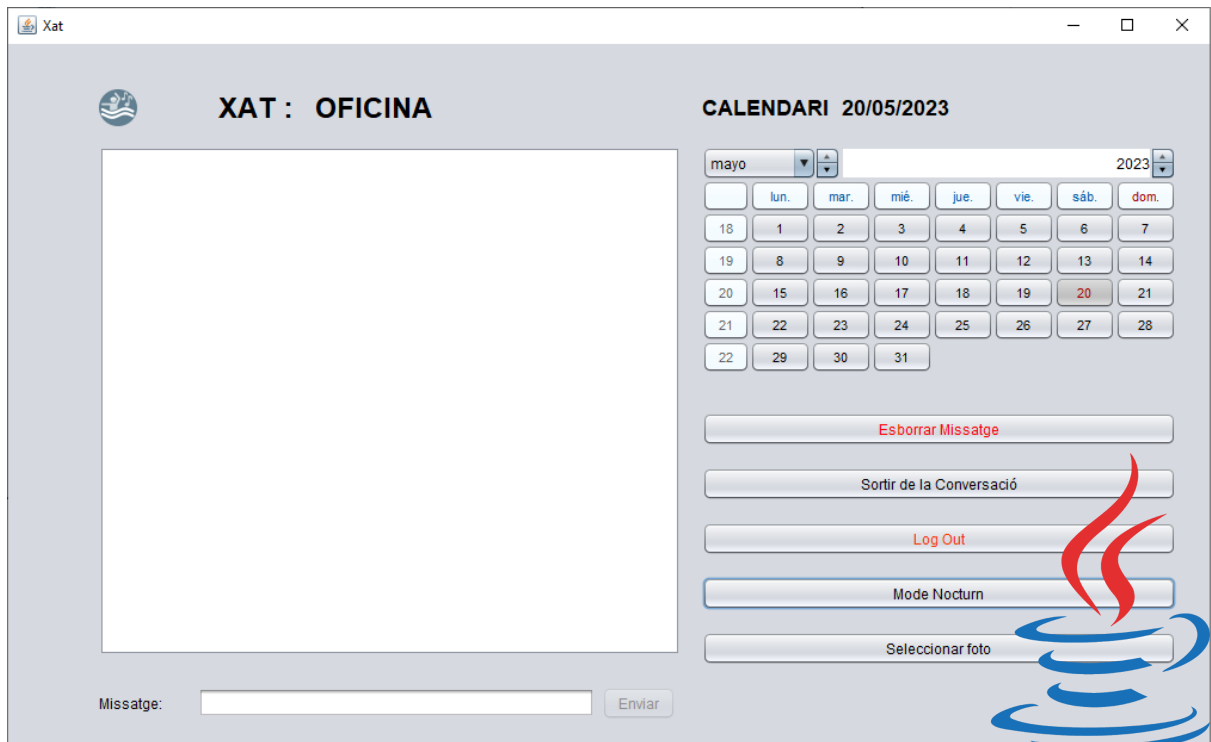


# Projecte M6 UF4

## Components d'accés a dades

### Java Beans



Nom: Èric Estivill i Oleg Blyznyuk  
Curs: 2n DAM  
Data: 21/05/2023

# ÍNDEX

<b>INTRODUCCIÓ</b>	<b>4</b>
<b>1. BASE DE DADES</b>	<b>5</b>
<b>2. LOGIN</b>	<b>6</b>
2.1 CODI	6
2.2 INTERFÍCIE	6
<b>3. CONVERSACIONS</b>	<b>7</b>
3.1 CODI	7
3.1.1 Carregar Conversacions	7
3.1.2 Seleccionar Conversació	8
3.1.3 Afegir Conversació	9
3.1.4 Unir-se a Conversació	9
3.1.5 Esborrar/Entrar Conversa	10
3.1.6 Afegir Conversa	11
3.2 INTERFÍCIE	11
3.2.1 Pantalla Inicial	11
3.2.2 Crear Conversa	12
<b>4. XAT</b>	<b>13</b>
4.1 CODI	13
4.1.1 Carregar Missatges d'Avui	13
4.1.2 Timer per Actualitzar Missatges	14
4.1.3 Obtenir Dades Missatge	15
4.1.4 Enviar Missatge	16
4.1.5 Calendari	17
4.1.6 Eliminar Missatge	18
4.1.7 Mode Diürn/Nocturn(Bean Propi)	18
4.1.8 Foto Perfil (Bean Propi)	19
4.2 INTERFÍCIE	20
4.2.1 Pantalla Inicial	20
4.2.2 Esborrar Missatge	20
4.2.3 Missatge	21
4.2.4 Mode Diürn/Nocturn	21
4.2.5 Canviar Foto Perfil	22
<b>5. BEAN</b>	<b>23</b>
5.1 CODI	23
5.1.1 Bean Mode Diürn/Nocturn	23
5.1.2 Bean Foto Perfil	24
5.2 INTERFÍCIE	25
5.2.1 Bean Mode Diürn/Nocturn	25
5.2.2 Bean Foto Perfil	25
5.3 CREAR BEAN	26
5.3.1 Primer Pas	26

5.3.2 Segon Pas	26
5.3.3 Tercer Pas	27
5.3.4 Quart Pas	27
5.3.5 Cinquè Pas	28
5.3.5 Últim Pas	28

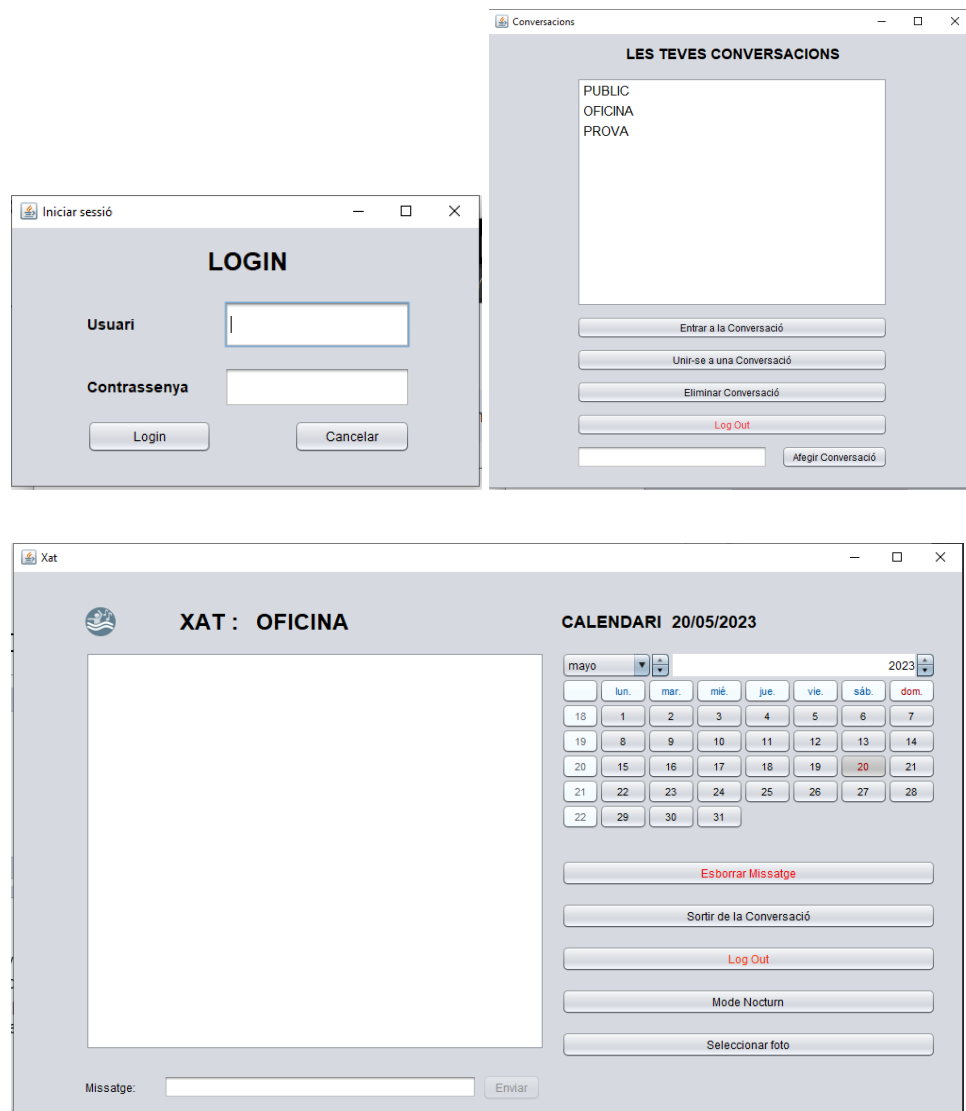
# INTRODUCCIÓ

En aquest projecte hem hagut de crear un 'whatsapp' en el qual havíem d'interaccionar amb la base de dades de mongodb on havíem de poder seleccionar converses, enviar missatges, iniciar sessió amb un usuari i contrassenya i més coses, poder seleccionar un dia en concret i que se't mostressin els missatges d'aquell dia, esborrar missatges de un dia seleccionat...

I, a més a més, hem hagut de crear un bean a elecció nostra.

