CPS714 Lab 8

Tech Stack

Backend: Django (python)
Frontend: HTML, CSS, JS

Schema Update

We haven't changed anything as of yet, the schema design remains the same:

• reward_id (PK) - INT, Auto Increment
• user_id (FK) - INT, References user(user_id)
• points_earned - INT
• points_redeemed - INT
• reward_description – VARCHAR (255)
• is_active - boolean
• earned_date - TIMESTAMP
• redeemed_date - TIMESTAMP, Nullable

Development Design

For Point accumulation, we were thinking about an api call that is sent based on actions that the user takes

- For instance, they purchase a product and based on the price and category of the product (an api call is sent in the backend with this information), they will earn a certain amount of points and the rewards table will be updated with a new tuple for the points gained as a result

For the rewards catalog, we were thinking of having a rewards page where users can redeem points for either discounts/store credit, or donations

- For example, redeem $5 credit for 50 points, or Plant A Tree for 100 points
- A new tuple will be created when something is redeemed with the points used for redemption in the points_redeemed attribute

For Redemption Tracking, we will just notify the user with a pop-up window immediately after they redeem a reward from the rewards catalog

For Bonus Points For Engagement, we will just be using the point accumulation logic with an api call that is sent when a user engages in special events/campaigns promoting sustainability, and the rewards table will be updated with a new tuple for the points gained as a result

Integration

For integration, it should be pretty simple because most of the interaction between the components can be done through api calls

- For the User profile team, we can set up an api call that accepts a user id and returns the amount of points they have, and redemption history

- For the admin dashboard, we can use the same api call, since they'll also be looking at the point redemption history for each user individually