Bachelor Thesis

# SOFTWARE SYSTEM FOR THE FINANCIAL MANAGEMENT OF ERASMUS+ STRATEGIC PARTNERSHIPS PROJECTS

**Maria Anastasiou**

**UNIVERSITY OF CYPRUS**

**DEPARTMENT OF COMPUTER SCIENCE**

**May 2021**

# UNIVERSITY OF CYPRUS
# DEPARTMENT OF COMPUTER SCIENCE

Software system for the financial management of Erasmus+ strategic partnerships projects

**Maria Anastasiou**

Supervising Professor
George Papadopoulos

Co-Supervisor
Christos Mettouris
Evangelia Vanezi

The Diploma Thesis was submitted for partial fulfilment of the requirements for obtaining a degree in Computer Science from the Department of Computer Science of the University of Cyprus

May 2021

## Acknowledgements

With the help of God, I managed to complete this work. Throughout the writing of this thesis, I have received a plethora of support and assistance. I would first like to express my thanks to my supervisor, Professor George Papadopoulos. Furthermore, I would like to thank the Researchers Evangelia Vanezi and Christo Mettouri at the Software Engineering and Internet Technologies (SEIT) Laboratory at the Computer Science department of the University of Cyprus for the excellent cooperation we have had throughout this thesis. Their advice and guidelines helped me to complete this work. Besides, I would also like to thank my parents for their help and their love. Finally, I could not have achieved all of this without the support of my friends. They helped me by listening to my problems and by providing happy distractions to rest my mind.

## Summary

For the purposes of this dissertation, we have implemented a platform for the financial management of Erasmus+ projects. On this platform, employees can complete their timesheets, see the projects and work packages they work on, and for each of them, keep track of their budget. Administrators have their own menu that allows them to execute additional functionalities. They can add new projects and work packages, perceive all the timesheets and contracts, or change the rates and the hours.

# CONTENT


*INTRODUCTION* ..... 1

1.1 MOTIVATION ..... 1
1.2 METHODOLOGY ..... 1
1.3 CONTENTS OUTLINE ..... 2

*RELATED WORK* ..... 4

2.1 INTRODUCTION ..... 4
2.2 OTHER SIMILAR TOOLS ..... 5
*1. TimeCamp ..... 5*
*2. Harvest ..... 5*
*3. Zoho Projects ..... 6*
*4. Toggl ..... 6*
*5. ClickTime ..... 7*
*6. TickSpot ..... 7*
*7. Hubstaff ..... 7*
*8. Timesheets.com ..... 8*
*9. Clockify ..... 8*
*10. Replicon ..... 9*
2.3 CHARACTERISTICS ..... 9
2.4 CONCLUSION ..... 11

*METHODOLOGY* ..... 12

3.1 INTRODUCTION ..... 12
3.2 SOFTWARE DEVELOPMENT LIFE CYCLE ..... 13
*3.2.1 What is Software Development Life Cycle? ..... 13*
*3.2.2 Software Development Process Models ..... 13*
*3.2.3 How to choose a development model ..... 29*
*3.2.4 What we chose? ..... 30*
3.3 QUESTIONNAIRE ..... 30
*3.3.1 Steps ..... 30*
*3.3.2 Questions ..... 31*
*3.3.3 Responses ..... 32*
3.4 INTERVIEW ..... 32
*3.4.1 Why Interview? ..... 32*
*3.4.2 Interview questions ..... 33*
*3.4.3 Interview responses ..... 33*
3.5 REQUIREMENTS ..... 34
*3.5.1 Steps ..... 34*
*3.5.2 List of Requirements ..... 34*

*IMPLEMENTATION* ..... 37

4.1 WORDPRESS ..... 37
4.2 DATABASE ..... 38
4.3 METAPHORS ..... 40
4.4 VISUAL DESIGN CONSTRAINTS ..... 41
4.5 USABILITY HEURISTICS ..... 41
4.6 PROTOTYPING ..... 43
*4.6.1 Low fidelity prototyping ..... 43*
*4.6.2 Medium fidelity prototype ..... 44*

*DEMONSTRATION* ..... 46

5.1 PERSONAS ..... 46

5.2 SCENARIOS ..... 48
*5.2.1 New employee ..... 48*
*5.2.2 New project ..... 48*
5.3 USER MANUAL ..... 49
*5.3.1 Login ..... 49*
*5.3.2 Register ..... 49*
*5.3.3 Forgot Password ..... 50*
*5.3.4 Menu ..... 50*
*5.3.5 Dashboard ..... 50*
*5.3.6 Projects ..... 51*
*5.3.7 Work Packages ..... 51*
*5.3.8 Work package hours ..... 51*
*5.3.9 Budget ..... 51*
*5.3.10 Contracts ..... 52*
*5.3.11 New Contract ..... 52*
*5.3.12 Timesheets ..... 53*
*5.3.13 Edit Timesheet ..... 54*
*5.3.14 New Timesheet ..... 54*
*5.3.15 Admin Panel ..... 54*

*EVALUATION* ..... 59

6.1 INTRODUCTION ..... 59
6.2 QUESTIONNAIRE ..... 59
*6.2.1 Steps ..... 59*
*6.2.2 Questions ..... 60*
*6.2.3 Results ..... 60*
6.3 INTERVIEW ..... 61

*CONCLUSION* ..... 63

7.1 CONCLUSION ..... 63
7.2. FUTURE WORK ..... 63
BIBLIOGRAPHY ..... 65

# FIGURES

*INTRODUCTION* ..... 1

*RELATED WORK* ..... 4

FIGURE 2.1 ..... 10
FIGURE 2.2 ..... 10

*METHODOLOGY* ..... 12

FIGURE 3.1 ..... 13
FIGURE 3.2 - THE WATERFALL MODEL ..... 17
FIGURE 3.3 – V-MODEL ..... 18
FIGURE 3.4 – INCREMENTAL DEVELOPMENT MODEL ..... 22
FIGURE 3.5 – PROTOTYPING DEVELOPMENT MODEL ..... 24
FIGURE 3.6 - EXTREME PROGRAMMING ..... 27
FIGURE 3.7 – SPIRAL MODEL ..... 28
FIGURE 3.8 ..... 30
FIGURE 3.9 ..... 31
FIGURE 3.10 ..... 34

*IMPLEMENTATION* ..... 37

FIGURE 4.1 – CMS TECHNOLOGIES ..... 38
FIGURE 4.2 – DATABASE SCHEMA ..... 40
FIGURE 4.3 –ER DIAGRAM ..... 40
FIGURE 4.4 - METAPHORS ..... 41
FIGURE 4.5 – CONSTRAINTS ..... 41
FIGURE 4.6 – LOW FIDELITY PROTOTYPE ..... 44
FIGURE 4.7– MEDIUM FIDELITY PROTOTYPES ..... 45

*DEMONSTRATION* ..... 46

FIGURE 5.1 – PERSONAS ..... 47
FIGURE 5.2 – NEW CONTRACT ..... 53
FIGURE 5.3 – NEW TIMESHEET ..... 54
FIGURE 5.4 – ASSIGN ..... 56

*EVALUATION* ..... 59

FIGURE 6.1 ..... 61

*CONCLUSION* ..... 63

# CHAPTER 1

## INTRODUCTION

CONTENT

INTRODUCTION ..... 1
1.1 MOTIVATION ..... 1
1.2 METHODOLOGY ..... 1
1.3 CONTENTS OUTLINE ..... 2

### 1.1 Motivation

Many researchers work on Erasmus+ research projects, but they do not have a specific platform or application to complete their timesheets and track their projects. A reason for this could be that other applications and platforms offering project management functionalities are a bit difficult to learn or are not specialized in meeting their needs. Many different platforms assist their users with project management tasks, but to the best of our knowledge, none is a good fit for the Erasmus+ type of projects. This is because researchers need to handle specific, predefined budget information and templates. Additionally, it is not efficient to have a lot of extra functionalities that they will eventually not use. Researchers need to have an easy-to-use and simple platform where they will find everything they need in just one place.

Currently, researchers mostly use Excel for timesheets and several different tools for project management. With that in mind, we have designed and developed this platform for their use.

### 1.2 Methodology

The methodology we followed consists of five steps:

1. We researched, found, and examined related applications and platforms/tools to determine what they offer. We analyzed and compared the various functions provided by them, found those that we may need, and got an idea of what already exists.
2. A questionnaire was implemented that was given to researchers and the public to find people with experience in Erasmus+ projects. The questionnaire's objective was to understand what exactly researchers want and need this platform to offer. After analyzing the results, a group interview was conducted with two researchers where feedback was given about the system's interface and functionalities.
3. We extracted the requirements from both the questionnaire and a meeting that we had with two researchers.
4. The most significant phase was the implementation. We started this phase by designing and building the database with the required tables and the columns. Then, we decided about the interface, and, last but not least, we created and tested the various functionalities.
5. When everything was ready, we sent the link to the researchers and they accessed it to try it and learn it and then we received their feedback.

## *1.3 Contents Outline*

### CHAPTER 1 - INTRODUCTION

In the introduction chapter, we briefly present the motivation that we had to develop this platform and who will be using it. Moreover, we analyze the methodology used to achieve this bachelor thesis. The first chapter is a general introduction to the topics that are going to be discussed in depth later. It includes the goals and motivation for my thesis, the methodology followed, and the structure of the rest bachelor thesis.

### CHAPTER 2 - RELATED WORK

In the second chapter, we study and analyze other work done on similar topics. We found a number of platforms and applications that have some of the functions

we are looking for, and we explain how their work is related or different from ours and why researchers need our platform and not something that already exists.

### CHAPTER 3 – METHODOLOGY

The methodology chapter defines our methodology and explains each step we took. First, we present what model we chose for software engineering and why. We describe how we got the requirements and how we use the information we acquired from the questionnaire.

### *CHAPTER 4 – IMPLEMENTATION*

In the implementation chapter, we are going to show the software design. Additionally, we will have an ER diagram for the database and explain the tables and columns. Flow data diagram and workflow chart will be there as well. In general, it will be the chapter that will describe everything about the design and the implementation.

### *CHAPTER 5 – DEMONSTRATION*

This fifth chapter discusses how this platform will be used by the users and have a user manual to help them find everything they need without any difficulty. It explains what an employee can or cannot do based on his/her role in a more pragmatic way.

### *CHAPTER 6 - EVALUATION*

In the evaluation chapter, we discuss the way we evaluate our results. We get feedback from some researchers who worked on this platform and add their projects and their timesheets.

### *CHAPTER 7 - CONCLUSION*

This final chapter briefly describes what other functions could be added in the future and some final though

# *CHAPTER 2*

## ***RELATED WORK***

---

**CONTENT**


*RELATED WORK* ..... 4

2.1 INTRODUCTION 4
2.2 OTHER SIMILAR TOOLS 5
*1. TimeCamp 5*
*2. Harvest 5*
*3. Zoho Projects 6*
*4. Toggl 6*
*5. ClickTime 7*
*6. TickSpot 7*
*7. Hubstaff 7*
*8. Timesheets.com 8*
*9. Clockify 8*
*10. Replicon 9*
2.3 CHARACTERISTICS 9
2.4 CONCLUSION 11


---

## *2.1 Introduction*

There are many time-tracking or budgeting tools and applications. Some of them are listed below, along with some characteristics explained in subchapter characteristics, shown in Figure 2.1. Nevertheless, none of them is the best option for Erasmus+ Projects because many of them have features that researchers do not need to use (for example, the features in Figure 2.2), and they do not have some functionalities that researchers need. Researchers want to have an easy-to-use application with all the characteristics they want in one place so they will do their job more efficiently. Unused components that researchers need to know how they work, but do not find helpful for their job, make an application's learning time increase. This is one reason why, at present, some researchers do not use any platform or application for their financial management.

Today, the researchers we worked with are completing their timesheets in a spreadsheet, and they have their projects and work packages in another sheet. Applications and platforms like ours or the ones below allow them to have all their projects and work packages in one place and in a more orderly way. With these applications, they can visualize many things and get a more general idea of what is going on.

## *2.2 Other similar tools*

In this subchapter, we are going to see a few platforms and applications that have similar features to those that we are looking for:

1. TimeCamp

TimeCamp is a platform that helps the users to track their team's performance, their project's profitability, and their productivity. It provides automatic time tracking, which automatically completes the timesheets, which users can edit or delete later. It also helps you with productivity tracking by tracking how much time they were in each application and platform while working or by setting up some goals. With TimeCamp, the users can also export some reports about various topics like projects, etc. Additionally, if they need it, they can have some customizable bill rates and smart invoicing with adjustable tax rates. As for the budget, users can define project costs and timespan budgets, track their performance, and get notifications when a project is about to overrun its time or cost budgets. Finally, timesheets can be approved by the manager, and after the approval, timesheets are locked, and no one can change them.

2. Harvest

Harvest is a simple time-tracking application. With Harvest, the users can track their time automatically, and their timesheets will be completed by themselves. It

also has reminders to help the users to track their time. Another feature is a wide selection of visual reports that help support the team and keep projects running smoothly. Project reports allow users to stay on budget while team reports visualize the team's capacity so they can adjust their workload accordingly. Harvest also has the feature for the manager to approve the timesheets. Invoicing page can generate invoices automatically and email them to the clients with few clicks. It supports PayPal and Stripe.

3. Zoho Projects

Zoho Projects is a cloud-based project management software that helps its users to plan their projects, track work efficiently, and collaborate with their team, wherever they are. With Zoho Projects, each user can manage both simple and complex projects by breaking them down into milestones, task lists, tasks, and subtasks and then visualize them. Additionally, they can assign tasks to other users, set work hours, and track them as they are finished. With the help of Gaant diagrams, users can plan their projects thoroughly and track progress closely. Also, they can set up dependencies, identify critical tasks, and create baselines at regular intervals so they can stay on track and get projects completed on time. Zoho Project has a timer that auto-completes the timesheets, and then the manager must approve them. With the Zoho Books or Zoho Invoice integration, Projects helps record their expenses, set up a budget for their project, and turn their timesheets into invoices for client billing and payroll purposes. Another feature that Zoho Projects has is the various reports extracted from timesheets based on the project or the user.

4. Toggl

Toggl is a simple platform that helps users with time and project management. It can help them get a visual overview of everything that's happening without spending hours keeping things up to date. Toggl Plan is fast and flexible so that

when the users' plans change, they can quickly get back to work. With Toggl, users can assign work to their team to make sure that everyone is working. It can also track time both online and offline to have a more complete and correct timesheet.

5. ClickTime

Click time is an application that can track, manage, and plan employee time with ease. With ClickTime, users can complete their timesheets, keep track of their expenses, track their time, etc. Another feature is that you can request some time off, and the manager can approve or deny their request. With this application, users can also have resource planning both in terms of time and people. They can also easily see which projects, activities, clients, or employees are going over budget and create some budget alerts to prevent overservicing and overspending.

6. TickSpot

TickSpot is a straightforward time tracking software and mobile app that helps its users to track their time and budget. It has some in-app timers that allow them to track time for their timesheets and some instant budget feedback for the project they are working on at that time. Additionally, administrators can look at all the projects and how the budget is spent between each task or person. Users can also set a specific budget for each job, so it will be easier to know if they are over-budget. Moreover, the administrator can assign the employees to the projects.

7. Hubstaff

Hubstaff's time tracking app is an easy way to keep track of time. Hubstaff offers time tracking, budgeting, online timesheets, and other things that researchers are looking for, but at the same time, it has a lot of other features too. Hubstaff provides its users with Geofencing and GPS tracking as well as employee monitoring.

## 8. Timesheets.com

It is a time tracking software for teams. It helps with timesheets for employees in the office or working from home. It is mainly used for small or medium-sized businesses. With timesheets.com, the users can complete their timesheets online either automatically, with the online timer, or manually. Timesheets.com offers many ways to get the time for billing, job costing, and productivity. Each user can have a customizable permission setting. Via this platform, they can also track the employees who work in the field and know their location. Another feature is that if a user is an administrator, he/she can check for time off conflicts between workers and approve requests from the calendar screen or the employee's timesheet. Timesheets.com also offers powerful reporting features. Reports can be filtered by employee, customer, or job so that the users will know precisely their costs.

