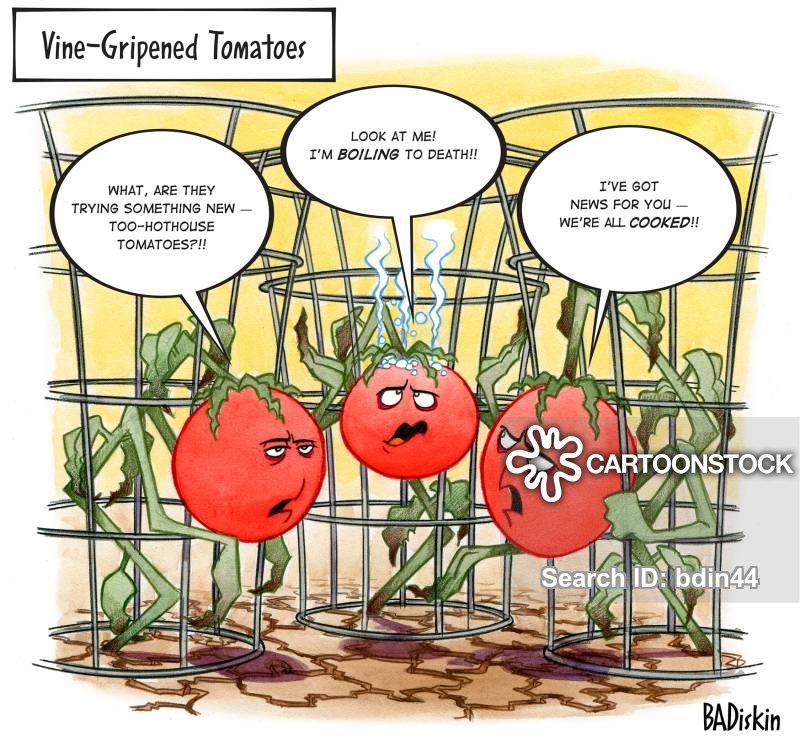
Tomato - The Automatic Greenhouse

productdocument

****

Student: Alwin Rodewijk

635653

Vak: Inleiding Software Engineering, D-B-INSE-O

Docent: Jos Onokiewicz

21-11-2019

Documenthistorie

|  |  |  |  |
| --- | --- | --- | --- |
| **Datum** | **Versie** | **Wie** | **Veranderingen** |
| 21-11-2019 | 0.1 | Alwin Rodewijk | Startup |
| 16-12-2019 | 0.2 | Alwin Rodewijk | Verschillende toevoegingen |
| 24-01-2020 | 0.3 | Alwin Rodewijk | Samenvatting t/m Eindresultaat aangevuld, gereed voor controle |
| 25-01-2020 | 0.3 | Alwin Rodewijk | TODO |
|  |  |  |  |
|  |  |  |  |

# Samenvatting

Het “Tomato” project is een volledig automatische kas die voor een type plant kan zorgen. Denk bij een type plant aan een tomaten plant, aardbeien plant, paprika plant, e.d. De actieve plant is de plant waar momenteel voor gezorgd word in de kas.

De volgende onderdelen worden beïnvloed om de plant efficiënt te laten groeien:

* Omgeving temperatuur.
* Vocht niveau van de aarde
* Licht uren

De omgeving temperatuur zal nooit boven de maximum of onder de minimum temperatuur komen. Wanneer dit wel gebeurt zal waarschuwing worden weergegeven.

Er word water gegeven wanneer het vocht niveau van de aarde onder de helft van de maximum waarde komt. Wanneer het vocht niveau onder de helft van het maximum blijft na het water geven word een waarschuwing weergegeven

Aan de hand van de licht uren worden de groei lichten aan of uit gezet zodat de plant de juiste hoeveelheid licht krijgt.

Er zijn 2 gebruikers gedefinieerd, de gebruiker en service. Onder gebruiker valt personeel dat klein onderhoud pleegt aan de installatie en informatie uitleest. Onder service vallen beheerders en service monteurs.

Als gebruiker van deze machine is het mogelijk om:

* De huidige status van de bovengenoemde onderdelen te zien.
* De tijd gesimuleerd te verhogen.
* In te loggen met de service login.

Als service zijn de volgende extra optie mogelijk:

* Een nieuw plant type aanmaken en opslaan.
* De actieve plant veranderen naar één van de opgeslagen plant types.
* De hardware componenten direct aansturen.
* Uitloggen als service en terug gaan naar het gebruikers menu.

Er word een logbestand bij gehouden. In dit logbestand is het volgende te vinden:

* De actieve plant
* Systeem error
* Simulatie systeem info

# Voorwoord

Ik ben Alwin Rodewijk een enthousiaste hardware engineer met een interesse in programmeren. Tijdens dit project heb ik met veel plezier gefocust op de architectuur van dit software project.

