

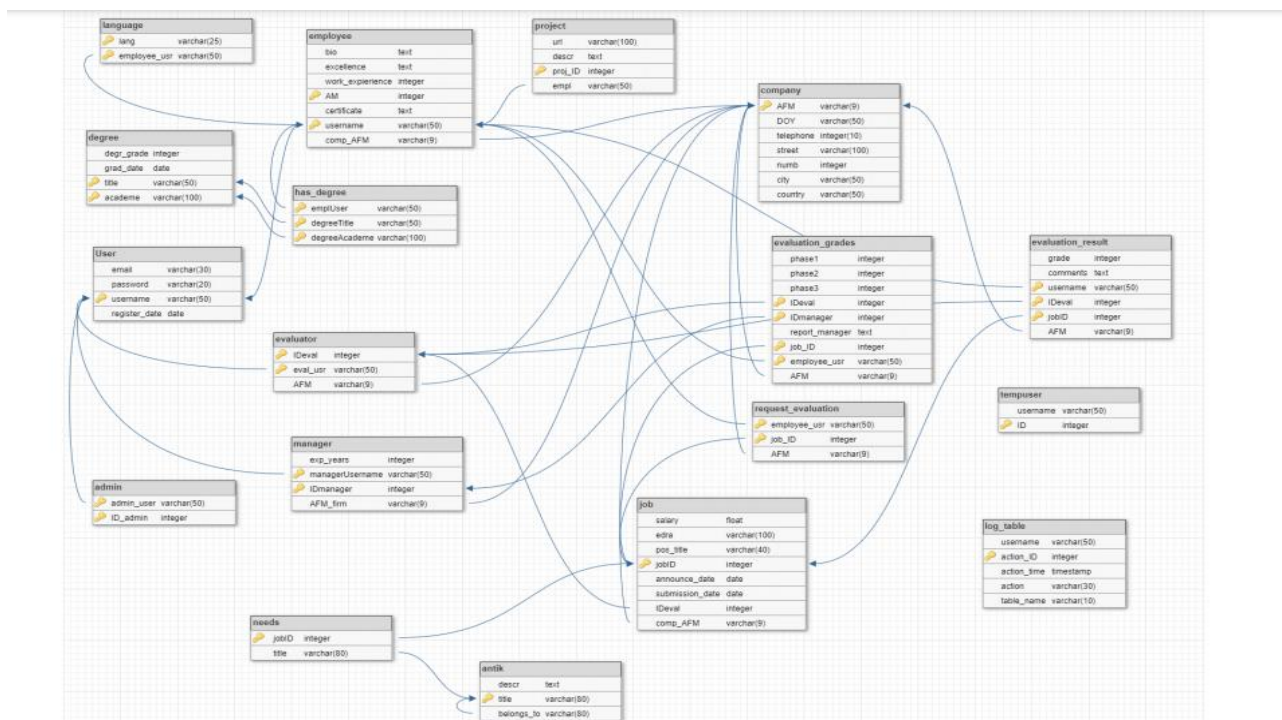
Project -Βάσεις Δεδομένων

Ονοματεπώνυμο: Μάριος Στεφανίδης, AM: 1067458

Ονοματεπώνυμο: Κατερίνα Μητροπούλου, AM: 1067409

Μέρος A: ΒΔ και SQL

Ερώτημα 1:



(*Το ER της βάσης υπάρχει και σε pdf στο .zip)

1. Αποφασίσαμε να αναπαραστήσουμε τους χρήστες τους συστήματος (evaluator, employee, admin και manager) με Is-A. Όλοι θα έχουν κάποια κοινά χαρακτηριστικά, τα οποία θα “κληρονομούν” μέσω της οντότητας-πίνακα user.
2. Οι χρήστες μπορούν να γνωρίζουν πολλές γλώσσες, επομένως το γνώρισμα language είναι πλειότιμο, γι’ αυτό και δημιουργούμε ξεχωριστό πίνακα.

3. Κάθε εργαζόμενος μπορεί να κατέχει παραπάνω από ένα πτυχία, όπως και κάθε πτυχίο μπορεί να ανήκει σε παραπάνω από έναν εργαζόμενο (M-N συσχέτιση), γι' αυτό δημιουργείται και ο πίνακας `has_degree`.
4. Θεωρούμε ότι κάθε `project` αναλαμβάνεται από έναν και μόνο εργαζόμενο, ενώ κάθε εργαζόμενος μπορεί να αναλάβει παραπάνω από ένα `project` (1-N συσχέτιση).
5. Ένας αξιολογητής μπορεί να ανακοινώσει παραπάνω από μια θέσεις, το αντίστροφο όμως δεν μπορεί να συμβεί.
6. Κάθε αντικείμενο μπορεί να αντιστοιχίζεται σε παραπάνω από μία θέσεις εργασίας, όπως και μία θέση εργασίας μπορεί να απαιτεί την γνώση παραπάνω από ενός αντικειμένου.

Όσον αφορά στην αξιολόγηση ενός υπαλλήλου για μια θέση εργασίας, αποφασίσαμε τα εξής:

1. Κρατάμε και τις 3 φάσεις της αξιολόγησης στον πίνακα `evaluation_grades`. Στα γνωρίσματα `phase1`, `phase2` και `phase3` αντιστοιχούν οι βαθμοί για κάθε στάδιο της αξιολόγησης, οι οποίοι θα καταχωρούνται μόνο από τον αξιολογητή. Το γνώρισμα `report_manager` αντιστοιχεί στην "έκθεση" του διευθυντή που γίνεται στην 2. φάση.
2. Στον πίνακα `request_evaluation` καταχωρούνται οι αιτήσεις από τους υπαλλήλους για τις θέσεις που ενδιαφέρονται.
3. Στο `evaluation_result` καταχωρούνται οι αξιολογήσεις που έχουν ολοκληρωθεί, με τον συνολικό βαθμό και κάποια σχόλια από τον αξιολογητή.

Όσον αφορά στον πίνακα ενεργειών (`=log`), δυσκολευτήκαμε πολύ να βρούμε τρόπο να αποθηκεύουμε κάθε φορά το όνομα του χρήστη που κάνει κάποια αλλαγή στην ΒΔ. Καταλήξαμε, όμως, στο εξής:

Αποθηκεύουμε, πριν γίνει η αλλαγή σε κάποιο πίνακα της ΒΔ και μέσω του `gui`, το όνομα του χρήστη στον πίνακα `tempuser` και από εκεί τα διάφορα `triggers` το ανακτούν (με την βοήθεια του κλειδιού ID που είναι `AUTO_INCREMENT`) και το μεταφέρουν στον πίνακα `log`. Θεωρώ ότι κάθε φορά ο χρήστης που έχει το μεγαλύτερο ID, είναι και αυτός που κάνει κάποια αλλαγή, αφού γίνεται η εισαγωγή του στον `tempuser`, ακριβώς πριν την αλλαγή.

Ερώτημα 2:

Σύνολο εντολών create:

```
CREATE TABLE company
```

```
(
```

```
    AFM CHAR(9) NOT NULL,
```

```
    DOY VARCHAR(50) NOT NULL,
```

```
    telephone INT(10),
```

```
    street VARCHAR(100),
```

```
    numb TINYINT,
```

```
    city VARCHAR(50),
```

```
    country VARCHAR(50),
```

```
    PRIMARY KEY (AFM)
```

```
)engine=InnoDB;
```

```
CREATE TABLE user
```

```
(
```

```
    email VARCHAR(30) NOT NULL,
```

```
    password VARCHAR(20) NOT NULL,
```

```
    username VARCHAR(50) NOT NULL,
```

```
    register_date DATE,
```

```
    PRIMARY KEY (username)
```

```
)engine=InnoDB;
```

```
CREATE TABLE admin
```

```
(
```

```
    admin_user VARCHAR(50) NOT NULL,
```

```
    ID_admin TINYINT NOT NULL AUTO_INCREMENT,
```

```
    PRIMARY KEY(ID_admin),
```

```
    FOREIGN KEY (admin_user) REFERENCES user(username)
```

```
    ON DELETE CASCADE ON UPDATE CASCADE
```

```
)engine=InnoDB;
```

```
CREATE TABLE employee
```

```
(  
    bio TEXT,  
    excellence TEXT,  
    work_experience TINYINT,  
    AM INT(10) NOT NULL,  
    certificate TEXT,  
    username VARCHAR(50) NOT NULL,  
    comp_AFM CHAR(9) NOT NULL,  
    PRIMARY KEY (username, AM),  
    FOREIGN KEY (username) REFERENCES user(username)  
    ON DELETE CASCADE ON UPDATE CASCADE,  
    FOREIGN KEY (comp_AFM) REFERENCES company(AFM)  
    ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

```
CREATE TABLE language
```

```
(  
    lang VARCHAR(25) NOT NULL,  
    employee_usr VARCHAR(50) NOT NULL,  
    PRIMARY KEY (lang, employee_usr),  
    FOREIGN KEY (employee_usr) REFERENCES employee(username)  
    ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

```
CREATE TABLE manager
```

```
(  
    exp_years TINYINT,  
    managerUsername VARCHAR(50) NOT NULL,  
    AFM_firm CHAR(9) NOT NULL,
```

```
IDmanager INT NOT NULL,  
  
PRIMARY KEY (managerUsername, IDmanager),  
  
FOREIGN KEY (managerUsername) REFERENCES user(username)  
  
ON DELETE CASCADE ON UPDATE CASCADE,  
  
FOREIGN KEY (AFM_firm) REFERENCES company(AFM)  
  
ON DELETE CASCADE ON UPDATE CASCADE  
  
)engine=InnoDB;
```

```
CREATE TABLE degree  
  
(  
  
    degr_grade TINYINT NOT NULL,  
  
    grad_date DATE NOT NULL,  
  
    title VARCHAR(50) NOT NULL,  
  
    academe VARCHAR(100) NOT NULL,  
  
    PRIMARY KEY (title, academe)  
  
)engine=InnoDB;
```

```
CREATE TABLE has_degree  
  
(  
  
    emplUser VARCHAR(50) NOT NULL,  
  
    degreeTitle VARCHAR(50) NOT NULL,  
  
    degreeAcademe VARCHAR(100) NOT NULL,  
  
    PRIMARY KEY (emplUser, degreeTitle, degreeAcademe),  
  
    FOREIGN KEY (emplUser) REFERENCES employee(username)  
  
    ON DELETE CASCADE ON UPDATE CASCADE,  
  
    FOREIGN KEY (degreeTitle, degreeAcademe) REFERENCES degree(title, academe)  
  
    ON DELETE CASCADE ON UPDATE CASCADE  
  
)engine=InnoDB;
```

```
CREATE TABLE project  
  
(  
  
    url VARCHAR(100),
```

```
descr TEXT,  
  
proj_ID TINYINT NOT NULL AUTO_INCREMENT,  
  
empl VARCHAR(50) NOT NULL,  
  
PRIMARY KEY (proj_ID),  
  
FOREIGN KEY (empl) REFERENCES employee(username)  
  
ON DELETE CASCADE ON UPDATE CASCADE  
  
)engine=InnoDB;
```

```
CREATE TABLE evaluator  
(  
  
  IDeval INT NOT NULL,  
  
  eval_usr VARCHAR(50) NOT NULL,  
  
  AFM CHAR(9) NOT NULL,  
  
  PRIMARY KEY (IDeval, eval_usr),  
  
  FOREIGN KEY (eval_usr) REFERENCES user(username)  
  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  
  FOREIGN KEY (AFM) REFERENCES company(AFM)  
  
  ON DELETE CASCADE ON UPDATE CASCADE  
  
)engine=InnoDB;
```

```
CREATE TABLE job  
(  
  
  salary FLOAT(6,1) NOT NULL,  
  
  edra VARCHAR(100) NOT NULL,  
  
  pos_title VARCHAR(40) NOT NULL,  
  
  jobID TINYINT NOT NULL AUTO_INCREMENT,  
  
  announce_date DATE NOT NULL,  
  
  submission_date DATE NOT NULL,  
  
  IDeval INT NOT NULL,  
  
  comp_AFM CHAR(9) NOT NULL,  
  
  PRIMARY KEY (jobID),  
  
  FOREIGN KEY (IDeval) REFERENCES evaluator(IDeval)
```

```
ON DELETE CASCADE ON UPDATE CASCADE,  
FOREIGN KEY (comp_AFM) REFERENCES company(AFM)  
ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