En el nostre cas n'hem creat dos, bastant bàsics els quals fan funcions totalment diferents. Un d'ells ens serveix per canviar el fons del disseny a fosc o clar i viceversa (Diürn/Nocturn), i l'altre ens serveix per pujar una foto al perfil de l'usuari.

En el nostre projecte també hem afegit l'opció de poder crear converses en cas de ser Administrador.



# 1.BASE DE DADES

```

print("INICIANT SCRIPT");

conn = new Mongo("localhost");
db = conn.getDB("whatsapp");

db.dropDatabase();

db.createCollection("usuaris");
db.createCollection("missatges");
db.createCollection("grups");

print("*****crean llibres*****");

usuari1 = {
  "_id" : "0",
  "usuari" : "eric",
  "contrassenya" : "123",
  "admin" : true,
  "xats" : ['0'],
  "img" : null
}

usuari2 = {
  "_id" : "1",
  "usuari" : "oleg",
  "contrassenya" : "123",
  "admin" : false,
  "xats" : ['1'],
  "img" : null
}

xat1 = {
  "_id" : "0",
  "nom" : "PUBLIC"
}

xat2 = {
  "_id" : "1",
  "nom" : "OFICINA"
}

print("*****guardant llibres*****");
db.usuaris.insertOne(usuari1);
db.usuaris.insertOne(usuari2);
db.grups.insertOne(xat1);
db.grups.insertOne(xat2);
print("SCRIPT FINALITZAT");

```

A la nostra base de dades hem creat tres col·leccions, usuaris, xats, i missatges.

Als usuaris els hi hem assignat un xat per default, i els hi posem imatge 'null' per default també.

Per crear la base de dades hem utilitzat un script creat per nosaltres per crear la base.

Ja que no hem insertat res a missatges per default, els missatges els guardem de la següent manera:

```

{
  "_id": ObjectId("646925ab1fa39a3581fac5e9"),
  "user": 'eric',
  "missatge": 'com esteu?',
  "hora": ISODate("2023-05-20T19:55:23.618Z"),
  "xat": 'OFICINA'
}

```

L'usuari que l'ha enviat, el missatge, la hora i el xat en el que s'ha enviat

## 2.LOGIN

### 2.1 CODI

```
private MongoClient mongoClient;
private MongoDB database;
public MongoClient<Document> collection2;
private boolean isAdmin;
private Document filtro;
public login() {
    initComponents();
    setTitle("Iniciar sessió");
    mongoClient = MongoClient.create("mongodb://127.0.0.1:27017");
    database = mongoClient.getDatabase("whatsapp");
    collection2 = database.getCollection("usuarios");

    passwordField.addKeyListener(new KeyAdapter() {
        public void keyPressed(KeyEvent e) {
            if (e.getKeyCode() == KeyEvent.VK_ENTER)
            {
                IniciarSessio();
            }
        }
    });
}
```

El que fem aquí és connectar-nos al mongodb, posant la ip, definim la base de dades d'on agafar les col·leccions que necessitem.

El 'passwordField.addKeyListener' es per saber si enlloc de prémer el botó de login premem el enter faci la mateixa acció.

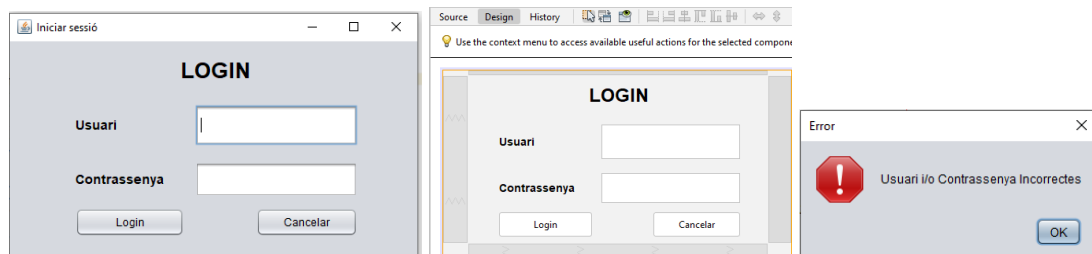
```
private void loginButtonActionPerformed(java.awt.event.ActionEvent evt) {
    IniciarSessio();
}

public void IniciarSessio() {
    String user = usernameField.getText();
    char[] password = passwordField.getPassword();
    filtro = new Document("usuari", user).append("contrassenya", new String(password));
    Document userDoc = collection2.find(filtro).first();

    if (userDoc != null) {
        // Si el usuario y la contraseña son correctos, abrir el chat
        dispose();
        isAdmin = userDoc.getBoolean("admin", false);
        String usuari = usernameField.getText();
        new chats(usuari, isAdmin, filtro).setVisible(true);
    } else {
        JOptionPane.showMessageDialog(null, "Usuari i/o Contrassenya Incorrectes", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

Aquesta funció és la que hem assignat pel botó de login i el 'enter' la qual agafa la informació del camp d'usuari i de la contrasenya i els comprova a la base de dades, en cas afirmatiu obra la pantalla de 'chats'

### 2.2 INTERFÍCIE



Hem creat la pantalla del login amb el JFrame, hem creat variables d'usuari i contrasenya i em fet que busqui introduït al mongodb, en cas de ser incorrecte apareix un missatge d'error i en cas contrari entra a la pantalla de conversacions.

## 3. CONVERSACIONS

### 3.1 CODI

```
public class chats extends javax.swing.JFrame {

    private DefaultListModel<String> messageListModel;
    private JList<String> messageList;
    private MongoClient mongoClient;
    private MongoDBDatabase database;
    private MongoCollection<Document> collection3;
    private MongoCollection<Document> collection2;
    private String usuari;
    private String xat;
    private boolean isAdmin;
    private boolean newConv;
    private Document filterouser;
    private ArrayList<String> xats;
    private ArrayList<String> xatsNom;

    public chats(String usuari2, boolean isAdmin, Document filtro) {
        initComponents();
        setTitle("Conversacions");

        this.usuari = usuari2;
        this.isAdmin = isAdmin;
        this.filterouser = filtro;
        mongoClient = MongoClient.create();
        database = mongoClient.getDatabase("whatsapp");
        collection3 = database.getCollection("grups");
        collection2 = database.getCollection("usuaris");

        messageListModel = new DefaultListModel<String>();
        messageList = new JList<String>(messageListModel);
        messageList.setFont(new Font(messageList.getFont().getName(), Font.PLAIN, 16));
        messageList.setAlignmentY(TOP_ALIGNMENT);
        jScrollPane1.setViewportView(messageList);

        recarregarConversacions();
    }
}
```

Hem creat una pantalla de conversacions on podem escollir entre diferents xats creats, en aquesta part del codi creem la connexió a mongodb igual que al login pero agafant la col·lecció grups.

#### 3.1.1 Carregar Conversacions

```
public void recarregarConversacions() {
    Document userDoc = collection2.find(filterouser).first();
    if (userDoc != null) {
        ArrayList<Object> xatsObject = userDoc.get("xats", ArrayList.class);
        xats = new ArrayList<>();
        xatsNom = new ArrayList<>();
        for (Object xatObject : xatsObject) {
            xats.add(String.valueOf(xatObject));
        }
    }
    Document query = new Document("_id", new Document("$in", xats));
    FindIterable<Document> conversacions = collection3.find(query);

    // Iterar sobre los xats y mostrarlos en la lista de mensajes
    messageListModel.clear();
    for (Document conversacio : conversacions) {
        String nom_grup = conversacio.getString("nom");
        xatsNom.add(nom_grup);
        messageListModel.addElement(nom_grup);
    }
}
```

La funció 'recarregarConversacions()' l'utilitzem per mostrar els xats que està unit l'usuari, buscant dintre de l'Array 'xats' i posem aquestes id en un ArrayList 'xats' el qual després utilitzem per fer una consulta a la col·lecció xats i que ens mostri els grups amb aquelles id, i per acabar els imprimim per pantalla.

### 3.1.2 Seleccionar Conversació

```

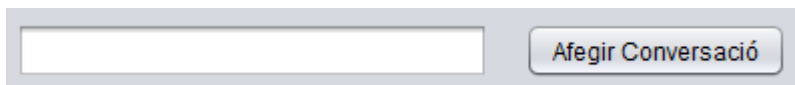
messageList.addListSelectionListener(new ListSelectionListener() {
    @Override
    public void valueChanged(ListSelectionEvent e) {
        if (!e.getValueIsAdjusting()) {
            int index = messageList.getSelectedIndex();
            if (index != -1) {
                if (newConv == true) {
                    addConvers();
                    recarregarConversacions();
                } else {
                    xat = "";
                    xat = messageList.getModel().getElementAt(index);
                }
            }
        }
    }
});

if (!isAdmin) {
    addXat.setVisible(false);
    nouXat.setVisible(false);
} else {
    addXat.setVisible(true);
    nouXat.setVisible(true);
}

```

El 'messageList' que veiem és per comprovar quin xat s'ha premut (el guardem en una variable per després utilitzar-la en cas de voler entrar al xat o eliminar-lo) , si el xat que s'ha premut l'usuari no s'havia unit, s'unirà cridant la funció addConvers() i després se li tornaran a mostrar els xats als quals està unit.

La condició de 'isAdmin' que tenim aquí és per comprovar si l'usuari es un administrador, en cas afirmatiu posarem en visible el 'addXat' i 'nouXat' els quals serveixen per crear un xat nou



The image shows a user interface element consisting of a rectangular text input field on the left and a button on the right. The button has a light blue gradient and the text 'Afegir Conversació' in a dark font. The entire element is set against a light gray background.



### 3.1.3 Afegir Conversació

