Developer Log

This log has been created as a screenshot of all blog entries in JetBrains Space, as this was the format which we chose the publish what work we have done.

Harry Wang, 4 Feb 17:19 · 1 min read

Subscribe

Server

Guys I've setup the server, U can link the server by using flowing info

- ***IP address***: 192.248.164.28
- ***Username:*** root
- ***Password:*** Xy2{Z3%2L7f,wvhh

If you want use the database visualization, you can visit >>>link>>>

- ***DB Username:*** root
- ***DB Password:*** grouph

If you want to do updates on the web page, please upload your files to `/data/wwwroot`

If you want to visit the website, you can go pjt.hanxiaofei.tk

```
Nginx install dir:              /usr/local/nginx

Database install dir:           /usr/local/mysql
Database data dir:              /data/mysql
Database user:                  root
Database password:              grouph

PHP install dir:                /usr/local/php
Opcache Control Panel URL:      http://192.248.164.28/ocp.php

phpMyAdmin dir:                 /data/wwwroot/default/phpMyAdmin
phpMyAdmin Control Panel URL:   http://192.248.164.28/phpMyAdmin

Index URL:                      http://192.248.164.28/
```

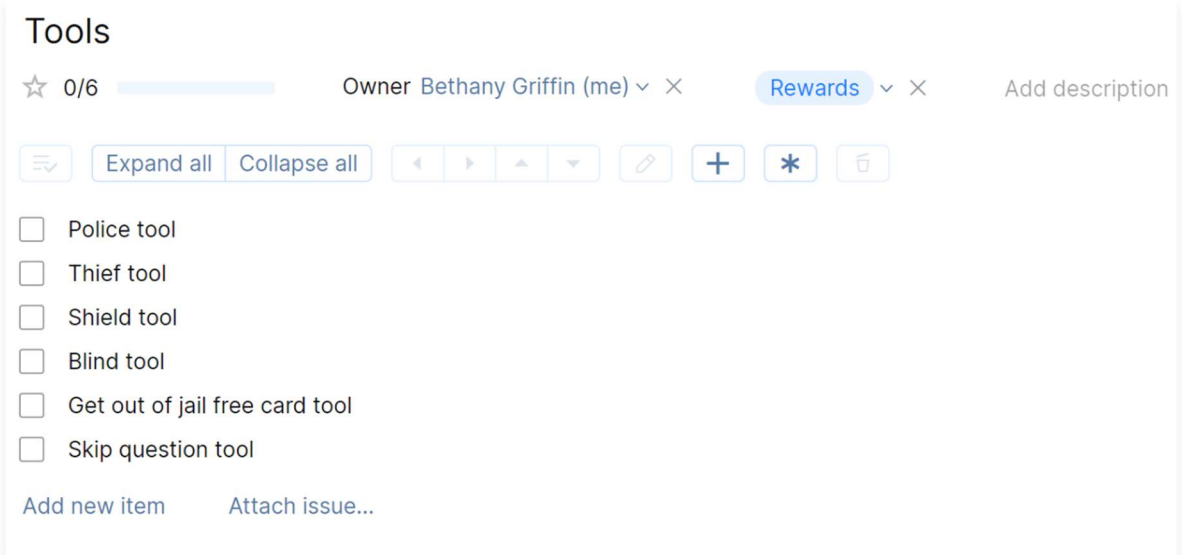
Bethany Griffin, 4 Feb 18:29 · 1 min read

Backlog Population

I've added a good portion of the basic requirements from the 2/2/21 meeting notes to the backlog.

Design Webpage	Webpage	unassigned	no due date
Create Card	Rewards	unassigned	no due date
Create Accessory Mods	Avatar Rewards	Annabelle Mo...	no due date
Create Avatar Accessories	Avatar Rewards	Annabelle Mo...	no due date
Create Avatar	Profile Avatar	Annabelle Mo...	no due date
Welfare Quiz	Quizzes	unassigned	no due date
Module Chat Room	Forum	unassigned	no due date
Course Chat Room	Forum	unassigned	no due date
Fill out profile	Profile	unassigned	no due date
Create Basic Profile	Profile	unassigned	no due date
Teams Leaderboard	Leaderboards	unassigned	no due date
College Leaderboard	Leaderboards	unassigned	no due date
Personal Leaderboard	Leaderboards	unassigned	no due date
Admin User	Users	unassigned	no due date
Staff User	Users	unassigned	no due date
Student User	Users	unassigned	no due date
Welfare User	Users	unassigned	no due date

I've also created and populated some checklists for specific areas of our requirements.


Tools
0/6
Owner Bethany Griffin (me)
Rewards
Add description
Expand all
Collapse all
Police tool
Thief tool
Shield tool
Blind tool
Get out of jail free card tool
Skip question tool
Add new item
Attach issue...


Avatar
0/3
Owner Annabelle Moore
Avatar
Add description
Expand all
Collapse all
Create Accessory Mods
Annabelle Moore
Create Avatar Accessories
Annabelle Moore
Create Avatar
Annabelle Moore
Add new item
Attach issue...

Cheng Zhu, 4 Feb 20:58 · 3 min read

(Raw) Data Flow between frontend and backend code

`Read from = frontend webpages will need this data from the database`

`Serve to = frontend webpages will serve this data to the database`

1. Login / Register Form (Using university cridentials?)

· (Read from) user name / password

2. Dashboard?

3. Profile page

· (Read from) College / Department / Modules

· (Read from) First / last name

· (Read from) Nickname

· (Read from) Currency / Resources (Ingredients etc.)

· (Read from) Milestone / (Achievement)

· (Read from) Avatar (frame / background)

· (Read from) (Link to articles)

· (Read from) Title, like “no.1 in math”

· (Read from) College / Courses

· (Read from) Personal Blog?

· Everyday quiz result (in database, only staff can access)

4. Chat Room

· (Personal) / Module / College chat room

5. Guest form

· (Serve to) name / related user / relation / authentication question (and answer)

6. Quiz page

· (Read from) questions

· (Serve to) answers

7. Marking page

· (Read from) answers

· (Serve to) marks

8. Quiz management page

· (Serve to) operation messages to quizzes

9. Shop page

· (Read from) Currency / Resources

· (Serve to) New items

· (Serve to) Update currency / resources

10. Trade page

· (Read from) current resources of both users

· (Serve to) updates to resources of both users

11. Team List Page

· (Read from) current teams

· (Serve to) Create new team

12. New Team Page

· (Serve to) Team name

· (Serve to) Team manager ID

· (Serve to) Team ID

13. Current team page

· (Read from) Does the user belong to a team?

· (Read from) Current team states

· (Read from) Is the user a manager? (to have operations)

14. Team operation page

· (Serve to) Team operation messages

15. College forum page

· (Read from) current status (Points, members etc.)

· (Read from) Lists of study notes / posts

· (Serve to) New Post

16. Study note(Post) page

· (Read from) content

· (Read from) comments

· (Serve to) leave new comments

· (Read from) Is the user the author of this post?