```
CREATE TABLE request_evaluation  
(  
    employee_usr VARCHAR(50) NOT NULL,  
    job_ID TINYINT NOT NULL,  
    AFM CHAR(9) NOT NULL,  
    PRIMARY KEY (employee_usr, job_ID),  
    FOREIGN KEY (employee_usr) REFERENCES employee(username)  
ON DELETE CASCADE ON UPDATE CASCADE,  
    FOREIGN KEY (job_ID) REFERENCES job(jobID)  
ON DELETE CASCADE ON UPDATE CASCADE,  
    FOREIGN KEY (AFM) REFERENCES company(AFM)  
ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

```
CREATE TABLE evaluation_grades  
(  
    phase1 TINYINT,  
    phase2 TINYINT,  
    phase3 TINYINT,  
    IDeval INT NOT NULL,  
    IDmanager INT NOT NULL,  
    report_manager TEXT,  
    job_ID TINYINT NOT NULL,  
    employee_usr VARCHAR(50) NOT NULL,  
    AFM CHAR(9) NOT NULL,  
    PRIMARY KEY (IDeval, IDmanager, job_ID, employee_usr),  
    FOREIGN KEY (job_ID) REFERENCES job(jobID)
```

```

ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (IDeval) REFERENCES evaluator(IDeval)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY(employee_usr) REFERENCES employee(username)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (AFM) REFERENCES company(AFM)
ON DELETE CASCADE ON UPDATE CASCADE
)engine=InnoDB;

```

```

CREATE TABLE evaluation_result
(
    grade TINYINT NOT NULL,
    comments TEXT,
    username VARCHAR(50) NOT NULL,
    IDeval INT NOT NULL,
    jobID TINYINT NOT NULL,
    AFM CHAR(9) NOT NULL,
    PRIMARY KEY (username, IDeval, jobID),
    FOREIGN KEY (username) REFERENCES employee(username)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (IDeval) REFERENCES evaluator(IDeval)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (jobID) REFERENCES job(jobID)
ON DELETE CASCADE ON UPDATE CASCADE,
FOREIGN KEY (AFM) REFERENCES company(AFM)
ON DELETE CASCADE ON UPDATE CASCADE
)engine=InnoDB;

```

```

CREATE TABLE antik
(
    descr TINYTEXT,
    title VARCHAR(80) NOT NULL,
    belongs_to VARCHAR(80),

```



```
PRIMARY KEY (title),  
FOREIGN KEY (belongs_to) REFERENCES antik(title)  
ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

CREATE TABLE needs

```
(  
  jobID TINYINT NOT NULL,  
  title VARCHAR(80) NOT NULL,  
  PRIMARY KEY (jobID, title),  
  FOREIGN KEY (jobID) REFERENCES job(jobID)  
  ON DELETE CASCADE ON UPDATE CASCADE,  
  FOREIGN KEY (title) REFERENCES antik(title)  
  ON DELETE CASCADE ON UPDATE CASCADE  
)engine=InnoDB;
```

CREATE TABLE log_table

```
(  
  username VARCHAR(50) NOT NULL,  
  action_ID INT NOT NULL AUTO_INCREMENT,  
  action_time TIMESTAMP NOT NULL,  
  result ENUM('success', 'failure'),  
  action VARCHAR(30) NOT NULL,  
  table_name VARCHAR(10) NOT NULL,  
  PRIMARY KEY(action_ID)  
)engine=InnoDB;
```

CREATE TABLE tempuser

```
(  
  username VARCHAR(50) NOT NULL,  
  ID TINYINT NOT NULL AUTO_INCREMENT,  
  PRIMARY KEY(ID)  
)engine=InnoDB;
```

Σύνολο εντολών insert:

INSERT INTO company VALUES

```
('24228', 'DOY A PATRWN', 2610555777, 'Karaiskaki', 20, 'Patra', 'Ellada' ),  
( '23558', 'DOY B PATRWN', 2610222444, 'Gounari', 8, 'Patra', 'Ellada'),  
( '28956', 'DOY Z ATHINWN', 2610666321, 'Omonoias', 144, 'Athina', 'Ellada');
```

INSERT INTO user VALUES

```
('xristo@gmail.com', 'idontknow', 'Xristopoulos Panagioths', '2020-12-20'),  
( 'papadop@gmail.com', 'mynameis', 'Papadopoulou Niki', '2020-04-03'),  
( 'kostopoulos@hotmail.com', 'iamfine', 'Kostopoulos Stavros', '2020-02-20'),  
( 'zaxariou@yahoo.com', 'whoisthis', 'Zaxariou Iwanna', '2020-09-25'),  
( 'neofytos@hotmail.com', 'iamaplant', 'Neofytos Kwstas', '2021-01-01'),  
( 'anagnos@gmail.com', 'iamawesome', 'Anagnostopoulos Xristos', '2021-01-03'),  
( 'dimitr@gmail.com', 'lolomgand', 'Dimitriou Vasiliki', '2019-08-23'),  
( 'mpapa@gmail.com', 'ihaveplant', 'Papa Mairh', '2016-03-30'),  
( 'stefanidis@gmail.com', 'youknowwho', 'Stefanidis Marios', '2015-04-12'),  
( 'papadatos@gmail.com', 'yesitis', 'Papadatos Kostas', '2021-01-17');
```

INSERT INTO admin(admin_user) VALUES

```
('Papadatos Kostas');
```

INSERT INTO manager VALUES

```
(10, 'Xristopoulos Panagioths', '24228', 1072333),  
(4, 'Papadopoulou Niki', '23558', 1011420);
```

INSERT INTO evaluator VALUES

```
(1067455, 'Kostopoulos Stavros', '24228'),  
(1069360, 'Zaxariou Iwanna', '23558'),  
(1067460, 'Stefanidis Marios', '24228');
```

```
INSERT INTO job(salary, edra, pos_title, announce_date, submission_date, IDeval, comp_AFM) VALUES
(1500, 'DOY A PATRWN', 'Personnel Director', '2021-01-01', '2021-02-15', 1067455, '24228'),
(2000, 'DOY B PATRWN', 'Marketing Director', '2021-01-01', '2021-02-28', 1069360, '23558'),
(1250, 'DOY A PATRWN', 'Assistant Manager', '2021-03-01', '2021-03-30', 1067460, '24228');
```

```
INSERT INTO degree VALUES
(8, '2011-06-20', 'Degree in Finance', 'University of Patras'),
(8, '2010-02-24', 'Degree in Marketing', 'University of Patras'),
(9, '2012-10-15', 'Degree in Computer Science', 'University of Thessalonikis');
```

```
INSERT INTO employee VALUES
('5 years of experience in the field', 'good in teamwork', 2, 1110, 'degree in computer sciene', 'Neofytos Kwstas', '24228'),
('worked in a big firm for 2.5 years', 'good in organising', 4, 1112, 'degree in Finance', 'Anagnostopoulos Xristos', '24228'),
('first of his class', 'good in PR', 1, 1113, 'good in marketing', 'Dimitriou Vasiliki', '23558'),
('while a student, worked as an intern for a big brand', 'hardworking and a try-hard', 7, 1114, 'degree in computer sciene', 'Papa Mairh', '23558');
```

```
INSERT INTO language VALUES
('English', 'Anagnostopoulos Xristos'),
('English', 'Neofytos Kwstas'),
('French', 'Neofytos Kwstas'),
('German', 'Dimitriou Vasiliki'),
('Chinese', 'Dimitriou Vasiliki');
```

```
INSERT INTO project(url, descr, empl) VALUES
('www.project1.com', 'Database for a hospital', 'Neofytos Kwstas'),
('www.project2.com', 'Create an operating system', 'Papa Mairh');
```

```
INSERT INTO has_degree VALUES
('Neofytos Kwstas', 'Degree in Computer Science', 'University of Thessalonikis'),
('Papa Mairh', 'Degree in Computer Science', 'University of Thessalonikis');
```

```
INSERT INTO request_evaluation VALUES
```

```
('Anagnostopoulos Xristos', 1, '24228'),
```

```
('Neofytos Kwstas', 3, '24228');
```

```
INSERT INTO evaluation_grades VALUES
```

```
(2, 2, 1, 1069360, 1011420, 'very anxious but managed it in the end', 2, 'Dimitriou Vasiliki', '23558'),
```

```
(2, 1, 0, 1069360, 1072333, 'very insufficient', 2, 'Papa Mairh', '23558');
```

```
INSERT INTO evaluation_result VALUES
```

```
(3, 'didnt seem very enthousiastic', 'Anagnostopoulos Xristos', 1067460, 3, '24228'),
```

```
(8, 'very kind', 'Neofytos Kwstas', 1067455, 1, '24228');
```

*Για τους πίνακες request_evaluation, evaluation_grades και evaluation_result ισχύει το εξής: Κάθε φορά που μια αίτηση ενός υπαλλήλου λαμβάνεται υπόψιν και εξετάζεται, διαγράφεται από τον πίνακα request_evaluation και εγγράφεται στον evaluation_grades. Αντίστοιχα, όταν ολοκληρωθεί η αξιολόγηση του υπαλλήλου, γίνεται η διαγραφή του από τον evaluation_grades και η εγγραφή του στον evaluation_result (αυτό πραγματοποιείται μέσω του gui και stored που θα αναπτύξουμε αργότερα).

Ερώτημα 3

3.1

