

Technisch ontwerp

November 2019

# 

# 

# Voorwoord

Dit technisch ontwerp bevat informatie over de technische aspecten van ons project: de Updatetracker.

Huidige situatie: er is bijvoorbeeld software wat een bedrijf niet vaak gebruikt maar wel cruciaal is voor de veiligheid is van de systemen, doordat de software niet vaak wordt gebruikt wordt de software niet vaak geupdate en ontstaat er een veiligheidsrisico.

Gewenste situatie: gebruiker (bijvoorbeeld bedrijf) geeft aan welke software hij gebruikt en welke versie hij er momenteel van heeft. Als er een nieuwere softwareversie is krijgt de gebruiker een mailtje met daarin de info (welke software moet worden geupdate en naar welke versie) de gebruiker kan vervolgens aangeven dat hij hem heeft geupdate naar de nieuwste versie. Op die manier weet het systeem altijd de juiste versie die de gebruiker heeft.

Als de klant niet betaald voor het product kan hij maar twee software systemen toevoegen aan de update tracker en krijgt hij maar één mailtje per maand. Als de gebruiker wel betaald kan hij unlimited software systemen toevoegen en krijgt hij elke dag een mailtje als dat nodig is.

# 

# Inhoudsopgave

[**Voorwoord**](#_t4ebjjtroxpo) **1**

[**Inhoudsopgave**](#_kcgtui27zbj2) **2**

[**Samenvatting**](#_e0u8nlwvtsto) **4**

[**Plan van aanpak**](#_dhtpx9b1jqxz) **4**

[**Ontwikkelomgeving**](#_pvb3z420aagr) **5**

[Technische infrastructuur](#_wv2x5ub2fk6t) 5

[Normalisatie](#_p2u8pr5h1i5h) 7

[Sequentiediagrammen](#_t8zdzt31d0v7) 8

[Login](#_xrtzg8dnsli5) 8

[Logout](#_2uysgju0vrvz) 9

[Register user](#_ohny48dwmway) 9

[Add software](#_7nqzulsrtyln) 10

[Remove software/ user](#_m5pul2ta5f2c) 10

[Edit software/ user](#_v8k4vwpa81l8) 11

[Show data](#_ihdccbq5u0di) 11

[Get software update](#_909c5iscx8db) 12

[Paying customer](#_luhh7m3n0kqd) 13

[Activiteitendiagrammen](#_e1f54i4ns0ii) 14

[Inloggen](#_6qlizw6zzdkx) 14

[Uitloggen](#_3996upmwad25) 14

[Register user](#_d0dlqlou3h3o) 15

[Add software](#_2n0znwmg9xsh) 15

[Remove software/ user](#_j9hj4jx4bt93) 16

[Edit software/ user](#_b6nsvxetafw0) 16

[Show data](#_prcvx2o37djk) 17

[Get software update](#_mbn35y5geuq7) 18

[Paying customer](#_rsmlmoabx9kl) 19

[Klassendiagramm Entity-Relationship Diagram (ERD)](#_4dt5uwooo0v2) 20

[Ontwikkeltools](#_ceb3609x8dxa) 20

[**Beveiliging**](#_jowl09jbbem) **21**

[Autorisatie](#_vf5zohc3fb7r) 21

[Ongewenst gebruik van de applicatie](#_fnedvyzdn4qi) 21

[**Beheer**](#_3hj9ju3zwsy4) **21**

[Back-up](#_b4ddhensczfo) 21

[Content](#_kzcl21a8thk7) 21

# 

# Samenvatting

Ik ga een systeem maken waar mensen hun software kunnen toevoegen aan een lijst. Het systeem haalt nieuwe software versies op uit een API van derden. Vervolgens vergelijkt het systeem de gebruiker zijn software met de nieuwe software versies. Als er een nieuwe softwareversie is voor een gebruiker dan krijgt de gebruiker een mailtje met het bericht dat de gebruiker moet updaten wegens veiligheidsredenen. In het mailtje zal een knop zitten waarmee de gebruiker kan aangeven dat hij/zij zijn software heeft geupdate.

