| **Project – External Documentation** |  |
| --- | --- |
| <ISYS6197003>  <Business Application Developement> |
| <Odd/Even/Compact> Semester Year <9999/9999> |

* Project Title

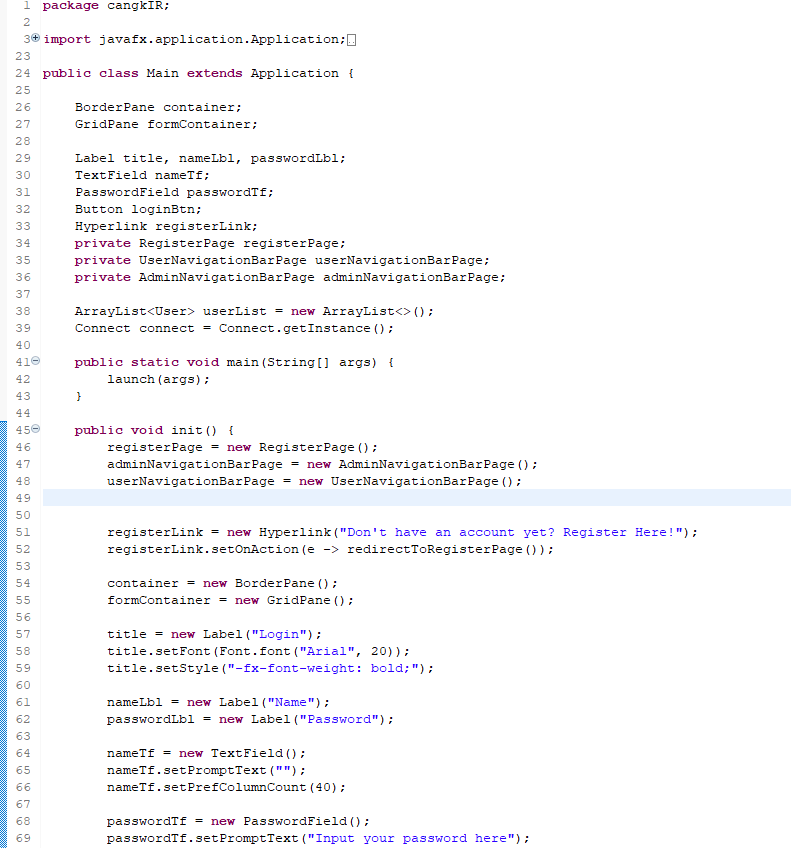
cangkIR

* Introduction

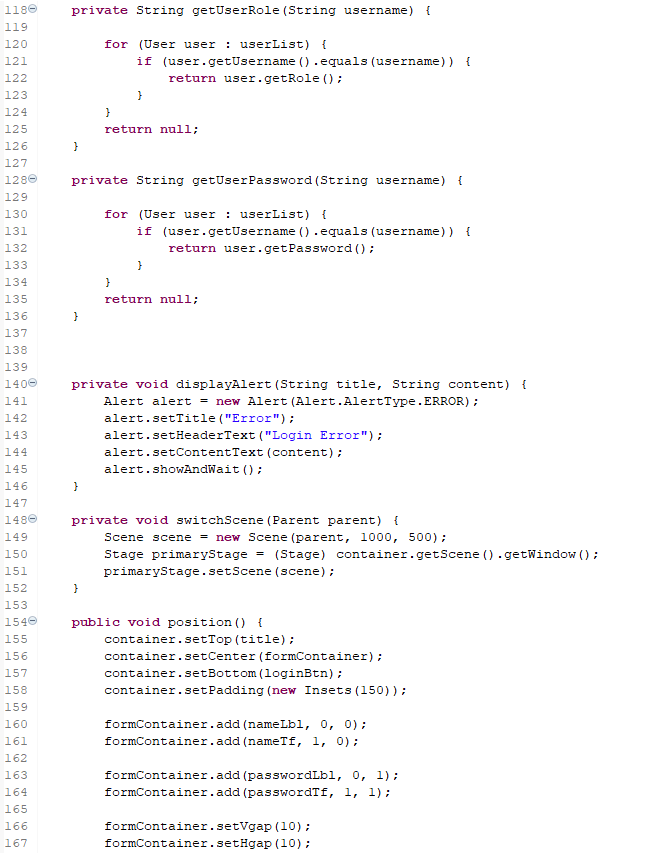
cangkIR is an online shop that sells various cups. The cups come in various shapes and sizes with a large price range as well. cangkIR does not have a physical shop. So, customers must use the cangkIR application to buy the cups. Inside the application, people can be assigned to either have an Admin role or a User role. User can buy the cups by inserting them into their cart first, then checking out their cart. Admin can manage the cups available. They can add, update and delete cups from the database.

