De 7 C’s

C1: Capture

Bij de eerste C moeten alle requirements opgeschreven worden en een gedetailleerde omschrijving geven. Hieronder zijn de requirements:

**Must haves:**

* Interactief Wezen: De applicatie moet een virtueel wezen met alleen 2 ogen presenteren.
* De Interactief Wezen moet goed op gebruikersinteracties kunnen reageren.
* Gebruikersinteractie: De applicatie moet ondersteuning bieden voor verschillende vormen van interactie, zoals aanraking.
* Beeld: Het virtuele wezen moet 30 frames per second hebben en op die manier een vloeiend beeld bieden.
* Response time: de response time van de interactie moet een halve seconde zijn.

**Should haves:**

* Personalisatieopties: De applicatie moet de mogelijkheid bieden voor gebruikers om het virtuele wezen aan te passen aan hun voorkeuren, zoals uiterlijk en gedrag.
* Gebruikersbegeleiding: Een duidelijke en intuïtieve gebruikersinterface is essentieel om gebruikers te begeleiden bij het gebruik van de applicatie en de interactie met het virtuele wezen.

**Could haves:**

* De interactief wezen kan aangesproken worden met spraakopdrachten.
* De applicatie is te koppelen met de fysieke PIA Desktop Companion.

**Won't haves:**

* Geen

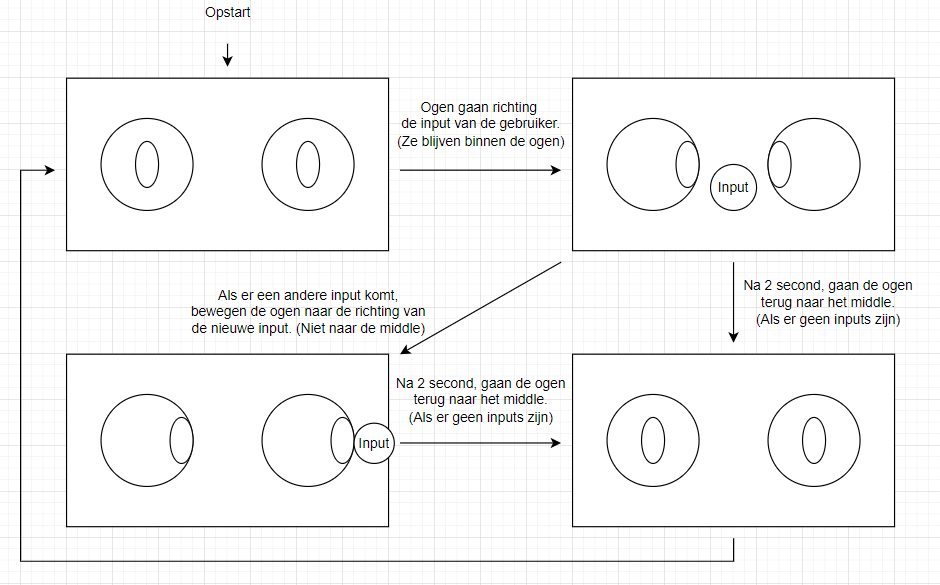
C2: Contemplate

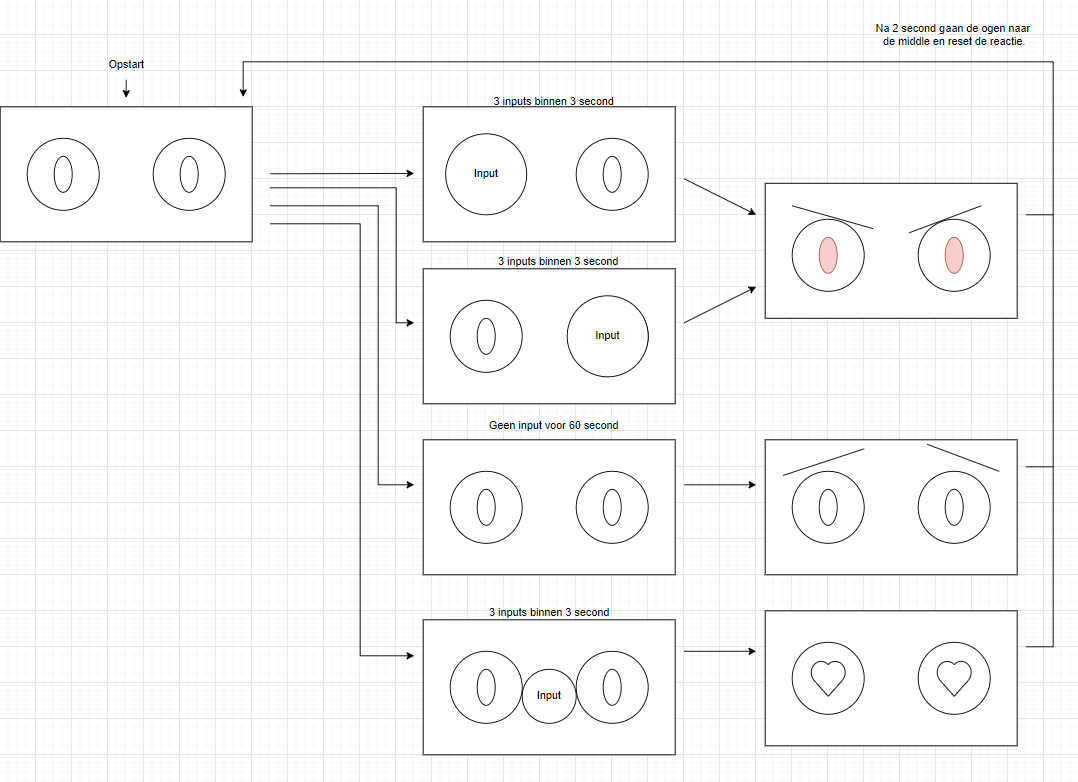
Bij de tweede C gaan we onze deelproblemen delen. Ook analyseren wij het probleem en zoeken welke data nodig is. Hieronder zijn de deelproblemen etc:

