[Functioneel ontwerp](https://sites.google.com/view/rijnijssel-ao/sjablonen/25187/algemeen?authuser=0#h.p_Ty7aduG2GfRB)

# [Versiebeheer](https://sites.google.com/view/rijnijssel-ao/sjablonen/25187/algemeen?authuser=0" \l "h.p_eKG6nG44Kd0X)

|  |  |  |  |
| --- | --- | --- | --- |
| Datum | Versie | Wie | Wijzigingen |
| 20-10-2022 | 1.0 | David Hanna | Document opstellen |
| 20-10-2022 | 1.1 | Joep Zomerveld | Functional Requirements toevoegen |
| 20-10-2022 | 1.2 | David Hanna | Domain en use case diagram toegevoegd |
| 06-11-2022 | 1.3 | David Hanna | Fully dressed use cases 1 tot 4 gemaakt en verbeterde domain diagram toegevoegd |

Inhoudsopgave

[Versiebeheer 2](#_Toc117170105)

[Inleiding 4](#_Toc117170106)

[Over dit document 4](#_Toc117170107)

[Over het project en de opdrachtgever 4](#_Toc117170108)

[Requirements 5](#_Toc117170109)

[Functional 5](#_Toc117170110)

[Non-functional 5](#_Toc117170111)

[Domain diagram 6](#_Toc117170112)

