Week 01: Introduction to Seminars

Welcome to the course!

Agenda

- Tutor introduction
- Course info
- Git Basics
- Gitflow
- Setup Gitlab & IDE
- Hands on: Data modeling

Who (am/are) (I/we)?

Let me introduce myself

Course info: Basics

- Seminars are compulsory (max. 4 unexcused seminars)
- Attendance is checked using ROPOTs at the start of each seminar
- Special Lukáš Grolig demo seminars
- Source of truth: Syllabus and Gitlab
- Communication: Discord
  - Help support almost 24/7
  - We ❤️ to help you

Deadlines

For iterations: 72h hours before seminar (soft deadline - each tutor can have their own policy)

For team projects: Date of your presentation (more information later during the semester)

Course info: Evaluation

- Up to **33 points from iterations** (for completing assignments of the semestral project with the best effort and clean code).
- Up to **42 points for your team project** (for creating a complex solution, dividing work, and collaborating with others).
- Up to **25 points for exams** (the final ROPOT contains all the topics from the semester)
- Up to **5.2 soft points** for seminar ROPOTs - these apply after you gain at least 70 points from iterations, team project, and exams to help you get a better grade.

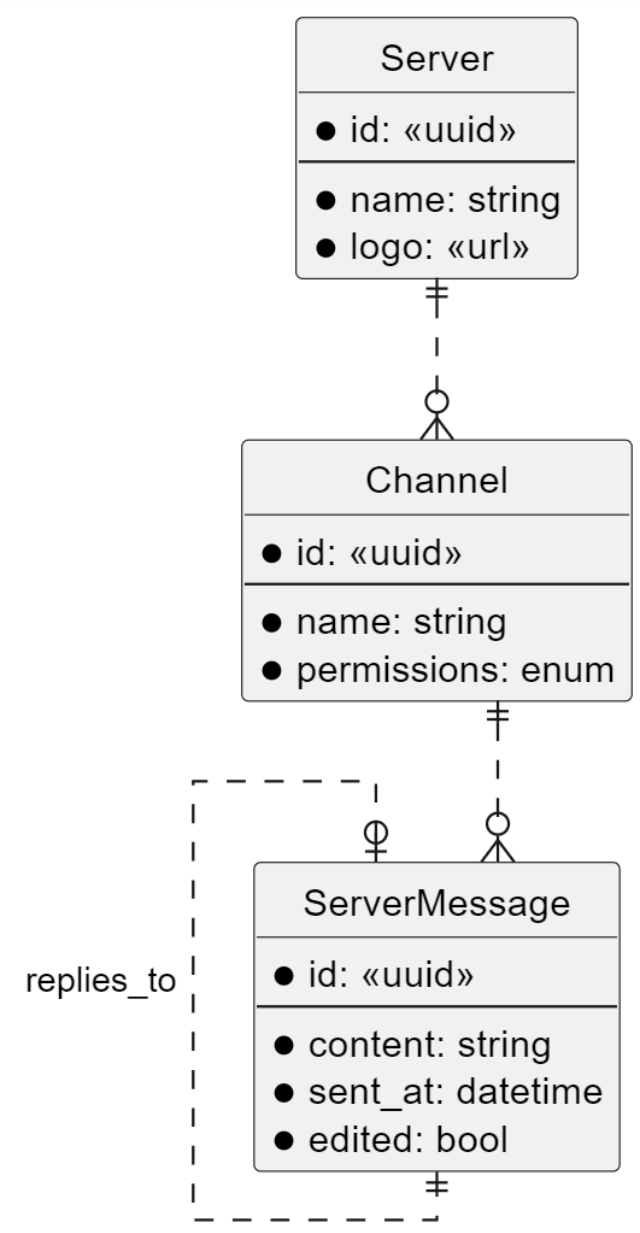
Grade	Points
A	100-94
B	93-88
C	87-82
D	81-76
E	75-70
F	69-0
Z	100-60
N	59-0

