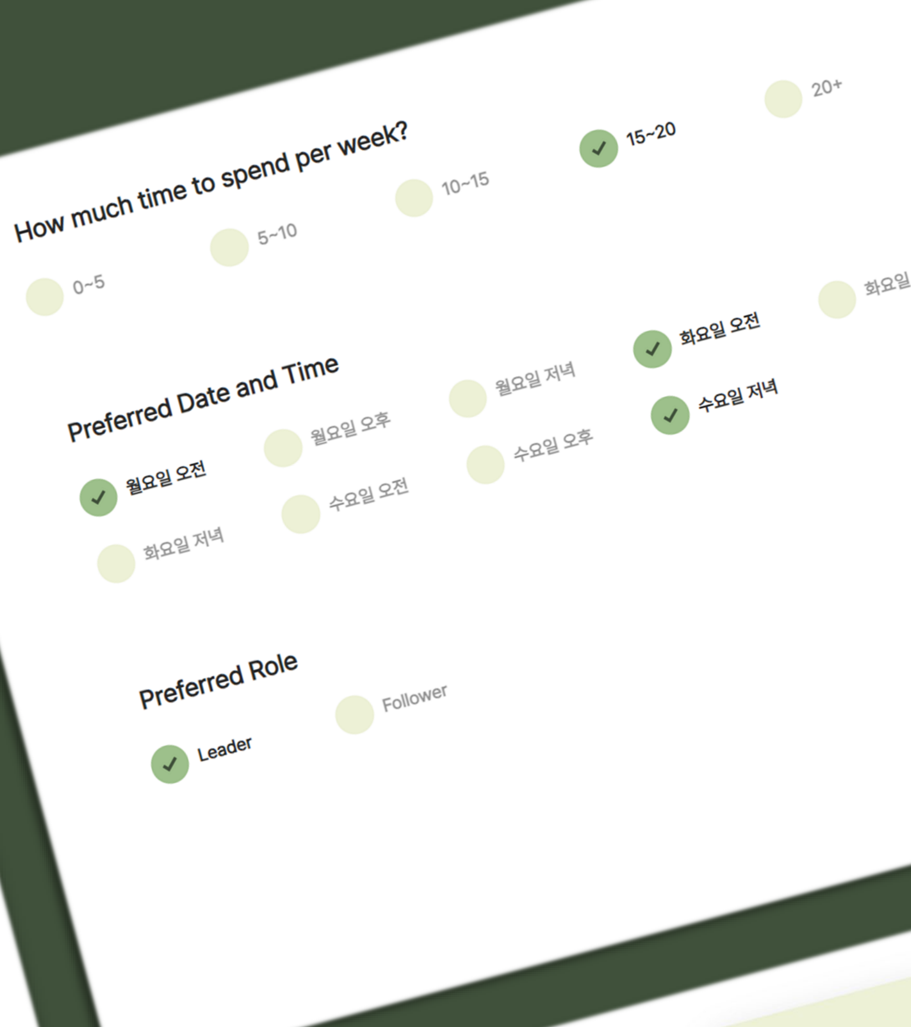
merge!

Team Matching
Based on Opitmal Algorithm
for Any Developers
and Designers



How much time to spend per week?

