|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.04.01 Информатика и вычислительная техника**

МАГИСТЕРСКАЯ ПРОГРАММА **09.04.01/12 Интеллектуальный анализ больших**

**данных в системах поддержки принятия решений.**

**Отчет**

**по лабораторной работе № 2**

**Вариант № 9**

**Название:** реализация простейшего генератора паролей

**Дисциплина:** Информационная безопасность автоматизированных систем

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-31М |  |  | И.С. Марчук |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | Д.А. Миков |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2024

**Цель:** изучение технологии аутентификации пользователя на основе пароля.

**Задание:** разработать программу, представляющую собой форму доступа к определённым информационным ресурсам на основе пароля.

• В качестве информационного ресурса использовать любой файл или приложение.

• Доступ к ресурсу должен быть разрешен только санкционированным пользователям. Для этого в программе должны храниться имена пользователей и их пароли. При попытке доступа пользователя к ресурсу проверяется наличие его идентификатора (имени) в системе и соответствие введенного пароля паролю, который хранится в системе.

• В системе должна храниться следующая информация о пользователе: ID или имя пользователя, пароль, ФИО, дата рождения, место рождения (город), номер телефона.

• Пользователь должен иметь возможность поменять пароль.

Условия варианта:

* Номер варианта - 9;
* Длина пароля (количество символов) - 7;
* Используемые символы – цифры;
* Дополнительные средства защиты - применение метода аутентификации на основе одноразовых паролей (к первой цифре каждого следующего пароля прибавляется 1).

**Ход работы**

Я реализовал на языке Kotlin при помощи стандартных компонентов библиотеки Swing. Интерфейс программы позволяет добавить в систему пользователей и разграничить доступ к личным файлам пользователей. Исходный код программы представлен в листинге 1.

Листинг программы 1 – Программа генерации пароля

package org.example

import java.awt.Component

import java.awt.Dimension

import java.awt.event.WindowEvent

import javax.swing.\*

import javax.swing.text.AttributeSet

import javax.swing.text.PlainDocument

data class User(

// Имя пользователя

val userLogin: String,

// Пароль

var userPassword: String,

// Фамилия

val userSecondName: String,

// Имя

val userName: String,

// Отчество

val userThirdName: String,

// Дата рождения

val userBirthday: String,

// Место рождения

val userBirthPlace: String,

// Телефон

val userPhone: String,

// тайная строка

var secretString: String

)

// пользователи

val usersDatabase: MutableList<User> = mutableListOf(

User(

"admin",

"1234567",

"Марчук",

"Иван",

"Сергеевич",

"25.12.2000",

"Москва",

"8(800)555-35-35",

"Секретные данные администратора"

)

)

// текущий пользователь

var currentUser: User? = null

fun main() {

println("Запуск!")

authorisationFrame()

}

// форма входа

fun authorisationFrame() {

// Очищаем текущего пользователя

currentUser = null

// создание окна

val authorisationFrame = JFrame("Вход")

authorisationFrame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE)

authorisationFrame.setSize(300, 200)

authorisationFrame.setLocationRelativeTo(null)

val panel = JPanel()

panel.setLayout(BoxLayout(panel, BoxLayout.Y\_AXIS))

authorisationFrame.add(panel)

// Имя пользователя

panel.add(JLabel("Имя пользователя").apply {

setAlignmentX(Component.CENTER\_ALIGNMENT)

})

val userNameInput = JTextField().apply {

maximumSize = Dimension(200, 20)

document = JTextFiledLimit(20, true)

setAlignmentX(Component.CENTER\_ALIGNMENT)

}

panel.add(userNameInput)

// Пароль

panel.add(JLabel("Пароль").apply {

setAlignmentX(Component.CENTER\_ALIGNMENT)

})

val passInput = panel.add(JPasswordField().apply {

maximumSize = Dimension(200, 20)

document = JTextFiledLimit(7, false)

setAlignmentX(Component.CENTER\_ALIGNMENT)

}) as JPasswordField

// Статус

val statusText = panel.add(JLabel("Status").apply {

setAlignmentX(Component.CENTER\_ALIGNMENT)

}) as JLabel

// кнопка войти

val enterButton = JButton("Войти")

enterButton.setAlignmentX(Component.CENTER\_ALIGNMENT)

panel.add(enterButton)

enterButton.addActionListener {

currentUser = null

usersDatabase.forEach { userIt ->

if (

userIt.userLogin == userNameInput.text.toString().trim() &&

userIt.userPassword == String(passInput.password).trim()

) currentUser = userIt

}

if (currentUser == null)

statusText.text = "Ошибка введенных данных"

else {

statusText.text = "Авторизация успешна"

