Czech University of Life Sciences

Faculty of Economics and Management

Department of Information Technologies

Diploma Thesis

Analysis and design of cloud based information portal for Erasmus Student Network

Bc. Vít Bareš

Supervisor: Ing. Miloš Ulman, Ph.D.

CZECH UNIVERSITY OF LIFE SCIENCES PRAGUE

Faculty of Economics and Management

DIPLOMA THESIS ASSIGNMENT

Vít Bareš

Informatics

Thesis title

Analysis and design of cloud based information portal for Erasmus Student Network

---

Objectives of thesis

The main goal of the thesis is to analyse and design information system for ESN (Erasmus Student Network) student organisation.

The partial goals are such as:
- to make overview of the current market and applications for the purpose of student organisations,
- to deliver an analysis of cloud solutions and deployment strategies and need analysis of a particular student organisation,
- to design and evaluate the proposed information system for ESN student organisation.

Methodology

Thesis will be divided into theoretical and practical section. Theoretical section will include research about cloud computing and other existing ESN information system solutions.

Practical section will use methodology of Systems Development Life Cycle (SDLC) to cover analysis and design of the proposed system. Evaluation and comparison will be done by appropriate methods, e.g. multicriteria analysis of variants. Based on the results of the theoretical part and practical solution, final conclusions will be formulated.

The proposed extent of the thesis

60 – 80 pages

Keywords

Cloud computing, ESN, Information System, Buddy Matching System

---

Recommended information sources

EELES, Peter a Peter CRIPPS. Architektura softwaru. Vyd. 1. Brno: Computer Press, 2011, 328 s. ISBN 978-80-251-3036-0.

CHACON, Scott a Ben STRAUB. Pro Git: The expert's voice. 2, revised. Apress, 2014. ISBN 1484200764, 9781484200766.

SNIA: Cloud Data Management Interface. 2014. Dostupné z: http://www.snia.org/sites/default/files/CDMI _ Spec _ v1.1.pdf

Velte, A.; Velte, T.; Elsenpeter, R.: Cloud Computing: praktický průvodce. Computer Press, 2011, ISBN 9788025133330

WIEGERS, Karl Eugene. Požadavky na software. Vyd. 1. Brno: Computer Press, 2008, 448 s. ISBN 978-80-251-1877-1

Expected date of thesis defence

2016/17 WS – FEM

The Diploma Thesis Supervisor

Ing. Miloš Ulman, Ph.D.

Supervising department

Department of Information Technologies

Electronic approval: 18. 10. 2016

Ing. Jiří Vaněk, Ph.D.

Head of department

Electronic approval: 24. 10. 2016

Ing. Martin Pelikán, Ph.D.

Dean

Prague on 28. 11. 2016

Declaration

I declare that I have worked on my Master's thesis titled "Analysis and design of cloud based information portal for Erasmus Student Network" by myself and I have used only the sources mentioned at the end of the thesis. As the author of the diploma thesis, I declare that the thesis does not break copyrights of any third person.

In Prague on 30th November 2016 .....

Acknowledgement

First and foremost, I want to thank my supervisor Ing. Miloš Ulman Ph.D. for his constant support and guidance throughout the process of writing the diploma thesis.

A special thanks to all the members of ESN CZ and ESN CULS Prague, for the support and suggestions regarding the analysis and design of ESN information system. Without them the work would have no value.

Analýza a design cloudového informačního portálu pro Erasmus Student Network

Souhrn

Hlavním tématem této diplomové práce je návrh cloudového řešení pro Informační systém Erasmus Student Network (ESN).

Teoretická část obsahuje přehled současných modelů cloud computingu, analýzu potřeb ESN studentské organizace, definici systému pro párování zahraničních a českých studentů, popis existujících řešení a analýzu platformy, na které by bylo možné provozovat webovou aplikaci - Informační systém ESN.

Praktická část využívá analýz z části teoretické. Na jejich základě navrhuje Front-end design, diagramy případů užití, digram datových tříd a platformu vytvořenou na míru pro potřeby provozu aplikace formou víceklientské architektury SaaS.

Klíčová slova: Cloud computing, Návrh architektury, ESN, Víceklientská architektura, PHP, Framework, SDLC, Buddy párování

Analysis and design of cloud based information portal for Erasmus Student Network

Summary

The main topic of this diploma thesis is the design of cloud based information system for Erasmus Student Network (ESN).

Theoretical part provides an overview of current cloud computing models, analysis of needs for ESN student association, defines what is buddy matching system, describes existing naive solutions and provides analysis of the best possible platform for web application - ESN information system.

Practical part is based on the analysis provided by part theoretical. It contains Front-end design, Use Case diagrams, Data class model and the tailored platform for deployment in the form of multi-tenant SaaS architecture.

Keywords: Cloud computing, Architecture design, ESN, Multi-tenant architecture, PHP, Framework, SDLC, Buddy matching

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since Erasmus programme was established in 1987, tens of thousands of students traveled all the countries in European Union. To help the international students to integrated, meet new friends and feel as home in a new environment and different culture, a group of volunteers established Erasmus Student Network (ESN). Soon the buddy programme become a core activity of ESN. Buddy programme's goal is to connect the incoming international students with local students (buddies), who can be the guides in new environment and most importantly new buddies, friends.

Now in 21st century new possibilities emerged to radically improve the buddy programme and overall performance of ESN. ESN has always been a volunteering organisation, where resources are usually scarce. It is definitely not a business organisation, but rather group of friends, who are doing what they like in their free time. To save the free time an idea of new information system emerged. This system would automate the repetitive tasks and safe the time for the ESN members to interact with the international students and organise interesting events for them.

After quick research of the existing solutions it was clear, that there is no user friendly and intuitive solution, which would meet the requirements. This thesis is the first step in development of the Infromation system, which will become vital tool in managing buddy programme and other activities of ESN student organisation.

Chapter 2

Objectives and Methodology

This chapter specifies objectives and goals which are about to be achieved and methodology used to achieve those objectives.

2.1 Objectives of thesis

The main goal of this thesis is to design information system for ESN (Erasmus Student Network) student organisation. This system will provide functions similar to CRM (Customer Relationship Management), HRM (Human Resources Management) and specific functions like buddy matching system for arriving International students (Erasmus, Exchange and other programmes). Part of the main goal is to propose how to deploy the information system as cloud service in form of SaaS (Software as a Service) and therefore make it easily accessible to ESN sections.

The partial goals are:

- to make overview of the current market and applications for the purpose of student organisations,
- to deliver an analysis of cloud solutions and deployment strategies and need analysis of a particular student organisation,
- to design and evaluate the proposed information system for ESN student organisation.

2.2 Methodology

Thesis will be divided into theoretical and practical section. Theoretical section will include research about cloud computing and other existing ESN

information system solutions.

Practical section will use methodology of Systems Development Life Cycle (SDLC), to ensure effective development of system SaaS in cloud. Based on the results of the theoretical part and practical solution, final conclusions will be formulated.

2.2.1 SDLC - System Development Life Cycle

The methodology of this thesis is based on the SDLC. As Kay mentions in his article about System Development Life Cycle[18] SDLC consists of multiple stages, which *"can be characterized and divided in different ways, including following:"*

- Project planning, feasibility study.
- Systems analysis.
- Systems design.
- Implementation.
- Integration and testing.
- Acceptance, installation, deployment.
- Maintenance.

To achieve objectives set in section 2.1 this thesis focus mainly on two of these stages: Systems analysis and design. Because ESN information systems is intended to provide cloud service as a software across the European countries, the legal aspects of this model are discussed in section 3.3

Chapter 3

Theoretical Part

This chapter focuses on research of current technologies, introduction of customer, cloud computing research and business analysis. All the information from this chapter is part of theoretical study, which is the base for the Practical part - the design of information system.

3.1 Erasmus Student Network

This section contains brief introduction into history of ESN CULS Prague and ESN, the largest European student organisation, to better understand the importance of this organisation and the impact of the newly created information system it will have on ESN CULS Prague.

"Erasmus Student Network (ESN) is the biggest student association in Europe. It was born on the 16th October 1989 and legally registered in 1990 for supporting and developing student exchange." [12]

ESN has 3 levels of operations:

- International;
- national;
- local.

3.1.1 ESN CULS Prague

ESN CZ[1] is part of the national level since 2001 and currently has 18 sections on the local level. One of this section is ESN CULS Prague. I am

[1] http://esn-cz.cz

one of the founding members of this section and during past 2 years I have experienced many situations, which could have been solved by an information system. ESN CULS Prague is the template section for modelling the business processes and analysing the needs in order to design meaningful system, which will help ESN members to manage their buddy system and other tasks related to the mission statement "Students helping students".

Every semester there is more than 300 International students coming to the Czech University of Life Sciences in Prague. In 2014 a group of local students decided to establish a student organisation, which would take care of those students. At first the student organisation was called #BUDDY_GO. Everything started with organising multiple trips and events. However in the next year #BUDDY_GO joined International European organisation ESN and changed it's name to Erasmus Student Network Czech University of Life Sciences in Prague (ESN CULS Prague) and become officially registered voluntary association.

After this important step ESN CULS decided to take care of the buddy program in CULS and the new information system would become a tool to increase the efficiency of this system and to raise the prestige of university mobility program.

3.1.2 General Organisational Structure

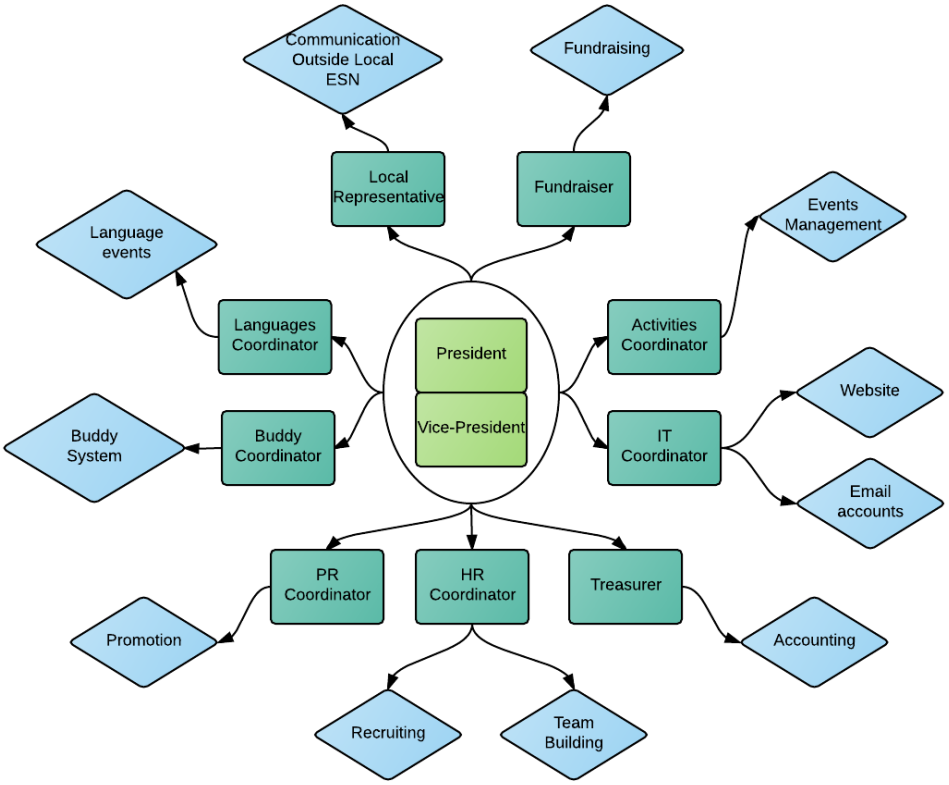
In order to obtain the right insight of the problematic from the right sources, it is necessary to understand who is responsible for which tasks in ESN CULS Prague. The organisational structure among different ESN sections might vary, but not significantly. The design of the information system will start at the local section ESN CULS Prague and later will be generalised to fit to other ESN sections in the network.

Every section has a Board. Board is the main management organ which is elected and during the mandate it organises all the operations. Positions in the board of ESN CULS Prague and the main responsibility of that position is shown in figure 3.1. Positions Buddy Coordinator and Languages Coordinator (figure 3.1) are not oficcialy part of the board according to the statues of ESN CULS Prague [2]. However both play vital role in functioning of the student organisation and they are important for the purpose of analysis the needs in ENS CULS Prague.

The most important input for the business process analysis is from the Buddy Coordinator. He is the responsible person for pairing, connecting and managing the connections between incoming international students

[2] https://or.justice.cz/ias/content/download?id=b0d503b0365c47a0b275c07ccbca6bb1

Figure 3.1: **Organisational chart of ESN Culs Prague Board; with main responsibilities**



Source: own drawing

and local students (buddies). Secondly the input from human resources coordinator, who is responsible for recruiting new buddies (local students) will provide useful insight into the problematic of managing local students. In many sections HR Coordinator and Buddy Coordinator positions are represented by one person. Therefore the actor use case analysis in table 4.1 recognizes only actor Buddy Coordinator, who is the manager for both local and international students and incorporates the scope of HR coordinator.

3.2 Business process analysis

One of the partial goals is to analyse the needs of ESN. The template section for business analysis is ESN CULS Prague. First step is to identify the core

business processes, which could be automated by the information system.

3.2.1 Buddy Program

Buddy program is certainly the core business process of almost every ESN section. Through buddy program the buddies and international students are connected. Without the information system, most of the communication with buddies and incoming students is done through email conversation. In figure 3.2 is the process of registering and pairing buddies and incoming students via email communication. From the diagram is evident, that Buddy Coordinator is the center of the buddy program and the success of the program depends on his or her performance as shown in figure 3.3. If the Buddy Coordinator for some reason becomes temporarily unavailable the buddy program is endangered. In this case some other member has to take over the communication which can be problematic in many cases.

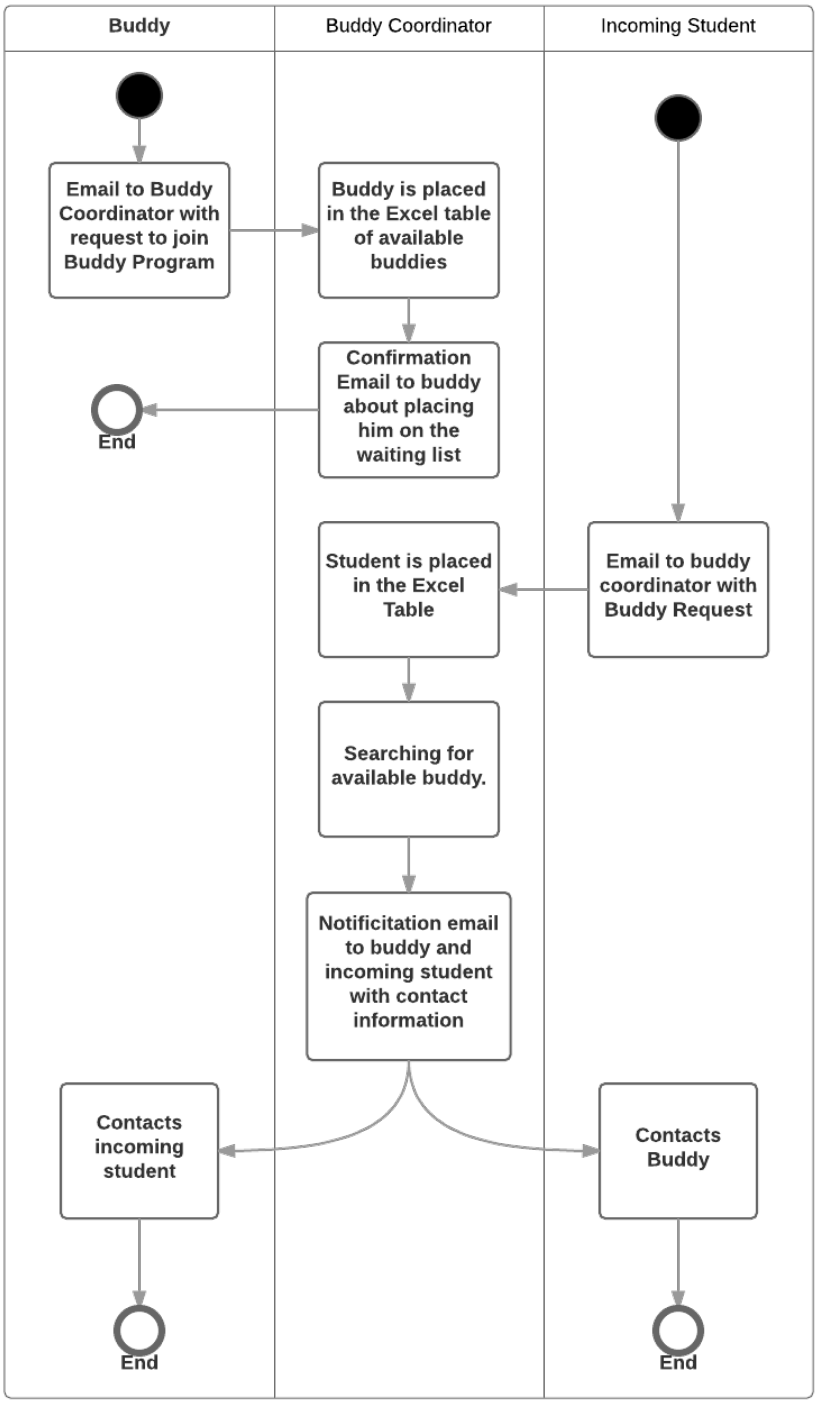
3.2.2 Event Registrations

Event registration is one of the tasks, which can become repetitive for the international students. The business process of event registration is shown in figure 3.4. International students come to the office and to register for an event. Many events are limited by capacity (for example the bus has maximum capacity of 52 people). Without the information system the student has no way how to obtain information about the numbers of already registered students. Only after coming to the office the student asks the office manager if the event is full or not. In case it is full, student either spend his free time by going to the office with no result or he has to choose another event, if more events are available for registration.