```
DELIMITER $
```

```
CREATE PROCEDURE employee_request(IN user VARCHAR(50))
```

```
BEGIN
```

```
    DECLARE not_found INT;
```

```
    DECLARE c1 INT;
```

```
    DECLARE requests INT;
```

```
    DECLARE number1 INT;
```

DECLARE comm TEXT;

DECLARE name VARCHAR(50);

DECLARE name1 VARCHAR(50);

DECLARE number1 INT;

DECLARE number2 INT;

DECLARE number3 INT;

DECLARE bcursor CURSOR FOR

SELECT job_ID FROM request_evaluation WHERE employee_usr LIKE user;

DECLARE bcursor1 CURSOR FOR

SELECT job_ID FROM evaluation_grades WHERE employee_usr LIKE user;

DECLARE bcursor2 CURSOR FOR

SELECT jobID FROM evaluation_result WHERE username LIKE user;

OPEN bcursor;

BEGIN

DECLARE flag INT DEFAULT 0;

DECLARE requests INT;

DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET flag=1;

bcursorLoop: LOOP

FETCH bcursor INTO requests;

IF (flag)

THEN

LEAVE bcursorLoop;

END IF;

SELECT count(job_ID) INTO c1 FROM request_evaluation WHERE requests = job_ID AND employee_usr = user;

IF (c1>0)

THEN

```

SELECT "The job applications of this user that haven't been examined yet are: ";

SELECT c1;

END IF;

END LOOP;

END;

CLOSE bcursor;


OPEN bcursor1;

BEGIN

DECLARE flag1 INT DEFAULT 0;

DECLARE requests1 INT;

DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET flag1=1;


bcursor1Loop: LOOP
FETCH bcursor1 INTO requests1;
IF (flag1)
THEN
LEAVE bcursor1Loop;
END IF;
SELECT grade INTO number1 FROM evaluation_result WHERE requests=ID AND username LIKE user;
IF (number1 IS NOT NULL)
THEN
SELECT comments INTO comm FROM evaluation_result WHERE requests=ID AND username LIKE user;
SELECT eval_usr INTO name1 FROM evaluation_result WHERE requests=ID AND username LIKE user;
SELECT user, number1, comm, name1;
END IF;
END LOOP;

END;


OPEN bcursor2;

BEGIN

DECLARE flag2 INT DEFAULT 0;

DECLARE requests2 INT;

```

```

DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET flag2=1;

bcursor2Loop: LOOP
  FETCH bcursor2 INTO requests2;
  IF (flag2)
  THEN
    LEAVE bcursor2Loop;
  END IF;

  SELECT employee_usr INTO name1 FROM evaluation_grades WHERE job_ID=requests AND employee_usr
  LIKE user;
  IF (name1 IS NOT NULL)
  THEN
    SELECT 'The evaluation is in progress.';
    SELECT phase1 INTO number1 FROM evaluation_grades WHERE requests=ID AND username LIKE user;
    SELECT phase2 INTO number2 FROM evaluation_grades WHERE requests=ID AND username LIKE user;
    SELECT phase3 INTO number3 FROM evaluation_grades WHERE requests=ID AND username LIKE user;
    SELECT user, number1, number2, number3;
  END IF;
END LOOP;
END;
END$

DELIMITER ;

```

3.2

```

DELIMITER $

CREATE PROCEDURE evaluationComplete(IN jobnumber INT, IN evalnumber INT)
BEGIN

  DECLARE grade1 INT;
  DECLARE grade2 INT;
  DECLARE grade3 INT;

```

```

DECLARE not_found INT;

DECLARE name VARCHAR(50);

DECLARE name1 VARCHAR(50);

DECLARE sum INT;

DECLARE evalname VARCHAR(50);

DECLARE comm VARCHAR(50);

DECLARE report TEXT;

DECLARE af CHAR(9);


DECLARE bcursor CURSOR FOR

SELECT employee_usr FROM evaluation_grades WHERE jobnumber=job_ID AND evalnumber=IDeval;


DECLARE CONTINUE HANDLER FOR NOT FOUND

SET not_found=1;


SET not_found=0;


OPEN bcursor;


REPEAT

FETCH bcursor INTO name;

IF(not_found=0)

THEN

SELECT phase1 INTO grade1 FROM evaluation_grades WHERE jobnumber=job_ID AND evalnumber=IDeval
AND name LIKE employee_usr;

SELECT phase2 INTO grade2 FROM evaluation_grades WHERE jobnumber=job_ID AND evalnumber=IDeval
AND name LIKE employee_usr;

SELECT phase3 INTO grade3 FROM evaluation_grades WHERE jobnumber=job_ID AND evalnumber=IDeval
AND name LIKE employee_usr;

SELECT report_manager INTO report FROM evaluation_grades WHERE jobnumber=job_ID AND
evalnumber=IDeval AND name LIKE employee_usr;

SELECT AFM INTO af FROM evaluation_grades WHERE jobnumber=job_ID AND evalnumber=IDeval AND name
LIKE employee_usr;

```



```

SELECT username INTO name1 FROM evaluation_result WHERE jobnumber=jobID AND username LIKE name;

IF(grade1 IS NOT NULL AND grade2 IS NOT NULL AND grade3 IS NOT NULL AND name1 IS NULL AND report IS
NOT NULL)

THEN

SET sum=grade1+grade2+grade3;

INSERT INTO evaluation_result VALUES

(sum, comm, name, evalnumber, jobnumber, af);

SELECT * FROM evaluation_result;

DELETE FROM evaluation_grades WHERE job_ID = jobnumber AND IDeval = evalnumber AND employee_usr
LIKE name;

ELSE

SELECT "The evaluation has been already completed.";

END IF;

END IF;

UNTIL (not_found=1)

END REPEAT;

CLOSE bcursor;

END$

DELIMITER ;

CALL evaluationComplete(2, 1067455);

```

Παρατηρώντας τα insert στον πίνακα evaluation_grades, η υπάλληλος Παρα Mairh έχει ολοκληρώσει την αξιολόγηση για την θέση εργασίας με ID=2 (μόνο όταν ο βαθμός σε μια φάση ισούται με null, θεωρείται ότι δεν έχει ολοκληρώσει την φάση αυτή), αλλά δεν έχει γίνει η διαγραφή της από τον πίνακα evaluation_grades και η εγγραφή της στον πίνακα evaluation_result.

Καλώντας την παραπάνω stored, παίρνουμε τα εξής αποτελέσματα:

```

+-----+-----+-----+-----+-----+
| grade | comments          | username          | IDeval | jobID |
+-----+-----+-----+-----+-----+
| 3     | didnt seem very enthusiastic | Anagnostopoulos Xristos | 1067460 | 3     |

```

8	very kind	Neofytos Kwstas	1067455	1
3	NULL	Papa Mairh	1067455	2

phase1	phase2	phase3	IDeval	IDmanager	report manager	job ID	employee usr
2	2	1	1069360	1011420	very anxious but managed it in the end	2	Dimitriou Vasiliki

3.3

DELIMITER \$

CREATE PROCEDURE showgrades(IN data INT, OUT result INT)

BEGIN

DECLARE c1 INT;

DECLARE c2 INT;

DECLARE flag INT;

DECLARE jid INT;

DECLARE username VARCHAR(50);

DECLARE grade INT;

DECLARE comments TEXT;

DECLARE bcursor CURSOR FOR

SELECT jobID FROM evaluation_result WHERE data = jobID;

DECLARE CONTINUE HANDLER FOR NOT FOUND

SET flag=1;

SET flag=0;

SET c1=0;

SET c2=0;

SET result=0;

```
SELECT COUNT(username) INTO c1 FROM evaluation_result WHERE jobID = data;
```

```
SELECT COUNT(employee_usr) INTO c2 FROM evaluation_grades WHERE data = job_ID;
```

```
IF(c1>0 AND c2=0)
```

```
THEN
```

```
    OPEN bcursor;
```

```
    REPEAT
```

```
        FETCH bcursor INTO jid;
```

```
        IF (flag=0)
```

```
        THEN
```

```
            SELECT username, grade, comments FROM evaluation_result WHERE jobID = jid;
```

```
        END IF;
```

```
    UNTIL (flag=1)
```

```
    END REPEAT;
```

```
    CLOSE bcursor;
```

```
END IF;
```

```
IF (c1>0 AND c2>0)
```

```
THEN
```

```
    OPEN bcursor;
```

```
    REPEAT
```

```
        FETCH bcursor INTO jid;
```

```
        IF (flag=0)
```

```
        THEN
```

```
            SELECT username, grade, comments FROM evaluation_result WHERE jobID = jid;
```

```
            SELECT c2 INTO result;
```

```
            SELECT result;
```

```
        END IF;
```

```
    UNTIL (flag=1)
```

```
    END REPEAT;
```

```

CLOSE bcursor;

END IF;

IF (c1=0 AND c2>0)
THEN
    SET username = null;
    SET grade = null;
    SET comments = null;
    SELECT c2 INTO result;
    SELECT username, grade, comments;
    SELECT result;
END IF;

END$

DELIMITER ;

```

```
CALL showgrades(2, @result);
```

Τα αποτελέσματα, καλώντας την παραπάνω stored, (λαμβάνω υπόψιν τα insert μετά την κλήση της stored 3.2) είναι τα εξής:

```

+-----+
| Final Table |
+-----+
+-----+-----+-----+-----+-----+
| grade | comments | username | IDeval | jobID |
+-----+-----+-----+-----+-----+
| 3 | NULL | Papa Mairh | 1067455 | 2 |
+-----+-----+-----+-----+-----+

+-----+
| The evaluations in progress are |
+-----+

```

```
+-----+
| result |
+-----+
|    1   |
+-----+
```

Ερώτημα 4

Triggers για τον πίνακα employee:

DELIMITER \$

CREATE TRIGGER employee_insert

AFTER INSERT ON employee

FOR EACH ROW

BEGIN

DECLARE user VARCHAR(50);

DECLARE uid TINYINT;

SELECT MAX(ID) INTO uid FROM tempuser;

SELECT username INTO user FROM tempuser WHERE ID = uid;

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

(user, NOW(), 'success', 'insert', 'employee');

END\$

CREATE TRIGGER employee_update

AFTER UPDATE ON employee

FOR EACH ROW

BEGIN

DECLARE user VARCHAR(50);

DECLARE uid TINYINT;

```
SELECT MAX(ID) INTO uid FROM tempuser;

SELECT username INTO user FROM tempuser WHERE ID = uid;


INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'success', 'update', 'employee');

END$
```

```
CREATE TRIGGER employee_delete
AFTER UPDATE ON employee
FOR EACH ROW
BEGIN
    DECLARE user VARCHAR(50);
    DECLARE uid TINYINT;

    SELECT MAX(ID) INTO uid FROM tempuser;
    SELECT username INTO user FROM tempuser WHERE ID = uid;

    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
    (user, NOW(), 'success', 'delete', 'employee');

END$

DELIMITER ;
```

Triggers για τον πίνακα job:

```
DELIMITER $

CREATE TRIGGER job_insert
AFTER INSERT ON job
FOR EACH ROW
BEGIN
    DECLARE user VARCHAR(50);
```

```
DECLARE uid TINYINT;
```

```
SELECT MAX(ID) INTO uid FROM tempuser;
```

```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES  
(user, NOW(), 'success', 'insert', 'job');
```

