Міністерство освіти та науки

Прикарпатський національний університет

Фізико-технічний факультет

Кафедра комп’ютерної інженерії та електроніки

Лабораторна робота № 10

З курсу “Захист інформації у комп’ютерних мережах та системах”

Виконав студент групи КІ-41

Воробій Віталій

Івано-Франківськ 2020

5 варіант

Програмна реалізація мовою *Java:*

Створив клас, для запуску JavaFX застосунку.

import javafx.application.Application;  
import javafx.fxml.FXMLLoader;  
import javafx.scene.Parent;  
import javafx.scene.Scene;  
import javafx.stage.Stage;  
  
import java.io.IOException;  
import java.net.URL;  
  
*/\*\*  
 \* JavaFX App  
 \*/*public class App extends Application {  
  
 @Override  
 public void start(Stage primaryStage) throws IOException {

// Завантаження графічного інтерфейсу  
 FXMLLoader loader = new FXMLLoader();  
 URL xmlUrl = getClass().getResource("/gui.fxml");  
 loader.setLocation(xmlUrl);  
 Parent root = loader.load();  
  
 primaryStage.setScene(new Scene(root));  
 primaryStage.setTitle("Лабораторна робота № 10");  
 primaryStage.show();  
  
 }  
  
 public static void main(String[] args) {  
 *launch*();  
 }  
  
}

Створив абстракцію CAPTCHA

*/\*\*  
 \* Клас що описує CAPTCHA  
 \*/*public class CaptchaResult {  
 private final String question;  
 private final String rightAnswer;  
  
 public CaptchaResult(String question, String rightAnswer) {  
 this.question = question;  
 this.rightAnswer = rightAnswer;  
 }  
  
 protected String getRightAnswer() {  
 return rightAnswer;  
 }  
  
 public String getQuestion() {  
 return question;  
 }  
}

Створив абстракцію генерування CAPTCHA

public interface CaptchaService {  
 CaptchaResult generateCaptcha();  
 boolean match(CaptchaResult captchaResult, String entered);  
}

Створив реалізацію генерування CAPTCHA

import java.util.Random;  
  
public class AdditionCaptchaService implements CaptchaService {  
 private final Random r = new Random();  
  
 @Override  
 public CaptchaResult generateCaptcha() {  
  
 *// Генеруємо два випадкові числа* int firstNumber = r.nextInt(20);  
 int secondNumber = r.nextInt(20);  
  
 *// Обчислюємо суму як вірну відповідь* final int sum = firstNumber + secondNumber;  
  
 *// Формуємо запитання і вірну відповідь* final String question = String.*format*("%d + %d = ?", firstNumber, secondNumber);  
 final String rightAnswer = Integer.*toString*(sum);  
  
 return new CaptchaResult(question, rightAnswer);  
 }  
  
 @Override  
 public boolean match(CaptchaResult captchaResult, String entered) {  
 return captchaResult.getRightAnswer().equals(entered.trim());  
 }  
}

Створив абстракцію функції трансформації

public interface TransformationFunction<T> {  
 T compute(T t);  
}

І її реалізацію

public class DoubleTransformationFunction implements TransformationFunction<Integer> {  
 @Override  
 public Integer compute(Integer number) {  
 return number \* 2;  
 }  
}

Абстракція отримання доступу до ресурсу

import org.example.solution.Response;  
  
public interface UserAuthorizationService {  
 Response tryToGetAccess(String userName, Integer input, Integer entered);  
}

