# Projeto T3 Microcontroladores e Sistemas Embarcados – Pi Clock

## Conceito/Descrição do projeto:

O Pi Clock é um sistema de relógio inteligente desenvolvido para rodar em uma Raspberry Pi 3, utilizando um display touchscreen de 7 polegadas próprio para a plataforma.  
  
A segunda versão do projeto expande o conceito original ao integrar o Pi Clock à plataforma IoT Ubidots, transformando o sistema em um dispositivo embarcado conectado capaz de enviar e receber informações pela internet. Essa integração permite monitorar o relógio, os alarmes e as informações meteorológicas remotamente, além de criar, editar e acionar alarmes diretamente através de um painel na nuvem.  
  
A aplicação, implementada em Python, utiliza uma interface gráfica desenvolvida em Tkinter, exibindo hora em tempo real, data, dia da semana e previsão do tempo local obtida via API do OpenWeatherMap. Os alarmes são armazenados localmente e podem ser criados tanto na própria interface quanto remotamente via Ubidots, sendo notificados ao usuário através de um buzzer.  
  
Com a nova integração, o Pi Clock envia periodicamente dados de temperatura, eventos de alarme e status de funcionamento para o Ubidots, enquanto também pode receber comandos remotos para tocar, parar ou agendar alarmes. Toda a comunicação é feita por meio da API REST do Ubidots, com autenticação por token.  
  
O projeto reforça a proposta da disciplina ao unir hardware, software e conectividade IoT, demonstrando a aplicação prática de sistemas embarcados modernos capazes de se comunicar com a nuvem e reagir a comandos externos.

## Requisitos do sistema:

|  |  |  |
| --- | --- | --- |
| ID | Requisito | Tipo |
| UR-01 | Utilizar a Raspberry Pi 3 como plataforma de desenvolvimento | Obrigatório |
| UR-02 | Integrar um display touchscreen de 7” para interação com o sistema | Obrigatório |
| UR-03 | Exibir hora em tempo real | Obrigatório |
| UR-04 | Exibir data e dia da semana | Obrigatório |
| UR-05 | Permitir a criação, edição e exclusão de alarmes programados | Obrigatório |
| UR-06 | Notificar o usuário através de um buzzer ao disparar o alarme | Obrigatório |
| UR-07 | Exibir a temperatura local (atual, mínima e máxima) obtida via API | Obrigatório |
| UR-08 | Implementar função de soneca (adiar alarme em 5 minutos) | Obrigatório |
| UR-09 | Exibir o próximo alarme programado na tela principal | Obrigatório |
| UR-10 | Enviar e receber dados através da plataforma Ubidots IoT | Obrigatório |
| UR-11 | Permitir o controle remoto do alarme via dashboard do Ubidots | Obrigatório |
| UR-12 | Permitir a criação remota de novos alarmes pela nuvem | Desejável |
| UR-13 | Realizar comunicação assíncrona (threads) sem travar a interface | Obrigatório |
| UR-14 | Utilizar arquivo .env para armazenamento seguro de tokens e chaves | Obrigatório |

## Diagrama de blocos:

A seguir, o diagrama de blocos representando a nova arquitetura do sistema com integração à nuvem Ubidots:

┌──────────────────────────────┐  
 │ Dashboard │  
 │ (Ubidots Cloud Platform) │  
 └──────────────┬───────────────┘  
 │ (REST API HTTPS)  
 ▼  
 ┌──────────────────────────────┐  
 │ Raspberry Pi 3 + PiClock │  
 │ Python + Tkinter + Threads │  
 └──────────────┬───────────────┘  
 │  
 ▼  
 Alarmes Locais