Design and development of a management and control application for School Units

Panagiotis Loukopoulos
Matr. Numb: 1043796

Supervisor
Professor Nikolaos Avouris

Member
Spyros Denazis, Professor

Member
Michalis Xenos, Professor

Patras, 2024
To Evangelia and Chara
Special thanks

Initially, I would like to thank my supervisor, Mr. Nikolaos Avouris, for his guidance and collaboration both during the completion of this thesis and throughout my postgraduate studies. Additionally, I extend my gratitude to the co-supervisor, Professor Mr. Spyros Denazis, for the opportunity to collaborate with his research team through this specific postgraduate work. I particularly want to thank him warmly for the trust he showed me and the freedom he provided during the implementation period of the project. I appreciate all the instructors of the postgraduate HCI program and especially Mr. Michalis Xenos for his advice and our collaboration. Furthermore, I would like to thank all the members of the Network Architectures and Management group (NAM), especially Kostas Bakoulias and Kostis Trantzas, for their assistance and valuable advice. Also, thanks to Persefoni Bekiari for her invaluable help. Last but not least, I express my gratitude to Mr. Alexis Birbas for his valuable assistance and advice throughout my studies. I would also like to thank my family for their unwavering and continuous support throughout my studies. Finally, I thank all the people close to me who stand by me at every step and support me.
Περίληψη
Στην ανά χείρα διπλωματική εργασία αναλύεται η ανάπτυξη διεπαφής για την καταγραφή σχολικών αιτημάτων που σχετίζονται με τις κτιριακές υποδομές (συντήρηση, θέρμανση) με σκοπό την καλύτερη οργάνωση και διαχείριση από πλευράς Δήμων. Περιγράφονται οι βασικές λειτουργίες και όλα τα στάδια υλοποίησης καθώς και η ανάγκη δημιουργίας της συγκεκριμένης πλατφόρμας. Ακολούθως, γίνεται ανασκόπηση βιβλιογραφίας σχετικά με τον ανθρωποκεντρικό σχεδιασμό και τις μεθόδους που περιλαμβάνει. Επίσης γίνεται εφαρμογή των παραπάνω κανόνων. Στη συνέχεια παρουσιάζεται η αρχιτεκτονική του συστήματος και αναλύεται κάθε συστατικό της. Ακολουθεί η παρουσίαση των τεχνολογιών που χρησιμοποιήθηκαν και η υλοποίηση της πλατφόρμας. Επιπρόσθετα, εφαρμόζονται διάφορα είδη τεστ και γίνεται αναφορά στην Ανάπτυξη λογισμικού με χρήση Τεστ και τα οφέλη της. Έπειτα η διαδικασία και οι μέθοδοι αξιολόγησής της πλατφόρμας με πραγματικούς χρήστες, σε πραγματικές συνθήκες. Παρουσιάζεται ο επανασχεδιασμός με βάση τα ευρήματα της αξιολόγησης. Τέλος, παρουσιάζονται τα βασικά συμπεράσματα, οι περιορισμοί που αντιμετωπίσαμε καθώς και προτάσεις για μελλοντικές εκδόσεις της πλατφόρμας.

Abstract
The present dissertation analyses the development of an application for recording school issues regarding building infrastructure (maintenance, heating) aiming at better organization and management from municipalities. The basic functions and all stages of implementation are described, as well as the need to create this specific platform. Following that, a literature review is conducted on user-centered design and the methods it includes. Furthermore, the application of the above rules is demonstrated. Next, the system architecture is presented, and each component is analyzed. The presentation of the technologies used, and the implementation of the platform follows. Additionally, various types of tests are applied, and reference is made to test-driven development and its benefits. The evaluation process and methods of assessing the platform with real and representative users, under real conditions, are then discussed. Redesigning based on the evaluation findings is presented. Finally, the main conclusions are presented, the limitations faced are outlined, and suggestions for future releases are provided.
**Acronyms**

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>API</td>
<td>Application Programming Interface</td>
</tr>
<tr>
<td>CI/CD</td>
<td>Continuous Integration/Continuous Delivery</td>
</tr>
<tr>
<td>CRUD</td>
<td>Create, Read, Update, Delete</td>
</tr>
<tr>
<td>GDPR</td>
<td>General Data Protection Regulation</td>
</tr>
<tr>
<td>GUI</td>
<td>Graphical User Interface</td>
</tr>
<tr>
<td>HCD</td>
<td>Human Centered Design</td>
</tr>
<tr>
<td>HCI</td>
<td>Human Computer Interaction</td>
</tr>
<tr>
<td>HTTP</td>
<td>Hypertext Transfer Protocol Secure</td>
</tr>
<tr>
<td>IP</td>
<td>Internet Protocol</td>
</tr>
<tr>
<td>JSON</td>
<td>JavaScript Object Notation</td>
</tr>
<tr>
<td>MEAN</td>
<td>MongoDB, Express, Angular, Node.js</td>
</tr>
<tr>
<td>MVW</td>
<td>Model View and Whatever</td>
</tr>
<tr>
<td>NAM</td>
<td>Network Architectures and Management group</td>
</tr>
<tr>
<td>PACT</td>
<td>People, Activities, Context, Technologies</td>
</tr>
<tr>
<td>UCD</td>
<td>User Centered Design</td>
</tr>
<tr>
<td>URL</td>
<td>Uniform Resource Locator</td>
</tr>
<tr>
<td>US</td>
<td>User Story</td>
</tr>
<tr>
<td>UX</td>
<td>User Experience</td>
</tr>
<tr>
<td>VM</td>
<td>Virtual Machine</td>
</tr>
<tr>
<td>XHRs</td>
<td>XMLHttpRequests</td>
</tr>
</tbody>
</table>
Table of Contents

Contents

Special thanks .................................................................................................................. 6
Περιλήψη ......................................................................................................................... 7
Abstract .......................................................................................................................... 7
Acronyms ......................................................................................................................... 8
Contents .......................................................................................................................... i

1 Introduction .................................................................................................................. 11
  1.1 Significance of the Issue ........................................................................................ 11
  1.2 Objectives of the Study ......................................................................................... 12
  1.3 Methodology Approach ......................................................................................... 12
  1.4 Contribution .......................................................................................................... 13
  1.5 Structure of the Thesis .......................................................................................... 13

2 Design of Schools plugin ............................................................................................ 14
  2.1 Human Centered Design ....................................................................................... 14
  2.2 The PACT Framework ......................................................................................... 16
  2.3 Schools’ plugin & PACT Framework ...................................................................... 18
  2.4 Personas ................................................................................................................ 20
  2.5 Use Cases ............................................................................................................. 21
  2.6 Design mockups and prototypes ......................................................................... 29
  2.7 Interview with an expert ....................................................................................... 31
  2.8 Issue lifecycle ....................................................................................................... 33

3 Implementation of Schools plugin ............................................................................. 35
  3.1 The software architecture ..................................................................................... 35
    3.1.1 The presentation layer ..................................................................................... 35
    3.1.2 The application layer ..................................................................................... 35
Design and development of a management and control application for School Units

3.1.3 The database layer .................................................. 36

3.2 Analysis of database ................................................................................. 36
  3.2.1 Mongoose schemas - smart data models .................................................. 36

3.3 Angular version Upgrade ...................................................................... 43

3.4 Technologies Used ............................................................................... 43
  3.4.1 Libraries and packages ........................................................................ 46
  3.4.2 Gitlab – SourceTree .......................................................................... 49

3.5 Hosting in NAM facilities ....................................................................... 50

3.6 Application pages & Functionalities ......................................................... 50
  3.6.1 Functionalities and Permitted Actions of Directors ..................................... 50
  3.6.2 Functionalities and Permitted Actions of School Manager ....................... 58
  3.6.3 Functionalities and Permitted Actions of Department Users .................... 64

4 .............................................................................................................. 67

Testing, Evaluation & Redesign .................................................................. 67

4.1 Angular testing .................................................................................... 67
  4.1.1 The testing pyramid in Angular .......................................................... 67

4.2 Unit Tests vs. Integration Tests vs. End-to-End Tests ......................... 69
  4.2.1 Implementation of tests ...................................................................... 71

4.3 Heuristic Evaluation ............................................................................ 75
  4.3.1 Evaluation results .............................................................................. 76

4.4 Think – Aloud Evaluation .................................................................... 78
  4.4.1 Participants ........................................................................................ 78
  4.4.2 The Environment ............................................................................... 79
  4.4.3 Conducting the Session ..................................................................... 79
  4.4.4 Think-Aloud Evaluation Results ......................................................... 82

4.5 Redesign based on evaluation results ..................................................... 83

4.6 Ecological validity of user study ............................................................ 88

5 .............................................................................................................. 90

Conclusions ............................................................................................... 90

5.1 Benefits of Using the Application .......................................................... 90

5.2 Limitations ............................................................................................ 91

5.3 Future Prospects .................................................................................. 91

Bibliography - References .......................................................................... 95
Design and development of a management and control application for School Units
Figure 1: Principles of Human-Centered Design [40] ................................................................. 15
Figure 2: The PACT Framework .................................................................................................. 17
Figure 3: Usability Heuristics [41] ............................................................................................ 31
Figure 4: Issue lifecycle .............................................................................................................. 34
Figure 5: MEAN stack architecture [44] ..................................................................................... 44
Figure 6: Hosting architecture .................................................................................................... 50
Figure 7: Registration email ........................................................................................................ 51
Figure 8: Registration form .......................................................................................................... 51
Figure 9: Login form .................................................................................................................... 51
Figure 10: Issue creation page ..................................................................................................... 52
Figure 11: Issues page ................................................................................................................ 53
Figure 12: Issue history page ....................................................................................................... 54
Figure 13: Issue history page ....................................................................................................... 54
Figure 14: Account settings page ................................................................................................ 55
Figure 15: Notifications creation page ......................................................................................... 55
Figure 16: Notifications page ...................................................................................................... 56
Figure 17: Issues calendar ........................................................................................................... 56
Figure 18: Issue view in calendar ................................................................................................ 57
Figure 19: Sensors monitoring page ............................................................................................ 57
Figure 20: Login page .................................................................................................................. 58
Figure 21: Overview page ............................................................................................................ 58
Figure 22: School Units page ...................................................................................................... 59
Figure 23: School Users creation ................................................................................................. 59
Figure 24: Edit school units’ information .................................................................................... 60
Figure 25: School Users .............................................................................................................. 60
Figure 26: Edit school user information ...................................................................................... 61
Figure 27: School Issues page .................................................................................................... 61
Figure 28: Edit School Issue page ............................................................................................... 62
Figure 29: Search Issue ................................................................................................................ 63
Figure 30: Fuel order ................................................................................................................... 63
Figure 31: School Issues ............................................................................................................. 64
Figure 32: Edit School Issue ....................................................................................................... 65
Figure 33: Search School Issue .................................................................................................. 65
Figure 34: The testing pyramid in Angular [45] ....................................................................... 66
Figure 35: Unit testing results ..................................................................................................... 68
Figure 36: END-2-END testing with cypress ............................................................................ 72
Figure 37: Jakob Nielsen's 10 general principles for interaction design [39] ............................... 73
Figure 38: View and editing buttons ......................................................................................... 76
Figure 39: Submit and cancel buttons ....................................................................................... 77
Figure 40: Login page ................................................................................................................ 77
Figure 41: Search Issue page ..................................................................................................... 77
Figure 42: View, editing and activation user page ................................................................. 83
Figure 43: Loading spinner ....................................................................................................... 84
Figure 44: Cancel and submit buttons ....................................................................................... 84
Figure 45: School Units redesign ............................................................................................... 85
Figure 46: Users page redesign .................................................................................................. 85
Figure 47: Create pdf ................................................................................................................. 86
Figure 48: Issues from other departments ............................................................................... 86
Figure 49: Create pdf ................................................................................................................. 87
Figure 50: Add issue ................................................................................................................... 87

Panagiotis Loukopoulos – Matr. Number: 1043796
Design and development of a management and control application for School Units

Figure 51: Create alert ........................................................................................................................................ 88
Figure 52: New colour palette ........................................................................................................................ 92
Figure 53: New colour palette ........................................................................................................................ 93
Figure 54: New colour palette ........................................................................................................................ 93
Design and development of a management and control application for School Units
Table List

Table 1: Backend libraries and packages ........................................................................................................... 46
Table 2: Frontend libraries and packages ........................................................................................................... 47
Design and development of a management and control application for School Units
1

Introduction

1.1 Significance of the Issue

In today's era of rapid technological advancement, there is a lack of a comprehensive digital strategy in education. Therefore, digital transformation is deemed necessary in the administration of school units so that they can derive maximum benefits from digital interventions and be capable of integrating into the modern digital world [1]. "The Mayor of Athens, X. Doukas, emphasizes in an interview, 'There is no organized recording for the schools.' Through meetings and interviews with both municipalities and school directors, we identified the need to create a platform aimed at the immediate recording of school-related issues and facilitating communication with municipalities.