```
END$
```

```
CREATE TRIGGER job_update
```

```
AFTER UPDATE ON job
```

```
FOR EACH ROW
```

```
BEGIN
```

```
DECLARE user VARCHAR(50);
```

```
DECLARE uid TINYINT;
```

```
SELECT MAX(ID) INTO uid FROM tempuser;
```

```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES  
(user, NOW(), 'success', 'update', 'job');
```

```
END$
```

```
CREATE TRIGGER job_delete
```

```
AFTER DELETE ON job
```

```
FOR EACH ROW
```

```
BEGIN
```

```
DECLARE user VARCHAR(50);
```

```
DECLARE uid TINYINT;
```

```
SELECT MAX(ID) INTO uid FROM tempuser;
```

```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'success', 'delete', 'job');
END$
```

```
DELIMITER ;
```

Triggers για τον πίνακα request_evaluation:

```
DELIMITER $
```

```
CREATE TRIGGER eval_insert
AFTER INSERT ON request_evaluation
FOR EACH ROW
BEGIN
    DECLARE user VARCHAR(50);
    DECLARE uid TINYINT;

    SELECT MAX(ID) INTO uid FROM tempuser;
    SELECT username INTO user FROM tempuser WHERE ID = uid;

    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
    (user, NOW(), 'success', 'insert', 'request_evaluation');
END$
```

```
CREATE TRIGGER eval_update
AFTER UPDATE ON request_evaluation
FOR EACH ROW
BEGIN
    DECLARE user VARCHAR(50);
    DECLARE uid TINYINT;

    SELECT MAX(ID) INTO uid FROM tempuser;
```



```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES  
(user, NOW(), 'success', 'update', 'request_evaluation');  
END$
```

```
CREATE TRIGGER eval_delete  
AFTER DELETE ON request_evaluation  
FOR EACH ROW  
BEGIN  
    DECLARE user VARCHAR(50);  
    DECLARE uid TINYINT;
```

```
    SELECT MAX(ID) INTO uid FROM tempuser;  
    SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES  
    (user, NOW(), 'success', 'delete', 'request_evaluation');  
END$
```

```
DELIMITER ;
```

Trigger για τον πίνακα company:

```
DELIMITER $  
CREATE TRIGGER update_company  
BEFORE UPDATE ON company  
FOR EACH ROW  
BEGIN  
    SET NEW.AFM=OLD.AFM;  
    SET NEW.DOY=OLD.DOY;  
END$
```

```
CREATE TRIGGER date
BEFORE INSERT ON user
FOR EACH ROW
BEGIN

    DECLARE currDate DATE;

    SET currDate = CURDATE();

    SET NEW.register_date = currDate;

END$

DELIMITER ;
```

Trigger για κάθε καινούργια εισαγωγή μιας θέσης εργασίας, το γνώρισμα announce_date ενημερώνεται αυτόματα με την εκάστοτε ημερομηνία:

```
DELIMITER $

CREATE TRIGGER date1
BEFORE INSERT ON job
FOR EACH ROW
BEGIN

    DECLARE currDate DATE;

    SET currDate = CURDATE();

    SET NEW.announce_date = currDate;

END$

DELIMITER ;
```

Triggers, τα οποία για κάθε αλλαγή γνωρισμάτων στους πίνακες χρηστών του συστήματος ενημερώνουν τον log πίνακα και ελέγχουν, αν η ενημέρωση μπορεί να γίνει ανάλογα με τα δικαιώματα που έχει ο κάθε χρήστης.

DELIMITER \$

CREATE TRIGGER change_user

BEFORE UPDATE ON user

FOR EACH ROW

BEGIN

DECLARE u VARCHAR(50);

DECLARE name1 VARCHAR(50);

DECLARE name2 VARCHAR(50);

DECLARE name3 VARCHAR(50);

DECLARE uid TINYINT;

SELECT MAX(ID) INTO uid FROM tempuser;

SELECT username INTO u FROM tempuser WHERE ID = uid;

SELECT username INTO name1 FROM employee WHERE username LIKE u;

SELECT admin_user INTO name2 FROM admin WHERE admin_user LIKE u;

SELECT managerUsername INTO name3 FROM manager WHERE managerUsername LIKE u;

IF (name3 IS NOT NULL)

THEN

SET NEW.username=OLD.username;

SET NEW.register_date=OLD.register_date;

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

(u, NOW(), 'success', 'update', 'user');

END IF;

IF (name1 IS NOT NULL)

THEN

SET NEW.username=OLD.username;

SET NEW.email=OLD.email;

```
SET NEW.register_date=OLD.register_date;

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

    (u, NOW(), 'success', 'update', 'user');

END IF;
```

```
IF (name2 IS NOT NULL)

THEN

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

(u, NOW(), 'success', 'update', 'user');

END IF;
```

```
IF (name2 IS NULL AND name1 IS NULL AND name3 IS NULL)

THEN

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

(u, NOW(), 'failure', 'update', 'user');

END IF;
```

```
END$
```

```
CREATE TRIGGER change_employee

BEFORE UPDATE ON employee

FOR EACH ROW

BEGIN
```

```
    DECLARE c INT;

    DECLARE name1 VARCHAR(50);

    DECLARE name2 VARCHAR(50);

    DECLARE name3 VARCHAR(50);

    DECLARE user VARCHAR(50);

    DECLARE uid TINYINT;
```

```
SELECT MAX(ID) INTO uid FROM tempuser;
```

```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
SELECT admin_user INTO name1 FROM admin WHERE admin_user LIKE user;
```

```
SELECT username INTO name2 FROM employee WHERE username LIKE user;
```

```
SELECT managerUsername INTO name3 FROM manager WHERE managerUsername LIKE user;
```

```
IF (name1 IS NOT NULL)
```

```
THEN
```

```
    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
```

```
    (user, NOW(), 'success', 'update', 'employee');
```

```
END IF;
```

```
IF (name2 IS NOT NULL)
```

```
THEN
```

```
    SET NEW.username=OLD.username;
```

```
    SET NEW.excellence=OLD.excellence;
```

```
    SET NEW.work_experience=OLD.work_experience;
```

```
    SET NEW.AM=OLD.AM;
```

```
    SET NEW.certificate=OLD.certificate;
```

```
    SET NEW.comp_AFM=OLD.comp_AFM;
```

```
    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
```

```
(user, NOW(), 'success', 'update', 'employee');
```

```
END IF;
```

```
IF (name3 IS NOT NULL)
```

```
THEN
```

```
SET NEW.username=OLD.username;
```

```
SET NEW.bio=OLD.bio;
```

```
SET NEW.AM=OLD.AM;
```

```
    SET NEW.comp_AFM=OLD.comp_AFM;
```

```
    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
```

```
(user, NOW(), 'success', 'update', 'employee');
```

END IF;

IF (name1 IS NULL AND name2 IS NULL AND name3 IS NULL)

THEN

SIGNAL SQLSTATE VALUE '45000'

SET MESSAGE_TEXT = 'Invalid Registration';

INSERT INTO log(action_name, action_time, result, action, table_name) VALUES

(user, NOW(), 'failure', 'update', 'employee');

END IF;

END\$

CREATE TRIGGER change_evaluator

BEFORE UPDATE ON evaluator

FOR EACH ROW

BEGIN

DECLARE c INT;

DECLARE name1 VARCHAR(50);

DECLARE name2 VARCHAR(50);

DECLARE user VARCHAR(50);

DECLARE uid TINYINT;

SELECT MAX(ID) INTO uid FROM tempuser;

SELECT username INTO user FROM tempuser WHERE ID = uid;

SELECT admin_user INTO name1 FROM admin WHERE admin_user LIKE user;

SELECT eval_usr INTO name2 FROM evaluator WHERE eval_usr LIKE user;

IF (name2 IS NOT NULL)

THEN

SET NEW.eval_usr=OLD.eval_usr;

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'success', 'update', 'evaluator');
END IF;
```

```
IF (name1 IS NOT NULL)
THEN
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'success', 'update', 'evaluator');
END IF;
```

```
IF (name1 IS NULL AND name2 IS NULL)
THEN
SIGNAL SQLSTATE VALUE '45000'
SET MESSAGE_TEXT = 'Invalid Registration';
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'failure', 'update', 'evaluator');
END IF;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
(user, NOW(), 'success', 'update', 'evaluator');
END$
```

```
CREATE TRIGGER change_manager
BEFORE UPDATE ON manager
FOR EACH ROW
BEGIN
```

```
DECLARE c INT;
DECLARE name1 VARCHAR(50);
DECLARE user VARCHAR(50);
DECLARE uid TINYINT;
```

```
SELECT MAX(ID) INTO uid FROM tempuser;
```

```
SELECT username INTO user FROM tempuser WHERE ID = uid;
```

```
SELECT admin_user INTO name1 FROM admin WHERE admin_user LIKE user;
```

```
IF (name1 IS NULL)
```

```
THEN
```

```
    SIGNAL SQLSTATE VALUE '45000'
```

```
    SET MESSAGE_TEXT = 'Invalid Registration';
```

```
    INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
```

```
    (user, NOW(), 'failure', 'update', 'manager');
```

```
END IF;
```

```
INSERT INTO log(action_name, action_time, result, action, table_name) VALUES
```

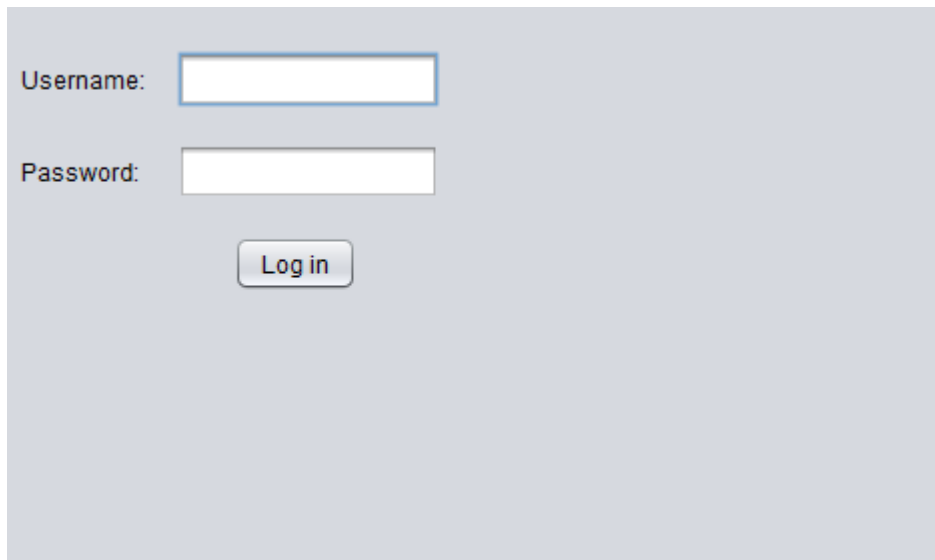
```
(user, NOW(), 'success', 'update', 'manager');
```

```
END$
```

```
DELIMITER ;
```

Μέρος B: GUIs

Το gui, αρχικά, ξεκινάει με το login page:

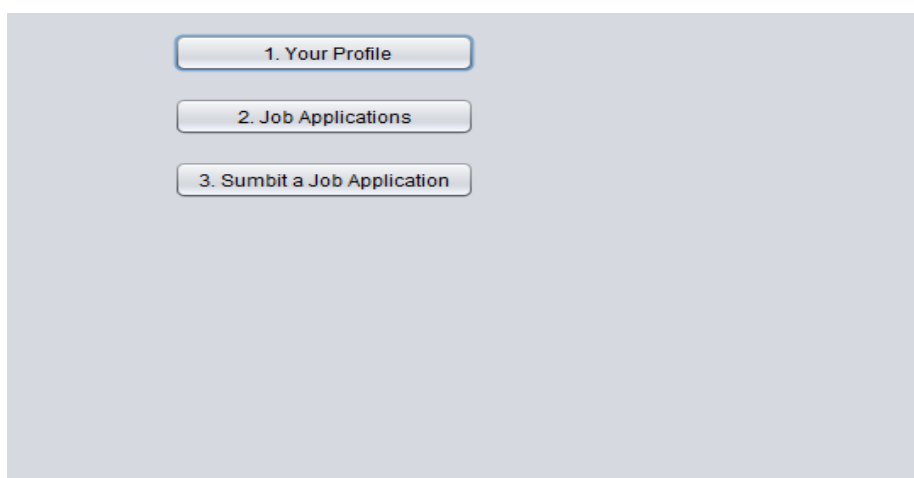


A login form on a light gray background. It contains two text input fields: the first is labeled 'Username:' and the second is labeled 'Password:'. Below the password field is a button labeled 'Log in'.

