

# Happy Go Lucky Introduction

---

Dirk Riehle, FAU Erlangen

**ADAP C01**

Licensed under [CC BY 4.0 International](https://creativecommons.org/licenses/by/4.0/)

# Agenda

---

1. Happy Go Lucky
2. Current state
3. Goal state
4. Homework

# **1. Happy Go Lucky**

---

# Happy Go Lucky Vision

---

Happy Go Lucky (HGL) is

- A web app to support our project based teaching

Original HGL features are

- Happiness index
- Standup emails
- Code tracking

# Happy Go Lucky Base Design

---

HGL is a web app to support our project based teaching

An admin (professor) can create courses (by semester)

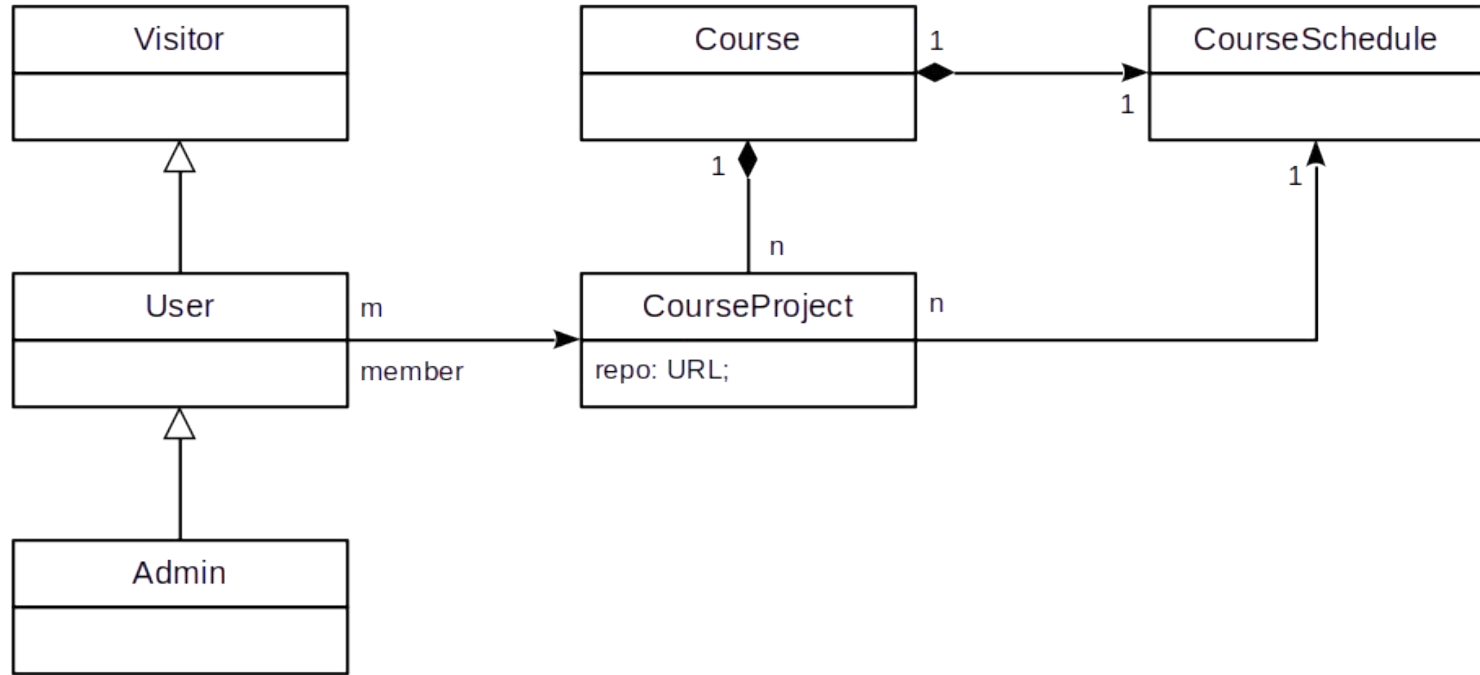
Courses have an associated schedule (homework delivery dates)

Users (anyone) can create projects and add them to courses

A project can have one or more members (ADAP = 1, AMOS = 6..12)

