**The Hong Kong Polytechnic University**

**Department of Computing**

COMP4913 Capstone Project

Report (Final)

An online system for examination timeslot allocation

Student name: Lee Ka Shue

Student ID No.: 19034824d

Program-stream code: 61431-FCS

Supervisor: Dr WANG Qixin

Co-Examiner: Dr SAXENA Divya

2nd Assessor: Prof. TAN KC

Submission date: 12 April 2024

# Abstract

This report briefly states the importance of an examination timeslot allocation system for higher educational institutes, outlines previous efforts in implementing such a system and its shortcomings, describes the steps taken to improve the project, and covers the shortcomings within the project and future mitigation steps.

The final project has partially completed the outcomes and objectives stated during the project proposal phase, improving the project’s development sustainability. However, due to the choice of framework has a high amount of complexity, caution is suggested for prospective contributors.

Contents

[Abstract 2](#_Toc162109196)

[List of tables and figures 5](#_Toc162109197)

[Background & problem statement 6](#_Toc162109198)

[Introduction 6](#_Toc162109199)

[Background 6](#_Toc162109200)

[Problem 6](#_Toc162109201)

[Objectives & outcomes 7](#_Toc162109202)

[Objectives 7](#_Toc162109203)

[Outcomes 7](#_Toc162109204)

[Implementation 8](#_Toc162109205)

[General 8](#_Toc162109206)

[Technical design 8](#_Toc162109207)

[Framework 8](#_Toc162109208)

[Database design 9](#_Toc162109209)

[Functional business logic 14](#_Toc162109210)

[UI/UX design 14](#_Toc162109211)

[Localization / Internationalization (l10n / i18n) 14](#_Toc162109212)

[Interaction routes 15](#_Toc162109213)

[Code health 17](#_Toc162109214)

[Reliability assurance 17](#_Toc162109215)

[Documentation 17](#_Toc162109216)

[Project structure 19](#_Toc162109217)

[Convention adherence 20](#_Toc162109218)

[Migration checklist 21](#_Toc162109219)

[Objective and outcome checklist 21](#_Toc162109220)

[Postmortem 22](#_Toc162109221)

[Overpromising & underachieving 23](#_Toc162109222)

[Conclusion 24](#_Toc162109223)

[Future work 24](#_Toc162109224)

[Resource utilization 24](#_Toc162109225)

[Used software 25](#_Toc162109226)

[Appendices 26](#_Toc162109227)

[Appendix A 26](#_Toc162109228)

[Appendix B 27](#_Toc162109229)

[Appendix C 28](#_Toc162109230)

[Appendix D 29](#_Toc162109231)

# List of tables and figures

[Figure 1 Database schema ER diagram of previous work 9](#_Toc162109158)

[Figure 2 Database schema of current work 10](#_Toc162109159)

[Figure 3 Simplified ER diagram from the teacher's POV 12](#_Toc162109160)

[Figure 4 Simplified ER diagram from the student's POV 13](#_Toc162109161)

[Figure 5 Web GUI in en locale 14](#_Toc162109162)

[Figure 6 Web GUI in zh\_HK locale 15](#_Toc162109163)

[Figure 7 GUI of teacherview in previous work 16](#_Toc162109164)

[Figure 8 Web GUI of allocation results of current implementation 16](#_Toc162109165)

[Figure 9 The only documentation of previous work 17](#_Toc162109166)

[Figure 10 Documentation of timeslot generation code 18](#_Toc162109167)

[Figure 11 Project structure of previous work 19](file:///D:\Users\Raphael\Desktop\PolyU\Y5\Capstone\ProjectRepo\BatchMeetingScheduler_LEE_Ka_Shue\documents\report\Final_report_19034824D_LEE_Ka_Shue.docx#_Toc162109168)

[Figure 12 Project structure of current work 20](file:///D:\Users\Raphael\Desktop\PolyU\Y5\Capstone\ProjectRepo\BatchMeetingScheduler_LEE_Ka_Shue\documents\report\Final_report_19034824D_LEE_Ka_Shue.docx#_Toc162109169)

[Figure 13 Project directory and file names of previous work 20](#_Toc162109170)

# Background & problem statement

## Introduction

COVID has caused major disruptions in our daily lives, including the education processes. As face-to-face interactions have been discouraged during the period, examinations were also to be conducted online, which brought forth the problem, invigilating a couple dozen of students through webcam feeds concurrently being quite difficult. While a severe resource demand, 1:1 monitoring of students is a much-needed process to eliminate cheating and ensure fairness. This then led to the need of allocating timeslots for individual students’ examinations.

Although COVID-19 is now a part of history, there are no guarantees that no future circumstance necessitates online examinations. Hence, a suitable platform is needed to coordinate the timeslot allocation process.

## Background

An existing teacher-student meeting reservation system has previously been extended and improved by senior students into a teacher-student examination timeslot preference/allocation system, and student Yu’s project has been selected to be iterated upon by this year’s students.

## Problem

For this year, while not having specified a problem, us students have been tasked to “perfect” the system.

Having seen the project demonstration, it can be seen that the finished work is more-or-less in a prototype state, which may require some further development and polishing to reach a production ready state. But upon further inspection, the maintenance of the project may prove difficult.

# Objectives & outcomes

## Objectives

As aforementioned, the given objective is to:

1. “Perfect” the system.

Additionally, the major objectives of the Capstone Project subject should be considered as well:

1. Develop generic competencies
2. Prepare for workplace professional practice
3. Further academic pursuits
4. Lifelong learning

## Outcomes

During the project proposal and project initialization phase, I concluded that the following facets of the system will need to be faced with scrutiny to identify suboptimal implementations and be iterated upon:

1. Technical design
   1. DB-level design
   2. Functional business logic
2. UI/UX design
   1. Language
   2. Localization / Internationalization
   3. Interaction flow
3. General code health
   1. Code tests
   2. Documentation
   3. Project structure
   4. Convention adherence

# Implementation

## General

As an unfortunate compromise out of my full-time SWE workload, outside of the aforementioned areas that needs improvement, the database administration and user interaction platform has also been changed to utilized JMix instead of MyPHPAdmin.

## Technical design

### Framework

#### Existing implementation

Yu’s work utilizes PhpMyAdmin, MySQL, PHP, and Apache to power the database and web interface (Appx. A).

#### Current implementation

As the author, I, have had minimal experience with web development, I have to learn all basics from the start. As such, whether continuing development with PHP and learning it, or implementing it from scratch using another framework is a decision I had to make.

Personally, I was intrigued by the Phoenix framework with Elixir as the programming language, as I have seen testimonials about its performance capabilities and am interested in learning functional programming.

Professionally, I consulted my internship workplace supervisor, and was recommended to use the Jmix framework with Java as the programming language, as the company uses that framework for multiple of their solutions as well, and I would be tasked on the job to use this framework.

Resultantly, I have decided to listen to my workplace supervisor due to multiple reasons:

1. Full-time software engineering work severely limits available time to learn new technologies and develop with it. If Jmix was used, the experience during work will help with my capstone project and vice versa.
2. Jmix is an open-source, bundled implementation of Vaadin, a frontend framework, and Spring, a backend framework. This allows quicker development of features and can empower me to do better documentation with the added time.
3. Utilizing Spring, Jmix is a solidly enterprise solution, as Spring is an overwhelmingly popular framework in the business world. This will enable future project contributors’ and my career development.
4. I was recommended by the one who feeds me.

Additionally, there are benefits of built-in user system, resulting in better security.

