

Arte-Tech

CMS Development

Wouter Vlaeyen

Academiejaar 2019-2020

New Media Development

Bachelor in de grafische en digitale media

Arteveldehogeschool

Inhoudstafel

Discover	5
Briefing	5
Define	7
Analyse	7
Planning	7
Design	8
Sitemap	8
Wireframes	9
Wireflow	12
Style Guide	13
Visual Design	20
Develop	26
Code Snippets	26
Schermafbeeldingen	32
Deliver	40
Handleiding	40
Deploy	44
Deployment Guide	44

Briefing

De klant heeft weinig kaas gegeten van computers, maar beseft wel dat zijn manier van werken een stuk gemakkelijker zou kunnen gemaakt worden door het maken van een webapplicatie. Hij had zelf het volgende idee:

Een all-in-one webapplicatie waarbij hij de uren niet meer zelf hoeft in te vullen, maar reeds gebeurd is door het personeel / ondераannemers die hij uitstuurt.

Mobiele WebApp

Het personeel/ondераannemers zullen via hun smartphone via een progressive web app - na inloggen - dagelijks het volgende kunnen invullen:

- klant
- startuur
- einduur
- pauze
- uitgevoerde activiteiten
- gebruikte materialen (vb schroeven, kabels, sensoren, ...)
- transport (aantal kilometers)

Onderaannemers hebben ook een extra optie waarbij ze hun tarief kunnen wijzigen (uurtarief, transportkost). Het zou handig zijn als de gebruikers van die app ook kunnen bekijken hoeveel uren ze in totaal al gepresteerd hebben (per week / per maand / per klant). Bedenk zelf een aantal praktische toevoegingen.

Backend

In de backend worden alle medewerkers van ARTE-TECH toegevoegd. In de backend worden alle klanten van ARTE-TECH toegevoegd. Het uurtarief van de techniekers en transportkost wordt per bedrijf op voorhand afgesproken en ingevoerd in de backend.

Volgende regel geldt altijd voor uurtarief bij uitzonderingen:

- overuren tijdens de week (> 8 uur): 20% meer
- zaterdag: 50% meerkost
- zondag: dubbel tarief

De manager kan op een gebruiksvriendelijke manier de gepresteerde uren opvolgen, die doorgegeven zijn door zijn personeel/ onderaannemers (en zo nodig ook corrigeren of handmatig invullen). De backend zou een functie moeten hebben dat een periode kan opgesteld worden en kan gemeld worden naar de klant (het bedrijf) per mail. Deze klant moet de periode nakijken via de website (zie hieronder) en bevestigen. (tot nog toe gebeurde dat immers op papier)

Via de backend moet het mogelijk zijn om totaalprijs per periode te berekenen en dit te exporteren naar een overzichtelijke xls / pdf. Je hoeft geen rekening te houden met BTW (alles excl. btw) Facturatie is niet nodig (maar mag uiteraard)

Website

Elke klant (de bedrijven dus) kan de gepresteerde uren nakijken via hun computer. Zij volgen dit op per periode (na notificatie per mail), waarbij ze in detail alle prestaties kunnen nakijken en kunnen bevestigen.

Indien er iets zou mis zijn, hebben ze de mogelijkheid om een opmerking te versturen. Er is ook een mogelijkheid om een periode als overzichtelijke pdf te printen voor eigen administratie.

Analyse

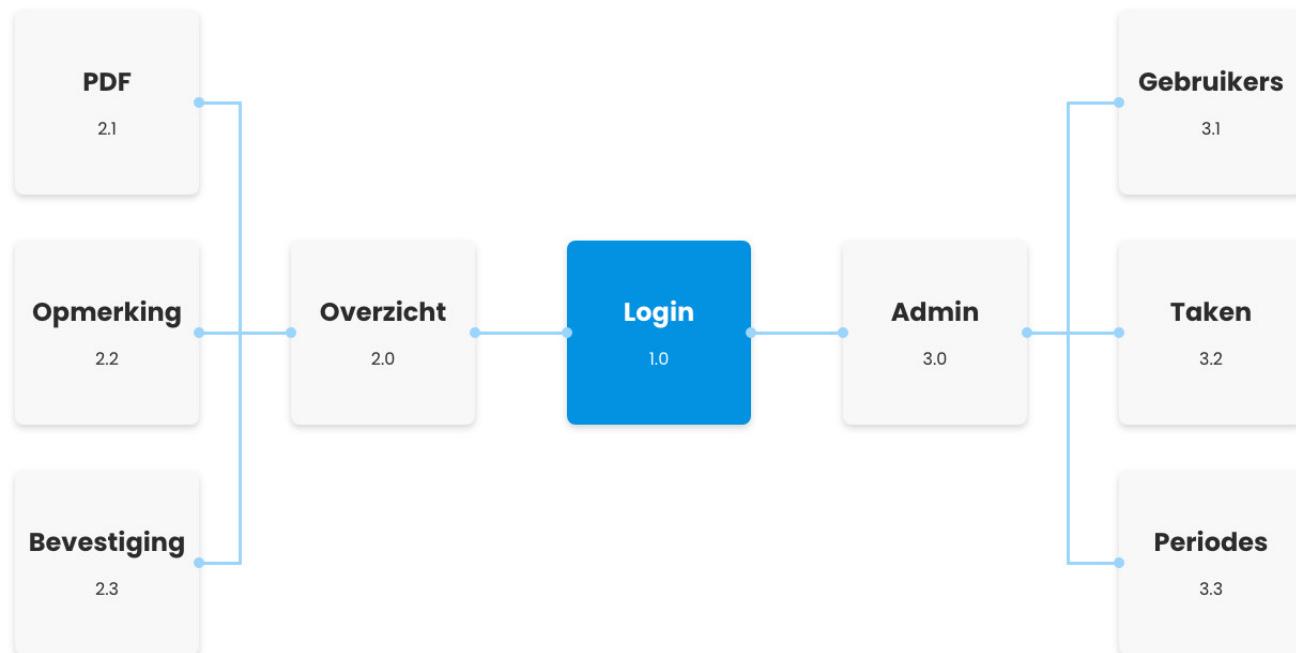
Het opgeleverde eindproduct bestaat uit twee delen. Enerzijds een Content Management Systeem gebouwd met Symfony, inclusief bijbehorende website en anderzijds een een web applicatie gebouwd met Vue.js.

Dit product wordt ten laatste opgelevereerd op 5 januari 2020.

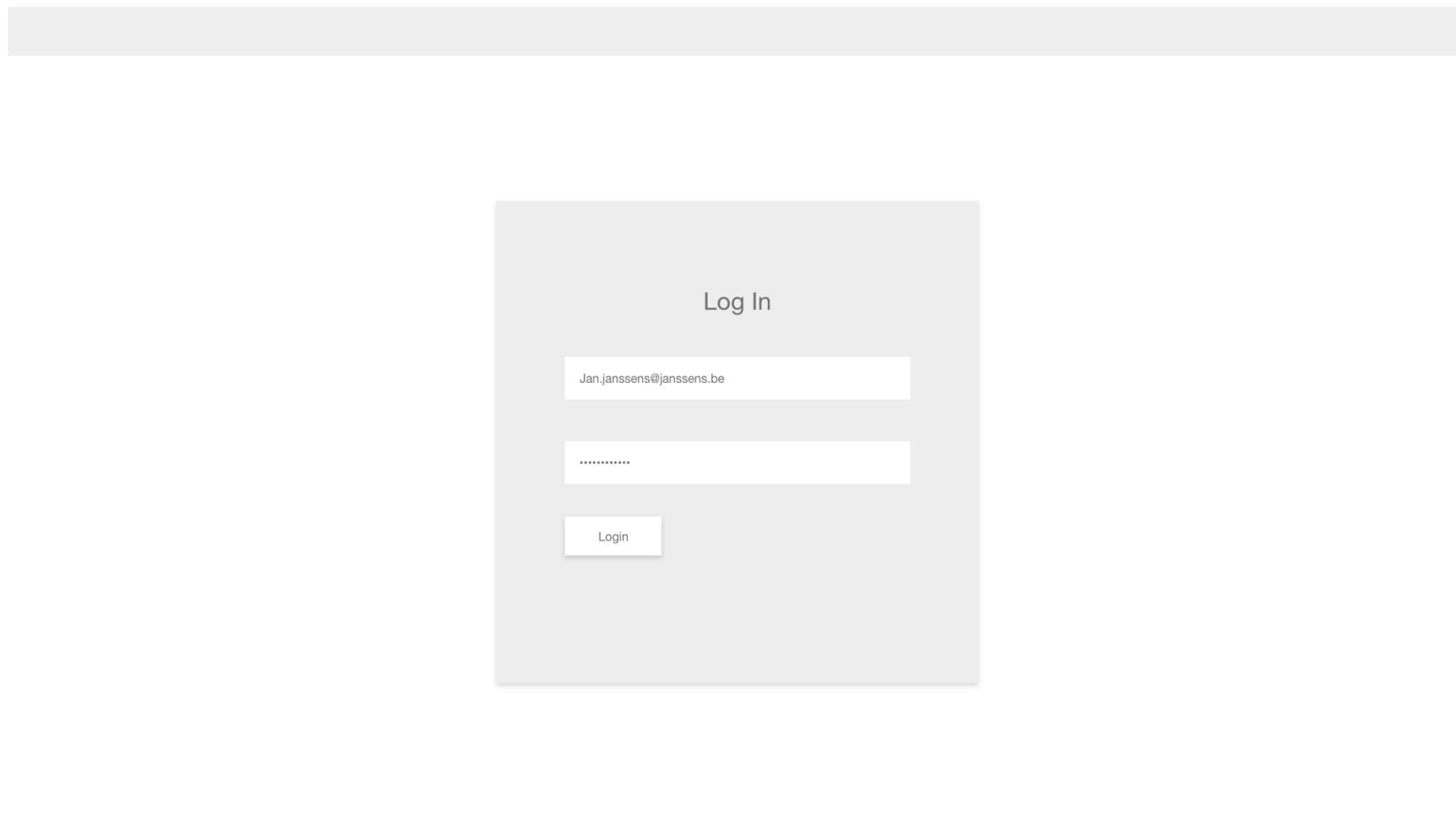
Planning

Taak	Planning											
	NOV				DEC				JAN			
	4	11	18	25	2	9	16	23	30		6	
Planning				■								
Wireframes			■	■								
Wireflow				■								
Visual Design				■	■							
Development CMS					■	■						
Development Web					■	■						
Development App						■	■					
Testing							■	■	■			
Schrijven dossier	■											

