EDMS Requirements Specification

Prepared by:

1. *Klevis Xhyra*
2. *Kristian Pashollari*
3. *Semih Zengindemir*
4. *Aldo Lisi*
5. *Nail Spahija*

Table of Contents

1. EXECUTIVE SUMMARY 3

1.1 Project Overview 3
1.2 Purpose and Scope of this Specification 3

2. PRODUCT/SERVICE DESCRIPTION 3

2.1 Product Context 3
2.2 User Characteristics 3
2.3 Assumptions 3
2.4 Constraints 3
2.5 Dependencies 4

3. REQUIREMENTS 4

3.1 Functional Requirements 5
3.2 Non-Functional Requirements 5
3.2.1 User Interface Requirements 5
3.2.2 Usability 5
3.2.3 Performance 6
3.2.4 Manageability/Maintainability 6
3.2.5 System Interface/Integration 7
3.2.6 Security 8
3.2.7 Data Management 8
3.2.8 Standards Compliance 8
3.2.9 Portability 8
3.2.10 Other Non-Functional Requirements 9
3.3 Domain Requirements 9

4. USER SCENARIOS/USE CASES 9

APPENDIX 10

Appendix A. Definitions, Acronyms, and Abbreviations 10
Appendix B. References 10
Appendix C. Requirements Traceability Matrix 10
Appendix D. Organizing the Requirements 12

1. Executive Summary

1.1 Project Overview

Nowadays, technology has advanced and most of the system are controlled on real time. The idea of our project is to reduce the time and to make the management of the dorm more easier. Every problem, technical or disciplinal is going to be managed and on real time. The dormitory, does not permit the entrance of the students which are not part of it, except for the ones which have a permission. For that problem, our project manages it by using a face recognition at dorm entrance.

1.2 Purpose and Scope of this Specification

The purpose of our project is to support the dormitory with the software which is going to make the life of the students and the staff more easier. Everyone will be informed on real time, and for every problem they are going to learn in the fastest way. Our software is going to manage every structure who has some rights on the dorm. Taking permission, in order to stay on the dormitory, the permission are going to be checked by three administrative staff: assistants , dormitory director, and the dorm manager.

2. Product/Service Description

EDMS is a web application which is aimed to improve the performance and to facilitate the life of students in Epoka Dormitory. As we as residents have seen many minor problems and latencies of some processes the application will be valuable for all the staff and students.

The platform will be used by Epoka Dormitory and it aims to facilitate the services offered by dormitory in the future. The application will address some of the most emerging problems of the building or said differently it will improve the facilities the services offered by the dormitory as listed below:

- Student applications (as a resident or as a visitor)
- Food menu-s and attendees in the weekends
- Technical problems
- Dormitory services evaluations by students
- Events happening and a short info about dormitory
- Entrance security (not to let non-residents to enter inside)

Our software's conception is a web application based on object oriented PHP and later it can easily be implemented in an application running in any mobile Android or iPhone.

2.1 Product Context

Our software is directly related to the Epoka Dormitory. The software is designed to be an independent system and later it may be included in the personal EIS profile of each student registered in the dormitory. The system will be available to four main genres main users divided in four categories : staff, students, master admins.

2.2 User Characteristics

Our software will have different roles of using it and the roles are assigned to 7 categories of users that are listed as below:

- ➔ **GUEST**
- ➔ **STUDENT**
- ➔ **ASSISTANT**
- ➔ **DIRECTOR**
- ➔ **TECHNICAL STAFF**
- ➔ **FINANCE**
- ➔ **MASTER ADMIN**

Also the competences for each role are determined according to the users.

➔ **Guest**

The guest role is given to every male student who studies at Epoka University. The interface and the capabilities of a guest will be simple and pretty straight forward. Using his Epoka mail he may apply to the appropriate section to stay at dormitory for a specified time.

➔ **Student**

As this platform is mainly built for facilitating the everyday activities for a student this role will have a lot of options and sections. All student which are part for the dormitory will be able to report a problem and this problem will be addressed to resident assistants and director. According to the report the problem is addressed to technical staff where they take their responsibility to fix the broken thing or it may be a problem that can be fixed by the resident assistants. Also the student will be able to request for any other necessary thing that can be offered by the dormitory like blankets etc. He can also view payments that he did and has to do and also a history of all of them.