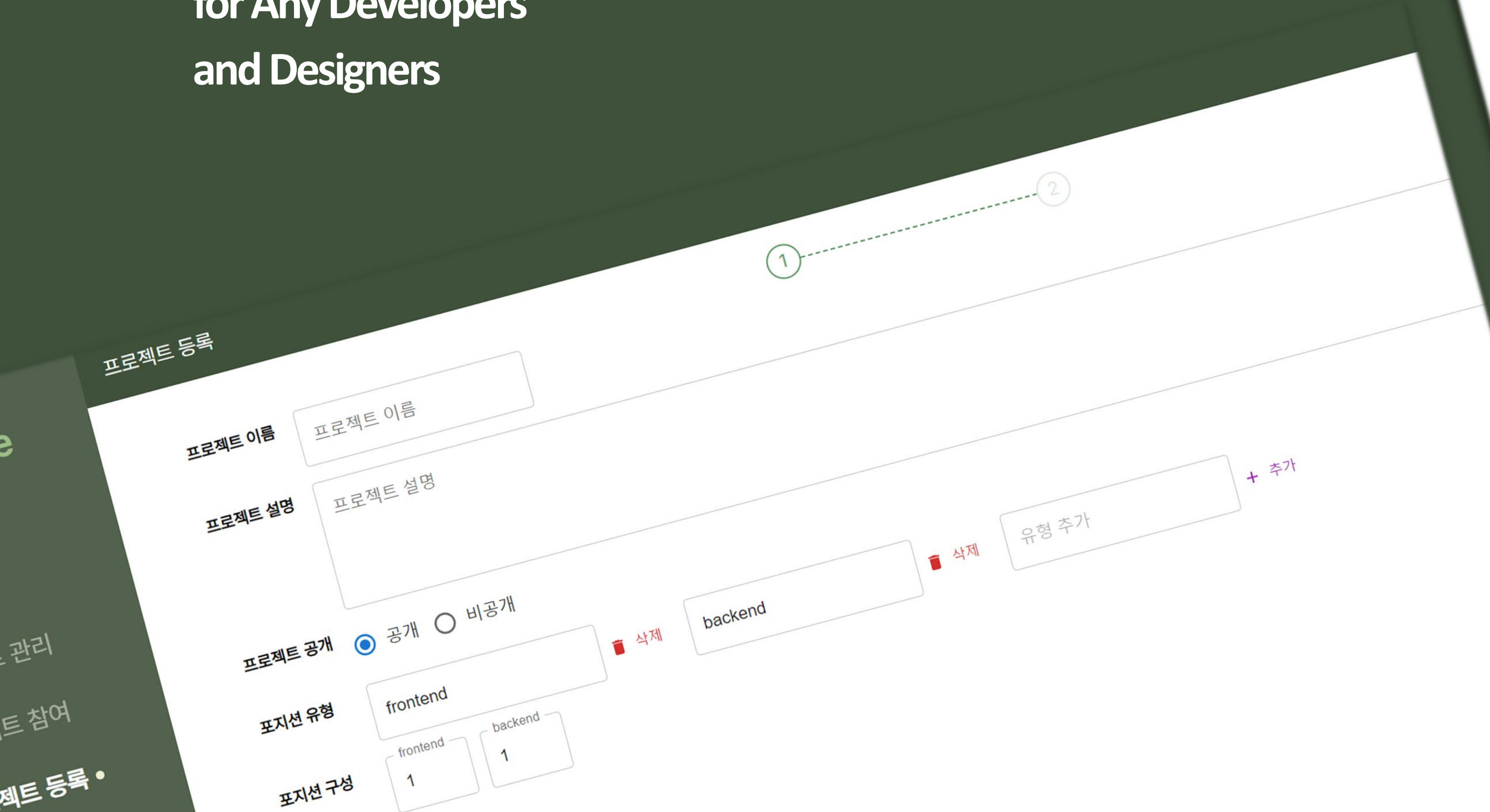
○ 0~5 ○ 5~10 ○ 10~15 ✓ 15~20 ○ 20+

Preferred Date and Time

✓ 월요일 오전 ○ 월요일 오후 ○ 월요일 저녁 ✓ 화요일 오전 ○ 화요일
○ 화요일 저녁 ○ 수요일 오전 ○ 수요일 오후 ✓ 수요일 저녁

Preferred Role

✓ Leader ○ Follower



프로젝트 등록

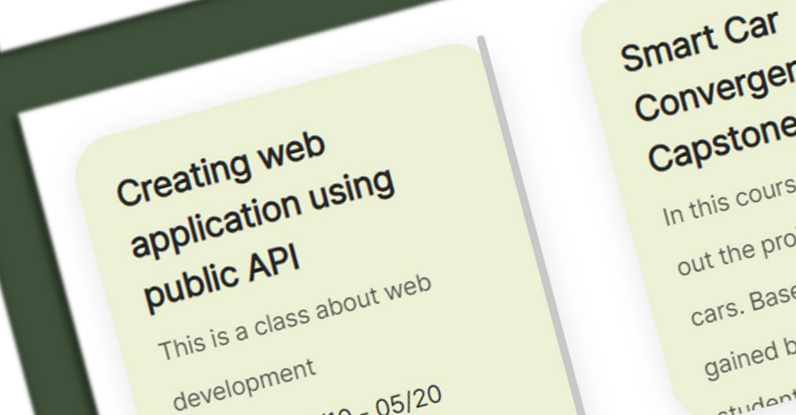
프로젝트 이름: 프로젝트 이름

프로젝트 설명: 프로젝트 설명

프로젝트 공개: ● 공개 ○ 비공개

포지션 유형: frontend 삭제 | backend 삭제 | 유형 추가 + 추가

포지션 구성: frontend 1 | backend 1



Creating web application using public API

This is a class about web development

10 - 05/20

Smart Car Convergen Capstone

In this cours out the pro cars. Base gained b student

Objective and Motivation

Optimal Matching

merge! builds each dev/design collaboration team based on a matching algorithm

Anyone Register

merge! allows anyone to register a project and form a team.