## 9. Clockify

Clockify is a simple time tracker and timesheet app that lets its users and their team track work hours across projects. They can track their time for each project and task manually or automatically with the timer and choose if it is billable. As for the timesheets, they can select the activity and then complete the hours they worked on each day. It offers them the possibility to see all the week together and the task they have done each day. Another feature Clockify has is the calendar, which visualizes every day, and the users can add or edit their block hours accordingly. It also has a dashboard to see the team's latest activity and who is currently working. With Clockify, the users have the option to download some reports or send them via email to someone else. Finally, they can see the projects and the team members, and if they are the administrator, they can edit rates, roles, etc.

10. Replicon

Replicon is an All-in-One Project Timesheet Software. With Replicon, users can track the time they spent on each activity, project, or task. They can also have some pre-populated timesheets records (for example, the previous month's timesheet). Some other features it has are GPS monitoring and time off scheduling. Replicon has notifications to remind its users about timesheets or approvals if you are the administrator. This application allows them to add projects and specify their budget, their milestones, etc. Lastly, it has an advanced analytics engine that enables the proper information access to the right people based on their organization's hierarchy and controls.

## *2.3 Characteristics*

In this subchapter, we present the features that most of the above applications and platforms have. We choose those features because it was the most common ones and those we believed might be useful for our web tool.

**Auto tracker:** It is a personal timer that helps you count the exact hours you worked on each project.

**Offline time tracker:** It is exactly like the auto tracker, but it works with or without the internet.

**Budgeting:** It is a section where you can see the actual spending in relation to the budgeted amount.

**Invoices:** It is a section where you can list the products and services your team provides to a client and establish an obligation on the client to pay your team for those products and services.

**Project dashboard:** It is a screen where all the projects and due dates are displayed.

**Absence analysis:** It is a screen where you can apply for leave days and see if they are approved.

**Computer using tracking:** It tracks the amount of RAM, storage, etc., used, the platforms visited, and the time employees spent on the computer. The system will keep this information, and the project manager can have access to this.

**Timeline chart:** It shows the calendar and the team's tasks at a specific time.

**Google synchronization**

**Financial reports:** They are reports that show how each project's budget is consumed through time and personnel.

*Figure 2.1*

| | Autotracker | Budgeting | Invoicing | Offline Time Tracking | Project Dashboard | Absence Analysis | Attendance | Computer Usage Tracking | Timeline | Google Calendar integration | Reports |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1. TimeCamp | YES | YES | YES | YES | YES | YES | YES | YES | YES | YES | YES |
| 2. Harvest | YES | YES | YES | YES | YES | NO | NO | NO | NO | NO | YES |
| 3. Zoho Projects | YES | YES | YES | NO | YES | NO | NO | NO | YES | NO | YES |
| 4. Toggl | YES | YES | YES | YES | YES | NO | NO | NO | YES | NO | YES |
| 5. ClickTime | YES | YES | YES | YES | YES | YES | YES | YES | NO | YES | YES |
| 6. TickSpot | YES | YES | YES | NO | YES | NO | NO | NO | NO | NO | YES |
| 7. Hubstaff | YES | YES | YES | YES | YES | NO | NO | YES | NO | NO | YES |
| 8. Timesheets.com | YES | YES | NO | NO | YES | YES | YES | NO | NO | NO | YES |
| 9. Clockify | YES | YES | NO | NO | YES | NO | NO | NO | NO | NO | YES |
| 10. Replicon | YES | YES | YES | NO | YES | YES | YES | YES | NO | NO | YES |

*Figure 2.2*

## *2.4 Conclusion*

Our platform needs budget information and report, a dashboard, timesheets, and an administrator panel, along with some basic features like view or add projects, work packages, and contracts. As described above, many applications and platforms have the functions we are looking for in our web tool. One essential feature that they do not have that will be easier for the researchers if they have it is the limitation in daily and monthly hours on timesheets. On our platform, we would like the administrator to define the daily and monthly hours that employees can complete for their timesheets. If someone is going to be over that hours on a daily basis, or over the budget of the project in general, or he/she is over the hours in his/her contract, there must be a notification, and the timesheet must not be completed.

Additionally, we need our platform not to allow employees to complete timesheets on weekends or national days. With these limitations, the administrator will not have to accept or deny a timesheet, giving researchers more time for work. Another significant issue is money. None of the above applications and platforms does all these things for free. All of those applications are proprietary, so researchers need to pay to get all the functions they require.

Many online applications with similar features have been recognized, but either they do not provide all the functionality that we need or provide additional complex functionality. Additionally, researchers need to pay for many of them to get some of the functionalities that they need. As we can see, researchers may need some different features that are not be offered by those tools, so we only got an idea of what already exists out there to create our own platform.

# *CHAPTER 3*

## *METHODOLOGY*

## CONTENT

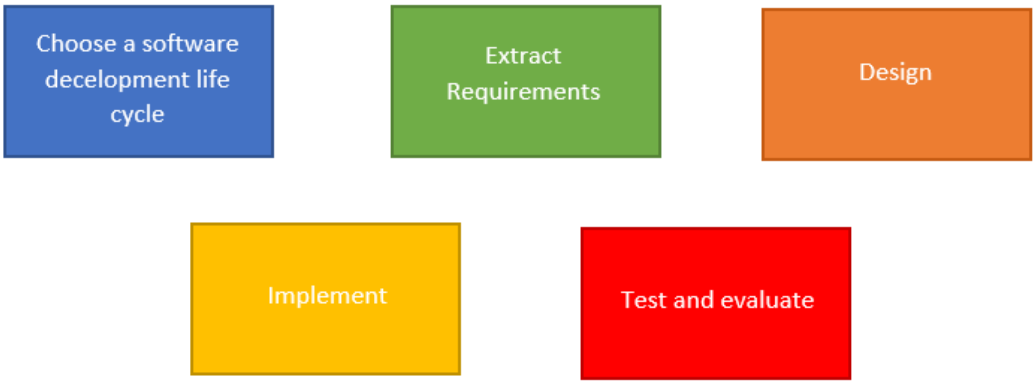

*METHODOLOGY*.....12

3.1 INTRODUCTION 12
3.2 SOFTWARE DEVELOPMENT LIFE CYCLE 13
*3.2.1 What is Software Development Life Cycle? 13*
*3.2.2 Software Development Process Models 13*
*3.2.3 How to choose a development model 29*
*3.2.4 What we chose? 30*
3.3 QUESTIONNAIRE 30
*3.3.1 Steps 30*
*3.3.2 Questions 31*
*3.3.3 Responses 32*
3.4 INTERVIEW 32
*3.4.1 Why Interview? 32*
*3.4.2 Interview questions 33*
*3.4.3 Interview responses 33*
3.5 REQUIREMENTS 34
*3.5.1 Steps 34*
*3.5.2 List of Requirements 34*
3.5.2.1 Projects 34
3.5.2.2 Contracts 35
3.5.2.3 Working Packets 35
3.5.2.4 Milestones 35
3.5.2.5 Timesheets 35


## *3.1 Introduction*

In this chapter, we describe the methodology followed and explained each step taken. Our software is based on the researchers' requirements, so we needed to find the best software development life cycle that covers our needs. After that, we had a questionnaire and interviews from where we extracted the requirements. Later on, in this chapter, we present the list of the requirements. In Figure 3.1 is seen the methodology we used to develop this platform.

*Figure 3.1*




## *3.2 Software Development Life Cycle*

### 3.2.1 What is Software Development Life Cycle?

Software Development Life Cycle (SDLC) is a process used by the software industry to design, develop and test high-quality software. The SDLC aims to produce high-quality software that meets or exceeds customer expectations, reaches completion within times and cost estimates.

ISO/IEC 12207 is an international standard for software life-cycle processes. It aims to be the standard that defines all the tasks required for developing and maintaining software.

Various SDLC models are followed during the software development process. These models are also referred to as "Software Development Process Models." Each process model follows a series of steps unique to its type to ensure success in software development. The most common models are explained below.

### 3.2.2 Software Development Process Models

**Waterfall Model**

The waterfall model breaks the software development process into several stages. Each phase must end before the next one starts. Additionally, the development team cannot return to a previous step when it is done; it is like a waterfall that goes down and never comes back. That is why this model is known as the waterfall model. The waterfall model is an instance of a plan-driven approach. They must plan and schedule every activity of the whole project before beginning software development. In each stage of the software development process, they have to examine the results obtained against those they expect. In all parts, quality has to be assured and controlled. After each stage in the waterfall model, they have one or more reports to be approved by the client.

The phases of the waterfall model are:

1. **Requirements analysis and definition**

   The development team tries to recognize the user's demands from the very beginning and as quickly as possible. They establish the system's services and restrictions by interviewing the system users or by making a questionnaire. The users then explain their work and their needs in detail, which is used as the system specification.

2. **System and software design**

   In the systems design process, the development team allocates the requirements to either hardware or software systems. That way, they set a system design that covers all the requirements of the client. In the software design, they need to recognize and explain the most necessary software system concepts and their connections.

3. **Implementation and unit testing**

   At this stage, the development team breaks the software into multiple program units, and they implement each of these units until they design every part of

the software. Afterward, they do some unit testing to verify that each part or unit meets its specifications.

4. **Integration and system testing**

   The development team combines the distinct program units, and they test them as a whole system to guarantee that they met the software requirements. After the various tests, they deliver the software system to the client.

5. **Operation and maintenance**

   In most cases, this is the phase that takes the most time. The system is set up and is functional in the user's place. The development team needs to maintain the system, which means:

   a. Fixing errors and omissions that were not detected before
   b. Enhancing the implementation of system units
   c. Upgrading the system's services with the new requirements that the development team discovered in order to evolve the system so it will remain useful

   However, they must consider that they may have to repeat previous process stages by changing things, which is unacceptable in the waterfall model.

As it is stated before, the next phase should not start until the previous one has finished. That leads to numerous difficulties where some of them are:

1. Actual projects rarely match the sequential flow that the model suggests.
2. It is frequently difficult for the client to declare all requirements explicitly. The waterfall model expects this and has difficulty accommodating the natural doubt at the beginning of many projects.
3. The customer must be patient. A working draft of the program will not be accessible until late in the project period.
4. If the development team does not detect a significant mistake until the final project review, it can be very harmful.

The fear that new information may appear leads to the possibility that both customers and developers may rush to freeze the software specification so that no one can make further changes to it. Sadly, this means that many obstacles are left for later resolution or ignored. Early freezing of requirements may mean that the system will not do what the user requires. [8]

In an impressive examination of actual projects, Bradac [Bra94] found that the linear nature of this life cycle leads to a loop, in which some project members must wait for coworkers to complete dependent tasks. The time wasted waiting can exceed the time spent on productive work.[6]

There is evidence that the waterfall model has many weaknesses. In many projects, the development team does not follow the linear sequence of phases as they see them in the waterfall model. In reality, the software has to be flexible and accommodate change while developing. Additionally, software work is fast-paced and is subject to continuous streamflow of changes (to features, functions, and information content). The waterfall model is often unsuitable for such work. [8]

The need for early dedication and system rework while having any changes means that the waterfall model is only appropriate for some varieties of systems such as:

1. **Enclosed systems where the software has to interface with hardware systems**

   Because of the firmness of hardware, it is not usually desirable to postpone decisions on the software's functionality until the development team implements it.[8]

2. **Crucial systems where the client needs boundless safety and security analysis of the software specification and design**

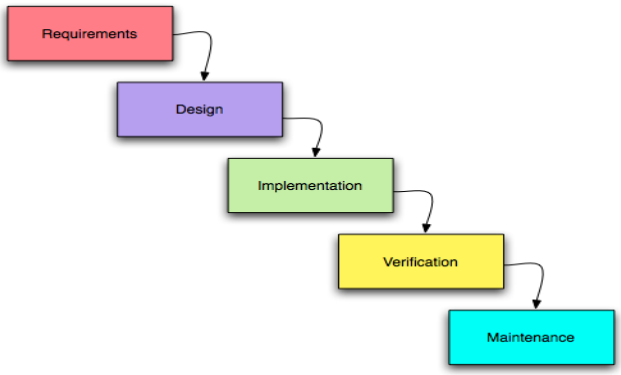
   In this type of software, the specification and design records must be perfect for making the different kinds of evaluation possible. Safety-

related restrictions in the specification and design are usually very costly to fix at the implementation stage. [8]

3. **Massive software systems that are part of more general engineering systems developed by several partner companies**
   The development team can develop the hardware using a similar model, and companies find it easier to use an ordinary model for hardware and software. Moreover, where numerous corporations are involved, complete specifications may be needed to allow for the autonomous improvement of different subsystems.[8]

In conclusion, the waterfall model is not the most suitable process model in situations where informal team communication is desirable. Additionally, this model will not be the best one if the software requirements change quickly.

*Figure 3.2 - The waterfall model*

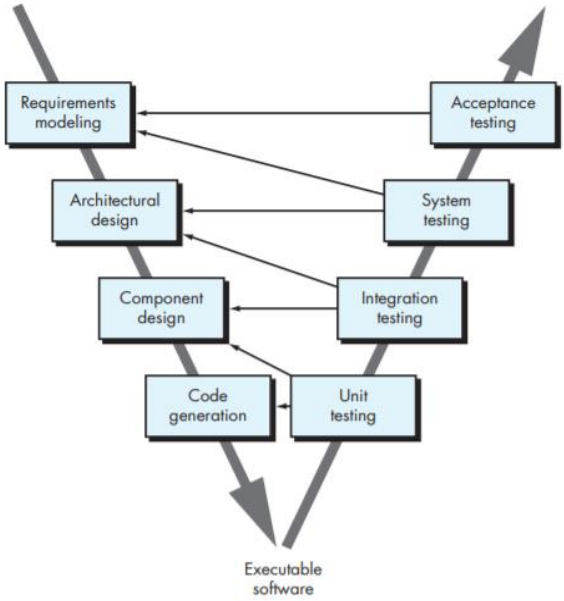



**V-model**

V-model is a variation in the design of the waterfall model. The V-model shows the relationship of tests to the primary activities. As the software team moves down the left side of the V, there are the basic activities of the waterfall as we saw them, and the code is generated. When the development team writes the code, they move up the right side of the V by performing a series of tests that validate each model they create as they move down the left side. However, like the waterfall

model, they cannot go back into previous stages. There is no significant difference between the waterfall life cycle and the V-model. The V-model gives us a way of visualizing how verification and validation actions are applied.

*Figure 3.3 – V-model*




**Agile Methodologies – Scrum model**

Agile development is a systems development strategy wherein the system developers can select from various tools and techniques to best accomplish their tasks. Agile development is believed to strike an optimal balance between productivity and quality for systems development [10]

The philosophy behind agile methods is reflected in the agile manifesto (HTTP:// agilemanifesto.org) issued by the leading developers of these methods. This manifesto states:

| Principle | Description |
|---|---|
| Customer involvement | Customers should be closely involved throughout the development process. Their role is provide and prioritize new system requirements and to evaluate the iterations of the system. |
| Embrace change | Expect the system requirements to change, and so design the system to accommodate these changes. |
| Incremental delivery | The software is developed in increments, with the customer specifying the requirements to be included in each increment. |
| Maintain simplicity | Focus on simplicity in both the software being developed and in the development process. Wherever possible, actively work to eliminate complexity from the system. |
| People, not process | The skills of the development team should be recognized and exploited. Team members should be left to develop their own ways of working without prescriptive processes. |

Agile methodologies have many different models with which developers can work. The most popular and typical examples are Scrum, eXtreme Programming (XP), Feature Driven Development (FDD), Dynamic Systems Development Method (DSDM), Adaptive Software Development (ASD), Crystal, and Lean Software Development (LSD).