Sitemap



Wireframes



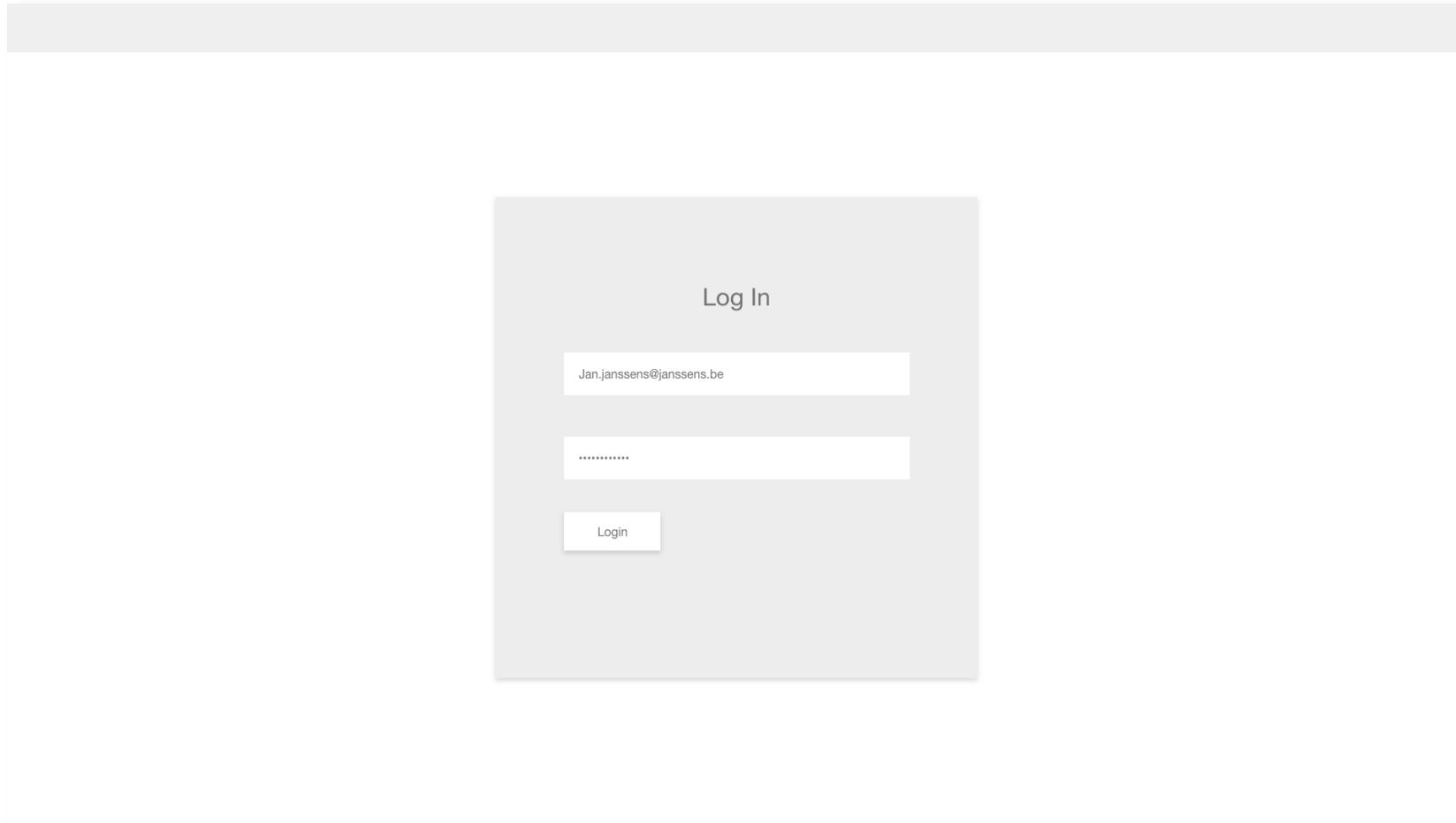
DESIGN – WIREFRAMES

The wireframe shows a service booking confirmation page. At the top right, there are links for 'Bedrijfsnaam Klant' and 'Log Out'. Below this, the title '18 November 2019 – 24 November 2019' is centered. On the left is a 'Vorige' button, and on the right is a 'Volgende' button. A table lists eight service entries, each with a date, technician name, hours, rate, and price. The total price at the bottom is € 2275.84. At the bottom left are buttons for 'Bevestig' and 'Opmerking'.

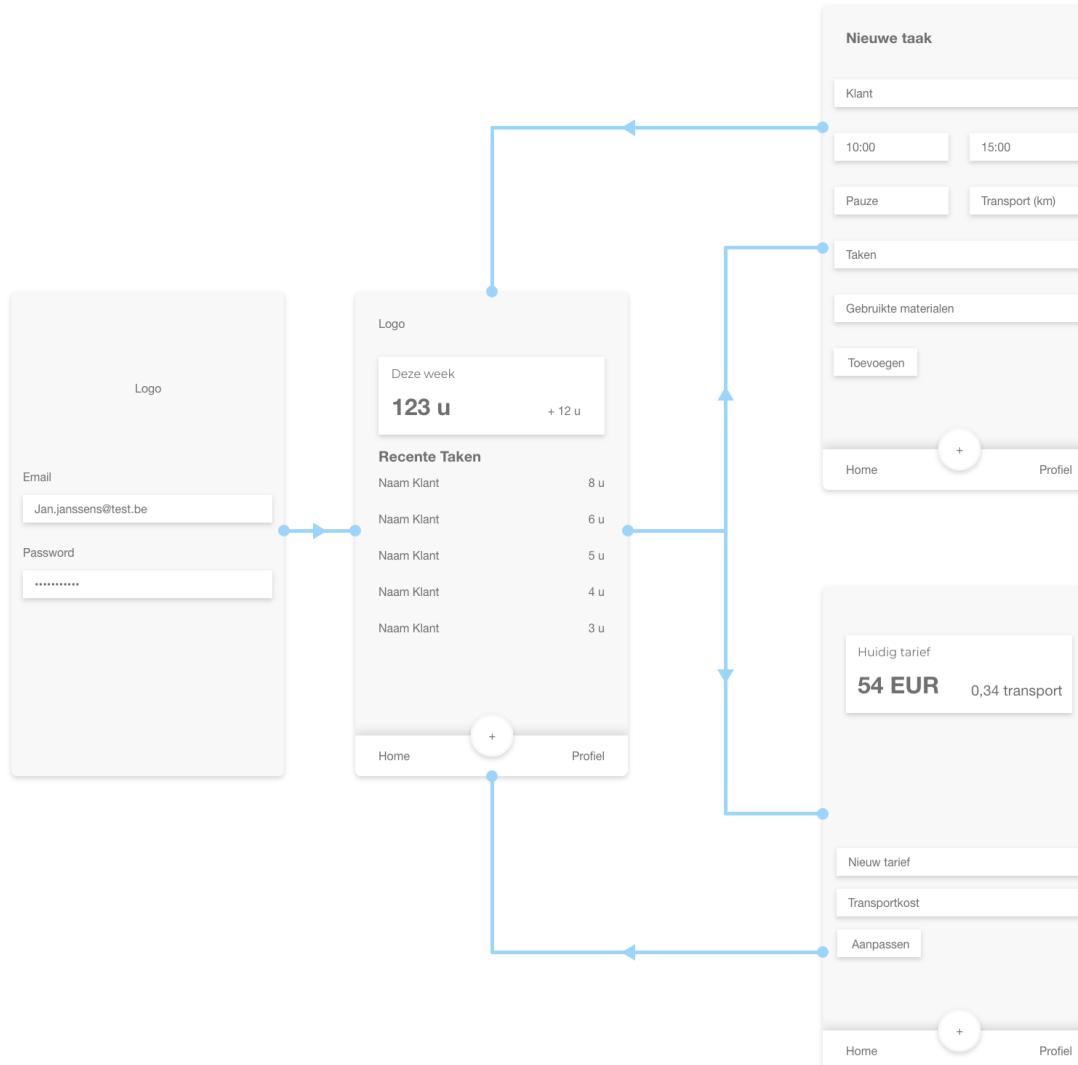
Dag	Technieker	Uren	Uurtarief	Prijs
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48
18 - 11 - 2019	Jan Janssens	8	€ 35.56	€ 284.48

Totaalprijs: **€ 2275.84**

Bevestig Opmerking



Wireflow



Style Guide

Grid

Het gebruikte gridsysteem is gebaseerd op de standaard 12-kolom layout gebruikt in Bootstrap. Met de verschillende beschikbare CSS-classes kan de layout makkelijk opgebouwd worden.

