

Metadata

Titel: Probleemanalyse - Project Studentenhuisvesting

Versie: 1.1

Auteur: Bram Wieringa

Afhankelijkheid: -

Probleemanalyse - Project Studentenhuisvesting

1. Projectbeschrijving

Projectnaam: Studentenhuisvesting

Doel: Ontwikkelen van een IT-systeem dat de belangrijkste operationele problemen van Student Housing B.V. oplost en schaalbaar is voor toekomstige uitbreidingen.

Hoofdvraag: Hoe kan een nieuw IT-systeem het klachtenproces, onderhoudsbeheer en huishoudelijke organisatie binnen studentenhuisvesting efficiënter en inzichtelijker maken?

2. Contextanalyse

Student Housing B.V. beheert meerdere studentencomplexen met gedeelde voorzieningen. Het huidige systeem is verouderd en ondersteunt de processen onvoldoende. Het nieuwe systeem moet:

- het klachten- en onderhoudsproces optimaliseren;
- ondersteuning bieden voor huishoudelijke organisatie;
- inzicht geven in kosten en leefomstandigheden;
- starten met een haalbare MVP;
- werken met CSV-data afkomstig uit het oude systeem.

2.1 Overzicht van uitdagingen

Operationele problemen

- Onvoldoende naleving van schoonmaaktaken
- Vergeten of onbetaalde gezamenlijke boodschappen
- Afval wordt niet tijdig afgevoerd
- Ongemelde feestjes of overlast
- Klachten over verwarming/airconditioning
- Storingen of slecht onderhoud van brandalarmen
- Geen goede traceerbaarheid van onderhoudsdiensten

Nieuwe mogelijkheden voor de toekomst

- Delen van studiehulpmiddelen
- Boodschappen- en voorraadbeheer
- Monitoring van luchtkwaliteit
- Smart-home integraties
- Community- en eventfunctionaliteiten

Kostenreductie en inzicht

- Inzicht in energieverbruik
- Betere onderhoudsplanung
- Inzicht in leefomstandigheden
- Betere studentplaatsing
- Minder schadegevallen aan eigendommen van Student Housing B.V.

3. Probleemanalyse

3.1 Scope

Binnen scope

Het project omvat het ontwikkelen van een webgericht IT-systeem dat:

- toegankelijk is voor studenten, beheerders, interne onderhoudsmedewerkers en externe onderhoudspartners;
- huishoudelijke organisatie ondersteunt (takenrooster, afvalmomenten, schoonmaaktaken);
- het klachtenproces automatiseert (registratie, opvolging, status, feedback);
- onderhoudsbeheer structureert (planning, uitvoering, registratie);
- CSV-data eenmalig kan importeren;
- rol- en rechtenstructuren implementeert;
- basisrapportages biedt;
- werkt binnen een veilig modelnetwerk voor studentenhuizen.

Buiten scope

- Smart-home integraties (sensoren, Arduino)
- Communityfunctionaliteiten (events, delen van spullen, voorraadbeheer)
- Integraties met externe platforms
- Organisatorische veranderprocessen binnen Student Housing B.V.

3.2 Doelgroepen

Studenten

Studenten wonen in de studentenhuizen die Student Housing B.V. aanbiedt. Zij hebben behoefte aan een overzichtelijk systeem waarmee zij klachten kunnen melden, de status van meldingen kunnen volgen en inzicht hebben in de afhandeling. Daarnaast willen zij duidelijkheid over huishoudtaken: wie welke taak uitvoert, wanneer dit moet gebeuren en wat al is gedaan.

Beheerders

Beheerders zijn verantwoordelijk voor het dagelijkse reilen en zeilen binnen de studentenhuizen. Zij moeten inzicht hebben in alle openstaande en afgeronde klachten, de planning en uitvoering van huishoudtaken, de onderhoudsstatus en operationele kosten. Het systeem moet hen ondersteunen bij coördinatie, besluitvorming en communicatie met studenten en onderhoudspartijen.

Onderhoudsmedewerker

Onderhoudsmedewerkers, zowel intern als extern, voeren reparaties en onderhoudstaken uit binnen de studentenhuizen. Zij hebben behoefte aan een duidelijke en actuele takenlijst, met per taak de locatie, omschrijving, prioriteit, benodigde materialen en eventuele veiligheidsinstructies. Het systeem moet hen helpen om werkzaamheden efficiënt en traceerbaar uit te voeren.

