PRETASK

DOKUMENTATION

ÜBER

das Projekt 1

VON

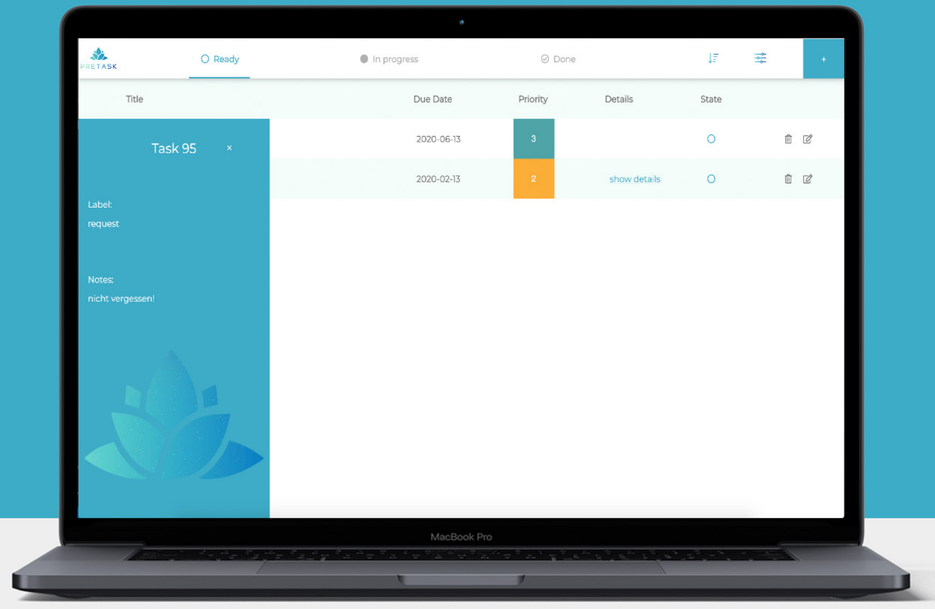
Elli Pensel

UNTER DER BETREUUNG VON

Prof. Dr. Marius Hofmeister

Prof. Jürgen Graef

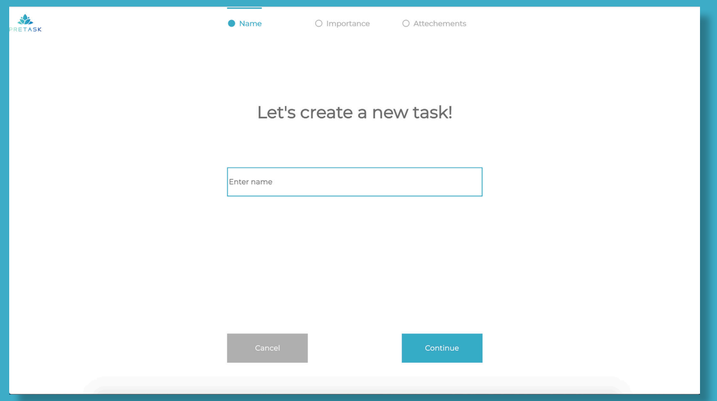
> *A place for everything,*
> *everything in its place.*
>
> ~ Benjamin Franklin