- container (& -fluid)
- row
- col-{1-12} (CMS)
- offset-{1-12} (CMS)
- col-{breakpoint}-{1-12} (Web App)
- offset-{breakpoint}-{1-12}

In het voorbeeld hiernaast kan je zien hoe een typische pagina in de CMS is opgebouwd.

The screenshot shows a dark blue sidebar on the left with a white header containing a logo and three navigation items: 'Overzicht', 'Gebruikers', and 'Taken'. Below these is a blue-highlighted item 'Periodes'. At the bottom of the sidebar is a 'Log Out' button. To the right of the sidebar is a white content area with a red border. The content area has a header 'Periodes' with a 'Start periode' button. Below this is a table with four columns: 'ID', 'Klant', 'Duur', 'Status', and 'Acties'. Three rows of data are shown:

ID	Klant	Duur	Status	Acties
1	Lang Bedrijf	09/12/2019 - 16/12/2019	Bevestigt	Bekijken Stuur e-mail
2	Lang Bedrijf	23/12/2019 - 31/12/2019	Wachten op bevestiging	Bekijken Stuur e-mail
3	Nieuw Bedrijf	30/12/2019 - 31/12/2019	Wachten op bevestiging	Bekijken Stuur e-mail

In de screenshot is te zien hoe de layout gebouwd is. De sidebar neemt de ruimte van een col-2 element in, terwijl een container element in de col-10 ruimte de rest van de UI bevat.

Typografie

Typografie is zeer belangrijk in het design van een applicatie. Bij Arte-Tech wordt er gebruik gemaakt van een simpele aanpak.

De gebruikte fonts voor alle Arte-Tech applicaties zijn Poppins en Muli. Beide zijn te vinden op fonts.google.com.

De type-scale die toegepast wordt is 1.250, zoals hiernaast in het voorbeeld te zien is.

H1 Header (Poppins 48px)

H2 Header (Poppins 39px)

H3 Header (Poppins 31px)

H4 Header (Poppins 28px)

H5 Header (Poppins 20px)

H6 Header (Muli 16px)

Base font (Muli 16px)

Kleuren

Arte-Tech heeft een simpel kleurenpalet dat vooral bestaat uit één hoofdkleur, een kleur voor tekst en een default achtergrondkleur. Extra kleuren worden gebruikt voor bepaalde statussen aan te duiden.

Colours

Primary



Danger



Warning



Success



Foreground



Background



Tabellen

Tabellen worden bij Arte-Tech gebruikt in de CMS om collecties van data te tonen. Als CMS-onderdeel zijn ze dus onmisbaar en is een juiste stijl belangrijk.

Standaard zijn de kolomlabels groter dan de andere rijen om een duidelijk verschil te tonen.

Cards

Naast tabellen worden cards ook gebruikt om data te laten zien. Een kaart is vergelijkbaar met een cel uit een tabel.

De card component heeft een label met daaronder een vorm van data. Dit kan een getal, grafiek of tekst zijn.

ID	Klant	Duur	Status	Acties
1	Lang Bedrijf	09/12/2019 - 16/12/2019	Bevestigd	Bekijken Stuur e-mail
2	Lang Bedrijf	23/12/2019 - 31/12/2019	Wachten op bevestiging	Bekijken Stuur e-mail
3	Nieuw Bedrijf	30/12/2019 - 31/12/2019	Wachten op bevestiging	Bekijken Stuur e-mail

Een tabel in de CMS met verschillende vormen van data in de cellen.



Verschillende kaarten op de homepagina van de CMS. De data in deze kaarten kan gaan van getallen tot volledige grafieken.

Buttons & Links

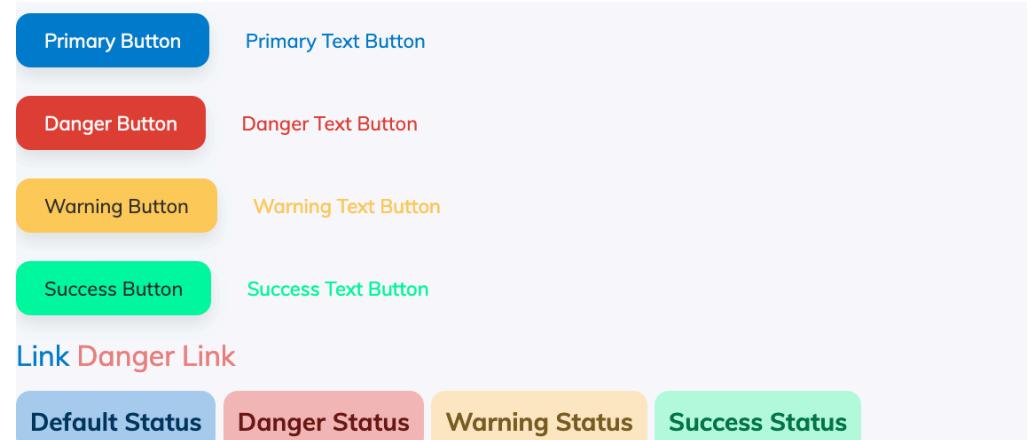
Buttons en links zijn belangrijk voor makkelijke navigatie doorheen de applicatie en de CMS. Om de navigatie makkelijk te houden hebben alle links en buttons een makkelijk herkenbare styling.

Beide buttons en links kunnen in alle kleuren toegevoegd worden om zo extra nadruk te geven aan de functie van deze button of link.

Labels

Labels zijn in de CMS belangrijk om verschillende statussen aan te duiden. Van taken tot periodes, een snelle blik aan het label toont meteen de status van een taak of periode, waarna de gebruiker de juiste actie kan ondernemen.

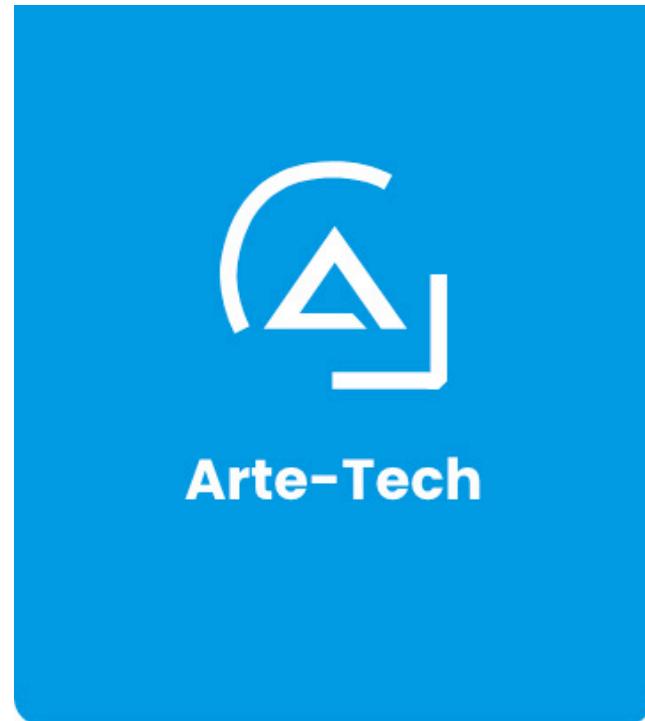
Net als buttons en links zijn labels beschikbaar in alle kleuren om de juiste soort status te tonen.



Responsive Design

In de web application is responsive design absoluut nodig. Aangezien de applicatie op mobile devices en desktops moet kunnen draaien.

In combinatie met het grid systeem is het opzetten van een responsive layout geen enkel probleem. De breakpoints zijn opgezet om verschillende layouts voor verschillende devices te maken: van smartphones tot tablets en desktops.