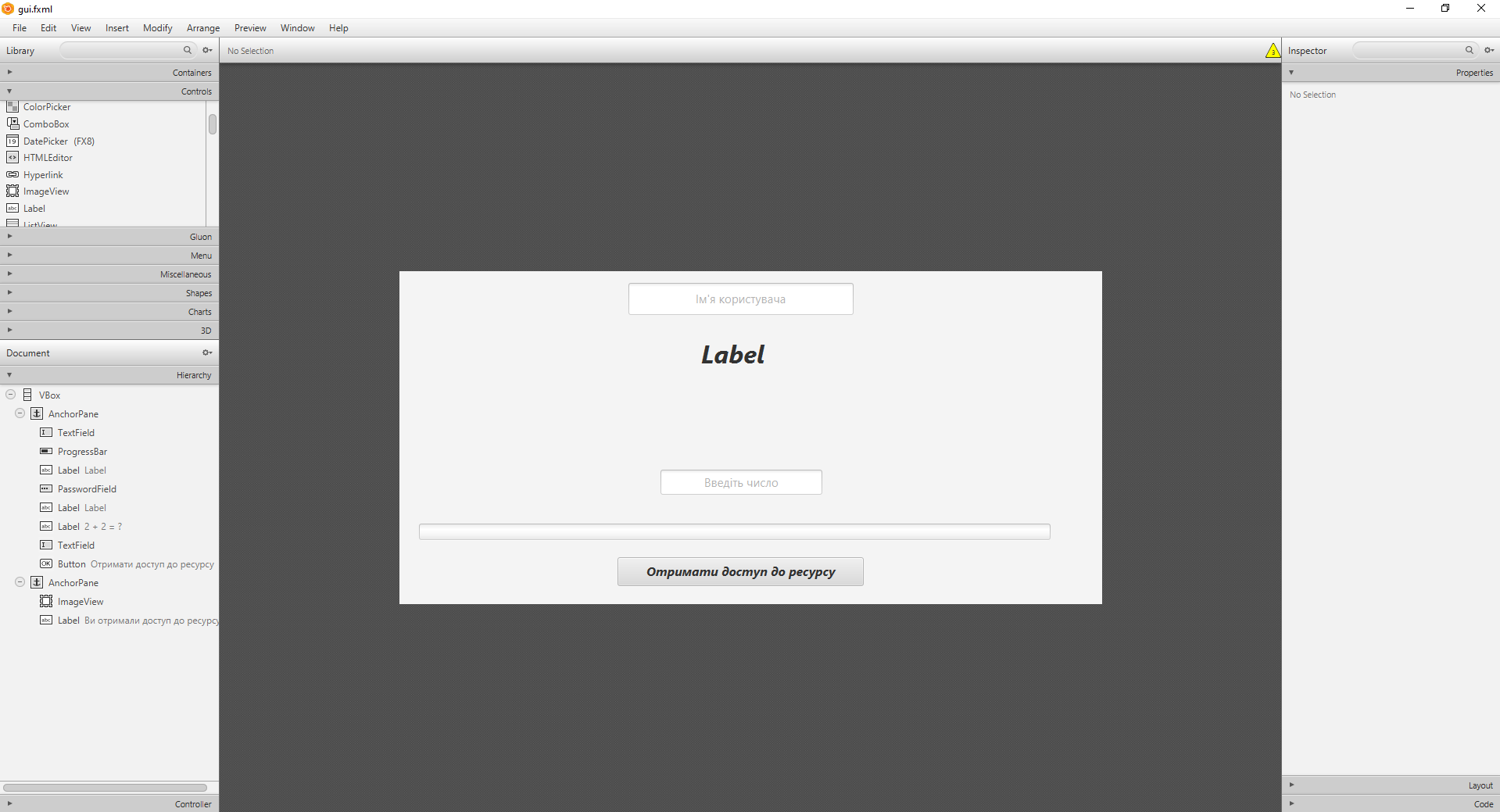
І її реалізація

import org.example.solution.Response;  
  
import java.util.HashMap;  
import java.util.Map;  
  
public class UserAuthorizationServiceImpl implements UserAuthorizationService {  
 private static final int *TOO\_MANY\_TRIES* = 3;  
  
 *// Кеш користувачів* private final Map<String, Integer> userAuthorizationAttempts = new HashMap<>();  
  
 *// Функція перетворення* private final TransformationFunction<Integer> transformationFunction = new DoubleTransformationFunction();  
  
 @Override  
 public Response tryToGetAccess(String userName, Integer input, Integer entered) {  
  
 *// Поточна кількість спроб* final int currentNumberOfAttempts =  
 userAuthorizationAttempts.getOrDefault(userName, 0);  
  