The scrum model encourages teams to learn through experiences, self-organize while working on a problem, and reflect on their wins and losses to improve continuously. Scrum has many roles. One of them is the product owner, which is the customer representative, he/she works with the team, decides the features, is responsible for the final approval, and he/she has to accept or reject the work results. Another role is the scrum master. The scrum master has a leadership role, and he/she helps the team members to be responsible. The scrum master is responsible for the daily scrums and the sprint planning meetings. The last type of role is a scrum team member. The team has to develop the product. Scrum is based on sprints. At the start of each sprint, there is the sprint planning meeting, in which all roles took part. The product owner is responsible for prioritizing the items which are more important for the specific sprint. The sprint planning meeting is focused on what to do and how to do it in that specific sprint. During the sprints, there are some daily scrums on which the scrum master and the development team members report to each other. At the end of each sprint, there is the sprint review meeting on which the product owner reviews the sprint. Another meeting on the scrum model is the sprint retrospective meeting. Everyone discusses their experiences and problems faced during the sprint and discusses what to be followed and not to be followed to improve further. [7,11]

In scrum, there are three artifacts, product backlog, sprint backlog, and burndown chart. The product backlog is an ordered list of "requirements" that is maintained for a product. The product backlog is the master list of work that needs to get done maintained by the product owner or product manager. This is a dynamic list of features, requirements, enhancements, and fixes that act as the sprint backlog's input. The product backlog is constantly revisited, re-prioritized, and maintained by the Product Owner. As we learn more or as the market changes, items may no longer be relevant, or problems may get solved in other ways. The sprint backlog is the list of items, user stories, or bug fixes selected by the development team for implementation in the current sprint. Before each sprint, in the sprint planning meeting, the team chooses which items it will work on for the sprint from the product backlog. A sprint backlog may be flexible and can evolve during a sprint. The sprint burndown chart is a displayed chart showing the remaining work in the sprint backlog. Updated every day, it gives a simple view of the sprint progress. It also provides quick visualizations for reference. [7,11]

**Incremental development model**

Incremental development is based on the idea of producing a first version of the software with the basic features, give it to the client to get feedback, and based on that feedback, develop the next version. Developers will create numerous versions until they implement all the requirements.

Most of the time, they start with the most critical or most urgently required functionalities. That means that they deliver some fundamental requirements, but many additional features remain undelivered until future versions. The customer gets the current product and gives us his/her thoughts and comments about the functionalities that have been implemented and the new functionalities that he may want. As a result of this use and evaluation, they plan the next increment. Each new version includes some other requirements that the client wants. They repeat those stages after each version until they produce the final product with all

the client's functionalities. Developing software this way avoids the 'Big Bang' effect, i.e., for a long time, nothing happens, and then, suddenly, there is an entirely new situation. Instead of building software, the software grows [9] because instead of building the software, the software grows itself within each version.

The incremental development model is most appropriate when:

a. The client does not know all the requirements from the very beginning and wants to see a version of the software and think about what else he needs
b. When the client needs some functionalities right away, and other functionalities can come in later versions.

Incremental development is now the most common approach for the development of application systems and software products.[6] We can have incremental development as plan-driven, agile, or both. In a plan-driven approach, the development team must say from the beginning which increments will be in each version. If they prefer agile, they have the main functionalities in the first version and they see with the client the increments for each of the subsequent versions.

Incremental development has three significant advantages over the waterfall model:

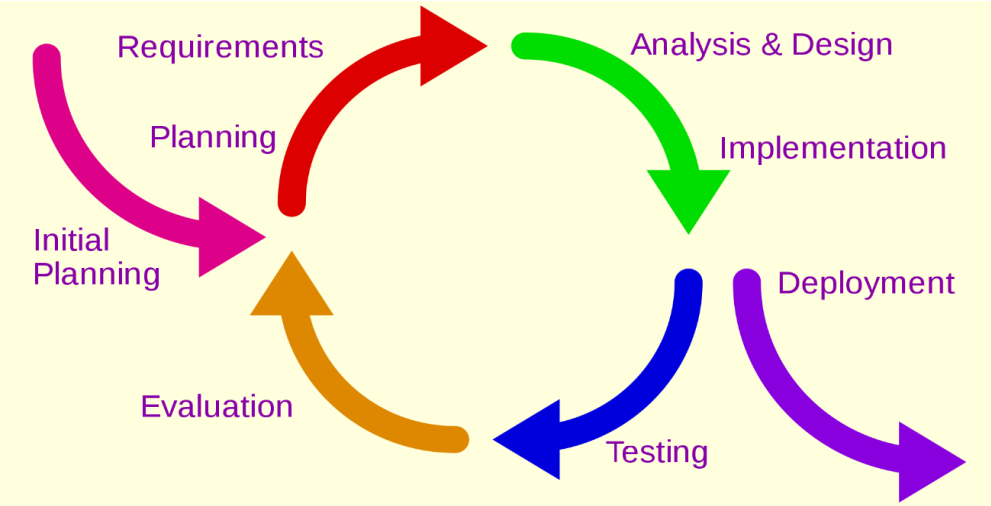
1. In the waterfall model, the development team cannot go back and change requirements, so they have to start from the beginning, making it very costly. In contrast, in the incremental development model, they can add new requirements after each version with much less cost.
2. With the incremental development model, they can get a lot more customer feedback because they evaluate each version after each phase. On the other hand, in the waterfall model, after each step, the client has just a document with the work that is completed until the current stage, which is not helpful because even if he does not like something, he could not do anything about it.

3. In the incremental development model, the client has an early version from the beginning and can use it, even if some functionalities are missing. That is very helpful for the client to familiarize himself with the software and give us feedback.

The incremental development model has two main drawbacks:

1. Managers need frequent deliverables to measure progress. If the various versions are developed quickly, it is costly both in terms of time and money to produce documents that reflect every system version.
2. The client can change the requirements and the functionalities, so this leads to messy code. Each time the client changes the requirements, it becomes more difficult for the manager and the programmers to include them effectively in the existing code without any problems appearing. To overcome these problems, developers should regularly refactor the software.

*Figure 3.4 – Incremental Development Model*




**Prototyping**

Prototyping is an alternative system model. Most of the time, the user has difficulties identifying detailed requirements for functions and features precisely. In situations like this, prototyping may be the best approach.

A prototype is an early version of the software system that the development team uses to show some functionalities, user interface, design alternatives, to get the perfect system for the client. They use rapid, iterative development to control the various costs, and the clients can experiment with each prototype throughout the software process. Prototyping can be used as a stand-alone model, but most of the time is used as a routine within the context of other process models such as the incremental development model.

The prototyping model helps the stakeholders and developers better understand what they need to build when they do not have specific requirements. The first thing they do is start with communication. They meet the clients to establish the goals of the software, identify the known requirements, and express other items that may be needed. They start by developing the user interface as the first prototype. The stakeholders who take the prototype provide us with feedback that we use to explain further the requirements that are not so clear and develop the next prototype with more features. Prototyping is an essential tool for requirements engineering. After the feedback of each prototype, it is possible that some functions or some designs that are in a particular prototype will be thrown away. This is known as throwaway prototyping.

The numerous prototypes allow the clients to see if the system will do the work they need it to do. Additionally, they may get new ideas for requirements and find some system vulnerabilities before the final stage so that they can overcome them quickly. Furthermore, while developing some prototypes, they may expose some omissions in the requirements when they combine the functions. In those cases, they can modify the requirements to be more clarified.

After a few iterations, the user is satisfied with a prototype, then the last version is developed, and the final product is delivered. This is known as evolutionary prototyping. In practice, evolutionary prototyping is used much more often than throwaway prototyping because it is not easy to throw away something that they worked on for many days.

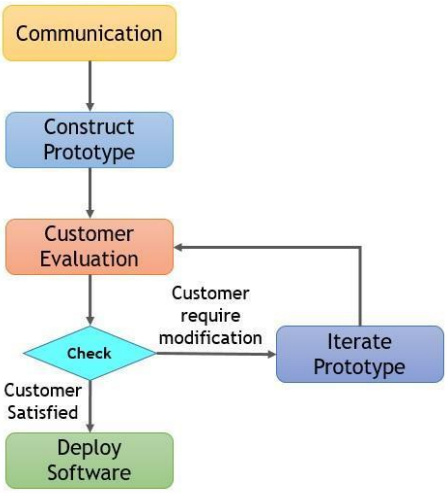
**Advantages**

- The resulting system is easier to use because the users will be already familiar
- User needs are better accommodated
- The resulting system has fewer features because the developers can throw away some unnecessary features (throwaway prototyping)
- Problems are detected earlier
- The design is of higher quality.
- The resulting system is easier to maintain.
- The development incurs less effort

***Disadvantages***

- The resulting system has more features if the developers do not like to throw away unnecessary features and, in each prototype, they find more requirements
- The performance of the resulting system is worse if the final product has many features
- The design is of lesser quality
- The resulting system is harder to maintain
- The prototyping approach requires more experienced team members

*Figure 3.5 – Prototyping development model*



**Extreme Programming**

Extreme Programming is a pure agile method and is the most widely used approach from agile software development models. Extreme Programming is based on four primary activities: planning, design, coding, and testing.

### *1. Planning*

The planning activity helps the team understand the business context of the software and get a general idea about the main features and functionalities needed. Then the customer creates user stories that explain the required features that the development team needs to build for the software. The customer writes the user stories and places them on an index card. Then he assigns a priority on each, based on the value of that feature on the software. Then the members of the team get each user story and allocate the cost of it.

Each user story must be calculated to require less than three weeks. Otherwise, the customer must split it into smaller user stories and reassign value and cost. A user story can be written at any time, not only at the beginning of the project. The whole team, customers, and developers decide how to group the user stories into the next increment developed. When the basic agreement on the stories to be included, the delivery date, and other project matters are made, then the team chooses how will implement the user stories. The three ways are the highest value first, highest risk first, or all user stories are equal.

When the developers deliver the first software increment, the team figures how many user stories they implement into the first increment. That helps to estimate the delivery dates and schedule each increment. Throughout the project, the client can add, change, split, or delete some user stories. Then the team must review the remaining releases and adjust the plan.

### *2. Design*

Extreme Programming design strictly follows the keep it simple principle. The development team prefers a simple design that is literally what the user story writes. They do not design additional functionalities that may be required later. Extreme Programming uses the CRC cards, which are the only design work product in Extreme Programming, to identify and organize the classes that are relevant to each increment. Extreme Programming support refactoring of the code to make it simpler. It is also a way to clean up the code and minimize the chances of introducing bugs. A problem that occurs is the increment of the effort they need to refactor the code when the software grows.

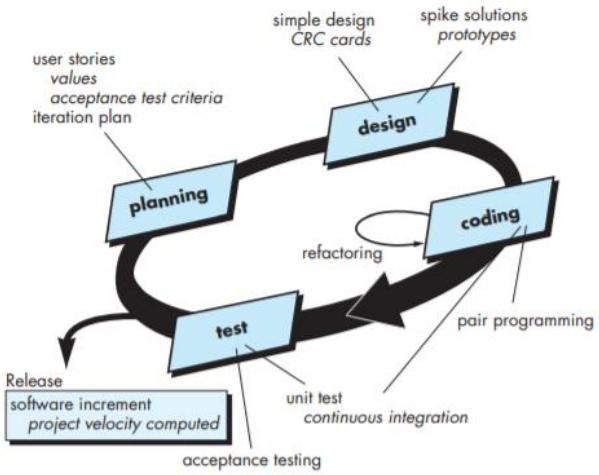
### *3. Coding & Testing*

Before the beginning of coding, the team must develop some unit tests that will be needed to test if the user stories that were included in that release have been implemented correctly. When they complete the tests, then they write the exact code that is needed to pass them. With that in mind, they keep the code of each feature very simple. When they finish coding, they unit-test the code instantly; that way, they get immediate feedback. During coding, they do pair programming, which means that two people work together on the same computer. For example, one is coding, and the other ensures that everything is going as the plan says.

Practical, Extreme Programming, as proposed initially, is more complicated than expected. It cannot be easily combined with the management practices and culture of most businesses. Therefore, the companies selecting the agile methods pick those Extreme Programming practices that are most appropriate for their way of working and not the Extreme Programming as a whole.

Extreme Programming is suited for particular projects, but not for all. If requirements are unsure, the system is not too big, and the customer can be available throughout the project, Extreme Programming deserves serious consideration.

*Figure 3.6 - Extreme Programming*




**Spiral Model**

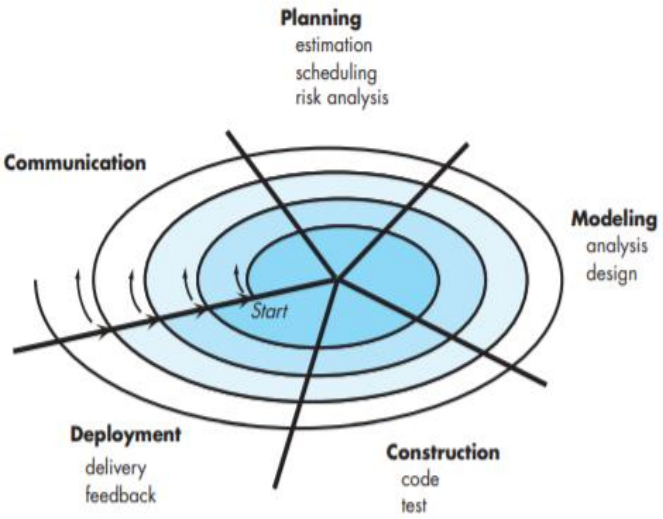
The spiral model is an evolutionary risk-driven software process model with repetitive prototypes like prototyping or incremental development and is controlled and systematic like the waterfall model. It has two main features, the cyclic approach for incrementally growing the implementation of the system while decreasing the risk and a set of milestones. By using the spiral model, the software is developed in a series of incremental releases.

At the beginning of the project, the releases are some models or prototypes. During later iterations, there are versions of the system that are more whole. A spiral model is divided into some activities established by the software engineering team like design or coding, which are the activities around the spiral clockwise. The project begins at the center, and clockwise it goes to the outer layers. Risk is considered each time the development team finishes all the activities. After the risk analysis, they start again with the activities adding more features. Anchor point milestones are written each time after the risk analysis. The first turn around the spiral might result in the development of a product blueprint. Next, loops around the spiral are used to produce a prototype and, later, more advanced software versions.

They recalculate the cost and readjust the schedule, depending on the feedback they receive from the customer after each delivery. In addition, the project manager calculates the number of iterations they seem to require until the software will be ready.

The spiral model is a more practical approach to the development of large-scale systems and software. As is known, the software evolves as the process proceeds. Both the developer and customer understand better the requirements and the risks after each iteration. The spiral model uses prototyping as a risk-reducing mechanism and lets us implement the prototyping approach at any phase in developing the product. It maintains the systematic stepwise approach suggested by the classic life cycle but organizes it into an iterative process that is more realistic. The spiral model requires attention to technical risks at all steps of the project. The risks that are found need to be reduced before they become problematic. However, the spiral model is not the perfect model. It demands significant risk evaluation expertise and depends on its success on this. If a considerable risk is not revealed and maintained, problems will indeed occur.

*Figure 3.7 – Spiral model*



### 3.2.3 How to choose a development model

***1. Fixed requirements VS Flexible requirements***

The Agile and Iterative methods are perfect if they want to build an application, and the requirements will be changing during the project. The waterfall fits a classic web and app development where they know right from the beginning of the project all the requirements.

***2. One major release VS Continuous delivery***

If the client wants only the final release, then they choose Waterfall or V-model. However, these two models are most suitable for small projects. If they have to develop a large project, then a single final release has a risk of a more significant number of bugs due to a more considerable amount of code for the developers to keep track of.

All other models suggest regular releases, every four weeks approximately, and represent 'iterative' delivery, where we get a working product early in the development and then see it develop. Any coding errors or disagreements with our requirements can be quickly spotted and quickly fixed.

***3. Documentation VS Communication***

Many models, like Spiral, V-model, and Waterfall, recommend very detailed documentation and limited communication, while the Iterative model tries to balance documentation and communication. On the other hand, in the models of the Agile group, direct and frequent communication is a must.

