Week 2.2: Software Processes

General Information

Week: 2

**Day**: Friday

# Task 1: Formative Feedback on Project Scope (45 minutes)

**Goal:** Finalise project scope and get formative feedback

## Instructions:

* In a shared document work as a team to finalise your project scope
* Make sure to discuss the scope with a tutor to get formative feedback

Scope:

An inventory management system for a shop to track, add, update stock, generate reports and invoices across multiple branches.

Requirement Gathering & Analysis:

* Functional requirements
* System should allow users to log in
* System users to log out
* System should allow user to view what is currently in stock
* System should allow users to edit stock count + info
* System should allow users to add and remove stock
* System should allow users to search for specific stock using a range of different filters or directly.
* System should allow users to inspect stock for more information.
* System should track amount of sales made in a day
* System should be able to generate a report on sales made over a period of
* System should format invoices
* System should allow admin user to add and remove users

Non-functional Requirements:

* Design supports both PC and mobile
* Follows all LSEPI protocols
* Fast response time
* Should be able to support users across multiple branches
* Accessible

Design:

Architectural design:

The system will follow a three tier design with a front end that supports web and mobile and a PHP backend

Database Design:

* We will use a relational database (phpMyadmin) to store games, employees, transactions.

Interface Design:

* Simple and responsive design that is easy to use for employees

Component:

-Stock management

- Generating invoices

- create reports

Implementation:

The system will be developed using HTML and PHP for front end and styled with CSS/tailwind CSS. The backend will be created solely in PHP.

Development will occur 8 weeks of bulk development and an extra 2 weeks of testing and editing features.

Testing:

Unit testing : testing individuals units a features of the programme.

Integration Testing: Ensure components interact with each other properly.

# Task 2: Initial Project Plan (30 minutes)

**Goal:** Work as a team, set an initial project plan for the assessment in consistency with your scope.

## Instructions:

Create a plan showing the main milestones in your project. This should be consistent with the project scope and development process (Scrum).

You may use any tool to document your plan, including:

* Trello (or similar tools)
* Gantt Chart
* Excel

For Excel, you may use the following template:

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Phase | Task ID | Task | Allocated To | Predecessor Task | Planned Start Date | Planned End Date | Actual Start Date | Actual End Date | Status |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

**Discuss the work plan with tutors.**

# Task 3: Git Branching

Follow this guide for branching using Git and GitHub for editing an HTML file and README file. We'll follow the structure you requested, where:

* **Main branch** holds the final, approved code.
* **Development branch** is used for ongoing work.
* **Feature-based branches** for each task (like adding overview or team members’ details).

**Prerequisites:**

* Ensure Git is installed on your local machine.
* You have a GitHub account, and a repository is created.
* Sign into GitHub using visual studio code (optional) or any IDE you prefer.
* Watch one of the following videos about branches:

<https://www.linkedin.com/learning/learning-git-and-github-23011330/branches>

<https://www.youtube.com/watch?v=QV0kVNvkMxc&ab_channel=NetNinja>

**Set up the GitHub repository and Main branch**

1. **Create a new repository on GitHub.**
   1. **Name it something like team-profile-project**
   2. **Initialize the repository with a README.md file**
   3. **Clone the repository to your local machine:**

git clone https://github.com/your-username/team-profile-project.git

cd team-profile-project

1. **Create and set up the Development branch**
   1. Create a development branch for working on changes:
   2. Push the development branch to GitHub:

git checkout -b development

git push origin development

**Now, your repository has a main branch (default) and a development branch (working branch).**

1. **Work with Feature Branches (Example: Add Overview Section to README)**

a)Create a feature branch for the "Overview" section

git checkout -b feature/overview

* 1. Open the README.md file in your preferred editor.
  2. Add the following:

## Overview

This project is about building a simple 'About Us' page for a team of 4 members using HTML. Each member will add their details to the page.

1. **Add and commit the changes:**

git add README.md

git commit -m "Added Overview section to README"

1. **Push the feature branch to GitHub:**

git push origin feature/overview\

1. **Create a Pull Request (PR) to Merge into Development**
   1. Go to your GitHub repository and click on the "Compare & pull request" button for the feature/add-overview-to-readme branch.
   2. Set the base branch as development and compare it with your feature branch feature/overview.
   3. Add a description and create the PR.
   4. Once approved (you can approve it yourself or have a teammate approve it), click "Merge pull request."
   5. Delete the feature/overview branch on GitHub after merging.
   6. Update your local development branch

git checkout development  
git pull origin development

1. Add Team Members to the About Us HTML Page
   1. Create the about-us.html file in the development branch
   2. In your editor, edit the file about-us.html with the following basic structure:

<!DOCTYPE html>

<html>

<head>

    <title>About Us</title>

</head>

<body>

    <h1>Meet the Team</h1>

    <div id="team">

        <!-- Team member details will go here -->

    </div>

</body>

</html>

Use GitHub Issues to assign each member a task to add their details to the HTML file.

If you are not familiar with GitHub Issues, This Tutorial can be Helpful:

<https://www.youtube.com/watch?v=TKJ4RdhyB5Y&ab_channel=TheCodex>

**Commit the initial structure:**

git add about-us.html

git commit -m "Created basic structure of about-us page"

**git push origin development**

**Add Team Member 1 (Example: Add John’s Details)**

git checkout -b feature/add-john-details

**Edit about-us.html by adding the following inside the <div id="team"> section:**

<div class="member">

    <h2>John Doe</h2>

    <p>Role: Frontend Developer</p>

    <p>Bio: John specializes in designing responsive interfaces and enjoys creating smooth user experiences.</p>

</div>

**Commit and push the changes:**

git add about-us.html

git commit -m "Added John’s details"

git push origin feature/add-john-details

**Create a Pull Request (PR) to merge into development:**

* **On GitHub, create a pull request for the feature/add-john-details branch.**
* **Merge it into development.**

**Update your local development branch:**

git checkout development

git pull origin development

**Repeat for Other Team Members**

**Merge Development Branch into Main**

**Once all features (overview in README.md and team member details in about-us.html) are completed and merged into the development branch:**

* **Switch to main branch**

git checkout main

* **Merge development into main:**

git merge development

* **Push the updated main branch to GitHub:**

git push origin main

**Ask your tutor to verify that Your project has both the README.md with an "Overview" section and an about-us.html file containing team member details.**

# Task 4: Learning Git and GitHub - LinkedIn Learning Certificate (Optional)

On your own pace, complete the following course on linked-in learning. You will need to complete all course content to earn your Certificate

<https://www.linkedin.com/learning/learning-git-and-github-23011330/understanding-version-control>