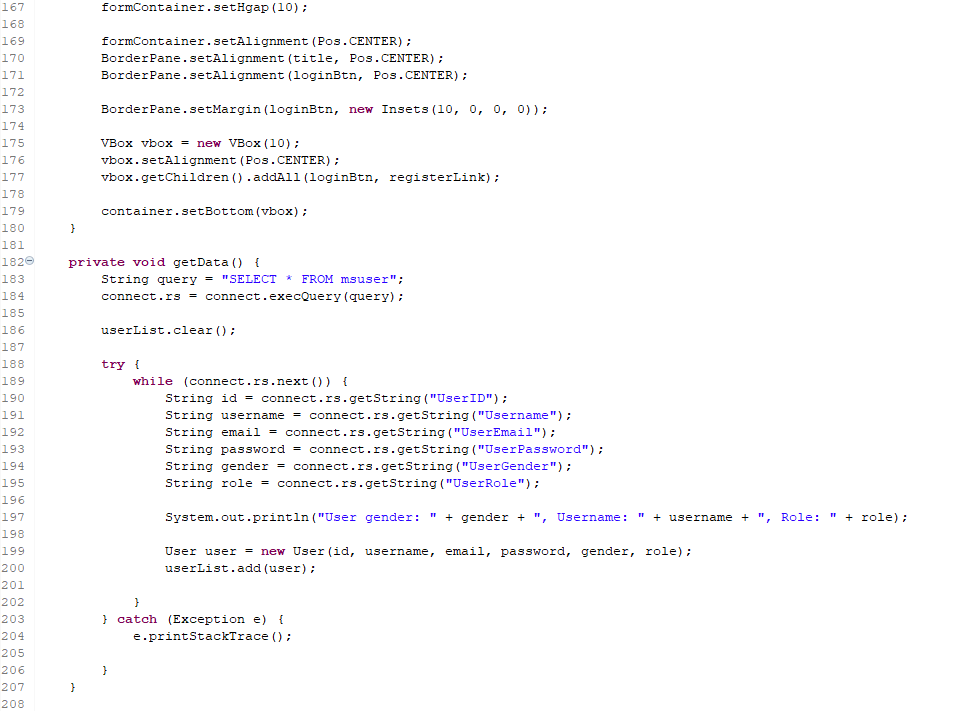
* Report / Documentation

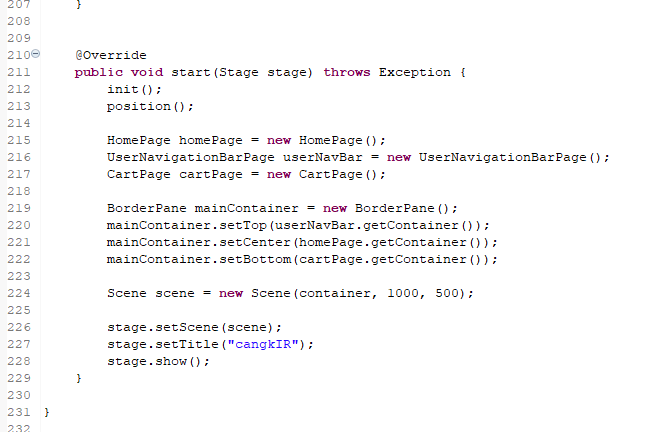
1. Log In Page





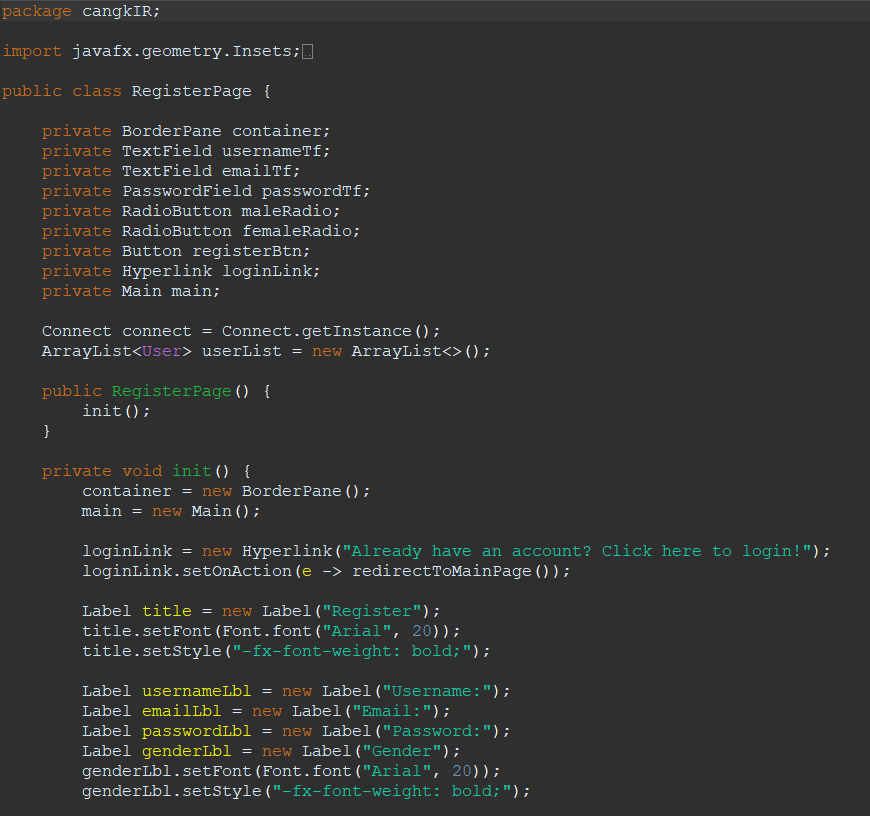


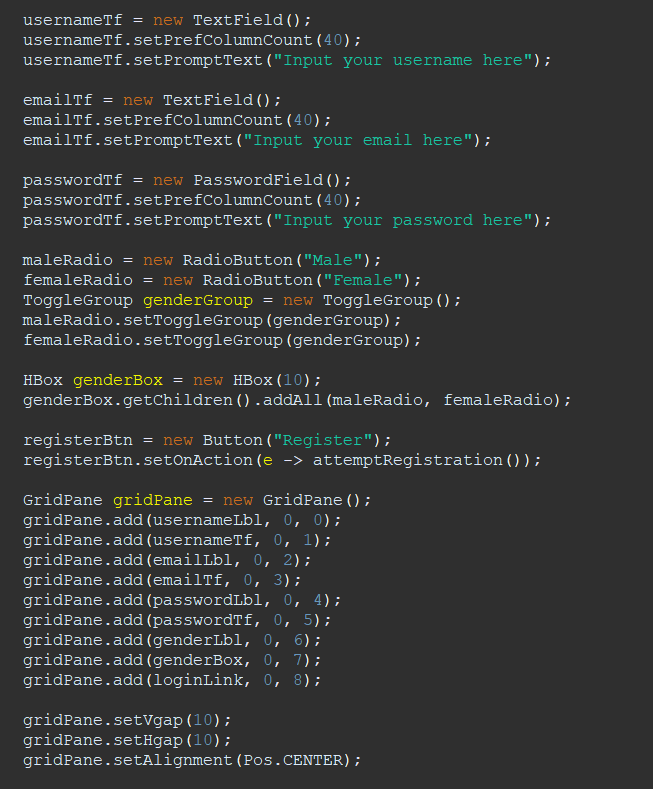


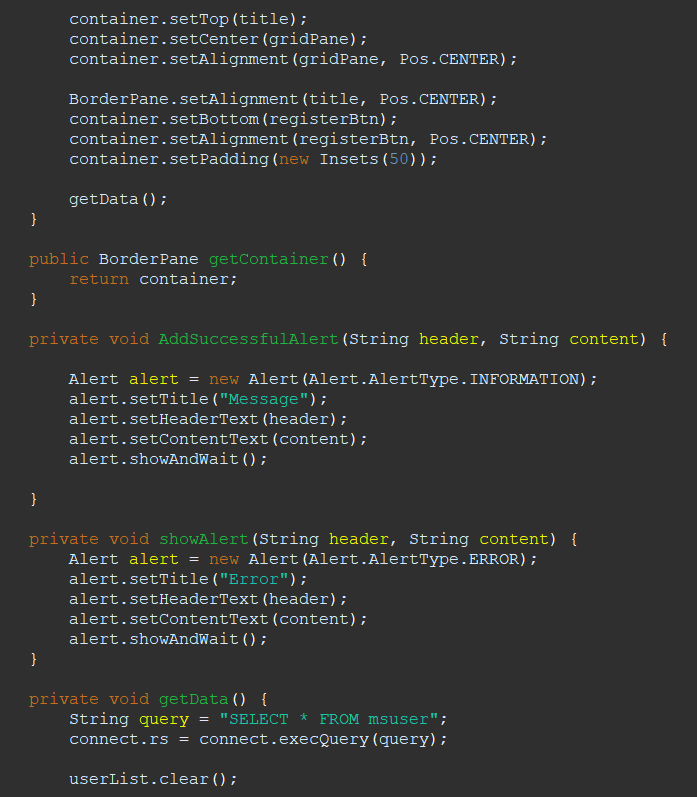


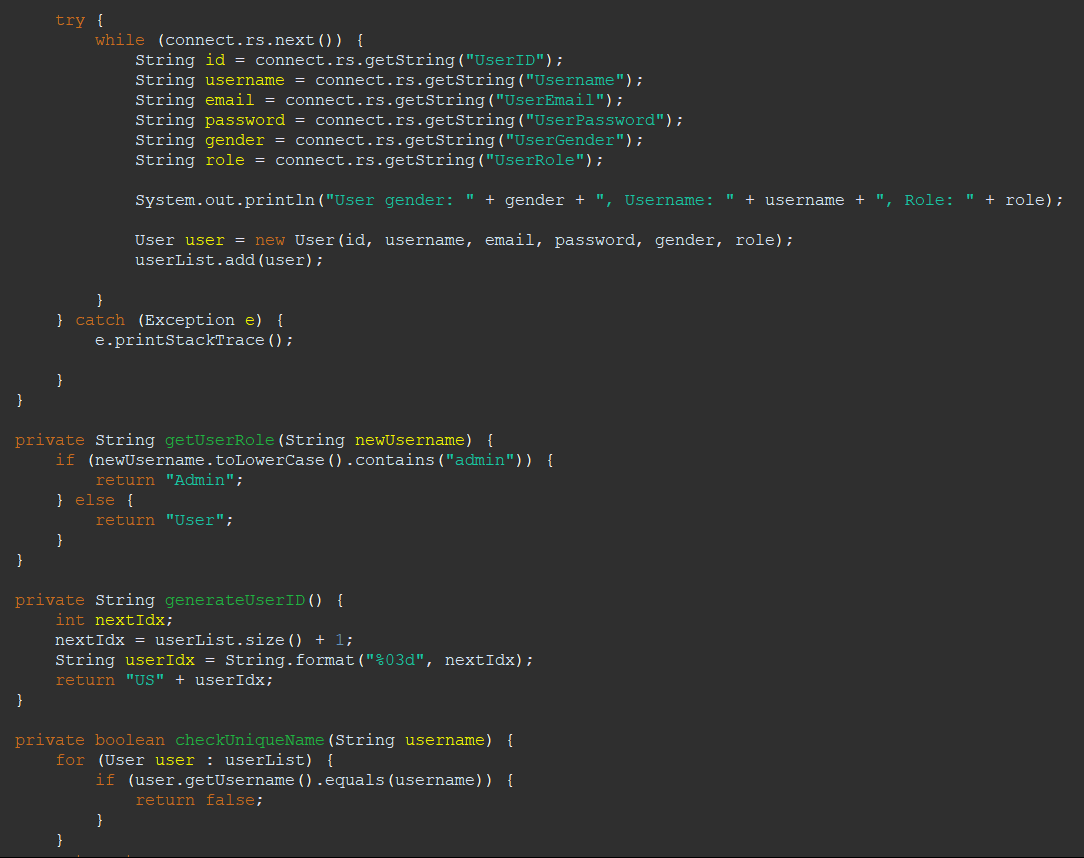
* BorderPane dan GridPane:
  + BorderPane yang kami gunakan untuk menyusun elemen-elemen GUI di dalamnya.
  + formContainer adalah GridPane yang kami gunakan sebagai wadah untuk elemen-elemen formulir.
* Label, TextField, dan PasswordField:
  + title, nameLbl, dan passwordLbl adalah label-label untuk menunjukkan judul dan label untuk kolom nama dan kata sandi.
  + nameTf dan passwordTf adalah TextField dan PasswordField yang kami gunakan untuk memasukkan nama user dan kata sandi.
* Button dan Hyperlink:
  + loginBtn adalah tombol yang, jika kami tekan, akan memanggil metode attemptLogin() untuk melakukan proses login.
  + registerLink adalah tautan yang, jika kami klik, akan memanggil metode redirectToRegisterPage() untuk menuju halaman pendaftaran.
* Method Init:
  + init() kami gunakan untuk inisialisasi, termasuk membuat objek halaman pendaftaran dan mendapatkan data user dari database.
* Method attemptLogin:
  + attemptLogin() kami panggil saat tombol login kami tekan. Kami periksa input nama user dan kata sandi, kemudian kami putuskan apakah login berhasil atau tidak.
* Method redirectToRegisterPage:
  + redirectToRegisterPage() kami gunakan untuk mengarahkan kami ke halaman pendaftaran saat tautan pendaftaran kami klik.
* Method getUserRole dan getUserPassword:
  + getUserRole dan getUserPassword kami gunakan untuk mendapatkan peran dan kata sandi kami berdasarkan nama user.
* Method displayAlert dan switchScene:
  + displayAlert menampilkan pesan kesalahan jika terjadi kesalahan saat login.
  + switchScene kami gunakan untuk beralih antar halaman.
* Method getData:
  + getData() kami gunakan untuk mendapatkan data user dari database dan memasukkannya ke dalam ArrayList userList.
* Method position:
  + position() kami gunakan untuk mengatur posisi elemen-elemen dalam halaman, termasuk label, teks field, tombol, dan tautan.
* Method start:
  + start() adalah method utama yang kami panggil saat aplikasi JavaFX kami jalankan. Kami memanggil init() dan position() untuk menginisialisasi dan menata tata letak elemen-elemen di halaman.

