Super Thermostat Project

Team Smart Thermo

“Smarter everyday”

Team members:

A. Roman

A. Slotboom

E. Thissen

Goal: create a working thermostat application based on the client wishes.

Rotterdam

20 04 2018

Document V 1.2

# Definition of Done

From the scrum.org glossary:

Definition of Done: a shared understanding of expectations that the Increment must live up to in order to be releasable into production. Managed by the Development Team.

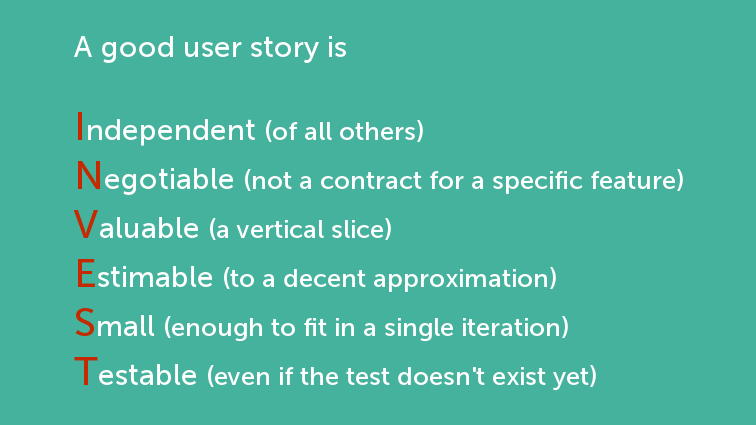
The following are our requirements for the acceptance of a user story as done:

1. Code builds without warnings (technical task)[[1]](#footnote-1)
2. Code unit tested (technical task)
3. Code peer reviewed
4. Story is functionally tested
5. Integration testing
6. Documentation updated (user story)
   1. Move user story to done
7. Build pushed to demo server (user story)
   1. Build and test with new user story
   2. Build can be showed during a demo

# Definition of Ready

The definition of ready for this project is based on the following image from themanifesto.co.uk website.

1. **Technical story requirements**: which technical blocks are needed for a specific story
2. **Same format**: all user stories should use the same format.
3. **Independent**: the user story is a stand-alone story, meaning it is not dependent on other stories to be able to be built in the current sprint.
4. **Negotiable**: user story can be changed if the team feel it is not adequate.
5. **Valuable**: user story add value to the product.
6. **Estimable**: user story can be estimated up to a certain degree of certainty. If user story is too big it need to be split into multiple user stories.
7. **Small**: user story can be built in 1 sprint.
8. **Testable**: user story is testable in the build.
9. **Clear**: user story must be understandable by the whole team.



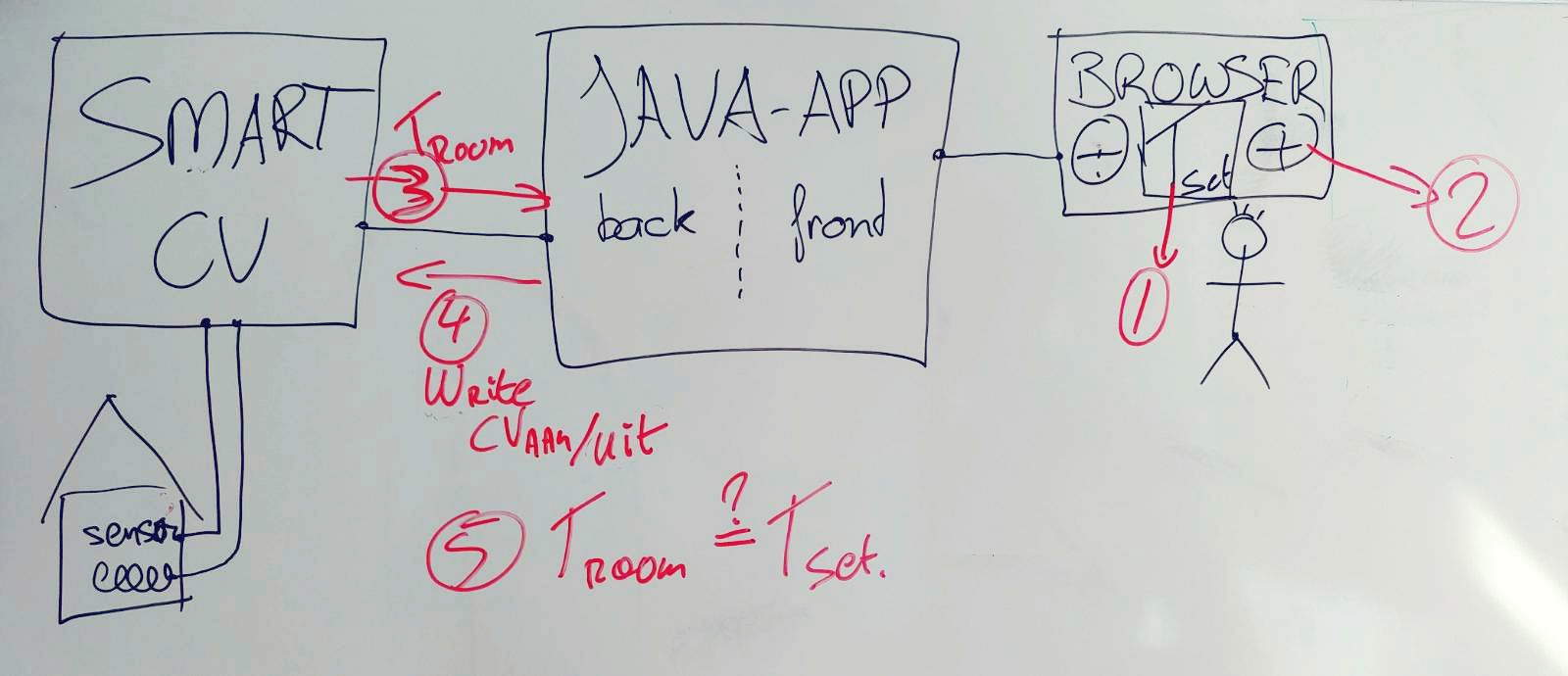
Figuur 1 <https://manifesto.co.uk/the-definition-of-ready/>