 *// Якщо число введене вірно - запит пройшов успішно* if (transformationFunction.compute(input).equals(entered)) {  
 *// Додатково видаляємо користувча з кешу* userAuthorizationAttempts.remove(userName);  
 return Response.*SUCCESS*;  
 }  
  
 final int incrementedNumberOfAttempts = currentNumberOfAttempts + 1;  
  
 *// Збільшуємо число спроб вводу пароля користувача на одиницю в кеші* userAuthorizationAttempts.put(userName, incrementedNumberOfAttempts);  
  
 *// Якщо число спроб вводу пароля кратна TOO\_MANY\_TRIES (3) ми просимо ввести CAPTCHA* if (incrementedNumberOfAttempts % *TOO\_MANY\_TRIES* == 0) {  
 return Response.*WRONG\_TOO\_MANY\_TRIES*;  
 }  
  
 *// В іншому випадку просто просимо користувача ввести пароль ще раз* return Response.*WRONG*;  
 }  
}

Перечислення можливих варіантів відповіді:

public enum Response {  
 *SUCCESS*,  
 *WRONG*,  
 *WRONG\_TOO\_MANY\_TRIES*}

Створив графічний інтерфейс у програмі Scene Builder



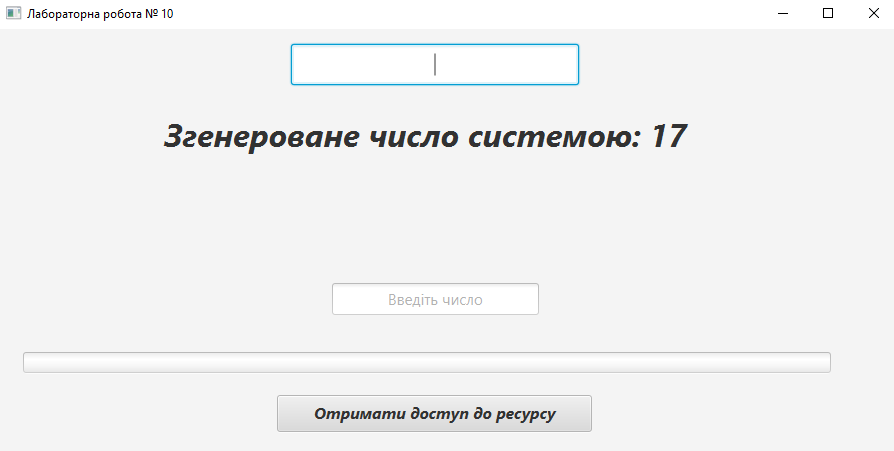
Створив клас-контроллер для вищенаведеного інтерфейсу

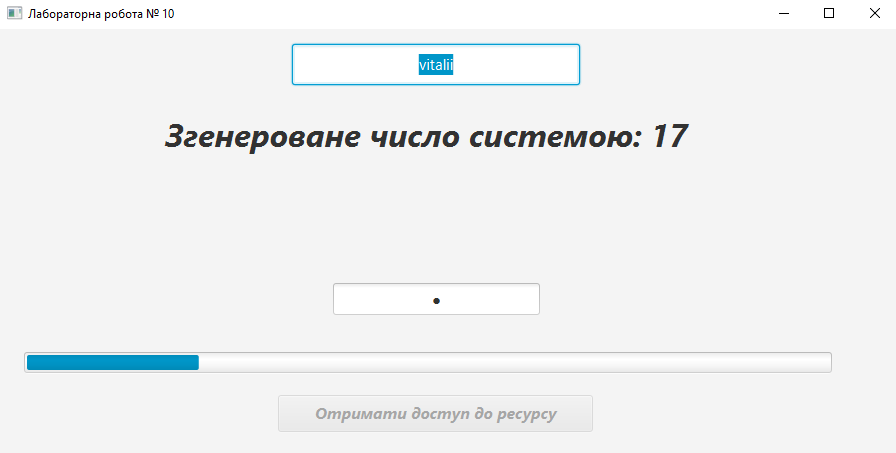
import javafx.scene.layout.AnchorPane;  
import org.example.solution.Response;  
import org.example.solution.service.\*;  
  
import java.util.Random;  
  
public class Controller {  
  
 private static final Random *r* = new Random();  
  