17. New Study Note(Post) page

· (Serve to) Post Title

· (Serve to) Post Content

· (Serve to) Author ID

18. Study Note(Post) management page

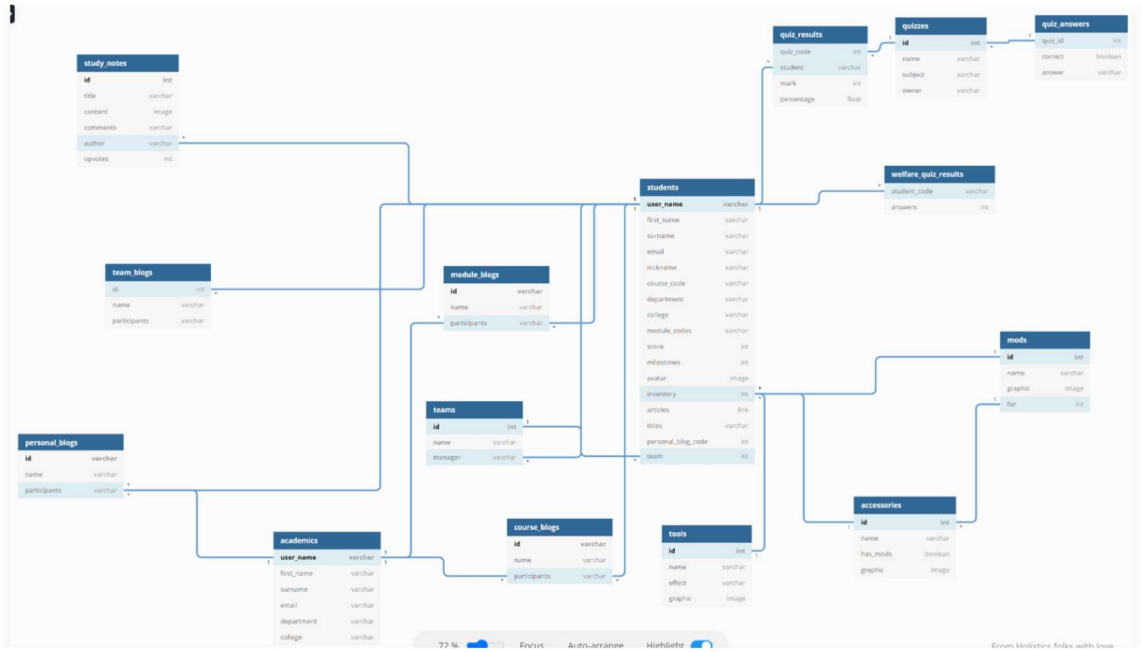
· (Serve to) management messages

Bethany Griffin, 4 Feb 22:02 · 1 min read

Database UML

I have created the following UML diagram for all the database tables we are planning to use.

At the moment it is just that, a plan. We can change it anytime we want, however I want to get it as right as possible so we can present it to Matt as part of our documentation (not sure if it will be under technical, process or product).

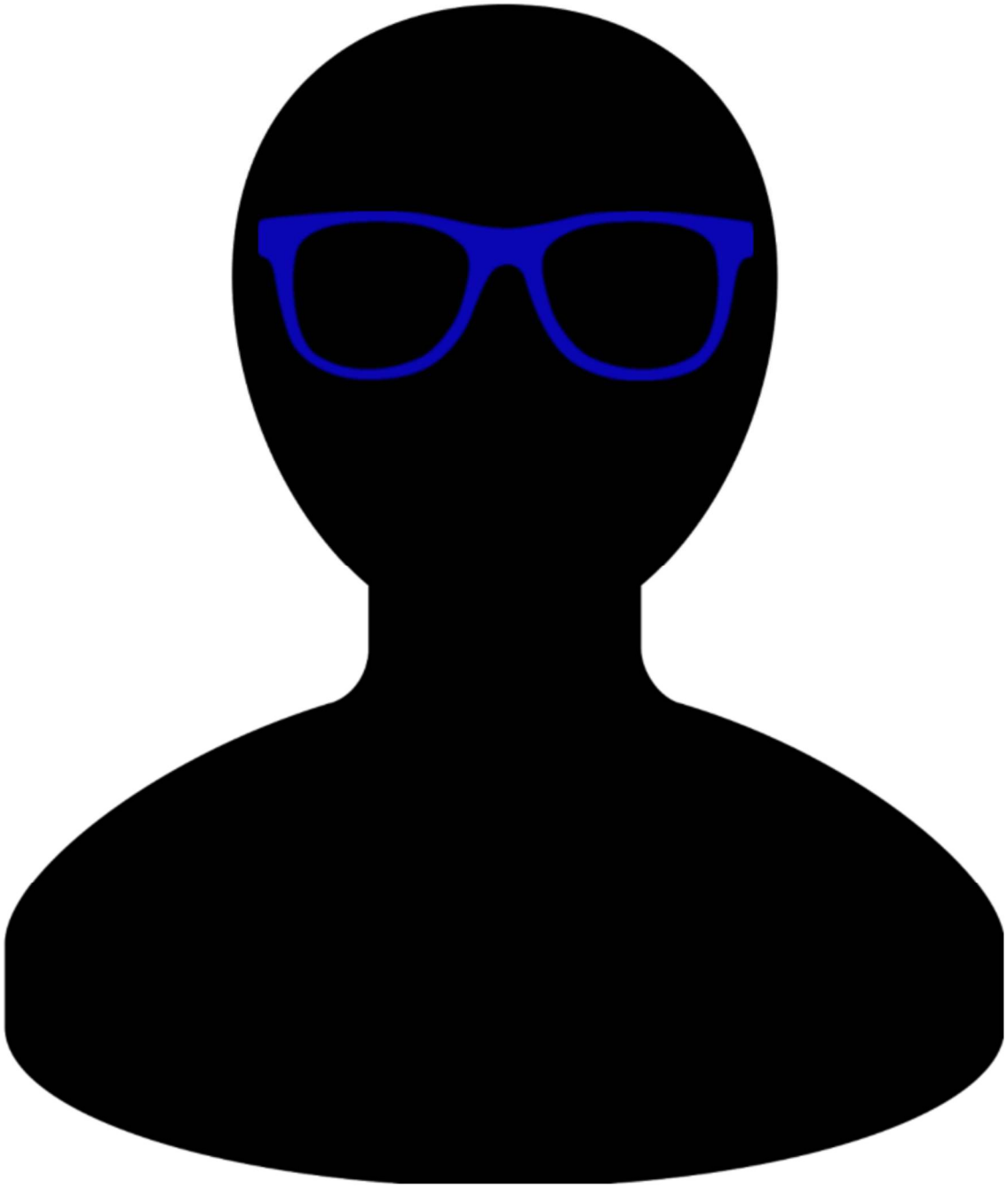


If you feel there is anything that should be changed/added then feel free to let me know!

Annabelle Moore, 4 Feb 23:12 · 1 min read

Started avatar design

I have started the avatar designs, so far I have done 13 colours each for 2 different types of glasses. I have made all layers of the same size so that the avatars are easy to assemble. For example:



Georgina Huang, 5 Feb 12:33 · 1 min read

Card pool design

Just forgot to post things here.

1. Personal quizzes are uploaded by lecturers and TAs so the users can do them and they will get credits when they do them, also get credits when they achieve higher than 60%
2. Group quizzes are uploaded by lecturers and TAs so the users could find groupmates to do them, they will get points when they do them, also because this is limited time so the quickest person that get the right answer for one question can get the highest points for example the first one get 5 points second one get 4 points and so on. All question points count together at the end of the quiz, the highest points of the person get extra winner credits.
3. Certain amount of credits that they gain can exchange keys. Then these keys can use in the card pool to draw a random card which contains tool cards such as steel points card, pass questions card, double credits cards, blind the others screen cards etc. and things like profile picture costumes such as fancy frames for profile photos, glasses for profile photos etc.
4. Ranking board:The highest credits you get the highest ranking you are. Top 100 in the ranking will also be shown in the profile. eg. No.23 in mathematics.
5. Chat room. All users are able to chat with other users as long as you search their name or user names. So students are able to leave message to lecturers to ask questions and students can make friends to each other.
6. Forum: all user can post blogs there. So others could comment on them.
7. ...feel free to add on :)

Georgina Huang, 5 Feb 12:35 · 1 min read

Website page design

Looking at html and how to design website page.

