**Visual OETPN Simulator**

**DESCRIEREA DEZVOLTĂRII APLICAȚIEI**

Autor: **Dragu Andreea-Ioana**

Conducător științific: **SL. Dr. Ing. Octavian Cuibus**

Cuprins

[1 Introducere 3](#_Toc100340245)

[1.1 Descrierea succintă a aplicației 3](#_Toc100340246)

[1.2 Obiective 3](#_Toc100340247)

[2 Mediile de dezvoltare, tehnologiile și limbajele utilizate 4](#_Toc100340248)

[3 Realizarea aplicației 5](#_Toc100340249)

[3.1 Sign Up Frame 5](#_Toc100340250)

[3.1.1 Scurtă descriere 5](#_Toc100340251)

[3.1.2 Descrierea fiecărei componente în parte 8](#_Toc100340252)

[**3.1.2.1** **JPanel-ul WVOS\_Signup** 8](#_Toc100340253)

[**3.1.2.2** **JPanel-ul SignupForm** 11](#_Toc100340254)

[3.1.3 Codul 13](#_Toc100340255)

[**3.1.3.1** **Librăriile necesare** 13](#_Toc100340256)

[**3.1.3.2** **Clasa SignupFrame** 13](#_Toc100340257)

[**3.1.3.3** **Variabilele definite de utilizator** 13](#_Toc100340258)

[**3.1.3.4** **Variabilele predefinite** 13](#_Toc100340259)

[**3.1.3.5** **Constructorul clasei SignupFrame:** public SignupFrame() 14](#_Toc100340260)

[**3.1.3.6** **Funcția pentru setarea stelelor / marker-elor de completare:** private void setStarsInvisible() 15](#_Toc100340261)

[**3.1.3.7** **Funcția care marchează câmpurile goale:** private boolean markEmptyFields() 15](#_Toc100340262)

[**3.1.3.8** **Funcția pentru verificarea parolelor (dacă acestea coincid):** private boolean matchingPasswords() 16](#_Toc100340263)

[**3.1.3.9** **Funcția pentru inițializarea tuturor componentelor din SignupFrame: private** void initComponents() 16](#_Toc100340264)

[**3.1.3.10** **Eveniment generat în momentul în care se apasă butonul Sign Up:** private void SignupButtonActionPerformed(java.awt.event.ActionEvent evt) 17](#_Toc100340265)

[**3.1.3.11** **Eveniment generat în momentul în care se apasă butonul HaveAccButton:** private void HaveAccButtonActionPerformed(java.awt.event.ActionEvent evt) 17](#_Toc100340266)

[**3.1.3.12** **Funcția pentru setarea elementului JOptionPane:** private void setJOptionPane() 18](#_Toc100340267)

[**3.1.3.13** **Funcția pentru setarea hyperlink-ului către pagina Home (VOS Web**): private void setHomepageHyperlink() 18](#_Toc100340268)

[**3.1.3.14** **Funcția pentru setarea hyperlink-ului către pagina About OETPNs (VOS Web):** private void setAboutOETPNHyperlink() 18](#_Toc100340269)

[**3.1.3.15** **Funcția pentru setarea hyperlink-ului către pagina About VOS (VOS Web):** private void setAboutVOSHyperlink() 19](#_Toc100340270)

[**3.1.3.16** **Funcția pentru setarea hyperlink-ului către profilul personal de GitHub al dezvoltatorului:** private void setGithubProfileHyperlink() 19](#_Toc100340271)

[**3.1.3.17** **Funcția pentru setarea hyperlink-ului către profilul personal de LinkedIn al dezvoltatorului:** private void setLinkedinProfileHyperlink() 20](#_Toc100340272)

[**3.1.3.18** **Funcția pentru setarea opțiunii de trimitere a unui e-mail către creatorul aplicației:** private void setEmailOption() 20](#_Toc100340273)

[**3.1.3.19** **Metoda main:** public static void main(String[] args) 21](#_Toc100340274)

[3.2 Login Frame 22](#_Toc100340275)

[3.2.1 Scurtă descriere 22](#_Toc100340276)

[3.2.2 Descrierea fiecărei componente în parte 25](#_Toc100340277)

[**3.2.2.1** **JPanel-ul WVOS\_Login** 25](#_Toc100340278)

[**3.2.2.2** **JPanel-ul SignupForm** 27](#_Toc100340279)

[3.2.3 Codul 30](#_Toc100340280)

[**3.2.3.1** **Librăriile necesare** 30](#_Toc100340281)

[**3.2.3.2** **Clasa SignupFrame** 30](#_Toc100340282)

[**3.2.3.3** **Variabilele definite de utilizator** 30](#_Toc100340283)

[**3.2.3.4** **Variabilele predefinite** 30](#_Toc100340284)

[**3.2.3.5** **Constructorul clasei SignupFrame:** public SignupFrame() 31](#_Toc100340285)

[**3.2.3.6** **Funcția pentru setarea stelelor / marker-elor de completare:** private void setStarsInvisible() 32](#_Toc100340286)

[**3.2.3.7** **Funcția care marchează câmpurile goale:** private boolean markEmptyFields() 32](#_Toc100340287)

[3.2.4 39](#_Toc100340288)

[4 Bibliografie 40](#_Toc100340289)

# **Introducere**

## Descrierea succintă a aplicației

**Visual OETPN Simulator (prescurtare: VOS)** este un utilitar open-source (oricine poate contribui cu modificări către acesta), care ajută studenții implicați în domenii tehnice să proiecteze și să simuleze modele OETPN (Object-Enhanced Timed Petri Nets). Pentru ca acest lucru să fie posibil, utilizatorul are nevoie de câteva cunoștințe în ceea ce privește Rețelele Petri în general, dar și noțiuni despre modelele OETPN. Pentru a afla mai multe despre Rețele Petri și modele OETPN, accesați <https://visual-oetpn-simulator.netlify.app/pages/about-oetpns.html> .

**Visual OETPN Simulator** a fost dezvoltat de Dragu Andreea, o studentă în anul IV de la Facultatea de Automatică și Calculatoare, secția Automatică și Informatică Aplicată, din cadrul Universității Tehnice din Cluj-Napoca. Această aplicație a fost dezvoltată cu scopul de a reprezenta lucrarea de licență, bazată pe cercetare asupra Rețelelor Petri, modelelor OETPN, dar și a tehnologiilor și limbajelor de programare necesare pentru realizarea acesteia (pentru partea de desktop: Java, Swing, MySQL; pentru partea de web: HTML, CSS, JavaScript).

Cele mai notabile caracteristici ale aplicației **Visual OETPN Simulator** includ:

* Ușurință în utilizare
* Diverse funcționalități, intuitive, ușor de înțeles
* Minimalism
* Gratuit (pentru ca utilitarul să poată fi folosit, utilizatorul trebuie să își creeze un cont în pagina de Sign Up a aplicației; dacă utilizatorul deja are un cont, tot ce trebuie să facă este să acceseze pagina de Log In pentru a avea acces la utilitar).

În acest document, se va descrie, pas cu pas, modul în care a fost realizată aplicația, de la tehnologiile și limbajele de programare folosite (Java, Swing, MySQL), până la componentele aplicației (frame-uri, controllere, bază de date etc.). De asemenea, se va descrie codul elaborat și utilizat în dezvoltarea acestei aplicații, pentru o mai bună înțelegere a funcționării aplicației.

## Obiective

* Familiarizarea cititorului cu modul în care a fost realizată aplicația Visual OETPN Simulator.
* Familiarizarea utilizatorului cu toate componentele create și utilizate, care alcătuiesc aplicația Visual OETPN Simulator.
* Detalierea tuturor claselor și a funcțiilor care alcătuiesc aplicația Visual OETPN Simulator.

# **Mediile de dezvoltare, tehnologiile și limbajele utilizate**

Visual OETPN Simulator reprezintă o sinergie de tehnologii și limbaje de programare pentru realizarea aplicațiilor desktop, utilizate cu ajutorul unor medii de dezvoltare, printre care se află următoarele:

* **NetBeans IDE:** permite dezvoltarea aplicațiilor dintr-un set de componente software modulare, numite module. Este un mediu de dezvoltare care permite dezvoltarea aplicațiilor cu ajutorul limbajelor de programare Java, PHP, C++, C, HTML5, JavaScript etc. Pentru aplicația Visual OETPN Simulator, s-a utilizat limbajul de programare **Java**, împreună cu tehnologia **Swing**, care oferă interfețe grafice utilizator (Graphical User Interface – GUI) pentru progamele Java.
* **MySQL:** este un sistem de gestiune a bazelor de date relaționale. Pentru aplicația Visual OETPN Simulator, MySQL este folosit pentru a ține evidența utilizatorilor înregistrați în baza de date, care indică totodată numărul de utilizatori care folosesc aplicația Visual OETPN Simulator.

Ca și tehnologii utilizate pentru realizarea acestei aplicații, s-au folosit:

* **Swing:** este o bibliotecă de dezvoltare pentru interfețele grafice utilizator (Graphical User Intefaces – GUIs), care dispune de diverse elemente pentru crearea acestor interfețe, cum ar fi JFrame, JButton, JTextField, JPanel șamd. Pachetul care conține majoritatea claselor pentru aplicațiile Swing este java.swing. În aplicația Visual OETPN Simulator, biblioteca Swing este utilizată pentru a crea formularul de înregistrare în aplicație (Sign Up), formularul de autentificare în aplicație (Log In) și fereastra de lucru / fereastra principală (Main Frame).

De asemenea, ca limbaje utilizate, avem următoarele:

* **Java:** este un limbaj de programare orientat-obiect, puternic tipizat, cu ajutorul căruia s-au dezvoltat cele mai multe aplicații, atât desktop, cât și mobile. Avantajul acestui limbaj de programare este că, un cod scris cu ajutorul acestuia poate rula pe orice platformă care instalată Mașina Virtuală Java (Java Virtual Machine – JVM). În ceea ce privește aplicația Visual OETPN Simulator, cea mai mare parte a acesteia este realizată cu ajutorul limbajului de programare Java, cu ajutorul căruia s-au definit clase, funcții, variabile etc.
* **SQL:** este un limbaj fourth-generation, ceea ce înseamnă că este un limbaj de scripting care nu necesită compilarea codului pentru ca acesta să ruleze. În schimb, ca orice limbaj fourth-generation, SQL are nevoie de un interpretor, care traduce codul în loc să îl compileze. Cu ajutorul limbajului SQL, în aplicația Visual OETPN Simulator, putem crea tabelele aferente aplicației noastre, vizualiza toți utilizatorii înregistrați, șterge un utilizator, modifica un utilizator șamd.

# **Realizarea aplicației**

## Sign Up Frame

### Scurtă descriere

Frame-ul de Sign Up este frame-ul care permite utilizatorului să își creeze un cont pentru a putea utiliza aplicația Visual OETPN Simulator. Acesta este prezentat în figura de mai jos.

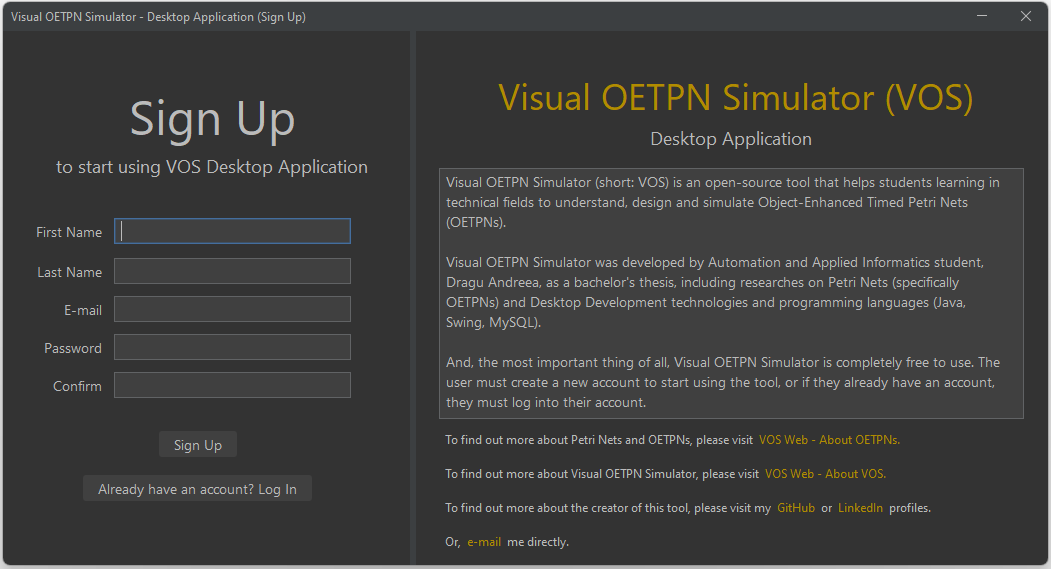


Figura 3.1.1.1. Frame-ul pentru înregistrare (Sign Up)

În partea stângă a frame-ului, se poate observa un formular de înregistrare, care permite utilizatorului să își creeze un cont folosindu-și câteva date personale. Toate datele introduse trebuie să fie corecte (adresa de e-mail să fie validă iar parolele să coincidă) pentru ca noul utilizator să fie înregistrat.