### Database design

#### Existing implementation

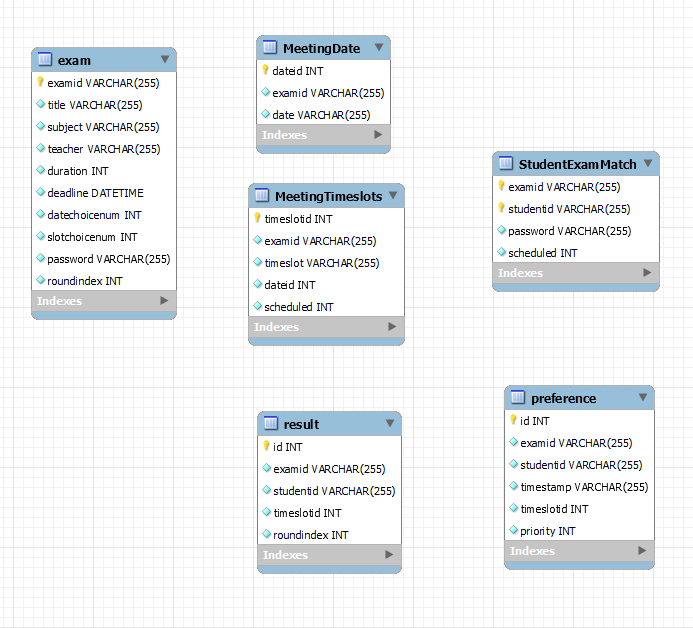


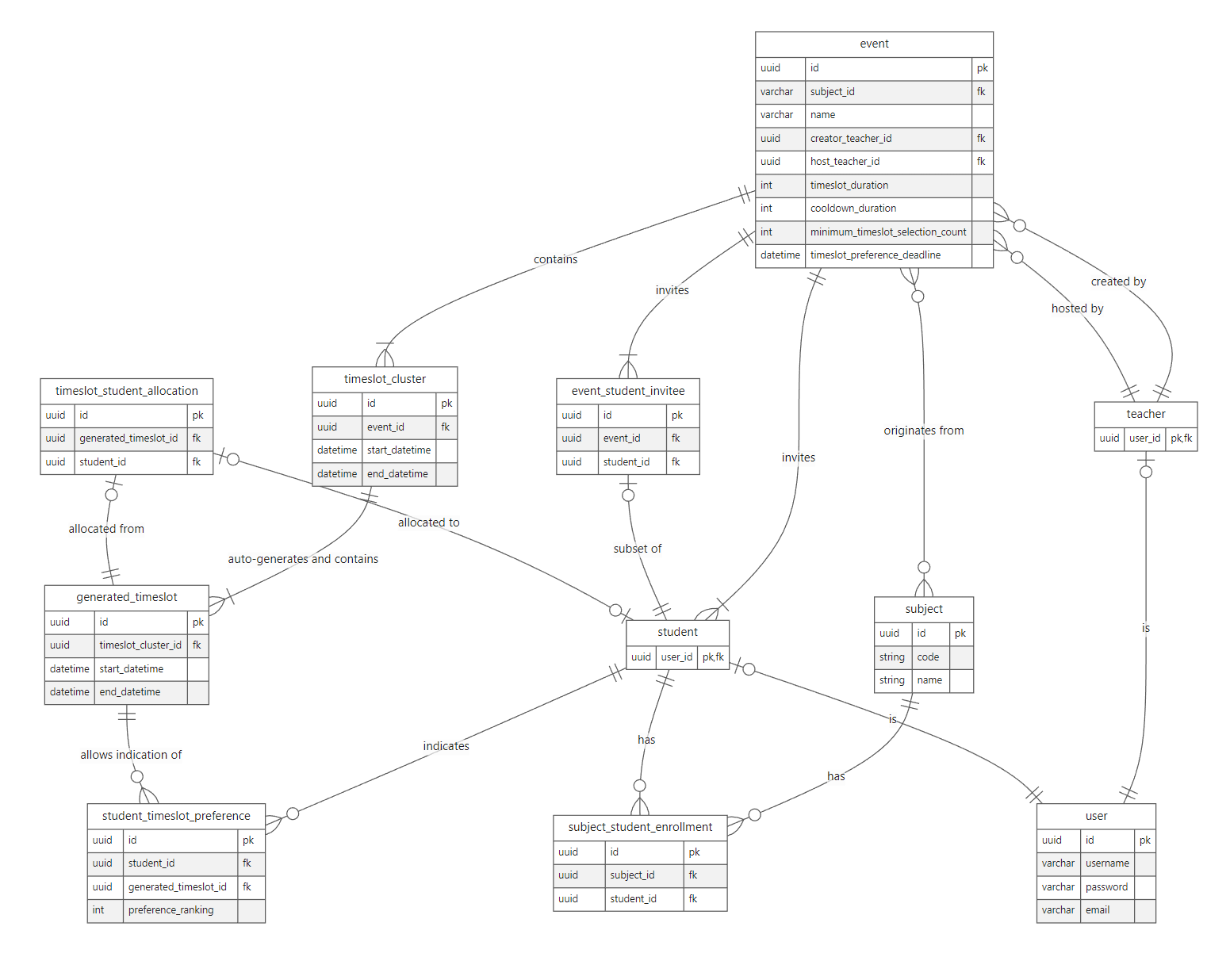
Figure 1 Database schema ER diagram of previous work

Above is the ER diagram of YU’s existing work, generated utilizing MySQL workbench’s model generation function from the provided SQL script with minor amendments on table studentexammatch’s name casing to avoid errors.

Interestingly, the SQL script seemingly does not enforce entity relationships. The naming scheme for tables and columns are less than ideal as well, with inconsistent table casing patterns, non-existent column casing patterns, inconsistent terminology, and vague naming. Following text references to such information will follow the casing convention described in [Convention adherence](#_Convention_adherence)

To give the benefit of doubt, as I am not familiar with PHP, the relationship business logic could be stored in that layer. As such, please refer to YU’s final report’s diagram (Appx. B) for the intended relationships.

#### Current implementation

Figure 2 Database schema of current work

Above is the ER diagram of my work, generated utilizing Obsidian and Mermaid.JS from my living design / implementation markdown document. For parity, a diagram is also generated using MySQL workbench, which is relegated to the (Appx. C).

There is a separation of graphics as there are some framework-induced additional tables and columns that are unnecessary for understanding the design, and I have either abstracted away or removed them.

##### Normalization

During the design of the new DB schema, I had considered the spectrum across de-normalization and normalization.

|  |  |  |  |
| --- | --- | --- | --- |
| Approach | Normalization | … | Denormalization |
| Consistency | Better |  | Worse |
| Complexity | Worse |  | Better |

After more research, I have discovered the distinction between Online analytical processing (OLAP) and Online transactional processing (OLTP) in the software world, where OLAP is geared towards performant analytics by de-normalization, whereas OLTP is geared towards consistent processes by normalization.

Therefore, the new implementation opted for a comparatively normalized approach, as the information are not used in a throughput-first manner.

##### Expansion

There are also new tables introduced, including subjects and subject enrollments, as these tables would bring forth a greater quality of life for users by simplifying certain processes, and this information should be readily available in the school’s systems.

For the ease of understanding, the above diagram will be further simplified to the point of view (POV) of the teacher and student respectively

##### Teacher POV

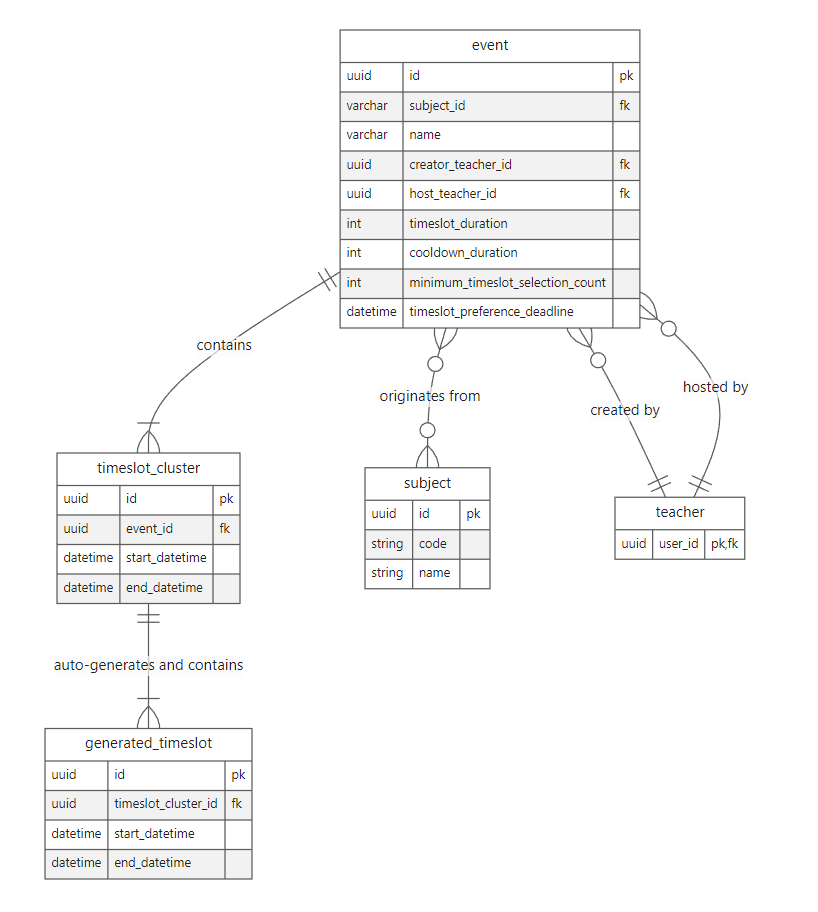


Figure 3 Simplified ER diagram from the teacher's POV

The new implementation majorly deviates from the previous’ in the following places:

1. MEETING\_DATE 🡪 TIMESLOT\_CLUSTER   
   Abstracted date separator into cluster for higher flexibility
2. Various normalizations by removing redundant columns
3. Explicit start and end times, and also a specified EVENT.cooldown\_duration, which allows for better timeslot scheduling and explicit communication
4. Separated event creator and host (i.e. possibility of explicit declaration by event creating professor of invigilating assistants)