```
private void addXatActionPerformed(java.awt.event.ActionEvent evt) {
    if(nouXat.getText().isEmpty()){
        return;
    }
    Document lastChat = collection3.find().sort(new Document("_id", -1)).first();
    int newId = 0;
    if (lastChat != null) {
        String lastId = lastChat.getString("_id");
        newId = Integer.parseInt(lastId) + 1;
    }

    String newNom = nouXat.getText();
    Document existingChat = collection3.find(new Document("nom", newNom)).first();
    if (existingChat != null) {
        JOptionPane.showMessageDialog(null, "El xat ja existeix", "Error", JOptionPane.ERROR_MESSAGE);
        return;
    }

    String newIdString = String.valueOf(newId);
    Document newChat = new Document("_id", newIdString)
        .append("nom", newNom);
    collection3.insertOne(newChat);
    nouXat.setText("");
    recarregarConversacions();
}
```

Aquesta funció serveix per crear el xat després de prémer el botó d'Afegir Conversació, el qual agafa l'última id dels xats i li suma 1, després agafa el nom que s'ha posat nou i es comprova que no existeixi, si no existeix crea el xat nou.

### 3.1.4 Unir-se a Conversació

```
private void joinXatActionPerformed(java.awt.event.ActionEvent evt) {
    if(newConv == false){
        Document query = new Document("_id", new Document("$nin", xats));
        FindIterable<Document> conversacions = collection3.find(query);
        newConv = true;
        conv.setText("CONVERSACIONS PER UNIR-SE");
        messageListModel.clear();
        for (Document conversacio : conversacions) {
            String nom_grup = conversacio.getString("nom");
            messageListModel.addElement(nom_grup);
        }
    }else{
        newConv = false;
        conv.setText("LES TEVES CONVERSACIONS");
        recarregarConversacions();
    }
}
```

Aquesta funció es la que cridem al prémer el botó de unir-se a conversacions, si fem un click se'ns obrirà la pantalla de conversacions que encara no ens hem unit, i si tornem a fer un altre ens tornarà a la pantalla inicial amb les nostres conversacions.

Per fer això hem utilitzat un booleà, després per mostrar les converses que no ens hem unit hem consultat a la base de dades tots els xats que no estan dins del array de 'xats' on estan tots els nostres xats.

### 3.1.5 Esborrar/Entrar Conversa

```
private void deleteXatActionPerformed(java.awt.event.ActionEvent evt) {
    String xatSeleccionado = messageList.getSelectedValue();
    Document filtro = new Document("nom", xatSeleccionado);
    Document chatDoc = collection3.find(filtro).first();
    String id_xat;
    if (chatDoc != null) {
        id_xat = chatDoc.getString("_id");
    } else {
        id_xat = "";
    }
    if (xatSeleccionado != null) {
        Document filtro2 = new Document("usuari", usuari);
        Document update = new Document("$pull", new Document("xats", id_xat));
        collection2.updateOne(filtro2, update);
        recarregarConversacions();
    }
}

private void entrarXatActionPerformed(java.awt.event.ActionEvent evt) {
    if(xat != null){
        String xatSeleccionado = messageList.getSelectedValue();

        if (xatsNom.contains(xatSeleccionado)) {
            dispose();
            new WhatsappVista(usuari, xat, isAdmin, filtrouser).setVisible(true);
        }else{
            JOptionPane.showMessageDialog(null, "No estàs unit al xat seleccionat.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }else{
        JOptionPane.showMessageDialog(null, "No has seleccionat cap xat.", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
}
```

Per borrar/entrar a un xat agafem la variable que hem guardat abans amb el xat que hem seleccionat per esborrar-lo agafem la id del xat seleccionat i li esborrem del array de xats de la base de dades.

I per entrar al xat simplement comprovem que l'usuari està unit al xat seleccionat i en cas afirmatiu entrem a la següent pantalla del xat.

### 3.1.6 Afegir Conversa

```

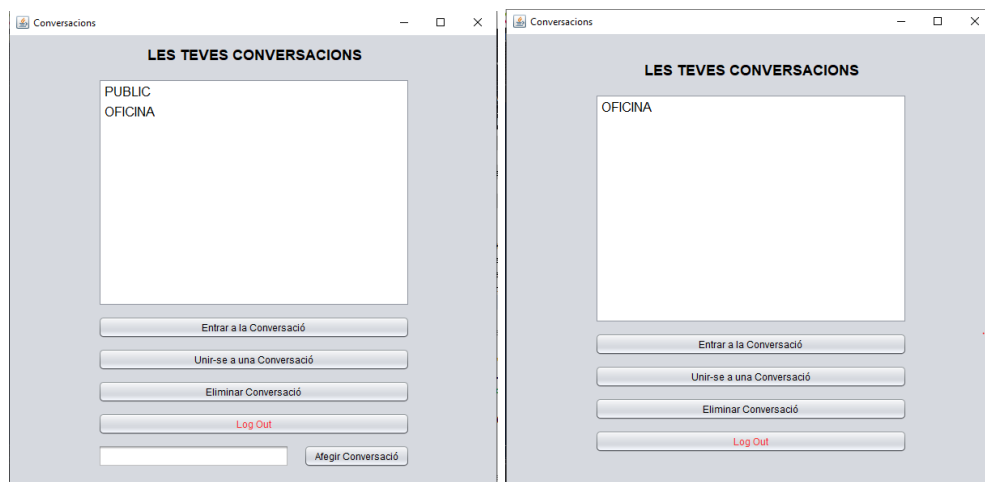
public void addConvers() {
    String xatSeleccionado = messageList.getSelectedValue();
    String id_xat;
    if (xatSeleccionado != null) {
        Document filtro = new Document("nom", xatSeleccionado);
        Document chatDoc = collection3.find(filtro).first();
        if (chatDoc != null) {
            id_xat = chatDoc.getString("_id");
        } else {
            id_xat = "";
        }
        if (id_xat != null) {
            Document filtro2 = new Document("usuari", usuari);
            Document update = new Document("$addToSet", new Document("xats", id_xat));
            collection2.updateOne(filtro2, update);
            newConv = false;
        } else {
            JOptionPane.showMessageDialog(null, "El chat seleccionado no existe.", "Error", JOptionPane.ERROR_MESSAGE);
        }
    }
}

```

Aquesta funció la cridem a l'hora de unir-nos a una conversa nova, la qual agafa el xat seleccionat, el busca pel nom, agafa la seva id i la inserta en l'array de xats de l'usuari

## 3.2 INTERFÍCIE

### 3.2.1 Pantalla Inicial



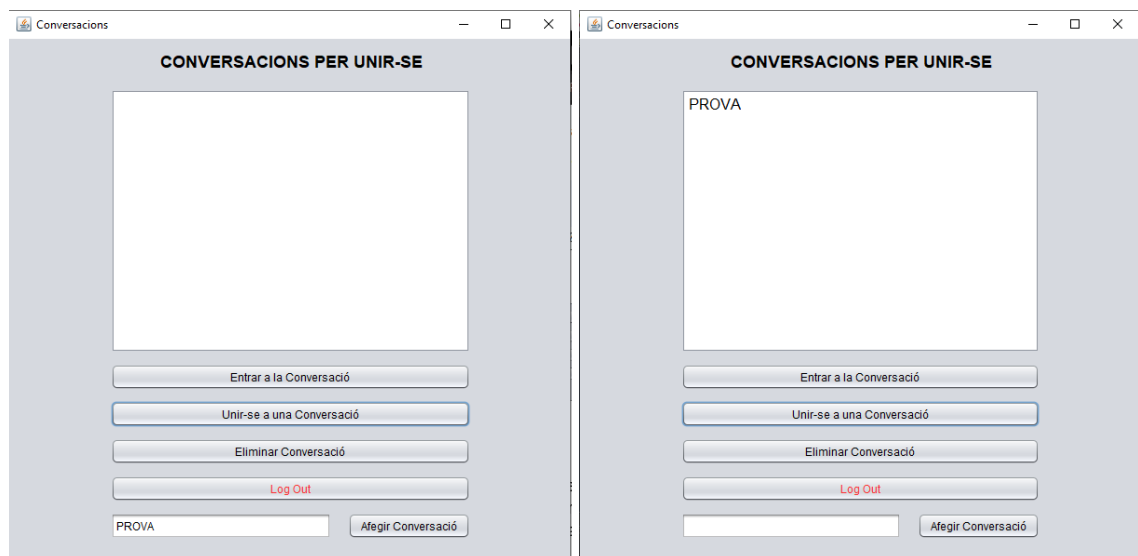
**Pantalla esquerra -> Admin**

**Pantalla dreta -> No Admin**

A la pantalla principal podem veure els xats als que està unit l'usuari, i els diferents botons disponibles, el primer serveix per entrar a la conversa que seleccionem, el segon ens mostrarà tots els xats als que no ens hem unit encara, el tercer servirà per eliminar la conversa seleccionada (ens podem tornar a unir si volem després), i l'últim botó per anar a la pantalla d'inici un altre cop.

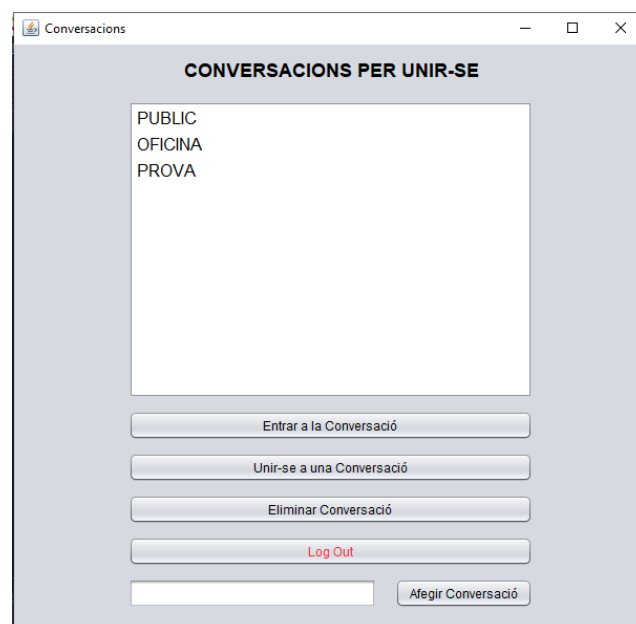
L'última opció serveix per crear una conversa la qual sol està disponible pels administradors.

### 3.2.2 Crear Conversa



Com podem veure a la imatge de l'esquerra no existeix cap xat al que no estiguem units, però després de crear el xat PROVA podem veure (imatge dreta) com apareix.

Si li fem clic a sobre se'ns unirà al xat i ja ens apareixerà a la pantalla inicial com podem veure abaix



Quan fem clic a una conversa i premem el botó de Eliminar Conversació, desapareixerà de la pantalla inicial i haurem de tornar a unir-nos si volem entrar al xat.

I per entrar a qualsevol conversa primer haurem de fer clic sobre la conversa a la que ens vulguem unir i prémer 'Entrar a la Conversació'

## 4. XAT

### 4.1 CODI

#### 4.1.1 Carregar Missatges d'Avui