Pentru procesul de înregistrare în aplicație, vom avea următoarele scenarii:

* În cazul în care utilizatorul nu a completat toate câmpurile, va apărea următorul mesaj:

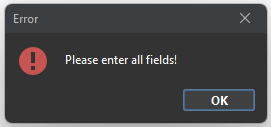


Figura 3.1.1.2. Casetă de dialog în cazul în care unul sau mai multe câmpuri au rămas necompletate

* În cazul în care utilizatorul introduce o adresă de e-mail deja în uz, va apărea următorul mesaj:

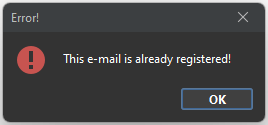


Figura 3.1.1.3. Casetă de dialog în cazul în care adresa de e-mail introdusă este deja în uz

* În cazul în care parolele introduse de utilizator nu coincid, va apărea următorul mesaj:

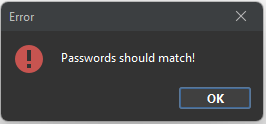


Figura 3.1.1.4. Casetă de dialog în cazul în care parolele nu coincid

* În cazul în care utilizatorul a introdus toate datele corect (adresa de e-mail nu este în uz și parolele coincid), va apărea următorul mesaj:

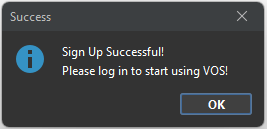


Figura 3.1.1.5. Casetă de dialog în cazul în care utilizatorul s-a înregistrat cu succes

În cazul în care utilizatorul este deja înregistrat pentru a utiliza aplicația, tot ce trebuie să facă este să acceseze butonul “Already have an account? Log In”. Prin apăsarea acestuia, se va deschide fereastra de autentificare “Log In”.

Prin accesarea butoanelor “Already have an account? Log In”, respectiv “Don’t have an account? Sign Up”, utilizatorul poate să se autentifice, respectiv înregistreze pentru a putea utiliza aplicația, după caz.

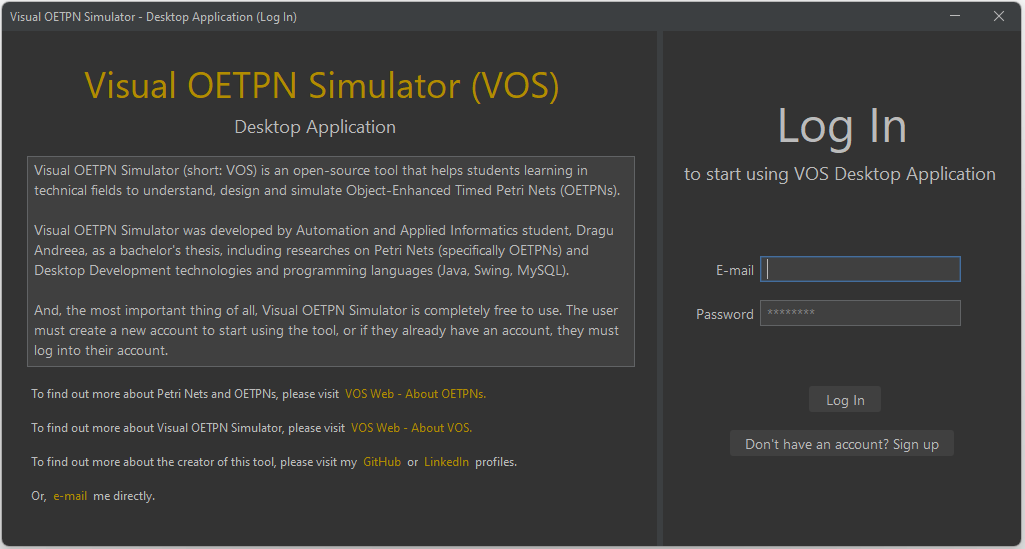


Figura 3.1.1.6. Frame-ul pentru autentificare (Log In)

În partea dreaptă a frame-ului, se pot observa următoarele:

* Un text care, odată ce a fost click-uit, redirecționează utilizatorul către pagina principală a site-ului Visual OETPN Simulator – Web Application: <https://visual-oetpn-simulator.netlify.app/index.html>

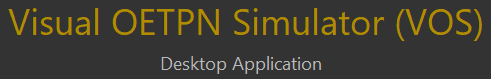


Figura 3.1.1.7. Textul (hyperlink-ul) către pagina principală a aplicației web

* Un text descriptiv, în care se prezintă, pe scurt, scopul aplicației, dezvoltatorul și caracteristicile acesteia.

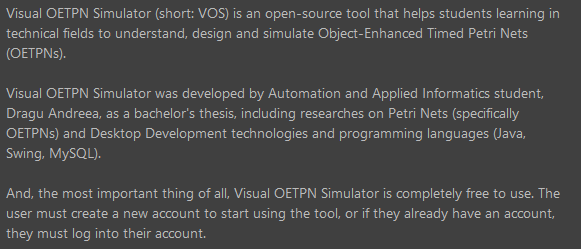


Figura 3.1.1.8. Caseta de text care descrie, pe scurt, aplicația

* Câteva link-uri către diferite resurse, pagini, profiluri, modalități de contact
* Redirecționare către pagina About OETPNs



Figura 3.1.1.9. Hyperlink către <https://visual-oetpn-simulator.netlify.app/pages/about-oetpns.html>

* Redirecționare către pagina About VOS



Figura 3.1.1.10. Hyperlink către <https://visual-oetpn-simulator.netlify.app/pages/about-vos.html>

* Redirecționare către profilul de GitHub / LinkedIn al dezvoltatorului



Figura 3.1.1.11. Hyperlink către <https://github.com/draguleee> și <https://www.linkedin.com/in/andreea-ioana-dragu-870a121a1/>

* Redirecționare către serviciul de e-mail



Figura 3.1.1.12. Redirecționare către e-mail

### Descrierea fiecărei componente în parte

Pentru a realiza o descriere a fiecărei componente a frame-ului de Sign Up, vom aborda fiecare JPanel, după cum urmează:

#### **JPanel-ul WVOS\_Signup**

1. JPanel-ul **WVOS\_Title**:

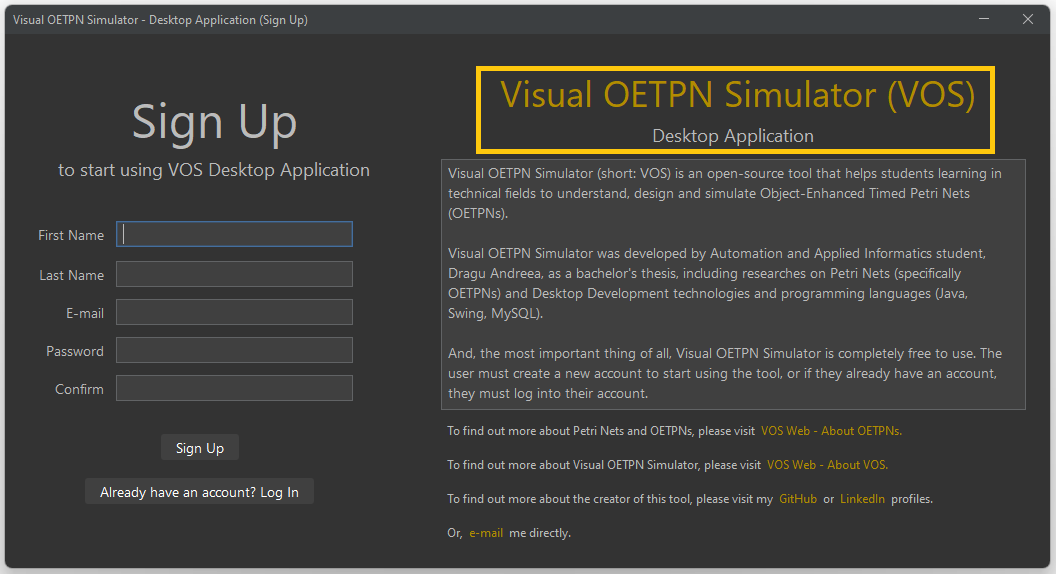


Figura 3.1.2.1.1. JPanel-ul WVOS\_Title, evidențiat cu chenarul portocaliu

* JLabel-ul **WVOS\_Welcome**: are ca text “Visual OETPN Simulator”, reprezintă titlul aplicației și are evenimentul mouseClicked(), care face ca acest JLabel să deschidă pagina Home din cadrul aplicației VOS Web.
* JLabel-ul **WVOS\_DesktopApp**: are ca text “Desktop Application”, reprezintă descrierea tipului de aplicație (desktop) și nu are niciun eveniment.

1. JscrollPane-ul **WVOS\_Text**:

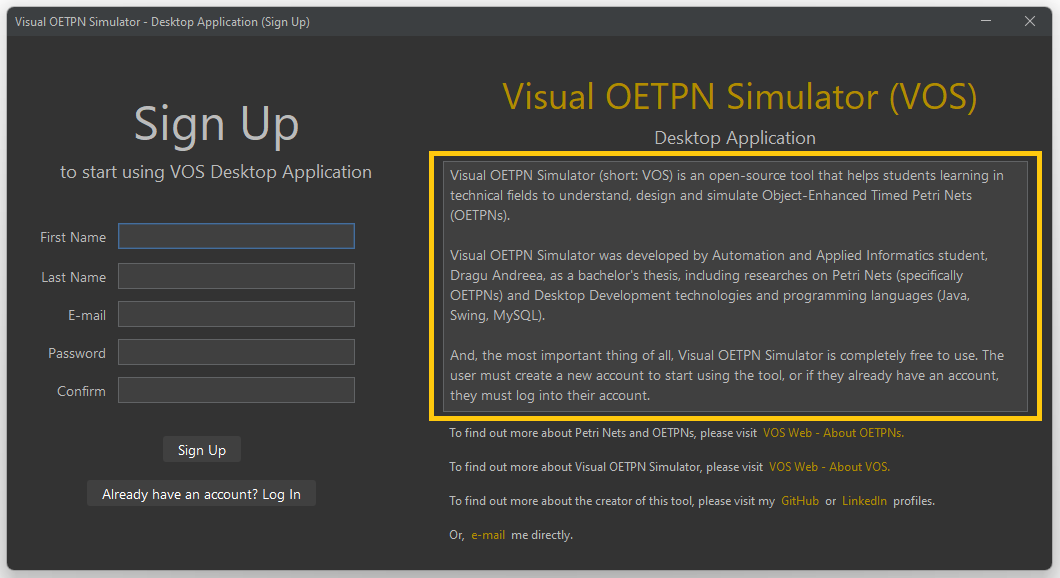


Figura 3.1.2.1.2. JscrollPane-ul WVOS\_Text, evidențiat cu portocaliu

* JTextArea **WelcomeText**: conține o descriere scurtă a aplicației Visual OETPN Simulator, a dezvoltatorului și a modului de înregistrare / autentificare în aplicație și nu are niciun eveniment.