Let's revise data modeling in ERD

- Entity, Primary key, Foreign key, Relationship
- Types of ERD: Conceptual, Logical, Physical models

PlantUML

```
@startuml lab01-diagram

hide circle

skinparam Linetype ortho

entity ServerMessage {
    * id: <<uuid>>
    ---
    * content: string
    * sent_at: datetime
    * edited: bool
}

entity Channel {
    * id: <<uuid>>
    ---
    * name: string
    * permissions: enum
}

entity Server {
    * id: <<uuid>>
    ---
    * name: string
    * logo: <<url>>
}

ServerMessage |o..|| ServerMessage: replies_to
Server ||..o{ Channel
Channel ||..o{ ServerMessage
@enduml
```



Git: Setup

1. Install git (Depends on your OS)

   Using a package manager:

   ```
   apt install git      # ubuntu/debian
   brew install git     # mac
   winget install Git   # windows package manager
   ```

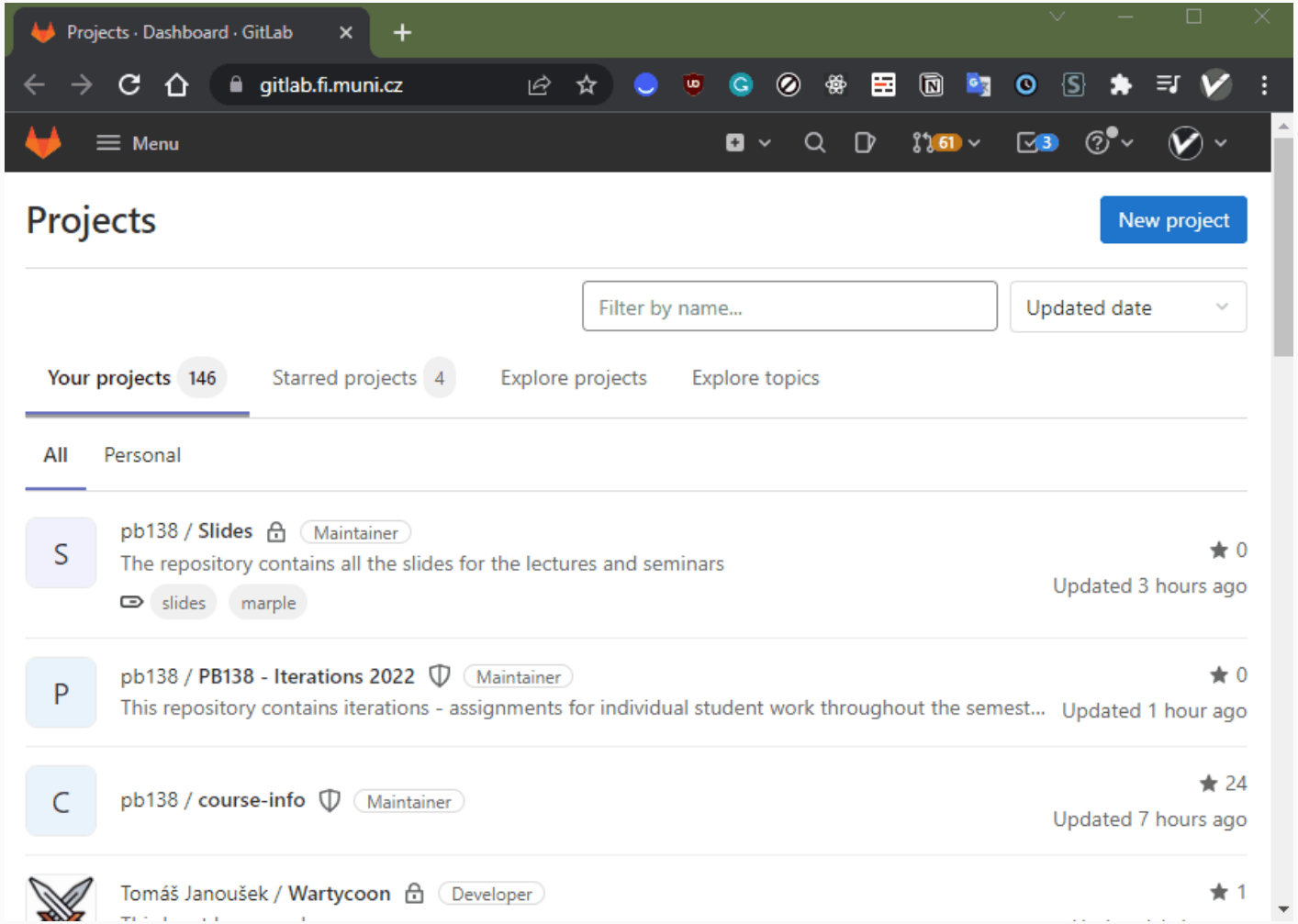
   Or you can use the Git website.

