 2024. *Creating Activity on Node Diagrams Made Easy.* [Online]   
Available at: https://kdi-ppi.com/article/activity-on-node-diagram-generator/  
[Accessed 18 02 2024].

Learn6Sigma, 2024. *Guide: MoSCoW Method.* [Online]   
Available at: https://www.learnleansigma.com/guides/moscow-method/  
[Accessed 17 02 2024].

Lutkevich, B., 2024. *waterfall model.* [Online]   
Available at: https://www.techtarget.com/searchsoftwarequality/definition/waterfall-model  
[Accessed 17 02 2024].

MeetingNotes.com, 2024. *What Are Scrum Meetings? The 5 Types of Scrum Meetings.* [Online]   
Available at: https://meetingnotes.com/blog/what-are-scrum-meetings  
[Accessed 17 02 2024].