# Epics

1. **Need to have (MVP):**
   1. Change temperature as user
   2. View temperature as user
   3. Display current temperature (every 60 seconds)
   4. Sub Set max and min temperature limits
   5. Keep temperature constant as desired (% pump and burner)
   6. Security measure: stop warming when temperature does not increase after X time
2. **Nice to have:**
   1. Ability to program the desired temperature at a certain time
   2. Set thermostat activity based on user movements
   3. Sub Change temperature as admin
   4. Sub View all lodge’s temperature as admin (with lodge numbers)
   5. --Wireframe (Make UI thermostat user)
   6. --Wireframe (Make UI thermostat admin)
   7. View CO2 footprint as user
   8. View CO2 footprint as admin
   9. View gas m3 usage as user
   10. View gas m3 usage as admin
   11. View cost (yearly) in euro as user
   12. View cost (yearly) in euro as admin
   13. View CO2 footprint compared to average (user is either above or below)
   14. View graph with temperature as user
   15. View graph with temperature as admin
3. **Tech requirements:**
   1. Set communication between thermostat and database
   2. Set communication between user environment and database
   3. Set communication between admin environment and database
   4. Set up server
   5. Sub Set time interval for temperature logging (log every 1 hour, total 24 hours)
   6. Log output data from thermostat to database(current temperature, status etc.)

Admin = fase 3

# User stories



Figuur 2 Program schematic and user stories as numbers

**Dev story: set up spring boot framework**

**Estimation: 8 points = +- one day of work**

1. **User story 001: park visitor can view set temperature**
   1. **As a**: park visitor
   2. **I want**: to view the set temperature
   3. **So that**: the set temperature can be seen
   4. **Technical story requirements:** front end, back end
   5. Acceptance criteria
      1. **I know this is done when**: I can view the set temperature
   6. **Estimation**: 3 points
2. **User story 002: park visitor can set the temperature**
   1. **As a**: park visitor
   2. **I want**: to be able to set the temperature
   3. **So that**: the actual room temperature becomes what I set it to
   4. **Technical story requirements:** Communication between front-end, back-end. Communication between back-end and Smart CV. User input. Pressure must always be between 1.5 and 2.0 bar.
   5. Acceptance criteria
      1. **I know this is done when**: I change the temperature and the set temperature changes and stays constant
   6. **Estimation:** 24 points
   7. **TASKS:**
      1. Application can read the current temperature
      2. Application can turn the cv on or off
      3. Application can control the room temperature
3. **User story <number>: <type of user/component><concise action>**
   1. **As a**: <type of user/component>
   2. **I want**: <desired action>
   3. **So that**: <desired effect>
   4. **Technical story requirements:** <all requirements>
   5. Acceptance criteria
      1. **I know this is done when**: <action> <effect>
   6. **Estimation**: <points>

# Questions

1. Efficiency of pump and burner in the element, how much % each for optimal use? Which ratio and relation.
2. What is the ‘gas use’ in the emulator? Is it cumulative or rate of use?
3. Gas use as Euros? Which price do we use?
4. Is the ability to time program the thermostat by the user a must have? Fase 2
5. Application server needed or only Java standalone executable? Which server is advisable? Spring boot quickstart vinden. Otherwise applicatie server.

1. <https://www.atlassian.com/blog/jira-software/8-steps-to-a-definition-of-done-in-jira> [↑](#footnote-ref-1)