PONTIFÍCIA UNIVERSIDADE CATÓLICA DO PARANÁ

PERFORMANCE EM SISTEMAS CIBERFISICOS

Dupla autenticação em um sistema WEB usando usuário/senha e RFID

[1]Eduardo Lourenço da Silva

[2]Hector Vieira Saldivar

[3]Theo Rocha Otto

[4]Vinicius Veiga de Sant'Ana

[5]Valdinei Jose Saugo

1. Especificação do sistema

O sistema desenvolvido é uma aplicação web com backend em PHP, estruturado com base no paradigma de programação orientada a objetos e na arquitetura MVC (Model-View-Controller). Foi implementado um sistema de rotas (routes) para o gerenciamento eficiente das requisições e utilização de DAOs (Data Access Objects) para o acesso ao banco de dados, garantindo uma separação clara entre as regras de negócio e a persistência de dados. O banco de dados utilizado foi o MySQL.

Além disso, o sistema conta com um mecanismo de autenticação de dois fatores (2FA), implementado por meio de um módulo ESP32-C3 SuperMini integrado a um leitor PN532 (NFC/RFID). No processo de login, o usuário deve informar sua senha e, também, autenticar-se utilizando um cartão RFID ou NFC, o que aumenta significativamente a segurança do acesso ao sistema.

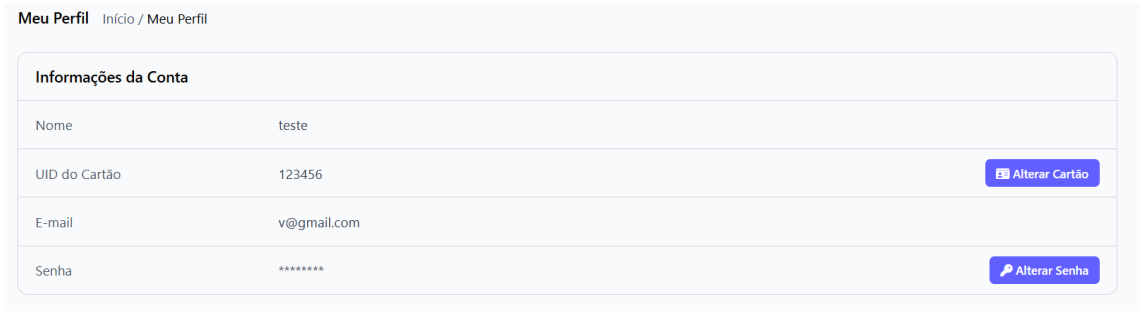
Login:
A primeira tela ao acessar o sistema é a de Login, o qual o usuário pode logar utilizando e-mail e senha, ou se cadastrar caso não possua uma conta.

Cadastro:
Ao se cadastrar, será solicitado o nome, e-mail e senha. Assim que o usuário clica em cadastrar é solicitado que o usuário encoste o cartão no leitor para salvar o ID dele no banco (imagem na próxima página):

[1] Aluno Graduando em Engenharia de software na PUCPR
[2] Aluno Graduando em Engenharia de software na PUCPR
[3] Aluno Graduando em Engenharia de software na PUCPR
[4] Aluno Graduando em Engenharia de software na PUCPR
[5] Professor na disciplina de Performance em Sistema Ciberfisicos na PUCPR

Neste momento o sistema está aguardando a aproximação do cartão no leitor PN532, assim que o cartão for encostado no leitor, será enviada uma requisição post com o número, caso não exista este banco, será criado o usuário com este cartão.