A project is linked to exactly one GitHub repository

# Class Model



DR

# Core Functions

---

For a given project, a member

- Can enter their happiness
- Can send out stand-up emails
- Can review coding their activities

all scoped by the course schedule

## **2. Current State**

---



# Starting Mini Mecho

---

Assuming the user role

1. Delete server/myDatabase.db
2. Start frontend and backend
3. Create new user via signup
4. Update status to confirmed using DB Browser
5. Logout, login (double-click needed)

# Course Setup

Assuming the administrator role, on project admin screen

1. In project group list section
  - a. Create new semester
  - b. Create new course (= project group)
  - c. Click create twice
2. In project lists section
  - a. Enter course
  - b. Create new project
  - c. Click create twice

# Joining a Project

Assuming the user role, under configuration

1. In the settings section
  - a. Select a course
  - b. Join a project by clicking on +
  - c. Enter your role (you can invent something)
2. In the project config section (optional)
  - a. Select a project that you are a member of
  - b. Provide a GitHub repository address

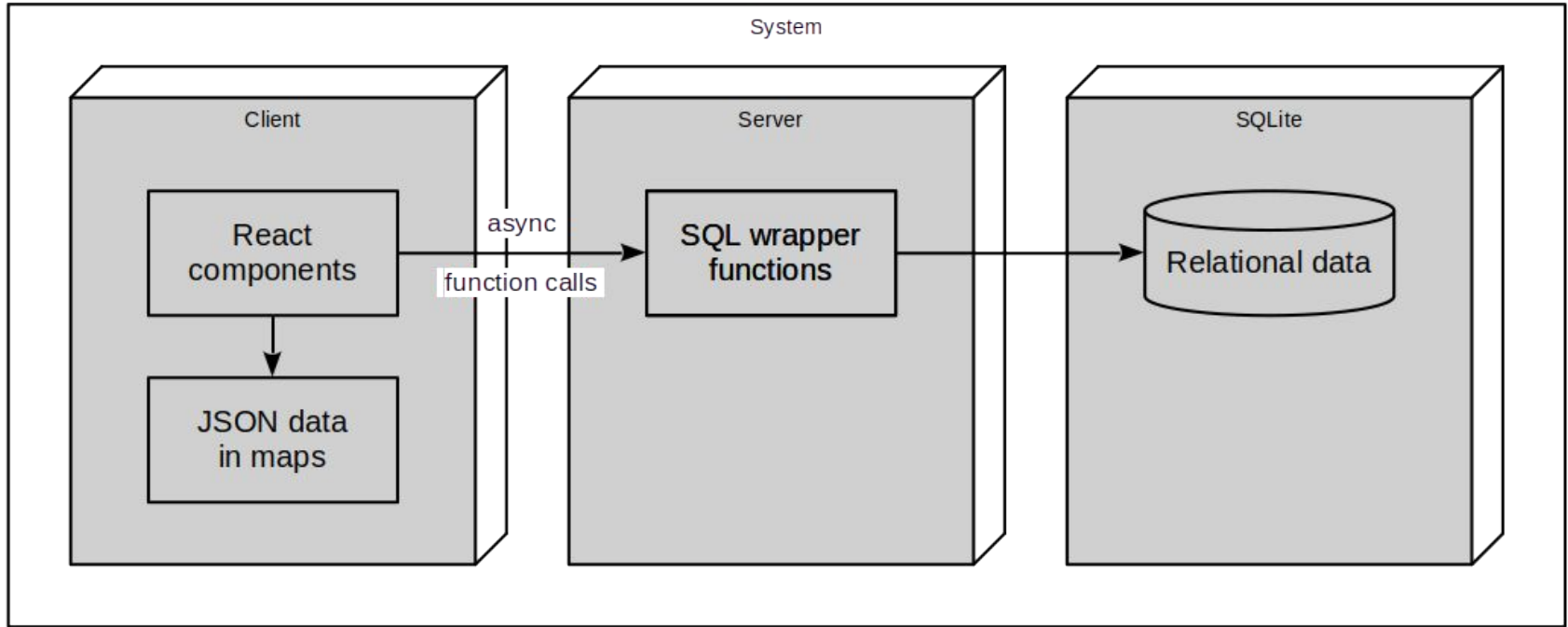
# Using Functions

---

Assuming the user role

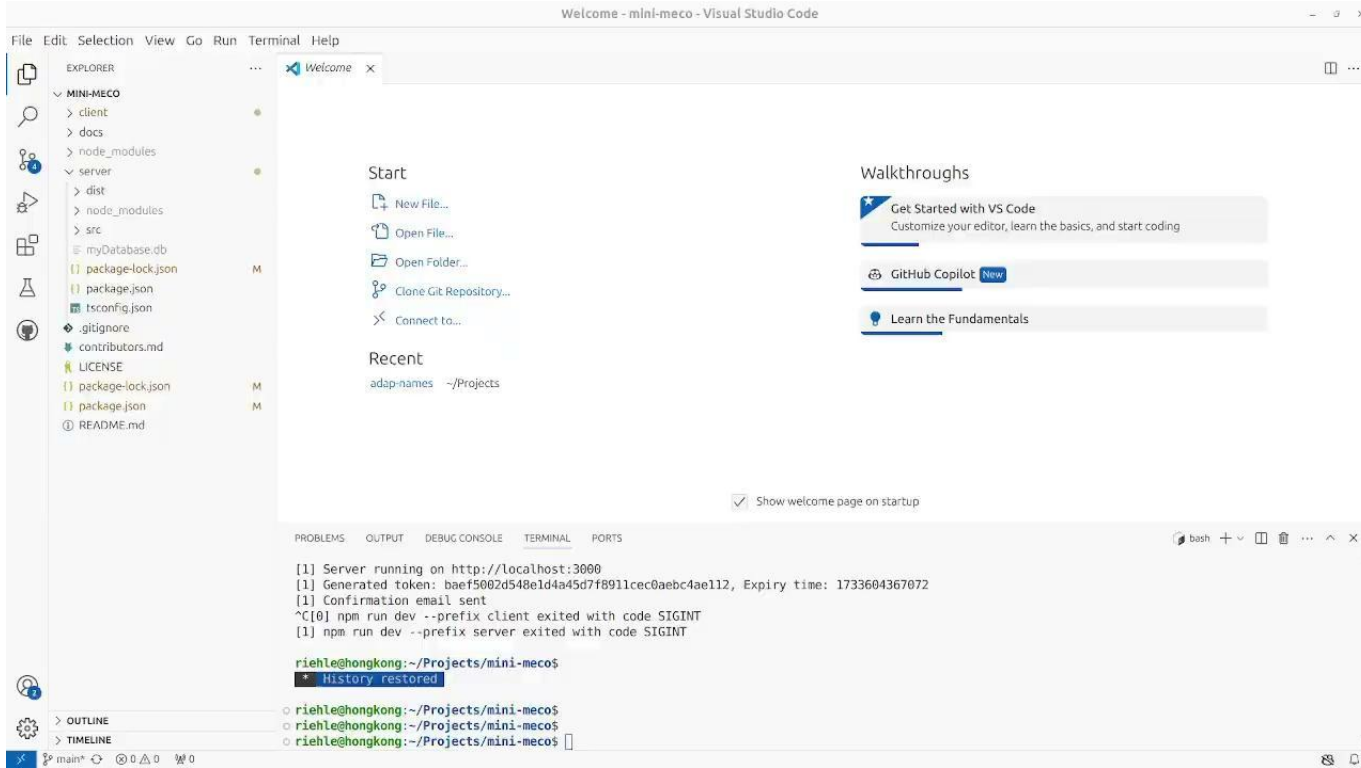
1. Select a project
  - a. Write stand-ups
  - b. Provide happiness
  - c. Review code activity (not working)