➔ **Director**

Director will be as a second admin for the platform. The Director capabilities are as following :

- **Assign assistants to students**
- **Give permissions to guests (Add/Remove guests)**
- **Add/Remove Students**
- **Send students to discipline**
- **Review technical problems**
- **Give permissions to students**

Also the director will have the ability to announce meetings or inform students for different events happening around the ambient.

➔ Technical Staff

Technical will receive a list of issues at the beginning of the day or depending on the reports. There will be a simple page where the list can be exported in WORD/EXCEL/PDF. Then after an issue is solved there will be some checkboxes that will inform Students and other personnel for the fix.

➔ Resident Assistant

The assistants will be determined by the master admin which control all the dormitory whose capabilities are beyond the director. Assistants will have capabilities as following:

- **Report an assistant/student to directory**
- **Report technical issues**
- **Evaluate the dorm services**
- **See the current list of students**
- **Confirm request from students regarding his responsibilities**

➔ Finance

The finance personnel that will be connected to the platform will have control about the payments of the dormitory. During the approval of the payment the director will be notified in real time. There will be an dashboard where they can see the status of all of the students showing all of their recent and history payments.

➔ Master Admin

The master admin competences will be given to the head of the management of the university. The Master Admin dashboard will have additional options and he is in charge of all the system and have all the competences. In the dashboard will be included all the functionalities and the master admin will have the right to view, edit and add student information. He/She also will have the right to manage all the activities happening on the dormitory. In short the project will be in control of Master Admin.

2.3 *Assumptions*

It is assumed that some actions performed behind the scenes are performed regularly according to the university and dormitory regulation and will be listed below.

Therefore the Master Admin together Director according to "Vendimit te Keshillit të Ministrave Nr.281 date 12.03.2008" they have to check a list of documents which include id card, 4-photos , certificate from university, payment receipt and the application form. After reviewing these documents provided they decide and consider for the applicant eligible for staying in dormitory or not.

Also it is assumed that according to the Law nr.7850, date 29.07.1994 "Kodi Civil i Republikes se Shqiperise" ndryshuar me ligjin nr.8536 date 18.10.1999 for the legislation in the field of higher education both University and student have agreed the conditions predefined by the university and dormitory regulation stating that the university offers quality conditions for accommodating the students for what they pay stated in the 3rd section of the contract.

For security issues, it is taken for granted that after master admini assigns the student to the dormitory, he send confidentially by mail the random generated password. This procedure is performed by school organs who provide the email addresses to students attending the first year. It is assumed that all the students have an personal school email address.

It is also assumed that the user responsible for accepting students that in this case is Director and Master Admin confirms the contract after notifying both sides respectively.

It is also assumed that the payments are firstly confirmed to the finance office and until the time the students hasn't paid the fee he isn't allowed to enter the system and after one week he can't also enter the dormitory.

It is also assumed that for confirming the attendance the system will communicate with the security part of the project and will automatically confirm the attendance when needed.

It is assumed that the students are registered in school.

It is assumed that all the data for the students is correct.

It is assumed that the students have their faces scanned when they apply to stay in dormitory which will be used by the entrance system.

It is assumed that all the students will have a device connected with internet.

It is assumed that every event happening in the dormitory is logged into the system.

2.3 Constraints

The project is constrained by the Internet connection. Having a stable Internet connection is crucial for the system to work since the application fetches data from the database over the Internet. It is also crucial to have a device where the system will run and process the solutions of the problems. Also for the entrance system the system is constrained by the cameras that will be used for the security. The cameras need to be connected to the internet also and be always turned on.