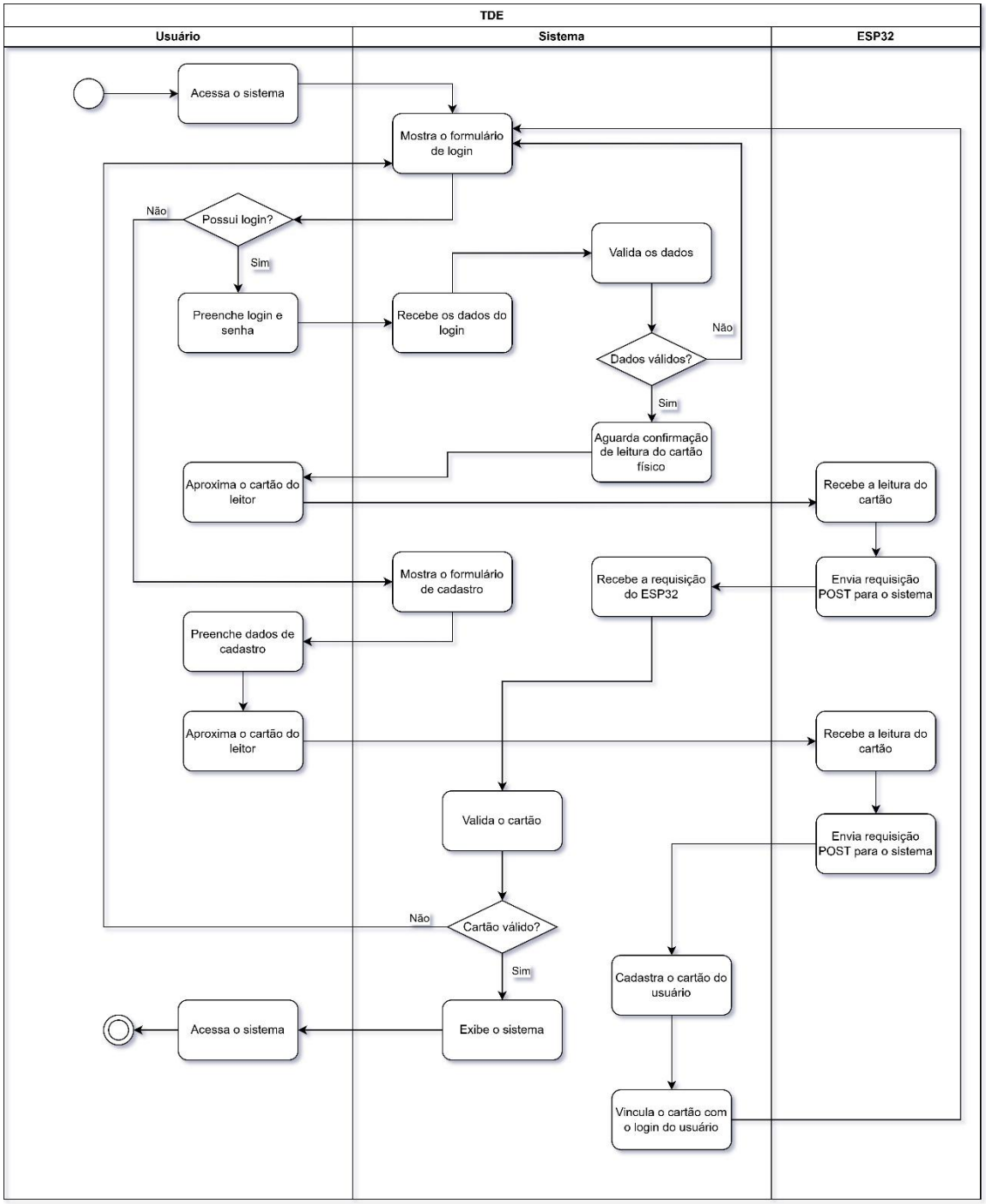
Ao acessar o sistema é exibida esta tela:



Onde é possível alterar o número do cartão e a senha.

Além disso, no sistema é possível analisar as últimas leituras do cartão do seu usuário.

2. Diagrama da arquitetura ciberfísica (fluxograma)



3. Código-fonte documentado

Todo o sistema foi feito utilizando a arquitetura de MVC e orientado a objetos, então haverão as classes de Controller, Model, Dao e View de todos os objetos abaixo:
-Cartao
-Usuario
-Leitura

As implementações relacionadas ao numero do cartão são feitas no Cartao, referente ao cadastro de validação de usuário no Usuario, e todas as leituras realizadas são

tratadas na Leitura. Esta é toda a parte WEB do projeto, o foco da documentação detalhada será no script (completo no GitHub) que acessa e valida o ESP32:

- As bibliotecas utilizadas foram:

Wire.h: Biblioteca de comunicação I2C (usada pelo leitor PN532).

Adafruit_PN532.h: Biblioteca específica para o módulo PN532 NFC RFID.

WiFi.h: Permite conexão com redes Wi-Fi no ESP32.

HTTPClient.h: Usada para realizar requisições HTTP (POST neste caso).

- Foram utilizados os pinos:

#define SDA_PIN 8: Pino “Serial Data Line” é utilizado para enviar/receber dados via I2C.

#define SCL_PIN 9: Pino “Serial Clock Line” é utilizado levar o sinal de clock, que sincroniza a cominucação .

- const char* serverUrl = "http://192.168.18.13/rfid2fa/api/leitura/cadastro"; URL do servidor para onde será enviado o UID do cartão via post.

- Adafruit_PN532 nfc(SDA_PIN, SCL_PIN);, Inicializa o PN532 usando I2C.

- A função setup() é executada uma única vez quando o ESP32 é ligado ou reiniciado. Ela tem como objetivo preparar o ambiente para o funcionamento do sistema. Primeiramente, inicia a comunicação serial com o computador usando Serial.begin(115200) para que seja possível exibir mensagens no console durante a execução. Em seguida, o ESP32 tenta se conectar à rede Wi-Fi com o SSID e senha fornecidos. Enquanto a conexão não for estabelecida, o código entra em um laço de espera imprimindo pontos para indicar o andamento. Após conectar, ele imprime o IP local do ESP32 na rede.

Logo depois, a comunicação com o módulo NFC é iniciada pela função nfc.begin(), que configura internamente os pinos e o protocolo de comunicação I2C ou SPI, conforme definido anteriormente. Em seguida, a função nfc.getFirmwareVersion() é chamada para verificar se o módulo está devidamente conectado e funcionando; ela retorna a versão do firmware caso o módulo seja reconhecido. Caso nenhum dado seja retornado (valor nulo), significa que o módulo não foi detectado corretamente, e uma mensagem de erro é exibida no monitor serial. Nesse caso, o programa entra em um loop infinito com while (1);, travando a execução para evitar que o código continue com falhas.

