HR Genie
Backend Algorithm
Compute the duration of the Absences

Daniele Scopece
started 2024-07-17

EXECUTIVE SUMMARY

- The calculation of the duration is made in the POST absence request
- At the moment only the period 2024

DATA NEEDED to make it work:

- For every user add variables in the timeDependentVar tables
  - Put dateStart = 2024-01-01 and dateEnd = 2024-12-31
  - nr_tot_vacation_days
  - pensum_perc
  - nr_working_hours_per_day_at100PercPensum -> it defines how many hours are in a “day” of holiday
  - day_<Weekday>_startTime, day_<Weekday>_endTime in format HH:MM
    - <Weekday> = Monday, Tuesday, Wednesday, Thursday, Friday, Saturday, Sunday
    - It specifies the working hours in the specific week day
    - No lunch break is included in this version
    - For every “start” there must be an “end”
    - There cannot be two or more “start” or “end” for the same user

- Checks added in the POST method with the relative messages to the front end
  - If the nr_working_hours_per_day_at100PercPensum is missing
  - If no day is added
  - If there are multiples “start” or “end” for the same day

Context

- We need to compute the duration of the absences in working hours in the backend
- This will allow to fetch the duration and to do statistics
- This will be computed in the serailizer

-

Keep In Mind

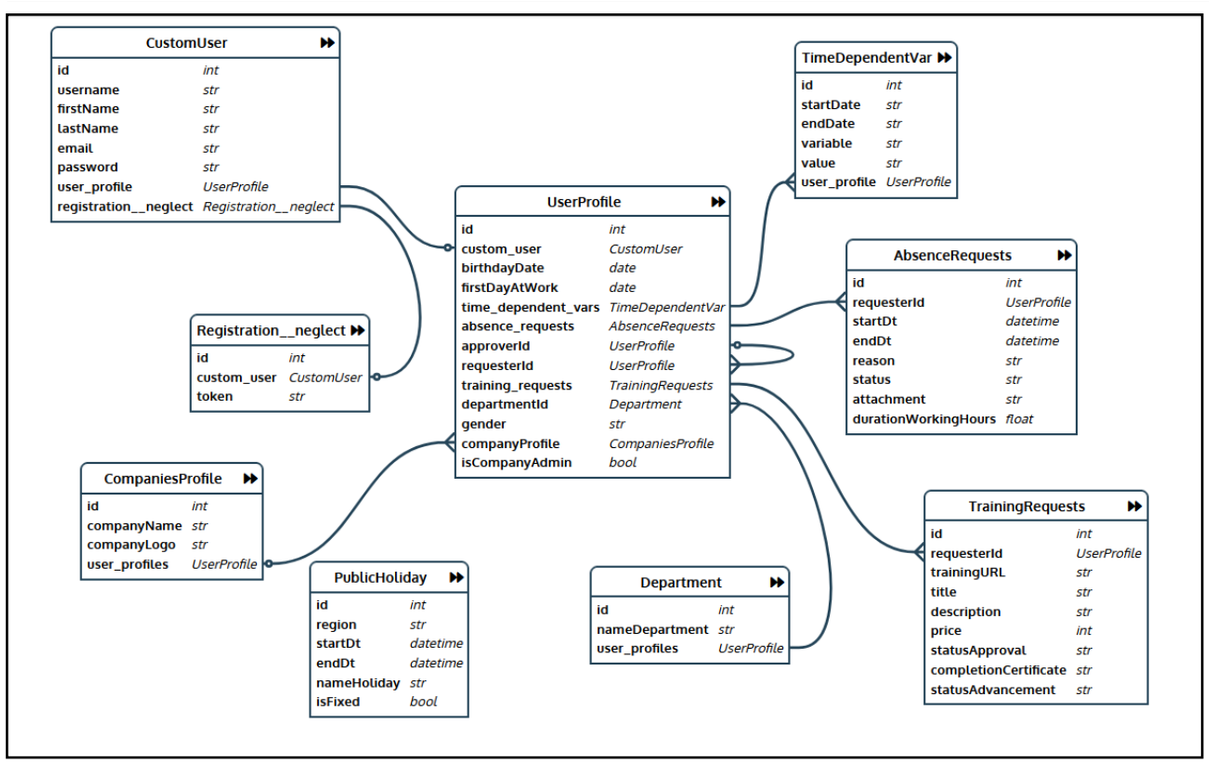
- We need to consider the actual absences in the WORKING time, also considering that the workers might work in different shifts, part time
- The amount of days is always computed by considering the hours in a FULL working day at 100% pensum
  - Example: a 50% worker will have 24 days * 0.5 = 12 full days
- We compute the HOURS of absences as float number
  - And then convert them into days, hours, minutes
- The user can put datetime as start and end of the absence, which can go beyond their work hours.
  Hence we need to compute the hours in the backend, ideally when POSTING an absence