Ο χρήστης εισάγει το username του (Lastname Firstname) και τον αντίστοιχο κωδικό του (password) και ανάλογα με την ιδιότητα του θα εμφανιστεί το αντίστοιχο menu. Σε περίπτωση λάθους (το όνομα και ο κωδικός δεν αντιστοιχίζεται σε κανέναν χρήστη του συστήματος), εμφανίζεται το κατάλληλο μήνυμα και ο χρήστης ξανά προσπαθεί.

*Σε κάθε παράθυρο υπάρχει το κουμπί back, το οποίο επιστρέφει τον χρήστη στο προηγούμενο παράθυρο που βρισκόταν και το κουμπί exit, που εκτελεί την έξοδο του από το πρόγραμμα.

MenuEmployee



A menu for employees on a light gray background. It contains three buttons stacked vertically: '1. Your Profile', '2. Job Applications', and '3. Submit a Job Application'.

Χρησιμοποιούμε τα εξής credentials για καλύτερη κατανόηση της λειτουργίας του gui: Username: Neofytos Kwstas, Password: iamaplant

1. Your profile

The screenshot shows a web application window titled "Your profile". At the top, there is a table with the following data:

Employee Username	Bio	Excellence	Work Experience	AIM	Certificate	Company AFM
Neofytos Kwstas	5 years of experience in the field	good in teamwork	2	1110	degree in computer sciene	24228

Below the table, there is a "Show info" button. To the left, there is a "Back" button and a section for changing the password with labels "Change your password:", "Enter your password:", and "Re-enter your password:". There is a "Submit" button at the bottom of this section. To the right, there is a "See your languages" button and a section for changing the bio with a label "Change your Bio:" and a large text input area. There is an "Enter" button at the bottom of this section. An "Exit" button is located in the top right corner.

Below the main form, there is a separate window showing a list of languages: "English" and "French". There is a "Back" button and a "Show Info" button.

Μπορεί να δει τα στοιχεία του, πατώντας το κουμπί show info. Επίσης, μπορεί να δει τις γλώσσες που έχει δηλώσει πατώντας το κουμπί see your languages και έπειτα το show info.

Τέλος, μπορεί να αλλάξει τον κωδικό πρόσβασης και το βιογραφικό του (με την υποβολή του βιογραφικού του, ο πίνακας ανανεώνεται αυτόματα και εμφανίζεται πλέον με το πεδίο bio ενημερωμένο).

2. Job Applications

The screenshot shows a web application window with a light gray background. At the top left is a small icon. At the top right are standard window controls (minimize, maximize, close). On the left side, there is a button labeled "Show Data". To the right of this button is a table with two columns: "Employee Username" and "Job ID". The table contains one row with the values "Neofytos Kwstas" and "3". Below the table, there is a text prompt "Type the ID of the job application you want to withdraw:" followed by a text input field. Below the input field is an "Enter" button. At the bottom left is a "Back" button, and at the bottom right is an "Exit" button.

Employee Username	Job ID
Neofytos Kwstas	3

Show Data

Type the ID of the job application you want to withdraw:

Enter

Back

Exit

Πατώντας το show Data, εμφανίζονται στον πίνακα οι αιτήσεις που έχει υποβάλλει για κάποια/ες θέση/εις εργασίας, μαζί με τον μοναδικό κωδικό της κάθε θέσης. Εάν επιθυμεί, μπορεί να αποσύρει κάποια υποβολή, εφόσον η εκάστοτε “σημερινή ημερομηνία” είναι μεταξύ της `announce_date` και `submission_date`. Επίσης, δεν μπορεί να αποσύρει κάποια υποβολή, εφόσον έχει ξεκινήσει ήδη η αξιολόγηση της (δηλαδή έχει πραγματοποιηθεί η εισαγωγή της στον πίνακα `evaluation_grades`), λόγω του τρόπου οργάνωσης των πινάκων `request_evaluation`, `evaluation_grades` και `evaluation_result` που έχει εξηγηθεί παραπάνω.

3. Submit a job application

The screenshot shows a web application window with a table of job listings and a form below it. The table has columns: Job ID, Salary, Edra, Position Title, Announce ..., Submissio..., Evaluator ID, and Company A... The table contains two rows of data. Below the table is a button labeled 'Show available jobs'. Underneath the button is a text prompt 'Fill the Job ID, you want to submit application for:' followed by a text input field and an 'Enter' button. At the bottom left is a 'Back' button and at the bottom right is an 'Exit' button.

Job ID	Salary	Edra	Position Title	Announce ...	Submissio...	Evaluator ID	Company A...
1	1500.0	DOY A PAT...	Personnel ...	2021-01-01	2021-02-15	1067455	24228
3	1250.0	DOY A PAT...	Assistant M...	2021-03-01	2021-03-30	1067460	24228

Show available jobs

Fill the Job ID, you want to submit application for:

Enter

Back Exit

Πατώντας το κουμπί Show available jobs, εμφανίζονται στον πίνακα οι διαθέσιμες θέσεις εργασίας (όπου $\text{currDate} < \text{submission_date}$), για τις οποίες μπορεί ο υπάλληλος να υποβάλλει αίτηση. Πατώντας το enter, αφού γίνει έλεγχος για το job ID που έχει δώσει (αν υπάρχει η συγκεκριμένη θέση εργασίας στην εταιρία που εργάζεται, ισχύει $\text{announce_date} < \text{currDate} < \text{submission_date}$ και αν δεν έχει ήδη υποβάλλει αίτηση για αυτή την θέση), τότε ενημερώνεται ο πίνακας request_evaluation.

MenuAdmin

A window with a title bar containing a small icon and standard window controls (minimize, maximize, close). The main area has a light gray background. In the center, there are six buttons stacked vertically, each with a number and a label: "1. Create an Employee", "2. Create a Manager", "3. Create an Evaluuator", "4. Create a company", "5. Insert new antik", and "6. Log table". At the bottom left is a "Back" button, and at the bottom right is an "Exit" button.

1. Create an Employee

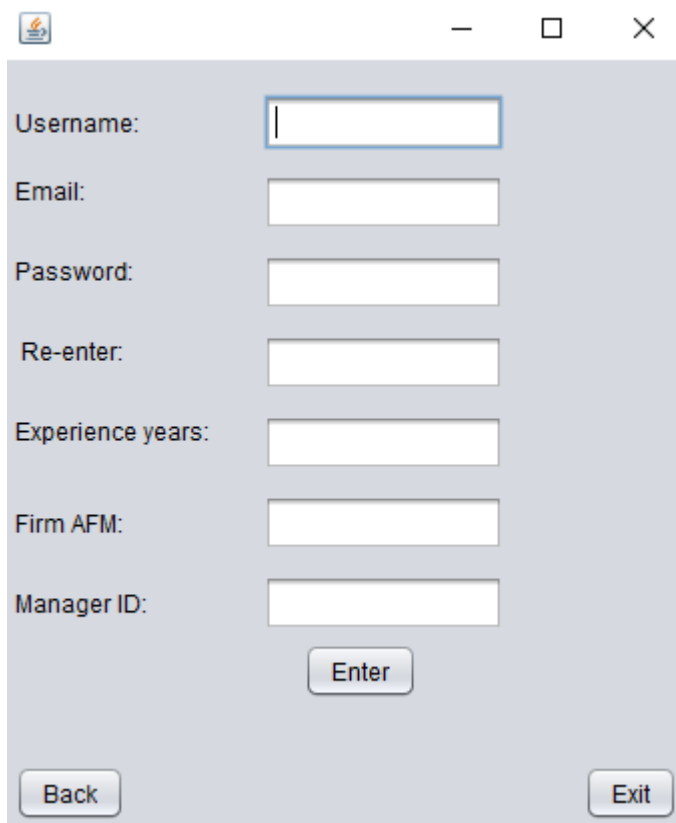
A window with a title bar containing a small icon and standard window controls (minimize, maximize, close). The main area has a light gray background. It contains several input fields and buttons. On the left side, there are four labels with corresponding text boxes: "Username:", "Work Experien...", "Employee AM:", and "Company AFM:". On the right side, there are three labels with corresponding text boxes: "Email:", "Password:", and "Re-enter:". Below the "Work Experien..." field is a "Enter Bio:" label followed by a text box with a vertical scrollbar. Below that is a "Enter Experience:" label followed by a text box with a vertical scrollbar. At the bottom left is a "Enter certificat..." label followed by a text box with a vertical scrollbar. On the right side, there are two more labels with corresponding text boxes: "Username:" and "Language:". There are three "Enter" buttons: one to the right of the "Language:" field, one below the "Enter certificat..." field, and one at the bottom center. At the bottom left is a "Back" button, and at the bottom right is an "Exit" button.

Πραγματοποιείται η δημιουργία ενός employee με τα αντίστοιχα πεδία (insert στον πίνακα employee και user), καθώς και εισαγωγή των γλωσσών που γνωρίζει στον πίνακα language (γίνεται έλεγχος, αν υπάρχει το όνομα του employee που καταχωρεί ο admin, πριν γίνει το insert, ώστε να μην γίνει κάποιο λάθος στην ΒΔ).

Γίνεται έλεγχος, αν υπάρχει στην ΒΔ εταιρεία με το afm που δίνει ο admin.

Γίνεται έλεγχος, αν το email που δίνει, είναι του τύπου [%@%.com](#).