According to current situation, a school unit director who wanted to submit a request, had to first approach the school committee president or the relevant department of the municipality. This process was carried out using an email and often through telephone communication. Additionally, on the municipality's side, there was no allocation of requests to the relevant department; instead, all requests from all departments of the municipality were received, leading to unnecessary paper usage.

As we understand, this was a chaotic, time-consuming, and unstructured process without proper recording and history. According to Law 4692/2020, school committees will be abolished in January 1, 2024, resulting in the exclusive responsibility of municipalities and directors. For this reason, we did not include the school committee president as a user of the system. However, if requested, we can easily incorporate them into the system.
1.2 Objectives of the Study

From section 1.1, the need for a new request management mechanism arises, specifically the development of an issue management platform for school-related requests. This platform will serve as an extension of the existing system (senseCity). It is a well-tested platform with a successful track record over the years in large municipalities, such as the Municipality of Patras and the Municipality of Aigialeia, where citizens can record issues addressed to the municipality. The design and development of such issue management platform following a user centered approach is described in this thesis. This is a design based research project and the research question to address is to evaluate with typical users the developed prototype.

1.3 Methodology Approach

Our goal was to create a comprehensive solution for recording requests from school units. Leveraging the idea of request management by citizens through the senseCity platform and its capabilities, we decided to develop the School platform as an extension of senseCity. The School plugin constitutes a complete solution that can be utilized and integrated into the municipality's organizational structure that possesses senseCity. Following that, it is feasible for municipalities that do not have senseCity to utilize the School platform according to their own organizational structure. Emphasizing human-centered design and following its methods, we applied the rules of the PACT (People, Activities, Context, Technologies) framework, created personas, and developed use cases to capture the requirements of our system.

We proceeded with designing mockups and prototypes and conducted interviews with future users of the system to gain insights into existing processes and regulations, both from the perspective of municipalities and school units. Following that, we moved on to the Architectural design of the Schools plugin.

The platform was implemented using modern web technologies such as MEAN (MongoDB, Express, Angular, Node.js) stack. Subsequently, testing of the platform was conducted with the goal of providing a stable and competitive solution.

Finally, the platform underwent evaluation with real users, in real-life scenarios within their environment. Considering the valuable results of this assessment, a redesign and completion of the platform were carried out.
1.4 Contribution

The current thesis was conducted in parallel with my work in the Network Architectures and Management group (NAM). In addition to the team's primary objectives for the initial platform release, I took the initiative within the scope of this work to develop features that we will integrate into future versions. These features include the notification system, a dashboard for monitoring data from sensors, and the capability for a user with the role of School Manager to create fuel orders based on the fuel inventory of every school unit.

1.5 Structure of the Thesis

The master thesis is structured as follows: Chapter 2 presents and applies concepts and methods of human-centered design. The end user and their needs are at the center of our analysis. In Chapter 3, the architectural design of the platform is analyzed. Also, it introduces the technologies used, and the system implementation is carried out. Chapter 4 presents and applies various types of testing, such as unit testing and end-to-end testing. Furthermore, the system is evaluated using methods that focus on the human end user and their needs. Additionally, a redesign of the system is performed, taking into consideration the valuable results of the evaluation. Finally, Chapter 5 summarizes the conclusions of the thesis, discusses the limitations faced, and presents suggestions for future versions of the system.
2

Design of Schools plugin

2.1 Human Centered Design

The Human-centered design (HCD) process is a fundamental principle in the field of human-computer interaction. In all phases of the design, our focus is on the system's users and the tasks they perform with it, starting from the initial design stages. Throughout the iterative design the system (simulated, prototype, and real) undergoes a circle of modification, testing, further modification, more testing, and this loop repeats [2]. This iterative design process allows for the improvement of user interaction and the customization of the system to meet their needs.
According to Donald Norman [3][4][5], human-centered design is guided by four basic principles:

- **People-centered**: The primary focus is on people and the specific context to create solutions that are truly appropriate for their needs and preferences.

- **Understand and solve the right problems, the root problems**: It is important to dig deep and address the root problems rather than just treating the symptoms.

- **Everything is a system**: Seeing everything as a system of interconnected parts emphasizes the importance of understanding the holistic nature of the design problem and its various components.

- **Small and simple interventions**: By adopting an iterative approach, designers should avoid rushing to find solutions. Instead, they should experiment with small, simple interventions, learning from each one and gradually building more effective and sophisticated solutions. Continuous prototyping, testing, and refinement are essential to ensure that solutions align with the specific needs of the people at the centre of the design process.

The Human Centered Design approach applied to the Schools platform ensures that it meets the needs and preferences of its users. Enhancing empathy for users, delving into their motivations, challenges and behaviors to identify key issues and areas for improvement.
In the ideation phase, creativity and open-mindedness lead to a range of potential solutions aimed at addressing the identified problem. We explore various avenues to enhance the platform's functionality and user experience (UX).

Subsequently, the prototyping stage involves creating physical or digital models of proposed solutions, offering a tangible way to visualize and evaluate concepts, providing a foundation for further refinement.

The final step is testing these prototypes with real users, collecting feedback and conducting usability tests to ensure effectiveness and user-friendliness. This iterative process guarantees the product aligns with user needs and expectations.

The user-centered approach ensures that the School platform continually evolves in response to the real needs and preferences of its intended audience. Through repeated testing, feedback collection, and refinement, the platform consistently improves its design, remaining user-centric, intuitive, and in line with evolving user requirements.

2.2 The PACT Framework

The PACT framework is a user-centered design framework that guides the design and development of interactive systems, including web applications like issue management platforms. It emphasizes the importance of understanding users, their activities, the context in which the application is used, and the enabling technology [6]. This framework is used to analyse how these components interact and impact the overall user experience. The goal of PACT analysis is to create efficient and satisfying interactive systems by understanding users' needs, preferences, and behaviours [7]. By examining each PACT component in detail, designers can identify potential improvement issues and make informed decisions to optimize the system [8]. The iterative nature of PACT analysis recognizes that user needs, and system requirements may evolve and emphasizes on the importance of continuous feedback, adjustment, and refinement throughout the project.
People:

People are at the heart of the interactive system's design, and their characteristics, needs, preferences, and behaviors are critical elements. This involves recognizing factors like age, gender, educational background, technical proficiency, as well as psychological elements like motivation, perception, and cognition. It is essential to acknowledge how these factors can substantially influence the design and usability of technology, considering the physical, mental, and social contrasts among individuals and how these distinctions may evolve across different contexts and over time.

Activities:

When focusing on activities within the PACT framework, it is crucial to consider the primary objectives of the system, temporal aspects, cooperative elements, complexity, and the nature of the content. This entails understanding the key tasks and activities that users will engage in when using the web application and designing features and functionality that effectively facilitate these tasks.

Context:

In the context dimension of the PACT framework, it is vital to grasp the surroundings in which the web application will operate. This requires considering the physical environment, social context, and organized context. Understanding these factors is essential for designing a web application that harmonizes effectively with its intended context of use.
Design and development of a management and control application for School Units

**Technologies:**
Within the technologies dimension of the PACT framework, it is imperative to comprehend the technologies employed to underpin the web application. This encompasses both the hardware and software technologies in use. It is essential to consider aspects like input methods, output mechanisms, communication tools, and content management to ensure the web application's seamless functionality and alignment with technological requirements.

2.3 **Schools’ plugin & PACT Framework**

The design approach of the school plugin is closely aligned with the principles of the PACT framework, establishing a strong connection to the requirements and preferences of its users.

**People:**

- The main groups targeted are School Directors, Municipality/department users, admin user (School Manager).
- There is a need to support various languages.
- There is a need to incorporate inclusive design to accommodate people with color blindness.
- Access for any additional parties is not possible, such as cleaning staff, teachers, which requires a password to log in.

**Activities:**

- The general purpose of the system is to record issues and requests that exist in school units.
- All Users must provide login credentials.
- All Users can change the visuals on the platform according to their liking (different language support).
- School Manager must perform CRUD (Create, Read, Update, delete) operation at components: school units and users possess the role of Director.
- School Manager should add/edit/delete information related to school and request information in the system.
• School Manager can search for a specific request, see its history, its location, and be able to comment on it.
• School Manager can see the Fuel Stocks in every school unit and to place an order.
• School Manager can send an issue in the corresponding department. (cleaning Dep)
• Directors can report issues and problems located in school units directly.
• Directors can add/close/cancel alerts that they have created.
• Directors can monitor data from sensors like oil level.
• Directors can enter the general information of the school unit.
• Directors can search for a request they've made, see its history, and be able to comment on it.
• Department users can add/edit/delete information related to a specific school request.
• Department users can create a new issue.
• Department users can search for a specific issue.

Context:

• The issue management platform is designed to integrate seamlessly with the diverse physical environments of both school buildings and municipal offices. It accommodates the unique features of these spaces such as location and number of floors. The system takes into consideration factors such as noise levels, temperature, environmental factors and fuel level. By adapting to this environment, the platform ensures a comfortable and efficient experience for users.
• Socially, access to the platform can take place at any time during the working hours of both employees and managers as well as administrators, without specific time restrictions.
• From an organizational perspective, the context pertains to determining which platform features are accessible to specific individuals.

Technologies:

• Ensuring platform responsiveness and accessibility from various devices, including tablets and mobile phones, is crucial. There is a need to support various languages.
• Establishing communication with a central database is imperative to validate login credentials and control the information pertaining to both employees (managed by admins) and directors (accessible to both admins and employees).

• It should be always consistently available and accessible online, irrespective of the chosen browser and operating system.

• The system should be intuitively usable for people unfamiliar with such systems. This requires the inclusion of action comments, notification messages, confirmations, help sections and other handy elements to ensure ease of use.

2.4 Personas

According to the provided details, we present three personas for our app, accompanied by additional relevant information:

1. **Persona name:** School Manager
   **Age:** 30-50
   **Profession:** Administrative Employee, possibly an Engineer.
   **Role:** Creation and editing of school units, creation and editing of users managing school units (Directors), user supervision, supervision of school units and requests, editing and submission of requests to the appropriate departments of the Municipality, creation of fuel orders.
   **Key Characteristics:** Experienced, analytical, administrative skills, proficient in communication and organization.

