Warsaw University of Technology

FACULTY OF
MATHEMATICS AND INFORMATION SCIENCE

Bachelor's diploma thesis

in the field of study Computer Science and Information Systems

Platform for hybrid learning

Kiryl Volkau

student record book number 30076

Illia Manzhela

student record book number 302514

thesis supervisor

Dr Eng., Janusz Oleniacz

consultation

Professor Władysław Homenda

WARSAW 2022

Abstract

Platform for hybrid learning

The platform for hybrid learning intends to demonstrate, how the design of educational software could be done using a non-object-oriented approach alongside applying principles of cloud computing. As students, during the pandemic of 2020-2021, we have seen how the educational system was struggling to handle such a change. Meanwhile, we have also discovered the advantages of studying online - it gave us a great level of flexibility and the possibility to re-access materials (in particular). Keeping in mind, that teachers would also benefit from the re-design of the current approach to knowledge transfer, we decided to try and implement a platform, that covers the interests of both groups. We did that using Rust programming language for our backend system, F# programming language for the frontend (transpiled to JavaScript), and Azure as a main hosting solution. This paper addresses the obstacles to implementing such a platform and how it differs from already existing solutions. We used our professional knowledge as acting software engineers and students to identify and solve arising issues. It is worth adding, that we do not focus on the software development pipeline here, as it would differ vastly from the real-world development team. Nevertheless, we address usability, extendability, supportability, and other important software traits, since they are crucial to the success of the design itself.

Keywords: hydrid learning, massive open online courses, functional programming, cloud computing, education

History of changes

Table of changes		
Author	Date	Change
Kiryl Volkau	20.10.2021	add: Abstract, Introduction
Kiryl Volkau	20.10.2021	add: Functional requirements
Illia Manzhela	20.10.2021	add: Non-functional requirements
Illia Manzhela	20.10.2021	add: SWOT analysis, schedule

Contents

Vocabulary

1. **Hybrid learning** - educational process, organised to be conducted both on-line and on-site, providing students and teachers with more flexibility and instant access to the educational materials.

2. **Massive Open Online Courses (MOOC)** platform - platform with educational content including videos, text content, discussion forums.

3. **Application Programming Interface (API)** - set of definitions for building and integrating application software.

4. **Infrastracture as a Service (IaaS)** - pay-as-you-go service where a third party provides you with infrastructure services, like storage and virtualization, as you need them, via a cloud, through the internet. https://www.redhat.com/en/topics/cloud-computing/iaas-vs-paas-vs-saas

5. **Platform as a Service (PaaS)** - on-premise infrastructure management where a provider hosts the hardware and software on its own infrastructure and delivers this platform to the user as an integrated solution, solution stack, or service through an internet connection. https://www.redhat.com/en/topics/cloud-computing/iaas-vs-paas-vs-saas

Introduction

Since the beginning of the COVID-19 pandemic, many academic institutions (including Warsaw University of Technologies) were forced to provide solutions for the transfer between on-site and on-line education formats. Moreover, today's educational systems get more and more criticism worldwide due to being too generic and inflexible [2]. Meanwhile, the popularity of platforms like Coursera (https://www.coursera.org), Udemy (https://www.udemy.com), Udacity (https://www.udacity.com), etc. has been growing over the years.