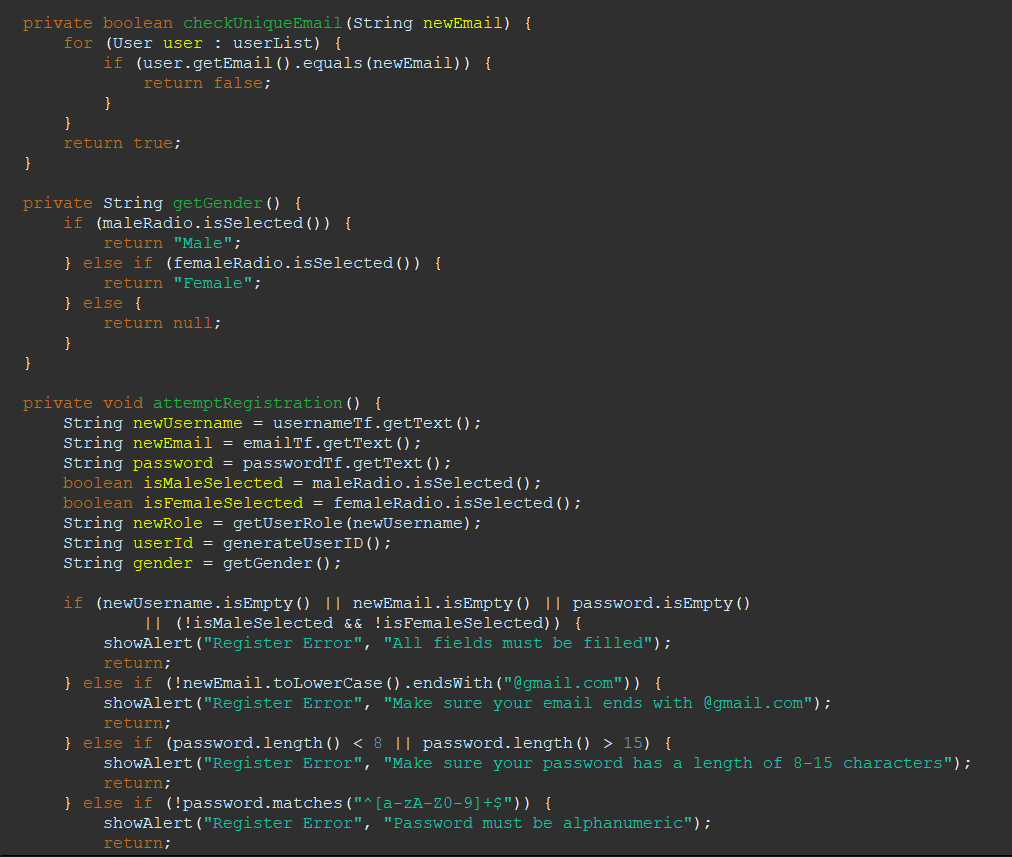
2. Register Page

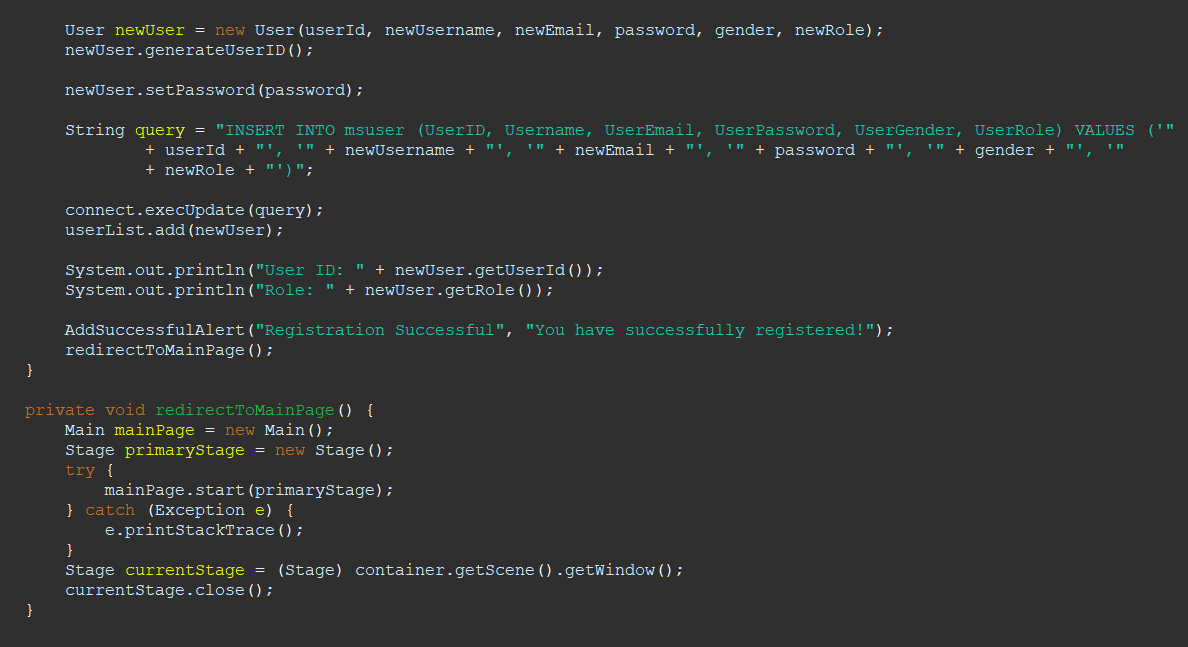






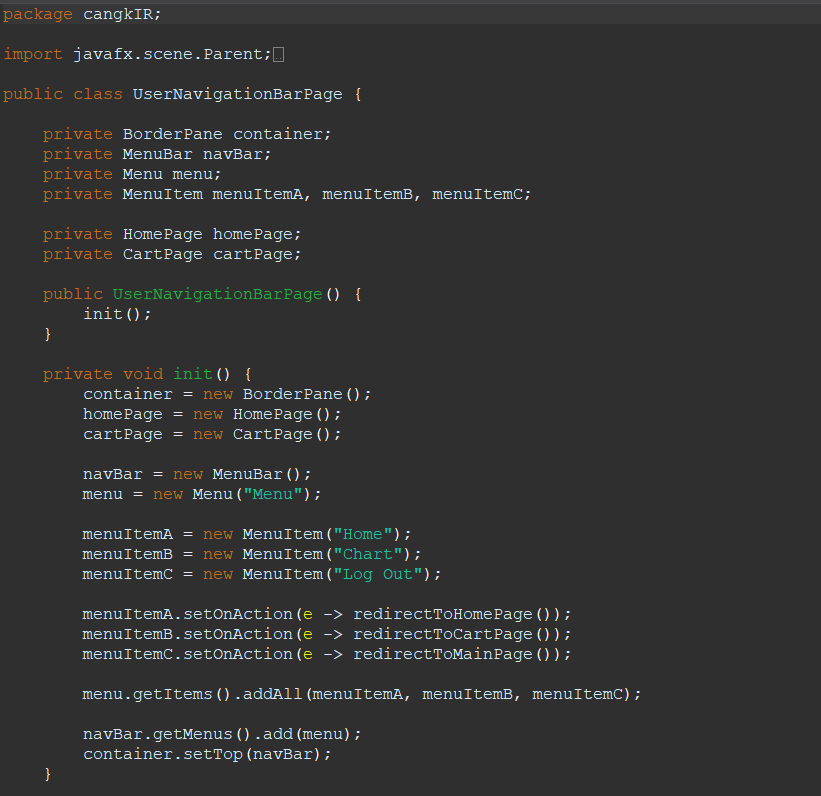






* BorderPane dan GridPane:
  + BorderPane kami gunakan untuk menyusun elemen-elemen GUI di dalamnya.
  + init() merupakan metode inisialisasi yang membuat objek-objek untuk menampilkan elemen-elemen halaman registrasi.
* Label, TextField, PasswordField, RadioButton, dan Button:
  + Elemen-elemen seperti Label, TextField, PasswordField, RadioButton, dan Button kami gunakan untuk memasukkan informasi pendaftaran seperti username, email, password, dan jenis kelamin.
* Validasi pada Metode attemptRegistration():
  + Melakukan validasi terhadap input user, seperti memastikan semua field terisi, email berakhir dengan "@gmail.com", panjang password antara 8-15 karakter inclusively, dan password bersifat alfanumerik.
* Method checkUniqueEmail dan checkUniqueName:
  + Digunakan untuk memastikan bahwa email dan username yang dimasukkan user unik dan belum terdaftar di database.
* Method redirectToMainPage
  + Digunakan untuk beralih ke halaman utama setelah proses pendaftaran selesai. Membuat objek Main dan menutup stage saat ini (halaman registrasi).
* Hyperlink untuk Login:
  + loginLink adalah hyperlink yang, jika diklik, akan memanggil metode redirectToMainPage() untuk kembali ke halaman login jika user sudah memiliki akun.
* Menyusun Elemen-elemen di GridPane:
  + GridPane untuk menyusun elemen-elemen seperti label, teks field, radio button, dan hyperlink dengan jarak dan tata letak yang sesuai.
* Pengaturan Tata Letak dan Padding:
  + Mengatur tata letak dan padding pada container dan elemen-elemen agar tampilan halaman registrasi terlihat rapi.

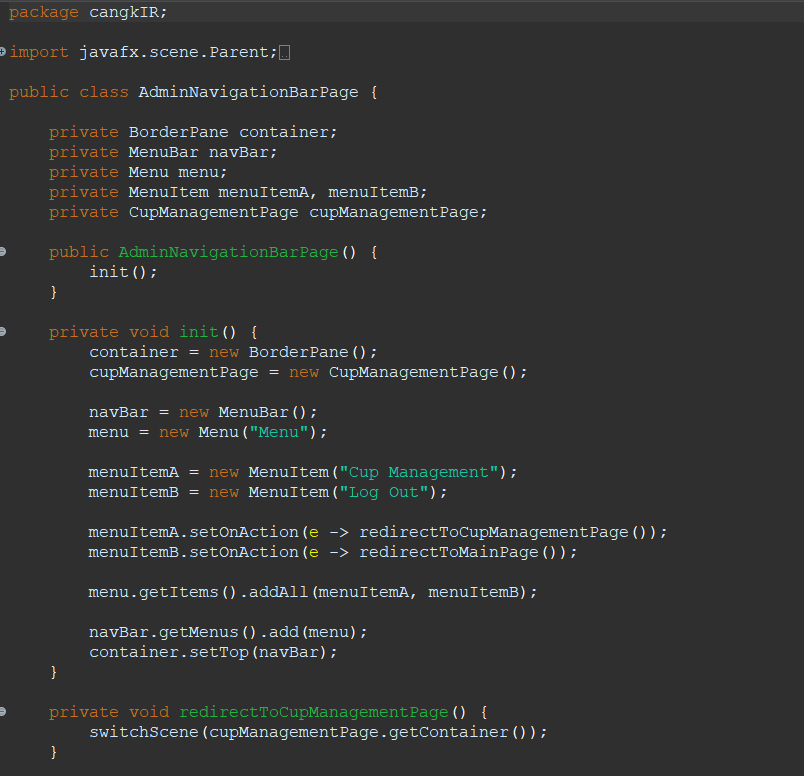
3. User Navigation Bar

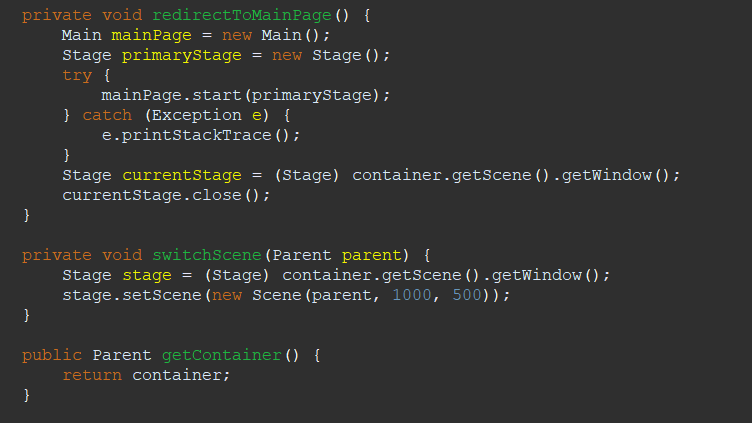




* Deklarasi Variabel:
  + container: Objek BorderPane yang kami gunakan sebagai wadah utama halaman navigasi user.
  + navBar: Objek MenuBar yang berisi menu untuk navigasi user.
  + menu: Sebuah objek Menu yang berisi opsi menu untuk navigasi user.
  + menuItemA, menuItemB, menuItemC: Objek MenuItem yang mewakili opsi menu seperti "Home", "Chart", dan "Log Out".
  + homePage, cartPage: Objek-objek dari kelas HomePage dan CartPage yang akan diakses melalui menu navigasi.
* Konstruktor:
  + Konstruktor UserNavigationBarPage memanggil Method init() saat objek kelas dibuat
* Method init():
  + Method ini digunakan untuk menginisialisasi komponen-komponen halaman navigasi user.
  + Membuat objek Menu dan menambahkannya ke MenuBar.
  + Membuat objek MenuItem dan menambahkannya ke dalam Menu.
  + Menetapkan aksi (event handler) untuk setiap MenuItem.
  + Menetapkan MenuBar ke bagian atas BorderPane.
* Method redirectToHomePage():
  + Method ini dipanggil saat user memilih opsi "Home" pada menu.
  + Menggunakan Method switchScene() untuk beralih ke halaman utama (homePage).
* Method redirectToCartPage():
  + Methode ini dipanggil saat user memilih opsi "Chart" pada menu.
  + Menggunakan Method witchScene() untuk beralih ke halaman keranjang (cartPage).
* Method redirectToMainPage():
  + Method ini dipanggil saat user memilih opsi "Log Out" pada menu.
  + Membuat objek Main dan memanggil Method start() untuk kembali ke halaman log in.
  + Menutup stage saat ini.
* Method switchScene(Parent parent):
  + Method ini digunakan untuk beralih antara scene (halaman) dalam stage.
  + Mengganti scene dengan scene baru yang diambil dari parameter Parent.
  + Menggunakan stage yang sedang berjalan untuk mengatur scene yang baru.
* Method getContainer():
  + Method ini mengembalikan objek Parent yang merupakan wadah utama halaman navigasi user.

4. Admin Navigation Bar

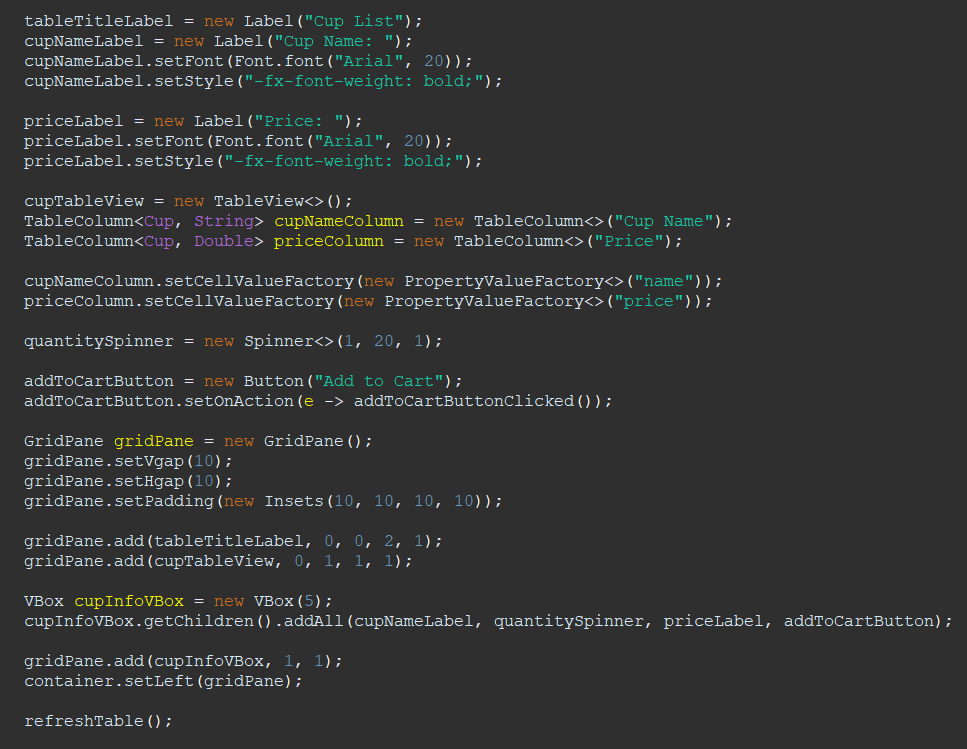




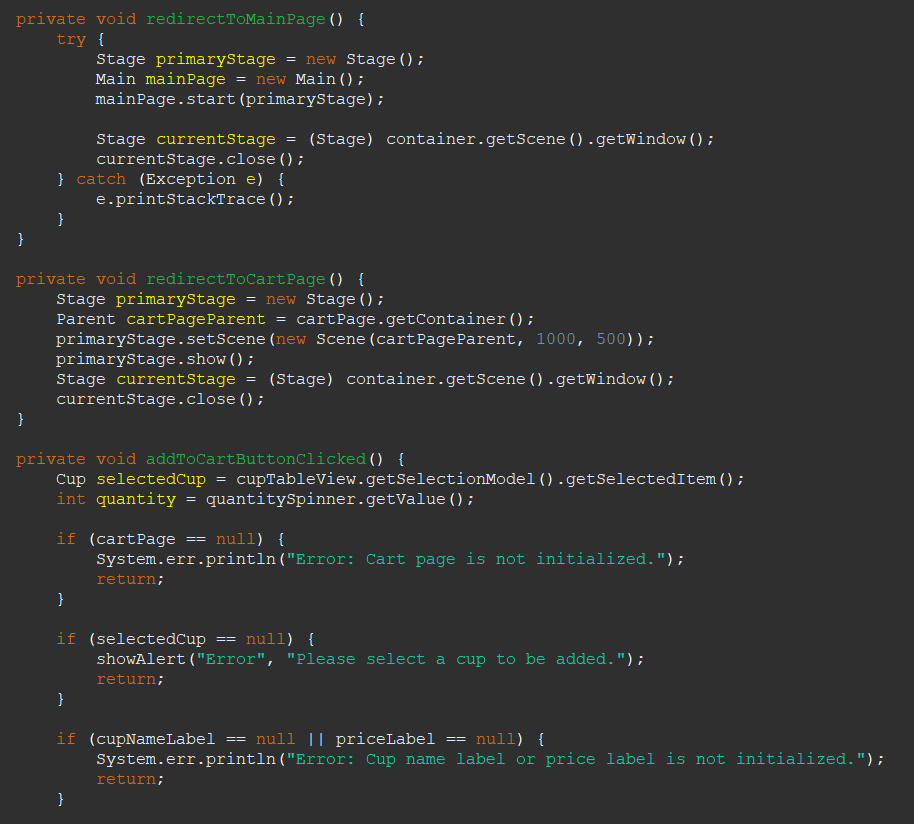
* Deklarasi Variabel:
  + container: Sebuah objek BorderPane yang kami gunakan sebagai wadah utama untuk menyusun elemen-elemen dalam tampilan.
  + navBar: Sebuah objek MenuBar yang berisi elemen-elemen navigasi.
  + menu: Sebuah objek Menu yang berisi item-menu untuk navigasi.
  + menuItemA dan menuItemB: Objek MenuItem yang mewakili opsi-opsi navigasi di dalam Menu.
* Konstruktor:
  + AdminNavigationBarPage() Constructor: Constructor digunakan untuk inisialisasi kelas dan memanggil metode init().
* Method init():
  + Membuat objek CupManagementPage untuk menangani halaman pengelolaan cangkir.
  + Membuat MenuBar dan MenuItem yang berisi opsi-opsi navigasi.
  + Menetapkan event handler untuk setiap MenuItem untuk menentukan perilaku saat dipilih.
  + Menambahkan elemen-elemen ke dalam container untuk menyusun tampilan.
* redirectToCupManagementPage() Method:
  + Method ini dipanggil saat MenuItem "Cup Management" dipilih. Admin akan beralih ke halaman pengelolaan cangkir.
* redirectToMainPage() Method:
  + Method ini dipanggil saat MenuItem "Log Out" dipilih. Admin akan beralih kembali ke halaman utama dan menutup halaman administrasi saat ini.
* switchScene(Parent parent) Method:
  + Method ini digunakan untuk melakukan perpindahan antar scene (halaman) dalam aplikasi dengan mengganti tampilan pada container dengan tampilan yang baru.
* getContainer() Method:
  + Method ini mengembalikan objek container yang berisi tampilan navigasi admin, sehingga dapat digunakan dalam menampilkan kontennya pada scene utama.

