

Projekt Tempera – Zwischenpräsentation

Jonas Bayer

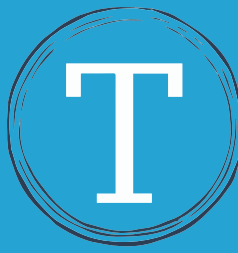
Lea Hof

Manuela Niedrist

Astrid Reisinger

Niklas Speckle

Time Tracking / Office Climate



[Schedule](#) [List](#) [Your Cumulative Work Hours](#)



Heute

15 – 21. Apr. 2024

Monat

W

	Mo. 15.4.	Di. 16.4.	Mi. 17.4.	Do. 18.4.	Fr. 19.4.	Sa. 20.4.	So. 21.4.
08:00		8:00 - 11:00 AVAILABLE					
09:00	9:00 - 12:00 AVAILABLE (Project 1)						
10:00							
11:00							
12:00		12:00 - 15:00 MEETING					
13:00	13:00 - 17:00 DEEP_WORK (Project 2)						
14:00							
15:00							
16:00							
17:00							

Event Time Record



Project:

no project assigned



Description:

Work Mode:

MEETING



Start Time:

12:00

End Time:

15:00

Abort

Save

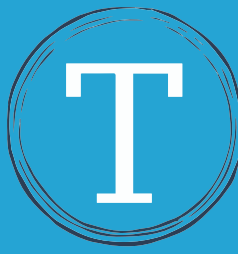
Split Time Record at:

12:00

Abort

Split

Time Tracking / Office Climate



Temperature

24.0 C°

Humidity

40.0 %

Air Quality

30.0 ppm

Light

690.0 lux

Temperature

Humidity

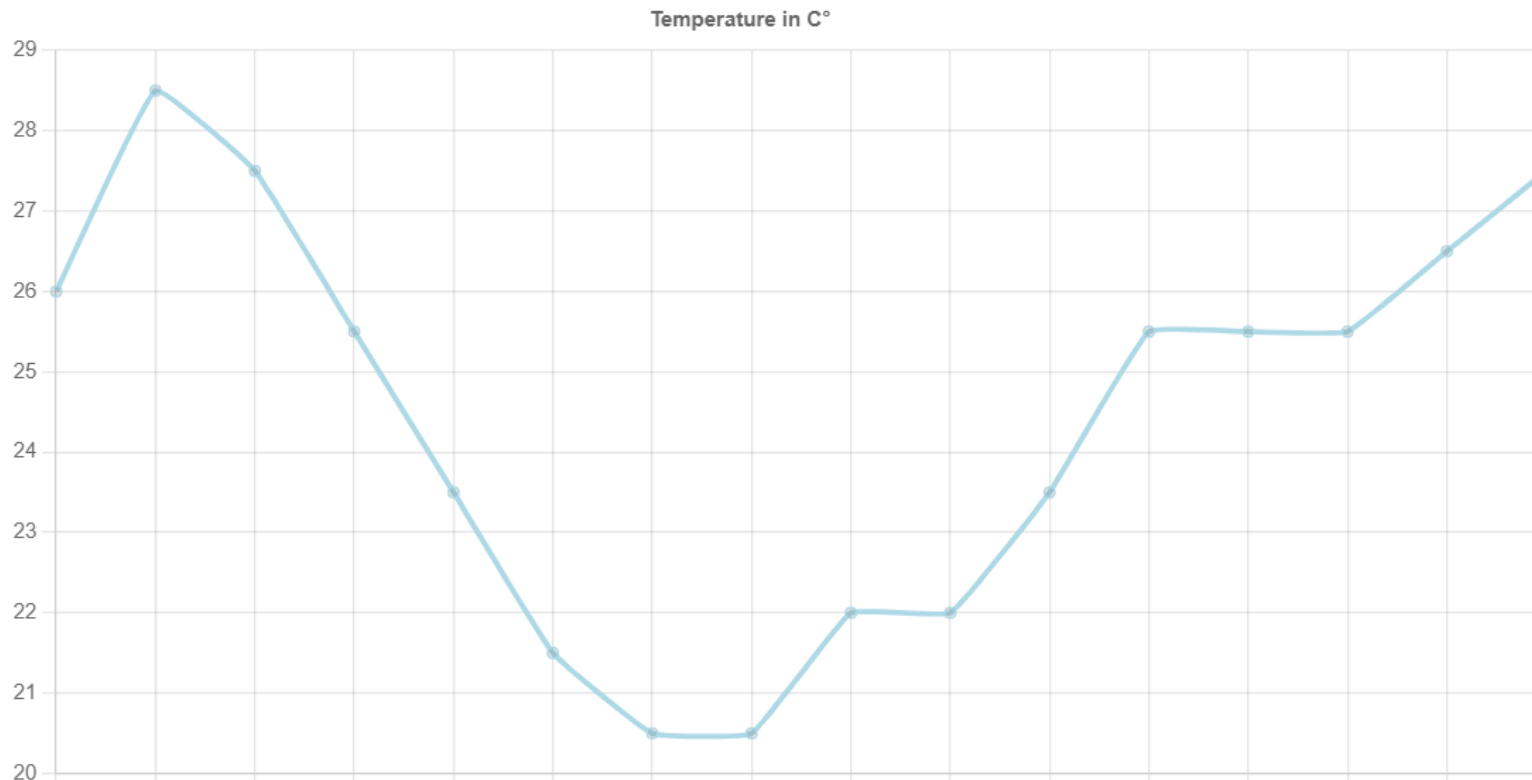
Air Quality

Light

Too warm: Check for open windows and doors and turn on ac

Too cold: Check for open windows and doors and turn up the heating.

Overview of past climate data:



Settings for Chart

datatype:

Air Temperature

Air Humidity

Air Quality

Light Intensity

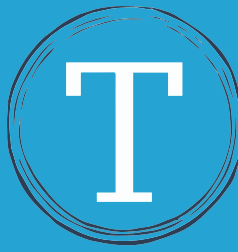
Range: 03.05.24 - 04.05.24

granularity in minutes:

2

✓ Submit

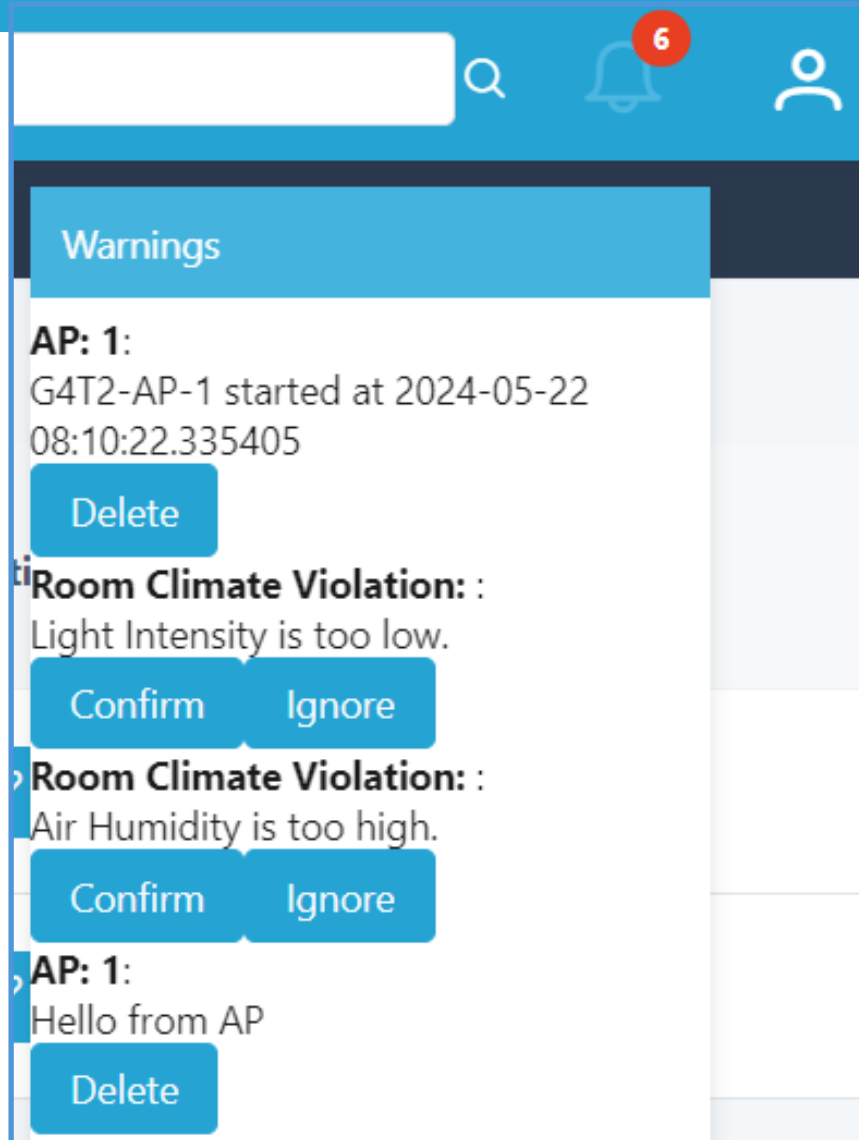
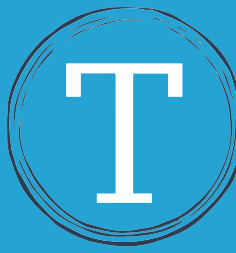
Feature: Notifications