# Current Mini Meco Architecture



DR

# Video Demonstration



### **3. Goal State**

---

# Project Mission

---

Learn how to refactor an application in need of improving

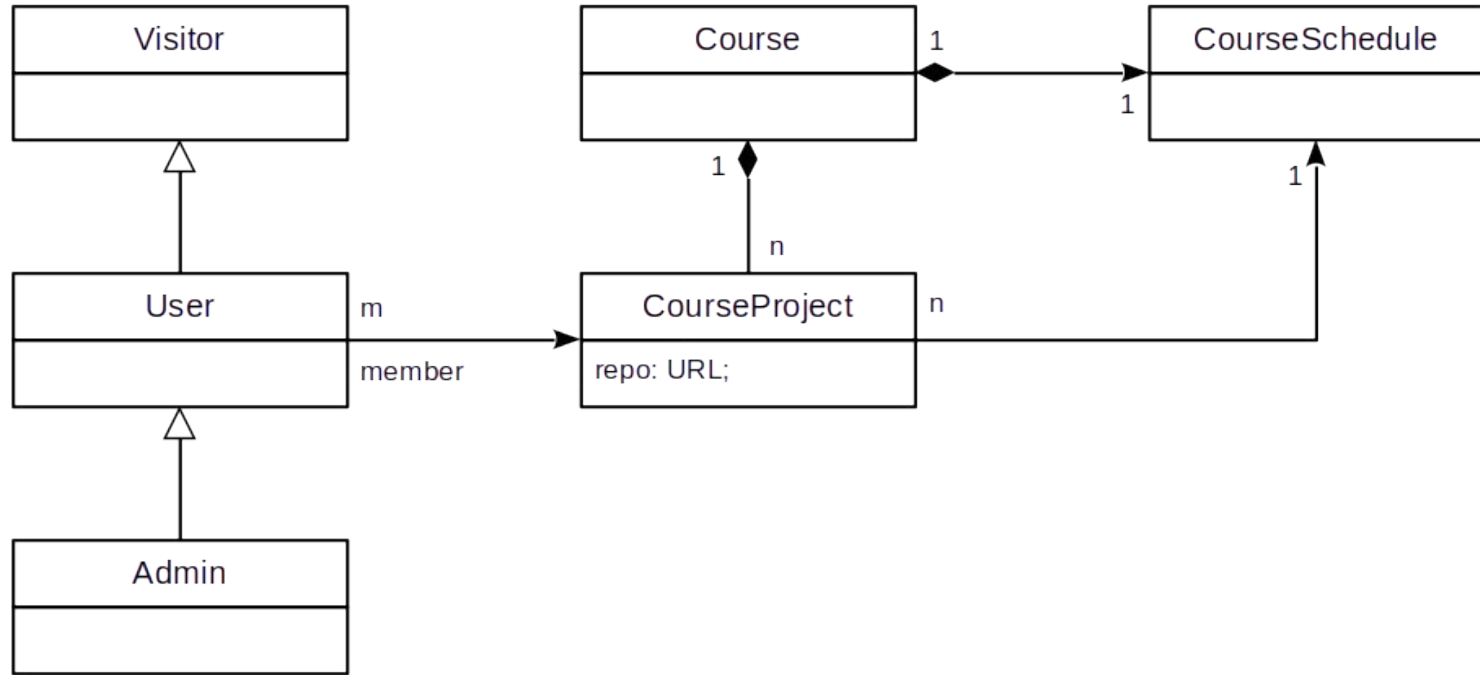


# Goal State Overview

---

1. Create domain model
  - a. Use class model from before
  - b. Introduce proper value types
2. Fix user interface (still React)
  - a. Fix logical structure of UI
  - b. Connect to domain model
3. Map to database (still SQLite)
  - a. Fix database schema
  - b. Use serializer pattern

# Class Model (Recap)



# (Potential) Value Types

---

User name, password

User life-cycle status (probably enums)

