Reflektion

Jag läste uppgiften noggrant och skapade specifika punkter som programmet måste ha. Som TCPclient som kan skriva och läsa, en HTTP client som använder TCP client för att skicka och läsa. En HTTP creator som skapar en HTTP request baserat på metod, endpoint och body.

Scrum Metoden gör det lättare att jobba med andra personer i ett stort projekt. Man skriver specifika user stories som beskriver behov och hur det ska lösas i backloggen. Man ser till att varje task kan lösas oberoende av andra delar av projektet så att man kan jobba oberoende av vad de andra gör. Detta kräver att man skriver specifika user tasks.

Makefile underlättar utvecklingen genom att göra det mycket lättare att snabbt bygga om och köra programmet utan att manuellt behöva använda gcc. Git gör det lättare att hantera hur projektet ändras över tid och sparar alla olika versioner. Det gör det lätt att gå tillbaka, om en commit har förstört programmet kan man bara göra en rollback. Det gör det lättare att jobba med flera andra personer på samma projekt.

TCP är ett protokoll hur data skickas mellan enheter. Den garanterar att all data kommer fram utan att förlora en enda byte. HTTP är ett protokoll för hur text ska vara formaterad som gör det lätt för en server att förstå och skicka ett relevant svar.

Pekare måste användas om man vill att en annan funktion ska ändra som man sedan vill använda utanför den funktionen. Annars så tar funktionen variabeln by value. Tex när HTTPClientent ska få access till sig själv i sina egna funktioner. Jag använde pek pekare för HTTPClient_initPtr för att funktionen ska kunna ändra vart pekaren pekar när man heap allokerar datan. Functions pekare är en funktion du skickar in till en annan funktion som den senare kan använda. Det används oftast för callback när en funktion ska skicka vidare något som ska användas någon annanstans. Det används också för att modularisera ett program genom att användaren kan välja vilken funktion som ska användas vilket gör det mindre hårdkodat och lättare att använda samma kod för andra projekt i framtiden.

Min TCPClient read-funktion läser in data som skickas från servern in till en given buffer. Om den buffern är mindre än längden av meddelanden den tar emot så kommer inte hela meddelandet att läsas in. Man kan fixa detta genom att använda malloc/realloc för att hela tiden öka storleken av buffern så att meddelandet får plats.

Repo:
https://github.com/TheFreeDuck/smart-sensor