##### Student POV

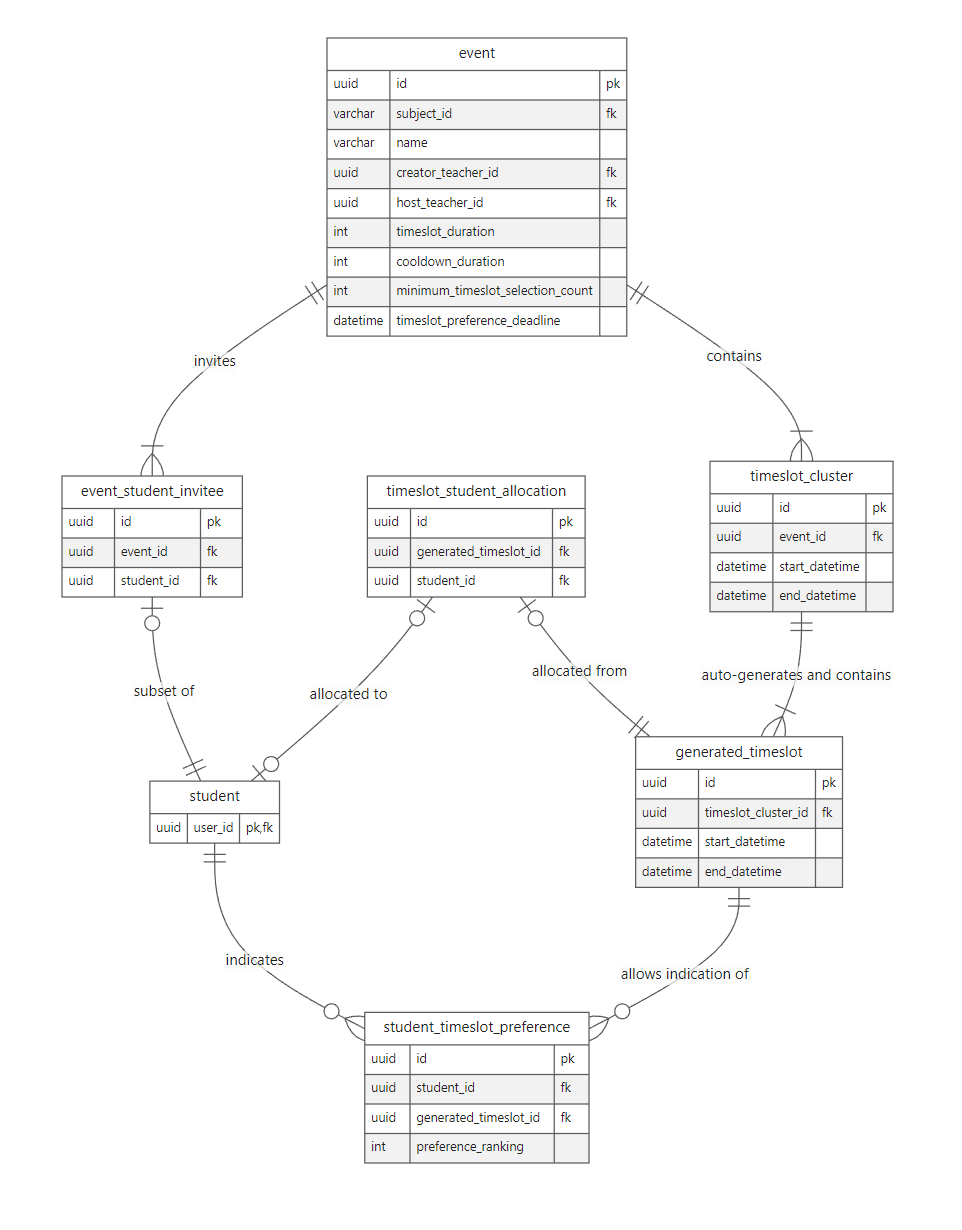


Figure 4 Simplified ER diagram from the student's POV

The new implementation majorly deviates from the previous’ in the following places:

1. TIMESLOT\_CLUSTER (formerly MEETING\_DATE) no longer directly related to EVENT\_STUDENT\_INVITEE (formerly STUDENT\_AND\_EXAM\_MATCHING)

### Functional business logic

To accommodate the new schema, changed framework, and altered UI/UX design, the relevant code has been altered accordingly, such as the timeslot generation code, in order to accommodate for EVENT inter-timeslot cooldown\_duration.

## UI/UX design

### Localization / Internationalization (l10n / i18n)

As a framework feature, the interface now has multi-language support, and has EN and ZH\_HK locales implemented.

