
TP BASES DE DONNEES

Rapport de projet final

Fait par :

Nom : Harzelli

Prénom : Yasser

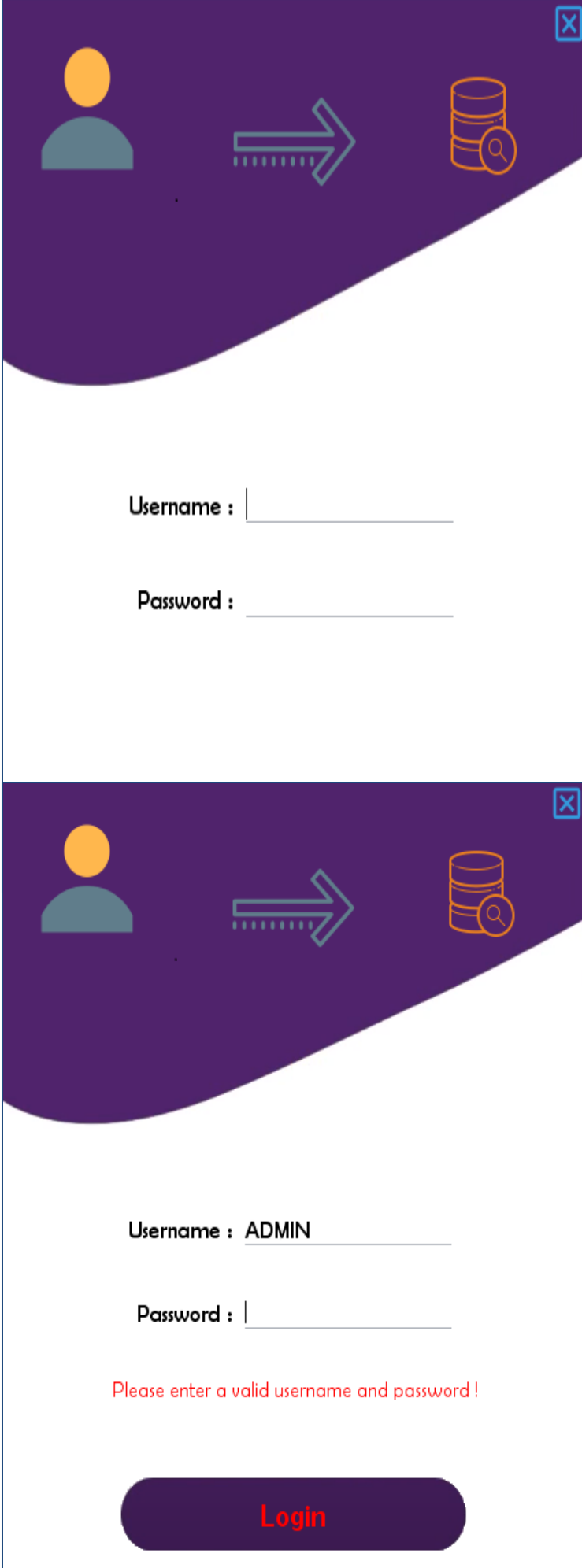
Matricule : 161732056786

Section : B

Groupe : 1

● Login Page :

- Si on insert user ou password qui n'existe pas, une erreur s'affiche en rouge
- Si user et password sont corrects le bouton « Login » devient vert c à d on peut accéder la frame suivante
- [Inspecter le code : cliquez ici](#)



The image shows a login page with a purple header containing icons for a user, an arrow, and a database. Below the header are two input fields: 'Username : ' and 'Password : '. The 'Username' field contains the text 'ADMIN'. Below the input fields is a red error message: 'Please enter a valid username and password !'. At the bottom is a dark purple rounded button with the text 'Login' in red.

Username :

Password :

Please enter a valid username and password !

Login



The image shows a login page with a purple header containing icons for a user, an arrow, and a database. Below the header are two input fields: 'Username : ' and 'Password : '. The 'Username' field contains the text 'BDDAdmin' and the 'Password' field contains '*****'. Below the input fields is a green rounded button with the text 'Login' in green.

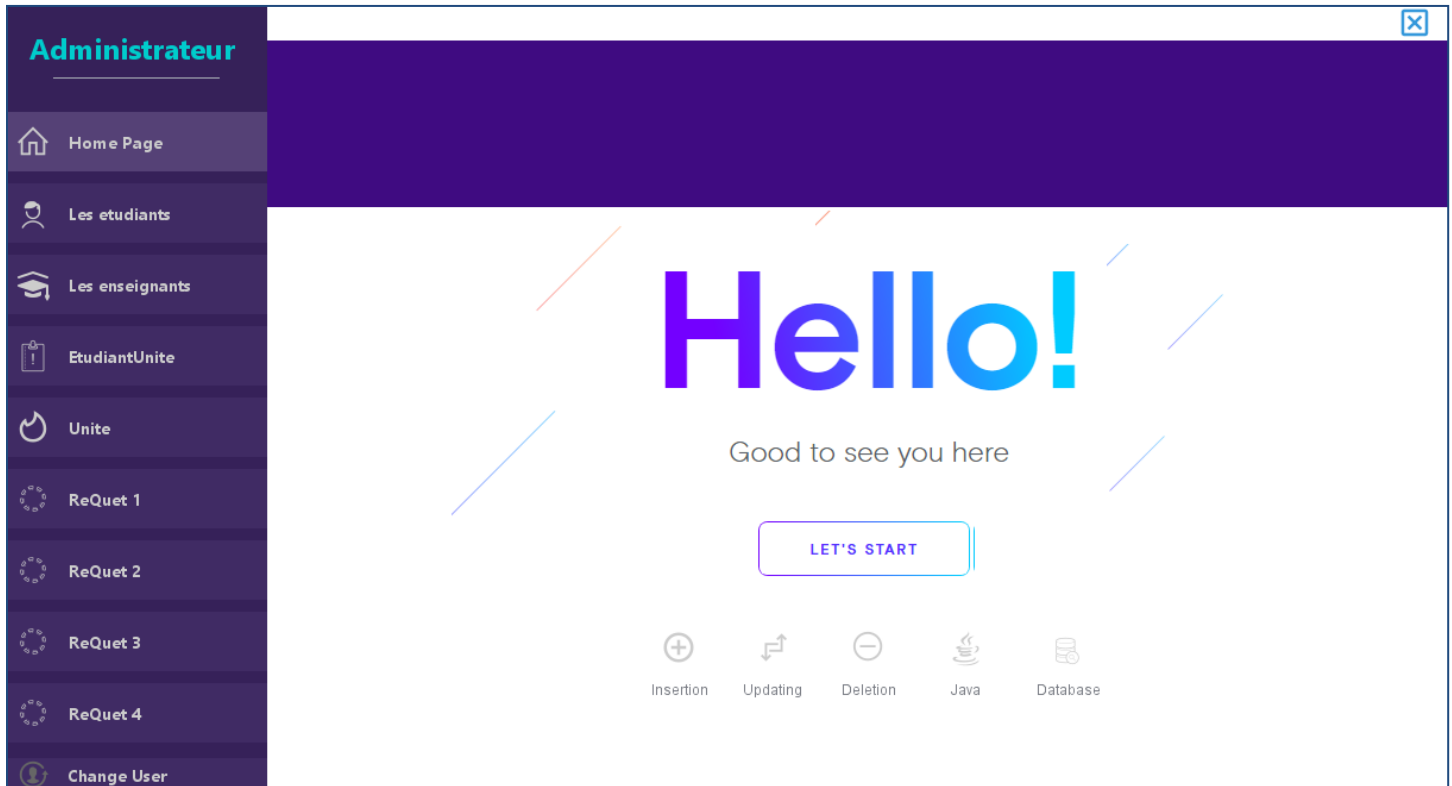
Username :

