

# Documentatie Pinautomaat

Bytegroep 10

20 juni 2018

## **Samenvatting**

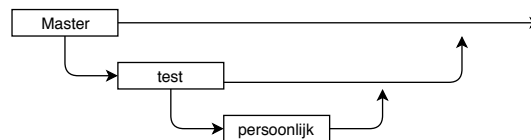
De opdracht voor dit project was om een werkende geldautomaat te maken. In dit verslag zijn onderzoeksanalysen met adviezen terug te vinden. Wij hebben aan de hand van deze adviezen hierna ontwerpen gemaakt en gerealiseerd. Wij moeten tijdens de loop van het project letten op vijf onderdelen: *beheren, analyseren, adviseren, ontwerpen en realiseren*.

# Inhoudsopgave

|          |                                     |           |
|----------|-------------------------------------|-----------|
| <b>1</b> | <b>Version control</b>              | <b>3</b>  |
| <b>2</b> | <b>Samenwerkingscontract</b>        | <b>3</b>  |
| <b>3</b> | <b>Kwaliteitseisen</b>              | <b>5</b>  |
| 3.0.1    | Code eisen . . . . .                | 5         |
| 3.0.2    | Product eisen . . . . .             | 5         |
| 3.0.3    | Eindgebruiker eisen . . . . .       | 5         |
| 3.0.4    | Git eisen . . . . .                 | 5         |
| <b>4</b> | <b>Analyse &amp; advies</b>         | <b>5</b>  |
| 4.1      | Dispenser . . . . .                 | 5         |
| 4.2      | Veiligheid . . . . .                | 6         |
| 4.3      | Biljetten . . . . .                 | 6         |
| 4.4      | Sensoren . . . . .                  | 7         |
| 4.5      | Communicatie . . . . .              | 7         |
| 4.6      | Bonnen . . . . .                    | 9         |
| 4.7      | Materiaal . . . . .                 | 10        |
| <b>5</b> | <b>Ontwerp &amp; realisatie</b>     | <b>11</b> |
| 5.1      | Ontwerp proces dispensers . . . . . | 11        |
| 5.2      | Bouwtekening ATM . . . . .          | 12        |
| 5.3      | Electrisch schema . . . . .         | 13        |
| 5.4      | Toepassen kwaliteitseisen . . . . . | 14        |
| 5.5      | Testen . . . . .                    | 14        |
| <b>6</b> | <b>Bijlagen</b>                     | <b>15</b> |

# 1 Version control

Voor dit onderdeel moesten wij kunnen werken met Git en Version control. Hier is veel onderzoek over gedaan zodat het goed toegepast kan worden dit project. In figuur 1 is het model wat wij gebruiken op onze Git pagina, Project4Bankalicious. Ieder groepslid werkt op zijn individuele branch, en als je klaar bent met jouw onderdeel, dan wordt hij in test gemerged. Als er geen problemen lijken te zijn op de test beslissen wij in een groep of wij test naar de master kunnen schrijven. Alle documentatie is terug te vinden op Github.



Figuur 1: git model

## 2 Samenwerkingscontract

**Financiën** De kosten voor het gezamenlijke deel van de opdracht(en) worden gezamenlijk betaald, iedereen betaald even veel.

**Schades** Als er iets kapot gaat, worden de kosten van het vervangen gedeeld door de groep tenzij er opzettelijk of onverantwoordelijk mee om is gegaan. In dat geval zullen de kosten voor de persoon die de schade heeft aangericht zijn.

### Github afspraken

- Niet op de master werken. Je werkt altijd op een eigen kopie van de test branch.
- Gelieve geen word bestanden te uploaden zonder ook een pdf te uploaden.
- Commits moeten duidelijk beschrijven wat er is aangepast. Liever een paar woorden teveel dan te weinig.
- De repo van project 4 is geen zandbak om Git te testen.
- Directories zijn met kleine letters en underscores, en hebben een duidelijke naam.
- Alle benamingen en commits moeten in het Nederlands.
- Branch namen zijn met kleine letters en woorden zijn verbonden met een min teken.

**Te laat bij afspraken** Al onze afspraken worden in goed overleg gepland daarom wordt er van groepsleden verwacht dat ze op tijd zijn bij een afspraak (zowel deadlines als overleg) als een groepslid zonder goede 1 reden meer dan 5 minuten te laat is dan staat daar een consequentie tegen over. Deze consequentie is of 20 chicken mcnuggets of een taart (let op: cake is geen taart!), als er meerdere mensen te laat zijn moeten zij allebei iets mee nemen.