Het grootste probleem dat we hebben is dat we de python file naar een apk moeten converten. Ook een probleem dat we hebben is dat we met linux vastzaten en daarom een nieuwe oplossing moesten vinden. We hadden ook een probleem met het design van de applicatie. Ook hebben we een probleem dat de pupillen buiten de ogen gaan als we de positie willen bepalen. Ons laatste probleem was dat we niet wisten welke reacties/functies de robot moest hebben.

C3: Contract

Bij de derde C hoort het design van de UI. Dus eigenlijk de inhoud en de structuur. Hieronder ziet u de het design:





C4: Compose

Bij de vierde C moeten de problemen opgesplitst worden in deelproblemen. Hieronder ziet u al onze deelproblemen:

**Python-file naar een apk:** Om een python file te runnen op een telefoon als een applicatie moeten we het eerst converten naar een apk file.

**Linux:** Om een python file naar apk te converten zijn er veel opties. Het makkelijkste optie is om WSL Ubuntu te gebruiken.

**Design van de applicatie:** Het design was nog niet bepaald, waardoor er geen basis lay-out kon worden gemaakt.

**De pupillen gaan buiten de ogen als we de positie willen bepalen:** De beweging van de ogen waren niet duidelijk.

**Welke reacties/functies de robot moest hebben:** De reacties/functies van de robot waren niet bepaald. We moesten zelf wat verzinnen.

C5: Chart  
Bij de vijfde C moeten de problemen oplossen. Hieronder ziet u al onze probleemoplossingen:  
  
**Python-file naar een apk:** We hebben de python file met buildozer in WSL Ubuntu naar een apk ge-convert, en die konden we als een applicatie op een mobile device runnen.