If the event is available, student fills in the registration form. The name and all the contact information and then proceeds with payment. Office manager then has to create a list of participants and manually manage this list. It can happen, that the list get lost and suddenly there is no information about how many people is registered and who is it, which is very bad situation and it can damage the reputation of ESN.

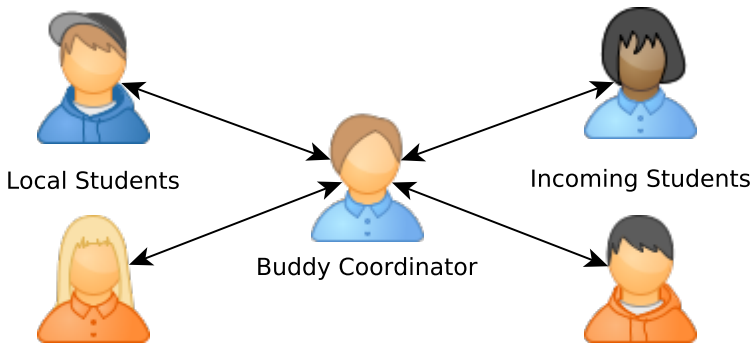
To address this issue, ESN information system should implement a way how to register students for events. It will eliminate all the problems mentioned above: list cannot be lost; students can check the capacity online; students does not have to fill in the registration form - only the first time. If they come to register again, their information is already saved in the system.

Figure 3.2: **Email Buddy System Business Process Diagram**



Source: own drawing

Figure 3.3: **Without information system is Buddy Coordinator the core element of the whole Buddy program.**



Source: own drawing

3.2.3 Proposed Solution

The proposed solution of information system is based on the idea of automation of the repetitive tasks which are usually manually performed by the Buddy Coordinator. The proposed business process after implementing information system for Buddy program is displayed in figure 3.5

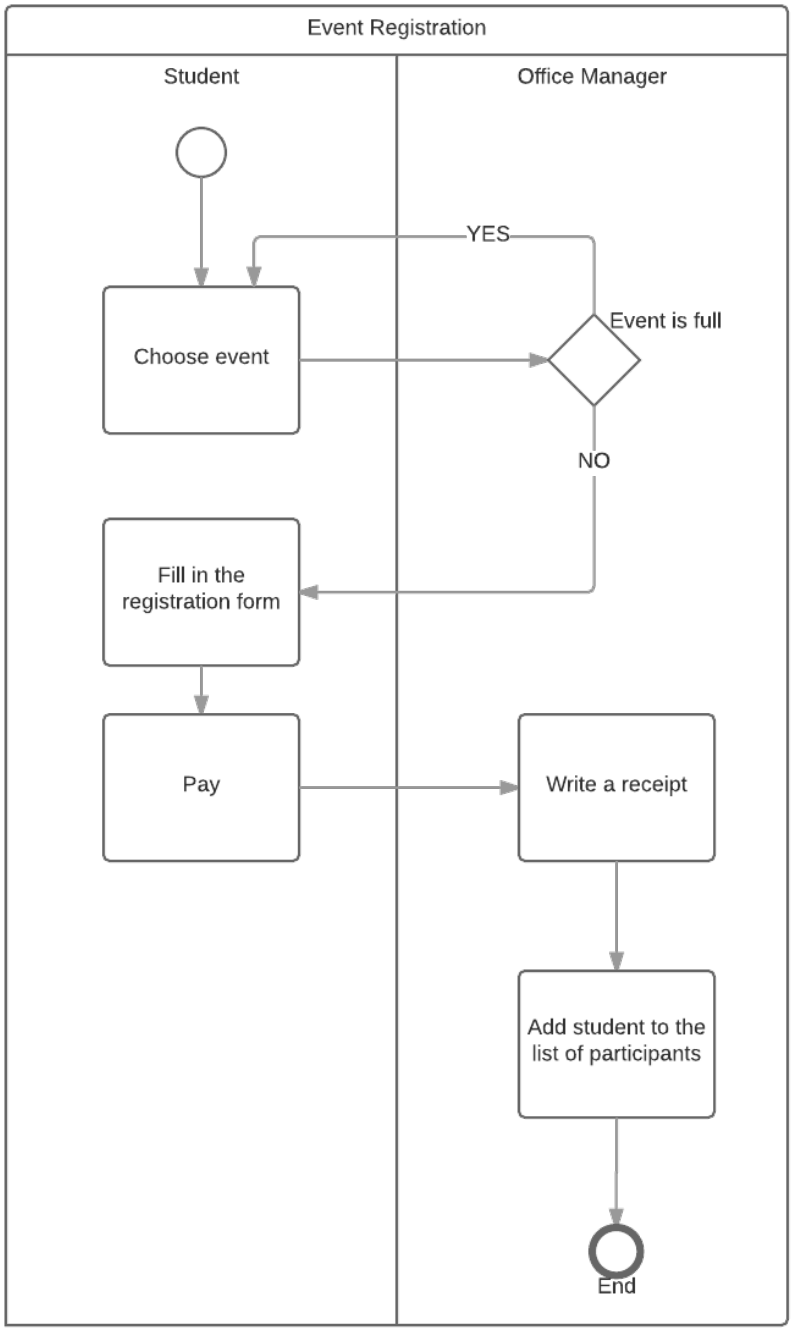
It is evident, that for the common process of pairing is Buddy Coordinator fully replaced by the information system. However there are still certain scenarios, which requires a human reaction and decision making. For example in case, where buddy is suddenly unavailable to pick up student he has to contact the buddy coordinator, to assign his international student to someone else. The situation after implementing information system is shown in figure 3.6

3.3 Legal Feasibility

After presenting idea of having Cloud Service for buddy prorgam for all ESN sections during ESN meeting 2015 in Varvažov[3], many of the delegates raised a question about privacy policies and issues with personal data. To answer those questions, ESN CULS Prague decided to provide finances for Legal feasibility study, which was conducted by the specialist in international law JUDr. Jan Bárta. The output of this study is legal document

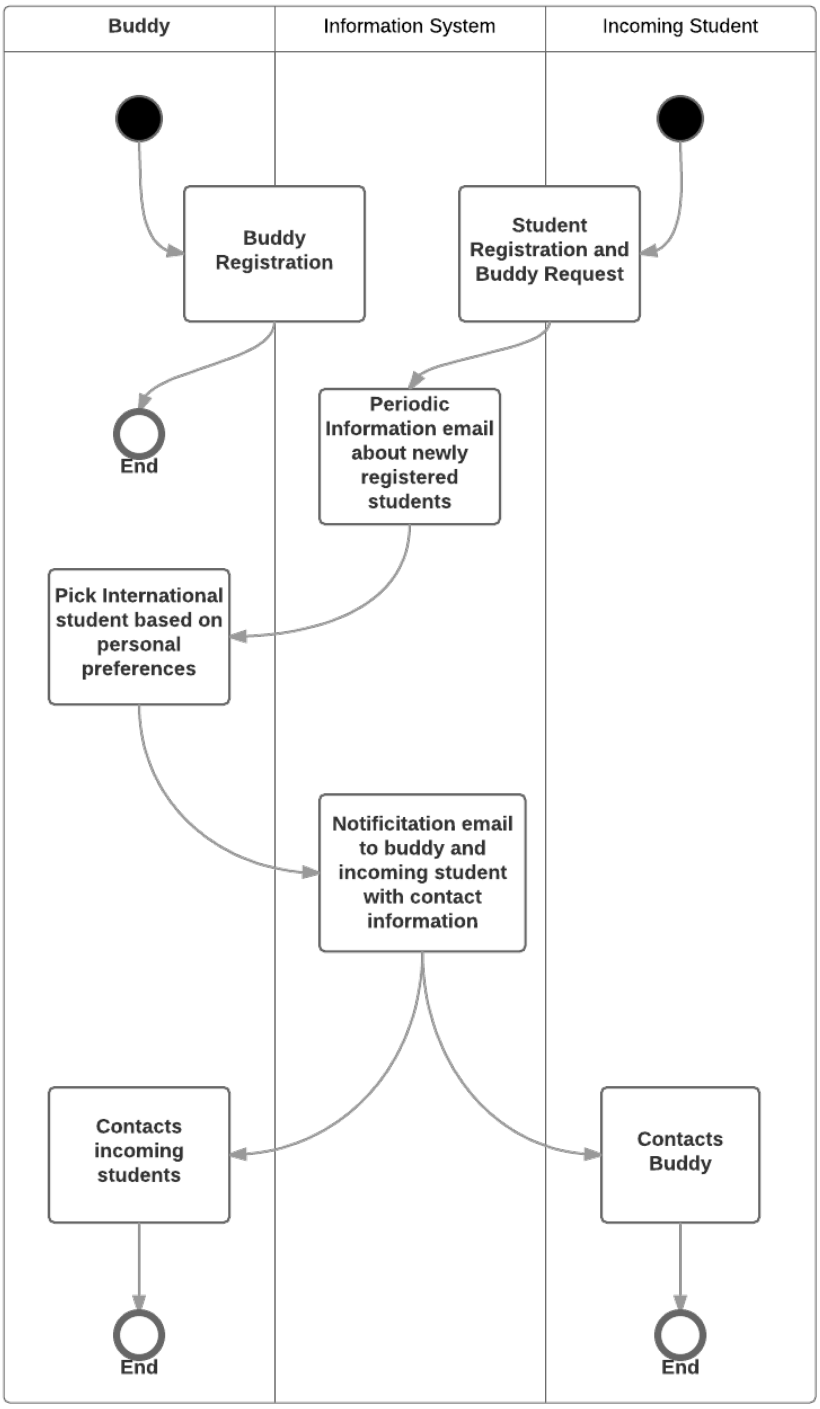
[3]https://www.esn-cz.cz/events/national-platform-varvazov-2015

Figure 3.4: **Business process model - Event Registration**



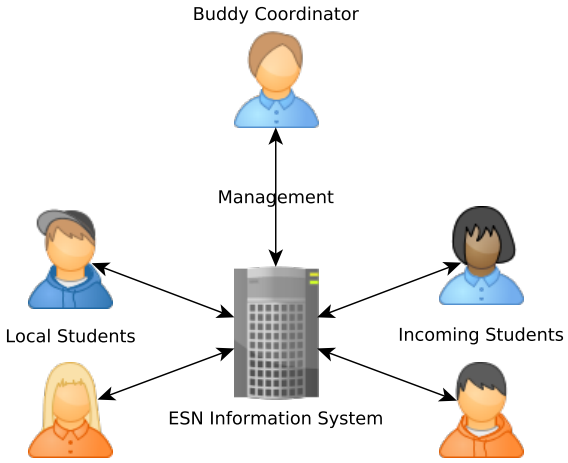
Source: own drawing

Figure 3.5: **Proposed Buddy System Business Process Diagram**



Source: own drawing

Figure 3.6: **Buddy Coordinator is replaced by information system and his new role is to manage special situations**



Source: own drawing

- service agreement, which will have to be signed by each customer of the provided service. In this case he also recommended to set up legal person, in order to transfer the responsibility from physical person. Therefore company unighters s.r.o was established on 26th of October 2016 and the ESN information system is presented under the name *uni-buddy*.

The provider (unighters s.r.o.) of the service ESN information system is registered by the The Office for Personal Data Protection. However, provider can't be held responsible for managing customer's data. The relation between *Provider of cloud service - Customer of cloud service* is defined by the service agreement. The draft of Service agreement is included in the appendix C .

The relation *Customer of cloud service - User of the system* is defined during the user registration. User has to confirm, that he agrees to the Terms of Service and Privacy Policy (see appendix B).

3.4 Existing Solutions For Matching System

The matching system is a tool, which helps creating links between buddy and international student. There are 2 naive solutions for matching system,

which are for free and many ESN Sections are using them.

3.4.1 Emails (Excel table)

The easiest way to connect students is to have a list of buddies and list of incoming international students. This business process is visualised in figure 3.2. Buddy Coordinator has to manually assign the international students to buddies and then send emails to buddies to contact the international student. Other solution is to send email to students with contact information of their buddy. Sending emails to both sides is also possible. Advantages of this system:

- Simple system.
- Buddy Coordinator has full control over the buddy system.
- Direct communication between buddy coordinator - international students and buddies.

Disadvantages:

- The complexity grows with number of international students and buddies.
- Buddies can't choose student depending on their preferences.
- Buddy Coordinator has to send large amount of emails, which increases the chance of human error.
- Not all of the international students need buddy.

3.4.2 Google Apps

A lot of Universities and ESN sections use Google Apps as easy, yet powerful tool for online registering and managing the buddies and students. The standard scenario is one registration form for Buddies, who are interested in participating in the program. Second registration form is for International Students who are interested in getting a buddy.

After both buddies and students are registered, buddies receive an online Google spreadsheet with names and information about incoming students. Very important information for buddy is date and time of arrival. However not all incoming students know the details about their arrival when they register. If they want to change or add this information later on, they usually have to send email to buddy coordinator. Advantages of this system:

- Online communication.
- Fairly Simple system.
- Buddies have option to choose the international student.

Disadvantages:

- Buddy Coordinator has low control over the system.
- In Google spreadsheat anyone can edit anything, if the access to edit is granted. This can lead to accidental or intentional changes in the information by deleting or overwriting some parts of the document.
- Buddy Coordinator has to manually adjust changes in the spreadsheat when contacted by buddy or student:
  - Contact information.
  - Arrival date and time.
  - Wrong information provided.
- All of the personal information of Incoming students is distributed among Buddies.

The common advantage of both naive solution is the simplicity. However with larger number of incoming students (50 and more) both systems disadvantages become stronger than the advantages. Another reason why using naive solutions is not recommended is the prestige of the university. Incoming student will have much better perception of the university if he has an easy to use online tool than if he has to send tens of emails.

3.5 FURPS+ Analysis

Based on the assignment, business process analysis and communication with the customer (ESN), it is possible to create FURPS analysis. FURPS is system for classifying requirements and was introduced by HP in the late 80s. FURPS analysis generally evaluate project or information system according to functionality and usability. Based on FURPS+ analysis it's possible to to define all system requirements with the customer. Lately it is popular to use FURPS+, where symbol + means non-functional system requirements. For example from the field of law or requirements for interoperability with other systems. [10] FURPS+ study is the the base for the design of the system architecture.

- **F**unctionality:
  - Student Registration for current semester.
  - Buddy Registration.
  - Connect incoming international student with local student.
  - Management of registered buddies.
  - Management of registered international students.
  - Creating custom events.
  - Registering students to events.
  - Export students and buddies to excel.
  - Export Event participants to excel.
  - Upload custom logo.
  - Customizable disclaimers.
  - Email notifications.
  - Automatic archive of inactive buddies.
  - Limiting access for buddies by faculty rules.
- **U**sability:
  - Windows 7>, Unix based operation systems (Ubuntu, Debian, Fedora), MacOS, Android and iPhone OS.
  - Web Interface - internet browsers Firefox, Google Chrome (Chromium), Opera, Internet Explorer v6+, Safari.
  - Responsive interface.
- **R**eliability:
  - Availability 99.9% (top downtime 525,6 minutes per year.);
  - Daily database backups.
  - Daily system backups.
- **P**erformance:
  - Response time less than 5s.
  - Average page size less than 500kb.
- **S**upportability:

  - Data storage and computational power is provided by selected cloud hosting.
  - Installation and maintenance is provided by SaaS provider.

- Non-functional requirements(+):
  - System has to be compliant with the international law of European Union about data exchange.
  - System will implement OAuth standard for registering using Google account and Facebook account.

3.6 Cloud Computing

The main goal of this thesis is to design *cloud based* information system. To choose the best solution for the implementation it is necessary to first understand the basic principles, advantages and disadvantages of Cloud Computing.