// закрываем это окно

authorisationFrame.dispatchEvent(WindowEvent(authorisationFrame, WindowEvent.WINDOW\_CLOSING))

// окно пользователя

userFrame()

}

}

// кнопка регистрация

val registerButton = JButton("Регистрация")

registerButton.setAlignmentX(Component.CENTER\_ALIGNMENT)

panel.add(registerButton)

registerButton.addActionListener {

// statusText.text = "hello"

registerFrame()

}

// переотрисовка

authorisationFrame.isVisible = true

}

// форма регистрация пользователя

fun registerFrame() {

// создание окна

val frame = JFrame("Регистрация")

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE)

frame.setSize(400, 400)

frame.setLocationRelativeTo(null)

// Имя пользователя

frame.add(JLabel("Имя пользователя").apply {

setBounds(50, 25, 150, 20)

})

val userLoginInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 25, 150, 20)

}) as JTextField

// Пароль

frame.add(JLabel("Пароль").apply {

setBounds(50, 50, 100, 20)

})

val userPasswordInput = frame.add(JPasswordField().apply {

document = JTextFiledLimit(7, false)

setBounds(170, 50, 150, 20)

}) as JPasswordField

// Фамилия

frame.add(JLabel("Фамилия").apply {

setBounds(50, 100, 100, 20)

})

val userSecondNameInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 100, 150, 20)

}) as JTextField

// Имя

frame.add(JLabel("Имя").apply {

setBounds(50, 125, 100, 20)

})

val userNameInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 125, 150, 20)

}) as JTextField

// Отчество

frame.add(JLabel("Отчество").apply {

setBounds(50, 150, 100, 20)

})

val userThirdNameInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 150, 150, 20)

}) as JTextField

// Дата рождения

frame.add(JLabel("Дата рождения").apply {

setBounds(50, 200, 100, 20)

})

val userBirthdayInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 200, 150, 20)

}) as JTextField

// Место рождения

frame.add(JLabel("Место рождения").apply {

setBounds(50, 225, 100, 20)

})

val userBirthPlaceInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 225, 150, 20)

}) as JTextField

// Телефон

frame.add(JLabel("Телефон").apply {

setBounds(50, 250, 100, 20)

})

val userPhoneInput = frame.add(JTextField().apply {

document = JTextFiledLimit(20, true)

setBounds(170, 250, 150, 20)

}) as JTextField

// кнопка регистрация

frame.add(JButton("Сохранить").apply {

setBounds(50, 300, 270, 20)

addActionListener {

// сохраняем пользователя

usersDatabase.add(

User(

userLogin = userLoginInput.text.toString().trim(),

userPassword = String(userPasswordInput.password).trim(),

userSecondName = userSecondNameInput.text.toString(),

userName = userNameInput.text.toString(),

userThirdName = userThirdNameInput.text.toString(),

userBirthday = userBirthdayInput.text.toString(),

userBirthPlace = userBirthPlaceInput.text.toString(),

userPhone = userPhoneInput.text.toString(),

secretString = "Введите ваши данные сюда"

)

)

// закрываем это окно

frame.dispatchEvent(WindowEvent(frame, WindowEvent.WINDOW\_CLOSING))

//lable.text = userNameInput.getText()

}

})

frame.layout = null

// переотрисовка

frame.isVisible = true

}

// форма смены пароля

fun changePasswordForm() {

// создание окна

val frame = JFrame("Смена пароля")

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE)

frame.setSize(400, 400)

frame.setLocationRelativeTo(null)

// Старый пароль

frame.add(JLabel("Старый пароль").apply {

setBounds(50, 25, 100, 20)

})

val userOldPasswordInput = frame.add(JPasswordField().apply {

document = JTextFiledLimit(7, false)

setBounds(170, 25, 150, 20)

}) as JPasswordField

// Новый пароль

frame.add(JLabel("Новый пароль").apply {

setBounds(50, 50, 100, 20)

})

val userNewPasswordInput = frame.add(JPasswordField().apply {

document = JTextFiledLimit(7, false)

setBounds(170, 50, 150, 20)

}) as JPasswordField

// Новый пароль повтор

frame.add(JLabel("Повторите пароль").apply {

setBounds(50, 75, 100, 20)

})

val userNewPassword2Input = frame.add(JPasswordField().apply {

document = JTextFiledLimit(7, false)

setBounds(170, 75, 150, 20)

}) as JPasswordField

// Статус

val statusOutput = frame.add(JLabel("").apply {

setBounds(50, 225, 200, 40)