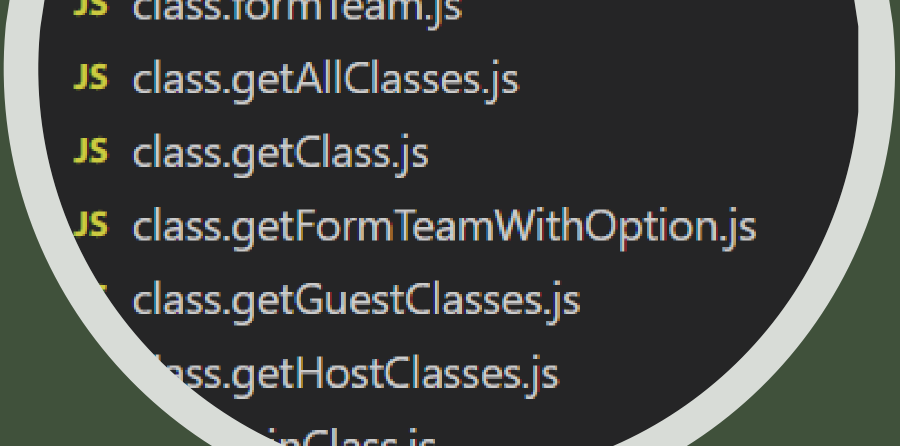
Team Chatting

merge! also provides a communication space for collaboration after team formation.

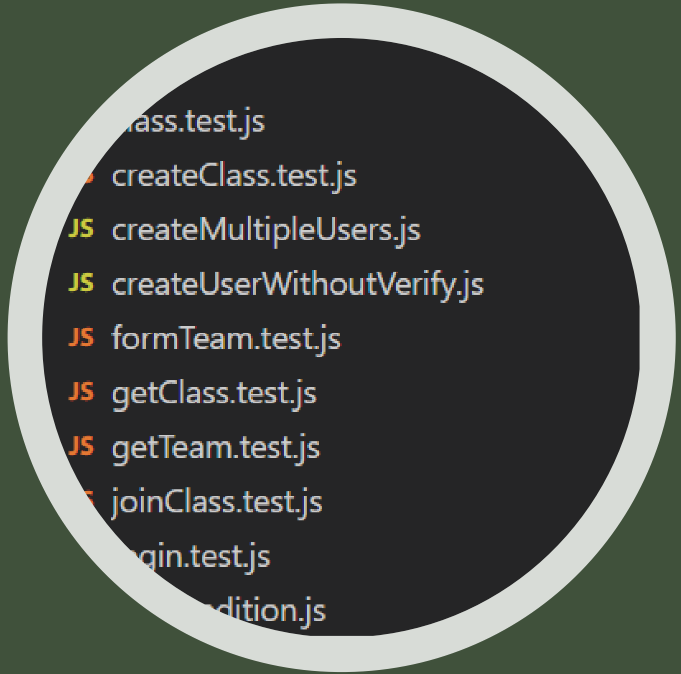
Project Progress

Week	3	4	5	6	7	8	9	10	11	12	13	14
Planning and DB design	🟩											
Register / Login		🟩	🟩									
UI / UX				🟩								
Create / Join Project					🟩	🟩	🟩					
Matching Algorithm					🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Team chat										🟩		
Usability test											🟩	🟩