Password :

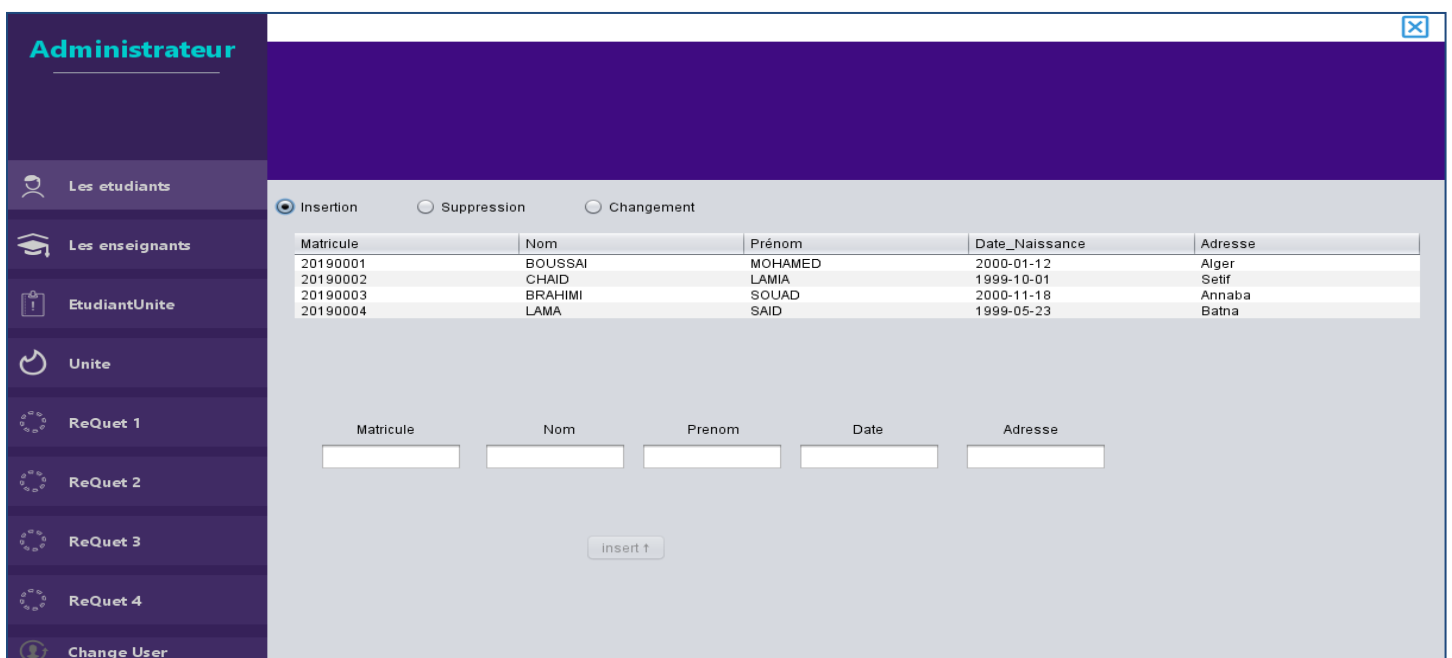
Login

● Panneau de user BDDAdmin :

Après en introduisant le user BDDAdmin et password TPAdmin, le panneau d'administrateur apparait avec la page d'accueil



Cliquons sur « LET'S START » pour que le **tableau des étudiants** apparaisse



-> Opération d'insertion :

après avoir sélectionné « Insertion », nous insérons les valeurs qui conviennent : matricule , nom , prénom , date et adresse

Matricule	Nom	Prénom	Date_Naissance	Adresse
20190001	BOUSSAI	MOHAMED	2000-01-12	Alger
20190002	CHAID	LAMIA	1999-10-01	Setif
20190003	BRAHIMI	SOUAD	2000-11-18	Annaba
20190004	LAMA	SAID	1999-05-23	Batna

Matricule	Nom	Prénom	Date	Adresse
<input type="text" value="20190005"/>	<input type="text" value="Harzelli"/>	<input type="text" value="Yasser"/>	<input type="text" value="1999-02-14"/>	<input type="text" value="Medea"/>

Puis on clique sur « Insert »

Matricule	Nom	Prénom	Date_Naissance	Adresse
20190005	Harzelli	Yasser	1999-02-14	Medea
20190001	BOUSSAI	MOHAMED	2000-01-12	Alger
20190002	CHAID	LAMIA	1999-10-01	Setif
20190003	BRAHIMI	SOUAD	2000-11-18	Annaba
20190004	LAMA	SAID	1999-05-23	Batna

Message

saved successfully

Matricule	Nom	Prénom	Date	Adresse
<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>	<input type="text"/>

- Inspecter le code : [cliquez ici](#)

Propriétés :

- le bouton insert s'active lorsqu' on introduit les informations dans tous les champs
- si on introduit un matricule qui existe déjà, le champ du nom se remplit automatiquement avec le nom qui correspond au matricule
- on doit introduire une date en forme « AAAA-MM-JJ c'est une contrainte dans la BDD

-> Opération de suppression :

Après avoir sélectionné « Suppression », nous insérons le matricule de l'étudiant qu'on veut supprimer

The screenshot shows the 'Administrateur' application interface. On the left is a sidebar with navigation links: 'Les étudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', 'ReQuet 1', 'ReQuet 2', and 'ReQuet 3'. The main area has three radio buttons: 'Insertion', 'Suppression' (selected), and 'Changement'. Below these is a table with columns: 'Matricule', 'Nom', 'Prénom', 'Date_Naissance', and 'Adresse'. The table contains four rows of student data. Below the table are input fields for 'Matricule' (containing '20190005') and 'Nom' (containing 'Harzelli'). At the bottom is a 'Supprimer X' button. A 'Message' dialog box is open in the center, displaying an information icon and the text 'deleted successfully', with an 'OK' button.