2. **Persona name:** School Director
   **Age:** 40-60
   **Profession:** Educational Professional
   **Role:** Creation, supervision, and management of requests, input of school unit details, creation of notifications (e.g., fire extinguisher replenishment), monitoring of consumptions, and environmental data (e.g., boiler fuel level).
   **Key Characteristics:** Experienced, analytical, strong organizational skills and leadership qualities.
3. Persona name: Department User  
Age: 30-50  
Profession: Engineer or related expertise in the respective department.  
Role: Supervision and management of requests in a selected category, creation and search of requests.  
Key Characteristics: Experienced, analytical, strong organizational and communication skills, attention to detail. They are responsible for ensuring the smooth flow of requests and communication within their area of expertise.

2.5 Use Cases

Before we dive into creative platform design, it's crucial to gain a deep insight of the people using it, the activities taking place, the context of those activities, and the technology implications. This process is based on the PACT analysis model, which we've already completed. From this understanding, we can generate system requirements. This step focuses on what users want to do, how they'll do it, and why, while also considering any problems they might encounter. In this design phase, we perform requirements analysis, which will guide us in shaping the service platform during the ideation process. These requirements are divided into two categories: functional (what the platform must have) and non-functional (quality considerations essential for usability and acceptability).

We'll capture the system's functional requirements using user stories. Each user story (US) will start with a brief description of a user's goal, followed by a more detailed account of the hypothetical steps they take to achieve that goal. With this approach we can precisely define the features and functionalities the platform must provide to meet users' needs.

Department User

US 1: Supervision and issue management

The user "Department User" logs into their account. They navigate to the section "School Unit Issues," where they can see the number of requests on the map. They select the option to export the issues as a PDF file. Then, they send the PDF file to the appropriate municipal team, for example, the "Cleansing Service."
The municipal team processes the requests and returns the PDF file to the Department User, having recorded which issues have been resolved and some comments. The Department User goes to the "Search" section, enters the unique request number in the "Issue Number" field, and clicks on the "Search" button.

Next, they select the request that appears, press the "Manage" button, view the request's history, and change the request's status from "In Progress" to "Completed." They also choose one of the possible status options for the completed request, such as "Restoration," "Wrong Report," "Not Restored/Rejected," "Already Reported."

They provide comments on the request based on the team's observations and upload any relevant files. Finally, they click the "Submit" button and continue this same process for the remaining requests that are still marked as "In Progress." Ater that they logout the system.

**Requested Requirements:**

1) **Register option.**
   - Account Name
   - Surname
   - Password
   - Position

2) **Login option**
   - Account Name
   - Password

2) **Exporting Problems to PDF.**

3) **Customized Sorting of Requests.**

4) **Customized Search for Requests.**

5) **Viewing Requests in List and Map View.**

6) **Ability to Modify a Request.**
   - Status (Confirmed, In Progress, Completed)
   - Assignment to the Appropriate Department (e.g., Cleaning Department)
   - Priority (High, Normal, Low)
   - Address (Selection on the Map)
   - Adding Comments


- Uploading Files (PDF, PNG)

7) Viewing Request History.

8) Logout capability

School Director

US 1: Registration, login and personal details.

The user with the role "School Director" registers on the platform. Subsequently, they log into their account, navigate to the "Account Settings" section, and fill in their account details.

US 2: Creation and request editing.

The user with the role "School Director" logs into their account, goes to the "Create Request" section, and submits a request. They then proceed to the "Issues" section, use search filters to locate their request, and select it. They can view the complete history of their request, including information such as its status, assignment to a department, and comments.

Finally, the School Director adds a comment and uploads a document, then clicks the "Submit" button.

US 3: Alert creation and fuel availability

The user with the role "School Director" logs into their account, goes to the "Notifications" section, selects the "Add" button, and creates a new notification, for example, "Fire Extinguisher Replenishment," by filling in the necessary details. Then, they choose to "Submit" the notification.

Next, they navigate to the "Monthly Notifications" section to check the notifications that are set to expire in the current month. Finally, they go to the "Dashboard" section to view the available quantity of fuel for the boiler. After that, they logout the system.

Requested Requirements:

1) Login
   - Account name
   - Password

2) Register option:
   - Account Name
   - Surname
• Password
• Position

3) Entering information into the system:
• Region
• Municipality
• Address (selection on the map or manual entry in the form)
• Postal Code
• Phone Number
• Number of School Unit Floors
• Number of Basement Floors

4) Creating a Request:
• Select Problem Category
• Select Problem Subcategory
• Add Comment
• Attach a File

5) Filter Selection for Search:
• Sorting (Newest First, Oldest First)
• Status (All, Confirmed, In Progress, Completed)
• Problem Category - Problem Subcategory
• Date Created Range

6) Viewing Request History:
• Status
• Assigned Department
• Problem Image
• Address
• Comment
• Problem Category

7) Comment on the Request:
8) Creating a Notification:

- Name
- Expiration Date
- Comments
- Recurring

9) Customized Search for Notifications:

- Status (Open, Completed, Canceled, Expired)
- Date Created Range

10) Viewing Monthly Notifications.

11) Changing the Status of Notifications.

12) Monitoring Consumption (fuel, electricity, water).

13) Environmental Data Monitoring (Air Quality, Temperature, Humidity).

14) Logout capability

School Manager

US 1: Creation, Editing of School Units, and Assignment of Users to Them.

The user with the role "School Manager" logs into their account and goes to the "School Units" section. They select the "Add" button and fill in the school unit's information, including its name, department type, and the School Director's email. After filling in the details, they click "Submit."

An email with a registration link is sent to the School Director's email. The School Director follows the link, registers, logs into their account, and provides their general account information.

At this point, the School Manager returns to the "School Units" section and uses filters to search for the desired school unit. They select the "Edit School Unit" button and modify the school unit's name before clicking "Submit."
Finally, the School Manager navigates to the "Users" section and uses filters to search for the user (Director) they wish to modify. They select the "Edit User" button and change the account name before clicking "Submit."

**US 2: Supervision and Management of Issues in the Selected Category.**

The user with the role of "School Manager" logs into their account, goes to the "Issues" section, selects the desired date range, the problem category, and clicks the "Refresh" button. From the displayed requests, they select the first one and then click the "Manage" button. They proceed to modify the request as follows:

1. Change the request status from "Confirmed" to "In Progress."
2. Assign it to the appropriate department.
3. Change the priority from "Normal" to "High."
4. Add a comment.
5. Upload a file.
6. Click the "Submit" button.

At this point, the request is forwarded to the relevant department for resolution. The School Manager continues this same process for the remaining requests that appear.

**US 3: Fuel Stocks and Order Creation.**

The user with the role of "School Manager" logs into their account and navigates to the "Heating" section. Then, using the search filters, they input the desired fuel quantity for each Secondary School Unit. Finally, they select the "Export Order" button and send the XLSX file to the relevant municipal user.

**Requested Requirements:**

1) Login / Register option
   - Account name
   - Password

2) Creating School Units.
   - Name
   - Type of Department (Kindergartens, Primary School Unit, Secondary School Unit)
• School Unit Email

3) Editing School Units.
• Name
• Email
• Address
• Geographic Longitude
• Geographic Latitude
• Postal Code

4) Creating Users Responsible for School Units.
• Sending an Email with the Registration Process.

5) Editing Users with the Role of Director.
• Account Name
• Email
• Position
• User Role
• School Unit
• Name (Optional)
• Surname (Optional)
• Password Change (Optional)

6) Viewing Requests in List and on Map.

7) Selecting search filters:
• Sorting (Newest first, Oldest first)
• Problem Category - Subcategory of Problem
• Creation Date Range

8) Ability to modify a request:
• Status (Confirmed, In Progress, Completed)
• Assignment to the relevant department (e.g., Cleaning Department)
• Priority (High, Normal, Low)
• Address (select from the map)
• Adding Comments
• Uploading Files (pdf, png)

9) Viewing Request History:
• Status
• Report Date
• Assigned Department
• Problem Image
• Comment

10) Customized Problem Search:
• Problem Number
• Mobile
• Email
• Search in Comments
• Status
• Problem Category

11) Fuel Stock Monitoring and Order Creation:
• Supervision of available fuel
• Placing an order
• Filter selection (School unit, department type, available fuel, fuel type)
• Fuel liters input

12) User Overview, School Units Overview, Open Requests Overview, Display of School Units on a Map.

Non functional requirements

In this section we analyze non-functional requirements that are quality considerations essential for usability and acceptability.

1. Respect and be compatible with senceCity platform User Interface.
Design and development of a management and control application for School Units

2. **Scalability:** Our platform should handle overload of users and data. We have already tested the platform with 100 users (directors) as many as the school units of a large municipality like the Municipality of Patras.

3. **Compatibility:** the platform should work on a variety of browsers and should be responsive in different screen sizes. We have already tested Google Chrome, Mozilla Firefox, and Safari.

4. **Usability:** the platform should be user-friendly with a responsive and intuitive user interface using material components (tables, paginators, loading indicators). Should support Accessibility features.

5. **Maintainability:** The codebase should be well-structured and maintainable. In the 'alerts' component, it should be thoroughly covered by both unit tests and end-to-end (e2e) tests.

6. **Caching:** Implement caching strategies to improve performance, reduce server load, and minimize database queries. We have already implemented caching mechanisms.

7. **Documentation:** Maintain comprehensive documentation. We have already implemented comprehensive documentation to ensure that everyone can understand and work with the platform effectively.

8. **Internationalization and Localization:** should support multiple languages. We have already included Greek and English.

9. **Compliance:** Should comply with relevant regulations. Our platform complies with data protection regulations like GDPR.

10. **Interoperability:** to integrate with other systems and services. Our platform is ready to integrate with third-party services, APIs (Application Programming Interfaces), and data interchange formats such as mySchool platform [9].

2.6 **Design mockups and prototypes**

Designers primarily employ mock-ups to gather user feedback on designs and design concepts during the early stages of the design process. With the term Mock-ups, we refer to prototypes at an early stage. Although mockups are not completely functional, users can interact with
them under the guidance of the designer. As a result, valuable feedback is provided by the users which refers to functionality, usability, comprehension of the core design concept, and other aspects of the design.

Mock-ups and prototypes offer several benefits in the design process. They encourage user feedback, as their low-cost and low-fidelity nature, often using materials like cardboard, make users more willing to criticize early system versions. These mock-ups facilitate collaborative design discussions, enabling both users and designers to make changes using familiar tools, such as pens and scissors. Moreover, they promote teamwork by bridging interdisciplinary gaps within the design team. Mock-ups also allow early usability testing, helping identify issues in the design before they become costly to address during development. Their affordability encourages experimentation and, importantly, they shift the focus to content and functionality, diverting attention from intricate graphic design details [10].

Prototyping is an essential experimental phase in the design process, where designing teams transform conceptual ideas into tangible forms, ranging from paper sketches to digital representations. These prototypes come in various degrees of fidelity and serve to encapsulate design concepts and gather user feedback. The use of prototypes enables iterative refinement and validation of designs, ensuring that a brand can bring the most suitable products to market [11]. “They slow us down to speed us up. By taking the time to prototype our ideas, we avoid costly mistakes such as becoming too complex too early and sticking with a weak idea for too long.” As said by — Tim Brown, CEO & President of IDEO.