Mijn voorkennis bestaat uit hobby projecten waarin ik Arduino als mijn besturingsplatform gebruik. Als hardware engineer kom ik niet vaak in aanraking met het schrijven van software. Daarin tegen schijf ik scripts om mijn werk te automatiseren of vergemakkelijken wanneer ik hiertoe de mogelijkheid zie. Zowel de Arduino projecten als de scripts zijn simpel, kort en missen een structuur.

Tijdens dit project heb ik geleerd om een *design patern*, een state machine, te gebruiken voor het schrijven van gestructureerde code. Ik kan hier veel meerwaarde uit halen aangezien ik tot nu toe moeite had met het schrijven van grotere software projecten. Dit project is hier een voorbeeld van, de hardware is af alleen de software was te groot om zonder structuur te schrijven. Het gebruiken van een state machine heeft mij geholpen om dit project gestructureerd op te zetten. Hierin zie ik veel meerwaarde in mijn toekomstige projecten, zowel professioneel als privé. Met deze ervaring kan ik grote software projecten overzichtelijk houden en efficiënte code schrijven.

Alwin Rodewijk

# Inhoudsopgave

[Samenvatting 3](#_Toc530151323)

[Voorwoord 4](#_Toc530151324)

[1. Inleiding 6](#_Toc530151325)

[2. Definitiefase 7](#_Toc530151326)

[2.1 Kennismaking en doel van …….. 7](#_Toc530151327)

[2.2 Functionele eisen 8](#_Toc530151328)

[2.3 Technische eisen 8](#_Toc530151329)

[2.4 Schets user interface 8](#_Toc530151330)

[3. Ontwerpen 9](#_Toc530151331)

[3.1 Architectuur apparaat 9](#_Toc530151332)

[3.2 State chart apparaat 9](#_Toc530151333)

[4. Realisate en testen 10](#_Toc530151334)

[4.1 Realisatie 10](#_Toc530151335)

[4.2 Acceptatietesten 10](#_Toc530151336)

[5. Eindresultaat en aanbevelingen 11](#_Toc530151337)

[5.1 Eindresultaat 11](#_Toc530151338)

[5.2 Aanbevelingen 11](#_Toc530151339)