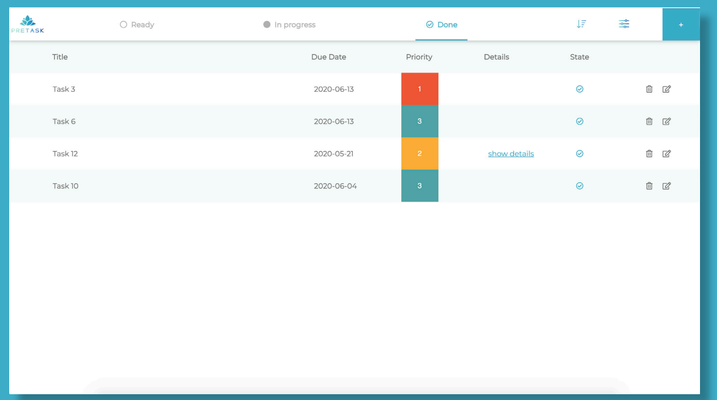


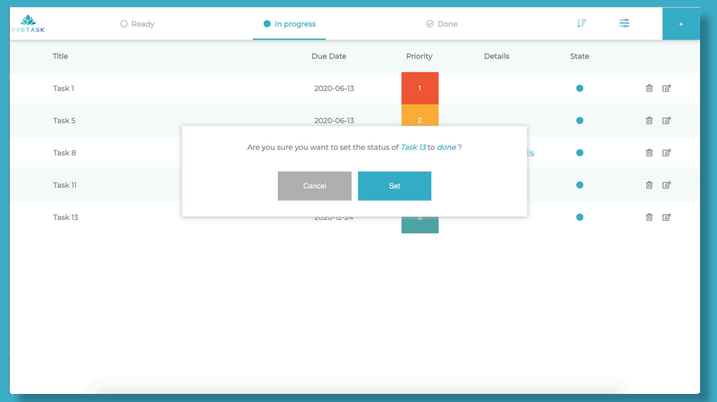
PRETASK

PROJEKTIDEE

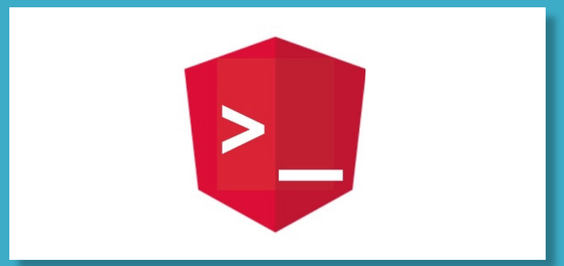
Pretask ist eine Taskverwaltungs-Webapplikation, die die Abarbeitung von Aufgaben erleichtern soll. Sie bietet die Möglichkeit, schnell und einfach neue Tasks zu erstellen und diese mit Labels, Start- sowie Enddaten, Prioritäten und Notizen zu individualisieren. Der Arbeitsprozess kann schnell und einfach dokumentiert und abgeschlossene Aufgaben abgehakt werden. Zudem kann der Nutzer die Taskliste beliebig sortieren und filtern, Tasks im Nachhinein bearbeiten oder löschen. Die Webapplikation Pretask soll zur Unterstützung dienen, indem sie einen klaren und strukturierten Überblick über Aufgaben bietet.

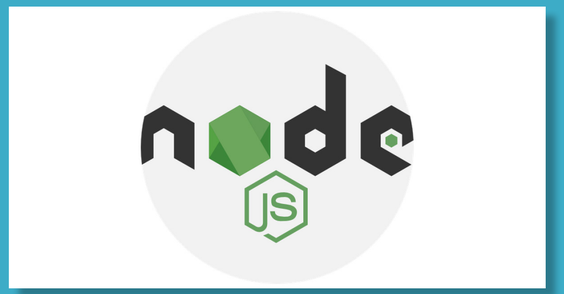






PRETASK





JSON Server

TECHNOLOGIEN

Angular

Angular CLI

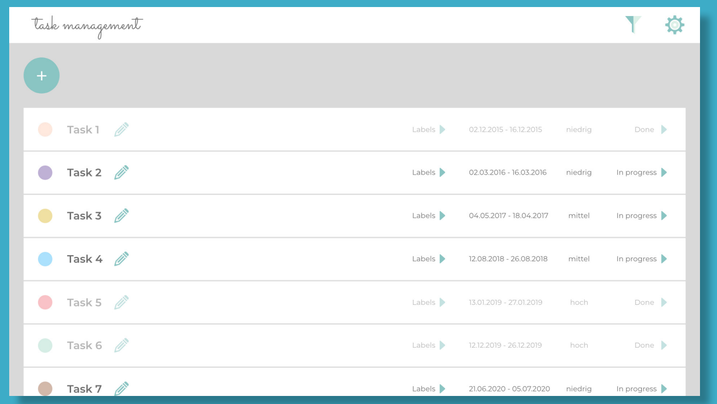
Node.js

JSON Server

PRETASK

UMSETZUNG

Für die Umsetzung der Webapplikation Pretask wurde das Front-End-Webapplikationsframework Angular, das Kommandozeilenwerkzeug Angular CLI, Node.js sowie ein JSON Server für das Backend verwendet.
Zu Beginn des Projektes wurden verschiedene Konzepte sowie Designs erarbeitet und immer wieder verbessert. Der daraus entstandene Designprototyp wurde anschließend technisch umgesetzt. Zur Planung und Einteilung von Aufgaben bot die Website Mondays.com eine hilfreiche Unterstützung.