Access rights

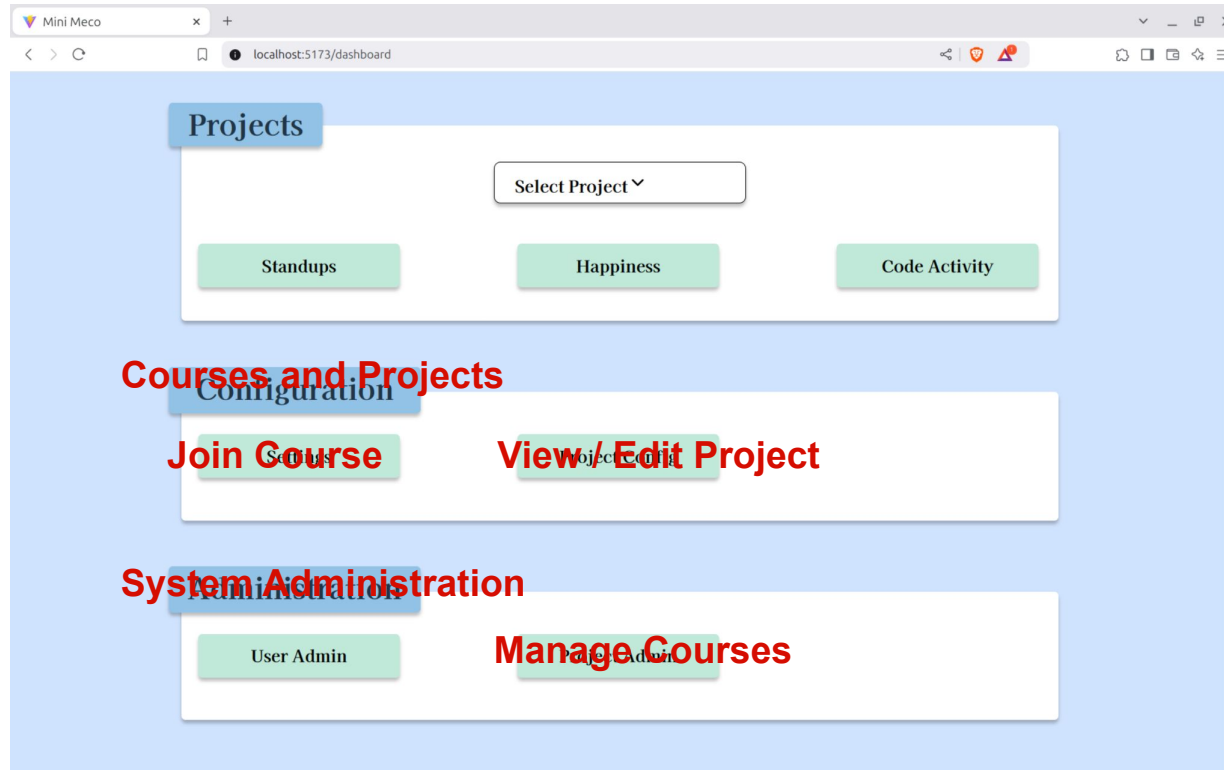
Happiness value (-3..0..3)