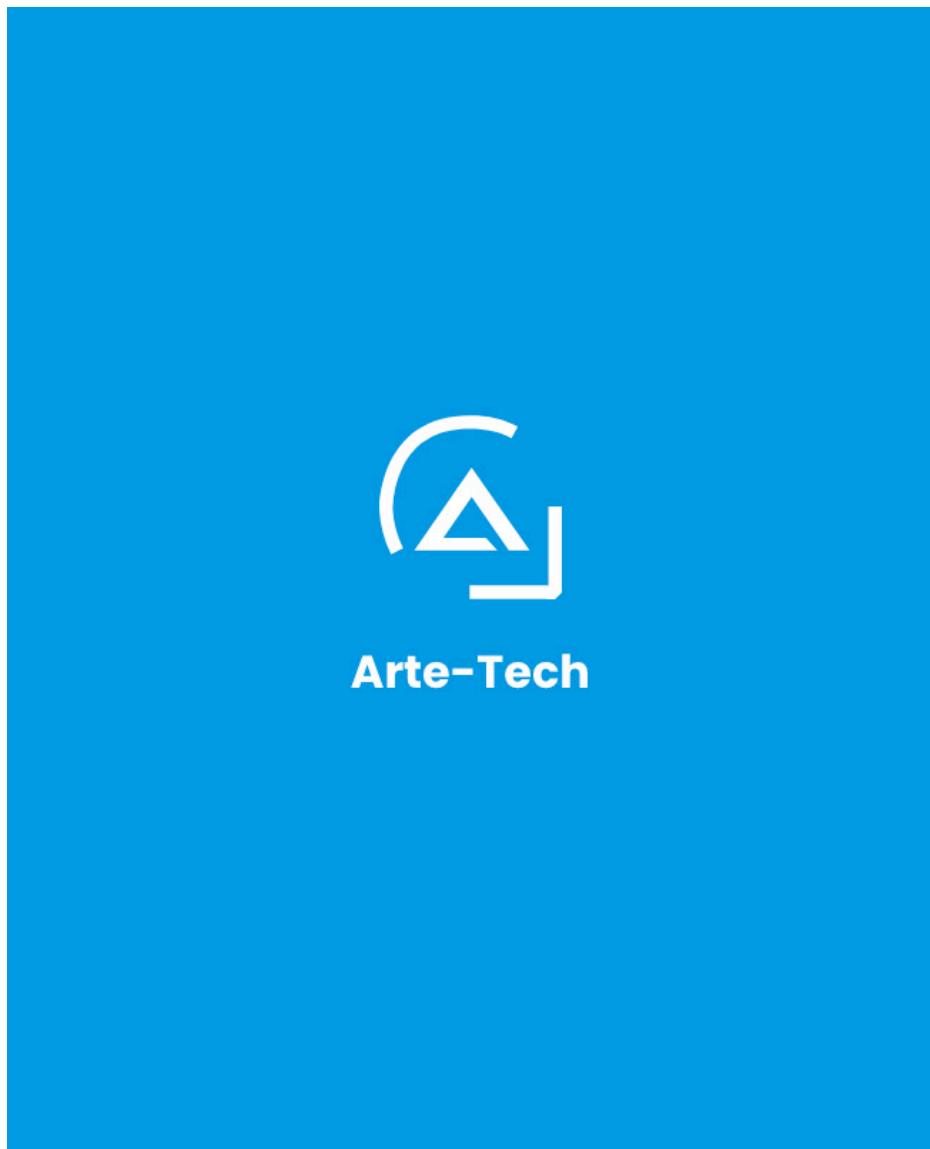


Meld je aan om de applicatie te gebruiken.

Gebruikersnaam

Wachtwoord

Log in



Log in.

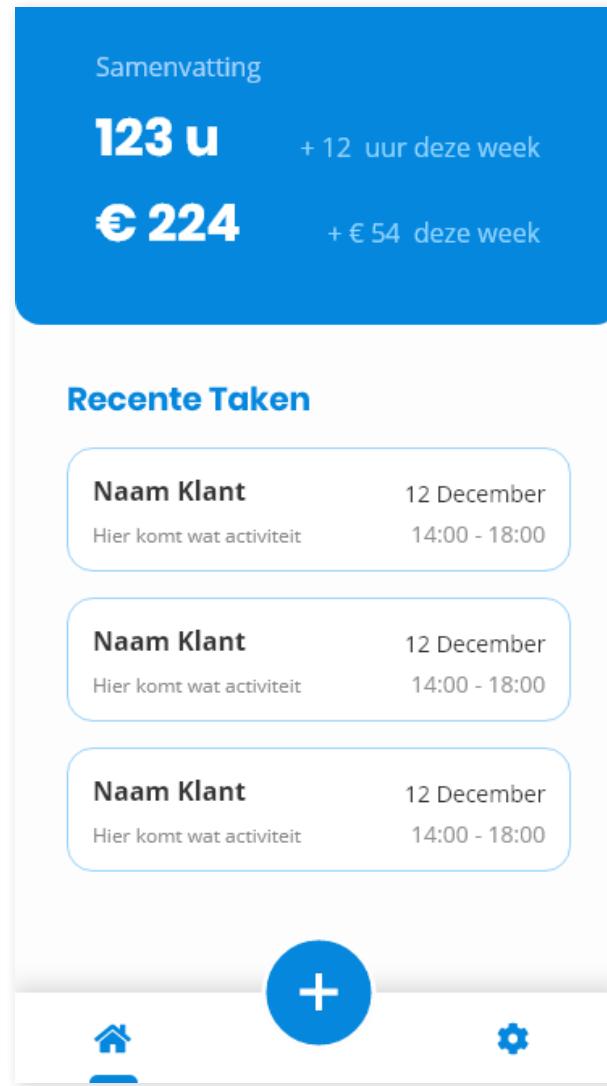
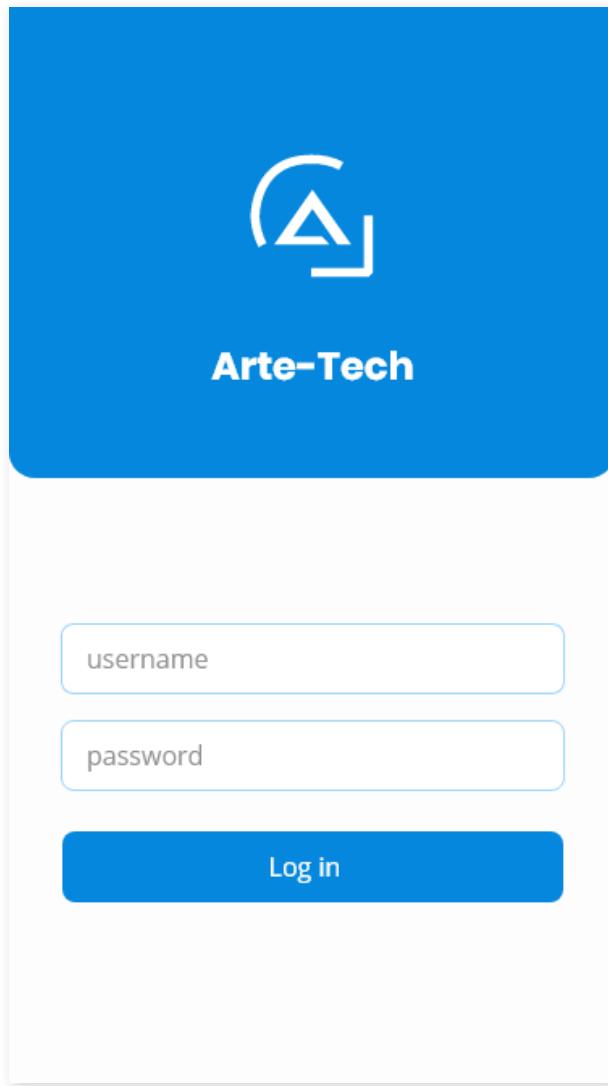
Meld je aan om de applicatie te gebruiken.

Gebruikersnaam

Wachtwoord

Log in

Visual Design



[←](#)

Taak voltooien

Naam Klant	12 December
Naam Klant	12 December
Naam Klant	12 December

[←](#)

Taak voltooien

Naam klant
12 December 2019

Startuur	Einduur
10:00	18:00
Pauze	Transport
15 minuten	25 km

Activiteiten
Heel veel activiteiten

Materialen
Helemaal niets

[Voltooien](#)

Huidige tarieven

€ 45,00

Prijs per uur

€ 0,26

Transportkost

[Tarieven aanpassen](#)

[Prijs per uur](#)

[Transportkost](#)

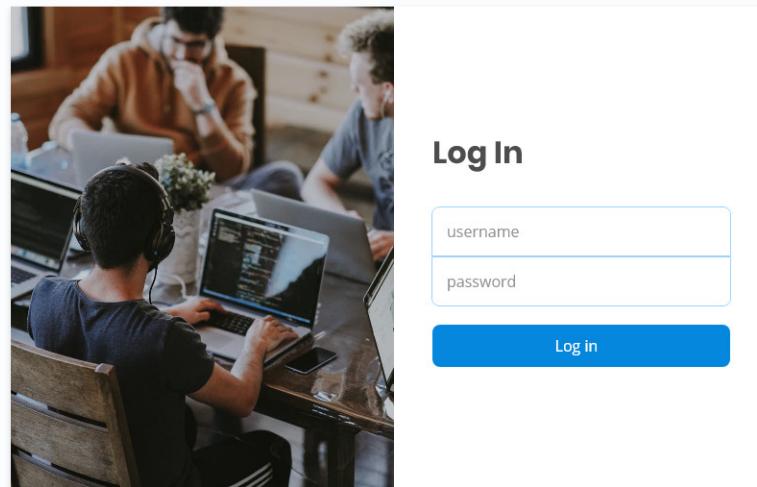
[Aanpassen](#)

[+](#)

[Home](#)

[Gears](#)

DESIGN – VISUAL DESIGN



18 November 2019 – 24 November 2019

Bedrijfsnaam		Goedkeuren
bedrijf@bedrijf.null		PDF Maken

Datum	Techneiker	Tijd	Prijs
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0,00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0,00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0,00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0,00

Opmerkingen

Dit is een mooie opmerking.

03 – 01 – 2020 10:10:59

The screenshot shows a software application interface. On the left is a vertical blue sidebar with white text and icons. At the top of the sidebar is a white logo icon. Below it are four menu items: "Overzicht", "Personen", "Taken", and "Periodes". At the bottom of the sidebar is the text "Uitloggen". The main content area to the right has a white header with the word "Titel" and a blue "Toevoegen" button. Below this is a table with four rows of data. The table has columns for "Datum", "Technieker", "Tijd", and "Prijs". All rows show the same information: "02/01/2020", "Voornaam Naam", "11:00 – 16:00", and "€ 0.00".

