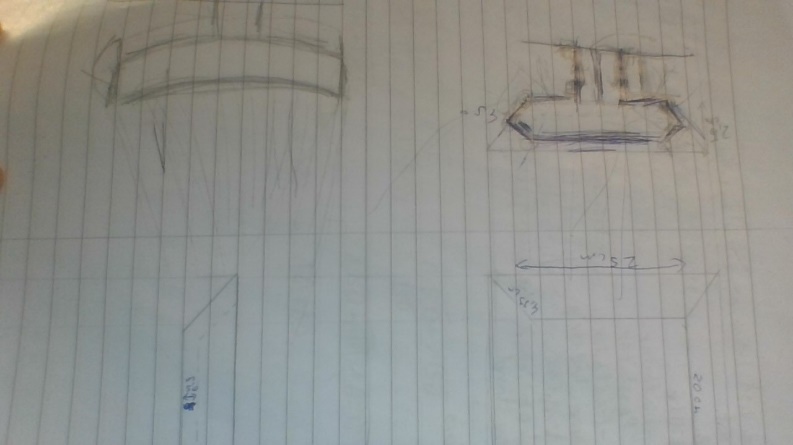
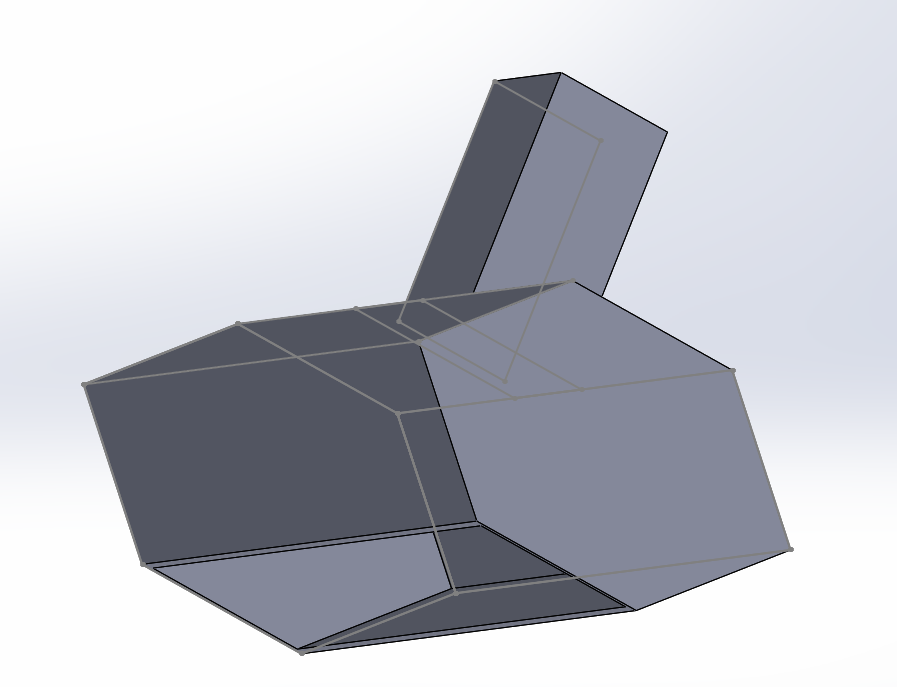
**En Ljus Ide av Grupp 6**