2. Setup keys

   ```
   ssh-keygen -o -a 100 -t ed25519 -f ~/.ssh/id_muni -C "xuser@fi.muni.cz"
   ```

Git: Setup

3. Add public key to Gitlab



Git: Basics

Commit in Conventional commits

```
git config --global core.excludesFile "**/node_modules"
git config --global user.name xuser
git config --global user.email xuser@fi.muni.cz
```

```
git clone <url>                    # Clones the repository
git status                         # Show status of added, removed files
git checkout -b submit-00          # Switch to new branch
git commit -am "feat: added erd" # Commit all tracked files
git push origin submit-00          # Push commited to remote branch
```

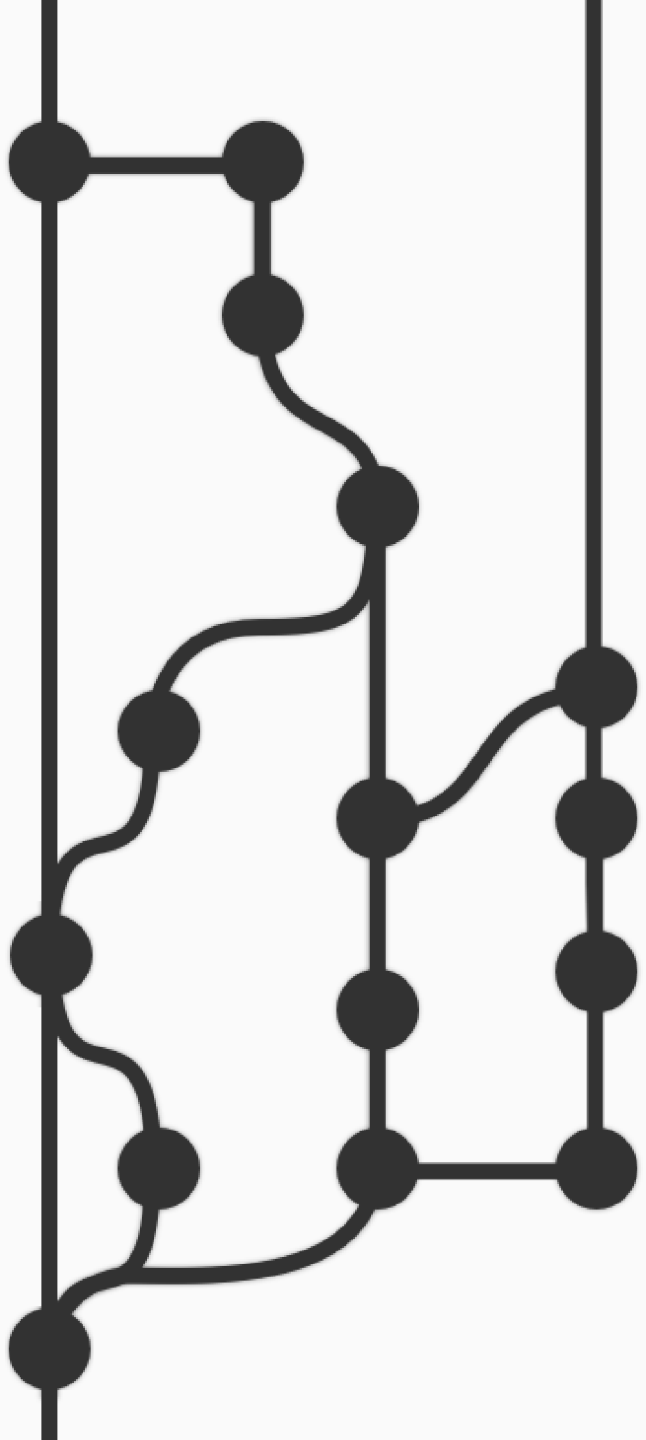
Cheatsheet

Git: Gitflow

- Starts from the main branch
- Feature branches contain new features, additions
- Master/Main is stable (tagged)

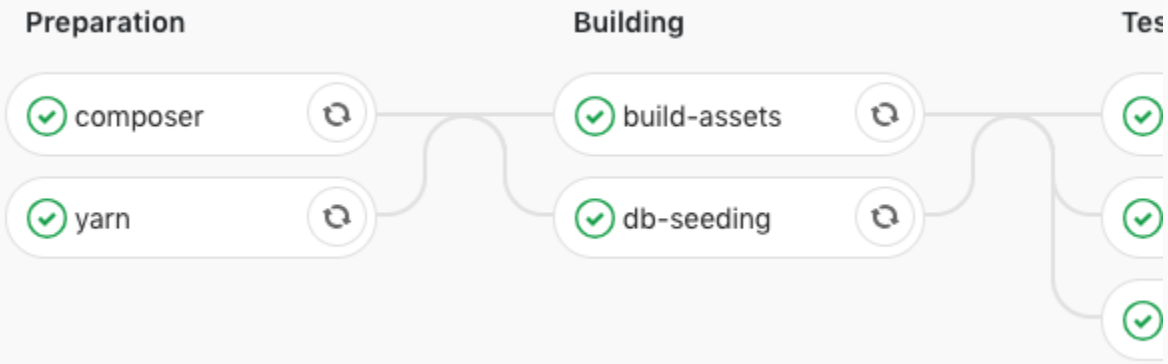
Note: You'll most likely work in trunk based development (it ignores the develop branch)

