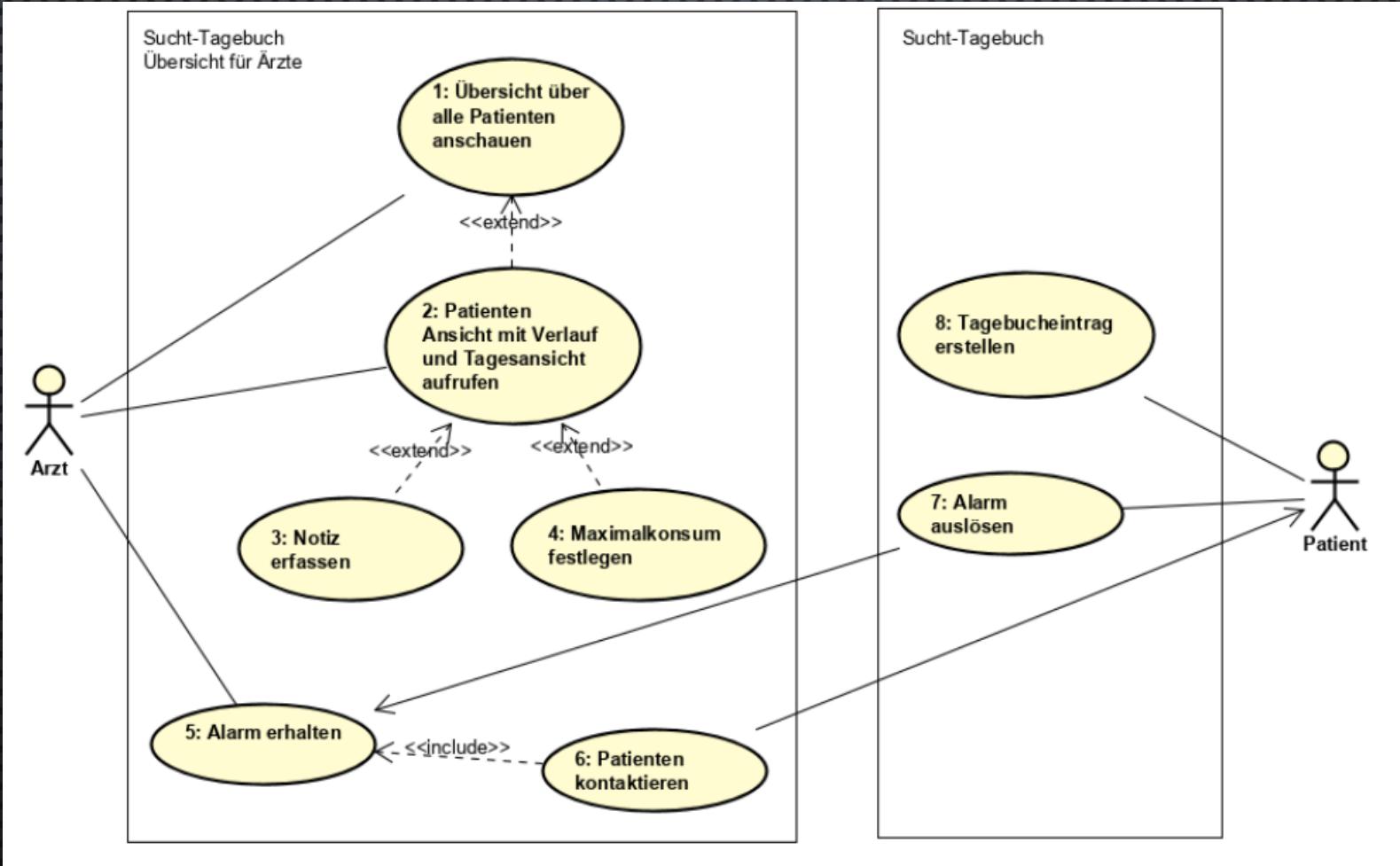


FINAL PRESENTATION

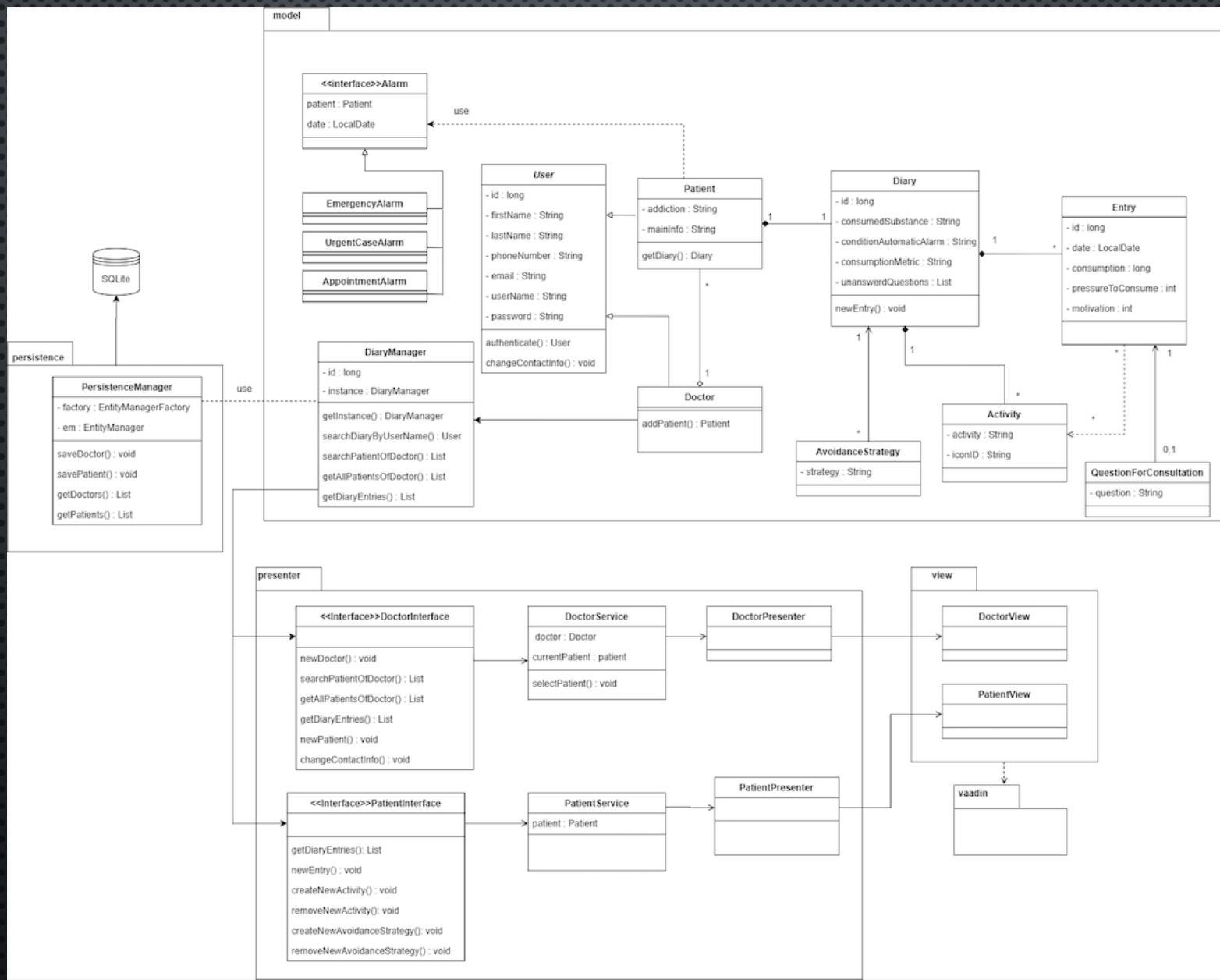
TEAM RED

DÉNERVAUD NATALYA, RODRIGUEZ JULIÁN, MEYER REMO, PELTS DMYTRIY,
KIRUPANANTHAN KAURISANKER

FEATURES IMPLEMENTED VS. DESIGN THINKING IDEAS



ARCHITECTURE



CODE HIGHLIGHT

```
public class Condition {  
    private Metric metric;  
    private Operand operand;  
    private int thresholdValue;  
    private int thresholdDays;  
    private int amountOfDays;  
  
    NextCondition nextCondition = NextCondition.NONE;  
  
    Condition condition;  
  
    public Condition(Metric metric, Operand operand, int thresholdValue, int thresholdDays, int amountOfDays) {  
        this.metric = metric;  
        this.operand = operand;  
        this.thresholdValue = thresholdValue;  
        this.thresholdDays = thresholdDays;  
        this.amountOfDays = amountOfDays;  
    }  
  
    protected Condition decorateAnd(Condition condition) {  
        this.nextCondition = NextCondition.AND;  
        this.condition = condition;  
        return this.condition;  
    }  
  
    protected Condition decorateOr(Condition condition) {  
        this.nextCondition = NextCondition.OR;  
        this.condition = condition;  
        return this.condition;  
    }  
  
    protected boolean isGiven(List<Entry> entries) {  
        Collections.sort(entries);  
        switch (this.isGiven(ConditionState.UNKNOWN, entries)) {  
            case FALSE:  
                return false;  
            case TRUE:  
                return true;  
            case UNKNOWN:  
                return false;  