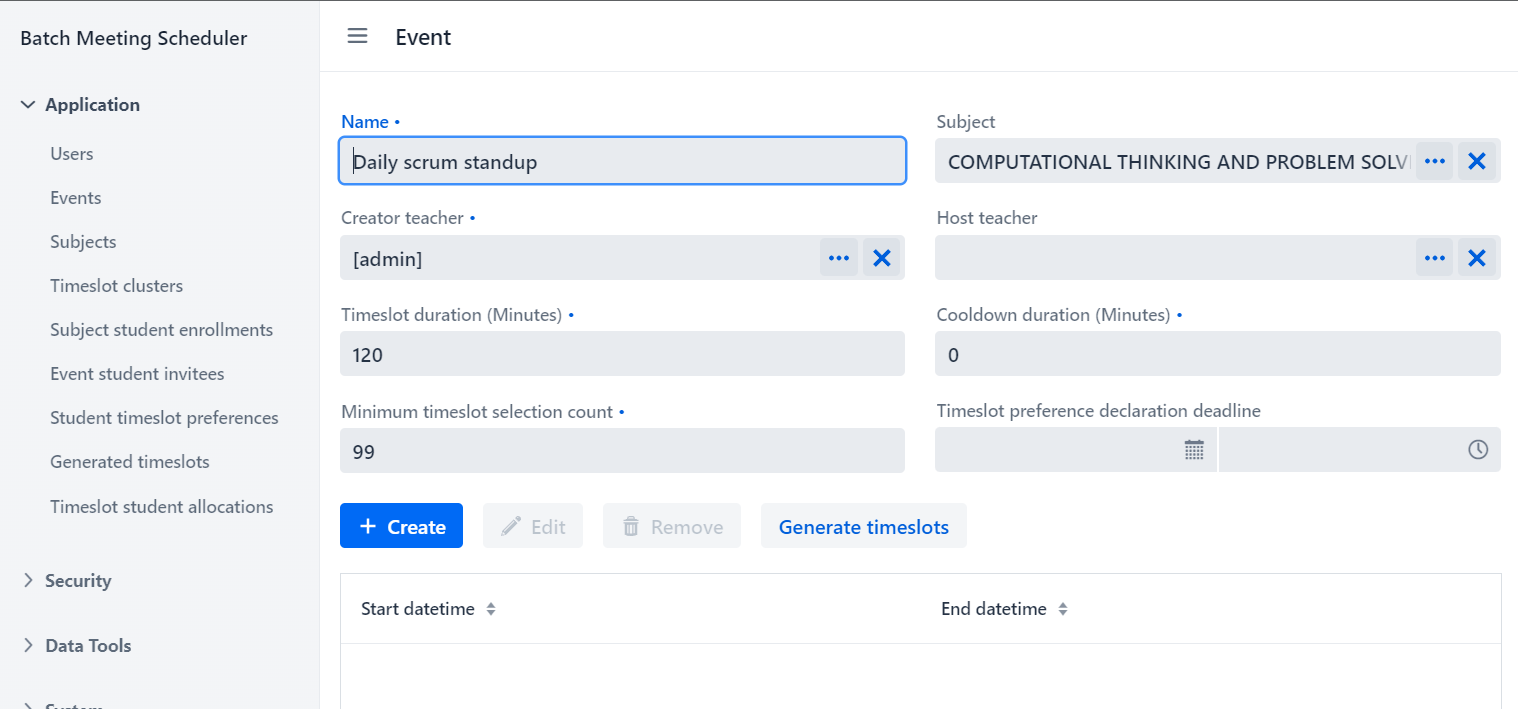


Figure 5 Web GUI in en locale

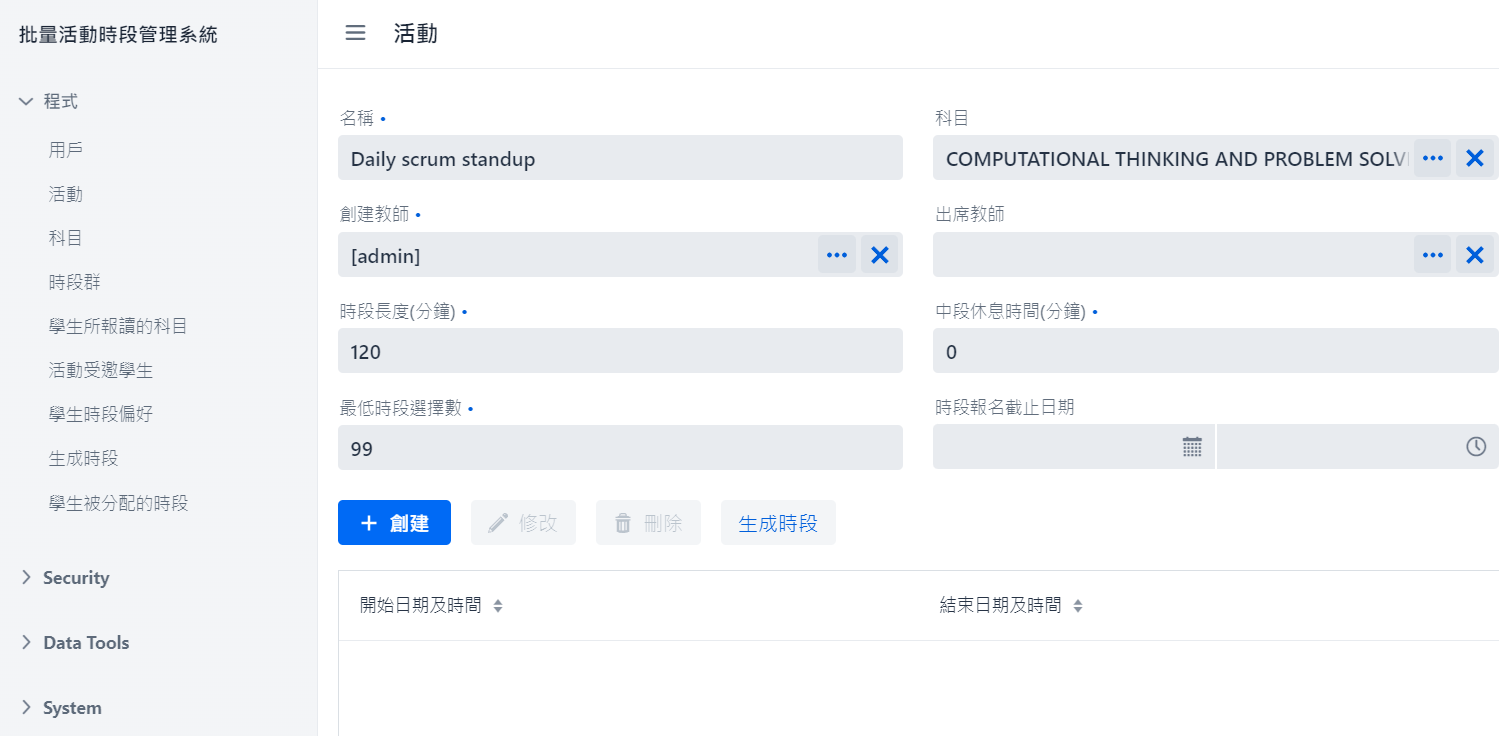


Figure 6 Web GUI in zh\_HK locale

### Interaction routes

The current revision has streamlined the UI, such as combining the view and edit pages of allocation results.

#### Previous implementation

Previously, the UI has been unnecessarily fragmented, and goes against the typical “modern” design, such as different navigation paths to the check results and edit results (Fig. 7).

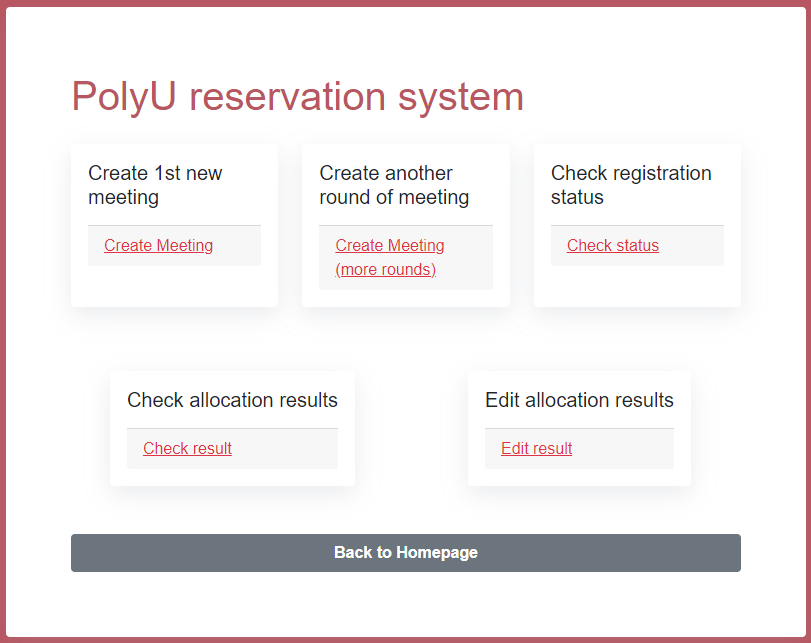


Figure 7 GUI of teacherview in previous work

#### Current implementation

Currently, the UI allows the user to view records, allocation results in this example, then navigate into a deeper route to create and edit (and remove) results.

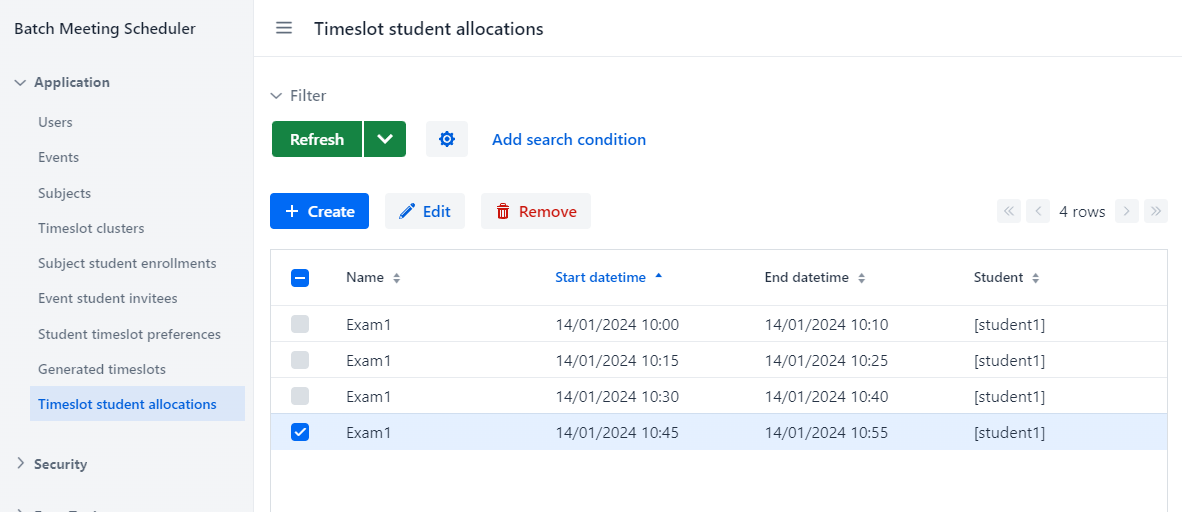


Figure 8 Web GUI of allocation results of current implementation

## Code health

### Reliability assurance

The project now has a minimal amount of tests covering user entities.