 @FXML  
 public AnchorPane authorization\_pane;  
  
 @FXML  
 public AnchorPane secret\_pane;  
  
 @FXML  
 public TextField username\_login;  
  
 @FXML  
 public ProgressBar loading\_indicator;  
  
 @FXML  
 public Label info\_label;

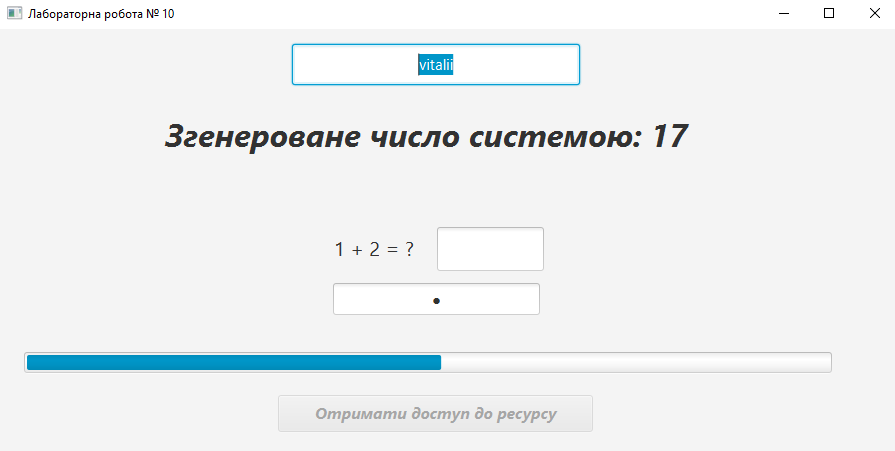
@FXML  
 public Button try\_get\_resource\_button;  
  
 @FXML  
 public PasswordField provided\_num\_input;  
  
 @FXML  
 public Label captcha\_question;  
  
 @FXML  
 public TextField captcha\_answer;  
  
 @FXML  
 public Label error;  
  
 @FXML  
 public ImageView secret\_image;  
  
 *// Сервіс каптчі* private final CaptchaService captchaService = new AdditionCaptchaService();  
  
 *// Сервіс авторизації* private final UserAuthorizationService userAuthorizationService = new UserAuthorizationServiceImpl();  
  
 *// Затримка вводу пароля* private static final int *DELAY\_WRONG* = 4000; *// У мілісекундах  
  
 // Поточне згенероване число* private Integer currentGeneratedValue = *r*.nextInt(50);  
  
 private CaptchaResult captchaResult;  
  
 @FXML  
 public void handle\_try\_get\_resource(ActionEvent event) {  
  
 final String userName = username\_login.getText();  
  
 *// Перевірка вводу ім'я користувача* if (isInputEmpty(userName)) {  
 error.setVisible(true);  
 error.setText("Ім'я користувача пусте");  
 return;  
 } else {  
 error.setVisible(false);  
 }  
  
 *// Перевірка вводу катчі (за наявності)* if (captcha\_question.isVisible()) {  
  
 if (!captchaService.match(captchaResult, captcha\_answer.getText())) {  
 error.setVisible(true);  
 error.setText("Капчу введено не вірно");  
 } else {  
 captcha\_question.setVisible(false);  
 captcha\_answer.setVisible(false);  
 return;  
 }  
 }  
  
 *// Перевірка вводу числа* final int enteredNum;  
 try {  
 enteredNum = Integer.*parseInt*(provided\_num\_input.getText().trim());  
 }  
 catch (NumberFormatException e) {  
 error.setVisible(true);  
 error.setText("Не вірний формат числа");  
 return;  
 }  
  