2. Create a manager



The image shows a web form titled "Create a manager" with a light blue background. The form contains the following fields and buttons:

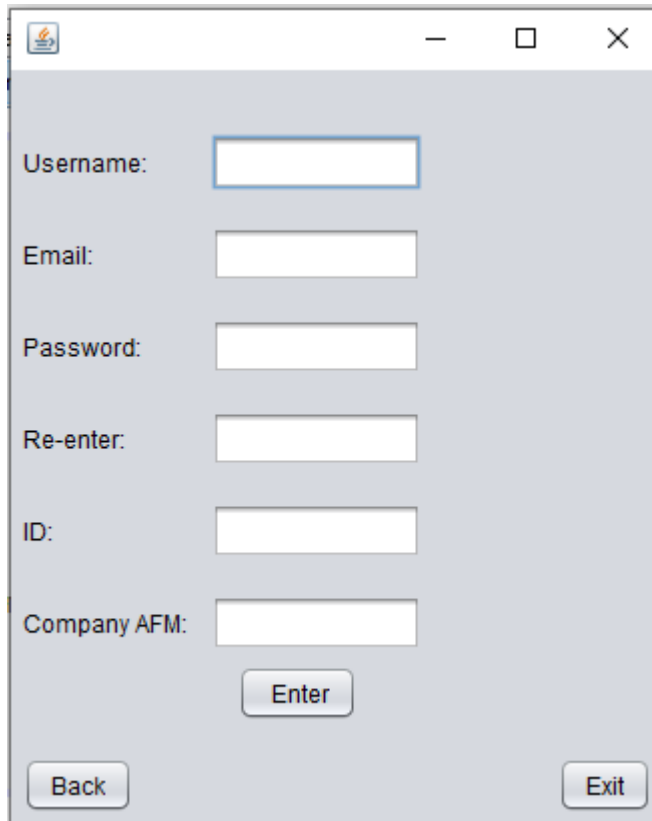
- Username:** A text input field with a blue border.
- Email:** A text input field.
- Password:** A text input field.
- Re-enter:** A text input field.
- Experience years:** A text input field.
- Firm AFM:** A text input field.
- Manager ID:** A text input field.
- Enter:** A button located below the Manager ID field.
- Back:** A button located at the bottom left.
- Exit:** A button located at the bottom right.

Πραγματοποιείται η δημιουργία ενός manager (insert στον πίνακα user και manager).

Γίνεται έλεγχος, αν υπάρχει στην ΒΔ εταιρεία με το afm που δίνει ο admin.

Γίνεται έλεγχος, αν το email που δίνει, είναι του τύπου [%@%.com](#).

3. Create an Evaluator



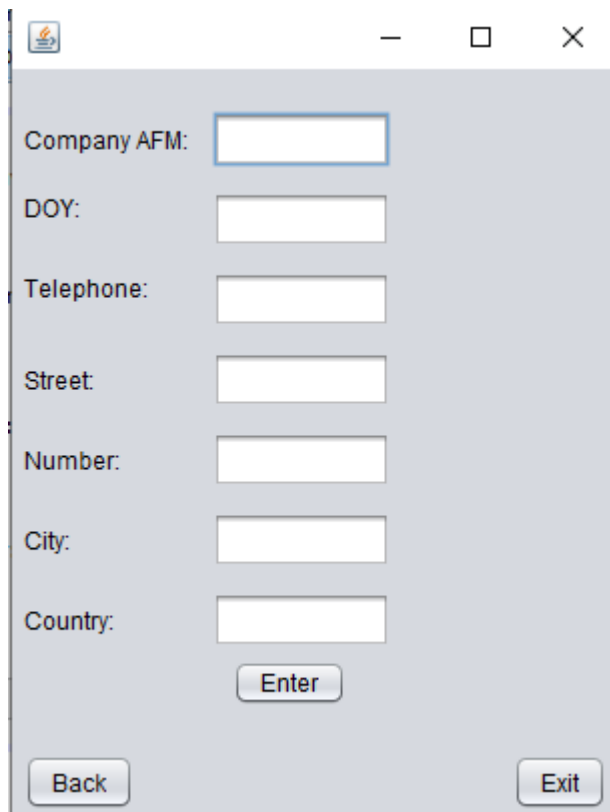
A screenshot of a software window titled "Create an Evaluator". The window has a standard Windows-style title bar with a minimize button, a maximize button, and a close button. The main area of the window is light gray and contains several input fields and buttons. The input fields are labeled "Username:", "Email:", "Password:", "Re-enter:", "ID:", and "Company AFM:". Each label is followed by a white rectangular input box. Below the "Company AFM:" field is a button labeled "Enter". At the bottom left of the window is a button labeled "Back", and at the bottom right is a button labeled "Exit".

Πραγματοποιείται η δημιουργία ενός evaluator (insert στον πίνακα user και evaluator).

Γίνεται έλεγχος, αν υπάρχει στην ΒΔ εταιρεία με το afm που δίνει ο admin.

Γίνεται έλεγχος, αν το email που δίνει, είναι του τύπου [%@%.com](#).

4. Create a company

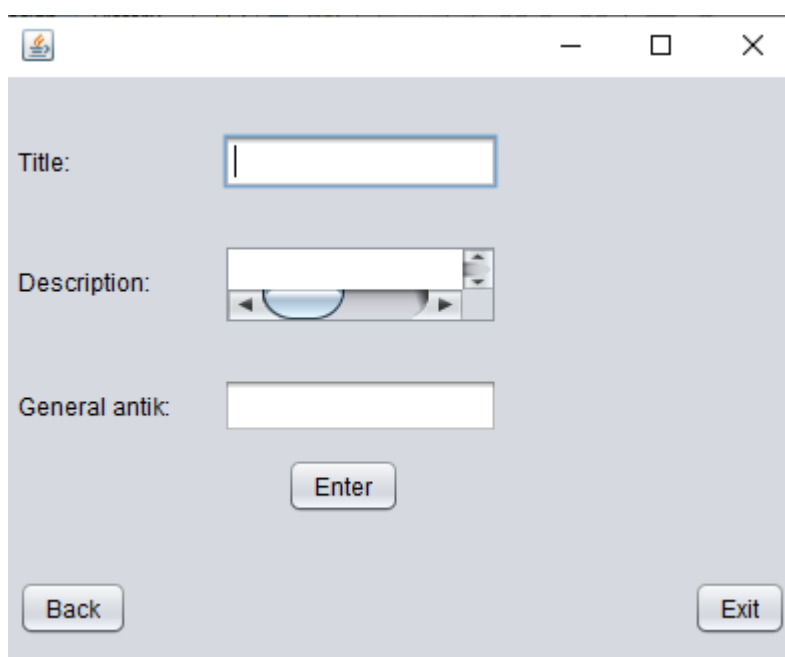


A screenshot of a Windows application window with a light gray background. The window has a standard title bar with a minimize button, a maximize button, and a close button. The main area contains several text input fields and buttons. The labels and fields are as follows:

- Company AFM: [text input field]
- DOY: [text input field]
- Telephone: [text input field]
- Street: [text input field]
- Number: [text input field]
- City: [text input field]
- Country: [text input field]
- [Enter button]
- [Back button]
- [Exit button]

Πραγματοποιείται η δημιουργία μιας εταιρείας (insert στον πίνακα company).

5. Insert new antik

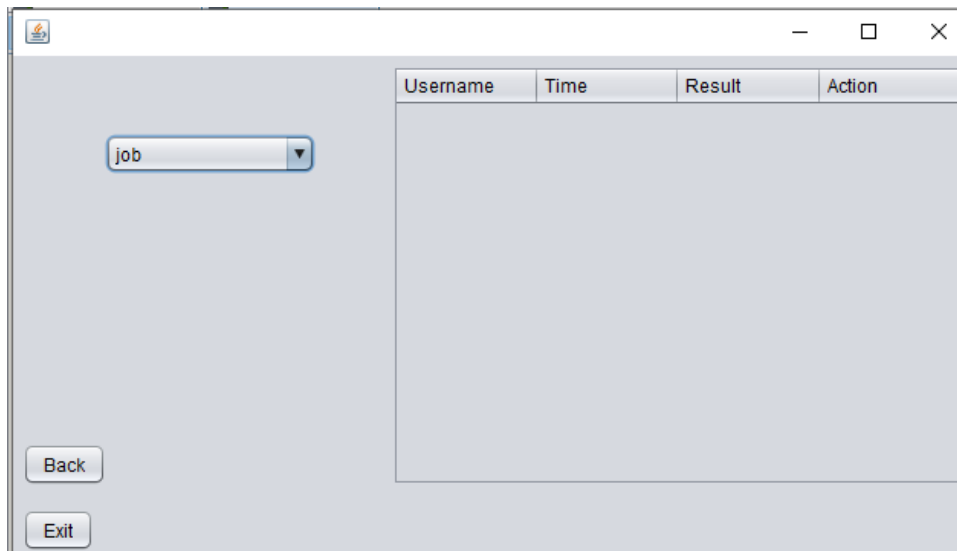


A screenshot of a Windows application window with a light gray background. The window has a standard title bar with a minimize button, a maximize button, and a close button. The main area contains several text input fields and buttons. The labels and fields are as follows:

- Title: [text input field]
- Description: [text input field with a scrollbar]
- General antik: [text input field]
- [Enter button]
- [Back button]
- [Exit button]

Δημιουργία ενός αντικειμένου (insert στον πίνακα antik).

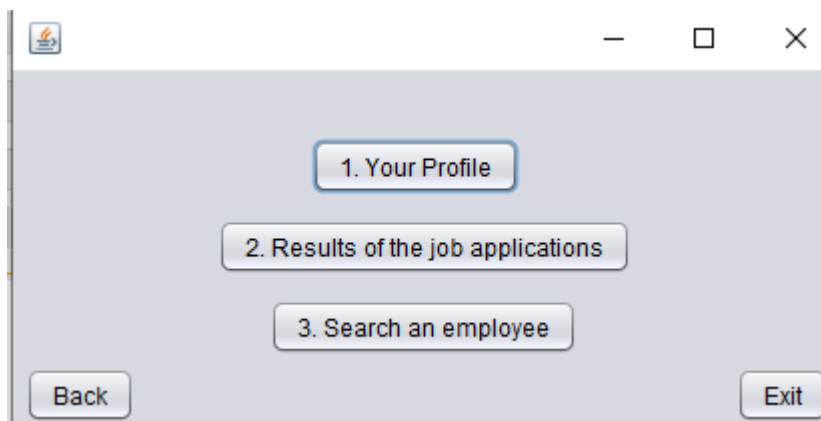
6. Log table



The screenshot shows a web application window with a light gray background. On the left side, there is a dropdown menu with the text 'job' and a downward arrow. Below the dropdown menu are two buttons: 'Back' and 'Exit'. On the right side, there is a table with four columns: 'Username', 'Time', 'Result', and 'Action'. The table is currently empty.

Επιλέγει ο admin τον πίνακα που θέλει να δει τις ενέργειες που έχουν συμβεί σ' αυτόν (μέσω του jComboBox) και οι τελευταίες εμφανίζονται στον πίνακα δεξιά.

