Universitatea Tehnică "Gheorghe Asachi" din Iași
Facultatea de Automatică și Calculatoare
Baze de date - Temă de acasă

NetStar - Administrarea unui Internet Café

Baltariu Ionuț-Alexandru - Grupa 1305A
Cadru didactic coordonator: Mironeanu Cătălin

1 Descrierea proiectului - scopul aplicației

Având în vedere numărul relativ ridicat de clienți din ultima vreme al Internet Café-urilor, se dorește patentarea unei aplicații care ar rezolva nevoile evidente ale afecerii anterior menționate. Printre acestea enumerăm:

- punerea la dispoziția clienților a unui sistem de logare/inregistrare pentru a putea rezerva anumite dispozitive;
- oferirea unei interfețe grafice intuitive și ușor de utilizat atât de clienți cât și de angajați;
- monitorizarea și contorizarea timpului petrecut la un dispozitiv de către clienți, a creditelor speciale aferente contului, si a altor date relevante;
- stocarea tuturor informațiilor relevante într-o bază de date.

Produsul final are în vizor, astfel, automatizarea și eficientizarea anumitor activități repetitive. Având un sistem automat de rezervări, este evident că șansele ca o persoană să apeleze la serviciile Internet Café-ului cresc într-un mod considerabil.

Clientul va putea vedea:

- statistici despre timpul petrecut în cadrul locației, calculatorul favorit, etc;
- dacă un interval orar este deja ocupat, fie integral sau parțial, de către alt client;
- configurațiile calculatoarelor, care sunt actualizate direct din baza de date.

De asemenea, pentru a-și forma o părere, orice client va putea vedea feedback-urile altor clienți, putând să lase în orice moment o părere.

2 Tehnologii folosite

2.1 Front-End

Având în vedere că este vorba despre o aplicație Web, pentru partea de **front-end** este folosit atât clasicul combo **HTML-CSS**, cât și framework-ul **Bootstrap**.

2.2 Back-End

Back-End-ul constă în **Flask**, bine cunoscutul micro-framework web al limbajului **Python**. Acesta a fost ales atât pentru ușurința cu care se pot crea aplicații web folosindu-l, cât și pentru versatilitatea pe care o presupune.

Pentru realizarea aplicației au mai fost folosite modulele:

- **cx-Oracle**(conectare la o bază de date Oracle);

- **bcrypt**(criptarea parolelor);
- **os**(transmitere variabile de mediu);
- **re**(parsare eficientă de text);
- **datetime**(lucru cu date în Python);
- **selenium**(populare aplicație cu utilizatori).

2.3 Tehnologii auxiliare

- **Git** - pentru controlarea versiunilor aplicației;
- **Docker** - pentru virtualizarea aplicației la nivel de sistem de operare;
- **PyCharm** și **VS Code** - IDE & editor de cod;
- **Oracle XE** - creare bază de date locală (”back-up” atunci când baza de date de la facultate nu funcționează sau rămâne agățată);
- **Oracle Data Modeler** - creare tabele, constrângeri, etc (în principiu, tot ce ține de tabelele din aplicație);
- **Oracle SQL Developer** - alternativa prietenoasă a unui command-line, folosit pentru că oferă posibilitatea de a vizualiza tabelele și datele dintr-o bază de date. A mai oferit posibilitatea de a testa comenzi și de a manipula datele.

