Describe the intended functionality of the web application from a user perspective. This should be about ½ page of text.

The current idea of the subscription manager is a simple database with an interactive website (web application) for the user to access their data about the websites the users have registered for. The web application would provide the data for the current status of their subscription to websites (e.g. Netflix, Amazon Prime video). The web application would also give a user an idea of their total expenditure for a given time frame on web subscriptions.

Describe in general terms the ways the user interacts with the system (make a list). What does the user see, what actions can s/he take, what are their results, what about illegal input/actions?

- First, the user logs into the web application, by providing their username and password.
- Upon successful authentication, the users will be directed to the homepage of the web-service, where they can check the accounts which are linked to the web application(in the ER diagram, listed as a General account).
- In addition to the websites, the users can also see what services are getting from the website, along with the fees associated with those services. It also shows when the next payment is due.
- In addition to the basic expenditure-related information, the users can also gain access to additional account details pertaining to the website to which they are subscribed to, including their payment methods, data usage, terms and conditions, and important updates and messages from those websites. The data related to these details, however, will not be stored in our database. Rather, they would be redirected to the relevant webpage. Thus, for each detail related to the user's certain account, there will be a link stored and that link will be used to redirect the user to the relevant page.
- Illegal Inputs and actions:
    - When the user gives account details for an account that does not exist.
    - When the user gives sign-in credentials that don't resemble any accounts in our web-application
    - Usernames that have already been taken cannot be used to create new accounts for the web-application.
    - More illegal/invalid inputs will be found as the development phase kicks in.

Explanation to new additions:

- Firstly we further specialized the accounts that the subscriber may be subscribed to using an IS-A hierarchy.
- We also added an is-a hierarchy for the different types of users: a premium user and a regular user.
- In addition, the er diagram has changed such that there is a payment part for the premium user.
- Also, we decided to store the payment details that the subscriber uses for paying for the services for any website using the dedicated table called payment details.

- All of the above additions in turn leave us with 3 is-a hierarchies and 5 relations along with 12 entities. We have added all these as using DDL CREATE TABLE in the SQL file.
- The functionality and the input a user can provide are the same however now they can add specific queries for their music subscription or their video subscription.
- This just enables the users to have more specific queries and this also provides better interfaces for the users which is easy and straightforward to use.

Below is the mapping is done:

- The first is-a is for the different types of subscribers premium and regular as mentioned before. This implies that the elements for the account of subscriber (email, username, password) are inherited to the subclasses. This allows us to have a nice distinction between the types of subscribers.
- The second is-a hierarchy is for the details this is also helpful as we can access all the subclasses using the superclasses detail_id as shown by the code. This is an elegant approach to split the details into the different types and access them.
- Final is-a hierarchy is for the different types of the accounts a subscriber may have such as a music sub or video sub or VPN etc. This is as explained before it is easier to access a specific type of subscription by traversing that table this may not always be space-efficient however we this would be a user-friendly approach to the database and for the web application.

Different approaches :

- We could have all the different subs types of such as VPN music sub and video sub in one table however there would be additional places which would have to be null as a video subscription would have a max number of devices which is not an attribute of the Software suite similarly software suites has an expiration date which is also present in the magazine entity. However, magazine also has a type paper or online which is not present in VPN.
    - If we decided to use this approach we would have one table which is easier enquires however there are a lot of NULL values in the table which may not be desirable depending on the type of hardware available for the database.
- Hypothetically (if we don't consider the code and the dependence of the following point on the subscriber's account) For the different types of subscriber account (premium or regular) we could just have those two as separate entities but this would imply that we cant traverse through all the users in one table we would have to do this twice for each table.
- We can debate the is-a hierarchy for the details in a similar fashion. It really comes down to what we want to optimize (space or usability (easier to traverse) and how user-friendly the overall product turns out.