The term "Cloud Computing" or simply "Cloud" in nowadays world is almost everywhere. The big companies are moving "into cloud" to cut the costs of running self-hosted IT infrastructure. Small start-ups are using cloud to run complex applications without the need of any investment into building IT infrastructure. Families are sharing photos on cloud which they edit and photo-shop directly in the cloud. The question is simple what is "Cloud"?

3.6.1 What is Cloud Computing?

Simply put, cloud computing is a form of computing in which the user accesses any computing resource remotely through a simple client, which in most cases is a web browser. This resource could be a software application or an operating system or remotely located hardware. [14]

Cloud Computing exists mainly thanks to technology called *virtualization*. As Hanjura[14] already stated, remote resource can be anything from single operating system to whole hardware infrastructure. It means that nowadays any IT resource can be virtualized. The main concepts of virtualization are:

- Operating System virtualization (Microsoft Virtual Desktop,...).
- Platform virtualization (Virtual CPU, RAM (VMware, VirtualBox, KVM...).

- Storage virtualization (Multiple physical HDDs visible as one,...).
- Network virtualization (Load Balancing, VLANs,...).
- Application virtualization (Software as a Service,...).

Cloud Computing usually combines multiple of mentioned concepts into one service. For example Infrastructure as a Service (IaaS) combines storage, network and platform virtualization to achieve dynamic resource management, scalability and high availability.

The whole concept brings many indisputable advantages, but also variety of threats and complications which have to be taken into account. Next sections briefly describe the core principals and different types of cloud computing services.

3.6.2 Layers of cloud services

Because cloud is generally any remotely located resource including hardware, it is convenient to divide cloud services into 3 different layers. Each layer is used by different users, however they are usually depending on each other. The most common division is into three layers:

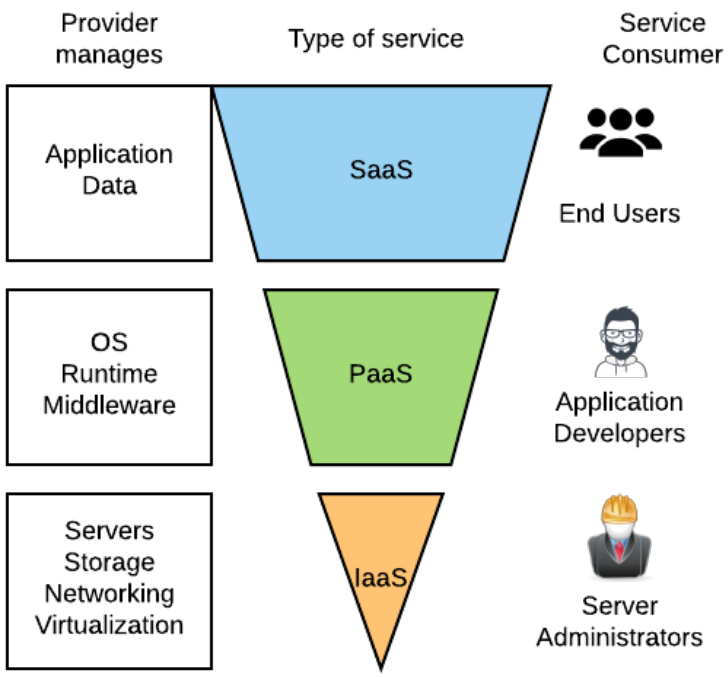
- SaaS - Software as a Service.
- PaaS - Platform as a Service.
- IaaS - Infrustructure as a Service.

Image 3.7 illustrates the dependencies between the three main layers. Starting from the bottom, IaaS is the "top" of the inverted pyramid. The pyramid is inverted, because it horizontally symbolises the general number of users. IaaS can be used only by the server administrators, as it requires advanced knowledge of environment configuration, networking, server administration and does not work "out of the box". More about IaaS in section 3.6.3.

Next layer is PaaS in the middle. PaaS has already wider range of users, as the provider of PaaS takes care of server administration in IaaS layer. Thanks to this the users of PaaS are usually application developers, who does not have to have the knowledge of server administration and only use the service of configured and running environment "out of the box". More about PaaS in 3.6.4

The last and largest layer is SaaS. Users of SaaS are the end users of applications and systems. SaaS provides software working out of the box

Figure 3.7: **Layers of cloud computing - users point of view**



Source: own drawing

without installation (for example Gmail). Thanks to this the users does not have to take care of software installation and updates and directly use the provided software "out of the box". More about SaaS in 3.6.5

3.6.3 IaaS - Infrastructure as a Service

IaaS provides customers with remote virtualized hardware resources on demand. It is common, that the resources are physically spread across multiple data-centers. The typical attributes of IaaS are:

- scalability,
- creation and configuration of requested resources in couple of minutes,
- hardware independent,
- geographically independent,

- easy migration,
- dynamic billing based on the actual usage.

Disadvantages of IaaS are usually:

- manual environment configuration (web servers, databases, firewalls...);
- dynamic billing requires specific application optimisation;
- unexpected costs - the price is not fixed and depends on the actual usage;
- some network configurations might be unavailable due to specific virtualization solution of IaaS provider.

The current most popular IaaS providers are:

- Amazon EC2 [4]
- Microsoft Azure [5]
- OVH [6]

Infrastructure as a Service is usually used by Server Administrators and Network Architects, who use it to create environments to run the platform. More about the platforms running on IaaS is in the next subsection 3.6.4.

3.6.4 PaaS - Platform as a Service

Platform as a Service "*provides developers with the underlying platform to use to develop their apps. It takes care to support a specific language or technology that the stack developers want to use.*" [14] PaaS is very important for stack developers, who does not have time or resources to run their own platform. This can be very useful for new startup companies, which are usually scarce on resources both financial and manpower. For bigger companies, or any other projects which are capable of building and maintaining their own infrastructure, PaaS is usually not a good solution. PaaS is always partly and sometimes fully depending on the provider of PaaS and represents a unique solution. In case the company decides to change

[4] https://aws.amazon.com/ec2/
[5] https://azure.microsoft.com/
[6] https://www.ovh.com

the provider, migrating code from this unique solution can be costly, time consuming or even impossible (so-called *vendor lock-in*).

Example of fully depending platform is Google's App Engine[7]. This is great platform which provides users with features like automatic scaling, security scanning, load balancing, health checks and application logging. The basic service is very cheap, however the price is based on actual usage. If the application is met with a great success, the price using Google's PaaS might grow rapidly. Other solution would be much cheaper or even more effective and faster. But in case of PaaS is our solution interconnected with provider's solution and it might be even more expensive to change the provider or technology than keep paying the current service.

One of the leading PaaS providers nowadays is certainly Heroku. [8] *The Heroku philosophy is to let developers focus solely on writing web applications and forget about servers.*[14] Unlike Google's App Engine, Heroku provides "private spaces", which are similar to virtual private servers and the applications developed for Heroku environment are less interconnected with the platform. The disadvantage of Heroku are the performance issues. Heroku runs on standard Amazon EC2 units. With all the additional services that Heroku provides the single unit provided by Heroku will be less effective than the same unit from EC2. Therefore for application requiring higher computational resources, the costs will be higher than using a standard EC2 unit in form of IaaS.

In conclusion before deciding whether to use PaaS and profit from the out-of-the-box infrastructure or use IaaS and build tailored infrastructure it is important to analyse and try to predict the size of the project, the resources which the application might require and if in the future the application will change technologies. If the application has to be fast done and will be fairly simple, it is good idea to use one of the PaaS provider. The development will be faster and further maintenance will require less human resources. On the other side for larger projects it might be wise to invest into building tailored platform using IaaS. This strategy brings higher degree of control and easier customization, than using pre-made solutions.

3.6.5 SaaS - Software as a Service

SaaS is the most common type of cloud service. It usually runs on top of PaaS and provides product for the end users - customers with basic IT

[7] https://cloud.google.com/appengine/
[8] https://www.heroku.com/

skills. *This model frees the customers from the need to install the software locally and thus to provide the required resources themselves.*[5]

The biggest advantage of SaaS is in it's multi-tenancy. *"In cloud computing, multi-tenancy means that a SaaS (Software as a Service) vendor provides a single version of its software for all its customers. This differs from a single-tenant hosted solution, where the application is housed on a vendor's server but the codebase is unique for each customer."* [17] Having multi-tenant architecture has following advantages:

- Lower cost of ownership.
- Worry free capacity.
- Access the latest releases.
- Configurable to customer needs.

Lower cost of ownership means that there is less expenses on continuous upgrades. Customer pays recurring subscription fee (monthly or annually...) which covers the vendor's running costs and further development. Therefore new releases and updates are included in the subscription and the customer does not need to pay extra for new version of the software.

Worry free capacity stands for the fact, that the customer does not have to worry about increasing the storage and computational capacity if the business and usage grows. Managing capacity is responsibility of the SaaS vendor. And because SaaS is build on top of PaaS/IaaS, increasing capacity is automatically managed by the PaaS/IaaS provider, therefore means no extra work for SaaS provider.

Accessing the latest releases without any delays is probably the most important feature of the SaaS model. Because there is only one version of the software the process of testing new releases is less complex. Customer does not have to worry about updates at all. Another advantage is that the vendor has all the data. Through data analysis and customer behaviour analysis (usually done with integrated SaaS analytics tool) the vendor is able to identify problems or potential upgrades and expedite it's development. This process is rather complicated and costly in case of proprietary software.

It might seem, that customization of multi-tenancy SaaS model is not possible, because there is only one version of the software and each customer has little bit different needs. To satisfy those different needs SaaS vendor has to implement service which is *configurable to customer needs*. This is usually done through various sets of properties and custom attributes. What can be configured?

- Brand identity.
- Custom design.
- Other functional properties.

Based on the research done in this section it is decided, that the information system will be implemented in the form Software as a Service. There is high probability that the system will be used by hundreds of universities across European Union and will be extended by custom modules. Therefore it is better to create tailored platform using Infrastructure as a Service in order to maintain high degree of customization.

3.7 Platform and Framework

One of the most important steps when building architecture of any system is to choose the right platform on which the system will be build. This section discuss which platform should be used to implement proposed information system. Nowadays the Web development world is too fast paced to try predicting which Framework or programming language will be the best or the most popular in few years or even months.

How to choose the right programming or scripting language? Britton[7] suggests to focus on the following comparison criteria:

- Ease of learning.
- Ease of understanding.
- Speed of development.
- Help with enforcement of correct code.
- Performance of compiled code.
- Supported platform environments.
- Portability.
- Fit-for-purpose.

However comparison of all the possible programming languages is not the purpose of this thesis. Therefore those criteria are used to justify the choice of PHP scripting language for implementing ESN information system. PHP is one of the most popular language in the area of web development. According to W3Tech's survey [25] 82.2% of all websites is powered by PHP.

3.7.1 PHP: Introduction and Performance

PHP is server-side scripting language, which means there is no compilation needed. Server only executes written scripts. This approach has many advantages and disadvantages which have been discussed over times, but no definitive conclusions were ever made.

PHP is generally popular language with large community basis and support from the open source community. PHP is now in version 7.0. Version 7.0 brings large performance improvement over the previous versions. Biggest disadvantage of PHP is the fact, that every single request on the web page triggers the execution of the code including compiling the human readable code into machine readable code. It means that the server has to load and parse the included libraries and execute all the files and load it in the memory. After the execution all the memory is freed again. If there is second request one second later, all this process is repeated regardless on the fact that the same code has been in the memory few seconds ago.

Best strategy to speed up the execution time is to use OPcache. *"OPcache improves PHP performance by storing precompiled script bytecode in shared memory, thereby removing the need for PHP to load and parse scripts on each request."* [6]. The reason why PHP 7.0 is up to 50% faster than the previous versions, is the native OPcache integrated by default.

3.7.2 PHP: Ease of Learning and Understanding

Britton [7] describes 3 stages of learning programming language:

1. Basic understanding.
2. Developing your own style.
3. Learning to read someone else's style.

One of the biggest advantage of PHP is the popularity. Most of the programmers already have the basic understanding. As soon as developer get used to the dollar signs ($) which identify variables, PHP code becomes easily readable. Another advantage is that the server manages the memory allocation and freeing.

PHP does not explicitly declare data types. Server during the execution "guess" the data types based on the content of the variables. The type conversion is automatically performed by the server:

```
<?php
    $int = 10;
```

```
    $string = "hello";

    $output = $int."␣".$string;
    echo $output;
    // result: 10 hello
?>
```

3.7.3 PHP: Speed of Development and Help With Enforcement of Correct Code.

The speed of development usually depends on the quality of the code and vice versa. The question is how to enforce the correct code while keeping the development at the right pace.

In case of PHP the enforcement of correct code is unfortunately rather vague. It is caused by the lack of data types in PHP. In section 3.7.2 this feature is described as an advantage. However from the point of view of correct code enforcement and coding standard, the same feature becomes disadvantage.

How is it possible to enforce correct code and don't slow down the speed of development? There is very convenient solution which solves this problem: Using PHP Framework. Reasons why and detailed description of frameworks is available in sections 3.7.6 and 3.7.7.

3.7.4 PHP: Supported Platform Environments, Portability

According to Britton [7] platform means *"not only the operating-system facilities, but also the middleware facilities, database facilities, and system-management facilities"*. PHP has overall very good portability between platforms. It provides libraries for all popular database systems. Because PHP is not compiled and does not depend on the operating system (it is executed by the web-server) the portability between operating systems is high and usually does not require any changes in the code. For development and even deployment it is possible to use already existing LAMP (Linux-Apache-MySQL-PHP) or WAMP (Windows-Apache-MySQL-PHP) software bundles, which works out of the box. Apache [9] is still the most popular web-server nowadays, however the performance is worse than in

[9] https://www.apache.org/

case of Nginx [10]. More information about choosing the web-server in section 3.7.11.

3.7.5 PHP: Fit-for-purpose

Purpose in the case of information system for ESN is web application. FURPS+ plus analysis in section 3.5 did not identify any functional requirements which would demand high amount of computational resources. PHP does not excel in tasks like computer graphics, encryption or any other algorithms which require full power of CPU. Moreover the execution time and amount of memory is usually limited by the web server. If the script runs out of memory it exits with error:

```
PHP: Fatal Error: Allowed Memory Size of 67108864 Bytes
    Exhausted - 64 MB
```

or in case of the run time limit:

```
Fatal error: Maximum execution time of 30 seconds
   exceeded
```

PHP is generally good fit-for-purpose for standard information system, where the main functions are usually fetching data from database, filtering data, searching data, saving data or deleting data.

One more criterion which has not been mentioned is the current knowledge of the developers. Most of the team already has long time experience with PHP, which makes it better fit than for example Ruby (another scripting language), even though lately the trend was to move from PHP to Ruby. Learning new language and environment would be more costly than implementing it in PHP.

3.7.6 What is Framework?

"A framework is a model of a particular domain or an important aspect thereof. A framework may model any domain, be it a technical domain like distribution or garbage collection, or an application domain like banking or insurance. A framework provides a reusable design and reusable implementations to clients."[21]

Framework generally speaking is a set of tools which take care of repetitive tasks and helps to focus on particular problem. Simple example: The task is to cut an iron tube in 20 pieces of the same length. It is very easy to take the tube, do the measuring and then cut it. But what if there are 1000

[10] https://www.nginx.com

tubes like this? Is it still worth measuring each tube one by one? The answer is no. In this case it's better to create a "frame" or "template", which has the specified lengths and cut all the tubes according to this template.

Software engineers don't cut iron tubes, however some of the tasks when developing a system are similar to this problem. For example almost every not naive system has to deal with some kind of authentication and authorisation of its users. In this case it is much better to use a framework, which already implements the tools to authenticate users. Like this the developer can focus on the specific logic of the system and just use the already implemented system for registering and authenticating the users.

3.7.7 Advantages of Using a Framework

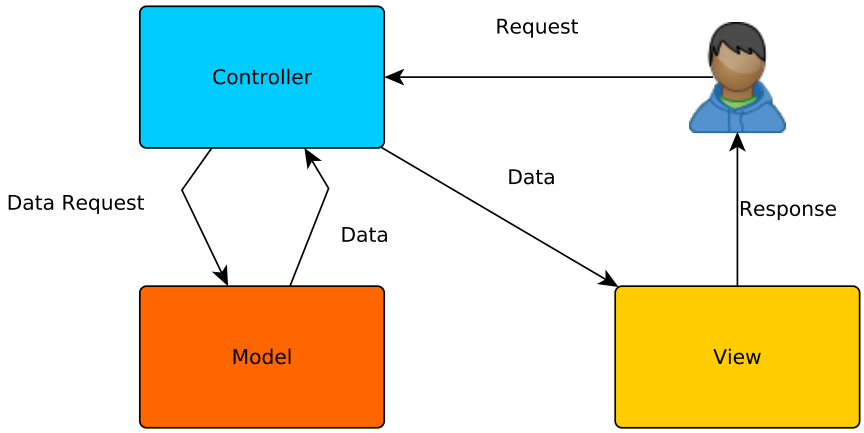
As stated in the previous subsection 3.7.6, the biggest advantage when using a framework are the tools, which makes development easier task. But there is more to it than just the tools. Framework also helps developers in the following tasks:

- Use or extend the Industry Standards.
- Use standardized file structure of the code.
- Use the best practices, sometimes enforced by the framework.
- Use of the Design Patterns.
- Write reusable code decoupled from the framework.
- Scale or refactor the application easier as it grows over time.
- Low level security of the application is provided and maintained by the framework community.
- Follow the MVC (Model-View-Controller (See 3.7.8)) pattern to decouple the frontend, backend and data.