Matricule	Nom	Prénom	Date_Naissance	Adresse
20190001	BOUSSAI	MOHAMED	2000-01-12	Alger
20190002	CHAI	LAMIA	1999-10-01	Setif
20190003	BRAHIMI	SOUAD	2000-11-18	Annaba
20190004	LAMA	SAID	1999-05-23	Batna

Propriétés :

- le bouton « supprimer » s'active lorsque les champs matricule et nom sont remplis.
- après en introduisant le matricule, le champ du nom se remplit automatiquement avec le nom qui correspond au matricule donc le bouton « supprimer » s'active

-> Opération de changement :

Après avoir sélectionné « Changement », nous insérons le matricule de l'étudiant qu'on veut changer ses informations

The screenshot shows the 'Administrateur' interface with a sidebar on the left containing menu items: 'Les étudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', 'ReQuet 1', 'ReQuet 2', and 'ReQuet 3'. The main area has three radio buttons: 'Insertion', 'Suppression', and 'Changement' (which is selected). Below these is a table with student data:

Matricule	Nom	Prénom	Date_Naissance	Adresse
20190001	BOUSSAI	MOHAMED	2000-01-12	Alger
20190002	CHAID	LAMIA	1999-10-01	Setif
20190003	BRAHIMI	SOUAD	2000-11-18	Annaba
20190004	LAMA	SAID	1999-05-23	Batna

Below the table, there are input fields for 'Matricule', 'Nom', 'Prénom', 'Date', and 'Adresse'. The 'Matricule' field contains '20190003'. The 'Nom' field contains 'BRAHIMI', 'Prénom' contains 'SOUAD', 'Date' contains '2000-11-18', and 'Adresse' contains 'Alger'. A 'Changer' button is at the bottom left of the form area.

Après introduire les changements on clique sur le bouton « Changer »

This screenshot shows the same interface as the previous one, but with a red box highlighting the row for Matricule 20190003 in the table. A modal dialog box titled 'Message' is displayed in the center, containing an information icon and the text 'changé avec succès'. An 'OK' button is at the bottom right of the dialog. The input fields and the 'Changer' button are still visible in the background.

Propriétés :

- Le bouton « changer » s'active lorsque tous les champs sont remplis.
- Tous les champs sauf le matricule deviennent visibles lorsqu'on introduit un matricule qui existe, alors tous les champs sont remplis automatiquement avec le tuple qui correspond au la matricule donc, le bouton « Changer » s'active

NB : toutes les opérations fonctionnent de la même façon que celles de la table d'étudiant, alors pour les prochaines tables (enseignant, etudiantunite, unite)

on met seulement le code de l'opération d'insertion et ces photos.

Tableau des enseignants :

Administrateur

Les étudiants

Les enseignants

EtudiantUnite

Unite

ReQuet 1

ReQuet 2

Insertion ☒ Suppression ☐ Changement ☐

Matricule	Nom	Prénom
20190001	Harouni	Amine
19990011	FATHI	OMAR
19980078	BOUZIDANE	FARAH
20170015	ARABI	ZOUBIDA
20190002	Harzelli	Yasser

Message

saved successfully

OK

Nom: Harzelli

Prénom: Yasser

insert ↑

- [Inspecter le code : cliquez ici](#)

Tableau EtudiantUnite :

Administrateur

Les étudiants
Les enseignants
EtudiantUnite
Unite
ReQuet 1
ReQuet 2

☒ Insertion ☐ Suppression ☐ Changement

Matricule	Code Unite	Note CC	Note TP	Note Examen
20190001	FEI0002	10	15	9
20190001	FEI0004	15	15	14
20190001	FEI0001	13	13	10
20190004	FEI0001	17	17	16
20190003	FEI0001	8	8	15
20190004	FEI0001	9	9	20
20190003	FEI0001	12	12	15
20190004	FEI0001	13	13	20

Message
i saved successfully
OK

Matricule Code TP Note Examen
20190001 FEI0004 15 15 14

insert t

- Inspecter le code : [cliquez ici](#)

Tableau Unite :

Administrateur

Les étudiants
Les enseignants
EtudiantUnite
Unite
ReQuet 1
ReQuet 2
ReQuet 3
ReQuet 4
Change User

☒ Insertion ☐ Suppression ☐ Changement

Code Unite	Libelle	NB Heures	Matricule Ens
FEI0001	POO	6	20190001
FEI0002	BDD	6	19990011
FEI0003	RESEAU	3	20170015
FEI0004	SYSTEME	6	19980078
FEI0005	Archi	6	20190001

Code Unite Libelle NB Heures Matricule
FEI0005 Archi 6 20190001

insert t

Message
i saved successfully
OK

- Inspecter le code : [cliquez ici](#)

➤ 1^{ère} Requête:

1^{ère} question, Tache 5, TP 7 : affiche les noms et prénoms des étudiants ayant obtenus des notes d'examens égales à 20

J'ai amélioré la requête, maintenant, elle vous donne la chance d'introduire la note que vous voulez et en plus, elle affiche le module d'où il/elle a pris cette note

The screenshot shows the 'Administrateur' interface. On the left is a sidebar with navigation links: 'Les etudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', and 'ReQuet 1'. The main area has a title bar with a close button. Below the title bar, there is a text input field with the placeholder 'Inserer une valeur !'. The main content area displays a table with the following columns: 'Matricule', 'Nom', 'Prénom', and 'Module'. The table is currently empty.