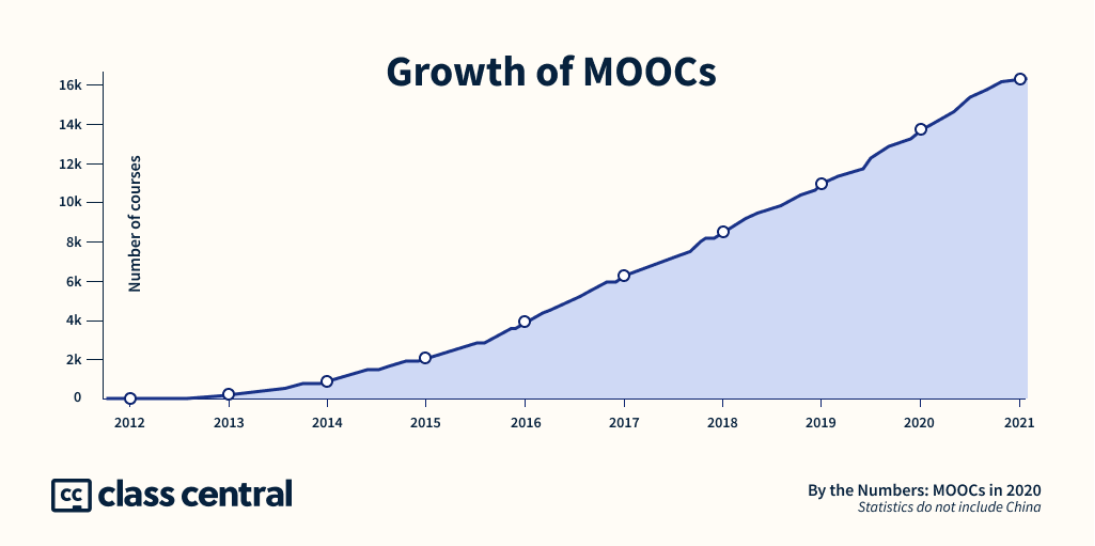

Figure 0.1: Growth of MOOC's platforms https://www.classcentral.com/report/mooc-stats-2020/[1]

At the same time, it has also been shown that the main problem with MOOCs is that students are far more likely to drop the course without completing it [3].

As the authors of this thesis believe, the solution should lie somewhere in between. Education is not our area of expertise, so we ask readers to not treat us as the authority on the topic. Nevertheless, we do believe that there will be a shift toward more flexible education, as the continuity of life grows [4] alongside the pace of living [5]. So, in this paper, we aimed at providing one of possible solutions to the operational side of the matter: platform. Although, as has been

mentioned by the experts [6], the principles of student involvement in the subject remain the same, the strategy of implementing those principles should adapt and evolve. This is why, not being completely sure about the approaches taken by future generations of researchers, we design our platform with adaptivity and evolvability in mind.
We have decided to implement our platform using a less popular technological stack: Rust [7] for the backend and F# [8] for the frontend (compiled to JavaScript using Fable [9]). Those are non-object-oriented languages greatly influenced by the functional programming paradigm. We go a little bit deeper into the topic in the following chapters, however, at the moment it is worth noticing the main strengths of the selected languages:

1. Algebraic data types - especially "sum", also known as "OR" type (discriminated union) - which allow us to express domain more exhaustively.

2. Immutability dy default - most of the software operations could be represented as a pipeline of data transformation, and without some complex state management underneath (which is object-oriented approach is famous for) it becomes easier to decouple and modularize application. Decoupling and modularization are two approaches, with which we can empower the evolvability and adaptivity of our platform.

3. Absence of null-value (in general), also known as a billion dollar mistake [10].

The hosting is done on Azure https://azure.microsoft.com/ due to its wide range of services and more clear billing, alongside some free student trials.
As for the potential competitors - there are multiple platforms with overlapping functionality / much more vast functionality (e.g. Moodle) already offered, than the one that we can deliver for a diploma thesis. Still, we do consider this project a solid basis for a future development - that's why it's code left open-source and we give specific recommendations and thoughts on extension of the existing codebase and functionality. However, we have not seen anything similar already developed or described somewhere as plans (it's worth noticing that the same functionality could be achieved using the set of multiple tools, but it just would be far less convenient).

Responsibilities and Schedule

Full Name	Responsibility
Kiryl Volkau	Frontend (Elm) & DevOps (Azure)
Illia Manzhella	Backend (Rust) & DevOps (GitHub Actions)

