OCTOBER

M	T	W	T	F	S	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15 Scaffold code: Design Database: front end routing logic:	16	17	18	19	20	21 weekly progress meeting
22 Build homepage: build song list	23	24	25	26	27	28 weekly progress meeting
29 Websocket RPC system: voting database:	30	31	1	2	3	4 weekly progress meeting
5	6	7	8	9	10	11

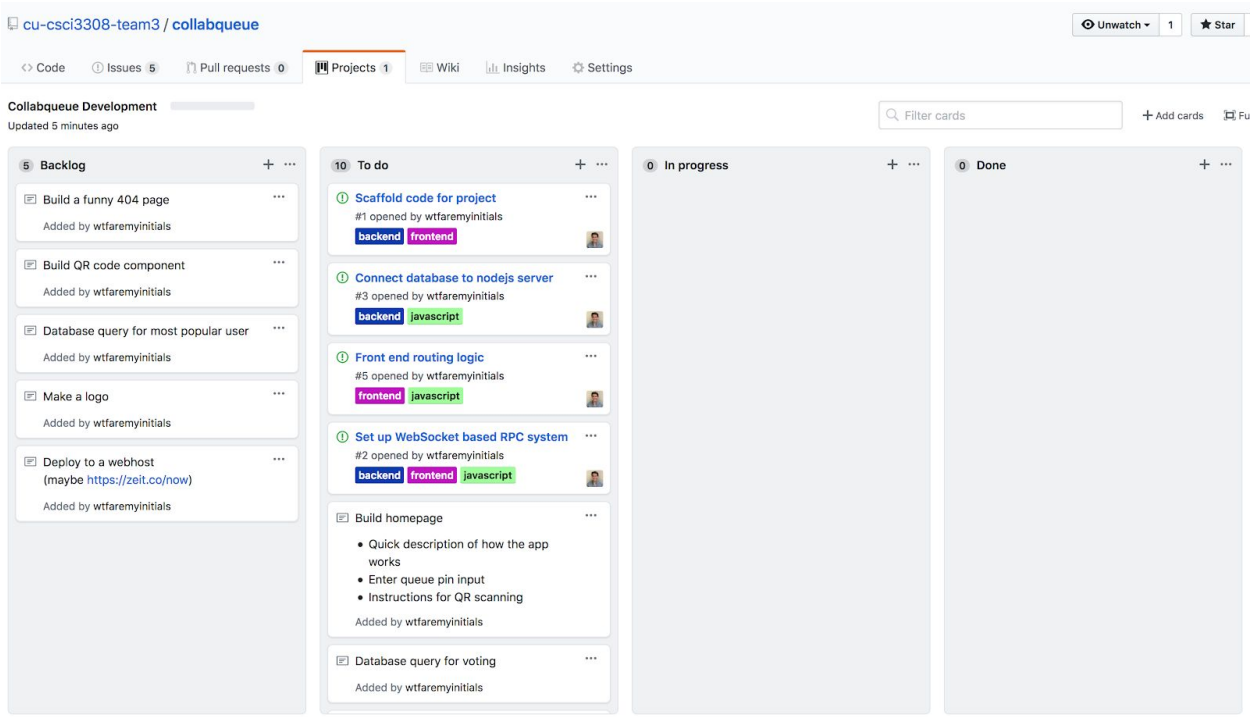
NOVEMBER

M	T	W	T	F	S	S
29 Websocket RPC system: voting database:	30	31	1	2	3	4 weekly progress meeting
5 Build master host page:	6	7	8	9	10	11 weekly progress meeting
12 connect database to NodeJS server	13	14	15	16	17 Thanksgiving break	18
19 Thanksgiving break	20	21	22	23	24	25
26 Current standing meeting	27 Deploy to webhost	28	29	30	1	2 weekly progress meeting
3	4	5	6	7	8	9

DECEMBER

M	T	W	T	F	S	S
26 Current standing meeting	27 Deploy to webhost	28	29	30	1	2 weekly progress meeting
3 Polish final project and finsih all debugging	4	5	6	7 Prepare and pactice presentation	8	9
10 Presentations	11	12	13	14	15	16
17	18	19	20	21	22	23

- Project Management software product
    - Github
        - We will use the Projects tab in our Collabqueue repository on Github
        - https://github.com/cu-csci3308-team3/collabqueue/projects/1



- Requirements
    - Functional
        - Users can vote on the next song to play
        - Music can come from YouTube and SoundCloud
        - No user accounts required to use the service
        - Software will generate random X-digit (X being a yet-undecided number) long code that everyone can use
        - QR code scanning to open a queue
        - Ability to skip songs either by voting or creator permissions
        - (not crucial) an little crown icon next to the user who has requested the most played songs during 1 session
    - Non-functional
        - UI looks not terrible
        - A clever domain name

- Project plan
    - One sprint weekly (6 sprints total)
    - Sprint 1

        - Scaffold code for project (Ryan N, Ryan G, Naji)
        - Design database schema (Will, Alex, Tyler)
        - Front end routing logic
    - Sprint 2
        - Build homepage (Ryan N, Ryan G, Naji)
        - Build song list component (Will, Alex, Tyler)
    - Sprint 3
        - Set up websocket based RPC system (Will, Alex, Tyler)
        - Database query for voting (Ryan N, Ryan G, Naji)
    - Sprint 4
        - Build master host page (Will, Alex, Tyler, Naji, Ryan N, Ryan G)
    - Sprint 5
        - Connect database to NodeJS server (Will, Alex, Tyler, Naji, Ryan N, Ryan G)
    - Sprint 6
        - Deploy to a web host (Will, Alex, Tyler, Naji, Ryan N, Ryan G)

- Agile standup
    - Completed since last meeting
        - Further brainstorming
        - Planned functionality requirements
    - Complete before next meeting
        - Front end routing logic
        - Build master/player page
        - Connect database to nodejs server
        - Set up websocket-based rpc system
        - Scaffold code for project
    - Obstacles/roadblocks
        - Needing to learn JavaScript
        - Tough to meet with entire group at once in order to properly delegate tasks
- Retrospective meeting
    - What went well
        - Creativity in functionality requirement ideas
    - What didn't go well
        - Mid-week communication
    - What should be improved upon
        - Consistent communication throughout the week
        - Finding a good time to allow meeting with the entire group