use std::io;

struct Account {

    saldo: f64,

    pin: String,

}

impl Account {

    fn new(initial\_saldo: f64, pin: String) -> Self {

        Account {

            saldo: initial\_saldo,

            pin,

        }

    }

    fn cek\_saldo(&self) {

        println!("Saldo Anda saat ini adalah: ${:.2}", self.saldo);

    }

    fn deposit(&mut self, amount: f64) {

        if amount > 0.0 {

            self.saldo += amount;

            println!("Berhasil disetorkan ${:.2}.", amount);

        }

    }

    fn withdraw(&mut self, amount: f64) {

        if amount > self.saldo {

            println!("Saldo tidak mencukupi.");

        } else if amount > 0.0 {

            self.saldo -= amount;

            println!("Berhasil menarik ${:.2}.", amount);

        }

    }

    fn validate\_pin(&self, input\_pin: &str) -> bool {

        self.pin == input\_pin

    }

}

fn main() {

    let mut account = Account::new(1000.0, "1234".to\_string());

    println!("Selamat datang di ATM kami.");

    println!("Silakan masukkan PIN Anda:");

    let mut input\_pin = String::new();

    io::stdin().read\_line(&mut input\_pin).expect("Gagal membaca baris");

    let input\_pin = input\_pin.trim();

    if !account.validate\_pin(input\_pin) {

        println!("PIN salah. Akses ditolak.");

        return;

    }

    println!("PIN benar. Selamat datang!");

    loop {

        println!("\nATM Menu:");

        println!("1. Cek saldo");

        println!("2. Setor uang");

        println!("3. Tarik uang");

        println!("4. Exit");

        println!("Masukkan pilihan Anda:");

        let mut choice = String::new();

        io::stdin().read\_line(&mut choice).expect("Gagal membaca baris");

        let choice = choice.trim();

        match choice {

            "1" => {

                account.cek\_saldo();

            }

            "2" => {

                println!("Masukkan jumlah setoran:");

                let mut amount = String::new();

                io::stdin().read\_line(&mut amount).expect("Gagal membaca baris");

                if let Ok(amount) = amount.trim().parse::<f64>() {

                    account.deposit(amount);

                } else {

                    println!("Input tidak valid.");

                }

            }

            "3" => {

                println!("Masukkan jumlah penarikan:");

                let mut amount = String::new();

                io::stdin().read\_line(&mut amount).expect("Gagal membaca baris");

                if let Ok(amount) = amount.trim().parse::<f64>() {

                    account.withdraw(amount);

                } else {

                    println!("Input tidak valid.");

                }

            }

            "4" => {

                println!("Terima kasih telah menggunakan ATM kami. Selamat tinggal!");

                break;

            }

            \_ => println!("Pilihan tidak valid, coba lagi."),

        }

    }

}

LANGKAH-LANGKAH RUN

TERMINAL => CARGO NEW TUGAS => CD TUGAS => CARGO RUN