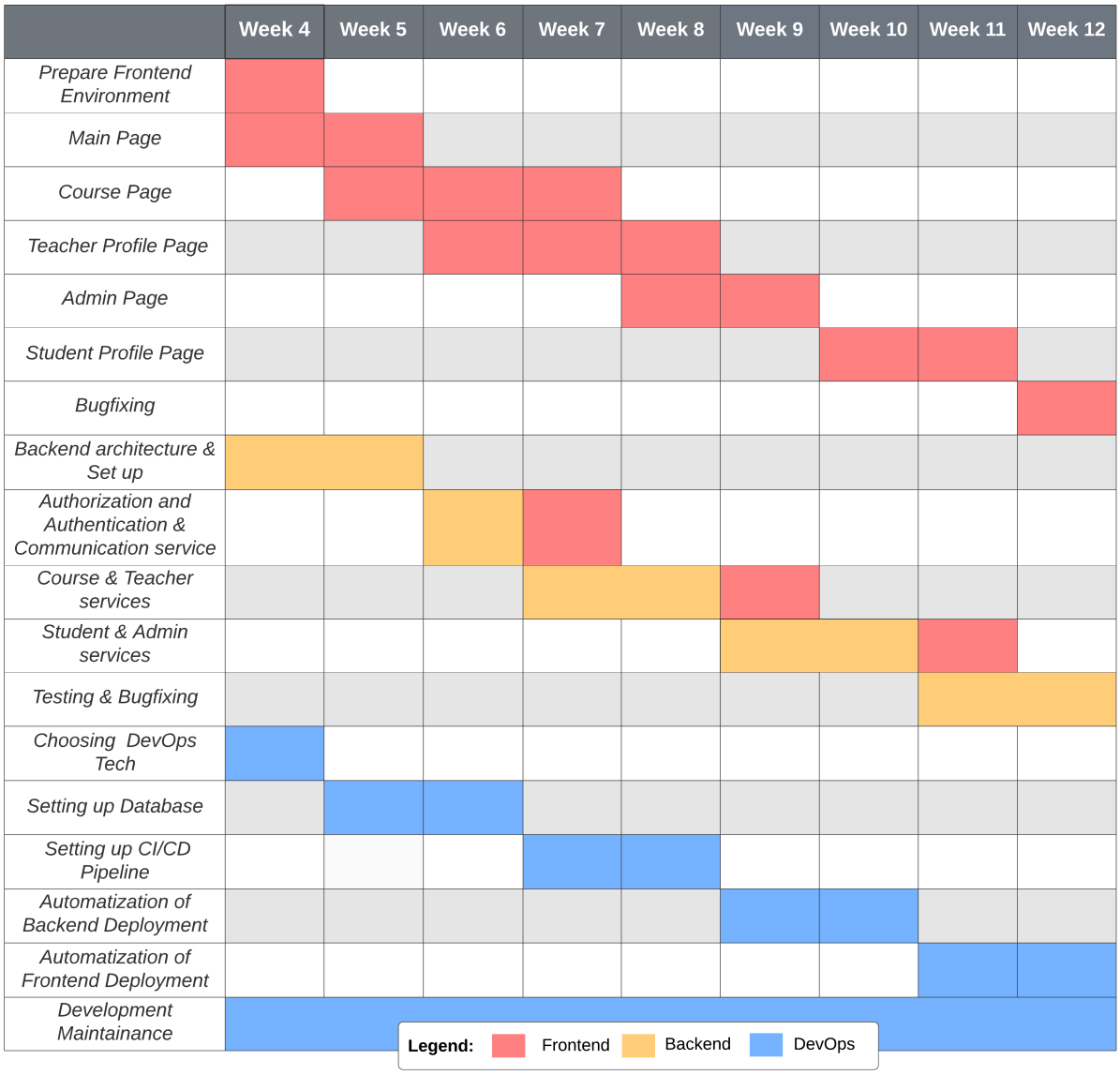

Figure 0.2: Gantt chart by task and person

Functional Requirements

We have decided to describe functionality of the platform from the perspective of users, given some specific rights and corresponding access level (roles). This allows us to represent functional requirements of the platform as a set of granular user stories and set of permission levels at the same time.

The roles are as follows:

- Everyone (combines all the groups below)
- Anonymous User (Unauthorised)
- Everyone Authorised (all the groups below)
- Teacher
- Student
- Administrator

1. **Everyone** can:
    - View universities that has been made public
    - View university courses (and its contents) that has been made public
    - Search for courses by name, university, tags
2. **Anonymous User** can :
    - Login to the existing account
    - Request the creation of the university account (administrator)
3. **Authenticated User** can:
    - Log out
    - Change personal details
    - See their University details (if they are hidden from anonymous users)
    - See their events calendar
    - Leave messages on university / courses boards
4. **Teacher** can :
    - Change own course information
    - Appoint events (exams, terms, meetings)

- Add materials to the course (videos, text, pdfs, quizes)

5. **University Administrator** can :
    - Review submitted university courses
    - Accept university submitted courses
    - Reject university submitted courses
    - Delete university courses (with possibility to recover)
    - See university students' and teachers' details
    - Make university courses open
    - Create teacher / student accounts
    - Delete teacheer / student accounts

There is also a super-admin role foreseen, which should be used for resolving software-specific issues and support existing users.