# [Inleiding](https://sites.google.com/view/rijnijssel-ao/sjablonen/25187/algemeen?authuser=0#h.p_nScE_Y3RO7c3)

## Over dit document

## Over het project en de opdrachtgever

# Requirements

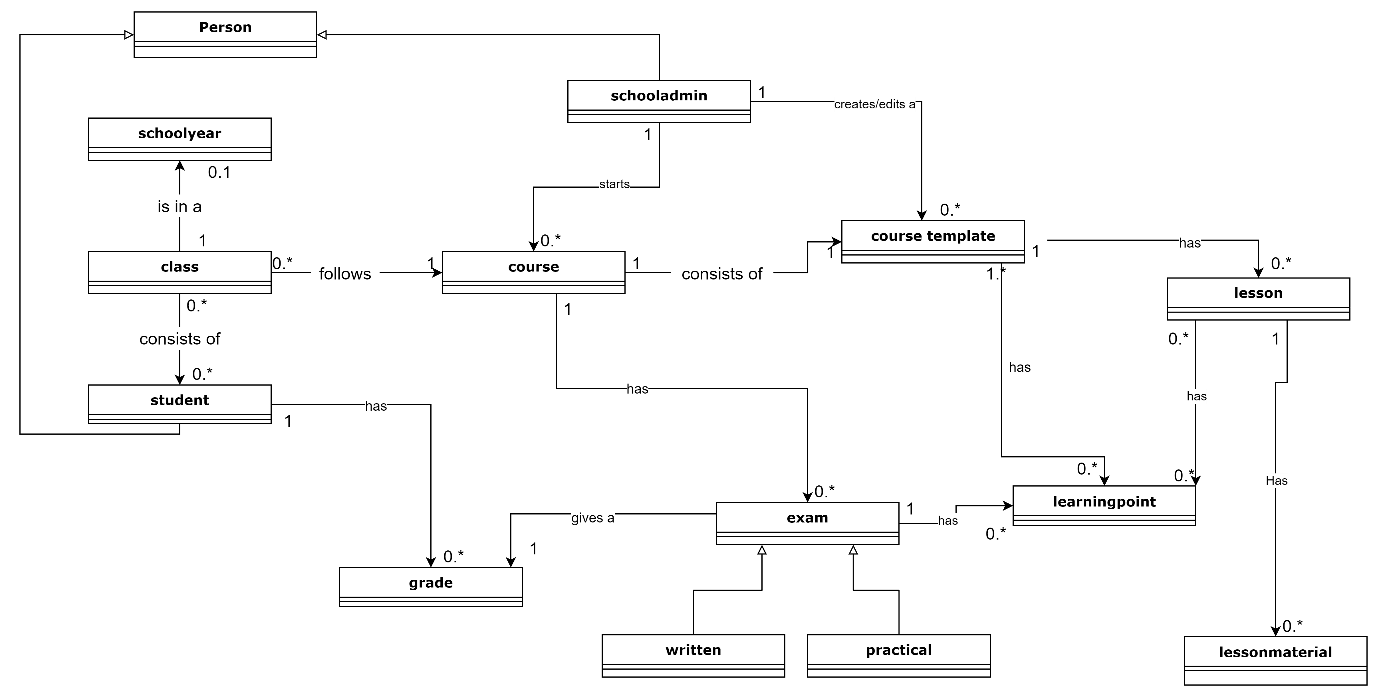
## Functional

1. administrator kan course templates maken en beheren
2. administrator kan een course vanuit een template inplannen (datum en klas toevoegen)
3. een administrator kan (en moet) controleren of de course aan voorwaarden voldoet bij het inplannen
4. een administrator kan lesmaterialen in courses en course templates uploaden en beheren bij elke les
5. een administrator kan leerlingen maken en aan een klas toevoegen
6. een gebruiker kan zijn rooster inzien
7. een leerling gebruiker kan zijn cijferlijst inzien
8. een leraar gebruiker kan cijferlijsten beheren
9. een leerling gebruiker moet de lessen van een course die hij volgt kunnen zien
10. een leerling gebruiker moet de lesmaterialen kunnen downloaden

## Non-functional

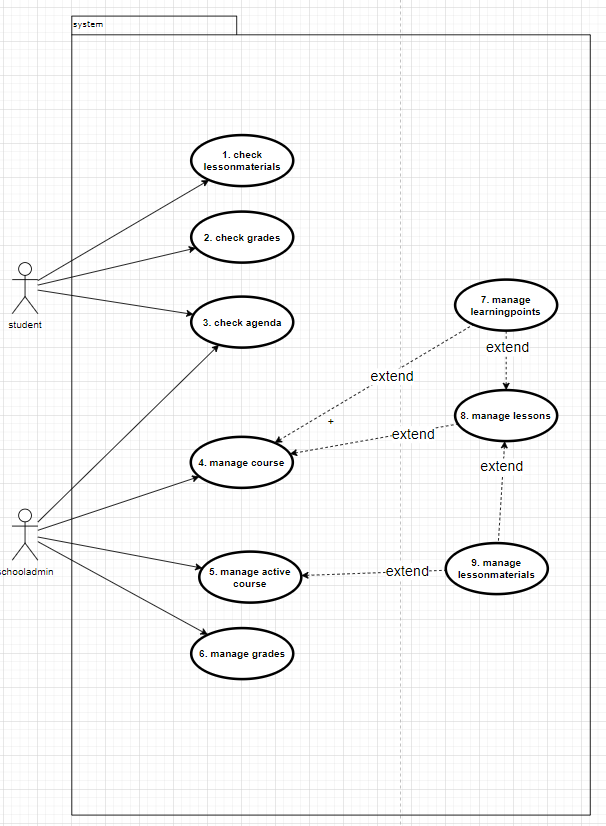
1. Compatability in de naam van student gegevens moeten gekoppeld kunnen worden met extrene omgevingen.

# Domain diagram



# Use cases

## Use case diagram



## Cases

### Use Case: 1. Check lesson materials

|  |  |
| --- | --- |
| Primary actor: student | |
| Stakeholders and Interests: student who wants to check the materials available for a lesson | |
| Cross references: | |
| Brief description:  Students check materials that are available for the lessons that they follow | |
| Preconditions:  Actor knows which course and lesson he wants to check the materials from | |
| Postconditions on success:  Student has checked the desired lesson materials  Postconditions on failure:  Student has no courses or materials for the lesson | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on the course where he wants to check the lesson materials from  3.Actor clicks the desired lesson | 2.system shows actor’s lesson for specified course  4.system shows actor’s lessonmaterials for specified lesson |
| Alternate flows: (Variations with successful UC) | |
|  |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
| 1A.Actor has not been added to any course yet | 2A.actor has no lessons for specified course  4A.no lesson materials have been added to the specified lesson |