PRETASK

```ts
import { Component, OnInit, Input, Output, EventEmitter } from '@angular/core';
import { Task } from '../../shared/task';
import { TaskmanagementService } from '../../shared/taskmanagement.service';

@Component({
  selector: 'bm-task-list-item',
  templateUrl: './task-list-item.component.html',
  styleUrls: ['./task-list-item.component.css']
})
export class TaskListItemComponent implements OnInit {

  @Input() task: Task;
  @Input() odd: boolean;
  @Input() even: boolean;

  @Output() detail = new EventEmitter<Task>();
  @Output() edit = new EventEmitter<Task>();
  @Output() confirm = new EventEmitter<Task>();
  @Output() delete = new EventEmitter<Task>();
  @Output() issue = new EventEmitter<string>();

  constructor(private tm: TaskmanagementService) {}

  ngOnInit() {}

  deleteTask(task: Task) {
    this.delete.emit(task);
  }
```

```
  </div>
  <div *ngIf="state === 'home'" class="home">
    <p>A place for everything, everything in its plase.</p>
    <p id="author">~ Benjamin Franklin</p>
  </div>

  <bm-task-list-container
  [tasksShown]="tasksShown"
  [state]="state"
  (sendTask)="onDetail($event)"
  (edit)="onEdit($event)"
  (confirm)="onConfirm($event)"
  (delete)="onDelete($event)"
  (issue)="onIssue($event)"
  class="list">
  </bm-task-list-container>

  <bm-task-details *ngIf="task && showDetail === false && edit !== true"
  [task]="task"
  (showDetail)="onShowDetail($event)" class="detail">
  </bm-task-details>

  <bm-task-edit class="edit" *ngIf="task && edit"
  [task]="task" (edit)="onCanceled($event)"></bm-task-edit>

  <bm-task-confirm class="confirm" *ngIf="task && confirm"
  [task]="task" [issue]="issue"
  (confirm)="onCanceledConfirm($event)"
  (updated)="onUpdated($event)"
  ></bm-task-confirm>
```

```ts
  getSingle(id: string): Observable<Task> {

    return this.http.get<Task>(`${environment.taskEndpoint}/${id}`);

  }

  create(task: Task): Observable<Task> {

    return this.http.post<Task>(environment.taskEndpoint, task);

  }

  update(task: Task): Observable<Task> {

    return this.http.put<Task>(`${environment.taskEndpoint}/${task.id}`, task);

  }

  delete(task: Task): Observable<Task> {

    return this.http.delete<Task>(`${environment.taskEndpoint}/${task.id}`);

  }

  filterPriority(selectedFilter: string): Task[] {
    return this.tasks.filter(element => element.priority === selectedFilter);
  }

  filterLabel(selectedFilter: string): Task[] {
    return this.tasks.filter(element => element.label === selectedFilter);
  }
```