On la teste en introduisant la note 20

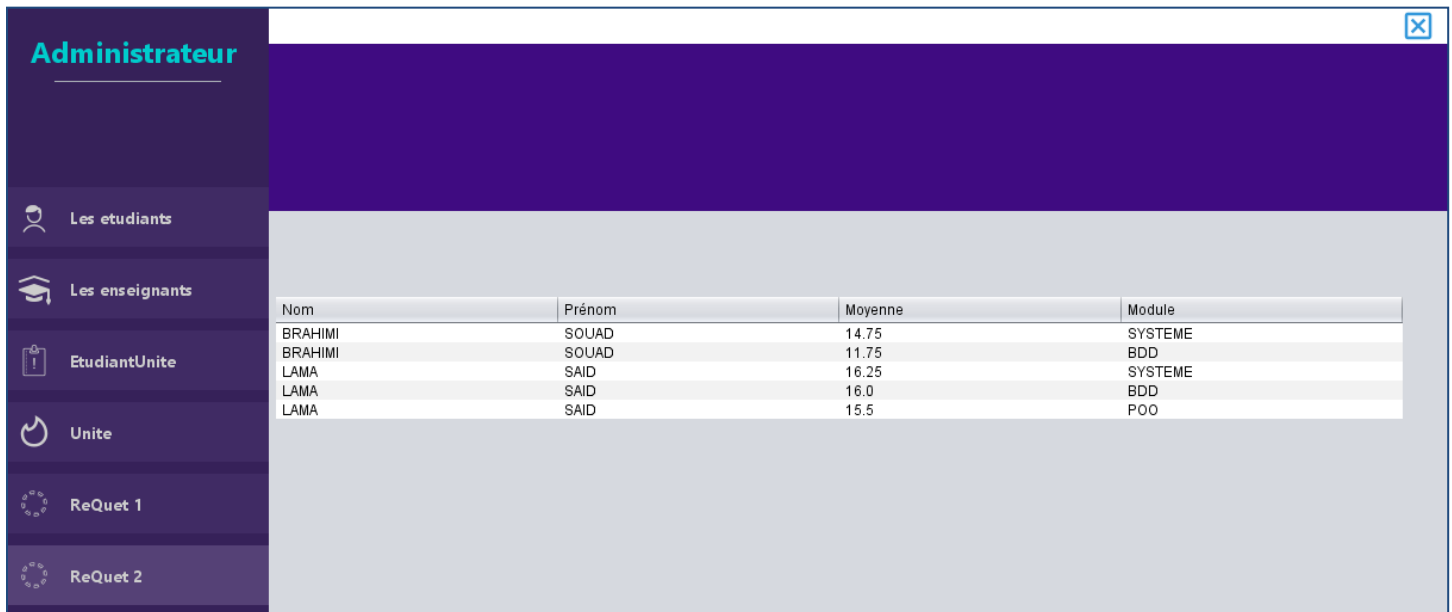
The screenshot shows the 'Administrateur' interface with the search criteria set to 20. The main content area displays a table with the following columns: 'Matricule', 'Nom', 'Prénom', and 'Module'. The table contains two rows of data:

Matricule	Nom	Prénom	Module
20190004	LAMA	SAID	BDD
20190004	LAMA	SAID	SYSTEME

[- Inspecter le code : cliquez ici](#)

➤ 2^{ème} Requête:

4^{ème} question, Tache 5, TP 7 : Affiche pour chaque étudiant, son nom, son prénom sa moyenne par unité d'enseignement ainsi que le libellé de l'unité.



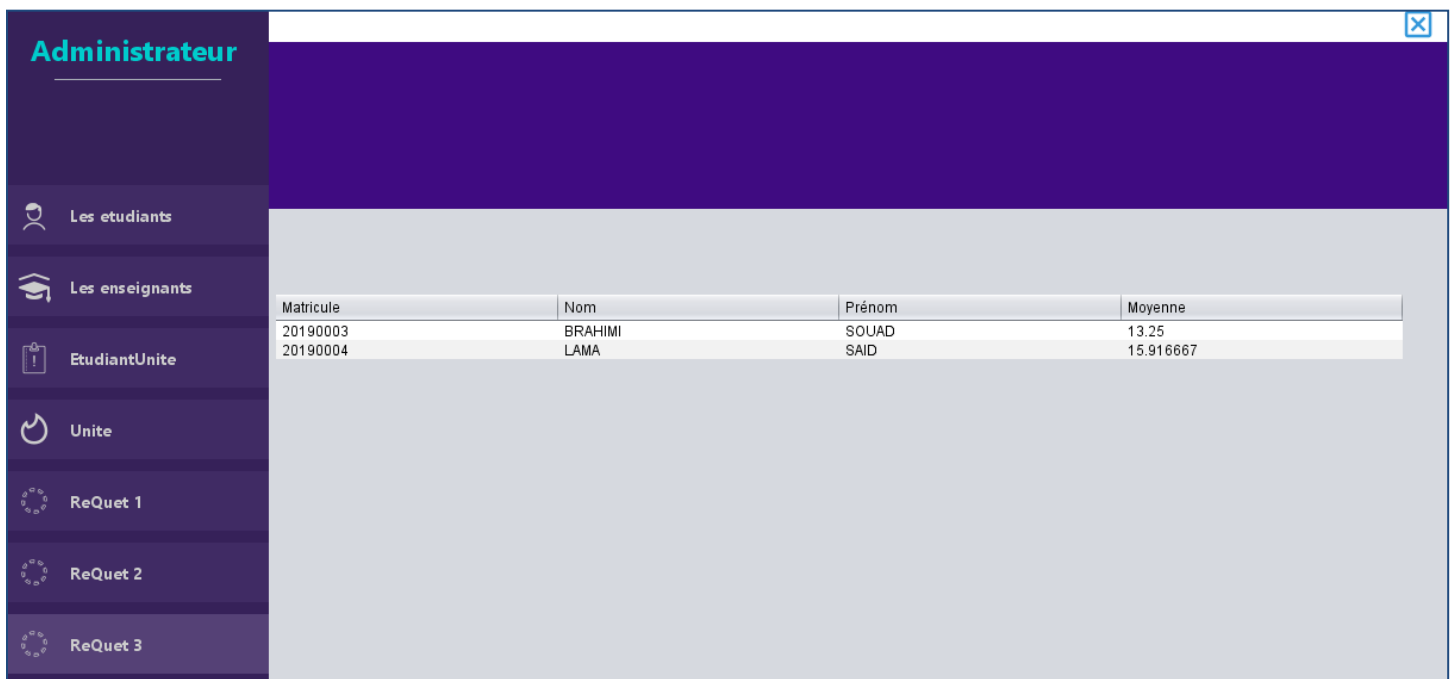
The screenshot shows the 'Administrateur' application interface. On the left is a dark purple sidebar with a menu containing: 'Les etudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', 'ReQuet 1', and 'ReQuet 2'. The main content area has a dark purple header bar and a light gray table below it. The table displays student information with the following columns: Nom, Prénom, Moyenne, and Module.