Datum	Technieker	Tijd	Prijs
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0.00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0.00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0.00
02/01/2020	Voornaam Naam	11:00 – 16:00	€ 0.00

Code Snippets

CMS



```
// Color functions
@function cms-color($key: 'primary') {
  @return map-get($colors, $key);
}

$color-interval: 10%;

@function cms-color-level($color-name: 'primary', $level: 0) {
  $color: cms-color($color-name);
  $color-base: if($level < 0, black, white);

  @return mix($color-base, $color, abs($level) * $color-interval);
}

@function cms-color-opacity($color-name: 'primary', $level: 0) {
  $color: cms-color($color-name);

  @return transparentize($color, $level / 10);
}

@function cms-foreground-color($color-name, $threshold: .65) {
  $color: cms-color($color-name);

  @return if(lightness($color) >= $threshold, cms-color('foreground'), cms-color('background'));
}
```

```
/***
 * Class TaskController
 * @package App\Controller
 * @Route("/admin/tasks", name="tasks_")
 */
class TaskController extends AbstractController
{
    /**
     * @Route("/", name="index")
     */
    public function index()
    {
        $tasks = $this->getDoctrine()->getRepository(Task::class)->findAndOrderByStatus();
        $periods = $this->getDoctrine()->getRepository(Period::class)->findAll();
        $alert = null;
        if(!$periods) {
            $alert = [
                "type" => "error",
                "message" => "Maak een periode aan voordat je een taak toevoegd",
                "icon" => "bx:bx-s-error-alt"
            ];
        }
        return $this->render('admin/task/index.html.twig', [
            'tasks' => $tasks,
            'title' => 'Taken',
            'alert' => $alert
        ]);
    }

    /**
     * @Route("/new", name="add")
     * @param Request $r
     * @return Response
     */
    public function add(Request $r) {
        $task = new Task();
        $form = $this->createForm(TaskType::class, $task);

        $form->handleRequest($r);
        if($form->isSubmitted() && $form->isValid()) {
            $task = $form->getData();

            // Save to db
            $em = $this->getDoctrine()->getManager();
            $em->persist($task);
            $em->flush();

            return $this->redirectToRoute('tasks_index');
        }

        return $this->render('admin/task/edit.html.twig', [
            'form' => $form->createView(),
            'title' => 'Nieuwe Taak toevoegen'
        ]);
    }
}
```

```

class UserType extends AbstractType
{
    public function buildForm(FormBuilderInterface $builder, array $options)
    {
        $builder
            ->add('first_name', TextType::class, [
                'label' => 'Voornaam',
                'required' => true,
            ])
            ->add('last_name', TextType::class, [
                'label' => 'Naam',
                'required' => true,
            ])
            ->add('email', EmailType::class)
            ->add('password', RepeatedType::class, [
                'type' => PasswordType::class,
                'invalid_message' => 'Beide wachtwoorden moeten hetzelfde zijn',
                'required' => true,
                'first_options' => [
                    'label' => 'Wachtwoord',
                ],
                'second_options' => [
                    'label' => 'Herhaal wachtwoord',
                ],
            ])
            ->add('roles', ChoiceType::class, [
                'choices' => [
                    'Admin' => 'ROLE_ADMIN',
                    'Klant' => 'ROLE_CUSTOMER',
                    'Werknemer' => 'ROLE_EMPLOYEE',
                    'Onderaannemer' => 'ROLE_SUBCONTRACTOR'
                ],
                'multiple' => true,
                'expanded' => true,
                'attr' => [
                    'class' => 'form__group__roles'
                ]
            ])
            ->add('Cost', MoneyType::class, [
                'label' => 'Tarief',
                'required' => true,
                'empty_data' => 0,
                'currency' => false,
            ])
            ->add('transport', MoneyType::class, [
                'label' => 'Transportkost',
                'required' => true,
                'currency' => false,
                'empty_data' => 0,
            ])
            ->add('save', SubmitType::class, [
                'attr' => [
                    'class' => 'btn -primary',
                ]
            ])
        ;
    }

    public function configureOptions(OptionsResolver $resolver)
    {
        $resolver->setDefaults([
            'data_class' => User::class,
        ]);
    }
}

```

```




<div>
    <h1>{{ users|length }} Gebruiker{{ if users|length > 1 or users|length == 0 }}s{{ endif }}</h1>
    </div>
    <a href="{{ path('user_new') }}" class="btn -primary"><span class="iconify" data-icon="gridicons:user-add" data-inline="true"></span> Toevoegen</a>
    <table class="table">
        <thead>
            <tr>
                <th>ID</th>
                <th>Naam</th>
                <th>Gebruikersnaam</th>
                <th>Email</th>
                <th>Rollen</th>
                <th>Tarief</th>
                <th>Acties</th>
            </tr>
        </thead>
        <tbody>
            {% for user in users %}
            <tr>
                <td class="number">{{ user.id }}</td>
                <td>{{ user.firstName }} {{ user.lastName }}</td>
                <td>{{ user.username }}</td>
                <td>{{ user.email }}</td>
                <td>{{ user.roles[0][5:]|title }}</td>
                <td class="number">€ {{ user.cost | format_number({fraction_digit: 2}) }} <br> <span class="text-light">€ {{ user.transport | format_number({fraction_digit: 2}) }}</span></td>
                <td class="actions"><a href="{{ path('user_edit', { 'user': user.id }) }}">Bewerken</a> | <a href="{{ path('user_destroy', { 'user': user.id }) }}" class="danger">Verwijder</a></td>
            </tr>
            {% endfor %}
        </tbody>
    </table>
    {% endblock %}

```

DEVELOP – CODE SNIPPETS

```
public function findOneByClient($client, $start_date, $end_date) {
    return $this->createQueryBuilder('p')
        ->join('p.tasks', 't')
        ->join('p.client', 'c')
        ->addSelect(['c', 't'])
        ->where('c.username = :client AND p.start_date = :start AND p.end_date = :end')
        ->setParameters(['client' => $client, 'start' => $start_date, 'end' => $end_date])
        ->orderBy('t.date', 'asc')
        ->getQuery()
        ->getOneOrNullResult();
}
```

```
class CircularReferenceHandler {
    /**
     * @var RouterInterface
     */
    private $router;

    public function __construct(RouterInterface $router)
    {
        $this->router = $router;
    }

    public function __invoke($object)
    {
        switch ($object) {
            case $object instanceof Period:
                return $this->router->generate('api_tasks_index', ['id' => $object->getId()]); break;
            default:
                return $object->getId();
        }
    }
}
```

```
class MailController extends AbstractController
{
    /**
     * @Route("/admin/{period}/email", name="mail_period")
     * @param Period $period
     * @param MailerInterface $mailer
     * @return JsonResponse
     * @throws TransportExceptionInterface
     */
    public function index(Period $period, MailerInterface $mailer, Request $r)
    {
        $referer = $r->headers->get('referer');
        if (!$referer) {
            $redirect = $this->generateUrl("page_index");
            return $this->redirect($redirect);
        } else {
            $redirect = $referer;
        }
        $mailURL = $this->generateUrl('page_show_period', [
            "client" => $period->getClient()->getUsername(),
            "start" => date_format($period->getStartDate(), 'Y-m-d'),
            "end" => date_format($period->getEndDate(), 'Y-m-d')
        ]);
        $email = (new TemplatedEmail())
            ->from(new Address('noreply@arte-tech.null', 'Arte-Tech'))
            ->to($period->getClient()->getEmail())
            ->subject("Afgeronde periode: ".date_format($period->getStartDate(), 'd-m-Y')."
                    .date_format($period->getEndDate(), 'd-m-Y'))
            ->htmlTemplate('mail/period.html.twig')
            ->context([
                'period' => $period,
                'url' => 'http://localhost:8000' . $mailURL
            ]);
        $response = $mailer->send($email);

        return $this->redirect($redirect);
    }
}
```

App

```

<template>
  <transition name="slide-up">
    <div class="home">
      <app-header>
        <h2>Overzicht</h2>
        <p>Laatste 7 dagen</p>
        <section class="overview__details">
          <p>{{ overview.cost }} EUR</p>
          <p>{{ overview.time }} uur</p>
        </section>
      </app-header>
      <section v-if="tasks" class="tasks">
        <h2>Recente taken</h2>
        <div v-for="task in tasks" :key="task.id">
          <card>
            <task-with-details :task="task" />
          </card>
        </div>
      </section>
    </div>
  </transition>
</template>

<style lang="scss" scoped>
.home {
  .overview {
    &__details {
      display: flex;
      justify-content: space-between;
      align-items: center;
      font-family: $header-font-family;
      font-size: 32px;
      height: 50%;
      font-weight: $header-font-weight;
    }
  }
  .tasks {
    padding: 0 16px;
    text-align: left;
  }
  .tasks {
    h2 {
      color: app-color-level();
    }
  }
}
</style>

<script>
import Card from "@/components/Card";
import TaskWithDetails from "@/components/card-content/TaskWithDetails";
import AppHeader from "@/components/AppHeader";

export default {
  name: "home",
  components: {
    Card,
    TaskWithDetails,
    AppHeader
  },
  computed: {
    tasks() {
      return this.$store.getters.pastTasks;
    },
    overview() {
      return this.$store.getters.overview;
    }
  }
}
</script>