# 

# Plan van aanpak

Dit is een opsomming van alle mijlpaalproducten. In de planning staat wanneer de documenten worden opgeleverd

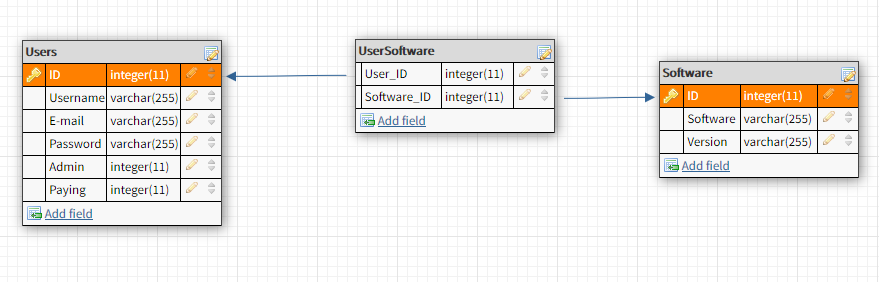
1. Projectplan
2. Functioneel ontwerp
3. Technisch ontwerp
4. Sprint 1
5. Sprint 2
   1. Test Plan
   2. Testrapport
6. Sprint 3
7. Sprint 4
   1. Oplever documentatie
8. Database
9. Admin pagina
10. Gebruiker pagina
11. Update check systeem

# 

# Ontwikkelomgeving

## Technische infrastructuur

Hieronder is een afbeelding die het databasemodel beschrijft. Het laat de logica zien van de database. Ook staat er bij wat voor data type het is. Er is te zien dat User\_ID gekoppeld is met ID van de tabel Users. Dit schema ga ik gebruiken als ik de database ga maken. Onder de afbeelding staat een tabel wat meer uitleg geeft over de tabel.



Verder leg ik nog uit welke data er in de tabellen komt te staan.

Users:

|  |  |  |
| --- | --- | --- |
| Name | Type | Beschrijving |
| ID | Int(11) | ID van Users |
| Username | Varchar(255) | De naam van de user |
| E-mail | varchar(255) | Het e-mail address van de user |
| Password | varchar(255) | Het wachtwoord van de user |
| Admin | integer(11) | Hier komt te staan of een user een admin is ja of nee |
| Software\_ID | integer(11) | De ID van de software |

Software:

|  |  |  |
| --- | --- | --- |
| Name | Type | Beschrijving |
| ID | Int(11) | ID van Software |
| Software | Varchar(255) | De naam van de software |
| Version | varchar(255) | De versie van de software |

UserSoftware:

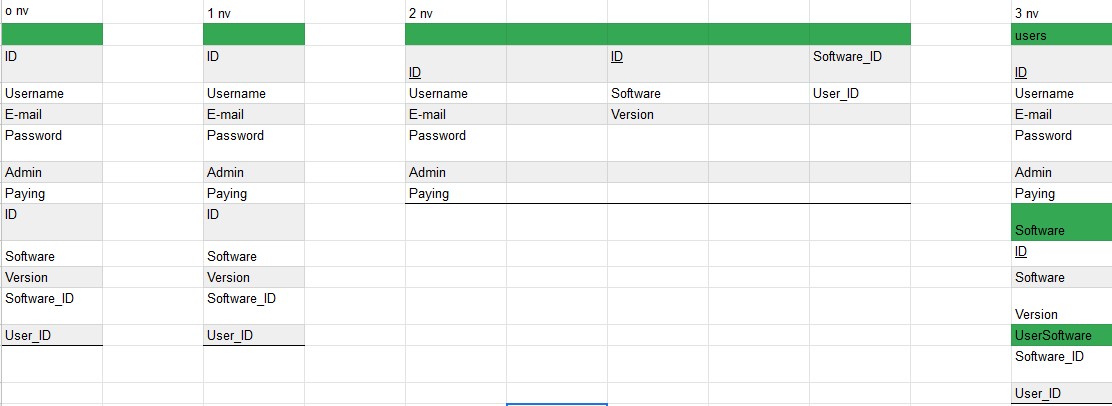
|  |  |  |
| --- | --- | --- |
| Name | Type | Beschrijving |
| User\_ID | Int(11) | ID van user |
| Software\_ID | Int(11) | ID van software |

## 

### 

### Normalisatie

Normalisatie is een techniek bij het ontwerpen van databases. Het heeft twee doelen: opslagruimte besparen en voorkomen dat we meerdere keren dezelfde data vastleggen.