1. JPanel-ul **WVOS\_Links**:

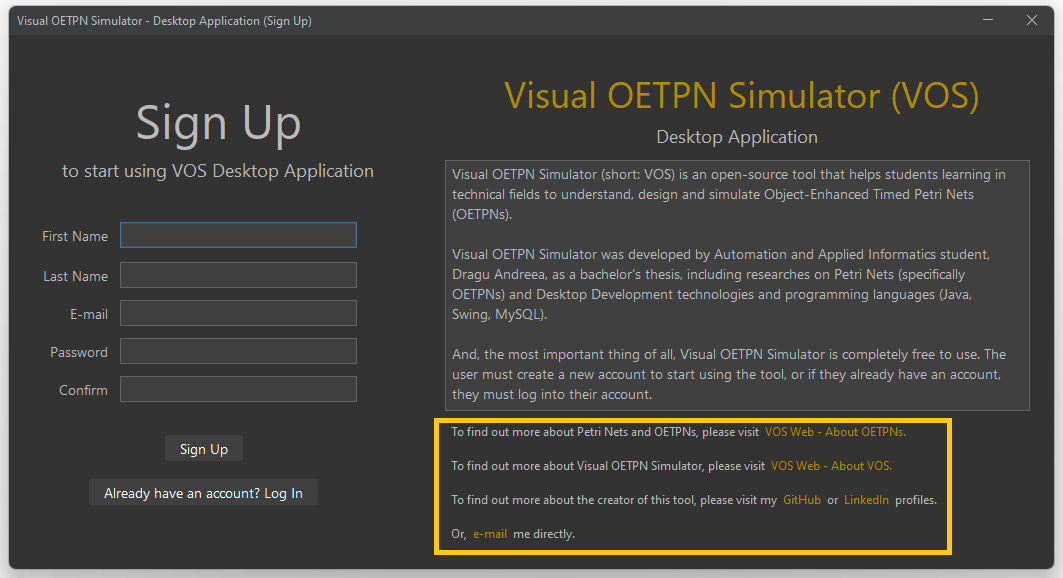


Figura 3.1.2.1.3. JPanel-ul WVOS\_Links, evidențiat cu portocaliu

* JLabel-ul **TextAboutOETPNs**: conține textul “To find out more about Petri Nets and OETPNs, please visit”, informează utilizatorul despre existența unor documentații ce privesc Rețelele Petri și modelele OETPN, și nu are nciun eveniment.
* JLabel-ul **AboutOETPN**: conține textul “VOS Web – About OETPNs”, redirecționează utilizatorul către pagina About OETPNs din cadrul aplicației VOS Web odată ce JLabel-ul este click-uit. Acest JLabel are ca eveniment mouseClicked(), iar funcția setAboutOETPNHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul**.
* JLabel-ul **TextAboutVOS**: conține textul “To find out more about Visual OETPN Simulator, please visit”, informează utilizatorul despre existența unei pagini din cadrul aplicației VOS Web, care dispune de informații legate de aplicația Visual OETPN Simulator, și nu are niciun eveniment.
* JLabel-ul **AboutVOS**: conține textul “VOS Web – About VOS”, redirecționează utilizatorul către pagina About VOS din cadrul aplicației VOS Web odată ce JLabel-ul este click-uit. Acest JLabel are ca eveniment mouseClicked(), iar funcția setAboutVOSHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Coduri**.
* JLabel-urile **TextAboutCreator1, TextAboutCreator2** și **TextAboutCreator3**: conțin textul “To find out more about the creator of this tool, please visit my … or … profiles”, informează utilizatorul despre existența unor profiluri ale dezvoltatorului aplicației, care pot fi folosite pentru a lua legătura cu acesta, și nu au niciun eveniment.
* JLabel-urile **GitHub** și **LinkedIn**: conțin textul “GitHub”, respectiv “LinkedIn”, redirecționează utilizatorul către profilul de GitHub, respectiv LinkedIn al dezvoltatorului aplicației, odată ce acestea au fost click-uite. Aceste două JLabel-uri au evenimentul mouseClicked(), iar funcțiile setGithubProfileHyperlink(), respectiv setLinkedinProfileHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**
* JLabel-urile **TextEmail1** și **TextEmail2**: conțin textul “Or, … directly”, informează utilizatorul despre existența posibilității de a lua legătura cu dezvoltatorul aplicației prin e-mail, și nu au niciun eveniment.
* JLabel-ul **Email**: conține textul “e-mail me”, redirecționează utilizatorul către aplicație de mesagerie electronică, unde poate lua legătura cu dezvoltatorul aplicației. Ca evenimente, JLabel-ul Email are evenimentul mouseClicked(), iar funcția setEmailOption() implementează acest eveniment.

#### **JPanel-ul SignupForm**

1. JPanel-ul **SFText**:

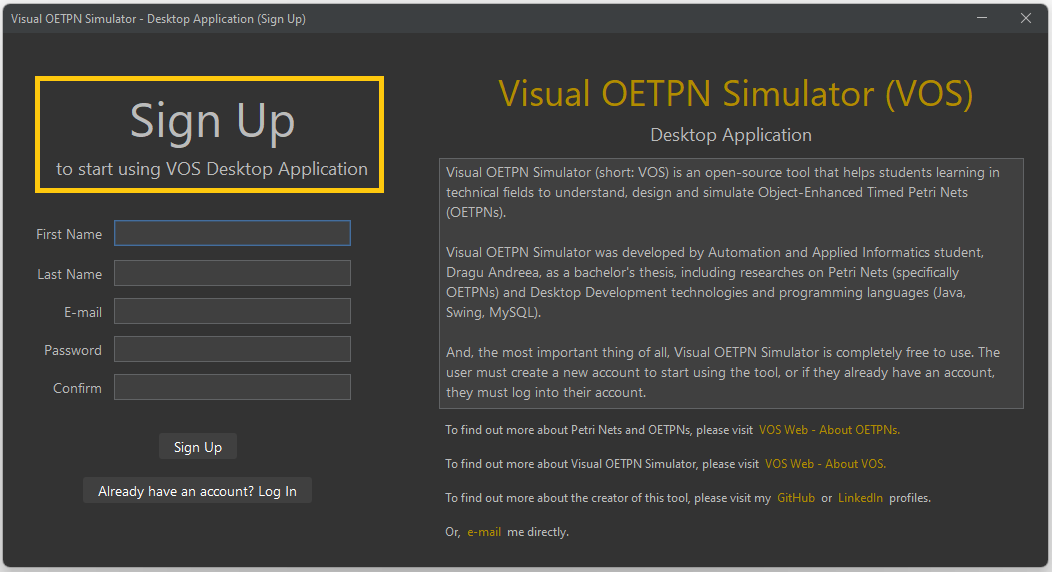


Figura 3.1.2.2.1. JPanel-ul SFText, evidențiat cu portocaliu

* JLabel-ul **SFMessage1:** conține textul “Sign Up”, informează utilizatorul că se află în formularul de înregistrare, și nu are niciun eveniment.
* JLabel-ul **SFMessage2**: conține textul “to start using VOS Desktop Application”, informează utilizatorul că are nevoie de crearea unui cont pentru a avea acces la aplicație, și nu are niciun eveniment.

1. JPanel-ul **SFUserData**:

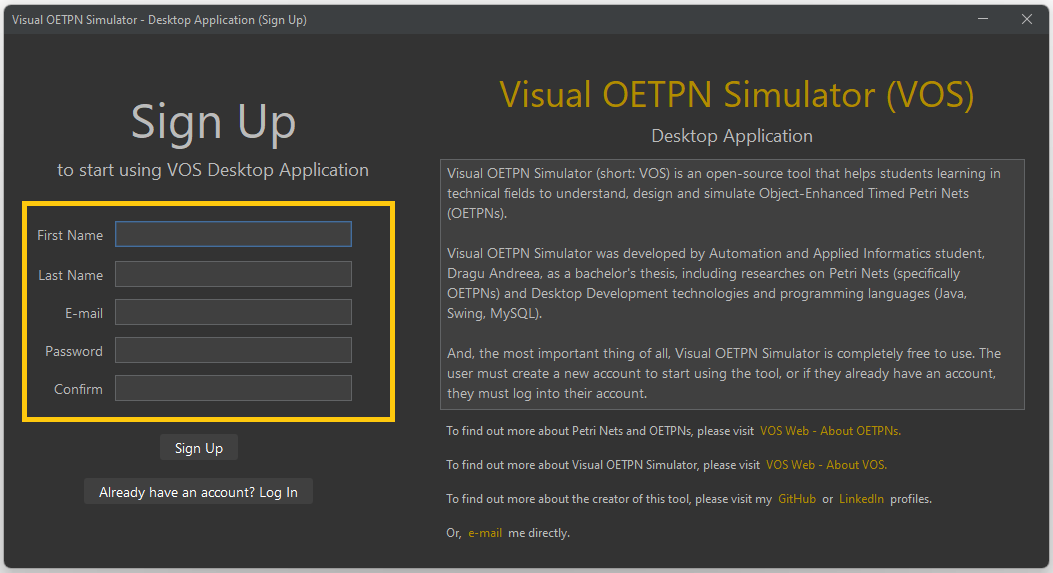


Figura 3.1.2.2.2.JPanel-ul SFUserData, evidențiat cu portocaliu

* JLabel-urile **SFFNameLabel, SFLNameLabel, SFEmailLabel, SFPassLabel, SFRepPassLabel**: conțin textele “First Name”, “Last Name”, “E-mail”, “Password”, “Confirm”, reprezintă label-uri pentru prenume, nume, e-mail, parolă și confirmarea parolei și nu au niciun eveniment.
* JTextField-urile **SFFNameField, SFLNameField, SFEmailField, SFPassField, SFRepPassField**: conțin credențialele date de utilizator, reprezintă câmpuri de completare pentru prenume, nume, e-mail, parolă și completarea parolei și nu au niciun eveniment.
* JLabel-urile **SFFNameStar, SFLNameStar, SFEmailStar, SFPassStar, SFRepPassStar**: conțin textul “\*”, reprezintă stelele / marker-ele pentru câmpurile de prenume, nume, e-mail, parolă și confirmarea parolei (sunt vizibile doar atunci când casetele de text rămân necompletate) și nu au niciun eveniment.

1. JPanel-ul **SFButtons**:

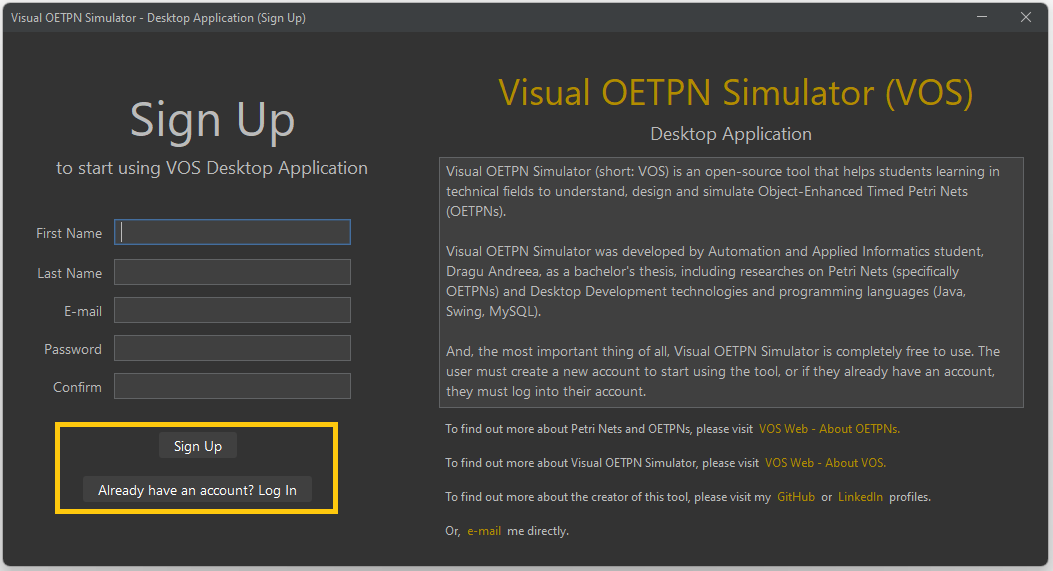


Figura 3.1.2.2.3. JPanel-ul SFButtons, evidențiat cu portocaliu

* JButton-ul **SignupButton**: conține textul “Sign Up”, este butonul care, odată apăsat, înregistrează utilizatorul în baza de date, în funcție de următoarele scenarii:
  + Dacă toate datele au fost introduse corect: se afișează mesajul “Sign up successful! Please log in to start using VOS!”;
  + Dacă unul sau mai multe câmpuri au rămas necompletate: se afișează mesajul “Please enter all fields!”;
  + Dacă adresa de e-mail este deja înregistrată: se afișează mesajul “This e-mail is already registered!”.

JButton-ul SignupButton are evenimentul actionPerformed(), iar funcția care implementează acest eveniment este SignupButtonActionPerformed(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**

* JButton-ul **HaveAccButton**: conține textul “Already have an account? Log In”, este un buton de redirecționare către pagina de autentificare (Log In), în cazul în care utilizatorul este deja înregistrat. Ca și evenimente, JButton-ul HaveAccButton are evenimentul actionPerformed(), iar funcția care implementează acest eveniment este HaveAccButtonPerformed(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**

### Codul

#### **Librăriile necesare**

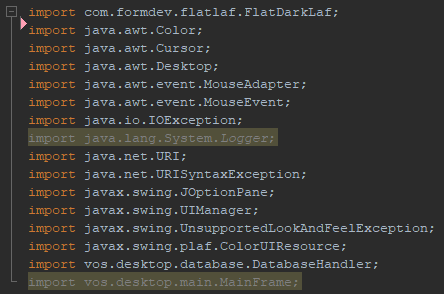
****

Figura 3.1.3.1.1. Librăriile necesare pentru frame-ul de Sign Up

#### **Clasa SignupFrame**

* Extinde clasa JFrame;
* Conține toate funcțiile și variabilele necesare pentru a asigura funcționalitatea corectă și completă a frame-ului de înregistrare a utilizatorului.