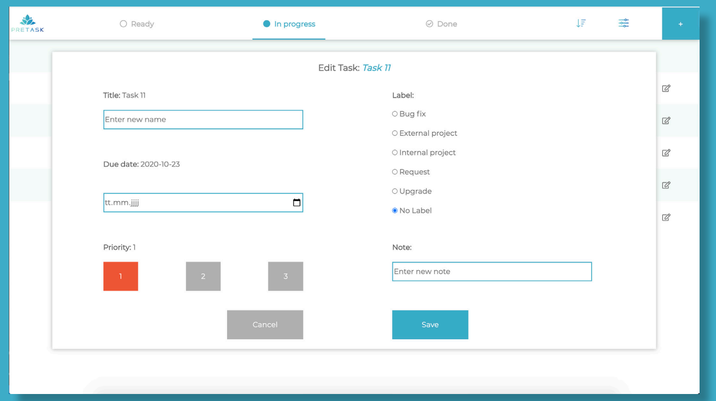
HERAUSFORDERUNGEN

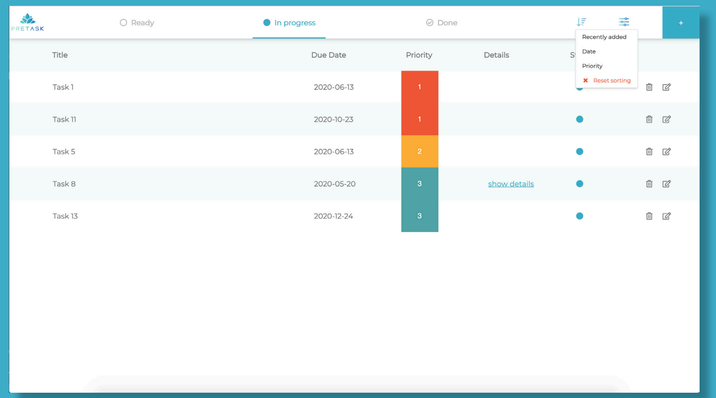
Die Umsetzung des Projektes brachte mehrer Hürden mit sich. Eine der größten Herausforderungen war die allgemeine Einarbeitung in Angular, insbesondere die Kommunikation zwischen den einzelenen Komponenten. Auch das richtige grundlegende Konzept zu finden, damit eine leichte und übersichtliche Bedienung gegeben ist, nahm einige Zeit in Anspruch. Eine weitere große Hürde was das Einbinden des Backends sowie die einzelnen HTTP-Requests, die erst nach mehreren Anläufen problemlos funktionierten. Doch mithilfe von Lehrbüchern, Viedokursen und vor allem der Unterstützung der betreuenden Professoren , war es möglich, Pretask wie geplant umzusetzen und alle Hürden erfolgreich zu meistern.

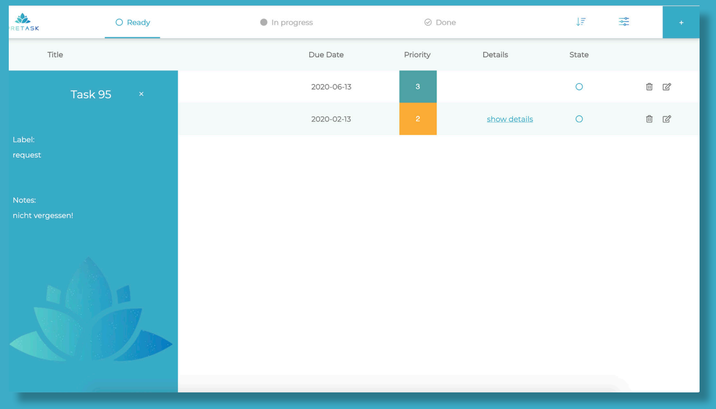
PRETASK

RÜCKBLICK

Zurückblickend war Pretask zwar ein sehr zeitaufwendiges Projekt, das einen oftmals vor Herausforderungen gestellt hat, doch hat es einem viel Spaß und Wissen beschert. Da ich noch nie zuvor mit Angular oder Node.js gearbeitet hatte, habe ich in den vergangenen 3 Monaten sehr viel dazu gelernt und ich bin stolz, was ich mit meinem neu erlangten Wissen umgesetzt habe. Ich möchte in Zukunft noch tiefer in die Richtung Webentwicklung eintauchen.



