# **SCORE Contest Project**

Project : CSyllabus

Available at <https://csyllabus.com/demo/>



|  |  |
| --- | --- |
| ***Project team :*** | ***Supervisors :*** |
| **Zvonimir Relja (**zvonimir\_relja@yahoo.com) | Ivana Bosnic (ivana.bosnic@fer.hr) |
|  |  |
| **Filip Turčinović (**filip.turcinovic@fer.hr) | Raffaela Mirandola (raffaela.mirandola@polimi.it) |
|  |  |
| **Emanuel Guberovic (emanuel.guberovic@gmail.com)** | ***Customer :*** |
|  | Gruia-Catalin Roman (gcroman@unm.edu) |
| **Matej Vukosav (matej.vukosav@gmail.com)** |  |
|  |  |
| **Sebastian Mayoral (obandomayoral@gmail.com)** |  |
|  |  |
| **Adrien Roques (adrien.roques.31@outlook.fr)** |  |
|  |  |
| **Thomas Benetti (thomas.benetti@gmail.com)** |  |

# Executive summary

In today’s modern world where educational boundaries slowly disappear, transit of students from one university to another is increasing. Most of time they want to conduct an experience of studying abroad. Often, different countries include different languages or cultures, what can be challenging or even too hard to overcome. Exploring or choosing a suitable university involves adaptation to different searching tools, sites and portals. Their mechanism of showing relevant data can be confusing and not understandable at first and demand exhaustive work to get familiar with. Even if they found a similar university or one that matches their preferences, comparing courses can be challenging. Such data are not centralized and easily offered as they should be. To remove unnecessarily work and unpleasant experience before they even start, this project aims to provide all relevant information and insure that they choose best possible destination for their further education.

CSyllabus is imagined as a web platform which should ease up process of finding and comparing courses on domestic and foreign faculties. It will enable users to discover and compare courses on interactive way through web application. This « one click » application will save time and provide very useful information to interested parties.

This project is developed inside the Distributed software development course provided by Politecnico di Milano – Milan and Faculty of Electrical Engineering and Computing – Zagreb. The aim of the course is to provide knowledge on how to deal with problems related to distance and different cultures in developing software distributed environment. The team of the project is composed of three students from Milan and four from Zagreb.

CSyllabus was implemented using modern technologies (Django rest framework, Python, Angular4, Angular material, JSON, …).

Now that CSyllabus lives, we are very pleased with the outcomes as well as our stakeholders. We think that the main reason for that was good team atmosphere and coordination. Stakeholders were especially pleased with intuitive interface and design produced for this project. We always improved it, taking into account reviews from students, friends or family to make sure that the whole system is user-friendly for best user experience.

# 1 Document overview

This paper starts with the Section 2 in which the functional requirements and the nonfunctional requirements of the software are presented, additionally the process from which were founded. In Section 3 we show the relevant aspects about working in a distributed team, the different locations, the used methodology and its coordination. Following in Section 4 we present the architecture of the system and the main taken decisions around its design. Later in Section 5 all the details about the implementation are presented, both in back end and front end. Section 6 presents the process of verification and validation chosen and some specific details. Section 7 contains information about the development process. Section 8 brings the outcomes and Section 9 the future aspects. Finally, in the Section 10 we state what we have learned and summarize the whole paper.

# 2 Requirements definition

## 2.1 Gathering Process

To understand in a good way the problem and the requirements, they were founded using an exhaustive and iterative process in which we were in contact with the stakeholders. Particularly we used the next canals:

* Meetings with Erasmus Coordinators: Erasmus (***European******Region******Action******Scheme for the******Mobility of******University******Students*** ) is an organization for students were they go to interchanges to multiple countries. We had meetings with the coordinators in the universities: Politecnico di Milano and FER
* Meetings with Politecnico and FER IT departments: It was useful to complement some requirements about the management of the data
* Meetings with customer: Our customer Catalin Roman was in constant contact with us using videocalls. Particularly we had 5 calls.
* Questionnaires to potential users (Students): Using questionnaires we proved prototypes but also we included open question in which the students could suggest new features or improvements to Csyllabus.

## 2.2 Domain

In today’s modern world where educational boundaries slowly disappear transit of students from one university to another is increasing. Most of time they want to conduct an experience of studying abroad. Often different county includes different language or culture what can be challenging or even too hard to overcome. Exploring or choosing suitable faculties involves adaptation to different searching tools, sites and portals. Their mechanism of showing relevant data can be confusing and not understandable at first and demand exhaustive work to get familiar with. Even if they found similar faculty to their own or one that matches their preferences comparing available courses can be challenging. Such data are not centralized and easily offered as they should be. To remove unnecessarily work and unpleasant experience before they even start this project aim to enable and provide all relevant information’s and insure that they choose best possible destination for their further education.