Data

- Time dependent variables
  - The best is to have the data as JSON with all the variables -> easier to fetch
    - StartValidity
    - End Validity
    - UserProfile
    - Entry = {
      “pensum_percentage”: “100”,
      “nr_vacation_days”: “24”,
      “nr_working_hours_per_day_at100PercPensum”: “8”, → the same company wide, but easier to put here

      # this below allow flexible shifts → needed to compute the exact time off
      “working_time”: {
      “Monday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Tuesday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Wednesday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Thursday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Friday”: {“startTime”: “08:00”, “endTime”: “17:00”}
      }
      }

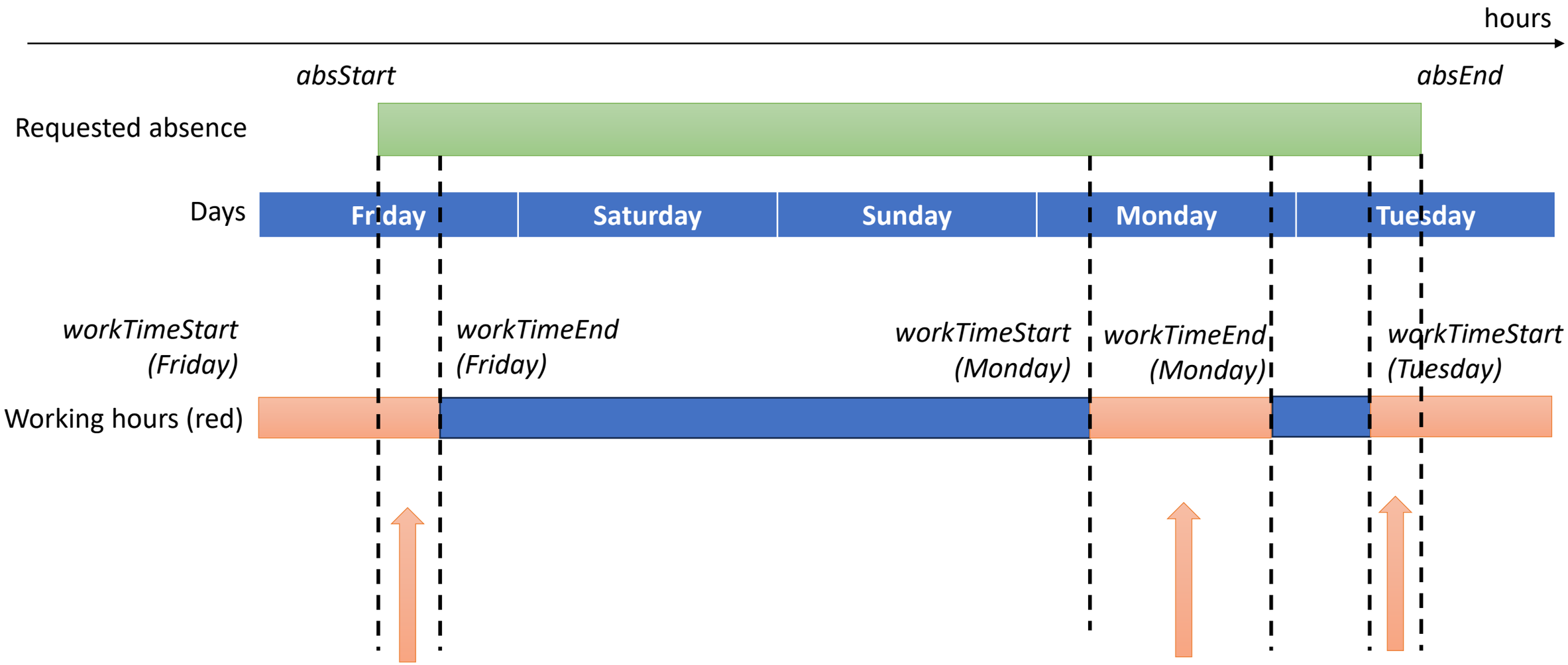
- Time dependent variables Simple
- Pensum_percentage =