Onderhoudsmanager

De onderhoudsmanager is verantwoordelijk voor de planning, prioritering en toewijzing van onderhoudstaken. Deze rol vereist inzicht in de totale onderhoudsbehoefte, beschikbare capaciteit, kostenramingen en materiaalgebruik. De onderhoudsmanager moet eenvoudig taken kunnen aanmaken, prioriteiten kunnen wijzigen, voortgang kunnen monitoren en rapportages kunnen genereren.

3.3 Beperkingen en voorkeuren

- CSV-data van onbekende kwaliteit
- Team heeft beperkte ervaring met softwareontwikkeling
- Hardware (Arduino) alleen optioneel na MVP-fase
- Voorkeur voor duidelijke ICT-infrastructuur binnen het project

3.4 Functieanalyse (MoSCoW)

Must-have

- Webapplicatie met rolgebaseerde toegang
- Klachtenregistratie (melden, status, feedback, toewijzing)
- Huishoudrooster (taken aanmaken, verdelen, afvinken)
- Onderhoudsbeheer (taakdetails, toewijzing, registraties)
- Eenmalige CSV-import

- Basisdashboard voor klachten, onderhoud en huishoudtaken
- Veilig modelnetwerk voor studentenhuis {lorum ipsum}

Should-have

- Communicatiefunctie (reacties, notificaties)
- Uitgebreidere onderhoudsflows (prioriteiten, teams)
- Validatierapportage op CSV-data
- Gebruikersprofielen

Could-have

- Documentupload
- Kalenderweergave
- Uitgebreide dashboards
- Tags en thema's voor klachten

Won't-have

- Smart-home integraties
- Communityfunctionaliteiten
- Externe integraties
- Organisatorische verandertrajecten

3.5 Use Case (Pixar Pitch)

Er was eens een studentenhuis dat werd beheerd door Student Housing B.V. Iedere dag worstelden studenten met klachten melden en huishoudtaken organiseren. Op een dag besloot Student Housing B.V. een webportaal te ontwikkelen. Daardoor konden studenten eenvoudig klachten indienen en huishoudtaken volgen. Tot op een dag waren de problemen opgelost en woonden de studenten prettiger en overzichtelijker samen.

4. Projectplanning

Elke sprint duurt twee weken.

- 10 november — Start groepsproject
- 24 november — Sprint 0 opleveren
- 7 december — Portfolioreview 3
- 8 december — Sprint 1 opleveren
- 22 december — Sprint 2 opleveren
- 5 januari — Sprint 3 opleveren
- 12 januari — Conclusie schrijven en opleveren

5. Technologische analyse

Zie de bijlage Technische analyse

6. Agile-setup en teamafspraken

6.1 Definition of Done

Een item is “done” wanneer:

- functionaliteit volledig werkt volgens de user story;
- code getest is;
- documentatie in GitHub is bijgewerkt;
- reviewer akkoord is;
- functionaliteit getoond kan worden in de sprintreview.

6.2 Definition of Ready

Een item is “ready” wanneer:

- de user story compleet is volgens het **INVEST-principe** (Independent, Negotiable, Valuable, Estimable, Small, Testable);
- acceptatiecriteria duidelijk en testbaar zijn;
- impact op database, rollen en UI besproken is;
- dependencies bekend zijn;
- story uitvoerbaar is binnen één sprint.

6.3 Tools

Zie de bijlage Technische analyse

6.4 Samenwerkingsafspraken

Scrumrollen

- Per sprint wordt één Scrum Master aangewezen.
- De groep vertegenwoordigt samen de Product Owner.

Communicatie

- WhatsApp: korte berichten
- Trello: sprintplanning
- Microsoft Teams: vaste weekly op donderdag 17:30
- GitHub: code en documentatie

Documentatie

- Actiepunten worden genoteerd in Trello
- Notulist rouleert
- Documentatie volgt:
Analyse – Adviseer – Design – Realisatie – Manage & Control

Oplevermomenten

- Elke sprint: review + retrospectief

Besluitvorming

- Consensus waar mogelijk
- Anders stemmen
- Bij gelijk: muntje opgooien

7. Initiële Product Backlog (MVP)

Hieronder staan de user stories, herschreven en gestructureerd.