```
query.put("hora", new BasicDBObject("$gte", startOfDay).append("$lte", endOfDay));
query.put("xat", new Document("$eq", xat));
FindIterable<Document> messages = collection.find(query);
for (Document message : messages) {
    String user = message.getString("user");
    String text = message.getString("missatge");
    String hora = horaFormat.format(message.getDate("hora"));
    avui = dateFormat.format(message.getDate("hora"));
    messageListModel.addElement(user + ": " + text + " (" + hora + ")");
}
```

Al entrar al xat afegim tots els missatges del dia d'Avui, els quals obtenim de la base de dades.

### 4.1.2 Timer per Actualitzar Missatges

```

timer = new Timer(100, new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(missatge.getText().isEmpty()){
            sendButton.setEnabled(false);
        }else{
            sendButton.setEnabled(true);
        }
        if(selectedDate == null){
            selectedDate = new Date();
        }

        Calendar calendar = Calendar.getInstance();
        calendar.setTime(selectedDate);
        calendar.set(Calendar.HOUR_OF_DAY, 0);
        calendar.set(Calendar.MINUTE, 0);
        calendar.set(Calendar.SECOND, 0);
        Date startOfDay = calendar.getTime();
        calendar.set(Calendar.HOUR_OF_DAY, 23);
        calendar.set(Calendar.MINUTE, 59);
        calendar.set(Calendar.SECOND, 59);
        Date endOfDay = calendar.getTime();
        // Añadir el filtro a la consulta existente

        cargarPerfil();

        query.put("hora", new BasicDBObject("$gte", startOfDay).append("$lte", endOfDay));
        query.put("xat", new Document("$eq", xat));
        FindIterable<Document> messages = collection.find(query);
        long count = collection.countDocuments(query);
        if (count!=missatges_antics){
            messageListModel.clear();
            for (Document message : messages) {
                String user = message.getString("user");
                String text = message.getString("missatge");
                String dateStr = horaFormat.format(message.getDate("hora"));
                avui = dateFormat.format(message.getDate("hora"));
                messageListModel.addElement(user + ": " + text + " (" + dateStr + ")");
            }
            missatges_antics = count;
        }
    }
}

```

Hem creat un timer per actualitzar la llista de missatges en cas de que envien un missatge nou, així tenir en temps real el llistat de missatges.

Abans de tornar a carregar els missatges comprovem que hi hagin missatges nous sino no recarreguem la informació de la llista, per així no saturar-ho.

També habiliten o deshabiliten el botó d'enviar en cas de que estigui buit o no el contingut del text.

### 4.1.3 Obtenir Dades Missatge

```
messageList.addListSelectionListener(new ListSelectionListener() {  
    @Override  
    public void valueChanged(ListSelectionEvent e) {  
        if (!e.getValueIsAdjusting()) {  
            int index = messageList.getSelectedIndex();  
            if (index != -1) {  
                String selectedMessage = messageList.getModel().getElementAt(index);  
                String[] parts = selectedMessage.split(":");  
  
                if (parts.length > 1) {  
                    usuariEl = parts[0];  
                    missatgeEl = parts[1].substring(0, parts[1].indexOf("(")).trim();  
                    horaEl = selectedMessage.substring(selectedMessage.lastIndexOf("(") + 1, selectedMessage.lastInd  
                }  
            }  
        }  
    }  
});
```

Aquí guardem dins les variables 'usuariEl', 'missatgeEl' i 'horaEl' el contingut del missatge seleccionat per posteriorment utilitzar aquestes variables per eliminar el missatge.

#### 4.1.4 Enviar Missatge

```
missatge.addKeyListener(new KeyAdapter() {
    public void keyPressed(KeyEvent e) {
        if (e.getKeyCode() == KeyEvent.VK_ENTER && !missatge.getText().isEmpty()) {
            String user = usuari;
            String message = missatge.getText();
            String dateStr = horaFormat.format(new Date());
            Document doc = new Document("user", user)
                .append("missatge", message)
                .append("hora", new Date())
                .append("xat", nomxat.getText());
            collection.insertOne(doc);
            if(selectedDate != new Date()){
            }else{
                messageListModel.addElement(user + ": " + message + " (" + dateStr + ")");
            }
            missatge.setText("");
        }
    }
});

private void sendButtonActionPerformed(java.awt.event.ActionEvent evt) {
    if(missatge.getText().isEmpty())
        return;

    String user = usuari;
    String message = missatge.getText();
    String dateStr = horaFormat.format(new Date());
    Document doc = new Document("user", user)
        .append("missatge", message)
        .append("hora", new Date())
        .append("xat", nomxat.getText());
    collection.insertOne(doc);
    if(selectedDate != new Date()){
    }else{
        messageListModel.addElement(user + ": " + message + " (" + dateStr + ")");
    }
    missatge.setText("");
}
}
```

Aquí agafem el missatge que escriu l'usuari i el guardem a la base de dades, amb el usuari, el missatge, la hora i el xat en el que s'ha enviat.



### 4.1.5 Calendari

```
private void jCalendar1PropertyChange(java.beans.PropertyChangeEvent evt) {

    if ("calendar".equals(evt.getPropertyName())) {
        selectedDate = jCalendar1.getCalendar().getTime();
        // Usa la fecha seleccionada en lo que necesites hacer a continuación
        SimpleDateFormat dateFormat = new SimpleDateFormat("dd/MM/yyyy");

        String dateStr = dateFormat.format(selectedDate);
        dia.setText(dateStr);

        // Crear una consulta para buscar los mensajes con la fecha seleccionada

        Calendar calendar = Calendar.getInstance();
        calendar.setTime(selectedDate);
        calendar.set(Calendar.HOUR_OF_DAY, 0);
        calendar.set(Calendar.MINUTE, 0);
        calendar.set(Calendar.SECOND, 0);
        Date startOfDay = calendar.getTime();
        calendar.set(Calendar.HOUR_OF_DAY, 23);
        calendar.set(Calendar.MINUTE, 59);
        calendar.set(Calendar.SECOND, 59);
        Date endOfDay = calendar.getTime();

        // Añadir el filtro a la consulta existente
        query.put("hora", new BasicDBObject("$gte", startOfDay).append("$lte", endOfDay));
        query.put("xat", new Document("$eq", xat));

        // Obtener los mensajes que coinciden con la consulta
        FindIterable<Document> messages = collection.find(query);

        // Añadir los mensajes al modelo de la lista
        messageListModel.clear();
        for (Document message : messages) {
            String user = message.getString("user");
            String text = message.getString("missatge");
            String messageDateStr = horaFormat.format(message.getDate("hora"));
            messageListModel.addElement(user + ": " + text + " (" + messageDateStr + ")");
        }
    }
}
```

Aquesta funció la cridem a l'hora de canviar el dia al jCalendar, i agafem el dia seleccionat i busquem els missatges d'aquell dia a la base de dades i els mostrem.

### 4.1.6 Eliminar Missatge