3.7.8 MVC - Model View Controller

The MVC or Model - View - Controller is general architectural pattern for building software application. Its advantage is in decoupling the Business Logic (Controller) from Data Representation (Model) and Graphical User Interface (View). MVC is nowadays used in many frameworks and is accepted as best practice for building web application. Even though recent

Figure 3.8: **Diagram of MVC architecture layers decoupling**



Source: own drawing

release of Google's Node.js and mainly Facebook's ReactJS caused many developers to focus on functional programming, which is based on Lambda Calculus concept. However this new technology is evolving so fast, that it is hard to predict its future and this diploma thesis will focus rather on traditional concepts of Object Oriented Programming like MVC.

Advantage of MVC is separation of the three layers. Separation of Graphical User Interface from the Business Logic is useful when working in a team. The frontend developer specifies which data are needed to create the particular view and the backend developer delivers the data to the view. Like this, frontend developer does not have to study the database scheme and business logic to create the view and focus on design.

Another reason why it is good practice to separate the three layers is the fact, that all three layers can use different technologies and those technologies can be easily changed, if the application implements MVC successfully. For example the database system of the Model layer can be migrated from MySQL to Oracle database system without the need of extensively re-factoring the View and Controller layer.

3.7.9 Which PHP Framework?

How to pick the right one? The most important is the requirements of the application. The FURPS+ analysis (3.5) indicates, that the system will not be naive solution. It will process large amount of data, handle personal data,

handle advanced user authentication and authorisation and process large amount of email communication. Another criterion is the high potential of further extending the system by new modules and functionality. The last criterion are the skills and experiences of the development team. Thanks to the popularity of PHP, it has several advanced and mature frameworks with large community support. Currently the 3 most popular and robust solutions are:[24]

- Laravel[11]
- Symfony [12]
- Nette [13]

The common features of the frameworks listed above are:

- Rappid application development (RAD).
- MVC architecture (see 3.7.8).
- Composer - PHP package manager. [14]
- Integrated templating engine (Twig, Blade, Latte ...).
- Object-relational mapping (ORM) - Doctrine, Eloquent..).
- Object oriented design.
- Open sourced.
- Large community basis.
- Stable development.
- Complete and detailed documentation.

The list of common features is large. Basically all three frameworks have the required functionality and they are good choice for the purpose of this project. It means that the final choice has to be made according to personal preferences and criteria other than functionality. For example each framework is using different templating engine. If the development

[11]https://laravel.com/
[12]https://symfony.com/
[13]https://nette.org/en/
[14]https://getcomposer.org/

team has some kind of preferences regarding the templating engine, it is best to take the preferred one. Although it is possible to integrate different templating engine than the one integrated "our of the box". Important is also the overall experience of the development team with each framework.

At the end the experience of the development team was the decisive criterion in favor of Symfony framework. *"Symfony is built to get back to basics: to develop tools that let you develop faster and build more robust applications, while staying out of your way."*[23]

The main advantage of Symfony framework is the support of configurable environments, where each environment can load different configuration files. This feature will be used to implement multi-tenant architecture. To read more see section 4.5.

3.7.10 Which Database System?

In 2016 it is possible to choose basically from 2 models of how to store data:

- SQL;
- NoSQL (Not Only SQL);

SQL database system is the classical relational database management system (RDBMS) based on tables and relations between the tables implemented through foreign keys. **NoSQL** on the other side has become popular lately with the increasing amount of data needed to be stored in the real time. The most popular NoSQL database system is MongoDB[15] which is *document-oriented database system.* Document store means mainly that the *"records do not need to have a uniform structure, i.e. different records may have different columns."*[2] However NoSQL database systems are good fit for use cases where "Big Data" has to processed in real time. ESN information system will not process large amount of data and classical RDBMS system is sufficient. From the possible open source systems the MySQL database system will be used to store the data. Specifically MariaDB[16], which is compatible with MySQL and developed by the original developers of MySQL under open source license.

3.7.11 Which Web-Server?

In the world of web servers there are 3 major players:

[15] https://www.mongodb.com/
[16] https://mariadb.org/

- Apache [17],
- Microsoft IIS [18],
- Nginx [19],

Because the run-time environment of the production server will be Unix based operating system, Microsoft IIS is not a possibility. The choice has to be made between Apache and Nginx.

Advantages of Apache:

- Reliable.
- Well documented.
- Distributed configuration (.htaccess configuration files).
- Embedded dynamic interpreter.

Advantages of Nginx:

- Asynchronous, non-blocking, event-driven connections.
- Configurable as reverse proxy.
- No .htaccess configuration files.
- Centralized configuration.
- Load balancing through the reverse proxy.

Apache is generally very reliable web server, which has long history of development and large support from the community. However the architecture of Apache is multi-threaded (multi-process), where each request receives it's "worker". Worker is either thread or process. Creating processes and threads is usually resource-intensive task. When Apache has to process thousands of concurrent connections it can run out of the memory or majority of the connections will receive timeout response. In the case of Nginx the handling of concurrent connections is based on asynchronous, non-blocking, event-driven connection. This allows the "workers" to handle not only one request at a time, but thousands connections. This

[17] https://www.apache.org/
[18] https://www.iis.net/
[19] https://www.nginx.com/

is the biggest advantage over Apache and Nginx will be the web server for ESN information system.

Nginx will be the front-end server and reverse proxy and PHP-FPM[20] server will execute the PHP scripts. This solution is more reliable and better performing, than embedded dynamic interpreter in case of Apache PHP-MOD.

[20]https://php-fpm.org/

Chapter 4

Practical part

In practical part the outputs of theoretical part (analysis) is used to design the architecture, front-end, data-model and strategy of SaaS deployment.

4.1 Front-End design

FURPS+ analysis in section 3.5 identified usability requirements which have to be considered in the front-end design:

1. Web Interface.
2. Responsive interface.

Web interface automatically implies, that the front-end has to be in the form of HTML. HTML5 is the most recent standard for HTML markup language. However pure HTML5 is not enought to meet the second requirement - responsive interface.

In the past web design was driven by desktop screens. However with emerging mobile devices the standard desktop layout become hard to read and navigation on this websites become very difficult. The first step was to implement 2 versions of design:

1. Mobile.
2. Desktop.

This solution worked at first, but very soon it became clear, that maintaining two versions of design is impractical. Any change in design required changing two different designs and it resulted in double amount of work.

In 2010 the first article describing principles of responsive design was published by Ethan Marcotte[19]. Even though there were experiments with dynamic size of screens in the past, no one described it and named it precisely. Until Marcotte published his article. It is certain that this was a new milestone in web design evolution. The basic principle of responsive design are dynamic values defined in CSS styles. In the past most of the websites had static web page width or even height. However W3C created media queries as part of the CSS3 specification. In media query the developer can specify media type (screen for example) and set the maximum width of the device which work as condition for loading the css file:

```
<link rel="stylesheet" type="text/css"
  media="screen␣and␣(max-device-width:␣480px)
␣␣and␣(resolution:␣163dpi)"
  href="shetland.css" />
```

[19]

The possibility of having multiple different layouts in one dynamic layout was very tempting and year 2013 was even named *A Responsive Year* by the Mashable[9]. In 2012 it was for the first time when sales for mobile devices exceeded sales of desktops. It meant that mobile devices couldn't be any longer ignored and most of the websites started to change the design into responsive. It is obsolete to create two designs - one for mobile and one for desktop - this is the reason why design of ESN Buddy System will be created also responsive.

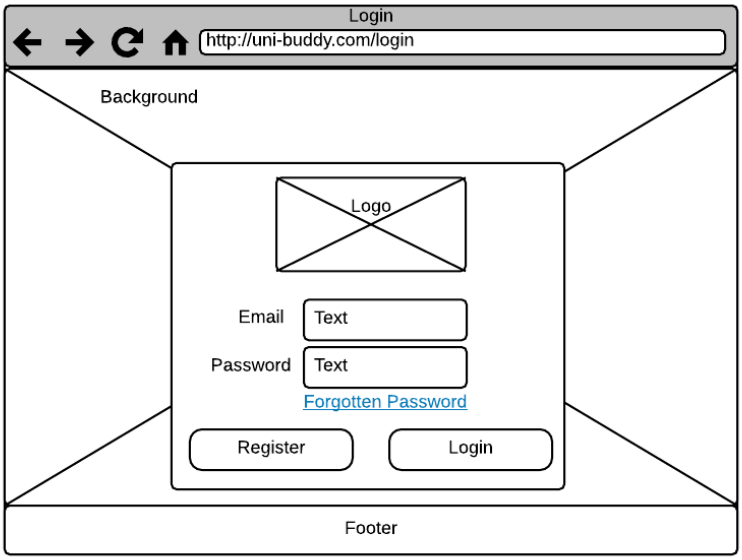
Every time some technology become widely used in the world of it development, there are some frameworks emerging. It's the same case for responsive design. The most popular CSS responsive framework is called Bootstrap. It provides the designer the possibilities to use pre-made CSS classes and elements to build fully responsive design compatible with W3C standards.

4.1.1 Wireframes

This section is dedicated to the wireframe design of the main and most important pages in ESN information system. Wireframe is very important communication tool between the customer and the software architects/designers. It is a *simplified view of what content will appear on each screen of the final product, usually devoid of color, typographical styles, and images. Also known as schematics, blueprints.* [8]

The list of wireframe designs:

Figure 4.1: **Wireframe of the login screen.**



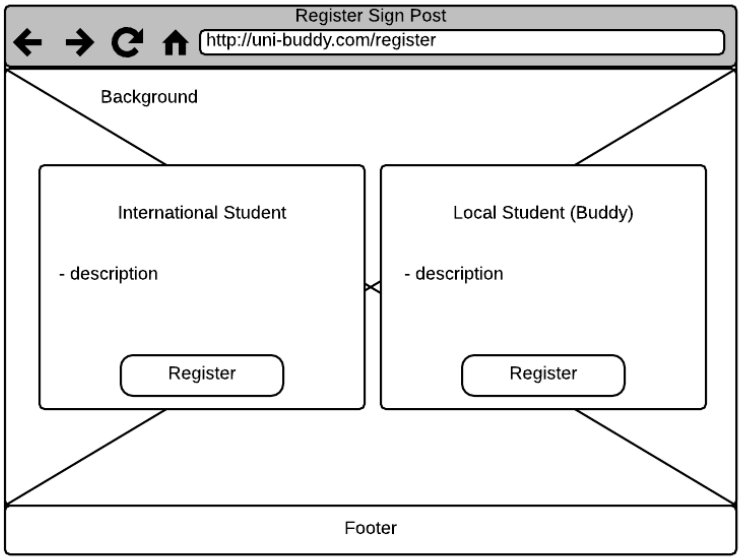
Source: own drawing

1. Login screen. (Figure 4.1)
2. Registration signpost. (Figure 4.2)
3. Dashboard:
    a) Buddy. (Figure 4.3)
    b) Student. (Figure 4.4)
4. User management. (Figure 4.5)
5. User profile. (Figure 4.6)

4.1.2 Login Screen

In Figure 4.1 is wireframe of login screen. This screen should be as simple as possible. The background is customizable background for every section, as well as the logo displayed above the login form. The button for registration and link for password reset are along with the login another important elements of the login screen.

Figure 4.2: **Wireframe of the sign post screen, where users choose which registration they want to proceed with.**



Source: own drawing

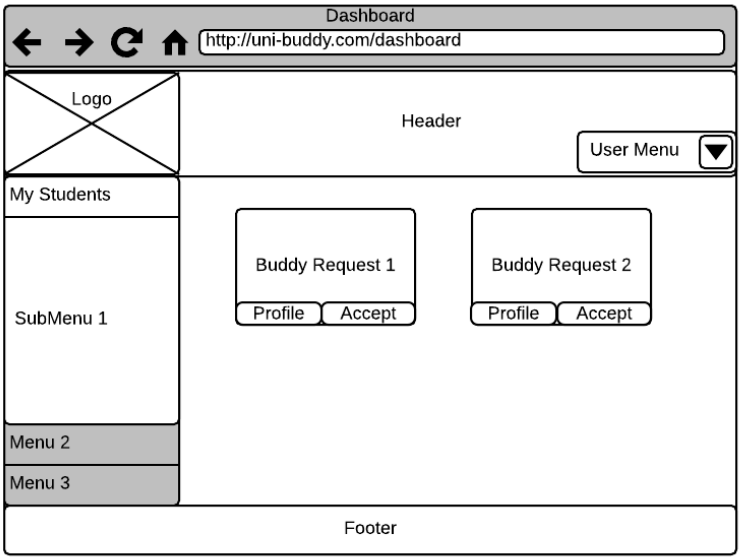
4.1.3 Registration Sign Post

In case that the user does not have account yet, he can create it by clicking the "Register" button on the login screen (figure 4.1). After clicking "Register" button user is redirected to Sign Post screen (figure 4.2), where the user has to decide which type of account wants to create. Description of the differences should help the user to choose the right registration form. First registration is intended for International students (students coming from different universities). Second option is Local student (buddy).

4.1.4 Dashboard

Dashboard is the central screen of the whole system. Dashboard should contain the most important information and elements user needs with simple and intuitive access. To achieve the best result, the system will dynamically include different elements depending on the user's role (see 4.3.17). The two most important roles are Buddy and Student.

Figure 4.3: **Wireframe of the buddy dashboard. Buddies can browse and accept available buddy requests.**



Source: own drawing

4.1.4.1 Buddy

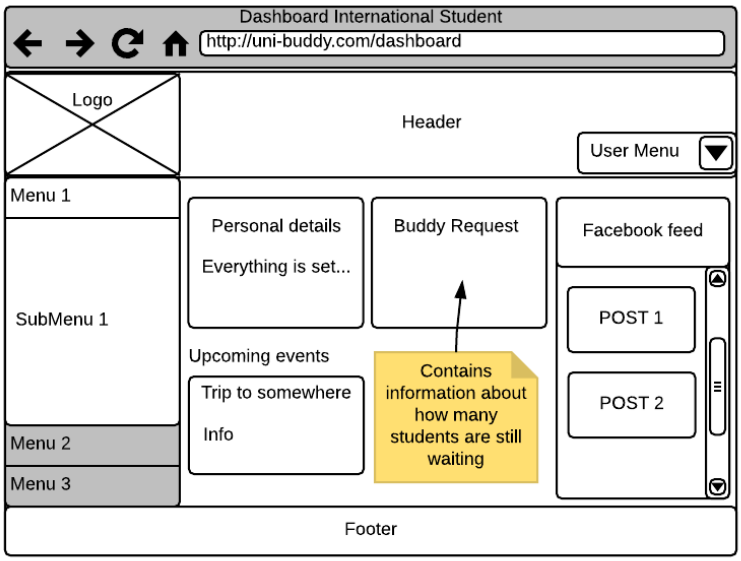
Central task of buddy in the buddy programme is to choose the arriving international students. For this reason buddy dashboard (figure 4.3) contains list of all the available buddy requests. When buddy see this list as the first thing after log in to the system, it should trigger curiosity and encourage buddy to choose some student. Buddy can browse list of students and inspect details of each buddy request by clicking on the profile button.

4.1.4.2 Student

Role of the student in the system is rather passive. The only purpose student can use the system for is to get information. To provide relevant information international student's dashboard contains multiple elements with relevant information for buddy (see figure 4.4):

- Number of buddy requests without buddy.
- Warning in case student didn't fill in some important info in his profile.

