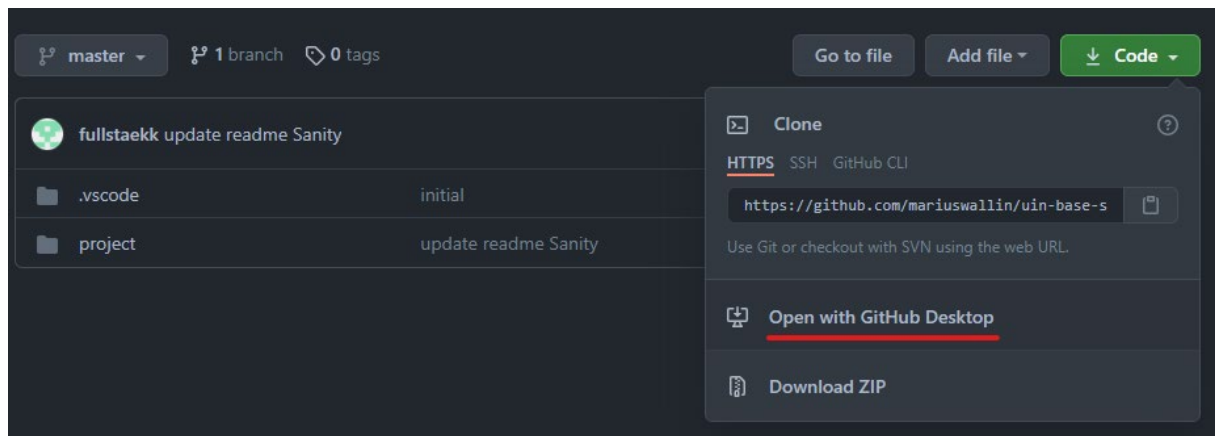


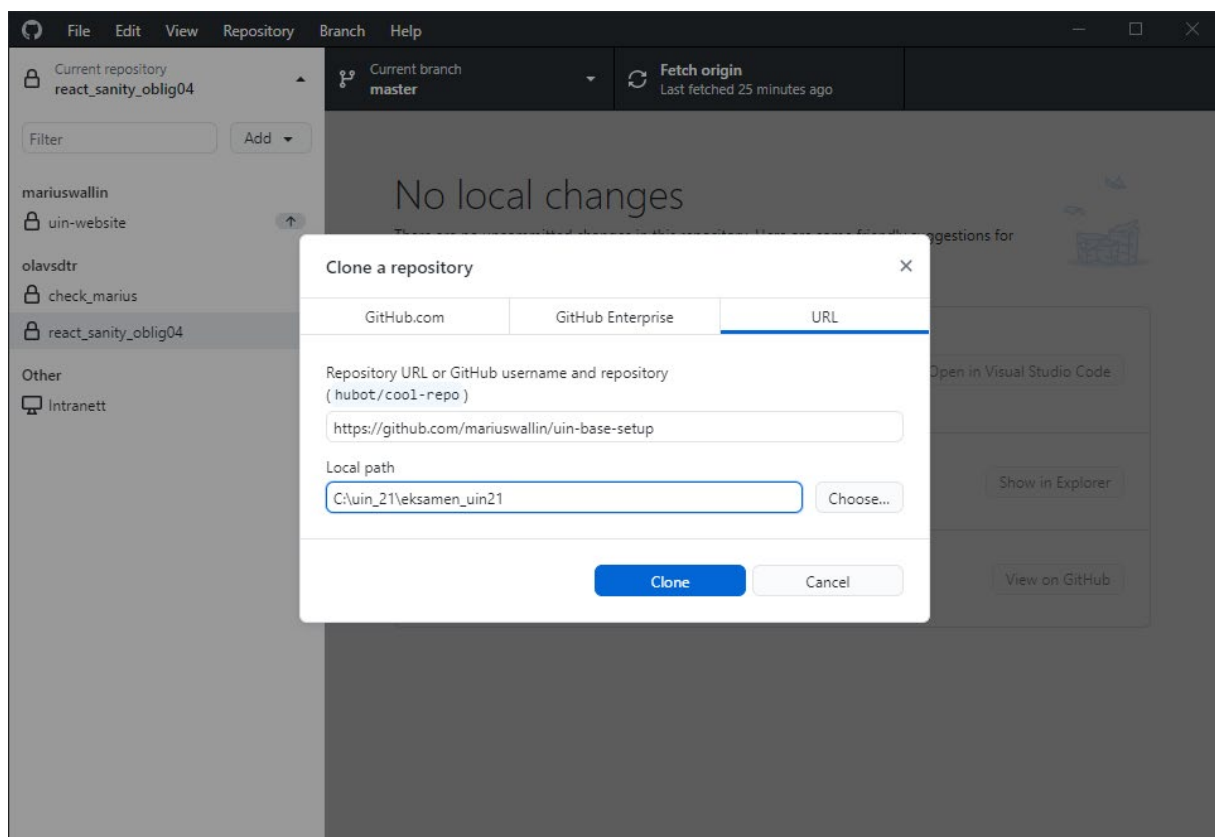
# Hvordan samarbeide med Github Desktop

## Setup:

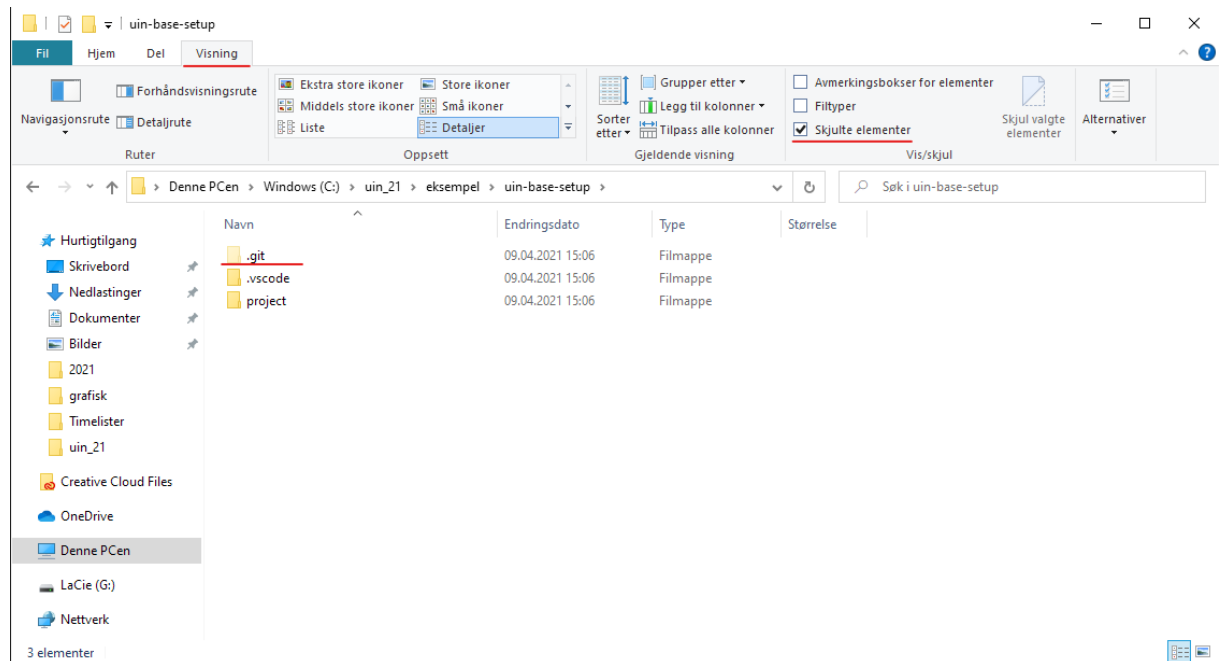
1. En person på gruppen lager en mappe på sin maskin hvor koden til eksamen skal være. Dette mappenavnet bør være det navnet dere ønsker å ha på repoet deres.
2. Samme person henter ned starter-templatene fra UIN-BASE-SETUP ([link](#))



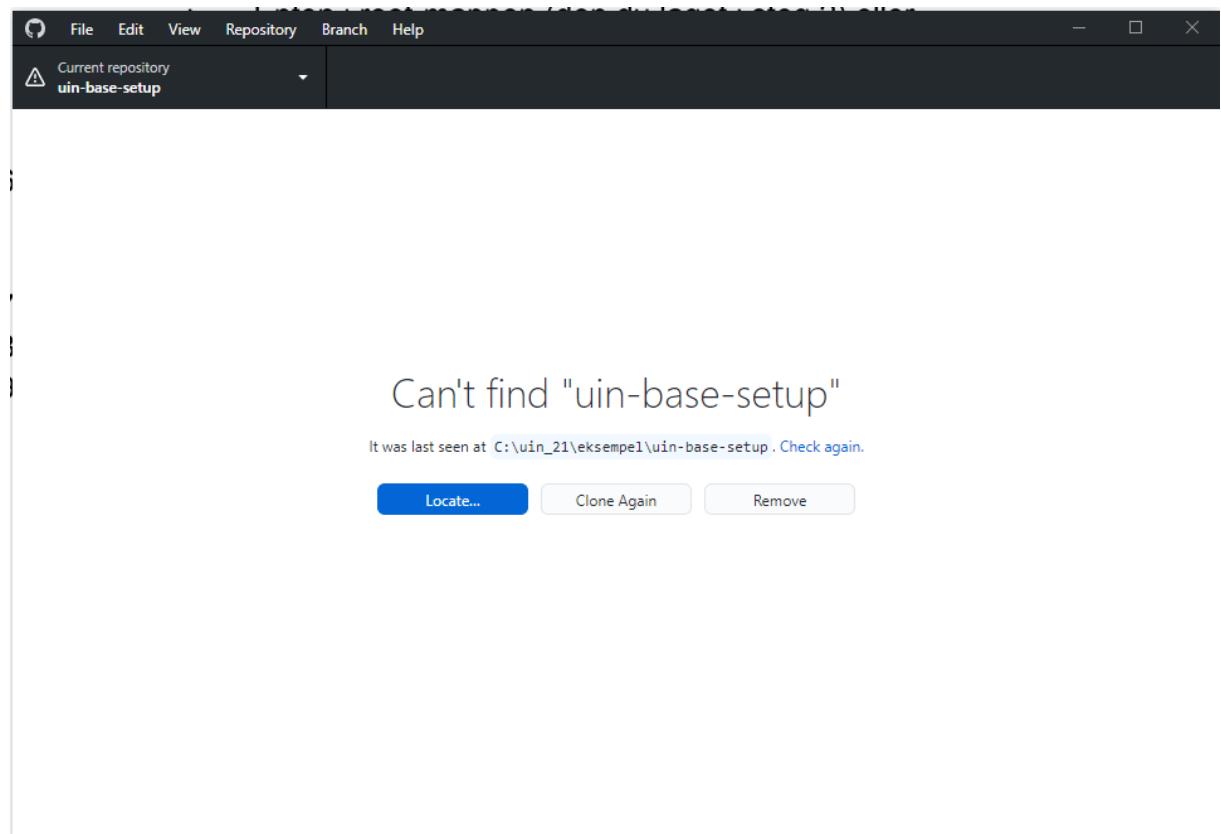
3. Velg mappen som ble lagd i steg 2 på **Local path**, og fjern **\uin-base-setup** (se bildet under)
4. Trykk på **Clone**