```
private void jButton1MouseClicked(java.awt.event.MouseEvent evt) {
    if(usuariEl == null || usuariEl == "")
        JOptionPane.showMessageDialog(null, "No has seleccionat cap missatge", "Error", JOptionPane.ERROR_MESSAGE);
    else{
        if(usuariEl.equals(usuari)){
            String diaMissatgeString = avui + " " + horaEl;
            SimpleDateFormat sdf = new SimpleDateFormat("dd/MM/yyyy HH:mm:ss");
            try {
                Date diaMissatge = sdf.parse(diaMissatgeString);
                BasicDBObject query = new BasicDBObject();
                query.put("user", usuariEl);
                query.put("missatge", missatgeEl);
                query.put("hora", new BasicDBObject("$gte", diaMissatge));
                collection.deleteOne(query);

                JOptionPane.showMessageDialog(null, "Missatge eliminat amb èxit", "Informació", JOptionPane.INFORMATION_
                usuariEl = "";
            } catch (ParseException e) {
                e.printStackTrace();
            }
        }else{
            JOptionPane.showMessageDialog(null, "No Pots Borrar Missatges de Altres Persones", "Error", JOptionPane.ERRC
        }
    }
}
```

Aquesta funció es la que cridem al esborrar un missatge seleccionat.

Primer comprovem que hi hagi un missatge seleccionat comprovant que la variable 'd'usuariEl' no estigui buit (és una de les variables que hem creat abans per guardar les dades del missatge seleccionat).

Si s'ha seleccionat un missatge comprovem que l'usuari del missatge es el mateix que el que ha iniciat sessió, si és així esborrem el missatge de la base de dades, agafant les dades de les variables anteriors.

### 4.1.7 Mode Diürn/Nocturn(Beans Propis)

```
bean_prova2.addModeChangeListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        newColor = bean_prova2.getMode();
        getContentPane().setBackground(newColor);
        Color newColor2 = bean_prova2.getMode2();
        miss.setForeground(newColor2);
        nomxat.setForeground(newColor2);
        titol.setForeground(newColor2);
        dia.setForeground(newColor2);
        calendari.setForeground(newColor2);
    }
});
```

Aquí comprovem que s'hagi premut el botó amb el actionPerformed i si es hagi canviar el color de la lletra i del fons.

### 4.1.8 Foto Perfil (Bean Propi)

```

public void cargarPerfil(){
    Document usuarioDocument = collection2.find(filtrouser).first();
    if (usuarioDocument != null) {
        // Obtener el valor del campo "img" como un objeto Binary
        Binary imageBinary = usuarioDocument.get("img", Binary.class);
        if (imageBinary != null) {
            try {
                byte[] imageBytes = imageBinary.getData();
                ByteArrayInputStream bis = new ByteArrayInputStream(imageBytes);
                BufferedImage image = ImageIO.read(bis);
                int maxIconSize = 32;
                int width = image.getWidth();
                int height = image.getHeight();
                int newWidth = width;
                int newHeight = height;
                if (width > maxIconSize || height > maxIconSize) {
                    if (width > height) {
                        newWidth = maxIconSize;
                        newHeight = (int) (height / (double) width * maxIconSize);
                    } else {
                        newHeight = maxIconSize;
                        newWidth = (int) (width / (double) height * maxIconSize);
                    }
                }
                scaledImage = image.getScaledInstance(newWidth, newHeight, Image.SCALE_SMOOTH);
                icon = new ImageIcon(scaledImage);
                imagelabel.setIcon(icon);
            } catch (IOException e2) {
                e2.printStackTrace();
            }
        } else {
            icon = new ImageIcon("user.png");
            imagelabel.setIcon(icon);
        }
    }
}

```

Aquesta funció es la que utilitzem per actualitzar la foto de perfil al xat, buscant el binary que guardem a la base de dades i el posem en un JLabel, en cas de no tenir foto de perfil posem una icona per default.

```

uploadPhoto1.addActionListener(new ActionListener() {
    @Override
    public void actionPerformed(ActionEvent e) {
        if(uploadPhoto1.getFileBytes() != null){
            img = uploadPhoto1.getFileBytes();
            uploadPhoto();
        }else{
        }
    }
});

```

Aquí comprovem si el bean de 'uploadPhoto1' ens retorna algo, en cas afirmatiu cridem a la funció 'uploadPhoto()' i guardem el que ens torna a la variable img.