To achieve a fully functional website, this study employed mockups and prototypes of the school plugin for evaluation [12],[13]. The design process commenced with a basic concept, creating mockups with varied color schemes and limited information. Subsequently, it evolved into prototypes featuring comprehensive interaction between commands and pages. Experts then subjected the prototype to various scenarios to identify heuristic violations, propose solutions to rectify errors, and enhance the website's usability and functionality.

Finally, it is worth noting that we strictly adhered to Travis D's Web Usability Guidelines [42], which are presented in the image below.
2.7 Interview with an expert

We organized meetings with various municipalities to introduce our idea. By using the prototypes, we actively involved users from the initial design stages. These meetings brought school unit principals together, school committee presidents, and Municipality employees who possessed experience with similar platforms like Sense City. In this section, we outline the questions posed during these sessions and the invaluable insights we gained from these collaborative meetings.

1) What are the categories and subcategories of problems-requests?

These are various problem categories and subcategories that a school unit might encounter:

1. **Plumbing**
   - A) Internal leakage
   - B) External leakage
   - C) Replacement of internal faucets
   - D) Replacement of external faucets

2. **Electrical**
   - A) Burnt-out bulbs
   - B) Issues with outlets
   - C) Network problems

3. **Painting**
   - A) Interior painting
   - B) Exterior painting

4. **Roof Issues**
   - A) Water leakage
   - B) Broken tiles
   - C) Gutter cleaning

Figure 3: Usability Heuristics [41]
5. **Courtyard** A) Lawn cutting B) Courtyard cleaning C) Insufficient lighting

6. **Heating** A) Boiler cleaning B) Radiator replacement C) Heating oil supply (if necessary)

7. **Equipment and Technical Infrastructure (under consideration)**
   1. Computers (PCs)
   2. Laptops
   3. Printers
   4. Photocopiers
   5. Landline phones
   6. Cordless phones
   7. Desks
   8. Televisions
   9. Toys
   10. Whiteboards
   11. Microphones
   12. Gym equipment

8. **General**
   1. Cleaning supplies shortage
   2. Shortage of art supplies
   3. Computer/printer shortages

Note: Each subcategory includes the option "other" where the school director can enter free-text information if the issue doesn't fit any of the existing categories.

2) **What regulations (protocols) exist for the various processes?**

For example, boiler maintenance, fire extinguishers expiration dates.

According to the existing regulations, the above-mentioned processes should be carried out once a year. However, there is no system for recording or notifications in place to assist in organizing these procedures.
3) **Is there any interest in recording additional information beyond what we currently have (e.g., Event Halls, Desks, Indoor Gyms)?**

There is potential interest in recording information about Event Halls for the purpose of utilizing them during non-operational hours of school facilities. Accessibility information is also important. Furthermore, relevant information is already recorded in mySchool platform, which can be extracted by creating an interface/integration between the two platforms.

4) **If a school has a damage or issue (request), which procedure is followed (municipality/school committee)?**

Initially, it should be noted that for anything that requires further municipality study, the responsibility lies with the Municipality. The director communicates their request to the School Committee president via phone or email. If the School Committee has the necessary resources, they will handle it; otherwise, they delegate it to the Municipality. Additionally, for minor issues such as a broken faucet or a damaged window, the Director can use funds from the school's treasury to expedite the resolution process. Through the discussion, it was clear that there is a deep need for organizing this chaotic process, where there is no historical record, and only emails and paperwork prevail.

From the above, the need for creating the "heating," "alerts," and "dashboard" components was born, which contributes to better organization and resource conservation.

### 2.8 Issue lifecycle

In this section, we will describe the lifecycle of a request after discussion with the stakeholders. As shown in Figure 3,4, a request from a school unit enters the system with the status 'confirmed'. The school manager assigned it to the relevant department of the municipality for resolution, and the request now has the status 'in progress'. After the relevant department resolves the specific request, it changes the status to 'resolved' by selecting one of the following states: restored, invalid report, not restored/rejected, already reported.
Design and development of a management and control application for School Units

Figure 4: Issue lifecycle
3

Implementation of Schools plugin

3.1 The software architecture

Software architecture plays a central role in system design, providing a structured framework for program development. For our system, we chose a clear layered based architecture to avoid disorganizing the code. Otherwise, it could lead to confusion about roles, responsibilities, and relationships. This approach organizes code into horizontal layers, each with a well-defined role in the operation of the application. These layers communicate sequentially, following the concept of abstract actions to manage requests. Typically, a tiered architecture includes three layers: the presentation layer, the application layer, and the database layer.

3.1.1 The presentation layer.
The presentation layer has a crucial role in system architecture. It is responsible for handling the presentation of data to the user. It is useful because it eases the communication between the user and the web browser. It contains logic related to how the data is presented and serves as the beginning of each CRUD operation flow. In JavaScript, it is where all the controllers and views are placed. The primary goal of this layer is to ensure that data is presented to the user in a clear, comprehensive, and user-friendly manner.

3.1.2 The application layer.
The application layer of the software architecture is responsible for housing the core logic of the application. This layer takes on the crucial role of executing instructions related to requests originating from either the presentation layer or the database layer. Upon receiving a request from the presentation layer, it orchestrates the necessary actions needed to fulfill that request. In essence, it liaises with the database communication layer to acquire the appropriate
models required to perform the necessary operations. This layer functions as the central hub of the application, as it encapsulates the logic and algorithms essential for processing data received from other layers. Its main function is to oversee the system, make logical decisions, take appropriate actions, and ensure a seamless transfer of results between all the layers.

### 3.1.3 The database layer.

The database layer plays a vital role in the general functionality and success of an application. Its initiative function is to permanently store and retrieve data. Should this layer be left out, that could lead to unstable data, and lost information when the application is closed or restarted. Moreover, it enables data storage that is persistent and accessible across multiple sessions and users. It implements security measures like user authentication and access control to secure data. In addition to its core functions, this layer can offer features such as data caching, indexing, and querying, significantly enhancing the application’s performance and efficiency in handling substantial data volumes. In summary, the database layer is a critical component of modern applications, providing reliable and secure data storage and access. Developing effective applications requires a sound understanding of database technologies and their implementation. In this thesis, the database was developed using the MongoDB database management system to store and manage the data.

### 3.2 Analysis of database

#### 3.2.1 Mongoose schemas - smart data models

We created our mongoose schemas based on smart data models [14] to enable portability for different applications including Smart Cities, Smart environment, Smart Sensoring and Smart Energy. Specifically, we used Building [15], Building Type [16] and issue tracking models [17].

Mongoose is a tool that makes the process of working with MongoDB in a Node.js application easy because it offers a more structured approach to data modeling and validation.

Next we define our Schemas:

- **User Schema:**

```
position: { type: String, required: true },
name: { type: String, default: "" },
surname: { type: String, default: "" },
email: { type: String, unique: true, required: true },
username: { type: String, unique: true, required: true },
password: { type: String, required: true },
role_id: { type: Array, required: true },
city: { type: String, required: true },
last_login: { type: Number },
departments: { type: Array, default: [ 0 ] },
uuid: { type: String },
active: { type: Boolean, default: true, required: true },
phone: { type: String, default: "" }
}

- Building Schema:

{  
  address: {
    addressCountry: { type: String, default: "" },
    addressLocality: { type: String, default: "" },
    addressRegion: { type: String, default: "" },
    postOfficeBoxNumber: { type: String, default: "" },
    postalCode: { type: String, default: "" },
    streetAddress: { type: String, default: "" }
  },
  city: { type: String, default: "" },
  phone: { type: String, default: "" },
  category: { type: String, default: "" },
  id: { type: String, default: "" }
}
Design and development of a management and control application for School Units

type: [
  {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'buildingType'
  }
],
loc: { type: { type: String }, coordinates: [Number] },
name: { type: String, default: '' },
dateCreated: { type: Date, default: Date.now },
dateModified: { type: Date, default: Date.now },
occupier: [
  {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'user'
  }
],
openingHours: [
  {
    addressCountry: { type: String, default: '' },
    addressLocality: { type: String, default: '' },
    addressRegion: { type: String, default: '' },
    postOfficeBoxNumber: { type: String, default: '' },
    postalCode: { type: String, default: '' },
    streetAddress: { type: String, default: '' }
  }
],
floorsAboveGround: { type: String, default: '' },
floorsBelowGround: { type: String, default: '' }
Design and development of a management and control application for School Units


▪ **Building Type Schema:**

```json
{
  id: { type: String, default: "" },""urn:ngsi-ld:BuildingType:57b912ab-eb47-4cd5-bc9d-73abece1f1b3"
  type: { type: String, default: "BuildingType" },
  name: { type: String, default: "" , required:true, unique: true }""ProtoVathmia|DeuteroVathmia|DimotikiStathmoi..."
}
```

▪ **School Category Schema:**

```json
{
  city: {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'municipality'
  },
  category: [ 
    {
      name:{ type: String},
      sub_categories: [
        {
          name: { type: String }
        }
      ]
    }
  ]
}
```

▪ **Track Issue Schema:**