1e nv: Herhalende groepen worden in een aparte tabel gezet (hier niet het geval).

2e nv: Er zijn afzonderlijke tabellen gemaakt voor sets met waarden die op meerdere records van toepassing zijn.

3e nv: Velden die niet belangrijk zijn voor de sleutel zijn geëlimineerd.

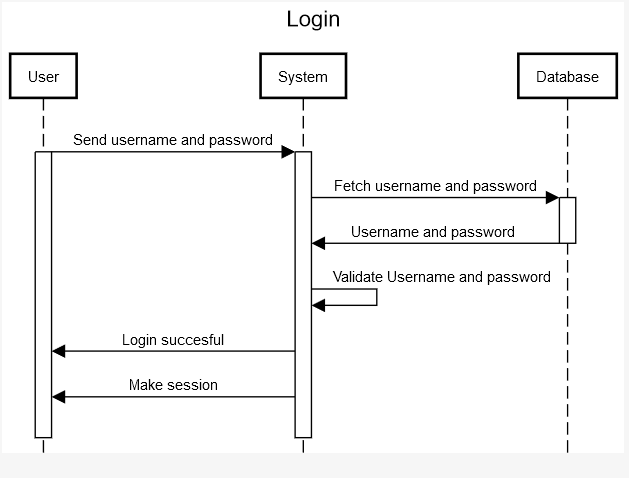
## 

## Sequentiediagrammen

Hieronder zie je de verschillende diagrammen die de interactie tussen objecten die op een opeenvolgende wijze wordt getoond.

### Login

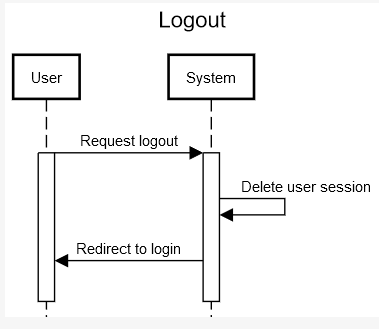
De user verstuurd Username en password naar het het systeem. Vervolgens gaan de gegevens naar de database en die stuurt ze weer terug. Als de inlog gegevens valide zijn wordt de user ingelogd en doorgestuurd.



### 

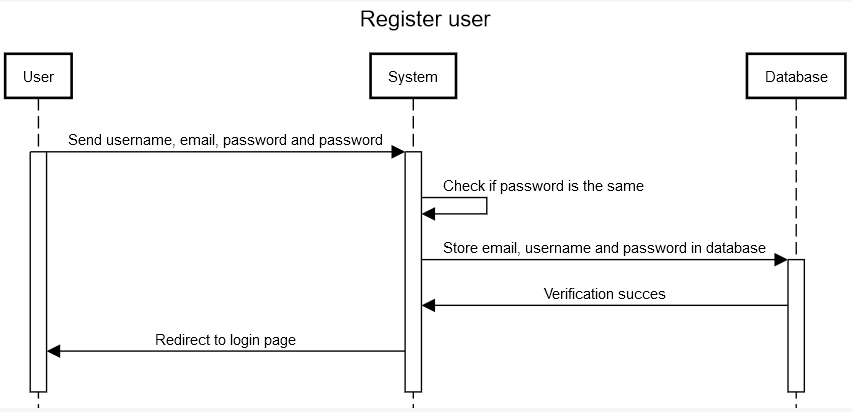
### Logout

De user verstuurd een logoutrequest. Het systeem verwerkt de request en beëindigt de sessie. User word doorgestuurd naar login pagina