```

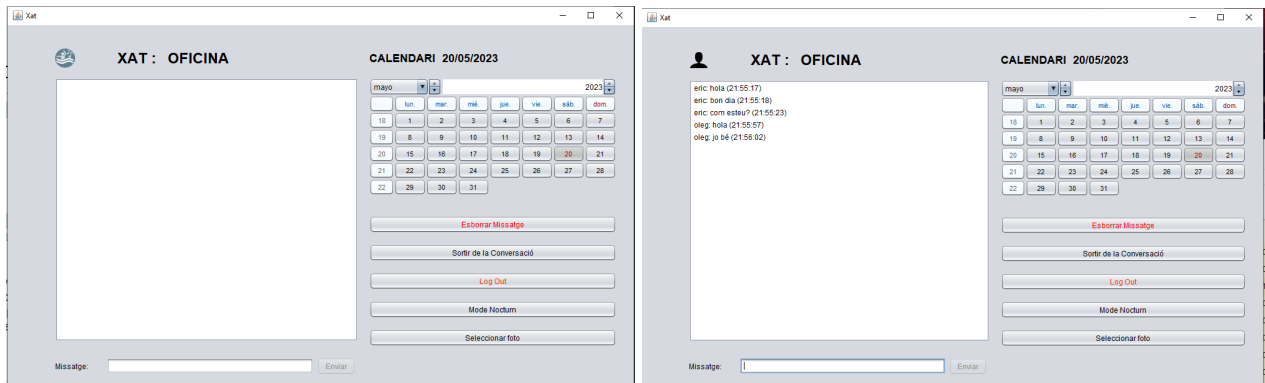
public void uploadPhoto(){
    Document usuarioDocument = collection2.find(filtrouser).first();
    if (usuarioDocument != null) {
        if (img != null) {
            // Guardar los bytes de la nueva imagen en MongoDB
            usuarioDocument.put("img", img);
            collection2.replaceOne(filtrouser, usuarioDocument);
        }
    }
}

```

I aquesta funció és la que utilitzem per pujar la foto del bean a la base de dades.

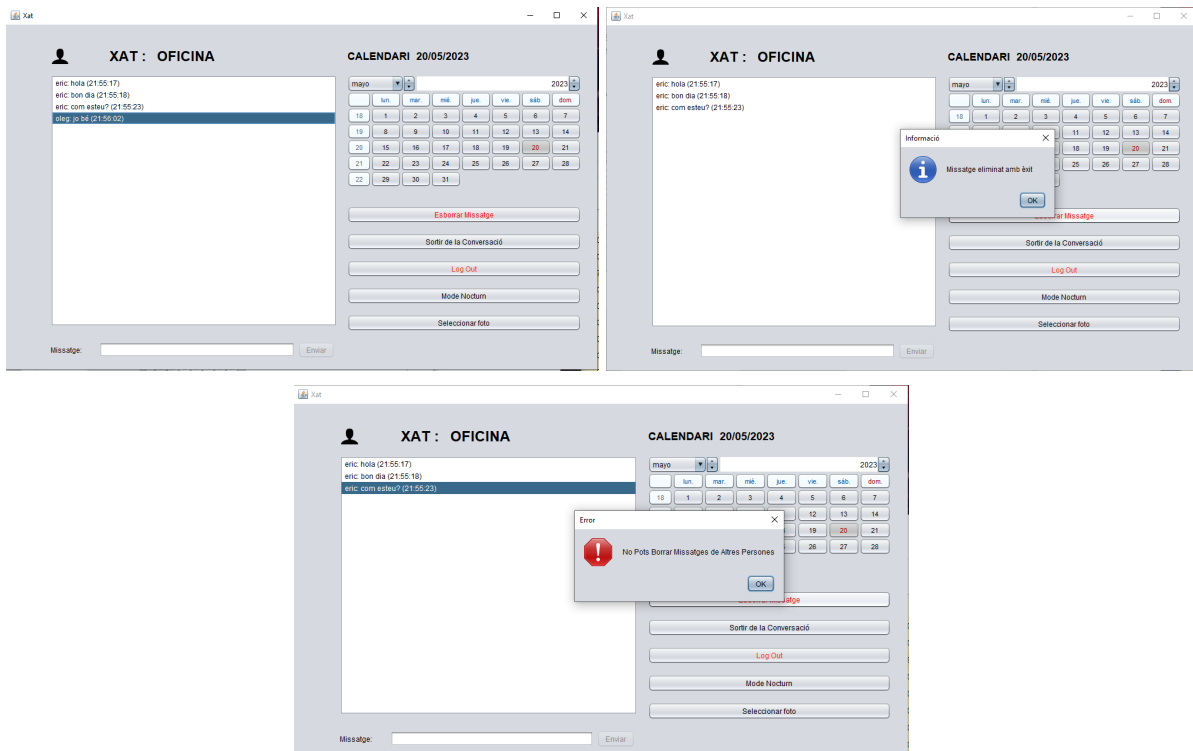
## 4.2 INTERFÍCIE

### 4.2.1 Pantalla Inicial



A la pestanya de Xat, podem comprobar que tenim un jCalendar, on podem escollir el dia que volem visualitzar els missatges, esborrar el missatge Seleccionat, sortir de la conversa per escollir una altra, fer Log Out, canviar el mode de diürn a nocturn i viceversa i canviar la foto de perfil, a més a més podem enviar un missatge prement el botó d'enviar el cual s'activa al escriure algo o amb la tecla enter.

### 4.2.2 Esborrar Missatge



A l'hora d'eliminar un missatge comprovem que el missatge sigui del usuari, i en cas afirmatiu l'elimini i aparegui el següent missatge, i en cas negatiu aparegui un missatge d'error.

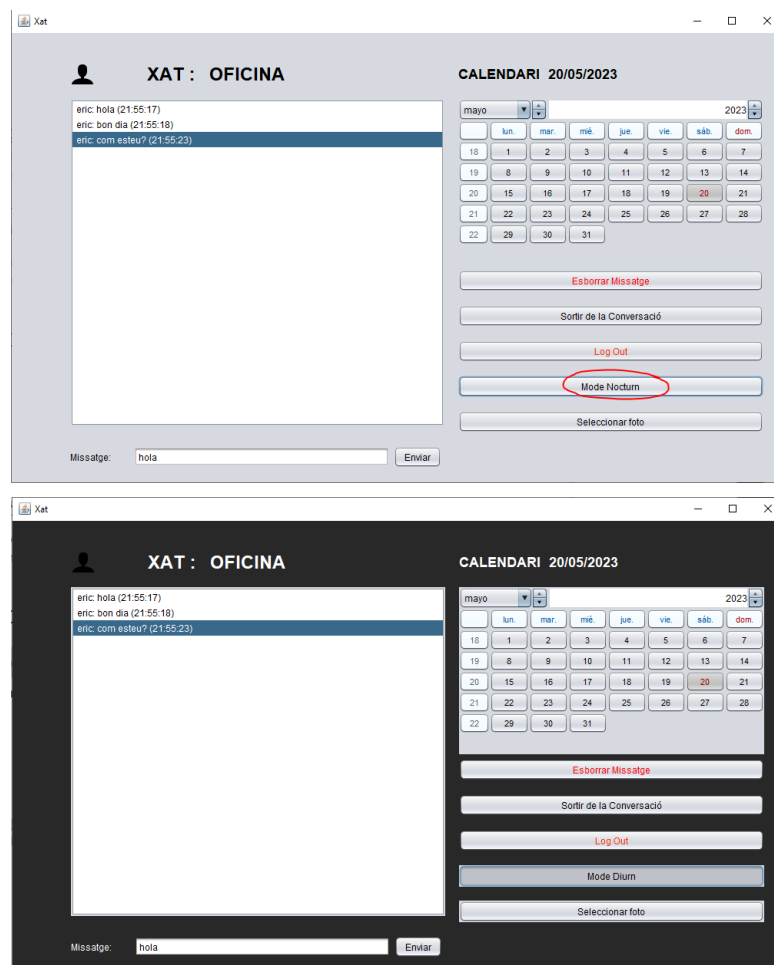
### 4.2.3 Missatge

Missatge:

A l'hora d'enviar missatges com podem veure el botó d'Enviar està desactivat si no tenim res escrit i en cas contrari se'ns activa.

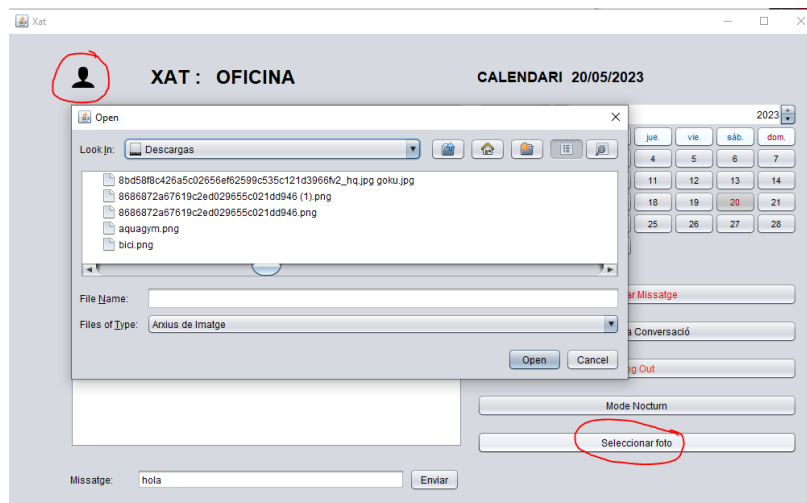
Missatge:

### 4.2.4 Mode Diürn/Nocturn



Un dels beans que hem creat és per canviar el fons a fosc, com si un mode nocturn i diürn.

### 4.2.5 Canviar Foto Perfil



Un altre bean dels que hem fet és per canviar la foto de perfil de l'usuari, encara que sol la pot veure el propi usuari actualment.

## 5. BEAN

### 5.1 CODI

#### 5.1.1 Bean Mode Diürn/Nocturn

```
package bean_final;

import javax.swing.*;

public class Bean_prova extends JPanel {
    private JToggleButton toggleButton;
    private Color backgroundColor;
    private Color newColor;
    private Color newColor2;
    private boolean isNightMode;
    private List<ActionListener> modeChangeListener;

    public Bean_prova() {
        setLayout(new BorderLayout());
        newColor = null;
        toggleButton = new JToggleButton("Mode Nocturn");
        toggleButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                isNightMode = toggleButton.isSelected();
                if (isNightMode) {
                    newColor = new Color(40, 40, 40);
                    newColor2 = Color.WHITE;
                    toggleButton.setText("Mode Diürn");
                    fireModeChangeEvent();
                } else {
                    newColor = null;
                    newColor2 = Color.BLACK;
                    toggleButton.setText("Mode Nocturn");
                    fireModeChangeEvent();
                }
            }
        });

        modeChangeListener = new ArrayList<>();

        add(toggleButton, BorderLayout.CENTER);
    }

    public Color getMode() {
        return newColor;
    }

    public Color getMode2() {
        return newColor2;
    }

    public void addModeChangeListener(ActionListener listener) {
        modeChangeListener.add(listener);
    }

    public void removeModeChangeListener(ActionListener listener) {
        modeChangeListener.remove(listener);
    }

    private void fireModeChangeEvent() {
        ActionEvent event = new ActionEvent(this, ActionEvent.ACTION_PERFORMED, "modeChange");
        for (ActionListener listener : modeChangeListener) {
            listener.actionPerformed(event);
        }
    }
}
```

L'únic que hem fet en aquest bean és crear un 'ToggleButton' el qual alterna entre diürn el qual retorna el color 'null' per posar-ho com default i el nocturn que fa que retorni el color en rgb(40,40,40) que és semblant al negre.

També retorna un altre color que és el que utilitzem després per canviar el color de la lletra entre blanc o negre segons el mode en el que es troba.

També hem creat un 'addModeChangeListener' que és com un addActionListener per poder utilitzar en el nostre disseny i saber quan hem clicat el botó.

### 5.1.2 Bean Foto Perfil

```
package bean_photo;

import javax.swing.*;

public class UploadPhoto extends JPanel {
    private JButton uploadButton;
    private byte[] selectedFileBytes;
    private ActionListener actionListener;

    public UploadPhoto() {
        setLayout(new BorderLayout());

        uploadButton = new JButton("Seleccionar foto");
        uploadButton.addActionListener(new ActionListener() {
            @Override
            public void actionPerformed(ActionEvent e) {
                JFileChooser fileChooser = new JFileChooser();