```json
{}
```
id: { type : String, default: "" },
reportId: { type: Number, default: 0 },
type: { type : String, default: "IssueReporting" },
owner: [{ type : String, default: "" }],
city: { type : String, default: "" },
    _owner: [],
    
    type: mongoose.Schema.Types.ObjectId,
    ref: 'user'
}
],
buildingId:{
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Building'
},
category: { type : String, default: "" },
subCategory: { type : String, default: "" },
resolutionStatus: { type : String, default: "" }, /*Assigned, InProgress, Closed*/
resolutionDetail:{ type : String, default: "" }, /*FIXED, INVALID, WONTFIX, DUPLICATE*/
duplicatedID : [
    
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Department'
]
],
bugzillaID : { type: String, default: "" },
bugzillaName:{ type: String, default: "" },

Design and development of a management and control application for School Units

dateCreated: { type: Date, default: Date.now },
dateModified: { type: Date, default: Date.now },
departmentId: [
  {
    type: mongoose.Schema.Types.ObjectId,
    ref: 'Department'
  }
],
address : {
  addressCountry : { type : String, default: "" },
  addressLocality : { type : String, default: "" },
  addressRegion : { type : String, default: "" },
  postOfficeBoxNumber : { type : String, default: "" },
  postalCode : { type : String, default: "" },
  streetAddress : { type : String, default: "" }
},
comments: [
  {
    comment:{ type : String, default: "" },
    dateCreated: { type: Date, default: Date.now },
    departmentId: [
      {
        type: mongoose.Schema.Types.ObjectId,
        ref: 'Department'
      }
    ],
    bugzillaID : { type: String, default: "" },
    bugzillaName: { type: String, default: "" }
  },
file:
{
    filename: { type: String, default: "" },
    file_extension: { type: String, default: "" }
}
]
,
adduser:
{
    email: { type: String, default: "" },
    mobile: { type: String, default: "" },
    name: { type: String, default: "" },
    role_id: [ Number ]
}
]
,
status_issue: { type: String, default: "" } // "Assigned, InProgress, Closed"
status_closed: { type: String, default: "" } // "Resolve, Duplicate...."
]
,
name: { type: String, default: "" } // "SchoolIssue | IssueReporting",
loc: { type: { type: String }, coordinates: [ Number ] },
priority: { type: String, default: "normal" }
}

- **School Alert Schema:**


name: { type: String, default: "" },
start: { type: Date },
end: { type: Date },
comment: { type: String },
recurring: { type: Boolean },
period: { type: String }, //year, month, day
number: { type: Number },
status: { type : String, default: "" }, // "Open, Closed, canceled" expired?
reportId: { type: Number }

### 3.3 Angular version Upgrade

Sense City platform was implemented with Angular version 5. The first step we took before starting code development was to upgrade the version from 5 to 7 following the upgrade instructions found in the Angular documentation [18]. Our main goal was to use the Angular Material [19] library version 7 which provides us with plenty of components. We removed several libraries and replaced them with Angular Material components to enhance maintenance and responsiveness in design. At the same time, if we consider the current version of Angular is 16, we will understand that it was an important step for the gradual upgrade of the platform to the latest versions so that we can use the new features they provide.

### 3.4 Technologies Used

*What is mean stack?*

The MEAN stack, an acronym for MongoDB, Express.js, Angular, and Node.js, is a highly popular and full-stack technology stack for developing web and mobile applications. This JavaScript-based open-source framework is widely used by programmers to create applications rapidly. The stack comprises the following key Technologies: [20]
2. Express.js: A web framework for Node.js that simplifies building server-side applications.
3. Angular: A client-side JavaScript framework for creating interactive user interfaces.

**MEAN stack architecture**

The MEAN architecture is designed to make building web applications in JavaScript and handling JSON incredibly easy.

![MEAN stack architecture](image)

*Figure 5: MEAN stack architecture [44]*

**Angular.js front end**

Angular.js front end, situated at the top of the MEAN stack, is a self-styled "JavaScript MVW (Model View and Whatever) [21]. Angular.js empowers you to augment your HTML tags with metadata, enabling the creation of dynamic and interactive web experiences with more potency than crafting them manually using static HTML and JavaScript (or jQuery) [21]. This front-end technology boasts a rich set of features, including form validation, localization, and seamless communication with your back-end service.
Angular is a development platform based on TypeScript. As a comprehensive platform, Angular encompasses: [21]

- A component-based framework for constructing scalable web applications.
- An array of well-integrated libraries that encompasses a wide spectrum of functionalities, such as routing, forms management and client-server communication,
- A suite of developer tools that facilitate the development, building, testing, and updating of your code.

**Express.js and Node.js server tier**

At the server tier, we have Express, which is a fast and minimalist web framework for Node.js applications. Express, the most popular Node web framework, serves as the foundational library for various other Node web frameworks [22],[23]. It offers the following capabilities:

- Handling requests with different HTTP (Hypertext Transfer Protocol Secure) verbs at distinct URL (Uniform Resource Locator) paths through request handlers.
- Integration with "view" rendering engines to generate responses by populating templates with data.
- Configuration of common web application settings, such as defining the connection port and specifying the template location for rendering responses.
- Incorporation of additional request processing "middleware" at any point in the request handling pipeline.

Express seamlessly connects to the Angular.js front end, allowing you to create dynamic and interactive web experiences by handling XML HTTP requests (XHRs), GETs, or POSTs from your front end [20]. These requests are routed to Express functions that drive your application. These functions, in turn, employ MongoDB's Node.js drivers, using either callbacks or promises, to access and modify data within your MongoDB database. This combination of Express and Angular.js forms a potent stack for developing web and mobile applications.

**MongoDB database tier**

At the database tier, MongoDB is the solution that seamlessly integrates with your application, making data storage as convenient as working with Angular, Express, and Node.
MongoDB is a document database offering the scalability and flexibility you require, along with robust querying and indexing capabilities. This database makes development straightforward, thanks to its simple document model that's easy for developers to grasp and utilize. It's designed to meet even the most intricate requirements, regardless of the scale of your project.

With MongoDB, the JSON documents generated in your Angular.js front end can be efficiently transmitted to the Express.js server [24]. There, they undergo processing and, provided they are valid, are stored directly in MongoDB for subsequent retrieval. MongoDB also provides a wide range of drivers for more than 10 programming languages, and a thriving community has developed numerous additional tools and resources, further enhancing its ease of use and versatility for developers.

*Why we use mean in schools platform?*

In senseCity project we focus on scalability so MEAN stack is the perfect choice. Also MEAN stack helps to improve productivity because it uses JavaScript for both front-end and back-end development. We use MongoDB version 5, Express version 4, Angular version 7 and Node.js version 16.20.0 (LTS).

### 3.4.1 Libraries and packages

For the installation of the above packages, we used Angular CLI in combination with npm.

**Backend:**

*Table 1: Backend libraries and packages*

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;express&quot;</td>
<td>&quot;^4.16.4&quot;</td>
<td><a href="https://expressjs.com/">https://expressjs.com/</a></td>
</tr>
<tr>
<td>&quot;mongoose&quot;</td>
<td>&quot;^5.3.11&quot;</td>
<td><a href="https://mongoosejs.com/">https://mongoosejs.com/</a></td>
</tr>
<tr>
<td>&quot;nodemailer&quot;</td>
<td>&quot;^4.6.7&quot;</td>
<td><a href="https://www.npmjs.com/package/nodemailer">https://www.npmjs.com/package/nodemailer</a></td>
</tr>
<tr>
<td>&quot;crypto&quot;</td>
<td>&quot;^4.6.7&quot;</td>
<td><a href="https://nodejs.org/api/crypto.html">https://nodejs.org/api/crypto.html</a></td>
</tr>
<tr>
<td>&quot;moment&quot;</td>
<td>&quot;^2.29.3&quot;</td>
<td><a href="https://momentjs.com/">https://momentjs.com/</a></td>
</tr>
<tr>
<td>&quot;jsonfile&quot;</td>
<td>&quot;^4.0.0&quot;</td>
<td><a href="https://www.npmjs.com/package/jsonfile">https://www.npmjs.com/package/jsonfile</a></td>
</tr>
<tr>
<td>&quot;formidable&quot;</td>
<td>&quot;^1.2.1&quot;</td>
<td><a href="https://www.npmjs.com/package/formidable">https://www.npmjs.com/package/formidable</a></td>
</tr>
</tbody>
</table>
Design and development of a management and control application for School Units

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>&quot;sharp&quot;</td>
<td>&quot;^0.27.2&quot;</td>
<td><a href="https://www.npmjs.com/package/sharp">https://www.npmjs.com/package/sharp</a></td>
</tr>
<tr>
<td>&quot;fs&quot;</td>
<td></td>
<td><a href="https://nodejs.org/dist/latest-v6.x/docs/api/fs.html">https://nodejs.org/dist/latest-v6.x/docs/api/fs.html</a></td>
</tr>
<tr>
<td>&quot;gm&quot;</td>
<td>&quot;^1.23.1&quot;</td>
<td><a href="https://www.npmjs.com/package/gm">https://www.npmjs.com/package/gm</a></td>
</tr>
<tr>
<td>&quot;minio&quot;</td>
<td>&quot;^5.0.2&quot;</td>
<td><a href="https://www.npmjs.com/package/minio">https://www.npmjs.com/package/minio</a></td>
</tr>
</tbody>
</table>

**Frontend:**

Table 2: Frontend libraries and packages

<table>
<thead>
<tr>
<th>Package name</th>
<th>Version</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>@angular/forms</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/guide/forms-overview">https://angular.io/guide/forms-overview</a></td>
</tr>
<tr>
<td>@angular/http</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/guide/http-request-data-from-server">https://angular.io/guide/http-request-data-from-server</a></td>
</tr>
<tr>
<td>@angular/material</td>
<td>&quot;^7.0.0&quot;</td>
<td><a href="https://material.angular.io/">https://material.angular.io/</a></td>
</tr>
<tr>
<td>@angular/platform-browser</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/platform-browser">https://angular.io/api/platform-browser</a></td>
</tr>
<tr>
<td>@angular/platform-browser-dynamic</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/platform-browser-dynamic">https://angular.io/api/platform-browser-dynamic</a></td>
</tr>
<tr>
<td>@angular/router</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/router">https://angular.io/api/router</a></td>
</tr>
<tr>
<td>@asymmetrik/ngx-leaflet</td>
<td>&quot;^3.0.1&quot;</td>
<td><a href="https://www.npmjs.com/package/@asymmetrik/ngx-leaflet">https://www.npmjs.com/package/@asymmetrik/ngx-leaflet</a></td>
</tr>
<tr>
<td>@angular/animations</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/guide/animations">https://angular.io/guide/animations</a></td>
</tr>
<tr>
<td>@angular/cdk</td>
<td>&quot;^7.0.0&quot;</td>
<td><a href="https://material.angular.io/cdk">https://material.angular.io/cdk</a></td>
</tr>
<tr>
<td>@angular/common</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/common">https://angular.io/api/common</a></td>
</tr>
<tr>
<td>@angular/compiler</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/core/Compiler">https://angular.io/api/core/Compiler</a></td>
</tr>
<tr>
<td>@angular/core</td>
<td>&quot;^7.2.16&quot;</td>
<td><a href="https://angular.io/api/core">https://angular.io/api/core</a></td>
</tr>
<tr>
<td>@asymmetrik/ngx-leaflet-markercluster</td>
<td>&quot;^1.0.0&quot;</td>
<td><a href="https://www.npmjs.com/package/@asymmetrik/ngx-leaflet-markercluster">https://www.npmjs.com/package/@asymmetrik/ngx-leaflet-markercluster</a></td>
</tr>
<tr>
<td>@ngx-translate/core</td>
<td>&quot;^10.0.2&quot;</td>
<td><a href="https://www.npmjs.com/package/@ngx-translate/core">https://www.npmjs.com/package/@ngx-translate/core</a></td>
</tr>
<tr>
<td>@ngx-translate/http-loader</td>
<td>&quot;^2.0.1&quot;</td>
<td><a href="https://www.npmjs.com/package/@ngx-translate/http-loader">https://www.npmjs.com/package/@ngx-translate/http-loader</a></td>
</tr>
<tr>
<td>Package</td>
<td>Version</td>
<td>URL</td>
</tr>
<tr>
<td>----------------------------------</td>
<td>----------</td>
<td>------------------------------------------</td>
</tr>
<tr>
<td><code>@swimlane/ngx-charts</code></td>
<td>&quot;^12.1.0&quot;</td>
<td><a href="https://www.npmjs.com/package/@swimlane/ngx-charts">https://www.npmjs.com/package/@swimlane/ngx-charts</a></td>
</tr>
<tr>
<td><code>@types/geojson</code></td>
<td>&quot;^7946.0.7&quot;</td>
<td><a href="https://www.npmjs.com/package/@types/geojson">https://www.npmjs.com/package/@types/geojson</a></td>
</tr>
<tr>
<td><code>@types/googlemaps</code></td>
<td>&quot;^3.30.9&quot;</td>
<td><a href="https://www.npmjs.com/package/@types/googlemaps">https://www.npmjs.com/package/@types/googlemaps</a></td>
</tr>
<tr>
<td><code>@types/leaflet.awesome-markers</code></td>
<td>&quot;^2.0.23&quot;</td>
<td><a href="https://www.npmjs.com/package/@types/leaflet.awesome-markers">https://www.npmjs.com/package/@types/leaflet.awesome-markers</a></td>
</tr>
<tr>
<td><code>@types/leaflet.markercluster</code></td>
<td>&quot;^1.0.3&quot;</td>
<td><a href="https://www.npmjs.com/package/@types/leaflet.markercluster">https://www.npmjs.com/package/@types/leaflet.markercluster</a></td>
</tr>
<tr>
<td><code>angular-bootstrap-md</code></td>
<td>&quot;^7.5.4&quot;</td>
<td><a href="https://www.npmjs.com/package/angular-bootstrap-md">https://www.npmjs.com/package/angular-bootstrap-md</a></td>
</tr>
<tr>
<td><code>angular-calendar</code></td>
<td>&quot;^0.28.28&quot;</td>
<td><a href="https://www.npmjs.com/package/angular-calendar">https://www.npmjs.com/package/angular-calendar</a></td>