#### **Variabilele definite de utilizator**

* **private DatabaseHandler handler = null:** este handler-ul care realizează conexiunea dintre aplicație și baza de date;
* **private UIManager uim = new UIManager():** este componenta care se ocupă cu stilizarea, în cazul aplicației Visual OETPN Studio, a JoptionPane-urilor care pot apărea, în funcție de situație.

#### **Variabilele predefinite**

* Reprezintă variabilele care nu pot fi modificate (NetBeans IDE le declară automat în cod)
* Reprezintă toate componentele utilizate pentru crearea frame-ului de Sign Up (JLabel-uri, JButton-uri, JTextField-uri, JtextArea, JPanel-uri, JpasswordField-uri șamd.).



Figura 3.1.3.1.2. Variabilele predefinite (nu pot fi modificate)

#### **Constructorul clasei SignupFrame:** public SignupFrame()

* Inițializează variabila handler, preluând instanța curentă (dacă există) / creând o nouă instanță (dacă nu există);
* Setează titlul ferestrei;
* Elimină focus-ul componentei JtextArea din JPanel-ul WVOS\_Signup;
* Setează hyperlink-urile (apelează funcțiile aferente);
* Setează stelele / marker-ele de completare;
* Setează poziționarea frame-ului în mijlocul ecranului;
* Setează frame-ul ca fiind vizibil;
* Setează toate componentele;

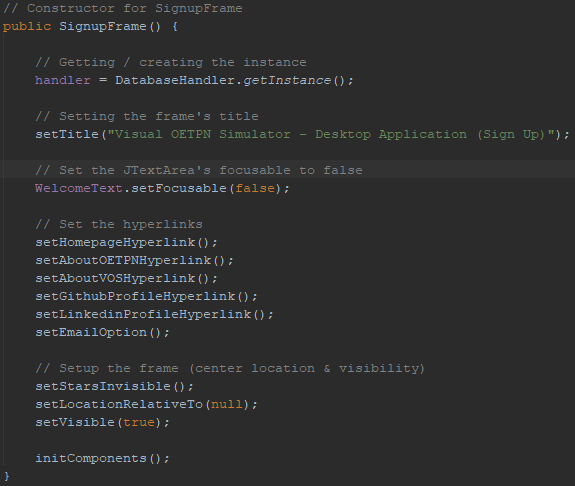


Figura 3.1.3.1.3. Constructorul clasei SignupFrame

#### **Funcția pentru setarea stelelor / marker-elor de completare:** private void setStarsInvisible()

* Setează toate stelele / marker-ele de completare ca fiind invizibile cât timp câmpurile sunt completate;
* Stelele / marker-ele devin vizibile atunci când unul sau mai multe câmpuri rămân necompletate.

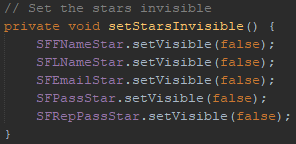


Figura 3.1.3.1.4. Funcția pentru setarea stelelor / marker-elor de completare

#### **Funcția care marchează câmpurile goale:** private boolean markEmptyFields()

* Marchează un câmp dacă acesta este sau nu gol;
* Se creează, în interiorul funcției, o variabilă isEMpty de tip boolean, care este inițializată cu valoarea false;
* Pentru fiecare câmp, dacă acesta este gol, isEmpty ia valoarea true, și steaua / marker-ul corespunzător devine vizibil;
* Funcția returnează variabila isEmpty;



Figura 3.1.3.1.5. Funcția pentru marcarea câmpurilor goale

#### **Funcția pentru verificarea parolelor (dacă acestea coincid):** private boolean matchingPasswords()

* Verifică dacă parola din câmpul “Password” și cea din câmpul “Confirm” coincide;
* Funcția returnează o valoare de tip boolean, care reprezintă valoarea de adevăr rezultată în urma comparării textului din câmpul “Password” și cel din câmpul “Confirm”.

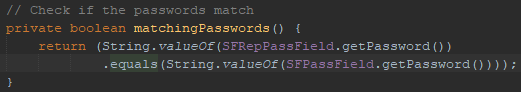


Figura 3.1.3.1.6. Funcția pentru verificarea parolelor (dacă acestea coincid)

#### **Funcția pentru inițializarea tuturor componentelor din SignupFrame: private** void initComponents()

* Funcție generată automat;
* Reprezintă funcția care inițializează, configurează și setează fiecare componentă din JFrame-ul SignupFrame (JLabel-uri, JButton-uri, JPanel-uri, JTextField-uri).

#### **Eveniment generat în momentul în care se apasă butonul Sign Up:** private void SignupButtonActionPerformed(java.awt.event.ActionEvent evt)

* Reprezintă funcția care setează un eveniment care trebuie să se întâmple în momentul în care butonul Sign Up este apăsat;
* Evenimentul este actionPerformed, care este implementat de funcția SignupButtonActionPerformed;
* Scenariile posibile, după apăsarea butonului, sunt:
* Unul sau mai multe câmpuri au rămas necompletate: “Please enter all fields!”;
* Parolele nu coincid: “Passwords should match!”;
* Adresa de e-mail este deja înregistrată: “This e-mail is already registered!”;
* Toate datele au fost introduse corect: “Sign Up Successful! Please log in to start using VOS!”.

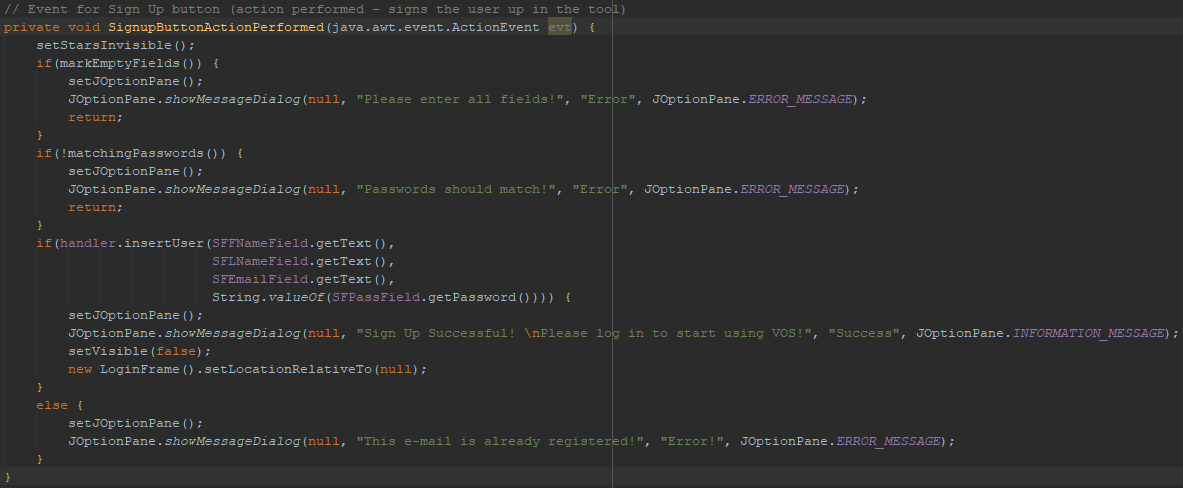
****

Figura 3.1.3.1.7. Funcția SignupButtonActionPerformed, care implementează evenimentul actionPerformed() (vor avea loc evenimente în funcție de cazurile menționate mai sus)

#### **Eveniment generat în momentul în care se apasă butonul HaveAccButton:** private void HaveAccButtonActionPerformed(java.awt.event.ActionEvent evt)

* Reprezintă funcția care setează un eveniment care trebuie să se întâmple în momentul în care butonul Have Account este apăsat;
* Evenimentul este actionPerformed, care este implementat de funcția HaveAccButtonActionPerformed;
* După apăsarea acestui buton, utilizatorul va fi redirecționat către pagina de autentificare: se va deschide fereastra de autentificare în mijlocul ecranului.

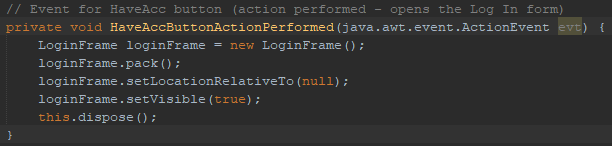


Figura 3.1.3.1.8. Funcția HaveAccButtonActionPerformed, care implementează evenimentul actionPerformed() (se va deschide fereastra de autentificare – Log In)

#### **Funcția pentru setarea elementului JOptionPane:** private void setJOptionPane()

* Reprezintă funcția care setează anumiți parametri pentru ferestrele de dialog;
* Parametrii care sunt configurați sunt:
* Fundalul ferestrei de dialog;
* Fundalul panel-ului;
* Culoarea textului;
* Culoarea de fundal a butonului;
* Culoarea textului butonului;

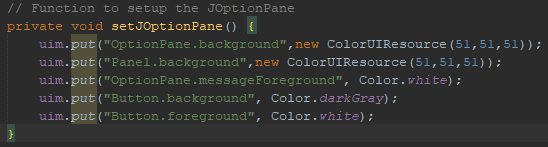


Figura 3.1.3.1.9. Funcția care setează ferestrele de dialog

#### **Funcția pentru setarea hyperlink-ului către pagina Home (VOS Web**): private void setHomepageHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina Home din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setHomepageHyperlink() se poate vedea în figura de mai jos.

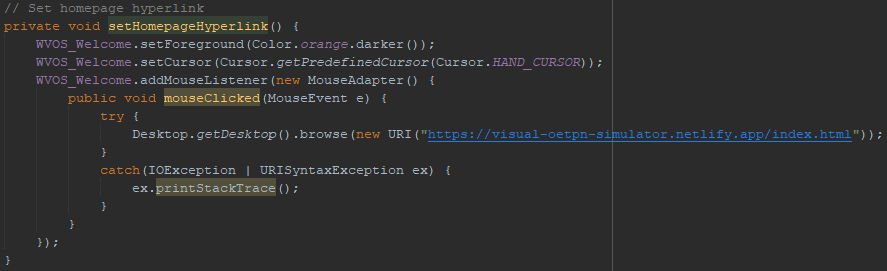


Figura 3.1.3.1.10. Funcția pentru setarea hyperlink-ului către pagina Home

#### **Funcția pentru setarea hyperlink-ului către pagina About OETPNs (VOS Web):** private void setAboutOETPNHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina About OETPNs din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setAboutOETPNHyperlink() se poate vedea în figura de mai jos.

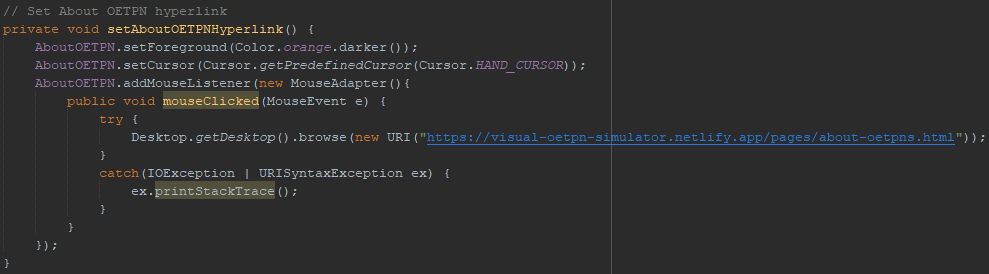


Figura 3.1.3.1.11. Funcția pentru setarea hyperlink-ului către pagina About OETPNs

#### **Funcția pentru setarea hyperlink-ului către pagina About VOS (VOS Web):** private void setAboutVOSHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina About VOS din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setAboutVOSHyperlink() se poate vedea în figura de mai jos.

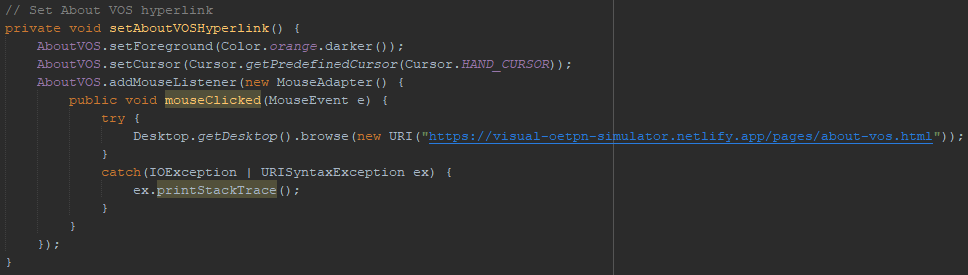


Figura 3.1.3.1.12. Funcția pentru setarea hyperlink-ului către pagina About VOS

#### **Funcția pentru setarea hyperlink-ului către profilul personal de GitHub al dezvoltatorului:** private void setGithubProfileHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide profilul personal de GitHub al dezvoltatorului aplicației
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setGithubProfileHyperlink() se poate vedea în figura de mai jos.