*Figure 3.8*

| Factors | Waterfall | V-Shaped | Evolutionary Prototyping | Spiral | Iterative and Incremental | Agile |
|---|---|---|---|---|---|---|
| Unclear User Requirement | Poor | Poor | Good | Excellent | Good | Excellent |
| Unfamiliar Technology | Poor | Poor | Excellent | Excellent | Good | Poor |
| Complex System | Good | Good | Excellent | Excellent | Good | Poor |
| Reliable system | Good | Good | Poor | Excellent | Good | Good |
| Short Time Schedule | Poor | Poor | Good | Poor | Excellent | Excellent |
| Strong Project Management | Excellent | Excellent | Excellent | Excellent | Excellent | Excellent |
| Cost limitation | Poor | Poor | Poor | Poor | Excellent | Excellent |
| Visibility of Stakeholders | Good | Good | Excellent | Excellent | Good | Excellent |
| Skills limitation | Good | Good | Poor | Poor | Good | Poor |
| Documentation | Excellent | Excellent | Good | Good | Excellent | Poor |
| Component reusability | Excellent | Excellent | Poor | Poor | Excellent | Poor |

### 3.2.4 What we chose?

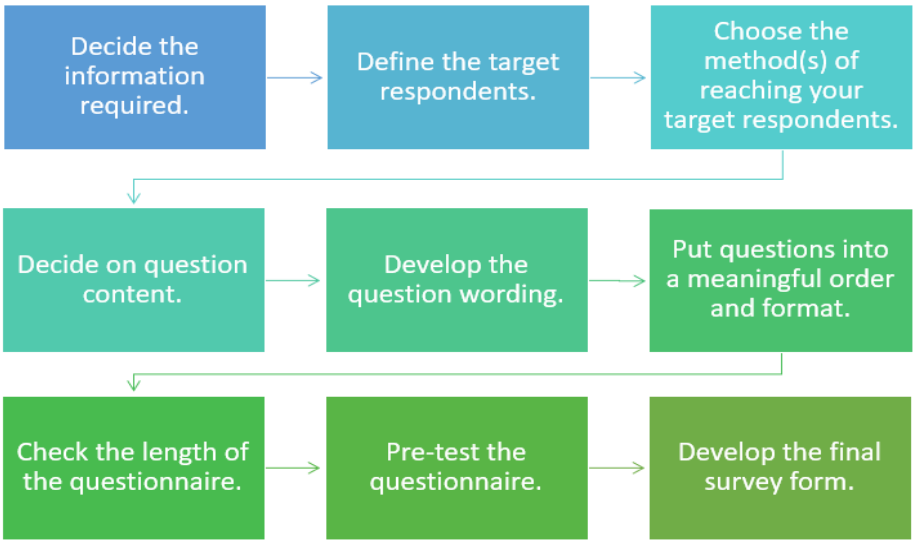
The user requirements were clear enough for our software but not a hundred percent, so the waterfall and V-shaped model was not our choice. Additionally, we preferred to have continuous delivery, so the users can check it and tell us if they want to change anything. So we were leaning on the incremental model or Agile. Finally, we wanted to have more communication and not documentation. So Agile was our best option. We choose to go with Scrum methodology, which is easy to implement. It has a series of sprints and a lot less documentation than other traditional models. With the Agile method, we had a lot more communication with the customer, so the platform has everything he/she needs.[11] On our team, the product owner was the researchers, the scrum master was the co-supervisors, and the team was me.

## 3.3 Questionnaire

### 3.3.1 Steps

To make the questionnaire, we followed some steps, as shown in Figure 3.9. First, we decided which information we needed to collect. Demographic data, experience, and which features users think as essential. Then we decided who will be our target respondents, who, for us, were the researchers that will use this platform. After that, we decided to create our questionnaire on google forms and send it to the respondents via email. Afterward, we decided on the exact questions and found the right way to write them to be easily understandable. Later we reorder the questions and cut them into sections to have a meaningful order that does not confuse the respondent. Finally, we checked the length and the content of the form, and we sent it to the respondents.

*Figure 3.9*




### 3.3.2 Questions

The questionnaire was developed in three main parts. First, there were some general instructions and explanations. The second part was about personal information and demographics, and the third part was the main body. In the second part, we have gathered information about the gender, age, and relevance of the person who is completing the questionnaire with the research projects in general and Erasmus+ specific. If he/she has previous experience, we had questions about the sector (Industry/Academia/Other) he/she had worked on, years of experience, and the role he/she has/had. After gathering this information, we asked about some scenarios to get a more specific idea of what they need. In the third section, the

body of the questionnaire, we had some functionalities for which we asked the importance they have for the person who was completing the questionnaire. We found those functionalities in other web tools, and we thought that there might be important. At the end of this section, we asked if something else is wanted. Questions can be found in Appendix A

### 3.3.3 Responses

We had sixteen answers, nine from Females and seven from males. Half of them were between eighteen and thirty years old, five persons forty-six to sixty and three persons thirty-one to forty-five. Six of those have worked or are currently working on a research project. Our responders have different years of experience; one-third of them has less than two years of experience, one-third has two to five years of experience, and one-third has more than five years of experience. All of our responders have experience in Erasmus+ Strategic Partnership projects in various roles except Admin. As stated by their responses, it was very important to have financial reports from several months and several projects, and it was essential to have for different personnel.

The most crucial feature from the ones we stated was the project dashboard. After that, the next necessary functionality based on the responses was the timeline chart, the budgeting, and the financial reports. The rest was important to neutral except computer using tracks which was the voted as slightly to not at all important.

## 3.4 Interview

### 3.4.1 Why Interview?

Interviews are a perfect way to gather requirements right away from the users. We get precisely the features that they need, and they can explain to us everything we need.

### 3.4.2 Interview questions

We conducted extensive interviews with two of the researchers. In the beginning, they showed us what they are using now. After that, we had some general questions like "What do you want the platform to look like?" or "Which functionalities do you believe are essential?". Throughout the process, we try to have more like a conversation. This way, we get as much as possible from those who will use it. But we were constantly having in mind not to get away from our theme.

### 3.4.3 Interview responses

After the interview with the two researchers, we got some time to analyze the responses and get the results. During the interview, the respondents state that the most critical functions were the timesheets and the budget. As it was said by them, they need a tool to expedite the process of completing and checking timesheets. Especially the checking of timesheets is a process that takes a lot of time when something went wrong. Another thing that was stated was that they need to have the budget information compacted in a place so it will be easier to analyze it. Having all the budget information in one place was a thought of relief because they do not want to update the budget manually to avoid potential errors. In conclusion, the interview with the researchers help us understand the following requirements:

1. Each type of employee has a different rate
2. Rates can be changed but are the same for each project
3. Total hours for the day and the month can be changed
4. Total hours for the day and the month are the same for everyone
5. Information needed for each project, work package, contract and milestone
6. How the contracts work
7. Limitations about timesheets
    a. Cannot add timesheet on weekends
    b. Cannot add timesheet on National days

## 3.5 Requirements

### 3.5.1 Steps

The steps needed to get the requirements we will need to develop the platform are listed in Figure 3.10.

*Figure 3.10*


### *3.5.2* List of Requirements

#### 3.5.2.1 Projects

- The user can see the projects that he/she works on
- Each project has a name
- Each project has a budget
- Each project has some work packages.
- Each project can have the following roles:
  - Manager with a rate of €164/day
  - Researcher with a rate of €137/day
  - Technician with a rate of €102/day
  - Administrator with a rate of €78/day
- Rates can be changed from settings, but there are the same for each project.
- Each project has some project milestones at which some deliverables must be delivered and some financial milestones.

#### 3.5.2.2 Contracts

- The user can see his/her active or inactive contracts
- Each contract is about one user
- Each contract has a start and an end date
- Each contract is about one project and one role. Different roles on the same project are separate contracts
- Each contract has some hours

#### 3.5.2.3 Working Packets

- Each working packet has a name
- Each working packet has a number
- Each working packet has a start date and end date
- Each working packet has some hours
- Each working packet has a budget
- At the beginning of a project, each working hour of a working packet is assigned to users with contracts. The program will show the working hours left for each working packet after the employees fill their timesheets each month.

#### 3.5.2.4 Milestones

- Each milestone has a date
- Each milestone has a name
- The platform must send a notification when the due date is near

#### 3.5.2.5 Timesheets

- On timesheets, the employees cannot claim hours on weekends or public holidays.
- Employees cannot claim more than a maximum of hours each day on all the projects they are working on.

- Employees cannot claim more than maximum hours each month on all the projects they are working on.
- For now, the maximum hours per day on all the projects that he/she is working are 7.5
- For now, the maximum number of hours each month on all the projects that he/she is working are 140
- Hours per day or month can be changed from the settings.
- Timesheets can be downloaded and printed as PDF
- Additionally: Total hours claimed for each employee (in all of her/his timesheets) as well as the individual hours claimed for each project (in each separate timesheet), should match their total and per agreed working hours according to their contracts

# *CHAPTER 4*

## *IMPLEMENTATION*

## CONTENT

*IMPLEMENTATION*.....37

4.1 WORDPRESS 37
4.2 DATABASE 38
4.3 METAPHORS 40
4.4 VISUAL DESIGN CONSTRAINTS 41
4.5 USABILITY HEURISTICS 41
4.6 PROTOTYPING 43
*4.6.1 Low fidelity prototyping* *43*
*4.6.2 Medium fidelity prototype* *44*

### *4.1 WordPress*

For the implementation, we chose to exploit the WordPress Content Management System (CMS) as a basis for our development. WordPress is a free software with a lot of plugins and themes we can choose from. It has a lot of existing themes which provide "user-friendly interfaces." WordPress is open source, so developers can add their functionalities on top of its code by creating their plugins. WordPress is safer and more secure than trying to secure our own platform, so we relied on this to focus on implementing in detail the financial functionalities studied in this thesis. WordPress takes privacy into consideration. It has taken some measures towards GDPR compliance. We constructed our own plugins to use, and we only used one existing plugin from the WordPress repository, *UltimateMember*. Ultimate Member plugin helped us with User management. It is responsible for new users, sign up, log in, forgot password, and account. We chose to use a plugin for these functions because it matters to be secure. After all, we need to focus on the financial details of the projects, and user-related functions must is essential

to be secured. WordPress is a very popular CMS that is used by millions of developers, as we can see in Figure 4.1.

*Figure 4.1 – CMS technologies*

| CMS Technology | Live Websites | Market share |
|---|---|---|
| WordPress | 28,183,568 | 44.29% |
| Wix | 4,565,423 | 7.17% |
| Squarespace | 2,583,557 | 4.06% |
| Joomla! | 1,662,593 | 2.61% |
| GoDaddy Website Builder | 1,624,438 | 2.55% |
| Weebly | 1,040,819 | 1.63% |
| Duda | 705,120 | 1.10% |
| Blogger | 672,560 | 1.05% |
| Drupal | 562,655 | 1.03% |
| Jimdo | 486,843 | 0.76% |

## 4.2 Database

We work on phpMyAdmin with a MariaDB database. We need the database to save the projects, the working packets, who works where etc. We have eight tables with many columns in each one. The Database Schema is presented in figure 4.2.

Firstly we have a table for the national holidays with three columns, the key, the name of the national holiday and the date. We use this table to check if an employee can or cannot add a timesheet one day.

Another table we have is Hours, and this table also has three columns, the id, daily hours, and monthly hours. This table only has a row showing the hours that an employee can work in a day and a month. This table helps us again with the timesheets to ensure that no employee completes timesheets for more ours than the one he/she be entitled to.

We also have the roles table, which is a table that held the information about the four positions, and we have the role id, the role name, and its rate.

Next, we have the Projects table, which has six columns. It has a project id which is used by other tables too, as a foreign key, the project name, the initial budget of this project and the budget left at this moment, the total Project hours, and the hours left at this moment. The project's budget is reduced when a work package with some budget is created. The project's hours are reduced when someone adds a contract to this project.

After that, we have the work packages table. It has fifteen columns. Each work package has a number, a name, a start date, an end date, an initial budget, and initial hours for each role. The remaining columns are the budget left and the hours that have been worked from each position. The work package budget and the corresponding hours are reduced when an employee completes his/her timesheets for a specific role and work package.

In addition, we have the works on table where we save all the contracts of the users. It has eleven columns, a unique id,  the id the user who has the agreement, the project on which he/she has the contract, when the contract start and ends, the role, the hours assigned to this contract, how many hours left on this contract, and if it is active or not with a description. We also compute the number of months the contract is on. Hours left on a contract are reduced when the user is assigned to a work package.

Another table we have is the work on work package, which is a table that shows us which user is assigned to which work package. It has seven columns, the user

who is assigned, the project and the work package that the user is set on, the hours he/she has to work, the role of this employee, the start, and the end date. We also have the hours left on this assignment  which are reduced when an employee completes his/her timesheets.

At last, we have the table with the timesheets. It has six columns, the date for which the employee is completing the timesheet, the hours, the employee id, the project, and the work package, and the role.

*Figure 4.2 – Database Schema*

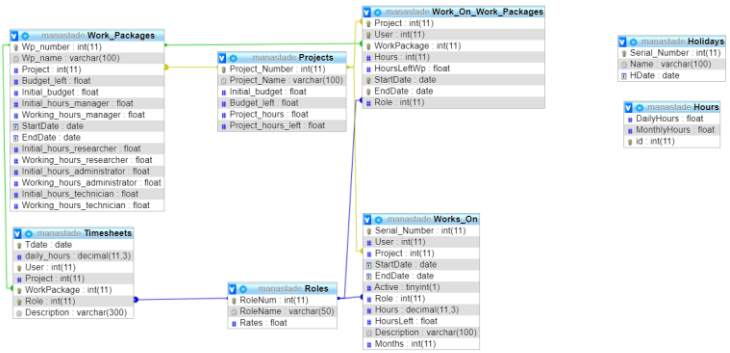
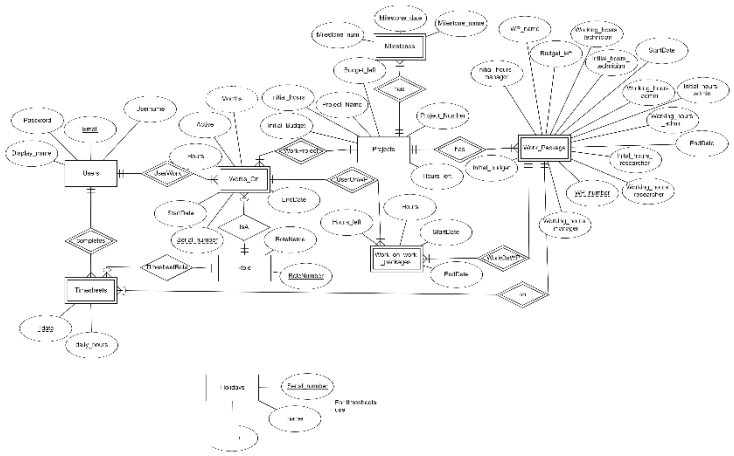



*Figure 4.3 –ER diagram*

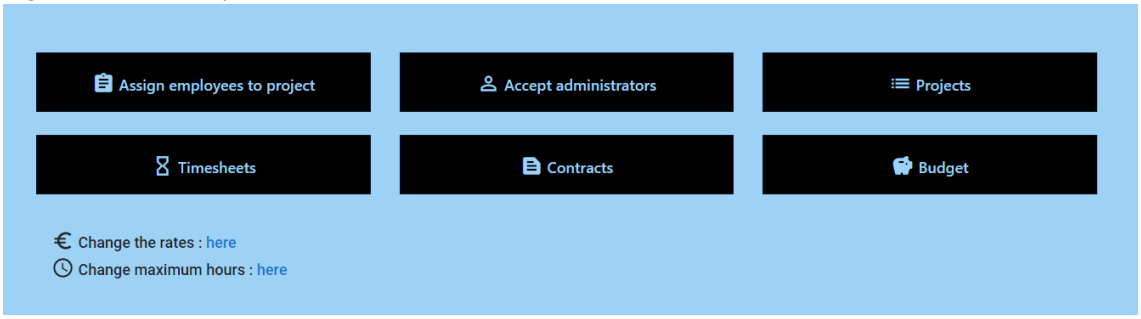


## 4.3 Metaphors

Metaphors in web design are essential. The advantage of using a metaphor is that the user can identify with the environment presented on the screen. Having a metaphor allows users to predict the outcome of their actions more quickly [1]. We used metaphors in the admin panel on our platform to be easier for him/her to

understand each button. We have on each line both the verbal description and an icon which is the metaphor. As we can is in Figure 4.4, we have the euro symbol for the rates, a clock for hours, a person for the accept an administrator, etc.