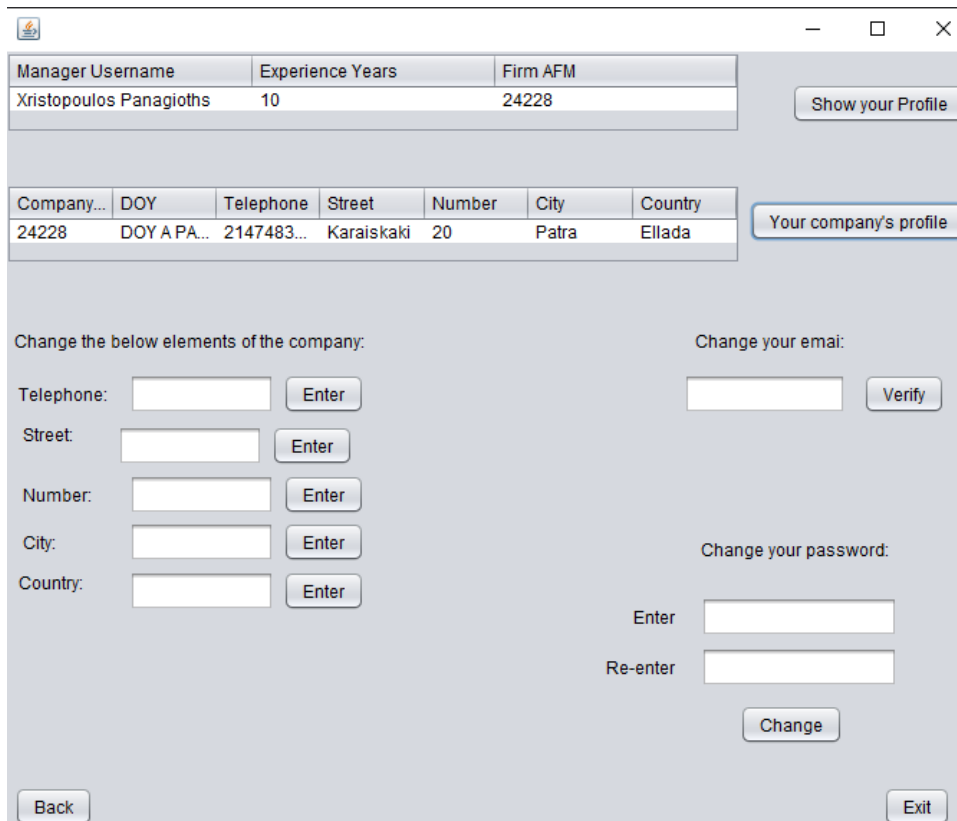
MenuManager



The screenshot shows a web application window with a light gray background. In the center, there are three buttons stacked vertically: '1. Your Profile', '2. Results of the job applications', and '3. Search an employee'. At the bottom left, there is a 'Back' button, and at the bottom right, there is an 'Exit' button.

Χρησιμοποιούμε τα εξής credentials για καλύτερη κατανόηση της λειτουργίας του gui: Username: Xristopoulos Panagioths, Password: idontknow

1. Your Profile



The screenshot shows a window titled "Your Profile" with a light blue background. At the top, there are two tables. The first table displays user information: Manager Username (Xristopoulos Panagioths), Experience Years (10), and Firm AFM (24228). To the right of this table is a button labeled "Show your Profile". Below this is a second table displaying company information: Company... (24228), DOY (DOY A PA...), Telephone (2147483...), Street (Karaiskaki), Number (20), City (Patra), and Country (Ellada). To the right of this table is a button labeled "Your company's profile". Below the tables, there are two sections for editing information. The left section is titled "Change the below elements of the company:" and contains five rows, each with a label (Telephone, Street, Number, City, Country), an input field, and an "Enter" button. The right section is titled "Change your email:" and contains one row with an input field and a "Verify" button. Below these sections, there is another section titled "Change your password:" which contains two rows: "Enter" and "Re-enter", each with an input field, and a "Change" button at the bottom. At the very bottom of the window, there are two buttons: "Back" on the left and "Exit" on the right.

Manager Username	Experience Years	Firm AFM
Xristopoulos Panagioths	10	24228

Show your Profile

Company...	DOY	Telephone	Street	Number	City	Country
24228	DOY A PA...	2147483...	Karaiskaki	20	Patra	Ellada

Your company's profile

Change the below elements of the company:

Telephone: Enter

Street: Enter

Number: Enter

City: Enter

Country: Enter

Change your email:

Verify

Change your password:

Enter

Re-enter

Change

Back Exit

Μπορεί να βλέπει το προφίλ του, καθώς και το προφίλ της εταιρείας στην οποία ανήκει (πατώντας τα κατάλληλα κουμπιά).

Μπορεί να αλλάξει κάποια από τα στοιχεία της εταιρείας (ο πίνακας ανανεώνεται αυτόματα και εμφανίζεται με τα πεδία ενημερωμένα).

Επίσης, μπορεί να αλλάξει και τον κωδικό πρόσβασής του.

2. Results of the job applications

Employee Username	Evaluator ID	Job ID	Grade	Comments
Anagnostopoulos Xr...	1067460	3	3	didnt seem very ent...
Neofytos Kwstas	1067455	1	8	very kind

Click to see the evaluation grades

Job ID	Salary	Edra	Position Title	Announce Da...	Submission ...	Evaluator ID
1	1500.0	DOY A PATR...	Personnel Di...	2021-01-01	2021-02-15	1067455

View the available jobs

Change the salary of the available jobs:

Salary:

Job ID:

Enter

Enter your report for a job evaluation of an employee:

Employee User:

Job ID:

Submit

Back

Exit

Ο manager μπορεί να δει τις τελικές αξιολογήσεις (μέσω του πίνακα evaluation_result).

Βλέπει τις διαθέσιμες θέσεις εργασίας της εταιρείας που δουλεύει. Επίσης, μπορεί να αλλάξει και το μισθό για κάποια από αυτές (πριν γίνει το update, ελέγχεται, αν το job ID, που έχει δώσει υπάρχει και ανήκει στην εταιρεία του, ενώ γίνεται και έλεγχος της ημερομηνίας).

Τέλος, μπορεί να κάνει submit το report για κάποιον υπάλληλο που έχει υποβάλλει αίτηση για κάποια θέση εργασίας και βρίσκεται στον πίνακα evaluation_grades. Φυσικά, πριν γίνει το κατάλληλο insert, πραγματοποιούνται οι απαραίτητοι περιορισμοί(δηλ. Αν υπάρχει το όνομα του employee, αν υπάρχει το job ID και ανήκει στην εταιρεία του κλπ).

Μετά το παραπάνω update, καλώ την stored2, ώστε σε περίπτωση που όλα τα πεδία έχουν συμπληρωθεί, η υποβολή να μεταφερθεί στον πίνακα evaluation_result και να διαγραφθεί από τον evaluation_grades.

3. Search an employee

Username	Bio	Excellence	Work exper...	AM	Certificate	Company ...
Anagnosto...	worked in ...	good in org...	4	1112	degree in F...	24228
Neofytos K...	5 years of ...	good in tea...	2	1110	degree in c...	24228

Show Info

Change the Excellence of the Employee:

Change the Certificate of the Employee:

Employee: Enter

Employee: Enter

Give the name of the employee, you want to see his/her evaluation progress:

Insert

Employee Username	Job ID
Neofytos Kwstas	3

Employee's Job Applications

Employee U...	Grade	Comments	Evaluator Us...	Job ID
Neofytos Kw...	8	very kind	Kostopoulos...	1

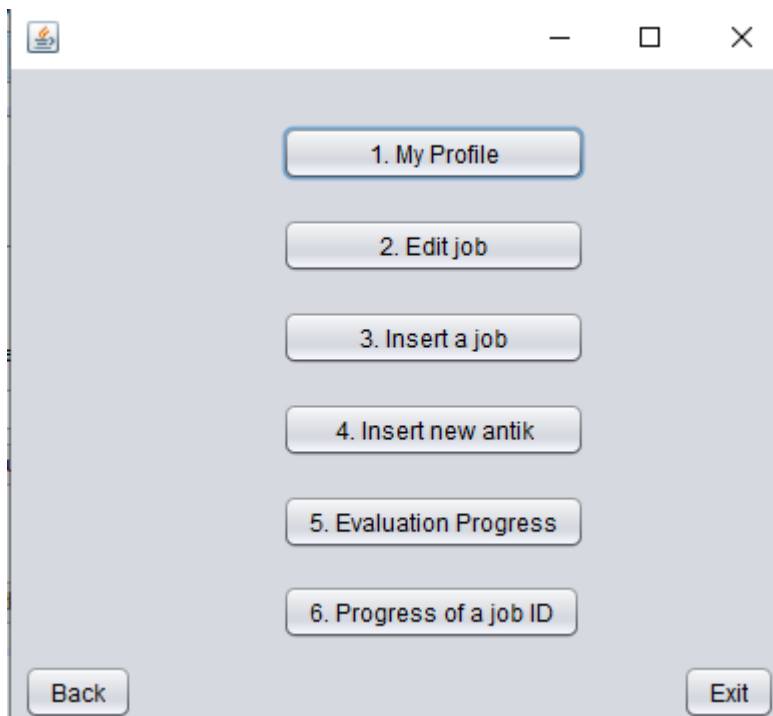
Employee's finalized Job Applications

Back Exit

Βλέπει το προφίλ όλων των υπαλλήλων που δουλεύουν στην ίδια εταιρεία με τον manager. Μπορεί να τροποποιήσει τις πιστοποιήσεις και διακρίσεις που έχει λάβει ο υπάλληλος κατά την διάρκεια της εργασίας του στην εταιρεία (γίνεται έλεγχος, αν υπάρχει το όνομα του υπαλλήλου).

Επίσης, μπορεί να δώσει ένα όνομα ενός υπαλλήλου και βλέπει τις αιτήσεις που έχει κάνει ο τελευταίος αλλά δεν έχουν ακόμα μπει στην διαδικασία της αξιολόγησης (πίνακας request_evaluation) και τις αιτήσεις που έχουν αξιολογηθεί (evaluation_result).