### Register user

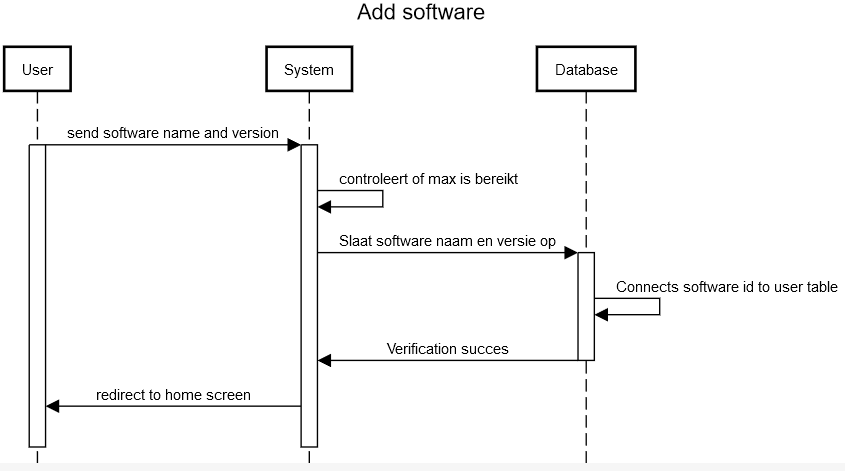
User verstuurt zijn gegevens naar het systeem. Systeem controleert of de wachtwoorden hetzelfde zijn. Het zet de user in de database. De database confirmed dat het gelukt is. User word doorgestuurd naar inlog pagina.



### 

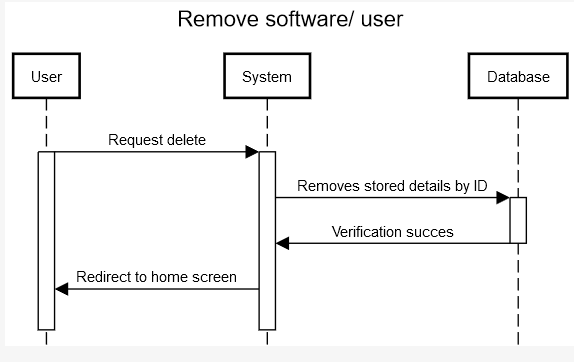
### Add software

User verstuurt software naam en versie het systeem controleert of max aantal software is bereikt. Zo niet slaat het systeem de gegevens op in de database. De database connect software id met user table. De database verstuurt een succes. User wordt doorgestuurd naar home scherm.



### Remove software/ user

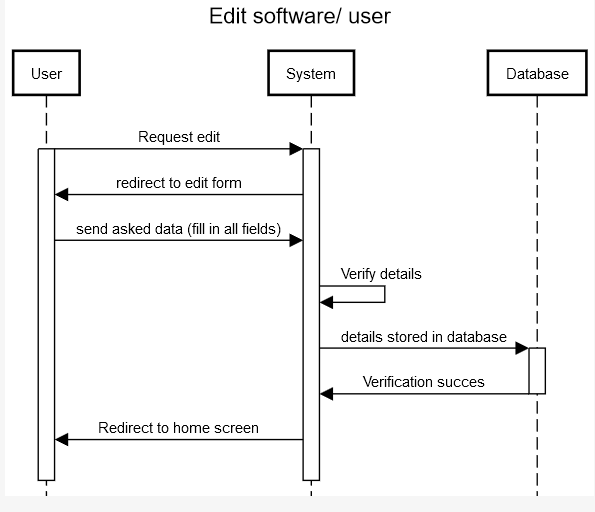
User verstuurd request delete naar systeem. Systeem verwijderd de gegevens uit de database. De database verstuurd succes bericht. User wordt doorgestuurd naar home scherm.



### 

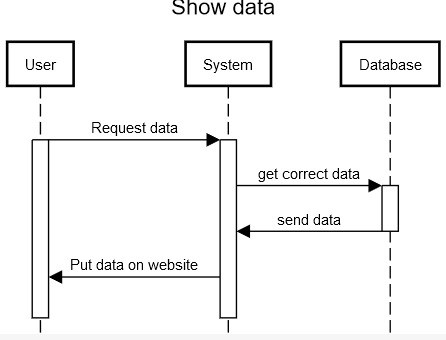
### Edit software/ user

User vraagt edit aan. Systeem stuurt user door naar edit form. User vult alle velden in. Systeem controleert de velden. Update de details in de database. Database verstuurd success message. User wordt doorgestuurd naar home scherm.