**Afwezigheid bij afspraken** Al onze afspraken worden in goed overleg gepland daarom wordt er van groepsleden verwacht dat ze bij afspraken aanwezig zijn. als een groepslid zonder goede reden afwezig is bij een overleg staat daar een consequentie tegenover. Deze consequentie is of 40 chicken mcnuggets of twee taarten. Deze reden zal met de groep besproken moeten worden voordat het een geldige reden is.

**Communicatie** Communiceer goed, dat betekent duidelijk en met respect durf anderen aan te spreken. Vraag hulp als je hulp nodig hebt. Geef op tijd aan als je wil stoppen met de studie en draag alle onderdelen van het project tijdig over aan een ander groepslid.

**Rolverdeling** De rolverdeling staat netjes gedocumenteerd op Github, en ieder teamlid is gediend deze zo goed mogelijk uit te voeren.

- Paul H. Organizer
- Paul W. Verbinder
- Gerard Expert
- Merijn Documentatie manager, onderzoeker
- Aron Tester
- Boas Kwaliteitsbewaker
- Floor Designer, expert
- Mohammed Designer

## **3 Kwaliteitseisen**

### **3.0.1 Code eisen**

### **3.0.2 Product eisen**

### **3.0.3 Eindgebruiker eisen**

### **3.0.4 Git eisen**

Zie sectie 2 voor de eisen voor onze git pagina.

## **4 Analyse & advies**

### **4.1 Dispenser**

Wij hebben onderzoek gedaan naar twee manieren van het uitgeven van biljetten. Wij hebben vooral onderzoek gedaan bij de tweede jaars. Optie één is om een zuigpomp te gebruiken en de biljetten naar boven te zuigen. De tweede jaars raden dit af, omdat zij hier zelf geen goede ervaring mee hadden.

Manier twee is om de biljetten uit te werpen met een elektromotor. De tweede jaars hadden hier goede ervaring mee, dit ding namelijk minder vaak fout bij het uitwerpen. De elektromotor zal de kaarten uit de dispenser rollen.

Voor het uitgeven van biljetten is het efficiënter om een elektromotor te gebruiken. Hier is er meer kans dat het juiste aantal biljetten uit de automaat komen.

## 4.2 Veiligheid

**Netwerk encryption** Er is veel onderzoek gedaan naar de beveiliging van het dataverkeer. Voor netwerken beveiliging heb je een encryptie nodig over het netwerk. Wij raden aan om hiervoor een bestaand protocol toe te passen. SSL wordt professioneel toegepast bij bedrijven. Het kan gewoon gebruikt worden als hij goed toegepast wordt met de meest recente versie.

**lokale encryption** Voor lokale encryption zou je de hele database kunnen encrypten, en decrypten wanneer je data nodig hebt. Dit kost echter een hoop rekenkracht, en je gegevens zullen op een paar plaatsen onversluiteld zijn.

Een oplossing hiervoor is om gevoelige gegevens te hashen, en vervolgens in de database op te slaan. Als je bijvoorbeeld wilt controleren of een pincode klopt, zal deze eerst gehasht worden en vervolgens vergeleken met wat er in de database staat. Een veelgebruikt hashing algoritme voor java en c++, de talen waar wij in programmeren, is bijvoorbeeld md5. Het is beter om een hashing algoritme te gebruiken wat iets zwaarder is dan md5, omdat je een hash makkelijk terug kunt rekenen. Een beter veelgebruikt algoritme is bijvoorbeeld SHA1. Deze adviseren wij te gebruiken.

$$\text{pincode} \xrightarrow{\text{hash}} \text{hashed pincode} \xrightarrow{\text{vergelijk}} \text{Databasemethashes}$$

**Afgesloten hardware** Het is heel cruciaal voor de automaat dat buitenstaanders geen toegang hebben tot de logica van de machine en de biljetten. Om dit te voorkomen is het het handigst om de hardware aan de binnenkant van de automaat te houden. Wij raden aan om een kast om de functionele onderdelen heel te bouwen, zodat deze niet meer toegankelijk zijn.