### Documentation

#### Previous implementation

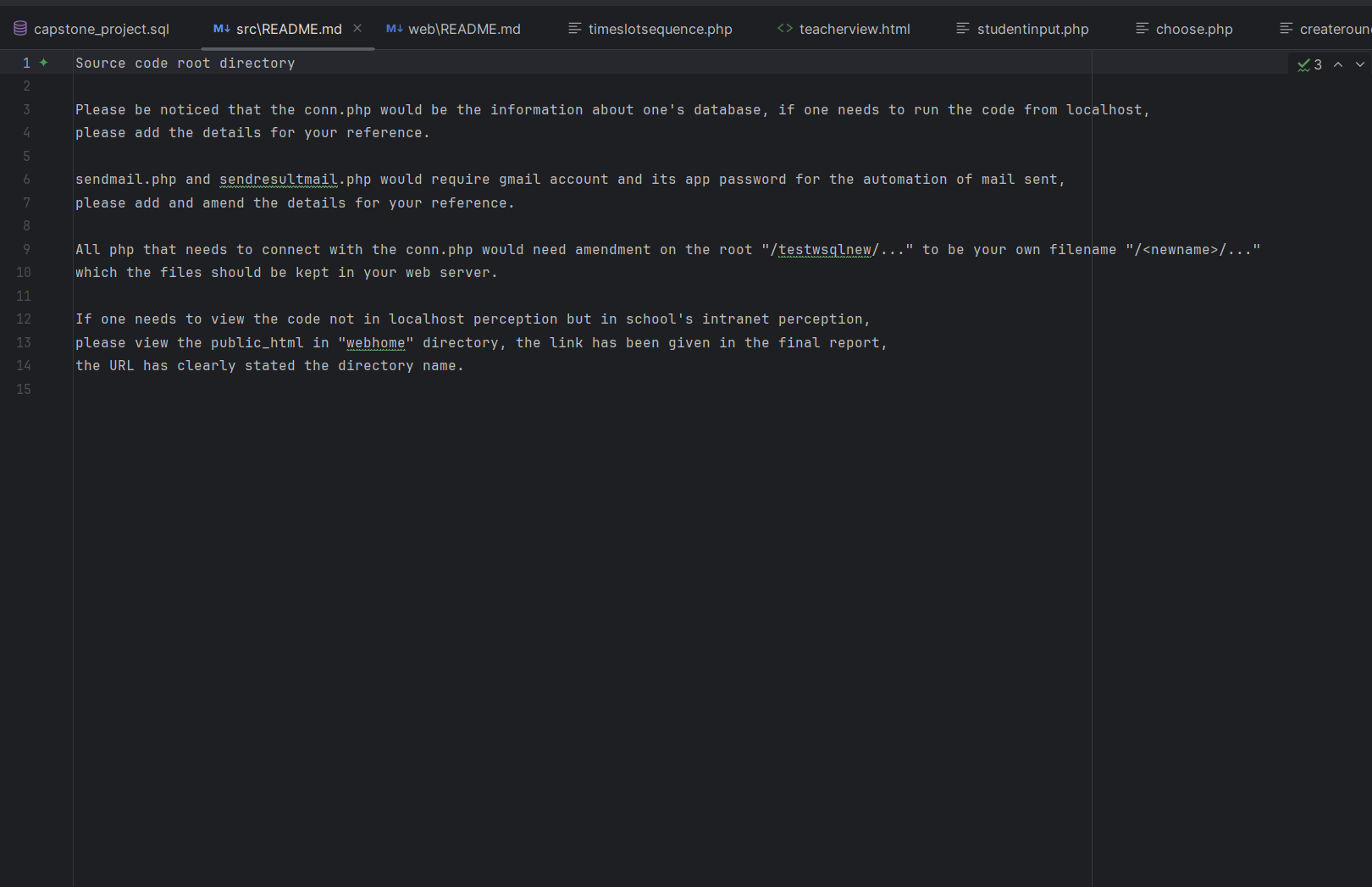


Figure 9 The only documentation of previous work

Previously, the repository has little-to-no documentation.

#### Current implementation

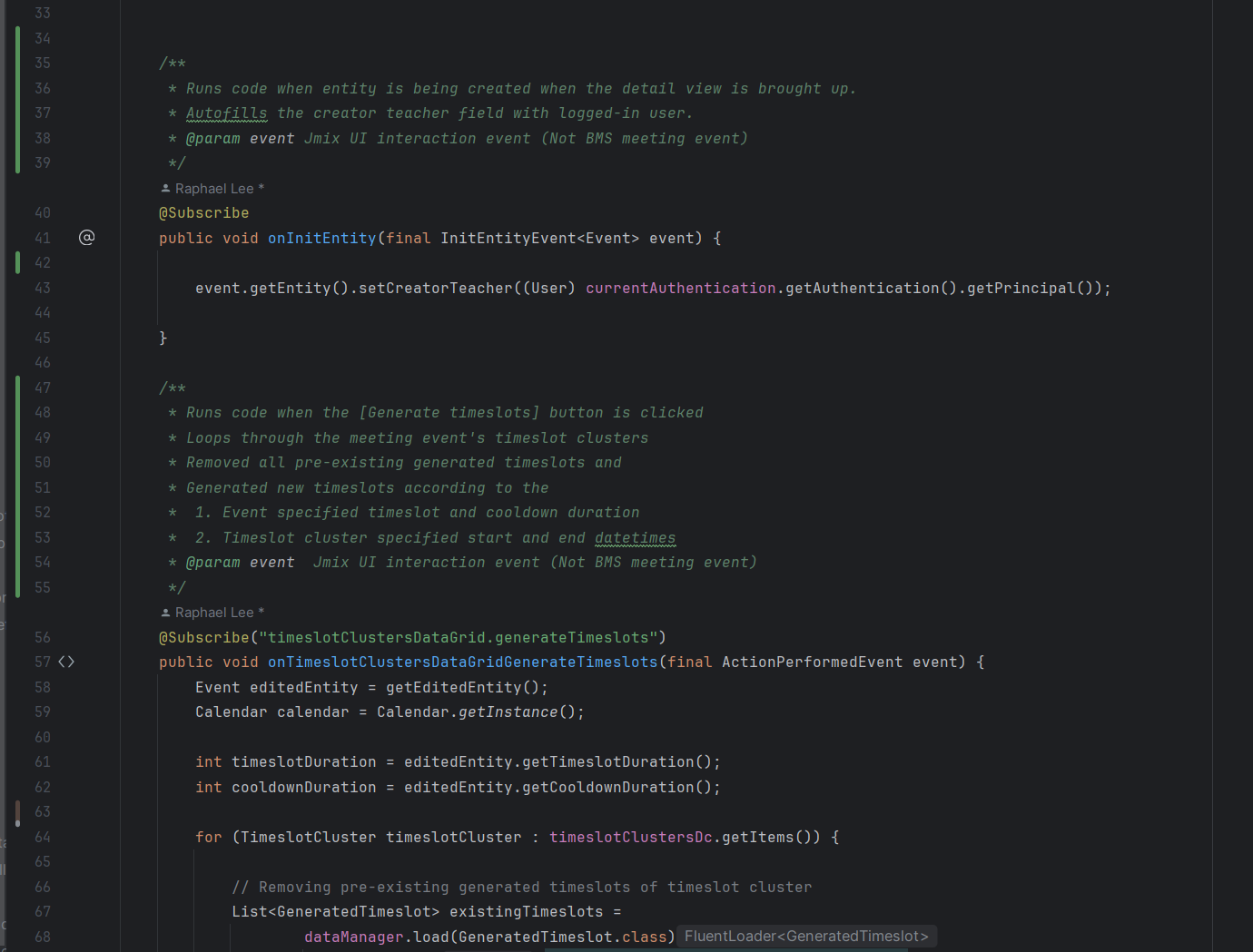


Figure 10 Documentation of timeslot generation code

Currently, manually created code are documented. Deployment guides are also authored for better handover experiences (Appx D.). Graphical elements are also written in an easily extendible way with Obsidian, Mermaid.JS and markdown files.