            default:  
                return false;  
        }  
    }  
}
```

```
protected ConditionState isGiven(ConditionState state, List<Entry> entries) {  
    switch (state) {  
        case FALSE:  
            switch (this.nextCondition) {  
                case AND:  
                    return this.condition.isGiven(ConditionState.FALSE, entries);  
                case NONE:  
                    return ConditionState.FALSE;  
                case OR:  
                    return this.condition.isGiven(ConditionState.UNKNOWN, entries);  
            }  
        case TRUE:  
            return ConditionState.TRUE;  
        case UNKNOWN:  
            switch (this.nextCondition) {  
                case AND:  
                    if (this.check(entries)) {  
                        return this.condition.isGiven(ConditionState.UNKNOWN, entries);  
                    } else {  
                        return this.condition.isGiven(ConditionState.FALSE, entries);  
                    }  
                case NONE:  
                    if (this.check(entries)) {  
                        return ConditionState.TRUE;  
                    } else {  
                        return ConditionState.FALSE;  
                    }  
                case OR:  
                    if (this.check(entries)) {  
                        return this.condition.isGiven(ConditionState.TRUE, entries);  
                    } else {  
                        return this.condition.isGiven(ConditionState.UNKNOWN, entries);  
                    }  
            }  
    }  
    return state;  
}
```

