Epoka e-Learner Requirements Specification

Version 1.0

April 23, 2021

CEN 302 – Software Engineering

Faculty of Architecture and Engineering

Department of Computer Engineering

Sonia Memaj, Kevin Kurtaja, Nos Qirjako, Romina Hoxha, Marinela Alishollari, Ledia Leka

Table of Contents

1. Executive Summary

1.1 Project Overview

Having to study remotely has made the use of online learning platforms an absolute necessity. Epoka university offers well-organized structures for this purpose however since all of the school-related activities are carried out online some deficiencies are made evident. The biggest problem being the use of too many platforms across all courses. This including Google Classroom, Epoka LMS, EIS, and Epoka official site.
What this project aims, is to create an improved learning experience for Epoka University students, based on the already existing online platforms that the university uses. We intend on creating a mobile application to facilitate the student experience. Our application will offer a space where students can access all the necessary information during their studies and also create a communication channel between them and the professors. One of the goals is to make this interaction easier for a smoother learning experience.

1.2 Purpose and Scope of this Specification

The purpose of this project is to create a mobile application that will be an extension of Epoka Interactive System. This will provide students with the utilities that EIS offers. At the same time some of the information offered from Epoka's official site will be included. The students will also be able to access their courses and lectures which will be updated by the professors.

In scope:

- Creating a simple and efficient user experience for the users of this application (Students and Lecturers).
- Updating the legislation accordingly to guarantee the privacy and data safety of its users.
- The retreival of official documents from the university with the appropriate signatures, seal and specifications.

Out of Scope:

- Managing and having access to student data as they are considered confidential information.
- Creating an actual connection to EIS. The project will be working with static data which will be used as test data to check the quality and functionality of the application. So, an actual connection could only be made by the university officially once the project is complete.

2. Product/Service Description

2.1 Product Context

This product will be connected to EIS and Epoka LMS to get the necessary student information and the lectures for each course.

2.2 User Characteristics

Users include:

1. **Students:**
    - Views courses, lectures, grades, attendance, finances.
    - Gets a specific timetable view based on year, department, and group.
    - Can request documents.
    - Can view events or news relating to the university.

- Can select courses.
- Can make posts on their course page.
- Can view or request transportation (request was made available past year because of the pandemic, not sure if it will be a permanent feature).
- Can access the school library digital catalog.
- Can communicate with the professors through private comments.

2. Lecturers/Professors:

- Views and edits courses, lectures, attendance.
- Gets a specific timetable view.
- Can view events or news relating to the university.
- Can access the school library digital catalog.
- Can view or request transportation (request was made available past year because of the pandemic, not sure if it will be a permanent feature).
- Can communicate with the students through private comments.

3. Advisor:

- Has the same privileges as the professor.
- Can see more information about the students including their GPA.
- Can make course approvals.

All users will be logged in with Epoka email account. Lecturers who were once students will be logged in with their new Epoka email account and with this account will have lecture privileges. The advisor will be a professor with that same account, they will just be granted more privileges.

2.3 Assumptions

The application will need a persistent network connection. Users will also need a mobile device to run this application on. They need to be using either iOS or Android.

2.4 Constraints

- Mobile-based frameworks and programming languages.
- Security concerns. Considering there will be a lot of personal information on the users, security is a main priority.
- Memory space.

2.5 Dependencies

- Most of the updates will be made by the professors, so students will rely heavily on them for their grades, attendance etc.
- Other updates will be made by the university staff which will be represented in the database.
- We need a simple user interface(prototype) to start testing the connection.

3. Requirements

3.1 Functional Requirements

EL_GR => E-Learner General Requirements

EL_SR => E-Learner Student Requirements

EL_PR => E-Learner Professor Requirements

EL_AR => E-Learner Advisor Requirements

Req#	Requirement	Comments	Priority	Date Rvwd	SME Reviewed / Approved
EL_GR_01	Users have different privileges but same base view.	Business Process = “Maintenance	3	20/04/21	Bob Dylan, Mick Jagger
EL_GR_02	Users should be able to log in and log out, no sign up available.		1	20/04/21	
EL_GR_03	After leaving the app users should not be logged out.		1	20/04/21	
EL_GR_04	Each user has their own profile, they cannot edit their profile data.	Same data as their Epoka email account.	1	20/04/21	
EL_GR_05	Users can access emergency tab.		1	20/04/21	
EL_GR_06	Users can see all of the events which will be updated in real time.		3	20/04/21	
EL_GR_07	Users can access the full timetable, the first view they get will be personalized.		2	20/04/21	
EL_GR_08	Grade and GPA will be calculated automatically.		1	20/04/21	
EL_AR_01	Advisors can confirm course selection.		1	20/04/21	
EL_AR_02	Advisors can confirm students eligible for GP.		2	20/04/21	
EL_AR_03	Advisors cannot edit student’s information.		1	20/04/21	
EL_PR_01	Professors cannot create courses. Courses are created automatically with data from the course selection.		1	20/04/21	
EL_SR_01	Students can access only selected courses.		1	20/04/21	

3.2 Non-Functional Requirements

3.2.1 User Interface Requirements

In addition to functions required, describe the characteristics of each interface between the product and its users (e.g., required screen formats/organization, report layouts, menu structures, error and other messages, or function keys).

1. Firstly, there is the login in view containing fields for username and password, forgot password button and continue with Gmail button.
2. After logging in the opening view will be the same for each user. There are two options so far. Having a feed for the news, a bottom navigation bar and a sidebar menu which contains all of the necessary tabs (courses, attendance etc). Or having small box like components with all of that information. More information on the views will be given in the sketches.

3.2.2 Usability

- The documentation relating to each possible use of this application should be complete.
- The application should be easy to use.
- The application should be secure.