Figure 4.4: **Wireframe of the student's dashboard. Students can review and edit their information,see upcoming events and browse the Facebook feed from customer's page.**



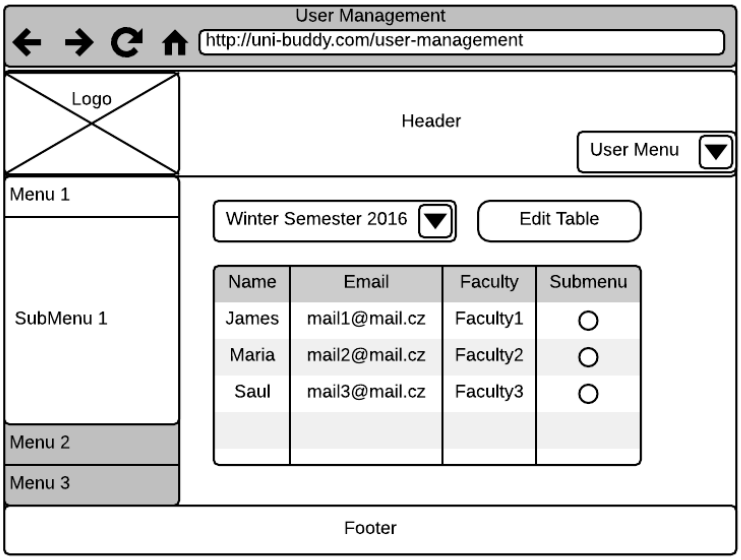
Source: own drawing

- Facebook feed from the customer's page, which can be configured by the administrator.
- List of available trips for student's to register to.

4.1.5 Buddy management

Buddy management is the most important screen for role Buddy Coordinator. Central element of this screen is table with user data. Wireframe 4.5 shows the basic layout. Why using table in this case? Table is the best way to display large number of entities with the same attributes. Users are generally used to work with tables. Using javascript the table will be editable (Button "Edit table"). This function has to be triggered by button to eliminate accidental edits of data. Table also provides automatic sorting and each user has it's own sub-menu (last column on the right). This sub-menu is generated based on the state of entity (if the user is archived, the sub-menu will display option to re-activate the user for example).

Figure 4.5: **Wireframe of the user management, where Buddy Coordinator can edit information of users and go to user's profiles.**



Source: own drawing

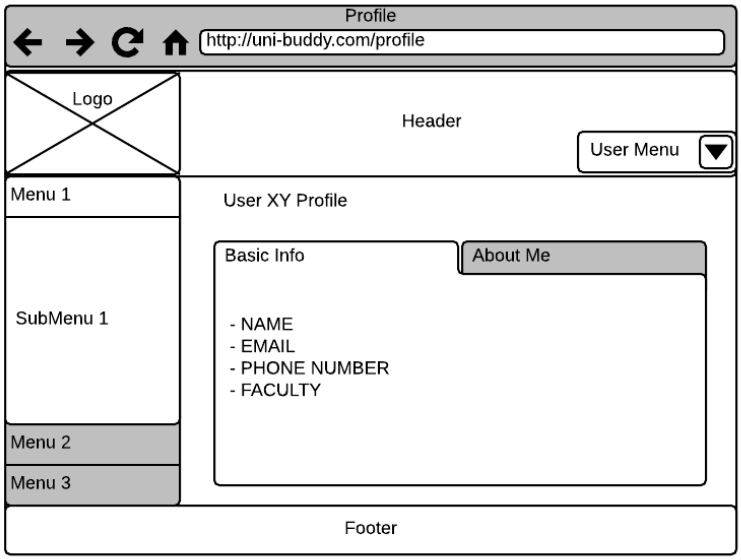
The last important element on wireframe 4.5 is the select box, where user (Buddy Coordinator) can choose from which semester should the page display users.

4.1.6 User Profile

User profile is profile accessible by registered users. Profile is designed in the form of tab-panel. First tab contains the basic information about user like email, name, date of birth and other information the user specified during registration. For example country the user is coming from or faculty where the user will study in the host university.

Second tab display content "About Me". This content is important when buddies are browsing through buddy requests. Both buddies and students are encouraged to write something "About me" to increase the chance of being accepted to the buddy program (in case of buddies) or increase the chance of getting a buddy (in case of international students). It is always better to have personalised request, rather than only the basic information.

Figure 4.6: Wireframe of the user profile.



Source: own drawing

4.2 Use Case Diagram

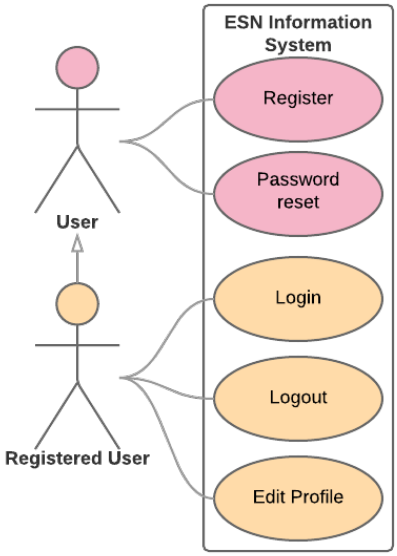
According to Fowler[13] use case is a *"technique for capturing the functional requirements of a system. Use cases work by describing the typical interactions between the users of a system and the system itself, providing a narrative of how a system is used."* According to Arlow[4], there are several steps to succesfuly create Use Case Diagrams:

1. Identifying system boundaries.
2. Identifying system actors (roles).
3. Identifying use cases related to the actors.
4. Specifying use cases.

Use Case diagram is modeled using UML annotations.

The system boundaries are in this case simple - system boundary is the information system for ESN. There are no other interconnected systems or environments.

Figure 4.7: **Use case diagram of basic use cases.**



Source: own drawing

Second step is to define all the stakeholders (actors) in ESN information system. Name and short description of all the actors is in table 4.1

Use case diagram of the basic roles is in figure 4.7. It shows actors User and Registered User with basic system scenarios like login or registration.

The real system roles are captured in figure 4.8. All the scenarios modeled in Use Case diagram 4.8 are the core of the system functionality based on the business process analysis (3.2) and FURPS+ analysis (3.5). The scenarios area described in tables 4.2, 4.3, 4.4, 4.5, 4.6, 4.7 and 4.8.

Figure 4.8: **Use case diagram of advanced roles/actors including their hierarchy.**

Source: own drawing

Table 4.1: **Overview of use case actors.**

Actor	**Description**
User (table 4.2)	Actor user represents general anonymous user. User has not yet decided whether to register as Buddy or Student. Other possible scenario for this actor is to reset password in case user is already registered and just forgot the login credentials.
Registered User (table 4.3)	Registered user is user which can perform set of basic tasks like login, logout or edit profile. This actor is just for illustration and registered user always has to be either buddy or student.
Student (table 4.4)	Student is registered user (international student), who is looking for buddy.
Buddy (table 4.5)	Buddy is registered user (local student), who is volunteering to "be a buddy" for incoming students.
Buddy Coordinator (table 4.7)	Buddy coordinator is buddy, who is in charge of the buddy program and handles special requests from other users which can't be solved via the system.
Event Manager (table 4.6)	Event manager is buddy, who is in charge of managing events, registering users to events and other tasks.
Administrator (table 4.8)	Administrator is buddy coordinator and event manager. Additionally is in charge of setting up the system and system customizations.

Source: own analysis

Table 4.2: **Use case scenarios for actor User.**

Use Case	**Description**
Register	Anonymous user can register in the system using 2 different registration forms: 1) Buddy 2) International student
Password Reset	In case anonymous user is already registered, but forgot the password, he can use form to send email to his mail box with link to Reset Password.

Source: own analysis

Table 4.3: **Use case scenarios for actor Registered User.**

Use Case	**Description**
Login	Registered user can Login the system using email and password.
Logout	Registered user can logout after he login the system. Necessary in case when user is using public computer for example.
Edit Profile	Any registered user can edit his profile information like name, email and other attributes

Source: own analysis

Table 4.4: **Use case scenarios for actor Student**

Use Case	**Description**
Create Buddy Request	Student can Create Buddy Request in order to express interest in getting a Buddy.
Register for event	Student can register for events imported to the system by Event manager.

Source: own drawing

Table 4.5: **Use case scenarios for actor Buddy.**

Use Case	**Description**
List all buddy requests	Buddy in order to pick the right buddy has the possibility to list all buddy requests created by Students and browse them.
Accept buddy request	Buddy accepts the request from Student and they are automatically paired together.

Source: own analysis

Table 4.6: **Use case scenarios for actor Event Manager**.

Use Case	**Description**
Import event from Facebook	Event Manager can use a link to public Facebook event and system will impot this event into ESN information system.
Register Student for event	Event Manager can register students to events created in the system.
Remove Event	In case that the event is not correct, or canceled, Event Manager can delete this event from the system.
Edit Event	When time, name, capacity or any other attributes of event changes, Event Manager can edit those values in the system.

Source: own analysis

Table 4.7: **Use case scenarios for actor Buddy Coordinator.**

Use Case	Description
Release Student	After buddy accepts Buddy request, it is not possible to revoke the request from the side of buddy. He has to contact Buddy Coordinator, explain the reasons why he wants to revoke the request and Buddy Coordinator then revokes the buddy request making it accessible to other buddies again.
Assign Buddy	In case Buddy Coordinator wants to have a control over the buddy program it is possible to assign buddy to buddy request manually.
Delete Student	Buddy Coordinator can delete student, usually when student accidentally creates duplicate account.
Edit Student and Buddy	Buddy Coordinator can edit some of the user details like phone number or number of maximum students the buddy can take.
Activate Buddy	After buddy registers, buddy coordinator has to activate his or her account, otherwise the buddy can't login.
Archive Buddy	After buddy is inactive for certain period of time, he can be archived, so the Buddy Coordinator can keep track of the active buddies, who are available for the next semester to pick up international students.
Reject Buddy	Buddy can be rejected by Buddy Coordinator. Rejected buddy does not have access to the system anymore. Rejecting is done usually when the Buddy profile violates the conditions to become a buddy.

Source: own analysis

Table 4.8: **Use case scenarios for actor Administrator**.

Use Case	**Description**
Set start of winter and summer semester	Administrator sets the date when system switches between semesters. Administrator set this date without specifying the year. Based on the current date system automatically calculates the correct years and semesters.
Upload Logo	Part of the system customization is also possibility to upload custom logo, which is then displayed on all the screens and send via emails.
Upload Login Background	Login background can be uploaded to customize the login and registration screen. Background image will be automatically covered with blur filter to not disturb the user during login and registration.
Set maximum of students per buddy globally	If the policies about the limit of international students one buddy can take change, Administrator can easily force this settings to all existing buddy accounts. This option however overwrites any custom changes made prior using this function.
Set name of the section	Administrator can set the name of the section/-customer which is using the system as part of custom customer identity along with the custom background and logo.
Set contact email	Contact email is included in all the automatic emails which are being sent users. It should be contact in case of any organizational problems ("my student/buddy does not communicate", "I don't have time to pick up my student" and similar scenarios).
Manage user roles	After user registers as buddy, the only role he has is "Buddy". However Administrator can assign or revoke special user roles for buddies: 1)Administrator, 2) Buddy Coordinator, 3) Event Manager.

Source: own analysis

4.3 Data Class Model

One of the features of Symfony Framework is, that it supports ORM framework Doctrine[21] by default. Thank to this it is possible to create the data model in UML as classes and then create the classes with Doctrine annotation syntax.

The UML Data class diagram had to be divided into two figures. Figure 4.9 includes the most important classes - User, Student and Buddy and shows the classes which connects Buddy and Student together.

The central element of the second figure 4.10 is data class Semester, which is connected to data class Student from the first part of diagram (figure 4.9).

The following sub-sections describe all data classes used in ESN Infromation system. Data classes are ordered in alphabetical order.

4.3.1 AppSettings

This class holds information about custom application settings:

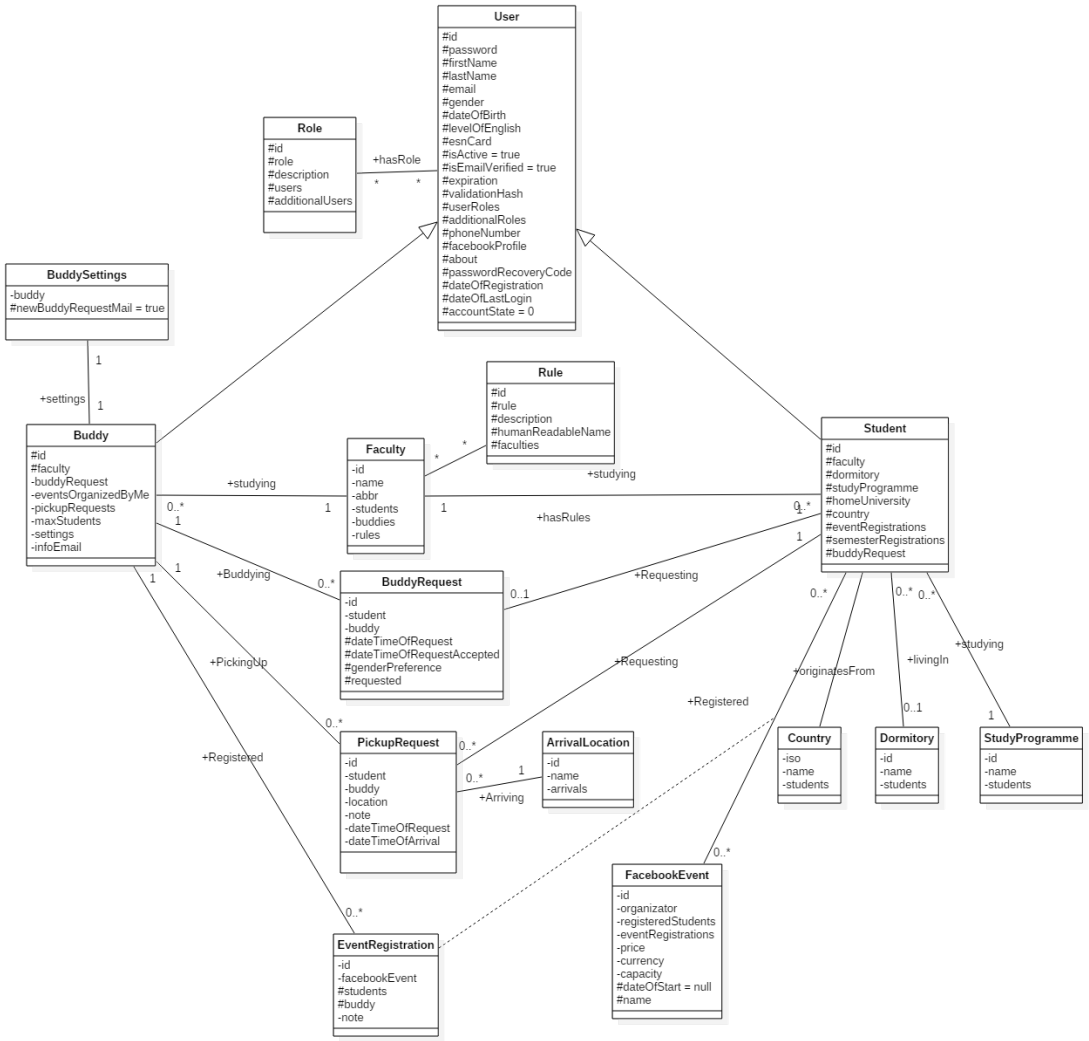
- Start date of winter and summer semesters.
- Link to current semester.
- Name of the ESN section.
- Facebook page url of the ESN section.
- Custom uploaded logo and background.
- Contact email, which is included in every notification sent to students and buddies.
- Application variables, which are universal attributes. (see 4.3.2)

4.3.2 AppVariables

Class AppVariables is introduced to deal with a problem of growing custom attributes, which are required to customize the system. If every attribute is added to AppSettings (see 4.3.1) the table will grow to the sides. This behaviour is not desirable. AppVariable can be easily defined and used for string or numeric attributes. It is identified by attribute *name*. *Human Readable Name* is defined to print it on the form where user set those attributes as well as *Description* and the last attribute is *Value*.