3 Structura și inter-relaționarea tabelelor

3.1 Diagrama ER

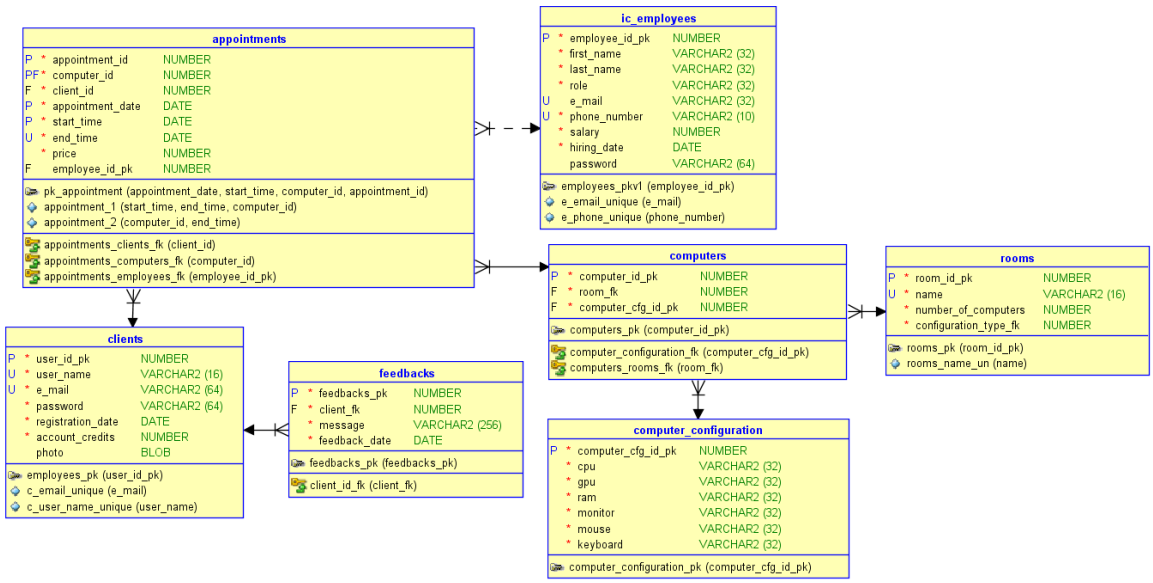

Figure 1: Diagramă preluată din Data Modeler

Deși tabela centrală pare a fi cea de programări(Appointments), o importanță mai mare le au tabelele **Clients** și **ic_employees**, întrucât, fără date din aceste tabele, aplicația nu ar avea nici un sens sau scop.

Flow:

- Se consideră **înregistrat** un **client** care este regăsit în tabela **clients**;
- Un **client** face o **programare**(*appointments*) la o anumită **dată** și pe un anumit **calculator**;

- **Calculatorul** face parte dintr-o anumită **cameră**(*rooms*) și are o anumită **configurație**(*computer_configurations* - din nou, legată de cameră);
- După ce s-a făcut programarea, un **angajat**(*ic_employees*) o poate **valida**(se consideră validată dacă este **nenul** câmpul *employee_id_pk* din **appointments**;
- **Clientul** poate lăsa un **feedback** după experiența din Internet Caffe.

3.2 Aspecte legate de normalizare

Structura inițială a tabelului nu a presupus foarte multe scenarii care ar fi necesitat normalizări.

Un exemplu ar fi alegerea de a avea o tabelă separată pentru configurațiile unui calculator, pentru a nu scrie în mod repetat sumedenia de informații aferente unei configurații în fiecare linie a tabelei *computers*.

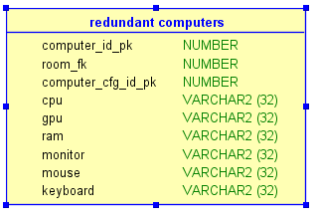

Figure 2: Implementare redundantă

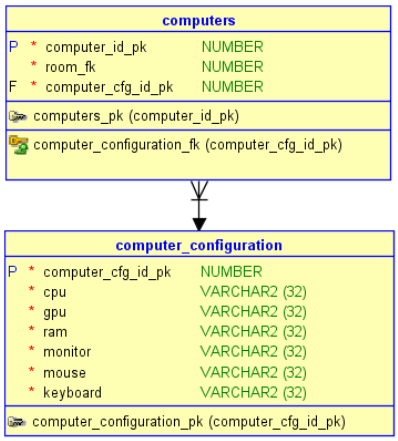

Figure 3: Implementare corectă

Nu au existat alte scenarii care să necesitate normalizare.

4 Descrierea constrângerilor din baza de date

În principiu, constrângerile au fost folosite pentru a nu permite utilizatorului să introducă date cu un format eronat.

- REGEXP_LIKE (first_name,`^[^0-9]+$`) - pentru a nu permite introducerea unui nume care să conțină cifre;
- e_mail LIKE `%_@__%.__%` - pentru a nu permite introducerea unui e-mail cu format fără sens;
- REGEXP_LIKE (phone_number,`^[0-9]{10}$`) - asigurarea faptului că numărul de telefon are exact 10 cifre;

- **end_time > start_time** - o programare nu poate avea data terminării mai ”mică” decât data începerii;
- **length(user_name) >= 4** - un user name ar trebui să aibă mai mult de 4 caractere;
- **length(password) > 4** - aceeași explicație pentru parolă;
- **account_credits >= 0** - creditele dintr-un cont nu pot avea valoarea negativă;
- mai există constrângeri pentru unicitatea datelor (**UNIQUE**), pentru **primary key** și pentru **foreign key**.

5 Conectarea la baza de date din aplicație

Având în vedere simplitatea sintaxei limbajului Python, conectarea la baza de date s-a realizat folosind doar următoarea comandă:

db_connection = cx_Oracle.connect(db_user + '/' + db_password +
'@//bd-dc.cs.tuiasi.ro:1539/orcl')

Unde:

- **db_user** reprezintă Username-ul;
- **db_password** reprezintă parola;
- **'@//bd-dc.cs.tuiasi.ro:1539/orcl'** este serverul la care dorim să ne conectam.