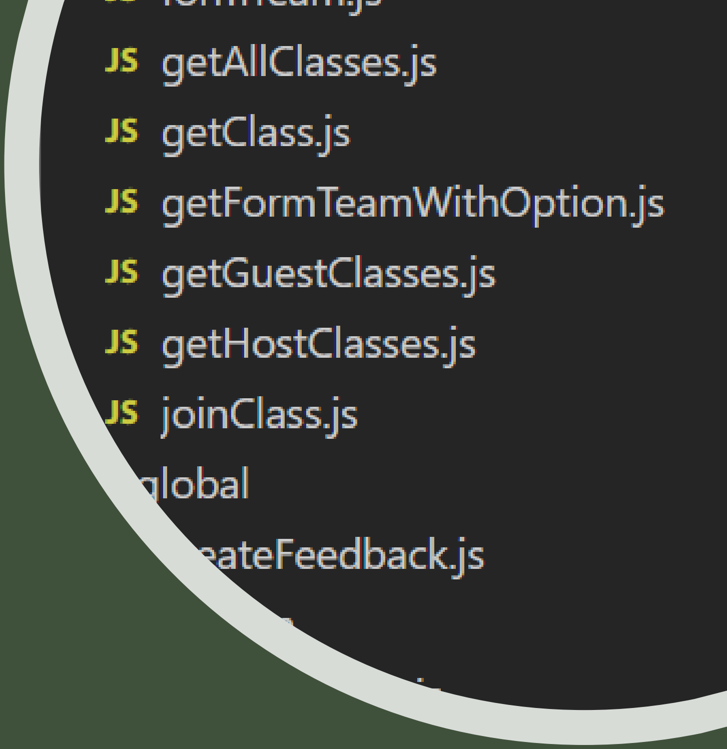
Final Design: Backend



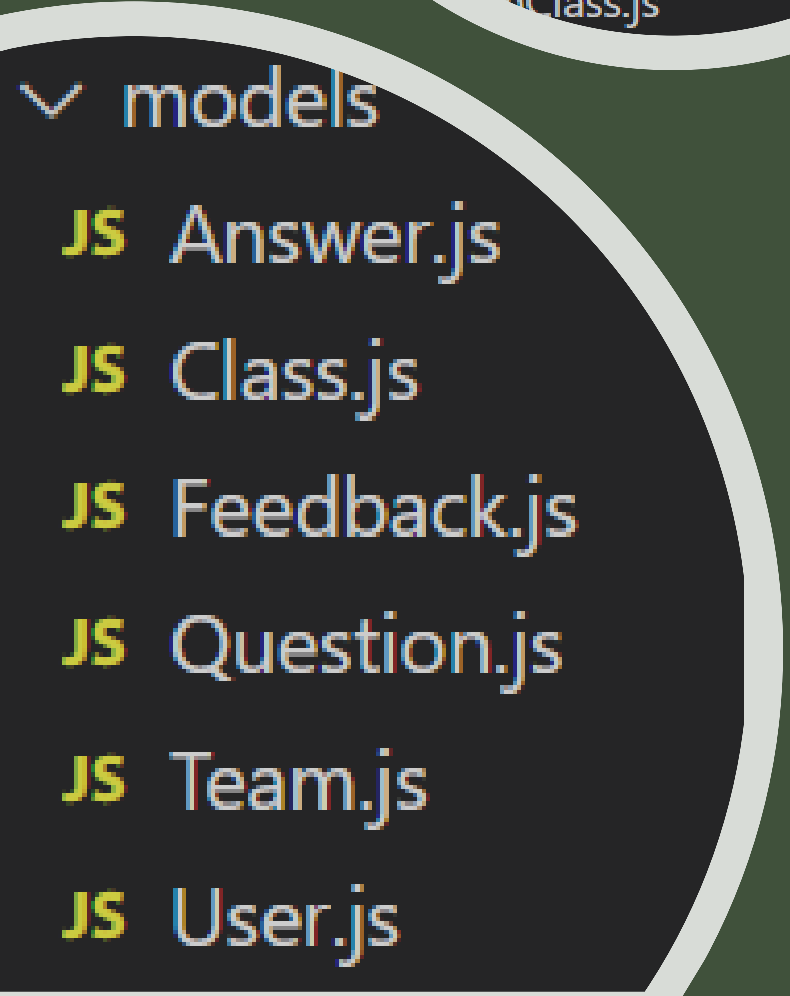
```
JS class.formTeam.js
JS class.getAllClasses.js
JS class.getClass.js
JS class.getFormTeamWithOption.js
   class.getGuestClasses.js
   ss.getHostClasses.js
   inClass.js
```



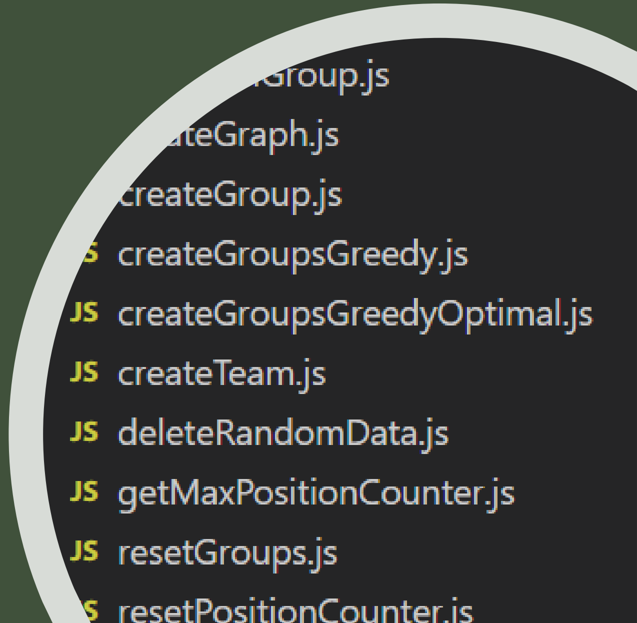
```
   ass.test.js
   createClass.test.js
JS createMultipleUsers.js
JS createUserWithoutVerify.js
JS formTeam.test.js
JS getClass.test.js
JS getTeam.test.js
   joinClass.test.js
   gin.test.js
   dition.js
```



```
JS formTeam.js
JS getAllClasses.js
JS getClass.js
JS getFormTeamWithOption.js
JS getGuestClasses.js
JS getHostClasses.js
JS joinClass.js
   global
   eateFeedback.js
```