```

```

const store = new Vuex.Store({
  state: {
    userToken: null,
    userId: null,
    tasks: null,
    taskFlow: false
  },
  mutations: {
    setUserId: (state, id) => (state.userId = id),
    setToken: (state, token) => (state.userToken = token),
    clearToken: state => (state.userToken = null),
    setTasks: (state, tasks) => (state.tasks = tasks),
    setTaskFlow: (state, taskFlow) => (state.taskFlow = taskFlow)
  },
  getters: {
    pastTasks: state =>
      state.tasks
        ? state.tasks.filter(task => {
            const taskDate = new Date(task.date);
            const today = new Date();
            return (
              (taskDate < today && taskDate.getDate() < today.getDate()) ||
              task.status == 1
            );
          })
        : null,
    unfinishedTasks: state =>
      state.tasks
        ? state.tasks.filter(task => {
            const taskDate = new Date(task.date);
            const today = new Date();
            return taskDate.getDate() == today.getDate() &&
              taskDate.getMonth() == today.getMonth() &&
              taskDate.getFullYear() == today.getFullYear() &&
              task.status == 0
            ? true
            : false;
          })
        : null,
    overview: (state, getters) => {
      // Get tasks for the past period
      const lastWeekTasks = getters.pastTasks.filter(task => {
        const earliest = new Date().getDay() - 7;
        return new Date(task.date).getDate() > earliest;
      });
      // Calculate cost and hours for these tasks
      let cost = 0;
      let time = 0;
      lastWeekTasks.forEach(task => {
        cost += task.cost;
        time += new Date(task.end_time).getHours() -
          new Date(task.begin_time).getHours();
      });
      // Return as object
      return {
        cost,
        time
      };
    }
  },
  actions: {
    async fetchJWT(userdata) {
      const { data } = await axios.post(
        "http://localhost:8000/api/token",
        userdata,
        {
          timeout: 5000
        }
      );
      return data;
    },
    async initApp({ commit }, userdata) {
      const { data: user } = await axios.post(
        "http://localhost:8000/api/token",
        userdata,
        {
          timeout: 5000
        }
      );
      const { data } = await axios.get(
        "http://localhost:8000/api/tasks/${user.user}",
        {
          headers: {
            Authorization: `Bearer ${user.token}`
          }
        }
      );
      commit("setToken", user.token);
      commit("setUserId", user.user);
      commit("setTasks", data);
    }
  }
});

```

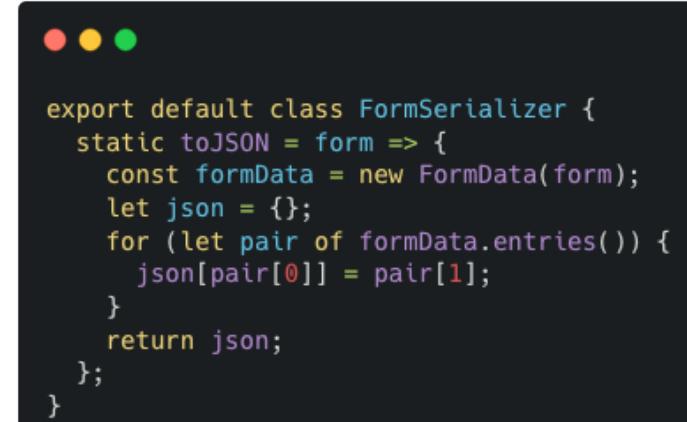
```

<template>
  <form id="taskForm" method="post">
    <div v-if="errors">
      <form-error
        :level="errors[0].level"
        :code="errors[0].code"
        :message="errors[0].message"
      />
    </div>
    <div class="form__row">
      <text-field>
        <label>
          labelText="Startuur"
          type="time"
          name="begin_time"
          required
          class="half"
        />
      </text-field>
      <text-field>
        <label>
          labelText="Einduur"
          type="time"
          name="end_time"
          required
          class="half"
        />
      </text-field>
      <select-field>
        <label>
          labelText="Pauze"
          fieldName="break"
          :options="selectOptions"
          class="half"
        />
        <option>Geen pauze</option>
        <option>15 minuten</option>
        <option>30 minuten</option>
        <option>45 minuten</option>
        <option>1 uur</option>
      </select-field>
    </div>
    <div class="form__row">
      <text-area-field label="Activiteiten" fieldName="activity" />
      <text-area-field label="Materiaal" fieldName="materials" />
      <button type="submit" colors="primary">Voltooien</button>
    </div>
  </form>
</template>

<style lang="scss" scoped>
.form {
  &__row {
    width: 100px;
    display: flex;
    justify-content: space-between;
    box-sizing: border-box;
  }
  .half {
    flex: 0 0 50px;
    padding: 0 2px;
    box-sizing: border-box;
  }
  &:first-child {
    padding-left: 0;
  }
  &:last-child {
    padding-right: 0;
  }
}
</style>

<script>
  import TextField from './fields/textField';
  import TextAreaField from './fields/textAreaField';
  import SelectField from './fields/selectField';
  import FormError from './FormError';
  import Button from '../Button';
  export default {
    name: 'TaskForm',
    props: {
      errors: Array
    },
    components: {
      TextField,
      TextAreaField,
      SelectField,
      FormError,
      Button
    },
    data: () => {
      return {
        selectOptions: [
          {label: "Geen pauze", value: 0},
          {label: "15 minuten", value: 0.25},
          {label: "30 minuten", value: 0.5},
          {label: "45 minuten", value: 0.75},
          {label: "1 uur", value: 1}
        ]
      };
    }
  };
</script>

```



```

export default class FormSerializer {
  static toJSON = form => {
    const formData = new FormData(form);
    let json = {};
    for (let pair of formData.entries()) {
      json[pair[0]] = pair[1];
    }
    return json;
  };
}

```



```

export default class Validator {
  static validate(values) {
    for (let [key, value] of Object.entries(values)) {
      if (!value) {
        return {
          isValid: false,
          errorMsg: `${key} cannot be empty.`,
          field: key
        };
      }
    }
    return {
      isValid: true,
      errorMsg: null
    };
  }

  static validateTaskDetail(task) {
    let errors = [];
    const begin = new Date(task.begin_time);
    const end = new Date(task.end_time);

    for (let [key, value] of Object.entries(task)) {
      if (!value) {
        errors.push({
          level: "error",
          code: null,
          message: `${key} kan niet leeg zijn`
        });
      }
    }

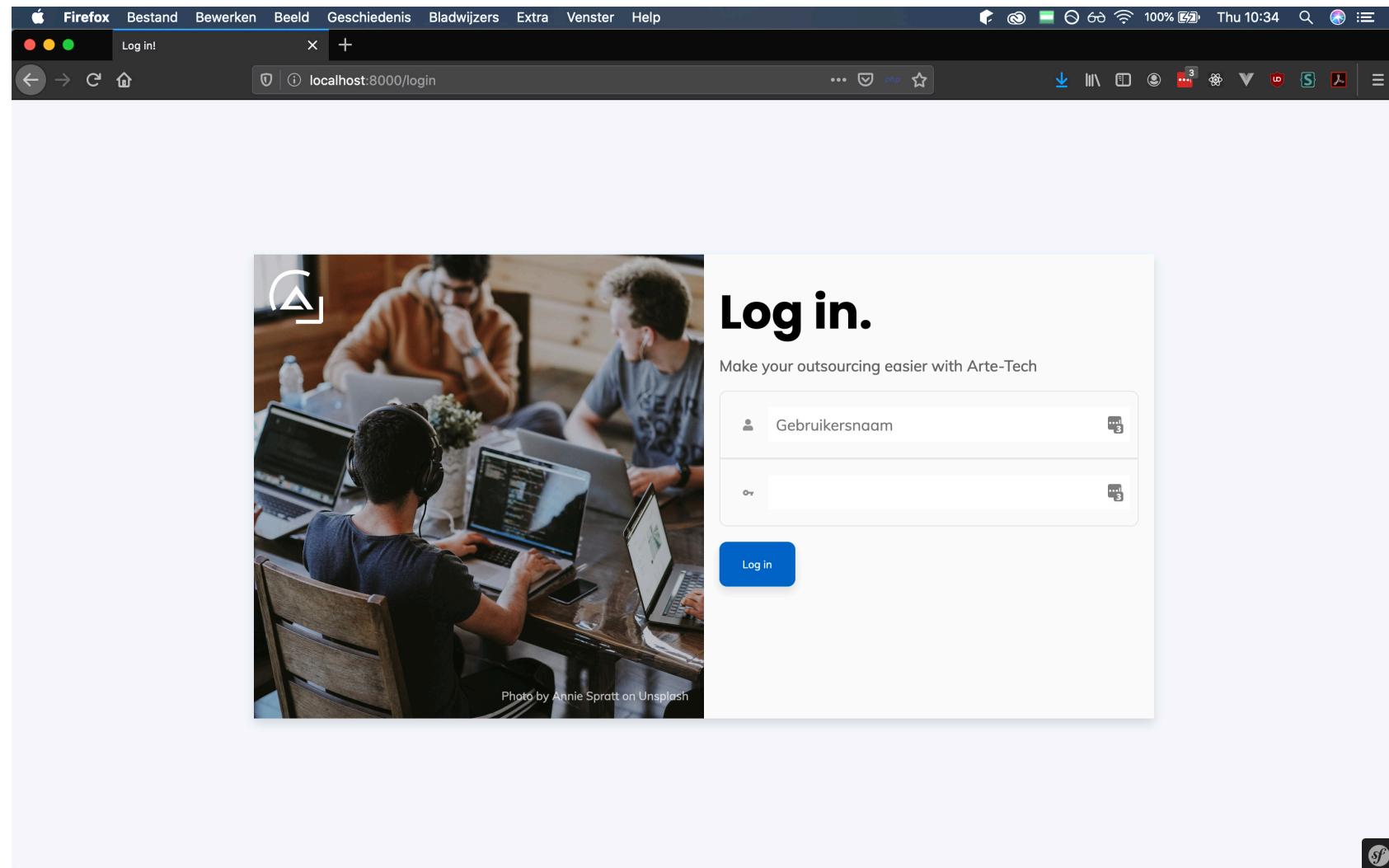
    if (begin.getTime() > end.getTime())
      errors.push({
        level: "error",
        code: null,
        message: "Einduur moet later zijn dan beginuur"
      });

    if (errors.length > 0)
      return {
        isValid: false,
        errors: errors
      };

    return {
      isValid: true,
      errors: null
    };
  }
}