Se vor putea executa comenzi folosind un **cursor**:

cursor = db_connection.cursor()

cursor.execute(comandă_SQL)

6 Mențiuni

- Datele au fost introduse în baza de date în 3 faze pentru că parola este criptată la momentul înregistrării, astfel introducerea acesteia direct printr-o comandă ar presupune calcularea hash-ului parolei dorite în avans.
- Printre elementele HTML/CSS ale aplicației se regăsesc structuri ale cărui cod a fost preluat de la alți creatori. Resursele se pot găsi pe pagina de **github**a proiectului.
  `https://github.com/xeno-john/NetStar_BD`
- Aplicația web rulează atât pe Windows cât și pe Linux.

7 Capturi de ecran și exemple de cod

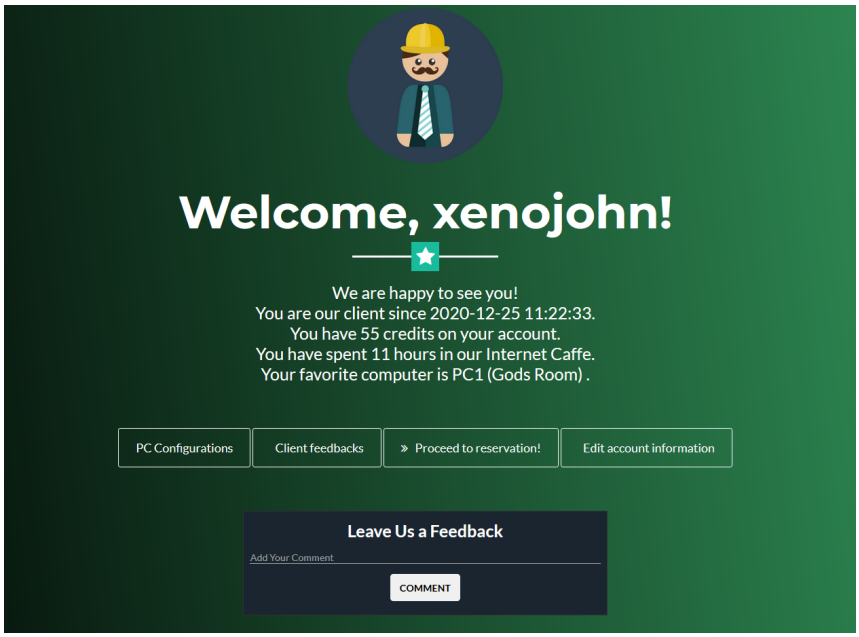

Figure 4: Pagina de profil

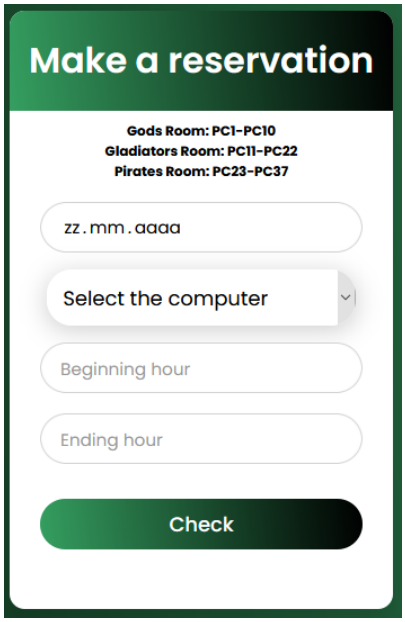

Figure 5: Pagina pentru rezervări

```python
@server.route("/profile", methods=['GET'])
def profile():
    if "username" in session:
        spent_hours = 0
        spent_minutes = 0
        favorite_computer = 0
        favorite_computer_room = ""

        account_credits = cursor.execute(
            "select account_credits from clients where user_name = '" + session["username"] + "'").fetchall()[0][0]
        # date from oracle translates easily into datetime object in python
        registration_date = cursor.execute(
            "select registration_date from clients where user_name = '" + session["username"] + "'").fetchall()[0][0]

        user_id_pk = cursor.execute("select user_id_pk from clients where user_name = '"
                                    + session["username"] + "'").fetchall()[0][0]
        user_name = session["username"]
        for row in cursor.execute("select start_time,end_time, computer_id from appointments " +
                                  "where client_id = " + str(user_id_pk) + " and employee_id_pk is not NULL"):
            if row[1] < datetime.now():
                spent_hours = spent_hours + row[1].hour - row[0].hour
                spent_minutes = spent_minutes + row[1].minute - row[0].minute

        command = cursor.execute("select computer_id from ( select count(computer_id) as freq, computer_id " +
                                 "from appointments WHERE client_id=" + str(user_id_pk) +
                                 " and appointment_date<sysdate and employee_id_pk is not NULL " +
                                 "group by computer_id order by freq desc ) " +
                                 "where rownum=1").fetchall()

        if cursor.rowcount != 0:
            favorite_computer = command[0][0]

        if favorite_computer != 0:
            favorite_computer_room = cursor.execute("select name from rooms where room_id_pk =" +
                                                    " (select room_fk from computers where computer_id_pk ="
                                                    + str(favorite_computer) + ")").fetchall()[0][0]
        while spent_minutes > 60:
            spent_minutes -= 60
            spent_hours += 1
```