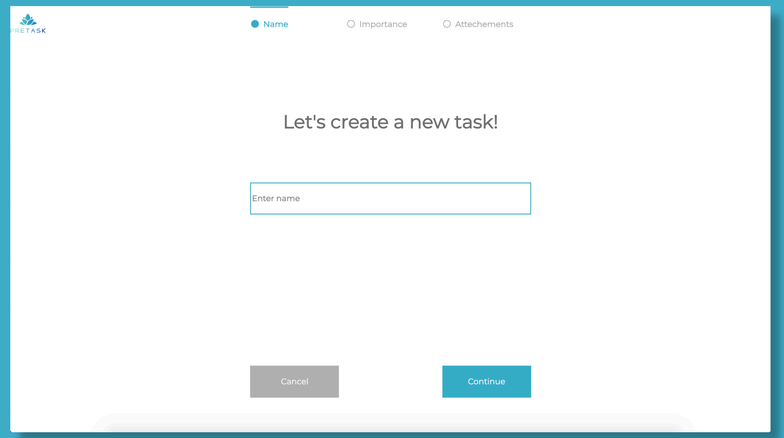


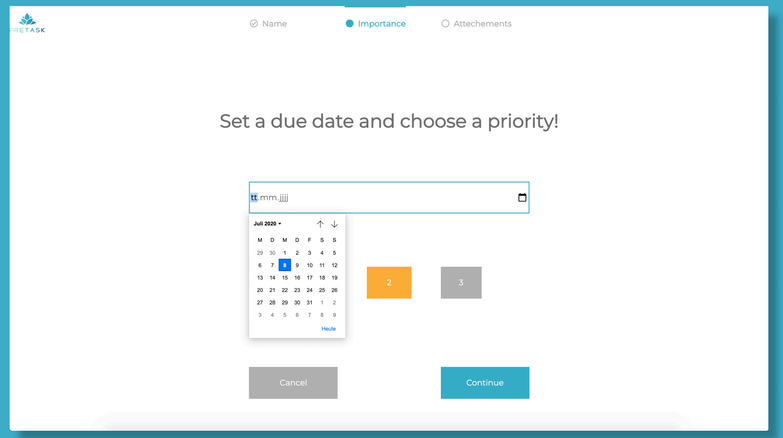


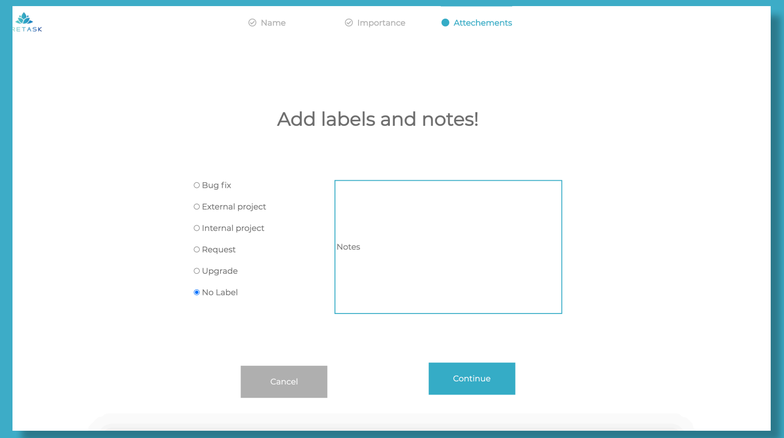
DURCHLAUF DURCH DIE WEBAPPLIKATION

PRETASK

NEUEN TASK ANLEGEN

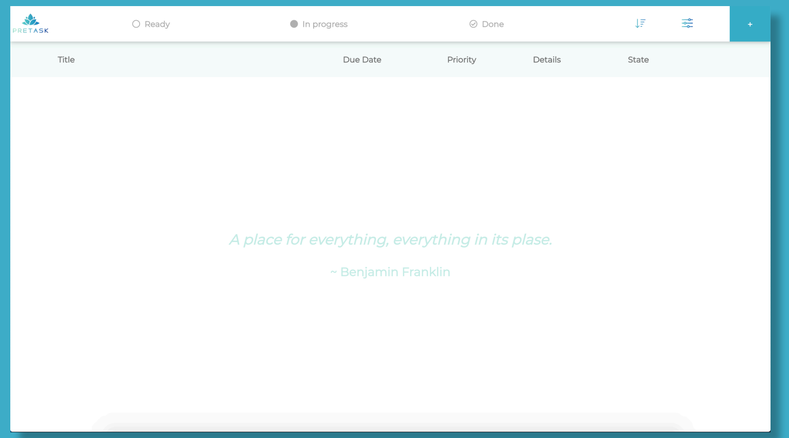


