Technology Requirements

Quintin edited this page on Aug 11 · 3 revisions

Technology Requirements - Version 3

Hardware

Raspberry Pi

- **Model**: Raspberry Pi 3 or higher
- **Operating System**: Raspbian or compatible Linux distribution
- **Processor**: ARMv7, Qualcomm Krait, or higher
- **Memory**: 1GB RAM or higher
- **Reason**: Required by user, part of MVP

IoT Device

- **Interface**: USB or UART for connection to Raspberry Pi
- **Firmware**: Must support two-way communication
- **Reason**: Device from which data must be extracted.

Software

Python Environment

- **Version**: Python 3.x installed on the Raspberry Pi
- **Libraries**:
  - Libraries for USB or UART communication (e.g., `pyserial`)
  - Libraries for XML parsing and formatting (e.g., `xml.etree.ElementTree`, `lxml`)
- **Reason**: Has a lot of support on Raspberry pi systems as Unix based systems have easy integration with running python applications and importing and using python libraries.

Storage (Optional)

- **MicroSD Card**: For storing retrieved data onboard the Raspberry Pi
- **Reason**: This is the main mechanism for storing and running the raspberry pi OS.

External System (Optional)

- **Receiving System**: External system (e.g., a computer or server) with network connectivity for receiving transmitted data, if not stored onboard

Development Environment

- **Integrated Development Environment (IDE)**: For writing Python scripts (e.g., Visual Studio Code)
- **Remote Access**:
  - **SSH**: For remote development and testing on the Raspberry Pi as this is the easiest way to access it securely.
  - **HDMI**: For graphical access to the Raspberry Pi

Additional Considerations

- **Networking**: Ensure that the Raspberry Pi has network connectivity for updates and data transmission if using an external system.
- **Power Supply**: Adequate power supply for both the Raspberry Pi and connected IoT device.
- **Security**: Secure SSH access with strong passwords or keys, and ensure the IoT firmware supports secure communication protocols.
- **Backup and Recovery**: Regular backups of the MicroSD card or external storage to prevent data loss.

Database

- **MongoDB Atlas**: For storing data using NoSQL
- **Reason**: Flexible structure, Open-source and is able to accommodate the free formed structure of the data that is extracted from the IoT devices.

Server

- **DigitalOcean**: Cloud based server running on Ubuntu in the cloud.

- **Reason**: Easy to use, cheap, still supported as opposed to Vercel. Secure and allows for the deployment of our web application easily and efficiently, minimal downtime and overhead.

Frontend

- **React**: React is a popular, efficient, and flexible JavaScript library for building user interfaces. It allows for the creation of reusable UI components and offers a declarative approach to managing the application state, which can improve the development process and user experience.
- **Reason**: Easy to use, widespread support, easy integration with NodeJS, familiar architecture and syntax.

Backend

- **NodeJS**: Used for building and running our RestFUL API.
- **Express**: Back end web application framework for building RESTful APIs with Node.js, free and open-source. It is the de facto standard server framework for Node.js
- **Reason**: Node.js is well-suited for building scalable and efficient backend services due to its non-blocking, event-driven architecture. It enables the use of JavaScript for both the frontend and backend, promoting code reusability and consistency. Additionally, Node.js has a rich ecosystem of libraries and modules that can accelerate development.

- Hardware
  - Raspberry Pi
  - IoT Device
- Software
  - Python Environment
- Storage (Optional)
- External System (Optional)
- Development Environment
- Additional Considerations
- Database
- Server
- Frontend
- Backend

▸ **Testing Document**

▸ **Use Case Diagram**

▸ **User Manual**

▸ **User Stories**

+ Add a custom sidebar

Clone this wiki locally

```
https://github.com/COS301-SE-2024/IoT-DIRfram.wiki.git
```