Cross-field validators

- Similar to custom validators. Allows validation on entire form group.

```
export const crossFieldValidator: ValidatorFn = (
  control: AbstractControl
): ValidationErrors | null => {
  const name = control.get('name');
  const email = control.get('note');
  return email?.value.split('@')[0] === name?.value
    ? { crossFieldError: true }
    : null;
};
```

```
constructor(private expensesService: ExpensesService) {
  this.expenseForm = new FormGroup(
    {
      name: new FormControl('', {
        updateOn: 'blur',
        validators: [Validators.required, customValidator],
      }),
      date: new FormControl(''),
      amount: new FormControl(''),
      note: new FormControl(''),
    },
    { validators: crossFieldValidator }
  );
}
```

```
<div [hidden]="expenseForm.pristine || expenseForm.valid">
  <div *ngIf="expenseForm.errors?.['crossFieldError']">
    Expense Note and Expense Name should not be identical.
  </div>
</div>
```

Reactive Forms Task #4

- Sharing is caring. Expenses that have the same note as the name are now not allowed.
- Implement a cross field validator that searches for usages of the expense name in the note and makes the field invalid.
- Add an error message at the bottom of the form “Please provide correct data”.