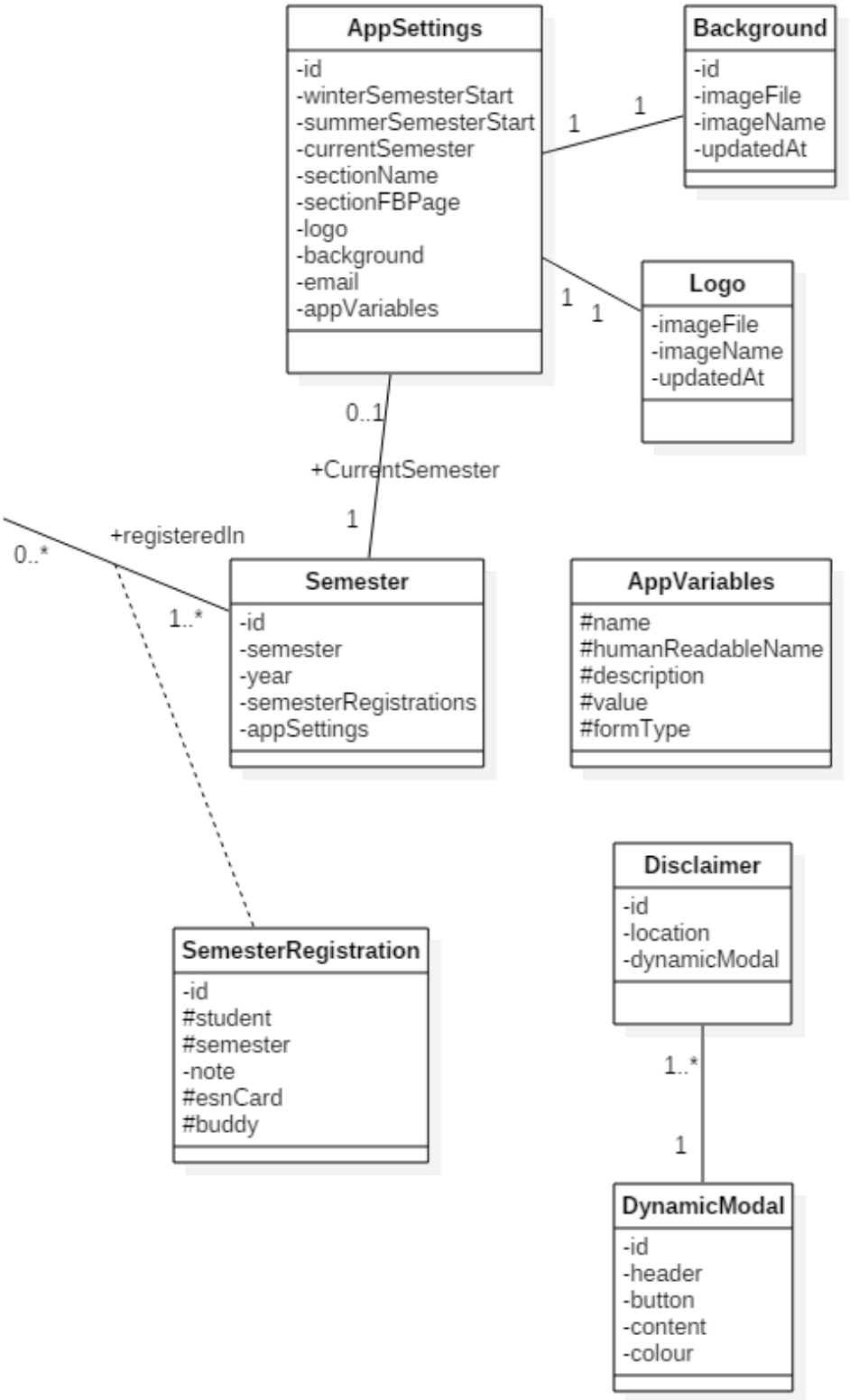
[21] http://www.doctrine-project.org/

Figure 4.9: **First part of data class diagram with the most important classes like User, Student, Buddy and others.**



Source: own drawing

Figure 4.10: **Second part of data class diagram.**



Source: own drawing

4.3.3 ArrivalLocation

This class contains records about all the locations, where international students can arrive. For example "Vaclav Havel International Airport". Administrator is responsible for correctly setting those arrival locations in the initial configuration.

4.3.4 Background

Simple class which holds information about user defined background for the login screen.

4.3.5 Buddy

Class Buddy extends class User (see 4.3.23). Thanks to this inheritance, every buddy is distinguished by type of class Buddy. Moreover, specific attributes like *maxStudents* (defines the maximum of students this buddy can take for each semester) or *settings* (private settings of buddy profile) are added to the base list of attributes inherited from class User (see 4.3.23).

4.3.6 BuddyRequest

During the registration for International students instance of BuddyRequest is created. It contains information about who is the international student, when did he requested for buddy and gender preference.

4.3.7 BuddySettings

This class contains only one attribute - *newBuddyRequestMail*. This attribute tells the system if the buddy wishes to receive automatically generated emails with information about new buddy requests. Those emails are sent every 12 hours with all buddy requests created in past 12 hours, which are still without buddy.

However BuddySettings are ready to hold more custom settings for buddy in the future.

4.3.8 Country

Class which holds information about the country. Those countries are uploaded according to ISO 3166-1.

"ISO 3166 is the International Standard for country codes and codes for their subdivisions. The purpose of ISO 3166 is to define internationally

recognised codes of letters and/or numbers that we can use when we refer to countries and subdivisions. However, it does not define the names of countries – this information comes from United Nations sources (Terminology Bulletin Country Names and the Country and Region Codes for Statistical Use maintained by the United Nations Statistics Divisions)." [16]

Because the ISO code of the country is unique, this iso code is used to identify the object Country. Second attribute is the name of the country.

4.3.9 Disclaimer

The purpose of this class is to hold information about the user defined disclaimers, which are displayed before entering the registration form.

4.3.10 Dormitory

Simple class which holds information about dormitories, where the incoming students can live.

4.3.11 DynamicModal

Is universal class, which defines modal html element. Modal is element defined by bootstrap - frontend framework [22]. Advantage of the dynamic modal is that the user can define custom header, content and button including colours.

4.3.12 Event Registration

Event Registration is an association class, which is attached to association between Student and FacebookEvent classes. Usage of association class is necessary in this case, because it holds additional information about the registration: *note* (for example if student is vegetarian) and *buddy* - the buddy who made this registration.

4.3.13 FacebookEvent

Purpose of class FacebookEvent is to save the information about facebook event, which was imported to the system. It is not necessary to store all the information, as it can always be downloaded from facebook throu facebook API. The important information to store is the event ID and additional information important for registering students:

[22] see http://getbootstrap.com/

- price;
- currency;
- capacity;
- date and time of event start;
- name;

4.3.14 Faculty

Simple class which holds information about faculties in the University.

4.3.15 Logo

Simple class which is used to store the custom uploaded logo of organisation.

4.3.16 PickupRequest

Because not all of the students know the date, time and place of arrival, pickup request is separated from buddy request. If the pickup request is not created, the buddy request shows that student does not know yet the details about his or her arrival. However the student can always create the pickup request on his or her profile after they are successfully logged in the system.

There is one important restriction for creating pickup request - the pickup request must be created at least 1 day (24 hours) in advance before the students arrival.

4.3.17 Role

This class holds system information about the roles defined in the application and reflecting the Actors specified in Use Cases (table 4.1):

- ROLE_STUDENT (actor Student);
- ROLE_BUDDY (actor Buddy);
- ROLE_EVENT_MANAGER (actor Event Manager);
- ROLE_BUDDY_COORDINATOR (actor Buddy Coordinator);
- ROLE_ADMINISTRATOR (actor Administrator);

4.3.18 Rule

Sometimes a special rules for matching buddies are necessary to be applied. For example one faculty wants that only students and buddies from their faculty are matched together. This class defines three rules:

- RULE_PRIORITY;
- RULE_ONLY;
- RULE_CLOSED;

Those rules are described in detail in section 4.4.1

4.3.19 Semester

Semester is Entity which stores information about name of the semester (either Winter or Summer) and a year of semester.

4.3.20 Semester Registration

This entity is very important for international students registrations. It distinguishes students between semesters. The attribute student is id of the registered student and semester is id of the semester the student is registered in.

4.3.21 Student

Class Student extends class User (see 4.3.23). This allows to have a core user entity with all the shared attributes like name, email, password and other. However Student entity is necessary in order to virtually distinguish between buddies and students in the system. Moreover the class Student has some additional attributes like studyProgramme (usually Erasmus or Exchange), which class Buddy (see 4.3.5) does not have.

4.3.22 StudyProgramme

User defined Study programmes for International students to choose from. Usually it is Exchange or Erasmus, however thanks to this class users can customize the Study Programmes according to their university standards.

4.3.23 User

MySQL does not implement class inheritance by default. However in Object Programming inheritance is very useful tool to lower the code complexity and create logical inherited classes. The classical use of inheritance is for the User classes. User is the base class, which is inherited by class Student and Buddy. User class contains attributes and methods which are necessary for correct implementation of application security. Class is identified by automatically generated ID. It was decided to avoid usage of nicknames for login and user logs in only by using email. Email is unique attribute.

4.4 Custom solution for specific requirements

FURPS+ analysis3.5 identified non-trivial functional requirement. To achieve a user friendly functionality it is necessary to provide thorough analysis of the problem, before implementing it. Custom analysis and solution is described in the following sub-section.

4.4.1 Faculty Rules

As mentioned already in section 4.3.18, one of the functional requirement from the FURPS+ study (section 3.5), is to limit access for buddies by faculty rules. There are 2 particular rules, which has to be customizable and later implemented in the system:

1. International student coming to faculty A can only be picked up by buddy from the same faculty A.
2. International students from faculty B will be assigned manually to available buddies.

The definition of the first rule is problematic, because it doesn't say if buddy from faculty A can take **only** students from his faculty, when this rule applies, or if he can take also students from **other** faculties, which have no rules set.

To overcome this dilemma a third rule is invented. The 3 rules created based on the requirements are:

1. Prioritized Faculty.
2. Only Faculty.

3. Closed.

Rule **Prioritized Faculty** is the "weakest". Buddies from certain faculty will see and be able to accept only students from the same faculty as they are studying. However, once all students from their faculty are taken, buddies are able to view and take any student requests available.

When a new student from their faculty registers, the rule applies again and student will be prioritized. International students from the faculty with applied rule will always get a buddy within the same faculty.

Rule **Only Faculty** is a strict limitation for buddies. Buddies from certain faculty are limited to view and choose only students who studies the same faculty, without any exception. International students will always get a buddy within the same faculty.

The last rule **Closed** is the most strict one: buddies from affected faculty are unable to view and pick any students.

The implementation of faculty rules has to have low impact on the application performance. In case that the rules are set and the system contains 100 buddy requests from 3 faculties, the application would have to go perform approximately $100 * 3$ evaluations of the eligibility of user to be able to view buddy request.

The proposed solution is matrix of evaluation results with values *true* or *false*. In table 4.9 is faculty matrix, which stores the result of performed evaluation when rule *Closed* is applied for all faculties. From now the evaluation results are obtained quickly from the matrix accessing the coordinates by **X= Student's faculty ID** and **Y: Buddy's faculty ID**. This decision system is universal and speeds up the process significantly, lowering the number of DB queries and performed if:else conditions.

Table 4.9: **Matrix of faculties in case of rule *Closed* for all faculties**

Faculties	A	B	C
A	0	0	0
B	0	0	0
C	0	0	0

Source: own analysis

Table 4.10 shows the faculty matrix in case of rule *Prioritized Faculty* for faculty **A**. From the table 4.10 is evident, that buddies from faculty A (axis Y) can access students from faculty A and also students from other faculties (axis Y). However buddies from faculty B and C have restricted access to students from faculty A.

Table 4.10: **Matrix of faculties in case of rule *Prioritized Faculty* for faculty A**

Faculties	A	B	C
A	1	1	1
B	0	1	1
C	0	1	1

Source: own analysis

The last table 4.10 shows the faculty matrix in case of rule *Only* for faculty **A**. The situation has changed from the previous scenario by restricting access to students from faculty **B** and **C** for buddies from faculty **A**.

Table 4.11: **Matrix of faculties in case of rule *Only* for faculty A**

Faculties	A	B	C
A	1	0	0
B	0	1	1
C	0	1	1

Source: own analysis

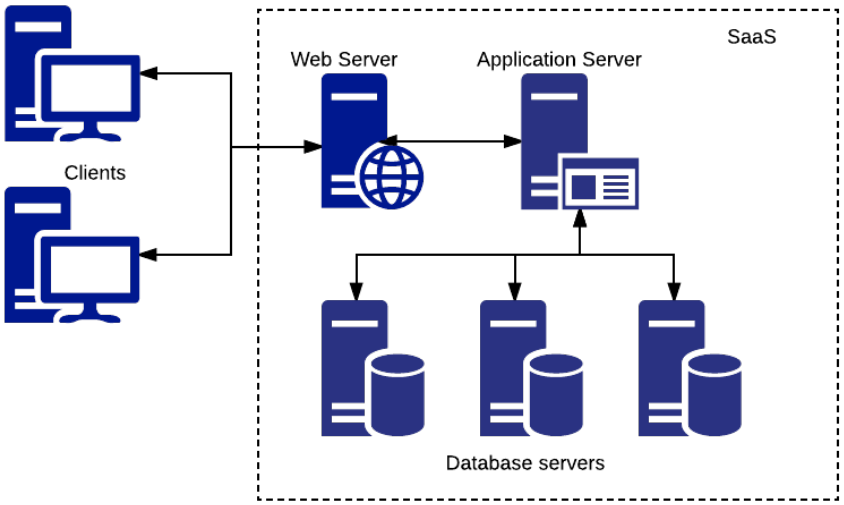
4.5 SaaS Architecture and Deployment Strategy

Analysis of SaaS in section 3.6.5 described the advantages of multi-tenant architecture when establishing SaaS application. However there are multiple ways how to implement it:

- One database with table prefix for each tenant.
- One database with logically separated tenants (via attribute ID for example)
- Each tenant with separate database.

Better fit for the purpose of ESN information system is the last solution - separated databases for each account of the service as shown in the figure 4.11.

Figure 4.11: **SaaS multi-tenant architecture of ESN information system**



Source: own drawing

Thanks to this architecture it is possible to implement load balancing, where each database might be physically located on different servers or services. Moreover the tenant benefits from higher security. Each database has unique user. If the database credentials are hacked in case of the first 2 solutions of single database for all tenants, the attacker gets access to the data of all tenants. If the same happens with multi-database architecture, the attacker would get access only to single instance of the application.

How does the application recognise which tenant to serve? Each instance of the application has it's own domain name to distinguish environments on the web server level. Therefore each instance has it's own Nginx configuration file, which contains the name of the of the environment in the parameter *SYMFONY__ENVIRONMENT*:

```
server {
  listen 80;

  server_name customer1.esnsystem.com;
  root /path/to/symfony/web;

  # pass the PHP scripts to FastCGI php7.0-fpm server
  location ~ ^/(app|app_dev)\.php(/|$) {
```

Figure 4.12: **Parameters for database connection**

```
parameters:
    database_driver: pdo_mysql
    database_host: 127.0.0.1
    database_port: null
    database_name: dev
    database_user: dev
    database_password: test
    mailer_transport: smtp
    mailer_host: 127.0.0.1
    mailer_user: null
    mailer_password: null
    locale: en
    secret: secret
    tracking_id: null
```

Source: own screen-shot

```
    fastcgi_pass   php7.0-fpm;
    include fastcgi_params;
    # environment name for customer1
    fastcgi_param SYMFONY__ENVIRONMENT customer1;
  }
}
```

According to the official Symfony documentation for version 2.8[3] *"any environment variable prefixed with SYMFONY__ is set as a parameter in the service container"*. Symfony framework has great power of customizations. To achieve multi-tenant architecture with Symfony, it is possible to use Symfony environments, which are separated by different configuration files: each Symfony environment loads different configuration file, where all important parameters for database connection are stored (figure4.12). Symfony decides which environment to use based on the parameter *SYMFONY__ENVIRONMENT*.

```
// /web/app.php - Front Controller
if (isset($_SERVER['SYMFONY__ENVIRONMENT'])){
    $kernel = new AppKernelSaaS(
    $_SERVER['SYMFONY__ENVIRONMENT'],
    false
    );
}
else { die ('something went wrong');}
```

Multi-tenant solution with Nginx configuration files brings one more feature. The customer can ask the provider for custom domain/sub-domain. Environment remains the same, however in the configuration file the *server_name* directive is set to the custom domain name. For example buddy.customer1.com. However this solution requires advanced cooperation with the customer. It is necessary to set new DNS record:

```
buddy.customer1.com      CNAME esnsystem.com
www.buddy.customer1.com CNAME esnsystem.com
```

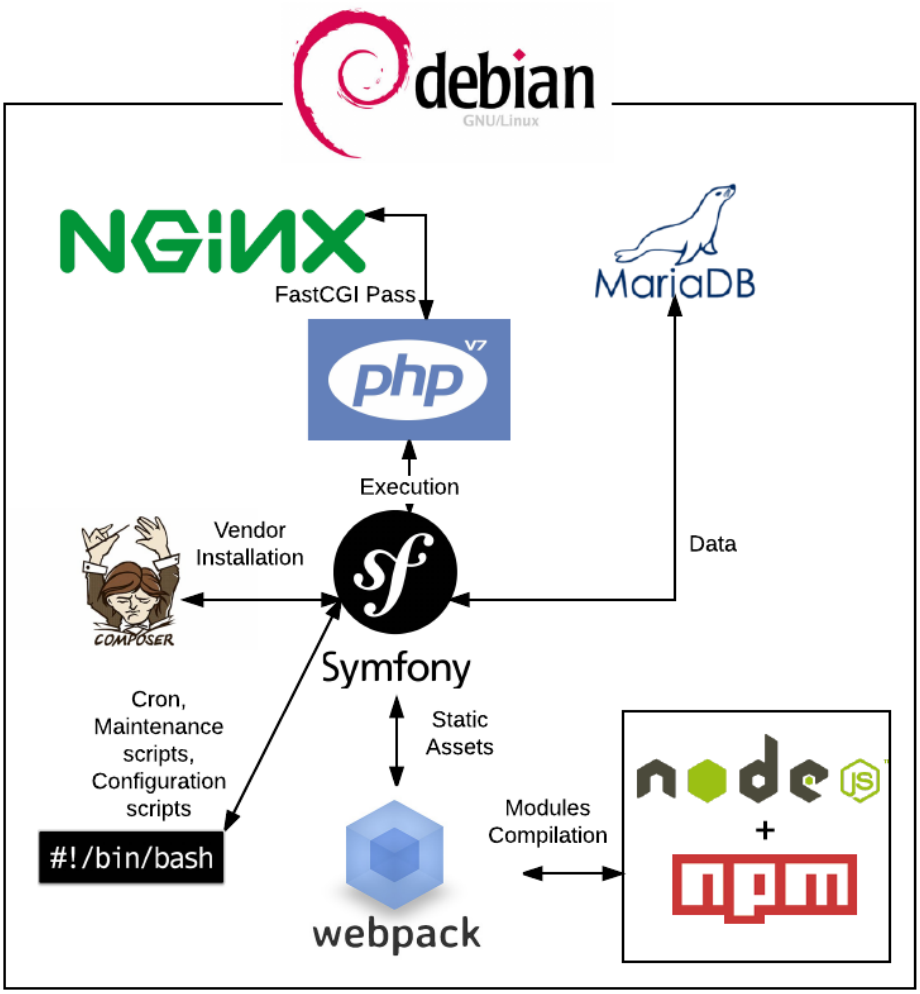
The DNS CNAME ("canonical name") record exists to provide the canonical name associated with an alias name.[11] Thanks to the CNAME alias the user accessing buddy.customer1.com will be associated with domain esnsystem.com and it's IP address. It would be possible to use A (IPv4) and AAAA (IPv6) record instead of CNAME. However this could lead to serious issues in case that the domain esnsystem.com changes it's IP address. In that case all the customers would have to change their A and AAAA DNS records too and it is very inconvenient. In case of CNAME record the customer does not even have to know, that there were some changes in server IP addresses.