Cheng Zhu, 5 Feb 16:00 · 1 min read

Suggestions to the UML

Hi, I think the UML looks good generally, and I have listed my suggestions below. Feel free to have comments or correct me.

1. is that possible to write "foreign key" tags to mark feoreign keys? (best to mark which attribute it points to)
2. image → blob
3. one user table (add tags to staffs to prevent they get inappropriate data; staffs have empty columns)
4. "study notes: content" is image?
5. team / personal / module / course blog could have content?
6. "Student: Inventory", "welfare_quiz_results: answers" could be another table (when attempt to store a list it's better to seperate the list into another table
7. it's probably better to leave quizzes as an independant table (since it's not necesarrily related to a student), where student can have a foreign key refers to an answer(result will be stored in this table as well), and the answer has a foreign key refers to a quiz. Similarly I think it's probably better to rename "welfare_quiz_result" to "welfare_quiz_answer"
8. same to 2, we can have one "inventory" table, leave "accessories" columns of rows with "tools" tag empty
9. to record what mods an accessory has, we can have a new table which only records keys in "mods" table. For example if accessory A1 has mods M1 and M2, we can have:

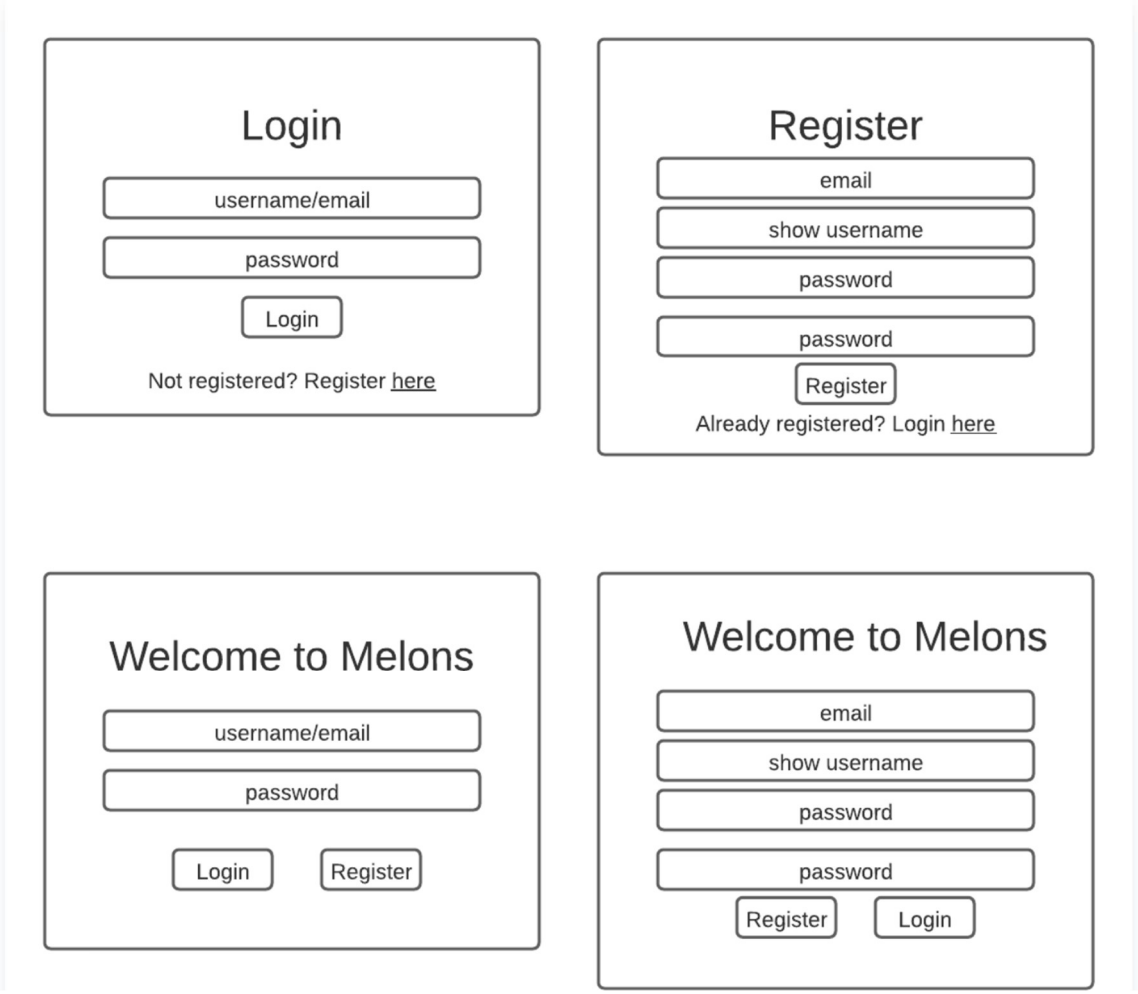
id	accessories(foreign key to accessories)	mods(foreign key to mods)
1	A1	M1
2	A1	M2

10. "students" table should have a column "resource" to record the current ingredients/currency.
11. can staffs have their accessories as well (they might be rewarded by posting quizzes/ marking quizzes/ helping students?) but can't have tools?
12. probably need a shop table to record currently available commodities?
13. Similar to 8, maybe let team table to bind with team blogs table?
14. maybe have independent college table and bind it with the college blog table?

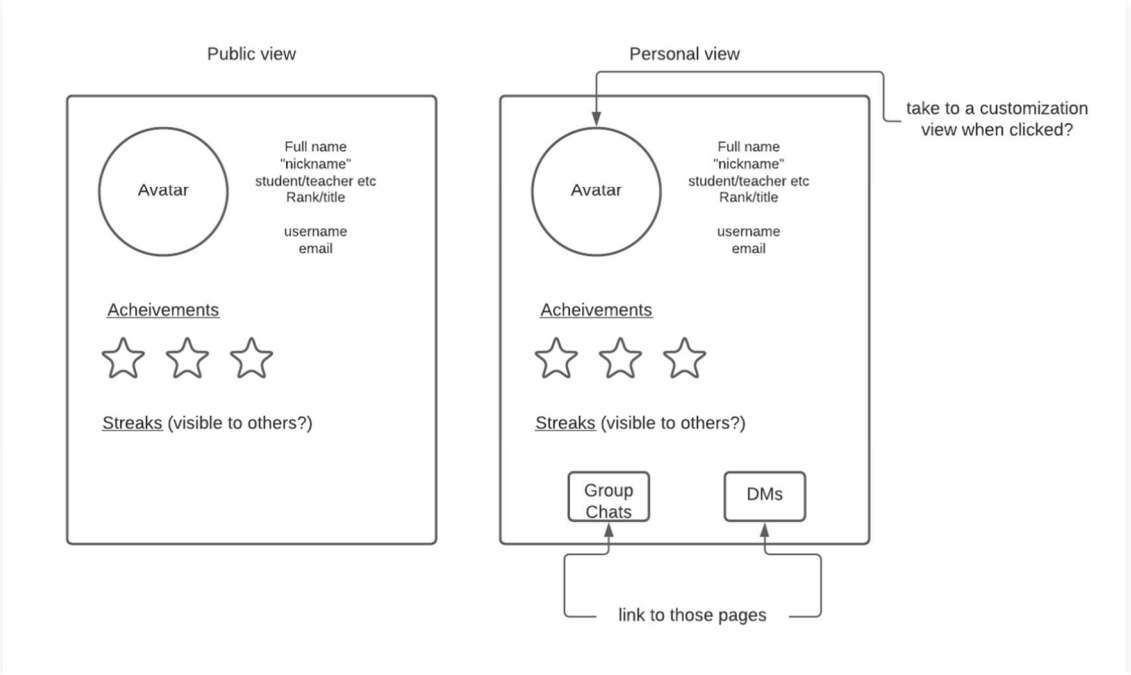
DEV TEAM-H Clair Newton, 5 Feb 17:03 · 1 min read

made a couple webpage layout ideas

Login/Registration page:



Profile page:



Cheng Zhu, 5 Feb 22:43 · 1 min read

Connect to the db via Django

The database we have would automatically reject any connection requests from outside by default, so I changed the 'Host Name' column from '127.0.0.1' to '%', and restarted the service, so that the database server will accept external connection requests. You can find this column in 'mysql' database, 'user' table.

According to the documentation the db configuration of a django project is controlled by settings.py, so we only need to change the 'DATABASES' dictionary as follows:

```
DATABASES = {
    'default': {
        'ENGINE': 'django.db.backends.mysql',
        'NAME': 'project',
        'USER': 'root',
        'PASSWORD': 'grouph',
        'HOST': '192.248.164.28',
        'PORT': '3306',

    }
}
```

Annabelle Moore, 5 Feb 23:57 · 1 min read

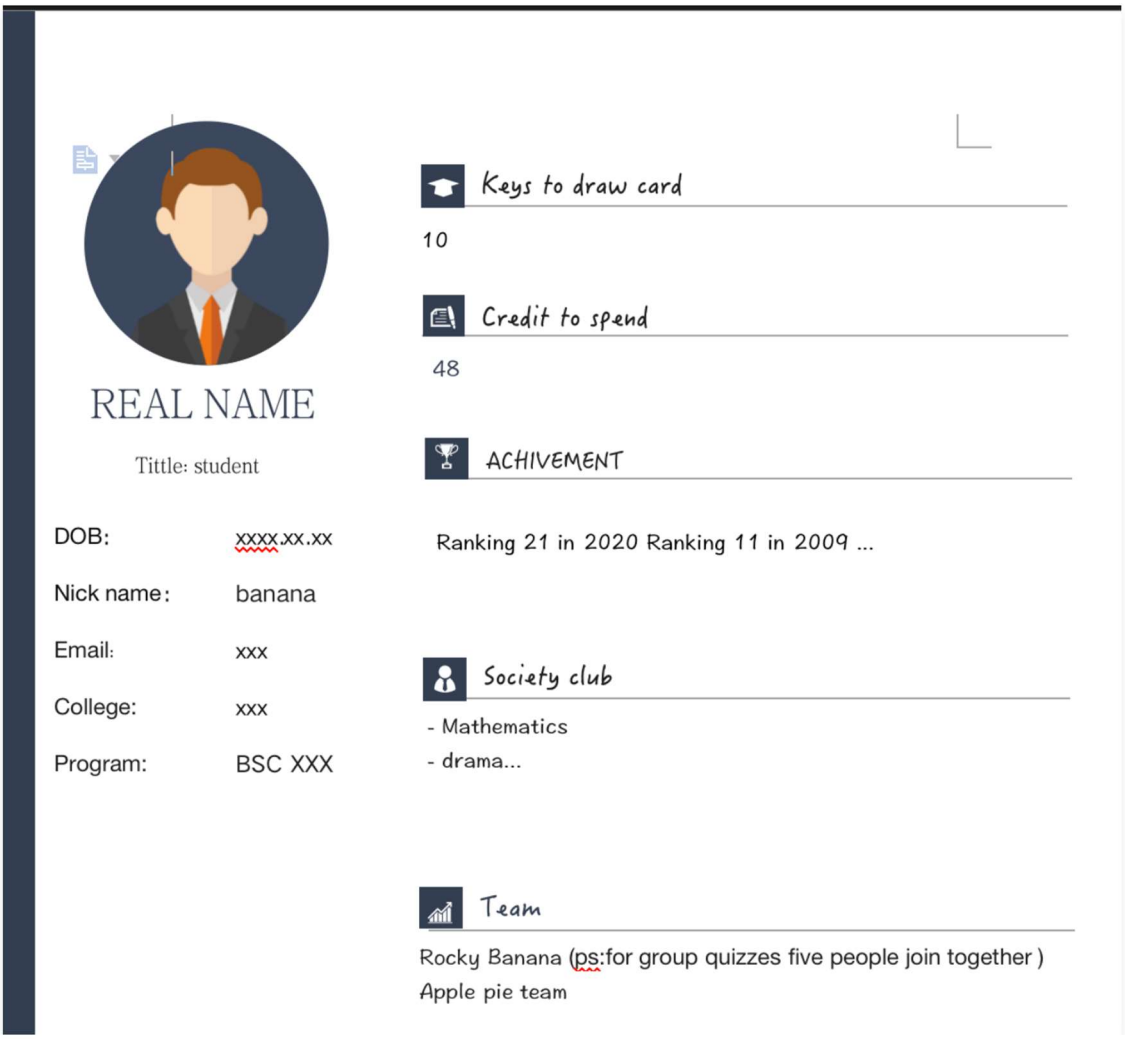
Subscribe

Hats

I made 9 or 10 different colours for the cowboy hat and top hat.

Georgina Huang, 6 Feb 14:36 · 1 min read

profile page?