 *// Спроба отримати доступ* final Response response =  
 userAuthorizationService.tryToGetAccess(userName, currentGeneratedValue, enteredNum);  
  
 switch (response) {  
 case *WRONG*: {  
 makeDelay();  
 break;  
 }  
 case *SUCCESS*: {  
 disableCaptcha();  
 openResource();  
 break;  
 }  
 case *WRONG\_TOO\_MANY\_TRIES*: {  
 initCaptcha();  
 makeDelay();  
 }  
 }  
  
 }

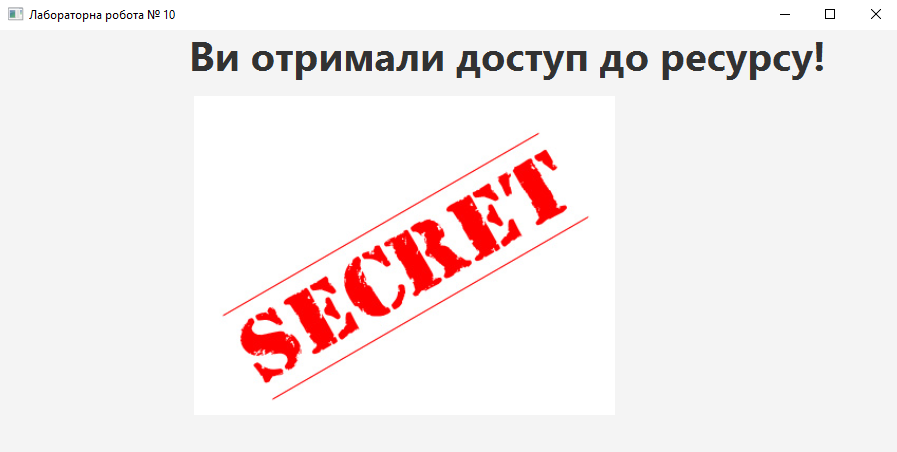
// Відкриття ресурсу   
 private void openResource() {  
 authorization\_pane.setVisible(false);  
 secret\_pane.setVisible(true);  
  
 *// Встановлення зображення* Image file = new Image(getClass().getResourceAsStream("/secret.jpg"));  
 secret\_image.setImage(file);  
 }  
  
 */\*\*  
 \* Приховання каптчі  
 \*/* private void disableCaptcha() {  
 captcha\_question.setVisible(false);  
 captcha\_answer.setVisible(false);  
 }  
  
 */\*\*  
 \* Ініціалізація каптчі  
 \*/* private void initCaptcha() {  
 captchaResult = captchaService.generateCaptcha();  
 captcha\_question.setVisible(true);  
 captcha\_question.setText(captchaResult.getQuestion());  
 captcha\_answer.setVisible(true);  
 }  
  
 @FXML  
 public void initialize() {  
 info\_label.setText("Згенероване число системою: " + currentGeneratedValue);  
 }  
  
 */\*\*  
 \* Функція створення затримки вводу  
 \*/* private void makeDelay() {  
  
 *// Робимо кнопку недоступну* try\_get\_resource\_button.setDisable(true);  
  
 *// В окремому потоці запускаємо програму що змінює індикатор прогрес бару* new Thread(() -> {  
 final int numberOfIterations = 100;  
 loading\_indicator.setVisible(true);  
 for (int i = 1; i <= numberOfIterations; i++) {  
 try {  
 Thread.*sleep*(*DELAY\_WRONG* / numberOfIterations);  
 } catch (InterruptedException ignored) {}  
 loading\_indicator.setProgress((double) i / numberOfIterations);  
 }  
 loading\_indicator.setVisible(false);  
 }).start();  
  
 *// В іншому потоці запускаємо програму, що перешкоджає натисканні на кнопку деякий час* new Thread(() -> {  
 try {  
 Thread.*sleep*(*DELAY\_WRONG*);  
 } catch (InterruptedException ignored) {}  
 try\_get\_resource\_button.setDisable(false);  
 }).start();  
 }  
  
 */\*\*  
 \* Функція для перевірки значення на пустоту  
 \* @param value Вхідне значення  
 \*/* private boolean isInputEmpty(String value) {  
 return value.trim().equals("");  
 }  
  
}

Результат виконання:









Висновок: на цій лабораторній роботі я успішно реалізував метод рукостискання.