

Git & GitHub

Git היא מערכת מבוזרת לניהול גרסאות (DVCS) שעוזרת למפתחים לנהל שינויים בקוד המקור שלהם לאורך זמן. היא נוצרה על ידי לינוס טורוואלדס, יוצר לינוקס,

בשנת 2005





ניהול גרסאות

Git מאפשר לעקוב אחר שינויים שבוצעו בקבצים לאורך זמן. זה אומר שניתן לצפות בגרסאות קודמות של הקוד, להשוות שינויים ולהחזיר את הקוד לגרסאות קודמות במידת הצורך.

מערכת מבוזרת

בניגוד למערכות ניהול גרסאות מרכזיות Git, היא מבוזרת. לכל מפתח יש עותק מלא של הריפוזיטורי, כולל כל ההיסטוריה שלו. משמעות הדבר היא שגם אם השרת נופל, המפתחים יכולים להמשיך לעבוד באופן עצמאי ולסנכרן את השינויים שלהם מאוחר יותר.

הגדרות ראשונות:

```
git config --global user.name "your_name"          # הגדרת שם משתמש גלובלית
git config --global user.email "your_email@example.com" # הגדרת אימייל גלובלית
git config --global user.name                      # בדיקת שם משתמש נוכחי
git config --global user.email                     # בדיקת אימייל נוכחי
git config --local user.name "your_name"           # שינוי שם משתמש לריפו ספציפי
git config --local user.email "your_email@example.com" # שינוי אימייל לריפו ספציפי
```



ב-Git יש שלושה אזורים עיקריים שבהם קבצים יכולים להימצא במהלך העבודה על פרויקט:

1. **Working Directory** – האזור שבו נמצאים כל הקבצים הפיזיים של הפרויקט כפי שהם

מאוחסנים במחשב. כל שינוי שמבוצע בקובץ כלשהו ולא נשמר עדיין ב-Git נחשב לשינוי

שנמצא ב-working directory.

2. **Staging Area (Index)** – אזור שבו שומרים שינויים לפני ביצוע commit. כאשר קובץ

מתווסף ל-staging area, הוא משמעות היא שהוא מוכן להיכלל ב-commit הבא, אך עדיין לא

נשמר בהיסטוריית ה-commits של Git.

3. **Repository (Local Git History)** – המקום שבו ה-commits נשמרים. לאחר ביצוע

commit, הקובץ מועבר מה-staging area – repository – ונרשם בהיסטוריה של Git.

קובץ שנמצא כאן נחשב לשמור באופן קבוע עד ביצוע שינויים נוספים עליו.



ריפוזיטוריז

Git מאחסן את כל הנתונים של הפרויקט בתוך ריפוזיטורי. זה כולל את ההיסטוריה המלאה של השינויים, הענפים, התגיות ומידע נוסף.

commit

קומיט הוא תמונת מצב של הפרויקט בנקודת זמן מסוימת. הוא מייצג קבוצת שינויים שבוצעו בקבצים בתוך הריפוזיטורי. לכל קומיט יש מזהה ייחודי (hash) והוא מכיל מידע כמו שם המחבר, תאריך והודעת הקומיט.

עבודה עם ריפו (Repository)

```
git init          # חדש Git יצירת ריפוזיטורי
git clone URL     # GitHub-שיבוט (העתקת) ריפו קיים מ
git status        # (commit שינויים, קבצים שלא בוצע להם) בדיקת מצב הריפו
```

עבודה עם commit

```
git add file_name # Staging-הוספת קובץ ספציפי ל
git add .          # Staging-הוספת כל הקבצים ששוננו ל
git commit -m "message" # עם הודעה Commit ביצוע
git commit --amend -m "new message" # האחרון | Commit-שינוי ההודעה של ה
```

עבודה עם לוג לראות היסטוריה של commit

```
git log                # הצגת ההיסטוריה של Commits
git log --oneline      # הצגת ההיסטוריה בצורה מקוצרת
git diff               # הצגת ההבדלים בין הקבצים ששונו אך לא נוספו ל Staging
git diff --staged      # הצגת ההבדלים בין קבצים שנמצאים כבר ב Commit ל Staging
```

```
git reset <commit_id> # Working Directory-האחרון והשארת השינויים ב Commit-ביטול ה
git reset --hard <commit_id>1 # האחרון ומחיקת השינויים לגמרי ב Commit-ביטול ה
```