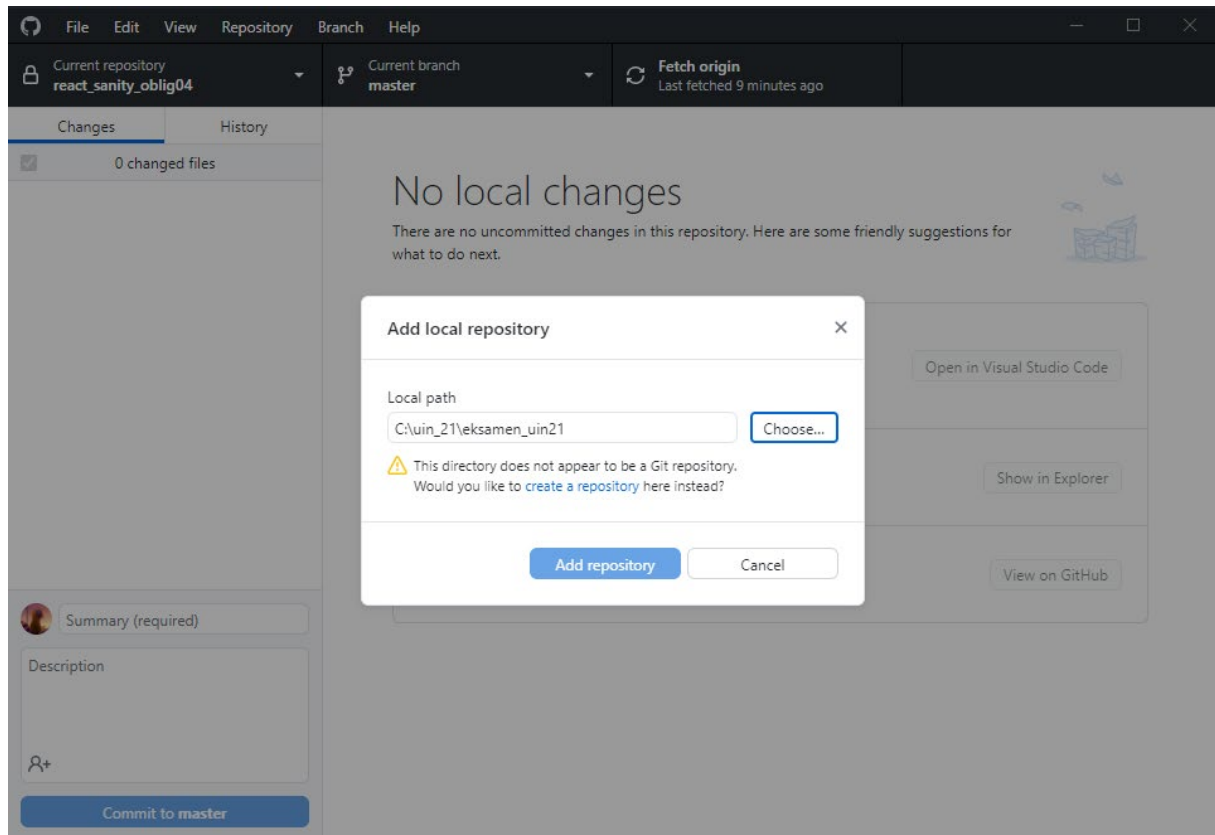
5. Slett **.git** mappen i prosjektet, se røde streker for hvordan vise skjulte elementer i Windows.  
([Hvordan gjøre det på Mac](#))



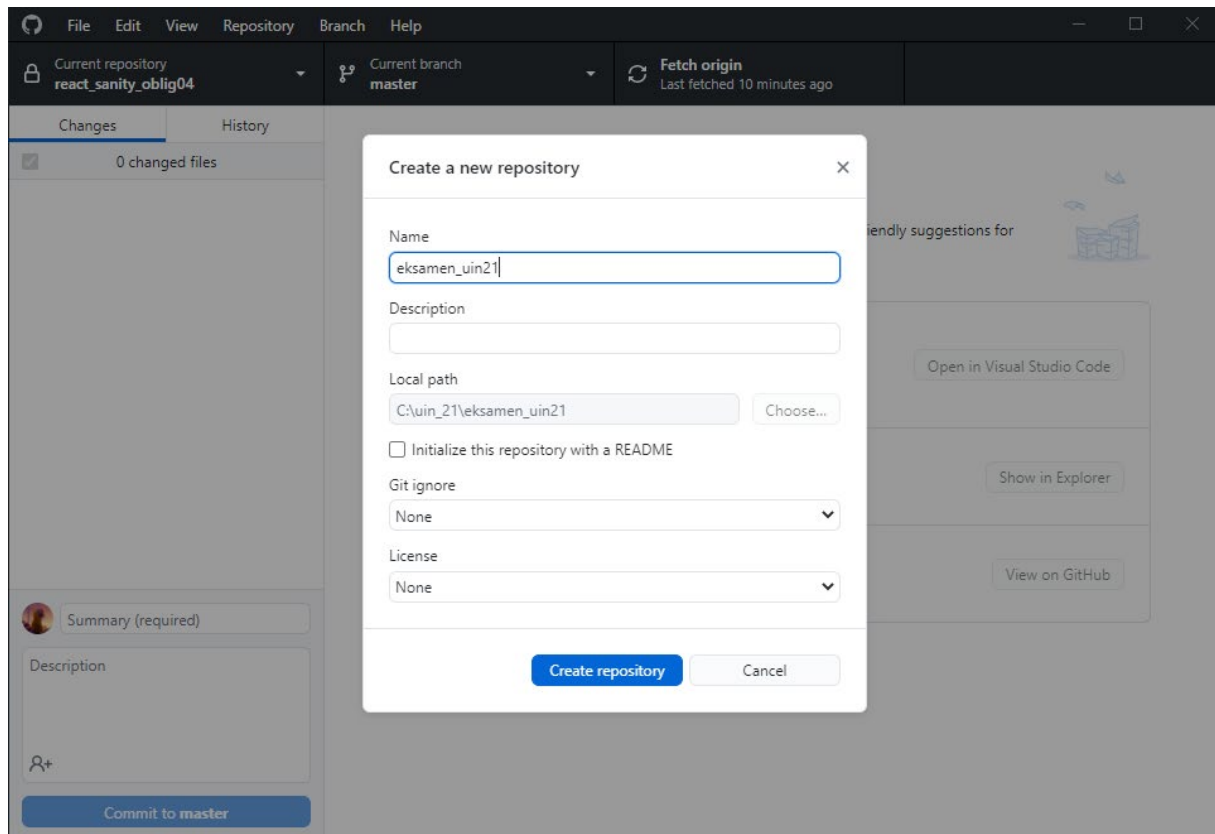
6. Samme person rydder bort det som ikke trengs i starter-templatene
- a. Behold gjerne README.md filer helt frem til dagen dere skal levere (om dere trenger dem for ref)
7. Velg **Remove** i Github Desktop etter å ha slettet **.git** mappen. Dette sletter ikke filene, kun repoets oppføring i Github Desktop.



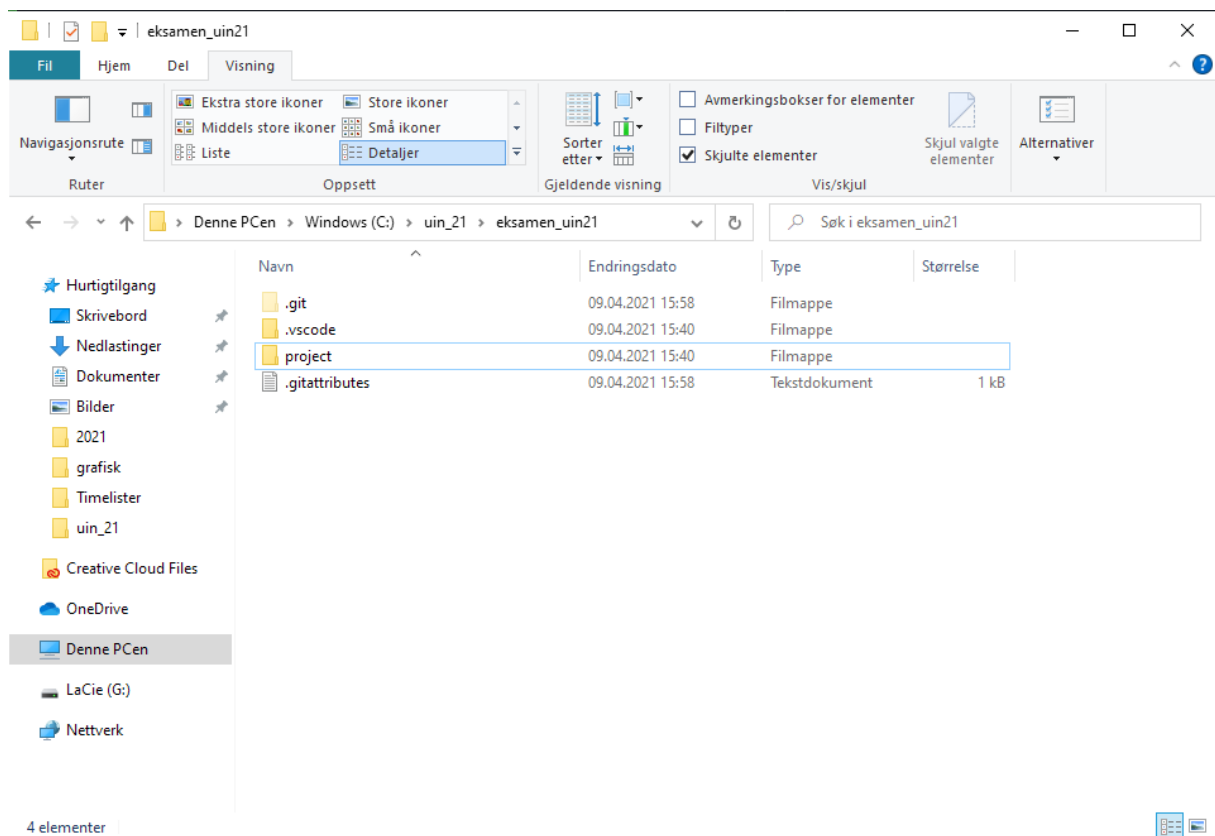
8. Samme person velger **File -> Add local repository...** i Github Desktop. Finn prosjektmappen. Det vil komme opp en beskjed om at mappen ikke er et Git repo. Trykk på **create a repository**.