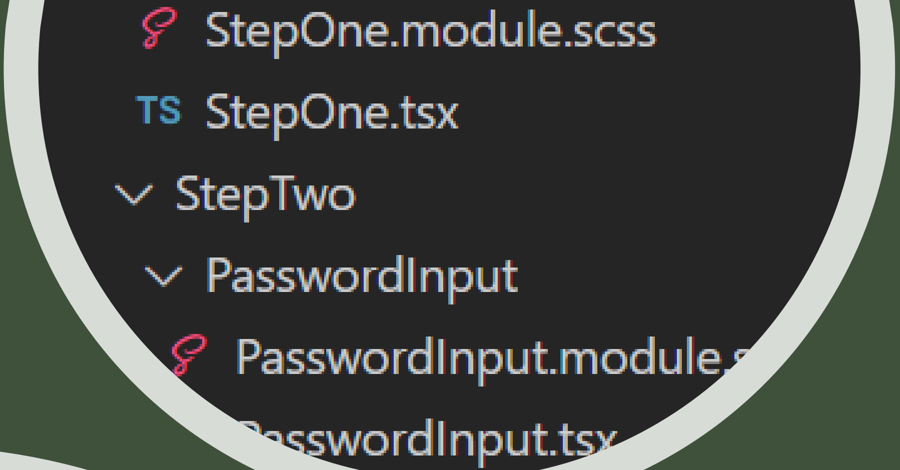


```
⌄ models
  JS Answer.js
  JS Class.js
  JS Feedback.js
  JS Question.js
  JS Team.js
  JS User.js
```

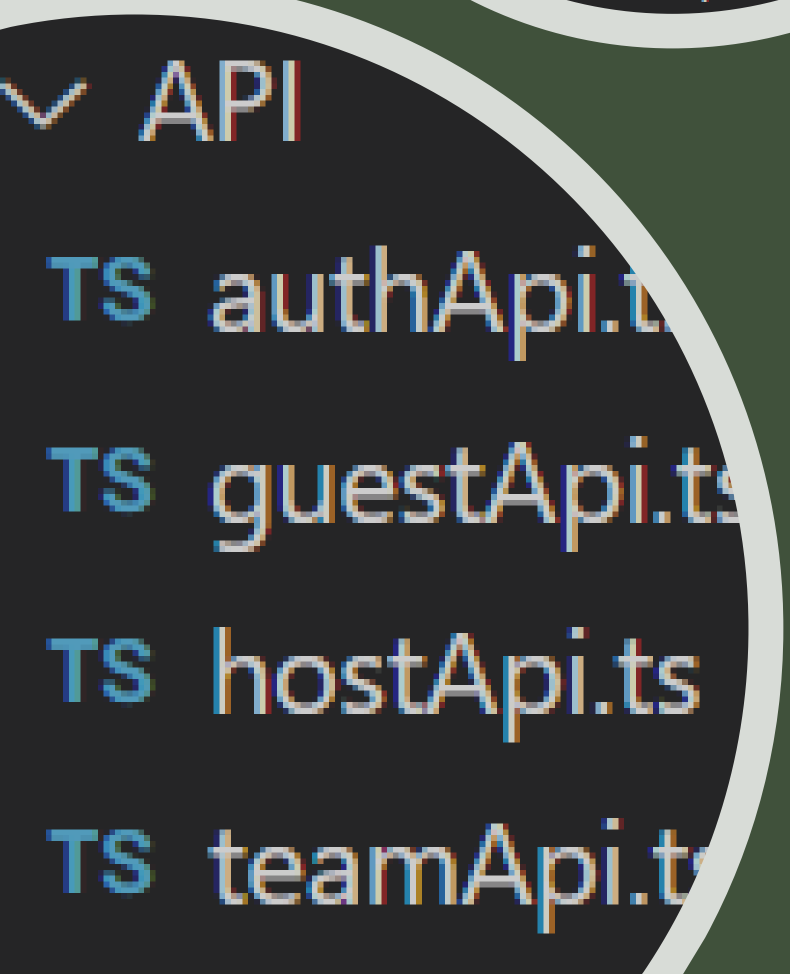


```
   Group.js
   teGraph.js
   createGroup.js
   createGroupsGreedy.js
JS createGroupsGreedyOptimal.js
JS createTeam.js
JS deleteRandomData.js
JS getMaxPositionCounter.js
JS resetGroups.js
   resetPositionCounter.js
```