Nom	Prénom	Moyenne	Module
BRAHIMI	SOUAD	14.75	SYSTEME
BRAHIMI	SOUAD	11.75	BDD
LAMA	SAID	16.25	SYSTEME
LAMA	SAID	16.0	BDD
LAMA	SAID	15.5	POO

- Inspecter le code : [cliquez ici](#)

➤ 3^{ème} Requête:

2^{ème} question, tache 2, TP 8 : Affiche le contenu de la vue V_MOYGENERALE



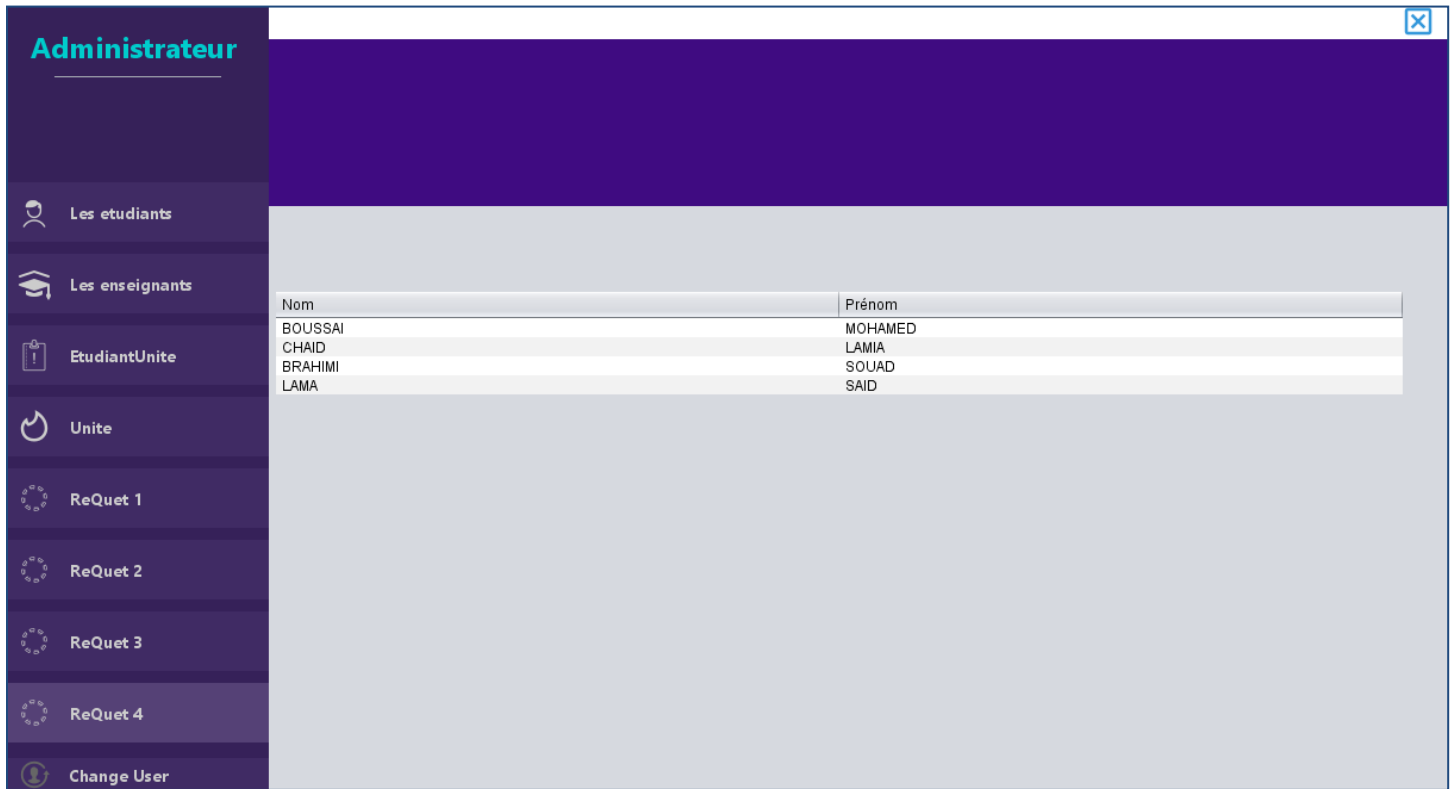
The screenshot shows the 'Administrateur' application interface. On the left is a dark purple sidebar with a menu containing: 'Les etudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', 'ReQuet 1', 'ReQuet 2', and 'ReQuet 3'. The main content area has a dark purple header bar and a light gray table below it. The table displays student information with the following columns: Matricule, Nom, Prénom, and Moyenne.

Matricule	Nom	Prénom	Moyenne
20190003	BRAHIMI	SOUAD	13.25
20190004	LAMA	SAID	15.916667

- Inspecter le code : [cliquez ici](#)

➤ 4^{ème} Requête:

1^{er} question, tache 2, TP 8 : Affiche le contenu de la vue V_ETUDIANT_LISTE



The screenshot shows the BDDAdmin web application interface. On the left is a dark purple sidebar with the title 'Administrateur' and several menu items: 'Les etudiants', 'Les enseignants', 'EtudiantUnite', 'Unite', 'ReQuet 1', 'ReQuet 2', 'ReQuet 3', 'ReQuet 4', and 'Change User'. The main content area has a dark purple header bar. Below it, a table displays data from the 'V_ETUDIANT_LISTE' view. The table has two columns: 'Nom' and 'Prénom'. It contains four rows of student data.