### Project structure

#### Previous implementation

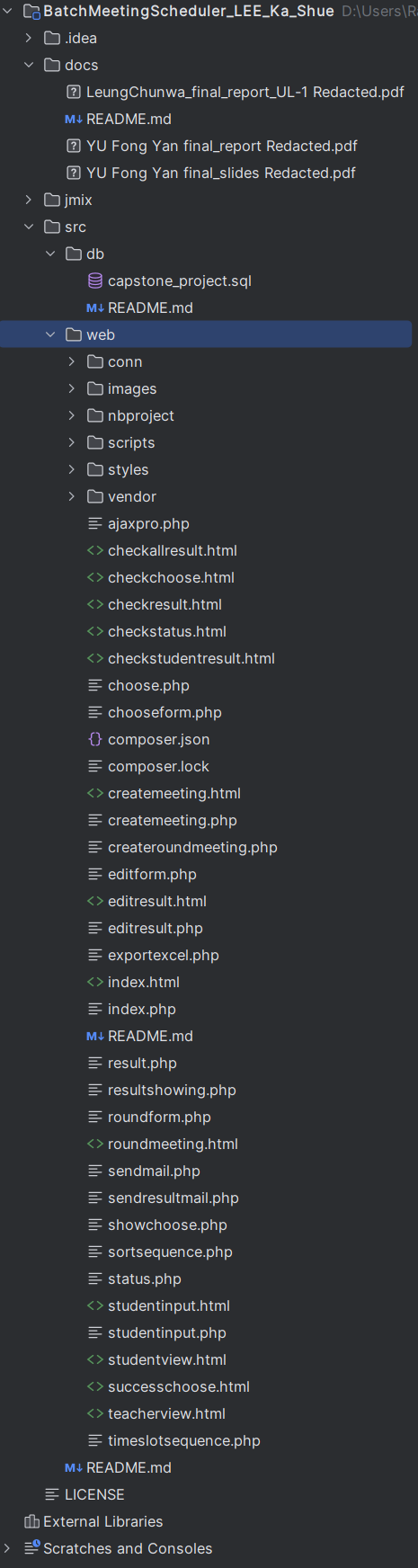
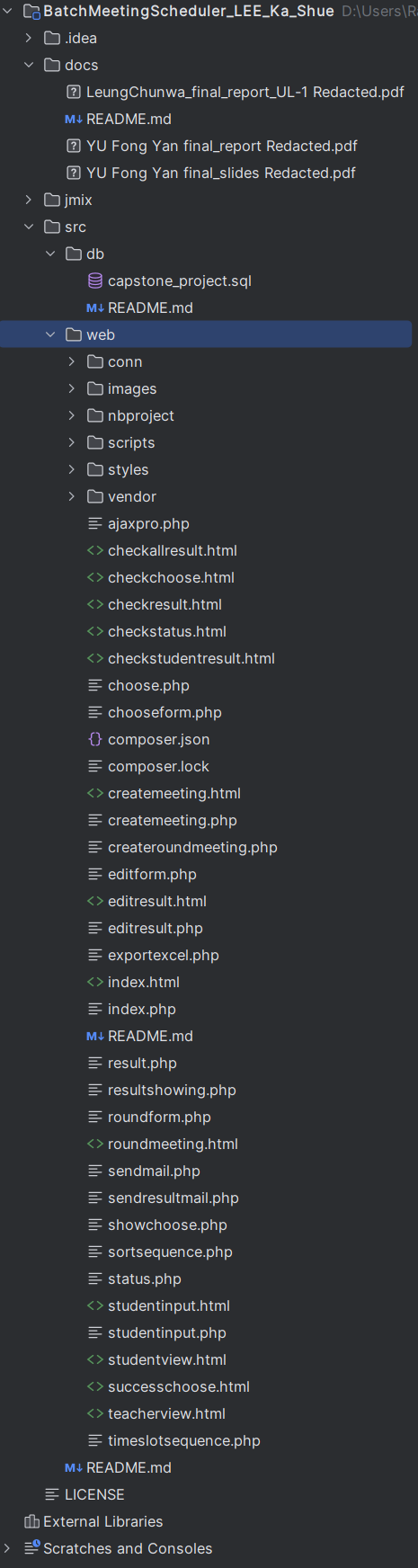


Figure 11 Project structure of previous work

#### Current implementation

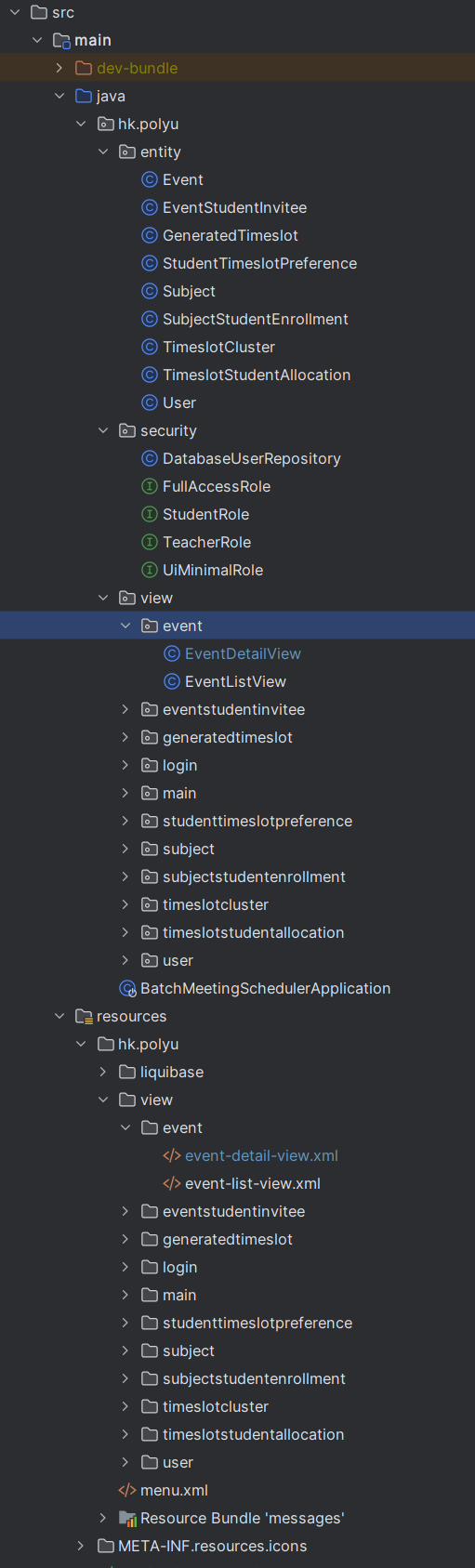
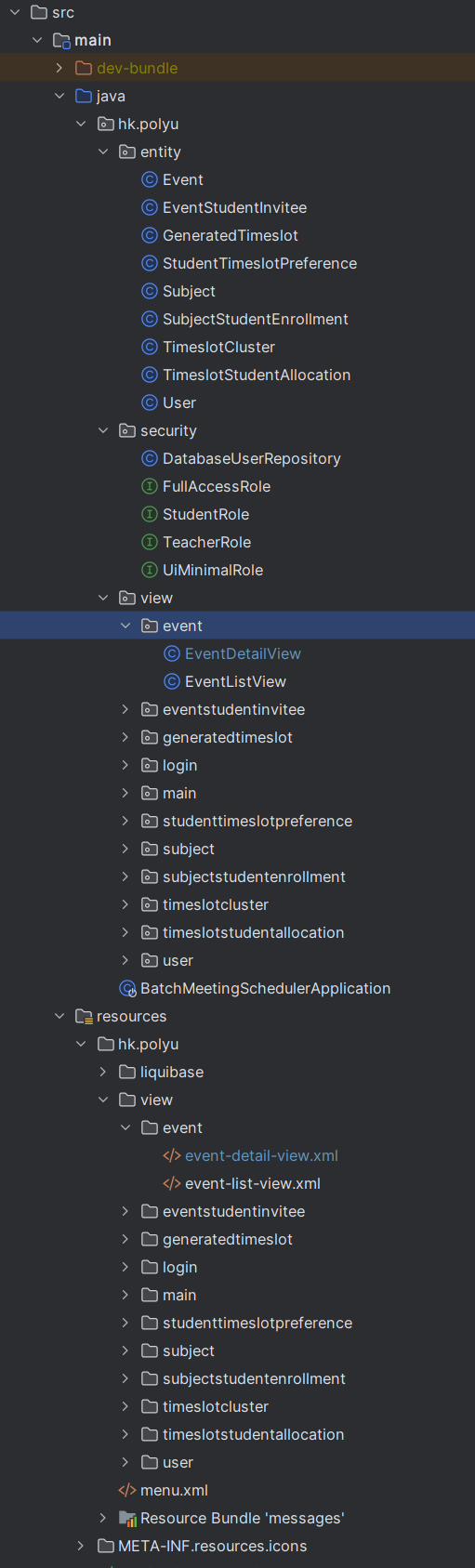


Figure 12 Project structure of current work

### Convention adherence

The code repository contents now follow a naming and casing convention.

Figure 13 Project directory and file names of previous work

|  |  |  |
| --- | --- | --- |
| Type | Casing (stylized) | Plurality |
| Table name | MACRO\_CASE | Singular |
| Column name | snake\_case | Singular |

|  |  |  |
| --- | --- | --- |
| Domain | Previous | Current |
| **Table** | MEETING | EVENT |
| **Table** | EXAM | EVENT |
| **Table** | MEETING\_TIMESLOTS | GENERATED\_TIMESLOT |
| **Table** | STUDENT\_AND  \_EXAM\_MATCHING | EVENT\_STUDENT\_INVITEE |
| **Table** | PREFERENCES | STUDENT\_TIMESLOT\_PREFERENCE |
| **Table** | RESULT | TIMESLOT\_STUDENT\_ALLOCATION |
|  |  |  |
| EVENT | title | name |
| EVENT | teacher | creator\_teacher |
| EVENT | duration | timeslot\_duration |
| EVENT | deadline | timeslot\_preference  \_declaration\_deadline |
| EVENT | slot\_choice\_num | minimum\_timeslot\_declaration\_count |

# Migration checklist

|  |  |
| --- | --- |
| Functionality | Migrated? |
| Database | Yes |
| Record input GUI | Partial |
| Timeslot generation | Yes |
| Student e-mail invitation | No |

# Objective and outcome checklist

|  |  |  |  |
| --- | --- | --- | --- |
| Objective / Outcome | | Chk | Description |
| 1 | Develop generic competencies | ✅ | Developed a database and front/backend system |
| 2 | Prep for workplace professional practice | ❔ | Developed a system with processes, efficiency, conventions, maintainability, etc. in mind  Neglected to communicate with professor |
| 3 | Further academic pursuits | ❔ | Performed minor research and write up in DB OLTP / OLAP design  Relates to CAP / PACELC theorem |
| 4 | Lifelong learning | ❔ | Learned some things about development, workplace, and myself? |
| A.I | DB design | ✅ | Extended and normalized DB with defined details |
| A.II | Functional business logic | ❔ | Partially migrated functions |
| B.I | Language | ✅ | Rewritten platform for cohesive experience |
| B.II | l10n / i18n | ✅ | Implemented and translated for multi-language support |
| B.III | Interaction flow | ✅ | Implemented nested interaction routes |
| C.I | Code tests | ❔ | Auto-created user tests  Failed to implement other tests  Resulted in low test coverage |
| C.II | Documentation | ✅ | Written project documents, code documents, etc |
| C.III | Project structure | ✅ | Created project according to Java package conventions, &  Relied on framework generated structures |
| C.IV | Convention adherence | ✅ | Rewritten platform for cohesive naming and casing experience |
| 0 | Perfection | ❔ | Introduced both improvements and regressions  Arguably for the better by allowing easier future development |

# Postmortem

As stated in the previous sections, not all stated objectives and outcomes are achieved, unfortunately. To prevent such incidents from happening again, reasons and preventive measures are to be investigated.