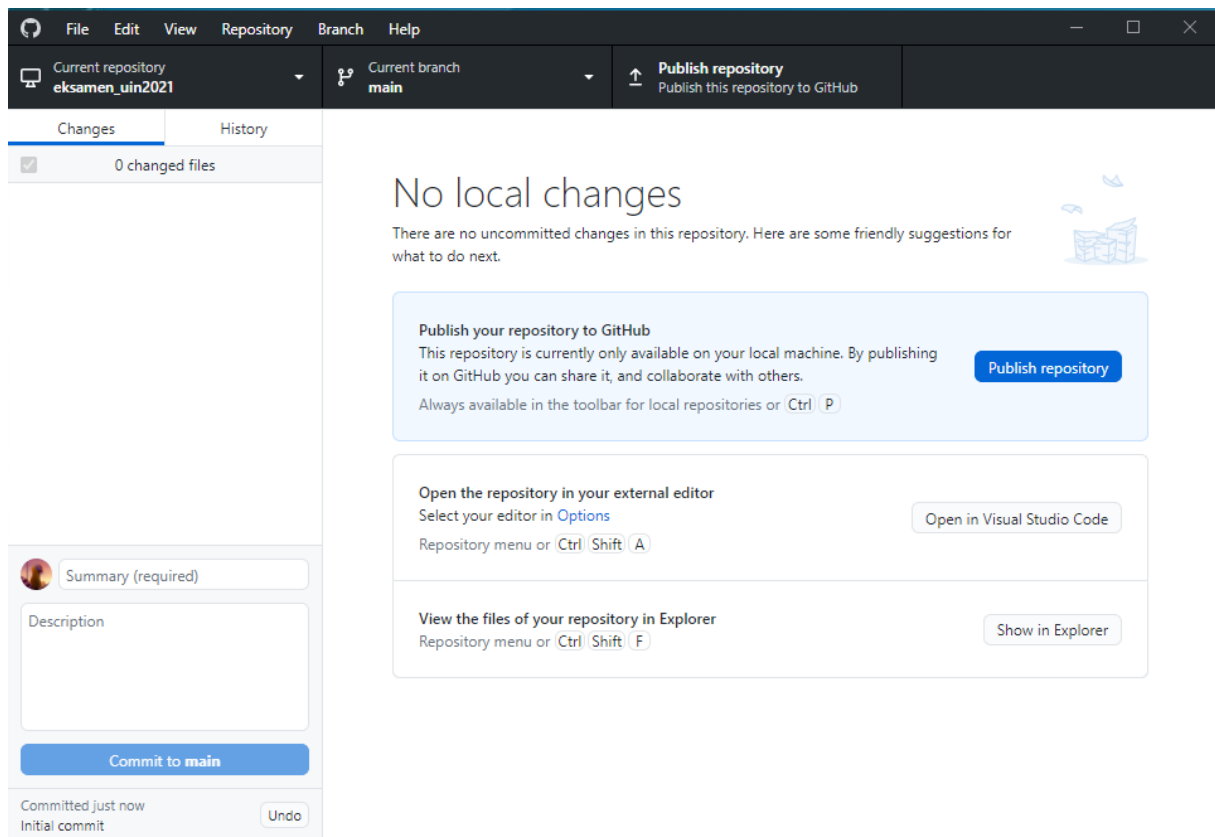
9. **Ikke endre navnet.** Da vil det legges til en ekstra mappe og bli rot. Trykk på **Create repository.**



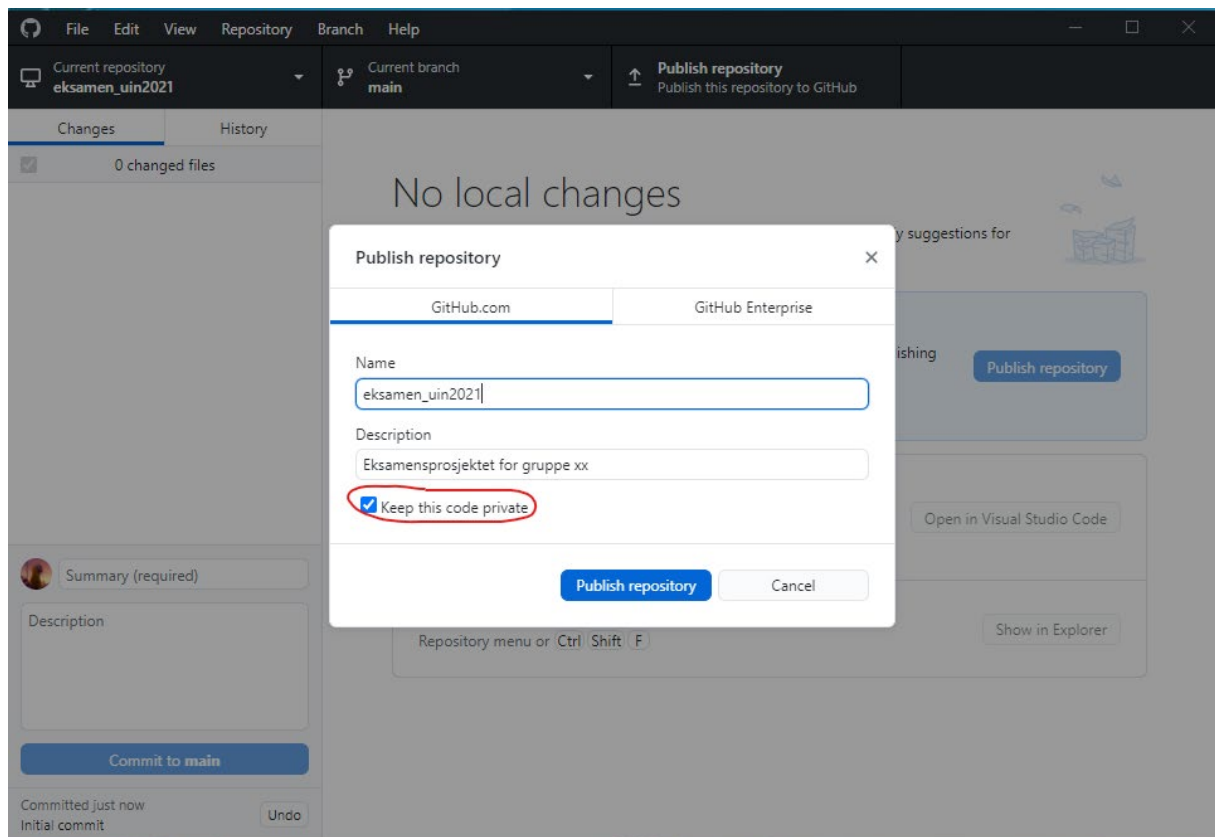
10. Nå har det kommet en `.git` og en `.gitattributes` i mappen.



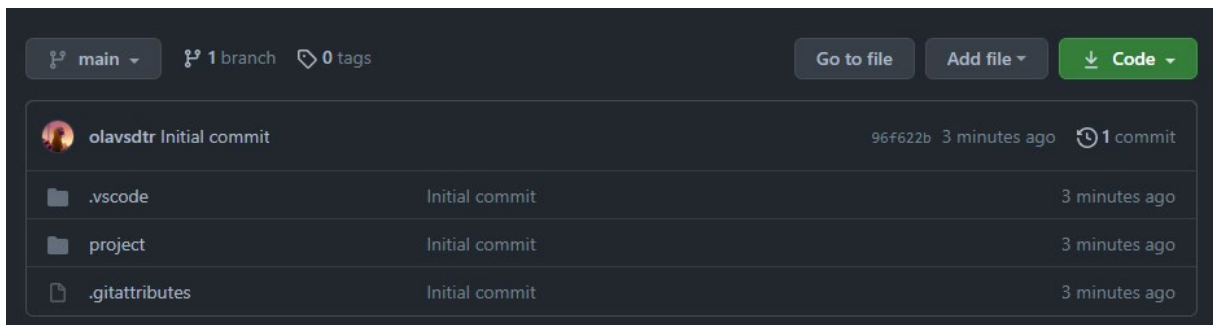
## 11. Trykk på **Publish Repository** i Github Desktop