4.5.1 Tailored platform using IaaS

Section 3.6.4 presents the analysis of services based on providing developers pre-configured platform. The good fit for PaaS is start-up company, which is planning to quickly generate high revenues. However ESN information system is based on voluntary program and PaaS solution would be too costly. Therefore tailored platform will be created on top of Infrastructure as a Service to achieve best fit for the purpose of ESN information system. The platform main components are:

- Debian 8 Operating System.
- Nginx web-server (front-end + reverse proxy).
- PHP7.0-FPM backend server.
- MariaDB datbase system.
- Composer php package management system.
- NodeJS - javascript server.
- Webpack module bundler.

Figure 4.13: **Tailored platform for ESN information system**



Source: own drawing

- NPM - Node Package Manager.
- Bash scripts.

The connections between the components is shown in figure 4.13

On top of this platform will be PaaS management tool. This management tool's main tasks will be managing server configuration, creating new instances os ESN information system, SSL certificate encryption and up-

dating/upgrading ESN information system to new versions. Development and design of this tool will be the next step after the information system is deployed and accepted.

4.6 Security

Section about security briefly describes the way, how to defend the server and application against attack attempts.

4.6.1 Application Security

In terms of application security the system relies on the robustness of security solution provided by Symfony framework. Symfony implements 2 layers of security:

1. Authentication.
2. Authorization.

The process of login is shown in figure 4.14. The process of Authentication is first security level and Symfony asks: 'Who is the user and does he has permission to log in the system".

In the second level - the authorization - symfony verifies, if the user has required roles (acces rights) to access requested resources.

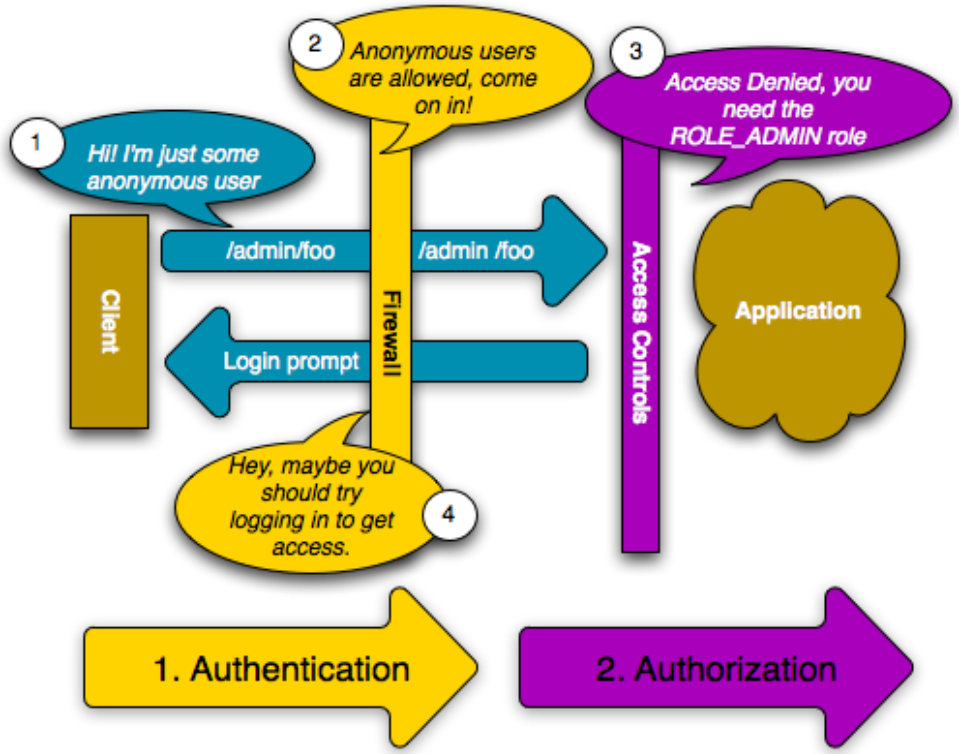
4.6.2 Server Security

To create safe environment for the designed platform (see section 4.5.1), it is necessary to implement multiple countermeasures against possible attacks.

The first countermeasure is firewall. Generally speaking, *"firewalls are network devices which enforce an organization's security policy."*[15]. It is vital to configure set of rules to allow or deny passage of network traffic. The strategy of setting firewall is to deny all traffic by default and then allow ports which are used by the server. For the purpose of ESN Infromation system following TCP/UDP ports are open:

1. 22 - SSH
2. 80 - http.
3. 443 - https.

Figure 4.14: **Authentication and Authorization using Symfony firewalls.**



Source: http://symfony.com/doc/2.0/book/security.html

4. 110 - Post Office Protocol (POP3)
5. 995 - Post Office Protocol 3 over TLS/SSL (POP3S)
6. 143 - Internet Message Access Protocol (IMAP).
7. 993 - Internet Message Access Protocol over TLS/SSL (IMAPS).
8. 873 - rsync file synchronization protocol.

No other ports are necessary. FTP access is not enabled and MariaDB database server on port 3306 can be accessed only on localhost or via SSH tunnel.

However any open ports always bring certain degree of threat to the server. Very common type of attack against open SSH ports (22) is brute-

force attack. According to Paar[20] the goal of brute-force attack is to *"simply decrypts the first piece of ciphertext with all possible keys"*. *"all possible keys"* is the important part of the brute-force definition. Attackers simply try all possible combinations of keys to decipher encrypted message. In case of brute force attack however the attacker does not have the piece of ciphertext. Instead attacker can access the interface on the SSH port and try to login using different combinations of user-name and password. To defend from this attack the server uses software *Fail2ban*. *"Fail2ban is a log-based brute force blocker."*[22] It tracks the access logs */var/log/auth.log* and based on the configurations put bans on ip addresses which were source of false login attempts. To not cut innocent users permanently from the service, the banned IP addresses are released (unbaned) after certain period of time. In the listing below part of /var/log/fail2ban.log file shows how brute force robots try to crack the password, evidently not detecting, that their IP address was banned. It is banned for 1 hour and if the attack continues after 3 failed attempts to login the IP address is banned again:

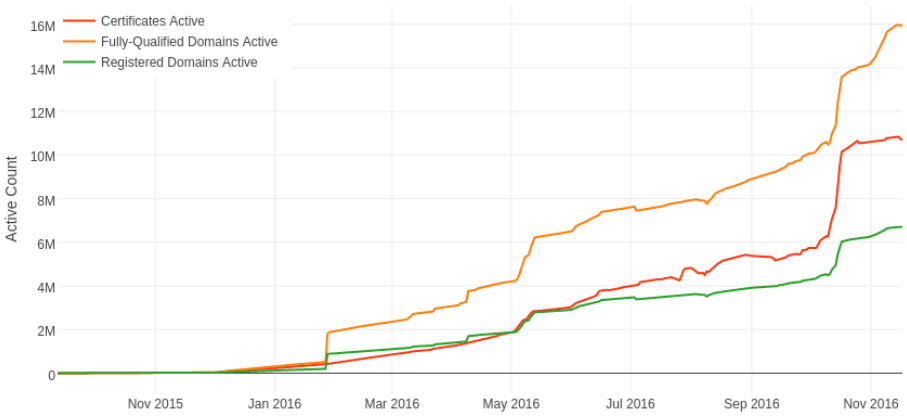
```
2016-11-20 16:19:25,712 fail2ban.actions[612]: WARNING
   [ssh] Ban 221.229.172.104
2016-11-20 17:19:26,131 fail2ban.actions[612]: WARNING
   [ssh] Unban 221.229.172.104
2016-11-20 17:20:59,229 fail2ban.actions[612]: WARNING
   [ssh] Ban 221.229.172.104
2016-11-20 18:20:59,699 fail2ban.actions[612]: WARNING
   [ssh] Unban 221.229.172.104
```

Nevertheless the best defend against this type of attack is to deny SSH authentication via password. Instead of password SSH can be also authenticated via private/public key pair.

4.6.3 SSL encryption - HTTPS

In order to raise the credibility and trust in the ESN information system SSL certificates will be used to provide the users with encrypted connection through HTTPS protocol. Each customer of ESN information system will have different domain name or sub-domain. In November 2015 emerged new Certification Authority which is the first in the world CA providing free SSL certificates. "Let's Encrypt is a free, automated, and open Certificate Authority."[1] Figure 4.15 shows how popular has this service become in 2015. In order to automate the process of configuration and obtaining certificates the bash scripts will be integrated with Nginx configurations.

Figure 4.15: **Usage of free certification authority LetsEncrypt.**



Source: https://letsencrypt.org/stats/

Conclusion

The main goal of this thesis was to analyse and propose a solution for ESN information system. First half of theoretical part was dedicated mainly to needs analysis and current situation analysis of ESN. The second half was dedicated to research technologies in order to build tailored platform for the needs of ESN information system.

The theoretical part is directly followed by the practical part which focus mainly on the design of ESN information system. It contains Front-end design, Use Case diagrams, Data class model and the tailored platform for deployment in the form of SaaS. Practical part also discusses the specific requirements and propose a solutions in order to solve them. Important section of practical part describes security precaution against the most common types of attacks which will be implemented on the server and application side.

The multi-tenant SaaS solution proposed in this thesis brings many advantages over the standard self-maintained solutions which require installation, configuration and constant maintenance.

Even though the core business processes for supporting ESN activities were implemented in the design, there is a large potential in further platform extension. This thesis was the first step in the future vision of universal customizable service compliant with the European laws.

The first version of ESN information system is already implemented and currently is used by more than 10 universities in Czech Republic, Slovakia, Russia, Greece and more countries and universities are expected to join.[23]

The next step will be design and implementation of mass mailing system, to better approach the registered users, custom questionnaires in order to better understand the needs of the international students and automatic

[23]https://uni-buddy.com

buddy matching, which would evaluate the best match based on preset attributes of each buddy and student.

In overall conclusion this thesis brings the overview of modern cloud technologies and implements design of system which helps raise the prestige of ESN and the university which is using it. For me personally this work helped me to deepen the knowledge of cloud computing problematic and it's different possible ways of solving the common problem.

Bibliography

[1] *About Let's Encrypt*. [online] [cit. 2016-11-3]. Available at WWW: <https://letsencrypt.org/about/>

[2] *Document Stores*. [online] [cit. 2016-10-14]. Available at WWW: <http://db-engines.com/en/article/Document+Stores>

[3] *How to Set external Parameters in the Service Container*. [online] [cit. 2016-11-14]. Available at WWW: <http://symfony.com/doc/2.8/configuration/external_parameters.html>

[4] Arlow, J.; Neustadt, I.: *UML 2 a unifikovaný proces vývoje aplikací*. Computer Press, a.s., 2011, ISBN 978-80-251-1503-9.

[5] Baun, C.; Kunze, M.; Nimis, J.; etc.: *Cloud Computing: Web-Based Dynamic IT Services*. Springer, 2011, ISBN 9783642209161.

[6] Britton, C.: *PHP: OPcache Introduction*. [online] [cit. 2016-10-14]. Available at WWW: <http://php.net/manual/en/intro.opcache.php>

[7] Britton, C.: *Choosing a Programming Language*. 2008, [online] [cit. 2016-11-02]. Available at WWW: <https://msdn.microsoft.com/en-us/library/cc168615.aspx>

[8] Brown, D. M.: *Communicating Design: Developing Web Site Documentation for Design and Planning (2nd Edition) (Voices That Matter)*. New Riders, second edition, 9 2010, ISBN 9780321712462.

[9] Cashmore, P.: *Why 2013 Is the Year of Responsive Web Design*. [online] [cit. 2016-11-03]. Available at WWW: <http://mashable.com/2012/12/11/responsive-web-design/>

[10] Eeles, P.: *Capturing Architectural Requirements*. 2005, [online] [cit. 2016-11-02]. Available at WWW: <http://www.ibm.com/developerworks/rational/library/4706.html>

[11] Elz, R.; Bush, R.: *Clarifications to the DNS Specification*. [online] [cit. 2016-10-20]. Available at WWW: <https://tools.ietf.org/html/rfc2181>

[12] esn.org: *Erasmus Student Network*. [online] [cit. 2016-11-06]. Available at WWW: <https://esn.org/about>

[13] Fowler: *UML Distilled 3/e*. Pearson, 2007, ISBN 8131715655.

[14] Hanjura, A.: *Heroku Cloud Application Development*. Packt Publishing - ebooks Account, 5 2014, ISBN 9781783550975.

[15] Ingham, K.; Forrest, S.: *A History and Survey of Network Firewalls*. [online] [cit. 2016-11-2]. Available at WWW: <http://www.cs.unm.edu/~treport/tr/02-12/firewall.pdf>

[16] *Country Codes - ISO 3166*. [online] [cit. 2016-08-26]. Available at WWW: <http://www.iso.org/iso/country_codes>

[17] Judith Hurwitz, F. H. D. K., Marcia Kaufman: *MULTI-TENANCY AND ITS BENEFITS IN A SAAS CLOUD COMPUTING ENVIRONMENT*. [online] [cit. 2016-11-05]. Available at WWW: <http://www.dummies.com/programming/cloud-computing/hybrid-cloud/multi-tenancy-and-its-benefits-in-a-saas-cloud-computing-environment>

[18] Kay, R.: *System Development Life Cycle*. 2002, [online] [cit. 2016-10-14]. Available at WWW: <http://www.computerworld.com/article/2576450/app-development/app-development-system-development-life-cycle.html>

[19] Marcotte, E.: *Responsive Web Design*. 2010, [online] [cit. 2016-11-05]. Available at WWW: <http://alistapart.com/article/responsive-web-design>

[20] Paar, C.; Pelzl, J.: *Understanding Cryptography: A Textbook for Students and Practitioners*. Springer, 2009, ISBN 978-3-642-04100-6.

[21] Riehle, D.: *Framework Design: A Role Modeling Approach*. Dissertation thesis, ETH Zürich, 2000.

[22] Schroder, C.: *Tip of the Trade: Fail2Ban*. 2006, [online] [cit. 2016-11-2]. Available at WWW: <http://www.serverwatch.com/tutorials/article.php/3626541>

[23] SensioLabs: *Symfony - The Book, Version: 2.8*. SensioLabs, 2016.