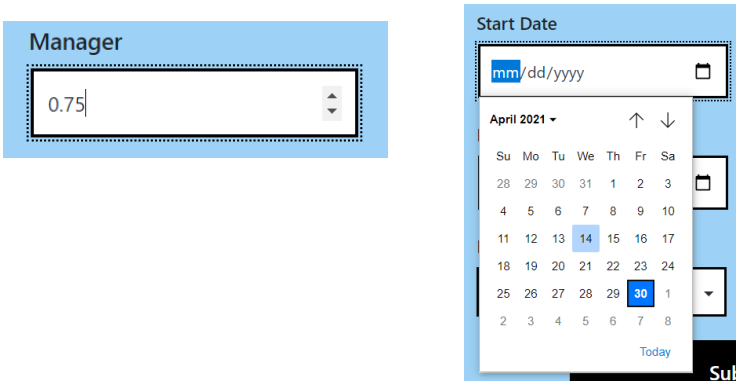
*Figure 4.4 - Metaphors*




## 4.4 *Visual design constraints*

On our platform, we use some visual design constraints. In the various forms we have on our platform, we have a specific input type in each cell so we can avoid some possible problems. For example, if a number is needed, the employee must complete a number. If he/she tries to write letters, then nothing will be registered. Another example is the dates, if the employee needs to complete a date, then a calendar will pop up, and he/she will choose a day. That way, all the dates will be passed in the same format. [1,3,4,5]

*Figure 4.5 – Constraints*




## 4.5 *Usability heuristics*

Heuristic evaluation (Nielsen and Molich, 1990; Nielsen 1994) is a usability engineering method for finding the usability problems in a user interface design so that they can be attended to as part of an iterative design process. Jakob Nielsen described the 10 general principles for interaction design. These principles were

developed based on years of experience in the field of usability engineering and they've became rules of thumb for human-computer interaction.
They can help to save development teams considerable amounts of time during early usability testing, so that they can direct their attention to more complex design challenges. In addition, it's also worth it to use them as a checklist when designing a new product or a feature. [2]

1. **Visibility of system status**

   We have messages about what is happening. If something went wrong or if everything is okay.

2. **Match between system and real world**

   We have simple words with which the employees are familiar. We used words that they are already using to complete their job.

3. **User control and freedom**

   User has many choices if he/she wants to leave a page. First of all, we have the menu on the header from where he/she can go to other pages. We also have the back button on pages where the employee can go back.

4. **Consistency and standards**

   All the buttons and links on the platform have the same colors. Additionally, all the forms have the same format.

5. **Help user recognize, diagnose, recover from errors**

   We have error messages which explain exactly what went wrong so the employee can correct it next time.

6. **Error prevention**

   We have some visual constraints on the forms to avoid errors.

7. **Recognition rather than recall**

We have a header menu on which there are various pages. Also, in some forms, where is needed to remember which projects the employee has contracts on and with what role we have some dropdown menus. Additionally, when the employee chooses a day on timesheets, we have an alert with the hours left to complete on that day and on that month.

8. **Flexibility and efficiency of use**

   We have shortcuts in the header menu because researchers can go from one page to another.

9. **Aesthetic and minimalistic design**

   We only show what is needed on each page.

10. **Help and documentation**

## *4.6 Prototyping*

### 4.6.1 Low fidelity prototyping

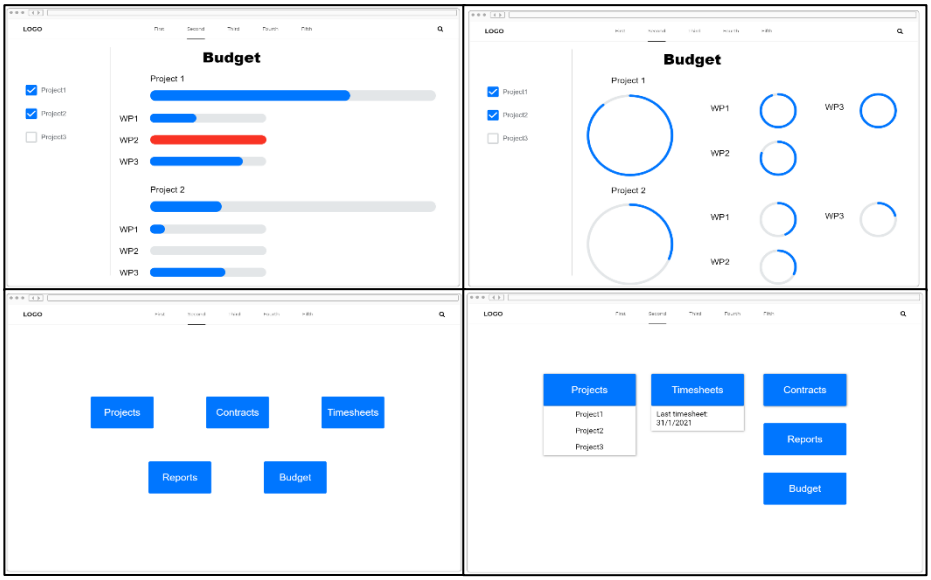
A low-fidelity prototype does not look very much like the final product, nor does it provide the same functionality. For example, it may use very different materials, such as paper and cardboard, rather than electronic screens and metal. It may perform only a limited set of functions or represent the functions and not perform any of them. Low-fidelity prototypes are helpful because they tend to be simple, cheap, and quick to produce. This also means that they are simple, inexpensive, and fast to modify to support the exploration of alternative designs and ideas. This is particularly important in the early stages of development during conceptual design. Low-fidelity prototypes are not meant to be kept and integrated into the final product. [5] For our system, we had a few low-fidelity prototypes on the tablet, mainly about the forms.

*Figure 4.6 – Low fidelity prototype*


### 4.6.2 Medium fidelity prototype

A medium-fidelity prototype is a prototype with limited functionality but clickable areas that present an application's interactions and navigation possibilities. Medium fidelity prototypes are usually built on some software. For every action employee is getting, a basic visual design is created. The medium-fidelity prototype is suited for the validation of the interaction concept. In contrast to a low fidelity prototype, the understandability of interaction elements can be validated. For our platform, we made some pages with proto io to choose the interface we wanted.

*Figure 4.7– Medium fidelity prototypes*



# *CHAPTER 5*

## *DEMONSTRATION*

## CONTENT


*DEMONSTRATION*.....46

5.1 PERSONAS 46
5.2 SCENARIOS 48
*5.2.1 New employee 48*
*5.2.2 New project 48*
5.3 USER MANUAL 49
*5.3.1 Login 49*
*5.3.2 Register 49*
*5.3.3 Forgot Password 50*
*5.3.4 Menu 50*
*5.3.5 Dashboard 50*
*5.3.6 Projects 51*
*5.3.7 Work Packages 51*
*5.3.8 Work package hours 51*
*5.3.9 Budget 51*
*5.3.10 Contracts 52*
*5.3.11 New Contract 52*
*5.3.12 Timesheets 53*
*5.3.13 Edit Timesheet 54*
*5.3.14 New Timesheet 54*
*5.3.15 Admin Panel 54*
5.3.15.1 Rates 55
5.3.15.2 Hours 55
5.3.15.3 Assign employees 55
5.3.15.4 Accept Administrators 56
5.3.15.5 Projects and Work packages 56
5.3.15.6 New Project 57
5.3.15.7 New Work Package 57
5.3.15.8 Timesheets 57
5.3.15.9 Contracts 58


## *5.1 Personas*

A persona is a rich picture of an imaginary person who represents the core user group. They do not describe specific people, but they are realistic and not idealized. Each persona represents a synthesis of several real users who have been involved in data gathering, and it is based on a set of user profiles. Each persona is characterized by a unique set of goals relating to the particular product under development, rather than a job description or a simple demographic. This is because goals often differ among people within the same job role or the same demographic. In addition to their goals, a persona will include a description of the user's behavior, attitudes, activities, and environment. These items are all specified in some detail. A persona has two goals (Caddick and Cable, 2011), to help the designer make design decisions and remind the team that real people will be using the product. When a design solution is proposed, the team can ask, 'how would Angie react to this?'. The detail is deliberately more than is strictly necessary, but this is essential. It is only by feeling that Angie is a real person that the team can imagine how she will behave. [1][4] In Figure 5.1 are two of the personas we had.

*Figure 5.1 – Personas*


## *5.2 Scenarios*

### 5.2.1 New employee

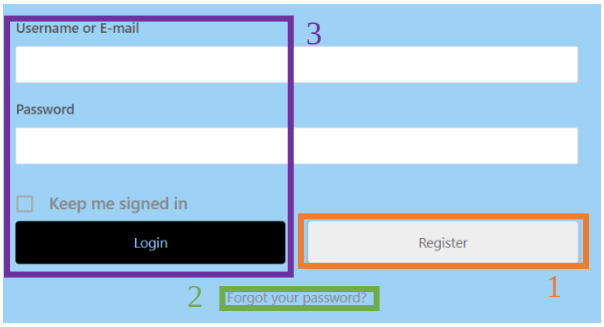
Angie just starts working at this lab. On her first day, her manager informed her that she needs to make an account on the platform. Angie then went to her desk, and she signed up on the platform using her email. After that, she logged in, and then she added her contract. Every day, Angie goes on the platform and sees if she is assigned anywhere else and completes her timesheets.

### 5.2.2 New project

Alex is the administrator of a new project that just came in. When he went to the job on Monday morning, he opened the platform, and he added the project and the work packages associated with this project. When all the employees added their contracts for this project, he went on the platform and assigned them to work packages.

## *5.3 User manual*

### 5.3.1 Login




1. If you do not have an account, you can register by pressing the "Register" button

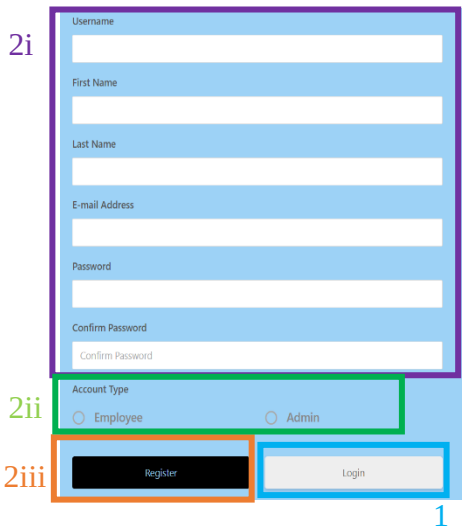
2. If you forget your password, you can press the link "Forgot your password?".

3. If you already have an account, you can log in by completing your username or email and the password and then press the "Login" button. ( You can also tick the "Keep me signed in" box if you want)

4. If the email (or username) or password is wrong, you will need to recomplete them.

5. When you are successfully logged in, you can see your account.

### 5.3.2 Register



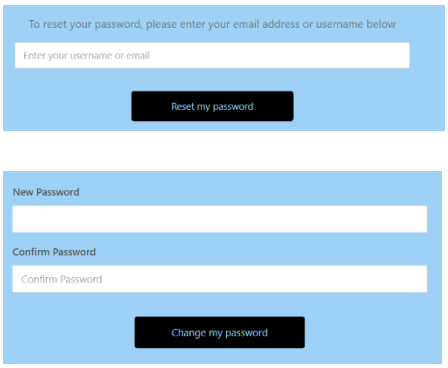

1. If you are here by accident and already have an account, you can press the "Login" button to go to the Login screen.

2. If you do not have an account, then:

i. Complete your information (Username, First Name, Last Name, Email, Password twice).

ii. Choose the account type you want to have (to become Admin, another Admin must accept you) iii. Press the "Register" button

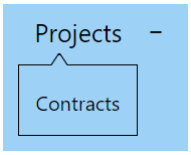
### 5.3.3 Forgot Password

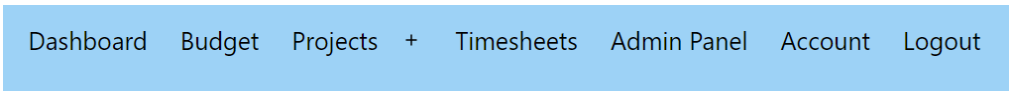



1. If you have an account, but you have forgotten the password, then complete the username or the email of your account and press the "Reset my password" button to get an email to change the password
2. If you are here by accident, you can press the "Login" or "Register" button on the header menu to go to that screen.

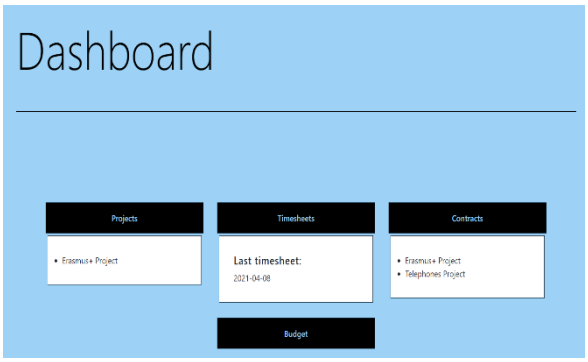
### 5.3.4 Menu

On the header, you can see the menu with the various pages. To see the contract's page, you must go on the + sign next to projects.



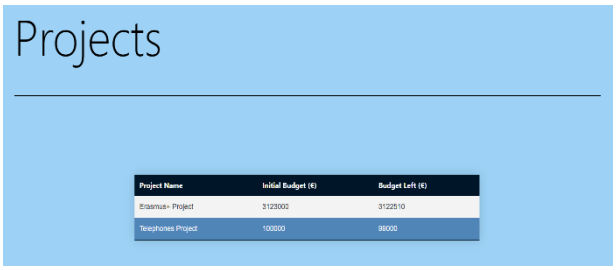

### 5.3.5 Dashboard




On the dashboard page, we have:

1. Projects: You can find the projects on which you have a contract and you are assigned right now.
2. Timesheets: You can see the last timesheet you have completed. (chronologically)
3. Contracts: You can see the contracts that you have active now.
4. Budget: You can go to the budget page.

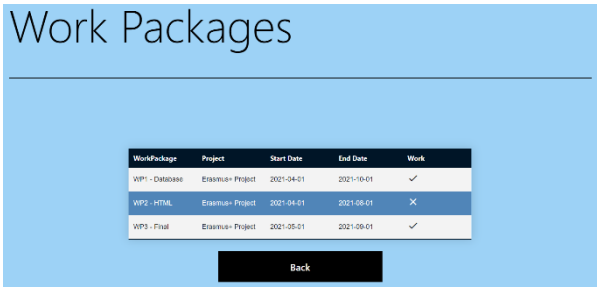
### 5.3.6 Projects




On the project's page, you can find the tasks that you have a contract with, along with their name and their budget.

**If you click on the project's name, you can see its work packages**

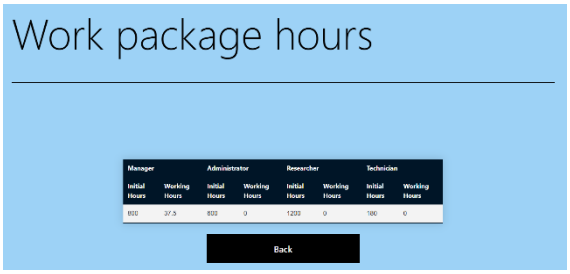
### 5.3.7 Work Packages




On the Work Packages page, you can find the work packages for a specific project along with the start and end date for each one and if you are assigned on it (✓) or not (✗).

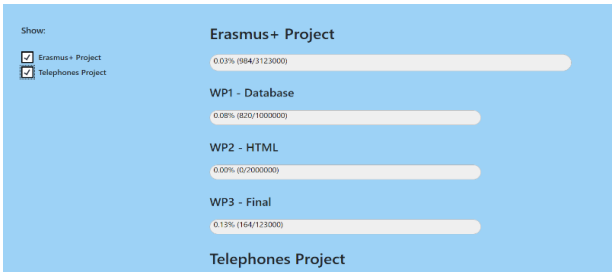
**If you click on the work package's name, you can see more information about the hours**

### 5.3.8 Work package hours




On this page, you can see the initial hours of each role for the specific work package and the hours left for each position.

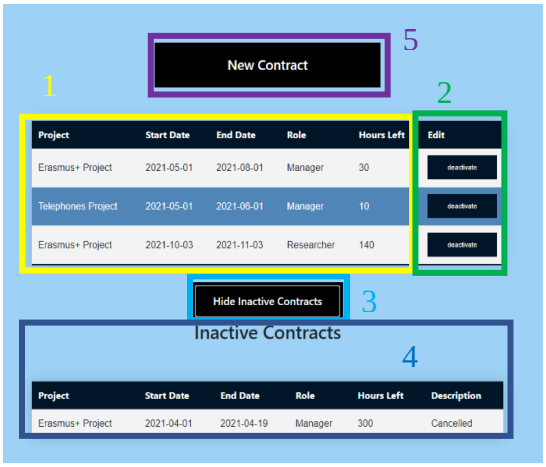
### 5.3.9 Budget

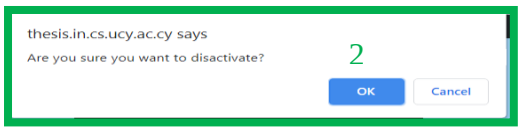



On the budget page, you can see the budget for every project you have a contract on and the work packages. You can also hide some projects if you want to.