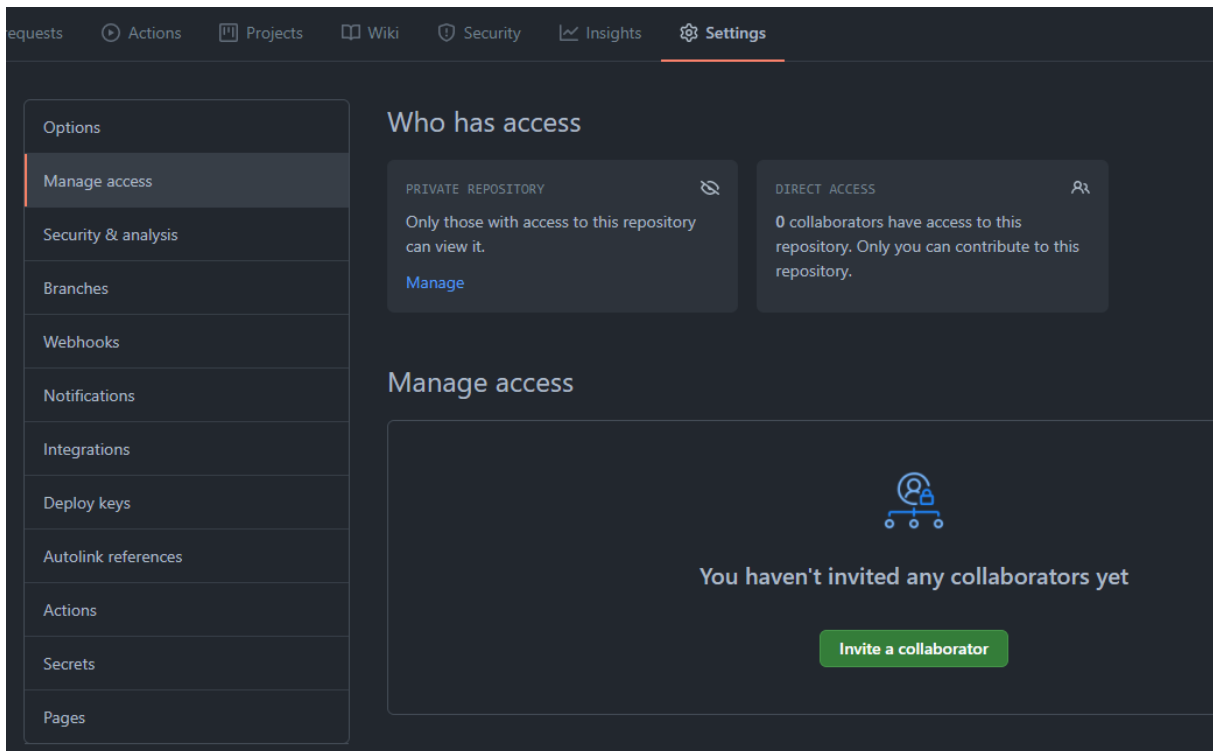
## 12. Her kan du endre reponavnet hvis ønskelig. Vær nøye på at repoet forblir **privat**



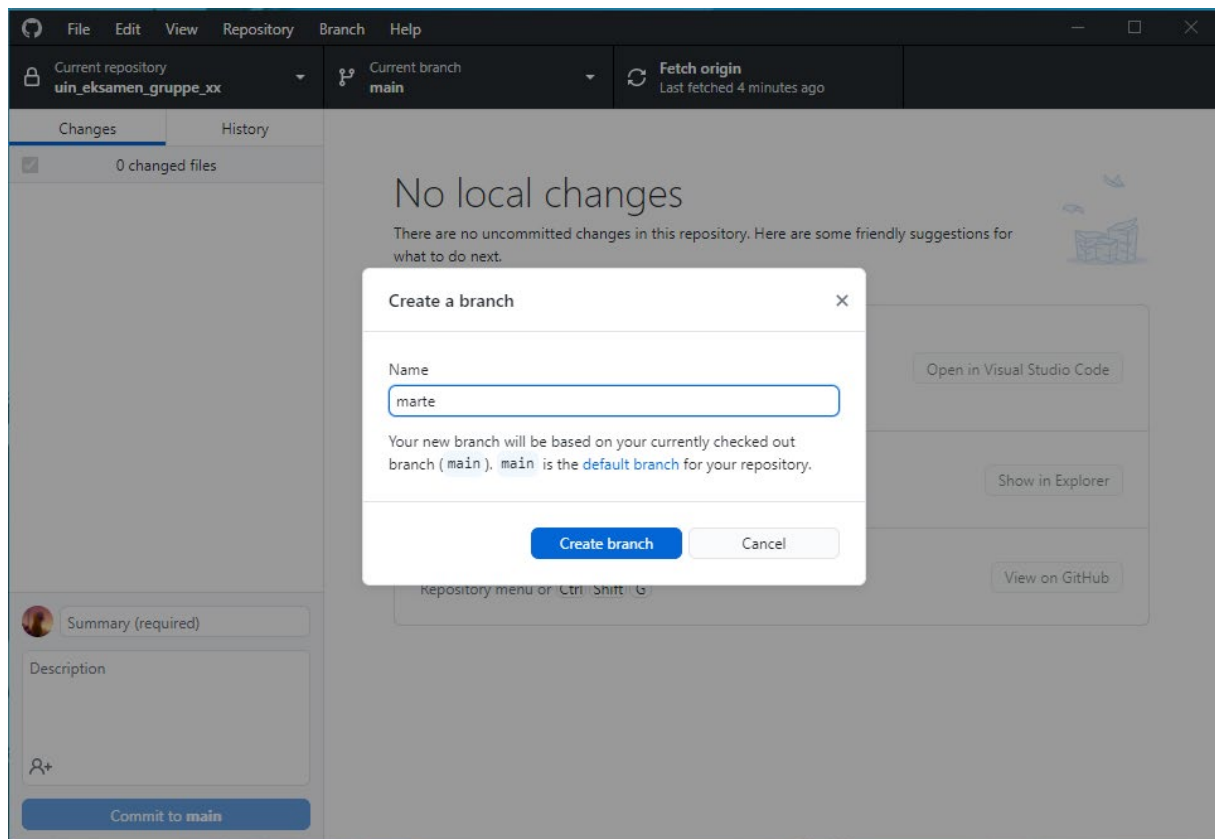
13. Prosjektet er nå publisert på Github.com



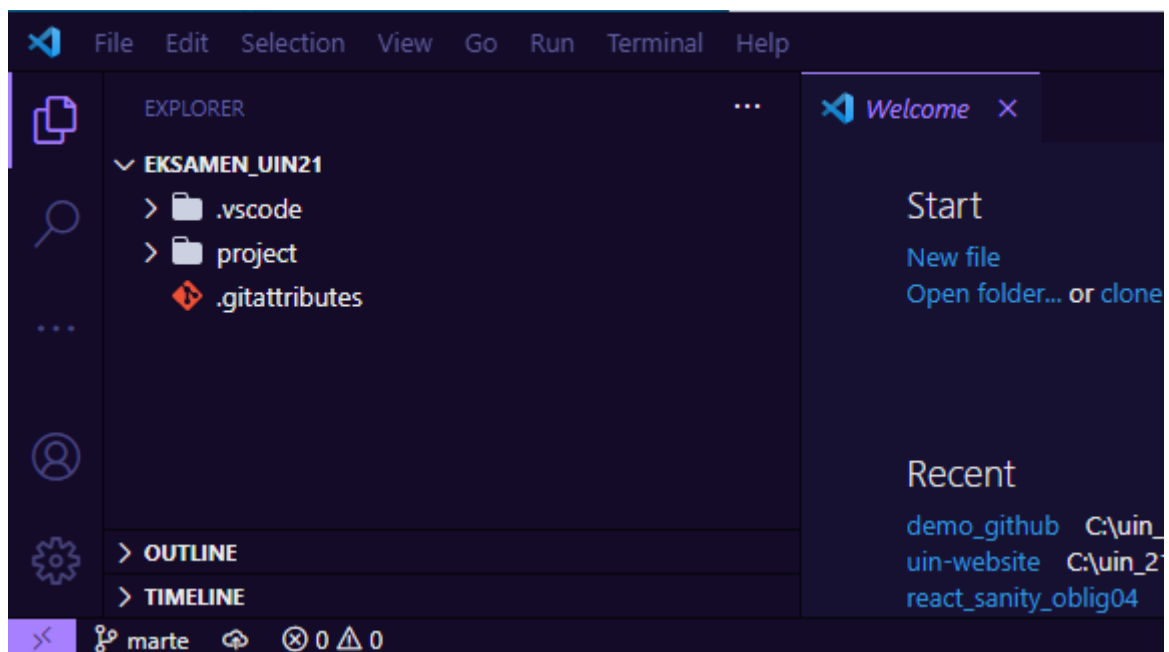
14. Legg til de andre på gruppen i repoet på Github.com ved å trykke på **Invite a collaborator**.