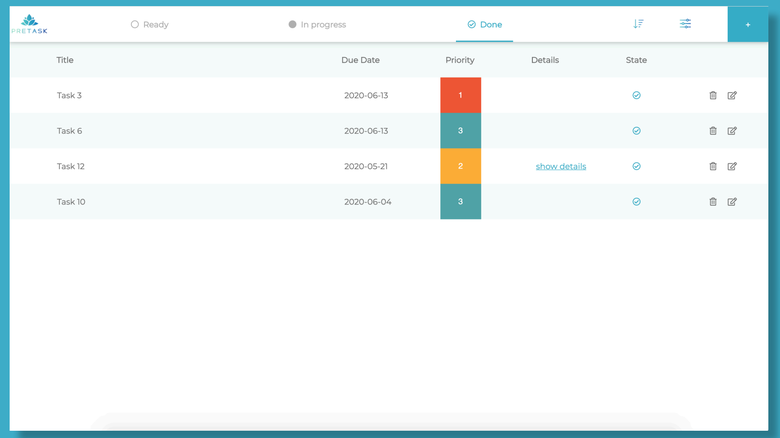


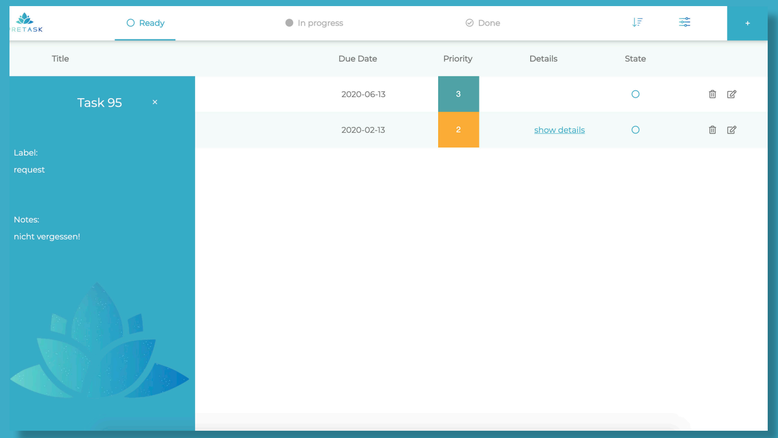


PRETASK

ALLGEMEINE ÜBERSICHT

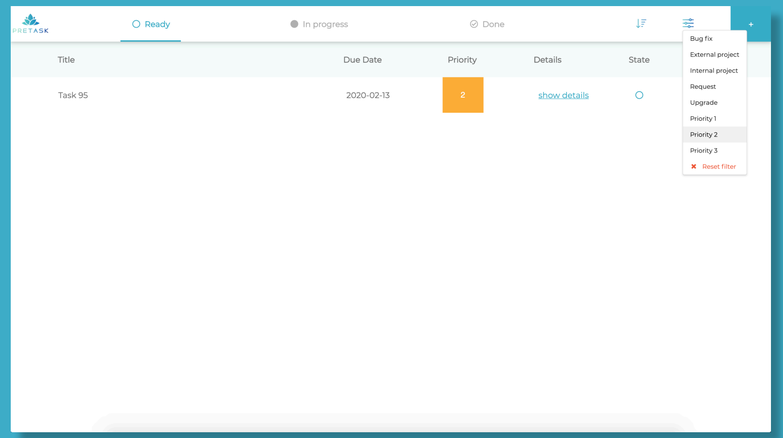


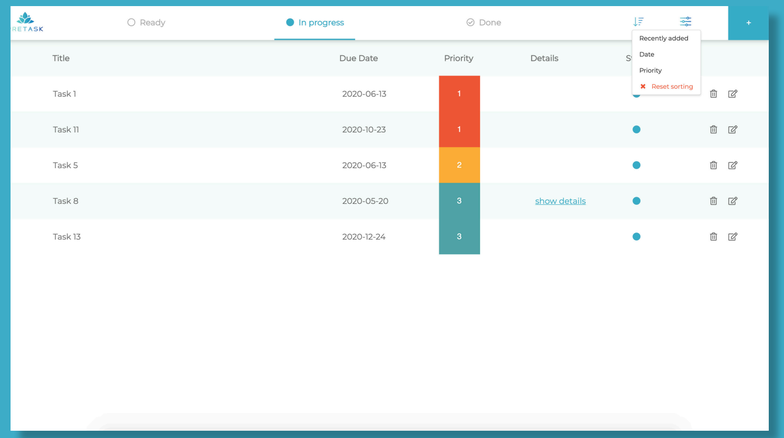