### 5.3.10 Contracts

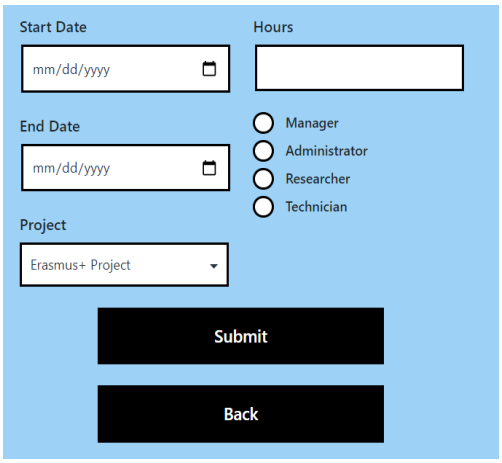
On the contract's page, you can:




1. View all the active contracts you have. You can see the project's name, the start and end date, the role, and the hours left for each one.
2. Deactivate an active agreement
3. Show or hide the inactive contracts
4. View all the inactive contracts. You can see all the information you had for active contracts and a description of why they are inactive (Either "Cancelled" or "Ended as planned")
5. Click the new contract button to go to a form to add a new contract.

### 5.3.11 New Contract

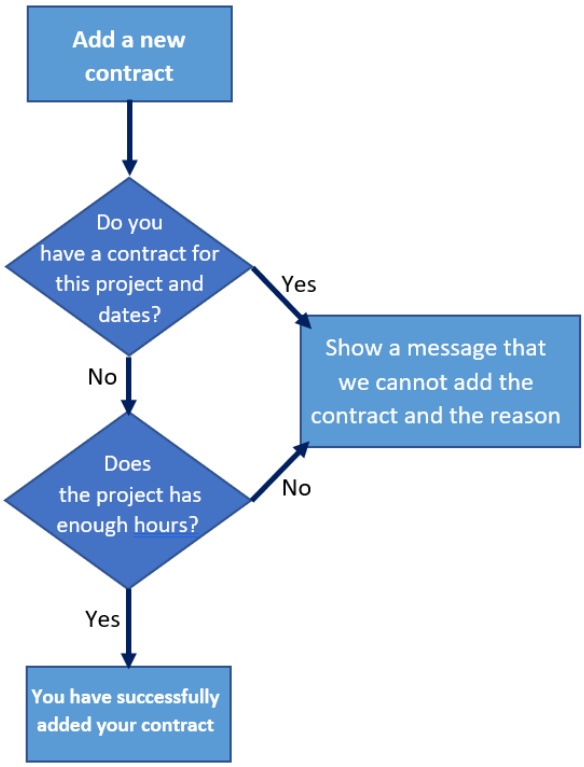



On this page, you can add your contract. To add a new contract, you will need:

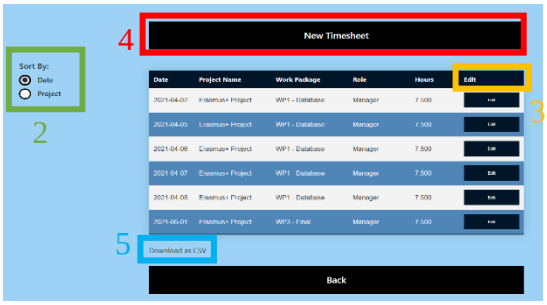
1. A start date
2. An end date
3. The project
4. The hours you will work on this project
5. The role that you will have.

**End date must be after start date**

*Figure 5.2 – New Contract*



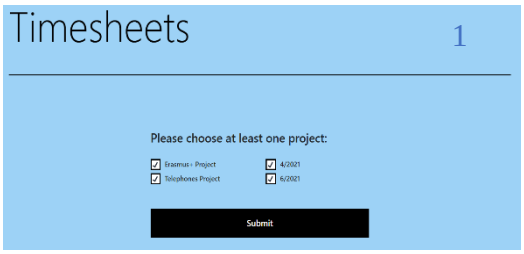

### 5.3.12 Timesheets



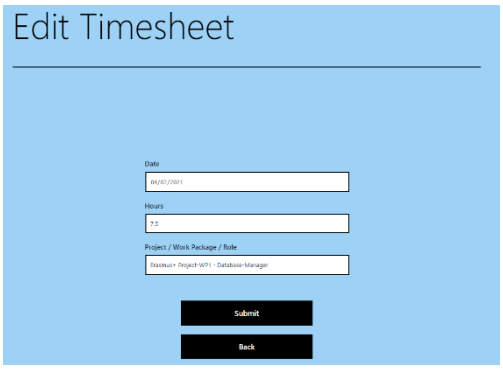


On the timesheet's page, you can:

1. Choose the projects and for which you want to see the timesheets
2. Sort the timesheets by date or by project
3. Edit your timesheets
4. Complete a timesheet by pressing the "New Timesheet" button
5. Download a CSV file with your completed timesheets

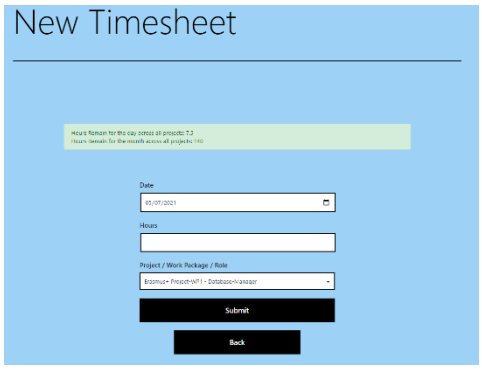


### 5.3.13 Edit Timesheet



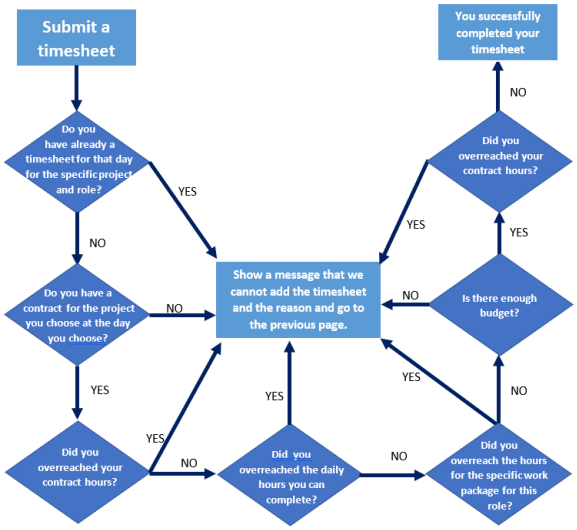

On this page, you can edit your timesheets. You only can change the hours for that specific day and project.

### 5.3.14 New Timesheet



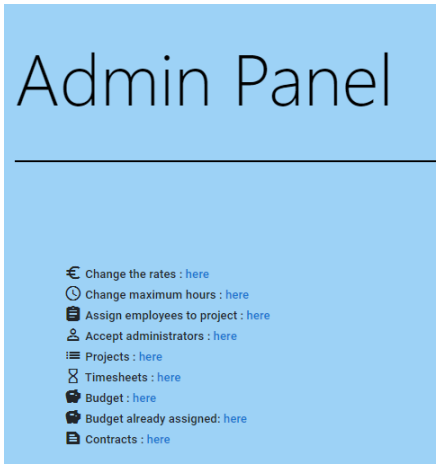

To add a new timesheet, you need to complete the date for the timesheet, the hours for that corresponding day, and choose the project work package and role from the dropdown menu. When you select a date, it will pop an alert about the daily and monthly hours that remain to complete for your timesheets.

*Figure 5.3 – New Timesheet*




### 5.3.15 Admin Panel

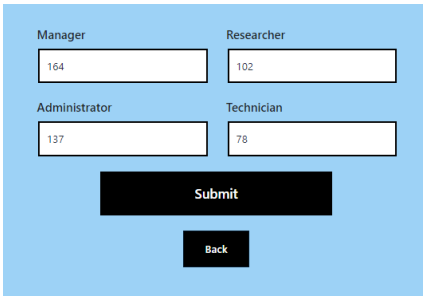
** You have access only if you are an administrator**




In the admin panel, you have access to:

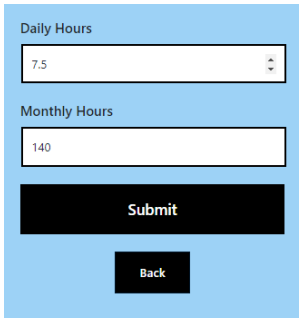
1. Change the rates
2. Change the maximum hours per day or month
3. Assign employees to projects
4. Accept other administrators
5. See all the projects
6. See the budget based on timesheets
7. See the budget that is already assigned
8. See all the contracts

#### 5.3.15.1 Rates



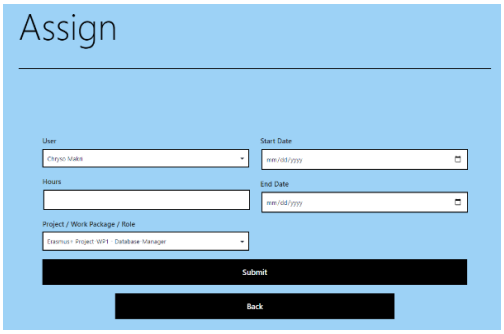

You can change the rates of each role

#### 5.3.15.2 Hours



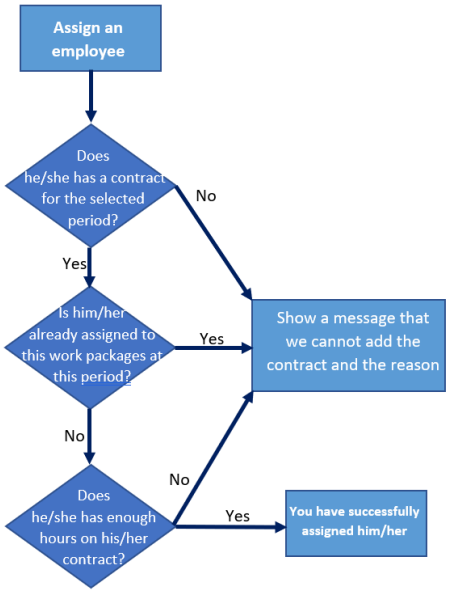

You can change daily and monthly hours that employees can complete on timesheets.

#### 5.3.15.3 Assign employees

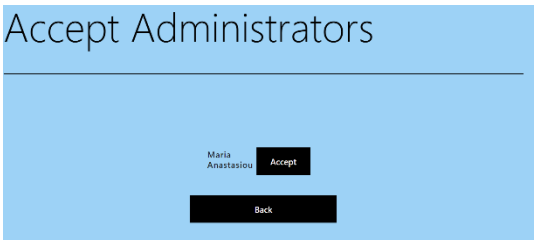



You can choose an employee and complete this form to assign him/her to a specific work package of a project he/she has a contract on. You must complete the start and end date, hours, and select the project, the work package, and the role from the dropdown menu.

*Figure 5.4 – Assign*

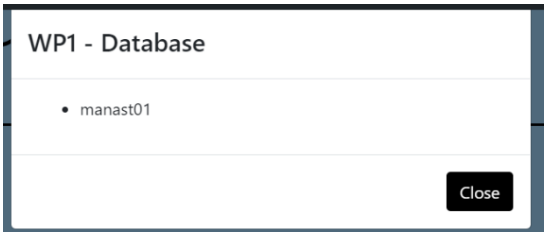



#### *5.3.15.4 Accept Administrators*



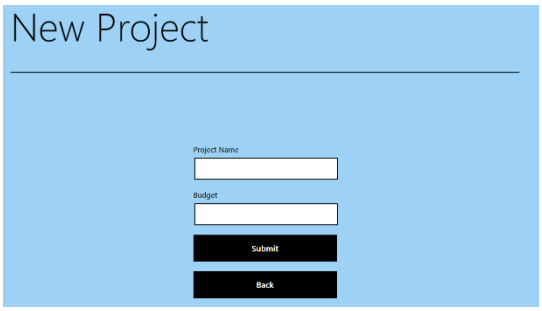

On this page, you can accept other users as administrators.

#### *5.3.15.5 Projects and Work packages*



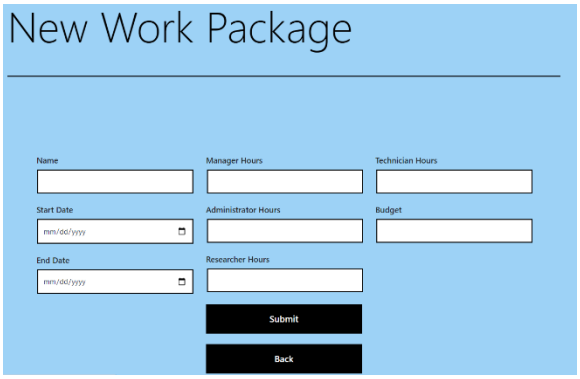

You can see all the projects and work packages in the same form as in 5.3.6, 5.3.7, 5.3.8, but you can also add new projects and work packages, and you can see who is assigned to each work package.

### 5.3.15.6 New Project



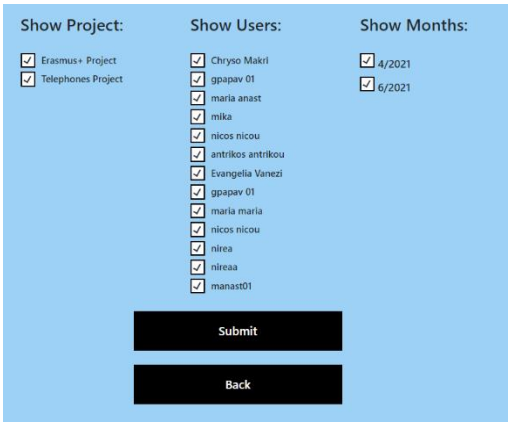

To add a new project, you only need to complete the project's name and budget.

### 5.3.15.7 New Work Package



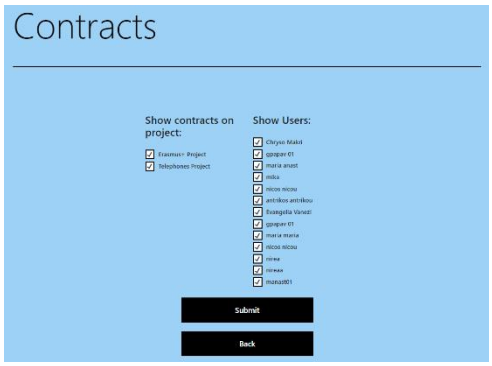

This is the form you have to complete in order to add a new work package. Work packages need a name, a start and end date, a budget, and the hours assigned to each role.

### 5.3.15.8 Timesheets




You can choose multiple project users and months for which you will see the timesheets. After submitting, everything is like in 5.3.12 but for many users.

### *5.3.15.9 Contracts*




You can choose multiple projects and users for which you will see the contracts. After submitting, everything is like in 5.3.10 but for many users.

# *CHAPTER 6*

## *EVALUATION*

CONTENT

*EVALUATION* ..... 59

6.1 INTRODUCTION 59
6.2 QUESTIONNAIRE 59
*6.2.1 Steps* *59*
*6.2.2 Questions* *60*
*6.2.3 Results* *60*
6.3 INTERVIEW 61

## *6.1 Introduction*

This chapter present the methodology we used for the evaluation of the web tool we developed. The evaluation was about the user's interaction with the system, the usability of the system, and the functionalities that have been offered. The evaluation by the researchers was happening during the development between the sprints. Nevertheless, we have some evaluations at the end of the project too. This evaluation was split into two phases. The first phase was a questionnaire that we sent to people who might use it in the future. In the second phase, we had some group interviews with the researchers with whom we have discussed at the beginning of this project to get their evaluation because they were with us throughout the project. This evaluation aimed to understand if this web tool will be easily adapted by the users, if it is user-friendly, and if everything was covered.

## *6.2 Questionnaire*

### 6.2.1 Steps

To make the questionnaire, we followed the same steps, as shown in Figure 3.9. First, we decided which information we needed to collect. Demographic data, experience with the specific topics, and how they feel about the web tool. At this time, we decided that our target responders will be people who know about

human-computer interaction and people who will use this web tool in the future. Afterward, we decided to create our questionnaire on google forms and send it to the respondents via email.