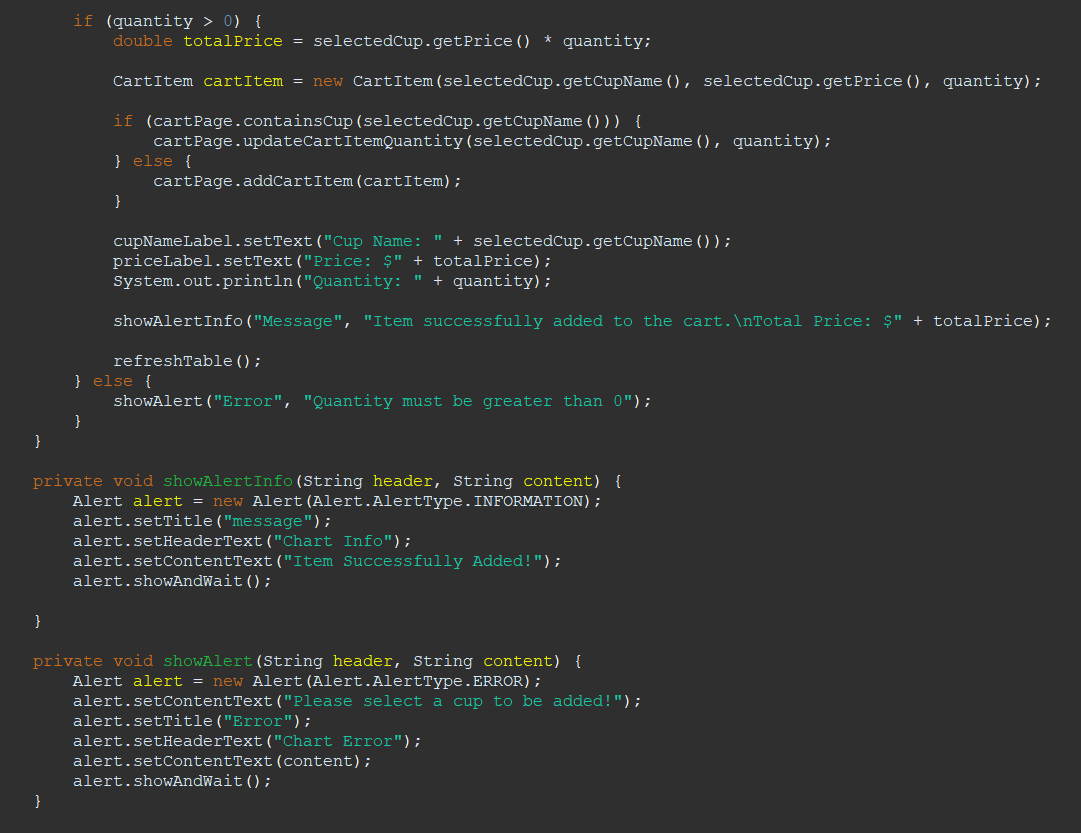
5. Home Page

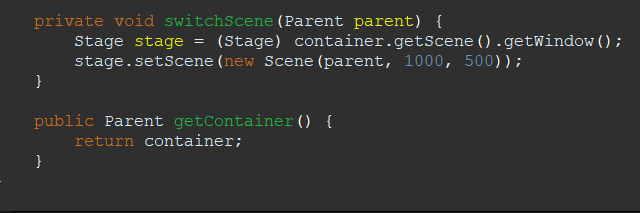












Variabel Kelas

* navBar, menu, menuItemA, menuItemB, menuItemC: Menangani elemen-elemen menu dan navigasi di bagian atas halaman.
* cartPage: Objek CartPage yang akan menangani tampilan cart.
* container: BorderPane yang digunakan sebagai wadah utama untuk menyusun elemen-elemen dalam tampilan.
* tableTitleLabel, cupNameLabel, priceLabel: Label-label yang menampilkan informasi di bagian atas tabel.
* cupTableView: TableView untuk menampilkan daftar cangkir.
* quantitySpinner: Spinner untuk memilih jumlah item yang akan ditambahkan ke keranjang belanja.
* addToCartButton: Tombol untuk menambahkan item ke keranjang belanja.

Method

1. HomePage(): Konstruktor kelas. Memanggil metode init() untuk inisialisasi.
2. init(): Method inisialisasi utama:
   * Membuat dan mengatur elemen-elemen GUI seperti MenuBar, TableView, label, tombol, dsb.
   * Menghubungkan aksi untuk setiap MenuItem dan tombol.
   * Memanggil refreshTable() untuk menampilkan data cangkir dalam TableView.
3. getData(): Mengambil data cangkir dari database
   * Menggunakan objek Connect untuk eksekusi query SQL dan mengisi cupList dengan objek Cup yang sesuai.
4. refreshTable(): Merefresh isi TableView dengan data terbaru dari database.
   * Memanggil getData() untuk mendapatkan data terbaru.
   * Membuat ObservableList dan menetapkan datanya ke TableView.
5. redirectToMainPage(): Beralih kembali ke halaman utama aplikasi.
   * Membuat objek Main dan memulai aplikasi.
6. redirectToCartPage(): Beralih ke halaman keranjang belanja.
   * Membuat jendela baru untuk menampilkan CartPage.
   * Menutup jendela utama.
7. addToCartButtonClicked(): Menangani penambahan item ke keranjang belanja.
   * Mendapatkan cangkir yang dipilih dan jumlah dari elemen GUI.
   * Menambahkan item ke keranjang belanja dan memperbarui tampilan.
   * Menampilkan pesan sukses dan merefresh tabel.
   * selectedCup == null: Pastikan user telah memilih cangkir sebelum menambahkannya ke keranjang belanja. Jika tidak, pesan kesalahan akan ditampilkan.
   * quantity > 0: Memeriksa bahwa jumlah item yang akan ditambahkan ke keranjang belanja lebih dari 0. Jika tidak, pesan kesalahan akan muncul.
8. showAlertInfo(): Menampilkan pesan informasi dengan tipe INFORMATION.
   * Digunakan untuk memberi tahu user bahwa item berhasil ditambahkan ke keranjang belanja.
9. showAlert(): Menampilkan pesan kesalahan dengan tipe ERROR.

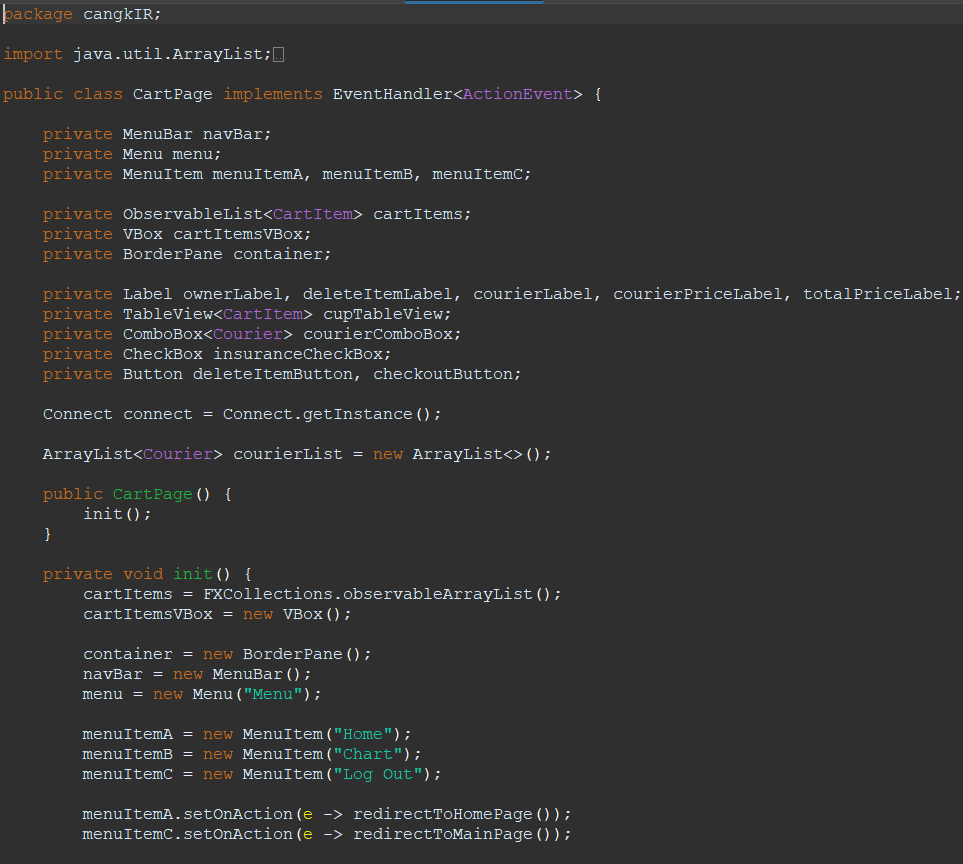
* Digunakan untuk memberi tahu user jika ada kesalahan dalam operasi.

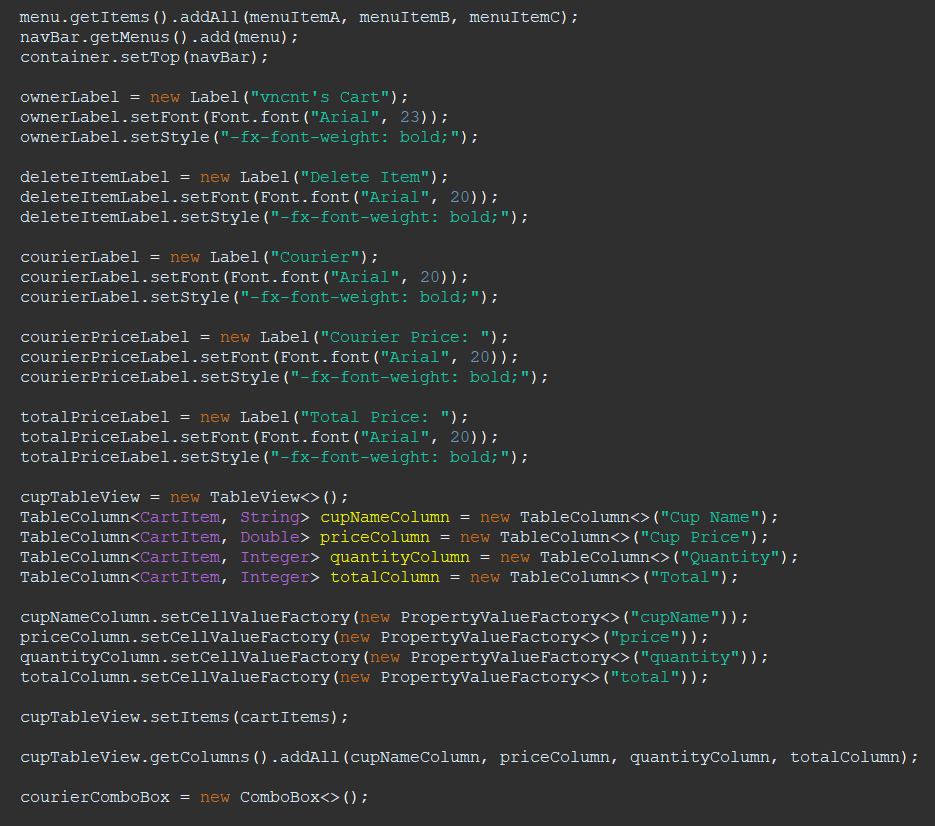
10. switchScene(Parent parent): Beralih ke tampilan lain.