## 4.3 Biljetten

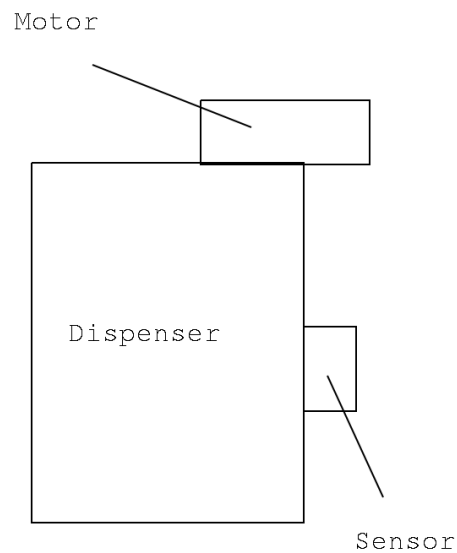
Wij hebben eerst aan de tweede jaars gevraagd wat voor soort materiaal wij het best konden gebruiken voor de biljetten. Zij kwamen met twee opties.

- Je kan zelf biljetten maken, om ze zo op je model af te stellen.
- Je kan speelkaarten gebruiken, omdat deze gemaakt zijn om met een dispenser uit te werpen.

Wij raden aan voor het gebruik van speelkaarten vanwege de dispenser. De manier hoe de dispenser gemaakt is zorgt ervoor dat alleen de speelkaarten geschikt zijn voor biljetten. Wij kunnen direct aan te slag met de dispenser als wij speelkaarten gebruiken, sinds wij deze al hebben. Er zijn 4 dispensers waarvan elk een bepaald type biljet bevat. Eén dispenser bevat biljetten van €5, een ander van €10, een ander weer van €20 en één van €50.

## 4.4 Sensoren

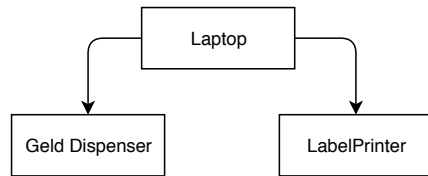
In een ATM kan er maar een bepaald hoeveelheid van biljetten. In de ATM moet er bijgehouden worden hoeveel biljetten er zijn uitgegeven. Om de uitgave van biljetten te meten gaan wij gebruik maken van sensoren. Er zullen 4 sensoren gebruikt worden bij elke dispencer. Elke sensor kan bijhouden hoeveel biljetten van zijn aangewezen dispencer zijn uitgegeven. In het systeem van de ATM wordt dit geregistreerd.



Figuur 2: Zijaanzicht Dispenser Sensor

## 4.5 Communicatie

De laptop zal opdrachten sturen naar de Labelprinter en de geld dispenser. Zie de netwerk diagram in figuur 3



Figuur 3: Netwerk Diagram

Hieronder is in pseudocode te zien welke opdrachten de laptop kan versturen naar de verbonden apparaten

```
1 public class App{  
2     new Printer("anaam", "vnaam", "atmNaam", "bedrag");  
3  
4     new Dispenser;  
5     dispenseMoney("aantal5euro", "aantal10euro", "aantal20euro",  
6         "aantal50euro");  
}
```

Voor de communicatie tussen de klassen in de GUI van de automaat zijn in een diagram weergegeven. Om deze te bezichten, zie de bijlage.



## 4.6 Bonnen



Figuur 4: Bon ontwerp

Voor het uitgeven van bonnen hebben wij een labelprinter gebruikt. In figuur 4 is de bon te zien die de printer uitprint. Hieronder is een link naar de sourcecode.

🔗 [Code labelprinter](#)