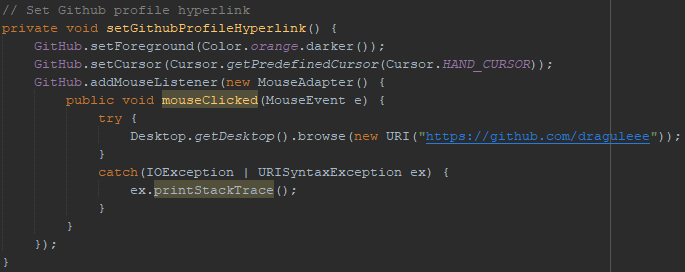


Figura 3.1.3.1.13. Funcția pentru setarea hyperlink-ului către profilul personal de GitHub al creatorului aplicației

#### **Funcția pentru setarea hyperlink-ului către profilul personal de LinkedIn al dezvoltatorului:** private void setLinkedinProfileHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide profilul personal de LinkedIn al dezvoltatorului aplicației
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setLinkedinProfileHyperlink() se poate vedea în figura de mai jos.

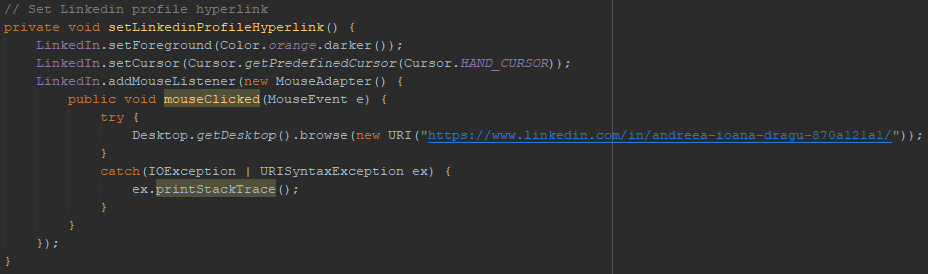


Figura 3.1.3.1.14. Funcția pentru setarea hyperlink-ului către profilul personal de Linkedin al creatorului aplicației

#### **Funcția pentru setarea opțiunii de trimitere a unui e-mail către creatorul aplicației:** private void setEmailOption()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide mesageria electronică (utilizatorul poate lua legătura cu dezvoltatorul aplicației direct prin e-mail)
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setEmailOption() se poate vedea în figura de mai jos.

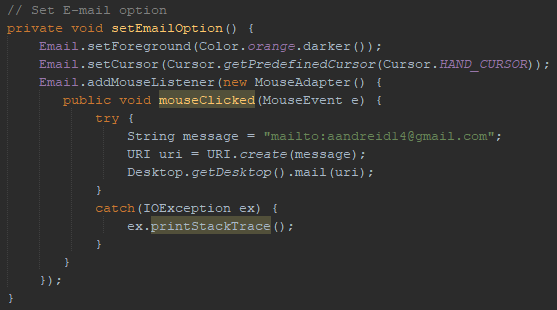


Figura 3.1.3.1.15. Funcția pentru setarea opțiunii de trimitere a unui e-mail către creatorul aplicației

#### **Metoda main:** public static void main(String[] args)

* Reprezintă punctul de intrare a programului;
* Fără această metodă, Mașina Virtuală Java nu va executa programul;
* Pentru clasa SignupFrame, în metoda main, vom avea:
* Metoda invokeLater(...): asigură apelul în siguranță al metodelor în clasele Swing;
* Parametrii metodei invokeLater: interfața Runnable, care trebuie să ofere o implementare pentru metoda run();
* Metoda run(): vom avea un bloc try-catch:
* Blocul try: se setează Look And Feel-ul FlatDarkLaf (link: <https://www.formdev.com/flatlaf/>);
* Blocul catch: se prinde excepția aruncată.
* Se crează un nou obiect de clasă SignupFrame (un nou frame de înregistrare) și i se setează Look And Feel-ul importat.

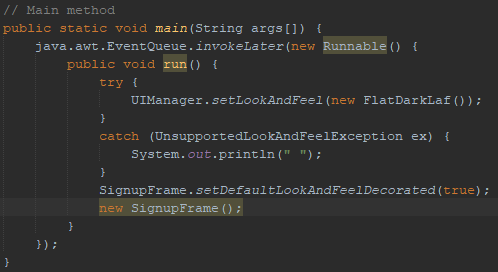


Figura 3.1.3.1.16. Funcția / metoda main

## Login Frame

### Scurtă descriere

Frame-ul de Log In este frame-ul care permite utilizatorului să se autentifice cu crdențialele deja create, pentru a putea utiliza aplicația Visual OETPN Simulator. Acesta este prezentat în figura de mai jos.

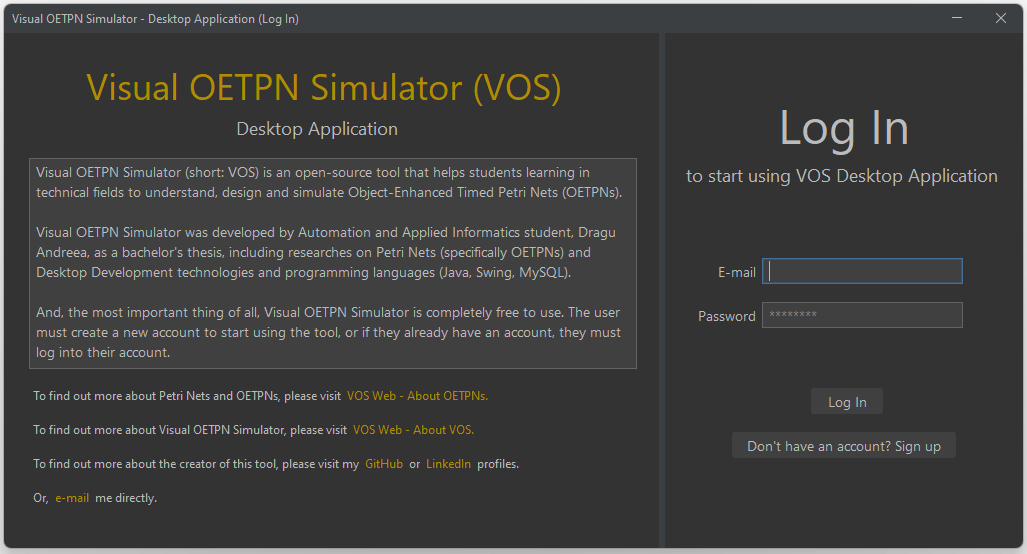


Figura 3.2.1.1. Frame-ul pentru autentificare (Log In)

În partea dreaptă a frame-ului, se poate observa un formular de autentificare, care permite utilizatorului să se logheze în contul existent pentru a putea folosi utilitarul. Toate datele introduse trebuie să fie corecte (adresa de e-mail și parola să fie asociate unui cont) pentru ca procesul de autentificare să fie realizat cu succes.

Pentru procesul de autentificare în aplicație, vom avea următoarele scenarii:

* În cazul în care adresa de e-mail sau parola sunt greșite, va apărea următorul mesaj:

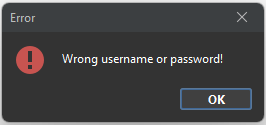


Figura 3.2.1.2. Casetă de dialog în cazul în care adresa de e-mail sau parola au fost introduse greșit

* În cazul în care utilizatorul a lăsat completat unul sau ambele casete de text:

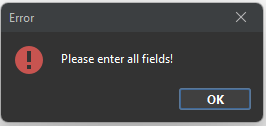


Figura 3.2.1.3. Casetă de dialog în cazul în care adresa de e-mail introdusă este deja în uz

* În cazul în care utilizatorul a introdus toate datele corect (adresa de e-mail și parola sunt asociate unui cont), va apărea următorul mesaj:

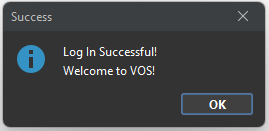


Figura 3.2.1.4. Casetă de dialog în cazul în care utilizatorul s-a înregistrat cu succes

După autentificarea utilizatorului, dacă se apasă butonul “OK” din fereastra de Success, se va deschide fereastra principală / de lucru (MainFrame).

În cazul în care utilizatorul nu este înregistrat pentru a utiliza aplicația, tot ce trebuie să facă este să acceseze butonul “Don’t have an account? Sign Up”. Prin apăsarea acestuia, se va deschide fereastra de înregistrare “Sign Up”.

Prin accesarea butoanelor “Already have an account? Log In”, respectiv “Don’t have an account? Sign Up”, utilizatorul poate să se autentifice, respectiv înregistreze pentru a putea utiliza aplicația, după caz.

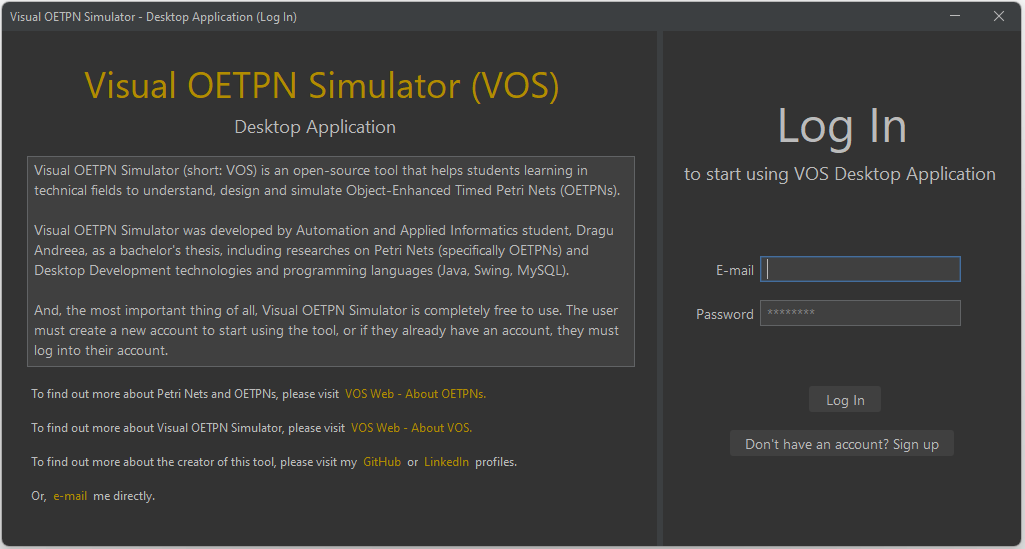


Figura 3.2.1.6. Frame-ul pentru autentificare (Log In)

În partea dreaptă a frame-ului, se pot observa următoarele:

* Un text care, odată ce a fost click-uit, redirecționează utilizatorul către pagina principală a site-ului Visual OETPN Simulator – Web Application: <https://visual-oetpn-simulator.netlify.app/index.html>

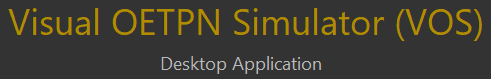


Figura 3.2.1.7. Textul (hyperlink-ul) către pagina principală a aplicației web

* Un text descriptiv, în care se prezintă, pe scurt, scopul aplicației, dezvoltatorul și caracteristicile acesteia.

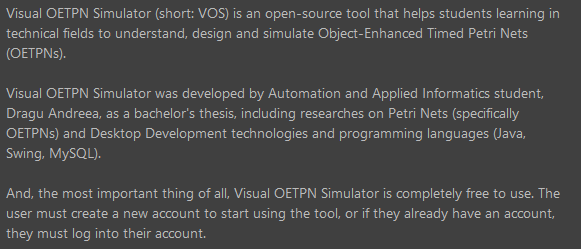


Figura 3.2.1.8. Caseta de text care descrie, pe scurt, aplicația

* Câteva link-uri către diferite resurse, pagini, profiluri, modalități de contact
* Redirecționare către pagina About OETPNs



Figura 3.2.1.9. Hyperlink către <https://visual-oetpn-simulator.netlify.app/pages/about-oetpns.html>

* Redirecționare către pagina About VOS



Figura 3.2.1.10. Hyperlink către <https://visual-oetpn-simulator.netlify.app/pages/about-vos.html>

* Redirecționare către profilul de GitHub / LinkedIn al dezvoltatorului



Figura 3.2.1.11. Hyperlink către <https://github.com/draguleee> și <https://www.linkedin.com/in/andreea-ioana-dragu-870a121a1/>

* Redirecționare către serviciul de e-mail



Figura 3.2.1.12. Redirecționare către e-mail

### Descrierea fiecărei componente în parte

Pentru a realiza o descriere a fiecărei componente a frame-ului de Log In, vom aborda fiecare JPanel, după cum urmează:

#### **JPanel-ul WVOS\_Login**

1. JPanel-ul **WVOS\_Title**:



Figura 3.2.2.1.1. JPanel-ul WVOS\_Title, evidențiat cu chenarul portocaliu

* JLabel-ul **WVOS\_Welcome**: are ca text “Visual OETPN Simulator”, reprezintă titlul aplicației și are evenimentul mouseClicked(), care face ca acest JLabel să deschidă pagina Home din cadrul aplicației VOS Web.
* JLabel-ul **WVOS\_DesktopApp**: are ca text “Desktop Application”, reprezintă descrierea tipului de aplicație (desktop) și nu are niciun eveniment.