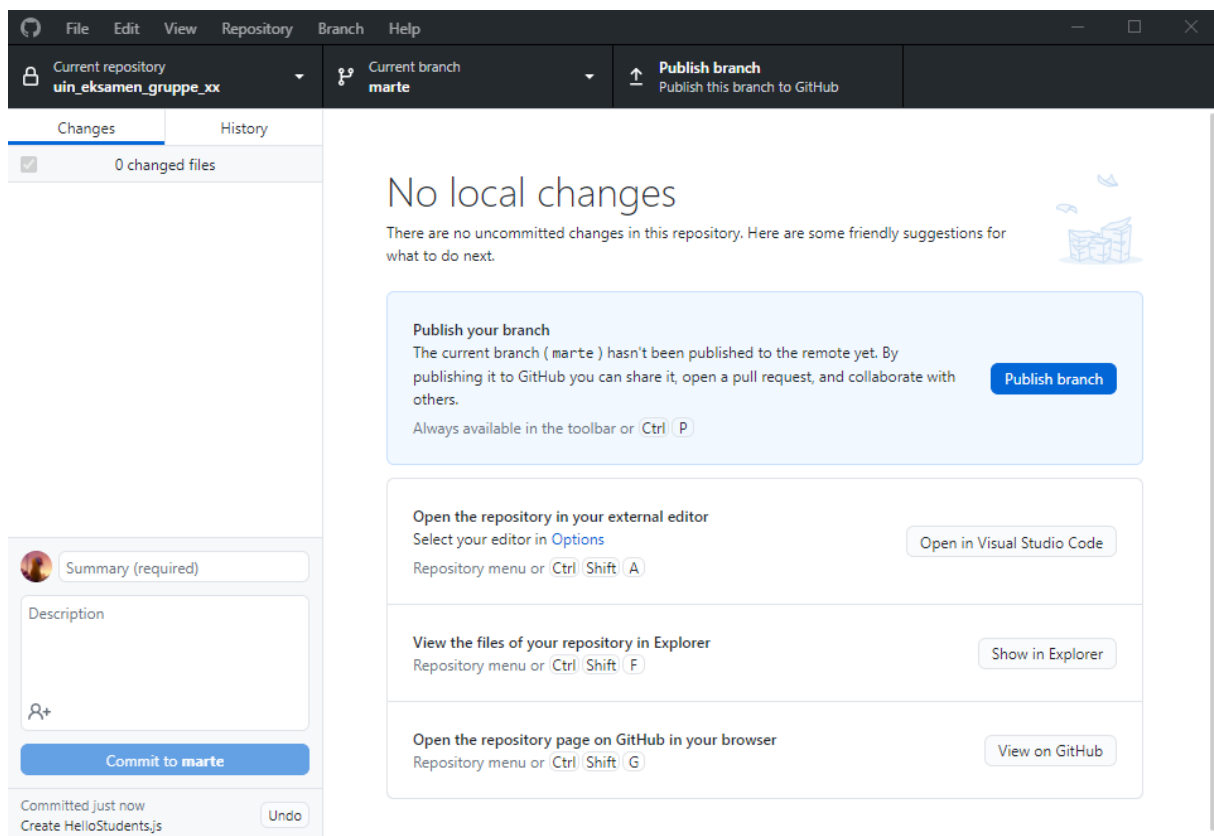
15. Når de andre har godkjent invitasjonen de har fått, kan de clone repoet. Se det første bildet.  
.git-mappen skal ikke slettes nå.
16. **Før kode skrives** skal alle på gruppen lage seg sin egen branch. Branch -> New branch



17. Når branchen er laget, kan man se den nederst til venstre i VS Code. Da er man sikker på at man jobber på riktig branch.



## 18. Branchen må publiseres.

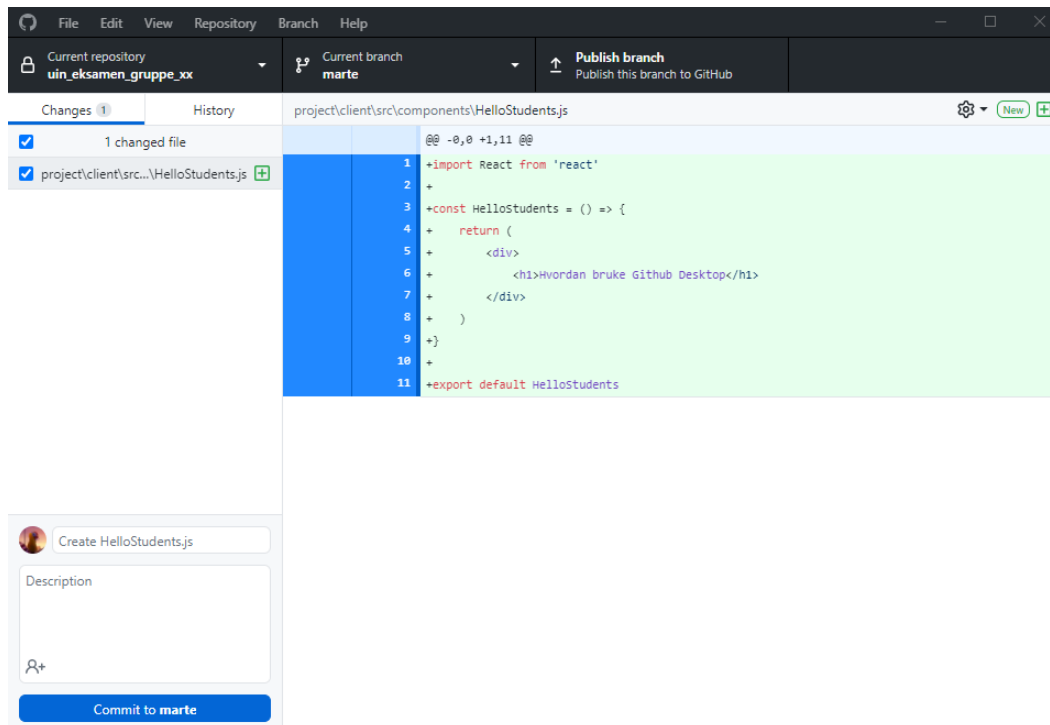


## 19. Nå kan man starte å skrive kode!

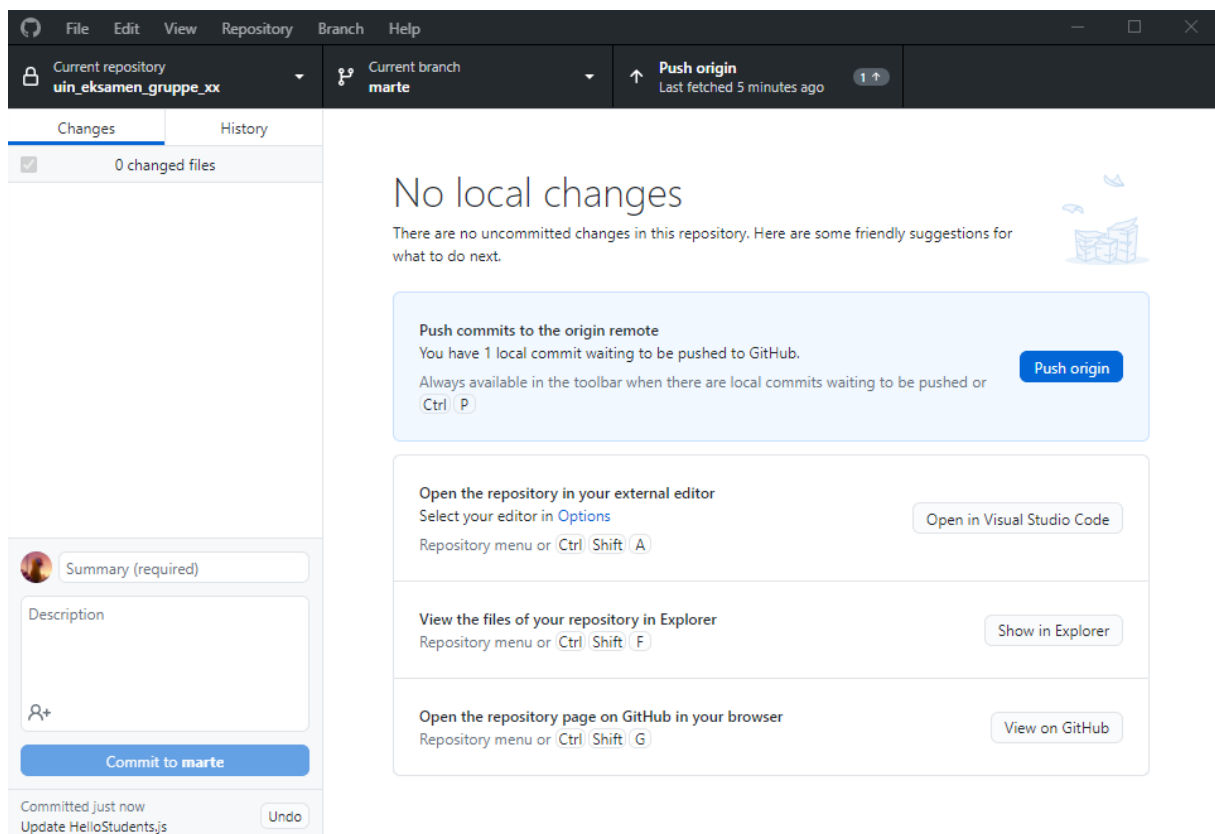


## Lagre det du har gjort og publisere det på Github

1. Endringer registreres automagisk av Github Desktop. Alle filer med endringer er huket av og klar for commit. Hvis du kun ønsker å commite noen få filer, huker du av de du vil commite senere. Skriv en kort tittel som forklarer det du har gjort og trykk på **Commit to ditt-navn**.



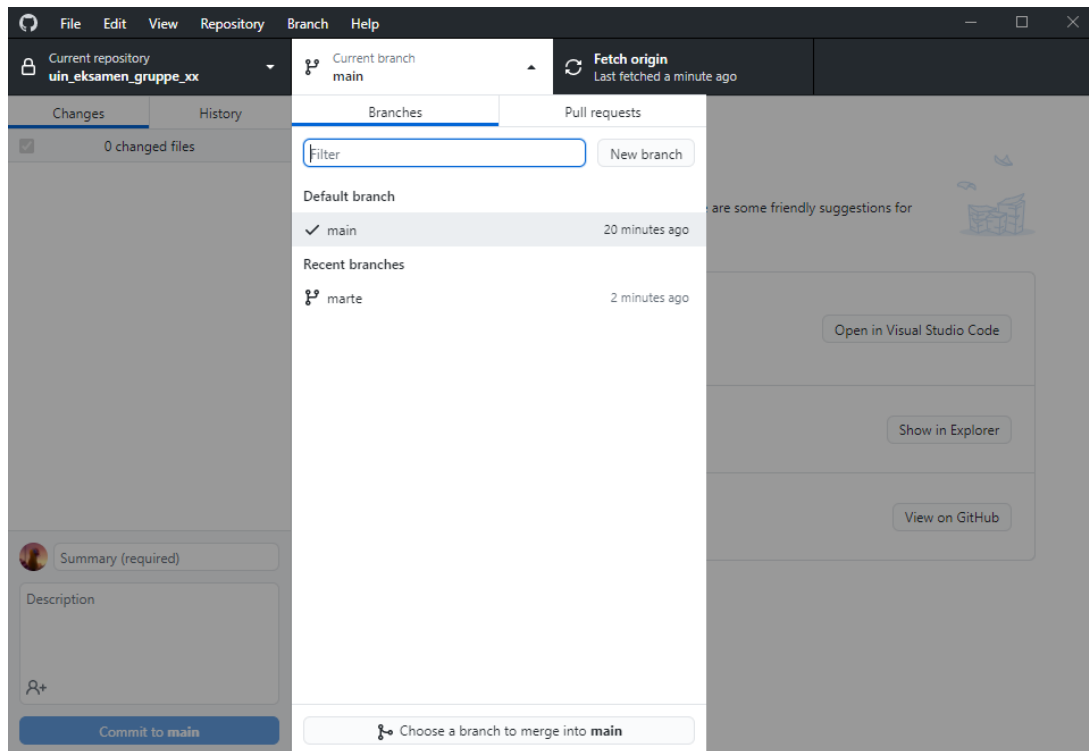
2. Deretter må det publiseres. Trykk på **Push Origin**.



