

**DevOps Fundamentals [Fall, 2024]**

# **OBJECTIVES**

The objective of this assignment is to provide hands-on experience in collaborative development using Git and GitHub. Students will learn how to effectively use Git in a team setting, including working with remote repositories, handling branches, managing pull requests, and resolving merge conflicts. By the end of this assignment, students will have practiced essential skills needed for working in a DevOps team.

# **IMPORTANT INSTRUCTIONS**

1. This is a group assignment and students will work in a **group of five (05)**.
2. **Read** the provided **scenario** (on page. 02) before attempting the task.
3. Each student **must** **configure** current repository **User Name with your own name** and **the email** with **University Email Address.**

**IMPORTANT: User name and email MUST BE IN ALL SMALL CASE.**

1. **One student** from each group will act as **team lead**. While the **other** will act as **team members.**
2. The **team lead** will **fork** the **instructor’s repository** from this **URL: https://github.com/imranucp/estore.git**
3. Once the work is complete, **only** the **lead student must upload** this document after **filling in** the details in **table:01**.

Table

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Group Repository URL: [Paste Group Repository URL Here]** | | | | |
| **Group Member Details** | | | | |
| **Sr #** | **Name** | **Roll Number** | **Role** | **Branch worked On** |
| **1** |  |  | Team Lead |  |
| **2** |  |  | Team Member |  |
| **3** |  |  | Team Member |  |
| **4** |  |  | Team Member |  |
| **5** |  |  | Team Member |  |

**Best Of Luck**

**Scenario**

Students will work in groups of 5 to develop a simple static website for an online store front selling mobile phones. The website will consist of **multiple pages** including a **Home page**, **Products page**, **Product Details page**, **Contact Us page**, and **Promotions page**. Each group will collaboratively work on the project by implementing features, styling the website with a **single CSS file**, and managing code through Git and GitHub. The **website** will have only **one css file** for styling all of the HTML pages.

**Tasks**

1. **Fork the Repository:**  **[20 Marks]**
   1. The **lead student** should **fork** the **instructor's repository**.
   2. Make sure default branch is “develop”.
   3. **Add** the remaining group members as **collaborators** to the forked repository.
   4. **Add** the **instructor** as a **collaborator** for oversight and feedback. The email address of the instructor is: **zsaing.ucp@gmail.com**
   5. The **lead student** will **create** the **branch protection rule/s** so that no feature branch can be merged without a PULL Request and no developer must be able to push the **local develop** branch to update the **remote develop** branch.
2. **Assign Tasks: [10 Marks]**
   1. The **lead student** will **create issues** in the repository, clearly outlining the tasks for each group member. Each issue must mention the **webpage** and the **corresponding** **css file.**
   2. **Assign** one **issue** to the **each member**.
3. **Clone and Create a Branch: [10 Marks]**
   1. **Each** group **member** should **clone** the repository.
   2. **Create** a **feature branch** from the default branch i.e “develop” branch. Name the feature branch after the **assigned issue** e.g **feature/issue#11**
4. **Implement Features: [15 Marks]**
   1. **Work** on your assigned feature, by **creating respective HTML page.**
   2. Every team member Modify the **shared CSS** file for all sorts of styling.

**NOTE:** It will create the possibility of having merge conflicts.

1. **Commit and Push: [10 Marks]**
   1. **Commit** your **changes** regularly, using descriptive commit messages.
   2. Once feature implementation is complete then **push** your **feature branch** to your **remote repository**.
2. **Create a Pull Request + Code Review: [20 Marks]**
   1. **Open/create** a **pull request** to merge feature branch to the **development branch**.
   2. The **developer** (who is creating the pull request) must **assign the rest of three developers as reviewers** and **the team lead** (who is repository owner) must be assigned as “assignee” to each pull request.
   3. The reviewers must carefully examine the code, ensuring it meets the project's requirements and coding standards.
   4. Provide feedback and suggestions for improvements (if required). The developer will work to fix the concerns (if raised by the reviewers).
   5. **After** successful PR review **Only** repo owner i.e **team lead** will be **authorized** to merge the feature branch into the development branch. **At least one approving review comment** will be required to merge the feature branch.
   6. Each developer **must “pull”** the latest changes in the remote develop branch into the their local develop branch. They must do it **after each successful PR** on the develop branch. It will keep local develop branch updated.
3. **Resolve Merge Conflicts: [5 Marks]**
   1. The **collaborator** whose branch will cause the **conflict** will **resolve** it. And update the PR accordingly.
4. **Create Release branch [10 Marks]**
   1. Once **work** for the current **spring** is **complete** then the **lead student** will create **“release”** branch from **“develop”** branch.
   2. Once the work on the release branch is carefully tested and debugged. The **lead student will create the PR to merge the “release” branch into “main” branch** which is the production branch. **The** **rest of the team** members will be **added** as **reviewers to this PR**. Finally, only **lead student** will **merge** this **release branch** into **main branch**. All the reviewers must give the approving comments in order to merge the release branch into production branch. Otherwise merge wont take place. So the required approvals will be 4.