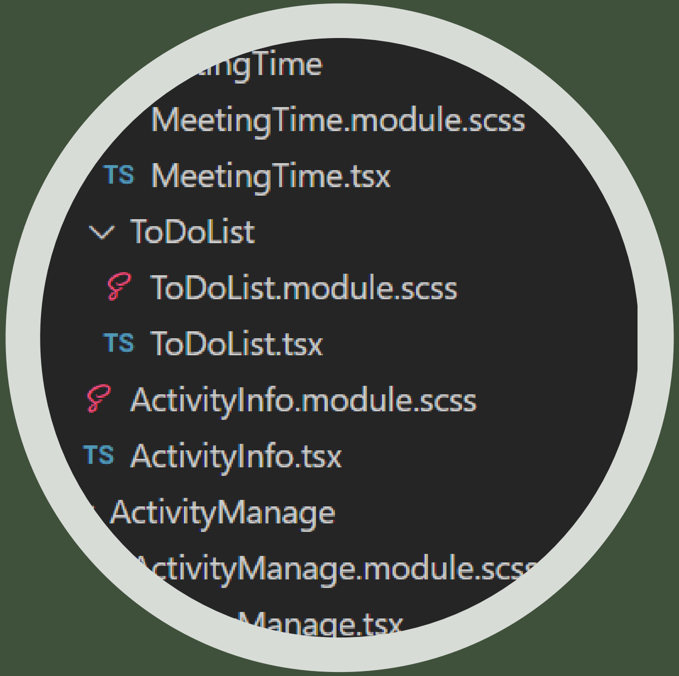
Final Design: Frontend



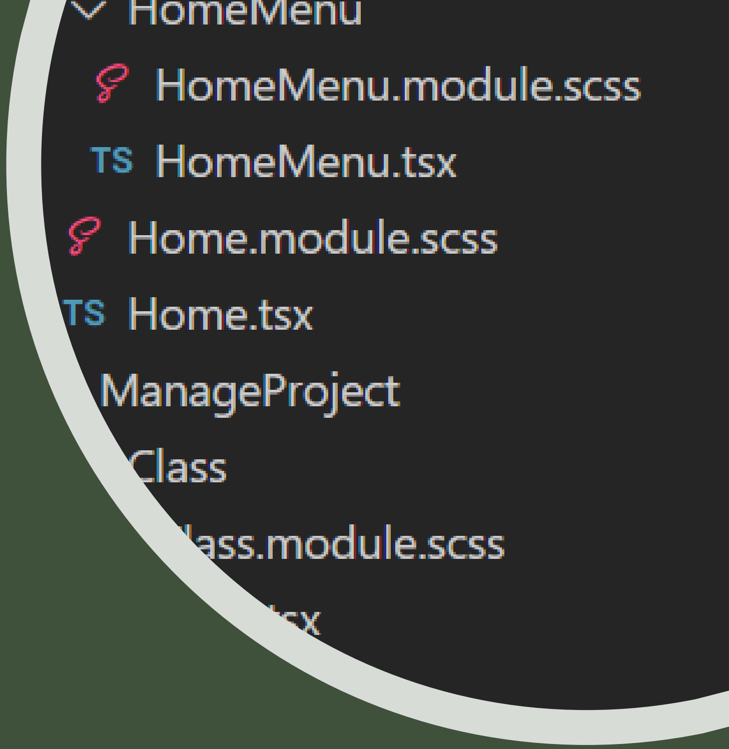
```
StepOne.module.scss
TS StepOne.tsx
StepTwo
PasswordInput
PasswordInput.module.s
asswordInput.tsx
```