## Overpromising & underachieving

Set out to migrate features to a new framework, I ended up short on implementing functionality of sending out e-mail invites to students and implementing distinct, well-defined GUI views for different roles (i.e. students and teachers).

The layer beneath this symptom is a lack of time, but the root-cause are multi-faceted:

1. Myself
   1. Arrogance: Thought too highly of myself in regards to the speed of learning a new framework, causing underestimation of fulfilment time needed
   2. Laziness: Was not able to dedicate more time to working instead of entertainment, causing overrun of estimated schedule
   3. Perfectionism: Scope-creeped during development, extending the DB as development went on, causing an increased fulfilment time needed
   4. Procrastination: Allowed my mind to wander when developing the system often, causing overrun of estimated schedule
   5. Naivety: Believed verbal future work plan promise on WFH policies and utilized framework, causing increased learning time needed and overrun of estimated schedule
2. Environment
   1. Unexpected toolset changes: Workplan / project changed to other domains instead of Jmix, causing increased learning time needed and overrun of estimated schedule
   2. Unexpected schedule changes: Workplace cancelled WFH policy, causing higher fatigue and decreased available time
   3. Unfamiliar environment: Started living on my own, causing a decreased available time

To avoid such a problem from arising again,

1. As suggestions to myself:
   1. Avoid over-employment / moonlighting, as my mental fortitude is weak and could not support large amount of work hours / week.  
      Work regular hours or even consider taking less pay for less work in the future.
   2. Avoid planning according to verbal promises, WFH and future work are not set in stone. Get promises in contract, and also plan contingency plans even if the contracts fall through.
   3. Avoid the underperform 🡪 “I can salvage this” 🡪 depression 🡪 underperform negative feedback loop. Sometimes things just go south, get help.
2. As suggestions to potential project contributors:
   1. Consider available time, interest in enterprise Java CRUD software, and resultant commitment-availability. Spring can be overwhelming complex to begin learning. So, decide to use familiar frameworks or interested ones accordingly.

# Conclusion

This project’s efforts are mostly for paying back the accumulated technical debt and temporarily alleviating the yearly knowledge-transferal (or the lack thereof) by establishing better conventions and patterns, which enables the project’s extendibility by allowing easier future changes.

While the framework migration was not fully realized and some regressions exist, it should be an overall improvement to the development experience and future possibilities.

The only thing to be concerned with, is the extensive level of involvement needed to learn the Jmix / Spring framework.

# Future work

As a byproduct of the framework change, any future developers can easily enable a REST API plugin which provides an alternative, totally decoupled method of enhancing the system, so even if future documentations fall short of any newcomers expectations, a system-wide grokking and re-engineering should not typically be necessary.

# Resource utilization

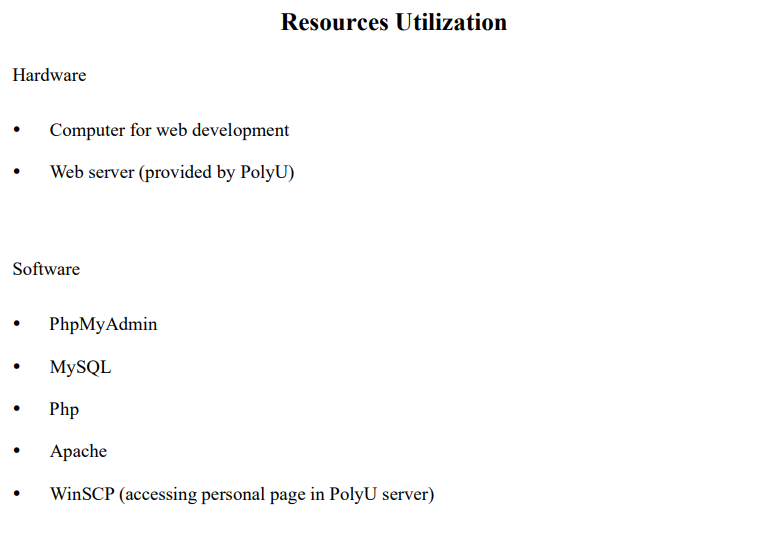
* Department managed URL
* Department allocated docker container running environment
* Department allocated MySQL DB connection
* Jmix academic license(s) (free)