</tr>
<tr>
<td><code>bootstrap</code></td>
<td>&quot;^3.4.1&quot;</td>
<td><a href="https://www.npmjs.com/package/bootstrap">https://www.npmjs.com/package/bootstrap</a></td>
</tr>
<tr>
<td><code>chart.js</code></td>
<td>&quot;^2.5.0&quot;</td>
<td><a href="https://www.npmjs.com/package/chart.js?activeTab=readme">https://www.npmjs.com/package/chart.js?activeTab=readme</a></td>
</tr>
<tr>
<td><code>core-js</code></td>
<td>&quot;^2.4.1&quot;</td>
<td><a href="https://www.npmjs.com/package/core-js">https://www.npmjs.com/package/core-js</a></td>
</tr>
<tr>
<td><code>date-fns</code></td>
<td>&quot;^2.30.0&quot;</td>
<td><a href="https://www.npmjs.com/package/date-fns">https://www.npmjs.com/package/date-fns</a></td>
</tr>
<tr>
<td><code>font-awesome</code></td>
<td>&quot;^4.7.0&quot;</td>
<td><a href="https://www.npmjs.com/package/font-awesome">https://www.npmjs.com/package/font-awesome</a></td>
</tr>
<tr>
<td><code>hammerjs</code></td>
<td>&quot;^2.0.8&quot;</td>
<td><a href="https://www.npmjs.com/package/hammerjs">https://www.npmjs.com/package/hammerjs</a></td>
</tr>
<tr>
<td><code>html2canvas</code></td>
<td>&quot;^1.0.0-alpha.12&quot;</td>
<td><a href="https://www.npmjs.com/package/html2canvas">https://www.npmjs.com/package/html2canvas</a></td>
</tr>
<tr>
<td><code>leaflet</code></td>
<td>&quot;^1.2.0&quot;</td>
<td><a href="https://www.npmjs.com/package/leaflet">https://www.npmjs.com/package/leaflet</a></td>
</tr>
<tr>
<td><code>leaflet.awesome-markers</code></td>
<td>&quot;^2.0.4&quot;</td>
<td><a href="https://www.npmjs.com/package/leaflet.awesome-markers">https://www.npmjs.com/package/leaflet.awesome-markers</a></td>
</tr>
<tr>
<td><code>leaflet.gridlayer.googlemutant</code></td>
<td>&quot;^0.6.4&quot;</td>
<td><a href="https://www.npmjs.com/package/leaflet.gridlayer.googlemutant">https://www.npmjs.com/package/leaflet.gridlayer.googlemutant</a></td>
</tr>
<tr>
<td><code>leaflet.markercluster</code></td>
<td>&quot;^1.3.0&quot;</td>
<td><a href="https://www.npmjs.com/package/leaflet.markercluster">https://www.npmjs.com/package/leaflet.markercluster</a></td>
</tr>
<tr>
<td><code>material-design-icons</code></td>
<td>&quot;^3.0.1&quot;</td>
<td><a href="https://www.npmjs.com/package/material-design-icons">https://www.npmjs.com/package/material-design-icons</a></td>
</tr>
<tr>
<td><code>moment</code></td>
<td>&quot;^2.21.0&quot;</td>
<td><a href="https://www.npmjs.com/package/moment">https://www.npmjs.com/package/moment</a></td>
</tr>
<tr>
<td><code>ngx-filesaver</code></td>
<td>&quot;^11.0.0&quot;</td>
<td><a href="https://www.npmjs.com/package/ngx-filesaver">https://www.npmjs.com/package/ngx-filesaver</a></td>
</tr>
<tr>
<td><code>ngx-image-compress</code></td>
<td>&quot;^8.0.4&quot;</td>
<td><a href="https://www.npmjs.com/package/ngx-image-compress">https://www.npmjs.com/package/ngx-image-compress</a></td>
</tr>
</tbody>
</table>
3.4.2 Gitlab – SourceTree

GitLab and SourceTree are two popular tools used in software development, particularly for version control and project management. Gitlab is the most popular open-source software for managing your devops workflow. It allows team collaboration in every phase of the project. Some key features and aspects of GitLab are Version Control, Repository Hosting, Collaboration and Continuous Integration/Continuous Delivery (CI/CD).

SourceTree is a graphical user interface (GUI) for Git that simplifies the interaction with your Git repositories in comparison with the command line interface you are allowed to concentrate on coding in a more user-friendly way. Some of the key features of GitLab are Repository Management, Visual Commit, Branching and Merging, Pull and Push and Conflict Resolution.

SourceTree can connect to GitLab repositories targeting at managing your code using a GUI while at the same time utilizing GitLab's capabilities for code hosting, issue tracking, and CI/CD pipelines.
3.5 Hosting in NAM facilities

The system architecture that hosts our platform is illustrated in the image below, in NAM facilities. Through the user interface, the client communicates with an Nginx web server located on VM (Virtual Machine) 1. Additionally, we use Nginx as a proxy server for components of the architecture that do not have a public Internet Protocol (IP) address, as seen in VM2. These components are served to the client through Nginx.

![Hosting architecture diagram](image)

*Figure 6: Hosting architecture*

3.6 Application pages & Functionalities

In the aforementioned chapters, we presented a thorough requirements analysis and platform design. In this chapter, our focus shifts to platform development based on the previous chapters. Below, we present application pages and functions for the roles supported by the platform.

3.6.1 Functionalities and Permitted Actions of Directors

In this section, we clarify the competencies and permissible actions bestowed upon an individual holding the role of Director. We present in detail the necessary steps / actions to be taken as well as the corresponding user interface.

1. Register/Login

School director receives an email to complete his registration as illustrated in the image below. He must follow the received link and complete the registration procedure. After that, he can login in the system as illustrated in image 7.
Design and development of a management and control application for School Units

Figure 7: Registration email

Figure 8: Registration form

Figure 9: Login form

2. Request creation
In this component, the user holding the School Director role has the capability to create a request following the steps below.

- Select Problem Category (Required)
- Select Problem Subcategory (Required)
  - If 'OTHER' is selected, then provide a BRIEF DESCRIPTION
- Create a comment (optional)
- Image selection (optional)
- Press the Submit button to send the request. You immediately see the Loading spinner and the toaster message. (Required)

![Figure 10: Issue creation page](image)

### 3. Supervision of Issues

In this component, the user holding the School Director role has the capability to access and review all the requests he has submitted. Furthermore, he is granted the ability to conduct searches utilizing pertinent filters following the steps below.

- Select Sorting (Newer first, Older first)
- Select Status (All, Confirmed, In progress, Resolved)
- Select Problem Category-Subcategory (All)
- Select Created from-to (Date from, Date to)
• Press the Refresh button. You immediately see the Loading spinner and the refreshed content.

**Figure 11: Issues page**

4) **Monitoring the progress of each issue/request and providing comments.**

In this component, the user holding the School Director role has the capability to Monitor the progress of each problem/request. By selecting an issue from the list, the director gains complete access to the issue's history. They are provided with information pertaining to their request, the status of their request, and the department responsible. Finally, they have the capability to comment on the request, both in text and with an image.
Design and development of a management and control application for School Units

5) Account Settings

In this component, the user holding the School Director role has the capability to Complete the general information of the School Unit, such as region, municipality, address, postal code, phone number, floors, and basement. For the school unit's address, the Director must locate the school unit on the map and, by clicking on it, the address is automatically filled in the general information.
6) Notifications

In this component, by assuming the role of School Director, the user can generate a notification. They can do so by providing the name, setting the expiration date, adding a comment, and specifying whether it is a recurring notification, as illustrated in the image below.

Figure 14: Account settings page

Figure 15: Notifications creation page
Also, they can use Filters like status and date range for specialized notification search. The status can take the following values: Open, Closed, Cancelled, Expired, as illustrated in the image below.

![Notifications page](image)

**Figure 16: Notifications page**

Additionally, they are provided with a calendar for viewing monthly notifications with an open status as illustrated in the image below. If they select a notification in the calendar, detailed information related to that notification is displayed below.

![Issues calendar](image)

**Figure 17: Issues calendar**
7) Supervision of Data from Sensors

In this component, the user holding the School Director role has the capability to oversee:

a) Consumption (electricity, water, oil)

b) Environmental Data (air quality, temperature, humidity, CO2)

as illustrated in the image below.

Figure 18: Issue view in calendar

Figure 19: Sensors monitoring page
3.6.2 Functionalities and Permitted Actions of School Manager

Below, we clarify the responsibilities and permitted actions granted to an individual holding the School Manager role. We present in detail the necessary steps/actions to be taken as well as the corresponding user interface.

1. Login

![Login page]

*Figure 20: Login page*

2. Overview

In this component, the user holding the School Manager role has the capability to view the total number of users, school units, and the count of open issues. Additionally, he can navigate to user settings, school unit settings, and issues as illustrated in the image below.
3. School Units

In this component, the user holding the School Manager role has the capability to view all School Units that are registered in the platform, to search for a specific one, to Create and edit School units following the steps below.

3.1 View all School Units

3.2 Create

- Select School Unit Name (Required)
- Select Department type (Required)
Design and development of a management and control application for School Units

- Complete School Unit Email (Required)
- Press the Submit button to create the school unit. You immediately see the Loading spinner and the toaster message.

**Figure 23: School Users creation**

3.3 **Edit School**

In this component, the user holding the School Manager role has the capability to edit School units’ information (name, email, Address, Location, Postal Code) as illustrated in the image below.

**Figure 24: Edit school units’ information.**

4. **Users**
In this component, the user holding the School Manager role has the capability to view all School Users that are registered in the platform, to search for a specific one and edit School users following the steps below.

### 4.1 View all School Users

![Figure 25: School Users](image)

### 4.2 Edit School

In this component, the user holding the School Manager role has the capability to edit School users’ information (account name, email, position, User Role, name, surname, password) as illustrated in the image below.

![Figure 26: Edit school user information](image)
5. Issues

In this component, the user holding the School Manager role has the capability to access and review all the requests that School Directors have submitted. Furthermore, he is granted the ability to conduct searches utilizing pertinent filters following the steps below.

- Select Sorting (Newer first, Older first)
- Select Problem Category-Subcategory (All)
- Select Created from-to (Date from, Date to)
- Press the Refresh button. You immediately see the Loading spinner and the refreshed content.

![School Issues page](image)

*Figure 27: School Issues page*

By selecting an issue from the list, the School Manager has the capability to view and edit issues’ information as illustrated in the image below. He can see the complete history of that request as well as its location. Here is the following procedure: the school Manager changes the status of the request from "confirmed" to "in progress." Then, they assign it to the appropriate department, change the priority, if necessary, add a comment, and upload a document.
Design and development of a management and control application for School Units

Figure 28: Edit School Issue page

6. Search

In this component, the user holding the School Manager role has the capability to search for a specific issue using parameters like issue number, mobile, email, comment search, state, issue category as illustrated in the image below.

Figure 29: Search Issue

7. Heat

In this component, the user holding the School Manager role has the capability to create a fuel order based on school’s available fuel as illustrated in the image below.
3.6.3 Functionalities and Permitted Actions of Department Users

Below, we clarify the responsibilities and permitted actions granted to an individual holding the Department User role. We present in detail the necessary steps / actions to be taken as well as the corresponding user interface.

1. **Login**

2. **View and administer issues of a selected category.**

In this section, all requests from School Directors are presented in a list format along with their location on the map. The Department user can use filters such as sorting, issue types/subtypes and a specific range of dates. By clicking on a request from the list, you can access information about the request, its status, assignment department, and more. Clicking the 'Manage' button allows you to change the request's status, assign it to the appropriate department for resolution, and add comments (text, files) as illustrated in the image below. (ο xριστής εγγράφεται από τον αδιμί του senseCity)
3. Search

In this component, the user holding the Department user role has the capability to search for a specific issue using parameters like issue number, mobile, email, comment search, state, issue category as illustrated in the image below.
Design and development of a management and control application for School Units

*Figure 33: Search School Issue*
Testing, Evaluation & Redesign

4.1 Angular testing

Angular testing is a fundamental feature to check if your application is working as intended [32]. It is available in every Angular project set up with the Angular CLI. In our project we use Jasmine framework to write the test scenarios and Karma test runner to run/execute them in browsers from the command line. The command-line not only executes tests and displays results but also auto-reruns tests on file changes, enabling real-time feedback and streamlined testing in the development workflow.