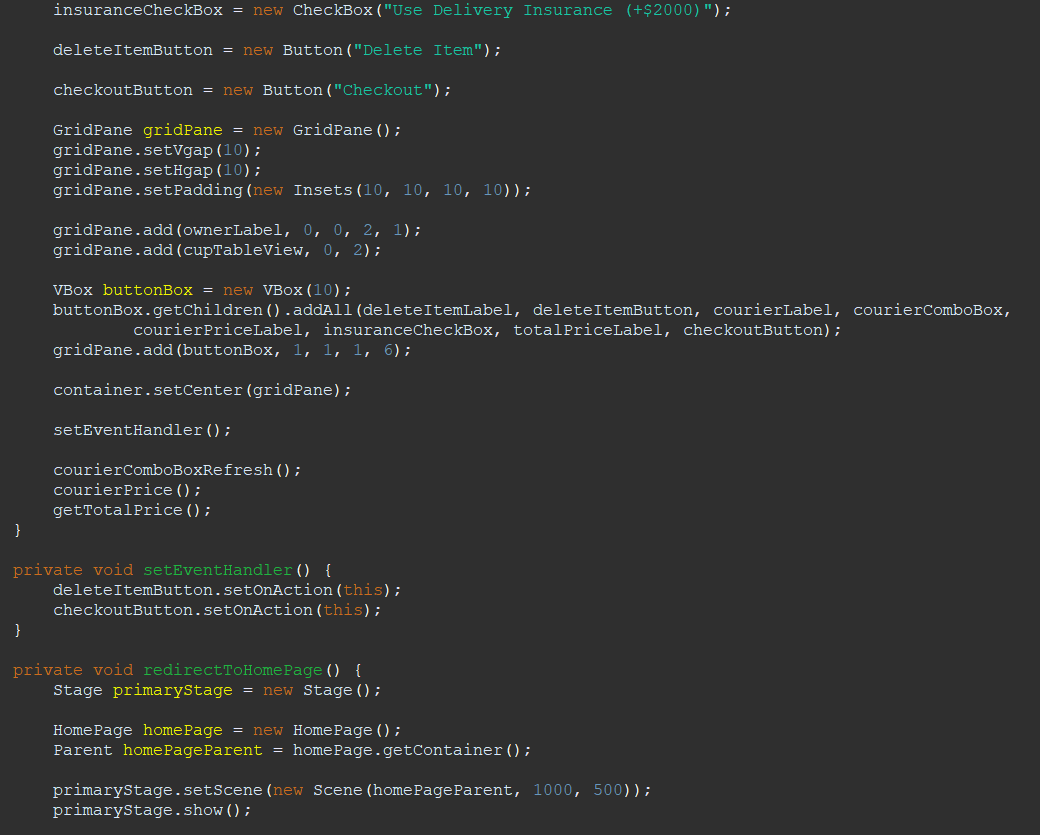
* + Digunakan untuk mengganti tampilan di jendela utama dengan tampilan lainnya.

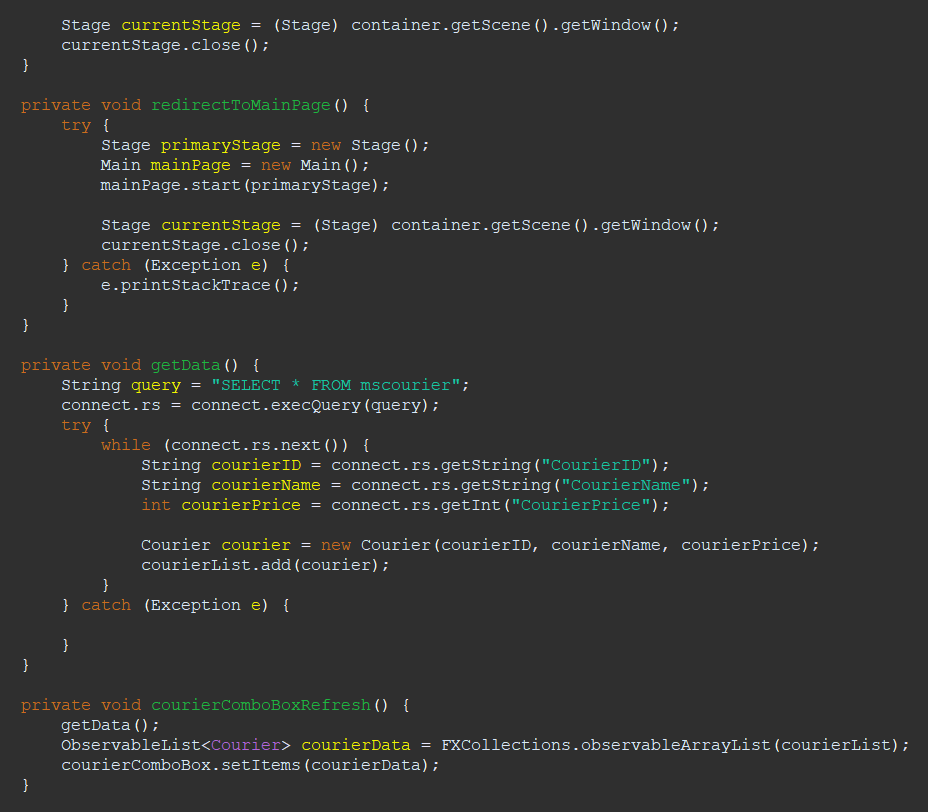
11. getContainer(): Mendapatkan BorderPane dari halaman utama.

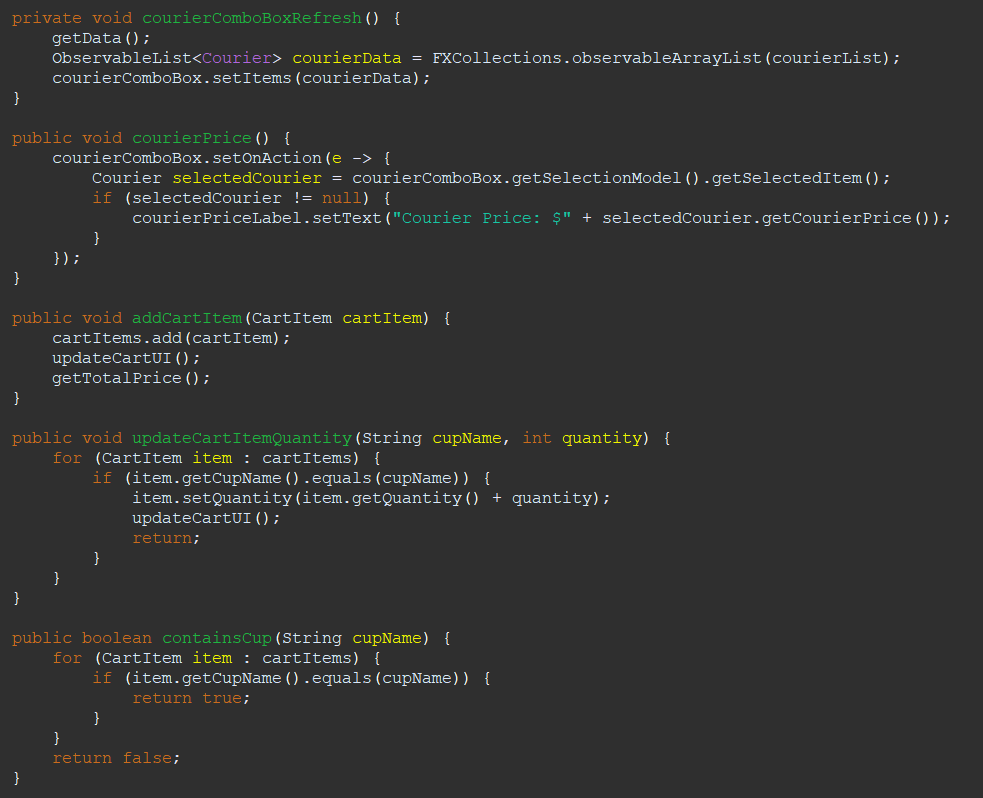
6. Cart Page

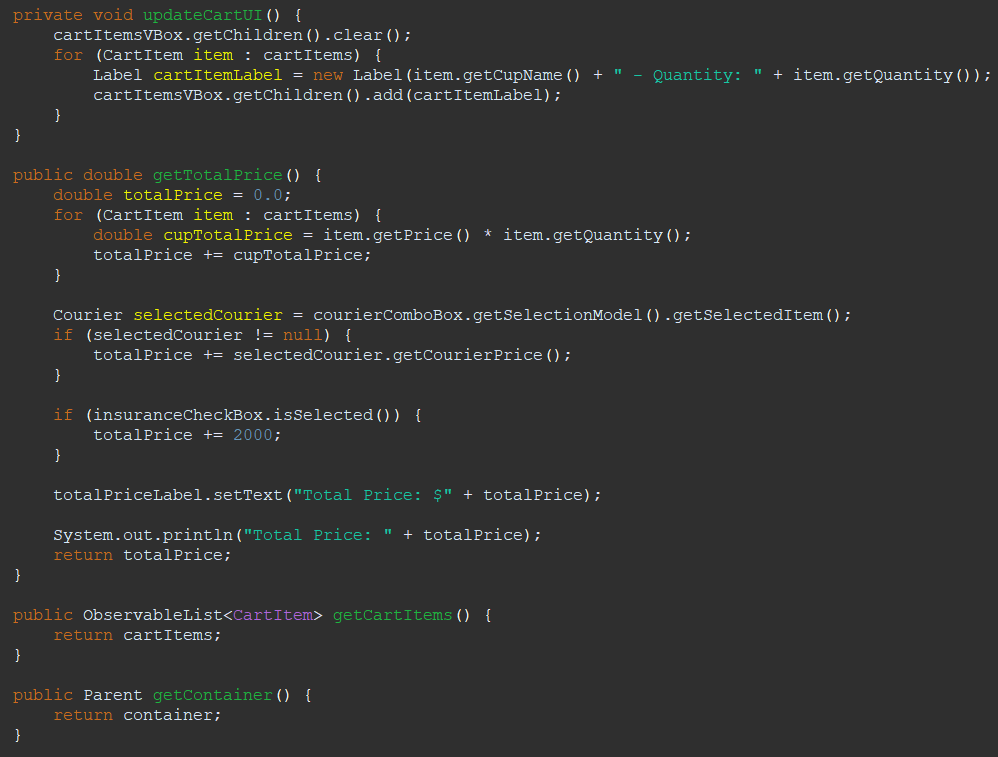


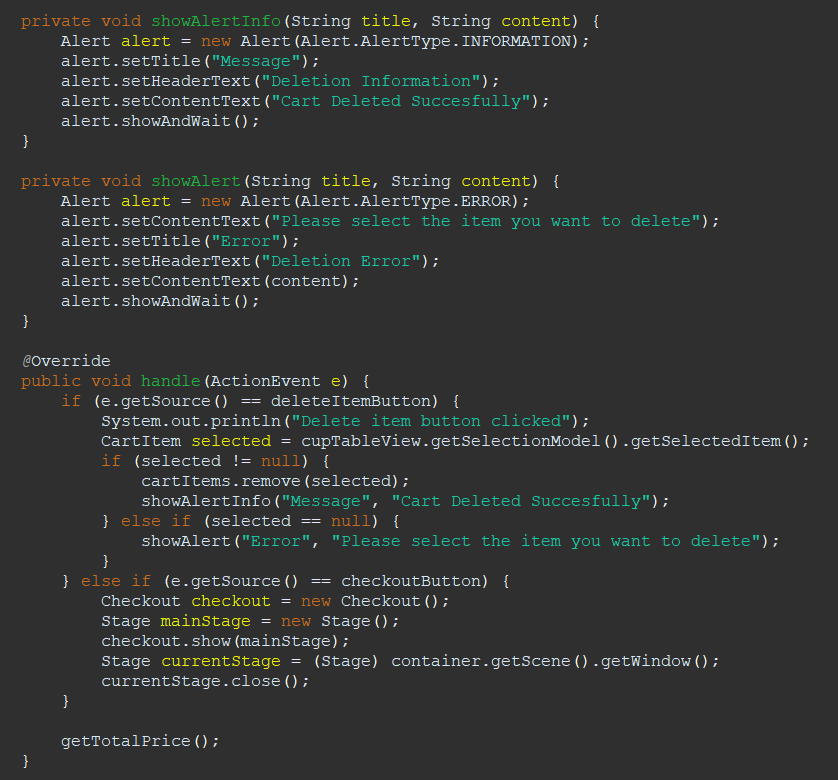












Method

* Menggunakan Vbox, gridpane, borderpane, tableview, combobox, checkbox, dll. Lalu kami melakukan inisiasi untuk semua componentnya. Melakukan beberapa customisasi pada text. Memasukkan beberapa component ke dalam VBox.
* redirectToHomePage → method untuk kembali ke halaman home
* redirectToMainPage → method untuk kembali halaman main
* getData → method untuk memanggil database dari XAMPP dengan menggunakan query dan connect. Dan juga membuat arraylist untuk menampung data dari database tersebut
* courierComboBoxRefresh → untuk menampung observable list dan menampilkan combo box.
* courierPrice → method untuk menampilkan courier price berdasarkan courier yang deselect oleh user
* addCartItem → method untuk menambah cart item, mengupdate UI, dan juga menampung total price
* updateCartItemQuantity → method untuk mengupdate kuantitas cart item
* updateCartUI → method untuk mengupdate UI pada cart
* getTotalPrice → Method untuk menentukan total price bergantung pada courier yang dipilih dan apakah pengguna menggunakan insurance.
* ObservableList<CartItem> getCartItems → untuk bisa diakses diluar class
* getContainer → Mendapatkan borderpane dari halaman main
* showAlertInfo → Method untuk menampung template alert yang menunjukkan keberhasilan (information)
* showAlert → Method untuk menampung template alert yang menunjukkan error
* e.getSource() == deleteItemButton … → Action event dimana ketika deleteItemButton diklik, jika ada yang terselect, maka cart item akan di remove. Jika tidak ada yang terselect, maka akan tampilkan alert error.
* e.getSource() == checkoutButton … → Action event dimana ketika checkoutButton diklik akan membawa ke halaman checkout

7. Class CartItem



Kami membuat class CartItem, dengan tujuan untuk menampung data-data yang diperlukan seperti cupName, price, quantity, dan total price.Kami juga melakukan inisialisasi dan memberi getter dan setter untuk mengambil nilai dan men-set nilai dari tiap variabel dalam class CartItem.