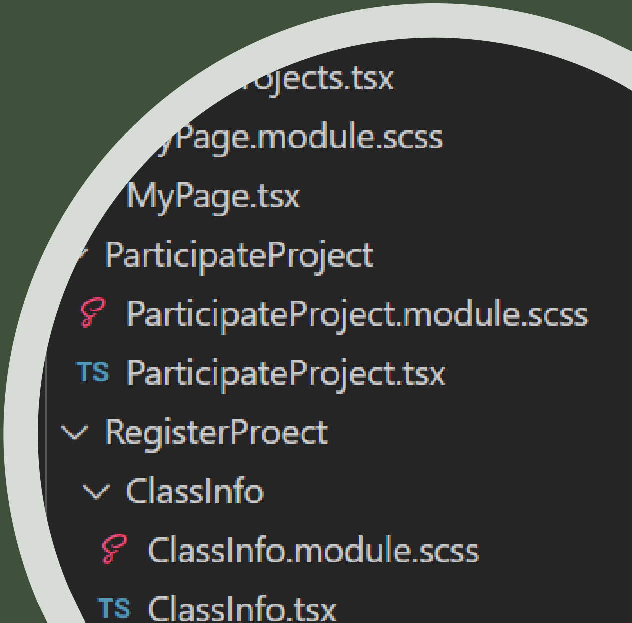
```
API
TS authApi.t
TS guestApi.ts
TS hostApi.ts
TS teamApi.t
```



```
ngTime
MeetingTime.module.scss
TS MeetingTime.tsx
ToDoList
ToDoList.module.scss
TS ToDoList.tsx
ActivityInfo.module.scss
TS ActivityInfo.tsx
ActivityManage
ctivityManage.module.scss
Manage.tsx
```



```
HomeMenu
HomeMenu.module.scss
TS HomeMenu.tsx
Home.module.scss
TS Home.tsx
ManageProject
Class
lass.module.scss
sx
```



```
ojects.tsx
yPage.module.scss
MyPage.tsx
ParticipateProject
ParticipateProject.module.scss
TS ParticipateProject.tsx
RegisterProect
ClassInfo
ClassInfo.module.scss
TS ClassInfo.tsx
```

Demo

Challenges

Backend

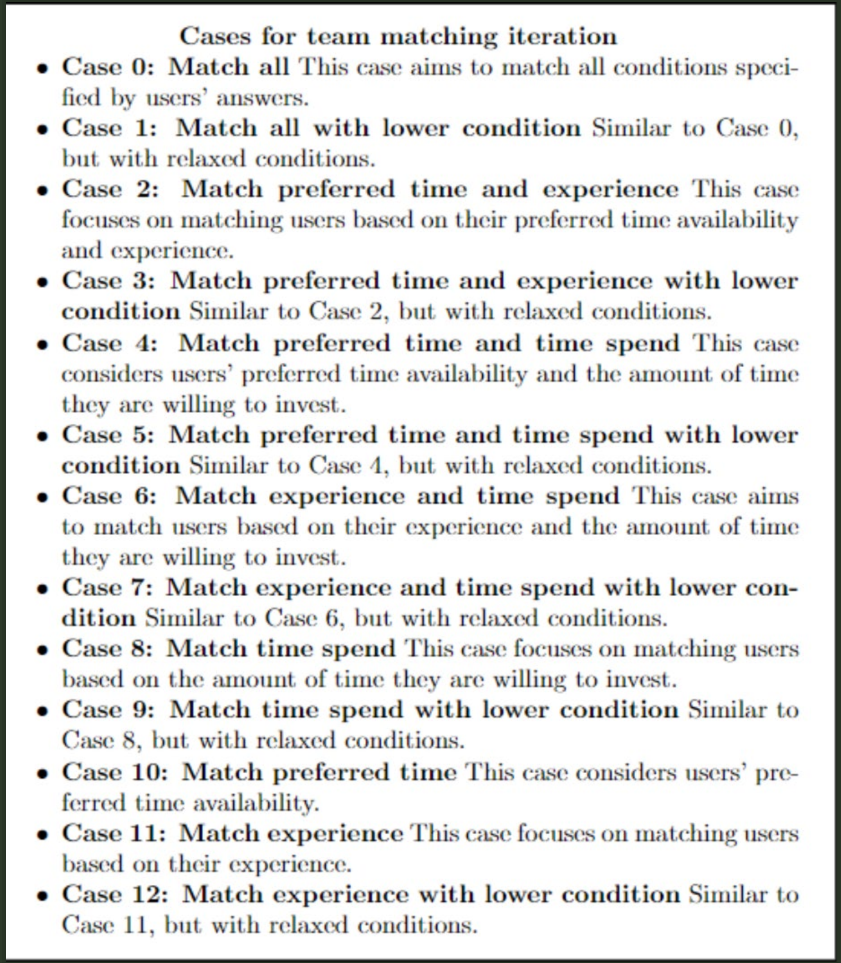
- Testing Environment
- Team forming Algorithm Implementation