7.1 User Stories (INVEST)

US-01a – Toegang tot portaal

Als gebruiker

wil ik kunnen inloggen op een online portaal

zodat ik toegang heb tot alle functionaliteiten van het studentenhuus.

Acceptatiecriteria:

- Inloggen met rol (Student, Beheerder, Onderhoudsmedewerker, Onderhoudsmanager)
- Onjuist wachtwoord geeft foutmelding
- Rolgebaseerde toegang: gebruiker ziet alleen wat relevant is

MoSCoW: Must-have

US-01b – Navigatie en dashboard

Als ingelogde gebruiker

wil ik een overzichtelijk dashboard

zodat ik snel kan zien welke taken, klachten of meldingen relevant zijn voor mij.

Acceptatiecriteria:

- Dashboard toont relevante modules per rol
- Klikken op een module opent de juiste pagina

- Overzichtelijk en eenvoudig te begrijpen

MoSCoW: Must-have

US-02a – Klacht indienen

Als student

wil ik een klacht kunnen indienen via een formulier

zodat ik problemen snel kan melden.

Acceptatiecriteria:

- Formulier bevat categorieën van veelvoorkomende problemen
- Mogelijkheid tot extra toelichting
- Bevestiging na indienen

MoSCoW: Must-have

US-02b – Klachtstatus volgen

Als student

wil ik de status van mijn ingediende klacht kunnen volgen

zodat ik transparantie heb over de afhandeling.

Acceptatiecriteria:

- Status zichtbaar: Open, In behandeling, Afgerond
- Toegewezen medewerker zichtbaar
- Historie van acties zichtbaar

MoSCoW: Must-have

US-03 – Klachten beheren

Als beheerder

wil ik klachten kunnen inzien, toewijzen, prioriteren en feedback geven

zodat klachten efficiënt worden opgevolgd.

Acceptatiecriteria:

- Lijst van openstaande klachten beschikbaar
- Toewijzen aan onderhoudsmanager of medewerker mogelijk
- Prioriteit instellen (Laag, Midden, Hoog)
- Reacties toevoegen die student kan zien
- Status aanpassen

MoSCoW: Must-have

US-04 – Huishoudrooster beheren

Als beheerder

wil ik huishoudtaken kunnen aanmaken, verdelen en aanpassen

zodat het schoonmaakrooster eerlijk wordt bijgehouden.

Acceptatiecriteria:

- Taken aanmaken met omschrijving, locatie, frequentie, student toegewezen
- Taken verdelen handmatig of automatisch
- Aanpassen of verwijderen van taken
- Studenten zien hun taken in rooster

MoSCoW: Must-have

US-05 – Huishoudtaken afvinken

Als student

wil ik mijn toegewezen huishoudtaken kunnen afvinken

zodat iedereen kan zien wat is gedaan.

Acceptatiecriteria:

- Taak markeren als uitgevoerd
- Status real-time zichtbaar voor alle gebruikers
- Historie beschikbaar voor beheerder

MoSCoW: Must-have

7.3 Toelichting backlog

- Alle stories zijn **INVEST-compliant**: klein, testbaar, waardevol, inschatbaar en onafhankelijk waar mogelijk.
- De **MVP-focus** ligt op de absolute basisfunctionaliteiten, zonder smart-home integraties, communityfunctionaliteiten of externe koppelingen.
- Stories kunnen later worden uitgebreid naar **Should-have** en **Could-have** functionaliteiten zoals notificaties, documentupload, uitgebreide dashboards en kalenderweergave.

Conclusie

Het project *Studentenhuysvesting* richt zich op het ontwikkelen van een schaalbare, webgebaseerde IT-oplossing die de belangrijkste operationele knelpunten binnen studentencomplexen van Student Housing B.V. oplost. De analyse laat zien dat vooral het klachtenproces, het onderhoudsbeheer en de huishoudelijke organisatie momenteel inefficiënt verlopen door het gebruik van verouderde systemen en beperkte traceerbaarheid.

Met de gekozen scope, duidelijke rolverdeling en een MVP die zich focust op de essentiële functionaliteiten, klachtenregistratie, huishoudtaken en onderhoudsbeheer, wordt een haalbaar en waardevol fundament gelegd. Door gebruik te maken van user stories die voldoen aan het INVEST-principe, een heldere Definition of Ready en Definition of Done, en een Agile-werkwijze met vaste sprintmomenten, ontstaat een structuur die zowel de kwaliteit van het eindproduct als de samenwerking binnen het team ondersteunt.