8. Class Courier



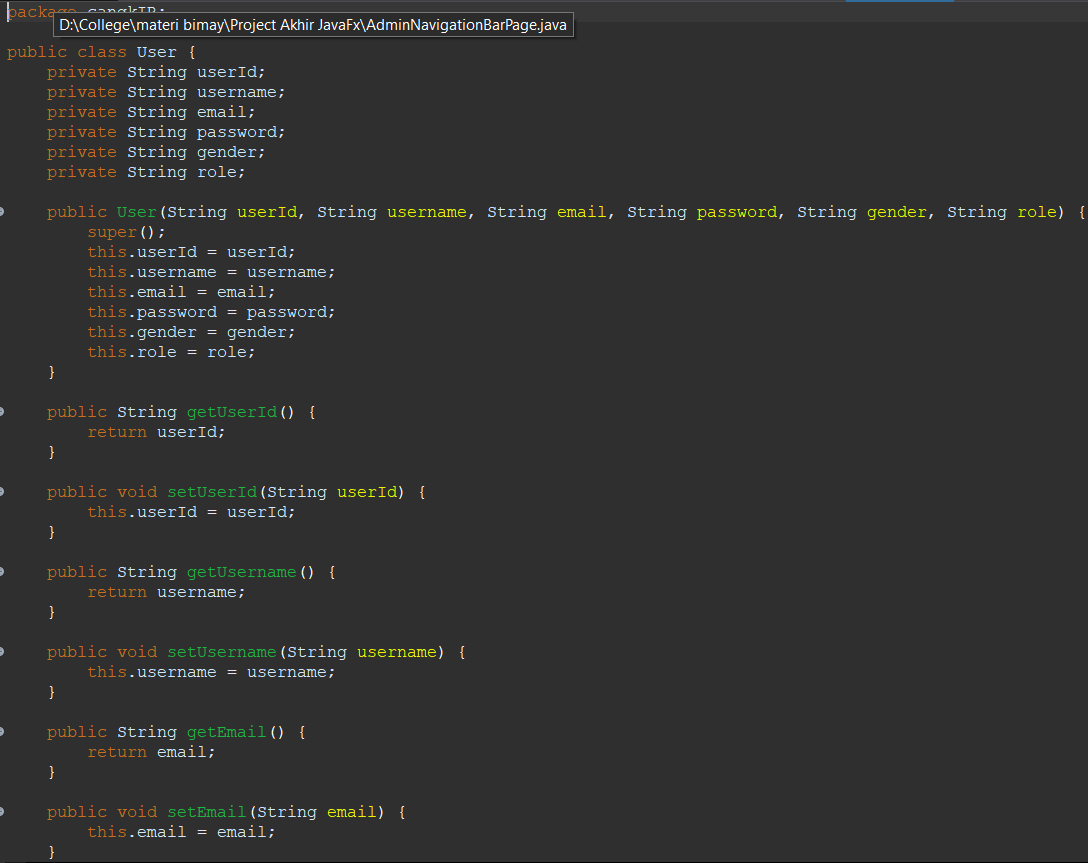
Pada class Courier, kami membuat class ini untuk menampung data-data yang akan kami berikan ke Courier. Dalam class ini kami melakukan inisialisasi untuk courierID, courierName, dan courierPrice. Kami juga melakukan getter setter untuk tiap variabel dalam class Courier ini.

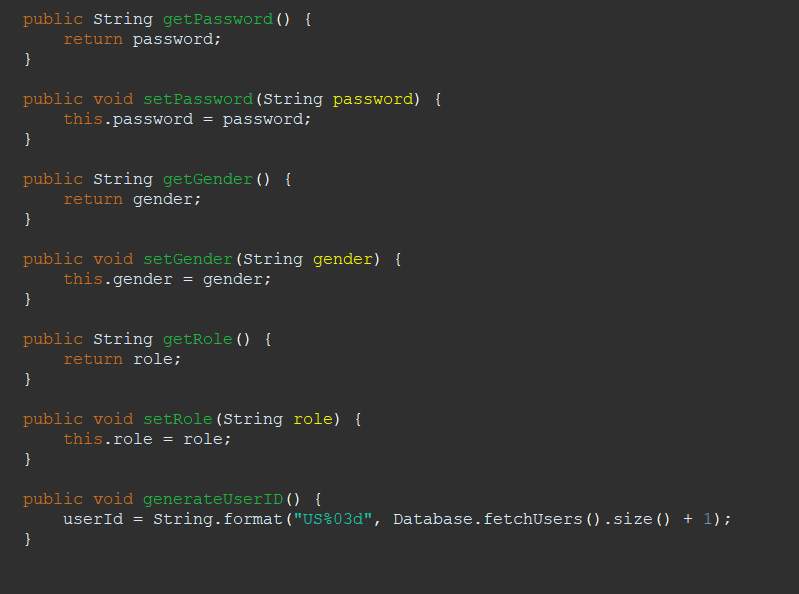
9. Class Cup



Kami membuat class Cup, dengan tujuan untuk menampung data-data yang diperlukan seperti cupID, cupName, dan price dari cup. Kami juga melakukan inisialisasi dan memberi getter dan setter untuk mengambil nilai dan men-set nilai dari tiap variabel dalam class CartItem.

10. Class user



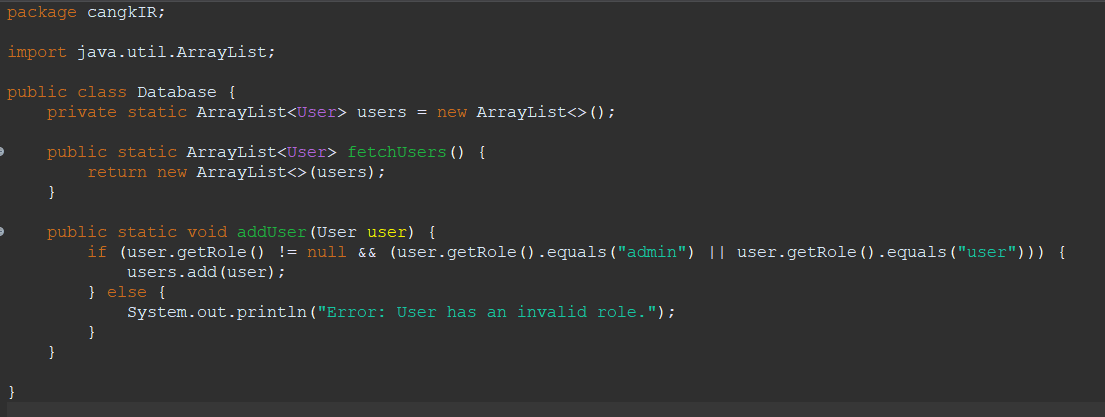


Class ini kami buat sebagai representasi dari entitas user. Class ini menampung data dari informasi user seperti ID user, Username, Email, Password, Gender, dan Role user. Kami juga melakukan inisialisasi dan memberi getter dan setter untuk mengambil nilai dan men-set nilai dari tiap variabel dalam class User..

Dengan adanya class ini, kami dapat mengelola dan menyimpan data user secara terstruktur dalam sistem kami.

Metode generateUserID() dalam kelas ini memiliki fungsi penting untuk menghasilkan ID User baru secara otomatis dengan format "USXXX", di mana "XXX" adalah angka yang disesuaikan agar setiap ID unik. Hal ini memudahkan kami dalam pemberian identifikasi unik kepada setiap user yang terdaftar dalam sistem kami. Dengan adanya variabel role dalam kelas ini, kami dapat membedakan antara user biasa (user) dan administrator (admin) dalam sistem.

11. Class Database

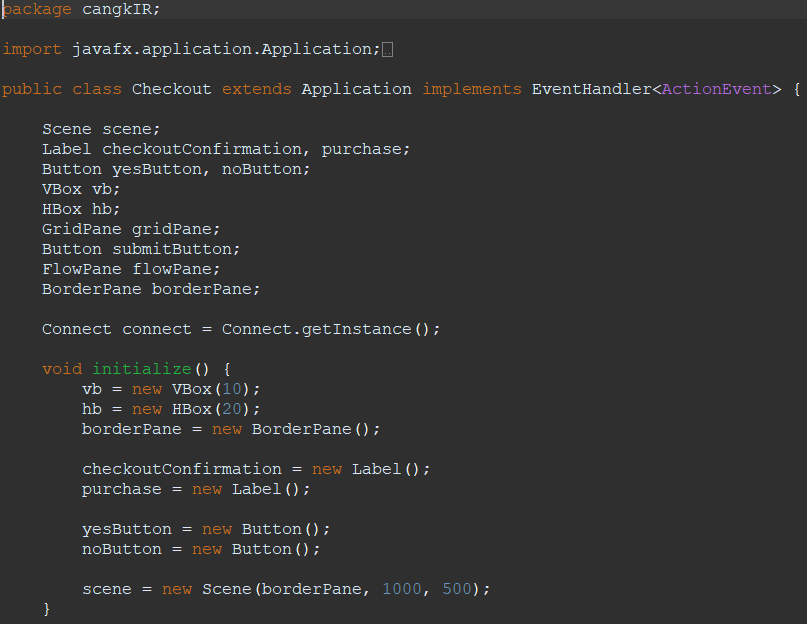


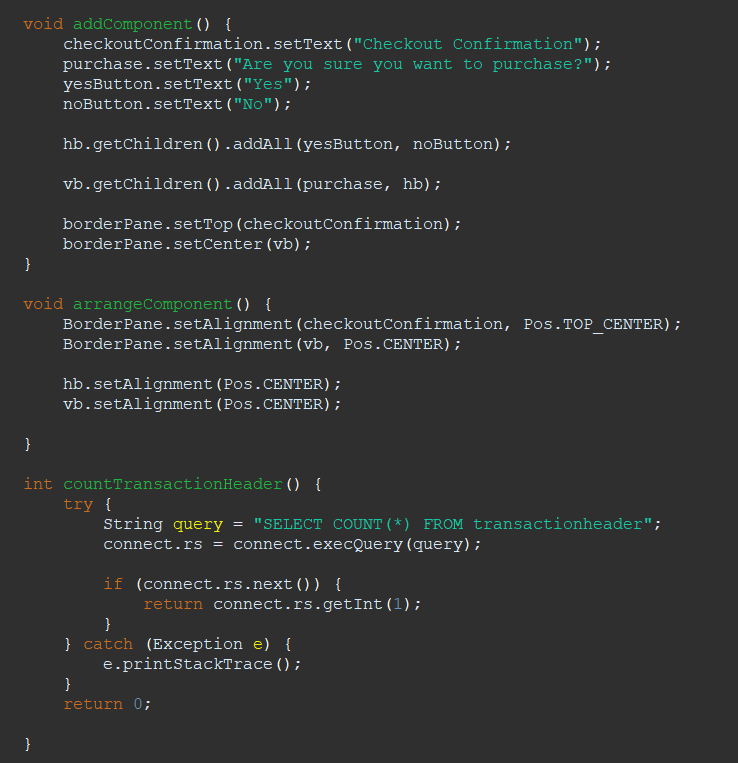
Kode dalam kelas Database kami dirancang untuk menyediakan data user dalam aplikasi. Variabel static users berfungsi sebagai penyimpanan untuk daftar user, dan metode-metode yang disediakan memungkinkan akses, penambahan, dan pemeriksaan keunikan informasi user seperti username dan alamat email.