1. JscrollPane-ul **WVOS\_Text**:

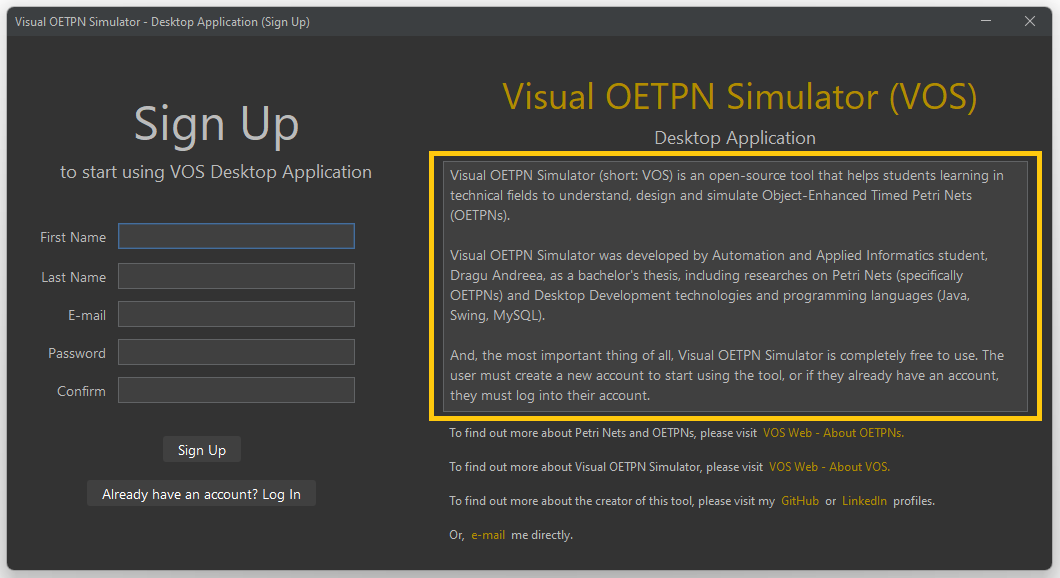


Figura 3.2.2.1.2. JscrollPane-ul WVOS\_Text, evidențiat cu portocaliu

* JTextArea **WelcomeText**: conține o descriere scurtă a aplicației Visual OETPN Simulator, a dezvoltatorului și a modului de înregistrare / autentificare în aplicație și nu are niciun eveniment.

1. JPanel-ul **WVOS\_Links**:

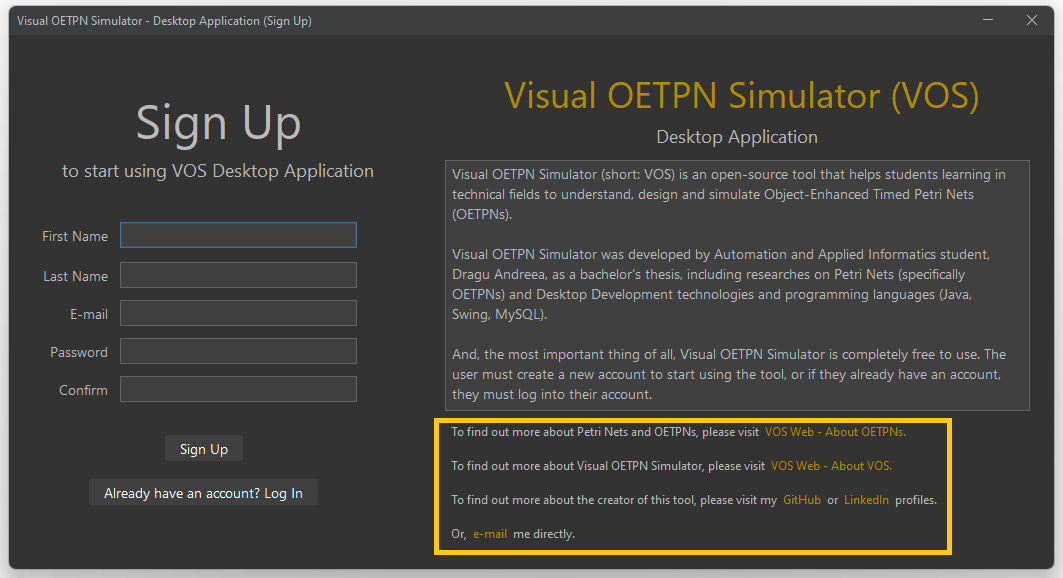


Figura 3.2.2.1.3. JPanel-ul WVOS\_Links, evidențiat cu portocaliu

* JLabel-ul **TextAboutOETPNs**: conține textul “To find out more about Petri Nets and OETPNs, please visit”, informează utilizatorul despre existența unor documentații ce privesc Rețelele Petri și modelele OETPN, și nu are nciun eveniment.
* JLabel-ul **AboutOETPN**: conține textul “VOS Web – About OETPNs”, redirecționează utilizatorul către pagina About OETPNs din cadrul aplicației VOS Web odată ce JLabel-ul este click-uit. Acest JLabel are ca eveniment mouseClicked(), iar funcția setAboutOETPNHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul**.
* JLabel-ul **TextAboutVOS**: conține textul “To find out more about Visual OETPN Simulator, please visit”, informează utilizatorul despre existența unei pagini din cadrul aplicației VOS Web, care dispune de informații legate de aplicația Visual OETPN Simulator, și nu are niciun eveniment.
* JLabel-ul **AboutVOS**: conține textul “VOS Web – About VOS”, redirecționează utilizatorul către pagina About VOS din cadrul aplicației VOS Web odată ce JLabel-ul este click-uit. Acest JLabel are ca eveniment mouseClicked(), iar funcția setAboutVOSHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Coduri**.
* JLabel-urile **TextAboutCreator1, TextAboutCreator2** și **TextAboutCreator3**: conțin textul “To find out more about the creator of this tool, please visit my … or … profiles”, informează utilizatorul despre existența unor profiluri ale dezvoltatorului aplicației, care pot fi folosite pentru a lua legătura cu acesta, și nu au niciun eveniment.
* JLabel-urile **GitHub** și **LinkedIn**: conțin textul “GitHub”, respectiv “LinkedIn”, redirecționează utilizatorul către profilul de GitHub, respectiv LinkedIn al dezvoltatorului aplicației, odată ce acestea au fost click-uite. Aceste două JLabel-uri au evenimentul mouseClicked(), iar funcțiile setGithubProfileHyperlink(), respectiv setLinkedinProfileHyperlink() implementează evenimentul mouseClicked(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**
* JLabel-urile **TextEmail1** și **TextEmail2**: conțin textul “Or, … directly”, informează utilizatorul despre existența posibilității de a lua legătura cu dezvoltatorul aplicației prin e-mail, și nu au niciun eveniment.
* JLabel-ul **Email**: conține textul “e-mail me”, redirecționează utilizatorul către aplicație de mesagerie electronică, unde poate lua legătura cu dezvoltatorul aplicației. Ca evenimente, JLabel-ul Email are evenimentul mouseClicked(), iar funcția setEmailOption() implementează acest eveniment.

#### **JPanel-ul LoginForm**

1. JPanel-ul **LFText**:

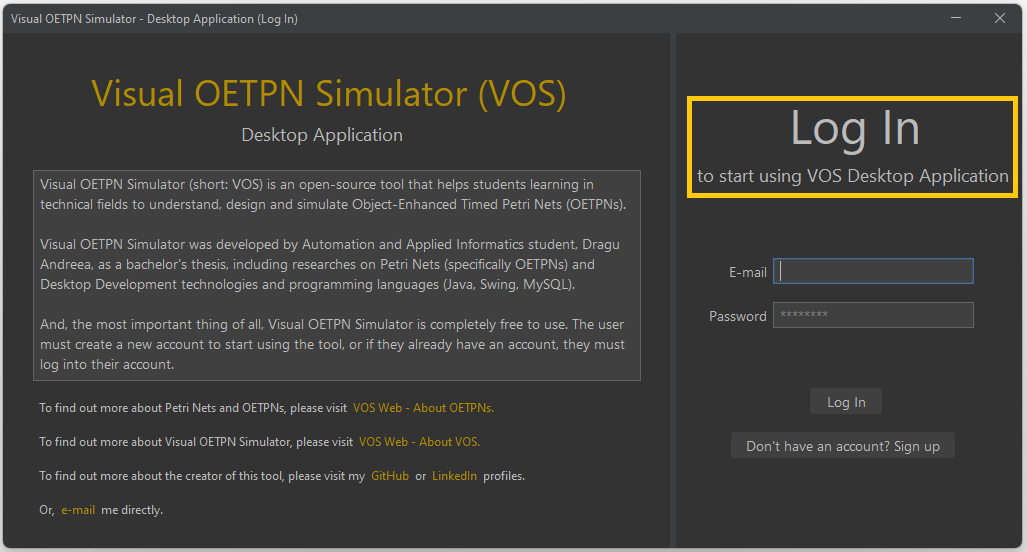


Figura 3.2.2.2.1. JPanel-ul LFText, evidențiat cu portocaliu

* JLabel-ul **LFMessage1:** conține textul “Log In”, informează utilizatorul că se află în formularul de înregistrare, și nu are niciun eveniment.
* JLabel-ul **LFMessage2**: conține textul “to start using VOS Desktop Application”, informează utilizatorul că are nevoie de crearea unui cont pentru a avea acces la aplicație, și nu are niciun eveniment.

1. JPanel-ul **LFUserData**:

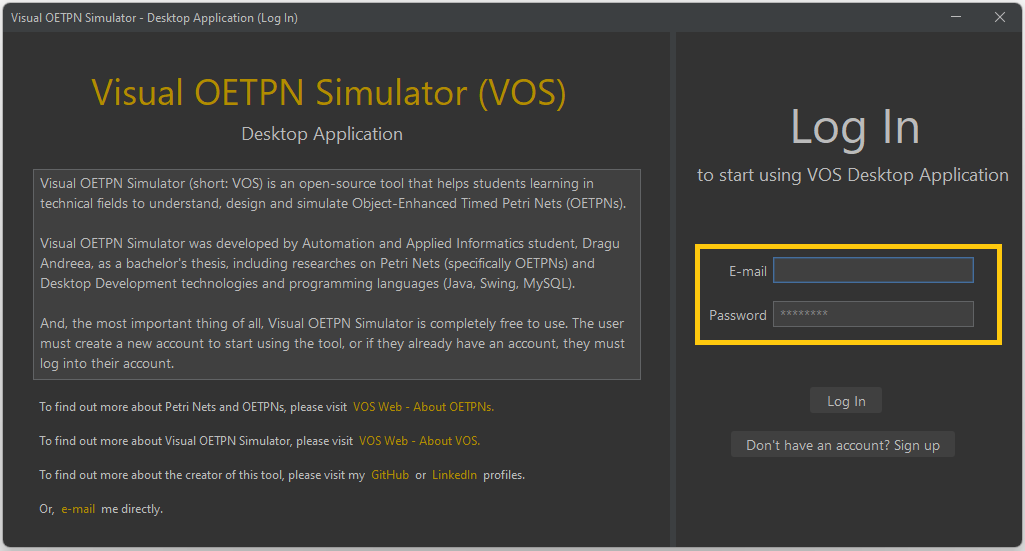


Figura 3.2.2.2.2. JPanel-ul SFUserData, evidențiat cu portocaliu

* JLabel-urile **LFEmailLabel, LFPassLabel**: conțin textele “E-mail” și “Password”, reprezintă label-uri pentru e-mail și parolă și nu au niciun eveniment.
* JTextField-urile **LFEmailField, LFPassField**: conțin credențialele utilizatorului, introduse de la tastatură (e-mail-ul și parola care corespund unui cont înregistrat în baza de date)
* JLabel-urile **LFEmailStar, LFPassStar**: conțin textul “\*”, reprezintă stelele / marker-ele pentru câmpurile de e-mail și parolă (sunt vizibile doar atunci când casetele de text rămân necompletate) și nu au niciun eveniment.