```

Schermafbeeldingen



Arte-Tech CMS

Opbrengst **EUR** +308 EUR

Voltooide taken **0** -1 taken

Top Werknemer **Wouter Vlaeyen**

Taken per dag

Datum	Aantal Taken
13 Dec	1
23 Dec	2
24 Dec	1
27 Dec	1
30 Dec	2

Recente opmerkingen

Periode	Opmerking	Datum	Acties
09-12-2019 — 16-12-2019 Lang Bedrijf	Hier is nog een opmerking	30-12-2019	Bekijken



The screenshot shows a web-based application interface for managing users. On the left is a vertical sidebar with a blue header containing a logo and navigation items: Overzicht, Gebruikers (selected), Taken, Periodes, and Log Out. The main content area has a white header with the title "5 Gebruikers" and a blue "Toevoegen" button. Below is a table listing five users:

ID	Naam	Gebruikersnaam	Email	Rollen	Tarief	Acties
4	Wouter Vlaeyen	woutvlae	woutvlae@student.arteveldhehs.be	Admin	€ 60.00 € 0.26	Bewerken Verwijder
6	Lang Bedrijf	langbedr	lang@bedrijf.null	Customer	€ 65.00 € 0.00	Bewerken Verwijder
8	Nieuw Bedrijf	nieubedr	scrimpydata@gmail.com	Customer	€ 65.00 € 0.38	Bewerken Verwijder
2	Test User	testuser	test@test.be	Employee	€ 45.00 € 0.00	Bewerken Verwijder
7	Jan Temmerman	jantemm	onder.aannemer@test.be	Subcontractor	€ 54.53 € 0.22	Bewerken Verwijder

The screenshot shows a user interface for adding a new user. On the left is a vertical navigation bar with icons and labels: Overzicht, Gebruikers (selected), Taken, Periodes, and Log Out. The main area has a title "Nieuwe gebruiker toevoegen". It contains fields for Email, Password, Re-enter password, First name, Last name, Roles (Admin, Klant, Werknemer, Onderaannemer), Tarief, Transportkost, and a Save button.

Nieuwe gebruiker toevoegen

Email

Wachtwoord

Herhaal wachtwoord

Voornaam

Naam

Roles

Admin

Klant

Werknemer

Onderaannemer

Tarief

Transportkost

Save



The screenshot shows a software application interface with a dark blue sidebar and a white main content area.

Sidebar (Left):

- Logo:** A white stylized 'A' icon inside a blue rounded square.
- Overzicht:** A blue button with a three-line menu icon.
- Gebruikers:** A blue button with a user icon.
- Taken:** A blue button with a checkmark icon, currently selected.
- Periodes:** A blue button with a clock icon.
- Log Out:** A blue button with a power-off icon.

Main Content Area:

7 Taken

[+ Toevoegen](#)

ID	Klant	Technieker	Datum	Status	Duur	Prijs
9	Lang Bedrijf	Wouter Vlaeyen	30/12/2019	Geannuleerd	-	-
10	Lang Bedrijf	Jan Temmerman	30/12/2019	Geannuleerd	-	-
8	Lang Bedrijf	Wouter Vlaeyen	27/12/2019	Geannuleerd	-	-
5	Lang Bedrijf	Wouter Vlaeyen	24/12/2019	Voltooid	11:00 - 16:00	€308.00
6	Lang Bedrijf	Test User	23/12/2019	Geannuleerd	-	-
7	Lang Bedrijf	Wouter Vlaeyen	23/12/2019	Geannuleerd	-	-
4	Lang Bedrijf	Wouter Vlaeyen	13/12/2019	Geannuleerd	-	-



23 December – 31 December

Lang Bedrijf

lang@bedrijf.null

Goedkeuren

PDF Maken

Datum	Technieker	Tijd	Prijs
23/12/2019	Test User	Geannuleerd	€0.00
23/12/2019	Wouter Vlaeyen	Geannuleerd	€0.00
24/12/2019	Wouter Vlaeyen	10:00 - 15:00	€308.00
27/12/2019	Wouter Vlaeyen	Geannuleerd	€0.00
30/12/2019	Wouter Vlaeyen	Geannuleerd	€0.00
30/12/2019	Jan Temmerman	Geannuleerd	€0.00
Totaal			€308.00

Geen opmerkingen



DEVELOP – SCHERMAFBEELDINGEN

The image displays three screenshots of a mobile application interface for Arte-Tech.

Screenshot 1: Login Screen

Meld je aan om de applicatie te gebruiken.

Gebruikersnaam

Wachtwoord

Log in

Screenshot 2: Dashboard

Overzicht
Laatste 7 dagen

555 EUR 5 uur

Recente taken

Lang Bedrijf 24 december
Testen van applicaties 11:00 — 16:00

Screenshot 3: Task Completion Screen

Taak voltooien

Nieuw Bedrijf gisteren

A back arrow icon is located above the task completion screen.

←

Taak voltooien

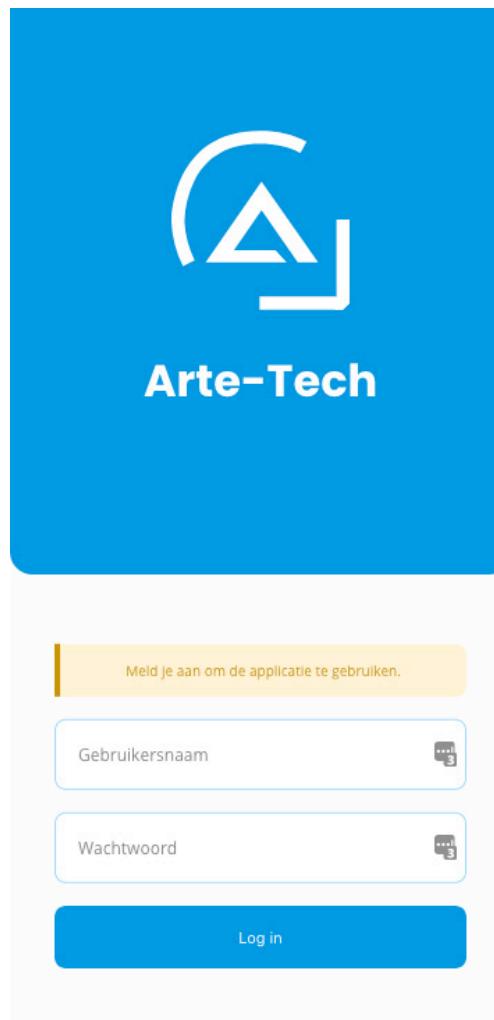
Nieuw Bedrijf
2 januari 2020

Startuur Einduur
 :

Pauze Transport
 Geen pauze Transport

Activiteiten

Materialen



Handleiding

CMS

De Arte-Tech CMS is een gebouwd om op een makkelijke manier verschillende acties uit te voeren. Deze handleiding kan gebruikt worden als extra gids om deze acties uit te voeren.

1. Inloggen en de overzichtspagina

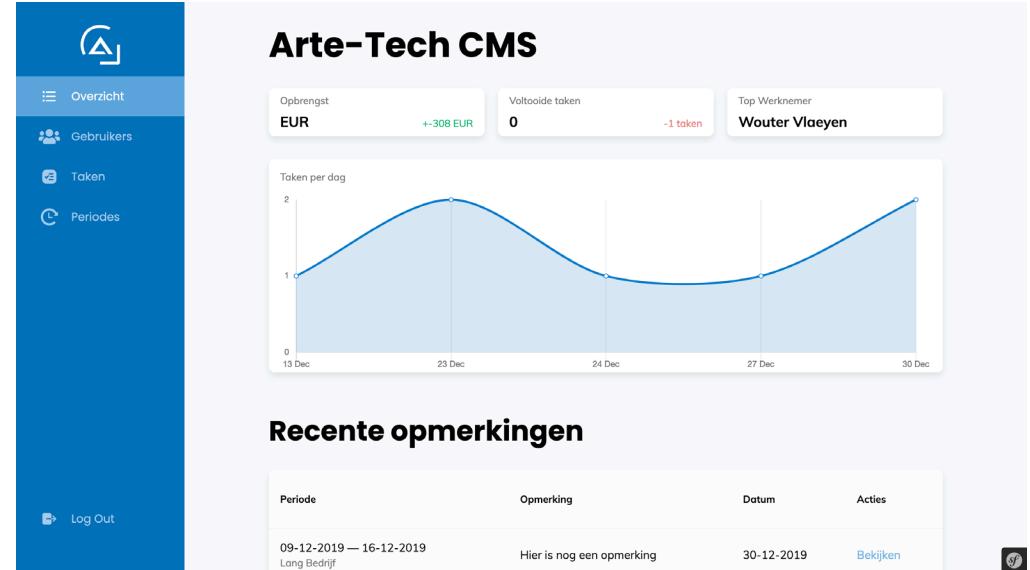
Om in te loggen in de CMS navigeer je naar </admin>. Als je nog niet bent ingelogd zal je omgeleid worden via de login pagina.