Frontend

- Project Management
- Team forming Algorithm Implementation

Limitation

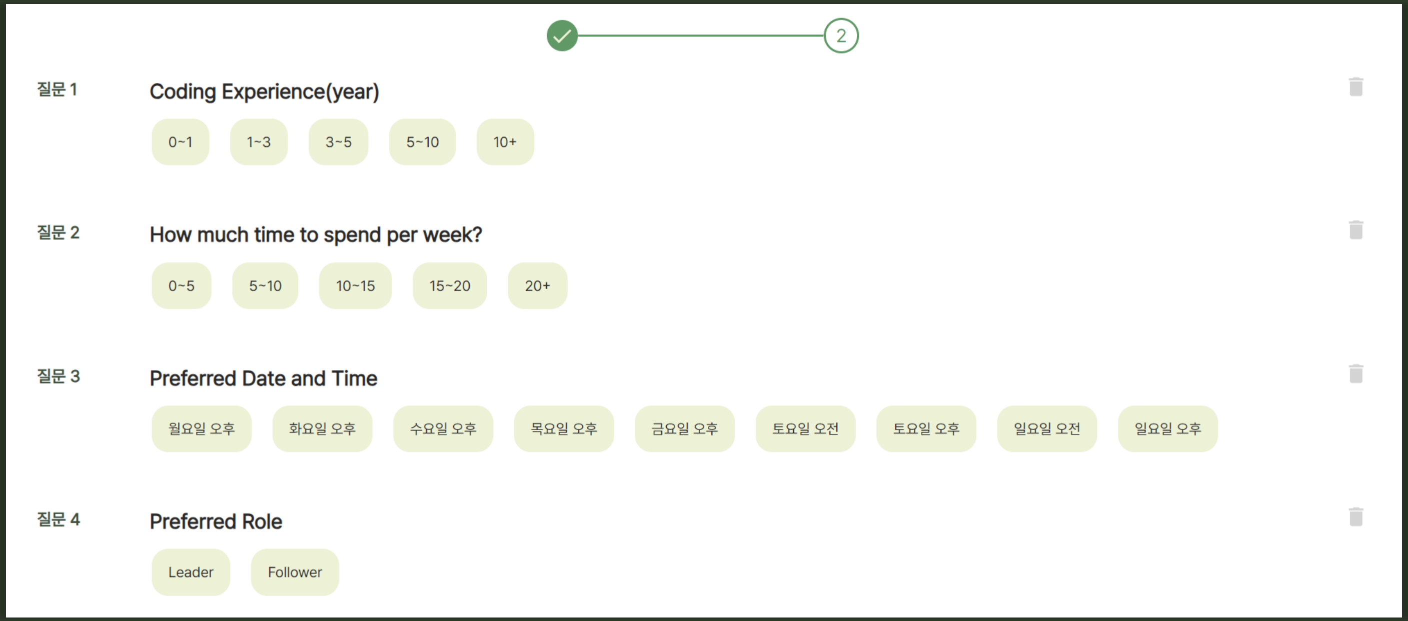
Extreme Cases



Cases for team matching iteration

- **Case 0: Match all** This case aims to match all conditions specified by users' answers.
- **Case 1: Match all with lower condition** Similar to Case 0, but with relaxed conditions.
- **Case 2: Match preferred time and experience** This case focuses on matching users based on their preferred time availability and experience.
- **Case 3: Match preferred time and experience with lower condition** Similar to Case 2, but with relaxed conditions.
- **Case 4: Match preferred time and time spend** This case considers users' preferred time availability and the amount of time they are willing to invest.
- **Case 5: Match preferred time and time spend with lower condition** Similar to Case 4, but with relaxed conditions.
- **Case 6: Match experience and time spend** This case aims to match users based on their experience and the amount of time they are willing to invest.
- **Case 7: Match experience and time spend with lower condition** Similar to Case 6, but with relaxed conditions.
- **Case 8: Match time spend** This case focuses on matching users based on the amount of time they are willing to invest.
- **Case 9: Match time spend with lower condition** Similar to Case 8, but with relaxed conditions.
- **Case 10: Match preferred time** This case considers users' preferred time availability.
- **Case 11: Match experience** This case focuses on matching users based on their experience.
- **Case 12: Match experience with lower condition** Similar to Case 11, but with relaxed conditions.

Predefined Questions



질문 1 Coding Experience(year)

0~1 1~3 3~5 5~10 10+

질문 2 How much time to spend per week?

0~5 5~10 10~15 15~20 20+

질문 3 Preferred Date and Time

월요일 오후 화요일 오후 수요일 오후 목요일 오후 금요일 오후 토요일 오전 토요일 오후 일요일 오전 일요일 오후

질문 4 Preferred Role

Leader Follower

Evaluations

- Effectiveness of our algorithm
- Error-free client server interaction
- Complex algorithm but easy UX
- Clear matching result feedback

Thank you