Se o módulo for detectado corretamente, uma mensagem de sucesso é exibida no console. Em seguida, a função nfc.SAMConfig() é chamada para configurar o módulo em modo de leitura — essa configuração prepara o dispositivo para detectar e processar cartões NFC. Por fim, é exibida a mensagem “Aproxime o cartão...” para indicar que o sistema está pronto para iniciar as leituras.

Dentro da função loop(), o código verifica continuamente se um cartão NFC foi aproximado do leitor.

Primeiro, são declaradas duas variáveis: uid[7], que armazenará o identificador único (UID) do cartão, e uidLength, que armazenará o comprimento do UID detectado.

A função nfc.readPassiveTargetID() é chamada utilizando o protocolo PN532_MIFARE_ISO14443A, responsável por detectar cartões do tipo MIFARE. Se um cartão for detectado, o código prossegue com a leitura dos dados.

Em seguida, é criada uma String chamada uid_str para armazenar o UID em formato hexadecimal. Um laço for percorre todos os bytes do UID e os converte para string, adicionando um zero à esquerda sempre que o valor for menor que 0x10, garantindo o formato correto. A string final é convertida para letras maiúsculas com uid_str.toUpperCase();.

Se o dispositivo estiver conectado ao Wi-Fi, o código cria um objeto HTTPClient para enviar uma requisição HTTP POST ao servidor, contendo o UID lido em formato JSON. O cabeçalho da requisição é configurado com o tipo de conteúdo "application/json", e o corpo é construído como {"uid_cartao":"VALOR_DO_UID"}.

Após enviar a requisição, o código verifica a resposta do servidor. Se a resposta for bem-sucedida (com código HTTP maior que 0), o conteúdo retornado é exibido no monitor serial. Em caso de erro, o código correspondente também é exibido. Ao final, http.end(); encerra a conexão com o servidor.

Se não houver conexão Wi-Fi no momento da leitura, uma mensagem de erro é impressa.

Por fim, a função delay(2000); impede que a leitura do mesmo cartão ocorra em sequência, inserindo um intervalo de 2 segundos entre leituras.

4. Relatório de desempenho e testes

Este relatório apresenta os resultados dos testes de latência, consumo de energia e eficiência da comunicação do sistema baseado no ESP32-C3 SuperMini com o módulo NFC PN532, utilizando comunicação HTTP com um servidor web.

4.1. Latência

A latência foi medida em dois momentos principais do ciclo de funcionamento:

- **Tempo de leitura e montagem do UID do cartão NFC:** Refere-se ao tempo desde a detecção do cartão até a conversão do UID em uma string hexadecimal.
  Média observada: 0,067 ms ou 67 µs
- **Tempo de envio da requisição HTTP e resposta do servidor:** Refere-se ao tempo entre o início do envio da requisição POST e a recepção da resposta do servidor.
  Média observada: 49,5 ms
- **Tempo total desde a leitura até a finalização da comunicação:** Soma dos tempos anteriores.
  Média observada: 49,567 ms

Foram realizados **10 testes** consecutivos para obtenção dessas médias.

4.2. Consumo de Energia

Não foi possível realizar medições diretas de consumo de energia durante os testes, devido à ausência de equipamentos de instrumentação adequados (como um analisador de corrente ou shunt de precisão). No entanto, com base na documentação e testes comuns com o **ESP32-C3 SuperMini**, os seguintes valores estimados foram considerados:

- **Consumo em repouso (sem cartão presente):** aproximadamente **25 mA**
- **Durante a leitura NFC:** aproximadamente **50–70 mA**
- **Durante a comunicação Wi-Fi:** picos de até **160–180 mA**

Esses valores são aproximados e podem variar conforme o ambiente, tipo de fonte de alimentação e qualidade da rede.

4.3. Eficiência da Comunicação

A eficiência foi avaliada com base no número de leituras bem-sucedidas e falhas ao longo dos **10 testes realizados**:

- **Leituras bem-sucedidas:** 10 de 10 tentativas
- **Taxa de sucesso: 100%**

Falhas observadas:

- Cartão não lido: **0 ocorrências (nenhuma falha)**
- Erro na requisição HTTP: **0 ocorrências (nenhuma falha)**
- Perda de conexão Wi-Fi: **0 ocorrências (nenhuma falha)**

O sistema demonstrou **alta confiabilidade** tanto na leitura do cartão quanto na comunicação com o servidor.

4.4. Considerações Finais

O sistema baseado em **ESP32-C3 SuperMini + PN532** demonstrou desempenho consistente e resposta rápida em todas as tentativas. Apesar da ausência de instrumentação precisa para medir o consumo de energia, os valores estimados indicam viabilidade para aplicações alimentadas por fonte externa ou bateria. A latência observada está dentro dos parâmetros aceitáveis para aplicações de controle de acesso, autenticação e sistemas similares.

5.Uso do Git

O sistema esta sendo versionado através do GitHub através do repositório:

https://github.com/ViniciusVeiga2002/esp32_rfid_2fa