Op deze pagina word je gevraagd om je gebruikersnaam en wachtwoord in te geven.

Eenmaal ingelogd kom je op de overzichtspagina terecht.
Op deze pagina vind je een snelle blik aan de meest recente informatie terug.

De bovenste rij toont een overzicht van de vorige week met totale opbrengst, voltooide taken en best verdienende werknemer.

De volgende rij toont een grafiek van gegeven taken, en onderaan vinden we een overzicht van recente opmerkingen.



De overzichtspagina van de CMS

2. Gebruikers, Taken en Periodes

In de sidebar aan de linkerzijde van het scherm vind je de verschillende soorten data terug: Gebruikers, Taken en Periodes. Elk van deze data heeft een eigen overzicht.

Bij de gebruikers vind je hier IDs, namen, emailadressen , rollen en tarieven terug. Voor taken is dit de klant, werknemer, status duur en prijs, en bij de periodes is dit de klant, duur van de periode en status.

Bij elke soort data staan ook enkele snelle acties die kunnen uitgevoerd worden voor deze data, zoals verwijderen, details bekijken of e-mails versturen.

3. Data toevoegen

Data toevoegen is net zo makkelijk als de snelle acties. Onder de titel staat een knop om nieuwe data toe te voegen. Dit brengt je naar het formulier waar je makkelijk nieuwe data kan ingeven en toevoegen aan de database.

4. Overzichten bekijken

De periodes en taken hebben nog een eigen manier om overzichten te bekijken. Bij elke taak en periode zal je een optie zien staan om deze te bekijken. Door hier op te klikken zal je een meer uitgebreid overzicht van de taak of periode te zien krijgen.



5 Gebruikers

Toevoegen

ID	Naam	Gebruikersnaam	Email	Rollen	Tarief	Acties
4	Wouter Vlaeyen	woutvlae	woutvlae@student.arteveldhehs.be	Admin	€ 60.00 € 0.26	Bewerken Verwijderen
6	Lang Bedrijf	langbedr	lang@bedrijf.null	Customer	€ 65.00 € 0.00	Bewerken Verwijderen
8	Nieuw Bedrijf	nieubedr	scrimpydata@gmail.com	Customer	€ 65.00 € 0.38	Bewerken Verwijderen
2	Test User	testuser	test@test.be	Employee	€ 45.00 € 0.00	Bewerken Verwijderen
7	Jan Temmerman	jantemm	onder.aannemer@test.be	Subcontractor	€ 54.53 € 0.22	Bewerken Verwijderen

De overzichtspagina voor gebruikers



Nieuwe gebruiker toevoegen

E-mail

Wachtwoord

Herhaal wachtwoord

Voornaam

Naam

Roles

 Admin
 Klant
 Werknemer
 Onderaannemer

Tarief

Transportkost

Save

Formulier om nieuwe gebruikers toe te voegen

DELIVER – HANDLEIDING

In het geval van de periodes zal je dezelfde pagina als de klant te zien krijgen.

App

1. Inloggen en overzicht

Wanneer je de applicatie probeert te gebruiken zonder account zal je automatisch worden omgeleid naar de loginpagina.

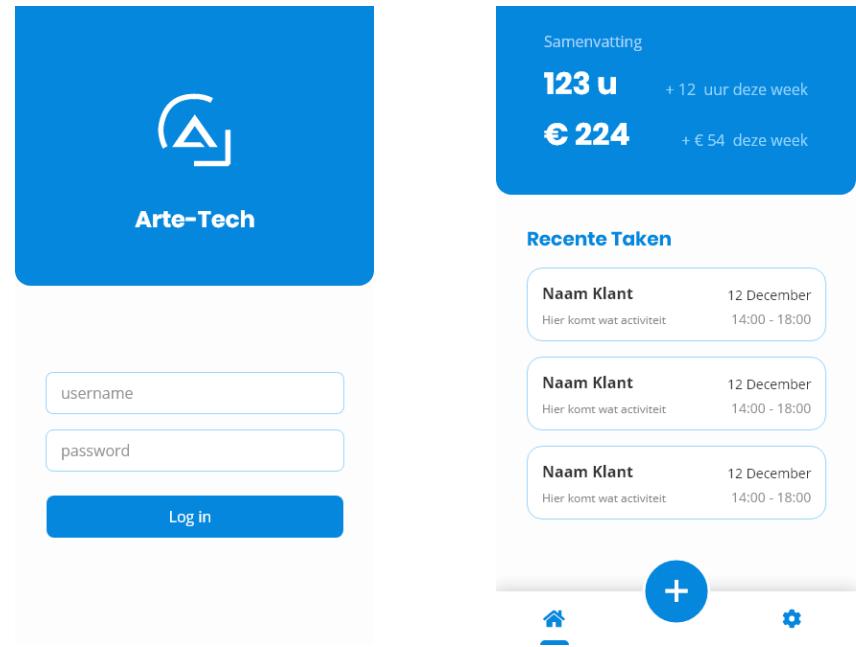
Net zoals bij de CMS zal hier gevraagd worden om in te loggen met je gebruikersnaam en wachtwoord.

Na het inloggen kom je op de overzichtspagina van de werknemer terecht. Hier vind je een overzicht van de laatste week terug, samen met een samenvatting van uren en verdiend loon.

2. Taken afwerken

Om een taak af te werken begin je met een klik op de knop om een taak toe te voegen (ofwel de + knop of de “Voeg toe” knop).

Het proces om een taak toe te voegen gebeurd in 2 stappen. De eerste stap is het kiezen van een actieve taak die nog niet afgewerkt is.



Login en overzichtspagina van de app

Hierna kan de werknemer de nodige informatie aanvullen en de geupdate taak versturen.

3. Instellingen aanpassen

Als de werknemer een onderaannemer is zijn er enkele instellingen die hij kan aanpassen.

Door op het gear-icon te klikken kom je op het instellingenschermscherm. Hier kan je via het formulier je tarief en transportkost aanpassen.

Een gewone werknemer kan deze niet aanpassen en zal dus ook het formulier niet zien.

Taal voltooien

Naam Klant	Datum
Naam Klant	12 December
Naam Klant	12 December
Naam Klant	12 December

Taal voltooien

Naam klant	12 December 2019
Startuur	Einduur
10:00	18:00
Pauze	Transport
15 minuten	25 km
Activiteiten	
Heel veel activiteiten	
Materialen	
Helemaal niets	

Voltooi

Schermen om een nieuwe taak toe te voegen

Deployment Guide

1. Inleiding

Deze deployment guide gaat ervan uit dat bepaalde programma's al geïnstalleerd zijn op de computer:

- NPM
- Composer
- Git

Indien dit niet het geval is wordt aangeraden om te zorgen dat deze eerst geïnstalleerd worden.

2. Repository

Het hele project is te vinden op de volgende Github repository:

<https://github.com/Vl-Wouter/eindopdracht-cmsdev-Vl-Wouter/>

Om dit project lokaal te kunnen gebruiken kopieer je het volgende commando in de terminal:

```
git clone https://github.com/Vl-Wouter/eindopdracht-cmsdev-Vl-Wouter.git
```

CMS

1. Setup

Installeer alle dependencies door het volgende commando in de map `/cms` uit te voeren:

```
composer install
```

Zonder de dependencies kan het project niet werken.

2. Environment Variables

Environment Variables zijn ook belangrijk voor het werken van het project. Deze file bevat alle belangrijke keys voor API's, dependencies, etc...

Symfony komt standaard met een `.env` file waar je alle waarden kan aanvullen.

3. Lokaal Draaien

Om de CMS en back-end lokaal te draaien moet je in de terminal het volgende commando ingeven;

```
symfony serve
```

4. Pushen naar productie

Symfony heeft goede documentatie voor het deployen van

een project:

<https://symfony.com/doc/current/deployment.html>

App

1. Setup

De applicatie is gemaakt met Vue.js als front-end framework. Installeer alle nodige dependencies door het volgende commando uit te voeren in de map `/app`:

```
npm install
```

2. Environment Variables

Omdat de enige beveiligde keys worden aangemaakt tijdens het draaien van de applicatie, moet voor deze geen `.env` file aangemaakt of aangevuld worde.

3. Lokaal Draaien

Tijdens het verdere development van de applicatie kan deze lokaal draaien door in de terminal het volgende in te geven:

```
npm run serve
```

Hiermee zal Vue een lokale server opstarten die automatisch update wanneer iets veranderd aan de app.

4. Pushen naar productie

Vue.js kan gebuild worden, wat dus betekent dat dit op elke vorm van static web hosting kan draaien. Dit zorgt ervoor dat dit ook op bijna elke vorm van hosting werkt.

Wat wel steeds moet gebeuren is het builden (wanneer dit niet door een host (bv. Netlify) gedaan wordt).

Dit doe je door het volgende commando in te geven:

```
npm run build
```

Hierna kan je de files in de `/dist` folder uploaden naar de host.