Timestamp, time, date, week, calendar year, semester, academic year

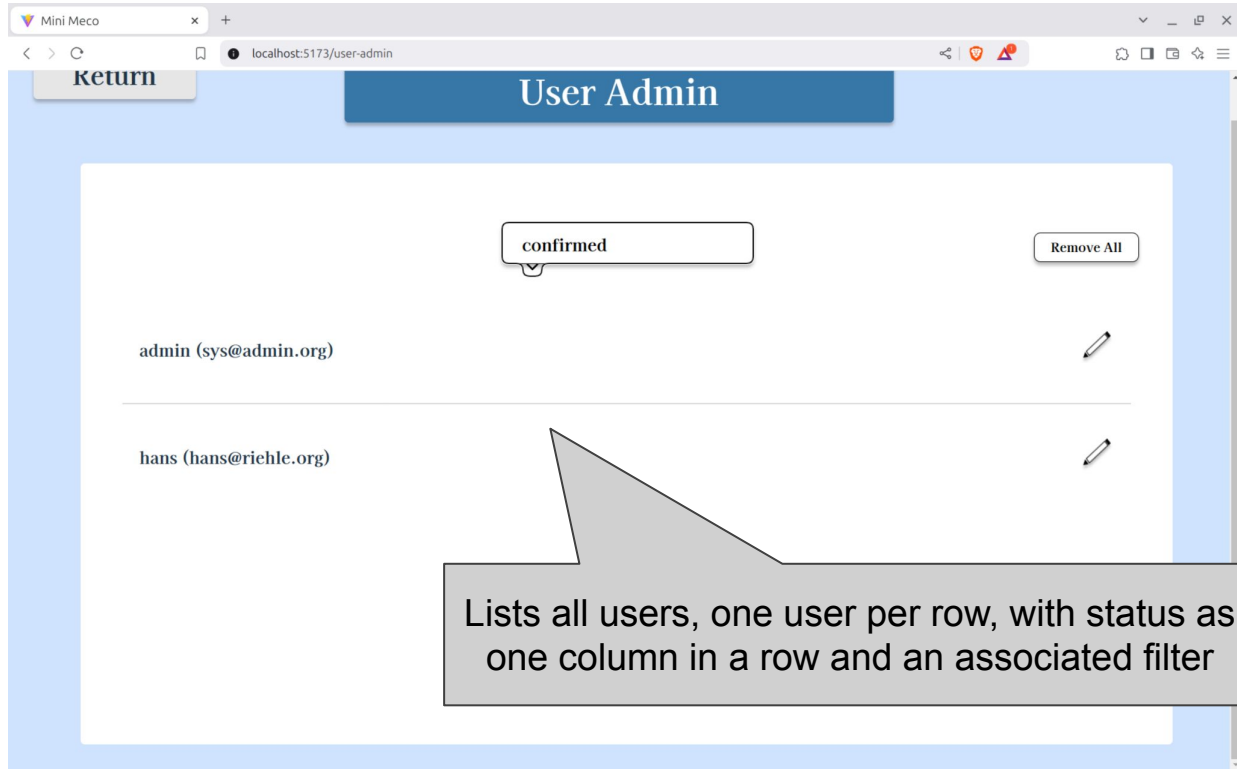
Commit

URL

# Dashboard



# Admin: User Admin



# Logic of the Course Administration Screen

---

Three sections: Semester (top) → Course (middle) → Project (bottom)

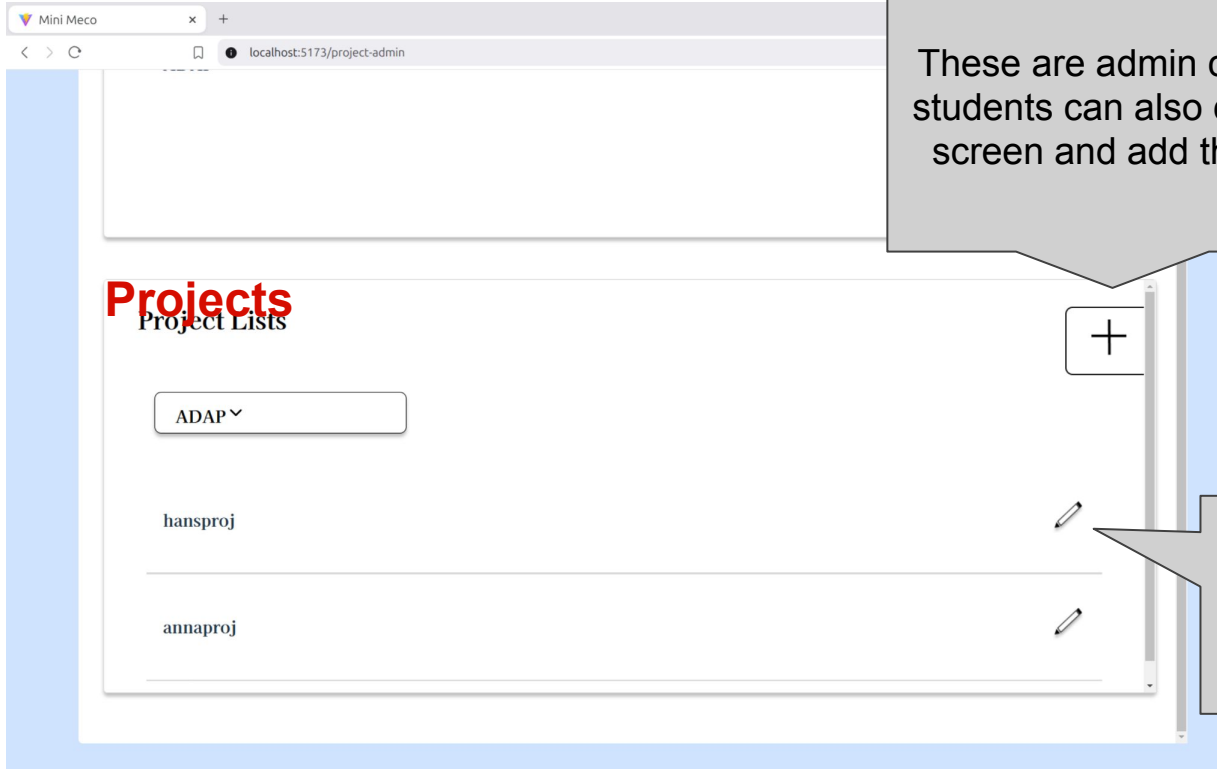
Each section above provides the context for the section below

# Admin: Project Admin 1

The screenshot shows a web browser window with the address bar displaying 'localhost:5173/project-admin'. The page has a light blue header with a 'Return' button and a 'Project Admin' title. Below the header, there is a 'Course Administration' section with a red 'Courses' title. Under 'Courses', there is a 'Project Group Lists' section with a dropdown menu showing 'SS25' and a list item 'AMOS' with a pencil icon. To the right of the 'Project Group Lists' section is a plus sign. Below the 'Project Group Lists' section is a 'Project Lists' section with a plus sign.

