服务器端使用的技术

服务器端整体架构：

Springboot + JPA + websocket + socket

Mysql数据库

Springboot(总框架):

Spring Boot框架的核心就是自动配置，只要存在相应的jar包，Spring就帮我们自动配置。如果默认配置不能满足需求，我们还可以替换掉自动配置类，使用我们自己的配置。另外，Spring Boot还集成了嵌入式的Web服务器，系统监控等很多有用的功，让我们快速构建企业及应用程序。

**Spring boot的特点:**

1. 创建独立的Spring应用程序

2. 嵌入的Tomcat，无需部署WAR文件

3. 简化Maven配置

4. 自动配置Spring

5. 提供生产就绪型功能，如指标，健康检查和外部配置

6. 绝对没有代码生成和对XML没有要求配置

JPA(服务器进行数据处理时用的技术):

全称Java Persistence API，通过JDK 5.0注解或XML描述对象－关系表的映射关系，并将运行期的实体对象持久化到数据库中。

JPA的出现有两个原因：

其一，简化现有Java EE和Java SE应用的对象持久化的开发工作；

其二，Sun希望整合对ORM技术，实现持久化领域的统一。

JPA提供的技术：

1）ORM映射元数据：JPA支持XML和JDK 5.0注解两种元数据的形式，元数据描述对象和表之间的映射关系，框架据此将实体对象持久化到数据库表中；

2）JPA 的API：用来操作实体对象，执行CRUD操作，框架在后台替我们完成所有的事情，开发者从繁琐的JDBC和SQL代码中解脱出来。

3）查询语言：通过面向对象而非面向数据库的查询语言查询数据，避免程序的SQL语句紧密耦合。

JPA是规范，Hibernate是框架，JPA是持久化规范，而Hibernate实现了JPA。

Websocket(和前端进行实时监测交互时使用的技术):

WebSocket是HTML5出的东西（协议），也就是说HTTP协议没有变化，或者说没关系。

相对于HTTP这种非持久的协议来说，Websocket是一个持久化的协议。

它将 TCP 的 Socket（套接字）应用在了web page上，从而使通信双方建立起一个保持在活动状态连接通道，并且属于全双工（双方同时进行双向通信）。

在WebSocket API中，浏览器和服务器只需要完成一次握手，两者之间就直接可以创建持久性的连接。（为了创建Websocket连接，需要通过浏览器发出请求，之后服务器进行回应，这个过程通常称为“握手”）

它的最大特点就是，服务器可以主动向客户端推送信息，客户端也可以主动向服务器发送信息，是真正的双向平等对话，属于服务器推送技术的一种。

**其他特点**:

建立在 TCP 协议之上，服务器端的实现比较容易。

与 HTTP 协议有着良好的兼容性。默认端口也是 80 和 443 ，并且握手阶段采用 HTTP 协议，因此握手时不容易屏蔽，能通过各种 HTTP 代理服务器。

数据格式比较轻量，性能开销小，通信高效。

可以发送文本，也可以发送二进制数据。

没有同源限制，客户端可以与任意服务器通信。

协议标识符是ws（如果加密，则为wss），服务器网址就是 URL。

WebSockets还会考虑代理和防火墙等危险，使得任何连接都可以进行流式传输。它支持单个连接的上游和下游通信。 它还减轻了服务器的负担，允许现有机器支持同时连接。

Socket(接收传感器数据时使用的技术)

在数据库中利用Trigger和Event 完成一些自动化的功能