# Inleiding

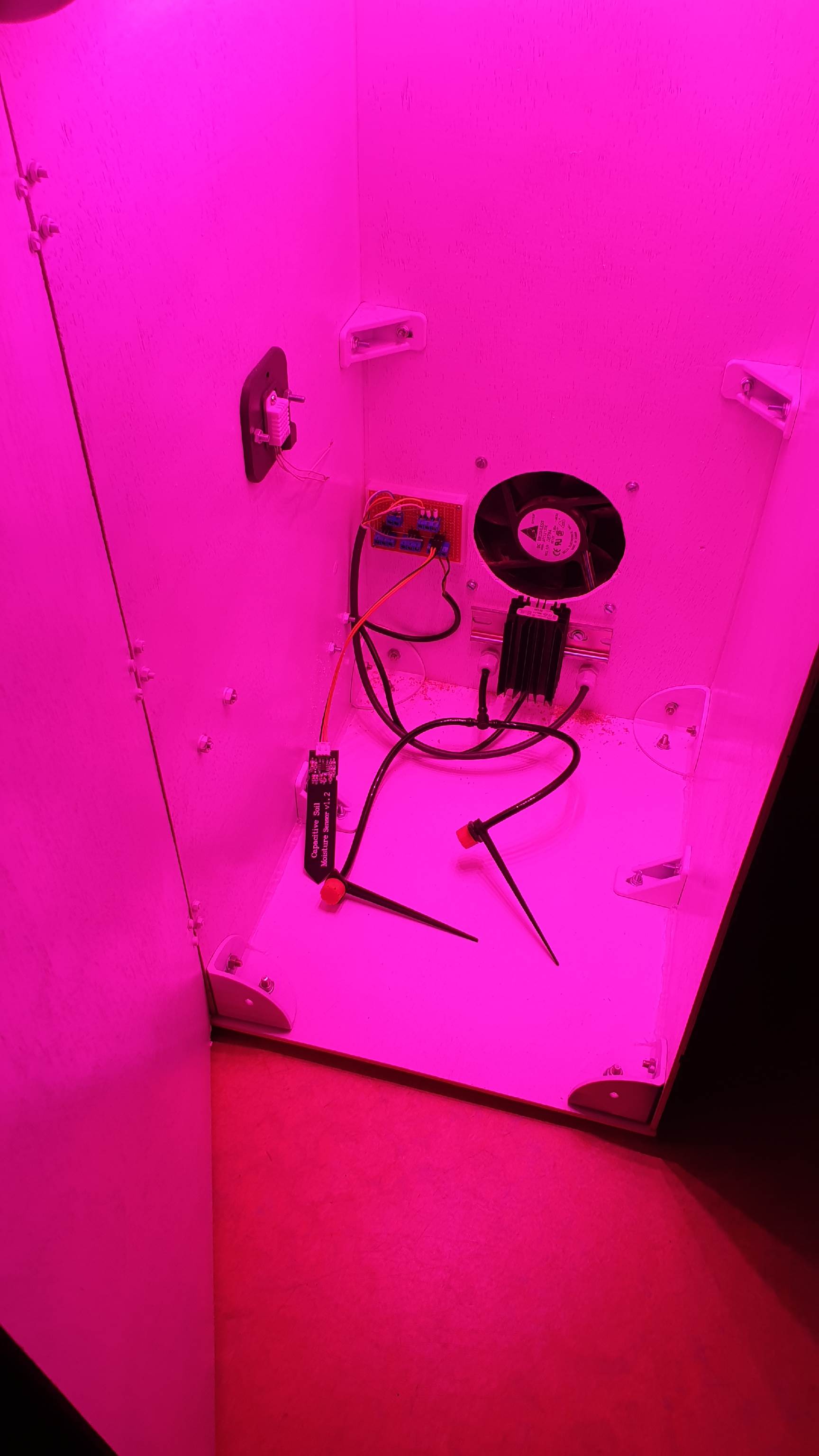
// todo

# Definitiefase

Dit hoofdstuk gaat in op de gestelde eisen aan de te realiseren besturingssoftware van   
Tomato – The Automatic Greenhouse.

## Kennismaking en doel van Tomato - The Automatic Greenhouse

Tomato – The Automatic Greenhouse is een volledig automatische kas waar verschillende type planten in kunnen groeien. Er word rekening gehouden met de licht uren, de vochtigheid van de aarde waar de plant in staat en de temperatuur. Per type plant kunnen deze eigenschappen worden ingesteld. De eigenschappen van de actieve plant zullen worden gebruikt op de kas te besturen. Hierdoor word gewaarborgd dat de plant optimaal groeid.



## Functionele eisen

De functionele eisen zijn als volgt:

1. Er word volledig automatisch voor een plant worden gezorgd.
2. De plant krijgt het aantal licht uren per dag wat gespecificeerd is voor dat type plant.
3. De plant zal voldoende water krijgen. Dit word gegarandeerd door wanneer het vocht niveau onder de helft van het maximum komt de plant water te geven.
4. De temperatuur blijft tussen het minimum en maximum van de actieve plant met behulp van een verwarming element en een temperatuur sensor.
5. Met een service login kan een ander soort plant worden geselecteerd.
6. Met een service login kan een nieuw soort plant worden aangemaakt en opgeslagen.
7. Er word een log bestand gegenereerd om de gebeurtenissen op te slaan. Log bestanden worden opgeslagen en kunnen worden terug gevonden in de ‘files/log’ map met het format ‘hhmm\_DDMMYYY\_log.txt’ als naam.
8. Aangezien de software via het terminal interface word bediend zal de tijd via de ‘time’ actie vooruit worden gezet om de automatische acties uit te voeren. Per 30 minuten zal een controle worden of de installatie een actie moet uitvoeren om de plant correct te verzorgen.
9. De installatie onthoud welke plant de actieve plant is, wanneer het programma word afgesloten en daarna opnieuw gestart word de vorige actieve plant gebruikt als huidige actieve plant.

## Technische eisen

De technische eisen zijn als volgt:

1. De programmeertaal C met compiler ‘MinGW 7.3.0 32-bit for C’ wordt gebruikt.
2. De ontwikkelomgeving ‘Qt Creator 4.11’ wordt gebruikt.
3. De software word met het terminal window bestuurd.
4. Voor het opslaan van plant types word een ‘.csv’ bestand gebruikt.
5. Voor het log bestand word een ‘.txt’ bestand gebruikt.
6. Er is gebruik gemaakt van GIT met het interface SourceTree voor het versie beheer.

## Schets user interface

Het user interface is vormgegeven in het terminal window. Er word een menu weergegeven waarin de huidige waarde van het systeem zijn weergegeven. Het terminal window heeft als gebrek dat weergegeven informatie lastig kan worden geüpdatet.

Door het gebruik van een toetsenbord kan de gebruiker acties uitvoeren. Deze acties zijn worden beschreven in het hoofdstuk *‘****TODO****’*.

Bij elke actie van het automatische systeem word de huidige tijd en de bijbehorende waarde weergegeven in het terminal window.

|  |
| --- |
| C:\Users\Alwin Rodewijk\AppData\Local\Microsoft\Windows\INetCache\Content.Word\20191216_205718.jpg |
| Terminal HMI |

# Ontwerpen

Dit hoofdstuk gaat in op het ontwerp van de software van de Tomato – The automatic greenhouse. Hierbij wordt in het hoofdstuk ‘3.1 Architectuur’ beschreven hoe de software is ontworpen. In het hoofdstuk ‘3.2 State chart’ wordt het ontwerp van de state machine weergegeven.