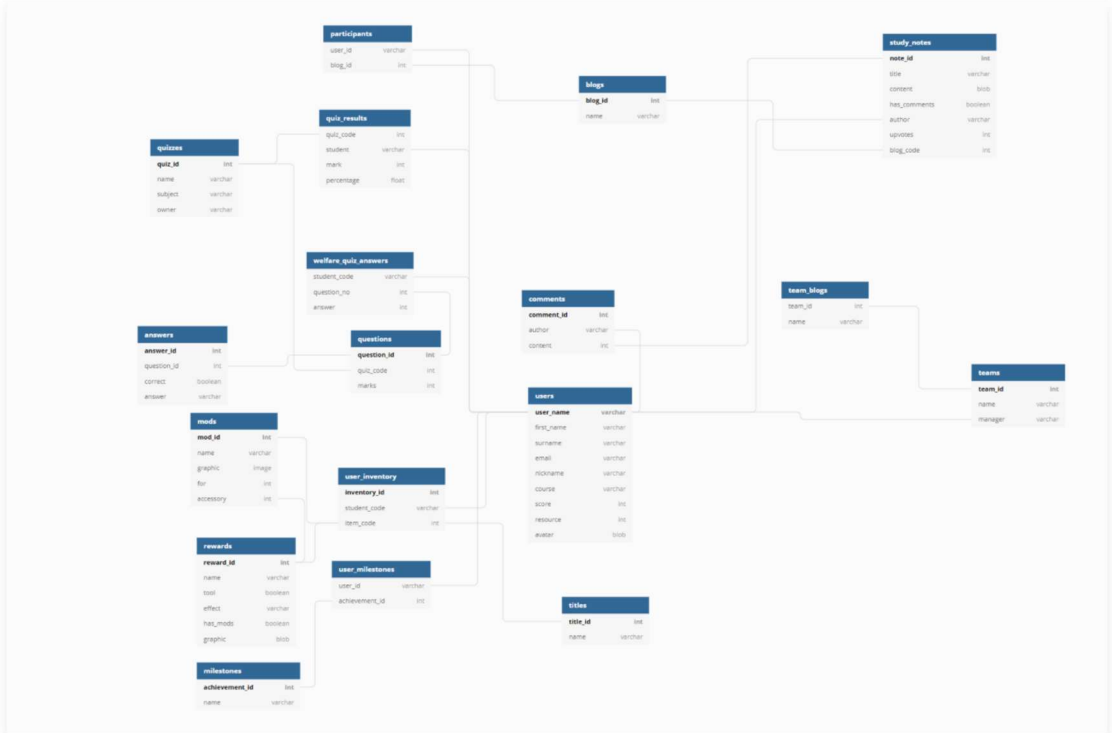


Bethany Griffin, 6 Feb 16:06 · 1 min read

Final UML?

Hopefully this is the final UML diagram!

I believe I have removed all unnecessary data and redundancies. I have also added link tables to ensure that no entry stores a list.



Again, feel free to suggest any corrections or mistakes.

Next I'm going to try to implement these tables into the DB.

Bethany Griffin, 6 Feb 17:18 · 1 min read

Repository Changes

Hey guys,

I've set up a GitHub repository for us to use instead of the space repository. This is for a variety of issues but the key one is that space doesn't allow for folders (which are key).

If you could all reply with your GitHub usernames or emails then I can add you guys as collaborators!

Cheng Zhu, 6 Feb 17:58 · 1 min read

Github Repository Updated

I have uploaded the folder of the project to the "main" branch(I tried to create a new branch and build a pull request, but due to unknown reason I can only push changes to the main branch from my local repository)

Also I have created another new branch which contains the user model. Feel free to check if anything is wrong.

Cheng Zhu, 6 Feb 21:11 · 1 min read

New Github Branch for all models

I created a new branch called "First-Draft-Models" which contains all models I've created so far. I think it covers all models we've discussed before, but since it costed me several hours I can't ensure it's perfect, so if anyone could help me to have a check of it I will be very thankful.

By the way I'm not very clear about the difference between a mod and an accessory, since it will vary the structure of models, could anyone tell me the that?

Bethany Griffin, 7 Feb 14:49 · 2 min read

Models Code Check

All of the code that has been created is great! This is a just my take it.

These are just some suggestions and questions I have. They may not be 'right', but feel free to correct me if I've gotten something wrong.

1. I believe that the use of auto_now and auto_now_add is correct.
2. Blogs will likely have multiple participants, but they can still have an 'author' (i.e. the person who created it if it's a user created blog).
3. Instead of binding stuff to a course do we want to bind it to a module? - Many courses have optional modules that people can choose so people won't be in all course modules
4. Do we potentially want to store all titles? Or will they be hard coded?
5. The only thing with having stuff hard coded is that it makes it more difficult for to add/change stuff in the future.
6. has_mods may be required if we want to allow the user to select an accessory and it show what mods it has - it seems easier on the code to just check this Boolean rather than searching the entire mod list to find if there are any or not, but that might just be me.
7. Do we want want IDs for each of the non-link tables? Or are those added my Django by default? (Like do we need to add primary keys for all the stuff that is being linked by a foreign key)
8. Also, an accessory can have multiple mods so do we maybe want to create a link table that links mods to accessories and vice versa?

Now as for things you want me to do:

- Do you want me to start filling out the choices for the effects of tools?

Anything else you want me to do, or any help you need, just let me know!

Bethany Griffin, 7 Feb 17:00 · 1 min read

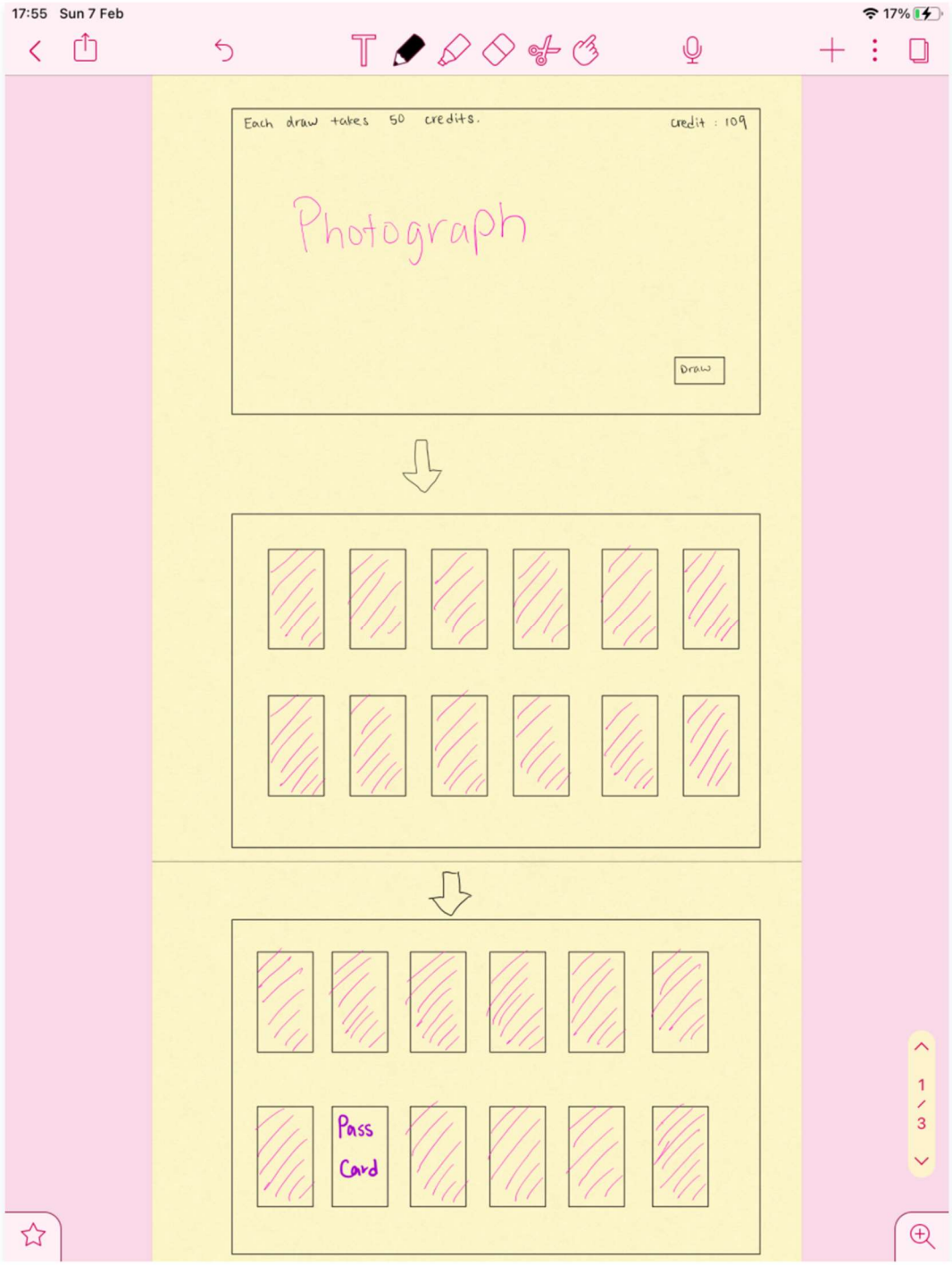
Loot Tier Allocation Code

I have began creating the code responsible for allocating the user a tier of loot based on different probabilities for each tier.