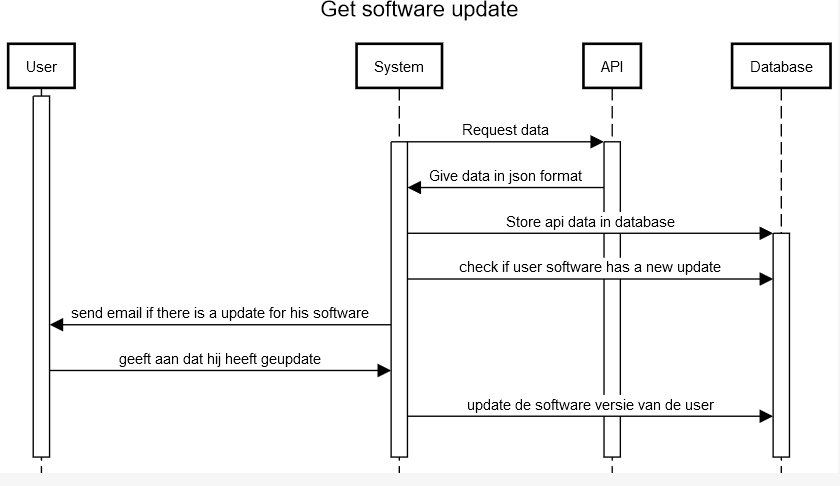
### Show data

User vraagt data aan. Vervolgens haalt het systeem de data uit de database. En toont het op het scherm.



### Get software update

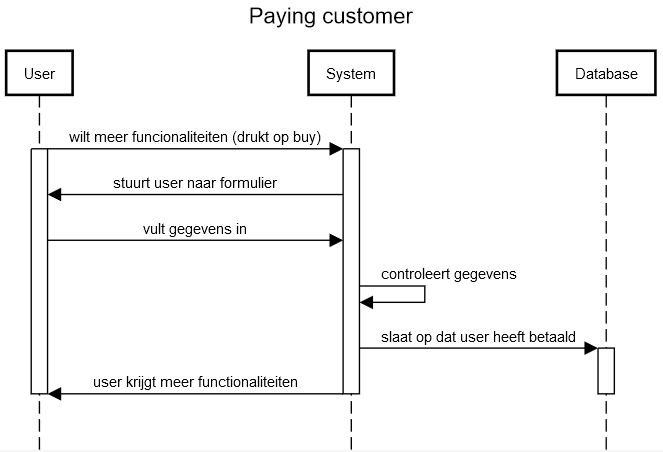
Systeem vraagt data aan bij api van derden. Die api levert data die vervolgens in de database wordt gezet. Daarna controleert het systeem of er users zijn die een software update moeten doen. Zo ja dan stuurt het systeem ze een email. User kan vervolgens aangeven dat hij/ zij heeft geupdate. Dan update het systeem de huidige versie van de software van de user.



### 

### Paying customer

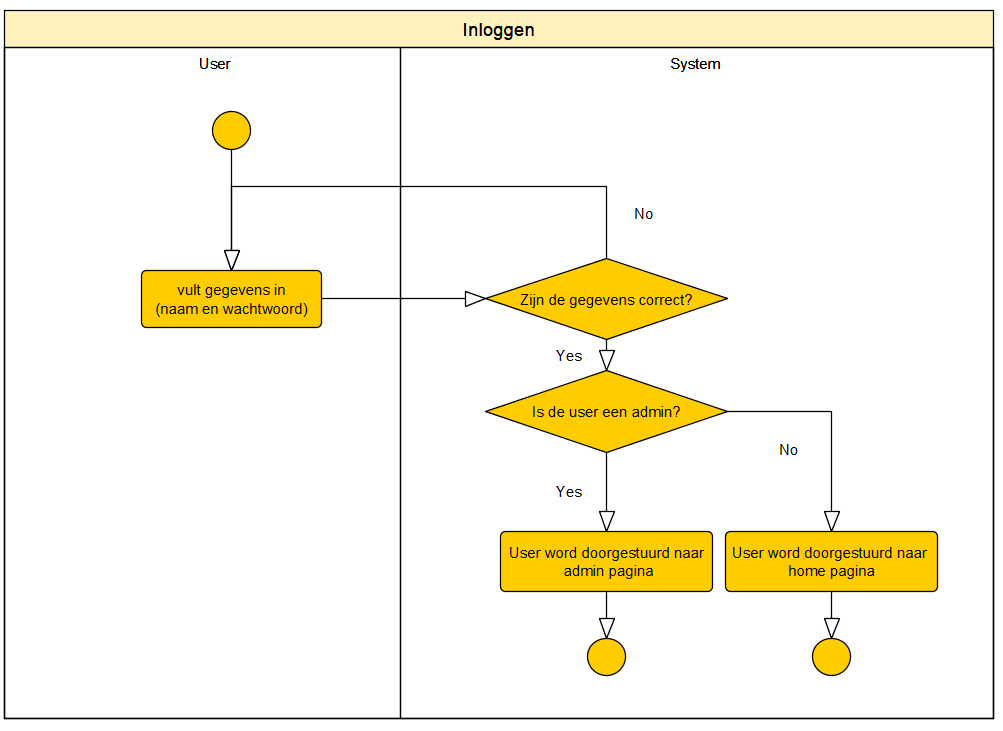
User wilt graag meer functionaliteiten en drukt op “buy”. De user word doorgestuurd naar een invulformulier voor de bank gegevens. User vult gegevens in en het systeem controleert ze. Vervolgens slaat het systeem op dat de user betaald en geeft de user meer functionaliteiten.