1. JPanel-ul **LFButtons**:

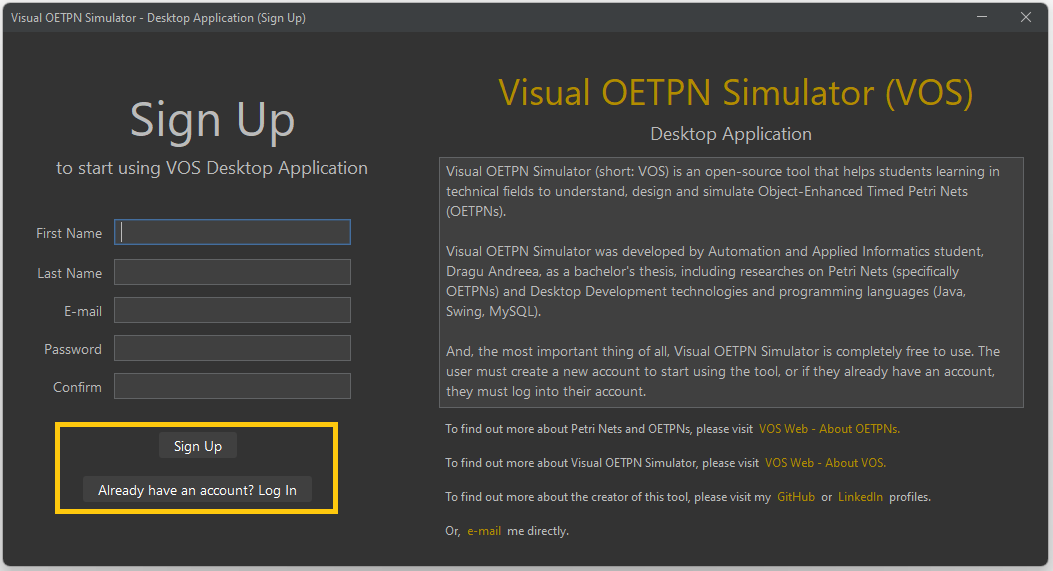


Figura 3.1.2.6. JPanel-ul SFButtons, evidențiat cu portocaliu

* JButton-ul **LoginButton**: conține textul “Log In”, este butonul care, odată apăsat, autentifică utilizatorul pentru a putea folosi utilitarul, în funcție de următoarele scenario:
  + Dacă toate datele au fost introduse corect: se afișează mesajul “Log In Successful! Welcome to VOS!”
  + Dacă unul sau mai multe câmpuri au rămas necompletate: se afișează mesajul “Please enter all fields!”;
  + Dacă e-mail-ul sau parola au fost introduse greșit: se afișează mesajul “Wrong username or password!”

JButton-ul LoginButton are evenimentul actionPerformed(), iar funcția care implementează acest eveniment este LoginButtonActionPerformed(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**

* JButton-ul **NoAccButton**: conține textul “Don’t have an account? Sign Up”, este un buton de redirecționare către pagina de înregistrare (Sign Up), în cazul în care utilizatorul nu are un cont. Ca și evenimente, JButton-ul NoAccButton are evenimentul actionPerformed(), iar funcția care implementează acest eveniment este NoAccButtonPerformed(). Mai multe despre aceste evenimente și funcții se vor discuta în secțiunea **Codul.**

### Codul

#### **Librăriile necesare**

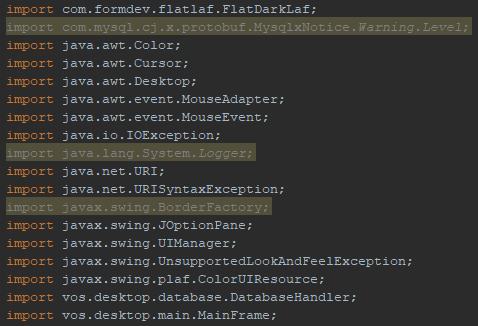


Figura 3.2.3.1.1. Librăriile necesare pentru frame-ul de Log In

#### **Clasa LoginFrame**

* Extinde clasa JFrame;
* Conține toate funcțiile și variabilele necesare pentru a asigura funcționalitatea corectă și completă a frame-ului de autentificare a utilizatorului.

#### **Variabilele definite de utilizator**

* **private DatabaseHandler handler = null:** este handler-ul care realizează conexiunea dintre aplicație și baza de date;
* **private static int userID = -1:** este ID-ul user-ului deja înregistrat, inițializat cu valoarea -1;
* **private UIManager uim = new UIManager():** este componenta care se ocupă cu stilizarea, în cazul aplicației Visual OETPN Studio, a JoptionPane-urilor care pot apărea, în funcție de situație.

#### **Variabilele predefinite**

* Reprezintă variabilele care nu pot fi modificate (NetBeans IDE le declară automat în cod)
* Reprezintă toate componentele utilizate pentru crearea frame-ului de Sign Up (JLabel-uri, JButton-uri, JTextField-uri, JtextArea, JPanel-uri, JpasswordField-uri șamd.).

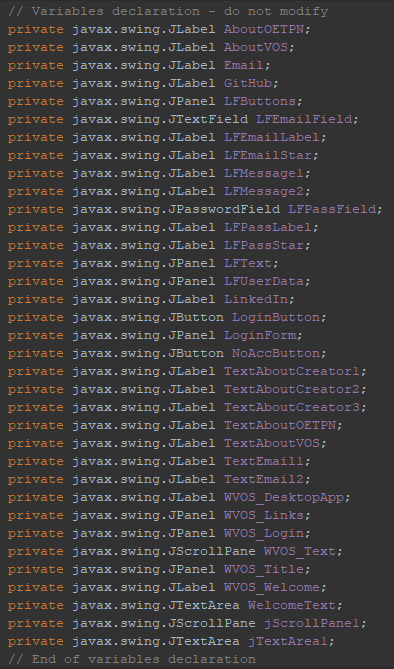


Figura 3.2.3.1.2. Variabilele predefinite (nu pot fi modificate)

#### **Constructorul clasei LoginFrame:** public LoginFrame()

* Inițializează variabila handler, preluând instanța curentă (dacă există) / creând o nouă instanță (dacă nu există);
* Setează titlul ferestrei;
* Elimină focus-ul componentei JtextArea din JPanel-ul WVOS\_Login;
* Setează hyperlink-urile (apelează funcțiile aferente);
* Seteaz[ placeholderel
* Setează stelele / marker-ele de completare;
* Setează poziționarea frame-ului în mijlocul ecranului;
* Setează frame-ul ca fiind vizibil;
* Setează toate componentele;

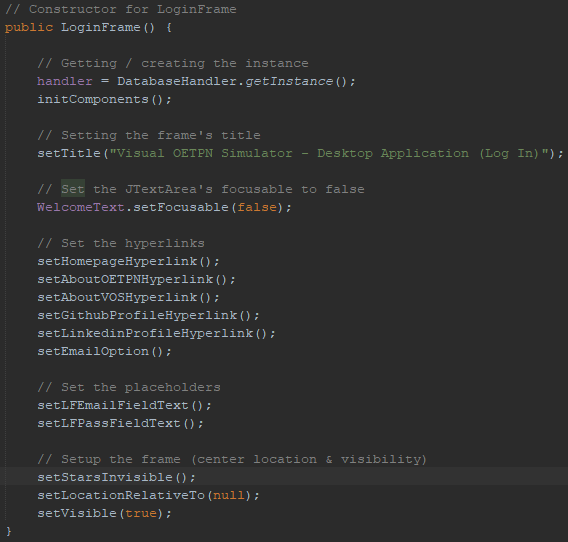


Figura 3.2.3.1.3. Constructorul clasei LoginFrame

#### **Funcția pentru setarea stelelor / marker-elor de completare:** private void setStarsInvisible()

* Setează toate stelele / marker-ele de completare ca fiind invizibile cât timp câmpurile sunt completate;
* Stelele / marker-ele devin vizibile atunci când unul sau mai multe câmpuri rămân necompletate.

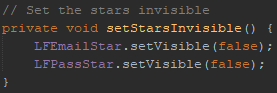


Figura 3.2.3.1.4. Funcția pentru setarea stelelor / marker-elor de completare

#### **Funcția care marchează câmpurile goale:** private boolean markEmptyFields()

* Marchează un câmp dacă acesta este sau nu gol;
* Se creează, în interiorul funcției, o variabilă isEmpty de tip boolean, care este inițializată cu valoarea false;
* Pentru fiecare câmp, dacă acesta este gol, isEmpty ia valoarea true, și steaua / marker-ul corespunzător devine vizibil;
* Funcția returnează variabila isEmpty;

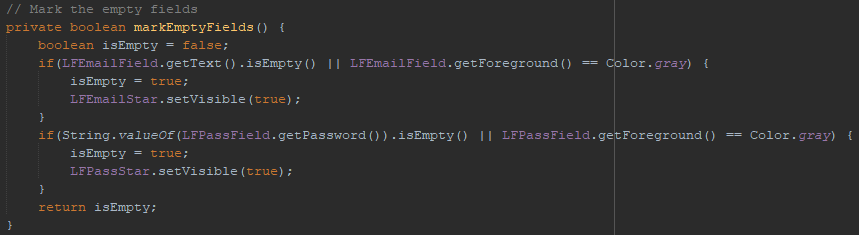


Figura 3.2.3.1.5. Funcția pentru marcarea câmpurilor goale

#### **Funcții pentru focusul field-ului de e-mail (focus gained / lost):** private void LFEmailFieldFocusGained() & LFEmailPasswordFocusLost()

* Reprezintă funcții care implementează evenimentul FocusEvent (focus gained: placeholder-ul nu este vizibil; focus lost: placeholder-ul este vizibil);
* Focus gained: dacă culoarea textului este gri, nu este setat niciun text, iar culoarea acestuia este setată cu valoarea “null”;
* Focus lost: se apelează funcția setLFEmailFieldText (dacă câmpul pentru e-mail este gol, atunci se setează placeholder-ul “E-mail” și culoarea textului va fi gri).

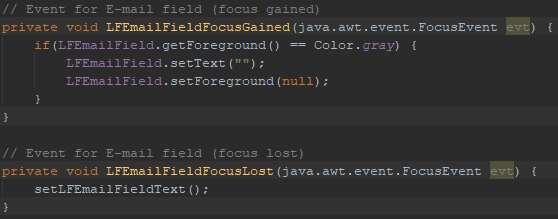


Figura 3.2.3.1.6. Funcții pentru focusul field-ului de e-mail

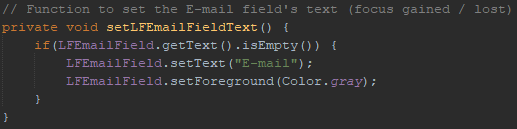


Figura 3.2.3.1.7. Funcția pentru setarea placeholder-ului pentru câmpul de e-mail

#### **Funcții pentru focusul field-ului de parolă (focus gained / lost):** private void LFPassFieldFocusGained() & LFPassFieldFocusLost()

* Reprezintă funcții care implementează evenimentul FocusEvent (focus gained: placeholder-ul nu este vizibil; focus lost: placeholder-ul este vizibil);
* Focus gained: dacă culoarea textului este gri, nu este setat niciun text, iar culoarea acestuia este setată cu valoarea “null”;
* Focus lost: se apelează funcția setLFPassFieldText (dacă câmpul pentru e-mail este gol, atunci se setează placeholder-ul “Password” și culoarea textului va fi gri).

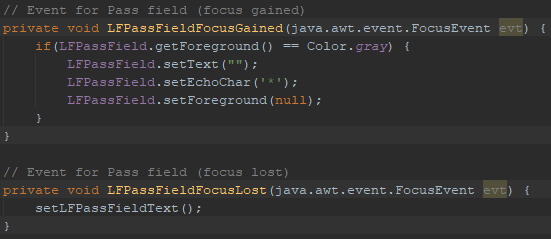


Figura 3.2.3.1.8. Funcții pentru focusul field-ului de parolă

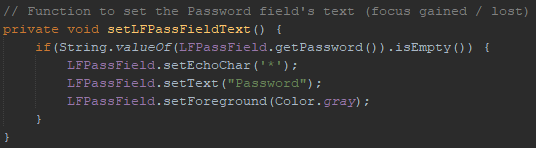


Figura 3.2.3.1.9. Funcția pentru setarea placeholder-ului pentru câmpul de parolă

#### **Funcția pentru inițializarea tuturor componentelor din Login Frame:** private void initComponents()

* Funcție generată automat;
* Reprezintă funcția care inițializează, configurează și setează fiecare componentă din JFrame-ul LoginFrame (JLabel-uri, JButton-uri, JPanel-uri, JTextField-uri).

#### **Eveniment generat în momentul în care se apasă butonul Log In:** private void LoginButtonActionPerformed(java.awt.event.ActionEvent evt)

* Reprezintă funcția care setează un eveniment care trebuie să se întâmple în momentul în care butonul Log In este apăsat;
* Evenimentul este actionPerformed, care este implementat de funcția LoginButtonActionPerformed;
* Scenariile posibile, după apăsarea butonului, sunt:
* Unul sau ambele câmpuri au rămas necompletate: “Please enter all fields!”;
* Adresa de e-mail sau parola este incorectă: “Wrong username or password!”;
* Toate datele au fost introduse corect: “Sign Up Successful! Please log in to start using VOS!”.

