```
<template>
  <!-- ToDO:
  1. Implementierung disable Checkbox -->
  <h2>@Empfängerkreis</h2>
  <p>
    Je nach Auswahl wird die Benachrichtigung an die administrierten
Verantwortlichen und Vertreter
    der OrgE sowie die Inhaber der gewählten Benutzerrollen gesandt.
  </p>
  <p v-if="selectedReceiver == 0" style="color: red">
    Bitte wähle mindestens eine OrgE oder Benutzerrolle aus
  </p>
  <div class="card-container">
    <div class="card-with-header">
      <h2>AN</h2>
      <scale-card class="half-width">
        <div class="checkbox-group-container">
          <scale-checkbox-group
            label="(Alle OrgE) - Ost PTI 31"
            name="AnOrgE"
            @change="handleSelectAllOrgE"
            :checked="allOrgESelected"
          >
            <scale-checkbox
              v-for="(item, index) in orgEItems"
              :key="item.value"
              :label="item.label"
              :value="item.value"
              v-model="selectedReceiver[index].selected"
              :disabled="isCcChecked(item.value)"
              @change="updateDisabledStates"
            />
          </scale-checkbox-group>
        </div>
        <div class="checkbox-group-container">
          <scale-checkbox-group
            label="(Alle Benutzerrollen) - Ost PTI 31"
            name="AnUser"
            @change="handleSelectAllUsers"
            :checked="allUsersSelected"
          >
            <scale-checkbox
              v-for="(item, index) in userItems"
              :key="item.value"
              :label="item.label"
              :value="item.value"
              v-model="selectedReceiver[8 + index].selected"
              :disabled="isCcChecked(item.value)"
              @change="updateDisabledStates"
            />
```

```
          </scale-checkbox-group>
        </div>
      </scale-card>
    </div>

    <div class="card-with-header">
      <h2>CC</h2>
      <scale-card class="half-width">
        <div class="checkbox-group-container">
          <scale-checkbox-group label="(Alle OrgE) - Ost PTI 31"
name="CcOrgE">
            <scale-checkbox
              v-for="(item, index) in orgEItems"
              :key="'cc-' + item.value"
              :label="item.label"
              :value="item.value"
              v-model="selectedCc[index].selected"
              :disabled="isAnChecked(item.value)"
              @change="updateDisabledStates"
            />
          </scale-checkbox-group>
        </div>
        <scale-checkbox-group label="(Alle Benutzerrollen) - Ost PTI 31"
name="CcUser">
          <scale-checkbox
            v-for="(item, index) in userItems"
            :key="'cc-' + item.value"
            :label="item.label"
            :value="item.value"
            v-model="selectedCc[8 + index].selected"
            :disabled="isAnChecked(item.value)"
            @change="updateDisabledStates"
          />
        </scale-checkbox-group>
      </scale-card>
    </div>
  </div>
</template>

<script>
export default {
  data() {
    return {
      orgEItems: [
        { label: "PTI31", value: "PTI31" },
        { label: "AS", value: "AS" },
        { label: "AM1", value: "AM1" },
        { label: "AM2", value: "AM2" },
        { label: "AT1", value: "AT1" },
        { label: "PL", value: "PL" },
```

```
      { label: "ÜL1", value: "ÜL1" },
      { label: "ÜL2", value: "ÜL2" }
    ],
    userItems: [
      { label: "Ost PTI 31 - Lean Experte", value: "LE" },
      { label: "Ost PTI 31 - Lean Officer", value: "LO" },
      { label: "Ost PTI 31 - Mitarbeiter", value: "EM" },
      { label: "Ost PTI 31 - Shopfloor Teilnehmer", value: "PT" }
    ],
    selectedReceiver: Array.from({ length: 12 }, (_, index) => ({
      value: index < 8 ? `OrgE${index}` : `Role${index - 8}`,
      selected: false
    })),
    selectedCc: Array.from({ length: 12 }, (_, index) => ({
      value: index < 8 ? `OrgE${index}` : `Role${index - 8}`,
      selected: false
    })),
    allUsersSelected: false,
    allOrgESelected: false
  };
},
methods: {
  handleSelectAllOrgE(event) {
    const selectAll = event.target.checked;
    this.allOrgESelected = selectAll;
    this.selectedReceiver.forEach((receiver, index) => {
      if (index < 8) receiver.selected = selectAll;
    });
    this.updateDisabledStates();
  },
  handleSelectAllUsers(event) {
    const selectAll = event.target.checked;
    this.allUsersSelected = selectAll;
    this.selectedReceiver.forEach((receiver, index) => {
      if (index >= 8) receiver.selected = selectAll;
    });
    this.updateDisabledStates();
  },
  isCcChecked(value) {
    return this.selectedReceiver.some(
      (receiver) => receiver.value === value && receiver.selected
    );
  },
  isAnChecked(value) {
    return this.selectedCc.some((cc) => cc.value === value && cc.selected);
  },
  updateDisabledStates() {
    // Disable checkboxes in CC based on the selection in AN
    this.selectedCc.forEach((cc, index) => {
      cc.disabled = this.isAnChecked(cc.value);
```

```
      });
      // Disable checkboxes in AN based on the selection in CC
      this.selectedReceiver.forEach((receiver, index) => {
        receiver.disabled = this.isCcChecked(receiver.value);
      });
    }
  },
  watch: {
    selectedReceiver: {
      handler() {
        this.updateDisabledStates();
      },
      deep: true
    },
    selectedCc: {
      handler() {
        this.updateDisabledStates();
      },
      deep: true
    }
  }
};
</script>

<style scoped>
.card-container {
  display: flex;
  gap: 10px;
}

.half-width {
  width: 50%;
  box-sizing: border-box;
  padding: 10px;
  padding-bottom: 30px;
}
.checkbox-group-container {
  margin-bottom: 20px;
}
.card-with-header {
  width: 50%;
  box-sizing: border-box;
  padding: 10px;
  padding-bottom: 30px;
}
</style>
```