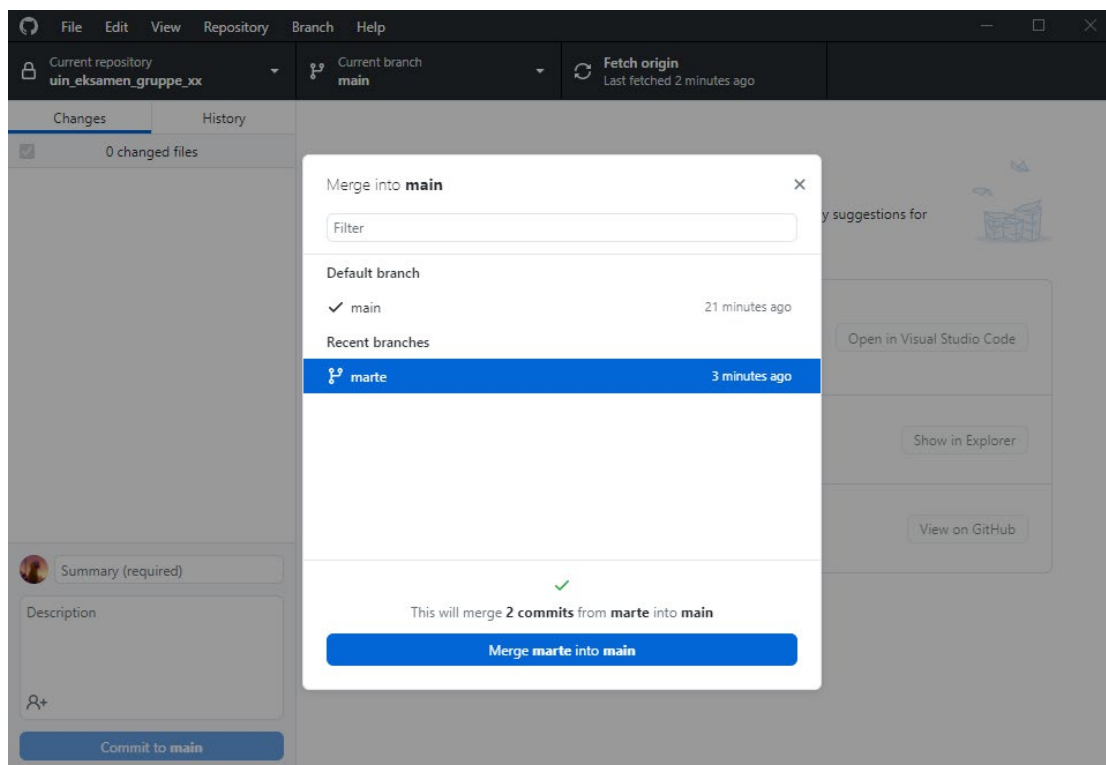
## Merge din kode inn til master

Gjør dette når du vil at hovedbranchen (der all ferdig kode skal være) skal inneholde dine endringer.

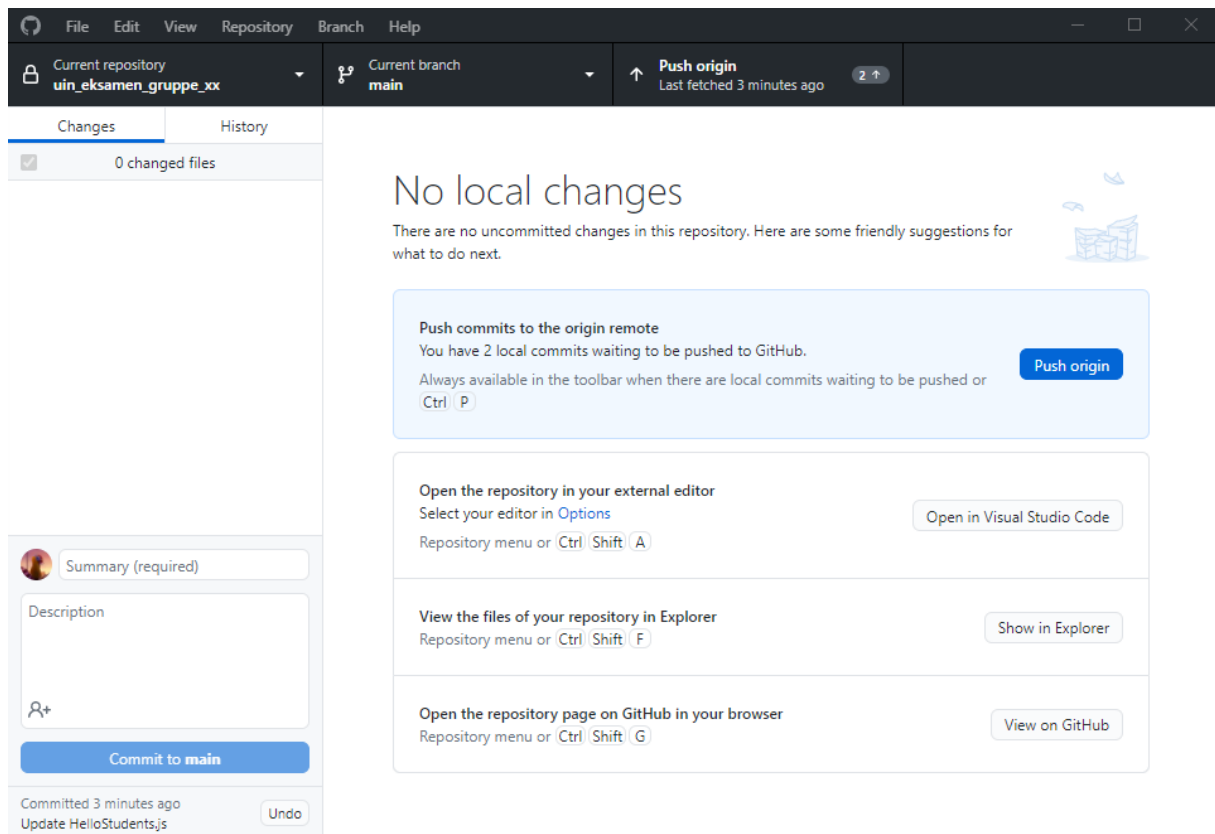
1. Trykk på main eller master istedenfor din branch



2. **Choose a branch to merge into main**, står nederst på bildet over. Velg din branch.



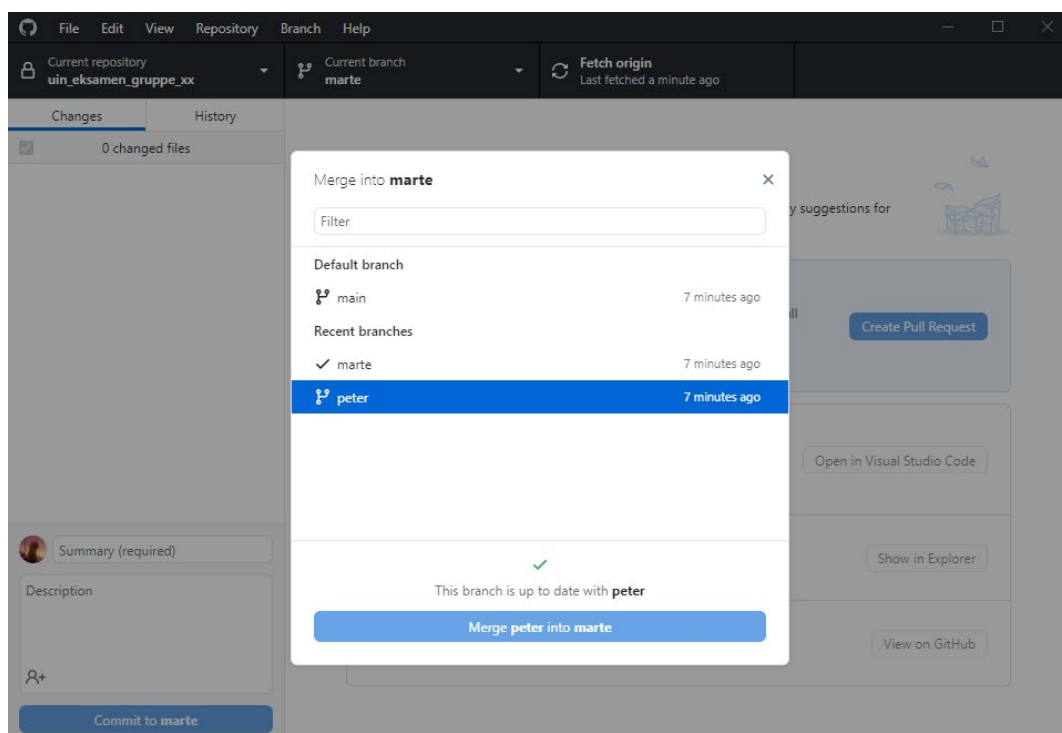
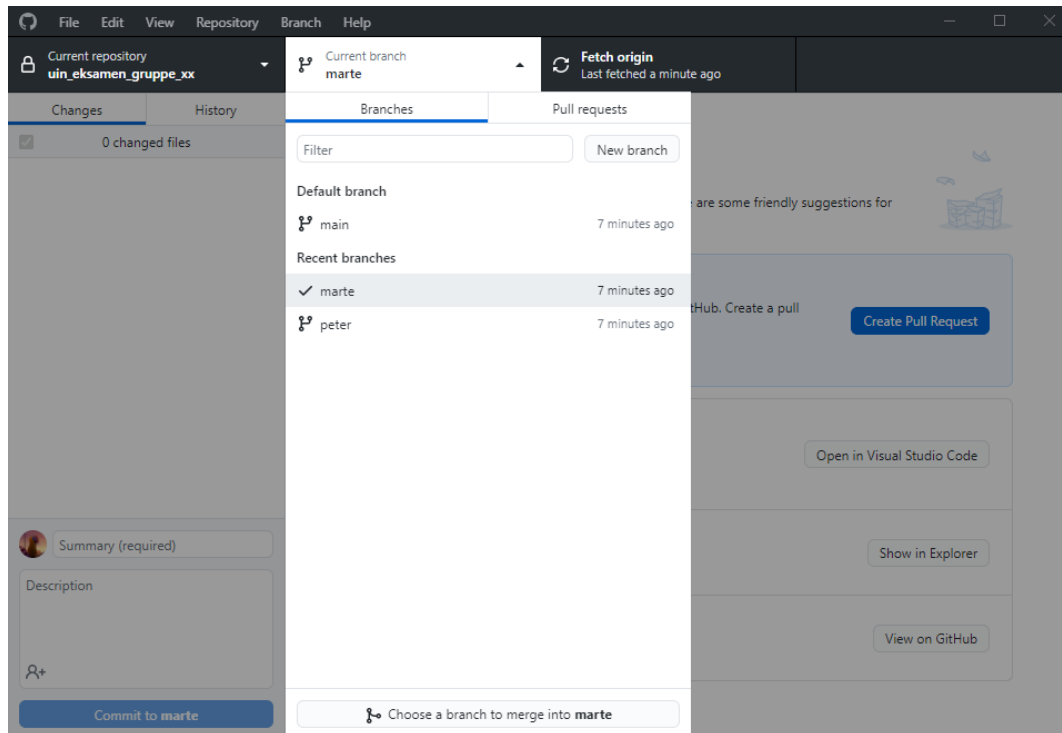
### 3. Husk **Push origin** etterpå.