MenuEvaluator



Χρησιμοποιούμε τα εξής credentials για καλύτερη κατανόηση της λειτουργίας του gui: Username: Kostopoulos Stavros, Password: iamfine

1. My profile

A screenshot of the "My Profile" window. At the top, there is a table with three columns: "ID", "Username", and "Company AFM". The table contains one row with the values "1067455", "Kostopoulos Stavros", and "24228". To the right of the table is a "Show info" button. Below the table, there are three sections for changing user information:

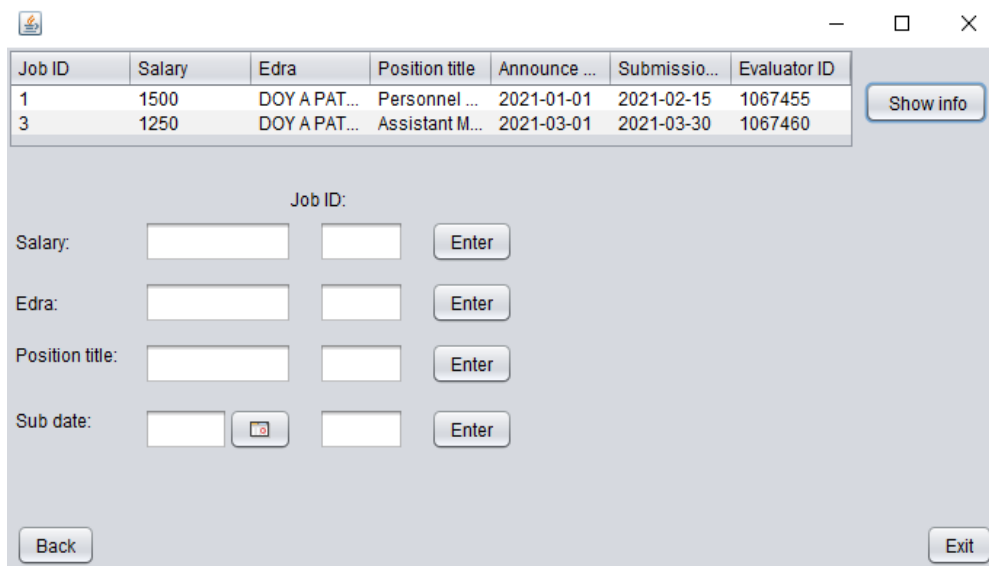
- Change your email:** A text input field followed by a "Submit" button.
- Change Your Password:** Two text input fields, the second labeled "(re-enter)", followed by a "Submit" button.
- Change the AFM to your current company:** A text input field followed by an "Enter" button.

At the bottom left is a "Back" button, and at the bottom right is an "Exit" button.

ID	Username	Company AFM
1067455	Kostopoulos Stavros	24228

Μπορεί να δει το προφίλ του, να αλλάξει το email του (γίνεται έλεγχος, αν το email που εισάγει, είναι της μορφής %@%.com), τον κωδικό πρόσβασής του και το AFM της εταιρείας που δουλεύει.

2. Edit job



The screenshot shows a window titled "Edit job" with a table of jobs and a form to edit a selected job. The table has columns: Job ID, Salary, Edra, Position title, Announce ..., Submissio..., and Evaluator ID. The form below the table has fields for Job ID, Salary, Edra, Position title, and Sub date, each with an "Enter" button. There are also "Back" and "Exit" buttons at the bottom.

Job ID	Salary	Edra	Position title	Announce ...	Submissio...	Evaluator ID
1	1500	DOY A PAT...	Personnel ...	2021-01-01	2021-02-15	1067455
3	1250	DOY A PAT...	Assistant M...	2021-03-01	2021-03-30	1067460

Job ID:

Salary:

Edra:

Position title:

Sub date:

Βλέπει όλες τις διαθέσιμες θέσεις εργασίας που ανήκουν στην εταιρεία που εργάζεται. Μπορεί για κάθε μία από αυτές να αλλάξει τον μισθό, την έδρα, τον τίτλο και την ημερομηνία λήξης.

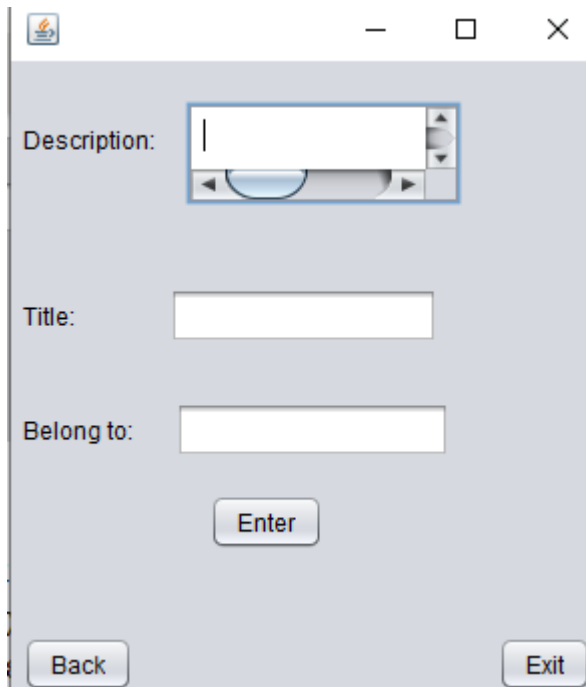
Γίνονται οι κατάλληλοι έλεγχοι κάθε φορά που εισάγει ένα job ID.

3. Insert a Job

The screenshot shows a software window with a light gray background. At the top, there is a standard Windows title bar with a small icon on the left and three control buttons (minimize, maximize, close) on the right. The main area of the window contains two sections. The first section has four labels: 'Salary:', 'Edra:', 'Position title:', and 'Sub date:'. Each label is followed by a white text input field. The 'Salary' field is highlighted with a blue border. To the right of the 'Sub date' field is a small button with a calendar icon. Below these fields is a rounded 'Enter' button. The second section is titled 'Match an antik to a job:'. It contains two labels: 'Job ID:' and 'Title:', each followed by a white text input field. Below these fields is another rounded 'Enter' button. At the bottom left of the window is a rounded 'Back' button, and at the bottom right is a rounded 'Exit' button.

Μπορεί να εισάγει μια θέση εργασίας και να αντιστοιχίσει κάποιο αντικείμενο που χρειάζεται γι' αυτήν. Όσον αφορά στην αντιστοίχιση, γίνονται οι κατάλληλοι έλεγχοι (δηλ. Αν υπάρχει ο τίτλος που εισάγει, το Job ID κλπ).

4. Insert new antik



Description:

Title:

Belong to:

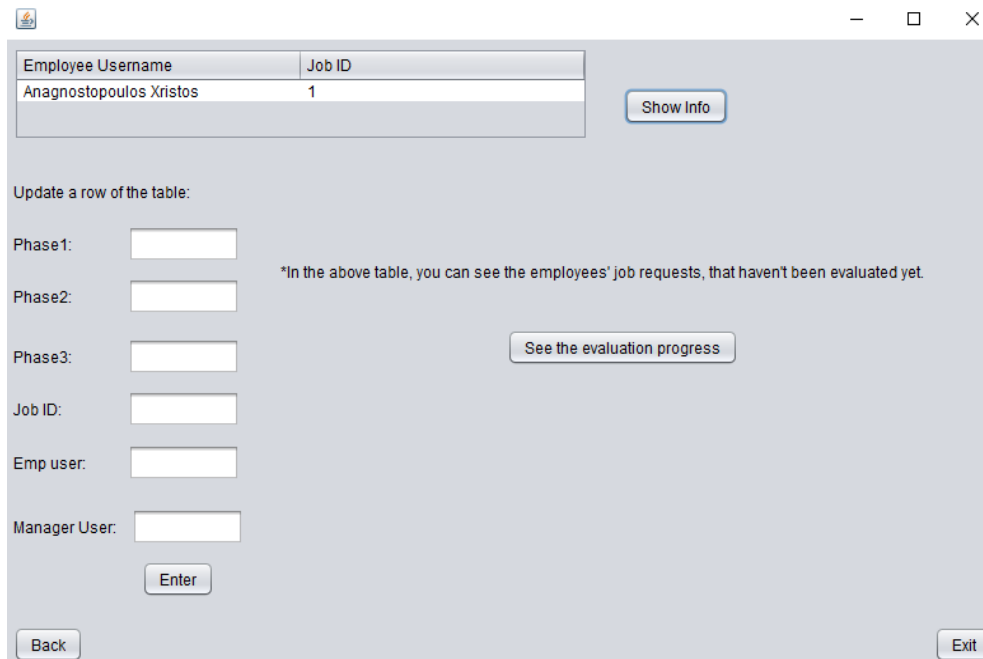
Enter

Back

Exit

Αντίστοιχα ό,τι συμβαίνει και με το παράθυρο του admin.

5. Evaluation Progress



Employee Username	Job ID
Anagnostopoulos Xristos	1

Show Info

Update a row of the table:

Phase1:

Phase2:

Phase3:

Job ID:

Emp user:

Manager User:

Enter

*In the above table, you can see the employees' job requests, that haven't been evaluated yet.

See the evaluation progress

Back

Exit

Βλέπει τις υποβολές (από τους υπαλλήλους της εταιρείας που εργάζεται) των θέσεων εργασίας που έχει αναλάβει μόνο. Αν κάποια υποβολή έχει περάσει στο στάδιο της αξιολόγησης, συμπληρώνει τα πεδία που φαίνονται παραπάνω (αν δεν έχει μπει σε κάποια από τις 3 φάσεις, μπορεί να μην την συμπληρώσει την συγκεκριμένη) και τότε αυτή μετακινείται από τον πίνακα request_evaluation, στον πίνακα evaluation_grades.

Πατώντας το κουμπί See the evaluation progress:

The screenshot shows a web application window with a title bar containing a small icon and standard window controls (minimize, maximize, close). The main content area features a table with the following headers: "Manager Us...", "Job ID", "Employee U...", "Phase1", "Phase2", "Phase3", and "Report Man...". Below the table is a "Show Info" button. Underneath this button, the text "Update a row of the above table:" is displayed. This is followed by a form with three rows, each corresponding to a phase. Each row contains three input fields labeled "Employee user:", "Job ID:", and an "Enter" button. The rows are labeled "Phase1:", "Phase2:", and "Phase3:". At the bottom left of the form is a "Back" button, and at the bottom right is an "Exit" button.

Βλέπει μόνο τις υποβολές (των υπαλλήλων της εταιρείας που εργάζεται) που έχει αναλάβει και έχουν ολοκληρώσει μία ή περισσότερες φάσεις. Όπως φαίνεται, μπορεί να συμπληρώσει το βαθμό σε κάποια φάση που δεν έχει ή να τον αλλάξει. Ο πίνακας ανανεώνεται αυτόματα και εμφανίζει τα καινούργια αποτελέσματα. Σε κάθε εισαγωγή καλείται η stored 3.2, ώστε σε περίπτωση που αυτή η εισαγωγή ολοκληρώσει την αξιολόγηση του υπαλλήλου να μεταφερθεί στον πίνακα evaluation_result.

6. Progress of a job ID

Enter the ID of a job, you've posted:

Enter

Employee U...	Job ID	Grade	Comments

Back Exit

Εισάγει το ID μιας θέσης εργασίας και στον πίνακα εμφανίζονται όλες οι υποβολές τον υπαλλήλων που έχουν ολοκληρώσει όλα τα στάδια της αξιολόγησης για αυτή τη θέση (άρα βρίσκονται στον πίνακα `evaluation_result`). Το παραπάνω συμβαίνει, καλώντας την `stored 3.3`. Την `out parameter result`, την χρησιμοποιώ, ώστε να εμφανίζω στον `evaluator` τον αριθμό υποβολών (για το ίδιο ID) που βρίσκονται σε αξιολόγηση αλλά δεν την έχουν ολοκληρώσει ακόμα.

*Το παρακάτω query: `String sql = "INSERT INTO tempuser(username) VALUES(?)";` που υπάρχει στο `gui`, γίνεται ώστε να αποθηκευτεί η εκάστοτε ενέργεια στον πίνακα `log_table`, όπως έχει εξηγηθεί και παραπάνω.