2.4 Dependencies

List dependencies that affect the requirements are as following :

- Normally everyone can view the introduction to Dormitory and can apply as a visitor without providing any information because all of them will be automatically taken by Epoka University only from his email. A guest can apply also for registration in Dormitory as a student. Both applications are reviewed by Director. For a simple visitor Director can approve or not his accommodation and besides that he will assign the room he will stay and the assistant who will look after him. If the guest is applying for registration in dormitory Director reviews the application form and then requests an approval from Finance (about the payment) and the Master admin (someone from the school). If the guest is approved to be a resident in Dormitory automatically director has the right to print the contract and both of them can sign. Of course from the system Director can assign his assistant and his room.

- A new facility called entrance security will be added for a better security to dormitory students. With face recognition (face ID) the system will identify all students and open the door to them. If a new visitor comes the system sends a notification to all assistants and Director. If one of them knows him and if the visitor have permission to enter the Dorm the door will be opened. Otherwise his photo will be saved in the Database
- The entrance system code has to be done before we start the the developing of the site.

- Every assistant have the opportunity to report any technical issue in the dormitory and once a week these issues are sent as a .doc or .xlsx file to the head of technical staff. After these problems are fixed any of assistant may check as repaired all fixed problems.

- Same as EISAPP system we thought that would be helpful a service evaluation as it gives a huge help to improvement of all facilities. All services will be graded from students and then those information will anonymously be sent to Director and Master admin.

As it can be seen from the explanation given above there is a dependency between the level of users that in same time represent different positions in school and dormitory.

3. Requirements

- Describe all system requirements in enough detail for designers to design a system satisfying the requirements and testers to verify that the system satisfies requirements.
- Organize these requirements in a way that works best for your project. See Appendix DAppendix D, Organizing the Requirements for different ways to organize these requirements.
- Describe every input into the system, every output from the system, and every function performed by the system in response to an input or in support of an output. (Specify what functions are to be performed on what data to produce what results at what location for whom.)

- Each requirement should be numbered (or uniquely identifiable) and prioritized.
  See the sample requirements in Functional Requirements, and System Interface/Integration, as well as these example priority definitions:

  Priority Definitions

The following definitions are intended as a guideline to prioritize requirements.

- Priority 1 – The requirement is a "must have" as outlined by policy/law
- Priority 2 – The requirement is needed for improved processing, and the fulfillment of the requirement will create immediate benefits
- Priority 3 – The requirement is a "nice to have" which may include new functionality

It may be helpful to phrase the requirement in terms of its priority, e.g., "The value of the employee status sent to DIS **must be** either A or I" or "It **would be nice** if the application warned the user that the expiration date was 3 business days away". Another approach would be to group requirements by priority category.

- A good requirement is:
  - Correct
  - Unambiguous (all statements have exactly one interpretation)
  - Complete (where TBDs are absolutely necessary, document why the information is unknown, who is responsible for resolution, and the deadline)
  - Consistent
  - Ranked for importance and/or stability
  - Verifiable (avoid soft descriptions like "works well", "is user friendly"; use concrete terms and specify measurable quantities)
  - Modifiable (evolve the Requirements Specification only via a formal change process, preserving a complete audit trail of changes)
  - Does not specify any particular design
  - Traceable (cross-reference with source documents and spawned documents).

3.1 Functional Requirements

Req#	Requirement	Comments	Priority	Date Rvwd
1	System should offer different interface and functionality to different kind of users	Each user type (Role) must enter into his/her own page which will be different from another type of user	1	30/03/2018
2	System should deliver emails to proper users for all types of applications	long term applications must be delivered to Master admin, Finance and Director whether short term applications must be delivered only to Director	2	30/03/2018
3	System should deliver emails to all residents about every kind of activity that will be held in dormitory	for example: it will send an email to all residents that in 2 days there is a cocktail for everyone in the main hall	3	30/03/2018
4	System should be able to retrieve every information of students living in dormitory	Personal informations written in the contract in the beginning	1	30/03/2018