4.1.1 The testing pyramid in Angular

According to Google a split of 70% unit tests, 20% integration tests, and 10% end-to-end tests is suggested. Also avoiding anti-patterns such as Inverted pyramid/ice cream cone is crucial [45].
As illustrated in the above picture there are the following types of tests:
Isolated and shallow unit testing, integration tests between components and UI/E2E tests, which can be functional and visual regression testing.

1) a) Isolated Unit tests
Angular Unit Testing is crucial for ensuring the robustness and reliability of our codebase. By employing isolated tests, developers can focus on testing individual units, such as components or services, in isolation. This approach is beneficial for several reasons:

- Testing Business Logic in Isolation
- Mocking External Dependencies
- Fast and Reliable Testing
- Isolation of Failures
- Facilitating Refactoring
- Bug Detection at Early Stages
- Building Confidence in Isolated Environments
- Adding new features without breaking anything.
In summary, Angular Unit Testing is a critical practice that enhances code quality, accelerates development workflows, and provides the confidence needed to make changes and improvements to the codebase.

1) b) **Shallow unit testing**
Shallow testing is a procedure where you test a component with a template, but you don’t render child component by setting schema to **NO_ERRORS_SCHEMA** [33]. This will ignore unknown tags in the template, making us not needing to import the child components.

2) **Integration testing**
Integration testing involves testing the collaboration of two or more components, particularly in areas of the application where the interaction between components is crucial [34]. The focus is on testing the behavior of interconnected units. While unit tests excel at ensuring that individual units function well in isolation, they lack the capability to confirm whether these units effectively work together [35]. This limitation emphasizes the importance of integration testing. Integration tests operate on a smaller scale compared to end-to-end tests, typically examining the behavior of a small group of units, often two, to validate their coherent collaboration.

3) **End to end testing**
Unlike the above tests, end-to-end tests prioritize the examination of the overall functionality of the application. Despite being slower and more intricate, these tests play a crucial role in guaranteeing that the application behaves accurately in a real-world scenario. The use of browser automation allows for a comprehensive evaluation of the application's behavior, providing a more holistic perspective on its performance and interactions [36].

### 4.2 Unit Tests vs. Integration Tests vs. End-to-End Tests

The goal of Manual testing is to identify bugs before the software is released. Manual Testers validate and verify key components of the application along with edge cases and scenarios that can never happen in real-life usage [38]. Manual testing is a crucial software testing process where test cases are executed by a tester without the use of automated tools. The focus is on assessing the application from an end user's perspective to ensure it aligns with the specified requirements. This method aims to achieve nearly 100 percent test coverage. Defects, discrepancies between expected and actual outputs, are identified and reported. After
developers address these defects, the software undergoes retesting. Manual testing is a foundational step preceding automated testing for newly developed software. Despite being resource-intensive, it provides assurance of a bug-free application. Manual testing demands knowledge of testing techniques rather than automated tools and is considered essential, acknowledging the impossibility of achieving 100% automation.

Below, we present the life cycle of a request:

**User Creation with SchoolManager Role:**
- Log in with a user having the Super Admin role.
- Define the central department of school units. The user of the department must have the SchoolManager role, and if the department doesn't exist, it should be created.

**Login with User Role SchoolManager:**
- The first page is the overview page (some statistics, the school units will be displayed on the map).
- Supervision and creation of school units.

**User Supervision with Director Role and Ability to Resend Activation Email and Edit Details:**
- List of issues as a school unit administrator.
- Search.
- Heating and placing an order.

**Login with User Role Director:**
- Follow the link in the email sent in the previous step and enter the details of the school unit's responsible person (Director).

**Login as a User with Director Role:**
- Go to the "Account Settings" option and fill in the details.
- Create a request.
- Display issues and requests with search filters.
- Display the progress of a request and the ability to comment.
• Notifications.
• Dashboard with some metrics.

Flow Request:
• Director submits a request.
• School Manager processes it and forwards it to a department (Department of cleaning).
• The Director sees the status change from the history.
• The department administrator (Department of cleaning) processes it and closes it.

4.2.1 Implementation of tests
It is worth noting that the version of Node.js we used caused issues with our tests, specifically during the command Fixture.detect.changes() (it applies change detection to update the DOM), which was crucial for interacting with the DOM. Initially, I upgraded the versions of the Karma and Jasmine packages to the latest versions. After extensive research on the internet, I decided to downgrade the version of Node.js from 16.20 to 14.15.1. This decision was made after careful consideration and contemplation. The rationale was as follows: other possible solutions on the internet had been exhausted. Changing the Node.js version might potentially create problems either in the API or in some other package dependency.

There was no other option but to change the version and proceed based on what would unfold. It's worth mentioning that we have created a test environment for the API, so in the worst case, we would use that. The alternative and initial solution/choice was to search for another stable version of Node.js.

Finally, it's worth noting that the opinions/suggestions regarding version change were not clear-cut. The decision we made was correct, and in our case, Node.js version 16.20 was responsible for the malfunction of the Karma runner. From the above, the need for the immediate use of Docker engine arises.

As shown in the image 35, tests were implemented for services, components, and asynchronous testing.
Figure 35: Unit testing results

For E2E tests with Cypress, we installed Cypress version 9.1.1, and it was necessary to upgrade TypeScript to version 3.2.4. In image 36, we present the results from the following scenario: the user with the role of School Director logs into their account, navigates to the section for creating requests, creates a new request, and then proceeds to the issues section, monitoring the progress of their request.
Design and development of a management and control application for School Units

Figure 36: END-2-END testing with cypress
4.3 Heuristic Evaluation

![Heuristic Evaluation Diagram]

Figure 37: Jakob Nielsen's 10 general principles for interaction design [39].

What is Heuristic Evaluation?

Heuristic evaluation is a broadly acknowledged method of usability analysis. It is a fast and practical way to solve problems or make decisions. According to this method, three to five evaluators are presented with an interface design and asked to comment on it [25].

Evaluators use established heuristics (e.g., Nielsen-Molich’s) and reveal insights that can assist design teams in improving product usability right from the early stages of development. The main goal of heuristic evaluations is to identify any problems associated with the design of user interfaces.

Heuristic evaluation serves as a formative assessment tool to aid designers in enhancing interaction during the initial stages. However, it is not intended for use as a summative assessment to assert the quality of the final product [26].

Besides the basic heuristics of Nielsen, the new methods that have been added are navigation, privacy, functionality, compatibility and explicitness [27].

Woolrych and Cockton conducted a heuristic evaluation, comparing expert assessments with user testing [28]. They found discrepancies, suggesting that heuristic features may offer limited advantages and can even produce counterproductive results. They recommend
combining heuristic techniques with careful examination of user skills, emphasizing on the necessity of participant involvement for a realistic assessment of a system's success.

**Execution and Implementation of heuristic evaluation**

Three experienced evaluators in usability and user experience were chosen to assess the system, providing a comprehensive perspective. They utilized the well-established Nielsen and Rolf Molich guidelines, [29] to systematically identify potential usability issues. The evaluators thoroughly analyzed the system's interface and functionality, examining elements such as system feedback, error prevention, consistency, and user control.

The evaluation process involved a meticulous review of each heuristic, highlighting instances of potential violations and suggesting areas for improvement to enhance overall usability. The inclusion of a diverse group of evaluators brought varied perspectives and expertise, ensuring a comprehensive evaluation. Collaborative interactions fostered discussions, expanded issue identification, and provided a holistic understanding of the system's strengths and weaknesses.

### 4.3.1 Evaluation results

**Violation 1**
- Recognition rather than recall: label icons without description on hover.

![Figure 38: View and editing buttons](image)

**Violation 2**
- Visibility of System Status: Absence of loading spinner.

**Violation 3**
- Recognition rather than recall, consistency and standards:

The "Cancel" and "Submit" buttons were reversed, causing confusion for the user.
Violation 4

- Aesthetic and minimalist Design, consistency and standards

The registration and login page uses a color palette of green, white, black, and gray, while the rest of the components use a palette consisting of blue, red, white, black, and gray colors.

Violation 5

- Recognition rather than recall, consistency.

Buttons above the map that do not provide clear information to the user about their functionality.
4.4 *Think – Aloud Evaluation*

Think Aloud is the most valuable usability engineering method as it allows researchers to understand what users really think about their design. They observe the users' thoughts, decision-making process and problem-solving strategies in real time as they interact with the system [30]. Specifically, you listen to the thoughts, perceptions, misconceptions and decision-making processes as they interact with the system. From the above you can gain insight into the reasons behind user misconceptions or suggestions for improvement and make user-friendly redesigns. Nielsen's suggested that a sample of 3-5 participants is considered ideal.

The thinking aloud method is cost-effectiveness, and no special equipment is needed. Also, it is robust as you'll still get reasonably good findings, even from a poorly run study. Furthermore, it is flexible as you can use the method at any stage in the development lifecycle. Finally, it's an easy way to learn techniques that ensure direct exposure to customer feedback can change the mindset of even the most skeptical developers, designers, and executives [31].

**4.4.1 Participants**

Participants are representative users based on three personas we analyzed in chapter 2. One School Manager, three School Directors and one Department User participated in our evaluation.
4.4.2 The Environment
The Think-Aloud evaluation in this research is conducted in the according place of our participants. The Directors were evaluated in their office school, the department user and the school manager in their office that are in Patras municipality facilities.

4.4.3 Conducting the Session
We had already sent our participants an email with explanations about the evaluation procedure. First, we wanted to confirm that they didn’t have any questions or misunderstandings, so we explained again the whole procedure. Next, we provided them with the assignment that contained the tasks they had to complete as illustrated below.

Dear Participant, (School Director)
As part of our ongoing efforts to enhance the user experience on the School Plugin platform, we are conducting a Think-Aloud study and would like to invite you to participate. The purpose of this study is to gain valuable insights into the usability of our design and to identify potential areas for improvement.

Your participation involves completing a series of tasks within the School Plugin platform while verbally expressing your thoughts, opinions, and any challenges you may encounter. No specific skills or prior knowledge are required to take part; we simply ask for your willingness to explore and share your experiences.

Your input is crucial in helping us create a more user-friendly and effective platform for all users. We appreciate your time and contribution to the ongoing improvement of the School Plugin experience. We assure you that we comply with General Data Protection Regulation (GDPR) and respect your data.

THINGS TO KNOW:
Issues page lists of all the issues that a Director has created.
Add Issue page allow a Director to create a new issue.
Account Settings page contains general information from your school unit that you must complete.
Notification page allows you to create a reminder like fire extinguisher recharge.
Dashboard page allows you to observe multiple sensors data like fuel consumption.
Task 1: Register / Login