## Merge et gruppemedlems kode inn i din branch

Gjør dette når du trenger en medstudents kode. **Før du kan gjøre dette må du commite dine kode først.**

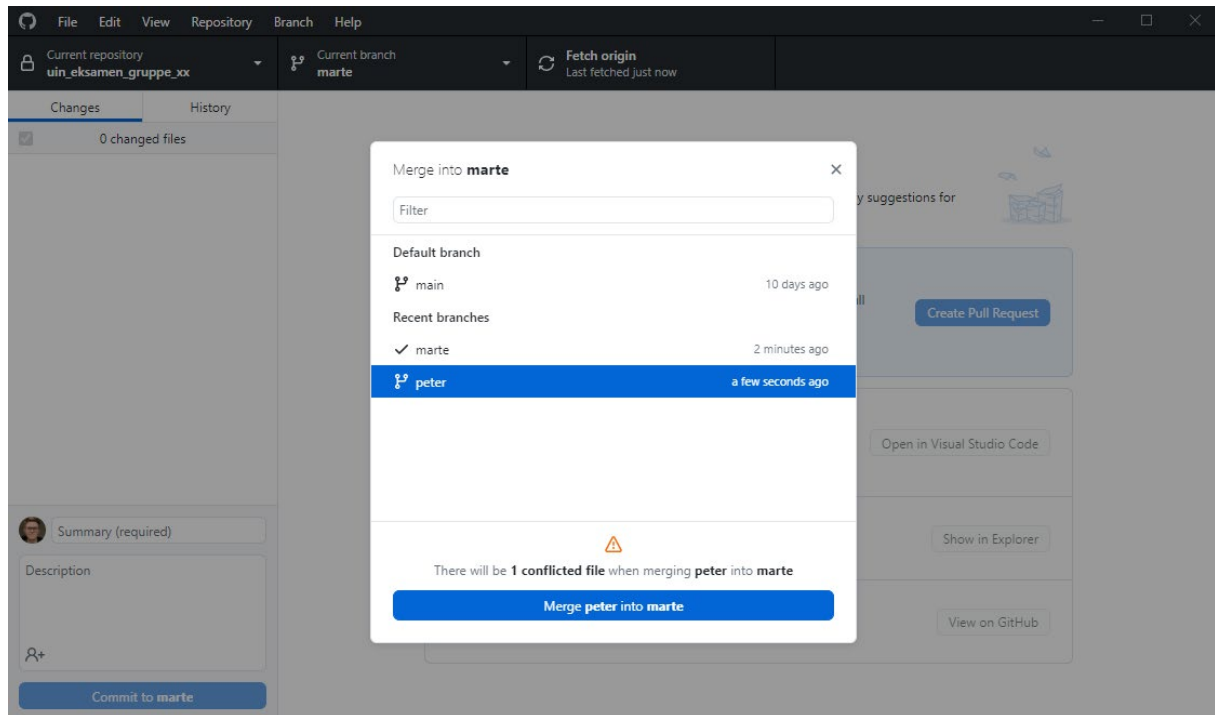
1. Hvis du ikke får opp medstudents branch, pull eller fetch origin.
2. Trykk på medstudents branch.
3. Fetch origin
4. I **Branch**, velg din egen branch igjen.
5. Trykk på Choose a branch to merge into ditt-navn



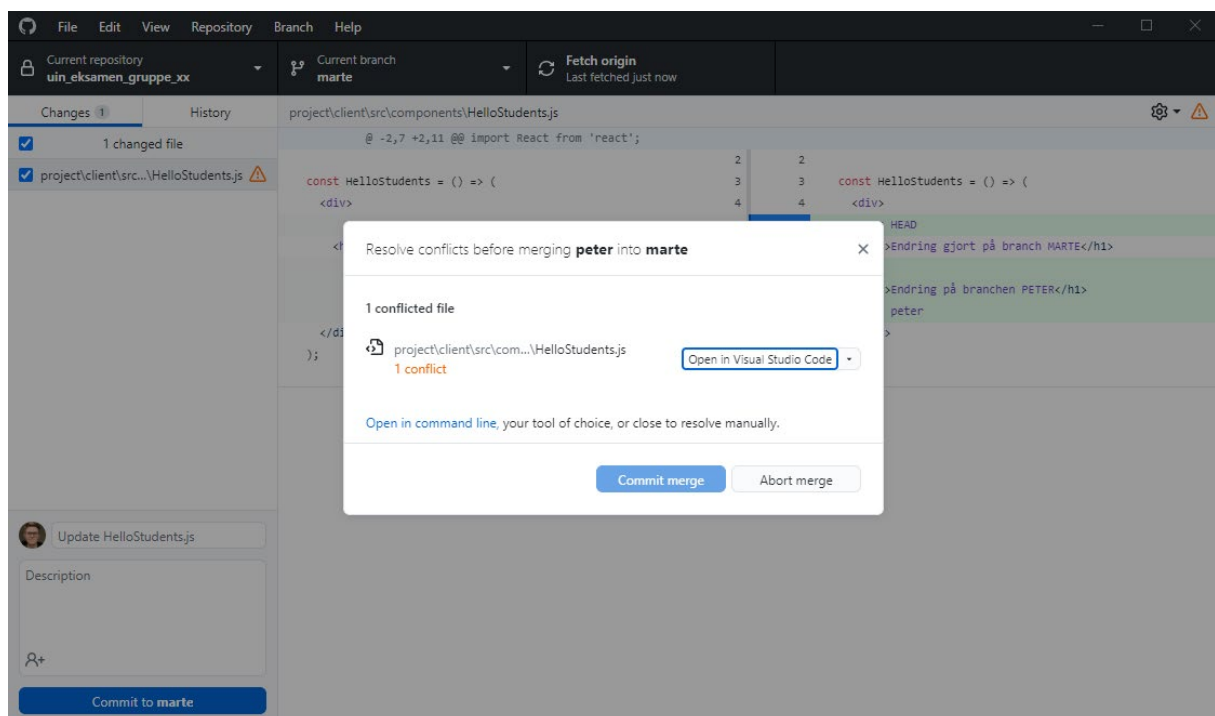
## Merging conflict

Hvis det er to grener med endringer på samme sted i samme fil, vil det oppstå en merging conflict. Da er man nødt til å velge hvilken endring man ønsker å beholde ved merging.