For analyzing the domain of the problem is necessary to see the different perspectives of the involved possible users:

* Students are interested in information’s about other universities and their study plans. Students want to go to the best university that suits their needs to do so, they need to get information’s about other faculties and their courses. Because foreign faculties are not often well known, and courses can have different names they will have opportunity to compare their familiar courses with available.
* Professors are in constant research in the academic world. A good professor is updated about the new ways to teach others. However, often he lacks quality information’s and good feedback on his work. Professors can have possibility to see other feedbacks on their work in a view of comments and votes.

# 2.3 Functional requirements.

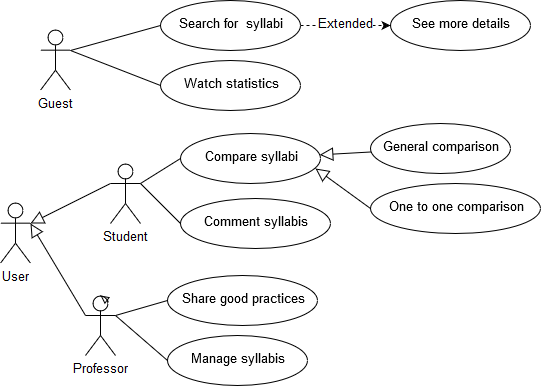
First of all, it is precise defining the actors that will use the application. It is important to clarify that Professor and Student are specializations of User actor. In the rest of the document both will be called as User. However here there are the different intentions they have to use Csyllabus:

* Guest: is a person that belongs to the academic community who does not have an account registered in Csyllabus system. Guest is user who has high level of overview on the main functionalities of the system but has no user-identification-required features available such as user profile or can build user engagement.
* User -> Student: is the registered user with verified Csyllabus account. He has all features as a guest plus user-identification-required features such as user profile and possibility to leave comments or vote for course they are familiar with.
* User -> Professor: is the registered user with verified [Subject] account. He has all features as a guest plus user-identification-required features such as user profile except possibility to leave comments or vote for course they are familiar with. Professors can learn from comments and use them to improve their personal skills. Professors can have option to upload new syllabus.

To define the functional requirements of the system we have used the artifact *User Stories* from the methodology SCRUM which will be explained later. Basically, they express the whishes of the actors.

|  |  |  |
| --- | --- | --- |
| **ID** | **Name** | **User Story** |
| US1 | Syllabus search | As a student I want to search for syllabi by name, faculty or country. |
| US2 | Syllabus details | As a student I want to see syllabi details, so I can see what it contains. |
| US3 | Course details | As a student I want to see details of course so I can see what they offer |
| US4 | Faculty details | As a student I want to see details of faculty, so I can know more about faculty. |
| US6 | Country and faculty choose | As a student I want to choose country and faculty I am studying at, so I can choose my own courses for comparison |
| US8 | Similar faculty | As a student I want to see which faculties are most similar to my own, so I can see what is my best option |
| US9 | Relevant subjects | As a student I want to see other subjects that are relevant for me |
| US10 | Destination country and faculty | As a student I want to choose destination country and faculty, so I can compare my courses with them. |
| US11 | Recommended courses | As a student I want to see recommended courses, so I can maybe come to know something new |
| US12 | Course comments | As a student I want to comment and evaluate a course |
| US13 | Syllabus share | As a student I want to share a course in social media so others can see it |
| US14 | Add syllabus to database | As a Professor I want to have option to add new syllabi to the database so that database can expand |
| US18 | Guest view | As a Guest I want to see the main functionalities without an account |

According SCRUM during the development process the functional requirements to be satisfied are the user stories. However, starting from the user stories for understanding better it was important to translate them into use cases because they are more concrete. Uses cases are exposed in the next diagram



To carry out a good planning from these user stories and uses cases, the main functionalities of the system are summarized three main features:

* *Explorer:* The possibility of searching for syllabi using name, country, university and watch all their details.
* *Comparator:* The possibility of comparing some home university syllabi with another destination university syllabi. Watching the results according a ranking and suggest to user the best ones.
* *Social part:* The possibility of commenting the syllabi, sharing in social networks and of creating a community around the tool.
* *Management:* The possibility of managing the data, for example adding new syllabi or modifying them.