## Activiteitendiagrammen

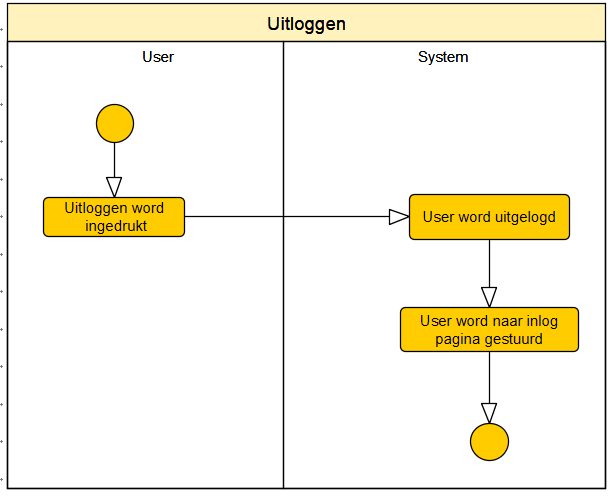
### Inloggen

Bij het inloggen vult de user zijn gegevens in (naam en wachtwoord). Het systeem controleert de gegevens. En controleert of de user een admin is. Als de user een admin is wordt die doorgestuurd naar de admin pagina. Anders naar de home pagina.



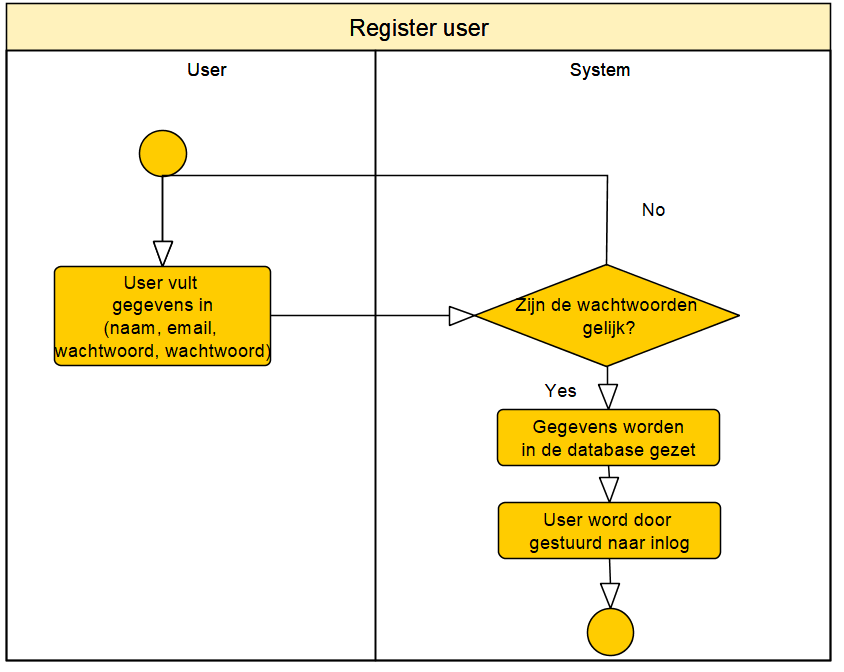
### Uitloggen

Bij het uitloggen klikt op uitloggen. Het systeem stuurt de user door naar de inlogpagina en beëindigd de sessie.