                FileNameExtensionFilter filter = new FileNameExtensionFilter("Arxius de Imatge", "png", "jpg");
                fileChooser.setFileFilter(filter);

                int result = fileChooser.showOpenDialog(UploadPhoto.this);

                if (result == JFileChooser.APPROVE_OPTION) {
                    File selectedFile = fileChooser.getSelectedFile();
                    selectedFileBytes = readFileBytes(selectedFile);

                    if (actionListener != null) {
                        actionListener.actionPerformed(e);
                    }
                }
            }
        });

        add(uploadButton);
    }

    public byte[] getFileBytes() {
        return selectedFileBytes;
    }

    public void addActionListener(ActionListener listener) {
        actionListener = listener;
    }

    private byte[] readFileBytes(File file) {
        try {
            FileInputStream fis = new FileInputStream(file);
            ByteArrayOutputStream bos = new ByteArrayOutputStream();

            byte[] buffer = new byte[1024];
            int bytesRead;
            while ((bytesRead = fis.read(buffer)) != -1) {
                bos.write(buffer, 0, bytesRead);
            }

            fis.close();
            bos.close();

            return bos.toByteArray();
        }
    }
}
```

Per fer aquest bean hem creat un botó el qual al clicar ens obre una nova pestanya per poder seleccionar fitxers png i jpg del nostre ordenador.

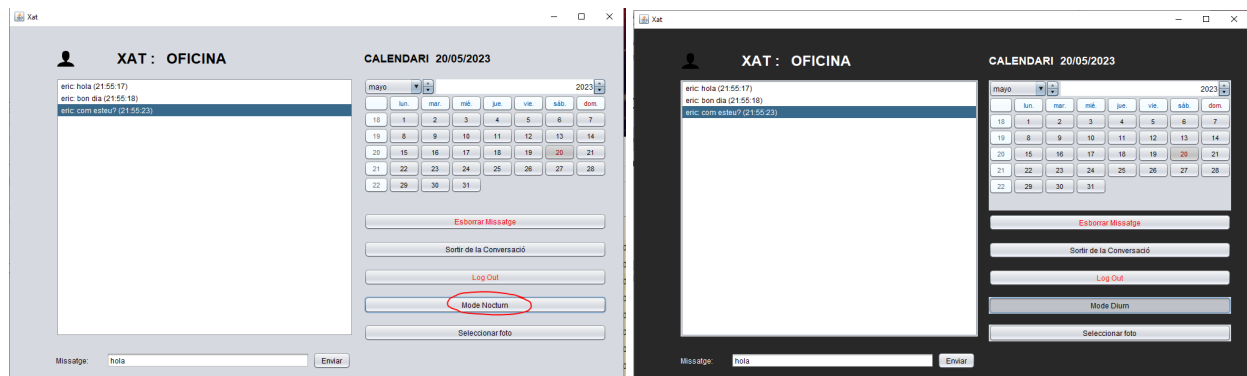
Després de seleccionar una imatge ens retorna una cadena de bytes la qual agafem en el nostre disseny i la pugem a la base de dades per després transformar-ho a una imatge

