# Création d’un site Web en utilisant Git

Le but de ce laboratoire est de casser la glace avec les systèmes de gestion de versions et la collaboration entre plusieurs personnes qui codent sur le même projet.

Ce laboratoire se fait en équipe de 2 ou 3 et simule une équipe de travail sur un nouveau projet.

### Le projet

Le projet consiste à faire un nouveau site Web pour votre compagnie/équipe. Il doit contenir trois pages Web : index.html, services.html et contact.html.

Ces pages doivent être liées à une feuille de styles CSS.

Un fichier README.txt doit exister.

Le nom de votre équipe est au choix.

La page d'index

La page d'index contient :

1. Un logo
2. Le menu
3. Un texte descriptif de la compagnie/équipe

La page services

Cette page contient la liste des services offerts par la compagnie/équipe

La page contact

La page de contact donne l'adresse et le numéro de téléphone de la compagnie.

### Première étape : Création d’un dépôt (repository)

Sur github, **l’un** des membres de l’équipe créer le dépôt : C32-nomEquipe.git

*Note : Vous pouvez faire des projets Git à la maison sans passer par un service du genre* github*, mais le fait de l’avoir sur un serveur public vous permet d’y avoir accès de n’importe où, pas juste à partir de votre réseau interne. Cela fait également un backup automatique du projet.*

### Deuxième étape : Ajouter les collaborateurs au dépôt

La personne qui a créé le dépôt ajoute les autres membres de l’équipe comme collaborateurs.

### Troisième étape : Cloner le projet

Lorsque tous les membres de l’équipe ont accès au dépôt, tous clone le dépôt afin d’en faire une copie sur son ordinateur.

### Quatrième étape : Ajouter, modifier, soumettre, push/pull

Chaque membre créé un nouveau fichier (ex : index.html, contact.html, etc.) et l’ajoute au projet.

Faire un « commit » local puis un « push » sur github pour que les autres puissent avoir accès à ces modifications (via la commande « pull »).

### Cinquième étape : Aller cherchez les modifications des autres collaborateurs

Chaque membre va chercher ce que les autres ont fait en effectuant un « pull ».

### Sixième étape : Séparation du travail

Séparez-vous le travail entre les membres de l'équipe et programmez le site Web demandé à la page précédente. Après chaque fonctionnalité, faire un commit & push. N’oubliez pas d’intégrer les modifications des autres en effectuant des « pull » régulièrement.

Petit conseil : Pour éviter les conflits de code, essayez de ne pas travailler sur les mêmes fichiers (ou sur les mêmes lignes du même fichier).

### Septième étape : Tenter de résoudre un conflit

À un certain point, essayez de générer un conflit, puis de le résoudre.

1. Les deux étudiants changent une même ligne de code, puis font un commit
2. Un des étudiants fait un push
3. L’autre étudiant fait un pull, ce qui engendra un conflit.

Il s’agit donc de nettoyer le code, de le commit, puis finalement de faire un push.