- Durch Tempera Device bei fehlerhafter Verbindung
- Durch Access Point mittels POST-Request

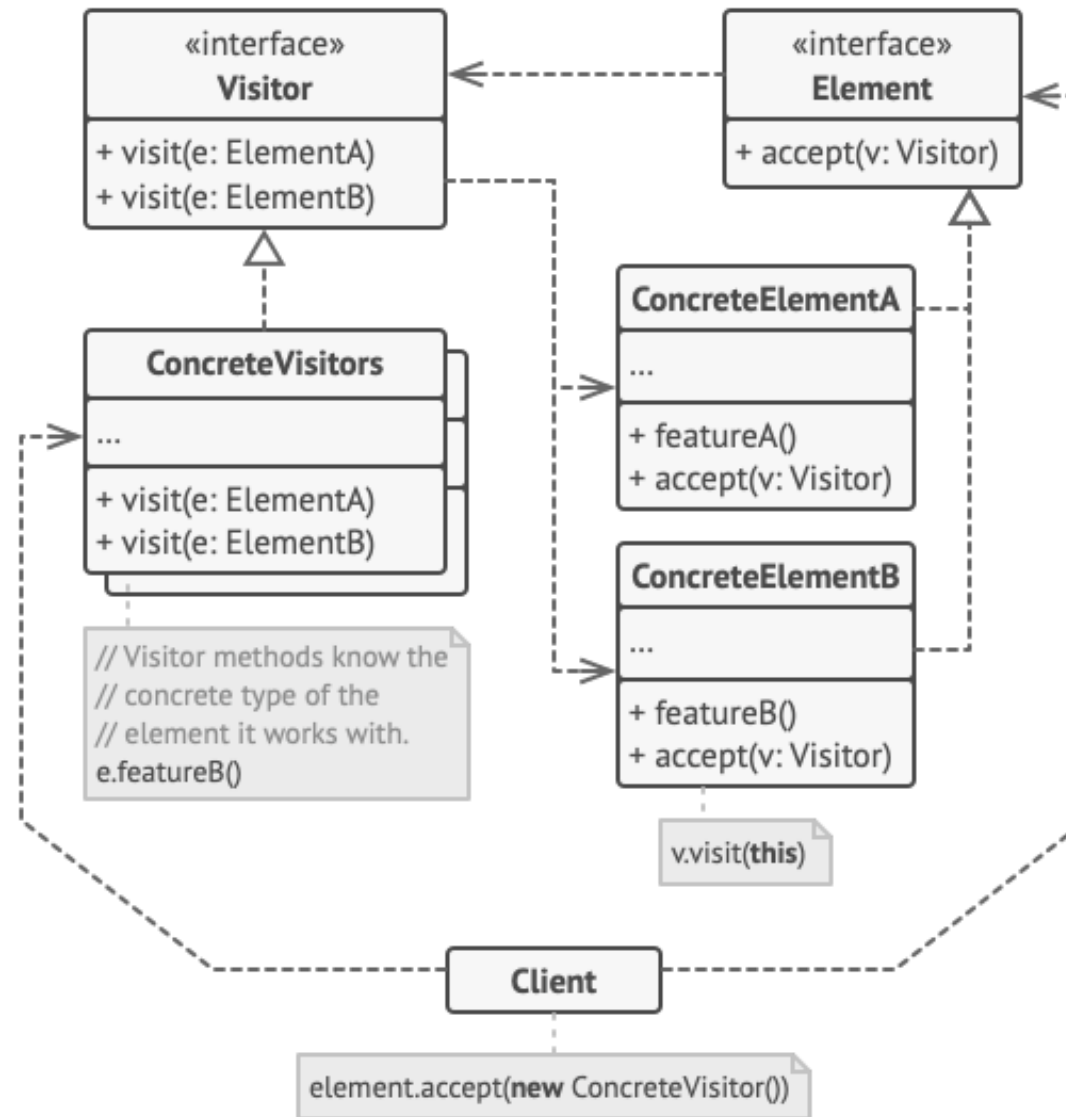
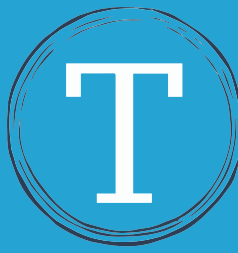