Notes on Data

- In order for the algorithm to work we need
  - Time Dependent variables are done with calendar year of validity -> easier

Visualization & Algorithm

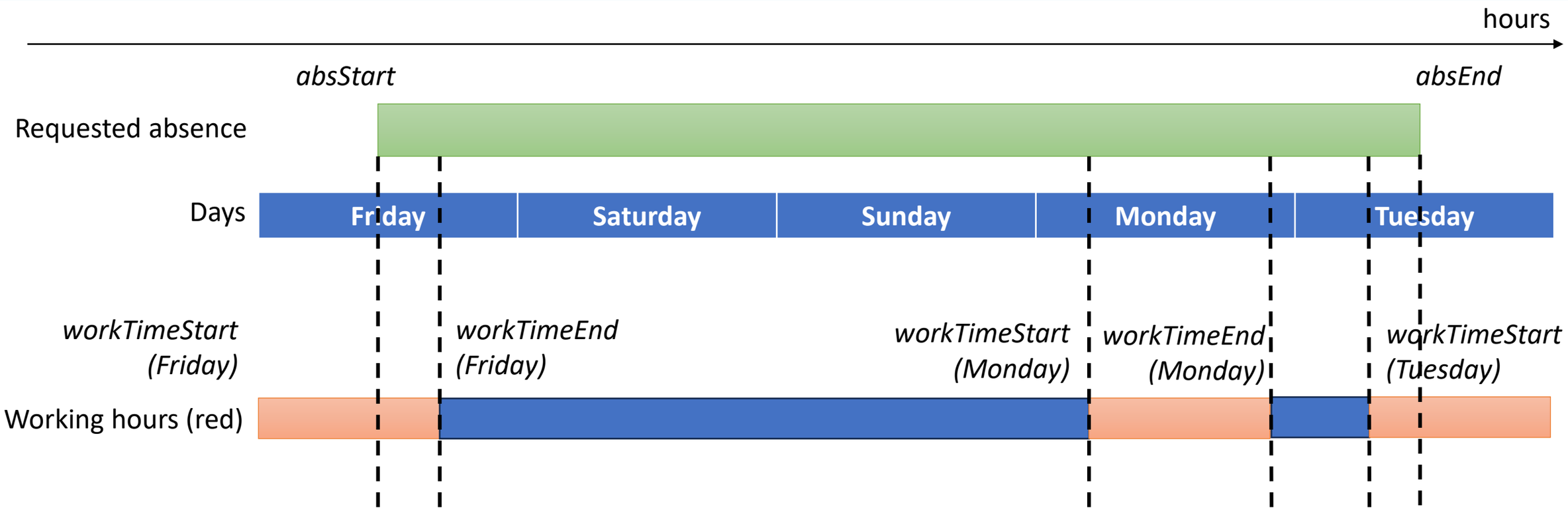


The duration to be considered for the absence is the overlap of the green bar to the red bars (red upward arrows)

ORIGIN of data

- absStart and absEnd ← from absenceRequest posted
- workTimeStart(day) and workTimeEnd(day) ← from timeDependent Variables

Algorithm & Pseudocode



- Do this calc in the serializer
- Add a check: duration > 0 → if zero → say “wrong interval or no data for working_time in the time dependent variables”
- Fetch the correct time dependent variables for the user → validate with interval inside the validity

