201522761 Hanmoon Hwang

- A statement of the purpose of this website

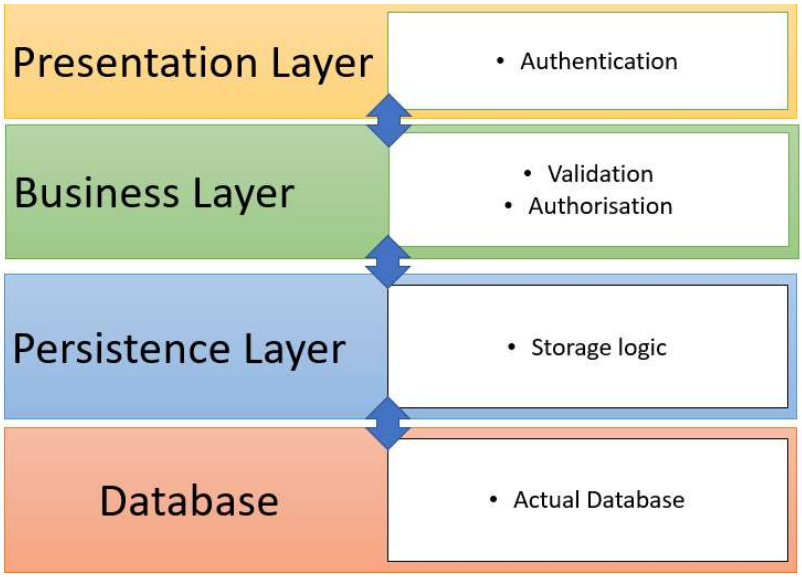
In a nutshell, this website is Player Transfer Platform website where users can hand in a bid on Football player as a team/owner of club and put their player on release market to provoke other's interest or offer.

- A link to the deployed website and username/ password

  General user who is a team has some player

  sc21h2h@gmail.com / 123

- Explain how the web application uses the 3-tier architecture



When it comes to First layer which is called ‘Presentation Layer’, This is the connection part of the user who uses football player biding website or client system directly. In a nutshell, it can be interpreted the UI in the web structure or endpoint in the backend API therefore, this part defines api endpoints and implements to read HTTP requests.

2nd layer, which is Business Layer, is quite important part to implement business logic literally. We have a lot of logics that should be implemented in this system such as the application documentation for player should be written down within valid word counts

Some logics relevant to database are required in 3rd layer, which is Persistence Layer. We can easily catch this logic in our website for instance player transfer between users or servers using release button, editing profile in ‘my-page’ or biding button in transfer market. Such processes accompany with database where data are edited, saved and read.