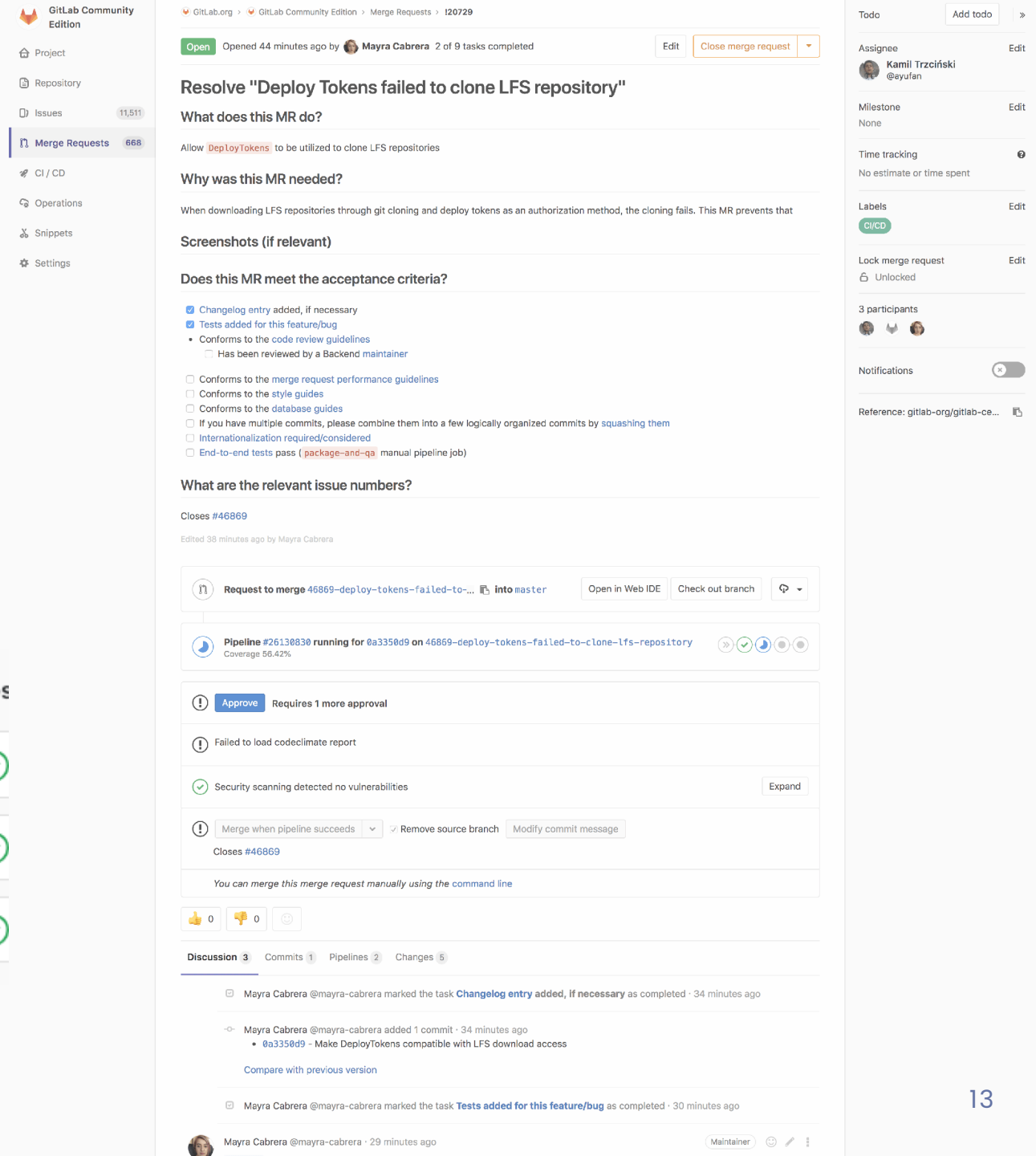
In-depth explanation



Git: Merge requests / Pull request

- MR/PR the way you check source code changes into a main branch
- Before pushing code, **check** common mistakes
- CI is not your enemy
- **We require you to discuss changes**





Workspace setup

IDE: **Webstorm, VSCode, vim**
Git: **Gitkraken, Github desktop***
Extensions: PlantUML, GraphViz

Extensions will be announced in every seminar session (We use many in this course)
**Note: ignore if you like solving conficts on your own*

Activity

1. Make group of 3-4 students
2. Model: Heureka domain on paper
    - Product, Category, Store, Product Price, Product Photo
3. **Logical model**: focus on attributes, relations
4. Compare within groups in seminar
5. Code PlantUML (save it locally)
6. Git-Push to Gitlab (next slide)

