Testing Strategy for Green Master

Overview

The main reason for the tests is to check that each of the models in the database accept only valid values and that functions can raise exceptions, and these be handled, and error messages returned in the UI. These tests will be split by app as these are where the tests.py files are stored.

Strategy

In order to make sure every new feature adheres to these tests; they will be carried out whenever a pull request is made to merge a feature branch into the dev branch and when the dev branch is merged into main. This will ensure that when a feature is created it is not having any unexpected effects on the models or other functions.

Accounts App

In this app the user model is created with a username, password, email, points and moderator status.

1. Test a user has the correct attributes saved.
    - This makes sure that when a user is created that it can then be fetched from the database by searching by username. It will make sure all attributes are as the user was created with.
2. Test user isn't created as a superuser.
    - This test makes sure that when the sign in function is called that a default user is created with no amin privileges.
3. Test user must be created with a username.
    - As a user must have a username to be displayed on the leaderboard, a user must be created with a username.
    - This test will make sure an error is thrown if someone attempts this.
4. Test username must be unique.
    - As a user must be uniquely identifiable the username must be unique, and it must be possible to know to when to throw an error if a user tries to use a taken username.
5. Test email must be unique.
    - To stop user's having multiple accounts the email must be unique for each of the users.
6. Test user is created with 0 points.
    - So that no users are given a head start this will test that a user is created with 0 points when they sign up.

Gamekeeper Homepage

This is the app in which a gamekeeper will approve submissions and from this points will be allocated to user's if they are valid.

1. Test negative points can't be added to a user.
    - Makes sure that when a user completes a task/submission only positive numbers of points will be added to their score otherwise an error will be thrown.
2. Test zero points can't be added.
    - Makes sure that when a user enters a valid submission at they will have at least one point added to their score.
3. Test that points are added to a user.
    - Will create a new user and then subsequently add different numbers of points to make sure a user is able to have points added. Will then get the user from the database and make sure these changes have been committed.

Leaderboard App

The app in which a user will be able to select between a leaderboard of user points of buildings by CO2 submissions.

1. Test leaderboard can handle no users.
    - Will create a leaderboard with no users on it and makes sure it can be created.
2. Test leaderboard can handle one user..
    - Will create a leaderboard with one user on it and makes sure it can be created without an index out of bounds error.
3. Test leaderboard can handle two users.
    - Will create a leaderboard with two users on it and makes sure it can be created without an index out of bounds error.
4. Test leaderboard can handle three users.
    - Will create a leaderboard with three users on it and makes sure it can be created without an index out of bounds error.
5. Test leaderboard can handle many users.
    - Tests when users are added to the bottom proportion of the leaderboard that they appear.

6. Test it is possible to calculate the CO2 emissions of a building with windows open and lights on.
    - Will calculate by hand the predicted CO2 emissions for the given building with both the lights on and window open and will make sure the value returned is approximately correct (to allow for floating point inaccuracies).
7. Test it is possible to calculate the CO2 emissions of a building with windows closed and lights on.
    - Will calculate by hand the predicted CO2 emissions for the given building with the lights on and will make sure the value returned is approximately correct (to allow for floating point inaccuracies).
8. Test it is possible to calculate the CO2 emissions of a building with windows openand lights off.

- Will calculate by hand the predicted CO2 emissions for the given building with the windows open and will make sure the value returned is approximately correct (to allow for floating point inaccuracies).

9. Test when CO2 calculated for a room with lights off or automatic and windows off or automatic the CO2 emissions are 0.
    - Will make sure a non-zero value isn't given for an environmentally friendly room.

Submission App

This is the app in which a user will submit their photo of the room as well as some information about the room in order to calculate environmental statistics for each building.

1. Test a user can upload an image in a variety of valid formats.
    - This is a sequence of tests ensuring that an image with the file extensions (.jpg, .jpeg, .gif, .png)
    - These will be simulated by creating random files with these extensions and checking that they are accepted.
2. Test input stats method changes the database if lights and windows are on.
    - Tests that when stats are input, they are committed to the database.
3. Test input stats doesn't change the database if the windows are closed and the lights off.
    - Tests that database isn't changed unnecessarily if there are no stats to add to it.
4. Test input stats doesn't change the database if the windows or lights are automatic.
    - As the windows and lights are assumed to be efficient if they are automatic there is no need for them to be input into the database.