## 2.4 Non functional requirements

## Usability

CSyllabus is a web-based application and responsive for mobile usage. In this way, the User can use the application with different browsers from various locations.

The application must offer a user-friendly interface.

The application must be easy to use and understandable. It must not require specific knowledge on new technologies.

The application must not require more than one hour of training to master its functionalities.

## Availability

System will be available through web page so users can use its functionalities. Interaction with the system data will be available through system API.

## Privacy and Security

The data used in this project will be publicly visible to all consumers except in situations where data owner insist otherwise. The personal data of the user will be publicly visible in user profile except when user manually hides information. Syllabuses will be available only with the authorization of the respective faculties or universities.

## Performance

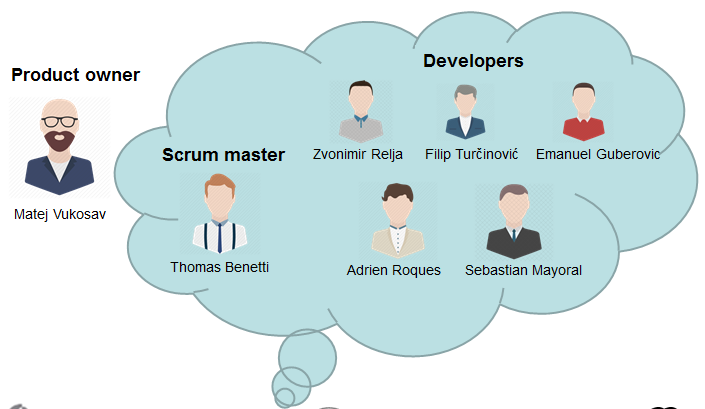
The system will be scaled according to user acquisition. Architecture will be designed taking in view later easier expanding and scalability. Since this application will be created in the context of the DSD course, our team will not build or require any dedicated infrastructure for it. Futhermore, it is impossible to estimate and prove the exact value for performance tests in this scope.

# 5 SCRUM management

## 5.1 Team introduction

Our team consists of four students from University of Zagreb, Croatia, and three international students from Politecnico di Milano, Milan, Italy. We are all students of computer engineering with no previous knowledge about distributed software development, so this was a great chance for us not only to learn something new but also experience full concept of distributed development.

Team members and roles are as follow :



*Figure 1. Roles repartition*

We used the SCRUM approach to define team roles, as following:

* The Product Owner, who is responsible for the product backlog and facilitate the communication between the customer and the development team
* The Scrum Master, who helps the team during all the SCRUM phases and facilitates the ceremonies
* The Development Team, who builds the increment product. The Product Owner and the Scrum Master works with the team.

We used Doodle to choose the project roles. The team has decided then that Matej Vukosav will be the Product Owner and that Thomas Benetti will be the Scrum Master. We chose to don’t change the Scrum master role if the person responsible has the expertise and does his job well.

## 5.2 Team expertise

From the total of seven members in our team, three already had advanced knowledge of web development with the frameworks we used (Django and Angular4), two members had some experience of web page development and two members had only theorical knowledge and expertise in other areas like object design and database modeling. All of this had to be taken into consideration when dividing roles among the team.

## 5.3 Team coordination

## 5.4 Programming management

On the very beginning, our team in overall was rather inexperienced in actual web programming as we had only three of seven programmers with advanced knowledge in all the project technologies, so there was a time needed for others to educate themselves into specific programming languages and techniques.

During this period, our three main programmers put a considerable and focused effort to come with an implementation on system, to install the frameworks and start the project, both on back-end and front-end as this was the most difficult programming task.

Second, which we have predicted, frameworks used were rather difficult to understand for the other four programmers. To overcome this situation, there was an additional need of educational period which required active collaboration of main programmers with others. This education was found rather difficult to accomplish in distributed environment as document itself was not always sufficient. The implementation of the whole application was done accurately, so we found easy to work with that after we got familiar with it. An advantage was that we focused on a component-based design with clear specificatinos all that helped our distributed work, saving efforts to put together components.

## 5.5 Communication

A good communication is vital part of any distributed project, especially ours because of our SCRUM development we always needed to be synchronized with our work. There were strict rules on reports of each team member as we all needed to fill during each sprint the KanbanFlow board used for project management but also Toggl page to see reports regarding our work and future work as well. Once all documents were completed at the end of the sprint, we summarized it in one Sprint report. Based on that report, initial project plan and on requirements possible evolution, we had telephone or video conferences on Sunday regarding work to be done for the next sprint.