Figure 6: Codul pentru pagina de profil

```python
# row[0] -> start_time
# row[1] -> end_time
for row in cursor.execute("select start_time,end_time from appointments where computer_id = "
                          + str(selected_computer)):
    if beginning_hour < row[1] and ending_hour >= row[0]:
        g_error_msg = "Reservation for the given computer overlaps with another reservation."
        return redirect(url_for("reservation"))

for row in cursor.execute("select start_time,end_time from appointments where client_id =" + str(user_id_pk)):
    if beginning_hour < row[1] and ending_hour >= row[0]:
        g_error_msg = "One cannot have more than a reservation for a given period."
        return redirect(url_for("reservation"))

price = calculate_appointment_price((ending_hour.hour - beginning_hour.hour) * 60
                                    + ending_hour.minute - beginning_hour.minute)

cursor.execute("insert into appointments(computer_id,client_id,appointment_date,start_time,end_time,price) " +
               "values(" + selected_computer + ",(select user_id_pk from clients where user_name = '" +
               session["username"] + "')," +
               "to_date('" + str(beginning_hour) + "','YYYY/MM/DD hh24:MI:SS'),to_date('" +
               str(beginning_hour) + "','YYYY/MM/DD hh24:MI:SS'),to_date('" + str(ending_hour) +
               "','YYYY/MM/DD hh24:MI:SS')," + str(price) + ")")
cursor.execute("commit work")
return redirect(url_for("profile"))
```

Figure 7: Codul pentru pagina de rezervări

8 Resurse

Sunt aceleași link-uri precum cele de pe **Github**.

1. `https://www.docker.com/blog/containerized-python-development-part-1/`
2. `https://docs.docker.com/engine/install/ubuntu/`
3. `https://cx-oracle.readthedocs.io/en/latest/`
4. `https://www.youtube.com/watch?v=IolxqkL7cD8`

5. https://gist.github.com/kimus/10012910
6. https://www.serverlab.ca/tutorials/linux/administration-linux/how-to-set-environment-variables-in-linux/
7. https://blog.bitsrc.io/how-to-pass-environment-info-during-docker-builds-1f7c5566dd0e
8. https://stackoverflow.com/questions/31198835/can-we-pass-env-variables-through-cmd-line-while-building-a-docker-image-through
9. https://stackoverflow.com/questions/51470/how-do-i-reset-a-sequence-in-oracle
10. https://dba.stackexchange.com/questions/137606/slot-time-challenge-doctor-appointment-database-schema?fbclid=IwAR1SKPAwIhcOnalEI4ugTBxJnezkqFX6lsMHvGPmewG9KYaN2Y5b2Ty_nuA
11. https://stackoverflow.com/questions/1167767/check-constraint-of-string-to-contain-only-digits-oracle-sql
12. https://www.youtube.com/watch?v=9MHYHgh4jYc
13. https://www.youtube.com/watch?v=Z1RJmh_OqeA&t=1829s
14. https://www.codingnepalweb.com/2020/08/login-form-with-floating-label-animation.html-template
15. https://stackoverflow.com/questions/39923686/bootstrap-change-background-color
16. https://freefrontend.com/css-select-boxes/
17. http://infolab.stanford.edu/~ullman/fcdb/oracle/or-time.html
18. https://developer.oracle.com/dsl/prez-python-times-and-dates.html
19. https://stackoverflow.com/questions/12378424/how-to-get-the-last-row-of-an-oracle-a-table/12378571
20. https://stackoverflow.com/questions/20233721/how-do-you-index-on-a-jinja-template
21. https://stackoverflow.com/questions/1465249/get-lengths-of-a-list-in-a-jinja2-template
22. https://github.com/habastil1/Comment-Box-As-On-Youtube
23. https://stackoverflow.com/questions/4151743/how-can-i-change-the-thickness-of-my-hr-tag
24. Laboratoare BD.