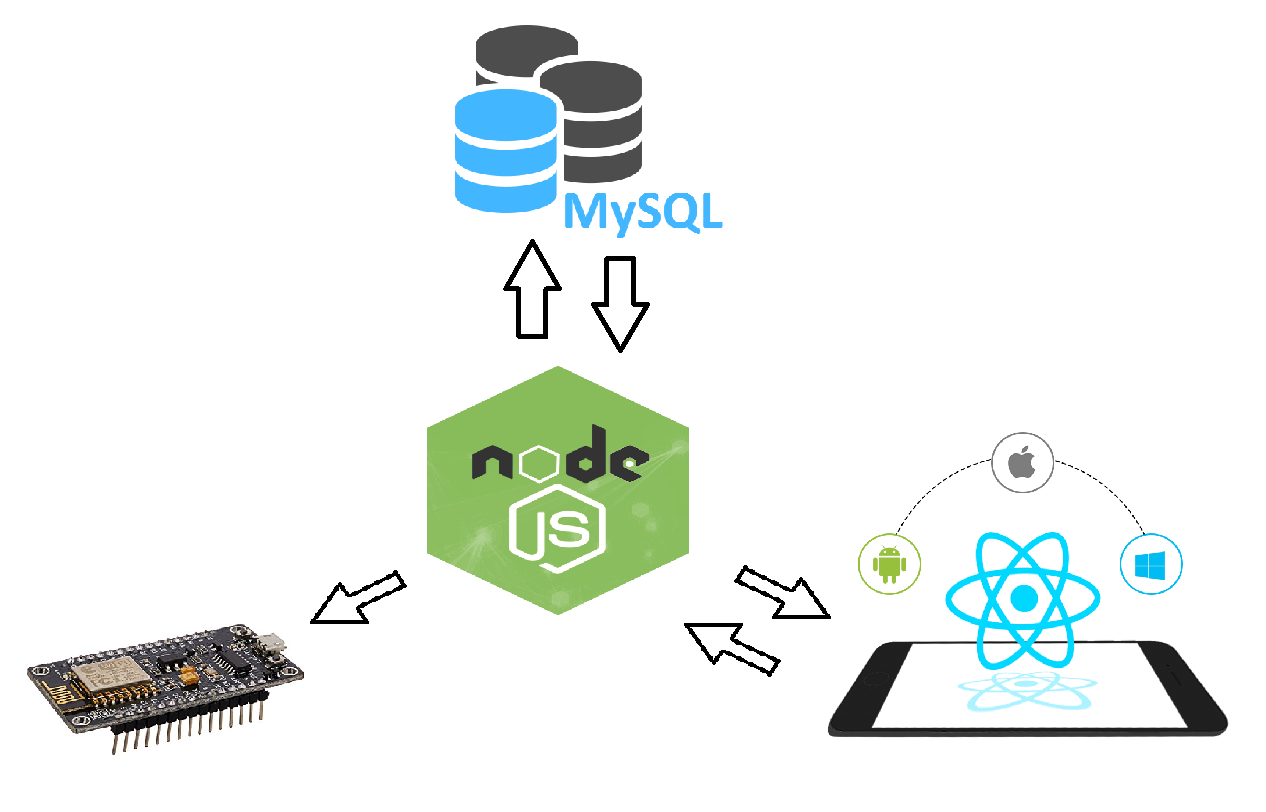
**Funktioner**

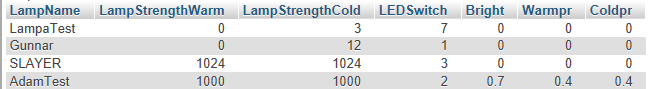
**Modellen**

****I början tänkte vi oss en mer traditionell lampa, d.v.s en upponedvänd triangel iprincip. Problemet vi stötte på var dels att vi inte gillade utseendet men främst så strävade vi efter en smartare lösning som lät oss få både direkt och indirekt ljus på ett snyggt sätt. Flera olika designer skapades men den vi fastnade för var en avlång lampa med hexagonformade sidor. Med denna design kunde vi få direkt ljus på ett indirekt sätt. Om LEDstripsen sitter på insidan kan vi få ljuset att reflekteras neråt, vilket gör att man inte behöver oroa sig för att råka kolla rakt in i starka LEDs. Men ifall lampan LEDsen var instängda i lampan skulle inget vi inte få något uppåt mot taket. För att lösa detta tänkte vi först sätta flera strips ovanpå laman för att få ljus, men beslutade att vi inte ville göra det. Dels för att vi trodde att man skulle kunna se LEDstripsen, vilket vi inte ville, och delvis för att det skulle innebära mer och krångligare koppling och fler sladdar. Dock så hade man kunnat ha mer kontroll över ljuset ifall man kunde styra LEDsen inuti och ovanpå separat, men vi bestämde oss för att det inte var värt. För att lösa problemet valde vi istället att använda plexiglas till två av väggarna som var vinklade uppåt. På detta vis skulle vi kunna få ljus både uppåt och neråt med samma LEDstrips. Detta gjorde att vi undvek en sladdhärva, kunde använde färre strips och dessutom så ser det snyggt ut.