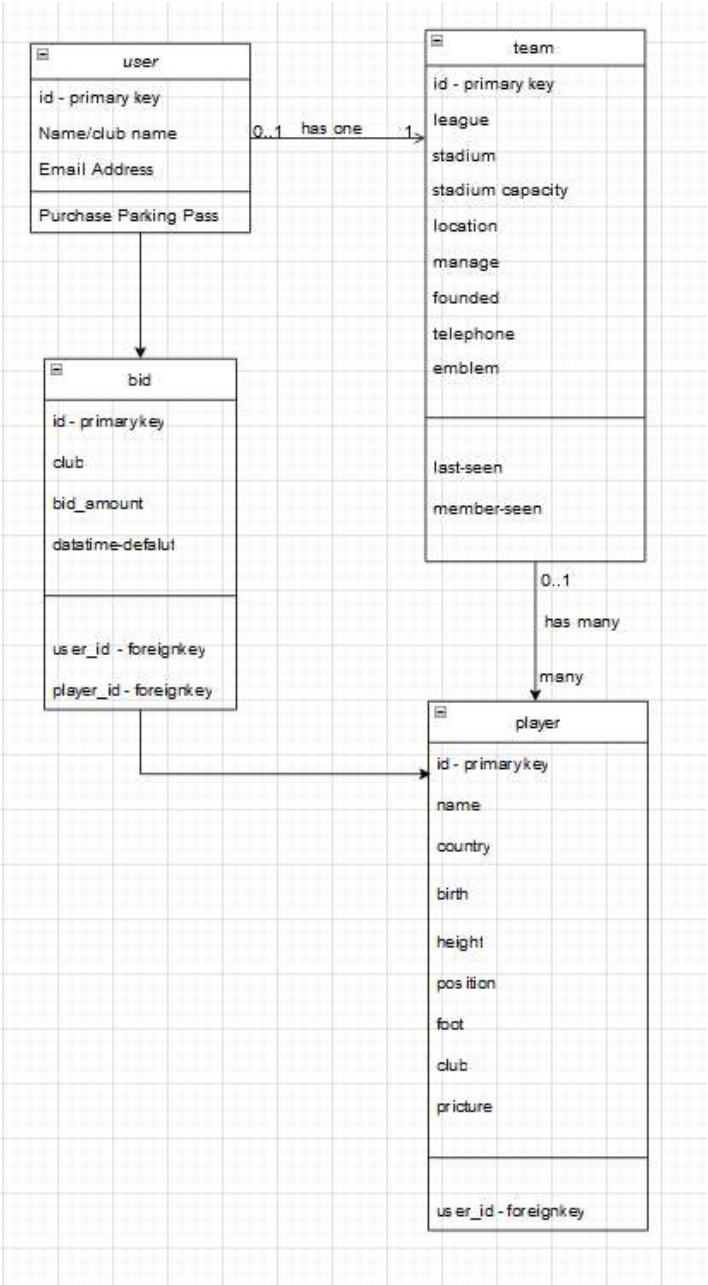
-Core component of this 3tier structure

The core part of these layered architecture is of one-way dependency. Each layer depends on sub layers respectively. Presentation layer(player bid amount or application form validation logic is not yet implemented) -> Business Layer(this is supposed to call persistence layer in order for processing player database logic) -> Persistence Layer

- Describe request handling and HTTP features, relating these to the web application.

  Overall services served in this website adopt Post method because browser requests something like new player data to URL server and should give a guarantee for saving new data like biding amount on player. When it comes to GET method in this website, player searching system seems to be suited as browser will request just some player data stored in player data page.

- An analysis of your web application. This would cover all sections listed in the specification section: Web forms, database including the models and relation- ships, sessions and authentication, styling, advanced features, deployment, tests and logging.



<Authentication>

I followed the basic authentication system (sign-up -> log-in) and gave a system to edit profile in order for releasing or retiring player. Unfortuately, I could not find out the way to edit user profile…

<styling>

Overall style sheet is made by bootstrap except for a small piece of things on page.

However, when putting styling on profile-edit page and sign-up page, there have happened a crush between former styling and the styling that I want even though I used {% extends %} from base.html.

<web form >

I used enum on the form on behalf of typing on form in form.league, form.capacity, foot.

There would be some default forms such as last_seen or datetime.now() that we don't have to type on.

- Explain how accessibility has been considered in the application

  This site is mainly subject to let football teams use. Therefore, main functions require signing up account and logging- in

  If user who is just public, only search, news, q and a are allowable to use.

- A description of potential security issues your web application might encounter

  and how you have mitigated to remove/reduce their impact.

  Potential issue…

1.Sql injection

User input data could cause an unexpected db query abuse such as replacing user_id into string format query which might be vulnerable for authentication.

- ➔ Double check any query which is formatted in a string
- ➔ Double check of using bind parameter when using raw query.

2. Session management

Session data is always not safe even if we save some in db. Session management in serve is required facing against maliciously unexpecting attack.

- ➔ We have PERMANENT_SESSION_LIFETIME variable in flask-session package for automatic session destroy as hours go by.
- ➔ Duplicate login block

3. XSS(cross-stie-scripting)

XSS is known for the situation that admin's js script could be accessed by user. For example, showing pages which are unrelated with normal page in order to bother user or sending some cookies, personal information into certain sites.

- ➔ Always turn on escape mode from html code by user command
- ➔ Don't turn off escape mode

4. Insecure Direct Object References

Direct object Reference or Bad URL is occurred when programmer reference inner object like file, directory, db and forth and expose them.

- ➔ Whether is User_id or value that related to db directly used or not
- ➔ Filter Special symbols which is on its way to upper or sub-directory.
- ➔ Avoid file name or path indeed as parameter in source code in downloading files.

5. Security Misconfiguration

Security configuration should be defined, implemented and maintained since default value is sometimes of unsafety. The most representative vase is of code obfuscation.

- ➔ Python provides encoding/decoding method via Base64 package or encryption in AES algorithm through Crypto in pycrypto package.
- ➔ Double check debug mode 'off'

6. Sensitive Data Exposure

Many web application do not protect personal data like credit card, identification and so on.

Hacker enables vulnerable data to be changed or stolen for abuse like fraud. Therefore it is important to take care about data in a process with browser.

In REST API, there is JSON encryption through JWE(JSON Web Encryption) as data is used in JSON method.

- ➔ JOSE(Js object Signing and Encryption) data encryption
- ➔ Check whether all password encrypt in a hash via bcrypt or PBKDF2
- ➔ Password generation rule

7. Components With known Vulnerabilities

Critical data lose or server down could be happened using vulnerable components as component, library, framework or different software module are mostly executed in titled authority

- ➔ Check Outdated library, keep updating
- ➔ Inactivation of unused functionality

8. Unvalidated Redirects

In a web, user often redirect or use unreliable data for forwarding. Users can be leaded to malignant code or fishing unless there is proper verification process.

- ➔ Be aware of flask-redirect which could be of unsafety