# Used software

* Obsidian
* Mermaid.JS
* IntelliJ Idea
* Jmix
* Java 17
* Docker
* MySQL

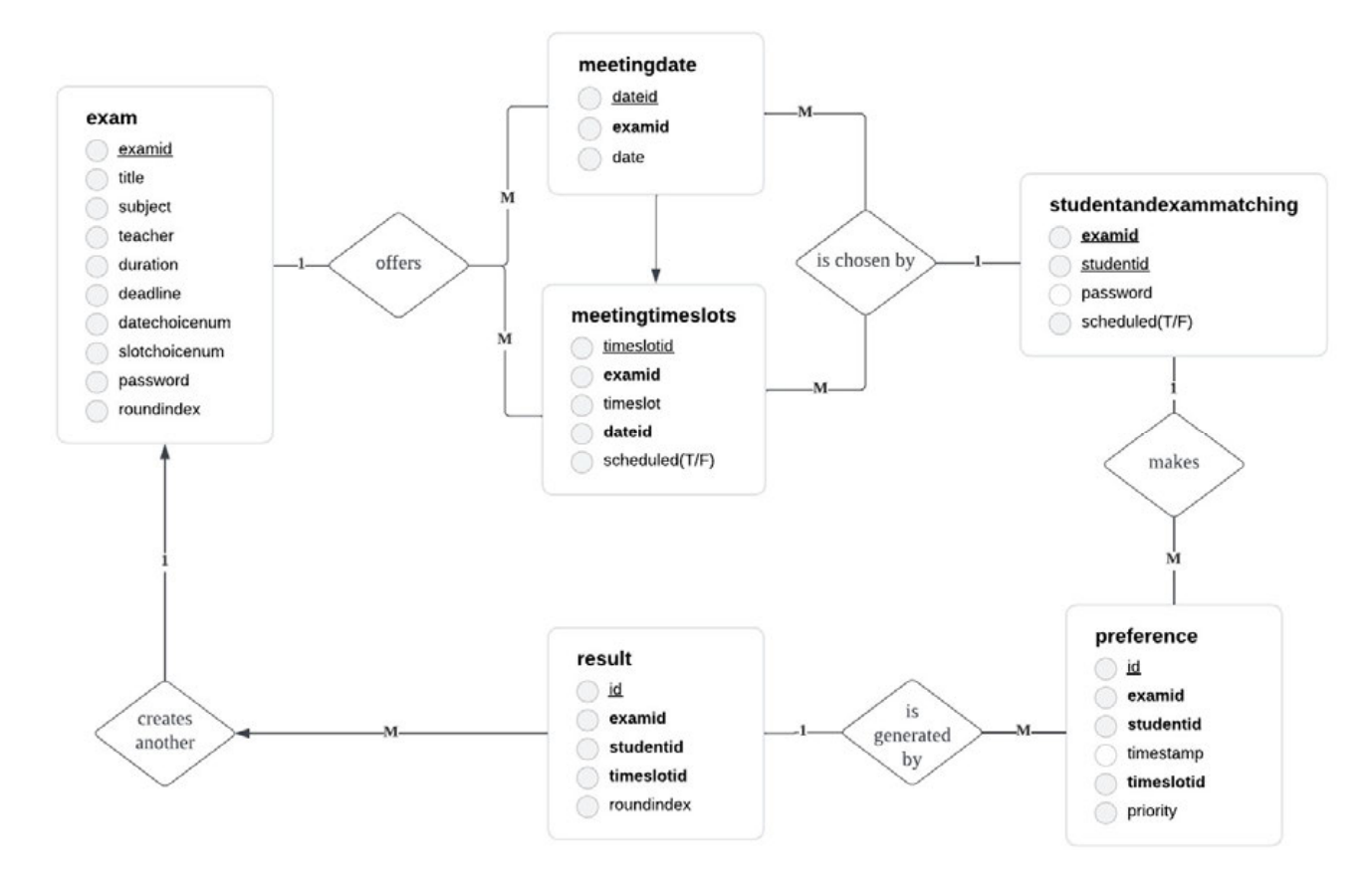
# Appendices

## Appendix A



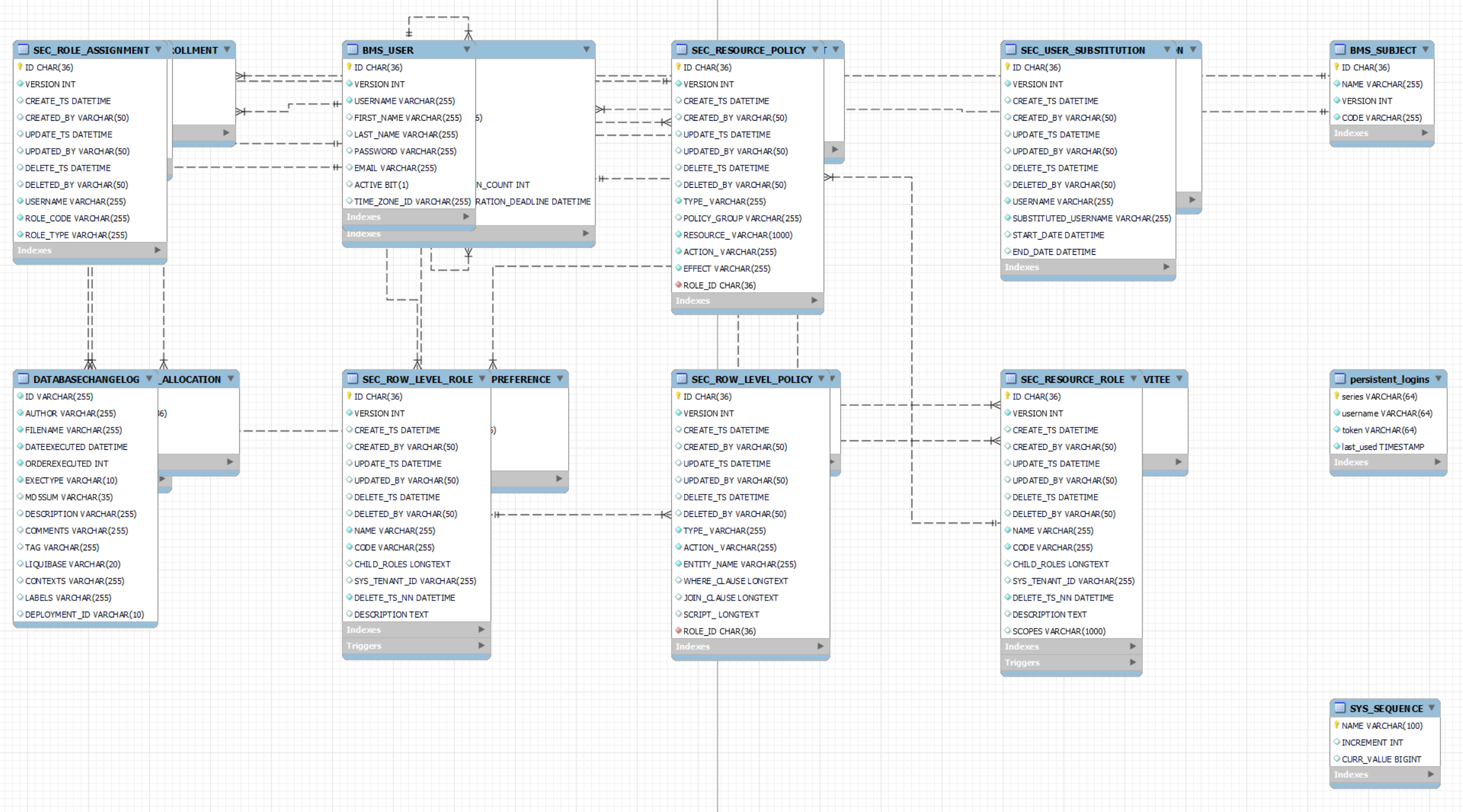
Resource utilization of senior YU Fong Yan’s project (Final report excerpt)

## Appendix B



ER diagram of Yu Fong Yan’s project’s DB schema (Final report excerpt)

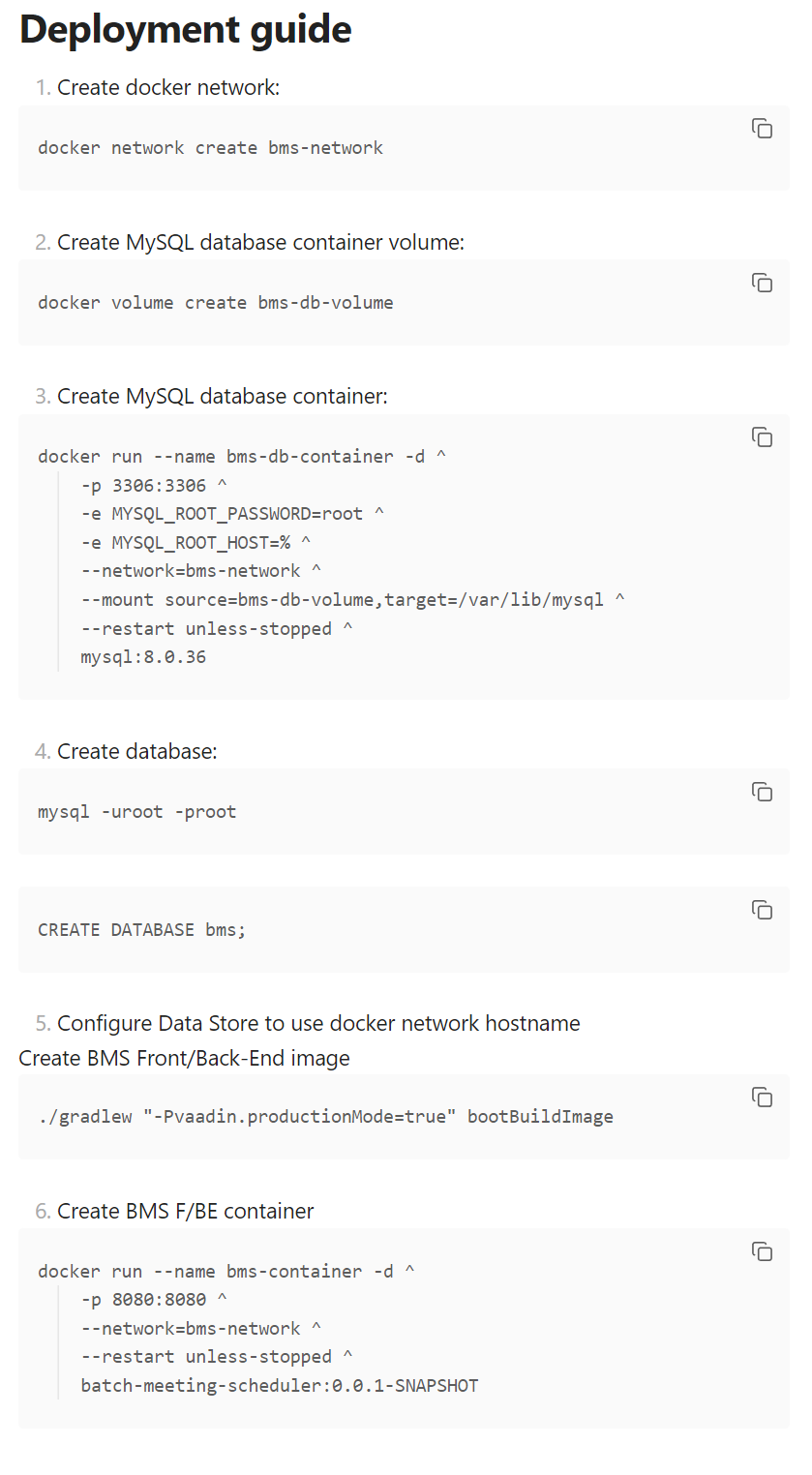
## Appendix C



ER diagram of current system’s DB schema from MySQL Workbench

Note that there are 21 tables and are visually stacked on generation

## Appendix D



Deployment guide with copy-able snippets, written in markdown, viewed in Obsidian