### Pseudocode uitleg

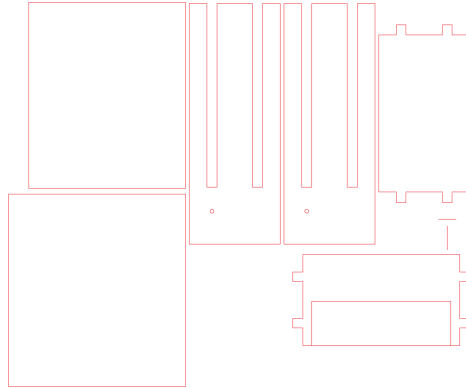
```
1 public class PrinterClass {
2     String string;
3     // constructor
4     public PrinterClass(voornaam, achternaam, atmNaam, bedrag)
5     {
6         string = inhoud wat geprint moet worden.
7
8         Printerjob pj = voorbereiden voor printen.
9         pageformat pf = het formaat waarin geprint word.
10        pf.setOrientation(PageFormat.PORTRAIT); zet de
11        orientatie naar portrait
12        pf.setPaper(paper); zet hoe groot het papier is zodat
13        alles op de bon komt.
14        pj.setPrintable(new MyPrintable(), pf);
15        try {
16            pj.print();
17        } catch (PrinterException ex) {
18            ex.printStackTrace();
19        }
20        Bovenstaande stukje zorgt ervoor dat het daadwerkelijk
21        geprint word en dat een error word opgevangen.
22    }
23 }
```

## 4.7 Materiaal

Er zijn hier een paar opties. We kunnen bijvoorbeeld onderdelen 3d printen, maar dat zou erg duur zijn. Het meest voor de hand liggende materiaal is triplex hout. Dit is een redelijk goedkoop, en makkelijk verkrijgbaar materiaal. Wij kunnen dit in het *Tesla Lab* ophalen en daar direct uitsnijden.

## 5 Ontwerp & realisatie

### 5.1 Ontwerp proces dispensers



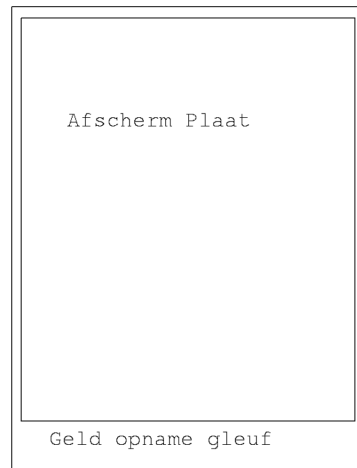
Figuur 5: Dispenser ontwerp

**Ontwerp** De automaat moest vier dispensers hebben. Wij zijn begonnen met een prototype te maken voor een van deze dispensers. In figuur 5 is een bouwtekening te zien die wij hebben uitgesneden en daarna in elkaar gezet.

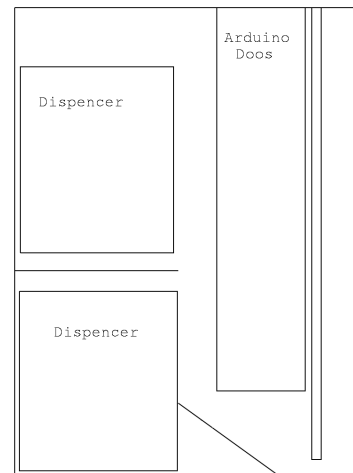
Tijdens het vormgeven bleek dat elastiekjes het makkelijkste werkten om het plateau waar het geld in ligt naar boven te trekken. Ook bleek dat de as van de motor het best van lego gemaakt kon worden. Hiervoor was het idee om de as van karton te maken, maar dit bleek niet sterk genoeg te zijn en moeilijk te maken. Na het testen met de elastiekjes en de as hebben wij de meeste onderdelen aan elkaar gelijmd.

**Ontwerp 2.0** Hierna hebben we op de millimeters nauwkeurig gekeken wat de beste afstanden zouden zijn voor de uitgaven van de kaarten. Aan de hand hiervan hebben wij een nieuwe print gemaakt en opnieuw alles getest. Deze nieuwe print wijkt enkele millimeters af van het eerste ontwerp Deze bleek goed, en wij hebben *Ontwerp 2.0* dus uiteindelijk drie keer extra geprint.