Feature: Notifications



	«interface» Notification
<ul style="list-style-type: none">- user: Userx- header: String- message: String- buttons: List<NotificaitonButton>	

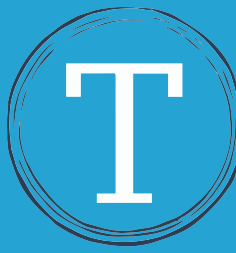
Visitor Pattern



NotificationButton

- NotificationDeleteB
- NotificationConfirmB
- NotificationIgnoreB

Observer Pattern – Token



@Transactional 2 usages csaf7520 *

```
public void updateWarningStatus(Warning warning,
                                WarningStatus warningStatus,
                                TemperaDevice temperaDevice) {
    warning.setWarningStatus(warningStatus);
    warningRepository.save(warning);

    tokenService.checkToken(warning);

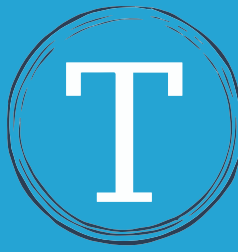
    applicationEventPublisher.publishEvent(new NotificationEvent
        (source: this, temperaDevice, warning));
}
```

Listener-Klassen:

- ListNotification
- EmailNotification

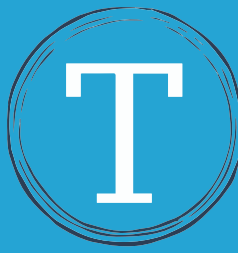
onApplicationEvent
(NotificationEvent event)

Feature: Logging als Aspect



- Aspect Oriented Programming
- Mittles Proxy
- Power Type Pattern: Entitätsklasse \leftrightarrow Metaklasse

Feature: Logging als Aspect



```
@AfterReturning(pointcut="execution(* at.qe.skeleton.services.UserService.deleteUser(..))" +
    "&& args(user)", argNames= "user")
public void logDeleteUser(Userx user) {
    AuditLog log = new AuditLog();
    log.setAction(Action.DELETE);
    log.setStatus(ActionStatus.SUCCESS);
    log.setAccessedResource("USER: "+user.getUsername()+" (id: "+user.getId()+")");
    log.setAuthenticatedUser(getAuthenticatedUser());
    log.setTimestamp(LocalDateTime.now());
    logRepository.save(log);
}
```

Auszug Aspect-Klasse

```
@Entity
@EntityListeners(AuditingEntityListener.class)
public class Userx extends Metadata
```

Power Type Pattern

Danke für eure Aufmerksamkeit!