Split into semester section and course section with separate CRUD functions; make sure an object's id is separate from their name (and that the objects can be renamed i.e. "WS24/25" to "Winter 2024/25")

# Admin: Project Admin 2

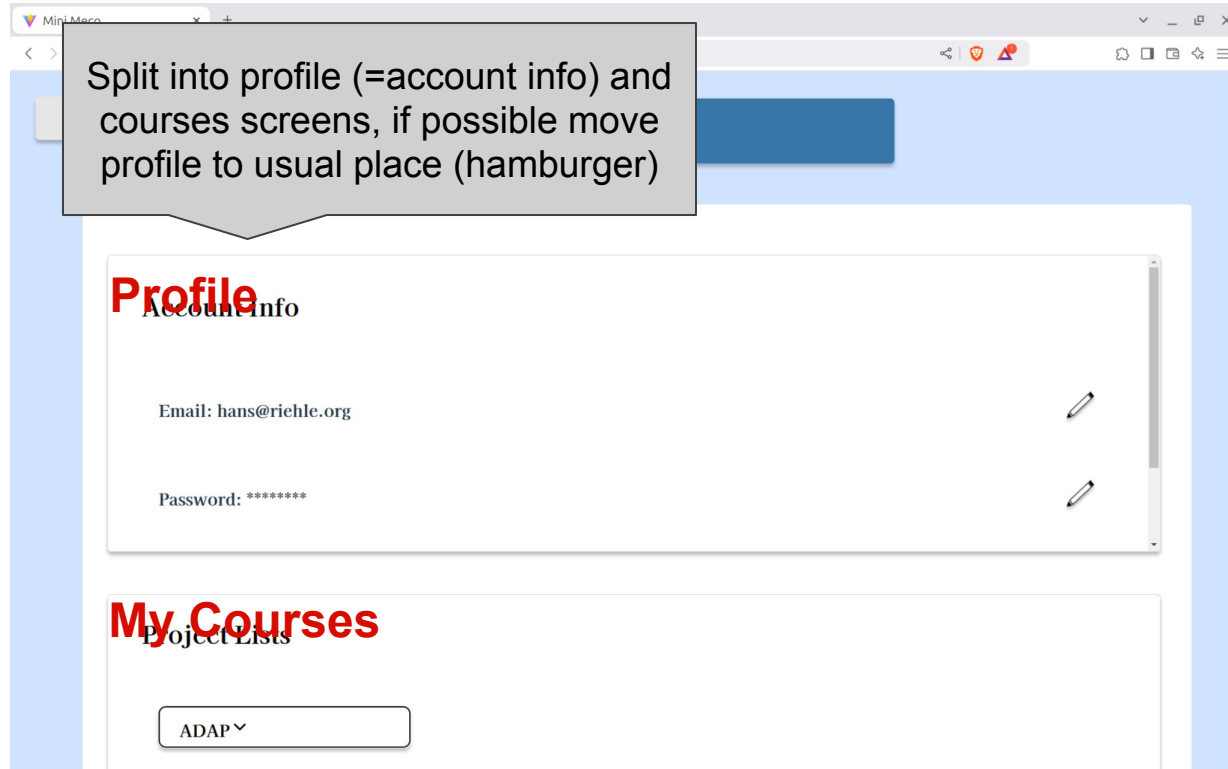


These are admin created projects (AMOS); students can also create projects in another screen and add them to a course (ADAP)

The edit dialog should allow editing of all project fields; however, only owners can edit, others can only join and view