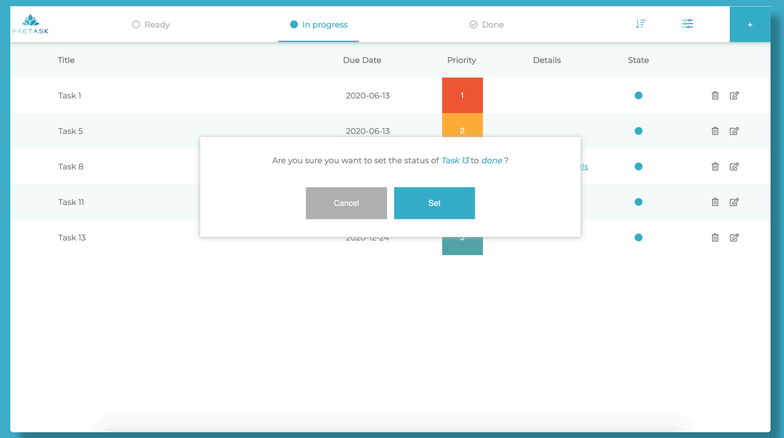
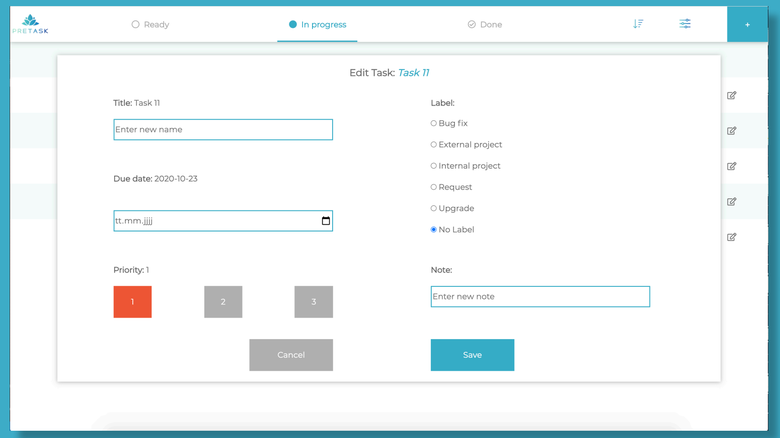
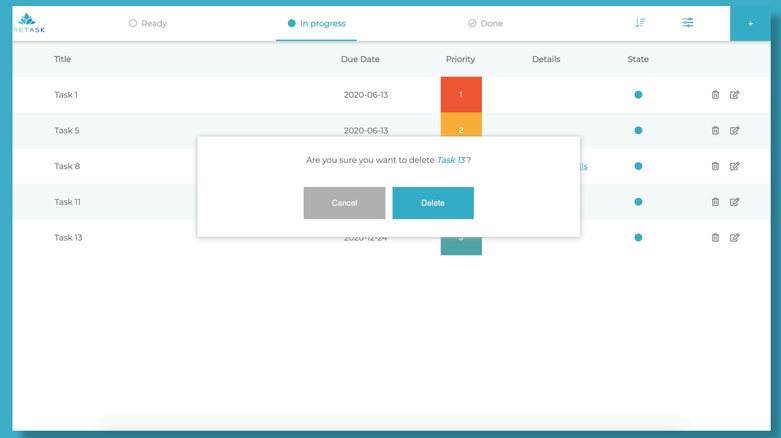


PRETASK

FILTERN UND SORTIEREN





PRETASK

STATUS ÄNDERN, TASK BEARBEITEN UND LÖSCHEN