Each action and system coding was followed by documentation… **(TO FINISH)**

## 5.6 Managing project artifacts

All the project files are located on our Git server to which each member has access. The files can also be seen on our GitHub page <http://www.github.com/csyllabus> by anyone. Every one of us had also our local checkout on which we developed new parts of system. After each unit or feature is developed, the code is sent to main server and merged with others code if necessary. Strict rules on coding conventions were also posted on group page which was our main document repository.

## 5.7 Integration and tracking

By the end of each sprint and in respect to the milestones, integration of components developed through that sprint took place online on our main server (<http://www.csyllabus.com/demo)> which is hosted by Emanuel. That way, we made sure that all parts worked not only on our local environment but in production as well. We used this website as our primary demonstration point for our client but also for our potential users (international students) to gather their opinions and ideas.

Also, we used this site to make clear for our supervisors that the project is on track, as they were able in every point of the project development to check project status and implemented features we stated in our reports.

# 6 Project plan

As CSyllabus is composed by different modules and developed by several people, we made a project plan in the first week to have good organization both in timing and tasks regarding requirements. Following our knowledge in software engineering, we created milestones for each phase and we divided the project in back-end and front-end side with sub-categories.

## 6.1 Planned sprints

|  |  |  |  |
| --- | --- | --- | --- |
| **Sprint** | **Start date** | **End date** | **Description** |
| 0 | 2017-10-13 | 2017-10-27 | Research |
| 1 | 2017-10-27 | 2017-11-10 | Explorer and database |
| 2 | 2017-11-10 | 2017-11-24 | Comparator |
| 3 | 2017-11-24 | 2017-12-08 | Community |
| 4 | 2017-12-08 | 2017-12-22 | Validation |
| 5 | 2017-01-05 | 2017-01-19 | Final report |

## 6.2 Milestones

|  |  |  |  |
| --- | --- | --- | --- |
| **Item** | **Type** | **Planned end date** | **Revised end date** |
| Project vision and project plan | Presentation | 2017-10-17 |  |
| Requirements, design and revised project plan | Presentation | 2017-10-31 |  |
| Project plan document (v1) | Deliverable | 2017-11-03 |  |
| Requirements definition document (v1) | Deliverable | 2017-11-10 |  |
| Design description document (v1) | Deliverable | 2017-11-10 |  |
| Status report | Presentation | 2017-11-14 |  |
| Milestone – Alpha prototype | Presentation | 2017-11-28 |  |
| Milestone – Beta prototype | Presentation | 2017-12-12 |  |
| Acceptance test plan | Deliverable | 2017-12-22 |  |
| Final project | Presentation | 2018-01-09 |  |
| SCORE report | Deliverable | 2018-01-15 |  |
| Test report | Deliverable | 2018-01-19 |  |
| Final project report | Deliverable | 2018-01-19 |  |
| Final product | Deliverable | 2018-01-19 |  |

## 6.3 Dividing tasks

As shown in *Figure 2*, we had different tasks to work on. Some of them were required to be completed before starting others, others needed to be done in parallel. We tried to imagine how the tasks flow was and how long each task would take.

*Figure 2. The Gantt chart created at the beginning of the project*

*Figure 3. The actual Gantt chart*

# 7 Architecture and design

## 7.1 Motivation and overview

## 7.2 Database

## 7.3 Back-end side modules

## 7.4 Front-end side modules

# 8 Implementation

## 8.1 Back-end implementation

## 8.2 Front-end implementation

## 8.3 Outcome

# 9 Verification and validation

## 9.1 Project policies

## 9.2 Unit tests on front-end

Unit tests on the frontend are executed according to the code convention defined section 9.1. And after all unit tests are executed, a test document is generated to show the quality of coverage of our CSyllabus product. Following test approaches are used in test execution:

Developers are responsible for unit test as white-box testing[[1]](#footnote-1). The implementation of each module and individual component are verified separately. Unit tests have been

used in web application to test correctness of user interface automatically. There were one hundred and ten unit tests.

## 9.3 Unit tests on back-end

## 9.4 Security and ping time

## 9.5 Validation of beta prototype

# 10 Development process

# 11 Outcomes

## 11.1 Implemented modules

## 11.2 The user interface

## 11.3 Users response to launch

## 11.4 Other outcomes

## 11.5 Our experiences and lessons learned

# 12 Summary

# 13 References

1. *https://en.wikipedia.org/wiki/White-box\_testing* [↑](#footnote-ref-1)