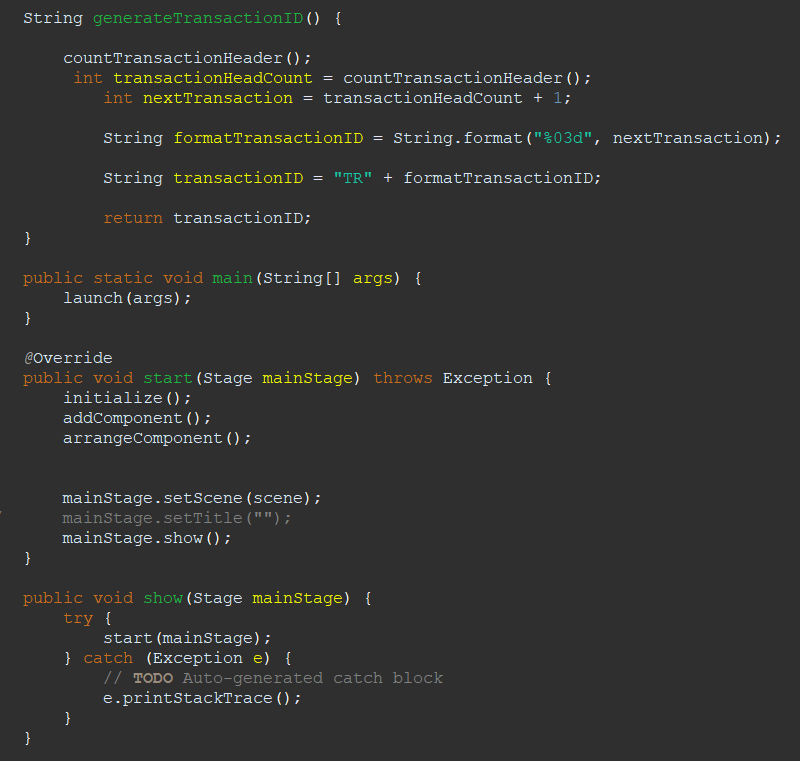
Method fetchUsers mengembalikan salinan ArrayList dari user yang telah kami simpan, memastikan bahwa daftar user asli tetap tidak dapat diubah secara langsung dari luar kelas.

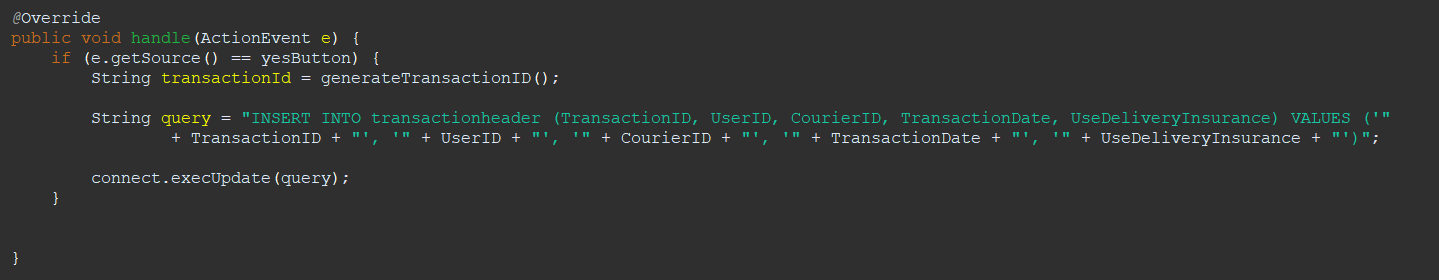
Method addUser bertujuan untuk menambahkan user ke dalam daftar. Sebelum melakukan penambahan, kami melakukan pemeriksaan terhadap validitas peran user (role), memastikan bahwa peran tersebut adalah "admin" atau "user". Jika peran tidak valid, kami mencetak pesan kesalahan.

12. Checkout



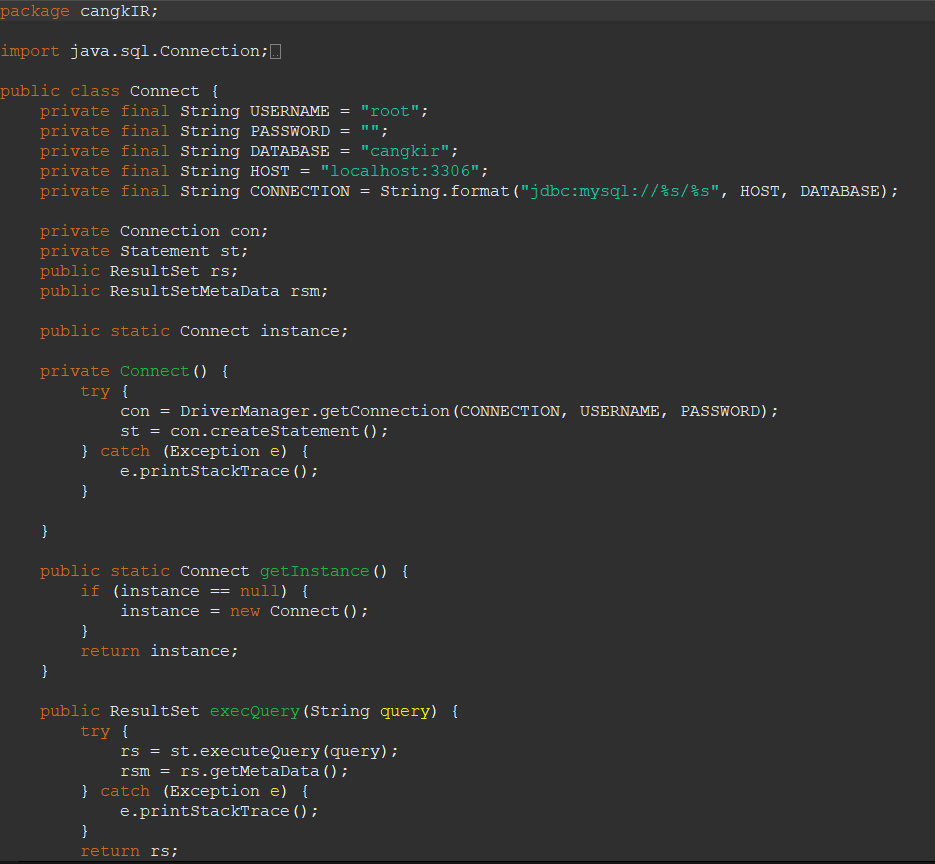


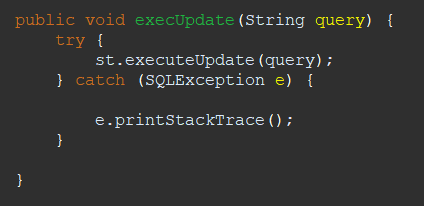




Method

* Menggunakan VBox, HBox, dan Borderpane. Pertama masukkan button ke dalam Hbox, kemudian Hbox dimasukkan ke dalam Vbox. Setelah itu Borderpane untuk menampung checkouTConfirmation dan juga Vbox dan mengatur posisinya.
* addComponent → Method untuk menambahkan component seperti checkoutConfirmation, button, dll
* arrangeComponent → Method untuk mengatur posisi component
* countTransactionHeader → Method untuk menghitung jumlah transaction header dengan menggunakan query COUNT
* generateTransactionID → Method untuk generate TransactionId
* e.getSource() == yesButton → Action event dimana ketika button yes dipencet, maka akan membuat transaction header dan transaction detail dan memasukkan informasi baru ke dalam database

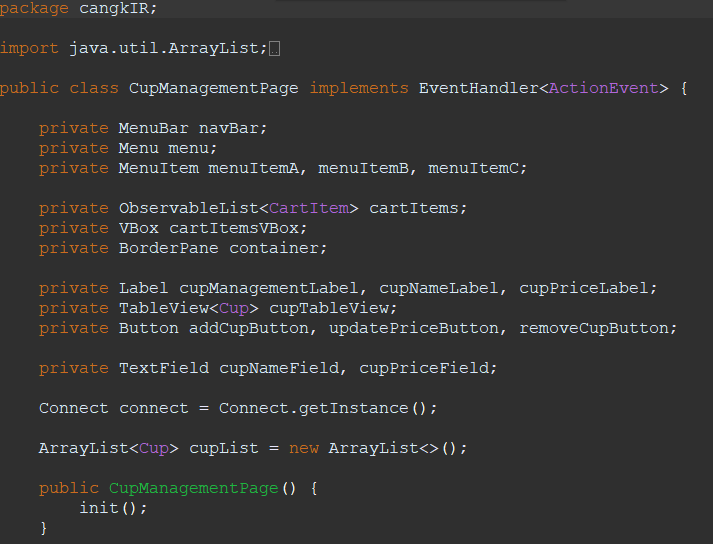
13. Connect

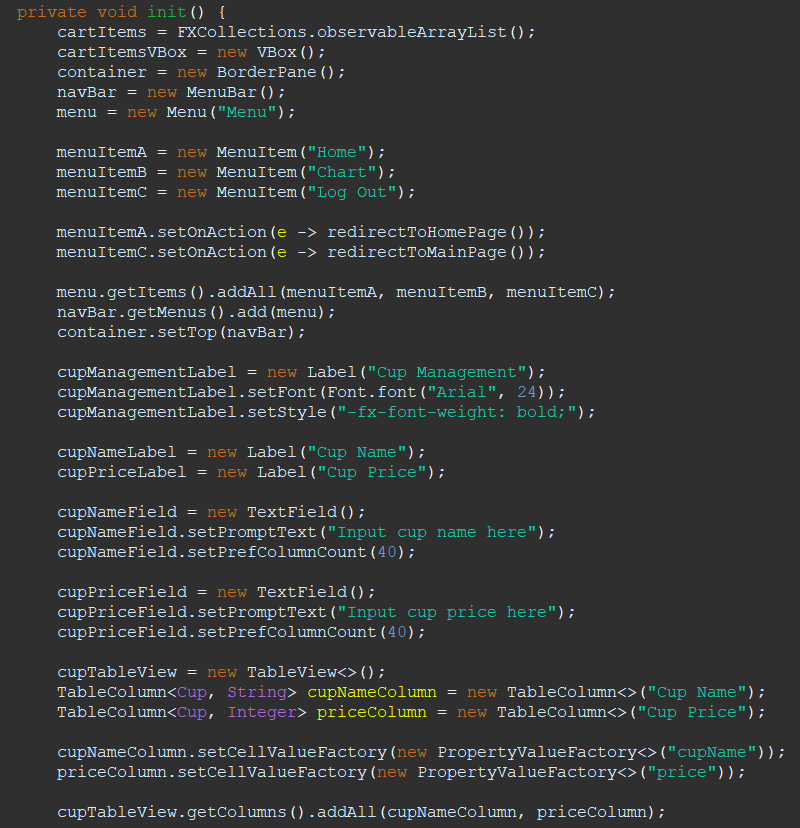


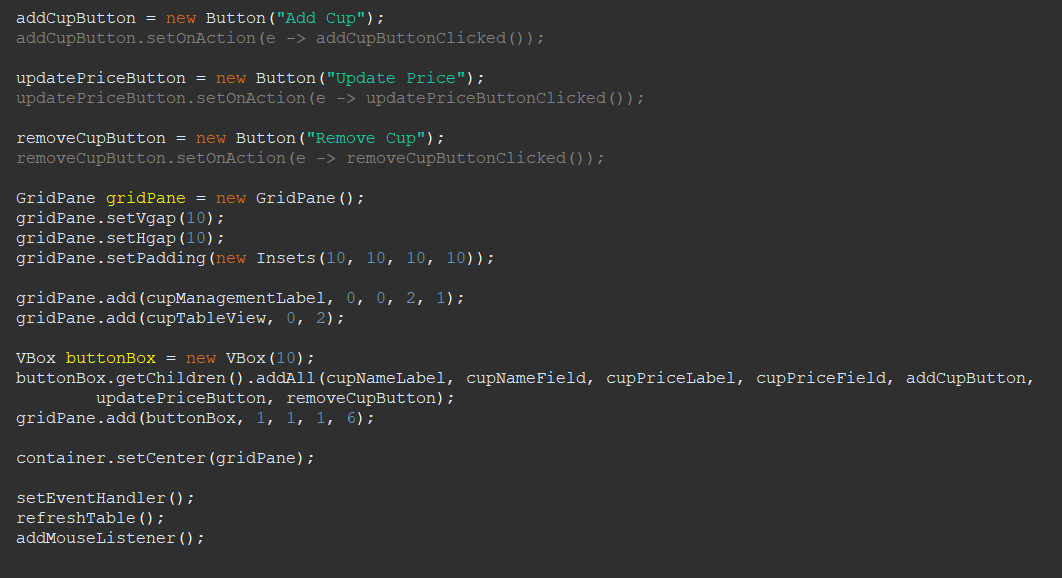
Kami membuat class ini dengan tujuan untuk menghubungkan dengan data di SQL, Variabel yang dibutuhkan untuk divalidasi adalah Username dan Password. Lalu kami juga membuat variabel Database yang di connect sesuai dengan nama database.