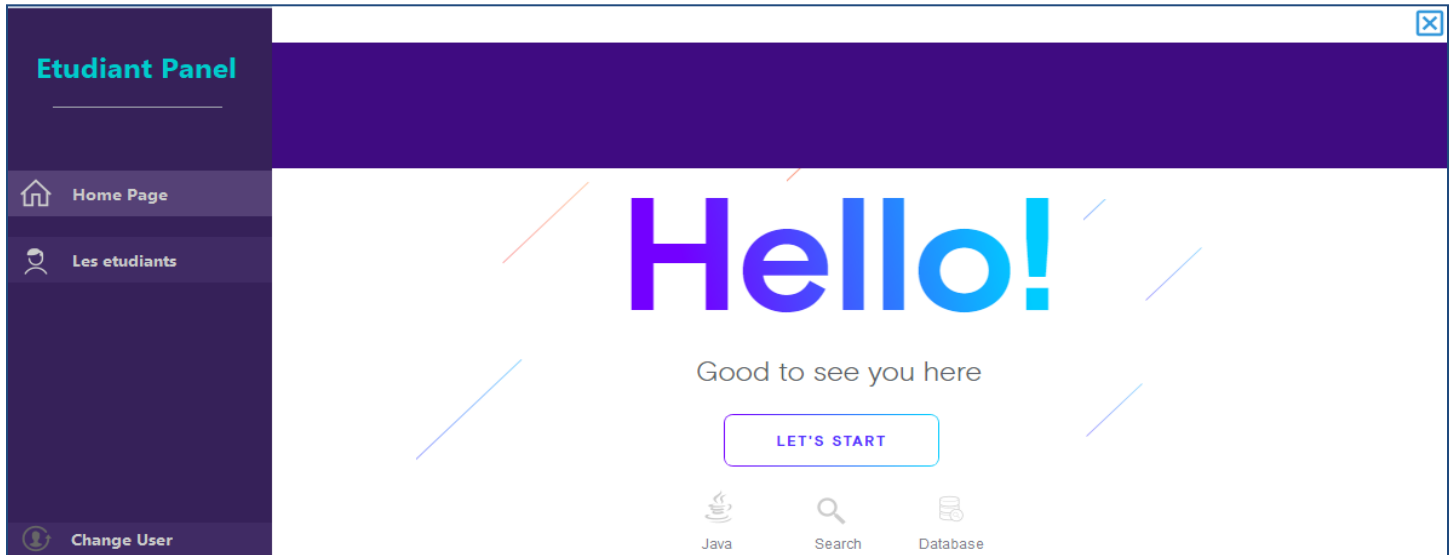
Nom	Prénom
BOUSSAI	MOHAMED
CHAD	LAMIA
BRAHIMI	SOUAD
LAMA	SAID

[- Inspecter le code : cliquez ici](#)

Après avoir terminé de consulter le panneau de BDDAdmin, on peut basculer entre les utilisateurs on cliquant « Change User »

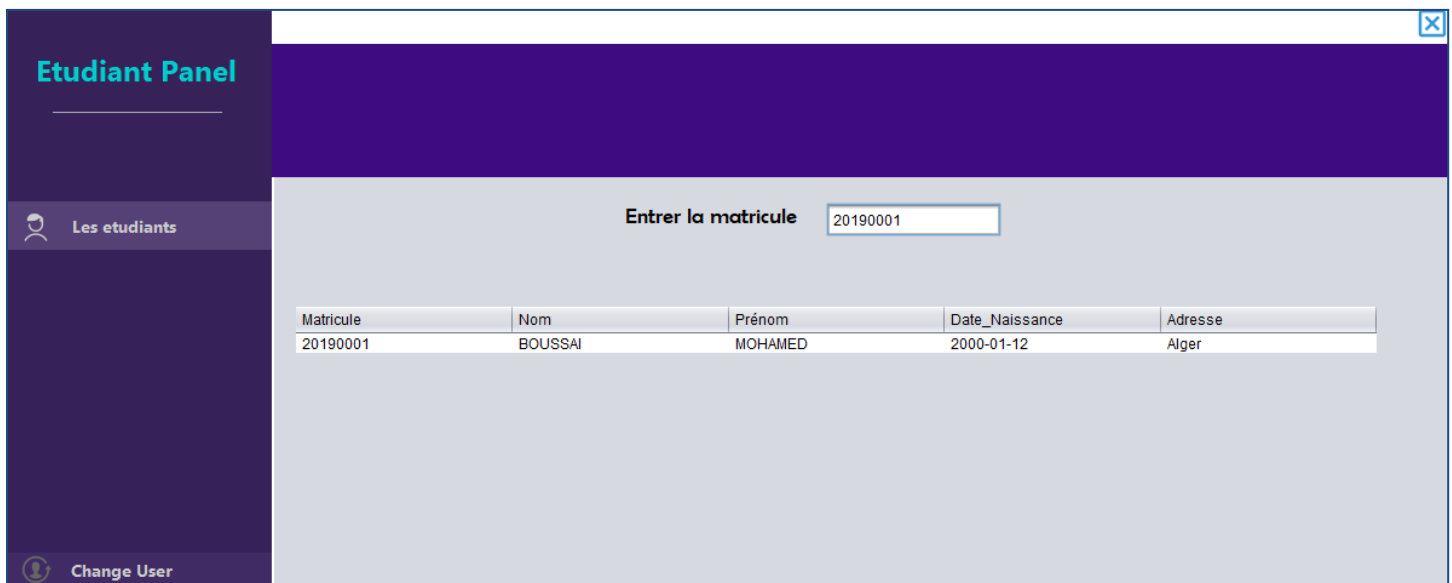
● Panneau de user Etudiant :

Après introduire le user Etudiant et password TPEtudiant , le panneau d'étudiant montre avec la page d'accueil



Cliquons sur « LET'S START » le tableau des étudiants apparaisse

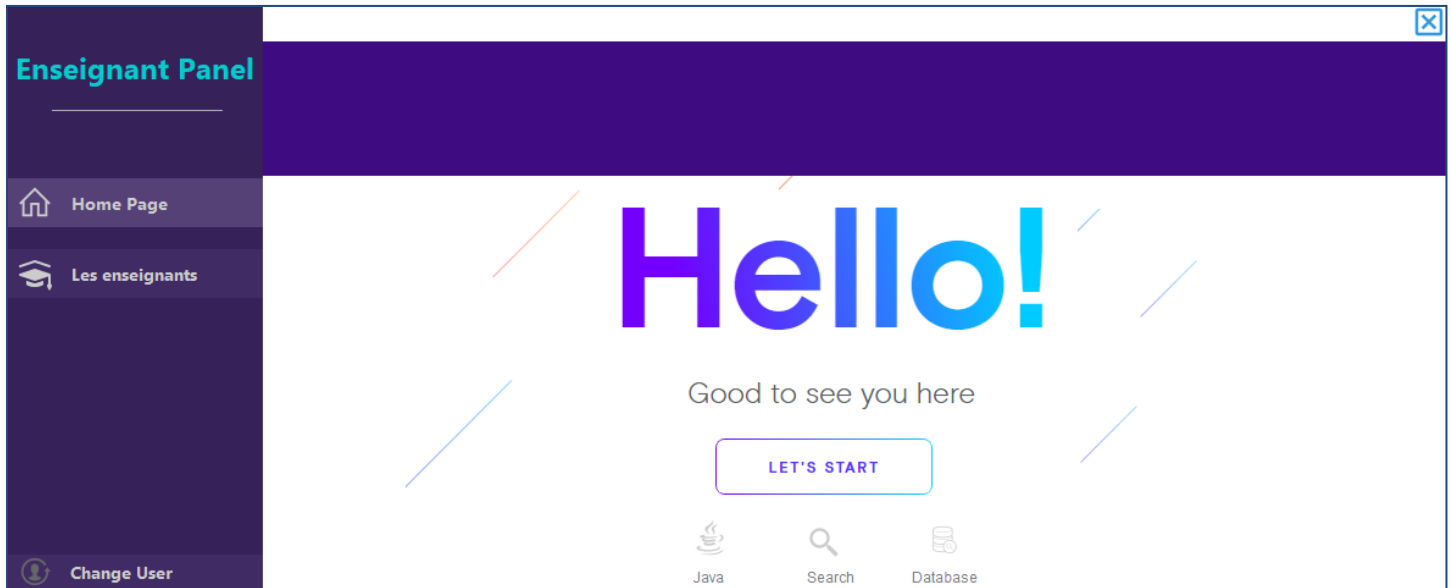
et après introduire l'un des matricules à rechercher, on obtient le résultat