Het project creëert niet alleen een oplossing voor de huidige problemen, maar legt ook een basis voor toekomstige uitbreidingen. Daarmee vormt de MVP een solide eerste stap richting een modern, gebruiksvriendelijk en betrouwbaar IT-systeem dat de leefomgeving van studenten verbetert en het beheer voor Student Housing B.V. aanzienlijk efficiënter maakt.

Bronnenlijst (APA-stijl)

OpenAI. (2025, 9 november). *Hoe ziet een sprintplanning eruit?* ChatGPT.

<https://chat.openai.com>

OpenAI. (2025, 10–24 november). *Prompt gebruikt voor taalkundige verbetering.* ChatGPT.

<https://chat.openai.com>

Schwaber, K., & Sutherland, J. (2020). *The Scrum Guide*. <https://scrumguides.org>

Visual Paradigm. (z.d.). *Definition of ready in Scrum*. [https://www.visual-](https://www.visual-paradigm.com/scrum/what-is-definition-of-ready-in-scrum/)

[paradigm.com/scrum/what-is-definition-of-ready-in-scrum/](https://www.visual-paradigm.com/scrum/what-is-definition-of-ready-in-scrum/)

Visual Paradigm. (z.d.). *Definition of done vs acceptance criteria*. [https://www.visual-](https://www.visual-paradigm.com/scrum/definition-of-done-vs-acceptance-criteria/)

[paradigm.com/scrum/definition-of-done-vs-acceptance-criteria/](https://www.visual-paradigm.com/scrum/definition-of-done-vs-acceptance-criteria/)

Visual Paradigm. (z.d.). *Write user story SMART goals*. [https://www.visual-](https://www.visual-paradigm.com/scrum/write-user-story-smart-goals/)

[paradigm.com/scrum/write-user-story-smart-goals/](https://www.visual-paradigm.com/scrum/write-user-story-smart-goals/)

huishouden%%%

- Vergeten of onbetaalde gezamenlijke boodschappen
- Afval wordt niet tijdig afgevoerd
- Ongemelde feestjes of verstoringen
- Klachten over verwarming/airco
- Storing of slecht onderhoud aan brandalarmen
- Geen goede traceerbaarheid van onderhoudsdiensten

Nieuwe mogelijkheden (toekomst)

- Delen van studiehulpmiddelen
- Voorraadbeheer en boodschappenmodule
- Monitoring van luchtkwaliteit
- Smart-home integraties
- Event- en communityfunctionaliteiten

Kostenreductie en inzicht

- Inzicht in energieverbruik
- Betere onderhoudsplanning
- Inzicht in leefomstandigheden
- Betere studentplaatsing
- Minder schadegevallen van studenten aan eigendommen Student Housing B.V.

3. Probleemanalyse

3.1 Scope

Binnen scope

- Ontwikkelen van een IT-systeem dat:
 - Via een website benaderbaar is
 - Ondersteuning biedt voor huishoudelijke organisatie
 - Huishoudrooster (schoonmaken, afval buiten zetten, boodschappen doen)
 - Klachtenproces van studenten automatiseert
 - melding, opvolging, feedback
 - Onderhoudsbeer automatiseert
 - planning, uitvoering, registratie
- Starten met een MVP (Minimal viable product)
- Oude data (CSV-export) is verwerkt in het systeem, eenmalige import
- Rollen en rechten
 - Student
 - Beheerder
 - Interne onderhoudsmedewerker
 - Extern onderhoudspartner
- Basisrapportage / Basis dashboard voor kosten, gebruik en leefomgeving
- Website
- Modelnetwerk voor studentenhuus
 - Security heel belangrijk

Buiten scope

- Geavanceerde smart home integratie en sensortechnologie
- Communityfunctionaliteit
 - Evenementen
 - Studiehulpmiddelen delen
 - voorraadbeheer
- Integratie met externe systemen
- Veranderingsproces bij Student Housing B.V.

3.2 Doelgroep / Gebruikers

Studenten

Studenten wonen in de studentenhuizen die Student Housing B.V. aanbiedt. Zij hebben behoefte aan een overzichtelijk systeem waarmee zij klachten kunnen melden, de status van meldingen kunnen volgen en inzicht hebben in de afhandeling. Daarnaast willen zij duidelijkheid over huishoudtaken: wie welke taak uitvoert, wanneer dit moet gebeuren en wat al is gedaan.