### Use Case: 2. Check grades

|  |  |
| --- | --- |
| Primary actor: student | |
| Stakeholders and Interests: student who wants to check his grades | |
| Cross references: | |
| Brief description:  Students checks grades that he has received for his courses | |
| Preconditions:  Actor knows which grades he wants to check | |
| Postconditions on success:  Student has checked grades  Postconditions on failure:  Student has no grades | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on the course where he wants to check the grades  3.Actor checks his grades | 2.system shows actor’s grades of the specified course |
| Alternate flows: (Variations with successful UC) | |
|  |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
| 1A.Actor has not been added to any course yet | 2A.actor has no grades for specified course |

### Use Case: 3. Check agenda

|  |  |
| --- | --- |
| Primary actor: student | |
| Stakeholders and Interests: student who wants to check his agenda | |
| Cross references: | |
| Brief description:  Students checks his agenda to see which lessons he has and when/where they take place | |
| Preconditions: | |
| Postconditions on success:  Student has knows when/where his upcoming lessons are  Postconditions on failure:  Student has no upcoming lessons | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on the agenda button  3.Actor checks his agenda | 2.system shows actor’s agenda with lesson details |
| Alternate flows: (Variations with successful UC) | |
|  |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
|  | 2A.there are no upcoming lessons |

### Use Case: 4. Manage course

|  |  |
| --- | --- |
| Primary actor: schooladmin | |
| Stakeholders and Interests: schooladmin creates a course | |
| Cross references: | |
| Brief description:  Schooladmin creates a course | |
| Preconditions: | |
| Postconditions on success:  Course is created  Postconditions on failure:  Course is not created | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on the create course button  3.Actor fills course title in.  4.Actor fills course description in  5.Actor adds lessons to course.  6.Actor adds exams to course.  7.Actor adds learningpoints to lessons and exams  8.Actor clicks save | 2.system shows course creation page  9.system creates new course |
| Alternate flows: (Variations with successful UC) | |
| 5A.Actor does not add lessons to course  6A.actor does not add exams to course.  7A. Actor does not add learningpoints to lessons and exams |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
| 8A.Actor does not click save |  |

|  |  |
| --- | --- |
| Primary actor: schooladmin | |
| Stakeholders and Interests: schooladmin edits a course | |
| Cross references: | |
| Brief description:  Schooladmin edits a course | |
| Preconditions: Actor knows which course he wants to change | |
| Postconditions on success:  Course is edited  Postconditions on failure:  Course is not edited | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on the edit course button  3.Actor specifies preferred course  5.Actor changes course details  6.Actor clicks save | 2.system shows list of available courses  4.system shows details of specified course  7.system saves course |
| Alternate flows: (Variations with successful UC) | |
|  |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
| 5A.Actor does not change course details  6A.Actor does not click save |  |

|  |  |
| --- | --- |
| Primary actor: schooladmin | |
| Stakeholders and Interests: add class to course | |
| Cross references: | |
| Brief description:  Schooladmin adds a course to a class | |
| Preconditions: Actor knows which course he wants to assign to which class | |
| Postconditions on success:  Course is added to class agenda  Postconditions on failure:  Course is not added to class | |
| Main Success Scenario (Basic Flow): | |
| Actor Action | System Responsibility |
| 1.Actor clicks on preferred class  3.Actor clicks preferred course  5.Actor specifies start date  6.Actor clicks save | 2.System shows which courses are available for the class  4.System checks if course details are complete  7.System adds course to class |
| Alternate flows: (Variations with successful UC) | |
|  |  |
| Exceptional flows: (UC failure leading to “post condition on failure”) | |
| 6A.Actor does not click save | 2A.no courses available  4A.Course details are not complete |