**Description 1:** You have received an email with the registration process; follow the link to complete your registration on the platform. Then log in to the account you created.

**Task 2:** Account settings

**Description 2:** Go to account settings, fill in, and save your school information.

**Task 3:** Create a new issue

**Description 3:** From the side menu select Add issue tab and create a new issue.

**Task 4:** Create a notification

**Description 4:** From the side menu select notifications tab and create a new notification. Take into account that you have the ability to create a recursive notification.

**Task 6:** View Monthly Notifications

**Description 5:** From the side menu select notifications tab and view your monthly notifications. You should find the notification on the calendar, and you should open it.

**Task 5:** Observe fuel level.

**Description 5:** From the side menu select Dashboard and check your fuel level.

**Task 6:** View your requests.

**Description 6:** From the side menu select Issues tab, find your request and open it. Then create a comment.

---

Dear Participant, (School Manager)

As part of our ongoing efforts to enhance the user experience on the School Plugin platform, we are conducting a Think-Aloud study and would like to invite you to participate. The purpose of this study is to gain valuable insights into the usability of our design and to identify potential areas for improvement.

Your participation involves completing a series of tasks within the School Plugin platform while verbally expressing your thoughts, opinions, and any challenges you may encounter. No
specific skills or prior knowledge are required to take part; we simply ask for your willingness to explore and share your experiences.

Your input is crucial in helping us create a more user-friendly and effective platform for all users. We appreciate your time and contribution to the ongoing improvement of the School Plugin experience. We assure you that we comply with General Data Protection Regulation (GDPR) and respect your data.

THINGS TO KNOW:

Overview Page: Overview of users, school units and issues count.

School Units Page: allows School managers to create School units.

Users Page: allows School managers to create users with the role Director.

Issues Page: lists of all the issues that a Director has created.

Search Page: search for a specific issue.

Heat Page: allows user to create an order.

Thank you for your participation.

----------------------------------------------------------------------

Dear Participant, (Department User/cleaning)

As part of our ongoing efforts to enhance the user experience on the School Plugin platform, we are conducting a Think-Aloud study and would like to invite you to participate. The purpose of this study is to gain valuable insights into the usability of our design and to identify potential areas for improvement.

Your participation involves completing a series of tasks within the School Plugin platform while verbally expressing your thoughts, opinions, and any challenges you may encounter. No specific skills or prior knowledge are required to take part; we simply ask for your willingness to explore and share your experiences.

Your input is crucial in helping us create a more user-friendly and effective platform for all users. We appreciate your time and contribution to the ongoing improvement of the School Plugin experience. We assure you that we comply with General Data Protection Regulation (GDPR) and respect your data.

THINGS TO KNOW:

Issues Page: View and administer issues of a selected category.
Search Page: search for a specific issue

Task 1: View School Issues.

Description 1: From the side menu select Issues tab, open a request and change its status to resolved.

Thank you for your participation.

Follow up questions:

- "Did you encounter any difficulty?"
- "Do you have any suggestions to facilitate the process?"

4.4.4 Think-Aloud Evaluation Results

In this section, the results of the Think-Aloud method are presented as described in section 7.2. All roles integrated into our application participated in the study, specifically 1 School Manager, 2 Department Users, and 3 Directors.

School Manager:

There is a need for the ability to print each request separately in PDF format. Additionally, the central administrator should be able to view requests that are in other departments.

Department User (Cleaning and Lighting):

There is a need for the ability to print each request separately in PDF format.

Director:

During the creation of a new request, our implementation was as follows: if the user has not selected the mandatory category and subcategory fields, the ability to submit the request was disabled, confusing the user.

The available reminders, as well as the ability to create a recurring notification, confuse the user.
4.5 Redesign based on evaluation results

Redesign based on Heuristics Evaluation Results:

Solution for Violation 1: Recognition rather recall.

Solution for Violation 2: Visibility of System Status. We use angular cdk overlay in combination with loading spinner to make our page unclickable until the successful loading of the corresponding content.
Solution for Violation 3: Recognition rather than recall, consistency and standards.

Solution for Violation 4: Aesthetic and minimalist Design, consistency and standards.

To solve the above problem, we created a color palette with the below combination of colors as illustrated in image X.

- **Green**: #00897b
- **Black**: #37474f
- **White**: #FFFFFF
- **Gray**: #333333

Our main color is green (#00897b). For background color, text and icons we use a combination of white (#ffffff) and black (#000000) to ensure good readability and a clean, modern look. We use Gray for borders, dividers, or subtle background elements. Green is the color of nature as it refers to the environment which can grab users’ attention easily. It is associated with meanings of growth, relaxation, harmony, freshness and fertility.
Solution for violation 5: Change in search filters, introduction of a status selection option, and removal of buttons above the map that do not provide clear information to the user about their functionality.

Redesign based on Think-Aloud Evaluation

School Manager: In images 47, 48, we present the redesigns based on the observations of the School Manager from 7.2.4.
Design and development of a management and control application for School Units

Figure 47: Create pdf

Figure 48: Issues from other departments
**Department User:** In image 49 we present the redesigns based on the observations of the Department User from 7.2.4.

*Figure 49: Create pdf*

**School Director:** In images 50, 51, we present the redesigns based on the observations of the School Director from 7.2.4.

*Figure 50: Add issue*
4.6 Ecological validity of user study

Ecological validity in the context of a user study refers to the extent to which the study accurately represents real-world scenarios and behaviors [26]. The findings and conclusions that arises from this study can be generalized. As a result, they can reflect the complexities and dynamics of the natural environment in which the target behavior typically occurs.

In user studies, especially in fields like human-computer interaction (HCI) or usability testing, ecological validity is crucial for ensuring that the results obtained are applicable and relevant in real-world settings.

Key considerations are Realistic Context, Naturalistic Behavior, Ecological Tasks and External Validity.

**Realistic Context:** The study took place in the physical environment of every user of the platform. For the Director, it took place in the school, and for both municipality users (school manager and department user) in municipalities facilities, using their own devices and facing typical distractions like interruptions for other tasks. This ensures that the user experience is authentic and representative of their typical interactions with the system.

**Naturalistic Behavior:** Participants were encouraged to behave as naturally as possible during the study.

Figure 51: Create alert
**Ecological Tasks:** The tasks assigned to participants were representative of the activities they would perform in the real world. This helps ensure that the skills and behaviors observed in the study are transferable to the intended application or use case. Tasks should mirror real-world scenarios, such as reporting facility maintenance issues or requesting community services.

**External Validity:** Ecological validity is closely related to the concept of external validity, which concerns the generalizability of study findings to other populations, settings, and times.

Consideration should be given to the generalizability of the findings beyond a specific school or municipality. If applicable, the platform should be designed to accommodate variations in school sizes, administrative structures, and municipal workflows to enhance external validity.

Taking the factor of ecological validity into consideration ensures that the user study reflects the functionality of the platform in a real environment and meets the needs of both the school director and the municipality users. This enhances the applicability of the findings to similar scenarios in other school units and municipalities.
Conclusions

5.1 Benefits of Using the Application

Following a User-Centered Design approach, it is evident that our primary goal, as described in Chapter 1.2, has been achieved. Both the functional and non-functional requirements that we have defined in Chapter 2 have been fully satisfied. Subsequently, the platform was evaluated with real users in real conditions, and finally, we proceeded with redesign based on their feedback following their interaction.

Within the scope of this specific thesis, the role and significance of technology in improving and organizing processes, such as recording the issues of school units, are highlighted. Specifically, the development of a platform has been achieved with the aim of enhancing and organizing processes between schools and municipalities. Below are the advantages for all involved parties.

Municipalities:

- Improved organization and management of school unit issues
- Categorization of school unit issues
- Centralized management of requests and better task scheduling
- Monitoring the status of issues and easy search
- Mapping of all school units
- Statistics
- Open data - Dissemination of information to third-party applications

School Units:
• Immediate reporting of issues. Classification of requests into open - in progress - completed.
• Request tracking system.
• Alarm systems.
• Historicity of issues

Overall, the design, development and evaluation of this online application have demonstrated its potential to meet the needs of their users. The foundation for a promising platform that can bridge the gap of communication between schools and municipalities has been set. In summary, our platform contributes to the improved management and control of school facilities, aligning with the new legislative framework for school buildings. It supports integration with Internet of Things (IoT) platforms and Building Management Systems (BMS), along with procurement systems for competitive bidding. Additionally, it has the capability to provide open data.

5.2 Limitations

A primary limitation during the development of our platform was the difficulty in finding a satisfactory sample of representative users. Nevertheless, addressing three Directors, one from each school level, and two users from our Municipality assisted in evaluating our platform and making significant improvements. Also, we haven’t connected our platform with real sensors, but we use dummy data for demo purposes.

5.3 Future Prospects

As for future improvements and expansions of the system, we have included the implementation of statistical information in our timeline, such as the number of resolved issues. Additionally, the assessment of requests by Directors has been scheduled for implementation when they reach the status: completed. It is also important for stability, flexibility, and scalability to host our platform in a major cloud provider such as AWS or Microsoft, taking advantage of the benefits these platforms offer. Furthermore, our next goal is to upgrade the Angular and node.js versions to the current versions to be able to use the
latest features. Additionally, enabling users with the role of director to set thresholds/values for the fuel level of the boiler is a significant feature that we intend to implement. Furthermore, a second mode, color palette that consisted of blue, black, gray and white colors will be a valuable next step as illustrated in images 48, 49, 50.

Last but not least, a fundamental and necessary goal is the application of evaluation methods such as Think Aloud Evaluation with more users, aiming to gain valuable insights and improve both platforms and the user experience.

Figure 52: New colour palette
Design and development of a management and control application for School Units

Figure 53: New colour palette

Figure 54: New colour palette
Design and development of a management and control application for School Units
Bibliography - References

[9] https://myschool.sch.gr/
[12] https://www.figma.com/file/lj3ZPza9Y9m1C6oWr61qNf/directorUI1?type=design&node-id=0-1&mode=design&t=KF3msLw8aGA2vI5m-0
[13] https://www.figma.com/file/0gdE6kPLTj8sIJB8YuqZ/schoolAdminUI2?type=design&mode=design&t=KF3msLw8aGA2vI5m-0
[14] https://smartdatamodels.org/index.php/about/
[18] https://update.angular.io/
[19] https://material.angular.io/
Design and development of a management and control application for School Units

[21] https://angular.io/guide/what-is-angular
[23] https://expressjs.com/


[31] https://www.nngroup.com/articles/thinking-aloud-the-1-usability-tool/
[32] https://angular.io/guide/testing
[33] https://christianlydemann.com/complete-guide-to-angular-testing/
[34] https://christianlydemann.com/complete-guide-to-angular-testing/
[36] https://medium.com/williambastidasblog/angular-testing-made-easy-understanding-the-testing-pyramid-and-focusing-on-unit-testing-2023-b4feb8ce5d8e
[37] https://testing-angular.com/testing-principles/#levels-of-testing
[39] https://www.interaction-design.org/literature/topics/heuristic-evaluation
[40] https://www.interaction-design.org/literature/topics/human-centered-design
[41] https://www.digital-mosaic.co.uk/blog/10-usability-heuristics-for-user-interface-design/
[42] https://www.userfocus.co.uk/resources/guidelines_translated.html
[44] https://www.mongodb.com/mean-stack
Design and development of a management and control application for School Units
Design and development of a management and control application for School Units