Coverage for **server.py**: 95%

85 statements | 81 run | 4 missing | 0 excluded

« prev *^ index* *» next* *coverage.py v7.3.2, created at 2023-11-03 01:37 +0100*

```python
# Importations de modules
import json
from flask import Flask, render_template, request, redirect, flash, url_for
from datetime import datetime  # Ajouté pour la gestion du temps


# Fonctions pour charger les données
def load_clubs():
    with open('clubs.json') as c:
        list_of_clubs = json.load(c)['clubs']
        return list_of_clubs


def load_competitions():
    with open('competitions.json') as comps:
        list_of_competitions = json.load(comps)['competitions']
        return list_of_competitions


# Fonction pour vérifier la validité de la date
def check_date_validity(competition: dict) -> bool:
    current_time = datetime.now()
    competition_time_str = competition["date"]
    competition_time = datetime.strptime(competition_time_str, "%Y-%m-%d %H:%M:%S")
    return current_time < competition_time


# Fonctions supplémentaires
def get_club_and_competition(club: str, competition: str) -> (bool, dict, dict):
    competition = [c for c in competitions if c['name'] == competition]
    club = [c for c in clubs if c['name'] == club]
    if not competition or not club:
        return False, None, None
    return True, club[0], competition[0]


def update_places(competition: dict, places_required: int, club: dict) -> bool:
    nbr_places = int(competition['numberOfPlaces'])
    club_nbr_points = int(club['points'])
    if (0 < places_required < 13) and (places_required <= club_nbr_points):
        competition_places = nbr_places - places_required
        if competition_places < 0:
            return False
        competition['numberOfPlaces'] = competition_places
        club_nbr_points = club_nbr_points - places_required
        club['points'] = str(club_nbr_points)
        return True
    else:
        return False


# Initialisation de l'application Flask
app = Flask(__name__)
app.secret_key = 'something_special'

competitions = load_competitions()
clubs = load_clubs()


# Vérification de l'email
def check_email(email: str) -> dict:
    clubs_found = []
    for club in clubs:
        if club['email'] and club['email'] == email:
            clubs_found.append(club)
    if not clubs_found:
        return {}
    else:
        return clubs_found[0]


# Routes
@app.route('/')
def index():
    return render_template('index.html')


@app.route('/showSummary', methods=['POST'])
def show_summary():
    club = check_email(request.form['email'])
    if not club:
        return render_template('index.html', email_error=True)
    else:
        return render_template('welcome.html', club=club, competitions=competitions)


@app.route("/book/<competition>/<club>", methods=['GET', 'POST'])
def book(competition, club):
    foundClub = [c for c in clubs if c["name"] == club][0]
    foundCompetition = [c for c in competitions if c["name"] == competition][0]

    # Assurez-vous que la compétition n'appartient pas au passé...
    if not check_date_validity(foundCompetition):
        flash("Selected competition is over")
        return render_template("welcome.html", club=foundClub, competitions=competitions)

    if foundClub and foundCompetition:
        max_places = min(
            12, int(foundClub["points"]), int(foundCompetition["numberOfPlaces"])
        )
        return render_template(
            "booking.html",
            club=foundClub,
            competition=foundCompetition,
            max_places=max_places,
        )
    else:
        flash("Something went wrong-please try again")
        return render_template("welcome.html", club=club, competitions=competitions)


@app.route('/purchasePlaces', methods=['POST'])
def purchase_places():
    ok_flag, club, competition = get_club_and_competition(request.form['club'], request.form['competition']
    if not ok_flag:
        return redirect('/')
    places_required = int(request.form['places'])
    if update_places(competition, places_required, club):
        flash('Great-booking complete!')
    else:
        flash('You tried to book an invalid number of places, sorry')
    return render_template('welcome.html', club=club, competitions=competitions)


@app.route('/club_table.html')
def club_table():
    return render_template('club_table.html', clubs=clubs)


@app.route('/logout')
def logout():
    return redirect(url_for('index'))

```

```python

# Lancement de l'application
if __name__ == '__main__':
    app.run(debug=True)
```