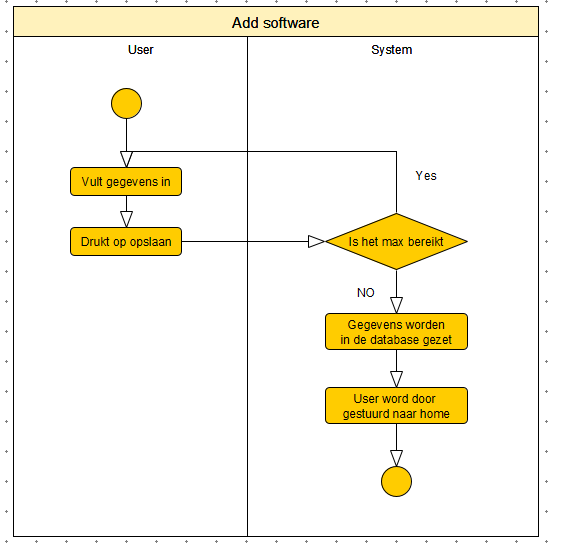
### Register user

User vult alle gevraagde gegevens in het formulier (naam, email wachtwoord, wachtwoord). Het systeem controleert of de wachtwoorden gelijk zijn. Zo ja dan worden de gegevens in de database gezet en wordt de user doorgestuurd naar de inlogpagina.



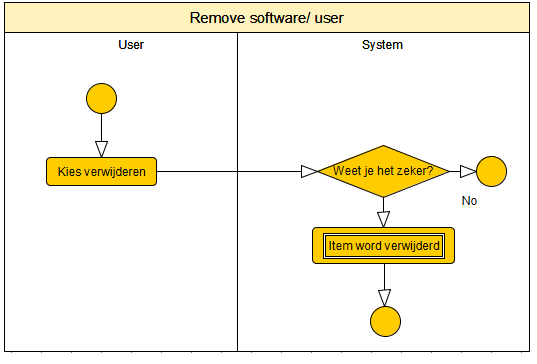
### Add software

User vult gevraagde gegevens in op formulier en drukt op opslaan. Als het user niet zijn max aantal software heeft bereikt zet het systeem de data in de database en stuurt de user door naar de homepagina.



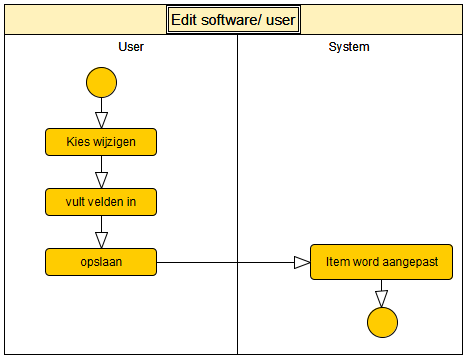
### Remove software/ user

User drukt op verwijderen. Het systeem vraagt om bevestiging. Als de user op ja drukt dan wordt het item verwijderd.



### Edit software/ user

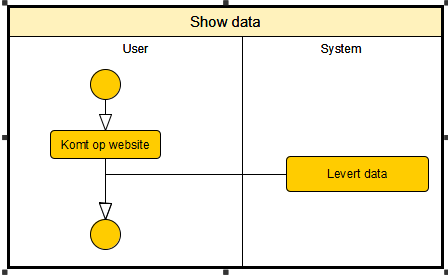
User drukt op wijzigen van item. Vervolgens past de user de gegevens aan en drukt op opslaan. Het systeem past vervolgens het item aan.



### 

### Show data

De user komt op website. Het systeem levert data.