Untuk Connection, statement, result kami import melalui java.sql. Method connect dibuat untuk menghubungkan variabel yang sudah diimpor dari java.sql di method Connect.Lalu kami juga membuat class connect menjadi singleton agar tidak ada duplicate object connect. GetInstance berfungsi untuk mengambil data dari database untuk dilakukan validasi. execQuery berfungsi untuk melakukan execute Queries.Lalu execUpdate untuk melakukan execute update Queries.

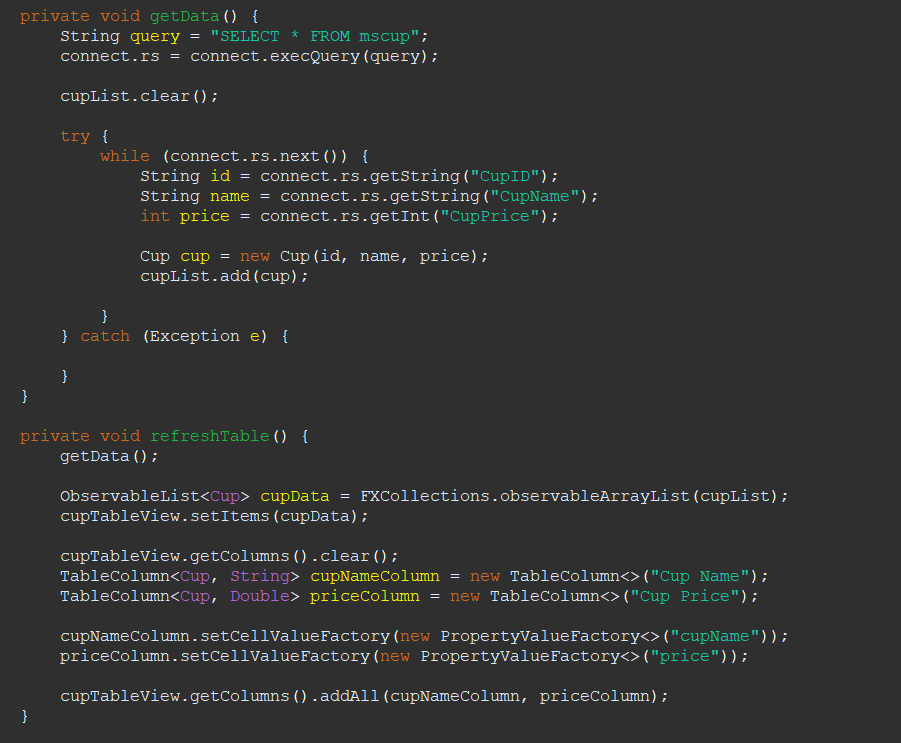
14. Cup Management



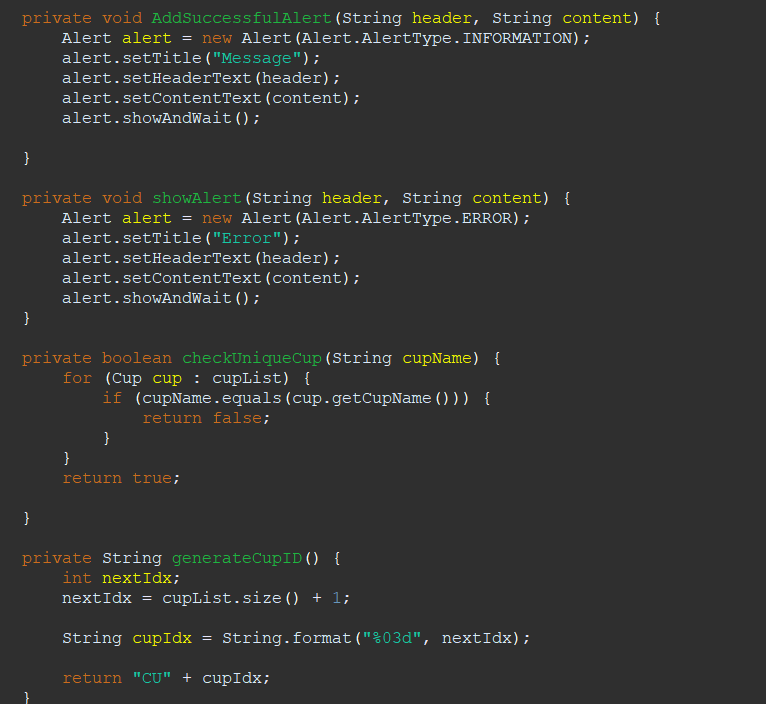


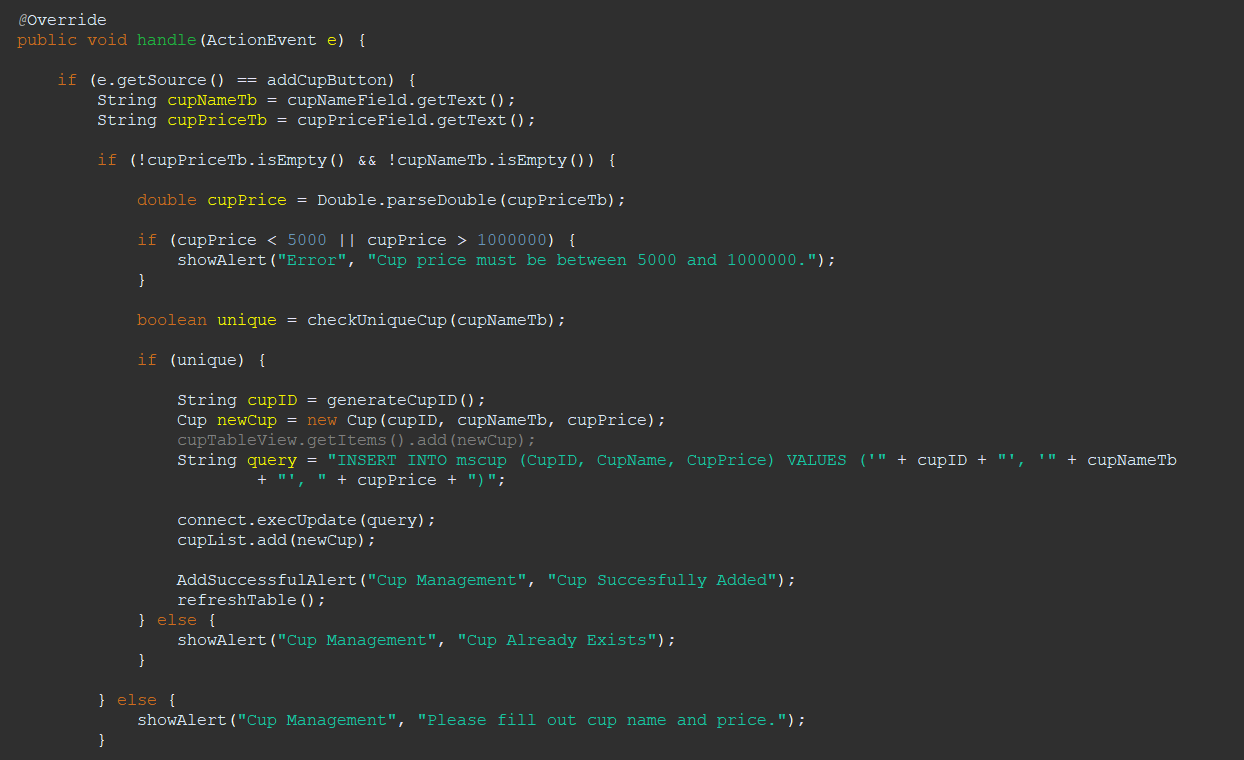


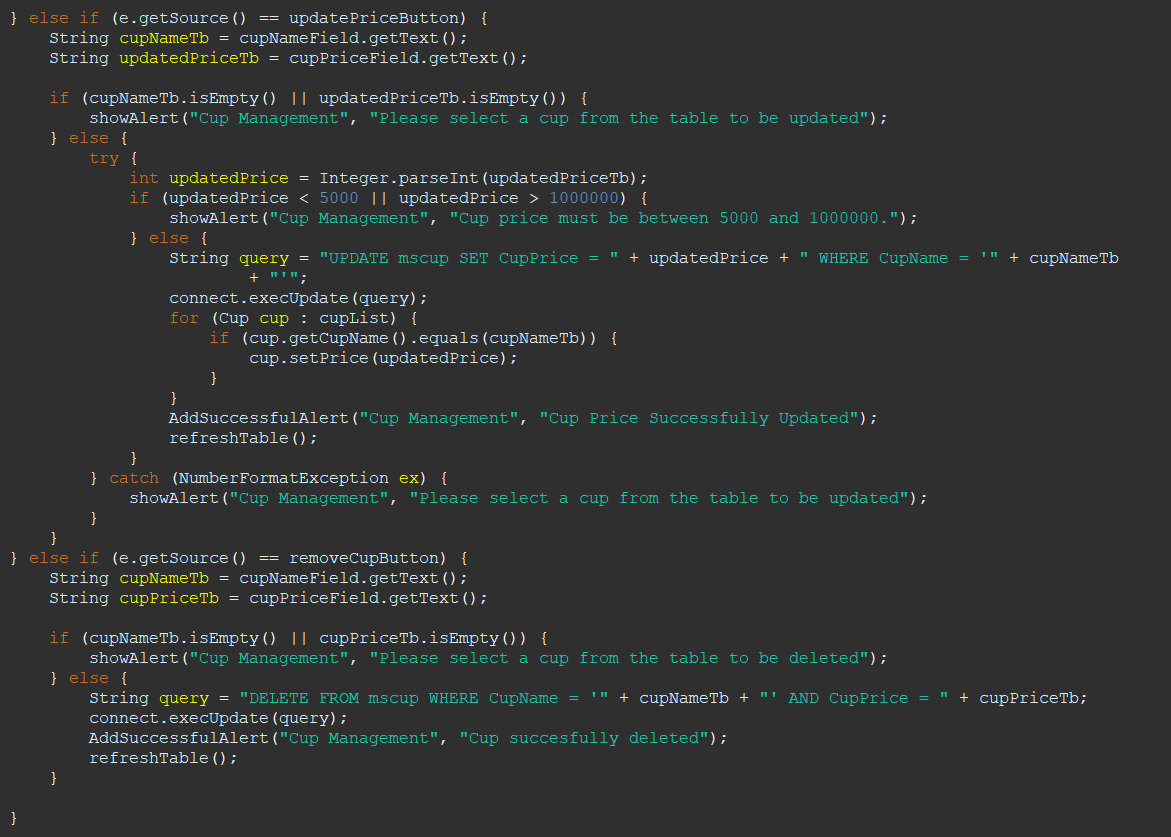












Method

* setEventHandler → Method untuk men set action agar jika button diklik dapat memberikan respons
* redirectToHomePage → Method untuk kembali ke halaman home
* redirectToMainPage → Method untuk kembali ke halaman Main
* getData → Method untuk mengambil data dari sql / XAMPP dan sekaligus membuat arraylist yang menampung data-data cup dari database
* refreshTable → Method untuk menampung observableList dan sekaligus menampilkan tabel pada UI
* addCartItem → method untuk menambah cart item, mengupdate UI, dan juga menampung total price
* updateCartItemQuantity → method untuk mengupdate kuantitas cart item
* updateCartUI → method untuk mengupdate UI pada cart
* AddSuccessfulAlert → Method untuk menampung template alert yang menunjukkan keberhasilan (information)
* showAlert → Method untuk menampung template alert yang menunjukkan error
* checkUniqueCup → Method untuk mengecek apakah nama cupnya unik atau tidak
* generateCupID → Method untuk mengenerate cupID dengan men concat “CU” dengan index yang sudah diformat
* addMouseListener → Method untuk menselect cup dan menampilkan informasi seperti nama cup dan harga cup berdasarkan apa yang deselect pengguna
* e.getSource() == addCupButton → Untuk menambahkan cup baru ke dalam database dengan menginput nama cup, harga cup dan memencet tombol add cup. Dengan beberapa validasi seperti nama cup harus unik, dll.
* e.getSource() == updatePriceButton → Untuk mengupdate harga dari cup yang sudah ada pada database
* e.getSource() == removeCupButton → Untuk meremove cup yang diinginkan oleh admin
* Reference

Ø Textbook:: Y. Daniel Liang (2020). Introduction to Java

Programming and Data Structures, Comprehensive

Version. New York: Pearson

Ø <https://stackoverflow.com/questions/29349197/show-alert-javafx>

Ø <https://www.w3schools.blog/java-singleton-design-pattern>

* Group Member
* <2602108880> – <1st Indra Cetta Daniswara>
* <2602109492> – <2nd Andreas Leonardo Hadylaw>
* <2602107985> – <3rd Nickson Yuna>
* <2602115394> – <4th Kelvin Malaka>