# User: Settings



# User: Project Config

Mini Meco

localhost:5173/project-config

Return

**Project Configuration**  
Project Config

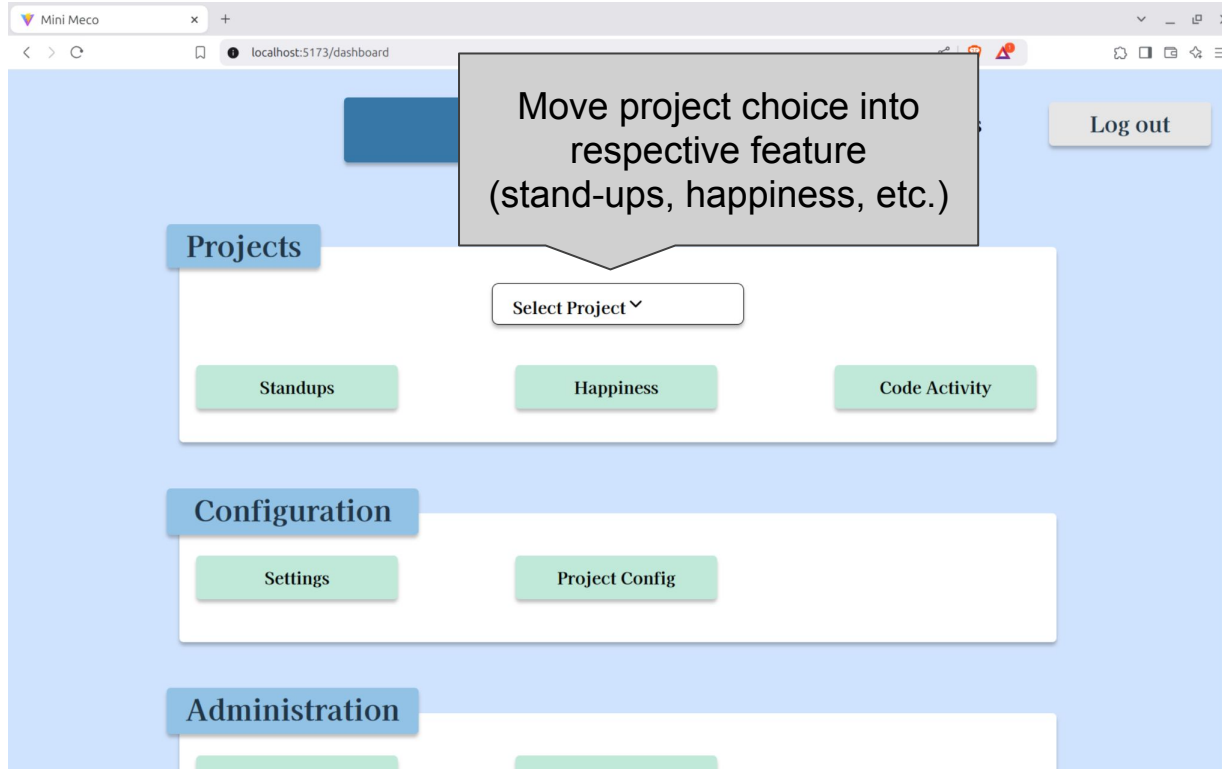
hansproj

Git URL

https://github.com/shumancheng/Mini-Meco

Nothing to fix but visuals  
and CRUD on object; a  
user can only edit  
projects they created

# User: Dashboard



## **4. Optional Homework**

---

# Mini Meco Extravaganza Homework is Optional

---

You will not be penalized if you don't participate

You'll get extra credit if you participate

Extra credit might lift your grade if you are at a grade boundary

I hope it will be fun; fun as in horror movies can be fun

Or, view this experience as an idea of your first industry job

# Mini Meco Extravaganza Timeline

Week	Dates	Topics
9	Jan-13 to Jan-20	UI fixes and improvements
10	Jan-20 to Jan-27	Domain model (classes)
11	Jan-27 to Feb-03	Value types introduction

# Mini Meco Extravaganza Timeline

---

Think about your possible contributions until next week (Dec 16)

Let us know by regular class time if you'd like to contribute

Sign up here <https://forms.gle/AD2Q7GT2K8QspcT37>

We will assign features using a feature board in time

<https://github.com/orgs/riehlegroup/projects/3>

You can also grab features yourself

# Thank you! Any questions?

---

[dirk.riehle@fau.de](mailto:dirk.riehle@fau.de) – <https://oss.cs.fau.de>

[dirk@riehle.org](mailto:dirk@riehle.org) – <https://dirkriehle.com> – [@dirkriehle](#)



# Legal Notices

---

## License

- Licensed under the [CC BY 4.0 International](https://creativecommons.org/licenses/by/4.0/) license

## Copyright

- © 2012, 2018, 2024 Dirk Riehle, some rights reserved