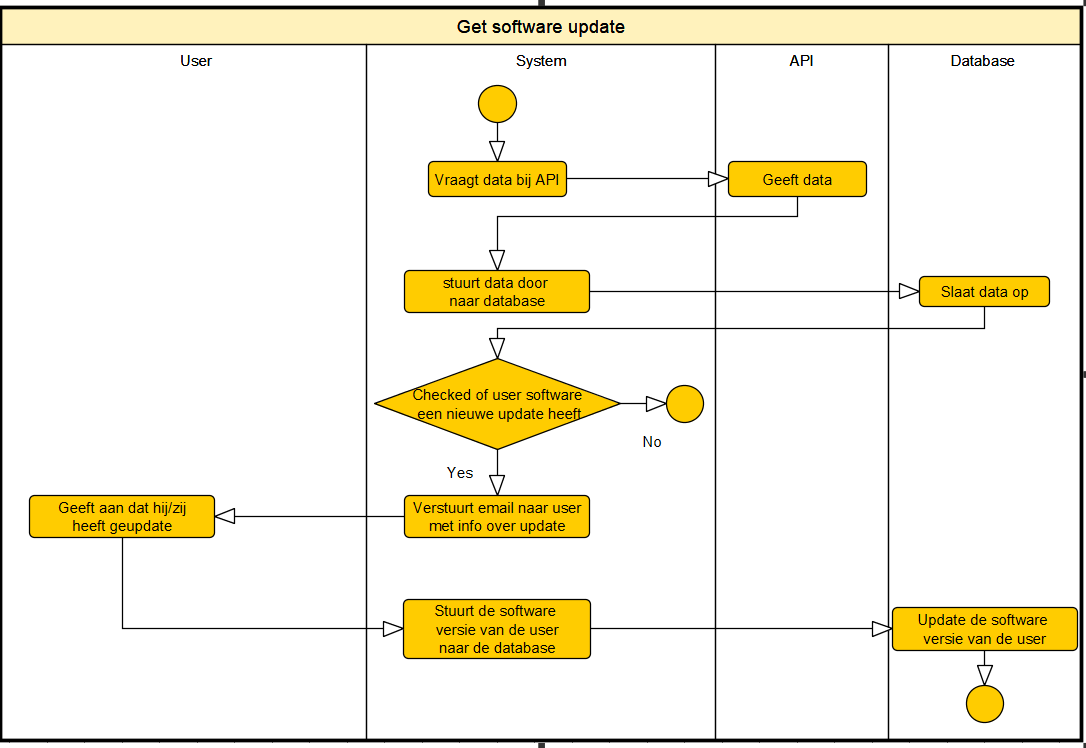


### 

### 

### Get software update

Het systeem vraagt data aan bij de API. De API levert data. Vervolgens stuurt het systeem data door naar de database. De database slaat de data op.Het systeem checked of een user zijn software een nieuwe update heeft. Zo ja verstuurt het systeem een email naar de user toe met info over update. De user geeft vervolgens aan dat hij/zij heeft geupdate. Het systeem verwerkt dat en update de software versie in de database.



### 

### Paying customer

User wilt meer functionaliteiten en drukt op “buy”. Het systeem stuurt user door naar invulformulier. User vult gegevens in en het systeem controleer ze. Als de gegevens correct zijn slaat het systeem op dat de user betaald en krijgt de user meer functionaliteiten. Anders moet de user het opnieuw proberen

## 

## Klassendiagramm Entity-Relationship Diagram (ERD)

Ik combineer het klassendiagram en de ERD. De user heeft meerdere softwares maar een software heeft geen users. De api heeft software en software heeft exact dezelfde data.

## 

## Ontwikkeltools

Voor het programmeren gebruik ik de code editor Sublime text 3. Ik heb deze code editor gekozen omdat ik er goede ervaring mee heb.

Verder gebruik ik xampp om het project lokaal te testen.

PHPMyAdmin gebruik ik om een database te maken. Die heb ik nodig om data op te slaan.

Om verder het systeem te testen worden zoveel mogelijk web browsers gebruikt. Zoals Chrome, firefox, edge en opera.

# 

# Beveiliging

## Autorisatie

Bij het inloggen wordt gecontroleerd of de persoon een admin is. Als dat zo is dan gaat diegene naar de admin pagina.

## Ongewenst gebruik van de applicatie

Alles wat in de database wordt opgeslagen is van te voren gecontroleerd door het systeem. Op die manier komt er geen data in de database die er niet thuis hoort.

# 

# Beheer

## Back-up

Back-ups van code worden opgeslagen op github. Verder is het duidelijk gemaakt in de planning wanneer ik wat ga doen. Het risico dat ik niet door kan als er iemand niet is vervalt omdat ik als enige aan dit project werk.

## Content

Content op de website wordt beheerd door elke user apart. Verder wordt er content geleverd door een API van derden. Dat gebeurt door een keer op een dag een API call te maken en de data op te slaan in de database