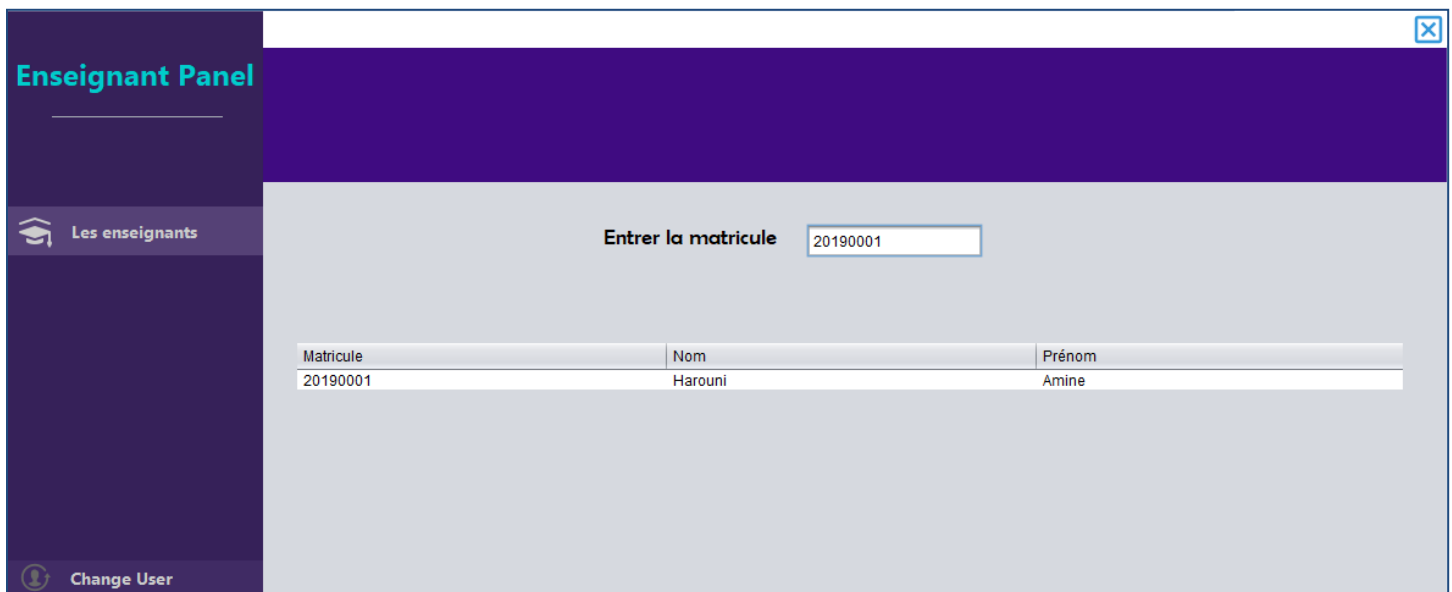
- [Inspecter le code : cliquez ici](#)

● Panneau de user Enseignant :

Après introduire le user Enseignant et password TPEnseignant , le panneau d'enseignant montre avec la page d'accueil



Cliquons sur « LET'S START » le tableau des enseignants apparaisse et après introduire l'un des matricules à rechercher, on obtient le résultat



- [Inspecter le code : cliquez ici](#)

PARTIE CODE:

Code 1 : le login

```
try{
    if(user_id.getText().equals("BDDAdmin") && password.getText().equals("TPAdmin")) {
        Class.forName("oracle.jdbc.OracleDriver");
        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl",user_id.getText(),password.getText());
        st = conn.createStatement();
        new HomeAdmin().setVisible(true);
        this.hide();
    }

    if(user_id.getText().equals("Etudiant") && password.getText().equals("TPEtudiant")) {
        Class.forName("oracle.jdbc.OracleDriver");
        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl",user_id.getText(),password.getText());
        st = conn.createStatement();
        new LoginEtudiant().setVisible(true);
        this.hide();
    }

    if(user_id.getText().equals("Enseignant") && password.getText().equals("TPEnseignant")) {
        Class.forName("oracle.jdbc.OracleDriver");
        conn = DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:orcl",user_id.getText(),password.getText());
        st = conn.createStatement();
        new LoginEnseignant().setVisible(true);
        this.hide();
    }
} catch (Exception e) {
    JOptionPane.showMessageDialog(null,e);
}
```

[Retourner](#)

Code 2 : Insertion dans la table Etudiant

```
private void inser_buttonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {

        String sql = "insert into etudiant(matricule_etu,nom_etu,prenom_etu,date_naissance,adresse) values (?, ?, ?, ?, ?)";
        PreparedStatement ps=cn.prepareStatement(sql);
        ps.setInt(1,Integer.parseInt(matricule.getText()));
        ps.setString(2,nom.getText());
        ps.setString(3,prenom.getText());
        ps.setDate(4,(java.sql.Date.valueOf(date.getText())));
        ps.setString(5,adress.getText());

        ps.executeUpdate();
        refresh();
        nom.setText("");
        prenom.setText("");
        date.setText("");
        matricule.setText("");
        adress.setText("");

        JOptionPane.showMessageDialog(this,"saved successfully");
    }
    catch (Exception e)
    {
        JOptionPane.showMessageDialog(this,e);
    }
}
```

[Retourner](#)