- Initialize duration_hours = 0
- Loop on days between date(absStart) and date(absEnd)
  - compute the week day
  - if no worktimeStart defined for that day -> go to next day
  - If here, then the workTimeStart and workTimeEnd is defined
  - If absStart >= dt(workTimeStart(day)) ➔ startNow = absStart, else startNow = dt(workTimeStart(day)
  - If absEnd <= dt(workTimeEnd(day)) ➔ endNow = absEnd, else endNow = dt(workTimeEnd(day))
  - Duration_hours += (endNow-startNow) converted to hours
- Here we have the total duration of the absence in hours -> float → durationWorkHours
- Convert into days of FULL PENSUM via “nr_working_hours_per_day_at100PercPensum”
  - → 2d_4h_30m = durationWorkTimeFormatted

Procedure to Implement

1. ~~Create my branch~~
2. Modify TimeDependentVariable → put as JSON objects → complicated I do that later
3. Add the time dep var for me admin_Daniele → test with this
4. ~~Change the model of absence Request ->add durationWorkHours & durationWorkTimeFormatted~~
5. Change the serializer of absenceRequest with the data validation
6. Test via Postman also the notice given if problem
7. Push to my branch
8. Merge into master locally -> push again
9. Merge into master on GitLab
10. Pull master and test again
11. Tell Luka to push to Heroku and test from there

Doing 2024-07-17

- IDEA simple:
  - I use hard-coded values for the serializer first so to test if the algorithm works
  - Then I take from TimeDependent variables

models.py

```
class AbsenceRequest(models.Model):
    reason = models.CharField(max_length=11, choices=ABSENCE_REASONS, default='vacation')


    STATUS_APPROVAL = [
        ('pending', 'Pending'),
        ('approved', 'Approved'),
        ('rejected', 'Rejected'),
    ]
    status = models.CharField(max_length=11, choices=STATUS_APPROVAL, default='pending')

    # duration computed by the serializer
    durationWorkHours = models.FloatField()  # duration in hours as float: only the duration in the hours
    durationWorkTimeFormatted = models.CharField()  # duration in format 2d_3h_15m --> easy to display in


    # TODO: relate to the MediaLink table
```

Doing 2024-07-17

- Serializers we have 2 now
  - AbsenceSerializersAll
    - CreateListModifyAbsenceMeView = get and post of me ➔ 2 different serializers for post and get
    - CreateListModifyAbsenceEmployeeMyTeamView = only get
    - CreateListModifyAbsenceEmployeeMyManagerView = only get
    - ListAbsenceManagerMyTeamView = only get
  - AbsenceSerializerManager
    - ModifyAbsenceManagerMyTeamView -> only patch approval status -> this unchanged!

- I need to
  - modify the serializer for get -> extract also the duration x 2
  - Create a new serializer only for POST -> compute the duration
    - Then it will be used also for the PATCH of the user to change the endtime but not in the first version

Doing 2024-07-17

- Tested the logic in python file in Documents
  - Algorithm-Duration-Test.v01.01.py
- It works
- Inserted in the code in the POST method with hard-coded variables

Doing 2024-07-17: 18:45 onwards

- Code works with hard-coded variables
- Now fetch the variables
- Modify TimeDepVar -> define as Char -> variables
- Elaborate the char in the POST

- Time dependent variables
  - The best is to have the data as JSON with all the variables -> easier to fetch
    - StartValidity
    - End Validity
    - UserProfile
    - Entry = {
      “pensum_percentage”: “100”,
      “nr_vacation_days”: “24”,
      “nr_working_hours_per_day_at100PercPensum”: “8”, → the same company wide, but easier to put here

      # this below allow flexible shifts → needed to compute the exact time off
      “working_time”: {
      “Monday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Tuesday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Wednesday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Thursday”: {“startTime”: “08:00”, “endTime”: “17:00”},
      “Friday”: {“startTime”: “08:00”, “endTime”: “17:00”}
      }
      }

Time dependent variables Simple

- Pensum_perc =
- Nr_tot_vacation_days
- nr_working_hours_per_day_at100PercPensum
- day_Monday_startTime
- day_Monday_endTime
- …
- day_Sunday_startTime
- day_Sunday_endTime

Doing 2024-07-17: 20:30 IT WORKS

- FINISHED and TESTED