També creem un addActionListener per poder cridar el getFileBytes desde el nostre disseny al fer clic al botó que hem creat.



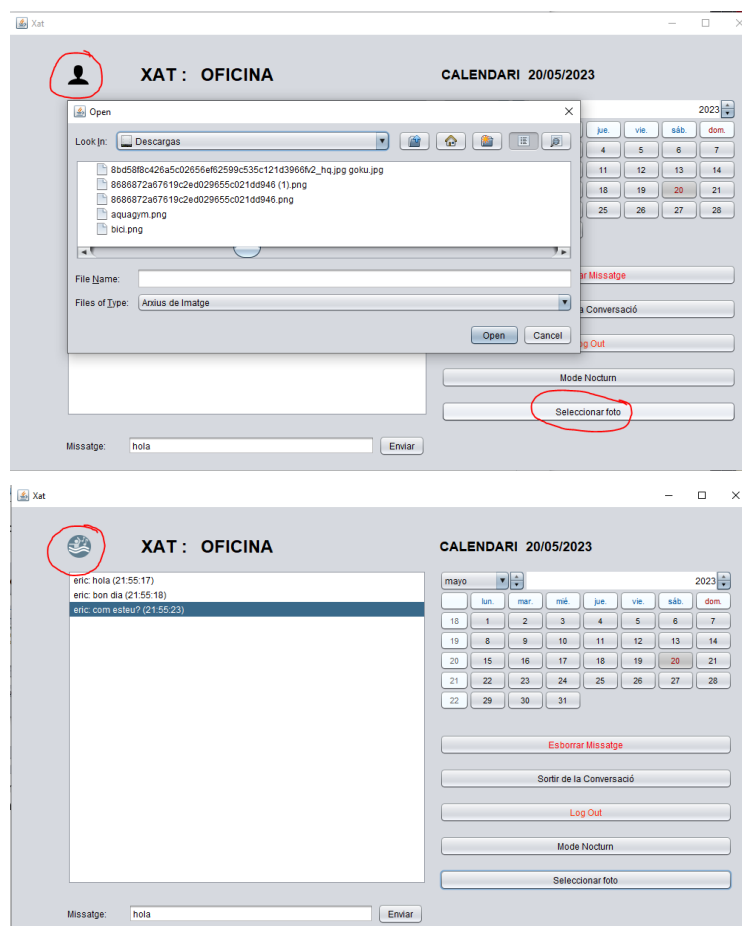
## 5.2 INTERFÍCIE

### 5.2.1 Bean Mode Diürn/Nocturn



Aquí podem veure com al fer clic a Mode Nocturn ens canvia el fons a fosc i viceversa

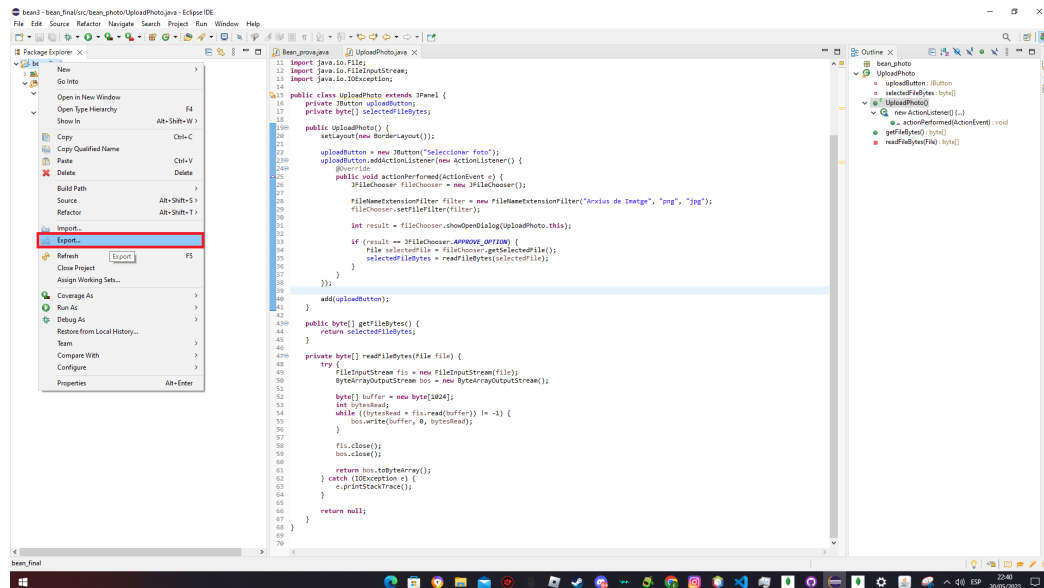
### 5.2.2 Bean Foto Perfil



I el bean per canviar la foto de perfil podem veure que al fer clic a l'opció Seleccionar foto se'ns obra una pantalla nova per poder seleccionar la nostra nova foto de perfil.

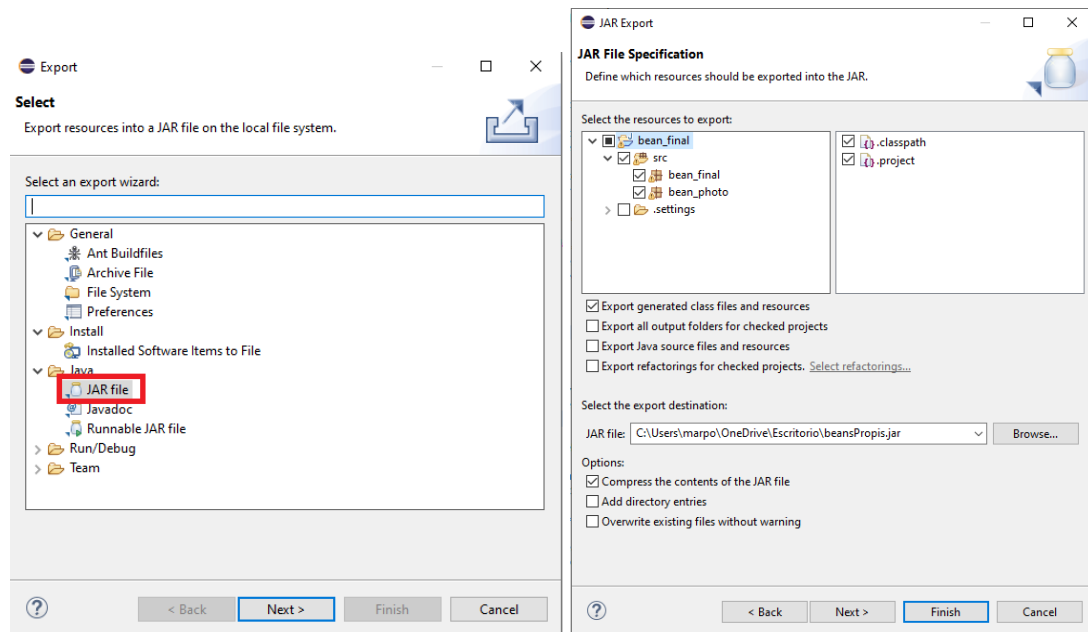
## 5.3 CREAR BEAN

### 5.3.1 Primer Pas



Primer creem el nostre bean i fem clic dret al nostre projecte i fem clic a exportar

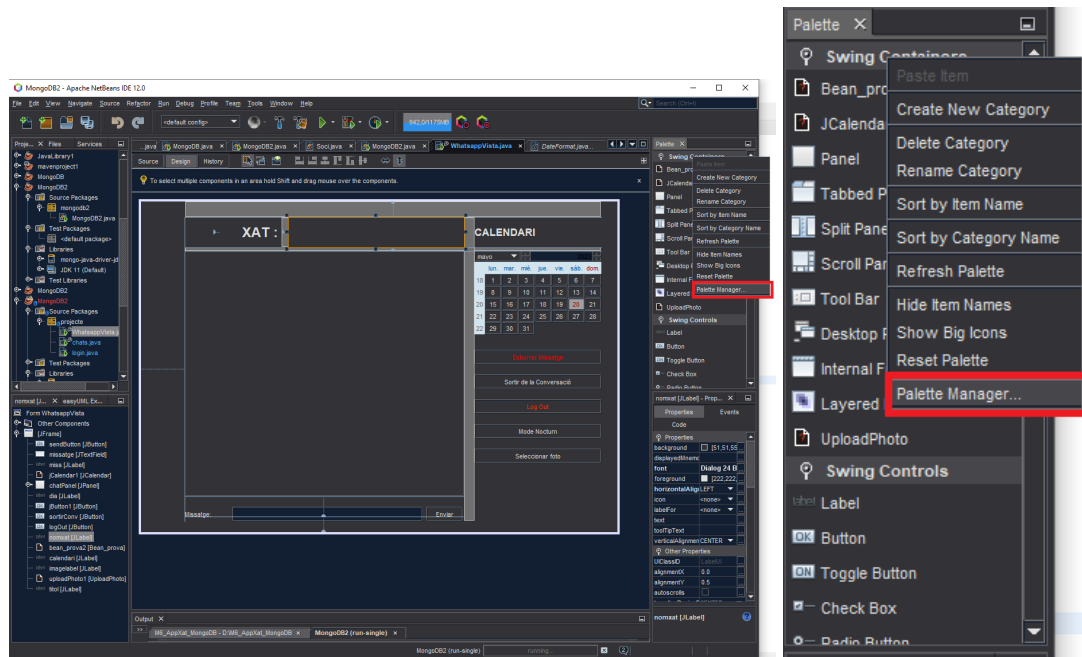
### 5.3.2 Segon Pas



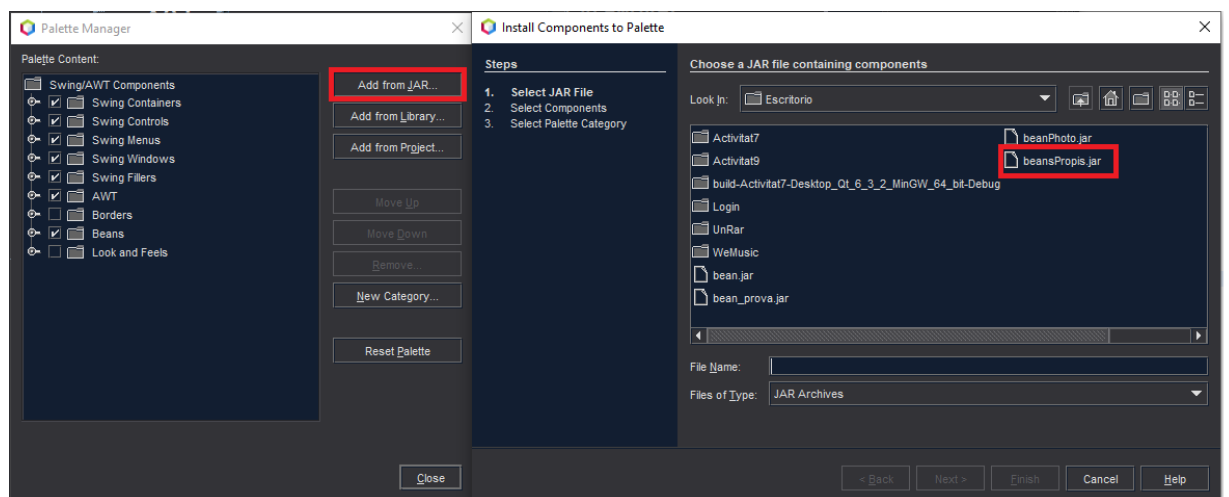
Fem clic on diu JAR file, i seleccionem els nostres beans que hem creat anteriorment i els guardem amb el nom de .jar que vulguem

### 5.3.3 Tercer Pas

Un cop creat el JAR anem al nostre disseny i fem clic dret a la nostra paleta on diu Swing Containers

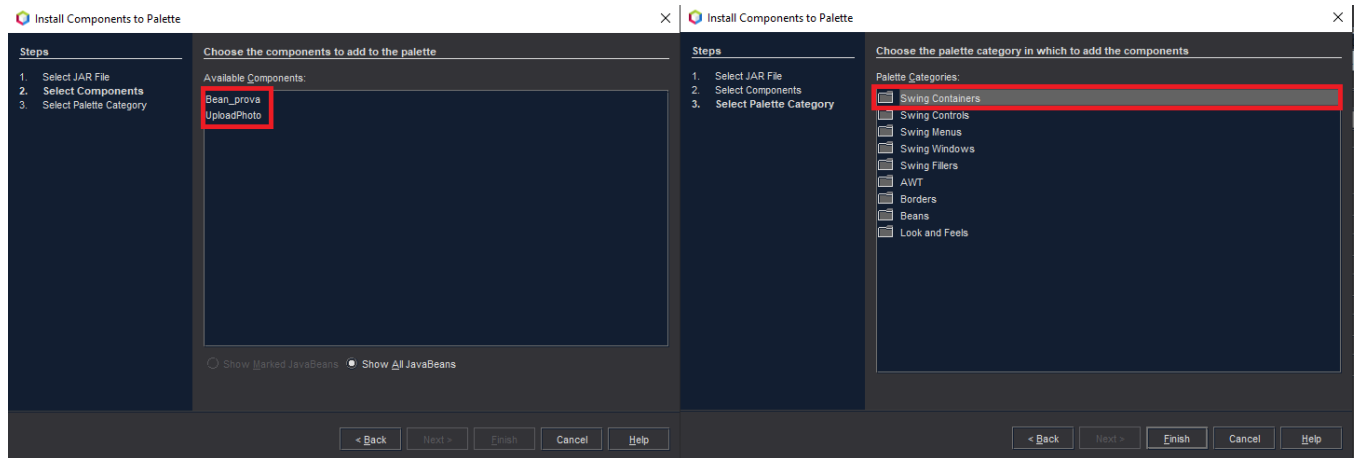


### 5.3.4 Quart Pas



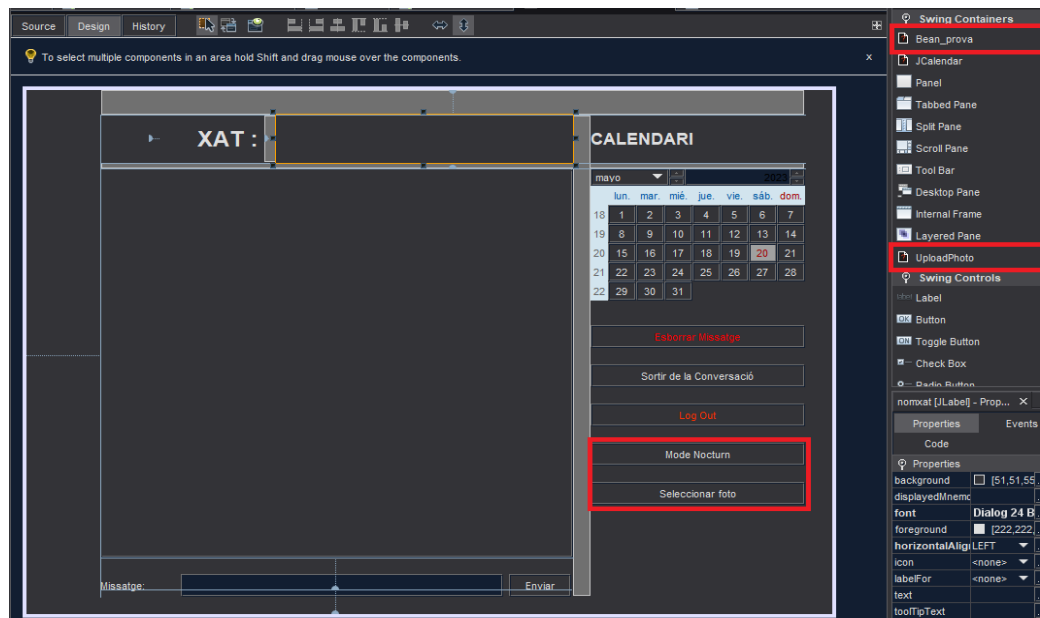
Fem clic on diu 'Add from JAR...' i seleccionem el jar que hem creat

### 5.3.5 Cinquè Pas



Escollim el bean que vulguem afegir i el posem a la carpeta/secció que vulguem nosaltres, en aquest cas en la primera

### 5.3.5 Últim Pas



Quan haguem fet tot lo anterior ens apareixeran els beans al apartat que havíem seleccionat i ja podríem arrastrar-los al nostre disseny