}) as JLabel

// кнопка смены

frame.add(JButton("Сменить пароль").apply {

setBounds(50, 275, 270, 20)

addActionListener {

if (currentUser!!.userPassword != String(userOldPasswordInput.password)) {

statusOutput.text = "Старый пароль не совпадает"

} else {

val newPass1 = String(userNewPasswordInput.password)

val newPass2 = String(userNewPassword2Input.password)

if (newPass1.length != 7) {

statusOutput.text = "Длина пароля должна быть 7 символов"

} else {

if (newPass1 != newPass2) {

statusOutput.text = "Пароли должны совпадать"

} else {

// Меняем пароль

currentUser?.userPassword = newPass1

// закрываем это окно

frame.dispatchEvent(WindowEvent(frame, WindowEvent.WINDOW\_CLOSING))

}

}

}

}

})

frame.layout = null

// переотрисовка

frame.isVisible = true

}

// форма личного кабинета пользователя

fun userFrame() {

if (currentUser == null) return

// создание окна

val frame = JFrame("Кабинет пользователя")

frame.setDefaultCloseOperation(JFrame.DISPOSE\_ON\_CLOSE)

frame.setSize(400, 400)

frame.setLocationRelativeTo(null)

frame.add(JLabel(

"<html>Параметры пользователя:<br> " +

"${currentUser!!.userLogin}<br>" +

"${currentUser!!.userSecondName} ${currentUser!!.userName} ${currentUser!!.userThirdName}<br>" +

"${currentUser!!.userPhone}<br>" +

"${currentUser!!.userBirthday} ${currentUser!!.userBirthPlace}" +

"</html>"

).apply {

setBounds(50, 25, 200, 80)

})

// Секретная строка

frame.add(JLabel("Файл пользователя").apply {

setBounds(50, 125, 250, 20)

})

val secretStringInput = JTextArea().apply {

document = JTextFiledLimit(300, true)

setBounds(50, 150, 250, 40)

text = currentUser?.secretString

}

frame.add(secretStringInput)

// Статус

val statusOutput = frame.add(JLabel().apply {

setBounds(50, 225, 270, 20)

}) as JLabel

// кнопка сохранения изменений

frame.add(JButton("Сохранить изменения").apply {

setBounds(50, 200, 270, 20)

addActionListener {

currentUser?.secretString = secretStringInput.text.toString()

statusOutput.text = "Сохранено ${secretStringInput.text}"

}

})

// кнопка смены

frame.add(JButton("Сменить пароль").apply {

setBounds(50, 275, 270, 20)

addActionListener {

changePasswordForm()

}

})

// кнопка выйти из учетки

frame.add(JButton("Выйти из учетной записи").apply {

setBounds(50, 300, 270, 20)

addActionListener {

// закрываем сессию

currentUser = null

// закрываем это окно

frame.dispatchEvent(WindowEvent(frame, WindowEvent.WINDOW\_CLOSING))

// аутентификация

authorisationFrame()

}

})

frame.layout = null

// переотрисовка

frame.isVisible = true

}

// вспомогательный метод

class JTextFiledLimit(

private val limit: Int,

private val isLetters: Boolean

) : PlainDocument() {

override fun insertString(offs: Int, str: String?, a: AttributeSet?) {

if (str == null)

return

// проверка на символы

val newInput: StringBuilder

if (!isLetters) {

newInput = StringBuilder()

str.forEach {

if (it.code in 48..58) newInput.append(it)

}

} else {

newInput = StringBuilder(str)

}

// если не достигли максимальной длины строки

if (length + newInput.length <= limit)

super.insertString(offs, newInput.toString(), a)