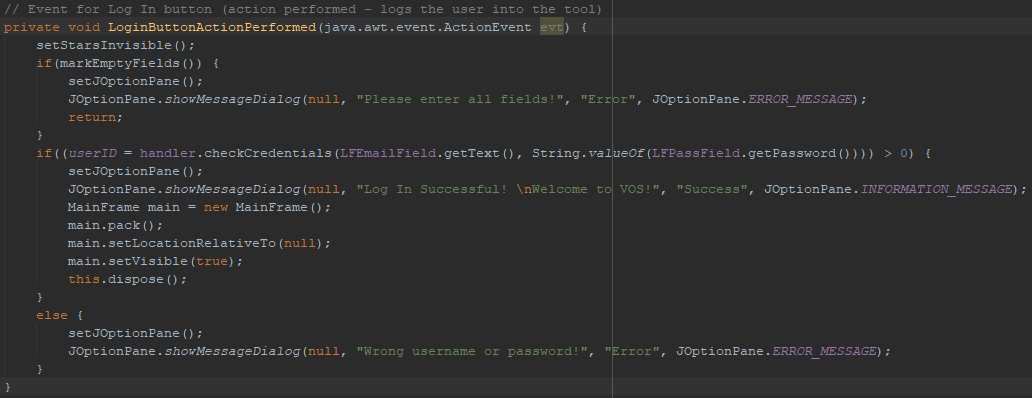
****

Figura 3.2.3.1.10. Funcția LoginButtonActionPerformed, care implementează evenimentul actionPerformed() (vor avea loc evenimente în funcție de cazurile menționate mai sus)

#### **Eveniment generat în momentul în care se apasă butonul NoAccButton:** private void NoAccButtonActionPerformed(java.awt.event.ActionEvent evt)

* Reprezintă funcția care setează un eveniment care trebuie să se întâmple în momentul în care butonul No Account este apăsat;
* Evenimentul este actionPerformed, care este implementat de funcția NoAccButtonActionPerformed;
* După apăsarea acestui buton, utilizatorul va fi redirecționat către pagina de înregistrare: se va deschide fereastra de înregistrare în mijlocul ecranului.

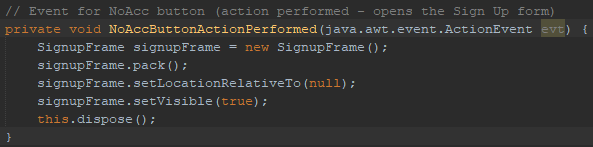


Figura 3.2.3.1.11. Funcția HaveAccButtonActionPerformed, care implementează evenimentul actionPerformed() (se va deschide fereastra de autentificare – Log In)

#### **Funcția pentru setarea elementului JOptionPane:** private void setJOptionPane()

* Reprezintă funcția care setează anumiți parametri pentru ferestrele de dialog;
* Parametrii care sunt configurați sunt:
* Fundalul ferestrei de dialog;
* Fundalul panel-ului;
* Culoarea textului;
* Culoarea de fundal a butonului;
* Culoarea textului butonului;

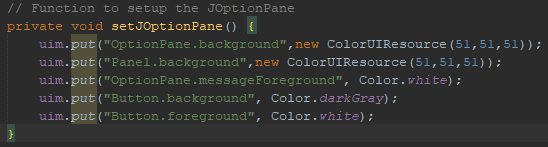


Figura 3.2.3.1.9. Funcția care setează ferestrele de dialog

#### **Funcția pentru setarea hyperlink-ului către pagina Home (VOS Web):** private void setHomepageHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina Home din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setHomepageHyperlink() se poate vedea în figura de mai jos.

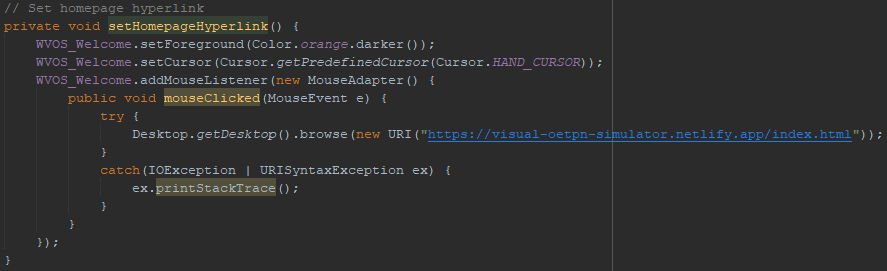


Figura 3.2.3.1.10. Funcția pentru setarea hyperlink-ului către pagina Home

#### **Funcția pentru setarea hyperlink-ului către pagina About OETPNs (VOS Web):** private void setAboutOETPNHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina About OETPNs din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setAboutOETPNHyperlink() se poate vedea în figura de mai jos.

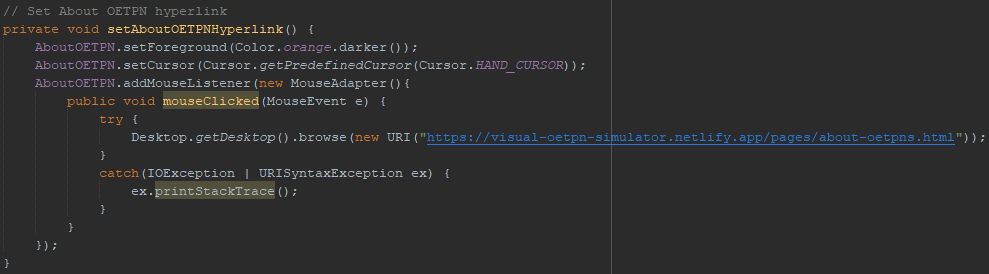


Figura 3.2.3.1.11. Funcția pentru setarea hyperlink-ului către pagina About OETPNs

#### **Funcția pentru setarea hyperlink-ului către pagina About VOS (VOS Web):** private void setAboutVOSHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide pagina About VOS din cadrul aplicației VOS Web
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setAboutVOSHyperlink() se poate vedea în figura de mai jos.

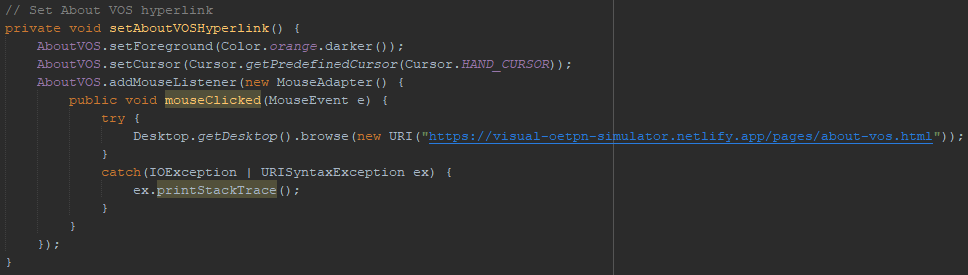


Figura 3.1.3.12. Funcția pentru setarea hyperlink-ului către pagina About VOS

#### **Funcția pentru setarea hyperlink-ului către profilul personal de GitHub al dezvoltatorului:** private void setGithubProfileHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide profilul personal de GitHub al dezvoltatorului aplicației
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setGithubProfileHyperlink() se poate vedea în figura de mai jos.

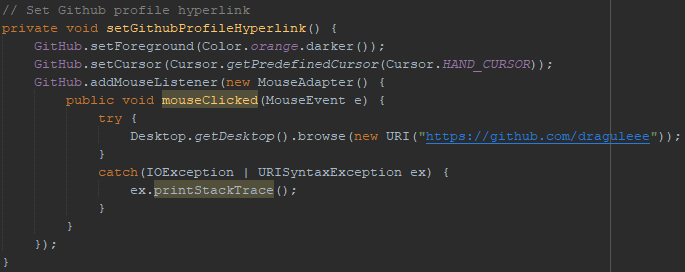


Figura 3.2.3.1.13. Funcția pentru setarea hyperlink-ului către profilul personal de GitHub al creatorului aplicației

#### **Funcția pentru setarea hyperlink-ului către profilul personal de LinkedIn al dezvoltatorului:** private void setLinkedinProfileHyperlink()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide profilul personal de LinkedIn al dezvoltatorului aplicației
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setLinkedinProfileHyperlink() se poate vedea în figura de mai jos.

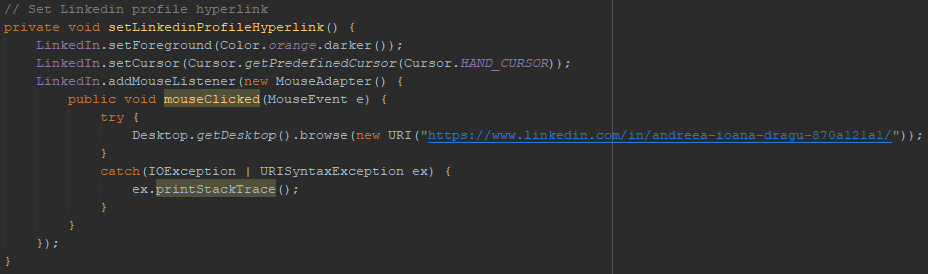


Figura 3.2.3.1.14. Funcția pentru setarea hyperlink-ului către profilul personal de Linkedin al creatorului aplicației

1. **Funcția pentru setarea opțiunii de trimitere a unui e-mail către creatorul aplicației:** private void setEmailOption()

* Reprezintă funcția care implementează evenimentul mouseClicked(), pentru a deschide mesageria electronică (utilizatorul poate lua legătura cu dezvoltatorul aplicației direct prin e-mail)
* Pentru a diferenția label-ul corespunzător de label-urile obișnuite, se va seta culoarea la portocaliu închis și i se va adăuga un cursor
* Evenimentul mouseClicked() implementat de funcția setEmailOption() se poate vedea în figura de mai jos.

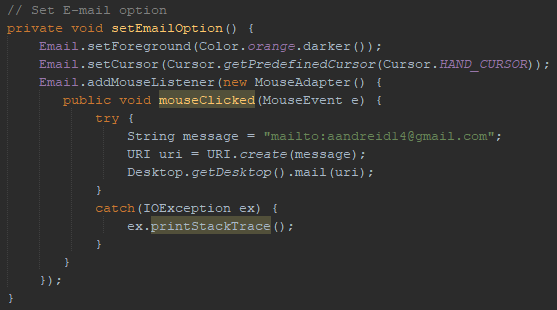


Figura 3.1.3.15. Funcția pentru setarea opțiunii de trimitere a unui e-mail către creatorul aplicației

#### **Metoda main:** public static void main(String[] args)

* Reprezintă punctul de intrare a programului;
* Fără această metodă, Mașina Virtuală Java nu va executa programul;
* Pentru clasa LoginFrame, în metoda main, vom avea:
* Metoda invokeLater(...): asigură apelul în siguranță al metodelor în clasele Swing;
* Parametrii metodei invokeLater: interfața Runnable, care trebuie să ofere o implementare pentru metoda run();
* Metoda run(): vom avea un bloc try-catch:
* Blocul try: se setează Look And Feel-ul FlatDarkLaf (link: <https://www.formdev.com/flatlaf/>);
* Blocul catch: se prinde excepția aruncată.
* Se crează un nou obiect de clasă LoginFrame (un nou frame de autentificare) și i se setează Look And Feel-ul importat.

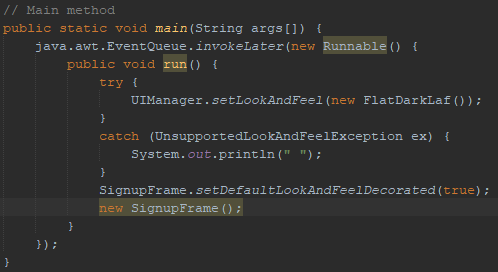


Figura 3.1.3.16. Funcția / metoda main

## Clasa MainFrame

# Bibliografie

|  |  |
| --- | --- |
| [1] | P. Nume, "Titlul capitolului," în *Titlul cartii*, Oras, Editura, 2016, pp. 1-24. |
| [2] | P. Nume, "Titlul articolului," *Titlul revistei,* vol. 1, no. 2, pp. 22-30, 2016. |
| [3] | P. Nume, "Titlul articolului," în *Numele conferintei*, Oras, 2015. |
| [4] | "IEEE Citation Reference," 2009. [Online]. Available: https://www.ieee.org/documents/ieeecitationref.pdf. |
| [5] | "IEEE Editorial Style Manual," 2016. [Online]. Available: https://www.ieee.org/documents/style\_manual.pdf. |