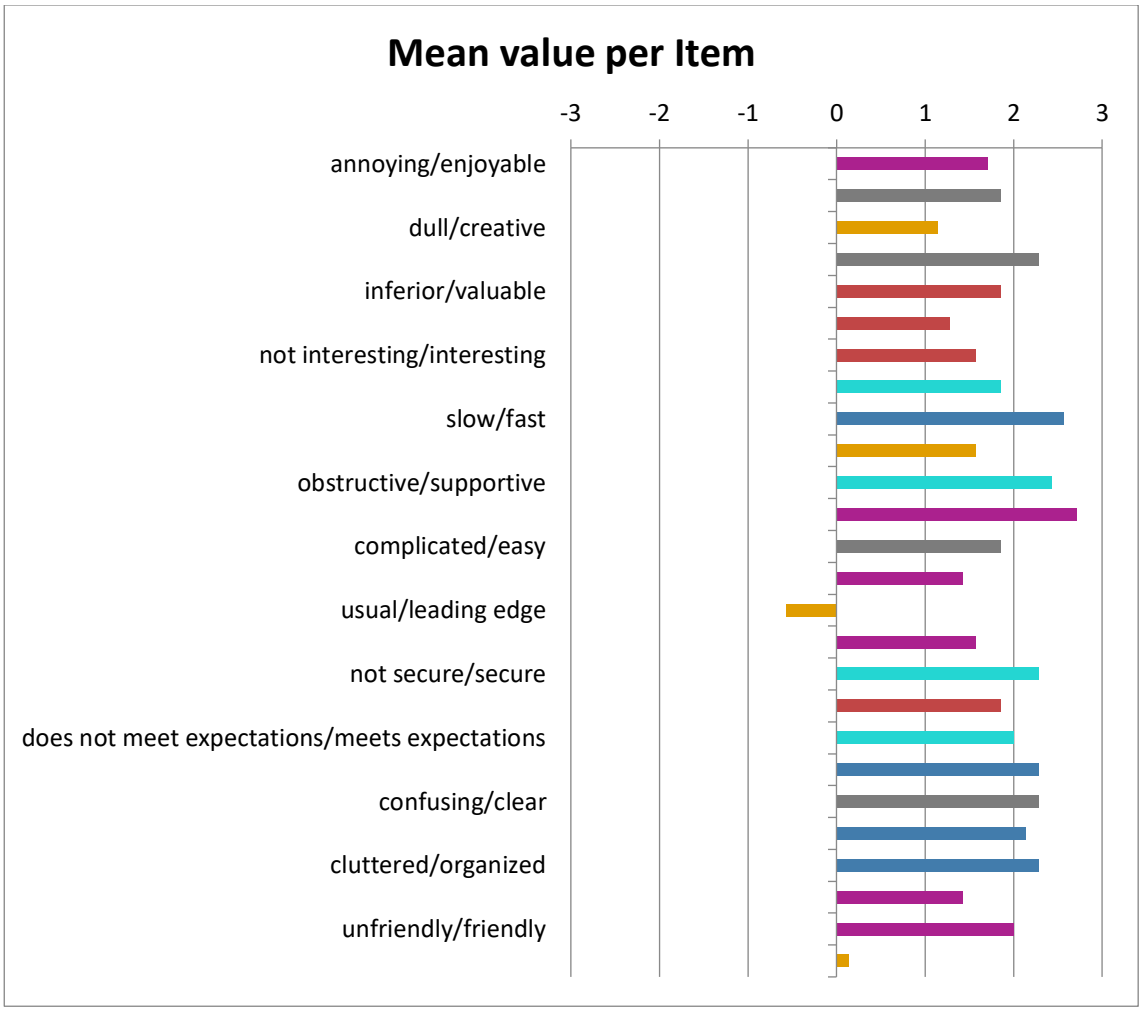
### 6.2.2 Questions

The questionnaire was developed in three main parts. First, there were some general instructions and explanations. The second part was about personal information and demographics, and the third part was the main body. In the second part, we have gathered information about the gender, age, and relevance of the person completing the questionnaire with the concepts of timesheets, budgeting, and contracts. In the third section, the body of the questionnaire, we had some statements about the web tool, and the responders had to answer with the Likert scale and then some other questions on which they had to choose between two words with opposite meaning for how they feel about the web tool.

### 6.2.3 Results

We had seven responses to the evaluation questionnaire from people with different knowledge. Based on the answers, the web tool will help them accomplish their work more quickly and improve their efficiency and effectiveness. Additionally, most of them stated that it will be easier to accomplish their financial tasks and that now they have to spend on average a day to complete them. Most of the responders believe that maybe it will be easy to learn the web tool. On a scale of 1-7, they found the web tool:

*Figure 6.1*




## *6.3 Interview*

We conducted extensive interviews with the two researchers being the product owner. During the last evaluation meeting, we had a discussion with another researcher as well. The three researchers worked at the web tool while we were in the meeting and provided some live feedback. They tried to get a full picture of the tool by making accounts, adding contracts, adding timesheets, and the admins by assigning people to work packages, etc. Through this process, we got some tips on the interface of the web tool.

The researchers preferred that the menu will be in a more compact and straightforward way, so we changed it to match their preferences. Additionally, they pointed out that they will prefer that the Admin Panel page is in the same format as

the employee dashboard to match it, so we changed that too. Moreover, in the process of adding a contract, there was a misunderstanding by some of the researchers about the hour's field. They thought that what was needed was the number of hours for each month when it was the total hours of the contract that they needed to complete. To overcome this problem, we changed the wording in that specific field.

Except from the interface tips, we got some extra functionalities that may be needed. An issue like that was that contacts could not be edited. Either if a contract was active or inactive, there was no edit button to edit it. That was found to be necessary by one of the participants who checked the wrong dates. Another thing that was stated during the interview was the process of assigning a work package to an employee if you are the administrator. After the assignment, you can only see who is assigned to each work package and not the hours and the duration of this assignment. That was something that all researchers agreed was necessary. The interview was very useful, feedback was collected and changes were applied.

# *CHAPTER 7*

## *CONCLUSION*

**CONTENT**

*CONCLUSION*.....63

7.1 CONCLUSION.....63
7.2. FUTURE WORK.....63

## *7.1 Conclusion*

My thesis aimed to develop a web tool that helps the researchers manage the financial aspect of Erasmus+ Strategic Partnerships projects. The use of a web tool for financial management will help the researchers be more efficient at their work and need less time to complete their financial tasks. We followed scrum, an agile methodology that helped us understand better what the researchers needed throughout the project because we had continuous communication, and they could try the web tool when something was ready. That way, if something else was needed, we could provide it. The final prototype is running on the web and has many features available. Any interested user can create an account and access it. Then users can navigate through the web application, add their contracts, see the budget and practice with the other functionalities.

## *7.2. Future Work*

The web tool we developed for this thesis is already functional and tested with projects and work packages. Nevertheless, there are always new functionalities that may be needed. During the interview with the researchers, we discussed some extra functionalities that will make the tool even more helpful and clear for them. A great future work will be to get some extra financial reports except from the budget and the timesheets that we have now. Additionally, something else that we would have done if we had time was to add the edit buttons everywhere

so the researchers can edit their contracts if they are employees or the assignments if they are administrators. Another functionality that might be useful was to let the administrator add contracts as well, and the researcher can accept it or deny it. Moreover, the tool can offer the possibility to become an Admin after you create the account.  Additionally, in a future update, we can add the milestones feature with some notifications.

## *Bibliography*


[1] Alan Dix, Janet Finlay, Gregory Abowd, Russell Beale. Human-Computer Interaction. 3rd edition, Prentice-Hall.

[2] Jacob Nielsen, Enhancing the Explanatory Power of Usability Heuristics, 1994.

[3] Donald Norman. The Design of Everyday Things. MIT Press, London

[4] Catherine Plaisant, Ben Shneiderman. Designing the User Interface: Strategies for Effective Human-Computer Interaction. 3th edition, Addison-Wesley.

[5] Jennifer Preece, Yvonne Rogers, Helen Sharp, Interaction Design: Beyond human-computer Interaction, John Wiley & Sons, Fifth Edition, 2019

[6] R. Pressman, Software Engineering: A Practitioner's Approach, 8th Edition, Mc-Graw-Hill Education, 2015.

[7] M. Sundararajan, M. Mahalakshmi Traditional SDLC Vs Scrum Methodology – A Comparative Study

[8] I. Sommerville, Software Engineering, 10th Edition, Addison-Wesley, 2016

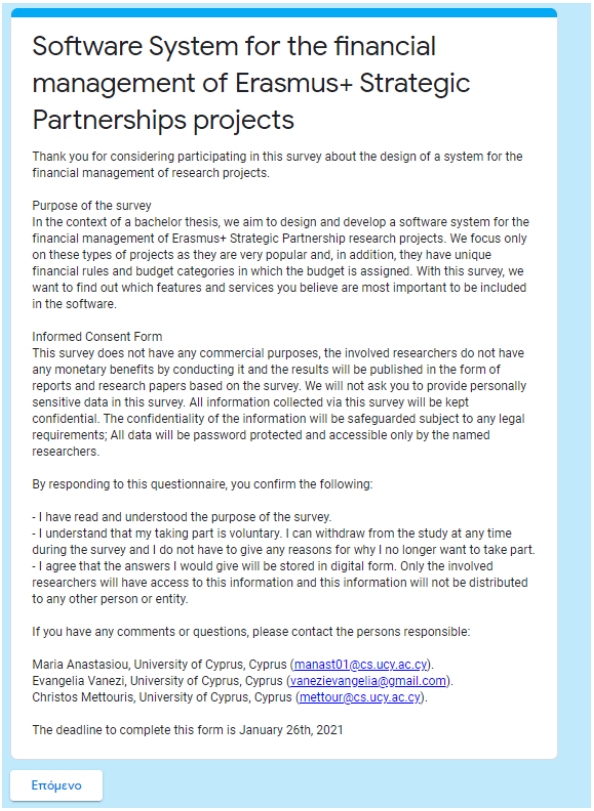
[9] H. van Vliet, Software Engineering: Principles and Practice, Third Edition, John Wiley & Sons, 2008.

[10] J. Whitten, L. Bentley και K. Dittman, Systems Analysis and Design Methods, 7th Edition, McGraw-Hill, 2007.

[11] International Journal of Emerging Technology and Advanced Engineering Platform: www.ijetae.com

# Appendix A

## Questionnaire



Software System for the financial management of Erasmus+ Strategic Partnerships projects

Thank you for considering participating in this survey about the design of a system for the financial management of research projects.

Purpose of the survey
In the context of a bachelor thesis, we aim to design and develop a software system for the financial management of Erasmus+ Strategic Partnership research projects. We focus only on these types of projects as they are very popular and, in addition, they have unique financial rules and budget categories in which the budget is assigned. With this survey, we want to find out which features and services you believe are most important to be included in the software.

Informed Consent Form
This survey does not have any commercial purposes, the involved researchers do not have any monetary benefits by conducting it and the results will be published in the form of reports and research papers based on the survey. We will not ask you to provide personally sensitive data in this survey. All information collected via this survey will be kept confidential. The confidentiality of the information will be safeguarded subject to any legal requirements; All data will be password protected and accessible only by the named researchers.

By responding to this questionnaire, you confirm the following:

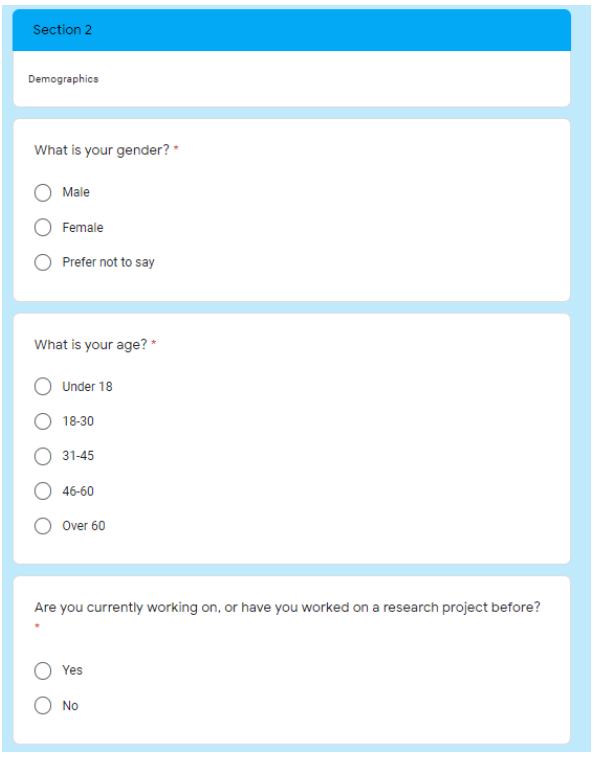
- I have read and understood the purpose of the survey.
- I understand that my taking part is voluntary. I can withdraw from the study at any time during the survey and I do not have to give any reasons for why I no longer want to take part.
- I agree that the answers I would give will be stored in digital form. Only the involved researchers will have access to this information and this information will not be distributed to any other person or entity.

If you have any comments or questions, please contact the persons responsible:

Maria Anastasiou, University of Cyprus, Cyprus (manast01@cs.ucy.ac.cy).
Evangelia Vanezi, University of Cyprus, Cyprus (vanezievangelia@gmail.com).
Christos Mettouris, University of Cyprus, Cyprus (mettour@cs.ucy.ac.cy).

The deadline to complete this form is January 26th, 2021

Επόμενο



Section 2

Demographics

What is your gender? *

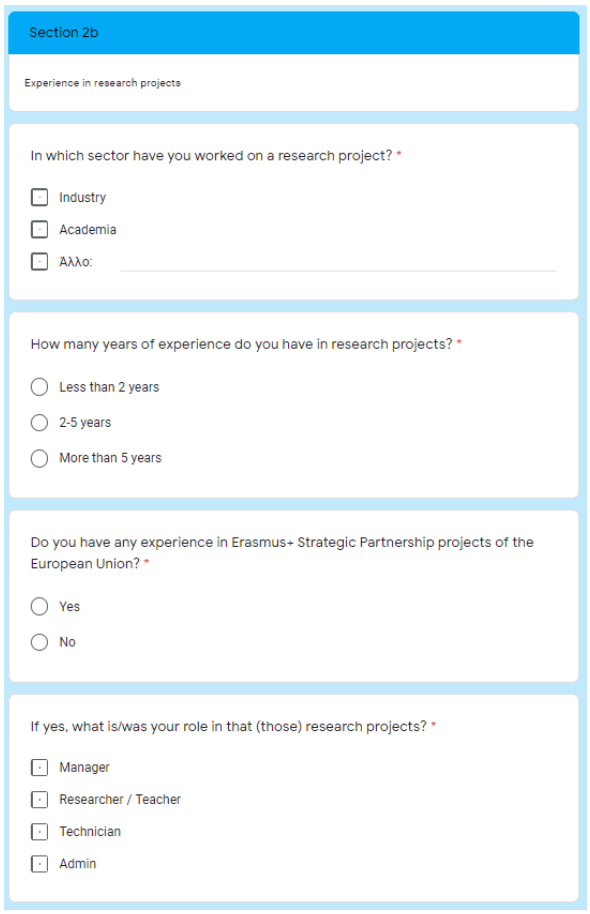
- Male
- Female
- Prefer not to say

What is your age? *

- Under 18
- 18-30
- 31-45
- 46-60
- Over 60

Are you currently working on, or have you worked on a research project before? *

- Yes
- No



Section 2b

Experience in research projects

In which sector have you worked on a research project? *

- [ ] Industry
- [ ] Academia
- [ ] Αλλο:

How many years of experience do you have in research projects? *

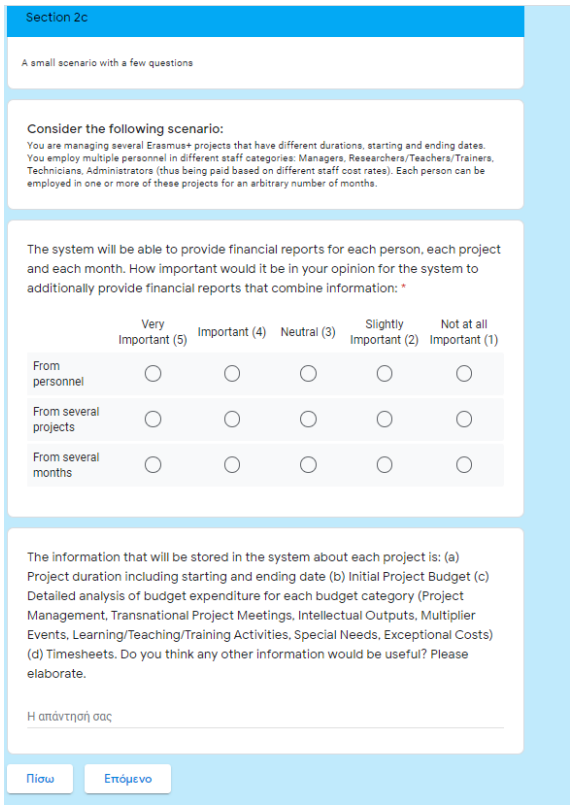
- ◯ Less than 2 years
- ◯ 2-5 years
- ◯ More than 5 years

Do you have any experience in Erasmus+ Strategic Partnership projects of the European Union? *

- ◯ Yes
- ◯ No

If yes, what is/was your role in that (those) research projects? *

- [ ] Manager
- [ ] Researcher / Teacher
- [ ] Technician
- [ ] Admin



Section 2c

A small scenario with a few questions

**Consider the following scenario:**

You are managing several Erasmus+ projects that have different durations, starting and ending dates. You employ multiple personnel in different staff categories: Managers, Researchers/Teachers/Trainers, Technicians, Administrators (thus being paid based on different staff cost rates). Each person can be employed in one or more of these projects for an arbitrary number of months.