This can be found under the loot-algorithm branch of GitHub.

Georgina Huang, 7 Feb 17:59 · 1 min read

draw card webpage look like?



Georgina Huang, 8 Feb 01:58 · 1 min read

Subscribe

fill out profile webpage

I have done the profile webpage and uploaded to github under the branch called fill out profile.

Cheng Zhu, 8 Feb 02:25 · 1 min read

Subscribe

Upload Bootstrap Library

Uploaded bootstrap library to github repository, which is stored in "study_platform/static" directory; created another directory "templates" for later uploads.

Clair Newton, 8 Feb 08:54 · 1 min read

Subscribe

added login, register, and profile pages

I uploaded login, register, and 'profile pages'.

They currently link to one another but don't do anything beyond that.

The spacing and sizing is still a work in progress.

I also changed the folder name within the static folder from study_platform to boostrap because study_platform was also the name of another folder

---

Cheng Zhu, 8 Feb 20:18 · 2 min read

Core mechanism change plan

Based on the client meeting today, I think the main problem of out current plan is that we don't have a clear clue about what do we need to achieve "gamification", and trying to add everything we could think of to approach gamification, which makes the whole system have no focus (and hard to implement).

In the previuos plan we only have the reward mechanism (score, rank, loot) and leave the "game" part as answering quizzes / posting study notes, which I think is not efficient to enhance passion of study, as the only motivation for people to play is the rewards, causing people get bored easily.

By definition, gamification means we apply (not add) game mechanisms to the study behaviors, which I think includes a target, rules and interactions. Based on that, I think a good way to approach gamification is we combine an existing game mechanism with out current system - this means we don't have to abandon our previous work, but to abandon unnecessary feature plans / add new feature plans. Me and Harry had a discussion just now about some possible ideas, and found that card game is basically doable, if anyone else has other ideas I'm more than happy to hear that.

Here is the basic idea:

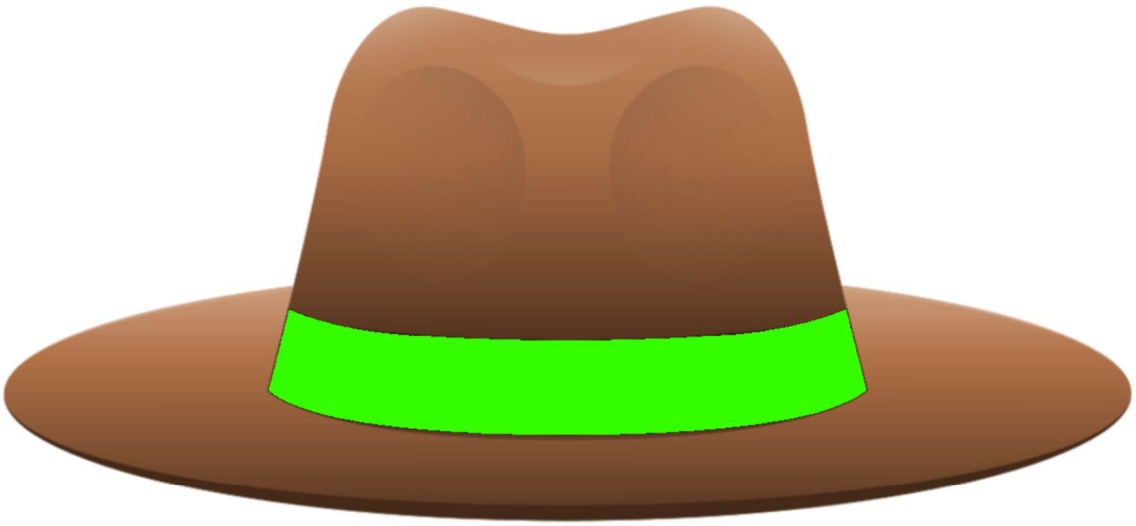
1. The basic game process is the same as other card games, a player wins if he or she manage to decrease the opponent's HP to 0
2. The difference is, in order to put a card to the table, a user must answer the corresponding question bind to that card , the answer of a card is set when created
3. Since the module leaders are too busy to design questions, we let any registered student user design a card (Probably require some resource). A card is allocated random resource points when created, the student can spend the corresponding resource points to purchase effects; he or she can write any question / answer to that card and the card will then be sent to the staff users
4. The staff users are expected to validate if the questions and the answers match. When the validation process finishes, the staff user can decide a superior effect to that card, which only has effects when certain requirements are satisfied. After that, the card will be added into the card pool (different from the previous one)

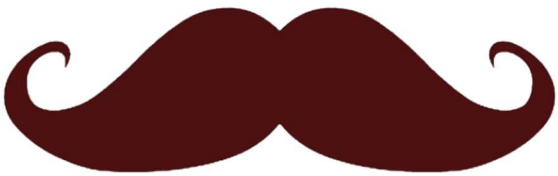
We had a research about this idea and found that this can be implemented by using Vue, which is another front-end framework. Vue is compatible with Bootstrap so the only change we need to do now (if we all agree to use this idea) is to update the Bootstrap library to a Bootstrap-Vue library. If you guys have any thoughts / suggestions about this plan, feel free to leave comment below.

Annabelle Moore, 8 Feb 20:27 · 1 min read

Completed 5 types of accessories

I have completed and uploaded to the repository; 2 types of glasses, 2 types of hat and a moustache. All come in a variety of colours and all images are the same size so that they can be layered easily. I have also uploaded the basic avatar silhouette. Please can you all leave a comment to let me know if these are okay? If you're all happy then I'll move these to verified on the Kanban board. Here are some examples if you can't be bothered to check out what I've uploaded to GitHub:





Annabelle Moore, 8 Feb 23:21 · 1 min read

Subscribe

Avatar backgrounds

I added 3 different coloured backgrounds to github, all of the same size for easy layering

---

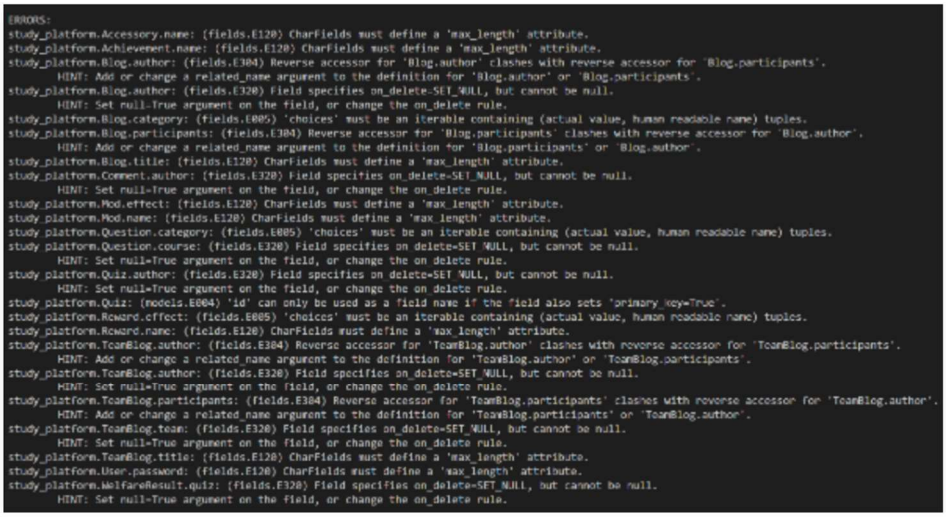
Bethany Griffin, 9 Feb 14:32 · 1 min read

Attempted Django Debugging

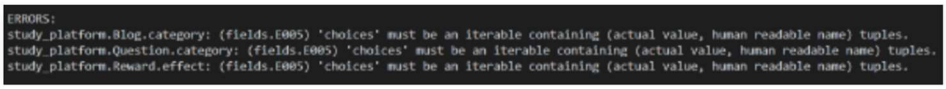
@Clair Newton and I have been attempting to get the backend and frontend to work together for the user login and registration.