Beheerders

Beheerders zijn verantwoordelijk voor het dagelijkse reilen en zeilen binnen de studentenhuizen. Zij moeten inzicht hebben in alle openstaande en afgeronde klachten, de planning en uitvoering van huishoudtaken, de onderhoudsstatus en operationele kosten. Het systeem moet hen ondersteunen bij coördinatie, besluitvorming en communicatie met studenten en onderhoudspartijen.

Onderhoudsmedewerker

Onderhoudsmedewerkers, zowel intern als extern, voeren reparaties en onderhoudstaken uit binnen de studentenhuizen. Zij hebben behoefte aan een duidelijke en actuele takenlijst, met per taak de locatie, omschrijving, prioriteit, benodigde materialen en eventuele veiligheidsinstructies. Het systeem moet hen helpen om werkzaamheden efficiënt en traceerbaar uit te voeren.

Onderhoudsmanager

De onderhoudsmanager is verantwoordelijk voor de planning, prioritering en toewijzing van onderhoudstaken. Deze rol vereist inzicht in de totale onderhoudsbehoefte, beschikbare capaciteit, kostenramingen en materiaalgebruik. De onderhoudsmanager moet eenvoudig taken kunnen aanmaken, prioriteiten kunnen wijzigen, voortgang kunnen monitoren en rapportages kunnen genereren.

3.3 Beperkingen en voorkeuren

- CSV-data van onbekende kwaliteit
- Team heeft beperkte ervaring
- Hardware (Arduino) alleen optioneel na MVP
- Voorkeur voor veel ICT infrastructuur

3.4 Functieanalyse (MoSCoW)

Must-have

- Webapplicatie (basisplatform)

- Toegang voor studenten, beheerders, onderhoudsmedewerkers en Onderhoudsmanager
- Inloggen met rolgebaseerde rechten
- **Klachtenregistratie**
 - Studenten kunnen een klacht melden
 - Statusoverzicht per klacht (open, in behandeling, afgerond)
 - Beheerders kunnen klachten toewijzen, prioriteren en afhandelen
 - Basishistorie en feedbackmogelijkheid
- **Huishoudrooster**
 - Aanmaken van huishoudtaken
 - Automatische of handmatige taakverdeling
 - Inzicht voor studenten: wie heeft welke taak en wanneer
 - Taken als uitgevoerd markeren + logging
- **Onderhoudsbeheer**
 - Takenlijst voor onderhoudsmedewerkers
 - Toewijzing door beheerder of onderhoudsmanager
 - Basisregistratie: locatie, omschrijving, status, datum
- **Data**
 - Eenmalige import van CSV-data
 - Minimale validatie op verplichte velden
- **Basisdashboard / rapportage**
 - Overzicht aantal klachten
 - Status onderhoud
 - Huishoudtakenoverzicht
- Modelnetwerk voor studenthuis
 - Lorum
 - Ipsum

Should-have

- **Communicatiefunctie**
 - Reacties op klachten en onderhoudstaken
 - Notificaties bij wijzigingen
- **Uitgebreidere onderhoudsflows**
 - Prioriteitsniveaus
 - Meerdere medewerkers per taak
- **Verbeterde Data**
 - Detectie ontbrekende velden
 - Validatierapport
- **Gebruikersprofielen**

- Contactinformatie
- Voorkeuren

Could-have

- **Documentupload bij klachten**
 - Foto's, video's, PDF's
- **Kalenderweergave**
 - Week- en maandplanning
- **Uitgebreide dashboards**
 - Energieverbruik
 - SLA-tijden onderhoud
- **Tags en thema's voor klachten**
 - Geluid, verwarming, veiligheid, etc.

Won't-have

- Smart-home integraties (luchtkwaliteit, sensoren, Arduino)
- Communityfunctionaliteiten (evenementen, hulpmiddelen delen, voorraadbeheer)
- Integraties met externe systemen
- Organisatorische veranderprocessen binnen Student Housing B.V.

welke vragen staan er nog open?%%%

waarom INVEST Principe, leg deze keuze uit%%%

- hiervoor wordt INVEST principe gebruikt.