[24] Skvorc, B.: *The Most Popular Framework of 2015*. 2015, [online] [cit. 2016-10-14]. Available at WWW: <https://www.sitepoint.com/best-php-framework-2015-sitepoint-survey-results/>

[25] W3Techs: *Usage of server-side programming languages for websites*. [online] [cit. 2016-11-09]. Available at WWW: <https://w3techs.com/technologies/overview/programming_language/all>

Appendix A

List of abbreviations

A IP version 4 DNS record

AAAA IP version 6 DNS record

CNAME Canonical name (alias DNS record)

CRM Customer relationship management

CULS Czech University of Life Sciences

DB Database

DNS Domaine name system

ESN Erasmus Student Network

FTP File transfer protocol

HDD Hard disk drive

HR Human resources

HRM Human resource management

IaaS Infrastructure as a Service

IP Internet protocol address

ISO International Organization for Standardization

IT Information technology

LAMP Linux - Apache - MySQL - PHP

MVC Model - View - Controller

NPM Node package manager

ORM Object-relational mapping

PaaS Platform as a Service

PR Public relations

RAD Rapid application development

RDBMS Relational database management system

SaaS Software as a Service

SDLC Systems development life cycle

SSH Secure shell

SSL Secure Sockets Layer

TCP Transmission Control Protocol

UDP User Datagram Protocol

UML The Unified Modeling Language

VLAN Virtual Local Area Network

WAMP Windows - Apache - MySQL - PHP

Appendix B

Consent to processing personal data

In compliance with provisions of Section 5 paragraph 2 of the Act no. 101/2000 Coll., on Protection of Personal Data, as amended (hereinafter the "Act on Protection of Personal Data")

I hereby agree with processing of my personal data contained in the form used for registration into uni-buddy system and all the other personal data I entered to uni-buddy system (e.g. in section "About") by controller unighters s.r.o, identification number: 05509858, with its registered seat at Pod Bání 21, Prague 8, 180 00, Czech Republic, an association registered in the register administered by the Municipal Court in Prague, section C entry 264594 (hereinafter as "Controller") for the purpose of registration into uni-buddy system and participating in the so-called "Buddy programme", connecting registered users with local students via uni-buddy system and related services, such as sending invitations to social events to the registered users.

The consent is provided for hand-over of this personal data to other entities and users using uni-buddy system and/or participating in so-called "Buddy programme" and that includes hand-over to such entities and users in the European Union, Switzerland, Norway, Iceland and Russian Federation.

The Controller will process personal data both manually and automatically, through the mediation of its members, employees and processors authorized by the Controller, especially an association vpsFree.cz, z.s., identification number: 26568055, with its registered seat at Nad Dalejským údolím 2699/9, Stodůlky, 155 00 Prague, Czech Republic, registered in the register administered by the Municipal Court in Prague, section L entry

19890.

The Controller processes the personal data of the data subjects only in the scope that is necessary for the purpose stated above. The Controller also stores personal data only for a period necessary for the purpose of their processing and removes all the personal data from all the systems and records as soon as the processing is no longer necessary.

The Controller is obliged to make every reasonable effort to prevent disclosure of the personal data and other private information of the data subjects. Summarized data contained in the registration forms may be used by the Controller for Controller's internal statistical purposes.

This consent to personal data processing is granted for 5 years and applies to all the information provided in the form used for registration into uni-buddy system and all the other information entered into uni-buddy system.

I affirm that I am aware of my rights granted by provisions of Sections 12 and 21 of the Act on Protection of Personal Data, especially that refusing to provide this consent will not cause any negative consequences for my person, and that in case I discover or presume the Controller is carrying out processing of my personal data which is in contradiction with the protection of my private and personal life or which is in contradiction with law, I am entitled particularly to ask the Controller for explanation or require the Controller to remedy the arisen state of affairs.

I state that all the personal data provided in registration form are accurate and truthful and that I provided all the personal data voluntarily.

Appendix C

The Service Agreement

The Service Agreement

Concluding to date

unighters s.r.o.
Based: Pod Bání 21, Praha 8, 180 00, Czech Republic
Identification number: 055 09 858
Bank account: 2301090889/2010
IBAN: CZ6720100000002301090889
BIC: FIOBCZPPXXX
An association registered in the register administered by the Municipal Court in Prague, section C entry 264594.

Represented by [ ]

(hereinafter as „**Contractor**“)

- as the first Party -

and

[ ]
Vysoká škola / Zapsaný spolek / Fyzická osoba

(hereinafter as „**Acquirer**“)

- as the second Party -

In compliance with provisions of paragraph 2358 and paragraph 2371 of the Act no. 89/2012 Sb., on Civil code
(hereinafter as „**Civil code**“)
and
provisions of paragraph 12 of the Act no. 121/2000 Sb., on copyright and rights related to copyright and amending some laws (Copyright Act), as amended (hereinafter as "**CA**"),

this

Service Agreement

(hereinafter as „**Agreement**“)

1 SUBJECT MATTER OF THE AGREEMENT

1.1 Subject matter of this agreement is provision of trial usage of the service uni-buddy by Acquirer (hereinafter as "**Service**"). This trial mode doesn't guarantee any functionality, nor perfection, and serves only for presentation of the Service and its features.

2 CREATION OF CLOUD INSTANCE

2.1 Service will be made available to Acquirer by providing the access data to the instance created using internet interface.

2.1.1 The Acquirer undertakes to obtain all hardware, software or other resources required to access to the Service. Preferred internet browser is Google Chrome (Chromium). The Contractor shall provide the Acquirer the access credentials

2.1.2 The Acquirer confirms, that the Service will not be provided on their own hardware resources, but will be provided by the Contractor, under „software as a service".

3 USE OF THE WORK AND LICENCE CONDITIONS

3.1 The Service includes the provision of a specific computer program (hereinafter as „**Work**"). The Provider provides the Acquirer of the right to use the work only as to the extent and under the conditions specified in this Agreement (hereinafter as „**Licence**"). The scope of the Licence is defined in the attachment no. 2 of this Agreement.

3.2 Unless the following provisions determine otherwise

3.2.1 The Contractor provides the Acquirer a nonexclusive Licence;

3.2.2 The Acquirer is entitled to use the Work exclusively for evaluation purposes of the Service, in the scope of so-called „Buddy programme", and related activities (particularly, connection of the international students with the students of the university that the Acquirer is associated with);

3.2.3 The Acquirer is neither entitled to provide, nor entitled to transfer the Licence, under the provision of this Agreement, to a third party;

3.2.4 The Acquirer is not entitled to intervene the Work, to edit the Work in any way, to join it with other Work, to put it in a collective work, or finish the unfinished Work (even using a third party);

3.3 The Acquirer is not entitled to make copies of the Work, unless it is a legal licence or free use of the work which can not be excluded by the Agreement.

3.4 The Licence is granted for an unlimited territory and for the effective period of this Agreement.

3.5 The Licence is provided free of charge.

4 TRIAL MODE

4.1 Acquirer confirms that the Contractor shall not be liable for damages in connection with its performance of this Agreement, because it is a trial version. Acquirer uses the service entirely at their own risk and responsibility and waives all claims, especially of compensation or liability for defects, which are incurred by the use of the Service against the Contractor. The Acquirer is obliged to notify about the trial mode all persons, whom, in accordance with this Agreement, the Service will be made available.

4.2 The Contractor shall not be liable for any potential Acquirer’s problems with using of the Service, particularly as a result of malfunction or failure of Acquirer's device or improper of the Service.

4.3 The Acquirer shall not enter any “live data” to the Service, especially not any personal information, but only fictitious test data.

4.4 The Acquirer is obliged to take appropriate measures to protect their data. The Contractor is not liable for data loss and their restoration, if this loss could have been prevented by fulfilling the legal obligations under this provision.

4.5 The Acquirer undertakes to compensate the Contractor for any damage caused as a result of violation of this article.

5 DURATION OF THE CONTRACT

5.1 This Agreement is concluded for a fixed period until [ ].

6 INFORMATION PROTECTION

6.1 Contracting Parties are aware that in fulfillment of the Agreement, they may mutually, both intentionally or by omission, provide information that would be considered confidential. Unless otherwise decided in written form by the Contracting Parties, all the information that is or might be a part of a trade secret is considered as confidential, ie., for example, information on operating methods, procedures and workflows, business or marketing plans, concepts and strategies (or their parts), offers, contracts, agreements, treaties or other arrangements with third parties, information on financial results, relationships with business partners on labor issues and all other information whose disclosure by the receiving Party could cause damage to the transmitting Party (hereinafter as "**Confidential information**")

6.2 All the Confidential Information shall remain the sole property of the transmitting Party. The receiving Party shall endeavor the same effort to maintain the confidentiality of such information and to protect it as if it were its own confidential information. Except to the extent that is necessary for cooperation in the performance of this Agreement, both Parties undertake not to make copies of any Confidential Information of the other Contracting Party nor to disclose it to third parties or to their own employees and representatives, with the exception of those who need to be familiarised with them in order to fulfill this Agreement. Both Parties also undertake not to use the Confidential Information of the other Party in any manner other than for the purposes of fulfilling this Agreement.

6.3 If the Confidential information is being provided in written form or in the form of text files on computer media, then the transmitting Contracting Party is obliged to notify the receiving Party about the confidentiality of such material at least on the front page of the material.

6.4 Notwithstanding the above mentioned provisions, as Confidential information shall not be considered information which:

6.4.1 has become publicly known without the intention or by omission of the receiving Contracting Party;

6.4.2 the receiving Contracting Party has had legally at their disposal before concluding this Agreement, if such information was not the subject to a previously concluded agreement on information protection between both Parties;

6.4.3 is the result of a process during which the receiving Contracting Party reached the information independently and is able to prove it using its own records or using Confidential information of a third party;

6.4.4 is provided to the receiving Party by a third party after the signing of the Agreement. Such information shall not be obtained directly or indirectly from the Contracting Party, which is its owner.

6.5 Paragraphs 6.4.2 and 6.4.4 of this Agreement shall not apply to information that is subject to a trade secret between the Contracting Parties, especially information related to the business scope of either of the Contracting Parties, internal and external work management, etc. After the termination of this Agreement, the Contracting Party may disclose in writing (through an explicit list) to the other party, which information is no longer subject to the rules of this paragraph.

6.6 During archivation of the Confidential Information, the Contracting Parties are required to act in a manner that ensures that this information is not obtained by any third party that is not involved in the performance of this Agreement. If the Contracting Party is no longer in need of any Confidential Information, it is obligated to properly dispose of any such information, so that it is not accessible to any third party.

6.7 The Contracting Parties may use ideas, concepts and know-how relating to their business activities, that their employees obtain during the performance of this Agreement and which remain in the memory of those employees, solely assuming that these ideas, concepts or know-how are from the field of information technology and which in effect (and in utilization) have no direct relationship to the communicated Confidential Information.

6.8 The provisions of this Article shall not be affected by the expiration of this Agreement for any reason, and its effectiveness will not end earlier than two years after termination of this Agreement.

6.9 For each violation of the provisions regarding the protection of information, the injured Contracting Party has the right to require a contractual penalty from the other Contracting Party in the amount of [ ].

7 COMMUNICATION BETWEEN THE CONTRACTING PARTIES

7.1 All communication between the Contracting Parties will be realized through authorized persons:

7.1.1 The authorized person on behalf of the Contractor is Mikuláš Josek, email: mikulas.josek@uni-buddy.com,

7.1.2 The authorized person on behalf of the Acquirer is [ ] email [ ].

7.2 All notices between the Contracting Parties relating to this Agreement or to be made under this Agreement shall be made in written form and delivered to the other Contracting Party.

7.3 Such a document (notice) that would be delivered under this Agreement to the other Party shall be considered as delivered on the date when it is received by the qualified person in accordance to paragraphs 7.1.1 and 7.1.2 of this Agreement or by a person authorized to represent the Party in accordance to the commercial registration or an employee authorized to receive documents. In case of doubts, it is considered that the person who confirmed the receipt of the document with their signature and the stamp of the Contracting Party is entitled to receive such documents.

7.4 If the delivery of a document under the provision of the preceding paragraph fails, as the day of delivery shall be considered the day when the receipt of this document was refused by the addressee. If delivered by post to their own hands to the address specified in this Agreement (or to the address that the Contracting Party has notified in case of a change of address), as the day of delivery shall be considered the third day of notification of the shipment at the post office (even if the addressee did not learn about the delivery), or the day when the shipment was sent back as undeliverable because the Contracting Party changed its registered office; after the termination of the Contract, this fiction is valid only if the document was also sent to the headquarters stated in a public register.

7.5 The effects of delivery of the document may also occur by e-mail to the listed addresses of the persons authorized to act on behalf of the Contracting Parties in accordance with paragraph 7.1. hereof.

7.6 The Contracting Parties undertake that in the event of a change of address or of authorized persons, they shall notify the other Contracting Party about this change no later than three (3) working days.

8 SEVERABILITY CLAUSE

8.1 If any provision of this Agreement or its part is invalid or unenforceable or may become invalid or unenforceable in the future, this invalidity or unenforceability shall not affect the validity or enforceability of the remaining provisions of this Agreement or parts thereof, if it is not apparent from the content of this Agreement, that the provision or parts thereof are inseparable from other content.

8.2 In the case referred to in the preceding paragraph of this article, the Contracting Parties undertake to replace the ineffective or invalid provision by a new provision that is in its purpose and economic meaning closest to the provision that has to be replaced.

9 GENERAL AND FINAL PROVISIONS

9.1 The Contractor is entitled to disclose the performance of this Agreement with the Acquirer for their own presentation as a reference.

9.2 The Contracting Parties do not wish that any of the rights and obligations, beyond the explicit provisions of this Agreement, are inferred from existing or future practice established between the Contracting parties or from practices maintained in general or maintained in the sector related to the subject matter of this Agreement. In addition, the Contracting Parties confirm that they are not aware of any already established business customs and practices among them.

9.3 The Contracting Parties exclude application of the following provisions of the Civil Code to this Agreement: paragraph 557 (contra proferentem rule).

9.4 The Contracting Parties explicitly confirm that the basic provisions of this Agreement are the result of negotiations between the parties, and that each party had the opportunity to influence the content of the basic provisions of this Agreement. To eliminate any doubts, the Contracting parties exclude application of the provisions of paragraph 1799 and paragraph 1800 of the Civil Code (clauses in adhesion contracts). The Contracting Parties declare that there is not any weaker party.

9.5 Changes and additions to this Agreement can be made only by written and consecutively numbered additions, signed by both parties.

9.6 This Agreement supersedes all previous written or oral agreements or arrangements relating to the subject matter hereof.

9.7 This Agreement is governed by Czech law.

9.8 This Agreement is executed in two copies, of which each Contracting Party shall receive one.

9.9 An integral part of this Agreement are the following annexes:

9.9.1 Annex no. 1 – Service

9.9.2 Annex no. 2 – Licence

9.10 The Contracting Parties have read the Agreement, agree with all of its contents, and prove this by attaching their signatures

In [.....] on [....]

_____

Contractor

[ ]

In [.....] on [....]

_____

Acquirer

[ ]

Annex n. 1
Service

Providing uni-buddy system as a service includes:

1) Contractor creates a cloud instance of the information system for the Acquirer
2) Contractor makes this instance available to the Acquirer through access data,
3) Contractor allows the Acquirer to use the functions of the instance (see below)
4) Contractor performs maintenance of the instance and security updates.

The service is provided as SaaS (Software as a Service), i.e. without installation on hardware devices belonging to the Acquirer. The service is accessed through the internet (web browsers) by the Acquirer's and user's devices (PC, tablet, smartphone ...).

System features:

Registration of local students (buddies, mentors, tutors) and international students

- An interface for registration of international and local students (buddies / mentors / tutors)
- Possibility to fill in personal information, to request a buddy and to specify the date of arrival
- Real-time overview of the registration process thanks to system notifications and automatic emails

Buddy pairing

- A chronological list of international students who requested a buddy (ordered by time of arrival or registration)
- Selection of international students (mentees) by local students (mentors) based on anonymous profiles of international students (only gender, country and voluntarily disclosed interests are visible)
- Contact information is exchanged automatically, but only after students are paired

Student data management

- Administration for Buddy Programme Coordinators and staff of the department of international relations (IRO officers) for user management and the pairing process of students.
- Standard data management functions (filter, sort, edit and export the data to Excel)
- Automatically generated statistics (overviews by gender, countries, study programmes, faculties)
- Automatic archiving of inactive users registered as Buddies after one year of inactivity.

Options for customization

- Administrator interface for settings and for allocation of system roles (administrator, buddy coordinator, event manager and student).

- Customizable inputs (setting the date of beginning of the semester, uploading logos and the name of the university, inserting of a contact email, linking to social networking sites such as Facebook, etc.)
- Customizable disclaimers and reminders for students (important information for students during registration)
- Customizable access to the instance (the possibility of the access being limited to certain students or faculty)

Annex n. 2
Licence

The license is granted to the extent that the Provider provides the Acquirer with the access data to the service of uni-buddy via an Internet interface in the extent of "Administrator".

Administrators have the authority to assign other permissible access roles in the extent of "administrator", "buddy coordinator" and "event manager".

The number of permissible accesses and allocation of roles is not restricted.

We recommend that the Service is made available to the local student association Erasmus Student Network (ESN section) present at the Acquirer's institution. The President, HR manager or Buddy Coordinator from the ESN section should gain access to and the role of "Administrator".

The content of the authorization of the the roles is defined by the permitted functions that are made available within the interface according to the relevant role.