At first we struggled to actually get the HTML to run the .py scripts. Then we realised this was because we weren't running manage.py.

When I ran manage.py I got a long list of issues (23 in total I believe, the errors can be see below)



I have currently managed to whittle it down to 3 errors



However to solve there it will likely mean I have to change how the choices are made in the models.py script.

Hopefully this is okay, and if anyone knows how to solve this that would be great!

EDIT: I have fixed those issues. Which urls.py is the correct one? We have two (one in ecm2434 and another in study_platform)

Clair Newton, 9 Feb 15:08 · 1 min read

Subscribe

added a home page

added a basic home page that takes the user to the login/registration pages

will add more information to it about what the project does ie customizable avatar, quizzes etc

---

Clair Newton, 9 Feb 15:09 · 1 min read

Subscribe

uploaded web pages to github

uploaded and merged login page, registration page, home page, and a placeholder profile page

---

Bethany Griffin, 9 Feb 16:09 · 1 min read

Django runs webpage!

@Clair Newton and I have now got it so that when you run manage.py with the runserver command it loads the webpages.

Currently it is only linked to register.html, but the CSS also works.

Later we will attempt to get it to gather information from forms!

Bethany Griffin, Yesterday, 00:26 · 3 min read

Database and Django...

Hi guys,

@Clair Newton and I have encountered some issues with Django that we are in the process of solving. Your feedback about what you think is the best step forward will be greatly appreciated!

Changing the User Model

One such, admittedly rather large, issue is with user registration and log-in. In order to run the *UserCreationForm* and *AuthenticationForm* so that Django to actually created and authenticate users I have 3 options:

1. Use a **proxy model** - This is where the inbuilt user model is linked to a model that isn't stored in the database. For this reason I don't think we'll use this as everything in the user model is something that we need to be able to access.
2. Use a **One-To-One mode**l - This will be a separate model that is stored in the database. We will use the inbuilt user model and everything that this doesn't store (i.e. the stuff that isn't needed for authentication). I feel like this is the best option as it still allows us to use the inbuilt login and authentication processes.
3. Create a **Custom User by extending *AbstractBaseUser* or *AbstractUser*** - This will be where we don't use any of the inbuilt processes, instead we will have to create and re-engineer it all ourselves. I really don't want to do this method if we can avoid it as when I say everything I really mean everything.

As I've said above I think using a One-To-One model will be the best way forward, but I'd like your input on it!

Linking Django to the Database

Another problem is that the Django wasn't actually correctly linked the the database table (if at all). After a great deal of mucking around (and more than a little backtracking) I believe I have eventually got it to connect correctly but (as I am currently unable to register or log in users) only time will tell.

views.py

I believe that the code that we have added into this file is correct, however it all relies on the above two issues resolving before we can fully test them. One function is *login_view* the other is *register_view*. I'm really hoping that when the issues with the models and database are solved then these will just slot into place and work as expected (but as always with code, you never know until you try!)

Changes to register.html and login.html

While the changes to *login.html* and more with the code than actually effecting it's overall aesthetic, the same can't be said for *register.html*. We have left the original HTML in there as comments in case we want to change it, but currently we are using the registration template that Django itself wants to use (when we create a *UserCreationForm* this is what it looks like).

Password Storage

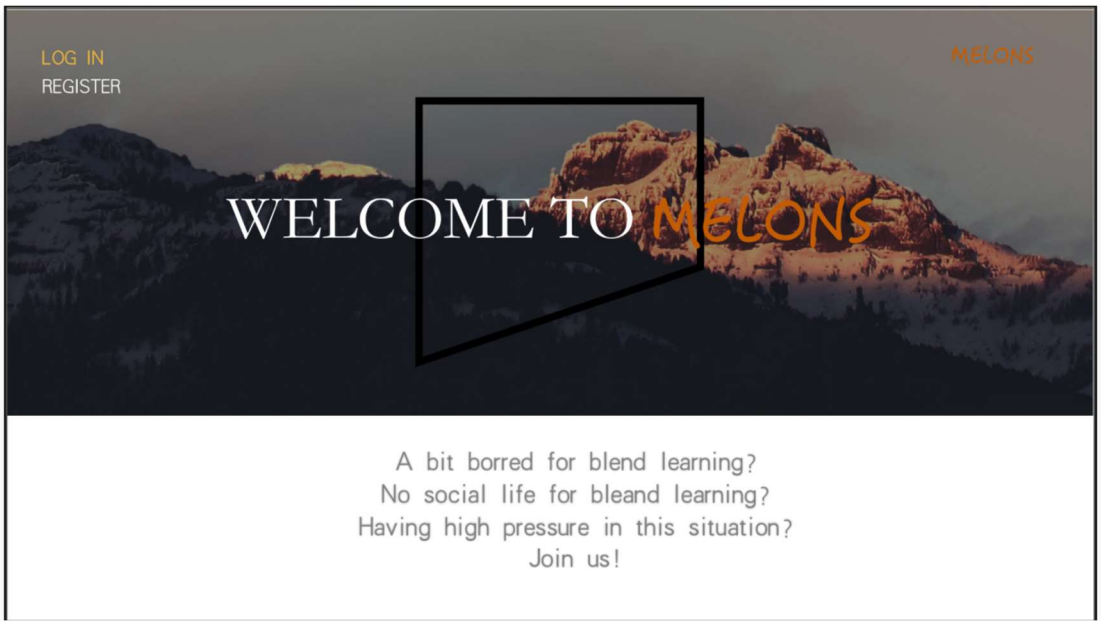
Now the biggest surprise of the evening came from the fact that Django doesn't actually like storing passwords in databases! It wants to store them elsewhere (where exactly I don't know). Alongside this it has the potential to implement different hashing algorithms to protect these passwords. This should be (I say with with more than a hint of nervous) relatively easy to implement - just add to settings.py. I'll have to do a little more research in regards to how this works, but it shouldn't be a major problem (fingers crossed).