- <https://www.visual-paradigm.com/scrum/write-user-story-smart-goals/>

- Acceptatiecriteria duidelijk en testbaar zijn.
- De impact op database, rollen of interface besproken is
- Dependencies benoemd zijn.
- Het team inschat dat de story binnen één sprint uitvoerbaar is.

6.3 Tools

De tools die binnen het project gebruikt worden zijn:

- **GitHub** - versiebeheer en centrale plek voor code
- **Microsoft teams** Voor documenten opslaan
- **Microsoft Word** Voor documentformaat
- **Visual Studio Community 2022** - voor ontwikkeling
- **SQL Server** - database
- **Trello** - sprintplanning, taakbeheer en notities tijdens bijeenkomsten

- **Mermaid** - primaire tool voor diagrammen (alternatief: PlantUML) Visual paradigm, DrawIO mag ook
- **GNS-3** - Netwerkarchitectuur

6.4 Samenwerkingsafspraken

6.4.x Scrum rollen

Elke sprint wordt iemand aangewezen die de rol van Scrum Master op zich neemt, de rest is het Scrum Team. Gezamenlijk vertegenwoordigen wij de Product owner

6.4.1 Communicatie

- **WhatsApp** wordt gebruikt voor korte vragen, snelle updates, aanmoediging en afmeldingen.
- **Trello** fungeert als onze gezamenlijke sprintplanning.
- **Microsoft Teams** wordt gebruikt voor de vaste weekly meeting:
 - Donderdag om **17:30 uur**
 - Vastlegging gebeurt via actielijst of iets vergelijkbaars
- **GitHub** wordt gebruikt voor het delen van code, documentatie en versiecontrole.
- **Teams** wordt gebruikt voor het delen van documenten

6.4.2 Documentatie

- Notities worden live toegevoegd in **Trello** tijdens bijeenkomsten.
- Het **voorzitterschap** en **Notulist** tijdens overleg rouleert per sessie.
 - De notulist noteert alleen de actiepunten plus datum, tijd, rollen en wie er aanwezig is/was.
- Om de twee weken wordt gezamenlijke documentatie ingevuld volgens het vaste framework:

Analyse – Adviseer – Design – Realisatie – Manage & Control

Dit framework vormt tevens de basis voor sprintdocumentatie en sprintreflectie.

- Sprint review
 - feedback geven
 - elkaar
 - leraren
 - laten we resultaat zien
- Sprint retoperspectief
 - Wat ging er goed?
 - Wat kan er anders?
 - Wat nemen we mee?

6.4.3 Oplevermomenten

- Elke sprint duurt **twee weken**.
- Aan het einde van elke sprint vindt een **Sprint Review** plaats:
 - Opgeleverde onderdelen worden besproken.
 - Feedback wordt verzameld.
 - De planning voor de volgende sprint wordt vastgesteld.

6.4.4 Besluitvorming

- Besluiten worden bij voorkeur genomen op basis van consensus
- Bij verschil van inzicht wordt gestemd.
- Bij een gelijke stemverdeling wordt het besluit bepaald door een muntje op te gooien.
- De teamleden die aanwezig zijn tijdens een overleg nemen een besluit namens de volledige groep

6.4.5 Sprintdocumentatie

Onderstaande structuur wordt elke twee weken aangehouden voor de sprintdocumentatie:

Analyse

- Korte beschrijving van onderzoek, inzichten, problemen en verbeterpunten.

Adviseer

- Formuleren van aanbevelingen, keuzes en richting voor de sprint.

Design

- Uitwerking van ontwerpen, schema's, prototypes en belangrijkste ontwerpkeuzes.

Realisatie

- Wat is gebouwd, getest en aangepast.
- Eventuele technische uitdagingen of afwijkingen van het plan.

Manage & Control

- Evaluatie van kwaliteit en voortgang van de sprint.
- Documenteren van verbeterpunten, feedback en afspraken voor de volgende sprint.

7. Initiële product backlog (MVP)

De initiële product backlog bevat de eerste set van user stories en taken die nodig zijn om het minimale werkende product (MVP) op te leveren. De nadruk ligt op de basisfunctionaliteiten: website/portaal, klachtenregistratie, huishoudrooster, onderhoudsbeheer en eenvoudige rapportage.

7.1 User Stories (INVEST)

US-01a – Toegang tot portaal

Als gebruiker

wil ik kunnen inloggen op een online portaal

zodat ik toegang heb tot alle functionaliteiten van het studentenhuus.