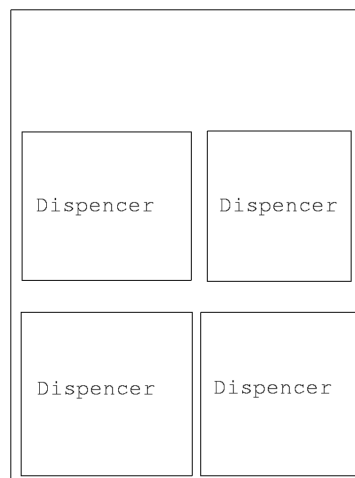
## 5.2 Bouwtekening ATM



Figuur 6: Vooraanzicht ATM

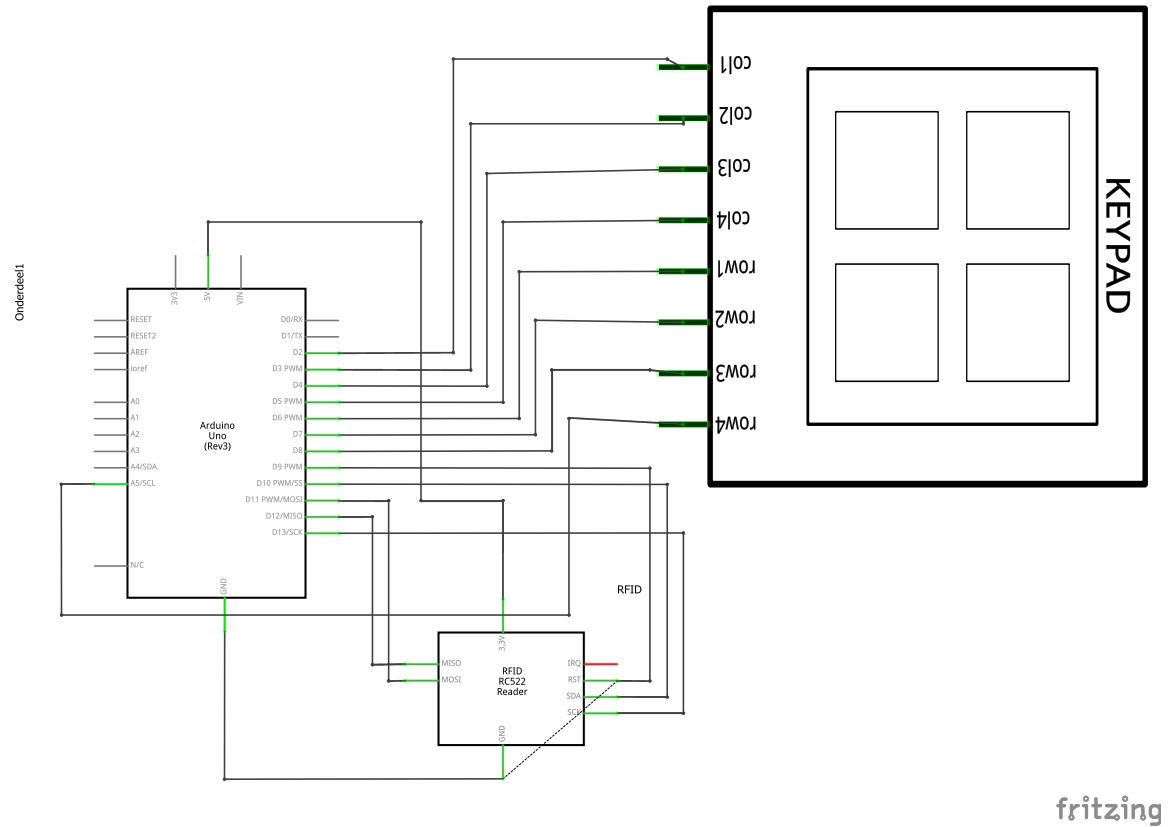


Figuur 7: Zijaanzicht ATM



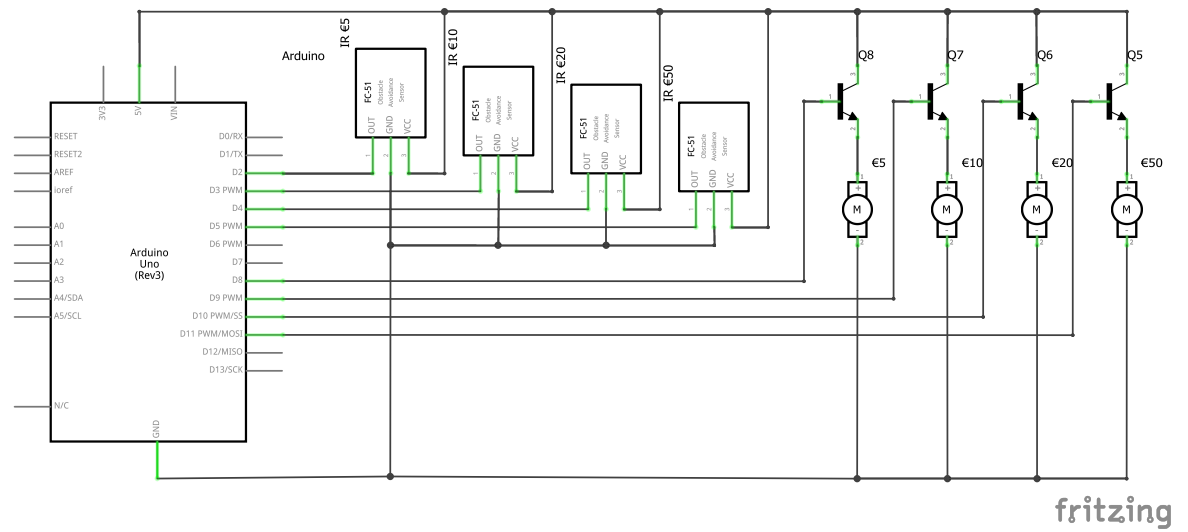
Figuur 8: Achterkant ATM