Ovanpå armaturen har vi en avlång hållare som vi tänkte att man skulle använda för att fästa lampan i taket. Den hade även syftet att hålla våran mikroprocessor dold.

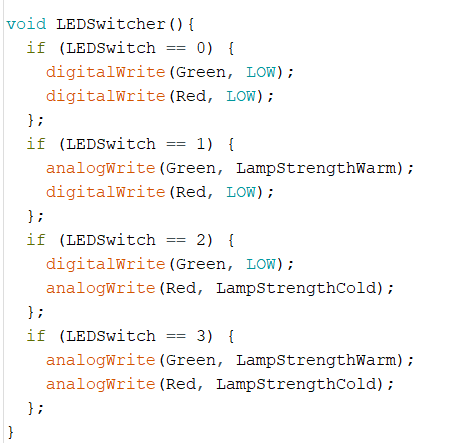
**Hur funkar det?**

****

Hela systemet bygger på vår backend. Man kan se det som huvudstationen i vårat system. Alla delar bygger använder backend. En av delarna är våran databas. Det är en MySQL databas och det är där vi har alla våra värden till lampan. Backended kan skicka en förfrågan till databasen om att få värden och databasen skickar värden till backended. Vi kan även använda oss av våran backend för att ändra i databasen för att t.ex ändra värden eller ta bort värden. Så här ser databasen ut:

Backenden är som sagt vår huvudstation i projektet. Det använder node.js och använder sig av GET, PATCH, POST och DELETE för att kommunicera med databasen. Den kan sedan vidarebefordra denna information till mobilappen eller mikroprocessorn.

Mobilappen….

Själva LEDStrippsen i lampan styrs utav en mikroprocessor (en NodeMCU ESP8266) som vi har laddat upp arduinokod till. Denna har internetuppkoppling vilket låter den kommunicera med backenden trådlöst. Det funkar så att mikroprocessorn ber om att få en värden från backenden genom att be den att göra en get request. Backenden kommer då skicka en GET-requst till databasen, få värden och sedan skicka de till mikroprocessorn. Mikroprocessorn läser sedan av värden den får och justerar LEDStrippsen utefter värdena. För att styra våra LEDs använder vi oss våra värden LampStrengthWarm, LampStrengthCold och LEDSwich. Båda LampStrength värdena avgör hur starkt de olika LEDstripparna lyser och LEDSwitch avgör vilka stripps som lyser. Vid 0 lyser ingen, vid 1 lyser en, vid 2 den andra och när värdet är 3 lyser båda sorterna av LEDstrip. Vi kunde göra detta med hjälp av en enkel metod som helt enkelt läser av värdet på LEDSwitch och justerar LEDsen efter det. Vi använder oss av analogWrite istället för digitalWrite eftersom att analogWrite har inbyggd pulsbreddsmodulering, vilket innebär att vi kan styra hur starkt vi vill att lampan ska lysa. Vi sätter styrkan efter värdena vi fått från backenden. Våra värden går mellan 1024 och 0, då 1024 är starkast och 0 är släckt.

Då vi använder node.js till backenden så kommer våra värden vara i json format. Det är viktigt att tänka på eftersom att jsonobjekt är i ett sådant format att vi inte kan använda de som de är i arduinokoden.

**Kod**

**Utvärdering**

**Lampans Framtid**