Acceptatiecriteria:

- Inloggen met rol (Student, Beheerder, Onderhoudsmedewerker, Onderhoudsmanager)
- Onjuist wachtwoord geeft foutmelding
- Rolgebaseerde toegang: gebruiker ziet alleen wat relevant is

MoSCoW: Must-have

US-01b – Navigatie en dashboard

Als ingelogde gebruiker

wil ik een overzichtelijk dashboard

zodat ik snel kan zien welke taken, klachten of meldingen relevant zijn voor mij.

Acceptatiecriteria:

- Dashboard toont relevante modules per rol
- Klikken op een module opent de juiste pagina
- Overzichtelijk en eenvoudig te begrijpen

MoSCoW: Must-have

US-02a – Klacht indienen

Als student

wil ik een klacht kunnen indienen via een formulier

zodat ik problemen snel kan melden.

Acceptatiecriteria:

- Formulier bevat categorieën van veelvoorkomende problemen
- Mogelijkheid tot extra toelichting
- Bevestiging na indienen

MoSCoW: Must-have

US-02b – Klachtstatus volgen

Als student

wil ik de status van mijn ingediende klacht kunnen volgen

zodat ik transparantie heb over de afhandeling.

Acceptatiecriteria:

- Status zichtbaar: Open, In behandeling, Afgerond
- Toegewezen medewerker zichtbaar
- Historie van acties zichtbaar

MoSCoW: Must-have

US-03 – Klachten beheren

Als beheerder

wil ik klachten kunnen inzien, toewijzen, prioriteren en feedback geven

zodat klachten efficiënt worden opgevolgd.

Acceptatiecriteria:

- Lijst van openstaande klachten beschikbaar
- Toewijzen aan onderhoudsmanager of medewerker mogelijk
- Prioriteit instellen (Laag, Midden, Hoog)
- Reacties toevoegen die student kan zien
- Status aanpassen

MoSCoW: Must-have

US-04 – Huishoudrooster beheren

Als beheerder

wil ik huishoudtaken kunnen aanmaken, verdelen en aanpassen

zodat het schoonmaakrooster eerlijk wordt bijgehouden.

Acceptatiecriteria:

- Taken aanmaken met omschrijving, locatie, frequentie, student toegewezen
- Taken verdelen handmatig of automatisch
- Aanpassen of verwijderen van taken
- Studenten zien hun taken in rooster

MoSCoW: Must-have

US-05 – Huishoudtaken afvinken

Als student

wil ik mijn toegewezen huishoudtaken kunnen afvinken

zodat iedereen kan zien wat is gedaan.

Acceptatiecriteria:

- Taak markeren als uitgevoerd

- Status real-time zichtbaar voor alle gebruikers
- Historie beschikbaar voor beheerder

MoSCoW: Must-have

7.3 Toelichting backlog

- Alle stories zijn **INVEST-compliant**: klein, testbaar, waardevol, inschatbaar en onafhankelijk waar mogelijk.
- De **MVP-focus** ligt op de absolute basisfunctionaliteiten, zonder smart-home integraties, communityfunctionaliteiten of externe koppelingen.
- Stories kunnen later worden uitgebreid naar **Should-have** en **Could-have** functionaliteiten zoals notificaties, documentupload, uitgebreide dashboards en kalenderweergave.

Bronnenlijst

OpenAI. (2025, november 10–16). Prompt gebruikt voor het taalkundig te verbeteren en tekst vloeiend te laten lopen [ChatGPT, GPT-5 mini]. Geraadpleegd van <https://chat.openai.com>

OpenAI. (2025, november 9). Hoe ziet een sprintplanning eruit? Geraadpleegd van <https://chat.openai.com/>

Visual Paradigm. (z.d.). Write user story SMART goals. Geraadpleegd op 16 november 2025, van <https://www.visual-paradigm.com/scrum/write-user-story-smart-goals/>

<https://scrumguides.org/scrum-guide.html#scrum-definition>

<https://www.visual-paradigm.com/scrum/what-is-definition-of-ready-in-scrum/>

<https://www.visual-paradigm.com/scrum/definition-of-done-vs-acceptance-criteria/>

Moeten wij nog rollen verdelen volgens de scrum indeling?

<https://www.visual-paradigm.com/scrum/>