3.2.3 Performance

Specify static and dynamic numerical requirements placed on the system or on human interaction with the system:

- Static numerical requirements may include the number of terminals to be supported, the number of simultaneous users to be supported, and the amount and type of information to be handled.
- Dynamic numerical requirements may include the number of transactions and tasks and the amount of data to be processed within certain time period for both normal and peak workload conditions.

All of these requirements should be stated in measurable form. For example, "95% of the transactions shall be processed in less than 1 second" rather than “an operator shall not have to wait for the transaction to complete”.

3.2.3.1 Capacity

Include measurable capacity requirements (e.g., the number of simultaneous users to be supported, the maximum simultaneous user load, per-user memory requirements, expected application throughput)

3.2.3.2 Availability

Include specific and measurable requirements for:

- Hours of operation
- Level of availability required
- Coverage for geographic areas
- Impact of downtime on users and business operations
- Impact of scheduled and unscheduled maintenance on uptime and maintenance communications procedures
- reliability (e.g., acceptable mean time between failures (MTBF), or the maximum permitted number of failures per hour).

3.2.3.3 Latency

Include explicit latency requirements, e.g., the maximum acceptable time (or average time) for a service request.

3.2.4 Manageability/Maintainability

3.2.4.1 Monitoring

Include any requirements for product or service health monitoring, failure conditions, error detection, logging, and correction.

3.2.4.2 Maintenance

Specify attributes of the system that relate to ease of maintenance. These requirements may relate to modularity, complexity, or interface design. Requirements should not be placed here simply because they are thought to be good design practices.

3.2.4.3 Operations

Specify any normal and special operations required by the user, including:

- periods of interactive operations and periods of unattended operations
- data processing support functions
- backup and recovery operations
- safety considerations and requirements
- disaster recovery and business resumption

3.2.5 System Interface/Integration

Specify the use of other required products (e.g., a database or operating system), and interfaces with other systems (e.g., UWHires package interfaces with PubCookie and ODS, HEPPS system interfaces with Budget system). For each interface, define the interface in terms of message format and content. For well-documented interfaces, simply provide a reference to the documentation.

Outline each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

3.2.5.1 Network and Hardware Interfaces

Specify the logical characteristics of each interface between the product and the hardware or network components of the system. This includes configuration characteristics (e.g., number of ports, instruction sets), what devices are to be supported, and protocols (e.g., signal handshake protocols).

3.2.5.2 Systems Interfaces

Example systems interface requirements:

A. System1-to-System2 Interface

The <external party> will create and send a fixed length text file as an email attachment to System2mail@u.washington.edu to be imported into the System2 system for payroll calculation. This file must be received on EDIT day by 4:00 PM in order to be processed in the EDIT night run. The requirements below document the file specifications, data transfer process, and specific schedule. This file is referred to as "FileName" in this document.

File Structure and Format

A1. The FileName file is a fixed length text file.

A2. The FileName file is an unformatted ASCII file (text-only).

A3. The FileName file contains a batch totals record and several detail records.

File Description: Batch Totals Record

A4. The batch totals record can be placed at the beginning, in the middle, or at the end of the file.

A5. The batch totals record contains the following:

- Record Type (value: XA)
- Process Type (value: A)

- Batch Number (3 digit number assigned by Payroll Dept)
- Origin Code (AIG)
- Total number of detail records
- Total deduction amount

File Description: Detail Records

A6. The FileName file contains a row for each record meeting xxx criteria.

A7. Each row in the FileName file contains the following fields, comma-delimited and encased in double-quotes where the data includes commas or spaces:

- Employee Id
- Record Type
- Process Date (MMDDYY)
- XYG Number
- Element Code
- Amount
- Amount Sign
- Year Flag
- Total Amount
- Total Amt Sign

3.2.6 Security

3.2.6.1 Protection

Specify the factors that will protect the system from malicious or accidental access, modification, disclosure, destruction, or misuse. For example:

- encryption
- activity logging, historical data sets
- restrictions on intermodule communications
- data integrity checks

3.2.6.2 Authorization and Authentication

Specify the Authorization and Authentication factors. Consider using standard tools such as PubCookie.

3.2.7 Data Management

Specify the requirements for any information that is to be placed into a database, including

- types of information used by various functions
- frequency of use
- data access rules
- data entities and relationships
- integrity constraints
- data retention
- valid range, accuracy, and/or tolerance
- units of measure
- data formats
- default or initial values

3.2.8 Standards Compliance

Specify the requirements derived from existing standards, policies, regulations, or laws (e.g., report format, data naming, accounting procedures, audit tracing). For example, this could specify the requirement for software to trace processing activity. Such traces are needed for some applications to meet minimum regulatory or financial standards. An audit trace requirement may, for example, state that all changes to a payroll database must be recorded in a trace file with before and after values.

3.2.9 Portability

If portability is a requirement, specify attributes of the system that relate to the ease of porting the system to other host machines and/or operating systems. For example,

- Percentage of components with host-dependent code;
- Percentage of code that is host dependent;
- Use of a proven portable language;
- Use of a particular compiler or language subset;
- Use of a particular operating system;
- The need for environment-independence - the product must operate the same regardless of operating systems, networks, development or production environments.

3.2.10 Other Non-Functional Requirements

Please provide all necessary non-functional requirements, similar to the requirements explained in the lesson slides or in the textbook.

3.3 Domain Requirements

Everything related to the domain that might be needed in the project shall be mentioned in here. Sometimes the domain Requirements might be thought as part of either functional or non-functional requirements.

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

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

BizReqID	Pri	Major Area	DevTstItems DelivID	Deliv Name	Status
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.