ענפים משמשים לעבודה על תכונות חדשות, תיקוני באגים או ניסויים מבלי להשפיע על קוד הבסיס הראשי. הם מאפשרים למפתחים לבודד שינויים עד שהם מוכנים להתמזג לתוך הענף הראשי) שלרוב נקרא master או (main)

```
git branch                # הצגת כל הענפים
git branch new_branch      # יצירת ענף חדש
git switch new_branch      # מעבר לענף חדש (`checkout`-חלופה ל)
git checkout -b new_branch # יצירת ענף חדש ומעבר אליו
git merge new_branch       # מיזוג ענף לתוך הענף הנוכחי
git branch -d new_branch  # מחיקת ענף
```

מיזוג (Merging)

מיזוג הוא התהליך של שילוב שינויים מענף אחד לתוך ענף אחר. לרוב, זה מתבצע כאשר פיצ'ר הושלם והוא מוכן להשתלב בקוד הבסיס הראשי.

Git מאפשר לעבוד עם ריפוזיטוריו מרוחקים המאוחסנים בשרתי פלטפורמות כמו GitHub, GitLab או Bitbucket. זה מאפשר שיתוף פעולה עם מפתחים אחרים ומספק מקום מרכזי לשיתוף הקוד.

<code>git remote -v</code>	<i># Remotes-הצגת רשימת ה</i>
<code>git remote add origin URL</code>	<i># GitHub-חיבור ריפו מקומי ל</i>
<code>git push -u origin main</code>	<i># GitHub-שליחת השינויים הראשונים ל</i>
<code>git pull origin main</code>	<i># משיכת שינויים מהריפו המרוחק</i>
<code>git fetch</code>	<i># בלי למזג GitHub-משיכת עדכונים מ</i>
<code>git rebase main</code>	<i># על הענף הראשי Rebase ביצוע</i>

Pull Requests

Pull Requests הם פיצ'ר בפלטפורמות אירוח של Git כמו GitHub ו GitLab-הם מאפשרים למפתחים להציע שינויים בריפוזיטורי ולבקש שיבחנו אותם וימזגו אותם לתוך הענף הראשי



GitHub שירות אירוח מבוסס ענן

לריפוזיטוריוז של Git.

• מספק פלטפורמה לשיתוף קוד ושיתוף פעולה בין מפתחים.

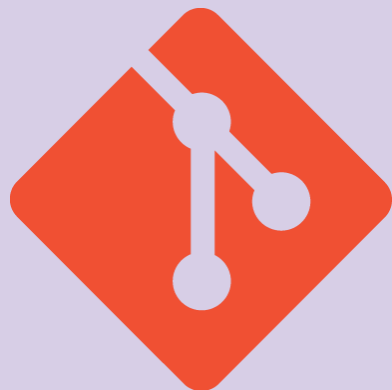
• כולל כלים לניהול פרויקטים, Pull Requests, ביקורת קוד, CI/CD ועוד.

• מאפשר עבודה עם ריפוזיטוריוז מרוחקים, כך שמפתחים יכולים לעבוד יחד על אותו פרויקט.



Git – מערכת מבוזרת לניהול גרסאות (DVCS)

- מאפשר לעקוב אחרי שינויים בקוד, לשמור היסטוריה, לחזור לגרסאות קודמות ולשתף קוד.
- עובד **מקומית** על מחשב המפתח, ללא תלות באינטרנט.
- מאפשר יצירת ענפים, ביצוע מיזוגים וניהול קבצים בשורת הפקודה.



GitHub שירות אירוח מבוסס ענן

- לריפוזיטוריז של Git.
- מספק פלטפורמה לשיתוף קוד ושיתוף פעולה בין מפתחים.
- כולל כלים לניהול פרויקטים, Pull Requests, ביקורת קוד, CI/CD ועוד.
- מאפשר עבודה עם ריפוזיטוריז מרוחקים, כך שמפתחים יכולים לעבוד יחד על אותו פרויקט.