**Linux:** We hadden eerst WSL uitgeprobeerd, het werkte niet. We hadden Ubuntu OS geprobeerd, het werkte niet. Dockers hebeben we ook uitgeprobeerd maar omdat we geen ervaring daarme hebben was het lastig. Uiteindelijk was de oplossing met WSL op Windows met een virtual Cython.

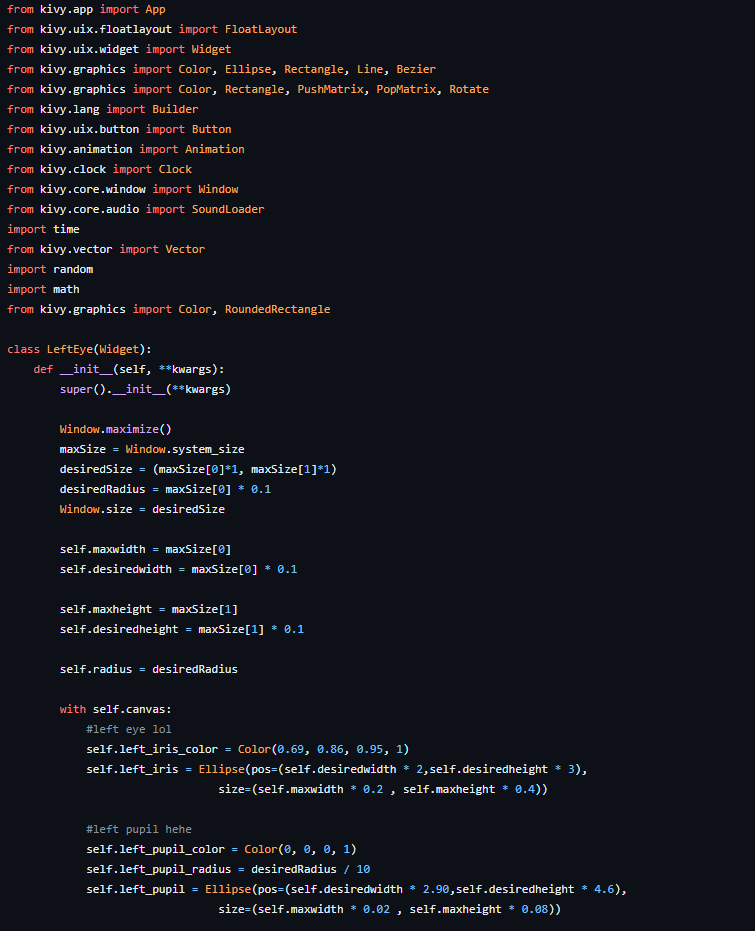
**Design van de applicatie:** We hebben het design van de applicatie besproken met Finn en zijn uiteindelijk tot de conclusie gekomen dat we zelf een design mochten maken. Ons desing is gebaseerd op "Hoe tem je een draak".

**De pupillen gaan buiten de ogen als we de positie willen bepalen:** We hebben een max position voor de ogen toegevoegd waardoor ze niet buiten het ogen kunnen gaan.

**Welke reacties/functies de robot moest hebben:** We hebben dit tijdens een gesprek met Finn besproken, en er moeten 3 standaard emoties komen namelijk blij, verdrietig en boos. Ook was het een optionele keuze om meer emoties of functies toe te voegen. Voor Max voegen we ook een dansende emotie toe.