5	All logs of students who have been in dormitory must be saved in the DB.	If whenever something happens or something is required from the last years, system must be able to show who was in the dormitory and in what period	2	30/03/2018
6	All users (residents) must have a unque ID	ID will be provided from University-s DB. It will be taken (queried) from the users epoka email	1	30/03/2018
7	All users must have their own interface (page)	according to the users credentials, the application will direct them to their own interface (according to position)	1	31/03/2018
8	All personal pages (interfaces) must be restricted with a password	Every user must enter his/her credentials and the password	1	31/03/2018
9	System must have 6 type of users	Student, assistant, director, technician, finance, Master admin	1	31/03/2018
10	System should be able to give access to different persons to enter the dormitory (security camera)	Give access to new students or visitors and remove access manually or automatically to the persons who have finished their time in dormitory	2	31/03/2018
11	System should handle technical and discipline reports	system should give permission to report technical issues and discipline problems to only some users	1	31/03/2018
12	System should produce contracts for all approved long term applications	if a student is approved as a resident system should print his contract with all needed data-s	2	31/03/2018
13	System should support at least twice a year a ranking form delivered to all students	twice a year students are required to rank all facilities provided to them	3	31/03/2018

3.2 Non-Functional Requirements

3.2.1 User Interface Requirements

The user interface is compatible with any device. It supports Chrome, Mozilla, Safari IOS and Android. It is also very simple, user such as Guests will only be able to see the main screen and they will be able to make requests only by using their email.

Other users at first will face the login page, after entering the credentials and the login is successful, they can proceed to their own interface. Students can make requests and can see their payments. The director will have more access on the page, his main duties will be editing students and guests.

The technical staff will have a much simpler page, they can see the reports exported in a pdf/word/excel file. For finance's users the page is also simple, they will be able to control payments.

The user with full control is Master Admin whose interface includes all of the above access and additional competences.

3.2.2 Learnability

The application has a user friendly interface so anybody without much computer skills can use it. It is very straight forward, the users simply login by using their emails from Epoka University and can view everything they have access in.

3.2.2.2 User Friendliness

Accessing data and making requests for different groups of users is beginner friendly so anybody can reach the data easily. Most of the requests are handled by forms which everyone is familiar with.

3.2.2.3 Error Tolerance

The application will be able to detect errors and ask the user to correct them. This will be used mostly in log in page and different forms. First the user will be shown with important fields that should be filled and if there is something wrong, the system will make a suggestion about the specific user input.

3.2.2.4 Accessibility

Since the interface is user friendly everything it is easy to be accessed by different users. Each group has its own page with different panels. Each of the actions will be placed in visible places so they can be accessed easily.

3.2.3 Performance

The performance of the application will depend mostly on the servers. Also its users should have performance internet in order to prevent any late response. Device users will not have any performance lag as the application is compatible with their browsers, but on other users the performance will also depend on their hardware. Also the application will not require more than 20 percent of the processor capacity.

3.2.4 Manageability/Maintainability

3.2.4.1 Monitoring

Even though the system will use a secure connection with the server (https) , there can be cases that the system will be down or it can be attacked by malicious users to steal data. Everything will be kept in a log file which can be accessed by the team and check what caused the system for an unexpected shut down or malicious entry.

3.2.4.2 Maintenance

Our team will be able to maintain the application and update it with the changes required. Even in any cases when our team is not able to maintain the application, we will provide a backup of the application until the team takes care of any problem that may have caused the application to be down. For maintenance the system shall not shut down more than 12 hours.

3.2.4.3 Operations

Operations required by the user include:

- login with their Epoka’s emails
- apply for accommodation
- make requests for different problems
- view payments
- CRUD for director
- view the issues
- approve payments
- CRUD for Master Admin

3.2.5 Security

3.2.5.1 Protection

The system will be resistant from any unauthorized, accidental or unintended usage and provide access only to legitimate users. It ensures that every data that will pass through the database has the highest security so no data will be lost or captured from malicious users.

3.2.6 Data Management

The required requirements for data management includes:

- data will be used frequently to query information about students
- only the Director and Master admin is able to view these data
- the student's data should be correct
- there will be no initial value, since information should be correct

3.2.7 Other Non-Functional Requirements

3.2.8.1 Access Security

- users must change their default password immediately after the first log in
- the payment category is restricted, so only the appropriate staff can use it
- the access restrictions can only be changed by the master admin
- password should never be viewable in the login page
- each incorrect login try will be recorded in the log file

3.2.8.2 Configuration

With the system will be one or more cameras connected. So the security will be higher but also a better and a faster way to access the dormitory. The cameras should be fast enough to capture the data so the students will not have to wait for the camera to response.