CODE HIGHLIGHT

```
private boolean check(List<Entry> entries) {
    int res = 0;
    switch (this.operand) {
        case ISEQUAL:
            for (Integer i : getValues(entries)) {
                if (i == this.thresholdValue) {
                    res++;
                }
            }
            break;
        case ISEQUALORLOWER:
            for (Integer i : getValues(entries)) {
                if (i <= this.thresholdValue) {
                    res++;
                }
            }
            break;
        case ISHIGHER:
            for (Integer i : getValues(entries)) {
                if (i > this.thresholdValue) {
                    res++;
                }
            }
            break;
        case ISHIGHEROREQUAL:
            for (Integer i : getValues(entries)) {
                if (i >= this.thresholdValue) {
                    res++;
                }
            }
            break;
        case ISLOWER:
            for (Integer i : getValues(entries)) {
                if (i < this.thresholdValue) {
                    res++;
                }
            }
            break;
        default:
            break;
    }
    if (res >= this.thresholdDays) {
        return true;
    } else {
        return false;
    }
}
```

```
private List<Integer> getValues(List<Entry> entries) {
    List<Integer> res = new ArrayList<Integer>();
    switch (this.metric) {
        case CONSUMPTION:
            for (int i = 0; i < this.amountOfDays; i++) {
                res.add(Math.toIntExact(entries.get(i).getConsumption()));
            }
            break;
        case MOTIVATION:
            for (int i = 0; i < this.amountOfDays; i++) {
                res.add(entries.get(i).getMotivation());
            }
            break;
        case PRESSURE:
            for (int i = 0; i < this.amountOfDays; i++) {
                res.add(entries.get(i).getPressureToConsume());
            }
            break;
        default:
            return null;
    }
    return res;
}
```

LESSONS LEARNT

PRO:

- VAADIN ALS FRONTENDTECHNOLOGIE KENNENGELERNT
- SCRUM METHODE KENNENGELERNT
- DETAILLIERTE DISKUSSION DER ARCHITEKTUR
- NACH MVP PATTERN IMPLEMENTIERT
- ARBEITSERFAHRUNG ALS GRUPPE
- CODEREVIEWS

CON:

- VAADIN DOCUMENTATION
- DB-ANBINDUNG

DEMO

Edit patient data All entries Activities Open questions Strategies Patient overview Log out

First name: Natalya Edit Info Show all entries

Last name: Dénervaud Activities Open questions

Addiction: Alcohol Strategies

Info: Had control over her addiction for 3 years. Relapse on january 2020 Back to patient list

Edit automatic alarm condition Log out

Statistic of patient's addiction

The chart displays three data series: Consumption (blue bars), Motivation (green line with dots), and Max Level Consumption (black line with dots). The X-axis represents days from 1 to 30. The left Y-axis for Consumption ranges from 0 mm to 7.5 mm. The right Y-axis for Motivation ranges from 0 to 7.5. The Max Level Consumption line peaks at day 2 (approx. 4.7) and then generally declines towards zero by day 30. Consumption shows several peaks, notably at days 4, 5, and 6. Motivation starts high (approx. 6.5) and drops sharply after day 8, remaining near zero thereafter.

Day	Consumption (mm)	Motivation	Max Level Consumption
1	0.0	6.5	3.0
2	5.8	6.0	4.7
3	4.7	4.7	3.8
4	5.8	4.7	2.8
5	4.7	3.0	2.2
6	3.8	0.8	1.2
7	1.6	0.8	1.2
8	1.6	0.8	1.2
9	0.0	0.0	0.0
10	0.0	0.0	0.0
11	0.0	0.0	0.0
12	0.0	0.0	0.0
13	0.0	0.0	0.0
14	0.0	0.0	0.0
15	0.0	0.0	0.0
16	0.0	0.0	0.0
17	0.0	0.0	0.0
18	0.0	0.0	0.0
19	0.0	0.0	0.0
20	0.0	0.0	0.0
21	0.0	0.0	0.0
22	0.0	0.0	0.0
23	0.0	0.0	0.0
24	0.0	0.0	0.0
25	0.0	0.0	0.0
26	0.0	0.0	0.0
27	0.0	0.0	0.0
28	0.0	0.0	0.0
29	0.0	0.0	0.0
30	0.0	0.0	0.0