Non-Functional Requirements

1. **Usability**

   Level of basic user expertise assumed. User interface standards will be used.

2. **Reliability**

   Platform will be available 24/7 with the possibility of restarting the system in case of emergency situations (using PaaS / IaaS functionality or kubernetes). Recoverability will be maintained by an engineer who will recover system from a shut-down failure (in case of several unsuccessful attempts to restart). It will accessible from the most popular browsers like Google Chrome (and Chromium-based), Safari, Firefox, and Opera.

3. **Performance**

   System response time is supposed to not exceed 2000ms (2 seconds), automatic recovery and start-up time is expected to take up to 30 minutes.

4. **Supportability**

   System should be testable, maintainable and easily scalable for the needs of higher throughput of media streaming for large number of students. This includes possibility of refactoring the architecture to be more decoupled.

5. **Extendibility**

   System should be easy to extend - meaning it should be possible to add more features to the existing architecture without significant need for the refactoring of previously implemented system.

SWOT Analysis

STRENGTHS

1. Budget & support from the university for hosting
2. Previous experience in developing commercial software
3. Strong believe in the right cause and passion for the project
4. Milestones are well-defined and understood
5. Strong support from the Rust and F# communities

WEAKNESSES

1. Lack of experience with functional paradigm for big projects
2. Lack of expertise in the educatinal domain
3. Lack of experience in the selected technologies

OPPORTUNITIES

1. Growing interest in educational products gives more confidence in the success of the product
2. Modernization of the market of educational websites
3. Popularization of the functional programming paradigm
4. Building ground for future projects in the sphere

THREATS

1. Not enough time to complete the project
2. Absence of team members due to some reasons

Threats

1. *Not enough time to complete the project* - there is a possibility that we will not be able to complete all the functional requirements on time. As the project plan itself is quite big, we will simply cut the functionality and leave it for the future - the core goal of experimenting with the technologies in the domain should be achieved in every outcome.

2. *Absence of team members due to some reasons* - in this case the left member will evaluate their powers and the amount of work left to do and either ask for permission to continue project alone or end it for the time being.

Bibliography

[1] Dhawal Shah, By The Numbers: MOOCs in 2020, *The Report by class central*, https://www.classcentral.com/report/mooc-stats-2020/, Nov 30th, 2020

[2] A Comparative Study on Effectiveness of Online and Offline Learning in Higher Education, *International Journal of Tourism & Hospitality in Asia Pasific (IJTHAP) Vol. 4 No. 3, 102-114, October, 2021* https://www.researchgate.net/publication/355433669_A_Comparative_Study_on_Effectiveness_of_Online_and_Offline_Learning_in_Higher_Education

[3] "Why Do Students Drop Out of MOOCs?", Rachelle Peterson, National Association of Scholars, https://www.nas.org/blogs/article/why_do_students_drop_out_of_moocs, November 12, 2013

[4] World Life Expectancy 1950-2022, https://www.macrotrends.net/countries/WLD/world/life-expectancy

[5] "Pace of Life and Enjoyment of Life", February 2002, Journal of Happiness Studies 3(3):217-256 https://www.researchgate.net/publication/23545478_Pace_of_Life_and_Enjoyment_of_Life

[6] "Education Now: What Makes a High-Quality Remote or Hybrid Learning Experience?", Harvard Graduate School of Education, https://www.youtube.com/watch?v=EFA9S09ysMc&t=1276s

[7] Rust programming language, https://www.rust-lang.org

[8] F# programming language, https://fsharp.org

[9] Fable compiler (from F# to JavaScript), https://fable.io

[10] Null References: The Billion Dollar Mistake, https://www.infoq.com/presentations/Null-References-The-Billion-Dollar-Mistake-Tony-Hoare/

List of Figures