3.3 Domain Requirements

Everyone in this project has different management roles. End every role will be secured to access only it's own rights.

The students which apply for the dorm residence but at the end are not staying on the dorm. Their application will be canceled, and deleted from the database. No need for unused data.

Also, there are going to be students, which like to stay on the dormitory for a short period of time. And they should have access to enter on the dorm only for that period of time.

For the moment, on the dormitory exist the entrance by using the student cards, but because of the system, many students complains about that. There were registered but the system is not accepting them, so the dorm guardian needs to open the door for them.

The student, in order to be accepted, he should be accepted by the dormitory management saff and then the finance gives the last hand, whether the student made the dormitory payment or not.

4. User Scenarios/Use Cases

Provide a summary of the major functions that the product will perform. Organize the functions to be understandable to the customer or a first time reader. Include use cases and business scenarios, or provide a link to a separate document (or documents). A business scenario:

- Describes a significant business need
- Identifies, documents, and ranks the problem that is driving the scenario
- Describes the business and technical environment that will resolve the problem
- States the desired objectives
- Shows the “Actors” and where they fit in the business model
- Is specific, and measurable, and uses clear metrics for success

In here you may define the written user scenarios tested in the UCED Application given to you.

5.

6.

APPENDIX

The appendixes are not always considered part of the actual Requirements Specification and are not always necessary. They may include

- Sample input/output formats, descriptions of cost analysis studies, or results of user surveys;
- Supporting or background information that can help the readers of the Requirements Specification;
- A description of the problems to be solved by the system;
- Special packaging instructions for the code and the media to meet security, export, initial loading, or other requirements.

When appendixes are included, the Requirements Specification should explicitly state whether or not the appendixes are to be considered part of the requirements.

Appendix A. Definitions, Acronyms, and Abbreviations

Define all terms, acronyms, and abbreviations used in this document.

Appendix B. References

List all the documents and other materials referenced in this document.

Appendix C. Requirements Traceability Matrix

The following trace matrix examples show one possible use of naming standards for deliverables (FunctionalArea-DocType-NN). The number has no other meaning than to keep the documents unique. For example, the Bargaining Unit Assignment Process Flow would be BUA-PF-01.

For example (1):

Business Requirement	Area	Deliverables	Status
BR_LR_01 The system should validate the relationship between Bargaining Unit/Location and Job Class.---Comments: Business Process = "Assigning a Bargaining Unit to an Appointment" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
		BUA-PF-01 Derive Bargaining Unit-Process Flow Diagram	Accepted
BR_LR_09 The system should provide the capability for the Labor Relations Office to maintain the job class/union relationship.---Comments: Business Process = "Maintenance" (Priority 1)	BUA	BUA-CD-01 Assign BU Conceptual Design	Accepted
		BUA-PF-02 BU Assignment Rules Maint Process Flow Diagram	ReadyForReview

For example (2):

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
BR_LR_01	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_01	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted
BR_LR_01	1	BUA	BUA-PF-01	Derive Bargaining Unit-Process Flow Diagram	Accepted
BR_LR_01	1	BUA	BUA-UCD-01	BU Assign LR UseCase Diagram	ReadyForReview