## Architectuur

De volgende subsystemen zijn gebruikt voor dit systeem.

* Display
* Keyboard
* Finite State machine
* Water control
* Light control
* Temperature control
* Plant manager
* Time manager
* File manager

*//* ***TODO: plaatje UML layer***

Display is verantwoordelijk voor het weergeven van informatie op het terminal interface. Vanuit Display wordt ook naar het log bestand geschreven bij ‘simulation System info’ en ‘show System error’.

Keyboard is verantwoordelijk voor het ophalen van de gebruiker input. Dit wordt gedaan in de vorm van tekst een numerieke waarde of een gebruiker/service actie.

Finite State Machine is verantwoordelijk voor bijhouden van de *state* waarin de installatie momenteel verkeerd en het afhandelen van *events*.

Water control is verantwoordelijk voor het monitoren van de vochtigheid van de grond, water geven aan de plant en de water pomp aan of uit zetten.

Light control is verantwoordelijk voor het aan en uit zetten van de groei lampen.

Temperature controle is verantwoordelijk voor het monitoren van de temperatuur en het aan of uit zetten van de verwarming.

Plant manager is verantwoordelijk voor het bijhouden van de huidige actieve plant.

Time manager is verantwoordelijk voor het bijhouden van de tijd die gebruikt word in deze installatie om de automatische actie uit te voeren.

File manager is verantwoordelijk voor het maken van het logbestand, het toevoegen van een nieuwe plant aan het ‘plant.csv’ bestand en het opslaan/uitlezen van de huidige actieve plant.

## State chart

Het onderstaande diagram beschrijft de werking van de installatie.

<UML + figuur titel + nr>

# Realisate en testen

Dit hoofdstuk bevat de omschrijving van de realisatie en acceptatietesten van dit project.

## Realisatie

Er is gebruik gemaakt van QT Creator 4.10 als programmeer omgeving.

Er is gebruik gemaakt van Plant\_UML voor het maken van de UML diagrammen.

Er is gebruik gemaakt van DoxyGen voor het genereren van de documentatie van de code.

In bijlage 1 ***// TODO: gebruikers handleiding*** vind u de handleiding die bij deze installatie hoort.

***// TODO: inhoud .pro file***

De code in modulair opgezet in verschillende header en .c files:

* main.c
* events.h
* fms.h / .c
* display.h / .c
* keyboard.h / .c
* time\_manager.h / .c
* file\_manager.h / .c
* plant\_manager.h / .c
* water\_control.h / .c
* temperature\_control.h / .c
* light\_control.h / .c

De folder structuur is als volgt:

* Tomato/
  + <Software files>
  + files/
    - log/
      * <gegenereerde log files>
    - Plants/
      * activePlants.txt
      * plants.csv
* Tomato\_UML/
  + <bron locatie UML bestanden>
  + <afbeelding bestanden UML>
* Tomato\_docs/
  + <Doxygen files>
* Productdocument-v1.0.docx

De volgende onderdelen zijn niet gemaakt.

De volgende bekende bugs bestaan momenteel nog in dit project.

## Acceptatietesten

Met deze testen word gecontroleerd of de geproduceerde software functioneert zoals in dit document beschreven.

De hieronder staande testen zijn uitgevoerd in QT Creator 4.10.0 in de volgorde zoals hier onder beschreven. Deze testen zijn uitgevoerd door Alwin Rodewijk.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test** | **Action/input** | **Expected result** | **Pass Fail** | **Actual result if test has failed** |
| 1. | Test help |  |  |  |
| 2. | Test time |  |  |  |
| 3. | Test update |  |  |  |
| 4. | Test service |  |  |  |
| 5. | Test light, heater, pump, water |  |  |  |
| 6. | Test add |  |  |  |
| 7. | Test change |  |  |  |
| 8. | Test user |  |  |  |
| 9. | Test close, restart, check actie plant |  |  |  |

<Samenvattende conclusie(s) over de testen, meer testen nog te doen? Waarom niet gedaan? >

# Eindresultaat en aanbevelingen

Dit hoofdstuk bevat een omschrijving van het eindresultaat van dit project. Waarna er aanbevelingen worden gedaan voor het verder ontwikkelen van dit product.

## Eindresultaat

<moet redelijk op zichzelf leesbaar zijn>

….

## Aanbevelingen

….

<wat nog te doen? verbeteringen? Bugs oplossen? Volgende versie? ….>

# Bijlage 1 – Gebruikers handleiding