Publishing the Heureka diagram as Submit 00

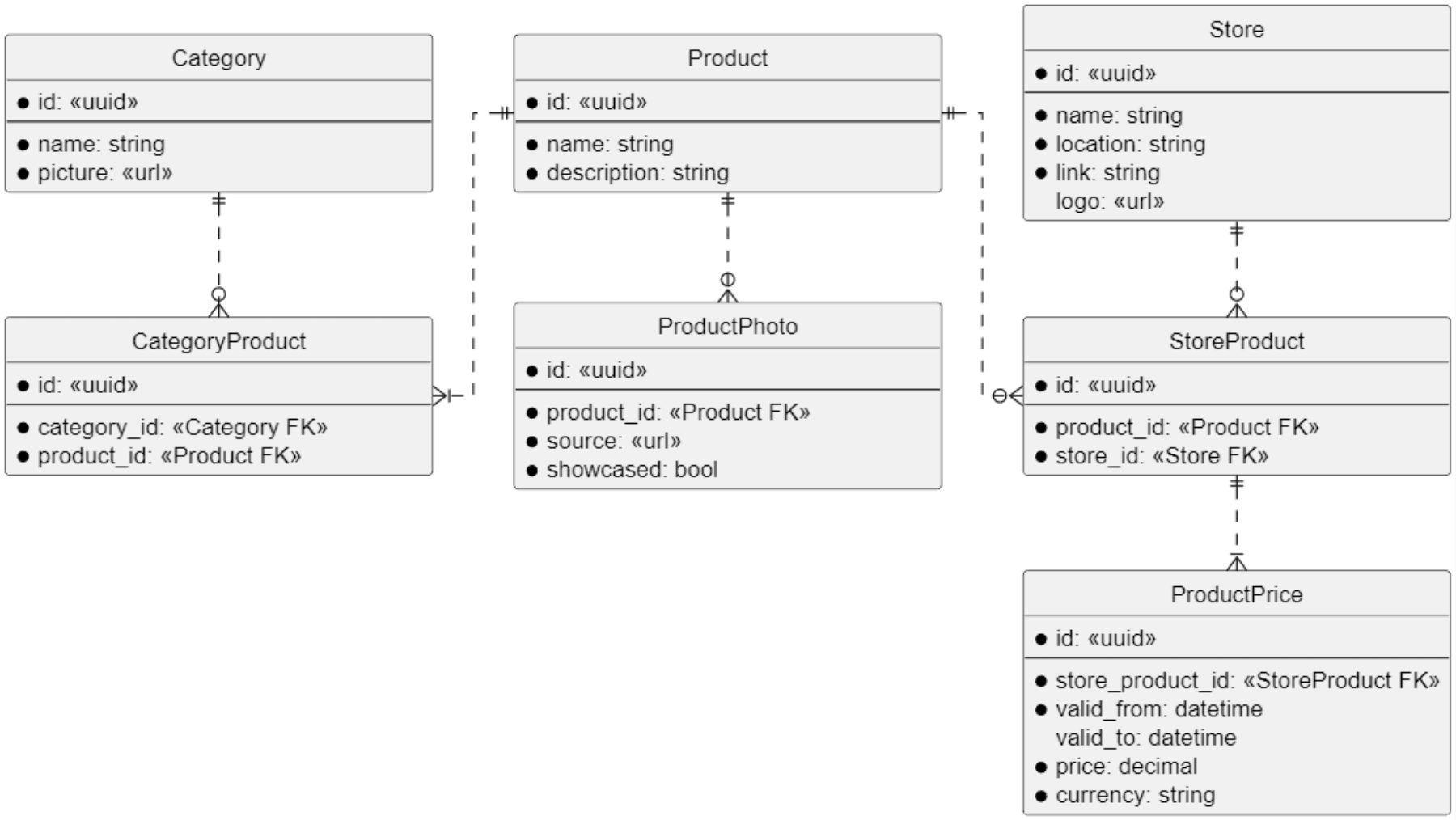
1. Fork the repository pb138/pb138-iterations-2023
2. Clone the repository
3. Create the branch **submit-00**
4. Save **heureka.puml** to branch **submit-00**
5. Invite your seminar tutor(s) to the project as developer(s)
6. Open merge request from **submit-00** to **main**
7. Assign tutor as Assignee and Reviewer on Merge request

Publishing the Heureka diagram as Submit 00

And now wait... For the code review.

Demo

Demo


Category
id: «uuid»
name: string
picture: «url»
Product
id: «uuid»
name: string
description: string
Store
id: «uuid»
name: string
location: string
link: string
logo: «url»
CategoryProduct
id: «uuid»
category_id: «Category FK»
product_id: «Product FK»
ProductPhoto
id: «uuid»
product_id: «Product FK»
source: «url»
showcased: bool
StoreProduct
id: «uuid»
product_id: «Product FK»
store_id: «Store FK»
ProductPrice
id: «uuid»
store_product_id: «StoreProduct FK»
valid_from: datetime
valid_to: datetime
price: decimal
currency: string

In case of fire 

1. git commit

2. git push

3. leave building