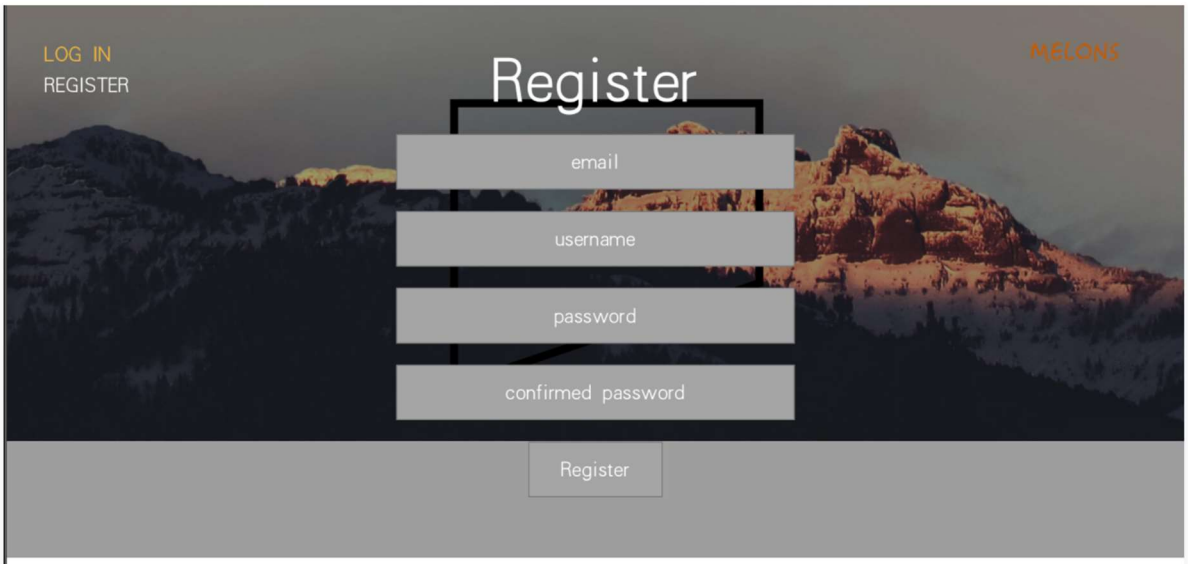
Again any feedback from you guys about what we want to do/change/implement out of all of this will be greatly appreciated. We'll try our best to get it done and dusted before Thursday!

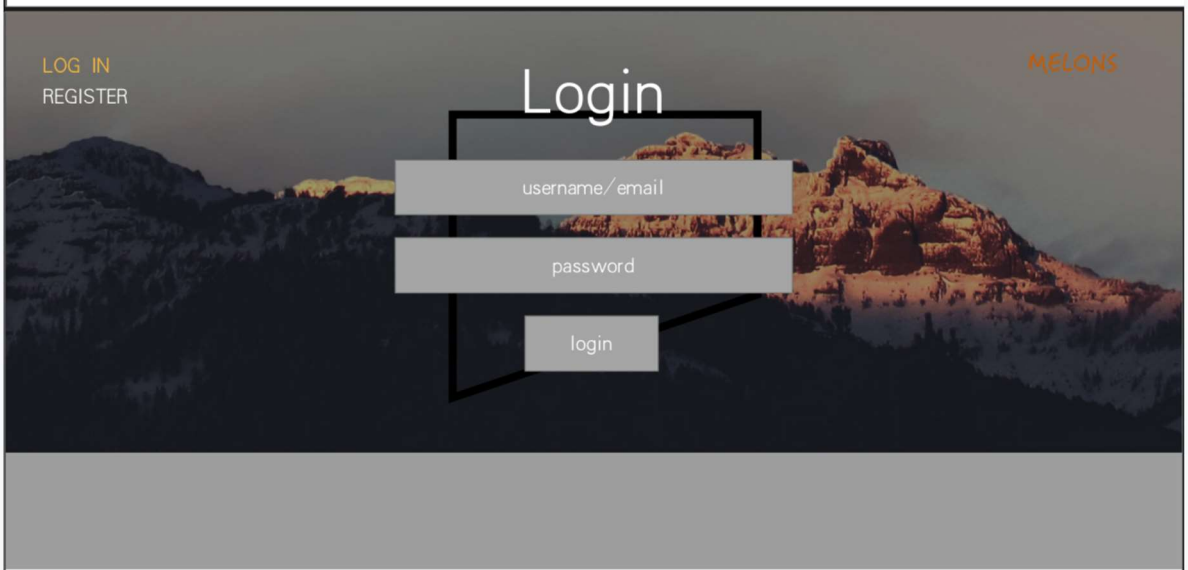
Georgina Huang, Yesterday, 17:45 · 1 min read

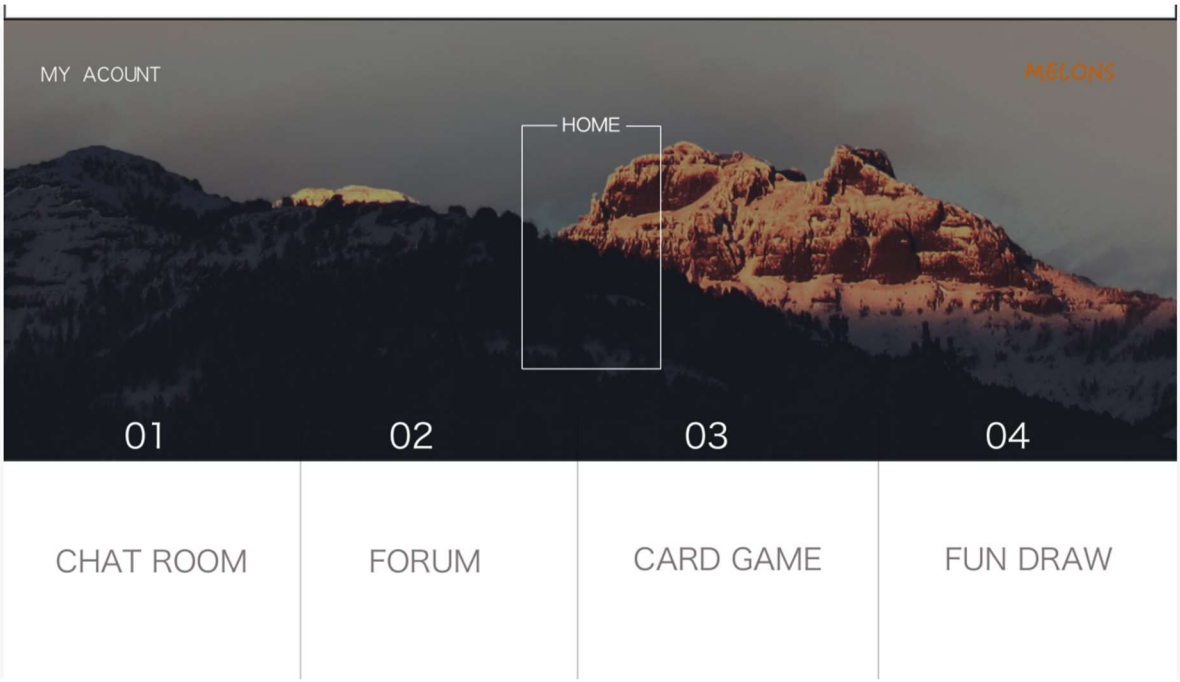
UI

just done our homepage, register page, login page and context page!!!




LOG IN
REGISTER
MELONS
Register
email
username
password
confirmed password
Register


LOG IN
REGISTER
MELONS
Login
username/email
password
login



Bethany Griffin, Today, 00:21 · 1 min read

Website Up and Running

The website is all up and running!

The Django has now linked to the database so users can register and log in.

Currently after logging in it just takes them to a placeholder for the profile, but this will change once the profile page has been made

---

Georgina Huang, Today, 00:43 · 1 min read

UI

card game page.