

# Examen pratique : Git, GitHub, Markdown et gestion de projet

Durée : 2,5 heures

Outils autorisés : Visual Studio Code, Terminal, GitHub

Livrable unique :

Un fichier **Word** nommé :

**ExamenGit\_NomPrenom.docx**

contenant :

- toutes les **captures d'écran** demandées (voir les consignes par section)
- le **lien vers votre dépôt GitHub privé**
- une section expliquant que vous avez **ajouté l'enseignant comme collaborateur**

Aucun dossier projet, aucun fichier .zip, aucun dépôt à envoyer autrement.

Tout doit être remis dans **un seul document Word**.

L'enseignant accédera à votre code via votre **dépôt GitHub privé**.

## Partie 1 – Initialiser et versionner localement

1. Créez un dossier nommé **ExamenGit**.
  2. Initialisez un dépôt Git local.
    - 📸 **Capture 1 : preuve de l'initialisation du dépôt Git**
  3. Créez un fichier **index.html** contenant un `<h1>` avec votre prénom et nom.
  4. Effectuez un premier commit.
    - 📸 **Capture 2 : preuve que le fichier est bien suivi par Git**
  5. Ajoutez un paragraphe `<p>` avec le titre du cours.
  6. Effectuez un deuxième commit.
    - 📸 **Capture 3 : aperçu de l'historique avec les deux commits**
- 

## Partie 2 – Connecter le projet à GitHub

1. Créez un dépôt **privé** nommé **ExamenGit** sur GitHub.
2. Associez votre dépôt local au dépôt distant.
  - 📸 **Capture 4 : preuve que le dépôt distant a été ajouté**
3. Poussez vos commits vers GitHub.
  - 📸 **Capture 5 : aperçu du dépôt en ligne avec les fichiers visibles**

## Partie 3 – Branches et fusions

1. Créez une branche **nouvelle-fonctionnalite**.
2. Dans cette branche, créez un fichier **style.css** puis faites un commit.
3. Envoyez la branche vers GitHub.
  - 📸 **Capture 6 : preuve que la branche a été créée et publiée**
4. Revenez sur **main** et fusionnez la branche localement.
  - 📸 **Capture 7 : aperçu de l'historique montrant la fusion**
5. Poussez vos changements sur GitHub.
6. Créez une branche **ajout-titre**, modifiez le `<h1>` de `index.html`, faites un commit, puis poussez la branche.
7. Créez une **Pull Request** vers **main**.
  - 📸 **Capture 8 : page de création de la Pull Request**
8. Fusionnez la PR sur GitHub.
  - 📸 **Capture 9 : aperçu confirmant la fusion**
9. Mettez à jour votre répertoire local.

## Partie 4 – Conflit de fusion

1. Créez une branche **conflit-test** à partir de **main**.
2. Sur **main**, modifiez le paragraphe de `index.html`, faites un commit et poussez vos changements.
3. Sur **conflit-test**, modifiez le même paragraphe avec un contenu différent puis faites un commit.
4. Tentez une fusion de **conflit-test** dans **main**.
  - 📸 **Capture 10 : affichage du conflit dans l'éditeur**
5. Résolvez le conflit, puis enregistrez la fusion et poussez les changements.
  - 📸 **Capture 11 : historique montrant la résolution**

---

## Partie 5 – Fichier Markdown

1. Créez un fichier **README.md** à la racine du projet.
2. Ajoutez du texte en gras.
3. Poussez ce fichier sur GitHub.
  - 📸 **Capture 12 : aperçu du fichier README.md sur GitHub**

## Partie 6 – Issues GitHub

1. Créez **deux Issues** :
    - une pour une amélioration
    - une pour un bug
  2. Ajoutez des étiquettes (labels).
    - 📸 **Capture 13 : liste des Issues avec leurs étiquettes**
  3. Assignez-vous l'une des Issues et ajoutez un commentaire.
    - 📸 **Capture 14 : preuve de l'assignation et du commentaire**
- 

## Partie 7 – GitHub Pages

1. Activez **GitHub Pages** dans les paramètres du dépôt.
  - Source : branche `main`, dossier racine
2. Attendez que GitHub génère votre site.
3. Ouvrez l'URL du site (affichant votre `index.html` ).
  - 📸 **Capture 15 : page GitHub Pages affichée dans un navigateur**
4. Ajoutez l'URL du site à la fin de votre document Word.