### 5.3 Electrisc schema



fritzing

Figuur 9: Keypad en RFID

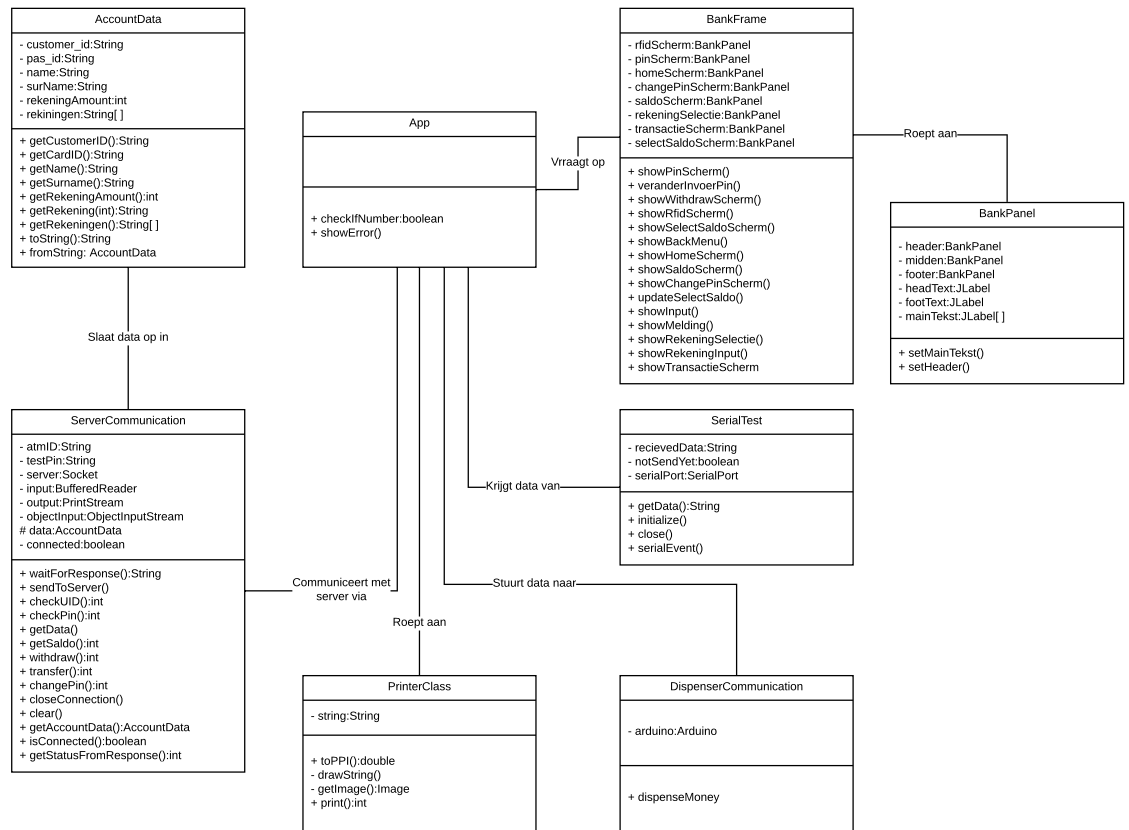


Figuur 10: Dispencer Schema

#### 5.4 Toepassen kwaliteitseisen

#### 5.5 Testen

## 6 Bijlagen



Figuur 11: Klassediagram GUI