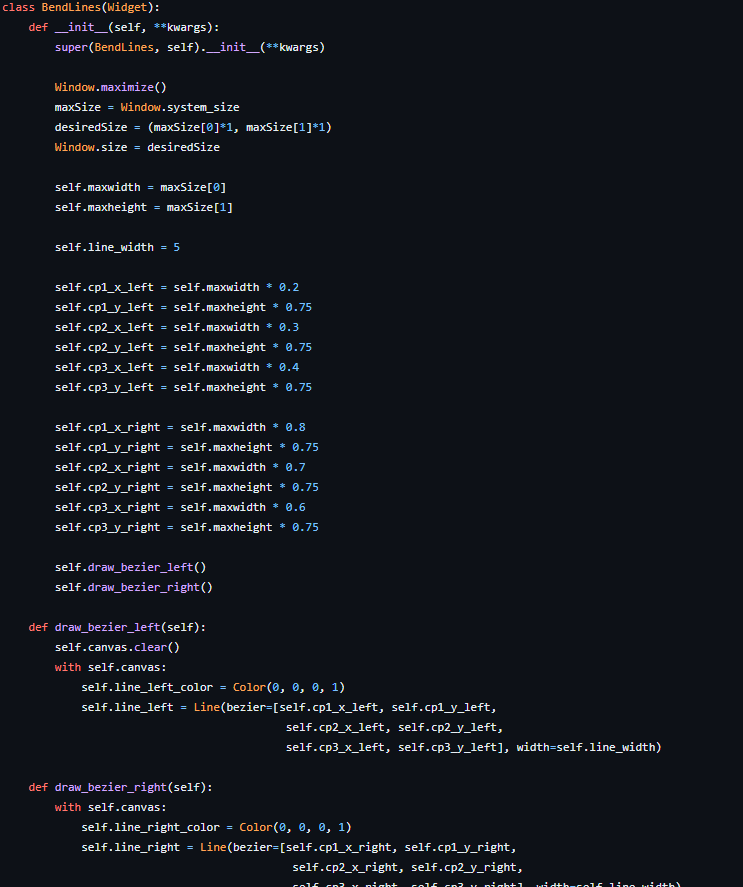
C6: Code  
Bij de zesde C moeten de code laten zien. Hieronder ziet u de code:

Dit is het eerste gedeelte van de code. In het eerste stuk staat de class LeftEye. Deze class duidt het linkeroog aan. Zie de code hieronder:

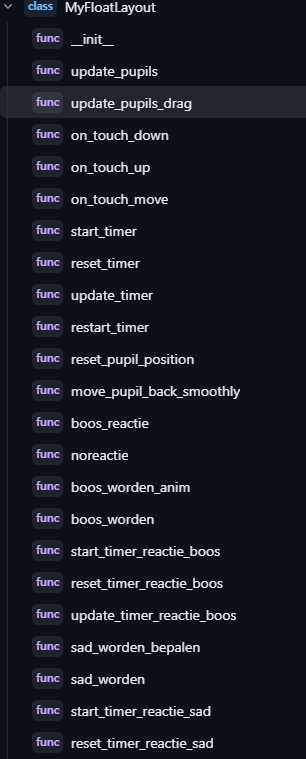


In het tweede deel staat de class RightEye. Deze class bevat hetzelfde als de LeftEye class, daarom is er geen foto nodig. Later realiseerde we dat we allebei de ogen in 1 class konden zetten. De up to date versie van onze code bevat dus de 2 classes samengevoegd.

In het derde deel staat de class Bendlines. In deze class worden de eyebrows aangesproken. Zie hieronder de code:



In het laatste deel staat de MyFloatLayout class. Hierin staat alles voor de applicatie zoals Functions, widgets toevoegen en Booleans. Ook staat er een Myapp class om de app te runnen. Zie hieronder de informatie wat erin staat:



C7: Check

Bij de laatste C hoort Check. Hierbij kijk je of alle eisen zijn voldaan en evalueer je het hele proces. Ook kijk je of er nog iets verbeterd kan worden door opnieuw de stappen te doorgaan en kijken wat beter kan.

Dit hebben we gedaan en we zijn erachter gekomen dat de code efficiënter geschreven kan worden door middel van OOP. Wat we nog van plan zijn, is de code efficiënt maken en eventueel optionele dingen toevoegen zoals een menu met nieuwe kleurenthema's en geluiden.