}

}

**Пример работы программы**

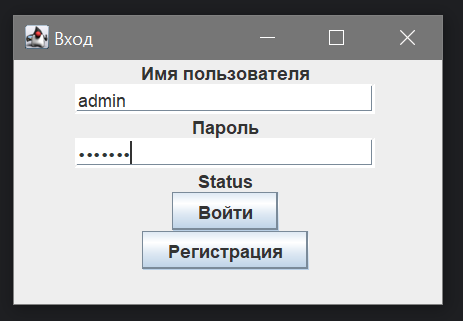


Рисунок 1 – Окно аутентификации пользователя

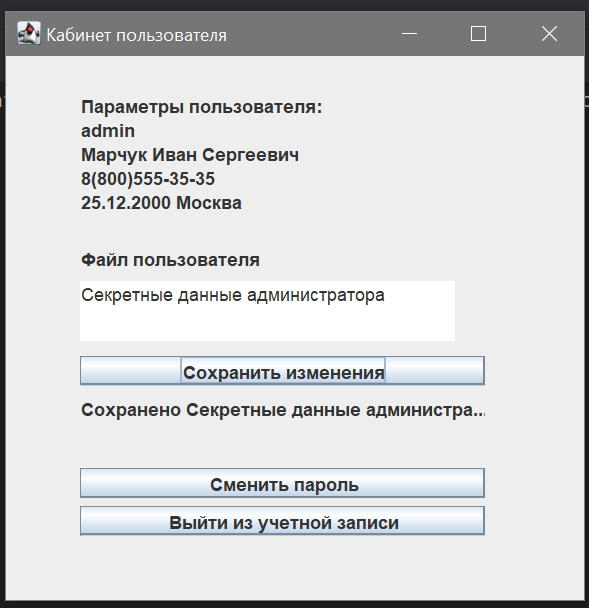


Рисунок 2 – Личный кабинет пользователя с редактируемым файлом

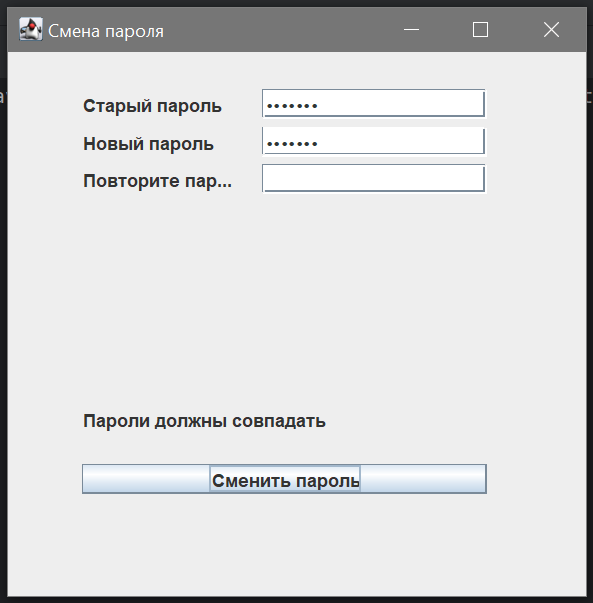


Рисунок 3 – Окно смены пароля

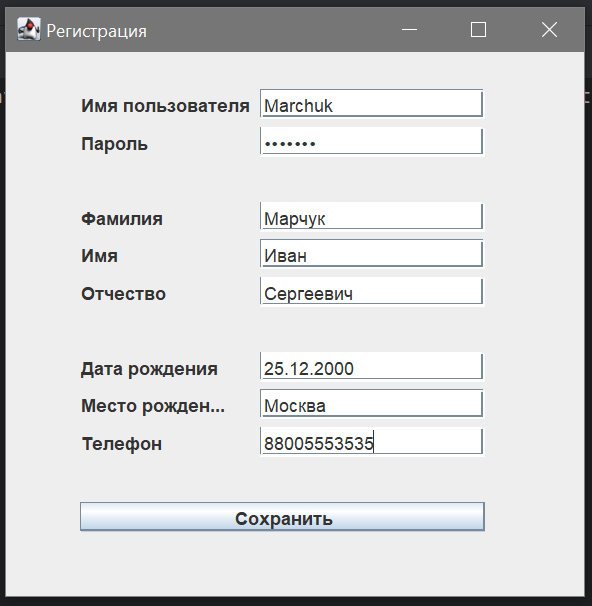


Рисунок 4 – Окно регистрации пользователя

**Контрольные вопросы**

1. Дать определение аутентификации. Привести примеры.

Аутентификация - процедура проверки подлинности заявленного пользователя, процесса или устройства, например:

— проверка подлинности пользователя путём сравнения введённого им пароля с паролем, сохранённым в базе данных пользователей;

— подтверждение подлинности электронного письма путём проверки цифровой подписи письма по открытому ключу отправителя;

— проверка контрольной суммы файла на соответствие сумме, заявленной автором этого файла.

2. Дать определение идентификации в информационных системах.

Идентификация в информационных системах — присвоение субъектам и объектам идентификатора и/или сравнение идентификатора с перечнем присвоенных идентификаторов.

3. Дать определение авторизации пользователя.

Авторизация - процедура предоставления субъекту определенных полномочий и ресурсов в данной системе. Иными словами, авторизация устанавливает сферу действия субъекта и доступные ему ресурсы.

С процедурами аутентификации и авторизации тесно связана процедура администрирования действий пользователя.

4. Дать определение пароля.

Пароль — это то, что знает пользователь и что также знает другой участник взаимодействия. Для взаимной аутентификации участников взаимодействия может быть организован обмен паролями между ними.

**Вывод**

Я разработал программу, представляющую собой форму доступа к определённым информационным ресурсам на основе пароля. А также изучил технологии аутентификации пользователя на основе пароля.