The system will be able to provide financial reports for each person, each project and each month. How important would it be in your opinion for the system to additionally provide financial reports that combine information: *

| | Very Important (5) | Important (4) | Neutral (3) | Slightly Important (2) | Not at all Important (1) |
|---|---|---|---|---|---|
| From personnel | ◯ | ◯ | ◯ | ◯ | ◯ |
| From several projects | ◯ | ◯ | ◯ | ◯ | ◯ |
| From several months | ◯ | ◯ | ◯ | ◯ | ◯ |

The information that will be stored in the system about each project is: (a) Project duration including starting and ending date (b) Initial Project Budget (c) Detailed analysis of budget expenditure for each budget category (Project Management, Transnational Project Meetings, Intellectual Outputs, Multiplier Events, Learning/Teaching/Training Activities, Special Needs, Exceptional Costs) (d) Timesheets. Do you think any other information would be useful? Please elaborate.

Η απάντησή σας

Πίσω Επόμενο

## Section 3b

Questions

How important are the following features for you? For your convenience, please find again the description of each function listed below in the following link: https://drive.google.com/file/d/1HrKjlYs-DuX122xZQ24VBpLrAKiTzQfV/view?usp=sharing *

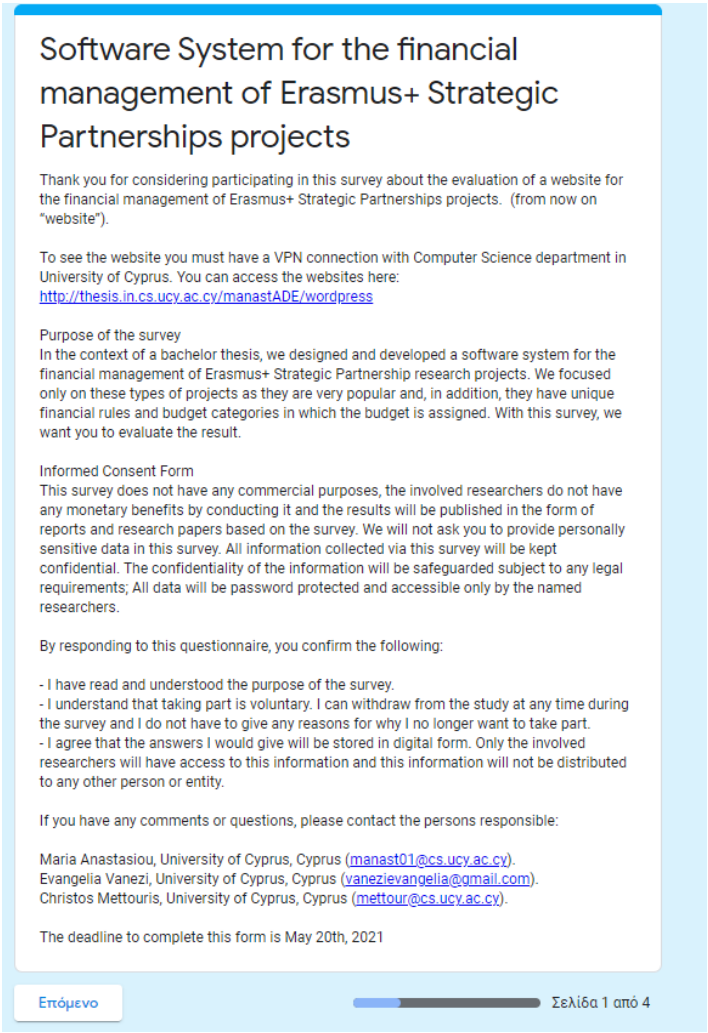
| | Very Important (5) | Important (4) | Neutral (3) | Slightly Important (2) | Not at all Important (1) |
|---|---|---|---|---|---|
| Autotracker | ◯ | ◯ | ◯ | ◯ | ◯ |
| Offline time tracker | ◯ | ◯ | ◯ | ◯ | ◯ |
| Budgeting | ◯ | ◯ | ◯ | ◯ | ◯ |
| Invoicing | ◯ | ◯ | ◯ | ◯ | ◯ |
| Project Dashboard | ◯ | ◯ | ◯ | ◯ | ◯ |
| Timeline Chart | ◯ | ◯ | ◯ | ◯ | ◯ |
| Computer using tracking | ◯ | ◯ | ◯ | ◯ | ◯ |
| Google synchronization | ◯ | ◯ | ◯ | ◯ | ◯ |
| Financial reports | ◯ | ◯ | ◯ | ◯ | ◯ |
| Absence analysis | ◯ | ◯ | ◯ | ◯ | ◯ |
| Attendance analysis | ◯ | ◯ | ◯ | ◯ | ◯ |

If you have any additional suggestions, feel free to share

Η απάντησή σας

# Appendix B

## Evaluation Questionnaire



Software System for the financial management of Erasmus+ Strategic Partnerships projects

Thank you for considering participating in this survey about the evaluation of a website for the financial management of Erasmus+ Strategic Partnerships projects. (from now on "website").

To see the website you must have a VPN connection with Computer Science department in University of Cyprus. You can access the websites here:
http://thesis.in.cs.ucy.ac.cy/manastADE/wordpress

Purpose of the survey
In the context of a bachelor thesis, we designed and developed a software system for the financial management of Erasmus+ Strategic Partnership research projects. We focused only on these types of projects as they are very popular and, in addition, they have unique financial rules and budget categories in which the budget is assigned. With this survey, we want you to evaluate the result.

Informed Consent Form
This survey does not have any commercial purposes, the involved researchers do not have any monetary benefits by conducting it and the results will be published in the form of reports and research papers based on the survey. We will not ask you to provide personally sensitive data in this survey. All information collected via this survey will be kept confidential. The confidentiality of the information will be safeguarded subject to any legal requirements; All data will be password protected and accessible only by the named researchers.

By responding to this questionnaire, you confirm the following:

- I have read and understood the purpose of the survey.
- I understand that taking part is voluntary. I can withdraw from the study at any time during the survey and I do not have to give any reasons for why I no longer want to take part.
- I agree that the answers I would give will be stored in digital form. Only the involved researchers will have access to this information and this information will not be distributed to any other person or entity.

If you have any comments or questions, please contact the persons responsible:

Maria Anastasiou, University of Cyprus, Cyprus (manast01@cs.ucy.ac.cy).
Evangelia Vanezi, University of Cyprus, Cyprus (vanezievangelia@gmail.com).
Christos Mettouris, University of Cyprus, Cyprus (mettour@cs.ucy.ac.cy).

The deadline to complete this form is May 20th, 2021

Επόμενο

Σελίδα 1 από 4

## Section 2

Demographics

What is your gender? *

- ◯ Male
- ◯ Female
- ◯ Prefer not to say

What is your age? *

- ◯ Under 18
- ◯ 18-30
- ◯ 31-45
- ◯ 46-60
- ◯ Over 60

Please state your experience in: *

| | No experience | Novice | Intermediate | Advanced | Expert |
|---|---|---|---|---|---|
| Completing timesheets | ◯ | ◯ | ◯ | ◯ | ◯ |
| Understand budget information | ◯ | ◯ | ◯ | ◯ | ◯ |
| Using a website to add your contracts | ◯ | ◯ | ◯ | ◯ | ◯ |
| Using this specific website | ◯ | ◯ | ◯ | ◯ | ◯ |

## Section 3

Using this website would enable me to acomplish my financial tasks more quickly *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Using this website would improve my performance in acomplishing my financial tasks *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Using this website would enhance my effectiveness in accomplishing my financial tasks *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Using this website would make it easier to accomplish my financial tasks *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Using this website would be useful in accomplishing my financial tasks *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Based on the current way I accomplish my financial tasks I believe that I am able to complete them *

- ◯ Within a few hours
- ◯ Within a day
- ◯ Within a few days
- ◯ Within a few weeks

I believe that it would be easier for me to accomplish my financial tasks without the use of the website *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

Learning to operate the website would be easy for me *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

I would find it easy to get the website to do what I want it to do *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

My interaction with the website will be clear and undrstandable *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

I would find the website to be flexible to interact with *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

It will be easy to me to becomme skillful at using the website *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

I would find the website easy to use *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Do you believe that it does not required a lot of mental effort to interactwith the website? *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Do you believe that you would revisit the website within a week's time if it was available for you to use? *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ○ | ○ | ○ | ○ | ○ | ○ | ○ | Strongly agree |

Do you believe that you would revisit the website regularly if it was available for you to use? *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

Do you feel positive about the website? *

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| Strongly disagree | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | ◯ | Strongly agree |

Πίσω Επόμενο Σελίδα 3 από 4

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| annoying | ○ | ○ | ○ | ○ | ○ | ○ | ○ | enjoyable |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| not understandable | ○ | ○ | ○ | ○ | ○ | ○ | ○ | understandable |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| creative | ○ | ○ | ○ | ○ | ○ | ○ | ○ | dull |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| easy to learn | ○ | ○ | ○ | ○ | ○ | ○ | ○ | difficult to learn |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| valuable | ○ | ○ | ○ | ○ | ○ | ○ | ○ | inferior |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| boring | ○ | ○ | ○ | ○ | ○ | ○ | ○ | exciting |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| not interesting | ○ | ○ | ○ | ○ | ○ | ○ | ○ | interesting |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| unpredictable | ○ | ○ | ○ | ○ | ○ | ○ | ○ | predictable |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| fast | ○ | ○ | ○ | ○ | ○ | ○ | ○ | slow |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| inventive | ○ | ○ | ○ | ○ | ○ | ○ | ○ | convetional |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| obstructive | ○ | ○ | ○ | ○ | ○ | ○ | ○ | supportive |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| good | ○ | ○ | ○ | ○ | ○ | ○ | ○ | bad |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| complicated | ○ | ○ | ○ | ○ | ○ | ○ | ○ | easy |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| unlikable | ○ | ○ | ○ | ○ | ○ | ○ | ○ | pleasing |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| usual | ○ | ○ | ○ | ○ | ○ | ○ | ○ | leading edge |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| unpleasant | ○ | ○ | ○ | ○ | ○ | ○ | ○ | pleasant |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| secure | ○ | ○ | ○ | ○ | ○ | ○ | ○ | not secure |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| motivating | ○ | ○ | ○ | ○ | ○ | ○ | ○ | demotivating |

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| meets expectations | ○ | ○ | ○ | ○ | ○ | ○ | ○ | does not meet expectations |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| inefficient | ○ | ○ | ○ | ○ | ○ | ○ | ○ | efficient |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| clear | ○ | ○ | ○ | ○ | ○ | ○ | ○ | confusing |

*

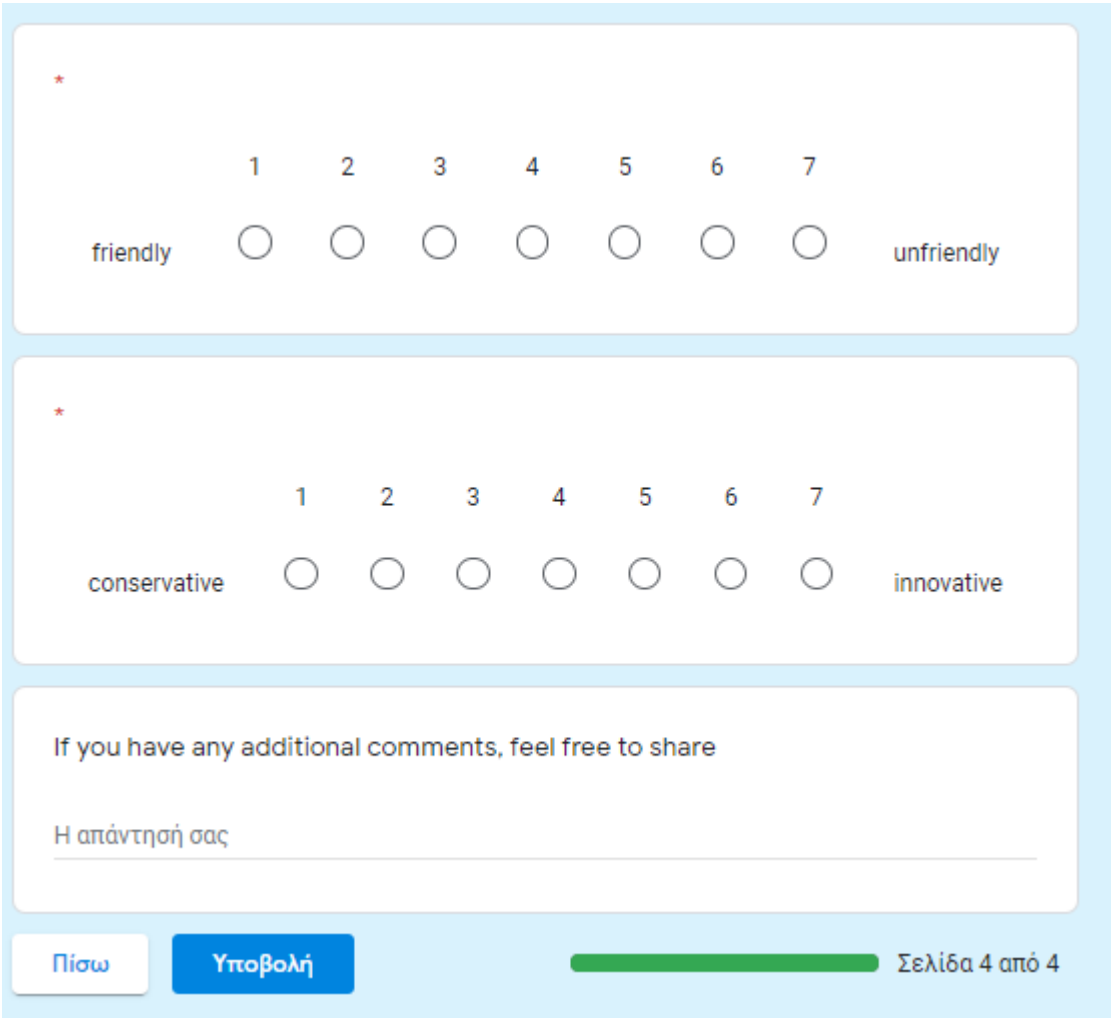
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| inpractical | ○ | ○ | ○ | ○ | ○ | ○ | ○ | practical |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| organized | ○ | ○ | ○ | ○ | ○ | ○ | ○ | cluttered |

*

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | |
|---|---|---|---|---|---|---|---|---|
| attractive | ○ | ○ | ○ | ○ | ○ | ○ | ○ | unattractive |




| UEQ Scales (Mean and Variance) | | |
|---|---|---|
| **Attractiveness** | 1,810 | 0,06 |
| **Perspicuity** | 2,071 | 0,29 |
| **Efficiency** | 2,321 | 0,06 |
| **Dependability** | 2,143 | 0,02 |
| **Stimulation** | 1,643 | 0,04 |
| **Novelty** | 0,571 | 0,08 |