Code 3 : Insertion dans la table Enseignant

```
private void inser_buttonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try {  
        String sql ="insert into enseignant(matricule_ens,nom_ens,prenom_ens) values (?, ?, ?)";  
        PreparedStatement ps=cn.prepareStatement(sql);  
        ps.setInt(1,Integer.parseInt(matricule.getText()));  
        ps.setString(2,nom.getText());  
        ps.setString(3,prenom.getText());  
  
        ps.executeUpdate();  
        refresh();  
        JOptionPane.showMessageDialog(this,"saved successfully");  
        nom.setText("");  
        prenom.setText("");  
        matricule.setText("");  
    }  
    catch(Exception e)  
    {  
        JOptionPane.showMessageDialog(this,e);  
    }  
}
```

[Retourner](#)

CODE 4 : Insertion dans la table EtudiantUnite

```
private void inser_buttonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try {  
        String sql ="insert into etudiantunite(matricule_etu,code_unite,note_cc,note_tp,note_examen) values (?, ?, ?, ?, ?)";  
        PreparedStatement ps=cn.prepareStatement(sql);  
        ps.setInt(1,Integer.parseInt(matricule.getText()));  
        ps.setString(2,codeUNITE.getText());  
        ps.setString(3,noteCC.getText());  
        ps.setString(4,noteTP.getText());  
        ps.setString(5,noteEX.getText());  
  
        ps.executeUpdate();  
        refresh();  
        JOptionPane.showMessageDialog(this,"saved successfully");  
        codeUNITE.setText("");  
        noteCC.setText("");  
        noteTP.setText("");  
        matricule.setText("");  
        noteEX.setText("");  
    }  
    catch(Exception e)  
    {  
        JOptionPane.showMessageDialog(this,e);  
    }  
}
```

[Retourner](#)

CODE 5 : Insertion dans la table Unite

```
private void inser_buttonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    try {  
  
        String sql ="insert into unite(code_unite,libelle,nbr_heures,matricule_ens) values (?, ?, ?, ?)";  
        PreparedStatement ps=cn.prepareStatement(sql);  
        ps.setInt(4,Integer.parseInt(matricule.getText()));  
        ps.setString(1,codeUNITE.getText());  
        ps.setString(2,lib.getText());  
        ps.setString(3,nbh.getText());  
        ps.executeUpdate();  
        refresh();  
        JOptionPane.showMessageDialog(this,"saved sucessfully");  
        codeUNITE.setText("");  
        nbh.setText("");  
        lib.setText("");  
        matricule.setText("");  
  
    }  
    catch(Exception e)  
    {  
        JOptionPane.showMessageDialog(this,e);  
    }  
}
```

[Retourner](#)

CODE 6 : requête 01

```
public void refresh(){  
    try{  
        PreparedStatement ps2=cn.prepareStatement("select * from etudiantunite where note_examen = ?");  
        ps2.setInt(1,Integer.parseInt(note.getText()));  
        ResultSet rs2 = ps2.executeQuery();  
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();  
        tm.setRowCount(0);  
        while(rs2.next()){  
            PreparedStatement ps=cn.prepareStatement("select * from etudiant");  
            ResultSet rs = ps.executeQuery();  
            while(rs.next()){  
                if (rs2.getInt("matricule_etu") == rs.getInt("matricule_etu")){  
                    PreparedStatement ps3=cn.prepareStatement("select * from unite where code_unite = ?");  
                    ps3.setString(1,rs2.getString("code_unite"));  
                    ResultSet rs3 = ps3.executeQuery();  
                    rs3.next();  
                    Object o[]={rs.getInt("matricule_etu"),rs.getString("nom_etu"),rs.getString("prenom_etu"),rs3.getString("libelle")};  
                    tm.addRow(o);  
                }  
            }  
        }  
    }catch(Exception e){  
        error_pane.setVisible(true);  
    }  
}
```

[Retourner](#)

Code 7 : requête 02


```

public void refresh() {
    try{

        PreparedStatement ps=cn.prepareStatement("select nom_etu,prenom_etu,libelle, ((note_cc+note_tp+(note_examen*2))/4) as moy"
            + " from etudiant,etudiantunite,unite where etudiant.matricule_etu = etudiantunite.matricule_etu and "
            + "etudiantunite.code_unite = unite.code_unite");
        ResultSet rs = ps.executeQuery();
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();
        tm.setRowCount(0);

        while(rs.next()){
            Object o[]={rs.getString("nom_etu"),rs.getString("prenom_etu"),rs.getFloat("moy"),rs.getString("libelle")};
            tm.addRow(o);
        }
    }catch(Exception e){
        JOptionPane.showMessageDialog(this,e);
    }
}

```

[Retourner](#)

CODE 8 : requête 03

```

public void refresh() {
    try{

        PreparedStatement ps=cn.prepareStatement("select * from v_moygenerale");
        ResultSet rs = ps.executeQuery();
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();
        tm.setRowCount(0);

        while(rs.next()){
            Object o[]={rs.getString("mat"),rs.getString("nom_etu"),rs.getString("prenom_etu"),rs.getFloat("moy_general")};
            tm.addRow(o);
        }
    }catch(Exception e){

    }
}

```

[Retourner](#)

CODE 9 : requête 04

```

public void refresh() {
    try{

        PreparedStatement ps=cn.prepareStatement("select * from v_etudiant_liste");
        ResultSet rs = ps.executeQuery();
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();
        tm.setRowCount(0);

        while(rs.next()){
            Object o[]={rs.getString("nom_etu"),rs.getString("prenom_etu")};
            tm.addRow(o);
        }
    }catch(Exception e){

    }
}

```

[Retourner](#)

Code 10 : utilisateur étudiant qui pourra consulter un tuple dans la table étudiant

```
public void refresh() {
    try{

        PreparedStatement ps=cn.prepareStatement("select * from etudiant where matricule_etu = "+matricule.getText());
        ResultSet rs = ps.executeQuery();
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();
        tm.setRowCount(0);
        while(rs.next()){
            Object o[]={rs.getInt("Matricule_etu"),rs.getString("nom_etu"),rs.getString("prenom_etu"),rs.getDate("date_naissance"),rs.getString("adresse")};
            tm.addRow(o);
        }
    }catch(Exception e){
        refresh();
    }
}
```

[Retourner](#)

Code 11 : utilisateur enseignant qui pourra consulter un tuple dans la table enseignant

```
public void refresh() {
    try{

        PreparedStatement ps=cn.prepareStatement("select * from enseignant where matricule_ens = "+matricule.getText());
        ResultSet rs = ps.executeQuery();
        DefaultTableModel tm=(DefaultTableModel)table1.getModel();
        tm.setRowCount(0);
        while(rs.next()){
            Object o[]={rs.getInt("Matricule_ens"),rs.getString("nom_ens"),rs.getString("prenom_ens")};
            tm.addRow(o);
        }
    }catch(Exception e){
    }
}
```

[Retourner](#)