BR_LR_01	1	BUA	BUA-UCT-001	BU Assignment by PC UseCase - Add Appointment and Derive UBU	Reviewed
BR_LR_01	1	BUA	BUA-UCT-002	BU Assignment by PC UseCase - Add Appointment (UBU Not Found)	Reviewed
BR_LR_01	1	BUA	BUA-UCT-006	BU Assignment by PC UseCase - Modify Appointment (Removed UBU)	Reviewed
BR_LR_09	1	BUA	BUA-CD-01	Assign BU Conceptual Design	Accepted
BR_LR_09	1	BUA	BUA-DS-02	Bargaining Unit Assignment DB Modification Description	Accepted
BR_LR_09	1	BUA	BUA-PF-02	BU Assignment Rules Maint Process Flow Diagram	Accepted
BR_LR_09	1	BUA	BUA-UCD-03	BU Assign Rules Maint UseCase Diagram	Reviewed
BR_LR_09	1	BUA	BUA-UCT-045	BU Assignment Rules Maint: Successfully Add New Assignment Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-051	BU Assignment Rules MaintUseCase: Modify Rule	Reviewed
BR_LR_09	1	BUA	BUA-UCT-053	BU Assignment Rules MaintUseCase - Review Assignment Rules	Reviewed
BR_LR_09	1	BUA	BUA-UCT-057	BU Assignment Rules MaintUseCase: Inactivate Last Rule for a BU	Reviewed
BR_LR_09	1	BUA	BUA-UI-02	BU AssignRules Maint UI Mockups	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-021	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-027	BU Assignment Rules Maint TestCase: Modify Rule - Success	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-035	BU Assignment Rules Maint TestCase: Add New Rule (Associated Job Class Does Not Exist) - Error Condition	ReadyForReview
BR_LR_09	1	BUA	BUA-TC-049	BU Assignment Rules Maint TestCase: Modify Rule - Error Condition	ReadyForReview

For example (3):

BizReqID	CD01	CD02	CD03	CD04	UI01	UI02	UCT01	UCT02	UCT03	TC01	TC02	TC03	TC04
BR_LR_01			X		X		X			X		X	
BR_LR_09	X			X		X			X		X		X
BR_LR_10	X			X					X		X		
BR_LR_11		X											

Appendix D.

Organizing the Requirements

This section is for information only as an aid in preparing the requirements document.

Detailed requirements tend to be extensive. Give careful consideration to your organization scheme. Some examples of organization schemes are described below:

By System Mode
Some systems behave quite differently depending on the mode of operation. For example, a control system may have different sets of functions depending on its mode: training, normal, or emergency.

By User Class
Some systems provide different sets of functions to different classes of users. For example, an elevator control system presents different capabilities to passengers, maintenance workers, and fire fighters.

By Objects
Objects are real-world entities that have a counterpart within the system. For example, in a patient monitoring system, objects include patients, sensors, nurses, rooms, physicians, medicines, etc. Associated with each object is a set of attributes (of that object) and functions (performed by that object). These functions are also called services, methods, or processes. Note that sets of objects may share attributes and services. These are grouped together as classes.

By Feature
A feature is an externally desired service by the system that may require a sequence of inputs to affect the desired result. For example, in a telephone system, features include local call, call forwarding, and conference call. Each feature is generally described in a sequence of stimulus-response pairs, and may include validity checks on inputs, exact sequencing of operations, responses to abnormal situations, including error handling and recovery, effects of parameters, relationships of inputs to outputs, including input/output sequences and formulas for input to output.

By Stimulus
Some systems can be best organized by describing their functions in terms of stimuli. For example, the functions of an automatic aircraft landing system may be organized into sections for loss of power, wind shear, sudden change in roll, vertical velocity excessive, etc.

By Response
Some systems can be best organized by describing all the functions in support of the generation of a response. For example, the functions of a personnel system may be organized into sections corresponding to all functions associated with generating paychecks, all functions associated with generating a current list of employees, etc.

By Functional Hierarchy
When none of the above organizational schemes prove helpful, the overall functionality can be organized into a hierarchy of functions organized by common inputs, common outputs, or common internal data access. Data flow diagrams and data dictionaries can be used to show the relationships between and among the functions and data.

Additional Comments
Whenever a new Requirements Specification is contemplated, more than one of the organizational techniques given above may be appropriate. In such cases, organize the specific requirements for multiple hierarchies tailored to the specific needs of the system under specification.

There are many notations, methods, and automated support tools available to aid in the documentation of requirements. For the most part, their usefulness is a function of organization. For example, when organizing by mode, finite state machines or state charts may prove helpful; when organizing by object, object-oriented analysis may prove helpful; when organizing by feature, stimulus-response sequences may prove helpful; and when organizing by functional hierarchy, data flow diagrams and data dictionaries may prove helpful.