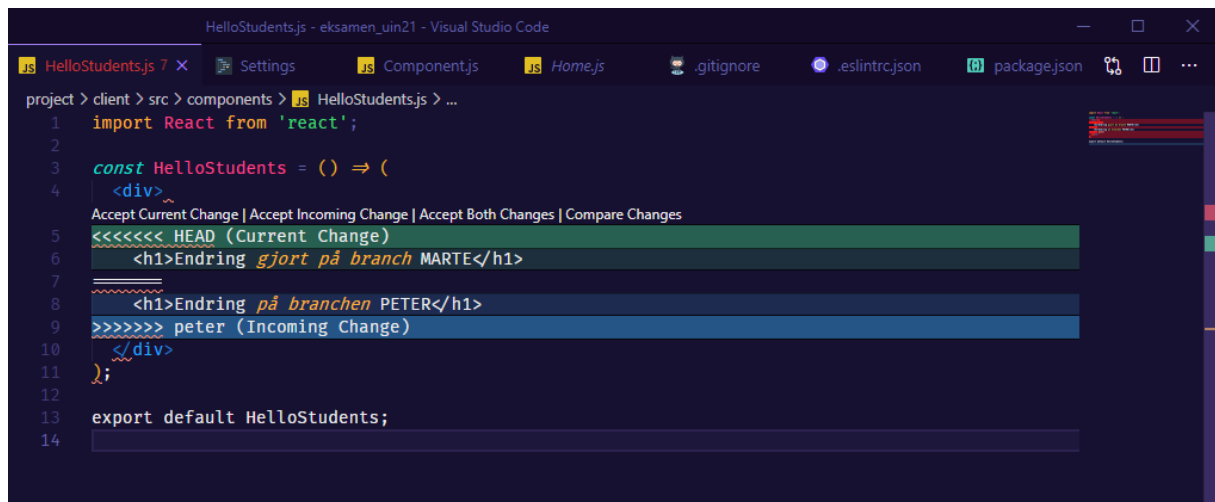
1. Denne meldingen vil komme opp i Github Desktop
2. Trykk Merge --- into ---



3. Denne meldingen vil komme opp, velg Open in Visual Studio Code

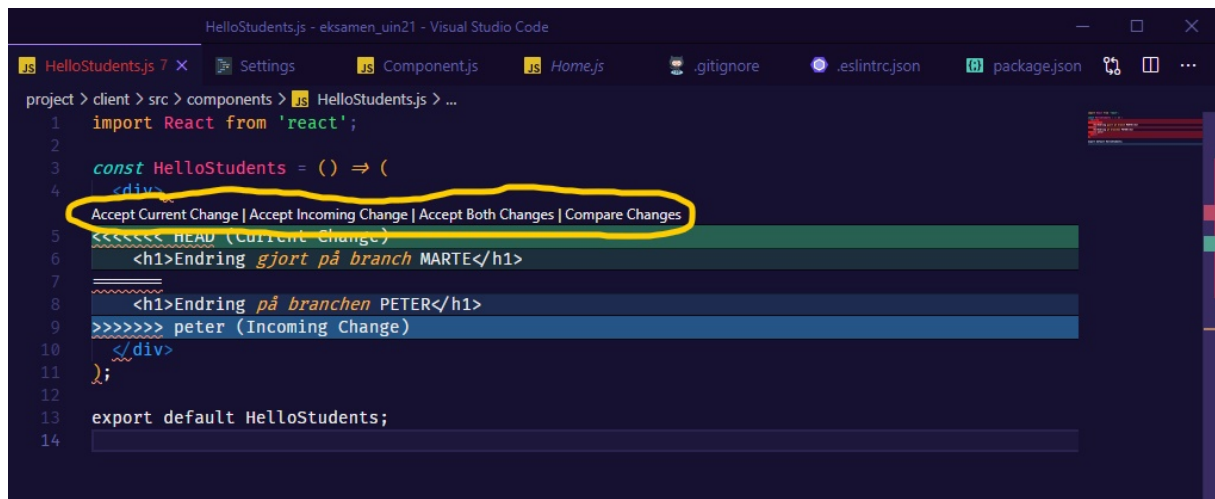


#### 4. Endringene vil vises i Visual Studio Code



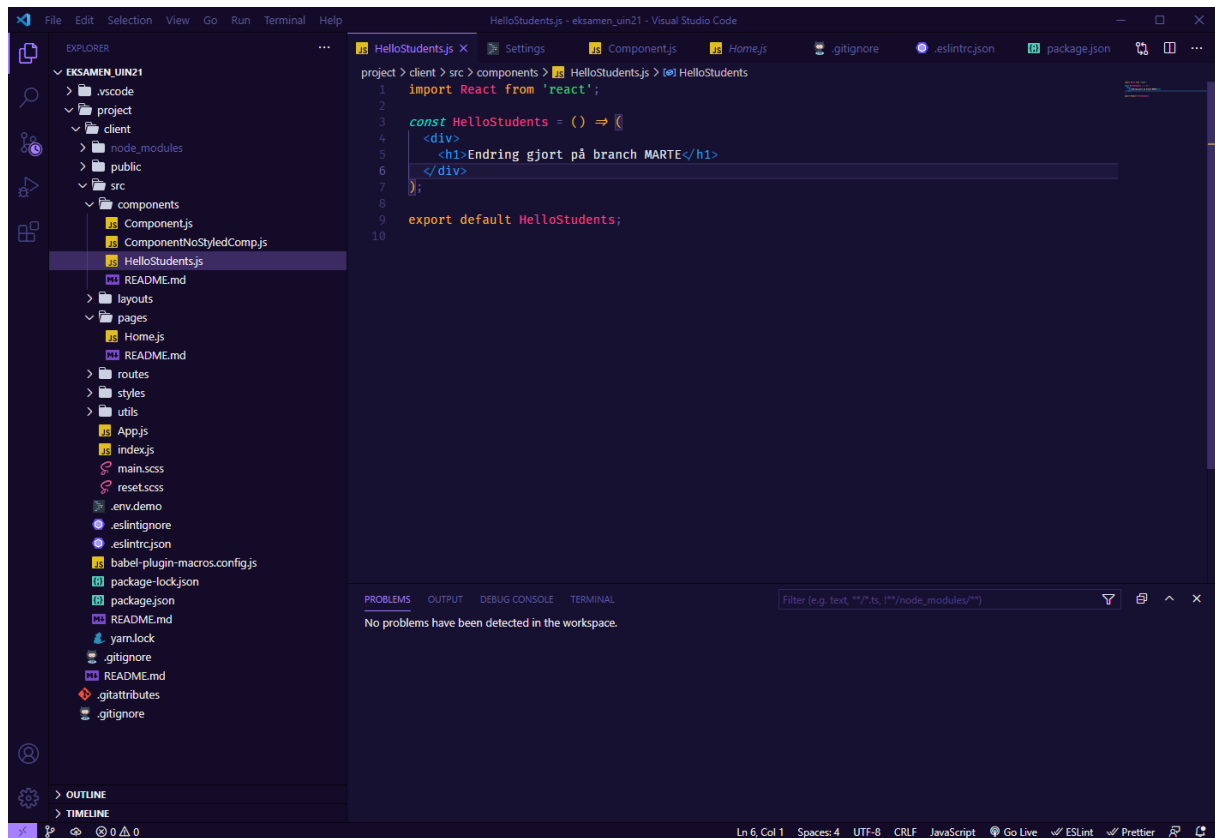
```
project > client > src > components > HelloStudents.js > ...
1  import React from 'react';
2
3  const HelloStudents = () => (
4    <div>
5      <<<<<< HEAD (Current Change)
6      <h1>Endring gjort på branch MARTE</h1>
7      </div>
8      <h1>Endring på branchen PETER</h1>
9      >>>>>> peter (Incoming Change)
10     </div>
11   );
12
13   export default HelloStudents;
14
```

5. Accept Current Change (den grønne) eller Incoming Change (den blå), eventuelt begge endringer.

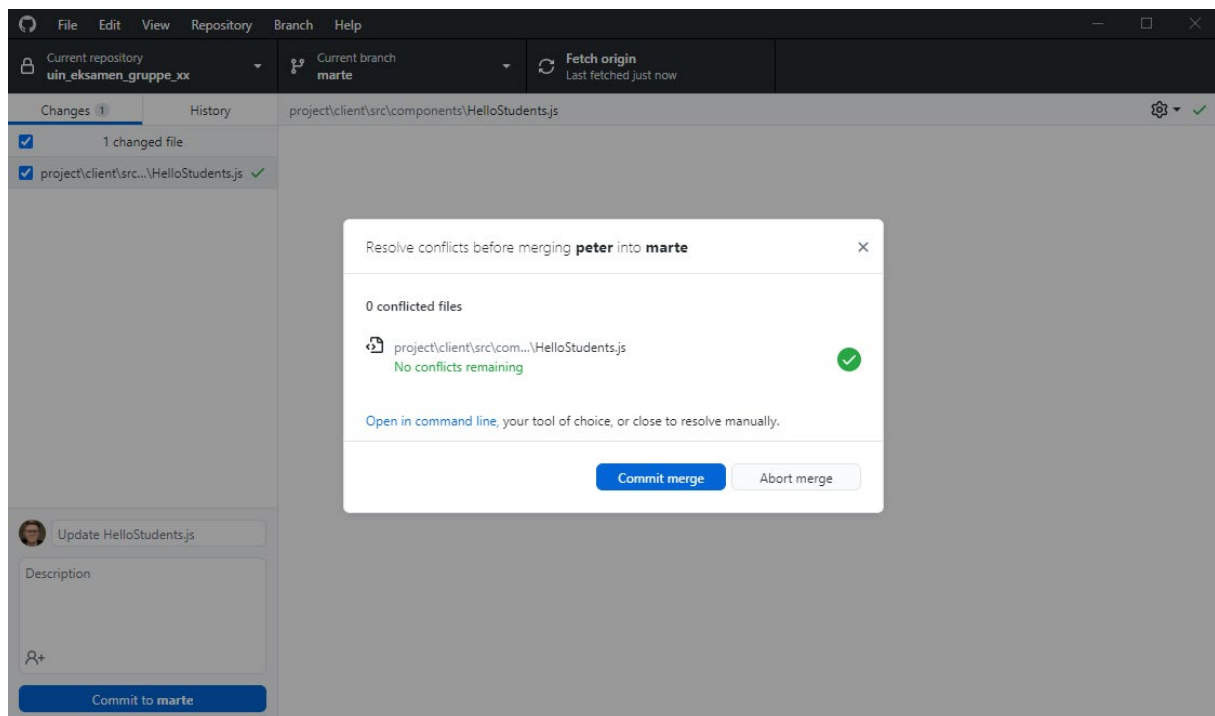


```
project > client > src > components > HelloStudents.js > ...
1  import React from 'react';
2
3  const HelloStudents = () => (
4    <div>
5      <<<<<< HEAD (Current Change)
6      <h1>Endring gjort på branch MARTE</h1>
7      </div>
8      <h1>Endring på branchen PETER</h1>
9      >>>>>> peter (Incoming Change)
10     </div>
11   );
12
13   export default HelloStudents;
14
```

6. Jeg ville beholde endringen på Current Change (branchen marte) og trykker på Accept Current Change. Da vil VS Code fjerne den andre.



7. I Github Desktop får du nå beskjed om at det ikke er noen konflikter lenger.



8. Commit merge og push origin
9. Finito!